aws

用户指南

AWS Systems Manager



Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Systems Manager: 用户指南

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon 的商标和商业外观不得用于任何非 Amazon 的商品或服务，也不得以任何可能引起客户混淆、贬低或诋毁 Amazon 的方式使用。所有非 Amazon 拥有的其他商标均为各自所有者的财产，这些所有者可能附属于 Amazon、与 Amazon 有关联或由 Amazon 赞助，也可能不是如此。

Table of Contents

什么是 AWS Systems Manager？ 1
Systems Manager 如何使我的操作受益？ 1
谁应该使用 Systems Manager？ 2
Systems Manager 有哪些主要功能？ 2
支持 AWS 区域 3
访问 Systems Manager 4
Systems Manager 服务名称历史 5
支持的操作系统和计算机类型 5
Systems Manager 支持的操作系统 5
混合和多云环境中支持的计算机类型 12
什么是统一控制台？ 12
为 AWS Systems Manager 设置托管节点 15
使用 Systems Manager 管理 EC2 实例 15
配置 Systems Manager 所需的实例权限 16
使用适用于 Systems Manager 的 VPC 端点提高 EC2 实例的安全性 25
使用 Systems Manager 管理混合和多云环境中的节点 29
在混合和多云环境中创建 Systems Manager 所需的 IAM 服务角色 31
创建混合激活以将节点注册到 Systems Manager 39
在混合 Linux 节点上安装 SSM Agent 45
在混合 Windows Server 节点上安装 SSM Agent 51
使用 Systems Manager 管理边缘设备 55
为您的边缘设备创建 IAM 服务角色 56
配置 AWS IoT Greengrass 的边缘设备 62
更新 AWS IoT Greengrass 令牌交换角色并在边缘设备上安装 SSM Agent 62
为 Systems Manager 创建 AWS Organizations 委派管理员 62
对 Change Manager 使用委派管理员 63
对 Explorer 使用委派管理员 63
对 OpsCenter 使用委派管理员 64
对 Quick Setup 使用委派管理员 64
常规设置 64
注册 AWS 账户 64
创建具有管理访问权限的用户 65
设置 AWS Systems Manager 67
设置 Systems Manager 控制台访问 67

Systems Manager 载入策略 67
AWS Systems Manager 控制台操作员策略 73
AWS Systems Manager 控制台操作员只读策略 77
为组织设置 Systems Manager 统一控制台 79
先决条件 80
统一控制台资源 80
设置统一控制台 82
为单个账户和区域设置 Systems Manager 统一控制台 84
统一控制台资源 84
设置统一控制台 85
禁用 Systems Manager 统一控制台 87
执行节点管理任务 88
查看节点见解 88
添加或移除小组件 91
重新排列小组件 91
浏览节点 92
使用控制台筛选条件探索节点 93
在 Amazon Q 中使用文本提示探索节点 97
查看单个节点详细信息并对节点执行操作 104
下载或导出托管节点报告 105
管理节点报告内容和外观 109
使用 Systems Manager 进行即时节点访问 110
使用 Systems Manager 设置即时访问 111
启动即时节点访问会话 156
管理即时访问请求 157
从Session Manager移至即时节点访问 159
使用 Systems Manager 禁止即时访问 163
常见问题解答 163
诊断并修复 164
诊断和修复失败部署 165
诊断和修复偏差配置 166
诊断和修复非托管 Amazon EC2 实例 167
运行手册操作的修复影响类型 173
查看修正的执行历史记录详细信息 175
调整 Systems Manager 设置 176
账户设置设定 176

组织设置设定 177
功能配置 177
诊断并修复设置 180
使用 Amazon S3 存储桶和 Systems Manager 的存储桶策略 182
使用 AWS Systems Manager 工具 191
节点工具 191
变更管理工具 193
应用程序工具 194
操作工具 194
使用 Systems Manager 执行节点任务 195
先决条件 195
使用已预安装 SSM Agent 的 AMI 启动实例 195
使用 Systems Manager 连接到托管实例 196
清除您的实例 197
节点工具 197
合规 198
Distributor 215
Fleet Manager 263
混合激活 341
Inventory（清单） 342
Patch Manager 432
Run Command 702
Session Manager 763
State Manager 897
更改管理工具 984
自动化 985
Change Calendar 1424
Change Manager 1441
文档 1517
Maintenance Windows 1648
Quick Setup 1769
应用程序工具 194
AWS AppConfig 1812
Application Manager 1812
Parameter Store 1854
操作工具 1999

Incident Manager 1999
Explorer 1999
OpsCenter 2035
CloudWatch 控制面板 2112
使用 SSM Agent 2114
了解有关 SSM Agent 的技术细节 2114
SSM Agent 版本 3.2.x.x 凭证行为 2115
SSM Agent 凭证优先级 2115
配置 SSM Agent 以与美国联邦信息处理标准（FIPS）结合使用 2117
关于本地 ssm-user 账户 2118
SSM Agent 和 Instance Metadata Service (IMDS) 2119
保持 SSM Agent 的最新状态 2119
确保 SSM Agent 安装目录未被修改、移动或删除 2120
按 AWS 区域 的 SSM Agent 滚动更新 2120
SSM Agent 与 AWS 托管 S3 存储桶进行通信 2120
查找预装了 SSM Agent 的 AMIs 2127
验证 SSM Agent 的状态 2128
在适用于 Linux 的 EC2 实例上使用 SSM Agent 2132
验证 SSM Agent 签名 2132
在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent 2146
配置 SSM Agent 以在 Linux 节点上使用代理 2200
在适用于 macOS 的 EC2 实例上使用 SSM Agent 2205
在适用于 macOS 的 EC2 实例上手动安装和卸载 SSM Agent 2206
在适用于 Windows Server 的 EC2 实例上使用 SSM Agent 2207
在适用于 Windows Server 的 EC2 实例上手动安装和卸载 SSM Agent 2208
配置 SSM Agent以使用 Windows Server 实例的代理 2211
正在检查 SSM Agent 状态并启动代理 2214
正在检查 SSM Agent 版本号 2216
查看 SSM Agent 日志 2221
允许 SSM Agent 调试日志记录 2222
通过 SSM Agent 限制对根级别命令的访问 2224
自动更新到 SSM Agent 2225
自动更新 SSM Agent 2226
订阅 SSM Agent 通知 2227
排查 SSM Agent 问题 2228
SSM Agent 已过时 2228

使用 SSM Agent 日志文件进行故障排除 2228
代理日志文件不会滚动 (Windows) 2229
无法连接到 SSM 端点 2230
验证您的 VPC 配置 2231
验证 VPC DNS 相关属性 2232
验证端点安全组的入口规则 2232
使用 ssm-cli 排查托管节点可用性的问题 2233
安全性 2234
数据保护 2234
数据加密 2235
互联网络流量隐私 2238
身份和访问管理 2238
受众 2239
使用身份进行身份验证 2239
使用策略管理访问 2241
AWS Systems Manager 如何与 IAM 协同工作 2245
基于身份的策略示例 2254
AWS 托管式策略 2265
故障排除 2311
使用服务相关角色 2312
清单和 Explorer 数据角色 2314
OpsCenter 和 Explorer 账户发现角色 2316
OpsData 和 OpsItems 创建角色 2318
Operational Insights (运营洞察) 创建角色 2322
Quick Setup 部署运行状况检查角色 2325
导出 OpsData 服务角色 2327
即时节点访问服务角色 2329
即时节点访问请求通知服务角色 2332
日志记录和监控 2336
合规性验证 2338
故障恢复能力 2338
基础设施安全性 2339
配置和漏洞分析 2339
安全最佳实践 2339
Systems Manager 预防性安全最佳实践 2340
Systems Manager 监控和审计最佳实践 2342

代码示例 2345
基本功能 2352
Hello Systems Manager 2356
了解基础知识 2360
操作 2411
日志记录和监控 2739
监控工具 2740
将节点日志发送到统一的 CloudWatch Logs（CloudWatch 代理） 2740
将 Windows Server 节点日志收集迁移到 CloudWatch 代理 2741
将 CloudWatch 代理配置设置存储到 Parameter Store 中 2751
回滚到使用 SSM Agent 进行日志收集 2751
将 SSM Agent 日志发送到 CloudWatch Logs 2754
监控您的变更请求事件 2757
监控您的自动化 2759
Automation 指标 2760
使用 Amazon CloudWatch 监控 Run Command 指标 2761
Systems Manager Run Command 指标和维度 2761
使用 AWS CloudTrail 记录 AWS Systems Manager API 调用 2762
CloudTrail 中的 Systems Manager 数据事件 2763
CloudTrail 中的 Systems Manager 管理事件 2764
Systems Manager 事件示例 2765
使用 CloudWatch Logs 记录自动化操作输出 2770
为 Run Command 配置 Amazon CloudWatch Logs 2774
在发送命令时指定 CloudWatch Logs 2775
在 CloudWatch Logs 中查看命令输出 2776
使用 Amazon EventBridge 监控 2776
为 Systems Manager 事件配置 EventBridge 2778
适用于 Systems Manager 的 Amazon EventBridge 事件示例 2781
示例场景：Amazon EventBridge 规则中的 Systems Manager 目标 2797
使用 Amazon SNS 通知监控 Systems Manager 状态更改 2798
为 AWS Systems Manager 配置 Amazon SNS 通知 2799
适用于 AWS Systems Manager 的 Amazon SNS 通知示例 2808
使用 Run Command 发送返回状态通知的命令 2810
使用维护时段发送返回状态通知的命令 2813
产品和服务集成 2818
与 AWS 服务 集成 2818

计算 2818
物联网 (IoT) 2820
存储 2821
开发工具 2822
安全、身份和合规性 2822
加密和 PKI 2825
管理和治理 2825
联网和内容分发 2829
Analytics 2830
应用程序集成 2831
AWS Management Console 2831
从 Amazon S3 运行脚本 2832
通过 Parameter Store 参数引用 AWS Secrets Manager 密钥 2836
Parameter Store SecureString 参数的 AWS KMS 加密 2842
在 Amazon Elastic Kubernetes Service 中使用 AWS Secrets Manager 密钥 2854
在 AWS Lambda 函数中使用 Parameter Store 参数 2872
与其他产品和服务的集成 2892
从 GitHub 运行脚本 2896
将 Chef InSpec 配置文件与 Systems Manager Compliance 结合使用 2903
集成 ServiceNow 2908
AWS Systems Manager 引用 2909
使用 AWS SDK 2910
用于修补操作的 Amazon S3 存储桶 2911
包含用于修补操作的 SSM 命令文档的存储桶（Linux 和 Windows Server） 2911
包含用于修补操作的 SSM 命令文档的存储桶 （macOS） 2914
包含 AWS 托管补丁基准快照的存储桶 2917
EventBridge 事件模式和 Systems Manager 类型 2920
事件类型：自动化 2921
事件类型：Change Calendar 2922
事件类型：Change Manager 2922
事件类型：配置合规性 2922
事件类型：库存 2923
事件类型：维护窗口 2923
事件类型：OpsCenter 2925
事件类型：Parameter Store 2926
事件类型：Run Command 2927

事件类型：State Manager 2927
Cron 和 Rate 表达式 2928
有关 Cron 和 Rate 表达式的一般信息 2928
适用于关联的 Cron 和 Rate 表达式 2933
适用于维护时段的 Cron 和 Rate 表达式 2936
ec2messages、ssmmessages 和其他 API 操作 2938
与代理相关的 API 操作（ssmmessages 和 ec2messages 端点） 2939
与 ssm:* 命名空间实例相关的 API 操作 2940
ssm: * 命名空间其他 API 操作 2941
Systems Manager 的日期和时间字符串格式 2942
Systems Manager 的格式化日期和时间字符串 2942
为 Systems Manager 创建自定义日期和时间字符串 2943
应用场景和最佳实践 2945
删除 Systems Manager 资源和构件 2948
在 State Manager 和 Maintenance Windows 之间选择 2951
State Manager 和 Maintenance Windows：关键使用案例 2951
相关信息 2957
文档历史记录 2959
2018 年 6 月之前的更新 3101
文档惯例 3115

什么是 AWS Systems Manager？

AWS Systems Manager 可帮助您在 AWS、本地和多云环境中大规模集中查看、管理和操作节点。随着统一控制台体验的推出，Systems Manager 整合了各种工具来帮助您跨 AWS 账户和 AWS 区域完成常见节点任务。

要使用 Systems Manager，节点必须处于托管状态，这意味着 SSM Agent 安装在计算机上并且代理可以与 Systems Manager 服务进行通信。为了帮助确定节点未报告为托管的原因，Systems Manager 提供了一个一键式代理问题诊断和修复运行手册，您可以将其配置为根据您定义的计划自动运行。此功能可帮助识别节点无法连接到 Systems Manager 的原因，包括网络配置错误。此功能还提供推荐的运行手册，用于修复网络问题以及其他导致无法将节点配置为托管节点的问题。

统一控制台体验还包括一个提供您的节点高级概览的控制面板。您可以深入了解更具体的节点见解，例如哪些节点正在运行过时的操作系统 (OS) 软件。您还可以根据实例元数据（如操作系统和操作系统版本、AWS 区域、AWS 账户 和 SSM Agent 版本）使用筛选条件获取精细视图。这些筛选条件可帮助您检索整个组织内特定账户级别或应用程序级别的相关信息。

主题

- Systems Manager 如何使我的操作受益？
- 谁应该使用 Systems Manager？
- Systems Manager 有哪些主要功能？
- 支持 AWS 区域
- 访问 Systems Manager
- Systems Manager 服务名称历史
- 支持的操作系统和计算机类型
- 什么是统一控制台？

Systems Manager 如何使我的操作受益？

Systems Manager 具有以下优势：

- 增强整个基础设施的可见性

  Systems Manager 提供您的组织的账户和区域的节点的集中视图。快速访问实例信息，例如 ID、名称、操作系统详细信息以及已安装的代理。使用 Amazon Q 开发者版通过自然语言查询实例元数据，以帮助您更快地识别问题并采取行动。

- 通过自动化提高运营效率

  自动执行常见的操作任务并减少维护您的系统所需的时间和工作。Systems Manager 无需登录您的服务器即可大规模、安全地远程管理您的节点。您不再需要使用堡垒主机、SSH 或者远程 PowerShell。Systems Manager 还提供了一种简单的方法来跨节点组自动执行常见的管理任务，例如注册表编辑、用户管理以及软件和补丁安装。
- 在任何环境中简化大规模节点管理

  Systems Manager 可帮助您跨 AWS、本地和多云环境管理节点。安排自动诊断来识别 SSM Agent 问题并通过一键式运行手册进行修复。在将您的节点配置为托管节点后，您可以执行应用安全补丁、启动记录的会话以及远程运行命令等关键操作任务。

谁应该使用 Systems Manager？

Systems Manager 可供 IT 运营经理和操作员、DevOps 工程师、安全和合规经理以及 IT 主管和 CIO 使用。从广义上讲，Systems Manager 适用于以下情况：

- 希望大规模提高其节点的管理和安全性的组织。
- 希望在管理基础设施时提高可见性和运营灵活性的组织。
- 希望大规模提高运营效率的组织。

Systems Manager 有哪些主要功能？

Systems Manager 的主要功能在统一控制台和 Systems Manager 提供的各项工具之间共享，可帮助您大规模管理节点。

统一控制台

统一控制台提供了查看和管理节点的集中式体验。此控制台利用多项 Systems Manager 工具以及更多工具为您提供以下内容：

- 您的节点的集中视图
- 详细的节点见解
- 自动诊断和修复常见的节点问题

有关统一控制台的更多信息，请参阅什么是统一控制台？。

工具

工具由 Systems Manager 的各项功能组成，例如 Run Command、Session Manager、Automation 和 Parameter Store。您利用 Systems Manager 工具执行以下操作：

- 大规模修补节点
- 无需打开入站端口即可安全地连接到节点
- 在节点上远程运行命令
- 安全地存储应用程序引用的数据
- 自动执行常见的系统管理任务

有关 Systems Manager 工具的更多信息，请参阅使用 AWS Systems Manager 工具。

支持 AWS 区域

有关支持 Systems Manager 工具的 AWS 区域列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints。

Systems Manager 统一控制台于 2024 年 11 月 21 日发布，可在以下 AWS 区域使用：

- US East (N. Virginia) Region
- 美国东部（俄亥俄州）区域
- 美国西部（北加利福尼亚）区域
- 美国西部（俄勒冈州）区域
- 加拿大（中部）区域
- 南美洲（圣保罗）区域
- 亚太地区（孟买）区域
- Asia Pacific（Tokyo）Region
- 亚太地区（首尔）区域
- 亚太地区（新加坡）区域
- 亚太地区（悉尼）区域
- 欧洲地区（法兰克福）区域
- 欧洲地区（斯德哥尔摩）区域
- 欧洲地区（爱尔兰）区域

- 欧洲地区（伦敦）区域
- 欧洲地区（巴黎）区域

访问 Systems Manager

您可以通过以下任何方式使用 Systems Manager：

Systems Manager 控制台

Systems Manager 控制台是基于浏览器的界面，用于访问和使用 Systems Manager。

AWS IoT Greengrass V2 管理控制台

您可以在 Greengrass 控制台中查看和管理为 AWS IoT Greengrass 配置的边缘设备。

AWS 命令行工具

可以使用 AWS 命令行工具，在系统的命令行中发出命令来执行 Systems Manager 和其他 AWS 任务。这些工具在 Linux，macOS, 和 Windows 上受支持。使用 AWS Command Line Interface (AWS CLI) 可能比控制台更快、更方便。如果要构建执行 AWS 任务的脚本，命令行工具也会十分有用。

AWS 提供两组命令行工具：AWS Command Line Interface 和 AWS Tools for Windows PowerShell。有关安装和使用 AWS CLI 的更多信息，请参阅 AWS Command Line Interface 用户指南。有关安装和使用 Tools for Windows PowerShell 的信息，请参阅 AWS Tools for PowerShell 用户指南。

> **Note**
>
> 在 Windows Server 实例上，需要使用 Windows PowerShell 3.0 或更高版本以运行某些 SSM 文档（例如，旧式 `AWS-ApplyPatchBaseline` 文档）。确认 Windows Server 实例运行 Windows Management Framework 3.0 或更高版本。该框架包括 Windows PowerShell。

AWS 开发工具包

AWS 提供了开发工具包 (SDK)，其中包含各种编程语言和平台的库和示例代码，例如，Java、Python、Ruby、.NET、iOS 和 Android ，以及其它等。开发工具包（SDK）提供了便捷的方式，以授权对 Systems Manager 的编程访问。有关 AWS 开发工具包的信息（包括如何下载及安装），请参阅适用于 Amazon Web Services 的工具。

Systems Manager 服务名称历史

AWS Systems Manager (Systems Manager) 以前称为“Amazon Simple Systems Manager (SSM)”和“Amazon EC2 Systems Manager (SSM)”。服务“SSM”的原始缩写名称仍反映在各个 AWS 资源中，包括其他一些服务控制台。一些示例：

- Systems Manager Agent：SSM Agent
- Systems Manager 参数：SSM 参数
- Systems Manager 服务端点：`ssm.region.amazonaws.com`
- AWS CloudFormation 资源类型：`AWS::SSM::Document`
- AWS Config 规则标识符：`EC2_INSTANCE_MANAGED_BY_SSM`
- AWS Command Line Interface (AWS CLI) 命令：`aws ssm describe-patch-baselines`
- AWS Identity and Access Management (IAM) 托管策略名称：`AmazonSSMReadOnlyAccess`
- Systems Manager 资源 ARN：`arn:aws:ssm:region:account-id:patchbaseline/pb-07d8884178EXAMPLE`

支持的操作系统和计算机类型

在使用 Systems Manager 之前，请验证您的操作系统（OS）、操作系统版本和计算机类型是否支持作为托管式节点。

主题

- Systems Manager 支持的操作系统
- 混合和多云环境中支持的计算机类型

Systems Manager 支持的操作系统

以下各节列出了 Systems Manager 支持的操作系统和操作系统版本。

> **Note**
>
> 如果您计划使用 Systems Manager 管理和配置 AWS IoT Greengrass 核心设备，这些设备必须满足 AWS IoT Greengrass 的要求。有关更多信息，请参阅 AWS IoT Greengrass 开发人员指南中的设置 AWS IoT Greengrass Version 2 核心设备。

如果您计划管理及配置 AWS IoT 和非 AWS 边缘设备，这些设备必须满足此处列出的要求，并配置为 Systems Manager 的本地托管式节点。有关更多信息，请参阅 使用 Systems Manager 管理边缘设备。

Important

Patch Manager是 Systems Manager 中的一项工具，可能不支持本主题中列出的所有操作系统（OS）版本。有关受 Patch Manager 支持的 OS 版本的列表，请参阅 Patch Manager 先决条件。

操作系统类型

- Linux
- macOS（仅 Amazon EC2 实例）
- Raspberry Pi OS（原 Raspbian）
- Windows Server

选择一个 OS 平台以查看支持的主要版本和次要版本。

Linux

AlmaLinux

版本	x86	x86_64	ARM64
8.3–8.10		✓	✓
9.x		✓	✓

Amazon Linux 1

版本	x86	x86_64	ARM64
2012.03-2018.03	✓	✓	

> **Note**
>
> 从版本 2015.03 开始，Amazon Linux 1 以 x86_64 版本发布。
> 正如 AWS News Blog 上 Update on Amazon Linux AMI end-of-life 一文宣布的那样，Amazon Linux 1 于 2020 年 12 月 31 日结束了标准支持，于 2023 年 12 月 31 日结束了生命周期。AWS 不再为该操作系统提供 Amazon Machine Images（AMIs）。不过，AWS Systems Manager 会继续为现有的 Amazon Linux 1 实例提供支持。

Amazon Linux 2

版本	x86	x86_64	ARM64
2.0 和所有更高版本		✓	✓

Amazon Linux 2023

版本	x86	x86_64	ARM64
2023.0.20230315.0 和所有更高版本		✓	✓

Bottlerocket

版本	x86_64	ARM64
1.0.0 和所有更高版本	✓	✓

CentOS

版本	x86	x86_64	ARM64
6.x[1]	✓	✓	
7.1 及更高的 7.x 版本		✓	✓
8.x		✓	✓

[1] 要使用这些版本，您必须使用 3.0.x 版本的 SSM Agent。我们建议您使用最新可用 3.0.x 版本的 SSM Agent。不支持更高的 SSM Agent 版本（3.1 或以上）。

CentOS Stream

版本	x86	x86_64	ARM64
8		✓	✓
9		✓	✓

Debian 服务器

版本	x86	x86_64	ARM64
Jessie (8)		✓	
Stretch (9)		✓	✓
Buster(10)		✓	✓
Bullseye (11)		✓	✓
Bookworm (12)		✓	✓

Oracle Linux

版本	x86	x86_64	ARM64
7.5-7.8		✓	
8.x		✓	
9.x		✓	

Red Hat Enterprise Linux (RHEL)

版本	x86	x86_64	ARM64
6.x[1]	✓	✓	
7.0-7.5		✓	
7.6–8.x		✓	✓
9.x		✓	✓
10.x		✓	✓

[1] 要使用这些版本，您必须使用 3.0.x 版本的 SSM Agent。我们建议您使用最新可用 3.0.x 版本的 SSM Agent。不支持更高的 SSM Agent 版本（3.1 或以上）。

Rocky Linux

版本	x86	x86_64	ARM64
8.x		✓	✓
9.x		✓	✓

SUSE Linux Enterprise Server (SLES)

版本	x86	x86_64	ARM64
12 及更高的 12.x 版本		✓	
15x		✓	✓

Ubuntu Server

版本	x86	x86_64	ARM64
12.04 LTS 和 14.04 LTS	✓	✓	

版本	x86	x86_64	ARM64
16.04 LTS 和 18.04 LTS		✓	✓
20.04 LTS 和 20.10 LTS		✓	✓
22.04 LTS		✓	✓
23.04、23.10		✓	✓
24.04 LTS		✓	✓
24.10		✓	✓
25.04		✓	✓

macOS（仅 Amazon EC2 实例）

版本	x86	x86_64	Mac with Apple silicon
10.14.x (Mojave)		✓	
10.15.x (Catalina)		✓	
11.x (Big Sur)		✓	✓
12.x (Monterey)		✓	✓
13.x (Ventura)		✓	✓
14.x (Sonoma)		✓	✓
15.x (Sequoia)		✓	✓

> **Note**
>
> 并非所有 AWS 区域 都支持 macOS。有关对适用于 macOS 的 Amazon EC2 支持的一般信息，请参阅《Amazon EC2 用户指南》中的 Amazon EC2 Mac 实例。

Raspberry Pi OS（原 Raspbian）

版本	ARM32
8 (Jessie)	✓
9 (Stretch)	✓

更多信息

- 使用 AWS Systems Manager 管理树莓派设备

Windows Server

SSM Agent 需要 Windows PowerShell 3.0 或更高版本在 Windows Server 实例上运行某些 AWS Systems Manager 文档（SSM 文档）（例如，旧式 `AWS-ApplyPatchBaseline` 文档)。确认 Windows Server 实例运行 Windows Management Framework 3.0 或更高版本。此框架包括 Windows PowerShell。有关更多信息，请参阅 Windows Management Framework 3.0。

版本	x86	x86_64	ARM64
2008[1]	✓	✓	
2008 R2[1]		✓	
2012 和 2012 R2[2]		✓	
2016		✓	
2019		✓	
2022 年		✓	

版本	x86	x86_64	ARM64
2025 年		✓	

[1]Windows Server 2008 和 2008 R2 支持：截至 2020 年 1 月 14 日，Windows Server 2008 不再受支持以获取来自微软的功能或安全更新。原有 Amazon Machine Images (AMIs) for Windows Server 2008 和 2008 R2 仍包含预安装的 SSM Agent 2 版，但 Systems Manager 不再正式支持 2008 版，并且不再针对更新这些 Windows Server 版本更新代理。此外，SSM Agent 版本 3 可能不兼容 Windows Server 2008 和 2008 R2 上的所有操作。适用于 Windows Server 2008 版的最后正式支持的 SSM Agent 版本为 2.3.1644.0。

[2] Windows Server 2012 和 2012 R2 支持：Windows Server 2012 和 2012 R2 支持已于 2023 年 10 月 10 日结束。要将 SSM Agent 与这些版本一起使用，建议使用 Microsoft 的扩展安全更新（ESU）。有关更多信息，请参阅 Microsoft 网站上的 Windows Server 2012 和 2012 R2 即将终止支持。

混合和多云环境中支持的计算机类型

Systems Manager 支持多种计算机类型作为托管式节点。托管式节点是配置用于与 Systems Manager 一起使用的任何计算机。

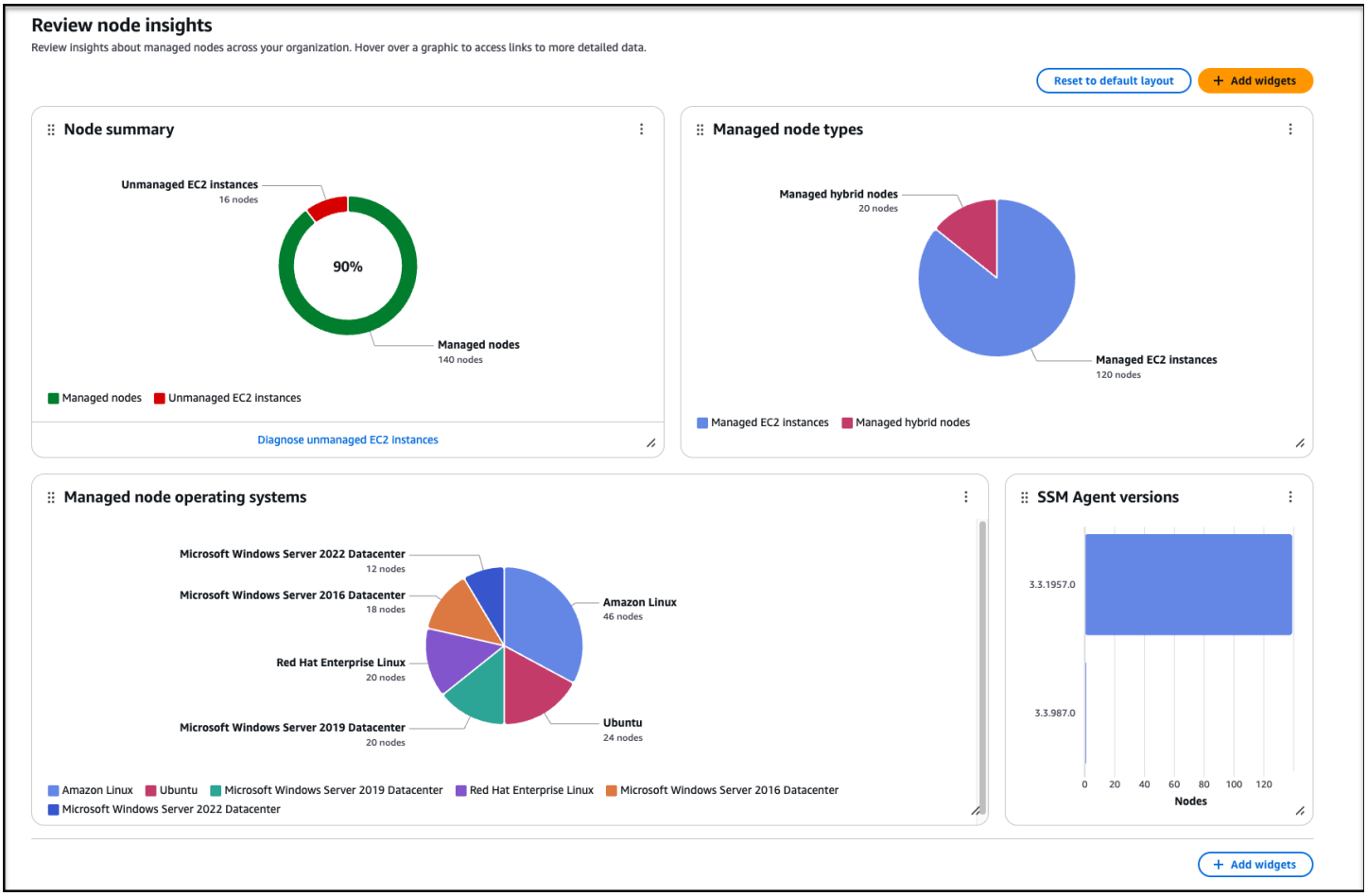
本用户指南使用混合和多云一词来指包含以下计算机类型任意组合的环境：

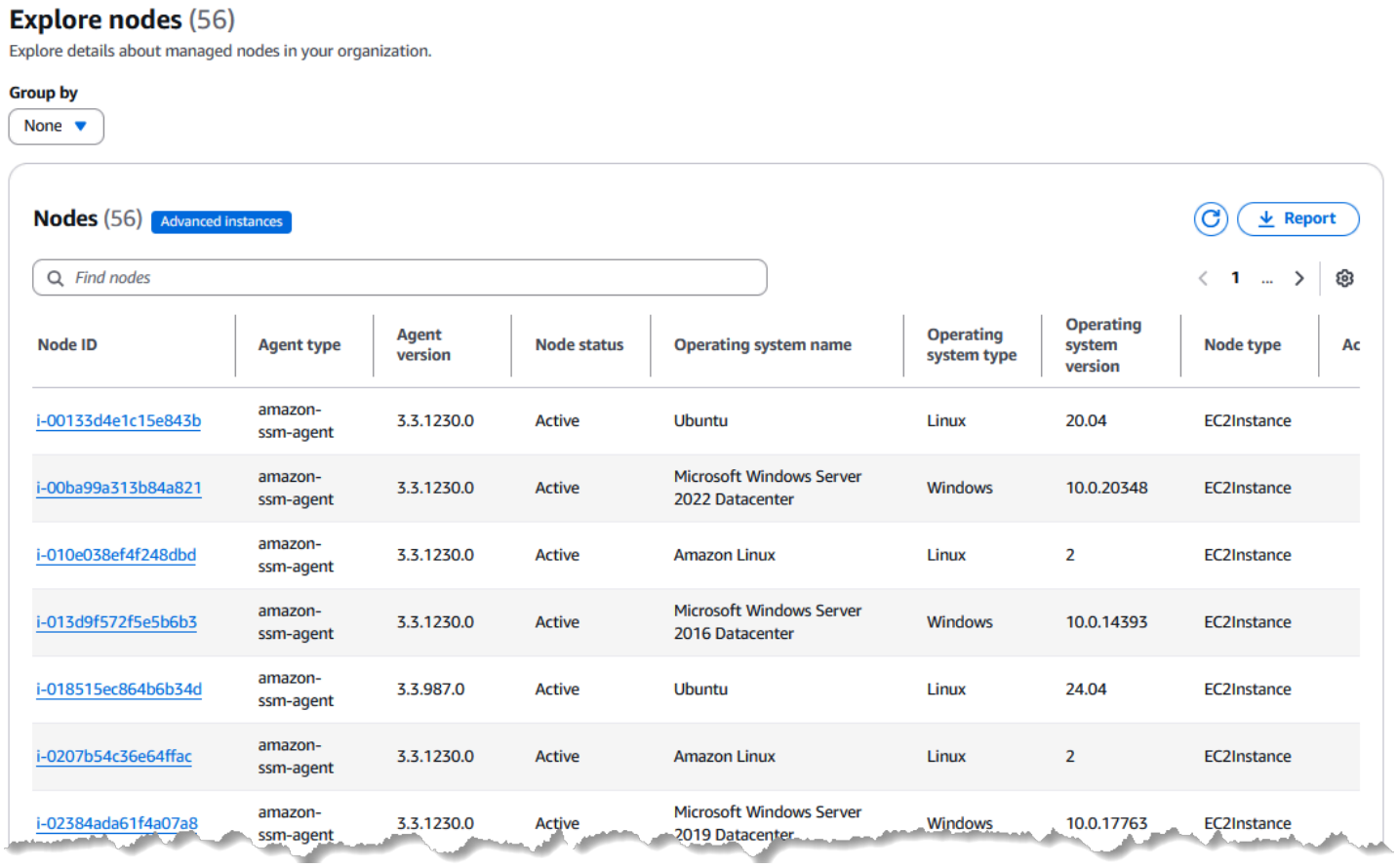
- Amazon Elastic Compute Cloud（Amazon EC2）实例
- 您本地的服务器（本地服务器）
- AWS IoT Greengrass 核心设备
- AWS IoT 和非 AWS 边缘设备
- 虚拟机，包括其他云环境中的虚拟机

有关对混合和多云环境的 AWS 支持的信息，请参阅适用于混合和多云的 AWS 解决方案。

什么是统一控制台？

Systems Manager 统一控制台是一种整合体验，结合了各项工具，可帮助您在 AWS Organizations 组织的多个 AWS 账户和 AWS 区域或单个账户和区域中完成常见的节点任务。节点可以是 EC2 实例、混合服务器或多云环境中运行的服务器。在统一控制台中，您可以了解有关节点的详细见解。您可以为节点生成报告，诊断和修复导致节点无法报告由 Systems Manager 管理的常见问题，例如连接问题。除了有关节点的摘要，您还可以查看有关节点的特定详细信息，例如软件清单和补丁状态。


Review node insights
Review insights about managed nodes across your organization. Hover over a graphic to access links to more detailed data.
Reset to default layout
Add widgets
Node summary
Unmanaged EC2 instances
16 nodes
90%
Managed nodes
140 nodes
Managed nodes
Unmanaged EC2 instances
Diagnose unmanaged EC2 instances
Managed node types
Managed hybrid nodes
20 nodes
Managed EC2 instances
120 nodes
Managed EC2 instances
Managed hybrid nodes
Managed node operating systems
Microsoft Windows Server 2022 Datacenter
12 nodes
Microsoft Windows Server 2016 Datacenter
18 nodes
Red Hat Enterprise Linux
20 nodes
Microsoft Windows Server 2019 Datacenter
20 nodes
Amazon Linux
46 nodes
Ubuntu
24 nodes
Amazon Linux
Ubuntu
Microsoft Windows Server 2019 Datacenter
Red Hat Enterprise Linux
Microsoft Windows Server 2016 Datacenter
Microsoft Windows Server 2022 Datacenter
SSM Agent versions
3.3.1957.0
3.3.987.0
0
20
40
60
80
100
120
Nodes
Add widgets



无论节点位于组织中的多个账户和区域，还是单个账户和区域，都建议使用统一控制台。Systems Manager 将持续向统一控制台添加可完成更多节点任务的功能。要了解现在可以使用统一控制台执行的节点任务，请参阅使用 AWS Systems Manager 执行节点管理任务。

继续阅读下一个主题，将节点设置为由 Systems Manager 管理。要了解有关设置 Systems Manager 节点的更多信息，请参阅为 AWS Systems Manager 设置托管节点。设置好节点后，您可以设置 Systems Manager 和统一控制台。要了解有关设置 Systems Manager 的更多信息，请参阅设置 AWS Systems Manager。

为 AWS Systems Manager 设置托管节点

完成本节中的任务来设置和配置使用 AWS Systems Manager 工具的角色、用户账户、权限和初始资源。本节中介绍的任务通常由 AWS 账户 和系统管理员执行。完成这些步骤后，贵企业中的用户便可以使用 Systems Manager 来配置、管理和访问您的托管式节点。托管节点是指在混合和多云环境中配置与 Systems Manager 一起使用的任何计算机。

> **Note**
>
> 如果您计划在混合和多云环境中使用 Amazon EC2 实例和您的计算资源，请按照 使用 Systems Manager 管理 EC2 实例 中的步骤操作。该主题介绍为 EC2 实例和非 EC2 计算机完成 Systems Manager 设置的最佳顺序步骤。

如果您已使用其他 AWS 服务，则表示您已完成这些步骤中的一部分。但是，其他步骤是特定于 Systems Manager 的。因此，建议您完整阅读本节内容，确保自己做好使用所有 Systems Manager 工具的准备。

主题

- 使用 Systems Manager 管理 EC2 实例
- 使用 Systems Manager 管理混合和多云环境中的节点
- 使用 Systems Manager 管理边缘设备
- 为 Systems Manager 创建 AWS Organizations 委派管理员
- AWS Systems Manager 的常规设置

使用 Systems Manager 管理 EC2 实例

完成此部分中的任务，为 AWS Systems Manager 设置和配置角色、权限以及初始资源。本节中介绍的任务通常由 AWS 账户 和系统管理员执行。完成这些步骤后，贵企业中的用户可以使用 Systems Manager 来配置、管理和访问 Amazon Elastic Compute Cloud (Amazon EC2) 实例。

> **Note**
>
> 如果您计划使用 Systems Manager 来管理和配置本地计算机，请按照 使用 Systems Manager 管理混合和多云环境中的节点 中的设置步骤操作。如果您计划在混合和多云环境中使用

Amazon EC2 实例和非 EC2 计算机，请先按照下列步骤操作。本节以建议顺序介绍一些步骤，这些步骤配置要在您的 Systems Manager 操作中使用的角色、用户、权限和初始资源。

如果您已使用其他 AWS 服务，则表示您已完成这些步骤中的一部分。但是，其他步骤是特定于 Systems Manager 的。因此，建议您完整阅读本节内容，确保自己做好使用所有 Systems Manager 工具的准备。

内容

- 配置 Systems Manager 所需的实例权限
- 使用适用于 Systems Manager 的 VPC 端点提高 EC2 实例的安全性

配置 Systems Manager 所需的实例权限

默认情况下，AWS Systems Manager 没有在您的实例上执行操作的权限。您可以使用 AWS Identity and Access Management（IAM）角色在账户级别提供实例权限，也可以使用实例配置文件在实例级别提供实例权限。如果您的应用场景允许，我们建议使用“默认主机管理配置”在账户级别授予访问权限。

> **Note**
>
> 您可以跳过此步骤，并允许 Systems Manager 在设置统一控制台时为您的实例应用所需的权限。有关更多信息，请参阅 设置 AWS Systems Manager。

EC2 实例权限的建议配置

“默认主机管理配置”允许 Systems Manager 自动管理您的 Amazon EC2 实例。在您打开此设置后，在 AWS 区域 和已安装 SSM Agent 版本 3.2.582.0 或更高版本的 AWS 账户 中，使用实例元数据服务版本 2（IMDSv2）的所有实例都将自动变为托管实例。“默认主机管理配置”不支持实例元数据服务版本 1。有关过渡到 IMDSv2 的信息，请参阅《Amazon EC2 用户指南》中的过渡到使用实例元数据服务版本 2。有关检查您的实例上安装的 SSM Agent 版本的信息，请参阅 正在检查 SSM Agent 版本号。有关更新 SSM Agent 的信息，请参阅 自动更新 SSM Agent。托管实例具有以下好处：

- 可以使用 Session Manager 安全地连接到您的实例。
- 可以使用 Patch Manager 执行自动补丁扫描。
- 可以使用 Systems Manager 清单查看有关您的实例的详细信息。

- 可以使用 Fleet Manager 追踪和管理实例。
- 可以自动保持 SSM Agent 处于最新状态。

Fleet Manager、Inventory、Patch Manager和Session Manager都是 AWS Systems Manager 中的工具。

“默认主机管理配置”允许在不使用实例配置文件的情况下进行实例管理，并可确保 Systems Manager 有权管理相应区域和账户中的所有实例。如果提供的权限不足以满足您的应用场景要求，您还可以向“默认主机管理配置”创建的默认 IAM 角色添加策略。或者，如果您不需要默认 IAM 角色提供的所有功能的权限，可以创建自己的自定义角色和策略。对您为“默认主机管理配置”选择的 IAM 角色所做的任何更改，都适用于相应区域和账户中的所有托管 Amazon EC2 实例。有关“默认主机管理配置”所用策略的更多信息，请参阅 AWS 托管式策略：AmazonSSMManagedEC2InstanceDefaultPolicy。有关“默认主机管理配置”的更多信息，请参阅使用默认主机管理配置自动管理 EC2 实例。

> **Important**
>
> 使用默认主机管理配置注册的实例将在 `/lib/amazon/ssm` 或 `C:\ProgramData\Amazon` 目录中本地存储注册信息。如果移除这些目录或其中的文件，将导致实例无法使用默认主机管理配置获取连接到 Systems Manager 所需的凭证。在这些情况下，您必须使用实例配置文件为您的实例提供所需的权限，或者重新创建实例。

> **Note**
>
> 此过程仅由管理员执行。在允许个人配置或修改“默认主机管理配置”时，实施最低权限访问。您必须在要自动管理 Amazon EC2 实例的每个 AWS 区域 中打开“默认主机管理配置”。

打开“默认主机管理配置”设置

您可以从 Fleet Manager 控制台打开“默认主机管理配置”。要使用 AWS Management Console 或您的首选命令行工具成功完成此过程，您必须拥有 GetServiceSetting、ResetServiceSetting 和 UpdateServiceSetting API 操作的权限。此外，您还必须有权获得 `AWSSystemsManagerDefaultEC2InstanceManagementRole` IAM 角色的 `iam:PassRole` 权限。以下是示例策略。将每个`#######`替换为您自己的信息。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetServiceSetting",
                "ssm:ResetServiceSetting",
                "ssm:UpdateServiceSetting"
            ],
            "Resource": "arn:aws:ssm:us-east-1:111122223333:servicesetting/ssm/
managed-instance/default-ec2-instance-management-role"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::111122223333:role/service-role/
AWSSystemsManagerDefaultEC2InstanceManagementRole",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ssm.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

在开始之前，如果您已将实例配置文件附加到您的 Amazon EC2 实例，请移除允许该 `ssm:UpdateInstanceInformation` 操作的任何权限。在使用“默认主机管理配置”权限之前，SSM Agent 将尝试使用实例配置文件权限。如果您在实例配置文件中允许该 `ssm:UpdateInstanceInformation` 操作，则该实例将不使用“默认主机管理配置”权限。

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。

2. 在导航窗格中，请选择 Fleet Manager。
3. 在账户管理下拉列表中选择配置默认主机管理配置。
4. 打开启用默认主机管理配置。
5. 选择用于为实例启用 Systems Manager 工具的 IAM角色。我们建议使用“默认主机管理配置”提供的默认角色。它包含使用 Systems Manager 管理您的 Amazon EC2 实例所需的最低权限集合。如果您更喜欢使用自定义角色，则该角色的信任策略必须允许 Systems Manager 作为可信实体。
6. 选择配置以完成设置。

在打开“默认主机管理配置”后，您的实例可能需要最长 30 分钟才能使用所选角色的凭证。您必须在要自动管理 Amazon EC2 实例的每个区域中打开“默认主机管理配置”。

EC2 实例权限的替代配置

您可以使用 AWS Identity and Access Management（IAM）实例配置文件来授予单个实例级别的访问权限。实例配置文件是一个容器，可在启动时将 IAM 角色信息传递给 Amazon Elastic Compute Cloud (Amazon EC2) 实例。您可以把一个或多个定义所需权限的 IAM policy 附加到新角色或已创建的角色上，以便为 Systems Manager 创建实例配置文件。

> **Note**
>
> 可以使用Quick Setup（AWS Systems Manager 中的一项工具），在您 AWS 账户中的所有实例上快速配置实例配置文件。Quick Setup还可创建 IAM 服务角色（或担任角色），该角色允许 Systems Manager 代表您在实例上安全地运行命令。在使用 Quick Setup 时，您可以跳过该步骤（步骤 3）和步骤 4。有关更多信息，请参阅 AWS Systems Manager Quick Setup。

请注意以下有关创建 IAM 实例配置文件的详细信息：

- 如果您在混合和多云环境中为 Systems Manager 配置非 EC2 计算机，则无需为它们创建实例配置文件。相反，您必须将服务器和 VM 配置为使用 IAM 服务角色。有关更多信息，请参阅在混合和多云环境中创建 Systems Manager 所需的 IAM 服务角色。
- 如果更改 IAM 实例配置文件，实例凭证可能需要一些时间才能刷新。刷新后，SSM Agent 才会处理请求。要加快刷新过程，您可以重新启动 SSM Agent或重新启动实例。

根据您是为实例配置文件创建新角色还是为现有角色添加所需权限，请使用以下过程之一。

为 Systems Manager 托管实例创建实例配置文件（控制台）

1. 访问：https://console.aws.amazon.com/iam/，打开 IAM 控制台。
2. 在导航窗格中，选择角色，然后选择创建角色。
3. 对于 Trusted entity type（可信实体类型），选择 AWS 服务。
4. 在 Use case（应用场景）下，选择 EC2，然后选择 Next（下一步）。
5. 在 Add permissions（添加权限）页面上，请执行以下操作：

   - 使用搜索字段查找 AmazonSSMManagedInstanceCore 策略。选中其名称旁边的复选框，如下图所示。



   即使您搜索其他策略，控制台也会保留您选择的内容。

   - 如果您在上一过程中创建了自定义 S3 存储桶策略（可选）为 S3 存储桶访问创建自定义策略，请搜索该策略并选中其名称旁边的复选框。
   - 如果您打算将实例加入 AWS Directory Service 管理的 Active Directory，请搜索 AmazonSSMDirectoryServiceAccess 并选中其名称旁边的复选框。
   - 如果您打算使用 Eventbridge 或 CloudWatch Logs 来管理或监控您的实例，请搜索 CloudWatchAgentServerPolicy 并选中其名称旁边的复选框。
6. 选择下一步。
7. 对于Role name（角色名称），请输入新实例配置文件的名称，例如 **`SSMInstanceProfile`**。

   > **Note**
   >
   > 记下角色名称。在创建希望使用 Systems Manager 进行管理的新实例时，将选择该角色。

8. （可选）对于 Description（描述），更新此实例配置文件的描述。
9. （可选）对于 Tags（标签），添加一个或多个标签键值对，以组织、跟踪或控制此角色的访问，然后选择 Create role（创建角色）。系统将让您返回到 角色 页面。

将 Systems Manager 的实例配置文件权限添加到现有角色（控制台）

1. 访问：https://console.aws.amazon.com/iam/，打开 IAM 控制台
2. 在导航窗格中，选择角色，然后选择要与 Systems Manager 操作关联的实例配置文件关联的现有角色。
3. 在 Permissions（权限）选项卡上，选择 Add Permissions, Attach policies（添加权限，附上策略）。
4. 在 Attach policy（附上策略）页面上，执行以下操作：
   - 使用搜索字段查找 AmazonSSMManagedInstanceCore 策略。选中其名称旁边的复选框。
   - 如果您已经创建了自定义 S3 存储桶策略，请搜索该策略并选中其名称旁边的复选框。有关实例配置文件的自定义 S3 存储桶策略的信息，请参阅（可选）为 S3 存储桶访问创建自定义策略。
   - 如果您打算将实例加入 AWS Directory Service 管理的 Active Directory，请搜索 AmazonSSMDirectoryServiceAccess 并选中其名称旁边的复选框。
   - 如果您打算使用 Eventbridge 或 CloudWatch Logs 来管理或监控您的实例，请搜索 CloudWatchAgentServerPolicy 并选中其名称旁边的复选框。
5. 选择附加策略。

有关如何更新角色以包含可信实体或进一步限制访问的信息，请参阅 IAM 用户指南中的修改角色。

（可选）为 S3 存储桶访问创建自定义策略

只有在 Systems Manager 操作中使用 VPC 终端节点或使用您自己的 S3 存储桶时，才需要为 Amazon S3 访问创建自定义策略。您可以将此策略附加到由“默认主机管理配置”创建的默认 IAM 角色，或您在之前的过程中创建的实例配置文件。

有关您在以下策略中提供访问权限的 AWS 托管 S3 存储桶的信息，请参阅 SSM Agent 与 AWS 托管 S3 存储桶进行通信。

1. 访问：https://console.aws.amazon.com/iam/，打开 IAM 控制台。
2. 在导航窗格中，选择 Policies (策略)，然后选择 Create policy (创建策略)。
3. 选择 JSON 选项卡，并将原定设置文本替换为以下内容。

JSON

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": [
                "arn:aws:s3:::aws-ssm-us-east-2/*",
                "arn:aws:s3:::aws-windows-downloads-us-east-2/*",
                "arn:aws:s3:::amazon-ssm-us-east-2/*",
                "arn:aws:s3:::amazon-ssm-packages-us-east-2/*",
                "arn:aws:s3:::us-east-2-birdwatcher-prod/*",
                "arn:aws:s3:::aws-ssm-distributor-file-us-east-2/*",
                "arn:aws:s3:::aws-ssm-document-attachments-us-east-2/*",
                "arn:aws:s3:::patch-baseline-snapshot-us-east-2/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject",
                "s3:PutObjectAcl",
                "s3:GetEncryptionConfiguration"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/*",
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        }
    ]
}
```

[1] 只有在使用 VPC 终端节点时，才需要使用第一个 `Statement` 元素。

[2] 只有在 Systems Manager 操作中使用您创建的 S3 存储桶时，才需要使用第二个 `Statement` 元素。

[3] 只有在您打算为其他账户中的 S3 存储桶提供跨账户访问支持时，才需要使用 `PutObjectAcl` 访问控制列表权限。

[4] 如果您的 S3 存储桶配置为使用加密，则 `GetEncryptionConfiguration` 元素是必需的。

[5] 如果您的 S3 存储桶配置为使用加密，则 S3 存储桶根（例如 `arn:aws:s3:::amzn-s3-demo-bucket`）必须列在资源部分。您的用户、组或角色必须配置为有权访问根存储桶。

4. 如果在操作中使用 VPC 终端节点，请执行以下操作：

   在第一个 `Statement` 元素中，将每个*##*占位符替换为将使用该策略 AWS 区域 的标识符。例如，对于美国东部 (俄亥俄) 区域，请使用 `us-east-2`。有关支持的 *`region`* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

   > **Important**
   >
   > 我们建议您避免在该策略中使用通配符 (*) 以替代特定区域。例如，使用 `arn:aws:s3:::aws-ssm-us-east-2/*` 而不使用 `arn:aws:s3:::aws-ssm-*/*`。使用通配符可能会提供对您不打算授予访问权限的 S3 存储桶的访问。如果要将实例配置文件用于多个区域，我们建议每个区域重复使用第一个 `Statement` 元素。

   -或者-

   如果在操作中不使用 VPC 终端节点，您可以删除第一个 `Statement` 元素。

5. 如果在 Systems Manager 操作中使用您自己的 S3 存储桶，请执行以下操作：

   在第二个 `Statement` 元素中，将 *`amzn-s3-demo-bucket`* 替换为您的账户中的 S3 存储桶的名称。将在 Systems Manager 操作中使用该存储桶。它将 `"arn:aws:s3:::my-bucket-name/*"` 作为资源，以便为存储桶中的对象提供权限。有关为存储桶或存储桶中的对象提供权限的更多信息，请参阅 Amazon Simple Storage Service 用户指南中的 Amazon S3 操作主题，以及 AWS 博客文章 IAM policy、存储桶策略和 ACL！Oh, My! (Controlling Access to S3 Resources)。

   > **Note**
   >
   > 如果您使用多个存储桶，请提供每个存储桶的 ARN。有关存储桶的权限，请参阅以下示例。
   >
   > ```
   > "Resource": [
   > "arn:aws:s3:::amzn-s3-demo-bucket1/*",
   > "arn:aws:s3:::amzn-s3-demo-bucket2/*"
   >               ]
   > ```

- 或者 -

如果在 Systems Manager 操作中不使用您自己的 S3 存储桶，您可以删除第二个 `Statement` 元素。

6. 选择下一步：标签。
7. （可选）通过选择 Add tag（添加标签）并输入策略的首选标签来添加标签。
8. 选择下一步：审核。
9. 在 Name（名称）中，输入用于标识此策略的名称，例如 **`SSMInstanceProfileS3Policy`**。
10. 选择创建策略。

托管实例的其他策略注意事项

本部分介绍了一些策略，您可以将这些策略添加到由“默认主机管理配置”或您的 AWS Systems Manager 实例配置文件创建的默认 IAM 角色中。要为实例和 Systems Manager API 之间的通信提供权限，我们建议您创建反映系统需求和安全要求的自定义策略。根据您的操作计划，您可能需要具有在其他策略的一个或多个策略中表示的权限。

策略：**`AmazonSSMDirectoryServiceAccess`**

只有在您打算将 Windows Server 的 Amazon EC2 实例加入 Microsoft AD 目录时才需要。

该 AWS 托管策略允许 SSM Agent 代表您访问 AWS Directory Service，以处理托管实例加入域的请求。有关更多信息，请参阅 AWS Directory Service 管理指南中的无缝加入 Windows EC2 实例。

策略：**`CloudWatchAgentServerPolicy`**

只有在您打算在实例上安装并运行 CloudWatch 代理，以读取实例上的指标和日志数据并将其写入到 Amazon CloudWatch 时才需要。这些操作可以帮助您监控、分析和快速响应存在的问题或对您的 AWS 资源的更改。

仅当您将使用 Amazon EventBridge 或 Amazon CloudWatch Logs 等功能时，由“默认主机管理配置”或实例配置文件创建的默认 IAM 角色才需要此策略。（您还可以创建限制性更强的策略，例如，限制对特定 CloudWatch Logs 日志流的写入访问。）

> **Note**
>
> 使用 EventBridge 和 CloudWatch Logs 功能是可选的。但是，如果您决定使用这些功能，我们建议您在开始执行 Systems Manager 配置过程时就设置这些功能。有关更多信息，请参阅 Amazon EventBridge User Guide 和 Amazon CloudWatch Logs User Guide。

要创建拥有其他 Systems Manager 工具权限的 IAM 策略，请参阅以下资源：

- 使用 IAM 策略限制对 Parameter Store 参数的访问
- 设置自动化
- 步骤 2：验证或添加 Session Manager 的实例权限

将 Systems Manager 实例配置文件附加到实例（控制台）

以下步骤介绍了如何使用 Amazon EC2 控制台将 IAM 实例配置文件附加到 Amazon EC2 实例。

1. 登录 AWS Management Console，打开 Amazon EC2 控制台：https://console.aws.amazon.com/ec2/。
2. 在导航窗格中的 Instances 下，选择 Instances。
3. 导航至并从列表中选择您的 EC2 实例。
4. 在操作 菜单中，选择安全、修改 IAM 角色。
5. 对于 IAM role (IAM 角色)，选择您使用 EC2 实例权限的替代配置 中的过程创建的实例配置文件。
6. 选择 Update IAM role（更新 IAM 角色）。

有关将 IAM 角色附加到实例的更多信息，请选择以下操作之一，具体取决于您选择的操作系统类型：

- 《Amazon EC2 用户指南》中的将 IAM 角色附加到实例
- 《Amazon EC2 用户指南》中的将 IAM 角色附加到实例

继续使用适用于 Systems Manager 的 VPC 端点提高 EC2 实例的安全性。

使用适用于 Systems Manager 的 VPC 端点提高 EC2 实例的安全性

您可以配置 AWS Systems Manager 在 Amazon Virtual Private Cloud（Amazon VPC）中使用接口 VPC 端点，以提高托管式节点（包括混合和多云环境中的非 EC2 实例）的安全状况。通过使用接口

VPC 终端节点（接口端点），您可以连接到 AWS PrivateLink 支持的服务。AWS PrivateLink 技术允许您使用私有 IP 地址私下访问 Amazon Elastic Compute Cloud (Amazon EC2) 和 Systems Manager API。

AWS PrivateLink 将托管实例、Systems Manager 和 Amazon EC2 之间的所有网络流量限制在 Amazon 网络以内。这意味着您的托管实例无法访问 Internet。如果您使用 AWS PrivateLink，则无需互联网网关、NAT 设备或虚拟私有网关。

不要求您配置 AWS PrivateLink，但推荐进行此配置。有关 AWS PrivateLink 和 VPC 终端节点的更多信息，请参阅AWS PrivateLink 和 VPC 终端节点。

> **Note**
>
> 使用 VPC 终端节点的替代方法是，在托管实例上允许出站 Internet 访问。在这种情况下，托管实例还必须允许以下终端节点的 HTTPS（端口 443）出站流量：
>
> - `ssm.region.amazonaws.com`
> - `ssmmessages.region.amazonaws.com`
> - `ec2messages.region.amazonaws.com`
>
> SSM Agent 将在云中启动所有与 Systems Manager 服务的连接。因此，您无需为 Systems Manager 配置防火墙以允许入站流量到达您的实例。
> 有关对这些终端节点的调用的更多信息，请参阅 参考：ec2messages、ssmmessages 和其他 API 操作。

关于 Amazon VPC

您可以使用 Amazon Virtual Private Cloud (Amazon VPC) 在 AWS Cloud 内您自己的逻辑隔离区域中定义虚拟化网络，我们称之为 虚拟私有云 (VPC)。可在 VPC 中启动实例等 AWS 资源。您的 VPC 与您可能在自己的数据中心中运行的传统网络极为相似，同时享有使用来自 AWS 的可扩展基础设施的优势。您可以配置您的 VPC；您可以选择它的 IP 地址范围、创建子网并配置路由表、网关和安全设置。您可以将您的 VPC 中的实例连接到网络。您可以将您的 VPC 连接到公司的数据中心，并将 AWS Cloud 作为数据中心的延伸。要保护各个子网中的资源，您可以利用多种安全层，包括安全组和网络访问控制列表。有关更多信息，请参阅《Amazon VPC 用户指南》。

主题

- VPC 终端节点限制

- 为 Systems Manager 创建 VPC 终端节点
- 创建接口 VPC 终端节点策略

VPC 终端节点限制

在配置 Systems Manager 的 VPC 终端节点之前，请注意以下限制。

VPC 对等连接

VPC 接口终端节点可以通过区域内和区域间 VPC 对等连接访问。有关 VPC 接口端点的 VPC 对等连接请求的更多信息，请参阅《Amazon Virtual Private Cloud 用户指南》中的 VPC 对等连接 (Quotas)。

无法将 VPC 网关端点连接扩展到 VPC 之外。VPC 中的 VPC 对等连接另一端的资源不能使用网关终端节点与网关终端节点服务中的资源进行通信。有关 VPC 网关端点的 VPC 对等连接请求的更多信息，请参阅《Amazon Virtual Private Cloud 用户指南》中的 VPC 端点 (Quotas)

传入连接

附加到 VPC 终端节点的安全组必须允许从托管实例的私有子网通过端口 443 进行传入连接。如果不允许传入连接，则托管实例无法连接到 SSM 和 EC2 终端节点。

DNS 解析

如果您使用自定义 DNS 服务器，则必须将针对 `amazonaws.com` 域的任何查询的条件转发器添加到 VPC 的 Amazon DNS 服务器。

S3 存储桶

VPC 端点策略必须允许访问至少 SSM Agent 与 AWS 托管 S3 存储桶进行通信 中列出的 Amazon S3 存储桶。

> **Note**
>
> 如果您使用本地防火墙并计划使用 Patch Manager，还必须允许该防火墙访问适当的补丁基准终端节点。

Amazon CloudWatch Logs

如果您不允许您的实例访问互联网，请为 CloudWatch Logs 创建 VPC 终端节点，以使用向 CloudWatch 日志发送日志的功能。有关为 CloudWatch Logs 创建终端节点的更多信息，请参阅《Amazon CloudWatch Logs 用户指南》中的为 CloudWatch Logs 创建 VPC 终端节点。

混合和多云环境中的 DNS

有关配置 DNS 以在混合和多云环境中与 AWS PrivateLink 端点一起使用的信息，请参阅《Amazon VPC 用户指南》中的用于接口端点的私有 DNS。如果需要使用自己的 DNS，可以使用 Route 53 解析程序。有关更多信息，请参阅《Amazon Route 53 开发人员指南》中的解析 VPC 与网络之间的 DNS 查询。

为 Systems Manager 创建 VPC 终端节点

使用以下信息为 AWS Systems Manager 创建 VPC 接口端点。本主题链接到《Amazon VPC 用户指南》中的过程。

> **Note**
>
> *region* 表示 AWS Systems Manager 支持的 AWS 区域 的标识符，例如 us-east-2 对应美国东部（俄亥俄）区域。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

按照创建接口端点中的步骤操作，创建以下接口端点：

- **com.amazonaws.*region*.ssm** – Systems Manager 服务的端点。
- **com.amazonaws.*region*.ec2messages** – Systems Manager 使用此端点从 SSM Agent 调用到 Systems Manager 服务。从 SSM Agent 的版本 3.3.40.0 开始，只要可用，Systems Manager 就会使用 ssmmessages:* 端点（Amazon Message Gateway Service）而非 ec2messages:* 端点（Amazon Message Delivery Service）。
- **com.amazonaws.*region*.ec2** – 如果您使用 Systems Manager 创建启用 VSS 的快照，则需要确保您具有连接到 EC2 服务的端点。如果未定义 EC2 端点，枚举附加的 Amazon EBS 卷的调用将失败，这会导致 Systems Manager 命令失败。
- **com.amazonaws.*region*.s3**：Systems Manager 使用此端点更新 SSM Agent。如果选择检索存储在存储桶中的脚本或其他文件，或将输出日志上传到存储桶，Systems Manager 也会使用此端点。如果与您的实例关联的安全组限制出站流量，则您必须添加一条规则，以允许流量到达 Amazon S3 的前缀列表。有关更多信息，请参阅《AWS PrivateLink 指南》中的修改安全组。

- **com.amazonaws.*region*.ssmmessages** – 对于 Run Command，SSM Agent 需要此端点才能与 Systems Manager 服务通信，并且在您使用 Session Manager 通过安全数据通道连接到您的实例时需要此端点。有关更多信息，请参阅AWS Systems Manager Session Manager和参考：ec2messages、ssmmessages 和其他 API 操作。
- （可选）**com.amazonaws.*region*.kms**：如果要对 Session Manager 或 Parameter Store 参数使用 AWS Key Management Service（AWS KMS）加密，请创建此端点。
- （可选）**com.amazonaws.*region*.logs**：如果要将 Amazon CloudWatch Logs (CloudWatch Logs) 用于 Session Manager、Run Command 或 SSM Agent 日志，请创建此端点。

有关 SSM Agent 必须能够访问的 AWS 托管 S3 存储桶的信息，请参阅 SSM Agent 与 AWS 托管 S3 存储桶进行通信。如果您正在 Systems Manager 操作中使用虚拟私有云（VPC）端点，则必须在 Systems Manager 的 EC2 实例配置文件中或在混合和多云环境中的非 EC2 托管式节点的服务角色中提供显式权限。

创建接口 VPC 终端节点策略

您可以为 AWS Systems Manager 的 VPC 接口终端节点创建策略，在其中可以指定：

- 可执行操作的主体
- 可执行的操作
- 可被执行操作的资源

有关更多信息，请参阅《Amazon VPC 用户指南》中的使用 VPC 终端节点控制对服务的访问权限。

使用 Systems Manager 管理混合和多云环境中的节点

您可以使用 AWS Systems Manager 管理 Amazon Elastic Compute Cloud（EC2）实例和许多非 EC2 计算机类型。本节介绍账户和系统管理员为在混合和多云环境中使用 Systems Manager 管理非 EC2 计算机而执行的设置任务。完成这些步骤后，已获得 AWS 账户 管理员授予权限的用户可以使用 Systems Manager 来配置和管理其组织的非 EC2 计算机。

任何已配置为与 Systems Manager 一起使用的计算机都称为托管节点。

> **Note**
>
> - 您可以使用与其他非 EC2 计算机相同的混合激活步骤，将边缘设备注册为托管式节点。这些类型的边缘设备包括 AWS IoT 设备和 AWS IoT 设备以外的设备。使用本部分说明的步骤来设置这些边缘设备类型。
>
> Systems Manager 还支持使用 AWS IoT Greengrass Core 软件的边缘设备。AWS IoT Greengrass 核心设备的设置过程与要求，不同于 AWS 边缘设备以外的 AWS IoT 和边缘设备的设置与要求。有关注册 AWS IoT Greengrass 设备以便与 Systems Manager 搭配使用的信息，请参阅 使用 Systems Manager 管理边缘设备。
> - Systems Manager 混合和多云环境不支持非 EC2 macOS 计算机。

如果您计划使用 Systems Manager 管理 Amazon Elastic Compute Cloud（Amazon EC2）实例，或者要在混合和多云环境中使用 Amazon EC2 实例和非 EC2 计算机，请先按照 使用 Systems Manager 管理 EC2 实例 中的步骤操作。

在为 Systems Manager 配置混合和多云环境后，您可以执行以下操作：

- 创建一种一致且安全的方式，使用相同的工具或脚本从一个位置远程管理混合和多云工作负载。
- 通过使用 AWS Identity and Access Management (IAM) 集中管理可在您的计算机上执行的操作访问控制。
- 通过查看 AWS CloudTrail 中记录的 API 活动，可集中审核在计算机上执行的操作。

  有关使用 CloudTrail 监控 Systems Manager 操作的信息，请参阅 使用 AWS CloudTrail 记录 AWS Systems Manager API 调用。
- 通过将 Amazon EventBridge 和 Amazon Simple Notification Service (Amazon SNS) 配置为发送有关服务执行成功的通知，从而实现集中监控。

  有关使用 EventBridge 监控 Systems Manager 事件的信息，请参阅 使用 Amazon EventBridge 监控 Systems Manager 事件。

关于托管式节点

在您按照本节所述为 Systems Manager 配置完非 EC2 计算机后，您的混合激活计算机将在 AWS Management Console 中列出并被称为托管式节点。在控制台中，混合激活托管式节点的 ID 与具有前缀“mi-”的 Amazon EC2 实例有区别。Amazon EC2 实例 ID 使用前缀“i-”。

托管式节点是为 Systems Manager 配置的任何计算机。托管式节点以前都被称为托管实例。现在，实例一词仅指 EC2 实例。在此术语更改之前，deregister-managed-instance 命令就已命名。

有关更多信息，请参阅 使用托管式节点。

关于实例层

Systems Manager 为混合和多云环境中的非 EC2 托管式节点提供标准实例套餐和高级实例套餐。通过标准实例套餐，每个 AWS 区域 中的每个 AWS 账户 最多可以注册 1000 台混合激活的计算机。如果您需要在单个账户和区域中注册超过 1000 台计算机，请使用高级实例套餐。通过高级实例，您还可以使用 AWS Systems Manager Session Manager 连接到非 EC2 计算机。Session Manager 提供对托管式节点的交互式 Shell 访问。

有关更多信息，请参阅 配置实例套餐。

主题

- 在混合和多云环境中创建 Systems Manager 所需的 IAM 服务角色
- 创建混合激活以将节点注册到 Systems Manager
- 在混合 Linux 节点上安装 SSM Agent
- 在混合 Windows Server 节点上安装 SSM Agent

在混合和多云环境中创建 Systems Manager 所需的 IAM 服务角色

混合和多云环境中的非 EC2（Amazon Elastic Compute Cloud）计算机需要一个 AWS Identity and Access Management（IAM）服务角色才能与 AWS Systems Manager 服务通信。此角色向 AWS Security Token Service (AWS STS) `AssumeRole` 授予对 Systems Manager 服务的信任。对于混合和多云环境，您只需为每个 AWS 账户 创建一次服务角色。但如果混合和多云环境中的计算机需要不同的权限，则可以选择为不同的混合激活创建多个服务角色。

以下过程介绍如何使用 Systems Manager 控制台或首选命令行工具来创建所需的服务角色。

使用 AWS Management Console 为 Systems Manager 混合激活创建 IAM 服务角色

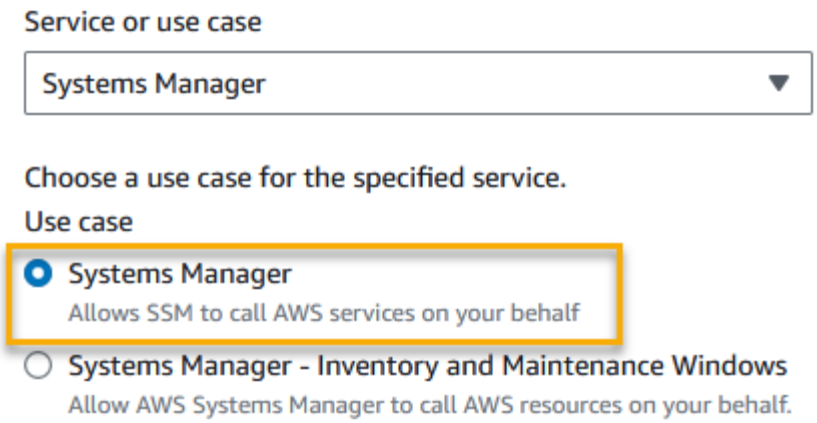
使用以下过程为混合激活创建服务角色。此过程使用 Systems Manager 核心功能的 `AmazonSSMManagedInstanceCore` 策略。您可能需要向服务角色添加其他策略，以便本地计算机能够访问其他 Systems Manager 工具或 AWS 服务，具体取决于您的应用场景。例如，如果无法访问所需的 AWS 托管式 Amazon Simple Storage Service (Amazon S3) 存储桶，Patch Manager 修补操作将会失败。

创建服务角色（控制台）

1. 通过 https://console.aws.amazon.com/iam/ 打开 IAM 控制台。
2. 在导航窗格中，选择角色，然后选择创建角色。
3. 对于 Select trusted entity（选择可信实体），完成以下选择：

   1. 对于 Trusted entity type（可信实体类型），选择 AWS 服务。
   2. 对于其他 AWS 服务 的应用场景，请选择 Systems Manager。
   3. 选择 Systems Manager。

   下图突出显示了 Systems Manager 选项的位置。



4. 选择下一步。
5. 在 Add permissions（添加权限）页面上，请执行以下操作：

   - 使用搜索字段查找 AmazonSSMManagedInstanceCore 策略。选中其名称旁边的复选框，如下图所示。



> **Note**
>
> 即使您搜索其他策略，控制台也会保留您选择的内容。

- 如果您在过程（可选）为 S3 存储桶访问创建自定义策略 中创建了自定义 S3 存储桶策略，请搜索该策略并选中其名称旁边的复选框。
- 如果您打算将非 EC2 计算机加入 AWS Directory Service 管理的 Active Directory，请搜索 AmazonSSMDirectoryServiceAccess 并选中其名称旁边的复选框。
- 如果您打算使用 Eventbridge 或 CloudWatch Logs 来管理或监控您的托管节点，请搜索 CloudWatchAgentServerPolicy 并选中其名称旁边的复选框。

6. 选择下一步。
7. 对于角色名称，请为新的 IAM 服务器角色输入名称，例如 **SSMServerRole**。

> **Note**
>
> 记下角色名称。在注册希望使用 Systems Manager 进行管理的新计算机时，将选择该角色。

8. （可选）对于描述，请更新该 IAM 服务器角色的描述。
9. （可选）在 Tags（标签）中添加一个或多个标签密钥值对，以组织、跟踪或控制此角色的访问权限。
10. 选择 Create role (创建角色)。系统将让您返回到 角色 页面。

使用 AWS CLI 为 Systems Manager 混合激活创建 IAM 服务角色

使用以下过程为混合激活创建服务角色。此过程使用 Systems Manager 核心功能的 `AmazonSSMManagedInstanceCore` 策略。您可能需要向服务角色添加其他策略，以便混合和多云环境中的非 EC2 计算机能够访问其他工具或 AWS 服务，具体取决于您的应用场景。

S3 存储桶策略要求

如果出现以下任一情况，您必须在完成此过程之前为 Amazon Simple Storage Service (Amazon S3) 存储桶创建自定义 IAM 权限策略：

- 情况 1：您通过一个 VPC 端点以私有方式将您的 VPC 连接到受支持的 AWS 服务，以及由 AWS PrivateLink 提供支持的 VPC 端点服务。
- 情况 2 - 您计划使用在 Systems Manager 操作期间创建的 Amazon S3 存储桶，如将 Run Command 命令或 Session Manager 会话的输出存储到 S3 存储桶中。在继续之前，请按照为实例配置文件创建自定义 S3 存储桶策略中的步骤进行操作。该主题中有关 S3 存储桶策略的信息也适用于服务角色。

AWS CLI

为混合和多云环境创建 IAM 服务角色（AWS CLI）

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 在本地计算机上使用以下信任策略创建一个文本文件，文件名称类似于 SSMService-Trust.json。确保使用 .json 文件扩展名保存该文件。请务必指定您在其中创建混合激活的 ARN 中的 AWS 账户 和 AWS 区域。请将账户 ID 和区域的####替换为您自己的信息。

   JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"",
         "Effect":"Allow",
         "Principal":{
            "Service":"ssm.amazonaws.com"
         },
         "Action":"sts:AssumeRole",
         "Condition":{
            "StringEquals":{
               "aws:SourceAccount":"123456789012"
            },
            "ArnEquals":{
               "aws:SourceArn":"arn:aws:ssm:us-east-2:123456789012:*"
            }
         }
      }
   ]
}
```

3. 打开 AWS CLI，在创建该 JSON 文件的目录下，运行 create-role 命令以创建服务角色。该示例创建一个名为 SSMServiceRole 的角色。如果愿意，您也可以选择其他名称。

   Linux & macOS

```
aws iam create-role \
```

```
    --role-name SSMServiceRole \
    --assume-role-policy-document file://SSMService-Trust.json
```

Windows

```
aws iam create-role ^
    --role-name SSMServiceRole ^
    --assume-role-policy-document file://SSMService-Trust.json
```

4. 按如下方式运行 attach-role-policy 命令，以允许您刚创建的服务角色创建会话令牌。此会话令牌向托管节点授予使用 Systems Manager 运行命令的权限。

> **Note**
>
> 您为混合和多云环境中托管式节点的服务配置文件添加的策略与用于为 Amazon Elastic Compute Cloud（Amazon EC2）实例创建实例配置文件的策略相同。有关以下命令中使用的 AWS 策略的更多信息，请参阅配置 Systems Manager 所需的实例权限。

（必需）运行以下命令以允许托管节点使用 AWS Systems Manager 服务核心功能。

Linux & macOS

```
aws iam attach-role-policy \
    --role-name SSMServiceRole \
    --policy-arn arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
```

Windows

```
aws iam attach-role-policy ^
    --role-name SSMServiceRole ^
    --policy-arn arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
```

如果为服务角色创建了自定义 S3 存储桶策略，请运行以下命令，以允许 AWS Systems Manager Agent (SSM Agent) 访问在该策略中指定的存储桶。将 *account-id* 和 *amzn-s3-demo-bucket* 替换为您的 AWS 账户 ID 和存储桶名称。

Linux & macOS

```
aws iam attach-role-policy \
    --role-name SSMServiceRole \
    --policy-arn arn:aws:iam::account-id:policy/amzn-s3-demo-bucket
```

Windows

```
aws iam attach-role-policy ^
    --role-name SSMServiceRole ^
    --policy-arn arn:aws:iam::account-id:policy/amzn-s3-demo-bucket
```

（可选）运行以下命令以允许 SSM Agent 代表您访问 AWS Directory Service，以处理托管节点加入域的请求。只有在将节点加入 Microsoft AD 目录时，服务角色才需要使用此策略。

Linux & macOS

```
aws iam attach-role-policy \
    --role-name SSMServiceRole \
    --policy-arn arn:aws:iam::aws:policy/AmazonSSMDirectoryServiceAccess
```

Windows

```
aws iam attach-role-policy ^
    --role-name SSMServiceRole ^
    --policy-arn arn:aws:iam::aws:policy/AmazonSSMDirectoryServiceAccess
```

（可选）运行以下命令以允许 CloudWatch 代理在托管式节点上运行。通过使用此命令，可以读取节点上的信息并将其写入 CloudWatch。仅当要使用 Amazon EventBridge 或 Amazon CloudWatch Logs 等服务时，服务配置文件才需要使用该策略。

```
aws iam attach-role-policy \
    --role-name SSMServiceRole \
    --policy-arn arn:aws:iam::aws:policy/CloudWatchAgentServerPolicy
```

Tools for PowerShell

为混合和多云环境创建 IAM 服务角色（AWS Tools for Windows PowerShell）

1. 如果您尚未安装和配置 AWS Tools for PowerShell（适用于 Windows PowerShell 的工具），请执行这些操作。

   有关信息，请参阅安装 AWS Tools for PowerShell。

2. 在本地计算机上使用以下信任策略创建一个文本文件，文件名称类似于 `SSMService-Trust.json`。确保使用 `.json` 文件扩展名保存该文件。请务必指定您在其中创建混合激活的 ARN 中的 AWS 账户 和 AWS 区域。

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": "ssm.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "123456789012"
                },
                "ArnEquals": {
                    "aws:SourceArn": "arn:aws:ssm:us-
east-1:123456789012:*"
                }
            }
        }
    ]
}
```

3. 在管理模式下打开 PowerShell，在创建此 JSON 文件的目录中，按如下方式运行 New-IAMRole 以创建服务角色。该示例创建一个名为 `SSMServiceRole` 的角色。如果愿意，您也可以选择其他名称。

```
New-IAMRole `
    -RoleName SSMServiceRole `
    -AssumeRolePolicyDocument (Get-Content -raw SSMService-Trust.json)
```

4. 按如下方式使用 Register-IAMRolePolicy，以允许您创建的服务角色创建会话令牌。此会话令牌向托管节点授予使用 Systems Manager 运行命令的权限。

> **Note**
>
> 您为混合和多云环境中托管式节点的服务配置文件添加的策略与用于为 EC2 实例创建实例配置文件的策略相同。有关以下命令中使用的 AWS 策略的更多信息，请参阅配置 Systems Manager 所需的实例权限。

（必需）运行以下命令以允许托管节点使用 AWS Systems Manager 服务核心功能。

```
Register-IAMRolePolicy `
    -RoleName SSMServiceRole `
    -PolicyArn arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
```

如果为服务角色创建了自定义 S3 存储桶策略，请运行以下命令，以允许 SSM Agent 访问在该策略中指定的存储桶。将 *account-id* 和 *my-bucket-policy-name* 替换为您的 AWS 账户 ID 和存储桶名称。

```
Register-IAMRolePolicy `
    -RoleName SSMServiceRole `
    -PolicyArn arn:aws:iam::account-id:policy/my-bucket-policy-name
```

（可选）运行以下命令以允许 SSM Agent 代表您访问 AWS Directory Service，以处理托管节点加入域的请求。只有在将节点加入 Microsoft AD 目录时，服务器角色才需要此策略。

```
Register-IAMRolePolicy `
    -RoleName SSMServiceRole `
    -PolicyArn arn:aws:iam::aws:policy/AmazonSSMDirectoryServiceAccess
```

（可选）运行以下命令以允许 CloudWatch 代理在托管式节点上运行。通过使用此命令，可以读取节点上的信息并将其写入 CloudWatch。仅当要使用 Amazon EventBridge 或 Amazon CloudWatch Logs 等服务时，服务配置文件才需要使用该策略。

```
Register-IAMRolePolicy `
    -RoleName SSMServiceRole `
    -PolicyArn arn:aws:iam::aws:policy/CloudWatchAgentServerPolicy
```

继续创建混合激活以将节点注册到 Systems Manager。

创建混合激活以将节点注册到 Systems Manager

要将 Amazon Elastic Compute Cloud（EC2）实例以外的计算机设置为混合和多云环境的托管式节点，您需要创建并应用混合激活。成功完成激活后，您将立即在控制台页面的顶部收到一个激活代码和激活 ID。在混合和多云环境中的非 EC2 计算机上安装 AWS Systems Manager SSM Agent 时，指定此代码和 ID 组合。此代码和 ID 提供从托管式节点对 Systems Manager 服务的安全访问。

> **Important**
>
> 根据您创建激活的方式，Systems Manager 立即将激活码和 ID 返回到控制台或命令窗口。复制此信息并将其存储在安全位置。如果您离开该控制台或关闭命令窗口，可能会丢失此信息。如果您丢失对应信息，则必须创建一个新激活。

关于激活过期时间

激活过期是您可向 Systems Manager 注册本地计算机的时间范围。过期的激活对您以前向 Systems Manager 注册的服务器或虚拟机没有任何影响。如果激活过期，则无法使用该特定激活向 Systems Manager 注册更多服务器或虚拟机。您只需创建新的激活。

在明确将以前注册的每个本地服务器和虚拟机取消注册以前，它们仍保留注册为 Systems Manager 托管节点。可按以下方式取消注册非 EC2 托管式节点：

- 使用 Systems Manager 控制台中 Fleet Manager 工具内的托管式节点选项卡
- 使用 AWS CLI 命令 deregister-managed-instance
- 使用 API 操作 DeregisterManagedInstance。

有关更多信息，请参阅以下主题

- 取消注册及重新注册托管式节点（Linux）

- 取消注册及重新注册托管式节点（Windows Server）

关于托管式节点

托管节点是为 AWS Systems Manager 配置的任何计算机。AWS Systems Manager 支持 Amazon Elastic Compute Cloud (Amazon EC2) 实例、边缘设备以及本地服务器或虚拟机 (VM)，包括其他云环境中的虚拟机。托管式节点以前都被称为托管实例。现在，实例一词仅指 EC2 实例。在此术语更改之前，deregister-managed-instance 命令就已命名。

关于激活标签

如果使用 AWS Command Line Interface (AWS CLI) 或 AWS Tools for Windows PowerShell 创建激活，则可以指定标签。标签是您分配给资源的可选元数据。标签可让您按不同的方式（如用途、拥有者或环境）对资源进行分类。以下是一个在包含可选标签的本地 Linux 计算机上在美国东部（俄亥俄州）区域运行的 AWS CLI 示例命令。

```
aws ssm create-activation \
  --default-instance-name MyWebServers \
  --description "Activation for Finance department webservers" \
  --iam-role service-role/AmazonEC2RunCommandRoleForManagedInstances \
  --registration-limit 10 \
  --region us-east-2 \
  --tags "Key=Department,Value=Finance"
```

如果在创建激活时指定了标签，那么在激活托管式节点后，系统将自动为它们分配这些标签。

您无法在现有激活中添加或删除标签。如果您不希望使用激活自动为本地服务器和虚拟机分配标签，则可以稍后在其中添加标签。更具体地说，您可以在本地服务器和虚拟机首次连接到 Systems Manager 后标记这些服务器和虚拟机。在连接后，将会为它们分配托管节点 ID，并在 Systems Manager 控制台中列出这些 ID 中带有“mi-”前缀的服务器和虚拟机。

> **Note**
>
> 如果使用 Systems Manager 控制台创建激活，则无法为其分配标签。必须使用 AWS CLI 或 Tools for Windows PowerShell 来创建激活。

如果您不再希望使用 Systems Manager 管理本地服务器或虚拟机 (VM)，可以将其取消注册。有关信息，请参阅在混合和多云环境中取消注册托管式节点。

主题

- 使用 AWS Management Console 创建用于将托管式节点注册到 Systems Manager 的激活
- 使用命令行创建用于将托管式节点注册到 Systems Manager 的激活

使用 AWS Management Console 创建用于将托管式节点注册到 Systems Manager 的激活

创建托管节点激活

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Hybrid Activations (混合激活)。
3. 选择创建激活。

   –或者–

   如果您在当前 AWS 区域首次访问 Hybrid Activations（混合激活）功能，请选择 Create an Activation（创建激活）。
4. （可选）对于 Activation description（激活描述），输入对此激活的描述。如果您计划激活大量服务器和虚拟机，我们建议您输入描述。
5. 在 Instance limit（实例限制）中，指定要在激活过程中注册到 AWS 的节点总数。默认值为 1 个实例。
6. 对于 IAM role（IAM 角色），选择一个将允许您的服务器和虚拟机与云中的 AWS Systems Manager 进行通信的服务角色选项：

   - 选项 1：选择 Use the default role created by the system（使用系统创建的默认角色）以使用由 AWS 提供的角色和托管式策略。
   - 选项 2：选择 Select an existing custom IAM role that has the required permissions（选择一个具有所需权限的现有自定义 IAM 角色）以使用您之前创建的可选自定义角色。此角色必须具有指定 `"Service": "ssm.amazonaws.com"` 的信任关系策略。如果您的 IAM 角色未在信任关系策略中指定此原则，则会收到以下错误：

     ```
     An error occurred (ValidationException) when calling the CreateActivation
                                        operation: Not existing role:
      arn:aws:iam::<accountid>:role/SSMRole
     ```

有关创建此角色的更多信息，请参阅 在混合和多云环境中创建 Systems Manager 所需的 IAM 服务角色。

7. 对于 Activation expiry date（激活到期日期），为该激活指定一个到期日期。到期日期必须是未来的某个日期，并且不能超出未来 30 天。默认值为 24 小时。

> **Note**
>
> 如果需要在过期日期后注册更多的托管式节点，您必须创建新的激活。过期日期对已注册且正在运行的节点没有任何影响。

8. （可选）在 Default instance name（默认实例名称）字段，指定要为与此激活关联的所有托管式节点显示的标识名称值。
9. 选择创建激活。Systems Manager 立即将激活码和 ID 返回到控制台。

使用命令行创建用于将托管式节点注册到 Systems Manager 的激活

以下过程介绍了如何使用 AWS Command Line Interface (AWS CLI)（在 Linux 或 Windows Server 上）或 AWS Tools for PowerShell 创建托管节点激活。

创建激活

1. 安装并配置 AWS CLI 或 AWS Tools for PowerShell（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本以及安装 AWS Tools for PowerShell。
2. 运行以下命令以创建一个激活。

> **Note**
>
> - 在下面的命令中，将 *region* 替换为您自己的信息。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。
> - 为 *iam-role* 参数指定的角色必须具有指定 `"Service": "ssm.amazonaws.com"` 的信任关系策略。如果您的 AWS Identity and Access Management (IAM) 角色未在信任关系策略中指定此原则，则会收到以下错误：
>
> ```
> An error occurred (ValidationException) when calling the CreateActivation
> ```

```
                                                operation: Not existing role:
arn:aws:iam::<accountid>:role/SSMRole
```

有关创建此角色的更多信息，请参阅 [在混合和多云环境中创建 Systems Manager 所需的 IAM 服务角色](#)。

- 对于 `--expiration-date`，请提供时间戳格式的日期（例如 `"2021-07-07T00:00:00"`），以指示激活码何时过期。您最多可以提前 30 天指定日期。如果您未提供到期日期，激活码将在 24 小时内过期。

Linux & macOS

```
aws ssm create-activation \
    --default-instance-name name \
    --iam-role iam-service-role-name \
    --registration-limit number-of-managed-instances \
    --region region \
    --expiration-date "timestamp" \\
    --tags "Key=key-name-1,Value=key-value-1" "Key=key-name-2,Value=key-value-2"
```

Windows

```
aws ssm create-activation ^
    --default-instance-name name ^
    --iam-role iam-service-role-name ^
    --registration-limit number-of-managed-instances ^
    --region region ^
    --expiration-date "timestamp" ^
    --tags "Key=key-name-1,Value=key-value-1" "Key=key-name-2,Value=key-value-2"
```

PowerShell

```
New-SSMActivation -DefaultInstanceName name `
    -IamRole iam-service-role-name `
    -RegistrationLimit number-of-managed-instances `
    -Region region `
    -ExpirationDate "timestamp" `
    -Tag @{"Key"="key-name-1";"Value"="key-value-1"},@{"Key"="key-
name-2";"Value"="key-value-2"}
```

见下列。

Linux & macOS

```
aws ssm create-activation \
    --default-instance-name MyWebServers \
    --iam-role service-role/AmazonEC2RunCommandRoleForManagedInstances \
    --registration-limit 10 \
    --region us-east-2 \
    --expiration-date "2021-07-07T00:00:00" \
    --tags "Key=Environment,Value=Production" "Key=Department,Value=Finance"
```

Windows

```
aws ssm create-activation ^
    --default-instance-name MyWebServers ^
    --iam-role service-role/AmazonEC2RunCommandRoleForManagedInstances ^
    --registration-limit 10 ^
    --region us-east-2 ^
    --expiration-date "2021-07-07T00:00:00" ^
    --tags "Key=Environment,Value=Production" "Key=Department,Value=Finance"
```

PowerShell

```
New-SSMActivation -DefaultInstanceName MyWebServers `
    -IamRole service-role/AmazonEC2RunCommandRoleForManagedInstances `
    -RegistrationLimit 10 `
    -Region us-east-2 `
    -ExpirationDate "2021-07-07T00:00:00" `
    -Tag
 @{"Key"="Environment";"Value"="Production"},@{"Key"="Department";"Value"="Finance"}
```

如果成功创建激活，系统将立即返回一个激活代码和 ID。

在混合 Linux 节点上安装 SSM Agent

本主题介绍如何在混合和多云环境中的非 EC2（Amazon Elastic Compute Cloud）Linux 计算机上安装 AWS Systems Manager SSM Agent。有关在适用于 Linux 的 EC2 实例上安装 SSM Agent 的信息，请参阅在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent。

开始之前，请找到在混合激活过程中生成的激活码和激活 ID，如 创建混合激活以将节点注册到 Systems Manager 中所述。按照以下流程指定激活代码和 ID。

在混合和多云环境中的非 EC2 计算机上安装 SSM Agent

1. 登录混合和多云环境中的服务器或虚拟机。
2. 如果使用 HTTP 或 HTTPS 代理，则必须在当前 Shell 会话中设置 `http_proxy` 或 `https_proxy` 环境变量。如果您不使用代理，则可跳过此步骤。

   对于 HTTP 代理服务器，请在命令行中输入以下命令：

   ```
   export http_proxy=http://hostname:port
   export https_proxy=http://hostname:port
   ```

   对于 HTTPS 代理服务器，请在命令行中输入以下命令：

   ```
   export http_proxy=http://hostname:port
   export https_proxy=https://hostname:port
   ```

3. 将以下命令块之一复制并粘贴到 SSH 中。将占位符的值替换为在混合激活过程中生成的激活码和激活 ID，以及要从中下载 SSM Agent 的 AWS 区域标识符，然后按 `Enter`。

   > **Important**
   >
   > 请注意以下重要详细信息：
   >
   > - `ssm-setup-cli` 用于非 EC2 安装可最大程度提高 Systems Manager 安装和配置的安全性。
   > - 如果您不是根用户，则 `sudo` 不是必需的。
   > - 从创建混合激活时所在的相同 AWS 区域 下载 `ssm-setup-cli`。
   > - `ssm-setup-cli` 支持 `manifest-url` 选项，该选项可用来确定代理的下载来源。除非您的组织要求，否则请勿为此选项指定任何值。

- 注册实例时，请仅使用为 ssm-setup-cli 提供的下载链接。ssm-setup-cli 不应另行存放以供将来使用。
- 您可以使用此处提供的脚本来验证 ssm-setup-cli 的签名。

region 表示 AWS Systems Manager 支持的 AWS 区域 的标识符，例如 us-east-2 对应美国东部（俄亥俄）区域。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

此外，ssm-setup-cli 包括以下选项：

- version –有效值为 latest 和 stable。
- downgrade – 允许将 SSM Agent 降级为早期版本。指定 true 安装早期版本的代理。
- skip-signature-validation – 在下载和安装代理期间跳过签名验证。

RHEL 6.x 和 CentOS 6.x

```
mkdir /tmp/ssm
curl https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/3.0.1479.0/linux_amd64/
amazon-ssm-agent.rpm -o /tmp/ssm/amazon-ssm-agent.rpm
sudo yum install -y /tmp/ssm/amazon-ssm-agent.rpm
sudo stop amazon-ssm-agent
sudo -E amazon-ssm-agent -register -code "activation-code" -id "activation-id" -region
 "region"
sudo start amazon-ssm-agent
```

Amazon Linux 1

```
mkdir /tmp/ssm
curl https://amazon-ssm-region.s3.region.amazonaws.com/latest/linux_amd64/ssm-setup-cli
 -o /tmp/ssm/ssm-setup-cli
sudo chmod +x /tmp/ssm/ssm-setup-cli
sudo /tmp/ssm/ssm-setup-cli -register -activation-code "activation-code" -id
 "activation-id" -region "region"
```

Amazon Linux 2、RHEL 7.x、Oracle Linux、CentOS 7.x 和 SLES

```
mkdir /tmp/ssm
```

```
curl https://amazon-ssm-region.s3.region.amazonaws.com/latest/linux_amd64/ssm-setup-cli
 -o /tmp/ssm/ssm-setup-cli
sudo chmod +x /tmp/ssm/ssm-setup-cli
sudo /tmp/ssm/ssm-setup-cli -register -activation-code "activation-code" -activation-id
 "activation-id" -region "region"
```

RHEL 8.x 和 CentOS 8.x

```
mkdir /tmp/ssm
curl https://amazon-ssm-region.s3.region.amazonaws.com/latest/linux_amd64/ssm-setup-cli
 -o /tmp/ssm/ssm-setup-cli
sudo chmod +x /tmp/ssm/ssm-setup-cli
sudo /tmp/ssm/ssm-setup-cli -register -activation-code "activation-code" -activation-id
 "activation-id" -region "region"
```

Debian Server

```
mkdir /tmp/ssm
curl https://amazon-ssm-region.s3.region.amazonaws.com/latest/debian_amd64/ssm-setup-
cli -o /tmp/ssm/ssm-setup-cli
sudo chmod +x /tmp/ssm/ssm-setup-cli
sudo /tmp/ssm/ssm-setup-cli -register -activation-code "activation-code" -activation-id
 "activation-id" -region "region"
```

Raspberry Pi OS（原 Raspbian）

```
mkdir /tmp/ssm
curl https://amazon-ssm-region.s3.region.amazonaws.com/latest/debian_arm/ssm-setup-cli
 -o /tmp/ssm/ssm-setup-cli
sudo chmod +x /tmp/ssm/ssm-setup-cli
sudo /tmp/ssm/ssm-setup-cli -register -activation-code "activation-code" -activation-id
 "activation-id" -region "region"
```

Ubuntu Server

- 使用 .deb 包

  ```
  mkdir /tmp/ssm
  curl https://amazon-ssm-region.s3.region.amazonaws.com/latest/debian_amd64/ssm-setup-
  cli -o /tmp/ssm/ssm-setup-cli
  sudo chmod +x /tmp/ssm/ssm-setup-cli
  ```

```
sudo /tmp/ssm/ssm-setup-cli -register -activation-code "activation-code" -activation-
id "activation-id" -region "region"
```

- 使用 Snap 包

  您无需为下载指定 URL，因为 snap 命令会自动从 Snap 应用商店 https://snapcraft.io 下载代理。

  在 Ubuntu Server 20.10 STR & 20.04、 18.04 和 16.04 LTS 上，SSM Agent 安装程序文件（包括代理二进制文件和配置文件）均存储在以下目录中：/snap/amazon-ssm-agent/current/。如果要对此目录中的任何配置文件进行更改，则必须将这些文件从 /snap 目录复制到 /etc/amazon/ssm/ 目录。日志和库文件未更改（/var/lib/amazon/ssm、/var/log/amazon/ssm）。

  ```
  sudo snap install amazon-ssm-agent --classic
  sudo systemctl stop snap.amazon-ssm-agent.amazon-ssm-agent.service
  sudo /snap/amazon-ssm-agent/current/amazon-ssm-agent -register -code "activation-
  code" -id "activation-id" -region "region"
  sudo systemctl start snap.amazon-ssm-agent.amazon-ssm-agent.service
  ```

  Important

  Snap Store 中的 candidate 通道包含最新版本的 SSM Agent；而 stable 通道中未包含。如果要跟踪 candidate 通道上的 SSM Agent 版本信息，请在您的 Ubuntu Server 18.04 和 16.04 LTS 64 位托管式节点上运行以下命令。

  ```
  sudo snap switch --channel=candidate amazon-ssm-agent
  ```

此命令将 SSM Agent 下载并安装到混合和多云环境中的混合激活计算机上。此命令会停止 SSM Agent，然后向 Systems Manager 服务注册计算机。该计算机现在已是托管节点。为 Systems Manager 配置的 Amazon EC2 实例也是托管式节点。但在 Systems Manager 控制台中，您的混合激活节点与具有前缀“mi-”的 Amazon EC2 实例有区别。

继续在混合 Windows Server 节点上安装 SSM Agent。

设置私有密钥自动轮换

为了加强安全状况，您可以将 AWS Systems Manager 代理（SSM Agent）配置为自动轮换混合和多云环境私有密钥。您可以使用 SSM Agent 版本 3.0.1031.0 或更高版本来访问此功能。使用以下过程开启此功能。

将 SSM Agent 配置为轮换混合和多云环境私有密钥

1. 在 Linux 计算机上导航到 `/etc/amazon/ssm/`，或者在 Windows 计算机上导航到 `C:\Program Files\Amazon\SSM`。
2. 将 `amazon-ssm-agent.json.template` 的内容复制到名为 `amazon-ssm-agent.json` 的新文件中。Save`amazon-ssm-agent.json`在同一目录中，其中`amazon-ssm-agent.json.template`位于中。
3. 找到 `Profile`、`KeyAutoRotateDays`。输入您希望自动轮换私有密钥的间隔天数。
4. 重新启动 SSM Agent。

每次更改配置时，请重新启动 SSM Agent。

您可以按照相同的过程自定义 SSM Agent 的其他功能。有关可用配置属性及其默认值的最新列表，请参阅配置属性定义。

取消注册及重新注册托管式节点（Linux）

您可以从 AWS CLI 或借助适用于 Windows PowerShell 的工具来调用 DeregisterManagedInstance API 操作，以取消注册混合激活托管节点。以下是一个示例 CLI 命令：

```
aws ssm deregister-managed-instance --instance-id "mi-1234567890"
```

要移除代理的其余注册信息，请移除 `amazon-ssm-agent.json` 文件中的 `IdentityConsumptionOrder` 键。然后，根据安装类型运行下列命令之一。

在使用 Snap 软件包安装了 SSM Agent 的 Ubuntu Server 节点上：

```
sudo /snap/amazon-ssm-agent/current/amazon-ssm-agent -register -clear
```

在其余 Linux 安装上：

```
amazon-ssm-agent -register -clear
```

在 非 EC2 Linux 计算机上重新注册托管节点

您可以在取消注册后重新注册计算机。但是，必须使用不同的激活码和激活 ID，不能使用先前注册计算机时用过的激活码和激活 ID。

1. 连接到您的计算机。
2. 运行以下命令。请务必将占位符的值替换为在创建托管节点激活时生成的激活码和激活 ID，以及要从中下载 SSM Agent 的区域标识符。

```
echo "yes" | sudo /tmp/ssm/ssm-setup-cli -register -activation-code "activation-code" -activation-id "activation-id" -region "region
```

对非 EC2 Linux 计算机上的 SSM Agent 安装进行故障排除

使用以下信息来帮助您解决混合激活 Linux 计算机在混合和多云环境中安装 SSM Agent 时出现的问题。

收到 DeliveryTimedOut 错误

问题：将一个 AWS 账户 中的计算机配置为单独 AWS 账户 的托管节点时，运行在目标计算机上安装 SSM Agent 的命令后，您将收到 `DeliveryTimedOut`。

解决方案：在此场景中，`DeliveryTimedOut` 是预期响应代码。在目标节点上安装 SSM Agent 的命令会更改源节点的节点 ID。由于节点 ID 已更改，源节点无法响应命令执行时失败、已完成或超时的目标节点。

无法加载节点关联

问题：运行安装命令后，SSM Agent 错误日志中显示以下错误：

```
Unable to load instance associations, unable to retrieve
associations unable to retrieve associations error occurred in
RequestManagedInstanceRoleToken: MachineFingerprintDoesNotMatch:
Fingerprint doesn't match
```

当计算机 ID 在重启后不存在时，便会显示此错误。

解决方案：要解决此问题，请运行以下命令。此命令强制计算机 ID 在重启后保持不变。

```
umount /etc/machine-id
systemd-machine-id-setup
```

在混合 Windows Server 节点上安装 SSM Agent

本主题介绍如何在混合和多云环境中的 Windows Server 计算机上安装 AWS Systems Manager SSM Agent。有关在适用于 Windows Server 的 EC2 实例上安装 SSM Agent 的信息，请参阅在适用于 Windows Server 的 EC2 实例上手动安装和卸载 SSM Agent。

开始之前，请找到在混合激活过程中生成的激活码和激活 ID，如 创建混合激活以将节点注册到 Systems Manager 中所述。按照以下流程指定激活代码和 ID。

在混合和多云环境中的非 EC2 Windows Server 计算机上安装 SSM Agent

1. 登录混合和多云环境中的服务器或虚拟机。
2. 如果使用 HTTP 或 HTTPS 代理，则必须在当前 Shell 会话中设置 `http_proxy` 或 `https_proxy` 环境变量。如果您不使用代理，则可跳过此步骤。

   对于 HTTP 代理服务器，请设置以下变量：

   ```
   http_proxy=http://hostname:port
   https_proxy=http://hostname:port
   ```

   对于 HTTPS 代理服务器，请设置以下变量：

   ```
   http_proxy=http://hostname:port
   https_proxy=https://hostname:port
   ```
3. 在提升（管理）模式下打开 Windows PowerShell。
4. 将以下命令块复制并粘贴到 Windows PowerShell 中。将每个*#######*替换为您自己的信息。例如，在创建混合激活时生成的激活代码和激活 ID，以及带有您要从中下载 SSM Agent 的 AWS 区域 标识符的激活代码和激活 ID。

   > **Important**
   >
   > 请注意以下重要详细信息：
   >
   > - `ssm-setup-cli` 用于非 EC2 安装可最大程度提高 Systems Manager 安装和配置的安全性。
   > - `ssm-setup-cli` 支持 `manifest-url` 选项，该选项可用来确定代理的下载来源。除非您的组织要求，否则请勿为此选项指定任何值。
   > - 您可以使用此处提供的脚本来验证 `ssm-setup-cli` 的签名。

- 注册实例时，请仅使用为 ssm-setup-cli 提供的下载链接。ssm-setup-cli 不应另行存放以供将来使用。

region 表示 AWS Systems Manager 支持的 AWS 区域 的标识符，例如 us-east-2 对应美国东部（俄亥俄）区域。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

此外，ssm-setup-cli 包括以下选项：

- version –有效值为 latest 和 stable。
- downgrade – 将代理恢复到早期版本。
- skip-signature-validation – 在下载和安装代理期间跳过签名验证。

64-bit

```
[System.Net.ServicePointManager]::SecurityProtocol = 'TLS12'
$code = "activation-code"
$id = "activation-id"
$region = "us-east-1"
$dir = $env:TEMP + "\ssm"
New-Item -ItemType directory -Path $dir -Force
cd $dir
(New-Object System.Net.WebClient).DownloadFile("https://amazon-ssm-$region.s3.
$region.amazonaws.com/latest/windows_amd64/ssm-setup-cli.exe", $dir + "\ssm-
setup-cli.exe")
./ssm-setup-cli.exe -register -activation-code="$code" -activation-id="$id" -
region="$region"
Get-Content ($env:ProgramData + "\Amazon\SSM\InstanceData\registration")
Get-Service -Name "AmazonSSMAgent"
```

32-bit

```
"[System.Net.ServicePointManager]::SecurityProtocol = 'TLS12'"
$code = "activation-code"
$id = "activation-id"
$region = "us-east-1"
$dir = $env:TEMP + "\ssm"
New-Item -ItemType directory -Path $dir -Force
cd $dir
```

```
(New-Object System.Net.WebClient).DownloadFile("https://amazon-ssm-$region.s3.
$region.amazonaws.com/latest/windows_386/ssm-setup-cli.exe", $dir + "\ssm-setup-
cli.exe")
./ssm-setup-cli.exe -register -activation-code="$code" -activation-id="$id" -
region="$region"
Get-Content ($env:ProgramData + "\Amazon\SSM\InstanceData\registration")
Get-Service -Name "AmazonSSMAgent"
```

5. 按 Enter。

> **Note**
>
> 如果命令失败，确认运行的是否是最新版本的 AWS Tools for PowerShell。

此命令执行以下操作：

- 下载并在计算机上安装 SSM Agent。
- 向 Systems Manager 服务注册计算机。
- 返回类似以下内容的请求响应：

```
    Directory: C:\Users\ADMINI~1\AppData\Local\Temp\2


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----       07/07/2018   8:07 PM                ssm
{"ManagedInstanceID":"mi-008d36be46EXAMPLE","Region":"us-east-2"}

Status      : Running
Name        : AmazonSSMAgent
DisplayName : Amazon SSM Agent
```

该计算机现在已是托管节点。现在，这些托管式节点带有前缀“mi-”作为标识。您可以使用 AWS CLI 命令 describe-instance-information 或使用 API 命令 DescribeInstanceInformation，在 Fleet Manager 中的托管式节点页面上查看托管式节点。

设置私有密钥自动轮换

为了加强安全状况，您可以将 AWS Systems Manager 代理（SSM Agent）配置为自动轮换混合和多云环境私有密钥。您可以使用 SSM Agent 版本 3.0.1031.0 或更高版本来访问此功能。使用以下过程开启此功能。

将 SSM Agent 配置为轮换混合和多云环境私有密钥

1. 在 Linux 计算机上导航到 `/etc/amazon/ssm/`，或者在 Windows Server 计算机上导航到 `C:\Program Files\Amazon\SSM`。
2. 将 `amazon-ssm-agent.json.template` 的内容复制到名为 `amazon-ssm-agent.json` 的新文件中。Save`amazon-ssm-agent.json`在同一目录中，其中`amazon-ssm-agent.json.template`位于中。
3. 找到 `Profile`、`KeyAutoRotateDays`。输入您希望自动轮换私有密钥的间隔天数。
4. 重新启动 SSM Agent。

每次更改配置时，请重新启动 SSM Agent。

您可以按照相同的过程自定义 SSM Agent 的其他功能。有关可用配置属性及其默认值的最新列表，请参阅配置属性定义。

取消注册及重新注册托管式节点（Windows Server）

您可以从 AWS CLI 或借助适用于 Windows PowerShell 的工具来调用 DeregisterManagedInstance API 操作，以取消注册托管节点。以下是一个示例 CLI 命令：

```
aws ssm deregister-managed-instance --instance-id "mi-1234567890"
```

要移除代理的其余注册信息，请移除 `amazon-ssm-agent.json` 文件中的 `IdentityConsumptionOrder` 键。然后运行以下命令：

```
amazon-ssm-agent -register -clear
```

在 Windows Server 混合计算机上重新注册托管节点

您可以在取消注册后重新注册计算机。但是，必须使用不同的激活码和激活 ID，不能使用先前注册计算机时用过的激活码和激活 ID。

1. 连接到您的计算机。

2. 运行以下命令。请务必将占位符的值替换为在创建混合激活时生成的激活码和激活 ID，以及要从中下载 SSM Agent 的区域标识符。

```
'yes' | & Start-Process ./ssm-setup-cli.exe -ArgumentList @("-register", "-
activation-code=$code", "-activation-id=$id", "-region=$region") -Wait
Get-Content ($env:ProgramData + "\Amazon\SSM\InstanceData\registration")
Get-Service -Name "AmazonSSMAgent"
```

使用 Systems Manager 管理边缘设备

本节介绍账户和系统管理员为启用 AWS IoT Greengrass 核心设备的配置和管理而执行的设置任务。完成这些任务后，已获得 AWS 账户 管理员所授予权限的用户可使用 AWS Systems Manager 来配置及管理其企业的 AWS IoT Greengrass 核心设备。

> **Note**
>
> - macOS 和 Windows 10 不支持适用于 AWS IoT Greengrass 的 SSM Agent。您不能使用 Systems Manager 工具来管理及配置使用这两种操作系统的边缘设备。
> - Systems Manager 还支持未配置为 AWS IoT Greengrass 核心设备的边缘设备。要使用 Systems Manager 管理 AWS IoT 核心设备和非 AWS 边缘设备，必须使用混合激活来对其进行配置。有关更多信息，请参阅 使用 Systems Manager 管理混合和多云环境中的节点。
> - 要将 Session Manager 和 Microsoft 应用程序补丁与您的边缘设备搭配使用，必须启用高级实例套餐。有关更多信息，请参阅 打开高级实例套餐。

开始前的准备工作

确认您的边缘设备满足以下要求。

- 边缘设备必须满足将其配置为 AWS IoT Greengrass 核心设备的要求。有关更多信息，请参阅 AWS IoT Greengrass 开发人员指南中的设置 AWS IoT Greengrass Version 2 核心设备。
- 边缘设备必须与 AWS Systems Manager Agent (SSM Agent) 兼容。有关更多信息，请参阅 Systems Manager 支持的操作系统。
- 边缘设备必须能够与云中的 Systems Manager 服务通信。Systems Manager 不支持已断开连接的边缘设备。

关于设置边缘设备

为 Systems Manager 设置 AWS IoT Greengrass 设备涉及以下过程。

> **Note**
>
> 有关从边缘设备中卸载 SSM Agent 的信息，请参阅《AWS IoT Greengrass Version 2 开发人员指南》中的卸载 AWS Systems Manager Agent。

为您的边缘设备创建 IAM 服务角色

AWS IoT Greengrass 核心设备需要 AWS Identity and Access Management (IAM) 服务角色才能与 AWS Systems Manager 通信。此角色向 AWS Security Token Service (AWS STS) AssumeRole 授予对 Systems Manager 服务的信任。您只需为每个 AWS 账户 创建此服务角色一次。在为 AWS IoT Greengrass 设备配置及部署 SSM Agent 组件时，您需为 `RegistrationRole` 参数指定此角色。如果在混合和多云环境中设置非 EC2 节点时已创建此角色，则可以跳过此步骤。

> **Note**
>
> 贵公司或企业中如果有用户要在边缘设备上使用 Systems Manager，则必须在 IAM 中为其授予调用 Systems Manager API 的权限。

S3 存储桶策略要求

如果出现以下任一情况，您必须在完成此过程之前为 Amazon Simple Storage Service (Amazon S3) 存储桶创建自定义 IAM 权限策略：

- 情况 1：您通过一个 VPC 端点以私有方式将您的 VPC 连接到受支持的 AWS 服务，以及由 AWS PrivateLink 提供支持的 VPC 端点服务。
- 情况 2：您打算使用在 Systems Manager 操作期间创建的 S3 存储桶，如将 Run Command 命令或 Session Manager 会话的输出存储到 S3 存储桶中。在继续之前，请按照为实例配置文件创建自定义 S3 存储桶策略中的步骤进行操作。该主题中有关 S3 存储桶策略的信息也适用于服务角色。

  > **Note**
  >
  > 如果您的设备受防火墙保护且您计划使用 Patch Manager，则防火墙必须允许对补丁基准端点 `arn:aws:s3:::patch-baseline-snapshot-region/*` 进行访问。

region 表示 AWS Systems Manager 支持的 AWS 区域 的标识符，例如 us-east-2 对应美国东部（俄亥俄）区域。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

AWS CLI

为 AWS IoT Greengrass 环境创建 IAM 服务角色 (AWS CLI)

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 在本地计算机上使用以下信任策略创建一个文本文件，文件名称类似于 SSMService-Trust.json。确保使用 .json 文件扩展名保存该文件。

   > **Note**
   >
   > 记录下名称。您需在将 SSM Agent 部署到 AWS IoT Greengrass 核心设备时指定此名称。

   JSON

   ```
   {
       "Version": "2012-10-17",
       "Statement": {
           "Effect": "Allow",
           "Principal": {
               "Service": "ssm.amazonaws.com"
           },
           "Action": "sts:AssumeRole"
       }
   }
   ```

3. 打开 AWS CLI，在创建该 JSON 文件的目录下，运行 create-role 命令以创建服务角色。将每个*#######*替换为您自己的信息。

   Linux 和 macOS

```
aws iam create-role \
    --role-name SSMServiceRole \
    --assume-role-policy-document file://SSMService-Trust.json
```

Windows

```
aws iam create-role ^
    --role-name SSMServiceRole ^
    --assume-role-policy-document file://SSMService-Trust.json
```

4. 按如下方式运行 attach-role-policy 命令，以允许您刚创建的服务角色创建会话令牌。该会话令牌向您的边缘设备授予使用 Systems Manager 运行命令的权限。

> **Note**
>
> 您为边缘设备的服务配置文件添加的策略与用于为 Amazon Elastic Compute Cloud (Amazon EC2) 实例创建实例配置文件的策略相同。有关以下命令中使用的 IAM 策略的更多信息，请参阅配置 Systems Manager 所需的实例权限。

（必需）运行以下命令以允许边缘设备使用 AWS Systems Manager 服务核心功能。

Linux 和 macOS

```
aws iam attach-role-policy \
    --role-name SSMServiceRole \
    --policy-arn arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
```

Windows

```
aws iam attach-role-policy ^
    --role-name SSMServiceRole ^
    --policy-arn arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
```

如果为服务角色创建了自定义 S3 存储桶策略，请运行以下命令，以允许 AWS Systems Manager Agent (SSM Agent) 访问在该策略中指定的存储桶。将 *account_ID* 和 *my_bucket_policy_name* 替换为您的 AWS 账户 ID 和存储桶名称。

Linux 和 macOS

```
aws iam attach-role-policy \
    --role-name SSMServiceRole \
    --policy-arn arn:aws:iam::account_ID:policy/my_bucket_policy_name
```

Windows

```
aws iam attach-role-policy ^
    --role-name SSMServiceRole ^
    --policy-arn arn:aws:iam::account_id:policy/my_bucket_policy_name
```

（可选）运行以下命令以允许 SSM Agent 代表您访问 AWS Directory Service，以处理边缘设备加入域的请求。只有在将边缘设备加入 Microsoft AD 目录时，服务角色才需要使用此策略。

Linux 和 macOS

```
aws iam attach-role-policy \
    --role-name SSMServiceRole \
    --policy-arn arn:aws:iam::aws:policy/AmazonSSMDirectoryServiceAccess
```

Windows

```
aws iam attach-role-policy ^
    --role-name SSMServiceRole ^
    --policy-arn arn:aws:iam::aws:policy/AmazonSSMDirectoryServiceAccess
```

（可选）运行以下命令以允许 CloudWatch 代理在边缘设备上运行。通过使用此命令，可以读取设备上的信息并将其写入 CloudWatch。仅当您要使用 Amazon EventBridge 或 Amazon CloudWatch Logs 等服务时，服务角色才需要使用此策略。

```
aws iam attach-role-policy \
    --role-name SSMServiceRole \
    --policy-arn arn:aws:iam::aws:policy/CloudWatchAgentServerPolicy
```

Tools for PowerShell

为 AWS IoT Greengrass 环境创建 IAM 服务角色 (AWS Tools for Windows PowerShell)

1. 如果您尚未安装和配置 AWS Tools for PowerShell（适用于 Windows PowerShell 的工具），请执行这些操作。

   有关信息，请参阅安装 AWS Tools for PowerShell。

2. 在本地计算机上使用以下信任策略创建一个文本文件，文件名称类似于 `SSMService-Trust.json`。确保使用 `.json` 文件扩展名保存该文件。

   > **Note**
   >
   > 记录下名称。您需在将 SSM Agent 部署到 AWS IoT Greengrass 核心设备时指定此名称。

   JSON

   ```
   {
       "Version": "2012-10-17",
       "Statement": {
           "Effect": "Allow",
           "Principal": {
               "Service": "ssm.amazonaws.com"
           },
           "Action": "sts:AssumeRole"
       }
   }
   ```

3. 在管理模式下打开 PowerShell，在创建此 JSON 文件的目录中，按如下方式运行 New-IAMRole 以创建服务角色。

   ```
   New-IAMRole `
       -RoleName SSMServiceRole `
       -AssumeRolePolicyDocument (Get-Content -raw SSMService-Trust.json)
   ```

4. 按如下方式使用 Register-IAMRolePolicy，以允许您创建的服务角色创建会话令牌。该会话令牌向您的边缘设备授予使用 Systems Manager 运行命令的权限。

> **Note**
>
> 您为 AWS IoT Greengrass 环境中边缘设备的服务角色添加的策略与用于为 EC2 实例创建实例配置文件的策略相同。有关以下命令中使用的 AWS 策略的更多信息，请参阅配置 Systems Manager 所需的实例权限。

（必需）运行以下命令以允许边缘设备使用 AWS Systems Manager 服务核心功能。

```
Register-IAMRolePolicy `
    -RoleName SSMServiceRole `
    -PolicyArn arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
```

如果为服务角色创建了自定义 S3 存储桶策略，请运行以下命令，以允许 SSM Agent 访问在该策略中指定的存储桶。将 *account_ID* 和 *my_bucket_policy_name* 替换为您的 AWS 账户 ID 和存储桶名称。

```
Register-IAMRolePolicy `
    -RoleName SSMServiceRole `
    -PolicyArn arn:aws:iam::account_ID:policy/my_bucket_policy_name
```

（可选）运行以下命令以允许 SSM Agent 代表您访问 AWS Directory Service，以处理边缘设备加入域的请求。只有在将边缘设备加入 Microsoft AD 目录时，服务角色才需要使用此策略。

```
Register-IAMRolePolicy `
    -RoleName SSMServiceRole `
    -PolicyArn arn:aws:iam::aws:policy/AmazonSSMDirectoryServiceAccess
```

（可选）运行以下命令以允许 CloudWatch 代理在边缘设备上运行。通过使用此命令，可以读取设备上的信息并将其写入 CloudWatch。仅当您要使用 Amazon EventBridge 或 Amazon CloudWatch Logs 等服务时，服务角色才需要使用此策略。

```
Register-IAMRolePolicy `
    -RoleName SSMServiceRole `
    -PolicyArn arn:aws:iam::aws:policy/CloudWatchAgentServerPolicy
```

配置 AWS IoT Greengrass 的边缘设备

将边缘设备设置为 AWS IoT Greengrass 核心设备。设置过程包括确认支持的操作系统和系统要求，以及安装和配置设备上的 AWS IoT Greengrass Core 软件。有关更多信息，请参阅 AWS IoT Greengrass 开发人员指南中的设置 AWS IoT Greengrass Version 2 核心设备。

更新 AWS IoT Greengrass 令牌交换角色并在边缘设备上安装 SSM Agent

在为 Systems Manager 设置和配置 AWS IoT Greengrass 核心设备的最后一步中，您需要更新 AWS IoT Greengrass AWS Identity and Access Management (IAM) 设备服务角色（也称为令牌交换角色），以及将 AWS Systems Manager Agent (SSM Agent) 部署到 AWS IoT Greengrass 设备。有关这些过程的信息，请参阅《AWS IoT Greengrass Version 2 Developer Guide》中的 Install the AWS Systems Manager Agent。

将 SSM Agent 部署到设备后，AWS IoT Greengrass 会自动向 Systems Manager 注册您的设备。除此之外，无需进行其他注册。您可以开始使用 Systems Manager 工具来访问、管理和配置 AWS IoT Greengrass 设备。

> **Note**
>
> 边缘设备必须能够与云中的 Systems Manager 服务通信。Systems Manager 不支持已断开连接的边缘设备。

为 Systems Manager 创建 AWS Organizations 委派管理员

在 AWS Organizations 中设置组织时，您可以分配管理账户来执行所有 AWS 服务 的所有管理任务。管理账户用户只能为 Systems Manager 分配委派管理员账户，以执行 Change Manager、Explorer 和 OpsCenter 的管理任务。AWS Organizations 是一项账户管理服务，您可以用它来创建组织并分配 AWS 账户 以集中管理这些账户。有关 AWS Organizations 的信息，请参阅《AWS Organizations 用户指南》中的 AWS Organizations。

Change Manager、Explorer 和 OpsCenter（都是 AWS Systems Manager 中的工具）与 AWS Organizations 配合使用，可在组织的所有成员账户上执行任务。您只能为所有 Systems Manager 工具分配一名委派管理员。委派管理员账户必须是其分配到的组织的成员。

主题

- 对 Change Manager 使用委派管理员

- 对 Explorer 使用委派管理员
- 对 OpsCenter 使用委派管理员
- 对 Quick Setup 使用委派管理员

对 Change Manager 使用委派管理员

Change Manager 是一个企业变更管理框架，用于请求、批准、实施和报告应用程序配置和基础设施的操作变更。

如果您在整个组织中使用 Change Manager，请分配委派管理员账户来管理所有成员账户的变更模板、批准和报告。使用 Quick Setup 功能，您可以将 Change Manager 设置为与组织一起使用并选择委派管理员账户。如果您将 Change Manager 与单个 AWS 账户 一起使用，则不需要委派管理员账户。

默认情况下，Change Manager 显示委派管理员账户中所有与变更相关的任务。有关在为组织设置 Change Manager 时配置委派管理员的说明，请参阅为组织设置 Change Manager（管理账户）。

> **Important**
>
> 如果您在整个组织中使用 Change Manager，我们建议始终从委托管理员账户进行更改。虽然您可以从组织中的其他账户进行更改，但这些更改不会在委派管理员账户中报告，也无法从该账户查看。

对 Explorer 使用委派管理员

Explorer 是一个可自定义的运营控制面板，用于报告跨 AWS 区域 的 AWS 账户 运营数据（OpsData）的聚合视图。

您可以为 Systems Manager 配置委派管理员账户，以通过将资源数据同步与 AWS Organizations 结合使用来聚合多个区域和账户的 Explorer 数据。委派管理员可以使用 AWS Management Console、AWS Command Line Interface（AWS CLI）或 AWS Tools for Windows PowerShell 搜索、筛选和聚合 Explorer 数据。

您为 Explorer 使用委派管理员账户时，您可以将管理员数量限制为一个 AWS 账户，只有此管理员可以创建或删除多账户和区域资源。

您可以使用 Explorer，跨组织里的所有 AWS 账户 同步操作数据。有关如何从 Explorer 中分配委派管理员的信息，请参阅 为 Explorer 配置委派管理员。

对 OpsCenter 使用委派管理员

OpsCenter 是提供了一个中心位置，运营工程师和 IT 专业人员可在此处管理与 AWS 资源相关的运营工作项（OpsItems）。如果您想使用 OpsCenter 跨账户集中管理 OpsItems，则必须在 AWS Organizations 中设置组织。

使用 OpsCenter 的 Quick Setup，您可以分配委派管理员账户并配置 OpsCenter 来集中管理 OpsItems。有关更多信息，请参阅 （可选）使用 Quick Setup 配置 OpsCenter 以跨账户管理 OpsItems。

对 Quick Setup 使用委派管理员

Quick Setup是 Systems Manager 中的一项工具，可帮助您使用推荐的最佳实践快速配置常用的 AWS 服务和功能。您可以为Quick Setup配置委派管理员账户，以帮助您使用 AWS Organizations 跨账户和区域部署和管理配置。Quick Setup的委派管理员可以在您的组织中创建、更新、查看和删除配置管理器资源。作为集成控制台体验设置过程的一部分，Systems Manager 会为Quick Setup注册一个委派管理员。有关更多信息，请参阅 为组织设置 Systems Manager 统一控制台。

AWS Systems Manager 的常规设置

如果尚未进行此操作，请注册 AWS 账户 并创建管理用户。

注册 AWS 账户

如果您还没有 AWS 账户，请完成以下步骤来创建一个。

注册 AWS 账户

1. 打开 https://portal.aws.amazon.com/billing/signup。
2. 按照屏幕上的说明操作。

   在注册时，将接到电话或收到短信，要求使用电话键盘输入一个验证码。

   当您注册 AWS 账户 时，系统将会创建一个 AWS 账户根用户。根用户有权访问该账户中的所有 AWS 服务和资源。作为最佳安全实践，请为用户分配管理访问权限，并且只使用根用户来执行需要根用户访问权限的任务。

注册过程完成后，AWS 会向您发送一封确认电子邮件。在任何时候，您都可以通过转至 https://aws.amazon.com/ 并选择我的账户来查看当前的账户活动并管理您的账户。

创建具有管理访问权限的用户

注册 AWS 账户 后，请保护好您的 AWS 账户根用户，启用 AWS IAM Identity Center，并创建一个管理用户，以避免使用根用户执行日常任务。

保护您的 AWS 账户根用户

1. 选择根用户并输入您的 AWS 账户 电子邮件地址，以账户拥有者身份登录 AWS Management Console。在下一页上，输入您的密码。

   要获取使用根用户登录方面的帮助，请参阅《AWS 登录 用户指南》中的 Signing in as the root user。
2. 为您的根用户启用多重身份验证（MFA）。

   有关说明，请参阅《IAM 用户指南》中的为 AWS 账户 根用户启用虚拟 MFA 设备（控制台）。

创建具有管理访问权限的用户

1. 启用 IAM Identity Center。

   有关说明，请参阅《AWS IAM Identity Center 用户指南》中的 Enabling AWS IAM Identity Center。
2. 在 IAM Identity Center 中，为用户授予管理访问权限。

   有关如何使用 IAM Identity Center 目录 作为身份源的教程，请参阅《AWS IAM Identity Center 用户指南》中的 Configure user access with the default IAM Identity Center 目录。

以具有管理访问权限的用户身份登录

- 要使用您的 IAM Identity Center 用户身份登录，请使用您在创建 IAM Identity Center 用户时发送到您的电子邮件地址的登录网址。

  要获取使用 IAM Identity Center 用户登录方面的帮助，请参阅《AWS 登录 用户指南》中的 Signing in to the AWS access portal。

将访问权限分配给其他用户

1. 在 IAM Identity Center 中，创建一个权限集，该权限集遵循应用最低权限的最佳做法。

有关说明，请参阅《AWS IAM Identity Center 用户指南》中的 Create a permission set。

2. 将用户分配到一个组，然后为该组分配单点登录访问权限。

   有关说明，请参阅《AWS IAM Identity Center 用户指南》中的 Add groups。

设置 AWS Systems Manager

以下主题介绍了如何为 AWS Organizations 组织和单个 AWS 账户设置统一 AWS Systems Manager 控制台。

主题

- 设置 Systems Manager 控制台访问
- 为组织设置 Systems Manager 统一控制台
- 为单个账户和区域设置 Systems Manager 统一控制台
- 禁用 Systems Manager 统一控制台

设置 Systems Manager 控制台访问

要 AWS Systems Manager在中使用 AWS Management Console，您必须配置正确的权限。

有关如何创建 AWS Identity and Access Management 策略并将其附加到 IAM 身份的更多信息，请参阅《IAM 用户指南》中的创建 IAM 策略

Systems Manager 载入策略

您可以创建一个如以下示例所示的 IAM 策略，并将该策略附加到您的 IAM 身份。此策略授予对载入到 Systems Manager 并对其进行配置的完全访问权限。

权限详细信息

该策略包含以下权限。

- `ssm-quicksetup`：允许主体访问所有 AWS Systems Manager 快速设置功能 操作。
- `ssm`：允许主体访问 Systems Manager Automation 和资源探索器。
- `organizations`：允许主体读取 AWS Organizations 中组织的结构，并在委派管理员以组织身份载入 Systems Manager 时管理委派管理员。
- `cloudformation`：允许主体管理其Quick Setup堆栈。
- `iam`：允许主体管理 Systems Manager 载入所需的 IAM 角色和策略。

JSON

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

     {
       "Sid": "SsmReadOnly",
       "Effect": "Allow",
       "Action": [
         "ssm:DescribeAutomationExecutions",
         "ssm:GetAutomationExecution",
         "ssm:ListAssociations",
         "ssm:DescribeAssociation",
         "ssm:ListDocuments",
         "ssm:ListResourceDataSync",
         "ssm:DescribePatchBaselines",
         "ssm:GetPatchBaseline",
         "ssm:DescribeMaintenanceWindows",
         "ssm:DescribeMaintenanceWindowTasks"
       ],
       "Resource": "*"
     },
     {
       "Sid": "SsmDocument",
       "Effect": "Allow",
       "Action": [
         "ssm:GetDocument",
         "ssm:DescribeDocument"
       ],
       "Resource": [
         "arn:aws:ssm:*:*:document/AWSQuickSetupType-*",
         "arn:aws:ssm:*:*:document/AWS-EnableExplorer"
       ]
     },
```

```
    {
      "Sid": "SsmEnableExplorer",
      "Effect": "Allow",
      "Action": "ssm:StartAutomationExecution",
      "Resource": "arn:aws:ssm:*:*:automation-definition/AWS-EnableExplorer:*"
    },
    {
      "Sid": "SsmExplorerRds",
      "Effect": "Allow",
      "Action": [
        "ssm:GetOpsSummary",
        "ssm:CreateResourceDataSync",
        "ssm:UpdateResourceDataSync"
      ],
      "Resource": "arn:aws:ssm:*:*:resource-data-sync/AWS-QuickSetup-*"
    },
    {
      "Sid": "OrgsReadOnly",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListRoots",
        "organizations:ListParents",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:DescribeOrganizationalUnit",
        "organizations:ListAWSServiceAccessForOrganization"
      ],
      "Resource": "*"
    },
    {
      "Sid": "OrgsAdministration",
      "Effect": "Allow",
      "Action": [
        "organizations:EnableAWSServiceAccess",
        "organizations:RegisterDelegatedAdministrator",
        "organizations:DeregisterDelegatedAdministrator"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "organizations:ServicePrincipal": [
            "ssm.amazonaws.com",
```

```
            "ssm-quicksetup.amazonaws.com",
            "member.org.stacksets.cloudformation.amazonaws.com",
            "resource-explorer-2.amazonaws.com"
          ]
        }
      }
    },
    {
      "Sid": "CfnReadOnly",
      "Effect": "Allow",
      "Action": [
        "cloudformation:ListStacks",
        "cloudformation:DescribeStacks",
        "cloudformation:ListStackSets",
        "cloudformation:DescribeOrganizationsAccess"
      ],
      "Resource": "*"
    },
    {
      "Sid": "OrgCfnAccess",
      "Effect": "Allow",
      "Action": [
        "cloudformation:ActivateOrganizationsAccess"
      ],
      "Resource": "*"
    },
    {
      "Sid": "CfnStackActions",
      "Effect": "Allow",
      "Action": [
        "cloudformation:CreateStack",
        "cloudformation:DeleteStack",
        "cloudformation:DescribeStackResources",
        "cloudformation:DescribeStackEvents",
        "cloudformation:GetTemplate",
        "cloudformation:RollbackStack",
        "cloudformation:TagResource",
        "cloudformation:UntagResource",
        "cloudformation:UpdateStack"
      ],
      "Resource": [
        "arn:aws:cloudformation:*:*:stack/StackSet-AWS-QuickSetup-*",
        "arn:aws:cloudformation:*:*:stack/AWS-QuickSetup-*",
        "arn:aws:cloudformation:*:*:type/resource/*"
```

```
        ]
      },
      {
        "Sid": "CfnStackSetActions",
        "Effect": "Allow",
        "Action": [
          "cloudformation:CreateStackInstances",
          "cloudformation:CreateStackSet",
          "cloudformation:DeleteStackInstances",
          "cloudformation:DeleteStackSet",
          "cloudformation:DescribeStackInstance",
          "cloudformation:DetectStackSetDrift",
          "cloudformation:ListStackInstanceResourceDrifts",
          "cloudformation:DescribeStackSet",
          "cloudformation:DescribeStackSetOperation",
          "cloudformation:ListStackInstances",
          "cloudformation:ListStackSetOperations",
          "cloudformation:ListStackSetOperationResults",
          "cloudformation:TagResource",
          "cloudformation:UntagResource",
          "cloudformation:UpdateStackSet"
        ],
        "Resource": [
          "arn:aws:cloudformation:*:*:stackset/AWS-QuickSetup-*",
          "arn:aws:cloudformation:*:*:type/resource/*",
          "arn:aws:cloudformation:*:*:stackset-target/AWS-QuickSetup-*:*"
        ]
      },
      {
        "Sid": "ValidationReadonlyActions",
        "Effect": "Allow",
        "Action": [
          "iam:ListRoles",
          "iam:GetRole"
        ],
        "Resource": "*"
      },
      {
        "Sid": "IamRolesMgmt",
        "Effect": "Allow",
        "Action": [
          "iam:CreateRole",
          "iam:DeleteRole",
          "iam:GetRole",
```

```
        "iam:AttachRolePolicy",
        "iam:DetachRolePolicy",
        "iam:GetRolePolicy",
        "iam:ListRolePolicies"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AWS-QuickSetup-*",
        "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
      ]
    },
    {
      "Sid": "IamPassRole",
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AWS-QuickSetup-*",
        "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
      ],
      "Condition": {
        "StringEquals": {
          "iam:PassedToService": [
            "ssm.amazonaws.com",
            "ssm-quicksetup.amazonaws.com",
            "cloudformation.amazonaws.com"
          ]
        }
      }
    },
    {
      "Sid": "IamRolesPoliciesMgmt",
      "Effect": "Allow",
      "Action": [
        "iam:AttachRolePolicy",
        "iam:DetachRolePolicy"
      ],
      "Resource": [
        "arn:aws:iam::*:role/AWS-QuickSetup-*",
        "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
      ],
      "Condition": {
        "ArnEquals": {
          "iam:PolicyARN": [
```

```
            "arn:aws:iam::aws:policy/
AWSSystemsManagerEnableExplorerExecutionPolicy",
            "arn:aws:iam::aws:policy/AWSQuickSetupSSMDeploymentRolePolicy"
          ]
        }
      }
    },
    {
      "Sid": "CfnStackSetsSLR",
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": [
        "arn:aws:iam::*:role/aws-service-role/
stacksets.cloudformation.amazonaws.com/
AWSServiceRoleForCloudFormationStackSetsOrgAdmin",
        "arn:aws:iam::*:role/aws-service-role/ssm.amazonaws.com/
AWSServiceRoleForAmazonSSM",
        "arn:aws:iam::*:role/aws-service-role/
accountdiscovery.ssm.amazonaws.com/AWSServiceRoleForAmazonSSM_AccountDiscovery",
        "arn:aws:iam::*:role/aws-service-role/ssm-quicksetup.amazonaws.com/
AWSServiceRoleForSSMQuickSetup",
        "arn:aws:iam::*:role/aws-service-role/resource-explorer-2.amazonaws.com/
AWSServiceRoleForResourceExplorer"
      ]
    }
  ]
}
```

AWS Systems Manager 控制台操作员策略

您可以创建一个如以下示例所示的 IAM 策略，并将该策略附加到您的 IAM 身份。此策略授予操作 Systems Manager 的完全访问权限，并允许 Systems Manager 运行自动化文档来进行诊断和修复。

权限详细信息

该策略包含以下权限。

- `ssm`：允许主体访问所有 Systems Manager API。
- `ssm-quicksetup`：允许主体管理其Quick Setup配置。

- `ec2`：允许 Systems Manager 确定您启用的 AWS 区域和 Amazon EC2 实例状态。
- `cloudformation`：允许主体读取其Quick Setup堆栈。
- `organizations`：允许主体读取 AWS Organizations 中组织的结构，并在委派管理员以组织身份载入 Systems Manager 时管理委派管理员。
- `s3`：允许主体列出并获取 Amazon S3 存储桶中的对象以进行诊断，该存储桶是在 Systems Manager 载入过程中创建的。
- `iam:PassRole`：允许主体在运行自动化来诊断和修复非托管节点时将要代入的角色传递给 Systems Manager。
- `iam:GetRole`：允许主体在 Systems Manager 中工作时获取Quick Setup角色的特定角色信息。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:*",
        "ssm-quicksetup:*"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AllowEC2DescribeActions",
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstanceStatus",
        "ec2:DescribeInstances",
        "ec2:DescribeRegions"
      ],
      "Resource": "*"
    },
    {
      "Sid": "CfnAccess",
      "Effect": "Allow",
      "Action": [
        "cloudformation:ListStacks",
        "cloudformation:ListStackSets",
```

```
        "cloudformation:ListStackInstances",
        "cloudformation:ListStackSetOperations",
        "cloudformation:ListStackSetOperationResults",
        "cloudformation:DescribeStacks",
        "cloudformation:DescribeStackSet",
        "cloudformation:DescribeStackSetOperation",
        "cloudformation:DescribeOrganizationsAccess",
        "cloudformation:DescribeStackInstance",
        "cloudformation:DetectStackSetDrift",
        "cloudformation:ListStackInstanceResourceDrifts"
      ],
      "Resource": "*"
    },
    {
      "Sid": "OrgsReadOnly",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListRoots",
        "organizations:ListParents",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:DescribeOrganizationalUnit",
        "organizations:ListAWSServiceAccessForOrganization"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AllowKMSOperations",
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey"
      ],
      "Resource": "arn:aws:kms:*:*:key/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/SystemsManagerManaged": "true"
        },
        "ArnLike": {
          "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::do-not-delete-ssm-
diagnosis-*"
        },
```

```
        "StringLike": {
          "kms:ViaService": "s3.*.amazonaws.com"
        },
        "Bool": {
          "aws:ViaAWSService": "true"
        }
      }
    },
    {
      "Sid": "AllowReadS3BucketFromOrganization",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": "arn:aws:s3:::do-not-delete-ssm-diagnosis*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceOrgId": "${aws:PrincipalOrgId}"
        }
      }
    },
    {
      "Sid": "AllowReadS3BucketFromSingleAccount",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": "arn:aws:s3:::do-not-delete-ssm-diagnosis*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": [
        "arn:aws:iam::*:role/AWS-SSM-DiagnosisAdminRole*",
        "arn:aws:iam::*:role/AWS-SSM-DiagnosisExecutionRole*",
        "arn:aws:iam::*:role/AWS-SSM-RemediationAdminRole*",
        "arn:aws:iam::*:role/AWS-SSM-RemediationExecutionRole*"
```

```
      ],
      "Condition": {
        "StringEquals": {
          "iam:PassedToService": "ssm.amazonaws.com"
        }
      }
    },
    {
      "Sid": "IamReadOnly",
      "Effect": "Allow",
      "Action": "iam:GetRole",
      "Resource": [
        "arn:aws:iam::*:role/AWS-QuickSetup-*",
        "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
      ]
    }
  ]
}
```

AWS Systems Manager 控制台操作员只读策略

您可以创建一个如以下示例所示的 IAM 策略，并将该策略附加到您的 IAM 身份。此策略授予使用 Systems Manager 的只读访问权限。

- `ssm`：允许主体访问 Systems Manager 只读 API。
- `ssm-quicksetup`：允许主体读取其Quick Setup配置。
- `cloudformation`：允许主体读取其Quick Setup堆栈。
- `iam:GetRole`：允许主体在使用 Systems Manager 时获取Quick Setup角色的特定角色信息。
- `ec2:DescribeRegions`：允许 Systems Manager 确定已启用的 AWS 区域。
- `organizations`：允许主体在以组织身份载入 Systems Manager 时读取 AWS Organizations 中组织的结构。
- `s3`：允许主体列出并获取在 Systems Manager 载入过程中创建的 Amazon S3 存储桶中的对象。

权限详细信息

该策略包含以下权限。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:Describe*",
        "ssm:Get*",
        "ssm:List*",
        "ssm-quicksetup:List*",
        "ssm-quicksetup:Get*",
        "cloudformation:Describe*",
        "cloudformation:Get*",
        "cloudformation:List*",
        "iam:GetRole",
        "ec2:DescribeRegions",
        "organizations:Describe*",
        "organizations:List*"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AllowKMSOperations",
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt"
      ],
      "Resource": "arn:aws:kms:*:*:key/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/SystemsManagerManaged": "true"
        },
        "ArnLike": {
          "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::do-not-delete-ssm-
diagnosis-*"
        },
        "StringLike": {
          "kms:ViaService": "s3.*.amazonaws.com"
        },
        "Bool": {
          "aws:ViaAWSService": "true"
```

```
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": "arn:aws:s3:::do-not-delete-ssm-diagnosis*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceOrgId": "${aws:PrincipalOrgId}"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": "arn:aws:s3:::do-not-delete-ssm-diagnosis*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
      }
    }
  ]
}
```

为组织设置 Systems Manager 统一控制台

只需单击几下即可从 AWS Management Console完成 Systems Manager 统一控制台体验的设置过程。要为 AWS Organizations 组织设置 Systems Manager，您必须有权访问您的组织的管理账户以及您的组织中的另一个账户以用作委派管理员。仅当启用或禁用 Systems Manager 时才需要访问管理账户。要管理您的节点，您将使用委派管理员账户。

先决条件

在管理整个组织的节点时，Systems Manager 会使用各种从属服务来设置和增强统一控制台的功能。因此，Systems Manager 必须启用可信访问并为以下服务注册委派管理员账户：

- AWS CloudFormation：将 Systems Manager 所需的资源部署到您的账户。
- AWS 资源探索器：搜索和筛选您账户中的 EC2 实例。
- AWS Systems Manager Explorer：监控您的账户中为 Systems Manager 部署的资源的运行状况并排查其故障。
- AWS Systems Manager Quick Setup：将 Systems Manager 所需的Quick Setup配置部署到您的账户。

在开始之前，请确保您尚未超过任何这些从属服务的委派管理员配额。否则，您将不能注册启用 Systems Manager 所需的委派管理员账户。当您为某组织启用 Systems Manager 时，您组织中的每个账户都将被包括在内。目前，尚没有关于将账户排除在设置过程之外的规定。启用 Systems Manager 时，您可以选择要包含的 AWS 区域。仅可选择当前支持 Systems Manager 统一控制台的区域。要了解有关控制台体验可用的区域的更多信息，请参阅支持的 AWS 区域。

统一控制台资源

Systems Manager 统一控制台的设置过程会为您完成许多先决条件任务。根据您选择配置的功能，这包括启用默认主机管理配置，为您的节点提供所需 IAM 权限等。以下是 Systems Manager 统一控制台创建的资源的详细列表。根据您选择配置的功能，某些资源可能无法创建。

AWS 资源探索器 托管视图

- `AWSManagedViewForSSM` – 使 Systems Manager 能够访问由 Resource Explorer 为贵组织编制索引的资源信息。这些托管视图只能由 Systems Manager 更新或删除。这意味着，如果要删除托管视图或关闭 Resource Explorer，则必须禁用统一控制台。有关禁用统一控制台的更多信息，请参阅禁用 Systems Manager 统一控制台。有关托管视图的更多信息，请参阅 Resource Explorer User Guide 中的 AWS Managed Views。

> **Note**
>
> 如果您在不同于主区域的区域中为资源探索器创建了聚合器索引，则 Systems Manager 会降级当前索引。然后，Systems Manager 会将您的主区域中的本地索引升级为新的聚合器索引。在此期间，仅显示您的主区域的节点。此过程最多可能需要 24 小时才能完成。

IAM 角色

- `RoleForOnboardingAutomation`：允许 Systems Manager 在设置过程中管理资源。有关策略的更多信息，请参阅 AWSQuickSetupSSMManageResourcesExecutionPolicy。
- `RoleForLifecycleManagement`：允许 Lambda 管理设置过程创建的资源的生命周期。有关策略的更多信息，请参阅 AWSQuickSetupSSMLifecycleManagementExecutionPolicy。
- `RoleForAutomation`：Systems Manager Automation 执行运行手册要承担的服务角色。有关更多信息，请参阅 使用控制台为 Automation 创建服务角色。
- `AWSSSMDiagnosisAdminRole`：用于启动使用诊断运行手册的自动化的管理员角色。有关这些策略的更多信息，请参阅 AWS-SSM-DiagnosisAutomation-AdministrationRolePolicy、AWS-SSM-Automation-DiagnosisBucketPolicy 和 AWS-SSM-DiagnosisAutomation-OperationalAccountAdministrationRolePolicy。
- `AWSSSMDiagnosisExecutionRole`：诊断运行手册的自动化执行角色。有关策略的更多信息，请参阅 AWS-SSM-DiagnosisAutomation-ExecutionRolePolicy 和 AWS-SSM-Automation-DiagnosisBucketPolicy。
- `AWSSSMRemediationAdminRole`：用于启动使用修复运行手册的自动化的管理员角色。有关这些策略的更多信息，请参阅 AWS-SSM-RemediationAutomation-AdministrationRolePolicy、AWS-SSM-Automation-DiagnosisBucketPolicy 和 AWS-SSM-RemediationAutomation-OperationalAccountAdministrationRolePolicy。
- `AWSSSMRemediationExecutionRole`：修复运行手册的自动化执行角色。有关策略的更多信息，请参阅 AWS-SSM-RemediationAutomation-ExecutionRolePolicy 和 AWS-SSM-Automation-DiagnosisBucketPolicy。
- `ManagedInstanceCrossAccountManagementRole`：允许 Systems Manager 跨账户收集托管式节点的信息。

State Manager 关联

- `EnableDHMCAssociation`：每天运行一次，确保启用默认主机管理配置。
- `SystemAssociationForEnablingExplorer`：每天运行一次，确保已启用 Explorer。Explorer 用于同步来自托管式节点的数据。
- `EnableAREXAssociation`：每天运行一次，确保已启用 AWS 资源探索器。资源探索器用于确定组织中的哪些 Amazon EC2 实例未由 Systems Manager 管理。
- `SSMAgentUpdateAssociation`：每 14 天运行一次，确保托管式节点上安装了 SSM Agent 的最新可用版本。

- SystemAssociationForInventoryCollection：每 12 小时运行一次，从托管式节点收集清单数据。

S3 存储桶

- DiagnosisBucket：存储从诊断运行手册执行中收集的数据。

Lambda 函数

- SSMLifecycleOperatorLambda：允许主体访问所有 AWS Systems Manager 快速设置功能 操作。
- SSMLifecycleResource：自定义资源，可帮助管理设置过程创建的资源的生命周期。

此外，在设置过程完成后，您可以选择诊断并修复节点任务，以自动将修复应用于未报告为由 Systems Manager 管理的节点。这可能包括识别诸如 Systems Manager 端点的网络连接问题等问题。有关更多信息，请参阅 诊断并修复。

设置统一控制台

为组织设置 Systems Manager

1. 登录您组织的管理账户。
2. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
3. 输入您要注册为委派管理员的账户的 ID。
4. 委派管理员账户注册成功后，登录您刚刚注册的委派管理员账户，然后返回 Systems Manager 控制台完成设置 Systems Manager。
5. 选择启用 Systems Manager。
6. 在主区域部分，确定您希望 Systems Manager 聚合您的节点数据的区域。默认情况下，Systems Manager 会选择您当前正在使用的区域。要选择其他主区域，请在设置 Systems Manager 之前将控制台更改为您想要使用的区域。节点数据将在您组织的各个账户和区域之间复制并存储在主区域中。设置 Systems Manager 后，您选择的区域无法更改。要使用其他区域作为您组织的主区域，您必须禁用统一控制台并再次完成设置过程。如果您的组织使用 IAM Identity Center，则必须选择与您设置 IAM Identity Center 的相同区域作为您的主区域。
7. 在区域部分中，选择您想要启用 Systems Manager 的区域。

8. 在功能配置部分中，选择要为配置启用的选项：

   启用默认主机管理配置（DHMC）

   允许 Systems Manager 配置 DHMC。此功能允许 Systems Manager 使用 IAM 角色，以确保账户和区域中的所有 Amazon EC2 实例都具有由 Systems Manager 管理所需的权限。您也可以指定偏差修复的频率。当用户对与通过配置进行的选择冲突的服务或功能进行任何更改时，会发生配置偏差。Systems Manager 会检查配置偏差，并尝试根据您指定的频率进行修复。您可以指定 1 到 31 天之间的值。如果您已经在某个区域配置了 DHMC，则 Systems Manager 不会更改您之前选择的 IAM 角色。有关 DHMC 的更多信息，请参阅使用默认主机管理配置自动管理 EC2 实例。

   DHMC 使您不必手动创建 AWS Identity and Access Management（IAM）实例配置文件即可管理 Amazon EC2 实例。建议您选择此选项，以确保您的 EC2 实例具有由 Systems Manager 管理所必需的权限。

   启用清单元数据收集

   让 Systems Manager 能够配置从节点的以下类型的元数据收集：

   - AWS 组件 - EC2 驱动程序、代理和版本等。
   - 应用程序 - 应用程序名称、发布者和版本等。
   - 节点详细信息 - 系统名称、操作系统（OS）名称、操作系统版本、上次启动时间、DNS、域、工作组和操作系统架构等。
   - 网络配置 - IP 地址、MAC 地址、DNS、网关和子网掩码等。
   - 服务 - 名称、显示名称、状态、相关服务、服务类型和启动类型等（仅限 Windows Server 节点）。
   - Windows 角色 - 名称、显示名称、路径、功能类型和安装状态等（仅限 Windows Server 节点）。
   - Windows 更新 - 补丁 ID、安装者和安装日期等（仅限 Windows Server 节点）。

   指定清单元数据的收集频率。您可以指定 1 到 744 之间的值。有关 Inventory（AWS Systems Manager 中的一项工具）的更多信息，请参阅 AWS Systems Manager 清单。

   启用 Systems Manager（SSM）Agent 自动更新

   让 Systems Manager 能够按您指定的频率检查是否存在新版本的代理。频率值必须在 1 到 31 天之间。如果存在新版本，Systems Manager 会自动将托管式节点上的代理更新为发布的最新版本。Systems Manager 不会在尚未安装代理的实例上安装代理。有关哪些 AMIs 预装了 SSM Agent 的信息，请参阅 查找预装了 SSM Agent 的 AMIs。

建议您选择此选项，以确保您的节点始终运行最新版本的 SSM Agent。有关 SSM Agent 的更多信息（包括有关如何手动安装该代理的信息），请参阅 使用 SSM Agent。

9. 选择提交。

根据您组织的规模，设置 Systems Manager 统一控制台体验可能需要较长的时间。

为单个账户和区域设置 Systems Manager 统一控制台

要为单个 AWS 账户和 AWS 区域设置 Systems Manager 统一控制台体验，您无需使用 Organizations 或注册委派管理员账户。Systems Manager 控制台体验的设置过程会为您完成许多先决条件任务。根据您选择配置的功能，这包括启用默认主机管理配置，为您的节点提供所需 IAM 权限等。以下是 Systems Manager 统一控制台创建的资源的详细列表。

统一控制台资源

根据您选择配置的功能，某些资源可能无法创建。

IAM 角色

- `RoleForOnboardingAutomation`：允许 Systems Manager 在设置过程中管理资源。有关策略的更多信息，请参阅 AWSQuickSetupSSMManageResourcesExecutionPolicy。
- `RoleForLifecycleManagement`：允许 Lambda 管理设置过程创建的资源的生命周期。有关策略的更多信息，请参阅 AWSQuickSetupSSMLifecycleManagementExecutionPolicy。
- `RoleForAutomation`：Systems Manager Automation 执行运行手册要承担的服务角色。有关更多信息，请参阅 使用控制台为 Automation 创建服务角色。
- `AWSSSMDiagnosisAdminRole`：诊断运行手册的自动化执行角色。有关这些策略的更多信息，请参阅 AWS-SSM-DiagnosisAutomation-AdministrationRolePolicy、AWS-SSM-Automation-DiagnosisBucketPolicy 和 AWS-SSM-DiagnosisAutomation-OperationalAccountAdministrationRolePolicy。
- `AWSSSMRemediationAdminRole`：修复运行手册的自动化执行角色。有关这些策略的更多信息，请参阅 AWS-SSM-RemediationAutomation-AdministrationRolePolicy、AWS-SSM-Automation-DiagnosisBucketPolicy 和 AWS-SSM-RemediationAutomation-OperationalAccountAdministrationRolePolicy。
- `ManagedInstanceCrossAccountManagementRole`：允许 Systems Manager 跨账户收集托管式节点的信息。

State Manager 关联

- `EnableDHMCAssociation`：每天运行一次，确保启用默认主机管理配置。
- `SystemAssociationForEnablingExplorer`：每天运行一次，确保已启用 Explorer。Explorer 用于同步来自托管式节点的数据。
- `EnableAREXAssociation`：每天运行一次，确保已启用 AWS 资源探索器。资源探索器用于确定组织中的哪些 Amazon EC2 实例未由 Systems Manager 管理。
- `SSMAgentUpdateAssociation`：每 14 天运行一次，确保托管式节点上安装了 SSM Agent 的最新可用版本。
- `SystemAssociationForInventoryCollection`：每 12 小时运行一次，从托管式节点收集清单数据。

S3 存储桶

- `DiagnosisBucket`：存储从诊断运行手册执行中收集的数据。

Lambda 函数

- `SSMLifecycleOperatorLambda`：允许主体访问所有 AWS Systems Manager 快速设置功能 操作。
- `SSMLifecycleResource`：自定义资源，可帮助管理设置过程创建的资源的生命周期。

此外，在设置过程完成后，您可以选择诊断并修复节点任务，以自动将修复应用于未报告为由 Systems Manager 管理的节点。这可能包括识别诸如 Systems Manager 端点的网络连接问题等问题。

设置统一控制台

为单个账户和区域设置 Systems Manager

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 选择启用 Systems Manager。
3. 在功能配置部分中，选择要为配置启用的选项：

启用默认主机管理配置（DHMC）

允许 Systems Manager 配置 DHMC。此功能允许 Systems Manager 使用 IAM 角色，以确保账户和区域中的所有 Amazon EC2 实例都具有由 Systems Manager 管理所需的权限。您也可以指定偏差修复的频率。当用户对与通过配置进行的选择冲突的服务或功能进行任何更改时，会发生配置偏差。Systems Manager 会检查配置偏差，并尝试根据您指定的频率进行修复。您可以指定 1 到 31 天之间的值。如果您已经在某个区域配置了 DHMC，则 Systems Manager 不会更改您之前选择的 IAM 角色。有关 DHMC 的更多信息，请参阅使用默认主机管理配置自动管理 EC2 实例。

DHMC 使您不必手动创建 AWS Identity and Access Management（IAM）实例配置文件即可管理 Amazon EC2 实例。建议您选择此选项，以确保您的 EC2 实例具有由 Systems Manager 管理所必需的权限。

启用清单元数据收集

让 Systems Manager 能够配置从节点的以下类型的元数据收集：

- AWS 组件 - EC2 驱动程序、代理和版本等。
- 应用程序 - 应用程序名称、发布者和版本等。
- 节点详细信息 - 系统名称、操作系统（OS）名称、操作系统版本、上次启动时间、DNS、域、工作组和操作系统架构等。
- 网络配置 - IP 地址、MAC 地址、DNS、网关和子网掩码等。
- 服务 - 名称、显示名称、状态、相关服务、服务类型和启动类型等（仅限 Windows Server 节点）。
- Windows 角色 - 名称、显示名称、路径、功能类型和安装状态等（仅限 Windows Server 节点）。
- Windows 更新 - 补丁 ID、安装者和安装日期等（仅限 Windows Server 节点）。

指定清单元数据的收集频率。您可以指定 1 到 744 之间的值。有关 Inventory（AWS Systems Manager 中的一项工具）的更多信息，请参阅 AWS Systems Manager 清单。

启用 Systems Manager（SSM）Agent 自动更新

让 Systems Manager 能够按您指定的频率检查是否存在新版本的代理。频率值必须在 1 到 31 天之间。如果存在新版本，Systems Manager 会自动将托管式节点上的代理更新为发布的最新版本。Systems Manager 不会在尚未安装代理的实例上安装代理。有关哪些 AMIs 预装了 SSM Agent 的信息，请参阅 查找预装了 SSM Agent 的 AMIs。

建议您选择此选项，以确保您的节点始终运行最新版本的 SSM Agent。有关 SSM Agent 的更多信息（包括有关如何手动安装该代理的信息），请参阅 使用 SSM Agent。

4. 选择提交。

禁用 Systems Manager 统一控制台

要为某组织禁用 Systems Manager 统一控制台，您需要访问您注册为 Systems Manager 委派管理员的账户。在登录您的组织的委派管理员账户后，您可以在统一控制台的设置部分中禁用组织的设置。当为您的组织禁用统一控制台设置时，Systems Manager 会删除在设置过程中创建的资源，包括 Resource Explorer 托管视图。为您的组织禁用设置不会撤消受信任的访问权限或取消注册从属服务的委派管理员账户。以下过程介绍了如何禁用统一控制台的设置。

禁用 Systems Manager 统一控制台的设置

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中选择设置。
3. 选择禁用。您必须确认您想要为您的组织禁用该设置。此操作将删除为统一控制台创建的资源，并且无法撤消。

如果您无法访问 Systems Manager 的委派管理员账户并且想要禁用统一控制台的设置，则也可以从组织的管理账户执行此操作。使用 AWS CLI 或 SDK，调用 `DeleteConfigurationManager` API 操作并传递您账户中组织设置的 `ManagerArn` 值。用于设置统一控制台的管理器 ARN 格式如下：

`arn:aws:ssm-quicksetup:`*`account-id`*`:configuration-manager/`*`configuration-manager-id`*.

> **Note**
>
> 如果不知道 *`configuration-manager-id`* 的值，请调用 `ListConfigurationManagers` API 操作并使用 `AWSQuickSetupType-SSM` 类型筛选结果。

使用 AWS Systems Manager 执行节点管理任务

以下主题介绍如何使用 AWS Organizations 组织和单个 AWS 账户的 AWS Systems Manager 统一控制台完成常见节点任务。

主题

- 查看节点见解
- 浏览节点
- 使用 Systems Manager 进行即时节点访问
- 诊断并修复
- 调整 Systems Manager 设置

查看节点见解

您可以使用 Systems Manager 统一控制台深入了解组织或账户中的托管式节点和非托管式 EC2 实例的总体状态。

Systems Manager 提供了对尚未由 Systems Manager 管理的托管节点和 EC2 实例的直观概览。（托管节点是指在混合和多云环境中配置为与 Systems Manager 一起使用的任何计算机。有关受支持的计算机类型的信息，请参阅混合和多云环境中支持的计算机类型。）

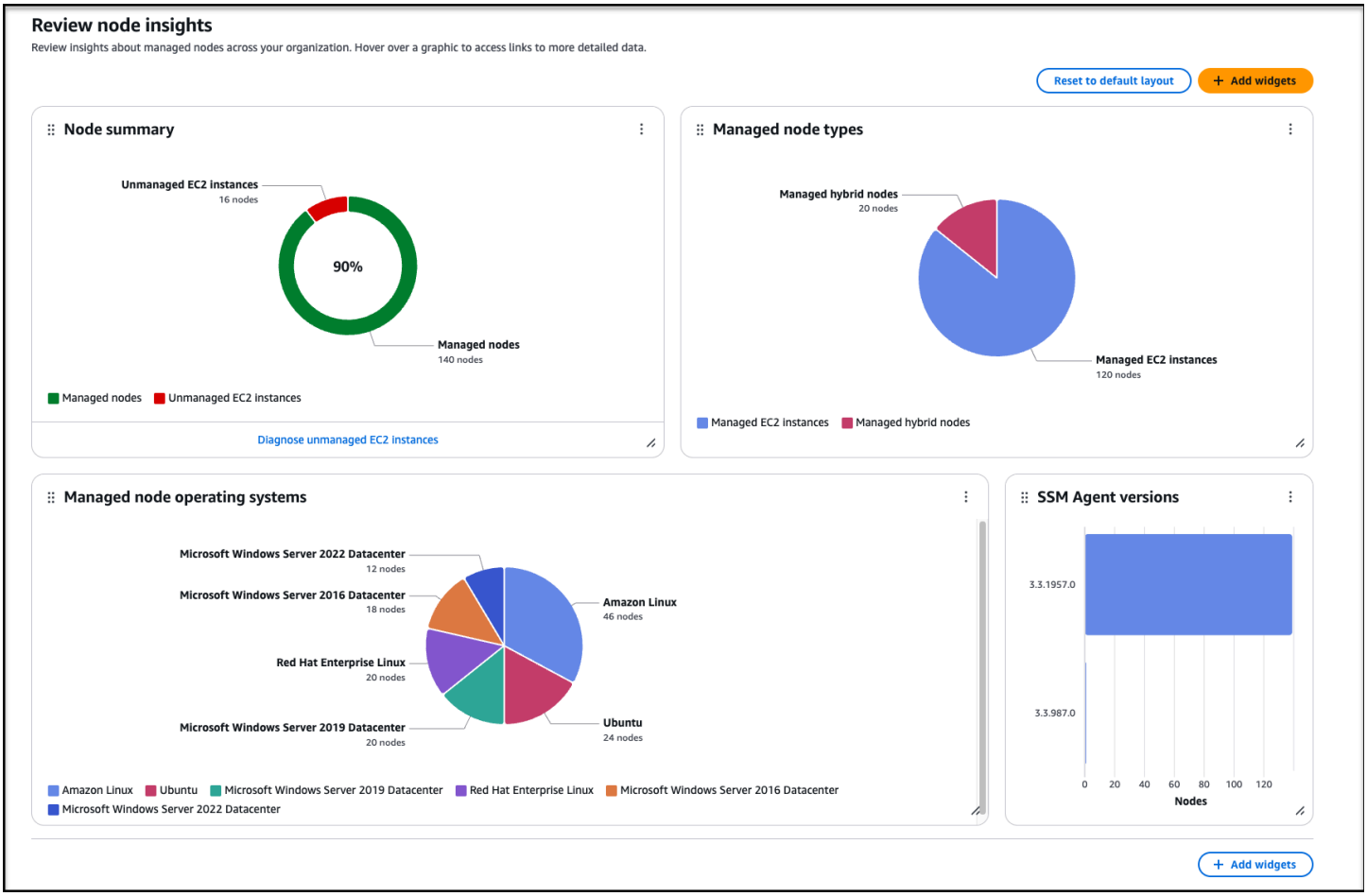
此概览通过单独的报告框（称为小组件）提供，其中包含交互式饼图和其他图形。

开始前的准备工作

为了查看节点见解，您必须首先将组织或账户加入 Systems Manager 统一控制台。有关更多信息，请参阅 设置 AWS Systems Manager。

加入后，请打开 Systems Manager 控制台，然后选择查看节点见解。

下图显示了单独的报告框（称为小组件），可在查看节点见解页面上找到。



显示支持为您提供以下信息的小组件。

节点摘要

指示您的组织或账户中有多少个 EC2 实例当前不是托管节点以及您的组织或账户的实例集中有多少个托管节点。

什么是非托管式实例？

停止托管式 EC2 实例后，Systems Manager 控制台会将其报告为“非托管”。此乃预期行为，因为 SSM Agent 没有与该服务建立活跃连接。

> **Note**
>
> 这与 AWS Config 将实例定义为非托管的方式不同。如果某个实例当前已停止，则 AWS Config 会报告该实例上的 SSM Agent 与 Systems Manager 服务之间最后一次建立“检测信号”连接时该实例的状态。

当实例重新启动后，会自动重新连接到 Systems Manager 服务，并且其在统一控制台中的状态将在五分钟内恢复为“托管”。无需手动干预，在停止/启动周期内，会保留实例的所有 Systems Manager 配置。

但是，如果实例在启动几分钟后仍未报告为“托管”，则该实例可能未正确配置为 Systems Manager 管理。在这种情况下，建议运行诊断以确定实例依旧处于非托管状态的原因。有关更多信息，请参阅 在 Systems Manager 中诊断和修复非托管 Amazon EC2 实例。

如果诊断扫描无法确定问题，请参阅以下主题以验证 SSM Agent 要求、AWS Identity and Access Management（IAM）角色要求以及 Systems Manager 先决条件是否均已满足：

- 排查 SSM Agent 问题
- 配置 Systems Manager 所需的实例权限
- 排除托管式节点可用性的问题

托管节点类型

指示您的实例集中有多少个托管节点是 EC2 实例以及有多少个是其他服务器类型，包括在您自己本地的服务器（本地服务器）、AWS IoT Greengrass 核心设备、AWS IoT 和非 AWS 边缘设备以及虚拟机 (VM)（包括其他云环境中的 VM）。您可以将鼠标悬停在节点类型图形上，以访问浏览节点页面中指向更多详细信息的链接。

有关对混合和多云环境的 AWS 支持的更多信息，请参阅适用于混合云和多云的 AWS 解决方案。

SSM Agent 版本

提供有关您的实例集中 AWS Systems Manager Agent (SSM Agent) 安装的信息。SSM Agent 是在您的托管节点上运行的 Amazon 软件。SSM Agent 使 Systems Manager 能够更新、管理和配置这些资源。该代理处理来自 AWS Cloud 中的 Systems Manager 服务的请求，然后按照请求中指定的方式运行它们。

对于您的实例集中的托管节点，此小组件会报告您的实例集中的 SSM Agent 版本（从最新到最旧）。您可以将鼠标悬停在 SSM Agent 版本图形上，以访问浏览节点页面中指向更多详细信息的链接。

有关 SSM Agent的更多信息，请参阅使用 SSM Agent。

托管节点操作系统

提供您的实例集中托管节点上每个操作系统的百分比明细。您可以将鼠标悬停在各操作系统的托管节点图形上，以访问浏览节点页面中指向更多详细信息的链接。

您可以使用拖放功能以及通过在显示中移除和添加小组件来自定义查看节点见解页面上的小组件布局。

使用以下主题中的信息来帮助您使用 Systems Manager 节点见解小组件。

主题

- 在查看节点见解页面中添加或移除小组件
- 重新排列查看节点见解页面中的小组件

在查看节点见解页面中添加或移除小组件

您可以通过添加和移除小组件来自定义 Systems Manager 查看节点见解页面的布局。

> **Note**
>
> 默认情况下，该页面会显示所有可用的小组件。

在查看节点见解页面中添加或移除小组件

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在左侧导航中，选择查看节点见解。
3. 要从显示中移除小组件，请执行下面的操作：
   a. 选择小部件的“更多选项”菜单（ ⋮ ）。
   b. 选择删除小组件。
4. 要向显示添加小组件，请执行下面的操作：
   a. 选择添加小部件。
   b. 在添加小组件窗格中，单击并按住要添加到显示中的小组件的拖动手柄（ ⠿ ）。
   c. 将小组件拖放到主窗格中。

重新排列查看节点见解页面中的小组件

您可以通过重新排列小组件来自定义查看节点见解页面的布局。

重新排列查看节点见解页面中的小组件

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择查看节点见解。
3. 要自定义小组件布局，请选择要移动的小组件。单击并按住小组件的拖动手柄 ()，然后将其拖到新位置。
4. 对于要移动的每个小部件，请重复该过程。

如果确定您不喜欢新布局，请选择重置为默认布局以将所有小组件移回到原始位置。

浏览节点

您可以使用 Systems Manager 中的浏览节点页面根据您在筛选条件中指定的标准来查看您的组织或账户中托管节点的详细信息。您还可以使用 Systems Manager 与 Amazon Q 开发者版 (Amazon Q)（一种 AWS 生成式人工智能解决方案）的集成来通过文本提示进行搜索。

开始前的准备工作

要使用浏览节点功能，必须首先将组织或账户加入 Systems Manager 统一控制台。有关更多信息，请参阅 为组织设置 Systems Manager 统一控制台。

加入后，请打开 Systems Manager 控制台，然后选择浏览节点。

> **Note**
>
> 如果您在不同于主区域的区域中为资源探索器创建了聚合器索引，则 Systems Manager 会降级当前索引。然后，Systems Manager 会将您的主区域中的本地索引升级为新的聚合器索引。在此期间，仅显示您的主区域的节点。此过程最多可能需要 24 小时才能完成。

主题

- 使用控制台筛选条件探索节点
- 在 Amazon Q 中使用文本提示探索节点
- 查看单个节点详细信息并对节点执行操作

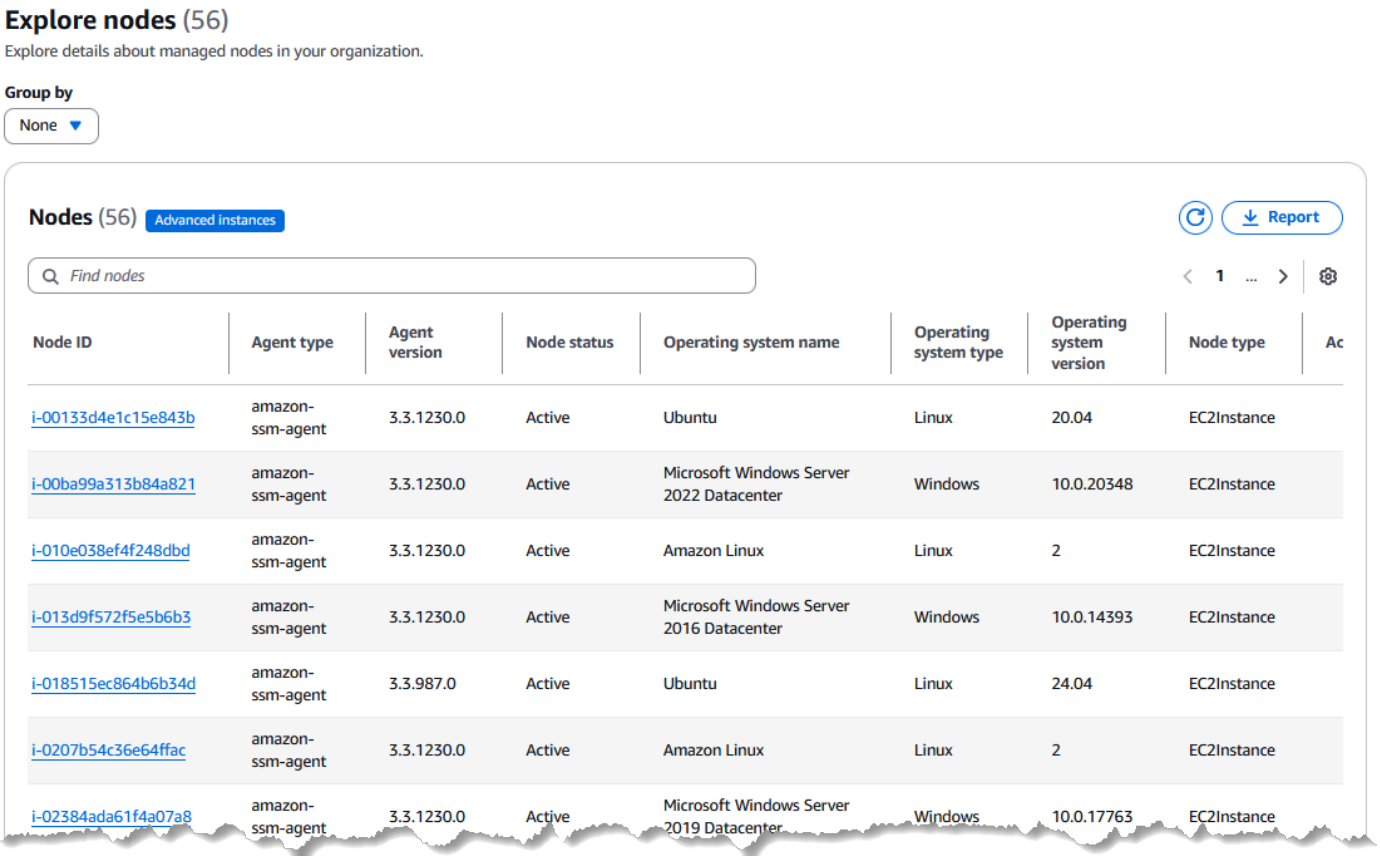
- 下载或导出托管节点报告
- 管理节点报告内容和外观

使用控制台筛选条件探索节点

然后，在 Systems Manager 控制台中，您可以根据以下视图对托管节点进行分组：

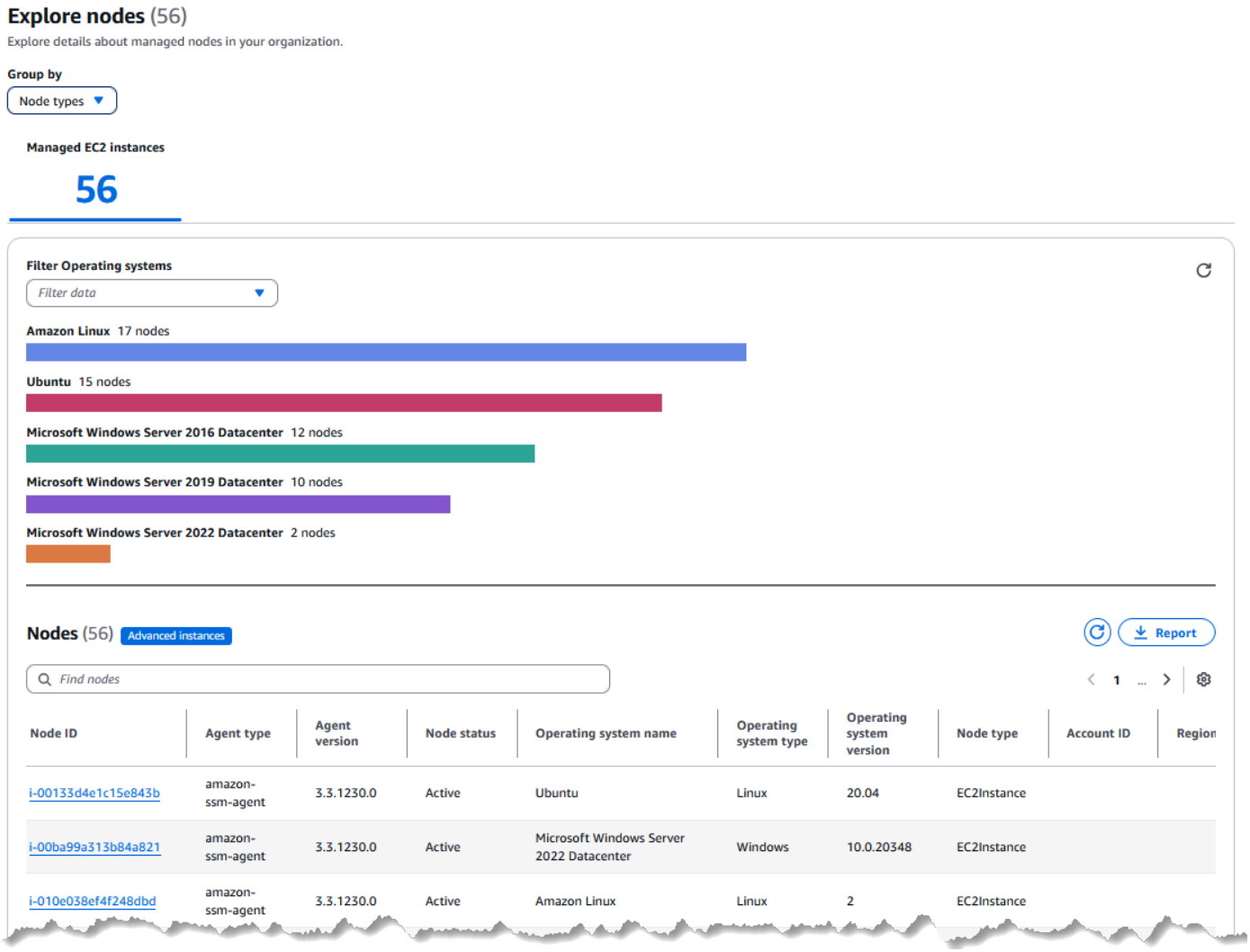
All nodes (No filter)

列出您的组织或账户中的所有托管节点。



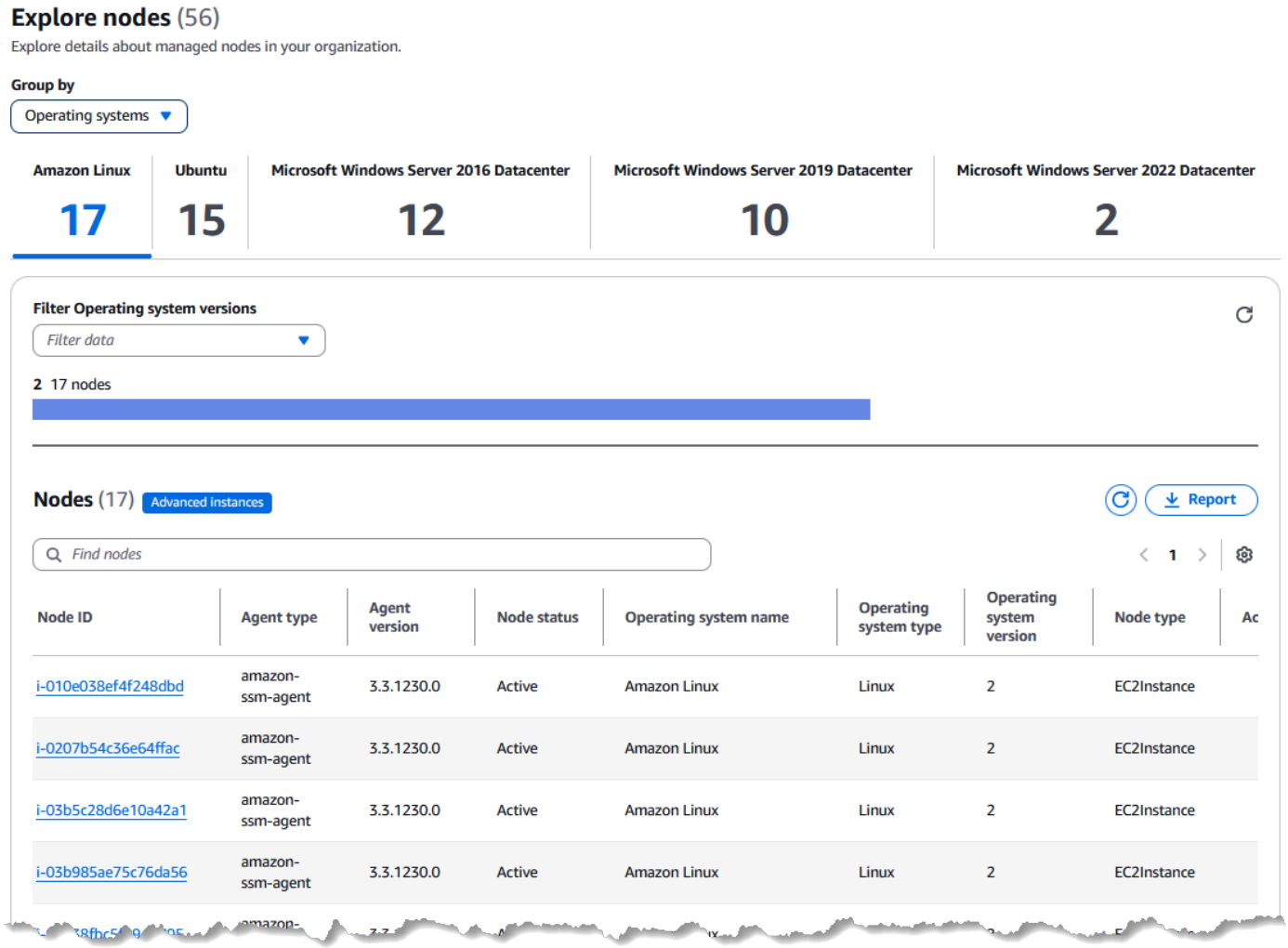
Node types

提供用于分别查看 Amazon Elastic Compute Cloud (Amazon EC2) 实例和其他计算机类型的数据的选项卡，包括您自己本地的服务器（本地服务器）、AWS IoT Greengrass 核心设备、AWS IoT 和非 AWS 边缘设备以及虚拟机 (VM)（包括其他云环境中的 VM）。



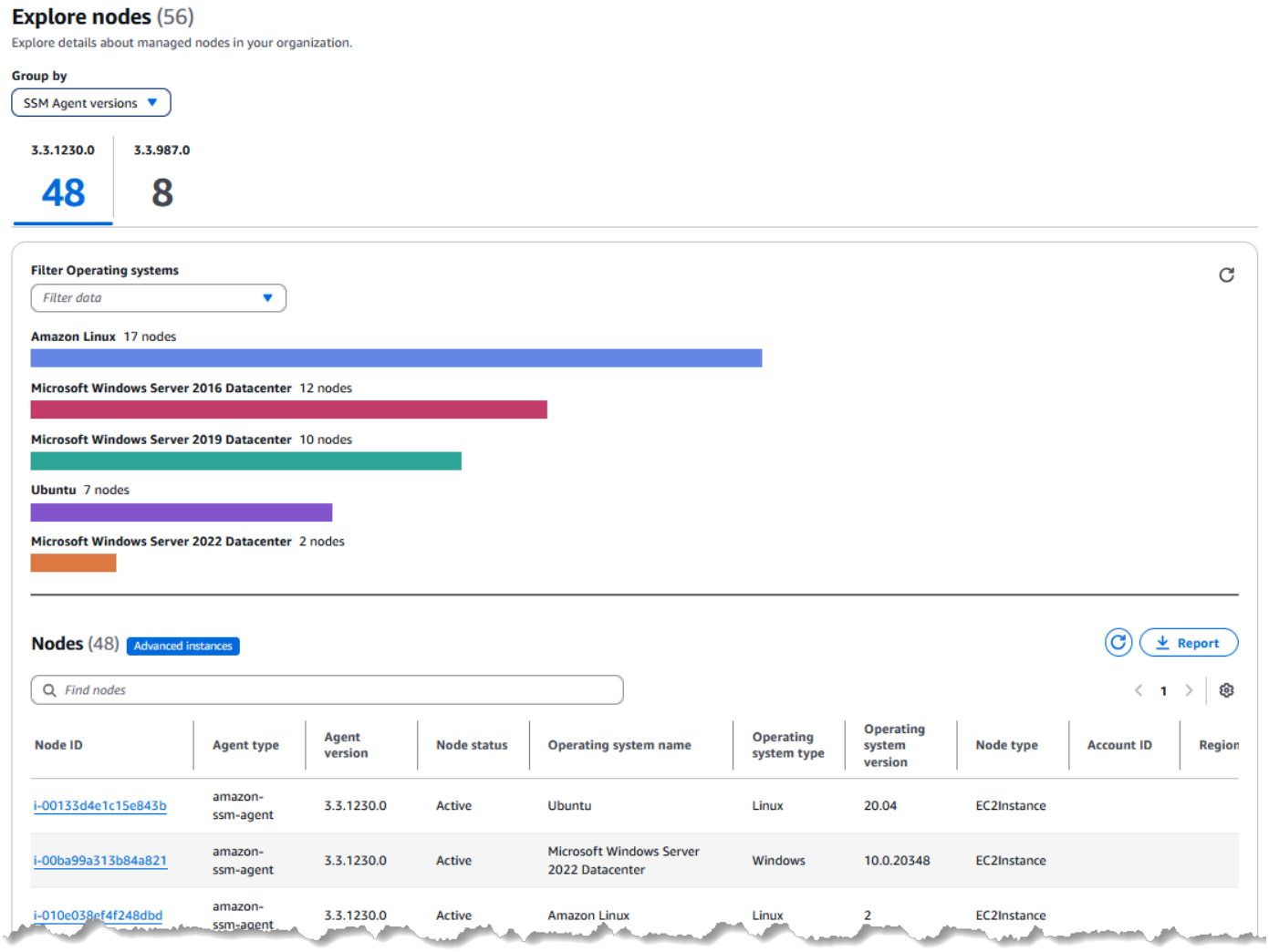
Operating systems

为您的组织或账户中的每种操作系统类型（例如 Amazon Linux 和 Microsoft Windows Server 2022 Datacenter）提供一个选项卡。在每个选项卡上，您可以通过仅选择特定版本的操作系统（例如 Amazon Linux 2 和 Amazon Linux 2023）来进一步筛选列表。



SSM Agent versions

为您的实例集中的托管节点上安装的每个 SSM Agent 版本提供一个选项卡。在每个选项卡上，您可以通过仅选择特定操作系统（例如 Amazon Linux 和 Microsoft Windows Server 2022 Datacenter）来进一步筛选列表。



此外，对于每个视图，您可以通过选择仅查看具有特定属性（例如节点状态、AWS 账户 ID、组织单位 ID 等）的节点来进一步优化报告的节点列表。

您可以通过选择在浏览节点页面中显示哪些可用数据列来自定义报告显示。您还可以下载 CSV 或 JSON 格式的报告，或者以 CSV 格式将报告导出至 Amazon S3。

主题

- 为托管节点摘要选择筛选视图

为托管节点摘要选择筛选视图

使用 Systems Manager 中的浏览节点页面可根据多个可用的筛选视图查看有关您的实例集的聚合数据。

为托管节点摘要选择筛选视图

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格上，选择浏览节点。
3. 对于筛选视图，选择其中一个筛选选项，然后可以选择进一步优化报告：

   - 托管节点：在搜索 () 框中，您可以选择属性和分隔符，例如 `Node type = Managed EC2 instances`。
   - 操作系统：在筛选操作系统版本列表中，您可以选择一个操作系统版本号。在搜索 () 框中，您可以选择属性和分隔符，例如 `Node type = Managed EC2 instances`。
   - SSM Agent 版本：在筛选操作系统列表中，您可以选择一个操作系统名称。在搜索 () 框中，您可以选择属性和分隔符，例如 `Node type = Managed EC2 instances`。
   - 节点类型：在筛选操作系统列表中，您可以选择一个操作系统名称。在搜索 () 框中，您可以选择属性和分隔符，例如 `Node type = Managed EC2 instances`。

在选择性地筛选列表后，您可以通过在节点 ID 列中选择特定托管节点的 ID 来查看有关该节点的详细信息。从该详细视图中，您可以对节点执行多项操作。

在 Amazon Q 中使用文本提示探索节点

通过将 Systems Manager 与 Amazon Q 开发者版集成，您可以使用文本提示来查看生成式人工智能创建的有关托管节点的信息。

Amazon Q 开发者版是一款由生成式人工智能（AI）提供支持的对话式助手，可为您理解、构建、扩展和操作 AWS 应用程序提供帮助。要在 AWS 上进行加速构建，为 Amazon Q 提供支持的模型辅以高质量的 AWS 内容，可为您生成更全面、更具可操作性和参考性的答案。有关更多信息，请参阅《Amazon Q Developer User Guide》中的 What is Amazon Q Developer?。

Systems Manager 与 Amazon Q 之间的集成让您能够快速查看和控制跨多个 AWS 账户和区域的大型分布式环境。您可以使用自然语言查询快速搜索节点数据，然后更快地识别问题并采取行动。

当您询问一个有关托管节点或托管实例的自然语言问题时，Amazon Q 会使用 Systems Manager `ListNodes` 操作，并根据您的文本输入创建筛选条件来检索结果。

例如，假设您向 Amazon Q 提供下面的提示：

`List my managed nodes running Red Hat Enterprise Linux 9.2`

Amazon Q 确定在请求中包含哪些筛选条件，然后运行类似于以下内容的查询：

```
aws ssm list-nodes \
    --filters Key=PlatformName,Values='Red Hat Enterprise Linux',Type=Equal
 Key=PlatformVersion,Values=9.2,Type=Equal
```

然后，Amazon Q 会生成一份有关您账户中的 Red Hat Enterprise Linux 实例的报告，列出实例数量、实例 ID 及其区域等信息。

您还可以查看每个实例的详细信息的 JSON 摘要，以及打开一个链接以在 Systems Manager 浏览节点页面中查看 EC2 实例或托管节点的完整列表。浏览节点显示与您在提示中包含的筛选条件匹配的结果。然后，您可以更改或优化您的请求的筛选条件，如浏览节点中所述。

主题

- 了解如何制作有效的提示来向 Amazon Q 询问您的实例集情况
- 使用 Amazon Q 浏览托管节点

了解如何制作有效的提示来向 Amazon Q 询问您的实例集情况

您向 Amazon Q 提出的问题或提示的质量越好，它为您提供的结果就越好。

查询提示的技巧

向 Amazon Q 查询您的实例集信息时，请记住以下技巧：

1. 为了帮助提高结果的准确性，请在提示中使用术语“托管节点”和“托管实例”，而不仅仅是“节点”和“实例”。
2. 要查要查询属于某个组织的多个账户的结果（如 AWS Organizations 中所配置），您必须登录到指定主区域中的委派管理员账户。
3. 在委派管理员账户中，使用术语帮助 Amazon Q 理解您询问的是整个组织中的节点和实例，具体使用“在我的组织中”或“在我的账户 123456789012 中”等术语。

主题

- Amazon Q 的问题示例
- 提示支持的操作系统名称和版本

Amazon Q 的问题示例

在下表中，我们提供了示例问题，演示了创建 Amazon Q 查询以获得更好结果的一些方法。

我们还提供了 Amazon Q 在运行 listNodes 命令时将应用的筛选条件示例，这些示例是根据您的提示内容生成的。

自然语言问题示例	Amazon Q 应用的筛选条件
Show me my Windows managed nodes.	`PlatformType = Windows`
List my managed instances in account 123456789012.	`AccountId = 123456789012`
Show me all managed nodes running Amazon Linux 1 across my organization.	`PlatformName = Amazon Linux` `PlatformVersion = 1`
Show me all managed instances running Microsoft Windows Server 2019 Datacenter in my organization.	`PlatformName = Microsoft Windows Server 2019 Datacenter`
Can you show me all managed nodes with SSM Agent version 3.3.1142.0?	`AgentType = amazon-ssm-agent` `AgentVersion = 3.3.1142.0`

List all Amazon Linux 2 managed instances in account 123456789012 that have SSM Agent version 3.3.1230.0.	`PlatformName = Amazon Linux`
	`PlatformVersion = 2`
	`AccountId = 123456789012`
	`AgentType = amazon-ssm-agent`
	`AgentVersion = 3.3.1230.0`

自然语言问题示例	Amazon Q 应用的筛选条件
What Microsoft Windows Server 2008 R2 Enterprise managed nodes are running in the eu-central-1 region across my entire organization?	`PlatformName = Microsoft Windows Server 2008 R2 Enterprise` `Region = eu-central-1`

Show me all managed instances running CentOS Linux 7 in ou-d6ty-g xdma6vm.	`PlatformName = CentOS Linux`
	`PlatformVersion = 7`
	`OrganizationalUnitId = ou-d6ty-g xdma6vm`

What Ubuntu managed instances are in account 123456789012?	`PlatformName = Ubuntu` `AccountId = 123456789012`
List my Linux managed instances.	`PlatformType = Linux`
Find my macOS managed nodes.	`PlatformType = macOS`
Show me all Amazon Linux managed nodes in my org.	`PlatformName = Amazon Linux`
List managed nodes running Amazon Linux 2 .	`PlatformName = Amazon Linux` `PlatformVersion = 2`

List the managed nodes with Ubuntu 16.04 in account 123456789012.	`PlatformName = Ubuntu`
	`PlatformVersion = 16.04`
	`AccountId = 123456789012`

Find all managed nodes that have an SSM Agent version that is not 3.3.987.0.	`AgentType = amazon-ssm-agent` `AgentVersion != 3.3.987.0`

自然语言问题示例	Amazon Q 应用的筛选条件
List all managed instances that are not running a Linux operating system.	`PlatformType != Linux`

提示支持的操作系统名称和版本

当向 Amazon Q 询问您的账户中的托管节点时，提供 Systems Manager 中标记的操作系统的名称非常有帮助。您还可以提供版本号来进一步缩小结果范围。例如，如下表所示，您可以要求提供有关 **macOS 14.5**、**Microsoft Windows Server 2019 Datacenter** 和 **AlmaLinux 9.2 through 9.4** 的具体结果，仅举几个例子。

这些列表可能并不详尽，仅作为示例提供。

macOS

平台名称	版本号
macOS	11.6.7、11.7.10、12.6.6、12.7.6、13.2、13.4、13.7、14.1、14.5、14.6.1、15.0

Windows

发行版	版本号
Microsoft Windows Server 2008 Enterprise	6.0.6003
Microsoft Windows Server 2008 R2 Datacenter	6.1.7601
Microsoft Windows Server 2008 R2 Enterprise	6.1.7601
Microsoft Windows Server 2008 R2 Standard	6.1.7601
Microsoft Windows Server 2012 R2 Datacenter	6.3.9600
Microsoft Windows Server 2012 R2 Standard	6.3.9600
Microsoft Windows Server 2012 Standard	6.2.9200

发行版	版本号
Microsoft Windows Server 2016 Datacenter	不适用
Microsoft Windows Server 2016 Standard	10.0.14393
Microsoft Windows Server 2019 Datacenter	不适用
Microsoft Windows Server 2019 Standard	不适用
Microsoft Windows Server 2022 Datacenter	不适用
Microsoft Windows Server 2022 Standard	10.0.20348

Linux

平台名称	版本号
AlmaLinux	8.10、9.2、9.3、9.4
Amazon Linux 2	2.0 及更高版本
Amazon Linux 2023	2023.0.20230315.0 及更高版本
Amazon Linux	2015.03、2015.09、2016.03、2016.09、2017.03、2017.09、2018.03
BottleRocket	1.14.3、1.16.1、1.18.0、1.19.1、1.19.2、1.19.5、1.20.0、1.20.1、1.20.2、1.20.3、1.20.5、1.21.1、1.23.0、1.24.0、1.24.1、1.25.0、1.26.1、
CentOS	6.7、6.8、6.9（最终版）、6.10、6.10（最终版）
CentOS Linux	7、7.2.1511、7.2.1511（核心）、7.3.1611、7.3.1611（核心）、7.4.1708、7.5.1804、7.5.1804（核心）、7.6.1810、7.7.

平台名称	版本号
	1908、7.8.2003、7.9.2009、7.9.2009（核心）、8.2.2004、8.5.2111
CentOS Stream	8、9
Debian GNU/Linux	8-12
Oracle Linux Server	7.8、8.2、8.3、8.8、8.9、8.10、9.4
Red Hat Enterprise Linux	8.2、8.3、8.4、8.5、8.6、8.7、8.8、8.9、8.10、9.2、9.3、9.4
Red Hat Enterprise Linux 服务器	6.9（圣地亚哥）、6.10（圣地亚哥）、7.3、7.6、7.7、7.8、7.9
Rocky Linux	8.6、8.7、8.8、8.9、8.10、9.1、9.2、9.3、9.4
SLES	12.4、15、15.1、15.2、15.3、15.4、15.5
Ubuntu	14.04、16.04、18.04、20.04、22.04、24.04

使用 Amazon Q 浏览托管节点

Systems Manager 与 Amazon Q Developer 集成让您可以从 AWS Management Console中可以使用 Amazon Q 界面的任意位置询问有关您的实例集中托管节点的问题。

有关与 Amazon Q 集成的更多信息，请参阅《Amazon Q Developer User Guide》中的 What is Amazon Q DeveloperAWS。

使用 Amazon Q 浏览托管节点

1. 从 AWS Management Console中的任意位置，选择 Amazon Q 图标 ()。
2. 在 Amazon Q 窗格底部的提示字段中，询问有关您的账户或组织中托管节点的问题。

> **Tip**
>
> 有关创建有效提示的技巧，请查看了解如何制作有效的提示来向 Amazon Q 询问您的实例集情况中的信息。

3. 查看有关特定节点的信息，或选择打开 AWS Systems Manager 控制台以继续浏览。

查看单个节点详细信息并对节点执行操作

从 Systems Manager 的浏览节点页面的列表中，您可以选择单个节点，以查看有关计算机的全面详细信息或对节点执行各种操作。详细信息页面中的常规页面提供了有关节点的全面信息。

查看单个节点详细信息并对节点执行操作

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格上，选择浏览节点。
3. （可选）按照为托管节点摘要选择筛选视图中的步骤优化为您的组织或账户显示的托管节点列表。
4. 在节点 ID 列中，选择节点的链接 ID。
5. 要查看有关该节点的更多详细信息，请在左侧导航的属性列表中，选择一个属性来查看有关以下内容的更多信息：
   - 标签：查看应用于节点的标签列表。您还可以添加或删除标签。
   - 清单：选择一种清单类型（例如 AWS:Application 或 AWS:Network）以查看节点的清单详细信息。
   - 关联：查看有关应用于节点的所有State Manager关联的详细信息，包括状态和关联的 SSM 文档名称等详细信息。
   - 补丁：查看有关节点的补丁和补丁状态的摘要信息。
   - 配置合规性：查看节点的合规性详细信息，例如合规性状态和合规性问题严重性。
6. 要对节点执行操作，请使用节点操作菜单中的下列选项：

> **Note**
>
> 这些操作仅适用于您当前正在工作的 AWS 账户和区域中的托管节点。对于其他账户或区域中您可能有权访问的托管节点，您可以改为访问属性列表。

- 连接、启动终端会话：使用 AWS Systems Manager Session Manager连接到节点。
- 工具
  - 查看文件系统：浏览节点的目录结构的内容。添加、重命名和移除目录。剪切或复制并粘贴文件。
  - 查看性能计数器：查看有关节点的性能信息，例如 CPU 利用率、网络流量和其他利用率类型。。
  - 托管进程：查看有关节点上资源使用情况的信息。启动或停止节点上的进程。
  - 管理用户和组：查看、添加或删除节点上的用户账户和用户组。
  - 执行运行命令：使用 AWS Systems Manager Run Command 来管理节点的配置。Run Command 使用 Systems Manager 文档执行按需更改，例如更新应用程序或运行 Linux Shell 脚本和 Windows PowerShell 命令。
  - 补丁节点：使用 AWS Systems Manager Patch Manager中的立即修补功能从控制台对节点运行按需修补操作。

  > **Note**
  >
  > 也可以从左侧导航的工具菜单中启动上述任务。

- 节点设置
  - 添加标签：将附加的标签键值对应用到节点。
  - 重置节点用户密码：为节点上的指定用户设置新密码。
  - 修改 IAM 角色：更改与节点关联的 IAM 角色。创建一个新的 IAM 角色以附加到该节点。

下载或导出托管节点报告

您可以使用 Systems Manager 浏览节点功能在 Systems Manager 控制台中查看您的 AWS 组织或账户的已筛选或未筛选的托管节点列表。如果您想要离线查看数据或在其他应用程序中处理数据，则可以将报告另存为 CSV 或 JSON 文件。

根据报告的大小，系统会提示您将报告下载到自己的本地计算机或将其导出至 Amazon S3 存储桶。报告仅以 CSV 格式保存到 S3 存储桶。

下载或导出托管节点报告

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格上，选择浏览节点。
3. （可选）按照为托管节点摘要选择筛选视图中的步骤优化为您的组织或账户显示的托管节点列表。
4. 选择报告
   ()。
5. 如果显示下载报告对话框，请执行下面的操作：
   a. 对于文件名，为文件输入名称。我们建议指定一个代表报告范围的名称，例如 `all-organization-nodes` 或 `ec2-instances-out-of-date-agent`。
   b. 对于包含列，指定是包含所有可用节点详细信息的列，还是仅包含您为当前显示选择的列。

      > **Tip**
      > 有关管理报告显示中的列的信息，请参阅管理节点报告内容和外观。

   c. 对于文件格式，选择 CSV 或 JSON，具体取决于您如何使用该文件。
   d. 对于电子表格标题，要在 CSV 文件中包含列标题行，请选择包括列名的行。
   e. 选择下载。

   报告将根据您的浏览器设置保存到默认下载位置。
6. 如果显示了导出到 Amazon S3 对话框，请执行下列操作：
   a. 对于 S3 URI，输入要将报告导出到的存储桶的 URI。

      > **Tip**
      > 要在 Amazon S3 控制台中查看您的存储桶的列表，请选择查看。要从您账户的存储桶列表中进行选择，请选择浏览 S3。

   b. 在授权方法中，指定用于提供将报告导出至存储桶的权限的服务角色。

      如果选择让 Systems Manager 为您创建角色，则它会提供操作所需的所有权限和信任声明。

如果想要使用或创建您自己的角色，则该角色必须包含所需的权限和信任声明。有关创建此角色的信息，请参阅 创建自定义服务角色以将诊断报告导出至 S3。

c. 选择提交。

创建自定义服务角色以将诊断报告导出至 S3

在 Systems Manager 浏览节点页面中查看您的 AWS 组织或账户的托管节点的已筛选或未筛选列表时，您可以将报告的列表以 CSV 文件形式导出到 Amazon S3 存储桶。

为此，您必须指定具有操作所需权限和信任策略的服务角色。您可以选择让 Systems Manager 在下载报告的过程中为您创建角色。或者，您可以自己创建角色及其所需的策略。

创建自定义服务角色以将诊断报告导出至 S3

1. 按照《IAM 用户指南》中使用 JSON 编辑器创建策略中的步骤进行操作。

   - 使用以下内容作为策略内容，确保将####替换为您自己的信息。

     JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::s3-bucket-name/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceAccount": "account-id"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketAcl",
        "s3:ListBucket",
```

```
        "s3:PutLifecycleConfiguration",
        "s3:GetLifecycleConfiguration"
      ],
      "Resource": "arn:aws:s3:::s3-bucket-name",
      "Condition": {
        "StringEquals": {
          "aws:ResourceAccount": "account-id"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "ssm:ListNodes"
      ],
      "Resource": "*"
    }
  ]
}
```

- 为该策略命名，以帮助您在下一步中轻松识别它。

2. 按照《IAM 用户指南》中使用自定义信任策略创建 IAM 角色（控制台）中的步骤进行操作。

   - 对于步骤 4，输入以下信任策略，确保将####替换为您自己的信息。

     JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "SSMAssumeRole",
      "Effect": "Allow",
      "Principal": {
        "Service": "ssm.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "account-id"
        }
      }
    }
  ]
```

```
}
```

3. 对步骤 10，选择步骤 2：添加权限，然后选择您在上一步中创建的策略的名称。

在创建角色后，您可以按照下载或导出托管节点报告中的步骤选择该角色。

管理节点报告内容和外观

您可以使用 Systems Manager 浏览节点功能在 Systems Manager 控制台中查看您的 AWS 组织或账户的已筛选或未筛选的托管节点列表。您可以从十几个字段中进行选择以包含在您的节点列表中，例如节点 ID、操作系统名称、区域等。您还可以重新排序列表和报告的列，并更改列表在控制台中的显示方式。

管理节点报告内容和外观

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格上，选择浏览节点。
3. 在节点区域中，选择首选项齿轮图标
   ()。
4. 在首选项对话框中，执行下面的操作：
   a. 对于页面大小，选择每个控制台视图显示包含多少行，即 10、25 或 50。
   b. 对于自动换行，选中该框以按可用列宽显示单元格的所有内容。
   c. 对于条纹行，选择该框以显示透明和阴影背景的交替行。
   d. 对于选择可见内容，执行下面的操作：
      - 打开或关闭您的列表显示和报告的单个列。
      - 要更改列的顺序，请单击并按住列名称的拖动手柄
        ()，
        然后在列表中向上或向下拖动。
5. 选择确认。

使用 Systems Manager 进行即时节点访问

Systems Manager 通过支持即时访问来帮助您提高节点的安全性。即时节点访问允许用户请求对节点的临时、有时限的访问，只有在真正需要访问时才可批准访问。这样就无需为由 IAM 策略管理的节点提供长期访问权限。此外，Systems Manager 还为 Windows Server 节点的 RDP 会话提供会话录制，以帮助您满足合规性要求、执行根本原因分析等。要使用即时节点访问，必须设置统一的 Systems Manager 控制台。

通过即时节点访问，您可以创建精细的 IAM 策略，确保只有您许可的用户才能向您的节点提交访问请求。然后，您可以创建审批策略，这些策略定义了连接到您的节点所需的审批。对于即时节点访问，有自动审批策略和手动审批策略。自动审批策略定义了用户可以自动连接到哪些节点。手动审批策略定义了访问您指定的节点时必须提供的手动审批的数量和级别。此外，您还可以创建拒绝访问策略。拒绝访问策略明确禁止对您指定节点的访问请求的自动审批。拒绝访问策略适用于 AWS Organizations 组织中的所有账户。自动审批和手动审批策略仅适用于其创建所在的 AWS 账户和 AWS 区域。

当用户尝试连接到某个节点时，系统会提示他们输入访问该节点的原因。然后您的审批策略会经过评估。根据您的策略，用户要么自动连接到目标节点，要么 Systems Manager 自动代表请求者创建手动审批请求。然后，在适用于该节点的手动审批策略中指定的审批者会收到访问请求的通知，他们可以批准或拒绝该请求。审批者和请求者可以通过电子邮件收到通知，也可以通过与 Slack 或 Microsoft Teams 集成的聊天应用程序中的 Amazon Q 开发者版收到通知。只有当指定的审批者提供全部所需批准时，Systems Manager 才会授予对所请求节点的访问权限。收到全部所需批准后，用户可以在审批策略中指定的访问时段期间根据需要启动任意数量的节点会话。Systems Manager 不会自动终止即时节点访问会话。作为最佳实践，请指定最长会话持续时间和空闲会话超时会话首选项的值。这些首选项可防止用户在批准的访问时段之外仍保持与节点的连接。

我们建议使用审批策略组合来帮助您保护包含更多关键数据的节点，同时允许用户无需干预即可连接到不太重要的节点。例如，您可以要求手动审批对数据库节点的访问请求，以及自动审批非持久性表示层节点的会话。

Systems Manager 支持与 IAM Identity Center 或 IAM 联合的用户进行即时节点访问。当联合用户提交访问请求时，他们会指定目标节点以及需要连接到该节点的原因。Systems Manager 会将用户的身份与组织的审批策略中定义的参数进行比较。当满足自动审批策略条件或审批者手动提供批准时，请求者便可连接到目标节点。用户尝试连接到已批准的节点时，Systems Manager 会创建并使用临时令牌来建立会话。

由于 Systems Manager 服务负责处理访问请求的身份验证和建立会话，您不必使用 IAM 策略来管理对节点的访问。通过使用即时节点访问，Systems Manager 可以帮助组织接近零常设权限，因为您只需要允许用户创建访问请求，而不必允许他们以对节点的永久权限启动会话。为帮助您满足合规

性要求，Systems Manager 会将所有访问请求保留 1 年。Systems Manager 还会针对以下情况发出 EventBridge 事件：失败访问请求的即时节点访问，以及手动审批的访问请求状态更新。有关更多信息，请参阅使用 Amazon EventBridge 监控 Systems Manager 事件。

主题

- 使用 Systems Manager 设置即时访问
- 启动即时节点访问会话
- 管理即时访问请求
- 从Session Manager移至即时节点访问
- 使用 Systems Manager 禁止即时访问
- 即时节点访问权限常见问题

使用 Systems Manager 设置即时访问

使用 Systems Manager 设置即时节点访问涉及多个步骤。首先，选择要设置即时节点访问的目标。目标包括 AWS Organizations 组织单元（OU）和 AWS 区域。默认情况下，会选择您在设置统一 Systems Manager 控制台时选择的相同目标进行即时节点访问。您可以选择为所有相同的目标设置即时节点访问权限，或为设置统一的 Systems Manager 控制台时指定的目标子集设置即时节点访问权限。不支持添加在设置统一的 Systems Manager 控制台时未选择的新目标。

接下来，您需创建审批策略，以确定节点连接何时需要手动审批以及何时会自动获得批准。审批策略由组织中的每个账户管理。您还可以共享来自委托管理员账户的策略，从而显式拒绝自动批准与特定节点的连接。

> **Note**
>
> 设置即时节点访问不会影响您已为Session Manager配置的现有 IAM 策略或首选项。您必须从 IAM 策略中移除 `StartSession` 等Session Manager操作的权限，确保当用户尝试连接到您的节点时，仅使用即时节点访问。设置即时节点访问后，建议在移除Session Manager权限之前，使用一个用户和节点子集测试您的审批策略，以验证策略是否按预期运行。

身份验证支持

请注意以下有关用于即时节点访问的身份验证支持的详细信息：

- 使用远程桌面连接到 Windows Server 实例时，即时节点访问不支持单点登录身份验证类型。

- 仅支持 AWS Security Token Service (AWS STS) `AssumeRole` 临时安全凭证。有关更多信息，请参阅 IAM 用户指南中的以下主题：
- • IAM 中的临时安全凭证
  - 比较 AWS STS 凭证
  - 请求临时安全凭证

以下 IAM 策略概述了管理和允许用户使用 Systems Manager 向节点创建即时节点访问请求所需的权限。确认您具有在 Systems Manager 中使用即时节点访问所需的权限后，便可以继续设置过程。将每个*#######*替换为您自己的信息。

用于启用即时节点访问的 IAM 策略

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "QuickSetupConfigurationManagers",
            "Effect": "Allow",
            "Action": [
                "ssm-quicksetup:CreateConfigurationManager",
                "ssm-quicksetup:DeleteConfigurationManager",
                "ssm-quicksetup:GetConfiguration",
                "ssm-quicksetup:GetConfigurationManager",
                "ssm-quicksetup:GetServiceSettings",
                "ssm-quicksetup:ListConfigurationManagers",
                "ssm-quicksetup:ListConfigurations",
                "ssm-quicksetup:ListQuickSetupTypes",
                "ssm-quicksetup:ListTagsForResource",
                "ssm-quicksetup:TagResource",
                "ssm-quicksetup:UntagResource",
                "ssm-quicksetup:UpdateConfigurationDefinition",
                "ssm-quicksetup:UpdateConfigurationManager",
                "ssm-quicksetup:UpdateServiceSettings"
            ],
            "Resource": "*"
        },
        {
            "Sid": "QuickSetupDeployments",
```

```
            "Effect": "Allow",
            "Action": [
                "cloudformation:DescribeStackSetOperation",
                "cloudformation:ListStacks",
                "cloudformation:DescribeStacks",
                "cloudformation:DescribeStackResources",
                "cloudformation:ListStackSetOperations",
                "cloudformation:ListStackInstances",
                "cloudformation:DescribeStackSet",
                "cloudformation:ListStackSets",
                "cloudformation:DescribeStackInstance",
                "cloudformation:DescribeOrganizationsAccess",
                "cloudformation:ActivateOrganizationsAccess",
                "cloudformation:GetTemplate",
                "cloudformation:ListStackSetOperationResults",
                "cloudformation:DescribeStackEvents",
                "cloudformation:UntagResource",
                "ssm:DescribeAutomationExecutions",
                "ssm:GetAutomationExecution",
                "ssm:ListAssociations",
                "ssm:DescribeAssociation",
                "ssm:GetDocument",
                "ssm:ListDocuments",
                "ssm:DescribeDocument",
                "ssm:GetOpsSummary",
                "organizations:DeregisterDelegatedAdministrator",
                "organizations:DescribeAccount",
                "organizations:DescribeOrganization",
                "organizations:ListDelegatedAdministrators",
                "organizations:ListRoots",
                "organizations:ListParents",
                "organizations:ListOrganizationalUnitsForParent",
                "organizations:DescribeOrganizationalUnit",
                "organizations:ListAWSServiceAccessForOrganization",
                "iam:ListRoles",
                "iam:ListRolePolicies",
                "iam:GetRole",
                "iam:CreatePolicy",
                "cloudformation:TagResource"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
```

```
            "Action": [
                "cloudformation:RollbackStack",
                "cloudformation:CreateStack",
                "cloudformation:UpdateStack",
                "cloudformation:DeleteStack"
            ],
            "Resource": [
                "arn:aws:cloudformation:*:*:stack/StackSet-AWS-QuickSetup-
JITNA*",
                "arn:aws:cloudformation:*:*:stack/AWS-QuickSetup-*",
                "arn:aws:cloudformation:*:*:type/resource/*",
                "arn:aws:cloudformation:*:*:stack/StackSet-SSMQuickSetup"
            ]
        },
        {
            "Sid": "StackSetOperations",
            "Effect": "Allow",
            "Action": [
                "cloudformation:CreateStackSet",
                "cloudformation:UpdateStackSet",
                "cloudformation:DeleteStackSet",
                "cloudformation:DeleteStackInstances",
                "cloudformation:CreateStackInstances",
                "cloudformation:StopStackSetOperation"
            ],
            "Resource": [
                "arn:aws:cloudformation:*:*:stackset/AWS-QuickSetup-JITNA*",
                "arn:aws:cloudformation:*:*:type/resource/*",
                "arn:aws:cloudformation:*:*:stackset-target/AWS-QuickSetup-
JITNA*:*"
            ]
        },
        {
            "Sid": "IamRolesMgmt",
            "Effect": "Allow",
            "Action": [
                "iam:CreateRole",
                "iam:DeleteRole",
                "iam:GetRole",
                "iam:AttachRolePolicy",
                "iam:PutRolePolicy",
                "iam:DetachRolePolicy",
                "iam:GetRolePolicy",
                "iam:ListRolePolicies"
```

```
            ],
            "Resource": [
                "arn:aws:iam::*:role/AWS-QuickSetup-JITNA*",
                "arn:aws:iam::*:role/service-role/AWS-QuickSetup-JITNA*"
            ]
        },
        {
            "Sid": "IamPassRole",
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:iam::*:role/AWS-QuickSetup-JITNA*",
                "arn:aws:iam::*:role/service-role/AWS-QuickSetup-JITNA*"
            ],
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ssm.amazonaws.com",
                        "ssm-quicksetup.amazonaws.com",
                        "cloudformation.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Sid": "SSMAutomationExecution",
            "Effect": "Allow",
            "Action": "ssm:StartAutomationExecution",
            "Resource": "arn:aws:ssm:us-east-1:111122223333:automation-
definition/AWS-EnableExplorer:*"
        },
        {
            "Sid": "SSMAssociationPermissions",
            "Effect": "Allow",
            "Action": [
                "ssm:DeleteAssociation",
                "ssm:CreateAssociation",
                "ssm:StartAssociationsOnce"
            ],
            "Resource": "arn:aws:ssm:us-east-1:111122223333:association/*"
        },
        {
```

```
            "Sid": "SSMResourceDataSync",
            "Effect": "Allow",
            "Action": [
                "ssm:CreateResourceDataSync",
                "ssm:UpdateResourceDataSync"
            ],
            "Resource": "arn:aws:ssm:us-east-1:111122223333:resource-data-sync/
AWS-QuickSetup-*"
        },
        {
            "Sid": "ListResourceDataSync",
            "Effect": "Allow",
            "Action": [
                "ssm:ListResourceDataSync"
            ],
            "Resource": "*"
        },
        {
            "Sid": "CreateServiceLinkedRoles",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole"
            ],
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": [
                        "accountdiscovery.ssm.amazonaws.com",
                        "ssm.amazonaws.com",
                        "ssm-quicksetup.amazonaws.com",
                        "stacksets.cloudformation.amazonaws.com"
                    ]
                }
            },
            "Resource": "*"
        },
        {
            "Sid": "CreateStackSetsServiceLinkedRole",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole"
            ],
            "Resource": "arn:aws:iam::*:role/aws-
service-role/stacksets.cloudformation.amazonaws.com/
AWSServiceRoleForCloudFormationStackSetsOrgAdmin"
```

```
        },
        {
            "Sid": "AllowSsmJitnaPoliciesCrudOperations",
            "Effect": "Allow",
            "Action": [
                "ssm:CreateDocument",
                "ssm:UpdateDocument",
                "ssm:UpdateDocumentDefaultVersion",
                "ssm:GetDocument",
                "ssm:DescribeDocument",
                "ssm:DeleteDocument"
            ],
            "Resource": [
                "arn:aws:ssm:us-east-1:111122223333:document/SSM-
JustInTimeAccessDenyAccessOrgPolicy"
            ],
            "Condition": {
                "StringEquals": {
                    "ssm:DocumentType": [
                        "AutoApprovalPolicy"
                    ]
                }
            }
        },
        {
            "Sid": "AllowAccessRequestOpsItemOperations",
            "Effect": "Allow",
            "Action": [
                "ssm:GetOpsItem",
                "ssm:DescribeOpsItems",
                "ssm:GetOpsSummary",
                "ssm:DeleteOpsItem",
                "ssm:ListOpsItemEvents"
            ],
            "Resource": "*"
        },
        {
            "Sid": "IdentityCenterPermissions",
            "Effect": "Allow",
            "Action": [
                "sso:DescribeRegisteredRegions",
                "sso:ListDirectoryAssociations",
                "identitystore:GetUserId",
                "identitystore:DescribeUser",
```

```
                "identitystore:DescribeGroup",
                "identitystore:ListGroupMembershipsForMember"
            ],
            "Resource": "*"
        }
    ]
}
```

用于配置即时节点访问的 IAM 策略

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSsmJitnaPoliciesCrudOperations",
            "Effect": "Allow",
            "Action": [
                "ssm:CreateDocument",
                "ssm:UpdateDocument",
                "ssm:UpdateDocumentDefaultVersion",
                "ssm:GetDocument",
                "ssm:DescribeDocument",
                "ssm:DeleteDocument"
            ],
            "Resource": [
                "arn:aws:ssm:us-east-1:111122223333:document/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ssm:DocumentType": [
                        "ManualApprovalPolicy",
                        "AutoApprovalPolicy"
                    ]
                }
            }
        },
        {
            "Sid": "AllowSsmJitnaPoliciesListOperations",
            "Effect": "Allow",
            "Action": [
```

```
                "ssm:ListDocuments",
                "ssm:ListDocumentVersions"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::111122223333:role/SSM-
JustInTimeAccessTokenRole",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "justintimeaccess.ssm.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Sid": "AllowAccessRequestOpsItemOperations",
            "Effect": "Allow",
            "Action": [
                "ssm:GetOpsItem",
                "ssm:DescribeOpsItems",
                "ssm:GetOpsSummary",
                "ssm:DeleteOpsItem",
                "ssm:ListOpsItemEvents"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowSessionManagerPreferencesOperation",
            "Effect": "Allow",
            "Action": [
                "ssm:CreateDocument",
                "ssm:GetDocument",
                "ssm:DescribeDocument",
                "ssm:UpdateDocument",
                "ssm:DeleteDocument"
            ],
            "Resource": "arn:aws:ssm:us-east-1:111122223333:document/SSM-
SessionManagerRunShell",
            "Condition": {
                "StringEquals": {
```

```
                    "ssm:DocumentType": "Session"
                }
            }
        },
        {
            "Sid": "AllowSessionManagerOperations",
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeSessions",
                "ssm:GetConnectionStatus",
                "ssm:TerminateSession"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowRDPConnectionRecordingOperations",
            "Effect": "Allow",
            "Action": [
                "ssm-guiconnect:UpdateConnectionRecordingPreferences",
                "ssm-guiconnect:GetConnectionRecordingPreferences",
                "ssm-guiconnect:DeleteConnectionRecordingPreferences"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowRDPConnectionRecordingKmsOperation",
            "Effect": "Allow",
            "Action": [
                "kms:CreateGrant"
            ],
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/SystemsManagerJustInTimeNodeAccessManaged":
"true"
                },
                "StringLike": {
                    "kms:ViaService": "ssm-guiconnect.*.amazonaws.com"
                },
                "Bool": {
                    "aws:ViaAWSService": "true"
                }
            }
        },
```

```
        {
            "Sid": "AllowFleetManagerOperations",
            "Effect": "Allow",
            "Action": [
                "ssm-guiconnect:GetConnection",
                "ssm-guiconnect:ListConnections"
            ],
            "Resource": "*"
        },
        {
            "Sid": "SNSTopicManagement",
            "Effect": "Allow",
            "Action": [
                "sns:CreateTopic",
                "sns:SetTopicAttributes"
            ],
            "Resource": [
                "arn:aws:sns:us-east-1:111122223333:SSM-JITNA*"
            ]
        },
        {
            "Sid": "SNSListTopics",
            "Effect": "Allow",
            "Action": [
                "sns:ListTopics"
            ],
            "Resource": "*"
        },
        {
            "Sid": "EventBridgeRuleManagement",
            "Effect": "Allow",
            "Action": [
                "events:PutRule",
                "events:PutTargets"
            ],
            "Resource": [
                "arn:aws:events:us-east-1::rule/SSM-JITNA*"
            ]
        },
        {
            "Sid": "ChatbotSlackManagement",
            "Effect": "Allow",
            "Action": [
                "chatbot:CreateSlackChannelConfiguration",
```

```
                "chatbot:UpdateSlackChannelConfiguration",
                "chatbot:DescribeSlackChannelConfigurations",
                "chatbot:DescribeSlackWorkspaces",
                "chatbot:DeleteSlackChannelConfiguration",
                "chatbot:RedeemSlackOauthCode",
                "chatbot:DeleteSlackWorkspaceAuthorization",
                "chatbot:GetSlackOauthParameters"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ChatbotTeamsManagement",
            "Effect": "Allow",
            "Action": [
                "chatbot:ListMicrosoftTeamsChannelConfigurations",
                "chatbot:CreateMicrosoftTeamsChannelConfiguration",
                "chatbot:UpdateMicrosoftTeamsChannelConfiguration",
                "chatbot:ListMicrosoftTeamsConfiguredTeams",
                "chatbot:DeleteMicrosoftTeamsChannelConfiguration",
                "chatbot:RedeemMicrosoftTeamsOauthCode",
                "chatbot:DeleteMicrosoftTeamsConfiguredTeam",
                "chatbot:GetMicrosoftTeamsOauthParameters",
                "chatbot:TagResource"
            ],
            "Resource": "*"
        },
        {
            "Sid": "SSMEmailSettings",
            "Effect": "Allow",
            "Action": [
                "ssm:UpdateServiceSetting",
                "ssm:GetServiceSetting"
            ],
            "Resource": [
                "arn:aws:ssm:us-east-1:111122223333:servicesetting/ssm/access-
request/email-role-mapping",
                "arn:aws:ssm:us-east-1:111122223333:servicesetting/ssm/access-
request/enabled-email-notifications"
            ]
        },
        {
            "Sid": "AllowViewingJitnaCloudWatchMetrics",
            "Effect": "Allow",
            "Action": [
```

```
                "cloudwatch:GetMetricData",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "cloudwatch:namespace": "AWS/SSM/JustInTimeAccess"
                }
            }
        },
        {
            "Sid": "QuickSetupConfigurationManagers",
            "Effect": "Allow",
            "Action": [
                "ssm-quicksetup:ListConfigurationManagers",
                "ssm-quicksetup:ListConfigurations",
                "ssm-quicksetup:ListQuickSetupTypes",
                "ssm-quicksetup:GetConfiguration",
                "ssm-quicksetup:GetConfigurationManager"
            ],
            "Resource": "*"
        },
        {
            "Sid": "QuickSetupDeployments",
            "Effect": "Allow",
            "Action": [
                "cloudformation:ListStacks",
                "cloudformation:DescribeStacks",
                "organizations:DescribeOrganization",
                "organizations:ListDelegatedAdministrators"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ManualPolicy",
            "Effect": "Allow",
            "Action": [
                "sso:DescribeRegisteredRegions",
                "ssm:GetServiceSetting",
                "iam:ListRoles"
            ],
            "Resource": "*"
        },
```

```
        {
            "Sid": "SessionPreference",
            "Effect": "Allow",
            "Action": [
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowIamListForKMS",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers"
            ],
            "Resource": "arn:aws:iam::111122223333:user/*"
        },
        {
            "Sid": "KMSPermission",
            "Effect": "Allow",
            "Action": [
                "kms:TagResource",
                "kms:ListAliases",
                "kms:CreateAlias"
            ],
            "Resource": "*"
        },
        {
            "Sid": "KMSCreateKey",
            "Effect": "Allow",
            "Action": [
                "kms:CreateKey"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/SystemsManagerJustInTimeNodeAccessManaged":
"true"
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "SystemsManagerJustInTimeNodeAccessManaged"
                    ]
                }
            }
```

```
        },
        {
            "Sid": "AllowIamRoleForChatbotAction",
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::111122223333:role/role name",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "chatbot.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Sid": "AllowIamServiceRoleForChat",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole"
            ],
            "Resource": "arn:aws:iam::111122223333:role/aws-service-role/
management.chatbot.amazonaws.com/AWSServiceRoleForAWSChatbot"
        },
        {
            "Sid": "CloudWatchLogs",
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups"
            ],
            "Resource": "arn:aws:logs:*:111122223333:log-group::log-stream:"
        },
        {
            "Sid": "IdentityStorePermissions",
            "Effect": "Allow",
            "Action": [
                "sso:ListDirectoryAssociations",
                "identitystore:GetUserId",
                "sso-directory:SearchUsers",
                "sso-directory:SearchGroups",
                "identitystore:DescribeGroup",
                "identitystore:DescribeUser"
            ],
```

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

适用于访问请求审批者的 IAM 策略

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccessRequestDescriptions",
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeOpsItems",
                "ssm:GetOpsSummary",
                "ssm:ListOpsItemEvents"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowGetSpecificAccessRequest",
            "Effect": "Allow",
            "Action": [
                "ssm:GetOpsItem"
            ],
            "Resource": "arn:aws:ssm:us-east-1:111122223333:opsitem/*"
        },
        {
            "Sid": "AllowApprovalRejectionSignal",
            "Effect": "Allow",
            "Action": [
                "ssm:SendAutomationSignal"
            ],
            "Resource": "arn:aws:ssm:*:*:automation-execution/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/SystemsManagerJustInTimeNodeAccessManaged":
 "true"
                }
```

```
            }
        },
        {
            "Sid": "QuickSetupConfigurationManagers",
            "Effect": "Allow",
            "Action": [
                "ssm-quicksetup:ListConfigurationManagers",
                "ssm-quicksetup:ListConfigurations",
                "ssm-quicksetup:GetConfigurationManager",
                "ssm-quicksetup:ListQuickSetupTypes",
                "ssm-quicksetup:GetConfiguration"
            ],
            "Resource": "*"
        },
        {
            "Sid": "QuickSetupDeployments",
            "Effect": "Allow",
            "Action": [
                "cloudformation:ListStacks",
                "cloudformation:DescribeStacks",
                "organizations:DescribeOrganization",
                "organizations:ListDelegatedAdministrators"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowSsmJitnaPoliciesCrudOperations",
            "Effect": "Allow",
            "Action": [
                "ssm:GetDocument",
                "ssm:DescribeDocument"
            ],
            "Resource": [
                "arn:aws:ssm:us-east-1:111122223333:document/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ssm:DocumentType": [
                        "ManualApprovalPolicy",
                        "AutoApprovalPolicy"
                    ]
                }
            }
        },
```

```
        {
            "Sid": "AllowSsmJitnaPoliciesListOperations",
            "Effect": "Allow",
            "Action": [
                "ssm:ListDocuments",
                "ssm:ListDocumentVersions"
            ],
            "Resource": "*"
        },
        {
            "Sid": "IDCPermissions",
            "Effect": "Allow",
            "Action": [
                "sso:DescribeRegisteredRegions",
                "sso:ListDirectoryAssociations",
                "identitystore:GetUserId",
                "identitystore:DescribeUser",
                "identitystore:DescribeGroup",
                "identitystore:ListGroupMembershipsForMember"
            ],
            "Resource": "*"
        }
    ]
}
```

适用于即时节点访问用户的 IAM 策略

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowJITNAOperations",
            "Effect": "Allow",
            "Action": [
                "ssm:StartAccessRequest",
                "ssm:GetAccessToken"
            ],
            "Resource": "*"
        },
        {
```

```
            "Sid": "AllowOpsItemCreationAndRetrieval",
            "Effect": "Allow",
            "Action": [
                "ssm:CreateOpsItem",
                "ssm:GetOpsItem"
            ],
            "Resource": "arn:aws:ssm:*:*:opsitem/*"
        },
        {
            "Sid": "AllowListAccessRequests",
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeOpsItems",
                "ssm:GetOpsSummary",
                "ssm:ListOpsItemEvents",
                "ssm:DescribeSessions"
            ],
            "Resource": "*"
        },
        {
            "Sid": "RequestManualApprovals",
            "Action": "ssm:StartAutomationExecution",
            "Effect": "Allow",
            "Resource": "arn:aws:ssm:*:*:document/*",
            "Condition": {
                "StringEquals": {
                    "ssm:DocumentType": "ManualApprovalPolicy"
                }
            }
        },
        {
            "Sid": "StartManualApprovalsAutomationExecution",
            "Effect": "Allow",
            "Action": "ssm:StartAutomationExecution",
            "Resource": "arn:aws:ssm:*:*:automation-execution/*"
        },
        {
            "Sid": "AllowManualApprovalAutomationExecutionTagging",
            "Effect": "Allow",
            "Action": [
                "ssm:AddTagsToResource"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:automation-execution/*"
```

```
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/SystemsManagerJustInTimeNodeAccessManaged":
"true"
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "SystemsManagerJustInTimeNodeAccessManaged"
                    ]
                }
            }
        },
        {
            "Sid": "CancelAccessRequestManualApproval",
            "Effect": "Allow",
            "Action": "ssm:StopAutomationExecution",
            "Resource": "arn:aws:ssm:*:*:automation-execution/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/SystemsManagerJustInTimeNodeAccessManaged":
"true"
                }
            }
        },
        {
            "Sid": "DescribeEC2Instances",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeTags",
                "ec2:GetPasswordData"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowListSSMManagedNodesAndTags",
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeInstanceInformation",
                "ssm:ListTagsForResource"
            ],
            "Resource": "*"
        },
```

```
        {
            "Sid": "QuickSetupConfigurationManagers",
            "Effect": "Allow",
            "Action": [
                "ssm-quicksetup:ListConfigurationManagers",
                "ssm-quicksetup:GetConfigurationManager",
                "ssm-quicksetup:ListConfigurations",
                "ssm-quicksetup:ListQuickSetupTypes",
                "ssm-quicksetup:GetConfiguration"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowSessionManagerOperations",
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeSessions",
                "ssm:GetConnectionStatus"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowRDPOperations",
            "Effect": "Allow",
            "Action": [
                "ssm-guiconnect:ListConnections",
                "ssm:GetConnectionStatus"
            ],
            "Resource": "*"
        },
        {
            "Sid": "QuickSetupDeployments",
            "Effect": "Allow",
            "Action": [
                "cloudformation:ListStacks",
                "cloudformation:DescribeStacks",
                "organizations:DescribeOrganization",
                "organizations:ListDelegatedAdministrators"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowSsmJitnaPoliciesReadOnly",
            "Effect": "Allow",
```

```
            "Action": [
                "ssm:GetDocument",
                "ssm:DescribeDocument"
            ],
            "Resource": [
                "arn:aws:ssm:*:111122223333:document/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ssm:DocumentType": [
                        "ManualApprovalPolicy",
                        "AutoApprovalPolicy"
                    ]
                }
            }
        },
        {
            "Sid": "AllowSsmJitnaPoliciesListOperations",
            "Effect": "Allow",
            "Action": [
                "ssm:ListDocuments",
                "ssm:ListDocumentVersions"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ExploreNodes",
            "Effect": "Allow",
            "Action": [
                "ssm:ListNodesSummary",
                "ssm:ListNodes",
                "ssm:DescribeInstanceProperties"
            ],
            "Resource": "*"
        },
        {
            "Sid": "IdentityStorePermissions",
            "Effect": "Allow",
            "Action": [
                "sso:DescribeRegisteredRegions",
                "sso:ListDirectoryAssociations",
                "identitystore:GetUserId",
                "identitystore:DescribeUser",
                "identitystore:DescribeGroup"
```

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

> **Note**
>
> 要限制对创建、更新或删除审批策略的 API 操作的访问权限，请使用 `AutoApprovalPolicy` 和 `ManualApprovalPolicy` 文档类型的 `ssm:DocumentType` 条件键。`StartAccessRequest` 和 `GetAccessToken` API 操作不支持以下全局上下文键：
>
> - aws:ViaAwsService
> - aws:MultiFactorAuthPresent
> - aws:SourceVpce
> - aws:UserAgent

有关 Systems Manager 的条件上下文键的更多信息，请参阅《Service Authorization Reference》中的 Condition keys for AWS Systems Manager。

以下过程介绍如何完成即时节点访问的第一个设置步骤。

设置即时节点访问

1. 登录贵组织的 Systems Manager 委派管理员账户。
2. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
3. 在导航窗格中选择即时节点访问。
4. 选择启用新体验。
5. 选择您想要启用即时节点访问的区域。默认情况下，系统会为即时节点访问选择您在设置统一 Systems Manager 控制台时选择的相同区域。不支持选择在设置统一 Systems Manager 控制台时未选择的新区域。
6. 选择启用即时节点访问。

启用该功能后，可在 30 天内免费使用即时节点访问。30 天试用期后，需付费才能使用即时节点访问。有关更多信息，请参阅 AWS Systems Manager定价。

为节点创建审批策略

审批策略定义了用户访问节点所需的批准。由于即时节点访问无需通过 IAM 策略获得节点的长期权限，因此您必须创建审批策略来允许访问您的节点。如果没有适用于节点的审批策略，则用户将无法请求访问该节点。

在即时节点访问中，有三种类型的策略。策略类型包括自动审批、拒绝访问和手动审批。

即时节点访问策略类型

- 自动审批策略定义了用户可以自动连接到哪些节点。
- 手动审批策略定义了访问您指定的节点时必须提供的手动审批的数量和级别。
- 拒绝访问策略明确禁止对您指定节点的访问请求的自动审批。

拒绝访问策略适用于 AWS Organizations 组织中的所有账户。例如，您可以明确拒绝 `Intern` 组对使用 `Production` 键标记的节点的自动审批。自动审批和手动审批策略仅适用于其创建所在的 AWS 账户和 AWS 区域。组织中的每个成员账户都管理自己的审批策略。审批策略按以下顺序评估：

1. 拒绝访问
2. 自动审批
3. 手动

虽然每个组织只能有一个拒绝访问策略，且每个账户和地区只能有一个自动审批策略，但一个账户中可能有多个手动审批策略。在评估手动审批策略时，即时节点访问始终偏向于针对节点的更具体的策略。手动审批策略按以下顺序评估：

1. 特定标签的目标
2. 所有节点目标

例如，您有一个使用 Demo 键标记的节点。在同一个账户中，您有一个针对所有节点的手动审批策略，并且需要从一个级别获得一次审批。您还有一个手动审批策略，对于使用 Demo 键标记的节点，需要从两个级别获得两次审批。Systems Manager 会将针对 Demo 标签的策略应用于节点，因为该策略比针对所有节点的策略更为具体。因此您可以为账户中的所有节点创建通用策略，确保用户可以提交访问请求，同时使您能够根据需要创建更精细的策略。

根据您的组织，可能有多个标签应用于您的节点。在这种情况下，如果多个手动审批策略应用于一个节点，则访问请求将失败。例如，节点使用 Production 和 Database 键进行标记。在同一个账户中，有一个适用于使用 Production 键标记的节点的手动审批策略，以及另一个适用于使用 Database 键标记的节点的手动审批策略。这会导致使用两个键标记的节点发生冲突，因此访问请求会失败。Systems Manager 会将用户重定向到失败的请求。用户可以在此处可以查看有关冲突策略和标签的详细信息，以便在拥有所需权限时可以进行必要的调整。如果没有必要的权限，可以通知组织中具有必要权限的同事修改策略。导致访问请求失败的策略冲突会触发 EventBridge 事件，这有助于您灵活地构建自己的响应工作流。此外，对于导致访问请求失败的策略冲突，Systems Manager 会向您指定的收件人发送电子邮件通知。有关为策略冲突配置电子邮件通知的更多信息，请参阅为即时访问请求配置通知。

在拒绝访问策略中，您可以使用 Cedar 策略语言来定义用户在组织中明确无法自动连接到的节点。此策略是通过组织委派的管理员账户创建和共享的。拒绝访问策略取代所有自动审批策略。您在每个组织只能有一个拒绝访问策略。

在自动审批策略中，您可以使用 Cedar 策略语言来定义哪些用户无需手动审批即可自动连接到指定节点。自动审批批准的访问请求的访问持续时间为 1 小时。此值不能更改。您在每个账户和区域只能有一个自动审批策略。

在手动审批策略中，您可以指定访问持续时间、需要多少级别的批准、每个级别所需的审批者数量以及他们可以批准即时访问请求的节点。手动审批策略的访问持续时间必须介于 1 到 336 小时之间。如果您指定多个批准级别，则访问请求的批准将一次处理一个级别。这意味着必须在您对一个级别要求的所有批准都提供后，批准流程才能进入后续级别。如果您在手动审批策略中指定多个标签，则这些标签将作为 or 语句而不是 and 语句进行评估。例如，如果您创建包含标签 Application、Web 和 Test 的手动审批策略，则该策略将应用于使用其中一个键标记的任何节点。该策略不仅适用于使用全部三个键标记的节点。

我们建议您使用手动策略与自动审批策略的组合来帮助您保护包含更多关键数据的节点，同时允许用户无需干预即可连接到不太重要的节点。例如，您可以要求手动审批对数据库节点的访问请求，以及自动审批非持久性表示层节点的会话。

以下过程介绍如何创建即时节点访问的审批策略。

主题

- 为即时节点访问创建手动审批策略
- 自动审批和拒绝访问策略的语句结构和内置运算符
- 为即时节点访问创建自动审批策略
- 为即时节点访问创建拒绝访问策略

- 使用 Amazon Q 为即时节点访问创建审批策略

为即时节点访问创建手动审批策略

以下过程介绍如何创建手动审批策略。Systems Manager 允许您为每个 AWS 账户和 AWS 区域创建最多 50 个手动审批策略。

创建手动审批策略

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中选择管理节点访问。
3. 在创建手动审批策略步骤的策略详细信息部分，输入审批策略的名称和描述。
4. 输入访问持续时间的值。这是访问请求获得批准后，用户可以启动与节点的会话的最长时间。该值必须在 1 到 336 之间。
5. 在节点目标部分中，输入与您要应用策略的节点关联的标签键值对。如果策略中指定的标签均未与节点相关联，则策略不会应用于该节点。
6. 在访问请求审批人部分，输入策略中您希望能够批准哪些用户或群组对节点目标的访问请求。访问请求审批人可以是 IAM Identity Center 用户和群组或 IAM 角色。每个级别最多可以指定 5 个审批人，最多可以指定 5 个审批人级别。
7. 选择创建手动审批策略。

自动审批和拒绝访问策略的语句结构和内置运算符

下表展示了自动审批和拒绝访问策略的结构。

组件	语法
效果	`permit | forbid`
范围	`(principal, action, resource)`
条件子句	`when { principal or resource has attribute name };`

策略组件

自动审批或拒绝访问策略包含以下部分：

- 效果 – permit（允许）或 forbid（拒绝）访问。
- 范围 – 指定效果适用于哪些主体、操作和资源。您可以通过不标识特定主体、操作或资源来使 Cedar 中的范围保持未定义状态。在这种情况下，策略适用于所有可能的主体、操作和资源。对于即时访问节点，action 始终为 AWS::SSM::Action::"getTokenForInstanceAccess"。
- 条件子句 – 应用效果的上下文。

评论

您可以在 策略中包含注释。注释被定义为以 // 开头、以换行符结尾的一行。

以下示例显示了策略中的注释。

```
// Allows users in the Engineering group from the Platform org to automatically connect
 to nodes tagged with Engineering and Production keys.
permit (
    principal in AWS::IdentityStore::Group::"d4q81745-r081-7079-d789-14da1EXAMPLE",
    action == AWS::SSM::Action::"getTokenForInstanceAccess",
    resource
)
when {
    principal has organization && resource.hasTag("Engineering") &&
 resource.hasTag("Production") && principal.organization == "Platform"
};
```

多子句

您可以利用 && 运算符，在一个策略声明中使用多个条件子句。

```
// Allow access if node has tag where the tag key is Environment
// & tag value is Development

permit(principal, action == AWS::SSM::getTokenForInstanceAccess, resource)
when {
    resource.hasTag("Environment") &&
    resource.getTag("Environment") == "Development"
};
```

预留字符

以下示例说明当上下文属性使用 :（分号）时如何编写策略，该符号是策略语言中的保留字符。

```
permit (
    principal,
    action == AWS::SSM::Action::"getTokenForInstanceAccess",
    resource
)
when {
    principal has employeeNumber && principal.employeeNumber like "E-1*" &&
 resource.hasTag("Purpose") && resource.getTag("Purpose") == "Testing"
}
```

有关其他示例，请参阅 示例策略语句。

即时节点访问架构

以下是即时节点访问的 Cedar 架构。

```
namespace AWS::EC2 {
    entity Instance tags String;
}


namespace AWS::IdentityStore {
    entity Group;

    entity User in [Group] {
    employeeNumber?: String,
    costCenter?: String,
    organization?: String,
    division?: String,
    };

}


namespace AWS::IAM {

    entity Role;

    type AuthorizationContext = {
```

```
        principalTags: PrincipalTags,
    };

    entity PrincipalTags tags String;
}

namespace AWS::SSM {

    entity ManagedInstance tags String;

    action "getTokenForInstanceAccess" appliesTo {
    principal: [AWS::IdentityStore::User],
    resource: [AWS::EC2::Instance, AWS::SSM::ManagedInstance],
    context: {
        "iam": AWS::IAM::AuthorizationContext
        }
    };
}
```

内置运算符

在使用各种条件创建自动审批或拒绝访问策略的上下文时，您可以使用 && 运算符来添加其他条件。您还可以使用许多其他内置运算符来为您的策略条件添加更多的表达能力。下表包含所有内置运算符，以供参考。

运算符	类型和重载	描述
!	Boolean → Boolean	逻辑非。
==	any → any	等于。适用于任何类型的参数，即使类型不匹配。不同类型的值永远不会彼此相等。
!=	any → any	不等于；与等于完全相反（见上文）。
<	(long, long) → Boolean	长整数小于。
<=	(long, long) → Boolean	长整数小于或等于。
>	(long, long) → Boolean	长整数大于。

运算符	类型和重载	描述
>=	(long, long) → Boolean	长整数大于或等于。
in	(entity, entity) → Boolean	层次结构隶属（自反：A in A 始终为真）。
	(entity, set(entity)) → Boolean	层次结构隶属：A in [B, C, ...] 为真，如果 (A and B) || (A in C) || … 错误，如果集合包含非实体。
&&	(Boolean, Boolean) → Boolean	逻辑与（短路）。
||	(Boolean, Boolean) → Boolean	逻辑或（短路）。
.exists()	entity → Boolean	实体存在。
has	(entity, attribute) → Boolean	中缀运算符。`e has f` 测试记录或实体 `e` 是否具有属性 `f` 的绑定。如果 `e` 不存在或者 `e` 存在但没有属性 `f`，则返回 `false`。属性可以表示为标识符或字符串文字。
like	(string, string) → Boolean	中缀运算符。`t like p` 检查文本 `t` 是否与模式 `p` 匹配，其中可能包含与 0 个或多个任意字符匹配的通配符 `*`。为了匹配 `t` 中的文字星形字符，可以在 `p` 中使用特殊的转义字符序列 `\*`。
.hasTag()	(entity, string) → Boolean	检查实体是否应用了指定的标签。
.getTag()	(entity, string) → Boolean	返回指定标签键的值。

运算符	类型和重载	描述
.contains()	(set, any) → Boolean	设置隶属关系（B 是 A 的元素吗）。
.containsAll()	(set, set) → Boolean	测试集合 A 是否包含集合 B 中的所有元素。
.containsAny()	(set, set) → Boolean	测试集合 A 是否包含集合 B 中的任意元素。

示例策略语句

以下是策略语句示例。

```
// Users assuming IAM roles with a principal tag of "Elevated" can automatically access
 nodes tagged with the "Environment" key when the value equals "prod"
permit(principal, action == AWS::SSM::getTokenForInstanceAccess, resource)
when {
    // Verify IAM role principal tag
    context.iam.principalTags.getTag("AccessLevel") == "Elevated" &&

    // Verify the node has a tag with "Environment" tag key and a tag value of "prod"
    resource.hasTag("Environment") &&
    resource.getTag("Environment") == "prod"
};
```

```
// Identity Center users in the "Contractor" division can automatically access nodes
 tagged with the "Environment" key when the value equals "dev"
permit(principal, action == AWS::SSM::getTokenForInstanceAccess, resource)
when {
    // Verify that the user is part of the "Contractor" division
    principal.division == "Contractor" &&

    // Verify the node has a tag with "Environment" tag key and a tag value of "dev"
    resource.hasTag("Environment") &&
    resource.getTag("Environment") == "dev"
};
```

```
// Identity Center users in a specified group can automatically access nodes tagged
 with the "Environment" key when the value equals "Production"
permit(principal in AWS::IdentityStore::Group::"d4q81745-r081-7079-d789-14da1EXAMPLE",
    action == AWS::SSM::getTokenForInstanceAccess,
    resource)
when {
    resource.hasTag("Environment") &&
    resource.getTag("Environment") == "Production"
};
```

为即时节点访问创建自动审批策略

自动审批策略使用 Cedar 策略语言来定义哪些用户无需手动审批即可自动连接到指定节点。自动审批策略包含多个指定 `principal` 和 `resource` 的 `permit` 语句。每个语句都包含一个定义自动审批条件的 `when` 子句。

以下是示例自动审批策略。

```
permit (
    principal in AWS::IdentityStore::Group::"e8c17310-e011-7089-d989-10da1EXAMPLE",
    action == AWS::SSM::Action::"getTokenForInstanceAccess",
    resource
)
when {
    principal has costCenter && resource.hasTag("CostCenter") && principal.costCenter
 == resource.getTag("CostCenter")
};

permit (
    principal in AWS::IdentityStore::Group::"d4q81745-r081-7079-d789-14da1EXAMPLE",
    action == AWS::SSM::Action::"getTokenForInstanceAccess",
    resource
)
when {
    principal has organization && resource.hasTag("Engineering") &&
 resource.hasTag("Production") && principal.organization == "Platform"
};

permit (
    principal,
    action == AWS::SSM::Action::"getTokenForInstanceAccess",
    resource
)
```

```
when {
    principal has employeeNumber && principal.employeeNumber like "E-1*" &&
 resource.hasTag("Purpose") && resource.getTag("Purpose") == "Testing"
};
```

以下过程说明了如何创建即时节点访问的自动审批策略。自动审批批准的访问请求的访问持续时间为 1 小时。此值不能更改。每个 AWS 账户 和 AWS 区域 只能有一个自动审批策略。有关如何构造策略语句的信息，请参阅自动审批和拒绝访问策略的语句结构和内置运算符。

创建自动审批策略

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中选择管理节点访问。
3. 在审批策略选项卡中，选择创建自动审批策略。
4. 在策略语句部分输入自动审批策略的策略语句。您可以使用提供的示例语句来帮助您创建策略。
5. 选择创建自动审批策略。

为即时节点访问创建拒绝访问策略

拒绝访问策略使用 Cedar 策略语言来定义用户未经手动审批时无法自动连接的节点。拒绝访问策略包含多个指定 `principal` 和 `resource` 的 `forbid` 语句。每份语句都包含一个 `when` 子句，定义了明确拒绝自动审批的条件。

下面是一个拒绝访问策略。

```
forbid (
    principal in AWS::IdentityStore::Group::"e8c17310-e011-7089-d989-10da1EXAMPLE",
    action == AWS::SSM::Action::"getTokenForInstanceAccess",
    resource
)
when {
    resource.hasTag("Environment") && resource.getTag("Environment") == "Production"
};

forbid (
    principal,
    action == AWS::SSM::Action::"getTokenForInstanceAccess",
    resource
)
```

```
when {
    principal has division && principal.division != "Finance" &&
 resource.hasTag("DataClassification") && resource.getTag("DataClassification") ==
 "Financial"
};


forbid (
    principal,
    action == AWS::SSM::Action::"getTokenForInstanceAccess",
    resource
)
when {

    principal has employeeNumber && principal.employeeNumber like "TEMP-*" &&
 resource.hasTag("Criticality") && resource.getTag("Criticality") == "High"
};
```

以下过程说明了如何创建即时节点访问的拒绝访问策略。有关如何构造策略语句的信息，请参阅自动审批和拒绝访问策略的语句结构和内置运算符。

> **Note**
>
> 请注意以下信息。
>
> - 您可以通过登录 AWS 管理账户或委派管理员账户来创建拒绝访问策略。您的 AWS Organizations 组织只能有一个拒绝访问策略。
> - 即时节点访问功能使用 AWS Resource Access Manager（AWS RAM）与组织中的成员账户共享拒绝访问策略。如果您想与组织中的成员账户共享拒绝访问策略，则必须使用组织的管理账户启用资源共享。有关更多信息，请参阅《AWS RAM 用户指南》中的允许在 AWS Organizations 内共享资源。

创建拒绝访问策略

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中选择管理节点访问。
3. 在审批策略选项卡中，选择创建拒绝访问策略。
4. 在策略语句部分输入拒绝访问策略的策略语句。您可以使用提供的示例语句来帮助您创建策略。

5. 选择创建拒绝访问策略。

使用 Amazon Q 为即时节点访问创建审批策略

使用适用于命令行的 Amazon Q 开发者版，可在软件开发的各个方面提供指导和支持。对于即时访问节点，Amazon Q 可通过生成和更新策略代码、分析策略语句等来帮助您创建审批策略。以下信息介绍如何使用适用于命令行的 Amazon Q 创建审批策略。

识别您的使用案例

创建审批策略的第一步是明确定义您的使用案例。例如，在您的组织中，您可能希望自动批准对带有 `Environment:Testing` 标签的节点的访问请求。如果员工 ID 以 `TEMP` 开头，您可能还想明确拒绝对带有 `Environment:Production` 标签的节点的自动审批。对于带有 `Tier:Database` 标签的节点，您可能需要两个级别的手动审批。

在任何给定的情况下，您可能会更喜欢某个策略或条件。因此，我们建议您明确定义想要的策略行为，以确定哪些语句最适合您的使用案例和首选项。

设置开发环境

在要制定审批策略的位置安装适用于命令行的 Amazon Q。有关通过命令行安装 Amazon Q 的信息，请参阅《Amazon Q Developer User Guide》中的 Installing Amazon Q for command line。

我们还建议安装 MCP 服务器以获取 AWS 文档。此 MCP 服务器可将适用于命令行的 Amazon Q 连接到最新的文档资源。有关将 MCP 与适用于命令行的 Amazon Q 结合使用的信息，请参阅《Amazon Q Developer User Guide》中的 Using MCP with Amazon Q Developer。

有关 AWS 文档 MCP 服务器的更多信息，请参阅 AWS Documentation MCP Server。

安装并配置 AWS CLI（如果尚未执行该操作）。有关更多信息，请参阅安装或更新最新版本的 AWS CLI。

制定审批策略内容

确定使用案例并设置环境后，您便做好了开发适用于策略内容的准备。您的使用案例和首选项在很大程度上决定了您使用的审批策略和语句的类型。

如果您不确定如何使用特定策略或需要有关策略架构的更多信息，请参阅为节点创建审批策略和后续主题。这些主题详细说明了如何评估策略并提供了工作示例语句。

以下过程介绍如何使用适用于命令行的 Amazon Q 创建审批策略。

> **Note**
>
> Amazon Q 开发者版使用生成式人工智能。您可能需要验证回复。请参阅 AWS 负责任的 AI 政策。

使用适用于命令行的 Amazon Q 创建审批策略

1. 打开您的终端并开始与 Amazon Q 的会话。

```
$ q chat
```

2. 输入提示并描述要创建的策略语句类型。例如，使用前面所述的审批策略场景，您可以输入以下提示：

```
Create a Cedar policy statement that allows a principal to use the
 “getTokenForInstanceAccess" action for a resource
that has a tag key “Environment” and tag value of “Testing”. Use the following
 example statement for reference to verify the syntax.

permit (
    principal in AWS::IdentityStore::Group::"e8c17310-e011-7089-d989-10da1EXAMPLE",
    action == AWS::SSM::Action::"getTokenForInstanceAccess",
    resource
)
when {
    principal has organization && resource.hasTag("Engineering") &&
 resource.hasTag("Production") && principal.organization == "Platform"
};

The Cedar policy statement you create must be valid. Validate the Cedar statement
 using the AWS documentation provided for just-in-time node access policy statement
 structure and make sure you abide by the schema.
```

3. 输入此示例提示会得到以下输出：

```
Based on the AWS documentation for Cedar policy statements and the examples
 provided, I'll create a valid Cedar policy
statement that allows a principal to use the "getTokenForInstanceAccess" action for
 resources with the tag key "Environment" and tag value "Testing".

Here's the Cedar policy statement:
```

```
permit (
    principal,
    action == AWS::SSM::Action::"getTokenForInstanceAccess",
    resource
)
when {
    resource.hasTag("Environment") &&
    resource.getTag("Environment") == "Testing"
};


This policy statement:
1. Uses the permit effect to allow access
2. Applies to any principal (since no specific principal is defined)
3. Specifically targets the "getTokenForInstanceAccess" action
4. Applies to resources that have a tag with key "Environment" and value "Testing"
5. Uses the proper syntax for the condition clause with the hasTag() and getTag()
 operators

The statement follows the Cedar policy language structure as documented in the AWS
 Systems Manager documentation for just-in-time node access policies.
```

4. 将策略语句粘贴到新的 Cedar 文件中以验证语法。您也可以使用 Cedar 操场来验证语句。
5. 验证语句后，保存 Cedar 文件。
6. 在您的终端上，运行以下 AWS CLI 命令，使用您的策略内容创建 SSM 文档。在生产环境中使用审批策略之前，请使用 AWS 账户 和 AWS 区域 中的身份和节点子集测试您的审批策略。

> **Note**
>
> 对于自动审批策略，文档名称必须为 `SSM-JustInTimeAccessAutoApprovalPolicy`。每个 AWS 账户 和 AWS 区域 只能有一个自动审批策略。对于拒绝访问策略，文档名称必须为 `SSM-JustInTimeAccessDenyAccessOrgPolicy`。每个 AWS Organizations 组织只能有一个拒绝访问策略，并且该策略必须在 Systems Manager 的委派管理员账户中创建。手动审批策略的命名限制与其他 SSM 文档相同。有关更多信息，请参阅 CreateDocument。

Linux & macOS

```
aws ssm create-document \
```

```
    --content file://path/to/file/policyContent.cedar \
    --name "SSM-JustInTimeAccessAutoApprovalPolicy" \
    --document-type "AutoApproval"
```

Windows

```
aws ssm create-document ^
    --content file://C:\path\to\file\policyContent.cedar ^
    --name "SSM-JustInTimeAccessAutoApprovalPolicy" ^
    --document-type "AutoApproval"
```

PowerShell

```
$cedar = Get-Content -Path "C:\path\to\file\policyContent.cedar" | Out-String
New-SSMDocument `
    -Content $cedar `
    -Name "SSM-JustInTimeAccessAutoApprovalPolicy" `
    -DocumentType "AutoApproval"
```

更新即时节点访问会话首选项

通过即时节点访问，您可以在组织的每个 AWS 账户和 AWS 区域中指定常规会话和日志记录首选项。或者，您可以使用 AWS CloudFormation StackSets 在多个账户和区域中创建会话首选项文档，帮助您获得一致的会话首选项。有关会话首选项文档的架构的更多信息，请参阅会话文档架构。

出于日志记录目的，我们建议在 Amazon CloudWatch Logs 中使用流选项。此功能允许您向 CloudWatch Logs 发送持续的会话数据日志流。在流式传输会话数据时，将包括必要的详细信息，例如用户在会话中运行的命令、运行命令的用户的 ID，以及将会话数据流式传输到 CloudWatch Logs 的时间戳。流式传输会话数据时，日志采用 JSON 格式，以帮助您与现有的日志记录解决方案相集成。

Systems Manager 不会自动终止即时节点访问会话。作为最佳实践，请指定最长会话持续时间和空闲会话超时设置的值。使用这些设置有助于防止用户保持与节点的连接时间超过访问请求中批准的时间范围。以下过程介绍如何更新即时节点访问的会话首选项。

> **Important**
>
> 您必须使用标签键 `SystemsManagerJustInTimeNodeAccessManaged` 和标签值 `true` 来标记用于即时节点访问中的 Session Manager 加密和 RDP 记录的 AWS KMS 密钥。

有关标记 KMS 密钥的信息，请参阅《AWS Key Management Service 开发人员指南》中的 Tags in AWS KMS。

更新会话首选项

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中选择设置。
3. 选择即时节点访问选项卡。
4. 在会话首选项部分中，选择编辑。
5. 根据需要更新您的常规和日志记录首选项，然后选择保存。

为即时访问请求配置通知

您可以将 Systems Manager 配置为在用户创建即时节点访问请求时，向审批者和请求者的电子邮件地址或聊天客户端发送通知。该通知包含请求者提供的访问请求的原因、AWS 账户、AWS 区域、请求状态以及目标节点的 ID。目前，Systems Manager 通过在聊天应用程序中与 Amazon Q 开发者版集成来支持 Slack 和 Microsoft Teams 客户端。通过聊天客户端使用通知时，访问请求审批者可以直接与访问请求进行交互。这样就无需登录控制台即可对访问请求采取行动。

开始前的准备工作

在为即时节点访问通知配置聊天客户端之前，请注意下面的要求：

- 如果使用 IAM 角色管理账户中的用户身份，则必须手动将要向其发送通知的审批者或请求者的电子邮件地址与关联角色相关联。否则，将无法通过电子邮件通知目标收件人。

以下过程介绍如何为即时节点访问请求配置通知。

为即时节点访问通知配置聊天客户端

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中选择设置。
3. 选择即时节点访问选项卡。
4. 在聊天部分中，选择配置新客户端。

5. 在选择客户端类型下拉列表中，选择要配置的聊天客户端类型，然后选择下一步。
6. 系统会提示您允许聊天应用程序中的 Amazon Q 开发者版访问您的聊天客户端。选择允许。
7. 在配置渠道部分中，输入聊天客户端渠道的信息，然后选择要接收的通知类型。
8. 如果正在配置 Slack 通知，请邀请“@Amazon Q”加入正在配置通知的每个 Slack 通道。
9. 选择配置渠道。

> **Note**
>
> 要允许直接从 Slack 通道批准/拒绝访问请求，请确保使用 Slack 通道配置的 IAM 角色具有 `ssm:SendAutomationSignal` 权限，并且具有包含聊天机器人的信任策略：
>
> ```
> {
>             "Effect": "Allow",
>             "Principal": {
>                 "Service": "chatbot.amazonaws.com"
>             },
>             "Action": "sts:AssumeRole"
> }
> ```

为即时节点访问通知配置电子邮件通知

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中选择设置。
3. 选择即时节点访问选项卡。
4. 在电子邮件部分中，选择编辑。
5. 选择添加电子邮件，然后选择要手动关联电子邮件地址的 IAM 角色。
6. 在电子邮件地址字段中输入电子邮件地址。每当创建需要您指定的 IAM 角色审批的访问请求时，与该角色关联的电子邮件地址都会收到通知。
7. 选择添加电子邮件地址。

录制 RDP 连接

即时节点访问包括录制与 Windows Server 节点建立的 RDP 连接的功能。录制 RDP 连接需要 S3 存储桶和 AWS Key Management Service（AWS KMS）客户自主管理型密钥。当录制数据生成并存储在 Systems Manager 资源上时，AWS KMS key用于临时加密这些数据。客户自主管理型密钥必须是对称密钥，其密钥使用方式为加密和解密。您可以为组织使用多区域密钥，也可以在已启用即时节点访问的每个区域中创建客户自主管理型密钥。

如果您在存储录制的 S3 存储桶上启用了 KMS 加密，则必须向 `ssm-guiconnect` 服务主体提供存储桶加密所用客户自主管理型密钥的访问权限。该客户自主管理型密钥可以与录制设置中指定的密钥不同，录制设置中必须包含建立连接所需的 `kms:CreateGrant` 权限。

配置用于 RDP 录制的 S3 存储桶加密

连接录制将存储在启用 RDP 录制时指定的 S3 存储桶中。

如果将 KMS 密钥作为 S3 存储桶的默认加密机制（SSE-KMS），则必须允许 `ssm-guiconnect` 服务主体访问该密钥的 `kms:GenerateDataKey` 操作。我们建议在将 SSE-KMS 加密用于 S3 存储桶时使用客户自主管理型密钥。这是因为您可以更新客户自主管理型密钥的关联密钥策略。您无法更新 AWS 托管式密钥的密钥策略。

> **Important**
>
> 您必须使用标签键 `SystemsManagerJustInTimeNodeAccessManaged` 和标签值 `true` 来标记用于即时节点访问中的 Session Manager 加密和 RDP 记录的 AWS KMS 密钥。
> 有关标记 KMS 密钥的信息，请参阅《AWS Key Management Service 开发人员指南》中的 Tags in AWS KMS。

使用以下客户自主管理型密钥策略允许 `ssm-guiconnect` 服务访问 S3 存储的 KMS 密钥。有关更新客户自主管理型密钥的信息，请参阅《AWS Key Management Service 开发人员指南》中的 Change a key policy。

将每个*#######*替换为您自己的信息：

- *account-id* 表示发起连接的 AWS 账户 的 ID。
- *region* 表示 S3 存储桶所在的 AWS 区域。（如果存储桶将接收来自多个区域的录音，则可以使用 `*`。示例：`s3.*.amazonaws.com`。）

> **Note**
>
> 如果账户属于 AWS Organizations 中的某个组织，则可以在策略中使用 `aws:SourceOrgID`，而不是 `aws:SourceAccount`。

```
{
    "Sid": "Allow the GUI Connect service principal to access S3",
    "Effect": "Allow",
    "Principal": {
        "Service": "ssm-guiconnect.amazonaws.com"
    },
    "Action": [
        "kms:GenerateDataKey*"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:SourceAccount": "account-id"
        },
        "StringLike": {
            "kms:ViaService": "s3.region.amazonaws.com"
        }
    }
}
```

配置 IAM 权限以录制 RDP 连接

除了即时节点访问所需的 IAM 权限外，根据您需要执行的任务，您使用的用户或角色还必须拥有以下权限。

配置连接录制的权限

要配置 RDP 连接录制，需要以下权限：

- `ssm-guiconnect:UpdateConnectionRecordingPreferences`
- `ssm-guiconnect:GetConnectionRecordingPreferences`
- `ssm-guiconnect:DeleteConnectionRecordingPreferences`
- `kms:CreateGrant`

启动连接的权限

要使用即时节点访问来建立 RDP 连接，需要以下权限：

- `ssm-guiconnect:CancelConnection`
- `ssm-guiconnect:GetConnection`
- `ssm-guiconnect:StartConnection`
- `kms:CreateGrant`

开始前的准备工作

要存储连接录制内容，必须先创建 S3 存储桶并添加以下存储桶策略。将每个*#######*替换为您自己的信息。

（有关添加存储桶策略的更多信息，请参阅《Amazon Simple Storage Service 用户指南》中的使用 Amazon S3 控制台添加存储桶策略。）

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ConnectionRecording",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ssm-guiconnect.amazonaws.com"
                ]
            },
            "Action": "s3:PutObject",
            "Resource": [
                "arn:aws:s3:::",
                "arn:aws:s3:::/*"
            ],
            "Condition":{
            "StringEquals":{
                "aws:SourceAccount":"111122223333"
                }
            }
```

```
        }
    ]
}
```

启用和配置 RDP 连接录制

以下过程介绍如何启用和配置 RDP 连接录制。

启用和配置 RDP 连接录制

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中选择设置。
3. 选择即时节点访问选项卡。
4. 在 RDP 录制部分中，选择启用 RDP 录制。
5. 请选择要将会话录制内容上传到的 S3 存储桶。
6. 选择在录制数据生成并存储在 Systems Manager 资源上时要用于临时加密录制数据的客户自主管理型密钥。（这可以与您用于加密存储桶的客户自主管理型密钥不同。）
7. 选择保存。

RDP 连接录制状态值

RPD 连接录制的有效状态值如下：

- `Recording`：正在录制连接
- `Processing`：连接终止后正在处理视频。
- `Finished`：最终成功状态；连接录制视频处理成功并上已传到指定存储桶。
- `Failed`：最终失败状态。连接录制未成功。
- `ProcessingError`：视频处理过程中出现一个或多个中间故障/错误。可能的原因包括服务依赖项故障，或指定用于存储录制的 S3 存储桶配置错误导致的权限缺失。处于此录制状态时，服务会继续尝试处理。

> **Note**
>
> `ProcessingError` 可能是因 `ssm-guiconnect` 服务主体在建立连接后缺乏将对象上传到 S3 存储桶的权限所致。另一个可能的原因是缺少用于 S3 存储桶加密的 KMS 密钥的 KMS 权限。

修改目标

设置即时节点访问时，可以选择要设置即时节点访问的目标。目标包括 AWS Organizations 组织单元（OU）和 AWS 区域。默认情况下，会选择您在设置统一 Systems Manager 控制台时选择的相同目标进行即时节点访问。您可以选择为所有相同的目标设置即时节点访问权限，或为设置统一的 Systems Manager 控制台时指定的目标子集设置即时节点访问权限。不支持添加在设置统一的 Systems Manager 控制台时未选择的新目标。在设置即时节点访问后，您可以更改所选的目标。

以下过程介绍如何修改目标以实现即时节点访问。

修改目标

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中选择设置。
3. 选择即时节点访问选项卡。
4. 在目标部分中，选择编辑。
5. 选择要使用即时节点访问的组织单位和区域。
6. 选择保存。

更改身份提供者

默认情况下，即时节点访问使用 IAM 作为身份提供者。启用即时节点访问后，在组织中使用统一控制台的客户可以修改此设置以使用 IAM Identity Center。在为单个账户和区域设置时，即时节点访问不支持 IAM Identity Center 作为身份提供者。

以下过程描述了如何修改身份提供者以实现即时节点访问。

修改身份提供者

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中选择设置。
3. 选择即时节点访问选项卡。
4. 在用户身份部分，选择编辑。
5. 选择 AWS IAM Identity Center。
6. 选择保存。

启动即时节点访问会话

启用和设置即时节点访问以及配置会话和通知首选项后，用户便可启动即时节点访问会话。您可以从 Systems Manager 控制台或从使用Session Manager插件的 AWS Command Line Interface，使用即时节点访问来启动会话。可以在同一账户和区域中的节点上启动即时节点访问会话。以下过程介绍如何使用即时节点访问来启动会话。

> **Note**
>
> 如果您的用户以前曾使用Session Manager连接到节点，则您必须从他们的 IAM 策略中移除 Session Manager权限（例如 `ssm:StartSession`），才能使用即时节点访问来启动会话。否则，当连接到节点时，将继续使用Session Manager。

通过控制台使用即时节点访问来启动会话

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中选择浏览节点。
3. 选择要连接到的节点。
4. 在操作下拉列表中，选择连接。

如果组织的批准政策不允许您自动连接到该节点，则系统会提示您提交访问请求。填写所需信息并提交访问请求后，在收到全部所需批准后，您便可开始与该节点的会话。

通过AWS CLI使用即时节点访问来启动会话

1. 运行以下命令以开始访问请求工作流，并确保将####替换为您自己的信息。

```
aws ssm start-access-request \
    --targets  Key=InstanceIds,Values=i-02573cafcfEXAMPLE
    --reason "Troubleshooting networking performance issue"
```

   根据组织的审批政策，要么会自动连接到节点，要么会启动手动审批流程。对于需要手动审批的请求，请记下响应中返回的访问请求的 ID。

2. 等待提供全部所需批准。

3. 提供全部所需批准后，运行以下命令以获取包含临时凭证的访问令牌。将#### 替换为您自己的信息。

```
aws ssm get-access-token \
    --access-request-id oi-12345abcdef
```

   记下响应中返回的访问令牌。

4. 运行以下命令以使用 AWS CLI 中的临时凭证，并确保将####替换为您自己的信息。

```
export AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of session token>
```

5. 运行以下命令以启动与节点的会话，并确保将####替换为您自己的信息。

```
aws ssm start-session \
    --target i-02573cafcfEXAMPLE
```

管理即时访问请求

为了提高整个组织的可见性，Systems Manager 会将访问请求复制到贵组织的委派管理员账户。为帮助您满足合规性要求，Systems Manager 会将所有访问请求保留 1 年。以下主题介绍了如何管理即时节点访问请求。此信息适用于访问请求审批人。在开始之前，我们建议您查看 IAM 策略，并确保您拥有管理即时节点访问所需的权限。有关更多信息，请参阅使用 Systems Manager 设置即时访问。

主题

- 批准和拒绝即时节点访问请求

批准和拒绝即时节点访问请求

访问请求审批人可以批准或拒绝来自统一 Systems Manager 控制台或使用您的首选命令行工具的即时节点访问请求。此信息适用于访问请求审批人。如果您没有批准或拒绝访问请求所需的权限，请联系您的管理员。以下过程介绍如何批准或拒绝即时节点访问请求。

使用控制台批准或拒绝即时节点访问请求

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中选择管理节点访问。
3. 选择访问请求选项卡。
4. 选择我的请求开关。
5. 在要批准或拒绝的访问请求旁边，选中相应的复选框。
6. 选择批准或拒绝。

批准访问请求后，您可以随时通过选择撤销来撤消您的批准。

使用命令行批准或拒绝即时节点访问请求

1. 记下通知中的访问请求 ID。例如，*oi-12345abcdef*。
2. 运行以下命令，返回有关访问请求批准工作流的详细信息，确保将####替换为您自己的信息。

```
aws ssm get-ops-item \
    --ops-item-id oi-12345abcdef
```

   请记下 OperationalData 的 /aws/accessrequest 字段中的 automationExecutionId 值。例如 *9231944f-61c6-40be-8bce-8ee2bEXAMPLE*。

3. 运行以下命令，批准或拒绝访问请求。使用 Approve 信号类型批准请求，使用 Deny 拒绝请求。确保将####替换为您自己的信息。

```
aws ssm send-automation-signal \
    --automation-execution-id 9231944f-61c6-40be-8bce-8ee2bEXAMPLE \
    --signal-type "Approve"
```

从Session Manager移至即时节点访问

启用即时节点访问后，会话管理器不会对Session Manager的现有资源进行任何更改。这样可确保您的现有环境不会中断，并且在您创建和验证审批策略的同时，用户可以继续启动会话。您准备好测试审批策略后，必须修改现有 IAM 策略以完成向即时节点访问的过渡。这包括向身份添加即时节点访问所需的权限，并移除对Session Manager `StartSession` API 操作的权限。我们建议使用 AWS 账户和 AWS 区域中的身份和节点子集来测试审批策略。

有关即时节点访问所需权限的更多信息，请参阅使用 Systems Manager 设置即时访问。

有关修改和身份的 IAM 权限的更多信息，请参阅《IAM 用户指南》中的添加和删除 IAM 身份权限。

下面介绍一种详细的方法，说明如何从Session Manager移至即时节点访问。

从会话管理器移至即时节点访问需要仔细的规划和测试，以确保在不中断操作的情况下平稳过渡。以下各部分将介绍如何完成此过程。

先决条件

在开始之前，请确保您已完成以下任务：

- 设置 Systems Manager 统一控制台。
- 已确认您有权修改账户中的 IAM 策略。
- 已确定当前授予Session Manager权限的所有 IAM 策略和角色。
- 记录了当前的Session Manager配置，包括会话首选项和日志记录设置。

评测

完成以下任务，评测当前的环境并概述所需的批准行为：

1. 清点您的节点：确定用户当前通过Session Manager访问的所有节点。
2. 确定用户访问模式：记录哪些用户或角色在什么情况下需要访问哪些节点。
3. 映射批准工作流：确定谁应批准不同类型节点的访问请求。
4. 审查标记策略：确保节点已正确标记，以支持计划的审批策略。
5. 审核现有 IAM 策略：确定所有包含Session Manager权限的策略。

规划

分阶段策略

从Session Manager移至即时节点访问时，建议使用如下所示的分阶段方法：

1. 第 1 阶段：设置和配置 – 启用即时节点访问，无需修改现有Session Manager权限。
2. 第 2 阶段：策略制定 – 为节点创建和测试审批策略。
3. 第 3 阶段：试点迁移 – 将一小部分非关键节点和用户或角色从Session Manager修改为即时节点访问。
4. 第 4 阶段：完全迁移 – 逐步迁移所有剩余的节点和用户或角色。

时间线注意事项

在创建时间线以从Session Manager移至即时节点访问时，请考虑以下因素：

- 留出时间以便根据新的审批工作流进行用户培训和调整。
- 计划在操作活动较少的时段进行的迁移。
- 将故障排除和调整的缓冲时间包含在内。
- 在两个系统都可用的情况下，计划一段并行操作的时期。

实施步骤

第 1 阶段：设置和配置

1. 在 Systems Manager 控制台中启用即时节点访问。有关详细步骤，请参阅使用 Systems Manager 设置即时访问。
2. 为即时节点访问配置会话首选项，使其与您当前的Session Manager设置相匹配。有关更多信息，请参阅 更新即时节点访问会话首选项。
3. 设置访问请求的通知首选项。有关更多信息，请参阅 为即时访问请求配置通知。
4. 如果您使用与 Windows Server 节点的 RDP 连接，请配置 RDP 录制。有关更多信息，请参阅 录制 RDP 连接。

第 2 阶段：策略制定

1. 为即时节点访问管理员和用户创建 IAM 策略。
2. 根据您的安全要求和应用场景制定审批策略。

3. 请在非生产环境中测试策略，确保其按预期运行。

第 3 阶段：试点迁移

1. 为试点选择一小群用户和非关键节点。
2. 为试点用户创建包含即时节点访问权限的新 IAM 策略。
3. 从试点用户的 IAM 策略中删除Session Manager权限 (`ssm:StartSession`)。
4. 对试点用户进行有关新的访问请求工作流的培训。
5. 监控试点是否存在问题并收集反馈。
6. 根据试点结果调整策略和程序。

试点用户的 IAM 策略修改示例

具有Session Manager权限的原始策略：

JSON

```
{
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": [
                                "ssm:StartSession",
                                "ssm:ResumeSession",
                                "ssm:TerminateSession"
                            ],
                            "Resource": "*"
                        }
                    ]
                }
```

修改后的即时节点访问策略：

JSON

```
{
```

```
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "ssm:StartAccessRequest",
                    "ssm:GetAccessToken",
                    "ssm:ResumeSession",
                    "ssm:TerminateSession"
                ],
                "Resource": "*"
            }
        ]
    }
```

第 4 阶段：完全迁移

制定分批迁移剩余用户和节点的计划。

测试方法

在整个迁移过程中，请进行以下测试：

- 策略验证：验证审批策略是否正确适用于目标节点和用户。
- 访问请求工作流：针对自动审批和手动审批方案，测试从访问请求到会话建立的完整工作流。
- 通知：验证审批者是否通过配置的渠道（电子邮件、Slack、Microsoft Teams）接收通知。
- 日志记录和监控：验证是否正确捕获和存储了会话日志和访问请求。

有关成功迁移的最佳实践

- 尽早并经常沟通：告知用户迁移时间线和即时节点访问的优势。
- 从非关键系统开始：在进入生产环境之前，从开发或测试环境开始迁移。
- 记录所有内容：保留您的审批策略、IAM 策略更改和配置设置的详细记录。
- 监控和调整：持续监控访问请求和批准工作流，根据需要调整策略。
- 建立治理方案：创建流程，以便在环境改变时定期审查和更新审批策略。

使用 Systems Manager 禁止即时访问

以下过程描述了如何禁用即时节点访问。禁用即时节点访问后，除非您已经实施了其他连接方法，否则组织中的用户可能无法连接到您的节点。

禁用即时节点访问

1. 登录贵组织的 Systems Manager 委派管理员账户。
2. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
3. 在导航窗格中选择设置。
4. 在即时节点访问选项卡中，选择禁用。

即时节点访问权限常见问题

如何从 Session Manager 迁移到即时节点访问？

设置统一控制台并启用即时节点访问后，必须修改现有 IAM 策略才能完成向即时节点访问的迁移。这包括为即时节点访问添加所需权限，并移除对 Session Manager `StartSession` API 操作的权限。有关用于即时节点访问的 IAM 策略的更多信息，请参阅使用 Systems Manager 设置即时访问。

是否必须设置统一控制台才能使用即时节点访问？

是的，设置统一控制台是使用即时节点访问的先决条件。但是，在设置统一控制台并启用即时节点访问后，可通过多种方法连接到节点。例如，可以从 Amazon EC2 控制台和 AWS CLI 启动即时节点访问会话。有关设置统一控制台的更多信息，请参阅为组织设置 Systems Manager 统一控制台。

即时节点访问会产生费用吗？

Systems Manager 为即时节点访问提供 30 天免费试用。试用期结束后，即时节点访问会产生费用。有关更多信息，请参阅 AWS Systems Manager 定价。

即时节点访问审批策略的优先顺序是什么？

审批策略按以下顺序评估：

1. 拒绝访问
2. 自动审批

3. 手动

如何评估手动审批策略？

即时节点访问始终优先应用针对节点的更具体策略。手动审批策略按以下顺序评估：

1. 特定标签的目标
2. 所有节点目标

如果没有适用于节点的审批策略会怎样？

要使用即时节点访问连接到节点，必须有适用于该节点的审批策略。如果没有适用于节点的审批策略，则用户将无法请求访问该节点。

多个审批策略可以针对同一标签吗？

每个标签在审批策略中只能被针对一次。

如果由于标签重叠导致多个手动审批策略应用于同一节点会怎样？

当多个手动审批策略应用于一个节点时，会导致冲突，用户将无法请求访问该节点。创建手动审批策略时请记住这一点，某些实例可能因具体用例带有多个标签。

能否使用即时节点访问，跨账户和跨区域请求访问节点并启动会话？

即时节点访问支持对与请求者位于同一账户和区域内的节点请求访问并启动会话。

能否使用即时节点访问，请求访问通过混合激活注册的节点并启动会话？

可以，即时节点访问支持对通过混合激活注册的节点请求访问并启动会话，但节点必须注册到请求者所在的同一账户和区域内。

诊断并修复

使用 Systems Manager 统一控制台，您可以通过一次诊断操作识别整个实例集中的问题。对于组织，您可以使用单个自动化操作尝试修复所有或仅选定目标。对于组织，作为委派账户管理员，您可以选择跨所有账户和区域的目标。如果您使用单个账户，则一次可以选择一个区域中的目标。

Systems Manager 可以诊断并帮助您修复多种类型的部署失败以及偏差配置。Systems Manager 还可以识别您账户或组织中 Systems Manager 无法将其视为托管节点的 Amazon Elastic Compute Cloud

(Amazon EC2) 实例。EC2 实例诊断过程可以识别与虚拟私有云 (VPC)、域名服务 (DNS) 设置或 Amazon Elastic Compute Cloud (Amazon EC2) 安全组中的错误配置相关的问题。

> **Note**
>
> Systems Manager 支持混合和多云环境中的 EC2 实例和其他计算机类型作为托管节点。要成为托管节点，必须在计算机上安装 AWS Systems Manager Agent (SSM Agent)，并且 Systems Manager 必须具有在计算机上执行操作的权限。
>
> 对于 EC2 实例，可以使用 AWS Identity and Access Management (IAM) 角色在账户级别提供此权限，或者使用实例配置文件在实例级别提供此权限。有关更多信息，请参阅 配置 Systems Manager 所需的实例权限。
>
> 对于非 EC2 计算机，此权限是使用 IAM 服务角色提供的。有关更多信息，请参阅 在混合和多云环境中创建 Systems Manager 所需的 IAM 服务角色。

开始前的准备工作

要使用诊断并修复功能来检测非托管的 EC2 实例，您必须首先将组织或账户载入 Systems Manager 统一控制台。在此过程中，您必须选择创建这些操作所需的 IAM 角色和托管式策略的选项。有关更多信息，请参阅 为组织设置 Systems Manager 统一控制台。

使用下面的主题来帮助您识别和修复某些常见类型的部署失败、偏差配置和非托管 EC2 实例。

主题

- 诊断和修复失败部署
- 诊断和修复偏差配置
- 在 Systems Manager 中诊断和修复非托管 Amazon EC2 实例
- 运行手册操作的修复影响类型
- 在 Systems Manager 中查看修正的执行进度和历史记录

诊断和修复失败部署

Systems Manager 可以诊断并帮助您修复下列类型的失败部署：

- 组织成员账户的核心设置
- 委派管理员账户的核心设置

- 您的账户的核心设置

使用以下过程来尝试修复这些类型的问题。

诊断和修复失败部署

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择诊断并修复。
3. 选择部署问题选项卡。
4. 在失败部署部分中，查看失败部署的调查发现列表。
5. 在设置步骤列中，选择一个调查发现的名称以查看有关该问题的其他详细信息。例如：组织成员账户的核心设置。
6. 在该失败部署的详细信息页面中，您可以查看账户列表以及每个账户中有多少个区域遇到部署失败。
7. 选择一个账户 ID 以查看有关该账户中失败的原因的信息。
8. 在失败区域区域，检查为状态原因提供的信息。此信息可能会指出部署失败的原因，从而可能提供需要进行的配置更改的见解。
9. 如果想要在不进行配置更改的情况下重试部署，请选择重新部署。

诊断和修复偏差配置

Systems Manager 可以诊断并帮助您修复下列类型的偏差配置：

- 组织成员账户的核心设置
- 委派管理员账户的核心设置
- 您的账户的核心设置

使用以下过程来尝试修复这些类型的偏差配置。

诊断和修复偏差配置

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择诊断并修复。

3. 选择部署问题选项卡。
4. 在偏差部署部分中，查看失败部署的调查发现列表。

   –或者–

   要运行新的诊断，请选择检测偏差。
5. 在设置步骤列中，选择一个调查发现的名称以查看有关该问题的其他详细信息。例如：组织成员账户的核心设置。
6. 在该失败部署的详细信息页面中，您可以查看账户列表以及每个账户中有多少个区域存在配置偏差。
7. 选择一个账户 ID 以查看有关该账户中配置偏差的原因的信息。
8. 在偏差的资源区域中，资源列会报告存在偏差的资源的名称。漂移类型列会报告资源是否被修改或删除。
9. 要重新部署预期配置，请选择重新部署。

在 Systems Manager 中诊断和修复非托管 Amazon EC2 实例

为帮助使用 Systems Manager 管理 Amazon Elastic Compute Cloud（Amazon EC2）实例，您可以使用 Systems Manager 统一控制台执行以下操作：

1. 运行手动或计划的诊断过程来确定您的账户或组织中的哪些 EC2 实例当前未由 Systems Manager 管理。
2. 确定导致 Systems Manager 无法接管实例管理的网络问题或其他问题。
3. 运行一个自动化执行来自动修复问题，或访问信息来帮助您手动解决问题。

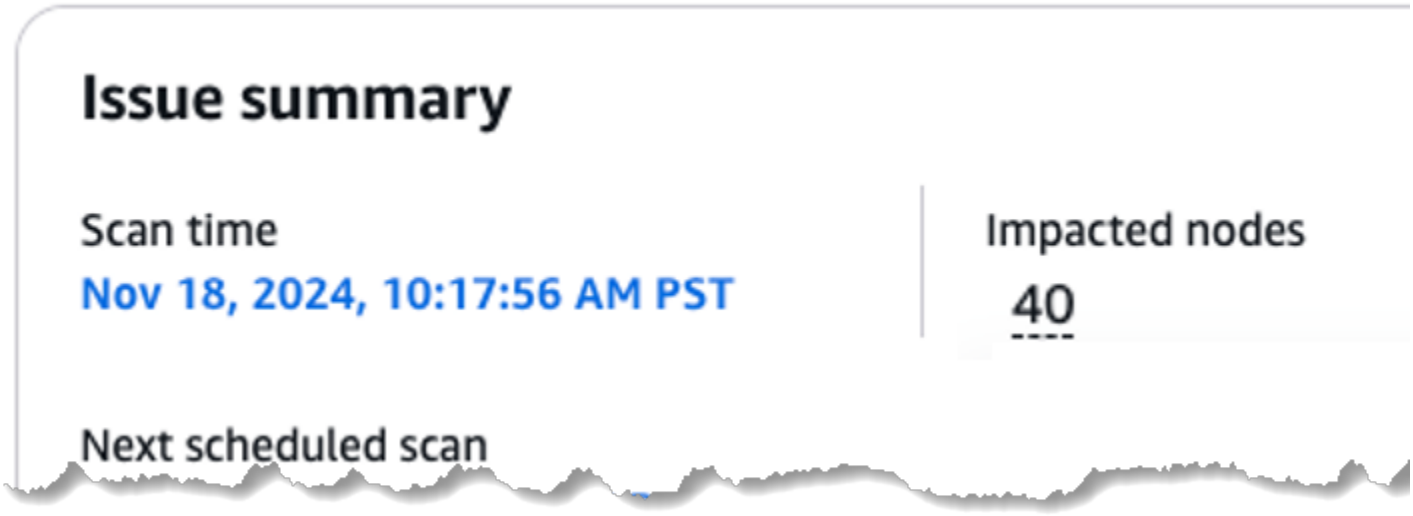
使用以下主题中的信息来帮助您诊断和修复导致 Systems Manager 无法管理您的 EC2 实例的问题。

Systems Manager 如何统计“非托管 EC2 实例问题”列表中受影响的节点

非托管 EC2 实例问题选项卡上报告为非托管的节点数量代表在诊断扫描时具有以下任何状态值的实例总数：

- `Running`
- `Stopped`
- `Stopping`

此数字在问题摘要区域中被报告为受影响的节点。在下图中，当前未由 Systems Manager 管理的受影响的节点数量为 40。



与查看节点见解页面上的非托管 EC2 实例报告不同，此 EC2 实例计数不是动态的。它代表上次报告的诊断扫描期间进行的调查发现，显示为扫描时间值。因此，我们建议定期对非托管 EC2 实例运行诊断扫描，以便此报告的受影响节点数量保持最新状态。

有关查看节点见解页面上非托管式实例计数的信息，请参阅查看节点见解主题中的什么是非托管式实例？。

主题

- 可诊断的非托管 EC2 实例问题类别
- 对非托管 EC2 实例运行诊断和可选修复
- 为非托管 EC2 实例安排定期扫描

可诊断的非托管 EC2 实例问题类别

本主题列出了 EC2 管理问题的主要类别以及每个类别中 Systems Manager 可帮助您诊断并修复的具体问题。请注意，对于某些问题，Systems Manager 可识别问题，但不能提供自动修正措施。在这些情况下，Systems Manager 控制台会将您引导到帮助您手动解决问题的信息。

诊断过程会根据每组 EC2 实例所属的虚拟私有云 (VPC) 一次检查这些实例。

问题类型

- 问题类别：安全组配置和 HTTPS 通信
- 问题类别：DNS 或 DNS 主机名配置

- 问题类别：VPC 端点配置

问题类别：安全组配置和 HTTPS 通信

诊断操作可能发现 SSM Agent 无法通过 HTTPS 与 Systems Manager 服务进行通信。在这些情况下，您可以选择执行一个自动化运行手册来尝试更新附加到实例的安全组。

> **Note**
>
> 有时，Systems Manager 可能无法自动修复这些问题，然而您可以手动编辑受影响的安全组。

支持的问题类型

- 实例安全组：端口 443 不允许出站流量
- **ssm** VPC 端点的安全组：端口 443 不允许入站流量
- **ssmmessages** VPC 端点的安全组：端口 443 不允许入站流量
- **ec2messages** VPC 端点的安全组：端口 443 不允许入站流量

有关更多信息，请参阅 验证端点安全组的入口规则主题中的 排查 SSM Agent 问题。

问题类别：DNS 或 DNS 主机名配置

诊断操作可能发现没有为 VPC 正确配置域名系统 (DNS) 或 DNS 主机名。在这些情况下，您可以选择执行一个自动化运行手册来尝试启用受影响 VPC 的 `enableDnsSupport` 和 `enableDnsHostnames` 属性。

支持的问题类型

- VPC 中已禁用 DNS 支持。
- VPC 中已禁用 DNS 主机名。

有关更多信息，请参阅 验证 VPC DNS 相关属性主题中的 排查 SSM Agent 问题。

问题类别：VPC 端点配置

诊断操作可能发现没有为 VPC 正确配置 VPC 端点。

如果 SSM Agent 所需的 VPC 端点不存在，则 Systems Manager 会尝试执行一个自动化运行手册来创建 VPC 端点，并将它们与每个相关区域性可用区 (AZ) 中的一个子网关联。如果所需的 VPC 端点存在，但没有与发现问题的子网相关联，则运行手册会将 VPC 端点关联到受影响的子网。

> **Note**
>
> Systems Manager 不支持修复所有配置错误的 VPC 端点问题。在这些情况下，Systems Manager 会将您引导到手动修复说明，而不是运行自动化运行手册。

支持的问题类型

- 未找到 PrivateLink 的 `ssm.region.amazonaws.com` 端点。
- 未找到 PrivateLink 的 `ssmmessages.region.amazonaws.com` 端点。
- 未找到 PrivateLink 的 `ec2messages.region.amazonaws.com` 端点。

可诊断的问题类型

Systems Manager 可以诊断下面的问题类型，但目前没有可用于修复其问题的运行手册。您可以手动编辑您的配置以解决这些问题。

- 实例的子网未连接到 `ssm.region.amazonaws.com` 端点。
- 实例的子网未连接到 `ssmmessages.region.amazonaws.com` 端点。
- 实例的子网未连接到 `ec2messages.region.amazonaws.com` 端点。

有关更多信息，请参阅 验证您的 VPC 配置主题中的 排查 SSM Agent 问题。

对非托管 EC2 实例运行诊断和可选修复

使用以下过程来诊断可能导致 Systems Manager 无法管理您的 EC2 实例的网络相关问题和 VPC 相关问题。

该诊断操作可以检测下面类型的问题，并将其分组到一起：

- 网络配置问题：可能导致 EC2 实例无法与云中的 Systems Manager 服务进行通信的网络问题类型。这些问题可能有修复操作。有关网络配置问题的更多信息，请参阅可诊断的非托管 EC2 实例问题类别。

- 未识别的问题：诊断操作无法确定 EC2 实例不能与云中的 Systems Manager 服务进行通信的原因的情况的调查发现列表。

对非托管 EC2 实例运行诊断和修复

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择诊断并修复。
3. 选择非托管 EC2 实例问题选项卡。
4. 在问题摘要部分，选择运行新诊断。

   –或者–

   如果这是您第一次诊断非托管 EC2 问题，请在诊断非托管 EC2 实例部分中选择执行。

   > **Tip**
   >
   > 在诊断运行时，选择查看进度或查看执行以监控执行的当前状态。有关更多信息，请参阅在 Systems Manager 中查看修正的执行进度和历史记录。

5. 在诊断完成后，请执行下面的操作：
   - 对于未识别的问题部分中报告的任何问题，请选择了解更多链接以获取有关解决问题的信息。
   - 对于网络配置问题部分中报告的问题，请继续下一步。
6. 在调查发现类型列表中，在推荐列中，对于特定问题选择链接，例如 2 条推荐。
7. 在打开的推荐窗格中，从可用的缓解措施中进行选择：
   - 了解更多：打开一个主题，其中包含有关如何手动解决问题的信息。
   - 查看运行手册：打开一个窗格，其中包含有关您可以执行以便解决 EC2 实例问题的自动化运行手册的信息，以及用于生成运行手册将执行的操作的预览的选项。继续执行下一步。
8. 在运行手册窗格中，执行下面的操作：
   a. 对于文档描述，请查看内容，其中概述了运行手册可以采取哪些操作来修复非托管 EC2 实例问题。选择查看步骤以预览运行手册将采取的各个操作。
   b. 对于目标，请执行以下操作：

- 如果要管理某组织的修复，对于账户，请指定此运行手册是针对所有账户，还是仅针对您选择的部分账户。
- 对于区域，请指定此运行手册是针对您的账户或组织中的所有 AWS 区域，还是仅针对您选择的部分区域。

c. 对于运行手册预览，请仔细查看信息。此信息说明了当您选择执行运行手册时的范围和影响。

> **Note**
>
> 选择执行运行手册将会产生费用。请仔细查看预览信息，然后再决定是否继续。

运行手册预览内容提供以下信息：

- 运行手册操作将在多少个区域中发生。
- （仅限 Organizations）操作将在多少个组织单位 (OU) 中运行。
- 将要采取的操作类型以及每种操作的数量。

  操作类型包括：

  - 变异：运行手册步骤将通过创建、修改或删除资源的操作对目标进行更改。
  - 非变异：运行手册步骤将检索有关资源的数据，但是不对其进行更改。此类别通常包括 `Describe*`、`List*`、`Get*` 和类似的只读 API 操作。
  - 未确定：未确定的步骤将调用由另一个编排服务（如 AWS Lambda、AWS Step Functions 或 AWS Systems Manager Run Command）运行的执行。未确定的步骤也可能会调用第三方 API。Systems Manager Automation 不知道编排流程或第三方 API 执行的结果，因此步骤的结果尚未确定。

d. 此时，您可以选择以下操作之一：

- 停止，不要执行运行手册。
- 选择执行以使用您已经选择的选项运行运行手册。

如果选择运行该操作，请选择查看进度或查看执行以监控执行的当前状态。有关更多信息，请参阅[在 Systems Manager 中查看修正的执行进度和历史记录]。

为非托管 EC2 实例安排定期扫描

您可以对您的账户或组织中由于各种配置问题而 Systems Manager 无法管理的 Amazon EC2 实例运行按需扫描。您还可以安排此扫描定期自动进行。

为非托管 EC2 实例安排定期扫描

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择诊断并修复。
3. 选择非托管 EC2 实例问题选项卡。
4. 在诊断非托管 EC2 实例部分中，启用安排定期诊断。
5. 对于诊断频率，选择每天运行一次还是每周运行一次诊断。
6. （可选）对于开始时间，以 24 小时格式输入诊断开始的时间。例如，对于晚上 8:15，请输入 **`20:15`**。

   您输入的时间是您当前当地时区的时间。

   如果未指定时间，则诊断扫描将立即运行。Systems Manager 还会安排扫描在未来的当前时间运行。如果指定了时间，则 Systems Manager 会等待在指定的时间运行诊断扫描。
7. 选择执行。诊断会立即运行，但也会按照您指定的计划运行。

运行手册操作的修复影响类型

Systems Manager 可以运行诊断操作来发现某些类型的失败部署和偏差配置，以及导致 Systems Manager 无法管理 EC2 实例的某些类型的配置问题。诊断结果可能包括您可以执行以尝试解决问题的自动化运行手册的建议。有关这些诊断操作的更多信息，请参阅以下主题：

- 诊断和修复失败部署
- 诊断和修复偏差配置
- 在 Systems Manager 中诊断和修复非托管 Amazon EC2 实例

当 Systems Manager 识别可能通过对受影响的资源运行自动化运行手册来修复的问题时，它会为您提供执行预览。该执行预览提供了有关运行手册执行将对您的目标做出的更改类型的信息。此信息包括诊断识别的三种更改类型中每种更改的数量。

这些更改类型如下：

- `Mutating`：运行手册步骤将通过创建、修改或删除资源的操作对目标进行更改。
- `Non-Mutating`：运行手册步骤将检索有关资源的数据，但是不对其进行更改。此类别通常包括 `Describe*`、`List*`、`Get*` 和类似的只读 API 操作。
- `Undetermined`：未确定的步骤将调用由另一个编排服务运行的执行，如 AWS Lambda、AWS Step Functions 或 Run Command（AWS Systems Manager 中的一项工具）。未确定的步骤可能还会调用第三方 API 或者运行 Python 或 PowerShell 脚本。Systems Manager Automation 无法检测到编排流程或第三方 API 执行的结果，所以无法对其进行评估。必须手动审查这些步骤的结果以确定其影响。

  有关受支持自动化操作的影响类型的信息，请参阅下表。

支持的修复操作的影响类型

该表列出了可包含在修复运行手册中的各种操作的影响类型（变异、非变异和未确定）。

操作[1]	影响类型
aws:approve	非变异
aws:assertAwsResourceProperty	非变异
aws:branch	非变异
aws:changeInstanceState	变异
aws:copyImage	变异
aws:createImage	变异
aws:createStack	变异
aws:createTags	变异
aws:deleteImage	变异
aws:deleteStack	变异
aws:executeAutomation	未确定
aws:executeAwsApi	未确定

操作[1]	影响类型
aws:executeScript	未确定
aws:executeStateMachine	未确定
aws:invokeLambdaFunction	未确定
aws:invokeWebhook	未确定
aws:loop	可变。取决于循环中的操作。
aws:pause	非变异
aws:runCommand	未确定
aws:runInstances	变异
aws:sleep	非变异
aws:updateVariable	变异
aws:waitForAwsResourceProperty	非变异

[1] 有关自动化操作的更多信息，请参阅 Systems Manager 自动化操作参考。

在 Systems Manager 中查看修正的执行进度和历史记录

您可以查看使用 Systems Manager 中的“诊断并修复”功能进行的所有正在进行和已完成的修复操作的列表。

执行历史记录列表中的数据会报告以下类型的信息：

- 执行的类型：`Diagnosis` 或 `Remediation`。
- 执行状态，例如 `Success` 或 `Failed`。
- 执行开始和结束的时间。

查看修正的执行进度和历史记录

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择诊断并修复。
3. 选择查看执行。

> **Tip**
>
> 当执行正在运行时，您也可以选择查看进度以打开执行历史记录页面。

4. （可选）在搜索
   ()
   框中，输入一个短语以帮助缩小执行列表的范围，例如 **EC2** 或 **VPC**。
5. （可选）要查看有关执行的更多详细信息，请在执行名称列中选择一个操作名称（例如 AWS-DiagnoseUnmanagedEC2NetworkIssues）。

   在详细信息窗格中，您可以查看有关操作期间尝试的所有步骤以及执行的所有输入和输出的信息。

调整 Systems Manager 设置

设置页面上的选项可启用和配置 Systems Manager 统一控制台中的功能。显示的选项取决于您登录的账户以及您是否已设置 Systems Manager。

> **Note**
>
> 设置页面上的选项不会影响 Systems Manager 工具（以前称为功能）。

账户设置设定

如果启用了 Systems Manager，您登录的账户不是 Organizations 的成员，或者委派管理员尚未将您的 Organizations 账户添加到 Systems Manager，则账户设置页面会显示禁用 Systems Manager 的选项。禁用 Systems Manager 意味着 Systems Manager 不会显示统一控制台。所有 Systems Manager 工具仍可运行。

组织设置设定

在组织设置选项卡上，主区域部分显示在设置期间选择为主区域的 AWS 区域。在使用 AWS Organizations 的多账户和多区域环境中，Systems Manager 会自动将所有账户和区域的节点数据聚合到主区域。通过这种方式聚合数据，您可以在单个位置跨账户和区域查看节点数据。

> **Note**
>
> 如果要更改主区域，则必须禁用 Systems Manager 然后重新启用它。要禁用 Systems Manager，请选择禁用。

组织设置部分显示在设置期间选择的 AWS 组织单位和 AWS 区域。要更改哪些组织单位和区域在 Systems Manager 中显示节点数据，请选择编辑。有关设置 Systems Manager for Organizations 的更多信息，请参阅设置 AWS Systems Manager。

功能配置

功能配置部分允许启用和配置可增强整个组织的节点管理的关键 Systems Manager 功能。这些功能协同工作，为托管节点提供自动管理、合规性监控和维护。

您可以在初始 Systems Manager 设置期间配置这些功能，也可以稍后通过“设置”页面修改它们。每个功能都可以根据组织的要求独立启用或禁用。

默认主机管理配置

默认主机管理配置 (DHMC) 会自动将您组织中的 Amazon Elastic Compute Cloud (Amazon EC2) 实例配置为由 Systems Manager 管理。启用后，DHMC 可确保新的和现有 EC2 实例具有与 Systems Manager 服务通信所需的 AWS Identity and Access Management (IAM) 权限和配置。

DHMC 提供以下好处：

- 自动 IAM 角色分配 - 确保 EC2 实例具有所需的 IAM 角色和策略以作为托管节点运行
- 偏差修复 - 当实例丢失其托管节点状态时，自动纠正配置偏差
- 简化启用过程 - 减少新实例的手动配置步骤
- 一致配置 - 在 EC2 实例集中保持统一的设置

配置偏差修复频率

当 EC2 实例丢失其托管节点配置时，偏差修复会自动检测并纠正。您可以配置 Systems Manager 检查和修复配置偏差的频率。

配置默认主机管理配置

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Settings（设置）。
3. 在功能配置部分，找到默认主机管理配置。
4. 要启用 DHMC，请启用切换开关。
5. 对于偏差修复频率，请选择希望 Systems Manager 检查和修复配置偏差的频率：
   - 每天 - 每天检查和修复偏差一次
   - 每周 - 每周检查和修复偏差一次
   - 每月 - 每月检查和修复偏差一次
6. 选择保存。

> **Note**
>
> 启用 DHMC 后，Systems Manager 会在您的账户中创建必要的 IAM 角色和策略。这些角色允许 EC2 实例与 Systems Manager 服务通信。有关 DHMC 创建的 IAM 角色的更多信息，请参阅使用 Systems Manager 管理 EC2 实例。

清单元数据收集

清单元数据收集会自动收集有关托管节点的详细信息，包括已安装的应用程序、网络配置、系统更新以及其他系统元数据。此信息可帮助您保持合规性、执行安全分析并了解基础设施组成。

清单收集具有以下好处：

- 合规性监控 - 跟踪已安装的软件和配置以进行合规性报告
- 安全分析 - 识别过时的软件和潜在安全漏洞
- 资产管理 - 维护基础设施的最新清单

- 查询功能 - 通过 Amazon Q 开发者版使用收集的数据进行自然语言查询

收集的清单数据的类型

启用清单元数据收集后，Systems Manager 会从托管节点收集以下类型的信息：

- 应用程序 - 已安装的软件包和应用程序
- 网络配置 - 网络接口、IP 地址和网络设置
- 系统更新 - 已安装的补丁和可用更新
- 系统属性 - 硬件规格、操作系统详细信息和系统配置
- 服务 - 正在运行的服务及其配置

配置清单收集频率

您可以配置 Systems Manager 从托管节点收集清单元数据的频率。更频繁的收集可提供更多最新信息，但可能会增加 AWS 服务的使用量。

配置清单元数据收集

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Settings（设置）。
3. 在功能配置部分，找到清单元数据收集。
4. 要启用清单收集，请启用切换开关。
5. 对于收集频率，选择希望 Systems Manager 收集清单数据的频率：
   - 每天 - 每天收集清单数据一次
   - 每周 - 每周收集清单数据一次
   - 每月 - 每月收集清单数据一次
6. 选择保存。

> **Important**
>
> 清单收集要求托管节点具有收集系统信息所需的权限。确保托管节点具有适当的 IAM 角色和策略。有关所需权限的更多信息，请参阅AWS Systems Manager 清单。

SSM Agent 更新

自动 SSM Agent 更新可确保托管节点运行最新版本的 SSM Agent。保持代理为最新版本可让您访问最新功能、安全改进和错误修复。

SSM Agent 自动更新具有以下好处：

- 最新功能 - 访问新的 Systems Manager 功能和改进
- 安全更新 - 自动安装安全补丁和修复程序
- 提高可靠性 - 错误修复和稳定性改进
- 减少维护 - 无需手动更新代理

配置自动代理更新

您可以配置 Systems Manager 在托管节点上检查和安装 SSM Agent 更新的频率。定期更新有助于确保最佳性能和安全性。

配置 SSM Agent 更新

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Settings（设置）。
3. 在功能配置部分，找到 SSM Agent 更新。
4. 要启用自动更新，请启用切换开关。
5. 对于更新频率，选择希望 Systems Manager 检查和安装代理更新的频率：
   - 每天 - 每天检查更新一次
   - 每周 - 每周检查更新一次
   - 每月 - 每月检查更新一次
6. 选择保存。

诊断并修复设置

诊断并修复设置确定 Systems Manager 是否自动扫描您的节点，以确保它们可以与 Systems Manager 进行通信。如果启用，则该功能将根据您定义的计划自动运行。该功能可识别哪些节点无法连接至

Systems Manager 以及原因。此功能还提供推荐的运行手册，用于修复网络问题以及其他导致无法将节点配置为托管节点的问题。

安排定期诊断扫描

Systems Manager 可以诊断并帮助您修复多种类型的部署失败以及偏差配置。Systems Manager 还可以识别您账户或组织中 Systems Manager 无法将其视为托管节点的 Amazon Elastic Compute Cloud (Amazon EC2) 实例。EC2 实例诊断过程可以识别与虚拟私有云 (VPC)、域名服务 (DNS) 设置或 Amazon Elastic Compute Cloud (Amazon EC2) 安全组中的错误配置相关的问题。

为了简化识别无法连接到 Systems Manager 的节点的任务，安排定期诊断功能使您能够自动执行定期诊断扫描。扫描有助于识别哪些节点无法连接到 Systems Manager 以及原因。使用以下过程启用并配置您的节点的定期诊断扫描。

安排定期诊断扫描

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择设置，然后选择诊断并修复选项卡。
3. 启用安排定期诊断选项。
4. 对于扫描周期，选择您希望扫描运行的频率。
5. （可选）对于开始时间，以 24 小时格式输入诊断开始的时间。例如，对于晚上 8:15，请输入 **`20:15`**。

   您输入的时间是您当前当地时区的时间。

   如果未指定时间，则诊断扫描将立即运行。Systems Manager 还会安排扫描在未来的当前时间运行。如果指定了时间，则 Systems Manager 会等待在指定的时间运行诊断扫描。
6. 选择保存。
7. 扫描完成后，通过选择左侧导航栏中的诊断并修复来查看详细信息。

有关诊断并修复功能的更多信息，请参阅诊断并修复。

更新 S3 存储桶加密

当您载入 Systems Manager 时，快速设置功能会在委派管理员账户中为 AWS Organizations 设置创建一个 Amazon Simple Storage Service (Amazon S3) 存储桶。对于单账户设置，存储桶会存储在正在设置的账户中。此存储桶用于存储诊断扫描期间生成的元数据。

有关设置 Systems Manager 统一控制台的更多信息，请参阅设置 AWS Systems Manager。

默认情况下，存储桶中您的数据使用 AWS 拥有并且为您管理的 AWS Key Management Service (AWS KMS) 密钥进行加密。

您可以选择使用其他 AWS KMS 密钥进行存储桶加密。另一种方法是，您可以使用具有 AWS KMS keys的服务器端加密密钥 (SSE-KMS) 以及客户自主管理型密钥 (CMK)。有关信息，请参阅使用 Amazon S3 存储桶和 Systems Manager 的存储桶策略。

使用其他 AWS KMS 密钥进行 S3 存储桶加密

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择设置，然后选择诊断并修复选项卡。
3. 在更新 S3 存储桶加密区域中，选择编辑。
4. 选中自定义加密设置（高级）复选框。
5. 在选择 AWS KMS 密钥中，选择或输入密钥的 Amazon 资源名称 (ARN)。

> **Tip**
>
> 要创建新密钥，请选择创建 AWS KMS 密钥。

6. 选择保存。

使用 Amazon S3 存储桶和 Systems Manager 的存储桶策略

在 AWS Systems Manager 的载入过程中，Quick Setup会在委派管理员账户中为组织设置创建一个 Amazon Simple Storage Service (Amazon S3) 存储桶。对于单账户设置，存储桶会存储在正在设置的账户中。

您可以使用 Systems Manager 对您的实例集运行诊断操作，以识别部署失败和偏差配置情况。Systems Manager 还可以检测配置问题导致 Systems Manager 无法管理您账户或组织中的 EC2 实例的情况。这些诊断操作的结果存储在此 Amazon S3 存储桶中，受加密方法和 S3 存储桶策略的保护。有关将数据输出到此存储桶的诊断操作的信息，请参阅诊断并修复。

更改存储桶加密方法

默认情况下，S3 存储桶使用具有 Amazon S3 托管密钥的服务器端加密 (SSE-S3)。

您可以改用具有 AWS KMS keys的服务器端加密 (SSE-KMS)，并使用客户自主管理型密钥 (CMK) 代替 Amazon S3 托管密钥，如 更改为 AWS KMS 客户自主管理型密钥以加密 S3 资源 中所述。

存储桶策略的内容

存储桶策略可防止组织内的成员账户相互发现。仅为 Systems Manager 创建的诊断和修复角色允许对存储桶具有读写权限。Systems Manager 统一控制台的 S3 存储桶策略中介绍了这些系统生成的策略的内容。

> **Warning**
>
> 修改默认存储桶策略可能允许组织内的成员账户相互发现，或者读取另一个账户中实例的诊断输出。如果您选择修改此策略，建议您小心谨慎。

主题

- 更改为 AWS KMS 客户自主管理型密钥以加密 S3 资源
- Systems Manager 统一控制台的 S3 存储桶策略

更改为 AWS KMS 客户自主管理型密钥以加密 S3 资源

在 Systems Manager 统一控制台的载入过程中，Quick Setup会在委派管理员账户中创建一个 Amazon Simple Storage Service（Amazon S3）存储桶。此存储桶用于存储在修复运行手册执行期间生成的诊断输出数据。默认情况下，存储桶使用具有 Amazon S3 托管密钥的服务器端加密 (SSE-S3)。

您可以在 Systems Manager 统一控制台的 S3 存储桶策略 中查看这些策略的内容。

但是，您可以改用具有 AWS KMS keys的服务器端加密 (SSE-KMS)，并使用客户自主管理型密钥 (CMK) 代替 AWS KMS key。

请完成以下任务，以将 Systems Manager 配置为使用您的 CMK。

任务 1：向现有 CMK 添加标签

仅当使用以下键值对进行标记时，AWS Systems Manager 才会使用您的 CMK：

- 键：`SystemsManagerManaged`
- 值：`true`

使用以下过程来提供使用您的 CMK 加密 S3 存储桶的访问权限。

向现有 CMK 添加标签

1. 从 https://console.aws.amazon.com/kms 打开 AWS KMS 控制台。
2. 在左侧导航中，选择客户管理的密钥。
3. 选择要与 AWS Systems Manager 一起使用的 AWS KMS key。
4. 选择标签选项卡，然后选择编辑。
5. 选择 Add tag（添加标签）。
6. 执行以下操作：
   a. 在 Tag key（标签键）中输入 **SystemsManagerManaged**。
   b. 对于标签值，输入 **true**。
7. 选择保存。

任务 2：修改现有 CMK 密钥政策

使用以下过程来更新您的 CMK 的 KMS 密钥政策，以允许 AWS Systems Manager 角色代表您加密 S3 存储桶。

修改现有 CMK 密钥政策

1. 从 https://console.aws.amazon.com/kms 打开 AWS KMS 控制台。
2. 在左侧导航中，选择客户管理的密钥。
3. 选择要与 AWS Systems Manager 一起使用的 AWS KMS key。
4. 在密钥策略选项卡上，选择编辑。
5. 在 `Statement` 字段中添加下面的 JSON 语句，然后将`####`替换为您自己的信息。

   确保在 `Principal` 字段中添加已载入您组织中的 AWS Systems Manager 的所有 AWS 账户 ID。

   要在 Amazon S3 控制台中找到正确的存储桶名称，请在委派管理员账户中找到格式为 `do-not-delete-ssm-`*`operational-account-id`*`-`*`home-region`*`-`*`disambiguator`* 的存储桶。

```
{
     "Sid": "EncryptionForSystemsManagerS3Bucket",
     "Effect": "Allow",
     "Principal": {
         "AWS": [
```

```
                "account-id-1",
                "account-id-2",
                ...
            ]
        },
        "Action": ["kms:Decrypt", "kms:GenerateDataKey"],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::amzn-s3-demo-bucket"
            },
            "StringLike": {
                "kms:ViaService": "s3.*.amazonaws.com"
            },
            "ArnLike": {
                "aws:PrincipalArn": "arn:aws:iam::*:role/AWS-SSM-*"
            }
        }
    }
```

> **Tip**
>
> 或者，您可以使用 aws:PrincipalOrgID 条件键更新 CMK 密钥政策，以授予 AWS Systems Manager 对您的 CMK 的访问权限。

任务 3：在 Systems Manager 设置中指定 CMK

完成前两个任务后，请使用以下过程来更改 S3 存储桶加密。此更改可确保关联的Quick Setup配置过程可以为 Systems Manager 添加接受 CMK 的权限。

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Settings（设置）。
3. 在诊断并修复选项卡上的更新 S3 存储桶加密部分中，选择编辑。
4. 选中自定义加密设置（高级）复选框。
5. 在搜索 (🔍) 框中，选择现有密钥的 ID 或粘贴现有密钥的 ARN。

6. 选择保存。

Systems Manager 统一控制台的 S3 存储桶策略

本主题包括当您将某组织或单个账户加入 Systems Manager 统一控制台时 Systems Manager 创建的 Amazon S3 存储桶策略。

> **Warning**
>
> 修改默认存储桶策略可能允许组织内的成员账户相互发现，或者读取另一个账户中实例的诊断输出。如果您选择修改此策略，建议您小心谨慎。

组织的 Amazon S3 存储桶策略

当将某组织加入 Systems Manager 时，将使用以下默认存储桶策略创建诊断存储桶。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyHTTPRequests",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ],
            "Condition": {
                "Bool": {
                    "aws:SecureTransport": "false"
                }
            }
        },
        {
            "Sid": "DenyNonSigV4Requests",
            "Effect": "Deny",
            "Principal": "*",
```

```
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "s3:SignatureVersion": "AWS4-HMAC-SHA256"
                }
            }
        },
        {
            "Sid": "AllowAccessLog",
            "Effect": "Allow",
            "Principal": {
                "Service": "logging.s3.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/access-logs/*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "operational-account-id"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:s3:::amzn-s3-demo-bucket"
                }
            }
        },
        {
            "Sid": "AllowCrossAccountRead",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/actions/*/
${aws:PrincipalAccount}/*",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalOrgID": "organization-id"
                }
            }
        },
        {
            "Sid": "AllowCrossAccountWrite",
            "Effect": "Allow",
```

```
            "Principal": "*",
            "Action": [
                "s3:PutObject",
                "s3:DeleteObject"
            ],
            "Resource": "arn:aws:s3:::bucket-name/actions/*/
${aws:PrincipalAccount}/*",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalOrgID": "organization-id"
                },
                "ArnLike": {
                    "aws:PrincipalArn": [
                        "arn:aws:iam::*:role/AWS-SSM-
DiagnosisExecutionRole-operational-account-id-home-region",
                        "arn:aws:iam::*:role/AWS-SSM-
DiagnosisAdminRole-operational-account-id-home-region",
                        "arn:aws:iam::*:role/AWS-SSM-
RemediationExecutionRole-operational-account-id-home-region",
                        "arn:aws:iam::*:role/AWS-SSM-
RemediationAdminRole-operational-account-id-home-region"
                    ]
                }
            }
        },
        {
            "Sid": "AllowCrossAccountListUnderAccountOwnPrefix",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalOrgID": "organization-id"
                },
                "StringLike": {
                    "s3:prefix": "*/${aws:PrincipalAccount}/*"
                }
            }
        },
        {
            "Sid": "AllowCrossAccountGetConfigWithinOrganization",
            "Effect": "Allow",
            "Principal": "*",
```

```
            "Action": "s3:GetEncryptionConfiguration",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalOrgID": "organization-id"
                }
            }
        }
    ]
}
```

单个账户的 Amazon S3 存储桶策略

当将单个账户加入 Systems Manager 时，将使用以下默认存储桶策略创建诊断存储桶。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyHTTPRequests",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket",
        "arn:aws:s3:::amzn-s3-demo-bucket/*"
      ],
      "Condition": {
        "Bool": {
          "aws:SecureTransport": "false"
        }
      }
    },
    {
      "Sid": "DenyNonSigV4Requests",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket",
```

```
        "arn:aws:s3:::amzn-s3-demo-bucket/*"
      ],
      "Condition": {
        "StringNotEquals": {
          "s3:SignatureVersion": "AWS4-HMAC-SHA256"
        }
      }
    }
  ]
}
```

使用 AWS Systems Manager 工具

Systems Manager 将工具分为以下四类。以下文档介绍了 AWS Systems Manager 的各种工具以及如何设置和使用这些工具。请选择每个类别下的选项卡以了解有关每个工具的更多信息。

节点工具

托管节点是指在混合和多云环境中配置为与 Systems Manager 一起使用的任何计算机。

Compliance

使用 Compliance（合规性）可扫描您的托管式节点机群，以了解补丁合规性和配置不一致性。您可以从多个 AWS 账户 和 AWS 区域 中收集并聚合数据，然后深入了解不合规的特定资源。默认情况下，合规性将显示有关 Patch Manager 修补和 State Manager 关联的合规性数据。您也可以根据 IT 或业务要求自定义服务并创建自己的合规性类型。

分销商

使用 Distributor 可以创建软件包并将它们部署到托管式节点。利用 Distributor，您可以将自己的软件打包或查找 AWS 提供的代理软件包（如 AmazonCloudWatchAgent），以便在 Systems Manager 托管式节点上安装。在首次安装软件包后，您可以使用 Distributor 卸载并重新安装新的软件包版本，或者执行就地更新以添加新文件或更改后的文件。Distributor 将资源（例如软件包）发布到 Systems Manager 托管式节点。

Fleet Manager

Fleet Manager 是一种统一的用户界面 (UI) 体验，可帮助您远程管理节点。利用 Fleet Manager，您可以从一个控制台查看整个机群的运行状况和性能状态。您还可以从单个设备和实例收集数据，以便从该控制台执行常见的故障排除和管理任务。这包括查看目录和文件内容、Windows 注册表管理、操作系统用户管理等。

Hybrid Activations

要将混合和多云环境中的非 EC2 计算机设置为托管式节点，请创建混合激活。完成激活后，您将收到一个激活代码和 ID。此代码和 ID 组合功能 [如 Amazon Elastic Compute Cloud (Amazon EC2) 访问 ID 和私有密钥]，用于提供从托管式实例对 Systems Manager 服务进行安全访问。

如果要使用 Systems Manager 管理边缘设备，您也可以为边缘设备创建激活。

Inventory

Inventory 可自动执行从托管实例中收集软件清单的流程。您可以使用 Inventory 收集有关应用程序、文件、组件、补丁等对象的元数据。

Patch Manager

使用 Patch Manager 可以通过安全性相关更新及其他类型的更新自动执行修补托管式节点的过程。您可以使用 Patch Manager 来应用操作系统和应用程序的补丁。（在 Windows Server 上，应用程序支持仅限于更新 Microsoft 发布的应用程序。）

利用此工具，您可以扫描托管式节点是否有缺失的补丁，然后单独应用缺失的补丁或使用标签将这些补丁应用到大型托管式节点组。Patch Manager 使用补丁基准，其中可以包含用于在补丁发布几天内自动批准补丁的规则，以及一系列已批准和已拒绝的补丁。您可以通过将修补安排为作为 Systems Manager 维护时段任务来运行，定期安装安全性补丁，也可以随时按需对托管式节点进行修补。

对于 Linux 操作系统，您可以定义存储库应该作为补丁基准的一部分用于修补操作。这样，无论托管式节点上配置的是什么存储库，您都可以确保仅安装信任的存储库中的更新。对于 Linux，您还能够更新托管式节点上的任何包，而不仅仅是被归类为操作系统安全更新的包。您还可以生成发送到您选择的 S3 存储桶的补丁报告。对于单个托管式节点，报告中包括计算机所有补丁的详细信息。对于所有托管式节点的报告，仅提供缺少多少补丁的摘要。

Run Command

使用 Run Command 可以远程方式安全、大规模地管理托管式节点的配置。使用 Run Command 在几十个或数百个托管式节点的目标集上执行按需更改，例如更新应用程序或运行 Linux Shell 脚本和 Windows PowerShell 命令。

会话管理器

使用 Session Manager 可通过基于浏览器的一键式交互 Shell 或 AWS CLI 来管理 Amazon Elastic Compute Cloud (Amazon EC2) 实例。Session Manager 提供安全且可审计的边缘设备和实例管理，无需打开入站端口、维护堡垒主机或管理 SSH 密钥。Session Manager 还可以确保您遵守要求边缘设备和实例受控访问权限的公司策略、严格的安全实践以及包含边缘设备和实例访问详细信息的完全可审计日志，同时能够让终端用户轻松地一键式跨平台访问您的边缘设备和 EC2 实例。要使用 Session Manager，必须启用高级实例套餐。有关更多信息，请参阅 打开高级实例套餐。

State Manager

使用 State Manager 可自动化执行使托管式节点保持为预先设定状态的过程。您可以使用 State Manager 确保您的托管式节点在启动时使用特定软件进行引导启动，加入某个 Windows 域（仅限 Windows Server 节点）或使用特定软件更新进行修补。

变更管理工具

Automation

使用 Automation（自动化）可自动执行常见的维护和部署任务。您可以使用自动化创建和更新 Amazon Machine Images (AMIs)、应用驱动程序和代理的更新、在 Windows Server 实例上重置密码、在 Linux 实例上重置 SSH 密钥，并应用 OS 补丁或应用程序更新。

更改日历

Change Calendar 可帮助您在指定操作时设置日期和时间范围（例如，Systems Manager 自动化运行手册）能或不能在 AWS 账户 中执行。在 Change Calendar 中，这些范围称为事件。在您创建 Change Calendar 条目时，您将创建类型 `ChangeCalendar` 的 Systems Manager 文档 在 Change Calendar 中，文档以纯文本格式存储 iCalendar 2.0 数据。您添加到 Change Calendar 条目的事件将成为该文档的一部分。您可以在 Change Calendar 界面中手动添加事件，或者使用 `.ics` 文件从支持的第三方日历导入事件。

变更管理器

Change Manager 是一个企业变更管理框架，用于请求、批准、实施和报告应用程序配置和基础设施的操作变更。如果您使用 AWS Organizations，可从单个委托管理员账户中，管理多个 AWS 区域 中多个 AWS 账户 的变更。或者，通过使用本地账户，您可以管理单个 AWS 账户 的变更。使用 Change Manager 可管理对 AWS 资源和本地部署资源的变更。

Documents

Systems Manager 文档（SSM 文档）定义 Systems Manager 执行的操作。SSM 文档类型包括由 State Manager 和 Run Command 使用的命令文档和由 Systems Manager 自动化使用的自动化运行手册。Systems Manager 包括几十个预先配置的文档，您可以通过在运行时指定参数进行使用。文档可以采用 JSON 或 YAML 表示，并包括您指定的步骤和参数。

维护时段

使用 Maintenance Windows 可以设置托管实例的周期性计划，以便运行诸如安装补丁和更新等管理任务，而不会中断业务关键性操作。

快速设置

使用 Quick Setup 以配置常用的 AWS 服务和功能，并提供建议的最佳实践。您可以通过与 AWS Organizations 集成在单个 AWS 账户 或在多个 AWS 账户 和 AWS 区域 之间使用 Quick Setup。Quick Setup通过自动执行常见或推荐的任务，简化了服务的设置（包括 Systems Manager）。例如，这些任务包括创建必需的 AWS Identity and Access Management (IAM) 实例配置文件角色和设操作运营最佳实践，例如定期补丁扫描和清单收集。

应用程序工具

AppConfig

AppConfig 帮助您创建、管理以及部署应用程序配置和功能标记。AppConfig 支持以受控方式部署到任意大小的应用程序。您可以将 AppConfig 与 Amazon EC2 实例上托管的应用程序、AWS Lambda 容器、移动应用程序或边缘设备一起使用。为了防止在部署应用程序配置时出现意外错误，AppConfig 包含验证程序。验证程序提供语法或语义检查，以确保要部署的配置正常工作。在配置部署期间，AppConfig 监控应用程序以确保部署成功。如果系统遇到错误或部署触发警报，AppConfig 将回滚更改以最大限度减少对应用程序用户的影响。

Application Manager

Application Manager 帮助 DevOps 工程师在他们的应用程序和集群环境中调查和解决他们的 AWS 资源问题。在 Application Manager 中，应用程序是一个您希望其作为一个单元运行的 AWS 资源的逻辑组。此逻辑组可以表示应用程序的不同版本、操作员的所有权边界或开发人员环境等。对容器集群的 Application Manager 支持包括 Amazon Elastic Kubernetes Service（Amazon EKS）和 Amazon Elastic Container Service（Amazon ECS）集群。Application Manager 将多个 AWS 服务和 Systems Manager 工具的操作信息聚合到单个 AWS Management Console中。

Parameter Store

Parameter Store 提供安全的分层存储，用于配置数据管理和密钥管理。可以将密码、数据库字符串、 Amazon Elastic Compute Cloud (Amazon EC2) 实例 ID、Amazon Machine Image (AMI) ID 和许可证代码等数据存储为参数值。可以将值存储为纯文本或加密数据。然后，可以使用创建参数时指定的唯一名称来引用对应值。

操作工具

CloudWatch Dashboards

Amazon CloudWatch 控制面板是 CloudWatch 控制台中的可自定义页面，可用于在单个视图中监控资源，即便资源分布在不同区域，也能对其进行监控。您可以使用 CloudWatch 控制面板创建 AWS 资源的指标和警报的自定义视图。

Explorer

Explorer 是一个可自定义的操作控制面板，用于报告有关 AWS 资源的信息。Explorer 可以显示您的 AWS 账户 和不同 AWS 区域 中的操作数据 (OpsData) 的聚合视图。在 Explorer 中，OpsData 包含有关 Amazon EC2 实例、补丁合规性详细信息和操作工作项 (OpsItems) 的元数据。Explorer

提供有关如何在业务单位或应用程序之间分配 OpsItems、它们随时间的变化趋势以及它们如何随类别变化的上下文。您可以在 Explorer 中对信息进行分组和筛选，以将重点放在与您相关的项目和需要采取措施的项目上。在查找高优先级问题时，您可以使用 OpsCenter（Systems Manager 中的一项工具）运行 Automation 运行手册来解决这些问题。

Incident Manager

Incident Manager 是一个事件管理控制台，可帮助用户缓解和恢复影响其 AWS 托管应用程序的事件。

Incident Manager 通过通知响应者影响、突出显示相关故障排除数据以及提供协作工具使服务备份并运行，从而提高事件解决方案。Incident Manager 还可以自动执行响应计划，并允许响应者团队上报。

OpsCenter

OpsCenter 提供了一个中心位置，运营工程师和 IT 专业人员可在该位置查看、调查并解决与 AWS 资源相关的操作工作项 (OpsItems)。OpsCenter 旨在缩短影响 AWS 资源的问题的平均解决时间。此 Systems Manager 工具跨服务聚合和标准化 OpsItems，同时提供有关每个OpsItem、相关 OpsItems 和相关资源的上下文调查数据。OpsCenter 还提供了可用于解决问题的 Systems Manager Automation 运行手册。您可以为每个 OpsItem 指定可搜索的自定义数据。您还可以按状态和源查看自动生成的 OpsItems 相关摘要报告。

使用 Systems Manager 执行节点任务

利用本教程开始使用 AWS Systems Manager。您将学习如何启动由 Systems Manager 管理的 Amazon Elastic Compute Cloud（Amazon EC2）实例，以及如何连接到托管实例。

由于 Systems Manager 是多项工具的集合，无法通过一个演练或教程来介绍整个服务。本教程介绍了某些工具。

先决条件

开始之前，请确保您已完成 使用 Systems Manager 管理 EC2 实例 中的步骤。

使用已预安装 SSM Agent 的 AMI 启动实例

您可以根据以下过程所述使用 AWS Management Console 启动 Amazon EC2 实例。本教程旨在帮助您快速启动第一个托管实例，因此不会涵盖所有可能的选项。

启动实例

1. 通过以下网址打开 Amazon EC2 控制台：https://console.aws.amazon.com/ec2/。
2. 从 EC2 控制台控制面板中，在 Launch instance（启动实例）框中，选择 Launch instance（启动实例），然后从显示的选项中选择Launch instance（启动实例）。
3. 对于名称与标签的名称，为您的实例输入一个描述性名称。
4. 对于应用程序和操作系统映像（亚马逊机器映像），执行以下操作：
   a. 选择快速启动，然后选择 Amazon Linux。这是适用于您的实例的操作系统 (OS)。
   b. 对于亚马逊机器映像（AMI），请选择 Amazon Linux 2 的一个 HVM 版本。
5. 对于实例类型，从实例类型列表中，为您的实例选择硬件配置。选择 `t2.micro` 实例类型，这是默认情况下的选择。`t2.micro` 实例类型适用于 AWS 免费套餐。在 `t2.micro` 不可用的 AWS 区域 中，您可以使用“免费套餐”下的 `t3.micro` 实例。有关更多信息，请参阅 AWS 免费套餐。
6. 对于密钥对（登录）的密钥对名称，选择一个密钥对。
7. 对于网络设置，选择编辑。对于安全组名称，请注意向导已经为您创建并选择了安全组。使用以下步骤，您可以使用此安全组，或者也可以选择先前创建的安全组：
   a. 选择 Select existing security group（选择现有安全组）。
   b. 从 Common security groups（通用安全组）中，从现有安全组列表中选择您的安全组。
8. 如果您没有使用“默认主机管理配置”，请展开高级详细信息部分，对于 IAM 实例配置文件，选择您在 配置 Systems Manager 所需的实例权限 中进行设置时创建的实例配置文件。
9. 对于您的实例的其他配置设置，保留默认选择。
10. 在摘要窗格中，查看您的实例配置摘要。在您准备就绪后，选择启动实例。
11. 随即显示一个确认页面，通知您实例正在启动。选择 View all instances（查看所有实例）以关闭确认页面并返回控制台。
12. 在实例屏幕上，您可以查看启动状态。启动实例只需很短的时间。
13. 可能需要几分钟时间，实例才能显示为已托管，并已准备好让您连接到它。要检查您的实例是否通过了状态检查，请在状态检查列中查看此信息。

使用 Systems Manager 连接到托管实例

连接到您的托管实例

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。

2. 在导航窗格中，选择 Fleet Manager。
3. 选择要连接到的实例旁边的按钮。
4. 在节点操作菜单中，选择启动终端会话。
5. 选择 Connect（连接）。

清除您的实例

如果您已完成使用为本教程创建的托管实例的操作，则请终止该实例。终止实例将有效删除该实例。

要终止您的实例，请执行以下操作：

1. 通过以下网址打开 Amazon EC2 控制台：https://console.aws.amazon.com/ec2/。
2. 在导航窗格中，选择实例。在实例列表中选择实例。
3. 依次选择实例状态、终止实例。
4. 当系统提示您确认时，选择终止。

   Amazon EC2 关闭并终止您的实例。您的实例在终止之后，短时间内仍将在控制台上可见，然后该条目将自动删除。您不能自己从控制台显示中移除已终止的实例。

AWS Systems Manager 节点工具

AWS Systems Manager 可提供以下工具，用于访问、管理和配置托管式节点。托管节点是指在混合和多云环境中配置与 Systems Manager 一起使用的任何计算机。

主题

- AWS Systems Manager Compliance
- AWS Systems Manager Distributor
- AWS Systems Manager Fleet Manager
- AWS Systems Manager 混合激活
- AWS Systems Manager 清单
- AWS Systems Manager Patch Manager
- AWS Systems Manager Run Command
- AWS Systems Manager Session Manager

- AWS Systems Manager State Manager

AWS Systems Manager Compliance

您可以使用 Compliance（AWS Systems Manager 中的一项工具）扫描托管式节点实例集，了解补丁合规性和配置不一致性。您可以从多个 AWS 账户 和区域中收集并聚合数据，然后深入了解不合规的特定资源。预设情况下，Compliance 会显示关于Patch Manager中的修补以及State Manager中的关联的当前合规性数据。（Patch Manager 和State Manager也都是 AWS Systems Manager 中的工具。）要开始使用 Compliance，请打开 Systems Manager 控制台。在导航窗格中，选择 合规性。

可将来自 Patch Manager 的补丁合规性数据发送到 AWS Security Hub。Security Hub 能让您全面了解高优先级安全警报和合规性状态。它还监控您的机群的修补状态。有关更多信息，请参阅 将 Patch Manager 与 AWS Security Hub 集成。

Compliance 具备以下额外优势和功能：

- 使用 AWS Config 查看 Patch Manager 修补数据和 State Manager 关联的合规性历史记录及变更跟踪。
- 自定义 Compliance 以根据 IT 或业务要求创建您自己的合规性类型。
- 使用 Run Command（AWS Systems Manager 中的另一项工具）、State Manager或 Amazon EventBridge 修复问题。
- 将数据传送到 Amazon Athena 和 Amazon QuickSight 以生成机群范围的报告。

EventBridge 支持

支持此 Systems Manager 工具作为 Amazon EventBridge 规则中的一个事件类型。有关更多信息，请参阅 使用 Amazon EventBridge 监控 Systems Manager 事件 和 引用：Amazon EventBridge 事件模式和 Systems Manager 类型。

Chef InSpec 集成

Systems Manager 与 Chef InSpec 集成在一起。InSpec 是一个开源的运行时框架，您可以使用该框架在 GitHub 或 Amazon Simple Storage Service（Amazon S3）上创建人类可读的配置文件。然后您可以使用 Systems Manager 运行合规性扫描，并查看合规和不合规的托管式节点。有关更多信息，请参阅 将 Chef InSpec 配置文件与 Systems Manager Compliance 结合使用。

定价

Compliance 不另外收取费用。您仅需为实际使用的 AWS 资源付费。

内容

- 开始使用 Compliance
- 配置合规性权限
- 为 Compliance 创建资源数据同步
- 了解有关合规性的详细信息
- 删除用于 Compliance 的资源数据同步
- 使用 EventBridge 修复合规性问题
- 使用 AWS CLI 分配自定义合规性元数据

开始使用 Compliance

要开始使用 Compliance（AWS Systems Manager 中的一项工具），请完成以下任务。

Task	有关更多信息
Compliance 适用于Patch Manager中的补丁数据和State Manager中的关联。（Patch Manager和State Manager都是 AWS Systems Manager 中的工具。）Compliance 还适用于使用 Systems Manager 管理的托管式节点上的自定义合规性类型。验证您是否已在混合和多云环境中完成 Amazon Elastic Compute Cloud（Amazon EC2）实例以及非 EC2 计算机的设置要求。	为组织设置 Systems Manager 统一控制台
更新托管式节点使用的 AWS Identity and Access Management（IAM）角色以限制合规性权限。	配置合规性权限
如果您计划监控补丁合规性，请验证您是否已配置 Patch Manager。您必须使用 Patch Manager 执行修补操作，然后 Compliance 才能显示补丁合规性数据。	AWS Systems Manager Patch Manager

Task	有关更多信息
如果您计划监控关联合规性，请验证您是否已创建 State Manager 关联。您必须创建关联，然后 Compliance 才能显示关联合规性数据。	AWS Systems Manager State Manager
（可选）配置系统以查看合规性历史记录和变更跟踪。	查看合规性配置历史记录和变更跟踪
（可选）创建自定义合规性类型。	使用 AWS CLI 分配自定义合规性元数据
（可选）创建资源数据同步以将所有合规性数据聚合在一个目标 Amazon Simple Storage Service (Amazon S3) 存储桶。	为 Compliance 创建资源数据同步

配置合规性权限

作为最佳安全实践，建议您使用以下权限来更新托管式节点使用的 AWS Identity and Access Management（IAM）角色，以限制该节点使用 PutComplianceItems API 操作的功能。该 API 操作会在指定资源（例如 Amazon EC2 实例或托管式节点）上注册合规性类型及其他合规性详细信息。

如果节点是 Amazon EC2 实例，则必须使用以下权限来更新该实例使用的 IAM 实例配置文件。有关 Systems Manager 托管的 EC2 实例的实例配置文件的更多信息，请参阅配置 Systems Manager 所需的实例权限。对于其他类型的托管式节点，请使用以下权限来更新节点使用的 IAM 角色。有关更多信息，请参阅《IAM 用户指南》中的更新角色的权限。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:PutComplianceItems"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
```

```
                    "ec2:SourceInstanceARN": "${ec2:SourceInstanceARN}"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:PutComplianceItems"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ssm:SourceInstanceARN": "${ssm:SourceInstanceARN}"
                }
            }
        }
    ]
}
```

为 Compliance 创建资源数据同步

您可以使用 AWS Systems Manager 中的资源数据同步功能，将来自所有托管式节点的合规性数据发送到目标 Amazon Simple Storage Service (Amazon S3) 存储桶。在创建同步时，可以指定来自多个 AWS 账户、AWS 区域 和混合和多云环境的托管式节点。收集新的合规性数据后，资源数据同步自动更新集中式数据。所有合规性数据存储在目标 Amazon S3 存储桶中后，您可以使用 Amazon Athena 和 Amazon QuickSight 等服务查询和分析聚合数据。为 Compliance 配置资源数据同步是一次性操作。

通过使用 AWS Management Console，使用以下过程为 Compliance 创建资源数据同步。

创建和配置一个 S3 存储桶用于资源数据同步（控制台）

1. 通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 创建用来存储聚合合规性数据的存储桶。有关更多信息，请参阅 Amazon Simple Storage Service 用户指南中的创建存储桶。请记下存储桶名称和创建此存储桶的 AWS 区域。
3. 打开存储桶，选择 Permissions (权限) 选项卡，然后选择 Bucket Policy (存储桶策略)。
4. 将下面的存储桶策略复制并粘贴到策略编辑器中。将 amzn-s3-demo-bucket 和 *Account-ID* 分别替换为您创建的 S3 存储桶的名称和有效的 AWS 账户 ID。或者，使用 Amazon S3 前缀（子

目录）的名称替换 *Bucket-Prefix*。如果您未创建前缀，则从该策略的 ARN 中删除 *Bucket-Prefix*/。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "SSMBucketPermissionsCheck",
            "Effect": "Allow",
            "Principal": {
                "Service": "ssm.amazonaws.com"
            },
            "Action": "s3:GetBucketAcl",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket"
        },
        {
            "Sid": " SSMBucketDelivery",
            "Effect": "Allow",
            "Principal": {
                "Service": "ssm.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/Bucket-Prefix/*/
accountid=Account_ID_number/*"],
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "bucket-owner-full-control"
                }
            }
        }
    ]
}
```

创建资源数据同步

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。

2. 在导航窗格中，请选择 Fleet Manager。
3. 选择 Account management (账户管理)、Resource Data Syncs (资源数据同步)，然后选择 Create resource data sync (创建资源数据同步)。
4. 在 Sync name (同步名称) 字段中，输入同步配置的名称。
5. 在 Bucket name (存储桶名称) 字段中，输入您在此过程开始时创建的 Amazon S3 存储桶的名称。
6. （可选）在存储桶前缀字段中，输入 S3 存储桶前缀（子目录）的名称。
7. 在存储桶区域字段中，如果您创建的 S3 存储桶位于当前的 AWS 区域 中，请选择此区域。如果存储桶位于其他 AWS 区域 中，则请选择其他区域，然后输入该区域的名称。

   > **Note**
   >
   > 如果同步和目标 S3 存储桶位于不同区域，您可能需要支付数据传输价格。有关更多信息，请参阅 Amazon S3 定价。

8. 选择创建。

了解有关合规性的详细信息

Compliance 是 AWS Systems Manager 中的一项工具，可收集和报告与Patch Manager修补中修补状态和State Manager中关联有关的数据。（Patch Manager 和State Manager也都是 AWS Systems Manager 中的工具。）Compliance 还可报告有关您为托管式节点指定的自定义合规性类型的信息。本节包含有关每个合规性类型以及如何查看 Systems Manager 合规性数据的详细信息。本节还包含有关如何查看合规性历史记录和变更跟踪的信息。

> **Note**
>
> Systems Manager 与 Chef InSpec 集成在一起。InSpec 是一个开源的运行时框架，您可以使用该框架在 GitHub 或 Amazon Simple Storage Service（Amazon S3）上创建人类可读的配置文件。然后，您可以使用 Systems Manager 运行合规性扫描并查看合规和不合规的实例。有关更多信息，请参阅 将 Chef InSpec 配置文件与 Systems Manager Compliance 结合使用。

关于补丁合规性

使用 Patch Manager 在实例上安装补丁之后，控制台、AWS Command Line Interface (AWS CLI) 命令响应或相应 Systems Manager API 操作的响应中将立即出现合规性状态信息。

有关补丁合规性状态值的信息，请参阅 补丁合规性状态值。

关于 State Manager 关联合规性

创建一个或多个 State Manager 关联之后，控制台、AWS CLI 命令响应或相应 Systems Manager API 操作的响应中将立即出现合规性状态信息。对于关联，Compliance 显示 `Compliant` 或 `Non-compliant` 的状态和分配给关联的严重性级别，如 `Critical` 或 `Medium`。

State Manager 在托管节点上执行关联时会触发合规聚合过程，该过程会更新该节点上所有关联的合规状态。合规报告中的 `ExecutionTime` 值表示 Systems Manager 捕获合规状态的时间，而非在托管节点上执行关联的时间。这意味着多个关联即使在不同时间执行，也可能显示相同的 `ExecutionTime` 值。要确定关联的实际执行时间，请使用 AWS CLI 命令 describe-association-execution-targets 来参阅关联执行历史记录，或在控制台中查看执行详细信息。

关于自定义合规性

您可以将合规性元数据分配给托管式节点。然后此元数据可以与用于合规性报告目的的其他合规性数据聚合。例如，假设您的企业在您的托管式节点上运行软件 X 的版本 2.0、3.0 和 4.0。公司希望在版本 4.0 上实现标准化，这意味着运行版本 2.0 和 3.0 的实例将不合规。您可以使用 PutComplianceItems API 操作显式注释哪些托管式节点在运行软件 X 的较旧版本。您只能使用 AWS CLI、AWS Tools for Windows PowerShell 或软件开发工具包分配合规性元数据。以下 CLI 示例命令会将合规性元数据分配给一个托管实例并以要求的格式 `Custom:` 指定合规性类型。将每个*#######*替换为您自己的信息。

Linux & macOS

```
aws ssm put-compliance-items \
    --resource-id i-1234567890abcdef0 \
    --resource-type ManagedInstance \
    --compliance-type Custom:SoftwareXCheck \
    --execution-summary ExecutionTime=AnyStringToDenoteTimeOrDate \
    --items
 Id=Version2.0,Title=SoftwareXVersion,Severity=CRITICAL,Status=NON_COMPLIANT
```

Windows

```
aws ssm put-compliance-items ^
    --resource-id i-1234567890abcdef0 ^
    --resource-type ManagedInstance ^
    --compliance-type Custom:SoftwareXCheck ^
    --execution-summary ExecutionTime=AnyStringToDenoteTimeOrDate ^
```

```
    --items
Id=Version2.0,Title=SoftwareXVersion,Severity=CRITICAL,Status=NON_COMPLIANT
```

> **Note**
>
> ResourceType 参数仅支持 ManagedInstance。如果您将自定义合规性添加到托管式 AWS IoT Greengrass 核心设备，则必须指定 ManagedInstance 的 ResourceType。

然后合规经理可以查看摘要，或创建有关哪些托管式节点合规或不合规的报告。您可将最多 10 个不同的自定义合规性类型分配给一个托管式节点。

有关如何创建自定义合规性类型并查看合规性数据的示例，请参阅 使用 AWS CLI 分配自定义合规性元数据。

查看当前合规性数据

本节介绍如何在 Systems Manager 控制台中以及如何使用 AWS CLI 查看合规性数据。有关如何查看补丁和关联合规性历史记录和变更跟踪的信息，请参阅 查看合规性配置历史记录和变更跟踪。

主题

- 查看当前合规性数据（控制台）
- 查看当前合规性数据 (AWS CLI)

查看当前合规性数据（控制台）

使用以下过程在 Systems Manager 控制台中查看合规性数据。

在 Systems Manager 控制台中查看合规性报告

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 合规性。
3. 在 Compliance dashboard filtering（合规性控制面板筛选）部分中，选择一个选项来筛选合规性数据。Compliance resources summary（合规性资源摘要）部分会根据您选择的筛选条件显示合规性数据的计数。
4. 要深入了解资源以获取更多信息，请向下滚动至 Details overview for resources（资源详细信息概览）区域，然后选择托管式节点的 ID。

5. 在 Instance ID（实例 ID）或 Name（名称）详细信息页面上，选择 Configuration compliance（配置合规性）选项卡以查看详细的托管式节点配置合规性报告。

> **Note**
>
> 有关修复合规性问题的信息，请参阅 使用 EventBridge 修复合规性问题。

查看当前合规性数据 (AWS CLI)

可以通过使用以下 AWS CLI 命令，在 AWS CLI 中查看修补、关联和自定义合规性类型的合规性数据摘要。

list-compliance-summaries

根据您指定的筛选条件返回合规和不合规关联状态的摘要计数。（API：ListComplianceSummaries）

list-resource-compliance-summaries

返回资源级摘要计数。根据您指定的筛选条件，摘要包括有关合规和不合规状态的信息，以及详细的合规性项目严重性计数。（API：ListResourceComplianceSummaries）

可以使用以下 AWS CLI 命令查看修补的其他合规性数据。

describe-patch-group-state

返回补丁组的高级聚合补丁合规性状态。（API：DescribePatchGroupState）

describe-instance-patch-states-for-patch-group

返回指定补丁组中实例的高级补丁状态。（API：DescribeInstancePatchStatesForPatchGroup）

> **Note**
>
> 有关如何使用 AWS CLI 配置修补和查看补丁合规性详细信息的说明，请参阅 教程：使用 AWS CLI 修补服务器环境。

查看合规性配置历史记录和变更跟踪

Systems Manager Compliance 显示您的托管式节点的最新修补和关联合规性数据。您可以使用 AWS Config 查看修补和关联合规性历史记录和变更追踪。AWS Config 提供关于 AWS 账户中 AWS 资源配置的详细视图。这些信息包括资源之间的关联方式以及资源以前的配置方式，让您了解资源的配置和关系如何随着的时间的推移而更改。要查看修补和关联合规性历史记录和变更跟踪，您必须在 AWS Config 中打开以下资源：

- `SSM:PatchCompliance`
- `SSM:AssociationCompliance`

有关如何在 AWS Config 中选择和配置这些特定资源的信息，请参阅 AWS Config Developer Guide 中的选择哪些资源 AWS Config 记录。

> **Note**
>
> 有关 AWS Config 定价的信息，请参阅 定价。

删除用于 Compliance 的资源数据同步

如果您不再需要使用 AWS Systems Manager Compliance 查看合规性数据，那么我们建议删除用于 Compliance 数据收集的资源数据同步。

删除 Compliance 资源数据同步

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Fleet Manager。
3. 选择 Account management (账户管理)、Resource data syncs (资源数据同步)。
4. 在列表中，选择一个同步。

> **Important**
>
> 确保您选择了用于 Compliance 的同步。Systems Manager 可为多项工具的资源数据同步提供支持。如果选择的同步错误，则可能会导致 Systems Manager Explorer 或 Systems Manager Inventory 的数据聚合中断。

5. 选择删除。
6. 删除存储数据的 Amazon Simple Storage Service (Amazon S3) 存储桶。有关删除 S3 存储桶的信息，请参阅删除存储桶。

使用 EventBridge 修复合规性问题

您可以使用 Run Command（AWS Systems Manager 中的一项工具）快速修复补丁和关联合规性问题。您可以将实例、AWS IoT Greengrass 核心设备 ID 或标签设为目标，并运行 AWS-RunPatchBaseline 文档或 AWS-RefreshAssociation 文档。如果刷新关联或重新运行补丁基准也未能解决合规性问题，则需要调查您的关联、补丁基准或实例配置，以了解 Run Command 操作未能解决问题的原因。

有关修补的更多信息，请参阅 AWS Systems Manager Patch Manager 和用于修补的 SSM 命令文档：AWS-RunPatchBaseline。

有关关联的更多信息，请参阅 在 Systems Manager 中使用关联。。

有关运行命令的更多信息，请参阅 AWS Systems Manager Run Command。

将 Compliance 指定为 EventBridge 事件的目标

您也可以将 Amazon EventBridge 配置为执行操作以响应 Systems Manager Compliance 事件。例如，如果一个或多个托管式节点未能安装重要补丁更新或运行安装反病毒软件的关联，则您可以将 EventBridge 配置为在 Compliance 事件发生时运行 AWS-RunPatchBaseline 文档或 AWS-RefreshAssocation 文档。

使用以下过程将 Compliance 配置为 EventBridge 事件的目标。

将 Compliance 配置为 EventBridge 事件的目标（控制台）

1. 访问 https://console.aws.amazon.com/events/，打开 Amazon EventBridge 控制台。
2. 在导航窗格中，选择规则。
3. 选择创建规则。
4. 为规则输入名称和描述。

   规则不能与同一 AWS 区域中和同一事件总线上的另一条规则的名称相同。
5. 对于事件总线，请选择要与此规则关联的事件总线。如果您希望此规则响应来自您自己的 AWS 账户的匹配事件，请选择 defaul（默认）。当您账户中的某个 AWS 服务发出一个事件时，它始终会发送到您账户的默认事件总线。

6. 对于规则类型，选择具有事件模式的规则。
7. 选择下一步。
8. 对于事件源，选择AWS 事件或 EventBridge 合作伙伴事件。
9. 在 Event pattern（事件模式）部分，选择 Event pattern form（事件模式表单）。
10. 对于事件源，选择AWS 服务。
11. 对于 service（AWS 服务），选择 Systems Manager。
12. 在 Event type (事件类型) 字段中，选择 Configuration Compliance (配置合规性)。
13. 对于 Specific detail type(s)（具体详细信息类型），选择 Configuration Compliance State Change（配置合规性状态更改）。
14. 选择下一步。
15. 对于目标类型，选择AWS 服务。
16. 对于 Select a target（选择一个目标），选择 Systems Manager Run Command。
17. 在 Document (文档) 列表中，选择在调用目标时要运行的 Systems Manager 文档（SSM 文档）。例如，对于不合规补丁事件选择 `AWS-RunPatchBaseline`，对于不合规关联事件选择 `AWS-RefreshAssociation`。
18. 指定其余字段和参数的信息。

> **Note**
>
> 必填字段和参数的名称旁有一个星号 (*)。要创建目标，您必须为每个必填参数或字段指定一个值。如果不指定，系统虽然会创建规则，但该规则不会运行。

19. 选择下一步。
20. （可选）为规则输入一个或多个标签。有关更多信息，请参阅 Amazon EventBridge 用户指南中的标记 Amazon EventBridge 资源。
21. 选择 Next（下一步）。
22. 查看规则详细信息并选择创建规则。

使用 AWS CLI 分配自定义合规性元数据

以下步骤为您演示了使用 AWS Command Line Interface (AWS CLI) 调用 AWS Systems Manager PutComplianceItems API 操作将自定义合规性元数据分配给某个资源的过程。您也可以使用此 API 操作将补丁或关联合规性元数据手动分配给某个托管式节点，如以下演练中所示。有关自定义合规性的更多信息，请参阅 关于自定义合规性。

将自定义合规性元数据分配给某个托管实例 (AWS CLI)

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 运行以下命令，将自定义合规性元数据分配给某个托管式节点。将每个*******替换为您自己的信息。ResourceType 参数仅支持 ManagedInstance 的值。即使您将自定义合规性元数据分配给托管式 AWS IoT Greengrass 核心设备，也请指定此值。

   Linux & macOS

   ```
   aws ssm put-compliance-items \
       --resource-id instance_ID \
       --resource-type ManagedInstance \
       --compliance-type Custom:user-defined_string \
       --execution-summary ExecutionTime=user-defined_time_and/or_date_value \
       --items Id=user-defined_ID,Title=user-
   defined_title,Severity=one_or_more_comma-separated_severities:CRITICAL, MAJOR,
    MINOR,INFORMATIONAL, or UNSPECIFIED,Status=COMPLIANT or NON_COMPLIANT
   ```

   Windows

   ```
   aws ssm put-compliance-items ^
       --resource-id instance_ID ^
       --resource-type ManagedInstance ^
       --compliance-type Custom:user-defined_string ^
       --execution-summary ExecutionTime=user-defined_time_and/or_date_value ^
       --items Id=user-defined_ID,Title=user-
   defined_title,Severity=one_or_more_comma-separated_severities:CRITICAL, MAJOR,
    MINOR,INFORMATIONAL, or UNSPECIFIED,Status=COMPLIANT or NON_COMPLIANT
   ```

3. 重复上一步以将其他自定义合规性元数据分配给一个或多个节点。您也可以使用以下命令，将补丁或关联合规性元数据手动分配给托管式节点：

   关联合规性元数据

   Linux & macOS

   ```
   aws ssm put-compliance-items \
       --resource-id instance_ID \
       --resource-type ManagedInstance \
       --compliance-type Association \
   ```

```
    --execution-summary ExecutionTime=user-defined_time_and/or_date_value \
    --items Id=user-defined_ID,Title=user-
defined_title,Severity=one_or_more_comma-separated_severities:CRITICAL, MAJOR,
 MINOR,INFORMATIONAL, or UNSPECIFIED,Status=COMPLIANT or NON_COMPLIANT
```

Windows

```
aws ssm put-compliance-items ^
    --resource-id instance_ID ^
    --resource-type ManagedInstance ^
    --compliance-type Association ^
    --execution-summary ExecutionTime=user-defined_time_and/or_date_value ^
    --items Id=user-defined_ID,Title=user-
defined_title,Severity=one_or_more_comma-separated_severities:CRITICAL, MAJOR,
 MINOR,INFORMATIONAL, or UNSPECIFIED,Status=COMPLIANT or NON_COMPLIANT
```

补丁合规性元数据

Linux & macOS

```
aws ssm put-compliance-items \
    --resource-id instance_ID \
    --resource-type ManagedInstance \
    --compliance-type Patch \
    --execution-summary ExecutionTime=user-defined_time_and/
or_date_value,ExecutionId=user-defined_ID,ExecutionType=Command  \
    --items Id=for_example, KB12345,Title=user-
defined_title,Severity=one_or_more_comma-separated_severities:CRITICAL,
 MAJOR, MINOR,INFORMATIONAL, or UNSPECIFIED,Status=COMPLIANT or
 NON_COMPLIANT,Details="{PatchGroup=name_of_group,PatchSeverity=the_patch_severity,
 for example, CRITICAL}"
```

Windows

```
aws ssm put-compliance-items ^
    --resource-id instance_ID ^
    --resource-type ManagedInstance ^
    --compliance-type Patch ^
    --execution-summary ExecutionTime=user-defined_time_and/
or_date_value,ExecutionId=user-defined_ID,ExecutionType=Command  ^
```

```
    --items Id=for_example, KB12345,Title=user-
defined_title,Severity=one_or_more_comma-separated_severities:CRITICAL,
 MAJOR, MINOR,INFORMATIONAL, or UNSPECIFIED,Status=COMPLIANT or
 NON_COMPLIANT,Details="{PatchGroup=name_of_group,PatchSeverity=the_patch_severity,
 for example, CRITICAL}"
```

4. 运行以下命令，查看特定托管式节点的合规性项目列表。使用筛选条件深入了解特定合规性数据。

   Linux & macOS

   ```
   aws ssm list-compliance-items \
       --resource-ids instance_ID \
       --resource-types ManagedInstance \
       --filters one_or_more_filters
   ```

   Windows

   ```
   aws ssm list-compliance-items ^
       --resource-ids instance_ID ^
       --resource-types ManagedInstance ^
       --filters one_or_more_filters
   ```

   以下示例向您演示如何将此命令与筛选条件结合使用。

   Linux & macOS

   ```
   aws ssm list-compliance-items \
       --resource-ids i-02573cafcfEXAMPLE \
       --resource-type ManagedInstance \
       --filters Key=DocumentName,Values=AWS-RunPowerShellScript
    Key=Status,Values=NON_COMPLIANT,Type=NotEqual
    Key=Id,Values=cee20ae7-6388-488e-8be1-a88ccEXAMPLE
    Key=Severity,Values=UNSPECIFIED
   ```

   Windows

   ```
   aws ssm list-compliance-items ^
       --resource-ids i-02573cafcfEXAMPLE ^
       --resource-type ManagedInstance ^
       --filters Key=DocumentName,Values=AWS-RunPowerShellScript
    Key=Status,Values=NON_COMPLIANT,Type=NotEqual
   ```

```
 Key=Id,Values=cee20ae7-6388-488e-8be1-a88ccEXAMPLE
 Key=Severity,Values=UNSPECIFIED
```

Linux & macOS

```
aws ssm list-resource-compliance-summaries \
    --filters Key=OverallSeverity,Values=UNSPECIFIED
```

Windows

```
aws ssm list-resource-compliance-summaries ^
    --filters Key=OverallSeverity,Values=UNSPECIFIED
```

Linux & macOS

```
aws ssm list-resource-compliance-summaries \
    --filters Key=OverallSeverity,Values=UNSPECIFIED
 Key=ComplianceType,Values=Association Key=InstanceId,Values=i-02573cafcfEXAMPLE
```

Windows

```
aws ssm list-resource-compliance-summaries ^
    --filters Key=OverallSeverity,Values=UNSPECIFIED
 Key=ComplianceType,Values=Association Key=InstanceId,Values=i-02573cafcfEXAMPLE
```

5. 运行以下命令查看合规性状态的摘要。使用筛选条件深入了解特定合规性数据。

```
aws ssm list-resource-compliance-summaries --filters One or more filters.
```

以下示例向您演示如何将此命令与筛选条件结合使用。

Linux & macOS

```
aws ssm list-resource-compliance-summaries \
    --filters Key=ExecutionType,Values=Command
```

Windows

```
aws ssm list-resource-compliance-summaries ^
    --filters Key=ExecutionType,Values=Command
```

Linux & macOS

```
aws ssm list-resource-compliance-summaries \
    --filters Key=AWS:InstanceInformation.PlatformType,Values=Windows
 Key=OverallSeverity,Values=CRITICAL
```

Windows

```
aws ssm list-resource-compliance-summaries ^
    --filters Key=AWS:InstanceInformation.PlatformType,Values=Windows
 Key=OverallSeverity,Values=CRITICAL
```

6. 运行以下命令查看某个合规性类型的合规资源和不合规资源的摘要计数。使用筛选条件深入了解特定合规性数据。

```
aws ssm list-compliance-summaries --filters One or more filters.
```

以下示例向您演示如何将此命令与筛选条件结合使用。

Linux & macOS

```
aws ssm list-compliance-summaries \
    --filters Key=AWS:InstanceInformation.PlatformType,Values=Windows
 Key=PatchGroup,Values=TestGroup
```

Windows

```
aws ssm list-compliance-summaries ^
    --filters Key=AWS:InstanceInformation.PlatformType,Values=Windows
 Key=PatchGroup,Values=TestGroup
```

Linux & macOS

```
aws ssm list-compliance-summaries \
    --filters Key=AWS:InstanceInformation.PlatformType,Values=Windows
 Key=ExecutionId,Values=4adf0526-6aed-4694-97a5-14522EXAMPLE
```

Windows

```
aws ssm list-compliance-summaries ^
    --filters Key=AWS:InstanceInformation.PlatformType,Values=Windows
 Key=ExecutionId,Values=4adf0526-6aed-4694-97a5-14522EXAMPLE
```

AWS Systems Manager Distributor

Distributor 是 AWS Systems Manager 中的一项工具，可以帮助您将软件打包并发布到 AWS Systems Manager 托管式节点。您可以打包和发布自己的软件，也可以使用 Distributor 查找和发布 AmazonCloudWatchAgent 等 AWS 提供的代理软件包，或者 Trend Micro 等第三方软件包。发布软件包会将特定版本的软件包文档公布给通过节点 ID、AWS 账户 ID、标签或 AWS 区域 标识的托管式节点。要开始使用 Distributor，请打开 Systems Manager 控制台。在导航窗格中，请选择 Distributor。

在 Distributor 中创建软件包后，可以使用以下方式之一安装该软件包：

- 使用 AWS Systems Manager Run Command 进行一次性安装
- 使用 AWS Systems Manager State Manager 按计划执行安装

> **Important**
>
> 第三方卖家分发的程序包不由 AWS 管理，而是由程序包供应商发布。建议开展额外的尽职调查，确保遵守您的内部安全控制措施。安全性是 AWS 和您的共同责任。这被描述为责任共担共担模式。要了解更多信息，请参阅责任共担模式。

我的组织如何从 Distributor 获益？

Distributor 具备下列优势：

- 一个软件包，多个平台

  在 Distributor 中创建软件包时，系统会创建 AWS Systems Manager 文档（SSM 文档）。您可以将 .zip 格式文件附加到此文档中。运行 Distributor 时，系统会处理 SSM 文档中的指令，并将.zip 格式文件中的软件包安装到指定目标上。Distributor 支持 Windows、Ubuntu Server、Debian Server 和 Red Hat Enterprise Linux 等多种操作系统。有关支持的平台的更多信息，请参阅 支持的软件包平台和架构。

- 跨托管实例组控制软件包访问权限

  您可以使用 Run Command 或State Manager控制哪些托管式节点能够获取软件包，以及获取该软件包的哪个版本。Run Command 和State Manager都是 AWS Systems Manager 中的工具。托管式节点可以按实例或设备 ID、AWS 账户 账号、标签或 AWS 区域 进行分组。您可以使用 State Manager 关联向不同的实例组提供不同版本的软件包。

- 包含众多 AWS 代理软件包并且随时可用

  Distributor 包含许多可供您随时部署到托管式节点的 AWS 代理软件包。可以在 Distributor `Packages` 列表页面中查找 `Amazon` 发布的软件包。示例包括 `AmazonCloudWatchAgent` 和 `AWSPVDriver`。

- 自动部署

  要使您的环境保持最新，请使用 State Manager，在首次启动这些计算机时，安排要在目标托管式节点上自动部署的软件包。

谁应该使用 Distributor？

- 希望创建新软件包或将现有软件包（包括 AWS 发布的软件包）同时部署到多个 Systems Manager 托管式节点的任何 AWS 客户。
- 创建软件包的软件开发人员。
- 负责保持 Systems Manager 托管式节点始终使用最新软件包的管理员。

Distributor 具有哪些功能？

- 将软件包部署到 Windows 和 Linux 实例

  借助 Distributor，您可以将软件包部署到适用于 Linux 和 Windows Server 的 Amazon Elastic Compute Cloud (Amazon EC2) 实例和 AWS IoT Greengrass 核心设备。有关支持的实例操作系统类型的列表，请参阅 the section called “支持的软件包平台和架构”。

> **Note**
>
> Distributor 在 macOS 操作系统上不受支持。

- 一次性或按计划自动部署软件包

  您可以选择一次性、定期或每当默认软件包版本更改为其他版本时部署软件包。

- 完全重新安装软件包或执行就地更新

  要安装新的软件包版本，可以根据您提供的更新脚本 完全卸载当前版本并在其位置安装新版本，或者仅使用新组件和更新的组件更新当前版本。软件包应用程序在重新安装期间不可用，但在就地更新期间保持可用。对于安全监控应用程序或需要避免应用程序停机的其他情况，就地更新是特别有用的。

- 支持通过控制台、CLI、PowerShell 和开发工具包等方式访问 Distributor 功能

  您可以通过 Systems Manager 控制台、AWS Command Line Interface (AWS CLI)、AWS Tools for PowerShell 或您选择的 AWS 开发工具包来使用 Distributor。

- IAM 访问控制

  通过使用 AWS Identity and Access Management (IAM) 策略，您可以控制组织的哪些成员能够创建、更新、部署或删除软件包或软件包版本。例如，您可能需要授予管理员部署软件包但不能更改软件包或创建新软件包版本的权限。

- 支持日志记录和审计功能

  您可以通过与其他 AWS 服务集成来审计和记录您的 AWS 账户中的 Distributor 用户操作日志。有关更多信息，请参阅 审计和记录 Distributor 活动。

Distributor 中的软件包是什么？

软件包 是包含以下内容的可安装软件或资产的集合。

- 每个目标操作系统平台的软件 .zip 文件。每个 .zip 文件必须包含以下内容。
  - install 和 uninstall 脚本。基于 Windows Server 的托管式节点需要使用 PowerShell 脚本（名为 `install.ps1` 和 `uninstall.ps1` 的脚本）。基于 Linux 的托管式节点需要使用 Shell 脚本（名为 `install.sh` 和 `uninstall.sh` 的脚本）。AWS Systems Manager SSM Agent 读取并执行 install 和 uninstall 脚本中的指令。
  - 可执行文件。SSM Agent 必须找到此可执行文件才能在目标托管式节点上安装软件包。

- 用于描述软件包内容的 JSON 格式的清单文件。清单不包含在 .zip 文件中，而是存储在与构成软件包的 .zip 文件相同的 Amazon Simple Storage Service (Amazon S3) 存储桶中。该清单指定软件包版本，并将软件包中的 .zip 格式文件映射到目标托管式节点属性，例如操作系统版本或架构。有关如何创建清单的信息，请参阅 步骤 2：创建 JSON 软件包清单。

当您在 Distributor 控制台中选择 Simple (简单) 软件包创建时，Distributor 将根据软件可执行文件名称以及目标平台和架构，为您生成安装和卸载脚本、文件哈希以及 JSON 软件包清单。

支持的软件包平台和架构

您可以使用 Distributor 将软件包发布到以下 Systems Manager 托管式节点平台。版本值必须与您设定为目标的操作系统 Amazon Machine Image (AMI) 的发行版本完全匹配。有关确定此版本的更多信息，请参阅 步骤 2：创建 JSON 软件包清单 的步骤 4。

> **Note**
>
> Systems Manager 不支持 AWS IoT Greengrass 核心设备的以下所有操作系统。有关更多信息，请参阅 AWS IoT Greengrass 开发人员指南中的设置 AWS IoT Greengrass Version 2 核心设备。

平台	清单文件中的代码值	支持的架构
AlmaLinux	`almalinux`	x86_64 ARM64
Amazon Linux 1、Amazon Linux 2 和 Amazon Linux 2023	`amazon`	x86_64 或 x86 ARM64（Amazon Linux 2 和 AL2023、A1 实例类型）
CentOS	`centos`	x86_64 或 x86
Debian Server	`debian`	x86_64 或 x86
openSUSE	`opensuse`	x86_64 或 x86
openSUSE Leap	`opensuseleap`	x86_64 或 x86

平台	清单文件中的代码值	支持的架构
Oracle Linux	`oracle`	x86_64
Red Hat Enterprise Linux (RHEL)	`redhat`	x86_64 或 x86 ARM64（RHEL 7.6 及更高版本，A1 实例类型）
Rocky Linux	`rocky`	x86_64 ARM64
SUSE Linux Enterprise Server (SLES)	`suse`	x86_64 或 x86
Ubuntu Server	`ubuntu`	x86_64 或 x86 ARM64（Ubuntu Server 16 及更高版本，A1 实例类型）
Windows Server	`windows`	x86_64 或者 x86

主题

- 设置 Distributor
- 处理 Distributor 程序包
- 审计和记录 Distributor 活动
- AWS Systems ManagerDistributor 故障排除

设置 Distributor

在使用 Distributor（AWS Systems Manager 中的一项工具）创建、管理和部署软件包之前，请按照以下步骤操作。

满足 Distributor 先决条件

在开始使用 Distributor（AWS Systems Manager 中的一项工具）之前，请确保环境满足以下要求。

Distributor 先决条件

要求	描述
SSM Agent	必须在要部署软件包或从中删除软件包的托管式节点上安装 AWS Systems Manager SSM Agent 版本 2.3.274.0 或更高版本。 要安装或更新 SSM Agent，请参阅 使用 SSM Agent。
AWS CLI	（可选）要使用 AWS Command Line Interface (AWS CLI) 而不是 Systems Manager 控制台创建和管理软件包，请在本地计算机上安装最新版本的 AWS CLI。 有关如何安装或升级 CLI 的更多信息，请参阅 AWS Command Line Interface 用户指南中的安装 AWS Command Line Interface。
AWS Tools for PowerShell	（可选）要使用 Tools for PowerShell 而不是 Systems Manager 控制台创建和管理软件包，请在本地计算机上安装最新版本的 Tools for PowerShell。 要详细了解如何安装或升级 Tools for PowerShell，请参阅《AWS Tools for PowerShell 用户指南》中的设置 AWS Tools for Windows PowerShell 或 AWS Tools for PowerShell Core。

> **Note**
>
> Systems Manager 不支持通过使用 Distributor 将软件包分发到 Oracle Linux 托管式节点。

验证或创建具有 Distributor 权限的 IAM 实例配置文件

默认情况下，AWS Systems Manager 没有在您的实例上执行操作的权限。您必须通过使用 AWS Identity and Access Management (IAM) 实例配置文件来授予访问权限。实例配置文件是一个容器，可在启动时将 IAM 角色信息传递给 Amazon Elastic Compute Cloud (Amazon EC2) 实例。此要求适用于所有 Systems Manager 工具的权限，而不仅仅是 Distributor 的权限。

> **Note**
>
> 配置边缘设备以运行 AWS IoT Greengrass Core 软件和 SSM Agent 时，请指定一个 IAM 服务角色，以便 Systems Manager 对其执行操作。您无需使用实例配置文件配置托管式边缘设备。

如果您已使用其他 Systems Manager 工具（例如 Run Command 和State Manager），说明已将具有 Distributor 所需权限的实例配置文件附加到实例。确保您有权执行 Distributor 任务的最简单方法是，将 AmazonSSMManagedInstanceCore 策略附加到实例配置文件。有关更多信息，请参阅配置 Systems Manager 所需的实例权限。

控制用户对程序包的访问

使用 AWS Identity and Access Management (IAM) policy，可以控制哪些用户能够创建、部署和管理软件包。您还可以控制他们能在托管式节点上执行哪些 Run Command 和 State Manager API 操作。与 Distributor 一样，Run Command 和State Manager也都是 AWS Systems Manager 中的工具。

ARN 格式

用户定义的软件包与文档 Amazon Resource Name (ARNs) 相关联，其格式如下。

```
arn:aws:ssm:region:account-id:document/document-name
```

示例如下：

```
arn:aws:ssm:us-west-1:123456789012:document/ExampleDocumentName
```

您可以使用 AWS 提供的一对默认 IAM policy（一个用于终端用户，一个用于管理员）授予对 Distributor 活动的权限。或者，您也可以创建适合您的权限要求的自定义 IAM policy。

有关在 IAM policy 中使用变量的更多信息，请参阅 IAM policy 元素：变量。

有关如何创建策略并将策略附加到用户或组的信息，请参阅《IAM 用户指南》中的创建 IAM policy 及添加和移除 IAM policy。

创建或选择 Amazon S3 存储桶来存储 Distributor 软件包

在 AWS Systems Manager 控制台中使用简单工作流创建软件包时，您可以选择 Distributor 会将软件上载到的现有 Amazon Simple Storage Service（Amazon S3）存储桶。Distributor 是 AWS Systems Manager 中的一项工具。在 Advanced (高级) 工作流中，您必须在开始之前将软件或资产的 .zip 文件上载到 Amazon S3 存储桶中。无论在控制台中是使用 Simple (简单) 还是 Advanced (高级) 工作流或是使用 API 来创建软件包，在开始创建软件包之前，您都必须具有 Amazon S3 存储桶。作为软件包创建过程的一部分，Distributor 将可安装的软件和资产从此存储桶复制到内部 Systems Manager 存储。由于资产已复制到内部存储，因此您可以在软件包创建完成时删除 Amazon S3 存储桶或调整其用途。

有关如何创建存储桶的更多信息，请参阅 Amazon Simple Storage Service 入门指南中的创建存储桶。有关如何运行 AWS CLI 命令创建存储桶的更多信息，请参阅 AWS CLI 命令参考中的 mb。

处理 Distributor 程序包

您可以使用 AWS Systems Manager 控制台、AWS 命令行工具（AWS CLI 和 AWS Tools for PowerShell）及 AWS SDK，在 Distributor 中添加、管理或部署软件包。Distributor 是 AWS Systems Manager 中的一项工具。在将软件包添加到 Distributor 之前，请：

- 创建和压缩可安装资产。
- （可选）为软件包创建 JSON 清单文件。使用 Distributor 控制台中的 Simple (简单) 软件包创建过程不需要执行此操作。简单软件包创建会生成一个 JSON 清单文件。

  您可以使用 AWS Systems Manager 控制台或文本或 JSON 编辑器来创建清单文件。
- 让 Amazon Simple Storage Service (Amazon S3) 存储桶准备好存储可安装的资产或软件。如果您使用 Advanced (高级) 软件包创建过程，在开始之前，请先将资产上载到您的 Amazon S3 存储桶。

> **Note**
>
> 完成软件包创建后，您可以删除此存储桶或调整其用途，因为 Distributor 会在软件包创建过程中将软件包内容移动到内部 Systems Manager 存储桶。

AWS 发布的软件包已打包，可随时进行部署。要将 AWS 发布的软件包部署到托管式节点，请参阅 安装或更新 Distributor 软件包。

您可以共享 AWS 账户 之间的 Distributor 软件包。在 AWS CLI 命令中使用从另一个账户共享的软件包时，请使用该软件包的 Amazon Resource Name (ARN)，而不是软件包名称。

主题

- 在 Distributor 中查看软件包
- 在 Distributor 中创建软件包
- 在控制台中编辑 Distributor 软件包权限
- 在控制台中编辑 Distributor 软件包标签
- 为 Distributor 软件包添加版本
- 安装或更新 Distributor 软件包
- 卸载 Distributor 软件包
- 删除 Distributor 软件包

在 Distributor 中查看软件包

要查看可用于安装的软件包，可以使用 AWS Systems Manager 控制台或您的首选 AWS 命令行工具。Distributor 是 AWS Systems Manager 中的一项工具。要访问 Distributor，请打开 AWS Systems Manager 控制台，然后在左侧导航窗格中选择 Distributor。您将看到可供您使用的所有软件包。

以下小节介绍如何使用您的首选命令行工具查看 Distributor 软件包。

使用命令行查看软件包

本节包含有关如何使用您的首选命令行工具查看使用提供的命令的 Distributor 软件包的信息。

Linux & macOS

在 Linux 上使用 AWS CLI 查看软件包

- 要查看所有软件包（不包含共享软件包），请运行以下命令。

```
aws ssm list-documents \
    --filters Key=DocumentType,Values=Package
```

- 要查看 Amazon 拥有的所有软件包，请运行以下命令。

```
aws ssm list-documents \
```

```
    --filters Key=DocumentType,Values=Package Key=Owner,Values=Amazon
```

- 要查看第三方拥有的所有软件包，请运行以下命令。

```
aws ssm list-documents \
    --filters Key=DocumentType,Values=Package Key=Owner,Values=ThirdParty
```

Windows

在 Windows 上使用 AWS CLI 查看软件包

- 要查看所有软件包（不包含共享软件包），请运行以下命令。

```
aws ssm list-documents ^
    --filters Key=DocumentType,Values=Package
```

- 要查看 Amazon 拥有的所有软件包，请运行以下命令。

```
aws ssm list-documents ^
    --filters Key=DocumentType,Values=Package Key=Owner,Values=Amazon
```

- 要查看第三方拥有的所有软件包，请运行以下命令。

```
aws ssm list-documents ^
    --filters Key=DocumentType,Values=Package Key=Owner,Values=ThirdParty
```

PowerShell

使用 Tools for PowerShell 查看软件包

- 要查看所有软件包（不包含共享软件包），请运行以下命令。

```
$filter = New-Object Amazon.SimpleSystemsManagement.Model.DocumentKeyValuesFilter
$filter.Key = "DocumentType"
$filter.Values = "Package"

Get-SSMDocumentList `
    -Filters @($filter)
```

- 要查看 Amazon 拥有的所有软件包，请运行以下命令。

```
$typeFilter = New-Object
 Amazon.SimpleSystemsManagement.Model.DocumentKeyValuesFilter
$typeFilter.Key = "DocumentType"
$typeFilter.Values = "Package"

$ownerFilter = New-Object
 Amazon.SimpleSystemsManagement.Model.DocumentKeyValuesFilter
$ownerFilter.Key = "Owner"
$ownerFilter.Values = "Amazon"

Get-SSMDocumentList `
    -Filters @($typeFilter,$ownerFilter)
```

- 要查看第三方拥有的所有软件包，请运行以下命令。

```
$typeFilter = New-Object
 Amazon.SimpleSystemsManagement.Model.DocumentKeyValuesFilter
$typeFilter.Key = "DocumentType"
$typeFilter.Values = "Package"

$ownerFilter = New-Object
 Amazon.SimpleSystemsManagement.Model.DocumentKeyValuesFilter
$ownerFilter.Key = "Owner"
$ownerFilter.Values = "ThirdParty"

Get-SSMDocumentList `
    -Filters @($typeFilter,$ownerFilter)
```

在 Distributor 中创建软件包

要创建软件包，请准备您的可安装软件或资产，每个操作系统平台一个文件。需要至少一个文件才能创建软件包。

有时，不同的平台可以使用同一个文件，但附加到软件包的所有文件都必须列在清单的 `Files` 部分中。如果您在控制台中使用简单工作流程创建软件包，则会为您生成清单。最多可以向一个文档附加 20 个文件。每个文件的最大大小为 1 GB。有关支持的平台的更多信息，请参阅 支持的软件包平台和架构。

创建软件包时，系统会创建新的 SSM 文档。借助此文档，您可以将该软件包部署到托管式节点。

仅出于演示目的，您可以从我们的网站下载示例软件包 ExplePackage.zip。在示例软件包中包括一个完整 JSON 清单和三个 .zip 文件，这些文件包含 PowerShell v7.0.0 的安装程序。安装和卸载脚本不包含有效命令。虽然您必须在 Advanced (高级) 工作流中将每个软件可安装文件和脚本压缩为 .zip 文件以创建软件包，但您不会在 Simple (简单) 工作流程中压缩可安装资产。

主题

- 使用简单工作流创建软件包
- 使用高级工作流创建软件包

使用简单工作流创建软件包

本节介绍如何通过在 Distributor 控制台中选择简单软件包创建工作流在 Distributor 中创建软件包。Distributor 是 AWS Systems Manager 中的一项工具。要创建软件包，请准备您的可安装资产，每个操作系统平台一个文件。需要至少一个文件才能创建软件包。简单软件包创建过程为您生成安装和卸载脚本、文件哈希以及 JSON 格式的清单。简单工作流程处理上传和压缩可安装文件以及创建新软件包和关联的 SSM 文档的过程。有关支持的平台的更多信息，请参阅 支持的软件包平台和架构。

在使用简单方法创建软件包时，Distributor 将为您创建 `install` 和 `uninstall` 脚本。但是，当您为就地更新创建软件包时，您必须在 Update script (更新脚本) 选项卡上提供自己的 `update` 脚本内容。在为 `update` 脚本添加输入命令时，Distributor 将该脚本与 `install` 和 `uninstall` 脚本一起包含在它创建的 .zip 包中。

> **Note**
>
> 使用 `In-place` 更新选项，可将新文件或更新的文件添加到现有的软件包安装中，而无需使关联的应用程序离线。

若要使用简单工作流创建软件包

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Distributor。
3. 在 Distributor 主页上，选择 Create package (创建软件包)，然后选择简单。
4. 在 Create package (创建软件包) 页面上，输入软件包的名称。软件包名称可以包含字母、数字、句点、短划线和下划线。名称应足够通用，以适用于所有版本的软件包附件，但又足够具体，可用于识别软件包的用途。

5. （可选）对于 Version name (版本名称)，请输入版本名称。版本名称最多可以包含 512 个字符，并且不能包含特殊字符。
6. 对于 Location (位置)，请使用存储桶名称和前缀或存储桶 URL 选择一个存储桶。
7. 对于 Upload software（上载软件），选择 Add software（添加软件），然后导航到具有 `.rpm`、`.msi` 或 `.deb` 扩展名的可安装软件文件。如果文件名包含空格，则上载将失败。您可以在单个操作中上传多个软件文件。
8. 对于目标平台，请验证为每个可安装文件显示的目标操作系统平台是否正确。如果显示的操作系统不正确，请从下拉列表中选择正确的操作系统。

   对于简单软件包创建工作流程中，由于您仅将每个可安装文件上传一次，因此，需要使用额外的步骤以指示 Distributor 在多个操作系统上将单个文件指定为目标。例如，如果您上传名为 `Logtool_v1.1.1.rpm` 的可安装软件文件，则必须更改简单工作流程中的某些默认值，以同时在 Amazon Linux 和 Ubuntu 操作系统上将相同的软件指定为目标。在针对多个平台时，请执行以下操作之一。

   - 改用高级工作流程，在开始之前将每个可安装文件压缩为 .zip 文件，然后手动编写清单，以便在多个操作系统平台或版本中将一个可安装文件指定为目标。有关更多信息，请参阅 使用高级工作流创建软件包。
   - 在简单工作流程中手动编辑清单文件，以便在多个操作系统平台或版本中将 .zip 文件指定为目标。有关如何执行此操作的更多信息，请参阅 步骤 2：创建 JSON 软件包清单 中步骤 4 的末尾。
9. 对于 Platform version (平台版本)，验证并确保所显示的操作系统平台版本是 **_any**（主要发行版本后跟一个通配符 (7.*)），或您希望软件适用于的确切操作系统发行版本。有关指定操作系统平台版本的更多信息，请参阅 步骤 2：创建 JSON 软件包清单 中的步骤 4。
10. 对于 Architecture (架构)，请从下拉列表中为每个可安装文件选择正确的处理器架构。有关支持的处理器架构的更多信息，请参阅 支持的软件包平台和架构。
11. （可选）展开 Scripts (脚本)，然后查看 Distributor 为可安装软件生成的脚本。
12. （可选）要提供用于就地更新的更新脚本，请展开 Scripts（脚本），选择 Update script（更新脚本）选项卡，然后输入更新脚本命令。

    Systems Manager 不会代表您生成更新脚本。
13. 要添加更多可安装的软件文件，请选择 Add software (添加软件)。否则，转到下一步。
14. （可选）展开清单，然后审核 Distributor 为您的可安装软件包生成的 JSON 清单。如果您在开始此过程后更改了有关软件的任何信息（例如平台版本或目标平台），请选择 Generate manifest (生成清单) 以显示更新的软件包清单。

如果要将可在多个操作系统上安装的软件指定为目标，则可以手动编辑清单，如步骤 8 中所述。有关编辑该清单的更多信息，请参阅 步骤 2：创建 JSON 软件包清单。

15. 选择 Create package (创建软件包)。

等待 Distributor 完成上传软件和创建软件包。 Distributor 显示每个可安装文件的上传状态。根据要添加的软件包的数量和大小，这可能需要几分钟时间。Distributor 会自动将您重定向到新软件包的 Package details (软件包详细信息) 页面，但您可以选择在上载软件之后自行打开此页面。Package details (软件包详细信息) 页面不会显示有关软件包的所有信息，直到 Distributor 完成软件包创建过程为止。要停止上传和软件包创建过程，请选择取消。

如果 Distributor 无法上载任何软件可安装文件，它会显示 Upload failed (上载失败) 消息。要重试上传，请选择 Retry upload (重试上传)。有关如何排除软件包创建失败的更多信息，请参阅 AWS Systems ManagerDistributor 故障排除。

使用高级工作流创建软件包

在本节中，了解高级用户在将压缩有安装和卸载脚本的可安装资产以及 JSON 清单文件上载到 Amazon S3 存储桶后，如何能在 Distributor 中创建软件包。

要创建软件包，请准备可安装资产的 .zip 文件（每个操作系统平台一个 .zip 文件）。需要具有至少一个 .zip 文件才能创建软件包。接下来，创建 JSON 清单。清单包含指向软件包代码文件的指针。在将所需的代码文件添加到一个文件夹或目录，并为清单填充正确的值后，将软件包上载到 Amazon S3 存储桶。

您可以从我们的网站下载示例软件包 ExamplePackage.zip。在示例软件包中包含一个完整 JSON 清单和三个 .zip 文件。

主题

- 步骤 1：创建 ZIP 文件
- 步骤 2：创建 JSON 软件包清单
- 步骤 3：将软件包和清单上载到 Amazon S3 存储桶
- 步骤 4：将软件包添加到 Distributor

步骤 1：创建 ZIP 文件

软件包至少包含软件或可安装资产的一个 .zip 文件。对于要支持的每个操作系统，软件包包含一个 .zip 文件，除非可以在多个操作系统上安装一个 .zip 文件。例如，Red Hat Enterprise Linux 和 Amazon

Linux 实例通常可以运行相同的 .RPM 可执行文件，因此，只需在软件包中附加一个 .zip 文件以支持这两种操作系统。

所需的文件

需要在每个 .zip 文件中包含以下项目：

- install 和 uninstall 脚本。基于 Windows Server 的托管式节点需要使用 PowerShell 脚本（名为 `install.ps1` 和 `uninstall.ps1` 的脚本）。基于 Linux 的托管式节点需要使用 Shell 脚本（名为 `install.sh` 和 `uninstall.sh` 的脚本）。SSM Agent 运行 install 和 uninstall 脚本中的指令。

  例如，安装脚本可能会运行安装程序（例如 .rpm 或 .msi），它们可能会复制文件，或者它们可能会设置配置。
- 可执行文件、安装程序软件包（.rpm、.deb、.msi 等）、其他脚本或配置文件。

可选的文件

以下项目在每个 .zip 文件中是可选的：

- update 脚本。通过提供更新脚本，您可以使用 `In-place update` 选项安装软件包。如果要将新文件或更新的文件添加到现有的软件包安装中，`In-place update` 选项不会在执行更新时使软件包应用程序离线。基于 Windows Server 的托管式节点需要使用 PowerShell 脚本（名为 `update.ps1` 的脚本）。基于 Linux 的托管式节点需要使用 shell 脚本（名为 `update.sh` 的脚本）。SSM Agent 运行 update 脚本中的指令。

有关安装或更新软件包的更多信息，请参阅 安装或更新 Distributor 软件包。

有关 .zip 文件的示例（包括示例 install 和 uninstall 脚本），请下载示例软件包 ExamplePackage.zip。

步骤 2：创建 JSON 软件包清单

准备并压缩可安装文件后，需要创建一个 JSON 清单。模板如下。本节中的过程将对清单模板的各个部分进行说明。您可以使用 JSON 编辑器在单独的文件中创建此清单。或者，也可以在创建软件包时在 AWS Systems Manager 控制台中创建清单。

```
{
  "schemaVersion": "2.0",
  "version": "your-version",
  "publisher": "optional-publisher-name",
```

```
  "packages": {
    "platform": {
      "platform-version": {
        "architecture": {
          "file": ".zip-file-name-1.zip"
        }
      }
    },
    "another-platform": {
      "platform-version": {
        "architecture": {
          "file": ".zip-file-name-2.zip"
        }
      }
    },
    "another-platform": {
      "platform-version": {
        "architecture": {
          "file": ".zip-file-name-3.zip"
        }
      }
    }
  },
  "files": {
    ".zip-file-name-1.zip": {
      "checksums": {
        "sha256": "checksum"
      }
    },
    ".zip-file-name-2.zip": {
      "checksums": {
        "sha256": "checksum"
      }
    }
  }
}
```

创建 JSON 软件包清单

1. 将架构版本添加到清单。在此版本中，架构版本始终为 `2.0`。

   ```
   { "schemaVersion": "2.0",
   ```

2. 将用户定义的软件包版本添加到清单。这也是您在将软件包添加到 Distributor 时指定的 Version name (版本名称) 的值。添加软件包时，它将成为 Distributor 创建的 AWS Systems Manager 文档的一部分。您还可以在 AWS-ConfigureAWSPackage 文档中将此值作为输入提供，以安装除最新版本以外的软件包版本。version 值可以包含字母、数字、下划线、连字符和句点，最大长度为 128 个字符。建议使用易读的软件包版本，以便您和其他管理员在部署时更轻松地指定确切的软件包版本。示例如下：

```
"version": "1.0.1",
```

3. （可选）添加发布者名称。示例如下：

```
"publisher": "MyOrganization",
```

4. 添加软件包。"packages" 部分描述软件包中的 .zip 文件支持的平台、发行版本和架构。有关更多信息，请参阅 支持的软件包平台和架构。

   platform-version 可以是通配符值 _any。可以使用它指示 .zip 文件支持任何平台版本。您还可以指定一个主要发行版本，后跟通配符，以使所有次要版本受到支持，例如 7.*。如果您选择为某一特定操作系统版本指定 *platform-version* 值，请确保它与您设为目标的操作系统 AMI 的确切发行版本匹配。以下是用于获取正确的操作系统值的推荐资源。

   - 在基于 Windows Server 的托管式节点上，发行版本可用作 Windows Management Instrumentation (WMI) 数据。您可以运行以下命令提示符命令来获取版本信息，然后解析 version 的结果。

     ```
     wmic OS get /format:list
     ```

   - 在基于 Linux 的托管式节点上，可以通过先扫描操作系统版本（下面的命令）来获取版本。然后查找 VERSION_ID 的值。

     ```
     cat /etc/os-release
     ```

     如果这未返回您需要的结果，请运行下面的命令，以从 /etc/lsb-release 文件获取 LSB 版本信息，并查找 DISTRIB_RELEASE 的值。

     ```
     lsb_release -a
     ```

如果上述方法都不奏效，您通常可以根据分发软件包查找版本。例如，在 Debian Server 上，您可以扫描 /etc/debian_version 文件；在 Red Hat Enterprise Linux 上，则可以扫描 /etc/redhat-release 文件。

```
hostnamectl
```

```
"packages": {
    "platform": {
      "platform-version": {
        "architecture": {
          "file": ".zip-file-name-1.zip"
        }
      }
    },
    "another-platform": {
      "platform-version": {
        "architecture": {
          "file": ".zip-file-name-2.zip"
        }
      }
    },
    "another-platform": {
      "platform-version": {
        "architecture": {
          "file": ".zip-file-name-3.zip"
        }
      }
    }
  }
```

示例如下：在该示例中，操作系统平台为 amazon，支持的发行版本为 2016.09，架构为 x86_64，支持该平台的 .zip 文件为 test.zip。

```
{
    "amazon": {
        "2016.09": {
            "x86_64": {
                "file": "test.zip"
            }
        }
```

```
    }
},
```

您可以添加通配符值 _any 来指示此软件包支持父元素的所有版本。例如，要指示 Amazon Linux 的任何发行版本都支持此软件包，您的软件包语句应与以下内容相似。您可以在版本或架构级别使用 _any 通配符来支持平台的所有版本、某个版本的所有架构或某个平台的所有版本和所有架构。

```
{
    "amazon": {
        "_any": {
            "x86_64": {
                "file": "test.zip"
            }
        }
    }
},
```

以下示例添加 _any 来表明 Amazon Linux 2016.09 的所有架构都支持第一个软件包 data1.zip。Amazon Linux 的所有版本都支持第二个软件包 data2.zip，但此软件包仅适用于采用 x86_64 架构的托管式节点。2016.09 和 _any 版本都是 amazon 下的条目。具有一个平台 (Amazon Linux)，但支持的版本、架构和关联的 .zip 文件不同。

```
{
    "amazon": {
        "2016.09": {
            "_any": {
                "file": "data1.zip"
            }
        },
        "_any": {
            "x86_64": {
                "file": "data2.zip"
            }
        }
    }
}
```

如果 .zip 文件支持多个平台，您可以在清单的 "packages" 部分中多次引用该 .zip 文件。例如，如果您有一个同时支持 Red Hat Enterprise Linux 7.x 版本和 Amazon Linux 的 .zip 文件，您可以在 "packages" 部分中包含两个指向相同 .zip 文件的条目，如以下示例中所示。

```
{
    "amazon": {
        "2018.03": {
            "x86_64": {
                "file": "test.zip"
            }
        }
    },
    "redhat": {
        "7.*": {
            "x86_64": {
                "file": "test.zip"
            }
        }
    }
},
```

5. 添加属于步骤 4 中的软件包的 .zip 文件列表。每个文件条目都需要文件名和 sha256 哈希值校验和。清单中的校验和值必须与压缩资产中的 sha256 哈希值匹配，以防软件包安装失败。

   要从可安装资产获取确切的校验和，可以运行以下命令。在 Linux 上，运行 `shasum -a 256 file-name.zip` 或 `openssl dgst -sha256 file-name.zip`。在 Windows 上，在 PowerShell 中运行 `Get-FileHash -Path path-to-.zip-file` cmdlet。

   清单的 `"files"` 部分包含对软件包中的每个 .zip 文件的引用。

```
"files": {
        "test-agent-x86.deb.zip": {
            "checksums": {
                "sha256":
 "EXAMPLE2706223c7616ca9fb28863a233b38e5a23a8c326bb4ae241dcEXAMPLE"
            }
        },
        "test-agent-x86_64.deb.zip": {
            "checksums": {
                "sha256":
 "EXAMPLE572a745844618c491045f25ee6aae8a66307ea9bff0e9d1052EXAMPLE"
            }
        },
        "test-agent-x86_64.nano.zip": {
            "checksums": {
```

```
                "sha256":
"EXAMPLE63ccb86e830b63dfef46995af6b32b3c52ce72241b5e80c995EXAMPLE"
            }
        },
        "test-agent-rhel5-x86.nano.zip": {
            "checksums": {
                "sha256":
"EXAMPLE13df60aa3219bf117638167e5bae0a55467e947a363fff0a51EXAMPLE"
            }
        },
        "test-agent-x86.msi.zip": {
            "checksums": {
                "sha256":
"EXAMPLE12a4abb10315aa6b8a7384cc9b5ca8ad8e9ced8ef1bf0e5478EXAMPLE"
            }
        },
        "test-agent-x86_64.msi.zip": {
            "checksums": {
                "sha256":
"EXAMPLE63ccb86e830b63dfef46995af6b32b3c52ce72241b5e80c995EXAMPLE"
            }
        },
        "test-agent-rhel5-x86.rpm.zip": {
            "checksums": {
                "sha256":
"EXAMPLE13df60aa3219bf117638167e5bae0a55467e947a363fff0a51EXAMPLE"
            }
        },
        "test-agent-rhel5-x86_64.rpm.zip": {
            "checksums": {
                "sha256":
"EXAMPLE7ce8a2c471a23b5c90761a180fd157ec0469e12ed38a7094d1EXAMPLE"
            }
        }
    }
```

6. 添加软件包信息后，保存并关闭清单文件。

下面是一个已完成的清单的示例。在该示例中，您具有一个 .zip 文件 NewPackage_LINUX.zip，它支持多个平台，但仅在 "files" 部分中引用一次。

```
{
    "schemaVersion": "2.0",
```

```
    "version": "1.7.1",
    "publisher": "Amazon Web Services",
    "packages": {
        "windows": {
            "_any": {
                "x86_64": {
                    "file": "NewPackage_WINDOWS.zip"
                }
            }
        },
        "amazon": {
            "_any": {
                "x86_64": {
                    "file": "NewPackage_LINUX.zip"
                }
            }
        },
        "ubuntu": {
            "_any": {
                "x86_64": {
                    "file": "NewPackage_LINUX.zip"
                }
            }
        }
    },
    "files": {
        "NewPackage_WINDOWS.zip": {
            "checksums": {
                "sha256":
 "EXAMPLEc2c706013cf8c68163459678f7f6daa9489cd3f91d52799331EXAMPLE"
            }
        },
        "NewPackage_LINUX.zip": {
            "checksums": {
                "sha256":
 "EXAMPLE2b8b9ed71e86f39f5946e837df0d38aacdd38955b4b18ffa6fEXAMPLE"
            }
        }
    }
}
```

软件包示例

您可以从我们的网站下载示例软件包 ExamplePackage.zip。在示例软件包中包含一个完整 JSON 清单和三个 .zip 文件。

步骤 3：将软件包和清单上载到 Amazon S3 存储桶

将所有 .zip 文件复制或移动到一个文件夹或目录以准备软件包。有效的软件包需要具有在步骤 2：创建 JSON 软件包清单中创建的清单以及在清单文件列表中指定的所有 .zip 文件。

将软件包和清单上载到 Amazon S3

1. 将清单中指定的所有 .zip 存档文件复制或移动到一个文件夹或目录中。不要压缩您要将 .zip 归档文件和清单文件移动到的文件夹或目录。
2. 创建存储桶或选择现有的存储桶。有关更多信息，请参阅 Amazon Simple Storage Service 入门指南中的创建存储桶。有关如何运行 AWS CLI 命令创建存储桶的更多信息，请参阅 AWS CLI 命令参考中的 mb。
3. 将此文件夹或目录上载到存储桶。有关更多信息，请参阅 Amazon Simple Storage Service 入门指南中的向存储桶添加对象。如果打算将 JSON 清单粘贴到 AWS Systems Manager 控制台，请不要上载该清单。有关如何运行 AWS CLI 命令以将文件上载到存储桶的更多信息，请参阅 AWS CLI 命令参考中的 mv。
4. 在存储桶的主页上，选择您上载的文件夹或目录。如果您将文件上传到存储桶中的子文件夹，请务必记下子文件夹（也称为前缀）。您需要此前缀以将软件包添加到 Distributor。

步骤 4：将软件包添加到 Distributor

您可以使用 AWS Systems Manager 控制台、AWS 命令行工具（AWS CLI 和 AWS Tools for PowerShell）或 AWS 开发工具包将新软件包添加到 Distributor。添加软件包时，您将添加一个新的 SSM 文档。使用此文档，可以将该软件包部署到托管式节点。

主题

- 使用控制台添加软件包
- 使用 AWS CLI 添加软件包

使用控制台添加软件包

可使用 AWS Systems Manager 控制台创建软件包。准备好您在步骤 3：将软件包和清单上载到 Amazon S3 存储桶中将软件包上传到的存储桶的名称。

将软件包添加到 Distributor（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Distributor。
3. 在 Distributor 主页上，选择 Create package (创建软件包)，然后选择高级。
4. 在 Create package (创建软件包) 页面上，输入软件包的名称。软件包名称可以包含字母、数字、句点、短划线和下划线。名称应足够通用，以适用于所有版本的软件包附件，但又足够具体，可用于识别软件包的用途。
5. 对于 Version name (版本名称)，请输入清单文件中的 `version` 条目的确切值。
6. 对于 S3 bucket name (S3 存储桶名称)，请选择在the section called “步骤 3：将软件包和清单上载到 Amazon S3 存储桶”中将 .zip 文件和清单上传到的存储桶的名称。
7. 对于 S3 key prefix (S3 键前缀)，请输入存储 .zip 文件和清单的存储桶子文件夹。
8. 对于 Manifest (清单)，请选择 Extract from package (从软件包提取) 以使用随 .zip 文件上载到 Amazon S3 存储桶的清单。

   （可选）如果未将 JSON 清单上载到存储 .zip 文件的 Amazon S3 存储桶，请选择 New manifest (新建清单)。您可以在 JSON 编辑器字段中编写或粘贴整个清单。有关如何创建 JSON 清单的更多信息，请参阅 步骤 2：创建 JSON 软件包清单。
9. 当您处理完清单后，选择 Create package (创建软件包)。
10. 等待 Distributor 从 .zip 文件和清单中创建软件包。根据要添加的软件包的数量和大小，这可能需要几分钟时间。Distributor 会自动将您重定向到新软件包的 Package details (软件包详细信息) 页面，但您可以选择在上传软件之后自行打开此页面。Package details (软件包详细信息) 页面不会显示有关软件包的所有信息，直到 Distributor 完成软件包创建过程为止。要停止上传和软件包创建过程，请选择取消。

使用 AWS CLI 添加软件包

可使用 AWS CLI 创建软件包。准备好您在步骤 3：将软件包和清单上载到 Amazon S3 存储桶中将软件包上传到的存储桶的 URL。

若要使用 AWS CLI 将软件包添加到 Amazon S3

1. 要使用 AWS CLI 创建软件包，请运行以下命令，将 *`package-name`* 替换为您的软件包名称，将 *`path-to-manifest-file`* 替换为 JSON 清单文件的文件路径。amzn-s3-demo-bucket 是存储

整个软件包的 Amazon S3 存储桶的 URL。在 Distributor 中运行 create-document 命令时，为 `--document-type` 指定 `Package` 值。

如果未将清单文件添加到 Amazon S3 存储桶，则 `--content` 参数值将是 JSON 清单文件的文件路径。

```
aws ssm create-document \
    --name "package-name" \
    --content file://path-to-manifest-file \
    --attachments Key="SourceUrl",Values="amzn-s3-demo-bucket" \
    --version-name version-value-from-manifest \
    --document-type Package
```

示例如下：

```
aws ssm create-document \
    --name "ExamplePackage" \
    --content file://path-to-manifest-file \
    --attachments Key="SourceUrl",Values="https://s3.amazonaws.com/amzn-s3-demo-
bucket/ExamplePackage" \
    --version-name 1.0.1 \
    --document-type Package
```

2. 通过运行以下命令验证软件包是否已添加并显示软件包清单，将 *package-name* 替换为您的软件包名称。要获取文档的特定版本（不同于软件包的版本），可以添加 `--document-version` 参数。

```
aws ssm get-document \
    --name "package-name"
```

有关可以与 create-document 命令结合使用的其他选项的信息，请参阅 AWS CLI 命令参考 AWS Systems Manager 一节中的 create-document。有关可以与 get-document 命令配合使用的其他选项的信息，请参阅 get-document。

在控制台中编辑 Distributor 软件包权限

在将软件包添加到 Distributor（AWS Systems Manager 中的一项工具）后，即可在 Systems Manager 控制台中编辑该软件包的权限。您可以将其他 AWS 账户添加到软件包的权限中。程序包只能与同一个 AWS 区域中的其他账户共享。不支持跨区域共享。预设情况下，软件包设置为 Private (私

有)，表示只有有权访问软件包创建者的 AWS 账户的用户才能查看该软件包的软件包信息，以及更新或删除该软件包。如果接受 Private (私有) 权限，则可以跳过此过程。

> **Note**
>
> 您可以更新与 20 个或以下账户共享的软件包的权限。

若要在控制台中编辑软件包权限

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Distributor。
3. 在 Packages (软件包) 页面上，选择要编辑其权限的软件包。
4. 在 Package details (软件包详细信息) 选项卡上，选择 Edit permissions (编辑权限) 更改权限。
5. 对于 Edit permissions (编辑权限)，请选择 Shared with specific accounts (与特定账户共享)。
6. 在 Shared with specific accounts (与特定账户共享) 下，添加 AWS 账户 账号，每次添加一个。完成后，选择 保存。

在控制台中编辑 Distributor 软件包标签

将软件包添加到 Distributor（AWS Systems Manager 中的一项工具）后，即可在 Systems Manager 控制台中编辑该软件包的标签。这些标签将应用于该软件包，并且不与要部署该软件包的托管式节点上的标签相关联。标签是区分大小写的键值对，可帮助您按照组织相关标准对软件包进行分组和筛选。如果不希望添加标签，则要准备好安装软件包或添加新版本。

若要在控制台中编辑软件包标签

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Distributor。
3. 在 Packages (软件包) 页面上，选择要编辑其标签的软件包。
4. 在 Package details (软件包详细信息) 选项卡上的 Tags (标签) 中，选择 Edit (编辑)。
5. 对于 Add tags (添加标签)，请输入一个标签键或标签键值对，然后选择 Add (添加)。如果要添加更多标签，请重复此操作。要删除标签，请选择窗口底部标签上的 X。

6. 为软件包添加完标签后，选择 Save (保存)。

为 Distributor 软件包添加版本

要添加软件包版本，您需要创建软件包，然后使用 Distributor 通过向旧版本已存在的 AWS Systems Manager（SSM）文档中添加条目来添加软件包版本。Distributor 是 AWS Systems Manager 中的一项工具。为了节省时间，可以更新旧版本包的清单，更改清单中 `version` 条目的值（例如，从 `Test_1.0` 更改为 `Test_2.0`），并将其保存为新版本的清单。Distributor 控制台中的简单 Add version (添加版本) 工作流将为您更新清单文件。

新的软件包版本可以：

- 替换附加到当前版本的至少一个可安装文件。
- 添加新的可安装文件以支持其他平台。
- 删除文件以停止对特定平台的支持。

较新的版本可以使用相同的 Amazon Simple Storage Service (Amazon S3) 存储桶，但必须具有末尾显示不同文件名的 URL。您可以使用 Systems Manager 控制台或 AWS Command Line Interface (AWS CLI) 来添加新版本。将具有确切名称的可安装文件上载为 Amazon S3 存储桶中的现有可安装文件将覆盖现有文件。没有可安装的文件从旧版本复制到新版本；您必须从旧版本上传可安装文件才能使它们成为新版本的一部分。在 Distributor 完成创建新软件包版本后，您可以删除 Amazon S3 存储桶或调整其用途，因为 Distributor 在版本控制过程中会将软件复制到内部 Systems Manager 存储桶。

> **Note**
>
> 每个软件包限制为最多 25 个版本。您可以删除不再需要的版本。

主题

- 使用控制台添加软件包版本
- 使用 AWS CLI 添加软件包版本

使用控制台添加软件包版本

在执行以下步骤前，请按照在 Distributor 中创建软件包中的说明执行操作，为此版本创建一个新的软件包。然后，使用 Systems Manager 控制台将新的软件包版本添加到 Distributor。

使用简单工作流添加软件包版本

要使用 Simple (简单) 工作流程添加软件包版本，请准备更新的可安装文件或添加可安装文件，以支持更多平台和架构。然后，使用 Distributor 上传新的和更新的可安装文件并添加软件包版本。Distributor 控制台中的简化 Add version (添加版本) 工作流程将为您更新清单文件和关联的 SSM 文档。

若要使用简单工作流添加软件包版本

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Distributor。
3. 在 Distributor 主页上，选择要为其添加其他版本的软件包。
4. 在 Add version (添加版本) 页面上，选择 Simple (简单)。
5. 对于 Version name (版本名称)，请输入版本名称。新版本的版本名称必须与旧版本名称不同。版本名称最多可以包含 512 个字符，并且不能包含特殊字符。
6. 对于 S3 存储桶名称，从列表中选择现有的 S3 存储桶。这可以是用于存储旧版本的可安装文件的同一个存储桶，但可安装文件名必须不同，以避免覆盖存储桶中的现有可安装文件。
7. 对于 S3 key prefix (S3 键前缀)，请输入存储可安装资产的存储桶子文件夹。
8. 对于 Upload software (上载软件)，请导航到要附加到新版本的可安装软件文件。现有版本中的可安装文件不会自动复制到新版本；如果您希望任何相同的可安装文件成为新版本的一部分，则必须上载旧版软件包中的任何可安装文件。您可以在单个操作中上传多个软件文件。
9. 对于目标平台，请验证为每个可安装文件显示的目标操作系统平台是否正确。如果显示的操作系统不正确，请从下拉列表中选择正确的操作系统。

   在简单版本控制工作流程中，由于您只上传每个可安装文件一次，因此需要额外的步骤在多个操作系统上将单个文件指定为目标。例如，如果您上传名为 `Logtool_v1.1.1.rpm` 的可安装软件文件，则必须更改简单工作流程中的某些默认值，以指示 Distributor 同时在 Amazon Linux 和 Ubuntu 操作系统上将相同的软件指定为目标。您可以执行以下操作之一来解决此限制。

   - 改用高级版本控制工作流程，在开始之前将每个可安装文件压缩为 .zip 文件，然后手动编写清单，以便在多个操作系统平台或版本中将一个可安装文件指定为目标。有关更多信息，请参阅使用高级工作流添加软件包版本。
   - 在简单工作流程中手动编辑清单文件，以便在多个操作系统平台或版本中将 .zip 文件指定为目标。有关如何执行此操作的更多信息，请参阅 步骤 2：创建 JSON 软件包清单 中步骤 4 的末尾。

10. 对于 Platform version (平台版本)，验证并确保所显示的操作系统平台版本是 **_any**（主要发行版本后跟一个通配符 (7.*)），或您希望软件适用于的确切操作系统发行版本。有关指定平台版本的更多信息，请参阅 步骤 2：创建 JSON 软件包清单 中的步骤 4。
11. 对于架构，请从下拉列表中为每个可安装文件选择正确的处理器架构。有关支持的架构的更多信息，请参阅 支持的软件包平台和架构。
12. （可选）展开 Scripts (脚本)，然后查看 Distributor 为可安装软件生成的安装和卸载脚本。
13. 要向新版本添加更多可安装的软件文件，请选择 Add software (添加软件)。否则，转到下一步。
14. （可选）展开清单，然后审核 Distributor 为您的可安装软件包生成的 JSON 清单。如果您在开始此过程后更改了有关可安装软件的任何信息（例如平台版本或目标平台），请选择生成清单以显示更新的软件包清单。

    如果要将可在多个操作系统上安装的软件指定为目标，则可以手动编辑清单，如步骤 9 中所述。有关编辑该清单的更多信息，请参阅 步骤 2：创建 JSON 软件包清单。
15. 完成添加软件和审核目标平台、版本和架构数据时，请选择 Add version (添加版本)。
16. 等待 Distributor 完成上传软件和创建新的软件包版本。Distributor 显示每个可安装文件的上传状态。根据要添加的软件包的数量和大小，这可能需要几分钟时间。Distributor 会自动将您重定向到软件包的 Package details (软件包详细信息) 页面，但您可以选择在上传软件之后自行打开此页面。Package details (软件包详细信息) 页面不会显示有关软件包的所有信息，直到 Distributor 完成创建新的软件包版本为止。要停止上传和软件包版本创建，请选择 Stop upload (停止上传)。
17. 如果 Distributor 无法上载任何软件可安装文件，它会显示 Upload failed (上载失败) 消息。要重试上传，请选择 Retry upload (重试上传)。有关如何排除软件包版本创建失败的更多信息，请参阅 AWS Systems ManagerDistributor 故障排除。
18. 在 Distributor 完成创建新的软件包版本后，在软件包的 Details (详细信息) 页面的 Versions (版本) 选项卡上，在可用软件包版本列表中查看新版本。通过以下方式设置软件包的默认版本：选择一个版本，然后选择 Set default version (设置默认版本)。

    如果不设置默认版本，则最新的软件包版本即为默认版本。

使用高级工作流添加软件包版本

要添加软件包版本，您需要创建软件包，然后使用 Distributor 通过向旧版本存在的 SSM 文档中添加条目来添加软件包版本。为了节省时间，可以更新旧版本包的清单，更改清单中 `version` 条目的值（例如，从 `Test_1.0` 更改为 `Test_2.0`），并将其保存为新版本的清单。您必须具有更新的清单才能使用高级工作流程添加新的软件包版本。

若要使用高级工作流添加软件包版本

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Distributor。
3. 在 Distributor 主页上，选择要向其添加其他版本的软件包，然后选择 Add version (添加版本)。
4. 对于 Version name (版本名称)，请输入位于清单文件的 `version` 条目中的确切值。
5. 对于 S3 存储桶名称，从列表中选择现有的 S3 存储桶。这可以是用于存储旧版本的可安装文件的同一个存储桶，但可安装文件名必须不同，以避免覆盖存储桶中的现有可安装文件。
6. 对于 S3 key prefix (S3 键前缀)，请输入存储可安装资产的存储桶子文件夹。
7. 对于 Manifest (清单)，请选择 Extract from package (从软件包提取) 以使用随 .zip 文件上传到 S3 存储桶的清单。

   （可选）如果未将修订的 JSON 清单上载到存储 .zip 文件的 Amazon S3 存储桶，请选择 New manifest (新建清单)。您可以在 JSON 编辑器字段中编写或粘贴整个清单。有关如何创建 JSON 清单的更多信息，请参阅 步骤 2：创建 JSON 软件包清单。
8. 当您完成处理清单后，选择 Add package version (添加软件包版本)。
9. 在软件包的 Details (详细信息) 页面的 Versions (版本) 选项卡上，在可用软件包版本列表中查看新版本。通过以下方式设置软件包的默认版本：选择一个版本，然后选择 Set default version (设置默认版本)。

   如果不设置默认版本，则最新的软件包版本即为默认版本。

使用 AWS CLI 添加软件包版本

您可以使用 AWS CLI 将新的软件包版本添加到 Distributor。在运行以下命令前，必须创建新的软件包版本并将其上传到 S3，如本主题开头所述。

若要使用 AWS CLI 添加软件包版本

1. 运行下面的命令并附上新软件包版本的条目来编辑 AWS Systems Manager 文档。将 *`document-name`* 替换为您的文档的名称。将 *`amzn-s3-demo-bucket`* 替换为您在 步骤 3：将软件包和清单上载到 Amazon S3 存储桶 中复制的 JSON 清单的 URL。*`S3-bucket-URL-of-package`* 是存储整个软件包的 Amazon S3 存储桶的 URL。将 *`version-name-from-updated-manifest`* 替换为清单中 `version` 的值。将 `--document-version` 参数设置为 `$LATEST`，以使与此软件包版本关联的文档成为文档的最新版本。

```
aws ssm update-document \
    --name "document-name" \
    --content "S3-bucket-URL-to-manifest-file" \
    --attachments Key="SourceUrl",Values="amzn-s3-demo-bucket" \
    --version-name version-name-from-updated-manifest \
    --document-version $LATEST
```

示例如下：

```
aws ssm update-document \
    --name ExamplePackage \
    --content "https://s3.amazonaws.com/amzn-s3-demo-bucket/ExamplePackage/
manifest.json" \
    --attachments Key="SourceUrl",Values="https://s3.amazonaws.com/amzn-s3-demo-
bucket/ExamplePackage" \
    --version-name 1.1.1 \
    --document-version $LATEST
```

2. 运行以下命令来验证软件包已更新并显示包清单。将 *package-name* 替换为您的软件包名称，（可选）将 *document-version* 替换为更新的文档的版本号（不同于软件包的版本）。如果此软件包版本与文档的最新版本关联，则可以为可选的 `--document-version` 参数的值指定 `$LATEST`。

```
aws ssm get-document \
    --name "package-name" \
    --document-version "document-version"
```

有关可以与 update-document 命令结合使用的其他选项的信息，请参阅 AWS CLI 命令参考 AWS Systems Manager 一节中的 update-document。

安装或更新 Distributor 软件包

您可以使用 Distributor（AWS Systems Manager 中的一项工具）将软件包部署到 AWS Systems Manager 托管式节点。要部署软件包，请使用 AWS Management Console 或 AWS Command Line Interface (AWS CLI)。一条命令只能部署一个软件包的一个版本。您可以安装新软件包或就地更新现有的安装。您可以选择部署软件包的特定版本，也可以选择始终部署软件包的最新版本。建议使用State Manager（AWS Systems Manager 中的一项工具）来安装软件包。使用 State Manager 有助于确保您的托管式节点始终运行最新版本的软件包。

> **Important**
>
> 使用 Distributor 安装的软件包只能通过 Distributor 卸载。否则，Systems Manager 仍可能将应用程序注册为 `INSTALLED`，导致其他意外结果。

Preference	AWS Systems Manager 操作	更多信息
立即安装或更新软件包。	Run Command	• 使用控制台一次性安装或更新软件包 • 使用 AWS CLI 一次性安装软件包 • 使用 AWS CLI 一次性更新软件包
按计划安装或更新软件包，以便安装始终包含默认版本。	State Manager	• 使用控制台计划软件包安装或更新 • 使用 AWS CLI 计划软件包安装 • 使用 AWS CLI 计划软件包更新
在具有特定标签或标签集的新托管式节点上自动安装软件包。例如，在新实例上安装 Amazon CloudWatch 代理。	State Manager	执行此操作的一种方法是向新托管式节点应用标签，然后在 State Manager 关联中将这些标签指定为目标。State Manager 会自动将关联中的软件包安装到具有匹配标签的托管式节点上。请参阅了解 State Manager 关联中的目标和速率控制。

主题

- 使用控制台一次性安装或更新软件包
- 使用控制台计划软件包安装或更新

- 使用 AWS CLI 一次性安装软件包
- 使用 AWS CLI 一次性更新软件包
- 使用 AWS CLI 计划软件包安装
- 使用 AWS CLI 计划软件包更新

使用控制台一次性安装或更新软件包

您可以使用 AWS Systems Manager 控制台一次性安装或更新软件包。在配置一次性安装时，Distributor 使用 AWS Systems Manager Run Command（AWS Systems Manager 中的一项工具）来执行安装。

若要使用控制台一次性安装或更新软件包

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Distributor。
3. 在 Distributor 主页上，选择要安装的软件包。
4. 选择 Install one time (一次性安装)。

   此命令将打开 Run Command，并选中命令文档 `AWS-ConfigureAWSPackage` 和 Distributor 软件包。
5. 对于 Document version（文档版本），请选择要运行的 `AWS-ConfigureAWSPackage` 文档版本。
6. 对于操作，选择安装。
7. 对于 Installation type (安装类型)，请选择以下选项之一：

   - Uninstall and reinstall (卸载并重新安装)：完全卸载软件包，然后重新安装。在重新安装完成之前，应用程序不可用。
   - In-place update (就地更新)：根据您在 `update` 脚本中提供的指令，仅将新文件或更改的文件添加到现有安装中。应用程序在整个更新过程中保持可用。对于 AWS 已发布软件包，除 `AWSEC2Launch-Agent` 软件包外，不支持此选项。
8. 对于 Name (名称)，请确认输入了选定的软件包的名称。
9. （可选）对于 Version (版本)，请输入软件包的版本名称值。如果将此字段留空，Run Command 将安装在 Distributor 中选择的默认版本。

10. 在 Targets（目标）部分中，通过指定标签、手动选择实例或设备或指定资源组，选择要在其上运行此操作的托管式节点。

> **Note**
>
> 如果列表没有显示托管式节点，请参阅 排除托管式节点可用性的问题。

11. 对于 Other parameters（其他参数）：

- 对于 Comment（注释），请输入有关此命令的信息。
- 对于 Timeout (seconds) (超时 (秒))，请指定在整个命令执行失败之前系统等待的秒数。

12. 对于 Rate Control（速率控制）：

- 对于 Concurrency（并发），请指定要同时运行该命令的目标数或百分比。

> **Note**
>
> 如果您通过指定标签或资源组来选择目标，但不确定有多少个托管式节点已被设为目标，则可通过指定百分比来限制可同时运行该文档的目标的数量。

- 对于 Error threshold（错误阈值），请指定当命令在一定数量或百分比的托管式节点上失败后，何时在其他目标上停止运行该命令。例如，如果您指定三个错误，Systems Manager 将在收到第四个错误时停止发送该命令。仍在处理该命令的托管式节点也可能发送错误。

13. （可选）对于 输出选项，要将命令输出保存到文件，请选中 将命令输出写入 S3 存储桶 框。在输入框中输入存储桶和前缀（文件夹）名称。

> **Note**
>
> 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给实例的实例配置文件（适用于 EC2 实例）或 IAM 服务角色（混合激活的计算机）的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确保与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

14. 在 SNS 通知部分，如果需要发送有关命令执行状态的通知，请选中 Enable SNS notifications（启用 SNS 通知）复选框。

有关为 Run Command 配置 Amazon SNS 通知的更多信息，请参阅 使用 Amazon SNS 通知监控 Systems Manager 状态更改。

15. 在准备好安装软件包时，请选择 Run (运行)。
16. Command status (命令状态) 区域报告执行进度。如果命令仍在执行，请选择控制台左上角的刷新图标，直到 Overall status (总体状态) 或 Detailed status (详细状态) 列显示 Success (成功) 或 Failed (失败)。
17. 在 Targets and outputs（目标和输出）区域中，选择托管式节点名称旁边的按钮，然后选择 View output（查看输出）。

    命令输出页面将显示命令执行的结果。
18. （可选）如果您选择将命令输出写入到 Amazon S3 存储桶，请选择 Amazon S3 以查看输出日志数据。

使用控制台计划软件包安装或更新

您可以使用 AWS Systems Manager 控制台计划安装或更新软件包。在计划安装或更新软件包时，Distributor 使用 AWS Systems Manager State Manager 进行安装或更新。

若要使用控制台计划软件包安装

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Distributor。
3. 在 Distributor 主页上，选择要安装或更新的软件包。
4. 对于 Package (软件包)，请选择 Install on a schedule (按计划安装)。

   此命令将打开 State Manager 并跳转到为您创建的新关联。
5. 对于 Name (名称)，请输入名称（例如，**`Deploy-test-agent-package`**）。您可以自由选择，但我们建议您这样做。名称不得包含空格。
6. 在 Document (文档) 列表中，已选定文档名称 `AWS-ConfigureAWSPackage`。
7. 对于 Action (操作)，请确认选择了 Install (安装)。
8. 对于 Installation type (安装类型)，请选择以下选项之一：

   - Uninstall and reinstall (卸载并重新安装)：完全卸载软件包，然后重新安装。在重新安装完成之前，应用程序不可用。

- In-place update (就地更新)：根据您在 `update` 脚本中提供的指令，仅将新文件或更改的文件添加到现有安装中。应用程序在整个更新过程中保持可用。

9. 对于 Name (名称)，请确认输入了软件包的名称。
10. 对于 Version (版本)，如果要安装最新发布的版本以外的软件包版本，请输入版本标识符。
11. 对于 Targets (目标)，选择 Selecting all managed instances in this account (选择此账户中的所有托管实例)、Specifying tags (指定标签) 或 Manually Selecting Instance (手动选择实例)。如果使用标签将资源设置为目标，请在提供的字段中输入标签键和标签值。

    > **Note**
    >
    > 您可以通过选择 Selecting all managed instances in this account（选择此账户中的所有托管式实例）或 Manually Selecting Instance（手动选择实例）来选择托管式 AWS IoT Greengrass 核心设备。

12. 对于 Specify schedule (指定计划)，选择 On Schedule (按计划) 定期运行此关联，或选择 No Schedule (无计划) 只运行此关联一次。有关这些选项的详细信息，请参阅在 Systems Manager 中使用关联。。使用控件为关联创建 `cron` 或频率计划。
13. 选择创建关联。
14. 在 Association (关联) 页面上，选择您创建的关联旁边的按钮，然后选择 Apply association now (立即应用关联)。

    State Manager 在指定的目标中创建并立即运行关联。有关运行关联的结果的更多信息，请参阅本指南中的 在 Systems Manager 中使用关联。。

有关使用 Advanced options (高级选项)、Rate control (速率控制) 和 Output options (输出选项) 中的选项的更多信息，请参阅 在 Systems Manager 中使用关联。。

使用 AWS CLI 一次性安装软件包

您可以在 AWS CLI 中运行 send-command 一次性安装 Distributor 软件包。如果已安装软件包，在卸载软件包并在其位置安装新版本时，应用程序将脱机。

若要使用 AWS CLI 一次性安装软件包

- 在 AWS CLI 中运行以下命令。

```
aws ssm send-command \
```

```
    --document-name "AWS-ConfigureAWSPackage" \
    --instance-ids "instance-IDs" \
    --parameters '{"action":["Install"],"installationType":["Uninstall and
 reinstall"],"name":["package-name (in same account) or package-ARN (shared from
 different account)"]}'
```

> **Note**
>
> installationType 的默认行为是 Uninstall and reinstall。在安装完整软件包时，您可以从该命令中省略 "installationType":["Uninstall and reinstall"]。

示例如下：

```
aws ssm send-command \
    --document-name "AWS-ConfigureAWSPackage" \
    --instance-ids "i-00000000000000" \
    --parameters '{"action":["Install"],"installationType":["Uninstall and
 reinstall"],"name":["ExamplePackage"]}'
```

有关可以与 send-command 命令结合使用的其他选项的信息，请参阅 AWS CLI 命令参考 AWS Systems Manager 一节中的 send-command。

使用 AWS CLI 一次性更新软件包

您可以在 AWS CLI 中运行 send-command 以更新 Distributor 软件包，而无需将关联的应用程序脱机。仅替换软件包中的新文件或更新的文件。

若要使用 AWS CLI 一次性更新软件包

- 在 AWS CLI 中运行以下命令。

```
aws ssm send-command \
    --document-name "AWS-ConfigureAWSPackage" \
    --instance-ids "instance-IDs" \
    --parameters '{"action":["Install"],"installationType":["In-place
 update"],"name":["package-name (in same account) or package-ARN (shared from
 different account)"]}'
```

> **Note**
>
> 在添加新文件或更改的文件时，您必须在命令中包含 "installationType":["In-place update"]。

示例如下：

```
aws ssm send-command \
    --document-name "AWS-ConfigureAWSPackage" \
    --instance-ids "i-02573cafcfEXAMPLE" \
    --parameters '{"action":["Install"],"installationType":["In-place
 update"],"name":["ExamplePackage"]}'
```

有关可以与 send-command 命令结合使用的其他选项的信息，请参阅 AWS CLI 命令参考 AWS Systems Manager 一节中的 send-command。

使用 AWS CLI 计划软件包安装

您可以在 AWS CLI 中运行 create-association 按计划安装 Distributor 软件包。--name的值（文档名称）始终为 AWS-ConfigureAWSPackage。以下命令使用键 InstanceIds 指定目标托管式节点。如果已安装软件包，在卸载软件包并在其位置安装新版本时，应用程序将脱机。

```
aws ssm create-association \
    --name "AWS-ConfigureAWSPackage" \
    --parameters '{"action":["Install"],"installationType":["Uninstall and
 reinstall"],"name":["package-name (in same account) or package-ARN (shared from
 different account)"]}' \
    --targets [{\"Key\":\"InstanceIds\",\"Values\":[\"instance-ID1\",\"instance-
ID2\"]}]
```

> **Note**
>
> installationType 的默认行为是 Uninstall and reinstall。在安装完整软件包时，您可以从该命令中省略 "installationType":["Uninstall and reinstall"]。

示例如下：

```
aws ssm create-association \
    --name "AWS-ConfigureAWSPackage" \
    --parameters '{"action":["Install"],"installationType":["Uninstall and
 reinstall"],"name":["Test-ConfigureAWSPackage"]}' \
    --targets [{\"Key\":\"InstanceIds\",\"Values\":[\"i-02573cafcfEXAMPLE\",
\"i-0471e04240EXAMPLE\"]}]
```

有关可以与 create-association 命令结合使用的其他选项的信息，请参阅 AWS CLI 命令参考 AWS Systems Manager 一节中的 create-association。

使用 AWS CLI 计划软件包更新

您可以在 AWS CLI 中运行 create-association 以按计划更新 Distributor 软件包，而无需将关联的应用程序脱机。仅替换软件包中的新文件或更新的文件。`--name`的值（文档名称）始终为 `AWS-ConfigureAWSPackage`。以下命令使用键 `InstanceIds` 指定目标实例。

```
aws ssm create-association \
    --name "AWS-ConfigureAWSPackage" \
    --parameters '{"action":["Install"],"installationType":["In-place update"],"name":
["package-name (in same account) or package-ARN (shared from different account)"]}' \
    --targets [{\"Key\":\"InstanceIds\",\"Values\":[\"instance-ID1\",\"instance-
ID2\"]}]
```

> **Note**
>
> 在添加新文件或更改的文件时，您必须在命令中包含 `"installationType":["In-place update"]`。

示例如下：

```
aws ssm create-association \
    --name "AWS-ConfigureAWSPackage" \
    --parameters '{"action":["Install"],"installationType":["In-place update"],"name":
["Test-ConfigureAWSPackage"]}' \
    --targets [{\"Key\":\"InstanceIds\",\"Values\":[\"i-02573cafcfEXAMPLE\",
\"i-0471e04240EXAMPLE\"]}]
```

有关可以与 create-association 命令结合使用的其他选项的信息，请参阅 AWS CLI 命令参考 AWS Systems Manager 一节中的 create-association。

卸载 Distributor 软件包

您可以借助 Run Command，使用 AWS Management Console或 AWS Command Line Interface（AWS CLI）从 AWS Systems Manager 托管式节点中卸载 Distributor 软件包。Distributor 和 Run Command 都是 AWS Systems Manager 中的工具。在此版本中，一条命令只能卸载一个软件包的一个版本。您可以卸载特定版本或默认版本。

> **Important**
>
> 使用 Distributor 安装的软件包只能通过 Distributor 卸载。否则，Systems Manager 仍可能将应用程序注册为 `INSTALLED`，导致其他意外结果。

主题

- 使用控制台卸载软件包
- 使用 AWS CLI 卸载软件包

使用控制台卸载软件包

您可以在 Systems Manager 控制台中使用 Run Command 一次性卸载软件包。Distributor 使用 AWS Systems Manager Run Command 卸载软件包。

若要使用控制台卸载软件包

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Run Command。
3. 在 Run Command 主页上，选择 Run command (运行命令)。
4. 选择 `AWS-ConfigureAWSPackage` 命令文档。
5. 从 Action (操作) 中，选择 Uninstall (卸载)
6. 对于 Name (名称)，输入要卸载的软件包的名称。
7. 对于 Targets（目标），选择您希望如何将托管式节点设为目标。您可以指定目标共享的标签键和值。您还可以通过选择属性（例如 ID、平台和 SSM Agent 版本）来指定目标。
8. 您可以使用高级选项添加有关该操作的注释、更改 Rate control (速率控制) 中的 Concurrency (并发) 和 Error threshold (错误阈值) 的值、指定输出选项，或者配置 Amazon Simple Notification Service (Amazon SNS) 通知。有关更多信息，请参阅本指南中的从控制台运行命令。

9. 做好卸载软件包的准备后，选择 Run (运行)，然后选择 View results (查看结果)。
10. 在命令列表中，选择您运行的 AWS-ConfigureAWSPackage 命令。如果命令仍在执行中，则选择控制台右上角的“刷新”图标。
11. 如果 Status（状态）列显示 Success（成功）或 Failed（失败），则选择 Output（输出）选项卡。
12. 选择 View Output (查看输出)。命令输出页面将显示命令执行的结果。

使用 AWS CLI 卸载软件包

您可以使用 AWS CLI 通过 Run Command 从托管式节点中卸载 Distributor 软件包。

若要使用 AWS CLI 卸载软件包

- 在 AWS CLI 中运行以下命令。

```
aws ssm send-command \
    --document-name "AWS-ConfigureAWSPackage" \
    --instance-ids "instance-IDs" \
    --parameters '{"action":["Uninstall"],"name":["package-name (in same account)
 or package-ARN (shared from different account)"]}'
```

示例如下：

```
aws ssm send-command \
    --document-name "AWS-ConfigureAWSPackage" \
    --instance-ids "i-02573cafcfEXAMPLE" \
    --parameters '{"action":["Uninstall"],"name":["Test-ConfigureAWSPackage"]}'
```

有关可以与 send-command 命令结合使用的其他选项的信息，请参阅 AWS CLI 命令参考 AWS Systems Manager 一节中的 send-command。

删除 Distributor 软件包

本节介绍如何删除软件包。您不能删除某个版本的软件包，只能删除整个软件包。

使用控制台删除软件包

您可以使用 AWS Systems Manager 控制台从 Distributor（AWS Systems Manager 中的一项工具）中删除软件包或软件包版本。删除软件包将从 Distributor 删除此软件包的所有版本。

若要使用控制台删除软件包

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Distributor。
3. 在 Distributor 主页上，选择要删除的软件包。
4. 在软件包的详细信息页面上，选择 Delete package (删除软件包)。
5. 当系统提示确认删除时，选择 Delete package (删除软件包)。

使用控制台删除软件包版本

您可以使用 Systems Manager 控制台从 Distributor 中删除软件包版本。

若要使用控制台删除软件包版本

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Distributor。
3. 在 Distributor 主页上，选择要删除其版本的软件包。
4. 在软件包的版本页面上，选择要删除的版本，然后选择 Delete version (删除版本)。
5. 当系统提示确认删除时，选择 Delete package version (删除软件包版本)。

使用命令行删除软件包

您可以使用首选命令行工具从 Distributor 中删除软件包。

Linux & macOS

若要使用 AWS CLI 删除软件包

1. 运行以下命令列出特定软件包的文档。在此命令的结果中，查找要删除的软件包。

```
aws ssm list-documents \
    --filters Key=Name,Values=package-name
```

2. 运行以下命令删除软件包。将 *package-name* 替换为软件包名称。

```
aws ssm delete-document \
```

```
    --name "package-name"
```

3. 再次运行 list-documents 命令，验证是否已删除此软件包。您删除的软件包不应在包含在该列表中。

```
aws ssm list-documents \
    --filters Key=Name,Values=package-name
```

Windows

若要使用 AWS CLI 删除软件包

1. 运行以下命令列出特定软件包的文档。在此命令的结果中，查找要删除的软件包。

```
aws ssm list-documents ^
    --filters Key=Name,Values=package-name
```

2. 运行以下命令删除软件包。将 *package-name* 替换为软件包名称。

```
aws ssm delete-document ^
    --name "package-name"
```

3. 再次运行 list-documents 命令，验证是否已删除此软件包。您删除的软件包不应在包含在该列表中。

```
aws ssm list-documents ^
    --filters Key=Name,Values=package-name
```

PowerShell

若要使用 Tools for PowerShell 删除软件包

1. 运行以下命令列出特定软件包的文档。在此命令的结果中，查找要删除的软件包。

```
$filter = New-Object
 Amazon.SimpleSystemsManagement.Model.DocumentKeyValuesFilter
$filter.Key = "Name"
$filter.Values = "package-name"

Get-SSMDocumentList `
```

```
    -Filters @($filter)
```

2. 运行以下命令删除软件包。将 *package-name* 替换为软件包名称。

```
Remove-SSMDocument `
    -Name "package-name"
```

3. 再次运行 Get-SSMDocumentList 命令，验证是否已删除此软件包。您删除的软件包不应在包含在该列表中。

```
$filter = New-Object
 Amazon.SimpleSystemsManagement.Model.DocumentKeyValuesFilter
$filter.Key = "Name"
$filter.Values = "package-name"

Get-SSMDocumentList `
    -Filters @($filter)
```

使用命令行删除软件包版本

您可以使用首选命令行工具从 Distributor 中删除软件包版本。

Linux & macOS

若要使用 AWS CLI 删除软件包版本

1. 运行以下命令列出软件包的版本。在此命令的结果中，查找要删除的软件包版本。

```
aws ssm list-document-versions \
    --name "package-name"
```

2. 运行以下命令删除软件包版本。将 *package-name* 替换为软件包名称，并将 *version* 替换为版本号。

```
aws ssm delete-document \
    --name "package-name" \
    --document-version version
```

3. 运行 list-document-versions 命令，验证软件包的版本是否已删除。此时应找不到已删除的软件包版本。

```
aws ssm list-document-versions \
    --name "package-name"
```

Windows

若要使用 AWS CLI 删除软件包版本

1. 运行以下命令列出软件包的版本。在此命令的结果中，查找要删除的软件包版本。

```
aws ssm list-document-versions ^
    --name "package-name"
```

2. 运行以下命令删除软件包版本。将 *package-name* 替换为软件包名称，并将 *version* 替换为版本号。

```
aws ssm delete-document ^
    --name "package-name" ^
    --document-version version
```

3. 运行 list-document-versions 命令，验证软件包的版本是否已删除。此时应找不到已删除的软件包版本。

```
aws ssm list-document-versions ^
    --name "package-name"
```

PowerShell

若要使用 Tools for PowerShell 删除软件包版本

1. 运行以下命令列出软件包的版本。在此命令的结果中，查找要删除的软件包版本。

```
Get-SSMDocumentVersionList `
    -Name "package-name"
```

2. 运行以下命令删除软件包版本。将 *package-name* 替换为软件包名称，并将 *version* 替换为版本号。

```
Remove-SSMDocument `
```

```
    -Name "package-name" `
    -DocumentVersion version
```

3. 运行 Get-SSMDocumentVersionList 命令，验证软件包的版本是否已删除。此时应找不到已删除的软件包版本。

```
Get-SSMDocumentVersionList `
    -Name "package-name"
```

有关可以与 list-documents 命令结合使用的其他选项的信息，请参阅 AWS CLI 命令参考 AWS Systems Manager 一节中的 list-documents。有关可以与 delete-document 命令配合使用的其他选项的信息，请参阅 delete-document。

审计和记录 Distributor 活动

您可以使用 AWS CloudTrail 来审核与 Distributor（AWS Systems Manager 中的一项工具）相关的活动。有关 Systems Manager 的审计和日志记录选项的更多信息，请参阅 AWS Systems Manager 中的日志记录和监控。

使用 CloudTrail 审计 Distributor 活动

CloudTrail 可以捕获在 AWS Systems Manager 控制台、AWS Command Line Interface (AWS CLI) 和 Systems Manager 软件开发工具包中进行的 API 调用。可以在 CloudTrail 控制台中查看这些信息，也可以将其存储在 Amazon Simple Storage Service (Amazon S3) 存储桶中。账户的所有 CloudTrail 日志都存储在一个存储桶中。

Run Command 和State Manager操作的日志显示了文档创建、软件包安装和软件包卸载活动。Run Command 和State Manager都是 AWS Systems Manager 中的工具。有关查看和使用 Systems Manager 活动的 CloudTrail 日志的更多信息，请参阅 使用 AWS CloudTrail 记录 AWS Systems Manager API 调用。

AWS Systems ManagerDistributor 故障排除

以下信息可帮助排查使用 Distributor（AWS Systems Manager 中的一项工具）时可能出现的问题。

主题

- 安装了具有相同名称的错误的软件包
- 错误：“无法检索清单：找不到软件包的最新版本”
- 错误：“无法检索清单：验证异常”

- 软件包不受支持（软件包缺少安装操作）
- 错误：无法下载清单：带有名称的文档不存在
- 上传失败。
- 错误：找不到平台：找不到平台清单：oracle，版本 8.9，架构 x86_64

安装了具有相同名称的错误的软件包

问题：您已安装了一个软件包，但 Distributor 安装了一个不同的软件包。

原因：在安装过程中，Systems Manager 先查找 AWS 发布的软件包作为结果，然后才查找用户定义的外部软件包。如果用户定义的软件包名称与 AWS 发布的软件包名称相同，则将安装 AWS 软件包而不是您的软件包。

解决方案：要避免此问题，请将您的软件包命名为与 AWS 发布的软件包不同的名称。

错误：“无法检索清单：找不到软件包的最新版本”

问题：您收到类似如下内容的错误。

```
Failed to retrieve manifest: ResourceNotFoundException: Could not find the latest
 version of package
arn:aws:ssm:::package/package-name status code: 400, request id: guid
```

原因：您使用的 SSM Agent 版本的 Distributor 版本低于 2.3.274.0。

解决方案：将 SSM Agent 版本更新为 2.3.274.0 或更高版本。有关更多信息，请参阅使用 Run Command 更新 SSM Agent或演练：使用 AWS CLI 自动更新 SSM Agent。

错误：“无法检索清单：验证异常”

问题：您收到类似如下内容的错误。

```
Failed to retrieve manifest: ValidationException: 1 validation error detected: Value
 'documentArn'
at 'packageName' failed to satisfy constraint: Member must satisfy regular expression
 pattern:
arn:aws:ssm:region-id:account-id:package/package-name
```

原因：您使用的 SSM Agent 版本的 Distributor 版本低于 2.3.274.0。

解决方案：将 SSM Agent 版本更新为 2.3.274.0 或更高版本。有关更多信息，请参阅使用 Run Command 更新 SSM Agent或演练：使用 AWS CLI 自动更新 SSM Agent。

软件包不受支持（软件包缺少安装操作）

问题：您收到类似如下内容的错误。

```
Package is not supported (package is missing install action)
```

原因：软件包目录结构不正确。

解决方案：不要压缩包含软件和所需脚本的父目录。相反，应直接在绝对路径中创建一个包含所有所需内容的 .zip 文件。要验证是否已正确创建 .zip 文件，请解压缩目标平台目录，并审核目录结构。例如，安装脚本绝对路径应为 */ExamplePackage_targetPlatform*/install.sh。

错误：无法下载清单：带有名称的文档不存在

问题：您收到类似如下内容的错误。

```
Failed to download manifest - failed to retrieve package document description:
 InvalidDocument: Document with name filename does not exist.
```

原因 1：在共享来自其他账户的 Distributor 软件包时，Distributor 无法通过软件包名称找到该软件包。

解决方案 1：在共享来自其他账户的软件包时，请使用软件包的 Amazon 资源名称（ARN），而不只是软件包名称。

原因 2：在使用 VPC 时，没有向自己的 IAM 实例配置文件提供对 AWS 托管式 S3 存储桶的访问权限，而该存储桶包含目标 AWS 区域适用的 AWS-ConfigureAWSPackage 档案。

解决方案 2：确保自己的 IAM 实例配置文件提供向 SSM Agent 提供了对 AWS 托管式 S3 存储桶（该存储桶包含目标 AWS 区域适用的 AWS-ConfigureAWSPackage 文档）的访问权限，如 SSM Agent 与 AWS 托管 S3 存储桶进行通信中所述。

上传失败。

问题：您收到类似如下内容的错误：

```
Upload failed. At least one of your files was not successfully uploaded to your S3
 bucket.
```

原因：您软件包的名称中包含空格。例如，Hello World.msi 将无法上传。

错误：找不到平台：找不到平台清单：oracle，版本 8.9，架构 x86_64

问题：您收到类似如下内容的错误：

```
Failed to find platform: no manifest found for platform: oracle, version 8.9,
 architecture x86_64
```

原因：此错误表示 JSON 软件包清单没有该平台（在本例中为 Oracle Linux）对应的任何定义。

解决方法：从 Trend Micro Deep Security Software 网站下载要分发的软件包。使用 Distributor 简单工作流创建 `.rpm` 软件包。为软件包设置以下值，然后使用 Distributor 完成软件包上传：

```
Platform version: _any
Target platform: oracle
Architecture: x86_64
```

AWS Systems Manager Fleet Manager

Fleet Manager（AWS Systems Manager 中的一项工具）是一种统一的用户界面（UI）体验，可帮助您远程管理在 AWS 上或在本地运行的节点。借助 Fleet Manager，您可以从一个控制台查看整个服务器机群的运行状况和性能状态。您还可以从单个节点收集数据，以便从该控制台执行常见的故障排除和管理任务。这包括使用远程桌面协议 (RDP) 连接到 Windows 实例、查看文件夹和文件内容、Windows 注册表管理、操作系统用户管理等等。

要开始使用 Fleet Manager，请打开 Systems Manager 控制台。在导航窗格中，选择 Fleet Manager。

谁应该使用 Fleet Manager？

任何想要集中管理节点实例集的 AWS 客户都应使用 Fleet Manager。

我的组织如何从 Fleet Manager 获益？

Fleet Manager 具备下列优势：

- 可以执行各种常见系统管理任务，而不必手动连接到您的托管式节点。
- 可从单个统一控制台管理在多个平台上运行的节点。
- 可从单个统一控制台管理运行不同操作系统的节点。
- 可以提高系统管理的效率。

Fleet Manager 具有哪些功能？

Fleet Manager 的主要功能包括以下方面：

- 访问 Red Hat 知识库门户

  通过 Red Hat Enterprise Linux (RHEL) 实例，访问 Red Hat 知识库门户上的二进制文件、知识共享和论坛。

- 托管式节点状态

  可以查看哪些托管式实例处于 `running` 状态，以及哪些实例处于 `stopped` 状态。有关已停止实例的更多信息，请参阅《Amazon EC2 用户指南》中的停止和启动您的实例。对于 AWS IoT Greengrass 核心设备，您可以查看状态显示为 `online`、`offline` 还是 `Connection lost`。

  > **Note**
  >
  > 如果您已在 2021 年 7 月 12 日之前停止托管式实例，它将不会显示 `stopped` 标记。要显示该标记，请启动和停止该实例。

- 查看实例信息

  可以查看有关连接到托管实例的卷上存储的文件夹和文件数据的信息、有关实例的实时性能数据，以及存储在实例上的日志数据。

- 查看边缘设备信息

  查看设备的 AWS IoT Greengrass 事物名称、SSM Agent ping 状态和版本等信息。

- 管理账户和注册表

  可以管理您的实例上的操作系统（OS）用户账户和 Windows 实例上的注册表。

- 控制对功能的访问权限

  可以使用 AWS Identity and Access Management (IAM) 策略控制对 Fleet Manager 功能的访问。借助这些策略，您可以控制企业中的哪些个人用户或组能够使用各项 Fleet Manager 功能，以及他们能够管理哪些托管式节点。

主题

- 设置 Fleet Manager
- 使用托管式节点

- 使用默认主机管理配置自动管理 EC2 实例
- 使用 Remote Desktop 连接到 Windows Server 托管式实例
- 在托管式实例上管理 Amazon EBS 卷
- 访问 Red Hat 知识库门户
- 排除托管式节点可用性的问题

设置 Fleet Manager

您 AWS 账户中的用户必须先获得必要权限，然后才能使用 Fleet Manager（AWS Systems Manager 中的一项工具）来监控和管理托管式节点。此外，任何 Amazon Elastic Compute Cloud（Amazon EC2）实例；AWS IoT Greengrass 核心设备；以及要使用 Fleet Manager 监控和管理的本地服务器、边缘设备和虚拟机（VM）必须是 Systems Manager 托管式节点。托管节点是指在混合和多云环境中配置为与 Systems Manager 一起使用的任何计算机。

这意味着节点必须满足某些先决条件并使用 AWS Systems Manager Agent (SSM Agent) 进行配置。

根据计算机类型，请参阅以下主题之一，以确保您的计算机满足托管式节点的要求。

- Amazon EC2 实例：使用 Systems Manager 管理 EC2 实例

> **Tip**
>
> 您还可以使Quick Setup（AWS Systems Manager 中的一项工具），帮助将 Amazon EC2 实例快速配置为个人账户中的托管式实例。如果您的企业使用 AWS Organizations，您还可以在多个企业单位 (OU) 和 AWS 区域 中配置实例。有关使用 Quick Setup 配置托管实例的更多信息，请参阅 使用 Quick Setup 设置 Amazon EC2 主机管理。

- 本地服务器和云中的其他服务器类型：使用 Systems Manager 管理混合和多云环境中的节点
- AWS IoT Greengrass（边缘）设备：使用 Systems Manager 管理边缘设备

示例策略

- 控制对 Fleet Manager 的访问

控制对 Fleet Manager 的访问

要使用 Fleet Manager（AWS Systems Manager 中的一项工具），您的 AWS Identity and Access Management（IAM）用户或角色必须拥有所需权限。您可以创建一个 IAM policy 提供对所有 Fleet

Manager 功能的访问权限，或者修改策略以授予对所选功能的访问权限。然后，您可以将这些权限授予您账户中的用户或身份。

任务 1：创建 IAM 策略以定义访问权限

按照《IAM 用户指南》中的以下主题中提供的方法之一创建 IAM，为身份（用户、角色或用户组）提供 Fleet Manager 的访问权限：

- 使用客户管理型策略定义自定义 IAM 权限

您可以使用我们在下面提供的示例策略之一，或者根据要授予的权限对其进行修改。我们为完全的 Fleet Manager 访问权限和只读访问权限提供策略示例。

任务 2：为用户附加 IAM 策略以授予权限

在您创建了一个或多个定义 Fleet Manager 的访问权限的 IAM 策略后，使用《IAM 用户指南》中的以下过程之一，向账户中的身份授予这些权限：

- 添加 IAM 身份权限（控制台）
- 添加 IAM 身份权限（AWS CLI）
- 添加 IAM 身份权限（AWS API）

主题

- Fleet Manager 管理员访问权限示例策略
- Fleet Manager 只读访问权限实例策略

Fleet Manager 管理员访问权限示例策略

以下策略提供了对所有 Fleet Manager 功能的权限。这意味着用户可以创建和删除本地用户和组、修改任何本地组的组成员资格，以及修改 Windows Server 注册表键或值。将每个#######替换为您自己的信息。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2",
            "Effect": "Allow",
            "Action": [
```

```
                "ec2:CreateTags",
                "ec2:DeleteTags",
                "ec2:DescribeInstances",
                "ec2:DescribeTags"
            ],
            "Resource": "*"
        },
        {
            "Sid": "General",
            "Effect": "Allow",
            "Action": [
                "ssm:AddTagsToResource",
                "ssm:DescribeInstanceAssociationsStatus",
                "ssm:DescribeInstancePatches",
                "ssm:DescribeInstancePatchStates",
                "ssm:DescribeInstanceProperties",
                "ssm:GetCommandInvocation",
                "ssm:GetServiceSetting",
                "ssm:GetInventorySchema",
                "ssm:ListComplianceItems",
                "ssm:ListInventoryEntries",
                "ssm:ListTagsForResource",
                "ssm:ListCommandInvocations",
                "ssm:ListAssociations",
                "ssm:RemoveTagsFromResource"
            ],
            "Resource": "*"
        },
        {
            "Sid": "DefaultHostManagement",
            "Effect": "Allow",
            "Action": [
                "ssm:ResetServiceSetting",
                "ssm:UpdateServiceSetting"
            ],
            "Resource": "arn:aws:ssm:us-east-1:111122223333:servicesetting/ssm/
managed-instance/default-ec2-instance-management-role"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
```

```
            "Resource": "arn:aws:iam::111122223333:role/service-role/
AWSSystemsManagerDefaultEC2InstanceManagementRole",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ssm.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Sid": "SendCommand",
            "Effect": "Allow",
            "Action": [
                "ssm:GetDocument",
                "ssm:SendCommand",
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ec2:*:111122223333:instance/*",
                "arn:aws:ssm:*:111122223333:managed-instance/*",
                "arn:aws:ssm:*:111122223333:document/SSM-SessionManagerRunShell",
                "arn:aws:ssm:*:*:document/AWS-PasswordReset",
                "arn:aws:ssm:*:*:document/AWSFleetManager-AddUsersToGroups",
                "arn:aws:ssm:*:*:document/AWSFleetManager-CopyFileSystemItem",
                "arn:aws:ssm:*:*:document/AWSFleetManager-CreateDirectory",
                "arn:aws:ssm:*:*:document/AWSFleetManager-CreateGroup",
                "arn:aws:ssm:*:*:document/AWSFleetManager-CreateUser",
                "arn:aws:ssm:*:*:document/AWSFleetManager-CreateUserInteractive",
                "arn:aws:ssm:*:*:document/AWSFleetManager-
CreateWindowsRegistryKey",
                "arn:aws:ssm:*:*:document/AWSFleetManager-DeleteFileSystemItem",
                "arn:aws:ssm:*:*:document/AWSFleetManager-DeleteGroup",
                "arn:aws:ssm:*:*:document/AWSFleetManager-DeleteUser",
                "arn:aws:ssm:*:*:document/AWSFleetManager-
DeleteWindowsRegistryKey",
                "arn:aws:ssm:*:*:document/AWSFleetManager-
DeleteWindowsRegistryValue",
                "arn:aws:ssm:*:*:document/AWSFleetManager-GetDiskInformation",
                "arn:aws:ssm:*:*:document/AWSFleetManager-GetFileContent",
                "arn:aws:ssm:*:*:document/AWSFleetManager-GetFileSystemContent",
                "arn:aws:ssm:*:*:document/AWSFleetManager-GetGroups",
                "arn:aws:ssm:*:*:document/AWSFleetManager-
GetPerformanceCounters",
```

```
                "arn:aws:ssm:*:*:document/AWSFleetManager-GetProcessDetails",
                "arn:aws:ssm:*:*:document/AWSFleetManager-GetUsers",
                "arn:aws:ssm:*:*:document/AWSFleetManager-GetWindowsEvents",
                "arn:aws:ssm:*:*:document/AWSFleetManager-
GetWindowsRegistryContent",
                "arn:aws:ssm:*:*:document/AWSFleetManager-MountVolume",
                "arn:aws:ssm:*:*:document/AWSFleetManager-MoveFileSystemItem",
                "arn:aws:ssm:*:*:document/AWSFleetManager-RemoveUsersFromGroups",
                "arn:aws:ssm:*:*:document/AWSFleetManager-RenameFileSystemItem",
                "arn:aws:ssm:*:*:document/AWSFleetManager-
SetWindowsRegistryValue",
                "arn:aws:ssm:*:*:document/AWSFleetManager-StartProcess",
                "arn:aws:ssm:*:*:document/AWSFleetManager-TerminateProcess"
            ]
        },
        {
            "Sid": "TerminateSession",
            "Effect": "Allow",
            "Action": [
                "ssm:TerminateSession"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "ssm:resourceTag/aws:ssmmessages:session-id": [
                        "${aws:userid}"
                    ]
                }
            }
        }
    ]
}
```

Fleet Manager 只读访问权限实例策略

以下策略提供了对只读 Fleet Manager 功能的权限。将每个#######替换为您自己的信息。

JSON

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

```
        {
            "Sid": "EC2",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeTags"
            ],
            "Resource": "*"
        },
        {
            "Sid": "General",
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeInstanceAssociationsStatus",
                "ssm:DescribeInstancePatches",
                "ssm:DescribeInstancePatchStates",
                "ssm:DescribeInstanceProperties",
                "ssm:GetCommandInvocation",
                "ssm:GetServiceSetting",
                "ssm:GetInventorySchema",
                "ssm:ListComplianceItems",
                "ssm:ListInventoryEntries",
                "ssm:ListTagsForResource",
                "ssm:ListCommandInvocations",
                "ssm:ListAssociations"
            ],
            "Resource": "*"
        },
        {
            "Sid": "SendCommand",
            "Effect": "Allow",
            "Action": [
                "ssm:GetDocument",
                "ssm:SendCommand",
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ec2:*:111122223333:instance/*",
                "arn:aws:ssm:*:111122223333:managed-instance/*",
                "arn:aws:ssm:*:111122223333:document/SSM-SessionManagerRunShell",
                "arn:aws:ssm:*:*:document/AWSFleetManager-GetDiskInformation",
                "arn:aws:ssm:*:*:document/AWSFleetManager-GetFileContent",
                "arn:aws:ssm:*:*:document/AWSFleetManager-GetFileSystemContent",
                "arn:aws:ssm:*:*:document/AWSFleetManager-GetGroups",
```

```
                "arn:aws:ssm:*:*:document/AWSFleetManager-
GetPerformanceCounters",
                "arn:aws:ssm:*:*:document/AWSFleetManager-GetProcessDetails",
                "arn:aws:ssm:*:*:document/AWSFleetManager-GetUsers",
                "arn:aws:ssm:*:*:document/AWSFleetManager-GetWindowsEvents",
                "arn:aws:ssm:*:*:document/AWSFleetManager-
GetWindowsRegistryContent"
            ]
        },
        {
            "Sid": "TerminateSession",
            "Effect": "Allow",
            "Action": [
                "ssm:TerminateSession"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "ssm:resourceTag/aws:ssmmessages:session-id": [
                        "${aws:userid}"
                    ]
                }
            }
        }
    ]
}
```

使用托管式节点

托管式节点是为 AWS Systems Manager 配置的任何计算机。您可以将以下计算机类型配置为托管式节点：

- Amazon Elastic Compute Cloud（Amazon EC2）实例
- 您本地的服务器（本地服务器）
- AWS IoT Greengrass 核心设备
- AWS IoT 和非 AWS 边缘设备
- 虚拟机，包括其他云环境中的虚拟机

在 Systems Manager 控制台中，任何带有“mi-”前缀的计算机都已使用混合激活配置为托管式节点。边缘设备显示其 AWS IoT 事物名称。

> **Note**
>
> 对于 macOS 实例，唯一受支持的功能是查看文件系统。

关于 Systems Manager 实例套餐

AWS Systems Manager 提供了标准实例层和高级实例层。两者都支持混合和多云环境中的托管式节点。通过标准实例套餐，每个 AWS 区域 内每个 AWS 账户 最多可以注册 1000 台计算机。如果您需要在一个账户和区域中注册超过 1000 台计算机，则使用高级实例套餐。在高级实例套餐中，您可以根据需要创建许多托管式节点。为 Systems Manager 配置的所有托管式节点均按使用量付费。有关启用高级实例套餐的更多信息，请参阅 打开高级实例套餐。有关定价的更多信息，请参阅 AWS Systems Manager 定价。

请注意以下关于标准实例层和高级实例层的附加信息：

- 通过高级实例，您还可以使用 AWS Systems Manager Session Manager 连接到混合和多云环境中的非 EC2 节点。Session Manager 提供了对您的实例的交互式 Shell 访问。有关更多信息，请参阅 AWS Systems Manager Session Manager。
- 标准实例配额也适用于使用 Systems Manager 本地激活的 Amazon EC2 实例（但这不是常见情况）。
- 要修补 Microsoft 在虚拟机 (VM) 本地实例上发布的应用程序，请激活高级实例套餐。使用高级实例套餐需支付费用。修补 Microsoft 在 Amazon Elastic Compute Cloud (Amazon EC2) 实例上发布的应用程序不收取额外费用。有关更多信息，请参阅 Microsoft 在 Windows Server 上发布的补丁应用程序。

显示托管式节点

如果您没有看到控制台中列出托管式节点，则执行以下操作：

1. 确认控制台在您创建托管式节点的 AWS 区域 中处于打开状态。您可以使用顶部的列表（控制台右上角）切换区域。
2. 确认托管式节点的设置步骤是否满足 Systems Manager 要求。有关信息，请参阅为 AWS Systems Manager 设置托管节点。
3. 对于非 EC2 计算机，请确认您已完成混合激活过程。有关更多信息，请参阅 使用 Systems Manager 管理混合和多云环境中的节点。

请注意以下额外信息：

- Fleet Manager 控制台不显示已终止的 Amazon EC2 节点。
- Systems Manager 需要准确的时间参考以便在计算机上执行操作。如果托管式节点上的日期和时间设置不正确，计算机可能与 API 请求的签名日期不匹配。有关更多信息，请参阅 应用场景和最佳实践。
- 创建或编辑标签时，系统可能需要最多 1 小时才能在表筛选条件中显示更改。
- 在托管节点的状态保持 `Connection Lost` 至少 30 天后，该节点可能不再会在 Fleet Manager 控制台中列出。必须首先解决导致连接中断的问题，然后才能将其恢复到列表中。有关故障排除提示，请参阅 排除托管式节点可用性的问题。

验证托管式节点的 Systems Manager 支持

AWS Config 提供了 AWS 托管式规则，AWS Config 使用这些可自定义的预定义规则来评估您的 AWS 资源配置是否符合常用的最佳实践。AWS ConfigManaged Rules 包括 ec2-instance-managed-by-systems-manager 规则。该规则会检查您账户中的 Amazon EC2 实例是否由 Systems Manager 托管。有关更多信息，请参阅 AWS Config Managed Rules。

改善托管式节点的安保状况

有关在托管式节点上提高针对未授权根级别命令的安保状况的更多信息，请参阅 通过 SSM Agent 限制对根级别命令的访问。

取消注册托管式节点

您可以随时取消注册托管式节点。例如，如果您正在管理多个具有相同 AWS Identity and Access Management (IAM) 角色的节点，并且您发现了任何类型的恶意行为，则您可以随时取消注册任意数量的计算机。（要重新注册同一台计算机，必须使用不同的混合激活码和激活 ID，不能使用先前注册时用过的激活码和激活 ID。）有关取消注册托管式节点的信息，请参阅 在混合和多云环境中取消注册托管式节点。

主题

- 配置实例套餐
- 在托管式节点上重置密码
- 在混合和多云环境中取消注册托管式节点
- 通过 Fleet Manager 来使用 OS 文件系统
- 监控托管式节点性能

- 使用进程
- 查看托管式节点上的日志
- 使用 Fleet Manager 来管理托管式节点上的 OS 用户账户和组
- 在托管式节点上管理 Windows 注册表

配置实例套餐

本主题描述了必须激活高级实例化套餐的情况。

AWS Systems Manager 为混合和多云环境中的非 EC2 计算机提供了标准实例套餐和高级实例套餐。

每个 AWS 区域 内每个账户可以注册最多 1000 个标准混合激活节点，无需额外收费。但是，注册超过 1000 个混合节点需要激活高级实例套餐。使用高级实例套餐需支付费用。有关更多信息，请参阅 AWS Systems Manager 定价。

即使注册的混合激活节点少于 1000 个，但还有两种场景也需要高级实例套餐：

- 您希望使用 Session Manager 连接到非 EC2 节点。
- 您希望修补 Microsoft 在非 EC2 节点上发布的应用程序（非操作系统）。

> **Note**
>
> 修补 Microsoft 在 Amazon EC2 实例上发布的应用程序不收取费用。

高级实例套餐详细情况

以下信息详细介绍了必须激活高级实例套餐的三种情况。

场景 1：您希望注册超过 1000 个混合激活节点

使用标准实例套餐，您可以在混合和多云环境中为特定账户中为每个 AWS 区域 注册最多 1000 个非 EC2 节点，无需额外收费。如果您需要在一个区域中注册超过 1000 个非 EC2 节点，则必须使用高级实例套餐。然后，您可以为混合和多云环境激活任意数量的计算机。高级实例套餐根据激活为 Systems Manager 托管式节点的高级节点数量以及这些节点运行的小时数收费。

如果您在特定账户的某个区域中有超过 1000 个本地节点，那么使用创建混合激活以将节点注册到 Systems Manager 中所述的激活过程的所有 Systems Manager 托管式节点都需支付费用。

> **Note**
>
> 您还可以使用 Systems Manager 混合激活现有的 Amazon Elastic Compute Cloud (Amazon EC2) 实例，并将其作为非 EC2 实例使用，例如用于测试。这些节点也可以成为混合节点。但这不是常见的情况。

情况 2：在混合激活节点上修补 Microsoft 发布的应用程序

如果要在混合和多云环境中的非 EC2 节点上修补 Microsoft 发布的应用程序，也需要高级实例套餐。如果您激活高级实例套餐以修补非 EC2 节点上的 Microsoft 应用程序，则即使您的节点少于 1000 个，也会对所有本地节点收取费用。

修补 Microsoft 在 Amazon Elastic Compute Cloud (Amazon EC2) 实例上发布的应用程序不收取额外费用。有关更多信息，请参阅 Microsoft 在 Windows Server 上发布的补丁应用程序。

情况 3：使用 Session Manager 连接到混合激活节点

Session Manager 提供了对实例的交互式 shell 访问。要使用 Session Manager 连接到混合激活托管式节点，必须激活高级实例套餐。然后，所有混合激活的节点都将收取费用，即使您的节点少于 1000 个。

摘要：何时需要高级实例套餐？

使用下表查看何时必须使用高级实例套餐，以及哪些情况将收取额外费用。

情况	是否需要高级实例套餐？	是否收取额外费用？
特定账户中我的区域的混合激活节点数超过 1000 个。	支持	是
我想用 Patch Manager 在任意数量（甚至少于 1000 个）的混合激活节点上修补 Microsoft 发布的应用程序。	支持	是
我想用 Session Manager 连接任意数量（甚至少于 1000 个）的混合激活节点。	支持	是

情况	是否需要高级实例套餐？	是否收取额外费用？
1. 特定账户中某个区域的混合激活节点数不超过 1000 个；且 2. 我没有在任何混合激活节点上修补 Microsoft 应用程序；且 3. 我没有使用 Session Manager 连接到任何混合激活的节点。	否	否

主题

- 打开高级实例套餐
- 从高级实例套餐还原到标准实例套餐

打开高级实例套餐

AWS Systems Manager 为混合和多云环境中的非 EC2 计算机提供了标准实例套餐和高级实例套餐。通过标准实例套餐，每个 AWS 区域 内每个 AWS 账户 最多可以注册 1000 台混合激活的计算机。使用 Patch Manager 在非 EC2 节点上修补 Microsoft 发布的应用程序，并使用 Session Manager 连接到非 EC2 节点也需要使用高级实例套餐。有关更多信息，请参阅 打开高级实例套餐。

本节介绍如何配置您的混合和多云环境来使用高级实例套餐。

开始前的准备工作

查看高级实例的定价详细信息。高级实例按使用量收费。有关更多信息，请参阅 AWS Systems Manager 定价。

配置打开高级实例套餐的权限

请验证并确保您在 AWS Identity and Access Management (IAM) 中拥有将您的环境从标准实例套餐更改为高级实例套餐的权限。您必须将 `AdministratorAccess` IAM policy 附加到您的用户、组或角色，或者您必须拥有更改 Systems Manager 激活套餐服务设置的权限。激活套餐设置使用以下 API 操作：

- GetServiceSetting

- UpdateServiceSetting
- ResetServiceSetting

使用以下过程可将内联 IAM policy 添加到用户账户。此策略使用户能够查看当前的托管实例套餐设置。此策略还使用户能够更改或重置指定 AWS 账户 和 AWS 区域 中的当前设置。

1. 登录 AWS Management Console，单击 https://console.aws.amazon.com/iam/ 打开 IAM 控制台。
2. 在导航窗格中，选择用户。
3. 在列表中，选择要在其中嵌入策略的用户的名称。
4. 选择 Permissions（权限）选项卡。
5. 在页面右侧，在 Permission policies (权限策略) 下，选择 Add inline policy (添加内联策略)。
6. 选择 JSON 选项卡。
7. 将默认内容替换为以下内容：

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetServiceSetting"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:ResetServiceSetting",
                "ssm:UpdateServiceSetting"
            ],
            "Resource": "arn:aws:ssm:us-east-1:111122223333:servicesetting/
ssm/managed-instance/activation-tier"
        }
    ]
}
```

8. 选择 Review policy (审核策略)。
9. 在 Review policy (审核策略) 页面上，对于 Name (名称)，输入内联策略的名称。例如：**`Managed-Instances-Tier`**。
10. 选择创建策略。

管理员可以将以下内联策略分配给该用户，从而指定只读权限。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetServiceSetting"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Deny",
            "Action": [
                "ssm:ResetServiceSetting",
                "ssm:UpdateServiceSetting"
            ],
            "Resource": "*"
        }
    ]
}
```

有关创建和编辑 IAM policy 的更多信息，请参阅《IAM 用户指南》中的创建 IAM policy。

打开高级实例套餐（控制台）

以下过程介绍如何使用 Systems Manager 控制台，将使用托管实例激活在指定 AWS 账户 和 AWS 区域 中添加的所有非 EC2 节点更改为使用高级实例套餐。

开始前的准备工作

验证并确保控制台在您创建托管实例的 AWS 区域 中处于打开状态。您可以使用顶部的列表（控制台右上角）切换区域。

验证您是否已在混合和多云环境中完成 Amazon Elastic Compute Cloud（Amazon EC2）实例以及非 EC2 计算机的设置要求。有关信息，请参阅为 AWS Systems Manager 设置托管节点。

> **Important**
>
> 以下过程介绍了如何更改账户级别设置。此更改会导致向您的账户计费。

打开高级实例套餐（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Fleet Manager。
3. 选择设置，然后选择更改实例等级设置。
4. 查看对话框中有关更改账户设置的信息。
5. 如果您批准，请选择要接受的选项，然后选择更改设置。

系统可能需要几分钟才能完成将所有实例从标准实例套餐转移至高级实例套餐的过程。

> **Note**
>
> 有关恢复为标准实例套餐的信息，请参阅 从高级实例套餐还原到标准实例套餐。

打开高级实例套餐 (AWS CLI)

以下过程介绍了如何使用 AWS Command Line Interface，将使用托管实例激活在指定 AWS 账户 和 AWS 区域 中添加的所有本地服务器和 VM 更改为使用高级实例套餐。

> **Important**
>
> 以下过程介绍了如何更改账户级别设置。此更改会导致向您的账户计费。

使用 AWS CLI 打开高级实例套餐

1. 打开 AWS CLI 并运行以下命令。将每个*#######*替换为您自己的信息。

   Linux & macOS

   ```
   aws ssm update-service-setting \
       --setting-id arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-instance/activation-tier \
       --setting-value advanced
   ```

   Windows

   ```
   aws ssm update-service-setting ^
       --setting-id arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-instance/activation-tier ^
       --setting-value advanced
   ```

   如果此命令成功，则无任何输出。

2. 运行以下命令来查看当前的 AWS 账户 和 AWS 区域 中托管式节点的当前服务设置。

   Linux & macOS

   ```
   aws ssm get-service-setting \
       --setting-id arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-instance/activation-tier
   ```

   Windows

   ```
   aws ssm get-service-setting ^
       --setting-id arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-instance/activation-tier
   ```

   此命令会返回如下信息。

   ```
   {
       "ServiceSetting": {
           "SettingId": "/ssm/managed-instance/activation-tier",
           "SettingValue": "advanced",
   ```

```
        "LastModifiedDate": 1555603376.138,
        "LastModifiedUser": "arn:aws:sts::123456789012:assumed-role/
Administrator/User_1",
        "ARN": "arn:aws:ssm:us-east-2:123456789012:servicesetting/ssm/managed-
instance/activation-tier",
        "Status": "PendingUpdate"
    }
}
```

打开高级实例套餐 (PowerShell)

以下过程介绍了如何使用 AWS Tools for Windows PowerShell，将使用托管实例激活在指定 AWS 账户 和 AWS 区域 中添加的所有 本地服务器和 VM 更改为使用高级实例套餐。

> **Important**
>
> 以下过程介绍了如何更改账户级别设置。此更改会导致向您的账户计费。

使用 PowerShell 打开高级实例套餐

1. 打开 AWS Tools for Windows PowerShell 并运行以下命令。将每个*#######*替换为您自己的信息。

   ```
   Update-SSMServiceSetting `
       -SettingId "arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-
   instance/activation-tier" `
       -SettingValue "advanced"
   ```

   如果此命令成功，则无任何输出。

2. 运行以下命令来查看当前的 AWS 账户 和 AWS 区域 中托管式节点的当前服务设置。

   ```
   Get-SSMServiceSetting `
       -SettingId "arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-
   instance/activation-tier"
   ```

   此命令会返回如下信息。

```
ARN:arn:aws:ssm:us-east-2:123456789012:servicesetting/ssm/managed-instance/
activation-tier
LastModifiedDate : 4/18/2019 4:02:56 PM
LastModifiedUser : arn:aws:sts::123456789012:assumed-role/Administrator/User_1
SettingId        : /ssm/managed-instance/activation-tier
SettingValue     : advanced
Status           : PendingUpdate
```

系统可能需要几分钟才能完成将所有节点从标准实例套餐转移至高级实例套餐的过程。

> **Note**
>
> 有关恢复为标准实例套餐的信息，请参阅 从高级实例套餐还原到标准实例套餐。

从高级实例套餐还原到标准实例套餐

本节介绍如何将高级实例套餐中运行的混合激活节点更改回标准实例套餐。此配置适用于 AWS 账户和单个 AWS 区域 中的所有混合激活节点。

开始前的准备工作

查看以下重要详细信息。

> **Note**
>
> - 如果您在账户和区域中运行的混合激活节点超过 1000 个，则无法恢复到标准实例套餐。您必须先取消注册节点，直到实例数为 1000 或更少为止。这也适用于使用 Systems Manager 混合激活的 Amazon Elastic Compute Cloud（Amazon EC2）实例（但这不是常见情况）。有关更多信息，请参阅 在混合和多云环境中取消注册托管式节点。
> - 恢复后，便无法使用Session Manager（AWS Systems Manager 中的一项工具）以交互方式访问混合激活节点。
> - 恢复后，便无法使用Patch Manager（AWS Systems Manager 中的一项工具）修补 Microsoft 在混合激活节点上发布的应用程序。
> - 将所有混合激活节点恢复为标准实例套餐的过程可能需要 30 分钟或更长时间才能完成。

本节介绍如何将 AWS 账户 和 AWS 区域 中的所有混合激活节点从高级实例套餐恢复到标准实例套餐。

恢复为标准实例套餐（控制台）

以下过程介绍了如何使用 Systems Manager 控制台将混合和多云环境中的所有混合激活节点更改为使用指定 AWS 账户 和 AWS 区域 中的标准实例套餐。

恢复为标准实例套餐（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择 Account settings (账户设置) 下拉菜单，然后选择 Instance tier settings (实例套餐设置)。
4. 选择 Change account settings (更改账户设置)。
5. 查看弹出窗口中有关更改账户设置的信息，然后，如果您批准，请选择接受并继续的选项。

恢复为标准实例套餐 (AWS CLI)

以下过程介绍了如何使用 AWS Command Line Interface 将混合和多云环境中的所有混合激活节点更改为使用指定 AWS 账户 和 AWS 区域 中的标准实例套餐。

使用 AWS CLI 恢复为标准实例套餐

1. 打开 AWS CLI 并运行以下命令。将每个*#######*替换为您自己的信息。

   Linux & macOS

   ```
   aws ssm update-service-setting \
       --setting-id arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-instance/activation-tier \
       --setting-value standard
   ```

   Windows

   ```
   aws ssm update-service-setting ^
       --setting-id arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-instance/activation-tier ^
       --setting-value standard
   ```

如果此命令成功，则无任何输出。

2. 请在 30 分钟后运行以下命令，以查看当前 AWS 账户 和 AWS 区域 中托管实例的设置。

Linux & macOS

```
aws ssm get-service-setting \
    --setting-id arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-
instance/activation-tier
```

Windows

```
aws ssm get-service-setting ^
    --setting-id arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-
instance/activation-tier
```

此命令会返回如下信息。

```
{
    "ServiceSetting": {
        "SettingId": "/ssm/managed-instance/activation-tier",
        "SettingValue": "standard",
        "LastModifiedDate": 1555603376.138,
        "LastModifiedUser": "System",
        "ARN": "arn:aws:ssm:us-east-2:123456789012:servicesetting/ssm/managed-
instance/activation-tier",
        "Status": "Default"
    }
}
```

在请求获得批准后，状态将更改为 Default（默认）。

恢复为标准实例套餐 (PowerShell)

以下过程介绍了如何使用 AWS Tools for Windows PowerShell 将混合和多云环境中的混合激活节点更改为使用指定 AWS 账户 和 AWS 区域 中的标准实例套餐。

使用 PowerShell 恢复为标准实例套餐

1. 打开 AWS Tools for Windows PowerShell 并运行以下命令。

```
Update-SSMServiceSetting `
    -SettingId "arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-
instance/activation-tier" `
    -SettingValue "standard"
```

如果此命令成功，则无任何输出。

2. 请在 30 分钟后运行以下命令，以查看当前 AWS 账户 和 AWS 区域 中托管实例的设置。

```
Get-SSMServiceSetting `
    -SettingId "arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-
instance/activation-tier"
```

此命令会返回如下信息。

```
ARN: arn:aws:ssm:us-east-2:123456789012:servicesetting/ssm/managed-instance/
activation-tier
LastModifiedDate : 4/18/2019 4:02:56 PM
LastModifiedUser : System
SettingId        : /ssm/managed-instance/activation-tier
SettingValue     : standard
Status           : Default
```

在请求获得批准后，状态将更改为 Default（默认）。

在托管式节点上重置密码

您可以在托管式节点上重置任何用户的密码。这包括 Amazon Elastic Compute Cloud (Amazon EC2) 实例；AWS IoT Greengrass 核心设备；以及由 AWS Systems Manager 管理的本地服务器、边缘设备和虚拟机 (VM)。密码重置功能构建于Session Manager（AWS Systems Manager 中的一项工具）的基础之上。您可以使用此功能连接到托管式节点，而无需打开入站端口、维护堡垒主机或管理 SSH 密钥。

当用户忘记密码或者您想快速更新密码而不与托管式节点建立 RDP 或 SSH 连接时，密码重置很有用。

先决条件

必须满足以下要求，才能在托管式节点上重置密码：

- 要在其上更改密码的托管式节点必须是 Systems Manager 托管式节点。此外，必须在托管式节点上安装 SSM Agent 版本 2.3.668.0 或更高版本。有关安装或更新 SSM Agent 的信息，请参阅 使用 SSM Agent。
- 密码重置功能使用为您的账户设置的 Session Manager 配置连接到托管式节点。因此，必须已在当前 AWS 区域中为您的账户完成使用 Session Manager 的先决条件。有关更多信息，请参阅 设置 Session Manager。

  > **Note**
  >
  > 仅为高级实例套餐提供对本地节点的 Session Manager 支持。有关更多信息，请参阅 打开高级实例套餐。

- 更改密码的 AWS 用户必须对托管式节点具有 `ssm:SendCommand` 权限。有关更多信息，请参阅 根据标签限制 Run Command 访问。

限制访问

您可以限制用户重置特定托管式节点的密码的能力。通过将基于身份的策略与 `AWS-PasswordReset` SSM 文档结合起来用于 Session Manager `ssm:StartSession` 操作，可以实现这一目的。有关更多信息，请参阅控制用户会话对实例的访问权限。

加密数据

可为 Session Manager 数据打开 AWS Key Management Service (AWS KMS) 完全加密，以将密码重置选项用于托管式节点。有关更多信息，请参阅 启用会话数据的 KMS 密钥加密（控制台）。

在托管式节点上重置密码

您可以使用 Systems Manager Fleet Manager 控制台或 AWS Command Line Interface (AWS CLI)，在 Systems Manager 托管式节点上重置密码。

在托管式节点（控制台）上更改密码

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Fleet Manager。
3. 选择需要新密码的节点旁边的按钮。

4. 选择实例操作，重置密码。
5. 对于 User name (用户名)，请输入您要更改其密码的用户的名称。这可以是任何在节点上具有账户的用户名。
6. 选择提交。
7. 按照输入新密码命令窗口中的提示指定新密码。

> **Note**
>
> 如果托管式节点上的 SSM Agent 版本不支持密码重置，系统将提示您使用 Run Command（AWS Systems Manager 中的一项工具）安装受支持的版本。

在托管式节点 (AWS CLI) 上重置密码

1. 要在托管式节点上重置用户的密码，请运行以下命令。将每个*#######*替换为您自己的信息。

> **Note**
>
> 要使用 AWS CLI 重置密码，Session Manager 插件必须已安装到您的本地计算机上。有关信息，请参阅为 AWS CLI 安装 Session Manager 插件。

Linux & macOS

```
aws ssm start-session \
    --target instance-id \
    --document-name "AWS-PasswordReset" \
    --parameters '{"username": ["user-name"]}'
```

Windows

```
aws ssm start-session ^
    --target instance-id ^
    --document-name "AWS-PasswordReset" ^
    --parameters username="user-name"
```

2. 按照输入新密码命令窗口中的提示指定新密码。

排除在托管式节点上重置密码的问题

通过确保您已完成密码重置先决条件，可以解决许多密码重置问题。对于其他问题，请使用以下信息来帮助您解决密码重置问题。

主题

- 托管式节点不可用
- SSM Agent 不是最新版本（控制台）
- 未提供密码重置选项 (AWS CLI)
- 未获授权，无法运行 ssm:SendCommand
- Session Manager 错误消息

托管式节点不可用

问题：您希望在 Managed instances（托管式实例）控制台页面上重置托管式节点的密码，但该节点不在列表中。

- 解决方案：可能未为 Systems Manager 配置您要连接的托管式节点。要将 Amazon EC2 实例与 Systems Manager 结合使用，必须将 AWS Identity and Access Management (IAM) 实例配置文件附加到该实例，该配置文件将向 Systems Manager 授予在您的实例上执行操作的权限。有关信息，请参阅配置 Systems Manager 所需的实例权限。

  要将非 EC2 计算机与 Systems Manager 结合使用，请创建一个 IAM 服务角色，该角色将向 Systems Manager 授予在您的托管式节点上执行操作的权限。有关更多信息，请参阅在混合和多云环境中创建 Systems Manager 所需的 IAM 服务角色。（仅为高级实例套餐提供对本地服务器和 VM 的 Session Manager 支持。有关更多信息，请参阅 打开高级实例套餐。）

SSM Agent 不是最新版本（控制台）

问题：一条消息报告 SSM Agent 的此版本不支持密码重置功能。

- 解决方案：执行密码重置要求使用 SSM Agent 版本 2.3.668.0 或更高版本。在控制台中，您可以通过选择 Update SSM Agent（更新 ）来更新托管式节点上的代理。

  如果有新工具添加至 Systems Manager 或者对现有工具进行了更新，则将发布 SSM Agent 的更新版本。无法使用代理的最新版本可能会阻止托管式节点使用 Systems Manager 的各项工具和功能。因此，我们建议您自动完成确保机器上的 SSM Agent 为最新的过程。有关更多信息，请参阅 自动

更新到 SSM Agent。要获得有关 SSM Agent 更新的通知，请在 GitHub 上订阅 SSM Agent 发布说明页面。

未提供密码重置选项 (AWS CLI)

问题：您使用 AWS CLI `start-session` 命令成功连接到托管式节点。您指定了 SSM 文档 `AWS-PasswordReset` 并提供了有效的用户名，但不显示更改密码的提示。

- 解决方案：托管式节点上的 SSM Agent 版本不是最新版本。需要版本 2.3.668.0 或更高版本才能执行密码重置。

  如果有新工具添加至 Systems Manager 或者对现有工具进行了更新，则将发布 SSM Agent 的更新版本。无法使用代理的最新版本可能会阻止托管式节点使用 Systems Manager 的各项工具和功能。因此，我们建议您自动完成确保机器上的 SSM Agent 为最新的过程。有关更多信息，请参阅 自动更新到 SSM Agent。要获得有关 SSM Agent 更新的通知，请在 GitHub 上订阅 SSM Agent 发布说明页面。

未获授权，无法运行 **`ssm:SendCommand`**

问题：您尝试连接到托管式节点以更改密码，但收到错误消息，说明您未获授权在托管式节点上运行 `ssm:SendCommand`。

- 解决方案：您的 IAM policy 必须包括运行 `ssm:SendCommand` 命令的权限。有关信息，请参阅根据标签限制 Run Command 访问。

Session Manager 错误消息

问题：您收到与 Session Manager 相关的错误消息。

- 解决方案：密码重置要求正确地配置 Session Manager。有关更多信息，请参阅 设置 Session Manager 和 排查 Session Manager 问题。

在混合和多云环境中取消注册托管式节点

如果您不再希望使用 AWS Systems Manager 管理本地服务器、边缘设备或虚拟机 (VM)，则可以将其取消注册。取消注册混合激活节点，会将其从 Systems Manager 中的托管式节点列表中删除。AWS Systems Manager由于不再注册在混合激活节点上运行的代理（SSM Agent），因此将无法刷新其授

权令牌。SSM Agent 将休眠，并将其对云中的 Systems Manager 执行 ping 操作的频率减少为每小时一次。Systems Manager 会存储已取消注册的托管式节点的命令历史记录 30 天。

如需重新注册本地服务器、边缘设备或虚拟机，必须使用不同的激活码和激活 ID，不能使用先前注册计算机时用过的激活码和激活 ID。若已达创建时指定的最大激活次数，则不能再使用相同的激活码和激活 ID。

以下过程介绍如何使用 Systems Manager 控制台取消注册混合激活节点。有关如何使用 AWS Command Line Interface 执行此操作的信息，请参阅 deregister-managed-instance。

有关信息，请参阅以下主题：

- 取消注册及重新注册托管式节点（Linux）（Linux）
- 取消注册及重新注册托管式节点（Windows Server）(Windows Server)

取消注册混合激活节点（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择要取消注册的托管式节点旁边的复选框。
4. 选择节点操作、工具、取消注册此托管式节点。
5. 查看取消注册此托管式节点对话框中的信息。如果您批准，请选择取消注册。

通过 Fleet Manager 来使用 OS 文件系统

您可以通过 Fleet Manager（AWS Systems Manager 中的一项工具）来使用托管式节点上的文件系统。使用 Fleet Manager，您可以查看有关存储在附加到托管式节点的卷的目录和文件数据的信息。例如，您可以查看目录和文件的名称、大小、扩展名、拥有者和权限。可从 Fleet Manager 控制台以文本形式预览最多 1 万行的文件数据。您还可以将此功能用于 `tail` 文件。在使用 `tail` 查看文件数据时，最初会显示文件的最后 10 行。随着新的数据行写入文件，视图也将实时更新。因此，您可以从控制台查看日志数据，这可以提高故障排除和系统管理的效率。此外，您可以创建目录以及复制、剪切、粘贴、重命名或删除文件和目录。

我们建议定期创建备份，或为附加到托管式节点的 Amazon Elastic Block Store (Amazon EBS) 卷拍摄快照。复制或剪切并粘贴文件时，目标路径中与新文件或目录同名的现有文件和目录将被替换。如果您替换或修改系统文件和目录，可能会出现严重问题。AWS 不能保证这些问题能得到解决。修改系统文

件需自行承担风险。您需对所有文件与目录更改负责，并确保您有备份。删除或替换文件和目录的操作无法撤销。

> **Note**
>
> Fleet Manager 使用Session Manager（AWS Systems Manager 中的一项工具）来查看文本预览和 `tail` 文件。对于 Amazon Elastic Compute Cloud (Amazon EC2) 实例，附加到托管式实例的实例配置文件必须向 Session Manager 提供权限才能使用此功能。有关向实例配置文件添加 Session Manager 权限的更多信息，请参阅 向现有 IAM 角色添加 Session Manager 权限。

主题

- 使用 Fleet Manager 来查看 OS 文件系统
- 使用 Fleet Manager 来预览 OS 文件
- 使用 Fleet Manager 来查看 OS 文件的末尾内容
- 使用 Fleet Manager 来复制、剪切和粘贴 OS 文件或目录
- 使用 Fleet Manager 来重命名 OS 文件和目录
- 使用 Fleet Manager 来删除 OS 文件和目录
- 使用 Fleet Manager 来创建 OS 目录
- 使用 Fleet Manager 来剪切、复制和粘贴 OS 目录

使用 Fleet Manager 来查看 OS 文件系统

您可以使用 Fleet Manager 来查看 Systems Manager 托管节点上的 OS 文件系统。

要使用 Fleet Manager 来查看 OS 文件系统

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择托管式节点的链接，其中包含要查看的文件系统。
4. 选择工具，文件系统。

使用 Fleet Manager 来预览 OS 文件

您可以使用 Fleet Manager 来预览 OS 上的文本文件。

要使用 Fleet Manager 来预览文本文件

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择托管式节点的链接，其中包含要预览的文件。
4. 选择工具，文件系统。
5. 选择目录（其中包含要预览的文件）的 File name（文件名）。
6. 选择要预览其内容的文件旁边的按钮。
7. 选择操作，以文本形式预览。

使用 Fleet Manager 来查看 OS 文件的末尾内容

您可以使用 Fleet Manager 在托管节点上查看文件的末尾内容。

使用 Fleet Manager 来查看 OS 文件的末尾内容

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择托管式节点的链接，其中包含要查看其末尾内容的文件。
4. 选择工具，文件系统。
5. 选择目录（其中包含要查看其末尾内容的文件）的 File name（文件名）。
6. 选择要查看其末尾内容的文件旁边的按钮。
7. 选择 Actions, Tail file（操作，查看文件末尾内容）。

使用 Fleet Manager 来复制、剪切和粘贴 OS 文件或目录

您可以使用 Fleet Manager 在托管节点上复制、剪切和粘贴 OS 文件。

要使用 Fleet Manager 来复制或剪切和粘贴文件或目录

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择托管式节点的链接，其中包含要复制或者剪切并粘贴的文件。

4. 选择工具，文件系统。
5. 要复制或剪切文件，请选择目录（其中包含要复制或剪切的文件）的 File name（文件名）。要复制或剪切目录，请选择要复制或剪切的目录旁边的按钮，然后继续执行步骤 8。
6. 选择要复制或剪切的文件旁边的按钮。
7. 在 Actions（操作）菜单中，选择 Copy（复制）或 Cut（剪切）。
8. 在 File system（文件系统）视图中，选择要粘贴文件的目录旁边的按钮。
9. 在 Actions（操作）菜单中，选择 Paste（粘贴）。

使用 Fleet Manager 来重命名 OS 文件和目录

您可以使用 Fleet Manager 来重命名您账户中托管节点上的文件和目录。

要使用 Fleet Manager 来重命名文件或目录

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择托管式节点的链接，其中包含要重命名的文件或目录。
4. 选择工具，文件系统。
5. 要重命名文件，请选择目录（其中包含要重命名的文件）的 File name（文件名）。要重命名目录，请选择要重命名的目录旁边的按钮，然后继续执行步骤 8。
6. 选择要对其内容进行重命名的文件旁边的按钮。
7. 选择操作，重命名。
8. 对于文件名，输入文件的新名称，然后选择重命名。

使用 Fleet Manager 来删除 OS 文件和目录

您可以使用 Fleet Manager 来删除您账户中托管节点上的文件和目录。

要使用 Fleet Manager 来删除文件或目录

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择托管式节点的链接，其中包含要删除的文件或目录。

4. 选择工具，文件系统。
5. 要删除文件，请选择目录（其中包含要删除的文件）的 File name（文件名）。要删除目录，请选择要删除的目录旁边的按钮，然后继续执行步骤 7。
6. 选择包含要删除的内容之文件旁边的按钮。
7. 选择操作，删除。

使用 Fleet Manager 来创建 OS 目录

您可以使用 Fleet Manager 来创建您账户中托管节点上的目录。

要使用 Fleet Manager 来创建目录

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择要在其中创建目录的托管式节点的链接。
4. 选择工具，文件系统。
5. 选择要在其中创建新目录的目录的 File name（文件名）。
6. 选择 Create directory（创建目录）。
7. 对于目录名称，输入新目录的名称，然后选择创建目录。

使用 Fleet Manager 来剪切、复制和粘贴 OS 目录

您可以使用 Fleet Manager 来剪切、复制和粘贴您账户中托管节点上的目录。

使用 Fleet Manager 来复制或剪切和粘贴目录

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择托管式节点的链接，其中包含要复制或者剪切并粘贴的文件。
4. 选择工具，文件系统。
5. 选择要复制或剪切的目录旁边的按钮，然后继续执行步骤 8。
6. 在 Actions（操作）菜单中，选择 Copy（复制）或 Cut（剪切）。
7. 在 File system（文件系统）视图中，选择要粘贴文件的目录旁边的按钮。

8. 在 Actions（操作）菜单中，选择 Paste（粘贴）。

监控托管式节点性能

您可以使用 Fleet Manager（AWS Systems Manager 中的一项工具）来实时查看有关托管式节点的性能数据。性能数据可从性能计数器检索。

Fleet Manager 中提供了以下性能计数器：

- CPU 使用率
- 磁盘输入/输出 (I/O) 利用率
- 网络流量
- 内存使用量

> **Note**
>
> Fleet Manager 使用Session Manager（AWS Systems Manager 中的一项工具）来检索性能数据。对于 Amazon Elastic Compute Cloud (Amazon EC2) 实例，附加到托管式实例的实例配置文件必须向 Session Manager 提供权限才能使用此功能。有关向实例配置文件添加 Session Manager 权限的更多信息，请参阅 向现有 IAM 角色添加 Session Manager 权限。

使用 Fleet Manager 查看性能数据

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择要监控其性能的托管式节点旁边的按钮。
4. 请选择查看详细信息。
5. 选择工具，性能计数器。

使用进程

您可以通过 Fleet Manager（AWS Systems Manager 中的一项工具）来使用托管式节点上运行的进程。通过 Fleet Manager，您可以查看有关进程的信息。例如，除句柄和线程外，您还可以查看进程的 CPU 利用率和内存使用情况。使用 Fleet Manager，您可以从控制台启动和终止进程。

> **Note**
>
> Fleet Manager 使用Session Manager（AWS Systems Manager 中的一项工具）来检索进程数据。对于 Amazon Elastic Compute Cloud (Amazon EC2) 实例，附加到托管式实例的实例配置文件必须向 Session Manager 提供权限才能使用此功能。有关向实例配置文件添加 Session Manager 权限的更多信息，请参阅 向现有 IAM 角色添加 Session Manager 权限。

主题

- 使用 Fleet Manager 来查看有关 OS 进程的详细信息
- 使用 Fleet Manager 在托管节点上启动 OS 进程
- 使用 Fleet Manager 来终止 OS 进程

使用 Fleet Manager 来查看有关 OS 进程的详细信息

您可以使用 Fleet Manager 来查看托管节点上进程的详细信息。

使用 Fleet Manager 来查看有关进程的详细信息

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择要查看其进程的节点的链接。
4. 选择 Tools, Processes（工具，进程）。

使用 Fleet Manager 在托管节点上启动 OS 进程

您可以使用 Fleet Manager 在托管节点上启动进程。

使用 Fleet Manager 启动进程

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择要在其上启动进程的托管节点的链接。
4. 选择 Tools, Processes（工具，进程）。

5. 选择 Start new process（启动新进程）。
6. 对于进程名称或完整路径，输入进程名称或可执行文件的完整路径。
7. （可选）对于工作目录，输入要运行进程的目录路径。

使用 Fleet Manager 来终止 OS 进程

使用 Fleet Manager 来终止 OS 进程

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择要在其上启动进程的托管节点的链接。
4. 选择 Tools, Processes（工具，进程）。
5. 选择要终止的进程旁边的按钮。
6. 选择操作，终止进程或操作，终止进程树。

> **Note**
>
> 终止进程树还会终止使用该进程的所有进程和应用程序。

查看托管式节点上的日志

您可以使用 Fleet Manager（AWS Systems Manager 中的一项工具）查看存储在托管式节点上的日志数据。对于 Windows 托管式节点，您可以从控制台查看 Windows 事件日志并复制其详细信息。要帮助您搜索事件，请按 Event level (事件级别)、Event ID (事件 ID)、Event source (事件源) 和 Time created (创建时间) 筛选 Windows 事件日志。您还可以使用查看文件系统的过程查看其他日志数据。有关使用 Fleet Manager 查看文件系统的更多信息，请参阅 通过 Fleet Manager 来使用 OS 文件系统。

使用 Fleet Manager 查看 Windows 事件日志

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择要查看其事件日志的托管式节点旁边的按钮。

4. 请选择查看详细信息。
5. 选择工具，Windows 事件日志。
6. 选择其中包含您要查看的事件的 Log name (日志名称)。
7. 选择您要查看的 Log name (日志名称) 旁边的按钮，然后选择 View events (查看事件)。
8. 选择您要查看的事件旁边的按钮，然后选择 View event details (查看事件详细信息)。
9. （可选）选择 Copy as JSON (作为 JSON 进行复制)，将事件详细信息复制到剪贴板。

使用 Fleet Manager 来管理托管式节点上的 OS 用户账户和组

您可以使用 Fleet Manager（AWS Systems Manager 中的一项工具）来管理托管式节点上的操作系统（OS）用户账户和组。例如，您可以创建和删除用户和组。此外，您还可以查看组成员资格、用户角色和状态等详细信息。

> **Important**
>
> Fleet Manager 使用 Run Command 和Session Manager（都是 AWS Systems Manager 中的工具）进行各种用户管理操作。因此，用户可以向操作系统用户账户授予权限，否则他们将无法授予这些权限。这是因为 AWS Systems Manager 代理 (SSM Agent) 使用根权限 (Linux) 或 SYSTEM 权限 (Windows Server) 在 Amazon Elastic Compute Cloud (Amazon EC2) 实例上运行。有关通过 SSM Agent 限制对根级别命令的访问的更多信息，请参阅 通过 SSM Agent 限制对根级别命令的访问。要限制对此功能的访问，我们建议为您的用户创建 AWS Identity and Access Management (IAM) policy，这些策略仅允许访问您定义的操作。有关为 Fleet Manager 创建 IAM policy 的更多信息，请参阅 控制对 Fleet Manager 的访问。

主题

- 使用 Fleet Manager 来创建 OS 用户或组
- 使用 Fleet Manager 来更新用户或组成员资格
- 使用 Fleet Manager 来删除 OS 用户或组

使用 Fleet Manager 来创建 OS 用户或组

> **Note**
>
> Fleet Manager 可以使用 Session Manager 为新用户设置密码。对于 Amazon EC2 实例，附加到托管式实例的实例配置文件必须向 Session Manager 提供权限才能使用此功能。有关向

实例配置文件添加 Session Manager 权限的更多信息，请参阅 向现有 IAM 角色添加 Session Manager 权限。

您可以使用 Fleet Manager 控制台来执行相同的任务，而不必直接登录到服务器来创建用户帐户或组。

使用 Fleet Manager 来创建 OS 用户账户

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择要在其上创建新用户的托管式节点旁边的按钮。
4. 请选择查看详细信息。
5. 选择工具，用户和组。
6. 选择 Users (用户) 选项卡，然后选择 Create user (创建用户)。
7. 为新用户的 Name (名称) 输入一个值。
8. （推荐）选中 Set password (设置密码) 旁边的复选框。在该过程结束时，系统将提示您为新用户提供密码。
9. 选择 Create user (创建用户)。如果选中此复选框为新用户创建密码，系统将提示您输入密码的值，然后选择 Done (已完成)。如果您指定的密码不符合托管式节点的本地或域策略指定的要求，将返回错误。

使用 Fleet Manager 来创建 OS 组

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择要在其中创建组的托管式节点旁边的按钮。
4. 请选择查看详细信息。
5. 选择工具，用户和组。
6. 选择 Groups (组) 选项卡，然后选择 Create group (创建组)。
7. 为新组的 Name (名称) 输入一个值。
8. （可选）为新组的 Description (描述) 输入一个值。

9. （可选）选择要添加到新组的 Group members (组成员) 中的用户。
10. 选择 Create group (创建组)。

使用 Fleet Manager 来更新用户或组成员资格

您可以使用 Fleet Manager 控制台来执行相同的任务，而不必直接登录到服务器来更新用户帐户或组。

要使用 Fleet Manager 将 OS 用户账户添加到新组

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择要更新的用户账户所在的托管式节点旁边的按钮。
4. 请选择查看详细信息。
5. 选择工具，用户和组。
6. 选择用户选项卡。
7. 选择要更新的用户旁边的按钮。
8. 选择操作，将用户添加到组。
9. 在 Add to group (添加到组) 下，选择要向其添加用户的组。
10. 选择 Add user to group (将用户添加到组)。

要使用 Fleet Manager 来编辑 OS 组的成员资格

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择要更新的组所在的托管式节点旁边的按钮。
4. 请选择查看详细信息。
5. 选择工具，用户和组。
6. 选择组选项卡。
7. 选择要更新的组旁边的按钮。
8. 选择操作，修改组。
9. 在组成员 下，选择要添加或删除的用户。

10. 选择修改组。

使用 Fleet Manager 来删除 OS 用户或组

您可以使用 Fleet Manager 控制台来执行相同的任务，而不必直接登录到服务器来删除用户帐户或组。

使用 Fleet Manager 来删除 OS 用户账户

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择要删除的用户账户所在的托管式节点旁边的按钮。
4. 请选择查看详细信息。
5. 选择用户和组。
6. 选择用户选项卡。
7. 选择要删除的用户旁边按钮。
8. 选择操作，删除本地用户。

要使用 Fleet Manager 来删除 OS 组

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择要删除的组所在的托管式节点旁边的按钮。
4. 请选择查看详细信息。
5. 选择工具，用户和组。
6. 选择组选项卡。
7. 选择要更新的组旁边的按钮。
8. 选择操作，删除本地组。

在托管式节点上管理 Windows 注册表

您可以使用 Fleet Manager（AWS Systems Manager 中的一项工具）来管理 Windows Server 托管式节点上的注册表。您可以从 Fleet Manager 控制台创建、复制、更新及删除注册表项和值。

> **Important**
>
> 在修改注册表之前，我们建议您创建注册表的备份，或者为附加到托管式节点的 Amazon Elastic Block Store (Amazon EBS) 根卷拍摄快照。如果不正确地修改注册表，可能会出现严重问题。这些问题可能需要您重新安装操作系统，或从快照还原节点的根卷。AWS 并不能保证这些问题能得到解决。修改注册表需自行承担风险。所有注册表更改均应由您承担责任，并确保您有备份。

创建 Windows 注册表键或项

使用 Fleet Manager 创建 Windows 注册表键

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择要在其上创建注册表项的托管式节点旁边的按钮。
4. 请选择查看详细信息。
5. 选择工具，Windows 注册表。
6. 通过选择 Registry name（注册表名称），选择要在其中创建新注册表键的配置单元。
7. 选择创建，创建注册表键。
8. 选择要在其中创建新键的注册表项旁边的按钮。
9. 选择 Create registry key (创建注册表键)。
10. 为新注册表键的 Name (名称) 输入一个值，然后选择 Submit (提交)。

使用 Fleet Manager 创建 Windows 注册表项

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择要在其上创建注册表项的实例旁边的按钮。
4. 请选择查看详细信息。
5. 选择工具，Windows 注册表。

6. 选择配置单元，然后通过选择 Registry name（注册表名称），选择要在其中创建新注册表项的后续注册表键。
7. 选择创建，创建注册表项。
8. 为新注册表项的 Name (名称) 输入一个值。
9. 选择要为该注册表项创建的值的 Type (类型)。有关注册表值类型的更多信息，请参阅注册表值的类型。
10. 为新注册表项的 Value (值) 输入一个值。

更新 Windows 注册表项

使用 Fleet Manager 更新 Windows 注册表项

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择要在其上更新注册表项的托管式节点旁边的按钮。
4. 请选择查看详细信息。
5. 选择工具，Windows 注册表。
6. 选择配置单元，然后通过选择 Registry name（注册表名称），选择要更新的后续注册表键。
7. 选择要更新的注册表项旁边的按钮。
8. 选择操作，更新注册表项。
9. 为注册表项的 Value (值) 输入新值。
10. 选择更新。

删除 Windows 注册表项或键

使用 Fleet Manager 删除 Windows 注册表键

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择要在其上删除注册表项的托管式节点旁边的按钮。
4. 选择工具，Windows 注册表。
5. 选择配置单元，然后通过选择 Registry name（注册表名称），选择要删除的后续注册表键。

6. 选择要删除的注册表键旁边的按钮。
7. 选择操作，删除注册表键。

使用 Fleet Manager 删除 Windows 注册表项

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择要在其上删除注册表项的托管式节点旁边的按钮。
4. 请选择查看详细信息。
5. 选择工具，Windows 注册表。
6. 选择配置单元，然后通过选择 Registry name（注册表名称），选择包含要删除的注册表项的后续注册表键。
7. 选择要删除的注册表项旁边的按钮。
8. 选择操作，删除注册表项。

使用默认主机管理配置自动管理 EC2 实例

通过“默认主机管理配置”设置，AWS Systems Manager 可以将 Amazon EC2 实例作为托管式实例自动管理。托管实例是一个配置为与 Systems Manager 一起使用的 EC2 实例。

使用 Systems Manager 管理实例的好处包括以下几点：

- 可以使用 Session Manager 安全地连接到您的 EC2 实例。
- 可以使用 Patch Manager 执行自动补丁扫描。
- 可以使用 Systems Manager 清单查看有关您的实例的详细信息。
- 可以使用 Fleet Manager 追踪和管理实例。
- 可以自动保持 SSM Agent 处于最新状态。

Fleet Manager、Inventory、Patch Manager和Session Manager都是 Systems Manager 中的工具。

使用默认主机管理配置时，无需手动创建 AWS Identity and Access Management（IAM）实例配置文件即可管理 EC2 实例。“默认主机管理配置”会创建并应用默认 IAM 角色，确保 Systems Manager 有权管理已激活该设置的 AWS 账户 和 AWS 区域 中的所有实例。

如果提供的权限不足以满足您的应用场景要求，您还可以向“默认主机管理配置”创建的默认 IAM 角色添加策略。或者，如果您不需要默认 IAM 角色提供的所有功能的权限，可以创建自己的自定义角色和策略。对您为“默认主机管理配置”选择的 IAM 角色所做的任何更改，都适用于相应区域和账户中的所有托管 Amazon EC2 实例。

有关“默认主机管理配置”所用策略的更多信息，请参阅 AWS 托管式策略：AmazonSSMManagedEC2InstanceDefaultPolicy。

实施最低权限访问

本主题中的过程仅由管理员执行。因此，我们建议实施最低权限访问权限，以防止非管理员用户配置或修改“默认主机管理配置”。要查看限制访问“默认主机管理配置”时的策略示例，请参阅本主题后面的“默认主机管理配置”的最低权限策略示例。

> **Important**
>
> 使用“默认主机管理配置”注册的实例的注册信息，存储在本地的 `var/lib/amazon/ssm` 或 `C:\ProgramData\Amazon` 目录中。如果移除这些目录或其中的文件，将导致实例无法使用默认主机管理配置获取连接到 Systems Manager 所需的凭证。在这些情况下，您必须使用 IAM 实例配置文件为实例提供所需的权限，或者重新创建实例。

主题

- 先决条件
- 激活“默认主机管理配置”设置
- 停用“默认主机管理配置”设置
- “默认主机管理配置”的最低权限策略示例

先决条件

要在激活该设置的 AWS 区域 和 AWS 账户 中使用“默认主机管理配置”，就必须满足以下要求。

- 要管理的实例必须使用实例元数据服务版本 2（imdsv2）。

  “默认主机管理配置”不支持实例元数据服务版本 1。有关过渡到 IMDSv2 的信息，请参阅《Amazon EC2 用户指南》中的转换为使用实例元数据服务版本 2。
- 3.2.582.0 版本或更高版本的 SSM Agent 必须安装在实例上。

  有关检查您的实例上安装的 SSM Agent 版本的信息，请参阅 正在检查 SSM Agent 版本号。

有关更新 SSM Agent 的信息，请参阅 自动更新 SSM Agent。

- 作为执行本主题中任务的管理员，您必须具有 GetServiceSetting、ResetServiceSetting 和 UpdateServiceSetting API 操作的权限。此外，您还必须有权获得 `AWSSystemsManagerDefaultEC2InstanceManagementRole` IAM 角色的 `iam:PassRole` 权限。以下是提供这些权限的示例策略。将每个*#######*替换为您自己的信息。

  JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetServiceSetting",
                "ssm:ResetServiceSetting",
                "ssm:UpdateServiceSetting"
            ],
            "Resource": "arn:aws:ssm:us-east-1:111122223333:servicesetting/ssm/
managed-instance/default-ec2-instance-management-role"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::111122223333:role/service-role/
AWSSystemsManagerDefaultEC2InstanceManagementRole",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ssm.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

- 如果已经使用 Systems Manager 将 IAM 实例配置文件附加到要管理的 EC2 实例，则必须从中移除任何允许 `ssm:UpdateInstanceInformation` 操作的权限。SSM Agent 会首先尝试使用实例配

置文件权限，然后才会使用“默认主机管理配置”权限。如果您在自己的 IAM 实例配置文件中允许该 `ssm:UpdateInstanceInformation` 操作，则该实例将不使用“默认主机管理配置”权限。

激活“默认主机管理配置”设置

您可以从 Fleet Manager 控制台或者使用 AWS Command Line Interface 或 AWS Tools for Windows PowerShell 激活“默认主机管理配置”。

您必须在想要通过此设置管理 Amazon EC2 实例的每个区域中，逐个开启“默认主机管理配置”。

在开启“默认主机管理配置”后，实例最多可能需要 30 分钟，就能使用您在下述过程步骤 5 中所选角色的凭证。

激活“默认主机管理配置”（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Fleet Manager。
3. 选择账户管理，配置默认主机管理配置。
4. 打开启用默认主机管理配置。
5. 选择用于为您的实例启用 Systems Manager 工具的 AWS Identity and Access Management（IAM）角色。我们建议使用“默认主机管理配置”提供的默认角色。它包含使用 Systems Manager 管理您的 Amazon EC2 实例所需的最低权限集合。如果您更喜欢使用自定义角色，则该角色的信任策略必须允许 Systems Manager 作为可信实体。
6. 选择配置以完成设置。

激活“默认主机管理配置”（命令行）

1. 在您的本地计算机上创建包含以下信任关系策略的 JSON 文件。

   JSON

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

```
            "Effect":"Allow",
            "Principal":{
                "Service":"ssm.amazonaws.com"
            },
            "Action":"sts:AssumeRole"
        }
    ]
}
```

2. 打开 AWS CLI 或 Tools for Windows PowerShell 并运行以下命令之一，以在您的账户中创建一个服务角色，具体取决于您本地计算机的操作系统类型。将每个*#######*替换为您自己的信息。

   Linux & macOS

   ```
   aws iam create-role \
   --role-name AWSSystemsManagerDefaultEC2InstanceManagementRole \
   --path /service-role/ \
   --assume-role-policy-document file://trust-policy.json
   ```

   Windows

   ```
   aws iam create-role ^
   --role-name AWSSystemsManagerDefaultEC2InstanceManagementRole ^
   --path /service-role/ ^
   --assume-role-policy-document file://trust-policy.json
   ```

   PowerShell

   ```
   New-IAMRole `
   -RoleName "AWSSystemsManagerDefaultEC2InstanceManagementRole" `
   -Path "/service-role/" `
   -AssumeRolePolicyDocument "file://trust-policy.json"
   ```

3. 运行以下命令，将 AmazonSSMManagedEC2InstanceDefaultPolicy 托管策略附加到您新建的角色。将每个*#######*替换为您自己的信息。

   Linux & macOS

   ```
   aws iam attach-role-policy \
   --policy-arn arn:aws:iam::aws:policy/AmazonSSMManagedEC2InstanceDefaultPolicy \
   --role-name AWSSystemsManagerDefaultEC2InstanceManagementRole
   ```

Windows

```
aws iam attach-role-policy ^
--policy-arn arn:aws:iam::aws:policy/AmazonSSMManagedEC2InstanceDefaultPolicy ^
--role-name AWSSystemsManagerDefaultEC2InstanceManagementRole
```

PowerShell

```
Register-IAMRolePolicy `
-PolicyArn "arn:aws:iam::aws:policy/AmazonSSMManagedEC2InstanceDefaultPolicy" `
-RoleName "AWSSystemsManagerDefaultEC2InstanceManagementRole"
```

4. 打开 AWS CLI 或 Tools for Windows PowerShell 并运行以下命令。将每个*#######*替换为您自己的信息。

   Linux & macOS

   ```
   aws ssm update-service-setting \
   --setting-id arn:aws:ssm:region:account-id:servicesetting/ssm/managed-instance/
   default-ec2-instance-management-role \
   --setting-value service-role/AWSSystemsManagerDefaultEC2InstanceManagementRole
   ```

   Windows

   ```
   aws ssm update-service-setting ^
   --setting-id arn:aws:ssm:region:account-id:servicesetting/ssm/managed-instance/
   default-ec2-instance-management-role ^
   --setting-value service-role/AWSSystemsManagerDefaultEC2InstanceManagementRole
   ```

   PowerShell

   ```
   Update-SSMServiceSetting `
   -SettingId "arn:aws:ssm:region:account-id:servicesetting/ssm/managed-instance/
   default-ec2-instance-management-role" `
   -SettingValue "service-role/AWSSystemsManagerDefaultEC2InstanceManagementRole"
   ```

   如果此命令成功，则无任何输出。

5. 运行以下命令，以查看当前的 AWS 账户 和 AWS 区域 中“默认主机管理配置”的当前服务设置。

Linux & macOS

```
aws ssm get-service-setting \
--setting-id arn:aws:ssm:region:account-id:servicesetting/ssm/managed-instance/
default-ec2-instance-management-role
```

Windows

```
aws ssm get-service-setting ^
--setting-id arn:aws:ssm:region:account-id:servicesetting/ssm/managed-instance/
default-ec2-instance-management-role
```

PowerShell

```
Get-SSMServiceSetting `
-SettingId "arn:aws:ssm:region:account-id:servicesetting/ssm/managed-instance/
default-ec2-instance-management-role"
```

此命令会返回如下信息。

```
{
    "ServiceSetting": {
        "SettingId": "/ssm/managed-instance/default-ec2-instance-management-role",
        "SettingValue": "service-role/
AWSSystemsManagerDefaultEC2InstanceManagementRole",
        "LastModifiedDate": "2022-11-28T08:21:03.576000-08:00",
        "LastModifiedUser": "System",
        "ARN": "arn:aws:ssm:us-east-2:-123456789012:servicesetting/ssm/managed-
instance/default-ec2-instance-management-role",
        "Status": "Custom"
    }
}
```

停用“默认主机管理配置”设置

您可以从 Fleet Manager 控制台或者使用 AWS Command Line Interface 或 AWS Tools for Windows PowerShell 停用“默认主机管理配置”。

您必须在不再想要通过此配置管理 Amazon EC2 实例的每个区域中，逐个关闭“默认主机管理配置”设置。在一个区域将其停用，不会在所有区域将其停用。

如果您停用“默认主机管理配置”，并且未将实例配置文件附加到允许访问 Systems Manager 的 Amazon EC2 实例，则这些实例将不再由 Systems Manager 管理。

停用“默认主机管理配置”（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Fleet Manager。
3. 选择账户管理，默认主机管理配置。
4. 关闭启用默认主机管理配置。
5. 选择配置以禁用“默认主机管理配置”。

停用“默认主机管理配置”（命令行）

- 打开 AWS CLI 或 Tools for Windows PowerShell 并运行以下命令。将每个*******替换为您自己的信息。

  Linux & macOS

  ```
  aws ssm reset-service-setting \
  --setting-id arn:aws:ssm:region:account-id:servicesetting/ssm/managed-instance/
  default-ec2-instance-management-role
  ```

  Windows

  ```
  aws ssm reset-service-setting ^
  --setting-id arn:aws:ssm:region:account-id:servicesetting/ssm/managed-instance/
  default-ec2-instance-management-role
  ```

  PowerShell

  ```
  Reset-SSMServiceSetting `
  -SettingId "arn:aws:ssm:region:account-id:servicesetting/ssm/managed-instance/
  default-ec2-instance-management-role"
  ```

“默认主机管理配置”的最低权限策略示例

以下示例策略演示如何防止组织成员更改您账户中的“默认主机管理配置”设置。

适用于 AWS Organizations 的服务控制策略

以下策略演示如何防止您 AWS Organizations 中的非管理员成员更新您的“默认主机管理配置”设置。将每个*#######*替换为您自己的信息。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Deny",
            "Action": [
                "ssm:UpdateServiceSetting",
                "ssm:ResetServiceSetting"
            ],
            "Resource": "arn:aws:ssm:*:*:servicesetting/ssm/managed-instance/
default-ec2-instance-management-role",
            "Condition": {
                "StringNotEqualsIgnoreCase": {
                    "aws:PrincipalTag/job-function": [
                        "administrator"
                    ]
                }
            }
        },
        {
            "Effect": "Deny",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::*:role/service-role/
AWSSystemsManagerDefaultEC2InstanceManagementRole",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ssm.amazonaws.com"
                },
                "StringNotEqualsIgnoreCase": {
                    "aws:PrincipalTag/job-function": [
                        "administrator"
```

```
                    ]
                }
            }
        },
        {
            "Effect": "Deny",
            "Resource": "arn:aws:iam::*:role/service-role/
AWSSystemsManagerDefaultEC2InstanceManagementRole",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:DeleteRole"
            ],
            "Condition": {
                "StringNotEqualsIgnoreCase": {
                    "aws:PrincipalTag/job-function": [
                        "administrator"
                    ]
                }
            }
        }
    ]
}
```

适用于 IAM 主体的策略

以下策略演示如何防止您 AWS Organizations 中的 IAM 组、角色或用户更新您的“默认主机管理配置”设置。将每个*******替换为您自己的信息。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "ssm:UpdateServiceSetting",
                "ssm:ResetServiceSetting"
            ],
            "Resource": "arn:aws:ssm:us-east-1:111122223333:servicesetting/ssm/
managed-instance/default-ec2-instance-management-role"
        },
```

```
        {
            "Effect": "Deny",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:DeleteRole",
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::111122223333:role/service-role/
AWSSystemsManagerDefaultEC2InstanceManagementRole"
        }
    ]
}
```

使用 Remote Desktop 连接到 Windows Server 托管式实例

您可以使用 Fleet Manager（AWS Systems Manager 中的一项工具），通过 Remote Desktop Protocol（RDP）连接到自己的 Windows Server Amazon Elastic Compute Cloud（Amazon EC2）实例。Fleet Manager远程桌面由 Amazon DCV 提供支持，可让您直接从 Systems Manager 控制台安全连接到 Windows Server 实例。在一个浏览器窗口中最多可以同时连接四个连接。

Fleet Manager 远程桌面 API 被命名为 AWS Systems Manager GUI Connect。有关 Systems Manager GUI Connect API 的使用信息，请参阅 AWS Systems Manager GUI ConnectAPI 参考。

目前，您只能对运行 Windows Server 2012 RTM 或更高版本的实例使用远程桌面。远程桌面仅支持英语输入。

Fleet Manager Remote Desktop 是一项仅适用于控制台的服务，不支持通过命令行连接到托管式实例。要通过 Shell 连接到 Windows Server 托管式实例，可以使用 Session Manager（AWS Systems Manager 中的另一项工具）。有关更多信息，请参阅 AWS Systems Manager Session Manager。

> **Note**
>
> RDP 连接的持续时间并不是由 AWS Identity and Access Management（IAM）凭证的持续时间决定的。相反，连接会持续存在，直到达到最大连接持续时间或空闲时间限制为止（以先到者为准）。有关更多信息，请参阅 远程连接持续时间和并发。

有关配置 AWS Identity and Access Management（IAM）权限以允许实例与 Systems Manager 交互的信息，请参阅为 Systems Manager 配置实例权限。

主题

- 设置环境
- 为远程桌面配置 IAM 权限
- 验证远程桌面连接
- 远程连接持续时间和并发
- 使用远程桌面连接到托管式节点
- 查看有关当前和已完成连接的信息

设置环境

在使用远程桌面之前，请确保环境满足以下要求：

- 托管式节点配置

  确保在 Systems Manager 中将您的 Amazon EC2 实例配置为托管式节点。

- 最低 SSM Agent 版本

  验证节点运行的 SSM Agent 版本是否为 3.0.222.0 或更高版本。有关如何检查节点上正在运行哪个代理版本的信息，请参阅 正在检查 SSM Agent 版本号。有关安装或更新 SSM Agent 的信息，请参阅 使用 SSM Agent。

- RDP 端口配置

  要接受远程连接，Windows Server 节点上的 Remote Desktop Services 服务必须使用默认 RDP 端口 3389。这是 AWS 提供的 Amazon Machine Images（AMIs）上的默认配置。系统没有明确要求您打开任何入站端口才能使用远程桌面。

- 键盘功能的 PSReadLine 模块版本

  为确保您的键盘在 PowerShell 中正常运行，请验证运行 Windows Server 2022 的节点是否安装了 PSReadLine 模块版本 2.2.2 或更高版本。如果它们运行的是旧版本，您可以使用以下命令安装所需的版本。

```
Install-PackageProvider -Name NuGet -MinimumVersion 2.8.5.201 -Force
```

  NuGet 程序包提供程序安装完成后，请运行以下命令。

```
Install-Module `
```

```
-Name PSReadLine `
-Repository PSGallery `
-MinimumVersion 2.2.2 -Force
```

- 会话管理器配置

  使用远程桌面之前，您必须先完成会话管理器设置的先决条件。当您使用远程桌面连接到实例时，将应用为 AWS 账户 和 AWS 区域 定义的所有会话首选项。有关更多信息，请参阅 设置 Session Manager。

  > **Note**
  >
  > 如果您使用 Amazon Simple Storage Service（Amazon S3）记录会话管理器活动，则远程桌面连接将在 `bucket_name/Port/stderr` 中生成以下错误。该错误是预期行为，可以安全忽略。
  >
  > ```
  > Setting up data channel with id SESSION_ID failed: failed to create websocket
  >  for datachannel with error: CreateDataChannel failed with no output or
  >  error: createDataChannel request failed: unexpected response from the service
  >  <BadRequest>
  > <ClientErrorMessage>Session is already terminated</ClientErrorMessage>
  > </BadRequest>
  > ```

为远程桌面配置 IAM 权限

除了 Systems Manager 和 Session Manager 所需的 IAM 权限外，您使用的用户或角色还必须拥有启动连接的权限。

启动连接的权限

要在控制台中与 EC2 实例建立 RDP 连接，需要以下权限：

- `ssm-guiconnect:CancelConnection`
- `ssm-guiconnect:GetConnection`
- `ssm-guiconnect:StartConnection`

列出连接的权限

要在控制台中查看连接列表，需要以下权限：

`ssm-guiconnect:ListConnections`

以下是 IAM policy 示例，您可以将这些策略附加到用户或角色以允许与远程桌面进行不同类型的交互。将每个*#######*替换为您自己的信息。

用于连接到 EC2 实例的标准策略

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:GetPasswordData"
            ],
            "Resource": "*"
        },
        {
            "Sid": "SSM",
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeInstanceProperties",
                "ssm:GetCommandInvocation",
                "ssm:GetInventorySchema"
            ],
            "Resource": "*"
        },
        {
            "Sid": "TerminateSession",
            "Effect": "Allow",
            "Action": [
                "ssm:TerminateSession"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "ssm:resourceTag/aws:ssmmessages:session-id": [
                        "${aws:userid}"
                    ]
```

```
                }
            }
        },
        {
            "Sid": "SSMStartSession",
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ec2:*:111122223333:instance/*",
                "arn:aws:ssm:*:111122223333:managed-instance/*",
                "arn:aws:ssm:*::document/AWS-StartPortForwardingSession"
            ],
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:CalledVia": "ssm-guiconnect.amazonaws.com"
                }
            }
        },
        {
            "Sid": "GuiConnect",
            "Effect": "Allow",
            "Action": [
                "ssm-guiconnect:CancelConnection",
                "ssm-guiconnect:GetConnection",
                "ssm-guiconnect:StartConnection",
                "ssm-guiconnect:ListConnections"
            ],
            "Resource": "*"
        }
    ]
}
```

用于连接到具有特定标签的 EC2 实例的策略

> **Note**
>
> 在以下 IAM 策略中，SSMStartSession 部分需要 ssm:StartSession 操作的 Amazon 资源名称（ARN）。如图所示，您指定的 ARN 不需要 AWS 账户 ID。如果您指定账户 ID，则 Fleet Manager 会返回 AccessDeniedException。

位于示例策略下方的 AccessTaggedInstances 部分也要求提供 ssm:StartSession 的 ARN。对于这些 ARN，您需要指定 AWS 账户 ID。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:GetPasswordData"
            ],
            "Resource": "*"
        },
        {
            "Sid": "SSM",
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeInstanceProperties",
                "ssm:GetCommandInvocation",
                "ssm:GetInventorySchema"
            ],
            "Resource": "*"
        },
        {
            "Sid": "SSMStartSession",
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ssm:*::document/AWS-StartPortForwardingSession"
            ],
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:CalledVia": "ssm-guiconnect.amazonaws.com"
                }
            }
```

```
        },
        {
            "Sid": "AccessTaggedInstances",
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ec2:*:111122223333:instance/*",
                "arn:aws:ssm:*:111122223333:managed-instance/*"
            ],
            "Condition": {
                "StringLike": {
                    "ssm:resourceTag/tag key": [
                        "tag value"
                    ]
                }
            }
        },
        {
            "Sid": "GuiConnect",
            "Effect": "Allow",
            "Action": [
                "ssm-guiconnect:CancelConnection",
                "ssm-guiconnect:GetConnection",
                "ssm-guiconnect:StartConnection",
                "ssm-guiconnect:ListConnections"
            ],
            "Resource": "*"
        }
    ]
}
```

用于 AWS IAM Identity Center 用户连接到 EC2 实例的策略

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "SSO",
```

```
            "Effect": "Allow",
            "Action": [
                "sso:ListDirectoryAssociations*",
                "identitystore:DescribeUser"
            ],
            "Resource": "*"
        },
        {
            "Sid": "EC2",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:GetPasswordData"
            ],
            "Resource": "*"
        },
        {
            "Sid": "SSM",
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeInstanceInformation",
                "ssm:DescribeInstanceProperties",
                "ssm:GetCommandInvocation",
                "ssm:GetInventorySchema"
            ],
            "Resource": "*"
        },
        {
            "Sid": "TerminateSession",
            "Effect": "Allow",
            "Action": [
                "ssm:TerminateSession"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "ssm:resourceTag/aws:ssmmessages:session-id": [
                        "${aws:userName}"
                    ]
                }
            }
        },
        {
            "Sid": "SSMStartSession",
```

```
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:instance/*",
                "arn:aws:ssm:*:*:managed-instance/*",
                "arn:aws:ssm:*:*:document/AWS-StartPortForwardingSession"
            ],
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:CalledVia": "ssm-guiconnect.amazonaws.com"
                }
            }
        },
        {
            "Sid": "SSMSendCommand",
            "Effect": "Allow",
            "Action": [
                "ssm:SendCommand"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:instance/*",
                "arn:aws:ssm:*:*:managed-instance/*",
                "arn:aws:ssm:*:*:document/AWSSSO-CreateSSOUser"
            ]
        },
        {
            "Sid": "GuiConnect",
            "Effect": "Allow",
            "Action": [
                "ssm-guiconnect:CancelConnection",
                "ssm-guiconnect:GetConnection",
                "ssm-guiconnect:StartConnection",
                "ssm-guiconnect:ListConnections"
            ],
            "Resource": "*"
        }
    ]
}
```

验证远程桌面连接

建立远程连接时，您可以使用 Windows 凭证或与实例关联的 Amazon EC2 密钥对（.pem 文件）进行身份验证。有关使用密钥对的信息，请参阅《Amazon EC2 用户指南》中的 Amazon EC2 密钥对和 Windows 实例。

或者，如果您已使用 AWS IAM Identity Center 对 AWS Management Console 进行身份验证，您可以在不提供额外凭证的情况下连接到实例。有关允许使用 IAM Identity Center 进行远程连接身份验证的策略示例，请参阅 为远程桌面配置 IAM 权限。

开始前的准备工作

在开始使用远程桌面进行连接之前，请注意以下有关使用 IAM Identity Center 身份验证的条件。

- 远程桌面支持您启用 IAM Identity Center 的同一 AWS 区域 中的节点的 IAM Identity Center 身份验证。
- 远程桌面支持最多 16 个字符的 IAM Identity Center 用户名。
- 远程桌面支持由字母数字字符和以下特殊字符组成的 IAM Identity Center 用户名：. - _

> **Important**
>
> 包含以下字符的 IAM Identity Center 用户名将无法成功连接：+ = ,
> IAM Identity Center 支持用户名中的这些字符，但 Fleet Manager RDP 连接不支持。
> 此外，如果 IAM Identity Center 用户名包含一个或多个 @ 符号，则 Fleet Manager 会忽略第一个 @ 符号及其后面的所有字符，无论 @ 是否引入电子邮件地址的域部分。例如，对于 IAM Identity Center 用户名 `diego_ramirez@example.com`，`@example.com` 部分将被忽略，而 Fleet Manager 的用户名变为 `diego_ramirez`。对于 `diego_r@mirez@example.com`，Fleet Manager 忽略 `@mirez@example.com`，并且 Fleet Manager 的用户名变为 `diego_r`。

- 使用 IAM Identity Center 对连接进行身份验证后，远程桌面会在实例的本地管理员组中创建一个本地 Windows 用户。远程连接结束后，此用户仍然存在。
- 远程桌面不允许对作为 Microsoft Active Directory 域控制器的节点进行 IAM Identity Center 身份验证。
- 尽管远程桌面允许您对已加入 Active Directory 域的节点使用 IAM Identity Center 身份验证，但我们不建议这样做。此身份验证方法向用户授予管理权限，这些权限可能会覆盖域授予的更严格的权限。

IAM Identity Center 身份验证支持的区域

以下 AWS 区域支持使用 IAM Identity Center 身份验证的 Remote Desktop 连接：

- 美国东部（俄亥俄州）(us-east-2)
- 美国东部（弗吉尼亚州北部）(us-east-1)
- 美国西部（北加利福尼亚）(us-west-1)
- 美国西部（俄勒冈州）(us-west-2)
- 非洲（开普敦）(af-south-1)
- 亚太地区（香港）(ap-east-1)
- 亚太地区（孟买）(ap-south-1)
- 亚太地区（东京）(ap-northeast-1)
- 亚太地区（首尔）(ap-northeast-2)
- 亚太地区（大阪）(ap-northeast-3)
- 亚太地区（新加坡）(ap-southeast-1)
- 亚太地区（悉尼）(ap-southeast-2)
- 亚太地区（雅加达）（ap-southeast-3）
- 加拿大（中部）(ca-central-1)
- 欧洲地区（法兰克福）(eu-central-1)
- 欧洲地区（斯德哥尔摩）(eu-north-1)
- 欧洲地区（爱尔兰）(eu-west-1)
- 欧洲（伦敦）(eu-west-2)
- 欧洲地区（巴黎）（eu-west-3）
- 以色列（特拉维夫）（il-central-1）
- 南美洲（圣保罗）（sa-east-1）
- 欧洲（米兰）(eu-south-1)
- 中东（巴林）(me-south-1)
- AWS GovCloud（美国东部）(us-gov-east-1)
- AWS GovCloud（美国西部）(us-gov-west-1)

远程连接持续时间和并发

以下条件适用于活动的远程桌面连接：

- 连接持续时间

  默认情况下，远程桌面连接会在 60 分钟后断开。为防止连接断开，您可以选择在断开连接之前续订会话以重置持续时间计时器。

- 连接超时

  远程桌面连接在闲置超过 10 分钟后会断开连接。

- 连接持久性

  使用远程桌面连接 Windows Server 后，连接会继续，直到达到最大连接持续时间（60 分钟）或空闲超时限制（10 分钟）。连接持续时间并不取决于 AWS Identity and Access Management（IAM）凭证的持续时间。即使 IAM 凭证到期，只要未达到连接持续时间限制，连接仍将继续。使用远程桌面时，如果 IAM 凭证到期，则应立即退出浏览器页面以终止连接。

- 并发连接

  默认情况下，对于相同的 AWS 账户 和 AWS 区域，您同一时间最多可以有 5 个处于活动状态的远程桌面连接。要请求将服务限额增加到最多 25 个并发连接，请参阅《Service Quotas User Guide》中的 Requesting a quota increase。

> **Note**
>
> Windows Server 的标准许可证允许两个并发 RDP 连接。要支持更多连接，必须从 Microsoft 购买额外的客户端访问许可证（CAL），或从 AWS 购买 Microsoft 远程桌面服务许可证。有关补充许可证的更多信息，请参阅以下主题：
>
> - Microsoft 网站上的 Client Access Licenses and Management Licenses
> - 《License Manager User Guide》中的 Use License Manager user-based subscriptions for supported software products

使用远程桌面连接到托管式节点

浏览器对文本的复制/粘贴支持

若使用 Google Chrome 和 Microsoft Edge 浏览器，可以将托管式节点中的文本复制并粘贴到本地计算机，也可以将本地计算机中的文本复制并粘贴到所连接的托管式节点。

若使用 Mozilla Firefox 浏览器，则只能将托管式节点中的文本复制并粘贴到本地计算机，不能将本地计算机中的文本复制到托管式节点。

使用 Fleet Manager 远程桌面连接到托管式节点

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Fleet Manager。
3. 选择您想要连接到的节点。您可以选择复选框或节点名称。
4. 在节点操作菜单中，选择连接远程桌面。
5. 选择首选的 Authentication type（身份验证类型）。如果您选择用户凭据，请输入您要连接的节点上的 Windows 用户账户的用户名和密码。如果您选择密钥对，您可以使用以下方法之一提供身份验证：

   a. 如果您想从本地文件系统中选择与您的实例关联的 PEM 密钥，请选择浏览本地计算机。

      –或者–

   b. 如果要复制 PEM 文件的内容并将其粘贴到提供的字段中，请选择粘贴密钥对内容。
6. 选择 Connect（连接）。
7. 要选择首选的显示分辨率，请在操作菜单中选择分辨率，然后从以下选项中进行选择：

   - 自动适应
   - 1920 x 1080
   - 1400 x 900
   - 1366 x 768
   - 800 x 600

   自动适应选项根据检测到的屏幕尺寸设置分辨率。

查看有关当前和已完成连接的信息

您可以使用 Systems Manager 控制台中的 Fleet Manager 部分来查看有关在您的账户中建立的 RDP 连接的信息。通过一组筛选条件，您可以按照时间范围、特定实例、建立连接的用户和特定状态的连接来缩小所显示的连接列表的范围。控制台还提供选项卡，显示有关所有当前活动连接和所有过去连接的信息。

要查看有关当前和已完成连接的信息

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Fleet Manager。
3. 选择账户管理，连接远程桌面。
4. 选择以下选项卡之一：
   - 活动连接
   - 连接历史记录
5. 要进一步缩小所显示的连接结果列表的范围，请在搜索 () 框中指定一个或多个筛选条件 您也可以输入自由文本搜索词。

在托管式实例上管理 Amazon EBS 卷

Amazon Elastic Block Store（Amazon EBS）提供了块级存储卷，以与 Amazon Elastic Compute Cloud（Amazon EC2）实例一起使用。EBS 卷的行为类似于原始、未格式化的块储存设备。您可以将这些卷作为设备挂载在实例上。

您可以使用 Fleet Manager（AWS Systems Manager 中的一项工具）来管理托管式实例上的 Amazon EBS 卷。例如，您可以初始化 EBS 卷，格式化分区，然后挂载该卷以使其可供使用。

> **Note**
>
> Fleet Manager 目前仅支持 Windows Server 实例的 Amazon EBS 卷管理。

查看 EBS 卷详细信息

使用 Fleet Manager 查看 EBS 卷的详细信息

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择要查看 EBS 卷详细信息的托管式实例旁边的按钮。

4. 请选择查看详细信息。
5. 选择工具，EBS 卷。
6. 要查看 EBS 卷的详细信息，请在卷 ID列中选择其 ID。

初始化和格式化 EBS 卷

使用 Fleet Manager 初始化和格式化 EBS 卷

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择要初始化、格式化和挂载 EBS 卷的托管式实例旁边的按钮。只有当磁盘为空时，才能对 EBS 卷进行初始化。
4. 请选择查看详细信息。
5. 在工具菜单中，选择 EBS 卷。
6. 选择要初始化和格式化的 EBS 卷的旁边的按钮。
7. 选择初始化并格式化。
8. 在分区样式中，选择要用于该 EBS 卷的分区样式。
9. （可选）为该分区选择一个驱动器盘符。
10. （可选）输入一个分区名称来标识该分区。
11. 选择用于整理分区中存储的文件和数据的文件系统。
12. 选择确认以使该 EBS 卷可供使用。确认后，您将无法从 AWS Management Console 更改分区配置，但可以使用 SSH 或 RDP 登录实例来更改分区配置。

访问 Red Hat 知识库门户

如果您是 Red Hat 客户，可以使用 Fleet Manager（AWS Systems Manager 中的一项工具）访问知识库门户。如果您在 AWS 上运行 Red Hat Enterprise Linux (RHEL) 实例或使用 RHEL 服务，您将被视为 Red Hat 客户。知识库门户包括二进制文件、知识共享和社群支持论坛，这些资源仅向获得 Red Hat 许可的客户提供。

除 Systems Manager 和 Fleet Manager 所需的 AWS Identity and Access Management（IAM）权限外，用于访问控制台的用户或角色还必须允许 `rhelkb:GetRhelURL` 操作能够访问知识库门户。

访问 Red Hat 知识库门户

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Fleet Manager。
3. 选择要用于连接到 Red Hat 知识库门户的 RHEL 实例。
4. 选择账户管理、访问 Red Hat 知识库以打开 Red Hat 知识库页面。

如果您使用 AWS 上的 RHEL 运行完全受支持的 RHEL 工作负载，您还可以使用 AWS 凭证，通过 Red Hat 的网站访问 Red Hat 知识库。

排除托管式节点可用性的问题

对于 Run Command、Distributor 和Session Manager等多项 AWS Systems Manager 工具，您可以选择手动选择要在其上运行操作的托管式节点。在这种情况下，指定要手动选择节点后，系统会显示托管式节点列表，您可以在其中运行该操作。

本主题提供的信息可帮助您诊断您已确认正在运行的托管式节点未包含在 Systems Manager 中的托管式节点列表中的原因。

为使节点能由 Systems Manager 管理，并在托管式节点列表中列出，它必须满足三个要求：

- SSM Agent 必须在具有受支持的操作系统的节点上安装和运行。

> **Note**
>
> 将一些 AWS 托管式 Amazon Machine Images（AMIs）配置为启动已预安装 SSM Agent 的实例。（您还可以配置自定义 AMI 以预安装 SSM Agent。）有关更多信息，请参阅 查找预装了 SSM Agent 的 AMIs。

- 对于 Amazon Elastic Compute Cloud (Amazon EC2) 实例，您必须将 AWS Identity and Access Management (IAM) 实例配置文件附加到实例。实例配置文件让实例能够与 Systems Manager 服务进行通信。如果您没有为实例分配实例配置文件，则可以使用混合激活进行注册，但这不是常见用例。
- SSM Agent 必须能够连接到 Systems Manager 端点，以便将其自身注册到服务中。此后，托管式节点必须可用于服务，这一点将通过以下方法来确认：服务每五分钟发送一个信号，以检查实例的运行状况。

- 在托管节点的状态保持 `Connection Lost` 至少 30 天后，该节点可能不再会在 Fleet Manager 控制台中列出。必须首先解决导致连接中断的问题，然后才能将其恢复到列表中。

在确认某个托管节点正在运行后，您可以使用以下命令来检查 SSM Agent 是否已成功注册到 Systems Manager 服务。在成功注册之前，此命令不会返回结果。

Linux & macOS

```
aws ssm describe-instance-associations-status \
    --instance-id instance-id
```

Windows

```
aws ssm describe-instance-associations-status ^
    --instance-id instance-id
```

PowerShell

```
Get-SSMInstanceAssociationsStatus `
    -InstanceId instance-id
```

如果注册成功并且托管式节点现在可用于 Systems Manager 操作，该命令会返回类似于以下内容的结果。

```
{
    "InstanceAssociationStatusInfos": [
        {
            "AssociationId": "fa262de1-6150-4a90-8f53-d7eb5EXAMPLE",
            "Name": "AWS-GatherSoftwareInventory",
            "DocumentVersion": "1",
            "AssociationVersion": "1",
            "InstanceId": "i-02573cafcfEXAMPLE",
            "Status": "Pending",
            "DetailedStatus": "Associated"
        },
        {
            "AssociationId": "f9ec7a0f-6104-4273-8975-82e34EXAMPLE",
            "Name": "AWS-RunPatchBaseline",
            "DocumentVersion": "1",
            "AssociationVersion": "1",
```

```
            "InstanceId": "i-02573cafcfEXAMPLE",
            "Status": "Queued",
            "AssociationName": "SystemAssociationForScanningPatches"
        }
    ]
}
```

如果注册尚未完成或失败，该命令将返回类似于以下内容的结果：

```
{
    "InstanceAssociationStatusInfos": []
}
```

如果命令在 5 分钟左右后未返回结果，请使用以下信息帮助您对托管式节点的问题进行故障排除。

主题

- 解决方案 1：确认 SSM Agent 已安装在托管式节点上，并且正在运行
- 解决方案 2：确认已为实例（仅限 EC2 实例）指定 IAM 实例配置文件
- 解决方案 3：验证并确保服务终端节点的连接性
- 解决方案 4：验证并确保目标操作系统支持
- 解决方案 5：确认您在与 Amazon EC2 实例相同的 AWS 区域 中工作。
- 解决方案 6：验证应用于托管节点上 SSM Agent 的代理配置
- 解决方案 7：在托管式实例上安装 TLS 证书
- 使用 ssm-cli 排除托管节点可用性的故障

解决方案 1：确认 SSM Agent 已安装在托管式节点上，并且正在运行

确保最新版本的 SSM Agent 已安装在托管式节点上，并且正在运行。

要确定 SSM Agent 是否已安装在托管式节点上并且正在运行，请参阅 正在检查 SSM Agent 状态并启动代理。

要在托管式节点上安装或重新安装 SSM Agent，请参阅以下主题：

- 在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent
- 如何在混合 Linux 节点上安装 SSM Agent
- 在适用于 Windows Server 的 EC2 实例上手动安装和卸载 SSM Agent

- 如何在混合 Windows 节点上安装 SSM Agent

解决方案 2：确认已为实例（仅限 EC2 实例）指定 IAM 实例配置文件

对于 Amazon Elastic Compute Cloud (Amazon EC2) 实例，确认实例已配置有允许实例与 Systems Manager API 通信的 AWS Identity and Access Management (IAM) 实例配置文件。此外，还要验证并确保您的用户具有 IAM 信任策略，允许您的用户与 Systems Manager API 通信。

> **Note**
>
> 本地服务器、边缘设备和虚拟机 (VM) 使用 IAM 服务角色而不是实例配置文件。有关更多信息，请参阅在混合和多云环境中创建 Systems Manager 所需的 IAM 服务角色。

确定拥有所需权限的实例配置文件是否附加到 EC2 实例

1. 通过以下网址打开 Amazon EC2 控制台：https://console.aws.amazon.com/ec2/。
2. 在导航窗格中，选择 Instances (实例)。
3. 选择要检查实例配置文件的实例。
4. 在底部窗格中的 Description (描述) 选项卡上，找到 IAM role (IAM 角色)，然后选择角色的名称。
5. 在实例配置文件的角色 Summary (摘要) 页面上的 Permissions (权限) 选项卡上，确保 `AmazonSSMManagedInstanceCore` 列在 Permissions policies (权限策略) 之下。

   如果改为使用自定义策略，请确保其提供的权限与 `AmazonSSMManagedInstanceCore` 相同。

   在控制台中打开 `AmazonSSMManagedInstanceCore`

   有关可以附加到 Systems Manager 的实例配置文件的其他策略的信息，请参阅配置 Systems Manager 所需的实例权限。

解决方案 3：验证并确保服务终端节点的连接性

验证并确保实例可连接到 Systems Manager 服务终端节点。通过为 Systems Manager 创建和配置 VPC 终端节点，或者允许 HTTPS（端口 443）出站流量传输到服务终端节点，来提供此连接性。

对于 Amazon EC2 实例，如果 Virtual Private Cloud (VPC) 配置允许出站流量，则 AWS 区域 的 Systems Manager 服务终端节点可用于注册该实例。但是，如果启动实例的 VPC 配置不允许出站流量，并且您无法更改此配置以允许连接到公有服务终端节点，则必须改成为 VPC 配置接口端点。

有关更多信息，请参阅使用适用于 Systems Manager 的 VPC 端点提高 EC2 实例的安全性。

解决方案 4：验证并确保目标操作系统支持

确认您选择的操作可在您希望看到列出的托管式节点类型上运行。某些 Systems Manager 操作既可仅以 Windows 实例作为目标，也可仅以 Linux 实例作为目标。例如，Systems Manager (SSM) 文档 `AWS-InstallPowerShellModule` 和 `AWS-ConfigureCloudWatch` 只能在 Windows 实例上运行。在 Run a command (运行命令) 页面上，如果选择这些文档之一，并选择 Choose instances manually (手动选择实例)，将仅列出您的 Windows 实例，并且仅有它们可供选择。

解决方案 5：确认您在与 Amazon EC2 实例相同的 AWS 区域 中工作。

Amazon EC2 实例将在特定 AWS 区域 中创建和使用，例如美国东部（俄亥俄）区域 (us-east-2) 或欧洲（爱尔兰）区域 (eu-west-1)。确保您在与要使用的 Amazon EC2 实例相同的 AWS 区域 中工作。有关更多信息，请参阅 AWS Management Console 入门中的选择区域。

解决方案 6：验证应用于托管节点上 SSM Agent 的代理配置

验证应用于托管节点上 SSM Agent 的代理配置正确无误。如果代理配置不正确，则节点将无法连接到所需的服务终端节点，或者 Systems Manager 可能错误识别托管式节点的操作系统。有关更多信息，请参阅配置 SSM Agent 以在 Linux 节点上使用代理和配置 SSM Agent以使用 Windows Server 实例的代理。

解决方案 7：在托管式实例上安装 TLS 证书

必须在与 AWS Systems Manager 结合使用的每个托管式实例上安装传输层安全性协议（TLS）证书。AWS 服务 使用这些证书来加密对其他 AWS 服务的调用。

默认情况下，从任何 Amazon Machine Image (AMI) 创建的每个 Amazon EC2 实例上都已安装 TLS 证书。大多数现代操作系统在其信任存储中包含来自 Amazon Trust Services CA 的所需 TLS 证书。

要验证实例上是否已安装所需的证书，请根据实例的操作系统运行以下命令。务必将 URL 的 *region* 部分替换为托管式实例所在的 AWS 区域。

Linux & macOS

```
curl -L https://ssm.region.amazonaws.com
```

Windows

```
Invoke-WebRequest -Uri https://ssm.region.amazonaws.com
```

命令应返回 `UnknownOperationException` 错误。如果您收到 SSL/TLS 错误消息，可能未安装所需的证书。

如果您发现所需的 Amazon Trust Services CA 证书未安装在您的基本操作系统、从并非由 Amazon 提供的 AMIs 创建的实例上或您自己的本地服务器和虚拟机上，则必须安装并允许来自 Amazon Trust Services 的证书，或使用 AWS Certificate Manager (ACM) 为受支持的集成服务创建和管理证书。

每个托管实例都必须安装以下传输层安全性 (TLS) 证书之一。

- Amazon Root CA 1
- Starfield Services Root Certificate Authority – G2
- Starfield Class 2 Certificate Authority

有关使用 ACM 的信息，请参阅 《AWS Certificate Manager 用户指南》。

如果您的计算环境中的证书由组策略对象 (GPO) 进行托管，则您可能需要将组策略配置为包含其中一个证书。

有关 Amazon Root 和 Starfield 证书的更多信息，请参阅博客文章 How to Prepare for AWS's Move to Its Own Certificate Authority。

使用 **`ssm-cli`** 排除托管节点可用性的故障

`ssm-cli` 是包含在 SSM Agent 安装中的独立命令行工具。在计算机上安装 SSM Agent 3.1.501.0 或更高版本后，可以在该计算机上运行 `ssm-cli` 命令。这些命令的输出有助于确定，计算机是否满足要由 AWS Systems Manager 托管的 Amazon EC2 实例或非 EC2 计算机的最低要求，从而将其添加到 Systems Manager 的托管式节点列表中。（SSM Agent 版本 3.1.501.0 已于 2021 年 11 月发布。）

最低要求

对于要由 AWS Systems Manager 管理并在托管式节点列表中可用的 Amazon EC2 实例或非 EC2 计算机，必须满足三个主要要求：

- SSM Agent 必须在具有受支持的操作系统的计算机上安装和运行。

  将一些适用于 EC2 的 AWS 托管 Amazon Machine Images（AMIs）配置为启动已预安装 SSM Agent 的实例。（您还可以配置自定义 AMI 以预安装 SSM Agent。）有关更多信息，请参阅 查找预装了 SSM Agent 的 AMIs。

- 必须将与 Systems Manager 服务进行通信提供所需权限的 AWS Identity and Access Management（IAM）实例配置文件（适用于 EC2 实例）或服务角色（适用于非 EC2 计算机）附加到该计算机上。
- SSM Agent 必须能够连接到 Systems Manager 端点，以便将其自身注册到服务中。此后，托管节点必须可用于该服务，这一点将通过以下方法来确认：服务每五分钟发送一个信号，以检查托管节点的运行状况。

ssm-cli 中预配置的命令

包含用于收集所需信息的预配置命令，以帮助您诊断您已确认正在运行的计算机未包含在 Systems Manager 中的托管式节点列表中的原因。这些命令会在您指定 `get-diagnostics` 选项时运行。

在计算机上运行以下命令，使用 `ssm-cli` 帮助您排除托管节点可用性的故障。

Linux & macOS

```
ssm-cli get-diagnostics --output table
```

Windows

在 Windows Server 计算机上，您必须先导航到 `C:\Program Files\Amazon\SSM` 目录，才能运行该命令。

```
ssm-cli.exe get-diagnostics --output table
```

PowerShell

在 Windows Server 计算机上，您必须先导航到 `C:\Program Files\Amazon\SSM` 目录，才能运行该命令。

```
.\ssm-cli.exe get-diagnostics --output table
```

该命令以类似于下表的形式返回输出。

> **Note**
>
> 对 ssmmessages、s3、kms、logs 和 monitoring 端点进行的连接检查针对其他可选功能，例如可以录入到 Amazon Simple Storage Service (Amazon S3) 或 Amazon CloudWatch Logs 并使用 AWS Key Management Service (AWS KMS) 加密的 Session Manager。

Linux & macOS

```
[root@instance]# ssm-cli get-diagnostics --output table
##################################################################################################
# Check                                  # Status  # Note
                                        #
##################################################################################################
# EC2 IMDS                               # Success # IMDS is accessible and has
 instance id i-0123456789abcdefa in Region  #
#                                       #         # us-east-2
                                        #
##################################################################################################
# Hybrid instance registration           # Skipped # Instance does not have hybrid
 registration                            #
##################################################################################################
# Connectivity to ssm endpoint           # Success # ssm.us-east-2.amazonaws.com is
 reachable                              #
##################################################################################################
# Connectivity to ec2messages endpoint  # Success # ec2messages.us-
east-2.amazonaws.com is reachable                    #
##################################################################################################
# Connectivity to ssmmessages endpoint  # Success # ssmmessages.us-
east-2.amazonaws.com is reachable                    #
##################################################################################################
# Connectivity to s3 endpoint            # Success # s3.us-east-2.amazonaws.com is
 reachable                               #
##################################################################################################
# Connectivity to kms endpoint           # Success # kms.us-east-2.amazonaws.com is
 reachable                              #
##################################################################################################
# Connectivity to logs endpoint          # Success # logs.us-east-2.amazonaws.com is
 reachable                             #
##################################################################################################
# Connectivity to monitoring endpoint   # Success # monitoring.us-
east-2.amazonaws.com is reachable                     #
```

```
##########################################################################################
# AWS Credentials                        # Success # Credentials are for
                                       #
#                                        #         #
 arn:aws:sts::123456789012:assumed-role/Fullaccess/i-0123456789abcdefa #
#                                        #         # and will expire at 2021-08-17
 18:47:49 +0000 UTC                       #
##########################################################################################
# Agent service                          # Success # Agent service is running and is
 running as expected user               #
##########################################################################################
# Proxy configuration                    # Skipped # No proxy configuration detected
                                       #
##########################################################################################
# SSM Agent version                      # Success # SSM Agent version is 3.0.1209.0,
 latest available agent version is     #
#                                        #         # 3.1.192.0
                                       #
##########################################################################################
```

Windows Server and PowerShell

```
PS C:\Program Files\Amazon\SSM> .\ssm-cli.exe get-diagnostics --output table
##########################################################################################
# Check                                  # Status  # Note
                                     #
##########################################################################################
# EC2 IMDS                               # Success # IMDS is accessible and has
 instance id i-0123456789EXAMPLE in       #
#                                        #         # Region us-east-2
                                     #
##########################################################################################
# Hybrid instance registration           # Skipped # Instance does not have hybrid
 registration                           #
##########################################################################################
# Connectivity to ssm endpoint           # Success # ssm.us-east-2.amazonaws.com is
 reachable                             #
##########################################################################################
# Connectivity to ec2messages endpoint  # Success # ec2messages.us-
east-2.amazonaws.com is reachable                    #
##########################################################################################
# Connectivity to ssmmessages endpoint  # Success # ssmmessages.us-
east-2.amazonaws.com is reachable                    #
```

```
################################################################################################
# Connectivity to s3 endpoint           # Success # s3.us-east-2.amazonaws.com is
 reachable                             #
################################################################################################
# Connectivity to kms endpoint          # Success # kms.us-east-2.amazonaws.com is
 reachable                            #
################################################################################################
# Connectivity to logs endpoint         # Success # logs.us-east-2.amazonaws.com is
 reachable                           #
################################################################################################
# Connectivity to monitoring endpoint   # Success # monitoring.us-
east-2.amazonaws.com is reachable                    #
################################################################################################
# AWS Credentials                       # Success # Credentials are for
                                    #
#                                       #         #
 arn:aws:sts::123456789012:assumed-role/SSM-Role/i-123abc45EXAMPLE  #
#                                       #         #  and will expire at 2021-09-02
 13:24:42 +0000 UTC                   #
################################################################################################
# Agent service                         # Success # Agent service is running and is
 running as expected user            #
################################################################################################
# Proxy configuration                   # Skipped # No proxy configuration detected
                                    #
################################################################################################
# Windows sysprep image state           # Success # Windows image state value is at
 desired value IMAGE_STATE_COMPLETE  #
################################################################################################
# SSM Agent version                     # Success # SSM Agent version is 3.2.815.0,
 latest agent version in us-east-2   #
#                                       #         # is 3.2.985.0
                                    #
################################################################################################
```

下表提供由 ssm-cli 执行的每项检查的其他详细信息。

ssm-cli 诊断检查

Check	详细信息
Amazon EC2 实例元数据服务	指示托管节点是否能够访问元数据服务。测试失败表明，与 http://169.254.169.254 的连接出现问题，这可能由本地路由、代理或操作系统（OS）防火墙和代理配置引起。
混合实例注册	指示是否使用混合激活注册 SSM Agent。
连接到 ssm 端点	指示节点是否能够在 TCP 端口 443 上访问 Systems Manager 的服务端点。测试失败表明，与 https://ssm. *region*.amazonaws.com 的连接出现问题，具体取决于节点所在的 AWS 区域。连接问题可能由 VPC 配置引起，包括安全组、网络访问控制列表、路由表或操作系统防火墙和代理。
连接到 ec2messages 端点	指示节点是否能够在 TCP 端口 443 上访问 Systems Manager 的服务端点。测试失败表明，与 https://ec2messages. *region*.amazonaws.com 的连接出现问题，具体取决于节点所在的 AWS 区域。连接问题可能由 VPC 配置引起，包括安全组、网络访问控制列表、路由表或操作系统防火墙和代理。
连接到 ssmmessages 端点	指示节点是否能够在 TCP 端口 443 上访问 Systems Manager 的服务端点。测试失败表明，与 https://ssmmessages. *region*.amazonaws.com 的连接出现问题，具体取决于节点所在的 AWS 区域。连接问题可能由 VPC 配置引起，包括安全组、网络访问控制列表、路由表或操作系统防火墙和代理。
连接到 s3 端点	指示节点是否能够在 TCP 端口 443 上访问 Amazon Simple Storage Service 的服务端点。测试失败表明，与 https://s3. *region*.amazonaws.com 的连接出现问

Check	详细信息
	题，具体取决于节点所在的 AWS 区域。节点无需连接到此端点，即可在托管式节点列表中显示。
连接到 kms 端点	节点能够在 TCP 端口 443 上访问 AWS Key Management Service 的服务端点。测试失败表明，与 https://kms. *region*.amazonaws.com 的连接出现问题，具体取决于节点所在的 AWS 区域。节点无需连接到此端点，即可在托管式节点列表中显示。
连接到 logs 端点	节点能够在 TCP 端口 443 上访问 Amazon CloudWatch Logs 的服务端点。测试失败表明，与 https://logs. *region*.amazonaws.com 的连接出现问题，具体取决于节点所在的 AWS 区域。节点无需连接到此端点，即可在托管式节点列表中显示。
连接到 monitoring 端点	节点能够在 TCP 端口 443 上访问 Amazon CloudWatch 的服务端点。测试失败表明，与 https://monitoring. *region*.amazonaws.com 的连接出现问题，具体取决于节点所在的 AWS 区域。节点无需连接到此端点，即可在托管式节点列表中显示。
AWS 凭证	指示 SSM Agent 是否有基于附加到计算机的 IAM 实例配置文件（适用于 EC2 实例）或 IAM 服务角色（适用于非 EC2 计算机）所需要的凭证。测试失败表明，没有 IAM 实例配置文件或 IAM 服务角色附加到计算机上，或者其不包含 Systems Manager 所需的权限。

Check	详细信息
代理服务	指示 SSM Agent 服务是否正在运行，以及该服务是否以 root（适用于 Linux 或 macOS）或 SYSTEM（适用于 Windows Server）运行。测试失败表明，SSM Agent 服务未运行或未作为 root 或 SYSTEM 运行。
代理配置	指示 SSM Agent 是否配置为使用代理。
Sysprep 映像状态（仅适用于 Windows）	指示节点上 `Sysprep` 的状态。如果 `Sysprep` 状态是 `IMAGE_STATE_COMPLETE` 之外的值，则 SSM Agent 不会在节点上启动。
SSM Agent 版本	指示是否已安装最新可用版本的 SSM Agent。

AWS Systems Manager 混合激活

要将非 EC2 计算机配置为在混合和多云环境中与 AWS Systems Manager 搭配使用，您需创建混合激活。作为托管式节点支持的非 EC2 计算机类型包括以下几种：

- 您本地的服务器（本地服务器）
- AWS IoT Greengrass 核心设备
- AWS IoT 和非 AWS 边缘设备
- 虚拟机，包括其他云环境中的虚拟机

当您运行 create-activation 命令启动混合激活过程时，您会在命令响应中收到激活码和 ID。然后，您需要在命令中包含激活码和 ID 以在计算机上安装 SSM Agent，如在混合 Linux 节点上安装 SSM Agent 中的步骤 3 和在混合 Windows Server 节点上安装 SSM Agent 中的步骤 4 所述。

此激活过程适用于除 AWS IoT Greengrass 核心设备之外的所有非 EC2 计算机类型。有关为 Systems Manager 配置 AWS IoT Greengrass 核心设备的信息，请参阅 使用 Systems Manager 管理边缘设备。

> **Note**
>
> 目前不为非 EC2 macOS 计算机提供支持。

关于 Systems Manager 实例套餐

AWS Systems Manager 提供了标准实例层和高级实例层。两者都支持混合和多云环境中的托管式节点。通过标准实例套餐，每个 AWS 区域 内每个 AWS 账户 最多可以注册 1000 台计算机。如果您需要在一个账户和区域中注册超过 1000 台计算机，则使用高级实例套餐。在高级实例套餐中，您可以根据需要创建许多托管式节点。为 Systems Manager 配置的所有托管式节点均按使用量付费。有关启用高级实例套餐的更多信息，请参阅 打开高级实例套餐。有关定价的更多信息，请参阅 AWS Systems Manager 定价。

请注意以下关于标准实例层和高级实例层的附加信息：

- 通过高级实例，您还可以使用 AWS Systems Manager Session Manager 连接到混合和多云环境中的非 EC2 节点。Session Manager 提供了对您的实例的交互式 Shell 访问。有关更多信息，请参阅 AWS Systems Manager Session Manager。
- 标准实例配额也适用于使用 Systems Manager 本地激活的 Amazon EC2 实例（但这不是常见情况）。
- 要修补 Microsoft 在虚拟机 (VM) 本地实例上发布的应用程序，请激活高级实例套餐。使用高级实例套餐需支付费用。修补 Microsoft 在 Amazon Elastic Compute Cloud (Amazon EC2) 实例上发布的应用程序不收取额外费用。有关更多信息，请参阅 Microsoft 在 Windows Server 上发布的补丁应用程序。

AWS Systems Manager 清单

可以通过 AWS Systems Manager Inventory 了解您的 AWS 计算环境。您可以使用 Inventory 从托管式节点收集元数据。您可以将此元数据存储在中央 Amazon Simple Storage Service（Amazon S3）存储桶中，然后使用内置工具查询数据并快速确定哪些节点正在运行软件和软件策略所需的配置，以及需要更新哪些节点。您可以使用一个一键式过程在所有托管式节点上配置 Inventory。您还可以通过使用 Amazon Athena 配置及查看多个 AWS 区域 和 AWS 账户 的清单数据。要开始使用“清单”，请打开 Systems Manager 控制台。在导航窗格中，选择 Inventory (清单)。

如果 Systems Manager Inventory 收集的预配置元数据类型未满足您的需求，则您可以创建自定义清单。自定义清单是一个简单的 JSON 文件，包含您提供并添加到特定目录中的托管式节点的信息。当 Systems Manager Inventory 收集数据时，它将捕获此自定义清单数据。例如，如果您运行一个大型数

据中心，您可以将每个服务器的机架位置指定为自定义清单。然后，在您查看其他清单数据时，则可以查看机架空间数据。

> **Important**
>
> Systems Manager Inventory 仅从托管式节点收集元数据。Inventory 不会访问专有信息或数据。

下表描述了可以使用 Systems Manager Inventory 收集的数据的类型。该表还介绍了定位节点的不同产品以及您可以指定的收集间隔。

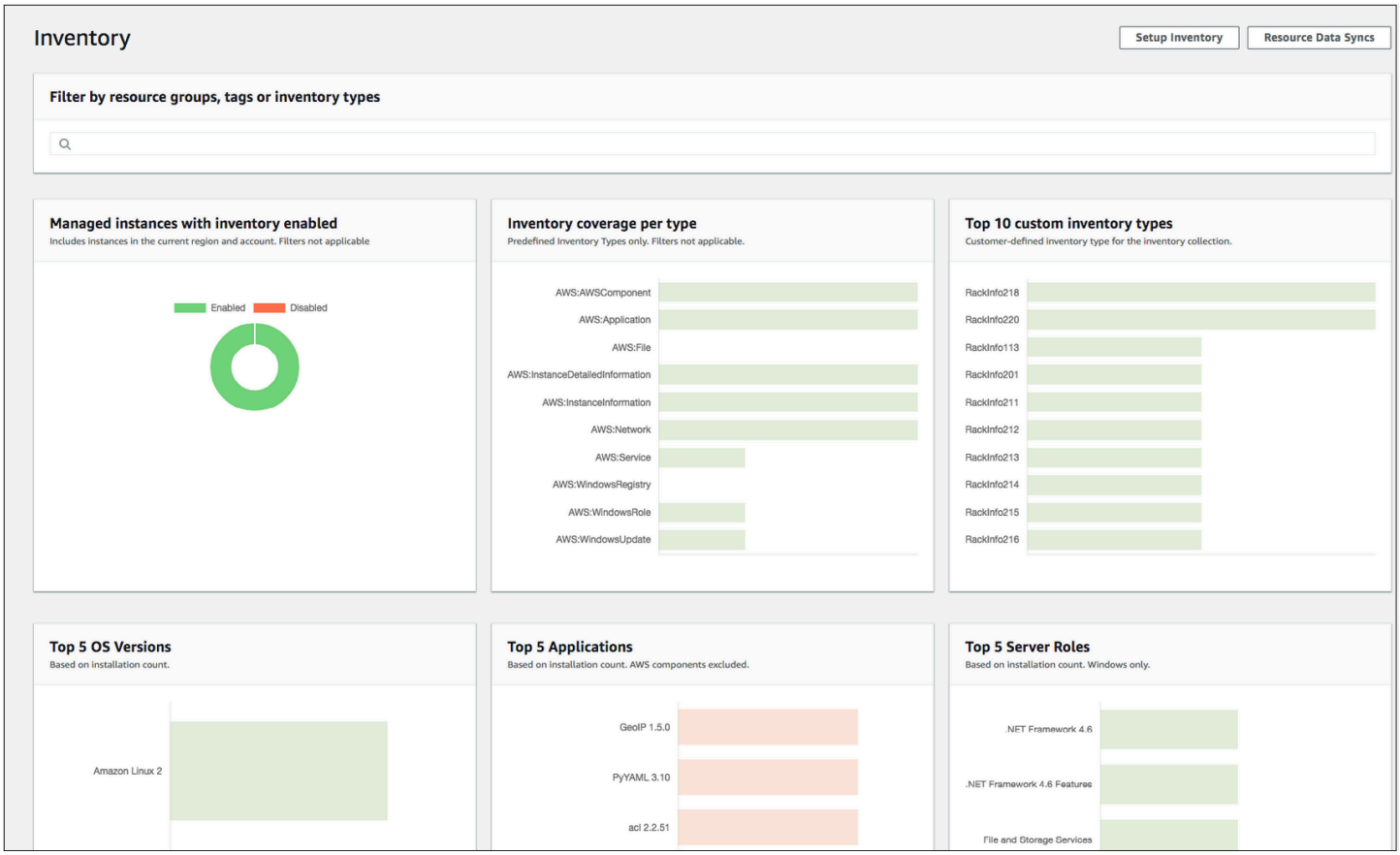
配置	详细信息
元数据类型	您可以配置 Inventory 以收集以下类型的数据： • 应用程序：应用程序名称、发布者和版本等。 • AWS 组件：Amazon EC2 驱动程序、代理、版本等。 • 文件：名称、大小、版本、安装日期、修改时间和上次访问时间等。 • 网络配置：IP 地址、MAC 地址、DNS、网关、子网掩码等。 • Windows 更新：补丁 ID、安装者、安装日期等。 • 实例详情：CPUModel、CPUCores、CPUs、CPUSpeedMHz、CPUSockets、CPUHyperThreadEnabled、OSServicePacketc。 • 服务：名称、显示名称、状态、相关服务、服务类型、启动类型等。 • 标签：分配给您节点的标签。 • Windows 注册表：注册表项路径、值名称、值类型和值。 • Windows 角色：名称、显示名称、路径、功能类型、安装状态等。

配置	详细信息
	• 自定义清单：按照 使用自定义清单 中所述分配给托管式节点的元数据。 **Note** 要查看清单收集的所有元数据的列表，请参阅 清单收集的元数据。
要定位的节点	您可以选择清点 AWS 账户 中的所有托管式节点、单独选择节点，或者使用标签将节点组设为目标。有关从所有托管节点收集清单数据的更多信息，请参阅 清点 AWS 账户 中的所有托管式节点。
何时收集信息	您可以按分钟、小时和天数指定收集间隔。最短收集间隔为 30 分钟。

Note

根据所收集的数据量，系统可能需要几分钟将数据报告到您指定的输出中。收集信息后，数据会通过安全 HTTPS 通道发送到一个只能从您的AWS 账户访问的纯文本 AWS 存储。

您可以在 Inventory（清单）页面上查看 Systems Manager 控制台中的数据，该页面包括几个预定义的卡，可以帮助您查询数据。



> **Note**
>
> Inventory 卡可以自动筛选掉状态为 Terminated (已终止) 和 Stopped (已停止) 的 Amazon EC2 托管实例。对于本地托管式节点和 AWS IoT Greengrass 核心设备托管式节点，Inventory 卡会自动筛选掉状态为 Terminated（已终止）的节点。

如果您创建资源数据同步，以同步所有数据并将其存储在单个 Amazon S3 存储桶中，则可以在 Inventory Detail View (Inventory 详细视图) 页面上详细查看这些数据。有关更多信息，请参阅 查询多个区域和账户的清单数据。

EventBridge 支持

支持此 Systems Manager 工具作为 Amazon EventBridge 规则中的一个事件类型。有关信息，请参阅 使用 Amazon EventBridge 监控 Systems Manager 事件和 引用：Amazon EventBridge 事件模式和 Systems Manager 类型。

内容

- 了解有关 Systems Manager Inventory 的更多信息
- 设置 Systems Manager Inventory
- 配置清单收集
- 查询多个区域和账户的清单数据
- 使用筛选条件查询清单收集
- 聚合清单数据
- 使用自定义清单
- 查看清单历史记录和变更跟踪
- 停止数据收集和删除清单数据
- 将自定义清单元数据分配给某个托管式节点
- 使用 AWS CLI 配置清单数据收集
- 演练：使用资源数据同步聚合清单数据
- 对 Systems Manager Inventory 的问题进行故障排除

了解有关 Systems Manager Inventory 的更多信息

当您配置 AWS Systems Manager Inventory 时，您可以指定要收集的元数据的类型、应从中收集元数据的托管式节点以及元数据收集的计划。这些配置将与您的 AWS 账户 一起保存为 AWS Systems Manager State Manager 关联。关联就是配置。

> **Note**
>
> Inventory 仅收集元数据。它不会收集任何个人数据或专有数据。

主题

- 清单收集的元数据
- 使用文件和 Windows 注册表清单

清单收集的元数据

以下示例显示了由每个 AWS Systems Manager Inventory 插件收集的元数据的完整列表。

```
{
    "typeName": "AWS:InstanceInformation",
```

```
    "version": "1.0",
    "attributes":[
      { "name": "AgentType",                                "dataType" : "STRING"},
      { "name": "AgentVersion",                             "dataType" : "STRING"},
      { "name": "ComputerName",                             "dataType" : "STRING"},
      { "name": "InstanceId",                               "dataType" : "STRING"},
      { "name": "IpAddress",                                "dataType" : "STRING"},
      { "name": "PlatformName",                             "dataType" : "STRING"},
      { "name": "PlatformType",                             "dataType" : "STRING"},
      { "name": "PlatformVersion",                          "dataType" : "STRING"},
      { "name": "ResourceType",                             "dataType" : "STRING"},
      { "name": "AgentStatus",                              "dataType" : "STRING"},
      { "name": "InstanceStatus",                           "dataType" : "STRING"}
    ]
  },
  {
    "typeName" : "AWS:Application",
    "version": "1.1",
    "attributes":[
      { "name": "Name",              "dataType": "STRING"},
      { "name": "ApplicationType",   "dataType": "STRING"},
      { "name": "Publisher",         "dataType": "STRING"},
      { "name": "Version",           "dataType": "STRING"},
      { "name": "Release",           "dataType": "STRING"},
      { "name": "Epoch",             "dataType": "STRING"},
      { "name": "InstalledTime",     "dataType": "STRING"},
      { "name": "Architecture",      "dataType": "STRING"},
      { "name": "URL",               "dataType": "STRING"},
      { "name": "Summary",           "dataType": "STRING"},
      { "name": "PackageId",         "dataType": "STRING"}
    ]
  },
  {
    "typeName" : "AWS:File",
    "version": "1.0",
    "attributes":[
      { "name": "Name",              "dataType": "STRING"},
      { "name": "Size",          "dataType": "STRING"},
      { "name": "Description",       "dataType": "STRING"},
      { "name": "FileVersion",       "dataType": "STRING"},
      { "name": "InstalledDate",     "dataType": "STRING"},
      { "name": "ModificationTime",  "dataType": "STRING"},
      { "name": "LastAccessTime",    "dataType": "STRING"},
      { "name": "ProductName",       "dataType": "STRING"},
```

```
      { "name": "InstalledDir",       "dataType": "STRING"},
      { "name": "ProductLanguage",    "dataType": "STRING"},
      { "name": "CompanyName",        "dataType": "STRING"},
      { "name": "ProductVersion",       "dataType": "STRING"}
    ]
  },
  {
    "typeName": "AWS:AWSComponent",
    "version": "1.0",
    "attributes":[
      { "name": "Name",               "dataType": "STRING"},
      { "name": "ApplicationType",    "dataType": "STRING"},
      { "name": "Publisher",          "dataType": "STRING"},
      { "name": "Version",            "dataType": "STRING"},
      { "name": "InstalledTime",      "dataType": "STRING"},
      { "name": "Architecture",       "dataType": "STRING"},
      { "name": "URL",                "dataType": "STRING"}
    ]
  },
  {
    "typeName": "AWS:WindowsUpdate",
    "version":"1.0",
    "attributes":[
      { "name": "HotFixId",           "dataType": "STRING"},
      { "name": "Description",        "dataType": "STRING"},
      { "name": "InstalledTime",      "dataType": "STRING"},
      { "name": "InstalledBy",        "dataType": "STRING"}
    ]
  },
  {
    "typeName": "AWS:Network",
    "version":"1.0",
    "attributes":[
      { "name": "Name",               "dataType": "STRING"},
      { "name": "SubnetMask",         "dataType": "STRING"},
      { "name": "Gateway",            "dataType": "STRING"},
      { "name": "DHCPServer",         "dataType": "STRING"},
      { "name": "DNSServer",          "dataType": "STRING"},
      { "name": "MacAddress",         "dataType": "STRING"},
      { "name": "IPV4",               "dataType": "STRING"},
      { "name": "IPV6",               "dataType": "STRING"}
    ]
  },
  {
```

```
    "typeName": "AWS:PatchSummary",
    "version":"1.0",
    "attributes":[
      { "name": "PatchGroup",                       "dataType": "STRING"},
      { "name": "BaselineId",                       "dataType": "STRING"},
      { "name": "SnapshotId",                       "dataType": "STRING"},
      { "name": "OwnerInformation",                 "dataType": "STRING"},
      { "name": "InstalledCount",                   "dataType": "NUMBER"},
      { "name": "InstalledPendingRebootCount",      "dataType": "NUMBER"},
      { "name": "InstalledOtherCount",              "dataType": "NUMBER"},
      { "name": "InstalledRejectedCount",           "dataType": "NUMBER"},
      { "name": "NotApplicableCount",               "dataType": "NUMBER"},
      { "name": "UnreportedNotApplicableCount",     "dataType": "NUMBER"},
      { "name": "MissingCount",                     "dataType": "NUMBER"},
      { "name": "FailedCount",                      "dataType": "NUMBER"},
      { "name": "OperationType",                    "dataType": "STRING"},
      { "name": "OperationStartTime",               "dataType": "STRING"},
      { "name": "OperationEndTime",                 "dataType": "STRING"},
      { "name": "InstallOverrideList",              "dataType": "STRING"},
      { "name": "RebootOption",                     "dataType": "STRING"},
      { "name": "LastNoRebootInstallOperationTime", "dataType": "STRING"},
      { "name": "ExecutionId",                      "dataType": "STRING",
   "isOptional": "true"},
      { "name": "NonCompliantSeverity",             "dataType": "STRING",
   "isOptional": "true"},
      { "name": "SecurityNonCompliantCount",        "dataType": "NUMBER",
   "isOptional": "true"},
      { "name": "CriticalNonCompliantCount",        "dataType": "NUMBER",
   "isOptional": "true"},
      { "name": "OtherNonCompliantCount",           "dataType": "NUMBER",
   "isOptional": "true"}
    ]
  },
  {
    "typeName": "AWS:PatchCompliance",
    "version":"1.0",
    "attributes":[
      { "name": "Title",                      "dataType": "STRING"},
      { "name": "KBId",                       "dataType": "STRING"},
      { "name": "Classification",             "dataType": "STRING"},
      { "name": "Severity",                   "dataType": "STRING"},
      { "name": "State",                      "dataType": "STRING"},
      { "name": "InstalledTime",              "dataType": "STRING"}
    ]
```

```
 },
 {
   "typeName": "AWS:ComplianceItem",
   "version":"1.0",
   "attributes":[
     { "name": "ComplianceType",              "dataType": "STRING",
"isContext": "true"},
     { "name": "ExecutionId",                 "dataType": "STRING",
"isContext": "true"},
     { "name": "ExecutionType",               "dataType": "STRING",
"isContext": "true"},
     { "name": "ExecutionTime",               "dataType": "STRING",
"isContext": "true"},
     { "name": "Id",                          "dataType": "STRING"},
     { "name": "Title",                       "dataType": "STRING"},
     { "name": "Status",                      "dataType": "STRING"},
     { "name": "Severity",                    "dataType": "STRING"},
     { "name": "DocumentName",                "dataType": "STRING"},
     { "name": "DocumentVersion",             "dataType": "STRING"},
     { "name": "Classification",              "dataType": "STRING"},
     { "name": "PatchBaselineId",             "dataType": "STRING"},
     { "name": "PatchSeverity",               "dataType": "STRING"},
     { "name": "PatchState",                  "dataType": "STRING"},
     { "name": "PatchGroup",                  "dataType": "STRING"},
     { "name": "InstalledTime",               "dataType": "STRING"},
     { "name": "InstallOverrideList",         "dataType": "STRING",
"isOptional": "true"},
     { "name": "DetailedText",                "dataType": "STRING",
"isOptional": "true"},
     { "name": "DetailedLink",                "dataType": "STRING",
"isOptional": "true"},
     { "name": "CVEIds",                      "dataType": "STRING",
"isOptional": "true"}
   ]
 },
 {
   "typeName": "AWS:ComplianceSummary",
   "version":"1.0",
   "attributes":[
     { "name": "ComplianceType",               "dataType": "STRING"},
     { "name": "PatchGroup",                   "dataType": "STRING"},
     { "name": "PatchBaselineId",              "dataType": "STRING"},
     { "name": "Status",                       "dataType": "STRING"},
     { "name": "OverallSeverity",              "dataType": "STRING"},
```

```
      { "name": "ExecutionId",                    "dataType": "STRING"},
      { "name": "ExecutionType",                  "dataType": "STRING"},
      { "name": "ExecutionTime",                  "dataType": "STRING"},
      { "name": "CompliantCriticalCount",         "dataType": "NUMBER"},
      { "name": "CompliantHighCount",             "dataType": "NUMBER"},
      { "name": "CompliantMediumCount",           "dataType": "NUMBER"},
      { "name": "CompliantLowCount",              "dataType": "NUMBER"},
      { "name": "CompliantInformationalCount",    "dataType": "NUMBER"},
      { "name": "CompliantUnspecifiedCount",      "dataType": "NUMBER"},
      { "name": "NonCompliantCriticalCount",      "dataType": "NUMBER"},
      { "name": "NonCompliantHighCount",          "dataType": "NUMBER"},
      { "name": "NonCompliantMediumCount",        "dataType": "NUMBER"},
      { "name": "NonCompliantLowCount",           "dataType": "NUMBER"},
      { "name": "NonCompliantInformationalCount", "dataType": "NUMBER"},
      { "name": "NonCompliantUnspecifiedCount",   "dataType": "NUMBER"}
    ]
  },
  {
    "typeName": "AWS:InstanceDetailedInformation",
    "version":"1.0",
    "attributes":[
      { "name": "CPUModel",                   "dataType": "STRING"},
      { "name": "CPUCores",                   "dataType": "NUMBER"},
      { "name": "CPUs",                       "dataType": "NUMBER"},
      { "name": "CPUSpeedMHz",                "dataType": "NUMBER"},
      { "name": "CPUSockets",                 "dataType": "NUMBER"},
      { "name": "CPUHyperThreadEnabled",      "dataType": "STRING"},
      { "name": "OSServicePack",              "dataType": "STRING"}
    ]
   },
   {
     "typeName": "AWS:Service",
     "version":"1.0",
     "attributes":[
       { "name": "Name",                      "dataType": "STRING"},
       { "name": "DisplayName",               "dataType": "STRING"},
       { "name": "ServiceType",               "dataType": "STRING"},
       { "name": "Status",                    "dataType": "STRING"},
       { "name": "DependentServices",         "dataType": "STRING"},
       { "name": "ServicesDependedOn",        "dataType": "STRING"},
       { "name": "StartType",                 "dataType": "STRING"}
     ]
    },
    {
```

```
      "typeName": "AWS:WindowsRegistry",
      "version":"1.0",
      "attributes":[
        { "name": "KeyPath",                          "dataType": "STRING"},
        { "name": "ValueName",                        "dataType": "STRING"},
        { "name": "ValueType",                        "dataType": "STRING"},
        { "name": "Value",                            "dataType": "STRING"}
      ]
    },
    {
      "typeName": "AWS:WindowsRole",
      "version":"1.0",
      "attributes":[
        { "name": "Name",                           "dataType": "STRING"},
        { "name": "DisplayName",                    "dataType": "STRING"},
        { "name": "Path",                           "dataType": "STRING"},
        { "name": "FeatureType",                    "dataType": "STRING"},
        { "name": "DependsOn",                      "dataType": "STRING"},
        { "name": "Description",                    "dataType": "STRING"},
        { "name": "Installed",                      "dataType": "STRING"},
        { "name": "InstalledState",                 "dataType": "STRING"},
        { "name": "SubFeatures",                    "dataType": "STRING"},
        { "name": "ServerComponentDescriptor",      "dataType": "STRING"},
        { "name": "Parent",                         "dataType": "STRING"}
      ]
    },
    {
      "typeName": "AWS:Tag",
      "version":"1.0",
      "attributes":[
        { "name": "Key",                      "dataType": "STRING"},
        { "name": "Value",                    "dataType": "STRING"}
      ]
    },
    {
      "typeName": "AWS:ResourceGroup",
      "version":"1.0",
      "attributes":[
        { "name": "Name",                   "dataType": "STRING"},
        { "name": "Arn",                    "dataType": "STRING"}
      ]
    },
    {
      "typeName": "AWS:BillingInfo",
```

```
    "version": "1.0",
    "attributes": [
      { "name": "BillingProductId",       "dataType": "STRING"}
    ]
  }
```

Note

- 对于 `"typeName": "AWS:InstanceInformation"`，`InstanceStatus` 可以是以下情况之一：Active (活动)、ConnectionLost (连接已丢失)、Stopped (已停止)、Terminated (已终止)。
- 随着 2.5 版本的发布，RPM 包管理器将 Serial 属性替换成了 Epoch。与 Serial 一样，Epoch 属性是一个单调递增整数。当您使用 `AWS:Application` 类型进行清点时，Epoch 的值越大就意味着版本越新。如果 Epoch 值相同或为空，请使用 Version 或 Release 属性值来确定较新版本。
- 某些元数据在 Linux 实例中不可用。具体而言，对于“typeName”：“AWS:Network”，Linux 实例尚不支持以下元数据类型。但在 Windows 中受支持。
  - { "name": "SubnetMask", "dataType": "STRING"} ,
  - { "name": "DHCPServer", "dataType": "STRING"} ,
  - { "name": "DNSServer", "dataType": "STRING"} ,
  - { "name": "Gateway", "dataType": "STRING"} ,

使用文件和 Windows 注册表清单

借助 AWS Systems Manager Inventory 清单可以搜索和清点 Windows、Linux 及 macOS 操作系统上的文件。您还可以搜索并清点 Windows 注册表。

文件：您可以收集关于文件的元数据信息，包括文件名称、文件创建时间、文件上次修改和访问时间以及文件大小等等。要开始收集文件清单，您需要指定要执行清点的文件路径、用于定义要清点的文件类型的一个或多个模式，以及是否应以递归的方式遍历路径。Systems Manager 将清点与模式相匹配的指定路径中的文件的所有文件元数据。清单文件使用以下参数输入。

```
{
"Path": string,
"Pattern": array[string],
"Recursive": true,
```

```
"DirScanLimit" : number // Optional
}
```

- 路径：您要清点文件的目录路径。对于 Windows，您可以使用 %PROGRAMFILES% 等环境变量，前提是该变量要映射到单个目录路径。例如，如果您使用映射到多个目录路径的 %PATH%，则清单会引发错误。
- 模式：确定文件的一组模式。
- 递归：指示清单是否应以递归方式遍历目录的布尔值。
- DirScanLimit：指定要扫描多少目录的可选值。使用此参数可以将对托管式节点的性能影响降至最低。清单最多扫描 5000 个目录。

> **Note**
>
> 清单在所有指定路径中收集最多 500 个文件的元数据。

下面是一些在执行文件清单时如何指定参数的示例。

- 在 Linux 和 macOS 上，将收集 /home/ec2-user 目录（不包括所有子目录）中的 .sh 文件的元数据。

```
[{"Path":"/home/ec2-user","Pattern":["*.sh", "*.sh"],"Recursive":false}]
```

- 在 Windows 上，会以递归方式收集程序文件夹 (包括子目录) 中的所有“.exe”文件的元数据。

```
[{"Path":"C:\Program Files","Pattern":["*.exe"],"Recursive":true}]
```

- 在 Windows 上，会收集指定日志模式的元数据。

```
[{"Path":"C:\ProgramData\Amazon","Pattern":["*amazon*.log"],"Recursive":true}]
```

- 在执行递归集合时会限制目录计数。

```
[{"Path":"C:\Users","Pattern":["*.ps1"],"Recursive":true, "DirScanLimit": 1000}]
```

Windows 注册表：您可以收集 Windows 注册表项和值。您可以选择一个键路径并以递归方式收集所有键和值。您还可以收集特定路径的特定注册表项及其值。清单会收集键路径、名称、类型和值。

```
{
"Path": string,
"Recursive": true,
"ValueNames": array[string] // optional
}
```

- 路径：注册表项的路径。
- 递归：指示清单是否应以递归方式遍历注册表路径的布尔值。
- ValueNames：执行注册表项的清单的一组值名称。如果使用此参数，Systems Manager 仅将清点指定路径的指定值名称。

> **Note**
>
> 清单针对所有指定路径收集最多 250 个注册表项值。

下面是一些在执行 Windows 注册表清单时如何指定参数的示例。

- 以递归方式针对特定路径收集所有键和值。

```
[{"Path":"HKEY_LOCAL_MACHINE\SOFTWARE\Amazon","Recursive": true}]
```

- 针对特定路径收集所有键和值（已关闭递归搜索）。

```
[{"Path":"HKEY_LOCAL_MACHINE\SOFTWARE\Intel\PSIS\PSIS_DECODER", "Recursive": false}]
```

- 使用 `ValueNames` 选项收集特定键。

```
{"Path":"HKEY_LOCAL_MACHINE\SOFTWARE\Amazon\MachineImage","ValueNames":["AMIName"]}
```

设置 Systems Manager Inventory

在您使用 AWS Systems Manager Inventory 收集有关托管式节点上运行的应用程序、服务、AWS 组件等对象的元数据之前，我们建议您配置资源数据同步，以将清单数据的存储集中在单个 Amazon Simple Storage Service (Amazon S3) 存储桶中。我们还建议您配置清单事件的 Amazon EventBridge 监控。这些过程将使查看和管理清单数据和收集变得更加轻松。

主题

- 为 Inventory 创建资源数据同步
- 使用 EventBridge 监控 Inventory 事件

为 Inventory 创建资源数据同步

本主题介绍如何为 AWS Systems Manager Inventory 设置和配置资源数据同步。有关 Systems Manager Explorer 的资源数据同步的信息，请参阅 设置 Systems Manager Explorer 以显示来自多个账户和区域的数据。

关于资源数据同步

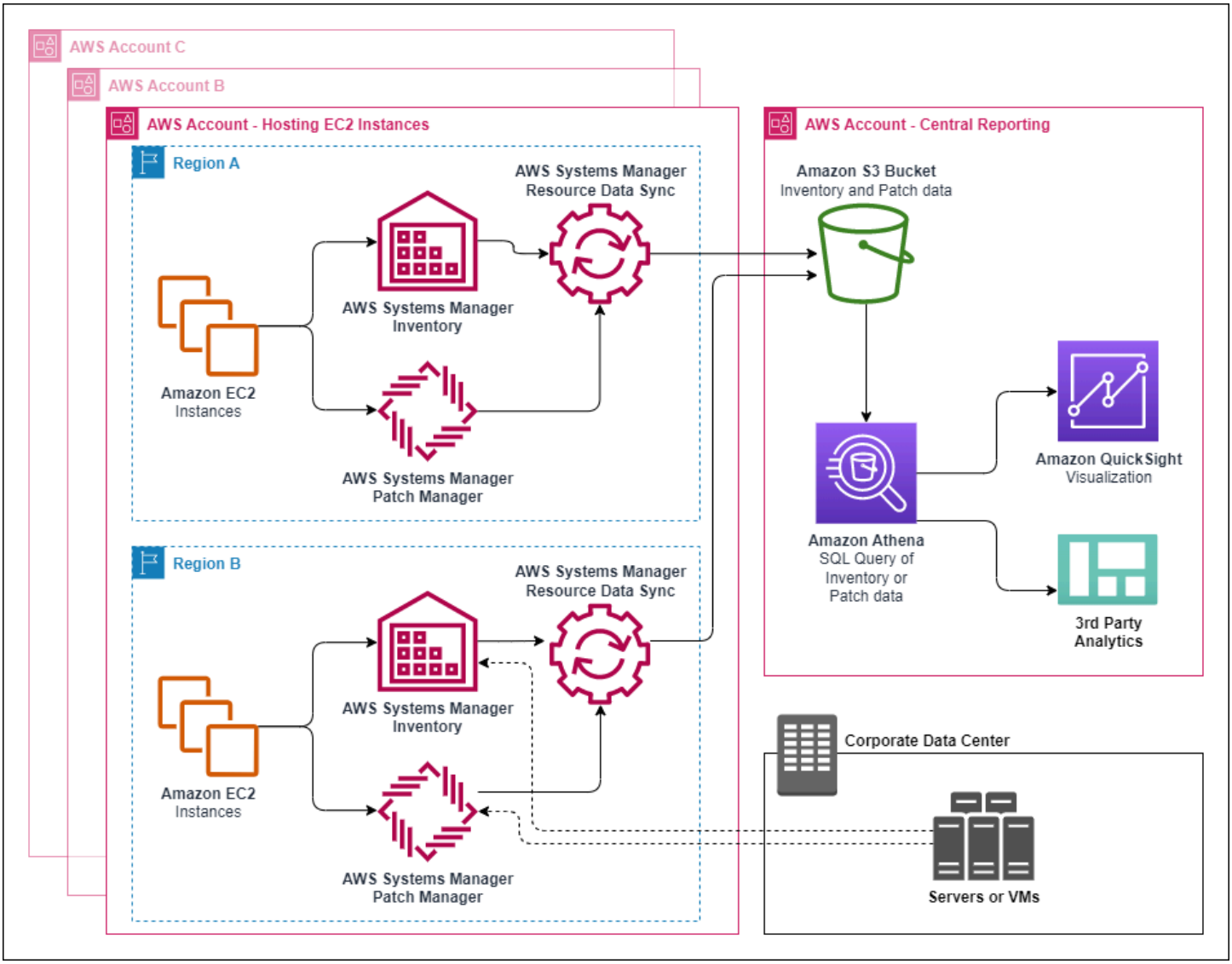
您可以使用 Systems Manager 资源数据同步，将从所有托管式节点收集到的清单数据发送到单个 Amazon Simple Storage Service（Amazon S3）存储桶。在收集新的清单数据时，资源数据同步自动更新集中式数据。在所有清单数据存储在目标 Amazon S3 存储桶中后，您可以使用 Amazon Athena 和 Amazon QuickSight 等服务查询和分析聚合数据。

例如，假设您将清单配置为收集有关操作系统（OS）和 150 个托管式节点机群上运行的应用程序的数据。其中某些节点位于本地数据中心内，而某些则在跨多个 AWS 区域 的 Amazon Elastic Compute Cloud（Amazon EC2）中运行。如果您没有配置资源数据同步，则需要手动收集为每个托管式节点收集的清单数据，或者必须创建脚本以收集该信息。然后您需要将数据传输到应用程序中，以便运行查询和分析数据。

通过使用资源数据同步，您可以执行一次性操作以同步所有托管式节点中的所有清单数据。在成功创建同步后，Systems Manager 会创建所有清单数据的基准，并将其保存在目标 Amazon S3 存储桶中。在收集新的清单数据时，Systems Manager 将自动更新 Amazon S3 存储桶中的数据。然后您就可以快速且经济高效地将数据移植到 Amazon Athena 和 Amazon QuickSight。

图 1 显示了资源数据同步如何将来自混合和多云环境中的 Amazon EC2 和其他计算机类型的清单数据聚合到目标 Amazon S3 存储桶。该图还显示了如何为多个 AWS 账户 和 AWS 区域 执行资源数据同步。

图 1：为多个 AWS 账户 和 AWS 区域 执行资源数据同步



如果删除一个托管式节点，则资源数据同步会保留已删除节点的清单文件。但是，对于正在运行的节点，在创建新文件并将它们写入 Amazon S3 存储桶时，资源数据同步会自动覆盖旧清单文件。如果要随着时间推移跟踪清单变化，您可以使用 AWS Config 服务跟踪 `SSM:ManagedInstanceInventory` 资源类型。有关更多信息，请参见 AWS Config 入门。

可以按照本节中的过程，使用 Amazon S3 和 AWS Systems Manager 控制台为 Inventory 创建资源数据同步。您也可以使用 AWS CloudFormation 创建或删除资源数据同步。要使用 AWS CloudFormation，请将 `AWS::SSM::ResourceDataSync` 资源添加到您的 AWS CloudFormation 模板。有关信息，请参阅以下文档资源：

- 适用于 AWS Systems Manager 中的资源数据同步的 AWS CloudFormation 资源（博客）
- 《AWS CloudFormation 用户指南》中的使用 AWS CloudFormation 模板

> **Note**
>
> 可以使用 AWS Key Management Service (AWS KMS) 加密 Amazon S3 存储桶中的清单数据。有关如何使用 AWS Command Line Interface (AWS CLI) 创建加密同步以及如何使用 Amazon Athena 和 Amazon QuickSight 中的集中式数据的示例，请参阅 演练：使用资源数据同步聚合清单数据。

开始前的准备工作

在创建资源数据同步之前，请使用以下过程创建中央 Amazon S3 存储桶，以存储聚合清单数据。该过程介绍如何分配存储桶策略，以便 Systems Manager 能将来自多个账户的清单数据写入存储桶。如果您已有一个要用于聚合清单数据以进行资源数据同步的 Amazon S3 存储桶，则必须在以下过程中将该存储桶配置为使用该策略。

> **Note**
>
> 如果指定的 Amazon S3 存储桶被配置为使用对象锁定，则 Systems Manager Inventory 无法将数据添加到该存储桶。请验证并确保您为资源数据同步创建或选择的 Amazon S3 存储桶未被配置为使用 Amazon S3 对象锁定。有关更多信息，请参阅《Amazon Simple Storage Service 用户指南》中的 Amazon S3 对象锁定的工作原理。

为资源数据同步创建和配置 Amazon S3 存储桶

1. 通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 创建用来存储聚合清单数据的存储桶。有关更多信息，请参阅《Amazon Simple Storage Service 用户指南》中的创建存储桶。请记下存储桶名称和创建此存储桶的 AWS 区域。
3. 选择权限选项卡，然后选择存储桶策略。
4. 将下面的存储桶策略复制并粘贴到策略编辑器中。将 *amzn-s3-demo-bucket* 替换为您创建的 S3 存储桶的名称。将 *account_ID_number* 替换为有效的 AWS 账户 ID 号码。

   JSON

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

```
        {
            "Sid": "SSMBucketPermissionsCheck",
            "Effect": "Allow",
            "Principal": {
                "Service": "ssm.amazonaws.com"
            },
            "Action": "s3:GetBucketAcl",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket"
        },
        {
            "Sid": " SSMBucketDelivery",
            "Effect": "Allow",
            "Principal": {
                "Service": "ssm.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/*/
accountid=account_ID_number/*",
                "arn:aws:s3:::amzn-s3-demo-bucket/*/
accountid=account_ID_number/*",
                "arn:aws:s3:::amzn-s3-demo-bucket/*/
accountid=account_ID_number/*",
                "arn:aws:s3:::amzn-s3-demo-bucket/*/
accountid=account_ID_number/*"
            ],
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "bucket-owner-full-control",
                    "aws:SourceAccount": "ID_number"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:ssm:*:111122223333:resource-
data-sync/*"
                }
            }
        }
    ]
}
```

5. 保存您的更改。

为 Inventory 创建资源数据同步

按照以下过程，使用 Systems Manager 控制台为 Systems Manager Inventory 创建资源数据同步。有关如何使用 AWS CLI 创建资源数据同步的信息，请参阅 使用 AWS CLI 配置清单数据收集。

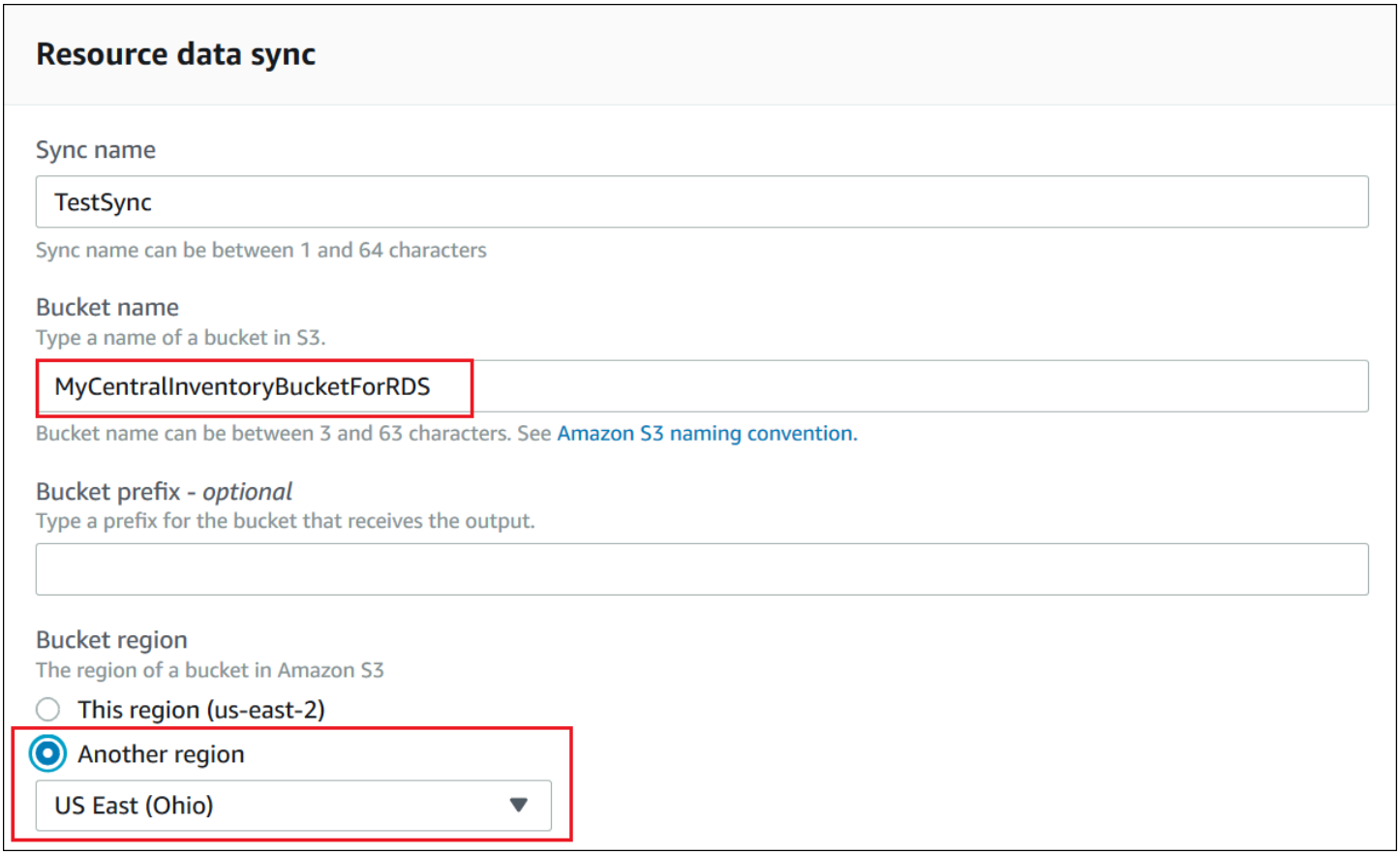
创建资源数据同步

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Fleet Manager。
3. 在账户管理菜单中，选择资源数据同步。
4. 选择创建资源数据同步。
5. 在同步名称字段中，输入同步配置的名称。
6. 在存储桶名称字段中，输入您使用为资源数据同步创建和配置 Amazon S3 存储桶过程创建的 Amazon S3 存储桶的名称。
7. （可选）在存储桶前缀字段中，输入 Amazon S3 存储桶前缀（子目录）的名称。
8. 在存储桶区域字段中，如果您创建的 Amazon S3 存储桶位于当前的 AWS 区域 中，请选择此区域。如果存储桶位于其他 AWS 区域 中，则请选择其他区域，然后输入该区域的名称。

   > **Note**
   >
   > 如果同步和目标 Amazon S3 存储桶位于不同区域中，您可能需要支付数据传输费用。有关更多信息，请参阅 Amazon S3 定价。

9. （可选）在 KMS 密钥 ARN 字段中，键入或粘贴 KMS 密钥 ARN，以加密 Amazon S3 中的清单数据。
10. 选择创建。

要同步来自多个 AWS 区域的清单数据，您必须在每个区域中创建一个资源数据同步。在要收集清单数据并将其发送到中央 Amazon S3 存储桶的每个 AWS 区域 中重复此过程。当您在每个区域中创建同步时，请在存储桶名称字段中指定中央 Amazon S3 存储桶。然后，使用存储桶区域选项选择要在其中创建中央 Amazon S3 存储桶的区域，如下面的屏幕截图中所示。下次关联运行以收集清单数据时，Systems Manager 会将数据存储在中央 Amazon S3 存储桶中。



为 AWS Organizations 中定义的多个账户创建清单资源数据同步

您可以将来自 AWS Organizations 中定义的 AWS 账户的清单数据同步到中央 Amazon S3 存储桶。完成以下过程后，清单数据将同步到中央存储桶中的各个 Amazon S3 密钥前缀。每个键前缀均代表一个不同的 AWS 账户 ID。

开始前的准备工作

在开始之前，请验证并确保您已在 AWS Organizations 中设置和配置了多个 AWS 账户。有关更多信息，请参阅《AWS Organizations 用户指南》中的。

另外，请注意，您必须为 AWS Organizations 中定义的每个 AWS 区域 和 AWS 账户 创建基于组织的资源数据同步。

创建中央 Amazon S3 存储桶

可以使用以下过程创建中央 Amazon S3 存储桶，以存储聚合清单数据。该过程介绍如何分配存储桶策略，以便 Systems Manager 能将来自您的 AWS Organizations 账户 ID 的清单数据写入存储桶。如果您已有一个要用于聚合清单数据以进行资源数据同步的 Amazon S3 存储桶，则必须在以下过程中将该存储桶配置为使用该策略。

为 AWS Organizations 中定义的多个账户的资源数据同步创建和配置 Amazon S3 存储桶

1. 通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 创建用来存储聚合清单数据的存储桶。有关更多信息，请参阅《Amazon Simple Storage Service 用户指南》中的创建存储桶。请记下存储桶名称和创建此存储桶的 AWS 区域。
3. 选择权限选项卡，然后选择存储桶策略。
4. 将下面的存储桶策略复制并粘贴到策略编辑器中。将 *amzn-s3-demo-bucket* 和 *organization-id* 分别替换为您创建的 Amazon S3 存储桶的名称和有效的 AWS Organizations 账户 ID。

   或者，将 *bucket-prefix* 替换为 Amazon S3 前缀（子目录）的名称。如果您未创建前缀，则从以下策略的 ARN 中删除 *bucket-prefix/*。

   JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "SSMBucketPermissionsCheck",
      "Effect": "Allow",
      "Principal": {
        "Service": "ssm.amazonaws.com"
      },
      "Action": "s3:GetBucketAcl",
      "Resource": "arn:aws:s3:::S3_bucket_name"
    },
    {
      "Sid": " SSMBucketDelivery",
      "Effect": "Allow",
      "Principal": {
        "Service": "ssm.amazonaws.com"
      },
      "Action": "s3:PutObject",
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/bucket-prefix/*/accountid=*/*"
      ],
      "Condition": {
        "StringEquals": {
          "s3:x-amz-acl": "bucket-owner-full-control",
```

```
            "aws:SourceOrgID": "organization-id"
                    }
      }
    },
    {
      "Sid": " SSMBucketDeliveryTagging",
      "Effect": "Allow",
      "Principal": {
        "Service": "ssm.amazonaws.com"
      },
      "Action": "s3:PutObjectTagging",
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/bucket-prefix/*/accountid=*/*"
      ]
    }
  ]
}
```

为 AWS Organizations 中定义的账户创建清单资源数据同步

以下过程介绍如何使用 AWS CLI 为 AWS Organizations 中定义的账户创建资源数据同步。您必须使用 AWS CLI 执行此任务。您还必须为 AWS Organizations 中定义的每个 AWS 区域 和 AWS 账户 执行此过程。

为 AWS Organizations 中定义的账户创建资源数据同步 (AWS CLI)

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关更多信息，请参阅安装或更新 AWS CLI 的最新版本。
2. 运行以下命令，验证并确保您没有任何其他基于 AWS Organizations 的资源数据同步。可以有多个标准同步，包括多个标准同步和一个基于组织的同步。但是只能有一个基于组织的资源数据同步。

   ```
   aws ssm list-resource-data-sync
   ```

   如果该命令返回了另一个基于组织的资源数据同步，则您必须将其删除，或选择不创建新资源数据同步。
3. 运行以下命令，为 AWS Organizations 中定义的账户创建资源数据同步。对于 amzn-s3-demo-bucket，请指定您在本主题的前面创建的 Amazon S3 存储桶的名称。如果您为存储桶创建了前缀（子目录），则为 *prefix-name* 指定此信息。

```
aws ssm create-resource-data-sync --sync-name name --s3-
destination "BucketName=amzn-s3-demo-bucket,Prefix=prefix-
name,SyncFormat=JsonSerDe,Region=AWS ##, for example us-
east-2,DestinationDataSharing={DestinationDataSharingType=Organization}"
```

4. 为您希望在其中将数据同步到中央 Amazon S3 存储桶的每个 AWS 区域 和 AWS 账户 重复步骤 2 和步骤 3。

管理资源数据同步

每个 AWS 账户 在每个 AWS 区域 可以有 5 个资源数据同步。您可以使用 AWS Systems Manager Fleet Manager 控制台来管理您的资源数据同步。

查看资源数据同步

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Fleet Manager。
3. 在账户管理下拉菜单中，选择资源数据同步。
4. 从表中选择资源数据同步，然后选择查看详细信息以查看有关资源数据同步的信息。

删除资源数据同步

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Fleet Manager。
3. 在账户管理下拉菜单中，选择资源数据同步。
4. 从表中选择资源数据同步，然后选择删除。

使用 EventBridge 监控 Inventory 事件

您可以在 Amazon EventBridge 中配置规则来创建事件，以响应 AWS Systems Manager Inventory 资源状态更改。EventBridge 支持针对以下 Inventory 状态更改的事件。将尽最大努力发送所有事件。

Custom inventory type deleted for a specific instance（针对特定实例删除的自定义清单类型）：如果配置了用于监控此事件的规则，EventBridge 将在删除特定托管式节点上的自定义清单类型时创建一个事件。EventBridge 将针对每个自定义清单类型的每个节点发送一个事件。以下是一个示例事件模式。

```
{
    "timestampMillis": 1610042981103,
    "source": "SSM",
    "account": "123456789012",
    "type": "INVENTORY_RESOURCE_STATE_CHANGE",
    "startTime": "Jan 7, 2021 6:09:41 PM",
    "resources": [
        {
            "arn": "arn:aws:ssm:us-east-1:123456789012:managed-instance/i-12345678"
        }
    ],
    "body": {
        "action-status": "succeeded",
        "action": "delete",
        "resource-type": "managed-instance",
        "resource-id": "i-12345678",
        "action-reason": "",
        "type-name": "Custom:MyCustomInventoryType"
    }
}
```

Custom inventory type deleted event for all instances（所有实例的自定义清单类型删除事件）：如果配置了用于监控此事件的规则，EventBridge 将在删除适用于所有托管式节点的自定义清单类型时创建一个事件。以下是一个示例事件模式。

```
{
    "timestampMillis": 1610042904712,
    "source": "SSM",
    "account": "123456789012",
    "type": "INVENTORY_RESOURCE_STATE_CHANGE",
    "startTime": "Jan 7, 2021 6:08:24 PM",
    "resources": [

    ],
    "body": {
        "action-status": "succeeded",
        "action": "delete-summary",
        "resource-type": "managed-instance",
        "resource-id": "",
        "action-reason": "The delete for type name Custom:SomeCustomInventoryType
 was completed. The deletion summary is: {\"totalCount\":1,\"remainingCount\":0,
\"summaryItems\":[{\"version\":\"1.1\",\"count\":1,\"remainingCount\":0}]}",
```

```
        "type-name": "Custom:MyCustomInventoryType"
    }
}
```

PutInventory call with old schema version event (使用旧架构版本事件进行 PutInventory 调用)：如果配置了用于监控此事件的规则，EventBridge 将在使用低于当前架构的架构版本进行 PutInventory 调用时创建一个事件。此事件适用于所有清单类型。以下是一个示例事件模式。

```
{
    "timestampMillis": 1610042629548,
    "source": "SSM",
    "account": "123456789012",
    "type": "INVENTORY_RESOURCE_STATE_CHANGE",
    "startTime": "Jan 7, 2021 6:03:49 PM",
    "resources": [
        {
            "arn": "arn:aws:ssm:us-east-1:123456789012:managed-instance/i-12345678"
        }
    ],
    "body": {
        "action-status": "failed",
        "action": "put",
        "resource-type": "managed-instance",
        "resource-id": "i-01f017c1b2efbe2bc",
        "action-reason": "The inventory item with type name
 Custom:MyCustomInventoryType was sent with a disabled schema verison 1.0. You must
 send a version greater than 1.0",
        "type-name": "Custom:MyCustomInventoryType"
    }
}
```

有关如何配置 EventBridge 以监控这些事件的信息，请参阅 为 Systems Manager 事件配置 EventBridge。

配置清单收集

本节介绍了如何使用 Systems Manager 控制台在一个或多个托管式节点上配置 AWS Systems Manager Inventory 收集。有关如何使用 AWS Command Line Interface (AWS CLI) 配置清单收集的示例，请参阅 使用 AWS CLI 配置清单数据收集。

在配置清单收集时，应该首先创建 AWS Systems Manager State Manager 关联。Systems Manager 将在关联运行时收集清单数据。如果您不首先创建关联，并尝试使用 AWS Systems Manager Run

Command（举例来说）调用 `aws:softwareInventory` 插件，则系统将返回以下错误：`The aws:softwareInventory plugin can only be invoked via ssm-associate.`

> **Note**
>
> 如果您为一个托管式节点创建多个清单关联，请注意以下行为：
>
> - 可为每个节点分配一个以所有节点为目标的清单关联 (--targets "Key=InstanceIds,Values=*")。
> - 还可以为每个节点分配一个特定关联，该关联使用标签键/值对或 AWS 资源组。
> - 如果为某个节点分配了多个清单关联，则对于尚未运行的关联，状态将显示为 Skipped（已跳过）。最近运行的关联将显示清单关联的实际状态。
> - 如果为某个节点分配了多个清单关联，并且每个关联都使用标签键/值对，则这些清单关联将因标签冲突而无法在该节点上运行。关联仍会在没有标签键/值冲突的节点上运行。

开始前的准备工作

配置清单收集前，请完成以下任务。

- 更新要清点的节点上的 AWS Systems Manager SSM Agent。通过运行最新版本的 SSM Agent，可确保您能够收集所有支持的清单类型的元数据。有关如何使用 State Manager 更新 SSM Agent 的信息，请参阅 演练：使用 AWS CLI 自动更新 SSM Agent。
- 验证您是否已在混合和多云环境中完成 Amazon Elastic Compute Cloud（Amazon EC2）实例以及非 EC2 计算机的设置要求。有关信息，请参阅为 AWS Systems Manager 设置托管节点。
- 对于 Microsoft Windows 节点，确认已使用 Windows PowerShell 3.0（或更高版本）配置了您的节点。SSM Agent 使用 PowerShell 中的 `ConvertTo-Json` cmdlet 将 Windows 更新清单数据转换为所需格式。
- （可选）创建资源数据同步，以将清单数据集中存储在 Amazon S3 存储桶中。然后，在收集新的清单数据时，资源数据同步将自动更新集中式数据。有关更多信息，请参阅 演练：使用资源数据同步聚合清单数据。
- (可选) 创建用于收集自定义清单的 JSON 文件。有关更多信息，请参阅 使用自定义清单。

清点 AWS 账户 中的所有托管式节点

您可以通过创建全局清单关联，清点 AWS 账户 中的所有托管式节点。全局清单关联执行以下操作：

- 自动将全局清单配置（关联）应用到 AWS 账户 中的所有现有托管式节点。应用和运行全局清单关联时，将跳过已具有清单关联的托管式节点。跳过某个节点时，详细状态消息将说明 `Overridden By Explicit Inventory Association`。全局关联跳过了这些节点，但在它们运行所分配的清单关联时，仍会报告清单。
- 将在您的 AWS 账户 中创建的新节点自动添加到全局清单关联。

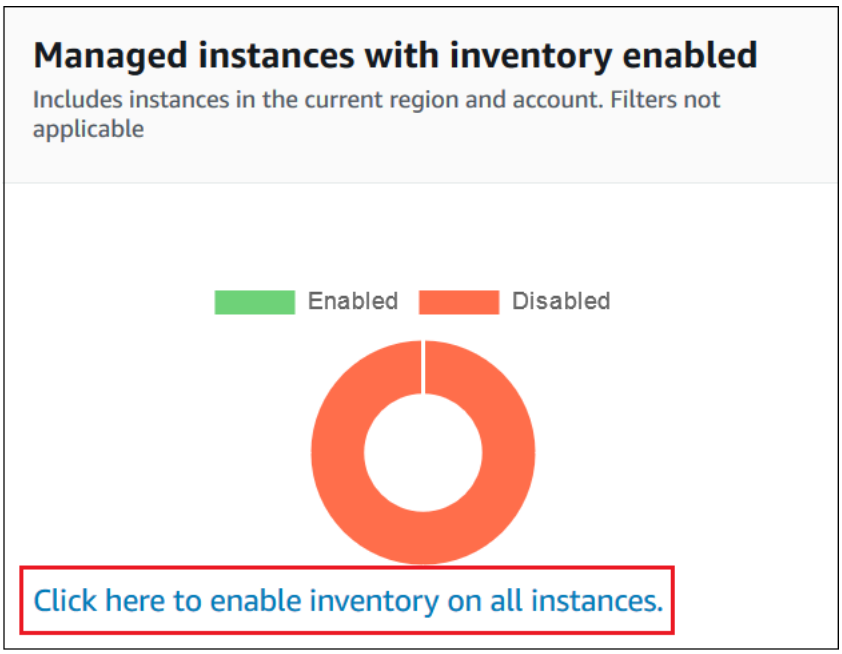
> **Note**
>
> - 如果为全局清单关联配置了一个托管式节点，并且您将特定关联分配给该节点，则 Systems Manager Inventory 会降低全局关联的优先级，并应用该特定关联。
> - 全局清单关联在 SSM Agent版本 2.0.790.0 或更高版本中可用。有关如何在节点上更新 SSM Agent 的信息，请参阅 使用 Run Command 更新 SSM Agent。

一键式配置清单收集（控制台）

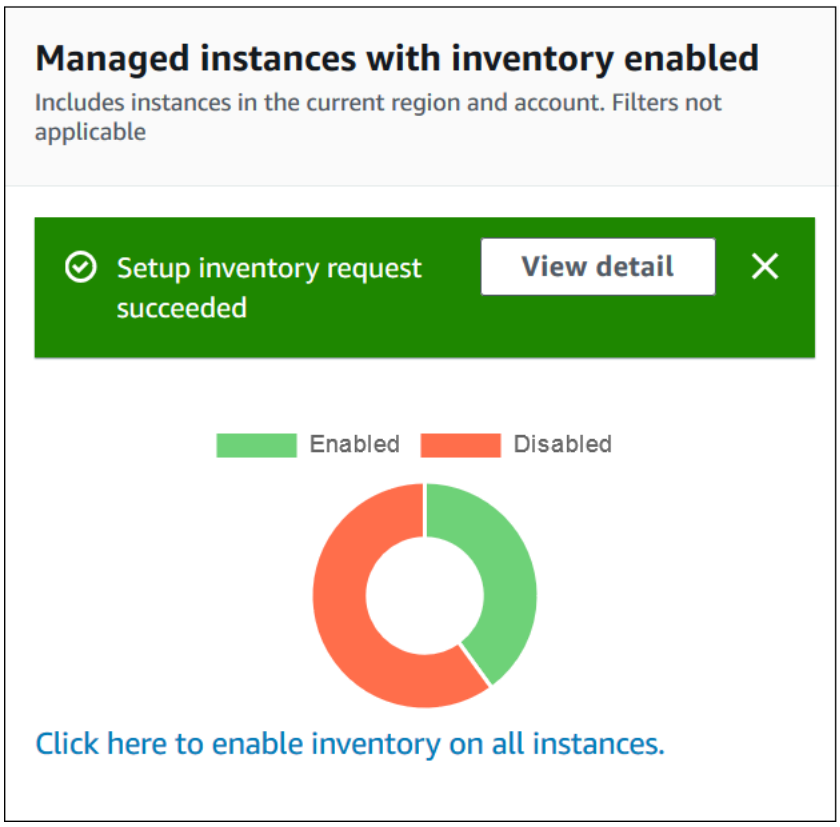
可以使用以下过程为您的 AWS 账户 和单个 AWS 区域 中的所有托管式节点配置 Systems Manager Inventory。

配置 Systems Manager Inventory 的当前区域中的所有托管式节点

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Inventory (清单)。
3. 在 Managed instances with inventory enabled (启用了清单的托管实例) 卡上，选择 Click here to enable inventory on all instances (单击此处在所有实例上启用清单)。



如果成功，控制台会显示以下消息。



根据您账户中的托管式节点的数量，可能需要几分钟才能应用全局库存关联。等待几分钟，然后刷新页面。验证图形更改是否反映在所有托管式节点上均配置了清单。

使用控制台配置收集

本节包含有关如何使用 Systems Manager 控制台将 Systems Manager Inventory 配置为收集托管式节点中的元数据的信息。您可以快速收集特定 AWS 账户 内所有节点（以及未来可能在该账户中创建的任何节点）中的元数据，也可以选择通过使用标签或节点 ID 收集清单数据。

> **Note**
>
> 在完成此过程之前，请检查是否已存在全局清单关联。如果全局清单关联已经存在，则无论何时启动新实例，该关联都将应用于该实例，并对新实例进行清点。

配置清单收集

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Inventory (清单)。
3. 选择设置清单。
4. 在 Targets（目标）部分中，通过选择以下选项之一来标识您要运行此操作的节点。

   - Selecting all managed instances in this account（选择此账户中的所有托管式实例）– 使用此选项可选择没有现有清单关联的所有托管式节点。如果您选择此选项，则在清单收集期间将跳过已有清单关联的节点，并会在清单结果中显示 Skipped（已跳过）状态。有关更多信息，请参阅清点 AWS 账户 中的所有托管式节点。
   - Specifying a tag（指定标签）– 使用此选项可以指定单个标签，用于标识要从中收集清单的账户中的节点。如果使用标签，则将来使用相同标签创建的所有节点也将报告清单。如果现有清单关联到所有节点，则使用标签选择特定节点作为其他清单的目标，会覆盖 All managed instances（所有托管式实例）目标组中的成员资格。未来从 All managed instances（所有托管式实例）进行清单收集时，将跳过具有指定标签的托管式节点。
   - Manually selecting instances（手动选择实例）– 使用此选项可以选择账户中的特定托管式节点。使用此选项明确选择特定节点，将覆盖 All managed instances（所有托管式实例）目标上的清单关联。未来从 All managed instances（所有托管式实例）进行清单收集时，将跳过该节点。

> **Note**
>
> 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

5. 在 Schedule（计划）部分中，选择您希望系统从节点收集清单元数据的频率。
6. 在 Parameters (参数) 部分中，使用列表打开或关闭不同类型的清单收集。有关收集文件和 Windows 注册表清单的更多信息，请参阅使用文件和 Windows 注册表清单。
7. 如果希望将关联执行状态存储在 Amazon S3 存储桶中，请在 Advanced (高级) 部分中选择 Sync inventory execution logs to an Amazon S3 bucket (将清单执行日志同步到 Amazon S3 存储桶)。
8. 选择设置清单。Systems Manager 会创建一个 State Manager 关联，并立即在节点上运行 Inventory。
9. 在导航窗格中，请选择 State Manager。验证并确保已创建了使用 **AWS-GatherSoftwareInventory** 文档的新关联。关联计划使用 Rate 表达式。此外，确认 Status 字段显示 Success。如果您选择了 Sync inventory execution logs to an Amazon S3 bucket (将清单执行日志同步到 Amazon S3 存储桶) 选项，则您可在几分钟后在 Amazon S3 中查看日志数据。要查看特定节点的清单数据，请在导航窗格中选择 Managed Instances（托管式实例）。
10. 选择一个节点，然后选择 View details（查看详细信息）。
11. 在节点详细信息页面上，选择 Inventory（清单）。使用 Inventory type 列表筛选清单。

查询多个区域和账户的清单数据

AWS Systems Manager Inventory 可与 Amazon Athena 集成，以帮助您查询来自多个 AWS 区域和 AWS 账户的清单数据。Athena 集成使用资源数据同步，以便您可以在 AWS Systems Manager 控制台中的 Detailed View（详细视图）页面上查看来自所有托管式节点的清单数据。

> **Important**
>
> 此功能将使用 AWS Glue 网络爬取您的 Amazon Simple Storage Service (Amazon S3) 存储桶中的数据，以及使用 Amazon Athena 查询这些数据。根据抓取和查询的数据量，您可能需要为使用这些服务付费。使用 AWS Glue 时，您需要按小时费率（按秒计）为爬网程序（发现数据）和 ETL 作业（处理和加载数据）付费。使用 Athena，您需要按每次查询所扫描的数据量付费。建议您在使用 Amazon Athena 与 Systems Manager Inventory 的集成之前查看这些服务的定价准则。有关更多信息，请参阅 Amazon Athena 定价和 AWS Glue 定价。

您可以在所有提供 Amazon Athena 的 AWS 区域 中的 Detailed View (详细视图) 页面上查看清单数据。有关受支持的区域列表，请参阅《Amazon Web Services 一般参考》中的 Amazon Athena Service Endpoints。

开始前的准备工作

Athena 集成使用资源数据同步。您必须设置并配置资源数据同步才能使用该功能。有关更多信息，请参阅 演练：使用资源数据同步聚合清单数据。

另请注意，Detailed View (详细视图) 页面将为资源数据同步使用的中央 Amazon S3 存储桶的拥有者显示清单数据。如果您不是中央 Amazon S3 存储桶的拥有者，则无法在 Detailed View (详细视图) 页面上查看清单数据。

配置访问权限

您必须将 IAM 实体配置为拥有查看数据的权限，然后才能在 Systems Manager 控制台中的详细视图页面上查询及查看来自多个账户和区域的数据。

如果清单数据存储在使用 AWS Key Management Service（AWS KMS）加密的 Amazon S3 存储桶中，您还必须配置 IAM 实体和 `Amazon-GlueServiceRoleForSSM` 服务角色，以便进行 AWS KMS 加密。

主题

- 配置您的 IAM 实体以访问“详细视图”页面
- （可选）配置查看 AWS KMS 加密数据的权限

配置您的 IAM 实体以访问“详细视图”页面

下文介绍了在详细视图页面上查看清单数据所需的最低权限。

`AWSQuicksightAthenaAccess` 托管策略

以下 `PassRole` 和其他所需权限块

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowGlue",
```

```
            "Effect": "Allow",
            "Action": [
                "glue:GetCrawler",
                "glue:GetCrawlers",
                "glue:GetTables",
                "glue:StartCrawler",
                "glue:CreateCrawler"
            ],
            "Resource": "*"
        },
        {
            "Sid": "iamPassRole",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                "arn:aws:iam::123456789012:role/SSMInventoryGlueRole",
                "arn:aws:iam::123456789012:role/SSMInventoryServiceRole"
            ],
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "glue.amazonaws.com"
                }
            }
        },
        {
            "Sid": "iamRoleCreation",
            "Effect": "Allow",
            "Action": [
                "iam:CreateRole",
                "iam:AttachRolePolicy"
            ],
            "Resource": "arn:aws:iam::111122223333:role/*"
        },
        {
            "Sid": "iamPolicyCreation",
            "Effect": "Allow",
            "Action": "iam:CreatePolicy",
            "Resource": "arn:aws:iam::111122223333:policy/*"
        }
    ]
}
```

（可选）如果用于存储清单数据的 Amazon S3 存储桶使用 AWS KMS 进行加密，则您还必须将以下数据块添加到策略中。

```
{
    "Effect": "Allow",
    "Action": [
        "kms:Decrypt"
    ],
    "Resource": [
        "arn:aws:kms:Region:account_ID:key/key_ARN"
    ]
}
```

要提供访问权限，请为您的用户、组或角色添加权限：

- AWS IAM Identity Center 中的用户和群组：

  创建权限集合。按照《AWS IAM Identity Center 用户指南》中创建权限集的说明进行操作。
- 通过身份提供商在 IAM 中托管的用户：

  创建适用于身份联合验证的角色。按照《IAM 用户指南》中针对第三方身份提供商创建角色（联合身份验证）的说明进行操作。
- IAM 用户：
  - 创建您的用户可以担任的角色。按照《IAM 用户指南》中为 IAM 用户创建角色的说明进行操作。
  - （不推荐使用）将策略直接附加到用户或将用户添加到用户组。按照《IAM 用户指南》中向用户添加权限（控制台）中的说明进行操作。

（可选）配置查看 AWS KMS 加密数据的权限

如果用于存储清单数据的 Amazon S3 存储桶使用 AWS Key Management Service（AWS KMS）进行加密，则您必须将 IAM 实体和 Amazon-GlueServiceRoleForSSM 角色配置为拥有对 AWS KMS 密钥的 `kms:Decrypt` 权限。

开始前的准备工作

要提供对 AWS KMS 密钥的 `kms:Decrypt` 权限，请将以下策略块添加到您的 IAM 实体：

```
{
    "Effect": "Allow",
```

```
    "Action": [
        "kms:Decrypt"
    ],
    "Resource": [
        "arn:aws:kms:Region:account_ID:key/key_ARN"
    ]
}
```

如果您尚未这样做，请完成该过程，并添加对 AWS KMS 密钥的 `kms:Decrypt` 权限。

使用以下过程将 Amazon-GlueServiceRoleForSSM 角色配置为拥有对 AWS KMS 密钥的 `kms:Decrypt` 权限。

将 Amazon-GlueServiceRoleForSSM 角色配置为拥有 **`kms:Decrypt`** 权限

1. 通过 https://console.aws.amazon.com/iam/ 打开 IAM 控制台。
2. 在导航窗格中，选择 Roles (角色)，然后使用搜索字段查找 Amazon-GlueServiceRoleForSSM 角色。此时将打开摘要页面。
3. 使用搜索字段查找 Amazon-GlueServiceRoleForSSM 角色。选择角色名称 。此时将打开摘要页面。
4. 选择角色名称 。此时将打开摘要页面。
5. 选择添加内联策略。此时将打开创建策略页面。
6. 选择 JSON 选项卡。
7. 删除编辑器中现有的 JSON 文本，然后将以下策略复制并粘贴到 JSON 编辑器中。

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/key_ARN"
            ]
        }
```

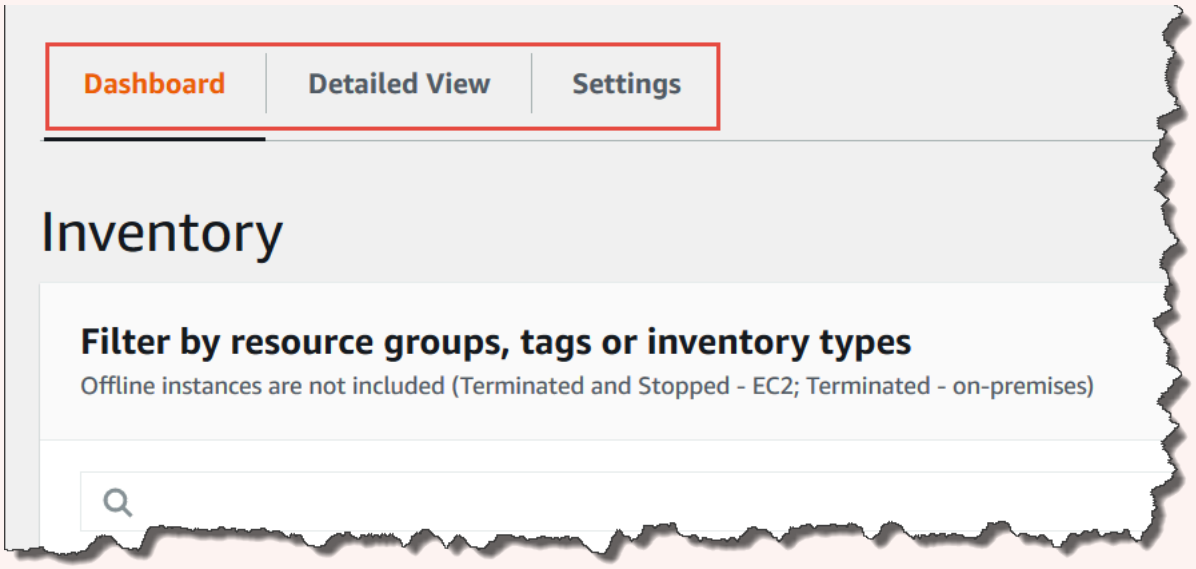
```
    ]
}
```

8. 选择查看策略
9. 在查看策略页面上的名称字段中输入名称。
10. 选择创建策略。

在清单详细视图页面上查询数据

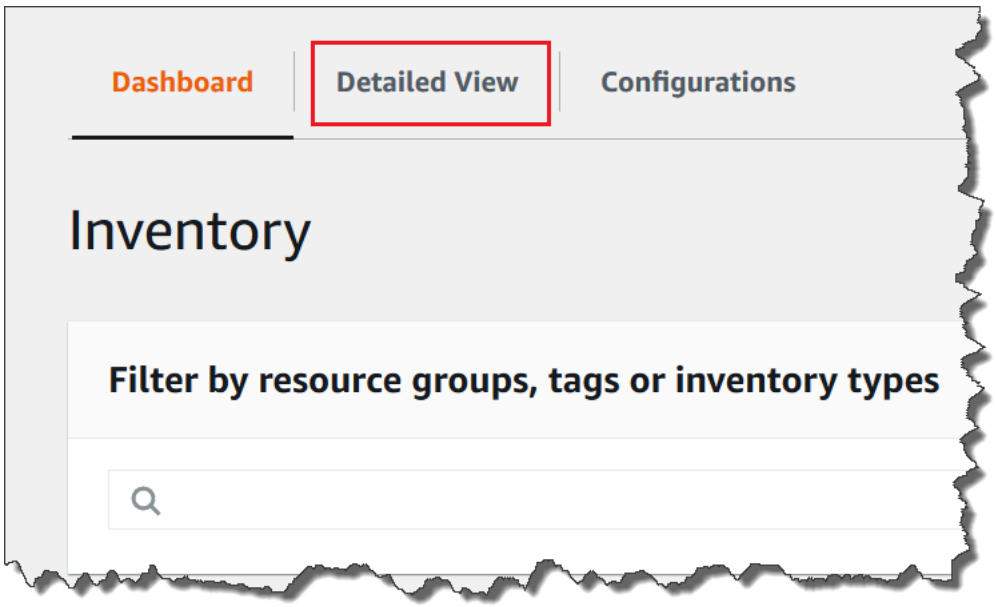
使用以下过程可在 Systems Manager Inventory Detailed View (详细视图) 页面上查看来自多个 AWS 区域和 AWS 账户的清单数据。

> **Important**
>
> 仅在提供 Amazon Athena 的 AWS 区域 中提供了 Inventory Detailed View (详细视图) 页面。如果在 Systems Manager Inventory 页面上未显示以下选项卡，则意味着 Athena 在该区域中不可用，并且您无法使用 Detailed View (详细视图) 查询数据。

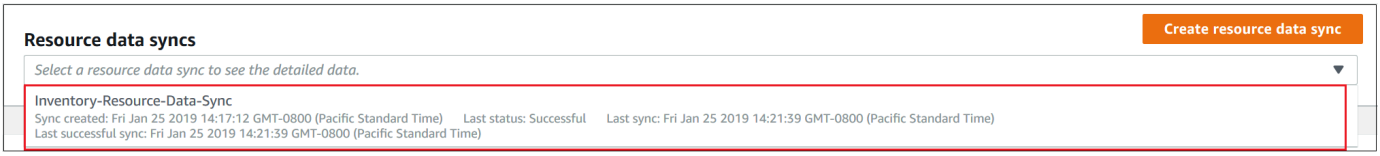


在 AWS Systems Manager 控制台中查看多个区域和账户的清单数据

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Inventory (清单)。
3. 选择 Detailed View (详细视图) 选项卡。



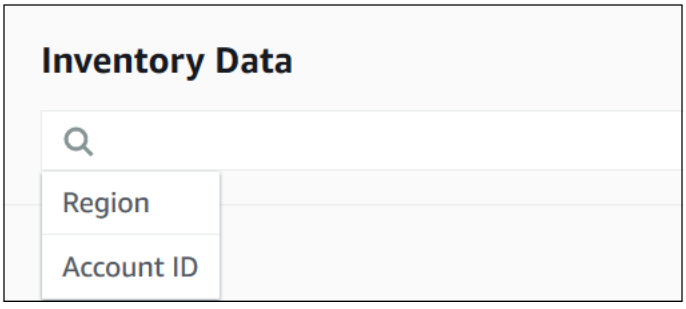
4. 选择要查询数据的资源数据同步。



5. 在 Inventory Type (清单类型) 列表中，请选择要查询的清单数据类型，然后按 Enter。



6. 要筛选数据，请选择筛选条件栏，然后选择筛选选项。



您可以使用 Export to CSV (导出到 CSV) 按钮在电子表格应用程序（如 Microsoft Excel）中查看当前查询集。您也可以使用 Query History (查询历史记录) 和 Run Advanced Queries (运行高级查询) 按钮查看历史记录详细信息，并与 Amazon Athena 中的数据进行交互。

编辑 AWS Glue 爬网程序计划

预设情况下，AWS Glue 每天会爬取中央 Amazon S3 存储桶中的清单数据两次。如果您经常更改要在节点上收集的数据的类型，则可能需要更频繁地抓取数据，如以下过程中所述。

> **Important**
>
> AWS Glue 按小时费率（按秒计费）向您的 AWS 账户收取执行爬网程序（发现数据）和 ETL 任务（处理和加载数据）的费用。在更改爬网程序计划前，请先查看 AWS Glue 定价页面。

更改清单数据爬网程序计划

1. 通过 https://console.aws.amazon.com/glue/ 打开 AWS Glue 控制台。
2. 在导航窗格中，选择 爬网程序。
3. 在爬网程序列表中，选择 Systems Manager Inventory 数据爬网程序旁边的选项。爬网程序名称使用以下格式：

   `AWSSystemsManager-s3-bucket-name-Region-account_ID`
4. 选择 Action (操作)，然后选择 Edit crawler (编辑爬网程序)。
5. 在导航窗格中，选择 Schedule (计划)。
6. 在 Cron expression (Cron 表达式) 字段中，使用 cron 格式指定一个新计划。有关更多信息，请参阅 AWS Glue Developer Guide 中的基于时间的任务和爬网程序日程。

> **Important**
>
> 您可以暂停爬网程序以停止 AWS Glue 产生的费用。如果暂停爬网程序或降低爬取数据的频率，则 Inventory Detailed View (详细视图) 可能会显示非当前数据。

使用筛选条件查询清单收集

收集清单数据后，可以使用 AWS Systems Manager 中的筛选功能查询满足特定筛选条件的托管式节点列表。

根据清单筛选条件查询节点

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Inventory (清单)。
3. 在 Filter by resource groups, tags or inventory types 部分中，选择筛选框。此时会显示预定义筛选器列表。
4. 选择要筛选的属性。例如，选择 **AWS:Application**。如果出现提示，选择要筛选的二级属性。例如，选择 **AWS:Application.Name**。
5. 从列表中选择一个分隔符。例如，选择 Begin with。此时筛选器中会显示一个文本框。
6. 在该文本框中输入一个值。例如，输入 Amazon（SSM Agent 命名为 Amazon SSM Agent）。
7. 按 Enter。系统返回一个托管式节点列表，其中包含以单词 Amazon 开头的应用程序名称。

> **Note**
>
> 您可以组合多个筛选条件来细化搜索。

聚合清单数据

在为 AWS Systems Manager Inventory 配置托管式节点后，您可以查看清单数据的聚合计数。例如，假设您配置了数十或数百个托管式节点来收集 `AWS:Application` 清单类型。通过使用此部分中的信息，您即可查看配置为收集此数据的节点数的准确计数。

您还可以通过针对某数据类型进行聚合以查看特定清单详细信息。例如，`AWS:InstanceInformation` 清单类型收集 `Platform` 数据类型的操作系统平台信息。通过聚合 `Platform` 数据类型的数据，您可以快速查看运行 Windows、Linux 以及 macOS 的节点的数量。

本节中的过程介绍了如何使用 AWS Command Line Interface (AWS CLI) 查看清单数据的聚合计数。您还可以在 AWS Systems Manager 控制台的 Inventory (清单) 页面上预配置聚合计数。这些预配置的控制面板称为 Inventory Insights，而且它们提供清单配置问题的一键修正。

请注意有关清单数据的聚合计数的以下重要详细信息：

- 如果您终止配置为收集清单数据的托管式节点，Systems Manager 会将清单数据保留 30 天，然后将其删除。对于正在运行的节点，系统将删除存在超过 30 天的清单数据。如果您需要将清单数据

存储 30 天以上，可以使用 AWS Config 来记录历史记录，或者定期查询数据并将其上传到 Amazon Simple Storage Service (Amazon S3) 存储桶。

- 如果先前已将某个节点配置为报告某一特定清单数据类型（例如 AWS:Network），后来您将该配置更改为停止收集该类型，聚合计数仍会显示 AWS:Network 数据，直到该节点已被终止并已经过 30 天为止。

有关如何快速配置并收集某一特定 AWS 账户 中所有节点（以及未来可能在该账户中创建的任何节点）中的清单数据的信息，请参阅 清点 AWS 账户 中的所有托管式节点。

主题

- 聚合清单数据以查看收集特定类型数据的节点的计数
- 使用组聚合清单数据，以查看已配置为以及未配置为收集某个清单类型的节点

聚合清单数据以查看收集特定类型数据的节点的计数

您可以使用 AWS Systems Manager GetInventory API 操作，查看收集一个或多个清单类型和数据类型的节点的聚合计数。例如，AWS:InstanceInformation 清单类型使您能够通过结合使用 GetInventory API 操作和 AWS:InstanceInformation.PlatformType 数据类型，来查看操作系统的聚合。以下是示例 AWS CLI 命令和输出。

```
aws ssm get-inventory --aggregators "Expression=AWS:InstanceInformation.PlatformType"
```

系统将返回类似于以下内容的信息。

```
{
   "Entities":[
      {
         "Data":{
            "AWS:InstanceInformation":{
               "Content":[
                  {
                     "Count":"7",
                     "PlatformType":"windows"
                  },
                  {
                     "Count":"5",
                     "PlatformType":"linux"
                  }
               ]
```

```
            }
         }
      }
   ]
}
```

开始使用

确定要查看其计数的清单类型和数据类型。您可以通过在 AWS CLI 中运行以下命令查看支持聚合的清单类型和数据类型的列表。

```
aws ssm get-inventory-schema --aggregator
```

该命令返回支持聚合的清单类型和数据类型的 JSON 列表。TypeName 字段显示支持的清单类型。此处，Name (名称) 字段显示每个数据类型。例如，在以下列表中，AWS:Application 清单类型包括 Name 和 Version 的数据类型。

```
{
    "Schemas": [
        {
            "TypeName": "AWS:Application",
            "Version": "1.1",
            "DisplayName": "Application",
            "Attributes": [
                {
                    "DataType": "STRING",
                    "Name": "Name"
                },
                {
                    "DataType": "STRING",
                    "Name": "Version"
                }
            ]
        },
        {
            "TypeName": "AWS:InstanceInformation",
            "Version": "1.0",
            "DisplayName": "Platform",
            "Attributes": [
                {
                    "DataType": "STRING",
                    "Name": "PlatformName"
```

```
                },
                {
                    "DataType": "STRING",
                    "Name": "PlatformType"
                },
                {
                    "DataType": "STRING",
                    "Name": "PlatformVersion"
                }
            ]
        },
        {
            "TypeName": "AWS:ResourceGroup",
            "Version": "1.0",
            "DisplayName": "ResourceGroup",
            "Attributes": [
                {
                    "DataType": "STRING",
                    "Name": "Name"
                }
            ]
        },
        {
            "TypeName": "AWS:Service",
            "Version": "1.0",
            "DisplayName": "Service",
            "Attributes": [
                {
                    "DataType": "STRING",
                    "Name": "Name"
                },
                {
                    "DataType": "STRING",
                    "Name": "DisplayName"
                },
                {
                    "DataType": "STRING",
                    "Name": "ServiceType"
                },
                {
                    "DataType": "STRING",
                    "Name": "Status"
                },
                {
```

```
                    "DataType": "STRING",
                    "Name": "StartType"
                }
            ]
        },
        {
            "TypeName": "AWS:WindowsRole",
            "Version": "1.0",
            "DisplayName": "WindowsRole",
            "Attributes": [
                {
                    "DataType": "STRING",
                    "Name": "Name"
                },
                {
                    "DataType": "STRING",
                    "Name": "DisplayName"
                },
                {
                    "DataType": "STRING",
                    "Name": "FeatureType"
                },
                {
                    "DataType": "STRING",
                    "Name": "Installed"
                }
            ]
        }
    ]
}
```

您可以通过创建使用以下语法的命令来聚合任何所列清单类型的数据。

```
aws ssm get-inventory --aggregators "Expression=InventoryType.DataType"
```

下面是一些示例。

示例 1

此示例聚合节点使用的 Windows 角色的计数。

```
aws ssm get-inventory --aggregators "Expression=AWS:WindowsRole.Name"
```

示例 2

此示例聚合安装在节点上的应用程序的计数。

```
aws ssm get-inventory --aggregators "Expression=AWS:Application.Name"
```

组合多个聚合器

您也可以在一个命令中组合多个清单类型和数据类型，以帮助您更好地了解数据。下面是一些示例。

示例 1

此示例聚合节点使用的操作系统类型的计数。它还会返回操作系统的特定名称。

```
aws ssm get-inventory --aggregators '[{"Expression":
 "AWS:InstanceInformation.PlatformType", "Aggregators":[{"Expression":
 "AWS:InstanceInformation.PlatformName"}]}]'
```

示例 2

此示例聚合在节点上运行的应用程序和每个应用程序的特定版本的计数。

```
aws ssm get-inventory --aggregators '[{"Expression": "AWS:Application.Name",
 "Aggregators":[{"Expression": "AWS:Application.Version"}]}]'
```

如果您愿意，也可以使用 JSON 文件中的一个或多个清单类型和数据类型创建聚合表达式并从 AWS CLI 调用该文件。该文件中的 JSON 必须使用以下语法。

```
[
       {
           "Expression": "string",
           "Aggregators": [
               {
                  "Expression": "string"
               }
           ]
       }
]
```

您必须使用 .json 文件扩展名保存该文件。

以下是使用多个清单类型和数据类型的示例。

```
[
       {
           "Expression": "AWS:Application.Name",
           "Aggregators": [
               {
                   "Expression": "AWS:Application.Version",
                   "Aggregators": [
                     {
                     "Expression": "AWS:InstanceInformation.PlatformType"
                     }
                   ]
               }
           ]
       }
]
```

使用以下命令从 AWS CLI 调用该文件。

```
aws ssm get-inventory --aggregators file://file_name.json
```

此命令会返回如下信息。

```
{"Entities":
 [
   {"Data":
     {"AWS:Application":
       {"Content":
         [
           {"Count": "3",
            "PlatformType": "linux",
            "Version": "2.6.5",
            "Name": "audit-libs"},
           {"Count": "2",
            "PlatformType": "windows",
            "Version": "2.6.5",
            "Name": "audit-libs"},
           {"Count": "4",
            "PlatformType": "windows",
            "Version": "6.2.8",
            "Name": "microsoft office"},
           {"Count": "2",
            "PlatformType": "windows",
```

```
            "Version": "2.6.5",
            "Name": "chrome"},
           {"Count": "1",
            "PlatformType": "linux",
            "Version": "2.6.5",
            "Name": "chrome"},
           {"Count": "2",
            "PlatformType": "linux",
            "Version": "6.3",
            "Name": "authconfig"}
         ]
       }
     },
    "ResourceType": "ManagedInstance"}
 ]
}
```

使用组聚合清单数据，以查看已配置为以及未配置为收集某个清单类型的节点

使用 Systems Manager Inventory 中的组，您可以快速查看已配置为和未配置为收集一个或多个清单类型的托管式节点的计数。利用组，可指定一个或多个清单类型和使用 `exists` 运算符的筛选条件。

例如，假设您已将四个托管式节点配置为收集以下清单类型：

- 节点 1：`AWS:Application`
- 节点 2：`AWS:File`
- 节点 3：`AWS:Application`、`AWS:File`
- 节点 4：`AWS:Network`

您可以从 AWS CLI 运行以下命令，以查看已配置为收集 `AWS:Application` 和 `AWS:File` `inventory` 类型的节点的数量。该响应还返回未配置为收集这两个清单类型的节点数的计数。

```
aws ssm get-inventory --aggregators
 'Groups=[{Name=ApplicationAndFile,Filters=[{Key=TypeName,Values=[AWS:Application],Type=Exists}
{Key=TypeName,Values=[AWS:File],Type=Exists}]}]'
```

该命令响应显示仅一个托管式节点已配置为收集 `AWS:Application` 和 `AWS:File` 清单类型。

```
{
   "Entities":[
```

```
      {
         "Data":{
            "ApplicationAndFile":{
               "Content":[
                  {
                     "notMatchingCount":"3"
                  },
                  {
                     "matchingCount":"1"
                  }
               ]
            }
         }
      }
   ]
}
```

> **Note**
>
> 组不会返回数据类型计数。此外，您也无法深入查询结果来查看已配置为或未配置为收集该清单类型的节点的 ID。

如果您愿意，也可以使用 JSON 文件中的一个或多个清单类型创建聚合表达式并从 AWS CLI 调用该文件。该文件中的 JSON 必须使用以下语法：

```
{
   "Aggregators":[
      {
         "Groups":[
            {
               "Name":"Name",
               "Filters":[
                  {
                     "Key":"TypeName",
                     "Values":[
                        "Inventory_type"
                     ],
                     "Type":"Exists"
                  },
                  {
                     "Key":"TypeName",
```

```
                    "Values":[
                       "Inventory_type"
                    ],
                    "Type":"Exists"
                 }
              ]
           }
        ]
     }
  ]
}
```

您必须使用 .json 文件扩展名保存该文件。

使用以下命令从 AWS CLI 调用该文件。

```
aws ssm get-inventory --cli-input-json file://file_name.json
```

其他示例

以下示例说明了如何聚合清单数据，以查看已配置为和未配置为收集指定清单类型的托管式节点。这些示例使用 AWS CLI。每个示例包含一个带筛选条件的完整命令，您可以从命令行和示例 input.json 文件运行该命令（如果您更喜欢在文件中输入该信息）。

示例 1

此示例聚合已配置为和未配置为收集 AWS:Application 或 AWS:File 清单类型的节点的计数。

从 AWS CLI 运行以下命令。

```
aws ssm get-inventory --aggregators
 'Groups=[{Name=ApplicationORFile,Filters=[{Key=TypeName,Values=[AWS:Application,
 AWS:File],Type=Exists}]}]'
```

如果您更喜欢使用文件，请将以下示例复制并粘贴到文件中并且将其另存为 input.json。

```
{
   "Aggregators":[
      {
         "Groups":[
            {
               "Name":"ApplicationORFile",
```

```
                "Filters":[
                   {
                      "Key":"TypeName",
                      "Values":[
                         "AWS:Application",
                         "AWS:File"
                      ],
                      "Type":"Exists"
                   }
                ]
             }
          ]
       }
   ]
}
```

从 AWS CLI 运行以下命令。

```
aws ssm get-inventory --cli-input-json file://input.json
```

此命令会返回如下信息。

```
{
   "Entities":[
      {
         "Data":{
            "ApplicationORFile":{
               "Content":[
                  {
                     "notMatchingCount":"1"
                  },
                  {
                     "matchingCount":"3"
                  }
               ]
            }
         }
      }
   ]
}
```

示例 2

此示例聚合已配置为和未配置为收集 AWS:Application、AWS:File 和 AWS:Network 清单类型的节点的计数。

从 AWS CLI 运行以下命令。

```
aws ssm get-inventory --aggregators
 'Groups=[{Name=Application,Filters=[{Key=TypeName,Values=[AWS:Application],Type=Exists}]},
 {Name=File,Filters=[{Key=TypeName,Values=[AWS:File],Type=Exists}]},
 {Name=Network,Filters=[{Key=TypeName,Values=[AWS:Network],Type=Exists}]}]'
```

如果您更喜欢使用文件，请将以下示例复制并粘贴到文件中并且将其另存为 input.json。

```
{
   "Aggregators":[
      {
         "Groups":[
            {
               "Name":"Application",
               "Filters":[
                  {
                     "Key":"TypeName",
                     "Values":[
                        "AWS:Application"
                     ],
                     "Type":"Exists"
                  }
               ]
            },
            {
               "Name":"File",
               "Filters":[
                  {
                     "Key":"TypeName",
                     "Values":[
                        "AWS:File"
                     ],
                     "Type":"Exists"
                  }
               ]
            },
            {
               "Name":"Network",
               "Filters":[
```

```
                {
                   "Key":"TypeName",
                   "Values":[
                      "AWS:Network"
                   ],
                   "Type":"Exists"
                }
             ]
          }
       ]
    }
 ]
}
```

从 AWS CLI 运行以下命令。

```
aws ssm get-inventory --cli-input-json file://input.json
```

此命令会返回如下信息。

```
{
   "Entities":[
      {
         "Data":{
            "Application":{
               "Content":[
                  {
                     "notMatchingCount":"2"
                  },
                  {
                     "matchingCount":"2"
                  }
               ]
            },
            "File":{
               "Content":[
                  {
                     "notMatchingCount":"2"
                  },
                  {
                     "matchingCount":"2"
                  }
               ]
```

```
            },
            "Network":{
               "Content":[
                  {
                     "notMatchingCount":"3"
                  },
                  {
                     "matchingCount":"1"
                  }
               ]
            }
         }
      }
   ]
}
```

使用自定义清单

您可以通过创建 AWS Systems Manager Inventory 自定义清单，将所需的任何元数据分配给您的节点。例如，假设您负责管理数据中心内多个机架中的大量服务器，而且这些服务器已配置为 Systems Manager 托管式节点。目前，您在电子表格中存储服务器机架位置的相关信息。借助自定义清单，您可以指定每个节点的机架位置作为节点上的元数据。当您使用 Systems Manager 收集清单时，该元数据将与其他清单元数据一起收集。然后，您可以使用资源数据同步将所有清单元数据移植到中央 Amazon S3 存储桶，并查询这些数据。

> **Note**
>
> Systems Manager 最多可为每个 AWS 账户 支持 20 个自定义清单类型。

要将自定义清单分配给某个节点，可以使用 Systems Manager PutInventory API 操作，如 将自定义清单元数据分配给某个托管式节点 中所述。或者，您可以创建自定义清单 JSON 文件并将其上传到该节点。本部分描述了如何创建 JSON 文件。

以下包含自定义清单的示例 JSON 文件指定有关本地服务器的机架信息。此示例指定一种类型的自定义清单数据 (`"TypeName": "Custom:RackInformation"`)，在描述数据的 `Content` 下有多个条目。

```
{
    "SchemaVersion": "1.0",
    "TypeName": "Custom:RackInformation",
```

```
    "Content": {
        "Location": "US-EAST-02.CMH.RACK1",
        "InstalledTime": "2016-01-01T01:01:01Z",
        "vendor": "DELL",
        "Zone" : "BJS12",
        "TimeZone": "UTC-8"
      }
 }
```

您也可以在 Content 部分指定不同条目，如以下示例所示。

```
{
"SchemaVersion": "1.0",
"TypeName": "Custom:PuppetModuleInfo",
    "Content": [{
        "Name": "puppetlabs/aws",
        "Version": "1.0"
      },
      {
        "Name": "puppetlabs/dsc",
        "Version": "2.0"
      }
    ]
}
```

自定义清单的 JSON 架构需要 SchemaVersion、TypeName 和 Content 部分，但您可以定义这些部分的信息。

```
{
    "SchemaVersion": "user_defined",
    "TypeName": "Custom:user_defined",
    "Content": {
        "user_defined_attribute1": "user_defined_value1",
        "user_defined_attribute2": "user_defined_value2",
        "user_defined_attribute3": "user_defined_value3",
        "user_defined_attribute4": "user_defined_value4"
      }
 }
```

TypeName 值的长度限制为 100 个字符。此外，TypeName 值必须以大写的单词 Custom 开头。例如 Custom:PuppetModuleInfo。因此，以下示例将导致异常：CUSTOM:PuppetModuleInfo、custom:PuppetModuleInfo。

Content 部分包括属性和 *data*。这些项目不区分大小写。但是，如果您定义了属性（例如：“Vendor”：“DELL”），则在自定义清单文件中必须一致地引用此属性。如果您在一个文件中指定“Vendor”：“DELL”（在 vendor 中使用大写字母“V”），在另一个文件中指定“vendor”：“DELL”（在 vendor 中使用小写字母“v”），系统会返回错误。

> **Note**
>
> 您必须使用 .json 扩展名保存文件，并且您定义的清单必须仅包含字符串值。

创建文件之后，您必须在节点上保存文件。下表说明了自定义清单 JSON 文件必须存储在节点上的哪个位置。

操作系统	路径
Linux	/var/lib/amazon/ssm/*node-id*/inventory/custom
macOS	/opt/aws/ssm/data/ *node-id*/ inventory/custom
Windows	%SystemDrive%\ProgramData\Amazon\SSM \InstanceData*note-id*\inventory\custom

有关如何使用自定义清单的示例，请参阅使用 EC2 Systems Manager 自定义清单类型获取队列的磁盘利用率。

删除自定义清单

您可以使用 DeleteInventory API 操作来删除自定义清单类型以及与该类型关联的数据。您可以使用 AWS Command Line Interface (AWS CLI) 调用 delete-inventory 命令来删除某一清单类型的所有数据。您可以使用 SchemaDeleteOption 调用 delete-inventory 命令来删除自定义清单类型。

> **Note**
>
> 清单类型也称为清单架构。

SchemaDeleteOption 参数包括以下选项：

- DeleteSchema：此选项删除指定的自定义类型和与之关联的所有数据。如果需要，可以稍后重新创建架构。
- DisableSchema：如果您选择此选项，系统将关闭当前版本，删除其所有数据，并在版本低于或等于已关闭版本时忽略所有新数据。您可以通过对高于已关闭版本的版本调用 PutInventory 操作来重新允许此清单类型。

使用 AWS CLI 删除或关闭自定义清单

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 运行以下命令来使用 `dry-run` 选项查看将从系统中删除哪些数据。此命令不会删除任何数据。

   ```
   aws ssm delete-inventory --type-name "Custom:custom_type_name" --dry-run
   ```

   系统将返回类似于以下内容的信息。

   ```
   {
      "DeletionSummary":{
         "RemainingCount":3,
         "SummaryItems":[
            {
               "Count":2,
               "RemainingCount":2,
               "Version":"1.0"
            },
            {
               "Count":1,
               "RemainingCount":1,
               "Version":"2.0"
            }
         ],
         "TotalCount":3
      },
      "TypeName":"Custom:custom_type_name"
   }
   ```

   有关如何了解和删除清单摘要的信息，请参阅 了解删除清单摘要。

3. 运行以下命令来删除自定义清单类型的所有数据。

```
aws ssm delete-inventory --type-name "Custom:custom_type_name"
```

> **Note**
>
> 此命令的输出未显示删除进度。因此，TotalCount (总计数) 和 Remaining Count (剩余计数) 始终相同，因为系统尚未删除任何数据。您可以使用 describe-inventory-deletions 命令来显示删除进度，如本主题的下文所述。

系统将返回类似于以下内容的信息。

```
{
   "DeletionId":"system_generated_deletion_ID",
   "DeletionSummary":{
      "RemainingCount":3,
      "SummaryItems":[
         {
            "Count":2,
            "RemainingCount":2,
            "Version":"1.0"
         },
         {
            "Count":1,
            "RemainingCount":1,
            "Version":"2.0"
         }
      ],
      "TotalCount":3
   },
   "TypeName":"custom_type_name"
}
```

系统将从 Systems Manager Inventory 服务中删除指定的自定义清单类型的所有数据。

4. 运行以下命令。此命令对清单类型的当前版本执行以下操作：关闭当前版本、删除其所有数据，并在版本低于或等于已关闭版本时忽略所有新数据。

```
aws ssm delete-inventory --type-name "Custom:custom_type_name" --schema-delete-
option "DisableSchema"
```

系统将返回类似于以下内容的信息。

```
{
   "DeletionId":"system_generated_deletion_ID",
   "DeletionSummary":{
      "RemainingCount":3,
      "SummaryItems":[
         {
            "Count":2,
            "RemainingCount":2,
            "Version":"1.0"
         },
         {
            "Count":1,
            "RemainingCount":1,
            "Version":"2.0"
         }
      ],
      "TotalCount":3
   },
   "TypeName":"Custom:custom_type_name"
}
```

您可以使用以下命令查看已关闭的清单类型。

```
aws ssm get-inventory-schema --type-name Custom:custom_type_name
```

5. 运行以下命令来删除清单类型。

```
aws ssm delete-inventory --type-name "Custom:custom_type_name" --schema-delete-
option "DeleteSchema"
```

系统将删除指定的自定义类型的架构和所有清单数据。

系统将返回类似于以下内容的信息。

```
{
   "DeletionId":"system_generated_deletion_ID",
   "DeletionSummary":{
      "RemainingCount":3,
      "SummaryItems":[
```

```
         {
            "Count":2,
            "RemainingCount":2,
            "Version":"1.0"
         },
         {
            "Count":1,
            "RemainingCount":1,
            "Version":"2.0"
         }
      ],
      "TotalCount":3
   },
   "TypeName":"Custom:custom_type_name"
}
```

查看删除状态

您可以使用 `describe-inventory-deletions` AWS CLI 命令来查看删除操作的状态。您可以指定删除 ID 来查看特定删除操作的状态。或者，您也可以省略删除 ID 来查看最近 30 天运行的所有删除的列表。

1. 运行以下命令来查看删除操作的状态。系统将返回 delete-inventory 摘要中的删除 ID。

   ```
   aws ssm describe-inventory-deletions --deletion-id system_generated_deletion_ID
   ```

   系统将返回最新状态。删除操作可能尚未完成。系统将返回类似于以下内容的信息。

   ```
   {"InventoryDeletions":
     [
       {"DeletionId": "system_generated_deletion_ID",
        "DeletionStartTime": 1521744844,
        "DeletionSummary":
         {"RemainingCount": 1,
          "SummaryItems":
           [
             {"Count": 1,
              "RemainingCount": 1,
              "Version": "1.0"}
           ],
   ```

```
        "TotalCount": 1},
      "LastStatus": "InProgress",
      "LastStatusMessage": "The Delete is in progress",
      "LastStatusUpdateTime": 1521744844,
      "TypeName": "Custom:custom_type_name"}
  ]
}
```

如果删除操作成功，则 `LastStatusMessage` 指出：Deletion is successful。

```
{"InventoryDeletions":
  [
    {"DeletionId": "system_generated_deletion_ID",
     "DeletionStartTime": 1521744844,
     "DeletionSummary":
      {"RemainingCount": 0,
       "SummaryItems":
        [
          {"Count": 1,
           "RemainingCount": 0,
           "Version": "1.0"}
        ],
       "TotalCount": 1},
     "LastStatus": "Complete",
     "LastStatusMessage": "Deletion is successful",
     "LastStatusUpdateTime": 1521745253,
     "TypeName": "Custom:custom_type_name"}
  ]
}
```

2. 运行以下命令来查看最近 30 天运行的所有删除的列表。

```
aws ssm describe-inventory-deletions --max-results a number
```

```
{"InventoryDeletions":
  [
    {"DeletionId": "system_generated_deletion_ID",
     "DeletionStartTime": 1521682552,
     "DeletionSummary":
      {"RemainingCount": 0,
       "SummaryItems":
```

```
        [
          {"Count": 1,
           "RemainingCount": 0,
           "Version": "1.0"}
        ],
       "TotalCount": 1},
    "LastStatus": "Complete",
    "LastStatusMessage": "Deletion is successful",
    "LastStatusUpdateTime": 1521682852,
    "TypeName": "Custom:custom_type_name"},
   {"DeletionId": "system_generated_deletion_ID",
    "DeletionStartTime": 1521744844,
    "DeletionSummary":
     {"RemainingCount": 0,
      "SummaryItems":
       [
         {"Count": 1,
          "RemainingCount": 0,
          "Version": "1.0"}
       ],
      "TotalCount": 1},
    "LastStatus": "Complete",
    "LastStatusMessage": "Deletion is successful",
    "LastStatusUpdateTime": 1521745253,
    "TypeName": "Custom:custom_type_name"},
   {"DeletionId": "system_generated_deletion_ID",
    "DeletionStartTime": 1521680145,
    "DeletionSummary":
     {"RemainingCount": 0,
      "SummaryItems":
       [
         {"Count": 1,
          "RemainingCount": 0,
          "Version": "1.0"}
       ],
      "TotalCount": 1},
    "LastStatus": "Complete",
    "LastStatusMessage": "Deletion is successful",
    "LastStatusUpdateTime": 1521680471,
    "TypeName": "Custom:custom_type_name"}
 ],
"NextToken": "next-token"
```

了解删除清单摘要

为帮助您了解删除清单摘要的内容，请考虑以下示例。某用户将 Custom:RackSpace 清单分配给了三个节点。清单项目 1 和 2 使用自定义类型版本 1.0 ("SchemaVersion":"1.0")。清单项目 3 使用自定义类型版本 2.0 ("SchemaVersion":"2.0")。

RackSpace 自定义清单 1

```
{
   "CaptureTime":"2018-02-19T10:48:55Z",
   "TypeName":"CustomType:RackSpace",
   "InstanceId":"i-1234567890",
   "SchemaVersion":"1.0"   "Content":[
      {
         content of custom type omitted
      }
   ]
}
```

RackSpace 自定义清单 2

```
{
   "CaptureTime":"2018-02-19T10:48:55Z",
   "TypeName":"CustomType:RackSpace",
   "InstanceId":"i-1234567891",
   "SchemaVersion":"1.0"   "Content":[
      {
         content of custom type omitted
      }
   ]
}
```

RackSpace 自定义清单 3

```
{
   "CaptureTime":"2018-02-19T10:48:55Z",
   "TypeName":"CustomType:RackSpace",
   "InstanceId":"i-1234567892",
   "SchemaVersion":"2.0"   "Content":[
      {
         content of custom type omitted
```

```
      }
   ]
}
```

用户运行以下命令来预览将删除哪些数据。

```
aws ssm delete-inventory --type-name "Custom:RackSpace" --dry-run
```

系统将返回类似于以下内容的信息。

```
{
   "DeletionId":"1111-2222-333-444-66666",
   "DeletionSummary":{
      "RemainingCount":3,
      "TotalCount":3,
                TotalCount and RemainingCount are the number of items that would be
 deleted if this was not a dry run. These numbers are the same because the system
 didn't delete anything.
      "SummaryItems":[
         {
            "Count":2,             The system found two items that use SchemaVersion
 1.0. Neither item was deleted.
            "RemainingCount":2,
            "Version":"1.0"
         },
         {
            "Count":1,             The system found one item that uses SchemaVersion
 1.0. This item was not deleted.
            "RemainingCount":1,
            "Version":"2.0"
         }
      ],

   },
   "TypeName":"Custom:RackSpace"
}
```

用户运行以下命令来删除 Custom:RackSpace 清单。

> **Note**
>
> 此命令的输出未显示删除进度。因此，TotalCount 和 RemainingCount 始终相同，因为系统尚未删除任何数据。您可以使用 describe-inventory-deletions 命令来显示删除进度。

```
aws ssm delete-inventory --type-name "Custom:RackSpace"
```

系统将返回类似于以下内容的信息。

```
{
   "DeletionId":"1111-2222-333-444-7777777",
   "DeletionSummary":{
      "RemainingCount":3,           There are three items to delete
      "SummaryItems":[
         {
            "Count":2,              The system found two items that use SchemaVersion
 1.0.
            "RemainingCount":2,
            "Version":"1.0"
         },
         {
            "Count":1,              The system found one item that uses SchemaVersion
 2.0.
            "RemainingCount":1,
            "Version":"2.0"
         }
      ],
      "TotalCount":3
   },
   "TypeName":"RackSpace"
}
```

在 EventBridge 中查看清单删除操作

您可以配置 Amazon EventBridge，在用户删除自定义清单时创建事件。EventBridge 为自定义清单删除操作提供三种类型的事件：

- 实例的删除操作：特定托管式节点的自定义清单是否已成功删除。
- 删除操作摘要：删除操作的摘要。

- 针对已关闭的自定义清单类型的警告：如果用户对先前关闭的自定义清单类型版本调用了 PutInventory API 操作，则会发出警告事件。

下面是每个事件的示例。

实例的删除操作


```
{
   "version":"0",
   "id":"998c9cde-56c0-b38b-707f-0411b3ff9d11",
   "detail-type":"Inventory Resource State Change",
   "source":"aws.ssm",
   "account":"478678815555",
   "time":"2018-05-24T22:24:34Z",
   "region":"us-east-1",
   "resources":[
      "arn:aws:ssm:us-east-1:478678815555:managed-instance/i-0a5feb270fc3f0b97"
   ],
   "detail":{
      "action-status":"succeeded",
      "action":"delete",
      "resource-type":"managed-instance",
      "resource-id":"i-0a5feb270fc3f0b97",
      "action-reason":"",
      "type-name":"Custom:MyInfo"
   }
}
```


删除操作摘要


```
{
   "version":"0",
   "id":"83898300-f576-5181-7a67-fb3e45e4fad4",
   "detail-type":"Inventory Resource State Change",
   "source":"aws.ssm",
   "account":"478678815555",
   "time":"2018-05-24T22:28:25Z",
   "region":"us-east-1",
   "resources":[

   ],
   "detail":{
```

```
      "action-status":"succeeded",
      "action":"delete-summary",
      "resource-type":"managed-instance",
      "resource-id":"",
      "action-reason":"The delete for type name Custom:MyInfo was completed. The
 deletion summary is: {\"totalCount\":2,\"remainingCount\":0,\"summaryItems\":
[{\"version\":\"1.0\",\"count\":2,\"remainingCount\":0}]}",
      "type-name":"Custom:MyInfo"
   }
}
```

针对已关闭的自定义清单类型的警告

```
{
   "version":"0",
   "id":"49c1855c-9c57-b5d7-8518-b64aeeef5e4a",
   "detail-type":"Inventory Resource State Change",
   "source":"aws.ssm",
   "account":"478678815555",
   "time":"2018-05-24T22:46:58Z",
   "region":"us-east-1",
   "resources":[
      "arn:aws:ssm:us-east-1:478678815555:managed-instance/i-0ee2d86a2cfc371f6"
   ],
   "detail":{
      "action-status":"failed",
      "action":"put",
      "resource-type":"managed-instance",
      "resource-id":"i-0ee2d86a2cfc371f6",
      "action-reason":"The inventory item with type name Custom:MyInfo was sent with a
 disabled schema version 1.0. You must send a version greater than 1.0",
      "type-name":"Custom:MyInfo"
   }
}
```

可以使用以下过程为自定义清单删除操作创建 EventBridge 规则。此过程向您介绍如何创建向 Amazon SNS 主题发送自定义清单删除操作通知的规则。开始前，请验证您是否拥有 Amazon SNS 主题，如果没有请新建一个。有关更多信息，请参阅 Amazon Simple Notification Service Developer Guide 中的入门。

为删除清单操作配置 EventBridge

1. 打开位于 https://console.aws.amazon.com/events/ 的 Amazon EventBridge 控制台。
2. 在导航窗格中，选择规则。
3. 选择创建规则。
4. 为规则输入名称和描述。

   规则不能与同一区域中的另一个规则和同一事件总线上的名称相同。
5. 对于事件总线，请选择要与此规则关联的事件总线。如果您希望此规则响应来自您自己的 AWS 账户的匹配事件，请选择 defaul（默认）。当您账户中的某个 AWS 服务发出一个事件时，它始终会发送到您账户的默认事件总线。
6. 对于规则类型，选择具有事件模式的规则。
7. 选择下一步。
8. 对于事件源，选择AWS 事件或 EventBridge 合作伙伴事件。
9. 在 Event pattern（事件模式）部分，选择 Event pattern form（事件模式表单）。
10. 对于事件源，选择AWS 服务。
11. 对于 service（AWS 服务），选择 Systems Manager。
12. 对于 Event type (事件类型)，选择 Inventory (清单)。
13. 对于 Specific detail type(s)（具体详细信息类型），选择 Inventory Resource State Change（清单资源状态更改）。
14. 选择下一步。
15. 对于目标类型，选择AWS 服务。
16. 对于 Select a target（选择一个目标），选择 SNS topic（SNS 主题），然后对于 Topic（主题），选择您的主题。
17. 在 Additional settings（其他设置）部分，对于 Configure target input（配置目标输入），请验证已选择 Matched event（匹配的事件）。
18. 选择下一步。
19. （可选）为规则输入一个或多个标签。有关更多信息，请参阅 Amazon EventBridge 用户指南中的标记 Amazon EventBridge 资源。
20. 选择 Next（下一步）。
21. 查看规则详细信息并选择创建规则。

查看清单历史记录和变更跟踪

您可以使用 AWS Config 查看所有托管式节点的 AWS Systems Manager Inventory 历史记录和更改跟踪。AWS Config 提供了关于 AWS 账户 中 AWS 资源配置的详细视图。这些信息包括资源之间的关联方式以及资源以前的配置方式，让您了解资源的配置和关系如何随着的时间的推移而更改。要查看清单历史记录和更改跟踪，您必须在 AWS Config 中打开以下资源：

- SSM:ManagedInstanceInventory
- SSM:PatchCompliance
- SSM:AssociationCompliance
- SSM:FileData

> **Note**
>
> 请注意关于 Inventory 历史记录和更改跟踪的以下重要详细信息：
>
> - 如果您使用 AWS Config 来跟踪系统中的更改，则必须配置 Systems Manager Inventory 以收集 `AWS:File` 元数据，以便能够查看 AWS Config (`SSM:FileData`) 中的文件更改。如果不进行此配置，则 AWS Config 不会跟踪系统上的文件更改。
> - 通过打开 SSM:PatchCompliance 和 SSM:AssociationCompliance，您可以查看 Systems Manager Patch Manager 修补和 Systems Manager State Manager 关联的合规历史记录及更改跟踪。有关这些资源的合规性管理的更多信息，请参阅 了解有关合规性的详细信息。

以下过程介绍了如何使用 AWS Command Line Interface (AWS CLI) 在 AWS Config 中打开清单历史记录和更改跟踪记录。有关如何在 AWS Config 中选择和配置这些资源的更多信息，请参阅 AWS Config Developer Guide 中的选择哪些资源 AWS Config 记录。有关 AWS Config 定价的信息，请参阅定价。

开始之前

AWS Config 需要 AWS Identity and Access Management (IAM) 权限才能获取有关 Systems Manager 资源的配置详细信息。在以下过程中，您必须为向 Systems Manager 资源提供 AWS Config 权限的 IAM 角色指定 Amazon Resource Name (ARN)。您可以将 `AWS_ConfigRole` 托管式策略附加到分配给 AWS Config 的 IAM 角色。有关该角色的更多信息，请参阅《AWS Config 开发人员指南》中的 AWS 托管式策略：AWS_ConfigRole。要了解如何创建 IAM 角色并将 `AWS_ConfigRole` 托管式策略分配给该角色，请参阅《IAM 用户指南》中的创建向 AWS 服务 委托权限的角色。

在 AWS Config 中打开清单历史记录和更改跟踪记录

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 将以下 JSON 示例复制并粘贴到一个简单的文本文件中，并将其另存为 recordingGroup.json。

```
{
   "allSupported":false,
   "includeGlobalResourceTypes":false,
   "resourceTypes":[
      "AWS::SSM::AssociationCompliance",
      "AWS::SSM::PatchCompliance",
      "AWS::SSM::ManagedInstanceInventory",
      "AWS::SSM::FileData"
   ]
}
```

3. 运行以下命令以将 recordingGroup.json 文件加载到 AWS Config 中。

```
aws configservice put-configuration-recorder --configuration-recorder
 name=myRecorder,roleARN=arn:aws:iam::123456789012:role/myConfigRole --recording-
group file://recordingGroup.json
```

4. 运行以下命令以开始记录清单历史记录和变更跟踪。

```
aws configservice start-configuration-recorder --configuration-recorder-
name myRecorder
```

在配置历史记录和更改跟踪后，您可以通过在 Systems Manager 控制台中选择 AWS Config 按钮来深入了解某一特定托管式节点的历史记录。您可以从 Managed Instances（托管式实例）页面或 Inventory（清单）页面访问 AWS Config 按钮。根据您的监视器大小，您可能需要滚动到页面右侧以查看该按钮。

停止数据收集和删除清单数据

如果您不再需要使用 AWS Systems Manager Inventory 以查看有关 AWS 资源的元数据，可以停止数据收集，并删除已收集到的数据。本节包含以下信息。

主题

- 停止数据收集
- 删除 Inventory 资源数据同步

停止数据收集

当您最初配置 Systems Manager 以收集清单数据时，系统会创建 State Manager 关联，用于定义计划和从中收集元数据的资源。您可以通过删除任何使用 `AWS-GatherSoftwareInventory` 文档的 State Manager 关联，来停止数据收集。

删除 Inventory 关联

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 State Manager。
3. 选择一个使用 `AWS-GatherSoftwareInventory` 文档的关联，然后选择 Delete (删除)。
4. 对于任何其余使用 `AWS-GatherSoftwareInventory` 文档的关联重复步骤 3。

删除 Inventory 资源数据同步

如果您不再需要使用 AWS Systems Manager Inventory 来查看有关您的 AWS 资源的元数据，那么我们还建议您删除用于清单数据收集的资源数据同步。

删除 Inventory 资源数据同步

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Inventory (清单)。
3. 选择 Resource Data Syncs (资源数据同步)。
4. 在列表中，选择一个同步。

> **Important**
>
> 确保您选择了用于 Inventory 的同步。Systems Manager 可为多项工具的资源数据同步提供支持。如果选择的同步错误，则可能会导致 Systems Manager Explorer 或 Systems Manager Compliance 的数据聚合中断。

5. 选择 Delete（删除）

6. 对您要删除的所有其余资源数据同步，重复这些步骤。
7. 删除存储数据的 Amazon Simple Storage Service (Amazon S3) 存储桶。有关删除 Amazon S3 存储桶的信息，请参阅删除存储桶。

将自定义清单元数据分配给某个托管式节点

以下过程为您演示了使用 AWS Systems Manager PutInventory API 操作将自定义清单元数据分配给托管式节点的过程。此示例将机架位置信息分配给某个节点。有关自定义清单的更多信息，请参阅 使用自定义清单。

将自定义清单元数据分配给某个节点

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本。
2. 运行以下命令，以便将机架位置信息分配给某个节点。

   Linux

   ```
   aws ssm put-inventory --instance-id "ID" --items '[{"CaptureTime":
    "2016-08-22T10:01:01Z", "TypeName": "Custom:RackInfo", "Content":[{"RackLocation":
    "Bay B/Row C/Rack D/Shelf E"}], "SchemaVersion": "1.0"}]'
   ```

   Windows

   ```
   aws ssm put-inventory --instance-id "ID" --items
    "TypeName=Custom:RackInfo,SchemaVersion=1.0,CaptureTime=2021-05-22T10:01:01Z,Content=[{Rac
    B/Row C/Rack D/Shelf F'}]"
   ```
3. 运行以下命令以便查看此节点的自定义清单条目。

   ```
   aws ssm list-inventory-entries --instance-id ID --type-name "Custom:RackInfo"
   ```

   系统会使用类似以下形式的信息进行响应。

   ```
   {
       "InstanceId": "ID",
       "TypeName": "Custom:RackInfo",
       "Entries": [
   ```

```
        {
            "RackLocation": "Bay B/Row C/Rack D/Shelf E"
        }
    ],
    "SchemaVersion": "1.0",
    "CaptureTime": "2016-08-22T10:01:01Z"
}
```

4. 运行以下命令以查看自定义清单架构。

```
aws ssm get-inventory-schema --type-name Custom:RackInfo
```

系统会使用类似以下形式的信息进行响应。

```
{
    "Schemas": [
        {
            "TypeName": "Custom:RackInfo",
            "Version": "1.0",
            "Attributes": [
                {
                    "DataType": "STRING",
                    "Name": "RackLocation"
                }
            ]
        }
    ]
}
```

使用 AWS CLI 配置清单数据收集

以下过程为您演示了将 AWS Systems Manager Inventory 配置为从托管式节点收集元数据的过程。在配置清单收集时，应该首先创建 Systems Manager State Manager 关联。Systems Manager 将在关联运行时收集清单数据。如果您不首先创建关联，并尝试使用 Systems Manager Run Command（举例来说）调用 `aws:softwareInventory` 插件，则系统将返回以下错误：

```
The aws:softwareInventory plugin can only be invoked via ssm-associate.
```

> **Note**
>
> 一个节点一次只能配置一个清单关联。如果您为一个节点配置了两个或更多清单关联，则关联不会运行，而且系统不会收集清单数据。

快速配置 Inventory 的所有托管式节点 (CLI)

您可以在您的 AWS 账户 和当前区域中快速配置所有托管式节点来收集清单数据。这称为创建全局清单关联。要使用 AWS CLI 创建全局清单关联，请为 `instanceIds` 值使用通配符选项，如以下过程所示。

在您的 AWS 账户 和当前区域中为所有托管式节点配置清单 (CLI)

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 运行以下命令。

   Linux & macOS

   ```
   aws ssm create-association \
   --name AWS-GatherSoftwareInventory \
   --targets Key=InstanceIds,Values=* \
   --schedule-expression "rate(1 day)" \
   --parameters
    applications=Enabled,awsComponents=Enabled,customInventory=Enabled,instanceDetailedInfo
   ```

   Windows

   ```
   aws ssm create-association ^
   --name AWS-GatherSoftwareInventory ^
   --targets Key=InstanceIds,Values=* ^
   --schedule-expression "rate(1 day)" ^
   --parameters
    applications=Enabled,awsComponents=Enabled,customInventory=Enabled,instanceDetailedInfo
   ```

> **Note**
>
> 此命令不允许 Inventory 收集 Windows 注册表或文件的元数据。要为这些数据类型启用清单，请使用下一个过程。

在托管式节点上手动配置 Inventory (CLI)

通过以下过程，使用节点 ID 或标签在您的托管式节点上手动配置 AWS Systems Manager Inventory。

手动配置 Inventory 的托管式节点 (CLI)

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 运行以下命令，创建一个在节点上运行 Systems Manager Inventory 的 State Manager 关联。将每个*#######*替换为您自己的信息。此命令将服务配置为每六小时运行一次，并从节点收集网络配置、Windows 更新和应用程序元数据。

   Linux & macOS

   ```
   aws ssm create-association \
   --name "AWS-GatherSoftwareInventory" \
   --targets "Key=instanceids,Values=an_instance_ID" \
   --schedule-expression "rate(240 minutes)" \
   --output-location "{ \"S3Location\": { \"OutputS3Region\": \"region_ID,
    for example us-east-2\", \"OutputS3BucketName\": \"amzn-s3-demo-bucket\",
    \"OutputS3KeyPrefix\": \"Test\" } }" \
   --parameters "networkConfig=Enabled,windowsUpdates=Enabled,applications=Enabled"
   ```

   Windows

   ```
   aws ssm create-association ^
   --name "AWS-GatherSoftwareInventory" ^
   --targets "Key=instanceids,Values=an_instance_ID" ^
   --schedule-expression "rate(240 minutes)" ^
   --output-location "{ \"S3Location\": { \"OutputS3Region\": \"region_ID,
    for example us-east-2\", \"OutputS3BucketName\": \"amzn-s3-demo-bucket\",
    \"OutputS3KeyPrefix\": \"Test\" } }" ^
   --parameters "networkConfig=Enabled,windowsUpdates=Enabled,applications=Enabled"
   ```

系统会使用类似以下形式的信息进行响应。

```
{
    "AssociationDescription": {
        "ScheduleExpression": "rate(240 minutes)",
        "OutputLocation": {
            "S3Location": {
                "OutputS3KeyPrefix": "Test",
                "OutputS3BucketName": "Test bucket",
                "OutputS3Region": "us-east-2"
            }
        },
        "Name": "The name you specified",
        "Parameters": {
            "applications": [
                "Enabled"
            ],
            "networkConfig": [
                "Enabled"
            ],
            "windowsUpdates": [
                "Enabled"
            ]
        },
        "Overview": {
            "Status": "Pending",
            "DetailedStatus": "Creating"
        },
        "AssociationId": "1a2b3c4d5e6f7g-1a2b3c-1a2b3c-1a2b3c-1a2b3c4d5e6f7g",
        "DocumentVersion": "$DEFAULT",
        "LastUpdateAssociationDate": 1480544990.06,
        "Date": 1480544990.06,
        "Targets": [
            {
                "Values": [
                   "i-02573cafcfEXAMPLE"
                ],
                "Key": "InstanceIds"
            }
        ]
    }
```

```
}
```

您可以搭配使用 `Targets` 参数和 EC2 标签来确定大型目标节点组。请参阅以下示例。

Linux & macOS

```
aws ssm create-association \
--name "AWS-GatherSoftwareInventory" \
--targets "Key=tag:Environment,Values=Production" \
--schedule-expression "rate(240 minutes)" \
--output-location "{ \"S3Location\": { \"OutputS3Region\": \"us-east-2\",
 \"OutputS3BucketName\": \"amzn-s3-demo-bucket\", \"OutputS3KeyPrefix\": \"Test
\" } }" \
--parameters "networkConfig=Enabled,windowsUpdates=Enabled,applications=Enabled"
```

Windows

```
aws ssm create-association ^
--name "AWS-GatherSoftwareInventory" ^
--targets "Key=tag:Environment,Values=Production" ^
--schedule-expression "rate(240 minutes)" ^
--output-location "{ \"S3Location\": { \"OutputS3Region\": \"us-east-2\",
 \"OutputS3BucketName\": \"amzn-s3-demo-bucket\", \"OutputS3KeyPrefix\": \"Test
\" } }" ^
--parameters "networkConfig=Enabled,windowsUpdates=Enabled,applications=Enabled"
```

您还可以搭配使用 `files` 和 `windowsRegistry` 清单类型与表达式来清点 Windows Server 节点上的清单文件和 Windows 注册表项。有关这些清单类型的更多信息，请参阅 使用文件和 Windows 注册表清单。

Linux & macOS

```
aws ssm create-association \
--name "AWS-GatherSoftwareInventory" \
--targets "Key=instanceids,Values=i-0704358e3a3da9eb1" \
--schedule-expression "rate(240 minutes)" \
--parameters '{"files":["[{\"Path\": \"C:\\Program Files\", \"Pattern\":
 [\"*.exe\"], \"Recursive\": true}]"], "windowsRegistry": ["[{\"Path\":
\"HKEY_LOCAL_MACHINE\\Software\\Amazon\", \"Recursive\":true}]"]}' \
--profile dev-pdx
```

Windows

```
aws ssm create-association ^
--name "AWS-GatherSoftwareInventory" ^
--targets "Key=instanceids,Values=i-0704358e3a3da9eb1" ^
--schedule-expression "rate(240 minutes)" ^
--parameters '{"files":["[{\"Path\": \"C:\\Program Files\", \"Pattern\":
 [\"*.exe\"], \"Recursive\": true}]"], "windowsRegistry": ["[{\"Path\":
\"HKEY_LOCAL_MACHINE\\Software\\Amazon\", \"Recursive\":true}]"]}' ^
--profile dev-pdx
```

3. 运行以下命令查看关联状态。

```
aws ssm describe-instance-associations-status --instance-id an_instance_ID
```

系统会使用类似以下形式的信息进行响应。

```
{
"InstanceAssociationStatusInfos": [
         {
            "Status": "Pending",
            "DetailedStatus": "Associated",
            "Name": "reInvent2016PolicyDocumentTest",
            "InstanceId": "i-1a2b3c4d5e6f7g",
            "AssociationId": "1a2b3c4d5e6f7g-1a2b3c-1a2b3c-1a2b3c-1a2b3c4d5e6f7g",
            "DocumentVersion": "1"
        }
]
}
```

演练：使用资源数据同步聚合清单数据

以下演练介绍了如何使用 AWS Command Line Interface (AWS CLI) 为 AWS Systems Manager Inventory 创建资源数据同步配置。资源数据同步可将所有托管式节点中的清单数据自动移植到中央 Amazon Simple Storage Service (Amazon S3) 存储桶。每当发现新的清单数据时，同步操作会自动更新中央 Amazon S3 存储桶中的数据。

此演练还介绍了如何使用 Amazon Athena 和 Amazon QuickSight 查询及分析聚合数据。有关在 AWS Management Console 中使用 Systems Manager 创建资源数据同步的信息，请参阅 演练：使用资源数

据同步聚合清单数据。有关在 AWS Management Console 中使用 Systems Manager 从多个 AWS 区域 和账户中查询清单的信息，请参阅 查询多个区域和账户的清单数据。

> **Note**
>
> 本演练包括有关如何使用 AWS Key Management Service (AWS KMS) 加密同步的信息。Inventory 不收集任何用户特定、专有或敏感数据，因此加密是可选的。有关 AWS KMS 的更多信息，请参阅 AWS Key Management Service Developer Guide。

开始前的准备工作

在开始本节中的演练之前，请审核或完成以下任务：

- 从托管式节点收集清单数据。根据本演练中的 Amazon Athena 和 Amazon QuickSight 两节的目的，我们建议您收集应用程序数据。有关如何收集清单数据的更多信息，请参阅 配置清单收集 或 使用 AWS CLI 配置清单数据收集。
- （可选）如果清单数据存储在使用 AWS Key Management Service (AWS KMS) 加密的 Amazon Simple Storage Service (Amazon S3) 存储桶中，您还必须配置 IAM 账户和 `Amazon-GlueServiceRoleForSSM` 服务角色，以便进行 AWS KMS 加密。如果您没有配置 IAM 账户和此角色，则当您选择控制台中的 Detailed View (详细视图) 选项卡时，Systems Manager 将显示 `Cannot load Glue tables`。有关更多信息，请参阅 （可选）配置查看 AWS KMS 加密数据的权限。
- （可选）如果希望使用 AWS KMS 来加密资源数据同步，则您必须创建包含以下策略的新密钥，或者必须更新现有密钥并向其添加此策略。

  JSON

```
{
    "Version": "2012-10-17",
    "Id": "ssm-access-policy",
    "Statement": [
        {
            "Sid": "ssm-access-policy-statement",
            "Action": [
                "kms:GenerateDataKey"
            ],
            "Effect": "Allow",
            "Principal": {
                "Service": "ssm.amazonaws.com"
```

```
            },
            "Resource": "arn:aws:kms:us-east-2:123456789012:key/KMS_key_id",
            "Condition": {
                "StringLike": {
                    "aws:SourceAccount": "123456789012"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:ssm:*:123456789012:resource-data-
sync/*"
                }
            }
        }
    ]
}
```

为 Inventory 创建资源数据同步

1. 通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
2. 创建用来存储聚合清单数据的存储桶。有关更多信息，请参阅《Amazon Simple Storage Service 用户指南》中的创建存储桶。请记下存储桶名称和创建此存储桶的 AWS 区域。
3. 创建存储桶后，选择 Permissions 选项卡，然后选择 Bucket Policy。
4. 将下面的存储桶策略复制并粘贴到策略编辑器中。将 amzn-s3-demo-bucket 和 *account-id* 分别替换为您创建的 Amazon S3 存储桶的名称和有效的 AWS 账户 ID。添加多个账户时，请为每个账户添加一个附加条件字符串和 ARN。添加一个账户时，请从示例中删除附加占位符。或者，将 *bucket-prefix* 替换为 Amazon S3 前缀（子目录）的名称。如果您未创建前缀，则从该策略的 ARN 中删除 *bucket-prefix/*。

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": " SSMBucketDelivery",
            "Effect": "Allow",
            "Principal": {
                "Service": "ssm.amazonaws.com"
            },
            "Action": "s3:PutObject",
```

```
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/bucket-prefix/*/
accountid=account-id/*"
            ],
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "bucket-owner-full-control",
                    "aws:SourceAccount": [
                        "account-id1",
                        "account-id2",
                        "account-id3",
                        "account-id4"
                    ]
                },
                "ArnLike": {
                    "aws:SourceArn": [
                        "arn:aws:ssm:*:111122223333:resource-data-sync/*",
                        "arn:aws:ssm:*:111122223333:resource-data-sync/*",
                        "arn:aws:ssm:*:111122223333:resource-data-sync/*",
                        "arn:aws:ssm:*:111122223333:resource-data-sync/*"
                    ]
                }
            }
        }
    ]
}
```

5. （可选）如果要加密同步，则必须将以下条件添加到上一步中列出的策略中。将这些内容添加在 StringEquals 部分中。

```
"s3:x-amz-server-side-encryption":"aws:kms",
"s3:x-amz-server-side-encryption-aws-kms-key-
id":"arn:aws:kms:region:account_ID:key/KMS_key_ID"
```

示例如下：

```
"StringEquals": {
          "s3:x-amz-acl": "bucket-owner-full-control",
          "aws:SourceAccount": "account-id",
          "s3:x-amz-server-side-encryption":"aws:kms",
          "s3:x-amz-server-side-encryption-aws-kms-key-
id":"arn:aws:kms:region:account_ID:key/KMS_key_ID"
```

```
        }
```

6. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关更多信息，请参阅安装或更新 AWS CLI 的最新版本。

7. （可选）如果要加密同步，请运行以下命令，以验证存储桶策略是否强制执行 AWS KMS 密钥要求。将每个*#######*替换为您自己的信息。

   Linux & macOS

   ```
   aws s3 cp ./A_file_in_the_bucket s3://amzn-s3-demo-bucket/prefix/ \
   --sse aws:kms \
   --sse-kms-key-id "arn:aws:kms:region:account_ID:key/KMS_key_id" \
   --region region, for example, us-east-2
   ```

   Windows

   ```
   aws s3 cp ./A_file_in_the_bucket s3://amzn-s3-demo-bucket/prefix/ ^
       --sse aws:kms ^
       --sse-kms-key-id "arn:aws:kms:region:account_ID:key/KMS_key_id" ^
       --region region, for example, us-east-2
   ```

8. 运行以下命令，以使用您在此过程开始时创建的 Amazon S3 存储桶创建资源数据同步配置。此命令从您登录的 AWS 区域创建同步。

   > **Note**
   >
   > 如果同步和目标 Amazon S3 存储桶位于不同区域中，您可能需要支付数据传输费用。有关更多信息，请参阅 Amazon S3 定价。

   Linux & macOS

   ```
   aws ssm create-resource-data-sync \
   --sync-name a_name \
   --s3-destination "BucketName=amzn-s3-demo-bucket,Prefix=prefix_name,
    if_specified,SyncFormat=JsonSerDe,Region=bucket_region"
   ```

Windows

```
aws ssm create-resource-data-sync ^
--sync-name a_name ^
--s3-destination "BucketName=amzn-s3-demo-bucket,Prefix=prefix_name,
 if_specified,SyncFormat=JsonSerDe,Region=bucket_region"
```

您可以使用 region 参数指定创建同步配置的位置。在下例中，来自 us-west-1 区域的清单数据将同步到 us-west-2 区域内的 Amazon S3 存储桶中。

Linux & macOS

```
aws ssm create-resource-data-sync \
    --sync-name InventoryDataWest \
    --s3-destination "BucketName=amzn-s3-demo-
bucket,Prefix=HybridEnv,SyncFormat=JsonSerDe,Region=us-west-2"
    --region us-west-1
```

Windows

```
aws ssm create-resource-data-sync ^
--sync-name InventoryDataWest ^
--s3-destination "BucketName=amzn-s3-demo-
bucket,Prefix=HybridEnv,SyncFormat=JsonSerDe,Region=us-west-2" ^ --region us-
west-1
```

（可选）如果要使用 AWS KMS 加密同步，请运行以下命令创建同步。如果您加密同步，则 AWS KMS 密钥和 Amazon S3 存储桶必须位于同一区域中。

Linux & macOS

```
aws ssm create-resource-data-sync \
--sync-name sync_name \
--s3-destination "BucketName=amzn-s3-demo-bucket,Prefix=prefix_name,
 if_specified,SyncFormat=JsonSerDe,AWSKMSKeyARN=arn:aws:kms:region:account_ID:key/
KMS_key_ID,Region=bucket_region" \
--region region
```

Windows

```
aws ssm create-resource-data-sync ^
--sync-name sync_name ^
--s3-destination "BucketName=amzn-s3-demo-bucket,Prefix=prefix_name,
 if_specified,SyncFormat=JsonSerDe,AWSKMSKeyARN=arn:aws:kms:region:account_ID:key/
KMS_key_ID,Region=bucket_region" ^
--region region
```

9. 运行以下命令查看同步配置的状态。

```
aws ssm list-resource-data-sync
```

如果您在其他区域中创建了同步配置，则必须指定 region 参数，如下例所示。

```
aws ssm list-resource-data-sync --region us-west-1
```

10. 在成功创建同步配置后，检查 Amazon S3 中的目标存储桶。清单数据应在几分钟内显示。

使用 Amazon Athena 中的数据

以下部分介绍如何在 Amazon Athena 中查看和查询数据。在开始之前，我们建议您首先了解 Athena。有关更多信息，请参阅 Amazon Athena 用户指南中的什么是 Amazon Athena？和使用数据。

在 Amazon Athena 中查看和查询数据

1. 从 https://console.aws.amazon.com/athena/ 打开 Athena 控制台。
2. 将以下语句复制并粘贴到查询编辑器中，然后选择 Run Query。

```
CREATE DATABASE ssminventory
```

系统将创建一个名为 ssminventory 的数据库。

3. 将以下语句复制并粘贴到查询编辑器中，然后选择 Run Query。将 amzn-s3-demo-bucket 和 *bucket_prefix* 分别替换为 Amazon S3 目标的名称和前缀。

```
CREATE EXTERNAL TABLE IF NOT EXISTS ssminventory.AWS_Application (
Name string,
ResourceId string,
```

```
ApplicationType string,
Publisher string,
Version string,
InstalledTime string,
Architecture string,
URL string,
Summary string,
PackageId string
)
PARTITIONED BY (AccountId string, Region string, ResourceType string)
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
WITH SERDEPROPERTIES (
  'serialization.format' = '1'
) LOCATION 's3://amzn-s3-demo-bucket/bucket_prefix/AWS:Application/'
```

4. 将以下语句复制并粘贴到查询编辑器中，然后选择 Run Query。

```
MSCK REPAIR TABLE ssminventory.AWS_Application
```

系统将对表进行分区。

> **Note**
>
> 如果您从其他 AWS 区域或 AWS 账户中创建资源数据同步，则必须再次运行此命令以更新分区。您可能还需要更新 Amazon S3 存储桶策略。

5. 要预览数据，请选择 AWS_Application 表旁边的视图图标。



6. 将以下语句复制并粘贴到查询编辑器中，然后选择 Run Query。

```
SELECT a.name, a.version, count( a.version) frequency
from aws_application a where
a.name = 'aws-cfn-bootstrap'
group by a.name, a.version
order  by frequency desc
```

该查询将返回不同版本的 aws-cfn-bootstrap（它是 Linux、macOS 和 Windows Server 的 Amazon Elastic Compute Cloud (Amazon EC2) 实例上出现的 AWS 应用程序）的计数。

7. 分别将以下语句复制并粘贴到查询编辑器中，将 amzn-s3-demo-bucket 和 *bucket-prefix* 分别替换为 Amazon S3 的相应信息，然后选择运行查询。这些语句将在 Athena 中设置其他清单表。

```
CREATE EXTERNAL TABLE IF NOT EXISTS ssminventory.AWS_AWSComponent (
 `ResourceId` string,
  `Name` string,
  `ApplicationType` string,
  `Publisher` string,
  `Version` string,
  `InstalledTime` string,
  `Architecture` string,
  `URL` string
)
PARTITIONED BY (AccountId string, Region string, ResourceType string)
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
WITH SERDEPROPERTIES (
  'serialization.format' = '1'
) LOCATION 's3://amzn-s3-demo-bucket/bucket-prefix/AWS:AWSComponent/'
```

```
MSCK REPAIR TABLE ssminventory.AWS_AWSComponent
```

```
CREATE EXTERNAL TABLE IF NOT EXISTS ssminventory.AWS_WindowsUpdate (
  `ResourceId` string,
  `HotFixId` string,
  `Description` string,
  `InstalledTime` string,
  `InstalledBy` string
)
PARTITIONED BY (AccountId string, Region string, ResourceType string)
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
WITH SERDEPROPERTIES (
  'serialization.format' = '1'
) LOCATION 's3://amzn-s3-demo-bucket/bucket-prefix/AWS:WindowsUpdate/'
```

```
MSCK REPAIR TABLE ssminventory.AWS_WindowsUpdate
```

```
CREATE EXTERNAL TABLE IF NOT EXISTS ssminventory.AWS_InstanceInformation (
  `AgentType` string,
  `AgentVersion` string,
  `ComputerName` string,
```

```
  `IamRole` string,
  `InstanceId` string,
  `IpAddress` string,
  `PlatformName` string,
  `PlatformType` string,
  `PlatformVersion` string
)
PARTITIONED BY (AccountId string, Region string, ResourceType string)
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
WITH SERDEPROPERTIES (
  'serialization.format' = '1'
) LOCATION 's3://amzn-s3-demo-bucket/bucket-prefix/AWS:InstanceInformation/'
```

```
MSCK REPAIR TABLE ssminventory.AWS_InstanceInformation
```

```
CREATE EXTERNAL TABLE IF NOT EXISTS ssminventory.AWS_Network (
  `ResourceId` string,
  `Name` string,
  `SubnetMask` string,
  `Gateway` string,
  `DHCPServer` string,
  `DNSServer` string,
  `MacAddress` string,
  `IPV4` string,
  `IPV6` string
)
PARTITIONED BY (AccountId string, Region string, ResourceType string)
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
WITH SERDEPROPERTIES (
  'serialization.format' = '1'
) LOCATION 's3://amzn-s3-demo-bucket/bucket-prefix/AWS:Network/'
```

```
MSCK REPAIR TABLE ssminventory.AWS_Network
```

```
CREATE EXTERNAL TABLE IF NOT EXISTS ssminventory.AWS_PatchSummary (
  `ResourceId` string,
  `PatchGroup` string,
  `BaselineId` string,
  `SnapshotId` string,
  `OwnerInformation` string,
  `InstalledCount` int,
```

```
  `InstalledOtherCount` int,
  `NotApplicableCount` int,
  `MissingCount` int,
  `FailedCount` int,
  `OperationType` string,
  `OperationStartTime` string,
  `OperationEndTime` string
)
PARTITIONED BY (AccountId string, Region string, ResourceType string)
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
WITH SERDEPROPERTIES (
  'serialization.format' = '1'
) LOCATION 's3://amzn-s3-demo-bucket/bucket-prefix/AWS:PatchSummary/'
```

```
MSCK REPAIR TABLE ssminventory.AWS_PatchSummary
```

使用 Amazon QuickSight 中的数据

以下部分提供了概述，包含用于在 Amazon QuickSight 中构建可视化内容的链接。

在 Amazon QuickSight 中构建可视化内容

1. 注册 Amazon QuickSight，然后登录到 QuickSight 控制台。
2. 从 AWS_Application 表和您创建的任何其他表创建数据集。有关更多信息，请参阅使用 Amazon Athena 数据创建数据集。
3. 联接表。例如，您可以联接 AWS_InstanceInformation 中的 instanceid 列，因为它与其他清单表中的 resourceid 列匹配。有关联接表的更多信息，请参阅联接表。
4. 构建可视化内容。有关更多信息，请参阅处理 Amazon QuickSight 视觉对象。

对 Systems Manager Inventory 的问题进行故障排除

本主题包括有关如何排除 AWS Systems Manager Inventory 中常见错误或问题的信息。如果您在 Systems Manager 中查看节点时遇到问题，请参阅 排除托管式节点可用性的问题。

主题

- 不支持多次应用所有使用文档“AWS-GatherSoftwareInventory”的关联
- 清单执行状态永远不会退出待处理状态
- AWS-ListWindowsInventory 文档无法运行

- 控制台不显示 Inventory“控制面板”|“详细视图”|“设置”选项卡
- UnsupportedAgent
- Skipped
- 失败
- Amazon EC2 实例的库存合规性失败
- S3 存储桶对象包含旧数据

不支持多次应用所有使用文档“**AWS-GatherSoftwareInventory**”的关联

错误消息 `Multiple apply all associations with document 'AWS-GatherSoftwareInventory' are not supported` 表示：您尝试在其中为所有节点配置 Inventory 关联的一个或多个 AWS 区域，已配置有一个适用于所有节点的清单关联。如有必要，您可以删除所有节点的现有清单关联，然后创建新的清单关联。要查看现有清单关联，在 Systems Manager 控制台中选择 State Manager，然后找到使用 `AWS-GatherSoftwareInventory` SSM 文档的关联。如果跨多个区域创建了所有节点的现有清单关联，并且您希望创建一个新的清单关联，则必须从现有清单关联存在的每个区域中删除它。

清单执行状态永远不会退出待处理状态

清单收集永远不会退出 `Pending` 状态的原因有两个：

- 所选 AWS 区域 中没有节点：

  如果您使用 Systems Manager Quick Setup 创建全局清单关联，并且如果所选区域中没有可用节点，则清单关联（`AWS-GatherSoftwareInventory` 文档）的状态将显示 `Pending`。
- 权限不足

  如果一个或多个节点没有运行 Systems Manager Inventory 的权限，清单关联将显示 `Pending`。验证并确保 AWS Identity and Access Management (IAM) 实例配置文件包含 AmazonSSMManagedInstanceCore 托管式策略。有关如何将此策略添加到实例配置文件的信息，请参阅EC2 实例权限的替代配置。

  实例配置文件必须至少拥有以下 IAM 权限。

  JSON

```
{
    "Version": "2012-10-17",
```

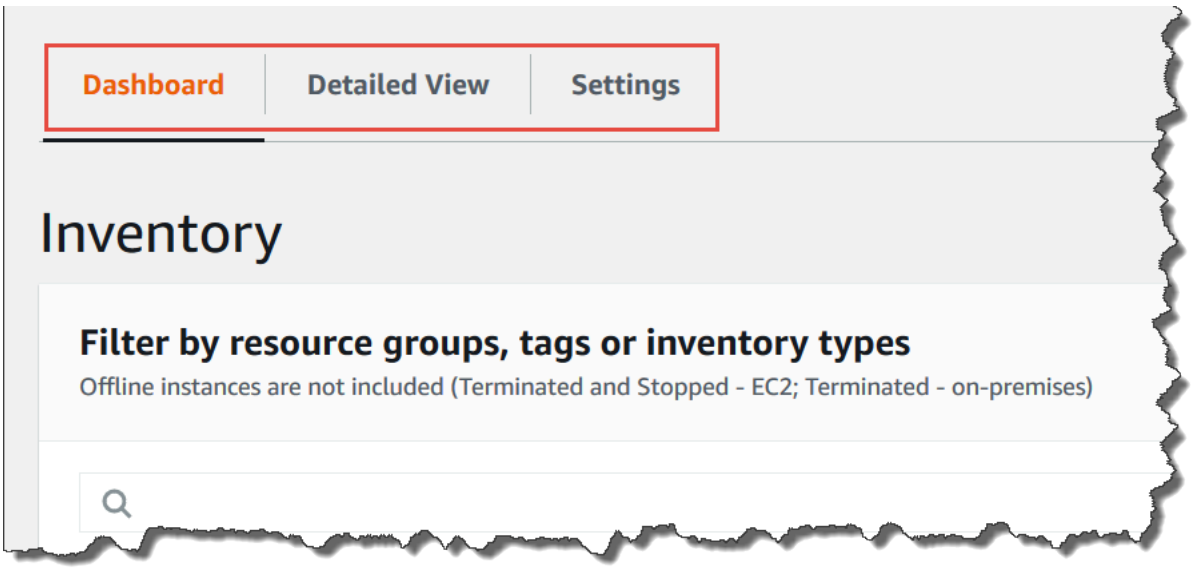
```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeAssociation",
                "ssm:ListAssociations",
                "ssm:ListInstanceAssociations",
                "ssm:PutInventory",
                "ssm:PutComplianceItems",
                "ssm:UpdateAssociationStatus",
                "ssm:UpdateInstanceAssociationStatus",
                "ssm:UpdateInstanceInformation",
                "ssm:GetDocument",
                "ssm:DescribeDocument"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS-ListWindowsInventory 文档无法运行

AWS-ListWindowsInventory 文档已被弃用。请勿使用此文档收集清单，而应使用 配置清单收集 中描述的其中一个进程。

控制台不显示 Inventory“控制面板”|“详细视图”|“设置”选项卡

仅在提供 Amazon Athena 的 AWS 区域 中提供了 Inventory Detailed View (详细视图) 页面。如果在 Inventory 页面上未显示以下选项卡，则意味着 Athena 在该区域中不可用，并且您无法使用 Detailed View (详细视图) 查询数据。



UnsupportedAgent

如果清单关联的详细状态显示为 UnsupportedAgent（不支持的代理），并且 Association status（关联状态）显示为 Failed（失败），则说明托管式节点上的 AWS Systems Manager SSM Agent 版本不正确。例如，要创建全局清单关联（用于清点 AWS 账户 中的所有节点），您必须使用 SSM Agent 版本 2.0.790.0 或更高版本。您可以在 Managed Instances（托管式实例）页面的 Agent version（代理版本）列中查看各个节点上运行的代理版本。有关如何在节点上更新 SSM Agent 的信息，请参阅 使用 Run Command 更新 SSM Agent。

Skipped

如果某个节点的清单关联状态显示为跳过，则意味着该节点已经有更高优先级清单关联在运行。当多个清单关联可能应用于同一个托管节点时，Systems Manager 会遵循特定的优先顺序。

清单关联优先顺序

Systems Manager 按照以下优先顺序应用清单关联：

1. Quick Setup清单关联：使用 Quick Setup 和统一控制台创建的关联。这些关联的名称以 `AWS-QuickSetup-SSM-CollectInventory-` 开头，并以所有托管节点为目标。
2. 显式清单关联：使用以下方法，以特定托管节点为目标的关联：
   - 实例 ID
   - 标签键值对
   - AWS 资源组

3. 全局清单关联：以所有托管节点为目标（使用 `--targets "Key=InstanceIds,Values=*"`），但并非通过 Quick Setup 创建的关联。

常见场景

场景 1：Quick Setup 关联优先于显式关联

- 您拥有一个以所有实例为目标的清单关联 Quick Setup
- 您通过标签创建一个以特定托管节点为目标的手动关联
- 结果：手动关联显示为 `Skipped`，详细状态为 `OverriddenByExplicitInventoryAssociation`
- 该 Quick Setup 关联继续从所有实例收集清单

场景 2：显式关联优先于全局关联

- 您拥有一个以所有实例为目标的全局清单关联（并非通过 Quick Setup 创建）
- 您创建一个以特定实例为目标的关联
- 结果：对于特定的目标实例，全局关联显示 `Skipped`
- 显式关联运行在目标实例上

解决步骤

如果您想使用自己的清单关联而非 Quick Setup：

1. 识别 Quick Setup 关联：在 Systems Manager 控制台中，转至 State Manager 并查找名字以 `AWS-QuickSetup-SSM-CollectInventory-` 开头的关联。
2. 删除 Quick Setup 配置：
   - 转至 Systems Manager 控制台中的 Quick Setup。
   - 查找清单收集配置。
   - 删除 Quick Setup 配置（这将移除已关联的清单关联）。

   > **Note**
   >
   > 无需手动删除通过 Quick Setup 创建的关联。

3. 验证关联是否运行：删除 Quick Setup 配置后，显式清单关联应该已成功开始运行。

如果您要修改现有行为：

- 要查看现有所有清单关联，请在 Systems Manager 控制台中选择 State Manager，然后找到使用 `AWS-GatherSoftwareInventory` SSM 文档的关联。
- 请谨记，每个托管节点一次只能有一个活动的清单关联。

> **Important**
>
> - 当为跳过的节点分配的（更高优先级）清单关联运行时，仍会从跳过的节点收集清单数据。
> - Quick Setup 清单关联优先于其他所有类型，即使是具有显式目标的类型。
> - 当任何关联被更高优先级的关联覆盖时，无论关联类型如何，都会显示详细的状态消息 `OverriddenByExplicitInventoryAssociation`。

失败

如果某个节点的清单关联状态显示 Failed（失败），则意味着节点上分配了多个清单关联。一个节点一次只能分配一个清单关联。清单关联使用 `AWS-GatherSoftwareInventory` AWS Systems Manager 文档（SSM 文档）。可以使用 AWS Command Line Interface (AWS CLI) 运行以下命令来查看节点的关联列表。

```
aws ssm describe-instance-associations-status --instance-id instance-ID
```

Amazon EC2 实例的库存合规性失败

如果为 Amazon Elastic Compute Cloud (Amazon EC2) 实例分配多个库存关联，则该实例的库存合规性可能会失败。

要解决此问题，请删除分配给实例的一个或多个库存关联。有关更多信息，请参阅删除关联。

> **Note**
>
> 如果您为一个托管式节点创建多个清单关联，请注意以下行为：
>
> - 可为每个节点分配一个以所有节点为目标的清单关联 (--targets "Key=InstanceIds,Values=*")。
> - 还可以为每个节点分配一个特定关联，该关联使用标签键-值对或 AWS 资源组。

- 如果为某个节点分配了多个清单关联，则对于尚未运行的关联，状态将显示为 Skipped（已跳过）。最近运行的关联将显示清单关联的实际状态。
- 如果为某个节点分配了多个清单关联，并且每个关联都使用标签键 - 值对，则这些清单关联将因标签冲突而无法在该节点上运行。关联仍会在没有标签键 - 值冲突的节点上运行。

S3 存储桶对象包含旧数据

清单关联成功并发现新数据后，Amazon S3 存储桶对象中的数据将会更新。关联运行但失败时，每个节点的 Amazon S3 存储桶对象都会更新，但在这种情况下，对象内的数据不会更新。只有当关联成功运行时，Amazon S3 存储桶对象内的数据才会更新。清单关联失败时，将在 Amazon S3 存储桶对象中看到旧数据。

AWS Systems Manager Patch Manager

Patch Manager是 AWS Systems Manager 中的一项工具，可使用安全相关更新及其他类型的更新自动执行修补托管式节点的过程。

> **Note**
>
> Systems Manager 为Quick Setup（AWS Systems Manager 中的一项工具）中的补丁策略提供支持。建议使用补丁策略来配置修补操作。使用单一补丁策略配置，您可以为贵组织中所有区域的所有账户定义修补、仅为所选的账户和区域定义修补，或为单个账户区域对定义修补。有关更多信息，请参阅 Quick Setup 中的补丁策略配置。

您可以使用 Patch Manager 来应用操作系统和应用程序的补丁。（在 Windows Server 上，应用程序支持仅限于更新 Microsoft 发布的应用程序。）您可以使用 Patch Manager 在 Windows 节点上安装 Service Pack，并在 Linux 节点上执行次要版本升级。可以按操作系统类型对 Amazon Elastic Compute Cloud（Amazon EC2）实例、边缘设备或本地服务器和虚拟机（VM）的实例集进行修补。这包括多个操作系统的支持版本，如 Patch Manager 先决条件中所列。您可以扫描实例以仅查看缺失补丁的报告，也可以扫描并自动安装所有缺失的补丁。要开始使用 Patch Manager，请打开 Systems Manager 控制台。在导航窗格中，请选择 Patch Manager。

在 Patch Manager 中提供补丁之前，AWS 不会测试这些补丁。此外，Patch Manager 不支持升级操作系统的主要版本，例如，将 Windows Server 2016 升级到 Windows Server 2019，或将 SUSE Linux Enterprise Server (SLES) 12.0 升级到 SLES 15.0。

对于报告修补程序严重性级别的基于 Linux 的操作系统类型，Patch Manager 将软件发布商报告的严重性级别用于更新通知或单个修补程序。Patch Manager 不会从第三方来源（例如常见漏洞评分系统 (CVSS)），或者国家漏洞数据库 (NVD) 发布的指标中获取严重性级别。

我的组织如何从 Patch Manager 获益？

Patch Manager 会使用安全相关更新及其他类型的更新自动执行修补托管式节点的过程。此功能有多个重要的好处：

- 集中控制修补：使用补丁策略，您可以为组织中所有区域的所有账户、特定账户和区域或者单个账户区域对设置定期修补操作。
- 灵活的修补操作：您可以选择通过扫描实例来仅查看缺失的补丁报告，也可以扫描并自动安装所有缺失的补丁。
- 全面的合规性报告：扫描操作完成后，您可以查看有关哪些托管式节点的补丁安装不合规以及缺失哪些补丁的详细信息。
- 跨平台支持：Patch Manager 支持多种操作系统，包括各种 Linux 发行版、macOS 和 Windows Server。
- 自定义补丁基准：您可以通过自定义补丁基准来指定批准安装的补丁，从而定义组织的补丁合规性要求。
- 与其他 AWS 服务集成：Patch Manager 可与 AWS Organizations、AWS Security Hub、AWS CloudTrail 和 AWS Config 集成，从而加强管理和安全性。
- 确定性升级：通过提供 Amazon Linux 2023 等操作系统的多版本存储库，支持确定性升级要求。

谁应该使用 Patch Manager？

Patch Manager 适合以下人员使用：

- 需要在跨托管式节点群保持补丁合规性的 IT 管理员
- 需要清晰了解整个基础设施环境中补丁合规状态的运营经理
- 需要大规模实施自动化修补解决方案的云架构师
- 需要将补丁安装功能集成到运营工作流的 DevOps 工程师
- 部署了多账户/多区域环境并且需要集中式补丁管理的组织
- 任何负责维护 AWS 托管式节点、边缘设备、本地服务器和虚拟机安全状况和运行状况的人员

Patch Manager 的主要功能是什么？

Patch Manager 具有多个重要的功能：

- 补丁策略：通过与 AWS Organizations 集成，使用单个策略来配置跨多个 AWS 账户和区域的修补操作。
- 自定义补丁基准：定义在补丁发布后几天内自动批准补丁的规则，以及已批准和已拒绝的补丁列表。
- 多种修补方法：根据自己的具体需求选择补丁策略、维护时段或按需“立即修补”操作。
- 合规性报告：生成有关补丁合规状态的详细报告，并且可将报告以 CSV 格式发送到 Amazon S3 存储桶。
- 跨平台支持：可跨 Windows Server、多种 Linux 发行版和 macOS 修补操作系统和应用程序。
- 灵活计划：使用自定义 CRON 或 Rate 表达式设置不同的扫描和补丁安装计划。
- 生命周期挂钩：使用 Systems Manager 文档在修补操作之前和之后运行自定义脚本。
- 侧重安全：默认情况下，Patch Manager 侧重于安全相关更新，而不是安装所有可用补丁。
- 速率控制：为修补操作配置并发和错误阈值，从而尽可能减少对运营的影响。

Patch Manager中的合规性是什么？

AWS、操作系统 (OS) 供应商或安全咨询公司等第三方未定义 Systems Manager 实例集中托管节点的补丁合规性构成要素的基准测试。

相反，您可以在补丁基准中定义补丁合规性对于您的组织或账户中的托管节点意味着什么。补丁基准是一种配置，它指定必须在托管节点上安装哪些补丁的规则。当托管节点的所有补丁都符合您在补丁基准中指定的批准标准时，该节点即符合补丁要求。

请注意，符合补丁基准并不意味着托管节点一定是安全的。“合规”意味着补丁基准定义的可用且已批准的补丁已安装在节点上。托管节点的整体安全性由Patch Manager范围之外的许多因素决定。有关更多信息，请参阅 安全性。

每个补丁基准都是针对特定受支持的操作系统 (OS) 类型的配置，例如 Red Hat Enterprise Linux (RHEL)、macOS 或 Windows Server。补丁基准可以为所有受支持的操作系统版本定义修补规则，或者也可以仅限于您指定的版本（例如 RHEL 6.10、RHEL 7.8 和 RHEL 9.3）。

在补丁基准中，您可以指定批准安装特定类别和严重级别的所有补丁。例如，您可以包括归类为 `Security` 的所有补丁，但排除其他补丁（例如 `Bugfix` 或 `Enhancement`）。并且，您可以包括严重性为 `Critical` 的所有补丁，但排除其他补丁（例如 `Important` 和 `Moderate`）。

您还可以通过将补丁 ID 添加到要批准或拒绝的特定补丁列表中，在补丁基准中明确定义补丁，例如 Windows Server 的 KB2736693 或 Amazon Linux 2023 (AL2023) 的 dbus.x86_64:1:1.12.28-1.amzn2023.0.1。您可以选择指定补丁可用后等待修补的特定天数。对于 Linux 和 macOS，您可以选择指定合规性外部补丁列表（安装覆盖列表），而不是补丁基准规则定义的列表。

运行修补操作时，Patch Manager会将当前应用于托管节点的补丁与应根据补丁基准中设置的规则或安装覆盖列表应用的补丁进行比较。您可以选择让 Patch Manager 只显示缺失补丁的报告（Scan 操作），也可以选择让 Patch Manager 自动安装它发现的托管节点中缺失的所有补丁（Scan and install 操作）。

Patch Manager提供了可用于修补操作的预定义补丁基准；但是，这些预定义的配置仅作为示例提供，而不是推荐的最佳实践。我们建议您创建自己的自定义补丁基准，以便更好地控制您的实例集的补丁合规性构成要素。

有关补丁基准的更多信息，请参阅以下主题：

- 预定义和自定义补丁基准
- 已批准补丁和已拒绝补丁列表的程序包名称格式
- 查看 AWS 预定义补丁基准
- 使用自定义补丁基准
- 使用补丁合规性报告

主要组件

在开始使用Patch Manager工具之前，您应该熟悉该工具的修补操作的一些主要组件和功能。

补丁基准

Patch Manager 使用补丁基准，该基准包含用于在补丁发布几天内自动批准补丁的规则以及一系列已批准和已拒绝的补丁。运行修补操作时，Patch Manager 会将当前应用于托管节点的补丁与应根据补丁基准中设置的规则应用的补丁进行比较。您可以选择让 Patch Manager 只显示缺失补丁的报告（Scan 操作），也可以选择让 Patch Manager 自动安装它发现的托管节点中缺失的所有补丁（Scan and install 操作）。

修补操作方法

Patch Manager 目前提供运行 Scan 和 Scan and install 操作的四种方法：

- （推荐）在 Quick Setup 中配置的补丁策略：基于与 AWS Organizations 的集成，单个补丁策略可以定义整个组织的修补计划和补丁基准（包括多个 AWS 账户 和这些账户操作所在的 AWS 区域）。补丁策略也可以仅针对组织中的某些组织单位（OU）。您可以使用单个补丁策略按不同的计划进行扫描安装。有关更多信息，请参阅使用 Quick Setup 补丁策略为组织中的实例配置修补和Quick Setup 中的补丁策略配置。
- 在 Quick Setup 中配置的主机管理选项：与 AWS Organizations 的集成还支持主机管理配置，从而可以对整个组织运行修补操作。但是，此选项仅限于使用当前默认补丁基准扫描缺失的补丁并在合规性报告中提供结果。此操作方法无法安装补丁。有关更多信息，请参阅 使用 Quick Setup 设置 Amazon EC2 主机管理。
- 运行补丁 **`Scan`** 或 **`Install`** 任务的维护时段：可以在 Systems Manager 的 Maintenance Windows 工具中设置维护时段，将其配置为按照您定义的计划运行不同类型的任务。Run Command 类型任务可用于运行 `Scan` 或 `Scan and install` 任务（用于您选择的一组托管节点）。每个维护时段任务只能针对单一 AWS 账户-AWS 区域 对中的托管节点。有关更多信息，请参阅 教程：使用控制台创建修补的维护时段。
- Patch Manager 中的按需立即修补操作 – 使用立即修补选项，可以在需要尽快修补托管式节点时，绕过计划设置。使用 Patch now（立即修补），您可以指定是运行 `Scan` 还是 `Scan and install` 操作，以及要在哪些托管节点上运行该操作。您还可以选择在修补操作期间将 Systems Manager 文档（SSM 文档）作为生命周期挂钩运行。每个 Patch now（立即修补）操作只能针对单一 AWS 账户-AWS 区域 对中的托管节点。有关更多信息，请参阅 按需修补托管式节点。

合规性报告

`Scan` 操作完成后，可以使用 Systems Manager 控制台查看相关信息，了解哪些托管节点不满足补丁合规性以及每个节点缺少哪些补丁。也可以生成 .csv 格式的补丁合规性报告，发送到您选择的 Amazon Simple Storage Service (Amazon S3) 存储桶。您可以生成一次性报告，也可以生成定期报告。对于单个托管式节点，报告包含节点所有补丁的详细信息。对于所有托管式节点的报告，仅提供缺少多少补丁的摘要。生成报告后，您可以使用 Amazon QuickSight 等工具导入和分析数据。有关更多信息，请参阅 使用补丁合规性报告。

> **Note**
>
> 通过使用补丁策略生成的合规性项的执行类型为 `PatchPolicy`。补丁策略操作中未生成的合规项的执行类型为 `Command`。

集成

Patch Manager 与以下其他 AWS 服务 服务集成在一起：

- AWS Identity and Access Management（IAM）：使用 IAM 控制哪些用户、群组和角色有权访问 Patch Manager 操作。有关更多信息，请参阅 AWS Systems Manager 如何与 IAM 协同工作 和配置 Systems Manager 所需的实例权限。
- AWS CloudTrail：使用 CloudTrail 记录由用户、角色或群组发起的修补操作事件历史记录以供审核。有关更多信息，请参阅 使用 AWS CloudTrail 记录 AWS Systems Manager API 调用。
- AWS Security Hub：可将来自 Patch Manager 的补丁合规性数据发送到 AWS Security Hub。Security Hub 能让您全面了解高优先级安全警报和合规性状态。它还监控您的机群的修补状态。有关更多信息，请参阅 将 Patch Manager 与 AWS Security Hub 集成。
- AWS Config：在 AWS Config 中设置录制以在 Patch Manager 控制面板中查看 Amazon EC2 实例管理数据。有关更多信息，请参阅 查看补丁程序控制面板摘要。

主题

- Quick Setup 中的补丁策略配置
- Patch Manager 先决条件
- Patch Manager 操作是如何工作的
- 适用于修补托管式节点的 SSM 命令文档
- 补丁基准
- 在 Amazon Linux 2 托管式节点上使用 Kernel Live Patching
- 使用控制台处理 Patch Manager 资源和合规性
- 使用 AWS CLI 处理 Patch Manager 资源使用
- AWS Systems Manager Patch Manager 教程
- 排查 Patch Manager 问题

Quick Setup 中的补丁策略配置

AWS 建议使用补丁策略为组织和 AWS 账户配置补丁。补丁政策于 2022 年 12 月在Patch Manager中推出。

补丁策略是您使用Quick Setup（AWS Systems Manager 中的一项工具）设置的一项配置。与以前配置修补的方法相比，补丁策略可以对修补操作进行更广泛、更集中的控制。补丁策略可用于 Patch Manager 支持的所有操作系统，包括支持的 Linux、macOS和 Windows Server 版本。有关创建补丁策略的说明，请参阅 使用 Quick Setup 补丁策略为组织中的实例配置修补。

补丁策略的主要功能

与其使用其他方法修补您的节点，不如使用补丁策略来利用以下主要功能：

- 单一设置：若使用维护时段或 State Manager 关联设置修补操作，可能需要在 Systems Manager 控制台的不同部分执行多项任务。使用补丁策略，可以在单个向导中设置所有修补操作。
- 多账户/多区域支持：使用 Patch Manager 中的维护时段、State Manager关联或 Patch now（立即修补）功能，您只能针对一个 AWS 账户 与 AWS 区域 对中的托管节点。如果您使用多个账户和多个区域，则设置和维护任务可能需要大量时间，因为您必须在每个账户—区域对中执行设置任务。但是，如果您使用 AWS Organizations，则可以在您的所有 AWS 账户 的所有 AWS 区域 设置一个应用于所有托管节点的补丁策略。或者，您也可以选择把补丁策略只应用于您选择的账户和区域中的某些组织单位 (OU)。还可以选择把补丁策略应用于单个本地账户。
- 组织层面的安装支持：Quick Setup 中的现有主机管理配置选项支持对托管节点进行每日扫描，以确定是否满足补丁合规性。但是，要在预定时间完成此扫描，且只生成补丁合规信息。不执行补丁安装。使用补丁策略，您可以指定不同的扫描和安装计划。您还可使用自定义的 CRON 或 Rate 表达式来选择这些操作的频率和时间。例如，您可以每天扫描缺失的补丁，以便为您提供定期更新的合规信息。但是，为了避免不必要的停机，您的安装计划可能仅为每周一次。
- 简化的补丁基准选择：补丁策略仍包含补丁基准，补丁基准的配置方式不变。但是，在创建或更新补丁策略时，可以在单个列表中为每种操作系统（OS）类型选择要使用的 AWS 托管或自定义基准。无需在单独的任务中为每种操作系统类型指定默认基准。

> **Note**
>
> 在运行基于补丁策略的修补操作时，他们使用的是 `AWS-RunPatchBaseline` SSM 文档。有关更多信息，请参阅 用于修补的 SSM 命令文档：AWS-RunPatchBaseline。

相关信息

使用 Systems Manager Quick Setup 在整个 AWS 组织中集中部署修补操作（AWS 云运营和迁移博客）

与补丁策略的其他区别

相比以前配置修补的方法，使用补丁策略时需要注意一些其他差异：

- 无需补丁组：在以前的修补操作中，您可以将多个节点标记为属于一个补丁组，然后指定用于该补丁组的补丁基准。如果没有定义补丁组，Patch Manager 使用操作系统类型当前默认的补丁基准修补实例。使用补丁策略则不再需要设置和维护补丁组。

  > **Note**
  >
  > 对于在 2022 年 12 月 22 日发布补丁策略支持之前尚未使用补丁组的账户-区域对，控制台不支持补丁组功能。补丁组功能在此日期之前开始使用补丁组的账户区域对中仍然可用。

- 已删除“配置修补”页面：在补丁策略发布之前，您可以在 Configure patching（配置修补）页面上指定要修补哪些节点的默认值、修补计划和修补操作。此页面已从 Patch Manager 中删除。现在已在补丁策略中指定这些选项。
- 不支持“立即修补”：按需修补节点的能力仍然仅限于每次一个 AWS 账户 - AWS 区域 对。有关信息，请参阅按需修补托管式节点。
- 补丁策略和合规信息：在根据修补策略配置扫描托管节点是否合规时，系统将为您提供合规数据。您可以像使用其他合规性扫描方法一样查看并处理数据。尽管您可以为整个组织或多个组织单位设置补丁策略，但会单独报告每个 AWS 账户 - AWS 区域 对的合规信息。有关更多信息，请参阅 使用补丁合规性报告。
- 关联合规性状态和补丁策略 – Quick Setup 补丁策略下的托管式节点的修补状态与该节点的 State Manager 关联执行状态相匹配。如果关联执行状态为 `Compliant`，则还会将托管式节点的修补状态标记为 `Compliant`。如果关联执行状态为 `Non-Compliant`，则还会将托管式节点的修补状态标记为 `Non-Compliant`。

补丁策略支持的 AWS 区域

以下区域当前支持 Quick Setup 中的补丁策略配置：

- 美国东部（俄亥俄州）(us-east-2)
- 美国东部（弗吉尼亚州北部）(us-east-1)
- 美国西部（北加利福尼亚）(us-west-1)
- 美国西部（俄勒冈州）(us-west-2)
- 亚太地区（孟买）(ap-south-1)
- 亚太地区（首尔）(ap-northeast-2)
- 亚太地区（新加坡）(ap-southeast-1)
- 亚太地区（悉尼）(ap-southeast-2)

- 亚太地区（东京）(ap-northeast-1)
- 加拿大（中部）(ca-central-1)
- 欧洲地区（法兰克福）(eu-central-1)
- 欧洲地区（爱尔兰）(eu-west-1)
- 欧洲（伦敦）(eu-west-2)
- 欧洲地区（巴黎）（eu-west-3）
- 欧洲地区（斯德哥尔摩）(eu-north-1)
- 南美洲（圣保罗）（sa-east-1）

Patch Manager 先决条件

使用Patch Manager（AWS Systems Manager 中的一项工具）之前，请确保自己已满足必要的先决条件。

主题

- SSM Agent 版本
- Python 版本
- 与补丁源的连接
- S3 终端节点访问
- 在本地安装补丁的权限
- Patch Manager 支持的操作系统

SSM Agent 版本

SSM Agent 版本 2.0.834.0 或更高版本在要用 Patch Manager 管理的托管式节点上运行。

> **Note**
>
> 如果有新工具添加至 Systems Manager 或者对现有工具进行了更新，则将发布 SSM Agent 的更新版本。无法使用代理的最新版本可能会阻止托管式节点使用 Systems Manager 的各项工具和功能。因此，我们建议您自动完成确保机器上的 SSM Agent 为最新的过程。有关更多信息，请参阅 自动更新到 SSM Agent。要获得有关 SSM Agent 更新的通知，请在 GitHub 上订阅 SSM Agent 发布说明页面。

Python 版本

对于 macOS 以及大多数 Linux 操作系统（OS），Patch Manager 目前支持 Python 版本 2.6 - 3.10。AlmaLinux、Debian Server、Raspberry Pi OS 和 Ubuntu Server 操作系统需要使用受支持版本的 Python 3（3.0 - 3.10）。

与补丁源的连接

如果您的托管式节点没有直接连接到互联网，并且您正在通过某个 VPC 端点使用 Amazon Virtual Private Cloud（Amazon VPC），则必须确保这些节点能够访问源补丁存储库（存储库）。在 Linux 节点上，通常是从节点上配置的远程存储库下载补丁更新。因此，节点必须能够连接到远程存储库，才能执行修补。有关更多信息，请参阅 如何选择安全性补丁。

CentOS 和 CentOS Stream：启用 **`EnableNonSecurity`** 标志

CentOS 6 和 7 托管式节点将 Yum 用作软件包管理器。CentOS 8 和 CentOS Stream 节点将 DNF 用作软件包管理器。两个程序包管理器都使用更新通知概念。更新通知只是修复特定问题的软件包的集合。

但是，CentOS 和 CentOS Stream 默认存储库不配置更新通知。这意味着，Patch Manager 不检测默认 CentOS 和 CentOS Stream 存储库上的软件包。要允许 Patch Manager 处理更新通知中未包含的软件包，您必须启用补丁基准规则中的 `EnableNonSecurity` 标记。

Windows Server：确保连接到 Windows 更新目录或 Windows 服务器更新服务（WSUS）

Windows Server 托管式节点必须能够连接到 Windows 更新目录或 Windows Server Update Service (WSUS)。确认您的节点通过互联网网关、NAT 网关或 NAT 实例已经连接到 Microsoft 更新目录。如果您使用的是 WSUS，请确认该节点已连接到环境中的 WSUS 服务器。有关更多信息，请参阅 问题：托管式节点无法访问 Windows 更新目录或 WSUS。

S3 终端节点访问

无论托管式节点是在私有网络还是公有网络中运行，如果无法访问所需的 AWS 托管式 Amazon Simple Storage Service (Amazon S3) 存储桶，则修补操作会失败。有关托管式节点必须能够访问的 S3 存储桶的信息，请参阅 SSM Agent 与 AWS 托管 S3 存储桶进行通信 和 使用适用于 Systems Manager 的 VPC 端点提高 EC2 实例的安全性。

在本地安装补丁的权限

在 Windows Server 和 Linux 操作系统上，Patch Manager分别假定使用管理员和根用户账户来安装补丁。

然而，在 macOS 上，对于 Brew 和 Brew Cask，Homebrew 不支持在根用户账户下运行其命令。因此，Patch Manager 以 Homebrew 目录的所有者或属于 Homebrew 目录的所有者组的有效用户身份查询并运行 Homebrew 命令。因此，为了安装补丁，homebrew 目录的所有者还需要 /usr/local 目录的递归所有者权限。

> **Tip**
>
> 以下命令可为指定的用户提供此权限：
>
> ```
> sudo chown -R $USER:admin /usr/local
> ```

Patch Manager 支持的操作系统

Patch Manager工具并不一定支持其他 Systems Manager 工具所支持的所有相同的操作系统版本。例如，Patch Manager 不支持 CentOS 6.3 或 Raspberry Pi OS 8 (Jessie)。（有关 Systems Manager 所支持的操作系统的完整列表，请参阅 Systems Manager 支持的操作系统。）因此，请确保要使用 Patch Manager 的托管式节点在下表所列操作系统之一中运行。

> **Note**
>
> Patch Manager 依靠在托管式节点上配置的补丁存储库（例如 Windows 的 Windows 更新目录和 Windows Server Update Services）来检索要安装的可用补丁。因此，对于生命周期终止（EOL）操作系统版本，如果没有新的更新可用，则 Patch Manager 可能无法报告新的更新。这可能是因为 Linux 发行版维护者、Microsoft 或 Apple 没有发布任何新更新，或者因为托管式节点没有访问新更新的适当许可。
>
> Patch Manager 报告托管式节点上可用补丁的合规性状态。因此，如果实例运行的是 EOL 操作系统，但没有可用的更新，则 Patch Manager 可能会将该节点报告为“合规”，具体取决于为修补操作配置的补丁基准。

操作系统	详细信息
Linux	• AlmaLinux 8.x、9.x • Amazon Linux 2012.03-2018.03 • Amazon Linux 2 版本 2.0 和所有更高版本 • Amazon Linux 2022

操作系统	详细信息
	• Amazon Linux 2023 • CentOS 6.5–7.9、8.x • CentOS Stream 8、9 • Debian Server 8.x、9.x、10.x、11.x 和 12.x • Oracle Linux 7.5–8.x、9.x • Raspberry Pi OS（原 Raspbian）9（Stretch） • Red Hat Enterprise Linux (RHEL) 6.5–8.x、9.x、10.x • Rocky Linux 8.x、9.x • SUSE Linux Enterprise Server (SLES) 12.0 及更高的 12.x 版本；15.x • Ubuntu Server 14.04 LTS、16.04 LTS、18.04 LTS、20.04 LTS、20.10 STR、22.04 LTS、23.04、23.10、24.04、24.10 和 25.04

操作系统	详细信息
macOS	macOS 仅支持 Amazon EC2 实例。 11.3.1；11.4–11.7（Big Sur） 12.0–12.7（Monterey） 13.0–13.7（Ventura） 14.x (Sonoma) 15.x (Sequoia) macOS 操作系统更新 Patch Manager 不支持 macOS 的操作系统 (OS) 更新或升级，例如从 12.x 到 13.x 或 13.1 到 13.2。要在 macOS 上执行操作系统版本更新，我们建议使用 Apple 的内置操作系统升级机制。有关更多信息，请参阅 Apple Developer 文档网站上的 Device Management。 Homebrew 支持 Patch Manager 要求将开源软件包管理系统 Homebrew 安装在以下任一默认安装位置： • `/usr/local/bin` • `/opt/homebrew/bin` 如果未安装 Homebrew，则使用 Patch Manager 执行的修补操作将无法正常运行。 Homebrew 已停止支持 macOS 10.14.x（Mojave）和 10.15.x（Catalina）。因此，目前不支持使用 Patch Manager 对这些版本执行修补操作。 区域支持

操作系统	详细信息
	并非所有 AWS 区域 都支持 macOS。有关对适用于 macOS 的 Amazon EC2 支持的一般信息，请参阅《Amazon EC2 用户指南》中的 Amazon EC2 Mac 实例。 macOS 边缘设备 macOS 不支持适用于 AWS IoT Greengrass 核心设备的 SSM Agent。您不能使用 Patch Manager 修补 macOS 边缘设备。

操作系统	详细信息
Windows	Windows Server 2008 至 Windows Server 2025，包括 R2 版本。 **Note** Windows 10 不支持适用于 AWS IoT Greengrass 核心设备的 SSM Agent。您不能使用 Patch Manager 修补 Windows 10 边缘设备。 Windows Server 2008 支持 从 2020 年 1 月 14 日开始，Microsoft 不再为 Windows Server 2008 的功能或安全性更新提供支持。原有 Amazon Machine Images (AMIs) for Windows Server 2008 和 2008 R2 仍包含预安装的 SSM Agent 2 版，但 Systems Manager 不再正式支持 2008 版，并且不再针对更新这些 Windows Server 版本更新代理。此外，SSM Agent 版本 3 可能不兼容 Windows Server 2008 和 2008 R2 上的所有操作。适用于 Windows Server 2008 版的最后正式支持的 SSM Agent 版本为 2.3.1644.0。 Windows Server 2012 和 2012 R2 支持 Windows Server 2012 和 2012 R2 支持已于 2023 年 10 月 10 日结束。要将 Patch Manager 与这些版本一起使用，还建议使用 Microsoft 的扩展安全更新（ESU）。有关更多信息，请参阅 Microsoft 网站上的 Windows Server 2012 和 2012 R2 即将终止支持。

Patch Manager 操作是如何工作的

本节介绍一些技术细节，说明Patch Manager（AWS Systems Manager 中的一项工具）如何确定安装哪些补丁，以及如何在每个受支持的操作系统上安装这些补丁。对于 Linux 操作系统，其还提供有关在自定义补丁基准中为补丁指定源存储库（而非托管式节点上配置的默认源存储库）的信息。此节还提供了有关补丁基准规则在不同的 Linux 操作系统分发版上的工作原理的详细信息。

> **Note**
>
> 无论您使用哪种配置方法或类型进行修补操作，以下主题中的信息都适用：
>
> - Quick Setup 中配置的补丁策略
> - Quick Setup 中配置的主机管理选项
> - 运行补丁 `Scan` 或 `Install` 任务的维护时段
> - 按需 Patch now（立即修补）操作

主题

- 如何计算软件包发布日期和更新日期
- 如何选择安全性补丁
- 如何指定备用补丁源存储库 (Linux)
- 如何安装补丁
- 补丁基准规则在基于 Linux 的系统上的工作原理
- Linux 和 Windows Server 之间的补丁操作差异

如何计算软件包发布日期和更新日期

> **Important**
>
> 本页面上的信息适用于 Amazon Elastic Compute Cloud（Amazon EC2）实例的 Amazon Linux 1、Amazon Linux 2、Amazon Linux 2022 和 Amazon Linux 2023 操作系统（OS）。这些 OS 类型的软件包由 Amazon Web Services 创建和维护。其他操作系统的软件包和存储库的发布日期和更新日期的计算方式受其制造商管理方式的影响。对于除 Amazon Linux、Amazon Linux 2、Amazon Linux 2022 和 Amazon Linux 2023 之外的操作系统，如

Red Hat Enterprise Linux（RHEL）和 SUSE Linux Enterprise Server（SLES），请参阅制造商的文档，以了解有关如何更新和维护其程序包的信息。

在您创建的自定义补丁基准设置中，对于大多数操作系统类型，您可以指定补丁在特定天数后自动批准安装。AWS 提供了几个预定义的补丁基准，其中包括 7 天的自动批准日期。

自动批准延迟是发布补丁后到自动批准补丁用于修补前等待的天数。例如，可以使用 `CriticalUpdates` 分类创建规则，并将其配置为自动批准延迟为 7 天。此时，发布日期或最后更新日期为 7 月 7 日的新关键补丁将在 7 月 14 日自动获得批准。

为避免在 Amazon Linux 1、Amazon Linux 2、Amazon Linux 2022 和 Amazon Linux 2023 上出现自动批准延迟的意外结果，了解其发布日期和更新日期的计算方式至关重要。

在大多数情况下，安装补丁之前的自动批准等待时间根据 `updateinfo.xml` 中的 `Updated Date` 值，而不是 `Release Date` 值计算。以下是有关这些日期计算的重要细节：

- `Release Date` 是指通知发布的日期。但这并不意味着该软件包在相关存储库中可用。
- `Update Date` 是指通知最后更新的日期。通知的更新可以指文本更新或描述更新这样的小更新。但这并不意味着该软件包在该日期发布，也不意味着在相关存储库中可用。

  这意味着如果软件包的 `Update Date` 值为 7 月 7 日，可能直到（例如）7 月 13 日才能安装。假设在这种情况下，如果补丁基准指定的自动批准延迟为 7 天，则会在 7 月 14 日的 `Install` 操作中运行。因为 `Update Date` 值是运行日期前 7 天，所以软件包中的补丁和更新将在 7 月 14 日安装。即使自软件包可用于实际安装以来仅过了 1 天，安装仍会进行。
- 包含操作系统或应用程序补丁的软件包在首次发布后可以更新多次。
- 软件包可以发布到 AWS 托管存储库中，但如果以后发现它有问题，则可以回滚。

在某些补丁操作中，这些因素可能并不重要。例如，如果补丁基准配置为安装严重性的值为 `Low` 和 `Medium`，且分类为 `Recommended` 的补丁，则任何自动批准延迟都可能几乎不会对您的操作产生影响。

但是，如果关键或高严重性补丁的安装时间较为重要，您可能需要对补丁的安装时间进行更多的控制。执行此操作的建议方法是使用替代补丁源存储库，而不是默认存储库来执行托管节点上的补丁操作。

您可以在创建自定义补丁基准时指定备用补丁源存储库。在每个自定义补丁基准中，您可以为多达 20 个版本的受支持的 Linux 操作系统指定补丁源配置。有关更多信息，请参阅 如何指定备用补丁源存储库 (Linux)。

如何选择安全性补丁

Patch Manager（AWS Systems Manager 中的一项工具）的主要设计意图在于在托管式节点上安装与安全性相关的操作系统更新。默认情况下，Patch Manager 并非安装所有可用的补丁，而只安装一小部分旨在提高安全性的补丁。

默认情况下，除非安装其他软件包更新需要替换，Patch Manager 不会将软件包存储库中标记为已过时软件包替换为任何具有不同名称的替换软件包。而对于更新软件包的命令，Patch Manager 仅会报告已安装但已过时的软件包并为其安装缺失的更新。这是因为替换已过时软件包通常需要卸载现有软件包并安装其替换软件包。替换已过时软件包可能会引入不在计划内的重大更改或额外功能。

这种行为与 YUM 和 DNF 的 `update-minimal` 命令一致，后者也侧重于安全更新而不是功能升级。有关更多信息，请参阅 the section called “如何安装补丁”。

对于报告修补程序严重性级别的基于 Linux 的操作系统类型，Patch Manager 将软件发布商报告的严重性级别用于更新通知或单个修补程序。Patch Manager 不会从第三方来源（例如常见漏洞评分系统 (CVSS)），或者国家漏洞数据库 (NVD) 发布的指标中获取严重性级别。

> **Note**
>
> 在 Patch Manager 支持的所有基于 Linux 的系统中，您可以选择为托管式节点配置的不同源存储库，以便通常用于安装非安全性更新。有关信息，请参阅如何指定备用补丁源存储库 (Linux)。

本节的剩余内容解释了 Patch Manager 如何为其他受支持的操作系统选择安全补丁。

Amazon Linux 1, Amazon Linux 2, Amazon Linux 2022, and Amazon Linux 2023

在 Amazon Linux 1 和 Amazon Linux 2 上处理预配置存储库的方式与 Amazon Linux 2022 和 Amazon Linux 2023 上不同。

在 Amazon Linux 1 和 Amazon Linux 2 上，Systems Manager 补丁基准服务使用托管式节点上的预配置存储库。节点上通常有两个预配置存储库（存储库）：

在 Amazon Linux 1 上

- 存储库 ID：`amzn-main/latest`

  存储库名称：`amzn-main-Base`
- 存储库 ID：`amzn-updates/latest`

存储库名称：amzn-updates-Base

在 Amazon Linux 2 上

- 存储库 ID：amzn2-core/2/*architecture*

  存储库名称：Amazon Linux 2 core repository
- 存储库 ID：amzn2extra-docker/2/*architecture*

  存储库名称：Amazon Extras repo for docker

> **Note**
>
> *##*可以是 x86_64 或 aarch64。

当您创建 Amazon Linux 2023（AL2023）实例时，它包含 AL2023 版本和您选择的特定 AMI 中可用的更新。AL2023 实例在启动时不会自动接收其他重大和重要的安全更新。相反，通过 AL2023 支持的默认开启的版本化存储库功能进行确定性升级，您可以根据满足您特定需求的计划应用更新。有关更多信息，请参阅《Amazon Linux 2023 User Guide》中的 Deterministic upgrades through versioned repositories。

在 Amazon Linux 2022 上，预配置的存储库与程序包更新的锁定版本相关。Amazon Linux 2022 的新 Amazon Machine Images（AMIs）在发布后会锁定到特定版本。对于补丁更新，Patch Manager 会检索补丁更新存储库的最新锁定版本，然后根据该锁定版本的内容更新托管式节点上的程序包。

在 AL2023 上，预配置的存储库如下所示：

- 存储库 ID：amazonlinux

  存储库名称：Amazon Linux 2023 存储库

在 Amazon Linux 2022（预览版）上，预配置的存储库与程序包更新的锁定版本相关。Amazon Linux 2022 的新 Amazon Machine Images（AMIs）在发布后会锁定到特定版本。对于补丁更新，Patch Manager 会检索补丁更新存储库的最新锁定版本，然后根据该锁定版本的内容更新托管式节点上的程序包。

在 Amazon Linux 2022 上，预配置的存储库如下所示：

- 存储库 ID：`amazonlinux`

  存储库名称：Amazon Linux 2022 存储库

> **Note**
>
> 所有更新都是从托管式节点上配置的远程存储库下载。因此，节点必须具有对 Internet 的出站访问权限才能连接到存储库，以便执行修补。

Amazon Linux 1 和 Amazon Linux 2 托管式节点使用 Yum 作为程序包管理器。Amazon Linux 2022 和 Amazon Linux 2023 使用 DNF 作为程序包管理器。

两个程序包管理器都将更新通知概念作为名为 `updateinfo.xml` 的文件使用。更新通知只是修复特定问题的软件包的集合。Patch Manager 将更新通知中的所有软件包视为安全性软件包。单个软件包未分配分类或严重性级别。因此，Patch Manager 会向相关软件包分配更新通知属性。

> **Note**
>
> 如果在创建补丁基准页面中选中了包括非安全性更新复选框，则未分配到 `updateinfo.xml` 文件中的软件包（或包含一个没有正确格式的分类、严重性和日期值的文件的软件包）可以包含在补丁的预筛选列表中。但是，为了应用补丁，补丁仍必须符合用户指定的补丁基准规则。
>
> 有关包括非安全性更新选项的更多信息，请参阅如何安装补丁和补丁基准规则在基于 Linux 的系统上的工作原理。

CentOS and CentOS Stream

在 CentOS 和 CentOS Stream 上，Systems Manager 补丁基准服务使用托管式节点上的预配置存储库（储存库）。以下列表提供了虚构 CentOS 8.2 的示例 Amazon Machine Image (AMI)：

- 存储库 ID：`example-centos-8.2-base`

  存储库名称：`Example CentOS-8.2 - Base`

- 存储库 ID：`example-centos-8.2-extras`

存储库名称：`Example CentOS-8.2 - Extras`

- 存储库 ID：`example-centos-8.2-updates`

  存储库名称：`Example CentOS-8.2 - Updates`

- 存储库 ID：`example-centos-8.x-examplerepo`

  存储库名称：`Example CentOS-8.x – Example Repo Packages`

> **Note**
>
> 所有更新都是从托管式节点上配置的远程存储库下载。因此，节点必须具有对 Internet 的出站访问权限才能连接到存储库，以便执行修补。

CentOS 6 和 7 托管式节点将 Yum 用作软件包管理器。CentOS 8 和 CentOS Stream 节点将 DNF 用作软件包管理器。两个程序包管理器都使用更新通知概念。更新通知只是修复特定问题的软件包的集合。

但是，CentOS 和 CentOS Stream 默认存储库不配置更新通知。这意味着，Patch Manager 不检测默认 CentOS 和 CentOS Stream 存储库上的软件包。要允许 Patch Manager 处理更新通知中未包含的软件包，您必须启用补丁基准规则中的 `EnableNonSecurity` 标记。

> **Note**
>
> 支持 CentOS 和 CentOS Stream 更新通知。带有更新通知的存储库发布后即可供下载。

Debian 服务器 and Raspberry Pi OS

在 Debian Server 和 Raspberry Pi OS（原 Raspbian）上，Systems Manager 补丁基准服务使用实例上的预配置存储库（存储库）。这些预配置存储库用于提取可用软件包升级的更新列表。在这一点上，Systems Manager 的作用类似于 `sudo apt-get update` 命令。

然后，在 `debian-security` *`codename`* 存储库中筛选软件包。这就表示，在 Debian Server 的每个版本上，Patch Manager 仅识别属于该版本关联存储库一部分的升级，如下所示：

- Debian Server 8：`debian-security jessie`
- Debian Server 9：`debian-security stretch`

- Debian Server 10: `debian-security buster`
- Debian Server 11: `debian-security bullseye`
- Debian Server 12: `debian-security bookworm`

> **Note**
>
> 仅在 Debian Server 8 上：由于某些 Debian Server 8.* 托管式节点引用的软件包存储库 (`jessie-backports`) 已被淘汰，Patch Manager 会执行其他步骤来确保修补操作成功。有关更多信息，请参阅 如何安装补丁。

Oracle Linux

在 Oracle Linux 上，Systems Manager 补丁基准服务使用托管式节点上的预配置存储库（储存库）。节点上通常有两个预配置存储库。

Oracle Linux 7：

- 存储库 ID：`ol7_UEKR5/x86_64`

  存储库名称：`Latest Unbreakable Enterprise Kernel Release 5 for Oracle Linux 7Server (x86_64)`
- 存储库 ID：`ol7_latest/x86_64`

  存储库名称：`Oracle Linux 7Server Latest (x86_64)`

Oracle Linux 8：

- 存储库 ID：`ol8_baseos_latest`

  存储库名称：`Oracle Linux 8 BaseOS Latest (x86_64)`
- 存储库 ID：`ol8_appstream`

  存储库名称：`Oracle Linux 8 Application Stream (x86_64)`
- 存储库 ID：`ol8_UEKR6`

  存储库名称：`Latest Unbreakable Enterprise Kernel Release 6 for Oracle Linux 8 (x86_64)`

Oracle Linux 9：

- 存储库 ID：`ol9_baseos_latest`

  存储库名称：`Oracle Linux 9 BaseOS Latest (x86_64)`
- 存储库 ID：`ol9_appstream`

  存储库名称：`Oracle Linux 9 Application Stream Packages(x86_64)`
- 存储库 ID：`ol9_UEKR7`

  存储库名称：`Oracle Linux UEK Release 7 (x86_64)`

> **Note**
>
> 所有更新都是从托管式节点上配置的远程存储库下载。因此，节点必须具有对 Internet 的出站访问权限才能连接到存储库，以便执行修补。

Oracle Linux 托管式节点将 Yum 用作软件包管理器，并且 Yum 将更新通知表现为名为 `updateinfo.xml` 的文件。更新通知只是修复特定问题的软件包的集合。单个软件包未分配分类或严重性级别。出于这个原因, Patch Manager 会向相关软件包分配更新通知属性，并根据补丁基准中指定的分类筛选器安装软件包。

> **Note**
>
> 如果在创建补丁基准页面中选中了包括非安全性更新复选框，则未分配到 `updateinfo.xml` 文件中的软件包（或包含一个没有正确格式的分类、严重性和日期值的文件的软件包）可以包含在补丁的预筛选列表中。但是，为了应用补丁，补丁仍必须符合用户指定的补丁基准规则。

AlmaLinux, RHEL, and Rocky Linux

在 AlmaLinux、Red Hat Enterprise Linux 和 Rocky Linux 上，Systems Manager 补丁基准服务使用托管式节点上的预配置存储库（储存库）。节点上通常有三个预配置存储库。

所有更新都是从托管式节点上配置的远程存储库下载。因此，节点必须具有对 Internet 的出站访问权限才能连接到存储库，以便执行修补。

> **Note**
>
> 如果在创建补丁基准页面中选中了包括非安全性更新复选框，则未分配到 `updateinfo.xml` 文件中的软件包（或包含一个没有正确格式的分类、严重性和日期值的文件的软件包）可以包含在补丁的预筛选列表中。但是，为了应用补丁，补丁仍必须符合用户指定的补丁基准规则。

Red Hat Enterprise Linux 7 托管式节点将 Yum 用作程序包管理器。AlmaLinux、Red Hat Enterprise Linux 8 和 Rocky Linux 托管式节点使用 DNF 作为程序包管理器。两个程序包管理器都使用更新通知概念作为名为 `updateinfo.xml` 的文件。更新通知只是修复特定问题的软件包的集合。单个软件包未分配分类或严重性级别。出于这个原因, Patch Manager 会向相关软件包分配更新通知属性，并根据补丁基准中指定的分类筛选器安装软件包。

RHEL 7

> **Note**
>
> 以下存储库 ID 与 RHUI 2 相关联。RHUI 3 于 2019 年 12 月推出，并为 Yum 存储库 ID 引入了不同的命名方案。根据您从中创建托管式节点的 RHEL-7 AMI，您可能需要更新命令。有关更多信息，请参阅 Red Hat 客户门户上的 Repository IDs for RHEL 7 in AWS Have Changed。

- 存储库 ID：`rhui-REGION-client-config-server-7/x86_64`

  存储库名称：`Red Hat Update Infrastructure 2.0 Client Configuration Server 7`
- 存储库 ID：`rhui-REGION-rhel-server-releases/7Server/x86_64`

  存储库名称：`Red Hat Enterprise Linux Server 7 (RPMs)`
- 存储库 ID：`rhui-REGION-rhel-server-rh-common/7Server/x86_64`

  存储库名称：`Red Hat Enterprise Linux Server 7 RH Common (RPMs)`

AlmaLinux、8、RHEL 8 和 Rocky Linux 8

- 存储库 ID：`rhel-8-appstream-rhui-rpms`

存储库名称：Red Hat Enterprise Linux 8 for x86_64 - AppStream from RHUI (RPMs)

- 存储库 ID：rhel-8-baseos-rhui-rpms

  存储库名称：Red Hat Enterprise Linux 8 for x86_64 - BaseOS from RHUI (RPMs)

- 存储库 ID：rhui-client-config-server-8

  存储库名称：Red Hat Update Infrastructure 3 Client Configuration Server 8

AlmaLinux 9、RHEL 9 和 Rocky Linux 9

- 存储库 ID：rhel-9-appstream-rhui-rpms

  存储库名称：Red Hat Enterprise Linux 9 for x86_64 - AppStream from RHUI (RPMs)

- 存储库 ID：rhel-9-baseos-rhui-rpms

  存储库名称：Red Hat Enterprise Linux 9 for x86_64 - BaseOS from RHUI (RPMs)

- 存储库 ID：rhui-client-config-server-9

  存储库名称：Red Hat Enterprise Linux 9 Client Configuration

SLES

在 SUSE Linux Enterprise Server (SLES) 托管式节点上，ZYPP 库从以下位置获取可用补丁的列表（软件包集合）：

- 存储库的列表：etc/zypp/repos.d/*
- 软件包信息：/var/cache/zypp/raw/*

SLES 托管式节点将 Zypper 用作软件包管理器，并且 Zypper 使用补丁的概念。补丁只是修复特定问题的程序包的集合。Patch Manager 处理补丁中所有引用为与安全性相关的软件包。因为没有为单个软件包提供分类或严重性，所以 Patch Manager 为软件包分配它们属于的补丁的属性。

Ubuntu Server

在 Ubuntu Server 上，Systems Manager 补丁基准服务使用托管式节点上的预配置存储库（储存库）。这些预配置存储库用于提取可用软件包升级的更新列表。在这一点上，Systems Manager 的作用类似于 `sudo apt-get update` 命令。

然后，将从 *codename*`-security` 存储库筛选软件包，其中的代号对应唯一的发行版本，例如 `trusty` 适用于 Ubuntu Server 14。Patch Manager 只识别包含在这些储存库的升级：

- Ubuntu Server 14.04 LTS：`trusty-security`
- Ubuntu Server 16.04 LTS：`xenial-security`
- Ubuntu Server 18.04 LTS：`bionic-security`
- Ubuntu Server 20.04 LTS：`focal-security`
- Ubuntu Server 20.10 STR：`groovy-security`
- Ubuntu Server 22.04 LTS：`jammy-security`
- Ubuntu Server 23.04 (`lunar-security`)
- Ubuntu Server 23.10 (`mantic-security`)
- Ubuntu Server 24.04 LTS (`noble-security`)
- Ubuntu Server 24.10 (`oracular-security`)
- Ubuntu Server 25.04 (`plucky-security`)

Windows Server

在 Microsoft Windows 操作系统上，Patch Manager 检索 Microsoft 通过其发布至 Microsoft 更新的以及 Windows Server Update Services (WSUS) 自动获取的可用更新的列表。

> **Note**
>
> Patch Manager 仅为支持 Patch Manager 的 Windows Server 操作系统版本提供可用补丁。例如，Patch Manager 不能用于修补 Windows RT。

Patch Manager 持续监控每个 AWS 区域 中的新更新。每个区域每天至少刷新一次可用更新的列表。在处理来自 Microsoft 的补丁信息时，Patch Manager 会删除已被其补丁列表中的后续更新取代的更新。因此，Patch Manager 只显示最新更新，以供您安装。例如，如果 `KB4012214` 取代了 `KB3135456`，则 `KB4012214` 会显示为 Patch Manager 的更新。

同样，Patch Manager 仅可安装在修补操作期间托管节点上可用的补丁。默认情况下，Windows Server 2019 和 Windows Server 2022 会移除由较新更新所取代的更新。因此，如果您在 Windows Server 补丁基准中使用 `ApproveUntilDate` 参数，但 `ApproveUntilDate` 参数中选择的日期早于最新补丁发布的日期，则会出现以下情况：

- 取代的补丁已从节点中删除，因此无法使用 Patch Manager 进行安装。
- 节点上存在最新的替换补丁，但尚未批准按照 `ApproveUntilDate` 参数的指定日期进行安装。

这意味着托管节点在 Systems Manager 操作方面是合规的，即使可能未安装上个月的关键补丁亦是如此。使用 `ApproveAfterDays` 参数时也可能出现同样的情况。由于 Microsoft 取代的补丁行为，可以设置一个数字（通常大于 30 天），这样，如果 Microsoft 的最新可用补丁是在经过 `ApproveAfterDays` 中的天数之前发布的，则绝不会安装 Windows Server 的补丁。请注意，如果您修改了 Windows 组策略对象（GPO）设置，以使取代的补丁在托管式节点上可用，则此系统行为不适用。

> **Note**
>
> 在某些情况下，Microsoft 会为未指定更新日期和时间的应用程序发布补丁。在这些情况下，系统会默认提供 `01/01/1970` 的更新日期和时间。

如何指定备用补丁源存储库 (Linux)

当您使用托管式节点上配置的默认存储库执行修补操作时，Patch Manager（AWS Systems Manager 中的一项工具）会扫描或安装与安全性相关的补丁。这是 Patch Manager 的默认行为。有关 Patch Manager 如何选择和安装安全性补丁的完整信息，请参阅 如何选择安全性补丁。

不过，在 Linux 系统中，您还可以使用 Patch Manager 安装与安全性无关的补丁，或安装与托管式节点上配置的默认源存储库不同的源存储库中的补丁。您可以在创建自定义补丁基准时指定备用补丁源存储库。在每个自定义补丁基准中，您可以为多达 20 个版本的受支持的 Linux 操作系统指定补丁源配置。

例如，假设您的 Ubuntu Server 机群同时包括 Ubuntu Server 14.04 和 Ubuntu Server 16.04 托管式节点。在这种情况下，您可以在相同的自定义补丁基准中为每个版本指定备用存储库。对于每个版本，您提供名称，指定操作系统版本类型 (产品)，并提供存储库配置。您也可以指定适用于所有版本的受支持的操作系统的单个备用源存储库。

> **Note**
>
> 运行为托管式节点指定备用补丁存储库的自定义补丁基准，不会使这些存储库成为操作系统中的新默认存储库。修补操作完成后，以前配置为节点操作系统的默认存储库的存储库仍为默认存储库。

有关使用此选项的示例场景的列表，请参阅本主题后面的备用补丁源存储库的示例用法。

有关默认和自定义补丁基准的信息，请参阅 预定义和自定义补丁基准。

示例：使用控制台

要在 Systems Manager 控制台中指定备用补丁源存储库，请使用创建补丁基准上的补丁来源部分。有关使用 Patch sources (补丁源) 选项的信息，请参阅创建适用于 Linux 的自定义补丁基准。

示例：使用 AWS CLI

有关将 `--sources` 选项与 AWS Command Line Interface (AWS CLI) 结合使用的示例，请参阅 创建对不同操作系统版本使用自定义存储库的补丁基准。

主题

- 备用存储库的重要注意事项
- 备用补丁源存储库的示例用法

备用存储库的重要注意事项

使用备用补丁存储库计划修补策略时，请注意以下几点：

只指定用于修补的存储库

指定备用存储库并不意味着指定额外 存储库。您可以选择指定除托管式节点上配置为默认存储库以外的存储库。但是，如果需要应用默认存储库更新，还必须在备用补丁源配置中指定默认存储库。

例如，在 Amazon Linux 2 托管式节点上，默认存储库为 `amzn2-core` 和 `amzn2extra-docker`。如果需要在修补操作中包括 Extra Packages for Enterprise Linux (EPEL) 存储库，您必须将所有三个存储库都指定为备用存储库。

> **Note**
>
> 运行为托管式节点指定备用补丁存储库的自定义补丁基准，不会使这些存储库成为操作系统中的新默认存储库。修补操作完成后，以前配置为节点操作系统的默认存储库的存储库仍为默认存储库。

基于 YUM 的分发版本的修补行为取决于 updateinfo.xml 清单

为基于 YUM 的分发版本（如 Amazon Linux 1 或 Amazon Linux 2、Red Hat Enterprise Linux 或 CentOS）指定备用补丁存储库时，修补行为取决于存储库是否包含一个采用正确格式的完整 `updateinfo.xml` 文件形式的更新清单。此文件指定各软件包的发行日期、分类和严重性。以下任一项都会影响修补行为：

- 如果筛选分类和严重性，但并未在 `updateinfo.xml` 中指定它们，筛选器将不会包含此软件包。这也意味着没有 `updateinfo.xml` 文件的软件包不会包含在修补中。
- 如果按 ApprovalAfterDays 筛选，但软件包的发行日期不是 Unix Epoch 格式（或未指定发行日期），筛选器将不会包含此软件包。
- 如果您在创建补丁基准页面中选择了包括非安全性更新复选框，则存在例外。在这种情况下，没有 `updateinfo.xml` 文件（或包含此文件但 Classification（分类）、Severity（严重性）和 Date（日期）值格式不正确）的软件包将包含在补丁的预筛选列表中。（它们仍必须满足其他补丁基准规则要求才能安装。）

备用补丁源存储库的示例用法

示例 1 – Ubuntu Server 的非安全性更新

在使用 AWS 提供的预定义补丁基准 `AWS-UbuntuDefaultPatchBaseline` 的 Ubuntu Server 托管式节点机群上，您已经在使用 Patch Manager 安装安全性补丁。您可以创建基于此默认补丁基准的新补丁基准，但在批准规则中指定您也希望安装属于默认分配的与安全性无关的更新。当对节点运行此补丁基准时，将应用针对安全性和非安全性问题的补丁。您还可以选择在为基准指定的补丁异常中批准非安全性补丁。

示例 2 – Ubuntu Server 的个人软件包存档 (PPA)

Ubuntu Server 托管式节点正在运行通过 Ubuntu 个人软件包归档 (PPA) 分发的软件。在这种情况下，需创建补丁基准，用于指定已在托管式节点上配置为修补操作源存储库的 PPA 存储库。然后使用 Run Command 在节点上运行补丁基准文档。

示例 3：Amazon Linux 上的企业内部应用程序

您需要在 Amazon Linux 托管式节点上运行满足行业监管合规性要求所需的一些应用程序。您可以在节点上为这些应用程序配置存储库，使用 YUM 对这些应用程序进行初始安装，然后更新或创建新的补丁基准以包括此新企业存储库。此后，您可以使用 Run Command 运行 `AWS-RunPatchBaseline` 文档，并通过 `Scan` 选项查看企业软件包是否列在已安装软件包中以及在托管式节点上是否为最新。如果它不是最新的，可以使用 `Install` 选项再次运行该文档来更新应用程序。

如何安装补丁

Patch Manager（AWS Systems Manager 中的一项工具）使用适合操作系统类型的相应内置机制在托管式节点上安装更新。例如，在 Windows Server 上，使用 Windows Update API；在 Amazon Linux 2 上，则使用 `yum` 软件包管理器。

Patch Manager 不会安装新软件包来替换当前安装的已过时软件包。（例外情况：新软件包是正在安装的其他软件包更新的依赖项，或者新软件包与已过时软件包同名。）相反，Patch Manager 会报告已安装的软件包并安装可用的更新。这种方法有助于防止在用一个软件包替换另一个软件包时，可能导致的系统功能意外更改。

如果需要卸载已过时软件包并安装替换软件包，则可能需要使用自定义脚本或软件包管理器命令，而不是使用标准的 Patch Manager 操作。

本部分的剩余内容解释了 Patch Manager 如何在操作系统上安装补丁。

Amazon Linux 1, Amazon Linux 2, Amazon Linux 2022, and Amazon Linux 2023

在 Amazon Linux 1、Amazon Linux 2、Amazon Linux 2022 和 Amazon Linux 2023 托管式节点上，补丁安装工作流如下：

1. 如果使用 `AWS-RunPatchBaseline` 或 `AWS-RunPatchBaselineAssociation` 文档的 `InstallOverrideList` 参数，使用 https URL 或 Amazon Simple Storage Service (Amazon S3) 路径样式 URL 来指定补丁列表，则会安装列出的补丁，并跳过步骤 2-7。
2. 依照补丁基准中的规定应用 GlobalFilters，只保留符合资格的软件包做进一步处理。
3. 依照补丁基准中的规定应用 ApprovalRules。每条批准规则可以将一个软件包定义为已批准。

   但是，批准规则也取决于在创建或上次更新补丁基准时是否选中包括非安全更新复选框。

   如果不包含非安全更新，则应用一条隐式规则，以便只选择在安全存储库中有升级的软件包。对于每个软件包，软件包的候选版本（通常为最新版本）必须包含在安全存储库中。

   如果包含非安全更新，也会考虑来自其他存储库的补丁。

4. 依照补丁基准中的规定应用 ApprovedPatches。已批准的补丁即使被 GlobalFilters 丢弃，也会批准更新；如果 ApprovalRules 中未指定任何批准规则，则给予批准。
5. 依照补丁基准中的规定应用 RejectedPatches。从已批准的补丁列表中删除已拒绝的补丁，并且不会应用已拒绝的补丁。
6. 如果批准了补丁的多个版本，则应用最新版本。
7. YUM 更新 API（Amazon Linux 1、Amazon Linux 2）或 DNF 更新 API（Amazon Linux 2022、Amazon Linux 2023）用于已批准的补丁，如下所示：
   - 对于 AWS 提供的预定义默认补丁基准，仅应用 `updateinfo.xml` 中指定的补丁（仅安全性更新）。这是因为未选中包括非安全性更新复选框。预定义的基准等同于具有以下内容的自定义基准：
     - 未选中包括非安全性更新复选框
     - 严重性列表为 `[Critical, Important]`
     - 分类列表为 `[Security, Bugfix]`

     对于 Amazon Linux 1 和 Amazon Linux 2，此工作流的等效 yum 命令为：

     ```
     sudo yum update-minimal --sec-severity=Critical,Important --bugfix -y
     ```

     对于 Amazon Linux 2022 和 Amazon Linux 2023，此工作流的等效 DNF 命令为：

     ```
     sudo dnf upgrade-minimal --sec-severity=Critical --sec-severity=Important --
     bugfix -y
     ```

     如果选中了包括非安全性更新复选框，则 `updateinfo.xml` 中的补丁和未在 `updateinfo.xml` 中的补丁都会应用（安全更新和非安全性更新）。

     对于 Amazon Linux 1 和 Amazon Linux 2，如果选择的基准选中了包括非安全性更新，并且严重性列表为 `[Critical, Important]`，分类列表为 `[Security, Bugfix]`，则等效的 yum 命令为：

     ```
     sudo yum update --security --sec-severity=Critical,Important --bugfix -y
     ```

     对于 Amazon Linux 2022 和 Amazon Linux 2023，等效的 dnf 命令为：

     ```
     sudo dnf upgrade --security --sec-severity=Critical --sec-severity=Important --
     bugfix -y
     ```

> **Note**
>
> 如果您在 Patch Manager 之外运行这些 `yum` 或 `dnf` 命令，则会使用不同的名称安装新软件包，来替换现已过时软件包。但是，这些软件包无法通过等效的 Patch Manager 操作安装。

Amazon Linux 2022 和 Amazon Linux 2023 的更多修补详细信息

支持严重性级别“无”

Amazon Linux 2022 和 Amazon Linux 2023 还支持补丁严重性级别 `None`，DNF 程序包管理器可识别该严重性级别。

支持严重性级别“中”

对于 Amazon Linux 2022 和 Amazon Linux 2023，`Medium` 补丁严重性级别相当于可能在某些外部存储库中定义的 `Moderate` 严重性。如果您在补丁基准中包含 `Medium` 严重性补丁，则外部补丁中的 `Moderate` 严重性补丁也会安装在实例上。

使用 API 操作 DescribeInstancePatches 查询合规性数据时，筛选严重性级别 `Medium` 会报告严重性级别为 `Medium` 和 `Moderate` 的补丁。

Amazon Linux 2022 和 Amazon Linux 2023 的传递依赖项处理

对于 Amazon Linux 2022 和 Amazon Linux 2023，Patch Manager 可能会安装与等效 `dnf` 命令安装不同的传递依赖项版本。传递依赖项是为了满足其他包的要求而自动安装的包（依赖项的依赖项）。

例如，`dnf upgrade-minimal --security` 安装解决已知安全问题所需的传递依赖项的最低版本，而 Patch Manager 安装相同传递依赖项的最新可用版本。

8. 如果安装了任何更新，则会重启托管式节点。（例外：如果将 `AWS-RunPatchBaseline` 文档中的 `RebootOption` 参数设置为 `NoReboot`，则在 Patch Manager 运行后不会重启托管式节点。有关更多信息，请参阅 参数名称: `RebootOption`。）

CentOS and CentOS Stream

在 CentOS 和 CentOS Stream 托管式节点上，补丁安装工作流如下：

1. 如果使用 AWS-RunPatchBaseline 或 AWS-RunPatchBaselineAssociation 文档的 InstallOverrideList 参数，使用 https URL 或 Amazon Simple Storage Service (Amazon S3) 路径样式 URL 来指定补丁列表，则会安装列出的补丁，并跳过步骤 2-7。

   依照补丁基准中的规定应用 GlobalFilters，只保留符合资格的软件包做进一步处理。

2. 依照补丁基准中的规定应用 ApprovalRules。每条批准规则可以将一个软件包定义为已批准。

   但是，批准规则也取决于在创建或上次更新补丁基准时是否选中包括非安全更新复选框。

   如果不包含非安全更新，则应用一条隐式规则，以便只选择在安全存储库中有升级的软件包。对于每个软件包，软件包的候选版本（通常为最新版本）必须包含在安全存储库中。

   如果包含非安全更新，也会考虑来自其他存储库的补丁。

3. 依照补丁基准中的规定应用 ApprovedPatches。已批准的补丁即使被 GlobalFilters 丢弃，也会批准更新；如果 ApprovalRules 中未指定任何批准规则，则给予批准。

4. 依照补丁基准中的规定应用 RejectedPatches。从已批准的补丁列表中删除已拒绝的补丁，并且不会应用已拒绝的补丁。

5. 如果批准了补丁的多个版本，则应用最新版本。

6. YUM 更新 API（在 CentOS 6.x 和 7.x 版本上）或 DNF 更新（在 CentOS 8 和 CentOS Stream 上）应用于批准的补丁。

   > **Note**
   >
   > 对于 CentOS，Patch Manager 可能会安装与等效 dnf 命令安装不同的传递依赖项版本。传递依赖项是为了满足其他包的要求而自动安装的包（依赖项的依赖项）。例如，dnf upgrade-minimal --security 安装解决已知安全问题所需的传递依赖项的最低版本，而 Patch Manager 安装相同传递依赖项的最新可用版本。

7. 如果安装了任何更新，则会重启托管式节点。（例外：如果将 AWS-RunPatchBaseline 文档中的 RebootOption 参数设置为 NoReboot，则在 Patch Manager 运行后不会重启托管式节点。有关更多信息，请参阅 参数名称: RebootOption。）

Debian 服务器 and Raspberry Pi OS

在 Debian Server 和 Raspberry Pi OS（原 Raspbian）实例上，补丁安装工作流如下：

1. 如果使用 AWS-RunPatchBaseline 或 AWS-RunPatchBaselineAssociation 文档的 InstallOverrideList 参数，使用 https URL 或 Amazon Simple Storage Service (Amazon S3) 路径样式 URL来指定补丁列表，则会安装列出的补丁，并跳过步骤 2-7。
2. 如果某个更新可用于 python3-apt（一个 libapt 的 Python 库接口），则将升级到最新版本。（即使您没有选择包括非安全更新选项，该非安全软件包也会更新。）

   > **Important**
   >
   > 仅在 Debian Server 8 上：由于某些 Debian Server 8.* 托管式节点引用的软件包存储库 (jessie-backports) 已被淘汰，Patch Manager 会执行以下其他步骤来确保修补操作成功：
   >
   > a. 在托管式节点上，将从源位置列表 (jessie-backports) 中注释掉对 /etc/apt/sources.list.d/jessie-backports 存储库的引用。因此，不会尝试从该位置下载补丁。
   >
   > b. 将导入 Stretch 安全更新签名密钥。此密钥为 Debian Server 8.* 发行版上的更新和安装操作提供必要的权限。
   >
   > c. 在这一点运行 apt-get 操作，以确保在修补过程开始之前已安装 python3-apt 的最新版本。
   >
   > d. 安装过程完成后，将恢复对 jessie-backports 存储库的引用，并从 apt 源密钥环中删除签名密钥。这样做是为了使系统配置保持修补操作之前的状态。
   >
   > 下次 Patch Manager 更新系统时，将重复执行同一过程。

3. 依照补丁基准中的规定应用 GlobalFilters，只保留符合资格的软件包做进一步处理。
4. 依照补丁基准中的规定应用 ApprovalRules。每条批准规则可以将一个软件包定义为已批准。

   > **Note**
   >
   > 由于无法可靠地确定 Debian Server 更新程序包的发布日期，因此该操作系统不支持自动审批选项。

   但是，批准规则也取决于在创建或上次更新补丁基准时是否选中包括非安全更新复选框。

   如果不包含非安全更新，则应用一条隐式规则，以便只选择在安全存储库中有升级的软件包。对于每个软件包，软件包的候选版本（通常为最新版本）必须包含在安全存储库中。

   如果包含非安全更新，也会考虑来自其他存储库的补丁。

> **Note**
>
> 对于 Debian Server 和 Raspberry Pi OS，补丁候选版本仅限于 `debian-security` 中包含的补丁。

5. 依照补丁基准中的规定应用 ApprovedPatches。已批准的补丁即使被 GlobalFilters 丢弃，也会批准更新；如果 ApprovalRules 中未指定任何批准规则，则给予批准。
6. 依照补丁基准中的规定应用 RejectedPatches。从已批准的补丁列表中删除已拒绝的补丁，并且不会应用已拒绝的补丁。
7. 使用 APT 库升级软件包。

> **Note**
>
> Patch Manager 不支持使用 APT `Pin-Priority` 选项为软件包分配优先级。Patch Manager 汇总所有已启用存储库的可用更新，并选择匹配每个已安装软件包基准的最新更新。

8. 如果安装了任何更新，则会重启托管式节点。（例外：如果将 `AWS-RunPatchBaseline` 文档中的 `RebootOption` 参数设置为 `NoReboot`，则在 Patch Manager 运行后不会重启托管式节点。有关更多信息，请参阅 参数名称: `RebootOption`。）

macOS

在 macOS 托管式节点上，补丁安装工作流如下：

1. `/Library/Receipts/InstallHistory.plist` 属性列表是软件的记录，已使用 `softwareupdate` 和 `installer` 软件包管理器安装和更新了这些软件。使用 `pkgutil` 命令行工具（适用于 `installer`）和 `softwareupdate` 软件包管理器，运行 CLI 命令来解析此列表。

   对于 `installer`，对 CLI 命令的响应包括 `package name`、`version`、`volume`、`location` 和 `install-time` 详细信息，但只有 `package name` 和 `version` 被 Patch Manager 使用。

   适用于 `softwareupdate`，对 CLI 命令的响应包括软件包名称 (`display name`)、`version` 和 `date`，但 Patch Manager 只使用软件包名称和版本。

对于 Brew 和 Brew 桶，自制软件不支持在根用户下运行的命令。因此, Patch Manager 以 Homebrew 目录的所有者或属于 Homebrew 目录的所有者组的有效用户身份查询并运行 Homebrew 命令。这些命令类似于 `softwareupdate` 和 `installer` 并通过 Python 子进程运行以收集软件包数据，并对输出进行解析以识别软件包名称和版本。

2. 依照补丁基准中的规定应用 GlobalFilters，只保留符合资格的软件包做进一步处理。
3. 依照补丁基准中的规定应用 ApprovalRules。每条批准规则可以将一个软件包定义为已批准。
4. 依照补丁基准中的规定应用 ApprovedPatches。已批准的补丁即使被 GlobalFilters 丢弃，也会批准更新；如果 ApprovalRules 中未指定任何批准规则，则给予批准。
5. 依照补丁基准中的规定应用 RejectedPatches。从已批准的补丁列表中删除已拒绝的补丁，并且不会应用已拒绝的补丁。
6. 如果批准了补丁的多个版本，则应用最新版本。
7. 在托管式节点上调用相应的软件包 CLI 以处理批准的补丁，如下所示：

   > **Note**
   >
   > `installer` 缺乏检查和安装更新的功能。因此，对于 `installer`、Patch Manager 仅报告安装了哪些软件包。因此, `installer` 软件包从未作为 `Missing` 来报告。

   - 对于 AWS 提供的预定义默认补丁基准，以及未选中包括非安全性更新复选框的自定义补丁基准，则将仅应用安全性更新。
   - 对于已选中包括非安全性更新复选框的自定义补丁基准，安全性和非安全性更新都将应用。
8. 如果安装了任何更新，则会重启托管式节点。（例外：如果将 `AWS-RunPatchBaseline` 文档中的 `RebootOption` 参数设置为 `NoReboot`，则在 Patch Manager 运行后不会重启托管式节点。有关更多信息，请参阅 参数名称: `RebootOption`。）

Oracle Linux

在 Oracle Linux 托管式节点上，补丁安装工作流如下：

1. 如果使用 `AWS-RunPatchBaseline` 或 `AWS-RunPatchBaselineAssociation` 文档的 `InstallOverrideList` 参数，使用 https URL 或 Amazon Simple Storage Service (Amazon S3) 路径样式 URL 来指定补丁列表，则会安装列出的补丁，并跳过步骤 2-7。
2. 依照补丁基准中的规定应用 GlobalFilters，只保留符合资格的软件包做进一步处理。

3. 依照补丁基准中的规定应用 ApprovalRules。每条批准规则可以将一个软件包定义为已批准。

   但是，批准规则也取决于在创建或上次更新补丁基准时是否选中包括非安全更新复选框。

   如果不包含非安全更新，则应用一条隐式规则，以便只选择在安全存储库中有升级的软件包。对于每个软件包，软件包的候选版本（通常为最新版本）必须包含在安全存储库中。

   如果包含非安全更新，也会考虑来自其他存储库的补丁。

4. 依照补丁基准中的规定应用 ApprovedPatches。已批准的补丁即使被 GlobalFilters 丢弃，也会批准更新；如果 ApprovalRules 中未指定任何批准规则，则给予批准。
5. 依照补丁基准中的规定应用 RejectedPatches。从已批准的补丁列表中删除已拒绝的补丁，并且不会应用已拒绝的补丁。
6. 如果批准了补丁的多个版本，则应用最新版本。
7. 在版本 7 托管式节点上，对已批准的补丁应用 YUM 更新 API，如下所示：

   - 对于 AWS 提供的预定义默认补丁基准，以及未选中包括非安全性更新复选框的自定义补丁基准，则将仅应用 `updateinfo.xml` 中指定的补丁（仅安全性更新）。

     此工作流的等效 yum 命令为：

     ```
     sudo yum update-minimal --sec-severity=Important,Moderate --bugfix -y
     ```

   - 对于已选中包括非安全性更新复选框的自定义补丁基准，`updateinfo.xml` 中的补丁和未在 `updateinfo.xml` 中的补丁都将应用（安全性和非安全性更新）。

     此工作流的等效 yum 命令为：

     ```
     sudo yum update --security --bugfix -y
     ```

     在版本 8 和 9 托管式节点上，对已批准的补丁应用 DNF 更新 API，如下所示：

     - 对于 AWS 提供的预定义默认补丁基准，以及未选中包括非安全性更新复选框的自定义补丁基准，则将仅应用 `updateinfo.xml` 中指定的补丁（仅安全性更新）。

       此工作流的等效 yum 命令为：

       ```
       sudo dnf upgrade-minimal --security --sec-severity=Moderate --sec-
       severity=Important
       ```

> **Note**
>
> 对于 Oracle Linux，Patch Manager 可能会安装与等效 dnf 命令安装不同的传递依赖项版本。传递依赖项是为了满足其他包的要求而自动安装的包（依赖项的依赖项）。
> 例如，dnf upgrade-minimal --security 安装解决已知安全问题所需的传递依赖项的最低版本，而 Patch Manager 安装相同传递依赖项的最新可用版本。

- 对于已选中包括非安全性更新复选框的自定义补丁基准，updateinfo.xml 中的补丁和未在 updateinfo.xml 中的补丁都将应用（安全性和非安全性更新）。

  此工作流的等效 yum 命令为：

  ```
  sudo dnf upgrade --security --bugfix
  ```

> **Note**
>
> 如果您在 Patch Manager 之外运行这些 yum 或 dnf 命令，则会使用不同的名称安装新软件包，来替换现已过时软件包。但是，这些软件包无法通过等效的 Patch Manager 操作安装。

8. 如果安装了任何更新，则会重启托管式节点。（例外：如果将 AWS-RunPatchBaseline 文档中的 RebootOption 参数设置为 NoReboot，则在 Patch Manager 运行后不会重启托管式节点。有关更多信息，请参阅 参数名称: RebootOption。）

AlmaLinux, RHEL, and Rocky Linux

在 AlmaLinux、Red Hat Enterprise Linux 和 Rocky Linux 托管式节点上，补丁安装工作流如下：

1. 如果使用 AWS-RunPatchBaseline 或 AWS-RunPatchBaselineAssociation 文档的 InstallOverrideList 参数，使用 https URL 或 Amazon Simple Storage Service (Amazon S3) 路径样式 URL 来指定补丁列表，则会安装列出的补丁，并跳过步骤 2-7。
2. 依照补丁基准中的规定应用 GlobalFilters，只保留符合资格的软件包做进一步处理。
3. 依照补丁基准中的规定应用 ApprovalRules。每条批准规则可以将一个软件包定义为已批准。

   但是，批准规则也取决于在创建或上次更新补丁基准时是否选中包括非安全更新复选框。

如果不包含非安全更新，则应用一条隐式规则，以便只选择在安全存储库中有升级的软件包。对于每个软件包，软件包的候选版本（通常为最新版本）必须包含在安全存储库中。

如果包含非安全更新，也会考虑来自其他存储库的补丁。

4. 依照补丁基准中的规定应用 ApprovedPatches。已批准的补丁即使被 GlobalFilters 丢弃，也会批准更新；如果 ApprovalRules 中未指定任何批准规则，则给予批准。
5. 依照补丁基准中的规定应用 RejectedPatches。从已批准的补丁列表中删除已拒绝的补丁，并且不会应用已拒绝的补丁。
6. 如果批准了补丁的多个版本，则应用最新版本。
7. YUM 更新 API（在 RHEL 7 上）或 DNF 更新 API（在 AlmaLinux 8 和 9、RHEL 8、9 和 10 以及 Rocky Linux 8 和 9 上）根据以下规则应用于已批准的补丁：

   场景 1：排除非安全性更新

   - 适用对象：AWS 提供的预定义默认补丁基准和自定义补丁基准。
   - 包括非安全性更新复选框：未选中。
   - 应用的补丁：仅当 `updateinfo.xml` 中指定的补丁（仅安全性更新）与补丁基准配置相匹配且位于所配置的存储库中时，才会应用这些补丁。

     在某些情况下，在所配置的存储库中，`updateinfo.xml` 中指定的补丁可能不再可用。配置的存储库通常只有最新版本的补丁，这是先前所有更新的累积汇总，但是最新版本可能与补丁基准规则不匹配，因此在修补操作中会省略。
   - 命令：对于 RHEL 7，此工作流的等效 Yum 命令为：

     ```
     sudo yum update-minimal --sec-severity=Critical,Important --bugfix -y
     ```

     对于 AlmaLinux、RHEL 8 和 Rocky Linux，此工作流的等效 DNF 命令为：

     ```
     sudo dnf update-minimal --sec-severity=Critical --bugfix -y ; \
     sudo dnf update-minimal --sec-severity=Important --bugfix -y
     ```

> **Note**
>
> 对于 AlmaLinux、RHEL 和 Rocky LinuxRocky Linux，Patch Manager 可能会安装与等效 `dnf` 命令安装不同的传递依赖项版本。传递依赖项是为了满足其他包的要求而自动安装的包（依赖项的依赖项）。
> 例如，`dnf upgrade-minimal --security` 安装解决已知安全问题所需的传递依赖项的最低版本，而 Patch Manager 安装相同传递依赖项的最新可用版本。

场景 2：包括非安全性更新

- 适用对象：自定义补丁基准。
- 包括非安全性更新复选框：选中。
- 应用的补丁：`updateinfo.xml` 中的补丁和未在 `updateinfo.xml` 中的补丁（安全性和非安全性更新）都会应用。
- 命令：对于 RHEL 7，此工作流的等效 Yum 命令为：

```
sudo yum update --security --bugfix -y
```

对于 AlmaLinux 8 和 9、RHEL 8 和 9 以及 Rocky Linux 8 和 9，此工作流的等效 DNF 命令为：

```
sudo dnf update --security --bugfix -y
```

> **Note**
>
> 如果您在 Patch Manager 之外运行这些 `yum` 或 `dnf` 命令，则会使用不同的名称安装新软件包，来替换现已过时软件包。但是，这些软件包无法通过等效的 Patch Manager 操作安装。

8. 如果安装了任何更新，则会重启托管式节点。（例外：如果将 `AWS-RunPatchBaseline` 文档中的 `RebootOption` 参数设置为 `NoReboot`，则在 Patch Manager 运行后不会重启托管式节点。有关更多信息，请参阅 参数名称: `RebootOption`。）

SLES

在 SUSE Linux Enterprise Server (SLES) 托管式节点上，补丁安装工作流如下：

1. 如果使用 AWS-RunPatchBaseline 或 AWS-RunPatchBaselineAssociation 文档的 InstallOverrideList 参数，使用 https URL 或 Amazon Simple Storage Service (Amazon S3) 路径样式 URL 来指定补丁列表，则会安装列出的补丁，并跳过步骤 2-7。
2. 依照补丁基准中的规定应用 GlobalFilters，只保留符合资格的软件包做进一步处理。
3. 依照补丁基准中的规定应用 ApprovalRules。每条批准规则可以将一个软件包定义为已批准。

   但是，批准规则也取决于在创建或上次更新补丁基准时是否选中包括非安全更新复选框。

   如果不包含非安全更新，则应用一条隐式规则，以便只选择在安全存储库中有升级的软件包。对于每个软件包，软件包的候选版本（通常为最新版本）必须包含在安全存储库中。

   如果包含非安全更新，也会考虑来自其他存储库的补丁。
4. 依照补丁基准中的规定应用 ApprovedPatches。已批准的补丁即使被 GlobalFilters 丢弃，也会批准更新；如果 ApprovalRules 中未指定任何批准规则，则给予批准。
5. 依照补丁基准中的规定应用 RejectedPatches。从已批准的补丁列表中删除已拒绝的补丁，并且不会应用已拒绝的补丁。
6. 如果批准了补丁的多个版本，则应用最新版本。
7. 对已批准的补丁应用 Zypper 更新 API。
8. 如果安装了任何更新，则会重启托管式节点。（例外：如果将 AWS-RunPatchBaseline 文档中的 RebootOption 参数设置为 NoReboot，则在 Patch Manager 运行后不会重启托管式节点。有关更多信息，请参阅 参数名称: RebootOption。）

Ubuntu Server

在 Ubuntu Server 托管式节点上，补丁安装工作流如下：

1. 如果使用 AWS-RunPatchBaseline 或 AWS-RunPatchBaselineAssociation 文档的 InstallOverrideList 参数，使用 https URL 或 Amazon Simple Storage Service (Amazon S3) 路径样式 URL来指定补丁列表，则会安装列出的补丁，并跳过步骤 2-7。
2. 如果某个更新可用于 python3-apt（一个 libapt 的 Python 库接口），则将升级到最新版本。（即使您没有选择包括非安全更新选项，该非安全软件包也会更新。）
3. 依照补丁基准中的规定应用 GlobalFilters，只保留符合资格的软件包做进一步处理。
4. 依照补丁基准中的规定应用 ApprovalRules。每条批准规则可以将一个软件包定义为已批准。

> **Note**
>
> 由于无法可靠地确定 Ubuntu Server 的更新程序包的发布日期，因此此操作系统不支持自动批准选项。

但是，批准规则也取决于在创建或上次更新补丁基准时是否选中包括非安全更新复选框。

如果不包含非安全更新，则应用一条隐式规则，以便只选择在安全存储库中有升级的软件包。对于每个软件包，软件包的候选版本（通常为最新版本）必须包含在安全存储库中。

如果包含非安全更新，也会考虑来自其他存储库的补丁。

但是，批准规则也取决于在创建或上次更新补丁基准时是否选中包括非安全更新复选框。

> **Note**
>
> 对于 Ubuntu Server 的每个版本，补丁候选版本仅限于作为该版本关联存储库一部分的补丁，如下所示：
>
> - Ubuntu Server 14.04 LTS：`trusty-security`
> - Ubuntu Server 16.04 LTS：`xenial-security`
> - Ubuntu Server 18.04 LTS：`bionic-security`
> - Ubuntu Server 20.04 LTS：`focal-security`
> - Ubuntu Server 20.10 STR：`groovy-security`
> - Ubuntu Server 22.04 LTS：`jammy-security`
> - Ubuntu Server 23.04 (`lunar-security`)
> - Ubuntu Server 23.10 (`mantic-security`)
> - Ubuntu Server 24.04 LTS (`noble-security`)
> - Ubuntu Server 24.10 (`oracular-security`)
> - Ubuntu Server 25.04 (`plucky-security`)

5. 依照补丁基准中的规定应用 ApprovedPatches。已批准的补丁即使被 GlobalFilters 丢弃，也会批准更新；如果 ApprovalRules 中未指定任何批准规则，则给予批准。
6. 依照补丁基准中的规定应用 RejectedPatches。从已批准的补丁列表中删除已拒绝的补丁，并且不会应用已拒绝的补丁。

7. 使用 APT 库升级软件包。

> **Note**
>
> Patch Manager 不支持使用 APT `Pin-Priority` 选项为软件包分配优先级。Patch Manager 汇总所有已启用存储库的可用更新，并选择匹配每个已安装软件包基准的最新更新。

8. 如果安装了任何更新，则会重启托管式节点。（例外：如果将 `AWS-RunPatchBaseline` 文档中的 `RebootOption` 参数设置为 `NoReboot`，则在 Patch Manager 运行后不会重启托管式节点。有关更多信息，请参阅 参数名称: `RebootOption`。）

Windows Server

在 Windows Server 托管式节点上执行修补操作时，节点从 Systems Manager 请求相应补丁基准快照。此快照包含已批准部署的补丁基准中可用的所有更新的列表。系统会将更新列表发送到 Windows 更新 API，Windows 更新 API 确定哪些更新适用于托管式节点并根据需要安装更新。Windows 仅允许安装最新可用版本的 KB。当 KB 或 KB 的任何先前版本与应用的补丁基准匹配时，Patch Manager 会安装 KB 的最新版本。如果安装了任何更新，则系统会根据完成所有必要修补所需的次数重启托管式节点。（例外：如果将 `AWS-RunPatchBaseline` 文档中的 `RebootOption` 参数设置为 `NoReboot`，则在 Patch Manager 运行后不会重启托管式节点。有关更多信息，请参阅 参数名称: `RebootOption`。）可在 Run Command 请求输出中找到修补操作摘要。其他日志可在托管式节点上的 `%PROGRAMDATA%\Amazon\PatchBaselineOperations\Logs` 文件夹中找到。

因为使用 Windows Update API 下载和安装 KB，所以操作遵循 Windows Update 的所有组策略设置。使用 Patch Manager 时不需要任何组策略设置，但会应用已定义的所有设置，以便（例如）将托管式节点定向到 Windows Server Update Services (WSUS) 服务器。

> **Note**
>
> 默认情况下，Windows 从 Microsoft 的 Windows Update 站点下载所有 KB，这是因为 Patch Manager 使用 Windows Update API 驱动 KB 的下载和安装。因此，托管式节点必须能够访问 Microsoft Windows 更新站点，否则修补将失败。或者，您也可以将 WSUS 服务器配置为 KB 存储库，然后使用组策略将托管式节点配置为将该 WSUS 服务器设为目标。Patch Manager可能会在为 Windows Server 创建自定义补丁基准时引用 KB ID，例如基准配置中包含已批准的补丁列表或已拒绝的补丁列表时。只有在 Microsoft Windows Update

或 WSUS 服务器中分配了 KB ID 的更新才由Patch Manager安装。修补操作中不会包含缺少 KB ID 的更新。
有关创建自定义补丁基准的信息，请参阅以下主题：

- 创建适用于 Windows Server 的自定义补丁基准
- 创建补丁基准（CLI）
- 适用于 Windows 操作系统的软件包名称格式

补丁基准规则在基于 Linux 的系统上的工作原理

对于 Linux 分发版，补丁基准中规则的工作方式因分发版类型的不同而有所差异。与 Windows Server 托管式节点上的补丁更新不同，将在每个节点上评估规则，以便考虑实例上配置的存储库。Patch Manager（AWS Systems Manager 中的一项工具）使用本机软件包管理器推动安装补丁基准批准的补丁。

对于报告修补程序严重性级别的基于 Linux 的操作系统类型，Patch Manager 将软件发布商报告的严重性级别用于更新通知或单个修补程序。Patch Manager 不会从第三方来源（例如常见漏洞评分系统 (CVSS)），或者国家漏洞数据库 (NVD) 发布的指标中获取严重性级别。

主题

- 补丁基准规则在 Amazon Linux 1、Amazon Linux 2、Amazon Linux 2022 和 Amazon Linux 2023 上的工作原理
- 补丁基准规则在 CentOS 和 CentOS Stream 上的工作原理
- 补丁基准规则在 Debian Server 和 Raspberry Pi OS 上的工作原理
- 补丁基准规则在 macOS 上的工作原理
- 补丁基准规则在 Oracle Linux 上的工作原理
- 补丁基准规则在 AlmaLinux、RHEL 和 Rocky Linux 上的工作原理
- 补丁基准规则在 SUSE Linux Enterprise Server 上的工作原理
- 补丁基准规则在 Ubuntu Server 上的工作原理

补丁基准规则在 Amazon Linux 1、Amazon Linux 2、Amazon Linux 2022 和 Amazon Linux 2023 上的工作原理

> **Note**
>
> Amazon Linux 2023（AL2023）使用版本控制的存储库，这些存储库可以通过一个或多个系统设置锁定到特定版本。对于 AL2023 EC2 实例的所有修补操作，Patch Manager 使用最新的存储库版本，与系统配置无关。有关更多信息，请参阅《Amazon Linux 2023 User Guide》中的 Deterministic upgrades through versioned repositories。

在 Amazon Linux 1、Amazon Linux 2、Amazon Linux 2022 和 Amazon Linux 2023 上，补丁选择过程如下：

1. 在托管式节点上，YUM 库（Amazon Linux 1、Amazon Linux 2）或 DNF 库（Amazon Linux 2022 和 Amazon Linux 2023）将访问每个已配置存储库的 `updateinfo.xml` 文件。

   如果未找到 `updateinfo.xml` 文件，是否安装补丁将取决于包括非安全性更新和自动批准设置。例如，如果允许非安全更新，则会在自动批准时间到达时安装这些更新。

2. `updateinfo.xml` 中的每个更新通知都包含几个属性，它们表示通知中的软件包的属性，如下表所述。

   更新通知属性

属性	描述
type	对应于补丁基准的 PatchFilter 数据类型中 Classification 键属性的值。表示更新通知中包含的软件包的类型。 可以使用 AWS CLI 命令 describe-patch-properties 或 API 操作 DescribePatchProperties 来查看受支持值的列表。您也可以在 Systems Manager 控制台中创建补丁基准页面或编辑补丁基准页面中的审批规则区中查看列表。
severity	对应于补丁基准的 PatchFilter 数据类型中 Severity 键属性的值。表示更新通知中包含的

属性	描述
	软件包的严重性。通常只适用于安全性更新通知。 可以使用 AWS CLI 命令 describe-patch-properties 或 API 操作 DescribePatchProperties 来查看受支持值的列表。您也可以在 Systems Manager 控制台中创建补丁基准页面或编辑补丁基准页面中的审批规则区中查看列表。
update_id	表示建议 ID，例如 ALAS-2017-867。补丁基准中的 ApprovedPatches 或 RejectedPatches 属性可以使用建议 ID。
重点	包含有关更新通知的其他信息，例如 CVE ID (格式：CVE-2017-1234567)。补丁基准中的 ApprovedPatches 或 RejectedPatches 属性可以使用 CVE ID。
已更新	对应于补丁基准中的 ApproveAfterDays。表示更新通知中包含的软件包的发行日期（更新的日期）。将当前时间戳与此属性的值比较并配合 `ApproveAfterDays` 可用来确定补丁是否已获得部署批准。

有关已批准的补丁和已拒绝的补丁列表的已接受格式的信息，请参阅 已批准补丁和已拒绝补丁列表的程序包名称格式。

3. 托管式节点的产品由 SSM Agent 确定。此属性对应于补丁基准的 PatchFilter 数据类型中 Product 键属性的值。
4. 根据以下准则为更新选择程序包。

安全性选项	补丁选择
AWS 提供的预定义默认补丁基准和未选中包括非安全性更新复选框的自定义补丁基准	对于 `updateinfo.xml` 中的每个更新通知，补丁基准用作筛选器，只允许更新包含符合条

安全性选项	补丁选择

	件的软件包。如果应用补丁基准定义后有多个软件包适用，则使用最新版本。
	
	对于 Amazon Linux 1 和 Amazon Linux 2，此工作流的等效 yum 命令为：
	
	`sudo yum update-minimal --sec-severity=Critical,Important --bugfix -y`
	
	对于 Amazon Linux 2022 和 Amazon Linux 2023，此工作流的等效 DNF 命令为：
	
	`sudo dnf upgrade-minimal --sec-severity=Critical --sec-severity=Important --bugfix -y`

安全性选项	补丁选择

已选中包括非安全性更新复选框，并且严重性列表为 [Critical, Important] ，分类列表为 [Security, Bugfix] 的自定义补丁基准	除了应用已从 updateinfo.xml 中选择的安全性更新，Patch Manager 还会应用符合补丁筛选规则的非安全性更新。
	
	对于 Amazon Linux 和 Amazon Linux 2，此工作流程的等效 Yum 命令为：
	
	`sudo yum update --security --sec-severity=Critical,Important --bugfix -y`
	
	对于 Amazon Linux 2022 和 Amazon Linux 2023，此工作流程的等效 DNF 命令为：
	
	`sudo dnf upgrade --security --sec-severity=Critical --sec-severity=Important --bugfix -y`
	
	Note
	如果您在 Patch Manager 之外运行这些 yum 或 dnf 命令，则会使用不同的名称安装新软件包，来替换现已过时软件包。但是，这些软件包无法通过等效的 Patch Manager 操作安装。

有关补丁合规性状态值的信息，请参阅 补丁合规性状态值。

补丁基准规则在 CentOS 和 CentOS Stream 上的工作原理

CentOS 和 CentOS Stream 默认存储库不包含 updateinfo.xml 文件。不过，您创建或使用的自定义存储库可能包含该文件。在本主题中，对 updateinfo.xml 的引用仅适用于这些自定义存储库。

在 CentOS 和 CentOS Stream 上，补丁选择过程如下：

1. 在托管式节点上，YUM 库（在 CentOS 6.x 和 7.x 版本上）或 DNF 库（在 CentOS 8.x 和 CentOS Stream 上），可访问每个已配置存储库的 `updateinfo.xml` 文件（如果该文件存在于自定义存储库中）。

   如果未找到 `updateinfo.xml`（其中始终包含默认存储库），是否安装补丁将取决于包括非安全性更新和自动批准设置。例如，如果允许非安全更新，则会在自动批准时间到达时安装这些更新。

2. 如果存在 `updateinfo.xml`，该文件中的每个更新通知都包含几个属性，这些属性表示通知中软件包的属性，如下表所述。

   更新通知属性

属性	描述
type	对应于补丁基准的 PatchFilter 数据类型中 Classification 键属性的值。表示更新通知中包含的软件包的类型。 可以使用 AWS CLI 命令 describe-patch-properties 或 API 操作 DescribePatchProperties 来查看受支持值的列表。您也可以在 Systems Manager 控制台中创建补丁基准页面或编辑补丁基准页面中的审批规则区中查看列表。
severity	对应于补丁基准的 PatchFilter 数据类型中 Severity 键属性的值。表示更新通知中包含的软件包的严重性。通常只适用于安全性更新通知。 可以使用 AWS CLI 命令 describe-patch-properties 或 API 操作 DescribePatchProperties 来查看受支持值的列表。您也可以在 Systems Manager 控制台中创建补丁基准页面或编辑补丁基准页面中的审批规则区中查看列表。
update_id	表示建议 ID，例如 CVE-2019-17055。补丁基准中的 ApprovedPatches 或 RejectedPatches 属性可以使用建议 ID。

属性	描述
重点	包含有关更新通知的其他信息，例如 CVE ID（格式：CVE-2019-17055）或 Bugzilla ID（格式：1463241）。补丁基准中的 ApprovedPatches 或 RejectedPatches 属性可以使用 CVE ID 和 Bugzilla ID。
已更新	对应于补丁基准中的 ApproveAfterDays。表示更新通知中包含的软件包的发行日期（更新的日期）。将当前时间戳与此属性的值比较并配合 `ApproveAfterDays` 可用来确定补丁是否已获得部署批准。

有关已批准的补丁和已拒绝的补丁列表的已接受格式的信息，请参阅 已批准补丁和已拒绝补丁列表的程序包名称格式。

3. 在所有情况下，托管式节点的产品由 SSM Agent 确定。此属性对应于补丁基准的 PatchFilter 数据类型中 Product 键属性的值。
4. 根据以下准则为更新选择程序包。

安全性选项	补丁选择
AWS 提供的预定义默认补丁基准和未选中包括非安全性更新复选框的自定义补丁基准	对于 `updateinfo.xml` 中的每个更新通知（如果该文件存在于自定义存储库中），补丁基准用作筛选条件，只允许更新包含符合条件的软件包。如果应用补丁基准定义后有多个软件包适用，则使用最新版本。 对于其中存在 `updateinfo.xml` 的 CentOS 6 和 7，此工作流的等效 yum 命令为： `sudo yum update-minimal --sec-severity=Critical,Important --bugfix -y`

安全性选项	补丁选择
	对于其中存在 `updateinfo.xml` 的 CentOS 8 和 CentOS Stream，此工作流的等效 dnf 命令为： `sudo dnf upgrade-minimal --sec-severity=Critical --sec-severity=Important --bugfix -y`

安全性选项	补丁选择

已选中包括非安全性更新复选框，并且严重性列表为 [Critical, Important] ，分类列表为 [Security, Bugfix] 的自定义补丁基准	除了应用已从 updateinfo.xml 中选择的安全性更新（如果该文件存在于自定义存储库中），Patch Manager 还会应用符合补丁筛选规则的非安全性更新。
	
	对于其中存在 updateinfo.xml 的 CentOS 6 和 7，此工作流的等效 yum 命令为：
	
	`sudo yum update --sec-severity=Critical,Important --bugfix -y`
	
	对于其中存在 updateinfo.xml 的 CentOS 8 和 CentOS Stream，此工作流的等效 dnf 命令为：
	
	`sudo dnf upgrade --security --sec-severity=Critical --sec-severity=Important --bugfix -y`
	
	对于不含 updateinfo.xml 的默认存储库和自定义存储库，必须选中包括非安全性更新复选框，才能更新操作系统（OS）软件包。
	
	Note
	如果您在 Patch Manager 之外运行这些 yum 或 dnf 命令，则会使用不同的名称安装新软件包，来替换现已过时软件包。但是，这些软件包无法通过等效的 Patch Manager 操作安装。

有关补丁合规性状态值的信息，请参阅 补丁合规性状态值。

补丁基准规则在 Debian Server 和 Raspberry Pi OS 上的工作原理

在 Debian Server 和 Raspberry Pi OS（原 Raspbian）上，补丁基准服务提供对 优先级 和 部分 字段的筛选。这些字段通常存在于所有 Debian Server 和 Raspberry Pi OS 软件包中。为确定补丁基准是否选择了某个补丁，Patch Manager 执行以下操作：

1. 在 Debian Server 和 Raspberry Pi OS 系统上，运行等效于 `sudo apt-get update` 的程序刷新可用软件包列表。不配置存储库，从 `sources` 列表中配置的存储库提取数据。
2. 如果某个更新可用于 `python3-apt`（一个 `libapt` 的 Python 库接口），则将升级到最新版本。（即使您没有选择包括非安全更新选项，该非安全软件包也会更新。）

> **Important**
>
> 仅在 Debian Server 8 上：由于 Debian Server 8.* 操作系统引用的软件包存储库 (`jessie-backports`) 已过时，因此 Patch Manager 会执行以下其他步骤来确保修补操作成功：
>
> a. 在托管式节点上，将从源位置列表 (`jessie-backports`) 中注释掉对 `/etc/apt/sources.list.d/jessie-backports` 存储库的引用。因此，不会尝试从该位置下载补丁。
>
> b. 将导入 Stretch 安全更新签名密钥。此密钥为 Debian Server 8.* 发行版上的更新和安装操作提供必要的权限。
>
> c. 在这一点运行 `apt-get` 操作，以确保在修补过程开始之前已安装 `python3-apt` 的最新版本。
>
> d. 安装过程完成后，将恢复对 `jessie-backports` 存储库的引用，并从 apt 源密钥环中删除签名密钥。这样做是为了使系统配置保持修补操作之前的状态。

3. 接下来将应用 GlobalFilters、ApprovalRules、ApprovedPatches 和 RejectedPatches 列表。

> **Note**
>
> 由于无法可靠地确定 Debian Server 更新程序包的发布日期，因此该操作系统不支持自动审批选项。

但是，批准规则也取决于在创建或上次更新补丁基准时是否选中包括非安全更新复选框。

如果不包含非安全更新，则应用一条隐式规则，以便只选择在安全存储库中有升级的软件包。对于每个软件包，软件包的候选版本（通常为最新版本）必须包含在安全存储库中。在这种情况下，对于 Debian Server，补丁候选版本仅限于以下存储库中包含的补丁：

这些存储库的命名如下：

- Debian Server 8：`debian-security jessie`
- Debian Server 和 Raspberry Pi OS 9：`debian-security stretch`
- Debian Server 10: `debian-security buster`
- Debian Server 11: `debian-security bullseye`
- Debian Server 12: `debian-security bookworm`

如果包含非安全更新，也会考虑来自其他存储库的补丁。

有关已批准的补丁和已拒绝的补丁列表的已接受格式的信息，请参阅 已批准补丁和已拒绝补丁列表的程序包名称格式。

要查看 Priority 和 Section 字段的内容，运行以下 `aptitude` 命令：

> **Note**
>
> 您需要先在 Debian Server 系统上安装 Aptitude。

```
aptitude search -F '%p %P %s %t %V#' '~U'
```

在对此命令的响应中，按以下格式报告所有可升级软件包：

```
name, priority, section, archive, candidate version
```

有关补丁合规性状态值的信息，请参阅 补丁合规性状态值。

补丁基准规则在 macOS 上的工作原理

在 macOS 上，补丁选择过程如下：

1. 在托管式节点上，Patch Manager 访问 `InstallHistory.plist` 文件的已解析内容并标识软件包名称和版本。

有关解析过程的详细信息，请参阅 如何安装补丁 中的 macOS 部分。

2. 托管式节点的产品由 SSM Agent 确定。此属性对应于补丁基准的 PatchFilter 数据类型中 Product 键属性的值。
3. 根据以下准则为更新选择程序包。

安全性选项	补丁选择
AWS 提供的预定义默认补丁基准和未选中包括非安全性更新复选框的自定义补丁基准	对于每个可用软件包更新，补丁基准用作筛选器，只允许更新包含符合条件的软件包。如果应用补丁基准定义后有多个软件包适用，则使用最新版本。
已选中包括非安全性更新复选框的自定义补丁基准	除了应用使用 `InstallHistory.plist` 标识出来的安全性更新，Patch Manager 还会应用符合补丁筛选规则的非安全性更新。

有关补丁合规性状态值的信息，请参阅 补丁合规性状态值。

补丁基准规则在 Oracle Linux 上的工作原理

在 Oracle Linux 上，补丁选择过程如下：

1. 在托管式节点上，YUM 库可访问每个已配置存储库的 `updateinfo.xml` 文件。

> **Note**
>
> 如果存储库不是由 Oracle 管理的，则可能不存在 `updateinfo.xml` 文件。如果未找到 `updateinfo.xml`，是否安装补丁将取决于包括非安全性更新和自动批准设置。例如，如果允许非安全更新，则会在自动批准时间到达时安装这些更新。

2. `updateinfo.xml` 中的每个更新通知都包含几个属性，它们表示通知中的软件包的属性，如下表所述。

更新通知属性

属性	描述
type	对应于补丁基准的 PatchFilter 数据类型中 Classification 键属性的值。表示更新通知中包含的软件包的类型。 可以使用 AWS CLI 命令 describe-patch-properties 或 API 操作 DescribePatchProperties 来查看受支持值的列表。您也可以在 Systems Manager 控制台中创建补丁基准页面或编辑补丁基准页面中的审批规则区中查看列表。
severity	对应于补丁基准的 PatchFilter 数据类型中 Severity 键属性的值。表示更新通知中包含的软件包的严重性。通常只适用于安全性更新通知。 可以使用 AWS CLI 命令 describe-patch-properties 或 API 操作 DescribePatchProperties 来查看受支持值的列表。您也可以在 Systems Manager 控制台中创建补丁基准页面或编辑补丁基准页面中的审批规则区中查看列表。
update_id	表示建议 ID，例如 CVE-2019-17055。补丁基准中的 ApprovedPatches 或 RejectedPatches 属性可以使用建议 ID。
重点	包含有关更新通知的其他信息，例如 CVE ID（格式：CVE-2019-17055）或 Bugzilla ID（格式：1463241）。补丁基准中的 ApprovedPatches 或 RejectedPatches 属性可以使用 CVE ID 和 Bugzilla ID。

属性	描述
已更新	对应于补丁基准中的 ApproveAfterDays。表示更新通知中包含的软件包的发行日期（更新的日期）。将当前时间戳与此属性的值比较并配合 ApproveAfterDays 可用来确定补丁是否已获得部署批准。

有关已批准的补丁和已拒绝的补丁列表的已接受格式的信息，请参阅 已批准补丁和已拒绝补丁列表的程序包名称格式。

3. 托管式节点的产品由 SSM Agent 确定。此属性对应于补丁基准的 PatchFilter 数据类型中 Product 键属性的值。
4. 根据以下准则为更新选择程序包。

安全性选项	补丁选择

AWS 提供的预定义默认补丁基准和未选中包括非安全性更新复选框的自定义补丁基准	对于 updateinfo.xml 中的每个更新通知，补丁基准用作筛选器，只允许更新包含符合条件的软件包。如果应用补丁基准定义后有多个软件包适用，则使用最新版本。
	
	对于版本 7 托管式节点，此工作流的等效 yum 命令为：
	
	`sudo yum update-minimal --sec-severity=Important,Moderate --bugfix -y`
	
	对于版本 8 和 9 托管式节点，此工作流的等效 DNF 命令为：
	
	`sudo dnf upgrade-minimal --security --sec-severity=Moderate --sec-severity=Important`

安全性选项	补丁选择

已选中包括非安全性更新复选框，并且严重性列表为 `[Critical, Important]` ，分类列表为 `[Security, Bugfix]` 的自定义补丁基准	除了应用已从 `updateinfo.xml` 中选择的安全性更新，Patch Manager 还会应用符合补丁筛选规则的非安全性更新。
	
	对于版本 7 托管式节点，此工作流的等效 yum 命令为：
	
	`sudo yum update --security --sec-severity=Critical,Important --bugfix -y`
	
	对于版本 8 和 9 托管式节点，此工作流的等效 DNF 命令为：
	
	`sudo dnf upgrade --security --sec-severity=Critical, --sec-severity=Important --bugfix y`
	
	Note
	如果您在 Patch Manager 之外运行这些 yum 或 dnf 命令，则会使用不同的名称安装新软件包，来替换现已过时软件包。但是，这些软件包无法通过等效的 Patch Manager 操作安装。

有关补丁合规性状态值的信息，请参阅 补丁合规性状态值。

补丁基准规则在 AlmaLinux、RHEL 和 Rocky Linux 上的工作原理

在 AlmaLinux、Red Hat Enterprise Linux（RHEL）和 Rocky Linux 上，补丁选择过程如下：

1. 在托管式节点上，YUM 库 (RHEL 7) 或 DNF 库（AlmaLinux 8 和 9、RHEL 8、9 和 10 以及 Rocky Linux 8 和 9）访问每个已配置存储库的 `updateinfo.xml` 文件。

> **Note**
>
> 如果存储库不是由 Red Hat 管理的，可能不存在 `updateinfo.xml` 文件。如果找不到 `updateinfo.xml`，则不会应用任何补丁。

2. `updateinfo.xml` 中的每个更新通知都包含几个属性，它们表示通知中的软件包的属性，如下表所述。

   更新通知属性

属性	描述
type	对应于补丁基准的 PatchFilter 数据类型中 Classification 键属性的值。表示更新通知中包含的软件包的类型。 可以使用 AWS CLI 命令 describe-patch-properties 或 API 操作 DescribePatchProperties 来查看受支持值的列表。您也可以在 Systems Manager 控制台中创建补丁基准页面或编辑补丁基准页面中的审批规则区中查看列表。
severity	对应于补丁基准的 PatchFilter 数据类型中 Severity 键属性的值。表示更新通知中包含的软件包的严重性。通常只适用于安全性更新通知。 可以使用 AWS CLI 命令 describe-patch-properties 或 API 操作 DescribePatchProperties 来查看受支持值的列表。您也可以在 Systems Manager 控制台中创建补丁基准页面或编辑补丁基准页面中的审批规则区中查看列表。
update_id	表示建议 ID，例如 RHSA-2017:0864。补丁基准中的 ApprovedPatches 或 RejectedPatches 属性可以使用建议 ID。

属性	描述
重点	包含有关更新通知的其他信息，例如 CVE ID (格式：CVE-2017-1000371) 或 Bugzilla ID (格式：1463241)。补丁基准中的 ApprovedPatches 或 RejectedPatches 属性可以使用 CVE ID 和 Bugzilla ID。
已更新	对应于补丁基准中的 ApproveAfterDays。表示更新通知中包含的软件包的发行日期（更新的日期）。将当前时间戳与此属性的值比较并配合 `ApproveAfterDays` 可用来确定补丁是否已获得部署批准。

有关已批准的补丁和已拒绝的补丁列表的已接受格式的信息，请参阅 已批准补丁和已拒绝补丁列表的程序包名称格式。

3. 托管式节点的产品由 SSM Agent 确定。此属性对应于补丁基准的 PatchFilter 数据类型中 Product 键属性的值。
4. 根据以下准则为更新选择程序包。

安全性选项	补丁选择
AWS 提供的预定义默认补丁基准和未在任何规则中选中包括非安全性更新复选框的自定义补丁基准	对于 `updateinfo.xml` 中的每个更新通知，补丁基准用作筛选器，只允许更新包含符合条件的软件包。如果应用补丁基准定义后有多个软件包适用，则使用最新版本。 对于 RHEL 7，此工作流程的等效 Yum 命令为： `sudo yum update-minimal --security=Critical,Important --bugfix -y`

安全性选项	补丁选择
	对于 AlmaLinux 8 和 9、RHEL 8、9 和 10 以及 Rocky Linux 8 和 9，此工作流的等效 DNF 命令为： `sudo dnf upgrade-minimal --sec-severity=Critical --sec-severity=Important --bugfix -y`

已选中包括非安全性更新复选框，并且严重性列表为 `[Critical, Important]` ，分类列表为 `[Security, Bugfix]` 的自定义补丁基准	除了应用已从 `updateinfo.xml` 中选择的安全性更新，Patch Manager 还会应用符合补丁筛选规则的非安全性更新。
	对于 RHEL 7，此工作流程的等效 Yum 命令为：
	`sudo yum update --security --sec-severity=Critical,Important --bugfix -y`
	对于 AlmaLinux 8 和 9、RHEL 8、9 和 10 以及 Rocky Linux 8 和 9，此工作流的等效 DNF 命令为：
	`sudo dnf upgrade --sec-severity=Critical --sec-severity=Important --bugfix -y`
	Note
	如果您在 Patch Manager 之外运行这些 yum 或 dnf 命令，则会使用不同的名称安装新软件包，来替换现已过时软件包。但是，这些软件包无法通过等效的 Patch Manager 操作安装。

有关补丁合规性状态值的信息，请参阅 补丁合规性状态值。

补丁基准规则在 SUSE Linux Enterprise Server 上的工作原理

在 SLES 上，每个补丁包括以下表示补丁中的软件包的属性的属性：

- 类别：对应于补丁基准的 数据类型中的分类PatchFilter键属性的值。表示更新通知中包含的补丁的类型。

  可以使用 AWS CLI 命令 describe-patch-properties 或 API 操作 DescribePatchProperties 来查看受支持值的列表。您也可以在 Systems Manager 控制台中创建补丁基准页面或编辑补丁基准页面中的审批规则区中查看列表。
- 严重性：对应于补丁基准的 PatchFilter 数据类型中的严重性键属性的值。表示补丁的严重性。

  可以使用 AWS CLI 命令 describe-patch-properties 或 API 操作 DescribePatchProperties 来查看受支持值的列表。您也可以在 Systems Manager 控制台中创建补丁基准页面或编辑补丁基准页面中的审批规则区中查看列表。

托管式节点的产品由 SSM Agent 确定。此属性对应于补丁基准的 数据类型中 ProductPatchFilter 键属性的值。

对于每个补丁，补丁基准用作筛选器，只允许更新包含符合条件的软件包。如果应用补丁基准定义后有多个软件包适用，则使用最新版本。

有关已批准的补丁和已拒绝的补丁列表的已接受格式的信息，请参阅 已批准补丁和已拒绝补丁列表的程序包名称格式。

补丁基准规则在 Ubuntu Server 上的工作原理

在 Ubuntu Server上，补丁基准服务提供对 Priority (优先级) 和 Section (部分) 字段的筛选。这些字段通常存在于所有 Ubuntu Server 软件包中。为确定补丁基准是否选择了某个补丁，Patch Manager 执行以下操作：

1. 在 Ubuntu Server 系统上，运行等效于 `sudo apt-get update` 的程序刷新可用软件包列表。不配置存储库，从 `sources` 列表中配置的存储库提取数据。
2. 如果某个更新可用于 `python3-apt`（一个 `libapt` 的 Python 库接口），则将升级到最新版本。（即使您没有选择包括非安全更新选项，该非安全软件包也会更新。）
3. 接下来将应用 GlobalFilters、ApprovalRules、ApprovedPatches 和 RejectedPatches 列表。

> **Note**
>
> 由于无法可靠地确定 Ubuntu Server 的更新程序包的发布日期，因此此操作系统不支持自动批准选项。

但是，批准规则也取决于在创建或上次更新补丁基准时是否选中包括非安全更新复选框。

如果不包含非安全更新，则应用一条隐式规则，以便只选择在安全存储库中有升级的软件包。对于每个软件包，软件包的候选版本（通常为最新版本）必须包含在安全存储库中。在这种情况下，对于 Ubuntu Server，补丁候选版本仅限于以下存储库中包含的补丁：

- Ubuntu Server 14.04 LTS：`trusty-security`
- Ubuntu Server 16.04 LTS：`xenial-security`
- Ubuntu Server 18.04 LTS：`bionic-security`
- Ubuntu Server 20.04 LTS：`focal-security`
- Ubuntu Server 20.10 STR：`groovy-security`
- Ubuntu Server 22.04 LTS：`jammy-security`
- Ubuntu Server 23.04 (`lunar-security`)
- Ubuntu Server 23.10 (`mantic-security`)
- Ubuntu Server 24.04 LTS (`noble-security`)
- Ubuntu Server 24.10 (`oracular-security`)
- Ubuntu Server 25.04 (`plucky-security`)

如果包含非安全更新，也会考虑来自其他存储库的补丁。

有关已批准的补丁和已拒绝的补丁列表的已接受格式的信息，请参阅 已批准补丁和已拒绝补丁列表的程序包名称格式。

要查看 Priority 和 Section 字段的内容，运行以下 `aptitude` 命令：

> **Note**
>
> 您需要先在 Ubuntu Server 16 系统上安装 Aptitude。

```
aptitude search -F '%p %P %s %t %V#' '~U'
```

在对此命令的响应中，按以下格式报告所有可升级软件包：

```
name, priority, section, archive, candidate version
```

有关补丁合规性状态值的信息，请参阅 补丁合规性状态值。

Linux 和 Windows Server 之间的补丁操作差异

本主题介绍Patch Manager（AWS Systems Manager 中的一项工具）中 Linux 和 Windows Server 修补之间的重要差异。

> **Note**
>
> 要修补 Linux 托管式节点，节点必须运行 SSM Agent 2.0.834.0 版或更高版本。
> 如果有新工具添加至 Systems Manager 或者对现有工具进行了更新，则将发布 SSM Agent 的更新版本。无法使用代理的最新版本可能会阻止托管式节点使用 Systems Manager 的各项工具和功能。因此，我们建议您自动完成确保机器上的 SSM Agent 为最新的过程。有关更多信息，请参阅 自动更新到 SSM Agent。要获得有关 SSM Agent 更新的通知，请在 GitHub 上订阅 SSM Agent 发布说明页面。

区别 1：补丁评估

Patch Manager 在 Windows 托管式节点和 Linux 托管式节点上使用不同的进程，以评估应该存在的补丁。

Linux

对于 Linux 修补，Systems Manager 会评估补丁基准规则以及 每个托管式节点上已批准和已被拒绝的补丁列表。Systems Manager 必须在每个节点上评估修补，因为该服务从托管式节点上配置的存储库中检索已知补丁和更新列表。

Windows

对于 Windows 修补，Systems Manager 直接在服务中评估补丁基准规则以及已批准和已拒绝的补丁列表。它可以执行此操作是因为 Windows 补丁从单个存储库 (Windows 更新) 拉取。

区别 2：**Not Applicable** 补丁

因为 Linux 操作系统有大量可用的软件包，所以 Systems Manager 不报告处于不适用状态的补丁的详细信息。例如，A `Not Applicable` 补丁是在实例未安装 Apache 时的 Apache 软件的补丁。Systems Manager 确实会在摘要中报告 `Not Applicable` 补丁的数量，但如果为某个托管式节点调用 DescribeInstancePatches API，则返回的数据中不包含状态为 `Not Applicable` 的补丁。这一行为不同于 Windows。

区别 3：SSM 文档支持

`AWS-ApplyPatchBaseline` Systems Manager 文档（SSM 文档）不支持 Linux 托管式节点。如需对 Linux、macOS 和 Windows Server 托管式节点应用补丁基准，则推荐的 SSM 文档是 `AWS-RunPatchBaseline`。有关更多信息，请参阅适用于修补托管式节点的 SSM 命令文档和用于修补的 SSM 命令文档：`AWS-RunPatchBaseline`。

区别 4：应用程序补丁

Patch Manager 的主要目的是将补丁应用到操作系统。不过，您也可以使用 Patch Manager 将补丁应用到托管式节点上的某些应用程序。

Linux

在 Linux 操作系统上，Patch Manager 使用配置的存储库进行更新，不会区分操作系统补丁和应用程序补丁。您可以使用 Patch Manager 来定义从哪些存储库获取更新。有关更多信息，请参阅 如何指定备用补丁源存储库 (Linux)。

Windows

在 Windows Server 托管式节点上，您可以为 Microsoft 发布的应用程序（如 Microsoft Word 2016 和 Microsoft Exchange Server 2016）应用已批准规则，以及已批准和已被拒绝补丁例外。有关更多信息，请参阅 使用自定义补丁基准。

区别 5：自定义补丁基准中已拒绝的补丁列表选项

创建自定义补丁基准时，您可以为此已拒绝的补丁列表指定一个或多个补丁。对于 Linux 托管节点，如果它们是基准允许的其他补丁的依赖项，则也可以选择允许安装它们。

但是，Windows Server 不支持补丁依赖项的概念。您可以将补丁添加到 Windows Server 的自定义基准的已拒绝的补丁列表中，但结果取决于（1）被拒绝的补丁是否已安装在托管节点上，以及（2）您为被拒绝的补丁操作选择了哪个选项。

有关 Windows Server 上已拒绝的补丁选项的详细信息，请参阅下表。

安装状态	选项：“作为依赖项允许”	选项：“语句块”
补丁已安装	报告状态：INSTALLED_OTHER	报告状态：INSTALLED_REJECTED
尚未安装补丁	补丁已跳过	补丁已跳过

Microsoft 发布的每个 Windows Server 补丁通常都包含成功安装所需的所有信息。但是，有时可能需要必备程序包，您必须手动安装该程序包。 Patch Manager 不报告有关这些先决条件的信息。有关相关信息，请参阅 Microsoft 网站上的 Windows 更新问题排查。

适用于修补托管式节点的 SSM 命令文档

本主题介绍了目前可用的九个 Systems Manager 文档（SSM 文档），有助于您保持为托管式节点提供最新的安全性相关更新补丁。

我们建议在修补操作中仅使用这些文档中的五个。结合使用这五个 SSM 文档，您即可获得使用 AWS Systems Manager 进行修补的全部选项。其中有四个文档是在四个早期 SSM 文档之后发布的，替换了这四个早期文档，对功能进行了扩展和整合。

适用于修补的推荐 SSM 文档

我们建议在修补操作中使用以下五个 SSM 文档。

- AWS-ConfigureWindowsUpdate
- AWS-InstallWindowsUpdates
- AWS-RunPatchBaseline
- AWS-RunPatchBaselineAssociation
- AWS-RunPatchBaselineWithHooks

适用于修补的旧版 SSM 文档

以下四个旧版 SSM 文档仍可在某些 AWS 区域 中使用，但已不再更新，无法保证在所有场景中都能正常工作，将来可能不再受支持。不建议在修补操作中使用这些版本。

- AWS-ApplyPatchBaseline

- AWS-FindWindowsUpdates
- AWS-InstallMissingWindowsUpdates
- AWS-InstallSpecificWindowsUpdates

请参考以下各节，了解有关如何在修补操作中使用这些 SSM 文档的更多信息。

主题

- 用于修补托管式节点的推荐 SSM 文档
- 适用于修补托管式节点的旧 SSM 文档
- 用于修补的 SSM 命令文档：AWS-RunPatchBaseline
- 用于修补的 SSM 命令文档：AWS-RunPatchBaselineAssociation
- 用于修补的 SSM 命令文档：AWS-RunPatchBaselineWithHooks
- 在 AWS-RunPatchBaseline 或 AWS-RunPatchBaselineAssociation 中使用 InstallOverrideList 参数的示例场景
- 使用基准覆盖参数

用于修补托管式节点的推荐 SSM 文档

建议在进行托管式节点的修补操作时使用以下五个 SSM 文档。

推荐的 SSM 文档

- AWS-ConfigureWindowsUpdate
- AWS-InstallWindowsUpdates
- AWS-RunPatchBaseline
- AWS-RunPatchBaselineAssociation
- AWS-RunPatchBaselineWithHooks

AWS-ConfigureWindowsUpdate

支持配置基本的 Windows 更新功能，并使用它们来自动安装更新 (或关闭自动更新)。在所有 AWS 区域 中可用。

此 SSM 文档可提示 Windows 更新下载并安装指定的更新，并根据需要重启托管式节点。将此文档与 State Manager（AWS Systems Manager 中的一项工具）结合使用，确保 Windows Update 保持其配

置。您也可以使用 Run Command（AWS Systems Manager 中的一项工具）手动运行该文档来更改 Windows Update 配置。

本文档中的可用参数支持指定要安装的更新类别 (或是否关闭自动更新)，还支持指定在星期几的什么时间运行修补操作。如果您不需要对 Windows 更新进行严格控制，也不需要收集合规性信息，则此 SSM 文档最为有用。

替换早期 SSM 文档：

- 无

AWS-InstallWindowsUpdates

在 Windows Server 托管式节点上安装更新。在所有 AWS 区域 中可用。

如果您希望安装指定更新 (使用 `Include Kbs` 参数)，或希望安装特定类别或分类的补丁，但不需要补丁合规性信息，则此 SSM 文档可提供基本修补功能。

替换早期 SSM 文档：

- `AWS-FindWindowsUpdates`
- `AWS-InstallMissingWindowsUpdates`
- `AWS-InstallSpecificWindowsUpdates`

这三个早期文档执行不同的功能，但您可以使用较新的 SSM 文档 `AWS-InstallWindowsUpdates` 的不同参数设置得到同样的结果。这些参数设置在 适用于修补托管式节点的旧 SSM 文档 中进行了介绍。

AWS-RunPatchBaseline

在托管式节点上安装补丁或扫描节点，以确定是否缺少任何符合条件的补丁。在所有 AWS 区域 中可用。

`AWS-RunPatchBaseline` 允许您使用指定为操作系统类型的“默认”补丁基准来控制补丁批准。报告补丁合规性信息，您可以使用 Systems Manager 合规性工具进行查看。您可使用这些工具深入了解托管式节点的补丁合规性状态，例如哪些节点缺少补丁以及缺少哪些补丁。当您使用 `AWS-RunPatchBaseline` 时，将使用 `PutInventory` API 命令记录补丁合规性信息。对于 Linux 操作系统，为来自托管式节点上配置的默认源存储库和来自在自定义补丁基准中指定的任何备用源存储库的补丁提供合规性信息。有关备用源存储库的更多信息，请参阅 如何指定备用补丁源存储库 (Linux)。有关 Systems Manager 合规性工具的更多信息，请参阅 AWS Systems Manager Compliance。

替换早期文档：

- AWS-ApplyPatchBaseline

旧文档 AWS-ApplyPatchBaseline 仅适用于 Windows Server 托管式节点，不提供应用程序修补支持。较新的 AWS-RunPatchBaseline 对 Windows 和 Linux 系统提供了同等支持。要使用 AWS-RunPatchBaseline 文档，需要版本 2.0.834.0 或 SSM Agent 的更高版本。

有关如何使用 AWS-RunPatchBaseline SSM 文档的更多信息，请参阅 用于修补的 SSM 命令文档：AWS-RunPatchBaseline。

AWS-RunPatchBaselineAssociation

在实例上安装补丁，或扫描实例以确定是否缺少任何符合条件的补丁。在所有商业 AWS 区域 中提供。

AWS-RunPatchBaselineAssociation 与 AWS-RunPatchBaseline 的不同表现在几个重要方面：

- AWS-RunPatchBaselineAssociation 主要用于使用Quick Setup（AWS Systems Manager 中的一项工具）创建的State Manager关联。具体来说，当使用 Quick Setup 主机管理配置类型时，如果选择每天扫描实例以查找缺少的补丁选项，系统将使用 AWS-RunPatchBaselineAssociation 进行操作。

  但是，在大多数情况下，在设置您自己的修补操作时，应选择 AWS-RunPatchBaseline 或者 AWS-RunPatchBaselineWithHooks，而非 AWS-RunPatchBaselineAssociation。

  有关更多信息，请参阅以下主题：

  - AWS Systems Manager Quick Setup
  - 用于修补的 SSM 命令文档：AWS-RunPatchBaselineAssociation
- AWS-RunPatchBaselineAssociation 支持使用标签来标识在运行时使用哪个补丁基准来用于一组目标。
- 对于使用 AWS-RunPatchBaselineAssociation 的修补操作，将按照特定的 State Manager 关联来编译补丁合规性数据。在运行 AWS-RunPatchBaselineAssociation 时收集的补丁合规性数据，使用 PutComplianceItems API 命令而不是 PutInventory 命令来记录。这样可以防止覆盖与此特定关联无关的合规性数据。

  对于 Linux 操作系统，为来自实例上配置的默认源存储库和来自您在自定义补丁基准中指定的任何备用源存储库的补丁提供合规性信息。有关备用源存储库的更多信息，请参阅 如何指定备用补丁源

存储库 (Linux)。有关 Systems Manager 合规性工具的更多信息，请参阅 AWS Systems Manager Compliance。

替换早期文档：

- 无

有关如何使用 AWS-RunPatchBaselineAssociation SSM 文档的更多信息，请参阅 用于修补的 SSM 命令文档：AWS-RunPatchBaselineAssociation。

AWS-RunPatchBaselineWithHooks

在托管式节点上安装补丁或扫描节点，以确定是否缺少任何符合条件的补丁，您可以使用可选钩子在修补周期内在三个点运行 SSM 文档。在所有商业 AWS 区域 中提供。在 macOS 上不受支持。

在其 Install 操作中，AWS-RunPatchBaselineWithHooks 不同于 AWS-RunPatchBaseline。

AWS-RunPatchBaselineWithHooks 支持托管式节点修补期间在指定点运行的生命周期钩子。由于补丁安装有时需要重启托管式节点，修补操作分为两个事件，共有三个支持自定义功能的钩子。第一个钩子先于 Install with NoReboot 操作。第二个钩子后于 Install with NoReboot 操作。节点重启后，即可使用第三个钩子。

替换早期文档：

- 无

有关如何使用 AWS-RunPatchBaselineWithHooks SSM 文档的更多信息，请参阅 用于修补的 SSM 命令文档：AWS-RunPatchBaselineWithHooks。

适用于修补托管式节点的旧 SSM 文档

以下四个 SSM 文档仍可在某些 AWS 区域 中使用。不过，这些版本已不再更新，将来可能不再受支持，因此不建议使用。请使用 用于修补托管式节点的推荐 SSM 文档 中介绍的文档。

早期 SSM 文档

- AWS-ApplyPatchBaseline
- AWS-FindWindowsUpdates
- AWS-InstallMissingWindowsUpdates

- AWS-InstallSpecificWindowsUpdates

AWS-ApplyPatchBaseline

仅支持 Windows Server 托管式节点，但不包括对在其替换文档 AWS-RunPatchBaseline 中找到的修补应用程序的支持。在 2017 年 8 月之后推出的 AWS 区域 中不提供。

> **Note**
>
> 此 SSM 文档 AWS-RunPatchBaseline 的替换文档需要版本 2.0.834.0 或 SSM Agent 的更高版本。可以使用 AWS-UpdateSSMAgent 文档将托管式节点更新为代理的最新版本。

AWS-FindWindowsUpdates

已被 AWS-InstallWindowsUpdates 取代，可执行所有相同的操作。在 2017 年 4 月之后推出的 AWS 区域 中不提供。

要得到与这个早期 SSM 文档相同的结果，请使用推荐的替换文档 AWS-InstallWindowsUpdates 的以下参数配置：

- Action = Scan
- Allow Reboot = False

AWS-InstallMissingWindowsUpdates

已被 AWS-InstallWindowsUpdates 取代，可执行所有相同的操作。在 2017 年 4 月之后推出的所有 AWS 区域 中均不提供。

要得到与这个早期 SSM 文档相同的结果，请使用推荐的替换文档 AWS-InstallWindowsUpdates 的以下参数配置：

- Action = Install
- Allow Reboot = True

AWS-InstallSpecificWindowsUpdates

已被 AWS-InstallWindowsUpdates 取代，可执行所有相同的操作。在 2017 年 4 月之后推出的所有 AWS 区域 中均不提供。

要得到与这个早期 SSM 文档相同的结果，请使用推荐的替换文档 `AWS-InstallWindowsUpdates` 的以下参数配置：

- `Action = Install`
- `Allow Reboot = True`
- `Include Kbs = KB #########`

用于修补的 SSM 命令文档：**`AWS-RunPatchBaseline`**

AWS Systems Manager 支持 `AWS-RunPatchBaseline`，这是Patch Manager（AWS Systems Manager 中的一项工具）的 Systems Manager 文档（SSM 文档）。此 SSM 文档在托管式节点上为安全性相关更新和其他类型的更新执行修补操作。运行文档时，如果未指定补丁组，则使用为操作系统类型指定的“默认”的补丁基准。否则，它将使用与补丁组关联的补丁基准。有关补丁组的信息，请参阅 补丁组。

您可以使用文档 `AWS-RunPatchBaseline` 为操作系统和应用程序应用补丁。（在 Windows Server 上，应用程序支持仅限于更新 Microsoft 发布的应用程序。）

本文档支持 Linux、macOS和 Windows Server 托管式节点。此文档将为每个平台执行适当的操作。

> **Note**
>
> Patch Manager 还支持传统 SSM 文档 `AWS-ApplyPatchBaseline`。但是，此文档仅支持在 Windows 托管式节点上进行修补。建议改用 `AWS-RunPatchBaseline`，因为其同时支持在 Linux、macOS 和 Windows Server 托管式节点上进行修补。要使用 `AWS-RunPatchBaseline` 文档，需要版本 2.0.834.0 或 SSM Agent 的更高版本。

Windows Server

在 Windows Server 托管式节点上，`AWS-RunPatchBaseline` 文档下载并调用 PowerShell 模块，然后该模块下载适用于托管式节点的补丁基准快照。此补丁基准快照包含已批准的补丁列表，这些补丁通过查询 Windows 服务器更新服务 (WSUS) 服务器的补丁基准进行编译。将此列表传递给 Windows Update API，Windows Update API 根据需要控制已批准的补丁的下载和安装。

Linux

在 Linux 托管式节点上，`AWS-RunPatchBaseline` 文档调用 Python 模块，然后该模块下载适用于托管式节点的补丁基准快照。此补丁基准快照使用已定义规则及已批准和已阻止补丁列表驱动每个节点类型相应的软件包管理器：

- Amazon Linux 1、Amazon Linux 2、CentOS、Oracle Linux 和 RHEL 7 托管式节点使用 YUM。对于 YUM 操作，Patch Manager 需要使用 `Python 2.6` 或受支持的更高版本（2.6-3.10）。
- RHEL 8 托管式节点使用 DNF。对于 DNF 操作，Patch Manager 需要使用受支持版本的 `Python 2` 或 `Python 3`（2.6-3.10）。（默认情况下，RHEL 8 上不安装其中任一版本。您必须手动安装其中一个版本。）
- Debian Server、Raspberry Pi OS 和 Ubuntu Server 实例使用 APT。对于 APT 操作，Patch Manager 需要使用受支持版本的 `Python 3`（3.0-3.10）。
- SUSE Linux Enterprise Server 托管式节点使用 Zypper。对于 Zypper 操作，Patch Manager 需要使用 `Python 2.6` 或受支持的更高版本（2.6-3.10）。

macOS

在 macOS 托管式节点上，`AWS-RunPatchBaseline` 文档调用 Python 模块，然后该模块下载适用于托管式节点的补丁基准快照。接下来，Python 子进程在节点上调用 AWS Command Line Interface (AWS CLI)，以检索指定软件包管理器的安装和更新信息，并为每个更新软件包驱动适当的软件包管理器。

每个快照都专用于一个 AWS 账户、补丁组、操作系统和快照 ID。快照通过预签名的 Amazon Simple Storage Service (Amazon S3) URL 传送，该 URL 在快照创建 24 小时后过期。但是，在 URL 过期后，如果要将相同的快照内容应用于其他托管式节点，您可以在创建快照后最久 3 天内生成新的预签名 Amazon S3 URL。为此，请使用 get-deployable-patch-snapshot-for-instance 命令。

安装完所有已批准和适用的更新后，根据需要执行重启，然后在托管式节点上生成补丁合规性信息，并向 Patch Manager 报告。

> **Note**
>
> 如果在 `AWS-RunPatchBaseline` 文档中将 `RebootOption` 参数设置为 `NoReboot`，在 Patch Manager 运行后不会重启托管式节点。有关更多信息，请参阅 参数名称: `RebootOption`。

有关查看补丁合规性数据的信息，请参阅 关于补丁合规性。

`AWS-RunPatchBaseline` 参数

`AWS-RunPatchBaseline` 支持六个参数。`Operation` 参数是必需的。`InstallOverrideList`、`BaselineOverride` 和 `RebootOption` 参数是可选的。从技术上讲，`Snapshot-ID` 是可选的，但

我们建议在维护时段之外运行 AWS-RunPatchBaseline 时为其提供一个自定义值，而在维护时段操作中运行此文档时，让 Patch Manager 自动提供该值。

参数

- 参数名称: Operation
- 参数名称: AssociationId
- 参数名称: Snapshot ID
- 参数名称: InstallOverrideList
- 参数名称: RebootOption
- 参数名称: BaselineOverride
- 参数名称: StepTimeoutSeconds

参数名称: **Operation**

用法：必需。

选项：Scan | Install。

Scan

选择 Scan 选项时，AWS-RunPatchBaseline 确定托管式节点的补丁合规性状态并向 Patch Manager 报告此信息。Scan 不提示要安装的更新或要重启的托管式节点。相反，该操作会标识哪些地方缺少已批准并且适用于节点的更新。

安装

选择 Install 选项时，AWS-RunPatchBaseline 会尝试安装托管式节点中缺失并已批准的适用更新。在 Install 操作中生成的补丁合规性信息不会列出任何缺失的更新。但是，如果更新的安装因任何原因失败，它会报告处于失败状态的更新。一旦在托管式节点上安装了更新，就一定会重启节点，以确保正常安装和激活更新。（例外：如果将 AWS-RunPatchBaseline 文档中的 RebootOption 参数设置为 NoReboot，则在 Patch Manager 运行后不会重启托管式节点。有关更多信息，请参阅 参数名称: RebootOption。）

> **Note**
>
> 如果在 Patch Manager 更新托管式节点之前安装了基准规则指定的补丁，则系统可能无法按预期重启。当补丁是由用户手动安装或由其他程序（例如 Ubuntu Server 上的 unattended-upgrades 程序包）自动安装时，可能会发生这种情况。

参数名称: **AssociationId**

用法：可选。

AssociationId 是State Manager（AWS Systems Manager 中的一项工具）中现有关联的 ID。Patch Manager 使用它来将合规性数据添加到指定的关联上。此关联与在 Quick Setup 的补丁策略中设置的修补操作有关。

> **Note**
>
> 使用 AWS-RunPatchBaseline 时，如果在补丁策略基准覆盖的同时提供了一个 AssociationId 值，系统将以 PatchPolicy 操作来完成修补，AWS:ComplianceItem 中报告的 ExecutionType 值也是 PatchPolicy。如果未提供任何 AssociationId 值，系统将以 Command 操作来完成修补，提交的 AWS:ComplianceItem 中报告的 ExecutionType 值也是 Command。

如果您还没有要使用的关联，可以通过运行 create-association 命令来创建关联。

参数名称: **Snapshot ID**

用法：可选。

Snapshot ID 是 Patch Manager 使用的唯一 ID (GUID)，用于确保在单一操作中修补的一组托管式节点均具有完全相同的一组已批准补丁。尽管它定义为可选参数，根据是否在维护时段中运行 AWS-RunPatchBaseline，我们还是提供了不同的最佳实践建议，如下表所述。

AWS-RunPatchBaseline 最佳实践

Mode	最佳实践	详细信息
正在维护时段内运行 AWS-RunPatchBaseline	不要提供快照 ID，Patch Manager 将为您提供。	如果使用维护时段运行 AWS-RunPatchBaseline ，则不应提供自己生成的快照 ID。这种情况下，Systems Manager 基于维护时段执行 ID 提供 GUID 值。这可确保在维护时段中为 AWS-RunPatchBaseline 的所有调用使用正确的 ID。

Mode	最佳实践	详细信息
		如果在这种情况下您指定值，请注意，补丁基准的快照最多保留 3 天。之后，即使您在快照到期后指定相同的 ID，也将生成新的快照。
正在在维护时段以外运行 AWS-RunPatchBaseline	为快照 ID[1] 生成和指定自定义 GUID 值。	如果您不使用维护时段运行 AWS-RunPatchBaseline ，我们建议您为每个补丁基准生成并指定一个唯一的快照 ID，特别是于同一操作中在多个托管式节点上运行 AWS-RunPatchBaseline 文档时。如果在这种情况下不指定 ID，Systems Manager 会为向其发送命令的每个托管式节点生成不同的快照 ID。这可能会导致在托管式节点间指定不同的补丁集。 例如，假设您正在直接通过 Run Command（AWS Systems Manager 中的一项工具）运行 AWS-RunPatchBaseline 文档，并以一组 50 个的托管式节点为目标。指定自定义快照 ID 将生成单一基准快照，用于评估和修补所有节点，从而确保所有节点最终处于一致状态。

[1] 您可以使用任何能够生成 GUID 的工具为快照 ID 参数生成值。例如，在 PowerShell 中，可以使用 New-Guid cmdlet 生成格式为 12345699-9405-4f69-bc5e-9315aEXAMPLE 的 GUID。

参数名称: **`InstallOverrideList`**

用法：可选。

使用 `InstallOverrideList`，可以指定一个到要安装的补丁列表的 https URL 或 Amazon S3 路径样式 URL。此补丁安装列表（以 YAML 格式维护）会覆盖当前的默认补丁基准指定的补丁。这样，您可以更精细地控制在托管式节点上安装哪些补丁。

> **Important**
>
> `InstallOverrideList` 文件名不能包含以下字符：反引号 (`)、单引号 (')、双引号 (") 和美元符号 ($)。

使用 `InstallOverrideList` 参数时的修补操作行为在 Linux 和 macOS 托管式节点以及 Windows Server 托管式节点之间有所不同。在 Linux 和 macOS 上，Patch Manager 尝试应用 `InstallOverrideList` 补丁列表中包含的补丁（存在于节点上启用的任何存储库中），无论补丁是否符合补丁基准规则。不过，在 Windows Server 节点上，只有当 `InstallOverrideList` 补丁列表中的补丁也符合补丁基准规则时，才会应用补丁列表中的补丁。

请注意，合规性根据补丁基准中指定的内容而不是您在补丁的 `InstallOverrideList` 列表中指定的内容反映补丁状态。也就是说，Scan 操作会忽略 `InstallOverrideList` 参数。这是为了确保合规性报告根据策略而不是针对特定修补操作批准的内容持续反映补丁状态。

有关如何使用 `InstallOverrideList` 参数按不同维护时段计划将不同类型的补丁应用到目标组，同时仍使用单个补丁基准的说明，请参阅 在 AWS-RunPatchBaseline 或 AWS-RunPatchBaselineAssociation 中使用 InstallOverrideList 参数的示例场景。

有效的 URL 格式

> **Note**
>
> 如果您的文件存储在公开可用的存储桶中，您可以指定 https URL 格式或 Amazon S3 路径样式的 URL。如果您的文件存储在私有存储桶中，则必须指定 Amazon S3 路径样式的 URL。

- https URL 格式：

```
https://s3.aws-api-domain/amzn-s3-demo-bucket/my-windows-override-list.yaml
```

- Amazon S3 路径样式 URL：

```
s3://amzn-s3-demo-bucket/my-windows-override-list.yaml
```

有效的 YAML 内容格式

用于在列表中指定补丁的格式取决于托管式节点的操作系统。但是，一般格式如下所示：

```
patches:
    -
        id: '{patch-d}'
        title: '{patch-title}'
        {additional-fields}:{values}
```

尽管可以在 YAML 中提供额外字段，但是补丁操作会将其忽略。

此外，建议在 S3 存储桶中添加或更新列表之前验证 YAML 文件格式是否有效。有关 YAML 格式的更多信息，请参阅 yaml.org。对于验证工具选项，请在 Web 中搜索“yaml 格式验证程序”。

Linux

id

id 字段是必需的。它用来使用软件包名称和架构指定补丁。例如：`'dhclient.x86_64'`。您可以在 id 中使用通配符来指示多个软件包。例如：`'dhcp*'` 和 `'dhcp*1.*'`。

标题

title 字段是可选的，但它在 Linux 系统上提供额外的筛选功能。如果使用 title，它应包含以下格式之一的软件包版本信息：

YUM/SUSE Linux Enterprise Server (SLES)：

```
{name}.{architecture}:{epoch}:{version}-{release}
```

APT

```
{name}.{architecture}:{version}
```

对于 Linux 补丁标题，您可以在任意位置使用一个或多个通配符来扩展匹配软件包的数量。例如：`'*32:9.8.2-0.*.rc1.57.amzn1'`。

例如：

- 托管式节点上目前安装了 apt 软件包版本 1.2.25，但版本 1.2.27 现已可用。
- 将 apt.amd64 版本 1.2.27 添加到补丁列表中。它依赖于 apt utils.amd64 版本 1.2.27，但在列表中指定了 apt-utils.amd64 版本 1.2.25。

在这种情况下，将阻止安装 apt 版本 1.2.27 并报告为“Failed-NonCompliant.”。

Windows Server

id

id 字段是必需的。它用来使用 Microsoft 知识库 ID（例如 KB2736693）和 Microsoft 安全公告 ID（例如 MS17-023）指定补丁。

您在 Windows 补丁列表中提供的任何其他字段都是可选的，仅供您自己参考。您可以使用其他字段，如 title、classification、severity 等来提供关于指定补丁的更详细信息。

macOS

id

id 字段是必需的。id 字段的值可以使用 `{package-name}.{package-version}` 格式或 {软件包名称} 格式。

示例补丁列表

- Amazon Linux

```
patches:
    -
        id: 'kernel.x86_64'
    -
        id: 'bind*.x86_64'
        title: '32:9.8.2-0.62.rc1.57.amzn1'
    -
        id: 'glibc*'
    -
        id: 'dhclient*'
        title: '*12:4.1.1-53.P1.28.amzn1'
    -
        id: 'dhcp*'
```

```
        title: '*10:3.1.1-50.P1.26.amzn1'
```

- CentOS

```
patches:
    -
        id: 'kernel.x86_64'
    -
        id: 'bind*.x86_64'
        title: '32:9.8.2-0.62.rc1.57.amzn1'
    -
        id: 'glibc*'
    -
        id: 'dhclient*'
        title: '*12:4.1.1-53.P1.28.amzn1'
    -
        id: 'dhcp*'
        title: '*10:3.1.1-50.P1.26.amzn1'
```

- Debian Server

```
patches:
    -
        id: 'apparmor.amd64'
        title: '2.10.95-0ubuntu2.9'
    -
        id: 'cryptsetup.amd64'
        title: '*2:1.6.6-5ubuntu2.1'
    -
        id: 'cryptsetup-bin.*'
        title: '*2:1.6.6-5ubuntu2.1'
    -
        id: 'apt.amd64'
        title: '*1.2.27'
    -
        id: 'apt-utils.amd64'
        title: '*1.2.25'
```

- macOS

```
patches:
    -
        id: 'XProtectPlistConfigData'
```

```
    -
        id: 'MRTConfigData.1.61'
    -
        id: 'Command Line Tools for Xcode.11.5'
    -
        id: 'Gatekeeper Configuration Data'
```

- Oracle Linux

```
patches:
    -
        id: 'audit-libs.x86_64'
        title: '*2.8.5-4.el7'
    -
        id: 'curl.x86_64'
        title: '*.el7'
    -
        id: 'grub2.x86_64'
        title: 'grub2.x86_64:1:2.02-0.81.0.1.el7'
    -
        id: 'grub2.x86_64'
        title: 'grub2.x86_64:1:*-0.81.0.1.el7'
```

- Red Hat Enterprise Linux (RHEL)

```
patches:
    -
        id: 'NetworkManager.x86_64'
        title: '*1:1.10.2-14.el7_5'
    -
        id: 'NetworkManager-*.x86_64'
        title: '*1:1.10.2-14.el7_5'
    -
        id: 'audit.x86_64'
        title: '*0:2.8.1-3.el7'
    -
        id: 'dhclient.x86_64'
        title: '*.el7_5.1'
    -
        id: 'dhcp*.x86_64'
        title: '*12:5.2.5-68.el7'
```

- SUSE Linux Enterprise Server (SLES)

```
patches:
    -
        id: 'amazon-ssm-agent.x86_64'
    -
        id: 'binutils'
        title: '*0:2.26.1-9.12.1'
    -
        id: 'glibc*.x86_64'
        title: '*2.19*'
    -
        id: 'dhcp*'
        title: '0:4.3.3-9.1'
    -
        id: 'lib*'
```

- Ubuntu Server

```
patches:
    -
        id: 'apparmor.amd64'
        title: '2.10.95-0ubuntu2.9'
    -
        id: 'cryptsetup.amd64'
        title: '*2:1.6.6-5ubuntu2.1'
    -
        id: 'cryptsetup-bin.*'
        title: '*2:1.6.6-5ubuntu2.1'
    -
        id: 'apt.amd64'
        title: '*1.2.27'
    -
        id: 'apt-utils.amd64'
        title: '*1.2.25'
```

- Windows

```
patches:
    -
        id: 'KB4284819'
```

```
        title: '2018-06 Cumulative Update for Windows Server 2016 (1709) for x64-
based Systems (KB4284819)'
    -
        id: 'KB4284833'
    -
        id: 'KB4284835'
        title: '2018-06 Cumulative Update for Windows Server 2016 (1803) for x64-
based Systems (KB4284835)'
    -
        id: 'KB4284880'
    -
        id: 'KB4338814'
```

参数名称: **RebootOption**

用法：可选。

选项：RebootIfNeeded | NoReboot

默认值：RebootIfNeeded

> **Warning**
>
> 默认选项是 RebootIfNeeded。务必为应用场景选择正确的选项。例如，如果您的托管式节点必须立即重启才能完成配置过程，请选择 RebootIfNeeded。或者，如果您需要在计划的重启时间之前保持托管式节点可用，请选择 NoReboot。

> **Important**
>
> 我们不建议使用 Patch Manager 在 Amazon EMR（原 Amazon Elastic MapReduce）中修补集群实例。特别是，不要为 RebootOption 参数选择 RebootIfNeeded 选项。（此选项在 SSM 命令文档中可用，用于修补 AWS-RunPatchBaseline、AWS-RunPatchBaselineAssociation 和 AWS-RunPatchBaselineWithHooks。）使用 Patch Manager 进行修补的底层命令使用 yum 和 dnf 命令。因此，由于软件包的安装方式，这些操作会导致不兼容。有关在 Amazon EMR 集群上更新软件的首选方法的信息，请参阅《Amazon EMR Management Guide》中的 Using the default AMI for Amazon EMR。

RebootIfNeeded

选择 RebootIfNeeded 选项时，托管式节点将在以下任一情况下重启：

- Patch Manager 已安装一个或多个补丁。

  Patch Manager 不评估补丁是否必需重启。即使补丁不需要重启，系统也会重启。

- Patch Manager 检测 Install 操作期间的一个或多个状态为 INSTALLED_PENDING_REBOOT 的补丁。

  INSTALLED_PENDING_REBOOT 状态可能表示在上次运行 Install 操作时选择了选项 NoReboot，或者是自上次重新启动托管式节点以来在 Patch Manager 外部安装了补丁。

在这两种情况下重启托管式节点，可确保从内存中刷新更新的软件包，并在所有操作系统中保持修补和重启行为一致。

NoReboot

选择 NoReboot 选项时，即使托管式节点在 Install 操作期间安装了补丁，Patch Manager 也不会重启托管式节点。如果您知道托管式节点在应用补丁后不需要重启，或者应用程序或进程在节点上运行，但不应因修补操作重启而中断，则此选项有用。当您希望更多地控制托管式节点的重启时间（例如，使用维护时段来控制）时，该选项也很有用。

如果选择 NoReboot 选项并安装了补丁，则会为该补丁分配状态 InstalledPendingReboot。但托管式节点本身将被标记为 Non-Compliant。重启并运行 Scan 操作后，托管式节点状态将更新为 Compliant。

> **Important**
>
> NoReboot 选项仅阻止操作系统级别的重启，服务级别的重启仍可作为修补过程的一部分进行。例如，当 Docker 更新时，即使启用了 NoReboot，Amazon Elastic Container Service 等依赖服务也可能会自动重启。要使关键服务不被中断，可考虑采取其他措施，例如暂时将实例从服务中删除或在维护时段内安排修补。

补丁安装跟踪文件：为跟踪补丁安装，特别是自上次系统重启以来安装的补丁，Systems Manager 会在托管式节点上维护一个文件。

> **Important**
>
> 请勿删除或修改该跟踪文件。如果删除或损坏此文件，托管式节点的补丁合规性报告则不准确。如果发生这种情况，请重启节点并运行补丁扫描操作以还原此文件。

此跟踪文件存储在托管式节点上的以下位置：

- Linux 操作系统：
  - `/var/log/amazon/ssm/patch-configuration/patch-states-configuration.json`
  - `/var/log/amazon/ssm/patch-configuration/patch-inventory-from-last-operation.json`
- Windows Server 操作系统:
  - `C:\ProgramData\Amazon\PatchBaselineOperations\State\PatchStatesConfiguration.json`
  - `C:\ProgramData\Amazon\PatchBaselineOperations\State\PatchInventoryFromLastOperation.json`

参数名称: **`BaselineOverride`**

用法：可选。

您可以在运行时使用 `BaselineOverride` 参数来定义修补首选项。此基准覆盖在 S3 存储桶中以 JSON 对象的形式进行维护。它确保修补操作使用的所提供的基准与主机的操作系统相匹配，而不是应用默认补丁基准中的规则

> **Important**
>
> `BaselineOverride` 文件名不能包含以下字符：反引号 (`)、单引号 (')、双引号 (") 和美元符号 ($)。

有关如何使用 `BaselineOverride` 参数的更多信息，请参阅 使用基准覆盖参数。

参数名称: **`StepTimeoutSeconds`**

用法：可选。

命令在被视为失败前将会运行的时间（以秒为单位），介于 1 至 36000 秒（10 小时）之间。

用于修补的 SSM 命令文档：**AWS-RunPatchBaselineAssociation**

就像 AWS-RunPatchBaseline 文档，AWS-RunPatchBaselineAssociation 对安全相关更新及其他更新类型执行修补操作。您也可以使用文档 AWS-RunPatchBaselineAssociation 为操作系统和应用程序应用补丁。（在 Windows Server 上，应用程序支持仅限于更新 Microsoft 发布的应用程序。）

此文档为 Linux、macOS 和 Windows Server 支持 Amazon Elastic Compute Cloud (Amazon EC2) 实例。它不支持混合和多云环境中的非 EC2 节点。此文档将为每个平台执行相应的操作，在 Linux 上调用 Python 模块和 macOS 实例，以及在 Windows 实例上调用 PowerShell 模块。

但是，AWS-RunPatchBaselineAssociation 与 AWS-RunPatchBaseline 存在以下差异：

- AWS-RunPatchBaselineAssociation 主要用于使用Quick Setup（AWS Systems Manager 中的一项工具）创建的State Manager关联。具体来说，当使用 Quick Setup 主机管理配置类型时，如果选择每天扫描实例以查找缺少的补丁选项，系统将使用 AWS-RunPatchBaselineAssociation 进行操作。

  但是，在大多数情况下，在设置自己的修补操作时，应选择 AWS-RunPatchBaseline 或者 AWS-RunPatchBaselineWithHooks，而非 AWS-RunPatchBaselineAssociation。
- 在使用 AWS-RunPatchBaselineAssociation 文档时，您可以在该文档的 BaselineTags 参数字段中指定标签键对。如果您 AWS 账户中的自定义补丁基准共享这些标签，则Patch Manager（AWS Systems Manager 中的一项工具）会在目标实例上运行时使用该标记的基准，而不是当前为操作系统类型指定的“默认”补丁基准。

  > **Note**
  >
  > 如果您选择在修补操作中使用 AWS-RunPatchBaselineAssociation，而不是使用 Quick Setup 设置的内容，并且您想要使用其可选的 BaselineTags 参数时，您必须为实例配置文件提供一些额外权限，以用于 Amazon Elastic Compute Cloud (Amazon EC2) 实例。有关更多信息，请参阅 参数名称: BaselineTags。

  以下两种格式都适用于 BaselineTags 参数：

  Key=*tag-key*,Values=*tag-value*

  Key=*tag-key*,Values=*tag-value1*,*tag-value2*,*tag-value3*

> **Important**
>
> 标签键和值不能包含以下字符：反引号 (`)、单引号 (')、双引号 (") 和美元符号 ($)。

- 在运行 AWS-RunPatchBaselineAssociation 时，其收集的补丁合规性数据将使用 PutComplianceItems API 命令而不是 PutInventory 命令（该命令被 AWS-RunPatchBaseline 使用）来记录。这种差异意味着，所存储和报告的补丁合规性信息是依据特定关联的。在此关联之外生成的补丁合规性数据不会被覆盖。
- 运行 AWS-RunPatchBaselineAssociation 后所报告的补丁合规性信息显示某个实例是否符合规定。它不包括补丁级别的详细信息，如以下 AWS Command Line Interface (AWS CLI) 命令的输出所显示的那样。命令筛选 Association 作为合规性类型：

```
aws ssm list-compliance-items \
    --resource-ids "i-02573cafcfEXAMPLE" \
    --resource-types "ManagedInstance" \
    --filters "Key=ComplianceType,Values=Association,Type=EQUAL" \
    --region us-east-2
```

系统将返回类似于以下内容的信息。

```
{
    "ComplianceItems": [
        {
            "Status": "NON_COMPLIANT",
            "Severity": "UNSPECIFIED",
            "Title": "MyPatchAssociation",
            "ResourceType": "ManagedInstance",
            "ResourceId": "i-02573cafcfEXAMPLE",
            "ComplianceType": "Association",
            "Details": {
                "DocumentName": "AWS-RunPatchBaselineAssociation",
                "PatchBaselineId": "pb-0c10e65780EXAMPLE",
                "DocumentVersion": "1"
            },
            "ExecutionSummary": {
                "ExecutionTime": 1590698771.0
            },
            "Id": "3e5d5694-cd07-40f0-bbea-040e6EXAMPLE"
        }
    ]
```

```
}
```

如果已将标签键对值指定为 AWS-RunPatchBaselineAssociation 文档的参数，则 Patch Manager 会搜索自定义补丁基准，该基准与操作系统类型匹配且已使用相同标签键对做了标记。此搜索不限于当前指定的默认补丁程序基准或分配给补丁组的基准。如果未找到具有指定标签的基准，下一步 Patch Manager 会查找补丁组（如果在运行 AWS-RunPatchBaselineAssociation 的命令中已经指定了）。如果没有匹配的补丁组，Patch Manager 将回退到操作系统账户的当前默认补丁基准。

如果找到多个具有 AWS-RunPatchBaselineAssociation 文档中的指定标签的补丁基准，则 Patch Manager 返回一条错误消息，指示只能使用该键值对标记一个补丁基准，以便继续执行操作。

> **Note**
>
> 在 Linux 实例上，每种实例类型的相应软件包管理器被用来安装软件包：
>
> - Amazon Linux 1、Amazon Linux 2、CentOS、Oracle Linux 和 RHEL 实例使用 YUM。对于 YUM 操作，Patch Manager 需要使用 Python 2.6 或受支持的更高版本（2.6-3.10）。
> - Debian Server、Raspberry Pi OS 和 Ubuntu Server 实例使用 APT。对于 APT 操作，Patch Manager 需要使用受支持版本的 Python 3（3.0-3.10）。
> - SUSE Linux Enterprise Server 实例使用 Zypper。对于 Zypper 操作，Patch Manager 需要使用 Python 2.6 或受支持的更高版本（2.6-3.10）。

在扫描完毕，或者所有已批准和适用的更新安装完毕后，根据需要执行重启，然后在实例上生成补丁合规性信息，并向 Patch Compliance 服务报告。

> **Note**
>
> 如果在 AWS-RunPatchBaselineAssociation 文档中将 RebootOption 参数设置为 NoReboot，在 Patch Manager 运行后不会重启实例。有关更多信息，请参阅 参数名称: RebootOption。

有关查看补丁合规性数据的信息，请参阅 关于补丁合规性。

AWS-RunPatchBaselineAssociation 参数

AWS-RunPatchBaselineAssociation 支持五个参数。Operation 和 AssociationId 参数是必需的。InstallOverrideList、 RebootOption 和 BaselineTags 参数可选。

参数

- 参数名称: Operation
- 参数名称: BaselineTags
- 参数名称: AssociationId
- 参数名称: InstallOverrideList
- 参数名称: RebootOption

参数名称: Operation

用法：必需。

选项：Scan | Install。

Scan

选择 Scan 选项时，AWS-RunPatchBaselineAssociation 确定实例的补丁合规性状态，并向 Patch Manager 报告此信息。Scan 不提示要安装的更新或要重启的实例。相反，此操作会标识缺少哪些已批准并且适用于此实例的更新。

安装

选择 Install 选项时，AWS-RunPatchBaselineAssociation 尝试安装实例中缺失的已批准并且适用的更新。在 Install 操作中生成的补丁合规性信息不会列出任何缺失的更新。但是，如果更新的安装因任何原因失败，它会报告处于失败状态的更新。只要在实例上安装了更新，就一定会重启实例，以确保更新正常安装和激活。（例外：如果将 AWS-RunPatchBaselineAssociation 文档中的 RebootOption 参数设置为 NoReboot，则在 Patch Manager 运行后不会重启实例。有关更多信息，请参阅 参数名称: RebootOption。）

> **Note**
>
> 如果在 Patch Manager 更新实例之前 安装了基准规则指定的补丁，则系统可能无法按预期重启。当补丁是由用户手动安装或由其他程序（例如 Ubuntu Server 上的 unattended-upgrades 程序包）自动安装时，可能会发生这种情况。

参数名称: **BaselineTags**

用法：可选。

BaselineTags 是您选择并分配给单个自定义补丁基准的唯一标签键值对。可以为此参数指定一个或多个值。以下两种格式均为有效：

Key=*tag-key*,Values=*tag-value*

Key=*tag-key*,Values=*tag-value1*,*tag-value2*,*tag-value3*

> **Important**
>
> 标签键和值不能包含以下字符：反引号 (`)、单引号 (')、双引号 (") 和美元符号 ($)。

BaselineTags 值是 Patch Manager 使用的值，用于确保在单一操作中修补的一组实例都具有完全相同的一组已批准补丁。当执行修补操作时，Patch Manager 检查操作系统类型的补丁基准是否使用为 BaselineTags 指定的相同键值对进行了标记。如果存在匹配项，则使用此自定义补丁基准。如果不匹配，则根据为修补操作指定的任何补丁组来标识补丁基准。如果没有，AWS 使用该操作系统的托管预定义补丁基准。

其他权限要求

如果您在修补操作中使用 AWS-RunPatchBaselineAssociation，而不是使用 Quick Setup 设置的内容，并且您想要使用可选的 BaselineTags 参数，则必须将以下权限添加到实例配置文件，以用于 Amazon Elastic Compute Cloud (Amazon EC2) 实例。

> **Note**
>
> Quick Setup 和 AWS-RunPatchBaselineAssociation 不支持本地服务器和虚拟机 (VM)。

```
{
    "Effect": "Allow",
    "Action": [
        "ssm:DescribePatchBaselines",
        "tag:GetResources"
    ],
```

```
    "Resource": "*"
},
{
    "Effect": "Allow",
    "Action": [
        "ssm:GetPatchBaseline",
        "ssm:DescribeEffectivePatchesForPatchBaseline"
    ],
    "Resource": "patch-baseline-arn"
}
```

将*#### ARN* 替换为使用要提供访问权限的补丁基准的 Amazon Resource Name (ARN)，格式为 `arn:aws:ssm:us-east-2:123456789012:patchbaseline/pb-0c10e65780EXAMPLE`。

参数名称: **`AssociationId`**

用法：必需。

`AssociationId` 是State Manager（AWS Systems Manager 中的一项工具）中现有关联的 ID。Patch Manager 使用它来将合规性数据添加到指定的关联上。此关联与在 Quick Setup 中创建的主机管理配置中启用的补丁 `Scan` 操作有关。通过将修补结果作为关联合规性数据而不是库存合规性数据发送，您的实例的现有库存合规性信息不会在修补操作后被覆盖，同样，也不会覆盖其他关联 ID 的信息。如果您还没有要使用的关联，可以通过运行 create-association 命令来创建关联。例如：

Linux & macOS

```
aws ssm create-association \
    --name "AWS-RunPatchBaselineAssociation" \
    --association-name "MyPatchHostConfigAssociation" \
    --targets
 "Key=instanceids,Values=[i-02573cafcfEXAMPLE,i-07782c72faEXAMPLE,i-07782c72faEXAMPLE]"
 \
    --parameters "Operation=Scan" \
    --schedule-expression "cron(0 */30 * * * ? *)" \
    --sync-compliance "MANUAL" \
    --region us-east-2
```

Windows Server

```
aws ssm create-association ^
    --name "AWS-RunPatchBaselineAssociation" ^
    --association-name "MyPatchHostConfigAssociation" ^
```

```
    --targets
"Key=instanceids,Values=[i-02573cafcfEXAMPLE,i-07782c72faEXAMPLE,i-07782c72faEXAMPLE]"
^
    --parameters "Operation=Scan" ^
    --schedule-expression "cron(0 */30 * * * ? *)" ^
    --sync-compliance "MANUAL" ^
    --region us-east-2
```

参数名称: **`InstallOverrideList`**

用法：可选。

使用 `InstallOverrideList`，为要安装的补丁列表指定 https URL 或 Amazon Simple Storage Service (Amazon S3) 路径样式 URL。此补丁安装列表（以 YAML 格式维护）会覆盖当前的默认补丁基准指定的补丁。这样，您可以更精细地控制实例上安装的补丁。

> **Important**
>
> `InstallOverrideList` 文件名不能包含以下字符：反引号 (`)、单引号 (')、双引号 (") 和美元符号 ($)。

使用 `InstallOverrideList` 参数时的修补操作行为在 Linux 和 macOS 托管式节点以及 Windows Server 托管式节点之间有所不同。在 Linux 和 macOS 上，Patch Manager 尝试应用 `InstallOverrideList` 补丁列表中包含的补丁（存在于节点上启用的任何存储库中），无论补丁是否符合补丁基准规则。不过，在 Windows Server 节点上，只有当 `InstallOverrideList` 补丁列表中的补丁也符合补丁基准规则时，才会应用补丁列表中的补丁。

请注意，合规性根据补丁基准中指定的内容而不是您在补丁的 `InstallOverrideList` 列表中指定的内容反映补丁状态。也就是说，Scan 操作会忽略 `InstallOverrideList` 参数。这是为了确保合规性报告根据策略而不是针对特定修补操作批准的内容持续反映补丁状态。

有效的 URL 格式

> **Note**
>
> 如果您的文件存储在公开可用的存储桶中，您可以指定 https URL 格式或 Amazon S3 路径样式的 URL。如果您的文件存储在私有存储桶中，则必须指定 Amazon S3 路径样式的 URL。

- https URL 格式示例：

```
https://s3.amazonaws.com/amzn-s3-demo-bucket/my-windows-override-list.yaml
```

- Amazon S3 路径样式 URL 示例：

```
s3://amzn-s3-demo-bucket/my-windows-override-list.yaml
```

有效的 YAML 内容格式

用在列表中指定补丁的格式取决于实例的操作系统。但是，一般格式如下所示：

```
patches:
    -
        id: '{patch-d}'
        title: '{patch-title}'
        {additional-fields}:{values}
```

尽管可以在 YAML 中提供额外字段，但是补丁操作会将其忽略。

此外，建议在 S3 存储桶中添加或更新列表之前验证 YAML 文件格式是否有效。有关 YAML 格式的更多信息，请参阅 yaml.org。对于验证工具选项，请在 Web 中搜索“yaml 格式验证程序”。

- Microsoft Windows

  id

  id 字段是必需的。它用来使用 Microsoft 知识库 ID（例如 KB2736693）和 Microsoft 安全公告 ID（例如 MS17-023）指定补丁。

  您在 Windows 补丁列表中提供的任何其他字段都是可选的，仅供您自己参考。您可以使用其他字段，如 title、classification、severity 等来提供关于指定补丁的更详细信息。

- Linux

  id

  id 字段是必需的。它用来使用软件包名称和架构指定补丁。例如：`'dhclient.x86_64'`。您可以在 id 中使用通配符来指示多个软件包。例如：`'dhcp*'` 和 `'dhcp*1.*'`。

  删除实例快照

title 字段是可选的，但它在 Linux 系统上提供额外的筛选功能。如果使用 title，它应包含以下格式之一的软件包版本信息：

YUM/SUSE Linux Enterprise Server (SLES)：

```
{name}.{architecture}:{epoch}:{version}-{release}
```

APT

```
{name}.{architecture}:{version}
```

对于 Linux 补丁标题，您可以在任意位置使用一个或多个通配符来扩展匹配软件包的数量。例如：`'*32:9.8.2-0.*.rc1.57.amzn1'`。

例如：

- 实例上目前安装了 Apt 软件包版本 1.2.25，但版本 1.2.27 现已可用。
- 将 apt.amd64 版本 1.2.27 添加到补丁列表中。它依赖于 apt utils.amd64 版本 1.2.27，但在列表中指定了 apt-utils.amd64 版本 1.2.25。

在这种情况下，将阻止安装 apt 版本 1.2.27 并报告为“Failed-NonCompliant.”。

其他字段

您在 Linux 补丁列表中提供的任何其他字段都是可选的，仅供您自己参考。您可以使用其他字段，如 classification、severity 等来提供关于指定补丁的更详细信息。

示例补丁列表

- Windows

```
patches:
    -
        id: 'KB4284819'
        title: '2018-06 Cumulative Update for Windows Server 2016 (1709) for x64-
based Systems (KB4284819)'
    -
        id: 'KB4284833'
    -
        id: 'KB4284835'
```

```
        title: '2018-06 Cumulative Update for Windows Server 2016 (1803) for x64-
based Systems (KB4284835)'
    -
        id: 'KB4284880'
    -
        id: 'KB4338814'
```

- APT

```
patches:
    -
        id: 'apparmor.amd64'
        title: '2.10.95-0ubuntu2.9'
    -
        id: 'cryptsetup.amd64'
        title: '*2:1.6.6-5ubuntu2.1'
    -
        id: 'cryptsetup-bin.*'
        title: '*2:1.6.6-5ubuntu2.1'
    -
        id: 'apt.amd64'
        title: '*1.2.27'
    -
        id: 'apt-utils.amd64'
        title: '*1.2.25'
```

- Amazon Linux

```
patches:
    -
        id: 'kernel.x86_64'
    -
        id: 'bind*.x86_64'
        title: '32:9.8.2-0.62.rc1.57.amzn1'
    -
        id: 'glibc*'
    -
        id: 'dhclient*'
        title: '*12:4.1.1-53.P1.28.amzn1'
    -
        id: 'dhcp*'
        title: '*10:3.1.1-50.P1.26.amzn1'
```

- Red Hat Enterprise Linux (RHEL)

```
patches:
    -
        id: 'NetworkManager.x86_64'
        title: '*1:1.10.2-14.el7_5'
    -
        id: 'NetworkManager-*.x86_64'
        title: '*1:1.10.2-14.el7_5'
    -
        id: 'audit.x86_64'
        title: '*0:2.8.1-3.el7'
    -
        id: 'dhclient.x86_64'
        title: '*.el7_5.1'
    -
        id: 'dhcp*.x86_64'
        title: '*12:5.2.5-68.el7'
```

- SUSE Linux Enterprise Server (SLES)

```
patches:
    -
        id: 'amazon-ssm-agent.x86_64'
    -
        id: 'binutils'
        title: '*0:2.26.1-9.12.1'
    -
        id: 'glibc*.x86_64'
        title: '*2.19*'
    -
        id: 'dhcp*'
        title: '0:4.3.3-9.1'
    -
        id: 'lib*'
```

- Ubuntu Server

```
patches:
    -
        id: 'apparmor.amd64'
```

```
        title: '2.10.95-0ubuntu2.9'
    -
        id: 'cryptsetup.amd64'
        title: '*2:1.6.6-5ubuntu2.1'
    -
        id: 'cryptsetup-bin.*'
        title: '*2:1.6.6-5ubuntu2.1'
    -
        id: 'apt.amd64'
        title: '*1.2.27'
    -
        id: 'apt-utils.amd64'
        title: '*1.2.25'
```

- Windows

```
patches:
    -
        id: 'KB4284819'
        title: '2018-06 Cumulative Update for Windows Server 2016 (1709) for x64-
based Systems (KB4284819)'
    -
        id: 'KB4284833'
    -
        id: 'KB4284835'
        title: '2018-06 Cumulative Update for Windows Server 2016 (1803) for x64-
based Systems (KB4284835)'
    -
        id: 'KB4284880'
    -
        id: 'KB4338814'
```

参数名称: **RebootOption**

用法：可选。

选项：RebootIfNeeded | NoReboot

默认值：RebootIfNeeded

Warning

默认选项是 RebootIfNeeded。务必为应用场景选择正确的选项。例如，如果您的实例必须立即重启才能完成配置过程，请选择 RebootIfNeeded。或者，如果您需要在计划的重启时间之前保持实例可用，请选择 NoReboot。

Important

NoReboot 选项仅阻止操作系统级别的重启，服务级别的重启仍可作为修补过程的一部分进行。例如，当 Docker 更新时，即使启用了 NoReboot，Amazon Elastic Container Service 等依赖服务也可能会自动重启。要使关键服务不被中断，可考虑采取其他措施，例如暂时将实例从服务中删除或在维护时段内安排修补。

Important

我们不建议使用 Patch Manager 在 Amazon EMR（原 Amazon Elastic MapReduce）中修补集群实例。特别是，不要为 RebootOption 参数选择 RebootIfNeeded 选项。（此选项在 SSM 命令文档中可用，用于修补 AWS-RunPatchBaseline、AWS-RunPatchBaselineAssociation 和 AWS-RunPatchBaselineWithHooks。）使用 Patch Manager 进行修补的底层命令使用 yum 和 dnf 命令。因此，由于软件包的安装方式，这些操作会导致不兼容。有关在 Amazon EMR 集群上更新软件的首选方法的信息，请参阅《Amazon EMR Management Guide》中的 Using the default AMI for Amazon EMR。

RebootIfNeeded

如果您在选择 RebootIfNeeded 选项，则在以下任一情况下重启实例：

- Patch Manager 已安装一个或多个补丁。

  Patch Manager 不评估补丁是否必需重启。即使补丁不需要重启，系统也会重启。

- Patch Manager 检测 Install 操作期间的一个或多个状态为 INSTALLED_PENDING_REBOOT 的补丁。

  INSTALLED_PENDING_REBOOT 状态可能表示在上次运行 Install 操作时选择了选项 NoReboot，或者是自上次重新启动托管式节点以来在 Patch Manager 外部安装了补丁。

在这两种情况下重启实例，可确保从内存中刷新更新的软件包，并在所有操作系统中保持修补和重启行为的一致。

NoReboot

选择 NoReboot 选项后，即使实例在 Install 操作期间安装了补丁，Patch Manager 也不会重启实例。如果您知道您的实例在应用补丁后不需要重启，或者您的应用程序或进程在实例上运行，但不应因修补操作重启而中断，则此选项有用。当您希望更多地控制实例重启的时间（例如，使用维护时段）时，该选项也很有用。

补丁安装跟踪文件：为了跟踪补丁安装，特别是自上次重启系统以来安装的补丁，Systems Manager 会在托管实例上维护一个文件。

> **Important**
>
> 请勿删除或修改该跟踪文件。如果删除或损坏此文件，实例的补丁合规性报告则不准确。如果发生这种情况，请重启实例并运行补丁扫描操作以还原文件。

此跟踪文件存储在托管实例上的以下位置：

- Linux 操作系统：
  - `/var/log/amazon/ssm/patch-configuration/patch-states-configuration.json`
  - `/var/log/amazon/ssm/patch-configuration/patch-inventory-from-last-operation.json`
- Windows Server 操作系统:
  - `C:\ProgramData\Amazon\PatchBaselineOperations\State\PatchStatesConfiguration.json`
  - `C:\ProgramData\Amazon\PatchBaselineOperations\State\PatchInventoryFromLastOperation.json`

用于修补的 SSM 命令文档：**`AWS-RunPatchBaselineWithHooks`**

AWS Systems Manager 支持 `AWS-RunPatchBaselineWithHooks`，这是Patch Manager（AWS Systems Manager 中的一项工具）的 Systems Manager 文档（SSM 文档）。此 SSM 文档在托管式节点上为安全性相关更新和其他类型的更新执行修补操作。

`AWS-RunPatchBaselineWithHooks` 与 `AWS-RunPatchBaseline` 存在以下差异：

- 包装文档– AWS-RunPatchBaselineWithHooks 是 AWS-RunPatchBaseline 的包装器并依赖 AWS-RunPatchBaseline 进行一些操作。
- **Install** 操作 – AWS-RunPatchBaselineWithHooks 支持托管式节点修补期间在指定点运行的生命周期钩子。由于补丁安装有时需要重启托管式节点，修补操作分为两个事件，共有三个支持自定义功能的钩子。第一个钩子先于 Install with NoReboot 操作。第二个钩子后于 Install with NoReboot 操作。托管式节点重启后，即可使用第三个钩子。
- 不支持自定义补丁列表– AWS-RunPatchBaselineWithHooks 不支持 InstallOverrideList 参数。
- SSM Agent 支持 – AWS-RunPatchBaselineWithHooks 要求 SSM Agent 3.0.502 或更高版本需安装在托管式节点上才能进行修补。

运行文档时，如果未指定补丁组，则使用当前指定为操作系统类型的 “默认” 补丁基准。否则，它将使用与补丁组关联的补丁基准。有关补丁组的更多信息，请参阅 补丁组。

您可以使用文档 AWS-RunPatchBaselineWithHooks 为操作系统和应用程序应用补丁。（在 Windows Server 上，应用程序支持仅限于更新 Microsoft 发布的应用程序。）

本文档支持 Linux 和 Windows Server 托管式节点。此文档将为每个平台执行适当的操作。

> **Note**
>
> AWS-RunPatchBaselineWithHooks 在 macOS 上不受支持。

Linux

在 Linux 托管式节点上，AWS-RunPatchBaselineWithHooks 文档调用 Python 模块，然后该模块下载适用于托管式节点的补丁基准快照。此补丁基准快照使用已定义规则及已批准和已阻止补丁列表驱动每个节点类型相应的软件包管理器：

- Amazon Linux 1、Amazon Linux 2、CentOS、Oracle Linux 和 RHEL 7 托管式节点使用 YUM。对于 YUM 操作，Patch Manager 需要使用 Python 2.6 或受支持的更高版本（2.6-3.10）。
- RHEL 8 托管式节点使用 DNF。对于 DNF 操作，Patch Manager 需要使用受支持版本的 Python 2 或 Python 3（2.6-3.10）。（默认情况下，RHEL 8 上不安装其中任一版本。您必须手动安装其中一个版本。）
- Debian Server、Raspberry Pi OS 和 Ubuntu Server 实例使用 APT。对于 APT 操作，Patch Manager 需要使用受支持版本的 Python 3（3.0-3.10）。

- SUSE Linux Enterprise Server 托管式节点使用 Zypper。对于 Zypper 操作，Patch Manager 需要使用 Python 2.6 或受支持的更高版本（2.6-3.10）。

Windows Server

在 Windows Server 托管式节点上，AWS-RunPatchBaselineWithHooks 文档下载并调用 PowerShell 模块，然后该模块下载适用于托管式节点的补丁基准快照。此补丁基准快照包含已批准的补丁列表，这些补丁通过查询 Windows 服务器更新服务 (WSUS) 服务器的补丁基准进行编译。将此列表传递给 Windows Update API，Windows Update API 控制相应的已批准补丁的下载和安装。

每个快照都特定于一个 AWS 账户、补丁组、操作系统和快照 ID。快照通过预签名的 Amazon Simple Storage Service (Amazon S3) URL 传送，该 URL 在快照创建 24 小时后过期。但是，在 URL 过期后，如果要将相同的快照内容应用于其他托管式节点，您可以在创建快照后最久三天内生成新的预签名 Amazon S3 URL。为此，请使用 get-deployable-patch-snapshot-for-instance 命令。

安装完所有已批准和适用的更新后，根据需要执行重启，然后在托管式节点上生成补丁合规性信息，并向 Patch Manager 报告。

如果在 AWS-RunPatchBaselineWithHooks 文档中将 RebootOption 参数设置为 NoReboot，在 Patch Manager 运行后不会重启托管式节点。有关更多信息，请参阅 参数名称: RebootOption。

> **Important**
>
> 尽管 NoReboot 选项可以阻止操作系统重启，但无法阻止某些软件包更新时可能发生的服务级别重启。例如，在更新 Docker 等软件包时，即使已指定 NoReboot，仍可能会自动重启依赖服务（例如容器编排服务）。

有关查看补丁合规性数据的信息，请参阅 关于补丁合规性。

AWS-RunPatchBaselineWithHooks 操作步骤

当 AWS-RunPatchBaselineWithHooks 运行时，将执行以下步骤：

1. 扫描 - Scan 操作使用 AWS-RunPatchBaseline 在托管式节点上运行，并生成和上传合规性报告。
2. 验证本地补丁状态 - 运行脚本以确定根据所选操作和步骤 1 的 Scan 结果将执行哪些步骤。
   a. 如果选定的操作为 Scan，则操作将被标记为完成。操作结束。

b. 如果选定的 操作为 Install，则 Patch Manager 评估步骤1的 Scan 结果，以确定接下来要运行的内容：

i. 如果未检测到缺少的补丁，并且不需要暂挂重启，则操作直接进行到最后一步（步骤 8），其中包括您提供的钩子。跳过两者之间的任何步骤。

ii. 如果未检测到缺少补丁，但需要挂起重新引导，并且选定的重启选项为 NoReboot，则操作直接进行到最后一步（步骤 8），其中包括您提供的钩子。跳过两者之间的任何步骤。

iii. 否则，操作将继续下一步。

3. 预修补钩子操作 – 为第一个生命周期钩子提供的 SSM 文档 PreInstallHookDocName 在托管式节点上运行。

4. 使用 NoReboot 安装 – Install 操作，重启选项为 NoReboot，使用 AWS-RunPatchBaseline 在托管式节点上运行，生成并上传合规性报告。

5. 安装后钩子操作 – 为第二个生命周期钩子提供的 SSM 文档 PostInstallHookDocName 在托管式节点上运行。

6. 验证重启 – 运行脚本以确定托管式节点是否需要重启以及要运行哪些步骤：

a. 如果选定的重启选项为 NoReboot，则操作直接进行到最后一步（步骤 8），其中包括您提供的钩子。跳过两者之间的任何步骤。

b. 如果选定的重启选项为 RebootIfNeeded、Patch Manager 检查是否需要在步骤 4 中收集的库存中进行任何暂挂重启。这意味着在以下任一情况下，操作会继续执行步骤 7，并且托管式节点将重新启动：

i. Patch Manager 安装了一个或多个补丁。（Patch Manager 不会评估补丁是否需要重启系统。即使补丁不需要重启，系统也会重启。）

ii. 在安装操作期间，Patch Manager 检测到一个或多个状态为 INSTALLED_PENDING_REBOOT 的补丁。INSTALLED_PENDING_REBOOT 状态可能表示上次运行安装操作时选择了选项 NoReboot，或者是自上次重新启动托管式节点以来在 Patch Manager 外部安装了补丁。

如果未找到符合标准的补丁，则托管式节点修补操作完成，操作直接进入最后一个步骤（步骤 8），该步骤包含您提供的挂钩。跳过两者之间的任何步骤。

7. 重启并报告 – 带有重启选项 RebootIfNeeded 的安装操作，使用 AWS-RunPatchBaseline 在托管式节点上运行，生成并上传合规性报告。

8. 重启后钩子操作 – 为第三个生命周期钩子提供的 SSM 文档 OnExitHookDocName 在托管式节点上运行。

对于 Scan 操作，如果步骤 1 失败，则运行文档的进程将停止，并将该步骤报告为失败，尽管后续步骤报告为成功。

对于 Install 操作，如果任一 aws:runDocument 步骤在操作期间失败，则这些步骤将被报告为失败，操作直接进行到最后一步（步骤 8），其中包括您提供的钩子。跳过两者之间的任何步骤。此步骤被报告为失败，最后一步报告其操作结果的状态，其间的所有步骤均报告为成功。

AWS-RunPatchBaselineWithHooks 参数

AWS-RunPatchBaselineWithHooks 支持六个参数。

Operation 参数是必需的。

RebootOption、 PreInstallHookDocName、 PostInstallHookDocName 和 OnExitHookDocName 参数可选。

Snapshot-ID 在技术上是可选的，但是我们建议您在维护时段以外运行 AWS-RunPatchBaselineWithHooks 时为其提供自定义值。让 Patch Manager 在维护时段操作期间运行文档时自动提供值。

参数

- 参数名称: Operation
- 参数名称: Snapshot ID
- 参数名称: RebootOption
- 参数名称: PreInstallHookDocName
- 参数名称: PostInstallHookDocName
- 参数名称: OnExitHookDocName

参数名称: **Operation**

用法：必需。

选项：Scan | Install。

Scan

选择 Scan 选项时，系统使用 AWS-RunPatchBaseline 文档确定托管式节点的补丁合规性状态并向 Patch Manager 报告此信息。Scan 不提示要安装的更新或要重启的托管式节点。相反，该操作会标识哪些地方缺少已批准并且适用于节点的更新。

安装

选择 Install 选项时，AWS-RunPatchBaselineWithHooks 会尝试安装托管式节点中缺失并已批准的适用更新。在 Install 操作中生成的补丁合规性信息不会列出任何缺失的更新。但是，如果更新的安装因任何原因失败，它会报告处于失败状态的更新。一旦在托管式节点上安装了更新，就一定会重启节点，以确保正常安装和激活更新。（例外：如果将 AWS-RunPatchBaselineWithHooks 文档中的 RebootOption 参数设置为 NoReboot，则在 Patch Manager 运行后不会重启托管式节点。有关更多信息，请参阅 参数名称: RebootOption。）

> **Note**
>
> 如果在 Patch Manager 更新托管式节点之前安装了基准规则指定的补丁，则系统可能无法按预期重启。当补丁是由用户手动安装或由其他程序（例如 Ubuntu Server 上的 unattended-upgrades 程序包）自动安装时，可能会发生这种情况。

参数名称: **Snapshot ID**

用法：可选。

Snapshot ID 是 Patch Manager 使用的唯一 ID (GUID)，用于确保在单一操作中修补的一组托管式节点均具有完全相同的一组已批准补丁。尽管它定义为可选参数，根据是否在维护时段中运行 AWS-RunPatchBaselineWithHooks，我们还是提供了不同的最佳实践建议，如下表所述。

AWS-RunPatchBaselineWithHooks 最佳实践

Mode	最佳实践	详细信息
正在维护时段内运行 AWS-RunPatchBaselineWithHooks	不要提供快照 ID，Patch Manager 将为您提供。	如果使用维护时段运行 AWS-RunPatchBaselineWithHooks，则不应提供自己生成的快照 ID。这种情况下，Systems Manager 基于维护时段执行 ID 提供 GUID 值。这可确保在维护时段中为 AWS-RunPatchBaselineWithHooks 的所有调用使用正确的 ID。

Mode	最佳实践	详细信息
		如果在这种情况下您指定值，请注意，补丁基准的快照最多保留 3 天。之后，即使您在快照到期后指定相同的 ID，也将生成新的快照。
正在在维护时段以外运行 AWS-RunPatchBaselineWithHooks	为快照 ID[1] 生成和指定自定义 GUID 值。	如果您不使用维护时段运行 AWS-RunPatchBaselineWithHooks ，我们建议您为每个补丁基准生成并指定一个唯一的快照 ID，特别是于同一操作中在多个托管式节点上运行 AWS-RunPatchBaselineWithHooks 文档时。如果在这种情况下不指定 ID，Systems Manager 会为向其发送命令的每个托管式节点生成不同的快照 ID。这会导致在节点间指定不同的补丁集。 例如，假设您正在直接通过 Run Command（AWS Systems Manager 中的一项工具）运行 AWS-RunPatchBaselineWithHooks 文档，并以一组 50 个的托管式节点为目标。指定自定义快照 ID 将生成单一基准快照，用于评估和修补所有托管式节点，从而确保所有托管式节点最终处于一致状态。

Mode	最佳实践	详细信息

[1] 您可以使用任何能够生成 GUID 的工具为快照 ID 参数生成值。例如，在 PowerShell 中，可以使用 New-Guid cmdlet 生成格式为 12345699-9405-4f69-bc5e-9315aEXAMPLE 的 GUID。

参数名称: **RebootOption**

用法：可选。

选项：RebootIfNeeded | NoReboot

默认值：RebootIfNeeded

> **Warning**
>
> 默认选项是 RebootIfNeeded。务必为应用场景选择正确的选项。例如，如果您的托管式节点必须立即重启才能完成配置过程，请选择 RebootIfNeeded。或者，如果您需要在计划的重启时间之前保持托管式节点可用，请选择 NoReboot。

> **Important**
>
> 我们不建议使用 Patch Manager 在 Amazon EMR（原 Amazon Elastic MapReduce）中修补集群实例。特别是，不要为 RebootOption 参数选择 RebootIfNeeded 选项。（此选项在 SSM 命令文档中可用，用于修补 AWS-RunPatchBaseline、AWS-RunPatchBaselineAssociation 和 AWS-RunPatchBaselineWithHooks。）使用 Patch Manager 进行修补的底层命令使用 yum 和 dnf 命令。因此，由于软件包的安装方式，这些操作会导致不兼容。有关在 Amazon EMR 集群上更新软件的首选方法的信息，请参阅《Amazon EMR Management Guide》中的 Using the default AMI for Amazon EMR。

RebootIfNeeded

选择 RebootIfNeeded 选项时，托管式节点将在以下任一情况下重启：

- Patch Manager 已安装一个或多个补丁。

  Patch Manager 不评估补丁是否必需重启。即使补丁不需要重启，系统也会重启。

- Patch Manager 检测 `Install` 操作期间的一个或多个状态为 `INSTALLED_PENDING_REBOOT` 的补丁。

  `INSTALLED_PENDING_REBOOT` 状态可能表示在上次运行 `Install` 操作时选择了选项 `NoReboot`，或者是自上次重新启动托管式节点以来在 Patch Manager 外部安装了补丁。

在这两种情况下重启托管式节点，可确保从内存中刷新更新的软件包，并在所有操作系统中保持修补和重启行为一致。

NoReboot

选择 `NoReboot` 选项时，即使托管式节点在 `Install` 操作期间安装了补丁，Patch Manager 也不会重启托管式节点。如果您知道托管式节点在应用补丁后不需要重启，或者应用程序或进程在节点上运行，但不应因修补操作重启而中断，则此选项有用。当您希望更多地控制托管式节点的重启时间（例如，使用维护时段来控制）时，该选项也很有用。

> **Note**
>
> 如果选择 `NoReboot` 选项并安装了补丁，则会为该补丁分配状态 `InstalledPendingReboot`。但托管式节点本身将被标记为 `Non-Compliant`。重启并运行 `Scan` 操作后，节点状态将更新为 `Compliant`。

补丁安装跟踪文件：为跟踪补丁安装，特别是自上次系统重启以来安装的补丁，Systems Manager 会在托管式节点上维护一个文件。

> **Important**
>
> 请勿删除或修改该跟踪文件。如果删除或损坏此文件，托管式节点的补丁合规性报告则不准确。如果发生这种情况，请重启节点并运行补丁扫描操作以还原此文件。

此跟踪文件存储在托管式节点上的以下位置：

- Linux 操作系统：
  - `/var/log/amazon/ssm/patch-configuration/patch-states-configuration.json`
  - `/var/log/amazon/ssm/patch-configuration/patch-inventory-from-last-operation.json`
- Windows Server 操作系统:

- C:\ProgramData\Amazon\PatchBaselineOperations\State\PatchStatesConfiguration.json
- C:\ProgramData\Amazon\PatchBaselineOperations\State\PatchInventoryFromLastOperation.json

参数名称: **PreInstallHookDocName**

用法：可选。

默认值：AWS-Noop。

要提供的值 PreInstallHookDocName 参数是您选择的 SSM 文档的名称或 Amazon Resource Name (ARN)。您可以在中提供 AWS 托管文档的名称，或者您已创建或已与您共享的自定义 SSM 文档的名称或 ARN。（对于来自不同的 AWS 账户 的已与您共享的 SSM 文档，则必须指定完整的资源 ARN，例如 arn:aws:ssm:us-east-2:123456789012:document/MySharedDocument。）

指定的 SSM 文档在 Install 操作之前运行并执行 SSM Agent 支持的任何操作，例如，在托管式节点上执行修补之前检查应用程序运行状况检查的 shell 脚本。（有关操作列表，请参阅 命令文档插件参考）。默认 SSM 文档名称为 AWS-Noop，不会对托管式节点执行任何操作。

有关创建自定义 SSM 文档的信息，请参阅 创建 SSM 文档内容。

参数名称: **PostInstallHookDocName**

用法：可选。

默认值：AWS-Noop。

要提供的值 PostInstallHookDocName 参数是您选择的 SSM 文档的名称或 Amazon Resource Name (ARN)。您可以在中提供 AWS 托管文档的名称，或者您已创建或已与您共享的自定义 SSM 文档的名称或 ARN。（对于来自不同 AWS 账户 的已与您共享的 SSM 文档，必须指定完整资源 ARN，例如 arn:aws:ssm:us-east-2:123456789012:document/MySharedDocument。）

在 Install with NoReboot 操作之后运行您指定的 SSM 文档，并执行 SSM Agent 支持的任一操作，例如，用于在重启之前安装第三方更新的 shell 脚本。（有关操作列表，请参阅 命令文档插件参考）。默认 SSM 文档名称为 AWS-Noop，不会对托管式节点执行任何操作。

有关创建自定义 SSM 文档的信息，请参阅 创建 SSM 文档内容。

参数名称: **OnExitHookDocName**

用法：可选。

默认值：AWS-Noop。

要提供的值 OnExitHookDocName 参数是您选择的 SSM 文档的名称或 Amazon Resource Name (ARN)。您可以在中提供 AWS 托管文档的名称，或者您已创建或已与您共享的自定义 SSM 文档的名称或 ARN。（对于来自不同的 AWS 账户 的已与您共享的 SSM 文档，则必须指定完整的资源 ARN，例如 arn:aws:ssm:us-east-2:123456789012:document/MySharedDocument。）

指定的 SSM 文档在托管式节点重启操作后运行并执行 SSM Agent 支持的任何操作，例如，在修补操作完成后验证节点运行状况的 shell 脚本。（有关操作列表，请参阅 命令文档插件参考）。默认 SSM 文档名称为 AWS-Noop，不会对托管式节点执行任何操作。

有关创建自定义 SSM 文档的信息，请参阅 创建 SSM 文档内容。

在 **AWS-RunPatchBaseline** 或 **AWS-RunPatchBaselineAssociation** 中使用 InstallOverrideList 参数的示例场景

当您要在Patch Manager（AWS Systems Manager 中的一项工具）中覆盖由当前默认补丁基准指定的补丁时，可以使用 InstallOverrideList 参数。本主题提供示例说明如何使用此参数来实现以下目标：

- 将不同的补丁集应用于托管式节点目标组。
- 以不同频率应用这些补丁集。
- 对这两个操作使用相同的补丁基准。

假设您要在 Amazon Linux 2 托管式节点上安装两种不同类别的补丁。您希望使用维护时段按不同的计划安装这些补丁。您希望每周运行一个维护时段并安装所有 Security 补丁。您希望另一个维护时段每月运行一次，并安装所有可用的补丁或 Security 之外类别的补丁。

但是，一次只能将一个补丁基准定义为操作系统的默认值。此要求有助于避免一个补丁基准批准补丁而另一个补丁阻止补丁的情况，这种情况可能导致相互冲突的版本之间出现问题。

以下策略允许您使用 InstallOverrideList 参数按照不同的计划将不同类型的补丁应用到目标组，同时仍使用相同的补丁基准。

1. 在默认补丁基准中，确保仅指定 Security 更新。
2. 创建每周运行 AWS-RunPatchBaseline 或 AWS-RunPatchBaselineAssociation 的维护时段。不要指定覆盖列表。
3. 创建要每月应用的所有类型的补丁的覆盖列表，并将其存储在 Amazon Simple Storage Service (Amazon S3) 存储桶中。

4. 创建每月运行一次的第二个维护时段。但是，对于您为此维护时段注册的 Run Command 任务，请指定覆盖列表的位置。

结果：每周只安装在默认补丁基准中定义的 `Security` 补丁。每月都会安装所有可用的补丁或您定义的任何补丁子集。

有关更多信息和示例列表，请参阅 参数名称: `InstallOverrideList`。

使用基准覆盖参数

您可以在运行时使用Patch Manager（AWS Systems Manager 中的一项工具）中的基准覆盖功能来定义修补首选项。通过指定包含一个带有补丁基准列表的 JSON 对象的 Amazon Simple Storage Service (Amazon S3) 存储桶来实施。该修补操作使用与主机操作系统匹配的 JSON 对象中提供的基准，而不是应用默认补丁基准中的规则。

> **Important**
>
> `BaselineOverride` 文件名不能包含以下字符：反引号 (`)、单引号 (')、双引号 (") 和美元符号 ($)。

除非补丁操作使用补丁策略，否则使用 `BaselineOverride` 参数不会覆盖参数中提供的基准的补丁合规性。输出结果记录在来自 Run Command（AWS Systems Manager 中的一项工具）的标准输出日志中。结果仅打印标记为 `NON_COMPLIANT` 的软件包。这意味着软件包被标记为 `Missing`、`Failed`、 `InstalledRejected` 或者 `InstalledPendingReboot`。

但是，当补丁操作使用补丁策略时，系统会传递关联的 S3 存储桶中的覆盖参数，并更新托管式节点的合规性值。有关补丁策略的更多信息，请参阅 Quick Setup 中的补丁策略配置。

使用快照 ID 的补丁基准覆盖，或者安装覆盖列表参数

在两种情况下，补丁基准覆盖具有值得注意的行为。

同时使用基准覆盖和快照 ID

快照 ID 可确保特定修补命令中的所有托管式节点都应用相同的内容。例如，如果您一次修补 1000 个节点，则补丁相同。

当同时使用快照 ID 和补丁基准覆盖时，快照 ID 优先于补丁基准覆盖。仍将使用基准覆盖规则，但只会对其进行评估一次。在前面的示例中，1000 个托管式节点中的补丁仍始终相同。如果在修补操作的

中程，您将引用的 S3 存储桶中的 JSON 文件更改为不同的内容，应用的补丁仍将保持不变。这是因为提供了快照 ID。

同时使用基准覆盖和安装覆盖列表

您不能同时使用这两个参数。如果提供了两个参数，则修补文档将失败，并且不会在托管式节点上执行任何扫描或安装。

代码示例

以下 Python 代码示例说明了如何生成补丁基准覆盖。

```
import boto3
import json

ssm = boto3.client('ssm')
s3 = boto3.resource('s3')
s3_bucket_name = 'my-baseline-override-bucket'
s3_file_name = 'MyBaselineOverride.json'
baseline_ids_to_export = ['pb-0000000000000000', 'pb-0000000000000001']

baseline_overrides = []
for baseline_id in baseline_ids_to_export:
    baseline_overrides.append(ssm.get_patch_baseline(
        BaselineId=baseline_id
    ))

json_content = json.dumps(baseline_overrides, indent=4, sort_keys=True, default=str)
s3.Object(bucket_name=s3_bucket_name, key=s3_file_name).put(Body=json_content)
```

这将产生类似下面的补丁基准覆盖。

```
[
    {
        "ApprovalRules": {
            "PatchRules": [
                {
                    "ApproveAfterDays": 0,
                    "ComplianceLevel": "UNSPECIFIED",
                    "EnableNonSecurity": false,
                    "PatchFilterGroup": {
                        "PatchFilters": [
                            {
```

```
                            "Key": "PRODUCT",
                            "Values": [
                                "*"
                            ]
                        },
                        {
                            "Key": "CLASSIFICATION",
                            "Values": [
                                "*"
                            ]
                        },
                        {
                            "Key": "SEVERITY",
                            "Values": [
                                "*"
                            ]
                        }
                    ]
                }
            }
        ]
    },
    "ApprovedPatches": [],
    "ApprovedPatchesComplianceLevel": "UNSPECIFIED",
    "ApprovedPatchesEnableNonSecurity": false,
    "GlobalFilters": {
        "PatchFilters": []
    },
    "OperatingSystem": "AMAZON_LINUX_2",
    "RejectedPatches": [],
    "RejectedPatchesAction": "ALLOW_AS_DEPENDENCY",
    "Sources": []
},
{
    "ApprovalRules": {
        "PatchRules": [
            {
                "ApproveUntilDate": "2021-01-06",
                "ComplianceLevel": "UNSPECIFIED",
                "EnableNonSecurity": true,
                "PatchFilterGroup": {
                    "PatchFilters": [
                        {
                            "Key": "PRODUCT",
```

```
                                "Values": [
                                    "*"
                                ]
                            },
                            {
                                "Key": "CLASSIFICATION",
                                "Values": [
                                    "*"
                                ]
                            },
                            {
                                "Key": "SEVERITY",
                                "Values": [
                                    "*"
                                ]
                            }
                        ]
                    }
                }
            ]
        },
        "ApprovedPatches": [
            "open-ssl*"
        ],
        "ApprovedPatchesComplianceLevel": "UNSPECIFIED",
        "ApprovedPatchesEnableNonSecurity": false,
        "GlobalFilters": {
            "PatchFilters": []
        },
        "OperatingSystem": "CENTOS",
        "RejectedPatches": [
            "python*"
        ],
        "RejectedPatchesAction": "ALLOW_AS_DEPENDENCY",
        "Sources": []
    }
]
```

补丁基准

本节主题介绍当您在托管式节点上运行 `Scan` 或 `Install` 操作时，补丁基准在Patch Manager（AWS Systems Manager 中的一项工具）中的工作方式。

主题

- 预定义和自定义补丁基准
- 已批准补丁和已拒绝补丁列表的程序包名称格式
- 补丁组
- Microsoft 在 Windows Server 上发布的补丁应用程序

预定义和自定义补丁基准

Patch Manager（AWS Systems Manager 中的一项工具）为Patch Manager支持的每个操作系统提供预定义补丁基准。您可以按照当前配置的方式使用这些基准（无法自定义它们），也可以创建您自己的自定义补丁基准。自定义补丁基准允许您更好地控制在您的环境下批准或拒绝哪些补丁。此外，预定义基准会为使用这些基准安装的所有补丁分配 `Unspecified` 合规性级别。对于要分配的合规性值，您可以创建预定义基准的副本，并指定要分配给补丁的合规性值。有关更多信息，请参阅自定义基准和使用自定义补丁基准。

> **Note**
>
> 无论您使用哪种方法或类型的配置进行修补操作，本主题中的信息均适用：
>
> - Quick Setup 中配置的补丁策略
> - Quick Setup 中配置的主机管理选项
> - 运行补丁 `Scan` 或 `Install` 任务的维护时段
> - 按需 Patch now（立即修补）操作

主题

- 预定义基准
- 自定义基准

预定义基准

下表介绍了 Patch Manager 提供的预定义补丁基准。

有关 Patch Manager 支持各操作系统的哪些版本的信息，请参阅 Patch Manager 先决条件。

名称	支持的操作系统	详细信息
AWS-AlmaLinuxDefaultPatchBaseline	AlmaLinux	批准分类为“Security”且严重性等级为“关键”或“重要”的所有操作系统补丁。还批准分类为“Bugfix”的所有补丁。补丁将在发布或更新后 7 天自动批准。[1]
AWS-AmazonLinuxDefaultPatchBaseline	Amazon Linux 1	批准分类为“Security”且严重性等级为“关键”或“重要”的所有操作系统补丁。还将自动批准分类为“缺陷修正”的所有补丁。补丁将在发布或更新后 7 天自动批准。[1]
AWS-AmazonLinux2DefaultPatchBaseline	Amazon Linux 2	批准分类为“Security”且严重性等级为“关键”或“重要”的所有操作系统补丁。还批准分类为“缺陷修正”的所有补丁。补丁将在发布后 7 天自动批准。[1]
AWS-AmazonLinux2022DefaultPatchBaseline	Amazon Linux 2022	批准分类为“Security”且严重性等级为“关键”或“重要”的所有操作系统补丁。补丁将在发布后七天自动批准。在发布 7 天后还批准分类为“缺陷修正”的所有补丁。
AWS-AmazonLinux2023DefaultPatchBaseline	Amazon Linux 2023	批准分类为“Security”且严重性等级为“关键”或“重要”的所有操作系统补丁。补丁将在发布后七天自动批准。在发布 7 天后还批准分类为“缺陷修正”的所有补丁。

名称	支持的操作系统	详细信息
AWS-CentOSDefaultPatchBaseline	CentOS 和 CentOS Stream	在所有更新可用 7 天后批准这些更新（包括非安全性更新）。
AWS-DebianDefaultPatchBaseline	Debian Server	立即批准优先级为“Required”、“Important”、“Standard”、“Optional”或“Extra”的与操作系统安全相关的所有补丁。由于存储库中未提供可靠的发布日期，因此批准之前无需等待。
AWS-MacOSDefaultPatchBaseline	macOS	批准分类为“Security”的所有操作系统补丁。同时批准具有当前更新的所有软件包。
AWS-OracleLinuxDefaultPatchBaseline	Oracle Linux	批准分类为“Security”且严重性等级为“重要”或“中等”的所有操作系统补丁。在发布 7 天后还批准分类为“Bugfix”的所有补丁。补丁将在发布或更新后 7 天自动批准。[1]
AWS-DefaultRaspbianPatchBaseline	Raspberry Pi OS	立即批准优先级为“Required”、“Important”、“Standard”、“Optional”或“Extra”的与操作系统安全相关的所有补丁。由于存储库中未提供可靠的发布日期，因此批准之前无需等待。

名称	支持的操作系统	详细信息
AWS-RedHatDefaultPatchBaseline	Red Hat Enterprise Linux (RHEL)	批准分类为“Security”且严重性等级为“关键”或“重要”的所有操作系统补丁。还批准分类为“Bugfix”的所有补丁。补丁将在发布或更新后 7 天自动批准。[1]
AWS-RockyLinuxDefaultPatchBaseline	Rocky Linux	批准分类为“Security”且严重性等级为“关键”或“重要”的所有操作系统补丁。还批准分类为“Bugfix”的所有补丁。补丁将在发布或更新后 7 天自动批准。[1]
AWS-SuseDefaultPatchBaseline	SUSE Linux Enterprise Server (SLES)	批准分类为“Security”且严重性为“严重”或“重要”的所有操作系统补丁。补丁将在发布或更新后 7 天自动批准。[1]
AWS-UbuntuDefaultPatchBaseline	Ubuntu Server	立即批准优先级为“Required”、“Important”、“Standard”、“Optional”或“Extra”的与操作系统安全相关的所有补丁。由于存储库中未提供可靠的发布日期，因此批准之前无需等待。
AWS-DefaultPatchBaseline	Windows Server	批准分类为“CriticalUpdates”或“SecurityUpdates”且 MSRC 严重性为“严重”或“重要”的所有 Windows Server 操作系统补丁。补丁将在发布或更新后 7 天自动批准。[2]

名称	支持的操作系统	详细信息
AWS-WindowsPredefinedPatchBaseline-OS	Windows Server	批准分类为“CriticalUpdates”或“SecurityUpdates”且 MSRC 严重性为“严重”或“重要”的所有 Windows Server 操作系统补丁。补丁将在发布或更新后 7 天自动批准。[2]
AWS-WindowsPredefinedPatchBaseline-OS-Applications	Windows Server	对于 Windows Server 操作系统，批准分类为“CriticalUpdates”或“SecurityUpdates”且 MSRC 严重性为“严重”或“重要”的所有补丁。对于 Microsoft 发布的应用程序，批准所有补丁。在发布或更新后 7 天自动批准操作系统和应用程序的补丁。[2]

[1] 对于 Amazon Linux 1 和 Amazon Linux 2，补丁自动批准前的 7 天等待时间根据 `updateinfo.xml` 中的 `Updated Date` 值，而不是 `Release Date` 值计算。有多种因素会影响 `Updated Date` 值。其他操作系统处理发布和更新日期的方式有所不同。有关帮助您避免自动批准延迟的意外结果的信息，请参阅 如何计算软件包发布日期和更新日期。

[2] 对于 Windows Server，默认基准包括 7 天的自动批准延迟。要在发布后 7 天内安装补丁，您必须创建自定义基准。

自定义基准

使用以下信息来帮助您创建自定义补丁基准以符合修补目标。

主题

- 在自定义基准中使用自动批准
- 有关创建补丁基准的其他信息

在自定义基准中使用自动批准

如果您创建自己的补丁基准，则可使用以下类别选择自动批准哪些补丁。

- 操作系统：Windows Server、Amazon Linux、Ubuntu Server 等。
- 产品名称（对于操作系统）：例如，RHEL 6.5、Amazon Linux 2014.09、Windows Server 2012、Windows Server 2012 R2 等。
- 产品名称（仅对于 Windows Server 上 Microsoft 发布的应用程序）：例如，Word 2016、BizTalk 服务器等。
- 分类：例如，关键更新、安全更新等。
- 严重性：例如，关键、重要等。

对于您创建的每个批准规则，您可以选择指定自动批准延迟或指定补丁批准截止日期。

> **Note**
>
> 由于无法可靠地确定 Ubuntu Server 的更新程序包的发布日期，因此此操作系统不支持自动批准选项。

自动批准延迟是发布或最后更新补丁后到自动批准补丁用于修补前等待的天数。例如，如果您使用 `CriticalUpdates` 分类创建一条规则并将其自动批准延迟配置为 7 天，则将在 7 月 14 日自动批准 7 月 7 日发布的新关键补丁。

如果 Linux 存储库未提供软件包的发布日期信息，Patch Manager 会将软件包的构建时间用作符合 Amazon Linux 1、Amazon Linux 2、SUSE Linux Enterprise Server（SLES）、Red Hat Enterprise Linux（RHEL）和 CentOS 自动批准日期规范的日期。如果无法确定软件包的构建时间，Patch Manager 会使用默认日期 1970 年 1 月 1 日。这将导致 Patch Manager 绕过补丁基准（配置为批准 1970 年 1 月 1 日之后任一日期发布的补丁）中的所有自动批准日期规范。

当您指定自动批准截止日期时，Patch Manager 会自动应用在该日期或之前发布或最后更新的所有补丁。例如，如果将 2023 年 7 月 7 日指定为截止日期，系统都不会自动安装于 2023 年 7 月 8 日或之后发布或最后更新的所有补丁。

创建自定义补丁基准时，您可以为此补丁基准批准的补丁指定合规性严重性级别，例如 `Critical` 或 `High`。如果任何已批准补丁的补丁状态报告为 `Missing`，则补丁基准报告的总体合规性严重性级别就是您指定的严重级别。

有关创建补丁基准的其他信息

创建补丁基准时，请牢记以下信息：

- Patch Manager 为每个受支持的操作系统提供一个预定义的补丁基准。这些预定义的补丁基准将被用作每个操作系统类型的默认补丁基准，除非您创建自己的补丁基准，并将其指定为相应操作系统类型的默认补丁基准。

  > **Note**
  >
  > 适用于 Windows Server，提供三个预定义的补丁基准。补丁基准 `AWS-DefaultPatchBaseline` 和 `AWS-WindowsPredefinedPatchBaseline-OS` 仅支持 Windows 操作系统本身的操作系统更新。除非您指定了其他补丁基准，`AWS-DefaultPatchBaseline` 用作 Windows Server 托管式节点的默认补丁基准。这两个补丁基准中的配置设置是相同的。两者中较新的,`AWS-WindowsPredefinedPatchBaseline-OS` 被创建来区分它和 Windows Server 的第三个预定义补丁基准。补丁基准 `AWS-WindowsPredefinedPatchBaseline-OS-Applications`，可用于将补丁应用到 Windows Server 操作系统和 Microsoft 发布的受支持的应用程序。

- 默认情况下，Windows Server 2019 和 Windows Server 2022 会移除由较新更新所取代的更新。因此，如果您在 Windows Server 补丁基准中使用 `ApproveUntilDate` 参数，但 `ApproveUntilDate` 参数中选择的日期早于最新补丁发布的日期，则在运行修补操作时，不会安装新的补丁。有关 Windows Server 修补规则的更多信息，请参阅 如何选择安全性补丁 中的 Windows Server 选项卡。

  这意味着托管节点在 Systems Manager 操作方面是合规的，即使可能未安装上个月的关键补丁亦是如此。使用 `ApproveAfterDays` 参数时也可能出现同样的情况。由于 Microsoft 取代的补丁行为，可以设置一个数字（通常大于 30 天），这样，如果 Microsoft 的最新可用补丁是在经过 `ApproveAfterDays` 中的天数之前发布的，则绝不会安装 Windows Server 的补丁。

- 仅对于 Windows Server：未经补丁基准批准的可用安全更新补丁可以具有合规性值、`Compliant` 或 `Non-Compliant`，如自定义补丁基准中所定义。

  创建或更新补丁基准时，需选择要分配给可用但尚未批准的安全补丁（这些补丁由于不符合补丁基准中指定的安装标准而未获批准）的状态。例如，如果您指定在补丁发布后等待较长时间再进行安装，则会跳过某些您可能希望安装的安全补丁。如果在您指定的等待期内发布了该补丁的更新版本，则该补丁的安装等待期将重新开始。如果等待期过长，则可能会发布多个版本的补丁，但不会进行安装。

  通过控制台创建或更新补丁基准，需在可用安全更新合规性状态字段中指定此选项。通过 AWS CLI 运行 create-patch-baseline 或 update-patch-baseline 命令时，需在 `available-security-updates-compliance-status` 参数中指定此选项。

- 对于本地服务器和虚拟机 (VM)，Patch Manager 尝试使用您的自定义默认补丁基准。如果不存在自定义默认补丁基准，系统将使用相应操作系统的预定义补丁基准。
- 如果某个补丁在相同的补丁基准中同时被列为已批准和已拒绝，则该补丁将被拒绝。
- 一个托管式节点只能有一个定义的补丁基准。
- 可以添加到补丁基准的已批准补丁和已拒绝补丁列表中的软件包名称格式取决于您正在修补的操作系统的类型。

  有关已批准的补丁和已拒绝的补丁列表的已接受格式的信息，请参阅 已批准补丁和已拒绝补丁列表的程序包名称格式。
- 如果您在 Quick Setup 中使用补丁策略配置，则系统会每小时与 Quick Setup 同步一次您对自定义补丁基准所做的更新。

  如果删除了补丁策略中引用的自定义补丁基准，那么补丁策略的 Quick Setup Configuration details（配置详细信息）页面上会显示一个横幅。横幅将通知您，补丁策略引用的补丁基准已不存在，且后续的修补操作将失败。在这种情况下，返回 Quick Setup Configurations（配置）页面，选择 Patch Manager 配置，然后选择 Actions（操作），Edit configuration（编辑配置）。已删除的补丁基准名称将突出显示，您必须为受影响的操作系统选择新的补丁基准。

有关创建补丁基准的信息，请参阅 使用自定义补丁基准 和 教程：使用 AWS CLI 修补服务器环境。

已批准补丁和已拒绝补丁列表的程序包名称格式

可以添加到已批准补丁和已拒绝补丁列表中的软件包名称格式取决于您正在修补的操作系统的类型。

适用于 Linux 操作系统的软件包名称格式

您可以在补丁基准中为已批准和已拒绝补丁指定的格式因 Linux 类型而异。更具体地说，支持的格式取决于 Linux 操作系统类型使用的软件包管理器。

主题

- Amazon Linux 1、Amazon Linux 2、Amazon Linux 2022、Amazon Linux 2023、CentOS、Oracle Linux 和 Red Hat Enterprise Linux（RHEL）
- Debian Server、Raspberry Pi OS（原 Raspbian）和 Ubuntu Server
- SUSE Linux Enterprise Server (SLES)

Amazon Linux 1、Amazon Linux 2、Amazon Linux 2022、Amazon Linux 2023、CentOS、Oracle Linux 和 Red Hat Enterprise Linux（RHEL）

程序包管理器：YUM（Amazon Linux 2022、Amazon Linux 2023、RHEL 8 和 CentOS 8 除外，它们使用 DNF 作为程序包管理器）

已批准的补丁：对于已批准的补丁，您可以指定以下任一项：

- Bugzilla ID，格式为 1234567 (将只包含系统进程编号的字符串作为 Bugzilla ID。)
- CVE ID，格式为 CVE-2018-1234567
- 公告 ID，格式为 RHSA-2017:0864 和 ALAS-2018-123
- 使用一个或多个可用于软件包命名的组件构造的软件包名称。举例来说，对于名为 dbus.x86_64:1:1.12.28-1.amzn2023.0.1 的软件包，组件如下：
  - name: dbus
  - architecture: x86_64
  - epoch: 1
  - version: 1.12.28
  - release: 1.amzn2023.0.1

  支持具有以下构造的软件包名称：
  - name
  - name.arch
  - name-version
  - name-version-release
  - name-version-release.arch
  - version
  - version-release
  - epoch:version-release
  - name-epoch:version-release
  - name-epoch:version-release.arch
  - epoch:name-version-release.arch
  - name.arch:epoch:version-release

  一些示例：

  - dbus.x86_64
  - dbus-1.12.28
  - dbus-1.12.28-1.amzn2023.0.1
  - dbus-1:1.12.28-1.amzn2023.0.1.x86_64
- 我们还支持上述格式带有单个通配符的软件包名称组件，例如：
  - dbus*
  - dbus-1.12.2*
  - dbus-*:1.12.28-1.amzn2023.0.1.x86_64

已拒绝的补丁：对于已拒绝的补丁，您可以指定以下任一项：

- 使用一个或多个可用于软件包命名的组件构造的软件包名称。举例来说，对于名为 dbus.x86_64:1:1.12.28-1.amzn2023.0.1 的软件包，组件如下：
  - name: dbus
  - architecture; x86_64
  - epoch: 1
  - version: 1.12.28
  - release: 1.amzn2023.0.1

  支持具有以下构造的软件包名称：
  - name
  - name.arch
  - name-version
  - name-version-release
  - name-version-release.arch
  - version
  - version-release
  - epoch:version-release
  - name-epoch:version-release
  - name-epoch:version-release.arch
  - epoch:name-version-release.arch
  - name.arch:epoch:version-release

一些示例：

- dbus.x86_64
- dbus-1.12.28
- dbus-1.12.28-1.amzn2023.0.1
- dbus-1:1.12.28-1.amzn2023.0.1.x86_64

- 我们还支持上述格式带有单个通配符的软件包名称组件，例如：
  - dbus*
  - dbus-1.12.2*
  - dbus-*:1.12.28-1.amzn2023.0.1.x86_64

Debian Server、Raspberry Pi OS（原 Raspbian）和 Ubuntu Server

软件包管理器：APT

已批准的补丁和已拒绝的补丁：对于已批准和已拒绝的补丁，指定以下内容：

- 软件包名称，格式为 ExamplePkg33

> **Note**
>
> 对于 Debian Server 列表、Raspberry Pi OS 列表和 Ubuntu Server 列表，不包括架构或版本等元素。例如，您可以指定软件包名称 ExamplePkg33 以在补丁列表中包含所有以下项：
>
> - ExamplePkg33.x86.1
> - ExamplePkg33.x86.2
> - ExamplePkg33.x64.1
> - ExamplePkg33.3.2.5-364.noarch

SUSE Linux Enterprise Server (SLES)

软件包管理器：Zypper

已批准的补丁和已拒绝的补丁：对于已批准和已拒绝的补丁列表，可以指定以下任一项：

- 完整软件包名称，格式为：

  - `SUSE-SLE-Example-Package-12-2018-123`
  - `example-pkg-2018.11.4-46.17.1.x86_64.rpm`
- 包含单个通配符的软件包名称，例如：
  - `SUSE-SLE-Example-Package-12-2018-*`
  - `example-pkg-2018.11.4-46.17.1.*.rpm`

macOS适用的软件包名称格式

受支持的软件包管理器: softwareupdate、 installer、 Brew、 Brew Cask

已批准的补丁和已拒绝的补丁：对于已批准和已拒绝的补丁列表，可以指定完整的软件包名称，格式如下：

- `XProtectPlistConfigData`
- `MRTConfigData`

对于 macOS，在已批准补丁和已拒绝的补丁列表中不支持通配符。

适用于 Windows 操作系统的软件包名称格式

对于 Windows 操作系统，请使用 Microsoft 知识库 ID 和 Microsoft 安全公告 ID 指定补丁；例如：

```
KB2032276,KB2124261,MS10-048
```

补丁组

> **Note**
>
> 补丁组不会用于基于补丁策略的修补操作。有关使用补丁策略的更多信息，请参阅 Quick Setup 中的补丁策略配置。
> 对于在 2022 年 12 月 22 日发布补丁策略支持之前尚未使用补丁组的账户-区域对，控制台不支持补丁组功能。补丁组功能在此日期之前开始使用补丁组的账户区域对中仍然可用。

您可以使用补丁组，将托管式节点与Patch Manager（AWS Systems Manager 中的一项工具）中的特定补丁基准关联。补丁组根据关联的补丁基准规则，帮助确保您将合适的补丁部署到正确的节点集。另

外还可以帮助您避免过早地部署补丁（在对补丁进行充分测试之前）。例如，您可以为不同的环境（例如，开发环境、测试环境和生产环境）创建补丁组，并将每个补丁组注册到合适的补丁基准。

运行 AWS-RunPatchBaseline 或其他 SSM 命令文档进行修补时，您可以使用托管式节点 ID 或标签，将托管式节点设为目标。SSM Agent 和 Patch Manager 会基于您添加到托管式节点的补丁组值评估要使用的补丁基准。

通过标签定义补丁组

您可以使用应用于混合和多云环境中的 Amazon Elastic Compute Cloud（Amazon EC2）实例和非 EC2 节点来创建补丁组。通过标签定义补丁组时，请注意以下详细信息：

- 必须使用应用于托管式节点的标签键 Patch Group 或 PatchGroup 来定义补丁组。为补丁基准注册补丁组时，为这两个键指定的任何相同值都将解释为属于同一补丁组。例如，假设您使用以下第一个键值对标记了五个节点，并使用第二个键值对标记了五个节点：
  - key=PatchGroup,value=DEV
  - key=Patch Group,value=DEV

  用于创建补丁基准的 Patch Manager 命令会根据值 DEV，将这 10 个托管式节点合并成一个组。为补丁组创建补丁基准的命令的 AWS CLI 等效命令如下所示：

  ```
  aws ssm register-patch-baseline-for-patch-group \
      --baseline-id pb-0c10e65780EXAMPLE \
      --patch-group DEV
  ```

  将来自不同键的值合并到同一目标是该 Patch Manager 命令所独有的，旨在创建新的补丁组，其他 API 操作则不支持。例如，如果您使用具有相同值的 PatchGroup 和 Patch Group 键运行 send-command 操作，则您的目标将是两组完全不同的节点：

  ```
  aws ssm send-command \
      --document-name AWS-RunPatchBaseline \
      --targets "Key=tag:PatchGroup,Values=DEV"
  ```

  ```
  aws ssm send-command \
      --document-name AWS-RunPatchBaseline \
      --targets "Key=tag:Patch Group,Values=DEV"
  ```

- 基于标签的目标有限制。SendCommand 的每个目标数组最多可以包含五个键值对。

- 建议您只选择其中一个标签键约定，PatchGroup（不带空格）或 Patch Group（带空格）。但是，如果允许在实例上的 EC2 实例元数据中使用标签，则必须使用 PatchGroup。
- 请注意，此键区分大小写。您可以指定任何值来帮助您识别和定位该组中的资源，例如“web servers”或“US-EAST-PROD”，但键必须是 Patch Group 或 PatchGroup。

创建补丁组和标记托管式节点后，可以使用补丁基准注册补丁组。将补丁组注册到补丁基准可确保补丁组内的节点使用关联的补丁基准中定义的规则。

有关如何创建补丁组并将补丁组与补丁基准关联的更多信息，请参阅 创建和管理补丁组 和 将补丁组添加到补丁基准。

要查看使用 AWS Command Line Interface (AWS CLI) 创建补丁基准和补丁组的示例，请参阅 教程：使用 AWS CLI 修补服务器环境。有关使用 Amazon EC2 标签的信息，请参阅《Amazon EC2 用户指南》中的标记您的 Amazon EC2 资源。

工作方式

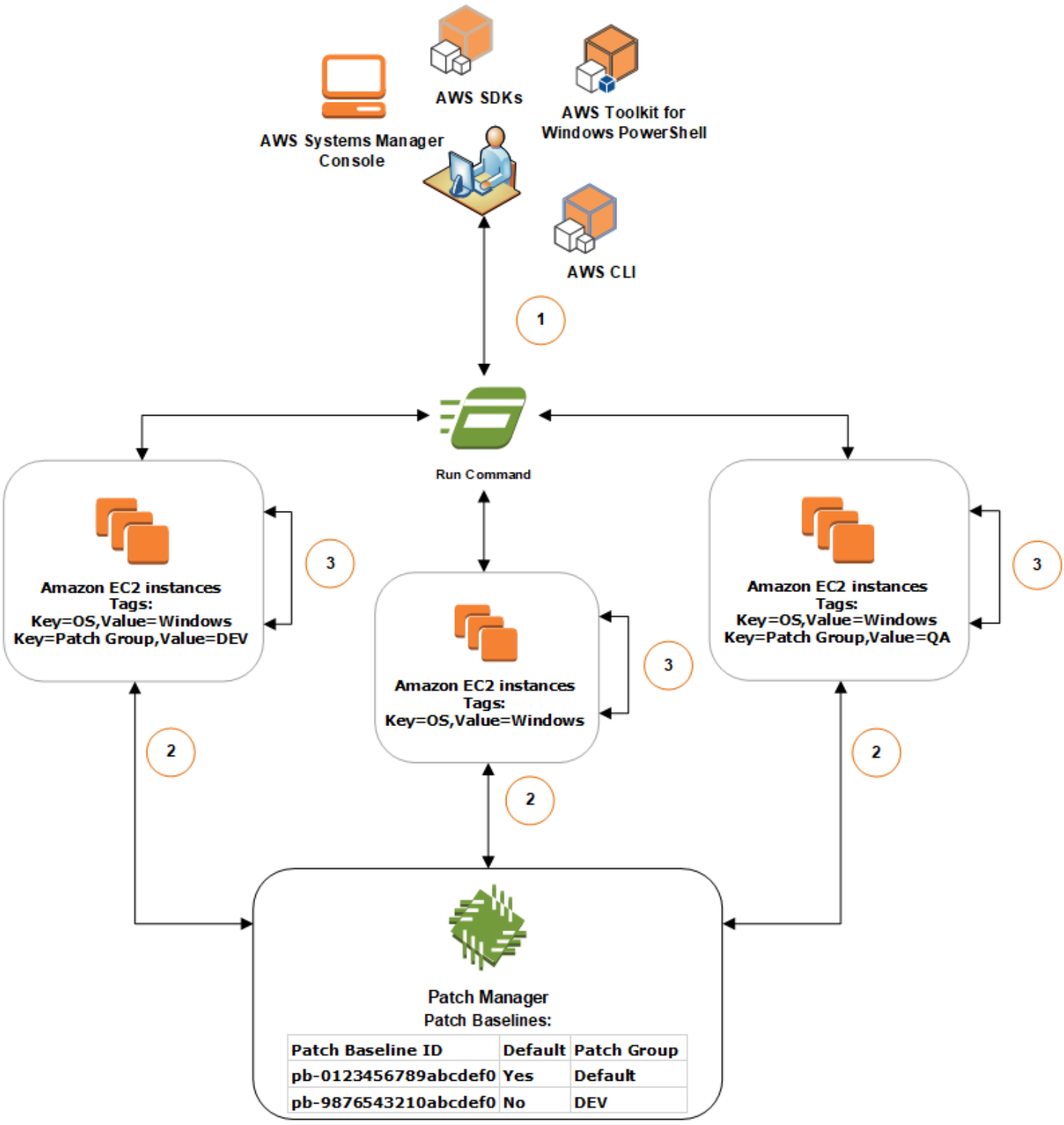
当系统运行将补丁基准应用于托管式节点的任务时，SSM Agent 会验证是否为该节点定义了补丁组值。如果该节点已分配给一个补丁组，Patch Manager 会验证哪个补丁基准注册到了该组。如果找到了该组的补丁基准，Patch Manager 将通知 SSM Agent 使用关联的补丁基准。如果没有为补丁组配置节点，Patch Manager 会自动通知 SSM Agent 使用当前配置的默认补丁基准。

> **Important**
>
> 一个托管式节点只能在一个补丁组中。
> 对每个操作系统类型，一个补丁组只能注册一个补丁基准。
> 如果在实例上启用 Allow tags in instance metadata（允许在实例元数据中使用标签）选项，则无法将 Patch Group 标签（带空格）应用于 Amazon EC2 实例。允许在实例元数据中使用标签会导致标签密钥名称不得包含空格。如果在 EC2 实例元数据中允许使用标签，则必须使用标签键 PatchGroup（不带空格）。

图 1：修补操作流程的一般示例

下图显示了 Systems Manager 在将 Run Command 任务发送到您的服务器实例集以使用 Patch Manager 进行修补时执行的流程的一般示例。这些流程决定了在修补操作中使用哪些补丁基准。（当维护时段配置为使用 Patch Manager 发送命令进行修补时，会使用类似的流程。）

整个流程如下图所示。



在本示例中，我们有三组 Windows Server EC2 实例，其应用了以下标签：

EC2 实例组	标签
组 1	key=OS,value=Windows key=PatchGroup,value=DEV
组 2	key=OS,value=Windows
组 3	key=OS,value=Windows key=PatchGroup,value=QA

在本示例中，我们还有这两个 Windows Server 补丁基准：

补丁基准 ID	默认	关联的补丁组
pb-0123456789abcdef0	是	Default
pb-9876543210abcdef0	否	DEV

使用 Run Command（AWS Systems Manager 中的一项工具）和Patch Manager扫描或安装补丁的一般流程，如下所示：

1. 发送修补命令：使用 Systems Manager 控制台、SDK、 AWS Command Line Interface (AWS CLI) 或 AWS Tools for Windows PowerShell 使用文档 AWS-RunPatchBaseline 发送 Run Command 任务。该图显示了将标签 key=OS,value=Windows 指定为目标执行修补托管实例的 Run Command 任务。
2. 补丁基准确定：SSM Agent 验证应用到 EC2 实例的补丁组标签，并查询 Patch Manager 查找相应的补丁基准。
   - 与补丁基准关联的匹配的补丁组值：
     1. 在组 1 中的 EC2 实例上安装的 SSM Agent 收到步骤 1 中发出的命令后开始修补操作。SSM Agent 验证 EC2 实例是否应用了补丁组标签值 DEV 并查询 Patch Manager 以查找关联的补丁基准。
     2. Patch Manager 验证补丁基准 pb-9876543210abcdef0 是否关联了补丁组 DEV 并通知 SSM Agent。

3. SSM Agent 根据在 pb-9876543210abcdef0 中配置的批准规则和例外从 Patch Manager 检索补丁基准快照，然后继续执行下一步。

- 未将补丁组标签添加到实例：

1. 在组 2 中的 EC2 实例上安装的 SSM Agent 收到步骤 1 中发出的命令后开始修补操作。SSM Agent 验证 EC2 实例是否未应用 Patch Group 或 PatchGroup 标签，并因此 SSM Agent 查询 Patch Manager 以查找默认的 Windows 补丁基准。
2. Patch Manager 验证默认的 Windows Server 补丁基准是否是 pb-0123456789abcdef0 并通知 SSM Agent。
3. SSM Agent 根据在默认的补丁基准 pb-0123456789abcdef0 中配置的批准规则和例外从 Patch Manager 检索补丁基准快照，然后继续执行下一步。

- 没有与补丁基准关联的匹配的补丁组值：

1. 在组 3 中的 EC2 实例上安装的 SSM Agent 收到步骤 1 中发出的命令后开始修补操作。SSM Agent 验证 EC2 实例是否应用了补丁组标签值 QA 并查询 Patch Manager 以查找关联的补丁基准。
2. Patch Manager 没有找到关联了补丁组 QA 的补丁基准。
3. Patch Manager 通知 SSM Agent 使用默认的 Windows 补丁基准 pb-0123456789abcdef0。
4. SSM Agent 根据在默认的补丁基准 pb-0123456789abcdef0 中配置的批准规则和例外从 Patch Manager 检索补丁基准快照，然后继续执行下一步。

3. 补丁扫描或安装：确定要使用的合适的补丁基准后，SSM Agent 将根据步骤 1 中指定的操作值开始扫描或安装补丁。扫描或安装的补丁由在 Patch Manager 提供的补丁基准快照中定义的批准规则和补丁例外确定。

更多信息

- 补丁合规性状态值

Microsoft 在 Windows Server 上发布的补丁应用程序

使用本主题中的信息可帮助您准备好使用Patch Manager（AWS Systems Manager 中的一项工具）在 Windows Server 上修补应用程序。

Microsoft 应用程序修补

对 Windows Server 托管式节点上应用程序的修补支持，仅限于 Microsoft 发布的应用程序。

> **Note**
>
> 在某些情况下，Microsoft 会为未指定更新日期和时间的应用程序发布补丁。在这些情况下，系统会默认提供 01/01/1970 的更新日期和时间。

修补 Microsoft 发布的应用程序的补丁基准

适用于 Windows Server，提供三个预定义的补丁基准。补丁基准 AWS-DefaultPatchBaseline 和 AWS-WindowsPredefinedPatchBaseline-OS 仅支持 Windows 操作系统本身的操作系统更新。除非您指定了其他补丁基准，AWS-DefaultPatchBaseline 用作 Windows Server 托管式节点的默认补丁基准。这两个补丁基准中的配置设置是相同的。两者中较新的,AWS-WindowsPredefinedPatchBaseline-OS 被创建来区分它和 Windows Server 的第三个预定义补丁基准。补丁基准 AWS-WindowsPredefinedPatchBaseline-OS-Applications，可用于将补丁应用到 Windows Server 操作系统和 Microsoft 发布的受支持的应用程序。

您也可以创建自定义补丁基准，用来在 Windows Server 计算机上更新 Microsoft 发布的应用程序。

支持在本地服务器、边缘设备、VM 和其他非 EC2 节点上修补由 Microsoft 发布的应用程序

要在虚拟机（VM）和其他非 EC2 托管式节点上修补 Microsoft 发布的应用程序，必须打开高级实例套餐。使用高级实例套餐需支付费用。但是，修补 Microsoft 在 Amazon Elastic Compute Cloud (Amazon EC2) 实例上发布的应用程序不收取额外费用。有关更多信息，请参阅 配置实例套餐。

“其他微软产品”的 Windows 更新选项

为了让 Patch Manager 能够在 Windows Server 托管式节点上修补 Microsoft 发布的应用程序，必须在托管式节点上启用 Windows 更新选项 Give me updates for other Microsoft products when I update Windows（更新 Windows 时向我提供其他 Microsoft 产品的更新）。

有关在单个托管式节点上允许此选项的信息，请参阅 Microsoft 支持网站上的使用 Microsoft 更新更新 Office。

对于运行 Windows Server 2016 及更高版本的托管式节点机群，您可以使用组策略对象 (GPO) 打开设置。在组策略管理编辑器中，转到计算机配置、管理模板、Windows 组件、Windows 更新，然后选择安装其他 Microsoft 产品的更新。我们还建议使用其他参数配置 GPO，以防止在 Patch Manager 之外意外自动更新和重启。有关更多信息，请参阅 Microsoft 技术文档网站上的 Configuring Automatic Updates in a Non-Active Directory Environment（在非 Active Directory 环境中配置自动更新）。

对于运行 Windows Server 2012 或 2012 R2 的托管式节点机群，您可以使用脚本打开该选项，如 Microsoft 文档博客网站上的通过脚本在 Windows 7 中启用和禁用 Microsoft 更新中所述。例如，可以：

1. 将脚本从博客文章保存在文件中。
2. 将文件上传到 Amazon Simple Storage Service (Amazon S3) 存储桶或其他可访问的位置。
3. 使用 Run Command（AWS Systems Manager 中的一项工具），通过 Systems Manager 文档（SSM 文档）AWS-RunPowerShellScript 在托管式节点上运行脚本，所使用的命令类似于以下内容。

```
Invoke-WebRequest `
    -Uri "https://s3.aws-api-domain/amzn-s3-demo-bucket/script.vbs" `
    -Outfile "C:\script.vbs" cscript c:\script.vbs
```

最低参数要求

要在自定义补丁基准中包括 Microsoft 发布的应用程序，您必须至少指定要修补的产品。下面的 AWS Command Line Interface (AWS CLI) 命令演示了修补产品（如 Microsoft Office 2016）的最低要求：

Linux & macOS

```
aws ssm create-patch-baseline \
    --name "My-Windows-App-Baseline" \
    --approval-rules
 "PatchRules=[{PatchFilterGroup={PatchFilters=[{Key=PRODUCT,Values='Office 2016'},
{Key=PATCH_SET,Values='APPLICATION'}]},ApproveAfterDays=5}]"
```

Windows Server

```
aws ssm create-patch-baseline ^
    --name "My-Windows-App-Baseline" ^
    --approval-rules
 "PatchRules=[{PatchFilterGroup={PatchFilters=[{Key=PRODUCT,Values='Office 2016'},
{Key=PATCH_SET,Values='APPLICATION'}]},ApproveAfterDays=5}]"
```

如果您指定了 Microsoft 应用程序产品系列，您指定的每个产品都必须是所选产品系列的受支持的成员产品。例如，要修补产品“Active Directory Rights Management Services Client 2.0”，您必须将其产品

系列指定为“Active Directory”，而不是“Office”或“SQL Server”（举例）。下面的 AWS CLI 命令演示了一对匹配的产品系列和产品。

Linux & macOS

```
aws ssm create-patch-baseline \
    --name "My-Windows-App-Baseline" \
    --approval-rules
 "PatchRules=[{PatchFilterGroup={PatchFilters=[{Key=PRODUCT_FAMILY,Values='Active
 Directory'},{Key=PRODUCT,Values='Active Directory Rights Management Services Client
 2.0'},{Key=PATCH_SET,Values='APPLICATION'}]},ApproveAfterDays=5}]"
```

Windows Server

```
aws ssm create-patch-baseline ^
    --name "My-Windows-App-Baseline" ^
    --approval-rules
 "PatchRules=[{PatchFilterGroup={PatchFilters=[{Key=PRODUCT_FAMILY,Values='Active
 Directory'},{Key=PRODUCT,Values='Active Directory Rights Management Services Client
 2.0'},{Key=PATCH_SET,Values='APPLICATION'}]},ApproveAfterDays=5}]"
```

> **Note**
>
> 如果收到有关产品和系列配对不匹配的错误消息，请参阅 问题：产品系列/产品对不匹配 以获取帮助解决问题。

在 Amazon Linux 2 托管式节点上使用 Kernel Live Patching

适用于 Amazon Linux 2 的 Kernel Live Patching 使您能够将安全漏洞和严重错误补丁应用于正在运行的 Linux 内核，而无需重启或中断正在运行的应用程序。这让您能够从改进的服务和应用程序可用性中受益，同时保持基础设施的安全和最新状态。在运行 Amazon Linux 2 的 Amazon EC2 实例、Kernel Live Patching 核心设备和本地虚拟机上支持 AWS IoT Greengrass。

有关 Kernel Live Patching 的一般信息，请参阅 Amazon Linux 2 User Guide 中的 Kernel Live Patching on AL2。

在 Amazon Linux 2 托管式节点上启用 Kernel Live Patching 之后，您可使用Patch Manager（AWS Systems Manager 中的一项工具）将内核实时补丁应用到托管式节点。使用 Patch Manager 是使用节点上的现有 yum 工作流来应用更新的替代方法。

开始前的准备工作

要使用 Patch Manager 将内核实时补丁应用到 Amazon Linux 2 托管式节点上，请确保节点基于正确的架构和内核版本。要了解有关信息，请参阅《Amazon EC2 用户指南》中的支持的配置和先决条件。

主题

- 使用 Patch Manager 的 Kernel Live Patching
- 使用 Patch Manager 的 Kernel Live Patching 的原理
- 使用 Run Command 来启用 Kernel Live Patching
- 使用 Run Command 应用内核实时补丁
- 使用 Run Command 来关闭 Kernel Live Patching

使用 Patch Manager 的 Kernel Live Patching

更新内核版本

应用内核实时补丁更新后，无需重启托管式节点。但是，AWS 为 Amazon Linux 2 内核版本提供内核实时补丁，最长可达其发布后三个月。在三个月期限之后，您必须更新到更高版本的内核才能继续接收内核实时补丁。我们建议使用维护时段至少每三个月一次安排重启节点，以提示内核版本更新。

卸载内核实时补丁

无法使用 Patch Manager 卸载内核实时补丁。相反，您可以关闭 Kernel Live Patching，从而删除所应用的内核实时补丁的 RPM 程序包。有关更多信息，请参阅 使用 Run Command 来关闭 Kernel Live Patching。

内核合规性

在某些情况下，从当前内核版本的实时补丁安装所有 CVE 补丁可能会使该内核进入与较新内核版本相同的合规性状态。发生这种情况时，较新版本报告为 `Installed`，而托管式节点报告为 `Compliant`。但是，对于较新的内核版本，不会报告安装时间。

一个内核实时补丁，多个 CVE

如果一个内核实时补丁解决多个 CVE，并且这些 CVE 具有各种分类和严重性值，则只针对该补丁报告 CVE 中的最高分类和严重性。

本节的其余部分介绍如何使用 Patch Manager 将内核实时补丁应用于满足这些要求的托管式节点。

使用 Patch Manager 的 Kernel Live Patching 的原理

AWS 为 Amazon Linux 2 发布了两种类型的内核实时补丁：安全更新和错误修复。要应用这些补丁类型，请使用仅针对下表列出的分类和严重性的补丁基准文档。

分类	严重性
Security	Critical, Important
Bugfix	All

您可以创建仅针对这些补丁的自定义补丁基准，也可以使用预定义的 AWS-AmazonLinux2DefaultPatchBaseline 补丁基准。换句话说，您可以将 AWS-AmazonLinux2DefaultPatchBaseline 与启用 Kernel Live Patching 的 Amazon Linux 2 托管式节点一起使用，并且系统会应用内核实时更新。

> **Note**
>
> AWS-AmazonLinux2DefaultPatchBaseline 配置指定发布最后更新补丁后的 7 天等待期，然后才会自动安装补丁。如果您不想等待 7 天再自动批准内核实时补丁，则可以创建并使用自定义补丁基准。在补丁基准中，您可以不指定自动批准等待期，也可以指定较短或更长的等待期。有关更多信息，请参阅 使用自定义补丁基准。

我们建议使用以下策略通过内核实时更新来修补托管式节点：

1. 在 Amazon Linux 2 托管式节点上启用 Kernel Live Patching。
2. 使用 Run Command（AWS Systems Manager 中的一项工具）在托管式节点上运行 Scan 操作，这些节点使用预定义 AWS-AmazonLinux2DefaultPatchBaseline 或自定义补丁基准，该基准也仅针对严重性分类为 Critical 和 Important，以及 All 的严重性为 Bugfix 的 Security 更新。
3. 使用 Compliance（AWS Systems Manager 中的一项工具）查看是否报告了任何被扫描托管式节点中存在不符合修补合规性的情况。如果是这样，请查看节点合规性详细信息，以确定托管式节点中是否缺少任何内核实时补丁。
4. 要安装缺少的内核实时补丁，请将 Run Command 与之前指定的相同补丁基准一起使用，但这次运行 Install 操作而不是 Scan 操作。

由于无需重启即可安装内核实时补丁，因此您可以为此操作选择 NoReboot 重启选项。

> **Note**
>
> 如果托管式节点上安装的其他类型补丁需要，或者要更新到较新的内核，您仍然可以重启托管式节点。在这些情况下，请改为选择 RebootIfNeeded 重启选项。

5. 返回到合规性以验证安装了内核实时补丁。

使用 Run Command 来启用 Kernel Live Patching

要启用 Kernel Live Patching，您可以在托管式节点上运行 yum 命令，或使用 Run Command 以及您创建的自定义 Systems Manager 文档（SSM 文档）。

有关通过直接在托管式节点上运行 yum 命令来打开 Kernel Live Patching 的信息，请参阅《Amazon EC2 用户指南》中的启用 Kernel Live Patching。

> **Note**
>
> 当您启用内核实时修补时，如果已在托管式节点上运行的内核为比 kernel-4.14.165-131.185.amzn2.x86_64（支持的最低版本）更早，进程将安装最新的可用内核版本并重启托管式节点。如果节点已运行 kernel-4.14.165-131.185.amzn2.x86_64 或更高版本，进程不会安装更新的版本，也不会重启节点。

使用 Run Command 来启用 Kernel Live Patching (控制台)

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Run Command。
3. 选择 Run command（运行命令）。
4. 在命令文档列表中，选择自定义 SSM 文档 AWS-ConfigureKernelLivePatching。
5. 在 Command parameters（命令参数）部分中，指定是否希望作为此操作的一部分重启托管式节点。
6. 有关使用此页上的其余控件的信息，请参阅 从控制台运行命令。
7. 选择运行。

启用 Kernel Live Patching (AWS CLI)

- 在本地计算机上运行以下命令。

  Linux & macOS

  ```
  aws ssm send-command \
      --document-name "AWS-ConfigureKernelLivePatching" \
      --parameters "EnableOrDisable=Enable" \
      --targets "Key=instanceids,Values=instance-id"
  ```

  Windows Server

  ```
  aws ssm send-command ^
      --document-name "AWS-ConfigureKernelLivePatching" ^
      --parameters "EnableOrDisable=Enable" ^
      --targets "Key=instanceids,Values=instance-id"
  ```

  将 *instance-id* 替换为要启用该功能的 Amazon Linux 2 托管式节点的 ID，例如 i-02573cafcfEXAMPLE。您可以使用以下任一格式，在多个托管式节点上启用该功能。

  - `--targets "Key=instanceids,Values=instance-id1,instance-id2"`
  - `--targets "Key=tag:tag-key,Values=tag-value"`

  有关可以在命令中使用的其他选项的信息，请参阅《AWS CLI Command Reference》中的 send-command。

使用 Run Command 应用内核实时补丁

要应用内核实时补丁，您可以在托管式节点上运行 yum 命令，也可以使用 Run Command 和 SSM 文档 AWS-RunPatchBaseline。

有关通过在托管式节点上直接运行 yum 命令来应用内核实时补丁的信息，请参阅《Amazon EC2 用户指南》中的应用内核实时补丁。

使用 Run Command 应用内核实时补丁（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。

2. 在导航窗格中，选择 Run Command。
3. 选择 Run command（运行命令）。
4. 在命令文档列表中，请选择 SSM 文档 AWS-RunPatchBaseline。
5. 在 Command parameters（命令参数）部分，执行以下操作之一：
   - 如果要检查是否有新的内核实时补丁，请对于 Operation (操作) 选择 Scan。对于 Reboot Option（重启选项），如果不希望托管式节点在此操作后重启，请选择 NoReboot。操作完成后，您可以在合规性中检查新的补丁和合规性状态。
   - 如果您已经检查了补丁合规性，并准备好应用可用的内核实时补丁，请对于 Operation (操作) 选择 Install。对于 Reboot Option（重启选项），如果不希望托管式节点在此操作后重启，请选择 NoReboot。
6. 有关使用此页上的其余控件的信息，请参阅 从控制台运行命令。
7. 选择运行。

使用 Run Command (AWS CLI) 应用内核实时补丁

1. 要在合规性检查结果出来之前执行 Scan 操作，请从本地计算机运行以下命令。

   Linux & macOS

   ```
   aws ssm send-command \
       --document-name "AWS-RunPatchBaseline" \
       --targets "Key=InstanceIds,Values=instance-id" \
       --parameters '{"Operation":["Scan"],"RebootOption":["RebootIfNeeded"]}'
   ```

   Windows Server

   ```
   aws ssm send-command ^
       --document-name "AWS-RunPatchBaseline" ^
       --targets "Key=InstanceIds,Values=instance-id" ^
       --parameters {\"Operation\":[\"Scan\"],\"RebootOption\":[\"RebootIfNeeded
   \"]}
   ```

   有关可以在命令中使用的其他选项的信息，请参阅《AWS CLI Command Reference》中的 send-command。
2. 要在合规性检查结果出来之后执行 Install 操作，请从本地计算机运行以下命令。

Linux & macOS

```
aws ssm send-command \
    --document-name "AWS-RunPatchBaseline" \
    --targets "Key=InstanceIds,Values=instance-id" \
    --parameters '{"Operation":["Install"],"RebootOption":["NoReboot"]}'
```

Windows Server

```
aws ssm send-command ^
    --document-name "AWS-RunPatchBaseline" ^
    --targets "Key=InstanceIds,Values=instance-id" ^
    --parameters {\"Operation\":[\"Install\"],\"RebootOption\":[\"NoReboot\"]}
```

在上述两个命令中，将 *instance-id* 替换为要应用内核实时补丁的 Amazon Linux 2 托管式节点的 ID，例如 i-02573cafcfEXAMPLE。您可以使用以下任一格式，在多个托管式节点上启用该功能。

- `--targets "Key=instanceids,Values=instance-id1,instance-id2"`
- `--targets "Key=tag:tag-key,Values=tag-value"`

有关可以在这些命令中使用的其他选项的信息，请参阅《AWS CLI Command Reference》中的 send-command。

使用 Run Command 来关闭 Kernel Live Patching

要关闭 Kernel Live Patching，您可以在托管式节点上运行 yum 命令，也可以使用 Run Command 以及自定义 SSM 文档 `AWS-ConfigureKernelLivePatching`。

> **Note**
>
> 如果您不再需要使用内核实时修补，可以随时关闭它。在大多数情况下，不需要关闭该功能。

有关通过直接在托管式节点上运行 yum 命令来关闭 Kernel Live Patching 的信息，请参阅《Amazon EC2 用户指南》中的启用 Kernel Live Patching。

> **Note**
>
> 关闭 Kernel Live Patching 时，该进程会卸载 Kernel Live Patching 插件，然后重启托管式节点。

使用 Run Command 来关闭 Kernel Live Patching (控制台)

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Run Command。
3. 选择 Run command（运行命令）。
4. 在命令文档 列表中，请选择 SSM 文档 `AWS-ConfigureKernelLivePatching`。
5. 在 Command parameters (命令参数) 部分中，为必需的参数指定值。
6. 有关使用此页上的其余控件的信息，请参阅 从控制台运行命令。
7. 选择运行。

关闭 Kernel Live Patching (AWS CLI)

- 运行类似于下面的命令。

  Linux & macOS

  ```
  aws ssm send-command \
      --document-name "AWS-ConfigureKernelLivePatching" \
      --targets "Key=instanceIds,Values=instance-id" \
      --parameters "EnableOrDisable=Disable"
  ```

  Windows Server

  ```
  aws ssm send-command ^
      --document-name "AWS-ConfigureKernelLivePatching" ^
      --targets "Key=instanceIds,Values=instance-id" ^
      --parameters "EnableOrDisable=Disable"
  ```

  将 *instance-id* 替换为要关闭该功能的 Amazon Linux 2 托管式节点的 ID，例如 i-02573cafcfEXAMPLE。要在多个托管式节点上关闭该功能，您可以使用以下任一格式。

- `--targets "Key=instanceids,Values=instance-id1,instance-id2"`
- `--targets "Key=tag:tag-key,Values=tag-value"`

有关可以在命令中使用的其他选项的信息，请参阅《AWS CLI Command Reference》中的 send-command。

使用控制台处理 Patch Manager 资源和合规性

要使用Patch Manager（AWS Systems Manager 中的一项工具），请完成以下任务。此部分更详细地说明了这些任务。

1. 验证您使用的每个操作系统类型的 AWS 预定义的补丁基准是否符合您的需求。如果不符合，则为该托管式节点类型创建定义一组标准补丁的补丁基准，并将其设置为默认值。
2. 使用 Amazon Elastic Compute Cloud (Amazon EC2) 标签将托管节点整理到补丁组中（可选，但建议执行）。
3. 请执行以下操作之一：
   - （推荐）在Quick Setup（Systems Manager 中的一项工具）中配置补丁策略让您可以按计划为整个组织、部分组织单位或单个 AWS 账户安装缺失的补丁。有关更多信息，请参阅 使用 Quick Setup 补丁策略为组织中的实例配置修补。
   - 在 Run Command 任务类型中，创建使用 Systems Manager 文档（SSM 文档）`AWS-RunPatchBaseline` 的维护时段。有关更多信息，请参阅 教程：使用控制台创建修补的维护时段。
   - 在 Run Command 操作中，手动运行 `AWS-RunPatchBaseline`。有关更多信息，请参阅 从控制台运行命令。
   - 使用 Patch now（立即修补）功能按需手动修补节点。有关更多信息，请参阅 按需修补托管式节点。
4. 监视修补以验证合规性和调查故障。

主题

- 创建补丁策略
- 查看补丁程序控制面板摘要
- 使用补丁合规性报告
- 按需修补托管式节点

- 使用补丁基准
- 查看可用的补丁
- 创建和管理补丁组
- 将 Patch Manager 与 AWS Security Hub 集成

创建补丁策略

补丁策略是您使用Quick Setup（AWS Systems Manager 中的一项工具）设置的一项配置。与其他配置修补的方法相比，补丁策略可以对补丁操作进行更广泛、更集中的控制。补丁策略定义了自动修补节点和应用程序时使用的计划和基准。

有关更多信息，请参阅以下主题：

- Quick Setup 中的补丁策略配置
- 使用 Quick Setup 补丁策略为组织中的实例配置修补

查看补丁程序控制面板摘要

Patch Manager中的控制面板选项卡在控制台中提供一个摘要视图，可用于在统一视图中监控修补操作。Patch Manager是 AWS Systems Manager 中的一项工具。在控制面板选项卡上，您可以查看以下内容：

- 有多少托管式节点符合和不符合修补规则的快照。
- 托管式节点补丁合规性结果的时间快照。
- 对于每种最常见的不合规原因，有多少不合规的托管式节点链接计数。
- 最近修补操作的链接列表。
- 已设置的定期修补任务的链接列表。

查看补丁控制面板摘要

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Patch Manager。
3. 选择控制面板选项卡。
4. 滚动到包含要查看的摘要数据的部分：

- Amazon EC2 实例管理
- 合规性摘要
- 不合规问题计数
- 合规性报告
- 基于非补丁策略的操作
- 基于非补丁策略的重复任务

使用补丁合规性报告

使用以下主题中的信息帮助您Patch Manager（AWS Systems Manager 中的一项工具）中生成和使用补丁合规性报告。

无论您使用哪种配置方法或类型进行修补操作，以下主题中的信息都适用：

- Quick Setup 中配置的补丁策略
- Quick Setup 中配置的主机管理选项
- 运行补丁 `Scan` 或 `Install` 任务的维护时段
- 按需 Patch now（立即修补）操作

> **Important**
>
> 如果您用多种操作来扫描实例的补丁合规性，请注意每次扫描都会覆盖先前扫描的补丁合规性数据。作为结果，您最终可能会导致补丁合规性数据中产生意外结果。有关更多信息，请参阅避免意外覆盖补丁合规性数据。
> 要验证使用哪个补丁基准来生成最新的合规性信息，请导航到 Patch Manager 中的合规性报告选项卡，找到要了解的托管节点行，然后在使用的基准 ID 列中选择基准 ID。

主题

- 查看补丁合规性结果
- 生成 .csv 补丁合规性报告
- 使用 Patch Manager 修复不合规的托管式节点
- 避免意外覆盖补丁合规性数据

查看补丁合规性结果

使用这些过程可查看有关托管式节点的补丁合规性信息。

此过程适用于使用 AWS-RunPatchBaseline 文档的补丁操作。查看有关使用 AWS-RunPatchBaselineAssociation 文档进行补丁操作的补丁合规性信息的信息，请参阅 标识不合规的托管式节点。

> **Note**
>
> Quick Setup和 Explorer 的补丁扫描操作使用 AWS-RunPatchBaselineAssociation 文档。Quick Setup和 Explorer 都是 AWS Systems Manager 中的工具。

确定针对特定 CVE 问题的补丁解决方案 (Linux)

对于许多基于 Linux 的操作系统，补丁合规性结果表明哪些常见漏洞和暴露 (CVE) 公告问题是通过哪些补丁来解决的。此信息可帮助您确定需要安装丢失或失败的补丁的紧迫程度。

受支持的以下操作系统的类型版本中包含 CVE 详细信息：

- AlmaLinux
- Amazon Linux 1
- Amazon Linux 2
- Amazon Linux 2022
- Amazon Linux 2023
- Oracle Linux
- Red Hat Enterprise Linux (RHEL)
- Rocky Linux
- SUSE Linux Enterprise Server (SLES)

> **Note**
>
> 默认情况下，CentOS 和 CentOS Stream 不提供有关更新的 CVE 信息。但是，您可以使用 Fedora 发布的 Extra Packages for Enterprise Linux (EPEL) 存储库等第三方存储库来允许这种支持。想要了解有关信息，请参阅 Fedora 维基词条中的 EPEL。
> 目前，仅报告状态为 Missing 或 Failed 的补丁的 CVE ID 值。

您还可以根据情况和修补目标的需要，将 CVE ID 添加到补丁基准中已批准或已拒绝的补丁列表中。

有关使用已批准和已拒绝的补丁列表的信息，请参阅以下主题：

- 使用自定义补丁基准
- 已批准补丁和已拒绝补丁列表的程序包名称格式
- 补丁基准规则在基于 Linux 的系统上的工作原理
- 如何安装补丁

> **Note**
>
> 在某些情况下，Microsoft 会为未指定更新日期和时间的应用程序发布补丁。在这些情况下，系统会默认提供 `01/01/1970` 的更新日期和时间。

查看补丁合规性结果

使用以下过程在 AWS Systems Manager 控制台中查看合规性数据。

> **Note**
>
> 有关生成下载到 Amazon Simple Storage Service (Amazon S3) 存储桶的补丁合规性报告的信息，请参阅 生成 .csv 补丁合规性报告。

查看补丁合规性结果

1. 请执行以下操作之一。

   选项 1（推荐）：从Patch Manager（AWS Systems Manager 中的一项工具）中导航：

   - 在导航窗格中，请选择 Patch Manager。
   - 选择 Compliance reporting（合规性报告）选项卡。
   - 在节点修补详细信息区域中，选择要查看其补丁合规性结果的托管式节点 ID。
   - 在详细信息区域的属性列表中，选择补丁。

   选项 2：从 Compliance（AWS Systems Manager 中的一项工具）中导航：

- 在导航窗格中，选择 合规性。
- 适用于合规性资源摘要，请在列中为要查看的补丁资源类型选择一个数字，例如不合规资源。
- 在下面的资源列表中，选择要查看其补丁合规性结果的托管式节点的 ID。
- 在详细信息区域的属性列表中，选择补丁。

选项 3：从 Fleet Manager（AWS Systems Manager 中的一项工具）中导航：

- 在导航窗格中，请选择 Fleet Manager。
- 在托管式实例区域中，选择要查看其补丁合规性结果的托管式节点的 ID。
- 在详细信息区域的属性列表中，选择补丁。

2. （可选）在搜索框
()
中，从可用筛选器中选择。

例如，对于 Red Hat Enterprise Linux (RHEL)，从以下选项中进行选择：

- 名称
- 分类
- 状态
- 严重性

对于 Windows Server，从以下选项中进行选择：

- KB
- 分类
- 状态
- 严重性

3. 为您选择的筛选器类型选择一个可用值。例如，如果您选择了状态，现在选择一个合规性状态，例如已安装待定重启、已失败或者缺少。

> **Note**
>
> 目前，仅报告状态为 `Missing` 或 `Failed` 的补丁的 CVE ID 值。

4. 根据托管式节点的合规性状态，您可以选择采取哪些操作来补救任何不合规的节点。

   例如，您可以选择立即修补不合规托管式节点。有关按需修补托管式节点的信息，请参阅 按需修补托管式节点。

   有关补丁合规性数据的信息，请参阅 补丁合规性状态值。

生成 .csv 补丁合规性报告

您可以使用 AWS Systems Manager 控制台来生成补丁合规性报告，这些报告将以 .csv 文件格式另存到您选择的 Amazon Simple Storage Service (Amazon S3) 存储桶。您可以生成单个按需报告或制定自动生成报告的计划。

可以在所选择的 AWS 账户 和 AWS 区域 中为单个托管式节点或所有托管式节点生成报告。对于单个节点，报告包含全面的详细信息，包括与不合规节点相关的补丁 ID。对于所有托管式节点的报告，仅提供摘要信息和不合规节点的补丁计数。

生成报告后，您可以使用 Amazon QuickSight 等工具导入和分析数据。QuickSight 是一项商业智能（BI）服务，可用于在交互式视觉环境中浏览和解读信息。有关更多信息，请参阅 Amazon QuickSight 用户指南。

> **Note**
>
> 创建自定义补丁基准时，您可以为此补丁基准批准的补丁指定合规性严重性级别，例如 `Critical` 或 `High`。如果任何已批准补丁的补丁状态报告为 `Missing`，则补丁基准报告的总体合规性严重性级别就是您指定的严重级别。

也可以指定一个在生成报告时发送通知的 Amazon Simple Notification Service (Amazon SNS) 主题。

生成补丁合规性报告的服务角色

首次生成报告时，Systems Manager 会创建名为 `AWS-SystemsManager-PatchSummaryExportRole` 的自动化担任角色，用于导出到 S3 的过程。

> **Note**
>
> 如果您要将合规性数据导出到加密的 S3 存储桶，则必须更新其关联的 AWS KMS 密钥政策以为 `AWS-SystemsManager-PatchSummaryExportRole` 提供必要的权限。例如，将与此类似的权限添加到 S3 存储桶的 AWS KMS 策略中：

```
{
    "Effect": "Allow",
    "Action": [
        "kms:GenerateDataKey"
    ],
    "Resource": "role-arn"
}
```

将 *role-arn* 替换为账户中创建的 Amazon 资源名称（ARN），格式为 arn:aws:iam::*111222333444*:role/service-role/AWS-SystemsManager-PatchSummaryExportRole。
有关更多信息，请参阅《AWS Key Management Service 开发人员指南》中的在 AWS KMS 中使用密钥策略。

首次按计划生成报告时，Systems Manager 会创建另一个名为 AWS-EventBridge-Start-SSMAutomationRole 的另一个服务角色，以及服务角色 AWS-SystemsManager-PatchSummaryExportRole（如果尚未创建）以用于导出过程。AWS-EventBridge-Start-SSMAutomationRole 使 Amazon EventBridge 能够使用运行手册AWS-导出补丁报告到 S3 来启动自动化。

我们建议不要尝试修改这些策略和角色。这样做可能会导致补丁合规性报告生成失败。有关更多信息，请参阅 解决补丁合规性报告生成中的问题。

主题

- 生成的补丁合规性报告中有哪些内容？
- 为单个托管式节点生成补丁合规性报告
- 为所有托管式节点生成补丁合规性报告
- 查看补丁合规性报告历史
- 查看补丁合规性报告计划
- 解决补丁合规性报告生成中的问题

生成的补丁合规性报告中有哪些内容？

本主题提供有关生成并下载到指定 S3 存储桶的补丁合规性报告中包含的内容类型的信息。

单个托管式节点的报告格式

为单个托管式节点生成的报告提供摘要信息和详细信息。

下载示例报告（单个节点）

单个托管式节点的摘要信息包括以下内容：

- 索引
- 实例 ID
- 实例名称
- 实例 IP
- 平台名称
- 平台版本
- SSM Agent 版本
- 补丁基准
- 补丁组
- 合规性状态
- 合规性严重性
- 不合规的重大严重性补丁计数
- 不合规的高严重性补丁计数
- 不合规的中等严重性补丁计数
- 不合规的低严重性补丁计数
- 不合规的信息严重性补丁计数
- 不合规的未指定严重性补丁计数

单个托管式节点的详细信息包括以下内容：

- 索引
- 实例 ID
- 实例名称
- 补丁名称
- KB ID/补丁 ID

- 修补状态
- 上次报告时间
- 合规级别
- 补丁严重性
- 补丁分类
- CVE ID
- 补丁基准
- 日志 URL
- 实例 IP
- 平台名称
- 平台版本
- SSM Agent 版本

> **Note**
>
> 创建自定义补丁基准时，您可以为此补丁基准批准的补丁指定合规性严重性级别，例如 `Critical` 或 `High`。如果任何已批准补丁的补丁状态报告为 `Missing`，则补丁基准报告的总体合规性严重性级别就是您指定的严重级别。

所有托管式节点的报告格式

为所有托管式节点生成的报告仅提供摘要信息。

下载示例报告（所有托管式节点）

所有托管式节点的摘要信息包括以下内容：

- 索引
- 实例 ID
- 实例名称
- 实例 IP
- 平台名称
- 平台版本

- SSM Agent 版本
- 补丁基准
- 补丁组
- 合规性状态
- 合规性严重性
- 不合规的重大严重性补丁计数
- 不合规的高严重性补丁计数
- 不合规的中等严重性补丁计数
- 不合规的低严重性补丁计数
- 不合规的信息严重性补丁计数
- 不合规的未指定严重性补丁计数

为单个托管式节点生成补丁合规性报告

在 AWS 账户 中使用以下过程为单个托管式节点生成补丁摘要报告。单个托管式节点的报告提供有关每个不合规补丁的详细信息，包括补丁名称和 ID。

为单个托管式节点生成补丁合规性报告

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Patch Manager。
3. 选择 Compliance reporting（合规性报告）选项卡。
4. 选择要为其生成报告的托管式节点所在行的按钮，然后选择 View detail（查看详细信息）。
5. 在 Patch summary（补丁摘要）部分中，选择 Export to S3（导出到 S3）。
6. 对于报告名称，输入名称以帮助您以后识别该报告。
7. 对于报告频率，选择下列选项之一：
   - 按需 — 创建一次性报告。跳至步骤 9。
   - 按计划 — 指定自动生成报告的定期计划。继续执行步骤 8。
8. 对于计划类型，请指定速率表达式（如每 3 天），或者提供 cron 表达式来设置报告频率。

   有关 Cron 表达式的更多信息，请参阅 参考：适用于 Systems Manager 的 Cron 和 Rate 表达式。
9. 对于于 Bucket name，选择要存储 .csv 报告文件的 S3 存储桶的名称。

> **Important**
>
> 如果您处于 2019 年 3 月 20 日之后启动的 AWS 区域 中，则必须在该区域选择 S3 存储桶。默认情况下，在该日期之后启动的区域会被关闭。有关这些区域的详细信息和列表，请参阅《Amazon Web Services 一般参考》中的 Enabling a Region。

10. （可选）要在报告生成时发送通知，请展开 SNS topic（SNS 主题）部分，然后从 SNS topic Amazon Resource Name (ARN) [SNS 主题的 Amazon Resource Name (ARN)] 中选择一个现存的 Amazon SNS 主题。
11. 选择提交。

有关查看生成报告的历史记录的信息，请参阅 查看补丁合规性报告历史。

有关查看已创建的报告计划详细信息的信息，请参阅 查看补丁合规性报告计划。

为所有托管式节点生成补丁合规性报告

在 AWS 账户 中使用以下过程为所有托管式节点生成补丁摘要报告。所有托管式节点的报告会指明哪些节点不合规以及不合规补丁的数量。它不提供补丁的名称或其他标识符。对于这些额外详细信息，您可以为单个托管式节点生成补丁合规性报告。有关更多信息，请参阅本主题前面的 为单个托管式节点生成补丁合规性报告。

为所有托管式节点生成补丁合规性报告

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Patch Manager。
3. 选择 Compliance reporting（合规性报告）选项卡。
4. 选择导出到 S3。（不要先选择节点 ID。）
5. 对于报告名称，输入名称以帮助您以后识别该报告。
6. 对于报告频率，选择下列选项之一：

   - 按需 — 创建一次性报告。跳至步骤 8。
   - 按计划 — 指定自动生成报告的定期计划。继续执行步骤 7。
7. 对于计划类型，请指定速率表达式（如每 3 天），或者提供 cron 表达式来设置报告频率。

   有关 Cron 表达式的更多信息，请参阅 参考：适用于 Systems Manager 的 Cron 和 Rate 表达式。

8. 对于于 Bucket name，选择要存储 .csv 报告文件的 S3 存储桶的名称。

> **Important**
>
> 如果您处于 2019 年 3 月 20 日之后启动的 AWS 区域 中，则必须在该区域选择 S3 存储桶。默认情况下，在该日期之后启动的区域会被关闭。有关这些区域的详细信息和列表，请参阅《Amazon Web Services 一般参考》中的 Enabling a Region。

9. （可选）要在报告生成时发送通知，请展开 SNS topic（SNS 主题）部分，然后从 SNS topic Amazon Resource Name (ARN) [SNS 主题的 Amazon Resource Name (ARN)] 中选择一个现存的 Amazon SNS 主题。
10. 选择提交。

有关查看生成报告的历史记录的信息，请参阅 查看补丁合规性报告历史。

有关查看已创建的报告计划详细信息的信息，请参阅 查看补丁合规性报告计划。

查看补丁合规性报告历史

使用本主题中的信息帮助您查看有关 AWS 账户 中生成的补丁合规性报告的详细信息。

查看补丁合规性报告历史

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Patch Manager。
3. 选择 Compliance reporting（合规性报告）选项卡。
4. 选择查看所有 S3 导出，然后选择导出历史记录选项卡。

查看补丁合规性报告计划

使用本主题中的信息帮助您查看在 AWS 账户 中生成的补丁合规性报告历史的详细信息。

查看补丁合规性报告历史

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Patch Manager。

3. 选择 Compliance reporting（合规性报告）选项卡。
4. 选择 View all S3 exports（查看所有 S3 导出），然后选择 Report schedule rules（报告计划规则）选项卡。

解决补丁合规性报告生成中的问题

利用以下信息，帮助排查Patch Manager（AWS Systems Manager 中的一项工具）中有关生成补丁合规性报告的问题。

主题

- 报告 AWS-SystemsManager-PatchManagerExportRolePolicy 策略已损坏的消息
- 在删除补丁合规性策略或角色后，未成功按照计划生成报告

报告 **AWS-SystemsManager-PatchManagerExportRolePolicy** 策略已损坏的消息

问题：您收到类似于以下内容的错误消息，指示 AWS-SystemsManager-PatchManagerExportRolePolicy 已损坏：

```
An error occurred while updating the AWS-SystemsManager-PatchManagerExportRolePolicy
policy. If you have edited the policy, you might need to delete the policy, and any
role that uses it, then try again. Systems Manager recreates the roles and policies
you have deleted.
```

- 解决方案：在生成新的补丁合规性报告之前，使用 Patch Manager 控制台或 AWS CLI 删除受影响的角色和策略。

  使用控制台删除损坏的策略

  1. 访问：https://console.aws.amazon.com/iam/，打开 IAM 控制台
  2. 请执行以下操作之一：

     按需报告 — 如果在生成一次性按需报告时出现问题，请在左侧导航栏中选择策略，搜索 AWS-SystemsManager-PatchManagerExportRolePolicy，然后删除该策略。下一步，选择角色，搜索 AWS-SystemsManager-PatchSummaryExportRole，然后删除该角色。

     计划报告：如果在按计划生成报告时出现问题，请在左侧导航栏中选择策略，一次搜索一个 AWS-EventBridge-Start-SSMAutomationRolePolicy 和 AWS-SystemsManager-PatchManagerExportRolePolicy，然后删除每个策略。下一步，选择角色，一次搜

索一个 AWS-EventBridge-Start-SSMAutomationRole 和 AWS-SystemsManager-PatchSummaryExportRole，然后删除每个角色。

使用 AWS CLI 删除损坏的策略

请将####替换为账户 ID。

- 如果在生成一次性按需报告时出现问题，请运行以下命令：

```
aws iam delete-policy --policy-arn arn:aws:iam::account-id:policy/AWS-SystemsManager-PatchManagerExportRolePolicy
```

```
aws iam delete-role --role-name AWS-SystemsManager-PatchSummaryExportRole
```

如果在按计划生成报告时出现问题，请运行以下命令：

```
aws iam delete-policy --policy-arn arn:aws:iam::account-id:policy/AWS-EventBridge-Start-SSMAutomationRolePolicy
```

```
aws iam delete-policy --policy-arn arn:aws:iam::account-id:policy/AWS-SystemsManager-PatchManagerExportRolePolicy
```

```
aws iam delete-role --role-name AWS-EventBridge-Start-SSMAutomationRole
```

```
aws iam delete-role --role-name AWS-SystemsManager-PatchSummaryExportRole
```

完成任一过程后，按照以下步骤生成或计划新的补丁合规性报告。

在删除补丁合规性策略或角色后，未成功按照计划生成报告

问题：首次生成报告时，Systems Manager 会创建一个服务角色和一个策略，用于导出过程 (AWS-SystemsManager-PatchSummaryExportRole 和 AWS-SystemsManager-PatchManagerExportRolePolicy)。首次按计划生成报告时，Systems Manager 会创建另一个服务角色和策略 (AWS-EventBridge-Start-SSMAutomationRole 和 AWS-EventBridge-Start-SSMAutomationRolePolicy)。这些功能让 Amazon EventBridge 使用运行手册启动自动化 AWS 导出补丁报告到 S3。

如果您删除这些策略或角色中的任何一个，则计划与指定的 S3 存储桶和 Amazon SNS 主题之间的连接可能会丢失。

- 解决方案：若要变通解决此问题，我们建议删除以前的计划，并创建一个新的计划来替换遇到问题的计划。

使用 Patch Manager 修复不合规的托管式节点

本节中的主题概述了如何标识不符合补丁合规性要求的托管式节点以及如何使节点合规。

主题

- 标识不合规的托管式节点
- 补丁合规性状态值
- 修补不合规的托管式节点

标识不合规的托管式节点

当运行两个 AWS Systems Manager 文档（SSM 文档）中的其中一个文档时，会标识不合规的托管式节点。这些 SSM 文档引用了Patch Manager（AWS Systems Manager 中的一项工具）中每个托管式节点相应的补丁基准。然后，这些文档会评估托管式节点的补丁状态，并向您提供合规性结果。

有两个 SSM 文档用于标识或更新不合规的托管式节点：`AWS-RunPatchBaseline` 和 `AWS-RunPatchBaselineAssociation`。每个文档的使用流程不同，其合规性结果通过不同的渠道提供。下表概述了这些文档之间的差异。

> **Note**
>
> 来自 Patch Manager 的补丁合规性数据可以发送到 AWS Security Hub。Security Hub 能让您全面了解高优先级安全警报和合规性状态。它还监控您的机群的修补状态。有关更多信息，请参阅 将 Patch Manager 与 AWS Security Hub 集成。

	AWS-RunPatchBaseline	**AWS-RunPatchBaselineAssociation**
使用文档的过程	按需修补 – 您可以使用 Patch now（立即修补）选项按需扫	Systems Manager Quick Setup 主机管理：您可以在

	AWS-RunPatchBaseline	AWS-RunPatchBaselineAssociation
	描或修补托管式节点。有关信息，请参阅按需修补托管式节点。 Systems Manager Quick Setup补丁策略：您可以在 Quick Setup（AWS Systems Manager 中的一项工具）中创建修补配置，可针对整个组织、部分组织单位或单个 AWS 账户按不同的计划扫描或安装缺失的补丁。有关信息，请参阅使用 Quick Setup 补丁策略为组织中的实例配置修补。 运行命令：在 Run Command（AWS Systems Manager 中的一项工具）的操作中，您可以手动运行 AWS-RunPatchBaseline 。有关信息，请参阅从控制台运行命令。 维护时段 – 在 Run Command 任务类型中，您可以使用 SSM 文档 AWS-RunPatchBaseline 创建维护时段。有关信息，请参阅教程：使用控制台创建修补的维护时段。	Quick Setup 中启用主机管理配置选项，每天扫描您的托管实例的补丁合规性。有关信息，请参阅使用 Quick Setup 设置 Amazon EC2 主机管理。 Systems Manager Explorer：只要允许 Explorer（AWS Systems Manager 中的一项工具），该工具就会定期扫描托管式实例来了解补丁合规性，并会在 Explorer 控制面板中报告结果。

	AWS-RunPatchBaseline	AWS-RunPatchBaselineAssociation
补丁扫描结果数据的格式	在 AWS-RunPatchBaseline 运行后，Patch Manager 会将 AWS:PatchSummary 对象发送至 Inventory（AWS Systems Manager 中的一项工具）。	在 AWS-RunPatchBaselineAssociation 运行后，Patch Manager 发送 AWS:ComplianceItem 对象至 Systems Manager 库存。
在控制台中查看补丁合规性报告	您可以查看使用 Systems Manager 配置合规性中的 AWS-RunPatchBaseline 和 使用托管式节点 进程的补丁合规性信息。有关更多信息，请参阅 查看补丁合规性结果。	如果您使用 Quick Setup 扫描托管实例的补丁合规性，您可以在 Systems Manager Fleet Manager 中查看合规性报告。在 Fleet Manager 控制台中，选择托管式节点的节点 ID。在常规菜单中，选择配置合规性。 如果您将Explorer 扫描托管实例的补丁合规性，您可以在 Explorer 和 Systems Manager OpsCenter 中查看合规性报告。
查看补丁合规性结果的 AWS CLI 命令	对于使用 AWS-RunPatchBaseline 的进程，您可以使用以下 AWS CLI 命令查看有关托管式节点上的补丁的摘要信息。 • describe-instance-patch-states • describe-instance-patch-states-for-patch-group • describe-patch-group-state	对于使用 AWS-RunPatchBaselineAssociation 的进程，您可以使用以下 AWS CLI 命令查看有关实例补丁的摘要信息。 • list-compliance-items

	AWS-RunPatchBaseline	AWS-RunPatchBaselineAssociation
修补操作	对于使用 AWS-RunPatchBaseline 的进程，可以指定是否仅希望该操作运行 Scan 操作，或者 Scan and install 操作。 如果您的目标是标识而非修复不合规的托管式节点，则仅运行 Scan 操作。	Quick Setup 和 Explorer 进程，使用 AWS-RunPatchBaselineAssociation ，请仅运行 Scan 操作。
更多信息	用于修补的 SSM 命令文档：AWS-RunPatchBaseline	用于修补的 SSM 命令文档：AWS-RunPatchBaselineAssociation

有关您可能看到报告的各种补丁合规性状态的信息，请参阅 补丁合规性状态值

有关修复不符合补丁合规性要求的托管式节点的信息，请参阅 修补不合规的托管式节点。

补丁合规性状态值

有关托管式节点补丁的信息包括每个单一补丁的状态报告。

> **Tip**
>
> 如果要将特定补丁合规性状态分配给托管式节点，可以使用 put-compliance-items AWS Command Line Interface（AWS CLI）命令或 PutComplianceItems API 操作。控制台中不支持分配合规性状态。

使用下表中的信息可帮助您确定托管式节点可能不符合补丁合规性要求的原因。

Debian Server、Raspberry Pi OS 和 Ubuntu Server 的补丁合规性值

对于 Debian Server、Raspberry Pi OS 和 Ubuntu Server，将软件包分类到不同合规性状态的规则如下表所述。

> **Note**
>
> 在评估 INSTALLED、INSTALLED_OTHER 和 MISSING 状态值时，请记住以下事项：如果在创建或更新补丁基准时未选中包括非安全性更新复选框，则补丁候选版本仅限于以下存储库中的补丁：
>
> - Ubuntu Server 14.04 LTS：trusty-security
> - Ubuntu Server 16.04 LTS：xenial-security
> - Ubuntu Server 18.04 LTS：bionic-security
> - Ubuntu Server 20.04 LTS：focal-security
> - Ubuntu Server 20.10 STR：groovy-security
> - Ubuntu Server 22.04 LTS：jammy-security
> - Ubuntu Server 23.04 (lunar-security)
> - Ubuntu Server 23.10 (mantic-security)
> - Ubuntu Server 24.04 LTS (noble-security)
> - Ubuntu Server 24.10 (oracular-security)
> - Ubuntu Server 25.04 (plucky-security)
> - debian-security (Debian Server 和 Raspberry Pi OS)
>
> 如果选择了包括非安全更新复选框，也会考虑来自其他存储库的补丁。

修补状态	描述	合规性状态
INSTALLED	补丁在补丁基准中列出，并已安装在托管式节点上。如果托管式节点上已运行 AWS-RunPatchBaseline 文档，则可能是由个人手动安装或由 Patch Manager 自动安装补丁。	合规
INSTALLED_OTHER	补丁不包括在基准中，或者未获基准批准，但已安装在	合规

修补状态	描述	合规性状态
	托管式节点上。该补丁可能已手动安装，该软件包可能是另一个已批准补丁的必需依赖项，或者该补丁可能已包含在 InstallOverrideList 操作中。如果您没有指定 Block 作为拒绝的修补行动,INSTALLED_OTHER 补丁还包括已安装但拒绝的补丁。	
INSTALLED_PENDING_REBOOT	INSTALLED_PENDING_REBOOT 可能表示以下任一情况： • Patch Manager Install 操作已对托管式节点应用补丁，但自应用补丁以来尚未重启该节点。这通常意味着，当 AWS-RunPatchBaseline 文档上次在托管式节点上运行时，为 RebootOption 参数选择了 NoReboot 选项。 有关更多信息，请参阅 参数名称: RebootOption 。 • 自上次重新启动托管式节点以来，在 Patch Manager 外部安装了补丁。 无论是哪种情况，都不表示具有此状态的补丁需要重新启动，而是只表示自安装补丁以来该节点尚未重新启动。	不合规

修补状态	描述	合规性状态
INSTALLED_REJECTED	该补丁已安装在托管式节点上，但列在 Rejected patches（已被拒绝补丁）列表中。这通常意味着补丁是在添加到已拒绝的补丁列表之前安装的。	不合规
MISSING	经过基准筛选且尚未安装的软件包。	不合规
FAILED	修补操作期间安装失败的软件包。	不合规

其他操作系统的补丁合规性值

对于除 Debian Server、Raspberry Pi OS 和 Ubuntu Server 之外的所有操作系统，将软件包分类到不同合规性状态的规则如下表所述。

修补状态	描述	合规性值
INSTALLED	补丁在补丁基准中列出，并已安装在托管式节点上。如果节点上已运行 `AWS-RunPatchBaseline` 文档，则可能是由个人手动安装或由 Patch Manager 自动安装补丁。	合规
INSTALLED_OTHER [1]	补丁不在基准中，但已安装在托管式节点上。有两个可能的原因： 1. 可能是手动安装该补丁。 2. 仅限 Linux：该软件包可能是作为其他已批准补丁的必	合规

修补状态	描述	合规性值
	需依赖项安装的。如果您将 `Allow as dependency` 指定为已拒绝的补丁操作，则作为依赖项安装的补丁将获得报告状态 `INSTALLED_OTHER` 。 **Note** Windows Server 不支持补丁依赖项的概念。有关 Patch Manager 如何处理 Windows Server 上的已拒绝的补丁列表中补丁的信息，请参阅自定义补丁基准中的“已拒绝的补丁”列表选项。	
`INSTALLED_REJECTED`	该补丁已安装在托管式节点上，但列在已被拒绝补丁列表中。这通常意味着补丁是在添加到已拒绝的补丁列表之前安装的。	不合规

修补状态	描述	合规性值
INSTALLED_PENDING_REBOOT	INSTALLED_PENDING_REBOOT 可能表示以下任一情况： • Patch Manager Install 操作已对托管式节点应用补丁，但自应用补丁以来尚未重启该节点。这通常意味着，当 AWS-RunPatchBaseline 文档上次在托管式节点上运行时，为 RebootOption 参数选择了 NoReboot 选项。 有关更多信息，请参阅 参数名称: RebootOption 。 • 自上次重新启动托管式节点以来，在 Patch Manager 外部安装了补丁。 无论是哪种情况，都不表示具有此状态的补丁需要重新启动，而是只表示自安装补丁以来该节点尚未重新启动。	不合规
MISSING	基准中已批准补丁，但托管式节点上未安装该补丁。如果将 AWS-RunPatchBaseline 文档任务配置为扫描（而不是安装），系统将为扫描期间找到但未安装的补丁报告此状态。	不合规

修补状态	描述	合规性值
FAILED	基准中已批准补丁，但无法安装补丁。要解决此问题，请查看命令输出中是否有可帮助您理解此问题的信息。	不合规
NOT_APPLICABLE [1]	此合规性状态仅针对 Windows Server 操作系统报告。 基准中已批准补丁，但托管式节点上未安装使用该补丁的服务或功能。例如，如果基准中已批准 Web 服务器服务 [如 Internet Information Services (IIS)] 的补丁，但托管式节点上未安装该 Web 服务，则该补丁将显示 NOT_APPLICABLE 。如果补丁已由后续更新取代，则也可以将补丁标记为 NOT_APPLICABLE 。这意味着安装了更高版本的更新，并且不再需要 NOT_APPLICABLE 更新。	不适用

修补状态	描述	合规性值
AVAILABLE_SECURITY_UPDATES	此合规性状态仅针对 Windows Server 操作系统报告。 未经补丁基准批准的可用安全更新补丁可以具有合规性值、Compliant 或 Non-Compliant ，如自定义补丁基准中所定义。 创建或更新补丁基准时，需选择要分配给可用但尚未批准的安全补丁（这些补丁由于不符合补丁基准中指定的安装标准而未获批准）的状态。例如，如果您指定在补丁发布后等待较长时间再进行安装，则会跳过某些您可能希望安装的安全补丁。如果在您指定的等待期内发布了该补丁的更新版本，则该补丁的安装等待期将重新开始。如果等待期过长，则可能会发布多个版本的补丁，但不会进行安装。 对于补丁摘要计数，当某个补丁报告为 Available SecurityUpdate 时，它将始终计入 AvailableSecurityUpdateCount 。如果基准配置为将此类补丁报告为 NonCompliant ，则它们也会计入 SecurityNonCompliantCount 。如果基准配置为将此类补丁报告为 Compliant ，则它们不会	“合规”或“不合规”，取决于为可用安全更新选择的选项。 **Note** 通过控制台创建或更新补丁基准，需在可用安全更新合规性状态字段中指定此选项。通过 AWS CLI 运行 create-patch-baseline 或 update-patch-baseline 命令时，需在 available-security-updates-compliance-status 参数中指定此选项。

修补状态	描述	合规性值
	计入 SecurityNonCompliantCount 。此类补丁始终以未指定严重性的形式报告，且永远不会计入 CriticalNonCompliantCount 。	

[1] 对于状态为 INSTALLED_OTHER 和 NOT_APPLICABLE 的补丁，根据 describe-instance-patches 命令，Patch Manager 会从查询结果中省略一些数据，例如 Classification 和 Severity 的值。这样做有助于防止超出 Inventory（AWS Systems Manager 中的一项工具）中单个节点的数据限制。要查看所有补丁的详细信息，可以使用 describe-available-patches 命令。

修补不合规的托管式节点

许多 AWS Systems Manager 工具和流程均可用来检查托管式节点是否符合补丁合规性要求，您也可以使用这些工具和流程来使节点符合当前应用的补丁规则。要使托管式节点符合补丁合规性要求，Patch Manager（AWS Systems Manager 中的一项工具）必须运行 Scan and install 操作。（如果您的目标仅是标识而非修复不合规的托管式节点，请改为运行 Scan 操作。有关更多信息，请参阅 标识不合规的托管式节点。）

使用 Systems Manager 安装补丁

您可以从多个工具中选择以运行 Scan and install 操作：

- （推荐）在Quick Setup（Systems Manager 中的一项工具）中配置补丁策略让您可以按计划为整个组织、部分组织单位或单个 AWS 账户安装缺失的补丁。有关更多信息，请参阅 使用 Quick Setup 补丁策略为组织中的实例配置修补。
- 在 Run Command 任务类型中，创建使用 Systems Manager 文档（SSM 文档）AWS-RunPatchBaseline 的维护时段。有关信息，请参阅教程：使用控制台创建修补的维护时段。
- 在 Run Command 操作中，手动运行 AWS-RunPatchBaseline。有关信息，请参阅从控制台运行命令。
- 使用修补选项，按需安装补丁。有关信息，请参阅按需修补托管式节点。

避免意外覆盖补丁合规性数据

如果您用多种操作来扫描实例的补丁合规性，则每次扫描都会覆盖先前扫描的补丁合规性数据。作为结果，您最终可能会导致补丁合规性数据中产生意外结果。

例如，假设您创建了一个补丁策略，该策略每天在当地时间凌晨 2 点扫描补丁合规性。该补丁策略使用的补丁基准是针对严重性标记为 `Critical`、`Important` 和 `Moderate` 的补丁。该补丁基准还指定了一些特定的已拒绝补丁。

同时，假设您已经设置了一个没有删除或停用的维护时段，每天在当地时间凌晨 4 点扫描同一组托管节点。该维护时段的任务使用其他补丁基准，该基准仅针对 `Critical` 严重性的补丁，且不排除任何特定补丁。

当维护时段执行第二次扫描时，第一次扫描的补丁合规性数据将被删除，并替换为第二次扫描的补丁合规性数据。

因此，我们强烈建议在修补操作中仅使用一种自动方法进行扫描和安装。如果您要设置补丁策略，应该删除或停用其他扫描补丁合规性的方法。有关更多信息，请参阅以下主题：

- 要从维护时段删除修补操作任务：使用控制台更新或注销维护时段任务
- 删除 State Manager 关联：删除关联。

要在“主机管理”配置中停用每日补丁合规性扫描，请在 Quick Setup 中执行以下操作：

1. 在导航窗格中，选择 Quick Setup。
2. 选择要更新的主机管理配置。
3. 选择 Actions, Edit configuration（操作、编辑配置）。
4. 清除 Scan instances for missing patches daily（每日扫描实例以查找缺失的补丁）复选框。
5. 选择更新。

> **Note**
>
> 使用 Patch now（立即修补）选项扫描托管节点的合规性也会覆盖补丁合规性数据。

按需修补托管式节点

使用Patch Manager（AWS Systems Manager 中的一项工具）中的立即修补选项，您可以从 Systems Manager 控制台运行按需修补操作。这意味着您不必创建计划来更新托管式节点的合规性状态，或在不合规的节点上安装补丁。您也不需要在Patch Manager和 Maintenance Windows（AWS Systems Manager 中的一项工具）之间切换 Systems Manager 控制台来设置或修改计划的补丁时段。

当您必须尽快在托管式节点上应用零日更新或安装其他关键补丁时，Patch now（立即修补）特别有用。

> **Note**
>
> 每次仅支持按需修补一个 AWS 账户 与 AWS 区域 对。这不能用于基于补丁策略的修补操作。我们建议使用补丁策略来确保所有托管节点满足合规性。有关使用补丁策略的更多信息，请参阅 Quick Setup 中的补丁策略配置。

主题

- “立即补丁”的工作原理
- 运行“立即修补”

“立即补丁”的工作原理

要运行立即修补，您只需指定两个必需的设置：

- 是仅扫描缺少的补丁，还是在托管式节点上扫描并安装补丁
- 要在哪个托管式节点上运行该操作

当 Patch now（立即修补）操作运行时，其确定以与为其他修补操作选择的基准相同的方式使用哪个补丁基准。如果托管式节点与补丁组关联，则将使用为该组指定的补丁基准。如果托管式节点未与补丁组关联，则该操作使用的补丁基准为当前设置为托管式节点操作系统类型的默认补丁基准。这可以是预定义基准，也可以是您设置为默认基准的自定义基准。有关选择补丁基准的更多信息，请参阅 补丁组。

您可以为 Patch now（立即修补）指定的选项包括：选择修补后重启托管式节点的时间或是否重启托管式节点，指定 Amazon Simple Storage Service (Amazon S3) 存储桶来存储修补操作的日志数据，以及在修补期间将 Systems Manager 文档（SSM 文档）作为生命周期钩子运行。

“立即修补”的并发和错误阈值

适用于立即修补操作，并发和错误阈值选项由 Patch Manager 来处理。您无需指定一次修补多少个托管式节点，也不需要指定操作失败之前允许的错误数。在按需进行修补时，Patch Manager 将应用下表中描述的并发和错误阈值设置。

> **Important**
>
> 以下阈值仅适用于 `Scan and install` 操作。对于 `Scan` 操作，Patch Manager 会尝试同时扫描多达 1000 个节点，并继续扫描，直到遇到多达 1000 个错误。

并发性：安装操作

Patch now（立即修补）操作中的托管式节点总数	一次扫描或修补的托管式节点数量
少于 25	1
25-100	5%
101 到 1000	8%
1000 以上	10%

错误阈值：安装操作

Patch now（立即修补）操作中的托管式节点总数	操作失败前允许的错误数
少于 25	1
25-100	5
101 到 1000	10
1000 以上	10

使用“立即修补”生命周期钩子

立即修补提供了在 Install 修补操作期间将 SSM 命令文档作为生命周期钩子运行的能力。您可以使用这些钩子执行诸如在修补之前关闭应用程序或在修补后或重启后对应用程序运行状况检查之类的任务。

有关使用生命周期钩子的更多信息，请参阅 用于修补的 SSM 命令文档：AWS-RunPatchBaselineWithHooks。

下表列出了适用于三个立即修补重启选项之一的可用生命周期钩子，以及每个钩子的示例用途。

生命周期钩子和示例用途

重启选项	钩子：安装前	钩子：安装后	钩子：退出时	钩子：计划重启后
如果需要，重启	在开始修补之前运行 SSM 文档。 示例用途：在修补过程开始之前安全地关闭应用程序。	在修补操作结束时和托管式节点重启前运行 SSM 文档。 示例用途：在潜在重启之前运行诸如安装第三方应用程序等操作。	修补操作完成后运行 SSM 文档，重新启动实例。 示例用途：确保应用程序在修补后按预期运行。	不可用
不重启我的实例	同上。	在修补操作结束时运行 SSM 文档。 示例用途：确保应用程序在修补后按预期运行。	不可用	不可用
计划重启时间	同上。	与不重启我的实例相同。	不可用	在计划的重启完成后立即运行 SSM 文档。

重启选项	钩子：安装前	钩子：安装后	钩子：退出时	钩子：计划重启后
				示例用途：确保应用程序在重启后按预期运行。

运行“立即修补”

使用以下过程按需修补托管式节点。

要运行“立即修补”

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Patch Manager。
3. 选择立即修补。
4. 适用于修补操作，请选择以下选项之一：
   - Scan（扫描）：Patch Manager 查找托管式节点中缺少的补丁，但不进行安装。您可以在合规性控制面板中查看结果，或在你使用的其他工具中查看补丁合规性。
   - Scan and install（扫描并安装）：Patch Manager 查找并安装托管式节点中缺少的补丁。
5. 仅在上一个步骤中选择扫描并安装时，使用此步骤。对于 重启选项，请选择以下选项之一：
   - Reboot if needed（如需要，则重启）：安装后，Patch Manager 仅在需要完成补丁安装时才会重启托管式节点。
   - Don't reboot my instances（不重启我的实例）：安装后，Patch Manager 不会重启托管式节点。当您在 Patch Manager 之外选择或管理重启时，您可以手动重启节点。
   - Schedule a reboot time（计划重启时间）：为 Patch Manager 重启托管式节点，指定日期、时间和 UTC 时区。在您运行立即修补操作时，计划的重启将在 State Manager 中列为关联内容，名称为 `AWS-PatchRebootAssociation`。
6. 对于 要修补的实例，请选择以下任一项：
   - Patch all instances（修补所有实例）：在当前 AWS 区域 中，Patch Manager 在 AWS 账户 中的所有托管式节点上运行指定操作。

- Patch only the target instances I specify（仅修补我指定的目标实例）：您可以指定要在下一步中作为目标的托管式节点。

7. 仅在上一个步骤中选择仅修补我指定的目标实例时，使用此步骤。在 Target selection（目标选择）部分中，通过指定标签、手动选择节点或指定资源组，标识要在其上运行此操作的节点。

> **Note**
>
> 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。
> 如果选择以资源组为目标，请注意，基于 AWS CloudFormation 堆栈的资源组仍然必须使用默认 `aws:cloudformation:stack-id` 标签来标记。如果已删除，Patch Manager 可能无法确定属于资源组的托管式节点。

8. （可选）对于补丁日志存储，如果要从此修补操作创建和保存日志，请选择用于存储日志的 S3 存储桶。

> **Note**
>
> 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给实例的实例配置文件（适用于 EC2 实例）或 IAM 服务角色（混合激活的计算机）的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或在混合和多云环境中创建 Systems Manager 所需的 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确保与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

9. （可选）如果要在修补操作的特定点期间将 SSM 文档作为生命周期钩子运行，请执行以下操作：

- 选择使用生命周期钩子。
- 对于每个可用钩子，选择要在操作的指定点运行的 SSM 文档：
  - 安装前
  - 安装后
  - 退出时
  - 计划重启后

> **Note**
>
> 默认文档 AWS-Noop，不运行任何操作。

10. 选择立即修补。

    打开 Association execution targets (关联执行目标)页面。[“立即修补”使用State Manager（AWS Systems Manager 中的一项工具）中的关联进行操作。] 在 Operation summary（操作摘要）区域中，您可以监控指定托管式节点上的扫描或修补状态。

使用补丁基准

Patch Manager（AWS Systems Manager 中的一项工具）中的补丁基准定义获批在托管式节点上安装的补丁。您可以逐个指定批准或拒绝的补丁。也可以创建自动批准规则，指定应自动批准的某些更新类型 (例如重要更新)。拒绝补丁列表将覆盖这些规则和批准列表。要使用一系列已批准的补丁来安装特定软件包，需要先删除所有自动批准规则。如果您将补丁明确标识为已拒绝，即使它匹配自动批准规则中的所有条件，也不会被批准或安装。此外，即使补丁获批用于某个托管式节点，只有该补丁适用于托管式节点上的软件时，才会安装该补丁。

主题

- 查看 AWS 预定义补丁基准
- 使用自定义补丁基准
- 将现有补丁基准设置为默认项

更多信息

- 补丁基准

查看 AWS 预定义补丁基准

Patch Manager（AWS Systems Manager 中的一项工具）包含用于Patch Manager支持的每个操作系统的预定义补丁基准。您可以利用这些补丁基准（您不能对其进行自定义），也可创建自己的补丁基准。以下过程介绍了如何查看预定义的补丁基准，以查看它是否满足您的需求。要了解有关补丁基准的更多信息，请参阅 预定义和自定义补丁基准。

查看 AWS 预定义补丁基准

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Patch Manager。
3. 在补丁基准列表中，请选择其中一个预定义补丁基准的基准 ID。

   –或者–

   如果您是在当前 AWS 区域 首次访问 Patch Manager，请选择从概览开始，然后选择补丁基准选项卡，然后选择预定义补丁基准之一的基准 ID。

   > **Note**
   >
   > 适用于 Windows Server，提供三个预定义的补丁基准。补丁基准 `AWS-DefaultPatchBaseline` 和 `AWS-WindowsPredefinedPatchBaseline-OS` 仅支持 Windows 操作系统本身的操作系统更新。除非您指定了其他补丁基准，`AWS-DefaultPatchBaseline` 用作 Windows Server 托管式节点的默认补丁基准。这两个补丁基准中的配置设置是相同的。两者中较新的,`AWS-WindowsPredefinedPatchBaseline-OS` 被创建来区分它和 Windows Server 的第三个预定义补丁基准。补丁基准 `AWS-WindowsPredefinedPatchBaseline-OS-Applications`，可用于将补丁应用到 Windows Server 操作系统和 Microsoft 发布的受支持的应用程序。
   > 有关更多信息，请参阅 将现有补丁基准设置为默认项。

4. 在批准规则部分中，查看补丁基准配置。
5. 如果托管式节点接受该配置，则可以直接跳至过程 创建和管理补丁组。

   –或者–

   要创建自己的默认补丁基准，请继续主题 使用自定义补丁基准。

使用自定义补丁基准

Patch Manager（AWS Systems Manager 中的一项工具）包含用于Patch Manager支持的每个操作系统的预定义补丁基准。您可以利用这些补丁基准（您不能对其进行自定义），也可创建自己的补丁基准。

以下过程描述如何创建、更新和删除您自己的自定义补丁基准。要了解有关补丁基准的更多信息，请参阅 预定义和自定义补丁基准。

主题

- 创建适用于 Linux 的自定义补丁基准
- 创建适用于 macOS 的自定义补丁基准
- 创建适用于 Windows Server 的自定义补丁基准
- 更新或删除自定义补丁基准

创建适用于 Linux 的自定义补丁基准

在Patch Manager（AWS Systems Manager 中的一项工具）中，按照以下过程为 Linux 托管式节点创建自定义补丁基准。

有关为 macOS 托管式节点创建补丁基准的信息，请参阅 创建适用于 macOS 的自定义补丁基准。有关为 Windows 托管式节点创建补丁基准的信息，请参阅 创建适用于 Windows Server 的自定义补丁基准。

为 Linux 托管式节点创建自定义补丁基准

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Patch Manager。
3. 选择补丁基准选项卡，然后选择创建补丁基准。

   –或者–

   如果您是在当前 AWS 区域首次访问 Patch Manager，请选择从概览开始，然后选择补丁基准选项卡，然后选择创建补丁基准。
4. 在 Name (名称) 中，输入新补丁基准的名称，例如，`MyRHELPatchBaseline`。
5. （可选）对于 Description (描述)，输入此补丁基准的描述。
6. 对于 Operating system (操作系统)，请选择操作系统，例如 `Red Hat Enterprise Linux`。
7. 如果要在创建后即开始将此补丁基准用作所选操作系统的默认项，请选中 Set this patch baseline as the default patch baseline for ***`operating system name`*** instances (将此补丁基准设置为 <操作系统名称> 实例的默认补丁基准) 旁边的框。

> **Note**
>
> 只有在 2022 年 12 月 22 日 补丁策略 发布之前首次访问 Patch Manager时，此选项才可用。
> 有关将现有补丁基准设置为默认项的信息，请参阅 将现有补丁基准设置为默认项。

8. 在 Approval rules for operating-systems (操作系统的批准规则) 部分，使用字段创建一个或多个自动批准规则。

   - 产品：批准规则适用的操作系统版本，例如 `RedhatEnterpriseLinux7.4`。默认选择为 `All`。
   - Classification (分类)：批准规则适用于的补丁的类型，例如 `Security` 或 `Enhancement`。默认选择为 `All`。

   > **Tip**
   >
   > 您可以配置补丁基准来控制是否安装 Linux 的次要版本升级，如 RHEL 7.8。次要版本升级可由 Patch Manager 自动安装，前提是此更新在适当的存储库中可用。
   > 对于 Linux 操作系统，次要版本升级的分类方式不一致。它们可以分类为错误修复或安全更新，或者不分类，即使在同一内核版本中也是如此。以下是控制补丁基准是否安装这些补丁的几个选项。
   >
   > - 选项 1：确保在可用时安装次要版本升级的最广泛的批准规则是将 Classification（分类）指定为 `All` (*)，然后选择 Include nonsecurity updates（包括非安全更新）选项。
   > - 选项 2：为确保安装操作系统版本的补丁，您可以使用通配符 (*) 在基准的 Patch exceptions (补丁例外) 部分指定其内核格式。例如，RHEL 7.* 的内核格式为 `kernel-3.10.0-*.el7.x86_64`。
   >
   >   在补丁基准中的 Approved patches（已批准补丁）列表中输入 `kernel-3.10.0-*.el7.x86_64`，以确保所有补丁（包括次要版本升级）都应用到 RHEL 7.* 托管式节点。（如果您知道次要版本补丁的确切程序包名称，则可以输入此名称。）
   > - Option 3（选项 3）：通过使用 `AWS-RunPatchBaseline` 文档中的 InstallOverrideList 参数，您可以最大程度地控制应用于托管式节点的补丁（包括次要版本升级）。有关更多信息，请参阅 用于修补的 SSM 命令文档：AWS-RunPatchBaseline。

- Severity (严重性)：规则适用于的补丁的严重性值，例如 `Critical`。默认选择为 `All`。
- Auto-approval（自动批准）：选择要自动批准的补丁的方法。

> **Note**
>
> 由于无法可靠地确定 Ubuntu Server 的更新程序包的发布日期，因此此操作系统不支持自动批准选项。

  - 在指定的天数后批准补丁：Patch Manager 在发布或最后更新补丁之后等待的天数，然后自动批准补丁。可以输入零 (0) 到 360 的任何整数。对于大多数情况，我们建议等待期不超过 100 天。
  - 批准在特定日期之前发布的补丁：补丁发布日期，Patch Manager 自动应用在该日期或之前发布或更新的所有补丁。例如，如果指定 2023 年 7 月 7 日，则不会自动安装在 2023 年 7 月 8 日或之后发布或最后更新的任何补丁。
- （可选）合规性报告：要分配给基准批准的补丁的严重性级别，例如 `Critical` 或 `High`。

> **Note**
>
> 如果您指定合规性报告级别以及任何已批准补丁的补丁状态报告为 `Missing`，则补丁基准报告的总体合规性严重性级别就是您指定的严重级别。

- Include non-security updates (包括非安全性更新)：选中此复选框，除了可以安装与安全性相关的补丁外，还可以安装源存储库中提供的非安全性 Linux 操作系统补丁。

> **Note**
>
> 对于 SUSE Linux Enterprise Server (SLES)，无需选中此复选框，因为 SLES 托管式节点上默认安装针对安全性和非安全性问题的补丁。有关更多信息，请参阅[如何选择安全性补丁]中的 SLES 内容。

有关在自定义补丁基准中使用批准规则的更多信息，请参阅 [自定义基准]。

9. 如果要明确批准除满足批准规则的补丁外的任何补丁，请在 Patch exceptions (补丁例外) 部分执行以下操作：

   - 对于 Approved patches (已批准的补丁)，输入要批准的补丁的逗号分隔列表。

有关已批准的补丁和已拒绝的补丁列表的已接受格式的信息，请参阅 已批准补丁和已拒绝补丁列表的程序包名称格式。

- （可选）对于 Approved patches compliance level (已批准补丁合规性级别)，为列表中的补丁分配合规性级别。
- 如果您指定的任何已批准的补丁与安全性无关，请选中包括非安全性更新复选框，以便也在 Linux 操作系统上安装这些补丁。

10. 如果要明确拒绝除满足批准规则的补丁外的任何补丁，请在 Patch exceptions (补丁例外) 部分执行以下操作：

    - 对于 Rejected patches (已拒绝的补丁)，输入要拒绝的补丁的逗号分隔列表。

      有关已批准的补丁和已拒绝的补丁列表的已接受格式的信息，请参阅 已批准补丁和已拒绝补丁列表的程序包名称格式。

    - 对于已拒绝的补丁操作，请选择 Patch Manager 要对已拒绝的补丁列表中包含的补丁采取的操作。
      - 允许作为依赖项：仅当已拒绝的补丁列表中的软件包是另一个软件包的依赖项时，才安装它。它被视为符合补丁基准，并且它的状态报告为 InstalledOther。这是未指定选项时的默认操作。
      - 阻止：在任何情况下，Patch Manager 都不安装已拒绝补丁列表中的软件包以及包含相关补丁作为依赖项的软件包。如果在将软件包添加到已拒绝补丁列表之前已安装该软件包，或者之后会在 Patch Manager 外部安装该软件包，则将其视为不符合补丁基准并将状态报告为 InstalledRejected。

    > **Note**
    >
    > Patch Manager 递归搜索补丁依赖项。

11. （可选）如果您要为不同版本的操作系统 (如 AmazonLinux2016.03 和 AmazonLinux2017.09) 指定备用补丁存储库，请为 Patch sources (补丁来源) 部分中的每个产品执行以下操作：

    - 在 Name (名称) 中，输入名称以帮助您标识源配置。
    - 在 Product（产品）中，请选择补丁源存储库适用于的操作系统的版本，例如 `RedhatEnterpriseLinux7.4`。
    - 在配置中，输入要以适当格式使用的存储库配置的值：

Example for yum repositories

```
[main]
name=MyCustomRepository
baseurl=https://my-custom-repository
enabled=1
```

> **Tip**
>
> 有关 yum 存储库配置可用的其他选项的信息，请参阅 dnf.conf(5)。

Examples for Ubuntu Server and Debian 服务器

```
deb http://security.ubuntu.com/ubuntu jammy main
```

```
deb https://site.example.com/debian distribution component1
component2 component3
```

必须在一行中指定 Ubuntu Server 存储库的存储库信息。有关更多示例和信息，请参阅 Ubuntu 服务器手册网站上的 jammy (5) sources.list.5.gz 以及 Debian Wiki 上的 sources.list 格式。

选择添加其他来源来为每个其他操作系统版本指定源存储库，最多 20 个。

有关备用源补丁存储库的更多信息，请参阅 如何指定备用补丁源存储库 (Linux)。

12. （可选）对于管理标签，将一个或多个标签键名称/值对应用到补丁基准。

标签是您分配给资源的可选元数据。标签允许您按各种标准（如用途、所有者或环境）对资源进行分类。例如，您可能想要标记补丁基准来确定其指定的补丁的严重性级别、它适用的操作系统系列以及环境类型。在这种情况下，您可以指定类似于以下键名称/值对的标签：

- `Key=PatchSeverity,Value=Critical`
- `Key=OS,Value=RHEL`
- `Key=Environment,Value=Production`

13. 请选择创建补丁基准。

创建适用于 macOS 的自定义补丁基准

在Patch Manager（AWS Systems Manager 中的一项工具）中，按照以下过程为 macOS 托管式节点创建自定义补丁基准。

有关为 Windows Server 托管式节点创建补丁基准的信息，请参阅 创建适用于 Windows Server 的自定义补丁基准。有关为 Linux 托管式节点创建补丁基准的信息，请参阅 创建适用于 Linux 的自定义补丁基准。

> **Note**
>
> 并非所有 AWS 区域 都支持 macOS。有关对适用于 macOS 的 Amazon EC2 支持的一般信息，请参阅《Amazon EC2 用户指南》中的 Amazon EC2 Mac 实例。

为 macOS 托管式节点创建自定义补丁基准

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Patch Manager。
3. 选择补丁基准选项卡，然后选择创建补丁基准。

   –或者–

   如果您是在当前 AWS 区域首次访问 Patch Manager，请选择从概览开始，然后选择补丁基准选项卡，然后选择创建补丁基准。
4. 在 Name (名称) 中，输入新补丁基准的名称，例如，`MymacOSPatchBaseline`。
5. （可选）对于 Description (描述)，输入此补丁基准的描述。
6. 对于 Operating system（操作系统），请选择 macOS。
7. 如果要在创建后即开始将此补丁基准用作 macOS 的默认项，请选中将此补丁基准设置为 macOS 实例的默认补丁基准 旁边的框。

   > **Note**
   >
   > 只有在 2022 年 12 月 22 日 补丁策略 发布之前首次访问 Patch Manager时，此选项才可用。
   > 有关将现有补丁基准设置为默认项的信息，请参阅 将现有补丁基准设置为默认项。

8. 在 Approval rules for operating-systems (操作系统的批准规则) 部分，使用字段创建一个或多个自动批准规则。

   - 产品：批准规则适用的操作系统版本，例如 `Mojave10.14.1` 或 `Catalina10.15.1`。默认选择为 `All`。

     > **Note**
     >
     > Homebrew 开源软件包管理系统已停止支持 macOS 10.14.x（Mojave）和 10.15.x（Catalina）。因此，目前不支持使用 Patch Manager 对这些版本执行修补操作。

   - 分类：要在修补过程中应用软件包的软件包管理器。可从以下选项中进行选择：
     - 软件更新
     - 安装程序
     - 酿造
     - 酿造桶

     默认选择为 `All`。

   - （可选）合规性报告：要分配给基准批准的补丁的严重性级别，例如 `Critical` 或 `High`。

     > **Note**
     >
     > 如果您指定合规性报告级别以及任何已批准补丁的补丁状态报告为 `Missing`，则补丁基准报告的总体合规性严重性级别就是您指定的严重级别。

   有关在自定义补丁基准中使用批准规则的更多信息，请参阅 自定义基准。

9. 如果要明确批准除满足批准规则的补丁外的任何补丁，请在 Patch exceptions (补丁例外) 部分执行以下操作：

   - 对于 Approved patches (已批准的补丁)，输入要批准的补丁的逗号分隔列表。

     有关已批准的补丁和已拒绝的补丁列表的已接受格式的信息，请参阅 已批准补丁和已拒绝补丁列表的程序包名称格式。

   - （可选）对于 Approved patches compliance level (已批准补丁合规性级别)，为列表中的补丁分配合规性级别。

10. 如果要明确拒绝除满足批准规则的补丁外的任何补丁，请在 Patch exceptions (补丁例外) 部分执行以下操作：

    - 对于 Rejected patches (已拒绝的补丁)，输入要拒绝的补丁的逗号分隔列表。

      有关已批准的补丁和已拒绝的补丁列表的已接受格式的信息，请参阅 已批准补丁和已拒绝补丁列表的程序包名称格式。

    - 对于已拒绝的补丁操作，请选择 Patch Manager 要对已拒绝的补丁列表中包含的补丁采取的操作。
      - 允许作为依赖项：仅当已拒绝的补丁列表中的软件包是另一个软件包的依赖项时，才安装它。它被视为符合补丁基准，并且它的状态报告为 InstalledOther。这是未指定选项时的默认操作。
      - 阻止：在任何情况下，Patch Manager 都不安装已拒绝补丁列表中的软件包以及包含相关补丁作为依赖项的软件包。如果在将软件包添加到已拒绝补丁列表之前已安装该软件包，或者之后会在 Patch Manager 外部安装该软件包，则将其视为不符合补丁基准并将状态报告为 InstalledRejected。
11. （可选）对于管理标签，将一个或多个标签键名称/值对应用到补丁基准。

    标签是您分配给资源的可选元数据。标签允许您按各种标准（如用途、所有者或环境）对资源进行分类。例如，您可能想要标记补丁基准来确定其指定的补丁的严重性级别、它适用的软件包管理器以及环境类型。在这种情况下，您可以指定类似于以下键名称/值对的标签：

    - `Key=PatchSeverity,Value=Critical`
    - `Key=PackageManager,Value=softwareupdate`
    - `Key=Environment,Value=Production`
12. 请选择创建补丁基准。

创建适用于 Windows Server 的自定义补丁基准

在Patch Manager（AWS Systems Manager 中的一项工具）中，按照以下过程为 Windows 托管式节点创建自定义补丁基准。

有关为 Linux 托管式节点创建补丁基准的信息，请参阅 创建适用于 Linux 的自定义补丁基准。有关为 macOS 托管式节点创建补丁基准的信息，请参阅 创建适用于 macOS 的自定义补丁基准。

有关创建仅限于安装 Windows Service Pack 的补丁基准的示例，请参阅 教程：使用控制台创建用于安装 Windows Service Pack 的补丁基准。

创建自定义补丁基准 (Windows)

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Patch Manager。
3. 选择补丁基准选项卡，然后选择创建补丁基准。

   –或者–

   如果您是在当前 AWS 区域首次访问 Patch Manager，请选择从概览开始，然后选择补丁基准选项卡，然后选择创建补丁基准。
4. 在 Name (名称) 中，输入新补丁基准的名称，例如，`MyWindowsPatchBaseline`。
5. （可选）对于 Description (描述)，输入此补丁基准的描述。
6. 对于 Operating system（操作系统），请选择 `Windows`。
7. 对于可用安全更新合规性状态，需选择要分配给可用但尚未批准的安全补丁（这些补丁不符合补丁基准中指定的安装标准因此未获批准）的状态，即不合规或合规。

   示例场景：如果您指定在补丁发布后等待较长时间再进行安装，则会跳过某些您可能希望安装的安全补丁。如果在您指定的等待期内发布了该补丁的更新版本，则该补丁的安装等待期将重新开始。如果等待期过长，则可能会发布多个版本的补丁，但不会进行安装。
8. 如果要在创建后即开始将此补丁基准用作 Windows 的默认项，请选择 Set this patch baseline as the default patch baseline for Windows Server instances（将此补丁基准设置为 Windows Server 实例的默认补丁基准）。

   > **Note**
   >
   > 只有在 2022 年 12 月 22 日 补丁策略 发布之前首次访问 Patch Manager时，此选项才可用。
   > 有关将现有补丁基准设置为默认项的信息，请参阅 将现有补丁基准设置为默认项。
9. 在 Approval rules for operating systems (操作系统的批准规则) 部分，使用字段创建一个或多个自动批准规则。

   - 产品：批准规则适用的操作系统版本，例如 `WindowsServer2012`。默认选择为 `All`。
   - Classification (分类)：批准规则适用于的补丁的类型，例如 `CriticalUpdates`、`Drivers` 和 `Tools`。默认选择为 `All`。

> **Tip**
>
> 您可以在您的批准规则中包括 Windows 服务包安装，方法是包含 `ServicePacks` 或通过在您的分类列表中选择 `All`。有关示例，请参阅 教程：使用控制台创建用于安装 Windows Service Pack 的补丁基准。

- Severity (严重性)：规则适用于的补丁的严重性值，例如 `Critical`。默认选择为 `All`。
- Auto-approval（自动批准）：选择要自动批准的补丁的方法。
  - 在指定的天数后批准补丁：Patch Manager 在发布或更新补丁之后等待的天数，然后自动批准补丁。可以输入零 (0) 到 360 的任何整数。对于大多数情况，我们建议等待期不超过 100 天。
  - 批准在特定日期之前发布的补丁：补丁发布日期，Patch Manager 自动应用在该日期或之前发布或更新的所有补丁。例如，如果指定 2023 年 7 月 7 日，则不会自动安装在 2023 年 7 月 8 日或之后发布或最后更新的任何补丁。
- （可选）Compliance reporting（合规性报告）：要分配给基准批准的补丁的严重性级别，例如 `High`。

> **Note**
>
> 如果您指定合规性报告级别以及任何已批准补丁的补丁状态报告为 `Missing`，则补丁基准报告的总体合规性严重性级别就是您指定的严重级别。

10. （可选）在应用程序的批准规则部分，使用字段创建一个或多个自动批准规则。

> **Note**
>
> 您可以将已批准和已拒绝的补丁列表指定为补丁例外，而不是指定批准规则。请参阅步骤 10 和 11。

- Product family (产品系列)：您要为其指定规则的一般 Microsoft 产品系列，如 `Office` 或 `Exchange Server`。
- 产品：批准规则适用的应用程序版本，例如 `Office 2016` 或 `Active Directory Rights Management Services Client 2.0 2016`。默认选择为 `All`。

- Classification (分类)：批准规则适用于的补丁的类型，例如 `CriticalUpdates`。默认选择为 `All`。
- Severity (严重性)：规则适用于的补丁的严重性值，如 `Critical`。默认选择为 `All`。
- Auto-approval（自动批准）：选择要自动批准的补丁的方法。
  - 在指定的天数后批准补丁：Patch Manager 在发布或更新补丁之后等待的天数，然后自动批准补丁。可以输入零 (0) 到 360 的任何整数。对于大多数情况，我们建议等待期不超过 100 天。
  - 批准在特定日期之前发布的补丁：补丁发布日期，Patch Manager 自动应用在该日期或之前发布或更新的所有补丁。例如，如果指定 2023 年 7 月 7 日，则不会自动安装在 2023 年 7 月 8 日或之后发布或最后更新的任何补丁。
- （可选）合规性报告：要分配给基准批准的补丁的严重性级别，例如 `Critical` 或 `High`。

> **Note**
>
> 如果您指定合规性报告级别以及任何已批准补丁的补丁状态报告为 `Missing`，则补丁基准报告的总体合规性严重性级别就是您指定的严重级别。

11. （可选）如果要明确批准任何补丁，而不是允许根据批准规则选择补丁，请在修补例外部分进行以下操作：

    - 对于 Approved patches (已批准的补丁)，输入要批准的补丁的逗号分隔列表。

      有关已批准的补丁和已拒绝的补丁列表的已接受格式的信息，请参阅 已批准补丁和已拒绝补丁列表的程序包名称格式。

    - （可选）对于 Approved patches compliance level (已批准补丁合规性级别)，为列表中的补丁分配合规性级别。

12. 如果要明确拒绝除满足批准规则的补丁外的任何补丁，请在 Patch exceptions (补丁例外) 部分执行以下操作：

    - 对于 Rejected patches (已拒绝的补丁)，输入要拒绝的补丁的逗号分隔列表。

      有关已批准的补丁和已拒绝的补丁列表的已接受格式的信息，请参阅 已批准补丁和已拒绝补丁列表的程序包名称格式。

    - 对于已拒绝的补丁操作，请选择 Patch Manager 要对已拒绝的补丁列表中包含的补丁采取的操作。

- 允许作为依赖项：Windows Server 不支持软件包依赖项的概念。如果已拒绝的补丁列表中的软件包已安装在节点上，则其状态将报告为 INSTALLED_OTHER。任何尚未安装在节点上的软件包都将被跳过。
- 阻止：Patch Manager 在任何情况下都不会安装已拒绝的补丁列表中的软件包。如果在将软件包添加到已拒绝补丁列表之前已安装该软件包，或者之后会在 Patch Manager 外部安装该软件包，则将其视为不符合补丁基准并将状态报告为 INSTALLED_REJECTED。

有关已拒绝软件包操作的更多信息，请参阅自定义补丁基准中的“已拒绝的补丁”列表选项。

13. （可选）对于管理标签，将一个或多个标签键名称/值对应用到补丁基准。

标签是您分配给资源的可选元数据。标签允许您按各种标准（如用途、所有者或环境）对资源进行分类。例如，您可能想要标记补丁基准来确定其指定的补丁的严重性级别、它适用的操作系统系列以及环境类型。在这种情况下，您可以指定类似于以下键名称/值对的标签：

- Key=PatchSeverity,Value=Critical
- Key=OS,Value=RHEL
- Key=Environment,Value=Production

14. 请选择创建补丁基准。

更新或删除自定义补丁基准

您可以更新或删除自己在Patch Manager（AWS Systems Manager 中的一项工具）中创建的自定义补丁基准。更新补丁基准时，可以更改它的名称或说明、批准规则，以及对已批准和已拒绝补丁的例外。另外还可以更新应用于补丁基准的标签。您不能更改为其创建补丁基准的操作系统类型，也不能对 AWS 提供的预定义补丁基准进行更改。

更新或删除补丁基准

请按照以下步骤更新或删除补丁基准。

> **Important**
>
> 在 Quick Setup 中删除补丁策略配置可能使用的自定义补丁基准时要谨慎。
> 如果您在 Quick Setup 中使用补丁策略配置，则系统会每小时与 Quick Setup 同步一次您对自定义补丁基准所做的更新。
> 如果删除了补丁策略中引用的自定义补丁基准，那么补丁策略的 Quick Setup Configuration details（配置详细信息）页面上会显示一个横幅。横幅将通知您，补丁策略引用的补丁基准

已不存在，且后续的修补操作将失败。在这种情况下，返回 Quick Setup Configurations（配置）页面，选择 Patch Manager 配置，然后选择 Actions（操作），Edit configuration（编辑配置）。已删除的补丁基准名称将突出显示，您必须为受影响的操作系统选择新的补丁基准。

更新或删除补丁基准

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Patch Manager。
3. 请选择要更新或删除的补丁基准，然后执行以下操作之一：

   - 要从您的 AWS 账户 中删除补丁基准，请选择删除。系统将提示您确认操作。
   - 要对补丁基准的名称或说明、批准规则或补丁例外进行更改，请选择 Edit（编辑）。在 Edit patch baseline（编辑补丁基准）页面中，更改所需的值和选项，然后选择 Save changes（保存更改）。
   - 要添加、更改或删除应用到补丁基准的标签，请选择 Tags (标签) 选项卡，然后选择 Edit tags (编辑标签)。在 Edit patch baseline tags (编辑补丁基准标签) 页面上，对补丁基准标签进行更新，然后选择 Save changes (保存更改)。

   有关您可以进行的配置选择的信息，请参阅 使用自定义补丁基准。

将现有补丁基准设置为默认项

> **Important**
>
> 您在此处所作的任何默认补丁基准选择均不适用于基于补丁策略的修补操作。补丁策略将使用自身的补丁基准规范。有关补丁策略的更多信息，请参阅 Quick Setup 中的补丁策略配置。

当您在Patch Manager（AWS Systems Manager 中的一项工具）中创建自定义补丁基准时，可在创建后便将此基准设置为关联的操作系统类型的默认项。有关信息，请参阅使用自定义补丁基准。

您还可以将现有补丁基准设置为某个操作系统类型的默认项。

> **Note**
>
> 您执行的步骤取决于您首次访问 Patch Manager 是在 2022 年 12 月 22 日补丁策略发布之前还是之后。如果您在此日期之前使用过 Patch Manager，则可以使用控制台过程。否则，请使用 AWS CLI 过程。补丁策略发布前未使用 Patch Manager 的区域不会显示控制台过程中引用的操作菜单。

将补丁基准设置为默认项

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Patch Manager。
3. 选择补丁基准选项卡。
4. 在补丁基准列表中，选择当前未设置为操作系统类型的默认项的补丁基准的按钮。

   Default baseline (默认基准) 列指示哪些基准当前被设置为默认项。
5. 在 Actions (操作) 菜单中，选择 Set default patch baseline (设置默认补丁基准)。

   > **Important**
   >
   > 如果您在 2022 年 12 月 22 日之前未在当前 AWS 账户 和区域中使用 Patch Manager，则操作菜单不可用。有关更多信息，请参阅本主题前面的注释。
6. 在确认对话框中，选择 Set default (设置默认值)。

将补丁基准设置为默认项（AWS CLI）

1. 运行 describe-patch-baselines 命令以查看可用补丁基准及其 ID 和 Amazon 资源名称（ARN）的列表。

   ```
   aws ssm describe-patch-baselines
   ```
2. 运行 register-default-patch-baseline 命令将基准设置为与其关联的操作系统的默认设置。将 *baseline-id-or-ARN* 替换为要使用的自定义补丁基准或预定义基准的 ID。

Linux & macOS

```
aws ssm register-default-patch-baseline \
    --baseline-id baseline-id-or-ARN
```

以下是将自定义基准设置为默认基准的示例。

```
aws ssm register-default-patch-baseline \
    --baseline-id pb-abc123cf9bEXAMPLE
```

以下是将 AWS 托管的预定义基准设置为默认基准的示例。

```
aws ssm register-default-patch-baseline \
    --baseline-id arn:aws:ssm:us-east-2:733109147000:patchbaseline/
pb-0574b43a65ea646e
```

Windows Server

```
aws ssm register-default-patch-baseline ^
    --baseline-id baseline-id-or-ARN
```

以下是将自定义基准设置为默认基准的示例。

```
aws ssm register-default-patch-baseline ^
    --baseline-id pb-abc123cf9bEXAMPLE
```

以下是将 AWS 托管的预定义基准设置为默认基准的示例。

```
aws ssm register-default-patch-baseline ^
    --baseline-id arn:aws:ssm:us-east-2:733109147000:patchbaseline/
pb-071da192df1226b63
```

查看可用的补丁

使用Patch Manager（AWS Systems Manager 中的一项工具），您可以查看指定操作系统以及特定操作系统版本（可选）的所有可用补丁。

> **Tip**
>
> 要生成可用补丁列表并将其保存到文件中，可以使用 describe-available-patches 命令并指定您的首选输出。

查看可用补丁

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Patch Manager。
3. 选择补丁选项卡。

   –或者–

   如果您是在当前 AWS 区域首次访问 Patch Manager，请选择从概览开始，然后选择补丁选项卡。

   > **Note**
   >
   > 对于 Windows Server，补丁选项卡显示 Windows Server 更新服务（WSUS）提供的更新。

4. 对于操作系统，选择要查看其可用补丁的操作系统，例如 `Windows` 或者 `Amazon Linux`。
5. （可选）对于产品，选择操作系统版本，例如 `WindowsServer2019` 或者 `AmazonLinux2018.03`。
6. （可选）要添加或删除结果的信息列，请选择位于补丁列表右上方的配置按钮 ()。（默认情况下，补丁选项卡仅显示部分可用补丁元数据的列。）

   有关可以添加到视图中的元数据类型的信息，请参阅 AWS Systems Manager API 参考中的补丁。

创建和管理补丁组

如果您在操作中未使用补丁策略，则可以通过使用标签将托管式节点添加到补丁组来组织修补工作。

> **Note**
>
> 补丁组不会用于基于补丁策略的修补操作。有关使用补丁策略的更多信息，请参阅 Quick Setup 中的补丁策略配置。
> 对于在 2022 年 12 月 22 日发布补丁策略支持之前尚未使用补丁组的账户-区域对，控制台不支持补丁组功能。补丁组功能在此日期之前开始使用补丁组的账户区域对中仍然可用。

要在修补操作中使用标签，必须将标签键 `Patch Group` 或 `PatchGroup` 应用于托管式节点。您还必须指定要为补丁组提供的名称作为标签的值。您可以指定任何标签值，但标签键必须为 `Patch Group` 或 `PatchGroup`。

如果在 EC2 实例元数据中允许使用标签，则必须使用 `PatchGroup`（不带空格）。

使用标签对托管式节点进行分组后，请将补丁组值添加到补丁基准。通过将补丁组注册到补丁基准，您可以确保在修补操作期间安装正确的补丁。有关补丁组的更多信息，请参阅 补丁组。

完成本主题中的任务，使用带节点和补丁基准的标签对托管式节点进行修补。只有在修补 Amazon EC2 实例时才需要执行任务 1。只有在混合和多云环境中修补非 EC2 实例时才需要执行任务 2。所有托管式节点都必须执行任务 3。

> **Tip**
>
> 您还可以使用 AWS CLI 命令 `add-tags-to-resource` 或 Systems Manager API 操作 ssm-agent-minimum-s3-permissions-required`AddTagsToResource`，向托管式节点添加标签。

任务

- 任务 1：请使用标签将 EC2 实例添加到补丁组
- 任务 2：使用标签将托管式节点添加到补丁组
- 任务 3：将补丁组添加到补丁基准

任务 1：请使用标签将 EC2 实例添加到补丁组

您可以使用 Systems Manager 控制台或 Amazon EC2 控制台向 EC2 实例添加标签。只有在修补 Amazon EC2 实例时才需要执行此任务。

> **Important**
>
> 如果在实例上启用 Allow tags in instance metadata（允许在实例元数据中使用标签）选项，则无法将 `Patch Group` 标签（带空格）应用于 Amazon EC2 实例。允许在实例元数据中使用标签会导致标签密钥名称不得包含空格。如果在 EC2 实例元数据中允许使用标签，则必须使用标签键 `PatchGroup`（不带空格）。

选项 1：将 EC2 实例添加到补丁组（Systems Manager 控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Fleet Manager。
3. 在托管式节点列表中，选择您要配置以进行修补的托管式 EC2 实例的 ID。EC2 实例的节点 ID 以 `i-` 开头。

   > **Note**
   >
   > 使用 Amazon EC2 控制台和 AWS CLI 时，可以将 `Key = Patch Group` 或 `Key = PatchGroup` 标签应用于尚未配置为与 Systems Manager 结合使用的实例。
   > 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

4. 选择标签选项卡，然后选择编辑。
5. 在左列中，输入 **`Patch Group`** 或 **`PatchGroup`**。如果在 EC2 实例元数据中允许使用标签，则必须使用 `PatchGroup`（不带空格）。
6. 在右列中，输入一个标签值作为此补丁组的名称。
7. 选择保存。
8. 重复此过程，向同一补丁组中添加其他 EC2 实例。

选项 2：将 EC2 实例添加到补丁组（Amazon EC2 控制台）

1. 打开 Amazon EC2 控制台，然后在导航窗格中选择实例。
2. 从实例列表中选择您要配置用于修补的实例。
3. 在操作菜单中，依次选择实例设置、管理标签。

4. 选择添加新标签。
5. 对于 Key（键），输入 **Patch Group** 或 **PatchGroup**。如果在 EC2 实例元数据中允许使用标签，则必须使用 PatchGroup（不带空格）。
6. 对于值，输入一个值作为此补丁组的名称。
7. 选择保存。
8. 重复此过程，向同一补丁组中添加其他实例。

任务 2：使用标签将托管式节点添加到补丁组

按照本主题中的步骤，向 AWS IoT Greengrass 核心设备和非 EC2 混合激活的托管式节点（mi-*）添加标签。只有在混合和多云环境中修补非 EC2 实例时才需要执行此任务。

> **Note**
>
> 您不能使用 Amazon EC2 控制台为非 EC2 托管式节点添加标签。

将非 EC2 托管式节点添加到补丁组（Systems Manager 控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Fleet Manager。
3. 在托管实例列表中，选择您要为修补配置的托管节点的名称。

   > **Note**
   >
   > 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

4. 选择标签选项卡，然后选择编辑。
5. 在左列中，输入 **Patch Group** 或 **PatchGroup**。如果在 EC2 实例元数据中允许使用标签，则必须使用 PatchGroup（不带空格）。
6. 在右列中，输入一个标签值作为此补丁组的名称。
7. 选择保存。
8. 重复此过程，向同一补丁组添加其他托管式节点。

任务 3：将补丁组添加到补丁基准

要将特定的补丁基准与托管式节点关联，必须将补丁组值添加到补丁基准。通过将补丁组注册到补丁基准，您可以确保在修补操作期间安装正确的补丁。无论是修补 EC2 实例、非 EC2 托管式节点还是两者，都需要执行此任务。

有关补丁组的更多信息，请参阅 补丁组。

> **Note**
>
> 您执行的步骤取决于您首次访问 Patch Manager 是在 2022 年 12 月 22 日补丁策略发布之前还是之后。

将补丁组添加到补丁基准（Systems Manager 控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Patch Manager。
3. 如果您是首次在当前 AWS 区域 中访问 Patch Manager，并且 Patch Manager 起始页打开，则请选择从概览开始。
4. 选择补丁基准选项卡，然后在补丁基准列表中，选择要为补丁组配置的补丁基准的名称。

   如果您在补丁策略发布后才首次访问 Patch Manager，则必须选择已创建的自定义基准。
5. 如果基准 ID 详细信息页面包含操作菜单，请执行以下操作：

   - 选择 Actions (操作)，然后选择 Modify patch groups (修改补丁组)。
   - 在 任务 2：使用标签将托管式节点添加到补丁组 中输入您添加到托管式节点的标签值，然后选择添加。

   如果基准 ID 详细信息页面不包含操作菜单，则无法在控制台中配置补丁组。请改为执行以下操作之一：

   - （推荐）在Quick Setup（AWS Systems Manager 中的一项工具）中设置补丁策略，将补丁基准映射到一个或多个 EC2 实例。

     有关更多信息，请参阅 Using Quick Setup patch policies 和 Automate organization-wide patching using a Quick Setup patch policy。

- 使用 AWS Command Line Interface（AWS CLI）中的 register-patch-baseline-for-patch-group 命令配置补丁组。

将 Patch Manager 与 AWS Security Hub 集成

AWS Security Hub 向您提供 AWS 中安全状态的全面视图。Security Hub 会跨 AWS 账户、AWS 服务和受支持的第三方合作伙伴产品采集安全数据。利用 Security Hub，您可以根据安全行业标准和最佳实践检查您的环境。Security Hub 帮助您分析安全趋势并确定最高优先级的安全问题。

通过使用Patch Manager（AWS Systems Manager 中的一项工具）和 Security Hub 之间的集成，您可以将有关不合规节点的调查发现从Patch Manager发送至 Security Hub。结果是安全检查或与安全相关的检测的可观察记录。然后，Security Hub 可以在其对您的安保状况分析中包含这些补丁相关的检查结果。

无论您使用哪种配置方法或类型进行修补操作，以下主题中的信息都适用：

- Quick Setup 中配置的补丁策略
- Quick Setup 中配置的主机管理选项
- 运行补丁 `Scan` 或 `Install` 任务的维护时段
- 按需 Patch now（立即修补）操作

目录

- Patch Manager 将结果发送到 Security Hub 的方式
  - Patch Manager 发送的结果类型
  - 发送调查发现的延迟
  - 当 Security Hub 不可用时重试
  - 查看 Security Hub 中的 调查发现
- 来自 Patch Manager 的典型结果
- 打开并配置集成
- 如何停止发送调查发现

Patch Manager 将结果发送到 Security Hub 的方式

在 Security Hub 中，安全问题按调查结果进行跟踪。一些检查结果来自其他 AWS 服务或第三方合作伙伴检测到的问题。Security Hub 还有一套用于检测安全问题和生成结果的规则。

Patch Manager是 Systems Manager 的一项工具，能将结果发送到 Security Hub。在您通过运行 SSM 文档（`AWS-RunPatchBaseline`、`AWS-RunPatchBaselineAssociation` 或者 `AWS-RunPatchBaselineWithHooks`）来执行修补操作之后，修补信息将被发送至 Inventory 或 Compliance（都是 AWS Systems Manager 中的工具），或同时发给这两者。在库存、合规性或两者收到数据后，Patch Manager 会收到通知。然后,Patch Manager 评估数据的准确性、格式化和合规性。如果满足所有条件,Patch Manager 将数据转发到 Security Hub。

Security Hub 提供了管理来自所有这些来源的结果的工具。您可以查看和筛选结果列表，并查看结果的详细信息。有关更多信息，请参阅 AWS Security Hub 用户指南中的查看结果。您还可以跟踪调查发现的调查状态。有关更多信息，请参阅 AWS Security Hub 用户指南中对结果采取行动。

Security Hub 中的所有结果都使用标准 JSON 格式，称为 AWS 安全结果格式 (ASFF)。ASFF 包含有关问题根源、受影响资源以及调查发现当前状态的详细信息。有关更多信息，请参阅 AWS Security Hub 用户指南中的 AWS Security Finding 格式 (ASFF)。

Patch Manager 发送的结果类型

Patch Manager 使用 AWS 安全结果格式 (ASFF) 将结果发送到 Security Hub。在 ASFF 中，`Types` 字段提供结果类型。来自 Patch Manager 的结果可能具有 `Types` 的以下值：

- 软件和配置检查/补丁管理

Patch Manager 为每个不合规的托管式节点发送一个结果。结果与资源类型 `AwsEc2Instance` 一起报告，以便结果可以与其他 Security Hub 集成相关联，这些集成会报告 `AwsEc2Instance` 资源类型。只有当操作发现托管式节点不合规时，Patch Manager 才会将结果转发给 Security Hub。结果中包括补丁摘要结果。

> **Note**
>
> 向 Security Hub 报告不合规的节点之后。节点合规后 Patch Manager 不会向 Security Hub 发送更新。将所需的补丁应用到托管式节点后，您可以在 Security Hub 中手动解决调查发现的问题。

有关合规性定义的更多信息，请参阅 补丁合规性状态值。有关 `PatchSummary` 的更多信息，请参阅 AWS Security Hub API 参考中的补丁程序摘要。

发送调查发现的延迟

当 Patch Manager 创建新结果时，通常会在几秒到 2 小时内将结果发送到 Security Hub。速度取决于处理 AWS 区域 时的流量。

当 Security Hub 不可用时重试

如果存在服务中断，则运行 AWS Lambda 函数，以便在服务再次运行后将消息放回主队列。当消息进入主队列后，将自动重试。

如果 Security Hub 不可用，Patch Manager 重试发送结果，直到收到这些结果。

查看 Security Hub 中的 调查发现

此过程介绍如何在 Security Hub 中查看有关实例集中不符合补丁合规性的托管式节点的调查发现。

查看补丁合规性的 Security Hub 调查发现

1. 登录到 AWS Management Console，然后通过以下网址打开 AWS Security Hub 控制台：https://console.aws.amazon.com/securityhub/。
2. 在导航窗格中，选择 调查发现。
3. 选择添加筛选条件（ ）框。
4. 在菜单中的筛选条件下，选择产品名称。
5. 在打开的对话框中，在第一个字段中选择 is，然后在第二个字段中输入 **`Systems Manager Patch Manager`**。
6. 选择应用。
7. 添加您需要的任何其他筛选条件以帮助缩小搜索范围。
8. 在结果列表中，选择您想了解更多信息的调查发现的标题。

   屏幕右侧将打开一个窗格，其中包含有关资源、发现的问题和建议补救措施的更多详细信息。

> **Important**
>
> 目前，Security Hub 将所有托管式节点的资源类型报告为 `EC2 Instance`。这包括您已注册用于 Systems Manager 的本地服务器和虚拟机（VM）。

严重性分类

Systems Manager Patch Manager 的调查发现列表包括有关调查发现严重性的报告。严重性级别包括以下内容（从最低到最高）：

- 性 – 未发现任何问题。
- 低 – 问题不需要修复。
- 中 – 必须解决问题，但不是紧急的。
- 高 – 必须优先解决问题。
- 严重 – 必须立即解决此问题，以避免它升级。

严重性由实例上最严重的不合规软件包确定。由于您可以拥有多个具有多个严重性级别的补丁基准，因此会在所有不合规软件包中报告最高的严重性。例如，假设您有两个不合规程序包，其中程序包 A 的严重性为“严重”，而程序包 B 的严重性为“低”。“严重”将报告为严重性。

请注意，严重性字段与 Patch Manager `Compliance` 字段直接相关。这是您设置分配给与规则匹配的各个补丁的字段。由于此 `Compliance` 字段已分配给各个补丁，因此不会反映在“补丁摘要”级别。

相关内容

- 《AWS Security Hub User Guide》中的 Findings
- AWS 管理与治理博客中的 Multi-Account patch compliance with Patch Manager and Security Hub

来自 Patch Manager 的典型结果

Patch Manager 使用 AWS 安全结果格式 (ASFF)将结果发送到 Security Hub。

下面是 Patch Manager 典型结果的示例。

```
{
  "SchemaVersion": "2018-10-08",
  "Id": "arn:aws:patchmanager:us-east-2:111122223333:instance/i-02573cafcfEXAMPLE/
document/AWS-RunPatchBaseline/run-command/d710f5bd-04e3-47b4-82f6-df4e0EXAMPLE",
  "ProductArn": "arn:aws:securityhub:us-east-1::product/aws/ssm-patch-manager",
  "GeneratorId": "d710f5bd-04e3-47b4-82f6-df4e0EXAMPLE",
  "AwsAccountId": "111122223333",
  "Types": [
    "Software & Configuration Checks/Patch Management/Compliance"
  ],
  "CreatedAt": "2021-11-11T22:05:25Z",
```

```
  "UpdatedAt": "2021-11-11T22:05:25Z",
  "Severity": {
    "Label": "INFORMATIONAL",
    "Normalized": 0
  },
  "Title": "Systems Manager Patch Summary - Managed Instance Non-Compliant",
  "Description": "This AWS control checks whether each instance that is managed by AWS
 Systems Manager is in compliance with the rules of the patch baseline that applies to
 that instance when a compliance Scan runs.",
  "Remediation": {
    "Recommendation": {
      "Text": "For information about bringing instances into patch compliance, see
 'Remediating out-of-compliance instances (Patch Manager)'.",
      "Url": "https://docs.aws.amazon.com/systems-manager/latest/userguide/patch-
compliance-remediation.html"
    }
  },
  "SourceUrl": "https://us-east-2.console.aws.amazon.com/systems-manager/fleet-manager/
i-02573cafcfEXAMPLE/patch?region=us-east-2",
  "ProductFields": {
    "aws/securityhub/FindingId": "arn:aws:securityhub:us-east-2::product/aws/ssm-
patch-manager/arn:aws:patchmanager:us-east-2:111122223333:instance/i-02573cafcfEXAMPLE/
document/AWS-RunPatchBaseline/run-command/d710f5bd-04e3-47b4-82f6-df4e0EXAMPLE",
    "aws/securityhub/ProductName": "Systems Manager Patch Manager",
    "aws/securityhub/CompanyName": "AWS"
  },
  "Resources": [
    {
      "Type": "AwsEc2Instance",
      "Id": "i-02573cafcfEXAMPLE",
      "Partition": "aws",
      "Region": "us-east-2"
    }
  ],
  "WorkflowState": "NEW",
  "Workflow": {
    "Status": "NEW"
  },
  "RecordState": "ACTIVE",
  "PatchSummary": {
    "Id": "pb-0c10e65780EXAMPLE",
    "InstalledCount": 45,
    "MissingCount": 2,
    "FailedCount": 0,
```

```
    "InstalledOtherCount": 396,
    "InstalledRejectedCount": 0,
    "InstalledPendingReboot": 0,
    "OperationStartTime": "2021-11-11T22:05:06Z",
    "OperationEndTime": "2021-11-11T22:05:25Z",
    "RebootOption": "NoReboot",
    "Operation": "SCAN"
  }
}
```

打开并配置集成

要使用 Patch Manager与 Security Hub 的集成，必须打开 Security Hub。有关如何打开 Security Hub 的信息，请参阅 AWS Security Hub 用户指南中设置 Security Hub。

以下过程介绍如何集成 Patch Manager 和 Security Hub（如果 Security Hub 已处于活动状态，但 Patch Manager 集成处于关闭状态）。只有在手动关闭集成时，才需要完成此过程。

把 Patch Manager 添加到 Security Hub 集成

1. 在导航窗格中，选择 Patch Manager。
2. 选择设置选项卡。

   –或者–

   如果您是在当前 AWS 区域首次访问 Patch Manager，请选择从概览开始，然后选择设置选项卡。
3. 在导出到 Security Hub 部分（在补丁合规性结果未导出到 Security Hub 的右侧），选择启用。

如何停止发送调查发现

要停止向 Security Hub 发送结果，您可以使用 Security Hub 控制台或 API。

有关更多信息，请参阅《AWS Security Hub 用户指南》中的以下主题：

- 禁用和启用来自集成的结果流（控制台）
- 禁用来自集成（Security Hub API，AWS CLI）的结果流

使用 AWS CLI 处理 Patch Manager 资源使用

本节包含可用于为Patch Manager（AWS Systems Manager 中的一项工具）执行配置任务的 AWS Command Line Interface（AWS CLI）命令的示例。

有关通过 AWS CLI 使用自定义补丁基准对服务器环境进行修补的说明，请参阅教程：使用 AWS CLI 修补服务器环境。

有关使用 AWS CLI 完成 AWS Systems Manager 任务的更多信息，请参阅AWS CLI 命令参考的 AWS Systems Manager 部分。

主题

- AWS CLI 命令用于补丁基准
- 用于补丁组的 AWS CLI 命令
- AWS CLI 命令用于查看补丁摘要和详细信息
- 用于扫描和修补托管式节点的 AWS CLI 命令

AWS CLI 命令用于补丁基准

补丁基准的示例命令

- 创建补丁基准
- 创建对不同操作系统版本使用自定义存储库的补丁基准
- 更新补丁基准
- 重命名补丁基准
- 删除补丁基准
- 列出所有补丁基准
- 列出所有 AWS 提供的补丁基准
- 列出我的补丁基准
- 显示补丁基准
- 获取默认补丁基准
- 将自定义补丁基准设置为默认基准
- 将 AWS 补丁基准重置为默认基准
- 标记补丁基准
- 列出补丁基准的标签
- 从补丁基准删除标签

创建补丁基准

以下命令将创建一个补丁基准，该补丁基准将在 Windows Server 2012 R2 的关键和重要安全更新发布 5 天后批准所有这些更新。还为已批准和已拒绝的补丁列表指定了补丁。此外，补丁基准已标记来指示它适用于生产环境。

Linux & macOS

```
aws ssm create-patch-baseline \
    --name "Windows-Server-2012R2" \
    --tags "Key=Environment,Value=Production" \
    --description "Windows Server 2012 R2, Important and Critical security updates"
 \
    --approved-patches "KB2032276,MS10-048" \
    --rejected-patches "KB2124261" \
    --rejected-patches-action "ALLOW_AS_DEPENDENCY" \
    --approval-rules
 "PatchRules=[{PatchFilterGroup={PatchFilters=[{Key=MSRC_SEVERITY,Values=[Important,Critical
{Key=CLASSIFICATION,Values=SecurityUpdates},
{Key=PRODUCT,Values=WindowsServer2012R2}]},ApproveAfterDays=5}]"
```

Windows Server

```
aws ssm create-patch-baseline ^
    --name "Windows-Server-2012R2" ^
    --tags "Key=Environment,Value=Production" ^
    --description "Windows Server 2012 R2, Important and Critical security updates"
 ^
    --approved-patches "KB2032276,MS10-048" ^
    --rejected-patches "KB2124261" ^
    --rejected-patches-action "ALLOW_AS_DEPENDENCY" ^
    --approval-rules
 "PatchRules=[{PatchFilterGroup={PatchFilters=[{Key=MSRC_SEVERITY,Values=[Important,Critical
{Key=CLASSIFICATION,Values=SecurityUpdates},
{Key=PRODUCT,Values=WindowsServer2012R2}]},ApproveAfterDays=5}]"
```

系统将返回类似于以下内容的信息。

```
{
   "BaselineId":"pb-0c10e65780EXAMPLE"
}
```

创建对不同操作系统版本使用自定义存储库的补丁基准

仅适用于 Linux 托管式节点。以下命令显示如何指定要用于特定版本的 Amazon Linux 操作系统的补丁存储库。此示例使用 Amazon Linux 2017.09 上默认启用的源存储库，但可以调整为您为托管式节点配置的不同源存储库。

> **Note**
>
> 为了更好地说明这一更为复杂的命令，我们将 `--cli-input-json` 选项与存储在外部 JSON 文件中的其他选项结合使用。

1. 创建一个名称类似于 `my-patch-repository.json` 的 JSON 文件并将以下内容添加到其中。

```
{
    "Description": "My patch repository for Amazon Linux 2017.09",
    "Name": "Amazon-Linux-2017.09",
    "OperatingSystem": "AMAZON_LINUX",
    "ApprovalRules": {
        "PatchRules": [
            {
                "ApproveAfterDays": 7,
                "EnableNonSecurity": true,
                "PatchFilterGroup": {
                    "PatchFilters": [
                        {
                            "Key": "SEVERITY",
                            "Values": [
                                "Important",
                                "Critical"
                            ]
                        },
                        {
                            "Key": "CLASSIFICATION",
                            "Values": [
                                "Security",
                                "Bugfix"
                            ]
                        },
                        {
                            "Key": "PRODUCT",
                            "Values": [
```

```
                                "AmazonLinux2017.09"
                            ]
                        }
                    ]
                }
            }
        ]
    },
    "Sources": [
        {
            "Name": "My-AL2017.09",
            "Products": [
                "AmazonLinux2017.09"
            ],
            "Configuration": "[amzn-main] \nname=amzn-main-Base
\nmirrorlist=http://repo./$awsregion./$awsdomain//$releasever/main/
mirror.list //nmirrorlist_expire=300//nmetadata_expire=300 \npriority=10
 \nfailovermethod=priority \nfastestmirror_enabled=0 \ngpgcheck=1
 \ngpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-amazon-ga \nenabled=1 \nretries=3
 \ntimeout=5\nreport_instanceid=yes"
        }
    ]
}
```

2. 在保存文件的目录中，键入以下命令。

```
aws ssm create-patch-baseline --cli-input-json file://my-patch-repository.json
```

系统将返回类似于以下内容的信息。

```
{
    "BaselineId": "pb-0c10e65780EXAMPLE"
}
```

更新补丁基准

以下命令将两个带已拒绝状态的补丁和一个带已批准状态的补丁添加到现有补丁基准。

有关已批准的补丁和已拒绝的补丁列表的已接受格式的信息，请参阅 [已批准补丁和已拒绝补丁列表的程序包名称格式](#)。

Linux & macOS

```
aws ssm update-patch-baseline \
    --baseline-id pb-0c10e65780EXAMPLE \
    --rejected-patches "KB2032276" "MS10-048" \
    --approved-patches "KB2124261"
```

Windows Server

```
aws ssm update-patch-baseline ^
    --baseline-id pb-0c10e65780EXAMPLE ^
    --rejected-patches "KB2032276" "MS10-048" ^
    --approved-patches "KB2124261"
```

系统将返回类似于以下内容的信息。

```
{
   "BaselineId":"pb-0c10e65780EXAMPLE",
   "Name":"Windows-Server-2012R2",
   "RejectedPatches":[
      "KB2032276",
      "MS10-048"
   ],
   "GlobalFilters":{
      "PatchFilters":[

      ]
   },
   "ApprovalRules":{
      "PatchRules":[
         {
            "PatchFilterGroup":{
               "PatchFilters":[
                  {
                     "Values":[
                        "Important",
                        "Critical"
                     ],
                     "Key":"MSRC_SEVERITY"
                  },
                  {
                     "Values":[
```

```
                        "SecurityUpdates"
                     ],
                     "Key":"CLASSIFICATION"
                  },
                  {
                     "Values":[
                        "WindowsServer2012R2"
                     ],
                     "Key":"PRODUCT"
                  }
               ]
            },
            "ApproveAfterDays":5
         }
      ]
   },
   "ModifiedDate":1481001494.035,
   "CreatedDate":1480997823.81,
   "ApprovedPatches":[
      "KB2124261"
   ],
   "Description":"Windows Server 2012 R2, Important and Critical security updates"
}
```

重命名补丁基准

Linux & macOS

```
aws ssm update-patch-baseline \
    --baseline-id pb-0c10e65780EXAMPLE \
    --name "Windows-Server-2012-R2-Important-and-Critical-Security-Updates"
```

Windows Server

```
aws ssm update-patch-baseline ^
    --baseline-id pb-0c10e65780EXAMPLE ^
    --name "Windows-Server-2012-R2-Important-and-Critical-Security-Updates"
```

系统将返回类似于以下内容的信息。

```
{
   "BaselineId":"pb-0c10e65780EXAMPLE",
```

```
    "Name":"Windows-Server-2012-R2-Important-and-Critical-Security-Updates",
    "RejectedPatches":[
       "KB2032276",
       "MS10-048"
    ],
    "GlobalFilters":{
       "PatchFilters":[

       ]
    },
    "ApprovalRules":{
       "PatchRules":[
          {
             "PatchFilterGroup":{
                "PatchFilters":[
                   {
                      "Values":[
                         "Important",
                         "Critical"
                      ],
                      "Key":"MSRC_SEVERITY"
                   },
                   {
                      "Values":[
                         "SecurityUpdates"
                      ],
                      "Key":"CLASSIFICATION"
                   },
                   {
                      "Values":[
                         "WindowsServer2012R2"
                      ],
                      "Key":"PRODUCT"
                   }
                ]
             },
             "ApproveAfterDays":5
          }
       ]
    },
    "ModifiedDate":1481001795.287,
    "CreatedDate":1480997823.81,
    "ApprovedPatches":[
       "KB2124261"
```

```
   ],
   "Description":"Windows Server 2012 R2, Important and Critical security updates"
}
```

删除补丁基准

```
aws ssm delete-patch-baseline --baseline-id "pb-0c10e65780EXAMPLE"
```

系统将返回类似于以下内容的信息。

```
{
   "BaselineId":"pb-0c10e65780EXAMPLE"
}
```

列出所有补丁基准

```
aws ssm describe-patch-baselines
```

系统将返回类似于以下内容的信息。

```
{
   "BaselineIdentities":[
      {
         "BaselineName":"AWS-DefaultPatchBaseline",
         "DefaultBaseline":true,
         "BaselineDescription":"Default Patch Baseline Provided by AWS.",
         "BaselineId":"arn:aws:ssm:us-east-2:111122223333:patchbaseline/
pb-0c10e65780EXAMPLE"
      },
      {
         "BaselineName":"Windows-Server-2012R2",
         "DefaultBaseline":false,
         "BaselineDescription":"Windows Server 2012 R2, Important and Critical security
 updates",
         "BaselineId":"pb-0c10e65780EXAMPLE"
      }
   ]
}
```

以下是另一个命令，该命令将列出 AWS 区域 中的所有补丁基准。

Linux & macOS

```
aws ssm describe-patch-baselines \
    --region us-east-2 \
    --filters "Key=OWNER,Values=[All]"
```

Windows Server

```
aws ssm describe-patch-baselines ^
    --region us-east-2 ^
    --filters "Key=OWNER,Values=[All]"
```

系统将返回类似于以下内容的信息。

```
{
   "BaselineIdentities":[
      {
         "BaselineName":"AWS-DefaultPatchBaseline",
         "DefaultBaseline":true,
         "BaselineDescription":"Default Patch Baseline Provided by AWS.",
         "BaselineId":"arn:aws:ssm:us-east-2:111122223333:patchbaseline/
pb-0c10e65780EXAMPLE"
      },
      {
         "BaselineName":"Windows-Server-2012R2",
         "DefaultBaseline":false,
         "BaselineDescription":"Windows Server 2012 R2, Important and Critical security
 updates",
         "BaselineId":"pb-0c10e65780EXAMPLE"
      }
   ]
}
```

列出所有 AWS 提供的补丁基准

Linux & macOS

```
aws ssm describe-patch-baselines \
    --region us-east-2 \
    --filters "Key=OWNER,Values=[AWS]"
```

Windows Server

```
aws ssm describe-patch-baselines ^
    --region us-east-2 ^
    --filters "Key=OWNER,Values=[AWS]"
```

系统将返回类似于以下内容的信息。

```
{
   "BaselineIdentities":[
      {
         "BaselineName":"AWS-DefaultPatchBaseline",
         "DefaultBaseline":true,
         "BaselineDescription":"Default Patch Baseline Provided by AWS.",
         "BaselineId":"arn:aws:ssm:us-east-2:111122223333:patchbaseline/
pb-0c10e65780EXAMPLE"
      }
   ]
}
```

列出我的补丁基准

Linux & macOS

```
aws ssm describe-patch-baselines \
    --region us-east-2 \
    --filters "Key=OWNER,Values=[Self]"
```

Windows Server

```
aws ssm describe-patch-baselines ^
    --region us-east-2 ^
    --filters "Key=OWNER,Values=[Self]"
```

系统将返回类似于以下内容的信息。

```
{
   "BaselineIdentities":[
      {
```

```
        "BaselineName":"Windows-Server-2012R2",
        "DefaultBaseline":false,
        "BaselineDescription":"Windows Server 2012 R2, Important and Critical security
 updates",
        "BaselineId":"pb-0c10e65780EXAMPLE"
      }
   ]
}
```

显示补丁基准

```
aws ssm get-patch-baseline --baseline-id pb-0c10e65780EXAMPLE
```

> **Note**
>
> 对于自定义补丁基准，您可以指定补丁基准 ID 或完整的 Amazon Resource Name (ARN)。对于 AWS 提供的补丁基准，必须指定完整 ARN。例如 `arn:aws:ssm:us-east-2:075727635805:patchbaseline/pb-0c10e65780EXAMPLE`。

系统将返回类似于以下内容的信息。

```
{
   "BaselineId":"pb-0c10e65780EXAMPLE",
   "Name":"Windows-Server-2012R2",
   "PatchGroups":[
      "Web Servers"
   ],
   "RejectedPatches":[

   ],
   "GlobalFilters":{
      "PatchFilters":[

      ]
   },
   "ApprovalRules":{
      "PatchRules":[
         {
            "PatchFilterGroup":{
               "PatchFilters":[
```

```
                {
                   "Values":[
                      "Important",
                      "Critical"
                   ],
                   "Key":"MSRC_SEVERITY"
                },
                {
                   "Values":[
                      "SecurityUpdates"
                   ],
                   "Key":"CLASSIFICATION"
                },
                {
                   "Values":[
                      "WindowsServer2012R2"
                   ],
                   "Key":"PRODUCT"
                }
             ]
          },
          "ApproveAfterDays":5
       }
    ]
 },
 "ModifiedDate":1480997823.81,
 "CreatedDate":1480997823.81,
 "ApprovedPatches":[

 ],
 "Description":"Windows Server 2012 R2, Important and Critical security updates"
}
```

获取默认补丁基准

```
aws ssm get-default-patch-baseline --region us-east-2
```

系统将返回类似于以下内容的信息。

```
{
   "BaselineId":"arn:aws:ssm:us-east-2:111122223333:patchbaseline/pb-0c10e65780EXAMPLE"
}
```

将自定义补丁基准设置为默认基准

Linux & macOS

```
aws ssm register-default-patch-baseline \
    --region us-east-2 \
    --baseline-id "pb-0c10e65780EXAMPLE"
```

Windows Server

```
aws ssm register-default-patch-baseline ^
    --region us-east-2 ^
    --baseline-id "pb-0c10e65780EXAMPLE"
```

系统将返回类似于以下内容的信息。

```
{
   "BaselineId":"pb-0c10e65780EXAMPLE"
}
```

将 AWS 补丁基准重置为默认基准

Linux & macOS

```
aws ssm register-default-patch-baseline \
    --region us-east-2 \
    --baseline-id "arn:aws:ssm:us-east-2:123456789012:patchbaseline/
pb-0c10e65780EXAMPLE"
```

Windows Server

```
aws ssm register-default-patch-baseline ^
    --region us-east-2 ^
    --baseline-id "arn:aws:ssm:us-east-2:123456789012:patchbaseline/
pb-0c10e65780EXAMPLE"
```

系统将返回类似于以下内容的信息。

```
{
```

```
    "BaselineId":"pb-0c10e65780EXAMPLE"
}
```

标记补丁基准

Linux & macOS

```
aws ssm add-tags-to-resource \
    --resource-type "PatchBaseline" \
    --resource-id "pb-0c10e65780EXAMPLE" \
    --tags "Key=Project,Value=Testing"
```

Windows Server

```
aws ssm add-tags-to-resource ^
    --resource-type "PatchBaseline" ^
    --resource-id "pb-0c10e65780EXAMPLE" ^
    --tags "Key=Project,Value=Testing"
```

列出补丁基准的标签

Linux & macOS

```
aws ssm list-tags-for-resource \
    --resource-type "PatchBaseline" \
    --resource-id "pb-0c10e65780EXAMPLE"
```

Windows Server

```
aws ssm list-tags-for-resource ^
    --resource-type "PatchBaseline" ^
    --resource-id "pb-0c10e65780EXAMPLE"
```

从补丁基准删除标签

Linux & macOS

```
aws ssm remove-tags-from-resource \
    --resource-type "PatchBaseline" \
```

```
    --resource-id "pb-0c10e65780EXAMPLE" \
    --tag-keys "Project"
```

Windows Server

```
aws ssm remove-tags-from-resource ^
    --resource-type "PatchBaseline" ^
    --resource-id "pb-0c10e65780EXAMPLE" ^
    --tag-keys "Project"
```

用于补丁组的 AWS CLI 命令

补丁组的示例命令

- 创建补丁组
- 将补丁组“Web 服务器”注册到补丁基准
- 将补丁组“后端”注册到 AWS 提供的补丁基准
- 显示补丁组注册
- 从补丁基准取消注册补丁组

创建补丁组

> **Note**
>
> 补丁组不会用于基于补丁策略的修补操作。有关使用补丁策略的更多信息，请参阅 Quick Setup 中的补丁策略配置。

为帮助您组织修补工作，我们建议您使用标签将托管式节点添加到补丁组。补丁组需要使用标签键 `Patch Group` 或 `PatchGroup`。如果在 EC2 实例元数据中允许使用标签，则必须使用 `PatchGroup`（不带空格）。您可以指定任何标签值，但标签键必须为 `Patch Group` 或 `PatchGroup`。有关补丁组的更多信息，请参阅 补丁组。

使用标签对托管式节点进行分组后，请将补丁组值添加到补丁基准。通过将补丁组注册到补丁基准，您可以确保在修补操作期间安装正确的补丁。

任务 1：请使用标签将 EC2 实例添加到补丁组

> **Note**
>
> 使用 Amazon Elastic Compute Cloud (Amazon EC2) 控制台和 AWS CLI，则可以应用 `Key = Patch Group` 或 `Key = PatchGroup` 标签到尚未配置为与 Systems Manager 一起使用的实例。如果在应用 `Patch Group` 或 `Key = PatchGroup` 标签之后，未列出您希望在 Patch Manager 中看到的 EC2 实例，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

运行以下命令将 `PatchGroup` 标签添加到 EC2 实例。

```
aws ec2 create-tags --resources "i-1234567890abcdef0" --tags
 "Key=PatchGroup,Value=GroupValue"
```

任务 2：使用标签将托管式节点添加到补丁组

运行以下命令以将 `PatchGroup` 标签添加到托管式节点。

Linux & macOS

```
aws ssm add-tags-to-resource \
    --resource-type "ManagedInstance" \
    --resource-id "mi-0123456789abcdefg" \
    --tags "Key=PatchGroup,Value=GroupValue"
```

Windows Server

```
aws ssm add-tags-to-resource ^
    --resource-type "ManagedInstance" ^
    --resource-id "mi-0123456789abcdefg" ^
    --tags "Key=PatchGroup,Value=GroupValue"
```

任务 3：将补丁组添加到补丁基准

运行以下命令将 `PatchGroup` 标签值关联到指定的补丁基准。

Linux & macOS

```
aws ssm register-patch-baseline-for-patch-group \
```

```
    --baseline-id "pb-0c10e65780EXAMPLE" \
    --patch-group "Development"
```

Windows Server

```
aws ssm register-patch-baseline-for-patch-group ^
    --baseline-id "pb-0c10e65780EXAMPLE" ^
    --patch-group "Development"
```

系统将返回类似于以下内容的信息。

```
{
  "PatchGroup": "Development",
  "BaselineId": "pb-0c10e65780EXAMPLE"
}
```

将补丁组“Web 服务器”注册到补丁基准

Linux & macOS

```
aws ssm register-patch-baseline-for-patch-group \
    --baseline-id "pb-0c10e65780EXAMPLE" \
    --patch-group "Web Servers"
```

Windows Server

```
aws ssm register-patch-baseline-for-patch-group ^
    --baseline-id "pb-0c10e65780EXAMPLE" ^
    --patch-group "Web Servers"
```

系统将返回类似于以下内容的信息。

```
{
   "PatchGroup":"Web Servers",
   "BaselineId":"pb-0c10e65780EXAMPLE"
}
```

将补丁组“后端”注册到 AWS 提供的补丁基准

Linux & macOS

```
aws ssm register-patch-baseline-for-patch-group \
    --region us-east-2 \
    --baseline-id "arn:aws:ssm:us-east-2:111122223333:patchbaseline/
pb-0c10e65780EXAMPLE" \
    --patch-group "Backend"
```

Windows Server

```
aws ssm register-patch-baseline-for-patch-group ^
    --region us-east-2 ^
    --baseline-id "arn:aws:ssm:us-east-2:111122223333:patchbaseline/
pb-0c10e65780EXAMPLE" ^
    --patch-group "Backend"
```

系统将返回类似于以下内容的信息。

```
{
   "PatchGroup":"Backend",
   "BaselineId":"arn:aws:ssm:us-east-2:111122223333:patchbaseline/pb-0c10e65780EXAMPLE"
}
```

显示补丁组注册

```
aws ssm describe-patch-groups --region us-east-2
```

系统将返回类似于以下内容的信息。

```
{
   "PatchGroupPatchBaselineMappings":[
      {
         "PatchGroup":"Backend",
         "BaselineIdentity":{
            "BaselineName":"AWS-DefaultPatchBaseline",
            "DefaultBaseline":false,
            "BaselineDescription":"Default Patch Baseline Provided by AWS.",
            "BaselineId":"arn:aws:ssm:us-east-2:111122223333:patchbaseline/
pb-0c10e65780EXAMPLE"
```

```
         }
      },
      {
         "PatchGroup":"Web Servers",
         "BaselineIdentity":{
            "BaselineName":"Windows-Server-2012R2",
            "DefaultBaseline":true,
            "BaselineDescription":"Windows Server 2012 R2, Important and Critical
 updates",
            "BaselineId":"pb-0c10e65780EXAMPLE"
         }
      }
   ]
}
```

从补丁基准取消注册补丁组

Linux & macOS

```
aws ssm deregister-patch-baseline-for-patch-group \
    --region us-east-2 \
    --patch-group "Production" \
    --baseline-id "arn:aws:ssm:us-east-2:111122223333:patchbaseline/
pb-0c10e65780EXAMPLE"
```

Windows Server

```
aws ssm deregister-patch-baseline-for-patch-group ^
    --region us-east-2 ^
    --patch-group "Production" ^
    --baseline-id "arn:aws:ssm:us-east-2:111122223333:patchbaseline/
pb-0c10e65780EXAMPLE"
```

系统将返回类似于以下内容的信息。

```
{
   "PatchGroup":"Production",
   "BaselineId":"arn:aws:ssm:us-east-2:111122223333:patchbaseline/pb-0c10e65780EXAMPLE"
}
```

AWS CLI 命令用于查看补丁摘要和详细信息

用于查看补丁摘要和详细信息的示例命令

- 获取补丁基准定义的所有补丁
- 获取 Amazonux2018.03 的所有分类为 SECURITY 、严重性为 Critical 的补丁
- 获取 Windows Server 2012 的所有补丁 ，其 MSRC 严重性为 Critical
- 获取所有可用补丁
- 获取每个托管式节点的补丁摘要状态
- 获取托管式节点的补丁合规性详细信息
- 查看修补合规性结果 (AWS CLI)

获取补丁基准定义的所有补丁

> **Note**
>
> 此命令仅受 Windows Server 补丁基准的支持。

Linux & macOS

```
aws ssm describe-effective-patches-for-patch-baseline \
    --region us-east-2 \
    --baseline-id "pb-0c10e65780EXAMPLE"
```

Windows Server

```
aws ssm describe-effective-patches-for-patch-baseline ^
    --region us-east-2 ^
    --baseline-id "pb-0c10e65780EXAMPLE"
```

系统将返回类似于以下内容的信息。

```
{
   "NextToken":"--token string truncated--",
   "EffectivePatches":[
      {
         "PatchStatus":{
```

```
            "ApprovalDate":1384711200.0,
            "DeploymentStatus":"APPROVED"
         },
         "Patch":{
            "ContentUrl":"https://support.microsoft.com/en-us/kb/2876331",
            "ProductFamily":"Windows",
            "Product":"WindowsServer2012R2",
            "Vendor":"Microsoft",
            "Description":"A security issue has been identified in a Microsoft
software
               product that could affect your system. You can help protect your system
               by installing this update from Microsoft. For a complete listing of the
               issues that are included in this update, see the associated Microsoft
               Knowledge Base article. After you install this update, you may have to
               restart your system.",
            "Classification":"SecurityUpdates",
            "Title":"Security Update for Windows Server 2012 R2 Preview (KB2876331)",
            "ReleaseDate":1384279200.0,
            "MsrcClassification":"Critical",
            "Language":"All",
            "KbNumber":"KB2876331",
            "MsrcNumber":"MS13-089",
            "Id":"e74ccc76-85f0-4881-a738-59e9fc9a336d"
         }
      },
      {
         "PatchStatus":{
            "ApprovalDate":1428858000.0,
            "DeploymentStatus":"APPROVED"
         },
         "Patch":{
            "ContentUrl":"https://support.microsoft.com/en-us/kb/2919355",
            "ProductFamily":"Windows",
            "Product":"WindowsServer2012R2",
            "Vendor":"Microsoft",
            "Description":"Windows Server 2012 R2 Update is a cumulative
               set of security updates, critical updates and updates. You
               must install Windows Server 2012 R2 Update to ensure that
               your computer can continue to receive future Windows Updates,
               including security updates. For a complete listing of the
               issues that are included in this update, see the associated
               Microsoft Knowledge Base article for more information. After
               you install this item, you may have to restart your computer.",
            "Classification":"SecurityUpdates",
```

```
            "Title":"Windows Server 2012 R2 Update (KB2919355)",
            "ReleaseDate":1428426000.0,
            "MsrcClassification":"Critical",
            "Language":"All",
            "KbNumber":"KB2919355",
            "MsrcNumber":"MS14-018",
            "Id":"8452bac0-bf53-4fbd-915d-499de08c338b"
         }
      }
     ---output truncated---
```

获取 Amazonux2018.03 的所有分类为 **SECURITY** 、严重性为 **Critical** 的补丁

Linux & macOS

```
aws ssm describe-available-patches \
    --region us-east-2 \
    --filters Key=PRODUCT,Values=AmazonLinux2018.03 Key=SEVERITY,Values=Critical
```

Windows Server

```
aws ssm describe-available-patches ^
    --region us-east-2 ^
    --filters Key=PRODUCT,Values=AmazonLinux2018.03 Key=SEVERITY,Values=Critical
```

系统将返回类似于以下内容的信息。

```
{
    "Patches": [
        {
            "AdvisoryIds": ["ALAS-2011-1"],
            "BugzillaIds": [ "1234567" ],
            "Classification": "SECURITY",
            "CVEIds": [ "CVE-2011-3192"],
            "Name": "zziplib",
            "Epoch": "0",
            "Version": "2.71",
            "Release": "1.3.amzn1",
            "Arch": "i686",
            "Product": "AmazonLinux2018.03",
            "ReleaseDate": 1590519815,
            "Severity": "CRITICAL"
```

```
        }
    ]
}
---output truncated---
```

获取 Windows Server 2012 的所有补丁，其 MSRC 严重性为 **Critical**

Linux & macOS

```
aws ssm describe-available-patches \
    --region us-east-2 \
    --filters Key=PRODUCT,Values=WindowsServer2012 Key=MSRC_SEVERITY,Values=Critical
```

Windows Server

```
aws ssm describe-available-patches ^
    --region us-east-2 ^
    --filters Key=PRODUCT,Values=WindowsServer2012 Key=MSRC_SEVERITY,Values=Critical
```

系统将返回类似于以下内容的信息。

```
{
   "Patches":[
      {
         "ContentUrl":"https://support.microsoft.com/en-us/kb/2727528",
         "ProductFamily":"Windows",
         "Product":"WindowsServer2012",
         "Vendor":"Microsoft",
         "Description":"A security issue has been identified that could
           allow an unauthenticated remote attacker to compromise your
           system and gain control over it. You can help protect your
           system by installing this update from Microsoft. After you
           install this update, you may have to restart your system.",
         "Classification":"SecurityUpdates",
         "Title":"Security Update for Windows Server 2012 (KB2727528)",
         "ReleaseDate":1352829600.0,
         "MsrcClassification":"Critical",
         "Language":"All",
         "KbNumber":"KB2727528",
         "MsrcNumber":"MS12-072",
         "Id":"1eb507be-2040-4eeb-803d-abc55700b715"
      },
```

```
      {
         "ContentUrl":"https://support.microsoft.com/en-us/kb/2729462",
         "ProductFamily":"Windows",
         "Product":"WindowsServer2012",
         "Vendor":"Microsoft",
         "Description":"A security issue has been identified that could
           allow an unauthenticated remote attacker to compromise your
           system and gain control over it. You can help protect your
           system by installing this update from Microsoft. After you
           install this update, you may have to restart your system.",
         "Classification":"SecurityUpdates",
         "Title":"Security Update for Microsoft .NET Framework 3.5 on
           Windows 8 and Windows Server 2012 for x64-based Systems (KB2729462)",
         "ReleaseDate":1352829600.0,
         "MsrcClassification":"Critical",
         "Language":"All",
         "KbNumber":"KB2729462",
         "MsrcNumber":"MS12-074",
         "Id":"af873760-c97c-4088-ab7e-5219e120eab4"
      }

---output truncated---
```

获取所有可用补丁

```
aws ssm describe-available-patches --region us-east-2
```

系统将返回类似于以下内容的信息。

```
{
   "NextToken":"--token string truncated--",
   "Patches":[
      {
         "ContentUrl":"https://support.microsoft.com/en-us/kb/2032276",
         "ProductFamily":"Windows",
         "Product":"WindowsServer2008R2",
         "Vendor":"Microsoft",
         "Description":"A security issue has been identified that could allow an
           unauthenticated remote attacker to compromise your system and gain
           control over it. You can help protect your system by installing this
           update from Microsoft. After you install this update, you may have to
           restart your system.",
         "Classification":"SecurityUpdates",
```

```
         "Title":"Security Update for Windows Server 2008 R2 x64 Edition (KB2032276)",
         "ReleaseDate":1279040400.0,
         "MsrcClassification":"Important",
         "Language":"All",
         "KbNumber":"KB2032276",
         "MsrcNumber":"MS10-043",
         "Id":"8692029b-a3a2-4a87-a73b-8ea881b4b4d6"
      },
      {
         "ContentUrl":"https://support.microsoft.com/en-us/kb/2124261",
         "ProductFamily":"Windows",
         "Product":"Windows7",
         "Vendor":"Microsoft",
         "Description":"A security issue has been identified that could allow
           an unauthenticated remote attacker to compromise your system and gain
           control over it. You can help protect your system by installing this
           update from Microsoft. After you install this update, you may have
           to restart your system.",
         "Classification":"SecurityUpdates",
         "Title":"Security Update for Windows 7 (KB2124261)",
         "ReleaseDate":1284483600.0,
         "MsrcClassification":"Important",
         "Language":"All",
         "KbNumber":"KB2124261",
         "MsrcNumber":"MS10-065",
         "Id":"12ef1bed-0dd2-4633-b3ac-60888aa8ba33"
      }
      ---output truncated---
```

获取每个托管式节点的补丁摘要状态

每个托管式节点的摘要可为您提供每个节点中具有以下状态的补丁数量：“NotApplicable”（不适用）、“Missing”（缺少）、“Failed”（失败）、“InstalledOther”（已安装其他）和“Installed”（已安装）。

Linux & macOS

```
aws ssm describe-instance-patch-states \
    --instance-ids i-08ee91c0b17045407 i-09a618aec652973a9
```

Windows Server

```
aws ssm describe-instance-patch-states ^
```

```
    --instance-ids i-08ee91c0b17045407 i-09a618aec652973a9
```

系统将返回类似于以下内容的信息。

```
{
   "InstancePatchStates":[
      {
            "InstanceId": "i-08ee91c0b17045407",
            "PatchGroup": "",
            "BaselineId": "pb-0c10e65780EXAMPLE",
            "SnapshotId": "6d03d6c5-f79d-41d0-8d0e-00a9aEXAMPLE",
            "InstalledCount": 50,
            "InstalledOtherCount": 353,
            "InstalledPendingRebootCount": 0,
            "InstalledRejectedCount": 0,
            "MissingCount": 0,
            "FailedCount": 0,
            "UnreportedNotApplicableCount": -1,
            "NotApplicableCount": 671,
            "OperationStartTime": "2020-01-24T12:37:56-08:00",
            "OperationEndTime": "2020-01-24T12:37:59-08:00",
            "Operation": "Scan",
            "RebootOption": "NoReboot"
        },
        {
            "InstanceId": "i-09a618aec652973a9",
            "PatchGroup": "",
            "BaselineId": "pb-0c10e65780EXAMPLE",
            "SnapshotId": "c7e0441b-1eae-411b-8aa7-973e6EXAMPLE",
            "InstalledCount": 36,
            "InstalledOtherCount": 396,
            "InstalledPendingRebootCount": 0,
            "InstalledRejectedCount": 0,
            "MissingCount": 3,
            "FailedCount": 0,
            "UnreportedNotApplicableCount": -1,
            "NotApplicableCount": 420,
            "OperationStartTime": "2020-01-24T12:37:34-08:00",
            "OperationEndTime": "2020-01-24T12:37:37-08:00",
            "Operation": "Scan",
            "RebootOption": "NoReboot"
        }
```

```
     ---output truncated---
```

获取托管式节点的补丁合规性详细信息

```
aws ssm describe-instance-patches --instance-id i-08ee91c0b17045407
```

系统将返回类似于以下内容的信息。

```
{
   "NextToken":"--token string truncated--",
   "Patches":[
      {
            "Title": "bind-libs.x86_64:32:9.8.2-0.68.rc1.60.amzn1",
            "KBId": "bind-libs.x86_64",
            "Classification": "Security",
            "Severity": "Important",
            "State": "Installed",
            "InstalledTime": "2019-08-26T11:05:24-07:00"
        },
        {
            "Title": "bind-utils.x86_64:32:9.8.2-0.68.rc1.60.amzn1",
            "KBId": "bind-utils.x86_64",
            "Classification": "Security",
            "Severity": "Important",
            "State": "Installed",
            "InstalledTime": "2019-08-26T11:05:32-07:00"
        },
        {
            "Title": "dhclient.x86_64:12:4.1.1-53.P1.28.amzn1",
            "KBId": "dhclient.x86_64",
            "Classification": "Security",
            "Severity": "Important",
            "State": "Installed",
            "InstalledTime": "2019-08-26T11:05:31-07:00"
        },
    ---output truncated---
```

查看修补合规性结果 (AWS CLI)

查看单个托管式节点的补丁合规性结果

在 AWS Command Line Interface (AWS CLI) 中运行以下命令，查看单个托管式节点的补丁合规性结果。

```
aws ssm describe-instance-patch-states --instance-id instance-id
```

以 i-02573cafcfEXAMPLE 或 mi-0282f7c436EXAMPLE 格式，将 *instance-id* 替换为要查看其结果的托管式节点的 ID。

系统将返回类似于以下内容的信息。

```
{
    "InstancePatchStates": [
        {
            "InstanceId": "i-02573cafcfEXAMPLE",
            "PatchGroup": "mypatchgroup",
            "BaselineId": "pb-0c10e65780EXAMPLE",
            "SnapshotId": "a3f5ff34-9bc4-4d2c-a665-4d1c1EXAMPLE",
            "CriticalNonCompliantCount": 2,
            "SecurityNonCompliantCount": 2,
            "OtherNonCompliantCount": 1,
            "InstalledCount": 123,
            "InstalledOtherCount": 334,
            "InstalledPendingRebootCount": 0,
            "InstalledRejectedCount": 0,
            "MissingCount": 1,
            "FailedCount": 2,
            "UnreportedNotApplicableCount": 11,
            "NotApplicableCount": 2063,
            "OperationStartTime": "2021-05-03T11:00:56-07:00",
            "OperationEndTime": "2021-05-03T11:01:09-07:00",
            "Operation": "Scan",
            "LastNoRebootInstallOperationTime": "2020-06-14T12:17:41-07:00",
            "RebootOption": "RebootIfNeeded"
        }
    ]
}
```

查看某个区域中所有 EC2 实例的补丁计数摘要

describe-instance-patch-states 支持一次只检索一个托管实例的结果。但是，使用自定义脚本与 describe-instance-patch-states 命令，您可以生成更精细的报告。

例如，如果在本地计算机上安装 jq 筛选工具，您可以运行以下命令来识别哪些 EC2 实例在特定 AWS 区域 中的状态为 InstalledPendingReboot。

```
aws ssm describe-instance-patch-states \
    --instance-ids $(aws ec2 describe-instances --region region | jq
 '.Reservations[].Instances[] | .InstanceId' | tr '\n|"' ' ') \
    --output text --query 'InstancePatchStates[*].{Instance:InstanceId,
 InstalledPendingRebootCount:InstalledPendingRebootCount}'
```

region 表示 AWS Systems Manager 支持的 AWS 区域 的标识符，例如 us-east-2 对应美国东部（俄亥俄）区域。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

例如：

```
aws ssm describe-instance-patch-states \
    --instance-ids $(aws ec2 describe-instances --region us-east-2 | jq
 '.Reservations[].Instances[] | .InstanceId' | tr '\n|"' ' ') \
    --output text --query 'InstancePatchStates[*].{Instance:InstanceId,
 InstalledPendingRebootCount:InstalledPendingRebootCount}'
```

系统将返回类似于以下内容的信息。

```
1       i-02573cafcfEXAMPLE
0       i-0471e04240EXAMPLE
3       i-07782c72faEXAMPLE
6       i-083b678d37EXAMPLE
0       i-03a530a2d4EXAMPLE
1       i-01f68df0d0EXAMPLE
0       i-0a39c0f214EXAMPLE
7       i-0903a5101eEXAMPLE
7       i-03823c2fedEXAMPLE
```

除了 InstalledPendingRebootCount，您可以搜索的计数类型列表包括以下内容：

- CriticalNonCompliantCount
- SecurityNonCompliantCount
- OtherNonCompliantCount
- UnreportedNotApplicableCount
- InstalledPendingRebootCount
- FailedCount
- NotApplicableCount

- InstalledRejectedCount
- InstalledOtherCount
- MissingCount
- InstalledCount

用于扫描和修补托管式节点的 AWS CLI 命令

运行以下命令来扫描补丁合规性或安装补丁后，可以使用 AWS CLI 命令用于查看补丁摘要和详细信息 部分里的命令来查看有关补丁状态和合规性的信息。

示例命令

- 扫描托管式节点以检查是否符合补丁合规性要求 (AWS CLI)
- 在托管式节点上安装补丁 (AWS CLI)

扫描托管式节点以检查是否符合补丁合规性要求 (AWS CLI)

扫描指定托管式节点以检查是否符合补丁合规性要求

运行以下命令。

Linux & macOS

```
aws ssm send-command \
    --document-name 'AWS-RunPatchBaseline' \
    --targets Key=InstanceIds,Values='i-02573cafcfEXAMPLE,i-0471e04240EXAMPLE' \
    --parameters 'Operation=Scan' \
    --timeout-seconds 600
```

Windows Server

```
aws ssm send-command ^
    --document-name "AWS-RunPatchBaseline" ^
    --targets Key=InstanceIds,Values="i-02573cafcfEXAMPLE,i-0471e04240EXAMPLE" ^
    --parameters "Operation=Scan" ^
    --timeout-seconds 600
```

系统将返回类似于以下内容的信息。

```
{
    "Command": {
        "CommandId": "a04ed06c-8545-40f4-87c2-a0babEXAMPLE",
        "DocumentName": "AWS-RunPatchBaseline",
        "DocumentVersion": "$DEFAULT",
        "Comment": "",
        "ExpiresAfter": 1621974475.267,
        "Parameters": {
            "Operation": [
                "Scan"
            ]
        },
        "InstanceIds": [],
        "Targets": [
            {
                "Key": "InstanceIds",
                "Values": [
                    "i-02573cafcfEXAMPLE,
                     i-0471e04240EXAMPLE"
                ]
            }
        ],
        "RequestedDateTime": 1621952275.267,
        "Status": "Pending",
        "StatusDetails": "Pending",
        "TimeoutSeconds": 600,

    ---output truncated---

    }
}
```

按补丁组标签扫描托管式节点以检查是否符合补丁合规性要求

运行以下命令。

Linux & macOS

```
aws ssm send-command \
    --document-name 'AWS-RunPatchBaseline' \
    --targets Key='tag:PatchGroup',Values='Web servers' \
    --parameters 'Operation=Scan' \
```

```
    --timeout-seconds 600
```

Windows Server

```
aws ssm send-command ^
    --document-name "AWS-RunPatchBaseline" ^
    --targets Key="tag:PatchGroup",Values="Web servers" ^
    --parameters "Operation=Scan" ^
    --timeout-seconds 600
```

系统将返回类似于以下内容的信息。

```
{
    "Command": {
        "CommandId": "87a448ee-8adc-44e0-b4d1-6b429EXAMPLE",
        "DocumentName": "AWS-RunPatchBaseline",
        "DocumentVersion": "$DEFAULT",
        "Comment": "",
        "ExpiresAfter": 1621974983.128,
        "Parameters": {
            "Operation": [
                "Scan"
            ]
        },
        "InstanceIds": [],
        "Targets": [
            {
                "Key": "tag:PatchGroup",
                "Values": [
                    "Web servers"
                ]
            }
        ],
        "RequestedDateTime": 1621952783.128,
        "Status": "Pending",
        "StatusDetails": "Pending",
        "TimeoutSeconds": 600,

    ---output truncated---

    }
}
```

在托管式节点上安装补丁 (AWS CLI)

在指定托管式节点上安装补丁

运行以下命令。

> **Note**
>
> 目标托管式节点按需重启以完成补丁安装。有关更多信息，请参阅 用于修补的 SSM 命令文档：AWS-RunPatchBaseline。

Linux & macOS

```
aws ssm send-command \
    --document-name 'AWS-RunPatchBaseline' \
    --targets Key=InstanceIds,Values='i-02573cafcfEXAMPLE,i-0471e04240EXAMPLE' \
    --parameters 'Operation=Install' \
    --timeout-seconds 600
```

Windows Server

```
aws ssm send-command ^
    --document-name "AWS-RunPatchBaseline" ^
    --targets Key=InstanceIds,Values="i-02573cafcfEXAMPLE,i-0471e04240EXAMPLE" ^
    --parameters "Operation=Install" ^
    --timeout-seconds 600
```

系统将返回类似于以下内容的信息。

```
{
    "Command": {
        "CommandId": "5f403234-38c4-439f-a570-93623EXAMPLE",
        "DocumentName": "AWS-RunPatchBaseline",
        "DocumentVersion": "$DEFAULT",
        "Comment": "",
        "ExpiresAfter": 1621975301.791,
        "Parameters": {
            "Operation": [
                "Install"
            ]
```

```
        },
        "InstanceIds": [],
        "Targets": [
            {
                "Key": "InstanceIds",
                "Values": [
                    "i-02573cafcfEXAMPLE,
                     i-0471e04240EXAMPLE"
                ]
            }
        ],
        "RequestedDateTime": 1621953101.791,
        "Status": "Pending",
        "StatusDetails": "Pending",
        "TimeoutSeconds": 600,

    ---output truncated---

    }
}
```

在指定补丁组中的托管式节点上安装补丁

运行以下命令。

Linux & macOS

```
aws ssm send-command \
    --document-name 'AWS-RunPatchBaseline' \
    --targets Key='tag:PatchGroup',Values='Web servers' \
    -parameters 'Operation=Install' \
    --timeout-seconds 600
```

Windows Server

```
aws ssm send-command ^
    --document-name "AWS-RunPatchBaseline" ^
    --targets Key="tag:PatchGroup",Values="Web servers" ^
    --parameters "Operation=Install" ^
    --timeout-seconds 600
```

系统将返回类似于以下内容的信息。

```
{
    "Command": {
        "CommandId": "fa44b086-7d36-4ad5-ac8d-627ecEXAMPLE",
        "DocumentName": "AWS-RunPatchBaseline",
        "DocumentVersion": "$DEFAULT",
        "Comment": "",
        "ExpiresAfter": 1621975407.865,
        "Parameters": {
            "Operation": [
                "Install"
            ]
        },
        "InstanceIds": [],
        "Targets": [
            {
                "Key": "tag:PatchGroup",
                "Values": [
                    "Web servers"
                ]
            }
        ],
        "RequestedDateTime": 1621953207.865,
        "Status": "Pending",
        "StatusDetails": "Pending",
        "TimeoutSeconds": 600,

    ---output truncated---

    }
}
```

AWS Systems Manager Patch Manager 教程

本节中的教程演示如何针对各种修补场景使用Patch Manager（AWS Systems Manager 中的一项工具）。

主题

- 教程：使用控制台创建用于安装 Windows Service Pack 的补丁基准
- 教程：使用控制台更新应用程序依赖项、修补托管式节点并执行特定于应用程序的运行状况检查
- 教程：使用 AWS CLI 修补服务器环境

教程：使用控制台创建用于安装 Windows Service Pack 的补丁基准

创建自定义补丁基准时，可以指定安装所有、部分或仅安装一种支持的补丁。

在适用于 Windows 的补丁基准中，您可以选择 `ServicePacks` 作为唯一的 Classification (分类) 选项，以便将补丁更新仅限于 Service Pack。Patch Manager（AWS Systems Manager 中的一项工具）可以自动安装服务包，前提是此更新在 Windows Update 或 Windows Server Update Services（WSUS）中可用。

您可以配置补丁基准，以控制是安装所有 Windows 版本的 Service Pack，还是仅安装特定版本（如 Windows 7 或 Windows Server 2016）的 Service Pack。

使用以下过程创建自定义补丁基准，专门用于在 Windows 托管式节点上安装所有 Service Pack。

创建用于安装 Windows Service Pack（控制台）的补丁基准

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Patch Manager。
3. 选择补丁基准选项卡，然后选择创建补丁基准。
4. 在 Name (名称) 中，输入新补丁基准的名称，例如，`MyWindowsServicePackPatchBaseline`。
5. （可选）对于 Description (描述)，输入此补丁基准的描述。
6. 对于 Operating system（操作系统），请选择 `Windows`。
7. 如果要在创建后即开始将此补丁基准用作 Windows 的默认项，请选择 Set this patch baseline as the default patch baseline for Windows Server instances（将此补丁基准设置为 Windows Server 实例的默认补丁基准）。

   > **Note**
   >
   > 只有在 2022 年 12 月 22 日 补丁策略 发布之前首次访问 Patch Manager时，此选项才可用。
   > 有关将现有补丁基准设置为默认项的信息，请参阅 将现有补丁基准设置为默认项。

8. 在 Approval rules for operating systems (操作系统的批准规则) 部分，使用字段创建一个或多个自动批准规则。

   - 产品：批准规则适用的操作系统版本，例如 `WindowsServer2012`。您可以选择一个、多个或所有受支持的 Windows 版本。默认选择为 `All`。

- Classification (分类)：选择 `ServicePacks`。
- Severity (严重性)：规则适用于的补丁的严重性值。要确保规则包含所有 Service Pack，请选择 `All`。
- Auto-approval（自动批准）：选择要自动批准的补丁的方法。
  - 在指定的天数后批准补丁：Patch Manager 在发布或更新补丁之后等待的天数，然后自动批准补丁。可以输入零 (0) 到 360 的任何整数。对于大多数情况，我们建议等待期不超过 100 天。
  - 批准在特定日期之前发布的补丁：补丁发布日期，Patch Manager 自动应用在该日期或之前发布或更新的所有补丁。例如，如果指定 2023 年 7 月 7 日，则不会自动安装在 2023 年 7 月 8 日或之后发布或最后更新的任何补丁。
- （可选）Compliance reporting (合规性报告)：要分配给由基准批准的 Service Pack 的严重性级别，例如 `High`。

> **Note**
>
> 如果您指定合规性报告级别以及任何已批准 Service Pack 的补丁状态报告为 `Missing`，则补丁基准报告的总体合规性严重性级别就是您指定的严重级别。

9. （可选）对于管理标签，将一个或多个标签键名称/值对应用到补丁基准。

   标签是您分配给资源的可选元数据。标签允许您按各种标准（如用途、所有者或环境）对资源进行分类。对于专用于更新 Service Pack 的此补丁基准，您可以指定键值对，如下所示：

   - `Key=OS,Value=Windows`
   - `Key=Classification,Value=ServicePacks`

10. 请选择创建补丁基准。

教程：使用控制台更新应用程序依赖项、修补托管式节点并执行特定于应用程序的运行状况检查

在许多情况下，在使用最新软件更新修补托管式节点后必须将其重启。但是，在没有安全措施的情况下重启生产中的节点可能会导致若干问题，例如调用告警、记录不正确的指标数据以及中断数据同步。

本教程将演示为了避免上述类似问题，如何通过使用 AWS Systems Manager 文档（SSM 文档）`AWS-RunPatchBaselineWithHooks` 来实现复杂的多步骤修补操作，该操作可完成以下事项：

1. 防止新连接到应用程序
2. 安装操作系统更新

3. 更新应用程序的软件包依赖项
4. 重启系统
5. 执行特定于应用程序的运行状况检查

在此示例中，我们以这种方式设置了我们的基础设施：

- 目标虚拟机在 Systems Manager 中注册为托管式节点。
- `Iptables` 作本地防火墙。
- 托管式节点上托管的应用程序正在端口 443 上运行。
- 托管式节点上托管的应用程序是 `nodeJS` 应用程序。
- 托管式节点上托管的应用程序由 pm2 进程管理器管理。
- 应用程序已经具有指定的运行状况检查终端节点。
- 应用程序的运行状况检查终结点不需要终端用户身份验证。终端节点允许进行运行状况检查，以满足组织在建立可用性方面的要求。（在您的环境中，可能只需确定正在运行 `nodeJS` 应用程序，并且能够侦听请求。在其他情况下，您可能还需要验证是否已建立到缓存层或数据库层的连接。）

本教程中的示例仅用于演示目的，并不意味着按原样实施到生产环境中。另请注意，Patch Manager（Systems Manager 中的一项工具）的生命周期挂钩功能和 `AWS-RunPatchBaselineWithHooks` 文档搭配使用，可以支持许多其他场景。下面是几个示例。

- 在修补之前停止指标报告代理，并在托管式节点重启后重启该代理。
- 在进行修补之前，将托管式节点与 CRM 或 PCS 集群分离，并在节点重启后重新连接。
- 在应用操作系统 (OS) 更新之后、托管式节点重启之前，在 Windows Server 计算机上更新第三方软件（例如，Java、Tomcat、Adobe 应用程序等）。

更新应用程序依赖项、修补托管式节点并执行特定于应用程序的运行状况检查

1. 使用以下内容为预安装脚本创建 SSM 文档，并将其命名为 `NodeJSAppPrePatch`。将`######`替换为应用程序的名称。

   此脚本会立即阻止新的传入请求，并在开始修补操作之前为已经激活的请求提供 5 秒钟的完成时间。对于 `sleep` 选项，请指定一个比传入请求完成通常所需秒数大的秒数。

   ```
   # exit on error
   set -e
   ```

```
# set up rule to block incoming traffic
iptables -I INPUT -j DROP -p tcp --syn --destination-port 443 || exit 1
# wait for current connections to end. Set timeout appropriate to your
 application's latency
sleep 5
# Stop your application
pm2 stop your_application
```

有关创建 SSM 文档的信息，请参阅 创建 SSM 文档内容。

2. 为安装后脚本创建包含以下内容的另一个 SSM 文档，以更新应用程序依赖项，并将其命名为 NodeJSAppPostPatch。将*/##/####/##*替换为通向您的应用程序的路径。

```
cd /your/application/path
npm update
# you can use npm-check-updates if you want to upgrade major versions
```

3. 创建另一个包含以下内容的 SSM 文档，其中的 onExit 脚本使应用程序备份并执行运行状况检查。命名此 SSM 文档为 NodeJSAppOnExitPatch。将*######*替换为您的应用程序的名称。

```
# exit on error
set -e
# restart nodeJs application
pm2 start your_application
# sleep while your application starts and to allow for a crash
sleep 10
# check with pm2 to see if your application is running
pm2 pid your_application
# re-enable incoming connections
iptables -D INPUT -j DROP -p tcp --syn --destination-port
# perform health check
/usr/bin/curl -m 10 -vk -A "" http://localhost:443/health-check || exit 1
```

4. 在State Manager（AWS Systems Manager 中的一项工具）中创建关联，通过执行下列步骤发出操作：

   1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
   2. 在导航窗格中，选择 State Manager，然后选择创建关联。
   3. 对于名称，请提供一个名称来帮助标识关联的用途。
   4. 在文档列表中，选择 AWS-RunPatchBaselineWithHooks。

5. 对于操作，选择安装。
6. （可选）对于快照 ID，提供您生成的 GUID，以帮助加快操作速度并确保一致性。GUID 值可以像 `00000000-0000-0000-0000-111122223333` 一样简单。
7. 对于安装前钩子文档名称，输入 `NodeJSAppPrePatch`。
8. 对于安装后钩子文档名称，输入 `NodeJSAppPostPatch`。
9. 对于针对 ExitHook 文档名称，输入 `NodeJSAppOnExitPatch`。

5. 对于 Targets（目标），通过指定标签、手动选择节点、选择资源组或选择所有托管式节点来标识托管式节点。
6. 对于指定时间表，指定运行关联的频率。对于托管式节点修补，每周修补一次是常见的节奏。
7. 在 Rate control（速率控制）部分中，选择用于控制如何在多个托管式节点上运行关联的选项。确保一次只更新部分托管式节点。否则，您的所有或大部分队列都可以同时离线。有关使用速率控制的更多信息，请参阅 [了解 State Manager 关联中的目标和速率控制]。
8. （可选）对于 Output options (输出选项)，要将命令输出保存到文件，请选中 Enable writing output to S3 (启用将输出写入 S3) 方框。在方框中输入存储桶和前缀（文件夹）名称。

> **Note**
>
> 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给托管式节点的实例配置文件的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅[配置 Systems Manager 所需的实例权限]或[为混合环境创建 IAM 服务角色]。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确认与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

9. 选择创建关联。

教程：使用 AWS CLI 修补服务器环境

以下过程介绍如何使用自定义补丁基准、补丁组和维护时段来修补服务器环境。

开始前的准备工作

- 在托管式节点上安装或更新 SSM Agent。要修补 Linux 托管式节点，节点必须运行 SSM Agent 2.0.834.0 版或更高版本。有关更多信息，请参阅 [使用 Run Command 更新 SSM Agent]。
- 配置 Maintenance Windows（AWS Systems Manager 中的一项工具）的角色和权限。有关更多信息，请参阅 [设置 Maintenance Windows]。
- 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

有关信息，请参阅安装或更新 AWS CLI 的最新版本。

配置 Patch Manager 并修补托管式节点（命令行）

1. 运行以下命令以创建名为 `Production-Baseline` 的适用于 Windows 的补丁基准。此补丁基准会在补丁发布或最后更新 7 天后批准用于生产环境。即，我们已标记补丁基准，以指示它适用于生产环境。

> **Note**
>
> `OperatingSystem` 参数和 `PatchFilters` 因补丁基准应用到的目标托管式节点的操作系统而异。有关详细信息，请参阅 OperatingSystem 和 PatchFilter。

Linux & macOS

```
aws ssm create-patch-baseline \
    --name "Production-Baseline" \
    --operating-system "WINDOWS" \
    --tags "Key=Environment,Value=Production" \
    --approval-rules
 "PatchRules=[{PatchFilterGroup={PatchFilters=[{Key=MSRC_SEVERITY,Values=[Critical,Impor
{Key=CLASSIFICATION,Values=[SecurityUpdates,Updates,ServicePacks,UpdateRollups,CriticalU
 \
    --description "Baseline containing all updates approved for production
 systems"
```

Windows Server

```
aws ssm create-patch-baseline ^
    --name "Production-Baseline" ^
    --operating-system "WINDOWS" ^
    --tags "Key=Environment,Value=Production" ^
    --approval-rules
 "PatchRules=[{PatchFilterGroup={PatchFilters=[{Key=MSRC_SEVERITY,Values=[Critical,Impor
{Key=CLASSIFICATION,Values=[SecurityUpdates,Updates,ServicePacks,UpdateRollups,CriticalU
 ^
    --description "Baseline containing all updates approved for production
 systems"
```

系统将返回类似于以下内容的信息。

```
{
   "BaselineId":"pb-0c10e65780EXAMPLE"
}
```

2. 运行以下命令为两个补丁组注册“生产-基准”补丁基准。这些组命名为“数据库服务器”和“前端服务器”。

Linux & macOS

```
aws ssm register-patch-baseline-for-patch-group \
    --baseline-id pb-0c10e65780EXAMPLE \
    --patch-group "Database Servers"
```

Windows Server

```
aws ssm register-patch-baseline-for-patch-group ^
    --baseline-id pb-0c10e65780EXAMPLE ^
    --patch-group "Database Servers"
```

系统将返回类似于以下内容的信息。

```
{
   "PatchGroup":"Database Servers",
   "BaselineId":"pb-0c10e65780EXAMPLE"
}
```

Linux & macOS

```
aws ssm register-patch-baseline-for-patch-group \
    --baseline-id pb-0c10e65780EXAMPLE \
    --patch-group "Front-End Servers"
```

Windows Server

```
aws ssm register-patch-baseline-for-patch-group ^
    --baseline-id pb-0c10e65780EXAMPLE ^
```

```
    --patch-group "Front-End Servers"
```

系统将返回类似于以下内容的信息。

```
{
   "PatchGroup":"Front-End Servers",
   "BaselineId":"pb-0c10e65780EXAMPLE"
}
```

3. 运行以下命令为生产服务器创建两个维护时段。第一个时段在每周二晚上 10 点运行。第二个时段在每周六晚上 10 点运行。此外，维护时段已标记来指示它适用于生产环境。

   Linux & macOS

   ```
   aws ssm create-maintenance-window \
       --name "Production-Tuesdays" \
       --tags "Key=Environment,Value=Production" \
       --schedule "cron(0 0 22 ? * TUE *)" \
       --duration 1 \
       --cutoff 0 \
       --no-allow-unassociated-targets
   ```

   Windows Server

   ```
   aws ssm create-maintenance-window ^
       --name "Production-Tuesdays" ^
       --tags "Key=Environment,Value=Production" ^
       --schedule "cron(0 0 22 ? * TUE *)" ^
       --duration 1 ^
       --cutoff 0 ^
       --no-allow-unassociated-targets
   ```

   系统将返回类似于以下内容的信息。

   ```
   {
      "WindowId":"mw-0c50858d01EXAMPLE"
   }
   ```

Linux & macOS

```
aws ssm create-maintenance-window \
    --name "Production-Saturdays" \
    --tags "Key=Environment,Value=Production" \
    --schedule "cron(0 0 22 ? * SAT *)" \
    --duration 2 \
    --cutoff 0 \
    --no-allow-unassociated-targets
```

Windows Server

```
aws ssm create-maintenance-window ^
    --name "Production-Saturdays" ^
    --tags "Key=Environment,Value=Production" ^
    --schedule "cron(0 0 22 ? * SAT *)" ^
    --duration 2 ^
    --cutoff 0 ^
    --no-allow-unassociated-targets
```

系统将返回类似于以下内容的信息。

```
{
   "WindowId":"mw-9a8b7c6d5eEXAMPLE"
}
```

4. 运行以下命令，将 Database 和 Front-End 服务器补丁组注册到其各自的维护时段。

Linux & macOS

```
aws ssm register-target-with-maintenance-window \
    --window-id mw-0c50858d01EXAMPLE \
    --targets "Key=tag:PatchGroup,Values=Database Servers" \
    --owner-information "Database Servers" \
    --resource-type "INSTANCE"
```

Windows Server

```
aws ssm register-target-with-maintenance-window ^
    --window-id mw-0c50858d01EXAMPLE ^
```

```
    --targets "Key=tag:PatchGroup,Values=Database Servers" ^
    --owner-information "Database Servers" ^
    --resource-type "INSTANCE"
```

系统将返回类似于以下内容的信息。

```
{
   "WindowTargetId":"e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
}
```

Linux & macOS

```
aws ssm register-target-with-maintenance-window \
--window-id mw-9a8b7c6d5eEXAMPLE \
--targets "Key=tag:PatchGroup,Values=Front-End Servers" \
--owner-information "Front-End Servers" \
--resource-type "INSTANCE"
```

Windows Server

```
aws ssm register-target-with-maintenance-window ^
    --window-id mw-9a8b7c6d5eEXAMPLE ^
    --targets "Key=tag:PatchGroup,Values=Front-End Servers" ^
    --owner-information "Front-End Servers" ^
    --resource-type "INSTANCE"
```

系统将返回类似于以下内容的信息。

```
{
   "WindowTargetId":"faa01c41-1d57-496c-ba77-ff9caEXAMPLE"
}
```

5. 运行以下命令注册一个补丁任务，该任务在 Database 和 Front-End 服务器各自的维护时段内安装缺少的更新。

Linux & macOS

```
aws ssm register-task-with-maintenance-window \
    --window-id mw-0c50858d01EXAMPLE \
```

```
    --targets "Key=WindowTargetIds,Values=e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
 \
    --task-arn "AWS-RunPatchBaseline" \
    --service-role-arn "arn:aws:iam::123456789012:role/MW-Role" \
    --task-type "RUN_COMMAND" \
    --max-concurrency 2 \
    --max-errors 1 \
    --priority 1 \
    --task-invocation-parameters "RunCommand={Parameters={Operation=Install}}"
```

Windows Server

```
aws ssm register-task-with-maintenance-window ^
    --window-id mw-0c50858d01EXAMPLE ^
    --targets "Key=WindowTargetIds,Values=e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
 ^
    --task-arn "AWS-RunPatchBaseline" ^
    --service-role-arn "arn:aws:iam::123456789012:role/MW-Role" ^
    --task-type "RUN_COMMAND" ^
    --max-concurrency 2 ^
    --max-errors 1 ^
    --priority 1 ^
    --task-invocation-parameters "RunCommand={Parameters={Operation=Install}}"
```

系统将返回类似于以下内容的信息。

```
{
   "WindowTaskId":"4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE"
}
```

Linux & macOS

```
aws ssm register-task-with-maintenance-window \
    --window-id mw-9a8b7c6d5eEXAMPLE \
    --targets "Key=WindowTargetIds,Values=faa01c41-1d57-496c-ba77-ff9caEXAMPLE"
 \
    --task-arn "AWS-RunPatchBaseline" \
    --service-role-arn "arn:aws:iam::123456789012:role/MW-Role" \
    --task-type "RUN_COMMAND" \
    --max-concurrency 2 \
    --max-errors 1 \
```

```
    --priority 1 \
    --task-invocation-parameters "RunCommand={Parameters={Operation=Install}}"
```

Windows Server

```
aws ssm register-task-with-maintenance-window ^
    --window-id mw-9a8b7c6d5eEXAMPLE ^
    --targets "Key=WindowTargetIds,Values=faa01c41-1d57-496c-ba77-ff9caEXAMPLE"
 ^
    --task-arn "AWS-RunPatchBaseline" ^
    --service-role-arn "arn:aws:iam::123456789012:role/MW-Role" ^
    --task-type "RUN_COMMAND" ^
    --max-concurrency 2 ^
    --max-errors 1 ^
    --priority 1 ^
    --task-invocation-parameters "RunCommand={Parameters={Operation=Install}}"
```

系统将返回类似于以下内容的信息。

```
{
   "WindowTaskId":"8a5c4629-31b0-4edd-8aea-33698EXAMPLE"
}
```

6. 运行以下命令以获取补丁组的高级补丁合规性摘要。概括性补丁合规性摘要包括补丁处于相应状态的托管式节点的数量。

> **Note**
>
> 在第一个维护时段内，在补丁任务运行之前，预计会在摘要中看到托管式节点数量为零。

Linux & macOS

```
aws ssm describe-patch-group-state \
    --patch-group "Database Servers"
```

Windows Server

```
aws ssm describe-patch-group-state ^
```

```
    --patch-group "Database Servers"
```

系统将返回类似于以下内容的信息。

```
{
   "Instances": number,
   "InstancesWithFailedPatches": number,
   "InstancesWithInstalledOtherPatches": number,
   "InstancesWithInstalledPatches": number,
   "InstancesWithInstalledPendingRebootPatches": number,
   "InstancesWithInstalledRejectedPatches": number,
   "InstancesWithMissingPatches": number,
   "InstancesWithNotApplicablePatches": number,
   "InstancesWithUnreportedNotApplicablePatches": number
}
```

7. 运行以下命令以获取补丁组的每个托管式节点的补丁摘要状态。每个托管式节点摘要包括处于相应补丁状态的许多补丁（按补丁组的每个托管式节点划分）。

   Linux & macOS

   ```
   aws ssm describe-instance-patch-states-for-patch-group \
       --patch-group "Database Servers"
   ```

   Windows Server

   ```
   aws ssm describe-instance-patch-states-for-patch-group ^
       --patch-group "Database Servers"
   ```

   系统将返回类似于以下内容的信息。

   ```
   {
      "InstancePatchStates": [
         {
            "BaselineId": "string",
            "FailedCount": number,
            "InstalledCount": number,
            "InstalledOtherCount": number,
            "InstalledPendingRebootCount": number,
            "InstalledRejectedCount": number,
   ```

```
            "InstallOverrideList": "string",
            "InstanceId": "string",
            "LastNoRebootInstallOperationTime": number,
            "MissingCount": number,
            "NotApplicableCount": number,
            "Operation": "string",
            "OperationEndTime": number,
            "OperationStartTime": number,
            "OwnerInformation": "string",
            "PatchGroup": "string",
            "RebootOption": "string",
            "SnapshotId": "string",
            "UnreportedNotApplicableCount": number
        }
    ]
}
```

有关可以用于 Patch Manager 配置任务的其他 AWS CLI 命令的示例，请参阅 使用 AWS CLI 处理 Patch Manager 资源使用。

排查 Patch Manager 问题

利用以下信息，帮助排查Patch Manager（AWS Systems Manager 中的一项工具）出现的问题。

主题

- 问题：baseline_overrides.json 出现“Invoke-PatchBaselineOperation : Access Denied”错误或“Unable to download file from S3”错误
- 问题：修补失败，没有明显的原因或错误消息
- 问题：意外的补丁合规结果
- 在 Linux 上运行 AWS-RunPatchBaseline 时出现的错误
- 在 Windows Server 上运行 AWS-RunPatchBaseline 时出现错误
- 使用 AWS 支持 Automation 运行手册
- 联系 AWS 支持

问题：**baseline_overrides.json** 出现“Invoke-PatchBaselineOperation : Access Denied”错误或“Unable to download file from S3”错误

问题：运行补丁策略指定的修补操作时，您会收到类似于以下示例的错误。

Example error on Windows Server

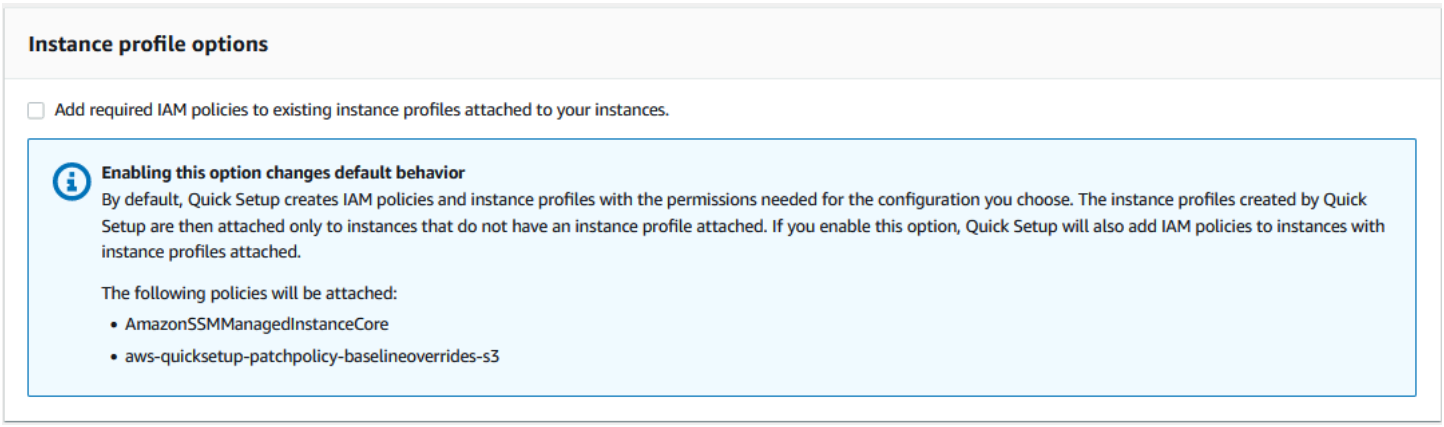
```
----------ERROR-------
Invoke-PatchBaselineOperation : Access Denied
At C:\ProgramData\Amazon\SSM\InstanceData\i-02573cafcfEXAMPLE\document\orchestr
ation\792dd5bd-2ad3-4f1e-931d-abEXAMPLE\PatchWindows\_script.ps1:219 char:13
+ $response = Invoke-PatchBaselineOperation -Operation Install -Snapsho ...
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
+ CategoryInfo : OperationStopped: (Amazon.Patch.Ba...UpdateOpera
tion:InstallWindowsUpdateOperation) [Invoke-PatchBaselineOperation], Amazo
nS3Exception
+ FullyQualifiedErrorId : PatchBaselineOperations,Amazon.Patch.Baseline.Op
erations.PowerShellCmdlets.InvokePatchBaselineOperation
failed to run commands: exit status 0xffffffff
```

Example error on Linux

```
[INFO]: Downloading Baseline Override from s3://aws-quicksetup-
patchpolicy-123456789012-abcde/baseline_overrides.json
[ERROR]: Unable to download file from S3: s3://aws-quicksetup-
patchpolicy-123456789012-abcde/baseline_overrides.json.
[ERROR]: Error loading entrance module.
```

原因：您在 Quick Setup 中创建了补丁策略，并且某些托管式节点已附加了实例配置文件（适用于 EC2 实例）或服务角色（适用于非 EC2 计算机）。

但是，如下图所示，您未选中将所需的 IAM 策略添加到附加到实例的现有实例配置文件复选框。



创建补丁策略时，还会创建一个 Amazon S3 存储桶来存储策略的配置 `baseline_overrides.json` 文件。如果您在创建策略时未选中将所需的 IAM policy 添加到附加到实例的现有实例配置文件复选

框，则在 S3 存储桶中访问 `baseline_overrides.json` 所需的 IAM policy 和资源标签不会自动添加到现有 IAM 实例配置文件和服务角色中。

解决方案 1：删除现有的补丁策略配置，然后创建替代策略配置，确保选中将所需的 IAM policy 添加到附加到实例的现有实例配置文件复选框。此选择会将此 Quick Setup 配置创建的 IAM policy 应用于已附加实例配置文件或服务角色的节点。（默认情况下，Quick Setup 向还没有实例配置文件或服务角色的实例和节点添加所需策略。）有关更多信息，请参阅 Automate organization-wide patching using a Quick Setup patch policy。

解决方案 2：手动将所需的权限和标签添加到与 Quick Setup 一起使用的每个 IAM 实例配置文件和 IAM 服务角色。有关说明，请参阅补丁策略 S3 存储桶的权限。

问题：修补失败，没有明显的原因或错误消息

问题：修补操作失败，但未返回错误消息。

可能的原因：如果一次对 `AWS-RunPatchBaseline` 进行多次调用，它们可能会相互冲突，从而导致修补任务失败。修补日志中可能未指明这一点。

要检查并发修补操作是否可能相互中断，请查看 Run Command（AWS Systems Manager 中的一项工具）中的命令历史记录。对于修补失败的托管式节点，请检查是否有多个操作在 2 分钟内尝试对计算机进行修补。这种情况有时会导致故障。

您还可以借助以下命令使用 AWS Command Line Interface（AWS CLI）检查并发修补尝试。将 *`node-id`* 的值替换为托管式节点的 ID。

```
aws ssm list-commands \
    --filter "key=DocumentName,value=AWS-RunPatchBaseline" \
    --query 'Commands[*].
{CommandId:CommandId,RequestedDateTime:RequestedDateTime,Status:Status}' \
    --instance-id node-id \
    --output table
```

解决方案：如果您确定由于同一托管式节点上的竞争修补操作而导致修补失败，请调整修补配置以避免再次发生这种情况。例如，如果两个维护时段指定了重叠的修补时间，请删除或修改其中一个时段。如果维护时段指定了一个修补操作，但补丁策略同时指定了不同的修补操作，则可以考虑将此务从维护时段中删除。

如果您确定冲突的修补操作不是导致这种情况失败的原因，我们建议联系 AWS 支持。

问题：意外的补丁合规结果

问题：在 Scan 操作后查看生成的修补合规性详细信息时，结果中包含的信息不符合补丁基准中设置的规则。例如，您在补丁基准中添加到 Rejected patches（已拒绝补丁）列表中的异常被列为 Missing。或者，即使您的补丁基准仅指定 Critical 补丁，分类为 Important 的补丁也被列为缺失。

原因：Patch Manager 目前支持运行 Scan 操作的多种方法：

- Quick Setup 中配置的补丁策略
- Quick Setup 中配置的主机管理选项
- 运行补丁 Scan 或 Install 任务的维护时段
- 按需 Patch now（立即修补）操作

运行 Scan 操作时，它会覆盖最近扫描的合规性详细信息。如果您设置了多个方法来运行 Scan 操作，并且这些方法使用不同的补丁基准和不同的规则，那么它们会导致不同的补丁合规结果。

解决方案：为避免出现意外的补丁合规结果，我们建议一次仅使用一种方法来运行 Patch Manager Scan 操作。有关更多信息，请参阅 避免意外覆盖补丁合规性数据。

在 Linux 上运行 **AWS-RunPatchBaseline** 时出现的错误

主题

- 问题：“没有这样的文件或目录”错误
- 问题：“另一个进程已获得 yum 锁定” 错误
- 问题：“权限拒绝/未能运行命令” 错误
- 问题：“无法下载有效负载” 错误
- 问题：“不支持的软件包管理器和 python 版本组合” 错误
- 问题：Patch Manager 没有应用指定来排除某些软件包的规则
- 问题：修补失败同时 Patch Manager 报告 TLS 的服务器名称指示扩展不可用
- 问题:Patch Manager 报告“没有更多的镜像要尝试”
- 问题：修补失败并显示消息“Error code returned from curl is 23”
- 问题：修补失败并显示消息“Error unpacking rpm package…”
- 问题：修补失败并显示消息“上传清单时出现服务端错误”

- 问题：修补失败并显示消息“Errors were encountered while downloading packages”
- 问题：修补失败并显示消息“The following signatures couldn't be verified because the public key is not available”
- 问题：修补失败并显示消息“NoMoreMirrorsRepoError”
- 问题：修补失败并显示消息“Unable to download payload”
- 问题：修补失败并显示消息“install errors: dpkg: error: dpkg frontend is locked by another process”
- 问题：Ubuntu Server 上的修补失败并出现“dpkg was interrupted”错误
- 问题：程序包管理器实用工具无法解析包依赖项
- 问题：SLES 托管节点上的 Zypper 软件包锁定依赖项故障

问题：“没有这样的文件或目录”错误

问题：当您运行 AWS-RunPatchBaseline 时，修补失败，并出现以下错误之一。

```
IOError: [Errno 2] No such file or directory: 'patch-baseline-operations-X.XX.tar.gz'
```

```
Unable to extract tar file: /var/log/amazon/ssm/patch-baseline-operations/patch-
baseline-operations-1.75.tar.gz.failed to run commands: exit status 155
```

```
Unable to load and extract the content of payload, abort.failed to run commands: exit
 status 152
```

原因 1：要运行 AWS-RunPatchBaseline 的两个命令同时在同一托管式节点上运行。这将创建一个竞争条件，导致临时 file patch-baseline-operations* 未能正确创建或访问。

原因 2：/var 目录中的存储空间依然不足。

解决方案 1：确保维护时段没有具有相同的优先级并且在相同的目标 ID 上运行的两个或多个运行 AWS-RunPatchBaseline 的 Run Command 任务。如果是这种情况，请重新排序优先级。Run Command 是 AWS Systems Manager 中的一项工具。

解决方案 2：确保一次只有一个维护时段正在运行 Run Command 任务，该任务对相同的目标、在相同的时间表上使用 AWS-RunPatchBaseline。如果出现这种情况，请更改时间表。

解决方案 3：确保按照相同的计划只有一个State Manager关联正在运行 AWS-RunPatchBaseline，并针对相同的托管式节点。State Manager是 AWS Systems Manager 中的一项工具。

解决方案 4：在 /var 目录中为更新程序包释放足够的存储空间。

问题：“另一个进程已获得 yum 锁定” 错误

问题：当您运行 AWS-RunPatchBaseline 时，修补失败并显示以下错误。

```
12/20/2019 21:41:48 root [INFO]: another process has acquired yum lock, waiting 2 s and
 retry.
```

原因：AWS-RunPatchBaseline 文档已经开始在某个托管式节点上运行（在该节点中，该文档已在其他操作中运行），并且已获得软件包管理器 yum 进程。

解决方案：确保按计划运行 AWS-RunPatchBaseline 的 State Manager 关联、维护时段任务或其他配置不会在几乎同一时间针对相同的托管节点。

问题：“权限拒绝/未能运行命令” 错误

问题：当您运行 AWS-RunPatchBaseline，则修补失败并显示以下错误。

```
sh:
/var/lib/amazon/ssm/instanceid/document/orchestration/commandid/PatchLinux/_script.sh:
 Permission denied
failed to run commands: exit status 126
```

原因：/var/lib/amazon/ 可能会使用 noexec 权限。这是一个问题，因为 SSM Agent 把有效负载脚本下载到 /var/lib/amazon/ssm 并从该位置运行它们。

解决方案：确保您已将独占分区配置为 /var/log/amazon 和 /var/lib/amazon，并且它们挂载了 exec 权限。

问题：“无法下载有效负载” 错误

问题：当您运行 AWS-RunPatchBaseline 时，修补失败并显示以下错误。

```
Unable to download payload: https://s3.amzn-s3-demo-bucket.region.amazonaws.com/
aws-ssm-region/patchbaselineoperations/linux/payloads/patch-baseline-operations-
X.XX.tar.gz.failed to run commands: exit status 156
```

原因：托管式节点没有访问指定的 Amazon Simple Storage Service (Amazon S3) 存储桶所需的权限。

解决方案：更新您的网络配置，以便可访问 S3 终端节点。有关更多详细信息，请参阅 SSM Agent 与 AWS 托管 S3 存储桶进行通信 中 Patch Manager 对 S3 存储桶的所需访问。

问题：“不支持的软件包管理器和 python 版本组合” 错误

问题：当您运行 AWS-RunPatchBaseline 时，修补失败并显示以下错误。

```
An unsupported package manager and python version combination was found. Apt requires
 Python3 to be installed.
failed to run commands: exit status 1
```

原因：Debian Server、Raspberry Pi OS 或 Ubuntu Server 实例上未安装受支持版本的 python3。

解决方案：在服务器上安装受支持版本的 python3（3.0-3.10），Debian Server、Raspberry Pi OS 以及 Ubuntu Server 托管式节点必须使用这些版本。

问题：Patch Manager 没有应用指定来排除某些软件包的规则

问题：您试图排除某些软件包，方法是在 /etc/yum.conf 文件中指定这些软件包，格式为 exclude=*package-name*，但在 Patch Manager Install 操作期间不排除它们。

原因：Patch Manager 不包含 /etc/yum.conf 文件中指定的排除项。

解决方案：要排除特定软件包，请创建自定义补丁基准并创建排除不想安装的软件包的规则。

问题：修补失败同时 Patch Manager 报告 TLS 的服务器名称指示扩展不可用

问题：修补操作发出以下消息。

```
/var/log/amazon/ssm/patch-baseline-operations/urllib3/util/ssl_.py:369:
SNIMissingWarning: An HTTPS request has been made, but the SNI (Server Name Indication)
 extension
to TLS is not available on this platform. This might cause the server to present an
 incorrect TLS
certificate, which can cause validation failures. You can upgrade to a newer version of
 Python
to solve this.
For more information, see https://urllib3.readthedocs.io/en/latest/advanced-
usage.html#ssl-warnings
```

原因：此消息不表示错误存在。相反，这是一个警告，即随操作系统一起分发的较旧版本的 Python 不支持 TLS 服务器名称指示。在连接到支持 SNI 的 AWS API 时，Systems Manager 补丁有效负载脚本发出这个警告。

解决方案：若要在报告此消息时解决任何修补失败问题，请查看 stdout 和 stderr 文件的内容。如果您尚未配置补丁基准以将这些文件存储在 S3 存储桶或 Amazon CloudWatch Logs 中，则可以在 Linux 托管节点上的以下位置找到这些文件。

/var/lib/amazon/ssm/*instance-id*/document/orchestration/*Run-Command-execution-id*/awsrunShellScript/PatchLinux

问题:Patch Manager 报告“没有更多的镜像要尝试”

问题：修补操作发出以下消息。

```
[Errno 256] No more mirrors to try.
```

原因：在托管式节点上配置的存储库无法正常工作。可能的原因包括：

- yum 缓存已损坏。
- 由于网络相关问题，无法访问存储库 URL。

解决方案：Patch Manager 使用托管式节点的默认软件包管理器执行修补操作。双重检查存储库是否已配置并正常运行。

问题：修补失败并显示消息“Error code returned from curl is 23”

问题：使用 AWS-RunPatchBaseline 的修补操作失败，并出现类似以下内容的错误：

```
05/01/2025 17:04:30 root [ERROR]: Error code returned from curl is 23
```

原因：系统上使用的 curl 工具缺少写入文件系统所需的权限。如果程序包管理器的默认 curl 工具被其他版本（例如随 snap 安装的版本）所取代，则可能会发生这种情况。

解决方案：如果安装其他版本时卸载了程序包管理器提供的 curl 版本，请重新安装。

如果您需要安装多个 curl 版本，请确保与程序包管理器关联的版本位于 PATH 变量中列出的第一个目录中。您可以通过运行命令 echo $PATH 来查看系统上检查可执行文件的目录的当前顺序。

问题：修补失败并显示消息“Error unpacking rpm package…”

问题：修补操作失败，并出现类似以下内容的错误：

```
Error : Error unpacking rpm package python-urllib3-1.25.9-1.amzn2.0.2.noarch
```

```
python-urllib3-1.25.9-1.amzn2.0.1.noarch was supposed to be removed but is not!
failed to run commands: exit status 1
```

原因 1：特定程序包存在于多个程序包安装程序（例如 pip 和 yum 或 dnf）中时，使用默认程序包管理器时可能会发生冲突。

一个常见的示例是 urllib3 程序包，可以在 pip、yum 和 dnf 中找到。

原因 2：python-urllib3 程序包已损坏。如果 yum 或 dnf 先前安装了 rpm 程序包之后，pip 安装或更新了程序包文件，则可能会发生这种情况。

解决方案：通过运行命令 sudo pip uninstall urllib3 从 pip 中删除 python-urllib3 程序包，仅将此程序包保留在默认程序包管理器（yum 或 dnf）中。

问题：修补失败并显示消息“上传清单时出现服务端错误”

问题：运行 AWS-RunPatchBaseline 文档时，收到以下错误消息：

```
Encounter service side error when uploading the inventory
```

原因：要运行 AWS-RunPatchBaseline 的两个命令同时在同一托管式节点上运行，这导致在修补操作期间初始化 boto3 客户端时产生了争用条件。

解决方案：确保按计划运行 AWS-RunPatchBaseline 的 State Manager 关联、维护时段任务或其他配置不会在几乎同一时间针对相同的托管节点。

问题：修补失败并显示消息“Errors were encountered while downloading packages”

问题：在修补期间，您会收到类似以下内容的错误：

```
YumDownloadError: [u'Errors were encountered while downloading
packages.', u'libxml2-2.9.1-6.el7_9.6.x86_64: [Errno 5] [Errno 12]
Cannot allocate memory', u'libxslt-1.1.28-6.el7.x86_64: [Errno 5]
[Errno 12] Cannot allocate memory', u'libcroco-0.6.12-6.el7_9.x86_64:
[Errno 5] [Errno 12] Cannot allocate memory', u'openldap-2.4.44-25.el7_9.x86_64:
[Errno 5] [Errno 12] Cannot allocate memory',
```

原因：托管式节点上的可用内存不足时，可能会发生此错误。

解决方案：配置交换内存，或将实例升级到其他类型以增加内存。然后开始新的修补操作。

问题：修补失败并显示消息“The following signatures couldn't be verified because the public key is not available”

问题：Ubuntu Server 上的修补失败，并出现类似以下内容的错误：

```
02/17/2022 21:08:43 root [ERROR]: W:GPG error:
http://repo.mysql.com/apt/ubuntu  bionic InRelease: The following
signatures couldn't be verified because the public key is not available:
NO_PUBKEY 467B942D3A79BD29, E:The repository ' http://repo.mysql.com/apt/ubuntu bionic
```

原因：GNU 隐私保护（GPG）密钥已过期或丢失。

解决方案：刷新 GPG 密钥，或者重新添加密钥。

例如，借助之前显示的错误，我们发现 467B942D3A79BD29 密钥丢失，必须添加此密钥。为此，请运行以下命令之一：

```
sudo apt-key adv --keyserver hkps://keyserver.ubuntu.com --recv-keys 467B942D3A79BD29
```

```
sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys 467B942D3A79BD29
```

或者，要刷新所有密钥，请运行以下命令：

```
sudo apt-key adv --keyserver hkps://keyserver.ubuntu.com --refresh-keys
```

如果之后再次出现此错误，我们建议将问题报告给维护存储库的组织。在修复可用之前，您可以编辑 `/etc/apt/sources.list` 文件以在修补过程中省略存储库。

为此，请打开 `sources.list` 文件进行编辑，找到存储库所在的行，然后在行首插入 # 字符将其注释掉。保存并关闭此文件。

问题：修补失败并显示消息“NoMoreMirrorsRepoError”

问题：您会收到类似以下内容的错误：

```
NoMoreMirrorsRepoError: failure: repodata/repomd.xml from pgdg94: [Errno 256] No more
 mirrors to try.
```

原因：源存储库中存在错误。

解决方案：我们建议将问题报告给维护存储库的组织。在错误修复之前，您可以在操作系统级别禁用存储库。为此，请运行以下命令，将 *repo-name* 的值替换为存储库名称：

```
yum-config-manager --disable repo-name
```

以下为示例。

```
yum-config-manager --disable pgdg94
```

运行此命令后，再次执行修补操作。

问题：修补失败并显示消息“Unable to download payload”

问题：您会收到类似以下内容的错误：

```
Unable to download payload:
https://s3.dualstack.eu-west-1.amazonaws.com/aws-ssm-eu-west-1/patchbaselineoperations/
linux/payloads/patch-baseline-operations-1.83.tar.gz.
failed  to run commands: exit status 156
```

原因：托管式节点配置存在错误或不完整。

解决方案：确保托管式节点配置如下：

- 安全组中的出站 TCP 443 规则。
- NACL 中的出口 TCP 443 规则。
- NACL 中的入口 TCP 1024-65535 规则。
- 路由表中的 NAT/IGW 用于提供与 S3 端点的连接。如果实例无法访问互联网，请为其提供与 S3 端点的连接。为此，请在 VPC 中添加 S3 网关端点，并将其与托管式节点的路由表集成。

问题：修补失败并显示消息“install errors: dpkg: error: dpkg frontend is locked by another process”

问题：修补失败，并出现类似以下内容的错误：

```
install errors: dpkg: error: dpkg frontend is locked by another process
failed to run commands: exit status 2
Failed to install package; install status Failed
```

原因：程序包管理器已经在操作系统级别的托管式节点上运行另一个进程。如果其他进程需要很长时间才能完成，则 Patch Manager 修补操作可能会超时并失败。

解决方案：使用程序包管理器的其他进程完成后，执行新的修补操作。

问题：Ubuntu Server 上的修补失败并出现“dpkg was interrupted”错误

问题：在 Ubuntu Server 上，修补失败并出现类似以下内容的错误：

```
E: dpkg was interrupted, you must manually run
'dpkg --configure -a' to correct the problem.
```

原因：一个或多个程序包配置错误。

解决方案：执行以下步骤：

1. 通过依次运行以下命令来检查受影响的程序包，以及每个程序包存在的问题：

   ```
   sudo apt-get check
   ```

   ```
   sudo dpkg -C
   ```

   ```
   dpkg-query -W -f='${db:Status-Abbrev} ${binary:Package}\n' | grep -E ^.[^nci]
   ```

2. 通过运行以下命令更正有问题的程序包：

   ```
   sudo dpkg --configure -a
   ```

3. 如果之前的命令未完全解决问题，可运行以下命令：

   ```
   sudo apt --fix-broken install
   ```

问题：程序包管理器实用工具无法解析包依赖项

问题：托管式节点上的本机程序包管理器无法解析程序包依赖项，修补失败。以下错误消息示例指示使用 yum 作为程序包管理器的操作系统上的此类故障。

```
09/22/2020 08:56:09 root [ERROR]: yum update failed with result code: 1,
```

```
message: [u'rpm-python-4.11.3-25.amzn2.0.3.x86_64 requires rpm = 4.11.3-25.amzn2.0.3',
u'awscli-1.18.107-1.amzn2.0.1.noarch requires python2-botocore = 1.17.31']
```

原因：在 Linux 操作系统上，Patch Manager 使用计算机上的本机程序包管理器来执行修补操作。例如 yum、dnf、apt 和 zypper。应用程序会根据需要自动检测、安装、更新或删除从属程序包。但是，某些情况可能会导致程序包管理器无法完成依赖项操作，例如：

- 操作系统上配置了多个相互冲突的存储库。
- 由于网络相关问题，无法访问远程存储库 URL。
- 存储库中发现了错误架构的包。

解决方案：由于各种原因，依赖项可能会导致修补失败。因此，我们建议您联系 AWS 支持 以帮助您进行故障排除。

问题：SLES 托管节点上的 Zypper 软件包锁定依赖项故障

问题：使用 `Install` 操作在 SUSE Linux Enterprise Server 实例上运行 `AWS-RunPatchBaseline` 时，修补程序失败，并出现与软件包锁定相关的依赖项检查错误。您可能会看到类似于以下内容的错误消息：

```
Problem: mock-pkg-has-dependencies-0.2.0-21.adistro.noarch requires mock-pkg-standalone
 = 0.2.0, but this requirement cannot be provided
  uninstallable providers: mock-pkg-standalone-0.2.0-21.adistro.noarch[local-repo]
 Solution 1: remove lock to allow installation of mock-pkg-
standalone-0.2.0-21.adistro.noarch[local-repo]
 Solution 2: do not install mock-pkg-has-dependencies-0.2.0-21.adistro.noarch
 Solution 3: break mock-pkg-has-dependencies-0.2.0-21.adistro.noarch by ignoring some
 of its dependencies

Choose from above solutions by number or cancel [1/2/3/c] (c): c
```

在此示例中，软件包 `mock-pkg-standalone` 已锁定，您可以通过运行 `sudo zypper locks` 并在输出中查找软件包名称来进行验证。

或者您可能会看到指示依赖项检查失败的日志条目：

```
Encountered a known exception in the CLI Invoker:
 CLIInvokerError(error_message='Dependency check failure during commit process',
 error_code='4')
```

> **Note**
>
> 该问题仅在 `Install` 操作期间出现。`Scan` 操作不应用软件包锁定，也不受现有软件包锁定的影响。”

原因：当 zypper 软件包锁定由于依赖项冲突而阻止安装或更新软件包时，就会发生这种错误。软件包锁定可能由若干原因所致：

- 客户应用的锁定：您或您的系统管理员使用 zypper 命令（例如 `zypper addlock`）手动锁定了软件包。
- Patch Manager 拒绝修补：当您在补丁基准的“拒绝的修补程序”列表中指定软件包时，Patch Manager 会自动应用软件包锁定以阻止安装软件包。
- 操作中断产生的残留锁：在极少数情况下，如果修补操作在 Patch Manager 清理临时锁之前中断（例如系统重启），则在托管节点上可能会留存残留的软件包锁。

解决方案：要解决 zypper 软件包锁定的问题，请根据原因执行以下步骤：

步骤 1：确定已锁定的软件包

连接到 SLES 托管节点并运行以下命令，列出所有当前锁定的软件包：

```
sudo zypper locks
```

步骤 2：确定锁的来源

- 如果锁定的软件包是您出于系统稳定性而故意锁定的，请考虑软件包是否需要保持锁定状态，或者是否可以临时解锁以进行修补。
- 如果锁定的软件包与补丁基准的“拒绝的修补程序”列表中的条目相匹配，则表明这些软件包可能是因为修补操作中断而产生的残留锁。在正常操作中，Patch Manager 会临时应用这些锁定，并在操作完成后自动移除。您可以从拒绝列表中移除软件包，也可以修改补丁基准的规则。
- 如果无法识别锁定的软件包，并且软件包不是故意锁定的，则可能是之前中断的修补操作产生的残留锁。

第 3 步：根据需要移除锁定

要移除特定的软件包锁定，请使用以下命令：

```
sudo zypper removelock package-name
```

要移除所有软件包锁定（谨慎使用），请运行：

```
sudo zypper cleanlocks
```

步骤 4：更新补丁基准（如适用）

如果锁定是由补丁基准中拒绝的修补程序造成的：

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Patch Manager。
3. 选择补丁基准选项卡，然后选择自定义补丁基准。
4. 选择操作、修改补丁基准。
5. 在“拒绝的修补程序”部分，查看列出的软件包并移除应允许安装的任何软件包。
6. 选择保存更改。

步骤 5：重试修补操作

移除有问题的锁定并根据需要更新补丁基准之后，请再次运行 AWS-RunPatchBaseline 文档。

> **Note**
>
> Patch Manager 在 Install 操作期间对拒绝的修补程序应用锁定时，其设计旨在于修补操作完成后自动清理这些锁。如果在运行 sudo zypper locks 时看到了这些锁定，则表示之前的修补操作在清理完成之前已中断。但是，如果修补操作中断，则可能需要按照本流程中的说明进行手动清理。

预防：为避免未来出现 zypper 锁定冲突，请：

- 仔细查看补丁基准中拒绝的修补程序，确保其中仅包含您真正想要排除的软件包。
- 避免手动锁定可能需要作为安全更新依赖项的软件包。
- 如果必须手动锁定软件包，请记录原因并定期检查锁定。
- 确保修补操作成功完成，且未因系统重启或其他因素而中断。

- 监控修补操作直至完成，避免因为系统重启或其他可能妨碍正确清理临时锁定的操作而中断。

在 Windows Server 上运行 **AWS-RunPatchBaseline** 时出现错误

主题

- 问题：产品系列/产品对不匹配
- 问题: AWS-RunPatchBaseline 输出返回 HRESULT(Windows Server）
- 问题：托管式节点无法访问 Windows 更新目录或 WSUS
- 问题：无法下载 PatchBaselineOperations （补丁基准操作） PowerShell 模块
- 问题：缺少补丁

问题：产品系列/产品对不匹配

问题：在 Systems Manager 控制台中创建补丁基准时，您指定了产品系列和产品。例如，您可能选择：

- 产品系列：`Office`

  产品：`Office 2016`

原因：如果您尝试使用不匹配的产品系列/产品对创建补丁基准，则会显示一条错误消息。以下是可能发生这种情况的原因：

- 您选择了有效的产品系列/产品对，但随后删除了所选的产品系列。
- 您从 Obsolete or mismatched options (过时或不匹配的选项) 子列表中选择的产品，而不是 Available and matching options (可用和匹配选项) 子列表。

  产品过时或不匹配的选项子列表中的项目，可能是通过 SDK 或 AWS Command Line Interface (AWS CLI) `create-patch-baseline` 命令错误输入的。这可能意味着存在拼写错误或产品被分配到了错误的产品系列。如果为之前的补丁基准指定了某个产品，但该产品目前没有 Microsoft 提供的补丁，它也会包含在 过时或不匹配的选项 子列表中。

解决方案为避免控制台中出现此问题，请始终从当前可用选项子列表中选择选项。

您还可以使用 AWS CLI 中的 `describe-patch-properties` 命令或 `DescribePatchProperties` API 命令查看具有可用补丁的产品。

问题: **AWS-RunPatchBaseline** 输出返回 **HRESULT**(Windows Server)

问题：您收到类似如下内容的错误：

```
----------ERROR-------
Invoke-PatchBaselineOperation : Exception Details: An error occurred when
attempting to search Windows Update.
Exception Level 1:
 Error Message: Exception from HRESULT: 0x80240437
 Stack Trace: at WUApiLib.IUpdateSearcher.Search(String criteria)..
(Windows updates)
11/22/2020 09:17:30 UTC | Info | Searching for Windows Updates.
11/22/2020 09:18:59 UTC | Error | Searching for updates resulted in error: Exception
 from HRESULT: 0x80240437
----------ERROR-------
failed to run commands: exit status 4294967295
```

原因：此输出表示本机 Windows 更新 API 无法运行修补操作。

解决方案：检查以下 microsoft.com 主题中的 `HResult` 代码以确定解决错误的故障排除步骤：

- 按组件划分的 Windows 更新错误代码
- Windows 更新常见错误和缓解措施

问题：托管式节点无法访问 Windows 更新目录或 WSUS

问题：您收到类似如下内容的错误。

```
Downloading PatchBaselineOperations PowerShell module from https://s3.aws-api-
domain/path_to_module.zip to C:\Windows\TEMP\Amazon.PatchBaselineOperations-1.29.zip.

Extracting PatchBaselineOperations zip file contents to temporary folder.

Verifying SHA 256 of the PatchBaselineOperations PowerShell module files.

Successfully downloaded and installed the PatchBaselineOperations PowerShell module.

Patch Summary for

PatchGroup :

BaselineId :
```

```
Baseline : null

SnapshotId :

RebootOption : RebootIfNeeded

OwnerInformation :

OperationType : Scan

OperationStartTime : 1970-01-01T00:00:00.0000000Z

OperationEndTime : 1970-01-01T00:00:00.0000000Z

InstalledCount : -1

InstalledRejectedCount : -1

InstalledPendingRebootCount : -1

InstalledOtherCount : -1

FailedCount : -1

MissingCount : -1

NotApplicableCount : -1

UnreportedNotApplicableCount : -1

EC2AMAZ-VL3099P - PatchBaselineOperations Assessment Results - 2020-12-30T20:59:46.169

----------ERROR-------

Invoke-PatchBaselineOperation : Exception Details: An error occurred when attempting to
 search Windows Update.

Exception Level 1:

Error Message: Exception from HRESULT: 0x80072EE2

Stack Trace: at WUApiLib.IUpdateSearcher.Search(String criteria)
```

```
at
 Amazon.Patch.Baseline.Operations.PatchNow.Implementations.WindowsUpdateAgent.SearchForUpdates(

searchCriteria)

At C:\ProgramData\Amazon\SSM\InstanceData\i-02573cafcfEXAMPLE\document\orchestration
\3d2d4864-04b7-4316-84fe-eafff1ea58

e3\PatchWindows\_script.ps1:230 char:13

+ $response = Invoke-PatchBaselineOperation -Operation Install -Snapsho ...

+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

+ CategoryInfo : OperationStopped:
 (Amazon.Patch.Ba...UpdateOperation:InstallWindowsUpdateOperation) [Inv

oke-PatchBaselineOperation], Exception

+ FullyQualifiedErrorId : Exception Level 1:

Error Message: Exception Details: An error occurred when attempting to search Windows
 Update.

Exception Level 1:

Error Message: Exception from HRESULT: 0x80072EE2

Stack Trace: at WUApiLib.IUpdateSearcher.Search(String criteria)

at
 Amazon.Patch.Baseline.Operations.PatchNow.Implementations.WindowsUpdateAgent.SearchForUpdates(
 searc

---Error truncated----
```

原因：此错误可能与 Windows 更新组件，或缺乏连接到 Windows 更新目录或 Windows 服务器更新服务 (WSUS)有关。

解决方案：确认托管式节点已通过互联网网关、NAT 网关或 NAT 实例与 Microsoft 更新目录连接。如果您使用的是 WSUS，请确认该托管式节点已连接到环境中的 WSUS 服务器。如果可以连接到预期目标，请查看 Microsoft 文档中 HResult 0x80072EE2 的其他潜在原因。这可能表明存在操作系统级问题。

问题：无法下载 PatchBaselineOperations（补丁基准操作）PowerShell 模块

问题：您收到类似如下内容的错误。

```
Preparing to download PatchBaselineOperations PowerShell module from S3.

Downloading PatchBaselineOperations PowerShell module from https://s3.aws-api-
domain/path_to_module.zip to C:\Windows\TEMP\Amazon.PatchBaselineOperations-1.29.zip.
----------ERROR-------

C:\ProgramData\Amazon\SSM\InstanceData\i-02573cafcfEXAMPLE\document\orchestration
\aaaaaaaa-bbbb-cccc-dddd-4f6ed6bd5514\

PatchWindows\_script.ps1 : An error occurred when executing PatchBaselineOperations:
 Unable to connect to the remote server

+ CategoryInfo : NotSpecified: (:) [Write-Error], WriteErrorException

+ FullyQualifiedErrorId : Microsoft.PowerShell.Commands.WriteErrorException,_script.ps1

failed to run commands: exit status 4294967295
```

解决方案：检查托管式节点与 Amazon Simple Storage Service (Amazon S3) 的连接和权限。托管式节点的 AWS Identity and Access Management (IAM) 角色必须使用 SSM Agent 与 AWS 托管 S3 存储桶进行通信 中引用的最低权限。节点必须通过 Amazon S3 网关端点、NAT 网关或互联网网关与 Amazon S3 端点进行通信。有关 AWS Systems Manager SSM Agent (SSM Agent) 的 VPC 终端节点要求的更多信息，请参阅 使用适用于 Systems Manager 的 VPC 端点提高 EC2 实例的安全性。

问题：缺少补丁

问题：AWS-RunPatchbaseline 已成功完成，但缺少一些补丁。

以下是一些常见原因及其解决方案。

原因 1：基准无效。

解决方案 1：要检查这是否是原因，请使用以下过程。

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Run Command。
3. 选择命令历史记录选项卡，然后选择要检查其基准的命令。

4. 选择缺少补丁的托管式节点。
5. 选择步骤 1 - 输出并查找 `BaselineId` 值。
6. 检查分配的补丁基准配置，即补丁基准的操作系统、产品名称、分类和严重性。
7. 转至微软更新目录。
8. 搜索 Microsoft 知识库 (KB) 文章 ID（例如 KB3216916）。
9. 确认 Product（产品）下的值与托管式节点的值相匹配，然后选择相应的 Title（标题）。一个新更新详细信息窗口将打开。
10. 在概述选项卡中，分类和 MSRC 严重性必须与之前找到的补丁基准配置匹配。

原因 2：已替换补丁。

解决方案 2：要检查是否为真，请使用以下过程。

1. 转至微软更新目录。
2. 搜索 Microsoft 知识库 (KB) 文章 ID（例如 KB3216916）。
3. 确认 Product（产品）下的值与托管式节点的值相匹配，然后选择相应的 Title（标题）。一个新更新详细信息窗口将打开。
4. 转至程序包详细信息选项卡。在此更新已被以下更新所替换：标头下面查找条目。

原因 3：相同的补丁可能具有不同的知识库编号，因为 WSUS 和窗口联机更新由 Microsoft 处理为独立的发布渠道。

解决方案 3：检查补丁的资格。如果软件包在 WSUS 下不可用，请安装操作系统构建 14393.3115。如果该软件包适用于所有操作系统构建，请安装操作系统构建 18362.1256 和 18363.1256。

使用 AWS 支持 Automation 运行手册

AWS 支持 提供了两个 Automation 运行手册，可用于排查与修补相关的某些问题。

- `AWSSupport-TroubleshootWindowsUpdate`：AWSSupport-TroubleshootWindowsUpdate 运行手册用于识别可能导致 Amazon Elastic Compute Cloud（Amazon EC2）Windows Server 实例 Windows Server 更新失败的问题。
- `AWSSupport-TroubleshootPatchManagerLinux`：AWSSupport-TroubleshootPatchManagerLinux 运行手册用于解决可能导致使用Patch Manager的基于 Linux 的托管式节点上出现补丁失败的常见问题。本运行手册的主要目标是确定补丁命令失败的根本原因并提出补救方案。

> **Note**
>
> 运行 Automation 运行手册会产生费用。有关信息，请参阅 AWS Systems Manager Automation 定价。

联系 AWS 支持

如果您无法在本节或 AWS re:Post 中的 Systems Manager 问题中找到故障排除解决方案，并且您有开发人员、业务或企业 支持 计划，可在 AWS 支持 创建一个技术支持用例。

在您联系 支持 之前，收集以下物品：

- SSM 代理日志
- Run Command 命令 ID、维护时段 ID 或自动化执行 ID
- 对于 Windows Server 托管式节点，还要收集以下内容：
  - `%PROGRAMDATA%\Amazon\PatchBaselineOperations\Logs`，如 Windows 的选项卡 如何安装补丁 中所描述的那样
  - Windows 更新日志：对于 Windows Server 2012 R2 及更高版本，使用 `%windir%/WindowsUpdate.log`。对于 Windows Server 2016 及更新版本，首先运行 PowerShell 命令 `Get-WindowsUpdateLog`，然后再使用 `%windir%/WindowsUpdate.log`
- 对于 Linux 托管式节点，还要收集以下内容：
  - 目录 `/var/lib/amazon/ssm/`*`instance-id`*`/document/orchestration/`*`Run-Command-execution-id`*`/awsrunShellScript/PatchLinux` 的内容

AWS Systems Manager Run Command

使用 Run Command（AWS Systems Manager 中的一项工具），您可以通过安全方式远程管理托管式节点的配置。托管式节点是在混合和多云环境中已为 Systems Manager 配置的任何 Amazon Elastic Compute Cloud（Amazon EC2）实例或非 EC2 计算机。Run Command 支持您自动完成常见管理任务以及大规模执行一次性配置更改。您可以从 AWS Management Console、AWS Command Line Interface（AWS CLI）、AWS Tools for Windows PowerShell 或 AWS SDK 使用 Run Command。Run Command 不另外收费。要开始使用 Run Command，请打开 Systems Manager 控制台。在导航窗格中，请选择 Run Command。

管理员可以使用 Run Command 安装或引导启动应用程序，构建部署管道，从 Auto Scaling 组移除实例时捕获日志文件，将实例加入 Windows 域，等等。

由于系统支持 API 的分布式特性，Run Command API 遵循最终一致性模型。这意味着您执行的对您的资源产生影响的 API 命令的结果可能不会立即对您运行的所有后续命令可见。在执行紧随上一个 API 命令的 API 命令时，应记住这一点。

开始使用

下表包含可帮助您开始使用 Run Command 的信息。

主题	详细信息
为 AWS Systems Manager 设置托管节点	验证您是否已在混合和多云环境中完成 Amazon Elastic Compute Cloud（Amazon EC2）实例以及非 EC2 计算机的设置要求。
使用 Systems Manager 管理混合和多云环境中的节点	（可选）将本地服务器和虚拟机注册到 AWS，以便可以使用 Run Command 来管理它们。
the section called “使用 Systems Manager 管理边缘设备”	（可选）配置边缘设备，以便可以使用 Run Command 管理它们。
在托管节点上运行命令	了解如何使用 AWS Management Console 运行以一个或多个托管式节点为目标的命令。
Run Command 演练	了解如何使用 Tools for Windows PowerShell 或 AWS CLI 运行命令。

EventBridge 支持

在 Amazon EventBridge 规则中，支持将此 Systems Manager 工具作为事件类型和目标类型。有关信息，请参阅 使用 Amazon EventBridge 监控 Systems Manager 事件 和 引用：Amazon EventBridge 事件模式和 Systems Manager 类型。

更多信息

- 在 EC2 实例上远程 Run Command（10 分钟教程）
- 请参阅《Amazon Web Services 一般参考》中的 Systems Manager 服务配额
- AWS Systems Manager API 参考

主题

- 设置 Run Command
- 在托管节点上运行命令
- 在命令中使用退出代码
- 了解命令状态
- Run Command 演练
- 对 Systems Manager Run Command 进行故障排除

设置 Run Command

必须先为将运行命令的用户配置 AWS Identity and Access Management（IAM）策略，然后才能使用 Run Command（AWS Systems Manager 中的一项工具）管理节点。如果您在 IAM 策略中使用任何全局条件键执行 SendCommand 操作，则必须包含 `aws:ViaAWSService` 条件键并将布尔值设置为 `true`。示例如下：

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:SendCommand"
            ],
            "Resource": [
                "arn:aws:ssm:us-east-1:111122223333:document/YourDocument"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:SourceVpce": [
                        "vpce-1234567890abcdef0"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:SendCommand"
```

```
            ],
            "Resource": [
                "arn:aws:ssm:us-east-1:111122223333:document/YourDocument"
            ],
            "Condition": {
                "Bool": {
                    "aws:ViaAWSService": "true"
                }
            }
        }
    ]
}
```

您还必须为 Systems Manager 配置节点。有关更多信息，请参阅 为 AWS Systems Manager 设置托管节点。

我们建议您完成以下可选设置任务，以帮助最大限度地改善托管式节点的安保状况并尽量减少节点的日常管理工作。

使用 Amazon EventBridge 监控命令执行情况

您可以使用 Eventbridge 记录命令执行状态更改。您可以创建一个规则，只要状态发生变换或者在变换到一个或多个感兴趣的状态时，就运行该规则。此外，您还可以将 Run Command 指定为发生 EventBridge 事件时的目标操作。有关更多信息，请参阅 为 Systems Manager 事件配置 EventBridge。

使用 Amazon CloudWatch Logs 监控命令执行情况

您可以将 Run Command 配置为定期将所有命令输出和错误日志发送到 Amazon CloudWatch 日志组。您可以近乎实时地监控这些输出日志，搜索特定短语、值或模式，以及基于搜索创建警报。有关更多信息，请参阅 为 Run Command 配置 Amazon CloudWatch Logs。

限制对特定托管式节点的 Run Command 访问

您可以通过使用 AWS Identity and Access Management (IAM) 来限制用户在托管式节点上运行命令的能力。特别是，您可以创建 IAM policy，其中包含一个条件，规定用户只能在使用特定标签标记的托管节点上运行命令。有关更多信息，请参阅 根据标签限制 Run Command 访问。

根据标签限制 Run Command 访问

本节介绍如何通过在 IAM policy 中指定标签条件来限制用户在托管式节点上运行命令的能力。托管式节点包括混合和多云环境中为 Systems Manager 配置的 Amazon EC2 实例和非 EC2 节点。尽管没有

明确显示相关信息，但您也可以限制对托管式 AWS IoT Greengrass 核心设备的访问。首先，您必须标记 AWS IoT Greengrass 设备。有关更多信息，请参阅 AWS IoT Greengrass Version 2 开发人员指南中的标记 AWS IoT Greengrass Version 2 资源。

您可以通过创建一个 IAM policy 将命令执行限制到特定托管节点，该策略包括一个条件，规定用户只能在带有特定标签的节点上运行命令。在以下示例中，通过以下方式来允许用户使用 Run Command (`Effect: Allow, Action: ssm:SendCommand`)：在任何节点 (`Resource: arn:aws:ec2:*:*:instance/*`) 上使用任何 SSM 文档 (`Resource: arn:aws:ssm:*:*:document/*`)，条件是节点为 Finance WebServer (`ssm:resourceTag/Finance: WebServer`)。如果用户向未经标记或具有除 `Finance: WebServer` 以外的任何标签的节点发送命令，则执行结果将显示 `AccessDenied`。

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "ssm:SendCommand"
         ],
         "Resource":[
            "arn:aws:ssm:*:*:document/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "ssm:SendCommand"
         ],
         "Resource":[
            "arn:aws:ec2:*:*:instance/*"
         ],
         "Condition":{
            "StringLike":{
               "ssm:resourceTag/Finance":[
                  "WebServers"
               ]
            }
         }
```

```
      }
   ]
}
```

您可以创建允许用户在使用多个标签标记的托管式节点上运行命令的 IAM policy。以下策略允许用户在具有两个标签的托管式节点上运行命令。如果用户向未使用这两个标签标记的节点发送命令，则执行结果将显示 AccessDenied。

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "ssm:SendCommand"
         ],
         "Resource":"*",
         "Condition":{
            "StringLike":{
               "ssm:resourceTag/tag_key1":[
                  "tag_value1"
               ],
               "ssm:resourceTag/tag_key2":[
                  "tag_value2"
               ]
            }
         }
      },
      {
         "Effect":"Allow",
         "Action":[
            "ssm:SendCommand"
         ],
         "Resource":[
            "arn:aws:ssm:us-west-1::document/AWS-*",
            "arn:aws:ssm:us-east-2::document/AWS-*"
         ]
      },
      {
```

```
         "Effect":"Allow",
         "Action":[
            "ssm:UpdateInstanceInformation",
            "ssm:ListCommands",
            "ssm:ListCommandInvocations",
            "ssm:GetDocument"
         ],
         "Resource":"*"
      }
   ]
}
```

您也可以创建允许用户在多个已标记托管式节点组上运行命令的 IAM policy。以下示例策略允许用户在任一已标记节点组或两个已标记节点组上运行命令。

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "ssm:SendCommand"
         ],
         "Resource":"*",
         "Condition":{
            "StringLike":{
               "ssm:resourceTag/tag_key1":[
                  "tag_value1"
               ]
            }
         }
      },
      {
         "Effect":"Allow",
         "Action":[
            "ssm:SendCommand"
         ],
         "Resource":"*",
         "Condition":{
```

```
            "StringLike":{
               "ssm:resourceTag/tag_key2":[
                  "tag_value2"
               ]
            }
         }
      },
      {
         "Effect":"Allow",
         "Action":[
            "ssm:SendCommand"
         ],
         "Resource":[
            "arn:aws:ssm:us-west-1::document/AWS-*",
            "arn:aws:ssm:us-east-2::document/AWS-*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "ssm:UpdateInstanceInformation",
            "ssm:ListCommands",
            "ssm:ListCommandInvocations",
            "ssm:GetDocument"
         ],
         "Resource":"*"
      }
   ]
}
```

有关创建 IAM policy 的更多信息，请参阅《IAM 用户指南》中的托管策略与内联策略。有关标记托管式节点的更多信息，请参阅 AWS Resource Groups 用户指南中的标签编辑器。

在托管节点上运行命令

本节包括有关如何发送从 AWS Systems Manager 控制台到托管式节点的命令的信息。此部分还包括有关如何取消命令的信息。

请注意，如果节点为 var 目录配置了 `noexec` 挂载选项，Run Command 将无法成功运行命令。

> **Important**
>
> 当您使用 Run Command 发送命令时，不要包含纯文本格式的敏感信息，例如密码、配置数据或其他密钥。您账户中的所有 Systems Manager API 活动都将记录在 AWS CloudTrail 日志的 S3 存储桶中。这意味着任何有权访问该 Amazon S3 存储桶的用户都能够查看这些密钥的纯文本值。因此，我们建议您创建和使用 `SecureString` 参数，以便加密您在 Systems Manager 操作中使用的敏感数据。
>
> 有关更多信息，请参阅 使用 IAM 策略限制对 Parameter Store 参数的访问。

执行历史记录保留

每个命令的历史记录最多可保留 30 天。此外，您可以在 Amazon Simple Storage Service 中存储所有日志文件的副本，或在 AWS CloudTrail 中保存所有 API 调用的审计跟踪。

相关信息

有关使用其他工具发送命令的信息，请参阅以下主题：

- 演练：将 AWS Tools for Windows PowerShell 与 Run Command 结合使用或者 AWS Tools for PowerShell Cmdlet 参考的 AWS Systems Manager 部分中的示例。
- 演练：将 AWS CLI 与 Run Command 结合使用或者 SSM CLI 参考中的示例

内容

- 从控制台运行命令
- 使用指令文档版本运行命令
- 大规模运行命令
- 取消命令

从控制台运行命令

在不必登录的情况下，可以通过 AWS Management Console使用 Run Command（AWS Systems Manager 中的一项工具）配置托管式节点。此主题包括演示如何使用 Run Command 在托管式节点上更新 SSM Agent 的示例。

开始前的准备工作

在使用 Run Command 发送命令之前，请确认托管式节点符合 Systems Manager 设置要求。

使用 Run Command 发送命令

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Run Command。
3. 选择 Run command（运行命令）。
4. 在命令文档列表中，请选择 Systems Manager 文档。
5. 在 Command parameters (命令参数) 部分中，为必需的参数指定值。
6. 在 Targets（目标）部分中，通过指定标签、手动选择实例或边缘设备或指定资源组，选择要在其上运行此操作的托管式节点。

   > **Tip**
   >
   > 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

7. 对于 Other parameters（其他参数）：
   - 对于 Comment（注释），请输入有关此命令的信息。
   - 对于 Timeout (seconds) (超时 (秒))，请指定在整个命令执行失败之前系统等待的秒数。
8. 对于 Rate control（速率控制）：
   - 对于 Concurrency（并发），请指定要同时运行该命令的托管式节点的数量或百分比。

     > **Note**
     >
     > 如果您通过指定应用于托管式节点的标签或指定 AWS Resource Groups 来选择目标，但不确定有多少个托管式节点已被设为目标，则可通过指定百分比来限制可同时运行该文档的目标的数量。

   - 对于 Error threshold（错误阈值），请指定当命令在一定数量或百分比的节点上失败后，何时在其他托管式节点上停止运行该命令。例如，如果您指定三个错误，Systems Manager 将在收到第四个错误时停止发送该命令。仍在处理该命令的托管式节点也可能发送错误。
9. （可选）选择一个 CloudWatch 警报以应用于您的命令进行监控。要将 CloudWatch 警报附加到命令，运行命令的 IAM 主体必须具有 `iam:createServiceLinkedRole` 操作的权限。有关 CloudWatch 警报的更多信息，请参阅使用 Amazon CloudWatch 警报。请注意，如果您的警报激活，任何待处理的命令调用都不会运行。

10. （可选）对于 输出选项，要将命令输出保存到文件，请选中 将命令输出写入 S3 存储桶 框。在输入框中输入存储桶和前缀（文件夹）名称。

> **Note**
>
> 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给实例的实例配置文件（适用于 EC2 实例）或 IAM 服务角色（混合激活的计算机）的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确保与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

11. 在 SNS 通知部分，如果需要发送有关命令执行状态的通知，请选中 Enable SNS notifications（启用 SNS 通知）复选框。

    有关为 Run Command 配置 Amazon SNS 通知的更多信息，请参阅 使用 Amazon SNS 通知监控 Systems Manager 状态更改。

12. 选择 Run（运行）。

有关取消命令的信息，请参阅 the section called “取消命令”。

重新运行命令

Systems Manager 包含两个选项，可帮助您从 Systems Manager 控制台中的 Run Command 页面重新运行命令。

- Rerun (重新运行)：利用此按钮，您可以运行同一个命令而不对其进行更改。
- 复制到新项目：此按钮将一个命令的设置复制到一个新命令，并为您提供在运行该命令之前编辑这些设置的选项。

重新运行命令

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Run Command。
3. 选择要重新运行的命令。从命令详细信息页面执行命令后，可以立即重新运行此命令。或者，您可以从 Command history (命令历史记录) 选项卡中选择先前运行的命令。

4. 选择 Rerun (重新运行) 以在不进行更改的情况下运行相同的命令，或者选择复制到新项目以在运行命令之前编辑命令设置。

使用指令文档版本运行命令

您可以使用文档版本参数指定在运行命令时使用 AWS Systems Manager 文档的哪个版本。您可以为此参数指定以下选项之一：

- $DEFAULT
- $LATEST
- 版本号

运行以下过程，使用文档版本参数运行命令。

Linux

在本地 Linux 计算机上使用 AWS CLI 运行命令

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 列出所有可用文档

   此命令将基于 AWS Identity and Access Management (IAM) 权限列出您的账户可用的所有文档。

   ```
   aws ssm list-documents
   ```

3. 运行以下命令，查看文档的不同版本。将####替换为您自己的信息。

   ```
   aws ssm list-document-versions \
       --name "document name"
   ```

4. 运行以下命令，运行使用 SSM 文档版本的命令。将每个#######替换为您自己的信息。

   ```
   aws ssm send-command \
       --document-name "AWS-RunShellScript" \
       --parameters commands="echo Hello" \
       --instance-ids instance-ID \
   ```

```
    --document-version '$LATEST'
```

Windows

在本地 Windows 计算机上使用 AWS CLI 运行命令

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 列出所有可用文档

   此命令将基于 AWS Identity and Access Management (IAM) 权限列出您的账户可用的所有文档。

   ```
   aws ssm list-documents
   ```

3. 运行以下命令，查看文档的不同版本。将####替换为您自己的信息。

   ```
   aws ssm list-document-versions ^
       --name "document name"
   ```

4. 运行以下命令，运行使用 SSM 文档版本的命令。将每个#######替换为您自己的信息。

   ```
   aws ssm send-command ^
       --document-name "AWS-RunShellScript" ^
       --parameters commands="echo Hello" ^
       --instance-ids instance-ID ^
       --document-version "$LATEST"
   ```

PowerShell

要使用 Tools for PowerShell 运行命令，请执行以下步骤：

1. 如果您尚未安装和配置 AWS Tools for PowerShell（适用于 Windows PowerShell 的工具），请执行这些操作。

   有关信息，请参阅安装 AWS Tools for PowerShell。

2. 列出所有可用文档

此命令将基于 AWS Identity and Access Management (IAM) 权限列出您的账户可用的所有文档。

```
Get-SSMDocumentList
```

3. 运行以下命令，查看文档的不同版本。将*####*替换为您自己的信息。

```
Get-SSMDocumentVersionList `
    -Name "document name"
```

4. 运行以下命令，运行使用 SSM 文档版本的命令。将每个*#######*替换为您自己的信息。

```
Send-SSMCommand `
    -DocumentName "AWS-RunShellScript" `
    -Parameter @{commands = "echo helloWorld"} `
    -InstanceIds "instance-ID" `
    -DocumentVersion $LATEST
```

大规模运行命令

您可以使用 Run Command（AWS Systems Manager 中的一项工具）通过使用 `targets` 在托管式节点实例集上运行命令。`targets` 参数根据您为托管式节点指定的标签接受 `Key,Value` 组合。当您运行该命令时，系统会找到并尝试在匹配指定标签的所有托管式节点上运行命令。有关标记托管式实例的更多信息，请参阅《Tagging AWS Resources User Guide》中的 Tagging your AWS resources。有关标记托管式 IoT 设备的信息，请参阅 AWS IoT Greengrass Version 2 开发人员指南中的标记 AWS IoT Greengrass Version 2 资源。

您也可以使用 `targets` 参数将特定托管式节点 ID 的列表设为目标，如下一部分中所述。

为控制数百个或数千个托管式节点的命令运行，Run Command 还包含一些参数，用于限制同时处理一个请求的节点数量以及取消命令前其可引发的错误数量。

内容

- 将多个托管式节点设为目标
- 使用速率控制

将多个托管式节点设为目标

您可以通过指定标签、AWS 资源组名称或托管式节点 ID 来运行命令并将托管式节点设为目标。

以下示例显示使用 AWS Command Line Interface (AWS CLI) 中的 Run Command 时的命令格式。将每个*#######*替换为您自己的信息。本部分的示例命令使用 `[...]` 进行截断。

示例 1：将标签设为目标

Linux & macOS

```
aws ssm send-command \
    --document-name document-name \
    --targets Key=tag:tag-name,Values=tag-value \
    [...]
```

Windows

```
aws ssm send-command ^
    --document-name document-name ^
    --targets Key=tag:tag-name,Values=tag-value ^
    [...]
```

示例 2：通过名称将 AWS 资源组设为目标

您最多可以为每个命令指定一个资源组名称。当您创建资源组时，我们建议您包含 `AWS::SSM:ManagedInstance` 和 `AWS::EC2::Instance` 作为分组条件中的资源类型。

> **Note**
>
> 为发送将资源组设为目标的命令，您必须已获得列出或查看属于该组的资源的 AWS Identity and Access Management (IAM) 权限。有关更多信息，请参阅 AWS Resource Groups 用户指南中的设置权限。

Linux & macOS

```
aws ssm send-command \
    --document-name document-name \
```

```
    --targets Key=resource-groups:Name,Values=resource-group-name \
    [...]
```

Windows

```
aws ssm send-command ^
    --document-name document-name ^
    --targets Key=resource-groups:Name,Values=resource-group-name ^
    [...]
```

示例 3：通过资源类型将 AWS 资源组设为目标

您最多可以为每个命令指定五个资源组类型。当您创建资源组时，我们建议您包含 `AWS::SSM:ManagedInstance` 和 `AWS::EC2::Instance` 作为分组条件中的资源类型。

> **Note**
>
> 为了发送将资源组设为目标的命令，您必须已被授予列出或查看属于该组的资源的 IAM 权限。有关更多信息，请参阅 AWS Resource Groups 用户指南中的设置权限。

Linux & macOS

```
aws ssm send-command \
    --document-name document-name \
    --targets Key=resource-groups:ResourceTypeFilters,Values=resource-
type-1,resource-type-2 \
    [...]
```

Windows

```
aws ssm send-command ^
    --document-name document-name ^
    --targets Key=resource-groups:ResourceTypeFilters,Values=resource-
type-1,resource-type-2 ^
    [...]
```

示例 4：将实例 ID 设为目标

以下示例演示如何使用 instanceids 密钥和 targets 参数将托管式节点设为目标。您可以使用此密钥来将托管式 AWS IoT Greengrass 核心设备设为目标，因为每台设备分配到了 mi-*ID_NUMBER*。您可以在 Fleet Manager（AWS Systems Manager 中的一项工具）中查看设备 ID。

Linux & macOS

```
aws ssm send-command \
    --document-name document-name \
    --targets Key=instanceids,Values=instance-ID-1,instance-ID-2,instance-ID-3 \
    [...]
```

Windows

```
aws ssm send-command ^
    --document-name document-name ^
    --targets Key=instanceids,Values=instance-ID-1,instance-ID-2,instance-ID-3 ^
    [...]
```

如果您使用名为 Environment 的 Key，以及 Development、Test、Pre-production 和 Production 的 Values 标记不同环境的托管式节点，则可以使用采用以下语法的 targets 参数，向其中一个环境的所有托管式节点发送命令。

Linux & macOS

```
aws ssm send-command \
    --document-name document-name \
    --targets Key=tag:Environment,Values=Development \
    [...]
```

Windows

```
aws ssm send-command ^
    --document-name document-name ^
    --targets Key=tag:Environment,Values=Development ^
    [...]
```

通过添加到 Values 列表，您可以将其他环境中的其他托管式节点设为目标。使用逗号分隔项目。

Linux & macOS

```
aws ssm send-command \
    --document-name document-name \
    --targets Key=tag:Environment,Values=Development,Test,Pre-production \
    [...]
```

Windows

```
aws ssm send-command ^
    --document-name document-name ^
    --targets Key=tag:Environment,Values=Development,Test,Pre-production ^
    [...]
```

变体：使用多个 Key 条件细化您的目标。

通过包括多个 Key 条件，您可以细化您的命令的目标数。如果包括多个 Key 条件，系统会将符合所有条件的托管式节点设为目标。以下命令会将标记为财务部门和标记为数据库服务器角色的所有托管式节点设为目标。

Linux & macOS

```
aws ssm send-command \
    --document-name document-name \
    --targets Key=tag:Department,Values=Finance Key=tag:ServerRole,Values=Database \
    [...]
```

Windows

```
aws ssm send-command ^
    --document-name document-name ^
    --targets Key=tag:Department,Values=Finance Key=tag:ServerRole,Values=Database ^
    [...]
```

变体：使用多个 Key 和 Value 条件

对上一个示例进行扩展，您可以通过在 Values 条件中包括其他项目来将多个部门和多个服务器角色设为目标。

Linux & macOS

```
aws ssm send-command \
    --document-name document-name \
    --targets Key=tag:Department,Values=Finance,Marketing
 Key=tag:ServerRole,Values=WebServer,Database \
    [...]
```

Windows

```
aws ssm send-command ^
    --document-name document-name ^
    --targets Key=tag:Department,Values=Finance,Marketing
 Key=tag:ServerRole,Values=WebServer,Database ^
    [...]
```

变体：使用多个 Values 条件将已标记托管式节点设为目标

如果您使用名为 Department 的 Key，以及 Sales 和 Finance 的 Values 标记不同环境的托管式节点，则可以使用采用以下语法的 targets 参数，向这些环境的所有节点发送命令。

Linux & macOS

```
aws ssm send-command \
    --document-name document-name \
    --targets Key=tag:Department,Values=Sales,Finance \
    [...]
```

Windows

```
aws ssm send-command ^
    --document-name document-name ^
    --targets Key=tag:Department,Values=Sales,Finance ^
    [...]
```

您最多可以为每个键指定五个键和五个值。

如果某个标签密钥（标签名称）或某个标签值包含空格，则需要将该标签密钥或该值用引号引起来，如以下示例所示。

示例： Value 标签中的空格

Linux & macOS

```
aws ssm send-command \
    --document-name document-name \
    --targets Key=tag:OS,Values="Windows Server 2016" \
    [...]
```

Windows

```
aws ssm send-command ^
    --document-name document-name ^
    --targets Key=tag:OS,Values="Windows Server 2016" ^
    [...]
```

示例： tag 键和 Value 中的空格

Linux & macOS

```
aws ssm send-command \
    --document-name document-name \
    --targets Key="tag:Operating System",Values="Windows Server 2016" \
    [...]
```

Windows

```
aws ssm send-command ^
    --document-name document-name ^
    --targets Key="tag:Operating System",Values="Windows Server 2016" ^
    [...]
```

示例：Values 的列表中一个项目中的空格

Linux & macOS

```
aws ssm send-command \
    --document-name document-name \
    --targets Key=tag:Department,Values="Sales","Finance","Systems Mgmt" \
```

```
    [...]
```

Windows

```
aws ssm send-command ^
    --document-name document-name ^
    --targets Key=tag:Department,Values="Sales","Finance","Systems Mgmt" ^
    [...]
```

使用速率控制

您可以使用并发控件和错误控件，控制将命令发送到组中托管式节点的速率。

主题

- 使用并发控件
- 使用错误控件

使用并发控件

您可以使用 max-concurrency 参数 [Run a command（运行命令）页面中的 Concurrency（并发）选项] 来控制同时运行命令的托管式节点数量。您可以指定绝对数量的托管式节点（例如 **10**），也可以指定目标集百分比（例如 **10%**）。队列系统将命令传递给单个节点，并等待系统确认了初始调用，再将命令发送到两个或更多节点。系统以指数增长方式将命令发送到更多节点，直到系统达到 max-concurrency 值。max-concurrency 值默认为 50。下列示例介绍如何为 max-concurrency 参数指定值。

Linux & macOS

```
aws ssm send-command \
    --document-name document-name \
    --max-concurrency 10 \
    --targets Key=tag:Environment,Values=Development \
    [...]
```

```
aws ssm send-command \
    --document-name document-name \
    --max-concurrency 10% \
    --targets Key=tag:Department,Values=Finance,Marketing
 Key=tag:ServerRole,Values=WebServer,Database \
```

```
    [...]
```

Windows

```
aws ssm send-command ^
    --document-name document-name ^
    --max-concurrency 10 ^
    --targets Key=tag:Environment,Values=Development ^
    [...]
```

```
aws ssm send-command ^
    --document-name document-name ^
    --max-concurrency 10% ^
    --targets Key=tag:Department,Values=Finance,Marketing
 Key=tag:ServerRole,Values=WebServer,Database ^
    [...]
```

使用错误控件

您也可以使用 max-errors 参数 [Run a command（运行命令）页面中的 Error threshold（错误阈值）字段] 设置错误限制，将命令的执行控制在几百个或几千个托管式节点范围内。该参数指定系统停止向其他托管式节点发送命令之前所允许的错误数。您可以指定绝对数量的错误（例如 **10**），也可以指定目标集百分比（例如 **10%**）。例如，如果您指定 **3**，系统将在收到第四个错误时停止发送命令。如果您指定 **0**，则系统会在返回第一个错误结果后停止向其他托管式节点发送命令。如果您向 50 个托管式节点发送命令并将 max-errors 设置为 **10%**，则系统会在收到第六个错误时停止向其他节点发送命令。

当达到 max-errors 时，允许完成已经运行命令的调用，但是其中一些调用也可能失败。如果您需要确保失败的调用数不超过 max-errors，请将 max-concurrency 设置为 **1**，以便一次进行一个调用。max-errors 默认为 0。下列示例介绍如何为 max-errors 参数指定值。

Linux & macOS

```
aws ssm send-command \
    --document-name document-name \
    --max-errors 10 \
    --targets Key=tag:Database,Values=Development \
    [...]
```

```
aws ssm send-command \
    --document-name document-name \
    --max-errors 10% \
    --targets Key=tag:Environment,Values=Development \
    [...]
```

```
aws ssm send-command \
    --document-name document-name \
    --max-concurrency 1 \
    --max-errors 1 \
    --targets Key=tag:Environment,Values=Production \
    [...]
```

Windows

```
aws ssm send-command ^
    --document-name document-name ^
    --max-errors 10 ^
    --targets Key=tag:Database,Values=Development ^
    [...]
```

```
aws ssm send-command ^
    --document-name document-name ^
    --max-errors 10% ^
    --targets Key=tag:Environment,Values=Development ^
    [...]
```

```
aws ssm send-command ^
    --document-name document-name ^
    --max-concurrency 1 ^
    --max-errors 1 ^
    --targets Key=tag:Environment,Values=Production ^
    [...]
```

取消命令

只要服务指明命令处于 Pending (待处理) 或 Executing (正在执行) 状态，您就可以尝试取消命令。但是，即使命令仍处于其中某种状态，我们也无法保证该命令将被取消并且基础流程将停止。

使用控制台取消命令

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Run Command。
3. 选择要取消的命令调用。
4. 选择取消命令。

使用 AWS CLI 取消命令

运行以下命令。将每个*#######*替换为您自己的信息。

Linux & macOS

```
aws ssm cancel-command \
    --command-id "command-ID" \
    --instance-ids "instance-ID"
```

Windows

```
aws ssm cancel-command ^
    --command-id "command-ID" ^
    --instance-ids "instance-ID"
```

有关已取消命令的状态的信息，请参阅 了解命令状态。

在命令中使用退出代码

在某些情况下，您可能需要使用退出代码管理处理命令的方式。

在命令中指定退出代码

使用 Run Command（AWS Systems Manager 中的一项工具），您可以指定退出代码来确定命令的处理方式。默认情况下，脚本中运行的最后一个命令的退出代码将报告为整个脚本的退出代码。例如，您有一个包含三个命令的脚本。第一个命令失败，但以下命令成功。由于最后一个命令成功，因此执行状态报告为 succeeded。

Shell 脚本

要在第一个命令失败时使整个脚本失败，您可以包含一个 shell 条件语句，以便在最后一个命令失败之前退出脚本。请使用以下方法。

```
<command 1>
    if [ $? != 0 ]
    then
        exit <N>
    fi
    <command 2>
    <command 3>
```

在以下示例中，如果第一个命令失败，则整个脚本将失败。

```
cd /test
    if [ $? != 0 ]
    then
        echo "Failed"
        exit 1
    fi
    date
```

PowerShell 脚本

PowerShell 要求您在脚本中显式调用 `exit`，使 Run Command 能够成功捕获退出代码。

```
<command 1>
    if ($?) {<do something>}
    else {exit <N>}
    <command 2>
    <command 3>
    exit <N>
```

示例如下：

```
cd C:\
    if ($?) {echo "Success"}
    else {exit 1}
    date
```

运行命令时处理重启问题

如果使用 Run Command（AWS Systems Manager 中的一项工具）运行脚本来重启托管式节点，建议在脚本中指定退出代码。如果您使用其他一些机制尝试通过脚本重启节点，则即使重启是脚本的最后一步，脚本执行状态也可能无法正确更新。对于 Windows 托管式节点，您需在脚本中指定 `exit 3010`。对于 Linux 和 macOS 托管式节点，需要指定 `exit 194`。退出代码用于指示 AWS Systems Manager Agent (SSM Agent) 重启托管式节点，然后在重启完成后重新启动脚本。在重启开始之前，SSM Agent 会通知云中的 Systems Manager 服务，通信将在服务器重启期间中断。

> **Note**
>
> 重启脚本不能作为 `aws:runDocument` 插件的一部分。如果一个文档包含重启脚本，另一个文档尝试通过 `aws:runDocument` 插件运行该文档，则 SSM Agent 会返回错误。

创建幂等脚本

在开发用于重启托管式节点的脚本时，使脚本具有幂等性，以便脚本执行在重启后从中断的位置继续进行。幂等脚本管理状态并验证是否执行了该操作。当一个步骤设定为仅运行一次时，可以防止该步骤多次运行。

以下是多次重启托管式节点的幂等脚本的概要示例。

```
$name = Get current computer name
If ($name -ne $desiredName)
    {
        Rename computer
        exit 3010
    }

$domain = Get current domain name
If ($domain -ne $desiredDomain)
    {
        Join domain
        exit 3010
    }

If (desired package not installed)
    {
        Install package
        exit 3010
```

```
    }
```

示例

以下脚本示例使用退出代码来重新启动托管式节点。Linux 示例在 Amazon Linux 上安装软件包更新，然后重新启动该节点。Windows Server 示例在节点上安装 Telnet-Client，然后重新启动该节点。

Amazon Linux

```
#!/bin/bash
yum -y update
needs-restarting -r
if [ $? -eq 1 ]
then
        exit 194
else
        exit 0
fi
```

Windows

```
$telnet = Get-WindowsFeature -Name Telnet-Client
if (-not $telnet.Installed)
    {
        # Install Telnet and then send a reboot request to SSM Agent.
        Install-WindowsFeature -Name "Telnet-Client"
        exit 3010
    }
```

了解命令状态

Run Command 是 AWS Systems Manager 中的一项工具，报告关于处理过程中命令经历的不同状态以及处理该命令的每个托管式节点的详细状态信息。您可以使用以下方法监控命令状态：

- 在 Run Command 控制台界面的 Commands（命令）选项卡上选择 Refresh（刷新）图标。
- 使用 AWS Command Line Interface (AWS CLI) 调用 list-commands 或 list-command-invocations。或者，使用 AWS Tools for Windows PowerShell 调用 Get-SSMCommand 或 Get-SSMCommandInvocation。
- 配置 Amazon EventBridge 以响应状态或状态更改。

- 将 Amazon Simple Notification Service (Amazon SNS) 配置为发送所有状态更改或特定状态（例如 `Failed` 或 `TimedOut`）的通知。

Run Command 状态

Run Command 报告以下三个区域的状态详情：插件、调用和总体命令状态。插件是在命令的 SSM 文档中定义的代码执行数据块。有关插件的更多信息，请参阅 命令文档插件参考。

在将一条命令同时发送给多个托管式节点时，针对每个节点的命令的每个副本均为一个命令调用。例如，如果您使用 `AWS-RunShellScript` 文档并将一个 `ifconfig` 命令发送到 20 个 Linux 实例，则该命令具有 20 个调用。每个命令调用都会单独报告状态。给定命令调用的插件也会单独报告状态。

最后，Run Command 包含适用于所有插件和调用的聚合命令状态。聚合命令状态可能不同于插件或调用报告的状态，如下表所述。

> **Note**
>
> 如果您使用 `max-concurrency` 或 `max-errors` 参数在大量托管式节点上运行命令，则命令状态会反映这些参数施加的限制，如下表所述。有关这些参数的更多信息，请参阅 大规模运行命令。

命令插件和调用的详细状态

状态	详细信息
待处理	命令尚未发送到托管式节点，或者 SSM Agent 尚未接收到此类命令。如果代理在等于 Timeout (seconds) (超时 (秒)) 参数和 Execution timeout (执行超时) 参数总和的时间长度之前没有收到命令，则状态将更改为 `Delivery Timed Out` (传输超时)。
InProgress	Systems Manager 正在尝试将命令发送到托管式节点，或者 SSM Agent 已接收到此类命令并且命令已开始在实例上运行。根据所有命令插件的结果，状态会变为 `Success` (成功)、`Failed` (失败)、`Delivery Timed Out` (传输超时) 或 `Execution Timed Out` (执行超时)。例

状态	详细信息
	外：如果代理未在节点上运行或在节点上不可用，则在代理再次可用或达到执行超时限制之前，命令状态将保持为 In Progress（正在进行）。随后，状态会更改为最终状态。
延迟	系统尝试向托管式节点发送命令，但未成功。系统再次重试。

状态	详细信息
成功	在各种条件下都会返回此状态。此状态并不意味着命令已在节点上得到处理。例如，由于您的 PowerShell `ExecutionPolicy` 阻止命令运行，因此在托管式节点上，SSM Agent 可接收到命令，而且命令返回为零的退出代码。这是最终状态。导致命令返回 `Success` 状态的条件是： • 当以单个实例为目标时，命令由托管式节点上的 SSM Agent 接收，并返回一个为零的退出代码。 • 当以多个实例为目标时，失败的调用次数未超过命令中指定的错误阈值。 • 当以多个实例为目标时，至少有 1 次调用成功，而其他调用已超时。指定的错误阈值仍然适用。 • 当以标签为目标时，找不到与该标签关联的实例。 • 当以标签为目标时，失败的调用次数未超过命令中指定的错误阈值。 • 当以标签为目标时，至少有 1 次调用成功，而其他调用已超时。指定的错误阈值仍然适用。 • 您在操作系统级别强制执行了应用程序或策略，这些应用程序或策略可以防止或覆盖返回为零退出代码的命令的执行。 **Note** 当以资源组为目标时，同样的条件也适用。要解决错误或获取有关命令执行的更多信息，请通过返回适当的退出代码

状态	详细信息
	(针对命令失败的非零退出代码) 来发送一个处理错误或异常的命令。
DeliveryTimedOut	命令未在总超时过期之前传输到托管式节点。总超时不计入父命令的 max-errors 限制，但是有助于区别父命令的状态是 Success（成功）、Incomplete （未完成）还是 Delivery Timed Out（传输超时）。这是最终状态。
ExecutionTimedOut	命令自动化在托管式节点上开始，但是命令未在执行超时过期之前完成。执行超时算作失败，这样就会发送一个非零回复，Systems Manager 将退出运行命令自动化的尝试，并报告失败状态。
失败	托管式节点上的命令失败。对于插件，这表示结果代码不为 0。对于命令调用，这表示一个或多个插件的结果代码不为 0。调用失败不计入父命令的 max-errors 限制。这是最终状态。
已取消	命令在完成之前被取消。这是最终状态。
无法传输	命令无法传输到托管式节点。节点可能不存在或可能未响应。无法传输的调用不计入父命令的 max-errors 限制，但是有助于区别父命令的状态是 Success (成功) 还是 Incomplete （未完成)。（例如，如果命令中的所有调用具有 Undeliverable （无法传输) 状态，则返回的命令状态为 Failed (失败)。不过，如果命令具有 5 个调用，其中的 4 个调用返回 Undeliverable （无法传输) 状态，一个调用返回 Success (成功) 状态，则父命令的状态为 Success (成功)。这是最终状态。

状态	详细信息
已终止	父命令超过其 max-errors 限制且系统取消了后续命令调用。这是最终状态。
InvalidPlatform	命令被发送到与所选文档指定的所需平台不匹配的托管式节点。Invalid Platform（无效平台）不计入父命令的最大错误数限制，但是有助于区别父命令的状态是 Success（成功）还是 Failed（失败）。（例如，如果命令中的所有调用具有 Invalid Platform (无法传输) 状态，则返回的命令状态为 Failed (失败)。不过，如果命令具有 5 个调用，其中的 4 个调用返回 Invalid Platform (无法传输) 状态，一个调用返回 Success (成功) 状态，则父命令的状态为 Success (成功)。这是最终状态。
AccessDenied	启动命令的 AWS Identity and Access Management (IAM) 用户或角色无权访问目标托管式节点。Access Denied（拒绝访问）不计入父命令的 max-errors （最大错误数）限制，但是有助于区别父命令的状态是 Success（成功）还是 Failed（失败）。（例如，如果命令中的所有调用具有 Access Denied (无法传输) 状态，则返回的命令状态为 Failed (失败)。不过，如果命令具有 5 个调用，其中的 4 个调用返回 Access Denied (无法传输) 状态，一个调用返回 Success (成功) 状态，则父命令的状态为 Success (成功)。这是最终状态。

命令的详细状态

状态	详细信息
待处理	任何托管式节点上的代理都尚未收到命令。

状态	详细信息
InProgress	命令已发送到至少一个托管式节点，但是在所有节点上都未达到最终状态。
延迟	系统尝试向节点发送命令，但未成功。系统再次重试。
成功	命令由所有指定的或设为目标的托管式节点上的 SSM Agent 接收，并返回了一个为零的退出代码。所有命令调用都已达到最终状态，且未达到 `max-errors` 的值。此状态并不意味着命令已在所有指定的或设为目标的托管式节点上成功得到处理。这是最终状态。 **Note** 要解决错误或获取有关命令执行的更多信息，请通过返回适当的退出代码 (针对命令失败的非零退出代码) 来发送一个处理错误或异常的命令。
DeliveryTimedOut	命令未在总超时过期之前传输到托管式节点。`max-errors` 值或更多命令调用显示 `Delivery Timed Out` 状态。这是最终状态。
失败	托管式节点上的命令失败。`max-errors` 值或更多命令调用显示 `Failed` 状态。这是最终状态。
未完成	已尝试在所有托管式节点上执行命令，且一个或多个调用不具有 `Success`（成功）值。不过，调用失败的次数不足以使状态变为 `Failed`。这是最终状态。
已取消	命令在完成之前被取消。这是最终状态。

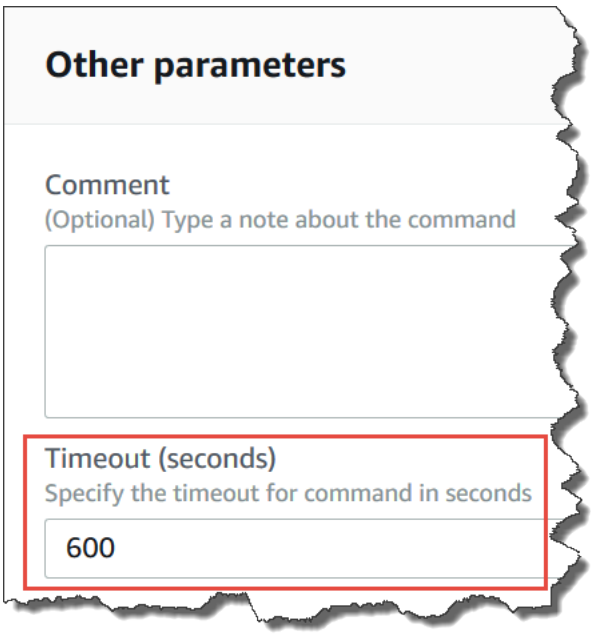
状态	详细信息
RateExceeded	作为命令目标的托管式节点数量超出了待处理调用的账户配额。系统在任何节点上执行命令之前取消了命令。这是最终状态。
AccessDenied	启动命令的用户或角色无权访问目标资源组。AccessDenied 不计入父命令的 max-errors 限制，但是有助于区别父命令的状态是 Success 还是 Failed。（例如，如果命令中的所有调用具有 AccessDenied (拒绝访问) 状态，则返回的命令状态为 Failed (失败)。不过，如果命令具有 5 个调用，其中的 4 个调用返回 AccessDenied (拒绝访问) 状态，1 个调用返回 Success (成功) 状态，则父命令的状态为 Success(成功)。）这是最终状态。
在标签中没有实例	作为命令目标的标签密钥对值或资源组与任何托管式节点都不匹配。这是最终状态。

了解命令超时值

在运行命令时，Systems Manager 强制执行以下超时值。

总超时

在 Systems Manager 控制台中，您可以在 Timeout (seconds)（超时（秒））字段中指定超时值。发送命令后，Run Command 会检查命令是否已过期。如果命令达到命令过期限制（总超时），它会将所有状态为 InProgress (正在进行)、Pending (待处理) 或 Delayed (延迟) 的调用的状态更改为 DeliveryTimedOut (传输超时)。



在更深的技术层面上，总超时（Timeout (seconds)（超时（秒）））是两个超时值的总和，如下所示：

```
Total timeout = "Timeout(seconds)" from the console + "timeoutSeconds":
"{{ executionTimeout }}" from your SSM document
```

例如，在 Systems Manager 控制台中，Timeout (seconds) (超时 (秒)) 的默认值是 600 秒。如果使用 `AWS-RunShellScript` SSM 文档运行命令，则 "timeoutSeconds": "{{ executionTimeout }}" 的默认值为 3600 秒，如以下示例文档所示：

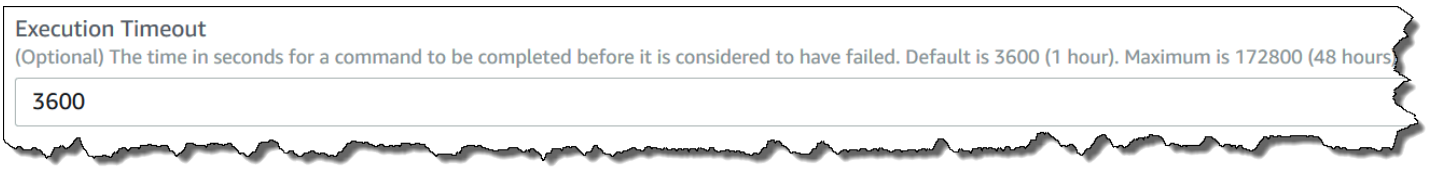
```
"executionTimeout": {
    "type": "String",
    "default": "3600",

"runtimeConfig": {
  "aws:runShellScript": {
    "properties": [
      {
        "timeoutSeconds": "{{ executionTimeout }}"
```

这意味着，在系统将命令状态设置为 `DeliveryTimedOut` 之前，该命令运行了 4200 秒（70 分钟）。

执行超时

在 Systems Manager 控制台中，您可以在 Execution Timeout (执行超时) 字段中指定执行超时值（如果可用）。并非所有 SSM 文档都要求指定执行超时值。Execution Timeout（执行超时）字段仅在 SSM 文档中定义了相应的输入参数时才会显示。如果已指定该值，则命令必须在此时段内完成。

> **Note**
>
> Run Command 依靠 SSM Agent 文档最终响应来确定命令是否已传输给代理。SSM Agent 必须发送一个 `ExecutionTimedOut` 信号，才能将调用或命令标记为 `ExecutionTimedOut`。



默认执行超时

如果 SSM 文档不要求您明确指定执行超时值，则 Systems Manager 会强制执行硬编码的默认执行超时。

Systems Manager 如何报告超时

如果 Systems Manager 收到来自目标上的 SSM Agent 的 `execution timeout` 回复，则 Systems Manager 会将命令调用标记为 `executionTimeout`。

如果 Run Command 没有收到 SSM Agent 文档最终响应，则命令调用被标记为 `deliveryTimeout`。

为了确定目标的超时状态，SSM Agent 将合并 SSM 文档的所有参数和内容来计算 `executionTimeout`。如果 SSM Agent 确定某条命令已超时，它会向服务发送 `executionTimeout`。

Timeout (seconds) (超时 (秒)) 的默认值为 3600 秒。Execution Timeout (执行超时) 的默认值也为 3600 秒。因此，命令的总超时默认为 7200 秒。

> **Note**
>
> SSM Agent 采用不同的方式处理 `executionTimeout`，具体取决于 SSM 文档的类型和文档版本。

Run Command 演练

本节中的演练说明如何借助 AWS Command Line Interface（AWS CLI）或 AWS Tools for Windows PowerShell，使用 Run Command（AWS Systems Manager 中的一项工具）来运行命令。

内容

- 使用 Run Command 更新软件
- 演练：将 AWS CLI 与 Run Command 结合使用
- 演练：将 AWS Tools for Windows PowerShell 与 Run Command 结合使用

您还可以查看以下参考中的示例命令。

- Systems Manager AWS CLI 参考
- AWS Tools for Windows PowerShell - AWS Systems Manager

使用 Run Command 更新软件

以下步骤说明如何在托管节点上更新软件。

使用 Run Command 更新 SSM Agent

以下过程介绍如何更新在托管式节点上运行的 SSM Agent。您可以更新到最新版本的 SSM Agent 或降级到较旧版本。在运行命令时，系统将从 AWS 下载并安装需要的版本，然后卸载运行命令前存在的版本。如果此过程中出现错误，系统将回滚到命令运行之前服务器上的版本，并且命令状态将显示命令失败。

> **Note**
>
> 如果实例运行 macOS 版本 11.0（Big Sur）或更高版本，则该实例必须带有 SSM Agent 版本 3.1.941.0 或更高版本才能运行 AWS-UpdateSSMAgent 文档。如果实例运行 3.1.941.0 之前发布的 SSM Agent 版本，那么您可以通过运行 `brew update` 和 `brew upgrade amazon-ssm-agent` 命令来更新 SSM Agent，从而运行 AWS-UpdateSSMAgent 文档。

要获得有关 SSM Agent 更新的通知，请在 GitHub 上订阅 SSM Agent 发布说明页面。

使用 Run Command 更新 SSM Agent

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Run Command。
3. 选择 Run command（运行命令）。
4. 在 Command document (命令文档) 列表中，请选择 **AWS-UpdateSSMAgent**。
5. 在命令参数部分中，根据需要为以下参数指定值：
   a. （可选）对于 Version (版本)，输入要安装的 SSM Agent 的版本。您可以安装代理的较旧版本。如果您不指定版本，则服务将安装最新版本。
   b. （可选）对于 Allow Downgrade (允许降级)，选择 true (真) 以安装 SSM Agent的早期版本。如果选择此选项，需要指定较早的版本号。选择 false 以仅安装此服务的最新版本。
6. 在 Targets（目标）部分中，通过指定标签、手动选择实例或边缘设备或指定资源组，选择要在其上运行此操作的托管式节点。

   > **Tip**
   > 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

7. 对于 Other parameters（其他参数）：
   - 对于 Comment（注释），请输入有关此命令的信息。
   - 对于 Timeout (seconds) (超时 (秒))，请指定在整个命令执行失败之前系统等待的秒数。
8. 对于 Rate control（速率控制）：
   - 对于 Concurrency（并发），请指定要同时运行该命令的托管式节点的数量或百分比。

   > **Note**
   > 如果您通过指定应用于托管式节点的标签或指定 AWS Resource Groups 来选择目标，但不确定有多少个托管式节点已被设为目标，则可通过指定百分比来限制可同时运行该文档的目标的数量。

- 对于 Error threshold（错误阈值），请指定当命令在一定数量或百分比的节点上失败后，何时在其他托管式节点上停止运行该命令。例如，如果您指定三个错误，Systems Manager 将在收到第四个错误时停止发送该命令。仍在处理该命令的托管式节点也可能发送错误。

9. （可选）对于 输出选项，要将命令输出保存到文件，请选中 将命令输出写入 S3 存储桶 框。在输入框中输入存储桶和前缀（文件夹）名称。

> **Note**
>
> 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给实例的实例配置文件（适用于 EC2 实例）或 IAM 服务角色（混合激活的计算机）的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确保与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

10. 在 SNS 通知部分，如果需要发送有关命令执行状态的通知，请选中 Enable SNS notifications（启用 SNS 通知）复选框。

    有关为 Run Command 配置 Amazon SNS 通知的更多信息，请参阅 使用 Amazon SNS 通知监控 Systems Manager 状态更改。

11. 选择 Run（运行）。

使用 Run Command 更新 PowerShell

以下过程介绍如何在 Windows Server 2012 和 2012 R2 托管式节点上将 PowerShell 更新到版本 5.1。此过程中提供的脚本将下载 Windows 管理框架 (WMF) 5.1 版更新，并开始安装此更新。在此过程中，节点会重启，因为在安装 WMF 5.1 时要求这么做。完成下载和安装更新大约需要 5 分钟。

要使用 Run Command 更新 PowerShell，请执行以下步骤：

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Run Command。
3. 选择 Run command（运行命令）。
4. 在 Command document（命令文档）列表中，请选择 **AWS-RunPowerShellScript**。
5. 在 Commands (命令) 部分中，粘贴以下适用于您的操作系统的命令。

Windows Server 2012 R2

```
Set-Location -Path "C:\Windows\Temp"

Invoke-WebRequest "https://go.microsoft.com/fwlink/?linkid=839516" -OutFile
 "Win8.1AndW2K12R2-KB3191564-x64.msu"

Start-Process -FilePath "$env:systemroot\system32\wusa.exe" -Verb RunAs -
ArgumentList ('Win8.1AndW2K12R2-KB3191564-x64.msu', '/quiet')
```

Windows Server 2012

```
Set-Location -Path "C:\Windows\Temp"

Invoke-WebRequest "https://go.microsoft.com/fwlink/?linkid=839513" -OutFile
 "W2K12-KB3191565-x64.msu"

Start-Process -FilePath "$env:systemroot\system32\wusa.exe" -Verb RunAs -
ArgumentList ('W2K12-KB3191565-x64.msu', '/quiet')
```

6. 在 Targets（目标）部分中，通过指定标签、手动选择实例或边缘设备或指定资源组，选择要在其上运行此操作的托管式节点。

> **Tip**
>
> 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

7. 对于 Other parameters（其他参数）：
   - 对于 Comment（注释），请输入有关此命令的信息。
   - 对于 Timeout (seconds) (超时 (秒))，请指定在整个命令执行失败之前系统等待的秒数。
8. 对于 Rate control（速率控制）：
   - 对于 Concurrency（并发），请指定要同时运行该命令的托管式节点的数量或百分比。

> **Note**
>
> 如果您通过指定应用于托管式节点的标签或指定 AWS Resource Groups 来选择目标，但不确定有多少个托管式节点已被设为目标，则可通过指定百分比来限制可同时运行该文档的目标的数量。

- 对于 Error threshold（错误阈值），请指定当命令在一定数量或百分比的节点上失败后，何时在其他托管式节点上停止运行该命令。例如，如果您指定三个错误，Systems Manager 将在收到第四个错误时停止发送该命令。仍在处理该命令的托管式节点也可能发送错误。

9. （可选）对于 输出选项，要将命令输出保存到文件，请选中 将命令输出写入 S3 存储桶 框。在输入框中输入存储桶和前缀（文件夹）名称。

> **Note**
>
> 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给实例的实例配置文件（适用于 EC2 实例）或 IAM 服务角色（混合激活的计算机）的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确保与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

10. 在 SNS 通知部分，如果需要发送有关命令执行状态的通知，请选中 Enable SNS notifications（启用 SNS 通知）复选框。

    有关为 Run Command 配置 Amazon SNS 通知的更多信息，请参阅 使用 Amazon SNS 通知监控 Systems Manager 状态更改。

11. 选择 Run（运行）。

在托管式节点重启且更新安装完成后，连接到您的节点，以确认 PowerShell 已成功升级到版本 5.1。要检查您的节点上的 PowerShell 版本，打开 PowerShell 并输入 `$PSVersionTable`。如果升级成功，输出表中的 `PSVersion` 值将显示为 5.1。

如果 `PSVersion` 值不是 5.1（例如 3.0 或 4.0），请在事件查看器中查看 Windows Logs (Windows 日志) 下的 Setup (设置) 日志。这些日志说明了更新安装失败的原因。

演练：将 AWS CLI 与 Run Command 结合使用

以下示例演练介绍了如何使用 AWS Command Line Interface (AWS CLI) 查看有关命令和命令参数的信息、如何运行命令以及如何查看这些命令的状态。

> **Important**
>
> 仅允许受信任的管理员使用本主题中所示的 AWS Systems Manager 预配置文档。在 Systems Manager 文档中指定的命令和脚本需要管理权限才能在您的托管式节点上运行。如果用户有权运行任何预定义的 Systems Manager 文档（任何以 `AWS-` 开头的文档），则该用户也具有节点的管理员访问权限。对于所有其他用户，您应创建限制性文档并与特定用户共享这些文档。

主题

- 步骤 1：入门
- 步骤 2：运行 Shell 脚本以查看资源详细信息
- 步骤 3：使用 AWS-RunShellScript 文档发送简单命令
- 步骤 4：使用 Run Command 运行简单 Python 脚本
- 步骤 5：使用 Run Command 运行 Bash 脚本

步骤 1：入门

您必须具有要配置的托管式节点的管理员权限，或必须已获得 AWS Identity and Access Management (IAM) 中的适当权限。另请注意，此示例使用美国东部（俄亥俄州）区域（us-east-2）。Run Command 在《Amazon Web Services 一般参考》的 Systems Manager service endpoints 列出的 AWS 区域 中可用。有关更多信息，请参阅 为 AWS Systems Manager 设置托管节点。

使用 AWS CLI 运行命令

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本。
2. 列出所有可用文档。

   此命令将基于 IAM 权限列出您的账户可用的所有文档。

   ```
   aws ssm list-documents
   ```

3. 确认托管式节点已准备好接收命令。

   以下命令的输出会显示托管式节点是否处于联机状态。

   Linux & macOS

   ```
   aws ssm describe-instance-information \
       --output text --query "InstanceInformationList[*]"
   ```

   Windows

   ```
   aws ssm describe-instance-information ^
       --output text --query "InstanceInformationList[*]"
   ```

4. 运行以下命令来查看有关特定托管式节点的详细信息。

   > **Note**
   >
   > 要运行本演练中的命令，请替换实例和命令 ID。对于托管式 AWS IoT Greengrass 核心设备，对实例 ID 使用 mi-*ID_NUMBER*。命令 ID 将作为对 send-command 的响应返回。可从 Fleet Manager（AWS Systems Manager 中的一项工具）中获得实例 ID。

   Linux & macOS

   ```
   aws ssm describe-instance-information \
       --instance-information-filter-list key=InstanceIds,valueSet=instance-ID
   ```

   Windows

   ```
   aws ssm describe-instance-information ^
       --instance-information-filter-list key=InstanceIds,valueSet=instance-ID
   ```

步骤 2：运行 Shell 脚本以查看资源详细信息

利用 Run Command 和 `AWS-RunShellScript` 文档，您可以在托管式节点上运行任何命令或脚本，就像您已在本地登录一样。

查看说明和可用参数

运行以下命令，查看 Systems Manager JSON 文档的描述。

Linux & macOS

```
aws ssm describe-document \
    --name "AWS-RunShellScript" \
    --query "[Document.Name,Document.Description]"
```

Windows

```
aws ssm describe-document ^
    --name "AWS-RunShellScript" ^
    --query "[Document.Name,Document.Description]"
```

运行以下命令，查看可用参数和有关这些参数的详细信息。

Linux & macOS

```
aws ssm describe-document \
    --name "AWS-RunShellScript" \
    --query "Document.Parameters[*]"
```

Windows

```
aws ssm describe-document ^
    --name "AWS-RunShellScript" ^
    --query "Document.Parameters[*]"
```

步骤 3：使用 **AWS-RunShellScript** 文档发送简单命令

运行以下命令，获取 Linux 托管式节点的 IP 信息。

如果您将 Windows Server 托管式节点设为目标，请将 `document-name` 更改为 `AWS-RunPowerShellScript`，并将 `command` 从 `ifconfig` 更改为 `ipconfig`。

Linux & macOS

```
aws ssm send-command \
    --instance-ids "instance-ID" \
```

```
    --document-name "AWS-RunShellScript" \
    --comment "IP config" \
    --parameters commands=ifconfig \
    --output text
```

Windows

```
aws ssm send-command ^
    --instance-ids "instance-ID" ^
    --document-name "AWS-RunShellScript" ^
    --comment "IP config" ^
    --parameters commands=ifconfig ^
    --output text
```

使用响应数据获取命令信息

以下命令使用从上一个命令返回的命令 ID 来获取命令执行的详细信息和响应数据。如果命令已完成，系统将返回响应数据。如果命令执行显示 "Pending" 或 "InProgress"，您再次运行此命令来查看响应数据。

Linux & macOS

```
aws ssm list-command-invocations \
    --command-id $sh-command-id \
    --details
```

Windows

```
aws ssm list-command-invocations ^
    --command-id $sh-command-id ^
    --details
```

标识用户

以下命令将显示运行命令的默认用户。

Linux & macOS

```
sh_command_id=$(aws ssm send-command \
    --instance-ids "instance-ID" \
```

```
    --document-name "AWS-RunShellScript" \
    --comment "Demo run shell script on Linux managed node" \
    --parameters commands=whoami \
    --output text \
    --query "Command.CommandId")
```

获取命令状态

以下命令使用命令 ID 获取托管式节点上的命令执行状态。此示例使用上一个命令中返回的命令 ID。

Linux & macOS

```
aws ssm list-commands \
    --command-id "command-ID"
```

Windows

```
aws ssm list-commands ^
    --command-id "command-ID"
```

获取命令详细信息

以下命令使用上一命令中的命令 ID 来获取每个托管式节点的命令执行状态。

Linux & macOS

```
aws ssm list-command-invocations \
    --command-id "command-ID" \
    --details
```

Windows

```
aws ssm list-command-invocations ^
    --command-id "command-ID" ^
    --details
```

获取包含特定托管式节点的响应数据的命令信息

以下命令会返回特定托管式节点的原始 `aws ssm send-command` 请求的输出。

Linux & macOS

```
aws ssm list-command-invocations \
    --instance-id instance-ID \
    --command-id "command-ID" \
    --details
```

Windows

```
aws ssm list-command-invocations ^
    --instance-id instance-ID ^
    --command-id "command-ID" ^
    --details
```

显示 Python 版本

以下命令会返回在节点上运行的 Python 的版本。

Linux & macOS

```
sh_command_id=$(aws ssm send-command \
    --instance-ids "instance-ID" \
    --document-name "AWS-RunShellScript" \
    --comment "Demo run shell script on Linux Instances" \
    --parameters commands='python -V' \
    --output text --query "Command.CommandId") \
    sh -c 'aws ssm list-command-invocations \
    --command-id "$sh_command_id" \
    --details \
    --query "CommandInvocations[].CommandPlugins[].{Status:Status,Output:Output}"'
```

步骤 4：使用 Run Command 运行简单 Python 脚本

以下命令使用 Run Command 运行一个简单的 Python“Hello World”脚本。

Linux & macOS

```
sh_command_id=$(aws ssm send-command \
    --instance-ids "instance-ID" \
    --document-name "AWS-RunShellScript" \
    --comment "Demo run shell script on Linux Instances" \
```

```
    --parameters '{"commands":["#!/usr/bin/python","print \"Hello World from python
\""]}' \
    --output text \
    --query "Command.CommandId") \
    sh -c 'aws ssm list-command-invocations \
    --command-id "$sh_command_id" \
    --details \
    --query "CommandInvocations[].CommandPlugins[].{Status:Status,Output:Output}"'
```

步骤 5：使用 Run Command 运行 Bash 脚本

此部分中的示例演示如何使用 Run Command 运行以下 Bash 脚本。

有关使用 Run Command 运行存储在远程位置的脚本的示例，请参阅从 Amazon S3 运行脚本和从 GitHub 运行脚本。

```
#!/bin/bash
yum -y update
yum install -y ruby
cd /home/ec2-user
curl -O https://aws-codedeploy-us-east-2.s3.amazonaws.com/latest/install
chmod +x ./install
./install auto
```

此脚本将在 Amazon Linux 和 Red Hat Enterprise Linux (RHEL) 实例上安装 AWS CodeDeploy 代理，如《AWS CodeDeploy 用户指南》中的为 CodeDeploy 创建 Amazon EC2 实例所述。

该脚本将从 AWS 托管的 S3 存储桶安装 CodeDeploy 代理程序，该存储桶位于美国东部（俄亥俄州）区域（us-east-2）（`aws-codedeploy-us-east-2`）。

在 AWS CLI 命令中运行 Bash 脚本

以下示例演示了如何使用 `--parameters` 选项在 CLI 命令中包含 Bash 脚本。

Linux & macOS

```
aws ssm send-command \
    --document-name "AWS-RunShellScript" \
    --targets '[{"Key":"InstanceIds","Values":["instance-id"]}]' \
    --parameters '{"commands":["#!/bin/bash","yum -y update","yum
 install -y ruby","cd /home/ec2-user","curl -O https://aws-codedeploy-us-
east-2.s3.amazonaws.com/latest/install","chmod +x ./install","./install auto"]}'
```

在 JSON 文件中运行 Bash 脚本

在以下示例中，Bash 脚本的内容存储在一个 JSON 文件中，该文件通过使用 `--cli-input-json` 选项包含在命令中。

Linux & macOS

```
aws ssm send-command \
    --document-name "AWS-RunShellScript" \
    --targets "Key=InstanceIds,Values=instance-id" \
    --cli-input-json file://installCodeDeployAgent.json
```

Windows

```
aws ssm send-command ^
    --document-name "AWS-RunShellScript" ^
    --targets "Key=InstanceIds,Values=instance-id" ^
    --cli-input-json file://installCodeDeployAgent.json
```

引用的 `installCodeDeployAgent.json` 文件的内容如以下示例所示。

```
{
    "Parameters": {
        "commands": [
            "#!/bin/bash",
            "yum -y update",
            "yum install -y ruby",
            "cd /home/ec2-user",
            "curl -O https://aws-codedeploy-us-east-2.s3.amazonaws.com/latest/install",
            "chmod +x ./install",
            "./install auto"
        ]
    }
}
```

演练：将 AWS Tools for Windows PowerShell 与 Run Command 结合使用

以下示例说明如何使用 AWS Tools for Windows PowerShell 查看有关命令和命令参数的信息、如何运行命令以及如何查看这些命令的状态。本演练为每个预定义的 AWS Systems Manager 文档包含了一个示例。

> **Important**
>
> 仅允许受信任的管理员使用本主题中所示的 Systems Manager 预配置文档。在 Systems Manager 文档中指定的命令和脚本需要管理权限才能在您的托管式节点上运行。如果用户有权运行任何预定义的 Systems Manager 文档（任何以 AWS 开头的文档），则该用户也具有节点的管理员访问权限。对于所有其他用户，您应创建限制性文档并与特定用户共享这些文档。

主题

- 配置 AWS Tools for Windows PowerShell 会话设置
- 列出所有可用文档
- 运行 PowerShell 命令或脚本
- 使用 AWS-InstallApplication 文档安装应用程序
- 使用 AWS-InstallPowerShellModule JSON 文档安装 PowerShell 模块
- 使用 AWS-JoinDirectoryServiceDomain JSON 文档将托管式节点加入域中
- 使用 AWS-ConfigureCloudWatch 文档将 Windows 指标发送到 Amazon CloudWatch Logs
- 使用 AWS-UpdateEC2Config 文档更新 EC2Config
- 使用 AWS-ConfigureWindowsUpdate 文档启用或关闭 Windows 自动更新
- 使用 Run Command 管理 Windows 更新

配置 AWS Tools for Windows PowerShell 会话设置

指定您的凭证

在本地计算机上打开 Tools for Windows PowerShell 并运行以下命令以指定您的凭证。您必须具有要配置的托管式节点的管理员权限，或必须已获得 AWS Identity and Access Management (IAM) 中的适当权限。有关更多信息，请参阅 为 AWS Systems Manager 设置托管节点。

```
Set-AWSCredentials -AccessKey key-name -SecretKey key-name
```

设置默认 AWS 区域

运行以下命令为 PowerShell 会话设置区域。此示例使用美国东部（俄亥俄州）区域（us-east-2）。Run Command 在《Amazon Web Services 一般参考》的 Systems Manager service endpoints 列出的 AWS 区域 中可用。

```
Set-DefaultAWSRegion `
    -Region us-east-2
```

列出所有可用文档

此命令将列出您的账户可用的所有文档。

```
Get-SSMDocumentList
```

运行 PowerShell 命令或脚本

利用 Run Command 和 AWS-RunPowerShell 文档，您可以在托管式节点上运行任何命令或脚本，就像您已在本地登录一样。您可以发出命令或输入本地脚本的路径以运行命令。

> **Note**
>
> 有关在使用 Run Command 调用脚本时重启托管式节点的信息，请参阅 运行命令时处理重启问题。

查看说明和可用参数

```
Get-SSMDocumentDescription `
    -Name "AWS-RunPowerShellScript"
```

查看有关参数的更多信息

```
Get-SSMDocumentDescription `
    -Name "AWS-RunPowerShellScript" | Select -ExpandProperty Parameters
```

使用 **AWS-RunPowerShellScript** 文档发送命令

以下命令在两个托管式节点上显示 "C:\Users" 目录的内容和 "C:\" 目录的内容。

```
$runPSCommand = Send-SSMCommand `
    -InstanceIds @("instance-ID-1", "instance-ID-2") `
    -DocumentName "AWS-RunPowerShellScript" `
    -Comment "Demo AWS-RunPowerShellScript with two instances" `
    -Parameter @{'commands'=@('dir C:\Users', 'dir C:\')}
```

获取命令请求详细信息

以下命令使用 CommandId 获取两个托管式节点上的命令执行状态。此示例使用上一个命令中返回的 CommandId。

```
Get-SSMCommand `
    -CommandId $runPSCommand.CommandId
```

此示例中命令的状态可以是 Success、Pending 或 InProgress。

获取每个托管式节点的命令信息

以下命令使用上一命令中的 CommandId 来获取每个托管式节点的命令执行状态。

```
Get-SSMCommandInvocation `
    -CommandId $runPSCommand.CommandId
```

获取包含特定托管式节点的响应数据的命令信息

以下命令会返回特定托管式节点的原始 Send-SSMCommand 的输出。

```
Get-SSMCommandInvocation `
    -CommandId $runPSCommand.CommandId `
    -Details $true `
    -InstanceId instance-ID | Select -ExpandProperty CommandPlugins
```

取消命令

以下命令取消 AWS-RunPowerShellScript 文档的 Send-SSMCommand。

```
$cancelCommand = Send-SSMCommand `
    -InstanceIds @("instance-ID-1","instance-ID-2") `
    -DocumentName "AWS-RunPowerShellScript" `
    -Comment "Demo AWS-RunPowerShellScript with two instances" `
    -Parameter @{'commands'='Start-Sleep -Seconds 120; dir C:\'}

Stop-SSMCommand -CommandId $cancelCommand.CommandId
```

查看命令状态

以下命令检查 Cancel 命令的状态。

```
Get-SSMCommand `
    -CommandId $cancelCommand.CommandId
```

使用 **AWS-InstallApplication** 文档安装应用程序

利用 Run Command 和 AWS-InstallApplication 文档，您可以在托管式节点上安装、修复或卸载应用程序。该命令需要 MSI 的路径或地址。

> **Note**
>
> 有关在使用 Run Command 调用脚本时重启托管式节点的信息，请参阅 运行命令时处理重启问题。

查看说明和可用参数

```
Get-SSMDocumentDescription `
    -Name "AWS-InstallApplication"
```

查看有关参数的更多信息

```
Get-SSMDocumentDescription `
    -Name "AWS-InstallApplication" | Select -ExpandProperty Parameters
```

使用 **AWS-InstallApplication** 文档发送命令

以下命令将以无人值守的模式在托管式节点上安装 Python 版本，并将输出记录在 C: 驱动器上的本地文本文件中。

```
$installAppCommand = Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-InstallApplication" `
    -Parameter @{'source'='https://www.python.org/ftp/python/2.7.9/python-2.7.9.msi';
 'parameters'='/norestart /quiet /log c:\pythoninstall.txt'}
```

获取每个托管式节点的命令信息

以下命令使用 CommandId 获取命令执行的状态。

```
Get-SSMCommandInvocation `
    -CommandId $installAppCommand.CommandId `
```

```
    -Details $true
```

获取包含特定托管式节点的响应数据的命令信息

以下命令返回 Python 安装的结果。

```
Get-SSMCommandInvocation `
    -CommandId $installAppCommand.CommandId `
    -Details $true `
    -InstanceId instance-ID | Select -ExpandProperty CommandPlugins
```

使用 **AWS-InstallPowerShellModule** JSON 文档安装 PowerShell 模块

您可使用 Run Command 在托管式节点上安装 PowerShell 模块。有关 PowerShell 模块的更多信息，请参阅 Windows PowerShell 模块。

查看说明和可用参数

```
Get-SSMDocumentDescription `
    -Name "AWS-InstallPowerShellModule"
```

查看有关参数的更多信息

```
Get-SSMDocumentDescription `
    -Name "AWS-InstallPowerShellModule" | Select -ExpandProperty Parameters
```

安装 PowerShell 模块

以下命令下载 EZOut.zip 文件，安装该文件，然后运行另一个命令来安装 XPS Viewer。最后，输出此命令将上传到一个名为“amzn-s3-demo-bucket”的 S3 存储桶中。

```
$installPSCommand = Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-InstallPowerShellModule" `
    -Parameter @{'source'='https://gallery.technet.microsoft.com/EZOut-33ae0fb7/
file/110351/1/EZOut.zip';'commands'=@('Add-WindowsFeature -name XPS-Viewer -restart')}
 `
    -OutputS3BucketName amzn-s3-demo-bucket
```

获取每个托管式节点的命令信息

以下命令使用 CommandId 获取命令执行的状态。

```
Get-SSMCommandInvocation `
    -CommandId $installPSCommand.CommandId `
    -Details $true
```

获取包含托管式节点的响应数据的命令信息

以下命令返回特定 CommandId 的原始 Send-SSMCommand 的输出。

```
Get-SSMCommandInvocation `
    -CommandId $installPSCommand.CommandId `
    -Details $true | Select -ExpandProperty CommandPlugins
```

使用 **AWS-JoinDirectoryServiceDomain** JSON 文档将托管式节点加入域中

借助 Run Command，您可以将托管式节点快速加入 AWS Directory Service 域中。执行此命令前，请创建目录。还建议您了解 AWS Directory Service 的更多信息。有关更多信息，请参阅 AWS Directory Service 管理指南。

您只能将托管式节点加入域中，无法从域中删除节点。

> **Note**
>
> 有关在使用 Run Command 调用脚本时托管式节点的信息，请参阅 运行命令时处理重启问题。

查看说明和可用参数

```
Get-SSMDocumentDescription `
    -Name "AWS-JoinDirectoryServiceDomain"
```

查看有关参数的更多信息

```
Get-SSMDocumentDescription `
    -Name "AWS-JoinDirectoryServiceDomain" | Select -ExpandProperty Parameters
```

将托管式节点加入域中

以下命令可将托管式节点加入给定的 AWS Directory Service 域中，并将生成的任何输出上传至示例 Amazon Simple Storage Service（Amazon S3）存储桶。

```
$domainJoinCommand = Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-JoinDirectoryServiceDomain" `
    -Parameter @{'directoryId'='d-example01'; 'directoryName'='ssm.example.com';
 'dnsIpAddresses'=@('192.168.10.195', '192.168.20.97')} `
    -OutputS3BucketName amzn-s3-demo-bucket
```

获取每个托管式节点的命令信息

以下命令使用 CommandId 获取命令执行的状态。

```
Get-SSMCommandInvocation `
    -CommandId $domainJoinCommand.CommandId `
    -Details $true
```

获取包含托管式节点的响应数据的命令信息

此命令返回特定 CommandId 的原始 Send-SSMCommand 的输出。

```
Get-SSMCommandInvocation `
    -CommandId $domainJoinCommand.CommandId `
    -Details $true | Select -ExpandProperty CommandPlugins
```

使用 **AWS-ConfigureCloudWatch** 文档将 Windows 指标发送到 Amazon CloudWatch Logs

您可以将应用程序、系统、安全和 Windows 事件跟踪 (ETW) 日志中的 Windows Server 消息发送到 Amazon CloudWatch Logs。在首次允许日志记录时，Systems Manager 会发送从您开始上传该应用程序、系统、安全和 ETW 日志时起一 (1) 分钟内生成的所有日志。其中不包括在此时间之前产生的日志。如果您关闭日志记录并在以后再次启用日志记录，Systems Manager 会从其上次停止的时间继续发送日志。对于任何自定义日志文件和 Internet Information Services (IIS) 日志，Systems Manager 会从头读取日志文件。此外，Systems Manager 还可以将性能计数器数据发送到 CloudWatch Logs。

如果先前在 EC2Config 中启用了 CloudWatch 集成，则 Systems Manager 设置会覆盖 C:\Program Files\Amazon\EC2ConfigService\Settings\AWS.EC2.Windows.CloudWatch.json 文件中本地存储在托管式节点上的任何设置。有关使用 EC2Config 管理单个托管式节点上的性能计数器和日志的更多信息，请参阅 Amazon CloudWatch 用户指南中的使用 CloudWatch 代理从 Amazon EC2 实例和本地服务器收集指标和日志。

查看说明和可用参数

```
Get-SSMDocumentDescription `
    -Name "AWS-ConfigureCloudWatch"
```

查看有关参数的更多信息

```
Get-SSMDocumentDescription `
    -Name "AWS-ConfigureCloudWatch" | Select -ExpandProperty Parameters
```

将应用程序日志发送到 CloudWatch

以下命令可配置托管式节点并将 Windows 应用程序日志移至 CloudWatch。

```
$cloudWatchCommand = Send-SSMCommand `
    -InstanceID instance-ID `
    -DocumentName "AWS-ConfigureCloudWatch" `
    -Parameter @{'properties'='{"engineConfiguration": {"PollInterval":"00:00:15",
 "Components":[{"Id":"ApplicationEventLog",
 "FullName":"AWS.EC2.Windows.CloudWatch.EventLog.EventLogInputComponent,AWS.EC2.Windows.CloudWa
 "Parameters":{"LogName":"Application", "Levels":"7"}},{"Id":"CloudWatch",
 "FullName":"AWS.EC2.Windows.CloudWatch.CloudWatchLogsOutput,AWS.EC2.Windows.CloudWatch",
 "Parameters":{"Region":"region", "LogGroup":"my-log-group", "LogStream":"instance-
id"}}], "Flows":{"Flows":["ApplicationEventLog,CloudWatch"]}}}'}
```

获取每个托管式节点的命令信息

以下命令使用 CommandId 获取命令执行的状态。

```
Get-SSMCommandInvocation `
    -CommandId $cloudWatchCommand.CommandId `
    -Details $true
```

获取包含特定托管式节点的响应数据的命令信息

以下命令返回 Amazon CloudWatch 配置的结果。

```
Get-SSMCommandInvocation `
    -CommandId $cloudWatchCommand.CommandId `
    -Details $true `
    -InstanceId instance-ID | Select -ExpandProperty CommandPlugins
```

使用 **AWS-ConfigureCloudWatch** 文档将性能计数器发送到 CloudWatch

以下演示命令将性能计数器数据上传到 CloudWatch。有关更多信息，请参阅《Amazon CloudWatch 用户指南》。

```
$cloudWatchMetricsCommand = Send-SSMCommand `
    -InstanceID instance-ID `
    -DocumentName "AWS-ConfigureCloudWatch" `
    -Parameter @{'properties'='{"engineConfiguration": {"PollInterval":"00:00:15",
 "Components":[{"Id":"PerformanceCounter",
 "FullName":"AWS.EC2.Windows.CloudWatch.PerformanceCounterComponent.PerformanceCounterInputComp
 "Parameters":{"CategoryName":"Memory", "CounterName":"Available
 MBytes", "InstanceName":"", "MetricName":"AvailableMemory",
 "Unit":"Megabytes","DimensionName":"", "DimensionValue":""}},{"Id":"CloudWatch",
 "FullName":"AWS.EC2.Windows.CloudWatch.CloudWatchOutputComponent,AWS.EC2.Windows.Cl
 "Parameters":{"AccessKey":"", "SecretKey":"","Region":"region", "NameSpace":"Windows-
Default"}}], "Flows":{"Flows":["PerformanceCounter,CloudWatch"]}}}'}
```

使用 **AWS-UpdateEC2Config** 文档更新 EC2Config

利用 Run Command 和 AWS-EC2ConfigUpdate 文档，您可以更新在 Windows Server 托管式节点上运行的 EC2Config 服务。此命令可将 EC2Config 服务更新为最新版本或您指定的版本。

查看说明和可用参数

```
Get-SSMDocumentDescription `
    -Name "AWS-UpdateEC2Config"
```

查看有关参数的更多信息

```
Get-SSMDocumentDescription `
    -Name "AWS-UpdateEC2Config" | Select -ExpandProperty Parameters
```

将 EC2Config 更新为最新版本

```
$ec2ConfigCommand = Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-UpdateEC2Config"
```

获取包含托管式节点的响应数据的命令信息

此命令从上一个 Send-SSMCommand 返回指定命令的输出。

```
Get-SSMCommandInvocation `
    -CommandId $ec2ConfigCommand.CommandId `
    -Details $true `
    -InstanceId instance-ID | Select -ExpandProperty CommandPlugins
```

将 EC2Config 更新为特定版本

以下命令将 EC2Config 降级到较旧版本。

```
Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-UpdateEC2Config" `
    -Parameter @{'version'='4.9.3519'; 'allowDowngrade'='true'}
```

使用 **AWS-ConfigureWindowsUpdate** 文档启用或关闭 Windows 自动更新

利用 Run Command 和 AWS-ConfigureWindowsUpdate 文档，您可以在 Windows Server 托管式节点上启用或关闭 Windows 自动更新。此命令将 Windows 更新代理配置为在您指定的日期和时间下载并安装 Windows 更新。如果更新需要重启，托管式节点将在安装更新 15 分钟后自动重启。利用此命令，您还可以将 Windows 更新配置为检查更新但不安装更新。AWS-ConfigureWindowsUpdate 文档可与 Windows Server 2008、2008 R2、2012、2012 R2 和 2016 兼容。

查看说明和可用参数

```
Get-SSMDocumentDescription `
    -Name "AWS-ConfigureWindowsUpdate"
```

查看有关参数的更多信息

```
Get-SSMDocumentDescription `
    -Name "AWS-ConfigureWindowsUpdate" | Select -ExpandProperty Parameters
```

启用 Windows 自动更新

以下命令将 Windows 更新配置为在每天晚上 22:00 自动下载和安装更新。

```
$configureWindowsUpdateCommand = Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-ConfigureWindowsUpdate" `
    -Parameters @{'updateLevel'='InstallUpdatesAutomatically';
 'scheduledInstallDay'='Daily'; 'scheduledInstallTime'='22:00'}
```

查看用于允许 Windows 自动更新的命令状态

以下命令使用 CommandId 获取用于允许 Windows 自动更新的命令执行的状态。

```
Get-SSMCommandInvocation `
    -Details $true `
    -CommandId $configureWindowsUpdateCommand.CommandId | Select -ExpandProperty
 CommandPlugins
```

关闭 Windows 自动更新

以下命令可降低 Windows 更新通知级别，使系统检查更新但不自动更新托管式节点。

```
$configureWindowsUpdateCommand = Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-ConfigureWindowsUpdate" `
    -Parameters @{'updateLevel'='NeverCheckForUpdates'}
```

查看用于关闭 Windows 自动更新的命令状态

以下命令使用 CommandId 获取用于关闭 Windows 自动更新的命令执行的状态。

```
Get-SSMCommandInvocation `
    -Details $true `
    -CommandId $configureWindowsUpdateCommand.CommandId | Select -ExpandProperty
 CommandPlugins
```

使用 Run Command 管理 Windows 更新

使用 Run Command 和 AWS-InstallWindowsUpdates 文档，您可以管理 Windows Server 托管式节点的更新。此命令在托管式节点上扫描或安装缺少的更新，并且可以选择在安装后重启。还可以为在您的环境中安装的更新指定相应的分类和严重性级别。

> **Note**
>
> 有关在使用 Run Command 调用脚本时重启托管式节点的信息，请参阅 运行命令时处理重启问题。

以下示例说明了如何执行指定的 Windows Update 管理任务。

搜索所有缺少的 Windows 更新

```
Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-InstallWindowsUpdates" `
    -Parameters @{'Action'='Scan'}
```

安装特定的 Windows 更新

```
Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-InstallWindowsUpdates" `
    -Parameters @{'Action'='Install';'IncludeKbs'='kb-ID-1,kb-ID-2,kb-
ID-3';'AllowReboot'='True'}
```

安装缺少的重要 Windows 更新

```
Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-InstallWindowsUpdates" `
    -Parameters @{'Action'='Install';'SeverityLevels'='Important';'AllowReboot'='True'}
```

安装缺少的 Windows 更新 (带特定排除内容)

```
Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-InstallWindowsUpdates" `
    -Parameters @{'Action'='Install';'ExcludeKbs'='kb-ID-1,kb-
ID-2';'AllowReboot'='True'}
```

对 Systems Manager Run Command 进行故障排除

Run Command（AWS Systems Manager 中的一项工具）在每次命令执行时提供状态详细信息。有关命令状态的更多信息，请参阅 了解命令状态。您也可以使用本主题中的信息来帮助排查 Run Command 问题。

主题

- 我的部分托管式节点缺失
- 我的脚本中的一个步骤失败，但总体状态为 “succeeded”。
- SSM Agent 未正常运行

我的部分托管式节点缺失

在 Run a command（运行命令）页面上，选择要运行的 SSM 文档并在 Targets（目标）部分中选择 Manually selecting instances（手动选择实例）后，将显示托管式节点的列表，您可以从中选择要在其上运行命令的节点。

如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

创建、激活、重新引导或重新启动托管式节点、在节点上安装 Run Command 或将 AWS Identity and Access Management (IAM) 实例配置文件附加到节点后，该托管式节点可能需要几分钟时间才会添加到列表中。

我的脚本中的一个步骤失败，但总体状态为 “succeeded”。

您可以使用 Run Command 定义脚本如何处理退出代码。默认情况下，脚本中运行的最后一个命令的退出代码将报告为整个脚本的退出代码。但是，如果在最后一个命令失败之前有任何命令，则可以包含条件语句来退出脚本。有关信息以及示例，请参阅 在命令中指定退出代码。

SSM Agent 未正常运行

如果在使用 Run Command 运行命令时遇到问题，可能是因为 SSM Agent 存在问题。有关调查 SSM Agent 的相关问题的信息，请参阅 排查 SSM Agent 问题。

AWS Systems Manager Session Manager

Session Manager是一项完全托管的 AWS Systems Manager 工具。借助 Session Manager，您可以管理 Amazon Elastic Compute Cloud（Amazon EC2）实例、边缘设备、本地服务器和虚拟机（VM）。您可以使用基于浏览器的一键式交互 Shell 或 AWS Command Line Interface (AWS CLI)。Session Manager 提供安全的节点管理，而无需打开入站端口、维护堡垒主机或管理 SSH 密钥。Session Manager 还可以帮助您遵守需要托管式节点受控访问权限的公司策略、严格的安全实践以及包含节点访问详细信息的日志，同时能够让终端用户轻松地一键式跨平台访问您的托管式节点。要开始使用 Session Manager，请打开 Systems Manager 控制台。在导航窗格中，请选择 Session Manager。

我的组织如何从 Session Manager 获益？

Session Manager 具备下列优势：

- 使用 IAM policy 集中控制对托管式节点的访问

  管理员可以集中地授予和撤销对托管式节点的访问。仅使用 AWS Identity and Access Management (IAM) 策略，您就可以控制贵企业中的哪些个人用户或组能够使用 Session Manager 以及他们能够访问的托管式节点。

- 无需打开入站端口，也无需管理堡垒主机或 SSH 密钥

  在托管式节点上保持打开入站 SSH 端口和远程 PowerShell 端口，会极大增加实体在托管式节点上运行未经授权或恶意命令的风险。Session Manager 让您能够关闭这些入站端口，并且无需管理 SSH 密钥和证书以及堡垒主机和跳转盒，从而帮助您提高安全状况。

- 从控制台和 CLI 一键访问托管式节点

  通过使用 AWS Systems Manager 控制台或 Amazon EC2 控制台，只需单击一下即可启动会话。通过使用 AWS CLI，您还可以启动一个会话以运行单个命令或一系列命令。由于对托管式节点的权限是通过 IAM policy 而非 SSH 密钥或其他机制提供，连接时间得到大幅缩短。

- 连接到混合和多云环境的 Amazon EC2 实例和非 EC2 托管式节点

  连接到混合和多云环境中的 Amazon Elastic Compute Cloud（Amazon EC2）实例和非 EC2 节点。

  要使用 Session Manager 连接到非 EC2 节点，必须首先激活高级实例套餐。使用高级实例套餐需支付费用。但是，使用 Session Manager 连接到 EC2 实例不需要额外收费。有关信息，请参阅配置实例套餐。

- 端口转发

  将托管式节点中的任意端口重新导向到客户端上的本地端口。然后，连接到本地端口，并访问在节点内运行的服务器应用程序。

- 对 Windows、Linux 和 macOS 的跨平台支持

  Session Manager 通过单个工具提供对 Windows、Linux 和 macOS 的支持。例如，您不需要为 Linux 和 macOS 托管式节点使用 SSH 客户端，也不需要为 Windows Server 托管式节点使用 RDP 连接。

- 记录会话活动

  为满足企业的运营或安全要求，您可能需要提供对托管式节点的连接以及在其上运行的命令的记录。您还可以在组织中的用户开始或结束会话活动时收到通知。

  通过与以下 AWS 服务集成来提供日志记录功能：

  - AWS CloudTrail – AWS CloudTrail 捕获有关在 AWS 账户中进行的 Session Manager API 调用的信息，并将其写入存储在指定 Amazon Simple Storage Service (Amazon S3) 存储桶中的日志文件。账户的所有 CloudTrail 日志都存储在一个存储桶中。有关更多信息，请参阅 使用 AWS CloudTrail 记录 AWS Systems Manager API 调用。
  - Amazon Simple Storage Service – 您可以选择将会话日志数据存储在您选择的 Amazon S3 存储桶中，以便用于调试和故障排除。在将日志数据发送到 Amazon S3 存储桶时，可以使用或不使

用您的 AWS KMS key 进行加密。有关更多信息，请参阅 使用 Amazon S3 记录会话数据（控制台）。

- Amazon CloudWatch Logs – 借助 CloudWatch Logs，您可以监控、存储和访问各种 AWS 服务的日志文件。您可以将会话日志数据发送到 CloudWatch Logs 日志组，以便用于调试和故障排除。在将日志数据发送到日志组时，可以使用您的 KMS 密钥进行 AWS KMS 加密，也可以不加密。有关更多信息，请参阅 使用 Amazon CloudWatch Logs 记录会话数据（控制台）。
- Amazon EventBridge 和 Amazon Simple Notification Service – 使用 EventBridge，您可以设置规则来检测指定的 AWS 资源何时发生更改。您可以创建规则来检测组织中的用户何时启动或停止会话，然后通过 Amazon SNS 接收有关事件的通知（例如，文本消息或电子邮件）。您还可以配置 CloudWatch 事件来启动其他响应。有关更多信息，请参阅 使用 Amazon EventBridge 监控会话活动（控制台）。

> **Note**
>
> 日志记录不可用于通过端口转发或 SSH 连接的 Session Manager 会话。这是因为 SSH 会加密所有会话数据，而 Session Manager 仅充当 SSH 连接的隧道。

谁应该使用 Session Manager？

- 任何希望改善其安全状况、通过集中控制对托管式节点的访问来减少运营开销以及希望减少入站节点访问的 AWS 客户。
- 希望监控并跟踪托管式节点访问和活动、关闭托管式节点上的入站端口或允许连接到没有公有 IP 地址的托管式节点的信息安全专家。
- 希望从单一位置授予及撤销访问权限并为用户提供一个同时适用于 Linux、macOS 和 Windows Server 托管式节点的解决方案的管理员。
- 希望只需单击一下即可从浏览器或 AWS CLI 连接到托管式节点而不必提供 SSH 密钥的用户。

Session Manager 的主要功能是什么？

- 对 Windows Server、Linux 和 macOS 托管式节点的支持

  Session Manager 可让您与 Amazon Elastic Compute Cloud（EC2）实例、边缘设备、本地服务器和虚拟机（VM）之间建立安全连接。有关支持的操作系统类型的列表，请参阅 设置 Session Manager。

> **Note**
>
> 仅为高级实例套餐提供对本地计算机的 Session Manager 支持。有关信息，请参阅打开高级实例套餐。

- 支持通过控制台、CLI 和开发工具包等方式访问 Session Manager 功能

  您可以通过以下方式使用Session Manager：

  AWS Systems Manager 控制台使管理员和终端用户能够访问所有 Session Manager 功能。您可以通过使用 Systems Manager 控制台来执行与会话相关的任何任务。

  利用 Amazon EC2 控制台，终端用户能够连接到其已获得会话权限的 EC2 实例。

  AWS CLI 包括对适用于最终用户的 Session Manager 功能的访问。您可以使用 AWS CLI 启动会话、查看会话列表和永久结束会话。

  > **Note**
  >
  > 要使用 AWS CLI 运行会话命令，您必须使用 1.16.12 版本（或更高版本）的 CLI，并且必须已在本地计算机上安装 Session Manager 插件。有关信息，请参阅为 AWS CLI 安装 Session Manager 插件。要在 GitHub 上查看插件，请参阅 session-manager-plugin。

- IAM 访问控制

  通过使用 IAM policy，您可以控制企业中的哪些成员能够启动与托管式节点的会话以及他们能够访问哪些节点。您还可以提供对托管式节点的临时访问。例如，您可能需要仅为执勤工程师（或一组执勤工程师）在当班期间提供对生产服务器的访问权限。

- 日志记录支持

  Session Manager通过与许多其他 AWS 服务集成，为您提供在 AWS 账户中记录会话历史记录的选项。有关更多信息，请参阅记录会话活动和启用和禁用会话日志记录。

- 可配置的 Shell 配置文件

  Session Manager 为您提供配置会话中的首选项的选项。您可以使用这些可自定义的配置文件定义首选项，例如 Shell 首选项、环境变量、工作目录以及在启动会话时运行多个命令。

- 客户密钥数据加密支持

您可以配置 Session Manager 以加密发送到 Amazon Simple Storage Service (Amazon S3) 存储桶或流式传输到 CloudWatch Logs 日志组的会话数据日志。您还可以配置 Session Manager 来进一步为会话期间在客户端计算机和托管式节点之间传输的数据加密。有关信息，请参阅 启用和禁用会话日志记录 和配置会话首选项。

- 为没有公有 IP 地址的托管式节点提供 AWS PrivateLink 支持

  您还可以使用 AWS PrivateLink 为 Systems Manager 设置 VPC 终端节点，以进一步保护您的会话。AWS PrivateLink 将托管式节点、Systems Manager 和 Amazon EC2 之间的所有网络流量限制在 Amazon 网络以内。有关更多信息，请参阅使用适用于 Systems Manager 的 VPC 端点提高 EC2 实例的安全性。

- 隧道

  在会话中，使用会话类型 AWS Systems Manager (SSM) 文档在客户端计算机上的本地端口与托管式节点上的远程端口之间通过隧道传输流量，例如 http 或自定义协议。

- 交互式命令

  创建会话类型的 SSM 文档，使用会话以交互方式运行单个命令，为您提供一种管理用户可以在托管式节点上执行的操作的方法。

什么是会话？

会话是使用 Session Manager 与托管式节点之间建立的连接。会话基于客户端（您）和远程托管式节点之间的安全双向通信通道，可流式传输命令的输入和输出。客户端和托管式节点之间的流量使用 TLS 1.2 进行加密，创建连接的请求使用 Sigv4 进行签名。这种双向通信允许交互式 Bash 和 PowerShell 访问托管式节点。您还可以使用 AWS Key Management Service (AWS KMS) 密钥在默认 TLS 加密之外进一步加密数据。

举例来说，假设 John 是 IT 部门的执勤工程师。他收到一个问题通知，要求他远程连接到某个托管式节点（例如需要处理的故障或用于更改节点上的简单配置选项的指令）。John 使用 AWS Systems Manager 控制台、Amazon EC2 控制台或 AWS CLI 启动一个将其连接到该托管式节点的会话，在完成任务所需的节点上运行命令，然后结束该会话。

当 John 发送第一个命令启动会话时，Session Manager 服务对其 ID 进行身份验证，验证 IAM policy 授予 John 的权限，检查配置设置（例如验证允许的会话限制），并向 SSM Agent发送消息以打开双向连接。建立连接并且 John 键入下一个命令后，SSM Agent的命令输出将上传到此通信通道并发送回 John 的本地计算机。

主题

- 设置 Session Manager
- 使用 Session Manager
- 记录会话活动
- 启用和禁用会话日志记录
- 会话文档架构
- 排查 Session Manager 问题

设置 Session Manager

在使用 AWS Systems Manager Session Manager 连接到账户中的托管式节点之前，请完成以下主题中的步骤。

主题

- 步骤 1：满足 Session Manager 先决条件
- 步骤 2：验证或添加 Session Manager 的实例权限
- 步骤 3：控制会话对托管式节点的访问
- 步骤 4：配置会话首选项
- 步骤 5：（可选）限制对会话中命令的访问
- 步骤 6：（可选）使用 AWS PrivateLink 为 Session Manager 设置 VPC 端点
- 步骤 7：（可选）开启或关闭 ssm-user 账户管理权限
- 步骤 8：（可选）通过 Session Manager 允许和控制 SSH 连接的权限

步骤 1：满足 Session Manager 先决条件

在使用 Session Manager 之前，请确保环境满足以下要求。

Session Manager 先决条件

要求	描述
支持的操作系统	除了混合和多云环境中使用高级实例套餐的非 EC2 计算机之外，Session Manager 还支持连接到 Amazon Elastic Compute Cloud（Amazon EC2）实例。

要求	描述
	Session Manager 支持以下操作系统版本： **Note** Session Manager 支持混合和多云环境中使用高级实例套餐的 EC2 实例、边缘设备、本地服务器和虚拟机（VM）。有关高级实例的更多信息，请参阅 配置实例套餐。 Linux 和 macOS Session Manager 支持 AWS Systems Manager 所支持的所有 Linux 和 macOS 版本。有关信息，请参阅支持的操作系统和计算机类型。 Windows Session Manager 支持 Windows Server 2012 至 Windows Server 2022。 **Note** 不支持 Microsoft Windows Server 2016 Nano。

要求	描述
SSM Agent	至少应在要通过会话连接的托管式节点上安装 2.3.68.0 版本或更高版本的 AWS Systems Manager SSM Agent。 要通过该选项以使用在 AWS Key Management Service (AWS KMS) 中创建的密钥加密会话数据，必须在托管式节点上安装 2.3.539.0 版本或更高版本的 SSM Agent。 要在会话中使用 Shell 配置文件，必须在托管式节点上安装 SSM Agent 3.0.161.0 版本或更高版本。 要开启 Session Manager 端口转发或 SSH 会话，必须在托管式节点上安装 SSM Agent 3.0.222.0 版本或更高版本。 要使用 Amazon CloudWatch Logs 流式传输会话数据，必须在托管式节点上安装 SSM Agent 3.0.284.0 版本或更高版本。 有关如何确定实例上运行的版本号的信息，请参阅 正在检查 SSM Agent 版本号。有关手动安装或自动更新 SSM Agent 的信息，请参阅 使用 SSM Agent。 关于 ssm-user 账户 从 SSM Agent 的版本 2.3.50.0 开始，代理在托管式节点上创建名为 `ssm-user` 的用户账户，此账户具有根权限或管理员权限。（在 2.3.612.0 之前的版本上，此账户在 SSM Agent 启动或重新启动时创建。在版本 2.3.612.0 及更高版本上，`ssm-user` 在托管式节点上首次启动会话时创建。）系统使用此用户账户的管理凭证启动会话。有关限制此账户的管理控制权限的

要求	描述
	信息，请参阅关闭或打开 ssm-user 账户管理权限。 Windows Server 域控制器上的 ssm-user 从 SSM Agent 版本 2.3.612.0 开始，ssm-user 账户不会在用作 Windows Server 域控制器的托管式节点上自动创建。要在用作域控制器的 Windows Server 计算机上使用 Session Manager，您必须手动创建 ssm-user 账户（如果此账户尚不存在），并为相关用户分配域管理员权限。在 Windows Server 上，每次会话启动时，SSM Agent 都会为 ssm-user 账户设置新密码，因此您无需在创建账户时指定密码。
连接到端点	您要连接到的托管式节点必须还允许到以下端点的 HTTPS（端口 443）出站流量： • ec2messages.*region*.amazonaws.com • ssm.*region*.amazonaws.com • ssmmessages.*region*.amazonaws.com 有关更多信息，请参阅以下主题： • 参考：ec2messages、ssmmessages 和其他 API 操作 • AWS re:Post 知识中心中的如何创建 VPC 端点，才能在不连接互联网的情况下使用 Systems Manager 管理私有 EC2 实例？ 或者，您可以使用接口端点连接到所需端点。有关更多信息，请参阅 步骤 6：（可选）使用 AWS PrivateLink 为 Session Manager 设置 VPC 端点。

要求	描述
AWS CLI	（可选）如果使用 AWS Command Line Interface (AWS CLI)（而不是使用 AWS Systems Manager 控制台或 Amazon EC2 控制台）启动会话，则必须在本地计算机上安装 1.16.12 版本或更高版本的 CLI。 您可调用 `aws --version` 来查看版本。 如果您需要安装或升级 CLI，请参阅 AWS Command Line Interface 用户指南中的安装 AWS Command Line Interface。 **Important** 如果有新工具添加至 Systems Manager 或者对现有工具进行了更新，则将发布 SSM Agent 的更新版本。无法使用代理的最新版本可能会阻止托管式节点使用 Systems Manager 的各项工具和功能。因此，我们建议您自动完成确保机器上的 SSM Agent 为最新的过程。有关更多信息，请参阅 自动更新到 SSM Agent。要获得有关 SSM Agent 更新的通知，请在 GitHub 上订阅 SSM Agent 发布说明页面。 此外，要使用 CLI 通过 Session Manager 管理节点，必须先在本地计算机上安装 Session Manager 插件。有关信息，请参阅为 AWS CLI 安装 Session Manager 插件。

要求	描述
启用高级实例套餐（混合和多云环境）	若要使用 Session Manager 连接到非 EC2 计算机，必须在您可以在其中创建混合激活以将非 EC2 计算机注册为托管式节点的 AWS 账户 和 AWS 区域 中启用高级实例套餐。使用高级实例套餐需支付费用。有关高级实例套餐的更多信息，请参阅 配置实例套餐。
验证 IAM 服务角色权限（混合和多云环境）	混合激活节点使用在混合激活中指定的 AWS Identity and Access Management（IAM）服务角色，与 Systems Manager API 操作进行通信。此服务角色必须包含使用 Session Manager 连接到混合和多云计算机所需的权限。如果您的服务角色包含 AWS 托管式策略 `AmazonSSMManagedInstanceCore` ，则已提供 Session Manager 所需的权限。 如果您发现服务角色不包含所需权限，则必须取消注册托管式实例，然后利用使用具有所需权限的 IAM 服务角色的新混合激活进行注册。有关取消注册托管式实例的更多信息，请参阅 在混合和多云环境中取消注册托管式节点。有关创建具有 Session Manager 权限的 IAM policy 的更多信息，请参阅步骤 2：验证或添加 Session Manager 的实例权限色。

步骤 2：验证或添加 Session Manager 的实例权限

默认情况下，AWS Systems Manager 没有在您的实例上执行操作的权限。您可以使用 AWS Identity and Access Management（IAM）角色在账户级别提供实例权限，也可以使用实例配置文件在实例级别提供实例权限。如果您的应用场景允许，我们建议使用“默认主机管理配置”在账户级别授予访问权限。如果您已经使用 `AmazonSSMManagedEC2InstanceDefaultPolicy` 策略为账户设置了“默认主机管理配置”，则可以继续执行下一步。有关“默认主机管理配置”的更多信息，请参阅使用默认主机管理配置自动管理 EC2 实例。

或者，您可以使用实例配置文件为实例提供所需权限。实例配置文件会将 IAM 角色传递给 Amazon EC2 实例。您可以将 IAM 实例配置文件附加到 Amazon EC2 实例（在启动该实例时）或之前启动的实例。有关更多信息，请参阅使用实例配置文件。

对于本地部署服务器或虚拟机 (VM)，权限由与混合激活关联的 IAM 服务角色提供，该激活用于将您的本地部署服务器和虚拟机注册到 Systems Manager。本地服务器和虚拟机不使用实例配置文件。

如果已使用其他 Systems Manager 工具（例如 Run Command 或 Parameter Store），则可能已将具有Session Manager所需的基本权限的实例配置文件附加到 Amazon EC2 实例。如果包含 AWS 托管式策略 `AmazonSSMManagedInstanceCore` 的实例配置文件已附上您的实例，则已提供 Session Manager 所需的权限。如果混合激活中使用的 IAM 服务角色包含 `AmazonSSMManagedInstanceCore` 托管式策略，则也已提供所需权限。

但在某些情况下，您可能需要修改附加到实例配置文件的权限。例如，您希望提供一组更少的实例权限，您已为实例配置文件创建自定义策略，或者您希望使用 Amazon Simple Storage Service (Amazon S3) 加密或 AWS Key Management Service (AWS KMS) 加密选项以保护会话数据。对于这些情况，请执行以下操作之一，以允许在您的实例上执行 Session Manager 操作：

- 在自定义 IAM 角色中嵌入 Session Manager 操作权限

  要将 Session Manager 操作的权限添加到不依赖于 AWS 提供的原定设置策略 `AmazonSSMManagedInstanceCore` 的现有 IAM 角色，请按照 向现有 IAM 角色添加 Session Manager 权限 中的步骤操作。
- 创建仅具有 Session Manager 权限的自定义 IAM 角色

  要创建仅具有 Session Manager 操作权限的 IAM 角色，请按照 为 Session Manager 创建自定义 IAM 角色 中的步骤操作。
- 创建和使用具有所有 Systems Manager 操作权限的新 IAM 角色

  要为 Systems Manager 托管式实例创建使用 AWS 提供的默认策略授予所有 Systems Manager 权限的 IAM 角色，请按照配置 Systems Manager 所需的实例权限中的步骤进行操作。

主题

- 向现有 IAM 角色添加 Session Manager 权限
- 为 Session Manager 创建自定义 IAM 角色

向现有 IAM 角色添加 Session Manager 权限

使用以下过程为现有 AWS Identity and Access Management（IAM）角色添加 Session Manager 权限。通过向现有角色添加权限，您可以增强计算环境的安全性，而不必使用 AWSAmazonSSMManagedInstanceCore 策略来获得实例权限。

> **Note**
>
> 请注意以下信息：
>
> - 此过程假设现有角色已包含您希望允许访问的操作的其他 Systems Manager ssm 权限。要使用 Session Manager，只有此策略是不够的。
> - 以下策略示例包括一项 s3:GetEncryptionConfiguration 操作。如果您在 Session Manager 日志首选项中选择了强制 S3 日志加密选项，则需要执行此操作。

向现有角色添加 Session Manager 权限（控制台）

1. 登录 AWS Management Console，然后通过以下网址打开 IAM 控制台：https://console.aws.amazon.com/iam/。
2. 在导航窗格中，选择角色。
3. 选择要向其添加权限的角色的名称。
4. 选择 Permissions（权限）选项卡。
5. 选择添加权限，然后选择创建内联策略。
6. 选择 JSON 选项卡。
7. 将默认策略内容替换为以下内容。将 *key-name* 替换为您要使用的 AWS Key Management Service 密钥（AWS KMS key）的 Amazon 资源名称（ARN）。

   JSON

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

```
                "ssmmessages:CreateDataChannel",
                "ssmmessages:OpenControlChannel",
                "ssmmessages:OpenDataChannel"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetEncryptionConfiguration"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/key-name"
        }
    ]
}
```

有关使用 KMS 密钥加密会话数据的信息，请参阅 启用会话数据的 KMS 密钥加密（控制台）。

如果您不为会话数据使用 AWS KMS 加密，可以从策略中删除以下内容。

```
,
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "key-name"
        }
```

8. 选择下一步：标签。
9. （可选）通过选择 Add tag（添加标签）并输入策略的首选标签来添加标签。
10. 选择下一步：审核。
11. 在查看策略页面上，对于名称，输入内联策略的名称，例如 **SessionManagerPermissions**。
12. （可选）对于 Description (描述)，输入策略的描述。

选择 Create policy (创建策略)。

有关 ssmmessages 操作的信息，请参阅 参考：ec2messages、ssmmessages 和其他 API 操作。

为 Session Manager 创建自定义 IAM 角色

您可以创建一个 AWS Identity and Access Management（IAM）角色来授予 Session Manager 对您的 Amazon EC2 托管实例执行操作的权限。您还可以在其中包含一个策略，以授予将会话日志发送到 Amazon Simple Storage Service（Amazon S3）和 Amazon CloudWatch Logs 所需的权限。

创建 IAM 角色后，如需关于如何将角色附加到实例的信息，请参阅 AWS re:Post 网站上的附加或替换实例配置文件。有关 IAM 实例配置文件和角色的更多信息，请参阅《IAM 用户指南》中的使用实例配置文件和《Amazon Elastic Compute Cloud 用户指南（适用于 Linux 实例）》中的适用于 Amazon EC2 的 IAM 角色。有关为本地计算机创建 IAM 服务角色的更多信息，请参阅在混合和多云环境中创建 Systems Manager 所需的 IAM 服务角色。

主题

- 创建具有最小 Session Manager 权限的 IAM 角色（控制台）
- 创建具有 Session Manager、Amazon S3 和 CloudWatch Logs 权限的 IAM 角色（控制台）

创建具有最小 Session Manager 权限的 IAM 角色（控制台）

使用以下过程可创建一个自定义 IAM 角色，其策略仅提供在实例上执行 Session Manager 操作的权限。

创建具有最小 Session Manager 权限的实例配置文件（控制台）

1. 登录 AWS Management Console，然后通过以下网址打开 IAM 控制台：https://console.aws.amazon.com/iam/。
2. 在导航窗格中选择策略，然后选择创建策略。（如果显示 Get Started (入门) 按钮，请选择此按钮，然后选择 Create Policy (创建策略)。）
3. 选择 JSON 选项卡。
4. 将原定设置内容替换为以下策略。要使用 AWS Key Management Service（AWS KMS）加密会话数据，请将 *key-name* 替换为您要使用的 AWS KMS key 的 Amazon 资源名称（ARN）。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:UpdateInstanceInformation",
                "ssmmessages:CreateControlChannel",
                "ssmmessages:CreateDataChannel",
                "ssmmessages:OpenControlChannel",
                "ssmmessages:OpenDataChannel"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/key-name"
        }
    ]
}
```

有关使用 KMS 密钥加密会话数据的信息，请参阅 启用会话数据的 KMS 密钥加密（控制台）。

如果您不为会话数据使用 AWS KMS 加密，可以从策略中删除以下内容。

```
,
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "key-name"
        }
```

5. 选择下一步：标签。

6. （可选）通过选择 Add tag（添加标签）并输入策略的首选标签来添加标签。
7. 选择下一步：审核。
8. 在查看策略页面上，对于名称，输入内联策略的名称，例如 **`SessionManagerPermissions`**。
9. （可选）对于 Description (描述)，输入策略的描述。
10. 选择 Create policy (创建策略)。
11. 在导航窗格中，选择 Roles（角色），然后选择 Create role（创建角色）。
12. 在 Create role（创建角色）页面上，选择 AWS service（ 服务），对于 Use case（使用案例），选择 EC2。
13. 选择下一步。
14. 在 Add permissions（添加权限）页面上，选中刚刚创建的策略名称左侧的复选框，例如 **`SessionManagerPermissions`**。
15. 选择下一步。
16. 在 Name, review, and create（名称、审核和创建）页面中，对于 Role name（角色名称），输入 IAM 角色的名称，例如 **`MySessionManagerRole`**。
17. （可选）对于角色描述，输入实例配置文件的描述。
18. （可选）通过选择 Add tag（添加标签）并输入角色的首选标签来添加标签。

    选择创建角色。

有关 `ssmmessages` 操作的信息，请参阅 参考：ec2messages、ssmmessages 和其他 API 操作。

创建具有 Session Manager、Amazon S3 和 CloudWatch Logs 权限的 IAM 角色（控制台）

使用以下过程创建一个自定义 IAM 角色，其策略提供在实例上执行 Session Manager 操作的权限。此策略还提供了将会话日志存储在 Amazon Simple Storage Service (Amazon S3) 存储桶和 Amazon CloudWatch Logs 日志组中所需的权限。

> **Important**
>
> 要将会话日志输出到其他 AWS 账户 拥有的 Amazon S3 存储桶，您必须将 `s3:PutObjectAcl` 权限添加到 IAM 角色策略。此外，您必须确保存储桶策略授予对拥有账户使用的 IAM 角色的跨账户存取权限，来授予 Systems Manager 托管实例的权限。如果存储桶使用密钥管理服务（KMS）加密，则存储桶的 KMS 策略还必须授予此跨账户存取权限。有关在 Amazon S3 中配置跨账户存储桶权限的信息，请参阅《Amazon Simple Storage Service

用户指南》中的授予跨账户存储桶权限。如果跨账户权限未添加，则拥有 Amazon S3 存储桶的账户将无法访问会话输出日志。

有关指定存储会话日志的首选项的信息，请参阅 启用和禁用会话日志记录。

要创建具有 Session Manager、Amazon S3 和 CloudWatch Logs 权限的 IAM 角色（控制台），请执行以下步骤

1. 登录 AWS Management Console，然后通过以下网址打开 IAM 控制台：https://console.aws.amazon.com/iam/。
2. 在导航窗格中选择策略，然后选择创建策略。（如果显示 Get Started (入门) 按钮，请选择此按钮，然后选择 Create Policy (创建策略)。）
3. 选择 JSON 选项卡。
4. 将原定设置内容替换为以下策略。将每个#######替换为您自己的信息。

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssmmessages:CreateControlChannel",
                "ssmmessages:CreateDataChannel",
                "ssmmessages:OpenControlChannel",
                "ssmmessages:OpenDataChannel",
                "ssm:UpdateInstanceInformation"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogGroups",
                "logs:DescribeLogStreams"
            ],
```

```
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/s3-prefix/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetEncryptionConfiguration"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/key-name"
        },
        {
            "Effect": "Allow",
            "Action": "kms:GenerateDataKey",
            "Resource": "*"
        }
    ]
}
```

5. 选择下一步：标签。
6. （可选）通过选择 Add tag（添加标签）并输入策略的首选标签来添加标签。
7. 选择下一步：审核。
8. 在查看策略页面上，对于名称，输入内联策略的名称，例如 **SessionManagerPermissions**。
9. （可选）对于 Description (描述)，输入策略的描述。
10. 选择 Create policy (创建策略)。
11. 在导航窗格中，选择 Roles（角色），然后选择 Create role（创建角色）。
12. 在 Create role（创建角色）页面上，选择 AWS service（ 服务），对于 Use case（使用案例），选择 EC2。

13. 选择下一步。
14. 在 Add permissions（添加权限）页面上，选中刚刚创建的策略名称左侧的复选框，例如 **SessionManagerPermissions**。
15. 选择下一步。
16. 在 Name, review, and create（名称、审核和创建）页面中，对于 Role name（角色名称），输入 IAM 角色的名称，例如 **MySessionManagerRole**。
17. （可选）对于 Role description(角色描述)，输入角色的描述。
18. （可选）通过选择 Add tag（添加标签）并输入角色的首选标签来添加标签。
19. 选择创建角色。

步骤 3：控制会话对托管式节点的访问

您可以使用 AWS Identity and Access Management（IAM）policy 授予或撤消对托管式节点的 Session Manager 访问权限。您可以创建策略并将其附加到 IAM 用户或群组，以指定此用户或群组可以连接到哪些托管式节点。您还可以指定用户或群组可以在这些托管式节点上执行的 Session Manager API 操作。

为了帮助您开始使用 Session Manager 的 IAM 权限策略，我们为最终用户和管理员用户创建了示例策略。您只需稍作修改即可使用这些策略。或者，将这些策略用作创建自定义 IAM policy 的指南。有关更多信息，请参阅 Session Manager 的 IAM policy 示例。有关如何创建 IAM policy 并将策略附加到用户或组的信息，请参阅《IAM 用户指南》中的创建 IAM policy 及添加和移除 IAM policy。

关于会话 ID ARN 格式

为 Session Manager 访问权限创建 IAM policy 时，您可以指定会话 ID 作为 Amazon 资源名称（ARN）的一部分。会话 ID 包括作为变量的用户名。为了帮助说明这一点，以下是 Session Manager ARN 的格式和示例：

```
arn:aws:ssm:region-id:account-id:session/session-id
```

例如：

```
arn:aws:ssm:us-east-2:123456789012:session/JohnDoe-1a2b3c4d5eEXAMPLE
```

有关在 IAM policy 中使用变量的更多信息，请参阅 IAM policy 元素：变量。

主题

- 通过在 IAM policy 中指定默认会话文档来启动默认 Shell 会话
- 通过在 IAM policy 中指定会话文档来启动与文档的会话
- Session Manager 的 IAM policy 示例
- Session Manager 的其他示例 IAM policy

通过在 IAM policy 中指定默认会话文档来启动默认 Shell 会话

当您在 Systems Manager 控制台中为 AWS 账户 配置 Session Manager 或更改会话首选项时，系统会创建一个名为 `SSM-SessionManagerRunShell` 的 SSM 会话文档。这是默认的会话文档。Session Manager 使用此文档存储您的会话首选项，其中包括以下信息：

- 您要保存会话数据的位置，例如 Amazon Simple Storage Service（Amazon S3）存储桶或 Amazon CloudWatch Logs 日志组。
- 用于加密会话数据的 AWS Key Management Service（AWS KMS）密钥 ID。
- 会话是否允许“运行身份”支持。

以下是 `SSM-SessionManagerRunShell` 会话首选项文档中包含的信息的示例。

```
{
  "schemaVersion": "1.0",
  "description": "Document to hold regional settings for Session Manager",
  "sessionType": "Standard_Stream",
  "inputs": {
    "s3BucketName": "amzn-s3-demo-bucket",
    "s3KeyPrefix": "MyS3Prefix",
    "s3EncryptionEnabled": true,
    "cloudWatchLogGroupName": "MyCWLogGroup",
    "cloudWatchEncryptionEnabled": false,
    "kmsKeyId": "1a2b3c4d",
    "runAsEnabled": true,
    "runAsDefaultUser": "RunAsUser"
  }
}
```

默认情况下，当用户 AWS Management Console 从启动会话时，Session Manager 使用默认会话文档。这适用于 Systems Manager 控制台中的 Fleet Manager 或 Session Manager，或 Amazon EC2 控制台中的 EC2 Connect。当用户使用如下示例所示的 AWS CLI 命令启动会话时，Session Manager 也会使用默认会话文档：

```
aws ssm start-session \
    --target i-02573cafcfEXAMPLE
```

要开始默认 Shell 会话，您必需在 IAM 策略中指定默认会话文档，如以下示例所示。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "EnableSSMSession",
      "Effect": "Allow",
      "Action": [
        "ssm:StartSession"
      ],
      "Resource": [
        "arn:aws:ec2:us-east-1:111122223333:instance/instance-id",
        "arn:aws:ssm:us-east-1:111122223333:document/SSM-SessionManagerRunShell"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "ssmmessages:OpenDataChannel"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

通过在 IAM policy 中指定会话文档来启动与文档的会话

如果使用默认会话文档的 start-session AWS CLI 命令，则可以省略文档名称。系统会自动调用 SSM-SessionManagerRunShell 会话文档。

在所有其他情况下，您必须为 document-name 参数指定一个值。当用户在命令中指定会话文档的名称时，系统会检查其 IAM policy，以验证他们是否有权访问该文档。如果他们没有权限，连接请求就会失败。以下示例包括 AWS-StartPortForwardingSession 会话文档中的 document-name 参数。

```
aws ssm start-session \
    --target i-02573cafcfEXAMPLE \
    --document-name AWS-StartPortForwardingSession \
    --parameters '{"portNumber":["80"], "localPortNumber":["56789"]}'
```

有关在 IAM 策略中指定 Session Manager 会话文档的示例，请参阅 Session Manager 的快速入门最终用户策略。

> **Note**
>
> 要使用 SSH 启动会话，必须在目标托管式节点和用户的本地计算机上完成配置步骤。有关信息，请参阅（可选）通过 Session Manager 允许和控制 SSH 连接的权限。

Session Manager 的 IAM policy 示例

使用此部分中的示例可帮助您创建 AWS Identity and Access Management (IAM) policy，以提供访问 Session Manager 所需的最常用权限。

> **Note**
>
> 您还可以使用 AWS KMS key 策略，控制授予哪些 IAM 实体（用户或角色）和 AWS 账户 访问您的 KMS 密钥的权限。有关信息，请参阅《AWS Key Management Service 开发人员指南》中管理 AWS KMS 资源的访问权限概览和在 AWS KMS 中使用密钥策略。

主题

- Session Manager 的快速入门最终用户策略
- （快速入门）Session Manager 管理员策略

Session Manager 的快速入门最终用户策略

使用以下示例为 Session Manager 创建 IAM 终端用户策略。

您可以创建一个策略，允许用户仅从 Session Manager 控制台和 AWS Command Line Interface (AWS CLI)、仅从 Amazon Elastic Compute Cloud (Amazon EC2) 控制台或从这三项中启动会话。

这些策略为终端用户提供启动连接到特定托管式节点的会话以及仅结束自己的会话的功能。有关可能需要对策略进行的自定义的示例，请参阅 Session Manager 的其他示例 IAM policy。

在以下示例策略中，将每个*#######*替换为您自己的信息。

请参阅以下部分，查看要提供的会话访问范围的示例策略。

会话管理器 and Fleet Manager

使用此示例策略使用户能够仅从 Session Manager 和 Fleet Manager 控制台启动和恢复会话。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ec2:us-
east-1:111122223333:instance/i-02573cafcfEXAMPLE",
                "arn:aws:ssm:us-east-1:111122223333:document/SSM-
SessionManagerRunShell"
            ]
        },
        {
         "Effect": "Allow",
         "Action": ["ssmmessages:OpenDataChannel"],
         "Resource": ["arn:aws:ssm:*:*:session/${aws:userid}-*"]
       },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeSessions",
                "ssm:GetConnectionStatus",
                "ssm:DescribeInstanceProperties",
                "ec2:DescribeInstances"
```

```
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:TerminateSession",
                "ssm:ResumeSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:userid}-*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey"
            ],
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/key-name"
        }
    ]
}
```

Amazon EC2

使用此示例策略使用户能够仅从 Amazon EC2 控制台启动和恢复会话。该策略不提供从 Session Manager 控制台和 AWS CLI 启动会话所需的所有权限。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession",
                "ssm:SendCommand"
            ],
            "Resource": [
                "arn:aws:ec2:us-
east-1:111122223333:instance/i-02573cafcfEXAMPLE",
```

```
                "arn:aws:ssm:us-east-1:111122223333:document/SSM-
SessionManagerRunShell"
            ]
        },
        {
         "Effect": "Allow",
         "Action": ["ssmmessages:OpenDataChannel"],
         "Resource": ["arn:aws:ssm:*:*:session/${aws:userid}-*"]
       },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetConnectionStatus",
                "ssm:DescribeInstanceInformation"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:TerminateSession",
                "ssm:ResumeSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:username}-*"
            ]
        }
    ]
}
```

AWS CLI

使用此示例策略使用户能够仅从 AWS CLI 启动和恢复会话。

JSON

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

```
                "ssm:StartSession",
                "ssm:SendCommand"
            ],
            "Resource": [
                "arn:aws:ec2:us-
east-1:111122223333:instance/i-02573cafcfEXAMPLE",
                "arn:aws:ssm:us-east-1:111122223333:document/SSM-
SessionManagerRunShell"
            ]
        },
        {
         "Effect": "Allow",
         "Action": ["ssmmessages:OpenDataChannel"],
         "Resource": ["arn:aws:ssm:*:*:session/${aws:userid}-*"]
       },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:TerminateSession",
                "ssm:ResumeSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:userid}-*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey"
            ],
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/key-name"
        }
    ]
}
```

[1] SSM-SessionManagerRunShell 是 Session Manager 创建的用于存储会话配置首选项的 SSM 文档的默认名称。您可以创建自定义会话文档，并在此策略中指定它。您还可以为使用 SSH 启动会话的用户指定 AWS 提供的文档 AWS-StartSSHSession。有关使用 SSH 支持会话所需的配置步骤的信息，请参阅（可选）通过 Session Manager 允许和控制 SSH 连接的权限。

[2] `kms:GenerateDataKey` 权限允许创建用于加密会话数据的数据加密密钥。如果您将为您的会话数据使用 AWS Key Management Service (AWS KMS) 加密，请将 *key-name* 替换为您要使用的 KMS 密钥的 Amazon Resource Name (ARN)，使用格式 `arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-12345EXAMPLE`。如果您不为会话数据使用 KMS 密钥加密，请从策略中删除以下内容。

```
{
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey"
            ],
            "Resource": "key-name"
        }
```

有关使用 AWS KMS 加密会话数据的信息，请参阅 启用会话数据的 KMS 密钥加密（控制台）。

[3] 如果用户尝试从 Amazon EC2 控制台启动会话，则需要 SendCommand 权限，但 SSM Agent 必须先更新到 Session Manager 所需的最低版本。Run Command 用于向实例发送命令，以便更新代理。

（快速入门）Session Manager 管理员策略

使用以下示例可为 Session Manager 创建 IAM 管理员策略。

这些策略为管理员提供启动与使用 `Key=Finance,Value=WebServers` 标记的托管式节点的会话的功能，创建、更新和删除首选项的权限，以及仅结束自己的会话的权限。有关可能需要对策略进行的自定义的示例，请参阅 Session Manager 的其他示例 IAM policy。

您可以创建一个策略，允许管理员仅从 Session Manager 控制台和 AWS CLI、仅从 Amazon EC2 控制台或从这三项中执行这些任务。

在以下示例策略中，将每个*#######*替换为您自己的信息。

请参阅以下部分，查看三种权限方案的示例策略。

会话管理器 and CLI

使用此示例策略使管理员能够仅从 Session Manager 控制台和 AWS CLI 执行会话相关任务。此策略不提供从 Amazon EC2 控制台执行会话相关任务所需的所有权限。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ec2:*:111122223333:instance/*"
            ],
            "Condition": {
                "StringLike": {
                    "ssm:resourceTag/Finance": [
                        "WebServers"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssmmessages:OpenDataChannel"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:userid}-*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeSessions",
                "ssm:GetConnectionStatus",
                "ssm:DescribeInstanceProperties",
                "ec2:DescribeInstances"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
```

```
                "ssm:CreateDocument",
                "ssm:UpdateDocument",
                "ssm:GetDocument",
                "ssm:StartSession"
            ],
            "Resource": "arn:aws:ssm:us-east-1:111122223333:document/SSM-
SessionManagerRunShell"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssmmessages:OpenDataChannel"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:userid}-*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:TerminateSession",
                "ssm:ResumeSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:userid}-*"
            ]
        }
    ]
}
```

Amazon EC2

使用此示例策略使管理员能够仅从 Amazon EC2 控制台执行会话相关任务。该策略不提供从 Session Manager 控制台和 AWS CLI 执行会话相关任务所需的所有权限。

JSON

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

```
            "Action": [
                "ssm:StartSession",
                "ssm:SendCommand"
            ],
            "Resource": [
                "arn:aws:ec2:region:account-id:instance/*"
            ],
            "Condition": {
                "StringLike": {
                    "ssm:resourceTag/tag-key": [
                        "tag-value"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ssm:region:account-id:document/SSM-
SessionManagerRunShell"
            ]
        },
        {
         "Effect": "Allow",
         "Action": ["ssmmessages:OpenDataChannel"],
         "Resource": ["arn:aws:ssm:*:*:session/${aws:userid}-*"]
       },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetConnectionStatus",
                "ssm:DescribeInstanceInformation"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:TerminateSession",
                "ssm:ResumeSession"
            ],
```

```
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:userid}-*"
            ]
        }
    ]
}
```

会话管理器, CLI, and Amazon EC2

使用此示例策略使管理员能够从 Session Manager 控制台、AWS CLI 和 Amazon EC2 控制台执行会话相关任务。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession",
                "ssm:SendCommand"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1:111122223333:instance/*"
            ],
            "Condition": {
                "StringLike": {
                    "ssm:resourceTag/tag-key": [
                        "tag-value"
                    ]
                }
            }
        },
        {
         "Effect": "Allow",
         "Action": ["ssmmessages:OpenDataChannel"],
         "Resource": ["arn:aws:ssm:*:*:session/${aws:userid}-*"]
       },
        {
            "Effect": "Allow",
            "Action": [
```

```
                "ssm:DescribeSessions",
                "ssm:GetConnectionStatus",
                "ssm:DescribeInstanceInformation",
                "ssm:DescribeInstanceProperties",
                "ec2:DescribeInstances"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:CreateDocument",
                "ssm:UpdateDocument",
                "ssm:GetDocument",
                "ssm:StartSession"
            ],
            "Resource": "arn:aws:ssm:us-east-1:111122223333:document/SSM-
SessionManagerRunShell"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:TerminateSession",
                "ssm:ResumeSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:userid}-*"
            ]
        }
    ]
}
```

[1] 如果用户尝试从 Amazon EC2 控制台启动会话，但必须发送命令以首先更新 SSM Agent，则需要 SendCommand 的权限。

Session Manager 的其他示例 IAM policy

请参阅以下示例策略，以帮助您为要支持的任何 Session Manager 用户访问场景创建自定义 AWS Identity and Access Management (IAM) policy。

主题

- 示例 1：在控制台中授予对文档的访问权限

- 示例 2：限制对特定托管式节点的访问
- 示例 3：根据标签限制访问
- 示例 4：仅允许用户结束自己启动的会话
- 示例 5：允许对所有会话进行完全（管理）访问

示例 1：在控制台中授予对文档的访问权限

您可以允许用户在使用会话管理器控制台启动会话时指定自定义文档。以下示例 IAM policy 授予访问指定 AWS 区域 和 AWS 账户 中名称以 **SessionDocument-** 开头的文档的权限。

将每个*#######*替换为您自己的信息。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetDocument",
                "ssm:ListDocuments"
            ],
            "Resource": [
                "arn:aws:ssm:us-east-1:111122223333:document/SessionDocument-*"
            ]
        }
    ]
}
```

> **Note**
>
> 会话管理器控制台仅支持用于定义会话首选项、其 `sessionType` 为 `Standard_Stream` 的会话文档。有关更多信息，请参阅 会话文档架构。

示例 2：限制对特定托管式节点的访问

您可以创建一个 IAM policy，用于定义允许用户使用 Session Manager 连接到哪些托管节点。例如，以下策略将授予用户在三个特定节点上开始、结束和恢复其会话的权限。该策略将限制用户连接到指定节点以外的节点。

> **Note**
>
> 有关联合用户的信息，请参阅 示例 4：仅允许用户结束自己启动的会话。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-2:123456789012:instance/
i-1234567890EXAMPLE",
                "arn:aws:ec2:us-east-2:123456789012:instance/i-
abcdefghijEXAMPLE",
                "arn:aws:ec2:us-east-2:123456789012:instance/
i-0e9d8c7b6aEXAMPLE",
                "arn:aws:ssm:us-east-2:123456789012:document/SSM-
SessionManagerRunShell"
            ]
        },
        {
         "Effect": "Allow",
         "Action": ["ssmmessages:OpenDataChannel"],
         "Resource": ["arn:aws:ssm:*:*:session/${aws:userid}-*"]
       },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:TerminateSession",
                "ssm:ResumeSession"
```

```
            ],
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:userid}-*"
            ]
        },
        {
         "Effect": "Allow",
         "Action": ["ssmmessages:OpenDataChannel"],
         "Resource": ["arn:aws:ssm:*:*:session/${aws:userid}-*"]
       }
    ]
}
```

示例 3：根据标签限制访问

您可以根据特定标签限制对托管式节点的访问。在以下示例中，允许用户在任何托管式节点 (Effect: Allow, Action: ssm:StartSession, ssm:ResumeSession) 上启动和恢复会话 (Resource: arn:aws:ec2:*region*:*987654321098*:instance/*)，条件是节点为 Finance WebServer (ssm:resourceTag/Finance: WebServer)。如果用户向未经标记或具有除 Finance: WebServer 以外的任何标签的托管式节点发送命令，则命令结果将包括 AccessDenied。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-2:123456789012:instance/*"
            ],
            "Condition": {
                "StringLike": {
                    "ssm:resourceTag/Finance": [
                        "WebServers"
                    ]
                }
            }
```

```
        },
        {
         "Effect": "Allow",
         "Action": ["ssmmessages:OpenDataChannel"],
         "Resource": ["arn:aws:ssm:*:*:session/${aws:userid}-*"]
       },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:TerminateSession",
                "ssm:ResumeSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:userid}-*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ssm:us-east-2:123456789012:document/SSM-
SessionManagerRunShell"
            ]
        }
    ]
}
```

您可以创建 IAM policy 来允许用户启动与使用多个标签标记的托管式节点的会话。以下策略允许用户启动与应用了指定标签的托管式节点的会话。如果用户向未使用这些标签标记的托管式节点发送命令，则命令结果将包括 AccessDenied。

JSON

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

```
            "ssm:StartSession"
         ],
         "Resource":"*",
         "Condition":{
            "StringLike":{
               "ssm:resourceTag/tag-key1":[
                  "tag-value1"
               ],
               "ssm:resourceTag/tag-key2":[
                  "tag-value2"
               ]
            }
         }
      },
      {
         "Effect": "Allow",
         "Action": ["ssmmessages:OpenDataChannel"],
         "Resource": ["arn:aws:ssm:*:*:session/${aws:userid}-*"]
       },
      {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ssm:us-east-2:123456789012:document/SSM-
SessionManagerRunShell"
            ]
      }
   ]
}
```

有关创建 IAM policy 的更多信息，请参阅《IAM 用户指南》中的托管策略与内联策略。有关标记托管式节点的更多信息，请参阅《Amazon EC2 用户指南》中的标记 Amazon EC2 资源（内容适用于 Windows 和 Linux 托管式节点）。有关在托管式节点上提高针对未授权根级别命令的安保状况的更多信息，请参阅 通过 SSM Agent 限制对根级别命令的访问

示例 4：仅允许用户结束自己启动的会话

Session Manager 提供了两种方法来控制允许您的 AWS 账户 中的联合用户结束哪些会话。

- 在 AWS Identity and Access Management (IAM) 权限策略中使用变量 `{aws:userid}`。联合用户只能结束自己启动的会话。对于非联合用户，请使用方法 1。对于联合用户，请使用方法 2。
- 在 IAM 权限策略中使用 AWS 标签提供的标签。在此策略中，您包含一个条件，该条件允许用户仅结束使用 AWS 提供的特定标签进行标记的会话。此方法适用于所有账户，包括使用联合 ID 授予 AWS 访问权限的账户。

方法 1：使用变量 **`{aws:username}`** 授予终止会话权限

以下 IAM policy 允许用户查看您账户中所有会话的 ID。但是，用户只能通过其启动的会话与托管式节点交互。分配了以下策略的用户无法连接或结束其他用户的会话。此策略使用变量 `{aws:username}` 来实现这一目的。

> **Note**
>
> 此方法不适用于使用联合 ID 授予 AWS 访问权限的账户。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ssm:DescribeSessions"
            ],
            "Effect": "Allow",
            "Resource": [
                "*"
            ]
        },
        {
            "Action": [
                "ssm:TerminateSession"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:username}-*"
            ]
        }
```

```
    ]
}
```

方法 2：使用 AWS 提供的标签授予 TerminateSession 权限

您可以通过在 IAM policy 中包括条件标签键变量，来控制用户可以结束哪些会话。该条件指定用户只能结束使用这些特定标记键变量和指定值之一或两者标记的会话。

当您的 AWS 账户中的用户启动会话时，Session Manager 会对会话应用两个资源标签。第一个资源标签是 `aws:ssmmessages:target-id`，用它可指定允许用户结束的目标的 ID。另一个资源标签是 `aws:ssmmessages:session-id`，值的格式为 *`role-id`*:*`caller-specified-role-name`*。

> **Note**
>
> Session Manager 不支持此 IAM 访问控制策略的自定义标签。您必须使用 AWS 提供的资源标签，如下所述。

`aws:ssmmessages:target-id`

使用此标签键，您可以在策略中包含托管式节点 ID 作为值。在以下策略数据块中，条件语句允许用户仅结束节点 i-02573cafcfEXAMPLE。

JSON

```
{
     "Version": "2012-10-17",
     "Statement": [
         {
             "Effect": "Allow",
             "Action": [
                "ssm:TerminateSession"
             ],
             "Resource": "*",
             "Condition": {
                 "StringLike": {
                     "ssm:resourceTag/aws:ssmmessages:target-id": [
                        "i-02573cafcfEXAMPLE"
                     ]
                 }
             }
```

```
        }
    ]
}
```

如果用户尝试结束未获得此 TerminateSession 权限的会话，则会收到 AccessDeniedException 错误。

aws:ssmmessages:session-id

此标签键包含会话 ID 的变量作为请求中的值来启动会话。

以下示例演示了调用方类型为 User 的情况的策略。您为 aws:ssmmessages:session-id 提供的值是用户的 ID。在此示例中，AIDIODR4TAW7CSEXAMPLE 表示您的 AWS 账户中的用户的 ID。要检索您的 AWS 账户中的用户的 ID，请使用 IAM 命令 get-user。有关信息，请参阅《IAM 用户指南》中 AWS Identity and Access Management 部分中的 get-user。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "ssm:TerminateSession"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "ssm:resourceTag/aws:ssmmessages:session-id": [
                       "AIDIODR4TAW7CSEXAMPLE"
                    ]
                }
            }
        }
    ]
}
```

以下示例演示了调用方类型为 AssumedRole 的情况的策略。针对您为 aws:ssmmessages:session-id 提供的值，您可以使用 {aws:userid} 变量。或者，您

可以针对您为 aws:ssmmessages:session-id 提供的值硬编码角色 ID。如果您对角色 ID 进行硬编码，则必须采用以格式 *role-id*:*caller-specified-role-name* 提供值。例如 AIDIODR4TAW7CSEXAMPLE:MyRole。

> **Important**
>
> 为了系统标签得以应用，您提供的角色 ID 只能包含以下字符：Unicode 字母、0-9、空格、_、.、:、/、=、+、-、@ 和 \。

要检索您的 AWS 账户中的角色的角色 ID，请使用 get-caller-identity 命令。有关更多信息，请参阅 AWS CLI 命令参考中的 get-caller-identity。

JSON

```
{
     "Version": "2012-10-17",
     "Statement": [
         {
             "Effect": "Allow",
             "Action": [
                "ssm:TerminateSession"
             ],
             "Resource": "*",
             "Condition": {
                 "StringLike": {
                     "ssm:resourceTag/aws:ssmmessages:session-id": [
                        "${aws:userid}*"
                     ]
                 }
             }
         }
     ]
}
```

如果用户尝试结束未获得此 TerminateSession 权限的会话，则会收到 AccessDeniedException 错误。

aws:ssmmessages:target-id 和 aws:ssmmessages:session-id

您还可以创建 IAM policy 来允许用户结束使用这两个系统标签标记的会话，如本示例所示。

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "ssm:TerminateSession"
         ],
         "Resource":"*",
         "Condition":{
            "StringLike":{
               "ssm:resourceTag/aws:ssmmessages:target-id":[
                  "i-02573cafcfEXAMPLE"
               ],
               "ssm:resourceTag/aws:ssmmessages:session-id":[
                  "${aws:userid}*"
               ]
            }
         }
      }
   ]
}
```

示例 5：允许对所有会话进行完全（管理）访问

以下 IAM policy 允许用户与所有托管式节点以及所有用户为所有节点创建的所有会话进行不受限制的交互。只应将其授予需要完全控制组织的 Session Manager 活动的管理员。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ssm:StartSession",
                "ssm:TerminateSession",
                "ssm:ResumeSession",
```

```
                "ssm:DescribeSessions",
                "ssm:GetConnectionStatus"
            ],
            "Effect": "Allow",
            "Resource": [
                "*"
            ]
        },
        {
         "Effect": "Allow",
         "Action": ["ssmmessages:OpenDataChannel"],
         "Resource": ["arn:aws:ssm:*:*:session/${aws:userid}-*"]
        }
    ]
}
```

步骤 4：配置会话首选项

在其 AWS Identity and Access Management（IAM）policy 中授予管理权限的用户可以配置会话首选项，包括以下内容：

- 为 Linux 托管式节点启用“运行身份”支持。这样，就可以使用指定操作系统用户的凭证启动会话，而非 `ssm-user` AWS Systems Manager 可以在托管式节点上创建的系统生成的 Session Manager 账户的凭证。
- 配置 Session Manager 以使用 AWS KMS key 加密，以便为在客户端计算机和托管式节点之间传输的数据提供额外保护。
- 配置 Session Manager，以创建会话历史记录日志并发送到 Amazon Simple Storage Service (Amazon S3) 存储桶或 Amazon CloudWatch Logs 日志组。然后，可以使用存储的日志数据来报告对您的托管节点进行的会话连接以及会话期间在这些节点上运行的命令。
- 配置会话超时。您可以使用此设置指定在处于不活动状态一段时间后结束会话。
- 配置 Session Manager 以使用可配置的 Shell 配置文件。您可以使用这些可自定义的配置文件定义会话中的首选项，例如 Shell 首选项、环境变量、工作目录以及在启动会话时运行多个命令。

有关配置 Session Manager 首选项所需权限的更多信息，请参阅the section called “授予或拒绝更新 Session Manager 首选项的用户权限”。

主题

- 授予或拒绝更新 Session Manager 首选项的用户权限

- 指定空闲会话超时值
- 指定最长会话持续时间
- 允许可配置的 Shell 配置文件
- 为 Linux 和 macOS 托管式节点开启“运行身份”支持
- 启用会话数据的 KMS 密钥加密（控制台）
- 创建 Session Manager 首选项文档（命令行）
- 更新 Session Manager 首选项（命令行）

有关使用 Systems Manager 控制台为记录会话数据配置选项的信息，请参阅以下主题：

- 使用 Amazon S3 记录会话数据（控制台）
- 使用 Amazon CloudWatch Logs 流式传输会话数据（控制台）
- 使用 Amazon CloudWatch Logs 记录会话数据（控制台）

授予或拒绝更新 Session Manager 首选项的用户权限

账户首选项存储为每个 AWS 区域的 AWS Systems Manager (SSM) 文档。必须先授予用户访问存储首选项的 SSM 文档类型所需的权限，之后他们才能更新账户中会话的账户首选项。这些权限是通过 AWS Identity and Access Management (IAM) 策略授予的。

允许创建和更新首选项的管理员策略

管理员可以随时使用以下策略来创建和更新首选项。以下策略允许访问和更新 us-east-2 账户 123456789012 中的 `SSM-SessionManagerRunShell` 文档。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ssm:CreateDocument",
                "ssm:GetDocument",
                "ssm:UpdateDocument",
                "ssm:DeleteDocument"
            ],
            "Effect": "Allow",
```

```
            "Resource": [
                "arn:aws:ssm:us-east-2:123456789012:document/SSM-
SessionManagerRunShell"
            ]
        }
    ]
}
```

阻止更新首选项的用户策略

使用以下策略可阻止账户中的最终用户更新或覆盖任何 Session Manager 首选项。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ssm:CreateDocument",
                "ssm:GetDocument",
                "ssm:UpdateDocument",
                "ssm:DeleteDocument"
            ],
            "Effect": "Deny",
            "Resource": [
                "arn:aws:ssm:us-east-1:111122223333:document/SSM-
SessionManagerRunShell"
            ]
        }
    ]
}
```

指定空闲会话超时值

Session Manager是 AWS Systems Manager 中的一项工具，使您能够指定在系统结束会话之前允许用户处于非活动状态的时间长度。默认情况下，会话在 20 分钟不活动后超时。您可以修改此设置，指定会话在处于不活动状态 1 到 60 分钟之间的时间后超时。一些专业计算安全机构建议将空闲会话超时设置为最多 15 分钟。

要允许空闲会话超时（控制台），请执行以下步骤：

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Session Manager。
3. 选择首选项选项卡，然后选择编辑。
4. 在 Idle session timeout (空闲会话超时) 的 Minutes (分钟) 字段中，指定在结束会话之前允许用户处于非活动状态的时间长度。
5. 选择保存。

指定最长会话持续时间

Session Manager（AWS Systems Manager 中的一项工具）允许您指定会话在结束前的最长持续时间。默认情况下，会话没有最长持续时间。您指定的最长会话持续时间值必须介于 1 到 1440 分钟之间。

指定最长会话持续时间（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Session Manager。
3. 选择首选项选项卡，然后选择编辑。
4. 选中 Enable maximum session duration（启用最长会话持续时间）旁边的复选框。
5. 在 Maximum session duration（最长会话持续时间）下的 minutes（分钟）字段中指定会话结束前的最长会话持续时间。
6. 选择保存。

允许可配置的 Shell 配置文件

默认情况下，适用于 Linux 的 EC2 实例上的会话使用 Bourne Shell（sh）启动。但是，您可能更喜欢使用 Bash 等其他 Shell。通过允许可配置的 Shell 配置文件，您可以自定义会话中的首选项，例如 Shell 首选项、环境变量、工作目录以及在启动会话时运行多个命令。

Important

Systems Manager 不会检查 Shell 配置文件中的命令或脚本，以查看它们在运行之前会对实例进行哪些更改。要限制用户修改在其 Shell 配置文件中输入的命令或脚本，建议执行以下操作：

- 为您的 AWS Identity and Access Management (IAM) 用户和角色创建自定义的会话类型文档。然后，修改这些用户和角色的 IAM policy，使 `StartSession` API 操作只能使用您为他们创建的会话类型文档。有关信息，请参阅 创建 Session Manager 首选项文档（命令行）和 Session Manager 的快速入门最终用户策略。
- 修改 IAM 用户和角色的 IAM policy，以拒绝访问您创建的会话类型文档资源的 `UpdateDocument` API 操作。这样，您的用户和角色便可以使用您为其会话首选项创建的文档，而不允许他们修改任何设置。

要启用可配置的 Shell 配置文件，请执行以下步骤：

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Session Manager。
3. 选择首选项选项卡，然后选择编辑。
4. 在适用操作系统的字段中指定要在会话启动时运行的环境变量、Shell 首选项或命令。
5. 选择保存。

以下是一些可以添加到 Shell 配置文件的示例命令。

在 Linux 实例上更改为 Bash Shell，并更改为 /usr 目录。

```
exec /bin/bash
cd /usr
```

在会话开始时输出时间戳和欢迎消息。

Linux & macOS

```
timestamp=$(date '+%Y-%m-%dT%H:%M:%SZ')
user=$(whoami)
```

```
echo $timestamp && echo "Welcome $user"'!'
echo "You have logged in to a production instance. Note that all session activity is
 being logged."
```

Windows

```
$timestamp = (Get-Date).ToString("yyyy-MM-ddTH:mm:ssZ")
$splitName = (whoami).Split("\")
$user = $splitName[1]
Write-Host $timestamp
Write-Host "Welcome $user!"
Write-Host "You have logged in to a production instance. Note that all session
 activity is being logged."
```

在会话开始时查看动态系统活动。

Linux & macOS

```
top
```

Windows

```
while ($true) { Get-Process | Sort-Object -Descending CPU | Select-Object -First 30;
 `
Start-Sleep -Seconds 2; cls
Write-Host "Handles  NPM(K)    PM(K)      WS(K) VM(M)   CPU(s)     Id ProcessName";
Write-Host "-------  ------    -----      ----- -----   ------     -- -----------"}
```

为 Linux 和 macOS 托管式节点开启“运行身份”支持

默认情况下，Session Manager 将使用在托管节点上创建的系统生成的 `ssm-user` 账户的凭证，对连接进行身份验证。（在 Linux 和 macOS 计算机上，此账户添加到 `/etc/sudoers/`。）如果您选择，则可以改为使用操作系统 (OS) 用户账户或已加入 Active Directory 的示例的域用户的凭据对会话进行身份验证。在这种情况下，Session Manager 会在启动会话之前验证节点上或域中是否存在您指定的操作系统账户。如果您尝试使用节点上或域中不存在的操作系统账户启动会话，则连接将失败。

> **Note**
>
> Session Manager 不支持使用操作系统的 `root` 用户账户对连接进行身份验证。对于使用操作系统用户账户进行身份验证的会话，节点的操作系统级别和目录策略（如登录限制或系统资源使用限制）可能不适用。

工作方式

如果为会话启用“运行身份”支持，系统将检查访问权限，如下所示：

1. 对于正在启动会话的用户，是否使用 `SSMSessionRunAs = os user account name` 标记了其 IAM 实体（用户或角色）？

   如果是，托管节点上是否存在该操作系统用户名？ 如果存在，启动会话。否则，不允会话启动。

   如果尚未使用 `SSMSessionRunAs = os user account name` 标记 IAM 实体，请继续执行步骤 2。
2. 如果尚未使用 `SSMSessionRunAs = os user account name` 标记 IAM 实体，是否已在 AWS 账户 的 Session Manager 首选项中指定了操作系统用户名？

   如果是，托管节点上是否存在该操作系统用户名？ 如果存在，启动会话。否则，不允会话启动。

> **Note**
>
> 当您激活“运行方式”支持时，它将阻止 Session Manager 使用托管节点上的 `ssm-user` 账户启动会话。这意味着，如果 Session Manager 无法使用指定的操作系统用户账户进行连接，则它不会退回到使用默认方法进行连接。
> 如果您在未指定操作系统账户或标记 IAM 实体的情况下激活“运行方式”，并且未在 Session Manager 首选项中指定操作系统账户，则会话连接尝试将失败。

为 Linux 和 macOS 托管式节点启用“运行身份”支持

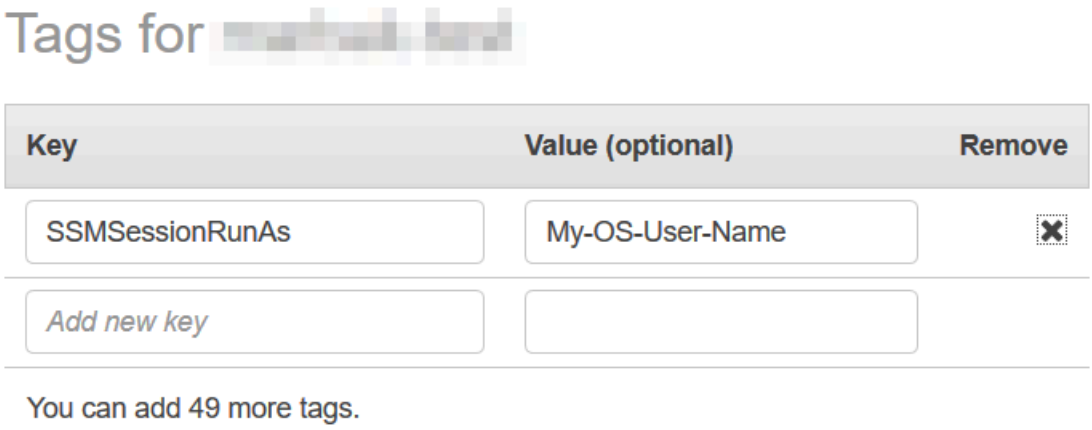
1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Session Manager。
3. 选择首选项选项卡，然后选择编辑。

4. 选中为 Linux 实例启用运行身份支持旁边的复选框。
5. 请执行以下操作之一：

   - 选项 1：在操作系统用户名字段中，输入您要用于启动会话的操作系统用户账户的名称。使用此选项，所有会话都由同一操作系统用户为您的 AWS 账户 中使用 Session Manager 连接的所有用户运行。
   - 选项 2（建议）：选择打开 IAM 控制台链接。在导航窗格中，选择用户或角色。选择要将标签添加到的实体（用户或角色），然后选择标签选项卡。对于密钥名称，输入 `SSMSessionRunAs`。对于密钥值，输入操作系统用户账户的名称。选择保存更改。

     如果您选择此选项，则您可以使用此选项为不同 IAM 实体指定唯一的操作系统用户。有关标记 IAM 实体（用户或角色）的更多信息，请参阅《IAM 用户指南》中的标记 IAM 资源。

     示例如下：



6. 选择保存。

启用会话数据的 KMS 密钥加密（控制台）

使用 AWS Key Management Service（AWS KMS）创建和管理加密密钥。借助 AWS KMS，您可以控制各个 AWS 服务之间以及应用程序中对加密的使用。您可以指定使用 KMS 密钥加密方式对在托管式节点与 AWS 账户 中用户的本地计算机之间传输的会话数据进行加密。（这是对 AWS 默认提供的 TLS 1.2/1.3 加密的补充）。要加密 Session Manager 会话数据，请使用 AWS KMS 创建对称 KMS 密钥。

AWS KMS 加密适用于 `Standard_Stream`、`InteractiveCommands` 和 `NonInteractiveCommands` 会话类型。要通过该选项使用在 AWS KMS 中创建的密钥加密会话数据，必须在托管式节点上安装 2.3.539.0 版本或更高版本的 AWS Systems Manager SSM Agent。

> **Note**
>
> 您必须允许 AWS KMS 加密，才能从 AWS Systems Manager 控制台重置托管式节点上的密码。有关更多信息，请参阅 在托管式节点上重置密码。

您可以使用您在 AWS 账户中创建的密钥。您还可以使用在其他 AWS 账户中创建的密钥。其他 AWS 账户中密钥的创建者必须为您提供使用密钥所需的权限。

在您为会话数据启用 KMS 密钥加密后，启动会话的用户及其所连接的托管式节点都必须具有使用密钥的权限。您可以通过 AWS Identity and Access Management (IAM) 策略向 Session Manager 提供使用 KMS 密钥的权限。有关更多信息，请参阅以下主题：

- 为您账户中的用户添加 AWS KMS 权限：Session Manager 的 IAM policy 示例。
- 为您账户中的托管式节点添加 AWS KMS 权限：步骤 2：验证或添加 Session Manager 的实例权限。

有关创建和管理 KMS 密钥的更多信息，请参阅 AWS Key Management Service 开发人员指南。

有关使用 AWS CLI 在您的账户中启用会话数据的 KMS 密钥加密的信息，请参阅 创建 Session Manager 首选项文档（命令行） 或 更新 Session Manager 首选项（命令行）。

> **Note**
>
> 使用 KMS 密钥需支付费用。有关更多信息，请参阅 AWS Key Management Service 定价。

要启用会话数据的 KMS 密钥加密（控制台），请执行以下步骤：

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Session Manager。
3. 选择首选项选项卡，然后选择编辑。
4. 选中 Enable KMS encryption（启用 KMS 加密）旁边的复选框。
5. 请执行以下操作之一：
   - 选择 Select a KMS key in my current account (选择我当前账户中的 KMS 密钥) 旁边的按钮，然后从列表中选择一个密钥。

-或者-

选择输入 KMS 密钥别名或 KMS 密钥 ARN 旁边的按钮。手动输入在您的当前账户中创建的密钥的 KMS 密钥别名，或输入另一账户中的密钥的密钥 Amazon Resource Name (ARN)。示例如下：

- 密钥别名：`alias/my-kms-key-alias`
- 密钥 ARN：`arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-12345EXAMPLE`

-或者-

选择 Create new key (创建新密钥)，在您的账户中创建新 KMS 密钥。在创建新密钥后，返回到 Preferences (首选项) 选项卡，然后选择用于在您的账户中加密会话数据的密钥。

有关共享密钥的更多信息，请参阅《AWS Key Management Service 开发人员指南》中的允许外部 AWS 账户 访问密钥。

6. 选择保存。

创建 Session Manager 首选项文档（命令行）

使用以下步骤创建 SSM 文档，这些文档定义了您的 AWS Systems ManagerSession Manager 会话首选项。您可以使用该文档来配置会话选项，包括数据加密、会话持续时间和日志记录。例如，您可以指定是否要在 Amazon Simple Storage Service（Amazon S3）存储桶或 Amazon CloudWatch Logs 日志组中存储会话日志数据。您可以创建定义 AWS 账户 和 AWS 区域 所有会话的常规首选项的文档，也可以创建定义各个会话首选项的文档。

> **Note**
>
> 您也可以使用会话管理器控制台配置常规会话首选项。

用于设置会话管理器首选项的文档必须有 `Standard_Stream` 的 `sessionType`。有关会话文档的更多信息，请参阅 the section called “会话文档架构”。

有关使用命令行更新现有 Session Manager 首选项的信息，请参阅 更新 Session Manager 首选项（命令行）。

有关如何使用 AWS CloudFormation 创建会话首选项的示例，请参阅《AWS CloudFormation 用户指南》中的为 Session Manager 首选项创建 Systems Manager 文档。

> **Note**
>
> 此过程介绍如何创建用于在 AWS 账户 级别设置 Session Manager 首选项的文档。
> 要创建用于设置会话级别首选项的文档，请为与文件名相关的命令输入指定 `SSM-SessionManagerRunShell` 以外的值。
> 要使用您的文档为从 AWS Command Line Interface（AWS CLI）开始的会话设置首选项，请提供文档名称作为 `--document-name` 参数值。要为从会话管理器控制台启动的会话设置首选项，可以键入或从列表中选择文档的名称。

创建 Session Manager 首选项（命令行）

1. 在本地计算机上使用 `SessionManagerRunShell.json` 之类的名称创建一个 JSON 文件，然后将以下内容粘贴到此文件中。

```
{
    "schemaVersion": "1.0",
    "description": "Document to hold regional settings for Session Manager",
    "sessionType": "Standard_Stream",
    "inputs": {
        "s3BucketName": "",
        "s3KeyPrefix": "",
        "s3EncryptionEnabled": true,
        "cloudWatchLogGroupName": "",
        "cloudWatchEncryptionEnabled": true,
        "cloudWatchStreamingEnabled": false,
        "kmsKeyId": "",
        "runAsEnabled": false,
        "runAsDefaultUser": "",
        "idleSessionTimeout": "",
        "maxSessionDuration": "",
        "shellProfile": {
            "windows": "date",
            "linux": "pwd;ls"
        }
    }
}
```

您还可以使用参数将值传递到会话首选项，而不是硬编码这些值，如以下示例所示。

```
{
   "schemaVersion":"1.0",
   "description":"Session Document Parameter Example JSON Template",
   "sessionType":"Standard_Stream",
   "parameters":{
      "s3BucketName":{
         "type":"String",
         "default":""
      },
      "s3KeyPrefix":{
         "type":"String",
         "default":""
      },
      "s3EncryptionEnabled":{
         "type":"Boolean",
         "default":"false"
      },
      "cloudWatchLogGroupName":{
         "type":"String",
         "default":""
      },
      "cloudWatchEncryptionEnabled":{
         "type":"Boolean",
         "default":"false"
      }
   },
   "inputs":{
      "s3BucketName":"{{s3BucketName}}",
      "s3KeyPrefix":"{{s3KeyPrefix}}",
      "s3EncryptionEnabled":"{{s3EncryptionEnabled}}",
      "cloudWatchLogGroupName":"{{cloudWatchLogGroupName}}",
      "cloudWatchEncryptionEnabled":"{{cloudWatchEncryptionEnabled}}",
      "kmsKeyId":""
   }
}
```

2. 指定要发送会话数据的位置。您可以指定 S3 存储桶名称（包含可选前缀）或 CloudWatch Logs 日志组名称。如果您要进一步加密本地客户端与托管式节点之间的数据，请提供用于加密的 KMS 密钥。示例如下：

```
{
  "schemaVersion": "1.0",
  "description": "Document to hold regional settings for Session Manager",
  "sessionType": "Standard_Stream",
  "inputs": {
    "s3BucketName": "amzn-s3-demo-bucket",
    "s3KeyPrefix": "MyS3Prefix",
    "s3EncryptionEnabled": true,
    "cloudWatchLogGroupName": "MyLogGroupName",
    "cloudWatchEncryptionEnabled": true,
    "cloudWatchStreamingEnabled": false,
    "kmsKeyId": "MyKMSKeyID",
    "runAsEnabled": true,
    "runAsDefaultUser": "MyDefaultRunAsUser",
    "idleSessionTimeout": "20",
    "maxSessionDuration": "60",
    "shellProfile": {
        "windows": "MyCommands",
        "linux": "MyCommands"
    }
  }
}
```

Note

如果不需要加密会话日志数据，请将 `s3EncryptionEnabled` 的 `true` 更改为 `false`。如果您不是将日志发送到 Amazon S3 存储桶或 CloudWatch Logs 日志组，不希望加密活动会话的数据，或不希望为账户中的会话启用“运行身份”支持，则可以删除这些选项的行。确保 `inputs` 部分中的最后一行不以逗号结尾。

如果您添加 KMS 密钥 ID 来加密会话数据，启动会话的用户及其所连接的托管式节点都必须具有使用该密钥的权限。您可以通过 IAM policy 向 Session Manager 提供使用 KMS 密钥的权限。有关信息，请参阅以下主题：

- 为您账户中的用户添加 AWS KMS 权限：Session Manager 的 IAM policy 示例。
- 为账户中的托管式节点添加 AWS KMS 权限：步骤 2：验证或添加 Session Manager 的实例权限

3. 保存该文件。
4. 在创建此 JSON 文件的目录中，运行以下命令。

Linux & macOS

```
aws ssm create-document \
    --name SSM-SessionManagerRunShell \
    --content "file://SessionManagerRunShell.json" \
    --document-type "Session" \
    --document-format JSON
```

Windows

```
aws ssm create-document ^
    --name SSM-SessionManagerRunShell ^
    --content "file://SessionManagerRunShell.json" ^
    --document-type "Session" ^
    --document-format JSON
```

PowerShell

```
New-SSMDocument `
    -Name "SSM-SessionManagerRunShell" `
    -Content (Get-Content -Raw SessionManagerRunShell.json) `
    -DocumentType "Session" `
    -DocumentFormat JSON
```

如果成功，该命令将返回类似于以下内容的输出。

```
{
    "DocumentDescription": {
        "Status": "Creating",
        "Hash": "ce4fd0a2ab9b0fae759004ba603174c3ec2231f21a81db8690a33eb66EXAMPLE",
        "Name": "SSM-SessionManagerRunShell",
        "Tags": [],
        "DocumentType": "Session",
        "PlatformTypes": [
            "Windows",
            "Linux"
        ],
        "DocumentVersion": "1",
        "HashType": "Sha256",
        "CreatedDate": 1547750660.918,
```

```
        "Owner": "111122223333",
        "SchemaVersion": "1.0",
        "DefaultVersion": "1",
        "DocumentFormat": "JSON",
        "LatestVersion": "1"
    }
}
```

更新 Session Manager 首选项（命令行）

以下过程介绍了如何使用首选的命令行工具在选择的 AWS 区域中为您的 AWS 账户更改 AWS Systems Manager Session Manager 首选项。使用 Session Manager 首选项来指定在 Amazon Simple Storage Service (Amazon S3) 存储桶或 Amazon CloudWatch Logs 日志组中记录会话数据的选项。您还可以使用 Session Manager 首选项来加密您的会话数据。

更新 Session Manager 首选项（命令行）

1. 在本地计算机上使用 `SessionManagerRunShell.json` 之类的名称创建一个 JSON 文件，然后将以下内容粘贴到此文件中。

```
{
    "schemaVersion": "1.0",
    "description": "Document to hold regional settings for Session Manager",
    "sessionType": "Standard_Stream",
    "inputs": {
        "s3BucketName": "",
        "s3KeyPrefix": "",
        "s3EncryptionEnabled": true,
        "cloudWatchLogGroupName": "",
        "cloudWatchEncryptionEnabled": true,
        "cloudWatchStreamingEnabled": false,
        "kmsKeyId": "",
        "runAsEnabled": true,
        "runAsDefaultUser": "",
        "idleSessionTimeout": "",
        "maxSessionDuration": "",
        "shellProfile": {
            "windows": "date",
            "linux": "pwd;ls"
        }
    }
```

```
}
```

2. 指定要发送会话数据的位置。您可以指定 S3 存储桶名称（包含可选前缀）或 CloudWatch Logs 日志组名称。如果您要进一步加密本地客户端与托管式节点之间的数据，请提供用于加密的 AWS KMS key。示例如下：

```
{
  "schemaVersion": "1.0",
  "description": "Document to hold regional settings for Session Manager",
  "sessionType": "Standard_Stream",
  "inputs": {
    "s3BucketName": "amzn-s3-demo-bucket",
    "s3KeyPrefix": "MyS3Prefix",
    "s3EncryptionEnabled": true,
    "cloudWatchLogGroupName": "MyLogGroupName",
    "cloudWatchEncryptionEnabled": true,
    "cloudWatchStreamingEnabled": false,
    "kmsKeyId": "MyKMSKeyID",
    "runAsEnabled": true,
    "runAsDefaultUser": "MyDefaultRunAsUser",
    "idleSessionTimeout": "20",
    "maxSessionDuration": "60",
    "shellProfile": {
        "windows": "MyCommands",
        "linux": "MyCommands"
    }
  }
}
```

> **Note**
>
> 如果不需要加密会话日志数据，请将 `s3EncryptionEnabled` 的 `true` 更改为 `false`。如果您不是将日志发送到 Amazon S3 存储桶或 CloudWatch Logs 日志组，不希望加密活动会话的数据，或不希望为账户中的会话启用“运行身份”支持，则可以删除这些选项的行。确保 `inputs` 部分中的最后一行不以逗号结尾。
>
> 如果您添加 KMS 密钥 ID 来加密会话数据，启动会话的用户及其所连接的托管式节点都必须具有使用该密钥的权限。您可以通过 AWS Identity and Access Management (IAM) 策略向 Session Manager 提供使用 KMS 密钥的权限。有关信息，请参阅以下主题：
>
> - 为您账户中的用户添加 AWS KMS 权限：Session Manager 的 IAM policy 示例。

- 为您账户中的托管式节点添加 AWS KMS 权限：步骤 2：验证或添加 Session Manager 的实例权限。

3. 保存该文件。
4. 在创建此 JSON 文件的目录中，运行以下命令。

Linux & macOS

```
aws ssm update-document \
    --name "SSM-SessionManagerRunShell" \
    --content "file://SessionManagerRunShell.json" \
    --document-version "\$LATEST"
```

Windows

```
aws ssm update-document ^
    --name "SSM-SessionManagerRunShell" ^
    --content "file://SessionManagerRunShell.json" ^
    --document-version "$LATEST"
```

PowerShell

```
Update-SSMDocument `
    -Name "SSM-SessionManagerRunShell" `
    -Content (Get-Content -Raw SessionManagerRunShell.json) `
    -DocumentVersion '$LATEST'
```

如果成功，该命令将返回类似于以下内容的输出。

```
{
    "DocumentDescription": {
        "Status": "Updating",
        "Hash": "ce4fd0a2ab9b0fae759004ba603174c3ec2231f21a81db8690a33eb66EXAMPLE",
        "Name": "SSM-SessionManagerRunShell",
        "Tags": [],
        "DocumentType": "Session",
        "PlatformTypes": [
            "Windows",
            "Linux"
```

```
        ],
        "DocumentVersion": "2",
        "HashType": "Sha256",
        "CreatedDate": 1537206341.565,
        "Owner": "111122223333",
        "SchemaVersion": "1.0",
        "DefaultVersion": "1",
        "DocumentFormat": "JSON",
        "LatestVersion": "2"
    }
}
```

步骤 5：（可选）限制对会话中命令的访问

您可以通过使用自定义 `Session` 类型 AWS Systems Manager（SSM）文档来限制用户可以在 AWS Systems Manager Session Manager 会话中运行的命令。在文档中，您可以定义用户启动会话时运行的命令以及用户可以向命令提供的参数。`Session` 文档的 `schemaVersion` 必须为 1.0，文档的 `sessionType` 必须为 `InteractiveCommands`。然后，您可以创建 AWS Identity and Access Management（IAM）policy 以仅允许用户访问您定义的 `Session` 文档。有关使用 IAM policy 限制对会话中命令的访问的更多信息，请参阅 交互式命令的 IAM policy 示例。

仅从 AWS Command Line Interface（AWS CLI）启动的会话支持 `sessionType` 为 `InteractiveCommands` 的文档。用户提供自定义文档名称作为 `--document-name` 参数值，并使用 `--parameters` 选项提供任何命令参数值。有关运行交互式命令的更多信息，请参阅 启动会话（交互式和非交互式命令）。

使用以下过程创建自定义 `Session` 类型 SSM 文档来定义允许用户运行的命令。

限制对会话中命令的访问（控制台）

限制用户可以在 Session Manager 会话中运行的命令（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择文档。
3. 选择 Create command or session (创建命令或会话)。
4. 对于 Name (名称)，为文档输入一个描述性名称。
5. 对于 Document type (文档类型)，选择 Session document (会话文档)。

6. 输入文档内容，用来定义用户可以使用 JSON 或 YAML 在 Session Manager 会话中运行的命令，如以下示例所示。

   YAML

```
---
schemaVersion: '1.0'
description: Document to view a log file on a Linux instance
sessionType: InteractiveCommands
parameters:
  logpath:
    type: String
    description: The log file path to read.
    default: "/var/log/amazon/ssm/amazon-ssm-agent.log"
    allowedPattern: "^[a-zA-Z0-9-_/]+(.log)$"
properties:
  linux:
    commands: "tail -f {{ logpath }}"
    runAsElevated: true
```

   JSON

```
{
    "schemaVersion": "1.0",
    "description": "Document to view a log file on a Linux instance",
    "sessionType": "InteractiveCommands",
    "parameters": {
        "logpath": {
            "type": "String",
            "description": "The log file path to read.",
            "default": "/var/log/amazon/ssm/amazon-ssm-agent.log",
            "allowedPattern": "^[a-zA-Z0-9-_/]+(.log)$"
        }
    },
    "properties": {
        "linux": {
            "commands": "tail -f {{ logpath }}",
            "runAsElevated": true
        }
    }
}
```

7. 选择创建文档。

限制对会话中命令的访问（命令行）

开始前的准备工作

安装并配置 AWS Command Line Interface (AWS CLI) 或 AWS Tools for PowerShell（如果尚未这样做）。有关信息，请参阅安装或更新 AWS CLI 的最新版本以及安装 AWS Tools for PowerShell。

限制用户可以在 Session Manager 会话中运行的命令（命令行）

1. 为文档内容创建 JSON 或 YAML 文件，用来定义用户可在 Session Manager 会话中运行的命令，如以下示例所示。

   YAML

```
---
schemaVersion: '1.0'
description: Document to view a log file on a Linux instance
sessionType: InteractiveCommands
parameters:
  logpath:
    type: String
    description: The log file path to read.
    default: "/var/log/amazon/ssm/amazon-ssm-agent.log"
    allowedPattern: "^[a-zA-Z0-9-_/]+(.log)$"
properties:
  linux:
    commands: "tail -f {{ logpath }}"
    runAsElevated: true
```

   JSON

```
{
    "schemaVersion": "1.0",
    "description": "Document to view a log file on a Linux instance",
    "sessionType": "InteractiveCommands",
    "parameters": {
        "logpath": {
            "type": "String",
            "description": "The log file path to read.",
            "default": "/var/log/amazon/ssm/amazon-ssm-agent.log",
            "allowedPattern": "^[a-zA-Z0-9-_/]+(.log)$"
        }
    },
```

```
    "properties": {
        "linux": {
            "commands": "tail -f {{ logpath }}",
            "runAsElevated": true
        }
    }
}
```

2. 运行以下命令，使用您的内容创建 SSM 文档，该内容定义了用户可以在 Session Manager 会话中运行的命令。

   Linux & macOS

   ```
   aws ssm create-document \
       --content file://path/to/file/documentContent.json \
       --name "exampleAllowedSessionDocument" \
       --document-type "Session"
   ```

   Windows

   ```
   aws ssm create-document ^
       --content file://C:\path\to\file\documentContent.json ^
       --name "exampleAllowedSessionDocument" ^
       --document-type "Session"
   ```

   PowerShell

   ```
   $json = Get-Content -Path "C:\path\to\file\documentContent.json" | Out-String
   New-SSMDocument `
       -Content $json `
       -Name "exampleAllowedSessionDocument" `
       -DocumentType "Session"
   ```

交互式命令参数和 AWS CLI

在使用 AWS CLI 时，可以通过多种方式提供交互式命令参数。根据您用于通过 AWS CLI 连接到托管式节点的客户端计算机的操作系统 (OS)，您为包含特殊字符或转义字符的命令提供的语法可能不同。以下示例显示了在使用 AWS CLI 时提供命令参数的一些不同方法，以及如何处理特殊字符或转义字符。

您的命令参数可在 AWS CLI 中引用存储在 Parameter Store 中的参数，如以下示例所示。

Linux & macOS

```
aws ssm start-session \
    --target instance-id \
    --document-name MyInteractiveCommandDocument \
    --parameters '{"command":["{{ssm:mycommand}}"]}'
```

Windows

```
aws ssm start-session ^
    --target instance-id ^
    --document-name MyInteractiveCommandDocument ^
    --parameters '{"command":["{{ssm:mycommand}}"]}'
```

以下示例显示了如何使用 AWS CLI 的简写语法来传递参数。

Linux & macOS

```
aws ssm start-session \
    --target instance-id \
    --document-name MyInteractiveCommandDocument \
    --parameters command="ifconfig"
```

Windows

```
aws ssm start-session ^
    --target instance-id ^
    --document-name MyInteractiveCommandDocument ^
    --parameters command="ipconfig"
```

您也可以提供 JSON 格式的参数，如以下示例所示。

Linux & macOS

```
aws ssm start-session \
    --target instance-id \
    --document-name MyInteractiveCommandDocument \
    --parameters '{"command":["ifconfig"]}'
```

Windows

```
aws ssm start-session ^
    --target instance-id ^
    --document-name MyInteractiveCommandDocument ^
    --parameters '{"command":["ipconfig"]}'
```

参数也可以存储在 JSON 文件中并提供给 AWS CLI，如以下示例所示。有关从文件中使用 AWS CLI 参数的更多信息，请参阅《AWS Command Line Interface 用户指南》中的从文件中加载 AWS CLI 参数。

```
{
    "command": [
        "my command"
    ]
}
```

Linux & macOS

```
aws ssm start-session \
    --target instance-id \
    --document-name MyInteractiveCommandDocument \
    --parameters file://complete/path/to/file/parameters.json
```

Windows

```
aws ssm start-session ^
    --target instance-id ^
    --document-name MyInteractiveCommandDocument ^
    --parameters file://complete/path/to/file/parameters.json
```

您还可以从 JSON 输入文件生成 AWS CLI 骨架，如以下示例所示。有关从 JSON 输入文件生成 AWS CLI 骨架的更多信息，请参阅《AWS Command Line Interface 用户指南》中的从 JSON 或 YAML 输入文件生成 AWS CLI 骨架和输入参数。

```
{
    "Target": "instance-id",
    "DocumentName": "MyInteractiveCommandDocument",
    "Parameters": {
```

```
        "command": [
            "my command"
        ]
    }
}
```

Linux & macOS

```
aws ssm start-session \
    --cli-input-json file://complete/path/to/file/parameters.json
```

Windows

```
aws ssm start-session ^
    --cli-input-json file://complete/path/to/file/parameters.json
```

要对引号内的字符进行转义，必须在转义字符中添加额外的反斜杠，如以下示例所示。

Linux & macOS

```
aws ssm start-session \
    --target instance-id \
    --document-name MyInteractiveCommandDocument \
    --parameters '{"command":["printf \"abc\\\\tdef\""]}'
```

Windows

```
aws ssm start-session ^
    --target instance-id ^
    --document-name MyInteractiveCommandDocument ^
    --parameters '{"command":["printf \"abc\\\\tdef\""]}'
```

有关在 AWS CLI 中将引号和命令参数结合使用的信息，请参阅《AWS Command Line Interface 用户指南》中的在 AWS CLI 中将引号和字符串结合使用。

交互式命令的 IAM policy 示例

您可以创建 IAM policy 以仅允许用户访问您定义的 `Session` 文档。这将用户可以在 Session Manager 会话中运行的命令限制为仅在自定义 `Session` 类型 SSM 文档中定义的命令。

允许用户在单个托管式节点上运行交互式命令

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":"ssm:StartSession",
         "Resource":[
            "arn:aws:ec2:us-east-1:444455556666:instance/i-02573cafcfEXAMPLE",
            "arn:aws:ssm:us-east-1:444455556666:document/allowed-session-
document"
         ]
      },
      {
         "Effect": "Allow",
         "Action": ["ssmmessages:OpenDataChannel"],
         "Resource": ["arn:aws:ssm:*:*:session/${aws:userid}-*"]
      }
   ]
}
```

允许用户在所有托管式节点上运行交互式命令

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":"ssm:StartSession",
         "Resource":[
            "arn:aws:ec2:us-east-1:444455556666:instance/*",
            "arn:aws:ssm:us-east-1:444455556666:document/allowed-session-
document"
         ]
      },
      {
         "Effect": "Allow",
         "Action": ["ssmmessages:OpenDataChannel"],
         "Resource": ["arn:aws:ssm:*:*:session/${aws:userid}-*"]
```

```
      }
   ]
}
```

允许用户在所有托管式节点上运行多个交互式命令

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":"ssm:StartSession",
         "Resource":[
            "arn:aws:ec2:us-east-1:444455556666:instance/*",
            "arn:aws:ssm:us-east-1:444455556666:document/allowed-session-
document",
            "arn:aws:ssm:us-east-1:444455556666:document/allowed-session-
document-2"
         ]
      },
      {
         "Effect": "Allow",
         "Action": ["ssmmessages:OpenDataChannel"],
         "Resource": ["arn:aws:ssm:*:*:session/${aws:userid}-*"]
      }
   ]
}
```

步骤 6：（可选）使用 AWS PrivateLink 为 Session Manager 设置 VPC 端点

您可以通过将 AWS Systems Manager 配置为使用接口 Virtual Private Cloud (VPC) 终端节点来进一步提高托管式节点的安保状况。接口端点由 AWS PrivateLink 提供支持，该技术使您能够通过使用私有 IP 地址私下访问 Amazon Elastic Compute Cloud (Amazon EC2) 和 Systems Manager API。

AWS PrivateLink 将托管式节点、Systems Manager 和 Amazon EC2 之间的所有网络流量限制在 Amazon 网络以内。（托管式节点无法访问互联网。）而且，您无需 Internet 网关、NAT 设备或虚拟专用网关。

有关创建 VPC 端点的信息，请参阅使用适用于 Systems Manager 的 VPC 终端节点提高 EC2 实例的安全性。

使用 VPC 终端节点的替代方法是，在托管式节点上允许出站互联网访问。在这种情况下，托管式节点还必须允许到以下端点的 HTTPS（端口 443）出站流量：

- ec2messages.*region*.amazonaws.com
- ssm.*region*.amazonaws.com
- ssmmessages.*region*.amazonaws.com

Systems Manager 使用上述最后一个端点（即 ssmmessages.*region*.amazonaws.com）从 SSM Agent 对云中的 Session Manager 服务进行调用。

要使用可选功能，例如 AWS Key Management Service (AWS KMS) 加密、将日志流式传输至 Amazon CloudWatch Logs (CloudWatch Logs) 以及将日志发送到 Amazon Simple Storage Service (Amazon S3)，您必须允许到以下端点的 HTTPS（端口 443）出站流量：

- kms.*region*.amazonaws.com
- logs.*region*.amazonaws.com
- s3.*region*.amazonaws.com

有关 Systems Manager 所需的端点的更多信息，请参阅参考：ec2messages、ssmmessages 和其他 API 操作。

步骤 7：（可选）开启或关闭 ssm-user 账户管理权限

从 AWS Systems Manager SSM Agent 版本 2.3.50.0 开始，此代理会创建一个名为 ssm-user 的本地用户账户，并将其添加到 /etc/sudoers（Linux 和 macOS）或管理员组（Windows）。在 2.3.612.0 之前的代理版本上，账户会在 SSM Agent 安装后首次启动或重启时创建。在版本 2.3.612.0 及更高版本上，会话在节点上首次启动时会创建 ssm-user 账户。当 AWS Systems Manager Session Manager 会话启动时，此 ssm-user 为默认操作系统（OS）用户。SSM Agent 版本 2.3.612.0 于 2019 年 5 月 8 日发布。

如果要阻止 Session Manager 用户在节点上运行管理命令，可以更新 ssm-user 账户权限。这些权限在删除后还可以恢复。

主题

- 在 Linux 和 macOS 上管理 ssm-user sudo 账户权限
- 在 Windows Server 上管理 ssm-user 管理员账户权限

在 Linux 和 macOS 上管理 ssm-user sudo 账户权限

使用以下过程之一在 Linux 和 macOS 托管式节点上启用或禁用 ssm-user 账户 sudo 权限。

使用 Run Command 修改 ssm-user sudo 权限（控制台）

- 在从控制台运行命令中的过程中使用以下值：
  - 对于 Command document (命令文档)，选择 AWS-RunShellScript。
  - 要删除 sudo 访问权限，请在 Command parameters (命令参数) 区域中，将以下内容粘贴到 Commands (命令) 框中：

    ```
    cd /etc/sudoers.d
    echo "#User rules for ssm-user" > ssm-agent-users
    ```

    -或者-

    要恢复 sudo 访问权限，请在 Command parameters (命令参数) 区域中，将以下内容粘贴到 Commands (命令) 框中：

    ```
    cd /etc/sudoers.d
    echo "ssm-user ALL=(ALL) NOPASSWD:ALL" > ssm-agent-users
    ```

使用命令行修改 ssm-user sudo 权限 (AWS CLI)

1. 连接到托管式节点并运行以下命令。

   ```
   sudo -s
   ```

2. 使用以下命令更改工作目录。

   ```
   cd /etc/sudoers.d
   ```

3. 打开名为 ssm-agent-users 的文件进行编辑。
4. 要删除 sudo 访问权限，请删除以下行。

   ```
   ssm-user ALL=(ALL) NOPASSWD:ALL
   ```

   -或者-

要恢复 sudo 访问权限，请添加以下行。

```
ssm-user ALL=(ALL) NOPASSWD:ALL
```

5. 保存该文件。

在 Windows Server 上管理 ssm-user 管理员账户权限

使用以下过程之一在 Windows Server 托管式节点上开启或关闭 ssm-user 账户管理员权限。

使用 Run Command 修改管理员权限（控制台）

- 在从控制台运行命令中的过程中使用以下值：

  对于 Command document (命令文档)，选择 AWS-RunPowerShellScript。

  要删除管理访问权限，请在 Command parameters (命令参数) 区域中，将以下内容粘贴到 Commands (命令) 框中：

  ```
  net localgroup "Administrators" "ssm-user" /delete
  ```

  -或者-

  要恢复管理访问权限，请在 Command parameters (命令参数) 区域中，将以下内容粘贴到 Commands (命令) 框中：

  ```
  net localgroup "Administrators" "ssm-user" /add
  ```

使用 PowerShell 或命令提示符窗口修改管理员权限

1. 连接到托管式节点并打开 PowerShell 或命令提示符窗口。
2. 要删除管理访问权限，请运行以下命令。

   ```
   net localgroup "Administrators" "ssm-user" /delete
   ```

   -或者-

   要恢复管理访问权限，请运行以下命令。

```
net localgroup "Administrators" "ssm-user" /add
```

使用 Windows 控制台修改管理员权限

1. 连接到托管式节点并打开 PowerShell 或命令提示符窗口。
2. 在命令行中，运行 `lusrmgr.msc` 打开本地用户和组控制台。
3. 打开用户目录，然后打开 ssm-user。
4. 在 Member Of (属于) 选项卡上，执行以下操作之一：

   - 要删除管理员权限，请选择 Administrators，然后选择删除。

     -或者-

     要恢复管理访问权限，请在文本框中输入 **`Administrators`**，然后选择 Add (添加)。
5. 选择 确定。

步骤 8：（可选）通过 Session Manager 允许和控制 SSH 连接的权限

您可以允许 AWS 账户 中的用户使用 AWS Systems Manager Session Manager 通过 AWS Command Line Interface (AWS CLI) 建立与托管式节点的 Secure Shell (SSH) 连接。使用 SSH 连接的用户还可以使用安全复制协议 (SCP) 在本地计算机和托管式节点之间复制文件。您可以使用此功能连接到托管式节点，而无需打开入站端口或维护堡垒主机。

允许 SSH 连接后，您可以使用 AWS Identity and Access Management (IAM) 策略显式允许或拒绝用户、组或角色使用 Session Manager 进行 SSH 连接。

> **Note**
>
> 日志记录不可用于通过端口转发或 SSH 连接的 Session Manager 会话。这是因为 SSH 会加密所有会话数据，而 Session Manager 仅充当 SSH 连接的隧道。

主题

- 允许 Session Manager 的 SSH 连接
- 通过 Session Manager 控制 SSH 连接的用户权限

允许 Session Manager 的 SSH 连接

使用以下步骤在托管式节点上通过 Session Manager 允许 SSH 连接。

允许 Session Manager 的 SSH 连接

1. 在要允许 SSH 连接的托管式节点上，执行以下操作：

   - 确保 SSH 正在托管式节点上运行。（您可以关闭节点上的入站端口。）
   - 确保托管式节点上安装了 SSM Agent 版本 2.3.672.0 或更高版本。

     有关在托管式节点上安装或更新 SSM Agent 的更多信息，请参阅以下主题：

     - 在适用于 Windows Server 的 EC2 实例上手动安装和卸载 SSM Agent.
     - 在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent
     - 在适用于 macOS 的 EC2 实例上手动安装和卸载 SSM Agent
     - 如何在混合 Windows 节点上安装 SSM Agent
     - 如何在混合 Linux 节点上安装 SSM Agent

     > **Note**
     >
     > 要将 Session Manager 与作为托管式节点激活的本地服务器、边缘设备和虚拟机 (VM) 一起使用，必须使用高级实例套餐。有关高级实例的更多信息，请参阅 配置实例套餐。

2. 在要使用 SSH 连接到托管式节点的本地计算机上，执行以下操作：

   - 确保安装了 Session Manager 插件的版本 1.1.23.0 或更高版本。

     有关安装 Session Manager 的信息，请参阅 为 AWS CLI 安装 Session Manager 插件。
   - 更新 SSH 配置文件以允许运行代理命令，此命令启动 Session Manager 会话并通过连接传输所有数据。

     Linux 和 macOS

     > **Tip**
     >
     > SSH 配置文件通常位于 `~/.ssh/config`。

     将以下内容添加到本地计算机上的配置文件中。

```
# SSH over Session Manager
Host i-* mi-*
    ProxyCommand sh -c "aws ssm start-session --target %h --document-name AWS-
StartSSHSession --parameters 'portNumber=%p'"
    User ec2-user
```

Windows

> **Tip**
>
> SSH 配置文件通常位于 C:\Users*<username>*\.ssh\config。

将以下内容添加到本地计算机上的配置文件中。

```
# SSH over Session Manager
Host i-* mi-*
    ProxyCommand C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe "aws
 ssm start-session --target %h --document-name AWS-StartSSHSession --parameters
 portNumber=%p"
```

- 创建或确认您具有在建立与托管式节点的连接时使用的 Privacy Enhanced Mail 证书（PEM 文件），或者至少具有公有密钥。此密钥必须是已与托管式节点关联的密钥。必须设置私有密钥文件的权限，以确保只有您可以读取该文件。您可以使用以下命令设置您的私有密钥文件的权限，以确保只有您可以读取该文件。

  ```
  chmod 400 <my-key-pair>.pem
  ```

  例如，对于 Amazon Elastic Compute Cloud (Amazon EC2) 实例，这是您在创建实例时创建或选择的密钥对文件。（您可以在启动会话的命令中指定证书或密钥的路径。有关使用 SSH 启动会话的信息，请参阅 启动会话 (SSH)。）

通过 Session Manager 控制 SSH 连接的用户权限

在托管式节点上通过 Session Manager 启用 SSH 连接后，您可以使用 IAM policy 允许或拒绝用户、组或角色通过 Session Manager 进行 SSH 连接。

要使用 IAM policy 允许通过 Session Manager 进行 SSH 连接，请执行以下步骤：

- 使用以下选项之一：
  - 选项 1：通过以下网址打开 IAM 控制台：https://console.aws.amazon.com/iam/。

    在导航窗格中，选择 Policies (策略)，然后更新您想要允许通过 Session Manager 启动 SSH 连接的用户或角色的权限策略。

    例如，将以下元素添加到您在Session Manager 的快速入门最终用户策略中创建的快速入门策略中。将每个*******替换为您自己的信息。

    JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ssm:StartSession",
            "Resource": [
                "arn:aws:ec2:us-east-1:111122223333:instance/instance-id",
                "arn:aws:ssm:*:*:document/AWS-StartSSHSession"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "ssmmessages:OpenDataChannel",
            "Resource": "arn:aws:ssm:*:*:session/${aws:userid}-*"
        }
    ]
}
```

  - 选项 2：使用 AWS Management Console、AWS CLI 或 AWS API 将一个内联策略附加到用户策略。

    通过使用所选的方法，将选项 1 中的策略语句附加到 AWS 用户、组或角色的策略中。

    有关信息，请参阅 IAM 用户指南中的添加和删除 IAM 身份权限。

要使用 IAM policy 拒绝通过 Session Manager 进行 SSH 连接，请执行以下步骤：

- 使用以下选项之一：
  - 选项 1：通过以下网址打开 IAM 控制台：https://console.aws.amazon.com/iam/。在导航窗格中，选择 Policies (策略)，然后更新用户或角色的权限策略以阻止启动 Session Manager 会话。

    例如，将以下元素添加到您在Session Manager 的快速入门最终用户策略中创建的快速入门策略中。

    JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "ssm:StartSession",
            "Resource": "arn:aws:ssm:*:*:document/AWS-StartSSHSession"
        },
        {
            "Effect": "Allow",
            "Action": "ssmmessages:OpenDataChannel",
            "Resource": "arn:aws:ssm:*:*:session/${aws:userid}-*"
        }
    ]
}
```

  - 选项 2：使用 AWS Management Console、AWS CLI 或 AWS API 将一个内联策略附加到用户策略。

    通过使用所选的方法，将选项 1 中的策略语句附加到 AWS 用户、组或角色的策略中。

    有关信息，请参阅 IAM 用户指南中的添加和删除 IAM 身份权限。

使用 Session Manager

您可以使用 AWS Systems Manager 控制台、Amazon Elastic Compute Cloud (Amazon EC2) 控制台或 AWS Command Line Interface (AWS CLI) 启动会话，以连接到系统管理员使用 AWS Identity and Access Management (IAM) 策略授予您访问权限的托管式节点。根据您的权限，您还可以查看有关会话的信息、恢复未超时的非活动会话以及结束会话。建立会话后，它不受 IAM 角色会话持续时间的影

响。关于使用 Session Manager 限制会话持续时间的信息，请参阅指定空闲会话超时值和指定最长会话持续时间。

有关会话的更多信息，请参阅 什么是会话？

主题

- 为 AWS CLI 安装 Session Manager 插件
- 启动会话
- 结束会话
- 查看会话历史记录

为 AWS CLI 安装 Session Manager 插件

要使用 AWS Command Line Interface（AWS CLI）启动与托管式节点的 Session Manager 会话，您必须在本地计算机上安装此 Session Manager 插件。您可以在支持的 Microsoft Windows Server、macOS、Linux 和 Ubuntu Server 版本上安装此插件。

> **Note**
>
> 要使用 AWS CLI 插件，必须在本地计算机上安装 Session Manager 版本 1.16.12 或更高版本。有关更多信息，请参阅安装或更新 AWS Command Line Interface 的最新版本。

主题

- Session Manager 插件最新版本和发布历史记录
- 在 Windows 上安装 Session Manager 插件
- 在 macOS 上安装 Session Manager 插件
- 在 Linux 上安装 Session Manager 插件
- 验证 Session Manager 插件安装
- GitHub 上的 Session Manager 插件
- （可选）开启 Session Manager 插件日志记录

Session Manager 插件最新版本和发布历史记录

本地计算机必须运行支持的 Session Manager 插件版本。当前支持的最低版本为 1.1.17.0。如果运行的是更早的版本，则 Session Manager 操作可能会失败。

要查看使用的是不是最新版本，请在 AWS CLI 中运行以下命令。

> **Note**
>
> 仅当插件位于操作系统类型的默认安装目录中时，此命令才返回结果。您也可以在 VERSION 文件（位于安装此插件的目录中）内容中找到版本信息。

```
session-manager-plugin --version
```

下表列出了 Session Manager 插件的所有版本，以及每个版本所含的功能和增强功能。

> **Important**
>
> 建议您始终运行最新版本。最新版本包括增强功能，可改善插件的使用体验。

版本	发行日期	详细信息
1.2.707.0	2025 年 2 月 6 日	增强功能：已将 Dockerfile 中的 Go 版本升级到 1.23。更新了 README 中的版本配置步骤。
1.2.694.0	2024 年 11 月 20 日	错误修复：回滚了向 OpenDataChannel 请求添加凭证的更改。
1.2.688.0	2024 年 11 月 6 日	此版本已于 2024 年 11 月 20 日弃用。 增强功能： • 为 OpenDataChannel 请求添加了凭证。 • 升级了 `testify` 和 `objx` 依赖的软件包。
1.2.677.0	2024 年 10 月 10 日	增强：增加了对通过 OpenDataChannel 请求传递插件版本的支持。
1.2.650.0	2024 年 7 月 2 日	增强：将 aws-sdk-go 升级到 1.54.10。 错误修复：修改了 gofmt 检查的注释。

版本	发行日期	详细信息
1.2.633.0	2024 年 5 月 30 日	增强：将 Dockerfile 更新为使用 Amazon Elastic Container Registry（Amazon ECR）映像。
1.2.553.0	2024 年 1 月 10 日	增强功能：升级了 aws-sdk-go 和相关的 Golang 程序包。
1.2.536.0	2023 年 12 月 4 日	增强：增加了对将 StartSession API 响应作为环境变量传递给会话管理器插件的支持。
1.2.497.0	2023 年 8 月 1 日	增强功能：将 Go SDK 升级到 v1.44.302。
1.2.463.0	2023 年 3 月 15 日	增强功能：在 macOS 捆绑安装程序和已签名的安装程序中增加了对 Apple Mac（M1）的 Mac with Apple silicon 支持。
1.2.398.0	2022 年 10 月 14 日	改进：支持 Golang 版本 1.17。更新 macOS 的默认会话管理器插件运行程序以使用 Python3。更新从 SSMCLI 到会话管理器插件的导入路径。
1.2.339.0	2022 年 6 月 16 日	错误修复：修复端口会话的空闲会话超时问题。
1.2.331.0	2022 年 5 月 27 日	错误修复：修复当本地服务器在超时前无法连接时提前关闭端口会话的问题。
1.2.323.0	2022 年 5 月 19 日	错误修复：禁用 smux 保持活动状态以使用空闲会话超时功能。
1.2.312.0	2022 年 3 月 31 日	增强功能：支持更多的输出消息有效负载类型。
1.2.295.0	2022 年 1 月 12 日	错误修复：客户端在代理变为非活动状态时重新发送流数据而导致的会话挂起，以及 `start_publication` 和 `pause_publication` 消息的不正确日志。
1.2.279.0	2021 年 10 月 27 日	增强功能：面向 Windows 平台打包成 zip 格式。
1.2.245.0	2021 年 8 月 19 日	增强功能：将 `aws-sdk-go` 升级到最新版本（v1.40.17）以支持 AWS IAM Identity Center。
1.2.234.0	2021 年 7 月 26 日	错误修复：处理交互式会话类型中会话突然终止的情况。

版本	发行日期	详细信息
1.2.205.0	2021 年 6 月 10 日	增强功能：添加对已签名 macOS 安装程序的支持。
1.2.54.0	2021 年 1 月 29 日	增强功能：添加对在 NonInteractiveCommands 执行模式中运行会话的支持。
1.2.30.0	2020 年 11 月 24 日	增强功能：（仅限端口转发会话）整体性能提升。
1.2.7.0	2020 年 10 月 15 日	增强功能：（仅限端口转发会话）延迟减少并且整体性能提升。
1.1.61.0	2020 年 4 月 17 日	增强功能：添加对 Linux 和 Ubuntu 的 ARM 支持。
1.1.54.0	2020 年 1 月 6 日	错误修复：处理 Session Manager 插件未准备就绪时丢弃数据包的争用情况。
1.1.50.0	2019 年 11 月 19 日	增强功能：添加了将端口转发到本地 unix 套接字的支持。
1.1.35.0	2019 年 11 月 7 日	增强功能：（仅限端口转发会话）在本地用户按 `Ctrl+C` 时将 TerminateSession 命令发送到 SSM Agent。
1.1.33.0	2019 年 9 月 26 日	增强功能：（仅限端口转发会话）当客户端断开 TCP 连接时，向服务器发送断开连接信号。
1.1.31.0	2019 年 9 月 6 日	增强功能：更新以保持端口转发会话打开，直到远程服务器关闭连接。
1.1.26.0	2019 年 7 月 30 日	增强功能：更新以限制会话期间的数据传输速率。
1.1.23.0	2019 年 7 月 9 日	增强功能：添加对使用 Session Manager 运行 SSH 会话的支持。
1.1.17.0	2019 年 4 月 4 日	增强功能：添加了使用 AWS Key Management Service (AWS KMS) 进一步加密会话数据的支持。
1.0.37.0	2018 年 9 月 20 日	增强功能：Windows 版本错误修复。

版本	发行日期	详细信息
1.0.0.0	2018 年 9 月 11 日	发布 Session Manager 插件的初始版本。

在 Windows 上安装 Session Manager 插件

您可以使用独立安装程序在 Windows Vista 上安装 Session Manager 插件。

更新发布后，您必须重复安装过程以获取最新版本的 Session Manager 插件。

> **Note**
>
> 请注意以下信息。
>
> - Session Manager 插件安装程序需要管理员权限才能安装插件。
> - 为获得最佳效果，建议使用 Windows PowerShell 版本 5 或更高版本在 Windows 客户端上启动会话。或者，您可以在 Windows 10 中使用 Shell 命令。Session Manager 插件只支持 PowerShell 和 Shell 命令。第三方命令行工具可能与此插件不兼容。

要使用 EXE 安装程序安装 Session Manager 插件，请执行以下步骤：

1. 使用以下 URL 下载安装程序。

   ```
   https://s3.amazonaws.com/session-manager-downloads/plugin/latest/windows/
   SessionManagerPluginSetup.exe
   ```

   或者，您可以访问以下 URL 以下载安装程序的 zip 格式版本。

   ```
   https://s3.amazonaws.com/session-manager-downloads/plugin/latest/windows/
   SessionManagerPlugin.zip
   ```

2. 运行下载的安装程序并按照屏幕上的说明操作。如果您下载了安装程序的 zip 格式版本，则必须先解压缩安装程序。

   将安装位置框留空以将插件安装到默认目录。

   - `%PROGRAMFILES%\Amazon\SessionManagerPlugin\bin\`

3. 验证安装是否成功。有关信息，请参阅验证 Session Manager 插件安装。

> **Note**
>
> 如果 Windows 无法找到可执行文件，您需要重新打开命令提示符或手动将安装目录添加到 PATH 环境变量。有关信息，请参阅故障排除主题 Session Manager 插件未自动添加到命令行路径（Windows）。

在 macOS 上安装 Session Manager 插件

选择以下主题之一，在 macOS 上安装 Session Manager 插件。

> **Note**
>
> 签名的安装程序是一个签名的 .pkg 文件。捆绑的安装程序使用 .zip 文件。文件解压缩后，您可以使用二进制文件安装插件。

使用已签名的安装程序在 macOS 上安装 Session Manager 插件

本节介绍如何使用已签名的安装程序在 macOS 上安装 Session Manager 插件。

要使用已签名的安装程序安装 Session Manager 插件 (macOS)，请执行以下步骤：

1. 下载已签名的安装程序。

   x86_64

   ```
   curl "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/mac/
   session-manager-plugin.pkg" -o "session-manager-plugin.pkg"
   ```

   搭载 Apple 硅芯片的 Mac

   ```
   curl "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/
   mac_arm64/session-manager-plugin.pkg" -o "session-manager-plugin.pkg"
   ```

2. 运行安装命令。如果命令失败，确认 /usr/local/bin 文件夹是否存在。如果不存在，创建后再次运该此命令。

   ```
   sudo installer -pkg session-manager-plugin.pkg -target /
   ```

```
sudo ln -s /usr/local/sessionmanagerplugin/bin/session-manager-plugin /usr/local/
bin/session-manager-plugin
```

3. 验证安装是否成功。有关信息，请参阅验证 Session Manager 插件安装。

在 macOS 上安装 Session Manager 插件

本节介绍如何使用捆绑安装程序在 macOS 上安装 Session Manager 插件。

> **Important**
>
> 请注意以下重要信息。
>
> - 默认情况下，安装程序需要 sudo 访问权限才能运行，因为脚本会将插件安装到 `/usr/local/sessionmanagerplugin` 系统目录中。如果不想使用 sudo 安装插件，则手动更新安装程序脚本，将插件安装到不需要 sudo 访问权限的目录中。
> - 捆绑安装程序不支持安装到包含空格的路径。

要使用捆绑安装程序安装 Session Manager 插件 (macOS)，请执行以下步骤：

1. 下载捆绑安装程序。

   x86_64

   ```
   curl "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/mac/
   sessionmanager-bundle.zip" -o "sessionmanager-bundle.zip"
   ```

   搭载 Apple 硅芯片的 Mac

   ```
   curl "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/
   mac_arm64/sessionmanager-bundle.zip" -o "sessionmanager-bundle.zip"
   ```

2. 解压缩程序包。

   ```
   unzip sessionmanager-bundle.zip
   ```

3. 运行安装命令。

```
sudo ./sessionmanager-bundle/install -i /usr/local/sessionmanagerplugin -b /usr/
local/bin/session-manager-plugin
```

> **Note**
>
> 此插件需要 Python 2.6.5 或更高版本或者 Python 3.3 或更高版本。默认情况下，安装脚本在系统默认版本的 Python 下运行。如果已安装 Python 的可选版本并希望使用该版本安装 Session Manager 插件，请使用该版本按 Python 可执行文件的绝对路径运行安装脚本。示例如下：
>
> ```
> sudo /usr/local/bin/python3.8 sessionmanager-bundle/install -i /usr/local/
> sessionmanagerplugin -b /usr/local/bin/session-manager-plugin
> ```

安装程序在 `/usr/local/sessionmanagerplugin` 中安装 Session Manager 插件，并在 `/usr/local/bin` 目录中创建符号链接 `session-manager-plugin`。这样，不必在用户的 `$PATH` 变量中指定安装目录。

要查看 `-i` 和 `-b` 选项的说明，请使用 `-h` 选项。

```
./sessionmanager-bundle/install -h
```

4. 验证安装是否成功。有关信息，请参阅[验证 Session Manager 插件安装](#)。

> **Note**
>
> 如果需要卸载此插件，请按照所示顺序运行以下两个命令。
>
> ```
> sudo rm -rf /usr/local/sessionmanagerplugin
> ```
>
> ```
> sudo rm /usr/local/bin/session-manager-plugin
> ```

在 Linux 上安装 Session Manager 插件

本节包含有关验证Session Manager插件安装程序包的签名，以及在以下 Linux 发行版上安装插件的信息：

- Amazon Linux 2
- AL2023
- RHEL
- Debian
- Ubuntu

主题

- 验证 Session Manager 插件的签名
- 在 Amazon Linux 2、Amazon Linux 2023 和 Red Hat Enterprise Linux 发行版上安装 Session Manager 插件
- 在 Debian Server 和 Ubuntu Server 上安装 Session Manager 插件

验证 Session Manager 插件的签名

适用于 Linux 实例的 Session Manager 插件 RPM 和 Debian 安装包是以加密方式签名的。您可以使用公有密钥验证插件二进制文件和软件包是否为未修改的原始包。如果文件有任何更改或损坏，则验证将失败。您可以使用 GNU Privacy Guard（GPG）工具验证安装包的签名。以下信息适用于 Session Manager 插件版本 1.2.707.0 或更高版本。

完成以下步骤以验证 Session Manager 插件安装包的签名。

主题

- 步骤 1：下载 Session Manager 插件安装包
- 步骤 2：下载关联的签名文件
- 步骤 3：安装 GPG 工具
- 步骤 4：验证 Linux 服务器上的 Session Manager 插件安装包

步骤 1：下载 Session Manager 插件安装包

下载您要验证的 Session Manager 插件安装包。

Amazon Linux 2、AL2023 和 RHEL RPM 软件包

x86_64

```
curl -o "session-manager-plugin.rpm" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/linux_64bit/session-manager-plugin.rpm"
```

x86

```
curl -o "session-manager-plugin.rpm" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/linux_32bit/session-manager-plugin.rpm"
```

ARM64

```
curl -o "session-manager-plugin.rpm" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/linux_arm64/session-manager-plugin.rpm"
```

Debian 和 Ubuntu Deb 软件包

x86_64

```
curl -o "session-manager-plugin.deb" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/ubuntu_64bit/session-manager-plugin.deb"
```

x86

```
curl -o "session-manager-plugin.deb" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/ubuntu_32bit/session-manager-plugin.deb"
```

ARM64

```
curl -o "session-manager-plugin.deb" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/ubuntu_arm64/session-manager-plugin.deb"
```

步骤 2：下载关联的签名文件

下载安装包后，请下载相关的签名文件以进行软件包验证。为了提供额外的保护，以防止未经授权的复制或使用软件包内的 session-manager-plugin 二进制文件，我们还提供了二进制签名，您可以使用该签名来验证单个二进制文件。您可以根据自己的安全需求来选择使用这些二进制签名。

Amazon Linux 2、AL2023 和 RHEL 签名软件包

x86_64

软件包：

```
curl -o "session-manager-plugin.rpm.sig" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/linux_64bit/session-manager-plugin.rpm.sig"
```

二进制文件：

```
curl -o "session-manager-plugin.sig" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/linux_64bit/session-manager-plugin.sig"
```

x86

软件包：

```
curl -o "session-manager-plugin.rpm.sig" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/linux_32bit/session-manager-plugin.rpm.sig"
```

二进制文件：

```
curl -o "session-manager-plugin.sig" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/linux_32bit/session-manager-plugin.sig"
```

ARM64

软件包：

```
curl -o "session-manager-plugin.rpm.sig" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/linux_arm64/session-manager-plugin.rpm.sig"
```

二进制文件：

```
curl -o "session-manager-plugin.sig" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/linux_arm64/session-manager-plugin.sig"
```

Debian 和 Ubuntu Deb 签名软件包

x86_64

软件包：

```
curl -o "session-manager-plugin.deb.sig" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/ubuntu_64bit/session-manager-plugin.deb.sig"
```

二进制文件：

```
curl -o "session-manager-plugin.sig" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/ubuntu_64bit/session-manager-plugin.sig"
```

x86

软件包：

```
curl -o "session-manager-plugin.deb.sig" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/ubuntu_32bit/session-manager-plugin.deb.sig"
```

二进制文件：

```
curl -o "session-manager-plugin.sig" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/ubuntu_32bit/session-manager-plugin.sig"
```

ARM64

软件包：

```
curl -o "session-manager-plugin.deb.sig" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/ubuntu_arm64/session-manager-plugin.deb.sig"
```

二进制文件：

```
curl -o "session-manager-plugin.sig" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/ubuntu_arm64/session-manager-plugin.sig"
```

步骤 3：安装 GPG 工具

要验证 Session Manager 插件的签名，您必须在系统上安装 GNU 隐私保护（GPG）工具。此验证过程需要使用 GPG 版本 2.1 或更高版本。可以运行以下命令查看 GPG 版本：

```
gpg --version
```

如果您的 GPG 版本低于 2.1，则请在继续验证过程之前进行版本更新。对于大多数系统，您可以使用软件包管理程序更新 GPG 工具。例如，在 Amazon Linux 和 RHEL 系统上，您可以使用以下命令：

```
sudo yum update
sudo yum install gnupg2
```

在 Ubuntu 或 Debian 系统上，您可以使用以下命令：

```
sudo apt-get update
sudo apt-get install gnupg2
```

在继续验证过程之前，请确保您具有所需的 GPG 版本。

步骤 4：验证 Linux 服务器上的 Session Manager 插件安装包

请按照以下程序验证 Linux 服务器上的 Session Manager 插件安装包。

> **Note**
>
> Amazon Linux 2 不支持 gpg 工具版本 2.1 或更高版本。如果以下程序不适用于您的 Amazon Linux 2 实例，请在将签名安装到您的 Amazon Linux 2 实例上之前，在其他平台上验证签名。

1. 复制以下公有密钥并将其保存到名为 session-manager-plugin.gpg 的文件。

```
-----BEGIN PGP PUBLIC KEY BLOCK-----

mFIEZ5ERQxMIKoZIzj0DAQcCAwQjuZy+IjFoYg57sLTGhF3aZLBaGpzB+gY6j7Ix
P7NqbpXyjVj8a+dy79gSd64OEaMxUb7vw/jug+CfRXwVGRMNtIBBV1MgU1NNIFNl
c3Npb24gTWFuYWdlciA8c2Vzc2lvbi1tYW5hZ2VyLXBsdWdpbi1zaWduZXJAYW1h
em9uLmNvbT4gKEFXUyBTeXN0ZW1zIE1hbmFnZXIgU2Vzc2lvbiBNYW5hZ2VyIFBs
dWdpbiBMaW51eCBTaWduZXIgS2V5KYkBAAQQEwgAqAUCZ5ERQ4EcQVdTIFNTTSBT
ZXNzaW9uIE1hbmFnZXIgPHNlc3Npb24tbWFuYWdlci1wbHVnaW4tc2lnbmVyQGFt
YXpvbi5jb20+IChBV1MgU3lzdGVtcyBNYW5hZ2VyIFNlc3Npb24gTWFuYWdlciBQ
bHVnaW4gTGludXggU2lnbmVyIEtleSkWIQR5WWNxJM4JOtUB1HosTUr/b2dX7gIe
AwIbAwIVCAAKCRAsTUr/b2dX7rO1AQCa1kig3lQ78W/QHGU76uHx3XAyv0tfpE9U
oQBCIwFLSgEA3PDHt3lZ+s6m9JLGJsy+Cp5ZFzpiF6RgluR/2gA861M=
```

```
=2DQm
-----END PGP PUBLIC KEY BLOCK-----
```

2. 将公有密钥导入到您的密钥环中。返回的密钥值应为 2C4D4AFF6F6757EE。

```
$ gpg --import session-manager-plugin.gpg
gpg: key 2C4D4AFF6F6757EE: public key "AWS SSM Session Manager <session-manager-
plugin-signer@amazon.com> (AWS Systems Manager Session Manager Plugin Linux Signer
 Key)" imported
gpg: Total number processed: 1
gpg:               imported: 1
```

3. 运行以下命令验证指纹。

```
gpg --fingerprint 2C4D4AFF6F6757EE
```

命令输出的指纹应与以下内容匹配。

```
7959 6371 24CE 093A D501 D47A 2C4D 4AFF 6F67 57EE
```

```
pub   nistp256 2025-01-22 [SC]
      7959 6371 24CE 093A D501  D47A 2C4D 4AFF 6F67 57EE
uid           [ unknown] AWS SSM Session Manager <session-manager-plugin-
signer@amazon.com> (AWS Systems Manager Session Manager Plugin Linux Signer Key)
```

如果指纹不匹配，则请勿安装插件。联系 AWS 支持。

4. 验证安装程序包签名。将 *signature-filename* 和 *downloaded-plugin-filename* 替换为下载签名文件和 session-manager-plugin 时指定的值，如本主题前面的表中所列。

```
gpg --verify signature-filename downloaded-plugin-filename
```

例如，对于 Amazon Linux 2 上的 x86_64 架构，命令如下所示：

```
gpg --verify session-manager-plugin.rpm.sig session-manager-plugin.rpm
```

该命令会返回类似以下内容的输出。

```
gpg: Signature made Mon Feb 3 20:08:32 2025 UTC gpg: using ECDSA key
 2C4D4AFF6F6757EE
```

```
gpg: Good signature from "AWS Systems Manager Session Manager <session-manager-
plugin-signer@amazon.com> (AWS Systems Manager Session Manager Plugin Linux Signer
 Key)" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg: There is no indication that the signature belongs to the owner.
Primary key fingerprint: 7959 6371 24CE 093A D501 D47A 2C4D 4AFF 6F67 57EE
```

如果输出包含短语 `BAD signature`，则检查是否正确执行了此过程。如果您继续获得该响应，则请联系 AWS 支持 且不要安装该软件包。有关信任的警告消息并不意味着签名无效，只是您尚未验证公有密钥而已。只有当您或您信任的某个人对密钥进行了签名，密钥才是可信的。如果输出包含短语 `Can't check signature: No public key`，则验证您下载的 Session Manager 插件是版本 1.2.707.0 或更高版本。

在 Amazon Linux 2、Amazon Linux 2023 和 Red Hat Enterprise Linux 发行版上安装 Session Manager 插件

按照以下步骤在 Amazon Linux 2、Amazon Linux 2023 (AL2023) 和 RHEL 发行版上安装 Session Manager 插件

> **Note**
>
> Amazon Linux 1 不支持 Session Manager 插件。Amazon Linux 2 及更高版本支持此插件。

1. 下载和安装 Session Manager 插件 RPM 软件包。

   x86_64

   在 Amazon Linux 2 和 RHEL 7 上，运行以下命令：

   ```
   sudo yum install -y https://s3.amazonaws.com/session-manager-downloads/plugin/
   latest/linux_64bit/session-manager-plugin.rpm
   ```

   在 AL2023 以及 RHEL 8 和 9 上，运行以下命令：

   ```
   sudo dnf install -y https://s3.amazonaws.com/session-manager-downloads/plugin/
   latest/linux_64bit/session-manager-plugin.rpm
   ```

x86

在 RHEL 7 上，运行以下命令：

```
sudo yum install -y https://s3.amazonaws.com/session-manager-downloads/plugin/
latest/linux_32bit/session-manager-plugin.rpm
```

在 RHEL 8 和 9 上，运行以下命令：

```
sudo dnf install -y https://s3.amazonaws.com/session-manager-downloads/plugin/
latest/linux_32bit/session-manager-plugin.rpm
```

ARM64

在 Amazon Linux 2 和 RHEL 7 上，运行以下命令：

```
sudo yum install -y https://s3.amazonaws.com/session-manager-downloads/plugin/
latest/linux_arm64/session-manager-plugin.rpm
```

在 AL2023 以及 RHEL 8 和 9 上，运行以下命令：

```
sudo dnf install -y https://s3.amazonaws.com/session-manager-downloads/plugin/
latest/linux_arm64/session-manager-plugin.rpm
```

2. 验证安装是否成功。有关信息，请参阅验证 Session Manager 插件安装。

> **Note**
>
> 如果需要卸载此插件，请运行 `sudo yum erase session-manager-plugin -y`

在 Debian Server 和 Ubuntu Server 上安装 Session Manager 插件

1. 下载 Session Manager 插件 deb 软件包。

x86_64

```
curl "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/
ubuntu_64bit/session-manager-plugin.deb" -o "session-manager-plugin.deb"
```

x86

```
curl "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/
ubuntu_32bit/session-manager-plugin.deb" -o "session-manager-plugin.deb"
```

ARM64

```
curl "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/
ubuntu_arm64/session-manager-plugin.deb" -o "session-manager-plugin.deb"
```

2. 运行安装命令。

```
sudo dpkg -i session-manager-plugin.deb
```

3. 验证安装是否成功。有关更多信息，请参阅 验证 Session Manager 插件安装。

> **Note**
>
> 如果需要卸载此插件，请运行 `sudo dpkg -r session-manager-plugin`

验证 Session Manager 插件安装

运行以下命令验证是否已成功安装 Session Manager 插件。

```
session-manager-plugin
```

如果安装成功，将返回以下消息。

```
The Session Manager plugin is installed successfully. Use the AWS CLI to start a
 session.
```

您还可以通过在 AWS Command Line Interface（AWS CLI）中运行 start-session 命令来测试安装。在下面的命令中，将 *instance-id* 替换为您自己的信息。

```
aws ssm start-session --target instance-id
```

仅当您已安装并配置了 AWS CLI，而且 Session Manager 管理员已授予您使用 Session Manager 访问目标托管式节点所需的 IAM 权限时，此命令才有效。

GitHub 上的 Session Manager 插件

GitHub 上提供 Session Manager 插件的源代码，便于您根据需要调整插件。我们鼓励您针对要包含的更改提交提取请求。但是，Amazon Web Services 不支持运行此软件的已修改副本。

（可选）开启 Session Manager 插件日志记录

Session Manager 插件包括一个允许对运行的会话进行日志记录的选项。默认情况下，日志记录处于关闭状态。

如果允许日志记录，则 Session Manager 插件会在本地计算机上为应用程序活动 (`session-manager-plugin.log`) 和错误 (`errors.log`) 创建日志文件。

主题

- 开启 Session Manager 插件的日志记录（Windows）
- 启用 Session Manager 插件的日志记录（Linux 和 macOS）

开启 Session Manager 插件的日志记录（Windows）

1. 找到插件的 `seelog.xml.template` 文件。

   默认位置是 `C:\Program Files\Amazon\SessionManagerPlugin\seelog.xml.template`。
2. 将文件名更改为 `seelog.xml`。
3. 打开文件，然后将 `minlevel="off"` 更改为 `minlevel="info"` 或 `minlevel="debug"`。

   > **Note**
   >
   > 默认情况下，有关打开数据通道和重新连接会话的日志条目在 INFO (信息) 级别记录。数据流（数据包和确认）条目在 DEBUG (调试) 级别记录。

4. 更改要修改的其他配置选项。可以更改的选项包括：

   - 调试级别：您可以将调试级别从 `formatid="fmtinfo"` 更改为 `formatid="fmtdebug"`。

- 日志文件选项：您可以更改日志文件选项，包括日志的存储位置，但日志文件名除外。

> **Important**
>
> 不要更改文件名，否则日志记录无法正常工作。

```
<rollingfile type="size" filename="C:\Program Files\Amazon\SessionManagerPlugin
\Logs\session-manager-plugin.log" maxsize="30000000" maxrolls="5"/>
<filter levels="error,critical" formatid="fmterror">
<rollingfile type="size" filename="C:\Program Files\Amazon\SessionManagerPlugin
\Logs\errors.log" maxsize="10000000" maxrolls="5"/>
```

5. 保存该文件。

启用 Session Manager 插件的日志记录（Linux 和 macOS）

1. 找到插件的 `seelog.xml.template` 文件。

   默认位置是 `/usr/local/sessionmanagerplugin/seelog.xml.template`。

2. 将文件名更改为 `seelog.xml`。
3. 打开文件，然后将 `minlevel="off"` 更改为 `minlevel="info"` 或 `minlevel="debug"`。

> **Note**
>
> 默认情况下，有关打开数据通道和重新连接会话的日志条目在 INFO (信息) 级别记录。数据流（数据包和确认）条目在 DEBUG (调试) 级别记录。

4. 更改要修改的其他配置选项。可以更改的选项包括：

   - 调试级别：您可以将调试级别从 `formatid="fmtinfo"` 更改为 `outputs formatid="fmtdebug"`。
   - 日志文件选项：您可以更改日志文件选项，包括日志的存储位置，但日志文件名除外。

> **Important**
>
> 不要更改文件名，否则日志记录无法正常工作。

```
<rollingfile type="size" filename="/usr/local/sessionmanagerplugin/logs/session-
manager-plugin.log" maxsize="30000000" maxrolls="5"/>
<filter levels="error,critical" formatid="fmterror">
<rollingfile type="size" filename="/usr/local/sessionmanagerplugin/logs/
errors.log" maxsize="10000000" maxrolls="5"/>
```

> **Important**
>
> 如果使用指定的默认目录存储日志，则必须使用 sudo 运行会话命令，或者给安装插件的目录指定完全读写权限。要绕过这些限制，请更改存储日志的位置。

5. 保存该文件。

启动会话

您可以使用 AWS Systems Manager 控制台、Amazon Elastic Compute Cloud (Amazon EC2) 控制台、AWS Command Line Interface (AWS CLI) 或 SSH 启动会话。

主题

- 启动会话（Systems Manager 控制台）
- 开启会话（Amazon EC2 控制台）
- 启动会话 (AWS CLI)
- 启动会话 (SSH)
- 启动会话（端口转发）
- 启动会话（至远程主机的端口转发）
- 启动会话（交互式和非交互式命令）

启动会话（Systems Manager 控制台）

您可以使用 AWS Systems Manager 控制台启动与账户中的托管式节点进行的会话。

> **Note**
>
> 在启动会话之前，确保您已经完成 Session Manager 的设置步骤。有关信息，请参阅设置 Session Manager。

启动会话（Systems Manager 控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Session Manager。
3. 选择 Start session (启动会话)。
4. （可选）在会话原因字段中输入会话描述。
5. 对于目标实例，选择要连接的托管式节点左侧的选项按钮。

   如果您需要节点不在列表中，或者如果您选择了节点并收到配置错误，请参阅 托管式节点不可用或未为 Session Manager 配置托管式节点，以了解故障排除步骤。
6. 选择启动会话立即启动会话。

   –或者–

   选择下一步查看会话选项。
7. （可选）对于会话文档，选择要在会话启动时运行的文档。如果您的文档支持运行时参数，则可以在每个参数字段中输入一个或多个逗号分隔的值。
8. 选择下一步。
9. 选择 Start session (启动会话)。

建立连接后，您可以像运行任何其他连接类型一样运行 Bash 命令（Linux 和 macOS）或 PowerShell 命令（Windows）。

> **Important**
>
> 如果要允许用户在会话管理器控制台中启动会话时指定文档，请注意以下几点：
>
> - 您必须在用户的 IAM policy 中向其授予 `ssm:GetDocument` 和 `ssm:ListDocuments` 权限。有关更多信息，请参阅 在控制台中授予对自定义会话文档的访问权限。
> - 控制台仅支持将 `sessionType` 定义为 `Standard_Stream` 的会话文档。有关更多信息，请参阅 会话文档架构。

开启会话（Amazon EC2 控制台）

您可以使用 Amazon Elastic Compute Cloud (Amazon EC2) 控制台来启动与账户中的实例的会话。

> **Note**
>
> 如果您收到一个错误，提示您无权执行一项或多项 Systems Manager 操作 (`ssm:command-name`)，则必须联系您的管理员寻求帮助。管理员是向您提供登录凭证的人。让其帮助您更新您的策略，以允许您从 Amazon EC2 控制台启动会话。如果您是管理员，请参阅 Session Manager 的 IAM policy 示例 了解更多信息。

要启动会话（Amazon EC2 控制台），请执行以下步骤：

1. 通过以下网址打开 Amazon EC2 控制台：https://console.aws.amazon.com/ec2/。
2. 在导航窗格中，选择实例。
3. 选择实例，然后选择连接。
4. 对于 Connection method (连接方法)，选择 Session Manager。
5. 选择连接。

建立连接后，您可以像运行任何其他连接类型一样运行 Bash 命令（Linux 和 macOS）或 PowerShell 命令（Windows）。

启动会话 (AWS CLI)

安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

有关信息，请参阅安装或更新 AWS CLI 的最新版本。

在启动会话之前，确保您已经完成 Session Manager 的设置步骤。有关信息，请参阅设置 Session Manager。

要使用 AWS CLI 运行会话命令，还必须在本地计算机上安装 Session Manager 插件。有关信息，请参阅为 AWS CLI 安装 Session Manager 插件。

要使用 AWS CLI 开启会话，请运行以下命令，将 *instance-id* 替换为您自己的信息。

```
aws ssm start-session \
    --target instance-id
```

有关可以与 start-session 命令结合使用的其他选项的信息，请参阅《AWS CLI Command Reference》中 AWS Systems Manager 部分中的 start-session。

启动会话 (SSH)

要启动 Session Manager SSH 会话，托管式节点上必须安装 2.3.672.0 版本或更高版本的 SSM Agent。

SSH 连接要求

对于使用 SSH 的会话连接，请注意以下要求和限制：

- 必须将目标托管式节点配置为支持 SSH 连接。有关更多信息，请参阅（可选）通过 Session Manager 允许和控制 SSH 连接的权限。
- 您必须使用与 Privacy Enhanced Mail (PEM) 证书关联的托管式节点账户进行连接，而不是用于其他类型的会话连接的 `ssm-user` 账户。例如，在适用于 Linux 和 macOS 的 EC2 实例上，默认用户为 `ec2-user`。有关确定每种实例类型的默认用户的信息，请参阅《Amazon EC2 用户指南》中的获取有关您的实例的信息。
- 日志记录不可用于通过端口转发或 SSH 连接的 Session Manager 会话。这是因为 SSH 会加密所有会话数据，而 Session Manager 仅充当 SSH 连接的隧道。

> **Note**
>
> 在启动会话之前，确保您已经完成 Session Manager 的设置步骤。有关信息，请参阅设置 Session Manager。

要使用 SSH 启动会话，请运行以下命令。将每个*#######*替换为您自己的信息。

```
ssh -i /path/my-key-pair.pem username@instance-id
```

> **Tip**
>
> 当您使用 SSH 启动会话时，您可以使用以下命令格式将本地文件复制到目标托管式节点。
>
> ```
> scp -i /path/my-key-pair.pem /path/ExampleFile.txt username@instance-id:~
> ```

有关可以与 start-session 命令结合使用的其他选项的信息，请参阅《AWS CLI Command Reference》中 AWS Systems Manager 部分中的 start-session。

启动会话（端口转发）

要启动 Session Manager 端口转发会话，必须在托管式节点上安装 2.3.672.0 版本或更高版本的 SSM Agent。

> **Note**
>
> 在启动会话之前，确保您已经完成 Session Manager 的设置步骤。有关信息，请参阅设置 Session Manager。
> 要使用 AWS CLI 运行会话命令，必须在本地计算机上安装 Session Manager 插件。有关信息，请参阅为 AWS CLI 安装 Session Manager 插件。
> 根据您的操作系统和命令行工具，放置引号的位置可能会有所不同，并可能需要转义字符。

要启动端口转发会话，请从 CLI 中运行以下命令。将每个*#######*替换为您自己的信息。

Linux & macOS

```
aws ssm start-session \
    --target instance-id \
    --document-name AWS-StartPortForwardingSession \
    --parameters '{"portNumber":["80"], "localPortNumber":["56789"]}'
```

Windows

```
aws ssm start-session ^
    --target instance-id ^
    --document-name AWS-StartPortForwardingSession ^
    --parameters portNumber="3389",localPortNumber="56789"
```

`portNumber` 是托管式节点上您希望将会话流量重定向到的远程端口。例如，您可以指定端口 3389，用于使用远程桌面协议（RDP）连接到 Windows 节点。如果您未指定 `portNumber` 参数，Session Manager 会将 `80` 用作默认值。

`localPortNumber` 是本地计算机上流量启动的端口，例如 56789。此值是您在使用客户端连接到托管式节点时输入的值。例如 **`localhost:56789`**。

有关可以与 start-session 命令结合使用的其他选项的信息，请参阅《AWS CLI Command Reference》中 AWS Systems Manager 部分中的 start-session。

有关端口转发会话的更多信息，请参阅 AWS 新闻博客中的 Port Forwarding Using AWS Systems ManagerSession Manager。

启动会话（至远程主机的端口转发）

要启动至远程主机的 Session Manager 端口转发会话，必须在托管式节点上安装 3.1.1374.0 版本或更高版本的 SSM Agent。远程主机不需要由 Systems Manager 进行管理。

> **Note**
>
> 在启动会话之前，确保您已经完成 Session Manager 的设置步骤。有关信息，请参阅设置 Session Manager。
> 要使用 AWS CLI 运行会话命令，必须在本地计算机上安装 Session Manager 插件。有关信息，请参阅为 AWS CLI 安装 Session Manager 插件。
> 根据您的操作系统和命令行工具，放置引号的位置可能会有所不同，并可能需要转义字符。

要启动端口转发会话，请从 AWS CLI 中运行以下命令。将每个*#######*替换为您自己的信息。

Linux & macOS

```
aws ssm start-session \
    --target instance-id \
    --document-name AWS-StartPortForwardingSessionToRemoteHost \
    --parameters '{"host":["mydb.example.us-east-2.rds.amazonaws.com"],"portNumber":
["3306"], "localPortNumber":["3306"]}'
```

Windows

```
aws ssm start-session ^
    --target instance-id ^
    --document-name AWS-StartPortForwardingSessionToRemoteHost ^
    --parameters host="mydb.example.us-
east-2.rds.amazonaws.com",portNumber="3306",localPortNumber="3306"
```

`host` 值表示要连接到的远程主机的主机名或 IP 地址。托管式节点和远程主机之间的常规连接和名称解析要求仍然适用。

portNumber 是托管式节点上您希望将会话流量重定向到的远程端口。例如，您可以指定端口 3389，用于使用远程桌面协议（RDP）连接到 Windows 节点。如果您未指定 portNumber 参数，Session Manager 会将 80 用作默认值。

localPortNumber 是本地计算机上流量启动的端口，例如 56789。此值是您在使用客户端连接到托管式节点时输入的值。例如 **localhost:56789**。

有关可以与 start-session 命令结合使用的其他选项的信息，请参阅《AWS CLI Command Reference》中 AWS Systems Manager 部分中的 start-session。

通过 Amazon ECS 任务启动会话

Session Manager 支持通过 Amazon Elastic Container Service（Amazon ECS）集群中的任务启动端口转发会话。为此，必须更新 IAM 中的任务角色来包含以下权限：

JSON

```
{
   "Version": "2012-10-17",
   "Statement": [
       {
       "Effect": "Allow",
       "Action": [
            "ssmmessages:CreateControlChannel",
            "ssmmessages:CreateDataChannel",
            "ssmmessages:OpenControlChannel",
            "ssmmessages:OpenDataChannel"
       ],
      "Resource": "*"
      }
   ]
}
```

要通过 Amazon ECS 任务启动端口转发会话，请从 AWS CLI 中运行以下命令。将每个#######替换为您自己的信息。

> **Note**
>
> 从 target 参数中移除 < 和 > 符号。这些符号仅供读者澄清之用。

Linux & macOS

```
aws ssm start-session \
    --target ecs:<ECS_cluster_name>_<ECS_container_ID>_<container_runtime_ID> \
    --document-name AWS-StartPortForwardingSessionToRemoteHost \
    --parameters '{"host":["URL"],"portNumber":["port_number"], "localPortNumber":
["port_number"]}'
```

Windows

```
aws ssm start-session ^
    --target ecs:<ECS_cluster_name>_<ECS_container_ID>_<container_runtime_ID> ^
    --document-name AWS-StartPortForwardingSessionToRemoteHost ^
    --parameters host="URL",portNumber="port_number",localPortNumber="port_number"
```

启动会话（交互式和非交互式命令）

在启动会话之前，确保您已经完成 Session Manager 的设置步骤。有关信息，请参阅设置 Session Manager。

要使用 AWS CLI 运行会话命令，还必须在本地计算机上安装 Session Manager 插件。有关信息，请参阅为 AWS CLI 安装 Session Manager 插件。

要启动交互式命令会话，请运行以下命令：将每个*******替换为您自己的信息。

Linux & macOS

```
aws ssm start-session \
    --target instance-id \
    --document-name CustomCommandSessionDocument \
    --parameters '{"logpath":["/var/log/amazon/ssm/amazon-ssm-agent.log"]}'
```

Windows

```
aws ssm start-session ^
    --target instance-id ^
    --document-name CustomCommandSessionDocument ^
    --parameters logpath="/var/log/amazon/ssm/amazon-ssm-agent.log"
```

有关可以与 start-session 命令结合使用的其他选项的信息，请参阅《AWS CLI Command Reference》中 AWS Systems Manager 部分中的 start-session。

更多信息

- 在 AWS Systems Manager Session Manager 中使用端口转发连接到远程主机
- 通过 AWS Systems Manager 实现 Amazon EC2 实例端口转发
- 通过 Session Manager 端口转发管理 AWS 托管的 Microsoft AD 资源
- AWS 新闻博客上的 Port Forwarding Using AWS Systems ManagerSession Manager。

结束会话

您可以使用 AWS Systems Manager 控制台或 AWS Command Line Interface（AWS CLI）结束在账户中启动的会话。如果您在控制台中选择会话的终止按钮或使用 AWS CLI 来调用 TerminateSession API 操作时，Session Manager 会永久结束会话并关闭托管式节点上 Session Manager 客户端与 SSM Agent 之间的数据连接。您无法恢复已终止的会话。

如果打开的会话在 20 分钟内没有用户活动，空闲状态就会触发超时。Session Manager 不会调用 `TerminateSession`，但会关闭底层通道。您无法恢复因空闲超时而关闭的会话。

建议在使用 AWS CLI 时始终通过 `terminate-session` 命令明确终止会话，或者在使用控制台时通过终止按钮明确终止会话。（终止按钮位于会话窗口和主 Session Manager 控制台页面。）如果仅关闭浏览器或命令窗口，则会话将在控制台中保持为活动状态 30 天。如果未明确终止会话或会话超时，则当时在托管式节点上运行的所有进程都将继续运行。

主题

- 结束会话（控制台）
- 结束会话 (AWS CLI)

结束会话（控制台）

您可以使用 AWS Systems Manager 控制台结束您账户中的会话。

结束会话（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Session Manager。

3. 对于 Sessions (会话)，选择要结束的会话左侧的选项按钮。
4. 选择终止。

结束会话 (AWS CLI)

要使用 AWS CLI 结束会话，请运行以下命令。将 *session-id* 替换为您自己的信息。

```
aws ssm terminate-session \
    --session-id session-id
```

有关 terminate-session 命令的更多信息，请参阅《AWS CLI Command Reference》的 AWS Systems Manager 部分中的 terminate-session。

查看会话历史记录

您可以使用 AWS Systems Manager 控制台或 AWS Command Line Interface (AWS CLI) 查看有关账户中的会话的信息。在控制台中，您可以查看会话详细信息，如下所示：

- 会话 ID
- 通过会话连接到托管式节点的用户
- 托管式节点的 ID
- 会话的开始和结束时间
- 会话的状态
- 指定用于存储会话日志的位置（如果开启）

使用 AWS CLI，您可以查看账户中会话的列表，但无法查看控制台中可用的其他详细信息。

有关记录会话历史记录的信息，请参阅 启用和禁用会话日志记录。

主题

- 查看会话历史记录（控制台）
- 查看会话历史记录 (AWS CLI)

查看会话历史记录（控制台）

您可以使用 AWS Systems Manager 控制台查看账户中的会话的详细信息。

查看会话历史记录（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Session Manager。
3. 选择 Session history (会话历史记录) 选项卡。

   –或者–

   如果 Session Manager 主页首先打开，请选择配置首选项，然后选择会话历史记录选项卡。

查看会话历史记录 (AWS CLI)

要使用 AWS CLI 查看账户中会话的列表，请运行以下命令。

```
aws ssm describe-sessions \
    --state History
```

> **Note**
>
> 此命令仅返回与使用 Session Manager 启动的目标所进行的连接的结果。它不列出通过远程桌面协议 (RDP) 或 Secure Shell 协议 (SSH) 等其他方式建立的连接。

有关可以与 describe-sessions 命令结合使用的其他选项的信息，请参阅《AWS CLI Command Reference》中 AWS Systems Manager 部分中的 describe-sessions。

记录会话活动

除了在 Systems Manager 控制台中提供有关当前和已完成会话的信息以外，Session Manager还提供使用 AWS CloudTrail 记录您的 AWS 账户中的会话活动的功能。

CloudTrail 可以捕获通过 Systems Manager 控制台、AWS Command Line Interface (AWS CLI) 和 Systems Manager SDK 进行的 API 调用。您可以在 CloudTrail 控制台查看信息，或将其存储在指定的 Amazon Simple Storage Service (Amazon S3) 存储桶中。您可以将账户的所有 CloudTrail 日志存储在一个 Amazon S3 存储桶中。有关更多信息，请参阅 使用 AWS CloudTrail 记录 AWS Systems Manager API 调用。

> **Note**
>
> 对于定期的、历史的、分析性的日志文件分析，请考虑使用 CloudTrail Lake 或您维护的表来查询 CloudTrail 日志。有关更多信息，请参阅《AWS CloudTrail 用户指南》中的查询 AWS CloudTrail 日志。

使用 Amazon EventBridge 监控会话活动（控制台）

使用 EventBridge，您可以设置规则来检测 AWS 资源何时发生更改。您可以创建规则来检测组织中的用户何时启动或结束会话，然后通过 Amazon SNS 接收有关此事件的通知等。

EventBridge 对 Session Manager 的支持依赖于 CloudTrail 记录的 API 操作记录。（您可以使用 CloudTrail 与 Eventbridge 集成来响应大多数 AWS Systems Manager 事件。）EventBridge 检测不到会话中发生的未进行 API 调用的操作，如 `exit` 命令。

以下步骤概述了如何在发生 Session Manager API 事件（如 StartSession）时通过 Amazon Simple Notification Service (Amazon SNS) 启动通知。

要使用 Amazon EventBridge 监控会话活动（控制台），请执行以下步骤：

1. 创建 Amazon SNS 主题，以便在发生要跟踪的 Session Manager 事件时发送通知。

   有关更多信息，请参阅《Amazon Simple Notification Service 开发人员指南》中的创建主题。
2. 创建 Eventbridge 规则来调用要跟踪的 Session Manager 事件类型的 Amazon SNS 目标。

   要了解如何创建规则，请参阅《Amazon EventBridge 用户指南》中的创建对事件作出反应的 Amazon EventBridge 规则。

   遵循步骤创建规则时，请做出以下选择：

   - 对于 service（AWS 服务），选择 Systems Manager。
   - 对于 Event type（事件类型），选择 AWS API Call through CloudTrail（通过 CloudTrail 进行的 API 调用）。
   - 选择特定操作，然后输入要接收其通知的一个或多个 Session Manager 命令（一次一个）。您可以选择 StartSession、ResumeSession 和 TerminateSession。（EventBridge 不支持 `Get*`、 `List*` 和 `Describe*` 命令。）
   - 对于 Select a target（选择一个目标），选择 SNS topic（SNS 主题）。对于 Topic (主题)，选择您在步骤 1 中创建的 Amazon SNS 主题的名称。

有关更多信息，请参阅 Amazon EventBridge 用户指南和 Amazon Simple Notification Service 入门指南。

启用和禁用会话日志记录

会话日志记录在 Systems Manager 控制台中记录有关当前会话和已完成会话的信息。您还可以在 AWS 账户 中记录有关会话期间运行的命令的详细信息。会话日志记录允许您执行以下操作：

- 创建和存储会话日志以用于存档目的。
- 生成报告，显示过去 30 天内使用 Session Manager 对托管式节点进行的每个连接的详细信息。
- 生成有关 AWS 账户 中的会话活动的通知，例如 Amazon Simple Notification Service（Amazon SNS）通知。
- 作为会话期间所执行操作的结果，在 AWS 资源上自动启动另一个操作，例如运行 AWS Lambda 功能、启动 AWS CodePipeline 管道或运行 AWS Systems Manager Run Command 文档。

Important

请注意Session Manager的以下要求和限制：

- Session Manager 根据您的会话首选项，记录您在会话期间输入的命令及其输出。为了防止敏感数据（如密码）在会话日志中被查看，建议您在会话期间输入敏感数据时使用以下命令。

  Linux & macOS

  ```
  stty -echo; read passwd; stty echo;
  ```

  Windows

  ```
  $Passwd = Read-Host -AsSecureString
  ```

- 如果使用的是 Windows Server 2012 或更早版本，则日志中的数据可能无法以最佳方式进行格式化。建议使用 Windows Server 2012 R2 及更高版本以获得最佳的日志格式。
- 如果使用的是 Linux 或 macOS 托管式节点，请确保已安装 screen 实用工具。否则，日志数据可能会被截断。在 Amazon Linux 1、Amazon Linux 2、AL2023 和 Ubuntu Server 上，默认会安装屏幕实用程序。要手动安装 screen，请根据您的 Linux 版本，运行 `sudo yum install screen` 或 `sudo apt-get install screen`。

- 日志记录不可用于通过端口转发或 SSH 连接的 Session Manager 会话。这是因为 SSH 会加密所有会话数据，而 Session Manager 仅充当 SSH 连接的隧道。

有关使用 Amazon S3 或 Amazon CloudWatch Logs 记录会话数据所需权限的更多信息，请参阅 创建具有 Session Manager、Amazon S3 和 CloudWatch Logs 权限的 IAM 角色（控制台）。

有关 Session Manager 的日志记录选项的更多信息，请参阅以下主题。

主题

- 使用 Amazon CloudWatch Logs 流式传输会话数据（控制台）
- 使用 Amazon S3 记录会话数据（控制台）
- 使用 Amazon CloudWatch Logs 记录会话数据（控制台）
- 配置会话日志记录到磁盘
- 调整Session Manager临时日志文件在磁盘上的存储时间
- 禁用 CloudWatch Logs 和 Amazon S3 中的 Session Manager 日志记录

使用 Amazon CloudWatch Logs 流式传输会话数据（控制台）

您可以向 Amazon CloudWatch Logs 持续发送会话数据日志流。在流式传输会话数据时，将包括必要的详细信息，例如用户在会话中运行的命令、运行命令的用户的 ID，以及将会话数据流式传输到 CloudWatch Logs 的时间戳。流式传输会话数据时，日志采用 JSON 格式，以帮助您与现有的日志记录解决方案相集成。交互式命令不支持流式传输会话数据。

> **Note**
>
> 要从 Windows Server 托管式节点流式传输会话数据，必须安装 PowerShell 5.1 或更高版本。默认情况下，Windows Server 2016 及更高版本已安装了所需的 PowerShell 版本。但是，默认情况下，Windows Server 2012 和 2012 R2 没有安装所需的 PowerShell 版本。如果您尚未在 PowerShell 2012 或 2012 R2 托管式节点上更新 Windows Server，可以使用 Run Command 执行此操作。有关使用 Run Command 更新 PowerShell 的信息，请参阅 使用 Run Command 更新 PowerShell。

> **Important**
>
> 如果您在 Windows Server 托管式节点上配置了 PowerShell 转录策略设置，则将无法流式传输会话数据。

要使用 Amazon CloudWatch Logs 流式传输会话数据（控制台），请执行以下步骤：

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Session Manager。
3. 选择首选项选项卡，然后选择编辑。
4. 选中 CloudWatch logging (CloudWatch 日志记录) 下 Enable (启用) 旁边的复选框。
5. 选择 Stream session logs (流式传输会话日志) 选项。
6. （推荐）选中 Allow only encrypted CloudWatch log groups (仅允许加密的 CloudWatch 日志组) 旁边的复选框。否则，日志数据将使用为日志组指定的服务器端加密密钥加密。如果您希望加密发送到 CloudWatch Logs 的日志数据，请清除此复选框。如果日志组不允许加密，也必须清除此复选框。
7. 对于 CloudWatch logs (CloudWatch Logs 日志)，要指定 AWS 账户中现有的 CloudWatch Logs 日志组来上传会话日志，请选择以下选项之一：
   - 在文本框中输入已在账户中创建的用于存储会话日志数据的日志组的名称。
   - Browse log groups (浏览日志组)：选择已在账户中创建的用于存储会话日志数据的日志组。
8. 选择保存。

使用 Amazon S3 记录会话数据（控制台）

您可以选择将会话日志数据存储在指定的 Amazon Simple Storage Service (Amazon S3) 存储桶中，以便用于调试和故障排除。默认选项是将日志发送到加密的 Amazon S3 存储桶。系统将使用为存储桶指定的密钥执行加密，可以是 AWS KMS key或 Amazon S3 服务器端加密 (SSE) 密钥 (AES-256)。

> **Important**
>
> 当通过安全套接字 (SSL) 使用虚拟托管类型存储桶时，SSL 通配符证书仅匹配不包含句点的存储桶。要解决此问题，请使用 HTTP 或编写自己的证书验证逻辑。在使用虚拟托管类型存储桶时，建议您不要在存储桶名称中使用句点（“.”）。

Amazon S3 存储桶加密

要将日志发送到 Amazon S3 存储桶并进行加密，必须在存储桶上允许加密。有关 Amazon S3 存储桶加密的更多信息，请参阅 Amazon S3 默认 S3 存储桶加密。

客户托管密钥

如果使用自己管理的 KMS 密钥来加密存储桶，则附加到实例的 IAM 实例配置文件必须具有读取此密钥的显式权限。如果使用 AWS 托管式密钥，则实例不需要此显式权限。有关为实例配置文件提供使用密钥的权限的更多信息，请参阅《AWS Key Management Service 开发人员指南》中的允许密钥用户使用密钥。

请按照以下步骤配置 Session Manager，以将会话日志存储在 Amazon S3 存储桶中。

> **Note**
>
> 您也可以使用 AWS CLI 指定或更改将会话数据发送到的 Amazon S3 存储桶。有关信息，请参阅更新 Session Manager 首选项（命令行）。

要使用 Amazon S3 记录会话数据（控制台），请执行以下步骤：

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Session Manager。
3. 选择首选项选项卡，然后选择编辑。
4. 选中 S3 logging (S3 日志记录) 下 Enable (启用) 旁边的复选框。
5. （推荐）选中 Allow only encrypted S3 buckets (仅允许加密的 S3 存储桶) 旁边的复选框。开启此选项后，将使用为存储桶指定的服务器端加密密钥对日志数据进行加密。如果不需要加密发送到 Amazon S3 的日志数据，请清除此复选框。如果 S3 存储桶不允许加密，也必须清除此复选框。
6. 对于 S3 bucket name (S3 存储桶名称)，请选择以下选项之一：

   > **Note**
   >
   > 在使用虚拟托管类型存储桶时，建议您不要在存储桶名称中使用句点（“.”）。有关 Amazon S3 存储桶命名约定的更多信息，请参阅 Amazon Simple Storage Service 用户指南中的存储桶限制和局限性。

- Choose a bucket name from the list (从列表中选择存储桶名称)：选择已在账户中创建的用于存储会话日志数据的 Amazon S3 存储桶。
- Enter a bucket name in the text box (在文本框中输入存储桶名称)：输入已在账户中创建的用于存储会话日志数据的 Amazon S3 存储桶名称。

7. （可选）对于 S3 key prefix (S3 键前缀)，输入现有或新文件夹的名称，以便将日志存储在所选的存储桶中。
8. 选择保存。

有关使用 Amazon S3 和 Amazon S3 存储桶的更多信息，请参阅 Amazon Simple Storage Service 用户指南和 Amazon Simple Storage Service 用户指南。

使用 Amazon CloudWatch Logs 记录会话数据（控制台）

借助 Amazon CloudWatch Logs，您可以监控、存储和访问各种 AWS 服务的日志文件。您可以将会话日志数据发送到 CloudWatch Logs 日志组，以便用于调试和故障排除。默认选项是使用您的 KMS 密钥在加密后发送日志数据，但您可以将数据发送到日志组（加密或不加密）。

请按照以下步骤配置 AWS Systems Manager Session Manager，以便在会话结束时将会话日志数据发送到 CloudWatch Logs 日志组。

> **Note**
>
> 您也可以使用 AWS CLI 指定或更改将会话数据发送到的 CloudWatch Logs 日志组。有关信息，请参阅更新 Session Manager 首选项（命令行）。

要使用 Amazon CloudWatch Logs 记录会话数据（控制台），请执行以下步骤：

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Session Manager。
3. 选择首选项选项卡，然后选择编辑。
4. 选中 CloudWatch logging (CloudWatch 日志记录) 下 Enable (启用) 旁边的复选框。
5. 选择 Upload session logs (上传会话日志) 选项。
6. （推荐）选中 Allow only encrypted CloudWatch log groups (仅允许加密的 CloudWatch 日志组) 旁边的复选框。否则，日志数据将使用为日志组指定的服务器端加密密钥加密。如果您希望加密发

送到 CloudWatch Logs 的日志数据，请清除此复选框。如果日志组不允许加密，也必须清除此复选框。

7. 对于 CloudWatch logs (CloudWatch Logs 日志)，要指定 AWS 账户中现有的 CloudWatch Logs 日志组来上传会话日志，请选择以下选项之一：

   - Choose a log group from the list (从列表中选择日志组)：选择已在账户中创建的日志组来存储会话日志数据。
   - Enter a log group name in the text box (在文本框中输入日志组名称)：输入已在您的账户中创建的用于存储会话日志数据的日志组的名称。

8. 选择保存。

有关使用 CloudWatch Logs 的更多信息，请参阅 Amazon CloudWatch Logs 用户指南。

配置会话日志记录到磁盘

在启用Session Manager日志记录到 CloudWatch 或 Amazon S3 后，会话期间执行的所有命令（以及这些命令产生的输出）都将记录到目标实例磁盘上的临时文件中。临时文件的名称为 `ipcTempFile.log`。

`ipcTempFile.log` 由 SSM Agent 配置文件中的 `SessionLogsDestination` 参数控制。此参数接受以下值：

- disk：如果指定此参数并且启用会话日志记录到 CloudWatch 或 Amazon S3，则 SSM Agent 将创建 `ipcTempFile.log` 临时日志文件并将会话命令和输出记录到磁盘。Session Manager会在会话期间或结束后将此日志上传到 CloudWatch 或 S3，具体取决于日志记录配置。然后根据为 SSM Agent `SessionLogsRetentionDurationHours` 配置参数指定的持续时间删除日志。

  如果指定此参数并禁用会话日志记录到 CloudWatch 和 Amazon S3，则 SSM Agent 仍会在 `ipcTempFile.log` 文件中记录命令历史记录和输出。然后将根据为 SSM Agent `SessionLogsRetentionDurationHours` 配置参数指定的持续时间删除该文件。
- none：如果指定此参数并且启用会话日志记录到 CloudWatch 或 Amazon S3，则日志记录到磁盘的工作方式与您指定 `disk` 参数时完全相同。当启用会话日志记录到 CloudWatch 或 Amazon S3 时，SSM Agent 需要临时文件。

  如果指定此参数并禁用会话日志记录到 CloudWatch 或 Amazon S3，则 SSM Agent 不会创建 `ipcTempFile.log` 文件。

使用以下过程启用或禁用在启动会话时在磁盘上创建 `ipcTempFile.log` 临时日志文件。

启用或禁用在磁盘上创建Session Manager临时日志文件

1. 在您的实例上安装 SSM Agent 或升级到版本 3.2.2086 或更高版本。有关如何检查代理版本号的信息，请参阅正在检查 SSM Agent 版本号。有关如何手动安装代理的信息，请在以下部分中找到适合您的操作系统的过程：

   - 在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent
   - 在适用于 macOS 的 EC2 实例上手动安装和卸载 SSM Agent
   - 在适用于 Windows Server 的 EC2 实例上手动安装和卸载 SSM Agent

2. 连接到您的实例并在以下位置找到 `amazon-ssm-agent.json` 文件。

   - Linux：/etc/amazon/ssm/
   - macOS：/opt/aws/ssm/
   - Windows Server：C:\Program Files\Amazon\SSM

   如果文件 `amazon-ssm-agent.json` 不存在，请将 `amazon-ssm-agent.json.template` 的内容复制到同一目录中的新文件中。将新文件命名为 `amazon-ssm-agent.json`。

3. 为 `SessionLogsDestination` 参数指定 `none` 或 `disk`。保存您的更改。

4. 重启 SSM Agent。

如果为 `SessionLogsDestination` 参数指定了 `disk`，则可以通过启动新会话，然后在以下位置找到 `ipcTempFile.log` 来验证 SSM Agent 是否创建了临时日志文件：

- Linux：/var/lib/amazon/ssm/*## ID*/session/orchestration/*## ID*/Standard_Stream/ipcTempFile.log
- macOS：/opt/aws/ssm/data/*## ID*/session/orchestration/*## ID*/Standard_Stream/ipcTempFile.log
- Windows Server：C:\ProgramData\Amazon\SSM\InstanceData*## ID*\session\orchestration*## ID*\Standard_Stream\ipcTempFile.log

> **Note**
>
> 默认情况下，临时日志文件会在实例上保存 14 天。

如果您想要更新多个实例中的 SessionLogsDestination 参数，我们建议您创建一个指定新配置的 SSM 文档。然后，您可以使用 Systems Manager Run Command 在实例上实施更改。有关更多信息，请参阅 Writing your own AWS Systems Manager documents (blog) 和在托管节点上运行命令。

调整Session Manager临时日志文件在磁盘上的存储时间

在启用Session Manager日志记录到 CloudWatch 或 Amazon S3 后，会话期间执行的所有命令（以及这些命令产生的输出）都将记录到目标实例磁盘上的临时文件中。临时文件的名称为 ipcTempFile.log。在会话期间或会话完成后，Session Manager会将此临时日志上传到 CloudWatch 或 S3。然后根据为 SSM Agent SessionLogsRetentionDurationHours 配置参数指定的持续时间删除临时日志。默认情况下，临时日志文件将在实例上的下列位置保存 14 天：

- Linux：/var/lib/amazon/ssm/*## ID*/session/orchestration/*## ID*/Standard_Stream/ipcTempFile.log
- macOS：/opt/aws/ssm/data/*## ID*/session/orchestration/*## ID*/Standard_Stream/ipcTempFile.log
- Windows Server：C:\ProgramData\Amazon\SSM\InstanceData*## ID*\session\orchestration*## ID*\Standard_Stream\ipcTempFile.log

使用下面的步骤调整Session Manager临时日志文件在磁盘上的存储时间。

调整 **ipcTempFile.log** 文件在磁盘上的存储时间

1. 连接到您的实例并在以下位置找到 amazon-ssm-agent.json 文件。

   - Linux：/etc/amazon/ssm/
   - macOS：/opt/aws/ssm/
   - Windows Server：C:\Program Files\Amazon\SSM

   如果文件 amazon-ssm-agent.json 不存在，请将 amazon-ssm-agent.json.template 的内容复制到同一目录中的新文件中。将新文件命名为 amazon-ssm-agent.json。
2. 将 SessionLogsRetentionDurationHours 的值更改为所需的小时数。如果 SessionLogsRetentionDurationHours 设置为 0，则将在会话期间创建临时日志文件，并在会话完成后删除该文件。此设置应确保日志文件在会话结束后不会保留。
3. 保存您的更改。
4. 重启 SSM Agent。

禁用 CloudWatch Logs 和 Amazon S3 中的 Session Manager 日志记录

您可以使用 Systems Manager 控制台或 AWS CLI 禁用账户中的会话日志记录。

要禁用会话日志记录（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Session Manager。
3. 选择首选项选项卡，然后选择编辑。
4. 要禁用 CloudWatch 日志记录，在 CloudWatch 日志记录部分中清除启用复选框。
5. 要禁用 S3 日志记录，在 S3 日志记录部分中，清除启用复选框。
6. 选择保存。

要禁用会话日志记录（AWS CLI）

要使用 AWS CLI 禁用会话日志记录，请按照 更新 Session Manager 首选项（命令行） 中的说明进行操作。

在您的 JSON 文件中，确保 `s3BucketName` 和 `cloudWatchLogGroupName` 的输入不包含任何值。例如：

```
"inputs": {
        "s3BucketName": "",
        ...
        "cloudWatchLogGroupName": "",
        ...
    }
```

或者，要禁用日志记录，您可以从 JSON 文件中删除所有 `S3*` 和 `cloudWatch*` 输入。

> **Note**
>
> 根据您的配置，在您禁用 CloudWatch 或 S3 后，SSM Agent 仍可能会在磁盘上生成一个临时日志文件。有关如何禁用日志记录到磁盘的信息，请参阅配置会话日志记录到磁盘。

会话文档架构

以下信息介绍了会话文档的架构元素。AWS Systems Manager Session Manager 使用会话文档来确定要启动哪些类型的会话，例如标准会话、端口转发会话或运行交互式命令的会话。

schemaVersion

会话文档的架构版本。会话文档仅支持版本 1.0。

类型：字符串

必需：是

description

为会话文档指定的描述。例如，“用于启动与 Session Manager 的端口转发会话的文档”。

类型：字符串

必需：否

sessionType

会话文档用于建立的会话类型。

类型：字符串

必需：是

有效值：`InteractiveCommands` | `NonInteractiveCommands` | `Port` | `Standard_Stream`

inputs

用于使用此会话文档建立的会话的会话首选项。用于创建 `Standard_Stream` 会话的会话文档需要此元素。

类型：StringMap

必需：否

s3BucketName

您希望在会话结束时将会话日志发送到的 Amazon Simple Storage Service (Amazon S3) 存储桶。

类型：字符串

必需：否

s3KeyPrefix

将日志发送到您在输入 s3BucketName 时指定的 Amazon S3 存储桶时使用的前缀。有关对 Amazon S3 中存储的对象使用共享前缀的更多信息，请参阅 Amazon Simple Storage Service 用户指南中的如何在 S3 存储桶中使用文件夹？。

类型：字符串

必需：否

s3EncryptionEnabled

如果设置为 true，则输入 s3BucketName 时指定的 Amazon S3 存储桶必须进行加密。

类型：布尔值

必需：是

cloudWatchLogGroupName

您希望在会话结束时将会话日志发送到的 Amazon CloudWatch Logs (CloudWatch Logs) 组名称。

类型：字符串

必需：否

cloudWatchEncryptionEnabled

如果设置为 true，则输入 cloudWatchLogGroupName 时指定的日志组必须进行加密。

类型：布尔值

必需：是

cloudWatchStreamingEnabled

如果设置为 true，会话数据日志流将持续发送到您在输入 cloudWatchLogGroupName 时指定的日志组。如果设置为 false，会话日志将在会话结束时发送到您在输入 cloudWatchLogGroupName 时指定的日志组。

类型：布尔值

必需：是

kmsKeyId

您要用来进一步加密本地客户端与所连接的 Amazon Elastic Compute Cloud (Amazon EC2) 托管式节点之间数据的 AWS KMS key 的 ID。

类型：字符串

必需：否

runAsEnabled

如果设置为 `true`，则必须在 `runAsDefaultUser` 输入中指定一个存在于要连接的托管式节点上的用户账户。否则，会话将无法启动。默认情况下，会话使用 AWS Systems Manager SSM Agent 创建的 `ssm-user` 账户启动。只有连接到 Linux 和 macOS 托管节点才支持“运行身份”功能。

类型：布尔值

必需：是

runAsDefaultUser

当 `runAsEnabled` 输入设置为 `true` 时，用于在 Linux 和 macOS 托管节点上启动会话的用户账户的名称。您为此输入指定的用户账户必须存在于要连接的托管式节点上；否则，会话将无法启动。

类型：字符串

必需：否

idleSessionTimeout

结束会话前允许处于非活动状态的时间长度。此输入以分钟为单位。

类型：字符串

有效值：1-60

必需：否

maxSessionDuration

会话结束前允许的最长时间。此输入以分钟为单位。

类型：字符串

有效值：1-1440

必需：否

shellProfile

根据操作系统指定的要在会话中应用的首选项，例如 Shell 首选项、环境变量、工作目录以及在启动会话时运行多个命令。

类型：StringMap

必需：否

windows

为 Windows Server 托管式节点上的会话指定的 Shell 首选项、环境变量、工作目录和命令。

类型：字符串

必需：否

linux

为 Linux 和 macOS 托管节点上的会话指定的 Shell 首选项、环境变量、工作目录和命令。

类型：字符串

必需：否

parameters

定义文档接受的参数的对象。有关定义文档参数的更多信息，请参阅顶级数据元素中的参数。对于经常引用的参数，建议在 Systems Manager Parameter Store 中存储这些参数，以便进行引用。您可以在文档的这一部分引用 `String` 和 `StringList` Parameter Store 参数。您不能在文档的这一部分引用 `SecureString` Parameter Store 参数。您可以使用以下格式引用 Parameter Store 参数：

```
{{ssm:parameter-name}}
```

有关 Parameter Store 的更多信息，请参阅 AWS Systems Manager Parameter Store。

类型：StringMap

必需：否

properties

在 `StartSession` API 操作中使用的一个对象，其值由您指定。

对于用于 InteractiveCommands 会话的会话文档，属性对象包含要在您指定的操作系统上运行的命令。您还可以 runAsElevated 布尔属性来确定命令是否作为 root 运行。有关更多信息，请参阅限制对会话中命令的访问。

对于用于 Port 会话的会话文档，属性对象包含应将流量重定向到的端口号。有关示例，请参阅本主题后文中的 Port 类型会话文档示例。

类型：StringMap

必需：否

Standard_Stream 类型会话文档示例

YAML

```
---
schemaVersion: '1.0'
description: Document to hold regional settings for Session Manager
sessionType: Standard_Stream
inputs:
  s3BucketName: ''
  s3KeyPrefix: ''
  s3EncryptionEnabled: true
  cloudWatchLogGroupName: ''
  cloudWatchEncryptionEnabled: true
  cloudWatchStreamingEnabled: true
  kmsKeyId: ''
  runAsEnabled: true
  runAsDefaultUser: ''
  idleSessionTimeout: '20'
  maxSessionDuration: '60'
  shellProfile:
    windows: ''
    linux: ''
```

JSON

```
{
    "schemaVersion": "1.0",
    "description": "Document to hold regional settings for Session Manager",
    "sessionType": "Standard_Stream",
    "inputs": {
```

```
        "s3BucketName": "",
        "s3KeyPrefix": "",
        "s3EncryptionEnabled": true,
        "cloudWatchLogGroupName": "",
        "cloudWatchEncryptionEnabled": true,
        "cloudWatchStreamingEnabled": true,
        "kmsKeyId": "",
        "runAsEnabled": true,
        "runAsDefaultUser": "",
        "idleSessionTimeout": "20",
        "maxSessionDuration": "60",
        "shellProfile": {
            "windows": "date",
            "linux": "pwd;ls"
        }
    }
}
```

InteractiveCommands 类型会话文档示例

YAML

```
---
schemaVersion: '1.0'
description: Document to view a log file on a Linux instance
sessionType: InteractiveCommands
parameters:
  logpath:
    type: String
    description: The log file path to read.
    default: "/var/log/amazon/ssm/amazon-ssm-agent.log"
    allowedPattern: "^[a-zA-Z0-9-_/]+(.log)$"
properties:
  linux:
    commands: "tail -f {{ logpath }}"
    runAsElevated: true
```

JSON

```
{
    "schemaVersion": "1.0",
    "description": "Document to view a log file on a Linux instance",
```

```
    "sessionType": "InteractiveCommands",
    "parameters": {
        "logpath": {
            "type": "String",
            "description": "The log file path to read.",
            "default": "/var/log/amazon/ssm/amazon-ssm-agent.log",
            "allowedPattern": "^[a-zA-Z0-9-_/]+(.log)$"
        }
    },
    "properties": {
        "linux": {
            "commands": "tail -f {{ logpath }}",
            "runAsElevated": true
        }
    }
}
```

Port 类型会话文档示例

YAML

```
---
schemaVersion: '1.0'
description: Document to open given port connection over Session Manager
sessionType: Port
parameters:
  paramExample:
    type: string
    description: document parameter
properties:
  portNumber: anyPortNumber
```

JSON

```
{
    "schemaVersion": "1.0",
    "description": "Document to open given port connection over Session Manager",
    "sessionType": "Port",
    "parameters": {
        "paramExample": {
            "type": "string",
            "description": "document parameter"
```

```
        }
    },
    "properties": {
        "portNumber": "anyPortNumber"
    }
}
```

包含特殊字符的会话文档示例

YAML

```
---
schemaVersion: '1.0'
description: Example document with quotation marks
sessionType: InteractiveCommands
parameters:
  Test:
    type: String
    description: Test Input
    maxChars: 32
properties:
  windows:
    commands: |
        $Test = '{{ Test }}'
        $myVariable = \"Computer name is $env:COMPUTERNAME\"
        Write-Host "Test variable: $myVariable`.`nInput parameter: $Test"
    runAsElevated: false
```

JSON

```
{
   "schemaVersion":"1.0",
   "description":"Test document with quotation marks",
   "sessionType":"InteractiveCommands",
   "parameters":{
      "Test":{
         "type":"String",
         "description":"Test Input",
         "maxChars":32
      }
   },
   "properties":{
```

```
      "windows":{
         "commands":[
            "$Test = '{{ Test }}'",
            "$myVariable = \\\"Computer name is $env:COMPUTERNAME\\\"",
            "Write-Host \"Test variable: $myVariable`.`nInput parameter: $Test\""
         ],
         "runAsElevated":false
      }
   }
}
```

排查 Session Manager 问题

可以使用以下信息来帮助解决 AWS Systems Manager Session Manager 问题。

主题

- 调用 TerminateSession 操作时的 AccessDeniedException
- 文档处理意外失败：文档工作线程超时
- Session Manager 无法从 Amazon EC2 控制台连接
- 没有启动会话的权限
- SSM Agent 不在线
- 没有更改会话首选项的权限
- 托管式节点不可用或未为 Session Manager 配置托管式节点
- 未找到 Session Manager 插件
- Session Manager 插件未自动添加到命令行路径（Windows）
- Session Manager 插件变得没有响应
- TargetNotConnected
- 启动会话后显示空白屏幕
- 托管式节点在长时间运行会话期间变得没有响应
- 调用 StartSession 操作时出现错误（InvalidDocument）

调用 TerminateSession 操作时的 AccessDeniedException

问题：尝试终止会话时，Systems Manager 会返回以下错误：

```
An error occurred (AccessDeniedException) when calling the TerminateSession operation:
User: <user_arn> is not authorized to perform: ssm:TerminateSession on resource:
<ssm_session_arn> because no identity-based policy allows the ssm:TerminateSession
 action.
```

解决方案 A：确认节点上已安装最新版本的 Session Manager 插件

在终端中输入以下命令，然后按 Enter。

```
session-manager-plugin --version
```

解决方案 B：安装或重新安装最新版本的插件

有关更多信息，请参阅 为 AWS CLI 安装 Session Manager 插件。

解决方案 C：尝试重新建立与节点的连接

验证节点是否在响应请求。尝试重新建立会话。或者，在必要时，打开 Amazon EC2 控制台并验证实例的状态是否正在运行。

文档处理意外失败：文档工作线程超时

问题：启动与 Linux 主机的会话时，Systems Manager 返回以下错误：

```
document process failed unexpectedly: document worker timed out,
check [ssm-document-worker]/[ssm-session-worker] log for crash reason
```

如果您配置了SSM Agent 日志记录（如 查看 SSM Agent 日志 中所述），则可以在调试日志中查看更多详细信息。对于此问题，Session Manager 显示以下日志条目：

```
failed to create channel: too many open files
```

此错误通常表示运行的 Session Manager 工作线程进程过多，底层操作系统达到了极限。您有两个选项来解决此问题。

解决方案 A：提高操作系统文件通知限制

您可以在单独的 Linux 主机上运行以下命令来提高限制。该命令使用 Systems Manager Run Command。指定值会将 `max_user_instances` 增加到 8192。此值远高于默认值 128，但它不会占用主机资源：

```
aws ssm send-command --document-name AWS-RunShellScript \
--instance-id i-02573cafcfEXAMPLE  --parameters \
"commands=sudo sysctl fs.inotify.max_user_instances=8192"
```

解决方案 B：减少 Session Manager 在目标主机中使用的文件通知

在单独的 Linux 主机上运行以下命令，以列出目标主机上运行的会话：

```
aws ssm describe-sessions --state Active --filters key=Target,value=i-02573cafcfEXAMPLE
```

查看命令输出，以确定不再需要的会话。您可以在单独的 Linux 主机上运行以下命令来终止这些会话：

```
aws ssm terminate-session –session-id session ID
```

或者，一旦远程服务器上不再运行会话，您可以通过在单独的 Linux 主机上运行以下命令来释放更多资源。此命令将终止远程主机上运行的所有 Session Manager 进程，从而终止与远程主机的所有会话。在运行此命令之前，请确认没有要保留的正在进行的会话：

```
aws ssm send-command --document-name AWS-RunShellScript \
            --instance-id i-02573cafcfEXAMPLE --parameters \
'{"commands":["sudo kill $(ps aux | grep ssm-session-worker | grep -v grep | awk
 '"'"'{print $2}'"'"')"]}'
```

Session Manager 无法从 Amazon EC2 控制台连接

问题：创建新实例后，Amazon Elastic Compute Cloud（Amazon EC2）控制台中的链接按钮 > 会话管理器选项卡不提供连接选项。

解决方案 A：创建实例配置文件：如果您尚未执行此操作（按照 EC2 控制台中会话管理器选项卡上的信息的说明），使用Quick Setup创建 AWS Identity and Access Management（IAM）实例配置文件。Quick Setup是 AWS Systems Manager 中的一项工具。

Session Manager 需要 IAM 实例配置文件才能连接到实例。您可以使用 Quick Setup 创建主机管理配置，以创建实例配置文件并将其分配给实例。主机管理配置创建具有所需权限的实例配置文件并将其分配给实例。主机管理配置还可以启用其他 Systems Manager 工具，并创建用于运行这些工具的 IAM 角色。使用Quick Setup或主机管理配置启用的工具不会产生任何费用。打开 Quick Setup 并创建主机管理配置。

> **Important**
>
> 创建主机管理配置后，Amazon EC2 可能需要几分钟来注册更改并刷新 会话管理器选项卡。如果选项卡在两分钟后没有显示连接按钮，请重新启动实例。如果重新启动后仍然看不到连接选项，请打开快速设置功能，确认只有一个主机管理配置。如果有两个主机管理配置，请删除较旧的配置并等待几分钟。

如果在创建主机管理配置后仍然无法连接，或者收到错误（包括关于 SSM Agent 的错误），请参阅以下解决方案之一：

- 解决方案 B：没有错误，但仍然无法连接
- 解决方案 C：关于缺失 SSM Agent 的错误

解决方案 B：没有错误，但仍然无法连接

如果您创建了主机管理配置，等待了几分钟才尝试连接，但仍然无法连接，则可能需要手动将主机管理配置应用于实例。使用以下过程更新 Quick Setup 主机管理配置并将更改应用于实例。

使用 Quick Setup 更新主机管理配置

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Quick Setup。
3. 在配置列表中，选择您创建的主机管理配置。
4. 选择操作，然后选择编辑配置。
5. 在目标部分的底部附近，于选择您希望如何定位目标下选择手动。
6. 在实例部分中，选择您创建的实例。
7. 选择更新。

等待几分钟，让 EC2 刷新会话管理器选项卡。如果仍然无法连接或收到错误，请查看该问题的其余解决方案。

解决方案 C：关于缺失 SSM Agent 的错误

如果您无法使用 Quick Setup 创建主机管理配置，或者收到有关未安装 SSM Agent 的错误，则可能需要在实例上手动安装 SSM Agent。SSM Agent 是 Amazon 软件，它允许 Systems Manager 通过使

用 Session Manager 连接到实例。SSM Agent 默认安装在大多数亚马逊机器映像（AMI）上。如果您的实例是从非标准 AMI 或较旧的 AMI 创建的，则可能需要手动安装代理。有关安装 SSM Agent 的过程，请参阅以下与您的实例操作系统相对应的主题。

- Windows Server
- macOS
- AlmaLinux
- Amazon Linux 1
- Amazon Linux 2 和 AL2023
- CentOS
- CentOS Stream
- Debian Server
- Oracle Linux
- Red Hat Enterprise Linux
- Rocky Linux
- SUSE Linux Enterprise Server
- Ubuntu Server

有关 SSM Agent 的问题，请参阅 排查 SSM Agent 问题。

没有启动会话的权限

问题：您尝试启动会话，但系统提示您没有必要的权限。

- 解决方案：系统管理员尚未授予您启动 Session Manager 会话的 AWS Identity and Access Management (IAM) 策略权限。有关更多信息，请参阅控制用户会话对实例的访问权限。

SSM Agent 不在线

问题：您在 Amazon EC2 实例 Session Manager 选项卡上看到一条消息，表明：“SSM Agent 不在线。SSM Agent 不能够连接到 Systems Manager 端点以将其自身注册到服务中。”

解决方案：SSM Agent 是在 Amazon EC2 实例上运行的 Amazon 软件，从而 Session Manager 可以与其建立连接。如果您看到此错误，则 SSM Agent 不能够建立与 Systems Manager 端点的连接。问

题的可能原因可能是防火墙限制、路由问题或互联网连接不足。要解决此问题，请调查网络连接问题。有关更多信息，请参阅排查 SSM Agent 问题和排除托管式节点可用性的问题。有关 Systems Manager 端点的信息，请参阅《AWS 一般参考》中的 AWS Systems Manager 端点和配额。

没有更改会话首选项的权限

问题：您尝试更新组织的全局会话首选项，但系统提示您没有必要的权限。

- 解决方案：系统管理员尚未授予您设置 Session Manager 首选项的 IAM policy 权限。有关信息，请参阅授予或拒绝更新 Session Manager 首选项的用户权限。

托管式节点不可用或未为 Session Manager 配置托管式节点

问题 1：您要在 Start a session（启动会话）控制台页面上启动一个会话，但托管式节点不在列表中。

- 解决方案 A：可能尚未为 AWS Systems Manager 配置您要连接的托管式节点。有关更多信息，请参阅 为组织设置 Systems Manager 统一控制台。

> **Note**
>
> 如果在附加 IAM 实例配置文件时 AWS Systems Manager SSM Agent 已在托管式节点上运行，则在 Start a session （启动会话）控制台页面上列出该实例之前，可能需要重新启动代理。

- 解决方案 B：您应用于托管式节点上 SSM Agent 的代理配置可能不正确。如果代理配置不正确，托管式节点将无法到达所需的服务终端节点，或节点可能会向 Systems Manager 报告为不同的操作系统。有关更多信息，请参阅配置 SSM Agent 以在 Linux 节点上使用代理和配置 SSM Agent以使用 Windows Server 实例的代理。

问题 2：Start a session（开启会话）控制台页面上的列表中显示了要连接的托管式节点，但页面报告“The instance you selected isn't configured to use Session Manager”（您选择的实例未配置为使用 ）。

- 解决方案 A：托管式节点已配置为可以使用 Systems Manager 服务，但附加到节点的 IAM 实例配置文件不包含Session Manager工具所需的权限。有关更多信息，请参阅验证或创建具有 Session Manager 权限的 IAM 实例配置文件。
- 解决方案 B：托管式节点未运行支持 Session Manager 的 SSM Agent 版本。将节点上的 SSM Agent 更新为版本 2.3.68.0 或更高版本。

根据操作系统类型，按照 在适用于 Windows Server 的 EC2 实例上手动安装和卸载 SSM Agent、在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent 或 在适用于 macOS 的 EC2 实例上手动安装和卸载 SSM Agent 中的步骤操作，手动更新托管式节点上的 SSM Agent。

或者，使用 Run Command 文档 `AWS-UpdateSSMAgent` 一次性更新一个或多个托管式节点上的代理版本。有关信息，请参阅使用 Run Command 更新 SSM Agent。

> **Tip**
>
> 为了让代理始终保持最新状态，建议使用以下方法之一按照定义的自动计划将 SSM Agent更新为最新版本：
>
> - 作为 State Manager 关联的一部分运行 `AWS-UpdateSSMAgent`。有关信息，请参阅演练：使用 AWS CLI 自动更新 SSM Agent。
> - 运行 `AWS-UpdateSSMAgent` 作为维护时段的一部分。有关使用维护时段的信息，请参阅使用控制台创建和管理维护时段 和 教程：使用 AWS CLI 创建和配置维护时段。

- 解决方案 C：托管式节点无法到达必要的服务终端节点。您可以使用由 AWS PrivateLink 提供支持的接口端点连接到 Systems Manager 端点，以提高托管式节点的安保状况。使用接口端点的替代方法是，在托管式节点上允许出站互联网访问。有关更多信息，请参阅使用 PrivateLink 为 Session Manager 设置 VPC 端点。
- 解决方案 D：托管式节点的可用 CPU 或内存资源有限。尽管您的托管式节点可能正常运行，但是如果该节点没有足够的可用资源，则您无法建立会话。有关更多信息，请参阅对无法访问的实例进行故障排除。

未找到 Session Manager 插件

要使用 AWS CLI 运行会话命令，还必须在本地计算机上安装 Session Manager 插件。有关信息，请参阅为 AWS CLI 安装 Session Manager 插件。

Session Manager 插件未自动添加到命令行路径（Windows）

在 Windows 上安装 Session Manager 插件时，`session-manager-plugin` 可执行文件应自动添加到操作系统的 `PATH` 环境变量中。如果在运行检查是否正确安装了 Session Manager 插件的命令后命令失败 (`aws ssm start-session --target` *`instance-id`*)，则需要使用以下过程进行手动设置。

修改您的 PATH 变量 (Windows)

1. 按 Windows 键并输入 **environment variables**。
2. 选择 Edit environment variables for your account（编辑您账户的环境变量）。
3. 选择 PATH，然后选择 Edit。
4. 向 Variable value (变量值) 字段添加路径，用分号分隔，如下例所示：*C:\existing\path*;***C:\new\path***

   C:\existing\path 表示字段中已存在的值。***C:\new\path*** 表示要添加的路径，如以下示例所示。

   - 64 位计算机：`C:\Program Files\Amazon\SessionManagerPlugin\bin\`
   - 32 位计算机：`C:\Program Files (x86)\Amazon\SessionManagerPlugin\bin\`
5. 选择 OK（确定）两次以应用新设置。
6. 关闭任何运行的命令提示符并重新打开。

Session Manager 插件变得没有响应

如果您的本地计算机上安装了防病毒软件，在端口转发会话期间，可能会停止转发流量。在某些情况下，防病毒软件会干扰 Session Manager 插件，从而导致进程死锁。要解决此问题，请在防病毒软件中设置允许 Session Manager 插件或将其排除在外。有关 Session Manager 插件的默认安装路径的信息，请参阅 为 AWS CLI 安装 Session Manager 插件。

TargetNotConnected

问题：您尝试启动会话，但系统返回错误消息：“An error occurred (TargetNotConnected) when calling the StartSession operation: *InstanceID* isn't connected.”（调用 StartSession 操作时发生错误 (TargetNotConnected)：InstanceID 未连接。）

- 解决方案 A：如果会话的指定目标托管式节点未完全配置为与 Session Manager 一起使用，则会返回此错误。有关信息，请参阅设置 Session Manager。
- 解决方案 B：如果您尝试在位于不同 AWS 账户 或 AWS 区域 的托管式节点上启动会话，也会返回此错误。

启动会话后显示空白屏幕

问题：在您启动会话后，Session Manager 显示空白屏幕。

- 解决方案 A：如果托管式节点上的根卷已满，则可能出现此问题。由于磁盘空间不足，节点上的 SSM Agent 停止工作。要解决此问题，请使用 Amazon CloudWatch 从操作系统中收集指标和日志。有关更多信息，请参阅 Amazon CloudWatch 用户指南中的使用 CloudWatch 代理收集指标、日志和跟踪信息。
- 解决方案 B：如果您使用包含不匹配端点和区域对的链接访问控制台，则会显示空白屏幕。例如，在以下控制台 URL 中，us-west-2 是指定的端点，但 us-west-1 是指定的 AWS 区域。

```
https://us-west-2.console.aws.amazon.com/systems-manager/session-manager/sessions?
region=us-west-1
```

- 解决方案 C：托管式节点正在使用 VPC 终端节点连接到 Systems Manager，并且您的 Session Manager 首选项将会话输出写入 Amazon S3 存储桶或 Amazon CloudWatch Logs 日志组，但 VPC 中不存在 s3 网关端点或 logs 接口端点。如果您的托管式节点使用 VPC 终端节点连接到 Systems Manager，并且您的 Session Manager 首选项将会话输出写入 Amazon S3 存储桶，则需要格式为 **com.amazonaws.*region*.s3** 的 s3 端点。或者，如果您的托管式节点使用 VPC 终端节点连接到 Systems Manager，并且您的 Session Manager 首选项将会话输出写入 CloudWatch Logs 日志组，则需要格式为 **com.amazonaws.*region*.logs** 的 logs 端点。有关更多信息，请参阅 为 Systems Manager 创建 VPC 终端节点。
- 解决方案 D：您在会话首选项中指定的日志组或 Amazon S3 存储桶已删除。要解决此问题，请使用有效的日志组或 S3 存储桶更新会话首选项。
- 解决方案 E：您在会话首选项中指定的日志组或 Amazon S3 存储桶未加密，但您已将 cloudWatchEncryptionEnabled 或 s3EncryptionEnabled 输入设置为 true。要解决此问题，请使用加密的日志组或 Amazon S3 存储桶更新会话首选项，或将 cloudWatchEncryptionEnabled 或 s3EncryptionEnabled 输入设置为 false。此方案仅适用于使用命令行工具创建会话首选项的客户。

托管式节点在长时间运行会话期间变得没有响应

问题：在长时间运行会话期间，托管式节点变得没有响应或崩溃。

解决方案：减少 Session Manager 的 SSM Agent 日志保留时间。

要减少会话的 SSM Agent 日志保留时间，请执行以下步骤：

1. 在 Linux 的 /etc/amazon/ssm/ 目录或 Windows 的 C:\Program Files\Amazon\SSM 中找到 amazon-ssm-agent.json.template。
2. 将 amazon-ssm-agent.json.template 的内容复制到同一目录中名为 amazon-ssm-agent.json 的新文件中。

3. 减小 SSM 属性中 `SessionLogsRetentionDurationHours` 值的默认值，然后保存该文件。
4. 重启 SSM Agent。

调用 StartSession 操作时出现错误（InvalidDocument）

问题：使用 AWS CLI 启动会话时，您会收到以下错误。

```
An error occurred (InvalidDocument) when calling the StartSession operation: Document
 type: 'Command' is not supported. Only type: 'Session' is supported for Session
 Manager.
```

解决方案：您为 `--document-name` 参数指定的 SSM 文档不是会话文档。请按照以下过程在 AWS Management Console 中查看会话文档列表。

查看会话文档列表

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择文档。
3. 在类别列表中，选择会话文档。

AWS Systems Manager State Manager

State Manager（AWS Systems Manager 中的一项工具）是一项安全并且可扩展的配置管理服务，可以自动将托管式节点和其他 AWS 资源保持在定义的状态。要开始使用 State Manager，请打开 Systems Manager 控制台。在导航窗格中，请选择 State Manager。

> **Note**
>
> State Manager 和 Maintenance Windows 可在托管式节点上执行一些相似类型的更新。您选择哪一项取决于您是需要自动执行系统合规性，还是在指定的时间段内执行高优先级、时效性强的任务。
> 有关更多信息，请参阅 在 State Manager 和 Maintenance Windows 之间选择。

我的组织如何从 State Manager 获益？

通过使用预配置的 Systems Manager 文档（SSM 文档），State Manager 可以为管理节点提供以下好处：

- 在启动时通过特定软件引导启动节点。
- 按照既定计划下载和更新代理，包括 SSM Agent。
- 配置网络设置。
- 将节点加入 Microsoft Active Directory 域。
- 在 Linux 、macOS 和 Windows 托管式节点的整个生命周期内对这些实例运行脚本。

要管理其他 AWS 资源的配置偏移，可搭配使用 Automation（Systems Manager 中的一项工具）与 State Manager来执行以下类型的任务：

- 将 Systems Manager 角色附上 Amazon Elastic Compute Cloud (Amazon EC2) 实例，以使其成为托管式节点。
- 对安全组强制实施所需的入口和出口规则。
- 创建或删除 Amazon DynamoDB 备份。
- 创建或删除 Amazon Elastic Block Store (Amazon EBS) 快照。
- 关闭 Amazon Simple Storage Service (Amazon S3) 存储桶的读写权限。
- 开启、重新开启或停止托管式节点和 Amazon Relational Database Service (Amazon RDS) 实例。
- 将修补程序应用到 Linux、macOS 和 Windows AMIs。

有关使用 State Manager 与自动化运行手册的信息，请参阅 使用 State Manager 关联调度自动化。

谁应该使用 State Manager？

State Manager 适用于任何希望改进其 AWS 资源管理和治理并减少配置偏移的 AWS 客户。

State Manager 具有哪些功能？

State Manager 的主要功能包括以下方面：

- State Manager 关联

  State Manager 关联是指分配给 AWS 资源的配置。该配置定义要在资源上保持的状态。例如，关联可以指定必须在托管式节点上安装并运行防病毒软件，或必须关闭特定端口。

关联指定了何时应用关联的配置和目标的计划。例如，防病毒软件的关联可能会每天在 AWS 账户中的所有托管式节点上运行一次。如果该软件未安装在节点上，则关联可以指示 State Manager 安装该软件。如果已安装该软件，但未运行服务，则关联可以指示 State Manager 开启服务。

- 灵活的调度选项

  State Manager 在关联运行时提供以下调度选项：

  - 立即或延迟处理

    在创建关联时，原定设置下，系统会立即在指定的资源上运行该关联。在初始运行后，关联会根据定义的计划按周期运行：

    通过使用控制台中的 Apply association only at the next specified Cron interval（仅在下一个指定的 Cron 周期应用关联）或命令行中的 `ApplyOnlyAtCronInterval` 参数，可以指示 State Manager 不要立即运行关联。

  - Cron 和 Rate 表达式

    创建关联时，您可以指定 State Manager 何时应用该配置的计划。State Manager 支持标准的 cron 和 rate 表达式，以便在关联运行时进行调度。State Manager 还支持使用星期几和数字符号（#）的 cron 表达式，以指定将在一个月的第 n 天运行某个关联，此外还支持在表达式中使用（L）符号来指示该月最后一个 X 天。

    > **Note**
    >
    > State Manager 目前不支持在 cron 表达式中为关联指定月数。

    要进一步控制关联运行的时间，例如，如果您想在星期二补丁后两天运行关联，可以指定偏移量。偏移量定义了在计划日期之后等待多少天再运行关联。

    有关构建 cron 和 rate 表达式的信息，请参阅 参考：适用于 Systems Manager 的 Cron 和 Rate 表达式。

- 多种目标选项

  关联还可以指定关联的目标。State Manager 支持通过使用标签、AWS Resource Groups、单个节点 ID 或当前 AWS 区域 和 AWS 账户 中的所有托管式节点来将 AWS 资源设为目标。

- Amazon S3 支持

将关联运行的命令输出存储在您选择的 Amazon S3 存储桶中。有关更多信息，请参阅 在 Systems Manager 中使用关联。。

- EventBridge 支持

  在 Amazon EventBridge 规则中，支持将此 Systems Manager 工具作为事件类型和目标类型。有关更多信息，请参阅 使用 Amazon EventBridge 监控 Systems Manager 事件 和 引用：Amazon EventBridge 事件模式和 Systems Manager 类型。

使用 State Manager 是否需要收取费用？

State Manager 不收取额外费用。

主题

- 了解 State Manager 的工作方式
- 在 Systems Manager 中使用关联。
- 创建运行 MOF 文件的关联
- 创建运行 Ansible Playbook 的关联
- 创建运行 Chef 配方的关联
- 演练：使用 AWS CLI 自动更新 SSM Agent
- 演练：在适用于 Windows Server 的 EC2 实例上自动更新半虚拟化驱动程序

更多信息

- 使用 Amazon EC2 Systems Manager 和 Windows PowerShell DSC 来防止配置偏差
- 使用 State Manager 在 Auto Scaling 组中配置 Amazon EC2 实例

了解 State Manager 的工作方式

State Manager（AWS Systems Manager 中的一项工具）是一项安全并且可扩展的服务，可以自动将混合和多云基础设施中的托管式节点保持在定义的状态。

State Manager 运行方式如下：

1. 确定要应用于 AWS 资源的状态。

   是否要确保您的托管式节点已配置特定应用程序，如防病毒或防恶意软件应用程序？ 是否要自动执行更新 SSM Agent 或其他 AWS 软件包（如 `AWSPVDriver`）的过程？ 是否需要确保特定端口已关闭或打开？ 要开启使用 State Manager，请确定要应用于 AWS 资源的状态。要应用的状态可决定您使用哪个 SSM 文档来创建 State Manager 关联。

   State Manager 关联是指分配给 AWS 资源的配置。该配置定义要在资源上保持的状态。例如，关联可以指定必须在托管式节点上安装并运行防病毒软件，或必须关闭特定端口。

   关联指定了何时应用关联的配置和目标的计划。例如，防病毒软件的关联可能会每天在 AWS 账户中的所有托管式节点上运行一次。如果该软件未安装在节点上，则关联可以指示 State Manager 安装该软件。如果已安装该软件，但未运行服务，则关联可以指示 State Manager 开启服务。

2. 确定预配置的 SSM 文档能否帮助您在 AWS 资源上创建所需的状态。

   Systems Manager 包含几十个预先配置的 SSM 文档，可用来创建关联。预先配置的文档已准备就绪，可执行常见任务，例如安装应用程序、配置 Amazon CloudWatch 、运行 AWS Systems Manager 自动化、运行 PowerShell 和 Shell 脚本、将托管式节点加入 Active Directory 的目录服务域等。

   您可以在 Systems Manager 控制台查看所有 SSM 文档。您可以选择文档的名称以了解更多有关每个文档的详情。以下是两个示例：`AWS-ConfigureAWSPackage` 和 `AWS-InstallApplication`。

3. 创建关联。

   您可以使用 Systems Manager 控制台、AWS Command Line Interface(AWS CLI)，AWS Tools for Windows PowerShell (Tools for Windows PowerShell) 或 Systems Manager API 创建关联。当创建关联时，需要指定以下信息：

   - 关联的名称。
   - SSM 文档的参数（例如，要在节点上安装的应用程序或要运行的脚本的路径）。
   - 关联目标。您可以通过指定标签、选择单个节点 ID 或在 AWS Resource Groups 中选择一个组来将托管式节点设为目标。您还可以将当前 AWS 区域 和 AWS 账户 中的所有托管式节点设为目标。如果目标包含超过 1000 个节点，系统将使用每小时节流机制。这意味着您可能会观察到状态聚合计数存在偏差，因为聚合过程每小时运行一次，且仅在节点的执行状态发生变化时才会运行。
   - 关于应用状态的时间或频率的计划。您可以指定 cron 或 rate 表达式。有关使用 cron 和 rate 表达式创建计划的更多信息，请参阅 适用于关联的 Cron 和 Rate 表达式。

> **Note**
>
> State Manager 目前不支持在 cron 表达式中为关联指定月数。

运行命令以创建关联时，Systems Manager 将指定的信息（计划、目标、SSM 文档和参数）绑定到目标资源。当系统尝试访问所有目标并立即 应用关联中指定的状态时，关联的状态最初会显示“待处理”。

> **Note**
>
> 如果您创建计划在之前的关联仍在运行时运行的新关联，那么之前的关联将超时，系统会运行新关联。

Systems Manager 会报告在资源中创建关联之请求的状态。您可以在控制台中或（对于托管式节点）使用 DescribeInstanceAssociationsStatus API 操作查看状态详细信息。如果在创建关联时选择将命令输出写入到 Amazon Simple Storage Service (Amazon S3)，则还可以在指定的 Amazon S3 存储桶中查看此输出。

有关更多信息，请参阅 在 Systems Manager 中使用关联。。

> **Note**
>
> 在关联运行期间由 SSM 文档发起的 API 操作未登录 AWS CloudTrail。

4. 监控并更新。

创建关联之后，State Manager 根据关联中定义的计划重新应用配置。您可以在控制台的 State Manager 页面上或通过直接调用在您创建关联时由 Systems Manager 生成的关联 ID 来查看关联的状态。有关更多信息，请参阅 查看关联历史记录。您可以更新您的关联文档并根据需要重新应用它们。您也可以创建多个关联版本。有关更多信息，请参阅 编辑和创建关联的新版本。

了解何时将关联应用于资源

在创建关联时，您需要指定用于定义配置的 SSM 文档、目标资源列表，以及应用配置的计划。默认情况下，State Manager 在您创建关联时运行该关联，然后根据您的计划运行该关联。State Manager 还会尝试在以下情况下运行关联：

- 关联编辑 – State Manager 在用户编辑并将其更改保存到以下任何关联字段 DOCUMENT_VERSION、PARAMETERS、SCHEDULE_EXPRESSION、OUTPUT_S3_LOCATION 中后运行关联。
- 文档编辑 – State Manager 在用户编辑并将更改保存到用于定义关联的配置状态的 SSM 文档中后运行关联。具体来说，关联在对文档进行以下编辑之后运行：
  - 用户指定了一个新的 $DEFAULT 文档版本，该关联使用 $DEFAULT 版本创建。
  - 用户更新文档，然后使用 $LATEST 版本创建了关联。
  - 用户删除创建关联时指定的文档。
- 手动启动 – State Manager 在用户从 Systems Manager 控制台启动或以编程的方式启动时运行关联。
- 目标更改 – State Manager 在目标节点上发生以下任何活动后运行关联：
  - 托管式节点第一次联机。
  - 托管式节点在错过了计划的关联运行后联机。
  - 托管式节点被停止超过 30 天后联机。

> **Note**
>
> State Manager不跨AWS 账户监视关联中使用的文档或包。如果您在一个账户中更新文档或包，则该更新不会导致在第二个账户中运行关联。您必须在第二个账户中手动运行关联。

防止关联在目标更改时运行

在某些情况下，您可能不希望关联在由托管节点组成的目标发生变化时运行，而只根据其指定的计划运行。

> **Note**
>
> 运行自动化运行手册会产生费用。如果与自动化运行手册的关联以您账户中的所有实例为目标，并且您定期启动大量实例，则运行手册将在启动时在每个实例上运行。这可能会导致自动化费用上升。

要防止某个关联在该关联的目标发生更改时运行，请选中仅在下一个指定的 cron 间隔内应用关联复选框。此复选框位于创建关联和编辑关联页面的指定计划区域。

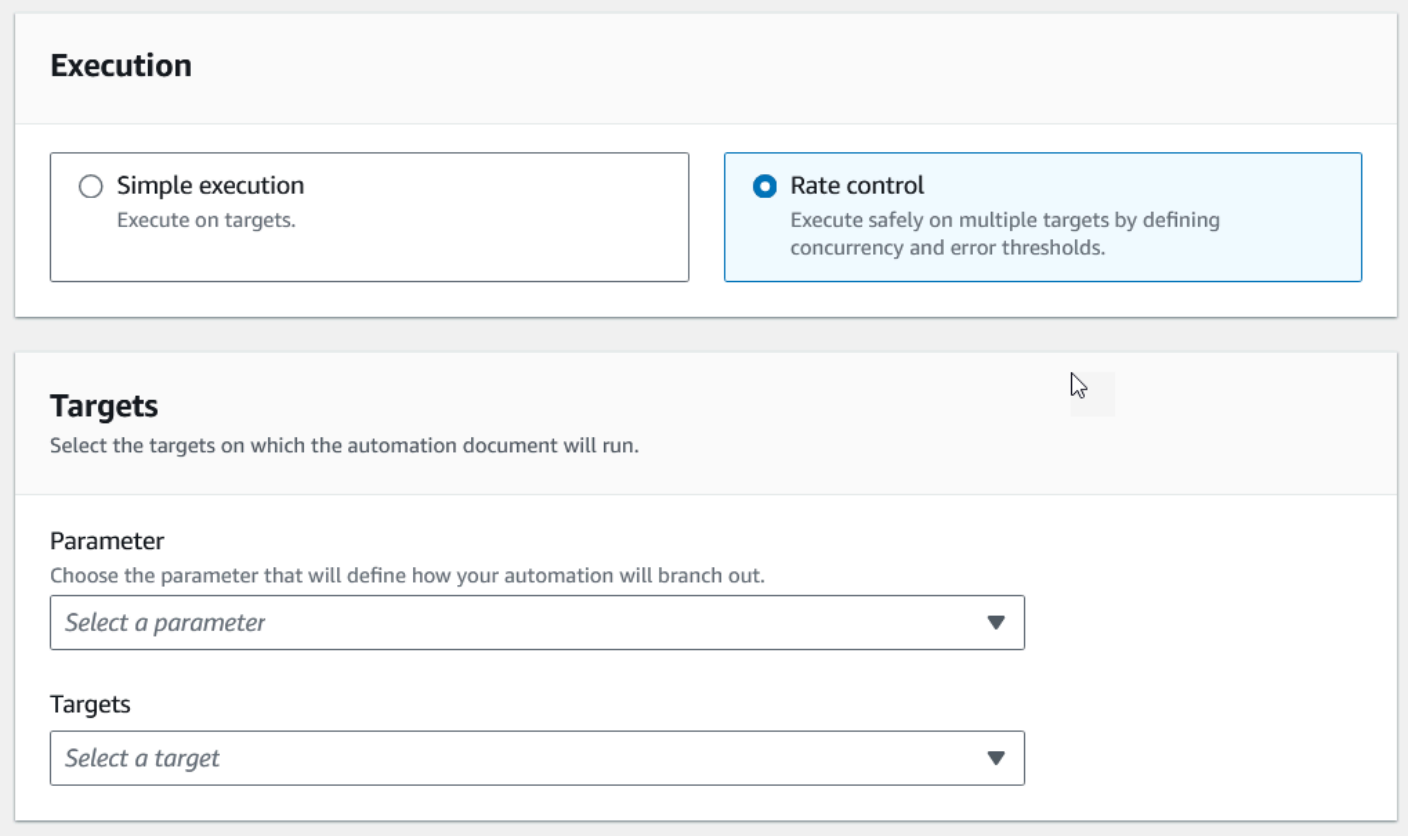
此选项适用于包含自动化运行手册或 SSM 文档的关联。

关于使用自动化运行手册进行目标更新

为了在检测到新的目标节点时应用使用自动化运行手册创建的关联，必须满足下面的条件：

- 关联必须由Quick Setup配置创建。Quick Setup是 AWS Systems Manager 中的一项工具。目前不支持由其他进程创建的关联。
- 自动化运行手册必须明确针对资源类型 `AWS::EC2::Instance` 或 `AWS::SSM::ManagedInstance`。
- 关联必须同时指定参数和目标。

在控制台中，当您选择一个速率控制执行时，系统会显示参数和目标字段。



当您使用 CreateAssociation、CreateAssociationBatch 或 UpdateAssociation API 操作时，您可以使用 AutomationTargetParameterName 和 Targets 输入指定这些值。在每个 API 操作中，您还可以通过将 ApplyOnlyAtCronInterval 参数设置为 true，防止关联在目标每次更改时运行。

有关使用控制台来控制关联何时运行的信息，包括避免自动化执行成本意外高昂的详细信息，请参阅了解何时将关联应用于资源。

在 Systems Manager 中使用关联。

本节将介绍如何使用 AWS Systems Manager 控制台、AWS Command Line Interface (AWS CLI) 和 AWS Tools for PowerShell 来创建和管理 State Manager 关联。

主题

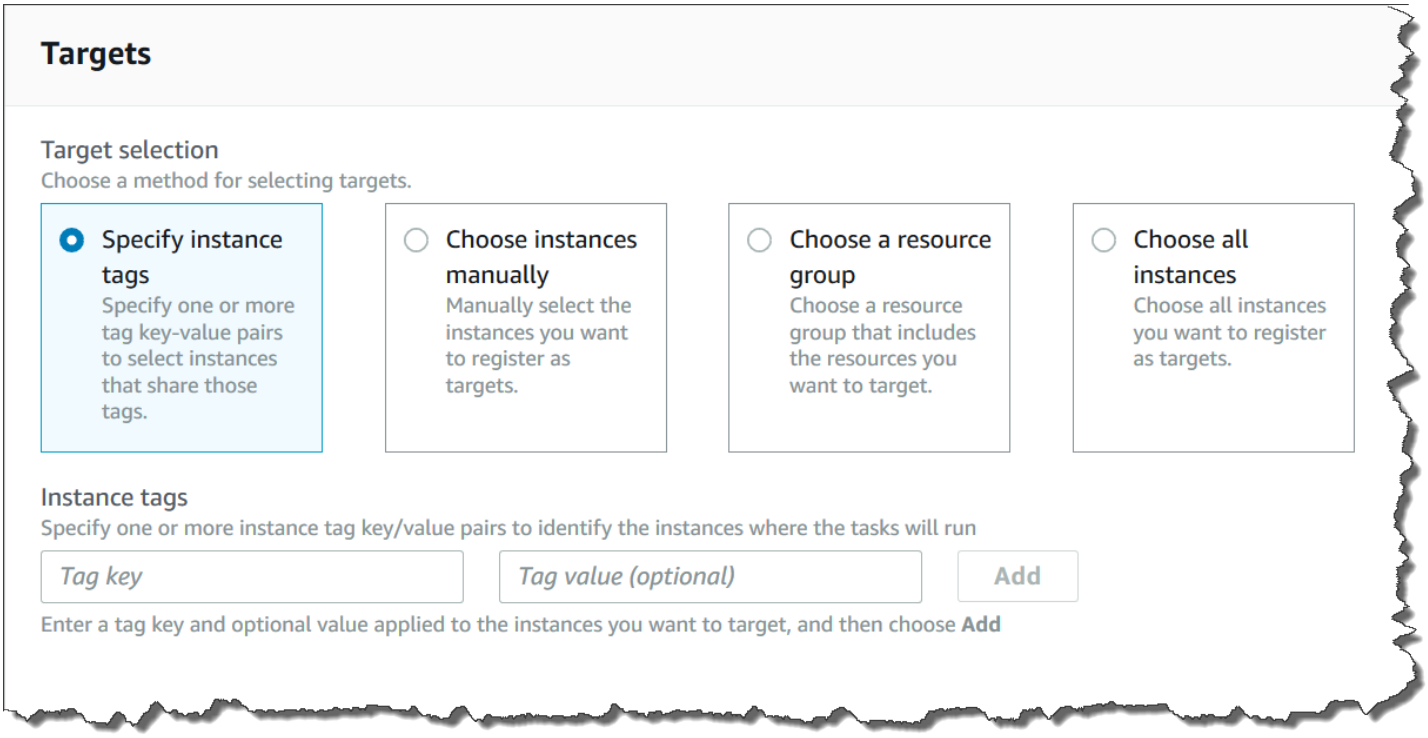
- 了解 State Manager 关联中的目标和速率控制
- 创建关联
- 编辑和创建关联的新版本
- 删除关联
- 运行 Auto Scaling 组
- 查看关联历史记录
- 使用 IAM 处理关联

了解 State Manager 关联中的目标和速率控制

本主题介绍State Manager工具。此工具可帮助您将关联部署到数十个或数百个节点，同时控制在计划时间运行关联的节点数。State Manager是 AWS Systems Manager 中的一项工具。

使用目标

在创建 State Manager 关联时，选择在 Systems Manager 控制台中的Targets（目标）部分中要配置哪些节点，如下所示。



如果使用命令行工具（如AWS Command Line Interface (AWS CLI)）创建关联，请指定 `targets` 参数。通过将节点设为目标，您可以配置数十个、数百个或数千个具有关联的节点，而无需指定或选择单个节点 ID。

每个托管节点最多可以有 20 个关联作为目标。

State Manager 在创建关联时包括以下目标选项。

指定标签

使用此选项可指定分配给节点的标签键和（可选）标签值。在您运行请求时，系统会在与指定标签键和值匹配的所有节点上查找并尝试创建关联。如果您指定了多个标签值，则关联将至少具有其中一个标签值的任何节点设为目标。当系统最初创建关联时，它将运行关联。在此初始运行之后，系统将根据您指定的计划运行关联。

如果您创建新节点并将指定的标签键和值分配给这些节点，系统会自动应用关联，立即运行它，然后根据计划运行它。这在关联使用命令或策略文档时适用，如果关联使用自动化运行手册，则不适用。如果从节点中删除指定的标签，系统将不再在这些节点上运行关联。

> **Note**
>
> 如果您将自动化运行手册与 State Manager 结合使用，但由于标记限制的原因，您无法实现特定目标，请考虑将自动化运行手册与 Amazon EventBridge 结合使用。有关更多信息，请参

阅 基于 EventBridge 事件运行自动化。有关将 State Manager 与自动化运行手册结合使用的信息，请参阅使用 State Manager 关联调度自动化。

作为最佳实践，我们建议在创建使用命令或策略文档的关联时使用标签。我们还建议在创建运行自动扩缩组的关联时使用标签。有关更多信息，请参阅 运行 Auto Scaling 组。

> **Note**
>
> 请注意以下信息。
>
> - 在 AWS Management Console中使用标签创建定位节点的关联时，只能为自动关联指定一个标签键，为命令关联指定五个标签键。关联中指定的所有标签键当前都必须分配给该节点。如果没有分配给该节点，则 State Manager 无法将该节点作为关联目标。
> - 如果要使用控制台，并且希望通过对自动关联使用多个标签键，对命令关联使用五个标签键来定位节点，请将标签键分配给 AWS Resource Groups 组并将节点添加到该组。然后，在创建 State Manager 关联时您可以在目标列表中选择资源组选项。
> - 您可以使用 AWS CLI 指定最多五个标签键。如果使用 AWS CLI，则 `create-association` 命令中指定的所有标签键当前都必须分配给该节点。如果没有分配给该节点，则 State Manager 无法将该节点作为关联目标。

手动选择节点

使用此选项可手动选择要在其中创建关联的节点。Instances（实例）窗格显示当前 AWS 账户 和 AWS 区域 中所有的 Systems Manager 托管式节点。您可以根据需要手动选择任意多个节点。当系统最初创建关联时，它将运行关联。在此初始运行之后，系统将根据您指定的计划运行关联。

> **Note**
>
> 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

选择资源组

使用此选项可在基于 AWS Resource Groups 标签或基于 AWS CloudFormation 堆栈的查询返回的所有节点上创建关联。

以下为有关将关联的 Resource Groups 设为目标的详细信息。

- 如果向组添加新节点，系统会自动将这些节点映射到将资源组设为目标的关联。当系统发现更改时，即会将关联应用于这些节点。在此初始运行之后，系统将根据您指定的计划运行关联。
- 如果您创建了以资源组为目标的关联，并且为该组指定了 `AWS::SSM::ManagedInstance` 资源类型，那么根据设计，该关联将在混合和多云环境中的 Amazon Elastic Compute Cloud（Amazon EC2）实例和非 EC2 节点上运行。

  反之亦然。如果您创建了以资源组为目标的关联，并且为该组指定了 `AWS::EC2::Instance` 资源类型，那么根据设计，该关联将同时在混合和多云环境中的非 EC2 节点和（Amazon EC2）实例上运行。
- 如果您创建了将某个资源组作为目标的关联，则分配给该资源组的标签键不得超过五个，或者为任何一个标签键指定的值不得超过五个。如果这些条件中的任何一个适用于分配给您的资源组的标签和键，则该关联将无法运行并返回 `InvalidTarget` 错误。
- 如果您使用标签创建以资源组为目标的关联，则无法为标签值选择（空值）选项。
- 如果删除资源组，则该组中的所有实例将不再运行关联。作为最佳实践，删除将该组设为目标的关联。
- 您只能将关联的单个资源组设为目标。不支持多个组或嵌套组。
- 创建关联后，State Manager 会使用资源组中资源的相关信息定期更新关联。如果您向资源组添加新资源，则系统何时将关联应用于新资源的计划取决于多个因素。您可以在 Systems Manager 控制台的 State Manager 页面上决定关联的状态。

> **Warning**
>
> 有权创建将 Amazon EC2 实例资源组设为目标的关联的 AWS Identity and Access Management (IAM) 用户、组或角色自动具有组中所有实例的根级控制权。应只允许受信任的管理员创建关联。

有关 Resource Groups 的详细信息，请参阅什么是 AWS Resource Groups？中的AWS Resource Groups 用户指南。

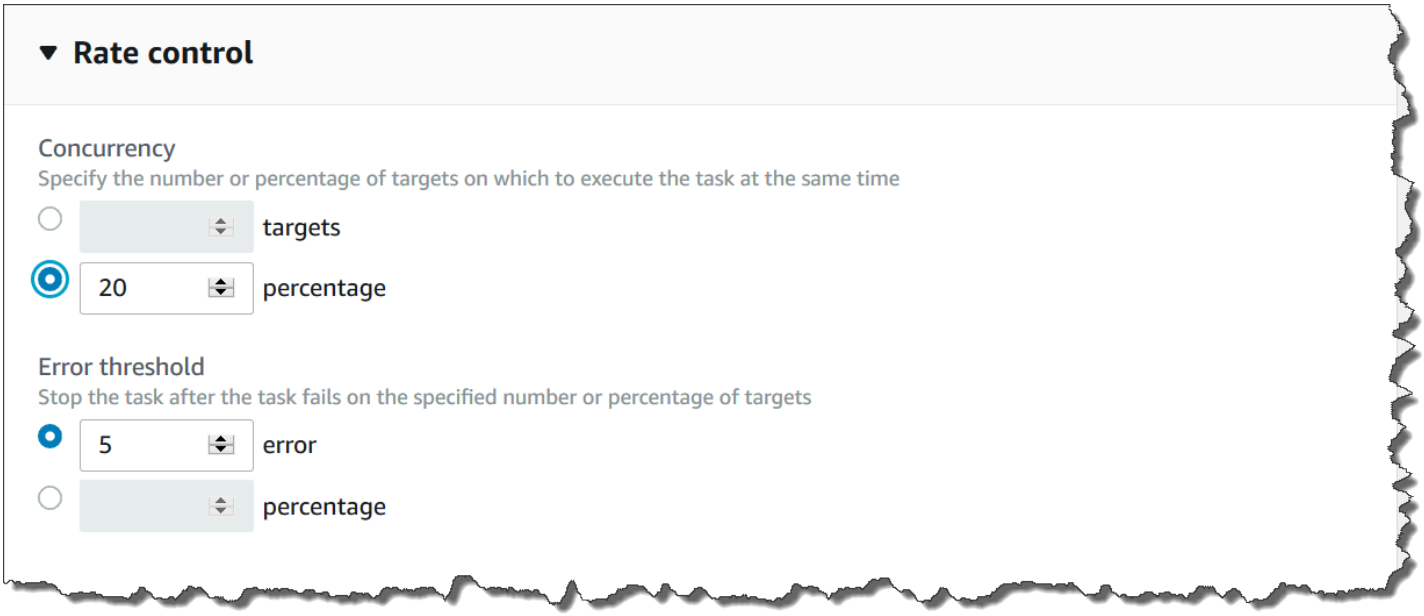
选择所有节点

使用此选项可将当前 AWS 账户 和 AWS 区域 中的所有节点设为目标。在您运行请求时，系统会在当前 AWS 账户 和 AWS 区域 中的所有节点上查找并尝试创建关联。当系统最初创建关联时，它将运行

关联。在此初始运行之后，系统将根据您指定的计划运行关联。如果您创建新节点，系统会自动应用关联，立即运行它，然后根据计划运行它。

使用速率控制

您可以通过指定并发值和错误阈值来控制关联在节点上的执行。并发值用于指定允许同时运行该关联的节点数量。错误阈值指定关联执行可以失败的次数，超过该次数后 Systems Manager 向已配置该关联的每个节点发送命令来停止运行该关联。此命令停止关联运行，直至下一次计划执行。并发和错误阈值功能统称为速率控制。



并发

并发通过允许您指定只有特定数量的节点可以一次处理一个关联，来帮助限制对节点的影响。您可以指定绝对数量的节点（例如 20），也可以指定目标节点集百分比（例如 10%）。

State Manager 并发具有以下限制：

- 如果选择使用目标创建关联，但不指定并发值，则State Manager 自动强制实施 50 个节点的最大并发。
- 如果在使用并发的关联运行时，与目标条件匹配的新节点变为线上状态，则新节点将在未超出并发值时运行关联。如果超出并发值，则会在当前的关联执行间隔内忽略这些节点。在满足并发要求的情况下，这些节点将在下一个计划间隔内运行关联。
- 如果更新一个使用并发的关联，并且在更新时有一个或多个节点在处理该关联，则允许正在运行该关联的任何节点完成。尚未开始的关联将停止。关联运行完成后，因为关联已更新，所以所有目标节点都将立即再次运行该关联。当关联再次运行时,将强制执行并发值。

错误阈值

错误阈值指定允许关联执行失败的次数，超过该次数后 Systems Manager 向已配置该关联的每个节点发送命令。此命令停止关联运行，直至下一次计划执行。您可以指定绝对数量的错误（如 10），也可以指定目标集百分比（如 10%）。

例如，如果指定错误的绝对数为 3，则 State Manager 将在返回第四个错误时发送停止命令。如果指定 0，则 State Manager 将在返回第一个错误结果后发送停止命令。

如果为 50 个关联指定错误阈值 10%，则 State Manager 将在返回第六个错误时发送停止命令。在达到错误阈值时，允许完成已经运行的关联，但是其中一些关联可能失败。要确保错误数不超过为错误阈值指定的数量，请将 Concurrency (并发) 值设置为 1，以便一次只处理一个关联。

State Manager 错误阈值具有以下限制：

- 为当前间隔强制实施错误阈值。
- 有关每个错误的信息（包括步骤级别详细信息）都记录在关联历史记录中。
- 如果选择使用目标创建关联，但不指定错误阈值，则 State Manager 自动强制实施 100% 失败的阈值。

创建关联

State Manager（AWS Systems Manager 中的一项工具）可以帮助您将 AWS 资源保持在您定义的状态，并减少配置偏移。为此，State Manager 会使用关联。关联是指分配给 AWS 资源的配置。该配置定义要在资源上保持的状态。例如，关联可以指定必须在托管式节点上安装并运行防病毒软件，或必须关闭特定端口。

关联指定了何时应用关联的配置和目标的计划。例如，防病毒软件的关联可能会每天在 AWS 账户中的所有托管式节点上运行一次。如果该软件未安装在节点上，则关联可以指示 State Manager 安装该软件。如果已安装该软件，但未运行服务，则关联可以指示 State Manager 开启服务。

> **Warning**
>
> 创建关联时，可以选择托管式节点的 AWS 资源组作为关联的目标。如果 AWS Identity and Access Management (IAM) 用户、组或角色有权限创建将托管式节点资源组设为目标的关联，则该用户、组或角色将自动具有组中所有节点的根级控制权。只允许受信任的管理员创建关联。

关联目标和速率控制

关联将指定哪些托管节点或目标应该接收关联。State Manager 包括的一些功能可帮助您将托管节点设置为目标，并控制如何将关联部署到这些目标。有关目标和速率控制的更多信息，请参阅 了解 State Manager 关联中的目标和速率控制。

为关联添加标签

通过使用命令行工具（如 AWS CLI 或 AWS Tools for PowerShell）创建关联时，可以将标签指定给关联。不支持使用 Systems Manager 控制台将标签添加到关联。

运行关联

默认情况下，State Manager 将在您创建关联后立即运行关联，然后根据您定义的计划运行。

该系统还会根据以下规则运行关联：

- State Manager 在间隔期间会尝试在所有指定或目标节点上运行关联。
- 如果在某个周期内未运行某一关联（例如，由于并发值限制了可以同时处理关联的节点数），State Manager 将尝试在下一个周期运行该关联。
- State Manager 将在更改关联的配置、目标节点、文档或参数后运行关联。有关更多信息，请参阅 了解何时将关联应用于资源。
- State Manager 会记录所有已跳过间隔的历史记录。可在 Execution History (执行历史记录) 选项卡上查看历史记录。

为关联制定计划

您可以安排关联以基本间隔（如每 10 小时）运行，也可以使用自定义 cron 和费率表达式创建更高级的计划。您还可以在首次创建关联时阻止关联运行。

使用 cron 和 rate 表达式为关联运行制定计划

除了标准 cron 和 rate 表达式外，State Manager 还支持多种 cron 表达式，其中包括在一周中的某一天运行关联，以及用于指定在一个月中的第 n 天运行关联的数字符号（#）。以下是一个在每月的第三个星期二 23:30 UTC 运行 cron 计划的示例：

```
cron(30 23 ? * TUE#3 *)
```

以下是一个在每月的第二个星期四在 UTC 午夜运行的示例：

```
cron(0 0 ? * THU#2 *)
```

State Manager 还支持 (L) 符号来表示一个月的最后一个 X 天。以下是一个在每月的最后一个星期二 UTC 午夜运行 cron 计划的示例：

```
cron(0 0 ? * 3L *)
```

要进一步控制关联运行的时间，例如，如果您想在星期二补丁后两天运行关联，可以指定偏移量。偏移量定义了在计划日期之后等待多少天再运行关联。例如，如果指定的 cron 计划为 `cron(0 0 ? * THU#2 *)`，则可以在 Schedule offset（计划偏移量）字段中指定数字 3，以便在每月的第二个星期四之后的每个星期天运行关联。

> **Note**
>
> 要使用偏移量，必须在控制台中选择仅在下一个指定的 Cron 周期应用关联，或从命令行指定 `ApplyOnlyAtCronInterval` 参数。在激活其中任一选项后，State Manager 不会在您创建关联后立即运行该关联。

有关 cron 和 rate 表达式的更多信息，请参阅 参考：适用于 Systems Manager 的 Cron 和 Rate 表达式。

创建关联（控制台）

以下过程介绍了如何使用 Systems Manager 控制台创建 State Manager 关联。

> **Note**
>
> 请注意以下信息。
>
> - 此过程描述如何创建使用 `Command` 或 `Policy` 文档来将托管式节点设为目标的关联。有关创建使用自动化运行手册以将节点或其他类型 AWS 资源设为目标的关联的信息，请参阅 使用 State Manager 关联调度自动化。
> - 创建关联时，您可以使用 AWS Management Console 指定最多五个标签键。当前必须将为关联指定的所有标签键分配给该节点。如果没有分配给该节点，则 State Manager 无法将该节点作为关联目标。

创建 State Manager 关联

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。

2. 在导航窗格中，请选择 State Manager。
3. 选择 Create association（创建关联）。
4. 在名称 字段中指定名称。
5. 在文档列表中，选择文档名称旁边的选项。请注意文档类型。此过程适用于 `Command` 和 `Policy` 文档。有关创建使用自动化运行手册的关联的信息，请参阅 使用 State Manager 关联调度自动化。

> **Important**
>
> 如果该文档是从其他账户共享的，则 State Manager 不支持运行使用该文档的新版本的关联。如果文档是共享自另一个账户，即使 Systems Manager 控制台显示已处理该文档的新版本，状态管理器仍会运行该文档的 `default` 版本。如果要使用共享自另一个账户的文档的新版本来运行关联，则必须将文档版本设置为 `default`。

6. 对于 Parameters (参数)，请指定所需的输入参数。
7. （可选）选择一个 CloudWatch 警报以应用于您的关联进行监控。

> **Note**
>
> 请注意与该步骤相关的以下信息。
>
> - 警报列表最多显示 100 个警报。如果您未在列表中看到您的警报，请使用 AWS Command Line Interface 创建关联。有关更多信息，请参阅 创建关联（命令行）。
> - 要将 CloudWatch 警报附加到命令，创建关联的 IAM 主体必须具有 `iam:createServiceLinkedRole` 操作的权限。有关 CloudWatch 警报的更多信息，请参阅使用 Amazon CloudWatch 警报。
> - 如果您的警报激活，任何待处理的命令调用或自动化都不会运行。

8. 对于 Targets (目标)，选择一个选项。有关使用目标的信息，请参阅 了解 State Manager 关联中的目标和速率控制。

> **Note**
>
> 为了在检测到新的目标节点时应用使用自动化运行手册创建的关联，必须满足某些条件。有关信息，请参阅关于使用自动化运行手册进行目标更新。

9. 在 Specify schedule (指定计划) 部分中，选择 On Schedule (按计划) 或 No schedule (无计划)。如果选择 On Schedule (按计划)，则可使用提供的按钮为关联创建 cron 或 rate 计划。

   如果您不希望关联在创建后立即运行，请选中 Apply association only at the next specified Cron interval (仅在下一个指定的 Cron 周期应用关联)。

10. （可选）在 Schedule offset（计划偏移量）字段中，指定一个介于 1 和 6 之间的数字。

11. 在高级选项部分使用合规性严重性选择关联的严重级别，然后使用更改日历选择关联的更改日历。

    合规性报告指示关联状态是合规还是不合规以及您在此处指示的严重级别。有关更多信息，请参阅关于 State Manager 关联合规性。

    更改日历确定关联何时运行。如果日历已关闭，则不应用关联。如果日历处于打开状态，则相应地运行关联。有关更多信息，请参阅 AWS Systems Manager Change Calendar。

12. 在 Rate control（速率控制）部分中，选择用于控制如何在多个节点上运行关联的选项。有关使用速率控制的更多信息，请参阅 了解 State Manager 关联中的目标和速率控制。

    在并发部分中，选择一个选项：

    - 选择 targets (目标) 输入可同时运行关联的目标的绝对数量。
    - 选择 percentage (百分比) 输入可同时运行关联的目标集的百分比。

    在错误阈值部分中，选择一个选项：

    - 选择 errors (错误) 以输入允许的错误绝对数量，超过该数量后 State Manager 停止对其他目标运行关联。
    - 选择 percentage (百分比) 以输入允许的错误百分比，超过该百分比后 State Manager 停止对其他目标运行关联。

13. （可选）对于 Output options (输出选项)，要将命令输出保存到文件，请选中 Enable writing output to S3 (启用将输出写入 S3) 方框。在方框中输入存储桶和前缀（文件夹）名称。

    > **Note**
    >
    > 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给托管式节点的实例配置文件的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确认与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

以下是为关联打开 Amazon S3 输出所需的最低权限。您可以通过将 IAM policy 附加到账户内的用户或角色，以进一步限制访问权限。Amazon EC2 实例配置文件至少应具有一个具有 `AmazonSSMManagedInstanceCore` 托管策略和以下内联策略的 IAM 角色。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:PutObjectAcl"
            ],
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
    ]
}
```

对于最低权限，导出到的 Amazon S3 存储桶必须具有 Amazon S3 控制台定义的默认设置。有关创建存储桶的更多信息，请转至 Amazon S3 用户指南中的创建存储桶。

> **Note**
>
> 在关联运行期间由 SSM 文档发起的 API 操作未登录 AWS CloudTrail。

14. 选择创建关联。

> **Note**
>
> 如果您删除已创建的关联，则该关联将不再在该关联的任何目标上运行。

创建关联（命令行）

以下过程介绍了如何使用 AWS CLI（在 Linux 或 Windows 上）或创建 State Manager 关联的 Tools for PowerShell。本节包括几个示例，说明如何使用目标和速率控制。通过目标和速率控制，您可以将关联分配给数十个或数百个节点，同时控制这些关联的执行。有关目标和速率控制的更多信息，请参阅[了解 State Manager 关联中的目标和速率控制](#)。

> **Important**
>
> 此过程描述如何创建使用 `Command` 或 `Policy` 文档来将托管式节点设为目标的关联。有关创建使用自动化运行手册以将节点或其他类型 AWS 资源设为目标的关联的信息，请参阅[使用 State Manager 关联调度自动化](#)。

开始前的准备工作

`targets` 参数是一组搜索条件，使用您指定的 `Key,Value` 组合将节点设为目标。如果您计划使用 `targets` 参数在数十个或数百个节点上创建关联，请在开始该过程之前查看以下设置目标选项。

通过指定 ID 将特定节点设为目标

```
--targets Key=InstanceIds,Values=instance-id-1,instance-id-2,instance-id-3
```

```
--targets
 Key=InstanceIds,Values=i-02573cafcfEXAMPLE,i-0471e04240EXAMPLE,i-07782c72faEXAMPLE
```

通过使用标签将实例设为目标

```
--targets Key=tag:tag-key,Values=tag-value-1,tag-value-2,tag-value-3
```

```
--targets Key=tag:Environment,Values=Development,Test,Pre-production
```

通过使用 AWS Resource Groups 将节点设为目标

```
--targets Key=resource-groups:Name,Values=resource-group-name
```

```
--targets Key=resource-groups:Name,Values=WindowsInstancesGroup
```

在当前 AWS 账户 和 AWS 区域 将所有的实例设为目标

```
--targets Key=InstanceIds,Values=*
```

> **Note**
>
> 请注意以下信息。
>
> - 如果该文档是从其他账户共享的，则 State Manager 不支持运行使用该文档的新版本的关联。如果文档是共享自另一个账户，即使 Systems Manager 控制台显示已处理该文档的新版本，状态管理器仍会运行该文档的 `default` 版本。如果要使用共享自另一个账户的文档的新版本来运行关联，则必须将文档版本设置为 `default`。
> - State Manager 不支持 TargetLocation 采用 `IncludeChildOrganizationUnits` 参数。
> - 您可以使用 AWS CLI 指定最多五个标签键。如果使用 AWS CLI，则 `create-association` 命令中指定的所有标签键当前都必须分配给该节点。如果没有分配给该节点，则 State Manager 无法将该节点作为关联目标。
> - 当创建关联时，需要指定计划运行的时间。使用 cron 或 rate 表达式指定计划。有关 cron 和 rate 表达式的更多信息，请参阅 适用于关联的 Cron 和 Rate 表达式。
> - 为了在检测到新的目标节点时应用使用自动化运行手册创建的关联，必须满足某些条件。有关信息，请参阅关于使用自动化运行手册进行目标更新。

创建关联

1. 安装并配置 AWS CLI 或 AWS Tools for PowerShell（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本以及安装 AWS Tools for PowerShell。

2. 使用以下格式创建一个命令以创建 State Manager 关联。将每个*#######*替换为您自己的信息。

   Linux & macOS

   ```
   aws ssm create-association \
       --name document_name \
       --document-version version_of_document_applied \
       --instance-id instances_to_apply_association_on \
       --parameters (if any) \
       --targets target_options \
       --schedule-expression "cron_or_rate_expression" \
       --apply-only-at-cron-interval required_parameter_for_schedule_offsets \
       --schedule-offset number_between_1_and_6 \
   ```

```
    --output-location s3_bucket_to_store_output_details \
    --association-name association_name \
    --max-errors a_number_of_errors_or_a_percentage_of_target_set \
    --max-concurrency a_number_of_instances_or_a_percentage_of_target_set \
    --compliance-severity severity_level \
    --calendar-names change_calendar_names \
    --target-locations aws_region_or_account \
    --tags "Key=tag_key,Value=tag_value"
```

Windows

```
aws ssm create-association ^
    --name document_name ^
    --document-version version_of_document_applied ^
    --instance-id instances_to_apply_association_on ^
    --parameters (if any) ^
    --targets target_options ^
    --schedule-expression "cron_or_rate_expression" ^
    --apply-only-at-cron-interval required_parameter_for_schedule_offsets ^
    --schedule-offset number_between_1_and_6 ^
    --output-location s3_bucket_to_store_output_details ^
    --association-name association_name ^
    --max-errors a_number_of_errors_or_a_percentage_of_target_set ^
    --max-concurrency a_number_of_instances_or_a_percentage_of_target_set ^
    --compliance-severity severity_level ^
    --calendar-names change_calendar_names ^
    --target-locations aws_region_or_account ^
    --tags "Key=tag_key,Value=tag_value"
```

PowerShell

```
New-SSMAssociation `
    -Name document_name `
    -DocumentVersion version_of_document_applied `
    -InstanceId instances_to_apply_association_on `
    -Parameters (if any) `
    -Target target_options `
    -ScheduleExpression "cron_or_rate_expression" `
    -ApplyOnlyAtCronInterval required_parameter_for_schedule_offsets `
    -ScheduleOffSet number_between_1_and_6 `
    -OutputLocation s3_bucket_to_store_output_details `
    -AssociationName association_name `
    -MaxError  a_number_of_errors_or_a_percentage_of_target_set
```

```
    -MaxConcurrency a_number_of_instances_or_a_percentage_of_target_set `
    -ComplianceSeverity severity_level `
    -CalendarNames change_calendar_names `
    -TargetLocations aws_region_or_account `
    -Tags "Key=tag_key,Value=tag_value"
```

以下示例在已贴标签 "Environment,Linux" 的节点上创建关联。该关联在每个星期日 2:00 UTC 时使用 AWS-UpdateSSMAgent 文档更新目标节点上的 SSM Agent。该关联在任意给定时间最多在 10 个节点上同时运行。此外，当错误计数超过 5 时，将在特定执行间隔内停止在更多节点上运行该关联。对于合规性报告，为该关联分配中等严重级别。

Linux & macOS

```
aws ssm create-association \
  --association-name Update_SSM_Agent_Linux \
  --targets Key=tag:Environment,Values=Linux \
  --name AWS-UpdateSSMAgent  \
  --compliance-severity "MEDIUM" \
  --schedule-expression "cron(0 2 ? * SUN *)" \
  --max-errors "5" \
  --max-concurrency "10"
```

Windows

```
aws ssm create-association ^
  --association-name Update_SSM_Agent_Linux ^
  --targets Key=tag:Environment,Values=Linux ^
  --name AWS-UpdateSSMAgent  ^
  --compliance-severity "MEDIUM" ^
  --schedule-expression "cron(0 2 ? * SUN *)" ^
  --max-errors "5" ^
  --max-concurrency "10"
```

PowerShell

```
New-SSMAssociation `
  -AssociationName Update_SSM_Agent_Linux `
  -Name AWS-UpdateSSMAgent `
  -Target @{
      "Key"="tag:Environment"
```

```
      "Values"="Linux"
    } `
  -ComplianceSeverity MEDIUM `
  -ScheduleExpression "cron(0 2 ? * SUN *)" `
  -MaxConcurrency 10 `
  -MaxError 5
```

以下示例通过指定通配符值 (*) 来将节点 ID 指定为目标。这允许 Systems Manager 在当前 AWS 账户 和 AWS 区域 中的所有节点上创建关联。该关联在任意给定时间最多在 10 个节点上同时运行。此外，当错误计数超过 5 时，将在特定执行间隔内停止在更多节点上运行该关联。对于合规性报告，为该关联分配中等严重级别。此关联使用了计划偏移量，这意味着将在指定的 cron 计划后两天运行。其还包括 ApplyOnlyAtCronInterval 参数，该参数是使用计划偏移量所必需的，这意味着关联在创建后不会立即运行。

Linux & macOS

```
aws ssm create-association \
  --association-name Update_SSM_Agent_Linux \
  --name "AWS-UpdateSSMAgent" \
  --targets "Key=instanceids,Values=*" \
  --compliance-severity "MEDIUM" \
  --schedule-expression "cron(0 2 ? * SUN#2 *)" \
  --apply-only-at-cron-interval \
  --schedule-offset 2 \
  --max-errors "5" \
  --max-concurrency "10" \
```

Windows

```
aws ssm create-association ^
  --association-name Update_SSM_Agent_Linux ^
  --name "AWS-UpdateSSMAgent" ^
  --targets "Key=instanceids,Values=*" ^
  --compliance-severity "MEDIUM" ^
  --schedule-expression "cron(0 2 ? * SUN#2 *)" ^
  --apply-only-at-cron-interval ^
  --schedule-offset 2 ^
  --max-errors "5" ^
  --max-concurrency "10" ^
```

```
  --apply-only-at-cron-interval
```

PowerShell

```
New-SSMAssociation `
  -AssociationName Update_SSM_Agent_All `
  -Name AWS-UpdateSSMAgent `
  -Target @{
      "Key"="InstanceIds"
      "Values"="*"
    } `
  -ScheduleExpression "cron(0 2 ? * SUN#2 *)" `
  -ApplyOnlyAtCronInterval `
  -ScheduleOffset 2 `
  -MaxConcurrency 10 `
  -MaxError 5 `
  -ComplianceSeverity MEDIUM `
  -ApplyOnlyAtCronInterval
```

以下示例在 Resource Groups 中的节点上创建关联。该组名为“HR-Department”。该关联在每个星期日 2:00 UTC 时使用 AWS-UpdateSSMAgent 文档更新目标节点上的 SSM Agent。该关联在任意给定时间最多在 10 个节点上同时运行。此外，当错误计数超过 5 时，将在特定执行间隔内停止在更多节点上运行该关联。对于合规性报告，为该关联分配中等严重级别。此关联按指定的 cron 计划运行。它不会在创建后立即运行。

Linux & macOS

```
aws ssm create-association \
  --association-name Update_SSM_Agent_Linux \
  --targets Key=resource-groups:Name,Values=HR-Department \
  --name AWS-UpdateSSMAgent  \
  --compliance-severity "MEDIUM" \
  --schedule-expression "cron(0 2 ? * SUN *)" \
  --max-errors "5" \
  --max-concurrency "10" \
  --apply-only-at-cron-interval
```

Windows

```
aws ssm create-association ^
```

```
  --association-name Update_SSM_Agent_Linux ^
  --targets Key=resource-groups:Name,Values=HR-Department ^
  --name AWS-UpdateSSMAgent  ^
  --compliance-severity "MEDIUM" ^
  --schedule-expression "cron(0 2 ? * SUN *)" ^
  --max-errors "5" ^
  --max-concurrency "10" ^
  --apply-only-at-cron-interval
```

PowerShell

```
New-SSMAssociation `
  -AssociationName Update_SSM_Agent_Linux `
  -Name AWS-UpdateSSMAgent `
  -Target @{
      "Key"="resource-groups:Name"
      "Values"="HR-Department"
    } `
  -ScheduleExpression "cron(0 2 ? * SUN *)" `
  -MaxConcurrency 10 `
  -MaxError 5 `
  -ComplianceSeverity MEDIUM `
  -ApplyOnlyAtCronInterval
```

以下示例创建一个关联，该关联在已标记特定节点 ID 的节点上运行。当更改日历处于打开状态时，该关联使用 SSM Agent 文档在目标节点上更新 SSM Agent。关联在运行时检查日历状态。如果日历在启动时关闭，并且关联仅运行一次，则不会再次运行，因为关联运行窗口已通过。如果日历处于打开状态，则相应地运行关联。

> **Note**
>
> 如果在更改日历关闭时将新节点添加到关联所依据的标签或 resource groups，则在更改日历打开后，关联将应用于这些节点。

Linux & macOS

```
aws ssm create-association \
  --association-name CalendarAssociation \
  --targets "Key=instanceids,Values=i-0cb2b964d3e14fd9f" \
```

```
  --name AWS-UpdateSSMAgent  \
  --calendar-names "arn:aws:ssm:us-east-1:123456789012:document/testCalendar1" \
  --schedule-expression "rate(1day)"
```

Windows

```
aws ssm create-association ^
  --association-name CalendarAssociation ^
  --targets "Key=instanceids,Values=i-0cb2b964d3e14fd9f" ^
  --name AWS-UpdateSSMAgent  ^
  --calendar-names "arn:aws:ssm:us-east-1:123456789012:document/testCalendar1" ^
  --schedule-expression "rate(1day)"
```

PowerShell

```
New-SSMAssociation `
  -AssociationName CalendarAssociation `
  -Target @{
      "Key"="tag:instanceids"
      "Values"="i-0cb2b964d3e14fd9f"
    } `
  -Name AWS-UpdateSSMAgent `
  -CalendarNames "arn:aws:ssm:us-east-1:123456789012:document/testCalendar1" `
  -ScheduleExpression "rate(1day)"
```

以下示例创建一个关联，该关联在已标记特定节点 ID 的节点上运行。该关联在每个星期日 2:00 时使用 SSM Agent 文档更新目标节点上的 SSM Agent。此关联仅在更改日历打开时按指定的 cron 计划运行。创建关联时，它会检查日历状态。如果日历已关闭，则不应用关联。当应用关联的时间间隔从星期日凌晨 2:00 开始时，关联将检查日历是否处于打开状态。如果日历处于打开状态，则相应地运行关联。

> **Note**
>
> 如果在更改日历关闭时将新节点添加到关联所依据的标签或 resource groups，则在更改日历打开后，关联将应用于这些节点。

Linux & macOS

```
aws ssm create-association \
  --association-name MultiCalendarAssociation \
  --targets "Key=instanceids,Values=i-0cb2b964d3e14fd9f" \
  --name AWS-UpdateSSMAgent  \
  --calendar-names "arn:aws:ssm:us-east-1:123456789012:document/testCalendar1"
 "arn:aws:ssm:us-east-2:123456789012:document/testCalendar2" \
  --schedule-expression "cron(0 2 ? * SUN *)"
```

Windows

```
aws ssm create-association ^
  --association-name MultiCalendarAssociation ^
  --targets "Key=instanceids,Values=i-0cb2b964d3e14fd9f" ^
  --name AWS-UpdateSSMAgent  ^
  --calendar-names "arn:aws:ssm:us-east-1:123456789012:document/testCalendar1"
 "arn:aws:ssm:us-east-2:123456789012:document/testCalendar2" ^
  --schedule-expression "cron(0 2 ? * SUN *)"
```

PowerShell

```
New-SSMAssociation `
  -AssociationName MultiCalendarAssociation `
  -Name AWS-UpdateSSMAgent `
  -Target @{
      "Key"="tag:instanceids"
      "Values"="i-0cb2b964d3e14fd9f"
    } `
  -CalendarNames "arn:aws:ssm:us-east-1:123456789012:document/testCalendar1"
 "arn:aws:ssm:us-east-2:123456789012:document/testCalendar2" `
  -ScheduleExpression "cron(0 2 ? * SUN *)"
```

Note

如果您删除已创建的关联，则该关联将不再在该关联的任何目标上运行。此外，如果指定了 `apply-only-at-cron-interval` 参数，还可以重置此选项。要执行此操作，请在从命令行

更新关联时指定 `no-apply-only-at-cron-interval` 参数。此参数会强制关联在更新后立即运行，以及按照指定的周期运行。

编辑和创建关联的新版本

您可以编辑 State Manager 关联以指定新名称、计划、严重级别、目标或其他值。对于基于 SSM 命令类型文档的关联，您也可以选择将命令输出写入 Amazon Simple Storage Service（Amazon S3）存储桶。编辑关联后，State Manager 将创建新版本。您可以在编辑后查看不同的版本，如以下过程中所述。

> **Note**
>
> 为了在检测到新的目标节点时应用使用自动化运行手册创建的关联，必须满足某些条件。有关信息，请参阅关于使用自动化运行手册进行目标更新。

以下过程介绍了如何使用 Systems Manager 控制台、AWS Command Line Interface (AWS CLI) 和 AWS Tools for PowerShell (Tools for PowerShell) 编辑和创建关联的新版本。

> **Important**
>
> 如果该文档是从其他账户共享的，则 State Manager 不支持运行使用该文档的新版本的关联。如果文档是从其他账户共享的，则 State Manager 始终会运行文档的 `default` 版本，即使 Systems Manager 控制台显示已处理了该文档的新版本。如果要使用共享自另一个账户的文档的新版本来运行关联，则必须将文档版本设置为 `default`。

编辑关联（控制台）

以下过程介绍了如何使用 Systems Manager 控制台编辑和创建关联的新版本。

> **Note**
>
> 对于使用 SSM 命令文档而非自动化运行手册的关联，此过程要求您具有对现有 Amazon S3 存储桶的写入权限。如果您之前未使用 Amazon S3，请注意使用 Amazon S3 会产生费用。有关如何创建存储桶的信息，请参阅创建存储桶。

编辑 State Manager 关联

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 State Manager。
3. 选择现有的关联，然后选择编辑。
4. 重新配置关联以满足当前要求。

   有关与 Command 和 Policy 文档的关联选项的信息，请参阅 创建关联。有关与自动化运行手册的关联选项的信息，请参阅 使用 State Manager 关联调度自动化。
5. 选择保存更改。
6. （可选）若要查看关联信息，在关联页面中，选择已编辑的关联的名称，然后选择版本选项卡。系统将列出您已创建和编辑的关联的每个版本。
7. （可选）若要查看基于 SSM Command 文档的关联输出，请执行以下操作：

   a. 通过以下网址打开 Amazon S3 控制台：https://console.aws.amazon.com/s3/。
   b. 选择您指定用于存储命令输出的 Simple Storage Service (Amazon S3) 存储桶的名称，然后选择以运行关联的节点的 ID 命名的文件夹。(如果您选择将输出存储在存储桶中的文件夹内，请先打开它。)
   c. 下拉多个级别至 stdout 文件夹中的 awsrunPowerShell 文件。
   d. 选择打开或下载查看主机名。

编辑关联（命令行）

以下过程介绍了如何使用 AWS CLI（在 Linux 或 Windows 上）或 AWS Tools for PowerShell 编辑和创建关联的新版本。

编辑 State Manager 关联

1. 安装并配置 AWS CLI 或 AWS Tools for PowerShell（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本以及安装 AWS Tools for PowerShell。
2. 使用以下格式创建一个命令，以编辑和创建现有 State Manager 关联的新版本。将每个######替换为您自己的信息。

> **Important**
>
> 当您调用 update-association 时，系统会从请求中删除所有可选参数，并用这些参数的 null 值覆盖关联。这是设计使然。即使没有更改参数，您也必须在调用中指定所有可选参数，包括 --name 参数。在调用此操作之前，建议先调用 describe-association 操作，并记下 update-association 调用所需的所有可选参数。

Linux & macOS

```
aws ssm update-association \
    --name document_name \
    --document-version version_of_document_applied \
    --instance-id instances_to_apply_association_on \
    --parameters (if any) \
    --targets target_options \
    --schedule-expression "cron_or_rate_expression" \
    --schedule-offset "number_between_1_and_6" \
    --output-location s3_bucket_to_store_output_details \
    --association-name association_name \
    --max-errors a_number_of_errors_or_a_percentage_of_target_set \
    --max-concurrency a_number_of_instances_or_a_percentage_of_target_set \
    --compliance-severity severity_level \
    --calendar-names change_calendar_names \
    --target-locations aws_region_or_account
```

Windows

```
aws ssm update-association ^
    --name document_name ^
    --document-version version_of_document_applied ^
    --instance-id instances_to_apply_association_on ^
    --parameters (if any) ^
    --targets target_options ^
    --schedule-expression "cron_or_rate_expression" ^
    --schedule-offset "number_between_1_and_6" ^
    --output-location s3_bucket_to_store_output_details ^
    --association-name association_name ^
    --max-errors a_number_of_errors_or_a_percentage_of_target_set ^
    --max-concurrency a_number_of_instances_or_a_percentage_of_target_set ^
```

```
    --compliance-severity severity_level ^
    --calendar-names change_calendar_names ^
    --target-locations aws_region_or_account
```

PowerShell

```
Update-SSMAssociation `
    -Name document_name `
    -DocumentVersion version_of_document_applied `
    -InstanceId instances_to_apply_association_on `
    -Parameters (if any) `
    -Target target_options `
    -ScheduleExpression "cron_or_rate_expression" `
    -ScheduleOffset "number_between_1_and_6" `
    -OutputLocation s3_bucket_to_store_output_details `
    -AssociationName association_name `
    -MaxError  a_number_of_errors_or_a_percentage_of_target_set
    -MaxConcurrency a_number_of_instances_or_a_percentage_of_target_set `
    -ComplianceSeverity severity_level `
    -CalendarNames change_calendar_names `
    -TargetLocations aws_region_or_account
```

以下示例更新现有关联以将名称更改为 TestHostnameAssociation2。新的关联版本每小时运行一次，并将命令输出写入到指定的 Amazon S3 存储桶中。

Linux & macOS

```
aws ssm update-association \
  --association-id 8dfe3659-4309-493a-8755-01234EXAMPLE \
  --association-name TestHostnameAssociation2 \
  --parameters commands="echo Association" \
  --output-location S3Location='{OutputS3Region=us-
east-1,OutputS3BucketName=amzn-s3-demo-bucket,OutputS3KeyPrefix=logs}' \
  --schedule-expression "cron(0 */1 * * ? *)"
```

Windows

```
aws ssm update-association ^
  --association-id 8dfe3659-4309-493a-8755-01234EXAMPLE ^
  --association-name TestHostnameAssociation2 ^
  --parameters commands="echo Association" ^
```

```
  --output-location S3Location='{OutputS3Region=us-
east-1,OutputS3BucketName=amzn-s3-demo-bucket,OutputS3KeyPrefix=logs}' ^
  --schedule-expression "cron(0 */1 * * ? *)"
```

PowerShell

```
Update-SSMAssociation `
  -AssociationId b85ccafe-9f02-4812-9b81-01234EXAMPLE `
  -AssociationName TestHostnameAssociation2 `
  -Parameter @{"commands"="echo Association"} `
  -S3Location_OutputS3BucketName amzn-s3-demo-bucket `
  -S3Location_OutputS3KeyPrefix logs `
  -S3Location_OutputS3Region us-east-1 `
  -ScheduleExpression "cron(0 */1 * * ? *)"
```

以下示例更新现有关联以将名称更改为 CalendarAssociation。新关联在日历打开时运行，并将命令输出写入指定的 Amazon S3 存储桶。

Linux & macOS

```
aws ssm update-association \
  --association-id 8dfe3659-4309-493a-8755-01234EXAMPLE \
  --association-name CalendarAssociation \
  --parameters commands="echo Association" \
  --output-location S3Location='{OutputS3Region=us-
east-1,OutputS3BucketName=amzn-s3-demo-bucket,OutputS3KeyPrefix=logs}' \
  --calendar-names "arn:aws:ssm:us-east-1:123456789012:document/testCalendar2"
```

Windows

```
aws ssm update-association ^
  --association-id 8dfe3659-4309-493a-8755-01234EXAMPLE ^
  --association-name CalendarAssociation ^
  --parameters commands="echo Association" ^
  --output-location S3Location='{OutputS3Region=us-
east-1,OutputS3BucketName=amzn-s3-demo-bucket,OutputS3KeyPrefix=logs}' ^
  --calendar-names "arn:aws:ssm:us-east-1:123456789012:document/testCalendar2"
```

PowerShell

```
Update-SSMAssociation `
  -AssociationId b85ccafe-9f02-4812-9b81-01234EXAMPLE `
  -AssociationName CalendarAssociation `
  -AssociationName OneTimeAssociation `
  -Parameter @{"commands"="echo Association"} `
  -S3Location_OutputS3BucketName amzn-s3-demo-bucket `
  -CalendarNames "arn:aws:ssm:us-east-1:123456789012:document/testCalendar2"
```

以下示例更新现有关联以将名称更改为 MultiCalendarAssociation。当日历打开并将命令输出写入指定的 Amazon S3 存储桶时，新关联将运行。

Linux & macOS

```
aws ssm update-association \
  --association-id 8dfe3659-4309-493a-8755-01234EXAMPLE \
  --association-name MultiCalendarAssociation \
  --parameters commands="echo Association" \
  --output-location S3Location='{OutputS3Region=us-
east-1,OutputS3BucketName=amzn-s3-demo-bucket,OutputS3KeyPrefix=logs}' \
  --calendar-names "arn:aws:ssm:us-east-1:123456789012:document/testCalendar1"
 "arn:aws:ssm:us-east-2:123456789012:document/testCalendar2"
```

Windows

```
aws ssm update-association ^
  --association-id 8dfe3659-4309-493a-8755-01234EXAMPLE ^
  --association-name MultiCalendarAssociation ^
  --parameters commands="echo Association" ^
  --output-location S3Location='{OutputS3Region=us-
east-1,OutputS3BucketName=amzn-s3-demo-bucket,OutputS3KeyPrefix=logs}' ^
  --calendar-names "arn:aws:ssm:us-east-1:123456789012:document/testCalendar1"
 "arn:aws:ssm:us-east-2:123456789012:document/testCalendar2"
```

PowerShell

```
Update-SSMAssociation `
  -AssociationId b85ccafe-9f02-4812-9b81-01234EXAMPLE `
  -AssociationName MultiCalendarAssociation `
```

```
  -Parameter @{"commands"="echo Association"} `
  -S3Location_OutputS3BucketName amzn-s3-demo-bucket `
  -CalendarNames "arn:aws:ssm:us-east-1:123456789012:document/testCalendar1"
 "arn:aws:ssm:us-east-2:123456789012:document/testCalendar2"
```

3. 要查看关联的新版本，请运行以下命令。

Linux & macOS

```
aws ssm describe-association \
  --association-id b85ccafe-9f02-4812-9b81-01234EXAMPLE
```

Windows

```
aws ssm describe-association ^
  --association-id b85ccafe-9f02-4812-9b81-01234EXAMPLE
```

PowerShell

```
Get-SSMAssociation `
  -AssociationId b85ccafe-9f02-4812-9b81-01234EXAMPLE | Select-Object *
```

系统将返回类似于以下内容的信息。

Linux & macOS

```
{
    "AssociationDescription": {
        "ScheduleExpression": "cron(0 */1 * * ? *)",
        "OutputLocation": {
            "S3Location": {
                "OutputS3KeyPrefix": "logs",
                "OutputS3BucketName": "amzn-s3-demo-bucket",
                "OutputS3Region": "us-east-1"
            }
        },
        "Name": "AWS-RunPowerShellScript",
        "Parameters": {
            "commands": [
                "echo Association"
            ]
```

```
        },
        "LastExecutionDate": 1559316400.338,
        "Overview": {
            "Status": "Success",
            "DetailedStatus": "Success",
            "AssociationStatusAggregatedCount": {}
        },
        "AssociationId": "b85ccafe-9f02-4812-9b81-01234EXAMPLE",
        "DocumentVersion": "$DEFAULT",
        "LastSuccessfulExecutionDate": 1559316400.338,
        "LastUpdateAssociationDate": 1559316389.753,
        "Date": 1559314038.532,
        "AssociationVersion": "2",
        "AssociationName": "TestHostnameAssociation2",
        "Targets": [
            {
                "Values": [
                    "Windows"
                ],
                "Key": "tag:Environment"
            }
        ]
    }
}
```

Windows

```
{
    "AssociationDescription": {
        "ScheduleExpression": "cron(0 */1 * * ? *)",
        "OutputLocation": {
            "S3Location": {
                "OutputS3KeyPrefix": "logs",
                "OutputS3BucketName": "amzn-s3-demo-bucket",
                "OutputS3Region": "us-east-1"
            }
        },
        "Name": "AWS-RunPowerShellScript",
        "Parameters": {
            "commands": [
                "echo Association"
            ]
        },
```

```
        "LastExecutionDate": 1559316400.338,
        "Overview": {
            "Status": "Success",
            "DetailedStatus": "Success",
            "AssociationStatusAggregatedCount": {}
        },
        "AssociationId": "b85ccafe-9f02-4812-9b81-01234EXAMPLE",
        "DocumentVersion": "$DEFAULT",
        "LastSuccessfulExecutionDate": 1559316400.338,
        "LastUpdateAssociationDate": 1559316389.753,
        "Date": 1559314038.532,
        "AssociationVersion": "2",
        "AssociationName": "TestHostnameAssociation2",
        "Targets": [
            {
                "Values": [
                    "Windows"
                ],
                "Key": "tag:Environment"
            }
        ]
    }
}
```

PowerShell

```
AssociationId                : b85ccafe-9f02-4812-9b81-01234EXAMPLE
AssociationName              : TestHostnameAssociation2
AssociationVersion           : 2
AutomationTargetParameterName :
ComplianceSeverity           :
Date                         : 5/31/2019 2:47:18 PM
DocumentVersion              : $DEFAULT
InstanceId                   :
LastExecutionDate            : 5/31/2019 3:26:40 PM
LastSuccessfulExecutionDate  : 5/31/2019 3:26:40 PM
LastUpdateAssociationDate    : 5/31/2019 3:26:29 PM
MaxConcurrency               :
MaxErrors                    :
Name                         : AWS-RunPowerShellScript
OutputLocation               :
 Amazon.SimpleSystemsManagement.Model.InstanceAssociationOutputLocation
```

```
Overview                        :
 Amazon.SimpleSystemsManagement.Model.AssociationOverview
Parameters                      : {[commands,
 Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
ScheduleExpression              : cron(0 */1 * * ? *)
Status                          :
Targets                         : {tag:Environment}
```

删除关联

可以使用以下过程，通过使用 AWS Systems Manager 控制台删除关联。

删除关联

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 State Manager。
3. 选择某一关联，然后选择删除。

通过从 AWS Systems Manager 控制台运行自动化，可以在单个操作中删除多个关联。当您选择多个关联进行删除时，State Manager 会启动自动化运行手册起始页面，并将关联 ID 作为输入参数值输入。

要在单个操作中删除多个关联

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 State Manager。
3. 选择要删除的每个关联，然后选择删除。
4. （可选）在其他输入参数区域中，选择您希望自动化在运行时使用的代入角色的 Amazon 资源名称（ARN）。要创建新的代入角色，请选择创建。
5. 选择提交。

运行 Auto Scaling 组

使用关联运行 Auto Scaling 组时的最佳做法是使用标记目标。不使用标签可能会导致您达到关联限制。

如果所有节点都使用相同的键和值进行标记，则您只需要一个关联即可运行 Auto Scaling 组。以下过程介绍了如何创建关联。

创建运行 Auto Scaling 组的关联

1. 确保 Auto Scaling 组中的所有节点都使用相同的键和值进行标记。有关标签节点的更多说明，请参阅AWS Auto Scaling 用户指南中的标记 Auto Scaling 组和实例。
2. 通过使用 在 Systems Manager 中使用关联。 中的过程来创建关联。

   如果您在控制台中使用，请选择目标字段中的指定实例标签返回的子位置类型。对于实例标签，请输入 Auto Scaling 组的标记键和值。

   如果您使用的是AWS Command Line Interface(AWS CLI)，请指定其中键和值与您标记节点的内容相匹配的`--targets Key=tag:tag-key,Values=tag-value`。

查看关联历史记录

您可以使用 DescribeAssociationExecutions API 操作查看特定关联 ID 的所有执行。使用此操作可查看State Manager关联的状态、详细状态、结果、上次执行时间以及其他信息。State Manager是 AWS Systems Manager 中的一项工具。此 API 操作还包括筛选条件，帮助您根据指定的条件快速找到关联。例如，您可以指定一个确切的日期和时间，并使用 GREATER_THAN 筛选条件以查看在该指定的日期和时间之后处理的那些执行。

例如，如果一个关联执行失败，则可使用 DescribeAssociationExecutionTargets API 操作来深入了解特定执行的详细信息。此操作将为您显示资源，例如节点 ID、关联运行的位置和各种关联状态。您可以查看哪些资源或节点无法运行关联。利用资源 ID，您随后可以查看命令执行详细信息，以了解命令中的哪个步骤已失败。

本节中的示例还包括有关如何在创建时使用 StartAssociationsOnce API 操作运行一次关联的信息。在调查失败的关联执行时，可以使用此 API 操作。如果发现一个关联失败，则可以在资源上进行更改，然后立即运行关联以查看对资源所做的更改是否可让关联成功运行。

> **Note**
>
> 在关联运行期间由 SSM 文档发起的 API 操作未登录 AWS CloudTrail。

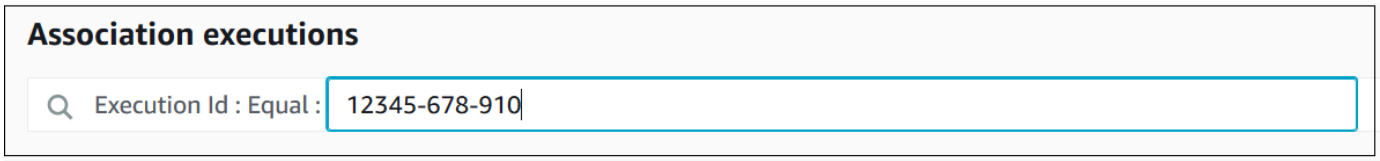
查看关联历史记录（控制台）

使用以下过程可查看特定关联 ID 的执行历史记录，然后查看一个或多个资源的执行详细信息。

查看特定关联 ID 的执行历史记录

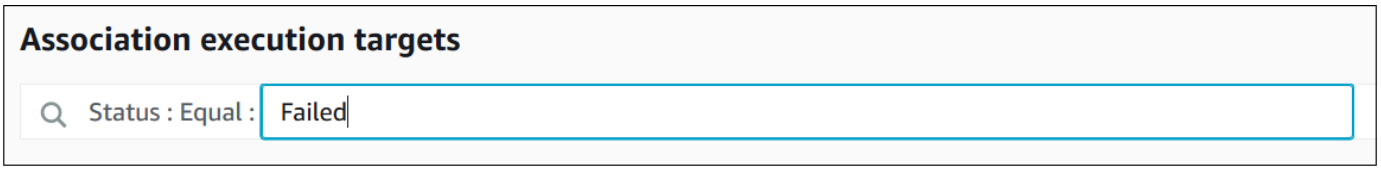
1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 选择 State Manager。
3. 在 Association id (关联 ID) 字段中，选择要查看其历史记录的关联。
4. 选择 View details (查看详细信息) 按钮。
5. 选择 Execution history (执行历史记录) 选项卡。
6. 选择要查看其资源级别执行详细信息的关联。例如，选择显示 Failed (失败) 状态的关联。然后，您可以查看无法运行关联的节点的执行详细信息。

   使用搜索框筛选条件以找到要查看其详细信息的执行。

7. 选择执行 ID。此时将打开 Association execution targets (关联执行目标) 页面。此页面显示所有已运行关联的资源。
8. 选择一个资源 ID 以查看有关此资源的特定信息。

   使用搜索框筛选条件以找到要查看其详细信息的资源。

9. 如果您正在调查无法运行的关联，则在创建时可使用 Apply association now (立即应用关联) 按钮来运行一次关联。在对无法运行关联的资源进行更改后，在导航位置提示中选择 Association ID (关联 ID) 链接。
10. 选择 Apply association now (立即应用关联) 按钮。在执行完成后，验证关联执行是否成功。

查看关联历史记录（命令行）

以下过程介绍了如何使用 AWS Command Line Interface (AWS CLI)（在 Linux 或 Windows 上）或 AWS Tools for PowerShell 查看特定关联 ID 的执行历史记录。然后，该过程介绍了如何查看一个或多个资源的执行详细信息。

查看特定关联 ID 的执行历史记录

1. 安装并配置 AWS CLI 或 AWS Tools for PowerShell（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本以及安装 AWS Tools for PowerShell。

2. 运行以下命令以查看特定关联 ID 的执行列表。

   Linux & macOS

   ```
   aws ssm describe-association-executions \
     --association-id ID \
     --filters Key=CreatedTime,Value="2018-04-10T19:15:38.372Z",Type=GREATER_THAN
   ```

   > **Note**
   >
   > 此命令包含一个筛选条件，可将结果限定为仅在特定日期和时间之后发生的那些执行。如果要查看特定关联 ID 的所有执行，请删除 `--filters` 参数和 `Key=CreatedTime,Value="2018-04-10T19:15:38.372Z",Type=GREATER_THAN` 值。

   Windows

   ```
   aws ssm describe-association-executions ^
     --association-id ID ^
     --filters Key=CreatedTime,Value="2018-04-10T19:15:38.372Z",Type=GREATER_THAN
   ```

   > **Note**
   >
   > 此命令包含一个筛选条件，可将结果限定为仅在特定日期和时间之后发生的那些执行。如果要查看特定关联 ID 的所有执行，请删除 `--filters` 参数和 `Key=CreatedTime,Value="2018-04-10T19:15:38.372Z",Type=GREATER_THAN` 值。

   PowerShell

   ```
   Get-SSMAssociationExecution `
   ```

```
  -AssociationId ID `
  -Filter
@{"Key"="CreatedTime";"Value"="2019-06-01T19:15:38.372Z";"Type"="GREATER_THAN"}
```

> **Note**
>
> 此命令包含一个筛选条件，可将结果限定为仅在特定日期和时间之后发生的那些执行。如果要查看特定关联 ID 的所有执行，请删除 `-Filter` 参数和 `@{"Key"="CreatedTime";"Value"="2019-06-01T19:15:38.372Z";"Type"="GREAT` 值。

系统将返回类似于以下内容的信息。

Linux & macOS


```
{
   "AssociationExecutions":[
      {
         "Status":"Success",
         "DetailedStatus":"Success",
         "AssociationId":"c336d2ab-09de-44ba-8f6a-6136cEXAMPLE",
         "ExecutionId":"76a5a04f-caf6-490c-b448-92c02EXAMPLE",
         "CreatedTime":1523986028.219,
         "AssociationVersion":"1"
      },
      {
         "Status":"Success",
         "DetailedStatus":"Success",
         "AssociationId":"c336d2ab-09de-44ba-8f6a-6136cEXAMPLE",
         "ExecutionId":"791b72e0-f0da-4021-8b35-f95dfEXAMPLE",
         "CreatedTime":1523984226.074,
         "AssociationVersion":"1"
      },
      {
         "Status":"Success",
         "DetailedStatus":"Success",
         "AssociationId":"c336d2ab-09de-44ba-8f6a-6136cEXAMPLE",
         "ExecutionId":"ecec60fa-6bb0-4d26-98c7-140308EXAMPLE",
         "CreatedTime":1523982404.013,
         "AssociationVersion":"1"
```

```
      }
   ]
}
```

Windows

```
{
   "AssociationExecutions":[
      {
         "Status":"Success",
         "DetailedStatus":"Success",
         "AssociationId":"c336d2ab-09de-44ba-8f6a-6136cEXAMPLE",
         "ExecutionId":"76a5a04f-caf6-490c-b448-92c02EXAMPLE",
         "CreatedTime":1523986028.219,
         "AssociationVersion":"1"
      },
      {
         "Status":"Success",
         "DetailedStatus":"Success",
         "AssociationId":"c336d2ab-09de-44ba-8f6a-6136cEXAMPLE",
         "ExecutionId":"791b72e0-f0da-4021-8b35-f95dfEXAMPLE",
         "CreatedTime":1523984226.074,
         "AssociationVersion":"1"
      },
      {
         "Status":"Success",
         "DetailedStatus":"Success",
         "AssociationId":"c336d2ab-09de-44ba-8f6a-6136cEXAMPLE",
         "ExecutionId":"ecec60fa-6bb0-4d26-98c7-140308EXAMPLE",
         "CreatedTime":1523982404.013,
         "AssociationVersion":"1"
      }
   ]
}
```

PowerShell

```
AssociationId         : c336d2ab-09de-44ba-8f6a-6136cEXAMPLE
AssociationVersion    : 1
CreatedTime           : 8/18/2019 2:00:50 AM
DetailedStatus        : Success
ExecutionId           : 76a5a04f-caf6-490c-b448-92c02EXAMPLE
LastExecutionDate     : 1/1/0001 12:00:00 AM
```

```
ResourceCountByStatus : {Success=1}
Status                : Success

AssociationId         : c336d2ab-09de-44ba-8f6a-6136cEXAMPLE
AssociationVersion    : 1
CreatedTime           : 8/11/2019 2:00:54 AM
DetailedStatus        : Success
ExecutionId           : 791b72e0-f0da-4021-8b35-f95dfEXAMPLE
LastExecutionDate     : 1/1/0001 12:00:00 AM
ResourceCountByStatus : {Success=1}
Status                : Success

AssociationId         : c336d2ab-09de-44ba-8f6a-6136cEXAMPLE
AssociationVersion    : 1
CreatedTime           : 8/4/2019 2:01:00 AM
DetailedStatus        : Success
ExecutionId           : ecec60fa-6bb0-4d26-98c7-140308EXAMPLE
LastExecutionDate     : 1/1/0001 12:00:00 AM
ResourceCountByStatus : {Success=1}
Status                : Success
```

您可以使用一个或多个筛选条件来限制结果。以下示例返回在特定日期和时间之前运行的所有关联。

Linux & macOS

```
aws ssm describe-association-executions \
  --association-id ID \
  --filters Key=CreatedTime,Value="2018-04-10T19:15:38.372Z",Type=LESS_THAN
```

Windows

```
aws ssm describe-association-executions ^
  --association-id ID ^
  --filters Key=CreatedTime,Value="2018-04-10T19:15:38.372Z",Type=LESS_THAN
```

PowerShell

```
Get-SSMAssociationExecution `
  -AssociationId 14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE `
```

```
  -Filter
 @{"Key"="CreatedTime";"Value"="2019-06-01T19:15:38.372Z";"Type"="LESS_THAN"}
```

下面返回在特定日期和时间之后成功 运行的所有关联。

Linux & macOS

```
aws ssm describe-association-executions \
  --association-id ID \
  --filters Key=CreatedTime,Value="2018-04-10T19:15:38.372Z",Type=GREATER_THAN
 Key=Status,Value=Success,Type=EQUAL
```

Windows

```
aws ssm describe-association-executions ^
  --association-id ID ^
  --filters Key=CreatedTime,Value="2018-04-10T19:15:38.372Z",Type=GREATER_THAN
 Key=Status,Value=Success,Type=EQUAL
```

PowerShell

```
Get-SSMAssociationExecution `
  -AssociationId 14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE `
  -Filter @{
      "Key"="CreatedTime";
      "Value"="2019-06-01T19:15:38.372Z";
      "Type"="GREATER_THAN"
    },
    @{
      "Key"="Status";
      "Value"="Success";
      "Type"="EQUAL"
    }
```

3. 运行以下命令可查看已运行特定执行的所有目标。

Linux & macOS

```
aws ssm describe-association-execution-targets \
  --association-id ID \
```

```
  --execution-id ID
```

Windows

```
aws ssm describe-association-execution-targets ^
  --association-id ID ^
  --execution-id ID
```

PowerShell

```
Get-SSMAssociationExecutionTarget `
  -AssociationId 14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE `
  -ExecutionId 76a5a04f-caf6-490c-b448-92c02EXAMPLE
```

您可以使用一个或多个筛选条件来限制结果。以下示例返回有关所有无法运行特定关联的目标的信息。

Linux & macOS

```
aws ssm describe-association-execution-targets \
  --association-id ID \
  --execution-id ID \
  --filters Key=Status,Value="Failed"
```

Windows

```
aws ssm describe-association-execution-targets ^
  --association-id ID ^
  --execution-id ID ^
  --filters Key=Status,Value="Failed"
```

PowerShell

```
Get-SSMAssociationExecutionTarget `
  -AssociationId 14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE `
  -ExecutionId 76a5a04f-caf6-490c-b448-92c02EXAMPLE `
  -Filter @{
      "Key"="Status";
      "Value"="Failed"
```

```
    }
```

以下示例返回有关无法运行关联的特定托管式节点的信息。

Linux & macOS

```
aws ssm describe-association-execution-targets \
  --association-id ID \
  --execution-id ID \
  --filters Key=Status,Value=Failed Key=ResourceId,Value="i-02573cafcfEXAMPLE"
 Key=ResourceType,Value=ManagedInstance
```

Windows

```
aws ssm describe-association-execution-targets ^
  --association-id ID ^
  --execution-id ID ^
  --filters Key=Status,Value=Failed Key=ResourceId,Value="i-02573cafcfEXAMPLE"
 Key=ResourceType,Value=ManagedInstance
```

PowerShell

```
Get-SSMAssociationExecutionTarget `
  -AssociationId 14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE `
  -ExecutionId 76a5a04f-caf6-490c-b448-92c02EXAMPLE `
  -Filter @{
      "Key"="Status";
      "Value"="Success"
    },
    @{
      "Key"="ResourceId";
      "Value"="i-02573cafcfEXAMPLE"
    },
    @{
      "Key"="ResourceType";
      "Value"="ManagedInstance"
    }
```

4. 如果您正在调查无法运行的关联，则可使用 StartAssociationsOnce API 操作来立即运行且仅运行一次关联。在更改关联无法运行的资源后，运行以下命令以立即运行关联，并且仅运行一次。

Linux & macOS

```
aws ssm start-associations-once \
  --association-id ID
```

Windows

```
aws ssm start-associations-once ^
  --association-id ID
```

PowerShell

```
Start-SSMAssociationsOnce `
  -AssociationId ID
```

使用 IAM 处理关联

State Manager（AWS Systems Manager 中的一项工具）使用目标选择要使用哪些实例配置关联。最初，关联是通过指定文档名称（Name）和实例 ID（InstanceId)创建的。这会在文档和实例或托管式节点之间创建关联。通过这些参数标识的关联。这些参数现在已弃用，但仍受支持。资源 instance 和 managed-instance 作为资源添加到具有 Name 和 InstanceId 的操作中。

AWS Identity and Access Management (IAM) 策略实施行为取决于指定的资源类型。State Manager 操作的资源仅根据传入的请求强制执行。State Manager 不会对账户中的资源属性进行深入检查。只有当请求参数包含指定的策略资源时，才会根据策略资源验证请求。例如，如果您在资源块中指定了一个实例，则如果请求使用 InstanceId 参数，策略会强制执行。不会为InstanceId 检查账户中每项资源的 Targets 参数。

以下是一些具有混淆的行为的情况：

- DescribeAssociation、DeleteAssociation 和 UpdateAssociation 使用 instance、managed-instance 和 document 资源来指定已弃用的引用关联的方式。这包括所有使用已弃用使用的 InstanceId 参数创建的关联。
- CreateAssociation、CreateAssociationBatch 和 UpdateAssociation 使用 instance 和 managed-instance 资源来指定已弃用的引用关联的方式。这包括所有使用已弃用的 InstanceId 参数创建的关联。document 资源类型是已弃用的引用关联方式的一部分，并且是关联的实际属性。这意味着您可以根据文档名称为 Create 和 Update 操作构建具有 Allow 或 Deny 权限的 IAM 策略。

有关将 IAM 策略与 Systems Manager 配合使用的更多信息，请参阅 服务授权参考中的 对 AWS Systems Manager 进行身份和访问管理 或者操作、资源和条件键 AWS Systems Manager。

创建运行 MOF 文件的关联

您可以使用 AWS-ApplyDSCMofs SSM 文档运行托管式对象格式（MOF）文件，通过State Manager（AWS Systems Manager 中的一项工具）在 Windows Server 托管式节点上强制执行期望状态。AWS-ApplyDSCMofs 文档有两种执行模式。在第一种模式下，您可以配置关联来扫描并报告托管式节点是否处于指定 MOF 文件中定义的期望状态。在第二种模式下，您可以运行 MOF 文件并根据 MOF 文件中定义的资源及其值更改节点的配置。AWS-ApplyDSCMofs 文档允许您从 Amazon Simple Storage Service (Amazon S3)、本地共享或具有 HTTPS 域的安全网站下载和运行 MOF 配置文件。

State Manager 在每个关联运行期间记录并报告每个 MOF 文件执行的状态。State Manager 还将每个 MOF 文件执行的输出报告为合规性事件，可在 AWS Systems Manager 合规性页面上查看该事件。

MOF 文件执行基于 Windows PowerShell Desired State Configuration(PowerShell DSC) 构建而成。PowerShell DSC 是一个声明性平台，用于配置、部署和管理 Windows 系统。PowerShell DSC 允许管理员用名为 DSC 配置的简单文本文档来描述其需要的服务器配置方式。PowerShell DSC 配置是一个专门的 PowerShell 脚本，该脚本描述任务是什么但不描述如何执行任务。运行配置会生成一个 MOF 文件。该 MOF 文件可以应用于一台或多台服务器，以在这些服务器中实现所需配置。PowerShell DSC 资源执行强制实施配置的实际工作。有关更多信息，请参阅 Windows PowerShell Desired State Configuration Overview。

主题

- 使用 Amazon S3 存储构件
- 解析 MOF 文件中的凭证
- 在 MOF 文件中使用令牌
- 创建运行 MOF 文件的关联的先决条件
- 创建运行 MOF 文件的关联
- 排查创建运行 MOF 文件的关联时出现的问题
- 查看 DSC 资源合规性详细信息

使用 Amazon S3 存储构件

如果使用 Amazon S3 存储 PowerShell 模块、MOF 文件、合规性报告或状态报告，则 AWS Systems Manager SSM Agent使用的 AWS Identity and Access Management (IAM) 角色必须拥有存储桶的

GetObject 和 ListBucket 权限。如果您不提供这些权限，系统将返回拒绝访问 错误。以下是有关在 Amazon S3 中存储构件的重要信息。

- 如果存储桶位于不同的 AWS 账户 中，请创建一个存储桶资源策略，向该账户（或 IAM 角色）授予 GetObject 和 ListBucket 权限。
- 如果要使用自定义 DSC 资源，则可从 Amazon S3 存储桶中下载这些资源。您也可以从 PowerShell 库中自动安装这些资源。
- 如果使用 Amazon S3 作为模块源，请以 Zip 文件形式上传该模块，采用以下区分大小写的格式：*ModuleName_ModuleVersion*.zip。例如：MyModule_1.0.0.zip。
- 所有文件都必须位于存储桶根中。不支持文件夹结构。

解析 MOF 文件中的凭证

可使用 AWS Secrets Manager 或 AWS Systems Manager Parameter Store 来解析凭证。这样，可以设置自动凭证轮换。DSC 也可以自动将凭证传播到服务器，而不必重新部署 MOF。

要在配置中使用 AWS Secrets Manager 密钥，请创建一个 PSCredential 对象，其中用户名是包含凭证的密钥的 SecretId 或 SecretARN。可为密码指定任意值。该值将被忽略。以下为示例。

```
Configuration MyConfig
{
   $ss = ConvertTo-SecureString -String 'a_string' -AsPlaintext -Force
   $credential = New-Object PSCredential('a_secret_or_ARN', $ss)

    Node localhost
    {
       File file_name
       {
           DestinationPath = 'C:\MyFile.txt'
           SourcePath = '\\FileServer\Share\MyFile.txt'
           Credential = $credential
       }
    }
}
```

请使用配置数据中的 PsAllowPlaintextPassword 设置来编译 MOF。因为凭证只包含一个标签，所以可以这样做。

在 Secrets Manager 中，确保节点具有 IAM 托管式策略和（可选的）密钥资源策略（如果存在）中的 GetSecretValue 访问权限。要使用 DSC，密钥必须采用以下格式。

```
{ 'Username': 'a_name', 'Password': 'a_password' }
```

密钥可以具有其他属性（例如，用于轮换的属性），但它至少需具有用户名和密码属性。

建议使用多用户轮换方法，这样，您有两组不同的用户名和密码，轮换 AWS Lambda 函数会轮流使用这两组用户名和密码。此方法允许您有多个活动账户，消除了在轮换期间锁定用户的风险。

在 MOF 文件中使用令牌

通过令牌可以在编译 MOF 之后 修改资源属性值。这样，您可以在需要相似配置的多台服务器上重用通用 MOF 文件。

令牌替换仅适用于 `String` 类型的资源属性。但是，如果资源具有嵌套的 CIM 节点属性，它也会从该 CIM 节点的 `String` 属性中解析令牌。不能将令牌替换用于数字或数组。

例如，考虑以下场景：您使用的是 xComputerManagement 资源，并且您希望使用 DSC 重命名计算机。通常，您需要对该计算机使用一个专用的 MOF 文件。但是，有了令牌支持，您可以创建一个 MOF 文件并将其应用于您的所有节点。在 `ComputerName` 属性中，不必将计算机名称硬编码到 MOF 中，您可以使用实例标签类型的令牌。在 MOF 解析期间会解析此值。请参阅以下示例。

```
Configuration MyConfig
{
    xComputer Computer
    {
        ComputerName = '{tag:ComputerName}'
    }
}
```

然后，您可以在 Systems Manager 控制台中的托管式节点上设置标签，或在 Amazon EC2 控制台中设置 Amazon Elastic Compute Cloud (Amazon EC2) 标签。运行该文档时，脚本将 {tag:ComputerName} 令牌替换为实例标签的值。

您还可以将多个标签组合到单个属性中，如下面的示例所示。

```
Configuration MyConfig
{
    File MyFile
    {
        DestinationPath = '{env:TMP}\{tag:ComputerName}'
        Type = 'Directory'
```

```
    }
}
```

您可以使用以下 5 种不同类型的令牌：

- 标签：Amazon EC2 或托管式节点标签。
- tagb64：与标签相同，但系统使用 base64 对值解码。这样您可以在标签值中使用特殊字符。
- env：解析环境变量。
- ssm：Parameter Store 值。仅支持字符串和安全字符串类型。
- tagssm：与标签相同，但如果未在节点上设置标签，系统将尝试从同名的 Systems Manager 参数中解析值。如果需要“原定设置全局值”，但希望能够在单个节点（例如，一站式部署）中覆盖该值时，此令牌很有用。

以下是使用 `ssm` 令牌类型的 Parameter Store 示例。

```
File MyFile
{
    DestinationPath = "C:\ProgramData\ConnectionData.txt"
    Content = "{ssm:%servicePath%/ConnectionData}"
}
```

令牌通过使 MOF 文件变为通用和可重用，在减少冗余代码方面发挥了重要作用。如果可以避免特定于服务器的 MOF 文件，则无需 MOF 构建服务。MOF 构建服务会增加成本、减少调配时间并增加分组节点之间因编译其 MOF 时在编译服务器上安装的模块版本不同而产生配置偏差的风险。

创建运行 MOF 文件的关联的先决条件

在创建用于运行 MOF 文件的关联之前，请验证托管式节点是否已安装以下必备组件：

- Windows PowerShell 5.0 或更高版本。有关更多信息，请参阅 Microsoft.com 上的 Windows PowerShell 系统要求。
- AWS Tools for Windows PowerShell 1.5.13 版或更高版本。
- SSM Agent 2.2 或更高版本。

创建运行 MOF 文件的关联

创建运行 MOF 文件的关联

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 State Manager。
3. 选择 State Manager，然后选择 创建关联。
4. 在名称 字段中指定名称。您可以自由选择，但我们建议您这样做。名称可以帮助了解您在创建关联时为关联考虑的用途。名称不得包含空格。
5. 在 Command (文档) 列表中，选择 **AWS-ApplyDSCMofs**。
6. 在 Parameters (参数) 部分中，指定您选择的必需和可选输入参数。

   a. Mofs To Apply (要应用的 Mof)：指定要在此关联运行时运行的一个或多个 MOF 文件。使用逗号分隔 MOF 文件的列表。您可以指定以下用于查找 MOF 文件的选项。

   - Amazon S3 存储桶名。存储桶名称必须使用小写字母。可使用以下格式指定此信息：

     ```
     s3:amzn-s3-demo-bucket:MOF_file_name.mof
     ```

     如果要指定 AWS 区域，请使用以下格式：

     ```
     s3:bucket_Region:amzn-s3-demo-bucket:MOF_file_name.mof
     ```

   - 安全的网站。可使用以下格式指定此信息。

     ```
     https://domain_name/MOF_file_name.mof
     ```

     见下列。

     ```
     https://www.example.com/TestMOF.mof
     ```

   - 本地共享中的文件系统。可使用以下格式指定此信息。

     ```
     \server_name\shared_folder_name\MOF_file_name.mof
     ```

     见下列。

```
\StateManagerAssociationsBox\MOFs_folder\MyMof.mof
```

b. Service Path (服务路径)：（可选）服务路径可以是要将报告和状态信息写入到的 Amazon S3 存储桶前缀。服务路径也可以是用于基于 Parameter Store 参数的标签的路径。在解析基于参数的标签时，系统将使用 {ssm:%servicePath%/*parameter_name*} 将 servicePath 值注入参数名称中。例如，如果服务路径是“WebServers/Production”，则系统将该参数解析为：WebServers/Production/*parameter_name*。在同一账户中运行多个环境时，此服务路径很有用。

c. Report Bucket Name (报告存储桶名称)：（可选）输入要写入合规性数据的 Amazon S3 存储桶的名称。报告以 JSON 格式保存在此存储桶中。

> **Note**
>
> 您可以用存储桶所在的区域作为存储桶名称的前缀。下面是一个示例：us-west-2:MyMOFBucket。如果对不包括 us-east-1 在内的特定区域中的 Amazon S3 端点使用代理，则用区域作为存储桶名称的前缀。如果存储桶名称不含前缀，它会使用 us-east-1 端点自动发现存储桶区域。

d. 操作模式：选择：选择 State Manager 行为时运行 **AWS-ApplyDSCMofs** 关联：

- Apply（应用）：更正不合规的节点配置。
- ReportOnly：不更正节点配置，但录入所有合规性数据并报告不合规节点。

e. Status Bucket Name (状态存储桶名称)：（可选）输入要将 MOF 执行状态信息写入到的 Amazon S3 存储桶的名称。这些状态报告是节点的最新合规性运行的单例摘要。这意味着，下次关联运行 MOF 文件时，将覆盖该报告。

> **Note**
>
> 您可以用存储桶所在的区域作为存储桶名称的前缀。示例如下：`us-west-2:amzn-s3-demo-bucket`。如果对不包括 us-east-1 在内的特定区域中的 Amazon S3 端点使用代理，则必须用区域作为存储桶名称的前缀。如果存储桶名称不含前缀，它会使用 us-east-1 端点自动发现存储桶区域。

f. Module Source Bucket Name (模块源存储桶名称)：（可选）输入包含 PowerShell 模块文件的 Amazon S3 存储桶的名称。如果您指定无中，为下一个选项选择真，允许 PS Gallery 模块源。

> **Note**
>
> 您可以用存储桶所在的区域作为存储桶名称的前缀。示例如下：`us-west-2:amzn-s3-demo-bucket`。如果对不包括 us-east-1 在内的特定区域中的 Amazon S3 端点使用代理，则必须用区域作为存储桶名称的前缀。如果存储桶名称不含前缀，它会使用 us-east-1 端点自动发现存储桶区域。

g. Allow PS Gallery Module Source (允许 PS 库模块源)：（可选）选择 True 从 https://www.powershellgallery.com/ 下载 PowerShell 模块。如果选择 假，请为上一个选项 ModuleSourceBucketName 指定源。

h. Proxy Uri (代理 URI)：（可选）使用此选项可从代理服务器下载 MOF 文件。

i. Reboot Behavior ()：（可选）如果 MOF 文件执行需要重启，请指定以下重启行为之一：

- AfterMof：在所有 MOF 执行完成后重启节点。即使多个 MOF 执行请求重启，系统也会等到所有 MOF 执行完成才重启。
- Immediately（立即）：只要 MOF 执行请求，即重启节点。如果实例运行请求重启的多个 MOF 文件，则会多次重启该节点。
- Never（从不）：即使 MOF 执行明确请求重启，节点也不重启。

j. Use Computer Name For Reporting (将计算机名称用于报告)：（可选）启用此选项可在报告合规性信息时使用计算机的名称。原定设置值为 false，这意味着系统在报告合规性信息时会使用节点 ID。

k. Enable Verbose Logging (启用详细日志记录)：（可选）建议在首次部署 MOF 文件时启用详细日志记录。

> **Important**
>
> 启用后，详细日志记录会向 Amazon S3 存储桶中写入比标准关联执行日志记录更多的数据。这可能会导致性能变慢，并且 Amazon S3 的存储费用可能会变高。要缓解存储大小问题，建议在 Amazon S3 存储桶中启用生命周期策略。有关更多信息，请参阅 Amazon Simple Storage Service 用户指南中的如何为 S3 Bucket 创建生命周期策略？。

l. Enable Debug Logging (启用调试日志记录)：（可选）建议启用调试日志记录排除 MOF 故障。我们还建议您在正常使用时禁用此选项。

> **Important**
>
> 启用后，调试日志记录会向 Amazon S3 存储桶中写入比标准关联执行日志记录更多的数据。这可能会导致性能变慢，并且 Amazon S3 的存储费用可能会变高。要缓解存储大小问题，建议在 Amazon S3 存储桶中启用生命周期策略。有关更多信息，请参阅 Amazon Simple Storage Service 用户指南中的如何为 S3 Bucket 创建生命周期策略？。

m. Compliance Type (合规性类型)：（可选）指定要在报告合规性信息时使用的合规性类型。默认的合规性类型为 Custom:DSC (自定义:DSC)。如果创建多个运行 MOF 文件的关联，请确保为每个关联指定不同的合规性类型。否则，使用 Custom:DSC 的每个其他关联都将覆盖现有的合规性数据。

n. Pre Reboot Script (重启前脚本)：（可选）指定在配置指示需要重启时要运行的脚本。该脚本将在重启之前运行。该脚本必须为单行。使用分号分隔其他行。

7. 在 Targets (目标) 部分中，选择 Specifying tags (指定标签) 或 Manually Selecting Instance (手动选择实例)。如果选择使用标签将资源设置为目标，请在提供的字段中输入标签键和标签值。有关使用目标的更多信息，请参阅 了解 State Manager 关联中的目标和速率控制。

8. 在 Specify schedule (指定计划) 部分中，选择 On Schedule (按计划) 或 No schedule (无计划)。如果选择 On Schedule (按计划)，则可使用提供的按钮为关联创建 cron 或 rate 计划。

9. 在 Advanced options (高级选项) 部分中：

   - 在 Compliance severity (合规性严重级别) 中，选择关联的严重级别。合规性报告指示关联状态是合规还是不合规以及您在此处指示的严重级别。有关更多信息，请参阅 关于 State Manager 关联合规性。

10. 在 Rate control（速率控制）部分，配置用于跨托管式节点机群运行 State Manager 关联的选项。有关这些选项的详细信息，请参阅了解 State Manager 关联中的目标和速率控制。

    在并发部分中，选择一个选项：

    - 选择 targets (目标) 输入可同时运行关联的目标的绝对数量。
    - 选择 percentage (百分比) 输入可同时运行关联的目标集的百分比。

    在错误阈值部分中，选择一个选项：

    - 选择 errors (错误) 以输入 State Manager 停止对其他目标运行关联之前允许的错误的绝对数量。

- 选择 percentage (百分比) 以输入 State Manager 停止对其他目标运行关联之前允许的错误的百分比。

11. （可选）对于 Output options (输出选项)，要将命令输出保存到文件，请选中 Enable writing output to S3 (启用将输出写入 S3) 方框。在方框中输入存储桶和前缀（文件夹）名称。

> **Note**
>
> 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给托管式节点的实例配置文件的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确认与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

12. 选择创建关联。

State Manager 在指定的节点或目标中创建并立即运行关联。在初始执行后，关联根据定义的计划和以下规则按间隔运行：

- State Manager 对间隔开启时处于线上状态的节点运行关联，并跳过离线节点。
- State Manager 在间隔期间会尝试对所有已配置的节点运行关联。
- 如果在某个周期内未运行某一关联（例如，由于并发值限制了可以同时处理关联的节点数），State Manager 将尝试在下一个周期运行该关联。
- State Manager 会记录所有已跳过间隔的历史记录。可在 Execution History (执行历史记录) 选项卡上查看历史记录。

> **Note**
>
> `AWS-ApplyDSCMofs` 是 Systems Manager 命令文档 这意味着，您也可以使用 Run Command（AWS Systems Manager 中的一项工具）运行此文档。有关更多信息，请参阅 AWS Systems Manager Run Command。

排查创建运行 MOF 文件的关联时出现的问题

本部分包含的信息可帮助您解决在创建运行 MOF 文件的关联时遇到的问题。

启用增强的日志记录

解决问题的第一步是启用增强型日志记录。具体来说，是执行以下操作：

1. 验证关联是否配置是将命令输出写入 Amazon S3 或 Amazon CloudWatch Logs (CloudWatch)。
2. 将 Enable Verbose Logging (启用详细日志记录) 参数设置为 True。
3. 将 Enable Debug Logging (启用调试日志记录) 参数设置为 True。

启用详细和调试日志记录后，Stdout 输出文件将包含有关脚本执行的详细信息。此输出文件可帮助您确定脚本失败的位置。Stderr 输出文件包含在脚本执行期间发生的错误。

创建运行 MOF 文件的关联时的常见问题

本节包含有关在创建运行 MOF 文件的关联时出现的常见问题的信息以及解决这些问题的步骤。

我的 MOF 未应用

如果 State Manager 未能将关联应用到您的节点，请首先查看 Stderr 输出文件。此文件可以帮助您了解问题的根本原因。另外，请确认：

- 此节点具有与 MOF 相关的所有 Simple Storage Service (Amazon S3) 存储桶所需访问权限。具体来说：
  - s3:GetObject permissions (s3:GetObject 权限)：这是私有 Amazon S3 存储桶中的 MOF 文件和 Amazon S3 存储桶中的自定义模块必需的权限。
  - s3:PutObject permission (s3:PutObject 权限)：在将合规性报告和合规性状态写入 Amazon S3 存储桶时需要此权限。
- 如果使用标签，则需确保该节点具有所需的 IAM policy。使用标签需要实例 IAM 角色具有允许 `ec2:DescribeInstances` 和 `ssm:ListTagsForResource` 操作的策略。
- 确保该节点已分配预期的标签或 SSM 参数。
- 请确保标签或 SSM 参数不包含拼写错误。
- 尝试对节点本地应用 MOF，以确保 MOF 文件本身没有问题。

我的 MOF 似乎失败了，但 Systems Manager 执行成功

如果 `AWS-ApplyDSCMofs` 文档成功运行，则 Systems Manager 执行状态将显示 Success (成功)。此状态并不反映节点针对 MOF 文件中的配置要求的合规性状态。要查看节点的合规性状态，请查看合规性报告。您可以查看 Amazon S3 报告存储桶中的 JSON 报告。这适用于 Run Command 和 State Manager 执行。此外，对于State Manager ，您可以在 Systems Manager 合规性页面上查看合规性详细信息。

Stderr 状态：尝试访问服务时名称解析失败

此错误指示脚本无法访问远程服务。最有可能的情况是，该脚本无法访问 Amazon S3 。当脚本尝试将合规性报告或合规性状态写入文档参数中提供的 Amazon S3 存储桶时，通常会发生此问题。通常，当计算环境使用包含允许清单的防火墙或透明代理时，会出现此错误。要解决此问题，请执行以下操作：

- 将特定于区域的存储桶语法用于所有 Amazon S3 存储桶参数。例如，Mofs to Apply (要应用的 Mof) 参数的格式应如下所示：

  s3:*bucket-region*:*amzn-s3-demo-bucket*:*mof-file-name*.mof。

  示例如下： `s3:us-west-2:amzn-s3-demo-bucket:my-mof.mof`

  报告、状态和模块源存储桶名称的格式应如下所示：

  bucket-region:*amzn-s3-demo-bucket*。下面是一个示例：`us-west-1:amzn-s3-demo-bucket;`
- 如果特定于区域的语法无法解决此问题，请确保目标节点可以访问所需区域中的 Simple Storage Service (Amazon S3)。对此进行确认：
  1. 在相应的 Amazon S3 区域中查找 Amazon S3 的端点名称。有关更多信息，请参阅《Amazon Web Services 一般参考》中的 Amazon S3 Service Endpoints。
  2. 登录目标节点并运行以下 ping 命令。

     ```
     ping s3.s3-region.amazonaws.com
     ```

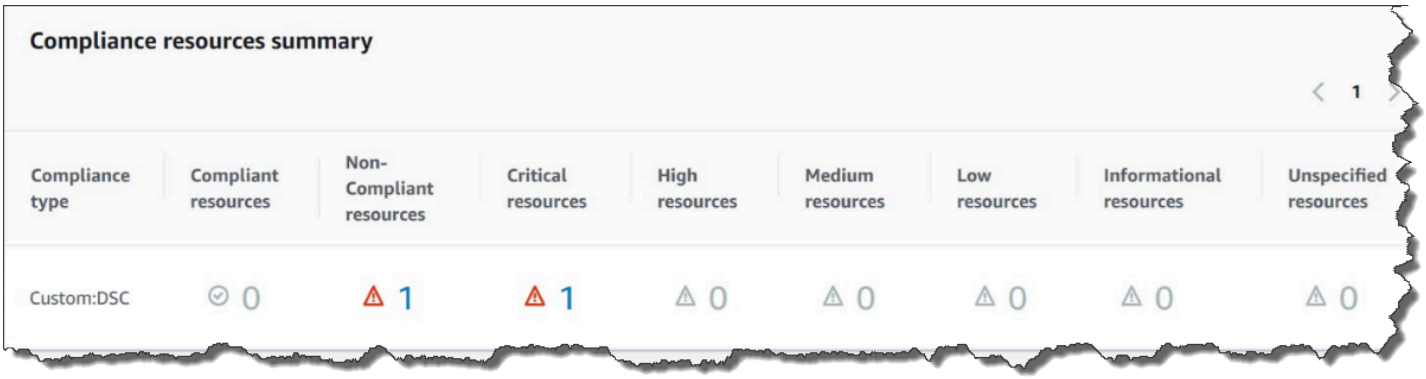
     如果 ping 命令失败，则意味着 Simple Storage Service (Amazon S3) 已关闭或防火墙/透明代理阻止访问 Simple Storage Service (Amazon S3) 区域，或节点无法访问 Internet。

查看 DSC 资源合规性详细信息

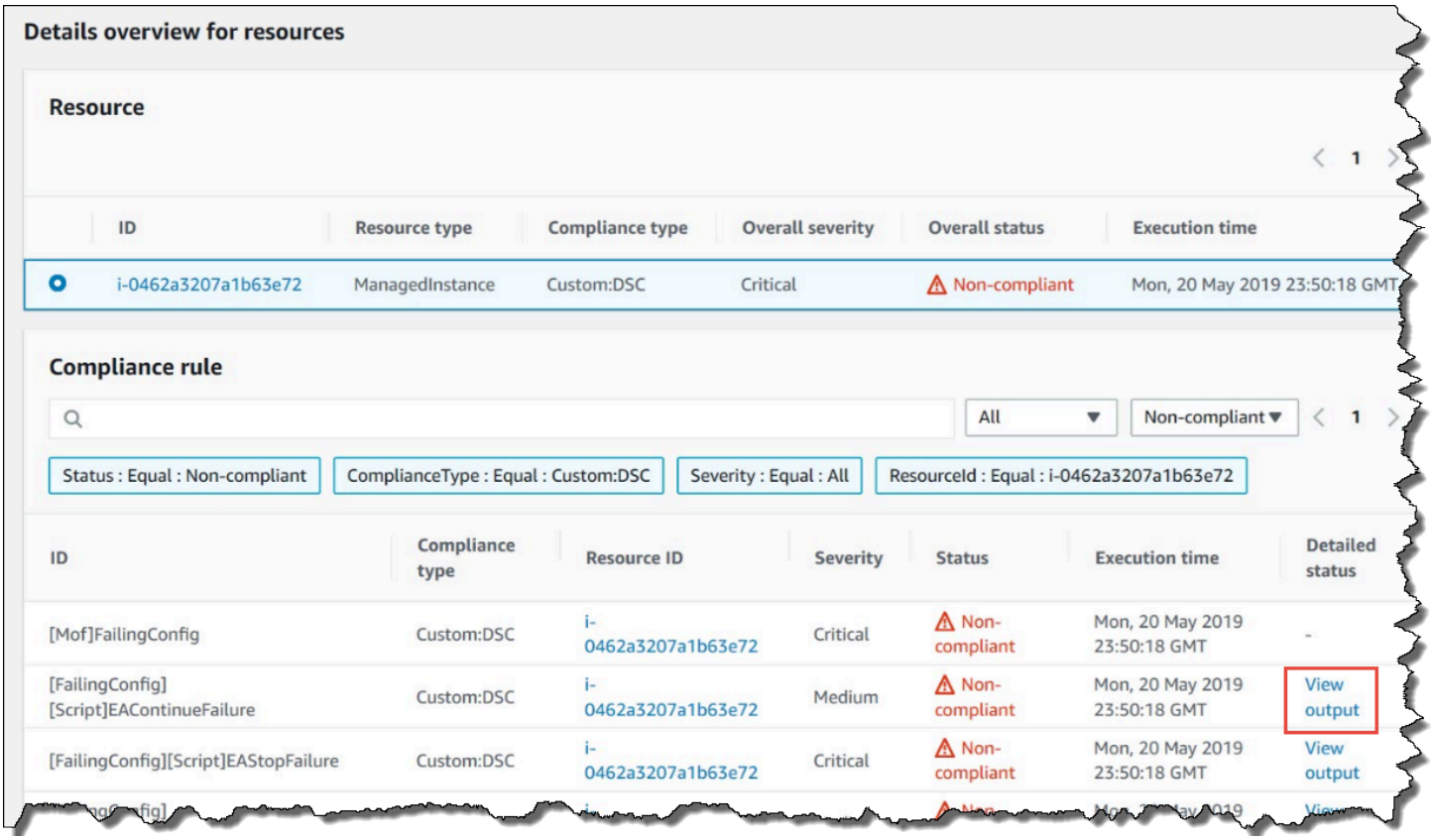
在运行 `AWS-ApplyDSCMofs` 文档时指定的 Amazon S3 状态存储桶中 Systems Manager 捕获有关 DSC 资源失败的合规性信息。在 Amazon S3 存储桶中搜索有关 DSC 资源失败的信息可能需要很长时间。相反，您可以在 Systems Manager Compliance (合规性) 页面上查看该信息。

合规性资源摘要部分显示失败的资源数。在以下示例中，ComplianceType 是 Custom:DSC，并且一个资源不合规。

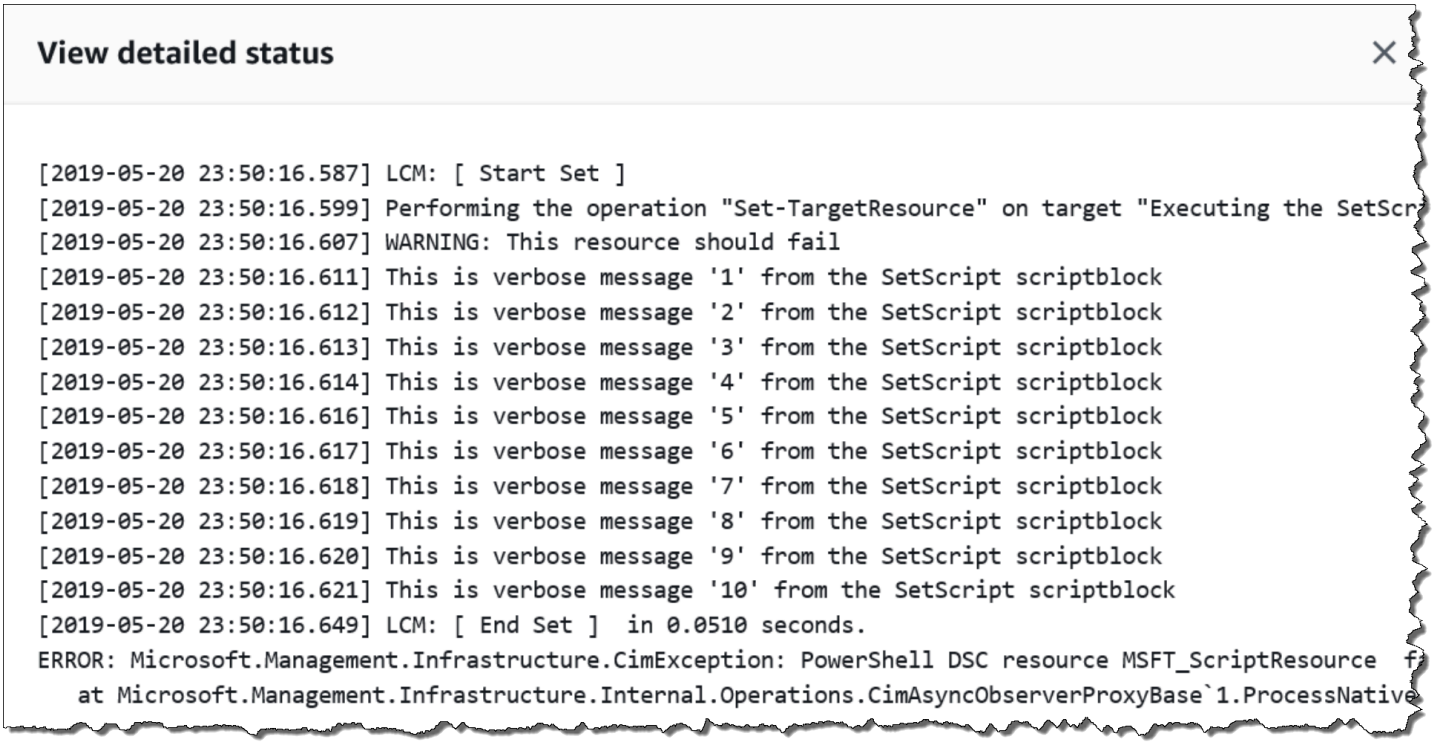
> **Note**
>
> Custom:DSC 是 `AWS-ApplyDSCMofs` 文档中的默认 ComplianceType 值。该值是可自定义的。



Details overview for resources (资源的详细概述) 部分显示有关具有不合规 DSC 资源的 AWS 资源的信息。该部分还包括 MOF 名称、脚本执行步骤以及用于查看详细状态信息的查看输出链接（如果适用）。



查看输出链接显示详细状态的最后 4,000 个字符。Systems Manager 将异常作为第一个元素，然后将扫描详细消息，并在前面添加尽可能多的内容，直至达到 4,000 个字符的配额。该过程显示在引发异常之前输出的日志消息，这些消息是用于故障排除的最相关消息。



有关如何查看合规性信息的信息，请参阅 AWS Systems Manager Compliance。

影响合规性报告的情况

如果 State Manager 关联失败，则不会报告合规性数据。更具体地说，如果 MOF 无法处理，则 Systems Manager 不会报告任何合规性项目，因为关联失败。例如，如果 Systems Manager 尝试从节点没有权限访问的 Simple Storage Service (Amazon S3) 存储桶中下载 MOF，则关联失败，并且不会报告合规性数据。

如果第二个 MOF 中的资源失败， Systems Manager 将不会报告合规性数据。例如，如果 MOF 尝试在不存在的驱动器上创建一个文件，Systems Manager 将报告合规性，因为 `AWS-ApplyDSCMofs` 文档可以完全处理，这意味着关联成功运行。

创建运行 Ansible Playbook 的关联

您可以使用 `AWS-ApplyAnsiblePlaybooks` SSM 文档创建运行 Ansible Playbook 的State Manager 关联。State Manager是 AWS Systems Manager 中的一项工具。本文档为运行 Playbook 提供了以下好处：

- 支持运行复杂的 Playbook
- 支持从 GitHub 和 Amazon Simple Storage Service（Amazon S3）下载 Playbook
- 支持压缩的 PlayBook 结构
- 增强的日志记录
- 捆绑 Playbook 时可以指定要运行的 Playbook

> **Note**
>
> Systems Manager 包括两个 SSM 文档，使您能够创建运行 Ansible Playbook 的 State Manager 关联：`AWS-RunAnsiblePlaybook` 和 `AWS-ApplyAnsiblePlaybooks`。`AWS-RunAnsiblePlaybook` 文档已被弃用。它在 Systems Manager 中仍然提供，用于传统用途。由于此处介绍的增强功能，我们建议您使用 `AWS-ApplyAnsiblePlaybooks` 文档。运行 Ansible Playbook 的关联在 macOS 中不受支持。

支持运行复杂的 Playbook

`AWS-ApplyAnsiblePlaybooks` 文档支持捆绑的复杂 Playbook，因为它会在执行指定的主 Playbook 之前将整个文件结构复制到本地目录中。您可以在 Zip 文件或目录结构中提供源 Playbook。Zip 文件或目录可以存储在 GitHub 或 Amazon S3 中。

支持从 GitHub 下载 Playbook

`AWS-ApplyAnsiblePlaybooks` 文档使用 `aws:downloadContent` 插件下载 Playbook 文件。文件可以存储在 GitHub 中的单个文件中，也可以存储为一组组合 Playbook 文件。要从 GitHub 下载内容，请以 JSON 格式指定有关 GitHub 存储库的信息。见下列。

```
{
   "owner":"TestUser",
   "repository":"GitHubTest",
   "path":"scripts/python/test-script",
   "getOptions":"branch:master",
   "tokenInfo":"{{ssm-secure:secure-string-token}}"
}
```

支持从 Amazon S3 下载 Playbook

您还可以将 Ansible Playbook 作为单个 .zip 文件或目录结构存储和下载到 Amazon S3 中。要从 Amazon S3 下载内容，您必须指定该文件的路径。以下是两个示例。

示例 1：下载特定 Playbook 文件

```
{
   "path":"https://s3.amazonaws.com/amzn-s3-demo-bucket/playbook.yml"
}
```

示例 2：下载目录的内容

```
{
   "path":"https://s3.amazonaws.com/amzn-s3-demo-bucket/ansible/webservers/"
}
```

> **Important**
>
> 如果指定 Amazon S3，托管式节点上的 AWS Identity and Access Management（IAM）实例配置文件必须包含 S3 存储桶的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限。

支持压缩的 PlayBook 结构

AWS-ApplyAnsiblePlaybooks 文档使您可以在下载的捆绑包中运行压缩的 .zip 文件。该文档检查下载的文件是否包含 .zip 格式的压缩文件。如果找到 .zip 文件，则文档将自动解压缩文件，然后运行指定的 Ansible 自动化。

增强的日志记录

AWS-ApplyAnsiblePlaybooks 文档包括一个可选参数，用于指定不同级别的日志记录。指定 -v 来表示低详细程度，-vv 或 –vvv 表示中等详细程度，-vvvv 表示调试级别日志记录。这些选项直接映射到 Ansible 详细程度选项。

捆绑 Playbook 时可以指定要运行的 Playbook

AWS-ApplyAnsiblePlaybooks 文档包含一个必需参数，用于指定捆绑多个 PlayBook 时要运行哪个 PlayBook。此选项为运行 PlayBook 提供了灵活性，以支持不同的使用案例。

了解安装的依赖项

如果您为 InstallDependencies 参数指定为 True，则 Systems Manager 会验证您的节点上是否安装了以下依赖项。

- Ubuntu Server/Debian Server：Apt-get（程序包管理）、Python 3、Ansible、Unzip
- Amazon Linux：Ansible
- RHEL：Python 3、Ansible、Unzip

如果找不到这些依赖项中的一个或多个，则 Systems Manager 将自动安装它们。

创建运行 Ansible Playbook 的关联（控制台）

以下过程介绍了如何使用 Systems Manager 控制台创建 State Manager 关联，以便使用 `AWS-ApplyAnsiblePlaybooks` 文档运行 Ansible Playbook。

创建运行 Ansible Playbook 的关联（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 State Manager。
3. 选择 Create Association，然后选择 State Manager。
4. 对于 Name（名称），请指定一个名称，该名称可帮助您记住关联的用途。
5. 在 Document (文档) 列表中，选择 **AWS-ApplyAnsiblePlaybooks**。
6. 在 Parameters (参数) 部分中，对于 Source Type (源类型)，选择 GitHub 或 S3。

   GitHub

   如果选择 GitHub，则采用以下格式输入存储库信息。

   ```
   {
      "owner":"user_name",
      "repository":"name",
      "path":"path_to_directory_or_playbook_to_download",
      "getOptions":"branch:branch_name",
      "tokenInfo":"{{(Optional)_token_information}}"
   }
   ```

   S3

   如果选择 S3，则采用以下格式输入路径信息：

   ```
   {
      "path":"https://s3.amazonaws.com/path_to_directory_or_playbook_to_download"
   ```

```
}
```

7. 对于 Install Dependencies (安装依赖项)，请选择一个选项。
8. （可选）对于 Playbook File (PlayBook 文件)，输入文件名。如果 Zip 文件包含 Playbook，则必须指定 Zip 文件的相对路径。
9. （可选）对于额外变量，请输入希望 State Manager 在运行时发送到 Ansible 的变量。
10. （可选）对于 Check (检查)，选择一个选项。
11. （可选）对于 Verbose (详细)，选择一个选项。
12. 对于 Targets (目标)，选择一个选项。有关使用目标的信息，请参阅 [了解 State Manager 关联中的目标和速率控制]。
13. 在 Specify schedule (指定计划) 部分中，选择 On schedule (按计划) 或 No schedule (无计划)。如果选择 On schedule (按计划)，则可使用提供的按钮为关联创建 cron 或 rate 计划。
14. 在 高级选项 部分中，对于合规性严重级别，选择关联的严重级别。合规性报告指示关联状态是合规还是不合规以及您在此处指示的严重级别。有关更多信息，请参阅 [关于 State Manager 关联合规性]。
15. 在 Rate control（速率控制）部分，配置用于在托管式节点机群中运行 State Manager 关联的选项。有关使用速率控制的信息，请参阅 [了解 State Manager 关联中的目标和速率控制]。

    在并发部分中，选择一个选项：

    - 选择 targets (目标) 输入可同时运行关联的目标的绝对数量。
    - 选择 percentage (百分比) 输入可同时运行关联的目标集的百分比。

    在错误阈值部分中，选择一个选项：

    - 选择 errors (错误) 以输入允许的错误绝对数量，超过该数量后 State Manager 停止对其他目标运行关联。
    - 选择 percentage (百分比) 以输入允许的错误百分比，超过该百分比后 State Manager 停止对其他目标运行关联。
16. （可选）对于 Output options (输出选项)，要将命令输出保存到文件，请选中 Enable writing output to S3 (启用将输出写入 S3) 方框。在方框中输入存储桶和前缀（文件夹）名称。

> **Note**
>
> 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给托管式节点的实例配置文件的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅[配置 Systems

Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确认与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

17. 选择创建关联。

> **Note**
>
> 如果使用标签在一个或多个目标节点上创建关联，然后从某一节点中删除标签，则该节点将不再运行该关联。该节点不再与 State Manager 文档关联。

创建运行 Ansible Playbook 的关联（CLI）

以下过程介绍了如何使用 AWS Command Line Interface（AWS CLI）创建 State Manager 关联，以便使用 `AWS-ApplyAnsiblePlaybooks` 文档运行 Ansible Playbook。

创建运行 Ansible Playbook 的关联（CLI）

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关更多信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 运行以下命令之一来创建运行 Ansible Playbook 的关联，该关联使用标签将节点设为目标。将每个*#######*替换为您自己的信息。命令（A）指定 GitHub 作为源类型。命令 (B) 指定 Amazon S3 作为源类型。

   (A) GitHub 源

   Linux & macOS

```
aws ssm create-association --name "AWS-ApplyAnsiblePlaybooks" \
    --targets Key=tag:TagKey,Values=TagValue \
    --parameters '{"SourceType":["GitHub"],"SourceInfo":
["{\"owner\":\"owner_name\", \"repository\": \"name\",
 \"getOptions\": \"branch:master\"}"],"InstallDependencies":
["True_or_False"],"PlaybookFile":["file_name.yml"],"ExtraVariables":["key/
value_pairs_separated_by_a_space"],"Check":["True_or_False"],"Verbose":["-v,-
vv,-vvv, or -vvvv"],"TimeoutSeconds":["3600"]}' \
    --association-name "name" \
```

```
    --schedule-expression "cron_or_rate_expression"
```

Windows

```
aws ssm create-association --name "AWS-ApplyAnsiblePlaybooks" ^
    --targets Key=tag:TagKey,Values=TagValue ^
    --parameters '{"SourceType":["GitHub"],"SourceInfo":
["{\"owner\":\"owner_name\", \"repository\": \"name\",
 \"getOptions\": \"branch:master\"}"],"InstallDependencies":
["True_or_False"],"PlaybookFile":["file_name.yml"],"ExtraVariables":["key/
value_pairs_separated_by_a_space"],"Check":["True_or_False"],"Verbose":["-v,-
vv,-vvv, or -vvvv"], "TimeoutSeconds":["3600"]}' ^
    --association-name "name" ^
    --schedule-expression "cron_or_rate_expression"
```

见下列。

```
aws ssm create-association --name "AWS-ApplyAnsiblePlaybooks" \
    --targets "Key=tag:OS,Values=Linux" \
    --parameters '{"SourceType":["GitHub"],"SourceInfo":["{\"owner\":
\"ansibleDocumentTest\", \"repository\": \"Ansible\", \"getOptions\":
 \"branch:master\"}"],"InstallDependencies":["True"],"PlaybookFile":["hello-world-
playbook.yml"],"ExtraVariables":["SSM=True"],"Check":["False"],"Verbose":["-v"]}' \
    --association-name "AnsibleAssociation" \
    --schedule-expression "cron(0 2 ? * SUN *)"
```

(B) S3 源

Linux & macOS

```
aws ssm create-association --name "AWS-ApplyAnsiblePlaybooks" \
    --targets Key=tag:TagKey,Values=TagValue \
    --parameters '{"SourceType":["S3"],"SourceInfo":["{\"path\":\"https://
s3.amazonaws.com/
path_to_Zip_file,_directory,_or_playbook_to_download\"}"],"InstallDependencies":
["True_or_False"],"PlaybookFile":["file_name.yml"],"ExtraVariables":["key/
value_pairs_separated_by_a_space"],"Check":["True_or_False"],"Verbose":["-v,-
vv,-vvv, or -vvvv"]}' \
    --association-name "name" \
    --schedule-expression "cron_or_rate_expression"
```

Windows

```
aws ssm create-association --name "AWS-ApplyAnsiblePlaybooks" ^
    --targets Key=tag:TagKey,Values=TagValue ^
    --parameters '{"SourceType":["S3"],"SourceInfo":["{\"path\":\"https://
s3.amazonaws.com/
path_to_Zip_file,_directory,_or_playbook_to_download\"}"],"InstallDependencies":
["True_or_False"],"PlaybookFile":["file_name.yml"],"ExtraVariables":["key/
value_pairs_separated_by_a_space"],"Check":["True_or_False"],"Verbose":["-v,-
vv,-vvv, or -vvvv"]}' ^
    --association-name "name" ^
    --schedule-expression "cron_or_rate_expression"
```

见下列。

```
aws ssm create-association --name "AWS-ApplyAnsiblePlaybooks" \
    --targets "Key=tag:OS,Values=Linux" \
    --parameters '{"SourceType":["S3"],"SourceInfo":["{\"path\":\"https://
s3.amazonaws.com/amzn-s3-demo-bucket/playbook.yml\"}"],"InstallDependencies":
["True"],"PlaybookFile":["playbook.yml"],"ExtraVariables":["SSM=True"],"Check":
["False"],"Verbose":["-v"]}' \
    --association-name "AnsibleAssociation" \
    --schedule-expression "cron(0 2 ? * SUN *)"
```

> **Note**
>
> State Manager 关联不支持所有的 cron 和 rate 表达式。有关为关联创建 cron 和 rate 表达式的更多信息，请参阅 参考：适用于 Systems Manager 的 Cron 和 Rate 表达式。

系统将尝试在节点上创建关联并立即应用状态。

3. 运行以下命令查看您刚才创建的关联的更新状态。

```
aws ssm describe-association --association-id "ID"
```

创建运行 Chef 配方的关联

使用 AWS-ApplyChefRecipes SSM 文档创建运行 Chef 配方的State Manager关联。State Manager 是 AWS Systems Manager 中的一项工具。可以使用 AWS-ApplyChefRecipes SSM 文档定位基于 Linux 的 Systems Manager 托管式节点。本文档为运行 Chef 配方提供了以下好处：

- 支持多个版本的 Chef（Chef 11 到 Chef 18）。
- 在目标节点上自动安装 Chef 客户端软件。
- （可选）在目标节点上运行 Systems Manager 合规性检查，并将合规性检查的结果存储在 Amazon Simple Storage Service (Amazon S3) 存储桶中。
- 在文档的单次运行中运行多个说明书和配方。
- （可选）在 why-run 模式下运行配方，以显示哪些配方会在未进行更改的情况下在目标节点上发生更改。
- （可选）将自定义 JSON 属性应用于 chef-client 运行。
- （可选）应用存储在指定位置源文件中的自定义 JSON 属性。

您可以使用 Git、GitHub、HTTP 或 Amazon S3 存储桶作为您在 AWS-ApplyChefRecipes 文档中指定的 Chef 说明书和配方的下载源。

> **Note**
>
> 运行 Chef 配方的关联在 macOS 上不受支持。

入门

在创建 AWS-ApplyChefRecipes 文档之前，请准备好 Chef 说明书和说明书存储库。如果尚未获得要使用的 Chef 说明书，则可以从使用 AWS 为您准备的测试 HelloWorld 说明书开始。默认情况下，AWS-ApplyChefRecipes 文档已指向此说明书。您的说明书的设置应类似于以下目录结构。在以下示例中，jenkins 和 nginx 是 Chef 网站上的 Chef Supermarket 中提供的 Chef 说明书的示例。

虽然 AWS 无法正式支持 Chef Supermarket 网站上的说明书，但其中的许多说明书将与 AWS-ApplyChefRecipes 文档结合使用。以下是在测试社群说明书时需要确认的标准示例：

- 说明书应支持您的目标 Systems Manager 托管式节点的基于 Linux 的操作系统。

- 说明书将适用于您使用的 Chef 客户端版本（Chef 11 到 Chef 18）。
- 说明书与 Chef Infra Client 兼容，并且不需要 Chef 服务器。

验证您是否能访问 `Chef.io` 网站，以便在运行列表中指定的任何说明书都能在 Systems Manager 文档（SSM 文档）运行时安装。支持使用嵌套的 `cookbooks` 文件夹，但这不是必需的；您可以将说明书直接存储在根级别下。

```
<Top-level directory, or the top level of the archive file (ZIP or tgz or tar.gz)>
    ### cookbooks (optional level)
        ### jenkins
        #   ### metadata.rb
        #   ### recipes
        ### nginx
            ### metadata.rb
            ### recipes
```

> **Important**
>
> 在创建运行 Chef 配方的 State Manager 关联之前需注意，文档运行时会在 Systems Manager 托管式节点上安装 Chef 客户端软件，除非将 Chef 客户端版本的值设置为 `None`。此操作使用 Chef 中的安装脚本代表您安装 Chef 组件。在运行 `AWS-ApplyChefRecipes` 文档之前，请确保您的企业能够遵守任何适用的法律要求，包括适用于对 Chef 软件的使用的许可条款。有关更多信息，请参阅 Chef 网站。

Systems Manager 可以向 S3 存储桶、Systems Manager 控制台发送合规性报告，或者提供合规性结果以响应 Systems Manager API 命令。要运行 Systems Manager 合规性报告，附上 Systems Manager 托管式节点的实例配置文件必须具有写入到 S3 存储桶的权限。该实例配置文件必须有权使用 Systems Manager `PutComplianceItem` API。有关 Systems Manager 合规性的更多信息，请参阅 AWS Systems Manager Compliance。

记录文档运行

当您通过使用 State Manager 关联运行 Systems Manager 文档（SSM 文档）时，您可以配置关联以选择文档运行的输出，并且可以将输出发送到 Amazon S3 或 Amazon CloudWatch Logs（CloudWatch Logs）。要帮助在关联运行完后轻松进行故障排除，请验证是否已将关联配置为将命令输出写入 Amazon S3 存储桶或 CloudWatch Logs 。有关更多信息，请参阅 在 Systems Manager 中使用关联。。

运行配方时向目标应用 JSON 属性

您可以为 Chef 客户端指定 JSON 属性，以便在关联运行期间应用于目标节点。设置关联时，您可以提供原始 JSON，也可以提供存储在 Amazon S3 中的 JSON 文件路径。

如果您想自定义配方的运行方式而无需修改配方本身，请使用 JSON 属性，例如：

- 覆盖少量属性

  使用自定义 JSON 可以避免维护一个配方的多个版本，以适应细微差异。
- 提供变量值

  使用自定义 JSON 指定可能随运行而变化的值。例如，如果您的 Chef 说明书配置了接受付款的第三方应用程序，则可以使用自定义 JSON 指定付款端点 URL。

在原始 JSON 中指定属性

以下是可用于为 Chef 配方指定自定义 JSON 属性的格式示例。

```
{"filepath":"/tmp/example.txt", "content":"Hello, World!"}
```

指定 JSON 文件的路径

以下是可用于为 Chef 配方指定自定义 JSON 属性路径的格式示例。

```
{"sourceType":"s3", "sourceInfo":"someS3URL1"}, {"sourceType":"s3",
 "sourceInfo":"someS3URL2"}
```

使用 Git 作为说明书源

`AWS-ApplyChefRecipes` 文档使用 aws:downloadContent 插件下载 Chef 说明书。要从 Git 下载内容，请以 JSON 格式指定有关 Git 存储库的信息，如下例所示。将每个 *example-resource-placeholder* 替换为您自己的信息。

```
{
   "repository":"GitCookbookRepository",
   "privateSSHKey":"{{ssm-secure:ssh-key-secure-string-parameter}}",
   "skipHostKeyChecking":"false",
   "getOptions":"branch:refs/head/main",
   "username":"{{ssm-secure:username-secure-string-parameter}}",
   "password":"{{ssm-secure:password-secure-string-parameter}}"
```

```
}
```

使用 GitHub 作为说明书源

AWS-ApplyChefRecipes 文档使用 aws:downloadContent 插件下载说明书。要从 GitHub 下载内容，请以 JSON 格式指定有关 GitHub 存储库的信息，如下例所示。将每个 *example-resource-placeholder* 替换为您自己的信息。

```
{
   "owner":"TestUser",
   "repository":"GitHubCookbookRepository",
   "path":"cookbooks/HelloWorld",
   "getOptions":"branch:refs/head/main",
   "tokenInfo":"{{ssm-secure:token-secure-string-parameter}}"
}
```

使用 HTTP 作为说明书源

您可以在自定义 HTTP 位置将 Chef 说明书存储为单个 .zip 或 tar.gz 文件，也可以存储为目录结构。要从 HTTP 下载内容，请以 JSON 格式指定文件或目录的路径，如下例所示。将每个 *example-resource-placeholder* 替换为您自己的信息。

```
{
   "url":"https://my.website.com/chef-cookbooks/HelloWorld.zip",
   "allowInsecureDownload":"false",
   "authMethod":"Basic",
   "username":"{{ssm-secure:username-secure-string-parameter}}",
   "password":"{{ssm-secure:password-secure-string-parameter}}"
}
```

使用 Amazon S3 作为说明书源

您还可以将 Chef 说明书作为单个 .zip 或 tar.gz 文件或者目录结构存储和下载到 Amazon S3 中。要从 Amazon S3 下载内容，请以 JSON 格式指定文件的路径，如下例所示。将每个 *example-resource-placeholder* 替换为您自己的信息。

示例 1：下载特定说明书

```
{
   "path":"https://s3.amazonaws.com/chef-cookbooks/HelloWorld.zip"
}
```

示例 2：下载目录的内容

```
{
   "path":"https://s3.amazonaws.com/chef-cookbooks-test/HelloWorld"
}
```

> **Important**
>
> 如果您指定 Simple Storage Service (Amazon S3)，则必须使用 `AmazonS3ReadOnlyAccess` 策略配置托管式节点上的 AWS Identity and Access Management (IAM) 实例配置文件。有关更多信息，请参阅配置 Systems Manager 所需的实例权限。

创建运行 Chef 配方的关联（控制台）

以下过程介绍了如何使用 Systems Manager 控制台创建 State Manager 关联，以便使用 `AWS-ApplyChefRecipes` 文档运行 Chef 说明书。

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 State Manager。
3. 选择 State Manager，然后选择 Create Association。
4. 对于 Name (名称)，请输入一个名称，该名称可帮助您记住关联的用途。
5. 在 Document (文档) 列表中，选择 **`AWS-ApplyChefRecipes`**。
6. 在参数中，对于源类型，选择 Git、GitHub、HTTP 或 S3。
7. 对于源信息，请使用您在步骤 6 中选择的源类型的适当格式输入说明书源信息。有关更多信息，请参阅以下主题：
   - the section called “使用 Git 作为说明书源”
   - the section called “使用 GitHub 作为说明书源”
   - the section called “使用 HTTP 作为说明书源”
   - the section called “使用 Amazon S3 作为说明书源”
8. 在 Run list (运行列表) 中，按以下格式列出要运行的配方，并用逗号将每个配方分隔开，如下所示。请不要在逗号后面包含空格。将每个 *`example-resource-placeholder`* 替换为您自己的信息。

```
recipe[cookbook-name1::recipe-name],recipe[cookbook-name2::recipe-name]
```

9. （可选）指定希望 Chef 客户端传递给目标节点的自定义 JSON 属性。

   a. 在 JSON 属性内容中，添加希望 Chef 客户端传递给目标节点的任何属性。

   b. 在 JSON 属性来源中，将路径添加到希望 Chef 客户端传递给目标节点的任何属性中。

   有关更多信息，请参阅 the section called “运行配方时向目标应用 JSON 属性”。

10. 对于 Chef 客户端版本，请指定 Chef 版本。有效值为 11 到 18 或 None。如果您指定 11 到 18 之间的一个数字（包含这两个数字），则 Systems Manager 会在目标节点上安装正确的 Chef 客户端版本。如果您指定 None，则在运行文档的配方之前，Systems Manager 不会在目标节点上安装 Chef 客户端。

11. （可选）对于 Chef 客户端参数，请指定您正在使用的 Chef 版本所支持的其他参数。要了解有关支持的参数的更多信息，请在运行 Chef 客户端的节点上运行 `chef-client -h`。

12. （可选）启用 Why-run 以显示在配方运行时对目标节点进行的更改，这不会实际更改目标节点。

13. 对于 Compliance severity (合规性严重性)，请选择要报告的 Systems Manager 合规性结果的严重性。合规性报告指示关联状态是合规还是不合规以及您在指定的严重级别。将合规性报告存储在一个 S3 存储桶中，您已将该存储桶指定为 Compliance report bucket (合规性报告存储桶) 参数的值（步骤 14）。有关合规性的更多信息，请参阅本指南中的 了解有关合规性的详细信息。

    合规性扫描测量 Chef 配方和节点资源中指定的配置之间的偏离。有效值为 `Critical`、`High`、`Medium`、`Low`、`Informational`、`Unspecified` 或 None。要跳过合规性报告，请选择 None。

14. 对于 Compliance type (合规性类型)，请指定要报告其结果的合规性类型。有效值为 `Association`（对于 State Manager 关联）或 `Custom:` *`#####`*。默认值为 `Custom:Chef`。

15. 对于合规性报告存储桶，输入 S3 存储桶的名称，该存储桶用于存储此文档执行的每个 Chef 运行的相关信息，包括资源配置和合规性结果。

16. 在 Rate control（速率控制）中，配置用于在托管式节点机群中运行 State Manager 关联的选项。有关使用速率控制的信息，请参阅 了解 State Manager 关联中的目标和速率控制。

    在 Concurrency (并发) 中，选择一个选项：

    - 选择 targets (目标) 输入可同时运行关联的目标的绝对数量。
    - 选择 percentage (百分比) 输入可同时运行关联的目标集的百分比。

在 Error threshold (错误阈值) 中，选择一个选项：

- 选择 errors (错误) 以输入允许的错误绝对数量，超过该数量后 State Manager 停止对其他目标运行关联。
- 选择 percentage (百分比) 以输入允许的错误百分比，超过该百分比后 State Manager 停止对其他目标运行关联。

17. （可选）对于 Output options (输出选项)，要将命令输出保存到文件，请选中 Enable writing output to S3 (启用将输出写入 S3) 方框。在方框中输入存储桶和前缀（文件夹）名称。

> **Note**
>
> 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给托管式节点的实例配置文件的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确认与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

18. 选择创建关联。

创建运行 Chef 配方的关联（CLI）

以下过程介绍了如何使用 AWS Command Line Interface (AWS CLI) 创建 State Manager 关联，以便使用 `AWS-ApplyChefRecipes` 文档运行 Chef 说明书。

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关更多信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 运行以下 命令之一来创建关联，该关联通过具有指定标签的目标节点来运行 Chef 说明书。使用适合您说明书源类型和操作系统的命令。将每个 *example-resource-placeholder* 替换为您自己的信息。

   a. Git 源

      Linux & macOS

```
aws ssm create-association --name "AWS-ApplyChefRecipes" \
    --targets Key=tag:TagKey,Values=TagValue \
```

```
    --parameters '{"SourceType":["Git"],"SourceInfo":["{\"repository\":
\"repository-name\", \"getOptions\": \"branch:branch-name\", \"username
\": \"{{ ssm-secure:username-secure-string-parameter }}\", \"password\":
 \"{{ ssm-secure:password-secure-string-parameter }}\"}"], "RunList":
["{\"recipe[cookbook-name-1::recipe-name]\", \"recipe[cookbook-
name-2::recipe-name]\"}"], "JsonAttributesContent": ["{custom-json-
content}"], "JsonAttributesSources": "{\"sourceType\":\"s3\", \"sourceInfo
\":\"s3-bucket-endpoint-1\"}, {\"sourceType\":\"s3\", \"sourceInfo\":
\"s3-bucket-endpoint-2\"}", "ChefClientVersion": ["version-number"],
 "ChefClientArguments":["{chef-client-arguments}"], "WhyRun": boolean,
 "ComplianceSeverity": ["severity-value"], "ComplianceType":
 ["Custom:Chef"], "ComplianceReportBucket": ["s3-bucket-name"]}' \
    --association-name "name" \
    --schedule-expression "cron-or-rate-expression"
```

Windows

```
aws ssm create-association --name "AWS-ApplyChefRecipes" ^
    --targets Key=tag:TagKey,Values=TagValue ^
    --parameters '{"SourceType":["Git"],"SourceInfo":["{\"repository\":
\"repository-name\", \"getOptions\": \"branch:branch-name\", \"username
\": \"{{ ssm-secure:username-secure-string-parameter }}\", \"password\":
 \"{{ ssm-secure:password-secure-string-parameter }}\"}"], "RunList":
["{\"recipe[cookbook-name-1::recipe-name]\", \"recipe[cookbook-
name-2::recipe-name]\"}"], "JsonAttributesContent": ["{custom-json}"],
 "JsonAttributesSources": "{\"sourceType\":\"s3\", \"sourceInfo\":
\"s3-bucket-endpoint-1\"}, {\"sourceType\":\"s3\", \"sourceInfo\":
\"s3-bucket-endpoint-2\"}", "ChefClientVersion": ["version-number"],
 "ChefClientArguments":["{chef-client-arguments}"], "WhyRun": boolean,
 "ComplianceSeverity": ["severity-value"], "ComplianceType":
 ["Custom:Chef"], "ComplianceReportBucket": ["s3-bucket-name"]}' ^
    --association-name "name" ^
    --schedule-expression "cron-or-rate-expression"
```

b. GitHub 源

Linux & macOS

```
aws ssm create-association --name "AWS-ApplyChefRecipes" \
    --targets Key=tag:TagKey,Values=TagValue \
```

```
    --parameters '{"SourceType":["GitHub"],"SourceInfo":["{\"owner\":
\"owner-name\", \"repository\": \"name\", \"path\": \"path-to-directory-
or-cookbook-to-download\", \"getOptions\": \"branch:branch-name\"}"],
 "RunList":["{\"recipe[cookbook-name-1::recipe-name]\", \"recipe[cookbook-
name-2::recipe-name]\"}"], "JsonAttributesContent": ["{custom-json}"],
 "ChefClientVersion": ["version-number"], "ChefClientArguments":["{chef-
client-arguments}"], "WhyRun": boolean, "ComplianceSeverity": ["severity-
value"], "ComplianceType": ["Custom:Chef"], "ComplianceReportBucket": ["s3-
bucket-name"]}' \
    --association-name "name" \
    --schedule-expression "cron-or-rate-expression"
```

Windows

```
aws ssm create-association --name "AWS-ApplyChefRecipes" ^
    --targets Key=tag:TagKey,Values=TagValue \
    --parameters '{"SourceType":["GitHub"],"SourceInfo":["{\"owner\":
\"owner-name\", \"repository\": \"name\", \"path\": \"path-to-directory-
or-cookbook-to-download\", \"getOptions\": \"branch:branch-name\"}"],
 "RunList":["{\"recipe[cookbook-name-1::recipe-name]\", \"recipe[cookbook-
name-2::recipe-name]\"}"], "JsonAttributesContent": ["{custom-json}"],
 "ChefClientVersion": ["version-number"], "ChefClientArguments":["{chef-
client-arguments}"], "WhyRun": boolean, "ComplianceSeverity": ["severity-
value"], "ComplianceType": ["Custom:Chef"], "ComplianceReportBucket": ["s3-
bucket-name"]}' ^
    --association-name "name" ^
    --schedule-expression "cron-or-rate-expression"
```

见下列。

Linux & macOS

```
aws ssm create-association --name "AWS-ApplyChefRecipes" \
    --targets Key=tag:OS,Values=Linux \
    --parameters '{"SourceType":["GitHub"],"SourceInfo":["{\"owner
\":\"ChefRecipeTest\", \"repository\": \"ChefCookbooks\", \"path
\": \"cookbooks/HelloWorld\", \"getOptions\": \"branch:master
\"}"], "RunList":["{\"recipe[HelloWorld::HelloWorldRecipe]\",
 \"recipe[HelloWorld::InstallApp]\"}"], "JsonAttributesContent":
 ["{\"state\": \"visible\",\"colors\": {\"foreground\": \"light-blue
\",\"background\": \"dark-gray\"}}"], "ChefClientVersion": ["14"],
 "ChefClientArguments":["{--fips}"], "WhyRun": false, "ComplianceSeverity":
```

```
 ["Medium"], "ComplianceType": ["Custom:Chef"], "ComplianceReportBucket":
 ["ChefComplianceResultsBucket"]}' \
    --association-name "MyChefAssociation" \
    --schedule-expression "cron(0 2 ? * SUN *)"
```

Windows

```
aws ssm create-association --name "AWS-ApplyChefRecipes" ^
    --targets Key=tag:OS,Values=Linux ^
    --parameters '{"SourceType":["GitHub"],"SourceInfo":["{\"owner
\":\"ChefRecipeTest\", \"repository\": \"ChefCookbooks\", \"path
\": \"cookbooks/HelloWorld\", \"getOptions\": \"branch:master
\"}"], "RunList":["{\"recipe[HelloWorld::HelloWorldRecipe]\",
 \"recipe[HelloWorld::InstallApp]\"}"], "JsonAttributesContent":
 ["{\"state\": \"visible\",\"colors\": {\"foreground\": \"light-blue
\",\"background\": \"dark-gray\"}}"], "ChefClientVersion": ["14"],
 "ChefClientArguments":["{--fips}"], "WhyRun": false, "ComplianceSeverity":
 ["Medium"], "ComplianceType": ["Custom:Chef"], "ComplianceReportBucket":
 ["ChefComplianceResultsBucket"]}' ^
    --association-name "MyChefAssociation" ^
    --schedule-expression "cron(0 2 ? * SUN *)"
```

c. HTTP 源

Linux & macOS

```
aws ssm create-association --name "AWS-ApplyChefRecipes" \
    --targets Key=tag:TagKey,Values=TagValue \
    --parameters '{"SourceType":["HTTP"],"SourceInfo":["{\"url\":\"url-
to-zip-file|directory|cookbook\", \"authMethod\": \"auth-method\",
 \"username\": \"{{ ssm-secure:username-secure-string-parameter }}\",
 \"password\": \"{{ ssm-secure:password-secure-string-parameter }}\"}"],
 "RunList":["{\"recipe[cookbook-name-1::recipe-name]\", \"recipe[cookbook-
name-2::recipe-name]\"}"], "JsonAttributesContent": ["{custom-json-
content}"], "JsonAttributesSources": "{\"sourceType\":\"s3\", \"sourceInfo
\":\"s3-bucket-endpoint-1\"}, {\"sourceType\":\"s3\", \"sourceInfo\":
\"s3-bucket-endpoint-2\"}", "ChefClientVersion": ["version-number"],
 "ChefClientArguments":["{chef-client-arguments}"], "WhyRun": boolean,
 "ComplianceSeverity": ["severity-value"], "ComplianceType":
 ["Custom:Chef"], "ComplianceReportBucket": ["s3-bucket-name"]}' \
    --association-name "name" \
    --schedule-expression "cron-or-rate-expression"
```

Windows

```
aws ssm create-association --name "AWS-ApplyChefRecipes" ^
    --targets Key=tag:TagKey,Values=TagValue ^
    --parameters '{"SourceType":["HTTP"],"SourceInfo":["{\"url\":\"url-
to-zip-file|directory|cookbook\", \"authMethod\": \"auth-method\",
 \"username\": \"{{ ssm-secure:username-secure-string-parameter }}\",
 \"password\": \"{{ ssm-secure:password-secure-string-parameter }}\"}"],
 "RunList":["{\"recipe[cookbook-name-1::recipe-name]\", \"recipe[cookbook-
name-2::recipe-name]\"}"], "JsonAttributesContent": ["{custom-json-
content}"], "JsonAttributesSources": "{\"sourceType\":\"s3\", \"sourceInfo
\":\"s3-bucket-endpoint-1\"}, {\"sourceType\":\"s3\", \"sourceInfo\":
\"s3-bucket-endpoint-2\"}", "ChefClientVersion": ["version-number"],
 "ChefClientArguments":["{chef-client-arguments}"], "WhyRun": boolean,
 "ComplianceSeverity": ["severity-value"], "ComplianceType":
 ["Custom:Chef"], "ComplianceReportBucket": ["s3-bucket-name"]}' \
    --association-name "name" ^
    --schedule-expression "cron-or-rate-expression"
```

d. Amazon S3 源

Linux & macOS

```
aws ssm create-association --name "AWS-ApplyChefRecipes" \
    --targets Key=tag:TagKey,Values=TagValue \
    --parameters '{"SourceType":["S3"],"SourceInfo":["{\"path\":\"https://
s3.amazonaws.com/path_to_Zip_file,_directory,_or_cookbook_to_download\"}"],
 "RunList":["{\"recipe[cookbook_name1::recipe_name]\",
 \"recipe[cookbook_name2::recipe_name]\"}"], "JsonAttributesContent":
 ["{Custom_JSON}"], "ChefClientVersion": ["version_number"],
 "ChefClientArguments":["{chef_client_arguments}"], "WhyRun": true_or_false,
 "ComplianceSeverity": ["severity_value"], "ComplianceType":
 ["Custom:Chef"], "ComplianceReportBucket": ["amzn-s3-demo-bucket"]}' \
    --association-name "name" \
    --schedule-expression "cron_or_rate_expression"
```

Windows

```
aws ssm create-association --name "AWS-ApplyChefRecipes" ^
    --targets Key=tag:TagKey,Values=TagValue ^
    --parameters '{"SourceType":["S3"],"SourceInfo":["{\"path\":\"https://
s3.amazonaws.com/path_to_Zip_file,_directory,_or_cookbook_to_download\"}"],
```

```
 "RunList":["{\"recipe[cookbook_name1::recipe_name]\",
 \"recipe[cookbook_name2::recipe_name]\"}"], "JsonAttributesContent":
 ["{Custom_JSON}"], "ChefClientVersion": ["version_number"],
 "ChefClientArguments":["{chef_client_arguments}"], "WhyRun": true_or_false,
 "ComplianceSeverity": ["severity_value"], "ComplianceType":
 ["Custom:Chef"], "ComplianceReportBucket": ["amzn-s3-demo-bucket"]}' ^
    --association-name "name" ^
    --schedule-expression "cron_or_rate_expression"
```

见下列。

Linux & macOS

```
aws ssm create-association --name "AWS-ApplyChefRecipes" \
    --targets "Key=tag:OS,Values= Linux" \
    --parameters '{"SourceType":["S3"],"SourceInfo":["{\"path
\":\"https://s3.amazonaws.com/amzn-s3-demo-bucket/HelloWorld
\"}"], "RunList":["{\"recipe[HelloWorld::HelloWorldRecipe]\",
 \"recipe[HelloWorld::InstallApp]\"}"], "JsonAttributesContent":
 ["{\"state\": \"visible\",\"colors\": {\"foreground\": \"light-blue
\",\"background\": \"dark-gray\"}}"], "ChefClientVersion": ["14"],
 "ChefClientArguments":["{--fips}"], "WhyRun": false, "ComplianceSeverity":
 ["Medium"], "ComplianceType": ["Custom:Chef"], "ComplianceReportBucket":
 ["ChefComplianceResultsBucket"]}' \
    --association-name "name" \
    --schedule-expression "cron(0 2 ? * SUN *)"
```

Windows

```
aws ssm create-association --name "AWS-ApplyChefRecipes" ^
    --targets "Key=tag:OS,Values= Linux" ^
    --parameters '{"SourceType":["S3"],"SourceInfo":["{\"path
\":\"https://s3.amazonaws.com/amzn-s3-demo-bucket/HelloWorld
\"}"], "RunList":["{\"recipe[HelloWorld::HelloWorldRecipe]\",
 \"recipe[HelloWorld::InstallApp]\"}"], "JsonAttributesContent":
 ["{\"state\": \"visible\",\"colors\": {\"foreground\": \"light-blue
\",\"background\": \"dark-gray\"}}"], "ChefClientVersion": ["14"],
 "ChefClientArguments":["{--fips}"], "WhyRun": false, "ComplianceSeverity":
 ["Medium"], "ComplianceType": ["Custom:Chef"], "ComplianceReportBucket":
 ["ChefComplianceResultsBucket"]}' ^
    --association-name "name" ^
```

```
    --schedule-expression "cron(0 2 ? * SUN *)"
```

系统将创建关联，除非指定的 cron 或 rate 表达式阻止，否则系统将在目标节点上运行该关联。

> **Note**
>
> State Manager 关联不支持所有的 cron 和 rate 表达式。有关为关联创建 cron 和 rate 表达式的更多信息，请参阅 参考：适用于 Systems Manager 的 Cron 和 Rate 表达式。

3. 运行以下命令，查看您刚才创建的关联的状态。

```
aws ssm describe-association --association-id "ID"
```

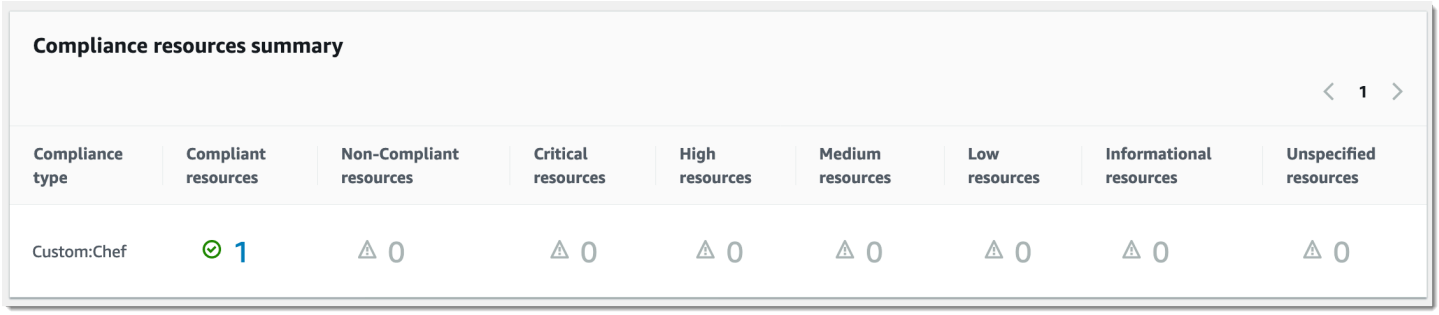
查看 Chef 资源合规性详细信息

Systems Manager 在运行 AWS-ApplyChefRecipes 文档时指定的 Amazon S3 合规性报告存储桶值中捕获有关 Chef 托管资源的合规性信息。在 S3 存储桶中搜索有关 Chef 资源失败的信息可能需要很长时间。相反，您可以在 Systems Manager Compliance (合规性) 页面上查看该信息。

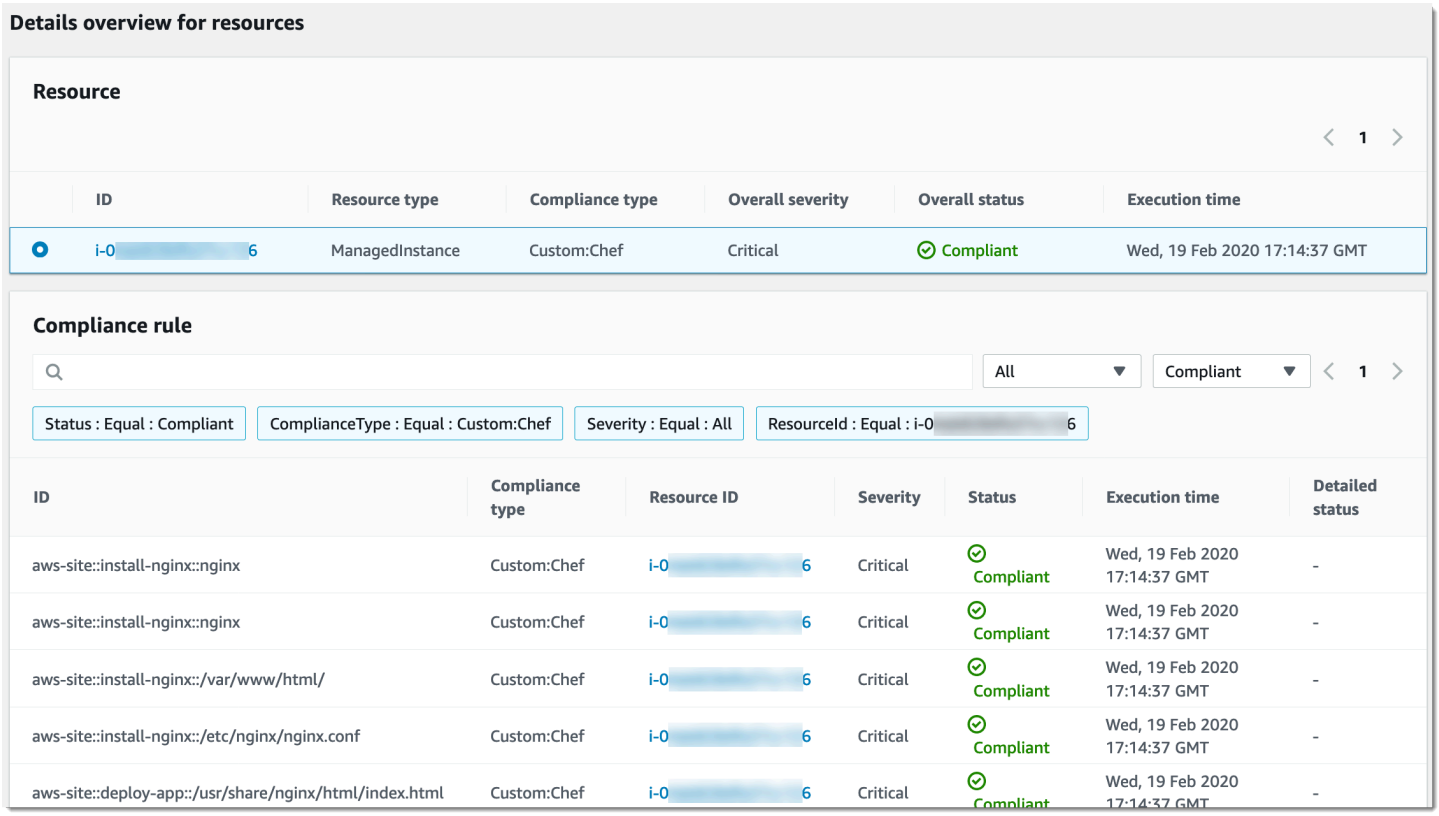
Systems Manager 合规性扫描收集在最近一次 Chef 运行中创建或检查的托管式节点上的资源的相关信息。资源可以包括文件、目录、systemd 服务、yum 软件包、模板化文件、gem 软件包和依赖说明书等。

合规性资源摘要部分显示失败的资源数。在以下示例中，ComplianceType 是 Custom:Chef，并且一个资源不合规。

> **Note**
>
> Custom:Chef 是 AWS-ApplyChefRecipes 文档中的默认 ComplianceType 值。该值是可自定义的。



Details overview for resources (资源的详细概述)部分显示有关不合规的 AWS 资源的信息。此部分还包括对其运行合规性的 Chef 资源类型、问题的严重性、合规性状态以及指向更多信息的链接（如适用）。



(View output) 查看输出 显示详细状态的最后 4,000 个字符。Systems Manager 先将异常作为第一个元素，查找详细消息，并显示它们，直至达到 4000 个字符的配额。该过程显示在引发异常之前输出的日志消息，这些消息是用于故障排除的最相关消息。

有关如何查看合规性信息的信息，请参阅 AWS Systems Manager Compliance。

> **Important**
>
> 如果 State Manager 关联失败，则不会报告合规性数据。例如，如果 Systems Manager 尝试从节点没有权限访问的 S3 存储桶中下载 Chef 说明书，则关联将失败，并且 Systems Manager 不会报告合规性数据。

演练：使用 AWS CLI 自动更新 SSM Agent

以下过程将指导您完成使用 AWS Command Line Interface 创建 State Manager 关联的过程。关联会根据您指定的计划自动更新 SSM Agent。有关 SSM Agent 的更多信息，请参阅 使用 SSM Agent。若要使用控制台自定义 SSM Agent，请参阅 自动更新 SSM Agent。

要获得有关 SSM Agent 更新的通知，请在 GitHub 上订阅 SSM Agent 发布说明页面。

开始前的准备工作

在完成以下过程之前，请确认您至少有一个适用于 Linux、macOS ，或针对 Systems Manager 而配置的 Windows Server 的 Amazon Elastic Compute Cloud (Amazon EC2)实例正在运行。有关更多信息，请参阅 为 AWS Systems Manager 设置托管节点。

如果您使用 AWS CLI 或 AWS Tools for Windows PowerShell 创建关联，请使用 `--Targets` 参数将实例指定为目标，如以下示例所示。不要使用 `--InstanceID` 参数。`--InstanceID` 参数是一个旧参数。

创建关联以实现 SSM Agent 的自动更新

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 运行以下命令，通过使用 Amazon Elastic Compute Cloud (Amazon EC2) 标签将实例设为目标来创建关联。将每个*#######*替换为您自己的信息。`Schedule` 参数可以设置一个计划，以在每周日凌晨 2:00 (UTC) 运行 关联。

   State Manager 关联不支持所有的 cron 和 rate 表达式。有关为关联创建 cron 和 rate 表达式的更多信息，请参阅 参考：适用于 Systems Manager 的 Cron 和 Rate 表达式。

   Linux & macOS

   ```
   aws ssm create-association \
   --targets Key=tag:tag_key,Values=tag_value \
   ```

```
--name AWS-UpdateSSMAgent \
--schedule-expression "cron(0 2 ? * SUN *)"
```

Windows

```
aws ssm create-association ^
--targets Key=tag:tag_key,Values=tag_value ^
--name AWS-UpdateSSMAgent ^
--schedule-expression "cron(0 2 ? * SUN *)"
```

您可以通过在以逗号分隔的列表中指定实例 ID 来将多个实例设为目标。

Linux & macOS

```
aws ssm create-association \
--targets Key=instanceids,Values=instance_ID,instance_ID,instance_ID \
--name AWS-UpdateSSMAgent \
--schedule-expression "cron(0 2 ? * SUN *)"
```

Windows

```
aws ssm create-association ^
--targets Key=instanceids,Values=instance_ID,instance_ID,instance_ID ^
--name AWS-UpdateSSMAgent ^
--schedule-expression "cron(0 2 ? * SUN *)"
```

您可以指定要更新到的 SSM Agent 版本。

Linux & macOS

```
aws ssm create-association \
--targets Key=instanceids,Values=instance_ID,instance_ID,instance_ID \
--name AWS-UpdateSSMAgent \
--schedule-expression "cron(0 2 ? * SUN *)" \
--parameters version=ssm_agent_version_number
```

Windows

```
aws ssm create-association ^
```

```
--targets Key=instanceids,Values=instance_ID,instance_ID,instance_ID ^
--name AWS-UpdateSSMAgent ^
--schedule-expression "cron(0 2 ? * SUN *)" ^
--parameters version=ssm_agent_version_number
```

系统将返回类似于以下内容的信息。

```
{
    "AssociationDescription": {
        "ScheduleExpression": "cron(0 2 ? * SUN *)",
        "Name": "AWS-UpdateSSMAgent",
        "Overview": {
            "Status": "Pending",
            "DetailedStatus": "Creating"
        },
        "AssociationId": "123.............",
        "DocumentVersion": "$DEFAULT",
        "LastUpdateAssociationDate": 1504034257.98,
        "Date": 1504034257.98,
        "AssociationVersion": "1",
        "Targets": [
            {
                "Values": [
                    "TagValue"
                ],
                "Key": "tag:TagKey"
            }
        ]
    }
}
```

系统将尝试在实例上创建关联并在创建后应用状态。关联状态显示 Pending。

3. 运行以下命令查看您创建的关联的更新状态。

```
aws ssm list-associations
```

如果您的实例没有运行 SSM Agent 的最新版本，则状态将显示 Failed。发布 SSM Agent 的新版本时，关联将自动安装新代理，而且状态将显示 Success。

演练：在适用于 Windows Server 的 EC2 实例上自动更新半虚拟化驱动程序

Amazon Windows Amazon Machine Images (AMIs) 包含一系列驱动程序，以允许访问虚拟化硬件。Amazon Elastic Compute Cloud (Amazon EC2) 使用这些驱动程序将实例存储和 Amazon Elastic Block Store (Amazon EBS) 卷映射到其设备。我们建议您安装最新的驱动程序来提高适用于 Windows Server 的 EC2 实例的稳定性和性能。有关半虚拟化驱动程序的更多信息，请参阅 AWS 半虚拟化驱动程序。

以下演练介绍如何配置State Manager关联，以便在驱动程序可用时自动下载和安装新的 AWS 半虚拟化驱动程序。State Manager是 AWS Systems Manager 中的一项工具。

开始前的准备工作

在完成以下过程之前，请确认您至少有一个适用于 Windows Server 的 Amazon EC2 实例（针对 Systems Manager 配置）正在运行。有关更多信息，请参阅 为 AWS Systems Manager 设置托管节点。

创建自动更新半虚拟化驱动程序的 State Manager 关联

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 State Manager。
3. 选择 Create association（创建关联）。
4. 在名称字段中，输入关联的描述性名称。
5. 在 Document (文档) 列表中，选择 `AWS-ConfigureAWSPackage`。
6. 在参数区域中，执行以下操作：

   - 对于操作，选择安装。
   - 适用于安装类型中，选择卸载并重新安装。

     > **Note**
     >
     > 此程序包不支持就地升级。必须卸载程序包进行重新安装。

   - 对于名称，请输入 **`AWSPVDriver`**。

     无需为版本和其他参数输入任何内容。
7. 在 Targets（目标）部分中，通过指定标签、手动选择实例或边缘设备或指定资源组，选择要在其上运行此操作的托管式节点。

> **Tip**
>
> 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

> **Note**
>
> 如果您选择使用标签将实例设为目标，并指定映射到 Linux 实例的标签，则关联在 Windows 实例上将成功，但在 Linux 实例上将失败。关联的总体状态将显示 Failed。

8. 在指定计划区域中，选择是按照配置的计划运行关联，还是只运行一次。更新的半虚拟化驱动程序每年发布若干次，因此如果需要，您可以计划每月运行一次关联。
9. 在高级选项中，为合规性严重级别选择关联的严重级别。合规性报告指示关联状态是合规还是不合规以及您在此处指示的严重级别。有关更多信息，请参阅 关于 State Manager 关联合规性。
10. 对于 Rate control（速率控制）：
    - 对于 Concurrency（并发），请指定要同时运行该命令的托管式节点的数量或百分比。

      > **Note**
      >
      > 如果您通过指定应用于托管式节点的标签或指定 AWS Resource Groups 来选择目标，但不确定有多少个托管式节点已被设为目标，则可通过指定百分比来限制可同时运行该文档的目标的数量。

    - 对于 Error threshold（错误阈值），请指定当命令在一定数量或百分比的节点上失败后，何时在其他托管式节点上停止运行该命令。例如，如果您指定三个错误，Systems Manager 将在收到第四个错误时停止发送该命令。仍在处理该命令的托管式节点也可能发送错误。
11. （可选）对于 Output options (输出选项)，要将命令输出保存到文件，请选中 Enable writing output to S3 (启用将输出写入 S3) 方框。在方框中输入存储桶和前缀（文件夹）名称。

> **Note**
>
> 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给托管式节点的实例配置文件的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶

位于不同的 AWS 账户 中，请确认与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

12. （可选）在 CloudWatch 警报部分中，为警报名称选择 CloudWatch 警报来应用到关联，以便进行监控。

> **Note**
>
> 请注意与该步骤相关的以下信息。
>
> - 警报列表最多显示 100 个警报。如果您未在列表中看到您的警报，请使用 AWS Command Line Interface 创建关联。有关更多信息，请参阅 创建关联（命令行）。
> - 要将 CloudWatch 警报附加到命令，创建关联的 IAM 主体必须具有 `iam:createServiceLinkedRole` 操作的权限。有关 CloudWatch 警报的更多信息，请参阅使用 Amazon CloudWatch 警报。
> - 如果您的警报激活，任何待处理的命令调用或自动化都不会运行。

13. 选择 Create Association，然后选择 Close。系统将尝试在实例上创建关联并立即应用状态。

    如果您在一个或多个适用于 Windows Server 的 Amazon EC2 实例上创建了关联，状态将更改为 Success (成功)。如果未为 Systems Manager 配置实例，或无意中将 Linux 实例设为了目标，则状态将显示 Failed (失败)。

    如果状态为 Failed (失败)，则选择关联 ID，选择资源选项卡并验证关联是否已在适用于 Windows Server 的 EC2 实例上成功创建。如果适用于 Windows Server 的 EC2 实例显示 Failed (失败) 状态，请验证 SSM Agent 是否在该实例上运行，并验证该实例是否配置有适用于 Systems Manager 的 AWS Identity and Access Management (IAM) 角色。有关更多信息，请参阅 为组织设置 Systems Manager 统一控制台。

AWS Systems Manager 更改管理工具

AWS Systems Manager 提供以下工具，用于对 AWS 资源进行更改。

主题

- AWS Systems Manager 自动化
- AWS Systems Manager Change Calendar
- AWS Systems Manager Change Manager

- AWS Systems Manager 文档
- AWS Systems Manager Maintenance Windows
- AWS Systems Manager Quick Setup

AWS Systems Manager 自动化

Automation 是 AWS Systems Manager 中的一项工具，其简化了 Amazon Elastic Compute Cloud（Amazon EC2）、Amazon Relational Database Service（Amazon RDS）、Amazon Redshift、Amazon Redshift、Amazon Simple Storage Service（Amazon S3）等 AWS 服务的常见维护、部署和修复任务。要开始使用自动化，请打开 Systems Manager 控制台。在导航窗格中，选择 自动化。

Automation 可帮助您构建用于大规模部署、配置和管理 AWS 资源的自动化解决方案。借助 Automation，您可以精确控制自动化的并发性。这意味着您可以指定同时定位多少资源，以及在停止自动化之前可能发生的错误数。

为了帮助您开始使用 Automation，AWS 开发和维护了几个预先定义的运行手册。根据您的应用场景，您可以使用这些预定义的运行手册来执行各种任务，或者创建更适合您的需求的自定义运行手册。要监控自动化的进度和状态，您可以使用 Systems Manager Automation 控制台或您的首选命令行工具。Automation 还与 Amazon EventBridge 集成，可帮助您大规模构建事件驱动型架构。

我的组织如何从 Automation 获益？

Automation 具备下列优势：

- 运行手册内容中的脚本支持

  使用 `aws:executeScript` 操作，您可以直接从运行手册中运行自定义 Python 和 PowerShell 函数。这使您可以更灵活地创建自定义运行手册，因为您可以完成其他 Automation 操作不支持的各种任务。您还可以更好地控制运行手册的逻辑。有关如何使用此操作以及如何帮助改进现有自动化解决方案的示例，请参阅 创作自动化运行手册。
- 从一个集中位置跨多个 AWS 账户 和 AWS 区域 运行自动化

  管理员可以从 Systems Manager 控制台跨多个账户和区域对资源运行自动化。
- 增强操作安全

管理员可以集中地授予和撤销对运行手册的访问。仅使用 AWS Identity and Access Management (IAM) 策略，您就可以控制贵企业中的哪些个人用户或组能够使用 Automation 以及他们能够访问的运行手册。

- 自动执行常见 IT 任务

  自动执行常见任务可以帮助提高运营效率、强制执行组织标准并减少操作员错误。例如，您可以使用 `AWS-UpdateCloudFormationStackWithApproval` 运行手册来更新通过使用 AWS CloudFormation 模板部署的资源。更新会应用新模板。在更新开始之前，您可以配置 Automation 以请求由一个或多个用户批准。

- 安全地批量执行中断性任务

  Automation 包含诸如速率控制之类的功能，使您可以通过指定并发值和错误阈值来控制自动化在整个机群上的部署。有关使用速率控制的更多信息，请参阅 大规模运行自动化操作。

- 简化复杂任务

  Automation 提供了预定义的运行手册，可简化复杂而耗时的任务，例如创建黄金 Amazon Machine Images (AMIs)。例如，您可以使用 `AWS-UpdateLinuxAmi` 和 `AWS-UpdateWindowsAmi` 运行手册从源 AMI 创建黄金 AMIs。使用这些运行手册，您可以在应用更新前后运行自定义脚本。您还可以包含特定软件包或从安装中排除这些软件包。有关如何使用这些运行手册的示例，请参阅 教程。

- 为输入定义约束条件

  您可以在自定义运行手册中定义约束条件，以限制 Automation 将为特定输入参数接受的值。例如，`allowedPattern` 将只接受与您定义的正则表达式匹配的输入参数的值。如果您为输入参数指定 `allowedValues`，只接受您在运行手册中指定的值。

- 将自动化操作输出记录到 Amazon CloudWatch Logs 中

  为了满足组织的操作或安全要求，您可能需要提供在运行手册期间运行的脚本记录。使用 CloudWatch Logs，您可以监控、存储和访问各种 AWS 服务 的日志文件。您可以将 `aws:executeScript` 操作的输出发送到 CloudWatch Logs 日志组中用于调试和故障排除目的。在将日志数据发送到日志组时，可以使用您的 KMS 密钥进行 AWS KMS 加密，也可以不加密。有关更多信息，请参阅 使用 CloudWatch Logs 记录自动化操作输出。

- Amazon EventBridge 集成

  支持将 Automation 作为 Amazon EventBridge 规则中的目标类型。这意味着您可以通过使用事件触发运行手册。有关更多信息，请参阅 使用 Amazon EventBridge 监控 Systems Manager 事件 和 引用：Amazon EventBridge 事件模式和 Systems Manager 类型。

- 分享组织最佳实践

  您可以在跨账户和区域共享的运行手册中定义资源管理、运营任务等的最佳实践。

谁应该使用 Automation？

- 任何希望大规模提高其运营效率、减少与手动干预相关的错误以及缩短解决常见问题的时间的 AWS 客户。
- 希望自动执行部署和配置任务的基础设施专家。
- 想要可靠地解决常见问题、提高故障排除效率和减少重复性操作的管理员。
- 想要自动执行通常需要手动执行的任务的用户。

什么是自动化？

自动化由运行手册中定义的所有任务组成，并由 Automation 服务执行。Automation 使用以下组件运行自动化。

概念	详细信息
自动化运行手册	Systems Manager 自动化运行手册定义 Systems Manager 对托管式节点和 AWS 资源执行的自动化操作。自动化包含几个预定义的自动化运行手册，您可以用它们来执行常见任务，例如重启一个或多个 Amazon EC2 实例，或创建 Amazon Machine Image (AMI)。您也可以创建自己的运行手册。运行手册使用 YAML 或 JSON,，并包括您指定的步骤和参数。步骤按先后顺序运行。有关更多信息，请参阅 创建您自己的运行手册。 运行手册是 `Automation` 类型的 Systems Manager 文档，而不是`Command`、`Policy`、`Session` 文档。运行手册支持 0.3 版架构。命令文档使用 1.2、2.0 或 2.2 版本的架构。策略文档使用 2.0 或更高版本的架构。

概念	详细信息
自动化操作	运行手册中定义的自动化包括一个或多个步骤。每个步骤都与特定操作关联。此操作确定本步的输入、行为和输出。在运行手册的 `mainSteps` 部分定义步骤。自动化支持 20 种不同的操作类型。有关更多信息，请参阅 Systems Manager 自动化操作参考。
自动化配额	每个 AWS 账户 账户可以同时运行 100 项自动化。这包括子自动化（由其他自动化启动的自动化）和速率控制自动化。如果您尝试运行的自动化多于此数量，Systems Manager 会将额外的自动化添加到一个队列中，状态显示为“待处理”。此限额可使用自适应并发来调整。有关更多信息，请参阅 允许 Automation 适应您的并发需求。有关运行自动化的更多信息，请参阅 运行由 Systems Manager Automation 支持的自动化操作。
自动化队列配额	如果尝试运行的自动化数量超过并发自动化限制，后续自动化将添加到队列中。每个 AWS 账户可以排列 5000 个自动化。自动化完成（或达到最终状态）后，队列中的第一个自动化将启动。
速率控制自动化配额	每个 AWS 账户 可以同时运行 25 项速率控制自动化。如果您尝试运行的速率控制自动化多于并发费率控制自动化限制，Systems Manager 会将后续速率控制自动化添加到一个队列中，状态显示为“待处理”。有关运行速率控制自动化的更多信息，请参阅 大规模运行自动化操作。

概念	详细信息
速率控制自动化队列配额	如果尝试运行的自动化数量超过并发速率控制自动化限制，后续自动化将添加到队列中。每个 AWS 账户 可以排列 1000 个速率控制自动化。自动化完成（或达到最终状态）后，队列中的第一个自动化将启动。

主题

- 设置自动化
- 运行由 Systems Manager Automation 支持的自动化操作
- 运行需要审批的自动化
- 大规模运行自动化操作
- 在多个 AWS 区域 和账户中运行自动化
- 基于 EventBridge 事件运行自动化
- 要按步骤运行自动化
- 使用 State Manager 关联调度自动化
- 使用维护窗口计划自动化
- Systems Manager 自动化操作参考
- 创建您自己的运行手册
- Systems Manager 自动化运行手册参考
- 教程
- 了解 Systems Manager Automation 返回的状态
- Systems Manager 自动化故障排除

设置自动化

要设置 Automation（AWS Systems Manager 中的一项工具），您必须确保用户对 Automation 服务具有访问权限并按情景配置角色，以便服务可在您的资源上执行操作。我们还建议您在 Automation 首选项中选择使用自适应并发模式。自适应并发会自动扩展您的自动化配额以满足您的需求。有关更多信息，请参阅 允许 Automation 适应您的并发需求。

为确保正确访问 AWS Systems Manager 自动化，请查看以下用户和服务角色要求。

验证用户的运行手册访问权限

确认您有权使用运行手册。如果已为您的用户、组或角色分配了管理员权限，则您有权访问 Systems Manager 自动化。如果您没有管理员权限，则管理员必须通过向您的用户、组或角色分配 `AmazonSSMFullAccess` 托管策略或提供类似权限的策略，来向您授予权限。

> **Important**
>
> IAM policy `AmazonSSMFullAccess` 授予对 Systems Manager 进行操作的权限。但是，某些运行手册需要其他服务的权限，例如运行手册 `AWS-ReleaseElasticIP`，它需要 `ec2:ReleaseAddress` 的 IAM 权限。因此，您必须查看运行手册中采取的操作，以确保为您的用户、组或角色分配了执行运行手册包括的操作所需的权限。

为自动化配置服务角色（担任角色）访问权限

自动化可以在服务角色（或代入角色）的上下文下启动。这样服务就能够代表您执行操作。如果未指定担任角色，则自动化将使用调用了自动化的用户的上下文。

不过，以下情况需要您为自动化指定服务角色：

- 当您想限制用户对资源的权限，但希望用户运行需要提升权限的自动化时。在这种情况下，您可以创建具有提升权限的服务角色并允许用户运行自动化。
- 创建 Systems Manager State Manager 关联，运行一个运行手册。
- 您有运行时长预计将超过 12 小时的操作时。
- 在运行不归 Amazon 所有并使用 `aws:executeScript` 操作调用 AWS API 操作或对 AWS 资源执行操作的运行手册时。有关信息，请参阅使用运行手册的权限。

如果您需要为自动化创建一个服务角色，可以使用以下方法之一。

主题

- 通过使用 AWS CloudFormation 为 Automation 创建服务角色
- 使用控制台为 Automation 创建服务角色
- 允许 Automation 适应您的并发需求
- 为自动化实施更改控制

通过使用 AWS CloudFormation 为 Automation 创建服务角色

您可以通过 AWS CloudFormation 模板为 Automation（AWS Systems Manager 中的一项工具）创建一个服务角色。创建服务角色后，可以使用参数 `AutomationAssumeRole` 在运行手册中指定服务角色。

使用 AWS CloudFormation 创建服务角色

使用 AWS CloudFormation，通过以下过程为 Systems Manager 自动化创建所需的 AWS Identity and Access Management (IAM) 角色。

创建所需的 IAM 角色

1. 下载并解压缩 `AWS-SystemsManager-AutomationServiceRole.zip` 文件。此文件夹包含 `AWS-SystemsManager-AutomationServiceRole.yaml` AWS CloudFormation 模板文件。
2. 通过以下网址打开 AWS CloudFormation 控制台：https://console.aws.amazon.com/cloudformation。
3. 选择创建堆栈。
4. 在指定模板部分，选择上传模板文件。
5. 选择浏览，然后选择 `AWS-SystemsManager-AutomationServiceRole.yaml` AWS CloudFormation 模板文件。
6. 选择下一步。
7. 在指定堆栈详细信息页面的堆栈名称字段中，输入名称。
8. 在配置堆栈选项页面上，您无需进行任何选择。选择下一步。
9. 在审核页面上，向下滚动并选择我确认 AWS CloudFormation 可能创建 IAM 资源选项。
10. 选择创建。

大约三分钟左右，CloudFormation 会显示 CREATE_IN_PROGRESS 状态。创建堆栈并且您的角色准备好使用之后，状态会变为 CREATE_COMPLETE。

> **Important**
>
> 如果您运行使用 AWS Identity and Access Management (IAM) 服务角色调用其他服务的自动化工作流程，请注意必须使用权限将该服务角色配置为调用这些服务。该要求适用于所有 AWS 自动化运行手册（`AWS-*` 运行手册），例如 `AWS-ConfigureS3BucketLogging`、`AWS-CreateDynamoDBBackup` 和 `AWS-RestartEC2Instance` 运行手册等。对于您创建的任何自定义自动化运行手册，如果这些

文档使用调用其他服务的操作来调用其他 AWS 服务，则此要求同样适用。例如，如果使用 `aws:executeAwsApi`、`aws:createStack` 或 `aws:copyImage` 操作，则您必须配置具有权限的服务角色来调用这些服务。您可以将 IAM 内联策略添加到该角色，从而向其他 AWS 服务授予权限。有关更多信息，请参阅 [（可选）添加自动化内联策略或客户管理型策略来调用其他 AWS 服务](#)。

复制自动化的角色信息

使用以下过程从 AWS CloudFormation 控制台复制关于自动化服务角色的信息。当您使用运行手册时，必须指定这些角色。

> **Note**
>
> 如果您运行 `AWS-UpdateLinuxAmi` 或 `AWS-UpdateWindowsAmi` 运行手册，则无需使用此过程来复制角色信息。这些运行手册已将必需角色指定为默认值。这些运行手册中指定的角色使用 IAM 托管式策略。

复制角色名称

1. 通过以下网址打开 AWS CloudFormation 控制台：https://console.aws.amazon.com/cloudformation。
2. 选择您在前一个过程中创建的自动化 Stack name (堆栈名称)。
3. 选择资源选项卡。
4. 选择 AutomationServiceRole 的物理 ID 链接 IAM 控制台将打开并显示自动化服务角色的摘要。
5. 复制角色 ARN 旁边的 Amazon Resource Name (ARN)。该 ARN 类似于以下这样：`arn:aws:iam::12345678:role/AutomationServiceRole`。
6. 将 ARN 粘贴到文本文件供以后使用。

您已完成了为自动化配置服务角色。现在，您可在运行手册中使用自动化服务角色 ARN。

使用控制台为 Automation 创建服务角色

如果需要为 Automation（AWS Systems Manager 中的一项工具）创建一个服务角色，请完成以下任务。有关自动化何时需要服务角色的更多信息，请参阅 [设置自动化](#)。

任务

- 任务 1：为自动化创建服务角色
- 任务 2：将 iam:PassRole 策略附加到您的自动化角色

任务 1：为自动化创建服务角色

使用以下过程为 Systems Manager 自动化创建服务角色（或承担角色）。

> **Note**
>
> 您也可以在运行手册中使用此角色，例如 `AWS-CreateManagedLinuxInstance` 运行手册。使用此角色或 AWS Identity and Access Management (IAM) 角色的 Amazon Resource Name (ARN)，在运行手册中允许自动化在您的环境中执行操作，例如启动新实例和代表您执行操作。

创建 IAM 角色并允许自动化担任该角色

1. 通过以下网址打开 IAM 控制台：https://console.aws.amazon.com/iam/。
2. 在导航窗格中，选择 Roles (角色)，然后选择 Create role (创建角色)。
3. 在 Select type of trusted entity (选择受信任实体的类型) 下，选择 AWS service (Amazon Web Services 服务)。
4. 在 Choose a use case (选择使用场景) 部分，选择 Systems Manager，然后选择 Next: Permissions (下一步：权限)。
5. 在附加的权限策略页面中，搜索 AmazonSSMAutomationRole 策略，选择它，然后选择下一步：审核。
6. 在审核页面上，在角色名称框中输入名称，然后输入描述。
7. 选择创建角色。系统将让您返回到 角色 页面。
8. 在角色页面中，选择刚刚创建的角色以打开摘要页面。记下角色名称和角色 ARN。在下一步骤中，当您将 iam:PassRole 策略附加到您的 IAM 账户时，将指定该角色 ARN。您还可以在运行手册中指定角色名称和 ARN。

> **Note**
>
> `AmazonSSMAutomationRole` 策略会将自动化角色权限分配给您账户内 AWS Lambda 函数的子集。这些函数以“自动化”开头。如果计划通过 Lambda 函数使用 自动化，则 Lambda ARN 必须使用以下格式：

"arn:aws:lambda:*:*:**function:Automation***"
如果现有 Lambda 函数的 ARN 未使用此格式，您还必须为自动化角色附加额外的 Lambda 策略，例如 AWSLambdaRole 策略。其他策略或角色必须提供对 AWS 账户 账户内 Lambda 函数更宽泛的访问权限。

创建服务角色后，我们建议编辑信任策略，以帮助防止跨服务的混淆代理问题。混淆代理问题是一个安全问题，即没有执行操作权限的实体可能会迫使更具权限的实体执行该操作。在 AWS 中，跨服务模拟可能会导致混淆代理问题。一个服务（呼叫服务）调用另一项服务（所谓的服务）时，可能会发生跨服务模拟。可以操纵调用服务，使用其权限以在其他情况下该服务不应有访问权限的方式对另一个客户的资源进行操作。为防止这种情况，AWS 提供可帮助您保护所有服务的数据的工具，而这些服务中的服务主体有权限访问账户中的资源。

我们建议使用资源策略中的 aws:SourceArn 和 aws:SourceAccount 全局条件上下文键，限制 Automation 为另一项服务提供的资源访问权限。如果 aws:SourceArn 值不包含账户 ID，例如 Amazon S3 存储桶 ARN，您必须使用两个全局条件上下文密钥来限制权限。如果同时使用全局条件上下文密钥和包含账户 ID 的 aws:SourceArn 值，则 aws:SourceAccount 值和 aws:SourceArn 值中的账户在同一策略语句中使用时，必须使用相同的账户 ID。如果您只希望将一个资源与跨服务访问相关联，请使用 aws:SourceArn。如果您想允许该账户中的任何资源与跨服务使用操作相关联，请使用 aws:SourceAccount。aws:SourceArn 的值必须是自动化执行的 ARN。如果您不知道资源的完整 ARN，或正在指定多个资源，请针对 ARN 未知部分使用带有通配符 (*) 的 aws:SourceArn 全局上下文条件键。例如 arn:aws:ssm:*:123456789012:automation-execution/*。

以下示例演示如何使用适用于自动化的 aws:SourceArn 和 aws:SourceAccount 全局条件上下文键，防止发生混淆代理问题。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "ssm.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole",
```

```
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:ssm:*:123456789012:automation-execution/*"
        }
      }
    }
  ]
}
```

要修改角色的信任策略

1. 通过以下网址打开 IAM 控制台：https://console.aws.amazon.com/iam/。
2. 在导航窗格中，选择角色。
3. 在账户的角色列表中，请选择 Automation 服务角色的名称。
4. 选择 信任关系 选项卡，然后选择 编辑信任关系。
5. 使用适用于 Automation 的 `aws:SourceArn` 和 `aws:SourceAccount` 全局条件上下文键编辑信任策略，以防止发生混淆代理问题。
6. 要保存更改，请选择 Update Trust Policy（更新信任策略）。

（可选）添加自动化内联策略或客户管理型策略来调用其他 AWS 服务

如果您运行使用 IAM 服务角色来调用其他 AWS 服务的自动化，则必须为该服务角色配置调用这些服务的权限。该要求适用于所有 AWS 自动化运行手册（`AWS-*` 运行手册），例如 `AWS-ConfigureS3BucketLogging`、`AWS-CreateDynamoDBBackup` 和 `AWS-RestartEC2Instance` 运行手册等。对于您创建的任何自定义运行手册，如果这些文档使用调用其他服务的操作来调用其他 AWS 服务，则此要求同样适用。例如，如果使用 `aws:executeAwsApi`、`aws:CreateStack` 或 `aws:copyImage` 操作等，则您必须配置具有权限的服务角色来调用这些服务。您可以将 IAM 内联策略或客户管理型策略添加到该角色，从而向其他 AWS 服务 授予权限。

为服务角色嵌入内联策略（IAM 控制台）

1. 登录 AWS Management Console，然后通过以下网址打开 IAM 控制台：https://console.aws.amazon.com/iam/。
2. 在导航窗格中，选择角色。

3. 在列表中，选择您要编辑的角色的名称。
4. 选择权限项卡。
5. 在添加权限下拉列表中，选择附加策略或创建内联策略。
6. 如果选择附加策略，请选中要添加的策略旁边的复选框，然后选择添加权限。
7. 如果选择创建策略，请选择 JSON 选项卡。
8. 输入您希望调用的 AWS 服务的 JSON 策略文档。以下是两个示例 JSON 策略文档。

   Amazon S3 PutObject 和 GetObject 示例

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
    ]
}
```

   Amazon EC2 CreateSnapshot 和 DescribeSnapShots 示例

   JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":"ec2:CreateSnapshot",
         "Resource":"*"
      },
```

```
        {
            "Effect":"Allow",
            "Action":"ec2:DescribeSnapshots",
            "Resource":"*"
        }
    ]
}
```

有关 IAM policy 语言的详细信息，请参阅《IAM 用户指南》中的 IAM JSON 策略参考。

9. 完成后，选择查看策略。策略验证程序将报告任何语法错误。
10. 在查看策略页面上，为创建的策略输入名称。查看策略摘要以查看您的策略授予的权限。然后，选择创建策略以保存您的工作。
11. 创建内联策略后，它会自动嵌入您的角色。

任务 2：将 iam:PassRole 策略附加到您的自动化角色

使用以下过程将 `iam:PassRole` 策略附加到您的自动化服务角色。这让 Automation 服务能够在运行自动化时将角色传递给其他服务或 Systems Manager 工具。

将 iam:PassRole 策略附加到您的自动化角色

1. 在刚刚创建的角色的摘要页面中，选择权限选项卡。
2. 选择 Add inline policy (添加内联策略)。
3. 在 Create policy (创建策略) 页面上，选择 Visual editor (可视化编辑器) 选项卡。
4. 选择服务，然后选择 IAM。
5. 选择选择操作。
6. 在筛选操作文本框中，键入 **`PassRole`**，然后选择 PassRole 选项。
7. 选择资源。确保已选择特定，然后选择添加 ARN。
8. 在为角色指定 ARN 字段中，粘贴您在任务 1 结束时复制的自动化角色 ARN。系统会自动填充账户和具有路径的角色名称字段。

> **Note**
>
> 如果希望自动化服务角色将 IAM 实例配置文件角色附加到 EC2 实例，则必须添加 IAM 实例配置文件角色的 ARN。这使自动化服务角色可以将 IAM 实例配置文件角色传递给目标 EC2 实例。

9. 选择添加。
10. 选择查看策略。
11. 在 Review Policy (审核策略) 页面上输入一个名称，然后选择 Create Policy (创建策略)。

允许 Automation 适应您的并发需求

默认情况下，Automation 允许您一次最多运行 100 次并发自动化。Automation 还提供了一个可选设置，您可以使用该设置自动调整并发自动化配额。使用此设置，您的并发自动化配额最多可容纳 500 个并发自动化，具体取决于可用资源。

> **Note**
>
> 如果您的自动化调用 API 操作，自适应地扩展到目标可能会导致限制异常。如果在启用自适应并发的情况下运行自动化时反复出现限制异常，则可能必须请求提高 API 操作的配额（如果可用）。

要使用 AWS Management Console 开启自适应并发

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 自动化。
3. 选择首选项选项卡，然后选择编辑。
4. 选中 Enable adaptive concurrency（启用自适应并发）旁边的复选框。
5. 选择保存。

要使用命令行开启自适应并发

- 打开 AWS CLI 或 Tools for Windows PowerShell，运行以下命令，在请求区域中为您的账户开启自适应并发。

Linux & macOS

```
aws ssm update-service-setting \
    --setting-id /ssm/automation/enable-adaptive-concurrency \
    --setting-value True
```

Windows

```
aws ssm update-service-setting ^
    --setting-id /ssm/automation/enable-adaptive-concurrency ^
    --setting-value True
```

PowerShell

```
Update-SSMServiceSetting `
    -SettingId "/ssm/automation/enable-adaptive-concurrency" `
    -SettingValue "True"
```

为自动化实施更改控制

默认情况下，自动化允许您不受日期和时间限制地使用运行手册。通过将自动化与 Change Calendar 集成，您可以对您的 AWS 账户 中的所有自动化实施更改控制。借助此设置，您账户中的 AWS Identity and Access Management（IAM）主体只能在更改日历允许的时间段内运行自动化。要了解有关使用 Change Calendar 的更多信息，请参阅 使用 Change Calendar。

打开更改控制（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 自动化。
3. 选择首选项选项卡，然后选择编辑。
4. 选择打开 Change Calendar 集成旁边的复选框。
5. 在选择更改日历下拉列表中，选择您希望自动化遵循的更改日历。
6. 选择保存。

运行由 Systems Manager Automation 支持的自动化操作

在您运行自动化时，默认情况下，自动化将在启动该自动化的用户的上下文中运行。也就是说，例如，如果您的用户拥有管理员权限，则该自动化运行时也拥有管理员权限，并且对该自动化配置的资源拥有完全访问权限。作为最佳安全实践，建议使用配置有 AmazonSSMAutomationRole 托管式策略的 IAM 服务角色（在本例中也称为担任角色）运行自动化。您可能需要将其他 IAM policy 添加到您的担任角色，以使用各种运行手册。使用 IAM 服务角色运行自动化称为委托管理。

如果使用服务角色，则允许对 AWS 资源运行自动化，但运行自动化的用户对这些资源的访问受限（或无法访问）。例如，您可以配置一个服务角色，并将其与自动化配合使用以自动重启一个或多个 Amazon Elastic Compute Cloud (Amazon EC2) 实例。Automation 是 AWS Systems Manager 中的一项工具。自动化会重启实例，但服务角色不会授予用户访问这些实例的权限。

可以在您运行自动化的运行时中指定服务角色，也可以创建自定义自动化运行手册，并直接在运行手册中指定服务角色。无论您是在运行时还是在运行手册中指定了服务角色，服务都会在所指定服务角色的上下文中运行。如果您未指定服务角色，则系统会在用户上下文中创建临时会话并运行自动化。

> **Note**
>
> 对于预计运行时长超过 12 小时的自动化，必须指定服务角色。如果您在用户环境中启动了长时间运行的自动化，用户的临时会话会在 12 小时后过期。

委托管理可确保更高的安全性，还能更好地控制您的 AWS 资源。它还能够增强审核体验，因为针对您的资源进行的操作是由集中服务角色执行的，而不是由多个 IAM 账户执行。

开始前的准备工作

在您完成以下过程之前，必须创建 IAM 服务角色并为 Automation（AWS Systems Manager 中的一项工具）配置信任关系。有关更多信息，请参阅 任务 1：为自动化创建服务角色。

以下过程介绍了如何使用 Systems Manager 控制台或您首选的命令行工具运行简单的自动化。

运行简单的自动化（控制台）

以下过程介绍了如何使用 Systems Manager 控制台运行简单的自动化。

运行简单的自动化

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。

2. 在导航窗格中，选择自动化，然后选择执行自动化。
3. 在自动化文档列表中，请选择运行手册。在文档类别窗格中选择一个或多个选项，以便根据 SSM 文档的用途对其进行筛选。要查看您拥有的运行手册，请选择我拥有的选项卡。要查看与您的账户共享的运行手册，请选择与我共享选项卡。要查看所有运行手册，请选择所有文档选项卡。

   > **Note**
   >
   > 您可以通过选择运行手册名称来查看有关该手册的信息。

4. 在文档详细信息部分中，验证文档版本已设置为要运行的版本。系统包括以下版本选项：
   - 运行时的默认版本 – 如果定期更新自动化运行手册并分配新的默认版本，请选择此选项。
   - 运行时的最新版本 – 如果定期更新自动化运行手册并且想要运行最新更新的版本，请选择此选项。
   - 1（默认）– 选择此选项可执行文档的第一个版本，即默认版本。
5. 选择下一步。
6. 在执行模式部分中，选择简单执行。
7. 在 输入参数 部分中，指定所需的输入。或者，您也可以从 AutomationAssumeRole 列表选择一个 IAM 服务角色。
8. （可选）选择一个 CloudWatch 警报以应用于您的自动化进行监控。要将 CloudWatch 警报附加到自动化，启动自动化的 IAM 主体必须具有 `iam:createServiceLinkedRole` 操作的权限。有关 CloudWatch 警报的更多信息，请参阅使用 Amazon CloudWatch 警报。请注意，如果您的警报激活，自动化将停止。如果使用 AWS CloudTrail，您将在跟踪中看到 API 调用。
9. 选择执行。

控制台将显示自动化的状态。如果自动化无法运行，请参阅 Systems Manager 自动化故障排除。

运行简单的自动化（命令行）

以下过程介绍了如何使用 AWS CLI（在 Linux 或 Windows 上）或 AWS Tools for PowerShell 运行简单的自动化。

运行简单的自动化

1. 安装并配置 AWS CLI 或 AWS Tools for PowerShell（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本以及安装 AWS Tools for PowerShell。

2. 运行以下命令以启动一个简单的自动化。将每个*#######*替换为您自己的信息。

Linux & macOS

```
aws ssm start-automation-execution \
    --document-name runbook name \
    --parameters runbook parameters
```

Windows

```
aws ssm start-automation-execution ^
    --document-name runbook name ^
    --parameters runbook parameters
```

PowerShell

```
Start-SSMAutomationExecution `
  -DocumentName runbook name `
  -Parameter runbook parameters
```

以下是使用 AWS-RestartEC2Instance 运行手册重新启动指定 EC2 实例的示例。

Linux & macOS

```
aws ssm start-automation-execution \
    --document-name "AWS-RestartEC2Instance" \
    --parameters "InstanceId=i-02573cafcfEXAMPLE"
```

Windows

```
aws ssm start-automation-execution ^
    --document-name "AWS-RestartEC2Instance" ^
    --parameters "InstanceId=i-02573cafcfEXAMPLE"
```

PowerShell

```
Start-SSMAutomationExecution `
  -DocumentName AWS-RestartEC2Instance `
  -Parameter @{"InstanceId"="i-02573cafcfEXAMPLE"}
```

系统将返回类似于以下内容的信息。

Linux & macOS

```
{
    "AutomationExecutionId": "4105a4fc-f944-11e6-9d32-0123456789ab"
}
```

Windows

```
{
    "AutomationExecutionId": "4105a4fc-f944-11e6-9d32-0123456789ab"
}
```

PowerShell

```
4105a4fc-f944-11e6-9d32-0123456789ab
```

3. 运行以下命令以检索自动化的状态。

Linux & macOS

```
aws ssm describe-automation-executions \
    --filter "Key=ExecutionId,Values=4105a4fc-f944-11e6-9d32-0123456789ab"
```

Windows

```
aws ssm describe-automation-executions ^
    --filter "Key=ExecutionId,Values=4105a4fc-f944-11e6-9d32-0123456789ab"
```

PowerShell

```
Get-SSMAutomationExecutionList | `
  Where {$_.AutomationExecutionId -eq "4105a4fc-f944-11e6-9d32-0123456789ab"}
```

系统将返回类似于以下内容的信息。

Linux & macOS

```
{
    "AutomationExecutionMetadataList": [
        {
            "AutomationExecutionStatus": "InProgress",
            "CurrentStepName": "stopInstances",
            "Outputs": {},
            "DocumentName": "AWS-RestartEC2Instance",
            "AutomationExecutionId": "4105a4fc-f944-11e6-9d32-0123456789ab",
            "DocumentVersion": "1",
            "ResolvedTargets": {
                "ParameterValues": [],
                "Truncated": false
            },
            "AutomationType": "Local",
            "Mode": "Auto",
            "ExecutionStartTime": 1564600648.159,
            "CurrentAction": "aws:changeInstanceState",
            "ExecutedBy": "arn:aws:sts::123456789012:assumed-role/Administrator/
Admin",
            "LogFile": "",
            "Targets": []
        }
    ]
}
```

Windows

```
{
    "AutomationExecutionMetadataList": [
        {
            "AutomationExecutionStatus": "InProgress",
            "CurrentStepName": "stopInstances",
            "Outputs": {},
            "DocumentName": "AWS-RestartEC2Instance",
            "AutomationExecutionId": "4105a4fc-f944-11e6-9d32-0123456789ab",
            "DocumentVersion": "1",
            "ResolvedTargets": {
                "ParameterValues": [],
                "Truncated": false
            },
```

```
            "AutomationType": "Local",
            "Mode": "Auto",
            "ExecutionStartTime": 1564600648.159,
            "CurrentAction": "aws:changeInstanceState",
            "ExecutedBy": "arn:aws:sts::123456789012:assumed-role/Administrator/
Admin",
            "LogFile": "",
            "Targets": []
        }
    ]
}
```

PowerShell

```
AutomationExecutionId       : 4105a4fc-f944-11e6-9d32-0123456789ab
AutomationExecutionStatus   : InProgress
AutomationType              : Local
CurrentAction               : aws:changeInstanceState
CurrentStepName             : startInstances
DocumentName                : AWS-RestartEC2Instance
DocumentVersion             : 1
ExecutedBy                  : arn:aws:sts::123456789012:assumed-role/
Administrator/Admin
ExecutionEndTime            : 1/1/0001 12:00:00 AM
ExecutionStartTime          : 7/31/2019 7:17:28 PM
FailureMessage              :
LogFile                     :
MaxConcurrency              :
MaxErrors                   :
Mode                        : Auto
Outputs                     : {}
ParentAutomationExecutionId :
ResolvedTargets             :
 Amazon.SimpleSystemsManagement.Model.ResolvedTargets
Target                      :
TargetMaps                  : {}
TargetParameterName         :
Targets                     : {}
```

运行需要审批的自动化

以下过程介绍了如何使用 AWS Systems Manager 控制台和 AWS Command Line Interface (AWS CLI) 通过简单执行来运行包含审批的自动化。该自动化使用自动化操作 `aws:approve`，这会暂停自动化，直到指定的委托人批准或拒绝该操作。自动化在当前用户的上下文中运行。因此，只要您有权使用运行手册和此运行手册调用的任何操作，就无需配置其他 IAM 权限。如果您在 IAM 中拥有管理员权限，则已有权运行此运行手册。

开始前的准备工作

除了运行手册所需的标准输入外，`aws:approve` 操作还需要以下两个参数：

- 审批者列表。审批者列表必须以用户名或用户 ARN 的形式包含至少一个审批者。如果提供了多个审批者，还必须在运行手册内指定对应的最少审批数。
- Amazon Simple Notification Service (Amazon SNS) 主题 ARN Amazon SNS 主题名称必须以 `Automation` 开头。

此过程假设您已经创建了 Amazon SNS 主题，这是提交审批请求所必需的。有关信息，请参阅《Amazon Simple Notification Service 开发人员指南》中的创建主题。

运行包含审批者的自动化（控制台）

运行包含审批者的自动化

以下过程介绍了如何使用 Systems Manager 控制台运行包含审批者的自动化。

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择自动化，然后选择执行自动化。
3. 在自动化文档列表中，请选择运行手册。在文档类别窗格中选择一个或多个选项，以便根据 SSM 文档的用途对其进行筛选。要查看您拥有的运行手册，请选择我拥有的选项卡。要查看与您的账户共享的运行手册，请选择与我共享选项卡。要查看所有运行手册，请选择所有文档选项卡。

   > **Note**
   >
   > 您可以通过选择运行手册名称来查看有关该手册的信息。

4. 在文档详细信息部分中，验证文档版本已设置为要运行的版本。系统包括以下版本选项：

- 运行时的默认版本 – 如果定期更新自动化运行手册并分配新的默认版本，请选择此选项。
- 运行时的最新版本 – 如果定期更新自动化运行手册并且想要运行最新更新的版本，请选择此选项。
- 1（默认） – 选择此选项可执行文档的第一个版本，即默认版本。

5. 选择下一步。
6. 在执行自动化文档页面上，选择简单执行。
7. 在输入参数 部分，指定所需的输入参数。

   例如，如果选择了 **AWS-StartEC2InstanceWithApproval** 运行手册，则必须为 InstanceId 参数指定或选择实例 ID。
8. 在审批者部分中，指定自动化操作的审批者的用户名或用户 ARN。
9. 在 SNSTopicARN 部分，指定要用于发送审批通知的 SNS 主题 ARN。SNS 主题名称必须以自动化开头。
10. 或者，您也可以从 AutomationAssumeRole 列表选择一个 IAM 服务角色。如果您将超过 100 个的账户和地区设置为目标，则必须指定 `AWS-SystemsManager-AutomationAdministrationRole`。
11. 选择执行自动化。

指定的审批者将收到包含详细信息的 Amazon SNS 通知以批准或拒绝自动化。此审批操作从发布之日起 7 天内有效，此操作可以使用 Systems Manager 控制台或 AWS Command Line Interface (AWS CLI) 发布。

如果选择批准自动化，自动化将继续运行指定的运行手册中包含的步骤。控制台将显示自动化的状态。如果自动化无法运行，请参阅 Systems Manager 自动化故障排除。

批准或拒绝自动化

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择自动化，然后选择上一过程中运行的自动化。
3. 选择操作，然后选择批准/拒绝。
4. 选择批准或拒绝，并选择性地提供注释。
5. 选择提交。

运行包含审批者的自动化（命令行）

以下过程介绍了如何使用 AWS CLI（在 Linux 或 Windows 上）或 AWS Tools for PowerShell 运行包含审批者的自动化。

运行包含审批者的自动化

1. 安装并配置 AWS CLI 或 AWS Tools for PowerShell（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本以及安装 AWS Tools for PowerShell。

2. 运行以下命令以运行包含审批者的自动化。将每个*#######*替换为您自己的信息。在文档名称部分中，指定包括自动化操作 `aws:approve` 的运行手册。

   对于 `Approvers`，指定该操作的审批者的用户名或用户 ARN。对于 `SNSTopic`，指定用于发送审批通知的 SNS 主题 ARN。Amazon SNS 主题名称必须以 `Automation` 开头。

   > **Note**
   >
   > 审批者和 SNS 主题的参数值的具体名称取决于您选择的运行手册中指定的值。

   Linux & macOS

   ```
   aws ssm start-automation-execution \
       --document-name "AWS-StartEC2InstanceWithApproval" \
       --parameters
    "InstanceId=i-02573cafcfEXAMPLE,Approvers=arn:aws:iam::123456789012:role/
   Administrator,SNSTopicArn=arn:aws:sns:region:123456789012:AutomationApproval"
   ```

   Windows

   ```
   aws ssm start-automation-execution ^
       --document-name "AWS-StartEC2InstanceWithApproval" ^
       --parameters
    "InstanceId=i-02573cafcfEXAMPLE,Approvers=arn:aws:iam::123456789012:role/
   Administrator,SNSTopicArn=arn:aws:sns:region:123456789012:AutomationApproval"
   ```

   PowerShell

   ```
   Start-SSMAutomationExecution `
   ```

```
    -DocumentName AWS-StartEC2InstanceWithApproval `
    -Parameters @{
        "InstanceId"="i-02573cafcfEXAMPLE"
        "Approvers"="arn:aws:iam::123456789012:role/Administrator"
        "SNSTopicArn"="arn:aws:sns:region:123456789012:AutomationApproval"
    }
```

系统将返回类似于以下内容的信息。

Linux & macOS

```
{
    "AutomationExecutionId": "df325c6d-b1b1-4aa0-8003-6cb7338213c6"
}
```

Windows

```
{
    "AutomationExecutionId": "df325c6d-b1b1-4aa0-8003-6cb7338213c6"
}
```

PowerShell

```
df325c6d-b1b1-4aa0-8003-6cb7338213c6
```

批准自动化

- 运行以下命令来批准自动化。将每个*#######*替换为您自己的信息。

  Linux & macOS

```
aws ssm send-automation-signal \
    --automation-execution-id "df325c6d-b1b1-4aa0-8003-6cb7338213c6" \
    --signal-type "Approve" \
    --payload "Comment=your comments"
```

  Windows

```
aws ssm send-automation-signal ^
```

```
    --automation-execution-id "df325c6d-b1b1-4aa0-8003-6cb7338213c6" ^
    --signal-type "Approve" ^
    --payload "Comment=your comments"
```

PowerShell

```
Send-SSMAutomationSignal `
    -AutomationExecutionId df325c6d-b1b1-4aa0-8003-6cb7338213c6 `
    -SignalType Approve `
    -Payload @{"Comment"="your comments"}
```

如果此命令成功，则无任何输出。

拒绝自动化

- 运行以下命令来拒绝自动化。将每个*#######*替换为您自己的信息。

  Linux & macOS

```
aws ssm send-automation-signal \
    --automation-execution-id "df325c6d-b1b1-4aa0-8003-6cb7338213c6" \
    --signal-type "Deny" \
    --payload "Comment=your comments"
```

  Windows

```
aws ssm send-automation-signal ^
    --automation-execution-id "df325c6d-b1b1-4aa0-8003-6cb7338213c6" ^
    --signal-type "Deny" ^
    --payload "Comment=your comments"
```

  PowerShell

```
Send-SSMAutomationSignal `
    -AutomationExecutionId df325c6d-b1b1-4aa0-8003-6cb7338213c6 `
    -SignalType Deny `
    -Payload @{"Comment"="your comments"}
```

如果此命令成功，则无任何输出。

大规模运行自动化操作

借助 AWS Systems Manager Automation，您可以使用目标在 AWS 资源的实例集上运行自动化。此外，您还可以通过指定并发值和错误阈值来控制自动化在整个队列上的部署。并发和错误阈值功能统称为速率控制。并发值决定允许同时运行自动化的资源数量。Automation 还提供了一种可以选择加入的自适应并发模式。自适应并发会自动将自动化配额从 100 个同时运行的自动化配额扩展到 500 个。错误阈值决定在 Systems Manager 停止将自动化发送到其他资源之前允许自动化执行失败的次数。

有关并发性和错误阈值的更多信息，请参阅 大规模控制自动化。有关目标的更多信息，请参阅 映射自动化目标。

以下程序介绍了如何开启自适应并发性，以及如何使用 Systems Manager 控制台和 AWS Command Line Interface (AWS CLI) 运行具有目标和速率控制的自动化。

运行具有目标和速率控制自动化（控制台）

以下过程介绍了如何使用 Systems Manager 控制台运行具有目标和速率控制的自动化。

运行具有目标和速率控制的自动化

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择自动化，然后选择执行自动化。
3. 在自动化文档列表中，请选择运行手册。在文档类别窗格中选择一个或多个选项，以便根据 SSM 文档的用途对其进行筛选。要查看您拥有的运行手册，请选择我拥有的选项卡。要查看与您的账户共享的运行手册，请选择与我共享选项卡。要查看所有运行手册，请选择所有文档选项卡。

   > **Note**
   >
   > 您可以通过选择运行手册名称来查看有关该手册的信息。

4. 在文档详细信息部分中，验证文档版本已设置为要运行的版本。系统包括以下版本选项：

   - 运行时的默认版本 – 如果定期更新自动化运行手册并分配新的默认版本，请选择此选项。
   - 运行时的最新版本 – 如果定期更新自动化运行手册并且想要运行最新更新的版本，请选择此选项。

- 1（默认）– 选择此选项可执行文档的第一个版本，即默认版本。

5. 选择下一步。
6. 在执行模式部分中，选择速率控制。如果要使用目标和速率控制，则必须使用此模式或多账户和多区域。
7. 在目标部分中，选择希望如何设置要在其中运行自动化的 AWS 资源。这些选项是必需的。

   a. 使用参数列表选择一个参数。参数列表中的项目由此过程开始时选择的自动化运行手册中的参数确定。通过选择参数，可以定义在其上运行自动化工作流的资源类型。

   b. 使用目标列表选择设置目标资源的方式。

      i. 如果选择使用参数值将资源设置为目标，请输入您在输入参数部分为参数选择的参数值。

      ii. 如果选择使用 AWS Resource Groups 将资源设置为目标，请从资源组列表中选择组的名称。

      iii. 如果选择使用标签将资源设置为目标，请在提供的字段中输入标签键和（可选）标签值。选择添加。

      iv. 如果要在当前 AWS 账户 和 AWS 区域 中的所有实例上运行自动化运行手册，则选择所有实例。

8. 在输入参数 部分中，指定所需的输入。或者，您也可以从 AutomationAssumeRole 列表中选择一个 IAM 服务角色。

   > **Note**
   >
   > 您可能不需要选择输入参数部分中的某些选项。原因是您使用标签或资源组将多个资源设置为目标。例如，如果选择了 `AWS-RestartEC2Instance` 运行手册，则无需在输入参数 部分中指定或选择实例 ID。自动化执行过程使用您指定的标签资源组查找要重新启动的实例。

9. 使用速率控制部分中的选项限制可在每个账户-区域对中运行自动化的 AWS 资源的数量。

   在并发部分中，选择一个选项：

   - 选择目标，以输入可同时运行自动化工作流目标的绝对数量。
   - 选择百分比，以输入可同时运行自动化工作流的目标集的百分比。

10. 在错误阈值部分中，选择一个选项：

    - 选择错误，以输入自动化停止将工作流程发送到其他资源之前允许的错误的绝对数量。

- 选择百分比，以输入自动化停止将工作流程发送到其他资源之前允许的错误的百分比。

11. （可选）选择一个 CloudWatch 警报以应用于您的自动化进行监控。要将 CloudWatch 警报附加到自动化，启动自动化的 IAM 主体必须具有 `iam:createServiceLinkedRole` 操作的权限。有关 CloudWatch 警报的更多信息，请参阅使用 Amazon CloudWatch 警报。请注意，如果您的警报激活，自动化将停止。如果使用 AWS CloudTrail，您将在跟踪中看到 API 调用。
12. 选择执行。

要查看由速率控制自动化启动的自动化程序，请在导航窗格中，选择自动化，然后选择显示子自动化程序。

运行具有目标和速率控制的自动化（命令行）

以下过程介绍了如何使用 AWS CLI（在 Linux 或 Windows 上）或 AWS Tools for PowerShell 运行具有目标和速率控制的自动化。

运行具有目标和速率控制的自动化

1. 安装并配置 AWS CLI 或 AWS Tools for PowerShell（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本以及安装 AWS Tools for PowerShell。
2. 运行以下命令以查看文档列表。

   Linux & macOS

   ```
   aws ssm list-documents
   ```

   Windows

   ```
   aws ssm list-documents
   ```

   PowerShell

   ```
   Get-SSMDocumentList
   ```

   记下要使用的运行手册的名称。

3. 运行以下命令以查看有关运行手册的详细信息。用您想要查看其详细信息的运行手册的名称替换 *runbook name*。同时，记下要用于 `--target-parameter-name` 选项的参数名称（例如 `InstanceId`）。此参数确定在其上运行自动化的资源的类型。

   Linux & macOS

   ```
   aws ssm describe-document \
       --name runbook name
   ```

   Windows

   ```
   aws ssm describe-document ^
       --name runbook name
   ```

   PowerShell

   ```
   Get-SSMDocumentDescription `
       -Name runbook name
   ```

4. 创建一个要运行的使用目标和速率控制选项的命令。将每个*#######*替换为您自己的信息。

   使用标签设置目标

   Linux & macOS

   ```
   aws ssm start-automation-execution \
       --document-name runbook name \
       --targets Key=tag:key name,Values=value \
       --target-parameter-name parameter name \
       --parameters "input parameter name=input parameter value,input parameter 2
    name=input parameter 2 value" \
       --max-concurrency 10 \
       --max-errors 25%
   ```

   Windows

   ```
   aws ssm start-automation-execution ^
       --document-name runbook name ^
       --targets Key=tag:key name,Values=value ^
       --target-parameter-name parameter name ^
   ```

```
    --parameters "input parameter name=input parameter value,input parameter 2
 name=input parameter 2 value" ^
    --max-concurrency 10 ^
    --max-errors 25%
```

PowerShell

```
$Targets = New-Object Amazon.SimpleSystemsManagement.Model.Target
$Targets.Key = "tag:key name"
$Targets.Values = "value"

Start-SSMAutomationExecution `
    DocumentName "runbook name" `
    -Targets $Targets `
    -TargetParameterName "parameter name" `
    -Parameter @{"input parameter name"="input parameter value";"input parameter
 2 name"="input parameter 2 value"} `
    -MaxConcurrency "10" `
    -MaxError "25%"
```

使用参数值设置目标

Linux & macOS

```
aws ssm start-automation-execution \
    --document-name runbook name \
    --targets Key=ParameterValues,Values=value,value 2,value 3 \
    --target-parameter-name parameter name \
    --parameters "input parameter name=input parameter value" \
    --max-concurrency 10 \
    --max-errors 25%
```

Windows

```
aws ssm start-automation-execution ^
    --document-name runbook name ^
    --targets Key=ParameterValues,Values=value,value 2,value 3 ^
    --target-parameter-name parameter name ^
    --parameters "input parameter name=input parameter value" ^
    --max-concurrency 10 ^
```

```
    --max-errors 25%
```

PowerShell

```
$Targets = New-Object Amazon.SimpleSystemsManagement.Model.Target
$Targets.Key = "ParameterValues"
$Targets.Values = "value","value 2","value 3"

Start-SSMAutomationExecution `
    -DocumentName "runbook name" `
    -Targets $Targets `
    -TargetParameterName "parameter name" `
    -Parameter @{"input parameter name"="input parameter value"} `
    -MaxConcurrency "10" `
    -MaxError "25%"
```

使用 AWS Resource Groups 设置目标

Linux & macOS

```
aws ssm start-automation-execution \
    --document-name runbook name \
    --targets Key=ResourceGroup,Values=Resource group nname \
    --target-parameter-name parameter name \
    --parameters "input parameter name=input parameter value" \
    --max-concurrency 10 \
    --max-errors 25%
```

Windows

```
aws ssm start-automation-execution ^
    --document-name runbook name ^
    --targets Key=ResourceGroup,Values=Resource group name ^
    --target-parameter-name parameter name ^
    --parameters "input parameter name=input parameter value" ^
    --max-concurrency 10 ^
    --max-errors 25%
```

PowerShell

```
$Targets = New-Object Amazon.SimpleSystemsManagement.Model.Target
$Targets.Key = "ResourceGroup"
$Targets.Values = "Resource group name"

Start-SSMAutomationExecution `
    -DocumentName "runbook name" `
    -Targets $Targets `
    -TargetParameterName "parameter name" `
    -Parameter @{"input parameter name"="input parameter value"} `
    -MaxConcurrency "10" `
    -MaxError "25%"
```

将当前 AWS 账户 和 AWS 区域 中的所有 Amazon EC2 实例设置为目标

Linux & macOS

```
aws ssm start-automation-execution \
    --document-name runbook name \
    --targets "Key=AWS::EC2::Instance,Values=*"  \
    --target-parameter-name instanceId \
    --parameters "input parameter name=input parameter value" \
    --max-concurrency 10 \
    --max-errors 25%
```

Windows

```
aws ssm start-automation-execution ^
    --document-name runbook name ^
    --targets Key=AWS::EC2::Instance,Values=* ^
    --target-parameter-name instanceId ^
    --parameters "input parameter name=input parameter value" ^
    --max-concurrency 10 ^
    --max-errors 25%
```

PowerShell

```
$Targets = New-Object Amazon.SimpleSystemsManagement.Model.Target
$Targets.Key = "AWS::EC2::Instance"
```

```
$Targets.Values = "*"

Start-SSMAutomationExecution `
    -DocumentName "runbook name" `
    -Targets $Targets `
    -TargetParameterName "instanceId" `
    -Parameter @{"input parameter name"="input parameter value"} `
    -MaxConcurrency "10" `
    -MaxError "25%"
```

该命令将会返回执行 ID。请将该 ID 复制到剪贴板。您可以使用此 ID 查看自动化的状态。

Linux & macOS

```
{
    "AutomationExecutionId": "a4a3c0e9-7efd-462a-8594-01234EXAMPLE"
}
```

Windows

```
{
    "AutomationExecutionId": "a4a3c0e9-7efd-462a-8594-01234EXAMPLE"
}
```

PowerShell

```
a4a3c0e9-7efd-462a-8594-01234EXAMPLE
```

5. 运行以下命令以查看自动化。将每个 *automation execution ID* 替换为您自己的信息。

Linux & macOS

```
aws ssm describe-automation-executions \
    --filter Key=ExecutionId,Values=automation execution ID
```

Windows

```
aws ssm describe-automation-executions ^
    --filter Key=ExecutionId,Values=automation execution ID
```

PowerShell

```
Get-SSMAutomationExecutionList | `
    Where {$_.AutomationExecutionId -eq "automation execution ID"}
```

6. 运行以下命令以查看有关自动化进展的详细信息。将每个 *automation execution ID* 替换为您自己的信息。

Linux & macOS

```
aws ssm get-automation-execution \
    --automation-execution-id automation execution ID
```

Windows

```
aws ssm get-automation-execution ^
    --automation-execution-id automation execution ID
```

PowerShell

```
Get-SSMAutomationExecution `
    -AutomationExecutionId automation execution ID
```

系统将返回类似于以下内容的信息。

Linux & macOS

```
{
    "AutomationExecution": {
        "StepExecutionsTruncated": false,
        "AutomationExecutionStatus": "Success",
        "MaxConcurrency": "1",
        "Parameters": {},
        "MaxErrors": "1",
        "Outputs": {},
        "DocumentName": "AWS-StopEC2Instance",
        "AutomationExecutionId": "a4a3c0e9-7efd-462a-8594-01234EXAMPLE",
        "ResolvedTargets": {
            "ParameterValues": [
                "i-02573cafcfEXAMPLE"
```

```
            ],
            "Truncated": false
        },
        "ExecutionEndTime": 1564681619.915,
        "Targets": [
            {
                "Values": [
                    "DEV"
                ],
                "Key": "tag:ENV"
            }
        ],
        "DocumentVersion": "1",
        "ExecutionStartTime": 1564681576.09,
        "ExecutedBy": "arn:aws:sts::123456789012:assumed-role/Administrator/
Admin",
        "StepExecutions": [
            {
                "Inputs": {
                    "InstanceId": "i-02573cafcfEXAMPLE"
                },
                "Outputs": {},
                "StepName": "i-02573cafcfEXAMPLE",
                "ExecutionEndTime": 1564681619.093,
                "StepExecutionId": "86c7b811-3896-4b78-b897-01234EXAMPLE",
                "ExecutionStartTime": 1564681576.836,
                "Action": "aws:executeAutomation",
                "StepStatus": "Success"
            }
        ],
        "TargetParameterName": "InstanceId",
        "Mode": "Auto"
    }
}
```

Windows

```
{
    "AutomationExecution": {
        "StepExecutionsTruncated": false,
        "AutomationExecutionStatus": "Success",
        "MaxConcurrency": "1",
        "Parameters": {},
```

```
        "MaxErrors": "1",
        "Outputs": {},
        "DocumentName": "AWS-StopEC2Instance",
        "AutomationExecutionId": "a4a3c0e9-7efd-462a-8594-01234EXAMPLE",
        "ResolvedTargets": {
            "ParameterValues": [
                "i-02573cafcfEXAMPLE"
            ],
            "Truncated": false
        },
        "ExecutionEndTime": 1564681619.915,
        "Targets": [
            {
                "Values": [
                    "DEV"
                ],
                "Key": "tag:ENV"
            }
        ],
        "DocumentVersion": "1",
        "ExecutionStartTime": 1564681576.09,
        "ExecutedBy": "arn:aws:sts::123456789012:assumed-role/Administrator/
Admin",
        "StepExecutions": [
            {
                "Inputs": {
                    "InstanceId": "i-02573cafcfEXAMPLE"
                },
                "Outputs": {},
                "StepName": "i-02573cafcfEXAMPLE",
                "ExecutionEndTime": 1564681619.093,
                "StepExecutionId": "86c7b811-3896-4b78-b897-01234EXAMPLE",
                "ExecutionStartTime": 1564681576.836,
                "Action": "aws:executeAutomation",
                "StepStatus": "Success"
            }
        ],
        "TargetParameterName": "InstanceId",
        "Mode": "Auto"
    }
}
```

PowerShell

```
AutomationExecutionId       : a4a3c0e9-7efd-462a-8594-01234EXAMPLE
AutomationExecutionStatus   : Success
CurrentAction               :
CurrentStepName             :
DocumentName                : AWS-StopEC2Instance
DocumentVersion             : 1
ExecutedBy                  : arn:aws:sts::123456789012:assumed-role/
Administrator/Admin
ExecutionEndTime            : 8/1/2019 5:46:59 PM
ExecutionStartTime          : 8/1/2019 5:46:16 PM
FailureMessage              :
MaxConcurrency              : 1
MaxErrors                   : 1
Mode                        : Auto
Outputs                     : {}
Parameters                  : {}
ParentAutomationExecutionId :
ProgressCounters            :
ResolvedTargets             :
 Amazon.SimpleSystemsManagement.Model.ResolvedTargets
StepExecutions              : {i-02573cafcfEXAMPLE}
StepExecutionsTruncated     : False
Target                      :
TargetLocations             : {}
TargetMaps                  : {}
TargetParameterName         : InstanceId
Targets                     : {tag:Name}
```

> **Note**
>
> 您也可以在控制台中监控自动化的状态。在自动化执行列表中，请选择您刚才运行的自动化，然后选择执行步骤选项卡。该选项卡显示自动化操作的状态。

映射自动化目标

通过 Targets 参数，可以快速定义自动化的目标资源。例如，如果需要运行一个重新启动托管实例的自动化，不必在控制台中手动选择数十个实例 ID 或在命令中键入它们，您可以通过使用 Targets 参数指定 Amazon Elastic Compute Cloud (Amazon EC2) 标签将实例设为目标。

运行使用目标的自动化时，AWS Systems Manager 会为每个目标创建一个子自动化。例如，如果您通过指定标签将 Amazon Elastic Block Store (Amazon EBS) 卷指定为目标，并且这些标签解析为 100 个 Amazon EBS 卷，则 Systems Manager 创建 100 个子自动化。当所有子自动化都到达最终状态时，父自动化完成。

> **Note**
>
> 在运行时指定的任何 input parameters（在控制台的输入参数部分或在命令行中使用 parameters 选项）都由所有子自动化自动处理。

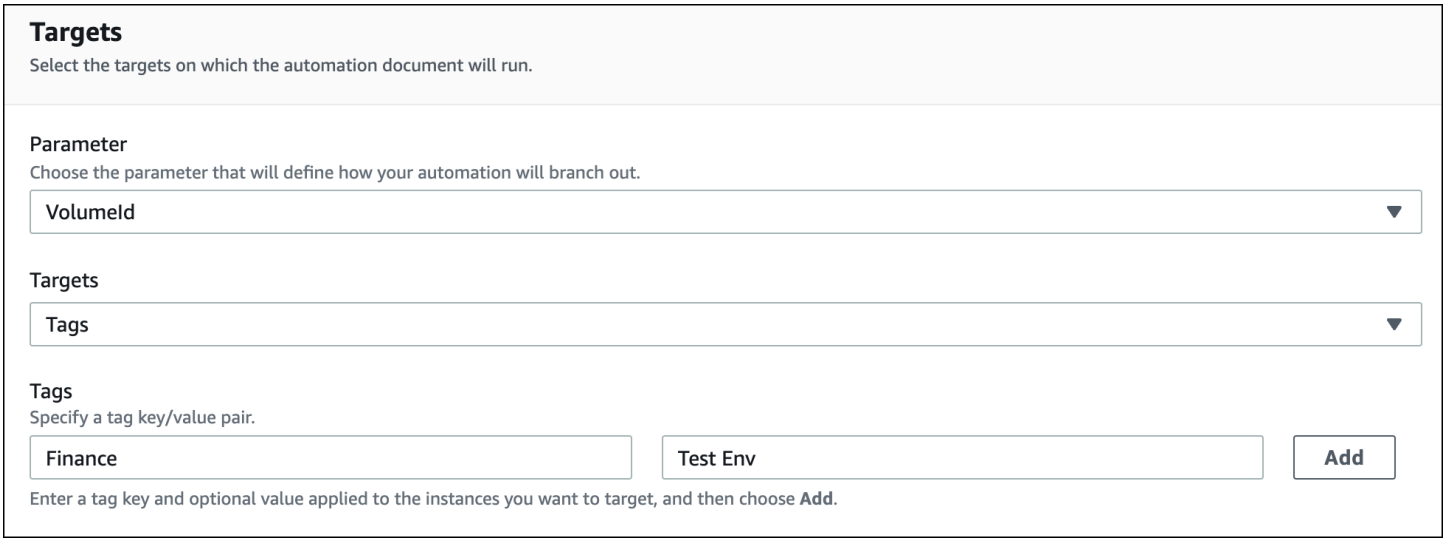
您可以使用标签、Resource Groups 和参数值为自动化设置目标资源。此外，您还可以使用 TargetMaps 选项在命令行或文件中将多个参数值设为目标。下一节更详细地介绍每个设置目标选项。

设置目标标签

您可以指定单个标签作为自动化目标。多个 AWS 资源支持标签，包括 Amazon Elastic Compute Cloud (Amazon EC2) 和 Amazon Relational Database Service (Amazon RDS) 实例、Amazon Elastic Block Store (Amazon EBS) 卷和快照、Resource Groups 以及 Amazon Simple Storage Service (Amazon S3) 存储桶（仅举几例）。您可以通过设置目标标签在 AWS 资源上快速运行自动化。标签是一种键值对，例如 Operating_System:Linux 或 Department:Finance。如果为资源指定一个特定名称，则还可以使用“Name”作为键，将资源名称用作值。

在将标签指定为自动化目标时，还可以指定目标参数。目标参数使用 TargetParameterName 选项。通过选择目标参数，可以定义在其上运行自动化的资源的类型。使用标签指定的目标参数必须是运行手册中定义的有效参数。例如，如果要使用标签将数十个 EC2 实例设为目标，请选择 InstanceId 目标参数。通过选择此参数，可以将实例 定义为自动化的资源类型。在创建自定义运行手册时，您必须将目标类型指定为 /AWS::EC2::Instance，以确保仅使用实例。否则，所有具有相同标签的资源都将被设置为目标。使用标签定位实例时，可能会包括已终止的实例。

下面的屏幕截图使用 AWS-DetachEBSVolume 运行手册。逻辑目标参数为 VolumeId。



AWS-DetachEBSVolume 运行手册还包含一个名为目标类型的特殊属性，设置为 / AWS::EC2::Volume。这意味着，如果标签键对 Finance:TestEnv 返回不同类型的资源（例如，EC2 实例、Amazon EBS 卷、Amazon EBS 快照），则仅使用 Amazon EBS 卷。

Important

目标参数名称区分大小写。如果使用 AWS Command Line Interface (AWS CLI) 或 AWS Tools for Windows PowerShell 运行自动化，则必须完全按照运行手册中的定义输入目标参数名称。否则，系统将返回 InvalidAutomationExecutionParametersException 错误。您可以使用 DescribeDocument API 操作来查看有关特定运行手册中可用目标参数的信息。下面是一个示例 AWS CLI 命令，其中提供有关 AWS-DeleteSnapshot 文档的信息。

```
aws ssm describe-document \
    --name AWS-DeleteSnapshot
```

下面是一些使用标签设置资源目标的示例 AWS CLI 命令。

示例 1：使用键值对将标签指定为目标以重启 Amazon EC2 实例

此示例重新启动带有键为 Department、值为 HumanResources 的标签的所有 Amazon EC2 实例。目标参数使用运行手册中的 InstanceId 参数。该示例使用了一个附加参数，通过使用自动化服务角色（也称为担任角色）来运行自动化。

```
aws ssm start-automation-execution \
```

```
    --document-name AWS-RestartEC2Instance \
    --targets Key=tag:Department,Values=HumanResources \
    --target-parameter-name InstanceId \
    --parameters "AutomationAssumeRole=arn:aws:iam::111122223333:role/
AutomationServiceRole"
```

示例 2：使用键值对将标签指定为目标以删除 Amazon EBS 快照

下面的示例使用 AWS-DeleteSnapshot 运行手册删除键为 Name、值为 January2018Backups 的所有快照。目标参数使用 VolumeId 参数。

```
aws ssm start-automation-execution \
    --document-name AWS-DeleteSnapshot \
    --targets Key=tag:Name,Values=January2018Backups \
    --target-parameter-name VolumeId
```

设置目标 AWS Resource Groups

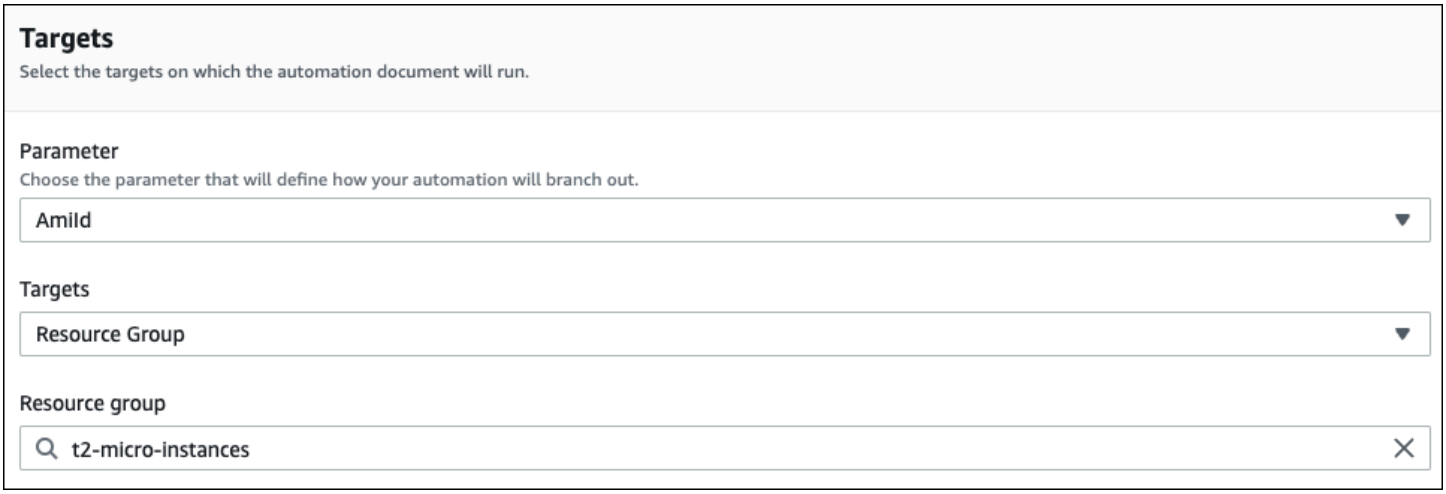
您可以指定单个 AWS 资源组作为自动化目标。Systems Manager 为目标资源组中的每个对象创建一个子自动化。

例如，假设您的一个 Resource Groups 名为 PatchedAMIs。此资源组包括一个列表，其中有 25 个定期进行修补的 Windows Amazon Machine Images (AMIs)。如果您运行使用 AWS-CreateManagedWindowsInstance 运行手册并以此资源组为目标的自动化，则 Systems Manager 会为 25 个 AMIs 中的每一个创建一个子自动化。也就是说，通过将 PatchedAMIs 资源组设置为目标，自动化会从修补的 AMIs 列表中创建 25 个实例。当所有子自动化都完成处理或到达最终状态时，父自动化完成。

下面的 AWS CLI 命令适用于此 PatchAMIs 资源组示例。该命令中，--target-parameter-name 选项采用 AmiId 参数。该命令不包含定义从每个 AMI 创建的实例类型的附加参数。AWS-CreateManagedWindowsInstance 运行手册默认为 t2.medium 实例类型，因此该命令将创建 25 个适用于 Windows Server 的 t2.medium Amazon EC2 实例。

```
aws ssm start-automation-execution \
    --document-name AWS-CreateManagedWindowsInstance \
    --targets Key=ResourceGroup,Values=PatchedAMIs  \
    --target-parameter-name AmiId
```

下面的控制台示例使用名为 t2-micro-instances 的资源组。



设置目标参数值

您也可以将参数值设置为目标：输入 `ParameterValues` 作为键，然后输入要在其上运行自动化的特定资源值。如果指定多个值，Systems Manager 将在指定的每个值上运行子自动化。

例如，假设运行手册包含 InstanceID 参数。如果在运行此自动化时将 InstanceID 参数值设置为目标，则 Systems Manager 会为指定的每个实例 ID 值运行一个子自动化。当自动化完成每个指定实例的运行或执行失败时，父自动化完成。您最多可以将 50 个参数值设置为目标。

以下示例使用 `AWS-CreateImage` 运行手册。指定的目标参数名称为 InstanceId。键使用 ParameterValues。值为两个 Amazon EC2 实例 ID。此命令为每个实例创建一个自动化，这将从每个实例生成一个 AMI。

```
aws ssm start-automation-execution
    --document-name AWS-CreateImage \
    --target-parameter-name InstanceId \
    --targets Key=ParameterValues,Values=i-02573cafcfEXAMPLE,i-0471e04240EXAMPLE
```

> **Note**
>
> `AutomationAssumeRole` 不是有效的参数。运行以参数值为目标的自动化时，不要选择此项。

设置目标参数值映射

TargetMaps 选项扩展了您设置 ParameterValues 目标的能力。您可以在命令行中使用 TargetMaps 输入一个参数值数组。在命令行中，最多可以指定 50 个参数值。如果要运行指定 50 个以上参数值的命令，请在 JSON 文件中输入这些值。然后，可以在命令行中调用该文件。

> **Note**
>
> 控制台中不支持 TargetMaps 选项。

在命令中使用 TargetMaps 选项以下面的格式指定多个参数值。将每个*#######*替换为您自己的信息。

```
aws ssm start-automation-execution \
    --document-name runbook name \
    --target-maps “parameter=value, parameter 2=value, parameter 3=value”  “parameter
 4=value, parameter 5=value, parameter 6=value”
```

如果要为 TargetMaps 选项输入 50 个以上的参数值，请使用以下 JSON 格式在文件中指定这些值。在提供多个参数值时，使用 JSON 文件还有助于提高可读性。

```
[

    {“parameter”: "value", “parameter 2”: "value", “parameter 3”: "value"},

    {“parameter 4”: "value", “parameter 5”: "value", "parameter 6": "value"}

]
```

使用 .json 文件扩展名保存该文件。可以使用下面的命令调用此文件。将每个*#######*替换为您自己的信息。

```
aws ssm start-automation-execution \
    --document-name runbook name \
    --parameters input parameters \
    --target-maps path to file/file name.json
```

您还可以从 Amazon Simple Storage Service (Amazon S3) 存储桶下载此文件，前提是您有权从存储桶读取数据。使用下面的命令格式。将每个*#######*替换为您自己的信息。

```
aws ssm start-automation-execution \
    --document-name runbook name \
    --target-maps http://amzn-s3-demo-bucket.s3.amazonaws.com/file_name.json
```

下面是一个帮助您了解 `TargetMaps` 选项的示例方案。在此方案中，用户想要从不同的 AMIs 创建不同类型的 Amazon EC2 实例。为了执行此任务，用户创建一个名为 AMI_Testing 的运行手册。此运行手册定义两个输入参数：`instanceType` 和 `imageId`。

```
{
  "description": "AMI Testing",
  "schemaVersion": "0.3",
  "assumeRole": "{{assumeRole}}",
  "parameters": {
    "assumeRole": {
      "type": "String",
      "description": "Role under which to run the automation",
      "default": ""
    },
    "instanceType": {
      "type": "String",
      "description": "Type of EC2 Instance to launch for this test"
    },
    "imageId": {
      "type": "String",
      "description": "Source AMI id from which to run instance"
    }
  },
  "mainSteps": [
    {
      "name": "runInstances",
      "action": "aws:runInstances",
      "maxAttempts": 1,
      "onFailure": "Abort",
      "inputs": {
        "ImageId": "{{imageId}}",
        "InstanceType": "{{instanceType}}",
        "MinInstanceCount": 1,
        "MaxInstanceCount": 1
      }
```

```
    }
  ],
  "outputs": [
    "runInstances.InstanceIds"
  ]
}
```

然后，用户在名为 AMI_instance_types.json 的文件中指定以下目标参数值。

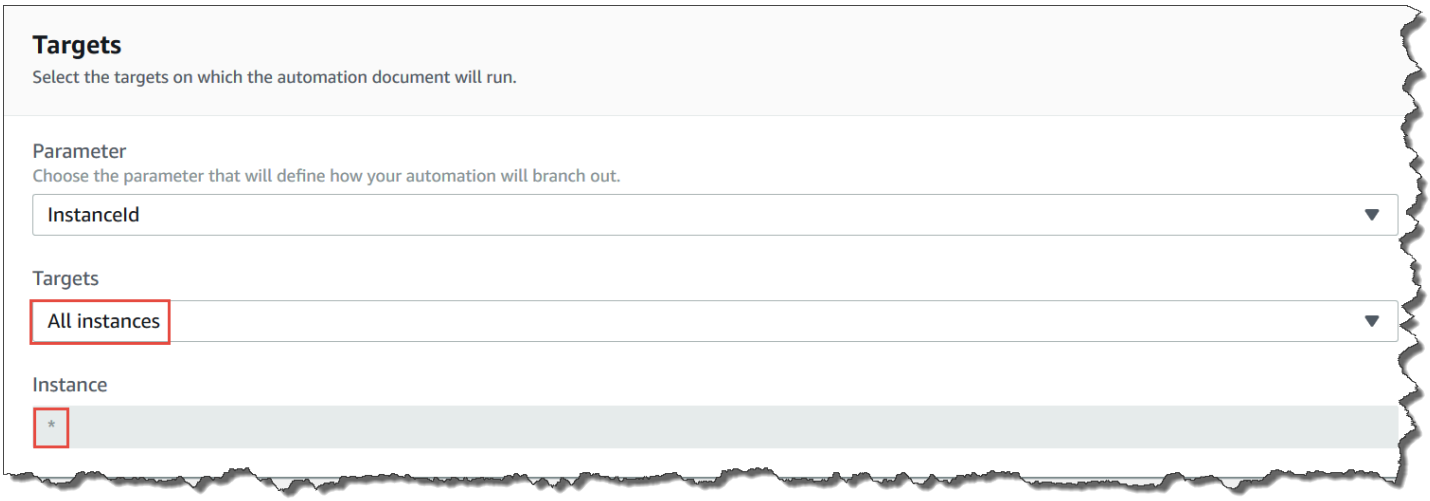
```
[
  {
    "instanceType" : ["t2.micro"],
    "imageId" : ["ami-b70554c8"]
  },
  {
    "instanceType" : ["t2.small"],
    "imageId" : ["ami-b70554c8"]
  },
  {
    "instanceType" : ["t2.medium"],
    "imageId" : ["ami-cfe4b2b0"]
  },
  {
    "instanceType" : ["t2.medium"],
    "imageId" : ["ami-cfe4b2b0"]
  },
  {
    "instanceType" : ["t2.medium"],
    "imageId" : ["ami-cfe4b2b0"]
  }
]
```

用户可以通过运行以下命令来运行此自动化并创建在 AMI_instance_types.json 中定义的五个 EC2 实例：

```
aws ssm start-automation-execution \
    --document-name AMI_Testing \
    --target-parameter-name imageId \
    --target-maps file:///home/TestUser/workspace/runinstances/AMI_instance_types.json
```

将所有 Amazon EC2 实例设置为目标

您可以通过选择目标列表中的所有实例，在当前 AWS 账户 和 AWS 区域 中的所有 Amazon EC2 实例上运行自动化。例如，如果要重新启动 AWS 账户 和当前 AWS 区域 的所有 Amazon EC2 实例，您可以选择 **`AWS-RestartEC2Instance`** 运行手册，然后选择目标列表中的所有实例。



在您选择所有实例后，Systems Manager 将填充带有星号 (*) 的实例字段，并使字段不可更改（该字段呈灰色）。Systems Manager 还会使 InstanceId 字段中的输入参数字段不可更改。如果您选择将所有实例设置为目标，则使这些字段不可更改是预期行为。

大规模控制自动化

您可以通过指定并发值和错误阈值来控制自动化在 AWS 资源队列上的部署。并发和错误阈值统称为速率控制。

并发

通过并发，您可以指定允许同时运行自动化的资源数量。并发有助于在处理自动化时限制对资源的影响或停机时间。您可以指定绝对数量的资源（如 20），也可以指定目标集百分比（如 10%）。

队列系统将自动化传输到单个资源，等到初始调用完成之后，再将自动化发送到其他两个资源。系统以指数增长方式将自动化发送到更多资源，直到达到并发值。

错误阈值

通过错误阈值，您可以指定在 AWS Systems Manager 停止将自动化发送到其他资源之前允许自动化失败的次数。您可以指定绝对数量的错误（如 10），也可以指定目标集百分比（如 10%）。

例如，如果指定错误的绝对数为 3，则系统将在收到第四个错误时停止发送自动化。如果指定 0，则系统会在返回第一个错误结果后停止向其他资源发送自动化。

例如，如果向 50 个实例发送自动化并将错误阈值设置为 10%，则系统会在收到第五个错误时停止向其他实例发送命令。当达到错误阈值时，允许完成已经运行自动化的调用，但是其中一些自动化也可能失败。如果需要确保错误数不超过为错误阈值指定的数量，请将并发值设置为 1，以便一次只处理一个自动化。

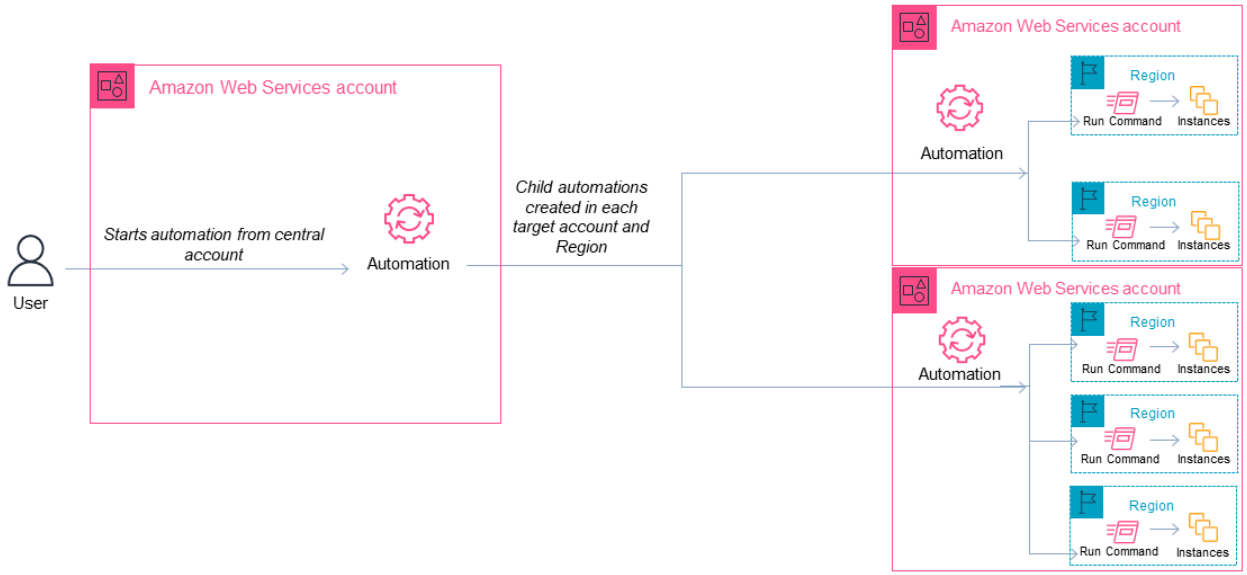
在多个 AWS 区域 和账户中运行自动化

您可以在中央账户的多个 AWS 区域 区域和 AWS 账户 账户或 AWS Organizations 组织部门 (OU) 中运行 AWS Systems Manager 自动化。Automation 是 AWS Systems Manager 中的一项工具。在多个区域和账户或 OU 中运行自动化可减少管理 AWS 资源所需的时间，同时提高计算环境的安全性。

例如，您可以使用自动化运行手册执行以下操作：

- 集中实施补丁和安全更新。
- 修复 VPC 配置或 Amazon S3 桶策略的合规性偏差。
- 大规模管理资源，例如 Amazon Elastic Compute Cloud (Amazon EC2) EC2 实例。

下图显示了一个用户在中央账户的多个区域和账户中运行 `AWS-RestartEC2Instances` 运行手册的示例。自动化在目标区域和账户中使用指定的标签以查找实例。



为自动化选择中央账户

如果要在 OU 中运行自动化，中央账户必须具有列出 OU 中所有账户的权限。这只能通过委派管理员账户或组织的管理账户实现。我们建议您遵循 AWS Organizations 最佳实践并使用委派管理员账户。有关 AWS Organizations 最佳实践的更多信息，请参阅 AWS Organizations 用户指南中的管理账户的最佳实践。要为 Systems Manager 创建委派管理员账户，可以将 `register-delegated-administrator` 命令与 AWS CLI 一起使用，如以下示例所示。

```
aws organizations register-delegated-administrator \
    --account-id delegated admin account ID \
    --service-principal ssm.amazonaws.com
```

如果您想跨多个不受 AWS Organizations 托管的账户运行自动化，我们建议您创建一个专用账户用于自动化管理。从专用账户运行所有跨账户自动化可简化 IAM 权限管理、故障排除工作，并在操作和管理之间建立分离层。如果您使用 AWS Organizations，但只希望针对个人账户而不是 OU，也建议使用此方法。

自动化的工作原理

跨多个区域和账户或 OU 运行自动化的工作方式如下：

1. 登录要配置为 Automation 中央账户的账户。
2. 使用本主题中的 设置进行多区域和多账户自动化的管理账户权限 过程创建以下 IAM 角色：
   - **AWS-SystemsManager-AutomationAdministrationRole** - 此角色为用户提供在多个账户和 OU 中运行自动化的权限。
   - **AWS-SystemsManager-AutomationExecutionRole** - 此角色为用户提供在目标账户中运行自动化的权限。
3. 选择要运行自动化的运行手册、区域、账户或 OU。

> **Note**
>
> 确保目标 OU 包含所需的账户。如果选择自定义运行手册，则必须与所有目标账户共享运行手册。有关共享运行手册的信息，请参阅 共享 SSM 文档。有关使用共享运行手册的信息，请参阅 使用共享 SSM 文档。

4. 运行自动化。

> **Note**
>
> 在多个区域、账户或 OU 之间运行自动化时，从主账户运行的自动化将在每个目标账户中启动子自动化。主账户中的自动化包含每个目标账户的 `aws:executeAutomation` 步骤。

5. 通过 AWS Systems Manager 控制台或 AWS CLI 使用 GetAutomationExecution、DescribeAutomationStepExecutions 和 DescribeAutomationExecutions API 操作监控自动化进度。您的主账户中自动化步骤的输出将是子自动化的 `AutomationExecutionId`。要查看在目标账户中创建的子自动化的输出，请确保在您的请求中指定相应的账户、区域和 `AutomationExecutionId`。

设置进行多区域和多账户自动化的管理账户权限

按照以下过程操作，使用 AWS CloudFormation 创建进行 Systems Manager 自动化多区域和多账户自动化所需的 IAM 角色。此过程介绍如何创建 **AWS-SystemsManager-AutomationAdministrationRole** 角色。您只需要在自动化中央账户中创建此角色。此过程还介绍了如何创建 **AWS-SystemsManager-AutomationExecutionRole** 角色。您必须在要设置为多区域和多账户自动化运行目标的每一个账户中创建该角色。我们建议使用 AWS CloudFormation StackSets 您想要设置为多区域和多账户自动化运行目标的账户创建 **AWS-SystemsManager-AutomationExecutionRole** 角色。

使用 AWS CloudFormation 创建进行多区域和多账户自动化所需的 IAM 管理角色

1. 下载并解压缩 AWS-SystemsManager-AutomationAdministrationRole.zip。或者，如果您的账户由 AWS Organizations AWS-SystemsManager-AutomationAdministrationRole (org).zip 管理。此文件包含 `AWS-SystemsManager-AutomationAdministrationRole.yaml` AWS CloudFormation 模板文件。
2. 通过以下网址打开 AWS CloudFormation 控制台：https://console.aws.amazon.com/cloudformation。
3. 选择创建堆栈。
4. 在 Specify template（指定模板）部分中，选择 Upload a template（上传模板）。
5. 选择 Choose file（选择文件），然后选择 `AWS-SystemsManager-AutomationAdministrationRole.yaml` AWS CloudFormation 模板文件。
6. 选择 Next（下一步）。
7. 在指定堆栈详细信息页面的堆栈名称字段中，输入名称。

8. 选择 Next（下一步）。
9. 在 Configure stack options（配置堆栈选项）页面上，为要使用的所有选项输入值。选择 Next（下一步）。
10. 在审核页面上，向下滚动并选择我确认 AWS CloudFormation 可能使用自定义名称创建了 IAM 资源选项。
11. 选择创建堆栈。

大约三分钟左右，AWS CloudFormation 会显示 CREATE_IN_PROGRESS 状态。状态变为 CREATE_COMPLETE。

在要设置为多区域和多账户自动化运行目标的每一个账户中，您必须重复以下过程。

使用 AWS CloudFormation 创建进行多区域和多账户自动化所需的 IAM 自动化角色

1. 下载 AWS-SystemsManager-AutomationExecutionRole.zip。或者，如果您的账户由 AWS Organizations AWS-SystemsManager-AutomationExecutionRole (org).zip 管理。此文件包含 AWS-SystemsManager-AutomationExecutionRole.yaml AWS CloudFormation 模板文件。
2. 通过以下网址打开 AWS CloudFormation 控制台：https://console.aws.amazon.com/cloudformation。
3. 选择创建堆栈。
4. 在 Specify template（指定模板）部分中，选择 Upload a template（上传模板）。
5. 选择 Choose file（选择文件），然后选择 AWS-SystemsManager-AutomationExecutionRole.yaml AWS CloudFormation 模板文件。
6. 选择 Next（下一步）。
7. 在指定堆栈详细信息页面的堆栈名称字段中，输入名称。
8. 在 Parameters（参数）部分的 AdminAccountId 字段中，输入自动化中央账户的 ID。
9. 如果您要为 AWS Organizations 环境设置此角色，则该部分中还有另一个名为 OrganizationID 的字段。输入 AWS 组织的 ID。
10. 选择 Next（下一步）。
11. 在 Configure stack options（配置堆栈选项）页面上，为要使用的所有选项输入值。选择 Next（下一步）。
12. 在审核页面上，向下滚动并选择我确认 AWS CloudFormation 可能使用自定义名称创建了 IAM 资源选项。

13. 选择创建堆栈。

大约三分钟左右，AWS CloudFormation 会显示 CREATE_IN_PROGRESS 状态。状态变为 CREATE_COMPLETE。

在多个区域和账户中运行自动化（控制台）

以下过程介绍了如何使用 Systems Manager 控制台在自动化管理账户的多个区域和账户中运行自动化。

开始前的准备工作

在完成以下过程之前，请记下以下信息：

- 用于运行多区域或多账户自动化的用户或角色必须拥有 `AWS-SystemsManager-AutomationAdministrationRole` 角色的 `iam:PassRole` 权限。
- 要在其中运行自动化的 AWS 账户 账户 ID 或 OU。
- 将在其中运行自动化的受 Systems Manager 支持区域。
- 要在其中运行自动化的标签键和标签值或资源组的名称。

在多个区域和账户中运行自动化

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择自动化，然后选择执行自动化。
3. 在自动化文档列表中，请选择运行手册。在文档类别窗格中选择一个或多个选项，以便根据 SSM 文档的用途对其进行筛选。要查看您拥有的运行手册，请选择我拥有的选项卡。要查看与您的账户共享的运行手册，请选择与我共享选项卡。要查看所有运行手册，请选择所有文档选项卡。

   > **Note**
   >
   > 您可以通过选择运行手册名称来查看有关该手册的信息。

4. 在文档详细信息部分中，验证文档版本已设置为要运行的版本。系统包括以下版本选项：

   - 运行时的默认版本 – 如果定期更新自动化运行手册并分配新的默认版本，请选择此选项。
   - 运行时的最新版本 – 如果定期更新自动化运行手册并且想要运行最新更新的版本，请选择此选项。

- 1（默认）– 选择此选项可执行文档的第一个版本，即默认版本。

5. 选择下一步。
6. 在执行自动化文档页面上，选择多账户和区域。
7. 在目标账户和区域部分中，使用账户和组织 (OU)字段指定要在其中运行自动化的不同 AWS 账户或 AWS 组织部门 (OU)。使用逗号分隔多个账户或 OU。
8. 使用 AWS 区域 列表选择要在其中运行自动化的一个或多个区域。
9. 使用多区域和账户速率控制选项将自动化限制为在有限区域中的有限账户运行。这些选项不限制可运行自动化的 AWS 资源的数量。

   a. 在位置(账户-区域对)并发)部分中，选择一个选项以限制可同时在多个账户和区域中运行的自动化的数量。例如，如果选择在位于四 (4) 个 AWS 区域 中的五 (5) 个 AWS 账户 账户中运行自动化，则 Systems Manager 在总共 20 个账户-区域对中运行自动化。您可以使用此选项指定一个绝对数量（例如 **2**），以使自动化仅同时在两个账户-区域对中运行。或者，您也可以指定可同时运行的账户-区域对的百分比。例如，对于 20 个账户-区域对，如果您指定 20%，自动化在最多五 (5) 个账户-区域对中同时运行。

      - 选择目标，以输入可同时运行自动化的账户-区域对的绝对数量。
      - 选择百分比，以输入可同时运行自动化的账户-区域对总数的百分比。

   b. 在错误阈值部分中，选择一个选项：

      - 选择错误，以输入自动化停止将自动化发送到其他资源之前允许的错误绝对数量。
      - 选择百分比，以输入自动化停止将工作流程发送到其他资源之前允许的错误的百分比。

10. 在目标部分中，选择希望如何设置要在其中运行自动化的 AWS 资源。这些选项是必需的。

    a. 使用参数列表选择一个参数。参数列表中的项目由此过程开始时选择的自动化运行手册中的参数确定。通过选择参数，可以定义在其上运行自动化工作流的资源类型。

    b. 使用目标列表选择设置目标资源的方式。

       i. 如果选择使用参数值将资源设置为目标，请输入您在输入参数部分为参数选择的参数值。

       ii. 如果选择使用 AWS Resource Groups 将资源设置为目标，请从资源组列表中选择组的名称。

       iii. 如果选择使用标签将资源设置为目标，请在提供的字段中输入标签键和（可选）标签值。选择添加。

       iv. 如果要在当前 AWS 账户 和 AWS 区域 中的所有实例上运行自动化运行手册，则选择所有实例。

11. 在输入参数 部分中，指定所需的输入。从 AutomationAssumeRole 列表选择 `AWS-SystemsManager-AutomationAdministrationRole` IAM 服务角色。

    > **Note**
    >
    > 您可能不需要选择输入参数部分中的某些选项。原因是您使用标签或资源组将多个区域和账户中的资源设置为目标。例如，如果选择了 `AWS-RestartEC2Instance` 运行手册，则无需在输入参数部分中指定或选择实例 ID。自动化使用您指定的标签以查找要重新启动的实例。

12. （可选）选择一个 CloudWatch 警报以应用于您的自动化进行监控。要将 CloudWatch 警报附加到自动化，启动自动化的 IAM 主体必须具有 `iam:createServiceLinkedRole` 操作的权限。有关 CloudWatch 警报的更多信息，请参阅使用 Amazon CloudWatch 警报。请注意，如果您的警报激活，自动化将取消，并且您定义的任何 `OnCancel` 步骤都会运行。如果使用 AWS CloudTrail，您将在跟踪中看到 API 调用。
13. 使用速率控制部分中的选项限制可在每个账户-区域对中运行自动化的 AWS 资源的数量。

    在并发部分中，选择一个选项：

    - 选择目标，以输入可同时运行自动化工作流目标的绝对数量。
    - 选择百分比，以输入可同时运行自动化工作流的目标集的百分比。
14. 在错误阈值部分中，选择一个选项：

    - 选择错误，以输入自动化停止将工作流程发送到其他资源之前允许的错误的绝对数量。
    - 选择百分比，以输入自动化停止将工作流程发送到其他资源之前允许的错误的百分比。
15. 选择执行。

在多个区域和账户中运行自动化（命令行）

以下过程介绍了如何使用 AWS CLI（在 Linux 或 Windows 上）或 AWS Tools for PowerShell 在自动化管理账户的多个区域和账户中运行自动化。

开始前的准备工作

在完成以下过程之前，请记下以下信息：

- 要在其中运行自动化的 AWS 账户 账户 ID 或 OU。
- 将在其中运行自动化的受 Systems Manager 支持区域。

- 要在其中运行自动化的标签键和标签值或资源组的名称。

在多个区域和账户中运行自动化

1. 安装并配置 AWS CLI 或 AWS Tools for PowerShell（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本以及安装 AWS Tools for PowerShell。

2. 使用以下格式创建一个命令，以便在多个区域和账户中运行自动化。将每个*#######*替换为您自己的信息。

   Linux & macOS

   ```
   aws ssm start-automation-execution \
           --document-name runbook name \
           --parameters AutomationAssumeRole=arn:aws:iam::management account
    ID:role/AWS-SystemsManager-AutomationAdministrationRole \
           --target-parameter-name parameter name \
           --targets Key=tag key,Values=value \
           --target-locations Accounts=account ID,account ID
    2,Regions=Region,Region 2,ExecutionRoleName=AWS-SystemsManager-
   AutomationExecutionRole
   ```

   Windows

   ```
   aws ssm start-automation-execution ^
           --document-name runbook name ^
           --parameters AutomationAssumeRole=arn:aws:iam::management account
    ID:role/AWS-SystemsManager-AutomationAdministrationRole ^
           --target-parameter-name parameter name ^
           --targets Key=tag key,Values=value ^
           --target-locations Accounts=account ID,account ID
    2,Regions=Region,Region 2,ExecutionRoleName=AWS-SystemsManager-
   AutomationExecutionRole
   ```

   PowerShell

   ```
   $Targets = New-Object Amazon.SimpleSystemsManagement.Model.Target
       $Targets.Key = "tag key"
       $Targets.Values = "value"

       Start-SSMAutomationExecution `
   ```

```
        -DocumentName "runbook name" `
        -Parameter @{
        "AutomationAssumeRole"="arn:aws:iam::management account ID:role/AWS-
SystemsManager-AutomationAdministrationRole" } `
        -TargetParameterName "parameter name" `
        -Target $Targets `
        -TargetLocation @{
        "Accounts"="account ID","account ID 2";
        "Regions"="Region","Region 2";
        "ExecutionRoleName"="AWS-SystemsManager-AutomationExecutionRole" }
```

下面是几个示例：

示例 1：此示例重启整个 AWS Organizations 组织中三个区域中的 EC2 实例。这是通过定位组织的根 ID 并包括子 OU 来实现的。

Linux & macOS

```
aws ssm start-automation-execution \
        --document-name "AWS-RestartEC2Instance" \
        --target-parameter-name InstanceId \
        --targets '[{"Key":"AWS::EC2::Instance","Values":["*"]}]' \
        --target-locations '[{
            "Accounts": ["r-example"],
            "IncludeChildOrganizationUnits": true,
            "Regions": ["us-east-1", "us-east-2", "us-west-2"]
        }]'
```

Windows

```
aws ssm start-automation-execution \
        --document-name "AWS-RestartEC2Instance" ^
        --target-parameter-name InstanceId ^
        --targets '[{"Key":"AWS::EC2::Instance","Values":["*"]}]' ^
        --target-locations '[{
            "Accounts": ["r-example"],
            "IncludeChildOrganizationUnits": true,
            "Regions": ["us-east-1", "us-east-2", "us-west-2"]
        }]'
```

PowerShell

```
Start-SSMAutomationExecution `
        -DocumentName "AWS-RestartEC2Instance" `
        -TargetParameterName "InstanceId" `
        -Targets '[{"Key":"AWS::EC2::Instance","Values":["*"]}]'
        -TargetLocation @{
            "Accounts"="r-example";
            "Regions"="us-east-1", "us-east-2", "us-west-2";
            "IncludeChildOrganizationUnits"=true}
```

示例 2：此示例重启不同账户和区域中的特定 EC2 实例。

Linux & macOS

```
aws ssm start-automation-execution \
        --document-name "AWS-RestartEC2Instance" \
        --target-parameter-name InstanceId \
        --target-locations '[{
            "Accounts": ["123456789012"],
            "Targets": [{
                "Key":"ParameterValues",
                "Values":["i-02573cafcfEXAMPLE", "i-0471e04240EXAMPLE"]
            }],
            "TargetLocationMaxConcurrency": "100%",
            "Regions": ["us-east-1"]
        }, {
            "Accounts": ["987654321098"],
            "Targets": [{
                "Key":"ParameterValues",
                "Values":["i-07782c72faEXAMPLE"]
            }],
            "TargetLocationMaxConcurrency": "100%",
            "Regions": ["us-east-2"]
        }]'
```

Windows

```
aws ssm start-automation-execution ^
        --document-name "AWS-RestartEC2Instance" ^
        --target-parameter-name InstanceId ^
```

```
        --target-locations '[{
            "Accounts": ["123456789012"],
            "Targets": [{
                "Key":"ParameterValues",
                "Values":["i-02573cafcfEXAMPLE", "i-0471e04240EXAMPLE"]
            }],
            "TargetLocationMaxConcurrency": "100%",
            "Regions": ["us-east-1"]
        }, {
            "Accounts": ["987654321098"],
            "Targets": [{
                "Key":"ParameterValues",
                "Values":["i-07782c72faEXAMPLE"]
            }],
            "TargetLocationMaxConcurrency": "100%",
            "Regions": ["us-east-2"]
        }]'
```

PowerShell

```
Start-SSMAutomationExecution `
        -DocumentName "AWS-RestartEC2Instance" `
        -TargetParameterName "InstanceId" `
        -Targets '[{"Key":"AWS::EC2::Instance","Values":["*"]}]'
        -TargetLocation @({
            "Accounts"="123456789012",
            "Targets"= @{
                "Key":"ParameterValues",
                "Values":["i-02573cafcfEXAMPLE", "i-0471e04240EXAMPLE"]
            },
            "TargetLocationMaxConcurrency"="100%",
            "Regions"=["us-east-1"]
        }, {
            "Accounts"="987654321098",
            "Targets": @{
                "Key":"ParameterValues",
                "Values":["i-07782c72faEXAMPLE"]
            },
            "TargetLocationMaxConcurrency": "100%",
            "Regions"=["us-east-2"]
        })
```

示例 3：此示例重启 123456789012 和 987654321098 账户中的 EC2 实例，它们位于 us-east-2 和 us-west-1 区域中。必须使用标签键对值 Env-PROD 标记这些实例。

Linux & macOS

```
aws ssm start-automation-execution \
        --document-name AWS-RestartEC2Instance \
        --parameters AutomationAssumeRole=arn:aws:iam::123456789012:role/AWS-
SystemsManager-AutomationAdministrationRole \
        --target-parameter-name InstanceId \
        --targets Key=tag:Env,Values=PROD \
        --target-locations Accounts=123456789012,987654321098,Regions=us-
east-2,us-west-1,ExecutionRoleName=AWS-SystemsManager-AutomationExecutionRole
```

Windows

```
aws ssm start-automation-execution ^
        --document-name AWS-RestartEC2Instance ^
        --parameters AutomationAssumeRole=arn:aws:iam::123456789012:role/AWS-
SystemsManager-AutomationAdministrationRole ^
        --target-parameter-name InstanceId ^
        --targets Key=tag:Env,Values=PROD ^
        --target-locations Accounts=123456789012,987654321098,Regions=us-
east-2,us-west-1,ExecutionRoleName=AWS-SystemsManager-AutomationExecutionRole
```

PowerShell

```
$Targets = New-Object Amazon.SimpleSystemsManagement.Model.Target
    $Targets.Key = "tag:Env"
    $Targets.Values = "PROD"

    Start-SSMAutomationExecution `
        -DocumentName "AWS-RestartEC2Instance" `
        -Parameter @{
        "AutomationAssumeRole"="arn:aws:iam::123456789012:role/AWS-
SystemsManager-AutomationAdministrationRole" } `
        -TargetParameterName "InstanceId" `
        -Target $Targets `
        -TargetLocation @{
        "Accounts"="123456789012","987654321098";
```

```
        "Regions"="us-east-2","us-west-1";
        "ExecutionRoleName"="AWS-SystemsManager-AutomationExecutionRole" }
```

示例 4：此示例重启 123456789012 和 987654321098 账户中的 EC2 实例，它们位于 eu-central-1 区域中。这些实例必须是 prod-instances AWS 资源组的成员。

Linux & macOS

```
aws ssm start-automation-execution \
        --document-name AWS-RestartEC2Instance \
        --parameters AutomationAssumeRole=arn:aws:iam::123456789012:role/AWS-
SystemsManager-AutomationAdministrationRole \
        --target-parameter-name InstanceId \
        --targets Key=ResourceGroup,Values=prod-instances \
        --target-locations Accounts=123456789012,987654321098,Regions=eu-
central-1,ExecutionRoleName=AWS-SystemsManager-AutomationExecutionRole
```

Windows

```
aws ssm start-automation-execution ^
        --document-name AWS-RestartEC2Instance ^
        --parameters AutomationAssumeRole=arn:aws:iam::123456789012:role/AWS-
SystemsManager-AutomationAdministrationRole ^
        --target-parameter-name InstanceId ^
        --targets Key=ResourceGroup,Values=prod-instances ^
        --target-locations Accounts=123456789012,987654321098,Regions=eu-
central-1,ExecutionRoleName=AWS-SystemsManager-AutomationExecutionRole
```

PowerShell

```
$Targets = New-Object Amazon.SimpleSystemsManagement.Model.Target
    $Targets.Key = "ResourceGroup"
    $Targets.Values = "prod-instances"

    Start-SSMAutomationExecution `
        -DocumentName "AWS-RestartEC2Instance" `
        -Parameter @{
        "AutomationAssumeRole"="arn:aws:iam::123456789012:role/AWS-
SystemsManager-AutomationAdministrationRole" } `
        -TargetParameterName "InstanceId" `
        -Target $Targets `
```

```
        -TargetLocation @{
        "Accounts"="123456789012","987654321098";
        "Regions"="eu-central-1";
        "ExecutionRoleName"="AWS-SystemsManager-AutomationExecutionRole" }
```

示例 5：此示例重启 ou-1a2b3c-4d5e6c AWS 组织单位 (OU) 中的 EC2 实例。这些实例位于 us-west-1 和 us-west-2 区域中。这些实例必须是 WebServices AWS 资源组的成员。

Linux & macOS

```
aws ssm start-automation-execution \
        --document-name AWS-RestartEC2Instance \
        --parameters AutomationAssumeRole=arn:aws:iam::123456789012:role/AWS-
SystemsManager-AutomationAdministrationRole \
        --target-parameter-name InstanceId \
        --targets Key=ResourceGroup,Values=WebServices \
        --target-locations Accounts=ou-1a2b3c-4d5e6c,Regions=us-west-1,us-
west-2,ExecutionRoleName=AWS-SystemsManager-AutomationExecutionRole
```

Windows

```
aws ssm start-automation-execution ^
        --document-name AWS-RestartEC2Instance ^
        --parameters AutomationAssumeRole=arn:aws:iam::123456789012:role/AWS-
SystemsManager-AutomationAdministrationRole ^
        --target-parameter-name InstanceId ^
        --targets Key=ResourceGroup,Values=WebServices ^
        --target-locations Accounts=ou-1a2b3c-4d5e6c,Regions=us-west-1,us-
west-2,ExecutionRoleName=AWS-SystemsManager-AutomationExecutionRole
```

PowerShell

```
$Targets = New-Object Amazon.SimpleSystemsManagement.Model.Target
    $Targets.Key = "ResourceGroup"
    $Targets.Values = "WebServices"

    Start-SSMAutomationExecution `
        -DocumentName "AWS-RestartEC2Instance" `
        -Parameter @{
        "AutomationAssumeRole"="arn:aws:iam::123456789012:role/AWS-
SystemsManager-AutomationAdministrationRole" } `
```

```
        -TargetParameterName "InstanceId" `
        -Target $Targets `
        -TargetLocation @{
        "Accounts"="ou-1a2b3c-4d5e6c";
        "Regions"="us-west-1";
        "ExecutionRoleName"="AWS-SystemsManager-AutomationExecutionRole" }
```

系统返回类似于以下内容的信息。

Linux & macOS

```
{
        "AutomationExecutionId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE"
    }
```

Windows

```
{
        "AutomationExecutionId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE"
    }
```

PowerShell

```
4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE
```

3. 运行以下命令以查看自动化的详细信息。将 *automation execution ID* 替换为您自己的信息。

Linux & macOS

```
aws ssm describe-automation-executions \
        --filters Key=ExecutionId,Values=automation execution ID
```

Windows

```
aws ssm describe-automation-executions ^
        --filters Key=ExecutionId,Values=automation execution ID
```

PowerShell

```
Get-SSMAutomationExecutionList | `
        Where {$_.AutomationExecutionId -eq "automation execution ID"}
```

4. 运行以下命令以查看自动化进程的详细信息。

Linux & macOS

```
aws ssm get-automation-execution \
        --automation-execution-id 4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE
```

Windows

```
aws ssm get-automation-execution ^
        --automation-execution-id 4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE
```

PowerShell

```
Get-SSMAutomationExecution `
        -AutomationExecutionId a4a3c0e9-7efd-462a-8594-01234EXAMPLE
```

> **Note**
>
> 您也可以在控制台中监控自动化的状态。在自动化执行列表中，请选择您刚才运行的自动化，然后选择执行步骤选项卡。该选项卡显示自动化操作的状态。

更多信息

通过 AWS Systems Manager 自动化进行集中式多账户和多区域修补

基于 EventBridge 事件运行自动化

您可以通过指定运行手册作为 Amazon EventBridge 事件的目标来启动自动化。您可以按计划或者在特定 AWS 系统事件发生时启动自动化。例如，假设您创建一个名为 BootStrapInstances 的运行手册，该运行手册在实例启动时在实例上安装软件。要指定 BootStrapInstances 运行手册（和对

应的工作流程）作为 EventBridge 事件的目标，请先创建新的 EventBridge 规则。(以下是示例规则：Service name：EC2，Event Type：EC2 实例状态更改通知，Specific state(s)：正在运行，Any instance。) 然后，您使用以下过程指定 BootStrapInstances 运行手册作为使用 EventBridge 控制台和 AWS Command Line Interface (AWS CLI) 的事件的目标。当新实例启动时，系统将运行自动化并安装软件。

有关创建运行手册的更多信息，请参阅 创建您自己的运行手册。

创建使用运行手册（控制台）的 EventBridge 事件

使用以下过程将运行手册配置为 EventBridge 事件目标。

将运行手册配置为 EventBridge 事件规则的目标

1. 打开位于 https://console.aws.amazon.com/events/ 的 Amazon EventBridge 控制台。
2. 在导航窗格中，选择规则。
3. 选择创建规则。
4. 为规则输入名称和描述。

   规则不能与同一区域中的另一个规则和同一事件总线上的名称相同。
5. 对于事件总线，请选择要与此规则关联的事件总线。如果您希望此规则响应来自您自己的 AWS 账户的匹配事件，请选择 defaul（默认）。当您账户中的某个 AWS 服务发出一个事件时，它始终会发送到您账户的默认事件总线。
6. 选择触发规则的方式。

要基于以下内容创建 VPC	请执行此操作...
事件	a. 对于规则类型，选择具有事件模式的规则。 b. 选择下一步。 c. 对于事件源，选择AWS 事件或 EventBridge 合作伙伴事件。 d. 在 Event pattern（事件模式）部分中，执行以下操作之一：

要基于以下内容创建 VPC	请执行此操作...	
	• 要使用模板创建您的事件模式，请选择事件模式表单，然后选择事件源、AWS 服务和 事件类型。如果您选择 All Events（所有事件）作为事件类型，此 AWS 服务发送的所有事件将会匹配规则。 要自定义模板，请选择 Custom pattern (JSON editor)（自定义模式（JSON 编辑器））进行您的更改。 • 要使用自定义事件模式，请选择 Custom pattern (JSON editor)（自定义模式（JSON 编辑器）），然后创建您的事件模式。	

要基于以下内容创建 VPC	请执行此操作...
计划	a. 对于 Rule type（规则类型），选择 Schedule（计划）。 b. 选择下一步。 c. 对于 Schedule pattern（计划模式），执行以下操作之一： • 要使用 cron 表达式定义计划，请选择 A fine-grained schedule that runs at a specific time, such as 8:00 a.m.（在特定时间（例如上午 8:00）运行的精细计划） PST on the first Monday of every month（每月第一个星期一，太平洋标准时间），然后输入 cron 表达式。 • 要使用速率表达式定义时间表，请选择 A schedule that runs at a regular rate, such as every 10 minutes（以常规速率运行的计划，例如每 10 分钟），然后输入速率表达式。

7. 选择下一步。
8. 对于目标类型，选择AWS 服务。
9. 对于 Target（目标），选择 Systems Manager Automation。
10. 对于文档，选择在调用目标时要使用的运行手册。

11. 在 Configure automation parameter(s)（配置自动化参数）部分中，保留默认参数值（如果可用）或输入您自己的值。

> **Note**
>
> 要创建目标，您必须为每个所需的参数指定一个值。如果不指定，系统虽然会创建规则，但该规则不会运行。

12. 对于许多目标类型，EventBridge 需要权限以便将事件发送到目标。在这些情况下，EventBridge 可以创建运行事件所需的 IAM 角色：请执行以下操作之一：

    - 要自动创建 IAM 角色，请选择为此特定资源创建新角色。
    - 要使用您之前创建的 IAM 角色，请选择 Use existing role（使用现有角色），然后从下拉列表中选择现有角色。请注意，您可能需要更新 IAM 角色的信任策略以包括 EventBridge。以下是示例：

    JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "events.amazonaws.com",
                    "ssm.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

13. 选择下一步。
14. （可选）为规则输入一个或多个标签。有关更多信息，请参阅 Amazon EventBridge 用户指南中的标记 Amazon EventBridge 资源。
15. 选择 Next（下一步）。

16. 查看规则详细信息并选择创建规则。

创建一个使用运行手册的 EventBridge 事件（命令行）

以下过程介绍了如何使用 AWS CLI（在 Linux 或 Windows 上）或 AWS Tools for PowerShell 创建 EventBridge 事件规则并将运行手册配置为目标。

将运行手册配置为 EventBridge 事件规则的目标

1. 安装并配置 AWS CLI 或 AWS Tools for PowerShell（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本以及安装 AWS Tools for PowerShell。

2. 创建一个命令以指定新的 EventBridge 事件规则。将每个*******替换为您自己的信息。

   基于计划的触发器

   Linux & macOS

   ```
   aws events put-rule \
   --name "rule name" \
   --schedule-expression "cron or rate expression"
   ```

   Windows

   ```
   aws events put-rule ^
   --name "rule name" ^
   --schedule-expression "cron or rate expression"
   ```

   PowerShell

   ```
   Write-CWERule `
   -Name "rule name" `
   -ScheduleExpression "cron or rate expression"
   ```

   以下示例创建一个 EventBridge 事件规则，将在每天上午 9:00 (UTC) 启动该规则。

   Linux & macOS

   ```
   aws events put-rule \
   ```

```
--name "DailyAutomationRule" \
--schedule-expression "cron(0 9 * * ? *)"
```

Windows

```
aws events put-rule ^
--name "DailyAutomationRule" ^
--schedule-expression "cron(0 9 * * ? *)"
```

PowerShell

```
Write-CWERule `
-Name "DailyAutomationRule" `
-ScheduleExpression "cron(0 9 * * ? *)"
```

基于事件的触发器

Linux & macOS

```
aws events put-rule \
--name "rule name" \
--event-pattern "{\"source\":[\"aws.service\"],\"detail-type\":[\"service event
 detail type\"]}"
```

Windows

```
aws events put-rule ^
--name "rule name" ^
--event-pattern "{\"source\":[\"aws.service\"],\"detail-type\":[\"service event
 detail type\"]}"
```

PowerShell

```
Write-CWERule `
-Name "rule name" `
-EventPattern '{"source":["aws.service"],"detail-type":["service event detail
 type"]}'
```

以下示例创建一个 EventBridge 事件规则，将在区域中的任何 EC2 实例更改状态时启动该规则。

Linux & macOS

```
aws events put-rule \
--name "EC2InstanceStateChanges" \
--event-pattern "{\"source\":[\"aws.ec2\"],\"detail-type\":[\"EC2 Instance
 State-change Notification\"]}"
```

Windows

```
aws events put-rule ^
--name "EC2InstanceStateChanges" ^
--event-pattern "{\"source\":[\"aws.ec2\"],\"detail-type\":[\"EC2 Instance
 State-change Notification\"]}"
```

PowerShell

```
Write-CWERule `
-Name "EC2InstanceStateChanges" `
-EventPattern '{"source":["aws.ec2"],"detail-type":["EC2 Instance State-change
 Notification"]}'
```

该命令返回新 EventBridge 规则的详细信息，类似于以下内容。

Linux & macOS

```
{
"RuleArn": "arn:aws:events:us-east-1:123456789012:rule/automationrule"
}
```

Windows

```
{
"RuleArn": "arn:aws:events:us-east-1:123456789012:rule/automationrule"
}
```

PowerShell

```
arn:aws:events:us-east-1:123456789012:rule/EC2InstanceStateChanges
```

3. 创建一个命令以将运行手册指定为在步骤 2 中创建的 EventBridge 事件规则的目标。将每个*########*替换为您自己的信息。

Linux & macOS

```
aws events put-targets \
--rule rule name \
--targets '{"Arn": " arn:aws:ssm:region:account ID:automation-definition/runbook
 name","Input":"{\"Message\":[\"{\\\"Key\\\":\\\"key name\\\",\\\"Values\\\":[\\
\"value\\\"]}\"]}","Id": "target ID","RoleArn": "arn:aws:iam::123456789012:role/
service-role/EventBridge service role"}'
```

Windows

```
aws events put-targets ^
--rule rule name ^
--targets '{"Arn": "arn:aws:ssm:region:account ID:automation-definition/runbook
 name","Input":"{\"Message\":[\"{\\\"Key\\\":\\\"key name\\\",\\\"Values\\\":[\\
\"value\\\"]}\"]}","Id": "target ID","RoleArn": "arn:aws:iam::123456789012:role/
service-role/EventBridge service role"}'
```

PowerShell

```
$Target = New-Object Amazon.CloudWatchEvents.Model.Target
$Target.Id = "target ID"
$Target.Arn = "arn:aws:ssm:region:account ID:automation-definition/runbook name"
$Target.RoleArn = "arn:aws:iam::123456789012:role/service-role/EventBridge
 service role"
$Target.Input = '{"input parameter":["value"],"AutomationAssumeRole":
["arn:aws:iam::123456789012:role/AutomationServiceRole"]}'

Write-CWETarget `
-Rule "rule name" `
-Target $Target
```

以下示例创建一个 EventBridge 事件目标，它使用 AWS-StartEC2Instance 运行手册启动指定的实例 ID。

Linux & macOS

```
aws events put-targets \
--rule DailyAutomationRule \
--targets '{"Arn": "arn:aws:ssm:region:*:automation-definition/AWS-
StartEC2Instance","Input":"{\"InstanceId\":[\"i-02573cafcfEXAMPLE\"],
\"AutomationAssumeRole\":[\"arn:aws:iam::123456789012:role/AutomationServiceRole
\"]}","Id": "Target1","RoleArn": "arn:aws:iam::123456789012:role/service-role/
AWS_Events_Invoke_Start_Automation_Execution_1213609520"}'
```

Windows

```
aws events put-targets ^
--rule DailyAutomationRule ^
--targets '{"Arn": "arn:aws:ssm:region:*:automation-definition/AWS-
StartEC2Instance","Input":"{\"InstanceId\":[\"i-02573cafcfEXAMPLE\"],
\"AutomationAssumeRole\":[\"arn:aws:iam::123456789012:role/AutomationServiceRole
\"]}","Id": "Target1","RoleArn": "arn:aws:iam::123456789012:role/service-role/
AWS_Events_Invoke_Start_Automation_Execution_1213609520"}'
```

PowerShell

```
$Target = New-Object Amazon.CloudWatchEvents.Model.Target
$Target.Id = "Target1"
$Target.Arn = "arn:aws:ssm:region:*:automation-definition/AWS-StartEC2Instance"
$Target.RoleArn = "arn:aws:iam::123456789012:role/service-role/
AWS_Events_Invoke_Start_Automation_Execution_1213609520"
$Target.Input = '{"InstanceId":["i-02573cafcfEXAMPLE"],"AutomationAssumeRole":
["arn:aws:iam::123456789012:role/AutomationServiceRole"]}'

Write-CWETarget `
-Rule "DailyAutomationRule" `
-Target $Target
```

系统将返回类似于以下内容的信息。

Linux & macOS

```
{
"FailedEntries": [],
"FailedEntryCount": 0
}
```

Windows

```
{
"FailedEntries": [],
"FailedEntryCount": 0
}
```

PowerShell

如果命令在 PowerShell 中成功，则没有输出。

要按步骤运行自动化

以下过程介绍了如何通过手动执行模式使用 AWS Systems Manager 控制台和 AWS Command Line Interface (AWS CLI) 运行自动化。通过使用手动执行模式，自动化在两个步骤之间从等待状态开始，在等待状态暂停。这允许您控制自动化何时继续，如果您需要先查看某个步骤的结果，然后再继续，这会很有用。

自动化在当前用户的上下文中运行。因此，只要您有权使用运行手册和此运行手册调用的任何操作，就无需配置其他 IAM 权限。如果您在 IAM 中拥有管理员权限，则已有权运行此自动化。

按步骤运行自动化（控制台）

以下过程介绍了如何使用 Systems Manager 控制台按步骤手动运行自动化。

要按步骤运行自动化，请执行以下操作

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择自动化，然后选择执行自动化。

3. 在自动化文档列表中，请选择运行手册。在文档类别窗格中选择一个或多个选项，以便根据 SSM 文档的用途对其进行筛选。要查看您拥有的运行手册，请选择我拥有的选项卡。要查看与您的账户共享的运行手册，请选择与我共享选项卡。要查看所有运行手册，请选择所有文档选项卡。

   > **Note**
   >
   > 您可以通过选择运行手册名称来查看有关该手册的信息。

4. 在文档详细信息部分中，验证文档版本已设置为要运行的版本。系统包括以下版本选项：

   - 运行时的默认版本 – 如果定期更新自动化运行手册并分配新的默认版本，请选择此选项。
   - 运行时的最新版本 – 如果定期更新自动化运行手册并且想要运行最新更新的版本，请选择此选项。
   - 1（默认）– 选择此选项可执行文档的第一个版本，即默认版本。

5. 选择下一步。
6. 在执行模式部分中，选择手动执行。
7. 在输入参数部分中，指定所需的输入。或者，您也可以从 AutomationAssumeRole 列表选择一个 IAM 服务角色。
8. 选择执行。
9. 当您准备好开始自动化的第一步时，选择执行此步骤。自动化将执行步骤 1，并在运行您在此过程的步骤 3 中选择的运行手册中指定的任何后续步骤之前暂停。如果运行手册具有多个步骤，则必须为每个步骤选择执行此步骤以让自动化继续。每次选择执行此步骤，操作就会运行。

   > **Note**
   >
   > 控制台将显示自动化的状态。如果自动化无法运行某个步骤，请参阅 Systems Manager 自动化故障排除。

10. 在完成运行手册中指定的所有步骤后，选择完成并查看结果以完成自动化并查看结果。

按步骤运行自动化（命令行）

以下过程介绍了如何使用 AWS CLI（在 Linux、macOS 或 Windows 中）或 AWS Tools for PowerShell 按步骤手动运行自动化。

要按步骤运行自动化，请执行以下操作：

1. 安装并配置 AWS CLI 或 AWS Tools for PowerShell（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本以及安装 AWS Tools for PowerShell。
2. 运行以下命令以启动一个手动自动化。将每个*#######*替换为您自己的信息。

   Linux & macOS

   ```
   aws ssm start-automation-execution \
       --document-name runbook name \
       --mode Interactive \
       --parameters runbook parameters
   ```

   Windows

   ```
   aws ssm start-automation-execution ^
       --document-name runbook name ^
       --mode Interactive ^
       --parameters runbook parameters
   ```

   PowerShell

   ```
   Start-SSMAutomationExecution `
       -DocumentName runbook name `
       -Mode Interactive `
       -Parameter runbook parameters
   ```

   以下是使用 `AWS-RestartEC2Instance` 运行手册重新启动指定 EC2 实例的示例。

   Linux & macOS

   ```
   aws ssm start-automation-execution \
       --document-name "AWS-RestartEC2Instance" \
       --mode Interactive \
       --parameters "InstanceId=i-02573cafcfEXAMPLE"
   ```

   Windows

   ```
   aws ssm start-automation-execution ^
   ```

```
    --document-name "AWS-RestartEC2Instance" ^
    --mode Interactive ^
    --parameters "InstanceId=i-02573cafcfEXAMPLE"
```

PowerShell

```
Start-SSMAutomationExecution `
    -DocumentName AWS-RestartEC2Instance `
    -Mode Interactive
    -Parameter @{"InstanceId"="i-02573cafcfEXAMPLE"}
```

系统将返回类似于以下内容的信息。

Linux & macOS

```
{
    "AutomationExecutionId": "ba9cd881-1b36-4d31-a698-0123456789ab"
}
```

Windows

```
{
    "AutomationExecutionId": "ba9cd881-1b36-4d31-a698-0123456789ab"
}
```

PowerShell

```
ba9cd881-1b36-4d31-a698-0123456789ab
```

3. 在准备好开始执行自动化的第一步时，运行以下命令。将每个*#######*替换为您自己的信息。自动化将执行步骤 1，并在运行您在该过程的步骤 1 中选择的运行手册中指定的任何后续步骤之前暂停。如果运行手册具有多个步骤，则必须为每个步骤运行以下命令以使自动化继续执行。

   Linux & macOS

```
aws ssm send-automation-signal \
    --automation-execution-id ba9cd881-1b36-4d31-a698-0123456789ab \
    --signal-type StartStep \
    --payload StepName="stopInstances"
```

Windows

```
aws ssm send-automation-signal ^
    --automation-execution-id ba9cd881-1b36-4d31-a698-0123456789ab ^
    --signal-type StartStep ^
    --payload StepName="stopInstances"
```

PowerShell

```
Send-SSMAutomationSignal `
    -AutomationExecutionId ba9cd881-1b36-4d31-a698-0123456789ab `
    -SignalType StartStep
    -Payload @{"StepName"="stopInstances"}
```

如果此命令成功，则无任何输出。

4. 运行以下命令以检索自动化中的每个步骤执行的状态。

Linux & macOS

```
aws ssm describe-automation-step-executions \
    --automation-execution-id ba9cd881-1b36-4d31-a698-0123456789ab
```

Windows

```
aws ssm describe-automation-step-executions ^
    --automation-execution-id ba9cd881-1b36-4d31-a698-0123456789ab
```

PowerShell

```
Get-SSMAutomationStepExecution `
    -AutomationExecutionId ba9cd881-1b36-4d31-a698-0123456789ab
```

系统将返回类似于以下内容的信息。

Linux & macOS

```
{
```

```
    "StepExecutions": [
        {
            "StepName": "stopInstances",
            "Action": "aws:changeInstanceState",
            "ExecutionStartTime": 1557167178.42,
            "ExecutionEndTime": 1557167220.617,
            "StepStatus": "Success",
            "Inputs": {
                "DesiredState": "\"stopped\"",
                "InstanceIds": "[\"i-02573cafcfEXAMPLE\"]"
            },
            "Outputs": {
                "InstanceStates": [
                    "stopped"
                ]
            },
            "StepExecutionId": "654243ba-71e3-4771-b04f-0123456789ab",
            "OverriddenParameters": {},
            "ValidNextSteps": [
                "startInstances"
            ]
        },
        {
            "StepName": "startInstances",
            "Action": "aws:changeInstanceState",
            "ExecutionStartTime": 1557167273.754,
            "ExecutionEndTime": 1557167480.73,
            "StepStatus": "Success",
            "Inputs": {
                "DesiredState": "\"running\"",
                "InstanceIds": "[\"i-02573cafcfEXAMPLE\"]"
            },
            "Outputs": {
                "InstanceStates": [
                    "running"
                ]
            },
            "StepExecutionId": "8a4a1e0d-dc3e-4039-a599-0123456789ab",
            "OverriddenParameters": {}
        }
    ]
}
```

Windows


```
{
    "StepExecutions": [
        {
            "StepName": "stopInstances",
            "Action": "aws:changeInstanceState",
            "ExecutionStartTime": 1557167178.42,
            "ExecutionEndTime": 1557167220.617,
            "StepStatus": "Success",
            "Inputs": {
                "DesiredState": "\"stopped\"",
                "InstanceIds": "[\"i-02573cafcfEXAMPLE\"]"
            },
            "Outputs": {
                "InstanceStates": [
                    "stopped"
                ]
            },
            "StepExecutionId": "654243ba-71e3-4771-b04f-0123456789ab",
            "OverriddenParameters": {},
            "ValidNextSteps": [
                "startInstances"
            ]
        },
        {
            "StepName": "startInstances",
            "Action": "aws:changeInstanceState",
            "ExecutionStartTime": 1557167273.754,
            "ExecutionEndTime": 1557167480.73,
            "StepStatus": "Success",
            "Inputs": {
                "DesiredState": "\"running\"",
                "InstanceIds": "[\"i-02573cafcfEXAMPLE\"]"
            },
            "Outputs": {
                "InstanceStates": [
                    "running"
                ]
            },
            "StepExecutionId": "8a4a1e0d-dc3e-4039-a599-0123456789ab",
            "OverriddenParameters": {}
        }
```

```
    ]
}
```

PowerShell

```
Action: aws:changeInstanceState
ExecutionEndTime     : 5/6/2019 19:45:46
ExecutionStartTime   : 5/6/2019 19:45:03
FailureDetails       :
FailureMessage       :
Inputs               : {[DesiredState, "stopped"], [InstanceIds,
 ["i-02573cafcfEXAMPLE"]]}
IsCritical           : False
IsEnd                : False
MaxAttempts          : 0
NextStep             :
OnFailure            :
Outputs              : {[InstanceStates,
 Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
OverriddenParameters : {}
Response             :
ResponseCode         :
StepExecutionId      : 8fcc9641-24b7-40b3-a9be-0123456789ab
StepName             : stopInstances
StepStatus           : Success
TimeoutSeconds       : 0
ValidNextSteps       : {startInstances}
```

5. 在完成所选运行手册中指定的所有步骤后，运行以下命令以完成自动化。将每个*#######*替换为您自己的信息。

Linux & macOS

```
aws ssm stop-automation-execution \
    --automation-execution-id ba9cd881-1b36-4d31-a698-0123456789ab \
    --type Complete
```

Windows

```
aws ssm stop-automation-execution ^
    --automation-execution-id ba9cd881-1b36-4d31-a698-0123456789ab ^
    --type Complete
```

PowerShell

```
Stop-SSMAutomationExecution `
    -AutomationExecutionId ba9cd881-1b36-4d31-a698-0123456789ab `
    -Type Complete
```

如果此命令成功，则无任何输出。

使用 State Manager 关联调度自动化

您可以通过创建与运行手册的State Manager关联启动自动化。State Manager是 AWS Systems Manager 中的一项工具。通过创建与运行手册的 State Manager 关联，您可以将不同类型的 AWS 资源设置为目标。例如，您可以创建强制 AWS 资源进入预期状态的关联，包括：

- 将 Systems Manager 角色附加到 Amazon Elastic Compute Cloud (Amazon EC2) 实例，以使其成为托管实例。
- 对安全组强制实施所需的入口和出口规则。
- 创建或删除 Amazon DynamoDB 备份。
- 创建或删除 Amazon Elastic Block Store (Amazon EBS) 快照。
- 关闭 Amazon Simple Storage Service (Amazon S3) 存储桶的读写权限。
- 启动、重新启动或停止托管实例和 Amazon Relational Database Service (Amazon RDS) 实例。
- 将修补程序应用到 Linux、macOS 和 Windows AMIs。

可以使用以下过程通过 AWS Systems Manager 控制台 和 AWS Command Line Interface (AWS CLI) 创建 State Manager 关联以运行自动化。有关关联的一般信息以及有关创建使用 SSM Command 文档或 Policy 文档的关联的信息，请参阅 创建关联。

开始前的准备工作

请注意以下重要详细信息，然后再使用 State Manager 运行自动化。

- 您必须先验证是否已为 Automation（AWS Systems Manager 中的一项工具）配置权限，然后才能创建运行某个运行手册的关联。有关更多信息，请参阅 设置自动化。

- 运行多个运行手册的 State Manager 关联将计入在您的 AWS 账户 中并发运行的自动化的最大数。一次最多可以运行 100 个并发自动化。有关信息，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service quotas。
- 运行自动化时，State Manager 不记录自动化在 AWS CloudTrail 中发起的 API 操作。
- Systems Manager 自动创建服务相关角色，以便 State Manager 有权调用 Systems Manager 自动化 API 操作。如果需要，您可以从 AWS CLI 或 AWS Tools for PowerShell 中运行以下命令以自行创建服务相关角色。

  Linux & macOS

  ```
  aws iam create-service-linked-role \
  --aws-service-name ssm.amazonaws.com
  ```

  Windows

  ```
  aws iam create-service-linked-role ^
  --aws-service-name ssm.amazonaws.com
  ```

  PowerShell

  ```
  New-IAMServiceLinkedRole `
  -AWSServiceName ssm.amazonaws.com
  ```

  有关服务相关角色的更多信息，请参阅 将服务相关角色用于 Systems Manager。

创建运行自动化的关联（控制台）

以下过程介绍了如何使用 Systems Manager 控制台创建运行自动化的 State Manager 关联。

创建 State Manager 关联以运行自动化

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 State Manager，然后选择创建关联。
3. 在名称 字段中指定名称。您可以自由选择，但我们建议您这样做。
4. 在文档列表中，选择运行手册。使用搜索栏来筛选 Document type : Equal : Automation 运行手册。使用搜索栏右侧的数字可查看更多运行手册。

> **Note**
>
> 您可以通过选择运行手册名称来查看有关该手册的信息。

5. 选择简单执行来通过指定一个或多个目标的资源 ID 在这些目标上运行自动化。选择速率控制可以通过指定设置目标选项（如标签或 AWS Resource Groups）来在 AWS 资源队列上运行自动化。您还可以通过指定并发和错误阈值来控制跨各个资源的自动化的执行。

   如果您选择速率控制，将显示目标部分。

6. 在目标部分中，选择一种设置资源目标的方法。

   a. （必需）在参数 列表中，选择一个参数。参数列表中的项目由此过程开始时选择的运行手册中的参数确定。通过选择参数，可以定义自动化运行的资源类型。

   b. （必需）在目标列表中，选择一种设置资源目标的方法。

      - 资源组：从资源组列表中选择组的名称。有关在运行手册中将 AWS Resource Groups 设置为目标的更多信息，请参阅 设置目标 AWS Resource Groups。
      - 标签：在提供的字段中输入标签键和标签值（可选）。选择添加。有关在运行手册中设置标签目标的更多信息，请参阅 设置目标标签。
      - 参数值：在输入参数部分输入值。如果指定多个值，Systems Manager 将在指定的每个值上运行子自动化。

        例如，假设运行手册包含 InstanceID 参数。如果在运行此自动化时将 InstanceID 参数值设置为目标，则 Systems Manager 会为指定的每个实例 ID 值运行一个子自动化。当自动化完成每个指定实例的运行或执行失败时，父自动化完成。您最多可以将 50 个参数值设置为目标。有关在运行手册中设置参数值目标的更多信息，请参阅 设置目标参数值。

7. 在输入参数部分，指定所需的输入参数。

   如果选择使用标签或资源组将资源设置为目标，则不需要选择输入参数部分中的某些选项。例如，如果选择 `AWS-RestartEC2Instance` 运行手册，并且选择使用标签将实例设置为目标，则无需在输入参数部分中指定或选择实例 ID。自动化使用您指定的标签查找要重新启动的实例。

> **Important**
>
> 您必须在 AutomationAssumeRole 字段中指定一个角色 ARN。State Manager 使用担任角色来调用运行手册中指定的 AWS 服务并代表您运行自动化关联。

8. 如果您希望定期运行关联，请在指定计划部分中，选择按计划。如果您选择此选项，则使用提供的选项使用 Cron 或 Rate 表达式创建计划。有关 State Manager 的 Cron 和 Rate 表达式的更多信息，请参阅 适用于关联的 Cron 和 Rate 表达式。

> **Note**
>
> Rate 表达式是运行多个运行手册的 State Manager 关联的首选计划机制。如果您达到了并发运行自动化的最大数，Rate 表达式会为运行关联提供更多灵活性。使用速率计划，Systems Manager 可以在收到并发自动化已达最大值并已被节流的通知后，立即重试自动化。

如果您希望运行关联一次，请选择无计划。

9. （可选）在 Rate Control（速率控制）部分中，选择 Concurrency（并发）和 Error threshold（错误阈值）选项来控制跨各个 AWS 资源的自动化部署。

   a. 在并发部分中，选择一个选项：

      - 选择目标，以输入可同时运行自动化的目标的绝对数量。
      - 选择百分比，以输入可同时运行自动化的目标集的百分比。

   b. 在错误阈值部分中，选择一个选项：

      - 选择错误，以输入自动化停止将自动化发送到其他资源之前允许的错误绝对数量。
      - 选择百分比，以输入自动化停止将自动化发送到其他资源之前允许的错误的百分比。

   有关使用自动化的目标和速率控制的更多信息，请参阅 大规模运行自动化操作。

10. 选择创建关联。

> **Important**
>
> 在创建关联时，将针对指定目标立即运行关联。然后，关联基于您选择的 Cron 或 Rate 表达式运行。如果您选择了无计划，则关联不会再次运行。

创建关联以运行自动化（命令行）

以下过程介绍了如何使用 AWS CLI（在 Linux 或 Windows 上）或 AWS Tools for PowerShell 创建 State Manager 关联以运行自动化。

开始前的准备工作

在您完成以下过程之前，请确保创建了包含运行运行手册所需权限的 IAM 服务角色，并为 Automation（AWS Systems Manager 中的一项工具）配置了信任关系。有关更多信息，请参阅 任务 1：为自动化创建服务角色。

创建关联以运行自动化

1. 安装并配置 AWS CLI 或 AWS Tools for PowerShell（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本以及安装 AWS Tools for PowerShell。

2. 运行以下命令以查看文档列表。

   Linux & macOS

   ```
   aws ssm list-documents
   ```

   Windows

   ```
   aws ssm list-documents
   ```

   PowerShell

   ```
   Get-SSMDocumentList
   ```

   记下要用于关联的运行手册的名称。

3. 运行以下命令以查看有关运行手册的详细信息。在下面的命令中，将 *runbook name* 替换为您自己的信息。

   Linux & macOS

   ```
   aws ssm describe-document \
   --name runbook name
   ```

记下要用于 --automation-target-parameter-name 选项的参数名称（例如 InstanceId）。此参数确定在其上运行自动化的资源的类型。

Windows

```
aws ssm describe-document ^
--name runbook name
```

记下要用于 --automation-target-parameter-name 选项的参数名称（例如 InstanceId）。此参数确定在其上运行自动化的资源的类型。

PowerShell

```
Get-SSMDocumentDescription `
-Name runbook name
```

记下要用于 AutomationTargetParameterName 选项的参数名称（例如 InstanceId）。此参数确定在其上运行自动化的资源的类型。

4. 创建一个命令以使用 State Manager 关联运行自动化。将每个#######替换为您自己的信息。

使用标签设置目标

Linux & macOS

```
aws ssm create-association \
--association-name association name \
--targets Key=tag:key name,Values=value \
--name runbook name \
--parameters AutomationAssumeRole=arn:aws:iam::123456789012:role/
RunbookAssumeRole \
--automation-target-parameter-name target parameter \
--schedule "cron or rate expression"
```

> **Note**
>
> 如果您使用 AWS CLI 创建关联，请使用 --targets 参数指定关联的目标实例。不要使用 --instance-id 参数。--instance-id 参数是一个旧参数。

Windows

```
aws ssm create-association ^
--association-name association name ^
--targets Key=tag:key name,Values=value ^
--name runbook name ^
--parameters AutomationAssumeRole=arn:aws:iam::123456789012:role/
RunbookAssumeRole ^
--automation-target-parameter-name target parameter ^
--schedule "cron or rate expression"
```

> **Note**
>
> 如果您使用 AWS CLI 创建关联，请使用 `--targets` 参数指定关联的目标实例。不要使用 `--instance-id` 参数。`--instance-id` 参数是一个旧参数。

PowerShell

```
$Targets = New-Object Amazon.SimpleSystemsManagement.Model.Target
$Targets.Key = "tag:key name"
$Targets.Values = "value"

New-SSMAssociation `
-AssociationName "association name" `
-Target $Targets `
-Name "runbook name" `
-Parameters @{
"AutomationAssumeRole"="arn:aws:iam::123456789012:role/RunbookAssumeRole" } `
-AutomationTargetParameterName "target parameter" `
-ScheduleExpression "cron or rate expression"
```

> **Note**
>
> 如果您使用 AWS Tools for PowerShell 创建关联，请使用 `Target` 参数指定关联的目标实例。不要使用 `InstanceId` 参数。`InstanceId` 参数是一个旧参数。

使用参数值设置目标

Linux & macOS

```
aws ssm create-association \
--association-name association name \
--targets Key=ParameterValues,Values=value,value 2,value 3 \
--name runbook name \
--parameters AutomationAssumeRole=arn:aws:iam::123456789012:role/
RunbookAssumeRole \
--automation-target-parameter-name target parameter \
--schedule "cron or rate expression"
```

Windows

```
aws ssm create-association ^
--association-name association name ^
--targets Key=ParameterValues,Values=value,value 2,value 3 ^
--name runbook name ^
--parameters AutomationAssumeRole=arn:aws:iam::123456789012:role/
RunbookAssumeRole ^
--automation-target-parameter-name target parameter ^
--schedule "cron or rate expression"
```

PowerShell

```
$Targets = New-Object Amazon.SimpleSystemsManagement.Model.Target
$Targets.Key = "ParameterValues"
$Targets.Values = "value","value 2","value 3"

New-SSMAssociation `
-AssociationName "association name" `
-Target $Targets `
-Name "runbook name" `
-Parameters @{
"AutomationAssumeRole"="arn:aws:iam::123456789012:role/RunbookAssumeRole"} `
-AutomationTargetParameterName "target parameter" `
-ScheduleExpression "cron or rate expression"
```

使用 AWS Resource Groups 设置目标

Linux & macOS

```
aws ssm create-association \
--association-name association name \
--targets Key=ResourceGroup,Values=resource group name \
--name runbook name \
--parameters AutomationAssumeRole=arn:aws:iam::123456789012:role/
RunbookAssumeRole \
--automation-target-parameter-name target parameter \
--schedule "cron or rate expression"
```

Windows

```
aws ssm create-association ^
--association-name association name ^
--targets Key=ResourceGroup,Values=resource group name ^
--name runbook name ^
--parameters AutomationAssumeRole=arn:aws:iam::123456789012:role/
RunbookAssumeRole ^
--automation-target-parameter-name target parameter ^
--schedule "cron or rate expression"
```

PowerShell

```
$Targets = New-Object Amazon.SimpleSystemsManagement.Model.Target
$Targets.Key = "ResourceGroup"
$Targets.Values = "resource group name"

New-SSMAssociation `
-AssociationName "association name" `
-Target $Targets `
-Name "runbook name" `
-Parameters @{
"AutomationAssumeRole"="arn:aws:iam::123456789012:role/RunbookAssumeRole"} `
-AutomationTargetParameterName "target parameter" `
-ScheduleExpression "cron or rate expression"
```

将多个账户和区域设置为目标

Linux & macOS

```
aws ssm create-association \
--association-name association name \
--targets Key=ResourceGroup,Values=resource group name \
--name runbook name \
--parameters AutomationAssumeRole=arn:aws:iam::123456789012:role/
RunbookAssumeRole \
--automation-target-parameter-name target parameter \
--schedule "cron or rate expression" \
--target-locations
 Accounts=111122223333,444455556666,444455556666,Regions=region,region
```

Windows

```
aws ssm create-association ^
--association-name association name ^
--targets Key=ResourceGroup,Values=resource group name ^
--name runbook name ^
--parameters AutomationAssumeRole=arn:aws:iam::123456789012:role/
RunbookAssumeRole ^
--automation-target-parameter-name target parameter ^
--schedule "cron or rate expression" ^
--target-locations
 Accounts=111122223333,444455556666,444455556666,Regions=region,region
```

PowerShell

```
$Targets = New-Object Amazon.SimpleSystemsManagement.Model.Target
$Targets.Key = "ResourceGroup"
$Targets.Values = "resource group name"

New-SSMAssociation `
-AssociationName "association name" `
-Target $Targets `
-Name "runbook name" `
-Parameters @{
"AutomationAssumeRole"="arn:aws:iam::123456789012:role/RunbookAssumeRole"} `
-AutomationTargetParameterName "target parameter" `
```

```
-ScheduleExpression "cron or rate expression" `
-TargetLocations @{
    "Accounts"=["111122223333,444455556666,444455556666"],
    "Regions"=["region,region"]
```

该命令返回新关联的详细信息，类似于以下内容。

Linux & macOS

```
{
"AssociationDescription": {
    "ScheduleExpression": "cron(0 7 ? * MON *)",
    "Name": "AWS-StartEC2Instance",
    "Parameters": {
        "AutomationAssumeRole": [
            "arn:aws:iam::123456789012:role/RunbookAssumeRole"
        ]
    },
    "Overview": {
        "Status": "Pending",
        "DetailedStatus": "Creating"
    },
    "AssociationId": "1450b4b7-bea2-4e4b-b340-01234EXAMPLE",
    "DocumentVersion": "$DEFAULT",
    "AutomationTargetParameterName": "InstanceId",
    "LastUpdateAssociationDate": 1564686638.498,
    "Date": 1564686638.498,
    "AssociationVersion": "1",
    "AssociationName": "CLI",
    "Targets": [
        {
            "Values": [
                "DEV"
            ],
            "Key": "tag:ENV"
        }
    ]
}
}
```

Windows

```
{
"AssociationDescription": {
    "ScheduleExpression": "cron(0 7 ? * MON *)",
    "Name": "AWS-StartEC2Instance",
    "Parameters": {
        "AutomationAssumeRole": [
            "arn:aws:iam::123456789012:role/RunbookAssumeRole"
        ]
    },
    "Overview": {
        "Status": "Pending",
        "DetailedStatus": "Creating"
    },
    "AssociationId": "1450b4b7-bea2-4e4b-b340-01234EXAMPLE",
    "DocumentVersion": "$DEFAULT",
    "AutomationTargetParameterName": "InstanceId",
    "LastUpdateAssociationDate": 1564686638.498,
    "Date": 1564686638.498,
    "AssociationVersion": "1",
    "AssociationName": "CLI",
    "Targets": [
        {
            "Values": [
                "DEV"
            ],
            "Key": "tag:ENV"
        }
    ]
}
}
```

PowerShell

```
Name                 : AWS-StartEC2Instance
InstanceId           :
Date                 : 8/1/2019 7:31:38 PM
Status.Name          :
Status.Date          :
Status.Message       :
Status.AdditionalInfo :
```

> **Note**
>
> 如果使用标签在一个或多个目标实例上创建关联，然后从某一实例中删除标签，则该实例将不再运行该关联。该实例不再与 State Manager 文档关联。

对 State Manager 关联运行的自动化进行故障排除

Systems Manager 自动化强制要求每个账户、每个区域最多有 100 个并发自动化和 1000 个排队自动化。如果使用运行手册的 State Manager 关联显示已失败状态和详细状态 AutomationExecutionLimitExceeded，则表明您的自动化可能已达到了限制。因此，Systems Manager 会限制自动化。要解决此问题，请执行以下操作：

- 为关联使用不同的 Rate 或 Cron 表达式。例如，如果关联计划为每 30 分钟运行一次，则更改表达式，使其每小时或每两小时运行一次。
- 删除状态为待处理的现有自动化。通过删除这些自动化，将清除当前队列。

使用维护窗口计划自动化

您可通过将运行手册配置为维护时段的注册任务来启动自动化。通过将运行手册注册为注册任务，维护时段将在计划维护期间内运行自动化。

例如，假设您创建了一个名为 `CreateAMI` 的运行手册，该运行手册创建了一个注册为维护时段目标的 Amazon Machine Image (AMI) 实例。要指定 `CreateAMI` 运行手册（和对应的工作流程）作为维护时段内的注册任务，您首先要创建一个维护时段并注册目标。然后，使用以下过程将 `CreateAMI` 文档指定为维护时段内的注册任务。当维护时段在计划期间内启动时，系统将运行自动化并创建注册目标的 AMI。

有关创建自动化运行手册的信息，请参阅 创建您自己的运行手册。Automation 是 AWS Systems Manager 中的一项工具。

可以使用以下过程通过 AWS Systems Manager 控制台、AWS Command Line Interface (AWS CLI) 或 AWS Tools for Windows PowerShell 将自动化配置为维护时段的注册任务。

在维护时段中注册自动化任务（控制台）

以下过程介绍了如何使用 Systems Manager 控制台将自动化配置为维护时段的注册任务。

开始前的准备工作

在完成以下过程之前，您必须创建维护时段并注册至少一个目标。有关更多信息，请参阅以下流程：

- 使用控制台创建维护时段.
- 使用控制台为维护时段分配目标

将自动化配置为维护时段的注册任务

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在左侧导航窗格中，选择 Maintenance Windows，然后选择您要向其注册自动化任务的维护时段。
3. 选择操作。然后选择注册自动化任务以使用运行手册在目标上运行您选择的自动化。
4. 对于名称，输入任务的名称。
5. 对于说明，输入说明。
6. 对于文)，选择定义要运行的任务的运行手册。
7. 对于文档版本，选择要使用的运行手册版本。
8. 对于任务优先级，选择一个优先级。1 是最高优先级。维护时段中的任务按优先级顺序计划，具有相同优先级的任务则并行计划。
9. 在目标部分，如果您选择的运行手册在资源上运行任务，则通过手动指定标签或选择实例来标识您要对其运行此自动化的目标。

   > **Note**
   >
   > 如果要通过输入参数而不是目标传递资源，则无需指定维护时段目标。
   > 在许多情况下，您无需显式指定自动化任务的目标。例如，假设您要创建 自动化 类型的任务，以使用 `AWS-UpdateLinuxAmi` 运行手册为 Linux 更新 Amazon Machine Image (AMI)。在该任务运行时，已使用最新可用的 Linux 分发版本的程序包和 Amazon 软件更新了 AMI。从 AMI 创建的新实例已经安装了这些更新。由于在运行手册的输入参数中指定了要进行更新的 AMI 的 ID，无需在维护时段任务中再次指定目标。

   有关不需要目标的维护时段任务的信息，请参阅 the section called “注册不含目标的维护时段任务”。
10. （可选）对于速率控制：

> **Note**
>
> 如果您正在运行的任务没有指定目标，则不需要指定速率控制。

- 对于并发，指定要同时对其运行自动化的目标的数量或百分比。

  如果通过选择键值对选择了目标，但不确定有多少个目标使用所选标签，则可以通过指定百分比来限制可同时运行的自动化的数量。

  当维护时段运行时，每个目标将启动一个新的自动化。每个 AWS 账户 限制 100 个并发自动化。如果您指定的并发速率大于 100，会有 100 个以上的并发执行自动添加到执行队列。有关信息，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service quotas。
- 对于错误阈值，指定此自动化在一定数量或百分比的目标上失败后何时停止在其他目标上运行这些工作流程。例如，如果您指定三个错误，在收到第四个错误时，Systems Manager 会停止运行自动化。仍在处理自动化的目标也可能发送错误。

11. 在输入参数部分中，为运行手册指定参数。对于运行手册，系统将自动填充某些值。您可以保留或更换这些值。

> **Important**
>
> 对于运行手册，您可以选择性地指定自动化担任角色。如果您不为此参数指定角色，那么自动化将担任您在步骤 11 中选择的维护时段服务角色。因此，您必须确保所选择的维护时段服务角色具有适当的 AWS Identity and Access Management (IAM) 权限来执行在运行手册中定义的操作。
> 例如，Systems Manager 的服务相关角色没有 IAM 权限 `ec2:CreateSnapshot`，而这是运行运行手册 `AWS-CopySnapshot` 所必需的。在这种情况下，您必须使用自定义的维护时段服务角色或指定具有 `ec2:CreateSnapshot` 权限的 Automation 担任角色。有关信息，请参阅设置自动化。

12. 在 IAM service role（IAM 服务角色）区域中，选择一个角色以向 Systems Manager 提供启动自动化的权限。

    要为维护时段任务创建自定义服务角色，请参阅设置 Maintenance Windows。

13. 选择注册自动化任务。

在维护时段中注册自动化任务（命令行）

以下过程介绍了如何使用 AWS CLI（在 Linux 或 Windows 上）或 AWS Tools for PowerShell 将自动化配置为维护时段的注册任务。

开始前的准备工作

在完成以下过程之前，您必须创建维护时段并注册至少一个目标。有关更多信息，请参阅以下流程：

- 步骤 1：使用 AWS CLI 创建维护时段.
- 步骤 2：使用 AWS CLI 将目标节点注册到维护时段

将自动化配置为维护时段的注册任务

1. 安装并配置 AWS CLI 或 AWS Tools for PowerShell（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本以及安装 AWS Tools for PowerShell。
2. 创建一个命令以将自动化配置为维护时段的注册任务。将每个*#######*替换为您自己的信息。

   Linux & macOS

```
aws ssm register-task-with-maintenance-window \
--window-id window ID \
--name task name \
--task-arn runbook name \
--targets Key=targets,Values=value \
--service-role-arn IAM role arn \
--task-type AUTOMATION \
--task-invocation-parameters task parameters \
--priority task priority \
--max-concurrency 10% \
--max-errors 5
```

> **Note**
>
> 如果使用 AWS CLI 将自动化配置为注册任务，请使用 --Task-Invocation-Parameters 参数指定在任务运行时为其传递的参数。不要使用 --Task-Parameters 参数。--Task-Parameters 参数是一个旧参数。
> 对于没有指定目标的维护时段任务，您无法为 --max-errors 和 --max-concurrency 提供值。作为替代方式，系统会插入一个占位符值 1，该值可能会在响

应诸如 describe-maintenance-window-tasks 和 get-maintenance-window-task 等命令时发出报告。这些值不影响任务的运行，可以忽略。
有关不需要目标的维护时段任务的信息，请参阅 注册不含目标的维护时段任务。

Windows

```
aws ssm register-task-with-maintenance-window ^
--window-id window ID ^
--name task name ^
--task-arn runbook name ^
--targets Key=targets,Values=value ^
--service-role-arn IAM role arn ^
--task-type AUTOMATION ^
--task-invocation-parameters task parameters ^
--priority task priority ^
--max-concurrency 10% ^
--max-errors 5
```

Note

如果使用 AWS CLI 将自动化配置为注册任务，请使用 `--task-invocation-parameters` 参数指定在任务运行时为其传递的参数。不要使用 `--task-parameters` 参数。`--task-parameters` 参数是一个旧参数。
对于没有指定目标的维护时段任务，您无法为 `--max-errors` 和 `--max-concurrency` 提供值。作为替代方式，系统会插入一个占位符值 1，该值可能会在响应诸如 describe-maintenance-window-tasks 和 get-maintenance-window-task 等命令时发出报告。这些值不影响任务的运行，可以忽略。
有关不需要目标的维护时段任务的信息，请参阅 注册不含目标的维护时段任务。

PowerShell

```
Register-SSMTaskWithMaintenanceWindow `
-WindowId window ID `
-Name "task name" `
-TaskArn "runbook name" `
-Target @{ Key="targets";Values="value" } `
-ServiceRoleArn "IAM role arn" `
```

```
-TaskType "AUTOMATION" `
-Automation_Parameter @{ "task parameter"="task parameter value"} `
-Priority task priority `
-MaxConcurrency 10% `
-MaxError 5
```

> **Note**
>
> 如果使用 AWS Tools for PowerShell 将自动化配置为注册任务，请使用 -Automation_Parameter 参数指定在任务运行时为其传递的参数。不要使用 -TaskParameters 参数。-TaskParameters 参数是一个旧参数。
> 对于没有指定目标的维护时段任务，您无法为 -MaxError 和 -MaxConcurrency 提供值。作为替代方式，系统会插入一个占位符值 1，该值可能会在响应诸如 Get-SSMMaintenanceWindowTaskList 和 Get-SSMMaintenanceWindowTask 等命令时发出报告。这些值不影响任务的运行，可以忽略。
> 有关不需要目标的维护时段任务的信息，请参阅 注册不含目标的维护时段任务。

以下示例将自动化配置为具有优先级 1 的维护时段注册任务。它还演示了省略 --targets、--max-errors 和 --max-concurrency 选项以执行无目标维护时段任务。自动化使用 AWS-StartEC2Instance 文档和指定的自动化担任角色启动在维护时段中注册为目标的 EC2 实例。在任何给定时间，维护时段最多在 5 个实例上同时运行自动化。此外，如果错误计数超过 1，注册任务将以特定执行间隔在更多实例上停止运行。

Linux & macOS

```
aws ssm register-task-with-maintenance-window \
--window-id mw-0c50858d01EXAMPLE \
--name StartEC2Instances \
--task-arn AWS-StartEC2Instance \
--service-role-arn arn:aws:iam::123456789012:role/MaintenanceWindowRole \
--task-type AUTOMATION \
--task-invocation-parameters "{\"Automation\":{\"Parameters\":{\"InstanceId\":
[\"{{TARGET_ID}}\"],\"AutomationAssumeRole\":[\"arn:aws:iam::123456789012:role/
AutomationAssumeRole\"]}}}" \
--priority 1
```

Windows

```
aws ssm register-task-with-maintenance-window ^
--window-id mw-0c50858d01EXAMPLE ^
--name StartEC2Instances ^
--task-arn AWS-StartEC2Instance ^
--service-role-arn arn:aws:iam::123456789012:role/MaintenanceWindowRole ^
--task-type AUTOMATION ^
--task-invocation-parameters "{\"Automation\":{\"Parameters\":{\"InstanceId\":
[\"{{TARGET_ID}}\"],\"AutomationAssumeRole\":[\"arn:aws:iam::123456789012:role/
AutomationAssumeRole\"]}}}" ^
--priority 1
```

PowerShell

```
Register-SSMTaskWithMaintenanceWindow `
-WindowId mw-0c50858d01EXAMPLE `
-Name "StartEC2" `
-TaskArn "AWS-StartEC2Instance" `
-ServiceRoleArn "arn:aws:iam::123456789012:role/MaintenanceWindowRole" `
-TaskType "AUTOMATION" `
-Automation_Parameter
 @{ "InstanceId"="{{TARGET_ID}}";"AutomationAssumeRole"="arn:aws:iam::123456789012:role/
AutomationAssumeRole" } `
-Priority 1
```

该命令返回新的注册任务的详细信息，类似于以下内容。

Linux & macOS

```
{
"WindowTaskId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE"
}
```

Windows

```
{
"WindowTaskId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE"
}
```

PowerShell

```
4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE
```

3. 要查看注册的任务，请运行以下命令。将 *maintenance windows ID* 替换为您自己的信息。

Linux & macOS

```
aws ssm describe-maintenance-window-tasks \
--window-id maintenance window ID
```

Windows

```
aws ssm describe-maintenance-window-tasks ^
--window-id maintenance window ID
```

PowerShell

```
Get-SSMMaintenanceWindowTaskList `
-WindowId maintenance window ID
```

系统将返回类似于以下内容的信息。

Linux & macOS

```
{
"Tasks": [
    {
        "ServiceRoleArn": "arn:aws:iam::123456789012:role/
MaintenanceWindowRole",
        "MaxErrors": "1",
        "TaskArn": "AWS-StartEC2Instance",
        "MaxConcurrency": "1",
        "WindowTaskId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE",
        "TaskParameters": {},
        "Priority": 1,
        "WindowId": "mw-0c50858d01EXAMPLE",
        "Type": "AUTOMATION",
        "Targets": [
        ],
```

```
        "Name": "StartEC2"
    }
]
}
```

Windows

```
{
"Tasks": [
    {
        "ServiceRoleArn": "arn:aws:iam::123456789012:role/
MaintenanceWindowRole",
        "MaxErrors": "1",
        "TaskArn": "AWS-StartEC2Instance",
        "MaxConcurrency": "1",
        "WindowTaskId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE",
        "TaskParameters": {},
        "Priority": 1,
        "WindowId": "mw-0c50858d01EXAMPLE",
        "Type": "AUTOMATION",
        "Targets": [
        ],
        "Name": "StartEC2"
    }
]
}
```

PowerShell

```
Description    :
LoggingInfo    :
MaxConcurrency : 5
MaxErrors      : 1
Name           : StartEC2
Priority       : 1
ServiceRoleArn : arn:aws:iam::123456789012:role/MaintenanceWindowRole
Targets        : {}
TaskArn        : AWS-StartEC2Instance
TaskParameters : {}
Type           : AUTOMATION
WindowId       : mw-0c50858d01EXAMPLE
WindowTaskId   : 4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE
```

Systems Manager 自动化操作参考

此参考介绍可在自动化运行手册中指定的自动化操作。Automation 是 AWS Systems Manager 中的一项工具。这些操作不能用于其他类型的 Systems Manager (SSM) 文档。有关其他 SSM 文档类型插件的信息，请参阅 命令文档插件参考。

Systems Manager 自动化会运行自动化运行手册中定义的步骤。每个步骤都与特定操作关联。此操作确定本步的输入、行为和输出。在运行手册的 `mainSteps` 部分定义步骤。

您无需指定操作或步骤的输出。由与本步关联的操作预先确定输出。当您在运行手册中指定步骤输入时，可以引用前面某步中的一个或多个输出。例如，您可以使 `aws:runInstances` 的输出可用于后续的 `aws:runCommand` 操作。还可以在运行手册的 `Output` 部分引用前面步骤中的输出。

> **Important**
>
> 如果您运行使用 AWS Identity and Access Management (IAM) 服务角色调用其他服务的自动化工作流程，请注意必须使用权限将该服务角色配置为调用这些服务。该要求适用于所有 AWS 自动化运行手册（`AWS-*` 运行手册），例如 `AWS-ConfigureS3BucketLogging`、`AWS-CreateDynamoDBBackup` 和 `AWS-RestartEC2Instance` 运行手册等。对于您创建的任何自定义自动化运行手册，如果这些文档使用调用其他服务的操作来调用其他 AWS 服务，则此要求同样适用。例如，如果使用 `aws:executeAwsApi`、`aws:createStack` 或 `aws:copyImage` 操作，则您必须配置具有权限的服务角色来调用这些服务。您可以将 IAM 内联策略添加到该角色，从而向其他 AWS 服务授予权限。有关更多信息，请参阅 （可选）添加自动化内联策略或客户管理型策略来调用其他 AWS 服务。

主题

- 所有操作共享的属性
- aws:approve – 暂停自动化以进行手动批准
- aws:assertAwsResourceProperty - 断言 AWS 资源状态或事件状态
- aws:branch – 运行条件自动化步骤
- aws:changeInstanceState – 更改或声明实例状态
- aws:copyImage – 复制或加密Amazon Machine Image
- aws:createImage – 创建亚马逊机器映像
- aws:createStack – 创建 AWS CloudFormation 堆栈。

- aws:createTags - 为 AWS 资源创建标签
- aws:deleteImage – 删除亚马逊机器映像
- aws:deleteStack - 删除 AWS CloudFormation 堆栈。
- aws:executeAutomation - 运行另一个自动化
- aws:executeAwsApi - 调用并运行 AWS API 操作
- aws:executeScript - 运行脚本
- aws:executeStateMachine - 运行 AWS Step Functions 状态机。
- aws:invokeWebhook – 调用 Automation Webhook 集成
- aws:invokeLambdaFunction – 调用 AWS Lambda 函数
- aws:loop – 迭代自动化中的步骤
- aws:pause - 暂停自动化
- aws:runCommand - 在托管实例上运行命令
- aws:runInstances – 启动 Amazon EC2 实例
- aws:sleep - 延迟自动化
- aws:updateVariable – 更新运行手册变量的值
- aws:waitForAwsResourceProperty - 等待 AWS 资源属性
- 自动化系统变量

所有操作共享的属性

通用属性是位于所有操作中的参数或选项。一些选项定义了步骤的行为，例如，等待步骤完成的时间以及在步骤失败时采取的措施。以下属性是所有操作的通用属性。

description

您提供的描述运行手册或步骤目的的信息。

类型：字符串

必需：否

name

在运行手册的所有步骤名称中必须唯一的标识符。

类型：字符串

允许的模式：[a-zA-Z0-9_]+$

必需：是

action

步骤要运行的操作的名称。aws:runCommand - 在托管实例上运行命令 是可在此处指定的操作的示例。本文档提供有关所有可用操作的详细信息。

类型：字符串

必需：是

maxAttempts

在发生故障的情况下应重试步骤的次数。如果值大于 1，则直到所有重试尝试失败后，才会将此步骤视为失败。默认值是 1。

类型：整数

必需：否

timeoutSeconds

步骤的超时值。如果超时并且 maxAttempts 的值大于 1，则本步未考虑超时，直至已尝试所有重试。

类型：整数

必需：否

onFailure

指示自动化在失败时是应停止、继续还是转到其他步骤。该选项的默认值为 abort。

类型：字符串

有效值：Abort | Continue | step:*step_name*

必需：否

onCancel

指示在用户取消自动化时，自动化应该转到哪个步骤。自动化将最多运行两分钟的取消工作流程。

类型：字符串

有效值：Abort | step:*step_name*

必需：否

onCancel 属性不支持移动到以下操作：

- aws:approve
- aws:copyImage
- aws:createImage
- aws:createStack
- aws:createTags
- aws:loop
- aws:pause
- aws:runInstances
- aws:sleep

isEnd

此选项在特定步骤结束时停止自动化。如果步骤失败或成功，自动化执行将停止。默认值为 False。

类型：布尔值

有效值：true | false

必需：否

nextStep

指定在成功完成自动化中的一个步骤后，接下来处理哪个步骤。

类型：字符串

必需：否

isCritical

将一个步骤指定为成功完成自动化的关键步骤。如果具有此分派的步骤失败，自动化会将自动化的最终状态报告为失败。仅当您在步骤中明确定义此属性时，才会计算该属性。如果 onFailure 属性在某个步骤中设定为 Continue，则此值默认为 False。否则，该选项的默认值为 True。

类型：布尔值

有效值：true | false

必需：否

inputs

特定于操作的属性。

类型：映射

必需：是

示例

```
---
description: "Custom Automation Example"
schemaVersion: '0.3'
assumeRole: "{{ AutomationAssumeRole }}"
parameters:
  AutomationAssumeRole:
    type: String
    description: "(Required) The ARN of the role that allows Automation to perform
      the actions on your behalf. If no role is specified, Systems Manager Automation
      uses your IAM permissions to run this runbook."
    default: ''
  InstanceId:
      type: String
      description: "(Required) The Instance Id whose root EBS volume you want to
 restore the latest Snapshot."
      default: ''
mainSteps:
- name: getInstanceDetails
  action: aws:executeAwsApi
  onFailure: Abort
  inputs:
    Service: ec2
    Api: DescribeInstances
    InstanceIds:
    - "{{ InstanceId }}"
  outputs:
    - Name: availabilityZone
      Selector: "$.Reservations[0].Instances[0].Placement.AvailabilityZone"
      Type: String
    - Name: rootDeviceName
      Selector: "$.Reservations[0].Instances[0].RootDeviceName"
      Type: String
  nextStep: getRootVolumeId
```

```
- name: getRootVolumeId
  action: aws:executeAwsApi
  maxAttempts: 3
  onFailure: Abort
  inputs:
    Service: ec2
    Api: DescribeVolumes
    Filters:
    -  Name: attachment.device
       Values: ["{{ getInstanceDetails.rootDeviceName }}"]
    -  Name: attachment.instance-id
       Values: ["{{ InstanceId }}"]
  outputs:
    - Name: rootVolumeId
      Selector: "$.Volumes[0].VolumeId"
      Type: String
  nextStep: getSnapshotsByStartTime
- name: getSnapshotsByStartTime
  action: aws:executeScript
  timeoutSeconds: 45
  onFailure: Abort
  inputs:
    Runtime: python3.8
    Handler: getSnapshotsByStartTime
    InputPayload:
      rootVolumeId : "{{ getRootVolumeId.rootVolumeId }}"
    Script: |-
      def getSnapshotsByStartTime(events,context):
        import boto3

        #Initialize client
        ec2 = boto3.client('ec2')
        rootVolumeId = events['rootVolumeId']
        snapshotsQuery = ec2.describe_snapshots(
          Filters=[
            {
              "Name": "volume-id",
              "Values": [rootVolumeId]
            }
          ]
        )
        if not snapshotsQuery['Snapshots']:
          noSnapshotFoundString = "NoSnapshotFound"
          return { 'noSnapshotFound' : noSnapshotFoundString }
```

```
        else:
          jsonSnapshots = snapshotsQuery['Snapshots']
          sortedSnapshots = sorted(jsonSnapshots, key=lambda k: k['StartTime'],
 reverse=True)
          latestSortedSnapshotId = sortedSnapshots[0]['SnapshotId']
          return { 'latestSnapshotId' : latestSortedSnapshotId }
  outputs:
  - Name: Payload
    Selector: $.Payload
    Type: StringMap
  - Name: latestSnapshotId
    Selector: $.Payload.latestSnapshotId
    Type: String
  - Name: noSnapshotFound
    Selector: $.Payload.noSnapshotFound
    Type: String
  nextStep: branchFromResults
- name: branchFromResults
  action: aws:branch
  onFailure: Abort
  onCancel: step:startInstance
  inputs:
    Choices:
    - NextStep: createNewRootVolumeFromSnapshot
      Not:
        Variable: "{{ getSnapshotsByStartTime.noSnapshotFound }}"
        StringEquals: "NoSnapshotFound"
  isEnd: true
- name: createNewRootVolumeFromSnapshot
  action: aws:executeAwsApi
  onFailure: Abort
  inputs:
    Service: ec2
    Api: CreateVolume
    AvailabilityZone: "{{ getInstanceDetails.availabilityZone }}"
    SnapshotId: "{{ getSnapshotsByStartTime.latestSnapshotId }}"
  outputs:
    - Name: newRootVolumeId
      Selector: "$.VolumeId"
      Type: String
  nextStep: stopInstance
- name: stopInstance
  action: aws:executeAwsApi
  onFailure: Abort
```

```
  inputs:
    Service: ec2
    Api: StopInstances
    InstanceIds:
    - "{{ InstanceId }}"
  nextStep: verifyVolumeAvailability
- name: verifyVolumeAvailability
  action: aws:waitForAwsResourceProperty
  timeoutSeconds: 120
  inputs:
    Service: ec2
    Api: DescribeVolumes
    VolumeIds:
    - "{{ createNewRootVolumeFromSnapshot.newRootVolumeId }}"
    PropertySelector: "$.Volumes[0].State"
    DesiredValues:
    - "available"
  nextStep: verifyInstanceStopped
- name: verifyInstanceStopped
  action: aws:waitForAwsResourceProperty
  timeoutSeconds: 120
  inputs:
    Service: ec2
    Api: DescribeInstances
    InstanceIds:
    - "{{ InstanceId }}"
    PropertySelector: "$.Reservations[0].Instances[0].State.Name"
    DesiredValues:
    - "stopped"
  nextStep: detachRootVolume
- name: detachRootVolume
  action: aws:executeAwsApi
  onFailure: Abort
  isCritical: true
  inputs:
    Service: ec2
    Api: DetachVolume
    VolumeId: "{{ getRootVolumeId.rootVolumeId }}"
  nextStep: verifyRootVolumeDetached
- name: verifyRootVolumeDetached
  action: aws:waitForAwsResourceProperty
  timeoutSeconds: 30
  inputs:
    Service: ec2
```

```
    Api: DescribeVolumes
    VolumeIds:
    - "{{ getRootVolumeId.rootVolumeId }}"
    PropertySelector: "$.Volumes[0].State"
    DesiredValues:
    - "available"
  nextStep: attachNewRootVolume
- name: attachNewRootVolume
  action: aws:executeAwsApi
  onFailure: Abort
  inputs:
    Service: ec2
    Api: AttachVolume
    Device: "{{ getInstanceDetails.rootDeviceName }}"
    InstanceId: "{{ InstanceId }}"
    VolumeId: "{{ createNewRootVolumeFromSnapshot.newRootVolumeId }}"
  nextStep: verifyNewRootVolumeAttached
- name: verifyNewRootVolumeAttached
  action: aws:waitForAwsResourceProperty
  timeoutSeconds: 30
  inputs:
    Service: ec2
    Api: DescribeVolumes
    VolumeIds:
    - "{{ createNewRootVolumeFromSnapshot.newRootVolumeId }}"
    PropertySelector: "$.Volumes[0].Attachments[0].State"
    DesiredValues:
    - "attached"
  nextStep: startInstance
- name: startInstance
  action: aws:executeAwsApi
  onFailure: Abort
  inputs:
    Service: ec2
    Api: StartInstances
    InstanceIds:
    - "{{ InstanceId }}"
```

`aws:approve` – 暂停自动化以进行手动批准

临时暂停自动化，直至指定委托人批准或拒绝操作。在达到所需批准数后，自动化执行将恢复。您可以将批准步骤插入到运行手册的 `mainSteps` 部分。

> **Note**
>
> 此操作并不支持多账户和区域自动化。此操作的默认超时时间为 7 天（604800 秒），最长时间为 30 天（2592000 秒）。您可以通过指定 aws:approve 步骤的 timeoutSeconds 参数来限制或延长超时。

在以下示例中，aws:approve 操作临时暂停自动化，直至一个审批者接受或拒绝自动化。批准后，此自动化将运行简单的 PowerShell 命令。

YAML

```
---
description: RunInstancesDemo1
schemaVersion: '0.3'
assumeRole: "{{ assumeRole }}"
parameters:
  assumeRole:
    type: String
  message:
    type: String
mainSteps:
- name: approve
  action: aws:approve
  timeoutSeconds: 1000
  onFailure: Abort
  inputs:
    NotificationArn: arn:aws:sns:us-east-2:12345678901:AutomationApproval
    Message: "{{ message }}"
    MinRequiredApprovals: 1
    Approvers:
    - arn:aws:iam::12345678901:user/AWS-User-1
- name: run
  action: aws:runCommand
  inputs:
    InstanceIds:
    - i-1a2b3c4d5e6f7g
    DocumentName: AWS-RunPowerShellScript
    Parameters:
      commands:
      - date
```

JSON

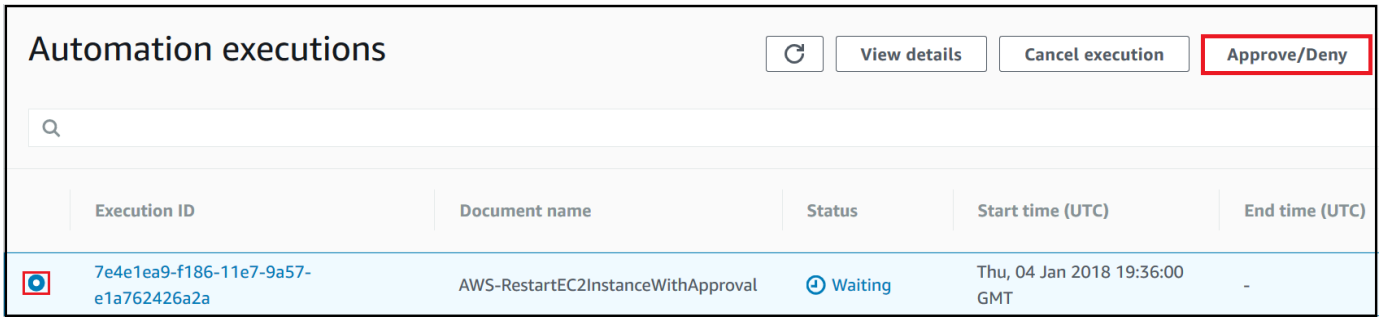
```
{
   "description":"RunInstancesDemo1",
   "schemaVersion":"0.3",
   "assumeRole":"{{ assumeRole }}",
   "parameters":{
      "assumeRole":{
         "type":"String"
      },
      "message":{
         "type":"String"
      }
   },
   "mainSteps":[
      {
         "name":"approve",
         "action":"aws:approve",
         "timeoutSeconds":1000,
         "onFailure":"Abort",
         "inputs":{
            "NotificationArn":"arn:aws:sns:us-
east-2:12345678901:AutomationApproval",
            "Message":"{{ message }}",
            "MinRequiredApprovals":1,
            "Approvers":[
               "arn:aws:iam::12345678901:user/AWS-User-1"
            ]
         }
      },
      {
         "name":"run",
         "action":"aws:runCommand",
         "inputs":{
            "InstanceIds":[
               "i-1a2b3c4d5e6f7g"
            ],
            "DocumentName":"AWS-RunPowerShellScript",
            "Parameters":{
               "commands":[
                  "date"
               ]
            }
         }
```

```
        }
    ]
}
```

您可以在控制台中批准或拒绝等待批准的自动化。

批准或拒绝等待的自动化

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 自动化。
3. 选择状态为正在等待的自动化旁边的选项。



4. 选择批准/拒绝。
5. 查看自动化的详细信息。
6. 选择批准或拒绝，键入评论（可选），然后选择提交。

输入示例

YAML

```
NotificationArn: arn:aws:sns:us-west-1:12345678901:Automation-ApprovalRequest
Message: Please approve this step of the Automation.
MinRequiredApprovals: 3
Approvers:
- IamUser1
- IamUser2
- arn:aws:iam::12345678901:user/IamUser3
- arn:aws:iam::12345678901:role/IamRole
```

JSON

```
{
   "NotificationArn":"arn:aws:sns:us-west-1:12345678901:Automation-ApprovalRequest",
   "Message":"Please approve this step of the Automation.",
   "MinRequiredApprovals":3,
   "Approvers":[
      "IamUser1",
      "IamUser2",
      "arn:aws:iam::12345678901:user/IamUser3",
      "arn:aws:iam::12345678901:role/IamRole"
   ]
}
```

NotificationArn

用于批准自动化的 Amazon Resource Name (即 ARN，属于 Amazon Simple Notification Service (Amazon SNS)) 的主题。当您在运行手册中指定 `aws:approve` 步骤时，自动化 将向此主题发送消息，以让委托人知道必须批准或拒绝自动化步骤。Amazon SNS 主题的标题必须使用前缀“自动化”。

类型：字符串

必需：否

消息

发送批准请求时要包含在 Amazon SNS 主题中的信息。最大消息长度为 4096 个字符。

类型：字符串

必需：否

MinRequiredApprovals

恢复自动化所需的最小批准数。如果您未指定值，系统将默认为 1。此参数的值必须为正数。此参数的值不能超过 `Approvers` 参数定义的审批者数。

类型：整数

必需：否

Approvers

能够批准或拒绝操作的经 AWS 身份验证的委托人的列表。最大审批者数量为 10。您可使用以下任意格式指定委托人：

- 一个用户名称
- 用户 ARN
- IAM 角色 ARN
- IAM 担任角色 ARN

类型：StringList

必需：是

EnhancedApprovals

此输入仅用于 Change Manager 模板。能够批准或拒绝操作的经过 AWS 身份验证的主体、IAM 主体的类型以及最少批准者数量的列表。以下是 示例：

```
schemaVersion: "0.3"
emergencyChange: false
autoApprovable: false
mainSteps:
    - name: ApproveAction1
    action: aws:approve
    timeoutSeconds: 604800
    inputs:
        Message: Please approve this change request
        MinRequiredApprovals: 3
        EnhancedApprovals:
        Approvers:
            - approver: John Stiles
            type: IamUser
            minRequiredApprovals: 0
            - approver: Ana Carolina Silva
            type: IamUser
            minRequiredApprovals: 0
            - approver: GroupOfThree
            type: IamGroup
            minRequiredApprovals: 0
            - approver: RoleOfTen
            type: IamRole
```

```
            minRequiredApprovals: 0
```

类型：StringList

必需：是

输出

ApprovalStatus

步骤的批准状态。状态可以为下列状态之一：已批准、已拒绝或正在等待。“正在等待”意味着自动化正在等待来自审批者的输入。

类型：字符串

ApproverDecisions

包括每位审批者的批准决定的 JSON 映射。

类型：MapList

aws:assertAwsResourceProperty - 断言 AWS 资源状态或事件状态

aws:assertAwsResourceProperty 操作可用于对特定自动化步骤的资源状态或事件状态进行断言。

有关如何使用此操作的更多示例，请参阅 其他运行手册示例。

输入

由您选择的 API 操作定义的输入。

YAML

```
action: aws:assertAwsResourceProperty
inputs:
  Service: The official namespace of the service
  Api: The API operation or method name
  API operation inputs or parameters: A value
  PropertySelector: Response object
```

```
  DesiredValues:
  - Desired property values
```

JSON

```
{
  "action": "aws:assertAwsResourceProperty",
  "inputs": {
    "Service":"The official namespace of the service",
    "Api":"The API operation or method name",
    "API operation inputs or parameters":"A value",
    "PropertySelector": "Response object",
    "DesiredValues": [
      "Desired property values"
    ]
  }
}
```

服务

包含要运行的 API 操作的 AWS 服务命名空间。例如，Systems Manager 的命名空间为 `ssm`。Amazon EC2 的命名空间为 `ec2`。您可以在《AWS CLI 命令参考》的可用服务部分查看支持的 AWS 服务命名空间列表。

类型：字符串

必需：是

API

要运行的 API 操作的名称。您可以在以下服务参考页面的左侧导航栏中选择服务来查看 API 操作（也称为方法）。在要调用的服务的客户端部分中选择一种方法。例如，下面的 Amazon RDS 方法页面中列出了 Amazon Relational Database Service (Amazon RDS) 的所有 API 操作（方法）。

类型：字符串

必需：是

API 操作输入

一个或多个 API 操作输入。您可以在以下服务参考页面的左侧导航栏中选择服务来查看可用的输入（也称为参数）。在要调用的服务的客户端部分中选择一种方法。例如，下面的 Amazon RDS 方

法页面中列出了 Amazon RDS 的所有方法。选择 describe_db_instances 方法并向下滚动以查看可用的参数，例如 DBInstanceIdentifier、Name 和 Values。使用以下格式指定多个输入。

YAML

```
inputs:
  Service: The official namespace of the service
  Api: The API operation name
  API input 1: A value
  API Input 2: A value
  API Input 3: A value
```

JSON

```
"inputs":{
      "Service":"The official namespace of the service",
      "Api":"The API operation name",
      "API input 1":"A value",
      "API Input 2":"A value",
      "API Input 3":"A value"
}
```

类型：由选择的 API 操作决定

必需：是

PropertySelector

响应对象中特定属性的 JSONPath。您可以在以下服务参考页面的左侧导航栏中选择服务来查看响应对象。在要调用的服务的客户端部分中选择一种方法。例如，下面的 Amazon RDS 方法页面中列出了 Amazon RDS 的所有方法：选择 describe_db_instances 方法，然后向下滚动到响应结构部分。DBInstances 被列为响应对象。

类型：字符串

必需：是

DesiredValues

要继续自动化的预期状态。如果指定布尔值，则必须使用大写字母，例如 True 或 False。

类型：StringList

必需：是

aws:branch – 运行条件自动化步骤

aws:branch 操作让您能够创建一个动态自动化，该自动化在一个步骤中评估不同选择，然后根据评估结果跳转到运行手册中的另一个步骤。

在为步骤指定 aws:branch 操作时，请指定自动化必须评估的 Choices。Choices 可以基于您在运行手册的 Parameters 部分中指定的值，也可以基于上一步的输出生成的动态值。自动化使用布尔表达式评估每个选择。如果第一个选择为真，则自动化跳转到为此选择指定的步骤。如果第一个选择为假，则自动化评估下一个选择。自动化继续评估每个选择，直到遇到结果为真的选择。然后，自动化跳转到为结果为真的选择指定的步骤。

如果所有选择都为假，则自动化检查该步骤是否包含 default 值。默认值定义当所有选择都为假时自动化应跳转到的步骤。如果没有为该步骤指定 default 值，则自动化处理运行手册中的下一个步骤。

aws:branch 操作通过组合使用 And、Not 和 Or 运算符来支持复杂的选择评估。有关如何使用 aws:branch 的更多信息，包括使用不同运算符的示例运行手册和示例，请参阅 在运行手册中使用条件语句。

输入

在步骤中指定一个或多个 Choices。Choices 可以基于您在运行手册的 Parameters 部分中指定的值，也可以基于上一步的输出生成的动态值。下面是一个评估参数的 YAML 示例。

```
mainSteps:
- name: chooseOS
  action: aws:branch
  inputs:
    Choices:
    - NextStep: runWindowsCommand
      Variable: "{{Name of a parameter defined in the Parameters section. For example:
 OS_name}}"
      StringEquals: windows
    - NextStep: runLinuxCommand
      Variable: "{{Name of a parameter defined in the Parameters section. For example:
 OS_name}}"
      StringEquals: linux
    Default:
      sleep3
```

下面是一个评估上一步输出的 YAML 示例。

```
mainSteps:
- name: chooseOS
  action: aws:branch
  inputs:
    Choices:
    - NextStep: runPowerShellCommand
      Variable: "{{Name of a response object. For example: GetInstance.platform}}"
      StringEquals: Windows
    - NextStep: runShellCommand
      Variable: "{{Name of a response object. For example: GetInstance.platform}}"
      StringEquals: Linux
    Default:
      sleep3
```

Choices

自动化 在确定下一个要处理的步骤时应评估的一个或多个表达式。通过使用布尔表达式对选择进行评估。每个选择都必须定义以下选项：

- NextStep：当指定的选择为真时，要处理的运行手册中的下一个步骤。
- 变量：指定在运行手册的 Parameters 部分中定义的参数名称。或指定来自运行手册中上一步的输出对象。有关为 aws:branch 创建变量的更多信息，请参阅 关于创建输出变量。
- 运算：用于评估选择的标准。aws:branch 操作支持以下运算：

  字符串运算

  - 字符串等于
  - EqualsIgnoreCase
  - StartsWith
  - EndsWith
  - 包含

  数值运算

  - NumericEquals
  - NumericGreater
  - NumericLesser
  - NumericGreaterOrEquals

- NumericLesser
- NumericLesserOrEquals

布尔运算

- BooleanEquals

> **Important**
>
> 创建运行手册时，系统将验证运行手册中的每个操作。在尝试创建运行手册时，如果某个操作不受支持，系统会返回错误。

默认

当所有 `Choices` 都为假时，自动化应跳转到的步骤的名称。

类型：字符串

必需：否

> **Note**
>
> `aws:branch` 操作支持 `And`、`Or` 和 `Not` 运算符。有关使用运算符的 `aws:branch` 的示例，请参阅 在运行手册中使用条件语句。

`aws:changeInstanceState` – 更改或声明实例状态

更改或断言实例的状态。

此操作可在断言模式下使用（不要运行 API 来更改状态，而应验证实例是否处于预期状态。）要使用断言模式，请将 `CheckStateOnly` 参数设置为 True。当在 Windows 上运行 Sysprep 命令时，此模式很有用。该命令是一种可在后台长时间运行的异步命令。您可以确保在创建 Amazon Machine Image (AMI) 之前停止实例。

> **Note**
>
> 此操作的默认超时值为 3600 秒（1 小时）。您可以通过指定 `aws:changeInstanceState` 步骤的 `timeoutSeconds` 参数来限制或延长超时。

输入

YAML

```
name: stopMyInstance
action: aws:changeInstanceState
maxAttempts: 3
timeoutSeconds: 3600
onFailure: Abort
inputs:
  InstanceIds:
  - i-1234567890abcdef0
  CheckStateOnly: true
  DesiredState: stopped
```

JSON

```
{
    "name":"stopMyInstance",
    "action": "aws:changeInstanceState",
    "maxAttempts": 3,
    "timeoutSeconds": 3600,
    "onFailure": "Abort",
    "inputs": {
        "InstanceIds": ["i-1234567890abcdef0"],
        "CheckStateOnly": true,
        "DesiredState": "stopped"
    }
}
```

InstanceIds

实例的 ID。

类型：StringList

必需：是

CheckStateOnly

如果为 false，请将实例状态设置为预期状态。如果为 true，请使用轮询断言预期状态。

默认：`false`

类型：布尔值

必需：否

DesiredState

预期状态。设置为 `running` 时，此操作在完成之前等待 Amazon EC2 的状态变为 `Running`、实例状态变为 `OK`、系统状态变为 `OK`。

类型：字符串

有效值：`running` |`stopped` |`terminated`

必需：是

Force

如果设置此项，则强制停止实例。则该实例没有机会来刷新文件系统缓存或文件系统元数据。如果您使用此选项，则必须执行文件系统检查和修复流程。我们不建议将该选项用于 Windows Server 的 EC2 实例。

类型：布尔值

必需：否

AdditionalInfo

预留。

类型：字符串

必需：否

输出

无

`aws:copyImage` – 复制或加密Amazon Machine Image

将 Amazon Machine Image (AMI) 从任何 AWS 区域 复制到当前区域中。此操作还可以对新的 AMI 进行加密。

输入

此操作支持大多数 `CopyImage` 参数。有关更多信息，请参阅 CopyImage。

以下示例是在首尔地区创建 AMI 的副本（SourceImageID：ami-0fe10819。SourceRegion：ap-northeast-2）。新的 AMI 将复制到您启动自动化操作的区域。将对复制的 AMI 进行加密，因为可选 Encrypted 标记将设置为 true。

YAML

```
name: createEncryptedCopy
action: aws:copyImage
maxAttempts: 3
onFailure: Abort
inputs:
  SourceImageId: ami-0fe10819
  SourceRegion: ap-northeast-2
  ImageName: Encrypted Copy of LAMP base AMI in ap-northeast-2
  Encrypted: true
```

JSON

```
{
    "name": "createEncryptedCopy",
    "action": "aws:copyImage",
    "maxAttempts": 3,
    "onFailure": "Abort",
    "inputs": {
        "SourceImageId": "ami-0fe10819",
        "SourceRegion": "ap-northeast-2",
        "ImageName": "Encrypted Copy of LAMP base AMI in ap-northeast-2",
        "Encrypted": true
    }
}
```

SourceRegion

源 AMI 当前所在的区域。

类型：字符串

必需：是

SourceImageId

要从源区域复制的 AMI ID。

类型：字符串

必需：是

ImageName

新映像的名称。

类型：字符串

必需：是

ImageDescription

目标映像的描述。

类型：字符串

必需：否

已加密

对目标 AMI 进行加密。

类型：布尔值

必需：否

KmsKeyId

在复制操作期间对映像快照进行加密时要使用的 AWS KMS key 的完整 Amazon Resource Name (ARN)。有关更多信息，请参阅 CopyImage。

类型：字符串

必需：否

ClientToken

您为确保请求幂等性而提供的唯一、区分大小写的标识符。有关更多信息，请参阅 CopyImage。

类型：字符串

必需：否

输出

ImageId

已复制映像的 ID。

ImageState

已复制映像的状态。

有效值：`available` |`pending` |`failed`

`aws:createImage` – 创建亚马逊机器映像

从正在运行、正在停止或已停止的实例创建 Amazon Machine Image（AMI），并轮询 `ImageState` 是否为 `available`。

输入

此操作支持以下 `CreateImage` 参数。有关更多信息，请参阅 CreateImage。

YAML

```
name: createMyImage
action: aws:createImage
maxAttempts: 3
onFailure: Abort
inputs:
  InstanceId: i-1234567890abcdef0
  ImageName: AMI Created on{{global:DATE_TIME}}
  NoReboot: true
  ImageDescription: My newly created AMI
```

JSON

```
{
    "name": "createMyImage",
    "action": "aws:createImage",
    "maxAttempts": 3,
    "onFailure": "Abort",
    "inputs": {
        "InstanceId": "i-1234567890abcdef0",
        "ImageName": "AMI Created on{{global:DATE_TIME}}",
```

```
        "NoReboot": true,
        "ImageDescription": "My newly created AMI"
    }
}
```

InstanceId

实例的 ID。

类型：字符串

必需：是

ImageName

映像的名称。

类型：字符串

必需：是

ImageDescription

映像的描述。

类型：字符串

必需：否

NoReboot

一种布尔文本。

默认情况下，Amazon Elastic Compute Cloud (Amazon EC2) 会尝试关闭并重新启动实例，然后再创建映像。如果不重启选项设置为 `true`，则 Amazon EC2 在创建映像前不会关闭实例。如果使用此选项，则无法保证所创建映像上的文件系统的完整性。

如果您希望在从实例创建 AMI 映像后，该实例不运行，请先使用 `aws:changeInstanceState` – 更改或声明实例状态 插件停止实例，然后在 NoReboot 选项设置为 `true` 的情况下使用此 `aws:createImage` 操作。

类型：布尔值

必需：否

BlockDeviceMappings

适用于实例的块储存设备。

类型：映射

必需：否

输出

ImageId

新建映像的 ID。

类型：字符串

ImageState

映像的当前状态。如果状态为可用，则表示映像已成功注册，并且可用于启动实例。

类型：字符串

`aws:createStack` – 创建 AWS CloudFormation 堆栈。

从模板创建 AWS CloudFormation 堆栈。

有关创建 CloudFormation 堆栈的补充信息，请参阅 AWS CloudFormationAPI 参考中的 CreateStack。

输入

YAML

```
name: makeStack
action: aws:createStack
maxAttempts: 1
onFailure: Abort
inputs:
  Capabilities:
  - CAPABILITY_IAM
  StackName: myStack
  TemplateURL: http://s3.amazonaws.com/amzn-s3-demo-bucket/myStackTemplate
```

```
  TimeoutInMinutes: 5
  Parameters:
    - ParameterKey: LambdaRoleArn
      ParameterValue: "{{LambdaAssumeRole}}"
    - ParameterKey: createdResource
      ParameterValue: createdResource-{{automation:EXECUTION_ID}}
```

JSON

```
{
    "name": "makeStack",
    "action": "aws:createStack",
    "maxAttempts": 1,
    "onFailure": "Abort",
    "inputs": {
        "Capabilities": [
            "CAPABILITY_IAM"
        ],
        "StackName": "myStack",
        "TemplateURL": "http://s3.amazonaws.com/amzn-s3-demo-bucket/
myStackTemplate",
        "TimeoutInMinutes": 5,
        "Parameters": [
          {
            "ParameterKey": "LambdaRoleArn",
            "ParameterValue": "{{LambdaAssumeRole}}"
          },
          {
            "ParameterKey": "createdResource",
            "ParameterValue": "createdResource-{{automation:EXECUTION_ID}}"
          }
    }
}
```

功能

必须在 CloudFormatio 可以创建某些堆栈之前指定的值列表。一些堆栈模板中包含的资源会影响您的 AWS 账户中的权限。对于这些堆栈，您必须通过指定此参数来明确确认它们的功能。

有效值包括 CAPABILITY_IAM、CAPABILITY_NAMED_IAM 和 CAPABILITY_AUTO_EXPAND。

CAPABILITY_IAM 和 CAPABILITY_NAMED_IAM

如果包含 IAM 资源，您可以指定任意一个功能。如果包含具有自定义名称的 IAM 资源，则必须指定 CAPABILITY_NAMED_IAM。如果您不指定此参数，则此操作会返回 InsufficientCapabilities 错误。以下资源要求您指定 CAPABILITY_IAM 或 CAPABILITY_NAMED_IAM。

- AWS::IAM::AccessKey
- AWS::IAM::Group
- AWS::IAM::InstanceProfile
- AWS::IAM::Policy
- AWS::IAM::Role
- AWS::IAM::User
- AWS::IAM::UserToGroupAddition

如果您的堆栈模板包含这些资源，我们建议您查看与之关联的所有权限并在必要时编辑其权限。

有关更多信息，请参阅确认 AWS CloudFormation 模板中的 IAM 资源。

CAPABILITY_AUTO_EXPAND

某些模板包含宏。宏对模板执行自定义处理，包括查找并替换等简单操作，以及整个模板的大幅转换。因此，用户通常会从已处理的模板创建更改集，这样他们便能在实际创建堆栈之前查看宏导致的更改。如果堆栈模板包含一个或多个宏，并且您选择直接从已处理的模板创建堆栈，而不首先查看更改集中生成的更改，则必须确认此功能。

有关更多信息，请参阅 AWS CloudFormation用户指南中的使用 AWS CloudFormation 宏对模板执行自定义处理。

类型：字符串的数组

有效值：CAPABILITY_IAM | CAPABILITY_NAMED_IAM | CAPABILITY_AUTO_EXPAND

必需：否

ClientRequestToken

该 CreateStack 请求的唯一标识符。如果将此步骤中的 maxAttempts 设置为大于 1 的值，请指定此令牌。通过指定此令牌，CloudFormation 知道您未在尝试使用相同的名称创建新堆栈。

类型：字符串

必需：否

长度限制：长度下限为 1。最大长度为 128。

模式：[a-zA-Z0-9][-a-zA-Z0-9]*

DisableRollback

如果堆栈创建失败，请设置为 `true` 以关闭堆栈回滚。

Conditional：您可以指定 `DisableRollback` 参数或 `OnFailure` 参数，但不能同时指定。

默认值：`false`

类型：布尔值

必需：否

NotificationARNs

用于发布堆栈相关事件的 Amazon Simple Notification Service (Amazon SNS) 主题 ARN。您可以使用 Amazon SNS 控制台 https://console.aws.amazon.com/sns/v3/home 查找 SNS 主题 ARN。

类型：字符串的数组

数组成员：最多 5 项。

必需：否

OnFailure

如果堆栈创建失败，确定要执行的操作。您必须指定 `DO_NOTHING`、`ROLLBACK` 或 `DELETE`。

Conditional：您可以指定 `OnFailure` 参数或 `DisableRollback` 参数，但不能同时指定。

默认值：`ROLLBACK`

类型：字符串

有效值: `DO_NOTHING | ROLLBACK | DELETE`

必需：否

参数

指定堆栈输入参数的 `Parameter` 结构列表。有关更多信息，请参阅参数数据类型。

类型：参数对象数组

必需：否

ResourceTypes

您有权用于此创建堆栈操作的模板资源类型。例如，AWS::EC2::Instance、AWS::EC2::* 或 Custom::*MyCustomInstance*。使用以下语法描述模板资源类型。

- 对于所有 AWS 资源：

```
AWS::*
```

- 对于所有自定义资源：

```
Custom::*
```

- 对于指定自定义资源：

```
Custom::logical_ID
```

- 对于特定 AWS 服务的所有资源：

```
AWS::service_name::*
```

- 对于特定的 AWS 资源：

```
AWS::service_name::resource_logical_ID
```

如果资源类型列表不包括您创建的资源，那么堆栈创建将会失败。默认情况下，CloudFormation 授予对所有资源类型的权限。IAM 将此参数用于 IAM policy 中特定于云的条件密钥。有关更多信息，请参阅使用 AWS Identity and Access Management 控制访问。

类型：字符串的数组

长度限制：最小长度为 1。最大长度为 256。

必需：否

RoleARN

CloudFormation 用于创建堆栈的 IAM 角色的 Amazon Resource Name (ARN)。CloudFormation 使用角色的凭证代表您进行调用。CloudFormation 始终将此角色用于堆栈上的所有未来操作。只要用户有权对堆栈进行操作，CloudFormation 会使用此角色，即使用户无权传递它。确保该角色授予最少的权限。

如果您不指定值，则 CloudFormation 会使用之前与堆栈关联的角色。如果角色不可用，则 CloudFormation 会使用您的用户凭证生成的一个临时会话。

类型：字符串

长度约束：最小长度为 20。最大长度为 2048。

必需：否

StackName

与堆栈关联的名称。名称在您创建堆栈的区域中必须是唯一的。

> **Note**
>
> 堆栈名称只能包含字母数字字符（区分大小写）和连字符。该名称必须以字母字符开头，且不得超过 128 个字符。

类型：字符串

必需：是

StackPolicyBody

包含堆栈策略正文的结构。有关更多信息，请参阅防止更新堆栈资源。

Conditional：您可以指定 `StackPolicyBody` 参数或 `StackPolicyURL` 参数，但不能同时指定。

类型：字符串

长度限制：长度下限为 1。长度上限为 16384。

必需：否

StackPolicyURL

包含堆栈策略的文件的位置。URL 指向的策略必须位于与堆栈处于同一区域的 S3 存储桶中。堆栈策略允许的最大文件大小为 16 KB。

Conditional：您可以指定 `StackPolicyBody` 参数或 `StackPolicyURL` 参数，但不能同时指定。

类型：字符串

长度限制：长度下限为 1。长度上限为 1350。

必需：否

Tags

与此堆栈关联的键值对。CloudFormation 还可以将这些标签传播到堆栈中创建的资源。您可以指定最多 10 个标签。

类型：标签对象数组

必需：否

TemplateBody

包含最小长度为 1 字节、最大长度为 51200 字节的模板正文的结构。有关更多信息，请参阅模板剖析。

Conditional：您可以指定 TemplateBody 参数或 TemplateURL 参数，但不能同时指定。

类型：字符串

长度限制：长度下限为 1。

必需：否

TemplateURL

包含模板正文的文件的位置。URL 必须指向一个位于 S3 存储桶中的模板。模板允许的最大大小为 460800 字节。有关更多信息，请参阅模板剖析。

Conditional：您可以指定 TemplateBody 参数或 TemplateURL 参数，但不能同时指定。

类型：字符串

长度限制：长度下限为 1。最大长度为 1024。

必需：否

TimeoutInMinutes

堆栈状态变为 CREATE_FAILED 前允许经过的时间。如果未设置 DisableRollback 或将其设置为 false，堆栈将被回滚。

类型：整数

有效范围：最小值为 1。

必需：否

输出

StackId

堆栈的唯一标识符。

类型：字符串

StackStatus

堆栈的当前状态。

类型：字符串

有效值：CREATE_IN_PROGRESS | CREATE_FAILED | CREATE_COMPLETE | ROLLBACK_IN_PROGRESS | ROLLBACK_FAILED | ROLLBACK_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | DELETE_COMPLETE | UPDATE_IN_PROGRESS | UPDATE_COMPLETE_CLEANUP_IN_PROGRESS | UPDATE_COMPLETE | UPDATE_ROLLBACK_IN_PROGRESS | UPDATE_ROLLBACK_FAILED | UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS | UPDATE_ROLLBACK_COMPLETE | REVIEW_IN_PROGRESS

必需：是

StackStatusReason

与堆栈状态相关联的成功或失败消息。

类型：字符串

必需：否

有关更多信息，请参阅 CreateStack。

安全性注意事项

您必须将以下策略分配给 IAM 自动化担任角色，才可以使用 aws:createStack 操作。有关担任角色的更多信息，请参阅 任务 1：为自动化创建服务角色。

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "sqs:*",
            "cloudformation:CreateStack",
            "cloudformation:DescribeStacks"
         ],
         "Resource":"*"
      }
   ]
}
```

`aws:createTags` - 为 AWS 资源创建标签

为 Amazon Elastic Compute Cloud (Amazon EC2) 实例或 AWS Systems Manager 托管实例创建新标签。

输入

此操作支持大多数 Amazon EC2 `CreateTags` 和 Systems Manager `AddTagsToResource` 参数。有关更多信息，请参阅 CreateTags 和 AddTagsToResource。

以下示例说明如何为 Amazon Machine Image (AMI) 和实例添加标签以作为特定部门的生产资源。

YAML

```
name: createTags
action: aws:createTags
maxAttempts: 3
onFailure: Abort
inputs:
  ResourceType: EC2
  ResourceIds:
  - ami-9a3768fa
  - i-02951acd5111a8169
  Tags:
  - Key: production
```

```
    Value: ''
  - Key: department
    Value: devops
```

JSON

```
{
    "name": "createTags",
    "action": "aws:createTags",
    "maxAttempts": 3,
    "onFailure": "Abort",
    "inputs": {
        "ResourceType": "EC2",
        "ResourceIds": [
            "ami-9a3768fa",
            "i-02951acd5111a8169"
        ],
        "Tags": [
            {
                "Key": "production",
                "Value": ""
            },
            {
                "Key": "department",
                "Value": "devops"
            }
        ]
    }
}
```

ResourceIds

要为其添加标签的资源的 ID。如果资源类型不是“EC2”，则此字段只能包含单个项目。

类型：字符串列表

必需：是

标签

要与资源关联的标签。

类型：映射列表

必需：是

ResourceType

要为其添加标签的资源的类型。如果未提供，则使用默认值“EC2”。

类型：字符串

必需：否

有效值: EC2 | ManagedInstance | MaintenanceWindow | Parameter

输出

无

aws:deleteImage – 删除亚马逊机器映像

删除指定 Amazon Machine Image (AMI) 和所有的相关快照。

输入

此操作仅支持一个参数。有关更多信息，请参阅 DeregisterImage 和 DeleteSnapshot 的相关文档。

YAML

```
name: deleteMyImage
action: aws:deleteImage
maxAttempts: 3
timeoutSeconds: 180
onFailure: Abort
inputs:
  ImageId: ami-12345678
```

JSON

```
{
    "name": "deleteMyImage",
    "action": "aws:deleteImage",
    "maxAttempts": 3,
    "timeoutSeconds": 180,
    "onFailure": "Abort",
    "inputs": {
        "ImageId": "ami-12345678"
```

```
    }
}
```

ImageId

要删除的映像的 ID。

类型：字符串

必需：是

输出

无

aws:deleteStack - 删除 AWS CloudFormation 堆栈。

删除 AWS CloudFormation 堆栈。

输入

YAML

```
name: deleteStack
action: aws:deleteStack
maxAttempts: 1
onFailure: Abort
inputs:
  StackName: "{{stackName}}"
```

JSON

```
{
   "name":"deleteStack",
   "action":"aws:deleteStack",
   "maxAttempts":1,
   "onFailure":"Abort",
   "inputs":{
      "StackName":"{{stackName}}"
   }
}
```

ClientRequestToken

此 DeleteStack 请求的唯一标识符。如果您计划重试请求以便 CloudFormation 知道您未在尝试删除同名堆栈，请指定此令牌。您可以重试 DeleteStack 请求以验证 CloudFormation 是否收到了它们。

类型：字符串

长度限制：长度下限为 1。最大长度为 128。

模式：[a-zA-Z][-a-zA-Z0-9]*

必需：否

RetainResources.member.N

此输入仅适用于处于 DELETE_FAILED 状态的堆栈。您想要保留的资源的逻辑资源 ID 的列表。在删除时，CloudFormation 删除堆栈，但不删除保留资源。

如果无法删除某个资源（例如非空 S3 存储桶），但需要删除堆栈，则保留资源会很有用。

类型：字符串数组

必需：否

RoleARN

CloudFormation 用于创建堆栈的 AWS Identity and Access Management (IAM) 角色的 Amazon Resource Name (ARN)。CloudFormation 使用角色的凭证代表您进行调用。CloudFormation 始终将此角色用于堆栈上的所有未来操作。只要用户有权对堆栈进行操作，CloudFormation 会使用此角色，即使用户无权传递它。确保该角色授予最少的权限。

如果您不指定值，则 CloudFormation 会使用之前与堆栈关联的角色。如果角色不可用，则 CloudFormation 会使用您的用户凭证生成的一个临时会话。

类型：字符串

长度约束：最小长度为 20。最大长度为 2048。

必需：否

StackName

与堆栈关联的名称或唯一堆栈 ID。

类型：字符串

必需：是

安全性注意事项

您必须将以下策略分配给 IAM 自动化担任角色，才可以使用 aws:deleteStack 操作。有关担任角色的更多信息，请参阅 任务 1：为自动化创建服务角色。

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "sqs:*",
            "cloudformation:DeleteStack",
            "cloudformation:DescribeStacks"
         ],
         "Resource":"*"
      }
   ]
}
```

aws:executeAutomation - 运行另一个自动化

通过调用辅助运行手册运行辅助自动化。借助此操作，您可以为最常见的操作创建运行手册，并在自动化过程中引用这些运行手册。因为无需跨类似运行手册复制步骤，此操作可以简化您的运行手册。

辅助自动化将在启动主自动化的用户环境中运行。这意味着第二个自动化将使用与启动第一个自动化的用户相同的 AWS Identity and Access Management（IAM）角色或用户。

> **Important**
>
> 如果您在使用担任角色（使用 iam:passRole 策略的角色）的辅助自动化中指定参数，则启动主要自动化的用户或角色必须具有在辅助自动化中传递指定的担任角色的权限。有关为自动化设置担任角色的更多信息，请参阅 使用控制台为 Automation 创建服务角色。

输入

YAML

```
name: Secondary_Automation
action: aws:executeAutomation
maxAttempts: 3
timeoutSeconds: 3600
onFailure: Abort
inputs:
  DocumentName: secondaryAutomation
  RuntimeParameters:
    instanceIds:
    - i-1234567890abcdef0
```

JSON

```
{
   "name":"Secondary_Automation",
   "action":"aws:executeAutomation",
   "maxAttempts":3,
   "timeoutSeconds":3600,
   "onFailure":"Abort",
   "inputs":{
      "DocumentName":"secondaryAutomation",
      "RuntimeParameters":{
         "instanceIds":[
            "i-1234567890abcdef0"
         ]
      }
   }
}
```

DocumentName

要在步骤中运行的辅助运行手册的名称。对于相同 AWS 账户中的运行手册，指定运行手册名称。对于从其他 AWS 账户分享的运行手册，指定运行手册的 Amazon Resource Name (ARN)。有关使用共享运行手册的信息，请参阅 使用共享 SSM 文档。

类型：字符串

必需：是

DocumentVersion

要运行的辅助运行手册的版本。如果未指定，自动化将运行默认运行手册版本。

类型：字符串

必需：否

MaxConcurrency

此任务可以并行运行的最大目标数。您可以指定一个数字或一个百分比（如 10 或 10%）。

类型：字符串

必需：否

MaxErrors

系统停止在其他目标上运行自动化之前允许的错误数。您可以指定绝对数量的错误（如 10），也可以指定目标集百分比（如 10%）。例如，如果您指定 3，系统将在收到第四个错误时停止运行自动化。如果指定 0，则系统会在返回第一个错误结果后停止在其他资源上运行自动化。如果您在 50 个资源上运行自动化并将 MaxErrors 设置为 10%，则系统在收到第六个错误时停止在其他目标上运行自动化。

当达到 MaxErrors 阈值时，允许完成已经运行的自动化，但是其中一些自动化也可能失败。如果您需要确保失败的自动化不会超过指定的 MaxErrors，将 MaxConcurrency 设置为 1，因此一次进行一个自动化。

类型：字符串

必需：否

RuntimeParameters

辅助运行手册所需的参数。映射使用以下格式：{"parameter1" : "value1", "parameter2" : "value2" }

类型：映射

必需：否

Tags

分配给资源的可选元数据。您最多可以为自动化指定 5 个标签。

类型：MapList

必需：否

TargetLocations

位置是要在其中运行自动化的 AWS 区域 和/或 AWS 账户 的组合。必须指定至少 1 个项目，最多可以指定 100 个项目。为此参数指定值时，输出不会返回到父自动化。如果需要，则必须后续调用 API 操作，以检索子自动化的输出。

类型：MapList

必需：否

TargetMaps

运行手册参数到目标资源的键-值映射列表。Targets 和 TargetMaps 不能一起指定。

类型：MapList

必需：否

TargetParameterName

用作速率控制自动化目标资源的参数的名称。当您指定 Targets 时需要

类型：字符串

必需：否

目标

指向目标资源的键-值映射列表。当您指定 TargetParameterName 时需要

类型：MapList

必需：否

输出

输出

辅助自动化生成的输出。您可以使用以下格式引用输出：*Secondary_Automation_Step_Name*.Output

类型：StringList

示例如下：

```
- name: launchNewWindowsInstance
  action: 'aws:executeAutomation'
  onFailure: Abort
  inputs:
    DocumentName: launchWindowsInstance
  nextStep: getNewInstanceRootVolume
- name: getNewInstanceRootVolume
  action: 'aws:executeAwsApi'
  onFailure: Abort
  inputs:
    Service: ec2
    Api: DescribeVolumes
    Filters:
    - Name: attachment.device
      Values:
      - /dev/sda1
    - Name: attachment.instance-id
      Values:
      - '{{launchNewWindowsInstance.Output}}'
  outputs:
  - Name: rootVolumeId
    Selector: '$.Volumes[0].VolumeId'
    Type: String
  nextStep: snapshotRootVolume
- name: snapshotRootVolume
  action: 'aws:executeAutomation'
  onFailure: Abort
  inputs:
    DocumentName: AWS-CreateSnapshot
    RuntimeParameters:
    VolumeId:
    - '{{getNewInstanceRootVolume.rootVolumeId}}'
    Description:
    - 'Initial root snapshot for {{launchNewWindowsInstance.Output}}'
```

ExecutionId

辅助自动化的 ID。

类型：字符串

状态

辅助自动化的状态。

类型：字符串

aws:executeAwsApi - 调用并运行 AWS API 操作

调用并运行 AWS API 操作。支持大多数 API 操作，但某些 API 操作未经过测试。不支持流式处理 API 操作，例如 GetObject 操作。如果不确定要使用的 API 操作是否属于流式操作，请查看该服务的 Boto3 文档，判断该 API 是否需要流式输入或输出。我们会定期更新此操作使用的 Boto3 版本。但是，在新的 Boto3 版本发布后，可能需要长达几周的时间才能将更改反映到此操作中。每个 `aws:executeAwsApi` 操作最多可以运行 25 秒。有关如何使用此操作的更多示例，请参阅 其他运行手册示例。

输入

由您选择的 API 操作定义的输入。

YAML

```
action: aws:executeAwsApi
inputs:
  Service: The official namespace of the service
  Api: The API operation or method name
  API operation inputs or parameters: A value
outputs: # These are user-specified outputs
- Name: The name for a user-specified output key
  Selector: A response object specified by using jsonpath format
  Type: The data type
```

JSON

```
{
   "action":"aws:executeAwsApi",
   "inputs":{
      "Service":"The official namespace of the service",
      "Api":"The API operation or method name",
      "API operation inputs or parameters":"A value"
   },
   "outputs":[ These are user-specified outputs
      {
```

```
            "Name":"The name for a user-specified output key",
            "Selector":"A response object specified by using JSONPath format",
            "Type":"The data type"
        }
    ]
}
```

服务

包含要运行的 API 操作的 AWS 服务命名空间。您可以在 适用于 Python (Boto3) 的 AWS SDK 的可用服务中查看支持的 AWS 服务命名空间列表。可以在客户端部分找到此命名空间。例如，Systems Manager 的命名空间为 `ssm`。Amazon Elastic Compute Cloud (Amazon EC2) 的命名空间为 `ec2`。

类型：字符串

必需：是

API

要运行的 API 操作的名称。您可以在以下服务参考页面的左侧导航栏中选择服务来查看 API 操作（也称为方法）。在要调用的服务的客户端部分中选择一种方法。例如，下面的 Amazon RDS 方法页面中列出了 Amazon Relational Database Service (Amazon RDS) 的所有 API 操作（方法）。

类型：字符串

必需：是

API 操作输入

一个或多个 API 操作输入。您可以在以下服务参考页面的左侧导航栏中选择服务来查看可用的输入（也称为参数）。在要调用的服务的客户端部分中选择一种方法。例如，下面的 Amazon RDS 方法页面中列出了 Amazon RDS 的所有方法。选择 describe_db_instances 方法并向下滚动以查看可用的参数，例如 DBInstanceIdentifier、Name 和 Values。

YAML

```
inputs:
  Service: The official namespace of the service
  Api: The API operation name
  API input 1: A value
  API Input 2: A value
  API Input 3: A value
```

JSON

```
"inputs":{
      "Service":"The official namespace of the service",
      "Api":"The API operation name",
      "API input 1":"A value",
      "API Input 2":"A value",
      "API Input 3":"A value"
}
```

类型：由选择的 API 操作确定

必需：是

输出

输出由用户根据所选 API 操作的响应指定。

名称

输出的名称。

类型：字符串

必需：是

Selector

响应对象中特定属性的 JSONPath。您可以在以下服务参考页面的左侧导航栏中选择服务来查看响应对象。在要调用的服务的客户端部分中选择一种方法。例如，下面的 Amazon RDS 方法页面中列出了 Amazon RDS 的所有方法：选择 describe_db_instances 方法，然后向下滚动到响应结构部分。DBInstances 被列为响应对象。

类型：Integer、Boolean、String、StringList、StringMap 或 MapList

必需：是

类型

响应元素的数据类型。

类型：可变

必需：是

`aws:executeScript` - 运行脚本

使用指定的运行时和处理程序提供的 Python 或 PowerShell 脚本。每个 `aws:executeScript` 操作最多可以运行 600 秒（10 分钟）时间。您可以通过指定 `aws:executeScript` 步骤的 `timeoutSeconds` 参数来限制超时。

在函数中使用 return 语句将输出添加到输出有效负载中。有关为您的 `aws:executeScript` 操作定义输出的示例，请参阅示例 2：脚本化运行手册。您还可以将运行手册中 `aws:executeScript` 操作的输出发送到您指定的 Amazon CloudWatch Logs 日志组。有关更多信息，请参阅 使用 CloudWatch Logs 记录自动化操作输出。

如果想要将 `aws:executeScript` 操作的输出发送到 CloudWatch Logs，或者如果您为 `aws:executeScript` 操作指定的脚本调用 AWS API 操作，则始终需要 AWS Identity and Access Management (IAM) 服务角色（或承担角色）运行运行手册。

`aws:executeScript` 操作包含以下预安装的 PowerShell 核心模块。

- Microsoft.PowerShell.Host
- Microsoft.PowerShell.Management
- Microsoft.PowerShell.Security
- Microsoft.PowerShell.Utility
- PackageManagement
- PowerShellGet

要使用未预装的 PowerShell 核心模块，脚本必须安装带有 `-Force` 标志的模块，如以下命令所示。不支持 `AWSPowerShell.NetCore` 模块。用您想要安装的模块替换 *`ModuleName`*。

```
Install-Module ModuleName -Force
```

要在脚本中使用 PowerShell 核心 cmdlet，我们建议您使用 `AWS.Tools` 模块，如以下命令所示。将每个*`#######`*替换为您自己的信息。

- Amazon S3 cmdlet。

```
Install-Module AWS.Tools.S3 -Force
Get-S3Bucket -BucketName amzn-s3-demo-bucket
```

- Amazon EC2 cmdlet

```
Install-Module AWS.Tools.EC2 -Force
Get-EC2InstanceStatus -InstanceId instance-id
```

- 通用或独立于服务的 AWS Tools for Windows PowerShell cmdlet。

```
Install-Module AWS.Tools.Common -Force
Get-AWSRegion
```

如果脚本除了使用 PowerShell 核心 cmdlet 之外还初始化新对象，则还必须导入模块，如以下命令所示。

```
Install-Module AWS.Tools.EC2 -Force
Import-Module AWS.Tools.EC2

$tag = New-Object Amazon.EC2.Model.Tag
$tag.Key = "Tag"
$tag.Value = "TagValue"

New-EC2Tag -Resource i-02573cafcfEXAMPLE -Tag $tag
```

有关安装和导入 AWS.Tools 模块以及在运行手册中使用 PowerShell 核心 cmdlet 的示例，请参阅 自动化运行手册的视觉对象设计体验。

输入

提供运行脚本所需的信息。将每个*******替换为您自己的信息。

> **Note**
>
> Python 脚本的附件可以是 .py 文件或包含该脚本的 .zip 文件。PowerShell 脚本必须存储在 .zip 文件中。

YAML

```
action: "aws:executeScript"
inputs:
 Runtime: runtime
 Handler: "functionName"
```

```
 InputPayload:
  scriptInput: '{{parameterValue}}'
 Script: |-
   def functionName(events, context):
   ...
 Attachment: "scriptAttachment.zip"
```

JSON

```
{
    "action": "aws:executeScript",
    "inputs": {
        "Runtime": "runtime",
        "Handler": "functionName",
        "InputPayload": {
            "scriptInput": "{{parameterValue}}"
        },
        "Attachment": "scriptAttachment.zip"
    }
}
```

运行时

用于运行提供的脚本的运行时语言。aws:executeScript 支持 Python 3.7 (python3.7)、Python 3.8 (python3.8)、Python 3.9 (python3.9)Python 3.10 (python3.10)、Python 3.11 (python3.11) PowerShell Core 6.0 (dotnetcore2.1)、PowerShell 7.0 (dotnetcore3.1) 脚本和 PowerShell 7.4 (dotnet8) 脚本。

支持的值：**python3.7** | **python3.8** | **python3.9** | **python3.10** | **python3.11** | **PowerShell Core 6.0** | **PowerShell 7.0** | **PowerShell 7.4**

类型：字符串

必需：是

> **Note**
>
> 对于 python 运行时系统，环境提供 512 MB 的内存和 512 MB 的磁盘空间。对于 PowerShell 运行时，环境会提供 1024 MB 的内存和 512 MB 的磁盘空间。

处理程序

函数的名称。您必须确保在处理程序中定义的函数具有两个参数：events 和 context。PowerShell 运行时不支持此参数。

类型：字符串

必需：是（Python）| 不支持（PowerShell）

InputPayload

将传递给处理程序的第一个参数的 JSON 或 YAML 对象。这可用于将输入数据传递给脚本。

类型：字符串

必需：否

Python

```
description: Tag an instance
schemaVersion: '0.3'
assumeRole: '{{AutomationAssumeRole}}'
parameters:
    AutomationAssumeRole:
        type: String
        description: '(Required) The Amazon Resource Name (ARN) of the IAM role
 that allows Automation to perform the actions on your behalf. If no role is
 specified, Systems Manager Automation uses your IAM permissions to operate this
 runbook.'
    InstanceId:
        type: String
        description: (Required) The ID of the EC2 instance you want to tag.
mainSteps:
  - name: tagInstance
    action: 'aws:executeScript'
    inputs:
        Runtime: "python3.11"
        Handler: tagInstance
        InputPayload:
            instanceId: '{{InstanceId}}'
        Script: |-
          def tagInstance(events,context):
            import boto3

            #Initialize client
```

```
            ec2 = boto3.client('ec2')
            instanceId = events['instanceId']
            tag = {
                "Key": "Env",
                "Value": "ExamplePython"
            }
            print(f"Adding tag {tag} to instance id {instanceId}")
            ec2.create_tags(
                Resources=[instanceId],
                Tags=[tag]
            )
            return tag
    outputs:
      - Type: String
        Name: TagKey
        Selector: $.Payload.Key
outputs:
  - tagInstance.TagKey
```

PowerShell

```
description: Tag an instance
schemaVersion: '0.3'
assumeRole: '{{AutomationAssumeRole}}'
parameters:
  AutomationAssumeRole:
    type: String
    description: (Required) The Amazon Resource Name (ARN) of the IAM role
 that allows Automation to perform the actions on your behalf. If no role is
 specified, Systems Manager Automation uses your IAM permissions to operate this
 runbook.
  InstanceId:
    type: String
    description: (Required) The ID of the EC2 instance you want to tag.
mainSteps:
  - name: tagInstance
    action: aws:executeScript
    isEnd: true
    inputs:
      Runtime: PowerShell 7.0
      InputPayload:
        instanceId: '{{InstanceId}}'
      Script: |-
```

```
        Install-Module AWS.Tools.EC2 -Force
        Import-Module AWS.Tools.EC2

        $input = $env:InputPayload | ConvertFrom-Json

        $tag = New-Object Amazon.EC2.Model.Tag
        $tag.Key = "Env"
        $tag.Value = "ExamplePowerShell"

        Write-Information "Adding tag key: $($tag.Key) and value: $($tag.Value)
 to instance id $($input.instanceId)"
        New-EC2Tag -Resource $input.instanceId -Tag $tag

        return $tag
    outputs:
      - Type: String
        Name: TagKey
        Selector: $.Payload.Key
outputs:
  - tagInstance.TagKey
```

Script

要在自动化期间运行的嵌入式脚本。

类型：字符串

必需：否 (Python) | 是 (PowerShell)

附件

可以由操作调用的单独脚本文件或 .zip 文件的名称。指定与您在 `Attachments` 请求参数中指定的文档附件文件的 `Name` 相同的值。有关更多信息，请参阅 AWS Systems Manager API 参考中的附件。如果您使用附件提供脚本，还必须在您的运行手册的顶级元素部分中定义一个 `files` 部分。有关更多信息，请参阅 架构版本 0.3。

要为 Python 调用文件，请在 `Handler` 中使用 `filename.method_name` 格式。

> **Note**
>
> Python 脚本的附件可以是 .py 文件或包含该脚本的 .zip 文件。PowerShell 脚本必须存储在 .zip 文件中。

当在附件中包含 Python 库时，我们建议在每个模块目录中添加一个空`__init__.py` 文件。这允许您从脚本内容的附件中的库导入模块。例如：`from library import module`

类型：字符串

必需：否

输出

有效负载

函数返回的对象的 JSON 表示形式。返回最多 100KB 的数据。如果输出列表，则最多返回 100 个项目。

通过 aws:executeScript 使用附件

附件提供了一种强大的方法，可以将复杂的脚本、多个模块和外部依赖项打包并重复用于 `aws:executeScript` 操作。在您需要执行以下操作时，请使用附件：

- 将多个 Python 模块或 PowerShell 脚本打包在一起。
- 在多个运行手册中重复使用相同的脚本逻辑。
- 在脚本中包含外部库或依赖项。
- 通过分离复杂的脚本逻辑，保持运行手册定义简洁。
- 在团队或自动化工作流之间共享脚本包。

附件结构和打包

您可以附加单个文件或包含多个文件的 zip 包。具体结构取决于使用案例：

单个文件附件

对于简单的脚本，您可以附加单个 `.py` 文件 (Python) 或包含单个 PowerShell 脚本的 `.zip` 文件。

多模块包

对于需要多个模块的复杂自动化，请创建具有以下建议结构的 zip 包：

```
my-automation-package.zip
### main.py                    # Entry point script
### utils/
```

```
#   ### __init__.py           # Required for Python module imports
#   ### helper_functions.py   # Utility functions
#   ### aws_operations.py     # AWS-specific operations
### config/
#   ### __init__.py
#   ### settings.py           # Configuration settings
### requirements.txt          # Optional: document dependencies
```

> **Important**
>
> 对于 Python 包，您必须在每个包含 Python 模块的目录中包含一个空的 `__init__.py` 文件。这样您就可以使用标准的 Python 导入语法（例如 `from utils import helper_functions`）来导入模块。

PowerShell 包结构

PowerShell 附件必须打包为 zip 文件，其结构如下：

```
my-powershell-package.zip
### Main.ps1                  # Entry point script
### Modules/
#   ### HelperFunctions.ps1   # Utility functions
#   ### AWSOperations.ps1     # AWS-specific operations
### Config/
    ### Settings.ps1          # Configuration settings
```

创建带附件的运行手册

请按照以下步骤创建带附件的运行手册：

1. 将附件上传到 Amazon S3

   将脚本文件或 zip 包上传到自动化角色可访问的 S3 存储桶。请记下 S3 URI，以便在下一步中使用。

   ```
   aws s3 cp my-automation-package.zip s3://my-automation-bucket/scripts/
   ```

2. 计算附件校验和

   计算附件文件的 SHA-256 校验和以进行安全验证：

```
# Linux/macOS
shasum -a 256 my-automation-package.zip

# Windows PowerShell
Get-FileHash -Algorithm SHA256 my-automation-package.zip
```

3. 在运行手册中定义 files 部分

   在运行手册的顶层添加 `files` 部分以引用附件：

```
files:
  my-automation-package.zip:
    sourceType: "S3"
    sourceInfo:
      path: "s3://my-automation-bucket/scripts/my-automation-package.zip"
      checksums:
        sha256: "your-calculated-checksum-here"
```

4. 在 executeScript 步骤中引用附件

   使用 `Attachment` 参数来引用上传的文件：

```
- name: runMyScript
  action: aws:executeScript
  inputs:
    Runtime: python3.11
    Handler: main.process_data
    Attachment: my-automation-package.zip
    InputPayload:
      inputData: "{{InputParameter}}"
```

aws:executeScript 附件示例

以下示例演示了使用附件和 `aws:executeScript` 操作的不同方法。

示例 1：单个文件附件

此示例展示如何使用单个 Python 文件作为附件来处理 EC2 实例数据。

附件文件：process_instance.py

创建一个包含以下内容的 Python 文件：

```
import boto3
import json

def process_instance_data(events, context):
    """Process EC2 instance data and return formatted results."""
    try:
        instance_id = events.get('instanceId')
        if not instance_id:
            raise ValueError("instanceId is required")

        ec2 = boto3.client('ec2')

        # Get instance details
        response = ec2.describe_instances(InstanceIds=[instance_id])
        instance = response['Reservations'][0]['Instances'][0]

        # Format the response
        result = {
            'instanceId': instance_id,
            'instanceType': instance['InstanceType'],
            'state': instance['State']['Name'],
            'availabilityZone': instance['Placement']['AvailabilityZone'],
            'tags': {tag['Key']: tag['Value'] for tag in instance.get('Tags', [])}
        }

        print(f"Successfully processed instance {instance_id}")
        return result

    except Exception as e:
        print(f"Error processing instance: {str(e)}")
        raise
```

完整运行手册

以下是使用单个文件附件的完整运行手册：

```
description: Process EC2 instance data using single file attachment
schemaVersion: '0.3'
assumeRole: '{{AutomationAssumeRole}}'
parameters:
  AutomationAssumeRole:
    type: String
    description: (Required) IAM role for automation execution
```

```
  InstanceId:
    type: String
    description: (Required) EC2 instance ID to process

files:
  process_instance.py:
    sourceType: "S3"
    sourceInfo:
      path: "s3://my-automation-bucket/scripts/process_instance.py"
      checksums:
        sha256: "abc123def456..."

mainSteps:
  - name: processInstance
    action: aws:executeScript
    inputs:
      Runtime: python3.11
      Handler: process_instance.process_instance_data
      Attachment: process_instance.py
      InputPayload:
        instanceId: '{{InstanceId}}'
    outputs:
      - Type: StringMap
        Name: InstanceData
        Selector: $.Payload

outputs:
  - processInstance.InstanceData
```

示例 2：多模块包

此示例演示如何使用包含多个 Python 模块的 zip 包进行复杂的 S3 存储桶操作。

包结构

创建具有以下结构的 zip 包：

```
s3-operations.zip
### main.py
### utils/
#   ### __init__.py
#   ### s3_helper.py
#   ### validation.py
```

```
### config/
    ### __init__.py
    ### settings.py
```

main.py（入口点）

编排操作的主脚本：

```
from utils.s3_helper import S3Operations
from utils.validation import validate_bucket_name
from config.settings import get_default_settings

def cleanup_s3_bucket(events, context):
    """Clean up S3 bucket based on specified criteria."""
    try:
        bucket_name = events.get('bucketName')
        max_age_days = events.get('maxAgeDays', 30)

        # Validate inputs
        if not validate_bucket_name(bucket_name):
            raise ValueError(f"Invalid bucket name: {bucket_name}")

        # Initialize S3 operations
        s3_ops = S3Operations()
        settings = get_default_settings()

        # Perform cleanup
        deleted_objects = s3_ops.delete_old_objects(
            bucket_name,
            max_age_days,
            settings['dry_run']
        )

        result = {
            'bucketName': bucket_name,
            'deletedCount': len(deleted_objects),
            'deletedObjects': deleted_objects[:10],  # Return first 10 for brevity
            'dryRun': settings['dry_run']
        }

        print(f"Cleanup completed for bucket {bucket_name}")
        return result

    except Exception as e:
```

```
        print(f"Error during S3 cleanup: {str(e)}")
        raise
```

aws:executeScript 附件故障排除

使用以下指导来解决 `aws:executeScript` 附件的常见问题：

模块导入错误

如果在使用多模块包时收到导入错误：

- 确保在包含 Python 模块的每个目录中都包含一个空的 `__init__.py` 文件。
- 验证导入语句是否与 zip 包中的实际文件和目录结构相匹配。
- 始终使用相对导入（例如 `from .utils import helper`）或绝对导入（例如 `from utils import helper`）。

未找到附件错误

如果自动化未能找到附件：

- 验证 `Attachment` 参数值是否与 `files` 部分中的键完全匹配。
- 在 `files` 部分中检查 S3 存储桶路径和文件名是否正确。
- 确保自动化角色对附件 S3 位置具有 `s3:GetObject` 权限。
- 验证运行手册中的校验和是否与实际文件校验和匹配。

处理程序函数错误

如果收到与处理程序相关的错误：

- 对于 Python：在 `Handler` 参数中使用格式 `filename.function_name`（例如 `main.process_data`）。
- 确保处理程序函数只接受两个参数：`events` 和 `context`。
- 对于 PowerShell：请勿指定 `Handler` 参数；脚本将直接运行。

脚本执行失败

如果脚本在执行过程中失败：

- 检查自动化执行历史记录，获取详细的错误消息和堆栈跟踪。
- 使用 `print()` 语句 (Python) 或 `Write-Information` (PowerShell) 添加调试输出。
- 验证自动化角色是否已被授予所有必需的 AWS 权限。
- 在将脚本逻辑打包为附件之前，请在本地测试脚本逻辑。

退出代码和错误处理

要正确处理错误并返回退出代码：

- 在 Python 中：使用 `raise Exception("error message")` 指示脚本失败。
- 在 PowerShell 中：使用 `throw "error message"` 或 `Write-Error` 指示失败。
- 从函数返回结构化数据以提供详细的成功/失败信息。
- 使用 try-catch 块优雅地处理异常并提供有意义的错误消息。

`aws:executeStateMachine` - 运行 AWS Step Functions 状态机。

运行 AWS Step Functions 状态机。

输入

此操作支持 Step Functions StartExecution API 操作的大多数参数。

所需的 AWS Identity and Access Management (IAM) 权限

- `states:DescribeExecution`
- `states:StartExecution`
- `states:StopExecution`

YAML

```
name: executeTheStateMachine
action: aws:executeStateMachine
inputs:
  stateMachineArn: StateMachine_ARN
  input: '{"parameters":"values"}'
  name: name
```

JSON

```
{
    "name": "executeTheStateMachine",
    "action": "aws:executeStateMachine",
    "inputs": {
        "stateMachineArn": "StateMachine_ARN",
        "input": "{\"parameters\":\"values\"}",
        "name": "name"
    }
}
```

stateMachineArn

Step Functions 状态机的 Amazon Resource Name (ARN)。

类型：字符串

必需：是

name

执行的名称。

类型：字符串

必需：否

input

包含执行的 JSON 输入数据的字符串。

类型：字符串

必需：否

输出

此操作预定义了以下输出。

执行 Arn

执行的 ARN。

类型：字符串

input

包含执行的 JSON 输入数据的字符串。长度限制适用于有效负载大小，在 UTF-8 编码中以字节表示。

类型：字符串

name

执行的名称。

类型：字符串

output

执行的 JSON 输出数据。长度限制适用于有效负载大小，在 UTF-8 编码中以字节表示。

类型：字符串

startDate

开始执行的日期。

类型：字符串

stateMachineArn

状态机的执行的 ARN。

类型：字符串

status

执行的当前状态。

类型：字符串

停止日期

如果执行已经结束，则为执行停止的日期。

类型：字符串

`aws:invokeWebhook` – 调用 Automation Webhook 集成

调用指定的自动化 Webhook 集成。有关创建 Automation 集成的信息，请参阅 为 Automation 创建 Webhook 集成。

> **Note**
>
> 要使用 `aws:invokeWebhook` 操作，您的用户或服务角色必须允许以下操作：
>
> - ssm:GetParameter
> - kms:Decrypt
>
> 只有当您使用客户托管密钥加密集成的参数时，才需要 AWS Key Management Service (AWS KMS) `Decrypt` 操作的权限。

输入

提供您希望调用的 Automation 集成的信息。

YAML

```
action: "aws:invokeWebhook"
inputs:
 IntegrationName: "exampleIntegration"
 Body: "Request body"
```

JSON

```
{
    "action": "aws:invokeWebhook",
    "inputs": {
        "IntegrationName": "exampleIntegration",
        "Body": "Request body"
    }
}
```

IntegrationName

Automation 集成的名称。例如，`exampleIntegration`。您指定的集成必须已存在。

类型：字符串

必需：是

Body

调用 Webhook 集成时要发送的有效负载。

类型：字符串

必需：否

输出

响应

从 Webhook 提供商响应中收到的文本。

ResponseCode

从 Webhook 提供商响应中收到的 HTTP 状态代码。

aws:invokeLambdaFunction – 调用 AWS Lambda 函数

调用指定的 AWS Lambda 函数。

> **Note**
>
> 每个 aws:invokeLambdaFunction 操作最多可以运行 300 秒（5 分钟）。您可以通过指定 aws:invokeLambdaFunction 步骤的 timeoutSeconds 参数来限制超时。

输入

此操作支持 Lambda 服务的大多数调用参数。有关更多信息，请参阅调用。

YAML

```
name: invokeMyLambdaFunction
action: aws:invokeLambdaFunction
maxAttempts: 3
timeoutSeconds: 120
onFailure: Abort
inputs:
  FunctionName: MyLambdaFunction
```

JSON

```
{
    "name": "invokeMyLambdaFunction",
    "action": "aws:invokeLambdaFunction",
    "maxAttempts": 3,
    "timeoutSeconds": 120,
    "onFailure": "Abort",
    "inputs": {
        "FunctionName": "MyLambdaFunction"
    }
}
```

FunctionName

Lambda 函数的名称。此函数必须存在。

类型：字符串

必需：是

限定词

函数版本或别名。

类型：字符串

必需：否

InvocationType

调用类型。默认值为 RequestResponse。

类型：字符串

有效值：Event |RequestResponse |DryRun

必需：否

LogType

如果默认值为 Tail，则调用类型必须是 RequestResponse。Lambda 返回 Lambda 函数生成的最后 4KB 日志数据，采用 base64 编码。

类型：字符串

有效值：None | Tail

必需：否

ClientContext

特定于客户端的信息。

必需：否

InputPayload

传递给处理程序的第一个参数的 YAML 或 JSON 对象。可以使用此输入将数据传递给函数。与传统 Payload 输入相比，此输入提供了更大的灵活性和支持。如果为操作同时定义了 InputPayload 和 Payload，则 InputPayload 优先，不使用 Payload 值。

类型：StringMap

必需：否

有效负载

传递给处理程序的第一个参数的 JSON 字符串。这可以用于将输入数据传递给函数。我们建议使用 InputPayload 输入来添加功能性。

类型：字符串

必需：否

输出

StatusCode

HTTP 状态代码

FunctionError

如果存在，则表明执行函数时发生错误。响应负载中包含错误详细信息。

LogResult

Lambda 函数调用的 base64 编码日志。只有在调用类型为 RequestResponse 并且请求了日志时，日志才存在。

有效负载

Lambda 函数返回的对象的 JSON 表示形式。只有在调用类型为 RequestResponse 时才存在有效负载。

以下是来自 AWS-PatchInstanceWithRollback 运行手册的部分，演示了如何引用来自 aws:invokeLambdaFunction 操作的输出。

YAML

```
- name: IdentifyRootVolume
  action: aws:invokeLambdaFunction
  inputs:
    FunctionName: "IdentifyRootVolumeLambda-{{automation:EXECUTION_ID}}"
    Payload: '{"InstanceId": "{{InstanceId}}"}'
- name: PrePatchSnapshot
  action: aws:executeAutomation
  inputs:
    DocumentName: "AWS-CreateSnapshot"
    RuntimeParameters:
      VolumeId: "{{IdentifyRootVolume.Payload}}"
      Description: "ApplyPatchBaseline restoration case contingency"
```

JSON

```
{
    "name": "IdentifyRootVolume",
    "action": "aws:invokeLambdaFunction",
    "inputs": {
      "FunctionName": "IdentifyRootVolumeLambda-{{automation:EXECUTION_ID}}",
      "Payload": "{\"InstanceId\": \"{{InstanceId}}\"}"
    }
  },
  {
    "name": "PrePatchSnapshot",
    "action": "aws:executeAutomation",
    "inputs": {
      "DocumentName": "AWS-CreateSnapshot",
      "RuntimeParameters": {
        "VolumeId": "{{IdentifyRootVolume.Payload}}",
        "Description": "ApplyPatchBaseline restoration case contingency"
      }
```

```
    }
  }
```

aws:loop – 迭代自动化中的步骤

此操作在自动化运行手册中的步骤子集上进行迭代。您可以选择 do while 或 for each 样式循环。要构造 do while 循环，请使用 LoopCondition 输入参数。要构造 for each 循环，请使用 Iterators 和 IteratorDataType 输入参数。使用 aws:loop 操作时，请仅指定 Iterators 或 LoopCondition 输入参数。最大迭代次数为 100。

onCancel 属性仅可用于在循环中定义的步骤。aws:loop 操作不支持 onCancel 属性。onFailure 属性可用于 aws:loop 操作，但只有在出现导致该步骤失败的意外错误时使用该属性。如果您为循环中的步骤定义 onFailure 属性，则 aws:loop 操作将继承这些属性，并在出现故障时相应做出反应。

示例

以下是如何构造不同类型的循环操作的示例。

do while

```
name: RepeatMyLambdaFunctionUntilOutputIsReturned
action: aws:loop
inputs:
    Steps:
    - name: invokeMyLambda
        action: aws:invokeLambdaFunction
        inputs:
        FunctionName: LambdaFunctionName
        outputs:
        - Name: ShouldRetry
            Selector: $.Retry
            Type: Boolean
    LoopCondition:
        Variable: "{{ invokeMyLambda.ShouldRetry }}"
        BooleanEquals: true
    MaxIterations: 3
```

for each

```
name: stopAllInstancesWithWaitTime
```

```
action: aws:loop
inputs:
    Iterators: "{{ DescribeInstancesStep.InstanceIds }}"
    IteratorDataType: "String"
    Steps:
    - name: stopOneInstance
        action: aws:changeInstanceState
        inputs:
        InstanceIds:
            - "{{stopAllInstancesWithWaitTime.CurrentIteratorValue}}"
        CheckStateOnly: false
        DesiredState: stopped
    - name: wait10Seconds
        action: aws:sleep
        inputs:
        Duration: PT10S
```

输入

输入如下。

迭代器

要迭代的步骤的项目列表。迭代器的最大数为 100。

类型：StringList

必需：否

IteratorDataType

用于指定 Iterators 数据类型的可选参数。可将此参数的值与 Iterators 输入参数一起提供。如果您没有为此参数和 Iterators 指定值，则必须为 LoopCondition 参数指定值。

类型：字符串

有效值：Boolean | Integer | String | StringMap

默认：字符串

必需：否

LoopCondition

由 Variable 和要评估的运算符条件组成。如果您没有为此参数指定值，则必须为 Iterators 和 IteratorDataType 参数指定值。您可以通过组合使用 And、Not 和 Or 运算符来使用复杂的运算符评估。循环中的步骤完成后会对条件进行评估。如果条件为 true 并且尚未达到 MaxIterations 值，则循环中的步骤将再次运行。运算符条件如下：

字符串运算

- 字符串等于
- EqualsIgnoreCase
- StartsWith
- EndsWith
- 包含

数值运算

- NumericEquals
- NumericGreater
- NumericLesser
- NumericGreaterOrEquals
- NumericLesser
- NumericLesserOrEquals

布尔运算

- BooleanEquals

类型：StringMap

必需：否

MaxIterations

循环中步骤运行的最大次数。一旦达到为此输入指定的值，即使 LoopCondition 仍是 true，或者 Iterators 参数中仍然存在对象，循环也会停止运行。

类型：整数

有效值：1-100

必需：否

步骤

要在循环中运行的步骤列表。这些功能类似于嵌套运行手册。在这些步骤中，您可以使用语法 {{loopStepName.CurrentIteratorValue}} 访问 for each 循环的当前迭代器值。您还可以使用语法 {{loopStepName.CurrentIteration}} 访问两种循环类型的当前迭代的整数值。

类型：步骤列表

必需：是

输出

CurrentIteration

当前循环迭代为整数。迭代值从 1 开始。

类型：整数

CurrentIteratorValue

当前迭代器的值为字符串。此输出仅存在于 for each 循环中。

类型：字符串

aws:pause - 暂停自动化

此操作会暂停自动化。暂停后的自动化状态为正在等待。要继续自动化，请使用信号类型为 Resume 的 SendAutomationSignal API 操作。我们建议使用 aws:sleep 或 aws:approve 操作对您的工作流进行更精细的控制。

> **Note**
>
> 此操作的默认超时时间为 7 天（604800 秒），最长时间为 30 天（2592000 秒）。您可以通过指定 aws:pause 步骤的 timeoutSeconds 参数来限制或延长超时。

输入

输入如下。

YAML

```
name: pauseThis
action: aws:pause
timeoutSeconds: 1209600
inputs: {}
```

JSON

```
{
    "name": "pauseThis",
    "action": "aws:pause",
    "timeoutSeconds": "1209600",
    "inputs": {}
}
```

输出

无

aws:runCommand - 在托管实例上运行命令

运行指定的命令。

> **Note**
>
> 自动化仅支持 AWS Systems Manager Run Command 操作的一个输出。一个运行手册可以包括多个 Run Command 操作，但一次仅支持对一个操作输出。

输入

此操作支持大多数 send command 参数。有关更多信息，请参阅 SendCommand。

YAML

```
- name: checkMembership
  action: 'aws:runCommand'
```

```
  inputs:
    DocumentName: AWS-RunPowerShellScript
    InstanceIds:
      - '{{InstanceIds}}'
    Parameters:
      commands:
        - (Get-WmiObject -Class Win32_ComputerSystem).PartOfDomain
```

JSON

```
{
    "name": "checkMembership",
    "action": "aws:runCommand",
    "inputs": {
        "DocumentName": "AWS-RunPowerShellScript",
        "InstanceIds": [
            "{{InstanceIds}}"
        ],
        "Parameters": {
            "commands": [
                "(Get-WmiObject -Class Win32_ComputerSystem).PartOfDomain"
            ]
        }
    }
}
```

DocumentName

如果命令类型文档属于您或 AWS，请指定文档的名称。如果您使用的是另一个 AWS 账户与您共享的文档，指定文档的 Amazon Resource Name (ARN)。有关如何使用共享文档的更多信息，请参阅使用共享 SSM 文档。

类型：字符串

必需：是

InstanceIds

要在其中运行命令的实例 ID。您可以指定最多 50 个 ID。

您还可以使用虚拟参数 {{RESOURCE_ID}} 代替实例 ID，以便在目标组中的所有实例上运行命令。有关伪参数的更多信息，请参阅注册维护时段任务时使用伪参数。

另一种替代方法是使用 Targets 参数向一组实例发送命令。Targets 参数接受 Amazon Elastic Compute Cloud (Amazon EC2) 标签。有关如何使用 Targets 参数的更多信息，请参阅 大规模运行命令。

类型：StringList

必需：否（如果未指定实例 ID 或使用 {{RESOURCE_ID}} 虚拟参数，则必须指定 Targets 参数。）

目标

一组搜索条件，使用您指定的键/值组合来设置实例目标。如果未在调用中提供一个或多个实例 ID，则 Targets 为必需。有关如何使用 Targets 参数的更多信息，请参阅 大规模运行命令。

类型：MapList（列表中 map 的架构必须与对象匹配。）有关更多信息，请参阅 AWS Systems ManagerAPI 参考中的目标。

必需：否（如果未指定 Targets，则必须指定实例 ID 或使用 {{RESOURCE_ID}} 虚拟参数。）

以下为示例。

YAML

```
- name: checkMembership
  action: aws:runCommand
  inputs:
    DocumentName: AWS-RunPowerShellScript
    Targets:
      - Key: tag:Stage
        Values:
          - Gamma
          - Beta
      - Key: tag-key
        Values:
          - Suite
    Parameters:
      commands:
        - (Get-WmiObject -Class Win32_ComputerSystem).PartOfDomain
```

JSON

```
{
    "name": "checkMembership",
```

```
    "action": "aws:runCommand",
    "inputs": {
        "DocumentName": "AWS-RunPowerShellScript",
        "Targets": [
            {
                "Key": "tag:Stage",
                "Values": [
                    "Gamma", "Beta"
                ]
            },
            {
                "Key": "tag:Application",
                "Values": [
                    "Suite"
                ]
            }
        ],
        "Parameters": {
            "commands": [
                "(Get-WmiObject -Class Win32_ComputerSystem).PartOfDomain"
            ]
        }
    }
}
```

参数

文档中指定的必需参数和可选参数。

类型：映射

必需：否

CloudWatchOutputConfig

用于将命令输出发送到 Amazon CloudWatch Logs 的配置选项。有关将命令输出发送到 CloudWatch Logs 的更多信息，请参阅 为 Run Command 配置 Amazon CloudWatch Logs。

类型：StringMap（map 的架构必须与对象匹配。有关更多信息，请参阅 AWS Systems ManagerAPI 参考 中的 CloudWatchOutputConfig）。

必需：否

以下为示例。

YAML

```
- name: checkMembership
  action: aws:runCommand
  inputs:
    DocumentName: AWS-RunPowerShellScript
    InstanceIds:
      - "{{InstanceIds}}"
    Parameters:
      commands:
        - "(Get-WmiObject -Class Win32_ComputerSystem).PartOfDomain"
    CloudWatchOutputConfig:
      CloudWatchLogGroupName: CloudWatchGroupForSSMAutomationService
      CloudWatchOutputEnabled: true
```

JSON

```
{
    "name": "checkMembership",
    "action": "aws:runCommand",
    "inputs": {
        "DocumentName": "AWS-RunPowerShellScript",
        "InstanceIds": [
            "{{InstanceIds}}"
        ],
        "Parameters": {
            "commands": [
                "(Get-WmiObject -Class Win32_ComputerSystem).PartOfDomain"
            ]
        },
        "CloudWatchOutputConfig" : {
                "CloudWatchLogGroupName":
 "CloudWatchGroupForSSMAutomationService",
                "CloudWatchOutputEnabled": true
        }
    }
}
```

注释

有关此命令的用户定义的信息。

类型：字符串

必需：否

DocumentHash

文档的哈希。

类型：字符串

必需：否

DocumentHashType

哈希的类型。

类型：字符串

有效值：Sha256 | Sha1

必需：否

NotificationConfig

用于发送通知的配置。

必需：否

OutputS3BucketName

命令输出响应的 S3 存储桶的名称。托管式节点必须具有 S3 存储桶的权限才能成功记录输出。

类型：字符串

必需：否

OutputS3KeyPrefix

前缀。

类型：字符串

必需：否

ServiceRoleArn

AWS Identity and Access Management (IAM) 角色的 ARN。

类型：字符串

必需：否

TimeoutSeconds

等待命令传递到实例上的 AWS Systems Manager SSM Agent 的时间（单位：秒）。如果在达到指定的值之前，命令未被实例上的 SSM Agent 接收，则命令的状态将更改为 `Delivery Timed Out`。

类型：整数

必需：否

有效值：30-2592000

输出

CommandId

命令的 ID。

状态

命令的状态。

ResponseCode

命令的响应代码。如果运行的文档有多个步骤，则不会为此输出返回值。

输出

命令的输出。如果您使用命令定位一个标记或多个实例，则不会返回任何输出值。您可以使用 `GetCommandInvocation` 和 `ListCommandInvocations` API 操作来检索单个实例的输出。

`aws:runInstances` – 启动 Amazon EC2 实例

启动新的 Amazon Elastic Compute Cloud (Amazon EC2) 实例。

输入

此操作支持大多数 API 参数。有关更多信息，请参阅 RunInstances API 文档。

YAML

```
name: launchInstance
action: aws:runInstances
maxAttempts: 3
timeoutSeconds: 1200
onFailure: Abort
inputs:
  ImageId: ami-12345678
  InstanceType: t2.micro
  MinInstanceCount: 1
  MaxInstanceCount: 1
  IamInstanceProfileName: myRunCmdRole
  TagSpecifications:
  - ResourceType: instance
    Tags:
    - Key: LaunchedBy
      Value: SSMAutomation
    - Key: Category
      Value: HighAvailabilityFleetHost
```

JSON

```
{
   "name":"launchInstance",
   "action":"aws:runInstances",
   "maxAttempts":3,
   "timeoutSeconds":1200,
   "onFailure":"Abort",
   "inputs":{
      "ImageId":"ami-12345678",
      "InstanceType":"t2.micro",
      "MinInstanceCount":1,
      "MaxInstanceCount":1,
      "IamInstanceProfileName":"myRunCmdRole",
      "TagSpecifications":[
         {
            "ResourceType":"instance",
            "Tags":[
               {
                  "Key":"LaunchedBy",
                  "Value":"SSMAutomation"
               },
```

```
                {
                   "Key":"Category",
                   "Value":"HighAvailabilityFleetHost"
                }
             ]
          }
       ]
    }
}
```

AdditionalInfo

预留。

类型：字符串

必需：否

BlockDeviceMappings

适用于实例的块储存设备。

类型：MapList

必需：否

ClientToken

用于确保请求的幂等性的标识符。

类型：字符串

必需：否

DisableApiTermination

打开或关闭实例 API 终止。

类型：布尔值

必需：否

EbsOptimized

打开或关闭 Amazon Elastic Block Store (Amazon EBS) 优化。

类型：布尔值

必需：否

IamInstanceProfileArn

针对实例的 AWS Identity and Access Management (IAM) 实例配置文件的 Amazon Resource Name (ARN)。

类型：字符串

必需：否

IamInstanceProfileName

实例的 IAM 实例配置文件的名称。

类型：字符串

必需：否

ImageId

Amazon Machine Image (AMI) 的 ID。

类型：字符串

必需：是

InstanceInitiatedShutdownBehavior

指示此实例是否在系统关闭时停止或终止。

类型：字符串

必需：否

InstanceType

实例类型。

> **Note**
>
> 如果未提供实例类型值，则使用 m1.小型实例类型。

类型：字符串

必需：否

KernelId

内核的 ID。

类型：字符串

必需：否

KeyName

密钥对的名称。

类型：字符串

必需：否

MaxInstanceCount

要启动的实例的最大数量。

类型：字符串

必需：否

MetadataOptions

实例的元数据选项。有关更多信息，请参阅 InstanceMetadataOptionsRequest。

类型：StringMap

必需：否

MinInstanceCount

要启动的实例的最小数量。

类型：字符串

必需：否

监控

打开或关闭详细监控。

类型：布尔值

必需：否

NetworkInterfaces

网络接口。

类型：MapList

必需：否

Placement

实例的置放。

类型：StringMap

必需：否

PrivateIpAddress

主要 IPv4 地址。

类型：字符串

必需：否

RamdiskId

RAM 磁盘的 ID。

类型：字符串

必需：否

SecurityGroupIds

实例的安全组的 ID。

类型：StringList

必需：否

SecurityGroups

实例的安全组的名称。

类型：StringList

必需：否

SubnetId

子网 ID。

类型：字符串

必需：否

TagSpecifications

在启动期间应用于资源的标签。您只能在启动时标记实例和卷。指定的标签将应用于在启动期间创建的所有实例或卷。要在启动实例后对其进行标记，请使用 aws:createTags - 为 AWS 资源创建标签 操作。

类型：MapList（有关更多信息，请参阅 TagSpecification。）

必需：否

UserData

作为字符串文本值提供的脚本。如果输入文本值，则必须为 Base64 编码。

类型：字符串

必需：否

输出

InstanceIds

实例的 ID。

InstanceState

实例的当前状态。

aws:sleep - 延迟自动化

将自动化延迟指定的时间。此操作使用国际标准化组织 (ISO) 8601 日期和时间格式。有关此日期和时间格式的更多信息，请参阅 ISO 8601。

输入

您可将自动化延迟指定的时间。

YAML

```
name: sleep
action: aws:sleep
inputs:
  Duration: PT10M
```

JSON

```
{
   "name":"sleep",
   "action":"aws:sleep",
   "inputs":{
      "Duration":"PT10M"
   }
}
```

还可以将自动化延迟到指定日期和时间。如果指定日期和时间已过，操作将立即执行。

YAML

```
name: sleep
action: aws:sleep
inputs:
  Timestamp: '2020-01-01T01:00:00Z'
```

JSON

```
{
    "name": "sleep",
    "action": "aws:sleep",
    "inputs": {
        "Timestamp": "2020-01-01T01:00:00Z"
    }
}
```

> **Note**
>
> 自动化支持的最大延迟为 604799 秒（7 天）。

持续时间

ISO 8601 持续时间。您不能指定负数持续时间。

类型：字符串

必需：否

Timestamp

ISO 8601 时间戳。如果您没有为此参数指定值，那么必须为 Duration 参数指定一个值。

类型：字符串

必需：否

输出

无

aws:updateVariable – 更新运行手册变量的值

此操作会更新运行手册变量的值。值的数据类型必须与要更新的变量的数据类型相匹配。不支持数据类型转换。aws:updateVariable 操作不支持 onCancel 属性。

输入

输入如下。

YAML

```
name: updateStringList
action: aws:updateVariable
inputs:
    Name: variable:variable name
    Value:
    - "1"
    - "2"
```

JSON

```
{
```

```
    "name": "updateStringList",
    "action": "aws:updateVariable",
    "inputs": {
        "Name": "variable:variable name",
        "Value": ["1","2"]
    }
}
```

名称

要更新其值的变量名称。必须使用格式 `variable:variable name`

类型：字符串

必需：是

值

要分配给变量的新值。值必须与变量的数据类型相匹配。不支持数据类型转换。

类型：Boolean | Integer | MapList | String | StringList | StringMap

必需：是

约束：

- MapList 最多可以包含 200 个项目。
- 密钥长度的最小长度可以为 1，最大长度可以为 50。
- StringList 最少可以为 0 项，最多可以为 50 项。
- 字符串长度的最小长度可以为 1，最大长度可以为 512。

输出

无

aws:waitForAwsResourceProperty - 等待 AWS 资源属性

`aws:waitForAwsResourceProperty` 操作可以让自动化等待特定的资源状态或事件状态，然后才继续自动化。有关如何使用此操作的更多示例，请参阅 其他运行手册示例。

> **Note**
>
> 此操作的默认超时值为 3600 秒（1 小时）。您可以通过指定 aws:waitForAwsResourceProperty 步骤的 timeoutSeconds 参数来限制或延长超时。有关如何使用此操作的更多信息和示例，请参阅 处理运行手册中的超时。

输入

由您选择的 API 操作定义的输入。

YAML

```
action: aws:waitForAwsResourceProperty
inputs:
  Service: The official namespace of the service
  Api: The API operation or method name
  API operation inputs or parameters: A value
  PropertySelector: Response object
  DesiredValues:
  - Desired property value
```

JSON

```
{
  "action": "aws:waitForAwsResourceProperty",
  "inputs": {
    "Service":"The official namespace of the service",
    "Api":"The API operation or method name",
    "API operation inputs or parameters":"A value",
    "PropertySelector": "Response object",
    "DesiredValues": [
      "Desired property value"
    ]
  }
}
```

服务

包含要运行的 API 操作的 AWS 服务命名空间。例如，AWS Systems Manager 的命名空间为 `ssm`。Amazon Elastic Compute Cloud (Amazon EC2) 的命名空间为 `ec2`。您可以在《AWS CLI 命令参考》的可用服务部分查看支持的 AWS 服务命名空间列表。

类型：字符串

必需：是

API

要运行的 API 操作的名称。您可以在以下服务参考页面的左侧导航栏中选择服务来查看 API 操作（也称为方法）。在要调用的服务的客户端部分中选择一种方法。例如，下面的 Amazon RDS 方法页面中列出了 Amazon Relational Database Service (Amazon RDS) 的所有 API 操作（方法）。

类型：字符串

必需：是

API 操作输入

一个或多个 API 操作输入。您可以在以下服务参考页面的左侧导航栏中选择服务来查看可用的输入（也称为参数）。在要调用的服务的客户端部分中选择一种方法。例如，下面的 Amazon RDS 方法页面中列出了 Amazon RDS 的所有方法。选择 describe_db_instances 方法并向下滚动以查看可用的参数，例如 DBInstanceIdentifier、Name 和 Values。

YAML

```
inputs:
  Service: The official namespace of the service
  Api: The API operation name
  API input 1: A value
  API Input 2: A value
  API Input 3: A value
```

JSON

```
"inputs":{
      "Service":"The official namespace of the service",
      "Api":"The API operation name",
      "API input 1":"A value",
      "API Input 2":"A value",
      "API Input 3":"A value"
```

```
}
```

类型：由选择的 API 操作确定

必需：是

PropertySelector

响应对象中特定属性的 JSONPath。您可以在以下服务参考页面的左侧导航栏中选择服务来查看响应对象。在要调用的服务的客户端部分中选择一种方法。例如，下面的 Amazon RDS 方法页面中列出了 Amazon RDS 的所有方法：选择 describe_db_instances 方法，然后向下滚动到响应结构部分。DBInstances 被列为响应对象。

类型：字符串

必需：是

DesiredValues

要继续自动化的预期状态。

类型：MapList、StringList

必需：是

自动化系统变量

AWS Systems Manager 自动化运行手册使用以下变量。有关如何使用这些变量的示例，请查看 `AWS-UpdateWindowsAmi` 运行手册的 JSON 源。

查看 `AWS-UpdateWindowsAmi` 运行手册的 JSON 源

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择文档。
3. 在文档列表中，使用“搜索栏”或“搜索栏”右侧的数字选择运行手册 **`AWS-UpdateWindowsAmi`**。
4. 选择内容选项卡。

系统变量

自动化运行手册目前支持以下系统变量。

Variable	详细信息
`global:ACCOUNT_ID`	在其中运行自动化的用户或角色的 AWS 账户 ID。
`global:DATE`	格式为 yyyy-MM-dd 的日期（运行时间）。
`global:DATE_TIME`	格式为 yyyy-MM-dd_HH.mm.ss 的日期和时间（运行时间）。
`global:AWS_PARTITION`	资源所处的分区。对于标准 AWS 区域，分区是 aws。对于位于其他分区中的资源，则分区是 aws-*partitionname* 。例如，AWS GovCloud (US-West) 区域中资源的区域为 aws-us-gov 。
`global:REGION`	运行手册在其中运行的区域。例如，us-east-2。

自动化变量

自动化运行手册支持以下自动化变量。

Variable	详细信息
`automation:EXECUTION_ID`	分配给当前自动化的唯一标识符。例如，1a2b3c-1a2b3c-1a2b3c-1a2b3c 1a2b3c1a2b3c 。

主题

- 术语
- 支持的场景
- 不支持的场景

术语

以下术语描述了如何解析变量和参数。

租期	定义	示例
常量 ARN	无变量的有效 Amazon Resource Name (ARN)。	`arn:aws:iam::123456789012:role/roleName`
运行手册参数	在运行手册级别定义的参数（例如 `instanceId` ）。可在替换基本字符串时使用该参数。系统会在启动执行时间提供该参数的值。	`{ "description": "Create Image Demo", "version": "0.3", "assumeRole": "Your_Automation_Assume_Role_ARN ", "parameters":{ "instanceId": { "type": "String", "description": "Instance to create image from" } }`
系统变量	在评估运行手册的任何部分时，被替换到运行手册中的常规变量。	`"activities": [ { "id": "copyImage", "activityType": "AWS-CopyImage", "maxAttempts": 1, "onFailure": "Continue", "inputs": { "ImageName": "{{imageName}}", "SourceImageId": "{{sourceImageId}}", "SourceRegion": "{{sourceRegion}}",`

租期	定义	示例
		"Encrypted": true, "ImageDescription": "Test CopyImage Description created on **{{global:DATE}}** " } }]

租期	定义	示例

自动化变量	在评估运行手册的任何部分时，被替换到运行手册中且与自动化执行相关的变量。	{
		"name": "runFixedCmds",
		"action": "aws:runCommand",
		"maxAttempts": 1,
		"onFailure": "Continue",
		"inputs": {
		"DocumentName": "AWS-RunPowerShellScript",
		"InstanceIds": [
		"{{LaunchInstance.InstanceIds}}"
		],
		"Parameters": {
		"commands": [
		"dir",
		"date",
		"“{{outputFormat}}” -f “left”,”right”,”{{global:DATE}}”,” **{{automation:EXECUTION_ID}}** ”
		]
		}
		}
		}

租期	定义	示例
Systems Manager 参数	在 AWS Systems Manager Parameter Store 内定义的变量。该参数不能在步骤输入中直接引用。访问该参数可能需要权限。	description: Launch new Windows test instance schemaVersion: '0.3' assumeRole: '{{AutomationAssumeRole}}' parameters: AutomationAssumeRole: type: String default: '' description: >- (Required) The ARN of the role that allows Automation to perform the actions on your behalf. If no role is specified, Systems Manager Automation uses your IAM permissions to run this runbook. LatestAmi: type: String default: >- {{ssm:/aws/service/ami-windows-latest/Windows_Server-2016-English-Full-Base}} description: The latest Windows Server 2016 AMI queried from the public parameter. mainSteps: - name: launchInstance action: 'aws:runInstances' maxAttempts: 3

租期	定义	示例
		timeoutSeconds: 1200 onFailure: Abort inputs: ImageId: '{{LatestAmi}}' ...

支持的场景

场景	注释	示例
创建时的常量 ARN assumeRole 。	执行授权检查来验证是否允许调用用户传递给定的 assumeRole 。	{ "description": "Test all Automation resolvable parameters", "schemaVersion": "0.3", "assumeRole": ***"arn:aws:iam::123456789012:role/roleName"*** , "parameters": { ...
启动自动化时为 AssumeRole 提供的运行手册参数。	必须在运行手册的参数列表中定义。	{ "description": "Test all Automation resolvable parameters", "schemaVersion": "0.3", "assumeRole": **"{{dynamicARN}}"** , "parameters": { ...

场景	注释	示例

启动时为运行手册参数提供的值。	客户提供要用于参数的值。需在运行手册的参数列表中定义在启动时提供的所有输入。	```
		...
		"parameters": {
		    "amiId": {
		      "type": "String",
		      "default":
		 "ami-12345678 ",
		      "description":
		 "list of commands to
		 run as part of first
		 step"
		    },
		...
		``
		“启动自动化执行”的输入包括：{"amiId" : ["ami-12345678 "] }

场景	注释	示例

在运行手册内容中引用的 Systems Manager 参数。	变量存在于客户账户中，或者是公开访问的参数，并且运行手册的 AssumeRole 有权访问该变量。创建时将执行检查，以确认 AssumeRole 都不能访问它。不能在步骤输入中直接引用参数。	```
		...
		parameters:
		    LatestAmi:
		    type: String
		    default: >-
		      {{ssm:/aws/
		service/ami-wind
		ows-latest/Windows
		_Server-2016-English-
		Full-Base}}
		    description: The
		 latest Windows Server
		 2016 AMI queried from
		 the public parameter.
		mainSteps:
		  - name: launchIns
		tance
		    action: 'aws:runI
		nstances'
		    maxAttempts: 3
		    timeoutSeconds:
		 1200
		    onFailure: Abort
		    inputs:
		      ImageId: '{{Latest
		Ami}}'
		...
		``

场景	注释	示例

在步骤定义中引用的系统变量	启动自动化时被替换到运行手册中的系统变量。注入到运行手册中的值与替换发生的时间相关。换言之，由于在运行步骤之间需要花费一定时间，因此在步骤 1 中注入的时间变量的值将与在步骤 3 中注入的值不同。无需在运行手册的参数列表中设置系统变量。	```
		...
		  "mainSteps": [
		    {
		      "name": "RunSomeC
		ommands",
		      "action":
		 "aws:runCommand",
		      "maxAttempts": 1,
		      "onFailure":
		 "Continue",
		      "inputs": {
		        "DocumentName":
		 "AWS:RunPowerShell",
		        "InstanceIds":
		 ["{{LaunchInstance
		.InstanceIds}}"],
		        "Parameters": {
		            "commands
		" : [
		                "echo
		 {The time is now
		 {{global:DATE_TIME
		}}}"
		            ]
		        }
		    }
		}, ...
		``

场景	注释	示例

在步骤定义中引用的自动化变量。	无需在运行手册的参数列表中设置自动化变量。唯一的受支持自动化变量是 自动化:EXECUTION_ID。	```
		...
		"mainSteps": [
		    {
		      "name": "invokeLa
		mbdaFunction",
		      "action":
		 "aws:invokeLambdaF
		unction",
		      "maxAttempts": 1,
		      "onFailure":
		 "Continue",
		      "inputs": {
		        "FunctionName":
		 "Hello-World-Lambd
		aFunction",
		
		"Payload" :
		 "{ "executionId" :
		 "{{automation:EXEC
		UTION_ID}}" }"
		      }
		    }
		...
		``

场景	注释	示例

在下一步的定义中参考上一步的输出。	这是一个参数重定向。可使用语法 {{stepName.OutputName}} 引用上一步的输出。客户不能将该语法用于运行手册参数。在引用步骤运行时解决此问题。运行手册的参数中未列出该参数。	```
		...
		"mainSteps": [
		    {
		      "name": "LaunchInstance",
		      "action": "aws:runInstances",
		      "maxAttempts": 1,
		      "onFailure": "Continue",
		      "inputs": {
		        "ImageId": "{{amiId}}",
		        "MinInstanceCount": 1,
		        "MaxInstanceCount": 2
		      }
		    },
		    {
		      "name":"changeState",
		      "action": "aws:changeInstanceState",
		      "maxAttempts": 1,
		      "onFailure": "Continue",
		      "inputs": {
		        "InstanceIds": ["{{LaunchInstance.InstanceIds}}"],
		        "DesiredState": "terminated"
		      }
		    }
		
		...
		``

不支持的场景

场景	注释	示例
为 assumeRole 提供的 Systems Manager 参数创建时	不支持。	... { "description": "Test all Automation resolvable parameters", "schemaVersion": "0.3", "assumeRole": "*{{ssm:administratorRoleARN}}* ", "parameters": { ...
步骤输入中直接引用的 Systems Manager 参数。	在创建时返回 InvalidDocumentContent 异常。	... mainSteps: - name: launchInstance action: 'aws:runInstances' maxAttempts: 3 timeoutSeconds: 1200 onFailure: Abort inputs: ImageId: '{{ssm:/aws/service/ami-windows-latest/Windows_Server-2016-English-Full-Base}}' ...
变量步骤定义	运行手册中步骤的定义由变量构建而成。	...

场景	注释	示例

		"mainSteps": [
		{
		"name": "LaunchInstance",
		"action": "aws:runInstances",
		"**{{attemptModel}}** ": 1,
		"onFailure": "Continue",
		"inputs": {
		"ImageId": "*ami-12345678* ",
		"MinInstanceCount": 1,
		"MaxInstanceCount": 2
		}
		
		...
		
		User supplies input :
		{ "attemptModel" :
		"**minAttempts** " }

场景	注释	示例
交叉引用运行手册参数	用户在启动时提供了输入参数，而该参数引用了运行手册中的另一参数。	... "parameters": { "amiId": { "type": "String", "default": "ami-7f2e6015 ", "description": "list of commands to run as part of first step" }, "alternateAmiId": { "type": "String", "description": "The alternate AMI to try if this first fails". "default" : "{{amiId} }" }, ...

场景	注释	示例
多层扩展	运行手册定义了评估变量名称的变量。它位于变量分隔符（即 {{ }}）内，并扩展为变量/参数的值。	... "parameters": { "*firstParameter* ": { "type": "String", "default": "param2", "description": "The parameter to reference" }, "*secondParameter* ": { "type": "String", "default" : "echo {Hello world}", "description": "What to run" } }, "mainSteps": [{ "name": "runFixedCmds", "action": "aws:runCommand", "maxAttempts": 1, "onFailure": "Continue", "inputs": { "DocumentName": "AWS-RunPowerShellScript", "InstanceIds" : "{{LaunchInstance.InstanceIds}}", "Parameters": { "commands": ["{{ ***{{firstParameter}}*** }}"] }

场景	注释	示例
		... Note: The customer intention here would be to run a command of "echo {Hello world}"

场景	注释	示例

引用属于不同变量类型的运行手册步骤的输出	用户在后续步骤中引用前面运行手册步骤的输出。输出是一个不符合后续步骤中的操作要求的变量类型。	```
		...
		mainSteps:
		- name: getImageId
		  action: aws:execu
		teAwsApi
		  inputs:
		    Service: ec2
		    Api: DescribeImages
		    Filters:
		    - Name: "name"
		      Values:
		      - "{{ImageName}}"
		  outputs:
		  - Name: ImageIdList
		    Selector: "$.Images
		"
		    Type: "StringList"
		- name: copyMyImages
		  action: aws:copyImage
		  maxAttempts: 3
		  onFailure: Abort
		  inputs:
		    SourceImageId:
		 {{getImageId.Image
		IdList}}
		    SourceRegion: ap-
		northeast-2
		    ImageName:
		 Encrypted Copies of
		 LAMP base AMI in ap-
		northeast-2
		    Encrypted: true
		...
		Note: You must provide
		 the type required
		 by the Automation
		 action.
		In this case, aws:copyI
		mage requires a
		 "String" type variable
		 but the preceding step
		``

场景	注释	示例
		outputs a "StringList" type variable.

创建您自己的运行手册

自动化运行手册定义了自动化运行时 Systems Manager 在托管实例和其他 AWS 资源上执行的操作。Automation 是 AWS Systems Manager 中的一项工具。运行手册包含一个或多个按顺序运行的步骤。每个步骤是根据单个操作生成的。可以将一个步骤的输出作为后面步骤的输入。

运行这些操作及其步骤的过程称为自动化。

让您可以在 AWS 环境中自动完成各种不同操作的运行手册支持的操作类型。例如，通过使用 `executeScript` 操作类型，您可以直接在运行手册中嵌入 Python 或 PowerShell 脚本。（在创建自定义运行手册时，您可以按内联方式添加脚本，或者从 S3 存储桶或本地计算机中附加脚本。）您可以使用 `createStack` 和 `deleteStack` 操作类型自动管理 AWS CloudFormation 资源。此外，通过使用 `executeAwsApi` 操作类型，步骤可以在任意 AWS 服务中运行任意 API 操作，包括创建或删除 AWS 资源、启动其他进程、触发通知等。

有关自动化支持的所有 20 种操作类型的列表，请参阅 Systems Manager 自动化操作参考。

AWS Systems Manager 自动化提供了几个包含预定义步骤的文档，您可以使用这些步骤执行常见任务，例如重新启动一个或多个 Amazon Elastic Compute Cloud (Amazon EC2) 实例或创建 Amazon Machine Image (AMI)。您还可以创建自己的运行手册并与其他 AWS 账户 共享，或者向所有自动化用户公开。

运行手册是使用 YAML 或 JSON 编写的。但是，通过使用 Systems Manager 自动化中的文档生成器，您可以创建运行手册，而无需使用本机 JSON 或 YAML 创作。

> **Important**
>
> 如果您运行使用 AWS Identity and Access Management (IAM) 服务角色调用其他服务的自动化工作流程，请注意必须使用权限将该服务角色配置为调用这些服务。该要求适用于所有 AWS 自动化运行手册（`AWS-*` 运行手册），例如 `AWS-ConfigureS3BucketLogging`、`AWS-CreateDynamoDBBackup` 和 `AWS-RestartEC2Instance` 运行手册等。对于您创建的任何自定义自动化运行手册，如果这些

文档使用调用其他服务的操作来调用其他 AWS 服务，则此要求同样适用。例如，如果使用 `aws:executeAwsApi`、`aws:createStack` 或 `aws:copyImage` 操作，则您必须配置具有权限的服务角色来调用这些服务。您可以将 IAM 内联策略添加到该角色，从而向其他 AWS 服务授予权限。有关更多信息，请参阅 （可选）添加自动化内联策略或客户管理型策略来调用其他 AWS 服务。

有关可在运行手册中指定的操作的信息，请参阅 Systems Manager 自动化操作参考。

有关使用 AWS Toolkit for Visual Studio Code 创建运行手册的信息，请参阅 《AWS Toolkit for Visual Studio Code 用户指南》 中的 使用 Systems Manager 自动化文档。

有关使用视觉设计器创建自定义运行手册的信息，请参阅 自动化运行手册的视觉对象设计体验。

目录

- 自动化运行手册的视觉对象设计体验
  - 开始前的准备工作
  - 视觉对象设计体验界面概述
    - 操作浏览器
    - 画布
    - 表单
    - 键盘快捷键
  - 使用视觉对象设计体验
    - 创建运行手册工作流程
    - 设计运行手册
    - 更新运行手册
    - 导出运行手册
  - 配置操作的输入和输出
    - 为操作提供输入数据
    - 定义操作的输出数据
  - 视觉对象设计体验中的错误处理
    - 出现错误时重试操作
    - 超时
    - 失败的操作

        - 取消的操作
        - 关键操作
        - 结束操作
    - 教程：使用视觉对象设计体验创建运行手册
        - 步骤 1：导航到视觉对象设计体验
        - 步骤 2：创建工作流程
        - 步骤 3：查看自动生成的代码
        - 步骤 4：运行新的运行手册
        - 第 5 步：清理
- 创作自动化运行手册
    - 识别您的使用案例
    - 设置开发环境
    - 开发运行手册内容
    - 示例 1：创建父子运行手册
        - 创建子运行手册
        - 创建父运行手册
    - 示例 2：脚本化运行手册
    - 其他运行手册示例
        - 部署 VPC 架构和 Microsoft Active Directory 域控制器
        - 从最新快照还原根卷
        - 创建 AMI 和跨区域副本
- 创建填充 AWS 资源的输入参数
- 正在使用文档生成器创建运行手册
    - 使用文档生成器创建运行手册
    - 创建运行脚本的运行手册
- 在运行手册中使用脚本
    - 使用运行手册的权限
    - 将脚本添加到运行手册
    - 运行手册的脚本限制
- 在运行手册中使用条件语句

- 使用 aws:branch 操作
  - 在运行手册中创建 aws:branch 步骤
    - 关于创建输出变量
  - 示例 aws:branch 运行手册
  - 使用运算符创建复杂的分支自动化
- 如何使用条件选项的示例
- 使用操作输出作为输入
  - 在运行手册中使用 JSONPath
- 为 Automation 创建 Webhook 集成
  - 创建集成（控制台）
  - 创建集成（命令行）
  - 为集成创建 Webhooks
- 处理运行手册中的超时

自动化运行手册的视觉对象设计体验

AWS Systems Manager Automation 提供低代码的视觉对象设计体验，可帮助您创建自动化运行手册。视觉对象设计体验提供拖放界面，可以选择添加自己的代码，这样您就可以更轻松地创建和编辑运行手册。借助视觉对象设计体验，您可以执行以下操作：

- 控制条件语句。
- 控制如何筛选或转换每个操作的输入和输出。
- 配置错误处理。
- 制作新运行手册的原型。
- 使用原型运行手册作为采用 AWS Toolkit for Visual Studio Code 进行本地开发的起点。

创建或编辑运行手册时，您可以从 Automation 控制台访问视觉对象设计体验。创建运行手册时，视觉对象设计体验会验证您的工作并自动生成代码。您可以查看生成的代码，也可以将其导出以供本地开发。完成后，您可以保存并运行运行手册以及在 Systems Manager Automation 控制台中检查结果。

开始前的准备工作

要使用视觉对象设计体验，您需要 AWS 账户 以及为您想要使用的任何资源提供正确权限的凭证。

在视觉对象设计体验中，自动化与 Amazon CodeGuru 安全防御工具集成，帮助您检测 Python 脚本中的安全策略违规行为和漏洞。要将此功能用于 `aws:executeScript` 操作，您的 AWS Identity and Access Management（IAM）policy 必须包含以下权限：

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codeguru-security:CreateUploadUrl",
                "codeguru-security:CreateScan",
                "codeguru-security:GetScan",
                "codeguru-security:GetFindings"
            ],
            "Resource": "*"
        }
    ]
}
```

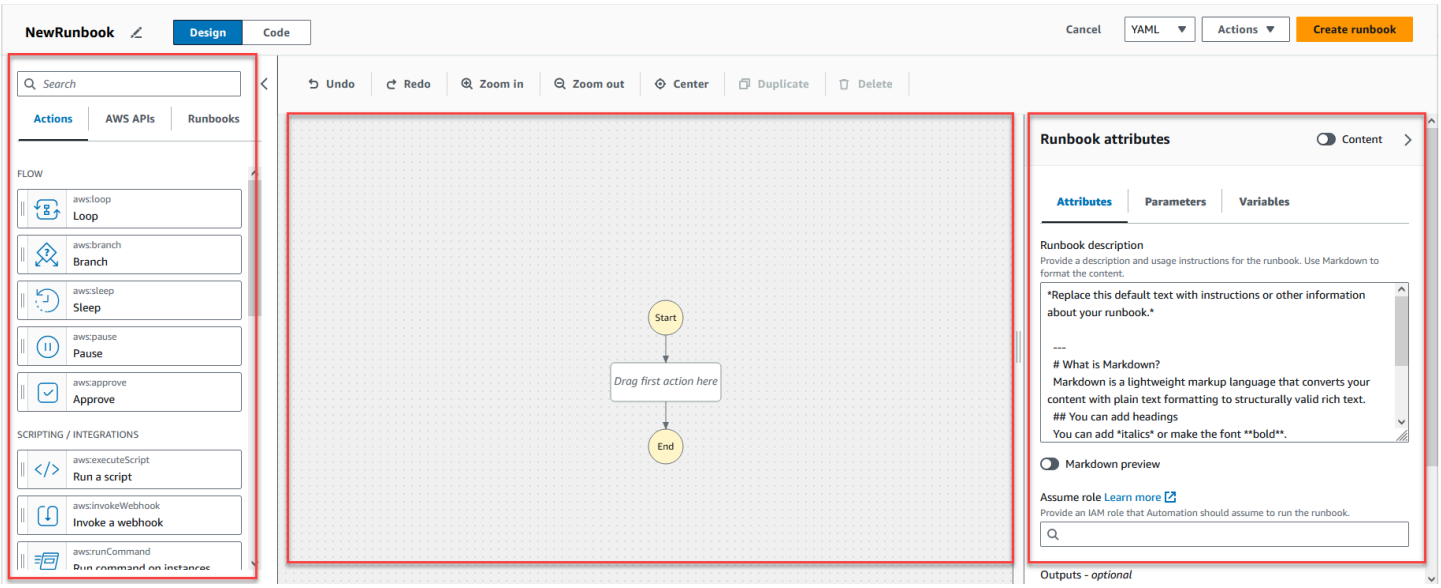
主题

- 视觉对象设计体验界面概述
- 使用视觉对象设计体验
- 配置操作的输入和输出
- 视觉对象设计体验中的错误处理
- 教程：使用视觉对象设计体验创建运行手册

视觉对象设计体验界面概述

Systems Manager Automation 的视觉对象设计体验是一种低代码的视觉对象工作流程设计器，可帮助您创建自动化运行手册。

通过界面组件的概述了解视觉对象设计体验：



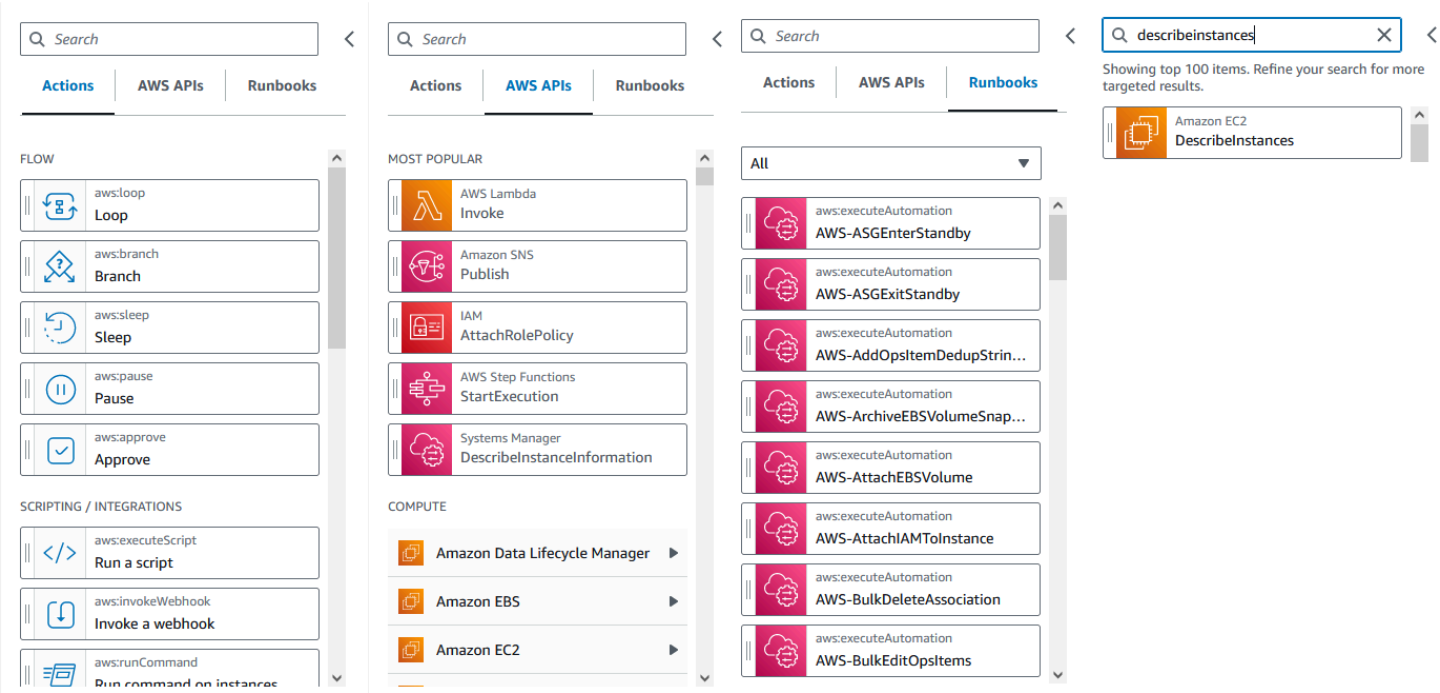
- 操作浏览器包含操作、AWS API 和运行手册选项卡。
- 在画布中，您可以将操作拖放到工作流程图中、更改操作顺序以及选择要配置或查看的操作。
- 在表单面板中，您可以查看和编辑在画布上选择的任何操作的属性。选择内容开关以查看运行手册的 YAML 或 JSON，并突出显示当前选定的操作。

当您需要帮助时，信息链接会打开一个包含上下文信息的面板。这些面板还包括指向 Systems Manager Automation 文档中相关主题的链接。

操作浏览器

在操作浏览器中，您可以选择要拖放到工作流程图中的操作。您可以使用操作浏览器顶部的搜索字段搜索所有操作。操作浏览器包含以下选项卡：

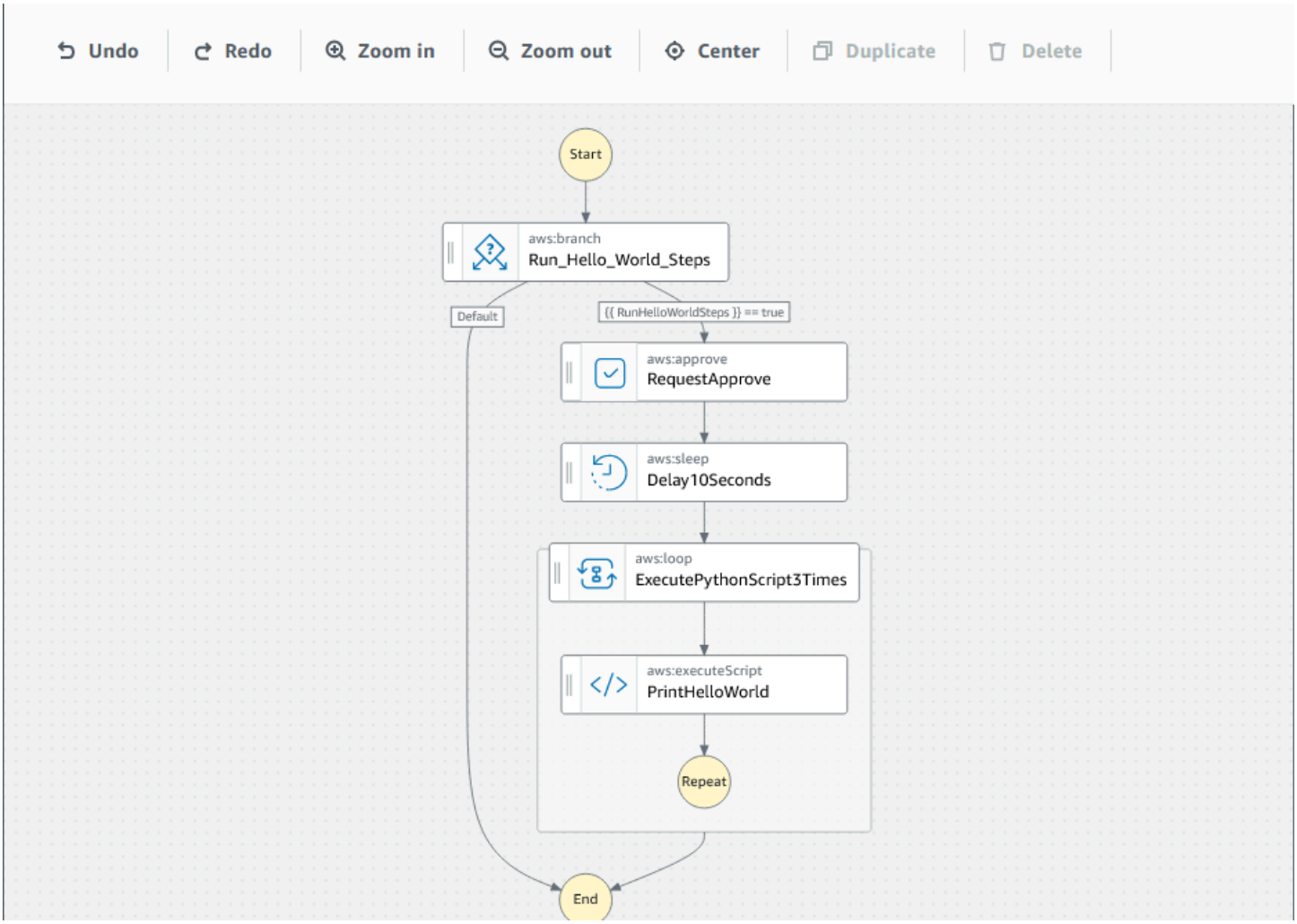
- 操作选项卡提供了自动化操作列表，您可以将这些操作拖放到画布中运行手册的工作流程图中。
- AWS API 选项卡提供了 AWS API 列表，您可以将这些 API 拖放到画布中运行手册的工作流程图中。
- 运行手册选项卡提供了几个随时可用、可重复使用的运行手册作为构建块，您可以将其用于各种用例。例如，您可以使用运行手册在工作流程中对 Amazon EC2 实例执行常见的修复任务，而无需重新创建相同的操作。



画布

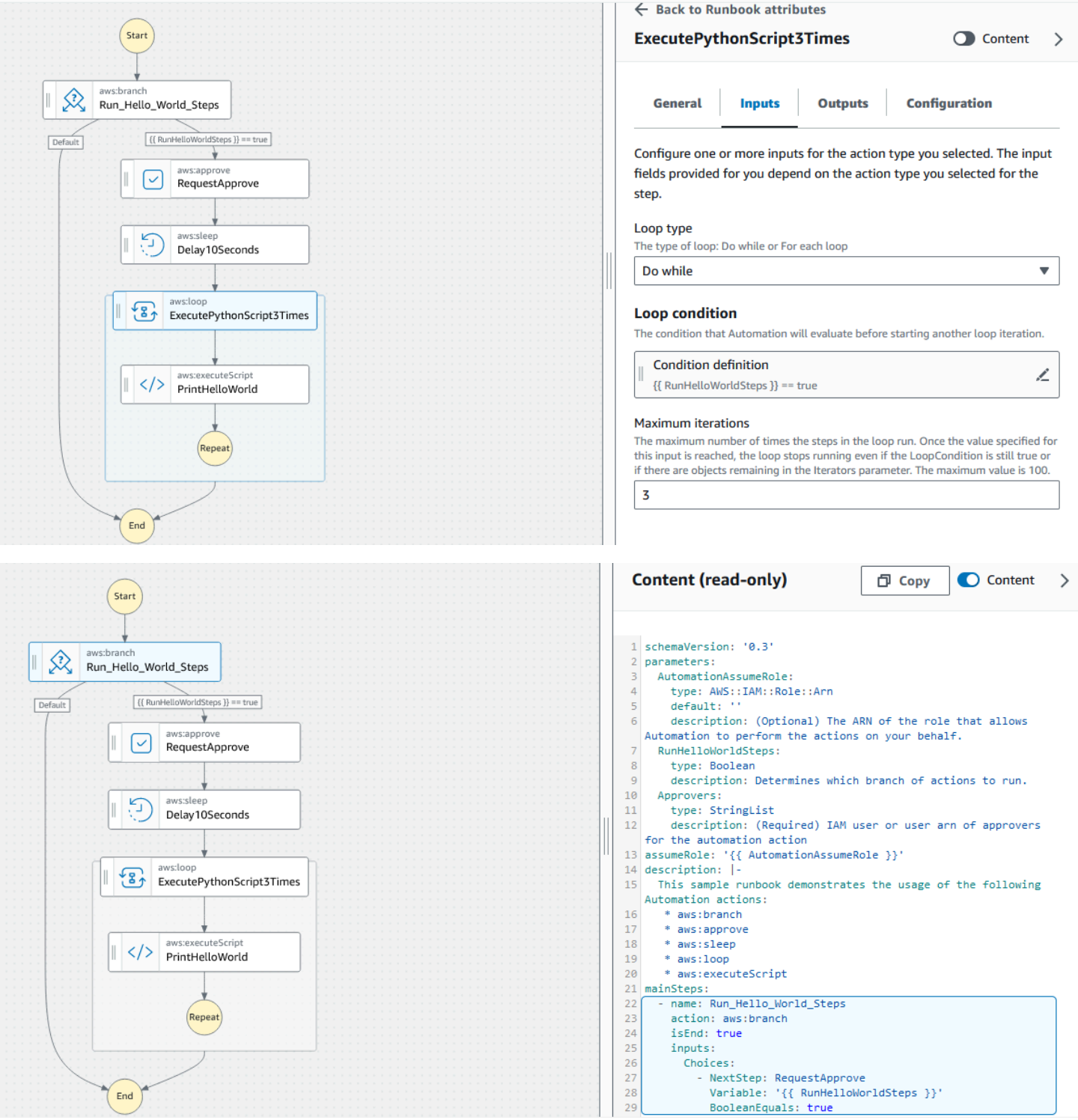
选择要添加到自动化中的操作后，将其拖动到画布并放入工作流程图中。您还可以拖放操作，将其移动到运行手册工作流程中的不同位置。如果您的工作流程很复杂，您可能无法在画布面板上查看其所有内容。使用画布顶部的控件来放大或缩小。要查看工作流程的不同部分，您可以在画布中拖动工作流程图。

在操作浏览器中拖动一项操作，将其放入运行手册的工作流程图中。有一条线将显示其在您工作流程中的放置位置。要更改操作的顺序，您可以将其拖动到工作流程中的其他位置。新操作已添加到您的工作流程中，其代码已自动生成。



表单

在运行手册工作流程中添加操作后，您可以对其进行配置以满足您的用例。选择希望配置的操作，您将在表单面板中看到其参数和选项。您还可以通过选择内容开关来查看 YAML 或 JSON 代码。与您所选操作关联的代码会突出显示。



键盘快捷键

视觉对象设计体验支持下表中所示的键盘快捷键。

键	键
数	数
快	快
捷	捷
键	键

Ctrl	Ctrl
键	键
上	上
次	次
的	的
操	操
作。	作。

Ctrl	Ctrl
代	代
Shift	Shift
上	上
次	次
的	的
操	操
作。	作。

Alt	Alt
工	工
C	C
作	作
流	流
程	程
置	置
于	于
画	画
布	布
中	中
心。	心。

删	删
除	除
键	键

> **键**
> **数**
> 快
> 捷
> 键
>
> 有
> 选
> 定
> 状
> 态。

> **删**
> **除**
> 所
> 有
> 选
> 定
> 状
> 态。

> **复**
> **制**
> 所
> 选
> 状
> 态。

使用视觉对象设计体验

学习使用视觉对象设计体验创建、编辑和运行运行手册工作流程。工作流程准备就绪后，您可以将其保存或导出。您还可以使用视觉对象设计体验进行快速原型制作。

创建运行手册工作流程

1. 登录 Systems Manager Automation 控制台。
2. 选择创建运行手册。
3. 在名称方框中，输入运行手册的名称，例如 *MyNewRunbook*。

4. 在设计和代码切换开关旁边，选择铅笔图标并输入运行手册的名称。

现在，您可以为新的运行手册设计工作流程。

设计运行手册

要使用视觉对象设计体验设计运行手册工作流程，您可以将自动化操作从操作浏览器拖动到画布中，将其放置在运行手册工作流程中所需的位置。您也可以通过将操作拖动到不同的位置来重新排序工作流程中的操作。当您将操作拖动到画布上时，工作流程中可以放置该操作的任何位置都会显示一条线。将操作放入画布后，其代码将自动生成并添加到运行手册的内容中。

如果您知道希望添加的操作的名称，请使用操作浏览器顶部的搜索框来查找该操作。

将操作放入画布后，使用右侧的表单面板对其进行配置。此面板包含您在画布上放置的每个自动化操作或 API 操作的常规、输入、输出和配置选项卡。例如，常规选项卡由以下部分组成：

- 步骤名称用于标识该步骤。为步骤名称指定唯一值。
- 描述可帮助您描述在运行手册的工作流程中操作正在执行的内容。

输入选项卡包含因操作而异的字段。例如，aws:executeScript 自动化操作由以下部分组成：

- 运行时系统是运行所提供脚本的语言。
- 处理程序是函数的名称。您必须确保在处理程序中定义的函数具有两个参数：events 和 context。PowerShell 运行时系统不支持此参数。
- 脚本是您想要在工作流程期间运行的嵌入式脚本。
- （可选）附件适用于可由操作调用的独立脚本或 .zip 文件。对于 JSON 运行手册，此参数为必需项。

输出选项卡可帮助您指定希望从操作中输出的值。您可以在工作流程的后续操作中引用输出值，也可以根据操作生成输出以用于日志记录。并非所有操作都将有输出选项卡，因为并非所有操作都支持输出。例如，aws:pause 操作就不支持输出。对于支持输出的操作，输出选项卡由以下部分组成：

- 名称是用于输出值的名称。您可以在工作流程的后续操作中引用输出。
- 选择器是以 "$." 开头的 JSONPath 表达式字符串，用于选择 JSON 元素中的一个或多个组件。
- 类型是输出值的数据类型。例如，String 或 Integer 数据类型。

配置选项卡包含所有自动化操作均可使用的属性和选项。该操作由以下部分组成：

- 最大尝试次数属性是操作失败时重试的次数。
- 超时秒数属性指定操作的超时值。
- 至关重要属性确定操作失败是否会停止整个自动化。
- 下一步属性确定自动化在运行手册中接下来要执行的操作。
- 失败时属性确定操作失败时自动化在运行手册中接下来要执行的操作。
- 取消时属性确定用户取消自动操作时自动化在运行手册中接下来要执行的操作。

要删除操作，您可以使用画布上方工具栏中的退格键，或者右键单击并选择删除操作。

随着工作流程的发展，其可能不适合画布。为帮助使工作流程适合画布，请尝试以下选项之一：

- 使用侧面板上的控件调整面板大小或关闭面板。
- 使用画布顶部的工具栏放大或缩小工作流程图。

更新运行手册

您可以通过创建新版本的运行手册来更新现有的运行手册工作流程。可以通过使用视觉对象设计体验或直接编辑代码来更新运行手册。要更新现有运行手册，请参照以下过程：

1. 登录 Systems Manager Automation 控制台。
2. 选择要更新的运行手册。
3. 选择 Create new version。
4. 视觉对象设计体验有两个窗格：代码窗格和视觉对象工作流程窗格。在视觉对象工作流程窗格中选择设计，以使用视觉对象设计体验编辑您的工作流程。完成后，选择创建新版本以保存更改并退出。
5. （可选）使用代码窗格以 YAML 或 JSON 格式编辑运行手册内容。

导出运行手册

要导出运行手册的工作流程 YAML 或 JSON 代码以及工作流程图，请参照以下过程：

1. 在文档控制台中选择您的运行手册。
2. 选择 Create new version。
3. 在操作下拉菜单中，选择是否要导出图表或运行手册，以及您喜欢的格式。

配置操作的输入和输出

每个自动化操作都会根据其收到的输入进行响应。在大多数情况下，您随后会将输出传递给后续操作。在视觉对象设计体验中，您可以在表单面板的输入和输出选项卡中配置操作的输入和输出数据。

有关如何定义和使用自动化操作输出的详细信息，请参阅 使用操作输出作为输入。

为操作提供输入数据

每个自动化操作都有一个或多个输入，您必须为其提供值。您为操作输入提供的值由操作接受的数据类型和格式决定。例如，这些 aws:sleep 操作要求 Duration 输入采用 ISO 8601 格式的字符串值。

通常，您在运行手册的工作流程中使用操作，这些操作将返回要在后续操作中使用的输出。请务必确保输入值正确，以避免运行手册的工作流程出现错误。输入值也很重要，因为其决定了操作是否返回预期的输出。例如，在使用 aws:executeAwsApi 操作时，您需要确保为 API 操作提供了正确的值。

定义操作的输出数据

一些自动化操作在执行其定义的操作后会返回输出。返回输出的操作要么具有预定义的输出，要么允许您自己定义输出。例如，aws:createImage 操作具有返回 ImageId 和 ImageState 的预定义输出。相比之下，通过 aws:executeAwsApi 操作，您可以定义您想要从指定 API 操作中获得的输出。因此，您可以从单个 API 操作中返回一个或多个值，以便在后续操作中使用。

为自动化操作定义自己的输出需要您指定输出的名称、数据类型和输出值。要继续使用 aws:executeAwsApi 操作作为示例，假设您正在从 Amazon EC2 调用 DescribeInstances API 操作。在此示例中，您想要返回或输出 Amazon EC2 实例的 State，并根据输出对运行手册的工作流程进行分支。您可以选择命名输出 **InstanceState**，然后使用 **String** 数据类型。

定义输出实际值的过程因操作而异。例如，如果您使用的是 aws:executeScript 操作，则必须在函数中使用 return 语句为输出提供数据。对于 aws:executeAwsApi、aws:waitForAwsResourceProperty 和 aws:assertAwsResourceProperty 等其他操作，则需要 Selector。Selector（或某些操作所指的 PropertySelector），是一个 JSONPath 字符串，用于处理来自 API 操作的 JSON 响应。了解 API 操作的 JSON 响应对象的结构至关重要，这样您才能为输出选择正确的值。使用前面提到 DescribeInstances API 操作，请参阅以下 JSON 响应示例：

```
{
  "reservationSet": {
    "item": {
      "reservationId": "r-1234567890abcdef0",
      "ownerId": 123456789012,
      "groupSet": "",
```

```
      "instancesSet": {
        "item": {
          "instanceId": "i-1234567890abcdef0",
          "imageId": "ami-bff32ccc",
          "instanceState": {
            "code": 16,
            "name": "running"
          },
          "privateDnsName": "ip-192-168-1-88.eu-west-1.compute.internal",
          "dnsName": "ec2-54-194-252-215.eu-west-1.compute.amazonaws.com",
          "reason": "",
          "keyName": "my_keypair",
          "amiLaunchIndex": 0,
          "productCodes": "",
          "instanceType": "t2.micro",
          "launchTime": "2018-05-08T16:46:19.000Z",
          "placement": {
            "availabilityZone": "eu-west-1c",
            "groupName": "",
            "tenancy": "default"
          },
          "monitoring": {
            "state": "disabled"
          },
          "subnetId": "subnet-56f5f000",
          "vpcId": "vpc-11112222",
          "privateIpAddress": "192.168.1.88",
          "ipAddress": "54.194.252.215",
          "sourceDestCheck": true,
          "groupSet": {
            "item": {
              "groupId": "sg-e4076000",
              "groupName": "SecurityGroup1"
            }
          },
          "architecture": "x86_64",
          "rootDeviceType": "ebs",
          "rootDeviceName": "/dev/xvda",
          "blockDeviceMapping": {
            "item": {
              "deviceName": "/dev/xvda",
              "ebs": {
                "volumeId": "vol-1234567890abcdef0",
                "status": "attached",
```

```
            "attachTime": "2015-12-22T10:44:09.000Z",
            "deleteOnTermination": true
          }
        }
      },
      "virtualizationType": "hvm",
      "clientToken": "xMcwG14507example",
      "tagSet": {
        "item": {
          "key": "Name",
          "value": "Server_1"
        }
      },
      "hypervisor": "xen",
      "networkInterfaceSet": {
        "item": {
          "networkInterfaceId": "eni-551ba000",
          "subnetId": "subnet-56f5f000",
          "vpcId": "vpc-11112222",
          "description": "Primary network interface",
          "ownerId": 123456789012,
          "status": "in-use",
          "macAddress": "02:dd:2c:5e:01:69",
          "privateIpAddress": "192.168.1.88",
          "privateDnsName": "ip-192-168-1-88.eu-west-1.compute.internal",
          "sourceDestCheck": true,
          "groupSet": {
            "item": {
              "groupId": "sg-e4076000",
              "groupName": "SecurityGroup1"
            }
          },
          "attachment": {
            "attachmentId": "eni-attach-39697adc",
            "deviceIndex": 0,
            "status": "attached",
            "attachTime": "2018-05-08T16:46:19.000Z",
            "deleteOnTermination": true
          },
          "association": {
            "publicIp": "54.194.252.215",
            "publicDnsName": "ec2-54-194-252-215.eu-west-1.compute.amazonaws.com",
            "ipOwnerId": "amazon"
          },
```

```
              "privateIpAddressesSet": {
                "item": {
                  "privateIpAddress": "192.168.1.88",
                  "privateDnsName": "ip-192-168-1-88.eu-west-1.compute.internal",
                  "primary": true,
                  "association": {
                    "publicIp": "54.194.252.215",
                    "publicDnsName": "ec2-54-194-252-215.eu-
west-1.compute.amazonaws.com",
                    "ipOwnerId": "amazon"
                  }
                }
              },
              "ipv6AddressesSet": {
                "item": {
                  "ipv6Address": "2001:db8:1234:1a2b::123"
                }
              }
            }
          },
          "iamInstanceProfile": {
            "arn": "arn:aws:iam::123456789012:instance-profile/AdminRole",
            "id": "ABCAJEDNCAA64SSD123AB"
          },
          "ebsOptimized": false,
          "cpuOptions": {
            "coreCount": 1,
            "threadsPerCore": 1
          }
        }
      }
    }
  }
}
```

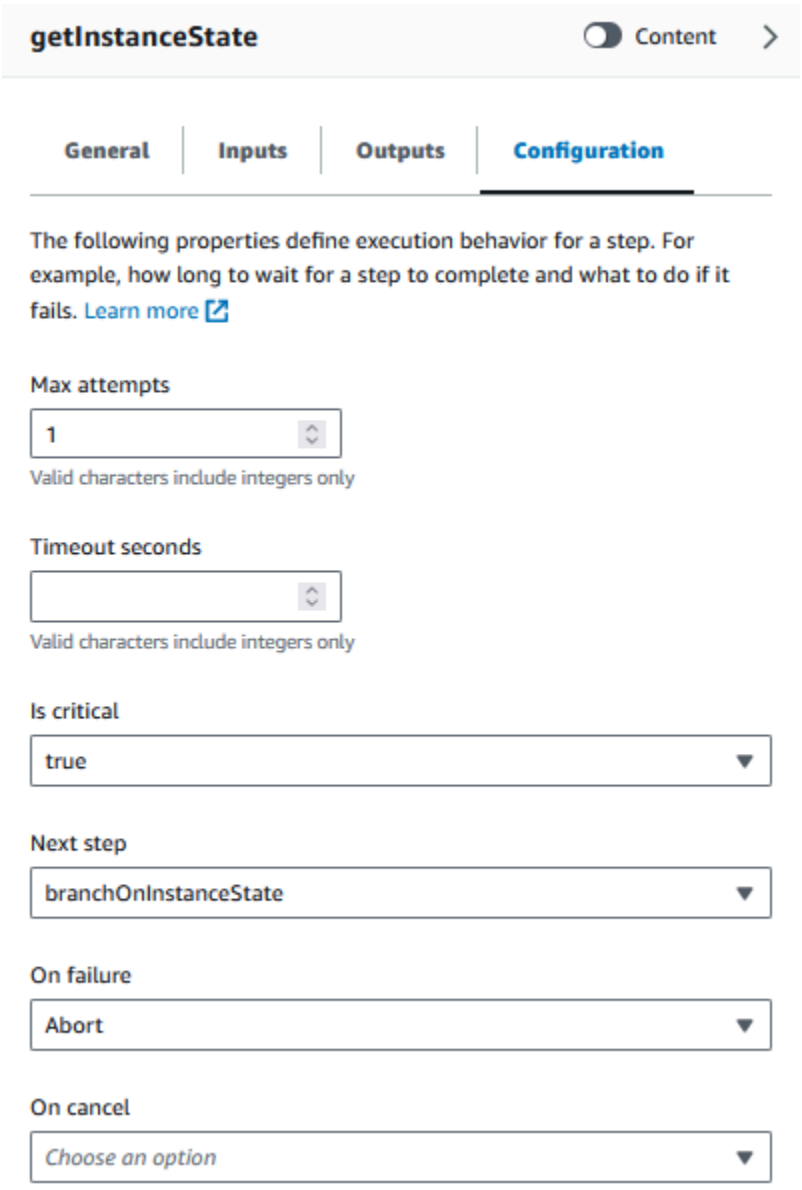
在 JSON 响应对象中，实例 State 嵌套在 Instances 对象中，而该对象嵌套在 Reservations 对象中。要返回实例 State 的值，请使用以下字符串作为 Selector，以便可以在输出中使用该值：**$.Reservations[0].Instances[0].State.Name**。

要在运行手册工作流程的后续操作中引用输出值，请使用以下格式：{{ *StepName*.*NameOfOutput* }}。例如，**{{ GetInstanceState.InstanceState }}**。在视觉对象设计体验中，您可以使用输入下拉菜单选择要在后续操作中使用的输出值。在后续操作中使用输出时，输出的数据类型必须与输入

的数据类型相匹配。在此示例中，InstanceState 输出为 String。因此，要在后续操作的输入中使用该值，输入必须接受 String。

视觉对象设计体验中的错误处理

默认情况下，当操作报告错误时，自动化会完全停止运行手册的工作流程。这是因为所有操作 onFailure 属性的默认值为 Abort。您可以配置自动化如何处理运行手册工作流程中错误。即使您配置了错误处理，某些错误仍可能导致自动化失败。有关更多信息，请参阅 Systems Manager 自动化故障排除。在视觉对象设计体验中，您可以在配置面板中配置错误处理。



出现错误时重试操作

要在出现错误时重试操作，请为最大尝试次数属性指定一个值。默认值是 1。如果您指定的值大于 1，则直到所有重试尝试失败后，才会将此操作视为失败。

超时

您可以为操作配置超时，以设置操作在失败之前可以运行的最大秒数。要配置超时，请在超时秒数属性中输入操作在失败之前应等待的秒数。如果达到超时并且操作的 Max attempts 值大于 1，则在重试完成之前该步骤不会被视为已超时。

失败的操作

默认情况下，当操作失败时，自动化会完全停止运行手册的工作流程。您可以通过为运行手册中操作的失败时属性指定替代值来修改此行为。如果您希望工作流程继续到运行手册中的下一步，请选择继续。如果您希望工作流程跳至运行手册中的其他后续步骤，请选择步骤，然后输入该步骤的名称。

取消的操作

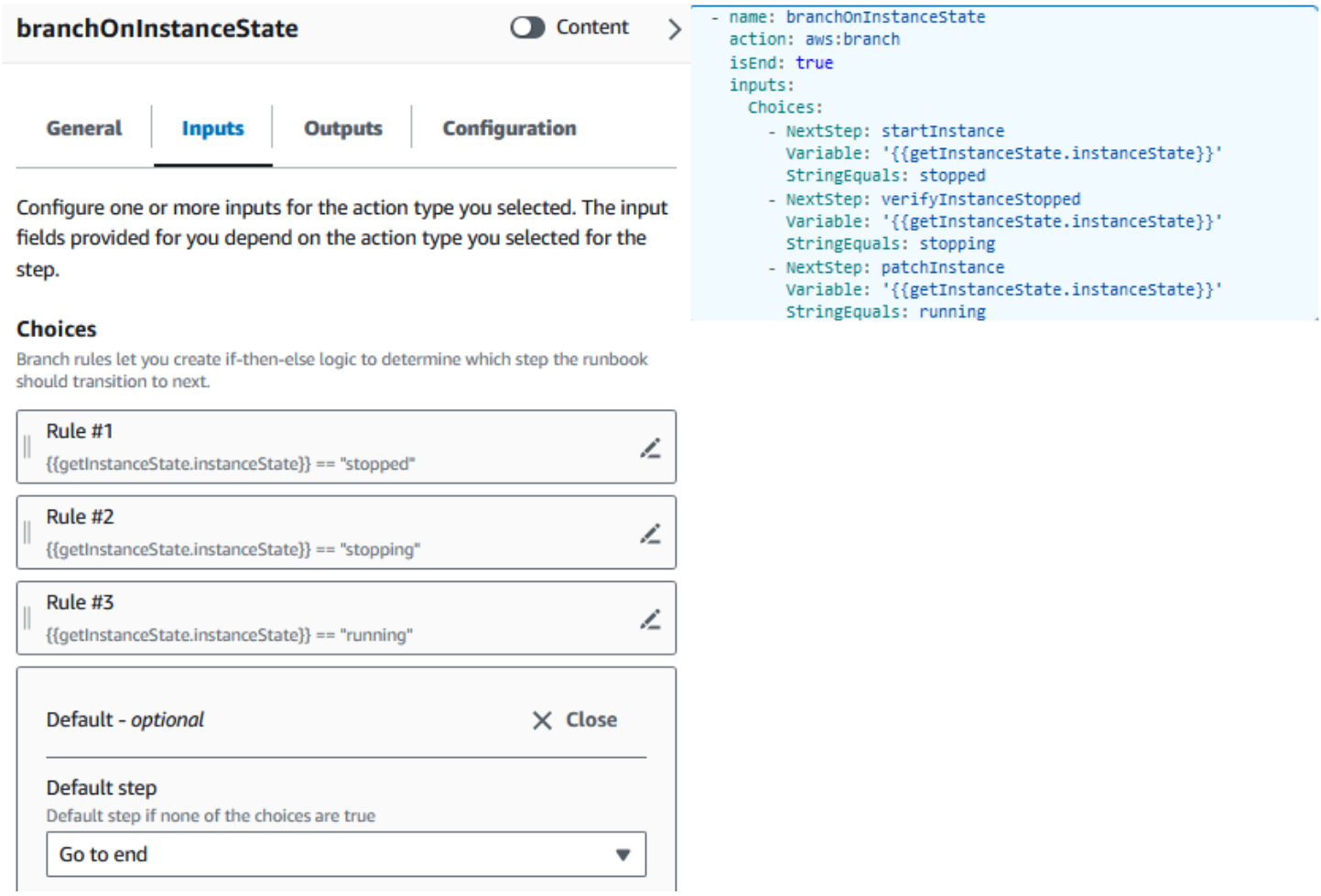
默认情况下，当用户取消操作时，自动化会完全停止运行手册的工作流程。您可以通过为运行手册中操作的取消时属性指定替代值来修改此行为。如果您希望工作流程跳至运行手册中的其他后续步骤，请选择步骤，然后输入该步骤的名称。

关键操作

您可以将某项操作指定为关键操作，这意味着其决定了自动化的总体报告状态。如果具有此指定的步骤失败，则无论其他操作是否成功，自动化都会将最终状态报告为 Failed。要将某项操作配置为关键，请将至关重要属性的默认值保留为 True。

结束操作

结束属性在指定操作结束时停止自动化。此属性的默认值为 false。如果您为操作配置此属性，则无论操作成功还是失败，自动化都会停止。此属性最常与 aws:branch 操作一起使用，以处理意外或未定义的输入值。以下示例显示了期望实例状态为 running、stopping 或 stopped 的运行手册。如果实例处于不同的状态，则自动化结束。



教程：使用视觉对象设计体验创建运行手册

在本教程中，您将学习使用 Systems Manager Automation 提供的视觉对象设计体验的基础知识。在视觉对象设计体验中，您可以创建使用多个操作的运行手册。您可以使用拖放功能在画布上排列操作。您还可以搜索、选择和配置这些操作。然后，您可以查看为运行手册工作流程自动生成的 YAML 代码、退出视觉对象设计体验、运行运行手册并查看执行详情。

本教程还向您展示了如何更新运行手册和查看新版本。在本教程的最后，您将执行清理步骤并删除运行手册。

完成本教程的学习后，您将知道如何使用视觉对象设计体验来创建运行手册。您还将了解如何更新、运行和删除运行手册。

> **Note**
>
> 在开始学习本教程之前，请确保完成 设置自动化。

主题

- 步骤 1：导航到视觉对象设计体验
- 步骤 2：创建工作流程
- 步骤 3：查看自动生成的代码
- 步骤 4：运行新的运行手册
- 第 5 步：清理

步骤 1：导航到视觉对象设计体验

1. 登录 Systems Manager Automation 控制台。
2. 选择创建自动化运行手册。

步骤 2：创建工作流程

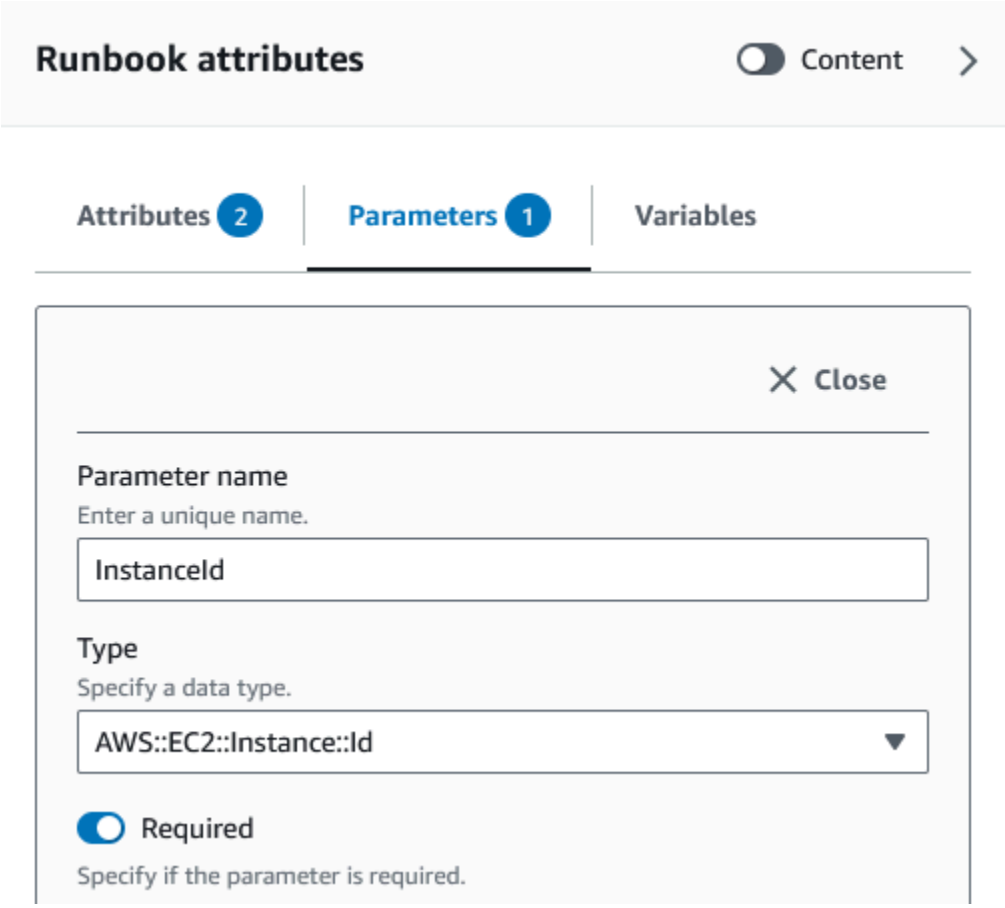
在视觉对象设计体验中，工作流程是画布上运行手册的图形表示。您可以使用视觉对象设计体验来定义、配置和检查运行手册的各个操作。

创建工作流程

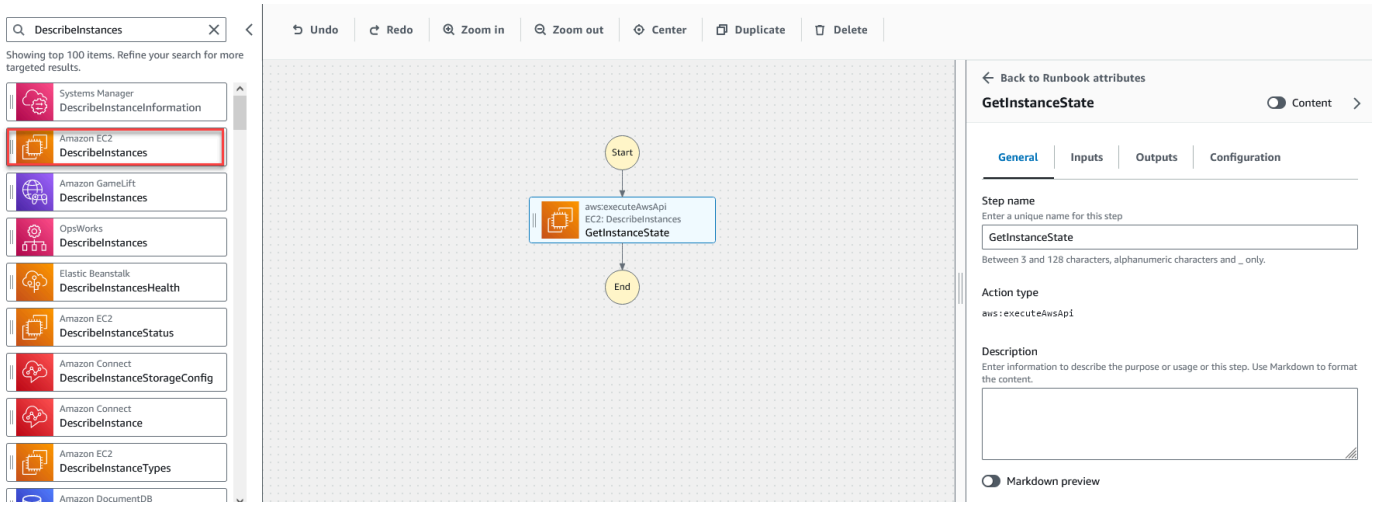
1. 在设计和代码切换开关旁边，选择铅笔图标并输入运行手册的名称。在本教程中，请输入 **`VisualDesignExperienceTutorial`**。



2. 在表单面板的文档属性部分，展开输入参数下拉菜单，然后选择添加参数。
   a. 在参数名称字段中，输入 **`InstanceId`**。
   b. 在类型下拉菜单中，选择 AWS::EC2::Instance。
   c. 选择必需开关。



3. 在 AWS API 浏览器中，在搜索栏中输入 **`DescribeInstances`**。
4. 将 Amazon EC2 – DescribeInstances 操作拖动到空白画布上。
5. 对于步骤名称，请输入一个值。在本教程中，您可以使用名称 **`GetInstanceState`**。



   a. 展开其他输入下拉菜单，然后在输入名称字段中输入 **`InstanceIds`**。

b. 选择输入选项卡。

c. 在输入值字段中，选择 **InstanceId** 文档输入。这将引用在过程开始时创建的输入参数的值。鉴于 DescribeInstances 操作的 InstanceIds 输入接受 StringList 值，必须将 InstanceId 输入用方括号括起来。输入值的 YAML 应与以下内容匹配：**['{{ InstanceId }}']**。

d. 在输出选项卡中，请选择添加输出，然后在名称字段中输入 **InstanceState**。

e. 在选择器字段中，请输入 **$.Reservations[0].Instances[0].State.Name**。

f. 在类型下拉菜单中，请选择字符串。

6. 从操作浏览器中拖动分支操作，然后将其放入 **GetInstanceState** 步骤下方。

7. 对于步骤名称，请输入一个值。在本教程中，请使用名称 **BranchOnInstanceState**。

   要定义分支逻辑，请执行以下操作：

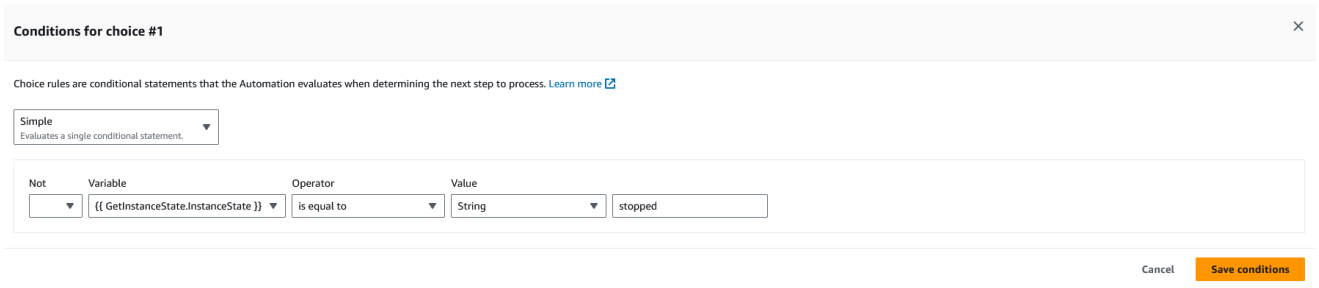
   a. 在画布上选择 **Branch** 状态。然后，在输入和选择下，选择铅笔图标以编辑规则 #1。

   b. 选择添加条件。

   c. 在规则 #1 的条件对话框中，从变量下拉菜单中选择 **GetInstanceState.InstanceState** 步骤输出。

   d. 对于运算符，请选择等于。

   e. 对于值，请从下拉列表中选择字符串。输入 **stopped**。



   f. 选择保存条件。

   g. 选择添加新的选择规则。

   h. 为规则 #2 选择添加条件。

   i. 在规则 #2 的条件对话框中，从变量下拉菜单中选择 **GetInstanceState.InstanceState** 步骤输出。

   j. 对于运算符，请选择等于。

   k. 对于值，请从下拉列表中选择字符串。输入 **stopping**。

   l. 选择保存条件。

m. 选择添加新的选择规则。

n. 对于规则 #3，请选择添加条件。

o. 在规则 #3 的条件对话框中，从变量下拉菜单中选择 **GetInstanceState.InstanceState** 步骤输出。

p. 对于运算符，请选择等于。

q. 对于值，请从下拉列表中选择字符串。输入 **running**。

r. 选择保存条件。

s. 在默认规则中，选择转到结尾作为默认步骤。

8. 将更改实例状态操作拖动到 {{ GetInstanceState.InstanceState }} == "stopped" 条件下的将操作拖动到此处空白框中。

a. 对于步骤名称，请输入 **StartInstance**。

b. 在输入选项卡的实例 ID 下，从下拉菜单中选择 InstanceId 文档输入值。

c. 对于期望状态，请指定 **running**。

9. 将等待 AWS 资源操作拖动到 {{ GetInstanceState.InstanceState }} == "stopping" 条件下的将操作拖动到此处空白框中。

10. 对于步骤名称，请输入一个值。在本教程中，请使用名称 **WaitForInstanceStop**。

a. 对于服务字段，请选择 Amazon EC2。

b. 对于 API 字段，请选择 DescribeInstances。

c. 对于属性选择器字段，请输入 **$.Reservations[0].Instances[0].State.Name**。

d. 对于期望值参数，请输入 **["stopped"]**。

e. 在 WaitForInstanceStop 操作的配置选项卡中，从下一步下拉菜单中选择 StartInstance。

11. 将在实例上运行命令操作拖动到 {{ GetInstanceState.InstanceState }} == "running" 条件下的将操作拖动到此处空白框中。

12. 对于步骤名称，请输入 **SayHello**。

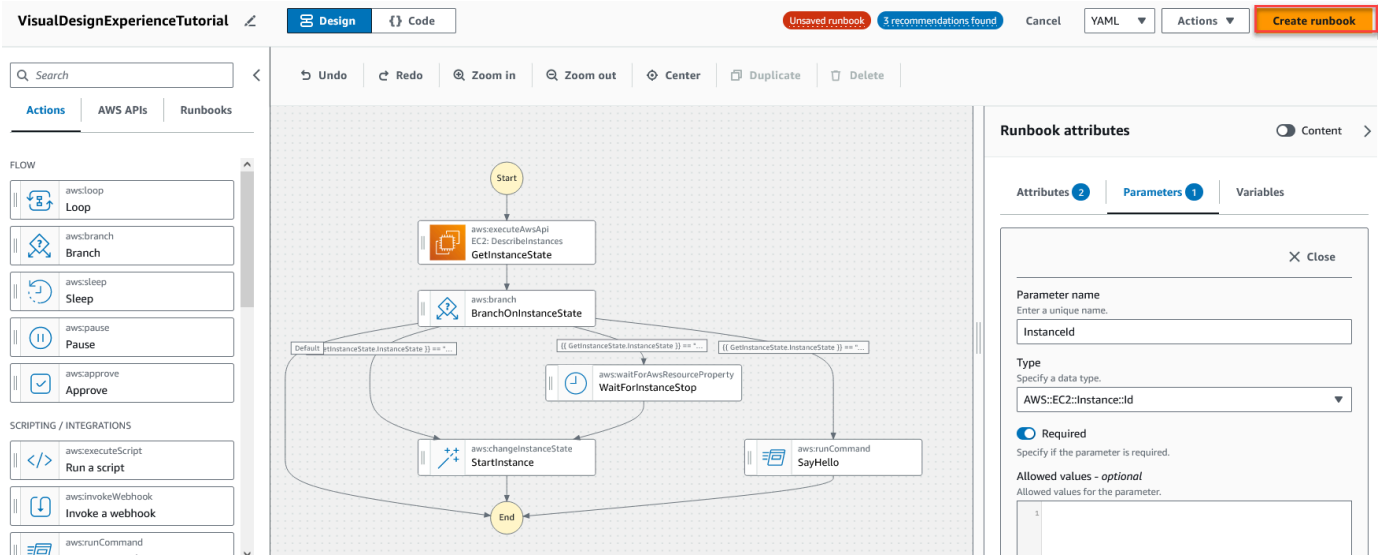
a. 在输入选项卡中，输入 **AWS-RunShellScript** 作为文档名称参数。

b. 对于 InstanceId，请从下拉菜单中选择 InstanceId 文档输入值。

c. 展开其他输入下拉菜单，在输入名称下拉菜单中，选择参数。

d. 在输入值字段中，请输入 **{"commands": "echo 'Hello World'"}**。

13. 在画布中查看已完成的运行手册，然后选择创建运行手册以保存教程运行手册。



步骤 3：查看自动生成的代码

当您将操作从操作浏览器拖放到画布上时，视觉对象设计体验会自动实时编写运行手册的 YAML 或 JSON 内容。您可以查看和编辑此代码。要查看自动生成的代码，在设计和代码切换开关中选择代码。

步骤 4：运行新的运行手册

创建运行手册后，您可以运行自动化。

运行新的自动化运行手册

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择自动化，然后选择执行自动化。
3. 在自动化文档列表中，请选择运行手册。在文档类别窗格中选择一个或多个选项，以便根据 SSM 文档的用途对其进行筛选。要查看您拥有的运行手册，请选择我拥有的选项卡。要查看与您的账户共享的运行手册，请选择与我共享选项卡。要查看所有运行手册，请选择所有文档选项卡。

> **Note**
>
> 您可以通过选择运行手册名称来查看有关该手册的信息。

4. 在文档详细信息部分中，验证文档版本已设置为要运行的版本。系统包括以下版本选项：
   - 运行时的默认版本 – 如果定期更新自动化运行手册并分配新的默认版本，请选择此选项。

- 运行时的最新版本 – 如果定期更新自动化运行手册并且想要运行最新更新的版本，请选择此选项。
- 1（默认）– 选择此选项可执行文档的第一个版本，即默认版本。

5. 选择下一步。
6. 在执行自动化运行手册部分，请选择简单执行。
7. 在 输入参数 部分中，指定所需的输入。或者，您也可以从 AutomationAssumeRole 列表选择一个 IAM 服务角色。
8. （可选）选择一个 Amazon CloudWatch 警报应用于您的自动化，以便进行监控。要将 CloudWatch 警报附加到自动化，启动自动化的 IAM 主体必须具有 `iam:createServiceLinkedRole` 操作的权限。有关 CloudWatch 警报的更多信息，请参阅使用 Amazon CloudWatch 警报。如果您的警报激活，自动化将停止。如果使用 AWS CloudTrail，您将在跟踪中看到 API 调用。
9. 选择执行。

第 5 步：清理

删除您的运行手册

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择文档。
3. 选择我拥有的选项卡。
4. 找到 VisualDesignExperienceTutorial 运行手册。
5. 在文档卡页面上选择按钮，然后从操作下拉菜单中选择删除文档。

创作自动化运行手册

Automation 是 AWS Systems Manager 中的一项工具，其中的每个运行手册定义了一个自动化。自动化运行手册定义了在自动化过程中执行的操作。在运行手册内容中，您可以定义 Systems Manager 对托管实例和 AWS 资源执行的输入参数、输出和操作。

自动化包含几个预定义的运行手册，您可以用它们来执行常见任务，例如重启一个或多个 Amazon Elastic Compute Cloud (Amazon EC2) 实例，或创建 Amazon Machine Image (AMI)。但是，您的使用案例可能会超出预定义运行手册的功能。如果是这种情况，您可以创建自己的运行手册并根据需要修改它们。

运行手册由自动化操作、这些操作的参数以及您指定的输入参数组成。运行手册的内容是以 YAML 或 JSON 格式编写的。如果不熟悉 YAML 或 JSON，建议使用视觉设计器，或者在尝试创作自己的运行手册之前了解更多关于任一标记语言的信息。有关视觉设计器的更多信息，请参阅 自动化运行手册的视觉对象设计体验。

以下部分将帮助您创作第一个运行手册。

识别您的使用案例

创作运行手册的第一步是确定您的使用案例。例如，您计划了 AWS-CreateImage 运行手册，以便每天在您的所有生产 Amazon EC2 实例上运行。月底时，您决定拥有的映像数量超过恢复点所需的映像。接下来，您希望在创建新 AMI 时，自动删除 Amazon EC2 实例的最早 AMI。为了实现此目的，您可以创建执行以下操作的新运行手册：

1. 运行 aws:createImage 操作，并在映像描述中指定实例 ID。
2. 运行 aws:waitForAwsResourceProperty 操作来轮询图像的状态，直到此状态为 available。
3. 在图像状态变为 available 后，aws:executeScript 操作会运行一个自定义 Python 脚本，用于收集与 Amazon EC2 实例关联的所有映像的 ID。脚本使用您在创建时指定的映像描述中的实例 ID 进行筛选，完成此操作。然后，脚本根据图像的 creationDate 对图像 ID 列表进行排序，并输出最早的 AMI。
4. 最后，aws:deleteImage 操作运行，使用上一步输出中的 ID 删除最早的 AMI。

在这种情况下，您已经使用 AWS-CreateImage 运行手册，但发现您的使用案例需要更大的灵活性。这是一种常见的情况，因为运行手册和自动化操作之间可能存在重叠。因此，您可能需要调整用于解决您的使用案例的运行手册或操作。

例如，aws:executeScript 和 aws:invokeLambdaFunction 操作都允许您在自动化过程中运行自定义脚本。要在它们之间进行选择，您可能更喜欢 aws:invokeLambdaFunction，因为它还支持其他运行时语言。但是，您可能更喜欢 aws:executeScript，因为它允许您直接在 YAML 运行手册中创作脚本内容，并提供脚本内容作为 JSON 运行手册的附件。您也可以考虑 aws:executeScript，因为其 AWS Identity and Access Management (IAM) 设置更简单。因为它使用 AutomationAssumeRole 提供的权限，aws:executeScript 不需要使用额外的 AWS Lambda 函数执行角色。

在任何给定的情况下，一个操作可能会比另一个操作提供更大的灵活性或更多功能。因此，我们建议您查看要使用的运行手册或操作的可用输入参数，以确定哪些参数最适合您的使用案例和首选项。

设置开发环境

确定您的使用案例以及要在运行手册中使用的预定义运行手册或自动化操作后，请为运行手册的内容设置开发环境。要开发您的运行手册内容，我们建议使用 AWS Toolkit for Visual Studio Code 而不是 Systems Manager 文档控制台。

Toolkit for VS Code 是 Visual Studio 代码（VS 代码）的开源扩展，提供了比 Systems Manager 文档控制台更多的功能。有用的功能包括针对 YAML 和 JSON 的模式验证、自动化操作类型的代码片段以及对各种 YAML 和 JSON 格式选项的自动完成支持。

有关安装 Toolkit for VS Code 的更多信息，请参阅安装 AWS Toolkit for Visual Studio Code。有关使用适用于 Toolkit for VS Code 开发运行手册的更多信息，请参阅 AWS Toolkit for Visual Studio Code 用户指南中的使用 Systems Manager 运行手册

开发运行手册内容

确定使用案例并设置环境后，您便做好了开发适用于运行手册内容的准备。您的使用案例和首选项在很大程度上决定了您在运行手册内容中使用的自动化操作或运行手册。与允许您完成类似任务的其他操作相比，某些操作仅支持输入参数的子集。其他操作具有特定的输出，例如 `aws:createImage`，其中一些操作允许您定义自己的输出，例如 `aws:executeAwsApi`。

如果您不确定如何在运行手册中使用特定操作，我们建议您查看 Systems Manager 自动化操作参考 中的操作对应条目。我们还建议您查看预定义运行手册的内容，以查看有关如何使用这些操作的实例。有关运行手册的实际应用程序的更多实例，请参阅 其他运行手册示例。

为了展示运行手册内容在简单性和灵活性方面的差异，以下教程提供了如何分阶段修补 Amazon EC2 实例组的示例：

- the section called “示例 1：创建父子运行手册” - 在此示例中，父子关系中使用了两个运行手册。父运行手册启动子运行手册的速率控制自动化。
- the section called “示例 2：脚本化运行手册” - 此示例演示如何通过将内容压缩到单个运行手册中并使用运行手册中的脚本来完成示例 1 的相同任务。

示例 1：创建父子运行手册

以下示例演示如何创建两个运行手册，以便分阶段修补已标记的 Amazon Elastic Compute Cloud (Amazon EC2) 实例组。这些运行手册用于父子关系，其中父运行手册用于启动子运行手册的速率控制自动化。有关使用速率控制自动化的更多信息，请参阅 大规模运行自动化操作。有关此示例中使用的自动化操作的更多信息，请参阅 Systems Manager 自动化操作参考。

创建子运行手册

此示例运行手册解决以下情形。Emily 是 AnyCompany Consultants, LLC 的系统工程师。她需要为托管主数据库和辅助数据库的 Amazon Elastic Compute Cloud (Amazon EC2) 实例组配置补丁程序。应用程序每天 24 小时访问这些数据库，因此两个数据库实例中必须有一个始终可用。

她确定分阶段修补实例是最佳方法。首先将对数据库实例的主组进行修补，然后修补数据库实例的辅助组。此外，为了避免由于使之前已停止的实例运行而产生额外的成本，Emily 希望在进行修补之前将修补的实例返回到其原始状态。

Emily 通过与实例关联的标签来标识数据库实例的主组和辅助组。她决定创建一个父运行手册，用于启动子运行手册的速率控制自动化。通过这种方法，她可以将与数据库实例的主组和辅助组关联的标签设置为目标，并管理子自动化的并发性。查看了可用于修补的 Systems Manager (SSM) 文档后，她选择了 `AWS-RunPatchBaseline` 文档。通过使用此 SSM 文档，她的同事可以在修补操作完成后查看相关的修补程序合规性信息。

为了开始创建运行手册内容，Emily 查看了可用的自动化操作，并开始为子运行手册创作内容，如下所示：

1. 首先，她为运行手册的架构和描述提供值，并定义子运行手册的输入参数。

   通过使用 `AutomationAssumeRole` 参数，Emily 和她的同事可以使用现有的 IAM 角色，以允许自动化代表他们执行运行手册中的操作。Emily 使用 `InstanceId` 参数来确定应修补的实例。（可选）`Operation`、`RebootOption` 和 `SnapshotId` 参数可用于为 `AWS-RunPatchBaseline` 的文档参数提供值。为了防止向这些运行手册参数提供无效值，她根据需要定义了 `allowedValues`。

   YAML

```
schemaVersion: '0.3'
description: 'An example of an Automation runbook that patches groups of Amazon
 EC2 instances in stages.'
assumeRole: '{{AutomationAssumeRole}}'
parameters:
  AutomationAssumeRole:
    type: String
    description: >-
      '(Optional) The Amazon Resource Name (ARN) of the IAM role that allows
 Automation to perform the
      actions on your behalf. If no role is specified, Systems Manager
      Automation uses your IAM permissions to operate this runbook.'
    default: ''
```

```
  InstanceId:
    type: String
    description: >-
      '(Required) The instance you want to patch.'
  SnapshotId:
    type: String
    description: '(Optional) The snapshot ID to use to retrieve a patch baseline
 snapshot.'
    default: ''
  RebootOption:
    type: String
    description: '(Optional) Reboot behavior after a patch Install operation. If
 you choose NoReboot and patches are installed, the instance is marked as non-
compliant until a subsequent reboot and scan.'
    allowedValues:
      - NoReboot
      - RebootIfNeeded
    default: RebootIfNeeded
  Operation:
    type: String
    description: '(Optional) The update or configuration to perform on the
 instance. The system checks if patches specified in the patch baseline are
 installed on the instance. The install operation installs patches missing from
 the baseline.'
    allowedValues:
      - Install
      - Scan
    default: Install
```

JSON

```
{
   "schemaVersion":"0.3",
   "description":"An example of an Automation runbook that patches groups of
 Amazon EC2 instances in stages.",
   "assumeRole":"{{AutomationAssumeRole}}",
   "parameters":{
      "AutomationAssumeRole":{
         "type":"String",
         "description":"(Optional) The Amazon Resource Name (ARN) of the IAM role
 that allows Automation to perform the actions on your behalf. If no role is
 specified, Systems Manager Automation uses your IAM permissions to operate this
 runbook.",
```

```
         "default":""
      },
      "InstanceId":{
         "type":"String",
         "description":"(Required) The instance you want to patch."
      },
      "SnapshotId":{
         "type":"String",
         "description":"(Optional) The snapshot ID to use to retrieve a patch
 baseline snapshot.",
         "default":""
      },
      "RebootOption":{
         "type":"String",
         "description":"(Optional) Reboot behavior after a patch Install
 operation. If you choose NoReboot and patches are installed, the instance is
 marked as non-compliant until a subsequent reboot and scan.",
         "allowedValues":[
            "NoReboot",
            "RebootIfNeeded"
         ],
         "default":"RebootIfNeeded"
      },
      "Operation":{
         "type":"String",
         "description":"(Optional) The update or configuration to perform on
 the instance. The system checks if patches specified in the patch baseline are
 installed on the instance. The install operation installs patches missing from
 the baseline.",
         "allowedValues":[
            "Install",
            "Scan"
         ],
         "default":"Install"
      }
   }
},
```

2. 定义完顶层元素后，Emily 继续创作构成运行手册的 mainSteps 的操作。第一步使用 aws:executeAwsApi 操作输出 InstanceId 输入参数中指定的目标实例的当前状态。此操作的输出将用于后续操作。

YAML

```
mainSteps:
  - name: getInstanceState
    action: 'aws:executeAwsApi'
    onFailure: Abort
    inputs:
      inputs:
      Service: ec2
      Api: DescribeInstances
      InstanceIds:
        - '{{InstanceId}}'
    outputs:
      - Name: instanceState
        Selector: '$.Reservations[0].Instances[0].State.Name'
        Type: String
    nextStep: branchOnInstanceState
```

JSON

```
"mainSteps":[
      {
         "name":"getInstanceState",
         "action":"aws:executeAwsApi",
         "onFailure":"Abort",
         "inputs":{
            "inputs":null,
            "Service":"ec2",
            "Api":"DescribeInstances",
            "InstanceIds":[
               "{{InstanceId}}"
            ]
         },
         "outputs":[
            {
               "Name":"instanceState",
               "Selector":"$.Reservations[0].Instances[0].State.Name",
               "Type":"String"
            }
         ],
         "nextStep":"branchOnInstanceState"
      },
```

3. Emily 没有手动启动和跟踪需要修补的每个实例的原始状态，而是根据目标实例的状态使用上一操作的输出对自动化进行分支。这允许自动化根据 aws:branch 操作中定义的条件运行不同的步骤，并提高了自动化的整体效率，无需人工干预。

   如果实例状态已为 running，则自动化将继续使用 aws:runCommand 操作，利用 AWS-RunPatchBaseline 文档修补实例。

   如果实例的状态为 stopping，则自动化使用 aws:waitForAwsResourceProperty 操作轮询实例以达到 stopped 状态，使用 executeAwsApi 操作启动实例，并轮询实例以达到 running 状态，然后再修补实例。

   如果实例的状态为 stopped，则自动化将启动实例并轮询实例以达到 running 状态，然后再使用相同的操作修补实例。

   YAML

```
- name: branchOnInstanceState
    action: 'aws:branch'
    onFailure: Abort
    inputs:
      Choices:
        - NextStep: startInstance
          Variable: '{{getInstanceState.instanceState}}'
          StringEquals: stopped
         - NextStep: verifyInstanceStopped
          Variable: '{{getInstanceState.instanceState}}'
          StringEquals: stopping
         - NextStep: patchInstance
          Variable: '{{getInstanceState.instanceState}}'
          StringEquals: running
    isEnd: true
  - name: startInstance
    action: 'aws:executeAwsApi'
    onFailure: Abort
    inputs:
      Service: ec2
      Api: StartInstances
      InstanceIds:
        - '{{InstanceId}}'
    nextStep: verifyInstanceRunning
  - name: verifyInstanceRunning
    action: 'aws:waitForAwsResourceProperty'
    timeoutSeconds: 120
```

```
    inputs:
      Service: ec2
      Api: DescribeInstances
      InstanceIds:
        - '{{InstanceId}}'
      PropertySelector: '$.Reservations[0].Instances[0].State.Name'
      DesiredValues:
        - running
    nextStep: patchInstance
  - name: verifyInstanceStopped
    action: 'aws:waitForAwsResourceProperty'
    timeoutSeconds: 120
    inputs:
      Service: ec2
      Api: DescribeInstances
      InstanceIds:
        - '{{InstanceId}}'
      PropertySelector: '$.Reservations[0].Instances[0].State.Name'
      DesiredValues:
        - stopped
      nextStep: startInstance
  - name: patchInstance
    action: 'aws:runCommand'
    onFailure: Abort
    timeoutSeconds: 5400
    inputs:
      DocumentName: 'AWS-RunPatchBaseline'
      InstanceIds:
      - '{{InstanceId}}'
      Parameters:
        SnapshotId: '{{SnapshotId}}'
        RebootOption: '{{RebootOption}}'
        Operation: '{{Operation}}'
```

JSON

```
{
        "name":"branchOnInstanceState",
        "action":"aws:branch",
        "onFailure":"Abort",
        "inputs":{
           "Choices":[
              {
```

```
                "NextStep":"startInstance",
                "Variable":"{{getInstanceState.instanceState}}",
                "StringEquals":"stopped"
             },
             {
                "Or":[
                   {
                      "Variable":"{{getInstanceState.instanceState}}",
                      "StringEquals":"stopping"
                   }
                ],
                "NextStep":"verifyInstanceStopped"
             },
             {
                "NextStep":"patchInstance",
                "Variable":"{{getInstanceState.instanceState}}",
                "StringEquals":"running"
             }
          ]
       },
       "isEnd":true
    },
    {
       "name":"startInstance",
       "action":"aws:executeAwsApi",
       "onFailure":"Abort",
       "inputs":{
          "Service":"ec2",
          "Api":"StartInstances",
          "InstanceIds":[
             "{{InstanceId}}"
          ]
       },
       "nextStep":"verifyInstanceRunning"
    },
    {
       "name":"verifyInstanceRunning",
       "action":"aws:waitForAwsResourceProperty",
       "timeoutSeconds":120,
       "inputs":{
          "Service":"ec2",
          "Api":"DescribeInstances",
          "InstanceIds":[
             "{{InstanceId}}"
```

```
            ],
            "PropertySelector":"$.Reservations[0].Instances[0].State.Name",
            "DesiredValues":[
               "running"
            ]
         },
         "nextStep":"patchInstance"
      },
      {
         "name":"verifyInstanceStopped",
         "action":"aws:waitForAwsResourceProperty",
         "timeoutSeconds":120,
         "inputs":{
            "Service":"ec2",
            "Api":"DescribeInstances",
            "InstanceIds":[
               "{{InstanceId}}"
            ],
            "PropertySelector":"$.Reservations[0].Instances[0].State.Name",
            "DesiredValues":[
               "stopped"
            ],
            "nextStep":"startInstance"
         }
      },
      {
         "name":"patchInstance",
         "action":"aws:runCommand",
         "onFailure":"Abort",
         "timeoutSeconds":5400,
         "inputs":{
            "DocumentName":"AWS-RunPatchBaseline",
            "InstanceIds":[
               "{{InstanceId}}"
            ],
            "Parameters":{
               "SnapshotId":"{{SnapshotId}}",
               "RebootOption":"{{RebootOption}}",
               "Operation":"{{Operation}}"
            }
         }
      },
```

4. 修补操作完成后，Emily 希望自动化将目标实例返回到自动化开始之前的状态。她通过再次使用第一个动作的输出来完成此操作。自动化根据目标实例的原始状态，使用 aws:branch 操作进行分支。如果实例之前处于 running 以外的任何状态，实例将停止。否则，如果实例状态为 running，则自动化结束。

YAML

```
- name: branchOnOriginalInstanceState
    action: 'aws:branch'
    onFailure: Abort
    inputs:
      Choices:
        - NextStep: stopInstance
          Not:
            Variable: '{{getInstanceState.instanceState}}'
            StringEquals: running
    isEnd: true
  - name: stopInstance
    action: 'aws:executeAwsApi'
    onFailure: Abort
    inputs:
      Service: ec2
      Api: StopInstances
      InstanceIds:
        - '{{InstanceId}}'
```

JSON

```
{
         "name":"branchOnOriginalInstanceState",
         "action":"aws:branch",
         "onFailure":"Abort",
         "inputs":{
            "Choices":[
               {
                  "NextStep":"stopInstance",
                  "Not":{
                     "Variable":"{{getInstanceState.instanceState}}",
                     "StringEquals":"running"
                  }
               }
            ]
         },
```

```
         "isEnd":true
      },
      {
         "name":"stopInstance",
         "action":"aws:executeAwsApi",
         "onFailure":"Abort",
         "inputs":{
            "Service":"ec2",
            "Api":"StopInstances",
            "InstanceIds":[
               "{{InstanceId}}"
            ]
         }
      }
   ]
}
```

5. Emily 检查完成的子运行手册内容，并在同一 AWS 账户 和 AWS 区域 中创建运行手册作为目标实例。现在，她已经准备好继续创建父运行手册的内容。下面是完成的子运行手册内容。

   YAML

```
schemaVersion: '0.3'
description: 'An example of an Automation runbook that patches groups of Amazon
 EC2 instances in stages.'
assumeRole: '{{AutomationAssumeRole}}'
parameters:
  AutomationAssumeRole:
    type: String
    description: >-
      '(Optional) The Amazon Resource Name (ARN) of the IAM role that allows
 Automation to perform the
      actions on your behalf. If no role is specified, Systems Manager
      Automation uses your IAM permissions to operate this runbook.'
    default: ''
  InstanceId:
    type: String
    description: >-
      '(Required) The instance you want to patch.'
  SnapshotId:
    type: String
    description: '(Optional) The snapshot ID to use to retrieve a patch baseline
 snapshot.'
    default: ''
```

```
  RebootOption:
    type: String
    description: '(Optional) Reboot behavior after a patch Install operation. If
 you choose NoReboot and patches are installed, the instance is marked as non-
compliant until a subsequent reboot and scan.'
    allowedValues:
      - NoReboot
      - RebootIfNeeded
    default: RebootIfNeeded
  Operation:
    type: String
    description: '(Optional) The update or configuration to perform on the
 instance. The system checks if patches specified in the patch baseline are
 installed on the instance. The install operation installs patches missing from
 the baseline.'
    allowedValues:
      - Install
      - Scan
    default: Install
mainSteps:
  - name: getInstanceState
    action: 'aws:executeAwsApi'
    onFailure: Abort
    inputs:
      inputs:
      Service: ec2
      Api: DescribeInstances
      InstanceIds:
        - '{{InstanceId}}'
    outputs:
      - Name: instanceState
        Selector: '$.Reservations[0].Instances[0].State.Name'
        Type: String
    nextStep: branchOnInstanceState
  - name: branchOnInstanceState
    action: 'aws:branch'
    onFailure: Abort
    inputs:
      Choices:
        - NextStep: startInstance
          Variable: '{{getInstanceState.instanceState}}'
          StringEquals: stopped
        - Or:
            - Variable: '{{getInstanceState.instanceState}}'
```

```
            StringEquals: stopping
          NextStep: verifyInstanceStopped
        - NextStep: patchInstance
          Variable: '{{getInstanceState.instanceState}}'
          StringEquals: running
    isEnd: true
  - name: startInstance
    action: 'aws:executeAwsApi'
    onFailure: Abort
    inputs:
      Service: ec2
      Api: StartInstances
      InstanceIds:
        - '{{InstanceId}}'
    nextStep: verifyInstanceRunning
  - name: verifyInstanceRunning
    action: 'aws:waitForAwsResourceProperty'
    timeoutSeconds: 120
    inputs:
      Service: ec2
      Api: DescribeInstances
      InstanceIds:
        - '{{InstanceId}}'
      PropertySelector: '$.Reservations[0].Instances[0].State.Name'
      DesiredValues:
        - running
    nextStep: patchInstance
  - name: verifyInstanceStopped
    action: 'aws:waitForAwsResourceProperty'
    timeoutSeconds: 120
    inputs:
      Service: ec2
      Api: DescribeInstances
      InstanceIds:
        - '{{InstanceId}}'
      PropertySelector: '$.Reservations[0].Instances[0].State.Name'
      DesiredValues:
        - stopped
      nextStep: startInstance
  - name: patchInstance
    action: 'aws:runCommand'
    onFailure: Abort
    timeoutSeconds: 5400
    inputs:
```

```
      DocumentName: 'AWS-RunPatchBaseline'
      InstanceIds:
      - '{{InstanceId}}'
      Parameters:
        SnapshotId: '{{SnapshotId}}'
        RebootOption: '{{RebootOption}}'
        Operation: '{{Operation}}'
  - name: branchOnOriginalInstanceState
    action: 'aws:branch'
    onFailure: Abort
    inputs:
      Choices:
        - NextStep: stopInstance
          Not:
            Variable: '{{getInstanceState.instanceState}}'
            StringEquals: running
    isEnd: true
  - name: stopInstance
    action: 'aws:executeAwsApi'
    onFailure: Abort
    inputs:
      Service: ec2
      Api: StopInstances
      InstanceIds:
        - '{{InstanceId}}'
```

JSON

```
{
   "schemaVersion":"0.3",
   "description":"An example of an Automation runbook that patches groups of
 Amazon EC2 instances in stages.",
   "assumeRole":"{{AutomationAssumeRole}}",
   "parameters":{
      "AutomationAssumeRole":{
         "type":"String",
         "description":"'(Optional) The Amazon Resource Name (ARN) of the IAM
 role that allows Automation to perform the actions on your behalf. If no role is
 specified, Systems Manager Automation uses your IAM permissions to operate this
 runbook.'",
         "default":""
      },
      "InstanceId":{
```

```
        "type":"String",
        "description":"'(Required) The instance you want to patch.'"
     },
     "SnapshotId":{
        "type":"String",
        "description":"(Optional) The snapshot ID to use to retrieve a patch
baseline snapshot.",
        "default":""
     },
     "RebootOption":{
        "type":"String",
        "description":"(Optional) Reboot behavior after a patch Install
operation. If you choose NoReboot and patches are installed, the instance is
marked as non-compliant until a subsequent reboot and scan.",
        "allowedValues":[
           "NoReboot",
           "RebootIfNeeded"
        ],
        "default":"RebootIfNeeded"
     },
     "Operation":{
        "type":"String",
        "description":"(Optional) The update or configuration to perform on
the instance. The system checks if patches specified in the patch baseline are
installed on the instance. The install operation installs patches missing from
the baseline.",
        "allowedValues":[
           "Install",
           "Scan"
        ],
        "default":"Install"
     }
  },
  "mainSteps":[
     {
        "name":"getInstanceState",
        "action":"aws:executeAwsApi",
        "onFailure":"Abort",
        "inputs":{
           "inputs":null,
           "Service":"ec2",
           "Api":"DescribeInstances",
           "InstanceIds":[
              "{{InstanceId}}"
```

```
            ]
         },
         "outputs":[
            {
               "Name":"instanceState",
               "Selector":"$.Reservations[0].Instances[0].State.Name",
               "Type":"String"
            }
         ],
         "nextStep":"branchOnInstanceState"
      },
      {
         "name":"branchOnInstanceState",
         "action":"aws:branch",
         "onFailure":"Abort",
         "inputs":{
            "Choices":[
               {
                  "NextStep":"startInstance",
                  "Variable":"{{getInstanceState.instanceState}}",
                  "StringEquals":"stopped"
               },
               {
                  "Or":[
                     {
                        "Variable":"{{getInstanceState.instanceState}}",
                        "StringEquals":"stopping"
                     }
                  ],
                  "NextStep":"verifyInstanceStopped"
               },
               {
                  "NextStep":"patchInstance",
                  "Variable":"{{getInstanceState.instanceState}}",
                  "StringEquals":"running"
               }
            ]
         },
         "isEnd":true
      },
      {
         "name":"startInstance",
         "action":"aws:executeAwsApi",
         "onFailure":"Abort",
```

```
            "inputs":{
               "Service":"ec2",
               "Api":"StartInstances",
               "InstanceIds":[
                  "{{InstanceId}}"
               ]
            },
            "nextStep":"verifyInstanceRunning"
         },
         {
            "name":"verifyInstanceRunning",
            "action":"aws:waitForAwsResourceProperty",
            "timeoutSeconds":120,
            "inputs":{
               "Service":"ec2",
               "Api":"DescribeInstances",
               "InstanceIds":[
                  "{{InstanceId}}"
               ],
               "PropertySelector":"$.Reservations[0].Instances[0].State.Name",
               "DesiredValues":[
                  "running"
               ]
            },
            "nextStep":"patchInstance"
         },
         {
            "name":"verifyInstanceStopped",
            "action":"aws:waitForAwsResourceProperty",
            "timeoutSeconds":120,
            "inputs":{
               "Service":"ec2",
               "Api":"DescribeInstances",
               "InstanceIds":[
                  "{{InstanceId}}"
               ],
               "PropertySelector":"$.Reservations[0].Instances[0].State.Name",
               "DesiredValues":[
                  "stopped"
               ],
               "nextStep":"startInstance"
            }
         },
         {
```

```
            "name":"patchInstance",
            "action":"aws:runCommand",
            "onFailure":"Abort",
            "timeoutSeconds":5400,
            "inputs":{
               "DocumentName":"AWS-RunPatchBaseline",
               "InstanceIds":[
                  "{{InstanceId}}"
               ],
               "Parameters":{
                  "SnapshotId":"{{SnapshotId}}",
                  "RebootOption":"{{RebootOption}}",
                  "Operation":"{{Operation}}"
               }
            }
         },
         {
            "name":"branchOnOriginalInstanceState",
            "action":"aws:branch",
            "onFailure":"Abort",
            "inputs":{
               "Choices":[
                  {
                     "NextStep":"stopInstance",
                     "Not":{
                        "Variable":"{{getInstanceState.instanceState}}",
                        "StringEquals":"running"
                     }
                  }
               ]
            },
            "isEnd":true
         },
         {
            "name":"stopInstance",
            "action":"aws:executeAwsApi",
            "onFailure":"Abort",
            "inputs":{
               "Service":"ec2",
               "Api":"StopInstances",
               "InstanceIds":[
                  "{{InstanceId}}"
               ]
            }
```

```
      }
   ]
}
```

有关此示例中使用的自动化操作的更多信息，请参阅 Systems Manager 自动化操作参考。

创建父运行手册

此示例运行手册继续了上一节介绍的场景。现在 Emily 已经创建了子运行手册，她开始为父运行手册创作内容，如下所示：

1. 首先，她为运行手册的架构和描述提供值，并定义父运行手册的输入参数。

   通过使用 AutomationAssumeRole 参数，Emily 和她的同事可以使用现有的 IAM 角色，使自动化代表他们执行运行手册中的操作。Emily 使用 PatchGroupPrimaryKey 和 PatchGroupPrimaryValue 参数来指定与将要修补的数据库实例的主组关联的标签。她使用 PatchGroupSecondaryKey 和 PatchGroupSecondaryValue 参数来指定与将要修补的数据库实例的辅助组关联的标签。

   YAML

```
description: 'An example of an Automation runbook that patches groups of Amazon
 EC2 instances in stages.'
schemaVersion: '0.3'
assumeRole: '{{AutomationAssumeRole}}'
parameters:
  AutomationAssumeRole:
    type: String
    description: '(Optional) The Amazon Resource Name (ARN) of the IAM role that
 allows Automation to perform the actions on your behalf. If no role is specified,
 Systems Manager Automation uses your IAM permissions to operate this runbook.'
    default: ''
  PatchGroupPrimaryKey:
    type: String
    description: '(Required) The key of the tag for the primary group of instances
 you want to patch.''
  PatchGroupPrimaryValue:
    type: String
    description: '(Required) The value of the tag for the primary group of
 instances you want to patch.'
  PatchGroupSecondaryKey:
    type: String
```

```
    description: '(Required) The key of the tag for the secondary group of
instances you want to patch.'
 PatchGroupSecondaryValue:
    type: String
    description: '(Required) The value of the tag for the secondary group of
instances you want to patch.'
```

JSON

```
{
   "schemaVersion": "0.3",
   "description": "An example of an Automation runbook that patches groups of
Amazon EC2 instances in stages.",
   "assumeRole": "{{AutomationAssumeRole}}",
   "parameters": {
      "AutomationAssumeRole": {
         "type": "String",
         "description": "(Optional) The Amazon Resource Name (ARN) of the IAM
role that allows Automation to perform the actions on your behalf. If no role is
specified, Systems Manager Automation uses your IAM permissions to operate this
runbook.",
         "default": ""
      },
      "PatchGroupPrimaryKey": {
         "type": "String",
         "description": "(Required) The key of the tag for the primary group of
instances you want to patch."
      },
      "PatchGroupPrimaryValue": {
         "type": "String",
         "description": "(Required) The value of the tag for the primary group of
instances you want to patch."
      },
      "PatchGroupSecondaryKey": {
         "type": "String",
         "description": "(Required) The key of the tag for the secondary group of
instances you want to patch."
      },
      "PatchGroupSecondaryValue": {
         "type": "String",
         "description": "(Required) The value of the tag for the secondary group
of instances you want to patch."
      }
```

```
    }
},
```

2. 定义顶层元素后，Emily 继续创作构成运行手册的 mainSteps 的操作。

   第一个操作使用她刚刚创建的子运行手册启动速率控制自动化，该子运行手册将与在 PatchGroupPrimaryKey 和 PatchGroupPrimaryValue 输入参数中所指定标签关联的实例设置为目标。她使用提供给输入参数的值来指定与要修补的主数据库实例组相关联的标签的键和值。

   第一个自动化完成后，第二个操作将使用子运行手册启动另一个速率控制自动化，该子运行手册将与在 PatchGroupSecondaryKey 和 PatchGroupSecondaryValue 输入参数中所指定标签关联的实例设置为目标。她使用提供给输入参数的值来指定与要修补的辅助数据库实例组相关联的标签的键和值。

   YAML

```
mainSteps:
  - name: patchPrimaryTargets
    action: 'aws:executeAutomation'
    onFailure: Abort
    timeoutSeconds: 7200
    inputs:
      DocumentName: RunbookTutorialChildAutomation
      Targets:
        - Key: 'tag:{{PatchGroupPrimaryKey}}'
          Values:
            - '{{PatchGroupPrimaryValue}}'
      TargetParameterName: 'InstanceId'
  - name: patchSecondaryTargets
    action: 'aws:executeAutomation'
    onFailure: Abort
    timeoutSeconds: 7200
    inputs:
      DocumentName: RunbookTutorialChildAutomation
      Targets:
        - Key: 'tag:{{PatchGroupSecondaryKey}}'
          Values:
            - '{{PatchGroupSecondaryValue}}'
      TargetParameterName: 'InstanceId'
```

   JSON

```
"mainSteps":[
```

```
      {
         "name":"patchPrimaryTargets",
         "action":"aws:executeAutomation",
         "onFailure":"Abort",
         "timeoutSeconds":7200,
         "inputs":{
            "DocumentName":"RunbookTutorialChildAutomation",
            "Targets":[
               {
                  "Key":"tag:{{PatchGroupPrimaryKey}}",
                  "Values":[
                     "{{PatchGroupPrimaryValue}}"
                  ]
               }
            ],
            "TargetParameterName":"InstanceId"
         }
      },
      {
         "name":"patchSecondaryTargets",
         "action":"aws:executeAutomation",
         "onFailure":"Abort",
         "timeoutSeconds":7200,
         "inputs":{
            "DocumentName":"RunbookTutorialChildAutomation",
            "Targets":[
               {
                  "Key":"tag:{{PatchGroupSecondaryKey}}",
                  "Values":[
                     "{{PatchGroupSecondaryValue}}"
                  ]
               }
            ],
            "TargetParameterName":"InstanceId"
         }
      }
   ]
}
```

3. Emily 检查完成的父运行手册内容，并在相同的 AWS 账户和 AWS 区域中创建运行手册作为目标实例。现在，她已经准备好测试自己的运行手册，以确保在将自动化实施到生产环境之前，自动化能够按需运行。下面是完成的父运行手册内容。

YAML

```
description: An example of an Automation runbook that patches groups of Amazon EC2
 instances in stages.
schemaVersion: '0.3'
assumeRole: '{{AutomationAssumeRole}}'
parameters:
  AutomationAssumeRole:
    type: String
    description: '(Optional) The Amazon Resource Name (ARN) of the IAM role that
 allows Automation to perform the actions on your behalf. If no role is specified,
 Systems Manager Automation uses your IAM permissions to operate this runbook.'
    default: ''
  PatchGroupPrimaryKey:
    type: String
    description: (Required) The key of the tag for the primary group of instances
 you want to patch.
  PatchGroupPrimaryValue:
    type: String
    description: '(Required) The value of the tag for the primary group of
 instances you want to patch. '
  PatchGroupSecondaryKey:
    type: String
    description: (Required) The key of the tag for the secondary group of
 instances you want to patch.
  PatchGroupSecondaryValue:
    type: String
    description: '(Required) The value of the tag for the secondary group of
 instances you want to patch.  '
mainSteps:
  - name: patchPrimaryTargets
    action: 'aws:executeAutomation'
    onFailure: Abort
    timeoutSeconds: 7200
    inputs:
      DocumentName: RunbookTutorialChildAutomation
      Targets:
        - Key: 'tag:{{PatchGroupPrimaryKey}}'
          Values:
            - '{{PatchGroupPrimaryValue}}'
      TargetParameterName: 'InstanceId'
  - name: patchSecondaryTargets
    action: 'aws:executeAutomation'
```

```
    onFailure: Abort
    timeoutSeconds: 7200
    inputs:
      DocumentName: RunbookTutorialChildAutomation
      Targets:
        - Key: 'tag:{{PatchGroupSecondaryKey}}'
          Values:
            - '{{PatchGroupSecondaryValue}}'
      TargetParameterName: 'InstanceId'
```

JSON

```
{
   "description":"An example of an Automation runbook that patches groups of
 Amazon EC2 instances in stages.",
   "schemaVersion":"0.3",
   "assumeRole":"{{AutomationAssumeRole}}",
   "parameters":{
      "AutomationAssumeRole":{
         "type":"String",
         "description":"(Optional) The Amazon Resource Name (ARN) of the IAM role
 that allows Automation to perform the actions on your behalf. If no role is
 specified, Systems Manager Automation uses your IAM permissions to operate this
 runbook.",
         "default":""
      },
      "PatchGroupPrimaryKey":{
         "type":"String",
         "description":"(Required) The key of the tag for the primary group of
 instances you want to patch."
      },
      "PatchGroupPrimaryValue":{
         "type":"String",
         "description":"(Required) The value of the tag for the primary group of
 instances you want to patch. "
      },
      "PatchGroupSecondaryKey":{
         "type":"String",
         "description":"(Required) The key of the tag for the secondary group of
 instances you want to patch."
      },
      "PatchGroupSecondaryValue":{
         "type":"String",
```

```
         "description":"(Required) The value of the tag for the secondary group of
 instances you want to patch.  "
      }
   },
   "mainSteps":[
      {
         "name":"patchPrimaryTargets",
         "action":"aws:executeAutomation",
         "onFailure":"Abort",
         "timeoutSeconds":7200,
         "inputs":{
            "DocumentName":"RunbookTutorialChildAutomation",
            "Targets":[
               {
                  "Key":"tag:{{PatchGroupPrimaryKey}}",
                  "Values":[
                     "{{PatchGroupPrimaryValue}}"
                  ]
               }
            ],
            "TargetParameterName":"InstanceId"
         }
      },
      {
         "name":"patchSecondaryTargets",
         "action":"aws:executeAutomation",
         "onFailure":"Abort",
         "timeoutSeconds":7200,
         "inputs":{
            "DocumentName":"RunbookTutorialChildAutomation",
            "Targets":[
               {
                  "Key":"tag:{{PatchGroupSecondaryKey}}",
                  "Values":[
                     "{{PatchGroupSecondaryValue}}"
                  ]
               }
            ],
            "TargetParameterName":"InstanceId"
         }
      }
   ]
}
```

有关此示例中使用的自动化操作的更多信息，请参阅 Systems Manager 自动化操作参考。

示例 2：脚本化运行手册

此示例运行手册解决以下情形。Emily 是 AnyCompany Consultants, LLC 的系统工程师。她之前创建了两个运行手册，这两个运行手册用于父子关系，以修补托管主数据库和辅助数据库的 Amazon Elastic Compute Cloud (Amazon EC2) 实例的组。应用程序每天 24 小时访问这些数据库，因此这两个数据库实例中必须有一个始终可用。

基于此要求，她构建了一个解决方案，使用 AWS-RunPatchBaseline Systems Manager (SSM) 文档分阶段修补实例。通过使用此 SSM 文档，她的同事可以在修补操作完成后查看相关的修补程序合规性信息。

首先修补数据库实例的主组，然后修补数据库实例的辅助组。此外，为了避免由于运行之前已停止的实例而产生额外的成本，Emily 确保自动化将修补的实例恢复到修补之前的原始状态。Emily 使用与数据库实例的主组和辅助组关联的标签来确定应按照所需顺序修补哪些实例。

她现有的自动化解决方案有效，但她希望尽可能改进自己的解决方案。为了帮助维护运行手册内容并简化故障排除工作，她希望将自动化压缩到一个运行手册中，并简化输入参数的数量。此外，她希望避免创建多个子自动化。

在审查了可用的自动化操作后，Emily 确定可以使用 aws:executeScript 操作来运行她的自定义 Python 脚本，从而改进她的解决方案。她现在开始创作运行手册的内容，如下所示：

1. 首先，她为运行手册的架构和描述提供值，并定义父运行手册的输入参数。

   通过使用 AutomationAssumeRole 参数，Emily 和她的同事可以使用现有的 IAM 角色，使自动化代表他们执行运行手册中的操作。与示例 1 不同的是，AutomationAssumeRole 参数现在是必需的，而不是可选的。因为这个运行手册包括 aws:executeScript 操作，所以始终需要一个 AWS Identity and Access Management (IAM) 服务角色（或担任角色）。此要求是必要的，因为某些为操作指定的 Python 脚本调用 AWS API 操作。

   Emily 使用 PrimaryPatchGroupTag 和 SecondaryPatchGroupTag 参数指定与要修补的数据库实例的主要和辅助组关联的标签。为了简化所需的输入参数，她决定使用 StringMap 参数，而不是使用多个 String参数，因为她在示例 1 运行手册中使用了后者。（可选）Operation、RebootOption 和 SnapshotId 参数可用于为 AWS-RunPatchBaseline 的文档参数提供值。为了防止向这些运行手册参数提供无效值，她根据需要定义了 allowedValues。

YAML

```
description: 'An example of an Automation runbook that patches groups of Amazon
 EC2 instances in stages.'
schemaVersion: '0.3'
assumeRole: '{{AutomationAssumeRole}}'
parameters:
  AutomationAssumeRole:
    type: String
    description: '(Required) The Amazon Resource Name (ARN) of the IAM role that
 allows Automation to perform the actions on your behalf. If no role is specified,
 Systems Manager Automation uses your IAM permissions to operate this runbook.'
  PrimaryPatchGroupTag:
    type: StringMap
    description: '(Required) The tag for the primary group of instances you want
 to patch. Specify a key-value pair. Example: {"key" : "value"}'
  SecondaryPatchGroupTag:
    type: StringMap
    description: '(Required) The tag for the secondary group of instances you want
 to patch. Specify a key-value pair. Example: {"key" : "value"}'
  SnapshotId:
    type: String
    description: '(Optional) The snapshot ID to use to retrieve a patch baseline
 snapshot.'
    default: ''
  RebootOption:
    type: String
    description: '(Optional) Reboot behavior after a patch Install operation. If
 you choose NoReboot and patches are installed, the instance is marked as non-
compliant until a subsequent reboot and scan.'
    allowedValues:
      - NoReboot
      - RebootIfNeeded
    default: RebootIfNeeded
  Operation:
    type: String
    description: '(Optional) The update or configuration to perform on the
 instance. The system checks if patches specified in the patch baseline are
 installed on the instance. The install operation installs patches missing from
 the baseline.'
    allowedValues:
      - Install
      - Scan
```

```
    default: Install
```

JSON

```
{
   "description":"An example of an Automation runbook that patches groups of
 Amazon EC2 instances in stages.",
   "schemaVersion":"0.3",
   "assumeRole":"{{AutomationAssumeRole}}",
   "parameters":{
      "AutomationAssumeRole":{
         "type":"String",
         "description":"(Required) The Amazon Resource Name (ARN) of the IAM role
 that allows Automation to perform the actions on your behalf. If no role is
 specified, Systems Manager Automation uses your IAM permissions to operate this
 runbook."
      },
      "PrimaryPatchGroupTag":{
         "type":"StringMap",
         "description":"(Required) The tag for the primary group of instances you
 want to patch. Specify a key-value pair. Example: {\"key\" : \"value\"}"
      },
      "SecondaryPatchGroupTag":{
         "type":"StringMap",
         "description":"(Required) The tag for the secondary group of instances
 you want to patch. Specify a key-value pair. Example: {\"key\" : \"value\"}"
      },
      "SnapshotId":{
         "type":"String",
         "description":"(Optional) The snapshot ID to use to retrieve a patch
 baseline snapshot.",
         "default":""
      },
      "RebootOption":{
         "type":"String",
         "description":"(Optional) Reboot behavior after a patch Install
 operation. If you choose NoReboot and patches are installed, the instance is
 marked as non-compliant until a subsequent reboot and scan.",
         "allowedValues":[
            "NoReboot",
            "RebootIfNeeded"
         ],
         "default":"RebootIfNeeded"
```

```
      },
      "Operation":{
         "type":"String",
         "description":"(Optional) The update or configuration to perform on
 the instance. The system checks if patches specified in the patch baseline are
 installed on the instance. The install operation installs patches missing from
 the baseline.",
         "allowedValues":[
            "Install",
            "Scan"
         ],
         "default":"Install"
      }
   }
},
```

2. 定义顶层元素后，Emily 继续创作构成运行手册的 mainSteps 的操作。第一步收集与 PrimaryPatchGroupTag 参数中指定的标签关联的所有实例的 ID，并输出 StringMap 参数，其中包含实例 ID 和实例的当前状态。此操作的输出将用于后续操作。

   请注意，script 输入参数不支持 JSON 运行手册。JSON 运行手册必须使用 attachment 输入参数提供脚本内容。

   YAML

```
mainSteps:
  - name: getPrimaryInstanceState
    action: 'aws:executeScript'
    timeoutSeconds: 120
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: getInstanceStates
      InputPayload:
        primaryTag: '{{PrimaryPatchGroupTag}}'
      Script: |-
        def getInstanceStates(events,context):
          import boto3

          #Initialize client
          ec2 = boto3.client('ec2')
          tag = events['primaryTag']
          tagKey, tagValue = list(tag.items())[0]
          instanceQuery = ec2.describe_instances(
```

```
          Filters=[
              {
                  "Name": "tag:" + tagKey,
                  "Values": [tagValue]
              }]
          )
          if not instanceQuery['Reservations']:
              noInstancesForTagString = "No instances found for specified tag."
              return({ 'noInstancesFound' : noInstancesForTagString })
          else:
              queryResponse = instanceQuery['Reservations']
              originalInstanceStates = {}
              for results in queryResponse:
                  instanceSet = results['Instances']
                  for instance in instanceSet:
                      instanceId = instance['InstanceId']
                      originalInstanceStates[instanceId] = instance['State']
['Name']
              return originalInstanceStates
    outputs:
      - Name: originalInstanceStates
        Selector: $.Payload
        Type: StringMap
    nextStep: verifyPrimaryInstancesRunning
```

JSON

```
"mainSteps":[
      {
         "name":"getPrimaryInstanceState",
         "action":"aws:executeScript",
         "timeoutSeconds":120,
         "onFailure":"Abort",
         "inputs":{
            "Runtime":"python3.7",
            "Handler":"getInstanceStates",
            "InputPayload":{
               "primaryTag":"{{PrimaryPatchGroupTag}}"
            },
            "Script":"..."
         },
         "outputs":[
            {
```

```
                "Name":"originalInstanceStates",
                "Selector":"$.Payload",
                "Type":"StringMap"
            }
         ],
         "nextStep":"verifyPrimaryInstancesRunning"
      },
```

3. Emily 将上一个操作的输出用于另一个 aws:executeScript 操作，以验证与 PrimaryPatchGroupTag 参数中所指定标签相关联的所有实例均处于 running 状态。

   如果实例状态已处于 running 或者 shutting-down 状态，则脚本将继续遍历剩余的实例。

   如果实例的状态为 stopping，则脚本轮询实例以达到 stopped 状态并启动实例。

   如果实例的状态为 stopped，脚本将启动实例。

   YAML

```
- name: verifyPrimaryInstancesRunning
    action: 'aws:executeScript'
    timeoutSeconds: 600
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: verifyInstancesRunning
      InputPayload:
        targetInstances: '{{getPrimaryInstanceState.originalInstanceStates}}'
      Script: |-
        def verifyInstancesRunning(events,context):
          import boto3

          #Initialize client
          ec2 = boto3.client('ec2')
          instanceDict = events['targetInstances']
          for instance in instanceDict:
            if instanceDict[instance] == 'stopped':
                print("The target instance " + instance + " is stopped. The
 instance will now be started.")
                ec2.start_instances(
                    InstanceIds=[instance]
                    )
            elif instanceDict[instance] == 'stopping':
```

```
                print("The target instance " + instance + " is stopping. Polling
 for instance to reach stopped state.")
                while instanceDict[instance] != 'stopped':
                    poll = ec2.get_waiter('instance_stopped')
                    poll.wait(
                        InstanceIds=[instance]
                    )
                ec2.start_instances(
                    InstanceIds=[instance]
                )
            else:
              pass
    nextStep: waitForPrimaryRunningInstances
```

JSON

```
{
         "name":"verifyPrimaryInstancesRunning",
         "action":"aws:executeScript",
         "timeoutSeconds":600,
         "onFailure":"Abort",
         "inputs":{
            "Runtime":"python3.7",
            "Handler":"verifyInstancesRunning",
            "InputPayload":{

 "targetInstances":"{{getPrimaryInstanceState.originalInstanceStates}}"
            },
            "Script":"..."
         },
         "nextStep":"waitForPrimaryRunningInstances"
      },
```

4. Emily 验证与 PrimaryPatchGroupTag 参数中指定的标签相关联的所有实例已启动或已处于 running 状态。然后，她使用另一个脚本来验证所有实例（包括在上一操作中启动的实例）已处于 running 状态。

   YAML

```
- name: waitForPrimaryRunningInstances
    action: 'aws:executeScript'
    timeoutSeconds: 300
    onFailure: Abort
```

```
    inputs:
      Runtime: python3.7
      Handler: waitForRunningInstances
      InputPayload:
        targetInstances: '{{getPrimaryInstanceState.originalInstanceStates}}'
      Script: |-
        def waitForRunningInstances(events,context):
          import boto3

          #Initialize client
          ec2 = boto3.client('ec2')
          instanceDict = events['targetInstances']
          for instance in instanceDict:
              poll = ec2.get_waiter('instance_running')
              poll.wait(
                  InstanceIds=[instance]
              )
    nextStep: returnPrimaryTagKey
```

JSON

```
{
         "name":"waitForPrimaryRunningInstances",
         "action":"aws:executeScript",
         "timeoutSeconds":300,
         "onFailure":"Abort",
         "inputs":{
            "Runtime":"python3.7",
            "Handler":"waitForRunningInstances",
            "InputPayload":{

 "targetInstances":"{{getPrimaryInstanceState.originalInstanceStates}}"
            },
            "Script":"..."
         },
         "nextStep":"returnPrimaryTagKey"
      },
```

5. Emily 使用两个脚本来返回键的单个 String 值和 PrimaryPatchGroupTag 参数中指定的标签的值。这些操作返回的值允许她直接向 Targets 文档的参数 AWS-RunPatchBaseline 提供值。然后，自动化将继续使用 aws:runCommand 操作，利用 AWS-RunPatchBaseline 文档修补实例。

YAML

```
- name: returnPrimaryTagKey
    action: 'aws:executeScript'
    timeoutSeconds: 120
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: returnTagValues
      InputPayload:
        primaryTag: '{{PrimaryPatchGroupTag}}'
      Script: |-
        def returnTagValues(events,context):
          tag = events['primaryTag']
          tagKey = list(tag)[0]
          stringKey = "tag:" + tagKey
          return {'tagKey' : stringKey}
    outputs:
      - Name: Payload
        Selector: $.Payload
        Type: StringMap
      - Name: primaryPatchGroupKey
        Selector: $.Payload.tagKey
        Type: String
    nextStep: returnPrimaryTagValue
  - name: returnPrimaryTagValue
    action: 'aws:executeScript'
    timeoutSeconds: 120
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: returnTagValues
      InputPayload:
        primaryTag: '{{PrimaryPatchGroupTag}}'
      Script: |-
        def returnTagValues(events,context):
          tag = events['primaryTag']
          tagKey = list(tag)[0]
          tagValue = tag[tagKey]
          return {'tagValue' : tagValue}
    outputs:
      - Name: Payload
        Selector: $.Payload
```

```
        Type: StringMap
      - Name: primaryPatchGroupValue
        Selector: $.Payload.tagValue
        Type: String
    nextStep: patchPrimaryInstances
  - name: patchPrimaryInstances
    action: 'aws:runCommand'
    onFailure: Abort
    timeoutSeconds: 7200
    inputs:
      DocumentName: AWS-RunPatchBaseline
      Parameters:
        SnapshotId: '{{SnapshotId}}'
        RebootOption: '{{RebootOption}}'
        Operation: '{{Operation}}'
      Targets:
        - Key: '{{returnPrimaryTagKey.primaryPatchGroupKey}}'
          Values:
            - '{{returnPrimaryTagValue.primaryPatchGroupValue}}'
      MaxConcurrency: 10%
      MaxErrors: 10%
    nextStep: returnPrimaryToOriginalState
```

JSON

```
{
         "name":"returnPrimaryTagKey",
         "action":"aws:executeScript",
         "timeoutSeconds":120,
         "onFailure":"Abort",
         "inputs":{
            "Runtime":"python3.7",
            "Handler":"returnTagValues",
            "InputPayload":{
               "primaryTag":"{{PrimaryPatchGroupTag}}"
            },
            "Script":"..."
         },
         "outputs":[
            {
               "Name":"Payload",
               "Selector":"$.Payload",
               "Type":"StringMap"
```

```
            },
            {
               "Name":"primaryPatchGroupKey",
               "Selector":"$.Payload.tagKey",
               "Type":"String"
            }
         ],
         "nextStep":"returnPrimaryTagValue"
      },
      {
         "name":"returnPrimaryTagValue",
         "action":"aws:executeScript",
         "timeoutSeconds":120,
         "onFailure":"Abort",
         "inputs":{
            "Runtime":"python3.7",
            "Handler":"returnTagValues",
            "InputPayload":{
               "primaryTag":"{{PrimaryPatchGroupTag}}"
            },
            "Script":"..."
         },
         "outputs":[
            {
               "Name":"Payload",
               "Selector":"$.Payload",
               "Type":"StringMap"
            },
            {
               "Name":"primaryPatchGroupValue",
               "Selector":"$.Payload.tagValue",
               "Type":"String"
            }
         ],
         "nextStep":"patchPrimaryInstances"
      },
      {
         "name":"patchPrimaryInstances",
         "action":"aws:runCommand",
         "onFailure":"Abort",
         "timeoutSeconds":7200,
         "inputs":{
            "DocumentName":"AWS-RunPatchBaseline",
            "Parameters":{
```

```
                "SnapshotId":"{{SnapshotId}}",
                "RebootOption":"{{RebootOption}}",
                "Operation":"{{Operation}}"
             },
             "Targets":[
                {
                   "Key":"{{returnPrimaryTagKey.primaryPatchGroupKey}}",
                   "Values":[
                      "{{returnPrimaryTagValue.primaryPatchGroupValue}}"
                   ]
                }
             ],
             "MaxConcurrency":"10%",
             "MaxErrors":"10%"
          },
          "nextStep":"returnPrimaryToOriginalState"
       },
```

6. 修补操作完成后，Emily 希望自动化将与 PrimaryPatchGroupTag 参数中所指定标签相关联的目标实例返回到自动化开始之前的状态。她通过再次使用脚本中第一个操作的输出来完成此操作。根据目标实例的原始状态，如果该实例以前处于 running 以外的任何状态，则实例将停止。否则，如果实例状态为 running，则脚本将继续遍历剩余的实例。

   YAML

```
- name: returnPrimaryToOriginalState
    action: 'aws:executeScript'
    timeoutSeconds: 600
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: returnToOriginalState
      InputPayload:
        targetInstances: '{{getPrimaryInstanceState.originalInstanceStates}}'
      Script: |-
        def returnToOriginalState(events,context):
          import boto3

          #Initialize client
          ec2 = boto3.client('ec2')
          instanceDict = events['targetInstances']
          for instance in instanceDict:
            if instanceDict[instance] == 'stopped' or instanceDict[instance] ==
 'stopping':
```

```
                ec2.stop_instances(
                    InstanceIds=[instance]
                    )
            else:
              pass
    nextStep: getSecondaryInstanceState
```

JSON

```
{
         "name":"returnPrimaryToOriginalState",
         "action":"aws:executeScript",
         "timeoutSeconds":600,
         "onFailure":"Abort",
         "inputs":{
            "Runtime":"python3.7",
            "Handler":"returnToOriginalState",
            "InputPayload":{

 "targetInstances":"{{getPrimaryInstanceState.originalInstanceStates}}"
            },
            "Script":"..."
         },
         "nextStep":"getSecondaryInstanceState"
      },
```

7. 针对与 PrimaryPatchGroupTag 参数中所指定标签相关联的实例的修补操作已完成。现在 Emily 复制了运行手册内容中以前的所有操作，将与 SecondaryPatchGroupTag 参数中所指定标签相关联的实例设置为目标。

   YAML

```
- name: getSecondaryInstanceState
    action: 'aws:executeScript'
    timeoutSeconds: 120
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: getInstanceStates
      InputPayload:
        secondaryTag: '{{SecondaryPatchGroupTag}}'
      Script: |-
        def getInstanceStates(events,context):
```

```
          import boto3

          #Initialize client
          ec2 = boto3.client('ec2')
          tag = events['secondaryTag']
          tagKey, tagValue = list(tag.items())[0]
          instanceQuery = ec2.describe_instances(
          Filters=[
              {
                  "Name": "tag:" + tagKey,
                  "Values": [tagValue]
              }]
          )
          if not instanceQuery['Reservations']:
              noInstancesForTagString = "No instances found for specified tag."
              return({ 'noInstancesFound' : noInstancesForTagString })
          else:
              queryResponse = instanceQuery['Reservations']
              originalInstanceStates = {}
              for results in queryResponse:
                  instanceSet = results['Instances']
                  for instance in instanceSet:
                      instanceId = instance['InstanceId']
                      originalInstanceStates[instanceId] = instance['State']
['Name']
              return originalInstanceStates
    outputs:
      - Name: originalInstanceStates
        Selector: $.Payload
        Type: StringMap
    nextStep: verifySecondaryInstancesRunning
  - name: verifySecondaryInstancesRunning
    action: 'aws:executeScript'
    timeoutSeconds: 600
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: verifyInstancesRunning
      InputPayload:
        targetInstances: '{{getSecondaryInstanceState.originalInstanceStates}}'
      Script: |-
        def verifyInstancesRunning(events,context):
          import boto3
```

```
          #Initialize client
          ec2 = boto3.client('ec2')
          instanceDict = events['targetInstances']
          for instance in instanceDict:
            if instanceDict[instance] == 'stopped':
                print("The target instance " + instance + " is stopped. The
instance will now be started.")
                ec2.start_instances(
                    InstanceIds=[instance]
                    )
            elif instanceDict[instance] == 'stopping':
                print("The target instance " + instance + " is stopping. Polling
for instance to reach stopped state.")
                while instanceDict[instance] != 'stopped':
                    poll = ec2.get_waiter('instance_stopped')
                    poll.wait(
                        InstanceIds=[instance]
                    )
                ec2.start_instances(
                    InstanceIds=[instance]
                )
            else:
              pass
    nextStep: waitForSecondaryRunningInstances
  - name: waitForSecondaryRunningInstances
    action: 'aws:executeScript'
    timeoutSeconds: 300
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: waitForRunningInstances
      InputPayload:
        targetInstances: '{{getSecondaryInstanceState.originalInstanceStates}}'
      Script: |-
        def waitForRunningInstances(events,context):
          import boto3

          #Initialize client
          ec2 = boto3.client('ec2')
          instanceDict = events['targetInstances']
          for instance in instanceDict:
              poll = ec2.get_waiter('instance_running')
              poll.wait(
                  InstanceIds=[instance]
```

```
            )
    nextStep: returnSecondaryTagKey
  - name: returnSecondaryTagKey
    action: 'aws:executeScript'
    timeoutSeconds: 120
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: returnTagValues
      InputPayload:
        secondaryTag: '{{SecondaryPatchGroupTag}}'
      Script: |-
        def returnTagValues(events,context):
          tag = events['secondaryTag']
          tagKey = list(tag)[0]
          stringKey = "tag:" + tagKey
          return {'tagKey' : stringKey}
    outputs:
      - Name: Payload
        Selector: $.Payload
        Type: StringMap
      - Name: secondaryPatchGroupKey
        Selector: $.Payload.tagKey
        Type: String
    nextStep: returnSecondaryTagValue
  - name: returnSecondaryTagValue
    action: 'aws:executeScript'
    timeoutSeconds: 120
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: returnTagValues
      InputPayload:
        secondaryTag: '{{SecondaryPatchGroupTag}}'
      Script: |-
        def returnTagValues(events,context):
          tag = events['secondaryTag']
          tagKey = list(tag)[0]
          tagValue = tag[tagKey]
          return {'tagValue' : tagValue}
    outputs:
      - Name: Payload
        Selector: $.Payload
        Type: StringMap
```

```
      - Name: secondaryPatchGroupValue
        Selector: $.Payload.tagValue
        Type: String
    nextStep: patchSecondaryInstances
  - name: patchSecondaryInstances
    action: 'aws:runCommand'
    onFailure: Abort
    timeoutSeconds: 7200
    inputs:
      DocumentName: AWS-RunPatchBaseline
      Parameters:
        SnapshotId: '{{SnapshotId}}'
        RebootOption: '{{RebootOption}}'
        Operation: '{{Operation}}'
      Targets:
        - Key: '{{returnSecondaryTagKey.secondaryPatchGroupKey}}'
          Values:
          - '{{returnSecondaryTagValue.secondaryPatchGroupValue}}'
      MaxConcurrency: 10%
      MaxErrors: 10%
    nextStep: returnSecondaryToOriginalState
  - name: returnSecondaryToOriginalState
    action: 'aws:executeScript'
    timeoutSeconds: 600
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: returnToOriginalState
      InputPayload:
        targetInstances: '{{getSecondaryInstanceState.originalInstanceStates}}'
      Script: |-
        def returnToOriginalState(events,context):
          import boto3

          #Initialize client
          ec2 = boto3.client('ec2')
          instanceDict = events['targetInstances']
          for instance in instanceDict:
            if instanceDict[instance] == 'stopped' or instanceDict[instance] ==
'stopping':
                ec2.stop_instances(
                    InstanceIds=[instance]
                    )
            else:
```

```
                pass
```

JSON

```
{
         "name":"getSecondaryInstanceState",
         "action":"aws:executeScript",
         "timeoutSeconds":120,
         "onFailure":"Abort",
         "inputs":{
            "Runtime":"python3.7",
            "Handler":"getInstanceStates",
            "InputPayload":{
               "secondaryTag":"{{SecondaryPatchGroupTag}}"
            },
            "Script":"..."
         },
         "outputs":[
            {
               "Name":"originalInstanceStates",
               "Selector":"$.Payload",
               "Type":"StringMap"
            }
         ],
         "nextStep":"verifySecondaryInstancesRunning"
      },
      {
         "name":"verifySecondaryInstancesRunning",
         "action":"aws:executeScript",
         "timeoutSeconds":600,
         "onFailure":"Abort",
         "inputs":{
            "Runtime":"python3.7",
            "Handler":"verifyInstancesRunning",
            "InputPayload":{

 "targetInstances":"{{getSecondaryInstanceState.originalInstanceStates}}"
            },
            "Script":"..."
         },
         "nextStep":"waitForSecondaryRunningInstances"
      },
      {
```

```
         "name":"waitForSecondaryRunningInstances",
         "action":"aws:executeScript",
         "timeoutSeconds":300,
         "onFailure":"Abort",
         "inputs":{
            "Runtime":"python3.7",
            "Handler":"waitForRunningInstances",
            "InputPayload":{

"targetInstances":"{{getSecondaryInstanceState.originalInstanceStates}}"
            },
            "Script":"..."
         },
         "nextStep":"returnSecondaryTagKey"
      },
      {
         "name":"returnSecondaryTagKey",
         "action":"aws:executeScript",
         "timeoutSeconds":120,
         "onFailure":"Abort",
         "inputs":{
            "Runtime":"python3.7",
            "Handler":"returnTagValues",
            "InputPayload":{
               "secondaryTag":"{{SecondaryPatchGroupTag}}"
            },
            "Script":"..."
         },
         "outputs":[
            {
               "Name":"Payload",
               "Selector":"$.Payload",
               "Type":"StringMap"
            },
            {
               "Name":"secondaryPatchGroupKey",
               "Selector":"$.Payload.tagKey",
               "Type":"String"
            }
         ],
         "nextStep":"returnSecondaryTagValue"
      },
      {
         "name":"returnSecondaryTagValue",
```

```
         "action":"aws:executeScript",
         "timeoutSeconds":120,
         "onFailure":"Abort",
         "inputs":{
            "Runtime":"python3.7",
            "Handler":"returnTagValues",
            "InputPayload":{
               "secondaryTag":"{{SecondaryPatchGroupTag}}"
            },
            "Script":"..."
         },
         "outputs":[
            {
               "Name":"Payload",
               "Selector":"$.Payload",
               "Type":"StringMap"
            },
            {
               "Name":"secondaryPatchGroupValue",
               "Selector":"$.Payload.tagValue",
               "Type":"String"
            }
         ],
         "nextStep":"patchSecondaryInstances"
      },
      {
         "name":"patchSecondaryInstances",
         "action":"aws:runCommand",
         "onFailure":"Abort",
         "timeoutSeconds":7200,
         "inputs":{
            "DocumentName":"AWS-RunPatchBaseline",
            "Parameters":{
               "SnapshotId":"{{SnapshotId}}",
               "RebootOption":"{{RebootOption}}",
               "Operation":"{{Operation}}"
            },
            "Targets":[
               {
                  "Key":"{{returnSecondaryTagKey.secondaryPatchGroupKey}}",
                  "Values":[
                     "{{returnSecondaryTagValue.secondaryPatchGroupValue}}"
                  ]
               }
```

```
            ],
            "MaxConcurrency":"10%",
            "MaxErrors":"10%"
         },
         "nextStep":"returnSecondaryToOriginalState"
      },
      {
         "name":"returnSecondaryToOriginalState",
         "action":"aws:executeScript",
         "timeoutSeconds":600,
         "onFailure":"Abort",
         "inputs":{
            "Runtime":"python3.7",
            "Handler":"returnToOriginalState",
            "InputPayload":{

 "targetInstances":"{{getSecondaryInstanceState.originalInstanceStates}}"
            },
            "Script":"..."
         }
      }
   ]
}
```

8. Emily 查看已完成的脚本化运行手册内容，并在同一 AWS 账户和 AWS 区域中创建运行手册作为目标实例。现在，她已经准备好测试运行手册，以确保在将自动化实施到生产环境之前自动化能够按需运行。下面是完成的脚本化运行手册内容。

   YAML

```
description: An example of an Automation runbook that patches groups of Amazon EC2
 instances in stages.
schemaVersion: '0.3'
assumeRole: '{{AutomationAssumeRole}}'
parameters:
  AutomationAssumeRole:
    type: String
    description: '(Required) The Amazon Resource Name (ARN) of the IAM role that
 allows Automation to perform the actions on your behalf. If no role is specified,
 Systems Manager Automation uses your IAM permissions to operate this runbook.'
  PrimaryPatchGroupTag:
    type: StringMap
    description: '(Required) The tag for the primary group of instances you want
 to patch. Specify a key-value pair. Example: {"key" : "value"}'
```

```
  SecondaryPatchGroupTag:
    type: StringMap
    description: '(Required) The tag for the secondary group of instances you want
 to patch. Specify a key-value pair. Example: {"key" : "value"}'
  SnapshotId:
    type: String
    description: '(Optional) The snapshot ID to use to retrieve a patch baseline
 snapshot.'
    default: ''
  RebootOption:
    type: String
    description: '(Optional) Reboot behavior after a patch Install operation. If
 you choose NoReboot and patches are installed, the instance is marked as non-
compliant until a subsequent reboot and scan.'
    allowedValues:
      - NoReboot
      - RebootIfNeeded
    default: RebootIfNeeded
  Operation:
    type: String
    description: '(Optional) The update or configuration to perform on the
 instance. The system checks if patches specified in the patch baseline are
 installed on the instance. The install operation installs patches missing from
 the baseline.'
    allowedValues:
      - Install
      - Scan
    default: Install
mainSteps:
  - name: getPrimaryInstanceState
    action: 'aws:executeScript'
    timeoutSeconds: 120
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: getInstanceStates
      InputPayload:
        primaryTag: '{{PrimaryPatchGroupTag}}'
      Script: |-
        def getInstanceStates(events,context):
          import boto3

          #Initialize client
          ec2 = boto3.client('ec2')
```

```
          tag = events['primaryTag']
          tagKey, tagValue = list(tag.items())[0]
          instanceQuery = ec2.describe_instances(
          Filters=[
              {
                  "Name": "tag:" + tagKey,
                  "Values": [tagValue]
              }]
          )
          if not instanceQuery['Reservations']:
              noInstancesForTagString = "No instances found for specified tag."
              return({ 'noInstancesFound' : noInstancesForTagString })
          else:
              queryResponse = instanceQuery['Reservations']
              originalInstanceStates = {}
              for results in queryResponse:
                  instanceSet = results['Instances']
                  for instance in instanceSet:
                      instanceId = instance['InstanceId']
                      originalInstanceStates[instanceId] = instance['State']
['Name']
              return originalInstanceStates
    outputs:
      - Name: originalInstanceStates
        Selector: $.Payload
        Type: StringMap
    nextStep: verifyPrimaryInstancesRunning
  - name: verifyPrimaryInstancesRunning
    action: 'aws:executeScript'
    timeoutSeconds: 600
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: verifyInstancesRunning
      InputPayload:
        targetInstances: '{{getPrimaryInstanceState.originalInstanceStates}}'
      Script: |-
        def verifyInstancesRunning(events,context):
          import boto3

          #Initialize client
          ec2 = boto3.client('ec2')
          instanceDict = events['targetInstances']
          for instance in instanceDict:
```

```
            if instanceDict[instance] == 'stopped':
                print("The target instance " + instance + " is stopped. The
instance will now be started.")
                ec2.start_instances(
                    InstanceIds=[instance]
                    )
            elif instanceDict[instance] == 'stopping':
                print("The target instance " + instance + " is stopping. Polling
for instance to reach stopped state.")
                while instanceDict[instance] != 'stopped':
                    poll = ec2.get_waiter('instance_stopped')
                    poll.wait(
                        InstanceIds=[instance]
                    )
                ec2.start_instances(
                    InstanceIds=[instance]
                )
            else:
              pass
    nextStep: waitForPrimaryRunningInstances
  - name: waitForPrimaryRunningInstances
    action: 'aws:executeScript'
    timeoutSeconds: 300
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: waitForRunningInstances
      InputPayload:
        targetInstances: '{{getPrimaryInstanceState.originalInstanceStates}}'
      Script: |-
        def waitForRunningInstances(events,context):
          import boto3

          #Initialize client
          ec2 = boto3.client('ec2')
          instanceDict = events['targetInstances']
          for instance in instanceDict:
              poll = ec2.get_waiter('instance_running')
              poll.wait(
                  InstanceIds=[instance]
              )
    nextStep: returnPrimaryTagKey
  - name: returnPrimaryTagKey
    action: 'aws:executeScript'
```

```
    timeoutSeconds: 120
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: returnTagValues
      InputPayload:
        primaryTag: '{{PrimaryPatchGroupTag}}'
      Script: |-
        def returnTagValues(events,context):
          tag = events['primaryTag']
          tagKey = list(tag)[0]
          stringKey = "tag:" + tagKey
          return {'tagKey' : stringKey}
    outputs:
      - Name: Payload
        Selector: $.Payload
        Type: StringMap
      - Name: primaryPatchGroupKey
        Selector: $.Payload.tagKey
        Type: String
    nextStep: returnPrimaryTagValue
  - name: returnPrimaryTagValue
    action: 'aws:executeScript'
    timeoutSeconds: 120
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: returnTagValues
      InputPayload:
        primaryTag: '{{PrimaryPatchGroupTag}}'
      Script: |-
        def returnTagValues(events,context):
          tag = events['primaryTag']
          tagKey = list(tag)[0]
          tagValue = tag[tagKey]
          return {'tagValue' : tagValue}
    outputs:
      - Name: Payload
        Selector: $.Payload
        Type: StringMap
      - Name: primaryPatchGroupValue
        Selector: $.Payload.tagValue
        Type: String
    nextStep: patchPrimaryInstances
```

```
  - name: patchPrimaryInstances
    action: 'aws:runCommand'
    onFailure: Abort
    timeoutSeconds: 7200
    inputs:
      DocumentName: AWS-RunPatchBaseline
      Parameters:
        SnapshotId: '{{SnapshotId}}'
        RebootOption: '{{RebootOption}}'
        Operation: '{{Operation}}'
      Targets:
        - Key: '{{returnPrimaryTagKey.primaryPatchGroupKey}}'
          Values:
            - '{{returnPrimaryTagValue.primaryPatchGroupValue}}'
      MaxConcurrency: 10%
      MaxErrors: 10%
    nextStep: returnPrimaryToOriginalState
  - name: returnPrimaryToOriginalState
    action: 'aws:executeScript'
    timeoutSeconds: 600
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: returnToOriginalState
      InputPayload:
        targetInstances: '{{getPrimaryInstanceState.originalInstanceStates}}'
      Script: |-
        def returnToOriginalState(events,context):
          import boto3

          #Initialize client
          ec2 = boto3.client('ec2')
          instanceDict = events['targetInstances']
          for instance in instanceDict:
            if instanceDict[instance] == 'stopped' or instanceDict[instance] ==
'stopping':
                ec2.stop_instances(
                    InstanceIds=[instance]
                    )
            else:
              pass
    nextStep: getSecondaryInstanceState
  - name: getSecondaryInstanceState
    action: 'aws:executeScript'
```

```
    timeoutSeconds: 120
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: getInstanceStates
      InputPayload:
        secondaryTag: '{{SecondaryPatchGroupTag}}'
      Script: |-
        def getInstanceStates(events,context):
          import boto3

          #Initialize client
          ec2 = boto3.client('ec2')
          tag = events['secondaryTag']
          tagKey, tagValue = list(tag.items())[0]
          instanceQuery = ec2.describe_instances(
          Filters=[
              {
                  "Name": "tag:" + tagKey,
                  "Values": [tagValue]
              }]
          )
          if not instanceQuery['Reservations']:
              noInstancesForTagString = "No instances found for specified tag."
              return({ 'noInstancesFound' : noInstancesForTagString })
          else:
              queryResponse = instanceQuery['Reservations']
              originalInstanceStates = {}
              for results in queryResponse:
                  instanceSet = results['Instances']
                  for instance in instanceSet:
                      instanceId = instance['InstanceId']
                      originalInstanceStates[instanceId] = instance['State']
['Name']
              return originalInstanceStates
    outputs:
      - Name: originalInstanceStates
        Selector: $.Payload
        Type: StringMap
    nextStep: verifySecondaryInstancesRunning
  - name: verifySecondaryInstancesRunning
    action: 'aws:executeScript'
    timeoutSeconds: 600
    onFailure: Abort
```

```
    inputs:
      Runtime: python3.7
      Handler: verifyInstancesRunning
      InputPayload:
        targetInstances: '{{getSecondaryInstanceState.originalInstanceStates}}'
      Script: |-
        def verifyInstancesRunning(events,context):
          import boto3

          #Initialize client
          ec2 = boto3.client('ec2')
          instanceDict = events['targetInstances']
          for instance in instanceDict:
            if instanceDict[instance] == 'stopped':
                print("The target instance " + instance + " is stopped. The
instance will now be started.")
                ec2.start_instances(
                    InstanceIds=[instance]
                    )
            elif instanceDict[instance] == 'stopping':
                print("The target instance " + instance + " is stopping. Polling
for instance to reach stopped state.")
                while instanceDict[instance] != 'stopped':
                    poll = ec2.get_waiter('instance_stopped')
                    poll.wait(
                        InstanceIds=[instance]
                    )
                ec2.start_instances(
                    InstanceIds=[instance]
                )
            else:
              pass
    nextStep: waitForSecondaryRunningInstances
  - name: waitForSecondaryRunningInstances
    action: 'aws:executeScript'
    timeoutSeconds: 300
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: waitForRunningInstances
      InputPayload:
        targetInstances: '{{getSecondaryInstanceState.originalInstanceStates}}'
      Script: |-
        def waitForRunningInstances(events,context):
```

```
          import boto3

          #Initialize client
          ec2 = boto3.client('ec2')
          instanceDict = events['targetInstances']
          for instance in instanceDict:
              poll = ec2.get_waiter('instance_running')
              poll.wait(
                  InstanceIds=[instance]
              )
    nextStep: returnSecondaryTagKey
  - name: returnSecondaryTagKey
    action: 'aws:executeScript'
    timeoutSeconds: 120
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: returnTagValues
      InputPayload:
        secondaryTag: '{{SecondaryPatchGroupTag}}'
      Script: |-
        def returnTagValues(events,context):
          tag = events['secondaryTag']
          tagKey = list(tag)[0]
          stringKey = "tag:" + tagKey
          return {'tagKey' : stringKey}
    outputs:
      - Name: Payload
        Selector: $.Payload
        Type: StringMap
      - Name: secondaryPatchGroupKey
        Selector: $.Payload.tagKey
        Type: String
    nextStep: returnSecondaryTagValue
  - name: returnSecondaryTagValue
    action: 'aws:executeScript'
    timeoutSeconds: 120
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: returnTagValues
      InputPayload:
        secondaryTag: '{{SecondaryPatchGroupTag}}'
      Script: |-
```

```
        def returnTagValues(events,context):
          tag = events['secondaryTag']
          tagKey = list(tag)[0]
          tagValue = tag[tagKey]
          return {'tagValue' : tagValue}
    outputs:
      - Name: Payload
        Selector: $.Payload
        Type: StringMap
      - Name: secondaryPatchGroupValue
        Selector: $.Payload.tagValue
        Type: String
    nextStep: patchSecondaryInstances
  - name: patchSecondaryInstances
    action: 'aws:runCommand'
    onFailure: Abort
    timeoutSeconds: 7200
    inputs:
      DocumentName: AWS-RunPatchBaseline
      Parameters:
        SnapshotId: '{{SnapshotId}}'
        RebootOption: '{{RebootOption}}'
        Operation: '{{Operation}}'
      Targets:
        - Key: '{{returnSecondaryTagKey.secondaryPatchGroupKey}}'
          Values:
          - '{{returnSecondaryTagValue.secondaryPatchGroupValue}}'
      MaxConcurrency: 10%
      MaxErrors: 10%
    nextStep: returnSecondaryToOriginalState
  - name: returnSecondaryToOriginalState
    action: 'aws:executeScript'
    timeoutSeconds: 600
    onFailure: Abort
    inputs:
      Runtime: python3.7
      Handler: returnToOriginalState
      InputPayload:
        targetInstances: '{{getSecondaryInstanceState.originalInstanceStates}}'
      Script: |-
        def returnToOriginalState(events,context):
          import boto3

          #Initialize client
```

```
          ec2 = boto3.client('ec2')
          instanceDict = events['targetInstances']
          for instance in instanceDict:
            if instanceDict[instance] == 'stopped' or instanceDict[instance] ==
 'stopping':
                ec2.stop_instances(
                    InstanceIds=[instance]
                    )
            else:
              pass
```

JSON

```
{
   "description":"An example of an Automation runbook that patches groups of
 Amazon EC2 instances in stages.",
   "schemaVersion":"0.3",
   "assumeRole":"{{AutomationAssumeRole}}",
   "parameters":{
      "AutomationAssumeRole":{
         "type":"String",
         "description":"(Required) The Amazon Resource Name (ARN) of the IAM role
 that allows Automation to perform the actions on your behalf. If no role is
 specified, Systems Manager Automation uses your IAM permissions to operate this
 runbook."
      },
      "PrimaryPatchGroupTag":{
         "type":"StringMap",
         "description":"(Required) The tag for the primary group of instances you
 want to patch. Specify a key-value pair. Example: {\"key\" : \"value\"}"
      },
      "SecondaryPatchGroupTag":{
         "type":"StringMap",
         "description":"(Required) The tag for the secondary group of instances
 you want to patch. Specify a key-value pair. Example: {\"key\" : \"value\"}"
      },
      "SnapshotId":{
         "type":"String",
         "description":"(Optional) The snapshot ID to use to retrieve a patch
 baseline snapshot.",
         "default":""
      },
      "RebootOption":{
```

```
        "type":"String",
        "description":"(Optional) Reboot behavior after a patch Install
operation. If you choose NoReboot and patches are installed, the instance is
marked as non-compliant until a subsequent reboot and scan.",
        "allowedValues":[
           "NoReboot",
           "RebootIfNeeded"
        ],
        "default":"RebootIfNeeded"
     },
     "Operation":{
        "type":"String",
        "description":"(Optional) The update or configuration to perform on
the instance. The system checks if patches specified in the patch baseline are
installed on the instance. The install operation installs patches missing from
the baseline.",
        "allowedValues":[
           "Install",
           "Scan"
        ],
        "default":"Install"
     }
  },
  "mainSteps":[
     {
        "name":"getPrimaryInstanceState",
        "action":"aws:executeScript",
        "timeoutSeconds":120,
        "onFailure":"Abort",
        "inputs":{
           "Runtime":"python3.7",
           "Handler":"getInstanceStates",
           "InputPayload":{
              "primaryTag":"{{PrimaryPatchGroupTag}}"
           },
           "Script":"..."
        },
        "outputs":[
           {
              "Name":"originalInstanceStates",
              "Selector":"$.Payload",
              "Type":"StringMap"
           }
        ],
```

```
            "nextStep":"verifyPrimaryInstancesRunning"
        },
        {
            "name":"verifyPrimaryInstancesRunning",
            "action":"aws:executeScript",
            "timeoutSeconds":600,
            "onFailure":"Abort",
            "inputs":{
               "Runtime":"python3.7",
               "Handler":"verifyInstancesRunning",
               "InputPayload":{

"targetInstances":"{{getPrimaryInstanceState.originalInstanceStates}}"
               },
               "Script":"..."
            },
            "nextStep":"waitForPrimaryRunningInstances"
        },
        {
            "name":"waitForPrimaryRunningInstances",
            "action":"aws:executeScript",
            "timeoutSeconds":300,
            "onFailure":"Abort",
            "inputs":{
               "Runtime":"python3.7",
               "Handler":"waitForRunningInstances",
               "InputPayload":{

"targetInstances":"{{getPrimaryInstanceState.originalInstanceStates}}"
               },
               "Script":"..."
            },
            "nextStep":"returnPrimaryTagKey"
        },
        {
            "name":"returnPrimaryTagKey",
            "action":"aws:executeScript",
            "timeoutSeconds":120,
            "onFailure":"Abort",
            "inputs":{
               "Runtime":"python3.7",
               "Handler":"returnTagValues",
               "InputPayload":{
                  "primaryTag":"{{PrimaryPatchGroupTag}}"
```

```
            },
            "Script":"..."
         },
         "outputs":[
            {
               "Name":"Payload",
               "Selector":"$.Payload",
               "Type":"StringMap"
            },
            {
               "Name":"primaryPatchGroupKey",
               "Selector":"$.Payload.tagKey",
               "Type":"String"
            }
         ],
         "nextStep":"returnPrimaryTagValue"
      },
      {
         "name":"returnPrimaryTagValue",
         "action":"aws:executeScript",
         "timeoutSeconds":120,
         "onFailure":"Abort",
         "inputs":{
            "Runtime":"python3.7",
            "Handler":"returnTagValues",
            "InputPayload":{
               "primaryTag":"{{PrimaryPatchGroupTag}}"
            },
            "Script":"..."
         },
         "outputs":[
            {
               "Name":"Payload",
               "Selector":"$.Payload",
               "Type":"StringMap"
            },
            {
               "Name":"primaryPatchGroupValue",
               "Selector":"$.Payload.tagValue",
               "Type":"String"
            }
         ],
         "nextStep":"patchPrimaryInstances"
      },
```

```
      {
         "name":"patchPrimaryInstances",
         "action":"aws:runCommand",
         "onFailure":"Abort",
         "timeoutSeconds":7200,
         "inputs":{
            "DocumentName":"AWS-RunPatchBaseline",
            "Parameters":{
               "SnapshotId":"{{SnapshotId}}",
               "RebootOption":"{{RebootOption}}",
               "Operation":"{{Operation}}"
            },
            "Targets":[
               {
                  "Key":"{{returnPrimaryTagKey.primaryPatchGroupKey}}",
                  "Values":[
                     "{{returnPrimaryTagValue.primaryPatchGroupValue}}"
                  ]
               }
            ],
            "MaxConcurrency":"10%",
            "MaxErrors":"10%"
         },
         "nextStep":"returnPrimaryToOriginalState"
      },
      {
         "name":"returnPrimaryToOriginalState",
         "action":"aws:executeScript",
         "timeoutSeconds":600,
         "onFailure":"Abort",
         "inputs":{
            "Runtime":"python3.7",
            "Handler":"returnToOriginalState",
            "InputPayload":{

"targetInstances":"{{getPrimaryInstanceState.originalInstanceStates}}"
            },
            "Script":"..."
         },
         "nextStep":"getSecondaryInstanceState"
      },
      {
         "name":"getSecondaryInstanceState",
         "action":"aws:executeScript",
```

```
         "timeoutSeconds":120,
         "onFailure":"Abort",
         "inputs":{
            "Runtime":"python3.7",
            "Handler":"getInstanceStates",
            "InputPayload":{
               "secondaryTag":"{{SecondaryPatchGroupTag}}"
            },
            "Script":"..."
         },
         "outputs":[
            {
               "Name":"originalInstanceStates",
               "Selector":"$.Payload",
               "Type":"StringMap"
            }
         ],
         "nextStep":"verifySecondaryInstancesRunning"
      },
      {
         "name":"verifySecondaryInstancesRunning",
         "action":"aws:executeScript",
         "timeoutSeconds":600,
         "onFailure":"Abort",
         "inputs":{
            "Runtime":"python3.7",
            "Handler":"verifyInstancesRunning",
            "InputPayload":{

"targetInstances":"{{getSecondaryInstanceState.originalInstanceStates}}"
            },
            "Script":"..."
         },
         "nextStep":"waitForSecondaryRunningInstances"
      },
      {
         "name":"waitForSecondaryRunningInstances",
         "action":"aws:executeScript",
         "timeoutSeconds":300,
         "onFailure":"Abort",
         "inputs":{
            "Runtime":"python3.7",
            "Handler":"waitForRunningInstances",
            "InputPayload":{
```

```
"targetInstances":"{{getSecondaryInstanceState.originalInstanceStates}}"
            },
            "Script":"..."
         },
         "nextStep":"returnSecondaryTagKey"
      },
      {
         "name":"returnSecondaryTagKey",
         "action":"aws:executeScript",
         "timeoutSeconds":120,
         "onFailure":"Abort",
         "inputs":{
            "Runtime":"python3.7",
            "Handler":"returnTagValues",
            "InputPayload":{
               "secondaryTag":"{{SecondaryPatchGroupTag}}"
            },
            "Script":"..."
         },
         "outputs":[
            {
               "Name":"Payload",
               "Selector":"$.Payload",
               "Type":"StringMap"
            },
            {
               "Name":"secondaryPatchGroupKey",
               "Selector":"$.Payload.tagKey",
               "Type":"String"
            }
         ],
         "nextStep":"returnSecondaryTagValue"
      },
      {
         "name":"returnSecondaryTagValue",
         "action":"aws:executeScript",
         "timeoutSeconds":120,
         "onFailure":"Abort",
         "inputs":{
            "Runtime":"python3.7",
            "Handler":"returnTagValues",
            "InputPayload":{
               "secondaryTag":"{{SecondaryPatchGroupTag}}"
```

```
            },
            "Script":"..."
         },
         "outputs":[
            {
               "Name":"Payload",
               "Selector":"$.Payload",
               "Type":"StringMap"
            },
            {
               "Name":"secondaryPatchGroupValue",
               "Selector":"$.Payload.tagValue",
               "Type":"String"
            }
         ],
         "nextStep":"patchSecondaryInstances"
      },
      {
         "name":"patchSecondaryInstances",
         "action":"aws:runCommand",
         "onFailure":"Abort",
         "timeoutSeconds":7200,
         "inputs":{
            "DocumentName":"AWS-RunPatchBaseline",
            "Parameters":{
               "SnapshotId":"{{SnapshotId}}",
               "RebootOption":"{{RebootOption}}",
               "Operation":"{{Operation}}"
            },
            "Targets":[
               {
                  "Key":"{{returnSecondaryTagKey.secondaryPatchGroupKey}}",
                  "Values":[
                     "{{returnSecondaryTagValue.secondaryPatchGroupValue}}"
                  ]
               }
            ],
            "MaxConcurrency":"10%",
            "MaxErrors":"10%"
         },
         "nextStep":"returnSecondaryToOriginalState"
      },
      {
         "name":"returnSecondaryToOriginalState",
```

```
            "action":"aws:executeScript",
            "timeoutSeconds":600,
            "onFailure":"Abort",
            "inputs":{
               "Runtime":"python3.7",
               "Handler":"returnToOriginalState",
               "InputPayload":{

 "targetInstances":"{{getSecondaryInstanceState.originalInstanceStates}}"
               },
               "Script":"..."
            }
         }
      ]
}
```

有关此示例中使用的自动化操作的更多信息，请参阅 Systems Manager 自动化操作参考。

其他运行手册示例

以下示例运行手册演示了如何使用 AWS Systems Manager 自动化操作来自动执行常见部署、故障排除和维护任务。

> **Note**
>
> 本部分提供的示例运行手册旨在演示如何创建自定义运行手册以支持您的特定操作需求。这些运行手册并非供您在生产环境中原样使用。但是，您可以自定义这些文档来满足自己的使用需求。

示例

- 部署 VPC 架构和 Microsoft Active Directory 域控制器
- 从最新快照还原根卷
- 创建 AMI 和跨区域副本

部署 VPC 架构和 Microsoft Active Directory 域控制器

要提高效率并实现常见任务的标准化，可以选择自动执行部署。如果您定期在多个账户和 AWS 区域中部署相同的架构，此操作会很有帮助。自动进行架构部署还可以降低在手动部署架构时发生人为错

误的可能性。AWS Systems Manager自动化操作可帮助您做到这一点。Automation 是 AWS Systems Manager 中的一项工具。

以下示例 AWS Systems Manager 运行手册执行这些操作：

- 使用 Systems Manager Parameter Store 检索最新的 Windows Server 2016 Amazon Machine Image（AMI），在启动将配置作为域控制器的 EC2 实例时使用。Parameter Store 是 AWS Systems Manager 中的一项工具。
- 使用 `aws:executeAwsApi` 自动化操作调用多个 AWS API 操作来创建 VPC 架构。域控制器实例在私有子网中启动，使用 NAT 网关连接到 Internet。这允许实例上的 SSM Agent 以访问必需的 Systems Manager 端点。
- 使用 `aws:waitForAwsResourceProperty` 自动化操作确认上一个操作所启动的实例对于 `Online` 处于 AWS Systems Manager 状态。
- 使用 `aws:runCommand` 自动化操作配置作为 Microsoft Active Directory 域控制器启动的实例。

YAML

```
---
description: Custom Automation Deployment Example
schemaVersion: '0.3'
parameters:
  AutomationAssumeRole:
    type: String
    default: ''
    description: >-
      (Optional) The ARN of the role that allows Automation to perform the
      actions on your behalf. If no role is specified, Systems Manager
      Automation uses your IAM permissions to run this runbook.
mainSteps:
  - name: getLatestWindowsAmi
    action: aws:executeAwsApi
    onFailure: Abort
    inputs:
      Service: ssm
      Api: GetParameter
      Name: >-
        /aws/service/ami-windows-latest/Windows_Server-2016-English-Full-Base
    outputs:
      - Name: amiId
```

```
            Selector: $.Parameter.Value
            Type: String
        nextStep: createSSMInstanceRole
      - name: createSSMInstanceRole
        action: aws:executeAwsApi
        onFailure: Abort
        inputs:
          Service: iam
          Api: CreateRole
          AssumeRolePolicyDocument: >-
            {"Version":"2012-10-17","Statement":[{"Effect":"Allow","Principal":
{"Service":["ec2.amazonaws.com"]},"Action":["sts:AssumeRole"]}]}
          RoleName: sampleSSMInstanceRole
        nextStep: attachManagedSSMPolicy
      - name: attachManagedSSMPolicy
        action: aws:executeAwsApi
        onFailure: Abort
        inputs:
          Service: iam
          Api: AttachRolePolicy
          PolicyArn: 'arn:aws:iam::aws:policy/service-role/
AmazonSSMManagedInstanceCore'
          RoleName: sampleSSMInstanceRole
        nextStep: createSSMInstanceProfile
      - name: createSSMInstanceProfile
        action: aws:executeAwsApi
        onFailure: Abort
        inputs:
          Service: iam
          Api: CreateInstanceProfile
          InstanceProfileName: sampleSSMInstanceRole
        outputs:
          - Name: instanceProfileArn
            Selector: $.InstanceProfile.Arn
            Type: String
        nextStep: addSSMInstanceRoleToProfile
      - name: addSSMInstanceRoleToProfile
        action: aws:executeAwsApi
        onFailure: Abort
        inputs:
          Service: iam
          Api: AddRoleToInstanceProfile
          InstanceProfileName: sampleSSMInstanceRole
          RoleName: sampleSSMInstanceRole
```

```
      nextStep: createVpc
    - name: createVpc
      action: aws:executeAwsApi
      onFailure: Abort
      inputs:
        Service: ec2
        Api: CreateVpc
        CidrBlock: 10.0.100.0/22
      outputs:
        - Name: vpcId
          Selector: $.Vpc.VpcId
          Type: String
      nextStep: getMainRtb
    - name: getMainRtb
      action: aws:executeAwsApi
      onFailure: Abort
      inputs:
        Service: ec2
        Api: DescribeRouteTables
        Filters:
          - Name: vpc-id
            Values:
              - '{{ createVpc.vpcId }}'
      outputs:
        - Name: mainRtbId
          Selector: '$.RouteTables[0].RouteTableId'
          Type: String
      nextStep: verifyMainRtb
    - name: verifyMainRtb
      action: aws:assertAwsResourceProperty
      onFailure: Abort
      inputs:
        Service: ec2
        Api: DescribeRouteTables
        RouteTableIds:
          - '{{ getMainRtb.mainRtbId }}'
        PropertySelector: '$.RouteTables[0].Associations[0].Main'
        DesiredValues:
          - 'True'
      nextStep: createPubSubnet
    - name: createPubSubnet
      action: aws:executeAwsApi
      onFailure: Abort
      inputs:
```

```
        Service: ec2
        Api: CreateSubnet
        CidrBlock: 10.0.103.0/24
        AvailabilityZone: us-west-2c
        VpcId: '{{ createVpc.vpcId }}'
      outputs:
        - Name: pubSubnetId
          Selector: $.Subnet.SubnetId
          Type: String
      nextStep: createPubRtb
    - name: createPubRtb
      action: aws:executeAwsApi
      onFailure: Abort
      inputs:
        Service: ec2
        Api: CreateRouteTable
        VpcId: '{{ createVpc.vpcId }}'
      outputs:
        - Name: pubRtbId
          Selector: $.RouteTable.RouteTableId
          Type: String
      nextStep: createIgw
    - name: createIgw
      action: aws:executeAwsApi
      onFailure: Abort
      inputs:
        Service: ec2
        Api: CreateInternetGateway
      outputs:
        - Name: igwId
          Selector: $.InternetGateway.InternetGatewayId
          Type: String
      nextStep: attachIgw
    - name: attachIgw
      action: aws:executeAwsApi
      onFailure: Abort
      inputs:
        Service: ec2
        Api: AttachInternetGateway
        InternetGatewayId: '{{ createIgw.igwId }}'
        VpcId: '{{ createVpc.vpcId }}'
      nextStep: allocateEip
    - name: allocateEip
      action: aws:executeAwsApi
```

```
    onFailure: Abort
    inputs:
      Service: ec2
      Api: AllocateAddress
      Domain: vpc
    outputs:
      - Name: eipAllocationId
        Selector: $.AllocationId
        Type: String
    nextStep: createNatGw
  - name: createNatGw
    action: aws:executeAwsApi
    onFailure: Abort
    inputs:
      Service: ec2
      Api: CreateNatGateway
      AllocationId: '{{ allocateEip.eipAllocationId }}'
      SubnetId: '{{ createPubSubnet.pubSubnetId }}'
    outputs:
      - Name: natGwId
        Selector: $.NatGateway.NatGatewayId
        Type: String
    nextStep: verifyNatGwAvailable
  - name: verifyNatGwAvailable
    action: aws:waitForAwsResourceProperty
    timeoutSeconds: 150
    inputs:
      Service: ec2
      Api: DescribeNatGateways
      NatGatewayIds:
        - '{{ createNatGw.natGwId }}'
      PropertySelector: '$.NatGateways[0].State'
      DesiredValues:
        - available
    nextStep: createNatRoute
  - name: createNatRoute
    action: aws:executeAwsApi
    onFailure: Abort
    inputs:
      Service: ec2
      Api: CreateRoute
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId: '{{ createNatGw.natGwId }}'
      RouteTableId: '{{ getMainRtb.mainRtbId }}'
```

```
      nextStep: createPubRoute
    - name: createPubRoute
      action: aws:executeAwsApi
      onFailure: Abort
      inputs:
        Service: ec2
        Api: CreateRoute
        DestinationCidrBlock: 0.0.0.0/0
        GatewayId: '{{ createIgw.igwId }}'
        RouteTableId: '{{ createPubRtb.pubRtbId }}'
      nextStep: setPubSubAssoc
    - name: setPubSubAssoc
      action: aws:executeAwsApi
      onFailure: Abort
      inputs:
        Service: ec2
        Api: AssociateRouteTable
        RouteTableId: '{{ createPubRtb.pubRtbId }}'
        SubnetId: '{{ createPubSubnet.pubSubnetId }}'
    - name: createDhcpOptions
      action: aws:executeAwsApi
      onFailure: Abort
      inputs:
        Service: ec2
        Api: CreateDhcpOptions
        DhcpConfigurations:
          - Key: domain-name-servers
            Values:
              - '10.0.100.50,10.0.101.50'
          - Key: domain-name
            Values:
              - sample.com
      outputs:
        - Name: dhcpOptionsId
          Selector: $.DhcpOptions.DhcpOptionsId
          Type: String
      nextStep: createDCSubnet1
    - name: createDCSubnet1
      action: aws:executeAwsApi
      onFailure: Abort
      inputs:
        Service: ec2
        Api: CreateSubnet
        CidrBlock: 10.0.100.0/24
```

```
        AvailabilityZone: us-west-2a
        VpcId: '{{ createVpc.vpcId }}'
      outputs:
        - Name: firstSubnetId
          Selector: $.Subnet.SubnetId
          Type: String
      nextStep: createDCSubnet2
    - name: createDCSubnet2
      action: aws:executeAwsApi
      onFailure: Abort
      inputs:
        Service: ec2
        Api: CreateSubnet
        CidrBlock: 10.0.101.0/24
        AvailabilityZone: us-west-2b
        VpcId: '{{ createVpc.vpcId }}'
      outputs:
        - Name: secondSubnetId
          Selector: $.Subnet.SubnetId
          Type: String
      nextStep: createDCSecGroup
    - name: createDCSecGroup
      action: aws:executeAwsApi
      onFailure: Abort
      inputs:
        Service: ec2
        Api: CreateSecurityGroup
        GroupName: SampleDCSecGroup
        Description: Security Group for Sample Domain Controllers
        VpcId: '{{ createVpc.vpcId }}'
      outputs:
        - Name: dcSecGroupId
          Selector: $.GroupId
          Type: String
      nextStep: authIngressDCTraffic
    - name: authIngressDCTraffic
      action: aws:executeAwsApi
      onFailure: Abort
      inputs:
        Service: ec2
        Api: AuthorizeSecurityGroupIngress
        GroupId: '{{ createDCSecGroup.dcSecGroupId }}'
        IpPermissions:
          - FromPort: -1
```

```
            IpProtocol: '-1'
            IpRanges:
              - CidrIp: 0.0.0.0/0
                Description: Allow all traffic between Domain Controllers
    nextStep: verifyInstanceProfile
  - name: verifyInstanceProfile
    action: aws:waitForAwsResourceProperty
    maxAttempts: 5
    onFailure: Abort
    inputs:
      Service: iam
      Api: ListInstanceProfilesForRole
      RoleName: sampleSSMInstanceRole
      PropertySelector: '$.InstanceProfiles[0].Arn'
      DesiredValues:
        - '{{ createSSMInstanceProfile.instanceProfileArn }}'
    nextStep: iamEventualConsistency
  - name: iamEventualConsistency
    action: aws:sleep
    inputs:
      Duration: PT2M
    nextStep: launchDC1
  - name: launchDC1
    action: aws:executeAwsApi
    onFailure: Abort
    inputs:
      Service: ec2
      Api: RunInstances
      BlockDeviceMappings:
        - DeviceName: /dev/sda1
          Ebs:
            DeleteOnTermination: true
            VolumeSize: 50
            VolumeType: gp2
        - DeviceName: xvdf
          Ebs:
            DeleteOnTermination: true
            VolumeSize: 100
            VolumeType: gp2
      IamInstanceProfile:
        Arn: '{{ createSSMInstanceProfile.instanceProfileArn }}'
      ImageId: '{{ getLatestWindowsAmi.amiId }}'
      InstanceType: t2.micro
      MaxCount: 1
```

```
      MinCount: 1
      PrivateIpAddress: 10.0.100.50
      SecurityGroupIds:
        - '{{ createDCSecGroup.dcSecGroupId }}'
      SubnetId: '{{ createDCSubnet1.firstSubnetId }}'
      TagSpecifications:
        - ResourceType: instance
          Tags:
            - Key: Name
              Value: SampleDC1
    outputs:
      - Name: pdcInstanceId
        Selector: '$.Instances[0].InstanceId'
        Type: String
    nextStep: launchDC2
  - name: launchDC2
    action: aws:executeAwsApi
    onFailure: Abort
    inputs:
      Service: ec2
      Api: RunInstances
      BlockDeviceMappings:
        - DeviceName: /dev/sda1
          Ebs:
            DeleteOnTermination: true
            VolumeSize: 50
            VolumeType: gp2
        - DeviceName: xvdf
          Ebs:
            DeleteOnTermination: true
            VolumeSize: 100
            VolumeType: gp2
      IamInstanceProfile:
        Arn: '{{ createSSMInstanceProfile.instanceProfileArn }}'
      ImageId: '{{ getLatestWindowsAmi.amiId }}'
      InstanceType: t2.micro
      MaxCount: 1
      MinCount: 1
      PrivateIpAddress: 10.0.101.50
      SecurityGroupIds:
        - '{{ createDCSecGroup.dcSecGroupId }}'
      SubnetId: '{{ createDCSubnet2.secondSubnetId }}'
      TagSpecifications:
        - ResourceType: instance
```

```
            Tags:
              - Key: Name
                Value: SampleDC2
      outputs:
        - Name: adcInstanceId
          Selector: '$.Instances[0].InstanceId'
          Type: String
      nextStep: verifyDCInstanceState
    - name: verifyDCInstanceState
      action: aws:waitForAwsResourceProperty
      inputs:
        Service: ec2
        Api: DescribeInstanceStatus
        IncludeAllInstances: true
        InstanceIds:
          - '{{ launchDC1.pdcInstanceId }}'
          - '{{ launchDC2.adcInstanceId }}'
        PropertySelector: '$.InstanceStatuses..InstanceState.Name'
        DesiredValues:
          - running
      nextStep: verifyInstancesOnlineSSM
    - name: verifyInstancesOnlineSSM
      action: aws:waitForAwsResourceProperty
      timeoutSeconds: 600
      inputs:
        Service: ssm
        Api: DescribeInstanceInformation
        InstanceInformationFilterList:
          - key: InstanceIds
            valueSet:
              - '{{ launchDC1.pdcInstanceId }}'
              - '{{ launchDC2.adcInstanceId }}'
        PropertySelector: '$.InstanceInformationList..PingStatus'
        DesiredValues:
          - Online
      nextStep: installADRoles
    - name: installADRoles
      action: aws:runCommand
      inputs:
        DocumentName: AWS-RunPowerShellScript
        InstanceIds:
          - '{{ launchDC1.pdcInstanceId }}'
          - '{{ launchDC2.adcInstanceId }}'
        Parameters:
```

```
            commands: |-
              try {
                  Install-WindowsFeature -Name AD-Domain-Services -
IncludeManagementTools
              }
              catch {
                  Write-Error "Failed to install ADDS Role."
              }
        nextStep: setAdminPassword
      - name: setAdminPassword
        action: aws:runCommand
        inputs:
          DocumentName: AWS-RunPowerShellScript
          InstanceIds:
            - '{{ launchDC1.pdcInstanceId }}'
          Parameters:
            commands:
              - net user Administrator "sampleAdminPass123!"
        nextStep: createForest
      - name: createForest
        action: aws:runCommand
        inputs:
          DocumentName: AWS-RunPowerShellScript
          InstanceIds:
            - '{{ launchDC1.pdcInstanceId }}'
          Parameters:
            commands: |-
              $dsrmPass = 'sample123!' | ConvertTo-SecureString -asPlainText -Force
              try {
                  Install-ADDSForest -DomainName "sample.com" -DomainMode 6
 -ForestMode 6 -InstallDNS -DatabasePath "D:\NTDS" -SysvolPath "D:\SYSVOL" -
SafeModeAdministratorPassword $dsrmPass -Force
              }
              catch {
                  Write-Error $_
              }
              try {
                  Add-DnsServerForwarder -IPAddress "10.0.100.2"
              }
              catch {
                  Write-Error $_
              }
        nextStep: associateDhcpOptions
      - name: associateDhcpOptions
```

```
        action: aws:executeAwsApi
        onFailure: Abort
        inputs:
          Service: ec2
          Api: AssociateDhcpOptions
          DhcpOptionsId: '{{ createDhcpOptions.dhcpOptionsId }}'
          VpcId: '{{ createVpc.vpcId }}'
        nextStep: waitForADServices
      - name: waitForADServices
        action: aws:sleep
        inputs:
          Duration: PT1M
        nextStep: promoteADC
      - name: promoteADC
        action: aws:runCommand
        inputs:
          DocumentName: AWS-RunPowerShellScript
          InstanceIds:
            - '{{ launchDC2.adcInstanceId }}'
          Parameters:
            commands: |-
              ipconfig /renew
              $dsrmPass = 'sample123!' | ConvertTo-SecureString -asPlainText -Force
              $domAdminUser = "sample\Administrator"
              $domAdminPass = "sampleAdminPass123!" | ConvertTo-SecureString -
asPlainText -Force
              $domAdminCred = New-Object
 System.Management.Automation.PSCredential($domAdminUser,$domAdminPass)

              try {
                  Install-ADDSDomainController -DomainName "sample.com" -InstallDNS
 -DatabasePath "D:\NTDS" -SysvolPath "D:\SYSVOL" -SafeModeAdministratorPassword
 $dsrmPass -Credential $domAdminCred -Force
              }
              catch {
                  Write-Error $_
              }
```

JSON

JSON

```
    {
      "description": "Custom Automation Deployment Example",
      "schemaVersion": "0.3",
      "assumeRole": "{{ AutomationAssumeRole }}",
      "parameters": {
        "AutomationAssumeRole": {
          "type": "String",
          "description": "(Optional) The ARN of the role that allows
 Automation to perform the actions on your behalf. If no role is specified,
 Systems Manager Automation uses your IAM permissions to run this runbook.",
          "default": ""
        }
      },
      "mainSteps": [
        {
          "name": "getLatestWindowsAmi",
          "action": "aws:executeAwsApi",
          "onFailure": "Abort",
          "inputs": {
            "Service": "ssm",
            "Api": "GetParameter",
            "Name": "/aws/service/ami-windows-latest/Windows_Server-2016-
English-Full-Base"
          },
          "outputs": [
            {
              "Name": "amiId",
              "Selector": "$.Parameter.Value",
              "Type": "String"
            }
          ],
          "nextStep": "createSSMInstanceRole"
        },
        {
          "name": "createSSMInstanceRole",
          "action": "aws:executeAwsApi",
          "onFailure": "Abort",
          "inputs": {
            "Service": "iam",
            "Api": "CreateRole",
            "AssumeRolePolicyDocument": "{\"Version\":\"2012-10-17\",
\"Statement\":[{\"Effect\":\"Allow\",\"Principal\":{\"Service\":
[\"ec2.amazonaws.com\"]},\"Action\":[\"sts:AssumeRole\"]}]}",
            "RoleName": "sampleSSMInstanceRole"
```

```
          },
          "nextStep": "attachManagedSSMPolicy"
        },
        {
          "name": "attachManagedSSMPolicy",
          "action": "aws:executeAwsApi",
          "onFailure": "Abort",
          "inputs": {
            "Service": "iam",
            "Api": "AttachRolePolicy",
            "PolicyArn": "arn:aws:iam::aws:policy/service-role/
AmazonSSMManagedInstanceCore",
            "RoleName": "sampleSSMInstanceRole"
          },
          "nextStep": "createSSMInstanceProfile"
        },
        {
          "name": "createSSMInstanceProfile",
          "action":"aws:executeAwsApi",
          "onFailure": "Abort",
          "inputs": {
            "Service": "iam",
            "Api": "CreateInstanceProfile",
            "InstanceProfileName": "sampleSSMInstanceRole"
          },
          "outputs": [
            {
              "Name": "instanceProfileArn",
              "Selector": "$.InstanceProfile.Arn",
              "Type": "String"
            }
          ],
          "nextStep": "addSSMInstanceRoleToProfile"
        },
        {
          "name": "addSSMInstanceRoleToProfile",
          "action": "aws:executeAwsApi",
          "onFailure": "Abort",
          "inputs": {
            "Service": "iam",
            "Api": "AddRoleToInstanceProfile",
            "InstanceProfileName": "sampleSSMInstanceRole",
            "RoleName": "sampleSSMInstanceRole"
          },
```

```
      "nextStep": "createVpc"
    },
    {
      "name": "createVpc",
      "action": "aws:executeAwsApi",
      "onFailure": "Abort",
      "inputs": {
        "Service": "ec2",
        "Api": "CreateVpc",
        "CidrBlock": "10.0.100.0/22"
      },
      "outputs": [
        {
          "Name": "vpcId",
          "Selector": "$.Vpc.VpcId",
          "Type": "String"
        }
      ],
      "nextStep": "getMainRtb"
    },
    {
      "name": "getMainRtb",
      "action": "aws:executeAwsApi",
      "onFailure": "Abort",
      "inputs": {
        "Service": "ec2",
        "Api": "DescribeRouteTables",
        "Filters": [
          {
            "Name": "vpc-id",
            "Values": ["{{ createVpc.vpcId }}"]
          }
        ]
      },
      "outputs": [
        {
          "Name": "mainRtbId",
          "Selector": "$.RouteTables[0].RouteTableId",
          "Type": "String"
        }
      ],
      "nextStep": "verifyMainRtb"
    },
    {
```

```
        "name": "verifyMainRtb",
        "action": "aws:assertAwsResourceProperty",
        "onFailure": "Abort",
        "inputs": {
          "Service": "ec2",
          "Api": "DescribeRouteTables",
          "RouteTableIds": ["{{ getMainRtb.mainRtbId }}"],
          "PropertySelector": "$.RouteTables[0].Associations[0].Main",
          "DesiredValues": ["True"]
        },
        "nextStep": "createPubSubnet"
      },
      {
        "name": "createPubSubnet",
        "action": "aws:executeAwsApi",
        "onFailure": "Abort",
        "inputs": {
          "Service": "ec2",
          "Api": "CreateSubnet",
          "CidrBlock": "10.0.103.0/24",
          "AvailabilityZone": "us-west-2c",
          "VpcId": "{{ createVpc.vpcId }}"
        },
        "outputs":[
          {
            "Name": "pubSubnetId",
            "Selector": "$.Subnet.SubnetId",
            "Type": "String"
          }
        ],
        "nextStep": "createPubRtb"
      },
      {
        "name": "createPubRtb",
        "action": "aws:executeAwsApi",
        "onFailure": "Abort",
        "inputs": {
          "Service": "ec2",
          "Api": "CreateRouteTable",
          "VpcId": "{{ createVpc.vpcId }}"
        },
        "outputs": [
          {
            "Name": "pubRtbId",
```

```
            "Selector": "$.RouteTable.RouteTableId",
            "Type": "String"
          }
        ],
        "nextStep": "createIgw"
      },
      {
        "name": "createIgw",
        "action": "aws:executeAwsApi",
        "onFailure": "Abort",
        "inputs": {
          "Service": "ec2",
          "Api": "CreateInternetGateway"
        },
        "outputs": [
          {
            "Name": "igwId",
            "Selector": "$.InternetGateway.InternetGatewayId",
            "Type": "String"
          }
        ],
        "nextStep": "attachIgw"
      },
      {
        "name": "attachIgw",
        "action": "aws:executeAwsApi",
        "onFailure": "Abort",
        "inputs": {
          "Service": "ec2",
          "Api": "AttachInternetGateway",
          "InternetGatewayId": "{{ createIgw.igwId }}",
          "VpcId": "{{ createVpc.vpcId }}"
        },
        "nextStep": "allocateEip"
      },
      {
        "name": "allocateEip",
        "action": "aws:executeAwsApi",
        "onFailure": "Abort",
        "inputs": {
          "Service": "ec2",
          "Api": "AllocateAddress",
          "Domain": "vpc"
        },
```

```
      "outputs": [
        {
          "Name": "eipAllocationId",
          "Selector": "$.AllocationId",
          "Type": "String"
        }
      ],
      "nextStep": "createNatGw"
    },
    {
      "name": "createNatGw",
      "action": "aws:executeAwsApi",
      "onFailure": "Abort",
      "inputs": {
        "Service": "ec2",
        "Api": "CreateNatGateway",
        "AllocationId": "{{ allocateEip.eipAllocationId }}",
        "SubnetId": "{{ createPubSubnet.pubSubnetId }}"
      },
      "outputs":[
        {
          "Name": "natGwId",
          "Selector": "$.NatGateway.NatGatewayId",
          "Type": "String"
        }
      ],
      "nextStep": "verifyNatGwAvailable"
    },
    {
      "name": "verifyNatGwAvailable",
      "action": "aws:waitForAwsResourceProperty",
      "timeoutSeconds": 150,
      "inputs": {
        "Service": "ec2",
        "Api": "DescribeNatGateways",
        "NatGatewayIds": [
          "{{ createNatGw.natGwId }}"
        ],
        "PropertySelector": "$.NatGateways[0].State",
        "DesiredValues": [
          "available"
        ]
      },
      "nextStep": "createNatRoute"
```

```
        },
        {
          "name": "createNatRoute",
          "action": "aws:executeAwsApi",
          "onFailure": "Abort",
          "inputs": {
            "Service": "ec2",
            "Api": "CreateRoute",
            "DestinationCidrBlock": "0.0.0.0/0",
            "NatGatewayId": "{{ createNatGw.natGwId }}",
            "RouteTableId": "{{ getMainRtb.mainRtbId }}"
          },
          "nextStep": "createPubRoute"
        },
        {
          "name": "createPubRoute",
          "action": "aws:executeAwsApi",
          "onFailure": "Abort",
          "inputs": {
            "Service": "ec2",
            "Api": "CreateRoute",
            "DestinationCidrBlock": "0.0.0.0/0",
            "GatewayId": "{{ createIgw.igwId }}",
            "RouteTableId": "{{ createPubRtb.pubRtbId }}"
          },
          "nextStep": "setPubSubAssoc"
        },
        {
          "name": "setPubSubAssoc",
          "action": "aws:executeAwsApi",
          "onFailure": "Abort",
          "inputs": {
            "Service": "ec2",
            "Api": "AssociateRouteTable",
            "RouteTableId": "{{ createPubRtb.pubRtbId }}",
            "SubnetId": "{{ createPubSubnet.pubSubnetId }}"
          }
        },
        {
          "name": "createDhcpOptions",
          "action": "aws:executeAwsApi",
          "onFailure": "Abort",
          "inputs": {
            "Service": "ec2",
```

```
            "Api": "CreateDhcpOptions",
            "DhcpConfigurations": [
              {
                "Key": "domain-name-servers",
                "Values": ["10.0.100.50,10.0.101.50"]
              },
              {
                "Key": "domain-name",
                "Values": ["sample.com"]
              }
            ]
          },
          "outputs": [
            {
              "Name": "dhcpOptionsId",
              "Selector": "$.DhcpOptions.DhcpOptionsId",
              "Type": "String"
            }
          ],
          "nextStep": "createDCSubnet1"
        },
        {
          "name": "createDCSubnet1",
          "action": "aws:executeAwsApi",
          "onFailure": "Abort",
          "inputs": {
            "Service": "ec2",
            "Api": "CreateSubnet",
            "CidrBlock": "10.0.100.0/24",
            "AvailabilityZone": "us-west-2a",
            "VpcId": "{{ createVpc.vpcId }}"
          },
          "outputs": [
            {
              "Name": "firstSubnetId",
              "Selector": "$.Subnet.SubnetId",
              "Type": "String"
            }
          ],
          "nextStep": "createDCSubnet2"
        },
        {
          "name": "createDCSubnet2",
          "action": "aws:executeAwsApi",
```

```
      "onFailure": "Abort",
      "inputs": {
        "Service": "ec2",
        "Api": "CreateSubnet",
        "CidrBlock": "10.0.101.0/24",
        "AvailabilityZone": "us-west-2b",
        "VpcId": "{{ createVpc.vpcId }}"
      },
      "outputs": [
        {
          "Name": "secondSubnetId",
          "Selector": "$.Subnet.SubnetId",
          "Type": "String"
        }
      ],
      "nextStep": "createDCSecGroup"
    },
    {
      "name": "createDCSecGroup",
      "action": "aws:executeAwsApi",
      "onFailure": "Abort",
      "inputs": {
        "Service": "ec2",
        "Api": "CreateSecurityGroup",
        "GroupName": "SampleDCSecGroup",
        "Description": "Security Group for Example Domain Controllers",
        "VpcId": "{{ createVpc.vpcId }}"
      },
      "outputs": [
        {
          "Name": "dcSecGroupId",
          "Selector": "$.GroupId",
          "Type": "String"
        }
      ],
      "nextStep": "authIngressDCTraffic"
    },
    {
      "name": "authIngressDCTraffic",
      "action": "aws:executeAwsApi",
      "onFailure": "Abort",
      "inputs": {
        "Service": "ec2",
        "Api": "AuthorizeSecurityGroupIngress",
```

```
            "GroupId": "{{ createDCSecGroup.dcSecGroupId }}",
            "IpPermissions": [
              {
                "FromPort": -1,
                "IpProtocol": "-1",
                "IpRanges": [
                  {
                    "CidrIp": "0.0.0.0/0",
                    "Description": "Allow all traffic between Domain
Controllers"
                  }
                ]
              }
            ]
          },
          "nextStep": "verifyInstanceProfile"
        },
        {
          "name": "verifyInstanceProfile",
          "action": "aws:waitForAwsResourceProperty",
          "maxAttempts": 5,
          "onFailure": "Abort",
          "inputs": {
            "Service": "iam",
            "Api": "ListInstanceProfilesForRole",
            "RoleName": "sampleSSMInstanceRole",
            "PropertySelector": "$.InstanceProfiles[0].Arn",
            "DesiredValues": [
              "{{ createSSMInstanceProfile.instanceProfileArn }}"
            ]
          },
          "nextStep": "iamEventualConsistency"
        },
        {
          "name": "iamEventualConsistency",
          "action": "aws:sleep",
          "inputs": {
            "Duration": "PT2M"
          },
          "nextStep": "launchDC1"
        },
        {
          "name": "launchDC1",
          "action": "aws:executeAwsApi",
```

```
        "onFailure": "Abort",
        "inputs": {
          "Service": "ec2",
          "Api": "RunInstances",
          "BlockDeviceMappings": [
            {
              "DeviceName": "/dev/sda1",
              "Ebs": {
                "DeleteOnTermination": true,
                "VolumeSize": 50,
                "VolumeType": "gp2"
              }
            },
            {
              "DeviceName": "xvdf",
              "Ebs": {
                "DeleteOnTermination": true,
                "VolumeSize": 100,
                "VolumeType": "gp2"
              }
            }
          ],
          "IamInstanceProfile": {
            "Arn": "{{ createSSMInstanceProfile.instanceProfileArn }}"
          },
          "ImageId": "{{ getLatestWindowsAmi.amiId }}",
          "InstanceType": "t2.micro",
          "MaxCount": 1,
          "MinCount": 1,
          "PrivateIpAddress": "10.0.100.50",
          "SecurityGroupIds": [
            "{{ createDCSecGroup.dcSecGroupId }}"
          ],
          "SubnetId": "{{ createDCSubnet1.firstSubnetId }}",
          "TagSpecifications": [
            {
              "ResourceType": "instance",
              "Tags": [
                {
                  "Key": "Name",
                  "Value": "SampleDC1"
                }
              ]
            }
```

```
            ]
          },
          "outputs": [
            {
              "Name": "pdcInstanceId",
              "Selector": "$.Instances[0].InstanceId",
              "Type": "String"
            }
          ],
          "nextStep": "launchDC2"
        },
        {
          "name": "launchDC2",
          "action": "aws:executeAwsApi",
          "onFailure": "Abort",
          "inputs": {
            "Service": "ec2",
            "Api": "RunInstances",
            "BlockDeviceMappings": [
              {
                "DeviceName": "/dev/sda1",
                "Ebs": {
                  "DeleteOnTermination": true,
                  "VolumeSize": 50,
                  "VolumeType": "gp2"
                }
              },
              {
                "DeviceName": "xvdf",
                "Ebs": {
                  "DeleteOnTermination": true,
                  "VolumeSize": 100,
                  "VolumeType": "gp2"
                }
              }
            ],
            "IamInstanceProfile": {
              "Arn": "{{ createSSMInstanceProfile.instanceProfileArn }}"
            },
            "ImageId": "{{ getLatestWindowsAmi.amiId }}",
            "InstanceType": "t2.micro",
            "MaxCount": 1,
            "MinCount": 1,
            "PrivateIpAddress": "10.0.101.50",
```

```
            "SecurityGroupIds": [
              "{{ createDCSecGroup.dcSecGroupId }}"
            ],
            "SubnetId": "{{ createDCSubnet2.secondSubnetId }}",
            "TagSpecifications": [
              {
                "ResourceType": "instance",
                "Tags": [
                  {
                    "Key": "Name",
                    "Value": "SampleDC2"
                  }
                ]
              }
            ]
          },
          "outputs": [
            {
              "Name": "adcInstanceId",
              "Selector": "$.Instances[0].InstanceId",
              "Type": "String"
            }
          ],
          "nextStep": "verifyDCInstanceState"
        },
        {
          "name": "verifyDCInstanceState",
          "action": "aws:waitForAwsResourceProperty",
          "inputs": {
            "Service": "ec2",
            "Api": "DescribeInstanceStatus",
            "IncludeAllInstances": true,
            "InstanceIds": [
              "{{ launchDC1.pdcInstanceId }}",
              "{{ launchDC2.adcInstanceId }}"
            ],
            "PropertySelector": "$.InstanceStatuses[0].InstanceState.Name",
            "DesiredValues": [
              "running"
            ]
          },
          "nextStep": "verifyInstancesOnlineSSM"
        },
        {
```

```
          "name": "verifyInstancesOnlineSSM",
          "action": "aws:waitForAwsResourceProperty",
          "timeoutSeconds": 600,
          "inputs": {
            "Service": "ssm",
            "Api": "DescribeInstanceInformation",
            "InstanceInformationFilterList": [
              {
                "key": "InstanceIds",
                "valueSet": [
                  "{{ launchDC1.pdcInstanceId }}",
                  "{{ launchDC2.adcInstanceId }}"
                ]
              }
            ],
            "PropertySelector": "$.InstanceInformationList[0].PingStatus",
            "DesiredValues": [
              "Online"
            ]
          },
          "nextStep": "installADRoles"
        },
        {
          "name": "installADRoles",
          "action": "aws:runCommand",
          "inputs": {
            "DocumentName": "AWS-RunPowerShellScript",
            "InstanceIds": [
              "{{ launchDC1.pdcInstanceId }}",
              "{{ launchDC2.adcInstanceId }}"
            ],
            "Parameters": {
              "commands": [
                "try {",
                "  Install-WindowsFeature -Name AD-Domain-Services -
IncludeManagementTools",
                "}",
                "catch {",
                "  Write-Error \"Failed to install ADDS Role.\"",
                "}"
              ]
            }
          },
          "nextStep": "setAdminPassword"
```

```
        },
        {
          "name": "setAdminPassword",
          "action": "aws:runCommand",
          "inputs": {
            "DocumentName": "AWS-RunPowerShellScript",
            "InstanceIds": [
              "{{ launchDC1.pdcInstanceId }}"
            ],
            "Parameters": {
              "commands": [
                "net user Administrator \"sampleAdminPass123!\""
              ]
            }
          },
          "nextStep": "createForest"
        },
        {
          "name": "createForest",
          "action": "aws:runCommand",
          "inputs": {
            "DocumentName": "AWS-RunPowerShellScript",
            "InstanceIds": [
              "{{ launchDC1.pdcInstanceId }}"
            ],
            "Parameters": {
              "commands": [
                "$dsrmPass = 'sample123!' | ConvertTo-SecureString -
asPlainText -Force",
                "try {",
                "   Install-ADDSForest -DomainName \"sample.com\" -DomainMode
 6 -ForestMode 6 -InstallDNS -DatabasePath \"D:\\NTDS\" -SysvolPath \"D:\
\SYSVOL\" -SafeModeAdministratorPassword $dsrmPass -Force",
                "}",
                "catch {",
                "   Write-Error $_",
                "}",
                "try {",
                "   Add-DnsServerForwarder -IPAddress \"10.0.100.2\"",
                "}",
                "catch {",
                "   Write-Error $_",
                "}"
              ]
```

```
            }
          },
          "nextStep": "associateDhcpOptions"
        },
        {
          "name": "associateDhcpOptions",
          "action": "aws:executeAwsApi",
          "onFailure": "Abort",
          "inputs": {
            "Service": "ec2",
            "Api": "AssociateDhcpOptions",
            "DhcpOptionsId": "{{ createDhcpOptions.dhcpOptionsId }}",
            "VpcId": "{{ createVpc.vpcId }}"
          },
          "nextStep": "waitForADServices"
        },
        {
          "name": "waitForADServices",
          "action": "aws:sleep",
          "inputs": {
            "Duration": "PT1M"
          },
          "nextStep": "promoteADC"
        },
        {
          "name": "promoteADC",
          "action": "aws:runCommand",
          "inputs": {
            "DocumentName": "AWS-RunPowerShellScript",
            "InstanceIds": [
              "{{ launchDC2.adcInstanceId }}"
            ],
            "Parameters": {
              "commands": [
                "ipconfig /renew",
                "$dsrmPass = 'sample123!' | ConvertTo-SecureString -
asPlainText -Force",
                "$domAdminUser = \"sample\\Administrator\"",
                "$domAdminPass = \"sampleAdminPass123!\" | ConvertTo-
SecureString -asPlainText -Force",
                "$domAdminCred = New-Object
 System.Management.Automation.PSCredential($domAdminUser,$domAdminPass)",
                "try {",
```

```
                "   Install-ADDSDomainController -DomainName \"sample.com
\" -InstallDNS -DatabasePath \"D:\\NTDS\" -SysvolPath \"D:\\SYSVOL\" -
SafeModeAdministratorPassword $dsrmPass -Credential $domAdminCred -Force",
                "}",
                "catch {",
                "   Write-Error $_",
                "}"
              ]
            }
          }
        }
      ]
    }
```

从最新快照还原根卷

根卷上的操作系统可能会出于各种原因而受损。例如，在执行修补操作后，实例可能会因内核或注册表损坏而无法成功启动。自动执行常见故障排除任务（例如，从修补操作前拍摄的最新快照还原根卷）可以减少停机时间并加快故障排除工作。AWS Systems Manager自动化操作可帮助您做到这一点。Automation 是 AWS Systems Manager 中的一项工具。

以下示例 AWS Systems Manager 运行手册执行这些操作：

- 使用 `aws:executeAwsApi` 自动化操作从实例的根卷检索详细信息。
- 使用 `aws:executeScript` 自动化操作检索根卷的最新快照。
- 如果找到了根卷的快照，则使用 `aws:branch` 自动化操作继续执行自动化。

YAML

```
---
description: Custom Automation Troubleshooting Example
schemaVersion: '0.3'
assumeRole: "{{ AutomationAssumeRole }}"
parameters:
  AutomationAssumeRole:
    type: String
    description: "(Required) The ARN of the role that allows Automation to
perform
```

```
          the actions on your behalf. If no role is specified, Systems Manager
Automation
          uses your IAM permissions to use this runbook."
        default: ''
      InstanceId:
          type: String
          description: "(Required) The Instance Id whose root EBS volume you want to
restore the latest Snapshot."
          default: ''
    mainSteps:
    - name: getInstanceDetails
      action: aws:executeAwsApi
      onFailure: Abort
      inputs:
        Service: ec2
        Api: DescribeInstances
        InstanceIds:
        - "{{ InstanceId }}"
      outputs:
        - Name: availabilityZone
          Selector: "$.Reservations[0].Instances[0].Placement.AvailabilityZone"
          Type: String
        - Name: rootDeviceName
          Selector: "$.Reservations[0].Instances[0].RootDeviceName"
          Type: String
      nextStep: getRootVolumeId
    - name: getRootVolumeId
      action: aws:executeAwsApi
      onFailure: Abort
      inputs:
        Service: ec2
        Api: DescribeVolumes
        Filters:
        -  Name: attachment.device
           Values: ["{{ getInstanceDetails.rootDeviceName }}"]
        -  Name: attachment.instance-id
           Values: ["{{ InstanceId }}"]
      outputs:
        - Name: rootVolumeId
          Selector: "$.Volumes[0].VolumeId"
          Type: String
      nextStep: getSnapshotsByStartTime
    - name: getSnapshotsByStartTime
      action: aws:executeScript
```

```
      timeoutSeconds: 45
      onFailure: Abort
      inputs:
        Runtime: python3.8
        Handler: getSnapshotsByStartTime
        InputPayload:
          rootVolumeId : "{{ getRootVolumeId.rootVolumeId }}"
        Script: |-
          def getSnapshotsByStartTime(events,context):
            import boto3

            #Initialize client
            ec2 = boto3.client('ec2')
            rootVolumeId = events['rootVolumeId']
            snapshotsQuery = ec2.describe_snapshots(
              Filters=[
                {
                  "Name": "volume-id",
                  "Values": [rootVolumeId]
                }
              ]
            )
            if not snapshotsQuery['Snapshots']:
              noSnapshotFoundString = "NoSnapshotFound"
              return { 'noSnapshotFound' : noSnapshotFoundString }
            else:
              jsonSnapshots = snapshotsQuery['Snapshots']
              sortedSnapshots = sorted(jsonSnapshots, key=lambda k: k['StartTime'],
reverse=True)
              latestSortedSnapshotId = sortedSnapshots[0]['SnapshotId']
              return { 'latestSnapshotId' : latestSortedSnapshotId }
      outputs:
      - Name: Payload
        Selector: $.Payload
        Type: StringMap
      - Name: latestSnapshotId
        Selector: $.Payload.latestSnapshotId
        Type: String
      - Name: noSnapshotFound
        Selector: $.Payload.noSnapshotFound
        Type: String
      nextStep: branchFromResults
    - name: branchFromResults
      action: aws:branch
```

```
    onFailure: Abort
    inputs:
      Choices:
      - NextStep: createNewRootVolumeFromSnapshot
        Not:
          Variable: "{{ getSnapshotsByStartTime.noSnapshotFound }}"
          StringEquals: "NoSnapshotFound"
    isEnd: true
  - name: createNewRootVolumeFromSnapshot
    action: aws:executeAwsApi
    onFailure: Abort
    inputs:
      Service: ec2
      Api: CreateVolume
      AvailabilityZone: "{{ getInstanceDetails.availabilityZone }}"
      SnapshotId: "{{ getSnapshotsByStartTime.latestSnapshotId }}"
    outputs:
      - Name: newRootVolumeId
        Selector: "$.VolumeId"
        Type: String
    nextStep: stopInstance
  - name: stopInstance
    action: aws:executeAwsApi
    onFailure: Abort
    inputs:
      Service: ec2
      Api: StopInstances
      InstanceIds:
      - "{{ InstanceId }}"
    nextStep: verifyVolumeAvailability
  - name: verifyVolumeAvailability
    action: aws:waitForAwsResourceProperty
    timeoutSeconds: 120
    inputs:
      Service: ec2
      Api: DescribeVolumes
      VolumeIds:
      - "{{ createNewRootVolumeFromSnapshot.newRootVolumeId }}"
      PropertySelector: "$.Volumes[0].State"
      DesiredValues:
      - "available"
    nextStep: verifyInstanceStopped
  - name: verifyInstanceStopped
    action: aws:waitForAwsResourceProperty
```

```
      timeoutSeconds: 120
      inputs:
        Service: ec2
        Api: DescribeInstances
        InstanceIds:
        - "{{ InstanceId }}"
        PropertySelector: "$.Reservations[0].Instances[0].State.Name"
        DesiredValues:
        - "stopped"
      nextStep: detachRootVolume
    - name: detachRootVolume
      action: aws:executeAwsApi
      onFailure: Abort
      inputs:
        Service: ec2
        Api: DetachVolume
        VolumeId: "{{ getRootVolumeId.rootVolumeId }}"
      nextStep: verifyRootVolumeDetached
    - name: verifyRootVolumeDetached
      action: aws:waitForAwsResourceProperty
      timeoutSeconds: 30
      inputs:
        Service: ec2
        Api: DescribeVolumes
        VolumeIds:
        - "{{ getRootVolumeId.rootVolumeId }}"
        PropertySelector: "$.Volumes[0].State"
        DesiredValues:
        - "available"
      nextStep: attachNewRootVolume
    - name: attachNewRootVolume
      action: aws:executeAwsApi
      onFailure: Abort
      inputs:
        Service: ec2
        Api: AttachVolume
        Device: "{{ getInstanceDetails.rootDeviceName }}"
        InstanceId: "{{ InstanceId }}"
        VolumeId: "{{ createNewRootVolumeFromSnapshot.newRootVolumeId }}"
      nextStep: verifyNewRootVolumeAttached
    - name: verifyNewRootVolumeAttached
      action: aws:waitForAwsResourceProperty
      timeoutSeconds: 30
      inputs:
```

```
        Service: ec2
        Api: DescribeVolumes
        VolumeIds:
        - "{{ createNewRootVolumeFromSnapshot.newRootVolumeId }}"
        PropertySelector: "$.Volumes[0].Attachments[0].State"
        DesiredValues:
        - "attached"
      nextStep: startInstance
    - name: startInstance
      action: aws:executeAwsApi
      onFailure: Abort
      inputs:
        Service: ec2
        Api: StartInstances
        InstanceIds:
        - "{{ InstanceId }}"
```

JSON

```
   {
      "description": "Custom Automation Troubleshooting Example",
      "schemaVersion": "0.3",
      "assumeRole": "{{ AutomationAssumeRole }}",
      "parameters": {
         "AutomationAssumeRole": {
            "type": "String",
            "description": "(Required) The ARN of the role that allows Automation
to perform the actions on your behalf. If no role is specified, Systems Manager
Automation uses your IAM permissions to run this runbook.",
            "default": ""
         },
         "InstanceId": {
            "type": "String",
            "description": "(Required) The Instance Id whose root EBS volume you
want to restore the latest Snapshot.",
            "default": ""
         }
      },
      "mainSteps": [
         {
            "name": "getInstanceDetails",
            "action": "aws:executeAwsApi",
```

```
            "onFailure": "Abort",
            "inputs": {
               "Service": "ec2",
               "Api": "DescribeInstances",
               "InstanceIds": [
                  "{{ InstanceId }}"
               ]
            },
            "outputs": [
               {
                  "Name": "availabilityZone",
                  "Selector":
"$.Reservations[0].Instances[0].Placement.AvailabilityZone",
                  "Type": "String"
               },
               {
                  "Name": "rootDeviceName",
                  "Selector": "$.Reservations[0].Instances[0].RootDeviceName",
                  "Type": "String"
               }
            ],
            "nextStep": "getRootVolumeId"
         },
         {
            "name": "getRootVolumeId",
            "action": "aws:executeAwsApi",
            "onFailure": "Abort",
            "inputs": {
               "Service": "ec2",
               "Api": "DescribeVolumes",
               "Filters": [
                  {
                     "Name": "attachment.device",
                     "Values": [
                        "{{ getInstanceDetails.rootDeviceName }}"
                     ]
                  },
                  {
                     "Name": "attachment.instance-id",
                     "Values": [
                        "{{ InstanceId }}"
                     ]
                  }
               ]
```

```
            },
            "outputs": [
               {
                  "Name": "rootVolumeId",
                  "Selector": "$.Volumes[0].VolumeId",
                  "Type": "String"
               }
            ],
            "nextStep": "getSnapshotsByStartTime"
         },
         {
            "name": "getSnapshotsByStartTime",
            "action": "aws:executeScript",
            "timeoutSeconds": 45,
            "onFailure": "Continue",
            "inputs": {
               "Runtime": "python3.8",
               "Handler": "getSnapshotsByStartTime",
               "InputPayload": {
                  "rootVolumeId": "{{ getRootVolumeId.rootVolumeId }}"
               },
               "Attachment": "getSnapshotsByStartTime.py"
            },
            "outputs": [
               {
                  "Name": "Payload",
                  "Selector": "$.Payload",
                  "Type": "StringMap"
               },
               {
                  "Name": "latestSnapshotId",
                  "Selector": "$.Payload.latestSnapshotId",
                  "Type": "String"
               },
               {
                  "Name": "noSnapshotFound",
                  "Selector": "$.Payload.noSnapshotFound",
                  "Type": "String"
               }
            ],
            "nextStep": "branchFromResults"
         },
         {
            "name": "branchFromResults",
```

```
            "action": "aws:branch",
            "onFailure": "Abort",
            "inputs": {
               "Choices": [
                  {
                     "NextStep": "createNewRootVolumeFromSnapshot",
                     "Not": {
                        "Variable":
"{{ getSnapshotsByStartTime.noSnapshotFound }}",
                        "StringEquals": "NoSnapshotFound"
                     }
                  }
               ]
            },
            "isEnd": true
         },
         {
            "name": "createNewRootVolumeFromSnapshot",
            "action": "aws:executeAwsApi",
            "onFailure": "Abort",
            "inputs": {
               "Service": "ec2",
               "Api": "CreateVolume",
               "AvailabilityZone": "{{ getInstanceDetails.availabilityZone }}",
               "SnapshotId": "{{ getSnapshotsByStartTime.latestSnapshotId }}"
            },
            "outputs": [
               {
                  "Name": "newRootVolumeId",
                  "Selector": "$.VolumeId",
                  "Type": "String"
               }
            ],
            "nextStep": "stopInstance"
         },
         {
            "name": "stopInstance",
            "action": "aws:executeAwsApi",
            "onFailure": "Abort",
            "inputs": {
               "Service": "ec2",
               "Api": "StopInstances",
               "InstanceIds": [
                  "{{ InstanceId }}"
```

```
            ]
         },
         "nextStep": "verifyVolumeAvailability"
      },
      {
         "name": "verifyVolumeAvailability",
         "action": "aws:waitForAwsResourceProperty",
         "timeoutSeconds": 120,
         "inputs": {
            "Service": "ec2",
            "Api": "DescribeVolumes",
            "VolumeIds": [
               "{{ createNewRootVolumeFromSnapshot.newRootVolumeId }}"
            ],
            "PropertySelector": "$.Volumes[0].State",
            "DesiredValues": [
               "available"
            ]
         },
         "nextStep": "verifyInstanceStopped"
      },
      {
         "name": "verifyInstanceStopped",
         "action": "aws:waitForAwsResourceProperty",
         "timeoutSeconds": 120,
         "inputs": {
            "Service": "ec2",
            "Api": "DescribeInstances",
            "InstanceIds": [
               "{{ InstanceId }}"
            ],
            "PropertySelector": "$.Reservations[0].Instances[0].State.Name",
            "DesiredValues": [
               "stopped"
            ]
         },
         "nextStep": "detachRootVolume"
      },
      {
         "name": "detachRootVolume",
         "action": "aws:executeAwsApi",
         "onFailure": "Abort",
         "inputs": {
            "Service": "ec2",
```

```
            "Api": "DetachVolume",
            "VolumeId": "{{ getRootVolumeId.rootVolumeId }}"
         },
         "nextStep": "verifyRootVolumeDetached"
      },
      {
         "name": "verifyRootVolumeDetached",
         "action": "aws:waitForAwsResourceProperty",
         "timeoutSeconds": 30,
         "inputs": {
            "Service": "ec2",
            "Api": "DescribeVolumes",
            "VolumeIds": [
               "{{ getRootVolumeId.rootVolumeId }}"
            ],
            "PropertySelector": "$.Volumes[0].State",
            "DesiredValues": [
               "available"
            ]
         },
         "nextStep": "attachNewRootVolume"
      },
      {
         "name": "attachNewRootVolume",
         "action": "aws:executeAwsApi",
         "onFailure": "Abort",
         "inputs": {
            "Service": "ec2",
            "Api": "AttachVolume",
            "Device": "{{ getInstanceDetails.rootDeviceName }}",
            "InstanceId": "{{ InstanceId }}",
            "VolumeId": "{{ createNewRootVolumeFromSnapshot.newRootVolumeId }}"
         },
         "nextStep": "verifyNewRootVolumeAttached"
      },
      {
         "name": "verifyNewRootVolumeAttached",
         "action": "aws:waitForAwsResourceProperty",
         "timeoutSeconds": 30,
         "inputs": {
            "Service": "ec2",
            "Api": "DescribeVolumes",
            "VolumeIds": [
               "{{ createNewRootVolumeFromSnapshot.newRootVolumeId }}"
```

```
            ],
            "PropertySelector": "$.Volumes[0].Attachments[0].State",
            "DesiredValues": [
               "attached"
            ]
         },
         "nextStep": "startInstance"
      },
      {
         "name": "startInstance",
         "action": "aws:executeAwsApi",
         "onFailure": "Abort",
         "inputs": {
            "Service": "ec2",
            "Api": "StartInstances",
            "InstanceIds": [
               "{{ InstanceId }}"
            ]
         }
      }
   ],
   "files": {
        "getSnapshotsByStartTime.py": {
            "checksums": {
                "sha256": "sampleETagValue"
            }
        }
    }
}
```

创建 AMI 和跨区域副本

创建实例的 Amazon Machine Image (AMI) 是备份和恢复操作中的常用过程。也可以选择将 AMI 作为灾难恢复架构的一部分复制到另一个 AWS 区域。如果问题需要进行故障转移，则自动执行常见的维护任务可以减少停机时间。AWS Systems Manager自动化操作可帮助您做到这一点。Automation 是 AWS Systems Manager 中的一项工具。

以下示例 AWS Systems Manager 运行手册执行这些操作：

- 使用 `aws:executeAwsApi` 自动化操作创建 AMI。
- 使用 `aws:waitForAwsResourceProperty` 自动化操作确认 AMI 的可用性。

- 使用 aws:executeScript 自动化操作将 AMI 复制到目标区域。

YAML

```
   ---
   description: Custom Automation Backup and Recovery Example
   schemaVersion: '0.3'
   assumeRole: "{{ AutomationAssumeRole }}"
   parameters:
     AutomationAssumeRole:
       type: String
       description: "(Required) The ARN of the role that allows Automation to
perform
         the actions on your behalf. If no role is specified, Systems Manager
Automation
         uses your IAM permissions to use this runbook."
       default: ''
     InstanceId:
       type: String
       description: "(Required) The ID of the EC2 instance."
       default: ''
   mainSteps:
   - name: createImage
     action: aws:executeAwsApi
     onFailure: Abort
     inputs:
       Service: ec2
       Api: CreateImage
       InstanceId: "{{ InstanceId }}"
       Name: "Automation Image for {{ InstanceId }}"
       NoReboot: false
     outputs:
       - Name: newImageId
         Selector: "$.ImageId"
         Type: String
     nextStep: verifyImageAvailability
   - name: verifyImageAvailability
     action: aws:waitForAwsResourceProperty
     timeoutSeconds: 600
     inputs:
       Service: ec2
       Api: DescribeImages
```

```
        ImageIds:
        - "{{ createImage.newImageId }}"
        PropertySelector: "$.Images[0].State"
        DesiredValues:
        - available
      nextStep: copyImage
    - name: copyImage
      action: aws:executeScript
      timeoutSeconds: 45
      onFailure: Abort
      inputs:
        Runtime: python3.8
        Handler: crossRegionImageCopy
        InputPayload:
          newImageId : "{{ createImage.newImageId }}"
        Script: |-
          def crossRegionImageCopy(events,context):
            import boto3

            #Initialize client
            ec2 = boto3.client('ec2', region_name='us-east-1')
            newImageId = events['newImageId']

            ec2.copy_image(
              Name='DR Copy for ' + newImageId,
              SourceImageId=newImageId,
              SourceRegion='us-west-2'
            )
```

JSON

```
   {
      "description": "Custom Automation Backup and Recovery Example",
      "schemaVersion": "0.3",
      "assumeRole": "{{ AutomationAssumeRole }}",
      "parameters": {
         "AutomationAssumeRole": {
            "type": "String",
            "description": "(Required) The ARN of the role that allows Automation
to perform\nthe actions on your behalf. If no role is specified, Systems Manager
Automation\nuses your IAM permissions to run this runbook.",
            "default": ""
```

```
        },
        "InstanceId": {
           "type": "String",
           "description": "(Required) The ID of the EC2 instance.",
           "default": ""
        }
     },
     "mainSteps": [
        {
           "name": "createImage",
           "action": "aws:executeAwsApi",
           "onFailure": "Abort",
           "inputs": {
              "Service": "ec2",
              "Api": "CreateImage",
              "InstanceId": "{{ InstanceId }}",
              "Name": "Automation Image for {{ InstanceId }}",
              "NoReboot": false
           },
           "outputs": [
              {
                 "Name": "newImageId",
                 "Selector": "$.ImageId",
                 "Type": "String"
              }
           ],
           "nextStep": "verifyImageAvailability"
        },
        {
           "name": "verifyImageAvailability",
           "action": "aws:waitForAwsResourceProperty",
           "timeoutSeconds": 600,
           "inputs": {
              "Service": "ec2",
              "Api": "DescribeImages",
              "ImageIds": [
                 "{{ createImage.newImageId }}"
              ],
              "PropertySelector": "$.Images[0].State",
              "DesiredValues": [
                 "available"
              ]
           },
           "nextStep": "copyImage"
```

```
        },
        {
           "name": "copyImage",
           "action": "aws:executeScript",
           "timeoutSeconds": 45,
           "onFailure": "Abort",
           "inputs": {
              "Runtime": "python3.8",
              "Handler": "crossRegionImageCopy",
              "InputPayload": {
                 "newImageId": "{{ createImage.newImageId }}"
              },
              "Attachment": "crossRegionImageCopy.py"
           }
        }
     ],
     "files": {
          "crossRegionImageCopy.py": {
              "checksums": {
                  "sha256": "sampleETagValue"
              }
          }
      }
  }
```

创建填充 AWS 资源的输入参数

Automation 是 Systems Manager 中的一项工具，可在 AWS Management Console中填充与您为输入参数定义的资源类型相匹配的 AWS 资源。与资源类型匹配的 AWS 账户 中的资源将显示在下拉列表中供您选择。您可以为 Amazon Elastic Compute Cloud (Amazon EC2) 实例、Amazon Simple Storage Service (Amazon S3) 存储桶以及 AWS Identity and Access Management (IAM) 角色定义输入参数类型。支持的类型定义与用于查找匹配资源的正则表达式如下：

- `AWS::EC2::Instance::Id` - `^m?i-([a-z0-9]{8}|[a-z0-9]{17})$`
- `List<AWS::EC2::Instance::Id>` - `^m?i-([a-z0-9]{8}|[a-z0-9]{17})$`
- `AWS::S3::Bucket::Name` - `^[0-9a-z][a-z0-9\\-\\.]{3,63}$`
- `List<AWS::S3::Bucket::Name>` - `^[0-9a-z][a-z0-9\\-\\.]{3,63}$`
- `AWS::IAM::Role::Arn` - `^arn:(aws|aws-cn|aws-us-gov|aws-iso|aws-iso-b):iam::[0-9]{12}:role/.*$`

- List<AWS::IAM::Role::Arn> - ^arn:(aws|aws-cn|aws-us-gov|aws-iso|aws-iso-b):iam::[0-9]{12}:role/.*$

以下是在运行手册内容中定义的输入参数类型的示例。

YAML

```
description: Enables encryption on an Amazon S3 bucket
schemaVersion: '0.3'
assumeRole: '{{ AutomationAssumeRole }}'
parameters:
  BucketName:
    type: 'AWS::S3::Bucket::Name'
    description: (Required) The name of the Amazon S3 bucket you want to encrypt.
  SSEAlgorithm:
    type: String
    description: (Optional) The server-side encryption algorithm to use for the
 default encryption.
    default: AES256
  AutomationAssumeRole:
    type: 'AWS::IAM::Role::Arn'
    description: (Optional) The Amazon Resource Name (ARN) of the role that allows
 Automation to perform the actions on your behalf.
    default: ''
mainSteps:
  - name: enableBucketEncryption
    action: 'aws:executeAwsApi'
    inputs:
      Service: s3
      Api: PutBucketEncryption
      Bucket: '{{BucketName}}'
      ServerSideEncryptionConfiguration:
        Rules:
          - ApplyServerSideEncryptionByDefault:
              SSEAlgorithm: '{{SSEAlgorithm}}'
    isEnd: true
```

JSON

```
{
   "description": "Enables encryption on an Amazon S3 bucket",
   "schemaVersion": "0.3",
```

```
  "assumeRole": "{{ AutomationAssumeRole }}",
  "parameters": {
     "BucketName": {
        "type": "AWS::S3::Bucket::Name",
        "description": "(Required) The name of the Amazon S3 bucket you want to
encrypt."
     },
     "SSEAlgorithm": {
        "type": "String",
        "description": "(Optional) The server-side encryption algorithm to use for
the default encryption.",
        "default": "AES256"
     },
     "AutomationAssumeRole": {
        "type": "AWS::IAM::Role::Arn",
        "description": "(Optional) The Amazon Resource Name (ARN) of the role that
allows Automation to perform the actions on your behalf.",
        "default": ""
     }
  },
  "mainSteps": [
     {
        "name": "enableBucketEncryption",
        "action": "aws:executeAwsApi",
        "inputs": {
           "Service": "s3",
           "Api": "PutBucketEncryption",
           "Bucket": "{{BucketName}}",
           "ServerSideEncryptionConfiguration": {
              "Rules": [
                 {
                    "ApplyServerSideEncryptionByDefault": {
                       "SSEAlgorithm": "{{SSEAlgorithm}}"
                    }
                 }
              ]
           }
        },
        "isEnd": true
     }
  ]
}
```

正在使用文档生成器创建运行手册

如果 AWS Systems Manager 公有运行手册不支持您希望在 AWS 资源上执行的操作，您可以创建自己的运行手册。要创建自定义运行手册，您可以手动创建包含相应自动化操作的本地 YAML 或 JSON 格式文件。或者，您也可以使用 Systems Manager Automation 控制台中的文档生成器来构建自定义运行手册。

通过使用文档生成器，您可以将自动化操作添加到自定义运行手册中，并提供所需的参数，而无需使用 JSON 或 YAML 语法。在添加步骤并创建运行手册后，系统将您添加的操作转换为 Systems Manager 可用于运行自动化的 YAML 格式。

自动化文档支持使用 Markdown（一种标记语言），它允许您为运行手册和其中的各个步骤添加 Wiki 样式的描述。有关使用 Markdown 的更多信息，请参阅在 AWS 中使用 Markdown。

使用文档生成器创建运行手册

开始前的准备工作

我们建议您阅读可在运行手册中使用的各种操作。有关更多信息，请参阅 Systems Manager 自动化操作参考。

使用文档生成器创建运行手册

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择文档。
3. 选择创建自动化。
4. 对于名称，为运行手册输入一个描述性名称。
5. 对于文档描述，请提供运行手册的 markdown 样式描述。您可以提供使用运行手册的说明、编号的步骤或描述运行手册的任何其他类型的信息。请参阅默认文本以了解设置内容格式的信息。

   > **Tip**
   >
   > 在隐藏预览和显示预览之间切换，以便在编写时查看描述内容的效果。

6. （可选）对于 Assume role (担任角色)，请输入代表您执行操作的服务角色的名称或 ARN。如果未指定角色，自动化使用运行自动化的用户的访问权限。

> **Important**
>
> 对于不归 Amazon 所有并使用 `aws:executeScript` 操作的运行手册，必须指定一个角色。有关信息，请参阅使用运行手册的权限。

7. （可选）对于输出，请输入执行该运行手册的自动化的任何输出以供其他进程使用。

   例如，如果文档创建新的 AMI，您可以指定 ["CreateImage.ImageId"]，然后使用该输出以在后续自动化中创建新的实例。

8. （可选）展开输入参数部分，然后执行以下操作。

   1. 对于参数名称，请输入要创建的运行手册参数的描述性名称。
   2. 对于类型，请选择参数的类型，例如 `String` 或 `MapList`。
   3. 对于必需，请执行以下操作之一：
      - 如果必须在运行时提供该运行手册参数的值，请选择是。
      - 如果该参数不是必需的，请选择否，然后（可选）在默认值中输入默认参数值。
   4. 对于描述，请输入运行手册参数的描述。

   > **Note**
   >
   > 要添加更多运行手册参数，请选择添加参数。要删除运行手册参数，请选择 X（删除）按钮。

9. （可选）展开目标类型部分，然后选择一种目标类型以定义可以运行自动化的资源的类型。例如，要在 EC2 实例上使用运行手册，请选择 `/AWS::EC2::Instance`。

   > **Note**
   >
   > 如果指定“/”值，则运行文档可以在所有类型的资源上运行。有关有效资源类型列表，请参阅 AWS CloudFormation 用户指南 中的 AWS 资源类型参考。

10. （可选）展开文档标签部分，然后输入一个或多个标签键值对以应用于运行手册。标签可以轻松标识、划分和搜索资源。
11. 在步骤 1 部分中，提供以下信息。

    - 对于步骤名称，请输入自动化的第一步的描述性名称。

- 对于操作类型，请选择用于该步骤的操作类型。

  有关可用操作类型的列表和信息，请参阅 Systems Manager 自动化操作参考。

- 对于描述，请输入自动化步骤的描述。您可以使用 Markdown 设置文本格式。
- 根据选定的操作类型，在步骤输入部分中输入操作类型所需的输入。例如，如果选择了 `aws:approve` 操作，您必须为 `Approvers` 属性指定一个值。

  有关步骤输入字段的信息，请参阅 Systems Manager 自动化操作参考中的选定操作类型的条目。例如：`aws:executeStateMachine` - 运行 AWS Step Functions 状态机。。

- （可选）对于其他输入，请提供运行手册所需的任何其他输入值。可用的输入类型取决于您为步骤选择的操作类型。（请注意，某些操作类型需要使用输入值。）

  > **Note**
  >
  > 要添加更多输入，请选择 Add optional input (添加可选的输入)。要删除输入，请选择 X（删除）按钮。

- （可选）对于输出，请输入执行该步骤的任何输出以供其他进程使用。

  > **Note**
  >
  > 输出并非适用于所有操作类型。

- （可选）展开通用属性部分，然后指定所有自动化操作的通用操作属性。例如，对于超时秒数，您可以提供一个值以指定步骤在停止之前可以运行多长时间（以秒为单位）。

  有关更多信息，请参阅 所有操作共享的属性。

> **Note**
>
> 要添加更多步骤，请选择添加步骤，然后重复创建步骤的过程。要删除步骤，请选择删除步骤。

12. 选择创建自动化以保存运行手册。

创建运行脚本的运行手册

以下过程展示了如何在 AWS Systems Manager Automation 控制台中使用文档生成器创建运行脚本的自定义运行手册。

您创建的运行手册的第一步运行一个脚本以启动 Amazon Elastic Compute Cloud (Amazon EC2) 实例。第二步运行另一个脚本来监控要更改为 ok 的实例状态检查。然后，报告自动化的整体状态 Success。

开始前的准备工作

确保您已完成以下步骤：

- 确认您具有管理员权限，或为您授予了相应的权限以在 AWS Identity and Access Management (IAM) 中访问 Systems Manager。

  有关信息，请参阅验证用户的运行手册访问权限。
- 确认在您的 AWS 账户 账户中具有自动化的 IAM 服务角色（也称为担任角色）。该角色是必需的，因为该演练使用 aws:executeScript 操作。

  有关创建此角色的信息，请参阅 为自动化配置服务角色（担任角色）访问权限。

  有关用于运行 aws:executeScript 的 IAM 服务角色要求的信息，请参阅 使用运行手册的权限。
- 确认您具有启动 EC2 实例的权限。

  有关更多信息，请参阅《Amazon EC2 用户指南》中的 IAM 与 Amazon EC2。

创建使用文档生成器运行脚本的自定义运行手册

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择文档。
3. 选择创建自动化。
4. 对于名称，请为运行手册键入以下描述性名称：**LaunchInstanceAndCheckStatus**。
5. （可选）对于文档描述)，请使用 Markdown 将默认文本替换为该运行手册的描述。示例如下：

```
##Title: LaunchInstanceAndCheckState
    -----
```

```
   **Purpose**: This runbook first launches an EC2 instance using the AMI
 ID provided in the parameter ```imageId```. The second step of this runbook
 continuously checks the instance status check value for the launched instance
 until the status ```ok``` is returned.

   ##Parameters:
   -----
   Name | Type | Description | Default Value
   ------------- | ------------- | ------------- | -------------
   assumeRole | String | (Optional) The ARN of the role that allows Automation to
 perform the actions on your behalf. | -
   imageId  | String | (Optional) The AMI ID to use for launching the instance.
 The default value uses the latest Amazon Linux AMI ID available. | {{ ssm:/aws/
service/ami-amazon-linux-latest/amzn-ami-hvm-x86_64-gp2 }}
```

6. 对于担任角色，请使用 **arn:aws:iam::111122223333:role/AutomationServiceRole** 格式输入自动化的自动化 IAM 服务角色（担任角色）的 ARN。使用您的 AWS 账户 ID 替代 111122223333。

   您指定的角色用于提供启动自动化执行所需的权限。

   > **Important**
   >
   > 对于不归 Amazon 所有并使用 `aws:executeScript` 操作的运行手册，必须指定一个角色。有关信息，请参阅使用运行手册的权限。

7. 展开输入参数，然后执行以下操作。

   1. 对于参数名称，请输入 **imageId**。
   2. 对于类型，选择 **String**。
   3. 对于必需，请选择 No。
   4. 对于默认值，请输入以下内容。

      ```
      {{ ssm:/aws/service/ami-amazon-linux-latest/amzn-ami-hvm-x86_64-gp2 }}
      ```

      > **Note**
      >
      > 该值使用最新的 Amazon Linux 1 Amazon Machine Image（AMI）ID 启动 Amazon EC2 实例。如果要使用不同的 AMI，请将该值替换为您的 AMI ID。

5. 对于描述，请输入以下内容。

```
(Optional) The AMI ID to use for launching the instance. The default value uses
 the latest released Amazon Linux AMI ID.
```

8. 选择添加参数)以创建第二个参数 **tagValue**，然后输入以下内容。

1. 对于参数名称，请输入 **tagValue**。
2. 对于类型，选择 **String**。
3. 对于必需，请选择 No。
4. 对于默认值，请输入 **LaunchedBySsmAutomation**。这会将标签键对值 Name:LaunchedBySsmAutomation 添加到实例中。
5. 对于描述，请输入以下内容。

```
(Optional) The tag value to add to the instance. The default value is
 LaunchedBySsmAutomation.
```

9. 选择添加参数 以创建第三个参数 **instanceType**，然后输入以下信息。

1. 对于参数名称，请输入 **instanceType**。
2. 对于类型，选择 **String**。
3. 对于必需，请选择 No。
4. 对于默认值，请输入 **t2.micro**。
5. 对于参数描述，请输入以下内容。

```
(Optional) The instance type to use for the instance. The default value is
 t2.micro.
```

10. 展开目标类型，然后选择 **"/"**。
11. （可选）展开文档标签以将资源标签应用于运行手册。为标签键 输入 **Purpose**，并为标签值输入 **LaunchInstanceAndCheckState**。
12. 在步骤 1 部分中，完成以下步骤。

1. 对于步骤名称，为自动化的第一步输入以下描述性步骤名称：**LaunchEc2Instance**。
2. 对于操作类型，请选择运行脚本) (**aws:executeScript**)。
3. 对于描述，请输入自动化步骤的描述，例如以下内容。

```
**About This Step**

   This step first launches an EC2 instance using the ```aws:executeScript```
 action and the provided script.
```

4. 展开输入。
5. 对于运行时，请选择运行提供的脚本所使用的运行时语言。
6. 对于处理程序，输入 **launch_instance**。这是在以下脚本中声明的函数名称。

> **Note**
>
> 这对于 PowerShell 不是必需的。

7. 对于脚本，请将默认内容替换为以下内容。请确保将脚本与相应的运行时值匹配。

   Python

```
def launch_instance(events, context):
      import boto3
      ec2 = boto3.client('ec2')

      image_id = events['image_id']
      tag_value = events['tag_value']
      instance_type = events['instance_type']

      tag_config = {'ResourceType': 'instance', 'Tags': [{'Key':'Name',
 'Value':tag_value}]}

      res = ec2.run_instances(ImageId=image_id, InstanceType=instance_type,
 MaxCount=1, MinCount=1, TagSpecifications=[tag_config])

      instance_id = res['Instances'][0]['InstanceId']

      print('[INFO] 1 EC2 instance is successfully launched', instance_id)

      return { 'InstanceId' : instance_id }
```

   PowerShell

```
Install-Module AWS.Tools.EC2 -Force
    Import-Module AWS.Tools.EC2
```

```
    $payload = $env:InputPayload | ConvertFrom-Json

    $imageid = $payload.image_id

    $tagvalue = $payload.tag_value

    $instanceType = $payload.instance_type

    $type = New-Object Amazon.EC2.InstanceType -ArgumentList $instanceType

    $resource = New-Object Amazon.EC2.ResourceType -ArgumentList 'instance'

    $tag = @{Key='Name';Value=$tagValue}

    $tagSpecs = New-Object Amazon.EC2.Model.TagSpecification

    $tagSpecs.ResourceType = $resource

    $tagSpecs.Tags.Add($tag)

    $res = New-EC2Instance -ImageId $imageId -MinCount 1 -MaxCount 1 -
InstanceType $type -TagSpecification $tagSpecs

    return @{'InstanceId'=$res.Instances.InstanceId}
```

8. 展开其他输入。
9. 对于输入名称，请选择 InputPayload。对于输入值，请输入以下 YAML 数据。

```
image_id: "{{ imageId }}"
    tag_value: "{{ tagValue }}"
    instance_type: "{{ instanceType }}"
```

13. 展开输出，然后执行以下操作：
    - 对于名称，请输入 **payload**。
    - 对于选择器，请输入 **$.Payload**。
    - 对于类型，选择 StringMap。
14. 选择添加步骤将第二步添加到运行手册中。第二步查询在步骤 1 中启动的实例的状态，并等到返回的状态为 ok。
15. 在 Step 2 (步骤 2) 部分中，执行以下操作。

1. 对于步骤名称，请为自动化的第二步输入以下描述性名称：**WaitForInstanceStatusOk**。
2. 对于操作类型，请选择运行脚本 (**aws:executeScript**)。
3. 对于描述，请输入自动化步骤的描述，例如以下内容。

```
**About This Step**

   The script continuously polls the instance status check value for the
 instance launched in Step 1 until the ```ok``` status is returned.
```

4. 对于运行时，选择用于执行提供的脚本的运行时语言。
5. 对于处理程序，输入 **poll_instance**。这是在以下脚本中声明的函数名称。

> **Note**
>
> 这对于 PowerShell 不是必需的。

6. 对于脚本，请将默认内容替换为以下内容。请确保将脚本与相应的运行时值匹配。

   Python

```
def poll_instance(events, context):
      import boto3
      import time

      ec2 = boto3.client('ec2')

      instance_id = events['InstanceId']

      print('[INFO] Waiting for instance status check to report ok',
 instance_id)

      instance_status = "null"

      while True:
        res = ec2.describe_instance_status(InstanceIds=[instance_id])

        if len(res['InstanceStatuses']) == 0:
          print("Instance status information is not available yet")
          time.sleep(5)
          continue
```

```
        instance_status = res['InstanceStatuses'][0]['InstanceStatus']
['Status']

        print('[INFO] Polling to get status of the instance', instance_status)

        if instance_status == 'ok':
          break

        time.sleep(10)

      return {'Status': instance_status, 'InstanceId': instance_id}
```

PowerShell

```
    Install-Module AWS.Tools.EC2 -Force

    $inputPayload = $env:InputPayload | ConvertFrom-Json

    $instanceId = $inputPayload.payload.InstanceId

    $status = Get-EC2InstanceStatus -InstanceId $instanceId

    while ($status.Status.Status -ne 'ok'){
       Write-Host 'Polling get status of the instance', $instanceId

       Start-Sleep -Seconds 5

       $status = Get-EC2InstanceStatus -InstanceId $instanceId
    }

    return @{Status = $status.Status.Status; InstanceId = $instanceId}
```

7. 展开其他输入。
8. 对于输入名称，请选择 InputPayload。对于输入值，请输入以下内容：

```
{{ LaunchEc2Instance.payload }}
```

16. 选择创建自动化以保存运行手册。

在运行手册中使用脚本

自动化运行手册支持将脚本作为自动化的一部分运行。Automation 是 AWS Systems Manager 中的一项工具。通过使用运行手册，您可以直接在 AWS 中运行脚本，而无需创建单独的计算环境来运行脚本。由于运行手册可以将脚本步骤与其他自动化步骤类型（例如批准）一起运行，因此您能够在紧急或不明确的情况下手动进行干预。您可以把来自运行手册中 `aws:executeScript` 操作的输出添加到 Amazon CloudWatch Logs。有关更多信息，请参阅 使用 CloudWatch Logs 记录自动化操作输出。

使用运行手册的权限

要使用运行手册，Systems Manager 必须使用 AWS Identity and Access Management (IAM) 角色的权限。自动化用于确定使用哪个角色的权限的方法取决于一些因素，以及步骤是否使用 `aws:executeScript` 操作。

对于不使用 `aws:executeScript` 的运行手册，自动化将使用以下两种权限来源之一：

- 在运行手册中指定或作为参数传递的 IAM 服务角色或担任角色的权限。
- 如果未指定 IAM 服务角色，则为启动自动化的用户的权限。

不过，如果运行手册中的步骤包含 `aws:executeScript` 操作，并且为该操作指定的 Python 或 PowerShell 脚本调用任何 AWS API 操作，则始终需要使用 IAM 服务角色（担任角色）。自动化按以下顺序检查该角色：

- 在运行手册中指定或作为参数传递的 IAM 服务角色或担任角色的权限。
- 如果找不到任何角色，自动化尝试在没有任何权限的情况下运行为 `aws:executeScript` 指定的 Python 或 PowerShell 脚本。如果脚本调用 AWS API 操作（例如，Amazon EC2 `CreateImage` 操作）或者尝试对 AWS 资源（例如 EC2 实例）执行操作，则包含脚本的步骤将失败，并且 Systems Manager 返回一条错误消息以报告失败。

将脚本添加到运行手册

您可以通过内联方式将脚本作为运行手册中的步骤的一部分，从而将其添加到运行手册中。您还可以从本地计算机中上传脚本或指定脚本所在的 Amazon Simple Storage Service (Amazon S3) 存储桶，以将脚本附加到运行手册。在运行脚本的步骤完成后，脚本的输出将作为 JSON 对象提供，您可以将其作为运行手册中的后续步骤的输入。有关 `aws:executeScript` 操作以及如何使用脚本附件的更多信息，请参阅 `aws:executeScript` - 运行脚本。

运行手册的脚本限制

运行手册强制执行 5 个文件附加文件的限制。脚本可以采用 Python 脚本 (.py) 或 PowerShell Core 脚本 (.ps1) 的形式，也可以作为 .zip 文件中的内容附加。

在运行手册中使用条件语句

默认情况下，在运行手册的 mainSteps 部分中定义的步骤将按先后顺序运行。在一个操作完成后，mainSteps 部分中指定的下一个操作将开始。此外，如果一个操作无法运行，则整个自动化将失败（默认情况下）。可以使用本节所述的 aws:branch 自动化操作和运行手册选项来创建执行条件分支的自动化。也就是说，您可以创建在评估不同选项后跳转到不同步骤或在某个步骤完成时动态响应更改的自动化。以下是可用于创建动态自动化的选项的列表：

- **aws:branch**：此自动化操作让您能够创建一个动态自动化，该自动化可以在一个步骤中评估多个选项，然后根据评估结果跳转到运行手册中的不同步骤。
- **nextStep**：此选项指定在成功完成一个步骤后，接下来处理自动化中的哪个步骤。
- **isEnd**：此选项在特定步骤结束时停止自动化。该选项的默认值为 false。
- **isCritical**：此选项将一个步骤指定为成功完成自动化的关键步骤。如果具有此分派的步骤失败，自动化会将自动化的最终状态报告为 Failed。该选项的默认值为 true。
- **onFailure**：此选项指示自动化在失败时是应中止、继续还是转到其他步骤。该选项的默认值为 abort。

下一节介绍 aws:branch 自动化操作。有关 nextStep、isEnd、isCritical 和 onFailure 选项的更多信息，请参阅 示例 aws:branch 运行手册。

使用 **aws:branch** 操作

aws:branch 操作提供适用于自动化的大部分动态条件分支选项。如前所述，此操作让自动化能够在一个步骤中评估多个条件，然后根据评估结果跳转到新的步骤。aws:branch 操作的功能类似于编程中的 IF-ELIF-ELSE 语句。

下面是一个 aws:branch 步骤的 YAML 示例：

```
- name: ChooseOSforCommands
  action: aws:branch
  inputs:
    Choices:
    - NextStep: runPowerShellCommand
      Variable: "{{GetInstance.platform}}"
      StringEquals: Windows
```

```
    - NextStep: runShellCommand
      Variable: "{{GetInstance.platform}}"
      StringEquals: Linux
    Default:
      PostProcessing
```

在为步骤指定 aws:branch 操作时，请指定自动化必须评估的 Choices。自动化可以根据在运行手册的 Parameters 部分中指定的参数值来评估 Choices。自动化也可以基于上一步的输出来评估 Choices。

自动化使用布尔表达式评估每个选择。如果评估确定第一个选择为 true，自动化跳转到为该选择指定的步骤。如果评估确定第一个选择为 false，自动化程评估下一个选择。如果您的步骤包含三个或更多的 Choices，自动化按顺序评估每个选择，直到某个选择的评估结果为 true。然后，自动化跳转到为结果为 true 的选择指定的步骤。

如果所有 Choices 都为 true，则工作流程检查该步骤是否包含 Default 值。Default 值定义当所有选择都为 true 时工作流程应跳转到的步骤。如果没有为该步骤指定 Default 值，自动化处理文档中的下一个步骤。

以下是 YAML 的 aws:branch 步骤，命名为 chooseOSfromParameter。此步骤包含两个 Choices：(NextStep: runWindowsCommand) 和 (NextStep: runLinuxCommand)。自动化评估这些 Choices，以确定应为适当的操作系统运行哪个命令。每个选择的 Variable 都使用 {{OSName}}，这是运行手册作者在运行手册的 Parameters 部分中定义的参数。

```
mainSteps:
- name: chooseOSfromParameter
  action: aws:branch
  inputs:
    Choices:
    - NextStep: runWindowsCommand
      Variable: "{{OSName}}"
      StringEquals: Windows
    - NextStep: runLinuxCommand
      Variable: "{{OSName}}"
      StringEquals: Linux
```

以下是 YAML 的 aws:branch 步骤，命名为 chooseOSfromOutput。此步骤包含两个 Choices：(NextStep: runPowerShellCommand) 和 (NextStep: runShellCommand)。自动化评估这些 Choices，以确定应为适当的操作系统运行哪个命令。每个选择的 Variable 都使用 {{GetInstance.platform}}，这是文档中上一步的输出。此示例还包含一个名为 Default 的

选项。如果自动化评估两个的结果都为 Choices，而且两个选择均为 true，自动化跳转到名为 PostProcessing 的步骤。

```
mainSteps:
- name: chooseOSfromOutput
  action: aws:branch
  inputs:
    Choices:
    - NextStep: runPowerShellCommand
      Variable: "{{GetInstance.platform}}"
      StringEquals: Windows
    - NextStep: runShellCommand
      Variable: "{{GetInstance.platform}}"
      StringEquals: Linux
    Default:
      PostProcessing
```

在运行手册中创建 **aws:branch** 步骤

在运行手册中创建 aws:branch 步骤时，请定义自动化应对其进行评估的 Choices 以确定自动化应跳转到哪个步骤。如前所述，Choices 使用布尔表达式进行评估。每个选择都必须定义以下选项：

- NextStep：当指定的选择为 true 时，运行手册要处理的下一个步骤。
- 变量：指定在运行手册的 Parameters 部分中定义的参数的名称（在 Variables 部分中定义的参数），或指定上一步的输出对象。

  使用以下格式指定变量值。

  `Variable: "{{variable name}}"`

  使用以下格式指定参数值。

  `Variable: "{{parameter name}}"`

  使用以下格式指定输出对象变量：

  `Variable: "{{previousStepName.outputName}}"`

> **Note**
>
> 下一节关于创建输出变量更详细地介绍如何创建输出变量。

- Operation：用于评估选择的标准，例如 `StringEquals: Linux`。`aws:branch` 操作支持以下运算：

  字符串运算

  - 字符串等于
  - EqualsIgnoreCase
  - StartsWith
  - EndsWith
  - 包含

  数值运算

  - NumericEquals
  - NumericGreater
  - NumericLesser
  - NumericGreaterOrEquals
  - NumericLesser
  - NumericLesserOrEquals

  布尔运算

  - BooleanEquals

  > **Important**
  >
  > 创建运行手册时，系统将验证运行手册中的每个操作。在尝试创建运行手册时，如果某个操作不受支持，系统会返回错误。

- Default：指定当所有 `Choices` 都为 `true` 时自动化应跳转到的回退步骤。

  > **Note**
  >
  > 如果不想指定 `Default` 值，则可以指定 `isEnd` 选项。如果所有 `Choices` 都为 `true` 并且未指定 `Default` 值，自动化在此步骤结束时停止。

使用以下模板可以帮助您在运行手册中构建 aws:branch 步骤：将每个#######替换为您自己的信息。

YAML

```
mainSteps:
- name: step name
  action: aws:branch
  inputs:
    Choices:
    - NextStep: step to jump to if evaluation for this choice is true
      Variable: "{{parameter name or output from previous step}}"
      Operation type: Operation value
    - NextStep: step to jump to if evaluation for this choice is true
      Variable: "{{parameter name or output from previous step}}"
      Operation type: Operation value
    Default:
      step to jump to if all choices are false
```

JSON

```
{
   "mainSteps":[
      {
         "name":"a name for the step",
         "action":"aws:branch",
         "inputs":{
            "Choices":[
               {
                  "NextStep":"step to jump to if evaluation for this choice is
 true",
                  "Variable":"{{parameter name or output from previous step}}",
                  "Operation type":"Operation value"
               },
               {
                  "NextStep":"step to jump to if evaluation for this choice is
 true",
                  "Variable":"{{parameter name or output from previous step}}",
                  "Operation type":"Operation value"
               }
            ],
            "Default":"step to jump to if all choices are false"
         }
```

```
      }
    ]
}
```

关于创建输出变量

要创建引用上一步输出的 aws:branch 选择，您需要标识上一步的名称和输出字段的名称。然后，使用以下格式组合步骤和字段的名称：

`Variable: "{{previousStepName.outputName}}"`

例如，以下示例中的第一个步骤名为 GetInstance。然后，在 outputs 下，有一个名为 platform 的字段。在第二个步骤 (ChooseOSforCommands) 中，作者希望将平台字段的输出引用为变量。要创建变量，只需将步骤名称 (GetInstance) 与输出字段名称 (platform) 组合在一起即可创建 `Variable: "{{GetInstance.platform}}"`。

```
mainSteps:
- Name: GetInstance
  action: aws:executeAwsApi
  inputs:
    Service: ssm
    Api: DescribeInstanceInformation
    Filters:
    - Key: InstanceIds
      Values: ["{{ InstanceId }}"]
  outputs:
  - Name: myInstance
    Selector: "$.InstanceInformationList[0].InstanceId"
    Type: String
  - Name: platform
    Selector: "$.InstanceInformationList[0].PlatformType"
    Type: String
- name: ChooseOSforCommands
  action: aws:branch
  inputs:
    Choices:
    - NextStep: runPowerShellCommand
      Variable: "{{GetInstance.platform}}"
      StringEquals: Windows
    - NextStep: runShellCommand
      Variable: "{{GetInstance.platform}}"
      StringEquals: Linux
```

```
    Default:
      Sleep
```

这是一个示例，展示了从上一步和输出创建 *"Variable": "{{ describeInstance.Platform }}"* 的方法。

```
- name: describeInstance
  action: aws:executeAwsApi
  onFailure: Abort
  inputs:
    Service: ec2
    Api: DescribeInstances
    InstanceIds:
    - "{{ InstanceId }}"
  outputs:
  - Name: Platform
    Selector: "$.Reservations[0].Instances[0].Platform"
    Type: String
  nextStep: branchOnInstancePlatform
- name: branchOnInstancePlatform
  action: aws:branch
  inputs:
    Choices:
    - NextStep: runEC2RescueForWindows
      Variable: "{{ describeInstance.Platform }}"
      StringEquals: windows
    Default: runEC2RescueForLinux
```

示例 **aws:branch** 运行手册

下面是一些使用 aws:branch 的示例运行手册。

示例 1：使用 **aws:branch** 和输出变量基于操作系统类型运行命令

在此示例的第一步（GetInstance）中，运行手册作者使用 aws:executeAwsApi 操作来调用 ssm DescribeInstanceInformation API 操作。作者使用此操作确定实例使用的操作系统的类型。aws:executeAwsApi 操作输出实例 ID 和平台类型。

在第二个步骤 (ChooseOSforCommands) 中，作者使用了 aws:branch 操作和两个 Choices：(NextStep: runPowerShellCommand) 和 (NextStep: runShellCommand)。自动化使用上一步 (Variable: "{{GetInstance.platform}}") 的输出评估实例的操作系统。自动化跳转到指定操作系统的步骤。

```
---
schemaVersion: '0.3'
assumeRole: "{{AutomationAssumeRole}}"
parameters:
  AutomationAssumeRole:
    default: ""
    type: String
mainSteps:
- name: GetInstance
  action: aws:executeAwsApi
  inputs:
    Service: ssm
    Api: DescribeInstanceInformation
  outputs:
  - Name: myInstance
    Selector: "$.InstanceInformationList[0].InstanceId"
    Type: String
  - Name: platform
    Selector: "$.InstanceInformationList[0].PlatformType"
    Type: String
- name: ChooseOSforCommands
  action: aws:branch
  inputs:
    Choices:
    - NextStep: runPowerShellCommand
      Variable: "{{GetInstance.platform}}"
      StringEquals: Windows
    - NextStep: runShellCommand
      Variable: "{{GetInstance.platform}}"
      StringEquals: Linux
    Default:
      Sleep
- name: runShellCommand
  action: aws:runCommand
  inputs:
    DocumentName: AWS-RunShellScript
    InstanceIds:
    - "{{GetInstance.myInstance}}"
    Parameters:
      commands:
      - ls
  isEnd: true
- name: runPowerShellCommand
```

```
  action: aws:runCommand
  inputs:
    DocumentName: AWS-RunPowerShellScript
    InstanceIds:
    - "{{GetInstance.myInstance}}"
    Parameters:
      commands:
      - ls
  isEnd: true
- name: Sleep
  action: aws:sleep
  inputs:
    Duration: PT3S
```

示例 2：使用 **aws:branch** 和参数变量基于操作系统类型运行命令

运行手册作者在 parameters 部分的运行手册开头定义了几个参数选项。一个参数名为 OperatingSystemName。在第一个步骤 (ChooseOS) 中，作者使用了 aws:branch 操作和两个 Choices：(NextStep: runWindowsCommand) 和 (NextStep: runLinuxCommand)。这些 Choices 的变量引用在参数部分 (Variable: "{{OperatingSystemName}}") 中指定的参数选项。当用户运行此自动化时，他们在运行时为 OperatingSystemName 指定值。自动化在 Choices 评估期间使用运行时参数。自动化基于为 OperatingSystemName 指定的运行时参数跳转到指定操作系统的步骤。

```
---
schemaVersion: '0.3'
assumeRole: "{{AutomationAssumeRole}}"
parameters:
  AutomationAssumeRole:
    default: ""
    type: String
  OperatingSystemName:
    type: String
  LinuxInstanceId:
    type: String
  WindowsInstanceId:
    type: String
mainSteps:
- name: ChooseOS
  action: aws:branch
  inputs:
    Choices:
```

```
    - NextStep: runWindowsCommand
      Variable: "{{OperatingSystemName}}"
      StringEquals: windows
    - NextStep: runLinuxCommand
      Variable: "{{OperatingSystemName}}"
      StringEquals: linux
    Default:
      Sleep
- name: runLinuxCommand
  action: aws:runCommand
  inputs:
    DocumentName: "AWS-RunShellScript"
    InstanceIds:
    - "{{LinuxInstanceId}}"
    Parameters:
      commands:
      - ls
  isEnd: true
- name: runWindowsCommand
  action: aws:runCommand
  inputs:
    DocumentName: "AWS-RunPowerShellScript"
    InstanceIds:
    - "{{WindowsInstanceId}}"
    Parameters:
      commands:
      - date
  isEnd: true
- name: Sleep
  action: aws:sleep
  inputs:
    Duration: PT3S
```

使用运算符创建复杂的分支自动化

您可以在 aws:branch 步骤中使用 And、Or 和 Not 运算符来创建复杂的分支自动化。

“And”运算符

如果某个选择的多个变量需要为 true，可以使用 And 运算符。在下面的示例中，第一个选择评估实例是否 running 并且使用的是 Windows 操作系统。如果这两个变量的评估结果都为真，自动化跳转到 runPowerShellCommand 步骤。如果一个或多个变量为 false，自动化评估第二个选择的变量。

```
mainSteps:
- name: switch2
  action: aws:branch
  inputs:
    Choices:
    - And:
      - Variable: "{{GetInstance.pingStatus}}"
        StringEquals: running
      - Variable: "{{GetInstance.platform}}"
        StringEquals: Windows
      NextStep: runPowerShellCommand

    - And:
      - Variable: "{{GetInstance.pingStatus}}"
        StringEquals: running
      - Variable: "{{GetInstance.platform}}"
        StringEquals: Linux
      NextStep: runShellCommand
    Default:
      sleep3
```

“Or”运算符

如果某个选择的多个变量需要为真，可以使用 Or 运算符。在下面的示例中，第一个选择评估参数字符串是不是 Windows 和 AWS Lambda 步骤的输出是不是真。如果任意一个变量的评估结果都为真，自动化跳转到 RunPowerShellCommand 步骤。如果两个变量都为假，自动化评估第二个选择的变量。

```
- Or:
  - Variable: "{{parameter1}}"
    StringEquals: Windows
  - Variable: "{{BooleanParam1}}"
    BooleanEquals: true
  NextStep: RunPowershellCommand
- Or:
  - Variable: "{{parameter2}}"
    StringEquals: Linux
  - Variable: "{{BooleanParam2}}"
    BooleanEquals: true
  NextStep: RunShellScript
```

“Not”运算符

当变量不为真时，如果您想要跳转到定义的步骤，则使用 Not 运算符。在下面的示例中，第一个选择评估参数字符串是不是 Not Linux。如果变量的评估结果不为 Linux，自动化跳转到 sleep2 步骤。如果第一个选择的评估结果为 Linux，自动化评估下一个选择。

```
mainSteps:
- name: switch
  action: aws:branch
  inputs:
    Choices:
    - NextStep: sleep2
      Not:
        Variable: "{{testParam}}"
        StringEquals: Linux
    - NextStep: sleep1
      Variable: "{{testParam}}"
      StringEquals: Windows
    Default:
      sleep3
```

如何使用条件选项的示例

本部分包括有关如何使用运行手册中的动态选项的其他示例。本部分中的每个示例均扩展以下运行手册。该运行手册包含两个操作。第一个操作名为 InstallMsiPackage。它使用 aws:runCommand 操作将应用程序安装到 Windows Server 实例上。第二个操作名为 TestInstall。它使用 aws:invokeLambdaFunction 操作在成功安装应用程序后测试该应用程序。步骤 1 指定 onFailure: Abort。这意味着，如果应用程序未成功安装，自动化会在进入步骤 2 前停止。

示例 1：包含两个线性操作的运行手册

```
---
schemaVersion: '0.3'
description: Install MSI package and run validation.
assumeRole: "{{automationAssumeRole}}"
parameters:
  automationAssumeRole:
    type: String
    description: "(Required) Assume role."
  packageName:
    type: String
    description: "(Required) MSI package to be installed."
  instanceIds:
    type: String
```

```
    description: "(Required) Comma separated list of instances."
mainSteps:
- name: InstallMsiPackage
  action: aws:runCommand
  maxAttempts: 2
  onFailure: Abort
  inputs:
    InstanceIds:
    - "{{instanceIds}}"
    DocumentName: AWS-RunPowerShellScript
    Parameters:
      commands:
      - msiexec /i {{packageName}}
- name: TestInstall
  action: aws:invokeLambdaFunction
  maxAttempts: 1
  timeoutSeconds: 500
  inputs:
    FunctionName: TestLambdaFunction
...
```

使用 **onFailure** 选项创建跳转到不同步骤的动态自动化

下面的示例使用 onFailure: step:*step name*、nextStep 和 isEnd 选项创建一个动态自动化。在此示例中，如果 InstallMsiPackage 操作失败，自动化将跳转到名为 PostFailure (onFailure: step:PostFailure) 的操作以运行 AWS Lambda 函数，从而在安装失败时执行某个操作。如果安装成功，自动化将跳转到 TestInstall 操作 (nextStep: TestInstall)。TestInstall 和 PostFailure 步骤都使用 isEnd 选项 (isEnd: true)，以便自动化在任一步骤完成时结束。

> **Note**
>
> 可以选择在 mainSteps 部分的最后一步中使用 isEnd 选项。如果最后一个步骤未跳转到其他步骤，自动化会在最后一步中运行操作后停止。

示例 2：跳转到不同步骤的动态自动化

```
mainSteps
- name: InstallMsiPackage
  action: aws:runCommand
  onFailure: step:PostFailure
  maxAttempts: 2
```

```
  inputs:
    InstanceIds:
    - "{{instanceIds}}"
    DocumentName: AWS-RunPowerShellScript
    Parameters:
      commands:
      - msiexec /i {{packageName}}
  nextStep: TestInstall
- name: TestInstall
  action: aws:invokeLambdaFunction
  maxAttempts: 1
  timeoutSeconds: 500
  inputs:
    FunctionName: TestLambdaFunction
  isEnd: true
- name: PostFailure
  action: aws:invokeLambdaFunction
  maxAttempts: 1
  timeoutSeconds: 500
  inputs:
    FunctionName: PostFailureRecoveryLambdaFunction
  isEnd: true
...
```

> **Note**
>
> 在处理运行手册之前，系统将验证运行手册不会创建无限循环。如果检测到无限循环，自动化将返回一个错误和一个显示哪些步骤创建循环的循环跟踪。

创建定义关键步骤的动态自动化

您可以指定一个对于自动化的整体成功很重要的步骤。如果关键步骤失败，自动化会将自动化状态报告为 `Failed`，即使已成功运行一个或多个步骤也是如此。在以下示例中，用户在 InstallMsiPackage 步骤失败 (`onFailure: step:VerifyDependencies`) 时标识 VerifyDependencies 步骤。用户指定 `InstallMsiPackage` 步骤为非关键步骤 (`isCritical: false`)。在此示例中，如果应用程序安装失败，自动化处理 `VerifyDependencies` 步骤以确定是否因一个或多个依赖项缺失而导致应用程序安装失败。

示例 3：定义自动化的关键步骤

```
---
```

```
name: InstallMsiPackage
action: aws:runCommand
onFailure: step:VerifyDependencies
isCritical: false
maxAttempts: 2
inputs:
  InstanceIds:
  - "{{instanceIds}}"
  DocumentName: AWS-RunPowerShellScript
  Parameters:
    commands:
    - msiexec /i {{packageName}}
nextStep: TestPackage
...
```

使用操作输出作为输入

多个自动化操作会返回预定义的输出。您可以使用格式 {{*stepName*.*outputName*}} 将这些输出作为输入传递到运行手册的后续步骤。您还可以在运行手册中为自动化操作定义自定义输出。让您可以运行脚本，或调用其他 AWS 服务 的 API 操作一次，以便您在后续操作中重复使用这些值作为输入。运行手册中的参数类型是静态的。这意味着参数类型在定义后无法更改。要定义步骤输出，请提供以下字段：

- 名称：（必需）输出名称，用于在后续步骤中引用输出值。
- 选择器：（必需）用于确定输出值的 JSONPath 表达式。
- 类型：（可选）选择器字段返回的值的数据类型。有效的类型值为 String、Integer、Boolean、StringList、StringMap、MapList。默认值为 String。

如果输出的值与您指定的数据类型不匹配，自动化会尝试转换该数据类型。例如，如果返回的值是 Integer，但指定的 Type 是 String，则最终输出值为 String 值。支持以下类型转换：

- String 值可以转换为 StringList、Integer 和 Boolean。
- Integer 值可以转换为 String 和 StringList。
- Boolean 值可以转换为 String 和 StringList。
- 包含一个元素的 StringList、IntegerList 或 BooleanList 值可以转换为 String、Integer 或 Boolean。

将参数或输出与自动化操作一起使用时，不能在操作的输入中动态更改数据类型。

这份示例运行手册演示了如何定义操作输出，并将该值引用为后续操作的输入。运行手册执行以下操作：

- 使用 aws:executeAwsApi 操作调用 Amazon EC2 DescribeImages API 操作来获取特定 Windows Server 2016 AMI 的名称。它将映像 ID 输出为 ImageId。
- 使用 aws:executeAwsApi 操作调用 Amazon EC2 RunInstances API 操作来启动一个实例，它使用上一步中的 ImageId。它将实例 ID 输出为 InstanceId。
- 使用 aws:waitForAwsResourceProperty 操作轮询 Amazon EC2 DescribeInstanceStatus API 操作来等待实例进入 running 状态。此操作的超时时间为 60 秒。如果实例在轮询 60 秒后未能进入 running 状态，则此步骤超时。
- 使用 aws:assertAwsResourceProperty 操作调用 Amazon EC2 DescribeInstanceStatus API 操作来断言实例处于 running 状态。如果实例状态不为 running，则此步骤失败。

```
---
description: Sample runbook using AWS API operations
schemaVersion: '0.3'
assumeRole: "{{ AutomationAssumeRole }}"
parameters:
  AutomationAssumeRole:
    type: String
    description: "(Optional) The ARN of the role that allows Automation to perform the
 actions on your behalf."
    default: ''
  ImageName:
    type: String
    description: "(Optional) Image Name to launch EC2 instance with."
    default: "Windows_Server-2022-English-Full-Base*"
mainSteps:
- name: getImageId
  action: aws:executeAwsApi
  inputs:
    Service: ec2
    Api: DescribeImages
    Filters:
    - Name: "name"
      Values:
      - "{{ ImageName }}"
  outputs:
  - Name: ImageId
    Selector: "$.Images[0].ImageId"
```

```
    Type: "String"
- name: launchOneInstance
  action: aws:executeAwsApi
  inputs:
    Service: ec2
    Api: RunInstances
    ImageId: "{{ getImageId.ImageId }}"
    MaxCount: 1
    MinCount: 1
  outputs:
  - Name: InstanceId
    Selector: "$.Instances[0].InstanceId"
    Type: "String"
- name: waitUntilInstanceStateRunning
  action: aws:waitForAwsResourceProperty
  timeoutSeconds: 60
  inputs:
    Service: ec2
    Api: DescribeInstanceStatus
    InstanceIds:
    - "{{ launchOneInstance.InstanceId }}"
    PropertySelector: "$.InstanceStatuses[0].InstanceState.Name"
    DesiredValues:
    - running
- name: assertInstanceStateRunning
  action: aws:assertAwsResourceProperty
  inputs:
    Service: ec2
    Api: DescribeInstanceStatus
    InstanceIds:
    - "{{ launchOneInstance.InstanceId }}"
    PropertySelector: "$.InstanceStatuses[0].InstanceState.Name"
    DesiredValues:
    - running
outputs:
- "launchOneInstance.InstanceId"
...
```

前面介绍的每个自动化操作都允许您通过指定服务命名空间、API 操作名称、输入参数和输出参数来调用特定的 API 操作。输入由您选择的 API 操作定义。您可以在以下服务参考页面的左侧导航栏中选择服务来查看 API 操作（也称为方法）。在要调用的服务的客户端部分中选择一种方法。例如，以下页面中列出了 Amazon Relational Database Service (Amazon RDS) 的所有 API 操作（方法）：Amazon RDS 方法。

您可以在以下位置查看每个自动化操作的架构：

- aws:assertAwsResourceProperty - 断言 AWS 资源状态或事件状态
- aws:executeAwsApi - 调用并运行 AWS API 操作
- aws:waitForAwsResourceProperty - 等待 AWS 资源属性

架构包括使用每个操作所需字段的描述。

使用 Selector/PropertySelector 字段

每个自动化操作都要求您指定一个输出 Selector（用于 aws:executeAwsApi）或 PropertySelector（用于 aws:assertAwsResourceProperty 和 aws:waitForAwsResourceProperty）。这些字段用于处理 AWS API 操作的 JSON 响应。这些字段使用 JSONPath 语法。

以下是帮助说明这一概念的 aws:executeAwsAPi 操作的示例：

```
---
mainSteps:
- name: getImageId
  action: aws:executeAwsApi
  inputs:
    Service: ec2
    Api: DescribeImages
    Filters:
      - Name: "name"
        Values:
          - "{{ ImageName }}"
  outputs:
    - Name: ImageId
      Selector: "$.Images[0].ImageId"
      Type: "String"
...
```

在 aws:executeAwsApi 步骤 getImageId 中，自动化将调用 DescribeImages API 操作并接收来自 ec2 的响应。然后，自动化将 Selector - "$.Images[0].ImageId" 应用于 API 响应，并将所选值分配给输出 ImageId 变量。通过指定 "{{ getImageId.ImageId }}"，同一自动化中的其他步骤可以使用 ImageId 的值。

以下是帮助说明这一概念的 aws:waitForAwsResourceProperty 操作的示例：

```
---
- name: waitUntilInstanceStateRunning
  action: aws:waitForAwsResourceProperty
  # timeout is strongly encouraged for action - aws:waitForAwsResourceProperty
  timeoutSeconds: 60
  inputs:
    Service: ec2
    Api: DescribeInstanceStatus
    InstanceIds:
    - "{{ launchOneInstance.InstanceId }}"
    PropertySelector: "$.InstanceStatuses[0].InstanceState.Name"
    DesiredValues:
    - running
...
```

在 aws:waitForAwsResourceProperty 步骤 waitUntilInstanceStateRunning 中，自动化将调用 DescribeInstanceStatus API 操作并接收来自 ec2 的响应。然后，自动化将 PropertySelector - "$.InstanceStatuses[0].InstanceState.Name" 应用于响应，并检查指定的返回值是否与 DesiredValues 列表中的值匹配（在本例中为 running）。此步骤重复这一过程，直到响应返回 running 实例状态。

在运行手册中使用 JSONPath

JSONPath 表达式是以“$”开头的字符串。用于选择 JSON 元素中的一个或多个组件。下面的列表包含有关 Systems Manager 自动化支持的 JSONPath 运算符的信息：

- 点表示的子字段 (.)：用于 JSON 对象。此运算符选择特定键的值。
- 深层扫描 (..)：用于 JSON 元素。此运算符逐级扫描 JSON 元素，并使用特定键选择值列表。此运算符的返回类型始终为 JSON 数组。在自动化步骤输出类型上下文中，运算符可以是 StringList 或 MapList。
- 数组索引 ([ ])：用于 JSON 数组。此运算符获取特定索引的值。
- 筛选条件 ([?(***expression***)])：与 JSON 数组一起使用。此运算符筛选条件 JSON 数组值与筛选条件表达式中定义的标准相匹配。筛选条件表达式只能使用以下运算符：==、!=、>、<、>= 或 <=。不支持将多个筛选条件表达式与 AND (&&) 或 OR (||) 组合使用。此运算符的返回类型始终为 JSON 数组。

为了更好地理解 JSONPath 运算符，请查看以下 ec2 DescribeInstances API 操作的 JSON 响应。此响应下面提供了几个示例，它们说明通过向 DescribeInstances API 操作响应应用不同的 JSONPath 表达式获取的不同结果。

```
{
    "NextToken": "abcdefg",
    "Reservations": [
        {
            "OwnerId": "123456789012",
            "ReservationId": "r-abcd12345678910",
            "Instances": [
                {
                    "ImageId": "ami-12345678",
                    "BlockDeviceMappings": [
                        {
                            "Ebs": {
                                "DeleteOnTermination": true,
                                "Status": "attached",
                                "VolumeId": "vol-000000000000"
                            },
                            "DeviceName": "/dev/xvda"
                        }
                    ],
                    "State": {
                        "Code": 16,
                        "Name": "running"
                    }
                }
            ],
            "Groups": []
        },
        {
            "OwnerId": "123456789012",
            "ReservationId": "r-12345678910abcd",
            "Instances": [
                {
                    "ImageId": "ami-12345678",
                    "BlockDeviceMappings": [
                        {
                            "Ebs": {
                                "DeleteOnTermination": true,
                                "Status": "attached",
                                "VolumeId": "vol-111111111111"
                            },
                            "DeviceName": "/dev/xvda"
                        }
                    ],
```

```
                    "State": {
                        "Code": 80,
                        "Name": "stopped"
                    }
                }
            ],
            "Groups": []
        }
    ]
}
```

JSONPath 示例 1：从 JSON 响应获取特定的 String

```
JSONPath:
$.Reservations[0].Instances[0].ImageId

Returns:
"ami-12345678"

Type: String
```

JSONPath 示例 2：从 JSON 响应获取特定的 Boolean

```
JSONPath:
$.Reservations[0].Instances[0].BlockDeviceMappings[0].Ebs.DeleteOnTermination

Returns:
true

Type: Boolean
```

JSONPath 示例 3：从 JSON 响应获取特定的 Integer

```
JSONPath:
$.Reservations[0].Instances[0].State.Code

Returns:
16

Type: Integer
```

JSONPath 示例 4：深层扫描 JSON 响应，然后以 StringList 的形式获取 VolumeId 的所有值

```
JSONPath:
$.Reservations..BlockDeviceMappings..VolumeId

Returns:
[
   "vol-000000000000",
   "vol-111111111111"
]

Type: StringList
```

JSONPath 示例 5：以 StringMap 的形式获取特定的 BlockDeviceMappings 对象

```
JSONPath:
$.Reservations[0].Instances[0].BlockDeviceMappings[0]

Returns:
{
   "Ebs" : {
      "DeleteOnTermination" : true,
      "Status" : "attached",
      "VolumeId" : "vol-000000000000"
   },
   "DeviceName" : "/dev/xvda"
}

Type: StringMap
```

JSONPath 示例 6：深层扫描 JSON 响应，然后以 MapList 的形式获取所有 State 对象

```
JSONPath:
$.Reservations..Instances..State

Returns:
[
   {
      "Code" : 16,
      "Name" : "running"
   },
   {
      "Code" : 80,
      "Name" : "stopped"
```

```
   }
]

Type: MapList
```

JSONPath 示例 7：筛选处于 **running** 状态的实例

```
JSONPath:
$.Reservations..Instances[?(@.State.Name == 'running')]

Returns:
[
  {
    "ImageId": "ami-12345678",
    "BlockDeviceMappings": [
      {
        "Ebs": {
          "DeleteOnTermination": true,
          "Status": "attached",
          "VolumeId": "vol-000000000000"
        },
        "DeviceName": "/dev/xvda"
      }
    ],
    "State": {
      "Code": 16,
      "Name": "running"
    }
  }
]

Type: MapList
```

JSONPath 示例 8：返回未处于 **running** 状态实例的 **ImageId**

```
JSONPath:
$.Reservations..Instances[?(@.State.Name != 'running')].ImageId

Returns:
[
  "ami-12345678"
]
```

```
Type: StringList | String
```

为 Automation 创建 Webhook 集成

要在自动化期间使用 Webhooks 发送消息，请创建集成。可以在自动化过程中使用您的运行手册中的 aws:invokeWebhook 操作调用集成。如果尚未创建 Webhook，请参阅 为集成创建 Webhooks。要了解有关 aws:invokeWebhook 操作的更多信息，请参阅 aws:invokeWebhook – 调用 Automation Webhook 集成。

如以下程序所示，您可以使用 Systems Manager Automation 控制台或您的首选命令行工具来创建集成。

创建集成（控制台）

要为 Automation 创建集成（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 自动化。
3. 请选择 Integrations（集成）选项卡。
4. 请选择 Add integration（添加集成），然后选择 Webhook。
5. 请输入要为集成包括的所需值和任何可选值。
6. 请选择 Add（添加）来创建集成。

创建集成（命令行）

要使用命令行工具创建集成，您必须为集成创建所需的 SecureString 参数。Automation 使用 Parameter Store（Systems Manager 中的一项工具）中的预留命名空间来存储有关集成的信息。如果您使用 AWS Management Console 创建集成，Automation 会为您处理此过程。在命名空间之后，您必须指定要创建的集成类型，然后指定集成的名称。目前，Automation 支持 webhook 类型集成。

webhook 类型集成支持的字段如下：

- 描述
- 标头
- payload
- URL

开始前的准备工作

安装并配置 AWS Command Line Interface (AWS CLI) 或 AWS Tools for PowerShell（如果尚未这样做）。有关信息，请参阅安装或更新 AWS CLI 的最新版本以及安装 AWS Tools for PowerShell。

要为 Automation 创建集成（命令行）

- 运行下列命令以为集成创建所需的 SecureString 参数。将每个*#######*替换为您自己的信息。/d9d01087-4a3f-49e0-b0b4-d568d7826553/ssm/integrations/webhook/ 命名空间在 Parameter Store 中为集成预留。参数的名称必须使用此命名空间，后跟集成的名称。例如 /d9d01087-4a3f-49e0-b0b4-d568d7826553/ssm/integrations/webhook/*myWebhookIntegration*。

  Linux & macOS

  ```
  aws ssm put-parameter \
      --name "/d9d01087-4a3f-49e0-b0b4-d568d7826553/ssm/integrations/
  webhook/myWebhookIntegration" \
      --type "SecureString" \
      --data-type "aws:ssm:integration" \
      --value '{"description": "My first webhook integration for Automation.",
   "url": "myWebHookURL"}'
  ```

  Windows

  ```
  aws ssm put-parameter ^
      --name "/d9d01087-4a3f-49e0-b0b4-d568d7826553/ssm/integrations/
  webhook/myWebhookIntegration" ^
      --type "SecureString" ^
      --data-type "aws:ssm:integration" ^
      --value  "{\"description\":\"My first webhook integration for Automation.\",
  \"url\":\"myWebHookURL\"}"
  ```

  PowerShell

  ```
  Write-SSMParameter `
      -Name "/d9d01087-4a3f-49e0-b0b4-d568d7826553/ssm/integrations/
  webhook/myWebhookIntegration" `
      -Type "SecureString"
      -DataType "aws:ssm:integration"
  ```

```
    -Value '{"description": "My first webhook integration for Automation.",
 "url": "myWebHookURL"}'
```

为集成创建 Webhooks

在使用您的提供商创建 Webhooks 时，请注意以下几点：

- 协议必须是 HTTPS。
- 支持自定义请求标头。
- 可以指定默认的请求正文。
- 当使用 aws:invokeWebhook 操作调用集成时，可以覆盖默认的请求正文。

处理运行手册中的超时

timeoutSeconds 属性由所有自动化操作共享。您可以使用此属性指定操作的执行超时值。此外，您还可以更改操作超时如何影响自动化和整体执行状态。另外，您可以通过定义操作的 onFailure 和 isCritical 共享属性来完成此操作。

例如，根据您的使用案例，您可能希望自动化继续执行其他操作，并且在操作超时的情况下不影响自动化的整体状态。在此示例中，您可以使用 timeoutSeconds 属性指定操作超时之前等待的时间长度。然后，您可以指定存在超时的情况下自动化应转到的操作（即步骤）。使用 onFailure 属性的格式 step:*step name* 指定值，而不是指定默认值 Abort。默认情况下，如果操作超时，在自动化执行状态将为 Timed Out。要防止超时影响自动化执行状态，请为 false 属性指定 isCritical。

以下示例演示如何为此情况中描述的操作定义共享属性。

YAML

```
- name: verifyImageAvailability
  action: 'aws:waitForAwsResourceProperty'
  timeoutSeconds: 600
  isCritical: false
  onFailure: 'step:getCurrentImageState'
  inputs:
    Service: ec2
    Api: DescribeImages
    ImageIds:
      - '{{ createImage.newImageId }}'
    PropertySelector: '$.Images[0].State'
    DesiredValues:
```

```
      - available
  nextStep: copyImage
```

JSON

```
{
    "name": "verifyImageAvailability",
    "action": "aws:waitForAwsResourceProperty",
    "timeoutSeconds": 600,
    "isCritical": false,
    "onFailure": "step:getCurrentImageState",
    "inputs": {
        "Service": "ec2",
        "Api": "DescribeImages",
        "ImageIds": [
            "{{ createImage.newImageId }}"
        ],
        "PropertySelector": "$.Images[0].State",
        "DesiredValues": [
            "available"
        ]
    },
    "nextStep": "copyImage"
}
```

有关所有自动化操作共享的属性的更多信息，请参阅 所有操作共享的属性。

Systems Manager 自动化运行手册参考

为了帮助您快速入门，AWS Systems Manager 提供了预定义运行手册。这些运行手册由 Amazon Web Services、AWS 支持 和 AWS Config 维护。运行手册参考描述了 Systems Manager、支持 和 AWS Config 提供的每个预定义运行手册。有关更多信息，请参阅 Systems Manager 自动化运行手册参考。

教程

以下教程可帮助您使用 AWS Systems Manager Automation 来解决常见应用场景的问题。这些教程演示了如何将自己的运行手册、Automation 提供的预定义运行手册以及别的 Systems Manager 工具与其他 AWS 服务一起使用。

目录

- 更新 AMIs
  - 更新 Linux AMI
  - 更新 Linux AMI（AWS CLI）
  - 更新 Windows Server AMI
  - 使用 Automation、AWS Lambda 和 Parameter Store 来更新黄金 AMI
    - 任务 1：在 Systems Manager Parameter Store 中创建一个参数
    - 任务 2：为 AWS Lambda 创建 IAM 角色
    - 任务 3：创建 AWS Lambda 函数
    - 任务 4：创建运行手册并修补 AMI
  - 使用 Automation 和 Jenkins 更新 AMIs
  - 更新自动扩缩组的 AMIs
    - 创建 PatchAMIAndUpdateASG 运行手册
- 使用 AWS 支持 自助服务运行手册
  - 在无法访问的实例上运行 EC2Rescue 工具
    - 工作方式
    - 开始前的准备工作
      - 向 AWSSupport-EC2Rescue 授予在您的实例上执行操作的权限
        - 使用 IAM policy 授予权限
        - 使用 AWS CloudFormation 模板授予权限
    - 运行自动化
  - 在 EC2 实例上重置密码和 SSH 密钥
    - 工作方式
    - 开始前的准备工作
      - 向 AWSSupport-EC2Rescue 授予在您的实例上执行操作的权限
        - 使用 IAM policy 授予权限
        - 使用 AWS CloudFormation 模板授予权限
    - 运行自动化
- 使用输入变压器将数据传递到 Automation

更新 AMIs

以下教程说明了如何更新 Amazon Machine Image（AMIs）以包含最新补丁。

主题

- 更新 Linux AMI
- 更新 Linux AMI（AWS CLI）
- 更新 Windows Server AMI
- 使用 Automation、AWS Lambda 和 Parameter Store 来更新黄金 AMI
- 使用 Automation 和 Jenkins 更新 AMIs
- 更新自动扩缩组的 AMIs

更新 Linux AMI

本 Systems Manager Automation 演练向您展示了如何使用控制台或 AWS CLI 和 `AWS-UpdateLinuxAmi` 运行手册，使用您指定的最新软件包补丁更新Linux AMI。Automation 是 AWS Systems Manager 中的一项工具。`AWS-UpdateLinuxAmi` 运行手册也能自动安装其他具体站点相关的软件包和配置。您可以使用该演练更新各种 Linux 分发版，包括 Ubuntu Server、CentOS、RHEL、SLES 或 Amazon Linux AMIs。有关支持的 Linux 版本的完整列表，请参阅 Patch Manager 先决条件。

使用 `AWS-UpdateLinuxAmi` 运行手册可以自动完成映像维护任务，而无需使用 JSON 或 YAML 编写运行手册。您可以使用 `AWS-UpdateLinuxAmi` 运行手册执行以下类型的任务。

- 在 Amazon Linux、Red Hat Enterprise Linux、Ubuntu Server、SUSE Linux Enterprise Server 或 CentOS Amazon Machine Image (AMI) 上升级所有分配程序包和 Amazon 软件。这是默认运行手册行为。
- 在现有映像上安装 AWS Systems Manager SSM Agent 可启用 Systems Manager 的各项工具，例如使用 AWS Systems Manager Run Command 执行远程命令，或者使用 Inventory 收集软件清单。
- 安装其他软件包。

开始前的准备工作

在您开始使用运行手册之前，请先为自动化配置角色和 EventBridge（后者可选）。有关更多信息，请参阅 设置自动化。此演练还要求您指定 AWS Identity and Access Management (IAM) 实例配置文件的名称。有关创建 IAM 实例配置文件的更多信息，请参阅配置 Systems Manager 所需的实例权限。

AWS-UpdateLinuxAmi 运行手册接受以下输入参数。

参数	类型	描述
SourceAmiId	字符串	（必需）源 AMI ID。
IamInstanceProfileName	字符串	（必需）您在配置 Systems Manager 所需的实例权限中创建的 IAM 实例配置文件角色的名称。实例配置文件角色为自动化提供在您实例上执行操作的权限，例如运行命令或启动和停止服务。运行手册仅使用实例配置文件角色的名称。如果您指定 Amazon Resource Name (ARN)，则自动化会失败。
AutomationAssumeRole	字符串	（必需）您在 设置自动化 中创建的 IAM 服务角色的名称。服务角色（也称为担任角色）为自动化提供权限，用于担任您的 IAM 角色和代表您执行操作。例如，在运行手册中运行 aws:createImage 操作时，服务角色允许自动化创建新的 AMI。对于此参数，必须指定完整的 ARN。
TargetAmiName	字符串	（可选）新 AMI 在创建之后的名称。默认名称是系统生成的字符串，其中包括源 AMI ID 以及创建时间和日期。
InstanceType	字符串	(可选) 启动作为工作区主机的实例的类型。实例类型因区域而异。默认类型为 t2.micro。

参数	类型	描述
PreUpdateScript	字符串	(可选) 在应用更新前要运行的脚本的 URL。默认值 (\"none\") 不运行脚本。
PostUpdateScript	字符串	(可选) 在应用软件包更新后要运行的脚本的 URL。默认值 (\"none\") 不运行脚本。
IncludePackages	字符串	(可选) 仅更新这些指定的软件包。默认值 (\"all\") 将应用所有可用的更新。
ExcludePackages	字符串	(可选) 在所有情况下从更新中排除的软件包的名称。默认值 (\"none\") 不排除任何软件包。

自动化步骤

AWS-UpdateLinuxAmi 运行手册默认情况下包括以下自动化操作。

步骤 1：launchInstance（**aws:runInstances** 操作）

此步骤使用 Amazon Elastic Compute Cloud (Amazon EC2) 用户数据和 IAM 实例配置文件角色启动实例。用户数据根据操作系统安装相应的 SSM Agent。安装 SSM Agent 后，即可使用 Systems Manager 工具，例如 Run Command、State Manager和 Inventory。

步骤 2：updateOSSoftware（**aws:runCommand**操作）

此步骤在已启动实例上运行以下命令：

- 从 Amazon S3 下载更新脚本。
- 运行可选的更新前脚本。
- 更新分发软件包和 Amazon 软件。
- 运行可选的更新后脚本。

执行日志存储在 /tmp 文件夹中，供用户以后查看。

如果您希望升级特定软件包集，则可以使用 `IncludePackages` 参数提供列表。在提供时，系统仅尝试更新这些软件包及其依赖项。不执行任何其他更新。默认情况下，如果未指定包含 软件包，则程序将更新所有可用软件包。

如果要在升级中排除特定软件包集，则可以向 `ExcludePackages` 参数提供列表。如果提供，这些软件包保持其当前版本，与任何其他指定的选项无关。默认情况下，在未指定任何排除软件包时，将不排除软件包。

步骤 3：stopInstance（**`aws:changeInstanceState`** 操作）

此步骤停止已更新实例。

步骤 4：createImage（**`aws:createImage`** 操作）

此步骤创建一个新 AMI，带有可将其链接到源 ID 和创建时间的描述性名称。例如：“EC2 自动化在 {{global:DATE_TIME}} 从 {{SourceAmiId}} 生成了 AMI”，其中 DATE_TIME 和 SourceID 表示自动化变量。

步骤 5：terminateInstance（**`aws:changeInstanceState`** 操作）

此步骤通过终止正在运行的实例来清除执行过程。

输出

自动化返回新的 AMI ID 作为输出。

> **Note**
>
> 默认情况下，当自动化运行 `AWS-UpdateLinuxAmi` 运行手册时，系统会在默认 VPC (172.30.0.0/16) 中创建一个临时实例。如果您删除了默认 VPC，会收到以下错误：
> `VPC not defined 400`
> 要解决此问题，您必须复制 `AWS-UpdateLinuxAmi` 运行手册并指定子网 ID。有关更多信息，请参阅 VPC not defined 400。

使用自动化创建经过修补的 AMI (AWS Systems Manager)

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 自动化。
3. 选择执行自动化。

4. 在自动化文档列表中，选择 **AWS-UpdateLinuxAmi**。
5. 在文档详细信息部分，验证文档版本是否设置为运行时的默认版本。
6. 选择下一步。
7. 在执行模式部分中，选择简单执行。
8. 在输入参数部分，输入在开始前的准备工作部分收集的信息。
9. 选择执行。控制台将显示自动化执行的状态。

在自动化完成后，请从更新后的 AMI 启动测试实例以验证更改。

> **Note**
>
> 如果自动化有任何步骤失败，自动化执行页面上会列出有关失败的信息。自动化设计为在成功完成所有任务后终止临时实例。如果步骤失败，系统可能不会终止实例。因此，如果某个步骤失败，请手动终止临时实例。

更新 Linux AMI（AWS CLI）

此 AWS Systems Manager 自动化演练介绍如何使用 AWS Command Line Interface (AWS CLI) 和 Systems Manager `AWS-UpdateLinuxAmi` 运行手册通过指定的最新版本软件包自动修补 Linux Amazon Machine Image (AMI)。Automation 是 AWS Systems Manager 中的一项工具。`AWS-UpdateLinuxAmi` 运行手册也能自动安装其他具体站点相关的软件包和配置。您可以使用该演练更新各种 Linux 分发版，包括 Ubuntu Server、CentOS、RHEL、SLES 或 Amazon Linux AMIs。有关支持的 Linux 版本的完整列表，请参阅 Patch Manager 先决条件。

使用 `AWS-UpdateLinuxAmi` 运行手册可以自动完成映像维护任务，而无需使用 JSON 或 YAML 编写运行手册。您可以使用 `AWS-UpdateLinuxAmi` 运行手册执行以下类型的任务。

- 在 Amazon Linux、Red Hat Enterprise Linux、Ubuntu Server、SLES 或 CentOS Amazon Machine Image（AMI）上升级所有分发程序包和 Amazon 软件。这是默认运行手册行为。
- 在现有映像上安装 AWS Systems Manager SSM Agent 以启用 Systems Manager 功能，例如使用 AWS Systems Manager Run Command 的远程命令执行，或者使用 Inventory 的软件清单收集。
- 安装其他软件包。

开始前的准备工作

在您开始使用运行手册之前，请先为自动化配置角色和 EventBridge（后者可选）。有关更多信息，请参阅 设置自动化。此演练还要求您指定 AWS Identity and Access Management (IAM) 实例配置文件的名称。有关创建 IAM 实例配置文件的更多信息，请参阅配置 Systems Manager 所需的实例权限。

`AWS-UpdateLinuxAmi` 运行手册接受以下输入参数。

参数	类型	描述
SourceAmiId	字符串	（必需）源 AMI ID。您可以使用 AWS Systems Manager Parameter Store 公有参数自动引用最新的 Linux Amazon EC2 AMI ID。有关更多信息，请参阅使用 AWS Systems Manager Parameter Store 查询最新 Amazon Linux AMI ID。
IamInstanceProfileName	字符串	（必需）您在配置 Systems Manager 所需的实例权限中创建的 IAM 实例配置文件角色的名称。实例配置文件角色为自动化提供在您实例上执行操作的权限，例如运行命令或启动和停止服务。运行手册仅使用实例配置文件角色的名称。
AutomationAssumeRole	字符串	（必需）您在 设置自动化 中创建的 IAM 服务角色的名称。服务角色（也称为担任角色）为自动化提供权限，用于担任您的 IAM 角色和代表您执行操作。例如，在运行手册中运行 `aws:createImage` 操作时，服务角色允许自动化创建新的 AMI。对于此参数，必须指定完整的 ARN。

参数	类型	描述
TargetAmiName	字符串	（可选）新 AMI 在创建之后的名称。默认名称是系统生成的字符串，其中包括源 AMI ID 以及创建时间和日期。
InstanceType	字符串	(可选) 启动作为工作区主机的实例的类型。实例类型因区域而异。默认类型为 t2.micro。
PreUpdateScript	字符串	(可选) 在应用更新前要运行的脚本的 URL。默认值 (\"none\") 不运行脚本。
PostUpdateScript	字符串	(可选) 在应用软件包更新后要运行的脚本的 URL。默认值 (\"none\") 不运行脚本。
IncludePackages	字符串	(可选) 仅更新这些指定的软件包。默认值 (\"all\") 将应用所有可用的更新。
ExcludePackages	字符串	(可选) 在所有情况下从更新中排除的软件包的名称。默认值 (\"none\") 不排除任何软件包。

自动化步骤

默认情况下，AWS-UpdateLinuxAmi 运行手册包括以下步骤。

步骤 1：launchInstance（**aws:runInstances** 操作）

此步骤使用 Amazon Elastic Compute Cloud (Amazon EC2) 用户数据和 IAM 实例配置文件角色启动实例。用户数据根据操作系统安装相应的 SSM Agent。安装 SSM Agent 后，即可使用 Systems Manager 工具，例如 Run Command、State Manager和 Inventory。

步骤 2：updateOSSoftware（**aws:runCommand**操作）

此步骤在已启动实例上运行以下命令：

- 从 Amazon Simple Storage Service (Amazon S3) 下载更新脚本。
- 运行可选的更新前脚本。
- 更新分发软件包和 Amazon 软件。
- 运行可选的更新后脚本。

执行日志存储在 /tmp 文件夹中，供用户以后查看。

如果您希望升级特定软件包集，则可以使用 `IncludePackages` 参数提供列表。在提供时，系统仅尝试更新这些软件包及其依赖项。不执行任何其他更新。默认情况下，如果未指定包含 软件包，则程序将更新所有可用软件包。

如果要在升级中排除特定软件包集，则可以向 `ExcludePackages` 参数提供列表。如果提供，这些软件包保持其当前版本，与任何其他指定的选项无关。默认情况下，在未指定任何排除软件包时，将不排除软件包。

步骤 3：stopInstance（**`aws:changeInstanceState`** 操作）

此步骤停止已更新实例。

步骤 4：createImage（**`aws:createImage`** 操作）

此步骤创建一个新 AMI，带有可将其链接到源 ID 和创建时间的描述性名称。例如：“EC2 自动化在 {{global:DATE_TIME}} 从 {{SourceAmiId}} 生成了 AMI”，其中 DATE_TIME 和 SourceID 表示自动化变量。

步骤 5：terminateInstance（**`aws:changeInstanceState`** 操作）

此步骤通过终止正在运行的实例来清除执行过程。

输出

自动化返回新的 AMI ID 作为输出。

> **Note**
>
> 默认情况下，当自动化运行 `AWS-UpdateLinuxAmi` 运行手册时，系统会在默认 VPC (172.30.0.0/16) 中创建一个临时实例。如果您删除了默认 VPC，会收到以下错误：
> `VPC not defined 400`
> 要解决此问题，您必须复制 `AWS-UpdateLinuxAmi` 运行手册并指定子网 ID。有关更多信息，请参阅 VPC not defined 400。

使用自动化创建经过修补的 AMI

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 运行以下命令来运行 AWS-UpdateLinuxAmi 运行手册。将每个*#######*替换为您自己的信息。

```
aws ssm start-automation-execution \
    --document-name "AWS-UpdateLinuxAmi" \
    --parameters \
    SourceAmiId=AMI ID, \
    IamInstanceProfileName=IAM instance profile, \
    AutomationAssumeRole='arn:aws:iam::
{{global:ACCOUNT_ID}}:role/AutomationServiceRole'
```

   该命令将会返回执行 ID。请将该 ID 复制到剪贴板。您将使用该 ID 查看自动化的状态。

```
{
    "AutomationExecutionId": "automation execution ID"
}
```

3. 要使用 AWS CLI 查看自动化，请运行以下命令：

```
aws ssm describe-automation-executions
```

4. 运行以下命令以查看有关运行手册进程的详细信息。将 *automation execution ID* 替换为您自己的信息。

```
aws ssm get-automation-execution --automation-execution-id automation execution ID
```

   更新过程可能需要 30 分钟或者更久。

> **Note**
>
> 您也可以在控制台中监控自动化的状态。在列表中，选择您刚才运行的自动化，然后选择步骤选项卡。该选项卡将显示自动化操作的状态。

在自动化完成后，请从更新后的 AMI 启动测试实例以验证更改。

> **Note**
>
> 如果自动化有任何步骤失败，自动化执行页面上会列出有关失败的信息。自动化设计为在成功完成所有任务后终止临时实例。如果步骤失败，系统可能不会终止实例。因此，如果某个步骤失败，请手动终止临时实例。

更新 Windows Server AMI

使用 `AWS-UpdateWindowsAmi` 运行手册可以在 Amazon Windows Amazon Machine Image (AMI) 上自动完成映像维护任务，而无需使用 JSON 或 YAML 编写工作流程。本运行手册在 Windows Server 2008 R2 或更高版本中受支持。您可以使用 `AWS-UpdateWindowsAmi` 运行手册执行以下类型的任务。

- 安装所有 Windows 更新并升级 Amazon 软件 (默认行为)。
- 安装特定 Windows 更新并升级 Amazon 软件。
- 使用您的脚本自定义 AMI。

开始前的准备工作

在开始使用运行手册之前，为自动化配置角色以添加 `iam:PassRole` 策略，此策略引用要授予其访问权限的实例配置文件的 ARN。也可选择为 Automation（AWS Systems Manager 中的一项工具）配置 Amazon EventBridge。有关更多信息，请参阅 设置自动化。此演练还要求您指定 AWS Identity and Access Management (IAM) 实例配置文件的名称。有关创建 IAM 实例配置文件的更多信息，请参阅配置 Systems Manager 所需的实例权限。

> **Note**
>
> AWS Systems Manager SSM Agent 的更新通常会在不同时间向不同区域推广。自定义或更新 AMI 时，请仅使用为您工作所在的区域发布的源 AMI。这将确保您使用该区域发布的最新 SSM Agent并避免兼容性问题。

`AWS-UpdateWindowsAmi` 运行手册接受以下输入参数。

参数	类型	描述
SourceAmiId	字符串	（必需）源 AMI ID。您可以使用 Systems Manager Parameter Store 公有参数自动引用最新的 Windows Server AMI ID。有关更多信息，请参阅使用 AWS Systems ManagerParameter Store 查询最新的 Windows AMI ID。
SubnetId	字符串	（可选）要在其中启动临时实例的子网。如果您已删除默认 VPC，则必须为此参数指定值。
IamInstanceProfileName	字符串	（必需）您在配置 Systems Manager 所需的实例权限中创建的 IAM 实例配置文件角色的名称。实例配置文件角色为自动化提供在您实例上执行操作的权限，例如运行命令或启动和停止服务。运行手册仅使用实例配置文件角色的名称。
AutomationAssumeRole	字符串	（必需）您在 设置自动化 中创建的 IAM 服务角色的名称。服务角色（也称为担任角色）为自动化提供权限，用于担任您的 IAM 角色和代表您执行操作。例如，在运行手册中运行 `aws:createImage` 操作时，服务角色允许自动化创建新的 AMI。对于此参数，必须指定完整的 ARN。

参数	类型	描述
TargetAmiName	字符串	（可选）新 AMI 在创建之后的名称。默认名称是系统生成的字符串，其中包括源 AMI ID 以及创建时间和日期。
InstanceType	字符串	(可选) 启动作为工作区主机的实例的类型。实例类型因区域而异。默认类型为 t2.medium。
PreUpdateScript	字符串	(可选) 要在更新 AMI 之前运行的脚本。在运行手册中或在运行时输入脚本作为参数。
PostUpdateScript	字符串	（可选）要在更新 AMI 之后运行的脚本。在运行手册中或在运行时输入脚本作为参数。
IncludeKbs	字符串	(可选) 指定一个或多个要包括的 Microsoft 知识库 (KB) 文章 ID。可以使用逗号分隔值安装多个 ID。有效格式：KB9876543 或 9876543。
ExcludeKbs	字符串	(可选) 指定一个或多个要排除的 Microsoft 知识库 (KB) 文章 ID。可以使用逗号分隔值排除多个 ID。有效格式：KB9876543 或 9876543。

参数	类型	描述
类别	字符串	(可选) 指定一个或多个更新类别。可以使用逗号分隔值筛选类别。选项：关键更新、安全更新、定义更新、Update Rollup、Service Pack、工具、更新或驱动程序。有效格式包括单个条目，例如：关键更新。或者，可以指定逗号分隔列表：关键更新,安全更新,定义更新。
SeverityLevels	字符串	(可选) 指定一个或多个与更新关联的 MSRC 严重性级别。可以使用逗号分隔值筛选严重性级别。选项：关键、重要、低、中或未指定。有效格式包括单个条目，例如：关键。或者，可以指定逗号分隔列表：关键,重要,低。

自动化步骤

默认情况下，AWS-UpdateWindowsAmi 运行手册包括以下步骤。

步骤 1：launchInstance（**aws:runInstances** 操作）

此步骤以从指定的 SourceAmiID 启动一个具有 IAM 实例配置文件角色的实例。

步骤 2：runPreUpdateScript（**aws:runCommand** 操作）

此步骤可让您以字符串形式指定一个在安装更新前运行的脚本。

步骤 3：更新 EC2Config（**aws:runCommand** 操作）

此步骤使用 AWS-InstallPowerShellModule 运行手册下载 AWS 共有 PowerShell 模块。Systems Manager 使用 SHA-256 哈希验证模块的完整性。然后，Systems Manager 将检查操作系统，以确定是更新 EC2Config 还是 EC2Launch。EC2Config 通过 Windows Server 2012 R2 在 Windows Server 2008 R2 上运行。EC2Launch 在 Windows Server 2016 上运行。

步骤 4：updateSSMAgent（**aws:runCommand**操作）

此步骤通过使用 AWS-UpdateSSMAgent 运行手册更新 SSM Agent。

步骤 5：updateAWSPVDriver（**aws:runCommand**操作）

此步骤通过使用 AWS-ConfigureAWSPackage 运行手册更新 AWS PV 驱动程序。

步骤 6：updateAwsEnaNetworkDriver（**aws:runCommand**操作）

此步骤通过使用 AWS-ConfigureAWSPackage 运行手册更新 AWS ENA 网络驱动程序。

步骤 7：installWindowsUpdates（**aws:runCommand** 操作）

此步骤使用 AWS-InstallWindowsUpdates 运行手册安装 Windows 更新。默认情况下，Systems Manager 搜索并安装任何缺失的更新。可以通过指定下列参数之一更改默认行为：IncludeKbs、ExcludeKbs、Categories 或 SeverityLevels。

步骤 8：runPostUpdateScript（**aws:runCommand** 操作)

此步骤可让您以字符串形式指定一个在安装更新后运行的脚本。

步骤 9：runSysprepGeneralize（**aws:runCommand** 操作)

此步骤使用 AWS-InstallPowerShellModule 运行手册下载 AWS 共有 PowerShell 模块。Systems Manager 使用 SHA-256 哈希验证模块的完整性。然后 Systems Manager 使用 AWS 支持的方法针对 EC2Launch (Windows Server 2016) 或 EC2Config（Windows Server 2008 R2 到 2012 R2）运行 sysprep。

步骤 10：stopInstance（**aws:changeInstanceState** 操作）

此步骤停止已更新实例。

步骤 11：createImage（**aws:createImage** 操作）

此步骤创建一个新 AMI，带有可将其链接到源 ID 和创建时间的描述性名称。例如：“EC2自动化在 {{global:DATE_TIME}} 从 {{SourceAmiId}} 生成了 AMI”，其中 DATE_TIME 和 SourceID 表示自动化变量。

步骤 12：TerminateInstance（**aws:changeInstanceState** 操作）

此步骤通过终止正在运行的实例来清除自动化。

输出

此部分可让您将各个步骤的输出或任何参数的值指定为自动化输出。默认情况下，输出是由执行创建的已更新 Windows AMI 的 ID。

> **Note**
>
> 默认情况下，当自动化运行 AWS-UpdateWindowsAmi 运行手册并创建一个临时实例时，系统会使用默认 VPC (172.30.0.0/16)。如果您删除了默认 VPC，会收到以下错误：
> VPC not defined 400
> 要解决此问题，您必须复制 AWS-UpdateWindowsAmi 运行手册并指定子网 ID。有关更多信息，请参阅 VPC not defined 400。

使用自动化创建经过修补的 Windows AMI

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 运行以下命令来运行 AWS-UpdateWindowsAmi 运行手册。将每个*#######*替换为您自己的信息。以下示例命令使用的是最新的 Amazon EC2 AMI，以最大限度减少需应用的补丁数量。如果您多次运行此命令，则必须为 targetAMIname 指定唯一的值。AMI 名称必须唯一。

   ```
   aws ssm start-automation-execution \
       --document-name="AWS-UpdateWindowsAmi" \
       --parameters SourceAmiId='AMI ID',IamInstanceProfileName='IAM
    instance profile',AutomationAssumeRole='arn:aws:iam::
   {{global:ACCOUNT_ID}}:role/AutomationServiceRole'
   ```

   该命令将会返回执行 ID。请将该 ID 复制到剪贴板。您将使用该 ID 查看自动化的状态。

   ```
   {
       "AutomationExecutionId": "automation execution ID"
   }
   ```

3. 要使用 AWS CLI 查看自动化，请运行以下命令：

   ```
   aws ssm describe-automation-executions
   ```

4. 要查看有关自动化进程的详细信息，请运行以下命令。

   ```
   aws ssm get-automation-execution
       --automation-execution-id automation execution ID
   ```

> **Note**
>
> 根据应用的补丁数量，在该示例自动化中运行的 Windows 修补过程可能需要 30 分钟或更长时间才能完成。

使用 Automation、AWS Lambda 和 Parameter Store 来更新黄金 AMI

以下示例使用组织维护和定期修补自己专有 AMIs 的模型，而非构建自 Amazon Elastic Compute Cloud (Amazon EC2) AMIs。

以下过程说明了如何将操作系统（OS）补丁自动应用到已被视为最新的 AMI 或最新 AMI。在示例中，参数 `SourceAmiId` 的默认值由名为 `latestAmi` 的 AWS Systems Manager Parameter Store 参数定义。`latestAmi` 的值由调用的 AWS Lambda 函数在自动化结束时更新。由于采用了此自动化流程，修补 AMIs 所需的时间和工作量可最大限度减少，因为修补操作始终都应用到最新的 AMI。Parameter Store 和 Automation 都是 AWS Systems Manager 的工具。

开始前的准备工作

配置自动化角色以及（可选）为自动化配置 Amazon EventBridge。有关更多信息，请参阅 设置自动化。

内容

- 任务 1：在 Systems Manager Parameter Store 中创建一个参数
- 任务 2：为 AWS Lambda 创建 IAM 角色
- 任务 3：创建 AWS Lambda 函数
- 任务 4：创建运行手册并修补 AMI

任务 1：在 Systems Manager Parameter Store 中创建一个参数

在 Parameter Store 中创建一个使用以下信息的字符串参数：

- 名称：`latestAmi`。
- 值：AMI ID。例如： `ami-188d6e0e`。

有关如何创建 Parameter Store 字符串参数的信息，请参阅 在 Systems Manager 中创建 Parameter Store 参数。

任务 2：为 AWS Lambda 创建 IAM 角色

使用以下过程为 AWS Lambda 创建 IAM 服务角色。这些策略授予 Lambda 权限，以便使用 Lambda 函数和 Systems Manager 来更新 `latestAmi` 参数的值。

为 Lambda 创建 IAM 服务角色

1. 登录 AWS Management Console，然后通过以下网址打开 IAM 控制台：https://console.aws.amazon.com/iam/。
2. 在导航窗格中选择策略，然后选择创建策略。
3. 选择 JSON 选项卡。
4. 将默认内容替换为以下策略。将每个*#######*替换为您自己的信息。

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "logs:CreateLogGroup",
            "Resource": "arn:aws:logs:us-east-1:111122223333:*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:us-east-1:111122223333:log-group:/aws/
lambda/function name:*"
            ]
        }
    ]
}
```

5. 选择下一步：标签。
6. （可选）添加一个或多个标签键值对，以组织、跟踪或控制此策略的访问权限。
7. 选择下一步：审核。

8. 在查看策略页面上，对于名称，输入内联策略的名称，例如 **amiLambda**。
9. 选择创建策略。
10. 重复步骤 2 和 3。
11. 粘贴以下策略。将每个*#######*替换为您自己的信息。

    JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ssm:PutParameter",
            "Resource": "arn:aws:ssm:us-
east-1:111122223333:parameter/latestAmi"
        },
        {
            "Effect": "Allow",
            "Action": "ssm:DescribeParameters",
            "Resource": "*"
        }
    ]
}
```

12. 选择下一步：标签。
13. （可选）添加一个或多个标签键值对，以组织、跟踪或控制此策略的访问权限。
14. 选择下一步：审核。
15. 在查看策略页面上，对于名称，输入内联策略的名称，例如 **amiParameter**。
16. 选择创建策略。
17. 在导航窗格中，选择 Roles（角色），然后选择 Create role（创建角色）。
18. 直接在应用场景下，选择 Lambda，然后选择下一步。
19. 在添加权限页面上，使用搜索字段查找之前创建的两个策略。
20. 选中策略旁边的复选框，然后选择下一步。
21. 对于角色名称，请输入新角色的名称，如 **lambda-ssm-role**或所需其他名称。

> **Note**
>
> 由于多个实体可能引用该角色，因此创建角色后无法更改角色的名称。

22. （可选）添加一个或多个标签键值对，以组织、追踪或控制对此角色的访问，然后选择创建角色。

任务 3：创建 AWS Lambda 函数

使用以下过程创建 Lambda 函数，该函数自动更新 `latestAmi` 参数的值。

创建 Lambda 函数

1. 通过以下网址登录 AWS Management Console 并打开 AWS Lambda 控制台：https://console.aws.amazon.com/lambda/。
2. 选择创建函数。
3. 在创建函数页面上，选择从头开始创作。
4. 对于函数名称，请输入 **`Automation-UpdateSsmParam`**。
5. 对于运行时系统，选择 Python 3.8。
6. 对于架构，选择 Lambda 用于运行该函数的计算机处理器类型，即 x86_64 或 arm64，
7. 在权限部分中，展开更改默认执行角色。
8. 选择使用现有角色，然后为您在任务 2 中创建的 Lambda 选择服务角色。
9. 选择创建函数。
10. 在代码源区域中的 lambda_function 选项卡上，删除该字段中的预填充代码，然后粘贴以下代码示例。

```
from __future__ import print_function

import json
import boto3

print('Loading function')


#Updates an SSM parameter
#Expects parameterName, parameterValue
def lambda_handler(event, context):
    print("Received event: " + json.dumps(event, indent=2))
```

```
   # get SSM client
   client = boto3.client('ssm')

   #confirm  parameter exists before updating it
   response = client.describe_parameters(
      Filters=[
         {
          'Key': 'Name',
          'Values': [ event['parameterName'] ]
         },
       ]
   )

   if not response['Parameters']:
       print('No such parameter')
       return 'SSM parameter not found.'

   #if parameter has a Description field, update it PLUS the Value
   if 'Description' in response['Parameters'][0]:
       description = response['Parameters'][0]['Description']

       response = client.put_parameter(
         Name=event['parameterName'],
         Value=event['parameterValue'],
         Description=description,
         Type='String',
         Overwrite=True
       )

   #otherwise just update Value
   else:
       response = client.put_parameter(
         Name=event['parameterName'],
         Value=event['parameterValue'],
         Type='String',
         Overwrite=True
       )

   responseString = 'Updated parameter %s with value %s.' %
(event['parameterName'], event['parameterValue'])

   return responseString
```

11. 依次选择文件、保存。
12. 要测试 Lambda 函数，请从测试菜单中选择配置测试事件。
13. 对于事件名称，输入测试事件的名称，如 **MyTestEvent**。
14. 使用以下 JSON 替换现有文本。将 *AMI ID* 替换为您自己的信息来设置 `latestAmi` 参数值。

```
{
   "parameterName":"latestAmi",
   "parameterValue":"AMI ID"
}
```

15. 选择保存。
16. 选择测试以测试该函数。在执行结果选项卡上，应将状态报告为已成功，以及有关更新的其他详细信息。

任务 4：创建运行手册并修补 AMI

使用以下过程创建并运行运行手册，该文档修补您为 latestAmi 参数指定的 AMI。在自动化完成后，使用新修补的 AMI 的 ID 更新 latestAmi 的值。后续自动化使用先前执行创建的 AMI。

创建运行手册并运行

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择文档。
3. 对于创建文档，选择自动化。
4. 对于名称，请输入 **UpdateMyLatestWindowsAmi**。
5. 选择编辑器选项卡，然后选择编辑。
6. 当系统提示时，选择确定。
7. 在文档编辑器字段中，将默认内容替换为以下 YAML 示例运行手册内容。

```
---
description: Systems Manager Automation Demo - Patch AMI and Update ASG
schemaVersion: '0.3'
assumeRole: '{{ AutomationAssumeRole }}'
parameters:
  AutomationAssumeRole:
    type: String
```

```
    description: '(Required) The ARN of the role that allows Automation to perform
  the actions on your behalf. If no role is specified, Systems Manager Automation
  uses your IAM permissions to execute this document.'
    default: ''
  SourceAMI:
    type: String
    description: The ID of the AMI you want to patch.
    default: '{{ ssm:latestAmi }}'
  SubnetId:
    type: String
    description: The ID of the subnet where the instance from the SourceAMI
  parameter is launched.
  SecurityGroupIds:
    type: StringList
    description: The IDs of the security groups to associate with the instance
  that's launched from the SourceAMI parameter.
  NewAMI:
    type: String
    description: The name of of newly patched AMI.
    default: 'patchedAMI-{{global:DATE_TIME}}'
  InstanceProfile:
    type: String
    description: The name of the IAM instance profile you want the source instance
  to use.
  SnapshotId:
    type: String
    description: (Optional) The snapshot ID to use to retrieve a patch baseline
  snapshot.
    default: ''
  RebootOption:
    type: String
    description: '(Optional) Reboot behavior after a patch Install operation. If
  you choose NoReboot and patches are installed, the instance is marked as non-
compliant until a subsequent reboot and scan.'
    allowedValues:
      - NoReboot
      - RebootIfNeeded
    default: RebootIfNeeded
  Operation:
    type: String
    description: (Optional) The update or configuration to perform on the instance.
  The system checks if patches specified in the patch baseline are installed on the
  instance. The install operation installs patches missing from the baseline.
    allowedValues:
```

```
      - Install
      - Scan
    default: Install
mainSteps:
  - name: startInstances
    action: 'aws:runInstances'
    timeoutSeconds: 1200
    maxAttempts: 1
    onFailure: Abort
    inputs:
      ImageId: '{{ SourceAMI }}'
      InstanceType: m5.large
      MinInstanceCount: 1
      MaxInstanceCount: 1
      IamInstanceProfileName: '{{ InstanceProfile }}'
      SubnetId: '{{ SubnetId }}'
      SecurityGroupIds: '{{ SecurityGroupIds }}'
  - name: verifyInstanceManaged
    action: 'aws:waitForAwsResourceProperty'
    timeoutSeconds: 600
    inputs:
      Service: ssm
      Api: DescribeInstanceInformation
      InstanceInformationFilterList:
        - key: InstanceIds
          valueSet:
            - '{{ startInstances.InstanceIds }}'
      PropertySelector: '$.InstanceInformationList[0].PingStatus'
      DesiredValues:
        - Online
    onFailure: 'step:terminateInstance'
  - name: installPatches
    action: 'aws:runCommand'
    timeoutSeconds: 7200
    onFailure: Abort
    inputs:
      DocumentName: AWS-RunPatchBaseline
      Parameters:
        SnapshotId: '{{SnapshotId}}'
        RebootOption: '{{RebootOption}}'
        Operation: '{{Operation}}'
      InstanceIds:
        - '{{ startInstances.InstanceIds }}'
  - name: stopInstance
```

```
    action: 'aws:changeInstanceState'
    maxAttempts: 1
    onFailure: Continue
    inputs:
      InstanceIds:
        - '{{ startInstances.InstanceIds }}'
      DesiredState: stopped
  - name: createImage
    action: 'aws:createImage'
    maxAttempts: 1
    onFailure: Continue
    inputs:
      InstanceId: '{{ startInstances.InstanceIds }}'
      ImageName: '{{ NewAMI }}'
      NoReboot: false
      ImageDescription: Patched AMI created by Automation
  - name: terminateInstance
    action: 'aws:changeInstanceState'
    maxAttempts: 1
    onFailure: Continue
    inputs:
      InstanceIds:
        - '{{ startInstances.InstanceIds }}'
      DesiredState: terminated
  - name: updateSsmParam
    action: aws:invokeLambdaFunction
    timeoutSeconds: 1200
    maxAttempts: 1
    onFailure: Abort
    inputs:
        FunctionName: Automation-UpdateSsmParam
        Payload: '{"parameterName":"latestAmi",
 "parameterValue":"{{createImage.ImageId}}"}'
outputs:
- createImage.ImageId
```

8. 选择创建自动化。
9. 在导航窗格中，选择自动化，然后选择执行自动化。
10. 在 Choose document（选择文档）页面上，选择 Owned by me（我拥有的）选项卡。
11. 搜索 UpdateMyLatestWindowsAmi 运行手册，然后选择 UpdateMyLatestWindowsAmi 卡中的按钮。
12. 选择下一步。

13. 选择简单执行。
14. 指定输入参数的值。
15. 选择执行。
16. 自动化完成后，在导航窗格中选择 Parameter Store，并确认 `latestAmi` 的新值与自动化返回的值匹配。您还可以验证新 AMI ID 与 Amazon EC2 控制台的 AMI 部分中的自动化输出匹配。

使用 Automation 和 Jenkins 更新 AMIs

如果组织在 CI/CD 管道中使用 Jenkins 软件，则可以添加 Automation 作为构建后步骤，用于将应用程序发行版预安装到 Amazon Machine Images（AMIs）中。Automation 是 AWS Systems Manager 中的一项工具。您还可以使用 Jenkins 计划功能来调用 Automation 并创建自己的操作系统（OS）修补计划。

以下示例显示如何从运行在本地或 Amazon Elastic Compute Cloud（Amazon EC2）上的 Jenkins 服务器调用 Automation。对于身份验证，Jenkins 服务器使用 AWS 凭证，该凭证基于您在示例中创建并附加到实例配置文件的 IAM 策略。

> **Note**
>
> 配置实例时，请务必遵循 Jenkins 安全最佳实践。

开始前的准备工作

在配置 Automation 用于 Jenkins 之前，请完成以下任务：

- 完成 使用 Automation、AWS Lambda 和 Parameter Store 来更新黄金 AMI 示例。以下示例使用在该示例中创建的 UpdateMyLatestWindowsAmi 运行手册。
- 为自动化配置 IAM 角色。Systems Manager 需要实例配置文件角色和服务角色 ARN 来处理自动化。有关更多信息，请参阅 设置自动化。

为 Jenkins 服务器创建 IAM 策略

1. 登录 AWS Management Console，然后通过以下网址打开 IAM 控制台：https://console.aws.amazon.com/iam/。
2. 在导航窗格中选择策略，然后选择创建策略。

3. 选择 JSON 选项卡。
4. 将每个*#######*替换为您自己的信息。

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ssm:StartAutomationExecution",
            "Resource": [
                "arn:aws:ssm:us-
east-1:111122223333:document/UpdateMyLatestWindowsAmi",
                "arn:aws:ssm:us-east-1:111122223333:automation-
definition/UpdateMyLatestWindowsAmi:$DEFAULT"
            ]
        }
    ]
}
```

5. 选择查看策略。
6. 在查看策略页面上，对于名称，输入内联策略的名称，例如 **JenkinsPolicy**。
7. 选择创建策略。
8. 在导航窗格中，选择角色。
9. 选择附加到 Jenkins 服务器的实例配置文件。
10. 在权限选项卡中，选择添加权限，然后选择附加策略。
11. 在其他权限策略部分中，输入您在之前步骤中创建的策略名称。例如，JenkinsPolicy。
12. 选中您的策略旁边的复选框，然后选择附加策略。

按照以下过程在 Jenkins 服务器上配置 AWS CLI。

为 Automation 配置 Jenkins 服务器

1. 使用首选浏览器连接到 8080 端口上的 Jenkins 服务器，以便访问管理界面。
2. 输入 `/var/lib/jenkins/secrets/initialAdminPassword` 在中找到的密码。要显示您的密码，请运行以下命令。

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

3. Jenkins 安装脚本会将您引导到自定义 Jenkins 页面。选择安装建议的插件。
4. 安装完成后，依次选择管理员凭证、保存凭证、开始使用 Jenkins。
5. 在左侧导航窗格中，选择管理 Jenkins，然后选择管理插件。
6. 选可用选项卡，然后输入 **`Amazon EC2 plugin`**。
7. 选中 **`Amazon EC2 plugin`** 的复选框，然后选择安装但不重新启动。
8. 安装完成后，选择返回首页。
9. 选择管理 Jenkins，然后选择管理节点和云。
10. 在配置云部分中，选择添加新云，然后选择 Amazon EC2。
11. 在其余字段中输入您的信息。确保选择使用 EC2 实例配置文件获取凭证选项。

按照以下过程配置 Jenkins 项目来调用 Automation。

配置 Jenkins 服务器来调用 Automation

1. 在 Web 浏览器中打开 Jenkins 控制台。
2. 选择要配置用于自动化的项目，然后选择配置。
3. 在构建选项卡上，选择添加构建步骤。
4. 选择执行 shell 或执行 Windows 批命令（具体取决于您的操作系统)。
5. 在 Command（命令）字段中，运行 AWS CLI 命令，如下所示。将每个*#######*替换为您自己的信息。

```
aws ssm start-automation-execution \
        --document-name runbook name \
        --region AWS ## of your source AMI \
        --parameters runbook parameters
```

以下示例命令使用 UpdateMyLatestWindowsAmi 运行手册以及在 使用 Automation、AWS Lambda 和 Parameter Store 来更新黄金 AMI 中创建的 Systems Manager 参数 `latestAmi`：

```
aws ssm start-automation-execution \
        --document-name UpdateMyLatestWindowsAmi \
        --parameters \
            "sourceAMIid='{{ssm:latestAmi}}'"
```

```
        --region region
```

在 Jenkins 中，命令类似于以下屏幕截图中的示例。



6. 在 Jenkins 项目中，选择立即构建。Jenkins 会返回类似于以下示例的输出。



更新自动扩缩组的 AMIs

以下示例使用新修补的 AMI 更新自动扩缩组。此方法确保新映像自动可供使用 Auto Scaling 组的不同计算环境使用。

此示例中自动化的最后一步使用 Python 函数创建将使用新修补的 AMI 的新启动模板。然后更新自动扩缩组以使用新的启动模板。在此类型的 Auto Scaling 方案中，用户可以终止 Auto Scaling 组中的现有实例以强制启动使用新映像的实例。或者，用户可以等待以允许缩减或扩展事件正常启动较新的实例。

开始前的准备工作

在开始本示例之前，请完成以下任务。

- 为 Automation（AWS Systems Manager 中的一项工具）配置 IAM 角色。Systems Manager 需要实例配置文件角色和服务角色 ARN 来处理自动化。有关更多信息，请参阅 设置自动化。

创建 PatchAMIAndUpdateASG 运行手册

使用以下过程创建 PatchAMIAndUpdateASG 运行手册，它将修补您为 SourceAMI 参数指定的 AMI。运行手册还会更新自动扩缩组以使用最新修补后的 AMI。

创建运行手册并运行

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择文档。
3. 在创建文档下拉菜单中，选择自动化。
4. 在名称字段中，输入 **PatchAMIAndUpdateASG**。
5. 选择 Editor（编辑器）选项卡，然后选择 Edit（编辑）。
6. 出现提示时选择 OK（确定），然后删除 Document editor（文档编辑器）字段中的内容。
7. 在 Document editor（文档编辑器）字段中，粘贴以下 YAML 示例运行手册内容。

```
---
description: Systems Manager Automation Demo - Patch AMI and Update ASG
schemaVersion: '0.3'
assumeRole: '{{ AutomationAssumeRole }}'
parameters:
  AutomationAssumeRole:
    type: String
    description: '(Required) The ARN of the role that allows Automation to perform
 the actions on your behalf. If no role is specified, Systems Manager Automation
 uses your IAM permissions to execute this document.'
    default: ''
  SourceAMI:
    type: String
    description: '(Required) The ID of the AMI you want to patch.'
  SubnetId:
    type: String
    description: '(Required) The ID of the subnet where the instance from the
 SourceAMI parameter is launched.'
  SecurityGroupIds:
    type: StringList
    description: '(Required) The IDs of the security groups to associate with the
 instance launched from the SourceAMI parameter.'
  NewAMI:
    type: String
    description: '(Optional) The name of of newly patched AMI.'
```

```
    default: 'patchedAMI-{{global:DATE_TIME}}'
  TargetASG:
    type: String
    description: '(Required) The name of the Auto Scaling group you want to
 update.'
  InstanceProfile:
    type: String
    description: '(Required) The name of the IAM instance profile you want the
 source instance to use.'
  SnapshotId:
    type: String
    description: (Optional) The snapshot ID to use to retrieve a patch baseline
 snapshot.
    default: ''
  RebootOption:
    type: String
    description: '(Optional) Reboot behavior after a patch Install operation. If
 you choose NoReboot and patches are installed, the instance is marked as non-
compliant until a subsequent reboot and scan.'
    allowedValues:
      - NoReboot
      - RebootIfNeeded
    default: RebootIfNeeded
  Operation:
    type: String
    description: (Optional) The update or configuration to perform on the instance.
 The system checks if patches specified in the patch baseline are installed on the
 instance. The install operation installs patches missing from the baseline.
    allowedValues:
      - Install
      - Scan
    default: Install
mainSteps:
  - name: startInstances
    action: 'aws:runInstances'
    timeoutSeconds: 1200
    maxAttempts: 1
    onFailure: Abort
    inputs:
      ImageId: '{{ SourceAMI }}'
      InstanceType: m5.large
      MinInstanceCount: 1
      MaxInstanceCount: 1
      IamInstanceProfileName: '{{ InstanceProfile }}'
```

```
      SubnetId: '{{ SubnetId }}'
      SecurityGroupIds: '{{ SecurityGroupIds }}'
  - name: verifyInstanceManaged
    action: 'aws:waitForAwsResourceProperty'
    timeoutSeconds: 600
    inputs:
      Service: ssm
      Api: DescribeInstanceInformation
      InstanceInformationFilterList:
        - key: InstanceIds
          valueSet:
            - '{{ startInstances.InstanceIds }}'
      PropertySelector: '$.InstanceInformationList[0].PingStatus'
      DesiredValues:
        - Online
    onFailure: 'step:terminateInstance'
  - name: installPatches
    action: 'aws:runCommand'
    timeoutSeconds: 7200
    onFailure: Abort
    inputs:
      DocumentName: AWS-RunPatchBaseline
      Parameters:
        SnapshotId: '{{SnapshotId}}'
        RebootOption: '{{RebootOption}}'
        Operation: '{{Operation}}'
      InstanceIds:
        - '{{ startInstances.InstanceIds }}'
  - name: stopInstance
    action: 'aws:changeInstanceState'
    maxAttempts: 1
    onFailure: Continue
    inputs:
      InstanceIds:
        - '{{ startInstances.InstanceIds }}'
      DesiredState: stopped
  - name: createImage
    action: 'aws:createImage'
    maxAttempts: 1
    onFailure: Continue
    inputs:
      InstanceId: '{{ startInstances.InstanceIds }}'
      ImageName: '{{ NewAMI }}'
      NoReboot: false
```

```
      ImageDescription: Patched AMI created by Automation
  - name: terminateInstance
    action: 'aws:changeInstanceState'
    maxAttempts: 1
    onFailure: Continue
    inputs:
      InstanceIds:
        - '{{ startInstances.InstanceIds }}'
      DesiredState: terminated
  - name: updateASG
    action: 'aws:executeScript'
    timeoutSeconds: 300
    maxAttempts: 1
    onFailure: Abort
    inputs:
      Runtime: python3.8
      Handler: update_asg
      InputPayload:
        TargetASG: '{{TargetASG}}'
        NewAMI: '{{createImage.ImageId}}'
      Script: |-
        from __future__ import print_function
        import datetime
        import json
        import time
        import boto3

        # create auto scaling and ec2 client
        asg = boto3.client('autoscaling')
        ec2 = boto3.client('ec2')

        def update_asg(event, context):
            print("Received event: " + json.dumps(event, indent=2))

            target_asg = event['TargetASG']
            new_ami = event['NewAMI']

            # get object for the ASG we're going to update, filter by name of
target ASG
            asg_query =
asg.describe_auto_scaling_groups(AutoScalingGroupNames=[target_asg])
            if 'AutoScalingGroups' not in asg_query or not
asg_query['AutoScalingGroups']:
                return 'No ASG found matching the value you specified.'
```

```
            # gets details of an instance from the ASG that we'll use to model the
 new launch template after
            source_instance_id = asg_query.get('AutoScalingGroups')[0]['Instances']
[0]['InstanceId']
            instance_properties = ec2.describe_instances(
                InstanceIds=[source_instance_id]
            )
            source_instance = instance_properties['Reservations'][0]['Instances']
[0]

            # create list of security group IDs
            security_groups = []
            for group in source_instance['SecurityGroups']:
                security_groups.append(group['GroupId'])

            # create a list of dictionary objects for block device mappings
            mappings = []
            for block in source_instance['BlockDeviceMappings']:
                volume_query = ec2.describe_volumes(
                    VolumeIds=[block['Ebs']['VolumeId']]
                )
                volume_details = volume_query['Volumes']
                device_name = block['DeviceName']
                volume_size = volume_details[0]['Size']
                volume_type = volume_details[0]['VolumeType']
                device = {'DeviceName': device_name, 'Ebs': {'VolumeSize':
 volume_size, 'VolumeType': volume_type}}
                mappings.append(device)

            # create new launch template using details returned from instance in
 the ASG and specify the newly patched AMI
            time_stamp = time.time()
            time_stamp_string =
 datetime.datetime.fromtimestamp(time_stamp).strftime('%m-%d-%Y_%H-%M-%S')
            new_template_name = f'{new_ami}_{time_stamp_string}'
            try:
                ec2.create_launch_template(
                    LaunchTemplateName=new_template_name,
                    LaunchTemplateData={
                        'BlockDeviceMappings': mappings,
                        'ImageId': new_ami,
                        'InstanceType': source_instance['InstanceType'],
                        'IamInstanceProfile': {
```

```
                            'Arn': source_instance['IamInstanceProfile']['Arn']
                        },
                        'KeyName': source_instance['KeyName'],
                        'SecurityGroupIds': security_groups
                    }
                )
            except Exception as e:
                return f'Exception caught: {str(e)}'
            else:
                # update ASG to use new launch template
                asg.update_auto_scaling_group(
                    AutoScalingGroupName=target_asg,
                    LaunchTemplate={
                        'LaunchTemplateName': new_template_name
                    }
                )
                return f'Updated ASG {target_asg} with new launch template
 {new_template_name} which uses AMI {new_ami}.'
outputs:
  - createImage.ImageId
```

8. 选择创建自动化。
9. 在导航窗格中，选择自动化，然后选择执行自动化。
10. 在 Choose document（选择文档）页面上，选择 Owned by me（我拥有的）选项卡。
11. 搜索 PatchAMIAndUpdateASG 运行手册，然后选择 PatchAMIAndUpdateASG 卡中的按钮。
12. 选择下一步。
13. 选择简单执行。
14. 指定输入参数的值。确保您指定的 `SubnetId` 和 `SecurityGroupIds` 允许访问公有 Systems Manager 端点或 Systems Manager 的接口端点。
15. 选择执行。
16. 自动化完成后，在 Amazon EC2 控制台中，选择 Auto Scaling（自动扩缩），然后选择 Launch Templates（启动配置）。验证您是否看到了新的启动模板，并且它使用了新的 AMI。
17. 选择 Auto Scaling，然后选择 Auto Scaling 组。验证自动扩缩组使用了新的启动模板。
18. 终止 Auto Scaling 组中的一个或多个实例。替换实例使用新的 AMI 启动。

使用 AWS 支持 自助服务运行手册

本部分介绍如何使用由 AWS 支持 团队创建的自助自动化。这些自动化可帮助您管理 AWS 的资源。

支持自动化工作流

Support Workflows (SAW) 是由 AWS 支持 团队撰写并维护的自动化运行手册。这些运行手册帮助您对有关 AWS 资源的常见问题进行故障排除，主动监控和识别网络问题，收集和分析日志，等等。

SAW 运行手册使用 **AWSSupport** 前缀 例如，AWSSupport-ActivateWindowsWithAmazonLicense。

此外,AWS 企业和业务支持客户还可以访问使用 **AWSPremiumSupport** 前缀的运行手册。例如，AWSPremiumSupport-TroubleshootEC2DiskUsage。

如需了解关于 AWS 支持 的更多信息，请参阅 AWS 支持 入门。

主题

- 在无法访问的实例上运行 EC2Rescue 工具
- 在 EC2 实例上重置密码和 SSH 密钥

在无法访问的实例上运行 EC2Rescue 工具

EC2Rescue 可以帮助您诊断有关 Linux 和 Windows Server 的 Amazon Elastic Compute Cloud (Amazon EC2) 实例的问题并进行故障排除。您可以手动运行工具，如使用适用于 Linux Server 的 EC2Rescue 和使用适用于 Windows Server 的 EC2Rescue。或者，您也可以使用 Systems Manager 自动化和 **AWSSupport-ExecuteEC2Rescue** 运行手册。Automation 是 AWS Systems Manager 中的一项工具。**AWSSupport-ExecuteEC2Rescue** 运行手册旨在全面执行 Systems Manager 的操作、AWS CloudFormation 的操作和 Lambda 功能，从而将使用 EC2Rescue 通常所需的步骤自动化。

您可以使用 **AWSSupport-ExecuteEC2Rescue** 运行手册，对不同类型的操作系统 (OS) 问题进行故障排除和潜在修复。不支持带有加密根卷的实例。有关完整列表，请参阅以下主题：

Windows：请参阅将适用于 Windows Server 的 EC2Rescue 与命令行配合使用中的抢救操作。

Linux 和 macOS：一些适用于 Linux 的 EC2Rescue 模块会检测并尝试修复问题。有关更多信息，请参阅 GitHub 上各个模块的 aws-ec2rescue-linux 文档。

工作方式

如下文所示，使用自动化和 **AWSSupport-ExecuteEC2Rescue** 运行手册对实例进行故障排除：

- 您为无法访问的实例指定 ID 并启动运行手册。
- 系统创建一个临时 VPC，然后运行一系列 Lambda 函数以配置该 VPC。

- 系统在与您的原始实例相同的可用区内为您的临时 VPC 标识一个子网。
- 系统启动一个临时的启用了 SSM 的 帮助程序实例。
- 系统停止您的原始实例并创建备份。然后，它将原始根卷附加到帮助程序实例。
- 系统使用 Run Command 在帮助程序实例上运行 EC2Rescue。EC2Rescue 识别并尝试修复有关已附加的原始根卷的问题。完成后，EC2Rescue 重新将根卷附加回原始实例。
- 系统重启您的原始实例，并终止临时实例。系统也将终止临时 VPC 和在自动化开始时创建的 Lambda 函数。

开始前的准备工作

运行以下自动化之前，请执行以下操作：

- 复制无法访问的实例的实例 ID。您将在过程中指定此 ID。
- (可选) 收集无法访问实例所在可用区中子网的 ID。EC2Rescue 实例将在此子网中创建。如果不指定子网，自动化会在您的 AWS 账户中创建新建一个临时 VPC。验证您的 AWS 账户是否至少有一个可用 VPC。默认情况下，一个区域中可以创建 5 个 VPC。如果您已在该区域中创建 5 个 VPC，自动化将会失败，但不会对您的实例进行更改。有关 Amazon VPC 配额的更多信息，请参阅 Amazon VPC 用户指南中的 VPC 和子网。
- 或者，您可以为自动化创建并指定一个 AWS Identity and Access Management (IAM) 角色。如果您不指定此角色，自动化将在运行自动化的用户的环境中运行。

向 **AWSSupport-EC2Rescue** 授予在您的实例上执行操作的权限

在自动化执行期间，EC2Rescue 需要权限才能在您的实例上执行一系列操作。这些操作调用 AWS Lambda、IAM 和 Amazon EC2 服务，安全地尝试修复实例的问题。如果您在 AWS 账户 和/或 VPC 中具有管理员级权限，您可能能够在不按照本部分中所述配置权限的情况下运行自动化。如果您没有管理员级权限，则您或管理员必须使用以下选项之一配置权限。

- 使用 IAM policy 授予权限
- 使用 AWS CloudFormation 模板授予权限

使用 IAM policy 授予权限

您可以将以下 IAM policy 作为内联策略附加到您的用户、组或角色；也可以创建新的 IAM 托管策略，并将其附加到您的用户、组或角色。有关将内联策略添加到您的用户、组或角色的更多信息，请参阅使用内联策略。有关创建新的托管策略的更多信息，请参阅使用托管策略。

> **Note**
>
> 如果创建新的 IAM 托管式策略，则还必须将 AmazonSSMAutomationRole 托管式策略附加到该策略，以便实例与 Systems Manager API 通信。

适用于 AWSSupport-EC2Rescue 的 IAM policy

将 *account ID* 替换为您自己的信息。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "lambda:InvokeFunction",
                "lambda:DeleteFunction",
                "lambda:GetFunction"
            ],
            "Resource": "arn:aws:lambda:*:111122223333:function:AWSSupport-
EC2Rescue-*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::awssupport-ssm.*/*.template",
                "arn:aws:s3:::awssupport-ssm.*/*.zip"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "iam:CreateRole",
                "iam:CreateInstanceProfile",
                "iam:GetRole",
                "iam:GetInstanceProfile",
```

```
                "iam:PutRolePolicy",
                "iam:DetachRolePolicy",
                "iam:AttachRolePolicy",
                "iam:PassRole",
                "iam:AddRoleToInstanceProfile",
                "iam:RemoveRoleFromInstanceProfile",
                "iam:DeleteRole",
                "iam:DeleteRolePolicy",
                "iam:DeleteInstanceProfile"
            ],
            "Resource": [
                "arn:aws:iam::111122223333:role/AWSSupport-EC2Rescue-*",
                "arn:aws:iam::111122223333:instance-profile/AWSSupport-EC2Rescue-
*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "lambda:CreateFunction",
                "ec2:CreateVpc",
                "ec2:ModifyVpcAttribute",
                "ec2:DeleteVpc",
                "ec2:CreateInternetGateway",
                "ec2:AttachInternetGateway",
                "ec2:DetachInternetGateway",
                "ec2:DeleteInternetGateway",
                "ec2:CreateSubnet",
                "ec2:DeleteSubnet",
                "ec2:CreateRoute",
                "ec2:DeleteRoute",
                "ec2:CreateRouteTable",
                "ec2:AssociateRouteTable",
                "ec2:DisassociateRouteTable",
                "ec2:DeleteRouteTable",
                "ec2:CreateVpcEndpoint",
                "ec2:DeleteVpcEndpoints",
                "ec2:ModifyVpcEndpoint",
                "ec2:Describe*",
                "autoscaling:DescribeAutoScalingInstances"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
```

```
    ]
}
```

使用 AWS CloudFormation 模板授予权限

AWS CloudFormation 使用预配置模板自动化创建 IAM 角色和策略的过程。使用 AWS CloudFormation，通过以下过程为 EC2Rescue 自动化创建所需的 IAM 角色和策略。

为 EC2Rescue 创建所需的 IAM 角色和策略

1. 下载 AWSSupport-EC2RescueRole.zip 文件并将 AWSSupport-EC2RescueRole.json 解压到本地计算机上的目录。
2. 如果您的 AWS 账户是一个特殊分区，请编辑模板以将 ARN 值更改为您的分区的值。

   例如，对于中国地区，将出现的所有 arn:aws 更改为 arn:aws-cn。
3. 登录到 AWS Management Console 并打开 AWS CloudFormation 控制台 https://console.aws.amazon.com/cloudformation。
4. 依次选择创建堆栈和使用新资源（标准）。
5. 在创建堆栈页面上，对于先决条件 - 准备模板，选择模板已就绪。
6. 对于指定模板，选择上传模板文件。
7. 选择选择文件，然后进行浏览并从您解压缩 AWSSupport-EC2RescueRole.json 文件的目录中选择此文件。
8. 选择下一步。
9. 在指定堆栈详细信息页上，对于堆栈名称字段，输入用于标识此堆栈的名称，然后选择下一步。
10. （可选）在标签 区域，将一个或多个标签键名称/值对应用到堆栈。

    标签是您分配给资源的可选元数据。标签可让您按各种标准（如用途、所有者或环境）对资源进行分类。例如，您可能需要对堆栈进行标记，以确定其运行的任务类型、涉及的目标或其他资源的类型以及其运行的环境。
11. 选择 Next（下一步）
12. 在审核页面上，审核堆栈详细信息，然后向下滚动并选择我确认 AWS CloudFormation 可能创建 IAM 资源选项。
13. 选择创建堆栈。

    AWS CloudFormation 将持续几分钟显示 CREATE_IN_PROGRESS 状态。创建堆栈后，状态将变为 CREATE_COMPLETE。您还可以选择刷新图标来检查创建过程的状态。

14. 在堆栈列表中，选择您刚刚创建的堆栈的选项按钮，然后选择输出选项卡。
15. 记下值。这是 AssumeRole 的 ARN。在下一个过程 运行自动化 中运行自动化时指定此 ARN。

运行自动化

> **Important**
>
> 以下自动化将停止无法访问的实例。停止实例可能导致附加的实例存储卷（如果存在）上的数据丢失。如果未关联弹性 IP，停止实例也可能导致公有 IP 更改。

运行 **AWSSupport-ExecuteEC2Rescue** 自动化

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 自动化。
3. 选择执行自动化。
4. 在自动化文档部分中，从列表中选择 Amazon 所拥有。
5. 在运行手册列表中，选择卡片中的 **AWSSupport-ExecuteEC2Rescue**，然后选择下一步。
6. 在执行自动化文档页面上，选择简单执行。
7. 在文档详细信息部分中，验证文档版本 是否设置为最高默认版本。例如，$DEFAULT 或 3 (默认值))。
8. 在输入参数部分中，指定以下参数：
   a. 对于 UnreachableInstanceId，指定无法访问实例的 ID。
   b. （可选）对于 EC2RescueInstanceType，为 EC2Rescue 实例指定实例类型。默认的实例类型为 `t2.medium`。
   c. 对于 AutomationAssumeRole，如果您通过使用本主题前面介绍的 AWS CloudFormation 过程为此自动化创建了角色，则选择您在 AWS CloudFormation 控制台中创建的 AssumeRole 的 ARN。
   d. （可选）如果在排查实例故障时需要收集操作系统级别的日志，请为 LogDestination 指定 S3 存储桶。日志会自动上传到此指定存储桶。
   e. 对于 SubnetId，指定无法访问实例所在可用区中某个现有 VPC 的子网。默认情况下，Systems Manager 会新建一个 VPC，但您可以根据需要指定现有 VPC 中的某个子网。

> **Note**
>
> 如果未看到指定存储桶或子网 ID 的选项，请确认使用的是不是运行手册的最新默认版本。

9. （可选）在标签区域，应用一个或多个标签键名称/值对以帮助识别自动化，例如 `Key=Purpose,Value=EC2Rescue`。
10. 选择执行。

运行手册创建一个备份 AMI，作为自动化的一部分。自动化创建的所有其他资源都会自动删除，但此 AMI 将保留在您的账户中。此 AMI 遵从以下命名约定：

备份 AMI：AWSSupport-EC2Rescue:*UnreachableInstanceId*

在 Amazon EC2 控制台中，可以通过搜索自动化执行 ID 查找此 AMI。

在 EC2 实例上重置密码和 SSH 密钥

您可以使用 `AWSSupport-ResetAccess` 运行手册在适用于 Windows Server 的 Amazon Elastic Compute Cloud（Amazon EC2）实例上自动重新启用本地管理员密码生成，以及在适用于 Linux 的 EC2 实例上生成新的 SSH 密钥。`AWSSupport-ResetAccess` 运行手册旨在执行 AWS Systems Manager 操作、AWS CloudFormation 操作和 AWS Lambda 函数的组合，从而将重置本地管理员密码通常所需的步骤自动化。

您可以使用 Automation（AWS Systems Manager 中的一项工具）搭配 `AWSSupport-ResetAccess` 运行手册来解决以下问题：

Windows

您丢失了 EC2 密钥对：要解决此问题，您可以使用 AWSSupport-ResetAccess 运行手册，从当前实例创建启用了密码的 AMI，从 AMI 启动新实例，然后选择您拥有的密钥对。

您丢失了本地管理员密码：要解决此问题，您可以使用 `AWSSupport-ResetAccess` 运行手册生成可以使用当前 EC2 密钥对解密的一个新密码。

Linux

您丢失了 EC2 密钥对，或者配置了对实例的 SSH 访问但丢失：要解决此问题，您可以使用 `AWSSupport-ResetAccess` 运行手册创建当前实例的新 SSH 密钥，这使您能够重新连接到该实例。

> **Note**
>
> 如果适用于 Windows Server 的 EC2 实例针对 Systems Manager 进行了配置，也可以使用 EC2Rescue 和 AWS Systems Manager Run Command 重置您的本地管理员密码。有关更多信息，请参阅《Amazon EC2 用户指南》中的将 EC2Rescue for Windows Server 与 Systems Manager Run Command 结合使用。

相关信息

《Amazon EC2 用户指南》中的使用 PuTTY 从 Windows 连接到 Linux 实例。

工作方式

使用自动化和 `AWSSupport-ResetAccess` 运行手册进行故障排除的工作原理如下：

- 您指定实例的 ID 并运行运行手册。
- 系统创建一个临时 VPC，然后运行一系列 Lambda 函数以配置该 VPC。
- 系统在与您的原始实例相同的可用区内为您的临时 VPC 标识一个子网。
- 系统启动一个临时的启用了 SSM 的 帮助程序实例。
- 系统停止您的原始实例并创建备份。然后，它将原始根卷附加到帮助程序实例。
- 系统使用 Run Command 在帮助程序实例上运行 EC2Rescue。在 Windows 上，EC2Rescue 通过在附加的原始根卷上使用 EC2Config 或 EC2Launch 为本地管理员启用密码生成。在 Linux 上，EC2Rescue 生成并注入新的 SSH 密钥并将私有密钥加密保存到 Parameter Store 中。完成后，EC2Rescue 重新将根卷附加回原始实例。
- 系统根据您的实例创建新 Amazon Machine Image (AMI)，现在密码生成已启用。您可以使用此 AMI 创建新 EC2 实例，并根据需要关联新密钥对。
- 系统重启您的原始实例，并终止临时实例。系统也将终止临时 VPC 和在自动化开始时创建的 Lambda 函数。
- Windows：您的实例生成一个新密码，您可以使用分配给实例的当前密钥对从 Amazon EC2 控制台对该密码进行解码。

  Linux：您可以使用存储在 Systems Manager Parameter Store 中的 SSH 密钥（格式为 /ec2rl/openssh/***instance ID***/key）通过 SSH 连接到实例。

开始前的准备工作

运行以下自动化之前，请执行以下操作：

- 复制要重置管理员密码的实例的实例 ID。您将在过程中指定此 ID。
- (可选) 收集无法访问实例所在可用区中子网的 ID。EC2Rescue 实例将在此子网中创建。如果不指定子网，自动化会在您的 AWS 账户中创建新建一个临时 VPC。验证您的 AWS 账户是否至少有一个可用 VPC。默认情况下，一个区域中可以创建 5 个 VPC。如果您已在该区域中创建 5 个 VPC，自动化将会失败，但不会对您的实例进行更改。有关 Amazon VPC 配额的更多信息，请参阅 Amazon VPC 用户指南中的 VPC 和子网。
- 或者，您可以为自动化创建并指定一个 AWS Identity and Access Management (IAM) 角色。如果您不指定此角色，自动化将在运行自动化的用户的环境中运行。

向 AWSSupport-EC2Rescue 授予在您的实例上执行操作的权限

在自动化期间，EC2Rescue 需要权限才能在您的实例上执行一系列操作。这些操作调用 AWS Lambda、IAM 和 Amazon EC2 服务，安全地尝试修复实例的问题。如果您在 AWS 账户 和/或 VPC 中具有管理员级权限，您可能能够在不按照本部分中所述配置权限的情况下运行自动化。如果您没有管理员级权限，则您或管理员必须使用以下选项之一配置权限。

- 使用 IAM policy 授予权限
- 使用 AWS CloudFormation 模板授予权限

使用 IAM policy 授予权限

您可以将以下 IAM policy 作为内联策略附加到您的用户、组或角色；也可以创建新的 IAM 托管策略，并将其附加到您的用户、组或角色。有关将内联策略添加到您的用户、组或角色的更多信息，请参阅使用内联策略。有关创建新的托管策略的更多信息，请参阅使用托管策略。

> **Note**
>
> 如果创建新的 IAM 托管式策略，则还必须将 AmazonSSMAutomationRole 托管式策略附加到该策略，以便实例与 Systems Manager API 通信。

AWSSupport-ResetAccess 的 IAM policy

将 *account ID* 替换为您自己的信息。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "lambda:InvokeFunction",
                "lambda:DeleteFunction",
                "lambda:GetFunction"
            ],
            "Resource": "arn:aws:lambda:*:111122223333:function:AWSSupport-
EC2Rescue-*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::awssupport-ssm.*/*.template",
                "arn:aws:s3:::awssupport-ssm.*/*.zip"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "iam:CreateRole",
                "iam:CreateInstanceProfile",
                "iam:GetRole",
                "iam:GetInstanceProfile",
                "iam:PutRolePolicy",
                "iam:DetachRolePolicy",
                "iam:AttachRolePolicy",
                "iam:PassRole",
                "iam:AddRoleToInstanceProfile",
                "iam:RemoveRoleFromInstanceProfile",
                "iam:DeleteRole",
                "iam:DeleteRolePolicy",
                "iam:DeleteInstanceProfile"
            ],
            "Resource": [
```

```
                "arn:aws:iam::111122223333:role/AWSSupport-EC2Rescue-*",
                "arn:aws:iam::111122223333:instance-profile/AWSSupport-EC2Rescue-
*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "lambda:CreateFunction",
                "ec2:CreateVpc",
                "ec2:ModifyVpcAttribute",
                "ec2:DeleteVpc",
                "ec2:CreateInternetGateway",
                "ec2:AttachInternetGateway",
                "ec2:DetachInternetGateway",
                "ec2:DeleteInternetGateway",
                "ec2:CreateSubnet",
                "ec2:DeleteSubnet",
                "ec2:CreateRoute",
                "ec2:DeleteRoute",
                "ec2:CreateRouteTable",
                "ec2:AssociateRouteTable",
                "ec2:DisassociateRouteTable",
                "ec2:DeleteRouteTable",
                "ec2:CreateVpcEndpoint",
                "ec2:DeleteVpcEndpoints",
                "ec2:ModifyVpcEndpoint",
                "ec2:Describe*"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

使用 AWS CloudFormation 模板授予权限

AWS CloudFormation 使用预配置模板自动化创建 IAM 角色和策略的过程。使用 AWS CloudFormation，通过以下过程为 EC2Rescue 自动化创建所需的 IAM 角色和策略。

为 EC2Rescue 创建所需的 IAM 角色和策略

1. 下载 AWSSupport-EC2RescueRole.zip 文件并将 AWSSupport-EC2RescueRole.json 解压到本地计算机上的目录。
2. 如果您的 AWS 账户是一个特殊分区，请编辑模板以将 ARN 值更改为您的分区的值。

   例如，对于中国地区，将出现的所有 arn:aws 更改为 arn:aws-cn。
3. 登录到 AWS Management Console 并打开 AWS CloudFormation 控制台 https://console.aws.amazon.com/cloudformation。
4. 依次选择创建堆栈和使用新资源（标准）。
5. 在创建堆栈页面上，对于先决条件 - 准备模板，选择模板已就绪。
6. 对于指定模板，选择上传模板文件。
7. 选择选择文件，然后进行浏览并从您解压缩 AWSSupport-EC2RescueRole.json 文件的目录中选择此文件。
8. 选择下一步。
9. 在指定堆栈详细信息页上，对于堆栈名称字段，输入用于标识此堆栈的名称，然后选择下一步。
10. （可选）在标签 区域，将一个或多个标签键名称/值对应用到堆栈。

    标签是您分配给资源的可选元数据。标签可让您按各种标准（如用途、所有者或环境）对资源进行分类。例如，您可能需要对堆栈进行标记，以确定其运行的任务类型、涉及的目标或其他资源的类型以及其运行的环境。
11. 选择 Next（下一步）
12. 在审核页面上，审核堆栈详细信息，向下滚动并选择我确认 AWS CloudFormation 可能创建 IAM 资源选项。
13. AWS CloudFormation 将持续几分钟显示 CREATE_IN_PROGRESS 状态。创建堆栈后，状态将变为 CREATE_COMPLETE。您还可以选择刷新图标来检查创建过程的状态。
14. 在堆栈列表中，选择您刚刚创建的堆栈旁边的选项，然后选择输出选项卡。
15. 复制值。这是 AssumeRole 的 ARN。在运行自动化时，您将指定此 ARN。

运行自动化

以下过程介绍如何使用 AWS Systems Manager 控制台运行 AWSSupport-ResetAccess 运行手册。

> **Important**
>
> 以下自动化执行将停止实例。停止实例可能导致附加的实例存储卷（如果存在）上的数据丢失。如果未关联弹性 IP，停止实例也可能导致公有 IP 更改。要避免这些配置更改，请使用 Run Command 重置访问权限。有关更多信息，请参阅《Amazon EC2 用户指南》中的将 EC2Rescue for Windows Server 与 Systems Manager Run Command 结合使用。

要运行 AWSSupport-ResetAccess Automation，请执行以下操作：

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 自动化。
3. 选择执行自动化。
4. 在自动化文档部分中，从列表中选择 Amazon 所拥有。
5. 在运行手册列表中，选择 AWSSupport-ResetAccess 卡片中的按钮，然后选择下一步。
6. 在执行运行手册页面上，选择简单执行。
7. 在文档详细信息部分中，验证文档版本 是否设置为最高默认版本。例如，$DEFAULT 或 3 (默认值))。
8. 在输入参数部分中，指定以下参数：
   a. 对于 InstanceID，指定无法访问实例的 ID。
   b. 对于 SubnetId，指定您指定的实例所在可用区中某个现有 VPC 的子网。默认情况下，Systems Manager 会新建一个 VPC，但您可以根据需要指定现有 VPC 中的某个子网。

      > **Note**
      >
      > 如果未看到指定子网 ID 的选项，请确认使用的是不是运行手册的最新默认版本。

   c. 对于 EC2RescueInstanceType，为 EC2Rescue 实例指定实例类型。默认的实例类型为 `t2.medium`。
   d. 对于 AssumeRole，如果是使用本主题前面介绍的 AWS CloudFormation 过程为此自动化创建的角色，则指定您在 AWS CloudFormation 控制台记下的 AssumeRole ARN。
9. （可选）在标签区域，应用一个或多个标签键名称/值对以帮助识别自动化，例如 `Key=Purpose,Value=ResetAccess`。
10. 选择执行。

11. 要监控自动化过程，请选择正在运行的自动化，然后选择步骤选项卡。完成自动化后，选择描述选项卡，然后选择查看输出以查看结果。要查看单个步骤的输出，请选择步骤选项卡，然后选择步骤旁的查看输出。

运行手册创建一个备份 AMI 和已启用密码的 AMI 作为自动化的一部分。自动化创建的所有其他资源都会自动删除，但这些 AMIs 将保留在您的账户中。这些 AMIs 遵从以下命名约定：

- 备份 AMI：`AWSSupport-EC2Rescue:`*`InstanceID`*
- 启用了密码的 AMI：AWSSupport-EC2Rescue: Password-enabled AMI from *`Instance ID`*

可以通过搜索自动化执行 ID 查找这些 AMIs。

对于 Linux，您的实例的新 SSH 私有密钥加密保存到 Parameter Store 中。参数名称为 /ec2rl/openssh/***`instance ID`***/key。

使用输入变压器将数据传递到 Automation

此 AWS Systems Manager Automation 教程展示了如何使用 Amazon EventBridge 的输入转换器功能，从实例状态更改事件中提取 Amazon Elastic Compute Cloud (Amazon EC2) 实例的 `instance-id`。Automation 是 AWS Systems Manager 中的一项工具。我们使用输入转换器将该数据作为 `InstanceId` 输入参数传递给 `AWS-CreateImage` 运行手册目标。当任何实例更改为 `stopped` 状态时，均将触发该规则。

有关使用输入转换器的更多信息，请参阅 Amazon EventBridge 用户指南中的教程：使用输入转换器自定义要传递给事件目标的内容。

开始前的准备工作

验证是否已向 Systems Manager 自动化服务角色添加了 EventBridge 所需的权限和信任策略。有关更多信息，请参阅 Amazon EventBridge 用户指南中的管理 EventBridge 资源访问权限概述。

将输入转换器与自动化结合使用

1. 打开位于 https://console.aws.amazon.com/events/ 的 Amazon EventBridge 控制台。
2. 在导航窗格中，选择规则。
3. 选择创建规则。
4. 为规则输入名称和描述。

   规则不能与同一区域中的另一个规则和同一事件总线上的名称相同。

5. 对于事件总线，请选择要与此规则关联的事件总线。如果您希望此规则响应来自您自己的 AWS 账户的匹配事件，请选择 defaul（默认）。当您账户中的某个 AWS 服务发出一个事件时，它始终会发送到您账户的默认事件总线。
6. 对于规则类型，选择具有事件模式的规则。
7. 选择下一步。
8. 对于事件源，选择AWS 事件或 EventBridge 合作伙伴事件。
9. 在 Event pattern（事件模式）部分，选择 Event pattern form（事件模式表单）。
10. 对于 Event source（事件源），选择 AWS services（服务）。
11. 在 AWS service（服务）中，选择 EC2。
12. 对于事件类型，请选择 EC2 实例状态更改通知。
13. 适用于 Specific state(s)（特定状态），选择 stopped（已停止）。
14. 选择 Next（下一步）。
15. 对于目标类型，选择AWS 服务。
16. 对于 Target（目标），选择 Systems Manager Automation。
17. 对于 Document（文档），选择 AWS-CreateImage。
18. 在 Configure automation parameter(s)（配置自动化参数）部分中，选择 Input Transformer（输入转换器）。
19. 对于 Input path（输入路径），输入 **`{"instance":"$.detail.instance-id"}`**。
20. 对于 Template（模板），输入 **`{"InstanceId":[<instance>]}`**。
21. 对于 Execution role（执行角色），选择 Use existing role（使用现有角色），然后选择您的自动化服务角色。
22. 选择 Next（下一步）。
23. （可选）为规则输入一个或多个标签。有关更多信息，请参阅 Amazon EventBridge 用户指南中的标记 Amazon EventBridge 资源。
24. 选择 Next（下一步）。
25. 查看规则详细信息并选择创建规则。

了解 Systems Manager Automation 返回的状态

AWS Systems Manager自动化报告有关运行自动化时自动化操作或步骤所经历的各种状态的详细状态信息，以及整体自动化的同类信息。Automation 是 AWS Systems Manager 中的一项工具。您可以使用以下方法监视自动化状态。

- 在 Systems Manager 自动化控制台中监控执行状态。
- 使用您的首选命令行工具。对于 AWS Command Line Interface(AWS CLI），您可以使用 describe-automation-step-executions 或 get-automation-execution。对于 AWS Tools for Windows PowerShell，您可以使用 Get-SSMAutomationStepExecution 或 Get-SSMAutomationExecution。
- 配置 Amazon EventBridge 以响应操作或自动化状态更改。

有关处理自动化中超时的更多信息，请参阅处理运行手册中的超时。

关于自动化状态

除了整体自动化之外，自动化还会报告单个自动化操作的状态详细信息。

整体自动化状态可能不同于单个操作或步骤报告的状态，如下表所述。

操作的详细状态

状态	详细信息
待处理	该步骤尚未开始运行。如果自动化使用条件操作，则在自动化完成后，如果运行步骤的条件未满足，步骤将保持此状态。如果在步骤运行之前取消了自动化，则步骤也会保持此状态。
InProgress	步骤正在运行。
IN LIST	步骤正在等待输入。
成功	步骤成功完成。这是最终状态。
超时	步骤或批准未在指定的超时期限之前完成。这是最终状态。
正在取消	请求者取消步骤后，该步骤正在停止。
已取消	该步骤在完成之前已由请求者停止。这是最终状态。
已失败	该步骤未成功完成。这是最终状态。
Exited	仅通过 `aws:loop` 操作返回。循环未完全完成。使用 `nextStep`、`onCancel` 或

状态	详细信息
	`onFailure` 属性将循环内的步骤移至外部步骤。

自动化的详细状态

状态	详细信息
待处理	自动化尚未开始运行。
InProgress	自动化正在运行。
IN LIST	自动化正在等待输入。
成功	已成功完成操作。这是最终状态。
超时	步骤或批准未在指定的超时期限之前完成。这是最终状态。
正在取消	请求者取消后，自动化正在停止。
已取消	自动化在完成之前被请求者停止。这是最终状态。
已失败	自动化未成功完成。这是最终状态。

Systems Manager 自动化故障排除

利用以下信息，帮助排查 AWS Systems Manager Automation（AWS Systems Manager 中的一项工具）出现的问题。本主题介绍了依据自动化错误消息解决问题的特定任务。

主题

- 常见自动化错误
- 自动化执行无法启动
- 执行已启动，但是状态为“失败”
- 执行已启动，但超时

常见自动化错误

此部分提供有关常见自动化错误的信息。

VPC not defined 400

默认情况下，当自动化运行 AWS-UpdateLinuxAmi 运行手册或 AWS-UpdateWindowsAmi 运行手册时，系统会在默认 VPC (172.30.0.0/16) 中创建一个临时实例。如果您删除了默认 VPC，会收到以下错误：

```
VPC not defined 400
```

要解决此问题，您必须为 SubnetId 输入参数指定值。

自动化执行无法启动

如果您没有为自动化正确配置 AWS Identity and Access Management（IAM）角色和策略，自动化可能因拒绝访问错误或担任角色无效错误而失败。

拒绝访问

以下示例描述了自动化执行因拒绝访问错误无法启动的情况。

对 Systems Manager API 的访问被拒绝

错误消息：`User: user arn isn't authorized to perform: ssm:StartAutomationExecution on resource: document arn (Service: AWSSimpleSystemsManagement; Status Code: 400; Error Code: AccessDeniedException; Request ID: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx)`

- 可能的原因 1：尝试启动自动化的用户没有调用 StartAutomationExecution API 的权限。要解决此问题，请将所需的 IAM policy 附加到用于启动自动化的用户。
- 可能的原因 2：尝试启动自动化的用户拥有调用 StartAutomationExecution API 的权限，但是无权使用特定运行手册来调用该 API。要解决此问题，请将所需的 IAM policy 附加到用于启动自动化的用户。

由于缺少 PassRole 权限而导致访问被拒

错误消息：`User: user arn isn't authorized to perform: iam:PassRole on resource: automation assume role arn (Service: AWSSimpleSystemsManagement;`

Status Code: 400; Error Code: AccessDeniedException; Request ID: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx)

尝试启动自动化的 IAM 用户没有担任角色所需的 PassRole 权限。要解决此问题，请将 iam:PassRole 策略附加到尝试启动自动化的用户的角色。有关更多信息，请参阅 任务 2：将 iam:PassRole 策略附加到您的自动化角色。

担任角色无效

运行自动化时，要么在运行手册中提供担任角色，要么将担任角色作为运行手册的一个参数值传递。如果未指定或未正确配置担任角色，可出现不同类型的错误。

担任角色格式错误

错误消息：The format of the supplied assume role ARN isn't valid.担任角色格式不正确。要解决该问题，请验证运行手册中是否指定了有效的担任角色，或者在启动自动化时指定为运行时的参数。

无法担任担任角色

错误消息：The defined assume role is unable to be assumed. (Service: AWSSimpleSystemsManagement; Status Code: 400; Error Code: InvalidAutomationExecutionParametersException; Request ID: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx)

- 可能的原因 1：担任角色不存在。要解决该问题，请创建角色。有关更多信息，请参阅 the section called “设置自动化”。创建此角色的特定详细信息在以下主题中介绍 任务 1：为自动化创建服务角色。
- 可能的原因 2：担任角色没有与 Systems Manager 服务的信任关系。要解决该问题，请创建信任关系。有关更多信息，请参阅 《IAM 用户指南》中的我无法担任角色。

执行已启动，但是状态为“失败”

操作特定的失败

运行手册包含步骤，并且步骤按顺序运行。每个步骤会调用一项或多项 AWS 服务 API。API 可确定本步的输入、行为和输出。在多个位置错误可能导致步骤失败。失败消息指示出现错误的时间和位置。

要查看 Amazon Elastic Compute Cloud (Amazon EC2) 控制台中的失败消息，请选择失败步骤的查看输出链接。要查看 AWS CLI 中的失败消息，请调用 get-automation-execution 并查找失败的 StepExecution 中的 FailureMessage 属性。

在以下示例中，与 aws:runInstance 操作相关联的步骤失败。每个示例探讨了不同类型的错误。

缺少映像

错误消息：Automation Step Execution fails when it's launching the instance(s). Get Exception from RunInstances API of ec2 Service. Exception Message from RunInstances API: [The image id '[ami id]' doesn't exist (Service: AmazonEC2; Status Code: 400; Error Code: InvalidAMIID.NotFound; Request ID: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx)]. Please refer to Automation Service Troubleshooting Guide for more diagnosis details.

aws:runInstances 操作收到的 ImageId 输入不存在。要解决该问题，请用正确的 AMI ID 更新运行手册或参数值。

担任角色策略缺少足够的权限

错误消息：Automation Step Execution fails when it's launching the instance(s). Get Exception from RunInstances API of ec2 Service. Exception Message from RunInstances API: [You aren't authorized to perform this operation. Encoded authorization failure message: xxxxxxx (Service: AmazonEC2; Status Code: 403; Error Code: UnauthorizedOperation; Request ID: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx)]. Please refer to Automation Service Troubleshooting Guide for more diagnosis details.

担任角色的权限不足，无法在 EC2 实例上调用 RunInstances API。要解决此问题，请将 IAM policy 附加到有权调用 RunInstances API 的担任角色。有关更多信息，请参阅 使用控制台为 Automation 创建服务角色。

意外状态

错误消息：Step fails when it's verifying launched instance(s) are ready to be used. Instance i-xxxxxxxxx entered unexpected state: shutting-down. Please refer to Automation Service Troubleshooting Guide for more diagnosis details.

- 可能的原因 1：实例或 Amazon EC2 服务有问题。要解决此问题，请登录到实例或查阅实例系统日志以了解实例启动关闭的原因。
- 可能的原因 2：为 aws:runInstances 操作指定的用户数据脚本有问题或语法错误。验证用户数据脚本的语法。另外，请验证用户数据脚本是否未关闭实例，或者调用了关闭该实例的其他脚本。

操作特定失败参考

如果步骤失败，失败消息可能指示失败时正在调用哪个服务。下表列出每个操作调用的服务。该表还提供指向有关每个服务的信息的链接。

操作	此操作调用的 AWS 服务	有关此服务的信息	排查内容问题
aws:runInstances	Amazon EC2	Amazon EC2 用户指南	EC2 实例故障排除
aws:changeInstanceState	Amazon EC2	Amazon EC2 用户指南	EC2 实例故障排除
aws:runCommand	Systems Manager	AWS Systems Manager Run Command	对 Systems Manager Run Command 进行故障排除
aws:createImage	Amazon EC2	Amazon Machine Images	
aws:createStack	AWS CloudFormation	《AWS CloudFormation 用户指南》	AWS CloudFormation 故障排除
aws:deleteStack	AWS CloudFormation	《AWS CloudFormation 用户指南》	AWS CloudFormation 故障排除
aws:deleteImage	Amazon EC2	亚马逊机器映像	
aws:copyImage	Amazon EC2	Amazon Machine Images	
aws:createTag	Amazon EC2，Systems Manager	EC2 资源和标签	
aws:invokeLambdaFunction	AWS Lambda	AWS Lambda 开发人员指南	Lambda 故障排除

自动化服务内部错误

错误消息：`Internal Server Error. Please refer to Automation Service Troubleshooting Guide for more diagnosis details.`

自动化 服务中的一个问题导致指定运行手册无法正确运行。要解决此问题，请联系 AWS 支持。请提供执行 ID 和客户 ID (如果有)。

执行已启动，但超时

错误消息：`Step timed out while step is verifying launched instance(s) are ready to be used. Please refer to Automation Service Troubleshooting Guide for more diagnosis details.`

`aws:runInstances` 操作中的步骤超时。如果运行本步骤操作所花的时间超出此步骤中为 `timeoutSeconds` 指定的值，就会发生这种情况。要解决此问题，请为 `aws:runInstances` 操作中的 `timeoutSeconds` 参数指定更长的值。如果不能解决问题，请调查步骤运行时间超出预期的原因

AWS Systems Manager Change Calendar

Change Calendar（AWS Systems Manager 中的一项工具）让您能够设置可以或不可以在 AWS 账户中执行指定操作（例如在 Systems Manager Automation 运行手册中）的日期和时间范围。在 Change Calendar 中，这些范围称为事件。在您创建 Change Calendar 条目时，您将创建类型 `ChangeCalendar` 的 Systems Manager 文档 在 Change Calendar 中，文档以纯文本格式存储 iCalendar 2.0 数据。您添加到 Change Calendar 条目的事件将成为该文档的一部分。要开始使用 Change Calendar，请打开 Systems Manager 控制台。在导航窗格中，请选择 Change Calendar。

您可以在 Systems Manager 控制台中创建日历及其事件。您还可以导入从受支持的第三方日历提供程序导出的 iCalendar (`.ics`) 文件，以将其事件添加到您的日历。受支持的提供程序包括 Google 日历、Microsoft Outlook 和 iCloud 日历。

Change Calendar 条目可以是以下两种类型之一：

`DEFAULT_OPEN`，即 Open by default (预设情况下打开)

默认情况下，除日历事件期间外，所有操作都可以运行。在事件期间，`DEFAULT_OPEN` 日历的状态为 `CLOSED`，且事件被阻止运行。

`DEFAULT_CLOSED`，即 Closed by default (预设情况下关闭)

默认情况下，除日历事件期间外，所有操作都被阻止运行。在事件期间，`DEFAULT_CLOSED` 日历的状态为 `OPEN`，且操作被允许运行。

您可以选择将所有计划的自动化工作流程、维护时段和 State Manager 关联自动添加到日历中。您也可以从日历显示中删除其中任何一种单独类型。

谁应该使用 Change Calendar？

- 执行以下操作类型的 AWS 客户：
  - 创建或运行自动化运行手册。
  - 在 Change Manager 中创建更改请求。
  - 运行维护时段。
  - 在 State Manager 中创建关联。

  Automation、Change Manager、Maintenance Windows、和State Manager都是 AWS Systems Manager 中的工具。通过将这些工具与 Change Calendar 集成在一起，您可以根据与每个操作类型关联的更改日历的当前状态，来允许或阻止这些操作类型。
- 负责维护 Systems Manager 托管式节点配置的一致性、稳定性和功能性的管理员。

Change Calendar 的优势

以下是 Change Calendar 的一些优势。

- 在应用更改之前进行审核

  Change Calendar 条目可以帮助确保在应用对环境可能会造成破坏的更改之前对其进行审核。
- 仅在合适时间应用更改

  Change Calendar 条目有助于在事件期间保持环境稳定。例如，在您预期会有大量资源需求（例如在会议或公共营销促销期间）时，可以创建一个 Change Calendar 条目来阻止更改。当您预计管理员只能提供有限的支持（例如在度假或假期）时，日历条目也可以阻止更改。在管理员提供的支持有限，无法对失败的操作或部署进行故障排除时，您可以使用日历条目允许在一天或一周中的特定时间以外进行更改。
- 获取日历当前或之后的状态

  您可以运行 Systems Manager `GetCalendarState` API 操作，显示日历的当前状态、在指定时间的状态或下次计划日历状态更改的时间。

> **Note**
>
> `GetCalendarState` API 的配额为每秒 10 个请求。有关 Systems Manager 配额的更多信息，请参阅《Amazon Web Services 一般参考》中的 Systems Manager 服务配额。

- EventBridge 支持

  支持此 Systems Manager 工具作为 Amazon EventBridge 规则中的一个事件类型。有关更多信息，请参阅 使用 Amazon EventBridge 监控 Systems Manager 事件 和 引用：Amazon EventBridge 事件模式和 Systems Manager 类型。

主题

- 设置 Change Calendar
- 使用 Change Calendar
- 向自动化运行手册添加 Change Calendar 依赖关系
- 排查 Change Calendar 问题

设置 Change Calendar

使用 Change Calendar（AWS Systems Manager 中的一项工具）前请完成以下操作。

安装最新的命令行工具

安装最新的命令行工具，以获取有关日历状态的信息。

要求	描述
AWS CLI	（可选）要使用 AWS Command Line Interface (AWS CLI) 获取有关日历状态的信息，请在本地计算机上安装 AWS CLI 的最新版本。 有关如何安装或升级 CLI 的更多信息，请参阅 AWS Command Line Interface 用户指南中的安装、更新和卸载 AWS CLI。

要求	描述
AWS Tools for PowerShell	（可选）要使用 Tools for PowerShell 获取有关日历状态的信息，请在本地计算机上安装 Tools for PowerShell 的最新版本。 有关如何安装或升级 Tools for PowerShell 的更多信息，请参阅 AWS Tools for PowerShell 用户指南中的安装 AWS Tools for PowerShell。

设置权限

如果已为您的用户、组或角色分配了管理员权限，则您对 Change Calendar 有完全访问权。如果您没有管理员权限，则管理员必须通过向您的用户、组或角色分配 AmazonSSMFullAccess 托管策略或分配提供必要权限的策略，来向您授予权限。

使用 Change Calendar 需要以下权限。

Change Calendar 条目

要创建、更新或删除 Change Calendar 条目（包括在条目中添加和删除事件），附加到用户、组或角色的策略必须允许执行以下操作：

- ssm:CreateDocument
- ssm:DeleteDocument
- ssm:DescribeDocument
- ssm:DescribeDocumentPermission
- ssm:GetCalendar
- ssm:ListDocuments
- ssm:ModifyDocumentPermission
- ssm:PutCalendar
- ssm:UpdateDocument
- ssm:UpdateDocumentDefaultVersion

日历状态

要获取有关日历当前或之后状态的信息，附加到用户、组或角色的策略必须允许执行以下操作：

- ssm:GetCalendarState

操作事件

要查看操作事件，例如维护时段、关联和计划的自动化，附加到用户、组或角色的策略必须允许执行以下操作：

- ssm:DescribeMaintenanceWindows
- ssm:DescribeMaintenanceWindowExecution
- ssm:DescribeAutomationExecutions
- ssm:ListAssociations

> **Note**
>
> 由您的账户以外的账户拥有（即该账户创建）的 Change Calendar 条目为只读，即使它们与您的账户共享也是如此。维护时段、State Manager 关联和自动化不会共享。

使用 Change Calendar

您可以使用 AWS Systems Manager 控制台添加、管理或删除 Change Calendar（AWS Systems Manager 中的一项工具）条目。您还可以从受支持的第三方日历提供程序导入事件，方法是导入从源日历导出的 iCalendar (.ics) 文件。而且，您也可以使用 GetCalendarState API 操作或 get-calendar-state AWS Command Line Interface (AWS CLI) 命令在特定时间获取有关 Change Calendar 状态的信息。

主题

- 创建更改日历
- 在 Change Calendar 中创建和管理事件
- 导入和管理来自第三方日历的事件
- 更新更改日历
- 共享更改日历
- 删除更改日历
- 获取更改日历的状态

创建更改日历

在 Change Calendar（AWS Systems Manager 中的一项工具）中创建条目时，您要用到使用 `text` 格式的 Systems Manager 文档（SSM 文档）。

创建更改日历

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Change Calendar。
3. 选择 Create calendar（创建日历）。

   –或者–

   如果 Change Calendar 主页首先打开，则选择 Create change calendar（创建更改日历）。
4. 在 Create calendar (创建日历) 页面上的 Calendar details (日历详细信息) 中，输入日历条目的名称。日历条目名称可以包含字母、数字、句点、短划线和下划线。名称应该足够具体，以便一目了然地确定日历条目的用途。例如，**`support-off-hours`**。创建日历条目后，您将无法更新此名称。
5. （可选）对于 Description（描述），输入日历条目的描述。
6. （可选）在 Import calendar (导入日历) 区域中，选择 Choose file (选择文件)，以选择您从第三方日历提供程序导出的 iCalendar (`.ics`) 文件。导入该文件会将其事件添加到您的日历中。

   受支持的提供程序包括 Google 日历、Microsoft Outlook 和 iCloud 日历。

   有关更多信息，请参阅 从第三方日历提供程序导入事件。
7. 在 Calendar type (日历类型) 中，选择以下选项之一。

   - Open by default (预设情况下打开) - 日历处于打开状态（自动化操作可以运行，直到事件开始），然后在关联事件的持续时间内关闭。
   - Closed by default (预设情况下关闭) - 日历处于关闭状态（自动化操作无法运行，直到事件开始），但在关联事件的持续时间内打开。
8. （可选）在变更管理事件中，选择向日历添加变更管理事件。此选项会在您的月度日历显示中显示所有的计划维护时段、State Manager 关联、自动化工作流程和 Change Manager 变更请求。

> **Tip**
>
> 如果稍后要从日历显示中永久移除这些事件类型，请编辑日历，清除此复选框，然后选择保存。

9. 选择 Create calendar（创建日历）。

   创建日历条目后，Systems Manager 将在 Change Calendar 列表中显示您的日历条目。列显示日历版本和日历拥有者的 AWS 账户 编号。在创建或导入至少一个事件之前，您的日历条目无法阻止或允许任何操作。有关创建事件的信息，请参阅 创建Change Calendar 事件。有关导入事件的信息，请参阅 从第三方日历提供程序导入事件。

在 Change Calendar 中创建和管理事件

在 AWS Systems Manager Change Calendar 中创建日历后，可以创建、更新和删除打开或关闭的日历中包含的事件。Change Calendar 是 AWS Systems Manager 中的一项工具。

> **Tip**
>
> 作为直接在 Systems Manager 控制台中创建事件的替代方法，您可以从受支持的第三方日历应用程序导入 iCalendar (`.ics`) 文件。有关信息，请参阅导入和管理来自第三方日历的事件。

主题

- 创建Change Calendar 事件
- 更新 Change Calendar 事件
- 删除 Change Calendar 事件

创建Change Calendar 事件

在向 Change Calendar（AWS Systems Manager 中的一项工具）中的条目添加事件时，您要指定暂停日历条目默认操作的时间段。例如，如果日历条目类型为预设情况下关闭，则日历在事件期间可以更改。（或者，您可以创建一个咨询活动，该活动仅在日历上发挥信息作用。）

目前，您只能使用控制台创建 Change Calendar 事件。事件将添加到您在创建 Change Calendar 条目时创建的 Change Calendar 文档中。

创建 Change Calendar 事件

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Change Calendar。
3. 在日历的列表中，请选择要向其添加事件的日历条目的名称。
4. 在日历条目的详细信息页面上，选择 Create event (创建事件)。
5. 在 Create scheduled event (创建计划事件) 页面的 Event details (事件详细信息) 中，输入事件的显示名称。事件名称可以包含字母、数字、句点、短划线和下划线。名称应该足够具体，以便确定事件的用途。例如，**`nighttime-hours`**。
6. 对于 Description（描述），输入事件的描述。例如，**`The support team isn't available during these hours`**。
7. （可选）如果您希望此事件仅作为视觉通知或提醒，请选择 Advisory（咨询）复选框。咨询事件在您的日历上不起任何功能作用。它们仅为查看日历的用户提供信息。
8. 对于 Event start date（事件开始日期），以 `MM/DD/YYYY` 格式输入或选择开始事件的日期，并以 `hh:mm:ss` 格式（小时、分钟和秒）输入在指定日期开始事件的时间。
9. 对于 Event end date（事件结束日期），以 `MM/DD/YYYY` 格式输入或选择结束事件的日期，并以 `hh:mm:ss` 格式（小时、分钟和秒）输入在指定日期结束事件的时间。
10. 在 Schedule time zone（计划时区）中，选择应用于事件开始和结束时间的时区。您可以输入城市名称或不同于格林威治标准时间 (GMT) 的时区的一部分，以便更快地查找时区。默认值为协调世界时 (UTC)。
11. （可选）要创建每天、每周或每月重复的事件，请启用 Recurrence（重复），然后指定重复的频率和可选的结束日期。
12. 选择 Create scheduled event (创建计划事件)。新事件将添加到您的日历条目中，并显示在日历条目详细信息页面的 Events (事件) 选项卡上。

更新 Change Calendar 事件

按照以下过程在 AWS Systems Manager 控制台中更新 Change Calendar 事件。Change Calendar 是 AWS Systems Manager 中的一项工具。

更新 Change Calendar 事件

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。

2. 在导航窗格中，选择 Change Calendar。
3. 在日历的列表中，请选择要编辑其事件的日历条目的名称。
4. 在日历条目的详细信息页面上，选择 Events (事件)。
5. 在日历页面中，选择要编辑的事件。

> **Tip**
>
> 使用左上角的按钮可向后或向前移动一年，或者向后或向前移动一个月。如果需要，请通过从右上角的列表中选择正确的时区来更改时区。

6. 在 Event details（事件详细信息）中，请选择 Edit（编辑）。

   要更改事件名称和描述，请添加或替换当前文本值。
7. 要更改 Event start date（事件开始日期）值，请选择当前的开始日期，然后从日历中选择一个新日期。要更改开始时间，请选择当前开始时间，然后从列表中选择一个新时间。
8. 要更改 Event end date（事件结束日期）值，请选择当前日期，然后从日历中选择一个新的结束日期。要更改结束时间，请选择当前结束时间，然后从列表中选择一个新时间。
9. 要更改 Schedule time zone（计划时区）值，请选择应用于事件开始和结束时间的时区。您可以输入城市名称或不同于格林威治标准时间 (GMT) 的时区的一部分，以便更快地查找时区。默认值为协调世界时 (UTC)。
10. （可选）如果您希望此事件仅作为视觉通知或提醒，请选择 Advisory（咨询）复选框。咨询事件在您的日历上不起任何功能作用。它们仅为查看日历的用户提供信息。
11. 选择保存。您的更改将显示在日历条目详细信息页面的 Events (事件) 选项卡上。选择您更新的事件以查看更改。

删除 Change Calendar 事件

您可以使用 AWS Management Console，在 Change Calendar（AWS Systems Manager 中的一项工具）中逐个删除事件。

> **Tip**
>
> 如果您在创建日历时选择了向日历添加变更管理事件，则可以执行以下操作：
>
> - 要在日历显示中暂时隐藏变更管理事件类型，请为每月预览顶部的类型选择 X。

- 要从日历显示中永久移除这些类型，请编辑日历，清除向日历添加变更管理事件复选框，然后选择保存。从日历显示中移除类型并不会将其从您的账户中删除。

删除 Change Calendar 事件

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Change Calendar。
3. 在日历的列表中，请选择要删除其事件的日历条目的名称。
4. 在日历条目的详细信息页面上，选择 Events (事件)。
5. 在日历页面中，选择要删除的事件。

> **Tip**
>
> 使用左上角的按钮可将日历向后或向前移动一年，或者向后或向前移动一个月。如果需要，请通过从右上角的列表中选择正确的时区来更改时区。

6. 在 Event details (事件详细信息) 页面上，选择 Delete (删除)。当系统提示您确认是否要删除该事件时，请选择 Confirm（确认）。

导入和管理来自第三方日历的事件

作为直接在 AWS Systems Manager 控制台中，创建事件的替代方法，您可以从受支持的第三方日历应用程序导入 iCalendar (`.ics`) 文件。日历可以同时包含导入的事件和您在 Change Calendar（AWS Systems Manager 中的一项工具）中创建的事件。

开始前的准备工作

在尝试导入日历文件之前，请查看以下要求和约束：

日历文件格式

只支持有效的 iCalendar 文件 (`.ics`)。

受支持的日历提供程序

仅支持从以下第三方日历提供程序导出的 `.ics` 文件：

- Google 日历（导出说明）

- Microsoft Outlook（导出说明）
- iCloud 日历（导出指令）

文件大小

您可以导入任意数量的有效 .ics 文件。但是，每个日历的所有导入文件的总大小不能超过 64KB。

> **Tip**
>
> 要最小化 .ics 文件的大小，请确保只导出有关日历条目的基本详细信息。如有必要，请减少用于导出的时间段的长度。

时区

除了日历名称、日历提供程序和至少一个事件之外，导出的 .ics 文件还应指明日历的时区。如果没有，或者在确定时区时出现问题，系统将在您导入文件后提示您指定一个时区。

重复性事件限制

您导出的 .ics 文件可以包含重复性事件。但是，如果曾在源日历中删除过某一重复性事件的一次或多次发生，则导入将失败。

主题

- 从第三方日历提供程序导入事件
- 更新来自第三方日历提供程序的所有事件
- 删除从第三方日历导入的所有事件

从第三方日历提供程序导入事件

可以使用以下过程从受支持的第三方日历应用程序导入 iCalendar (.ics) 文件。该文件中包含的事件将合并到您打开或关闭的日历的规则中。您可以将文件导入自己使用 Change Calendar（AWS Systems Manager 中的一项工具）创建的新日历或现有日历。

导入 .ics 文件后，可以使用 Change Calendar 界面从该文件中删除单个事件。有关信息，请参阅删除 Change Calendar 事件。您还可以通过删除 .ics 文件，来删除源日历中的所有事件。有关信息，请参阅删除从第三方日历导入的所有事件。

从第三方日历提供程序导入事件

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Change Calendar。
3. 要从新日历开始，请选择 Create calendar (创建日历)。在 Import calendar (导入日历) 区域中，选择 Choose file (选择文件)。有关创建新日历的其他步骤的信息，请参阅 创建更改日历。

   –或者–

   要将第三方事件导入到现有日历中，请选择现有日历的名称，以将其打开。
4. 请选择 Actions, Edit（操作、编辑），然后在 Import calendar（导入日历）区域中，请选择 Choose file（选择文件）。
5. 在您的本地计算机上，导航到已导出的 `.ics` 文件，并选择该文件。
6. 如果出现提示，对于 Select a time zone (选择时区)，请选择哪一个时区适用于该日历。
7. 选择 Save (保存)。

更新来自第三方日历提供程序的所有事件

如果在导入源日历的 iCalendar `.ics` 文件后，向源日历中添加了多个事件，或从其中删除了多个事件，则可以将这些更改反映在 Change Calendar 中。首先，重新导出源日历，再将新文件导入 Change Calendar（AWS Systems Manager 中的一项工具）。更改日历中的事件将被更新，以反映较新文件的内容。

更新来自第三方日历提供程序的所有事件

1. 在您的第三方日历中，只要您希望将某些事件反映在 Change Calendar 中，即可添加或删除这些事件，然后将该日历重新导出到新 `.ics` 文件。
2. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
3. 在导航窗格中，选择 Change Calendar。
4. 从日历列表中，选择日历名称。
5. 选择选择文件，然后导航到替换 `.ics` 文件，并选择该文件。
6. 为了响应有关覆盖现有文件的通知，请选择 Confirm (确认)。

删除从第三方日历导入的所有事件

如果您不再希望将从第三方提供程序导入的任何事件包含在日历中，则可删除已导入的 iCalendar `.ics` 文件。

删除从第三方日历导入的所有事件

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Change Calendar。
3. 从日历列表中，选择日历名称。
4. 在 Import calendar (导入日历) 区域中的 My imported calendars (我导入的日历) 下，找到导入日历的名称，然后选择其卡片中的 X。
5. 选择保存。

更新更改日历

您可以更新更改日历的描述，但不能更新其名称。尽管您可以更改日历的默认状态，但请注意，这会反转更改操作在与该日历关联的事件期间的行为。例如，如果您将日历的状态从 Open by default (预设情况下打开) 更改为 Closed by default (预设情况下关闭)，则当创建了关联事件的用户不希望进行更改时，可能会在事件期间发生不希望的更改。

在更新更改日历时，您要编辑在创建该条目时创建的 Change Calendar 文档。Change Calendar 是 AWS Systems Manager 中的一项工具。

更新更改日历

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Change Calendar。
3. 在日历的列表中，请选择要更新的日历的名称。
4. 在日历的详细信息页面上，请选择 Actions, Edit（操作、编辑）。
5. 在 Description (描述) 中，您可以更改描述文本。您不能编辑更改日历的名称。
6. 要更改日历状态，请在 Calendar type (日历类型) 中选择其他值。请注意，这会反转更改操作在与该日历关联的事件期间的行为。在更改日历类型之前，您应与其他 Change Calendar 用户验证并确保更改日历类型不会在其已创建的事件期间导致不希望的更改。

- Open by default (预设情况下打开) - 日历处于打开状态（自动化操作可以运行，直到事件开始），然后在关联事件的持续时间内关闭。
- Closed by default (预设情况下关闭) - 日历处于关闭状态（自动化操作无法运行，直到事件启动），但在关联事件的持续时间内打开。

7. 选择保存。

   在至少添加一个事件之前，您的日历无法阻止或允许任何操作。有关如何添加事件的信息，请参阅创建Change Calendar 事件。

共享更改日历

您可以使用 AWS Systems Manager 控制台，在 Change Calendar（AWS Systems Manager 中的一项工具）中与其他 AWS 账户共享日历。共享日历时，日历对于共享账户中的用户为只读。维护时段、State Manager 关联和自动化不会共享。

共享更改日历

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Change Calendar。
3. 在日历的列表中，请选择要共享的日历的名称。
4. 在日历的详细信息页面上，请选择 Sharing（共享）选项卡。
5. 依次选择 Actions, Share（操作、共享）。
6. 在 Share calendar (共享日历) 中，对于 Account ID (账户 ID)，输入有效 AWS 账户 的 ID 号，然后选择 Share (共享)。

   共享账户的用户可以读取更改日历，但他们不能进行更改。

删除更改日历

您可以使用 Systems Manager 控制台或 AWS Command Line Interface（AWS CLI），在 Change Calendar（AWS Systems Manager 中的一项工具）中删除日历。删除更改日历会删除所有关联的事件。

删除更改日历

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Change Calendar。
3. 在日历的列表中，请选择要删除的日历的名称。
4. 在日历的详细信息页面上，请选择 Actions, Delete（操作、删除）。当系统提示您确认是否要删除日历时，选择 Delete (删除)。

获取更改日历的状态

您可以在 Change Calendar（AWS Systems Manager 中的一项工具）中获取日历的整体状态或者日历在特定时间的状态。您还可以显示日历状态下次从 OPEN 更改为 CLOSED（或反之）的时间。

> **Note**
>
> 有关将 Change Calendar 与 Amazon EventBridge 集成以自动监控日历状态变化的信息，请参阅the section called “Change Calendar 与 Amazon EventBridge 集成”。EventBridge 集成在日历状态转换时提供事件驱动的通知，补充基于轮询的 GetCalendarState API 操作方法。

您只能使用 GetCalendarState API 操作完成此任务。本节中的过程使用 AWS Command Line Interface (AWS CLI)。

获取更改日历的状态

- 运行以下命令，以显示一个或多个日历在特定时间的状态。--calendar-names 参数是必需项，但 --at-time 是可选项。将每个*#######*替换为您自己的信息。

  Linux & macOS

  ```
  aws ssm get-calendar-state \
      --calendar-names "Calendar_name_or_document_ARN_1"
   "Calendar_name_or_document_ARN_2" \
      --at-time "ISO_8601_time_format"
  ```

  示例如下：

  ```
  aws ssm get-calendar-state \
  ```

```
    --calendar-names "arn:aws:ssm:us-east-2:123456789012:document/
MyChangeCalendarDocument" "arn:aws:ssm:us-east-2:123456789012:document/
SupportOffHours" \
    --at-time "2020-07-30T11:05:14-0700"
```

Windows

```
aws ssm get-calendar-state ^
    --calendar-names "Calendar_name_or_document_ARN_1"
 "Calendar_name_or_document_ARN_2" ^
    --at-time "ISO_8601_time_format"
```

示例如下：

```
aws ssm get-calendar-state ^
    --calendar-names "arn:aws:ssm:us-east-2:123456789012:document/
MyChangeCalendarDocument" "arn:aws:ssm:us-east-2:123456789012:document/
SupportOffHours" ^
    --at-time "2020-07-30T11:05:14-0700"
```

此命令会返回如下信息。

```
{
    "State": "OPEN",
    "AtTime": "2020-07-30T16:18:18Z",
    "NextTransitionTime": "2020-07-31T00:00:00Z"
}
```

结果显示指定日历条目（由您的账户拥有或与您的账户共享）在 `--at-time` 的值所指定时间的日历状态（无论日历的类型为 DEFAULT_OPEN 还是 DEFAULT_CLOSED），以及下一次转换的时间。如果您不添加 `--at-time` 参数，则使用当前时间。

> **Note**
>
> 如果在一个请求中指定了多个日历，则仅当该请求中的所有日历都处于打开状态时，该命令才会返回 OPEN 的状态。如果该请求中的一个或多个日历处于关闭状态，则返回的状态为 CLOSED。

向自动化运行手册添加 Change Calendar 依赖关系

要让 Automation 操作遵循 Change Calendar（AWS Systems Manager 中的一项工具），请在使用 aws:assertAwsResourceProperty 操作的 Automation 运行手册中添加一个步骤。将该操作配置为运行 GetCalendarState 以验证指定的日历条目是否处于所需状态（OPEN 或 CLOSED）。只有在日历状态为 OPEN 时才允许自动化运行手册继续执行下一步。以下是基于 YAML 的示例，摘录了只有当日历状态与在 DesiredValues 中指定的状态 OPEN 匹配时，才能继续到下一步 LaunchInstance 的自动化运行手册。

示例如下：

```
mainSteps:
  - name: MyCheckCalendarStateStep
    action: 'aws:assertAwsResourceProperty'
    inputs:
      Service: ssm
      Api: GetCalendarState
      CalendarNames: ["arn:aws:ssm:us-east-2:123456789012:document/SaleDays"]
      PropertySelector: '$.State'
      DesiredValues:
      - OPEN
    description: "Use GetCalendarState to determine whether a calendar is open or
 closed."
    nextStep: LaunchInstance
  - name: LaunchInstance
    action: 'aws:executeScript'
    inputs:
      Runtime: python3.8
...
```

排查 Change Calendar 问题

利用以下信息，帮助排查Change Calendar（AWS Systems Manager 中的一项工具）出现的问题。

主题

- “Calendar import failed (日历导入失败)”错误

“Calendar import failed (日历导入失败)”错误

问题：在导入 iCalendar (.ics) 文件时，系统报告日历导入失败。

- 解决方案 1 – 确保您导入的是从受支持的第三方日历提供程序导出的文件，受支持的第三方日历提供程序包括以下几个：
  - Google 日历（导出说明）
  - Microsoft Outlook（导出说明）
  - iCloud 日历（导出指令）
- 解决方案 2 – 如果源日历包含任何重复性事件，请确保没有取消或删除该事件的任何一次发生。目前，Change Calendar 不支持导入发生单次取消的重复性事件。要解决该问题，请从源日历中删除重复性事件，重新导出日历，再将该日历重新导入到 Change Calendar 中，然后使用 Change Calendar 界面添加该重复性事件。有关信息，请参阅创建Change Calendar 事件。
- 解决方案 3 – 确保源日历至少包含一个事件。上载 `.ics` 不包含事件的文件不会成功。
- 解决方案 4 – 如果系统报告导入因 `.ics` 过大而失败，请确保只导出有关日历条目的基本详细信息。如有必要，请减少用于导出的时间段的长度。
- 解决方案 5 – 如果 Change Calendar 无法确定您导出的日历的时区，则当您尝试从 Events (事件) 选项卡中导入该日历时，您可能会收到以下消息：“Calendar import failed. Change Calendar couldn't locate a valid time zone.（日历导入失败，找不到有效时区。）您可以从 Edit (编辑) 菜单中导入该日历。在本例中，请选择 Actions (操作)、Edit (编辑)，然后尝试从 Edit calendar (编辑日历) 页面导入该文件。
- 解决方案 6 – 请勿编辑 `.ics` 文件，然后再导入。尝试修改文件的内容可能会损坏日历数据。如果您在尝试导入之前修改了文件，则请再次从源日历导出日历，然后重新尝试上载。

AWS Systems Manager Change Manager

Change Manager（AWS Systems Manager 中的一项工具）是一个企业变更管理框架，用于请求、批准、实施和报告应用程序配置及基础设施的操作变更。如果您使用 AWS Organizations，可从单个委托管理员账户管理多个 AWS 账户 和整个 AWS 区域中的更改。或者，也可使用本地账户管理单个 AWS 账户 的更改。使用 Change Manager 可管理对 AWS 资源和本地部署资源的变更。要开始使用 Change Manager，请打开 Systems Manager 控制台。在导航窗格中，请选择 Change Manager。

借助 Change Manager，您可以使用预先批准的更改模板，帮助自动完成对资源的更改过程，并帮助避免在进行操作更改时出现意外结果。每个更改模板指定以下内容：

- 在创建更改请求时，有一个或多个自动化运行手册可供用户选择。对您的资源进行的更改将在自动化 Runbook 中定义。您可以将自定义运行手册或 AWS 托管运行手册包含在您创建的更改模板中。当用户创建更改请求时，他们可以选择要在该请求中包含哪一个可用的运行手册。此外，您还可以创建更改模板，让提出请求的用户在更改请求中指定任何运行手册。

- 账户中的用户，必须审核使用该更改模板提出的更改请求。
- Amazon Simple Notification Service (Amazon SNS) 主题，用于通知指定审批人员，更改请求已准备好进行审核。
- Amazon CloudWatch 告警，用于监控运行手册工作流。
- Amazon SNS 主题，用于针对使用更改模板创建的更改请求，发送有关状态变化的通知。
- 要应用于更改模板的标签，用于对更改模板进行分类和筛选。
- 是否可以在没有批准步骤的情况下运行从更改模板创建的更改请求（自动批准的请求）。

通过与 Systems Manager 中的另一项工具 Change Calendar 集成，Change Manager 还可以帮助您安全地实施更改，同时避免计划与重要业务事件发生冲突。Change Manager 与 AWS Organizations 和 AWS IAM Identity Center 集成，可帮助您使用现有的身份管理系统从单个账户管理整个组织中的更改。您可以从 Change Manager 监控更改进度，并审核整个组织中的操作更改，从而提高可见性、改善问责制。

Change Manager 为您的持续集成 (CI) 实践和持续交付 (CD) 方法补充了安全控制措施。Change Manager 不适用于作为自动发布流程（例如 CI/CD 管道）的组成部分进行的更改，除非存在例外情况，或者需要批准。

Change Manager 的工作原理

在确定需要标准或紧急操作更改时，组织中的某个人员会根据为在您的组织或账户中使用而创建的更改模板之一创建更改请求。

如果请求的更改需要手动批准，Change Manager 将通过 Amazon SNS 通知来通知指定的审批人员，更改请求已准备好供其审核。您可以在更改模板中为更改请求指定审批者，也可以让用户在更改请求本身中指定审批者。您可以将不同的审核人员分配给不同的模板。例如，分配一个用户、用户组或 AWS Identity and Access Management (IAM) 角色，他们必须批准对托管式节点的更改请求；而对数据库更改，则分配另一个用户、组或 IAM 角色。如果更改模板允许自动审批，并且请求者的用户策略不禁止它，则用户还可以选择为其请求运行自动化运行手册，而无需审核步骤（更改冻结事件除外）。

对于每个更改模板，您可以添加最多五个级别的审批人员。例如，您可能要求技术审核人员先批准根据更改模板创建的更改请求，然后需要一个或多个经理的第二级批准。

Change Manager 已与 AWS Systems Manager Change Calendar 集成。当请求的更改获得批准后，系统将首先确定该请求是否与其他已计划的业务活动冲突。如果检测到冲突，Change Manager 可以阻止更改，或在启动运行手册工作流之前要求进行额外批准。例如，您可能只允许在工作时间进行更

改，以确保团队可以管理任何意外问题。对于请求在上述时间以外运行的任何更改，您可以要求更高级别的管理人员以更改冻结审批人员的形式批准。对于紧急更改，Change Manager 可以跳过检查 Change Calendar 是否存在冲突或在批准更改请求后阻止事件的步骤。

当需要实施已批准的更改时，Change Manager 将运行在关联的更改请求中指定的自动化运行手册。在运行手册工作流运行时，只允许在已批准的更改请求中定义的操作。这种方法有助于您避免在实施更改时出现意外结果。

除了限制在运行手册工作流运行时可以进行的更改之外，Change Manager 还可以帮助您控制并发和错误阈值。您可以选择运行手册工作流一次可在多少个资源上运行，更改一次可在多少个账户中运行，以及在停止流程和（如果运行手册包含回滚脚本）回滚之前允许多少次故障。您还可以使用 CloudWatch 告警监控正在进行的更改的进度。

运行手册工作流完成后，您可以查看有关所做更改的详细信息。这些详细信息包括更改请求的原因、使用的更改模板、请求和批准更改的人员，以及更改的实施方式。

更多信息

在 AWS 新闻博客上介绍 AWS Systems ManagerChange Manager

我的操作如何从 Change Manager 获益？

Change Manager 具有以下优势：

- 降低服务中断和停机风险

  Change Manager 可以确保在运行手册工作流运行时，仅实施已批准的更改，从而使操作更改更安全。您可以阻止计划外和未经审核的更改。Change Manager 可以帮助您避免由于人为错误导致的多种类型的意外结果，这些结果需要花费宝贵时间进行研究和回溯。
- 获取更改历史记录的详细审核和报告

  Change Manager 提供了一种一致的问责制方法，用于报告和审核在整个组织中所做的更改、更改的目的，以及与批准和实施这些更改的人员有关的详细信息。
- 避免计划冲突或违规

  Change Manager 可以根据您的组织的活动更改日历来检测计划冲突，例如假日事件或新产品上市。您可以允许运行手册工作流仅在工作时间运行，也可以仅在获得额外批准的情况下才允许它们运行。
- 使更改要求适应不断变化的业务

在不同的业务期间，您可以实施不同的更改管理要求。例如，在月末报告、纳税季节或其他关键业务期间，您可以阻止更改，或对可能带来不必要的运营风险的更改要求董事级批准。

- 集中管理多个账户的更改

  通过与 Organizations 集成，Change Manager 使您可以通过单个委托管理员账户管理所有组织单位 (OU) 中的更改。您可以启用 Change Manager，与您的整个组织一起使用，也可以仅与您的部分 OU 一起使用。

谁应该使用 Change Manager？

Change Manager 适用于以下 AWS 客户和组织：

- 任何希望改善对其云环境或本地环境所做操作更改的安全性和治理性的 AWS 客户。
- 希望提高多个团队间的协作和可见性，通过避免停机提高应用程序可用性，以及降低与手动和重复性任务相关的风险的组织。
- 必须遵循更改管理的最佳做法的组织。
- 对于其应用程序配置和基础设施所做的更改，需要完全可审核的历史记录的客户。

Change Manager 的主要功能是什么？

Change Manager 包含以下主要功能：

- 对更改管理最佳做法的综合支持

  借助 Change Manager，您可以将所选择的更改管理最佳做法应用于您的操作。您可以选择启用选项：

  - 检查 Change Calendar，以查看事件当前是否受到限制，以便仅在已打开日历期间进行更改。
  - 在获得更改冻结审批人员额外批准的情况下，允许在受限事件期间进行更改。
  - 要求为所有更改模板指定 CloudWatch 告警。
  - 要求在您的账户中创建的所有更改模板经过审核和批准，然后才能使用它们创建更改请求。
- 针对已关闭日历期间和紧急更改请求的不同批准路径

  您可以允许一个选项来检查受限事件的 Change Calendar，并阻止已批准的更改请求，直到事件完成为止。但是，您也可以指定第二组审批人员，即更改冻结审批人员，即使日历已关闭，他们也可以允许进行更改。您还可以创建紧急更改模板。从紧急更改模板创建的更改请求仍然需要常规批准，但不受日历限制的约束，也不需要更改冻结批准。

- 控制如何以及何时开始运行手册工作流

  运行手册工作流可以根据计划开始，也可以在批准完成后立即开始（受日历限制规则约束）。

- 内置通知支持

  指定您的组织中应该审核和批准更改模板及更改请求的人员。将 Amazon SNS 主题分配给更改模板，以便向该主题的订阅者发送与使用该更改模板创建的更改请求的状态变化有关的通知。

- 与 AWS Systems Manager Change Calendar 集成

  Change Manager 允许管理员限制在指定时间段内的计划更改。例如，您可以创建一个策略，该策略仅允许在工作时间进行更改，以确保团队有时间处理任何问题。您还可以在重要业务活动期间限制更改。例如，零售业务可在大型销售活动期间限制更改。您还可以在受限时段期间内要求获得额外批准。

- 与 AWS IAM Identity Center 和 Active Directory 支持集成

  借助 IAM Identity Center 集成，您的组织成员可以访问 AWS 账户，并使用 Systems Manager 基于通用用户身份管理其资源。使用 IAM Identity Center，您可以分配对整个 AWS 中账户的用户访问权限。

  通过与 Active Directory 集成，可以指定 Active Directory 账户中的用户担任为 Change Manager 操作创建的更改模板的审批人员。

- 与 Amazon CloudWatch 告警集成

  Change Manager 与 CloudWatch 告警集成。Change Manager 可在运行手册工作流期间侦听 CloudWatch 告警，并执行为相应告警定义的任何操作，包括发送通知。

- 与 AWS CloudTrail Lake 集成

  通过在 AWS CloudTrail Lake 中创建事件数据存储，您可以查看有关您的账户或组织中运行的变更请求所做更改的可审批信息。存储的事件信息包括以下详细信息：

  - 运行的 API 操作
  - 这些操作所包含的请求参数
  - 运行该操作的用户
  - 在此过程中更新的资源

- 与 AWS Organizations 集成

使用 Organizations 提供的跨账户功能，您可以使用委托管理员账户来管理您的组织内 OU 中的 Change Manager 操作。在您的 Organizations 管理账户中，可以指定哪个账户作为委托管理员账户。您还能够控制可在您的哪些 OU 中使用 Change Manager。

使用 Change Manager 是否需要收取费用？

是的。Change Manager 是按使用量付费的。您仅需按实际用量付费。有关更多信息，请参阅 AWS Systems Manager 定价。

Change Manager 的主要组件是什么？

用于管理组织或账户中的更改过程的 Change Manager 组件包括以下内容：

委托管理员账户

如果您在整个组织中使用 Change Manager，则将使用委托管理员账户。此 AWS 账户被指定为管理包括 Change Manager 在内的整个 Systems Manager 中的操作活动的账户。委托管理员账户可以管理整个组织中的更改活动。在将组织设置为使用 Change Manager 时，可以指定您的哪些账户以此角色提供服务。委托管理员账户必须是其分配到的组织单位 (OU) 的唯一成员。如果您仅将 Change Manager 用于单个AWS 账户，则不需要委托管理员账户。

> **Important**
>
> 如果您在整个组织中使用 Change Manager，我们建议始终从委托管理员账户进行更改。虽然您可以从组织中的其他账户进行更改，但这些更改不会在委派管理员账户中报告，也无法从该账户查看。

更改模板

更改模板是 Change Manager 中的一个配置设置集合，它定义了所需的批准、可用的运行手册和更改请求的通知选项等内容。

您可以要求您的组织或账户中的用户创建的更改模板经过批准流程，然后才能使用这些模板。

Change Manager 支持两种类型的更改模板。对于基于紧急更改模板，请求的更改可以进行，即使在 Change Calendar。对于基于标准更改模板，则无法进行请求的更改，如果 Change Calendar 除非收到指定的额外批准更改冻结事件审批。

更改请求

更改请求是 Change Manager 中的一个请求，旨在运行更新您的 AWS 或本地环境中一个或多个资源的自动化运行手册。更改请求是使用更改模板创建的。

在创建更改请求时，您的组织或账户中的一个或多个审批人员必须审核和批准该请求。如果没有获得所需的批准，则不允许运行应用您请求的更改的运行手册工作流。

在系统中，更改请求是 AWS Systems Manager OpsCenter 中一种类型的 OpsItem。不过，`/aws/changerequest` 类型的 OpsItems 不会显示在 OpsCenter 中。作为 OpsItems，更改请求与其他类型的 OpsItems 一样，也受相同强制配额的约束。

此外，要以编程方式创建更改请求，不能调用 `CreateOpsItem` API 操作，而应使用 `StartChangeRequestExecution` API 操作。但必须先批准更改请求，而不能立即运行，并且 Change Calendar 中不能有任何阻止性事件阻碍工作流运行。当收到批准并且日历未被阻止（或者已经获得绕过阻止性日历事件的权限）时，`StartChangeRequestExecution` 操作才能完成。

运行手册工作流

运行手册工作流是对您的云环境或本地环境中的目标资源进行所请求的更改的过程。每个更改请求都将指定一个自动化运行手册，用于进行所请求的更改。在获得所有所需的批准后，并且 Change Calendar 中不存在阻止性事件的情况下，运行手册工作流才能发生。如果已为特定日期和时间安排更改，即便已经获得所有批准并且日历未被阻止，运行手册工作流也将在到达计划时间后才会开始。

主题

- 设置 Change Manager
- 使用 Change Manager
- 审计和记录 Change Manager 活动
- 排查 Change Manager 问题

设置 Change Manager

您可以使用 Change Manager（AWS Systems Manager 中的一项工具）来管理对整个组织的更改（按照 AWS Organizations 中的配置）或对单个 AWS 账户的更改。

如果您将 Change Manager 用于组织，请从主题 为组织设置 Change Manager（管理账户） 开始，然后继续阅读 配置 Change Manager 选项和最佳实践。

如果您将 Change Manager 用于单个账户，则请直接阅读 配置 Change Manager 选项和最佳实践。

> **Note**
>
> 如果您首先将 Change Manager 用于某一单个账户，而后来又将该账户添加到允许 Change Manager 的组织单位，则您的单个账户设置将被忽略。

主题

- 为组织设置 Change Manager（管理账户）
- 配置 Change Manager 选项和最佳实践
- 为 Change Manager 配置角色和权限
- 控制对自动批准运行手册工作流的访问

为组织设置 Change Manager（管理账户）

如果要将 Change Manager（AWS Systems Manager 中的一项工具）用于在 AWS Organizations 中设置的组织，则本主题中的任务适用。如果您仅要将 Change Manager 用于单个 AWS 账户，则请跳至主题 配置 Change Manager 选项和最佳实践。

在充当 Organizations 内管理账户的 AWS 账户 中执行本节中的任务。有关管理账户和其他 Organizations 概念的信息，请参阅 AWS Organizations 术语和概念。

如果您需要打开 Organizations 并在继续操作之前将您的账户指定为管理账户，请参阅 AWS Organizations 用户指南中的创建和管理组织。

> **Note**
>
> 此设置过程无法在以下 AWS 区域中执行：
>
> - 欧洲（米兰）(eu-south-1)
> - 中东（巴林）(me-south-1)
> - 非洲（开普敦）(af-south-1)
> - 亚太地区（香港）(ap-east-1)
>
> 对于此过程，请确保您在管理账户中的其他区域工作。

在设置过程中，您要在Quick Setup（AWS Systems Manager 中的一项工具）中执行以下主要任务。

- 任务 1：为您的组织注册委托管理员账户

  使用 Change Manager 执行的、与更改相关的任务，在您的某一成员账户中进行管理，您将该账户指定为委托管理员账户。您为 Change Manager 注册的委托管理员账户，将成为您的所有 Systems Manager 操作的委托管理员账户。（您可以有多个用于其他 AWS 服务 的委托管理员账户。）您的 Change Manager 委托管理员账户（与管理账户不同）可以管理整个组织的更改活动，包括更改模板、更改请求，以及对每个更改模板和更改请求的批准。在委托管理员账户中，您还可以为 Change Manager 操作指定其他配置选项。

  > **Important**
  >
  > 委托管理员账户必须是其在 Organizations 中分配到的组织单位 (OU) 的唯一成员。

- 任务 2：为您要用于 Change Manager 操作的更改请求者角色或自定义工作职能定义和指定运行手册访问策略

  为在 Change Manager 中创建更改请求，则必须向您的成员账户中的用户授予 AWS Identity and Access Management (IAM) 权限，这些权限允许他们仅访问自动化运行手册，以及更改您选择提供给他们的模板。

  > **Note**
  >
  > 当用户创建更改请求时，他们首先需要选择更改模板。此更改模板可能会提供多个运行手册，但用户只能为每个更改请求选择一个运行手册。还可以将更改模板配置为允许用户在其请求中包含任何可用的运行手册。

  要授予所需权限，Change Manager 使用了工作职能的概念，IAM 也使用了此概念。但是，与 IAM 中的 AWS 工作职能托管策略不同，您既可以指定 Change Manager 工作职能的名称，也可以为这些工作职能指定 IAM 权限。

  在配置工作职能时，我们建议创建自定义策略，并仅提供执行更改管理任务所需的权限。例如，您可以根据定义的工作职能指定权限，以限制用户访问特定的运行手册集。

  例如，您可以创建名为 `DBAdmin` 的工作职能。对于此工作职能，您可以仅授予与 Amazon DynamoDB 数据库相关的运行手册所需的权限，例如 `AWS-CreateDynamoDbBackup` 和 `AWSConfigRemediation-DeleteDynamoDbTable`。

作为另一个示例，您可能希望仅授予某些用户使用与 Amazon Simple Storage Service (Amazon S3) 存储桶相关的运行手册所需的权限，例如 `AWS-ConfigureS3BucketLogging` 和 `AWSConfigRemediation-ConfigureS3BucketPublicAccessBlock`。

Change Manager 的 Quick Setup 中的配置过程还会为您提供一组完整 Systems Manager 管理权限，以应用于您创建的管理角色。

您部署的每一项 Change Manager Quick Setup 配置，都会在您的委托管理员账户中创建一项工作职能，这些工作职能拥有在您选择的组织单位中运行 Change Manager 模板和自动化运行手册的权限。您最多可以为 Change Manager 创建 15 项 Quick Setup 配置。

- 任务 3：选择您的组织中哪些成员账户能够使用 Change Manager

  您可以将 Change Manager 用于在 Organizations 中设置的所有组织单位内的所有成员账户，以及所有 AWS 区域（成员账户在其中进行操作）内的所有成员账户。如果愿意，您可以改为仅将 Change Manager 用于您的某些组织单位。

> **Important**
>
> 在开始此过程之前，我们强烈建议您认真阅读其步骤，以了解您所做的配置选择和所授予的权限。特别是应规划您要创建的自定义工作职能，以及您分配给各项工作职能的权限。这可确保稍后将您创建的工作职能策略附加到各个用户、用户组或 IAM 角色时，仅会向他们授予您希望他们拥有的权限。
>
> 作为最佳时间，首先使用 AWS 账户 管理员登录来设置委托管理员账户。然后，在创建更改模板并确定每个更改模板使用的运行手册之后，配置工作职能及其权限。

要设置 Change Manager 以便用于组织，请在 Systems Manager 控制台的 Quick Setup 区域中执行以下任务。

对于要为您的组织创建的每个工作职能，均应重复此任务。您创建的每项工作职能均可拥有针对一组不同组织单位的权限。

在 Organizations 管理账户中为 Change Manager 设置组织

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Quick Setup。

3. 在 Change Manager 选项卡上，选择 Create（创建）。
4. 对于 Delegated administrator account (委托管理员账户)，输入要用于在 Change Manager 中管理更改模板、更改请求和运行手册工作流的 AWS 账户 的 ID。

   如果您之前已为 Systems Manager 指定了委托管理员账户，则此字段中已报告其 ID。

   > **Important**
   >
   > 委托管理员账户必须是其在 Organizations 中分配到的组织单位 (OU) 的唯一成员。
   > 如果您注册的委托管理员账户后来从该角色注销，则系统会同时删除其管理 Systems Manager 操作的权限。请注意，您必须返回 Quick Setup，指定不同的委托管理员账户，然后再次指定所有工作职能和权限。
   > 如果您在整个组织中使用 Change Manager，我们建议始终从委托管理员账户进行更改。虽然您可以从组织中的其他账户进行更改，但这些更改不会在委派管理员账户中报告，也无法从该账户查看。

5. 在 Permissions to request and make changes (请求和进行更改的权限) 部分中，执行以下操作。

   > **Note**
   >
   > 您创建的每一项部署配置仅会为一项工作功能提供权限策略。您可以在以后返回到 Quick Setup，在创建要在操作中使用的更改模板时创建更多工作职能。

   创建管理角色 – 对于拥有所有 AWS 操作的 IAM 权限的管理员工作职能，请执行以下操作。

   > **Important**
   >
   > 授予用户完整管理权限应该谨慎行事，并且仅当用户的角色需要完整的 Systems Manager 访问权限时才能执行。有关 Systems Manager 访问权限的安全注意事项的重要信息，请参阅 对 AWS Systems Manager 进行身份和访问管理 和 Systems Manager 的安全最佳实践。

   1. 对于 Job function (工作职能)，请输入用于标识此角色及其权限的名称，例如 **`MyAWSAdmin`**。
   2. 对于 Role and permissions option (角色和权限选项)，请选择 Administrator permissions (管理员权限)。

创建其他工作职能 – 要创建非管理角色，请执行以下操作：

1. 对于 Job function (工作职能)，输入用于标识此角色的名称，并建议其权限。您选择的名称应能表示您将为其提供权限的运行手册的范围，例如 `DBAdmin` 或 `S3Admin`。
2. 对于 Role and permissions option (角色和权限选项)，请选择 Custom permissions (自定义权限)。
3. 在 Permissions policy editor (权限策略编辑器) 中，以 JSON 格式输入 IAM 权限，以向此工作职能授予这些权限。

> **Tip**
>
> 我们建议您使用 IAM policy 编辑器构建策略，然后将策略 JSON 粘贴到 Permissions policy（权限策略）字段中。

示例策略：DynamoDB 数据库管理

例如，您可以从策略内容开始，该内容为使用工作职能需要访问的 Systems Manager 文档（SSM 文档）提供权限。下面是一段示例策略内容，该内容将授予对与 DynamoDB 数据库相关的所有 AWS 托管式自动化运行手册和在位于美国东部（俄亥俄）区域 (`us-east-2`) 的示例 AWS 账户 `123456789012` 中创建的两个更改模板的访问权限。

该策略还包括对 StartChangeRequestExecution 操作的权限，在 Change Calendar 中创建更改请求时需要该权限。

> **Note**
>
> 此示例并不全面。可能需要额外的权限才能使用其他 AWS 资源，例如数据库和节点。

JSON

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

```
            "Effect": "Allow",
            "Action": [
                "ssm:CreateDocument",
                "ssm:DescribeDocument",
                "ssm:DescribeDocumentParameters",
                "ssm:DescribeDocumentPermission",
                "ssm:GetDocument",
                "ssm:ListDocumentVersions",
                "ssm:ModifyDocumentPermission",
                "ssm:UpdateDocument",
                "ssm:UpdateDocumentDefaultVersion"
            ],
            "Resource": [
                "arn:aws:ssm:region:*:document/AWS-CreateDynamoDbBackup",
                "arn:aws:ssm:region:*:document/AWS-AWS-DeleteDynamoDbBackup",
                "arn:aws:ssm:region:*:document/AWS-
DeleteDynamoDbTableBackups",
                "arn:aws:ssm:region:*:document/AWSConfigRemediation-
DeleteDynamoDbTable",
                "arn:aws:ssm:region:*:document/AWSConfigRemediation-
EnableEncryptionOnDynamoDbTable",
                "arn:aws:ssm:region:*:document/AWSConfigRemediation-
EnablePITRForDynamoDbTable",
                "arn:aws:ssm:region:111122223333:document/
MyFirstDBChangeTemplate",
                "arn:aws:ssm:region:111122223333:document/
MySecondDBChangeTemplate"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "ssm:ListDocuments",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ssm:StartChangeRequestExecution",
            "Resource": "arn:aws:ssm:region:111122223333:automation-
definition/*:*"
        }
    ]
}
```

有关 IAM policy 的更多信息，请参阅《IAM 用户指南》中的 AWS 资源的访问权限管理和创建 IAM policy。

6. 在 Targets (目标) 部分中，请选择是将对您创建的工作职能的权限授予整个组织，还是仅授予部分组织单位。

   如果选择 Entire organization (整个组织)，请继续执行步骤 9。

   如果选择 Custom (自定义)，则请继续执行步骤 8。

7. 在 Target OUs (目标 OU) 部分中，选中要使用 Change Manager 的组织单位的复选框。

8. 选择 Create（创建）。

当系统为您的组织完成 Change Manager 设置后，它将显示您的部署摘要。此摘要信息包含为您配置的工作职能创建的角色的名称。例如 `AWS-QuickSetup-SSMChangeMgr-DBAdminInvocationRole`。

> **Note**
>
> Quick Setup 将使用 AWS CloudFormation StackSets 来部署您的配置。您还可以查看有关 AWS CloudFormation 控制台中已完成的部署配置的信息。有关 StackSets 的信息，请参阅 AWS CloudFormation 用户指南中的使用 AWS CloudFormation StackSets。

下一步是配置其他 Change Manager 选项。您可以在您的委托管理员账户或您允许使用 Change Manager 的组织单位内的任何账户中完成此任务。您可以配置多个选项，例如选择用户身份管理选项、指定哪些用户可以审核及批准或拒绝更改模板和更改请求，以及选择允许组织使用哪些最佳实践选项。有关信息，请参阅配置 Change Manager 选项和最佳实践。

配置 Change Manager 选项和最佳实践

无论是在整个组织中还是在单个 AWS 账户中使用 Change Manager（AWS Systems Manager 中的一项工具），都必须执行本节中的任务。

如果您将 Change Manager 用于组织，可以在您的委托管理员账户或您允许使用 Change Manager 的组织单位内的任何账户中执行以下任务。

主题

- 任务 1：配置 Change Manager 用户身份管理和模板审核人员
- 任务 2：配置 Change Manager 更改冻结事件审批人员和最佳实践
- 为 Change Manager 通知配置 Amazon SNS 主题

任务 1：配置 Change Manager 用户身份管理和模板审核人员

在首次访问 Change Manager 时，请执行此过程中的该任务。您可以在以后更新这些配置设置，方法是返回到 Change Manager，然后选择 Settings (设置) 选项卡上的 Edit (编辑)。

配置 Change Manager 用户身份管理和模板审核人员

1. 登录到 AWS Management Console。

   如果您将 Change Manager 用于组织，请使用您的委托管理员账户的凭证登录。用户必须拥有更新 Change Manager 设置的必要 AWS Identity and Access Management（IAM）权限。
2. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
3. 在导航窗格中，请选择 Change Manager。
4. 在服务主页上，根据可用选项，执行以下操作之一：

   - 如果您将 Change Manager 用于 AWS Organizations，请选择 Set up delegated account (设置委托账户)。
   - 如果您将 Change Manager 用于单个 AWS 账户，请选择 Set up Change Manager (设置 Change Manager)。

     - 或者 -

     选择 Create sample change request (创建示例更改请求)、Skip (跳过)，然后选择 Settings (设置) 选项卡。
5. 对于 User identity management (用户身份管理)，请选择以下选项之一。

   - AWS Identity and Access Management（IAM）- 确定提出和批准请求以及使用您的现有用户、组和角色在 Change Manager 中执行其他操作的用户的身份。
   - AWS IAM Identity Center（IAM Identity Center）– 允许 IAM Identity Center 创建和管理身份，或连接到您的现有身份源，以确定在 Change Manager 中执行操作的用户的身份。
6. 在 Template reviewer notification（模板审核人员通知）部分，指定 Amazon Simple Notification Service (Amazon SNS) 主题，这些主题将要用于通知模板审核人员新的更改模板或更改模板版本

已准备就绪可供审核。请确保已将您选择的 Amazon SNS 主题配置为向您的模板审核人员发送通知。

有关为更改模板审核人员通知创建和配置 Amazon SNS 主题的信息，请参阅 为 Change Manager 通知配置 Amazon SNS 主题。

1. 要为模板审核人员通知指定 Amazon SNS 主题，请选择以下选项之一：
   - Enter an SNS Amazon Resource Name (ARN) (输入 SNS Amazon 资源名称 (ARN)) – 对于 Topic ARN (主题 ARN)，请输入现有 Amazon SNS 主题的 ARN。此主题可以位于您组织的任何账户中。
   - Select an existing SNS topic（选择现有 SNS 主题）– 对于 Target notification topic（设定通知主题目标），选择您当前的 AWS 账户中现有 Amazon SNS 主题的 ARN。（如果您尚未在当前的 AWS 账户 和 AWS 区域 中创建任何 Amazon SNS 主题，则此选项不可用。）

   > **Note**
   >
   > 您选择的 Amazon SNS 主题必须被配置为指定它所发送的通知，以及接收这些通知的订阅者。其访问策略还必须向 Systems Manager 授予权限，以便 Change Manager 可以发送通知。有关信息，请参阅为 Change Manager 通知配置 Amazon SNS 主题。

2. 选择 Add notification（添加通知）。

7. 在 Change template reviewers (更改模板审核人员) 部分中，在您的组织或账户选择用户审核新的更改模板或更改模板版本，然后才能在操作中使用它们。

   更改模板审核人员负责验证其他用户提交的、要在 Change Manager 运行手册工作流中使用的模板的适用性和安全性。

   可以通过执行以下操作选择更改模板审核人员：

   1. 选择 Add (添加)。
   2. 选中您要指定为更改模板审核人员的每个用户、组或 IAM 角色名称旁边的复选框。
   3. 选择 Add approvers (添加审批人员)。

8. 选择 Submit (提交)。

完成此初始设置过程后，请按 任务 2：配置 Change Manager 更改冻结事件审批人员和最佳实践 中的步骤操作，配置其他 Change Manager 设置和最佳实践。

任务 2：配置 Change Manager 更改冻结事件审批人员和最佳实践

在您完成 任务 1：配置 Change Manager 用户身份管理和模板审核人员 中的步骤之后，可为更改冻结事件期间的更改请求指定额外的审核人员，还可指定您希望允许哪些可用的最佳实践用于 Change Manager 操作。

更改冻结事件意味着在当前更改日历中设置限制（AWS Systems Manager Change Calendar 中的日历状态为 CLOSED）。在这些情况下，除了更改请求的常规审批人员以外，或者如果更改请求是使用允许自动批准的模板创建的，更改冻结审批人员必须为此更改请求的运行授予权限。如果他们不这样做，则不会处理该更改，直到日历状态再次为 OPEN 时为止。

配置 Change Manager 更改冻结事件审批人员和最佳实践

1. 在导航窗格中，请选择 Change Manager。
2. 选择 Settings (设置) 选项卡，然后选择 Edit (编辑)。
3. 在 Approvers for change freeze events (更改冻结事件的审批人员) 部分中，选择您的组织或账户中可以批准更改即便在 Change Calendar 中使用的日历当前处于 CLOSED (已关闭) 状态也能运行的用户。

   > **Note**
   >
   > 要允许更改冻结审核，必须打开 Best practices (最佳实践) 中的 Check Change Calendar for restricted change events (检查更改日历中的受限更改事件) 选项。

   通过执行以下操作，选择更改冻结事件的审批人员：

   1. 选择添加。
   2. 选中您要指定为更改冻结事件审批人员的每个用户、组或 IAM 角色名称旁边的复选框。
   3. 选择 Add approvers (添加审批人员)。
4. 在 Best practices (最佳实践) 部分中，启用您要针对以下各个选项强制实施的最佳实践。

   - 选项：Check Change Calendar for restricted change events (检查更改日历中的受限更改事件)

     要指定 Change Manager 检查 Change Calendar 中的日历，以确保更改不会已被计划的事件阻止，请首先选中 Enabled (已启用) 复选框，然后从 Change Calendar (更改日历) 列表中选择要检查受限事件的日历。

     有关 Change Calendar 的更多信息，请参阅 AWS Systems Manager Change Calendar。

- 选项：SNS topic for approvers for closed events (用于已关闭事件的审批人员的 SNS 主题)

  1. 选择以下选项之一，在您的账户中指定 Amazon Simple Notification Service (Amazon SNS) 主题，用于在更改冻结事件期间向审批人员发送通知。（请注意，您还必须在 Best practices (最佳实践) 上方的 Approvers for change freeze events (更改冻结事件的审批人员) 部分中指定审批人员。）

     - Enter an SNS Amazon Resource Name (ARN) (输入 SNS Amazon 资源名称 (ARN)) – 对于 Topic ARN (主题 ARN)，请输入现有 Amazon SNS 主题的 ARN。此主题可以位于您组织的任何账户中。
     - Select an existing SNS topic（选择现有 SNS 主题）– 对于 Target notification topic（设定通知主题目标），选择您当前的 AWS 账户中现有 Amazon SNS 主题的 ARN。（如果您尚未在当前的 AWS 账户 和 AWS 区域 中创建任何 Amazon SNS 主题，则此选项不可用。）

     > **Note**
     >
     > 您选择的 Amazon SNS 主题必须被配置为指定它所发送的通知，以及接收这些通知的订阅者。其访问策略还必须向 Systems Manager 授予权限，以便 Change Manager 可以发送通知。有关信息，请参阅为 Change Manager 通知配置 Amazon SNS 主题。

  2. 选择 Add notification（添加通知）。

- 选项：Require monitors for all templates (所有模板都需要监控器)

  如果您希望确保您的组织或账户的所有模板都指定 Amazon CloudWatch 告警来监控您的更改操作，请选中 Enabled（已启用）复选框。

- 选项：Require template review and approval before use (使用前需要审核和批准模板)

  要确保在不基于已审核和已批准的模板的情况下，不创建任何更改请求，也不运行任何运行手册工作流，请选中 Enabled (已启用) 复选框。

5. 选择保存。

为 Change Manager 通知配置 Amazon SNS 主题

您可以将 Change Manager（AWS Systems Manager 中的一项工具）配置为针对与更改请求和更改模板有关的事件向 Amazon Simple Notification Service（Amazon SNS）主题发送通知。要接收针对您向其添加了主题的 Change Manager 事件的通知，请完成以下任务。

主题

- 任务 1：创建并订阅 Amazon SNS 主题
- 任务 2：更新 Amazon SNS 访问策略
- 任务 3：（可选）更新 AWS Key Management Service 访问策略

任务 1：创建并订阅 Amazon SNS 主题

首先，必须创建一个 Amazon SNS 主题并订阅此主题。有关更多信息，请参阅 Amazon Simple Notification Service 开发人员指南中的创建 Amazon SNS 主题和订阅 Amazon SNS 主题。

> **Note**
>
> 要接收通知，您必须指定与委托管理员账户位于相同 AWS 区域 和 AWS 账户 的 Amazon SNS 主题的 Amazon Resource Name (ARN)。

任务 2：更新 Amazon SNS 访问策略

请使用以下过程更新 Amazon SNS 访问策略，以便 Systems Manager 能将 Change Manager 通知发布到您在“任务 1”中创建的 Amazon SNS 主题。如果不完成这项任务，Change Manager 将没有权限针对您为其添加主题的事件发送通知。

1. 访问 https://console.aws.amazon.com/sns/v3/home，登录 AWS Management Console 并打开 Amazon SNS 控制台。
2. 在导航窗格中，选择 Topics (主题)。
3. 选择您在“任务 1”中创建的主题，然后选择 Edit (编辑)。
4. 展开 Access policy (访问策略)。
5. 将以下 `Sid` 数据块添加并更新到现有策略中，并将每个*#######*替换为您的信息。

```
{
    "Sid": "Allow Change Manager to publish to this topic",
    "Effect": "Allow",
    "Principal": {
        "Service": "ssm.amazonaws.com"
    },
    "Action": "sns:Publish",
    "Resource": "arn:aws:sns:region:account-id:topic-name",
    "Condition": {
```

```
        "StringEquals": {
            "aws:SourceAccount": [
                "account-id"
            ]
        }
    }
}
```

在现有 Sid 数据块后输入此数据块，然后针对您创建的主题，将 *region*、*account-id* 和 *topic-name* 替换为适当的值。

6. 选择保存更改。

现在，当您添加到主题中的事件类型发生时，系统将向 Amazon SNS 主题发送通知。

> **Important**
>
> 如果您已使用 AWS Key Management Service (AWS KMS) 服务器端加密密钥配置了 Amazon SNS 主题，则必须完成“任务 3”。

任务 3：（可选）更新 AWS Key Management Service 访问策略

如果您为 Amazon SNS 主题启用了 AWS Key Management Service (AWS KMS) 服务器端加密，则还必须更新您在配置该主题时选择的 AWS KMS key 的访问策略。请使用以下过程更新访问策略，以便 Systems Manager 能将 Change Manager 批准通知发布到您在“任务 1”中创建的 Amazon SNS 主题。

1. 访问 https://console.aws.amazon.com/kms，打开 AWS KMS 控制台。
2. 在导航窗格中，选择 Customer managed keys (客户托管密钥)。
3. 选择您在创建主题时选择的客户托管密钥的 ID。
4. 在 Key Policy (密钥策略) 部分中，选择 Switch to policy view (切换到策略视图)。
5. 选择编辑。
6. 在现有策略中的现有 Sid 块之一之后输入以下 Sid 块。将每个*#######*替换为您自己的信息。

```
{
    "Sid": "Allow Change Manager to decrypt the key",
    "Effect": "Allow",
    "Principal": {
        "Service": "ssm.amazonaws.com"
```

```
    },
    "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey*"
    ],
    "Resource": "arn:aws:kms:region:account-id:key/key-id",
    "Condition": {
        "StringEquals": {
            "aws:SourceAccount": [
                "account-id"
            ]
        }
    }
}
```

7. 现在，在资源策略中现有的 Sid 块之一之后输入以下 Sid 块，以帮助防止跨服务混淆代理问题。

   此块使用 aws:SourceArn 和 aws:SourceAccount 全局条件上下文键来限制 Systems Manager 向资源提供其他服务的权限。

   将每个*#######*替换为您自己的信息。

   JSON

```
{
    "Version": "2008-10-17",
    "Statement": [
        {
            "Sid": "Configure confused deputy protection for AWS KMS keys
 used in Amazon SNS topic when called from Systems Manager",
            "Effect": "Allow",
            "Principal": {
                "Service": "ssm.amazonaws.com"
            },
            "Action": [
                "sns:Publish"
            ],
            "Resource": "arn:aws:sns:us-east-1:111122223333:topic-name",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:ssm:us-east-1:111122223333:*"
                },
                "StringEquals": {
```

```
                    "aws:SourceAccount": "111122223333"
                }
            }
        }
    ]
}
```

8. 选择保存更改。

为 Change Manager 配置角色和权限

默认情况下，Change Manager 没有对您的资源执行操作的权限。您必须使用 AWS Identity and Access Management (IAM) 服务角色或担任角色授予访问权限。该角色可以让 Change Manager 代表您安全地运行在批准的更改请求中指定的运行手册工作流。该角色向 AWS Security Token Service (AWS STS) AssumeRole 授予对 Change Manager 的信任。

通过向角色提供这些权限以代表企业中的用户行事，用户本身不需要被授予该权限数组。权限允许的操作仅限于批准的操作。

当您的账户或企业中的用户创建更改请求时，他们可以选择此担任角色来执行更改操作。

您可以为 Change Manager 创建新的担任角色，也可以利用所需的权限更新现有角色。

如果您需要为 Change Manager 创建一个服务角色，请完成以下任务。

任务

- 任务 1：为 Change Manager 创建担任角色策略
- 任务 2：为 Change Manager 创建担任角色
- 任务 3：将 iam:PassRole 策略附加到其他角色
- 任务 4：向担任角色添加内联策略以调用其他 AWS 服务
- 任务 5：配置用户对 Change Manager 的访问权限

任务 1：为 Change Manager 创建担任角色策略

使用以下过程创建将附加到您的 Change Manager 担任角色的策略。

为 Change Manager 创建担任角色策略

1. 访问：https://console.aws.amazon.com/iam/，打开 IAM 控制台。

2. 在导航窗格中选择 Policies，然后选择 Create Policy。
3. 在 Create policy（创建策略）页面上，选择 JSON 选项卡，将默认内容替换为以下内容，您将在以下步骤中针对自己的 Change Manager 操作对其进行修改。

> **Note**
>
> 如果要创建用于单个 AWS 账户 的策略，而不是具有多个账户和 AWS 区域 的企业，则可以省略第一个数据块。对于使用 Change Manager 的单个账户，不需要 `iam:PassRole` 权限。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::111122223333:role/AWS-
SystemsManager-job-functionAdministrationRole",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ssm.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeDocument",
                "ssm:GetDocument",
                "ssm:StartChangeRequestExecution"
            ],
            "Resource": [
                "arn:aws:ssm:us-east-1:111122223333:automation-
definition/template-name:$DEFAULT",
                "arn:aws:ssm:us-east-1::document/template-name"
            ]
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "ssm:ListOpsItemEvents",
                "ssm:GetOpsItem",
                "ssm:ListDocuments",
                "ssm:DescribeOpsItems"
            ],
            "Resource": "*"
        }
    ]
}
```

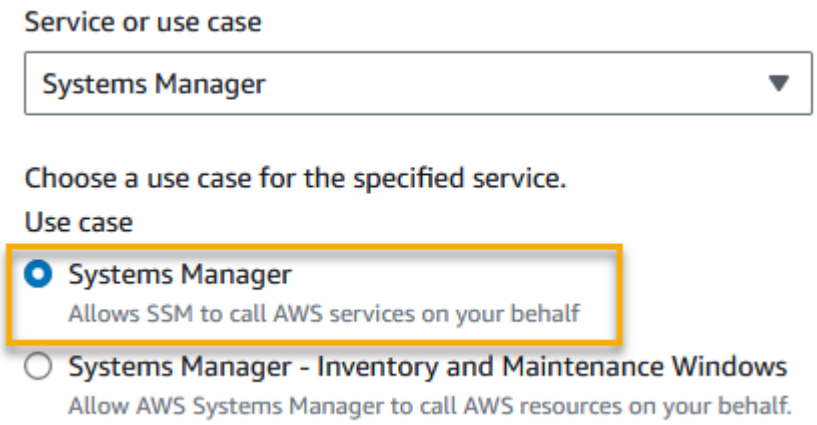
4. 对于 `iam:PassRole` 操作，更新 `Resource` 值以包括为您的企业定义的所有工作职能的 ARN，您希望对这些工作职能授予启动运行手册工作流的权限。
5. 将 *region*、*account-id*、*template-name*、*delegated-admin-account-id* 和 *job-function* 占位符替换为 Change Manager 操作的值。
6. 对于第二个 `Resource` 语句，修改列表以包括要授予权限的所有更改模板。或者，指定 `"Resource": "*"` 为企业中的所有更改模板授予权限。
7. 选择下一步：标签。
8. （可选）添加一个或多个标签键值对，以组织、跟踪或控制此策略的访问权限。
9. 选择下一步：审核。
10. 在 Review policy（查看策略）页面上的 Name（名称）框中输入一个名称，例如 **`MyChangeManagerAssumeRole`**，然后输入可选描述。
11. 选择 Create policy（创建策略），然后继续转至 任务 2：为 Change Manager 创建担任角色。

任务 2：为 Change Manager 创建担任角色

使用以下过程为 Change Manager 创建 Change Manager 担任角色（一种服务角色类型）。

为 Change Manager 创建担任角色

1. 通过 https://console.aws.amazon.com/iam/ 打开 IAM 控制台。
2. 在导航窗格中，选择角色，然后选择创建角色。
3. 对于 Select trusted entity（选择可信实体），完成以下选择：
    1. 对于可信实体类型，选择 AWS 服务。
    2. 对于其他 AWS 服务 的应用场景，选择 Systems Manager
    3. 选择 Systems Manager，如下图所示。



4. 选择下一步。
5. 在 Attached permissions policy（附加的权限策略）页面上，搜索您在 任务 1：为 Change Manager 创建担任角色策略 中创建的担任角色策略，比如 **MyChangeManagerAssumeRole**。
6. 选中担任角色策略名称旁边的复选框，然后选择 Next: Tags（下一步：标签）。
7. 对于Role name（角色名称），请输入新实例配置文件的名称，例如 **MyChangeManagerAssumeRole**。
8. （可选）对于 Description（描述），更新该实例角色的描述。
9. （可选）添加一个或多个标签键值对，以组织、跟踪或控制此角色的访问权限。
10. 选择下一步：审核。
11. （可选）对于 Tags（标签），添加一个或多个标签键值对，以组织、跟踪或控制此角色的访问，然后选择 Create role（创建角色）。系统将让您返回到 角色 页面。
12. 选择 Create role (创建角色)。系统将让您返回到 角色 页面。
13. 在角色页面中，选择刚刚创建的角色以打开摘要页面。

任务 3：将 **iam:PassRole** 策略附加到其他角色

使用以下过程将 `iam:PassRole` 策略附加到 IAM 实例配置文件或 IAM 服务角色。（Systems Manager 服务使用 IAM 实例配置文件与 EC2 实例进行通信。对于混合和多云环境中的非 EC2 托管式节点，改用 IAM 服务角色。）

通过附加 `iam:PassRole` 策略，Change Manager 服务可以在运行运行手册工作流时将担任角色权限传递给其他服务或 Systems Manager 工具。

将 **iam:PassRole** 策略附加到 IAM 实例配置文件或服务角色

1. 通过以下网址打开 IAM 控制台：https://console.aws.amazon.com/iam/。

2. 在导航窗格中，选择角色。
3. 搜索您创建的 Change Manager 担任角色（如 **MyChangeManagerAssumeRole**），然后选择其名称。
4. 在担任角色的 Summary（摘要）页面中，选择 Permissions（权限）选项卡。
5. 选择 Add permissions, Create inline policy（添加权限、创建内联策略）。
6. 在 Create policy (创建策略) 页面上，选择 Visual editor (可视化编辑器) 选项卡。
7. 选择服务，然后选择 IAM。
8. 在 Filter actions（筛选操作）文本框中，输入 **PassRole**，然后选择 PassRole 选项。
9. 展开 Resources（资源）。确保已选择特定，然后选择添加 ARN。
10. 在 Specify ARN for role（为角色指定 ARN）字段中，输入要向其传递担任角色权限的 IAM 实例配置文件角色或 IAM 服务角色的 ARN。系统会自动填充账户和具有路径的角色名称字段。
11. 选择 Add (添加)。
12. 选择查看策略。
13. 对于 Name（名称），输入一个名称来标识此策略，然后选择 Create policy（创建策略）。

更多信息

- 配置 Systems Manager 所需的实例权限
- 在混合和多云环境中创建 Systems Manager 所需的 IAM 服务角色

任务 4：向担任角色添加内联策略以调用其他 AWS 服务

当更改请求使用 Change Manager 代入角色调用其他 AWS 服务时，必须为代入角色配置调用这些服务的权限。此要求适用于可能在更改请求中使用的所有 AWS 自动化运行手册（AWS-* 运行手册），例如 `AWS-ConfigureS3BucketLogging`、`AWS-CreateDynamoDBBackup` 和 `AWS-RestartEC2Instance` 运行手册。对于您创建的任何自定义运行手册，如果这些文档使用调用其他服务的操作来调用其他 AWS 服务，则此要求同样适用。例如，如果您使用 `aws:executeAwsApi`、`aws:CreateStack` 或 `aws:copyImage` 操作，则您必须配置具有权限的服务角色来调用这些服务。您可以通过将 IAM 内联策略添加到角色以向其他 AWS 服务授予权限。

将内联策略添加到代入角色以调用其他 AWS 服务（IAM 控制台）

1. 登录 AWS Management Console，然后通过以下网址打开 IAM 控制台：https://console.aws.amazon.com/iam/。

2. 在导航窗格中，选择角色。
3. 在列表中，选择要更新的担任角色的名称，例如 MyChangeManagerAssumeRole。
4. 选择 Permissions（权限）选项卡。
5. 选择 Add permissions, Create inline policy（添加权限、创建内联策略）。
6. 选择 JSON 选项卡。
7. 输入您希望调用的 AWS 服务的 JSON 策略文档。以下是两个示例 JSON 策略文档。

   Simple Storage Service (Amazon S3) **PutObject** 和 **GetObject** 示例

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
    ]
}
```

   Amazon EC2 **CreateSnapshot** 和 **DescribeSnapShots** 示例

   JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":"ec2:CreateSnapshot",
         "Resource":"*"
      },
```

```
        {
            "Effect":"Allow",
            "Action":"ec2:DescribeSnapshots",
            "Resource":"*"
        }
    ]
}
```

有关 IAM policy 语言的详细信息，请参阅《IAM 用户指南》中的 IAM JSON 策略参考。

8. 完成后，选择查看策略。策略验证程序将报告任何语法错误。
9. 对于 Name（名称），输入一个名称来标识您创建的策略。查看策略摘要以查看您的策略授予的权限。然后，选择创建策略以保存您的工作。
10. 创建内联策略后，它会自动嵌入您的角色。

任务 5：配置用户对 Change Manager 的访问权限

如果已为您的用户、组或角色分配了管理员权限，则您有权访问 Change Manager。如果您没有管理员权限，则管理员必须为您的用户、组或角色分配 `AmazonSSMFullAccess` 托管策略或提供类似权限的策略。

使用以下过程配置用户，以使用 Change Manager。您选择的用户将拥有配置并运行 Change Manager 的权限。

根据您在组织中使用的身份应用程序，您可以选择三个可用选项中的任何一个来配置用户访问权限。在配置用户访问权限时，请分配或添加以下内容：

1. 分配提供访问 Systems Manager 权限的 `AmazonSSMFullAccess` 策略或类似策略。
2. 分配 `iam:PassRole` 策略。
3. 为您在 任务 2：为 Change Manager 创建担任角色 末尾复制的 Change Manager 担任角色添加 ARN。

要提供访问权限，请为您的用户、组或角色添加权限：

- AWS IAM Identity Center 中的用户和群组：

  创建权限集合。按照《AWS IAM Identity Center 用户指南》中创建权限集的说明进行操作。
- 通过身份提供商在 IAM 中托管的用户：

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

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

您已完成了为 Change Manager 配置所需的角色。您现在可以在 Change Manager 操作中使用 Change Manager 担任角色 ARN。

控制对自动批准运行手册工作流的访问

在为您的组织或账户创建的每个更改模板中，您可以指定从该模板创建的更改请求是否可以作为自动批准的更改请求运行，这意味着这些更改请求将在没有审核步骤的情况下自动运行（更改冻结事件除外）。

但是，您可能希望阻止某些用户、组或 AWS Identity and Access Management (IAM) 角色运行自动批准的更改请求，即使更改模板允许此操作也是如此。您可以将 `ssm:AutoApprove` 条件键用于分配给用户、组或 IAM 角色的 IAM policy 中的 `StartChangeRequestExecution` 操作，借此实现上述目标。

您可以将以下策略添加为内联策略，其中将该条件指定为 `false`，以阻止用户运行可自动批准的更改请求。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
            {
            "Effect": "Allow",
            "Action": "ssm:StartChangeRequestExecution",
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "ssm:AutoApprove": "false"
                }
            }
```

```
        }
    ]
}
```

有关指定内联策略的信息，请参阅 IAM 用户指南中的内联策略及添加和删除 IAM 身份权限。

有关 Systems Manager 策略条件键的更多信息，请参阅 Systems Manager 的条件键。

使用 Change Manager

借助 Change Manager（AWS Systems Manager 中的一项工具），整个组织或单个 AWS 账户中的（已获必要权限）用户可以执行与更改相关的任务。Change Manager 任务如下：

- 创建、审核和批准或拒绝更改模板。

  更改模板是 Change Manager 中的一个配置设置集合，它定义了所需的批准、可用的运行手册和更改请求的通知选项等内容。
- 创建、审核和批准或拒绝更改请求。

  更改请求是 Change Manager 中的一个请求，旨在运行更新您的 AWS 或本地环境中一个或多个资源的自动化运行手册。更改请求是使用更改模板创建的。
- 可以指定组织或账户中的哪些用户可以成为更改模板和更改请求的审核人员。
- 可以编辑配置设置，例如如何在 Change Manager 中管理用户身份，以及在您的 Change Manager 操作中强制实施哪些可用的最佳实践选项。有关配置这些设置的信息，请参阅 配置 Change Manager 选项和最佳实践。

主题

- 使用更改模板
- 使用更改请求
- 审核更改请求详细信息、任务和时间表（控制台）
- 查看更改请求的聚合计数（命令行）

使用更改模板

更改模板是 Change Manager 中的一个配置设置集合，它定义了所需的批准、可用的运行手册和更改请求的通知选项等内容。

> **Note**
>
> AWS 提供了一个示例 Hello World 更改模板，您可使用该示例来试用 Change Manager（AWS Systems Manager 中的一项工具）。但是，您可以创建自己的更改模板，来定义您要允许对组织或账户中的资源进行的更改。

在运行手册工作流运行时进行的更改基于自动化运行手册的内容。在您创建的每个更改模板中，都可以包含一个或多个自动化运行手册，提出更改请求的用户可以从中选择这些运行手册，以便在更新过程中运行。您还可以创建允许请求者为更改请求选择任何可用的自动化运行手册的更改模板。

要创建更改模板，可以使用 Create template (创建模板) 控制台页面中的 Builder (生成器) 选项，来构建更改模板。或者，使用 Editor (编辑器) 选项，可以借助您要用于运行手册工作流的配置手动编写 JSON 或 YAML 内容。您还可以使用命令行工具创建更改模板，将该更改模板的 JSON 内容存储在外部文件中。

主题

- 试用 AWS 托管 Hello World 更改模板
- 创建更改模板
- 审核和批准或拒绝更改模板
- 删除更改模板

试用 AWS 托管 **Hello World** 更改模板

您可以使用示例更改模板 `AWS-HelloWorldChangeTemplate`（该模板使用示例 Automation 运行手册 `AWS-HelloWorld`），在完成 Change Manager（AWS Systems Manager 中的一项工具）的设置后，对审核和批准流程进行测试。此模板用于测试或验证已配置的权限、审批人员指定和批准流程。AWS 已经批准在您的组织或账户中使用此更改模板。但是，基于此更改模板的任何更改请求，仍然必须得到您的组织或账户中的审核人员的批准。

与此模板关联的运行手册工作流的结果，不会对资源进行更改，而是在“自动化”步骤的输出中打印一条消息。

开始前的准备工作

在开始之前，请确保您已完成以下任务：

- 如果您使用 AWS Organizations 管理整个组织中的更改，请完成 为组织设置 Change Manager（管理账户） 中描述的组织设置任务。
- 按照 配置 Change Manager 选项和最佳实践中的描述，为您的委托管理员账户或单个账户配置 Change Manager。

> **Note**
>
> 如果您在 Change Manager 设置中打开了最佳实践选项 Require monitors for all templates (所有模板都需要监控器)，请在测试 Hello World 更改模板时暂时关闭该选项。

试用 AWS 托管 Hello World 更改模板

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Change Manager。
3. 选择 Create request (创建请求)。
4. 选择名为 `AWS-HelloWorldChangeTemplate` 的更改模板，然后选择 Next (下一步)。
5. 对于 Name (名称)，请输入更改请求的名称，以便于识别其用途，例如 **`MyChangeRequestTest`**。
6. 有关创建更改请求的其余步骤，请参阅 创建更改请求。

后续步骤

有关批准更改请求的信息，请参阅 审核和批准或拒绝更改请求。

要查看您的更改请求的状态和结果，请在 Change Manager 中的 Requests (请求) 选项卡上选择您的更改请求的名称。

创建更改模板

更改模板是 Change Manager 中的一个配置设置集合，它定义了所需的批准、可用的运行手册和更改请求的通知选项等内容。

您可以使用控制台（内含“生成器”和“编辑器”选项）或命令行工具，在 Change Manager（AWS Systems Manager 中的一项工具）中为操作创建更改模板。

主题

- 关于更改模板中的批准
- 使用 Builder (生成器) 创建更改模板
- 使用 Editor (编辑器) 创建更改模板
- 使用命令行工具创建更改模板

关于更改模板中的批准

对于您创建的每个更改模板，您最多可以为通过该更改模板创建的更改请求指定五个批准级别。对于其中每个级别，您最多可以指定五个潜在审批者。审批者并不限于单个用户。您还可以将 IAM 组或 IAM 角色指定为单独的审批者。对于 IAM 组和 IAM 角色，属于该组或角色的一个或多个用户可以提供批准，以获得更改请求所需的批准总数。您还可以指定数量超过您的更改模板所需数量的审批者。

Change Manager 支持两种主要的批准方法：逐级批准和逐行批准。在某些情况下，也可以将这两种类型组合在一起。我们建议在您的 Change Manager 操作中仅使用逐级批准。

Per-level approvals

推荐。自 2023 年 1 月 23 日起，Change Manager 支持逐级批准。在此模型中，对于更改模板中的每个批准级别，您首先需要指定该级别需要多少次批准。然后，您需要为该级别至少指定对应数量的审批者，并且可以指定更多审批者。但只有您指定数量的逐级审批者才需要批准更改请求。例如，您可以指定五个审批者，但需要三次批准。

有关此批准类型的控制台视图和 JSON 示例，请参阅 the section called “逐级批准配置示例”。

Per-line approvals

支持向后兼容性。原始版本的 Change Manager 仅支持逐行批准。在此模型中，为批准级别指定的每个审批者都以一个批准行来表示。每个审批者都必须批准一次更改请求，该更改请求才能在该级别获得批准。在 2023 年 1 月 23 日之前，这是唯一受支持的批准模型。在此日期之前创建的更改模板继续支持逐行批准，但我们建议改用逐级批准。

有关此批准类型的控制台视图和 JSON 示例，请参阅 the section called “逐行批准配置示例”。

Combined per-line and per-level approvals

不推荐。在控制台中，构建器选项卡不再支持添加逐行批准。但在某些情况下，您最终可能会在更改模板中同时包含逐行和逐级批准。如果您更新在 2023 年 1 月 23 日之前创建的更改模板，或者通过手动编辑更改模板的 YAML 内容来创建或更新更改模板，则可能会发生这种情况。

有关此批准类型的控制台视图和 JSON 示例，请参阅 the section called “逐级和逐行组合批准配置示例”。

> **Important**
>
> 虽然可以创建将逐行和逐级批准组合在一起的更改模板，但不建议或不必使用此配置。无论哪种批准类型，需要较多批准者（逐行或逐级批准）优先。例如：
>
> - 如果更改模板指定了三次逐级批准，但指定了五次逐行批准，则需要五次批准。
> - 如果更改模板指定了四次逐级批准，但指定了两次逐行批准，则需要四次批准。

您可以通过手动编辑 YAML 或 JSON 内容，来创建同时包括逐行和逐级批准的级别。然后，构建器选项卡将显示用于为该级别和单个行指定所需批准数量的控件。但您使用控制台添加的新级别仍然仅支持逐级批准配置。

更改请求通知和拒绝

Amazon SNS 通知

在使用您的更改模板创建更改请求后，将向已为该级别的批准通知指定的 Amazon Simple Notification Service（Amazon SNS）主题的订阅用户发送通知。您可以在更改模板中指定通知主题，也可以允许创建更改请求的用户指定通知主题。

在某一级别获得最低所需批准数量后，将向订阅下一级别 Amazon SNS 主题的审批者发送通知，依此类推。

> **Important**
>
> 确保您指定的 IAM 角色、组和用户共同提供足够的审批者，以满足您指定的所需批准数量。例如，如果您仅将一个包含三个用户的 IAM 组指定为审批者，则您无法指定在该级别必须获得五次批准，只能指定三次或以下。

更改请求拒绝

无论您指定了多少个批准级别和审批者，只需拒绝一次更改请求，就会阻止执行该请求的运行手册工作流。

Change Manager 批准类型示例

以下示例演示了 Change Manager 中三种批准类型的控制台视图和 JSON 内容。

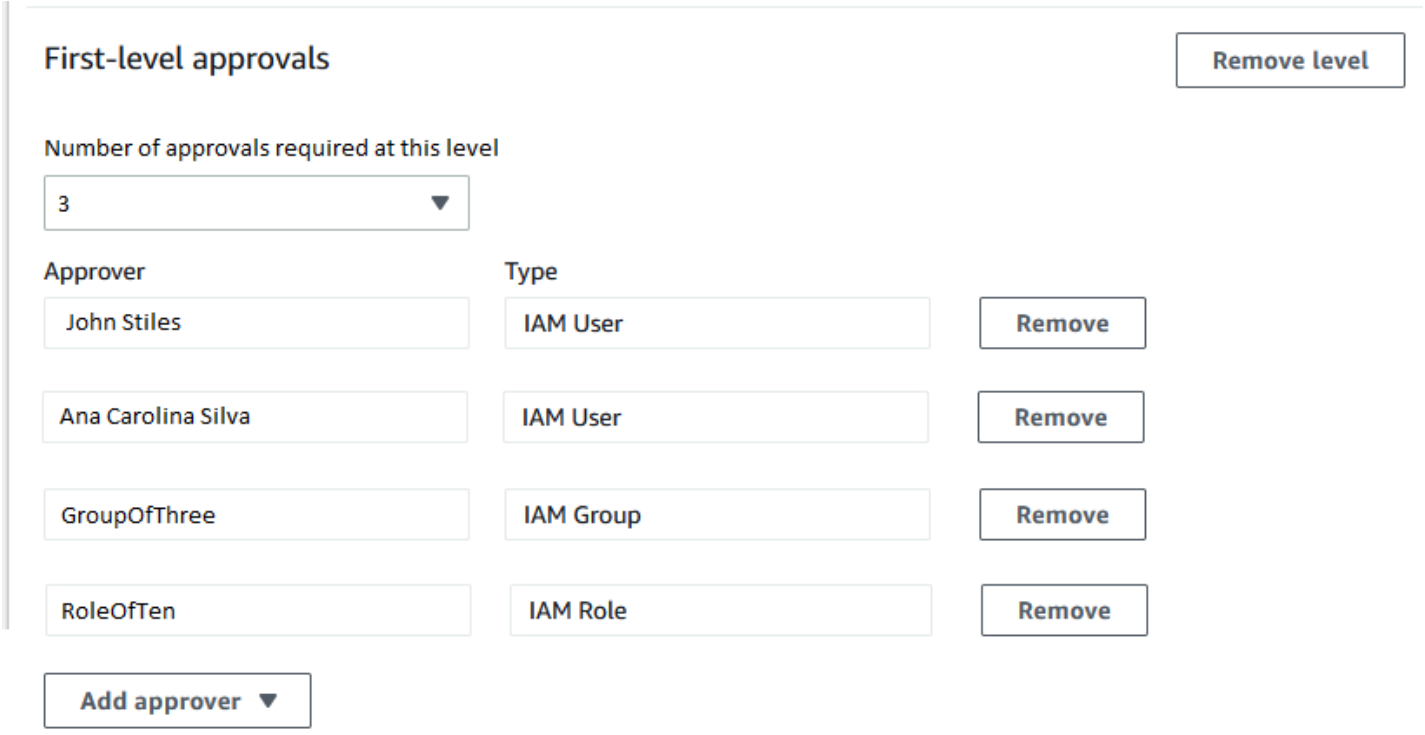
主题

- 逐级批准配置示例
- 逐行批准配置示例
- 逐级和逐行组合批准配置示例

逐级批准配置示例

在下图所示的逐级批准级别设置中，需要三次批准。这些批准可以来自被指定为审批者的 IAM 用户、组和角色的任意组合。指定的审批者包括两个 IAM 用户（John Stiles 和 Ana Carolina Silva），一个包含三个成员的用户组（`GroupOfThree`），以及一个代表十个用户的用户角色（`RoleOfTen`）。

如果 `GroupOfThree` 组中的所有三个用户都批准了更改请求，则该更改请求即获得了该级别的批准。不必获得来自每个用户、组或角色的批准。最低批准数量可以来自指定审批者的任意组合。我们建议对您的 Change Manager 操作进行逐级批准。



以下示例说明了此配置的部分 YAML 代码。

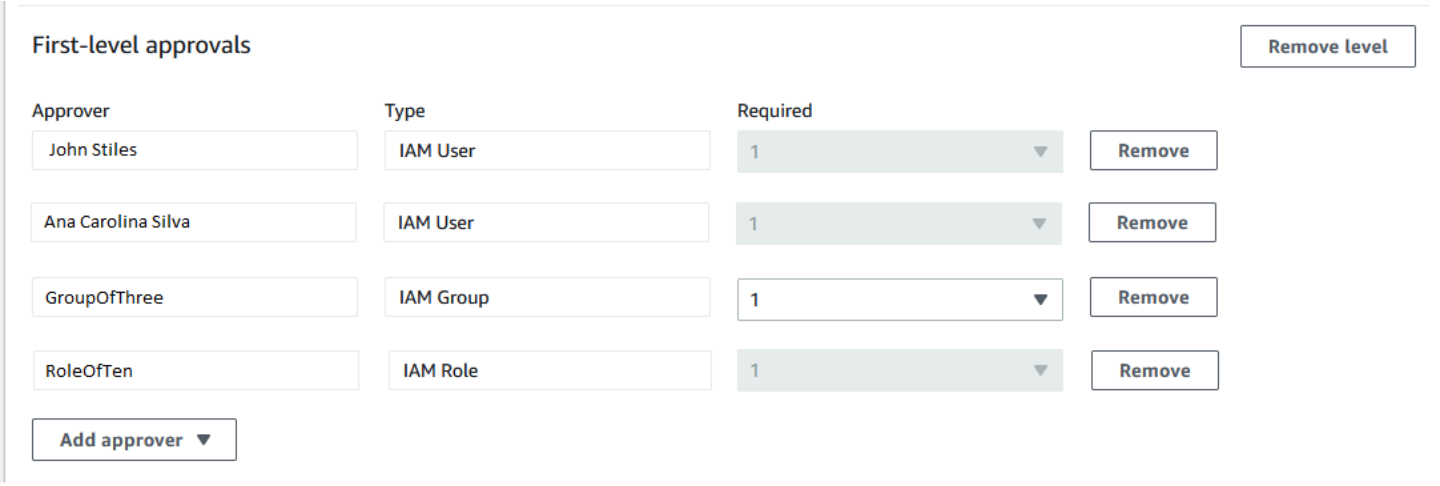
> **Note**
>
> 此版本的 YAML 代码包括一个额外的输入 `MinRequiredApprovals`（包含首字母大写 `M`）。此输入的值表示需要从所有可用审核者获得多少次批准。另请注意，`Approvers` 列表中每个

审批者的 minRequiredApprovals（首字母小写 m）值为 0（零）。这表示相应审批者可以为总批准数做出贡献，但不需要这样做。

```
schemaVersion: "0.3"
emergencyChange: false
autoApprovable: false
mainSteps:
  - name: ApproveAction1
    action: aws:approve
    timeoutSeconds: 604800
    inputs:
      Message: Please approve this change request
      MinRequiredApprovals: 3
      EnhancedApprovals:
        Approvers:
          - approver: John Stiles
            type: IamUser
            minRequiredApprovals: 0
          - approver: Ana Carolina Silva
            type: IamUser
            minRequiredApprovals: 0
          - approver: GroupOfThree
            type: IamGroup
            minRequiredApprovals: 0
          - approver: RoleOfTen
            type: IamRole
            minRequiredApprovals: 0
templateInformation: >
  #### What is the purpose of this change?
    //truncated
```

逐行批准配置示例

在下图所示的批准级别设置中，指定了四个审批者。其中包括两个 IAM 用户（John Stiles 和 Ana Carolina Silva），一个包含三个成员的用户组（GroupOfThree），以及一个代表十个用户的用户角色（RoleOfTen）。为了实现向后兼容性支持逐行批准，但不推荐使用该类型。



要使更改请求在此逐行批准配置中获得批准，该更该请求必须获得所有审批者行的批准：John Stiles、Ana Carolina Silva、GroupOfThree 组的一名成员，以及 RoleOfTen 角色的一名成员。

以下示例说明了此配置的部分 YAML 代码。

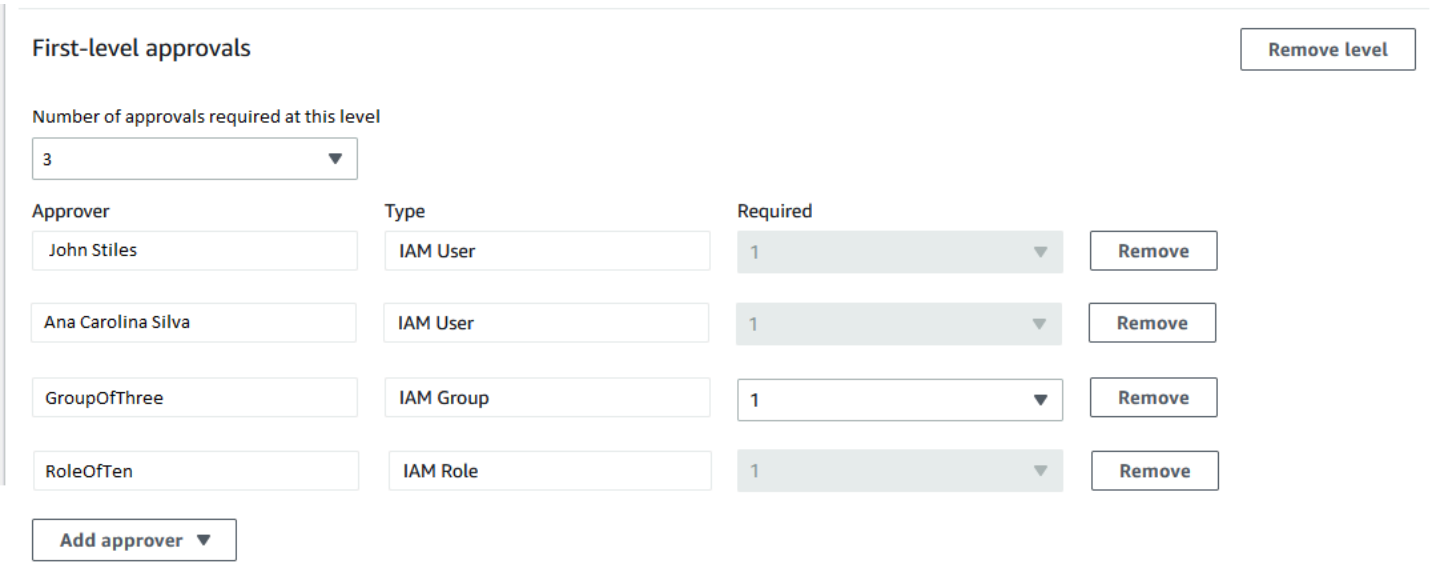
> **Note**
>
> 请注意，每个 minRequiredApprovals 审批者的值均为 1。这表示需要从每个审批者获得一次批准。

```
schemaVersion: "0.3"
emergencyChange: false
autoApprovable: false
mainSteps:
  - name: ApproveAction1
    action: aws:approve
    timeoutSeconds: 10000
    inputs:
      Message: Please approve this change request
      EnhancedApprovals:
        Approvers:
          - approver: John Stiles
            type: IamUser
            minRequiredApprovals: 1
          - approver: Ana Carolina Silva
            type: IamUser
            minRequiredApprovals: 1
          - approver: GroupOfThree
```

```
            type: IamGroup
            minRequiredApprovals: 1
          - approver: RoleOfTen
            type: IamRole
            minRequiredApprovals: 1
executableRunBooks:
  - name: AWS-HelloWorld
    version: $DEFAULT
templateInformation: >
  #### What is the purpose of this change?
    //truncated
```

逐级和逐行组合批准配置示例

在下图所示的逐级和逐行组合批准设置中，为级别指定了三次批准，但为行项目批准指定了四次批准。无论哪种批准类型，需要更多批准的类型都优先于另一种类型，所以此配置需要四次批准。不推荐使用按级别和按行组合批准。




```
schemaVersion: "0.3"
emergencyChange: false
autoApprovable: false
mainSteps:
  - name: ApproveAction1
    action: aws:approve
    timeoutSeconds: 604800
    inputs:
      Message: Please approve this change request
      MinRequiredApprovals: 3
```

```
      EnhancedApprovals:
        Approvers:
          - approver: John Stiles
            type: IamUser
            minRequiredApprovals: 1
          - approver: Ana Carolina Silva
            type: IamUser
            minRequiredApprovals: 1
          - approver: GroupOfThree
            type: IamGroup
            minRequiredApprovals: 1
          - approver: RoleOfTen
            type: IamRole
            minRequiredApprovals: 1
templateInformation: >
  #### What is the purpose of this change?
    //truncated
```

主题

- 使用 Builder (生成器) 创建更改模板
- 使用 Editor (编辑器) 创建更改模板
- 使用命令行工具创建更改模板

使用 Builder (生成器) 创建更改模板

将生成器用于 Change Manager（AWS Systems Manager 中的一项工具）中的更改模板，即可配置在更改模板中定义的运行手册工作流，免于使用使用 JSON 或 YAML 语法。在您指定选项后，系统会将您的输入转换为 YAML 格式，Systems Manager 可以将其用于运行运行手册工作流。

使用 Builder (生成器) 创建更改模板

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Change Manager。
3. 选择创建模板。
4. 对于 Name (名称)，请输入模板的名称，以便于识别其用途，例如 **UpdateEC2LinuxAMI**。
5. 在 Change template details (更改模板详细信息) 部分中，执行以下操作：
   - 对于 Description (描述)，请提供有关如何以及何时使用您所创建的更改模板的简要说明。

此描述可帮助创建更改请求的用户确定他们是否使用了正确的更改模板。它可以帮助审核更改请求的人员了解是否应该批准该请求。

- 对于 Change template type (更改模板类型)，请指定您是创建标准更改模板还是紧急更改模板。

  紧急更改模板用于即便 AWS Systems Manager Change Calendar 所使用的日历中的某个事件阻止更改，也必须进行更改的情况。从紧急更改模板创建的更改请求仍然必须由其指定的审批人员批准，但即便日历被阻止，所请求的更改仍可运行。

- 对于 Runbook options (运行手册选项)，请指定用户在创建更改请求时可以从中选择的运行手册。您可以添加一个运行手册或多个运行手册。或者，您也可以允许请求者指定要使用哪一个运行手册。在上述任何情况下，更改请求中都只能包含一个运行手册。

- 对于 Runbook (运行手册)，请选择用户可以为其更改请求从中选择的运行手册的名称及其版本。无论您向更改模板添加了多少个运行手册，每个更改请求只能选择一个运行手册。

  如果您先前选择了 Any runbook can be used (可以使用任何运行手册)，则您无需指定运行手册。

  > **Tip**
  >
  > 选择运行手册和运行手册版本，然后选择 View (查看)，以便在 Systems Manager Documents (文档) 界面中检查运行手册的内容。

6. 在 Template information (模板信息) 部分中，使用 Markdown 输入从此更改模板创建更改请求的用户的信息。我们提供了一组问题，您可以为创建更改请求的用户提供这些问题，也可以添加其他信息和问题作为替代。

   > **Note**
   >
   > Markdown 是一种标记语言，允许您为文档和文档中的各个步骤中添加维基百科式的描述。有关使用 Markdown 的更多信息，请参阅在 AWS 中使用 Markdown。

   我们建议为用户提供问题，询问有关其更改请求的信息，以帮助审批人员决定是否批准每个更改请求，例如列出作为更改的组成部分需要运行的任何手动步骤和回滚计划。

> **Tip**
>
> 在 Hide preview (隐藏预览) 和 Show preview (显示预览) 之间切换，以便查看您的哪些内容看起来符合您的编写要求。

7. 在 Change request approvals (更改请求批准) 部分中，执行以下操作：

   - （可选）如果您希望允许从此更改模板创建的更改请求自动运行，而不需要任何审批人员进行审核（更改冻结事件除外），则请选择 Enable auto-approval (启用自动批准)。

     > **Note**
     >
     > 在更改模板中启用自动批准，可为用户提供绕过审核人员的选项。他们仍然可以在创建更改请求时选择指定审核人员。因此，您仍然必须在更改模板中指定审批人员选项。

     > **Important**
     >
     > 如果您为更改模板启用了自动批准，则用户可以使用该模板提交更改请求，这些更改请求在运行前无需审核人员进行审核（更改冻结事件审批人员除外）。如果您要限制特定用户、组或 IAM 角色提交自动批准请求，可以使用 IAM policy 中的某一条件来实现此目的。有关更多信息，请参阅 控制对自动批准运行手册工作流的访问。

   - 对于此级别所需的批准数量，选择通过此更改模板创建的更改请求对于此级别必须获得的批准数量。
   - 要添加强制的第一级审批人员，请选择 Add approver (添加审批人员)，然后从以下选项中进行选择：
     - Template specified approvers (模板指定的审批人员) – 从您的账户中选择一个或多个用户、组或 AWS Identity and Access Management (IAM) 角色，以批准从此更改模板创建的更改请求。使用此模板创建的任何更改请求都必须由您指定的每个审批人员进行审核和批准。
     - Request specified approvers（请求指定的审批人员）：提出更改请求的用户将在提出该请求时指定审核人员，并可从您账户中的用户列表中进行选择。

       您在 Required (必需) 列中输入的数字确定使用此更改模板的更改请求必须指定的审核人员的数量。

> **Important**
>
> 在 2023 年 1 月 23 日之前，构建器选项卡仅支持指定逐行批准。新更改模板和您使用构建器选项卡添加到现有更改模板的新级别，仅支持逐级批准。我们建议在您的 Change Manager 操作中仅使用逐级批准。
> 有关更多信息，请参阅 关于更改模板中的批准。

- 对于 SNS topic to notify approvers (要通知审批人员的 SNS 主题)，请执行以下操作：

  1. 选择以下选项之一，在您的账户中指定 Amazon Simple Notification Service (Amazon SNS) 主题，用于向审批人员发送更改请求已准备好供其审核的通知：

     - Enter an SNS Amazon Resource Name (ARN) (输入 SNS Amazon 资源名称 (ARN)) – 对于 Topic ARN (主题 ARN)，请输入现有 Amazon SNS 主题的 ARN。此主题可以位于您组织的任何账户中。
     - Select an existing SNS topic（选择现有 SNS 主题）– 对于 Target notification topic（设定通知主题目标），选择您当前的 AWS 账户中现有 Amazon SNS 主题的 ARN。（如果您尚未在当前的 AWS 账户 和 AWS 区域 中创建任何 Amazon SNS 主题，则此选项不可用。）
     - Specify SNS topic when the change request is created (创建更改请求时指定 SNS 主题) – 创建更改请求的用户可以指定要用于通知的 Amazon SNS 主题。

     > **Note**
     >
     > 您选择的 Amazon SNS 主题必须被配置为指定它所发送的通知，以及接收这些通知的订阅者。其访问策略还必须向 Systems Manager 授予权限，以便 Change Manager 可以发送通知。有关信息，请参阅为 Change Manager 通知配置 Amazon SNS 主题。

  2. 选择 Add notification（添加通知）。

8. （可选）要添加其他级别的审批人员，请选择 Add approval level (添加批准级别)，然后在与此级别对应的模板指定的审批人员和请求指定的审批人员之间进行选择。然后选择 SNS 主题以通知此级别的审批人员。

   在第一级审批人员收到所有批准后，会通知第二级审批人员，依此类推。

   您可以在每个模板中添加最多五个级别的审批人员。例如，您可能需要担任技术角色的用户提供第一级别的批准，然后需要管理人员提供第二级别的批准。

9. 在 Monitoring (监控) 部分中，对于 CloudWatch alarm to monitor (要监控的 CloudWatch 告警)，请输入当前账户中 Amazon CloudWatch 告警的名称，以监控基于此模板的运行手册工作流的进度。

> **Tip**
>
> 要创建新告警，或要审核您要指定的告警的设置，请选择 Open the Amazon CloudWatch console (打开 Amazon CloudWatch 控制台)。有关使用 CloudWatch 告警的信息，请参阅 Amazon CloudWatch 用户指南中的使用 CloudWatch 告警。

10. 在 Notifications（通知）部分执行以下操作：

    1. 选择以下选项之一，在您的账户中指定 Amazon SNS 主题，用于发送有关使用此更改模板创建的更改请求的通知：

       - Enter an SNS Amazon Resource Name (ARN) (输入 SNS Amazon 资源名称 (ARN)) – 对于 Topic ARN (主题 ARN)，请输入现有 Amazon SNS 主题的 ARN。此主题可以位于您组织的任何账户中。
       - Select an existing SNS topic（选择现有 SNS 主题）– 对于 Target notification topic（设定通知主题目标），选择您当前的 AWS 账户中现有 Amazon SNS 主题的 ARN。（如果您尚未在当前的 AWS 账户 和 AWS 区域 中创建任何 Amazon SNS 主题，则此选项不可用。）

    > **Note**
    >
    > 您选择的 Amazon SNS 主题必须被配置为指定它所发送的通知，以及接收这些通知的订阅者。其访问策略还必须向 Systems Manager 授予权限，以便 Change Manager 可以发送通知。有关信息，请参阅为 Change Manager 通知配置 Amazon SNS 主题。

    2. 选择 Add notification（添加通知）。

11. （可选）在 Tags (标签) 部分中，将一个或多个标签键名/键值对应用于更改模板。

    标签是您分配给资源的可选元数据。通过使用标签，您可以按各种方式（如用途、所有者或环境）对资源进行分类。例如，您可能希望标记更改模板，以标识它所进行的更改的类型及其运行的环境。在这种情况下，您可以指定以下键名/键值对：

    - `Key=TaskType,Value=InstanceRepair`
    - `Key=Environment,Value=Production`

12. 选择 Save and preview (保存和预览)。

13. 审核您所创建的更改模板的详细信息。

    如果要在提交更改模板以供审核之前对其进行更改，请选择 Actions (操作)、Edit (编辑)。

    如果您对更改模板的内容感到满意，请选择 Submit for review (提交以供审核)。您的组织或账户内在 Change Manager 中的 Settings (设置) 选项卡上被指定为模板审核人员的用户，将收到新的更改模板正等待其审核的通知。

    如果已为更改模板指定 Amazon SNS 主题，则当更改模板被拒绝或批准时，系统会发送通知。如果您没有收到与此更改模板相关的通知，可在稍后返回 Change Manager 以检查其状态。

使用 Editor (编辑器) 创建更改模板

按照本主题中的步骤操作，通过输入 JSON 或 YAML，而不是使用控制台控件，即可在 Change Manager（AWS Systems Manager 中的一项工具）中配置更改模板。

使用 Editor (编辑器) 创建更改模板

1. 在导航窗格中，选择 Change Manager。
2. 选择创建模板。
3. 对于 Name (名称)，请输入模板的名称，以便于识别其用途，例如 **`RestartEC2LinuxInstance`**。
4. 在 Change template details (更改模板详细信息) 上方，选择 Editor (编辑器)。
5. 在 Document editor (文档编辑器) 部分中，选择 Edit (编辑)，然后为您的更改模板输入 JSON 或 YAML 内容。

   示例如下：

   > **Note**
   >
   > 参数 `minRequiredApprovals` 用于指定对于使用此模板创建的更改请求，必须取得多少指定级别的审阅者批准。
   > 此示例演示了两个级别的批准。您可以指定最多五个级别的批准，但只需要一个级别。
   > 在第一级，每个更改请求必须取得指定用户“John-Doe”的批准。然后，该更改请求必须由 IAM 角色 `Admin` 的任意三个成员批准。
   > 有关批准更改模板的更多信息，请参阅 关于更改模板中的批准。

YAML

```
description: >-
  This change template demonstrates the feature set available for creating
  change templates for Change Manager. This template starts a Runbook workflow
  for the Automation runbook called AWS-HelloWorld.
templateInformation: >
  ### Document Name: HelloWorldChangeTemplate

  ## What does this document do?

  This change template demonstrates the feature set available for creating
  change templates for Change Manager. This template starts a Runbook workflow
  for the Automation runbook called AWS-HelloWorld.

  ## Input Parameters

  * ApproverSnsTopicArn: (Required) Amazon Simple Notification Service ARN for
  approvers.

  * Approver: (Required) The name of the approver to send this request to.

  * ApproverType: (Required) The type of reviewer.
    * Allowed Values: IamUser, IamGroup, IamRole, SSOGroup, SSOUser

  ## Output Parameters

  This document has no outputs
schemaVersion: '0.3'
parameters:
  ApproverSnsTopicArn:
    type: String
    description: Amazon Simple Notification Service ARN for approvers.
  Approver:
    type: String
    description: IAM approver
  ApproverType:
    type: String
    description: >-
      Approver types for the request. Allowed values include IamUser, IamGroup,
      IamRole, SSOGroup, and SSOUser.
executableRunBooks:
```

```
  - name: AWS-HelloWorld
    version: '1'
emergencyChange: false
autoApprovable: false
mainSteps:
  - name: ApproveAction1
    action: 'aws:approve'
    timeoutSeconds: 3600
    inputs:
      Message: >-
        A sample change request has been submitted for your review in Change
        Manager. You can approve or reject this request.
      EnhancedApprovals:
        NotificationArn: '{{ ApproverSnsTopicArn }}'
        Approvers:
          - approver: John-Doe
            type: IamUser
            minRequiredApprovals: 1
  - name: ApproveAction2
    action: 'aws:approve'
    timeoutSeconds: 3600
    inputs:
      Message: >-
        A sample change request has been submitted for your review in Change
        Manager. You can approve or reject this request.
      EnhancedApprovals:
        NotificationArn: '{{ ApproverSnsTopicArn }}'
        Approvers:
          - approver: Admin
            type: IamRole
            minRequiredApprovals: 3
```

JSON

```
{
   "description": "This change template demonstrates the feature set available
 for creating
  change templates for Change Manager. This template starts a Runbook workflow
  for the Automation runbook called AWS-HelloWorld",
   "templateInformation": "### Document Name: HelloWorldChangeTemplate\n\n
    ## What does this document do?\n
    This change template demonstrates the feature set available for creating
 change templates for Change Manager.
```

```
   This template starts a Runbook workflow for the Automation runbook called
AWS-HelloWorld.\n\n
   ## Input Parameters\n* ApproverSnsTopicArn: (Required) Amazon Simple
Notification Service ARN for approvers.\n
   * Approver: (Required) The name of the approver to send this request to.\n
   * ApproverType: (Required) The type of reviewer.  * Allowed Values: IamUser,
IamGroup, IamRole, SSOGroup, SSOUser\n\n
   ## Output Parameters\nThis document has no outputs\n",
  "schemaVersion": "0.3",
  "parameters": {
     "ApproverSnsTopicArn": {
        "type": "String",
        "description": "Amazon Simple Notification Service ARN for approvers."
     },
     "Approver": {
        "type": "String",
        "description": "IAM approver"
     },
     "ApproverType": {
        "type": "String",
        "description": "Approver types for the request. Allowed values include
IamUser, IamGroup, IamRole, SSOGroup, and SSOUser."
     }
  },
  "executableRunBooks": [
     {
        "name": "AWS-HelloWorld",
        "version": "1"
     }
  ],
  "emergencyChange": false,
  "autoApprovable": false,
  "mainSteps": [
     {
        "name": "ApproveAction1",
        "action": "aws:approve",
        "timeoutSeconds": 3600,
        "inputs": {
           "Message": "A sample change request has been submitted for your
review in Change Manager. You can approve or reject this request.",
           "EnhancedApprovals": {
              "NotificationArn": "{{ ApproverSnsTopicArn }}",
              "Approvers": [
                 {
```

```
                    "approver": "John-Doe",
                    "type": "IamUser",
                    "minRequiredApprovals": 1
                 }
              ]
           }
        }
     },
       {
        "name": "ApproveAction2",
        "action": "aws:approve",
        "timeoutSeconds": 3600,
        "inputs": {
           "Message": "A sample change request has been submitted for your
 review in Change Manager. You can approve or reject this request.",
           "EnhancedApprovals": {
              "NotificationArn": "{{ ApproverSnsTopicArn }}",
              "Approvers": [
                 {
                    "approver": "Admin",
                    "type": "IamRole",
                    "minRequiredApprovals": 3
                 }
              ]
           }
        }
     }
  ]
}
```

6. 选择 Save and preview (保存和预览)。
7. 审核您所创建的更改模板的详细信息。

   如果要在提交更改模板以供审核之前对其进行更改，请选择 Actions (操作)、Edit (编辑)。

   如果您对更改模板的内容感到满意，请选择 Submit for review (提交以供审核)。您的组织或账户内在 Change Manager 中的 Settings (设置) 选项卡上被指定为模板审核人员的用户，将收到新的更改模板正等待其审核的通知。

   如果已为更改模板指定 Amazon Simple Notification Service (Amazon SNS) 主题，则当更改模板被拒绝或批准时，系统会发送通知。如果您没有收到与此更改模板相关的通知，可在稍后返回 Change Manager 以检查其状态。

使用命令行工具创建更改模板

以下过程介绍了如何使用 AWS Command Line Interface（AWS CLI）（在 Linux、macOS 或 Windows 上）或 AWS Tools for Windows PowerShell 在 Change Manager（AWS Systems Manager 中的一项工具）中创建更改请求。

创建更改模板

1. 安装并配置 AWS Tools for PowerShell (AWS CLI)（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本以及安装 AWS Tools for PowerShell。

2. 在本地计算机上使用 `MyChangeTemplate.json` 之类的名称创建一个 JSON 文件，然后将更改模板的内容粘贴到此文件中。

   > **Note**
   >
   > 更改模板使用架构 0.3 版本，该版本包含的支持与针对自动化运行手册的支持不完全相同。

   示例如下：

   > **Note**
   >
   > 参数 `minRequiredApprovals` 用于指定对于使用此模板创建的更改请求，必须取得多少指定级别的审阅者批准。
   > 此示例演示了两个级别的批准。您可以指定最多五个级别的批准，但只需要一个级别。
   > 在第一级，每个更改请求必须取得指定用户“John-Doe”的批准。然后，该更改请求必须由 IAM 角色 `Admin` 的任意三个成员批准。
   > 有关批准更改模板的更多信息，请参阅 关于更改模板中的批准。

```
{
   "description": "This change template demonstrates the feature set available for
 creating
  change templates for Change Manager. This template starts a Runbook workflow
  for the Automation runbook called AWS-HelloWorld",
   "templateInformation": "### Document Name: HelloWorldChangeTemplate\n\n
    ## What does this document do?\n
```

```
    This change template demonstrates the feature set available for creating change
 templates for Change Manager.
    This template starts a Runbook workflow for the Automation runbook called AWS-
HelloWorld.\n\n
    ## Input Parameters\n* ApproverSnsTopicArn: (Required) Amazon Simple
 Notification Service ARN for approvers.\n
    * Approver: (Required) The name of the approver to send this request to.\n
    * ApproverType: (Required) The type of reviewer.  * Allowed Values: IamUser,
 IamGroup, IamRole, SSOGroup, SSOUser\n\n
    ## Output Parameters\nThis document has no outputs\n",
  "schemaVersion": "0.3",
  "parameters": {
     "ApproverSnsTopicArn": {
        "type": "String",
        "description": "Amazon Simple Notification Service ARN for approvers."
     },
     "Approver": {
        "type": "String",
        "description": "IAM approver"
     },
     "ApproverType": {
        "type": "String",
        "description": "Approver types for the request. Allowed values include
 IamUser, IamGroup, IamRole, SSOGroup, and SSOUser."
     }
  },
  "executableRunBooks": [
     {
        "name": "AWS-HelloWorld",
        "version": "1"
     }
  ],
  "emergencyChange": false,
  "autoApprovable": false,
  "mainSteps": [
     {
        "name": "ApproveAction1",
        "action": "aws:approve",
        "timeoutSeconds": 3600,
        "inputs": {
           "Message": "A sample change request has been submitted for your review
 in Change Manager. You can approve or reject this request.",
           "EnhancedApprovals": {
              "NotificationArn": "{{ ApproverSnsTopicArn }}",
```

```
                "Approvers": [
                   {
                      "approver": "John-Doe",
                      "type": "IamUser",
                      "minRequiredApprovals": 1
                   }
                ]
             }
          }
       },
         {
          "name": "ApproveAction2",
          "action": "aws:approve",
          "timeoutSeconds": 3600,
          "inputs": {
             "Message": "A sample change request has been submitted for your review
 in Change Manager. You can approve or reject this request.",
             "EnhancedApprovals": {
                "NotificationArn": "{{ ApproverSnsTopicArn }}",
                "Approvers": [
                   {
                      "approver": "Admin",
                      "type": "IamRole",
                      "minRequiredApprovals": 3
                   }
                ]
             }
          }
       }
   ]
}
```

3. 运行以下命令创建更改模板。

   Linux & macOS

   ```
   aws ssm create-document \
       --name MyChangeTemplate \
       --document-format JSON \
       --document-type Automation.ChangeTemplate \
       --content file://MyChangeTemplate.json \
       --tags Key=tag-key,Value=tag-value
   ```

Windows

```
aws ssm create-document ^
    --name MyChangeTemplate ^
    --document-format JSON ^
    --document-type Automation.ChangeTemplate ^
    --content file://MyChangeTemplate.json ^
    --tags Key=tag-key,Value=tag-value
```

PowerShell

```
$json = Get-Content -Path "C:\path\to\file\MyChangeTemplate.json" | Out-String
New-SSMDocument `
    -Content $json `
    -Name "MyChangeTemplate" `
    -DocumentType "Automation.ChangeTemplate" `
    -Tags "Key=tag-key,Value=tag-value"
```

有关可以指定的其他选项的信息，请参阅 create-document。

系统将返回类似于以下内容的信息。

```
{
   "DocumentDescription":{
      "CreatedDate":1.585061751738E9,
      "DefaultVersion":"1",
      "Description":"Use this template to update an EC2 Linux AMI. Requires one
      approver specified in the template and an approver specified in the
 request.",
      "DocumentFormat":"JSON",
      "DocumentType":"Automation",
      "DocumentVersion":"1",
      "Hash":"0d3d879b3ca072e03c12638d0255ebd004d2c65bd318f8354fcde820dEXAMPLE",
      "HashType":"Sha256",
      "LatestVersion":"1",
      "Name":"MyChangeTemplate",
      "Owner":"123456789012",
      "Parameters":[
         {
            "DefaultValue":"",
            "Description":"Level one approvers",
```

```
            "Name":"LevelOneApprovers",
            "Type":"String"
         },
         {
            "DefaultValue":"",
            "Description":"Level one approver type",
            "Name":"LevelOneApproverType",
            "Type":"String"
         },
   "cloudWatchMonitors": {
      "monitors": [
         "my-cloudwatch-alarm"
      ]
   }
      ],
      "PlatformTypes":[
         "Windows",
         "Linux"
      ],
      "SchemaVersion":"0.3",
      "Status":"Creating",
      "Tags":[

      ]
   }
}
```

您的组织或账户内在 Change Manager 中的 Settings (设置) 选项卡上被指定为模板审核人员的用户，将收到新的更改模板正等待其审核的通知。

如果已为更改模板指定 Amazon Simple Notification Service (Amazon SNS) 主题，则当更改模板被拒绝或批准时，系统会发送通知。如果您没有收到与此更改模板相关的通知，可在稍后返回 Change Manager 以检查其状态。

审核和批准或拒绝更改模板

如果您在 Change Manager（AWS Systems Manager 中的一项工具）中被指定为更改模板的审核人员，只要有新的更改模板或新版本的更改模板等待审核，您就会收到通知。Amazon Simple Notification Service (Amazon SNS) 主题可以发送通知。

> **Note**
>
> 此功能取决于是否已将您的账户配置为使用 Amazon SNS 主题发送更改模板审核通知。有关指定模板审核人员通知主题的信息，请参阅 任务 1：配置 Change Manager 用户身份管理和模板审核人员。

要审核更改模板，请访问您的通知中的链接，登录到 AWS Management Console，然后按照本过程中的步骤进行操作。

审核和批准或拒绝更改模板

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Change Manager。
3. 在 Overview (概述) 选项卡底部的 Change templates (更改模板) 部分中，选择 Pending review (等待审核) 中的编号。
4. 在 Change templates (更改模板) 列表中，找到并选择要审核的更改模板的名称。
5. 在摘要页面中，审核更改模板的建议内容，然后执行以下操作之一：
   - 要批准更改模板（这将允许在更改请求中使用该模板），请选择 Approve（批准）。
   - 要拒绝更改模板（这将阻止在更改请求中使用该模板），请选择 Reject（拒绝）。

删除更改模板

本主题介绍如何删除您在 Change Manager（Systems Manager 中的一项工具）中创建的模板。如果您正在为企业使用 Change Manager，则此过程将在委托管理员账户中执行。

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Change Manager。
3. 选择 Templates 选项卡。
4. 选择要删除的模板的名称。
5. 选择 Actions, Delete template（操作，删除模板）。
6. 在确认对话框中，输入 **DELETE** 一词，然后选择 Delete（删除）。

使用更改请求

更改请求是 Change Manager 中的一个请求，旨在运行更新您的 AWS 或本地环境中一个或多个资源的自动化运行手册。更改请求是使用更改模板创建的。

在 Change Manager（AWS Systems Manager 中的一项工具）中创建更改请求时，组织或账户中的一个或多个审批人员必须审核并批准该请求。如果没有获得所需的批准，则不允许运行进行您请求的更改的运行手册工作流。

主题

- 创建更改请求
- 审核和批准或拒绝更改请求

创建更改请求

在 Change Manager（AWS Systems Manager 中的一项工具）中创建更改请求时，您选择的更改模板通常会执行以下操作：

- 为更改请求指定审批人员或者指定所需审批人员的数量
- 指定要用于将您的更改请求通知审批人员的 Amazon Simple Notification Service (Amazon SNS) 主题
- 指定 Amazon CloudWatch 告警，以监控更改请求的运行手册工作流
- 确定您可以从哪些自动化运行手册中进行选择，以进行所请求的更改

在某些情况下，可能会配置更改模板，以使您能指定自己的自动化运行手册以供使用，并指定应该审核和批准该请求的人员。

> **Important**
>
> 如果您在整个组织中使用 Change Manager，我们建议始终从委托管理员账户进行更改。虽然您可以从组织中的其他账户进行更改，但这些更改不会在委派管理员账户中报告，也无法从该账户查看。

主题

- 关于更改请求批准

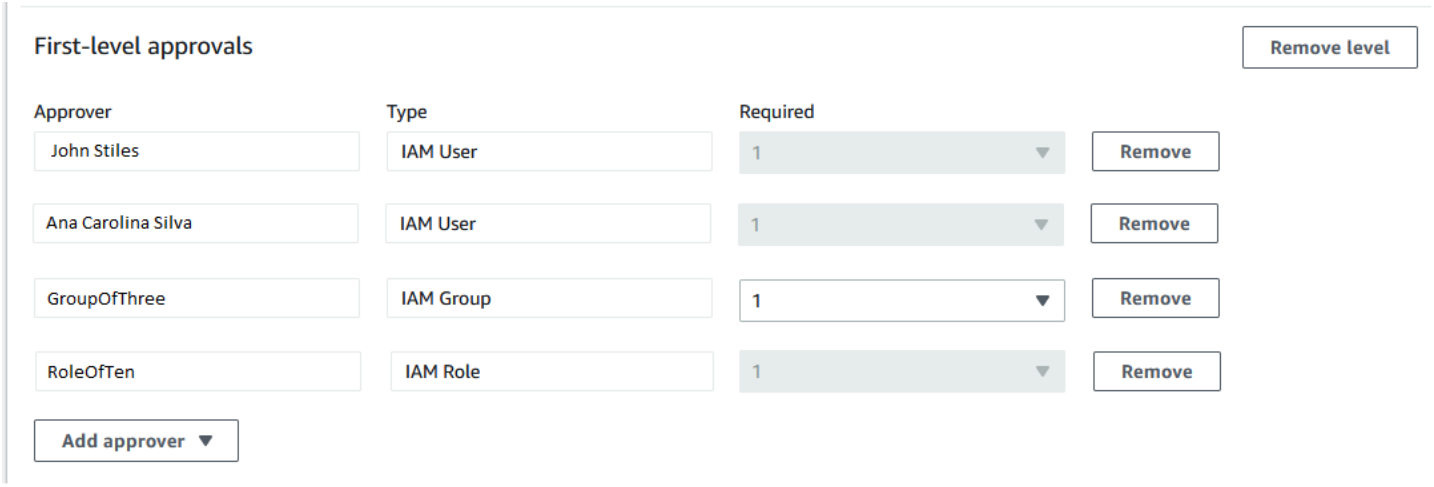
- 创建更改请求（控制台）
- 创建更改请求 (AWS CLI)

关于更改请求批准

根据更改模板中指定的要求，您在该模板中创建的更改请求可能需要获得最多五个级别的批准，然后才能为该请求执行运行手册工作流。对于每个级别，模板创建者最多可以指定五个潜在审批者。审批者并不限于单个用户。从这个意义上讲，审批者也可以是 IAM 组或 IAM 角色。对于 IAM 组和 IAM 角色，属于该组或角色的一个或多个用户可以提供批准，以获得更改请求所需的批准总数。模板创建者还可以指定数量超过更改模板所需数量的审批者。

原始审批工作流以及经过更新的审批工作流和/或审批

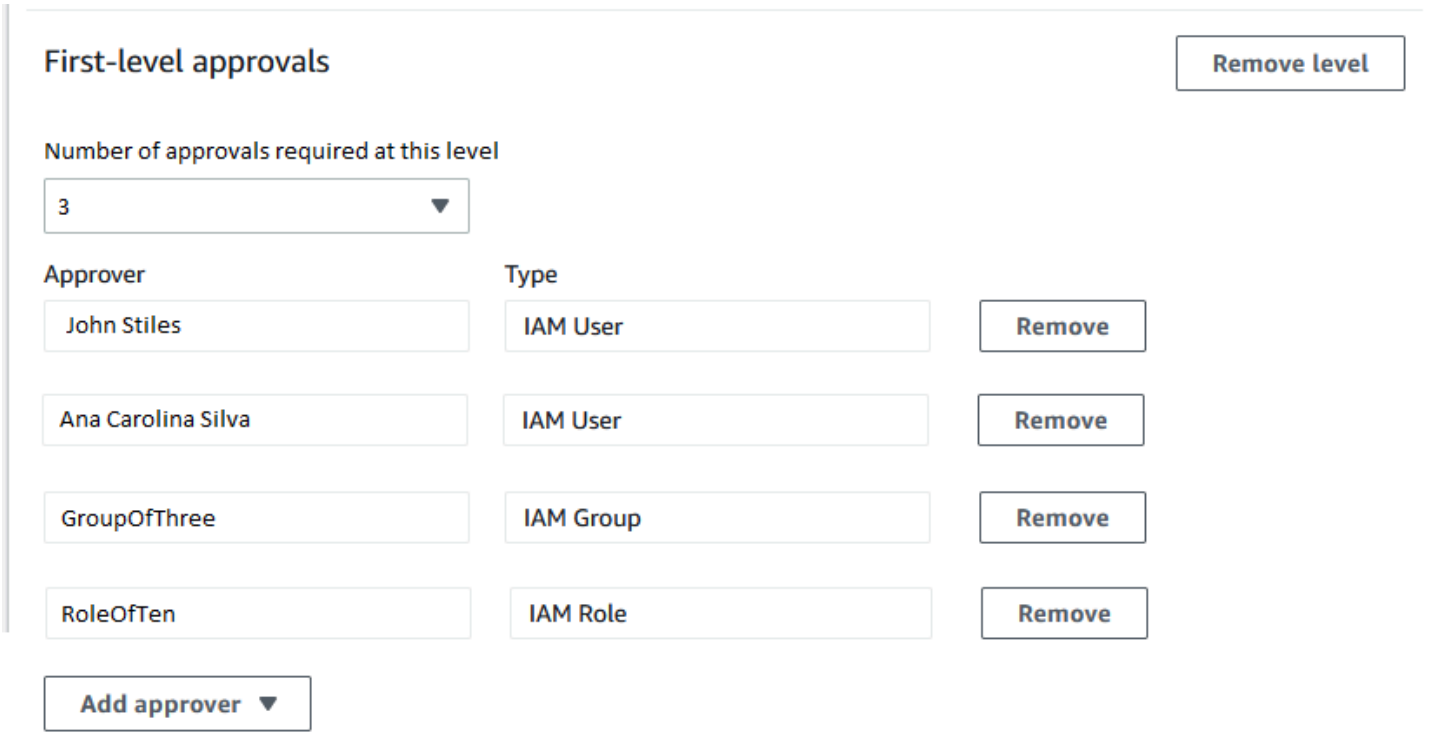
使用在 2023 年 1 月 23 日之前创建的更改模板，必须获得每个指定审批者的批准，更改请求才能在该级别获得批准。例如，在下图所示的批准级别设置中，指定了四个审批者。指定的审批者包括两个用户（John Stiles 和 Ana Carolina Silva），一个包含三个成员的用户组（GroupOfThree），以及一个代表十个用户的用户角色（RoleOfTen）。



要使更改请求在此级别获得批准，该更改请求必须经过 John Stiles、Ana Carolina Silva、GroupOfThree 组的一名成员和 RoleOfTen 角色的一名成员的批准。

使用在 2023 年 1 月 23 日或之后创建的更改模板，模板创建者可为每个批准级别指定所需批准的总数。这些批准可以来自已被指定为审批者的用户、组和角色的任意组合。一个更改模板对于一个级别可能只需要一次批准，但举例来说，需要指定两个单独的用户、两个组和一个角色作为潜在审批者。

例如，在下图所示的批准级别区域中，需要三次批准。模板指定的审批者包括两个用户（John Stiles 和 Ana Carolina Silva），一个包含三个成员的用户组（GroupOfThree），以及一个代表十个用户的用户角色（RoleOfTen）。



如果 `GroupOfThree` 组中的所有三个用户都批准了您的更改请求，则该更改请求即获得了该级别的批准。不必获得来自每个用户、组或角色的批准。最低批准数量可以来自潜在审批者的任意组合。

在创建更改请求后，将向已为该级别的批准通知指定的 Amazon SNS 主题的订阅用户发送通知。更改模板创建者可能已经指定了必须使用的通知主题，或者允许您指定通知主题。

在某一级别获得最低所需批准数量后，将向订阅下一级别 Amazon SNS 主题的审批者发送通知，依此类推。

无论指定了多少个批准级别和审批者，只需拒绝一次更改请求，就会阻止执行该请求的运行手册工作流。

创建更改请求（控制台）

以下过程介绍了如何使用 Systems Manager 控制台创建更改请求。

创建更改请求（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Change Manager。
3. 选择 Create request (创建请求)。

4. 搜索并选择要用于此更改请求的更改模板。
5. 选择下一步。
6. 对于 Name (名称)，请输入更改请求的名称，以便于识别其用途，例如 **UpdateEC2LinuxAMI-us-east-2**。
7. 对于 Runbook (运行手册)，请选择您要用于进行所请求的更改的运行手册。

   > **Note**
   >
   > 如果选择运行手册的选项不可用，则更改模板作者已指定了必须使用哪个运行手册。

8. 对于 Change request information (更改请求信息)，请使用 Markdown 提供有关更改请求的其他信息，以帮助审核人员决定是批准还是拒绝更改请求。您使用的模板的作者可能已提供了说明或需要您解答的问题。

   > **Note**
   >
   > Markdown 是一种标记语言，允许您为文档和文档中的各个步骤中添加维基百科式的描述。有关使用 Markdown 的更多信息，请参阅在 AWS 中使用 Markdown。

9. 在 Workflow start time (工作流开始时间) 部分中，选择以下选项之一：

   - Run the operation at a scheduled time (在计划时间运行该操作) – 对于 Requested start time (请求的开始时间)，请输入您建议运行此请求的运行手册工作流的日期和时间。对于 Estimated end time (估计结束时间)，请输入您预计运行手册工作流完成的日期和时间。（此时间仅是您为审核人员提供的估计值。）

     > **Tip**
     >
     > 选择 View Change Calendar (查看更改日历)，以检查您指定的时间内是否存在任何阻止事件。

   - Run the operation as soon as possible after approval (批准后尽快运行操作)— 如果更改请求获得批准，则运行手册工作流将在出现可以进行更改的非限制时段时立即运行。

10. 在 Change request approvals (更改请求批准) 部分中，执行以下操作：

    1. 如果出现 Approval type (批准类型) 选项，请选择以下选项之一：
       - Automatic approval (自动批准) – 您选择的更改模板将被配置为允许更改请求自动运行，而无需任何审批人员审核。继续执行步骤 11。

> **Note**
>
> 在管理 Systems Manager 使用的 IAM policy 中指定的权限，不得限制您为使更改请求能够自动运行而提交自动批准更改请求。

- Specify approvers (指定审批人员) – 您必须添加一个或多个用户、组或 IAM 角色，才能查看和批准此更改请求。

> **Note**
>
> 即便在管理 Systems Manager 使用的 IAM policy 中指定的权限允许您运行自动批准更改请求，您也可以选择指定审核人员。

2. 选择 Add approver (添加审批人员)，然后从可用审核人员的列表中选择一个或多个用户、组或 AWS Identity and Access Management (IAM) 角色。

> **Note**
>
> 可能已指定了一个或多个审批人员。这意味着已在您选择的更改模板中指定了强制审批人员。不能从请求中移除这些批准者。如果添加审批者按钮不可用，则您选择的模板不允许将其他审核者添加到请求。

有关批准更改请求的更多信息，请参阅 关于更改请求批准。

3. 在 SNS topic to notify approvers (用于通知审批人员的 SNS 主题) 中，请选择以下选项之一，以便在您的账户中指定要用于向您添加到此更改请求的审批人员发送通知的 Amazon SNS 主题。

> **Note**
>
> 如果用于指定 Amazon SNS 主题的选项不可用，则您选择的更改模板已指定了要使用的 Amazon SNS 主题。

- Enter an SNS Amazon Resource Name (ARN) [输入 SNS Amazon 资源名称 (ARN)] – 对于 Topic ARN（主题 ARN），请输入现有 Amazon SNS 主题的 ARN。此主题可以位于您组织的任何账户中。

- Select an existing SNS topic (选择现有 SNS 主题) – 对于 Target notification topic (设定通知主题目标)，选择您的当前账户中现有 Amazon SNS 主题的 ARN。（如果您尚未在当前的 AWS 账户 和 AWS 区域 中创建任何 Amazon SNS 主题，则此选项不可用。）

> **Note**
>
> 您选择的 Amazon SNS 主题必须被配置为指定它所发送的通知，以及接收这些通知的订阅者。其访问策略还必须向 Systems Manager 授予权限，以便 Change Manager 可以发送通知。有关信息，请参阅为 Change Manager 通知配置 Amazon SNS 主题。

4. 选择 Add notification（添加通知）。

11. 选择下一步。
12. 对于 IAM role (IAM 角色)，请选择您的当前账户中拥有所需权限、能够运行为此更改请求指定的运行手册的 IAM 角色。

    此角色也称为 自动化的服务角色或代入角色。有关该角色的更多信息，请参阅 设置自动化。
13. 在 Deployment location (部署位置) 部分中，选择以下选项之一：

> **Note**
>
> 如果您仅将 Change Manager 用于单个 AWS 账户，而非用于在 AWS Organizations 中设置的组织，则不需要指定部署位置。

- Apply change to this account (将更改应用于此账户) – 运行手册工作流仅在当前账户中运行。对于组织，这意味着委托管理员账户。
- Apply change to multiple organizational units (OUs) (将更改应用于多个组织单元 (OU)) – 请执行以下操作：
  1. 对于 Accounts and organizational units (OUs) (账户和组织单位 (OU))，请输入您的组织中成员账户的 ID，格式为 **123456789012**；或组织单位的 ID，格式为 **o-o96EXAMPLE**。
  2. （可选）对于 Execution role name (执行角色名称)，请输入目标账户或 OU 中拥有所需权限、能够运行为此更改请求指定的运行手册的 IAM 角色的名称。您指定的任何 OU 中的所有账户都应该为此角色使用相同的名称。
  3. （可选）对于您要指定的每个额外账户或 OU，请选择 Add another target location (添加其他目标位置)，并重复步骤 a 和 b。

4. 对于 Target AWS 区域 (目标 Amazon Web Services Region)，请选择要在其中进行更改的 Region，例如为美国东部（俄亥俄）区域选择 Ohio (us-east-2)。
5. 展开 Rate control (速率控制)。

   对于 Concurrency (并发)，请输入一个数字，然后从列表中选择此数字是表示可在其中同时运行运行手册工作流的账户数量还是百分比。

   对于 Error threshold (错误阈值)，请输入一个数字，然后从列表中选择此数字是表示在停止操作之前其中的运行手册工作流可能失败的账户数量还是百分比。

14. 在 Deployment targets (部署目标) 部分中，执行以下操作：

    1. 选择下列选项之一：

       - Single resource (单个资源) – 仅对一个资源进行更改。例如，单个节点或单个 Amazon Machine Image (AMI)，具体取决于在此更改请求的运行手册中定义的操作。
       - Multiple resources (多个资源) – 对于 Parameter (参数)，请从此更改请求的运行手册内的可用参数中进行选择。此选择反映了要更新的资源的类型。

         例如，如果此更改请求的运行手册为 AWS-RetartEC2Instance，则可选择 InstanceId，然后通过从以下选项中进行选择，来定义要更新的实例：

         - Specify tags (指定标签) - 输入标记要更新的所有资源时使用的键值对。
         - Choose a resource group (选择资源组) - 选择要更新的所有资源所属的资源组的名称。
         - Specify parameter values (指定参数值) - 确定 Runbook parameters (运行手册参数) 部分中要更新的资源。
         - Target all instances（将所有实例设为目标）- 对目标位置中的所有托管式节点进行更改。

    2. 如果您选择了 Multiple resources (多个资源)，请展开 Rate control (速率控制)。

       对于 Concurrency (并发)，请输入一个数字，然后从列表中选择此数字是表示运行手册工作流可以同时更新的目标的数量还是百分比。

       对于 Error threshold (错误阈值)，请输入一个数字，然后从列表中选择此数字是表示在停止操作之前更新可能失败的目标的数量还是百分比。

15. 如果您选择了 Specify parameter values (指定参数值)，以更新上一步中的多个资源：在 Runbook parameters (运行手册参数) 部分中，指定所需输入参数的值。您必须提供的参数值基于与所选更改模板相关联的自动化运行手册的内容。

例如，如果更改模板使用 AWS-RetartEC2Instance 运行手册，则您必须为 InstanceId 参数输入一个或多个实例 ID。或者，选择 Show interactive instance picker (显示交互式实例选择器)，然后逐个选择可用实例。

16. 选择下一步。
17. 在 Review and submit (审核并提交) 页面上，仔细检查您为此更改请求指定的资源和选项。

    对于您要对其进行更改的任何部分，请选择 Edit (编辑) 按钮。

    如果您对更改请求详细信息感到满意，请选择 Submit for approval (提交以供批准)。

如果已在您为请求选择的更改模板中指定了 Amazon SNS 主题，则当该请求被拒绝或批准时，系统会发送通知。如果您未收到有关该请求的通知，您可以返回到 Change Manager 以检查您的请求的状态。

创建更改请求 (AWS CLI)

您可以使用 AWS Command Line Interface (AWS CLI) 创建更改请求，方法是在 JSON 文件中指定更改请求的选项和参数，并使用 `--cli-input-json` 选项将其包含在您的命令中。

创建更改请求 (AWS CLI)

1. 安装并配置 AWS Tools for PowerShell (AWS CLI)（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本以及安装 AWS Tools for PowerShell。
2. 在本地计算机上使用 MyChangeRequest.json 之类的名称创建一个 JSON 文件，然后将以下内容粘贴到此文件中。

   将###替换为您的更改请求的值。

   > **Note**
   >
   > 此示例 JSON 使用 AWS-HelloWorldChangeTemplate 更改模板和 AWS-HelloWorld 运行手册创建更改请求。要帮助您根据自己的更改请求调整此示例，请参阅《AWS Systems Manager API Reference》中的 StartChangeRequestExecution，以了解有关所有可用参数的信息
   > 有关批准更改请求的更多信息，请参阅 关于更改请求批准。

```
{
    "ChangeRequestName": "MyChangeRequest",
    "DocumentName": "AWS-HelloWorldChangeTemplate",
    "DocumentVersion": "$DEFAULT",
    "ScheduledTime": "2021-12-30T03:00:00",
    "ScheduledEndTime": "2021-12-30T03:05:00",
    "Tags": [
        {
            "Key": "Purpose",
            "Value": "Testing"
        }
    ],
    "Parameters": {
        "Approver": [
            "JohnDoe"
        ],
        "ApproverType": [
            "IamUser"
        ],
        "ApproverSnsTopicArn": [
            "arn:aws:sns:us-east-2:123456789012:MyNotificationTopic"
        ]
    },
    "Runbooks": [
        {
            "DocumentName": "AWS-HelloWorld",
            "DocumentVersion": "1",
            "MaxConcurrency": "1",
            "MaxErrors": "1",
            "Parameters": {
                "AutomationAssumeRole": [
                    "arn:aws:iam::123456789012:role/MyChangeManagerAssumeRole"
                ]
            }
        }
    ],
    "ChangeDetails": "### Document Name: HelloWorldChangeTemplate\n\n## What does
 this document do?\nThis change template demonstrates the feature set available
 for creating change templates for Change Manager. This template starts a Runbook
 workflow for the Automation document called AWS-HelloWorld.\n\n## Input Parameters
\n* ApproverSnsTopicArn: (Required) Amazon Simple Notification Service ARN for
 approvers.\n* Approver: (Required) The name of the approver to send this request
```

```
 to.\n* ApproverType: (Required) The type of reviewer.\n  * Allowed Values:
 IamUser, IamGroup, IamRole, SSOGroup, SSOUser\n\n## Output Parameters\nThis
 document has no outputs \n"
}
```

3. 在创建此 JSON 文件的目录中，运行以下命令。

```
aws ssm start-change-request-execution --cli-input-json file://MyChangeRequest.json
```

系统将返回类似于以下内容的信息。

```
{
    "AutomationExecutionId": "b3c1357a-5756-4839-8617-2d2a4EXAMPLE"
}
```

审核和批准或拒绝更改请求

如果在 Change Manager（AWS Systems Manager 中的一项工具）中被指定为更改请求的审核人员，只要有新的更改请求等待审核，您就会通过 Amazon Simple Notification Service（Amazon SNS）主题收到通知。

> **Note**
>
> 此功能取决于是否在更改模板中指定了 Amazon SNS 来发送审核通知。有关信息，请参阅为 Change Manager 通知配置 Amazon SNS 主题。

要审核更改请求，可以访问您的通知中的链接，或直接登录到 AWS Management Console，然后按照本过程中的步骤进行操作。

> **Note**
>
> 如果在更改模板中为审核人员分配了 Amazon SNS 主题，则当更改请求状态发生变化时，会向该主题的订阅者发送通知。
> 有关批准更改请求的更多信息，请参阅 关于更改请求批准。

审核和批准或拒绝更改请求（控制台）

以下过程介绍了如何使用 Systems Manager 控制台审核和批准或拒绝更改请求。

审核和批准或拒绝更改请求

1. 打开您收到的电子邮件通知中的链接，然后登录到 AWS Management Console，它将引导您转到供您审核的更改请求。
2. 在摘要页面中，审核更改请求的建议内容。

   要批准更改请求，请选择 Approve (批准)。在对话框中，提供您要为此批准添加的任何注释，然后选择 Approve (批准)。此请求所代表的运行手册工作流将按计划开始运行，或在更改未被任何限制阻止时立即开始运行。

   –或者–

   要拒绝更改请求，请选择 Reject (拒绝)。在对话框中，提供您要为此拒绝添加的任何注释，然后选择 Reject (拒绝)。

批量审核和批准或拒绝更改请求

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Change Manager。
3. 选择 Approvals（批准）选项卡。
4. （可选）通过选择每个请求的名称来查看等待批准的请求的详细信息，然后返回到 Approvals（批准）选项卡。
5. 选中要批准的每个更改请求的复选框。

   –或者–

   选中要拒绝的每个更改请求的复选框。
6. 在对话框中，提供您要为此批准或拒绝添加的任何注释。
7. 根据您是要批准还是拒绝选定的更改请求，选择 Approve（批准）或 Reject（拒绝）。

审核和批准或拒绝更改请求（命令行）

以下过程介绍了如何使用 AWS Command Line Interface (AWS CLI)（在 Linux、macOS 或 Windows 上）审核和批准或拒绝更改请求。

审核和批准或拒绝更改请求

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 在您的本地计算机上创建一个 JSON 文件，该文件将为您的 AWS CLI 调用指定参数。

```
{
  "OpsItemFilters":
  [
    {
      "Key": "OpsItemType",
      "Values": ["/aws/changerequest"],
      "Operator": "Equal"
    }
  ],
  "MaxResults": number
}
```

   您可以通过在该 JSON 文件中指定审批人员的 Amazon Resource Name (ARN) 来筛选特定审批人员的结果。见下列。

```
{
  "OpsItemFilters":
  [
    {
      "Key": "OpsItemType",
      "Values": ["/aws/changerequest"],
      "Operator": "Equal"
    },
    {
      "Key": "ChangeRequestByApproverArn",
      "Values": ["arn:aws:iam::account-id:user/user-name"],
      "Operator": "Equal"
    }
  ],
  "MaxResults": number
```

```
}
```

3. 运行以下命令可以查看您在 JSON 文件中指定的最大更改请求数量。

   Linux & macOS

   ```
   aws ssm describe-ops-items \
   --cli-input-json file://filename.json
   ```

   Windows

   ```
   aws ssm describe-ops-items ^
   --cli-input-json file://filename.json
   ```

4. 运行以下命令可以批准或拒绝更改请求。

   Linux & macOS

   ```
   aws ssm send-automation-signal \
       --automation-execution-id ID \
       --signal-type Approve_or_Reject \
       --payload Comment="message"
   ```

   Windows

   ```
   aws ssm send-automation-signal ^
   --automation-execution-id ID ^
       --signal-type Approve_or_Reject ^
       --payload Comment="message"
   ```

   如果已在您为请求选择的更改模板中指定了 Amazon SNS 主题，则当该请求被拒绝或批准时，系统会发送通知。如果您未收到有关该请求的通知，您可以返回到 Change Manager 以检查您的请求的状态。有关使用此命令时其他选项的信息，请参阅 《AWS CLI Command Reference》的 AWS Systems Manager 部分中的 send-automation-signal。

审核更改请求详细信息、任务和时间表（控制台）

您可以在 Change Manager（AWS Systems Manager 中的一项工具）的控制面板中查看有关更改请求（包括已处理其更改的请求）的信息。这些详细信息包含一个链接，它指向运行进行更改的运行手册的

自动化操作。在创建请求时会生成自动化执行 ID，但仅在获得所有批准并且没有部署任何阻止更改的限制的情况下，该流程才会运行。

审核更改请求详细信息、任务和时间表

1. 在导航窗格中，选择 Change Manager。
2. 选择 Requests (请求) 选项卡。
3. 在 Change requests (更改请求) 部分中，搜索您要审核的更改请求。

   您可以使用 Create date range (创建日期范围) 选项，将结果限制在特定时间段内。

   您可以通过以下属性筛选请求：

   - `Status`
   - `Request ID`
   - `Approver`
   - `Requester`

   例如，要查看有关在过去 24 小时内成功完成的所有更改请求的详细信息，请执行以下操作：

   1. 对于 Create date range (创建日期范围)，请选择 1d。
   2. 在搜索框中，选择 Status (状态)、CompletedWithSuccess。
   3. 在结果中，选择要审核其结果的、已成功完成的更改请求的名称。
4. 在以下选项卡上查看有关更改请求的信息：

   - Request details (请求详细信息) - 查看有关更改请求的基本详细信息，包括请求者、更改模板，和为更改选择的自动化运行手册。您还可以访问指向自动化操作详细信息的链接，查看有关请求中指定的任何运行手册参数、分配给更改请求的 Amazon CloudWatch 告警以及为请求提供的批准和注释的信息。
   - Task (任务) - 查看有关更改中的任务的信息，包括已完成的更改请求的任务状态、目标资源、关联的自动化运行手册中的步骤，以及并发和错误阈值详细信息。
   - Timeline (时间表) - 查看与更改请求关联的所有事件的摘要，按日期和时间列出。该摘要将指明创建更改请求的时间、指定的审批人员的操作、已批准的更改请求计划运行时间的记录、运行手册工作流详细信息，以及运行手册中整个更改流程和每个步骤的状态变化。

- Associated events（关联事件）：查看有关在 AWS CloudTrail Lake 中记录的变更请求的可审批详细信息。详细信息包括运行了哪些 API 操作、这些操作包含的请求参数、运行操作的用户账户、在此过程中更新的资源等。

  当您启用 CloudTrail Lake 事件跟踪时，CloudTrail Lake 会为与您的变更请求相关的事件创建事件数据存储。事件详细信息适用于提出变更请求的账户或组织。您可以通过账户或组织中的任何变更请求启用 CloudTrail Lake 事件跟踪。有关启用 CloudTrail Lake 集成和创建事件数据存储的信息，请参阅 监控您的变更请求事件。

  > **Note**
  >
  > 使用 CloudTrail Lake 需要付费。有关详细信息，请参阅 AWS CloudTrail 定价。

查看更改请求的聚合计数（命令行）

您可以使用 GetOpsSummary API 操作，查看 Change Manager（AWS Systems Manager 中的一项工具）中更改请求的汇总计数。此 API 操作可以返回单个 AWS 区域中的单个 AWS 账户的计数，或者多个账户和多个区域的计数。

> **Note**
>
> 如果您要查看多个 AWS 账户 和多个 AWS 区域 的聚合计数，必须设置和配置资源数据同步。有关更多信息，请参阅 为 Inventory 创建资源数据同步。

以下过程介绍了如何使用 AWS Command Line Interface (AWS CLI)（在 Linux、macOS 或 Windows 上）查看更改请求的聚合计数。

查看更改请求的聚合计数

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 运行以下命令之一。

   Single account and Region (单个账户和区域)

   此命令将返回为其配置您的 AWS CLI 会话的 AWS 账户 和 AWS 区域 的所有更改请求的计数。

Linux & macOS

```
aws ssm get-ops-summary \
--filters Key=AWS:OpsItem.OpsItemType,Values="/aws/changerequests",Type=Equal \
--aggregators AggregatorType=count,AttributeName=Status,TypeName=AWS:OpsItem
```

Windows

```
aws ssm get-ops-summary ^
--filters Key=AWS:OpsItem.OpsItemType,Values="/aws/changerequests",Type=Equal ^
--aggregators AggregatorType=count,AttributeName=Status,TypeName=AWS:OpsItem
```

该调用将返回类似于以下内容的信息。

```
{
    "Entities": [
        {
            "Data": {
                "AWS:OpsItem": {
                    "Content": [
                        {
                            "Count": "38",
                            "Status": "Open"
                        }
                    ]
                }
            }
        }
    ]
}
```

Multiple accounts and/or Regions (多个账户和/或区域)

此命令将返回在资源数据同步中指定的 AWS 账户和 AWS 区域的所有更改请求的计数。

Linux & macOS

```
aws ssm get-ops-summary \
    --sync-name resource_data_sync_name \
```

```
    --filters Key=AWS:OpsItem.OpsItemType,Values="/aws/
changerequests",Type=Equal \
    --aggregators AggregatorType=count,AttributeName=Status,TypeName=AWS:OpsItem
```

Windows

```
aws ssm get-ops-summary ^
    --sync-name resource_data_sync_name ^
    --filters Key=AWS:OpsItem.OpsItemType,Values="/aws/
changerequests",Type=Equal ^
    --aggregators AggregatorType=count,AttributeName=Status,TypeName=AWS:OpsItem
```

该调用将返回类似于以下内容的信息。

```
{
    "Entities": [
        {
            "Data": {
                "AWS:OpsItem": {
                    "Content": [
                        {
                            "Count": "43",
                            "Status": "Open"
                        },
                        {
                            "Count": "2",
                            "Status": "Resolved"
                        }
                    ]
                }
            }
        }
    ]
}
```

Multiple accounts and a specific Region (多个账户和某一特定区域)

此命令将返回在资源数据同步中指定的 AWS 账户 的所有更改请求的计数。但是，它只会返回该命令中指定的区域中的数据。

Linux & macOS

```
aws ssm get-ops-summary \
    --sync-name resource_data_sync_name \
    --filters Key=AWS:OpsItem.SourceRegion,Values='Region',Type=Equal
 Key=AWS:OpsItem.OpsItemType,Values="/aws/changerequests",Type=Equal \
    --aggregators AggregatorType=count,AttributeName=Status,TypeName=AWS:OpsItem
```

Windows

```
aws ssm get-ops-summary ^
    --sync-name resource_data_sync_name ^
    --filters Key=AWS:OpsItem.SourceRegion,Values='Region',Type=Equal
 Key=AWS:OpsItem.OpsItemType,Values="/aws/changerequests",Type=Equal ^
    --aggregators AggregatorType=count,AttributeName=Status,TypeName=AWS:OpsItem
```

Multiple accounts and Regions with output grouped by Region (输出按区域分组的多个账户和区域)

此命令将返回在资源数据同步中指定的 AWS 账户和 AWS 区域的所有更改请求的计数。输出将显示每个区域的计数信息。

Linux & macOS

```
aws ssm get-ops-summary \
    --sync-name resource_data_sync_name \
    --filters Key=AWS:OpsItem.OpsItemType,Values="/aws/
changerequests",Type=Equal \
    --aggregators
 '[{"AggregatorType":"count","TypeName":"AWS:OpsItem","AttributeName":"Status","Aggregat
[{"AggregatorType":"count","TypeName":"AWS:OpsItem","AttributeName":"SourceRegion"}]}]'
```

Windows

```
aws ssm get-ops-summary ^
    --sync-name resource_data_sync_name ^
    --filters Key=AWS:OpsItem.OpsItemType,Values="/aws/
changerequests",Type=Equal ^
```

```
    --aggregators
 '[{"AggregatorType":"count","TypeName":"AWS:OpsItem","AttributeName":"Status","Aggregat
[{"AggregatorType":"count","TypeName":"AWS:OpsItem","AttributeName":"SourceRegion"}]}]'
```

该调用将返回类似于以下内容的信息。

```
{
        "Entities": [
            {
                "Data": {
                    "AWS:OpsItem": {
                        "Content": [
                            {
                                "Count": "38",
                                "SourceRegion": "us-east-1",
                                "Status": "Open"
                            },
                            {
                                "Count": "4",
                                "SourceRegion": "us-east-2",
                                "Status": "Open"
                            },
                            {
                                "Count": "1",
                                "SourceRegion": "us-west-1",
                                "Status": "Open"
                            },
                            {
                                "Count": "2",
                                "SourceRegion": "us-east-2",
                                "Status": "Resolved"
                            }
                        ]
                    }
                }
            }
        ]
    }
```

Multiple accounts and Regions with output grouped by accounts and Regions (输出按帐户和区域分组的多个账户和区域)

此命令将返回在资源数据同步中指定的 AWS 账户 和 AWS 区域 的所有更改请求的计数。输出将按账户和区域对计数信息进行分组。

Linux & macOS

```
aws ssm get-ops-summary \
    --sync-name resource_data_sync_name \
    --filters Key=AWS:OpsItem.OpsItemType,Values="/aws/
changerequests",Type=Equal \
    --aggregators
 '[{"AggregatorType":"count","TypeName":"AWS:OpsItem","AttributeName":"Status","Aggregat
[{"AggregatorType":"count","TypeName":"AWS:OpsItem","AttributeName":"SourceAccountId","A
[{"AggregatorType":"count","TypeName":"AWS:OpsItem","AttributeName":"SourceRegion"}]}]}]
```

Windows

```
aws ssm get-ops-summary ^
    --sync-name resource_data_sync_name ^
    --filters Key=AWS:OpsItem.OpsItemType,Values="/aws/
changerequests",Type=Equal ^
    --aggregators
 '[{"AggregatorType":"count","TypeName":"AWS:OpsItem","AttributeName":"Status","Aggregat
[{"AggregatorType":"count","TypeName":"AWS:OpsItem","AttributeName":"SourceAccountId","A
[{"AggregatorType":"count","TypeName":"AWS:OpsItem","AttributeName":"SourceRegion"}]}]}]
```

该调用将返回类似于以下内容的信息。

```
{
    "Entities": [
        {
            "Data": {
                "AWS:OpsItem": {
                    "Content": [
                        {
                            "Count": "38",
                            "SourceAccountId": "123456789012",
                            "SourceRegion": "us-east-1",
                            "Status": "Open"
                        },
                        {
```

```
                        "Count": "4",
                        "SourceAccountId": "111122223333",
                        "SourceRegion": "us-east-2",
                        "Status": "Open"
                    },
                    {
                        "Count": "1",
                        "SourceAccountId": "111122223333",
                        "SourceRegion": "us-west-1",
                        "Status": "Open"
                    },
                    {
                        "Count": "2",
                        "SourceAccountId": "444455556666",
                        "SourceRegion": "us-east-2",
                        "Status": "Resolved"
                    },
                    {
                        "Count": "1",
                        "SourceAccountId": "222222222222",
                        "SourceRegion": "us-east-1",
                        "Status": "Open"
                    }
                ]
            }
        }
    ]
}
```

审计和记录 Change Manager 活动

您可以使用 Amazon CloudWatch 和 AWS CloudTrail 警报，审核 Change Manager（AWS Systems Manager 中的一项工具）中的活动。

有关 Systems Manager 的审计和日志记录选项的更多信息，请参阅 AWS Systems Manager 中的日志记录和监控。

使用 CloudWatch 告警审计 Change Manager

您可以配置 CloudWatch 告警并将其分配给更改模板。如果满足该告警中定义的任何条件，则将执行为该告警指定的操作。在告警配置中，您可以指定 Amazon Simple Notification Service (Amazon SNS) 主题，以便在满足告警条件时发送通知。

有关创建 Change Manager 模板的信息，请参阅 使用更改模板。

有关创建 CloudWatch 告警的信息，请参阅 Amazon CloudWatch 用户指南中的使用 CloudWatch 告警。

使用 CloudTrail 审计 Change Manager 的活动

CloudTrail 可以捕获在 Systems Manager 控制台、AWS Command Line Interface (AWS CLI) 和 Systems Manager 软件开发工具包中进行的 API 调用。您可以在 CloudTrail 控制台或 Amazon Simple Storage Service (Amazon S3) 存储桶中查看信息，这些信息存储在其中。账户的所有 CloudTrail 日志都存储在一个存储桶中。

Change Manager 操作的日志可以显示更改模板文档创建、更改模板和更改请求批准及拒绝、自动化运行手册生成的活动等。有关查看和使用 Systems Manager 活动的 CloudTrail 日志的更多信息，请参阅 使用 AWS CloudTrail 记录 AWS Systems Manager API 调用。

排查 Change Manager 问题

利用以下信息，帮助排查Change Manager（AWS Systems Manager 中的一项工具）出现的问题。

主题

- 在使用 Active Directory（组）时，在更改请求批准期间出现“Group {GUID} not found”（未找到组 {GUID}）错误。

在使用 Active Directory（组）时，在更改请求批准期间出现“Group ***{GUID}*** not found”（未找到组 {GUID}）错误。

问题：在将 AWS IAM Identity Center（IAM Identity Center）用于用户身份管理时，已在 Change Manager 中获得批准权限的 Active Directory 组成员收到“not authorized”（未授权）或“group not found”（未找到组）错误。

- 解决方法：当您在 IAM Identity Center 中选择 Active Directory 组以访问 AWS Management Console 时，系统将计划定期同步，以将这些 Active Directory 组中的信息复制到 IAM Identity Center。必须完成此流程，然后通过 Active Directory 组成员身份授权的用户才能成功批准请求。有关更多信息，请参阅 AWS IAM Identity Center 用户指南中的连接到您的 Microsoft AD 目录。

AWS Systems Manager 文档

AWS Systems Manager 文档 (SSM document) 定义 Systems Manager 对您的托管实例执行的操作。Systems Manager 包括 100 个预先配置的文档，您可以在运行时通过指定参数进行使用。您可以在 Systems Manager 文档控制台中找到预配置的文档，方法是选择 Owned by Amazon（Amazon 所有）选项卡，或者通过在调用 `ListDocuments` API 操作时指定 Amazon `Owner` 筛选条件。文档使用 JavaScript 对象表示法 (JSON) 或 YAML，并包括您指定的步骤和参数。

为增强安全性，从 2025 年 7 月 14 日起，SSM 文档在处理参数时支持环境变量插值。此功能在 2.2 版架构和 3.3.2746.0 或更高版本的 SSM Agent 中可用，可帮助防范命令注入攻击。

要开始使用 SSM 文档，请打开 Systems Manager 控制台。在导航窗格中，选择文档。

> **Important**
>
> 在 Systems Manager 中，Amazon 拥有的 SSM 文档是由 Amazon Web Services 自己创建和管理的文档。Amazon 拥有的文档在文档名称中包含 `AWS-*` 之类的前缀。文档的所有者被视为 Amazon，而不是 AWS 内的特定用户账户。这些文档可供所有人公开使用。

我的组织如何从 Documents 工具获益？

Documents（AWS Systems Manager 中的一项工具）具有以下优点：

- 文档类别

  为了帮助您找到所需的文档，请根据要搜索的文档类型选择一个类别。要扩大搜索范围，您可以选择同一文档类型的多个类别。不支持选择不同文档类型的类别。分类仅支持 Amazon 拥有的文档。

- 文档版本

  您可以创建并保存不同版本的文档。您可以为每个文档指定默认版本。文档的默认版本可以更新到较新版本或恢复到较旧版本。当您更改文档的内容时，Systems Manager 将自动递增文档的版本。您可以通过在控制台、AWS Command Line Interface (AWS CLI) 命令或 API 调用中指定文档版本来检索或使用任何版本的文档。

- 根据您的需求自定义文档

  如果您要通过文档自定义步骤和操作，可以自行创建文档。系统将文档与您的 AWS 账户 一起存储在创建其所在的 AWS 区域 中。有关如何创建 SSM 文档的更多信息，请参阅 创建 SSM 文档内容。

- 标记文档

  您可以标记文档，以便根据为其分配的标签快速识别一个或多个文档。例如，您可以为特定环境、部门、用户、组或时间段的文档添加标签。此外，您还可以通过创建一个指定用户或组可访问的标签的 AWS Identity and Access Management (IAM) policy 来限制对文档的访问。

- 共享文档

  您可以将文档公开，或者将它们与同一 AWS 区域 区域中的特定 AWS 账户 共享。在账户之间共享文档可能很有用，例如，如果您希望提供给客户或员工的所有 Amazon Elastic Compute Cloud (Amazon EC2) 实例都能够具有相同配置。除了保持实例上的应用程序或补丁的最新状态，您可能还希望限制客户实例执行某些活动。或者，您可能需要确保整个组织的员工账户使用的实例均被授予访问特定内部资源的访问权限。有关更多信息，请参阅 共享 SSM 文档。

谁应该使用文档？

- 任何希望使用 Systems Manager 工具大规模提高其运营效率、减少与手动干预相关的错误以及缩短解决常见问题的时间的 AWS 客户。
- 希望自动执行部署和配置任务的基础设施专家。
- 想要可靠地解决常见问题、提高故障排除效率和减少重复性操作的管理员。
- 想要自动执行通常需要手动执行的任务的用户。

SSM 文档有哪些类型？

下表介绍了不同类型的 SSM 文档及其使用。

类型	一起使用	详细信息
ApplicationConfiguration ApplicationConfigurationSchema	AWS AppConfig	AWS AppConfig 是 AWS Systems Manager 中的一项工具，可让您创建、管理以及快速部署应用程序配置。您可以通过创建使用 `ApplicationConfiguration` 文档类型的文档，从而将配置数据存储到 SSM 文档中。有关更多信息，请参阅《AWS

类型	一起使用	详细信息
		AppConfig 用户指南》中的自由度配置。 如果在 SSM 文档中创建配置，则必须指定对应的 JSON Schema。该 schema 使用 `ApplicationConfigurationSchema` 文档类型，并且与一组规则类似，它会定义每个应用程序配置设置允许的属性。有关更多信息，请参阅《AWS AppConfig 用户指南》中的关于验证器。
自动化运行手册	自动化 State Manager Maintenance Windows	在执行常规维护和部署任务 – 如创建或更新 Amazon Machine Image (AMI)时使用自动化手册 - 使用自动化运行手册。State Manager 使用自动化运行手册应用配置。这些操作可以在实例生命周期内的任何时刻在一个或多个目标上运行。Maintenance Windows 使用自动化运行手册基于指定的计划执行常规维护和部署任务。 macOS 的 EC2 实例也支持基于 Linux 的操作系统支持的所有自动化运行手册。

类型	一起使用	详细信息
更改日历文档	Change Calendar	Change Calendar 是 AWS Systems Manager 中的一项工具，使用 `ChangeCalendar` 文档类型。Change Calendar 文档存储日历条目和关联的事件，这些事件可以允许或阻止自动化操作更改您的环境。在 Change Calendar 中，文档以明文格式存储 iCalendar 2.0 数据。 Change Calendar 在 EC2 实例上不支持 macOS。
AWS CloudFormation 模板	AWS CloudFormation	AWS CloudFormation 模板描述您要在 CloudFormation 堆栈中调配的资源。通过将 CloudFormation 模板存储为 Systems Manager 文档，您可以受益于 Systems Manager 文档功能。其中包括创建和比较模板的多个版本，以及与同一 AWS 区域 中的其它账号共享模板。 您可以使用 Application Manager（Systems Manager 中的一项工具）创建和编辑 CloudFormation 模板及堆栈。有关更多信息，请参阅 在 Application Manager 中使用 AWS CloudFormation 模板和堆栈。

类型	一起使用	详细信息
命令文档	Run Command State Manager Maintenance Windows	Run Command 是 AWS Systems Manager 中的一项工具，使用命令文档运行命令。State Manager是 AWS Systems Manager 中的一项工具，使用命令文档应用配置。这些操作可以在实例生命周期内的任何时刻在一个或多个目标上运行。Maintenance Windows 是 AWS Systems Manager 中的一项工具，使用命令文档基于指定的计划应用配置。 大多数命令文档在所有 Linux 和 Systems Manager 所支持的 Windows Server 操作系统上受支持。macOS 的 EC2 实例支持以下命令文档： • `AWS-ConfigureAWSPackage` • `AWS-RunPatchBaseline` • `AWS-RunPatchBaselineAssociation` • `AWS-RunShellScript`

类型	一起使用	详细信息
AWS Config 一致性包模板	AWS Config	AWS Config 一致性包模板是 YAML 格式的文档，用于创建包含 AWS Config 规则（托管或自定义）和修复操作列表的一致性包。 有关更多信息，请参阅一致性包。
软件包文档	Distributor	在 Distributor（AWS Systems Manager 中的一项工具）中，软件包用 SSM 文档表示。软件包文档包括附加的 ZIP 存档文件，其中包含要在托管实例上安装的软件或资产。在 Distributor 中创建软件包会创建软件包文档。 Oracle Linux 和 macOS 托管实例上不受支持 Distributor。
策略文档	State Manager	Inventory 是 AWS Systems Manager 中的一项工具，使用 `AWS-GatherSoftwareInventory` 策略文档和State Manager关联从托管式实例中收集清单数据。在创建您自己的 SSM 文档时，自动化文档和命令文档是在托管实例上实施策略的首选方法。 Systems Manager 清单和 `AWS-GatherSoftwareInventory` 策略文档在 Systems Manager 所支持的所有操作系统上均受支持。

类型	一起使用	详细信息
事件后分析模板	Incident Manager 事件后分析	Incident Manager 使用事件后分析模板基于 AWS 运营管理最佳实践创建分析。 使用模板创建分析，您的团队可以使用该分析来了解如何改进事件响应。
会话文档	Session Manager	Session Manager 是 AWS Systems Manager 中的一项工具，使用会话文档确定要启动哪些类型的会话，例如端口转发会话、运行交互式命令的会话或创建 SSH 隧道的会话。 Systems Manager 支持的所有 Linux 和 Windows Server 操作系统都支持会话文档。macOS 的 EC2 实例支持以下命令文档： • `AWS-PasswordReset` • `AWS-StartInteractiveCommand` • `AWS-StartPortForwardingSession` • `AWS-StartPortForwardingSessionToSocket` • `AWS-StartSSHSession`

SSM 文档配额

有关 SSM 服务配额的更多信息，请参阅《Amazon Web Services 一般参考》中的 Systems Manager Service Quotas。

主题

- 文档组件
- 创建 SSM 文档内容
- 使用文档
- 参数处理问题的疑难解答

文档组件

本部分包含有关构成 SSM 文档的组件的信息。

内容

- 架构、功能和示例
- 数据元素和参数
- 命令文档插件参考

架构、功能和示例

AWS Systems Manager (SSM) 文档当前使用以下架构版本。

- Command 型文档可以使用 1.2、2.0 和 2.2 版架构。如果您使用架构 1.2 文档，我们建议您创建使用架构版本 2.2 的文件。
- Policy 型文档必须使用 2.0 版或更高版本的架构。
- Automation 型文档必须使用 0.3 版架构。
- Session 型文档必须使用 1.0 版架构。
- 您可以创建 JSON 或 YAML 格式的文档。

有关 Session 文档架构的更多信息，请参阅 会话文档架构。

通过为 Command 和 Policy 文档使用最新架构版本，您可以利用以下功能。

2.2 版架构文档功能

功能	详细信息
编辑文档	文档现在可以更新。如果版本为 1.2，对文档的任何更新都需要另存为其他名称。

功能	详细信息
自动版本控制	对文档的任何更新都会创建一个新版本。这不是架构版本，而是文档版本。
默认版本	如果您拥有某个文档的多个版本，可以指定哪个版本是默认文档。
顺序	文档中的插件或步骤按指定的顺序运行。
跨平台支持	跨平台支持可让您为同一个 SSM 文档中的不同插件指定不同操作系统。跨平台支持在某个步骤中使用 `precondition` 参数。
参数插值	允许在执行命令之前将字符串参数插值到环境变量中，从而更好地防范命令注入攻击。设置为 `ENV_VAR` 时，代理会创建一个包含参数值的环境变量 `SSM_parameter-name` 。

> **Note**
>
> 您必须将实例上的 AWS Systems Manager SSM Agent 更新到最新版本才能使用新的 Systems Manager 功能和 SSM 文档功能。有关更多信息，请参阅 使用 Run Command 更新 SSM Agent。

下表列出了主要架构版本之间的区别。

版本 1.2	版本 2.2（最新版本）	详细信息
runtimeConfig	mainSteps	在版本 2.2 中，`mainSteps` 部分替换了 `runtimeConfig` 部分。这些区域有：`mainSteps` 部分允许 Systems Manager 按顺序运行步骤。

版本 1.2	版本 2.2（最新版本）	详细信息
属性	inputs	在版本 2.2 中，inputs 部分替换了 properties 部分。inputs 部分接受步骤参数。
命令	runCommand	在版本 2.2 中，inputs 部分接收 runCommand 参数，而不是 commands 参数。
id	action	在版本 2.2 中，Action 替换了 ID。这只是更改了名称。
不适用	名称	在版本 2.2 中，name 是用户定义的任意步骤名称。

使用前提条件参数

在 2.2 版或更高版本架构中，您可以使用 precondition 参数为每个插件指定目标操作系统验证您在 SSM 文档中定义的输入参数。precondition 参数支持引用 SSM 文档的输入参数，platformType 使用Linux 的值、MacOS,和 Windows。仅支持 StringEquals 运算符。

对于使用 2.2 版或更高版本架构的文档，如果未指定 precondition，每个插件将被运行或跳过，具体取决于插件与操作系统的兼容性。插件与操作系统的兼容性在 precondition 之前被评估。对于使用 2.0 版或更低版本架构的文档，不兼容的插件将会引发错误。

例如，在 2.2 版架构文档中，如果未指定 precondition，将会列出 aws:runShellScript 插件，该步骤在 Linux 实例上运行，但系统会在 Windows Server 实例上跳过该步骤，因为 aws:runShellScript 与 Windows Server 实例不兼容。但是，对于 2.0 版架构文档，如果您指定 aws:runShellScript 插件，然后在 Windows Server 实例上运行文档，执行将会失败。本节稍后将介绍在 SSM 文档中使用前提条件参数的示例。

架构版本 2.2

顶级元素

以下示例使用 2.2 版架构显示 SSM 文档的顶级元素。

YAML

```
---
schemaVersion: "2.2"
description: A description of the document.
parameters:
  parameter 1:
    property 1: "value"
    property 2: "value"
  parameter 2:
    property 1: "value"
    property 2: "value"
mainSteps:
  - action: Plugin name
    name: A name for the step.
    inputs:
      input 1: "value"
      input 2: "value"
      input 3: "{{ parameter 1 }}"
```

JSON

```
{
   "schemaVersion": "2.2",
   "description": "A description of the document.",
   "parameters": {
       "parameter 1": {
          "property 1": "value",
          "property 2": "value"
       },
       "parameter 2":{
          "property 1": "value",
          "property 2": "value"
       }
    },
   "mainSteps": [
      {
         "action": "Plugin name",
         "name": "A name for the step.",
         "inputs": {
            "input 1": "value",
            "input 2": "value",
            "input 3": "{{ parameter 1 }}"
```

```
         }
      }
   ]
}
```

2.2 版架构示例

以下示例使用 aws:runPowerShellScript 插件在目标实例上运行 PowerShell 命令。

YAML

```
---
schemaVersion: "2.2"
description: "Example document"
parameters:
  Message:
    type: "String"
    description: "Example parameter"
    default: "Hello World"
    allowedValues:
    - "Hello World"
mainSteps:
  - action: "aws:runPowerShellScript"
    name: "example"
    inputs:
      timeoutSeconds: '60'
      runCommand:
      - "Write-Output {{Message}}"
```

JSON

```
{
   "schemaVersion": "2.2",
   "description": "Example document",
   "parameters": {
      "Message": {
         "type": "String",
         "description": "Example parameter",
         "default": "Hello World",
         "allowedValues": ["Hello World"]
      }
   },
```

```
   "mainSteps": [
      {
         "action": "aws:runPowerShellScript",
         "name": "example",
         "inputs": {
            "timeoutSeconds": "60",
            "runCommand": [
               "Write-Output {{Message}}"
            ]
         }
      }
   ]
}
```

架构版本 2.2 前提条件参数示例

2.2 版架构提供跨平台支持。这意味着在一个 SSM 文档内，您可以为不同的插件指定不同的操作系统。跨平台支持在某个步骤中使用 precondition 参数，如下例所示。您也可以使用 precondition 参数来验证您在 SSM 文档中定义的输入参数。您可以在以下第二个示例中看到此内容。

YAML

```
---
schemaVersion: '2.2'
description: cross-platform sample
mainSteps:
- action: aws:runPowerShellScript
  name: PatchWindows
  precondition:
    StringEquals:
    - platformType
    - Windows
  inputs:
    runCommand:
    - cmds
- action: aws:runShellScript
  name: PatchLinux
  precondition:
    StringEquals:
    - platformType
    - Linux
```

```
  inputs:
    runCommand:
    - cmds
```

JSON

```
{
   "schemaVersion": "2.2",
   "description": "cross-platform sample",
   "mainSteps": [
      {
         "action": "aws:runPowerShellScript",
         "name": "PatchWindows",
         "precondition": {
            "StringEquals": [
               "platformType",
               "Windows"
            ]
         },
         "inputs": {
            "runCommand": [
               "cmds"
            ]
         }
      },
      {
         "action": "aws:runShellScript",
         "name": "PatchLinux",
         "precondition": {
            "StringEquals": [
               "platformType",
               "Linux"
            ]
         },
         "inputs": {
            "runCommand": [
               "cmds"
            ]
         }
      }
   ]
}
```

YAML

```
---
schemaVersion: '2.2'
parameters:
  action:
    type: String
    allowedValues:
    - Install
    - Uninstall
  confirmed:
    type: String
    allowedValues:
    - True
    - False
mainSteps:
- action: aws:runShellScript
  name: InstallAwsCLI
  precondition:
    StringEquals:
    - "{{ action }}"
    - "Install"
  inputs:
    runCommand:
    - sudo apt install aws-cli
- action: aws:runShellScript
  name: UninstallAwsCLI
  precondition:
    StringEquals:
    - "{{ action }} {{ confirmed }}"
    - "Uninstall True"
  inputs:
    runCommand:
    - sudo apt remove aws-cli
```

JSON

```
{
   "schemaVersion": "2.2",
   "parameters": {
      "action": {
         "type": "String",
         "allowedValues": [
```

```
            "Install",
            "Uninstall"
         ]
      },
      "confirmed": {
         "type": "String",
         "allowedValues": [
            true,
            false
         ]
      }
   },
   "mainSteps": [
      {
         "action": "aws:runShellScript",
         "name": "InstallAwsCLI",
         "precondition": {
            "StringEquals": [
               "{{ action }}",
               "Install"
            ]
         },
         "inputs": {
            "runCommand": [
               "sudo apt install aws-cli"
            ]
         }
      },
      {
         "action": "aws:runShellScript",
         "name": "UninstallAwsCLI",
         "precondition": {
            "StringEquals": [
               "{{ action }} {{ confirmed }}",
               "Uninstall True"
            ]
         },
         "inputs": {
            "runCommand": [
               "sudo apt remove aws-cli"
            ]
         }
      }
   ]
```

```
}
```

2.2 版架构插值示例（3.3.2746.0 之前的 SSM Agent 版本）

在 3.3.2746.0 之前的 SSM Agent 版本中，代理会忽略 `interpolationType` 参数，转而执行原始字符串的替换。若要显式引用 `SSM_parameter-name`，则必须明确设置此项。以下 Linux 示例显式引用了环境变量 `SSM_Message`。

```
{
    "schemaVersion": "2.2",
    "description": "An example document",
    "parameters": {
        "Message": {
            "type": "String",
            "description": "Message to be printed",
            "default": "Hello",
            "interpolationType" : "ENV_VAR",
      "allowedPattern: "^[^"]*$"

        }
    },
    "mainSteps": [{
        "action": "aws:runShellScript",
        "name": "printMessage",
        "inputs": {
            "runCommand": [
              "if [ -z "${SSM_Message+x}" ]; then",
              "    export SSM_Message=\"{{Message}}\"",
              "fi",
              "",
              "echo $SSM_Message"
            ]
        }
    }
}
```

> **Note**
>
> 如果 SSM 文档未使用以下双大括号，则 `allowedPattern` 在技术上是不必要的：`{{ }}`

2.2 版架构 State Manager 示例

您可以将以下 SSM 文档用于State Manager（Systems Manager 中的一项工具）来下载并安装 ClamAV 防病毒软件。State Manager会强制实施特定配置，这意味着每次运行State Manager关联时，系统都会检查是否安装了 ClamAV 软件。如果未安装，State Manager 会重新运行本文档。

YAML

```
---
schemaVersion: '2.2'
description: State Manager Bootstrap Example
parameters: {}
mainSteps:
- action: aws:runShellScript
  name: configureServer
  inputs:
    runCommand:
    - sudo yum install -y httpd24
    - sudo yum --enablerepo=epel install -y clamav
```

JSON

```
{
   "schemaVersion": "2.2",
   "description": "State Manager Bootstrap Example",
   "parameters": {},
   "mainSteps": [
      {
         "action": "aws:runShellScript",
         "name": "configureServer",
         "inputs": {
            "runCommand": [
               "sudo yum install -y httpd24",
               "sudo yum --enablerepo=epel install -y clamav"
            ]
         }
      }
   ]
}
```

2.2 版架构清单示例

您可以将以下 SSM 文档用于 State Manager,以收集关于实例的清单元数据。

YAML

```
---
schemaVersion: '2.2'
description: Software Inventory Policy Document.
parameters:
  applications:
    type: String
    default: Enabled
    description: "(Optional) Collect data for installed applications."
    allowedValues:
    - Enabled
    - Disabled
  awsComponents:
    type: String
    default: Enabled
    description: "(Optional) Collect data for AWS Components like amazon-ssm-agent."
    allowedValues:
    - Enabled
    - Disabled
  networkConfig:
    type: String
    default: Enabled
    description: "(Optional) Collect data for Network configurations."
    allowedValues:
    - Enabled
    - Disabled
  windowsUpdates:
    type: String
    default: Enabled
    description: "(Optional) Collect data for all Windows Updates."
    allowedValues:
    - Enabled
    - Disabled
  instanceDetailedInformation:
    type: String
    default: Enabled
    description: "(Optional) Collect additional information about the instance,
 including
      the CPU model, speed, and the number of cores, to name a few."
    allowedValues:
    - Enabled
```

```
    - Disabled
  customInventory:
    type: String
    default: Enabled
    description: "(Optional) Collect data for custom inventory."
    allowedValues:
    - Enabled
    - Disabled
mainSteps:
- action: aws:softwareInventory
  name: collectSoftwareInventoryItems
  inputs:
    applications: "{{ applications }}"
    awsComponents: "{{ awsComponents }}"
    networkConfig: "{{ networkConfig }}"
    windowsUpdates: "{{ windowsUpdates }}"
    instanceDetailedInformation: "{{ instanceDetailedInformation }}"
    customInventory: "{{ customInventory }}"
```

JSON

```
{
   "schemaVersion": "2.2",
   "description": "Software Inventory Policy Document.",
   "parameters": {
      "applications": {
         "type": "String",
         "default": "Enabled",
         "description": "(Optional) Collect data for installed applications.",
         "allowedValues": [
            "Enabled",
            "Disabled"
         ]
      },
      "awsComponents": {
         "type": "String",
         "default": "Enabled",
         "description": "(Optional) Collect data for AWS Components like amazon-ssm-
agent.",
         "allowedValues": [
            "Enabled",
            "Disabled"
         ]
```

```
      },
      "networkConfig": {
         "type": "String",
         "default": "Enabled",
         "description": "(Optional) Collect data for Network configurations.",
         "allowedValues": [
            "Enabled",
            "Disabled"
         ]
      },
      "windowsUpdates": {
         "type": "String",
         "default": "Enabled",
         "description": "(Optional) Collect data for all Windows Updates.",
         "allowedValues": [
            "Enabled",
            "Disabled"
         ]
      },
      "instanceDetailedInformation": {
         "type": "String",
         "default": "Enabled",
         "description": "(Optional) Collect additional information about the
instance, including\nthe CPU model, speed, and the number of cores, to name a
few.",
         "allowedValues": [
            "Enabled",
            "Disabled"
         ]
      },
      "customInventory": {
         "type": "String",
         "default": "Enabled",
         "description": "(Optional) Collect data for custom inventory.",
         "allowedValues": [
            "Enabled",
            "Disabled"
         ]
      }
   },
   "mainSteps": [
      {
         "action": "aws:softwareInventory",
         "name": "collectSoftwareInventoryItems",
```

```
         "inputs": {
            "applications": "{{ applications }}",
            "awsComponents": "{{ awsComponents }}",
            "networkConfig": "{{ networkConfig }}",
            "windowsUpdates": "{{ windowsUpdates }}",
            "instanceDetailedInformation": "{{ instanceDetailedInformation }}",
            "customInventory": "{{ customInventory }}"
         }
      }
   ]
}
```

2.2 版架构 **AWS-ConfigureAWSPackage** 示例

以下示例显示 AWS-ConfigureAWSPackage 文档。这些区域有：mainSteps部分包括aws:configurePackage插件action步骤。

> **Note**
>
> 在 Linux 操作系统上，仅支持 AmazonCloudWatchAgent 和 AWSSupport-EC2Rescue 软件包。

YAML

```
---
schemaVersion: '2.2'
description: 'Install or uninstall the latest version or specified version of an AWS
  package. Available packages include the following: AWSPVDriver,
 AwsEnaNetworkDriver,
  AwsVssComponents, and AmazonCloudWatchAgent, and AWSSupport-EC2Rescue.'
parameters:
  action:
    description: "(Required) Specify whether or not to install or uninstall the
 package."
    type: String
    allowedValues:
    - Install
    - Uninstall
  name:
    description: "(Required) The package to install/uninstall."
```

```
    type: String
    allowedPattern: "^arn:[a-z0-9][-.a-z0-9]{0,62}:[a-z0-9][-.a-z0-9]{0,62}:([a-
z0-9][-.a-z0-9]{0,62})?:([a-z0-9][-.a-z0-9]{0,62})?:package\\/[a-zA-Z][a-zA-Z0-9\\-
_]{0,39}$|^[a-zA-Z][a-zA-Z0-9\\-_]{0,39}$"
  version:
    type: String
    description: "(Optional) A specific version of the package to install or
 uninstall."
mainSteps:
- action: aws:configurePackage
  name: configurePackage
  inputs:
    name: "{{ name }}"
    action: "{{ action }}"
    version: "{{ version }}"
```

JSON

```
{
   "schemaVersion": "2.2",
   "description": "Install or uninstall the latest version or specified version
 of an AWS package. Available packages include the following: AWSPVDriver,
 AwsEnaNetworkDriver, AwsVssComponents, and AmazonCloudWatchAgent, and AWSSupport-
EC2Rescue.",
   "parameters": {
      "action": {
         "description":"(Required) Specify whether or not to install or uninstall
 the package.",
         "type":"String",
         "allowedValues":[
            "Install",
            "Uninstall"
         ]
      },
      "name": {
         "description": "(Required) The package to install/uninstall.",
         "type": "String",
         "allowedPattern": "^arn:[a-z0-9][-.a-z0-9]{0,62}:[a-z0-9][-.a-z0-9]{0,62}:
([a-z0-9][-.a-z0-9]{0,62})?:([a-z0-9][-.a-z0-9]{0,62})?:package\\/[a-zA-Z][a-zA-
Z0-9\\-_]{0,39}$|^[a-zA-Z][a-zA-Z0-9\\-_]{0,39}$"
      },
      "version": {
         "type": "String",
```

```
            "description": "(Optional) A specific version of the package to install or
 uninstall."
         }
      },
      "mainSteps":[
         {
            "action": "aws:configurePackage",
            "name": "configurePackage",
            "inputs": {
               "name": "{{ name }}",
               "action": "{{ action }}",
               "version": "{{ version }}"
            }
         }
      ]
}
```

架构版本 1.2

以下示例显示 1.2 版架构文档的顶级元素。

```
{
   "schemaVersion":"1.2",
   "description":"A description of the SSM document.",
   "parameters":{
      "parameter 1":{
         "one or more parameter properties"
      },
      "parameter 2":{
         "one or more parameter properties"
      },
      "parameter 3":{
         "one or more parameter properties"
      }
   },
   "runtimeConfig":{
      "plugin 1":{
         "properties":[
            {
               "one or more plugin properties"
            }
         ]
      }
```

```
    }
}
```

1.2 版架构 **aws:runShellScript** 示例

以下示例显示 AWS-RunShellScript SSM 文档。runtimeConfig 部分包含 aws:runShellScript 插件。

```
{
    "schemaVersion":"1.2",
    "description":"Run a shell script or specify the commands to run.",
    "parameters":{
        "commands":{
            "type":"StringList",
            "description":"(Required) Specify a shell script or a command to run.",
            "minItems":1,
            "displayType":"textarea"
        },
        "workingDirectory":{
            "type":"String",
            "default":"",
            "description":"(Optional) The path to the working directory on your
 instance.",
            "maxChars":4096
        },
        "executionTimeout":{
            "type":"String",
            "default":"3600",
            "description":"(Optional) The time in seconds for a command to complete
 before it is considered to have failed. Default is 3600 (1 hour). Maximum is 172800
 (48 hours).",
            "allowedPattern":"([1-9][0-9]{0,3})|(1[0-9]{1,4})|(2[0-7][0-9]{1,3})|
(28[0-7][0-9]{1,2})|(28800)"
        }
    },
    "runtimeConfig":{
        "aws:runShellScript":{
            "properties":[
                {
                    "id":"0.aws:runShellScript",
                    "runCommand":"{{ commands }}",
                    "workingDirectory":"{{ workingDirectory }}",
                    "timeoutSeconds":"{{ executionTimeout }}"
```

```
                }
            ]
        }
    }
}
```

架构版本 0.3

顶级元素

以下示例显示 JSON 格式的架构 0.3 版自动化运行手册的顶级元素。

```
{
    "description": "document-description",
    "schemaVersion": "0.3",
    "assumeRole": "{{assumeRole}}",
    "parameters": {
        "parameter1": {
            "type": "String",
            "description": "parameter-1-description",
            "default": ""
        },
        "parameter2": {
            "type": "String",
            "description": "parameter-2-description",
            "default": ""
        }
    },
    "variables": {
        "variable1": {
            "type": "StringMap",
            "description": "variable-1-description",
            "default": {}
        },
        "variable2": {
            "type": "String",
            "description": "variable-2-description",
            "default": "default-value"
        }
    },
    "mainSteps": [
        {
            "name": "myStepName",
            "action": "action-name",
```

```
            "maxAttempts": 1,
            "inputs": {
                "Handler": "python-only-handler-name",
                "Runtime": "runtime-name",
                "Attachment": "script-or-zip-name"
            },
            "outputs": {
                "Name": "output-name",
                "Selector": "selector.value",
                "Type": "data-type"
            }
        }
    ],
    "files": {
        "script-or-zip-name": {
            "checksums": {
                "sha256": "checksum"
            },
            "size": 1234
        }
    }
}
```

YAML 自动化运行手册示例

以下示例显示了 YAML 格式的自动化运行手册的内容。0.3 版文档架构的此工作示例还演示了如何使用 Markdown 格式化文档描述。

```
description: >-
  ##Title: LaunchInstanceAndCheckState

  -----

  **Purpose**: This Automation runbook first launches an EC2 instance
  using the AMI ID provided in the parameter ```imageId```. The second step of
  this document continuously checks the instance status check value for the
  launched instance until the status ```ok``` is returned.


  ##Parameters:

  -----
```

```
  Name | Type | Description | Default Value

  ------------- | ------------- | ------------- | -------------

  assumeRole | String | (Optional) The ARN of the role that allows Automation to
  perform the actions on your behalf. | -

  imageId  | String | (Optional) The AMI ID to use for launching the instance.
  The default value uses the latest Amazon Linux AMI ID available. | {{
  ssm:/aws/service/ami-amazon-linux-latest/amzn-ami-hvm-x86_64-gp2 }}
schemaVersion: '0.3'
assumeRole: 'arn:aws:iam::111122223333::role/AutomationServiceRole'
parameters:
  imageId:
    type: String
    default: '{{ ssm:/aws/service/ami-amazon-linux-latest/amzn-ami-hvm-x86_64-gp2 }}'
    description: >-
      (Optional) The AMI ID to use for launching the instance. The default value
      uses the latest released Amazon Linux AMI ID.
  tagValue:
    type: String
    default: ' LaunchedBySsmAutomation'
    description: >-
      (Optional) The tag value to add to the instance. The default value is
      LaunchedBySsmAutomation.
  instanceType:
    type: String
    default: t2.micro
    description: >-
      (Optional) The instance type to use for the instance. The default value is
      t2.micro.
mainSteps:
  - name: LaunchEc2Instance
    action: 'aws:executeScript'
    outputs:
      - Name: payload
        Selector: $.Payload
        Type: StringMap
    inputs:
      Runtime: python3.8
      Handler: launch_instance
      Script: ''
      InputPayload:
        image_id: '{{ imageId }}'
```

```
        tag_value: '{{ tagValue }}'
        instance_type: '{{ instanceType }}'
      Attachment: launch.py
    description: >-
      **About This Step**


      This step first launches an EC2 instance using the ```aws:executeScript```
      action and the provided python script.
  - name: WaitForInstanceStatusOk
    action: 'aws:executeScript'
    inputs:
      Runtime: python3.8
      Handler: poll_instance
      Script: |-
        def poll_instance(events, context):
          import boto3
          import time

          ec2 = boto3.client('ec2')

          instance_id = events['InstanceId']

          print('[INFO] Waiting for instance status check to report ok', instance_id)

          instance_status = "null"

          while True:
            res = ec2.describe_instance_status(InstanceIds=[instance_id])

            if len(res['InstanceStatuses']) == 0:
              print("Instance status information is not available yet")
              time.sleep(5)
              continue

            instance_status = res['InstanceStatuses'][0]['InstanceStatus']['Status']

            print('[INFO] Polling to get status of the instance', instance_status)

            if instance_status == 'ok':
              break

            time.sleep(10)
```

```
          return {'Status': instance_status, 'InstanceId': instance_id}
      InputPayload: '{{ LaunchEc2Instance.payload }}'
    description: >-
      **About This Step**


      The python script continuously polls the instance status check value for
      the instance launched in Step 1 until the ```ok``` status is returned.
files:
  launch.py:
    checksums:
      sha256: 18871b1311b295c43d0f...[truncated]...772da97b67e99d84d342ef4aEXAMPLE
```

安全处理参数示例

以下示例演示使用环境变量 interpolationType 安全处理参数。

基本安全命令执行

本示例说明如何安全处理命令参数：

> **Note**
>
> 在不使用以下双大括号的 SSM 文档中，allowedPattern 在技术上是不必要的：{{ }}

YAML

```
---

schemaVersion: '2.2'
description: An example document.
parameters:
  Message:
    type: String
    description: "Message to be printed"
    default: Hello
    interpolationType: ENV_VAR
    allowedPattern: "^[^"]*$"
mainSteps:
  - action: aws:runShellScript
    name: printMessage
    precondition:
```

```
      StringEquals:
        - platformType
        - Linux
    inputs:
      runCommand:
        - echo {{Message}}
```

JSON

```
{
    "schemaVersion": "2.2",
    "description": "An example document.",
    "parameters": {
        "Message": {
            "type": "String",
            "description": "Message to be printed",
            "default": "Hello",
            "interpolationType": "ENV_VAR",
            "allowedPattern": "^[^"]*$"
        }
    },
    "mainSteps": [{
        "action": "aws:runShellScript",
        "name": "printMessage",
        "precondition": {
           "StringEquals": ["platformType", "Linux"]
        },
        "inputs": {
            "runCommand": [
              "echo {{Message}}"
            ]
        }
    }]
}
```

在解释性语言中使用参数

本示例演示在 Python 中安全处理参数：

YAML

```
---
```

```
schemaVersion: '2.2'
description: 'Secure Python script execution'
parameters:
  inputData:
    type: String
    description: 'Input data for processing'
    interpolationType: 'ENV_VAR'
mainSteps:
  - action: aws:runPowerShellScript
    name: runPython
    inputs:
      runCommand:
        - |
          python3 -c '
          import os
          import json

          # Safely access parameter through environment variable
          input_data = os.environ.get("SSM_inputData", "")

          # Process the data
          try:
              processed_data = json.loads(input_data)
              print(f"Successfully processed: {processed_data}")
          except json.JSONDecodeError:
              print("Invalid JSON input")
          '
```

向后兼容性示例

本示例展示如何安全处理参数，同时保持向后兼容性：

YAML

```
---
schemaVersion: '2.2'
description: 'Backwards compatible secure parameter handling'
parameters:
  userInput:
    type: String
    description: 'User input to process'
    interpolationType: 'ENV_VAR'
    allowedPattern: '^[^"]*$'
```

```
mainSteps:
  - action: aws:runShellScript
    name: processInput
    inputs:
      runCommand:
        - |
          # Handle both modern and legacy agent versions
          if [ -z "${SSM_userInput+x}" ]; then
              # Legacy agent - fall back to direct parameter reference
              export SSM_userInput="{{userInput}}"
          fi

          # Process the input securely
          echo "Processing input: $SSM_userInput"
```

> **Note**
>
> 在不使用以下双大括号的 SSM 文档中，allowedPattern 在技术上是不必要的：{{ }}

参数安全的最佳实践

在处理 SSM 文档中的参数时，请遵循以下最佳实践：

- 使用环境变量插值：即将用于命令执行的字符串参数始终使用 interpolationType: "ENV_VAR"。
- 实现输入验证：使用 allowedPattern 将参数值限制为安全模式。
- 处理遗留系统：为不支持环境变量插值的旧版 SSM Agent 添加回退逻辑。
- 转义特殊字符：在命令中使用参数值时，请对特殊字符进行恰当转义以防止被 shell 解释。
- 限制参数范围：为用例使用尽可能严格的参数模式。

数据元素和参数

本主题介绍 SSM 文档中使用的数据元素。用于创建文档的架构版本定义了文档接受的语法和数据元素。我们建议您对命令文档使用架构版本 2.2 或更高版本。自动化运行手册使用架构版本 0.3。此外，自动化运行手册还支持使用 Markdown（一种标记语言），它允许您为文档和文档中的各个步骤添加

Wiki 样式的描述。有关使用 Markdown 的详细信息，请参阅《AWS Management Console 入门指南》中的在控制台中使用 Markdown。

以下部分介绍了 SSM 文档中可以包含的数据元素。

顶级数据元素

schemaVersion

要使用的架构版本。

类型：版本

必需：是

description

您提供的描述文档目的的信息。您还可以使用此字段来指定参数是否需要一个值才能运行文档，或者为参数提供值是否为可选项。可在本主题的所有示例中查看必需参数和可选参数。

类型：字符串

必需：否

参数

定义文档接受的参数的结构。

为增强处理字符串参数的安全性，可通过指定 `interpolationType` 属性来使用环境变量插值。设置为 `ENV_VAR` 时，系统会创建一个包含参数值的环境变量 `SSM_parameter-name`。

下面是一个使用环境变量 `interpolationType` 的参数的示例：

```
{
    "schemaVersion": "2.2",
    "description": "An example document.",
    "parameters": {
        "Message": {
            "type": "String",
            "description": "Message to be printed",
            "default": "Hello",
            "interpolationType" : "ENV_VAR",
            "allowedPattern": "^[^"]*$"
```

```
        }
    },
    "mainSteps": [{
        "action": "aws:runShellScript",
        "name": "printMessage",
        "precondition" : {
           "StringEquals" : ["platformType", "Linux"]
        },
        "inputs": {
            "runCommand": [
              "echo {{Message}}"
            ]
        }
    }
}
```

> **Note**
>
> 在不使用以下双大括号的 SSM 文档中，`allowedPattern` 在技术上是不必要的：`{{ }}`

对于经常使用的参数，建议将这些参数存储在 Parameter Store（AWS Systems Manager 中的一项工具）中。然后，可以在文档中定义参数，并引用 Parameter Store 参数作为默认值。要引用 Parameter Store 参数，请使用以下语法。

```
{{ssm:parameter-name}}
```

可以使用参数通过与任何其他文档参数相同的方式引用 Parameter Store 参数。在以下示例中，`commands` 参数的默认值是 Parameter Store 参数 `myShellCommands`。如果将 `commands` 参数指定为 `runCommand` 字符串，文档将运行存储在 `myShellCommands` 参数中的命令。

YAML

```
---
schemaVersion: '2.2'
description: runShellScript with command strings stored as Parameter Store
 parameter
parameters:
  commands:
    type: StringList
```

```
    description: "(Required) The commands to run on the instance."
    default: ["{{ ssm:myShellCommands }}"],
            interpolationType : 'ENV_VAR'
            allowedPattern: '^[^"]*$'

mainSteps:
- action: aws:runShellScript
  name: runShellScriptDefaultParams
  inputs:
    runCommand:"{{ commands }}"
```

JSON

```
{
    "schemaVersion": "2.2",
    "description": "runShellScript with command strings stored as Parameter Store
 parameter",
    "parameters": {
      "commands": {
        "type": "StringList",
        "description": "(Required) The commands to run on the instance.",
        "default": ["{{ ssm:myShellCommands }}"],
        "interpolationType" : "ENV_VAR"
      }
    },
    "mainSteps": [
      {
        "action": "aws:runShellScript",
        "name": "runShellScriptDefaultParams",
        "inputs": {
            "runCommand": [
              "{{ commands }}"
          ]
        }
      }
    ]
  }
```

> **Note**
>
> 您可以在文档的 `parameters` 部分引用 `String` 和 `StringList` Parameter Store 参数。您不能引用 `SecureString` Parameter Store 参数。

有关 Parameter Store 的更多信息，请参阅 AWS Systems Manager Parameter Store。

类型：结构

parameters 结构接受以下字段和值：

- type：(必需) 允许的值包括：String、StringList、Integer、Boolean、MapList 和 StringMap。要查看每种类型的示例，请参阅下一节中的 文档参数 type 示例。

  > **Note**
  >
  > 命令类型文档仅支持 String 和 StringList 参数类型。

- description：(可选) 关于参数的描述。
- default：（可选）参数的默认值或对 Parameter Store 中参数的引用。
- allowedValues：（可选）参数允许的值数组。定义参数的允许值将验证用户输入。如果用户输入了不允许的值，则执行将无法启动。

  YAML

  ```
  DirectoryType:
    type: String
    description: "(Required) The directory type to launch."
    default: AwsMad
    allowedValues:
    - AdConnector
    - AwsMad
    - SimpleAd
  ```

  JSON

  ```
  "DirectoryType": {
    "type": "String",
    "description": "(Required) The directory type to launch.",
    "default": "AwsMad",
    "allowedValues": [
      "AdConnector",
      "AwsMad",
      "SimpleAd"
    ]
  }
  ```

- `allowedPattern`：（可选）验证用户输入是否与参数的定义模式匹配的正则表达式。如果用户输入与允许的模式不匹配，则执行无法启动。

  > **Note**
  >
  > Systems Manager 会执行两次 `allowedPattern` 验证。第一次验证是在您使用文档时利用 Java 正则表达式库在 API 级别进行。第二次验证是在处理文档之前通过使用 GO 正则表达式库在 SSM Agent 上进行。

  YAML

  ```
  InstanceId:
    type: String
    description: "(Required) The instance ID to target."
    allowedPattern: "^i-(?:[a-f0-9]{8}|[a-f0-9]{17})$"
    default: ''
  ```

  JSON

  ```
  "InstanceId": {
    "type": "String",
    "description": "(Required) The instance ID to target.",
    "allowedPattern": "^i-(?:[a-f0-9]{8}|[a-f0-9]{17})$",
    "default": ""
  }
  ```

- `displayType`：（可选）用于在 AWS Management Console 中显示 `textfield` 或 `textarea`。`textfield` 是单行文本框。`textarea` 是多行文本区域。
- `minItems`：(可选) 允许的最小项目数。
- `maxItems`：(可选) 允许的最大项目数。
- `minChars`：(可选) 允许的最小参数字符数。
- `maxChars`：(可选) 允许的最大参数字符数。
- `interpolationType`：（可选）定义在执行命令之前如何处理参数值。如果设置为 `ENV_VAR`，则可以将参数值用作名为 `SSM_`*`parameter-name`* 的环境变量。此功能通过将参数值视为文字字符串，从而帮助防范命令注入攻击。

  类型：字符串

有效值：ENV_VAR

必需：否

variables

（仅限架构版本 0.3）您可以在自动化运行手册的整个步骤中引用或更新的值。变量与参数类似，但在重要方面有所区别。参数值在运行手册的上下文中是静态的，但是变量的值可以在运行手册的上下文中更改。更新变量的值时，数据类型必须与定义的数据类型相匹配。有关更新自动化中的变量值的信息，请参阅 aws:updateVariable – 更新运行手册变量的值

类型：Boolean | Integer | MapList | String | StringList | StringMap

必需：否

YAML

```
variables:
    payload:
        type: StringMap
        default: "{}"
```

JSON

```
{
    "variables": [
        "payload": {
            "type": "StringMap",
            "default": "{}"
        }
    ]
}
```

runtimeConfig

(仅限 1.2 版架构) 由一个或多个 Systems Manager 插件应用的实例的配置。不保证插件按顺序运行。

类型：Dictionary<string,PluginConfiguration>

必需：否

mainSteps

（仅架构版本 0.3、2.0 和 2.2）可以包含多个步骤（插件）的对象。插件在步骤内定义。步骤按文档中列出的先后顺序运行。

类型：Dictionary<string,PluginConfiguration>

必需：是

outputs

（仅架构版本 0.3）通过执行本文档而生成的可用于其他进程的数据。例如，如果文档创建新的 AMI，您可以指定 “CreateImage.ImageId” 作为输出值，然后使用该输出以在后续自动化执行中创建新的实例。有关输出的更多信息，请参阅 使用操作输出作为输入。

类型：Dictionary<string,OutputConfiguration>

必需：否

文件

（仅架构版本 0.3）附加到文档并在自动化执行期间运行的脚本文件（及其校验和）。仅适用于包含 `aws:executeScript` 操作且已在一个或多个步骤中指定附件的文档。

要了解自动化运行手册支持的运行时，请参阅 `aws:executeScript` - 运行脚本。有关在自动化运行手册中包含脚本的更多信息，请参阅 在运行手册中使用脚本 和 自动化运行手册的视觉对象设计体验。

使用附件创建自动化运行手册时，可以使用 `--attachments` 选项（对于 AWS CLI）或 `Attachments`（对于 API 和开发工具包）指定附件文件。您可以为存储在 Amazon Simple Storage Service（Amazon S3）存储桶中的 SSM 文档和文件指定文件位置。有关更多信息，请参阅 AWS Systems Manager API 参考中的附件。

YAML

```
---
files:
  launch.py:
    checksums:
      sha256: 18871b1311b295c43d0f...
[truncated]...772da97b67e99d84d342ef4aEXAMPLE
```

JSON

```
"files": {
    "launch.py": {
        "checksums": {
            "sha256": "18871b1311b295c43d0f...
[truncated]...772da97b67e99d84d342ef4aEXAMPLE"
        }
    }
}
```

类型：Dictionary<string,FilesConfiguration>

必需：否

文档参数 **type** 示例

SSM 文档中的参数类型是静态的。这意味着参数类型在定义后无法更改。将参数用于 SSM 文档插件时，不能在插件的输入中动态更改参数的类型。例如，您不能在 aws:runShellScript 插件的 runCommand 输入中引用 Integer 参数，因为此输入接受字符串或字符串列表。要将参数用于插件输入，参数类型必须与接受的类型匹配。例如，您必须为aws:updateSsmAgent 插件的 allowDowngrade 输入指定 Boolean 类型参数。如果参数类型与插件的输入类型不匹配，则 SSM 文档无法验证，并且系统不会创建文档。在其他插件或 AWS Systems Manager 自动化操作的输入中使用下游参数时也是如此。例如，您不能引用 aws:runDocument 插件的 documentParameters 输入内的 StringList 参数。documentParameters 输入接受字符串映射，即使下游 SSM 文档参数类型是 StringList 参数并与您要引用的参数匹配。

将参数用于自动化操作时，大多数情况下创建 SSM 文档时不会验证参数类型。只有在使用 aws:runCommand 操作的情况下，才会在创建 SSM 文档时验证参数类型。在所有其他情况下，在运行操作之前验证该操作的输入时，会在自动化执行期间进行参数验证。例如，如果输入参数为 String 并将其引用为 aws:runInstances 操作 MaxInstanceCount 输入的值，则会创建 SSM 文档。但是，在运行该文档期间，验证 aws:runInstances 操作时自动化将失败，因为 MaxInstanceCount 输入需要 Integer。

下面是每个参数的示例 type。

字符串

使用引号括起来的零个或多个 Unicode 字符序列。例如，"i-1234567890abcdef0"。使用反斜杠转义。

字符串参数可包括一个值为 ENV_VAR 的可选 interpolationType 字段，以启用环境变量插值增强安全性。

YAML

```
---
InstanceId:
  type: String
  description: "(Optional) The target EC2 instance ID."
  interpolationType: ENV_VAR
```

JSON

```
"InstanceId":{
  "type":"String",
  "description":"(Optional) The target EC2 instance ID.",
  "interpolationType": "ENV_VAR"
}
```

StringList

以逗号分隔的字符串项目列表。例如，["cd ~", "pwd"]。

YAML

```
---
commands:
  type: StringList
  description: "(Required) Specify a shell script or a command to run."
  default: ""
  minItems: 1
  displayType: textarea
```

JSON

```
"commands":{
  "type":"StringList",
  "description":"(Required) Specify a shell script or a command to run.",
  "minItems":1,
  "displayType":"textarea"
}
```

布尔值

仅接受 `true` 或 `false`。不接受 "true" 或 0。

YAML

```
---
canRun:
  type: Boolean
  description: ''
  default: true
```

JSON

```
"canRun": {
  "type": "Boolean",
  "description": "",
  "default": true
}
```

整数

整数。不接受小数（例如 3.14159）或使用引号的数字（例如 "3"）。

YAML

```
---
timeout:
  type: Integer
  description: The type of action to perform.
  default: 100
```

JSON

```
"timeout": {
  "type": "Integer",
  "description": "The type of action to perform.",
  "default": 100
}
```

StringMap

键到值的映射。密钥和值必须是字符串。例如，{"Env": "Prod"}。

YAML

```
---
notificationConfig:
  type: StringMap
  description: The configuration for events to be notified about
  default:
    NotificationType: 'Command'
    NotificationEvents:
    - 'Failed'
    NotificationArn: "$dependency.topicArn"
  maxChars: 150
```

JSON

```
"notificationConfig" : {
  "type" : "StringMap",
  "description" : "The configuration for events to be notified about",
  "default" : {
    "NotificationType" : "Command",
    "NotificationEvents" : ["Failed"],
    "NotificationArn" : "$dependency.topicArn"
  },
  "maxChars" : 150
}
```

MapList

StringMap 对象的列表。

YAML

```
blockDeviceMappings:
  type: MapList
  description: The mappings for the create image inputs
  default:
  - DeviceName: "/dev/sda1"
    Ebs:
      VolumeSize: "50"
  - DeviceName: "/dev/sdm"
    Ebs:
      VolumeSize: "100"
  maxItems: 2
```

JSON

```
"blockDeviceMappings":{
  "type":"MapList",
  "description":"The mappings for the create image inputs",
  "default":[
    {
      "DeviceName":"/dev/sda1",
      "Ebs":{
        "VolumeSize":"50"
      }
    },
    {
      "DeviceName":"/dev/sdm",
      "Ebs":{
        "VolumeSize":"100"
      }
    }
  ],
  "maxItems":2
}
```

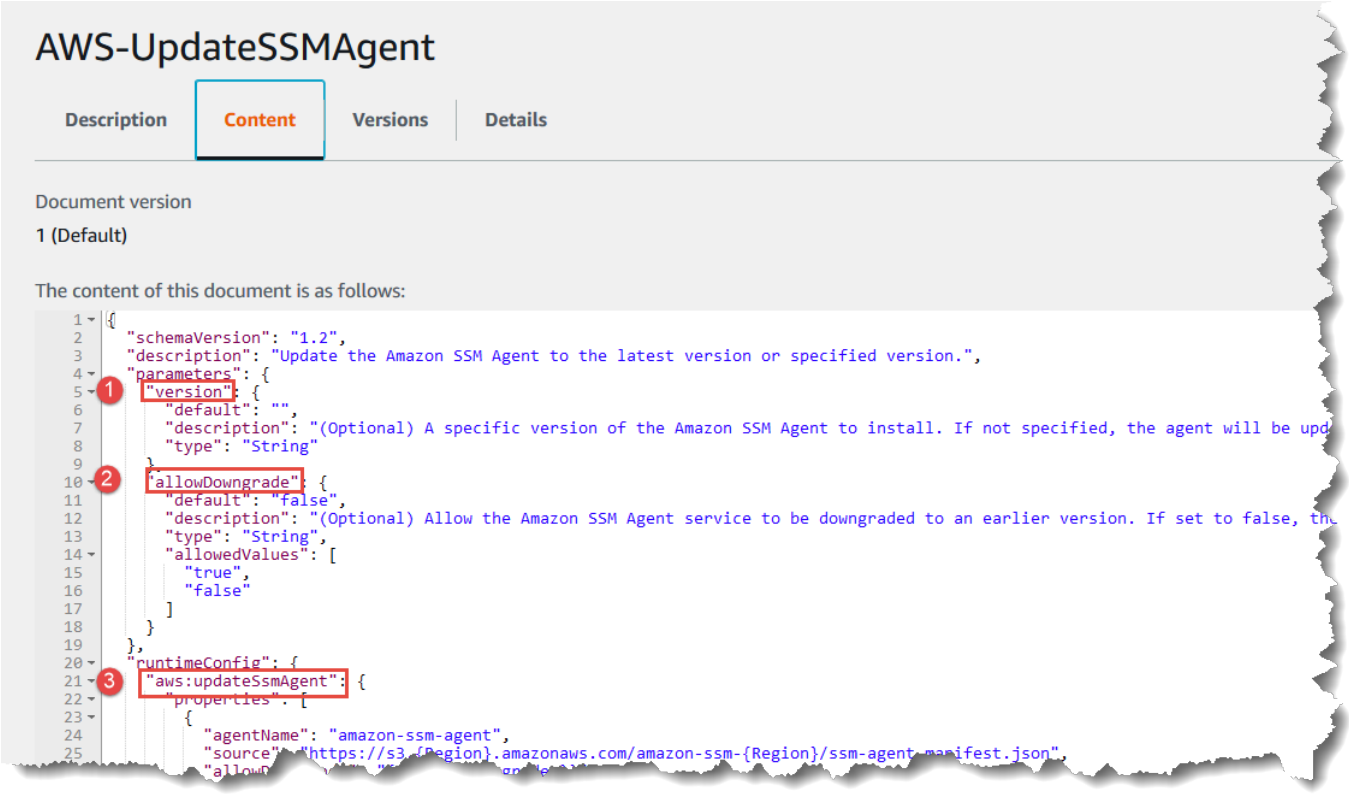
查看 SSM 命令文档内容

要预览需要的和可选参数 AWS Systems Manager(SSM) 命令文档，除了文档运行的操作外，您还可以在 Systems Manager 控制台中查看此文档的内容。

查看 SSM 命令文档内容

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择文档。
3. 在搜索框中，选择文档类型，然后选择命令。
4. 选择文档的名称，然后选择内容选项卡。
5. 在内容字段中，查看文档的可用参数和操作步骤。

   例如，下图显示 (1) `version` 和 (2) `allowDowngrade` 是 `AWS-UpdateSSMAgent` 文档的可选参数，并且文档运行的第一个操作是 (3) `aws:updateSsmAgent`。



命令文档插件参考

此参考介绍可在 AWS Systems Manager (SSM) 命令类型文档中指定的插件。这些插件不能在使自动化操作的 SSM 自动化运行手册中使用。有关 AWS Systems Manager 自动化操作的信息，请参阅 Systems Manager 自动化操作参考。

Systems Manager 通过读取 SSM 文档的内容确定在托管实例上执行的操作。每个文档都包含代码执行部分。根据文档的架构版本，此代码执行部分可能包含一个或多个插件或步骤。为了便于理解本帮助主题，我们将这些插件和步骤都称为插件。本部分包含有关每个 Systems Manager 插件的信息。有关文档的详细信息（包括有关创建文档和架构版本之间的差异的信息），请参阅 AWS Systems Manager 文档。

对于接受字符串参数（例如 `aws:runShellScript` 和 `aws:runPowerShellScript`）的插件，可以使用 `interpolationType` 参数增强安全性，方法是将参数输入视为字符串文本值，而非可能的可执行命令。例如：

```
{
    "schemaVersion": "2.2",
    "description": "runShellScript with command strings stored as Parameter Store
 parameter",
```

```
    "parameters": {
      "commands": {
        "type": "StringList",
        "description": "(Required) The commands to run on the instance.",
        "default": ["{{ ssm:myShellCommands }}"],
        "interpolationType" : "ENV_VAR"
      }
    },
    //truncated
}
```

> **Note**
>
> 此处介绍的部分插件仅在 Windows Server 实例或 Linux 实例上运行。每个插件都记录了平台依赖关系。
>
> macOS 的 Amazon Elastic Compute Cloud (Amazon EC2) 实例支持以下文档插件：
>
> - `aws:refreshAssociation`
> - `aws:runShellScript`
> - `aws:runPowerShellScript`
> - `aws:softwareInventory`
> - `aws:updateSsmAgent`

内容

- 共享输入
- aws:applications
- aws:cloudWatch
- aws:configureDocker
- aws:configurePackage
- aws:domainJoin
- aws:downloadContent
- aws:psModule
- aws:refreshAssociation
- aws:runDockerAction
- aws:runDocument

- aws:runPowerShellScript
- aws:runShellScript
- aws:softwareInventory
- aws:updateAgent
- aws:updateSsmAgent

共享输入

仅在版本 3.0.502 和更高版本的 SSM Agent 中，所有插件都可以使用以下输入：

最终步骤

您希望文档运行的最后一步。如果此输入是为某个步骤定义的，则它优先于 onFailure 或 onSuccess 输入中指定的 exit 值。为了使具有此输入的步骤按预期运行，该步骤必须是文档 mainSteps 中定义的最后一个步骤。

类型：布尔值

有效值：true | false

必需：否

onFailure

如果您为插件指定此输入的 exit 值并且步骤失败，则步骤状态会显示失败，并且文档不会运行任何剩余步骤，除非 finallyStep 已定义。如果您为插件指定此输入的 successAndExit 值并且步骤失败，则步骤状态会显示成功，并且文档不会运行任何剩余的步骤，除非 finallyStep 已定义。

类型：字符串

有效值：exit | successAndExit

必需：否

onSuccess

如果您为插件指定此输入并且步骤成功运行，则文档不会运行任何剩余的步骤，除非 finallyStep 已定义。

类型：字符串

有效值：exit

必需：否

YAML

```
---
schemaVersion: '2.2'
description: Shared inputs example
parameters:
  customDocumentParameter:
    type: String
    description: Example parameter for a custom Command-type document.
mainSteps:
- action: aws:runDocument
  name: runCustomConfiguration
  inputs:
    documentType: SSMDocument
    documentPath: "yourCustomDocument"
    documentParameters: '"documentParameter":{{customDocumentParameter}}'
    onSuccess: exit
- action: aws:runDocument
  name: ifConfigurationFailure
  inputs:
    documentType: SSMDocument
    documentPath: "yourCustomRepairDocument"
    onFailure: exit
- action: aws:runDocument
  name: finalConfiguration
  inputs:
    documentType: SSMDocument
    documentPath: "yourCustomFinalDocument"
    finallyStep: true
```

JSON

```
{
   "schemaVersion": "2.2",
   "description": "Shared inputs example",
   "parameters": {
      "customDocumentParameter": {
         "type": "String",
         "description": "Example parameter for a custom Command-type document."
      }
```

```
   },
   "mainSteps":[
      {
         "action": "aws:runDocument",
         "name": "runCustomConfiguration",
         "inputs": {
            "documentType": "SSMDocument",
            "documentPath": "yourCustomDocument",
            "documentParameters": "\"documentParameter\":
{{customDocumentParameter}}",
            "onSuccess": "exit"
         }
      },
      {
         "action": "aws:runDocument",
         "name": "ifConfigurationFailure",
         "inputs": {
            "documentType": "SSMDocument",
            "documentPath": "yourCustomRepairDocument",
            "onFailure": "exit"
         }
      },
      {
         "action": "aws:runDocument",
         "name":"finalConfiguration",
         "inputs": {
            "documentType": "SSMDocument",
            "documentPath": "yourCustomFinalDocument",
            "finallyStep": true
         }
      }
   ]
}
```

aws:applications

在 EC2 实例上安装、修复或卸载应用程序。此插件仅在 Windows Server 操作系统中运行。

语法

Schema 2.2

YAML

```
---
schemaVersion: '2.2'
description: aws:applications plugin
parameters:
  source:
    description: "(Required) Source of msi."
    type: String
mainSteps:
- action: aws:applications
  name: example
  inputs:
    action: Install
    source: "{{ source }}"
```

JSON

```
{
  "schemaVersion":"2.2",
  "description":"aws:applications",
  "parameters":{
    "source":{
    "description":"(Required) Source of msi.",
    "type":"String"
    }
  },
  "mainSteps":[
    {
      "action":"aws:applications",
      "name":"example",
      "inputs":{
        "action":"Install",
        "source":"{{ source }}"
      }
    }
  ]
}
```

Schema 1.2

YAML

```
---
runtimeConfig:
  aws:applications:
    properties:
    - id: 0.aws:applications
      action: "{{ action }}"
      parameters: "{{ parameters }}"
      source: "{{ source }}"
      sourceHash: "{{ sourceHash }}"
```

JSON

```
{
   "runtimeConfig":{
      "aws:applications":{
         "properties":[
            {
               "id":"0.aws:applications",
               "action":"{{ action }}",
               "parameters":"{{ parameters }}",
               "source":"{{ source }}",
               "sourceHash":"{{ sourceHash }}"
            }
         ]
      }
   }
}
```

属性

action

要执行的操作。

类型：Enum

有效值：`Install` | `Repair` | `Uninstall`

必需：是

参数

安装程序的参数。

类型：字符串

必需：否

源

应用程序的 .msi 文件的 URL。

类型：字符串

必需：是

sourceHash

.msi 文件的 SHA256 哈希值。

类型：字符串

必需：否

aws:cloudWatch

从中导出数据 Windows Server 添加到 Amazon CloudWatch 视台或 Amazon CloudWatch Logs 中，并使用云监控指标监控数据。此插件仅在 Windows Server 操作系统中运行。要详细了解如何配置 CloudWatch 与 Amazon Elastic Compute Cloud（Amazon EC2）的集成，请参阅《Amazon CloudWatch 用户指南》中的使用 CloudWatch 代理收集指标、日志和跟踪信息。

Important

统一的 CloudWatch 代理已取代 SSM Agent，作为将日志数据发送到 Amazon CloudWatch Logs 的工具。不支持 SSM Agent aws:cloudWatch 插件。我们建议仅将统一的 CloudWatch 代理用于您的日志收集过程。有关更多信息，请参阅以下主题：

- 将节点日志发送到统一的 CloudWatch Logs（CloudWatch 代理）
- 将 Windows Server 节点日志收集迁移到 CloudWatch 代理
- 《Amazon CloudWatch 用户指南》中的使用 CloudWatch 代理收集指标、日志和跟踪信息。

您可以导出和监视以下数据类型：

ApplicationEventLog

将应用程序事件日志数据发送到 CloudWatch Logs。

CustomLogs

将任何基于文本的日志文件发送到 Amazon CloudWatch Logs。CloudWatch 插件会为日志文件创建指纹。系统随后将数据偏移与每个指纹进行关联。该插件会在出现更改时上传文件，记录偏移，然后将该偏移与指纹关联。此方法用于避免出现这种情况：用户启用插件后，将服务与包含大量文件的目录关联，然后系统会上传所有文件。

> **Warning**
>
> 请注意，如果应用程序在轮询期间截断或尝试清除日志，为 `LogDirectoryPath` 指定的任何日志都可能丢失条目。例如，如果您要限制日志文件大小，请在达到此限制时创建新的日志文件，然后继续将数据写入新文件。

ETW

将 Windows (ETW) 事件跟踪数据发送到 CloudWatch Logs。

IIS

将 IIS 日志数据发送到 CloudWatch Logs

PerformanceCounter

将 Windows 性能计数器发送到 CloudWatch。您可以选择不同类别作为指标上传到 CloudWatch。对于要上传的每个性能计数器，创建具有唯一 ID 的 PerformanceCounter 部分 (例如“PerformanceCounter2”、“PerformanceCounter3”等)，然后配置其属性。

> **Note**
>
> 如果 AWS Systems Manager SSM Agent 或 CloudWatch 插件已停止，则性能计数器数据不再记录到 CloudWatch 中。此行为不同于自定义日志或 Windows 事件日志。自定义日志和 Windows 事件日志会保留性能计数器数据，并在 SSM Agent 后或 CloudWatch 插件可用时将其上传到 CloudWatch。

SecurityEventLog

将安全日志数据发送到 CloudWatch Logs。

SystemEventLog

将系统事件日志数据发送到 CloudWatch Logs。

可为数据定义以下目标：

CloudWatch

发送性能计数器指标数据时所在的目标。可添加具有唯一 ID 的更多部分 (例如，“CloudWatch2”和“CloudWatch3”等)，并为每个新 ID 指定一个不同的区域，将相同数据发送到不同位置。

CloudWatchLogs

发送日志数据时所在的目标。可添加具有唯一 ID 的更多部分 (例如，“CloudWatchLogs2”和“CloudWatchLogs3”等)，并为每个新 ID 指定一个不同的区域，将相同数据发送到不同位置。

语法

```
"runtimeConfig":{
        "aws:cloudWatch":{
            "settings":{
                "startType":"{{ status }}"
            },
            "properties":"{{ properties }}"
        }
    }
```

设置和属性

AccessKey

您的访问密钥 ID。如果未使用 IAM 角色启动实例，则必须指定此属性。此属性不能与 SSM 一起使用。

类型：字符串

必需：否

CategoryName

性能监视器提供的性能计数器类别。

类型：字符串

必需：是

CounterName

性能监视器提供的性能计数器名称。

类型：字符串

必需：是

CultureName

记录时间戳的区域设置。如果 CultureName 为空，则它默认为您的 Windows Server 实例所使用的相同区域位置。

类型：字符串

有效值：有关支持的值的列表，请参阅 Microsoft 网站上的国家语言支持 (NLS)。不支持 div、div-MV、hu 和 hu-HU 值。

必需：否

DimensionName

Amazon CloudWatch 指标的维度。如果您要指定 `DimensionName`，则必须指定 `DimensionValue`。这些参数在列出指标时提供另一个视图。您可以对多个指标使用同一个维度，以便查看属于特定维度的所有指标。

类型：字符串

必需：否

DimensionValue

Amazon CloudWatch 指标的维度值。

类型：字符串

必需：否

编码

要使用的文件编码 (例如 UTF-8)。使用编码名称，而不是显示名称。

类型：字符串

有效值：有关支持的值的列表，请参阅 Microsoft Learn 库中的 Encoding Class。

必需：是

筛选条件

日志名称的前缀。将此参数保留空白以监控所有文件。

类型：字符串

有效值：有关支持的值的列表，请参阅 MSDN 库中的 FileSystemWatcherFilter 属性主题。

必需：否

流

要上传的每个数据类型以及数据的目标（CloudWatch 或 CloudWatch Logs ）。例如，要将在 `"Id": "PerformanceCounter"` 下定义的性能计数器发送到在 `"Id": "CloudWatch"` 下定义的 CloudWatch 目标，请输入 “PerformanceCounter,CloudWatch”。同样，要将自定义日志、ETW 日志和系统日志发送到 `"Id": "ETW"` 下定义的 CloudWatch Logs 目标，请输入 “(ETW),CloudWatchLogs”。此外，还可以将相同性能计数器或日志文件发送到多个目标。例如，要将应用程序日志发送到在 `"Id": "CloudWatchLogs"` 和 `"Id": "CloudWatchLogs2"` 下定义的两个不同目标，请输入 "ApplicationEventLog,(CloudWatchLogs, CloudWatchLogs2)"。

类型：字符串

有效值 (源)：`ApplicationEventLog` | `CustomLogs` | `ETW` | `PerformanceCounter` | `SystemEventLog` | `SecurityEventLog`

有效值 (目标)：`CloudWatch` | `CloudWatchLogs` | `CloudWatch`*n* | `CloudWatchLogs`*n*

必需：是

FullName

组件的完整名称。

类型：字符串

必需：是

Id

标识数据源或目标。此标识符在配置文件中必须是唯一的。

类型：字符串

必需：是

InstanceName

性能计数器实例的名称。请勿使用星号 (*) 标识所有实例，因为每个性能计数器组件仅支持一个指标。不过可以使用 _Total。

类型：字符串

必需：是

级别

发送到 Amazon CloudWatch 的消息类型。

类型：字符串

有效值：

- 1 – 仅上传错误消息。
- 2 – 仅上传警告消息。
- 4 – 仅上传信息消息。

您可以将这些值加在一起以包含多种类型的消息。例如，3 表示将包含错误消息 (1) 和警告消息 (2)。值 7 表示将包含错误消息 (1)、警告消息 (2) 和说明性消息 (4)。

必需：是

> **Note**
>
> 应将 Windows 安全日志的级别设置为 7。

LineCount

标识日志文件的标头中的行数。例如，IIS 日志文件拥有几乎相同的标头。您可以输入 3，系统会读取日志文件标头的前三行以进行识别。在 IIS 日志文件中，第三行是日期和时间戳，各日志文件的日期和时间戳互不相同。

类型：整数

必需：否

LogDirectoryPath

对于 CustomLogs，这是日志在 EC2 实例上的存储路径。对于 IIS 日志，这是为单个站点存储 IIS 日志的文件夹 (例如 C:\\inetpub\\logs\\LogFiles\\W3SVC*n*)。对于 IIS 日志，仅支持 W3C 日志格式。不支持 IIS、NCSA 和自定义格式。

类型：字符串

必需：是

LogGroup

日志组的名称。此名称显示在 CloudWatch 控制台的 Log Groups（日志组）屏幕上。

类型：字符串

必需：是

LogName

日志文件的名称。

1. 要查找日志的名称，请在事件查看器的导航窗格中，选择 Applications and Services Logs (应用程序和服务日志)。
2. 在日志列表中，右键单击要上传的日志 (例如“Microsoft>Windows>备份>可操作”)，然后选择创建自定义视图。
3. 在 Create Custom View (创建自定义视图) 对话框中，选择 XML 选项卡。LogName 位于 <Select Path=> 标签中（例如 `Microsoft-Windows-Backup`）。将此文本复制到 LogName 参数。

类型：字符串

有效值：`Application` | `Security` | `System` | `Microsoft-Windows-WinINet/Analytic`

必需：是

LogStream

目标日志流。如果使用默认值 {instance_id}，则将该实例的实例 ID 用作日志流名称。

类型：字符串

有效值：{instance_id} | {hostname} | {ip_address} *<log_stream_name>*

如果输入的日志流名称不存在，则 CloudWatch Logs 会自动创建该名称。可以使用文字字符串或预定义的变量（{instance_id}、{hostname}、{ip_address}），或所有这三个变量的组合来定义日志流名称。

此参数中指定的日志流名称将在 CloudWatch 控制台中的 Log Groups > Streams for ***<YourLogStream>***（日志组 > <YourLogStream> 的流）屏幕上显示。

必需：是

MetricName

您希望性能数据包含在其下的 CloudWatch 指标。

> **Note**
>
> 不要在名称中使用特殊字符。如果您这样做，指标和相关警报可能无法正常工作。

类型：字符串

必需：是

NameSpace

您希望将写入性能计数器数据的指标命名空间。

类型：字符串

必需：是

PollInterval

必须在多少秒之后才能上传新性能计数器和日志数据。

类型：整数

有效值：将其设置为 5 秒或 5 秒以上。建议设置为十五秒 (00:00:15)。

必需：是

区域

要发送日志数据的 AWS 区域。虽然可以将性能计数器发送到与日志数据发送目标不同的区域，但是我们建议您将此参数设置为运行实例的区域。

类型：字符串

有效值：Systems Manager 和 CloudWatch Logs 都支持的 AWS 区域 区域ID，例如 `us-east-2`、`eu-west-1`, 和 `ap-southeast-1`。对于每个服务支持的 AWS 区域 的列表，请参阅《Amazon Web Services 一般参考》中的 Amazon CloudWatch Logs Service Endpoints 和 Systems Manager service endpoints。

必需：是

SecretKey

您的秘密访问密钥。如果未使用 IAM 角色启动实例，则必须指定此属性。

类型：字符串

必需：否

startType

打开或关闭实例上的 CloudWatch 功能。

类型：字符串

有效值：`Enabled | Disabled`

必需：是

TimestampFormat

要使用的时间戳格式。有关支持的值的列表，请参阅 MSDN 库中的自定义日期和时间格式字符串。

类型：字符串

必需：是

TimeZoneKind

在日志时间戳中不包含时区信息时提供时区信息。如果此参数留空且时间戳不包括时区信息，则 CloudWatch Logs 默认为本地时区。如果时间戳已包含时区信息，则忽略此参数。

类型：字符串

有效值：Local | UTC

必需：否

单位

适当的指标计量单位。

类型：字符串

有效值：秒 | 微秒 | 毫秒 | 字节 | 千字节 | 兆字节 | 千兆字节 | 太兆字节 | 位 | 千位 | 兆位 | 千兆位 | 太兆位 | 百分比 | 计数 | 字节/秒 | 千字节/秒 | 兆字节/秒 | 千兆字节/秒 | 太兆字节/秒 | 位/秒 | 千位/秒 | 兆位/秒 | 千兆位/秒 | 太兆位/秒 | 计数/秒 | 无

必需：是

aws:configureDocker

(2.0 版或更高版本架构) 将实例配置为使用容器和 Docker。此插件在大多数 Linux 变体和 Windows Server 操作系统中受支持。

语法

Schema 2.2

YAML

```
---
schemaVersion: '2.2'
description: aws:configureDocker
parameters:
  action:
    description: "(Required) The type of action to perform."
    type: String
    default: Install
    allowedValues:
    - Install
    - Uninstall
mainSteps:
- action: aws:configureDocker
```

```
  name: configureDocker
  inputs:
    action: "{{ action }}"
```

JSON

```
{
  "schemaVersion": "2.2",
  "description": "aws:configureDocker plugin",
  "parameters": {
    "action": {
      "description": "(Required) The type of action to perform.",
      "type": "String",
      "default": "Install",
      "allowedValues": [
        "Install",
        "Uninstall"
      ]
    }
  },
  "mainSteps": [
    {
      "action": "aws:configureDocker",
      "name": "configureDocker",
      "inputs": {
        "action": "{{ action }}"
      }
    }
  ]
}
```

输入

action

要执行的操作类型。

类型：Enum

有效值：`Install` | `Uninstall`

必需：是

aws:configurePackage

（2.0 版或更高版本架构）安装或卸载 AWS Systems Manager Distributor 程序包。您可以安装所指定软件包的最新版本、默认版本或其中一个版本。AWS 提供的软件包也受支持。此插件可在 Windows Server 和 Linux 操作系统上运行，但 Linux 操作系统不支持所有可用的软件包。

Windows Server 的可用 AWS 程序包包括：AWSPVDriver、AWSNVMe、AwsEnaNetworkDriver、AwsVssComponents、AmazonCloudWatchA和 AWSSupport-EC2Rescue.

适用于 Linux 操作系统的可用 AWS 软件包包括：AmazonCloudWatchAgent、CodeDeployAgent,和 AWSSupport-EC2Rescue。

语法

Schema 2.2

YAML

```
---
schemaVersion: '2.2'
description: aws:configurePackage
parameters:
  name:
    description: "(Required) The name of the AWS package to install or uninstall."
    type: String
  action:
    description: "(Required) The type of action to perform."
    type: String
    default: Install
    allowedValues:
    - Install
    - Uninstall
  ssmParameter:
    description: "(Required) Argument stored in Parameter Store."
    type: String
    default: "{{ ssm:parameter_store_arg }}"
mainSteps:
- action: aws:configurePackage
  name: configurePackage
  inputs:
    name: "{{ name }}"
    action: "{{ action }}"
```

```
    additionalArguments:
      "{\"SSM_parameter_store_arg\": \"{{ ssmParameter }}\", \"SSM_custom_arg\":
 \"myValue\"}"
```

JSON

```
{
   "schemaVersion": "2.2",
   "description": "aws:configurePackage",
   "parameters": {
      "name": {
         "description": "(Required) The name of the AWS package to install or
 uninstall.",
         "type": "String"
      },
      "action": {
         "description": "(Required) The type of action to perform.",
         "type": "String",
         "default": "Install",
         "allowedValues": [
            "Install",
            "Uninstall"
         ]
      },
      "ssmParameter": {
         "description": "(Required) Argument stored in Parameter Store.",
         "type": "String",
         "default": "{{ ssm:parameter_store_arg }}"
      }
   },
   "mainSteps": [
      {
         "action": "aws:configurePackage",
         "name": "configurePackage",
         "inputs": {
            "name": "{{ name }}",
            "action": "{{ action }}",
            "additionalArguments": "{\"SSM_parameter_store_arg\":
 \"{{ ssmParameter }}\", \"SSM_custom_arg\": \"myValue\"}"
         }
      }
   ]
}
```

输入

名称

要安装或卸载的 AWS 软件包名称。可用的软件包如下所述：AWSPVDriver、AwsEnaNetworkDriver、AwsVssComponents，和 AmazonCloudWatchAgent。

类型：字符串

必需：是

action

安装或卸载软件包。

类型：Enum

有效值：Install | Uninstall

必需：是

安装类型

要执行的安装类型。如果指定 Uninstall and reinstall，该软件包将完全卸载，然后重新安装。在重新安装完成之前，应用程序不可用。如果指定 In-place update，将根据您在更新脚本中提供的指令，仅将新文件或更改的文件添加到现有的安装中。应用程序在整个更新过程中保持可用。这些区域有:已发布的软件包 AWS 不支持 In-place update 选项。Uninstall and reinstall 为默认值。

类型：Enum

有效值：Uninstall and reinstall | In-place update

必需：否

附加参数

为安装、卸载或更新脚本提供的其他参数的 JSON 字符串。每个参数的前缀必须为 SSM_。可以使用约定 {{ssm:*parameter-name*}} 在附加参数中引用 Parameter Store 参数。要在安装、卸载或更新脚本过程中使用附加参数，必须使用适合操作系统的语法将参数作为环境变量引用。例如，在 PowerShell 中，您引用 SSM_arg 参数为 $Env:SSM_arg。您定义的参数数量没有限制，但附加参数输入有 4096 个字符限制。此限制包括您定义的所有秘钥和值。

类型：StringMap

必需：否

version

要卸载或安装的特定版本的软件包。如果要安装，系统默认安装最新发布的版本。如果要卸载，系统默认卸载当前安装的版本。如果没有找到已安装的版本，则会下载最新发布的版本，然后运行卸载操作。

类型：字符串

必需：否

aws:domainJoin

将 EC2 实例加入域。此插件在 Linux 和 Windows Server 操作系统上运行。该插件会将 Linux 实例的主机名更改为 EC2AMAZ-*XXXXXXX* 格式。有关加入 EC2 实例的更多信息，请参阅 AWS Directory Service 管理指南中的将 EC2 实例加入到 AWS 托管 Microsoft AD 目录。

语法

Schema 2.2

YAML

```
---
schemaVersion: '2.2'
description: aws:domainJoin
parameters:
  directoryId:
    description: "(Required) The ID of the directory."
    type: String
  directoryName:
    description: "(Required) The name of the domain."
    type: String
  directoryOU:
    description: "(Optional) The organizational unit to assign the computer object
 to."
    type: String
  dnsIpAddresses:
    description: "(Required) The IP addresses of the DNS servers for your
 directory."
```

```
    type: StringList
  hostname:
    description: "(Optional) The hostname you want to assign to the node."
    type: String
mainSteps:
- action: aws:domainJoin
  name: domainJoin
  inputs:
    directoryId: "{{ directoryId }}"
    directoryName: "{{ directoryName }}"
    directoryOU: "{{ directoryOU }}"
    dnsIpAddresses: "{{ dnsIpAddresses }}"
    hostname: "{{ hostname }}"
```

JSON

```
{
  "schemaVersion": "2.2",
  "description": "aws:domainJoin",
  "parameters": {
    "directoryId": {
      "description": "(Required) The ID of the directory.",
      "type": "String"
    },
    "directoryName": {
      "description": "(Required) The name of the domain.",
      "type": "String"
    },
    "directoryOU": {
        "description": "(Optional) The organizational unit to assign the computer
 object to.",
        "type": "String"
      },
    "dnsIpAddresses": {
      "description": "(Required) The IP addresses of the DNS servers for your
 directory.",
      "type": "StringList"
    },
    "hostname": {
        "description": "(Optional) The hostname you want to assign to the node.",
        "type": "String"
      }
  },
```

```
  "mainSteps": [
    {
      "action": "aws:domainJoin",
      "name": "domainJoin",
      "inputs": {
        "directoryId": "{{ directoryId }}",
        "directoryName": "{{ directoryName }}",
        "directoryOU":"{{ directoryOU }}",
        "dnsIpAddresses":"{{ dnsIpAddresses }}",
        "hostname":"{{ hostname }}"
      }
    }
  ]
}
```

Schema 1.2

YAML

```
---
runtimeConfig:
  aws:domainJoin:
    properties:
      directoryId: "{{ directoryId }}"
      directoryName: "{{ directoryName }}"
      directoryOU: "{{ directoryOU }}"
      dnsIpAddresses: "{{ dnsIpAddresses }}"
```

JSON

```
{
   "runtimeConfig":{
      "aws:domainJoin":{
         "properties":{
            "directoryId":"{{ directoryId }}",
            "directoryName":"{{ directoryName }}",
            "directoryOU":"{{ directoryOU }}",
            "dnsIpAddresses":"{{ dnsIpAddresses }}"
         }
      }
   }
}
```

属性

directoryId

目录的 ID。

类型：字符串

必需：是

示例："directoryId": "d-1234567890"

directoryName

域的名称。

类型：字符串

必需：是

示例："directoryName": "example.com"

directoryOU

组织部门 (OU)。

类型：字符串

必需：否

示例："directoryOU": "OU=test,DC=example,DC=com"

dnsIpAddresses

DNS 服务器的 IP 地址。

类型：StringList

必需：是

示例："dnsIpAddresses": ["198.51.100.1","198.51.100.2"]

hostname

要分配给节点的主机名。如果未提供，则 Windows 实例的名称不会更改，而 Linux 实例将使用默认的命名模式。如果提供，Windows 实例将使用所提供的确切值，而 Linux 实例将以此为前缀（除非将 `keepHostName` 设置为“true”）。

类型：字符串

必需：否

keepHostName

确定 Linux 实例在加入域时是否更改主机名。此参数仅适用于 Linux。默认情况下（hostname、hostnameNumAppendDigits 无输入，且 keepHostName 设置为“false”），Linux 主机将重命名为 EC2AMAZ-XXXXXX 模式。设置为“true”时，将会保留原始主机名并忽略 hostname 和 hostnameNumAppendDigits 的输入。

类型：布尔值

必需：否

hostnameNumAppendDigits

定义要在主机名值之后附加的随机数值位数。此参数仅适用于 Linux，并且应与 hostname 参数结合使用。如果未提供 hostname，则会忽略此参数。

类型：字符串

允许的值：1 至 5

必需：否

示例

有关示例，请参阅《AWS Directory Service 管理指南》中的将 Amazon EC2 实例加入您的 AWS Managed Microsoft AD。

aws:downloadContent

（2.0 版或更高版本架构）从远程位置下载 SSM 文档和脚本。不支持 GitHub Enterprise 存储库。此插件在 Linux 和 Windows Server 操作系统中受支持。

语法

Schema 2.2

YAML

```
---
schemaVersion: '2.2'
```

```
description: aws:downloadContent
parameters:
  sourceType:
    description: "(Required) The download source."
    type: String
  sourceInfo:
    description: "(Required) The information required to retrieve the content from
      the required source."
    type: StringMap
mainSteps:
- action: aws:downloadContent
  name: downloadContent
  inputs:
    sourceType: "{{ sourceType }}"
    sourceInfo: "{{ sourceInfo }}"
```

JSON

```
{
  "schemaVersion": "2.2",
  "description": "aws:downloadContent",
  "parameters": {
    "sourceType": {
    "description": "(Required) The download source.",
    "type": "String"
  },
  "sourceInfo": {
    "description": "(Required) The information required to retrieve the content from
 the required source.",
    "type": "StringMap"
    }
  },
  "mainSteps": [
    {
      "action": "aws:downloadContent",
      "name": "downloadContent",
      "inputs": {
        "sourceType":"{{ sourceType }}",
        "sourceInfo":"{{ sourceInfo }}"
      }
    }
  ]
}
```

输入

sourceType

下载源 Systems Manager 支持以下下载脚本和 SSM 文档的源类型：`GitHub`、`Git`、`HTTP`、`S3`，和`SSMDocument`。

类型：字符串

必需：是

sourceInfo

从所需源检索内容所需的信息。

类型：StringMap

必需：是

对于 sourceType **`GitHub`**，请指定以下内容：

- owner：存储库所有者。
- repository：存储库的名称。
- path：您要下载的文件或目录所在的路径。
- getOptions：用于从分支（而非主分支）或存储库中的特定提交中检索内容的额外选项。如果您在主分支中使用最新提交，则可以省略 getOptions。如果您的存储库是在 2020 年 10 月 1 日之后创建的，则默认分支可能被命名为 main 而不是 master。在这种情况下，需要为 getOptions 参数指定值。

  此参数采用以下格式：

  - branch:refs/heads/*`branch_name`*

    默认值为 `master`。

    要指定非默认分支，请使用以下格式：

    branch:refs/heads/*`branch_name`*
  - commitID：*`commitID`*

    默认值为 `head`。

    要在最新提交以外的提交中使用 SSM 文档的版本，请指定完整的提交 ID。例如：

```
"getOptions": "commitID:bbc1ddb94...b76d3bEXAMPLE",
```

- tokenInfo：以 {{ssm-secure:*secure-string-token-name*}} 格式存储 GitHub 访问令牌信息的 Systems Manager 参数（SecureString 参数）。

> **Note**
>
> 该 tokenInfo 字段是唯一支持 SecureString 参数的 SSM 文档插件字段。任何其他字段和任何其他 SSM 文档插件都不支持 SecureString 参数。

```
{
    "owner":"TestUser",
    "repository":"GitHubTest",
    "path":"scripts/python/test-script",
    "getOptions":"branch:master",
    "tokenInfo":"{{ssm-secure:secure-string-token}}"
}
```

对于 sourceType **Git**，必须指定以下内容：

- 存储库

  path：要下载的文件或目录的 Git repository URL。

  类型：字符串

此外，您可以指定以下任何可选参数：

- getOptions

  用于从分支（而非主分支）或存储库中的特定提交中检索内容的额外选项。如果您在主分支中使用最新提交，则可以省略 getOptions。

  类型：字符串

  此参数采用以下格式：

  - branch:refs/heads/*branch_name*

    默认值为 master。

    仅当您的 SSM 文档存储在 master 以外的分支中时，"branch" 才是必需的。例如：

```
"getOptions": "branch:refs/heads/main"
```

- commitID：*commitID*

  默认值为 head。

  要在最新提交以外的提交中使用 SSM 文档的版本，请指定完整的提交 ID。例如：

```
"getOptions": "commitID:bbc1ddb94...b76d3bEXAMPLE",
```

- privateSSHKey

  连接指定的repository 时要使用的 SSH 秘钥。可以使用以下格式为引用 SSH 密钥值 SecureString 参数：{{ssm-secure:*your-secure-string-parameter*}}。

  类型：字符串

- 跳过主机键检查

  确定连接到您指定的 repository 时 StrictHostKeyChecking 选项的值。默认值为 false。

  类型：布尔值

- username

  连接到使用 HTTP 指定的 repository 时使用的用户名。可以使用以下格式引用用户名值的 SecureString 参数：{{ssm-secure:*your-secure-string-parameter*}}。

  类型：字符串

- password

  连接到使用 HTTP 指定的 repository 时使用的密码。可以使用以下格式为引用密码值 SecureString 参数：{{ssm-secure:*your-secure-string-parameter*}}。

  类型：字符串

对于 sourceType**HTTP**，必须指定以下内容：

- url

  要下载的文件或目录的 URL。

  类型：字符串

此外，您可以指定以下任何可选参数：

- allowInsecureDownload

  确定是否可以通过未使用安全套接字层 (SSL) 或传输层安全 (TLS) 加密的连接执行下载。默认值为 `false`。我们建议您不要在未加密的情况下执行下载。如果您选择这样做，您将承担所有相关风险。安全性是 AWS 和您的共同责任。这被描述为责任共担共担模式。要了解更多信息，请参阅责任共担模式。

  类型：布尔值

- authMethod

  确定在连接到指定的 `url` 时是否要使用用户名和密码。如果指定 `Basic` 或者 `Digest` 时，您必须为 `username` 和 `password` 参数提供值。使用 `Digest` 方法，3.0.1181.0 版本或更高版本的 SSM Agent 必须安装在实例上。`Digest` 方法支持 MD5 和 SHA256 加密。

  类型：字符串

  有效值：None |Basic |Digest

- username

  连接到 `url` 您指定使用 `Basic` 身份验证。可以使用以下格式引用用户名值的 `SecureString` 参数：{{ssm-secure:*your-secure-string-parameter*}}。

  类型：字符串

- password

  连接到您使用 `Basic` 身份验证指定的 `url` 时使用的密码。可以使用以下格式为引用密码值 `SecureString` 参数：{{ssm-secure:*your-secure-string-parameter*}}。

  类型：字符串

对于 sourceType **`S3`**，请指定以下内容：

- 路径：要从 Amazon S3 下载的文件或目录的 URL。

```
{
    "path": "https://s3.amazonaws.com/amzn-s3-demo-bucket/powershell/
helloPowershell.ps1"
}
```

对于 SourceType **`SSMDocument`**，请指定以下其中一 项：

- 名称：采用以下格式的文档的名称和版本：name:version。版本为可选项。

```
{
    "name": "Example-RunPowerShellScript:3"
}
```

- name：采用以下格式的文档
  ARN：arn:aws:ssm:*region*:*account_id*:document/*document_name*

```
{
   "name":"arn:aws:ssm:us-east-2:3344556677:document/MySharedDoc"
}
```

destinationPath

实例上的可选本地路径，用于下载文件。如果不指定路径，内容将下载到命令 ID 的相对路径。

类型：字符串

必需：否

aws:psModule

在 Amazon EC2 实例上安装 PowerShell 模块。此插件仅在 Windows Server 操作系统中运行。

语法

Schema 2.2

YAML

```
---
schemaVersion: '2.2'
description: aws:psModule
parameters:
  source:
    description: "(Required) The URL or local path on the instance to the
 application
     .zip file."
    type: String
mainSteps:
- action: aws:psModule
```

```
  name: psModule
  inputs:
    source: "{{ source }}"
```

JSON

```
{
  "schemaVersion": "2.2",
  "description": "aws:psModule",
  "parameters": {
    "source": {
      "description": "(Required) The URL or local path on the instance to the
 application .zip file.",
      "type": "String"
    }
  },
  "mainSteps": [
    {
      "action": "aws:psModule",
      "name": "psModule",
      "inputs": {
        "source": "{{ source }}"
      }
    }
  ]
}
```

Schema 1.2

YAML

```
---
runtimeConfig:
  aws:psModule:
    properties:
    - runCommand: "{{ commands }}"
      source: "{{ source }}"
      sourceHash: "{{ sourceHash }}"
      workingDirectory: "{{ workingDirectory }}"
      timeoutSeconds: "{{ executionTimeout }}"
```

JSON

```
{
   "runtimeConfig":{
      "aws:psModule":{
         "properties":[
            {
               "runCommand":"{{ commands }}",
               "source":"{{ source }}",
               "sourceHash":"{{ sourceHash }}",
               "workingDirectory":"{{ workingDirectory }}",
               "timeoutSeconds":"{{ executionTimeout }}"
            }
         ]
      }
   }
}
```

属性

runCommand

安装模块后要运行的 PowerShell 命令。

类型：StringList

必需：否

源

访问实例上应用程序 `.zip` 文件的 URL 或本地路径。

类型：字符串

必需：是

sourceHash

`.zip` 文件的 SHA256 哈希值。

类型：字符串

必需：否

timeoutSeconds

在被视为已失败前命令将运行的时间 (以秒为单位)。

类型：字符串

必需：否

workingDirectory

实例上工作目录的路径。

类型：字符串

必需：否

aws:refreshAssociation

(2.0 版或更高版本架构) 按需刷新 (强制应用) 关联。此操作将根据所选关联或绑定到目标的所有关联中定义的内容来更改系统状态。此插件仅在 Linux 和 Windows Server 操作系统上运行。

语法

Schema 2.2

YAML

```
---
schemaVersion: '2.2'
description: aws:refreshAssociation
parameters:
  associationIds:
    description: "(Optional) List of association IDs. If empty, all associations
 bound
      to the specified target are applied."
    type: StringList
mainSteps:
- action: aws:refreshAssociation
  name: refreshAssociation
  inputs:
    associationIds:
    - "{{ associationIds }}"
```

JSON

```
{
  "schemaVersion": "2.2",
  "description": "aws:refreshAssociation",
  "parameters": {
    "associationIds": {
      "description": "(Optional) List of association IDs. If empty, all associations
 bound to the specified target are applied.",
      "type": "StringList"
    }
  },
  "mainSteps": [
    {
      "action": "aws:refreshAssociation",
      "name": "refreshAssociation",
      "inputs": {
        "associationIds": [
          "{{ associationIds }}"
        ]
      }
    }
  ]
}
```

输入

associationIds

关联 ID 的列表。如果为空，则应用与指定目标绑定的所有关联。

类型：StringList

必需：否

aws:runDockerAction

（2.0 版或更高版本架构）在容器上运行 Docker 操作。此插件仅在 Linux 和 Windows Server 操作系统上运行。

语法

Schema 2.2

YAML

```
---
mainSteps:
- action: aws:runDockerAction
  name: RunDockerAction
  inputs:
    action: "{{ action }}"
    container: "{{ container }}"
    image: "{{ image }}"
    memory: "{{ memory }}"
    cpuShares: "{{ cpuShares }}"
    volume: "{{ volume }}"
    cmd: "{{ cmd }}"
    env: "{{ env }}"
    user: "{{ user }}"
    publish: "{{ publish }}"
    workingDirectory: "{{ workingDirectory }}"
    timeoutSeconds: "{{ timeoutSeconds }}"
```

JSON

```
{
   "mainSteps":[
      {
         "action":"aws:runDockerAction",
         "name":"RunDockerAction",
         "inputs":{
            "action":"{{ action }}",
            "container":"{{ container }}",
            "image":"{{ image }}",
            "memory":"{{ memory }}",
            "cpuShares":"{{ cpuShares }}",
            "volume":"{{ volume }}",
            "cmd":"{{ cmd }}",
            "env":"{{ env }}",
            "user":"{{ user }}",
            "publish":"{{ publish }}",
            "workingDirectory": "{{ workingDirectory }}",
```

```
            "timeoutSeconds": "{{ timeoutSeconds }}"
         }
      }
   ]
}
```

输入

action

要执行的操作类型。

类型：字符串

必需：是

容器

Docker 容器 ID。

类型：字符串

必需：否

image

Docker 映像名称。

类型：字符串

必需：否

cmd

容器命令。

类型：字符串

必需：否

memory

容器内存限制。

类型：字符串

必需：否

cpuShares

容器 CPU 份额 (相对权重)。

类型：字符串

必需：否

volume

容器卷挂载。

类型：StringList

必需：否

env

容器的环境变量。

类型：字符串

必需：否

用户

容器的用户名。

类型：字符串

必需：否

发布

容器已发布端口。

类型：字符串

必需：否

workingDirectory

您的托管节点上的工作目录的路径。

类型：字符串

必需：否

timeoutSeconds

在被视为已失败前命令将运行的时间 (以秒为单位)。

类型：字符串

必需：否

aws:runDocument

（2.0 版或更高版本架构）运行存储在 Systems Manager 或本地共享存储中的 SSM 文档。您可以将此插件与 aws:downloadContent 插件配合使用，以将远程位置的 SSM 文档下载到本地共享存储，然后运行该文档。此插件在 Linux 和 Windows Server 操作系统中受支持。此插件不支持运行 AWS-UpdateSSMAgent 文档或使用 aws:updateSsmAgent 插件的任何文档。

语法

Schema 2.2

YAML

```
---
schemaVersion: '2.2'
description: aws:runDocument
parameters:
  documentType:
    description: "(Required) The document type to run."
    type: String
    allowedValues:
    - LocalPath
    - SSMDocument
mainSteps:
- action: aws:runDocument
  name: runDocument
  inputs:
    documentType: "{{ documentType }}"
```

JSON

```
{
  "schemaVersion": "2.2",
  "description": "aws:runDocument",
```

```
  "parameters": {
    "documentType": {
      "description": "(Required) The document type to run.",
      "type": "String",
      "allowedValues": [
        "LocalPath",
        "SSMDocument"
      ]
    }
  },
  "mainSteps": [
    {
      "action": "aws:runDocument",
      "name": "runDocument",
      "inputs": {
        "documentType": "{{ documentType }}"
      }
    }
  ]
}
```

输入

documentType

要运行的文档类型。您可以运行本地文档 (LocalPath) 或存储在 Systems Manager (SSMDocument) 中的文档。

类型：字符串

必需：是

documentPath

文档的路径。如果 documentType 是 LocalPath，则指定本地共享存储上文档的路径。如果 documentType 是 SSMDocument，则指定文档的名称。

类型：字符串

必需：否

documentParameters

文档的参数。

类型：StringMap

必需：否

aws:runPowerShellScript

运行 PowerShell 脚本或者指定要运行的脚本的路径。此插件在 Microsoft Windows Server 和 Linux 操作系统上运行。

语法

Schema 2.2

YAML

```
---
schemaVersion: '2.2'
description: aws:runPowerShellScript
parameters:
  commands:
    type: String
    description: "(Required) The commands to run or the path to an existing script
      on the instance."
    default: Write-Host "Hello World"
mainSteps:
- action: aws:runPowerShellScript
  name: runPowerShellScript
  inputs:
    timeoutSeconds: '60'
    runCommand:
    - "{{ commands }}"
```

JSON

```
{
  "schemaVersion": "2.2",
  "description": "aws:runPowerShellScript",
  "parameters": {
    "commands": {
      "type": "String",
      "description": "(Required) The commands to run or the path to an existing
 script on the instance.",
```

```
      "default": "Write-Host \"Hello World\""
    }
  },
  "mainSteps": [
    {
      "action": "aws:runPowerShellScript",
      "name": "runPowerShellScript",
      "inputs": {
        "timeoutSeconds": "60",
        "runCommand": [
          "{{ commands }}"
        ]
      }
    }
  ]
}
```

Schema 1.2

YAML

```
---
runtimeConfig:
  aws:runPowerShellScript:
    properties:
    - id: 0.aws:runPowerShellScript
      runCommand: "{{ commands }}"
      workingDirectory: "{{ workingDirectory }}"
      timeoutSeconds: "{{ executionTimeout }}"
```

JSON

```
{
   "runtimeConfig":{
      "aws:runPowerShellScript":{
         "properties":[
            {
               "id":"0.aws:runPowerShellScript",
               "runCommand":"{{ commands }}",
               "workingDirectory":"{{ workingDirectory }}",
               "timeoutSeconds":"{{ executionTimeout }}"
            }
```

```
            ]
        }
    }
}
```

属性

runCommand

指定要运行的命令或实例上现有脚本的路径。

类型：StringList

必需：是

timeoutSeconds

在被视为已失败前命令将运行的时间（以秒为单位）。如果达到超时，Systems Manager 停止命令执行。

类型：字符串

必需：否

workingDirectory

实例上工作目录的路径。

类型：字符串

必需：否

aws:runShellScript

运行 Linux shell 脚本或者指定要运行的脚本的路径。此插件仅在 Linux 操作系统中运行。

语法

Schema 2.2

YAML

```
---
```

```
schemaVersion: '2.2'
description: aws:runShellScript
parameters:
  commands:
    type: String
    description: "(Required) The commands to run or the path to an existing script
      on the instance."
    default: echo Hello World
mainSteps:
- action: aws:runShellScript
  name: runShellScript
  inputs:
    timeoutSeconds: '60'
    runCommand:
    - "{{ commands }}"
```

JSON

```
{
  "schemaVersion": "2.2",
  "description": "aws:runShellScript",
  "parameters": {
    "commands": {
      "type": "String",
      "description": "(Required) The commands to run or the path to an existing
 script on the instance.",
      "default": "echo Hello World"
    }
  },
  "mainSteps": [
    {
      "action": "aws:runShellScript",
      "name": "runShellScript",
      "inputs": {
        "timeoutSeconds": "60",
        "runCommand": [
          "{{ commands }}"
        ]
      }
    }
  ]
}
```

Schema 1.2

YAML

```
---
runtimeConfig:
  aws:runShellScript:
    properties:
    - runCommand: "{{ commands }}"
      workingDirectory: "{{ workingDirectory }}"
      timeoutSeconds: "{{ executionTimeout }}"
```

JSON

```
{
   "runtimeConfig":{
      "aws:runShellScript":{
         "properties":[
            {
               "runCommand":"{{ commands }}",
               "workingDirectory":"{{ workingDirectory }}",
               "timeoutSeconds":"{{ executionTimeout }}"
            }
         ]
      }
   }
}
```

属性

runCommand

指定要运行的命令或实例上现有脚本的路径。

类型：StringList

必需：是

timeoutSeconds

在被视为已失败前命令将运行的时间（以秒为单位）。如果达到超时，Systems Manager 停止命令执行。

类型：字符串

必需：否

workingDirectory

实例上工作目录的路径。

类型：字符串

必需：否

aws:softwareInventory

（2.0 版或更高版本架构）收集有关托管实例上应用程序、文件和配置的元数据。此插件仅在 Linux 和 Windows Server 操作系统上运行。在配置清单收集时，应该首先创建 AWS Systems Manager State Manager 关联。Systems Manager 会在关联运行时收集清单数据。如果您不先创建关联，并试图调用 aws:softwareInventory 插件，则系统会返回以下错误：

```
The aws:softwareInventory plugin can only be invoked via ssm-associate.
```

一个实例一次只能配置一个清单关联。如果您为一个实例配置了两个或更多关联，则清单关联不会运行，而且系统不会收集清单数据。有关收集文件清单的更多信息，请参阅 AWS Systems Manager 清单。

语法

Schema 2.2

YAML

```
---
mainSteps:
- action: aws:softwareInventory
  name: collectSoftwareInventoryItems
  inputs:
    applications: "{{ applications }}"
    awsComponents: "{{ awsComponents }}"
    networkConfig: "{{ networkConfig }}"
    files: "{{ files }}"
    services: "{{ services }}"
    windowsRoles: "{{ windowsRoles }}"
```

```
    windowsRegistry: "{{ windowsRegistry}}"
    windowsUpdates: "{{ windowsUpdates }}"
    instanceDetailedInformation: "{{ instanceDetailedInformation }}"
    customInventory: "{{ customInventory }}"
```

JSON

```
{
   "mainSteps":[
      {
         "action":"aws:softwareInventory",
         "name":"collectSoftwareInventoryItems",
         "inputs":{
            "applications":"{{ applications }}",
            "awsComponents":"{{ awsComponents }}",
            "networkConfig":"{{ networkConfig }}",
            "files":"{{ files }}",
            "services":"{{ services }}",
            "windowsRoles":"{{ windowsRoles }}",
            "windowsRegistry":"{{ windowsRegistry}}",
            "windowsUpdates":"{{ windowsUpdates }}",
            "instanceDetailedInformation":"{{ instanceDetailedInformation }}",
            "customInventory":"{{ customInventory }}"
         }
      }
   ]
}
```

输入

应用程序

（可选）收集已安装应用程序的元数据。

类型：字符串

必需：否

awsComponents

（可选）收集AWS 组件（例如 amazon-ssm-agent）的元数据。

类型：字符串

必需：否

文件

（可选，需要 2.2.64.0 版本或更高版本的 SSM Agent ）收集文件的元数据，包括文件名称、文件创建时间、文件上次修改和访问时间以及文件大小等等。有关收集文件清单的更多信息，请参阅 使用文件和 Windows 注册表清单。

类型：字符串

必需：否

networkConfig

（可选）收集网络配置的元数据。

类型：字符串

必需：否

billingInfo

（可选）收集与 AMI 账单代码关联的平台详细信息的元数据。

类型：字符串

必需：否

windowsUpdates

（可选）收集所有 Windows 更新的元数据。

类型：字符串

必需：否

instanceDetailedInformation

（可选）收集多于默认清单插件（`aws:instanceInformation`）提供的信息的实例信息，包括 CPU 型号、速度和核心数量，等等。

类型：字符串

必需：否

服务

（可选，仅限 Windows 操作系统，需要 2.2.64.0 版本或更高版本的SSM Agent ）收集服务配置的元数据。

类型：字符串

必需：否

windowsRegistry

（可选，仅限 Windows 操作系统，需要 2.2.64.0 版本或更高版本的SSM Agent）收集 Windows 注册表项和值。您可以选择一个键路径并以递归方式收集所有键和值。您还可以收集特定路径的特定注册表项及其值。清单会收集键路径、名称、类型和值。有关收集 Windows 注册表清单的更多信息，请参阅 使用文件和 Windows 注册表清单。

类型：字符串

必需：否

windowsRoles

（可选，仅限 Windows 操作系统，需要 2.2.64.0 版本或更高版本的 SSM Agent ）收集 Microsoft Windows 角色配置的元数据。

类型：字符串

必需：否

customInventory

（可选）收集自定义清单数据。有关自定义清单的更多信息，请参阅 使用自定义清单

类型：字符串

必需：否

customInventoryDirectory

（可选）从指定目录收集自定义清单数据。有关自定义清单的更多信息，请参阅 使用自定义清单

类型：字符串

必需：否

aws:updateAgent

此命令可将 EC2Config 服务更新为最新版本或指定较旧版本。此插件仅在 Microsoft Windows Server 操作系统上运行。有关 EC2Config 服务的更多信息，请参阅《Amazon EC2 用户指南》中的使用 EC2Config 服务配置 Windows 实例（旧版）。

语法

Schema 2.2

YAML

```
---
schemaVersion: '2.2'
description: aws:updateAgent
mainSteps:
- action: aws:updateAgent
  name: updateAgent
  inputs:
    agentName: Ec2Config
    source: https://s3.{Region}.amazonaws.com/aws-ssm-{Region}/manifest.json
```

JSON

```
{
  "schemaVersion": "2.2",
  "description": "aws:updateAgent",
  "mainSteps": [
    {
      "action": "aws:updateAgent",
      "name": "updateAgent",
      "inputs": {
        "agentName": "Ec2Config",
        "source": "https://s3.{Region}.amazonaws.com/aws-ssm-{Region}/manifest.json"
      }
    }
  ]
}
```

Schema 1.2

YAML

```
---
runtimeConfig:
  aws:updateAgent:
    properties:
      agentName: Ec2Config
```

```
      source: https://s3.{Region}.amazonaws.com/aws-ssm-{Region}/manifest.json
      allowDowngrade: "{{ allowDowngrade }}"
      targetVersion: "{{ version }}"
```

JSON

```
{
   "runtimeConfig":{
      "aws:updateAgent":{
         "properties":{
            "agentName":"Ec2Config",
            "source":"https://s3.{Region}.amazonaws.com/aws-ssm-{Region}/
manifest.json",
            "allowDowngrade":"{{ allowDowngrade }}",
            "targetVersion":"{{ version }}"
         }
      }
   }
}
```

属性

agentName

EC2Config。这是运行 EC2Config 服务的代理的名称。

类型：字符串

必需：是

allowDowngrade

允许将 EC2Config 服务降级为早期版本。如果设置为 False，则只能将该服务升级为更新的版本(默认)。如果设置为 True，则指定早期版本。

类型：布尔值

必需：否

源

Systems Manager 复制要安装的 EC2Config 版本的位置。您无法更改此位置。

类型：字符串

必需：是

targetVersion

要安装的特定版本的 EC2Config 服务。如果未指定，服务将更新到最新版本。

类型：字符串

必需：否

aws:updateSsmAgent

将 SSM Agent 更新到最新版本或指定较旧版本。此插件在 Linux 和 Windows Server 操作系统上运行。有关更多信息，请参阅 使用 SSM Agent。

语法

Schema 2.2

YAML

```
---
schemaVersion: '2.2'
description: aws:updateSsmAgent
parameters:
  allowDowngrade:
    default: 'false'
    description: "(Optional) Allow the Amazon SSM Agent service to be downgraded to
      an earlier version. If set to false, the service can be upgraded to newer
 versions
      only (default). If set to true, specify the earlier version."
    type: String
    allowedValues:
    - 'true'
    - 'false'
mainSteps:
- action: aws:updateSsmAgent
  name: updateSSMAgent
  inputs:
    agentName: amazon-ssm-agent
    source: https://s3.{Region}.amazonaws.com/amazon-ssm-{Region}/ssm-agent-
manifest.json
```

```
    allowDowngrade: "{{ allowDowngrade }}"
```

JSON

```
{
  "schemaVersion": "2.2",
  "description": "aws:updateSsmAgent",
  "parameters": {
    "allowDowngrade": {
      "default": "false",
      "description": "(Required) Allow the Amazon SSM Agent service to be downgraded
 to an earlier version. If set to false, the service can be upgraded to newer
 versions only (default). If set to true, specify the earlier version.",
      "type": "String",
      "allowedValues": [
        "true",
        "false"
      ]
    }
  },
  "mainSteps": [
    {
      "action": "aws:updateSsmAgent",
      "name": "awsupdateSsmAgent",
      "inputs": {
        "agentName": "amazon-ssm-agent",
        "source": "https://s3.{Region}.amazonaws.com/amazon-ssm-{Region}/ssm-agent-
manifest.json",
        "allowDowngrade": "{{ allowDowngrade }}"
      }
    }
  ]
}
```

Schema 1.2

YAML

```
---
runtimeConfig:
  aws:updateSsmAgent:
    properties:
```

```
    - agentName: amazon-ssm-agent
      source: https://s3.{Region}.amazonaws.com/aws-ssm-{Region}/manifest.json
      allowDowngrade: "{{ allowDowngrade }}"
```

JSON

```
{
   "runtimeConfig":{
      "aws:updateSsmAgent":{
         "properties":[
            {
               "agentName":"amazon-ssm-agent",
               "source":"https://s3.{Region}.amazonaws.com/aws-ssm-{Region}/
manifest.json",
               "allowDowngrade":"{{ allowDowngrade }}"
            }
         ]
      }
   }
}
```

属性

agentName

amazon-ssm-agent。这是在实例上处理请求并运行命令的 Systems Manager agent 的名称。

类型：字符串

必需：是

allowDowngrade

允许将 SSM Agent 降级为早期版本。如果设置为 False，则只能将该代理升级为更新的版本 (默认)。如果设置为 True，则指定早期版本。

类型：布尔值

必需：是

源

复制 Systems Manager 要安装的 SSM Agent 版本的位置。您无法更改此位置。

类型：字符串

必需：是

targetVersion

要安装的特定版本的 SSM Agent。如果未指定，代理将更新到最新版本。

类型：字符串

必需：否

创建 SSM 文档内容

如果 AWS Systems Manager 公有文档不执行要对 AWS 资源执行的所有操作，您可以创建自己的 SSM 文档。您还可以使用控制台克隆 SSM 文档。克隆文档将内容从现有文档复制到可以修改的新文档。创建或克隆文档时，文档的内容不得超过 64KB。此配额还包括在运行时指定的输入参数内容。创建新的 `Command` 或 `Policy` 文档时，我们建议您使用版本 2.2 或更高版本的架构，以便利用最新功能，例如文档编辑、自动版本控制、排序等。

编写 SSM 文档内容

要创建您自己的 SSM 文档内容，请务必了解可用于 SSM 文档的不同架构、功能、插件和语法。我们建议您熟悉以下资源。

- 编写您自己的 AWS Systems Manager 文档
- 数据元素和参数
- 架构、功能和示例
- 命令文档插件参考
- Systems Manager 自动化操作参考
- 自动化系统变量
- 其他运行手册示例
- 使用AWS Toolkit for Visual Studio Code 处理 Systems Manager 自动化运行手册
- 自动化运行手册的视觉对象设计体验
- 在运行手册中使用脚本

AWS 预定义的 SSM 文档可能会执行您所需的一些操作。您可以通过在自定义 SSM 文档中使用 `aws:runDocument`、`aws:runCommand` 或 `aws:executeAutomation` 插件来调用这些文档，具体

取决于文档类型。您还可以将这些文档的某些部分复制到自定义的 SSM 文档中，并编辑内容以满足您的要求。

Tip

创建 SSM 文档内容时，您可能会在测试时多次更改内容并更新 SSM 文档。以下命令使用最新内容更新 SSM 文档，并将文档的默认版本更新为文档的最新版本。

> **Note**
>
> Linux 和 Windows 命令使用 `jq` 命令行工具筛选 JSON 响应数据。

Linux & macOS

```
latestDocVersion=$(aws ssm update-document \
    --content file://path/to/file/documentContent.json \
    --name "ExampleDocument" \
    --document-format JSON \
    --document-version '$LATEST' \
    | jq -r '.DocumentDescription.LatestVersion')

aws ssm update-document-default-version \
    --name "ExampleDocument" \
    --document-version $latestDocVersion
```

Windows

```
latestDocVersion=$(aws ssm update-document ^
    --content file://C:\path\to\file\documentContent.json ^
    --name "ExampleDocument" ^
    --document-format JSON ^
    --document-version "$LATEST" ^
    | jq -r '.DocumentDescription.LatestVersion')

aws ssm update-document-default-version ^
    --name "ExampleDocument" ^
    --document-version $latestDocVersion
```

PowerShell

```
$content = Get-Content -Path "C:\path\to\file\documentContent.json" | Out-
String
$latestDocVersion = Update-SSMDocument `
    -Content $content `
    -Name "ExampleDocument" `
    -DocumentFormat "JSON" `
    -DocumentVersion '$LATEST' `
    | Select-Object -ExpandProperty LatestVersion

Update-SSMDocumentDefaultVersion `
    -Name "ExampleDocument" `
    -DocumentVersion $latestDocVersion
```

SSM 文档的安全最佳实践

创建 SSM 文档时，请遵循以下安全最佳实践，以帮助防范命令注入攻击并确保安全处理参数：

- 对将用于命令或脚本的字符串参数使用环境变量插值。为字符串参数添加值为 ENV_VAR 的 `interpolationType` 属性：

```
{
    "command": {
        "type": "String",
        "description": "Command to execute",
        "interpolationType": "ENV_VAR"
    }
}
```

您可以指定插值生成的值不接受双引号，从而进一步增强 SSM 文档的安全性：

```
{
    "command": {
        "type": "String",
        "description": "Command to execute",
        "interpolationType": "ENV_VAR",
            "allowedPattern": "^[^"]*$"
    }
```

```
}
```

- 在使用 Python、Ruby 或 Node.js 等解释性语言时，请使用恰当的环境变量语法来引用参数：

```
# Python example
import os
command = os.environ['SSM_Message']
```

- 为了向后兼容旧版 SSM Agent（3.3.2746.0 版之前），请为环境变量添加回退逻辑：

```
if [ -z "${SSM_command+x}" ]; then
    export SSM_command="{{command}}"
fi
```

- 将环境变量插值与 `allowedPattern` 相结合，提供额外的输入验证。在以下示例中，`allowedPattern` 的值 `^[^"]*$` 专门用于防止在字符串值中使用双引号：

```
{
    "command": {
        "type": "String",
        "interpolationType": "ENV_VAR",
        "allowedPattern": "^[a-zA-Z0-9_-]+$"
    }
}
```

- 在实现 SSM 文档之前，请确认以下安全注意事项：
  - 适用时，所有接受用户输入的字符串参数均使用环境变量插值。
  - 尽可能使用 `allowedPattern` 来实现输入验证。
  - 该文档包含了处理参数的恰当错误处理方法。
  - 对使用旧版 SSM Agent 的环境保持了向后兼容性。

克隆 SSM 文档

您可以克隆 AWS Systems Manager 文档，使用 Systems Manager 文档控制台创建 SSM 文档。克隆 SSM 文档将内容从现有文档复制到可以修改的新文档中。无法克隆大于 64KB 的文档。

克隆 SSM 文档

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。

2. 在导航窗格中，选择文档。
3. 在搜索框中，输入要克隆的文档的名称。
4. 选择要克隆的文档的名称，然后选择操作下拉菜单中的克隆文档。
5. 根据需要修改文档，然后选择创建文档保存文档。

编写 SSM 文档内容后，您可以通过以下方法之一使用您的内容创建 SSM 文档。

创建 SSM 文档

- 创建复合文档

创建复合文档

复合 AWS Systems Manager (SSM) 文档是自定义文档，可通过运行一个或多个次要 SSM 文档执行一系列操作。复合文档允许您为常见任务（例如引导软件或域加入实例）创建一组标准 SSM 文档，从而升级基础设施代码。您随后可以在同一个 AWS 区域 中跨 AWS 账户 共享这些文档，从而减少 SSM 文档的维护工作并确保文档一致性。

例如，您可以创建执行以下操作的复合文档：

1. 安装允许列表中的所有修补程序。
2. 安装防病毒软件。
3. 从 GitHub 下载脚本并运行这些脚本。

在本例中，您的自定义 SSM 文档包含执行下面这些操作的以下插件：

1. 用于运行 `AWS-RunPatchBaseline` 文档的 `aws:runDocument` 插件，它将安装所有允许列出的补丁。
2. 可运行 `AWS-InstallApplication` 文档的 `aws:runDocument` 插件，这会安装防病毒软件。
3. 可从 GitHub 下载脚本并运行这些脚本的 `aws:downloadContent` 插件。

复合和次要文档可存储在 Systems Manager、GitHub（公有和私有存储库）或 Amazon S3 中。可创建 JSON 或 YAML 格式的复合文档和次要文档。

> **Note**
>
> 复合文档只能运行三个文档的最大文档深度。这意味着复合文档可以调用子文档；该子文档可调用最后一个文档。

要创建复合文档，请在自定义 SSM 文档中添加 `aws:runDocument` 插件并指定必需的输入。下面是可执行以下操作的复合文档示例：

1. 运行 `aws:downloadContent` 插件，可从 GitHub 公有存储库将 SSM 文档下载到名为 bootstrap 的本地目录。SSM 文档称为 StateManagerBootstrap.yml (YAML 文档)。
2. 运行 `aws:runDocument` 插件可运行 StateManagerBootstrap.yml 文档。未指定任何参数。
3. 运行 `aws:runDocument` 插件以运行 `AWS-ConfigureDocker pre-defined` SSM 文档。指定的参数会在实例上安装 Docker。

```
{
  "schemaVersion": "2.2",
  "description": "My composite document for bootstrapping software and installing
 Docker.",
  "parameters": {
  },
  "mainSteps": [
    {
      "action": "aws:downloadContent",
      "name": "downloadContent",
      "inputs": {
        "sourceType": "GitHub",
        "sourceInfo": "{\"owner\":\"TestUser1\",\"repository\":\"TestPublic\", \"path
\":\"documents/bootstrap/StateManagerBootstrap.yml\"}",
        "destinationPath": "bootstrap"
      }
    },
    {
      "action": "aws:runDocument",
      "name": "runDocument",
      "inputs": {
        "documentType": "LocalPath",
        "documentPath": "bootstrap",
        "documentParameters": "{}"
      }
```

```
    },
    {
      "action": "aws:runDocument",
      "name": "configureDocker",
      "inputs": {
        "documentType": "SSMDocument",
        "documentPath": "AWS-ConfigureDocker",
        "documentParameters": "{\"action\":\"Install\"}"
      }
    }
  ]
}
```

更多信息

- 有关在使用 Run Command 调用脚本时重启服务器和实例的信息，请参阅 运行命令时处理重启问题。
- 有关可添加到自定义 SSM 文档的插件的更多信息，请参阅 命令文档插件参考。
- 如果只想从远程位置运行文档 (无需创建复合文档)，请参阅 从远程位置运行文档。

使用文档

本节包含有关如何使用 SSM 文档的信息。

主题

- 比较 SSM 文档版本
- 创建 SSM 文档
- 删除自定义 SSM 文档
- 从远程位置运行文档
- 共享 SSM 文档
- 搜索 SSM 文档

比较 SSM 文档版本

您可以在 Systems Manager 文档控制台中比较 AWS Systems Manager (SSM) 文档版本之间的内容差异。比较 SSM 文档的版本时，会突出显示版本内容之间的差异。

比较 SSM 文档内容（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择文档。
3. 在文档列表中，选择要比较其内容的文档。
4. 在存储库的内容选项卡上，选择比较版本，然后选择要将内容与其进行比较的文档版本。

创建 SSM 文档

如 编写 SSM 文档内容 中所述，为自定义 SSM 文档创建内容后，您可以通过 Systems Manager 控制台使用您的内容创建 SSM 文档。

要创建 SSM 文档

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择文档。
3. 选择 Create command or session (创建命令或会话)。
4. 输入文档的描述性名称。
5. （可选）对于 Target type (目标类型)，指定可以运行文档的资源类型。
6. 在文档类型列表中，选择您要创建的文档类型。
7. 删除 Content (内容) 字段中的方括号，然后将您之前创建的文档内容粘贴到这里。
8. （可选）在 Document tags (文档标签) 部分，将一个或多个标签键名称/值对应用于文档。

   标签是您分配给资源的可选元数据。标签可让您按各种标准（如用途、所有者或环境）对资源进行分类。例如，您可能希望标记文档以标识它运行的任务类型、它指向的操作系统的类型以及它在其中运行的环境。在这种情况下，您可以指定以下键名/键值对：

   - `Key=TaskType,Value=MyConfigurationUpdate`
   - `Key=OS,Value=AMAZON_LINUX_2`
   - `Key=Environment,Value=Production`
9. 选择创建文档以保存文档。

删除自定义 SSM 文档

如果您不再想使用自定义 SSM 文档，则可以使用 AWS Systems Manager 控制台来删除它。

要删除 SSM 文档

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择文档。
3. 选择要删除的角色。
4. 选择删除。在提示删除文档时，选择删除。

有关使用命令行工具或 SDK 删除 SSM 文档的示例，请参阅 将 DeleteDocument 与 AWS SDK 或 CLI 配合使用。

从远程位置运行文档

您可以使用 AWS-RunDocument 预定义的 SSM 文档从远程位置运行 AWS Systems Manager (SSM) 文档。此文档支持运行存储在以下位置的 SSM 文档：

- 公共和私有 GitHub 存储库（不支持 GitHub Enterprise）
- Amazon S3 存储桶
- Systems Manager

虽然您也可以使用State Manager或 Automation（都是 AWS Systems Manager 中的工具）运行远程文档，下面的过程仅介绍了如何在 Systems Manager 控制台中使用 AWS Systems Manager Run Command 运行远程 SSM 文档。

> **Note**
>
> AWS-RunDocument 仅可用于运行命令类型的 SSM 文档，而不是其他类型（如自动化运行手册）。AWS-RunDocument 使用 aws:downloadContent 插件。有关 aws:downloadContent 插件的更多信息，请参阅 aws:downloadContent。

开始前的准备工作

必须先完成以下任务才能运行远程文档。

- 创建 SSM 命令文档并在远程位置保存该文档。有关更多信息，请参阅 创建 SSM 文档内容。
- 如果打算运行存储在私有 GitHub 存储库中的远程文档，则必须为 GitHub 安全访问令牌创建 Systems Manager SecureString 参数。通过 SSH 手动传递令牌无法访问私有 GitHub 存储库中的远程文档。该访问令牌必须作为 Systems Manager SecureString 参数传递。有关创建 SecureString 参数的更多信息，请参阅 在 Systems Manager 中创建 Parameter Store 参数。

运行远程文档（控制台）

运行远程文档

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Run Command。
3. 选择 Run command（运行命令）。
4. 在 文档列表中，选择 **AWS-RunDocument**。
5. 在 命令参数 中，对于 源类型，请选择一个选项。

   - 如果选择 GitHub，则采用以下格式指定源信息信息：

   ```
   {
       "owner": "owner_name",
       "repository": "repository_name",
       "path": "path_to_document",
       "getOptions":"branch:branch_name",
       "tokenInfo": "{{ssm-secure:secure-string-token}}"
   }
   ```

   例如：

   ```
   {
       "owner":"TestUser",
       "repository":"GitHubTestExamples",
       "path":"scripts/python/test-script",
       "getOptions":"branch:exampleBranch",
       "tokenInfo":"{{ssm-secure:my-secure-string-token}}"
   }
   ```

> **Note**
>
> getOptions 是用于从分支（而非主分支）或存储库中的特定提交中检索内容的额外选项。如果您在主分支中使用最新提交，则可以省略 getOptions。仅当您的 SSM 文档存储在 master 以外的分支中时，branch 参数才是必需的。
> 要使用存储库中特定提交 中的 SSM 文档版本，请使用 commitID 和 getOptions，而不是 branch。例如：
>
> ```
> "getOptions": "commitID:bbc1ddb94...b76d3bEXAMPLE",
> ```

- 如果选择 S3，则采用以下格式指定源信息信息：

  ```
  {"path":"URL_to_document_in_S3"}
  ```

  例如：

  ```
  {"path":"https://s3.amazonaws.com/amzn-s3-demo-bucket/scripts/ruby/
  mySSMdoc.json"}
  ```

- 如果选择 SSMDocument，则采用以下格式指定源信息信息：

  ```
  {"name": "document_name"}
  ```

  例如：

  ```
  {"name": "mySSMdoc"}
  ```

6. 在 文档参数 字段中，输入远程 SSM 文档的参数。例如，如果运行 AWS-RunPowerShell 文档，则可指定：

   ```
   {"commands": ["date", "echo \"Hello World\""]}
   ```

   如果运行 AWS-ConfigureAWSPack 文档中，则可指定：

   ```
   {
      "action":"Install",
      "name":"AWSPVDriver"
   ```

```
}
```

7. 在 Targets（目标）部分中，通过指定标签、手动选择实例或边缘设备或指定资源组，选择要在其上运行此操作的托管式节点。

> **Tip**
>
> 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

8. 对于 Other parameters（其他参数）：

   - 对于 Comment（注释），请输入有关此命令的信息。
   - 对于 Timeout (seconds) (超时 (秒))，请指定在整个命令执行失败之前系统等待的秒数。

9. 对于 Rate control（速率控制）：

   - 对于 Concurrency（并发），请指定要同时运行该命令的托管式节点的数量或百分比。

   > **Note**
   >
   > 如果您通过指定应用于托管式节点的标签或指定 AWS Resource Groups 来选择目标，但不确定有多少个托管式节点已被设为目标，则可通过指定百分比来限制可同时运行该文档的目标的数量。

   - 对于 Error threshold（错误阈值），请指定当命令在一定数量或百分比的节点上失败后，何时在其他托管式节点上停止运行该命令。例如，如果您指定三个错误，Systems Manager 将在收到第四个错误时停止发送该命令。仍在处理该命令的托管式节点也可能发送错误。

10. （可选）对于 输出选项，要将命令输出保存到文件，请选中 将命令输出写入 S3 存储桶 框。在输入框中输入存储桶和前缀（文件夹）名称。

> **Note**
>
> 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给实例的实例配置文件（适用于 EC2 实例）或 IAM 服务角色（混合激活的计算机）的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确保与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

11. 在 SNS 通知部分，如果需要发送有关命令执行状态的通知，请选中 Enable SNS notifications（启用 SNS 通知）复选框。

    有关为 Run Command 配置 Amazon SNS 通知的更多信息，请参阅 使用 Amazon SNS 通知监控 Systems Manager 状态更改。
12. 选择 Run（运行）。

> **Note**
>
> 有关在使用 Run Command 调用脚本时重启服务器和实例的信息，请参阅 运行命令时处理重启问题。

共享 SSM 文档

您可以与同一个 AWS 区域 中的账户私下或公开共享 AWS Systems Manager (SSM) 文档。要私下共享文档，请修改文档权限并允许特定个人根据其 AWS 账户 ID 访问文档。要公开共享 SSM 文档，请修改文档权限并指定 `All` 。不能同时公开和私下共享文档。

> **Warning**
>
> 请仅使用从可信来源获取的共享 SSM 文档。使用任何共享文档时，请务必在使用前仔细查看文件内容，了解它会如何更改您的实例配置。有关共享文档最佳实践的更多信息，请参阅 共享 SSM 文档的最佳做法。

限制

在开始使用 SSM 文档时，请注意下列限制。

- 仅所有者可共享文档。
- 您必须先停止共享文档，然后才能删除它。有关更多信息，请参阅 修改共享 SSM 文档的权限。
- 您最多可与 1000 个 AWS 账户 共享一个文档。您可以在 支持 中心中请求提高该限制。对于限制类型，选择 EC2 Systems Manager 并描述您的请求理由。
- 您可公开共享最多 5 个 SSM 文档。您可以在 支持支持中心中请求提高该限制。对于限制类型，选择 EC2 Systems Manager 并描述您的请求理由。
- 文档只能与同一个 AWS 区域 中的其他账户共享。不支持跨区域共享。

> **Important**
>
> 在 Systems Manager 中，Amazon 拥有的 SSM 文档是由 Amazon Web Services 自己创建和管理的文档。Amazon 拥有的文档在文档名称中包含 `AWS-*` 之类的前缀。文档的所有者被视为 Amazon，而不是 AWS 内的特定用户账户。这些文档可供所有人公开使用。

有关 Systems Manager Service Quotas 的更多信息，请参阅 AWS Systems ManagerService Quotas 。

内容

- 共享 SSM 文档的最佳做法
- 阻止 SSM 文档的公开共享
- 共享 SSM 文档
- 修改共享 SSM 文档的权限
- 使用共享 SSM 文档

共享 SSM 文档的最佳做法

在共享或使用共享文档之前，请阅读以下指南。

删除敏感信息

请仔细审查您的 AWS Systems Manager (SSM) 文档并删除任何敏感信息。例如，请确保文档不包含您的 AWS 凭证。如果您与特定个人共享文档，这些用户可查看文档中的信息。如果您公开共享文档，则任何人都可查看文档中的信息。

阻止文档的公开共享

查看您账户中所有公开共享的 SSM 文档，并确认是否要继续共享这些文档。要停止与公众共享文档，必须按照本主题 修改共享 SSM 文档的权限 部分所述来修改文档权限设置。启用“阻止公开共享”设置不会影响您当前与公众共享的任何文档。除非您的应用场景需要您与公众共享文档，否则建议在 Systems Manager 文档控制台的首选项部分，为您的 SSM 文档开启阻止公开共享设置。启用此设置可防止对 SSM 文档进行不需要的访问。阻止公开共享设置是一个帐户级别设置，每个 AWS 区域 的设置可能不同。

使用 IAM 信任策略限制 Run Command 操作

为将有权访问该文档的用户创建限制性 AWS Identity and Access Management（IAM）policy。IAM policy 确定用户可在 Amazon Elastic Compute Cloud（Amazon EC2）控制台中或通过使用 AWS Command Line Interface（AWS CLI）或 AWS Tools for Windows PowerShell 调用 `ListDocuments` 查看哪些 SSM 文档。该策略还限制用户可使用 SSM 文档执行的操作。您可创建限制性策略，以便用户只能使用特定文档。有关更多信息，请参阅 客户管理型策略示例。

使用共享 SSM 文档时要小心

审查与您共享的每个文档（特别是公开文档）的内容，以了解将在您的实例上运行的命令。一个文档在运行后可能会有意或无意具有负面影响。如果文档引用外部网络，请在使用文档前审查外部源。

使用文档哈希发送命令

在共享文档时，系统将创建 Sha-256 哈希并将其分配给文档。系统还将保存文档内容的快照。使用共享文档发送命令时，您可在命令中指定哈希以确保下列条件为 true：

- 您正在从正确的 Systems Manager 文档运行命令
- 在与您共享之后文档内容未更改。

如果哈希与指定文档不匹配，或者共享文档的内容已更改，则命令将返回 `InvalidDocument` 异常。哈希无法验证来自外部位置的文档内容。

使用插值参数来增强安全性

对于 SSM 文档中的 `String` 类型参数，可使用参数和值 `interpolationType": "ENV_VAR` 增强对命令注入攻击的防御能力，方法是将参数输入视为字符串文本值，而非可能的可执行命令。在这种情况下，代理会创建一个包含参数值的环境变量 `SSM_parameter-name`。我们建议对现有所有包含 `String` 类型参数的 SSM 文档进行更新，以纳入 `"interpolationType": "ENV_VAR"`。有关更多信息，请参阅 the section called “编写 SSM 文档内容”。

阻止 SSM 文档的公开共享

开始前，请查看您 AWS 账户 中所有公开共享的 SSM 文档，并确认是否要继续共享它们。要停止与公众共享 SSM 文档，必须按照本主题 修改共享 SSM 文档的权限 部分所述修改文档权限设置。启用“阻止公开共享”设置不会影响您当前与公众共享的任何 SSM 文档。启用“阻止公开共享”设置后，您将无法与公众共享任何其他 SSM 文档。

除非您的应用场景需要您与公众共享文档，否则我们建议为您的 SSM 文档开启阻止公开共享设置。启用此设置可防止对 SSM 文档进行不需要的访问。阻止公开共享设置是一个帐户级别设置，每个 AWS 区域 的设置可能不同。完成以下任务以阻止您目前正在共享的任何 SSM 文档的公开共享。

阻止公开共享（控制台）

要阻止 SSM 文档的公开共享

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择文档。
3. 选择首选项，然后在阻止公开共享部分中选择编辑。
4. 选择阻止公开共享复选框，然后选择保存。

阻止公开共享（命令行）

打开 AWS Command Line Interface (AWS CLI）或 AWS Tools for Windows PowerShell 并运行以下命令以阻止 SSM 文档的公开共享。

Linux & macOS

```
aws ssm update-service-setting  \
    --setting-id /ssm/documents/console/public-sharing-permission \
    --setting-value Disable \
    --region 'The AWS ## you want to block public sharing in'
```

Windows

```
aws ssm update-service-setting ^
    --setting-id /ssm/documents/console/public-sharing-permission ^
    --setting-value Disable ^
    --region "The AWS ## you want to block public sharing in"
```

PowerShell

```
Update-SSMServiceSetting `
    -SettingId /ssm/documents/console/public-sharing-permission `
    -SettingValue Disable `
    -Region The AWS ## you want to block public sharing in
```

使用以下命令确认设置值已更新。

Linux & macOS

```
aws ssm get-service-setting   \
    --setting-id /ssm/documents/console/public-sharing-permission \
    --region The AWS ## you blocked public sharing in
```

Windows

```
aws ssm get-service-setting  ^
    --setting-id /ssm/documents/console/public-sharing-permission ^
    --region "The AWS ## you blocked public sharing in"
```

PowerShell

```
Get-SSMServiceSetting `
    -SettingId /ssm/documents/console/public-sharing-permission `
    -Region The AWS ## you blocked public sharing in
```

使用 IAM 限制访问阻止公开共享

您可以创建 AWS Identity and Access Management (IAM) 策略，这些策略限制用户修改阻止公开共享设置。这可以防止用户允许对 SSM 文档进行不需要的访问。

以下是一个 IAM policy 示例，该策略阻止用户更新阻止公开共享设置。要使用此示例，您必须将示例 Amazon Web Services 账户 ID 替换为您自己的账户 ID。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "ssm:UpdateServiceSetting",
            "Resource": "arn:aws:ssm:*:444455556666:servicesetting/ssm/documents/
console/public-sharing-permission"
        }
    ]
```

```
}
```

共享 SSM 文档

您可以通过 Systems Manager 控制台共享 AWS Systems Manager (SSM) 文档。从控制台共享文档时，只能共享文档的默认版本。您还可以通过使用 AWS Command Line Interface (AWS CLI)、 AWS Tools for Windows PowerShell 或 AWS SDK 调用 `ModifyDocumentPermission` API 操作，以编程方式共享 SSM 文档。在共享文档之前，获取要与之共享文档的人的 AWS 账户 账户 ID。您将在共享文档时指定这些账户 ID。

共享文档（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择文档。
3. 在文档列表中，选择要共享的文档，然后选择查看详细信息。在权限选项卡中，确保您是文档所有者。只有文档所有者才可共享文档。
4. 选择编辑。
5. 要公开共享命令，请选择公开，然后选择保存。要私下共享命令，请选择私有，输入 AWS 账户 ID，选择添加权限，然后选择保存。

共享文档（命令行）

以下过程要求您为命令行会话指定 AWS 区域。

1. 在本地计算机上打开 AWS CLI 或 AWS Tools for Windows PowerShell 并运行以下命令来指定凭证。

   在下面的命令中，将 *region* 替换为您自己的信息。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

   Linux & macOS

   ```
   aws config

   AWS Access Key ID: [your key]
   AWS Secret Access Key: [your key]
   Default region name: region
   ```

```
Default output format [None]:
```

Windows

```
aws config

AWS Access Key ID: [your key]
AWS Secret Access Key: [your key]
Default region name: region
Default output format [None]:
```

PowerShell

```
Set-AWSCredentials -AccessKey your key -SecretKey your key
Set-DefaultAWSRegion -Region region
```

2. 使用以下命令列出可供您使用的所有 SSM 文档。此列表包括您已创建的文档和与您共享的文档。

Linux & macOS

```
aws ssm list-documents
```

Windows

```
aws ssm list-documents
```

PowerShell

```
Get-SSMDocumentList
```

3. 使用以下命令获取特定文档。

Linux & macOS

```
aws ssm get-document \
    --name document name
```

Windows

```
aws ssm get-document ^
```

```
    --name document name
```

PowerShell

```
Get-SSMDocument `
    -Name document name
```

4. 使用以下命令获取文档的描述。

Linux & macOS

```
aws ssm describe-document \
    --name document name
```

Windows

```
aws ssm describe-document ^
    --name document name
```

PowerShell

```
Get-SSMDocumentDescription `
    -Name document name
```

5. 使用以下命令查看文档的权限。

Linux & macOS

```
aws ssm describe-document-permission \
    --name document name \
    --permission-type Share
```

Windows

```
aws ssm describe-document-permission ^
    --name document name ^
    --permission-type Share
```

PowerShell

```
Get-SSMDocumentPermission `
    -Name document name `
    -PermissionType Share
```

6. 使用以下命令修改文档的权限并共享文档。您必须是文档的所有者才能编辑权限。或者，对于与特定 AWS 账户 ID 共享的文档，您可以使用 --shared-document-version 参数指定要共享的文档的版本。如果您不指定版本，系统将共享文档的Default版本。如果您公开（与 all）共享文档，则默认共享指定文档的所有版本。以下示例命令根据特定个人的 AWS 账户 ID 与该个人私下共享文档。

   Linux & macOS

   ```
   aws ssm modify-document-permission \
       --name document name \
       --permission-type Share \
       --account-ids-to-add AWS ## ID
   ```

   Windows

   ```
   aws ssm modify-document-permission ^
       --name document name ^
       --permission-type Share ^
       --account-ids-to-add AWS ## ID
   ```

   PowerShell

   ```
   Edit-SSMDocumentPermission `
       -Name document name `
       -PermissionType Share `
       -AccountIdsToAdd AWS ## ID
   ```

7. 使用以下命令公开共享文档。

   > **Note**
   >
   > 如果您公开（与 all）共享文档，则默认共享指定文档的所有版本。

Linux & macOS

```
aws ssm modify-document-permission \
    --name document name \
    --permission-type Share \
    --account-ids-to-add 'all'
```

Windows

```
aws ssm modify-document-permission ^
    --name document name ^
    --permission-type Share ^
    --account-ids-to-add "all"
```

PowerShell

```
Edit-SSMDocumentPermission `
    -Name document name `
    -PermissionType Share `
    -AccountIdsToAdd ('all')
```

修改共享 SSM 文档的权限

如果您共享一条命令，则在您删除对 AWS Systems Manager (SSM) 文档的访问权限或删除 SSM 文档之前，用户可查看和使用该命令。但是，只要文档已共享，您就无法删除它。您必须先停止共享，然后再删除它。

停止共享文档（控制台）

停止共享文档

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择文档。
3. 在文档列表中，选择要停止共享的文档，然后选择详细信息。在权限部分中，确认自己是文档所有者。只有文档所有者才可停止共享文档。
4. 选择编辑。

5. 选择 X 以删除不应再具有此命令的访问权限的 AWS 账户 ID，然后选择保存。

停止共享文档（命令行）

在本地计算机上打开 AWS CLI 或 AWS Tools for Windows PowerShell，然后运行以下命令停止共享命令。

Linux & macOS

```
aws ssm modify-document-permission \
    --name document name \
    --permission-type Share \
    --account-ids-to-remove 'AWS ## ID'
```

Windows

```
aws ssm modify-document-permission ^
    --name document name ^
    --permission-type Share ^
    --account-ids-to-remove "AWS ## ID"
```

PowerShell

```
Edit-SSMDocumentPermission `
    -Name document name `
    -PermissionType Share `
    -AccountIdsToRemove AWS ## ID
```

使用共享 SSM 文档

共享 AWS Systems Manager (SSM) 文档时，系统将生成一个 Amazon Resource Name (ARN) 并将其分配给命令。如果您从 Systems Manager 控制台选择并运行某个共享文档，则不会看到此 ARN。但如果您要使用 Systems Manager 控制台以外的方法运行共享 SSM 文档，则必须在 `DocumentName` 请求参数中指定文档的完整 ARN。当您运行列出文档的命令时，将为您显示 SSM 文档的完整 ARN。

> **Note**
>
> 您无需为 AWS 公有文档（以 `AWS-*` 开头的文档）或您拥有的文档指定 ARN。

使用共享 SSM 文档（命令行）

列出所有公有 SSM 文档

Linux & macOS

```
aws ssm list-documents \
    --filters Key=Owner,Values=Public
```

Windows

```
aws ssm list-documents ^
    --filters Key=Owner,Values=Public
```

PowerShell

```
$filter = New-Object Amazon.SimpleSystemsManagement.Model.DocumentKeyValuesFilter
$filter.Key = "Owner"
$filter.Values = "Public"

Get-SSMDocumentList `
    -Filters @($filter)
```

列出已与您共享的私有 SSM 文档

Linux & macOS

```
aws ssm list-documents \
    --filters Key=Owner,Values=Private
```

Windows

```
aws ssm list-documents ^
    --filters Key=Owner,Values=Private
```

PowerShell

```
$filter = New-Object Amazon.SimpleSystemsManagement.Model.DocumentKeyValuesFilter
$filter.Key = "Owner"
```

```
$filter.Values = "Private"

Get-SSMDocumentList `
    -Filters @($filter)
```

列出可供您使用的所有 SSM 文档

Linux & macOS

```
aws ssm list-documents
```

Windows

```
aws ssm list-documents
```

PowerShell

```
Get-SSMDocumentList
```

获取有关已与您共享的 SSM 文档的信息

Linux & macOS

```
aws ssm describe-document \
    --name arn:aws:ssm:us-east-2:12345678912:document/documentName
```

Windows

```
aws ssm describe-document ^
    --name arn:aws:ssm:us-east-2:12345678912:document/documentName
```

PowerShell

```
Get-SSMDocumentDescription `
    -Name arn:aws:ssm:us-east-2:12345678912:document/documentName
```

运行共享的 SSM 文档

Linux & macOS

```
aws ssm send-command \
    --document-name arn:aws:ssm:us-east-2:12345678912:document/documentName \
    --instance-ids ID
```

Windows

```
aws ssm send-command ^
    --document-name arn:aws:ssm:us-east-2:12345678912:document/documentName ^
    --instance-ids ID
```

PowerShell

```
Send-SSMCommand `
    -DocumentName arn:aws:ssm:us-east-2:12345678912:document/documentName `
    -InstanceIds ID
```

搜索 SSM 文档

您可以使用自由文本搜索或基于筛选器的搜索在 AWS Systems Manager (SSM) 文档存储中搜索 SSM 文档。您还可以收藏文档，以帮助您查找常用 SSM 文档。以下部分介绍如何使用这些功能。

使用自由文本搜索

Systems Manager 文档页面上的搜索框支持自由文本搜索。自由文本搜索会将您输入的一个或多个搜索词与每个 SSM 文档中的文档名称进行比较。如果输入单个搜索词，例如 **`ansible`**，然后 Systems Manager 返回发现此术语的所有 SSM 文档。如果输入多个搜索词，Systems Manager 将使用 OR 网页。例如，如果指定 **`ansible`** 和 **`linux`**，然后搜索将返回名称中包含任一关键字的所有文档。

如果输入自由文本搜索词并选择搜索选项（例如平台类型），则搜索将使用 AND 语句，并返回所有名称中包含关键字和指定平台类型的文档。

> **Note**
>
> 请注意以下有关自由文本搜索的详细信息。
>
> - 搜索不区分大小写。
> - 搜索词至少需要三个字符，且最多包含 20 个字符。
> - 自由文本搜索最多可接受五个搜索词。

- 如果在搜索词之间输入空格，系统会在搜索时包含空格。
- 您可以将自由文本搜索与其他搜索选项，如：文档类型或者平台类型结合使用。
- 文档名称前缀过滤器和自由文本搜索不能一起使用，它们是相互排斥的。

搜索 SSM 文档

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择文档。
3. 在搜索框中输入搜索词，然后按输入。

使用 AWS CLI 执行自由文本文档搜索

使用 CLI 执行自由文本文档搜索

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本。
2. 要使用单个术语执行自由文本文档搜索，请运行以下命令。在此命令中，用自己的信息替换*###*。

   ```
   aws ssm list-documents --filters Key="SearchKeyword",Values="search_term"
   ```

   以下为示例。

   ```
   aws ssm list-documents --filters Key="SearchKeyword",Values="aws-asg" --region us-
   east-2
   ```

   要使用多个术语进行搜索，可创建 AND 语句，运行以下命令。在此命令中，用自己的信息替换*### 1* 和*### 2*。

   ```
   aws ssm list-documents --filters
    Key="SearchKeyword",Values="search_term_1","search_term_2","search_term_3" --
   region us-east-2
   ```

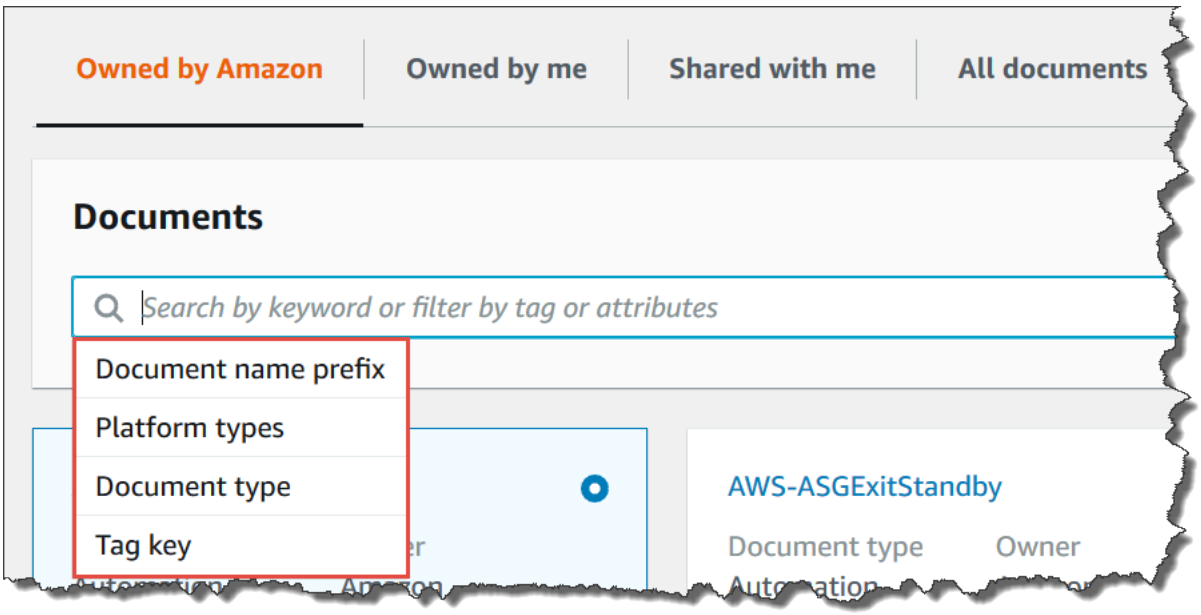
   以下为示例。

```
aws ssm list-documents --filters Key="SearchKeyword",Values="aws-asg","aws-
ec2","restart" --region us-east-2
```

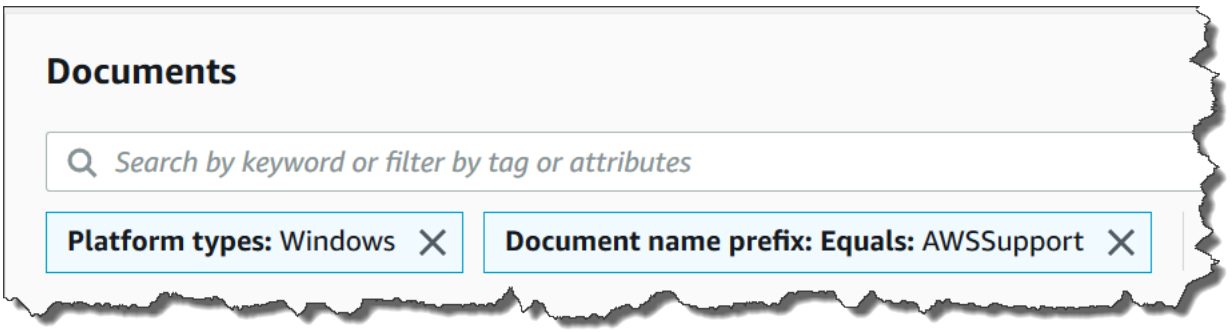
使用筛选条件

Systems Manager 文档页面会在您选择搜索框时自动显示以下筛选条件。

- 文档名称前缀
- 平台类型
- 文档类型
- 标签键



您可以使用单个筛选条件搜索 SSM 文档。如果要返回一组更具体的 SSM 文档，可以应用多个筛选条件。下面是一个搜索示例，使用平台类型和文档名称前缀筛选条件。



如果应用多个筛选条件，Systems Manager 会根据您选择的筛选条件创建不同的搜索语句：

- 如果您多次应用相同的筛选条件，例如文档名称前缀，然后 Systems Manager 将使用 OR 语句进行搜索。例如，如果您指定一个文档名称前缀**=AWS** 和第二个筛选条件文档名称前缀**=Lambda**，然后搜索将返回带有前缀 “AWS” 的所有文档以及带有前缀 “Lambda” 的所有文档。
- 如果您应用其他筛选条件，例如 Document name prefix（文档名称前缀）和 Platform types（平台类型），则 Systems Manager 将使用 AND 语句进行搜索。例如，如果指定 Document name prefix（文档名称前缀）=**AWS** 筛选条件和 Platform types（平台类型）=**Linux** 筛选条件，则搜索会返回特定于 Linux 平台的带有前缀“AWS”的所有文档。

> **Note**
>
> 使用筛选条件区分大小写。

将文档添加到您的收藏夹

为帮助您查找常用 SSM 文档，可将文档添加到您的收藏夹。每种文档类型、每种 AWS 账户 和 AWS 区域 最多可以收藏 20 份文档。您可以从文档 AWS Management Console 中选择、修改和查看您的收藏夹。以下过程介绍如何选择、修改和查看您的收藏夹。

收藏 SSM 文档

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择文档。
3. 选择您要收藏的文档名称旁边的星形图标。

从您的收藏夹中移除 SSM 文档

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择文档。
3. 取消选择要从您的收藏中移除的文档名称旁边的星形图标。

从文档 AWS Management Console 中查看您的收藏夹

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择文档。
3. 选择收藏夹选项卡。

参数处理问题的疑难解答

常见参数处理问题

执行期间环境变量不可用

问题：找不到环境变量 (SSM_*parameter-name*)，命令失败。

可能的原因：

- SSM Agent 版本不支持环境变量插值
- `interpolationType` 未设置为 `ENV_VAR`
- 参数名称与预期的环境变量名称不匹配

解决方案：

- 验证 SSM Agent 版本是否为 3.3.2746.0 或更高
- 为旧版代理添加回退逻辑：

```
if [ -z "${SSM_parameterName+x}" ]; then
    export SSM_parameterName="{{parameterName}}"
fi
```

参数值包含特殊字符

问题：参数值包含空格、引号或其他特殊字符时会导致命令失败。

解决方案：

- 引用环境变量时，请使用正确的引号：

```
# Correct
echo "$SSM_parameter-name"

# Incorrect
echo $SSM_parameter-name
```

- 添加输入验证，使用 allowedPattern 来限制特殊字符

跨平台行为不一致

问题：在 Linux 和 Windows Server 系统中，参数处理方式有所不同。

解决方案：

- 请使用特定平台的环境变量语法：

```
# PowerShell
$env:SSM_parameter-name

# Bash
$SSM_parameter-name
```

- 在文档中使用特定平台的前提条件检查

参数值未正确转义

问题：尽管使用了环境变量插值，但仍存在命令注入的漏洞。

解决方案：

- 在命令中包含参数值时，务必使用正确的转义：

```
# Correct
mysql_command="mysql -u \"$SSM_username\" -p\"$SSM_password\""

# Incorrect
mysql_command="mysql -u $SSM_username -p$SSM_password"
```

参数验证提示

使用这些方法验证参数处理：

1. 测试环境变量的可用性：

```
#!/bin/bash
# Print all SSM_ environment variables
env | grep ^SSM_

# Test specific parameter
if [ -n "$SSM_parameter" ]; then
    echo "Parameter is available"
else
    echo "Parameter is not available"
fi
```

2. 验证参数模式：

```
parameters:
  myParameter:
    type: String
    allowedPattern: "^[a-zA-Z0-9_-]+$"
    description: "Test this pattern with sample inputs"
```

3. 包含错误处理：

```
if [[ ! "$SSM_parameter" =~ ^[a-zA-Z0-9_-]+$ ]]; then
    echo "Parameter validation failed"
    exit 1
fi
```

AWS Systems Manager Maintenance Windows

Maintenance Windows（AWS Systems Manager 中的一项工具）可帮助您制定计划，规定何时在节点上执行可能造成中断的操作，例如修补操作系统、更新驱动程序或者安装软件或补丁。

> **Note**
>
> State Manager 和 Maintenance Windows 可在托管式节点上执行一些相似类型的更新。您选择哪一项取决于您是需要自动执行系统合规性，还是在指定的时间段内执行高优先级、时效性强的任务。
> 有关更多信息，请参阅 在 State Manager 和 Maintenance Windows 之间选择。

借助 Maintenance Windows，您可以为许多其他类型的 AWS 资源（例如 Amazon Simple Storage Service (Amazon S3) 存储桶、Amazon Simple Queue Service (Amazon SQS) 队列、AWS Key Management Service (AWS KMS) 键等等）上的操作制定计划。

有关可以包含在维护时段目标中的受支持资源类型的完整列表，请参阅 AWS Resource Groups 用户指南中的可与 AWS Resource Groups 和标签编辑器结合使用的资源。要开始使用 Maintenance Windows，请打开 Systems Manager 控制台。在导航窗格中，请选择 Maintenance Windows。

每个维护时段都有一个计划、一段最大持续时间、一组已注册的目标（要对其执行操作的托管式节点或其他 AWS 资源）和一组已注册的任务。在创建或更新维护时段时，您可以向其添加标签。（标签是帮助您在组织中识别和排序资源的键。）您还可以指定维护时段在某天之前或之后不应运行，以及维护时段计划所基于的国际时区。

有关维护时段的各种计划相关选项如何相互关联的说明，请参阅 维护时段计划和活动期间选项。

有关使用 `--schedule` 选项的更多信息，请参阅 参考：适用于 Systems Manager 的 Cron 和 Rate 表达式。

受支持的任务类型

对于维护时段，可运行四种类型的任务：

- Run Command（Systems Manager 中的一项工具）中的命令

  有关 Run Command的更多信息，请参阅AWS Systems Manager Run Command。
- Automation（Systems Manager 中的一项工具）中的工作流

  有关 自动化 工作流的更多信息，请参阅 AWS Systems Manager 自动化。
- AWS Lambda 中的函数

  有关 Lambda 函数的更多信息，请参阅 AWS Lambda 开发人员指南中的 Lambda 入门。
- AWS Step Functions 中的任务

> **Note**
>
> 维护时段任务仅支持 Step Functions 标准状态机工作流。它们不支持快速状态机工作流。有关状态机工作流类型的信息，请参阅《AWS Step Functions 开发人员指南》中的标准与快速工作流。

有关 Step Functions 的更多信息，请参阅《AWS Step Functions 开发人员指南》。

这意味着，您可以使用维护时段在选定目标上执行类似以下内容的任务。

- 安装或更新应用程序。
- 应用补丁。
- 安装或更新 SSM Agent。
- 使用 Systems Manager Run Command 任务运行 PowerShell 命令和 Linux Shell 脚本。
- 使用 Systems Manager 自动化任务构建 Amazon Machine Images (AMIs)、引导软件，以及配置节点。
- 运行调用其他操作（例如扫描您的节点是否有补丁更新）的 AWS Lambda 函数。
- 运行 AWS Step Functions 状态机以执行多项任务，例如删除 Elastic Load Balancing 环境中的某个节点，修补该节点，然后再将该节点添加回 Elastic Load Balancing 环境。
- 通过将 AWS 资源组指定为目标，将离线节点设为目标。

> **Note**
>
> 必须为维护时段 Run Command 类型的任务指定一个或多个目标。根据任务的不同，目标对于其他维护时段任务类型（自动化、AWS Lambda 和 AWS Step Functions）是可选的。有关运行未指定目标的任务的更多信息，请参阅 注册不含目标的维护时段任务。

EventBridge 支持

支持此 Systems Manager 工具作为 Amazon EventBridge 规则中的一个事件类型。有关信息，请参阅 使用 Amazon EventBridge 监控 Systems Manager 事件和 引用：Amazon EventBridge 事件模式和 Systems Manager 类型。

内容

- 设置 Maintenance Windows
- 使用控制台创建和管理维护时段
- 教程
- 注册维护时段任务时使用伪参数

- 维护时段计划和活动期间选项
- 注册不含目标的维护时段任务
- 对维护时段进行故障排除

设置 Maintenance Windows

您 AWS 账户中的用户必须先获得必要权限，然后才能使用 Maintenance Windows（AWS Systems Manager 中的一项工具）创建和计划维护时段任务。此外，您必须为维护时段创建 IAM 服务角色，以及要附加到该角色的 IAM 策略。

开始前的准备工作

除了您在本部分中配置的权限外，将在维护时段中使用的 IAM 实体（用户、角色或组）应该已经拥有常规维护时段权限。您可以通过将 IAM 策略 `AmazonSSMFullAccess` 分配给实体，或者分配一个自定义 IAM 策略来授予这些权限，自定义 IAM 策略可为处理维护时段任务的 Systems Manager 提供一组较小的访问权限集。

主题

- 使用控制台控制对维护时段的访问权限
- 使用 AWS CLI 控制对维护时段的访问权限

使用控制台控制对维护时段的访问权限

以下过程介绍如何使用 AWS Systems Manager 控制台为维护时段创建所需的权限和角色。

主题

- 任务 1：使用控制台为维护时段服务角色创建自定义策略
- 任务 2：使用控制台为维护时段创建自定义服务角色
- 任务 3：使用控制台向注册维护时段任务的特定用户授予权限
- 任务 4：使用控制台阻止指定用户注册维护时段任务

任务 1：使用控制台为维护时段服务角色创建自定义策略

维护时段任务需要一个 IAM 角色才能提供在目标资源上运行所需的权限。通过附加到角色的 IAM 策略提供权限。您运行的任务类型和其他操作要求决定了此策略的内容。我们提供了一个基本策略，您可以

根据需要进行调整。根据维护时段运行的任务和任务类型，您可能不需要此策略中的所有权限，并且可能需要包含额外的权限。您需要将此策略附加到您稍后在任务 2：使用控制台为维护时段创建自定义服务角色中创建的角色。

要使用控制台创建自定义策略

1. 访问：https://console.aws.amazon.com/iam/，打开 IAM 控制台。
2. 在导航窗格中，选择 Policies (策略)，然后选择 Create policy (创建策略)。
3. 在策略编辑器区域中，选择 JSON。
4. 将默认内容替换为以下内容：

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:SendCommand",
                "ssm:CancelCommand",
                "ssm:ListCommands",
                "ssm:ListCommandInvocations",
                "ssm:GetCommandInvocation",
                "ssm:GetAutomationExecution",
                "ssm:StartAutomationExecution",
                "ssm:ListTagsForResource",
                "ssm:DescribeInstanceInformation",
                "ssm:GetParameters"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "states:DescribeExecution",
                "states:StartExecution"
            ],
            "Resource": [
                "arn:aws:states:*:*:execution:*:*",
                "arn:aws:states:*:*:stateMachine:*"
```

```
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction"
            ],
            "Resource": [
                "arn:aws:lambda:*:*:function:*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "resource-groups:ListGroups",
                "resource-groups:ListGroupResources"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "tag:GetResources"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::111122223333:role/maintenance-window-
role-name",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ssm.amazonaws.com"
                    ]
                }
            }
        }
    ]
```

```
}
```

5. 根据需要为您在账户中运行的维护任务修改 JSON 内容。您做出的更改是特定于所计划操作的。

   例如：

   - 您可以提供特定功能和状态机的 Amazon 资源名称（ARN），而不是使用通配符（*）。
   - 如果您不计划运行 AWS Step Functions 任务，则可以移除 `states` 权限和 ARN。
   - 如果您不计划运行 AWS Lambda 任务，则可以移除 `lambda` 权限和 ARN。
   - 如果您不计划运行自动化任务，则可以移除 `ssm:GetAutomationExecution` 和 `ssm:StartAutomationExecution` 权限。
   - 添加要运行的任务可能需要的其他权限。例如，一些 自动化 操作使用 AWS CloudFormation 堆栈。因此，权限 `cloudformation:CreateStack`、`cloudformation:DescribeStacks` 和 `cloudformation:DeleteStack` 是必需的。

     另一个示例：自动化运行手册 `AWS-CopySnapshot` 需要权限来创建 Amazon Elastic Block Store（Amazon EBS）快照，因此，服务角色需要权限 `ec2:CreateSnapshot`。

     有关自动化运行手册所需角色权限的信息，请参阅 AWS Systems Manager 自动化运行手册参考中的运行手册描述。

6. 完成策略修订后，选择下一步。
7. 对于策略名称，输入一个名称以将此标识为您创建的服务角色所附加的策略。例如：**`my-maintenance-window-role-policy`**。
8. （可选）在添加标签区域中，添加一个或多个标签键值对，以组织、跟踪或控制此策略的访问权限。
9. 选择创建策略。

   记下为此策略指定的名称。您将在下一过程任务 2：使用控制台为维护时段创建自定义服务角色中引用它。

任务 2：使用控制台为维护时段创建自定义服务角色

您在上一个任务中创建的策略将附加到您在此任务中创建的维护时段服务角色。当用户注册维护时段任务时，他们需要在任务配置中指定此 IAM 角色。此角色中的权限将允许 Systems Manager 代表您运行维护时段中的任务。

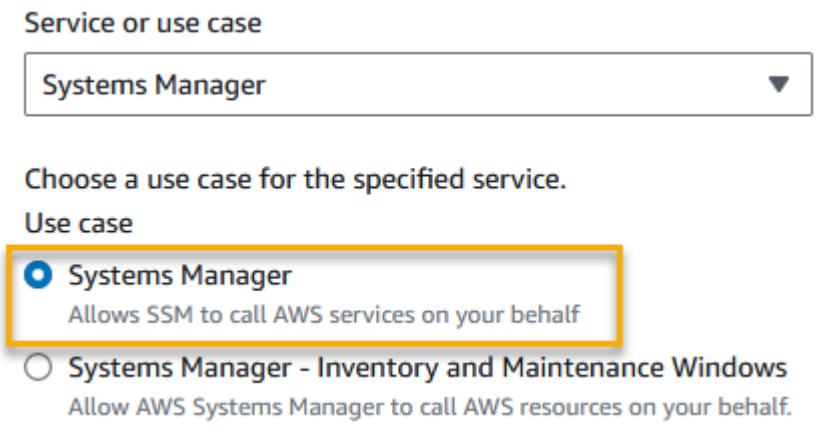
> **Important**
>
> 以前，Systems Manager 控制台允许您选择 AWS 托管式 IAM 服务相关角色 `AWSServiceRoleForAmazonSSM`，以用作任务的维护角色。现在不再建议将此角色及其相关策略 `AmazonSSMServiceRolePolicy` 用于维护时段任务。如果您目前在将此角色用于维护时段任务，我们建议您停止使用它。而应创建您自己的 IAM 角色，以便您的维护时段任务运行时在 Systems Manager 与其他 AWS 服务之间进行通信。

使用以下过程为 Maintenance Windows 创建自定义服务角色，以便 Systems Manager 可以代表您运行 Maintenance Windows 任务。您会将您在上一任务中创建的策略附加到您创建的自定义服务角色。

要使用控制台为维护时段创建自定义服务角色

1. 通过 https://console.aws.amazon.com/iam/ 打开 IAM 控制台。
2. 在导航窗格中，选择角色，然后选择创建角色。
3. 对于 Select trusted entity（选择可信实体），完成以下选择：
   1. 在可信实体类型中选择 AWS 服务。
   2. 对于应用场景，选择 Systems Manager
   3. 选择 Systems Manager。

   下图突出显示了 Systems Manager 选项的位置。



4. 选择下一步。
5. 在权限策略区域中的搜索框中，输入您在 任务 1：使用控制台为维护时段服务角色创建自定义策略 中创建的策略的名称，选中其名称旁边的复选框，然后选择下一步。

6. 对于 Role name（角色名称），输入用于将此角色标识为 Maintenance Windows 角色的名称。例如：**my-maintenance-window-role**。
7. （可选）更改默认角色描述，以反映此角色的用途。例如：**Performs maintenance window tasks on your behalf**。
8. 对于步骤 1：选择可信实体，请确认可信策略框中显示了以下策略。

   JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "ssm.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

9. 对于步骤 2：添加权限，请验证您在 任务 1：使用控制台为维护时段服务角色创建自定义策略 中创建的策略是否已存在。
10. （可选）在步骤 3：添加标签中，添加一个或多个标签键值对，以组织、跟踪或控制此角色的访问权限。
11. 选择 Create role (创建角色)。系统将让您返回到 角色 页面。
12. 选择刚才创建的 IAM 角色的名称。
13. 复制或记下在 Summary（摘要）区域显示的角色名称和 ARN 值。您账户中的用户在创建维护时段时需要指定此信息。

任务 3：使用控制台向注册维护时段任务的特定用户授予权限

为用户提供访问自定义维护时段服务角色的权限，使他们可以将该角色用于维护时段任务。这是对已授予权限的补充，以便与适用于 Maintenance Windows 工具的 Systems Manager API 命令结合使用。此 IAM 角色传递了运行维护时段任务所需的权限。因此，如果无法传递这些 IAM 权限，用户将无法使用您的自定义服务角色向维护时段注册任务。

在将任务注册到维护时段时，您可以指定服务角色来运行实际任务操作。这是代表您运行任务时服务要代入的角色。在此之前，要注册任务本身，请将 IAM PassRole 策略分配给 IAM 实体（如用户或组）。这将允许 IAM 实体指定运行任务时应使用的角色，作为将这些任务注册到维护时段的一部分。有关更多信息，请参阅《IAM 用户指南》中的向用户授予将角色传递给 AWS 服务 的权限。

要配置权限以允许用户注册维护时段任务

如果将 IAM 实体（用户、角色或组）设置为拥有管理员权限，则 IAM 用户或角色将有权访问维护时段。对于没有管理员权限的 IAM 实体，管理员必须向 IAM 实体授予以下权限。以下是将任务注册到维护时段所需的最低权限：

- AmazonSSMFullAccess 托管策略，或提供类似权限的策略。
- 以下 iam:PassRole 和 iam:ListRoles 权限。

  JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::111122223333:role/my-maintenance-window-role"
        },
        {
            "Effect": "Allow",
            "Action": "iam:ListRoles",
            "Resource": "arn:aws:iam::111122223333:role/"
        },
        {
            "Effect": "Allow",
            "Action": "iam:ListRoles",
            "Resource": "arn:aws:iam::111122223333:role/aws-service-role/ssm.amazonaws.com/"
        }
    ]
}
```

my-maintenance-window-role 表示您之前创建的自定义维护时段服务角色的名称。

account-id 表示 AWS 账户 的 ID。为资源 arn:aws:iam::*account-id*:role/ 添加此权限允许用户在创建维护时段任务时在控制台中查看和选择客户角色。为 arn:aws:iam::*account-id*:role/aws-service-role/ssm.amazonaws.com/ 添加此权限允许用户在创建维护时段任务时在控制台中选择 Systems Manager 服务相关角色。

要提供访问权限，请为您的用户、组或角色添加权限：

- AWS IAM Identity Center 中的用户和群组：

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

- 通过身份提供商在 IAM 中托管的用户：

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

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

要使用控制台为已获允许注册维护时段任务的组配置权限

1. 通过 https://console.aws.amazon.com/iam/ 打开 IAM 控制台。
2. 在导航窗格中，选择用户组。
3. 在组列表中，选择要将 iam:PassRole 权限分配到的组的名称，或在需要时先创建一个新组
4. 在 Permissions（权限）选项卡上，请选择 Add permissions, Create inline policy（添加权限、创建内联策略）。
5. 在策略编辑器区域中，选择 JSON，然后将方框的默认内容替换为以下内容。

   JSON

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

```
            "Resource": "arn:aws:iam::111122223333:role/my-maintenance-
window-role"
        },
        {
            "Effect": "Allow",
            "Action": "iam:ListRoles",
            "Resource": "arn:aws:iam::111122223333:role/"
        },
        {
            "Effect": "Allow",
            "Action": "iam:ListRoles",
            "Resource": "arn:aws:iam::111122223333:role/aws-service-role/
ssm.amazonaws.com/"
        }
    ]
}
```

my-maintenance-window-role 表示您之前创建的自定义维护时段角色的名称。

account-id 表示 AWS 账户 的 ID。为资源 arn:aws:iam::*account-id*:role/ 添加此权限允许用户在创建维护时段任务时在控制台中查看和选择客户角色。为 arn:aws:iam::*account-id*:role/aws-service-role/ssm.amazonaws.com/ 添加此权限允许用户在创建维护时段任务时在控制台中选择 Systems Manager 服务相关角色。

6. 选择下一步。
7. 在查看和创建页面上的策略名称框中，输入名称以标识此 PassRole 策略，例如 **my-group-iam-passrole-policy**，然后选择创建策略。

任务 4：使用控制台阻止指定用户注册维护时段任务

对于您的 AWS 账户中您不希望其将任务注册到维护时段的用户，您可以拒绝向其授予 ssm:RegisterTaskWithMaintenanceWindow 权限。这将提供额外的防护，从而阻止不应注册维护时段任务的用户。

要使用控制台为不允许注册维护时段任务的组配置权限

1. 通过 https://console.aws.amazon.com/iam/ 打开 IAM 控制台。
2. 在导航窗格中，选择用户组。
3. 在组列表中，选择要从中拒绝 ssm:RegisterTaskWithMaintenanceWindow 权限的组的名称，或在需要时先创建一个新组。

4. 在 Permissions（权限）选项卡上，请选择 Add permissions, Create inline policy（添加权限、创建内联策略）。
5. 在策略编辑器区域中，选择 JSON，然后将方框的默认内容替换为以下内容。

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "ssm:RegisterTaskWithMaintenanceWindow",
            "Resource": "*"
        }
    ]
}
```

6. 选择下一步。
7. 在查看和创建页面上，对于策略名称，输入名称以标识此策略，例如 **my-groups-deny-mw-tasks-policy**，然后选择创建策略。

使用 AWS CLI 控制对维护时段的访问权限

以下过程介绍如何使用 AWS Command Line Interface（AWS CLI）为 Maintenance Windows（AWS Systems Manager 中的一项工具）创建所需的角色和权限。

主题

- 任务 1：以 JSON 格式创建信任策略和客户管理型策略
- 任务 2：使用 AWS CLI 为维护时段创建和验证自定义服务角色
- 任务 3：使用 AWS CLI 向注册维护时段任务的特定用户授予权限
- 任务 4：使用 AWS CLI 阻止指定用户注册维护时段任务

任务 1：以 JSON 格式创建信任策略和客户管理型策略

维护时段任务需要一个 IAM 角色才能提供在目标资源上运行所需的权限。通过附加到角色的 IAM 策略提供权限。您运行的任务类型和其他操作要求决定了此策略的内容。我们提供了一个基本策略，您可以根据需要进行调整。根据维护时段运行的任务和任务类型，您可能不需要此策略中的所有权限，并且可能需要包含额外的权限。

在此任务中，您可以在一对 JSON 文件中指定自定义维护时段角色所需的权限。您需要将此策略附加到您稍后在任务 2：使用 AWS CLI 为维护时段创建和验证自定义服务角色中创建的角色。

要创建信任策略和客户管理型策略文件

1. 将以下信任策略复制并粘贴到文本文件中。使用以下名称和文件扩展名保存此文件：**mw-role-trust-policy.json**。

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ssm.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

2. 将以下 JSON 策略复制并粘贴到另一个文本文件中。在您创建第一个文件的同一目录中，使用以下名称和文件扩展名保存此文件：**mw-role-custom-policy.json**。

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:SendCommand",
                "ssm:CancelCommand",
                "ssm:ListCommands",
                "ssm:ListCommandInvocations",
                "ssm:GetCommandInvocation",
                "ssm:GetAutomationExecution",
                "ssm:StartAutomationExecution",
```

```
                "ssm:ListTagsForResource",
                "ssm:GetParameters"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "states:DescribeExecution",
                "states:StartExecution"
            ],
            "Resource": [
                "arn:aws:states:*:*:execution:*:*",
                "arn:aws:states:*:*:stateMachine:*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction"
            ],
            "Resource": [
                "arn:aws:lambda:*:*:function:*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "resource-groups:ListGroups",
                "resource-groups:ListGroupResources"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "tag:GetResources"
            ],
            "Resource": [
                "*"
            ]
        },
```

```
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::111122223333:role/maintenance-window-
role-name",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ssm.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

3. 根据需要为您在账户中运行的维护任务修改 `mw-role-custom-policy.json` 的内容。您做出的更改是特定于所计划操作的。

   例如：

   - 您可以提供特定功能和状态机的 Amazon 资源名称（ARN），而不是使用通配符（*）。
   - 如果您不计划运行 AWS Step Functions 任务，则可以移除 `states` 权限和 ARN。
   - 如果您不计划运行 AWS Lambda 任务，则可以移除 `lambda` 权限和 ARN。
   - 如果您不计划运行自动化任务，则可以移除 `ssm:GetAutomationExecution` 和 `ssm:StartAutomationExecution` 权限。
   - 添加要运行的任务可能需要的其他权限。例如，一些 自动化 操作使用 AWS CloudFormation 堆栈。因此，权限 `cloudformation:CreateStack`、`cloudformation:DescribeStacks` 和 `cloudformation:DeleteStack` 是必需的。

     另一个示例：自动化运行手册 `AWS-CopySnapshot` 需要权限来创建 Amazon Elastic Block Store（Amazon EBS）快照，因此，服务角色需要权限 `ec2:CreateSnapshot`。

     有关自动化运行手册所需角色权限的信息，请参阅 AWS Systems Manager 自动化运行手册参考中的运行手册描述。

   进行任何需要的更改后，请再次保存该文件。

任务 2：使用 AWS CLI 为维护时段创建和验证自定义服务角色

您在上一个任务中创建的策略将附加到您在此任务中创建的维护时段服务角色。当用户注册维护时段任务时，他们需要在任务配置中指定此 IAM 角色。此角色中的权限将允许 Systems Manager 代表您运行维护时段中的任务。

> **Important**
>
> 以前，Systems Manager 控制台允许您选择 AWS 托管式 IAM 服务相关角色 `AWSServiceRoleForAmazonSSM`，以用作任务的维护角色。现在不再建议将此角色及其相关策略 `AmazonSSMServiceRolePolicy` 用于维护时段任务。如果您目前在将此角色用于维护时段任务，我们建议您停止使用它。而应创建您自己的 IAM 角色，以便您的维护时段任务运行时在 Systems Manager 与其他 AWS 服务之间进行通信。

在此任务中，您将运行 CLI 命令来创建维护窗口服务角色，以添加您创建的 JSON 文件中的策略内容。

使用 AWS CLI 为维护时段创建自定义服务角色

1. 打开 AWS CLI 并在放置 `mw-role-custom-policy.json` 和 `mw-role-trust-policy.json` 文件的目录中运行以下命令。该命令创建一个名为 `my-maintenance-window-role` 的维护时段服务角色，并将信任策略附加到该角色上。

   Linux & macOS

   ```
   aws iam create-role \
       --role-name "my-maintenance-window-role" \
       --assume-role-policy-document file://mw-role-trust-policy.json
   ```

   Windows

   ```
   aws iam create-role ^
       --role-name "my-maintenance-window-role" ^
       --assume-role-policy-document file://mw-role-trust-policy.json
   ```

   系统返回类似于以下内容的信息。

   ```
   {
   ```

```
    "Role": {
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Action": "sts:AssumeRole",
                    "Effect": "Allow",
                    "Principal": {
                        "Service": "ssm.amazonaws.com"
                    }
                }
            ]
        },
        "RoleId": "AROAIIZKPBKS2LEXAMPLE",
        "CreateDate": "2024-08-19T03:40:17.373Z",
        "RoleName": "my-maintenance-window-role",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:role/my-maintenance-window-role"
    }
}
```

> **Note**
>
> 记下 RoleName 和 Arn 值。您将在下一命令中包含这些值。

2. 运行以下命令以将客户管理型策略附加到角色。请将 *account-id* 占位符替换为您自己的 AWS 账户 ID

   Linux & macOS

   ```
   aws iam attach-role-policy \
       --role-name "my-maintenance-window-role" \
       --policy-arn "arn:aws:iam::account-id:policy/mw-role-custom-policy.json"
   ```

   Windows

   ```
   aws iam attach-role-policy ^
       --role-name "my-maintenance-window-role" ^
       --policy-arn "arn:aws:iam::account-id:policy/mw-role-custom-policy.json"
   ```

3. 运行以下命令验证您的角色是否已创建，且已附加信任策略。

```
aws iam get-role --role-name my-maintenance-window-role
```

命令返回类似于下文的信息：

```
{
    "Role": {
        "Path": "/",
        "RoleName": "my-maintenance-window-role",
        "RoleId": "AROA123456789EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:role/my-maintenance-window-role",
        "CreateDate": "2024-08-19T14:13:32+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {
                        "Service": "ssm.amazonaws.com"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        },
        "MaxSessionDuration": 3600,
        "RoleLastUsed": {
            "LastUsedDate": "2024-08-19T14:30:44+00:00",
            "Region": "us-east-2"
        }
    }
}
```

4. 运行以下命令以验证客户管理型策略是否已附加到角色。

```
aws iam list-attached-role-policies --role-name my-maintenance-window-role
```

命令返回类似于下文的信息：

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "mw-role-custom-policy",
```

```
            "PolicyArn": "arn:aws:iam::123456789012:policy/mw-role-custom-policy"
        }
    ]
}
```

任务 3：使用 AWS CLI 向注册维护时段任务的特定用户授予权限

为用户提供访问自定义维护时段服务角色的权限，使他们可以将该角色用于维护时段任务。这是对已授予权限的补充，以便与适用于 Maintenance Windows 工具的 Systems Manager API 命令结合使用。此 IAM 角色传递了运行维护时段任务所需的权限。因此，如果无法传递这些 IAM 权限，用户将无法使用您的自定义服务角色向维护时段注册任务。

在将任务注册到维护时段时，您可以指定服务角色来运行实际任务操作。这是代表您运行任务时服务要代入的角色。在此之前，要注册任务本身，请将 IAM `PassRole` 策略分配给 IAM 实体（如用户或组）。这将允许 IAM 实体指定运行任务时应使用的角色，作为将这些任务注册到维护时段的一部分。有关更多信息，请参阅《IAM 用户指南》中的向用户授予将角色传递给 AWS 服务 的权限。

使用 AWS CLI 为已获允许注册维护时段任务的用户配置权限

1. 将以下 AWS Identity and Access Management IAM 策略复制并粘贴到文本编辑器中，然后使用以下名称和文件扩展名保存：`mw-passrole-policy.json`。

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::111122223333:role/my-maintenance-
window-role"
        },
        {
            "Effect": "Allow",
            "Action": "iam:ListRoles",
            "Resource": "arn:aws:iam::111122223333:role/"
        },
        {
            "Effect": "Allow",
```

```
            "Action": "iam:ListRoles",
            "Resource": "arn:aws:iam::111122223333:role/aws-service-role/
ssm.amazonaws.com/"
        }
    ]
}
```

将 *my-maintenance-window-role* 替换为您之前创建的自定义维护时段角色的名称。

将 *account-id* 替换为 AWS 账户 的 ID。为资源 arn:aws:iam::*account-id*:role/ 添加此权限允许组中的用户在创建维护时段任务时在控制台中查看和选择客户角色。为 arn:aws:iam::*account-id*:role/aws-service-role/ssm.amazonaws.com/ 添加此权限允许组中的用户在创建维护时段任务时在控制台中选择 Systems Manager 服务相关角色。

2. 打开 AWS CLI。
3. 根据您是否要将权限分配到 IAM 实体（用户或组），运行以下命令之一。

   - 对于 IAM 实体：

     Linux & macOS

     ```
     aws iam put-user-policy \
         --user-name "user-name" \
         --policy-name "policy-name" \
         --policy-document file://path-to-document
     ```

     Windows

     ```
     aws iam put-user-policy ^
         --user-name "user-name" ^
         --policy-name "policy-name" ^
         --policy-document file://path-to-document
     ```

     对于 *user-name*，请指定将向维护时段分配任务的用户。对于 *policy-name*，请指定要用于标识策略的名称，如 **my-iam-passrole-policy**。对于 *path-to-document*，请指定步骤 1 中保存文件的路径。例如：file://C:\Temp\mw-passrole-policy.json

> **Note**
>
> 要授予用户使用 Systems Manager 控制台为维护时段注册任务的访问权限，您还必须将 AmazonSSMFullAccess 策略分配给您的用户（或为涵盖维护时段任务的 Systems Manager 提供较小访问权限集的 IAM 策略）。运行以下命令将 AmazonSSMFullAccess 策略分配给您的用户。
>
> Linux & macOS
>
> ```
> aws iam attach-user-policy \
>     --policy-arn "arn:aws:iam::aws:policy/AmazonSSMFullAccess" \
>     --user-name "user-name"
> ```
>
> Windows
>
> ```
> aws iam attach-user-policy ^
>     --policy-arn "arn:aws:iam::aws:policy/AmazonSSMFullAccess" ^
>     --user-name "user-name"
> ```

- 对于 IAM 组：

  Linux & macOS

  ```
  aws iam put-group-policy \
      --group-name "group-name" \
      --policy-name "policy-name" \
      --policy-document file://path-to-document
  ```

  Windows

  ```
  aws iam put-group-policy ^
      --group-name "group-name" ^
      --policy-name "policy-name" ^
      --policy-document file://path-to-document
  ```

  对于 *group-name*，指定其成员将向维护时段分配任务的组。对于 *policy-name*，请指定要用于标识策略的名称，如 **my-iam-passrole-policy**。对于 *path-to-document*，请指定步骤 1 中保存文件的路径。例如：file://C:\Temp\mw-passrole-policy.json。

> **Note**
>
> 要授予组成员使用 Systems Manager 控制台为维护时段注册任务的访问权限，还必须将 AmazonSSMFullAccess 策略分配给组。运行以下命令将此策略分配给您的组。
>
> Linux & macOS
>
> ```
> aws iam attach-group-policy \
>     --policy-arn "arn:aws:iam::aws:policy/AmazonSSMFullAccess" \
>     --group-name "group-name"
> ```
>
> Windows
>
> ```
> aws iam attach-group-policy ^
>     --policy-arn "arn:aws:iam::aws:policy/AmazonSSMFullAccess" ^
>     --group-name "group-name"
> ```

4. 运行以下命令验证策略是否已分配给该组。

   Linux & macOS

   ```
   aws iam list-group-policies \
       --group-name "group-name"
   ```

   Windows

   ```
   aws iam list-group-policies ^
       --group-name "group-name"
   ```

任务 4：使用 AWS CLI 阻止指定用户注册维护时段任务

对于您的 AWS 账户中您不希望其将任务注册到维护时段的用户，您可以拒绝向其授予 ssm:RegisterTaskWithMaintenanceWindow 权限。这将提供额外的防护，从而阻止不应注册维护时段任务的用户。

根据您是拒绝向单个用户还是组授予 ssm:RegisterTaskWithMaintenanceWindow 权限，使用以下过程之一，阻止用户将任务注册到维护时段。

要使用 AWS CLI 为不允许注册维护时段任务的用户配置权限

1. 将以下 IAM 策略复制并粘贴到文本编辑器中，然后使用以下名称和文件扩展名保存：**deny-mw-tasks-policy.json**。

   JSON

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Deny",
               "Action": "ssm:RegisterTaskWithMaintenanceWindow",
               "Resource": "*"
           }
       ]
   }
   ```

2. 打开 AWS CLI。
3. 根据您是否要将权限分配到 IAM 实体（用户或组），运行以下命令之一。

   - 对于用户：

     Linux & macOS

     ```
     aws iam put-user-policy \
         --user-name "user-name" \
         --policy-name "policy-name" \
         --policy-document file://path-to-document
     ```

     Windows

     ```
     aws iam put-user-policy ^
         --user-name "user-name" ^
         --policy-name "policy-name" ^
         --policy-document file://path-to-document
     ```

     对于 *user-name*，请指定要阻止其向维护时段分配任务的用户。对于 *policy-name*，请指定要用于标识策略的名称，如 **my-deny-mw-tasks-policy**。对于 *path-to-document*，请指定步骤 1 中保存文件的路径。例如：`file://C:\Temp\deny-mw-tasks-policy.json`

- 对于组：

  Linux & macOS

  ```
  aws iam put-group-policy \
      --group-name "group-name" \
      --policy-name "policy-name" \
      --policy-document file://path-to-document
  ```

  Windows

  ```
  aws iam put-group-policy ^
      --group-name "group-name" ^
      --policy-name "policy-name" ^
      --policy-document file://path-to-document
  ```

  对于 *group-name*，请指定要阻止其成员向维护时段分配任务的组。对于 *policy-name*，请指定要用于标识策略的名称，如 **my-deny-mw-tasks-policy**。对于 *path-to-document*，请指定步骤 1 中保存文件的路径。例如：`file://C:\Temp\deny-mw-tasks-policy.json`

4. 运行以下命令验证策略是否已分配给该组。

   Linux & macOS

   ```
   aws iam list-group-policies \
       --group-name "group-name"
   ```

   Windows

   ```
   aws iam list-group-policies ^
       --group-name "group-name"
   ```

使用控制台创建和管理维护时段

本节介绍如何使用 AWS Systems Manager 控制台创建、配置、更新和删除维护时段。此外，本节还提供有关管理维护时段的目标和任务的信息。

> **Important**
>
> 我们建议您一开始在测试环境中创建和配置维护时段。

开始前的准备工作

在创建维护时段之前，必须先配置对 Maintenance Windows（AWS Systems Manager 中的一项工具）的访问权限。有关更多信息，请参阅 设置 Maintenance Windows。

主题

- 使用控制台创建维护时段
- 使用控制台为维护时段分配目标
- 使用控制台为维护时段分配任务
- 使用控制台禁用或启用维护时段
- 使用控制台更新或删除维护时段资源

使用控制台创建维护时段

在此过程中，您需要在 Maintenance Windows（AWS Systems Manager 中的一项工具）中创建维护时段。您可以指定其基本选项，例如名称、计划和持续时间。在后面的步骤中，您将选择它更新的目标或资源，以及在维护时段运行时运行的任务。

> **Note**
>
> 有关维护时段的各种计划相关选项如何相互关联的说明，请参阅 维护时段计划和活动期间选项。
> 有关使用 `--schedule` 选项的更多信息，请参阅 参考：适用于 Systems Manager 的 Cron 和 Rate 表达式。

要使用控制台创建维护时段

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Maintenance Windows。

3. 选择 Create maintenance window。
4. 对于 Name (名称)，请输入描述性名称，以帮助您标识此维护时段。
5. （可选）对于 Description（说明），输入说明以确定将如何使用此维护时段。
6. （可选）如果您希望即便没有将托管式节点注册为目标，也允许维护时段任务在这些节点上运行，则选择 Allow unregistered targets（允许未注册的目标）。

   如果选择了此选项，您在将任务注册到维护时段时便可以选择已注销节点（按节点 ID）。

   如果未选择此选项，则在将任务注册到维护时段时就必须选择之前注册的目标。
7. 使用三个计划选项之一为维护时段指定计划。

   有关构建 cron/rate 表达式的信息，请参阅 参考：适用于 Systems Manager 的 Cron 和 Rate 表达式。
8. 对于 Duration (持续时间)，输入维护时段将运行的小时数。您指定的值根据维护时段的开始时间确定维护时段的具体结束时间。在最终结束时间减去您在下一步中为 Stop initiating tasks (停止启动任务) 指定的小时数后，不允许启动维护时段任务。

   例如，如果维护时段从下午 3 点开始，持续时间是 3 个小时，并且 Stop initiating tasks (停止启动任务) 值是 1 个小时，则下午 5 点之后将无法启动维护时段任务。
9. 在停止启动任务中，输入系统应该在维护时段结束前几小时停止计划要运行的新任务。
10. （可选）对于 Window start date（时段开始日期），请以 ISO-8601 扩展格式指定您希望维护时段变为活动状态的日期和时间。这让您可以将维护时段的激活时间推迟到指定的将来日期。

    > **Note**
    >
    > 您无法指定过去发生的开始日期和时间。

11. （可选）对于 Window end date（时段结束日期），请以 ISO-8601 扩展格式指定您希望维护时段变为不活动状态的日期和时间。这样可以设置在某个未来的日期和时间后不再运行维护时段。
12. （可选）对于 Schedule timezone（计划时区），请以互联网编号分配机构 (IANA) 格式指定时区，用作计划的维护时段运行时的依据。例如：“America/Los_Angeles”、“etc/UTC”或“Asia/Seoul”。

    有关有效格式的更多信息，请参阅 IANA 网站上的 Time Zone Database。
13. （可选）对于 Schedule offset（计划偏移），输入在运行维护时段之前但在 cron 或 rate 表达式指定的日期和时间之后等待的天数。您可以指定一到六天之间。

> **Note**
>
> 仅当您通过手动输入 cron 或 rate 表达式来指定计划时，此选项才可用。

14. （可选）在管理标签区域，将一个或多个标签键名称/值对应用到维护时段。

    标签是您分配给资源的可选元数据。标签运行您按各种标准（如用途、所有者或环境）对资源进行分类。例如，您可能需要标记维护时段来标识它所运行的任务的类型、目标类型和它所运行的环境。在这种情况下，您可以指定以下键名/键值对：

    - `Key=TaskType,Value=AgentUpdate`
    - `Key=OS,Value=Windows`
    - `Key=Environment,Value=Production`

15. 选择 Create maintenance window。系统将返回维护时段页面。您刚刚创建的维护时段的状态是已启用。

使用控制台为维护时段分配目标

在本过程中，您将向维护时段注册目标。换言之，您指定维护时段对其执行操作的资源。

> **Note**
>
> 如果将单个维护时段任务注册到多个目标，则其任务调用将按顺序进行，而不是并行发生。如果您的任务必须同时在多个目标上运行，请为每个目标分别注册一个任务，并为每个任务分配相同的优先级。

要使用控制台为维护时段分配目标

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Maintenance Windows。
3. 在维护时段列表中，选择您要向其中添加目标的维护时段。
4. 选择 Actions，然后选择 Register targets。
5. （可选）对于 Target Name (目标名称)，输入目标的名称。
6. （可选）对于描述，输入描述。

7. （可选）对于 Owner information (所有者信息)，请指定在此维护时段内对这些目标运行任务时引发的任何 Amazon EventBridge 事件中要包含的信息。

   有关使用 EventBridge 监控 Systems Manager 事件的信息，请参阅 使用 Amazon EventBridge 监控 Systems Manager 事件。

8. 在目标区域中，选择下表中所述的选项之一。

选项	描述
指定实例标签	对于 Specify instance tags（指定实例标签）框，请指定已经或将要添加到您账户中的托管式节点的一个或多个标签键和（可选）值。当维护时段运行时，它会尝试在这些标签添加到的所有托管式节点上执行任务。 如果您指定了多个标签键，则必须使用您指定的所有标签键和值来标记节点，才能将该节点包含在目标组中。
手动选择实例	从列表中，选中要包含在维护时段目标中的每个节点所对应的框。 该列表包括您的账户中配置为用于 Systems Manager 的所有节点。 如果未列出您希望看到的托管式节点，请参阅排除托管式节点可用性的问题 以获取故障排除技巧。 有关边缘设备和本地服务器和虚拟机 (VM)，请参阅 使用 Systems Manager 管理混合和多云环境中的节点
选择资源组	对于资源组，从列表中选择您的账户中现有资源组的名称。 有关创建和使用资源组的信息，请参阅以下主题：

选项	描述
	• AWS Resource Groups 用户指南中的什么是资源组？ • AWS 新闻博客中的 AWS Resource Groups 和标记 （可选）对于 Resource types（资源类型），最多可选择五种可用资源类型，或选择 All resource types（所有资源类型）。 如果您分配给维护时段的任务不对您添加到目标的其中一种资源类型执行操作，系统可能会报告错误。尽管存在这些错误，但找到了支持的资源类型的任务仍会继续运行。 例如，假设您将以下资源类型添加到此目标： • `AWS::S3::Bucket` • `AWS::DynamoDB::Table` • `AWS::EC2::Instance` 但后来，当您将任务添加到维护时段时，您只包括对节点执行操作的任务，例如应用补丁基准或重启节点。在维护时段日志中，可能会报告未找到 Amazon Simple Storage Service (Amazon S3) 存储桶或 Amazon DynamoDB 表的错误。但是，维护时段仍会在资源组中的节点上运行任务。

9. 选择注册目标。

如果您想将更多目标分配到此维护时段，请选择目标选项卡，然后选择注册目标。利用此选项，您可以选择不同的目标设定方式。例如，如果您之前按节点 ID 将节点设为目标，则可以注册新目标并通过指定应用于托管式节点的标签或从资源组中选择资源类型来将节点设为目标。

使用控制台为维护时段分配任务

在此过程中，您向维护时段添加任务。任务是在维护时段运行时执行的操作。

以下四种类型的任务可以添加到维护时段：

- AWS Systems Manager Run Command 命令
- Systems Manager 自动化 工作流
- AWS Step Functions 任务
- AWS Lambda 函数

> **Important**
>
> Maintenance Windows 的 IAM policy 要求您将前缀 SSM 添加到 Lambda 函数（或别名）名称。继续注册此类型的任务之前，请在 AWS Lambda 中更新其名称，以包含 SSM。例如，如果您的 Lambda 函数名称为 MyLambdaFunction，请将其更改为 SSMMyLambdaFunction。

为维护时段分配任务

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Maintenance Windows。
3. 在维护时段列表中，选择一个维护时段。
4. 选择 Actions (操作)，然后选择您要注册到维护时段的任务的类型对应的选项。
   - 选择注册 Run command 任务。
   - 选择注册 自动化 任务
   - 注册 Lambda 任务
   - 注册 Step Functions 任务

   > **Note**
   >
   > 维护时段任务仅支持 Step Functions 标准状态机工作流。它们不支持快速状态机工作流。有关状态机工作流类型的信息，请参阅《AWS Step Functions 开发人员指南》中的标准与快速工作流。

5. （可选）对于 Name (名称)，请输入任务的名称。
6. （可选）对于描述，输入描述。
7. 对于 New task invocation cutoff（新任务调用截止），如果您不希望在到达维护时段截止时间后开始任何新任务调用，请选择 Enabled（已启用）。

   如果未启用此选项，任务会在到达截止时间后继续运行，并会开始新的任务调用直至完成。

   > **Note**
   >
   > 启用此选项后，未完成任务的状态为 `TIMED_OUT`。

8. 在此步骤中，请按照所选任务类型的子步骤进行操作。

   Run Command

   1. 在命令文档列表中，选择定义待运行任务的 Systems Manager 命令文档（SSM 文档）。
   2. 对于 Document Version (文档版本)，选择要使用的文档版本。
   3. 对于 Task priority (任务优先级)，指定此任务的优先级。零 (0) 表示最高优先级。维护时段中的任务按优先级顺序计划，具有相同优先级的任务则并行计划。

   Automation

   1. 在自动化文档列表中，选择定义待运行任务的 Automation 运行手册。
   2. 对于文档版本，选择要使用的运行手册版本。
   3. 对于 Task priority (任务优先级)，指定此任务的优先级。零 (0) 表示最高优先级。维护时段中的任务按优先级顺序计划，具有相同优先级的任务则并行计划。

   Lambda

   1. 在 Lambda 参数区域中，从列表中选择 Lambda 函数。
   2. （可选）提供您想要包括的 Payload（有效负载）、Client Context（客户端上下文）或 Qualifier（限定词）的任何内容。

> **Note**
>
> 在某些情况下，您可以使用伪参数作为 `Payload` 值的一部分。待维护时段任务运行时，该函数会传递正确的值而不是伪参数占位符。有关信息，请参阅注册维护时段任务时使用伪参数。

3. 对于 Task priority (任务优先级)，指定此任务的优先级。零 (0) 表示最高优先级。维护时段中的任务按优先级顺序计划，具有相同优先级的任务则并行计划。

Step Functions

1. 在 Step Functions 参数区域中，从列表中选择状态机。
2. （可选）提供状态机执行的名称以及您想要包括的 Input（输入）的任何内容。

> **Note**
>
> 在某些情况下，您可以使用伪参数作为 `Input` 值的一部分。待维护时段任务运行时，该函数会传递正确的值而不是伪参数占位符。有关信息，请参阅注册维护时段任务时使用伪参数。

3. 对于 Task priority (任务优先级)，指定此任务的优先级。零 (0) 表示最高优先级。维护时段中的任务按优先级顺序计划，具有相同优先级的任务则并行计划。

9. 在 Targets (目标) 区域中，选择以下选项之一：

- Selecting registered target groups (选择已注册的目标群)：选择已注册到当前维护时段的一个或多个维护时段目标。
- Selecting unregistered targets (选择未注册的目标)：逐个选择可用资源作为任务的目标。

  如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。
- Task target not required (不需要任务目标)：可能已在其他函数中为除 Run Command 类型以外的所有其他任务指定了任务的目标。

  为维护时段 Run Command 类型任务指定一个或多个目标。根据任务的不同，目标对于其他维护时段任务类型（自动化、AWS Lambda 和 AWS Step Functions）是可选的。有关运行未指定目标的任务的更多信息，请参阅 注册不含目标的维护时段任务。

> **Note**
>
> 在许多情况下，您无需为自动化任务明确指定目标。例如，假设您要创建 自动化 类型的任务，以使用 `AWS-UpdateLinuxAmi` 运行手册为 Linux 更新 Amazon Machine Image (AMI)。在该任务运行时，已使用最新可用的 Linux 分发版本的程序包和 Amazon 软件更新了 AMI。从 AMI 创建的新实例已经安装了这些更新。由于要更新的 AMI 的 ID 是在运行手册的输入参数中指定的，因此无需在维护时段任务中再次指定目标。

10. 仅限 Automation 任务：

    在 Input parameters（输入参数）区域中，为运行任务所需的任何必需或可选的参数提供值。

> **Note**
>
> 在某些情况下，您可以为某些输入参数值使用伪参数。待维护时段任务运行时，该函数会传递正确的值而不是伪参数占位符。有关信息，请参阅注册维护时段任务时使用伪参数。

11. 对于 Rate control（速率控制）：

    - 对于 Concurrency（并发），请指定要同时运行该命令的托管式节点的数量或百分比。

> **Note**
>
> 如果您通过指定应用于托管式节点的标签或指定 AWS Resource Groups 来选择目标，但不确定有多少个托管式节点已被设为目标，则可通过指定百分比来限制可同时运行该文档的目标的数量。

    - 对于 Error threshold（错误阈值），请指定当命令在一定数量或百分比的节点上失败后，何时在其他托管式节点上停止运行该命令。例如，如果您指定三个错误，Systems Manager 将在收到第四个错误时停止发送该命令。仍在处理该命令的托管式节点也可能发送错误。

12. （可选）对于 IAM 服务角色，选择一个角色以向 Systems Manager 提供运行维护时段任务时所承担的权限。

    如果您未指定服务角色 ARN，Systems Manager 将使用您账户中的服务相关角色。下拉菜单中未列出此角色。如果您的账户中没有适用于 Systems Manager 的适当服务相关角色，则将在成功注册任务后创建该角色。

> **Note**
>
> 为了改善安全状况，强烈建议您创建自定义策略和自定义服务角色来运行维护时段任务。可以精心设计该策略，只提供特定维护时段任务所需的权限。有关更多信息，请参阅 设置 Maintenance Windows。

13. 仅限 Run Command 任务：

    （可选）对于 Output options（输出选项），请执行以下操作：

    - 选中 Enable writing to S3 (启用写入到 S3) 复选框，将命令输出保存到文件。在输入框中输入存储桶和前缀（文件夹）名称。
    - 选中 CloudWatch output (CloudWatch 输出) 复选框，将完整的输出写入到 Amazon CloudWatch Logs。输入 CloudWatch Logs 日志组的名称。

    > **Note**
    >
    > 授予将数据写入 S3 存储桶或 CloudWatch Logs 的能力的权限，是分配给节点的实例配置文件的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限。此外，如果指定的 S3 存储桶或日志组位于不同的 AWS 账户 中，请确认与该节点关联的实例配置文件具有写入该存储桶的所需权限。

14. 仅限 Run Command 任务：

    在 SNS 通知部分，如果需要发送有关命令执行状态的通知，请选中 Enable SNS notifications（启用 SNS 通知）复选框。

    有关为 Run Command 配置 Amazon SNS 通知的更多信息，请参阅 使用 Amazon SNS 通知监控 Systems Manager 状态更改。

15. 仅限 Run Command 任务：

    在 Parameters (参数) 部分中，为文档指定参数。

    > **Note**
    >
    > 在某些情况下，您可以为某些输入参数值使用伪参数。待维护时段任务运行时，该函数会传递正确的值而不是伪参数占位符。有关信息，请参阅注册维护时段任务时使用伪参数。

16. 仅限 Run Command 和 Automation 任务：

    （可选）在 CloudWatch 警报区域中，为警报名称选择现有的 CloudWatch 警报来应用到任务，以便进行监控。

    如果警报激活，任务将停止。

    > **Note**
    >
    > 要将 CloudWatch 警报附加到任务，运行任务的 IAM 主体必须具有 `iam:createServiceLinkedRole` 操作的权限。有关 CloudWatch 警报的更多信息，请参阅使用 Amazon CloudWatch 警报。

17. 根据您的任务类型，选择以下选项之一：

    - 选择注册 Run command 任务。
    - 选择注册 自动化 任务
    - 注册 Lambda 任务
    - 注册 Step Functions 任务

使用控制台禁用或启用维护时段

您可以在 Maintenance Windows（ AWS Systems Manager 中的一项工具）中禁用或启用维护时段。您可以一次选择一个维护时段，以禁用或启用维护时段的运行。您也可以选择多个或所有维护时段来启用和禁用。

本节介绍如何使用 Systems Manager 控制台禁用或启用维护时段。有关如何使用 AWS Command Line Interface (AWS CLI) 执行此操作的示例，请参阅 教程：使用 AWS CLI 更新维护时段。

主题

- 使用控制台禁用维护时段
- 使用控制台启用维护时段

使用控制台禁用维护时段

您可以禁用维护时段以便在指定时间内暂停任务，并且该维护时段将保持可用状态，稍后可以再次启用。

禁用维护时段

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Maintenance Windows。
3. 使用要禁用的维护时段旁边的复选框，选择一个或多个维护时段。
4. 在操作菜单上，选择禁用维护时段。系统将提示您确认操作。

使用控制台启用维护时段

您可以启用维护时段来恢复任务。

> **Note**
>
> 如果维护时段使用 Rate 计划，并且起始日期当前设置为过去的日期和时间，则使用当前日期和时间作为维护时段的起始日期。您可以在启用维护时段之前或之后更改其起始日期。有关信息，请参阅使用控制台更新或删除维护时段资源。

启用维护时段

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Maintenance Windows。
3. 选中维护时段旁边的复选框即可启用。
4. 选择操作，选择启用维护时段。系统将提示您确认操作。

使用控制台更新或删除维护时段资源

您可以在 Maintenance Windows（AWS Systems Manager 中的一项工具）中更新或删除维护时段。也可以更新或删除维护时段的目标或任务。如果要编辑维护时段的详细信息，您可以更改计划、目标和任务。您还可以指定时段、目标和任务的名称和描述，从而可以更好地理解它们的用途和更轻松地管理您的时段队列。

本节介绍如何使用 Systems Manager 控制台更新或删除维护时段、目标和任务。有关如何使用 AWS Command Line Interface (AWS CLI) 执行此操作的示例，请参阅 教程：使用 AWS CLI 更新维护时段。

主题

- 使用控制台更新或删除维护时段
- 使用控制台更新或注销维护时段目标
- 使用控制台更新或注销维护时段任务

使用控制台更新或删除维护时段

您可以更新维护时段以更改其名称、描述和计划，以及该维护时段是否应允许已注销目标。

更新或删除维护时段

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Maintenance Windows。
3. 选择要更新或删除的维护时段旁边的按钮，然后执行以下操作之一：
   - 选择删除。系统将提示您确认操作。
   - 选择编辑。在 Edit maintenance window（编辑维护时段）页面中，更改所需的值和选项，然后选择 Save changes（保存更改）。

     有关您可以进行的配置选择的信息，请参阅 使用控制台创建维护时段。

使用控制台更新或注销维护时段目标

您可以更新或注销维护时段的目标。如果您选择更新维护时段目标，则可以指定新的目标名称、描述和所有者。您也可以选择不同的目标。

更新或删除维护时段目标

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Maintenance Windows。
3. 请选择要更新的维护时段的名称，选择 Targets（目标）选项卡，然后执行以下操作之一：
   - 要更新目标，请选择要更新的目标旁边的按钮，然后选择 Edit（编辑）。

- 要取消注册目标，请选择要注销的目标旁边的按钮，然后选择 Deregister target（注销目标）。在 Deregister maintenance windows target (注销维护时段目标) 对话框中，选择 Deregister (注销)。

使用控制台更新或注销维护时段任务

您可以更新或注销维护时段的任务。如果您选择更新，则可以指定新的任务名称、描述和所有者。对于 Run Command 和 自动化 任务，可以为任务选择其他 SSM 文档。但是，您无法编辑任务以更改其类型。例如，如果您创建了一个 自动化 任务，则无法编辑该任务并将其更改为 Run Command 任务。

要使用控制台更新或删除维护时段的任务

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Maintenance Windows。
3. 请选择要更新的维护时段的名称。
4. 请选择 Tasks（任务）选项卡，然后选择要更新的任务旁边的按钮。
5. 请执行以下操作之一：

   - 要注销任务，请选择 Deregister task（注销任务）。
   - 要编辑任务，请选择 Edit (编辑)。更改所需的值和选项，然后选择 Edit task（编辑任务）。

教程

本部分中的教程说明如何在使用维护时段时执行常见任务。

满足先决条件

在尝试这些教程介绍的内容之前，请满足以下先决条件：

- 在本地计算机上配置 AWS CLI - 您必须先在本地计算机上安装和配置 CLI，然后才能运行 AWS CLI 命令。有关信息，请参阅安装或更新 AWS CLI 的最新版本以及安装 AWS Tools for PowerShell。
- 验证维护时段角色和权限 - 您账户中的 AWS 管理员必须授予您使用 CLI 管理维护时段所需的 AWS Identity and Access Management（IAM）权限。有关信息，请参阅 设置 Maintenance Windows。
- 创建或配置与 Systems Manager 兼容的实例 - 您需要至少一个配置为与 Systems Manager 配合使用的 Amazon Elastic Compute Cloud（Amazon EC2）实例，以完成教程。这意味着 SSM Agent 已安装在该实例上，并且该实例上附加了适用于 Systems Manager 的 IAM 实例配置文件。

我们建议从一个预安装了代理的 AWS 托管式 Amazon Machine Image（AMI）中启动实例。有关更多信息，请参阅 查找预装了 SSM Agent 的 AMIs。

有关在实例上安装 SSM Agent 的更多信息，请参阅以下主题：

- 在适用于 Windows Server 的 EC2 实例上手动安装和卸载 SSM Agent
- 在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent

有关为 Systems Manager 配置实例的 IAM 权限的信息，请参阅配置 Systems Manager 所需的实例权限。

- 根据需要创建其他资源：Run Command（Systems Manager 中的一项工具）包含很多任务，除了本先决条件主题中列出的资源，这些任务不需要您创建其他资源。因此，我们提供了一个简单的 Run Command 任务，供您在首次学习教程时使用。您还需要一个配置为与 Systems Manager 配合使用的 EC2 实例，如本主题前面所述。配置该实例之后，您可以注册简单 Run Command 任务。

  Systems Manager Maintenance Windows 工具支持运行以下四种类型的任务：

  - Run Command 命令
  - Systems Manager 自动化 工作流
  - AWS Lambda 函数
  - AWS Step Functions 任务

  通常而言，如果您要运行的维护时段任务需要额外的资源，您应该先创建这些资源。例如，如果您需要一个运行 AWS Lambda 函数的维护时段，则在开始之前，请创建 Lambda 函数；对于 Run Command 任务，创建您可将命令输出保存到其中的 S3 存储桶（如果您计划这样做）；等等。

教程

- 教程：使用 AWS CLI 创建和管理维护时段
- 教程：使用控制台创建修补的维护时段

教程：使用 AWS CLI 创建和管理维护时段

此部分包含的教程可帮助您了解如何使用 AWS Command Line Interface (AWS CLI) 来执行以下操作：

- 创建和配置维护时段
- 查看有关维护时段的信息

- 查看有关维护时段任务和任务执行的信息
- 更新维护时段
- 删除维护时段

跟踪资源 ID

在您完成这些 AWS CLI 教程中的任务之后，跟踪您运行的命令生成的资源 ID。您可以将其用作后续命令的输入。例如，当您创建维护时段时，系统会向您提供以下格式的维护时段 ID。

```
{
   "WindowId":"mw-0c50858d01EXAMPLE"
}
```

记下以下系统生成的 ID，因为本节中的教程会使用它们：

- `WindowId`
- `WindowTargetId`
- `WindowTaskId`
- `WindowExecutionId`
- `TaskExecutionId`
- `InvocationId`
- `ExecutionId`

您还需要计划在这些教程中使用的 EC2 实例的 ID。例如：`i-02573cafcfEXAMPLE`

教程

- 教程：使用 AWS CLI 创建和配置维护时段
- 教程：使用 AWS CLI 查看有关维护时段的信息
- 教程：查看有关任务和使用 AWS CLI 执行任务的信息
- 教程：使用 AWS CLI 更新维护时段
- 教程：使用 AWS CLI 删除维护时段

教程：使用 AWS CLI 创建和配置维护时段

本教程演示如何使用 AWS Command Line Interface (AWS CLI) 创建和配置维护时段、其目标和任务。完成教程的主要路径由几个简单步骤组成。您可以创建单个维护时段，确定单个目标，并为要运行的维护时段建立一个简单任务。在此过程中，我们提供了您可用于尝试更为复杂场景的信息。

在按照本教程中的步骤操作时，请将以斜体*##*文本显示的值替换为您自己的选项和 ID。例如，请将维护时段 ID *mw-0c50858d01EXAMPLE* 和实例 ID *i-02573cafcfEXAMPLE* 替换为您创建的资源的 ID。

内容

- 步骤 1：使用 AWS CLI 创建维护时段
- 步骤 2：使用 AWS CLI 将目标节点注册到维护时段
- 步骤 3：使用 AWS CLI 向维护时段注册任务

步骤 1：使用 AWS CLI 创建维护时段

在此步骤中，您将创建维护时段并指定其基本选项，如名称、计划和持续时间。在后面的步骤中，您选择它将更新的实例和运行的任务。

在我们的示例中，您将创建一个每 5 分钟运行的维护时段。通常您不会如此频繁地运行维护时段。不过，此速率可让您快速看到教程的结果。在任务成功运行之后，我们将向您展示如何改为较低频率的速率。

> **Note**
>
> 有关维护时段的各种计划相关选项如何相互关联的说明，请参阅 维护时段计划和活动期间选项。
>
> 有关使用 `--schedule` 选项的更多信息，请参阅 参考：适用于 Systems Manager 的 Cron 和 Rate 表达式。

要使用 AWS CLI 创建维护时段

1. 在您的本地计算机上打开 AWS Command Line Interface (AWS CLI) 并运行以下命令，创建执行以下任务的维护时段：

   - 每五分钟运行一次，持续时间最长两小时（根据需要）。
   - 在维护时段运行结束后的一小时内，阻止新任务启动。

- 允许无关联的目标（您未注册到维护时段的实例）。
- 通过使用自定义标签指示其创建方计划在本教程中使用它。

Linux & macOS

```
aws ssm create-maintenance-window \
    --name "My-First-Maintenance-Window" \
    --schedule "rate(5 minutes)" \
    --duration 2 \
    --cutoff 1 \
    --allow-unassociated-targets \
    --tags "Key=Purpose,Value=Tutorial"
```

Windows

```
aws ssm create-maintenance-window ^
    --name "My-First-Maintenance-Window" ^
    --schedule "rate(5 minutes)" ^
    --duration 2 ^
    --cutoff 1 ^
    --allow-unassociated-targets ^
    --tags "Key"="Purpose","Value"="Tutorial"
```

系统返回类似于以下内容的信息。

```
{
   "WindowId":"mw-0c50858d01EXAMPLE"
}
```

2. 现在，运行以下命令来查看有关此维护时段以及您的账户中已有的任何其他维护时段的详细信息。

```
aws ssm describe-maintenance-windows
```

系统返回类似于以下内容的信息。

```
{
   "WindowIdentities":[
      {
            "WindowId": "mw-0c50858d01EXAMPLE",
```

```
            "Name": "My-First-Maintenance-Window",
            "Enabled": true,
            "Duration": 2,
            "Cutoff": 1,
            "NextExecutionTime": "2019-05-11T16:46:16.991Z"
        }
    ]
}
```

继续步骤 2：使用 AWS CLI 将目标节点注册到维护时段。

步骤 2：使用 AWS CLI 将目标节点注册到维护时段

在此步骤中，您使用新的维护时段注册目标。在这种情况下，您需指定维护时段运行时要更新的节点。

有关使用节点 ID 一次注册多个节点的示例、使用标签来标识多个节点的示例，以及将资源组指定为目标的示例，请参阅 示例：向维护时段注册目标。

> **Note**
>
> 您应该已经创建了要在此步骤中使用的 Amazon Elastic Compute Cloud (Amazon EC2) 实例，如 Maintenance Windows 教程先决条件中所述。

要使用 AWS CLI 将目标节点注册到维护时段

1. 在本地计算机上运行以下命令。将每个*#######*替换为您自己的信息。

   Linux & macOS

   ```
   aws ssm register-target-with-maintenance-window \
       --window-id "mw-0c50858d01EXAMPLE" \
       --resource-type "INSTANCE" \
       --target "Key=InstanceIds,Values=i-02573cafcfEXAMPLE"
   ```

   Windows

   ```
   aws ssm register-target-with-maintenance-window ^
       --window-id "mw-0c50858d01EXAMPLE" ^
       --resource-type "INSTANCE" ^
   ```

```
    --target "Key=InstanceIds,Values=i-02573cafcfEXAMPLE"
```

系统返回类似于以下内容的信息。

```
{
   "WindowTargetId":"e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
}
```

2. 现在，在本地计算机上运行以下命令，查看有关您的维护时段目标的详细信息。

Linux & macOS

```
aws ssm describe-maintenance-window-targets \
    --window-id "mw-0c50858d01EXAMPLE"
```

Windows

```
aws ssm describe-maintenance-window-targets ^
    --window-id "mw-0c50858d01EXAMPLE"
```

系统返回类似于以下内容的信息。

```
{
    "Targets": [
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "WindowTargetId": "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE",
            "ResourceType": "INSTANCE",
            "Targets": [
                {
                    "Key": "InstanceIds",
                    "Values": [
                        "i-02573cafcfEXAMPLE"
                    ]
                }
            ]
        }
    ]
}
```

继续步骤 3：使用 AWS CLI 向维护时段注册任务。

示例：向维护时段注册目标

您可以使用节点 ID 将单个节点注册为目标，如 步骤 2：使用 AWS CLI 将目标节点注册到维护时段 中所示。您还可以使用此页面的命令格式将一个或多个节点注册为目标。

一般情况下，有两种方法用于标识您要用作维护时段目标的节点：指定各个节点和使用资源标签。资源标签方法提供了更多选项，如示例 2-3 中所示。

您也可以指定一个或多个资源组作为维护时段的目标。资源组可包含节点和许多其他类型的受支持的 AWS 资源。接下来的示例 4 和 5 演示如何将资源组添加到您的维护时段目标。

> **Note**
>
> 如果将单个维护时段任务注册到多个目标，则其任务调用将按顺序进行，而不是并行发生。如果您的任务必须同时在多个目标上运行，请为每个目标分别注册一个任务，并为每个任务分配相同的优先级。

有关创建和管理资源组的更多信息，请参阅 AWS Resource Groups 用户指南中的什么是资源组？，以及 AWS 新闻博客中的 AWS 的资源组和标记。

有关 Maintenance Windows（AWS Systems Manager 中的一项工具）配额的信息，除了以下示例中指定的内容，另请参阅《Amazon Web Services 一般参考》中的 Systems Manager service quotas。

示例 1：使用节点 ID 注册多个目标

在本地计算机上运行以下命令，使用多个节点的节点 ID 将它们注册为目标。将每个*#######*替换为您自己的信息。

Linux & macOS

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-0c50858d01EXAMPLE" \
    --resource-type "INSTANCE" \
    --target
 "Key=InstanceIds,Values=i-02573cafcfEXAMPLE,i-0471e04240EXAMPLE,i-07782c72faEXAMPLE"
```

Windows

```
aws ssm register-target-with-maintenance-window ^
```

```
    --window-id "mw-0c50858d01EXAMPLE ^
    --resource-type "INSTANCE" ^
    --target
 "Key=InstanceIds,Values=i-02573cafcfEXAMPLE,i-0471e04240EXAMPLE,i-07782c72faEXAMPLE
```

建议使用：在首次将唯一一组节点注册到任何维护时段，并且这些节点不共享公用节点标签时最有用。

配额：您总共可以为每个维护时段目标指定最多 50 个节点。

示例 2：使用应用到节点的资源标签注册目标

在本地计算机上运行以下命令，注册所有已使用您分配的键值对标记的节点。将每个*#######*替换为您自己的信息。

Linux & macOS

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-0c50858d01EXAMPLE" \
    --resource-type "INSTANCE" \
    --target "Key=tag:Region,Values=East"
```

Windows

```
aws ssm register-target-with-maintenance-window ^
    --window-id "mw-0c50858d01EXAMPLE" ^
    --resource-type "INSTANCE" ^
    --target "Key=tag:Region,Values=East"
```

建议使用：在首次将唯一一组节点注册到任何维护时段，并且这些节点共享公用节点标签时最有用。

配额：您总共可以为每个目标指定最多 5 个键/值对。如果您指定了多个键/值对，则必须使用您指定的所有 标签键和值来标记节点，才能将该节点包含在目标组中。

> **Note**
>
> 您可以使用标签键`Patch Group`或`PatchGroup`来标记一组节点并向节点分配通用键值，例如 `my-patch-group`。（如果在 EC2 实例元数据中允许使用标签，则必须使用 `PatchGroup`，且不能使用空格。）Patch Manager是 Systems Manager 中的一项工具，用于评估节点上的 `Patch Group` 或 `PatchGroup` 键，以便确定适用的补丁基准。如果您的任

务将运行 AWS-RunPatchBaseline SSM 文档（或传统 AWS-ApplyPatchBaseline SSM 文档），则在将目标注册到维护时段时，可以指定相同的Patch Group或PatchGroup键值对。例如：--target "Key=tag:PatchGroup,Values=*my-patch-group*。这样做可让您使用维护时段，在已经与相同补丁基准关联的一组节点上更新补丁。有关更多信息，请参阅补丁组。

示例 3：使用一组标签键（无标签值）注册目标

在本地计算机上运行以下命令，注册所有已分配有一个或多个标签键的节点（不考虑其键值）。将每个*#######*替换为您自己的信息。

Linux & macOS

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-0c50858d01EXAMPLE" \
    --resource-type "INSTANCE" \
    --target "Key=tag-key,Values=Name,Instance-Type,CostCenter"
```

Windows

```
aws ssm register-target-with-maintenance-window ^
    --window-id "mw-0c50858d01EXAMPLE" ^
    --resource-type "INSTANCE" ^
    --target "Key=tag-key,Values=Name,Instance-Type,CostCenter"
```

建议使用：当您想要通过指定多个标签键（无值），而非仅指定一个标签键或标签键值对来将节点设为目标时很有用。

配额：您总共可以为每个目标指定最多 5 个标签键。如果您指定了多个标签键，则必须使用您指定的所有标签键来标记节点，才能将该节点包含在目标组中。

示例 4：使用资源组名称注册目标

在本地计算机上运行以下命令，注册指定的资源组，而不管其包含的资源类型。将 *mw-0c50858d01EXAMPLE* 替换为您自己的信息。如果分配给维护时段的任务不会在此资源组中包含的某资源类型上执行操作，则系统可能会报告错误。尽管存在这些错误，但找到了支持的资源类型的任务仍会继续运行。

Linux & macOS

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-0c50858d01EXAMPLE" \
    --resource-type "RESOURCE_GROUP" \
    --target "Key=resource-groups:Name,Values=MyResourceGroup"
```

Windows

```
aws ssm register-target-with-maintenance-window ^
    --window-id "mw-0c50858d01EXAMPLE" ^
    --resource-type "RESOURCE_GROUP" ^
    --target "Key=resource-groups:Name,Values=MyResourceGroup"
```

建议使用：当您想要将资源组快速指定为目标而不评估维护时段是否将其所有资源类型设为目标时，或者当您知道资源组仅包含任务对其执行操作的资源类型时，此选项很有用。

限额：您只能将一个资源组指定为目标。

示例 5：通过筛选资源组中的资源类型来注册目标

在本地计算机上运行以下命令，只注册属于您所指定的资源组的某些资源类型。将 *mw-0c50858d01EXAMPLE* 替换为您自己的信息。使用此选项，即使您为属于资源组的资源类型添加任务，但如果尚未将资源类型显式添加到筛选器，则任务也不会运行。

Linux & macOS

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-0c50858d01EXAMPLE" \
    --resource-type "RESOURCE_GROUP" \
    --target "Key=resource-groups:Name,Values=MyResourceGroup" \
    "Key=resource-
groups:ResourceTypeFilters,Values=AWS::EC2::Instance,AWS::ECS::Cluster"
```

Windows

```
aws ssm register-target-with-maintenance-window ^
    --window-id "mw-0c50858d01EXAMPLE" ^
    --resource-type "RESOURCE_GROUP" ^
    --target "Key=resource-groups:Name,Values=MyResourceGroup" ^
```

```
    "Key=resource-
groups:ResourceTypeFilters,Values=AWS::EC2::Instance,AWS::ECS::Cluster"
```

建议使用：当您希望严格控制维护时段可在其上运行操作的 AWS 资源的类型时，或者当您的资源组包含大量资源类型而您希望在维护时段日志中避免不必要的错误报告时，这非常有用。

限额：您只能将一个资源组指定为目标。

步骤 3：使用 AWS CLI 向维护时段注册任务

在教程的此步骤中，您将注册 AWS Systems Manager Run Command 任务，该任务将在适用于 Linux 的 Amazon Elastic Compute Cloud (Amazon EC2) 实例上运行 df 命令。此标准 Linux 命令的结果显示实例的磁盘文件系统上的空闲空间数量和已用空间数量。

–或者–

如果您设定的目标是适用于 Windows Server 而不是 Linux 的 Amazon EC2 实例，请将以下命令中的 df 替换为 ipconfig。此命令的输出列出了有关目标实例上适配器的 IP 地址、子网掩码和默认网关等详细信息。

当您准备好注册其他任务类型，或使用更多可用 Systems Manager Run Command 选项时，请参阅 示例：向维护时段注册任务。我们在这里提供了有关全部四种任务类型的更多信息，以及它们最重要的一些选项，以帮助您针对更广泛的真实情况制定计划。

向维护时段注册任务

1. 在本地计算机上运行以下命令。将每个*#######*替换为您自己的信息。从本地 Windows 计算机运行的版本包含转义字符 (“/”)，您需要从命令行工具运行命令。

   Linux & macOS

   ```
   aws ssm register-task-with-maintenance-window \
       --window-id mw-0c50858d01EXAMPLE \
       --task-arn "AWS-RunShellScript" \
       --max-concurrency 1 --max-errors 1 \
       --priority 10 \
       --targets "Key=InstanceIds,Values=i-0471e04240EXAMPLE" \
       --task-type "RUN_COMMAND" \
       --task-invocation-parameters '{"RunCommand":{"Parameters":{"commands":
   ["df"]}}}'
   ```

Windows

```
aws ssm register-task-with-maintenance-window ^
    --window-id mw-0c50858d01EXAMPLE ^
    --task-arn "AWS-RunShellScript" ^
    --max-concurrency 1 --max-errors 1 ^
    --priority 10 ^
    --targets "Key=InstanceIds,Values=i-02573cafcfEXAMPLE" ^
    --task-type "RUN_COMMAND" ^
    --task-invocation-parameters={\"RunCommand\":{\"Parameters\":{\"commands\":
[\"df\"]}}}
```

系统返回类似于下文的信息：

```
{
    "WindowTaskId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE"
}
```

2. 现在，运行以下命令查看有关您创建的维护时段任务的详细信息。

Linux & macOS

```
aws ssm describe-maintenance-window-tasks \
    --window-id mw-0c50858d01EXAMPLE
```

Windows

```
aws ssm describe-maintenance-window-tasks ^
    --window-id mw-0c50858d01EXAMPLE
```

3. 系统返回类似于以下内容的信息。

```
{
    "Tasks": [
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "WindowTaskId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE",
            "TaskArn": "AWS-RunShellScript",
            "Type": "RUN_COMMAND",
            "Targets": [
```

```
                {
                    "Key": "InstanceIds",
                    "Values": [
                        "i-02573cafcfEXAMPLE"
                    ]
                }
            ],
            "TaskParameters": {},
            "Priority": 10,
            "ServiceRoleArn": "arn:aws:iam::123456789012:role/
MyMaintenanceWindowServiceRole",
            "MaxConcurrency": "1",
            "MaxErrors": "1"
        }
    ]
}
```

4. 请耐心等待，直到有足够的时间根据您在步骤 1：使用 AWS CLI 创建维护时段 中指定的计划运行任务。例如，如果您指定了 **--schedule "rate(5 minutes)"**，则等待五分钟。然后，运行以下命令来查看此任务所进行的任何执行的相关信息。

   Linux & macOS

   ```
   aws ssm describe-maintenance-window-executions \
       --window-id mw-0c50858d01EXAMPLE
   ```

   Windows

   ```
   aws ssm describe-maintenance-window-executions ^
       --window-id mw-0c50858d01EXAMPLE
   ```

   系统返回类似于以下内容的信息。

   ```
   {
       "WindowExecutions": [
           {
               "WindowId": "mw-0c50858d01EXAMPLE",
               "WindowExecutionId": "14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE",
               "Status": "SUCCESS",
               "StartTime": 1557593493.096,
               "EndTime": 1557593498.611
   ```

```
        }
    ]
}
```

Tip

任务成功运行后，您可以降低维护时段运行的速率。例如，运行以下命令以将频率减少为每周一次。将 *mw-0c50858d01EXAMPLE* 替换为您自己的信息。

Linux & macOS

```
aws ssm update-maintenance-window \
    --window-id mw-0c50858d01EXAMPLE \
    --schedule "rate(7 days)"
```

Windows

```
aws ssm update-maintenance-window ^
    --window-id mw-0c50858d01EXAMPLE ^
    --schedule "rate(7 days)"
```

有关管理维护时段计划的信息，请参阅 参考：适用于 Systems Manager 的 Cron 和 Rate 表达式 和 维护时段计划和活动期间选项。
有关使用 AWS Command Line Interface (AWS CLI) 修改维护时段的信息，请参阅 教程：使用 AWS CLI 更新维护时段。

有关运行 AWS CLI 命令以查看有关维护时段任务及其执行详细信息的练习，请转到教程：查看有关任务和使用 AWS CLI 执行任务的信息。

访问教程命令输出

使用 AWS CLI 查看与您的维护时段任务执行相关联的 Run Command 命令的输出，不在本教程的讨论范围内。

不过，您可以使用 AWS CLI 查看此数据。（您还可以在 Systems Manager 控制台中或存储在 Amazon Simple Storage Service (Amazon S3) 存储桶内的日志文件中（如果您已配置维护时段将命令

输出存储在其中）查看输出。）您会看到，适用于 Linux 的 Amazon EC2 实例上 df 命令的输出类似于下文：

```
Filesystem 1K-blocks Used Available Use% Mounted on

devtmpfs 485716 0 485716 0% /dev

tmpfs 503624 0 503624 0% /dev/shm

tmpfs 503624 328 503296 1% /run

tmpfs 503624 0 503624 0% /sys/fs/cgroup

/dev/xvda1 8376300 1464160 6912140 18% /
```

Windows Server 的 EC2 实例上 ipconfig 命令的输出类似于下文：

```
Windows IP Configuration


Ethernet adapter Ethernet 2:

   Connection-specific DNS Suffix  . : example.com
   IPv4 Address. . . . . . . . . . . : 10.24.34.0/23
   Subnet Mask . . . . . . . . . . . : 255.255.255.255
   Default Gateway . . . . . . . . . : 0.0.0.0

Ethernet adapter Ethernet:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . : abc1.wa.example.net

Wireless LAN adapter Local Area Connection* 1:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::100b:c234:66d6:d24f%4
   IPv4 Address. . . . . . . . . . . : 192.0.2.0
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
```

```
   Default Gateway . . . . . . . . . : 192.0.2.0

Ethernet adapter Bluetooth Network Connection:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
```

示例：向维护时段注册任务

您可以使用 AWS Command Line Interface（AWS CLI）将 Run Command（AWS Systems Manager 中的一项工具）中的任务注册到维护时段，如向维护时段注册任务中所示。您还可以为 Systems Manager 自动化 工作流、AWS Lambda 函数和 AWS Step Functions 任务注册任务，如本主题后面的内容所示。

> **Note**
>
> 为维护时段 Run Command 类型任务指定一个或多个目标。根据任务的不同，目标对于其他维护时段任务类型（自动化、AWS Lambda 和 AWS Step Functions）是可选的。有关运行未指定目标的任务的更多信息，请参阅 注册不含目标的维护时段任务。

在本主题中，我们提供的示例演示了如何使用 AWS Command Line Interface (AWS CLI) 命令 `register-task-with-maintenance-window`，分别将四种支持的任务类型注册到维护时段。这些示例仅用于演示目的，不过您可以修改它们以创建工作任务注册命令。

使用 --cli-input-json 选项

为了更好地管理任务选项，您可以使用命令选项 `--cli-input-json`，其选项值在 JSON 文件中引用。

要使用我们在下列示例中提供的示例 JSON 文件内容，请在您本地计算机上执行以下操作：

1. 创建一个文件，采用类似于 `MyRunCommandTask.json`、`MyAutomationTask.json` 或其他您偏好的名称。
2. 将 JSON 示例内容复制到该文件中。
3. 为您的任务注册修改文件内容，然后保存该文件。
4. 在存储文件的同一个目录中，运行以下命令：用您的文件名替换 *`MyFile.json`*。

Linux & macOS

```
aws ssm register-task-with-maintenance-window \
    --cli-input-json file://MyFile.json
```

Windows

```
aws ssm register-task-with-maintenance-window ^
    --cli-input-json file://MyFile.json
```

维护时段任务中的伪参数

在某些示例中，我们使用伪参数 作为将 ID 信息传递到您任务的方法。例如，{{TARGET_ID}} 和 {{RESOURCE_ID}} 可用于将 AWS 资源的 ID 传递到 自动化、Lambda 和 Step Functions 任务。有关 --task-invocation-parameters 内容中伪参数的更多信息，请参阅 注册维护时段任务时使用伪参数。

更多信息

- register-task-with-maintenance-windows 命令的参数选项.
- 《AWS CLI Command Reference》中的 register-task-with-maintenance-window。
- 《AWS Systems Manager API 参考》中的 RegisterTaskWithMaintenanceWindow

任务注册示例

以下部分提供了一个示例 AWS CLI 命令，用于注册支持的任务类型和可与 --cli-input-json 选项一起使用的 JSON 示例。

注册 Systems Manager Run Command 任务

以下示例演示了如何使用 AWS CLI 将 Systems Manager Run Command 任务注册到维护时段。

Linux & macOS

```
aws ssm register-task-with-maintenance-window \
    --window-id mw-0c50858d01EXAMPLE \
    --task-arn "AWS-RunShellScript" \
    --max-concurrency 1 --max-errors 1 --priority 10 \
    --targets "Key=InstanceIds,Values=i-02573cafcfEXAMPLE" \
```

```
    --task-type "RUN_COMMAND" \
    --task-invocation-parameters '{"RunCommand":{"Parameters":{"commands":["df"]}}}'
```

Windows

```
aws ssm register-task-with-maintenance-window ^
    --window-id mw-0c50858d01EXAMPLE ^
    --task-arn "AWS-RunShellScript" ^
    --max-concurrency 1 --max-errors 1 --priority 10 ^
    --targets "Key=InstanceIds,Values=i-02573cafcfEXAMPLE" ^
    --task-type "RUN_COMMAND" ^
    --task-invocation-parameters "{\"RunCommand\":{\"Parameters\":{\"commands\":
[\"df\"]}}}"
```

用于 **--cli-input-json** 文件选项的 JSON 内容：

```
{
    "TaskType": "RUN_COMMAND",
    "WindowId": "mw-0c50858d01EXAMPLE",
    "Description": "My Run Command task to update SSM Agent on an instance",
    "MaxConcurrency": "1",
    "MaxErrors": "1",
    "Name": "My-Run-Command-Task",
    "Priority": 10,
    "Targets": [
        {
            "Key": "WindowTargetIds",
            "Values": [
                "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
            ]
        }
    ],
    "TaskArn": "AWS-UpdateSSMAgent",
    "TaskInvocationParameters": {
        "RunCommand": {
            "Comment": "A TaskInvocationParameters test comment",
            "NotificationConfig": {
                "NotificationArn": "arn:aws:sns:region:123456789012:my-sns-topic-name",
                "NotificationEvents": [
                    "All"
                ],
                "NotificationType": "Invocation"
```

```
            },
            "OutputS3BucketName": "amzn-s3-demo-bucket",
            "OutputS3KeyPrefix": "S3-PREFIX",
            "TimeoutSeconds": 3600
        }
    }
}
```

注册 Systems Manager 自动化 任务

以下示例演示了如何使用 AWS CLI 将 Systems Manager 自动化 任务注册到维护时段：

AWS CLI 命令：

Linux & macOS

```
aws ssm register-task-with-maintenance-window \
    --window-id "mw-0c50858d01EXAMPLE" \
    --task-arn "AWS-RestartEC2Instance" \
    --service-role-arn arn:aws:iam::123456789012:role/MyMaintenanceWindowServiceRole
 \
    --task-type AUTOMATION \
    --task-invocation-parameters
 "Automation={DocumentVersion=5,Parameters={InstanceId='{{RESOURCE_ID}}'}}" \
    --priority 0 --name "My-Restart-EC2-Instances-Automation-Task" \
    --description "Automation task to restart EC2 instances"
```

Windows

```
aws ssm register-task-with-maintenance-window ^
    --window-id "mw-0c50858d01EXAMPLE" ^
    --task-arn "AWS-RestartEC2Instance" ^
    --service-role-arn arn:aws:iam::123456789012:role/MyMaintenanceWindowServiceRole
 ^
    --task-type AUTOMATION ^
    --task-invocation-parameters
 "Automation={DocumentVersion=5,Parameters={InstanceId='{{TARGET_ID}}'}}" ^
    --priority 0 --name "My-Restart-EC2-Instances-Automation-Task" ^
    --description "Automation task to restart EC2 instances"
```

用于 **--cli-input-json** 文件选项的 JSON 内容：

```
{
    "WindowId": "mw-0c50858d01EXAMPLE",
        "TaskArn": "AWS-PatchInstanceWithRollback",
    "TaskType": "AUTOMATION","TaskInvocationParameters": {
        "Automation": {
            "DocumentVersion": "1",
            "Parameters": {
                "instanceId": [
                    "{{RESOURCE_ID}}"
                ]
            }
        }
    }
}
```

注册 AWS Lambda 任务

以下示例演示了如何使用 AWS CLI 将 Lambda 函数任务注册到维护时段。

对于这些示例，创建 Lambda 函数的用户将其命名为 `SSMrestart-my-instances`，并创建了名为 `instanceId` 和 `targetType` 的两个参数。

> **Important**
>
> Maintenance Windows 的 IAM policy 要求您将前缀 `SSM` 添加到 Lambda 函数（或别名）名称。继续注册此类型的任务之前，请在 AWS Lambda 中更新其名称，以包含 `SSM`。例如，如果您的 Lambda 函数名称为 `MyLambdaFunction`，请将其更改为 `SSMMyLambdaFunction`。

AWS CLI 命令：

Linux & macOS

> **Important**
>
> 如果您使用的是版本 2 的 AWS CLI，并且您的 Lambda 有效负载没有进行 base64 编码，则您必须在以下命令中包含选项 `--cli-binary-format raw-in-base64-out`。`cli_binary_format` 选项仅在版本 2 中可用。有关此设置和其他 AWS CLI

config 文件设置的信息，请参阅 AWS Command Line Interface 用户指南中的受支持的 config 文件设置。

```
aws ssm register-task-with-maintenance-window \
    --window-id "mw-0c50858d01EXAMPLE" \
    --targets "Key=WindowTargetIds,Values=e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE" \
    --priority 2 --max-concurrency 10 --max-errors 5 --name "My-Lambda-Example" \
    --description "A description for my LAMBDA example task" --task-type "LAMBDA" \
    --task-arn "arn:aws:lambda:region:123456789012:function:serverlessrepo-
SSMrestart-my-instances-C4JF9EXAMPLE" \
    --task-invocation-parameters '{"Lambda":{"Payload":"{\"InstanceId\":
\"{{RESOURCE_ID}}\",\"targetType\":\"{{TARGET_TYPE}}\"}","Qualifier": "$LATEST"}}'
```

PowerShell

Important

如果您使用的是版本 2 的 AWS CLI，并且您的 Lambda 有效负载没有进行 base64 编码，则您必须在以下命令中包含选项 `--cli-binary-format raw-in-base64-out`。`cli_binary_format` 选项仅在版本 2 中可用。有关此设置和其他 AWS CLI config 文件设置的信息，请参阅 AWS Command Line Interface 用户指南中的受支持的 config 文件设置。

```
aws ssm register-task-with-maintenance-window `
    --window-id "mw-0c50858d01EXAMPLE" `
    --targets "Key=WindowTargetIds,Values=e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE" `
    --priority 2 --max-concurrency 10 --max-errors 5 --name "My-Lambda-Example" `
    --description "A description for my LAMBDA example task" --task-type "LAMBDA" `
    --task-arn "arn:aws:lambda:region:123456789012:function:serverlessrepo-
SSMrestart-my-instances-C4JF9EXAMPLE" `
    --task-invocation-parameters '{\"Lambda\":{\"Payload\":\"{\\\"InstanceId\\\":\\
\"{{RESOURCE_ID}}\\\",\\\"targetType\\\":\\\"{{TARGET_TYPE}}\\\"}\",\"Qualifier\":
 \"$LATEST\"}}'
```

用于 **`--cli-input-json`** 文件选项的 JSON 内容：

```
{
```

```
    "WindowId": "mw-0c50858d01EXAMPLE",
    "Targets": [
        {
            "Key": "WindowTargetIds",
            "Values": [
                "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
            ]
        }
    ],
    "TaskArn": "SSM_RestartMyInstances",
    "TaskType": "LAMBDA",
    "MaxConcurrency": "10",
    "MaxErrors": "10",
    "TaskInvocationParameters": {
        "Lambda": {
            "ClientContext": "ew0KICAi--truncated--0KIEXAMPLE",
            "Payload": "{ \"instanceId\": \"{{RESOURCE_ID}}\", \"targetType\":
 \"{{TARGET_TYPE}}\" }",
            "Qualifier": "$LATEST"
        }
    },
    "Name": "My-Lambda-Task",
    "Description": "A description for my LAMBDA task",
    "Priority": 5
}
```

注册 Step Functions 任务

以下示例演示了如何使用 AWS CLI 将 Step Functions 状态机任务注册到维护时段。

> **Note**
>
> 维护时段任务仅支持 Step Functions 标准状态机工作流。它们不支持快速状态机工作流。有关状态机工作流类型的信息，请参阅《AWS Step Functions 开发人员指南》中的标准与快速工作流。

对于这些示例，创建 Step Functions 状态机的用户使用名为 `instanceId` 的参数创建名为 `SSMMyStateMachine` 的状态机。

> **Important**
>
> Maintenance Windows 的 AWS Identity and Access Management （IAM）policy 要求您使用 SSM 作为 Step Functions 状态机名称的前缀。继续注册此类型的任务之前，您必须在 AWS Step Functions 中更新此名称以包括 SSM。例如，如果您的状态机名为 MyStateMachine，则将其更改为 SSMMyStateMachine。

AWS CLI 命令：

Linux & macOS

```
aws ssm register-task-with-maintenance-window \
    --window-id "mw-0c50858d01EXAMPLE" \
    --targets "Key=WindowTargetIds,Values=e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE" \
    --task-arn arn:aws:states:region:123456789012:stateMachine:SSMMyStateMachine-
MggiqEXAMPLE \
    --task-type STEP_FUNCTIONS \
    --task-invocation-parameters '{"StepFunctions":{"Input":"{\"InstanceId\":
\"{{RESOURCE_ID}}\"}", "Name":"{{INVOCATION_ID}}"}}' \
    --priority 0 --max-concurrency 10 --max-errors 5 \
    --name "My-Step-Functions-Task" --description "A description for my Step
 Functions task"
```

PowerShell

```
aws ssm register-task-with-maintenance-window `
    --window-id "mw-0c50858d01EXAMPLE" `
    --targets "Key=WindowTargetIds,Values=e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE" `
    --task-arn arn:aws:states:region:123456789012:stateMachine:SSMMyStateMachine-
MggiqEXAMPLE `
    --task-type STEP_FUNCTIONS `
    --task-invocation-parameters '{\"StepFunctions\":{\"Input\":\"{\\\"InstanceId\\
\":\\\"{{RESOURCE_ID}}\\\"}\", \"Name\":\"{{INVOCATION_ID}}\"}}' `
    --priority 0 --max-concurrency 10 --max-errors 5 `
    --name "My-Step-Functions-Task" --description "A description for my Step
 Functions task"
```

用于 **`--cli-input-json`** 文件选项的 JSON 内容：

```
{
    "WindowId": "mw-0c50858d01EXAMPLE",
    "Targets": [
        {
            "Key": "WindowTargetIds",
            "Values": [
                "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
            ]
        }
    ],
    "TaskArn": "SSM_MyStateMachine",
    "TaskType": "STEP_FUNCTIONS",
    "MaxConcurrency": "10",
    "MaxErrors": "10",
    "TaskInvocationParameters": {
        "StepFunctions": {
            "Input": "{ \"instanceId\": \"{{TARGET_ID}}\" }",
            "Name": "{{INVOCATION_ID}}"
        }
    },
    "Name": "My-Step-Functions-Task",
    "Description": "A description for my Step Functions task",
    "Priority": 5
}
```

register-task-with-maintenance-windows 命令的参数选项

register-task-with-maintenance-window 命令提供多个选项用于根据您的需求配置任务。其中一些任务为必需，一些为可选，还有一些仅适用于单个维护时段任务类型。

本主题提供了有关其中一些选项的信息，以帮助您使用本教程这一部分中的示例。有关所有命令选项的信息，请参阅 AWS CLI 命令参考中的 register-task-with-maintenance-window。

命令选项：**`--task-arn`**

`--task-arn` 选项用于指定执行任务的资源。您指定的值取决于所要注册的任务类型，如下表中所述。

维护时段任务的 TaskArn 格式

维护时段任务类型	TaskArn 值
RUN_COMMAND 和 **AUTOMATION**	TaskArn 是 SSM 文档名称或 Amazon Resource Name (ARN)。例如： `AWS-RunBatchShellScript` –或者– `arn:aws:ssm:` *`region`*`:11112222 3333:document/My-Document` .
LAMBDA	TaskArn 是函数名称或 ARN。例如： `SSMMy-Lambda-Function` –或者– `arn:aws:lambda:` *`region`*`:11112222 3333:function:SSMMyLambdaFu nction` . **Important** Maintenance Windows 的 IAM policy 要求您将前缀 SSM 添加到 Lambda 函数（或别名）名称。继续注册此类型的任务之前，请在 AWS Lambda 中更新其名称，以包含 SSM。例如，如果您的 Lambda 函数名称为 `MyLambdaF unction` ，请将其更改为 `SSMMyLamb daFunction` 。
STEP_FUNCTIONS	TaskArn 是状态机 ARN。例如： `arn:aws:states:us-east-2:11 1122223333:stateMachine:SSM MyStateMachine` .

维护时段任务类型	TaskArn 值
	Important 维护时段的 IAM policy 要求您使用 SSM 作为 Step Functions 状态机名称的前缀。您应先注册此类型的任务，然后必须在 AWS Step Functions 中更新其名称以包含 SSM。例如，如果您的状态机名为 MyStateMachine ，则将其更改为 SSMMyStateMachine 。

命令选项：**--service-role-arn**

运行维护时段任务时 AWS Systems Manager 要代入的角色。

有关更多信息，请参阅 设置 Maintenance Windows。

命令选项：**--task-invocation-parameters**

--task-invocation-parameters 选项用于指定分别对于四种任务类型唯一的参数。四种任务类型分别支持的参数如下表中所述。

> **Note**
> 有关在 --task-invocation-parameters 内容中使用伪参数的信息，例如 {{TARGET_ID}}，请参阅注册维护时段任务时使用伪参数。

维护任务时段的任务调用参数选项

维护时段任务类型	可用参数	示例
RUN_COMMAND	Comment DocumentHash DocumentHashType	"TaskInvocationPar ameters": { "RunCommand": { "Comment" : "My Run Command task comment",

维护时段任务类型	可用参数	示例

	NotificationConfig
	
	OutputS3BucketName
	
	OutPutS3KeyPrefix
	
	Parameters
	
	ServiceRoleArn
	
	TimeoutSeconds
		<pre>"Document Hash": "6554ed3d--truncated--5EXAMPLE",
		"Document HashType": "Sha256",
		"NotificationConfig": {
		"NotificationArn": "arn:aws:sns: *region*:123456789012:my-sns-topic-name",
		"NotificationEvents": [
		"FAILURE"
		],
		"NotificationType": "Invocation"
		},
		"OutputS3BucketName": "amzn-s3-demo-bucket",
		"OutputS3KeyPrefix": " *S3-PREFIX* ",
		"Parameters": {
		"commands": [
		"Get-ChildItem$env:temp-Recurse|Remove-Item-Recurse-force"
		]
		},
		"ServiceRoleArn": "arn:aws:iam::123456789012:</pre>

维护时段任务类型	可用参数	示例
		role/MyMaintenance WindowServiceRole", "TimeoutSeconds": 3600 } }
AUTOMATION	DocumentVersion Parameters	"TaskInvocationParameters": { "Automation": { "DocumentVersion": "3", "Parameters": { "instanceid": ["{{TARGET_ID}}"] } } }

维护时段任务类型	可用参数	示例

LAMBDA	ClientContext
	
	Payload
	
	Qualifier
		<pre>"TaskInvocationPar
		ameters": {
		"Lambda": {
		"ClientCo
		ntext": "ew0KICAi
		--truncated--0KIEX
		AMPLE",
		"Payload"
		: "{ \"targetId\":
		\"{{TARGET_ID}}\",
		\"targetType\":
		\"{{TARGET_TYPE}}\
		" }",
		"Qualifie
		r": "$LATEST"
		}
		}</pre>

STEP_FUNCTIONS	Input
	
	Name
		<pre>"TaskInvocationPar
		ameters": {
		"StepFunc
		tions": {
		"Input":
		"{ \"targetId\":
		\"{{TARGET_ID}}\" }",
		"Name":
		"{{INVOCATION_ID}}"
		}
		}</pre>

教程：使用 AWS CLI 查看有关维护时段的信息

本教程包含的命令可帮助您更新维护时段、任务、执行和调用或获取相关信息。这些示例按命令进行组织，说明如何使用命令选项筛选要查看的详细信息类型。

在按照本教程中的步骤操作时，请将以斜体*##*文本显示的值替换为您自己的选项和 ID。例如，请将维护时段 ID *mw-0c50858d01EXAMPLE* 和实例 ID *i-02573cafcfEXAMPLE* 替换为您创建的资源的 ID。

有关设置和配置 AWS Command Line Interface (AWS CLI) 的信息，请参阅安装、更新和卸载 AWS CLI 和配置 AWS CLI。

命令示例

- 'describe-maintenance-windows' 的示例
- 'describe-maintenance-window-targets' 的示例
- 'describe-maintenance-window-tasks' 的示例
- 'describe-maintenance-windows-for-target' 的示例
- 'describe-maintenance-window-executions' 的示例
- 'describe-maintenance-window-schedule' 的示例

'describe-maintenance-windows' 的示例

列出您的 AWS 账户中的所有维护时段

运行以下命令。

```
aws ssm describe-maintenance-windows
```

系统返回类似于以下内容的信息。

```
{
   "WindowIdentities":[
      {
         "WindowId":"mw-0c50858d01EXAMPLE",
         "Name":"My-First-Maintenance-Window",
         "Enabled":true,
         "Duration":2,
         "Cutoff":0,
         "NextExecutionTime": "2019-05-18T17:01:01.137Z"
      },
      {
         "WindowId":"mw-9a8b7c6d5eEXAMPLE",
         "Name":"My-Second-Maintenance-Window",
         "Enabled":true,
         "Duration":4,
         "Cutoff":1,
         "NextExecutionTime": "2019-05-30T03:30:00.137Z"
      },
```

```
   ]
}
```

列出所有已启用的维护时段

运行以下命令。

```
aws ssm describe-maintenance-windows --filters "Key=Enabled,Values=true"
```

系统返回类似于以下内容的信息。

```
{
   "WindowIdentities":[
      {
         "WindowId":"mw-0c50858d01EXAMPLE",
         "Name":"My-First-Maintenance-Window",
         "Enabled":true,
         "Duration":2,
         "Cutoff":0,
         "NextExecutionTime": "2019-05-18T17:01:01.137Z"
      },
      {
         "WindowId":"mw-9a8b7c6d5eEXAMPLE",
         "Name":"My-Second-Maintenance-Window",
         "Enabled":true,
         "Duration":4,
         "Cutoff":1,
         "NextExecutionTime": "2019-05-30T03:30:00.137Z"
      },
   ]
}
```

列出所有已禁用的维护时段

运行以下命令。

```
aws ssm describe-maintenance-windows --filters "Key=Enabled,Values=false"
```

系统返回类似于以下内容的信息。

```
{
    "WindowIdentities": [
```

```
        {
            "WindowId": "mw-6e5c9d4b7cEXAMPLE",
            "Name": "My-Disabled-Maintenance-Window",
            "Enabled": false,
            "Duration": 2,
            "Cutoff": 1
        }
    ]
}
```

列出名称以特定前缀开头的所有维护时段

运行以下命令。

```
aws ssm describe-maintenance-windows --filters "Key=Name,Values=My"
```

系统返回类似于以下内容的信息。

```
{
    "WindowIdentities": [
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Name": "My-First-Maintenance-Window",
            "Enabled": true,
            "Duration": 2,
            "Cutoff": 0,
            "NextExecutionTime": "2019-05-18T17:01:01.137Z"
        },
        {
            "WindowId": "mw-9a8b7c6d5eEXAMPLE",
            "Name": "My-Second-Maintenance-Window",
            "Enabled": true,
            "Duration": 4,
            "Cutoff": 1,
            "NextExecutionTime": "2019-05-30T03:30:00.137Z"
        },
        {
            "WindowId": "mw-6e5c9d4b7cEXAMPLE",
            "Name": "My-Disabled-Maintenance-Window",
            "Enabled": false,
            "Duration": 2,
            "Cutoff": 1
        }
```

```
    ]
}
```

'describe-maintenance-window-targets' 的示例

显示匹配特定所有者信息值的维护时段的目标

运行以下命令。

Linux & macOS

```
aws ssm describe-maintenance-window-targets \
    --window-id "mw-6e5c9d4b7cEXAMPLE" \
    --filters "Key=OwnerInformation,Values=CostCenter1"
```

Windows

```
aws ssm describe-maintenance-window-targets ^
    --window-id "mw-6e5c9d4b7cEXAMPLE" ^
    --filters "Key=OwnerInformation,Values=CostCenter1"
```

> **Note**
>
> 支持的筛选条件键为 Type、WindowTargetId 和 OwnerInformation。

系统返回类似于以下内容的信息。

```
{
    "Targets": [
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "WindowTargetId": "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE",
            "ResourceType": "INSTANCE",
            "Targets": [
                {
                    "Key": "tag:Name",
                    "Values": [
                        "Production"
                    ]
                }
```

```
            ],
            "OwnerInformation": "CostCenter1",
            "Name": "Target1"
        }
    ]
}
```

'describe-maintenance-window-tasks' 的示例

显示调用 SSM 命令文档 **AWS-RunPowerShellScript** 的所有已注册任务

运行以下命令。

Linux & macOS

```
aws ssm describe-maintenance-window-tasks \
    --window-id "mw-0c50858d01EXAMPLE" \
    --filters "Key=TaskArn,Values=AWS-RunPowerShellScript"
```

Windows

```
aws ssm describe-maintenance-window-tasks ^
    --window-id "mw-0c50858d01EXAMPLE" ^
    --filters "Key=TaskArn,Values=AWS-RunPowerShellScript"
```

系统返回类似于以下内容的信息。

```
{
   "Tasks":[
      {
         "ServiceRoleArn": "arn:aws:iam::111122223333:role/
MyMaintenanceWindowServiceRole",
         "MaxErrors":"1",
         "TaskArn":"AWS-RunPowerShellScript",
         "MaxConcurrency":"1",
         "WindowTaskId":"4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE",
         "TaskParameters":{
            "commands":{
               "Values":[
                  "driverquery.exe"
               ]
```

```
            }
         },
         "Priority":3,
         "Type":"RUN_COMMAND",
         "Targets":[
            {
               "TaskTargetId":"i-02573cafcfEXAMPLE",
               "TaskTargetType":"INSTANCE"
            }
         ]
      },
      {
         "ServiceRoleArn":"arn:aws:iam::111122223333:role/
MyMaintenanceWindowServiceRole",
         "MaxErrors":"1",
         "TaskArn":"AWS-RunPowerShellScript",
         "MaxConcurrency":"1",
         "WindowTaskId":"4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE",
         "TaskParameters":{
            "commands":{
               "Values":[
                  "ipconfig"
               ]
            }
         },
         "Priority":1,
         "Type":"RUN_COMMAND",
         "Targets":[
            {
               "TaskTargetId":"i-02573cafcfEXAMPLE",
               "TaskTargetType":"WINDOW_TARGET"
            }
         ]
      }
   ]
}
```

显示优先级为“3”的所有已注册任务

运行以下命令。

Linux & macOS

```
aws ssm describe-maintenance-window-tasks \
```

```
    --window-id "mw-9a8b7c6d5eEXAMPLE" \
    --filters "Key=Priority,Values=3"
```

Windows

```
aws ssm describe-maintenance-window-tasks ^
    --window-id "mw-9a8b7c6d5eEXAMPLE" ^
    --filters "Key=Priority,Values=3"
```

系统返回类似于以下内容的信息。

```
{
   "Tasks":[
      {
         "ServiceRoleArn":"arn:aws:iam::111122223333:role/
MyMaintenanceWindowServiceRole",
         "MaxErrors":"1",
         "TaskArn":"AWS-RunPowerShellScript",
         "MaxConcurrency":"1",
         "WindowTaskId":"4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE",
         "TaskParameters":{
            "commands":{
               "Values":[
                  "driverquery.exe"
               ]
            }
         },
         "Priority":3,
         "Type":"RUN_COMMAND",
         "Targets":[
            {
               "TaskTargetId":"i-02573cafcfEXAMPLE",
               "TaskTargetType":"INSTANCE"
            }
         ]
      }
   ]
}
```

显示优先级为 1 且使用 Run Command 的所有已注册任务

运行以下命令。

Linux & macOS

```
aws ssm describe-maintenance-window-tasks \
    --window-id "mw-0c50858d01EXAMPLE" \
    --filters "Key=Priority,Values=1" "Key=TaskType,Values=RUN_COMMAND"
```

Windows

```
aws ssm describe-maintenance-window-tasks ^
    --window-id "mw-0c50858d01EXAMPLE" ^
    --filters "Key=Priority,Values=1" "Key=TaskType,Values=RUN_COMMAND"
```

系统返回类似于以下内容的信息。

```
{
    "Tasks": [
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "WindowTaskId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE",
            "TaskArn": "AWS-RunShellScript",
            "Type": "RUN_COMMAND",
            "Targets": [
                {
                    "Key": "InstanceIds",
                    "Values": [
                        "i-02573cafcfEXAMPLE"
                    ]
                }
            ],
            "TaskParameters": {},
            "Priority": 1,
            "ServiceRoleArn": "arn:aws:iam::111122223333:role/
MyMaintenanceWindowServiceRole",
            "MaxConcurrency": "1",
            "MaxErrors": "1"
        },
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "WindowTaskId": "8a5c4629-31b0-4edd-8aea-33698EXAMPLE",
            "TaskArn": "AWS-UpdateSSMAgent",
            "Type": "RUN_COMMAND",
            "Targets": [
```

```
                {
                    "Key": "InstanceIds",
                    "Values": [
                        "i-0471e04240EXAMPLE"
                    ]
                }
            ],
            "TaskParameters": {},
            "Priority": 1,
            "ServiceRoleArn": "arn:aws:iam::111122223333:role/
MyMaintenanceWindowServiceRole",
            "MaxConcurrency": "1",
            "MaxErrors": "1",
            "Name": "My-Run-Command-Task",
            "Description": "My Run Command task to update SSM Agent on an instance"
        }
    ]
}
```

'describe-maintenance-windows-for-target' 的示例

列出与特定节点关联的维护时段目标或任务的相关信息

运行以下命令。

Linux & macOS

```
aws ssm describe-maintenance-windows-for-target \
    --resource-type INSTANCE \
    --targets "Key=InstanceIds,Values=i-02573cafcfEXAMPLE" \
    --max-results 10
```

Windows

```
aws ssm describe-maintenance-windows-for-target ^
    --resource-type INSTANCE ^
    --targets "Key=InstanceIds,Values=i-02573cafcfEXAMPLE" ^
    --max-results 10
```

系统返回类似于以下内容的信息。

```
{
```

```
    "WindowIdentities": [
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Name": "My-First-Maintenance-Window"
        },
        {
            "WindowId": "mw-9a8b7c6d5eEXAMPLE",
            "Name": "My-Second-Maintenance-Window"
        }
    ]
}
```

'describe-maintenance-window-executions' 的示例

列出在特定日期前运行的所有任务

运行以下命令。

Linux & macOS

```
aws ssm describe-maintenance-window-executions \
    --window-id "mw-9a8b7c6d5eEXAMPLE" \
    --filters "Key=ExecutedBefore,Values=2019-05-12T05:00:00Z"
```

Windows

```
aws ssm describe-maintenance-window-executions ^
    --window-id "mw-9a8b7c6d5eEXAMPLE" ^
    --filters "Key=ExecutedBefore,Values=2019-05-12T05:00:00Z"
```

系统返回类似于以下内容的信息。

```
{
    "WindowExecutions": [
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "WindowExecutionId": "14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE",
            "Status": "FAILED",
            "StatusDetails": "The following SSM parameters are invalid: LevelUp",
            "StartTime": 1557617747.993,
            "EndTime": 1557617748.101
        },
```

```
        {
            "WindowId": "mw-9a8b7c6d5eEXAMPLE",
            "WindowExecutionId": "791b72e0-f0da-4021-8b35-f95dfEXAMPLE",
            "Status": "SUCCESS",
            "StartTime": 1557594085.428,
            "EndTime": 1557594090.978
        },
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "WindowExecutionId": "ecec60fa-6bb0-4d26-98c7-140308EXAMPLE",
            "Status": "SUCCESS",
            "StartTime": 1557593793.483,
            "EndTime": 1557593798.978
        }
    ]
}
```

列出在特定日期后运行的所有任务

运行以下命令。

Linux & macOS

```
aws ssm describe-maintenance-window-executions \
    --window-id "mw-9a8b7c6d5eEXAMPLE" \
    --filters "Key=ExecutedAfter,Values=2018-12-31T17:00:00Z"
```

Windows

```
aws ssm describe-maintenance-window-executions ^
    --window-id "mw-9a8b7c6d5eEXAMPLE" ^
    --filters "Key=ExecutedAfter,Values=2018-12-31T17:00:00Z"
```

系统返回类似于以下内容的信息。

```
{
    "WindowExecutions": [
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "WindowExecutionId": "14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE",
            "Status": "FAILED",
```

```
            "StatusDetails": "The following SSM parameters are invalid: LevelUp",
            "StartTime": 1557617747.993,
            "EndTime": 1557617748.101
        },
        {
            "WindowId": "mw-9a8b7c6d5eEXAMPLE",
            "WindowExecutionId": "791b72e0-f0da-4021-8b35-f95dfEXAMPLE",
            "Status": "SUCCESS",
            "StartTime": 1557594085.428,
            "EndTime": 1557594090.978
        },
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "WindowExecutionId": "ecec60fa-6bb0-4d26-98c7-140308EXAMPLE",
            "Status": "SUCCESS",
            "StartTime": 1557593793.483,
            "EndTime": 1557593798.978
        }
    ]
}
```

'describe-maintenance-window-schedule' 的示例

显示特定节点接下来的十次已计划的维护时段运行

运行以下命令。

Linux & macOS

```
aws ssm describe-maintenance-window-schedule \
    --resource-type INSTANCE \
    --targets "Key=InstanceIds,Values=i-07782c72faEXAMPLE" \
    --max-results 10
```

Windows

```
aws ssm describe-maintenance-window-schedule ^
    --resource-type INSTANCE ^
    --targets "Key=InstanceIds,Values=i-07782c72faEXAMPLE" ^
    --max-results 10
```

系统返回类似于以下内容的信息。

```
{
    "ScheduledWindowExecutions": [
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Name": "My-First-Maintenance-Window",
            "ExecutionTime": "2019-05-18T23:35:24.902Z"
        },
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Name": "My-First-Maintenance-Window",
            "ExecutionTime": "2019-05-25T23:35:24.902Z"
        },
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Name": "My-First-Maintenance-Window",
            "ExecutionTime": "2019-06-01T23:35:24.902Z"
        },
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Name": "My-First-Maintenance-Window",
            "ExecutionTime": "2019-06-08T23:35:24.902Z"
        },
        {
            "WindowId": "mw-9a8b7c6d5eEXAMPLE",
            "Name": "My-Second-Maintenance-Window",
            "ExecutionTime": "2019-06-15T23:35:24.902Z"
        },
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Name": "My-First-Maintenance-Window",
            "ExecutionTime": "2019-06-22T23:35:24.902Z"
        },
        {
            "WindowId": "mw-9a8b7c6d5eEXAMPLE",
            "Name": "My-Second-Maintenance-Window",
            "ExecutionTime": "2019-06-29T23:35:24.902Z"
        },
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Name": "My-First-Maintenance-Window",
            "ExecutionTime": "2019-07-06T23:35:24.902Z"
        },
        {
```

```
            "WindowId": "mw-9a8b7c6d5eEXAMPLE",
            "Name": "My-Second-Maintenance-Window",
            "ExecutionTime": "2019-07-13T23:35:24.902Z"
        },
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Name": "My-First-Maintenance-Window",
            "ExecutionTime": "2019-07-20T23:35:24.902Z"
        }
    ],
    "NextToken": "AAEABUXdceT92FvtKld/dGHELj5Mi+GKW/EXAMPLE"
}
```

显示使用特定键值对标记的节点的维护时段计划

运行以下命令。

Linux & macOS

```
aws ssm describe-maintenance-window-schedule \
    --resource-type INSTANCE \
    --targets "Key=tag:prod,Values=rhel7"
```

Windows

```
aws ssm describe-maintenance-window-schedule ^
    --resource-type INSTANCE ^
    --targets "Key=tag:prod,Values=rhel7"
```

系统返回类似于以下内容的信息。

```
{
    "ScheduledWindowExecutions": [
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Name": "DemoRateStartDate",
            "ExecutionTime": "2019-10-20T05:34:56-07:00"
        },
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Name": "DemoRateStartDate",
```

```
            "ExecutionTime": "2019-10-21T05:34:56-07:00"
        },
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Name": "DemoRateStartDate",
            "ExecutionTime": "2019-10-22T05:34:56-07:00"
        },
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Name": "DemoRateStartDate",
            "ExecutionTime": "2019-10-23T05:34:56-07:00"
        },
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Name": "DemoRateStartDate",
            "ExecutionTime": "2019-10-24T05:34:56-07:00"
        }
    ],
    "NextToken": "AAEABccwSXqQRGKiTZ1yzGELR6cxW4W/EXAMPLE"
}
```

显示某个维护时段接下来的四个运行的开始时间

运行以下命令。

Linux & macOS

```
aws ssm describe-maintenance-window-schedule \
    --window-id "mw-0c50858d01EXAMPLE" \
    --max-results "4"
```

Windows

```
aws ssm describe-maintenance-window-schedule ^
    --window-id "mw-0c50858d01EXAMPLE" ^
    --max-results "4"
```

系统返回类似于以下内容的信息。

```
{
    "WindowSchedule": [
```

```
        {
            "ScheduledWindowExecutions": [
                {
                    "ExecutionTime": "2019-10-04T10:10:10Z",
                    "Name": "My-First-Maintenance-Window",
                    "WindowId": "mw-0c50858d01EXAMPLE"
                },
                {
                    "ExecutionTime": "2019-10-11T10:10:10Z",
                    "Name": "My-First-Maintenance-Window",
                    "WindowId": "mw-0c50858d01EXAMPLE"
                },
                {
                    "ExecutionTime": "2019-10-18T10:10:10Z",
                    "Name": "My-First-Maintenance-Window",
                    "WindowId": "mw-0c50858d01EXAMPLE"
                },
                {
                    "ExecutionTime": "2019-10-25T10:10:10Z",
                    "Name": "My-First-Maintenance-Window",
                    "WindowId": "mw-0c50858d01EXAMPLE"
                }
            ]
        }
    ]
}
```

教程：查看有关任务和使用 AWS CLI 执行任务的信息

本教程演示了如何使用 AWS Command Line Interface (AWS CLI) 查看有关您已完成的维护时段任务的详细信息。

如果您直接从 教程：使用 AWS CLI 创建和配置维护时段 继续，请确保您已为维护时段留出足够时间至少运行一次，以查看其执行结果。

在按照本教程中的步骤操作时，请将以斜体*##*文本显示的值替换为您自己的选项和 ID。例如，请将维护时段 ID *mw-0c50858d01EXAMPLE* 和实例 ID *i-02573cafcfEXAMPLE* 替换为您创建的资源的 ID。

要查看有关任务和使用 AWS CLI 执行任务的信息

1. 运行以下命令，查看特定维护时段的任务执行列表。

Linux & macOS

```
aws ssm describe-maintenance-window-executions \
    --window-id "mw-0c50858d01EXAMPLE"
```

Windows

```
aws ssm describe-maintenance-window-executions ^
    --window-id "mw-0c50858d01EXAMPLE"
```

系统返回类似于以下内容的信息。

```
{
    "WindowExecutions": [
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "WindowExecutionId": "14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE",
            "Status": "SUCCESS",
            "StartTime": 1557593793.483,
            "EndTime": 1557593798.978
        },
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "WindowExecutionId": "791b72e0-f0da-4021-8b35-f95dfEXAMPLE",
            "Status": "SUCCESS",
            "StartTime": 1557593493.096,
            "EndTime": 1557593498.611
        },
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "WindowExecutionId": "ecec60fa-6bb0-4d26-98c7-140308EXAMPLE",
            "Status": "SUCCESS",
            "StatusDetails": "No tasks to execute.",
            "StartTime": 1557593193.309,
            "EndTime": 1557593193.334
        }
    ]
}
```

2. 运行以下命令，获取有关维护时段任务执行的信息。

Linux & macOS

```
aws ssm get-maintenance-window-execution \
    --window-execution-id "14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE"
```

Windows

```
aws ssm get-maintenance-window-execution ^
    --window-execution-id "14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE"
```

系统返回类似于以下内容的信息。

```
{
    "WindowExecutionId": "14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE",
    "TaskIds": [
        "c9b05aba-197f-4d8d-be34-e73fbEXAMPLE"
    ],
    "Status": "SUCCESS",
    "StartTime": 1557593493.096,
    "EndTime": 1557593498.611
}
```

3. 运行以下命令，列出作为维护时段执行的组成部分运行的任务。

Linux & macOS

```
aws ssm describe-maintenance-window-execution-tasks \
    --window-execution-id "14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE"
```

Windows

```
aws ssm describe-maintenance-window-execution-tasks ^
    --window-execution-id "14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE"
```

系统返回类似于以下内容的信息。

```
{
    "WindowExecutionTaskIdentities": [
```

```
        {
            "WindowExecutionId": "14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE",
            "TaskExecutionId": "c9b05aba-197f-4d8d-be34-e73fbEXAMPLE",
            "Status": "SUCCESS",
            "StartTime": 1557593493.162,
            "EndTime": 1557593498.57,
            "TaskArn": "AWS-RunShellScript",
            "TaskType": "RUN_COMMAND"
        }
    ]
}
```

4. 运行以下命令，获取有关任务执行的详情。

Linux & macOS

```
aws ssm get-maintenance-window-execution-task \
    --window-execution-id "14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE" \
    --task-id "c9b05aba-197f-4d8d-be34-e73fbEXAMPLE"
```

Windows

```
aws ssm get-maintenance-window-execution-task ^
    --window-execution-id "14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE" ^
    --task-id "c9b05aba-197f-4d8d-be34-e73fbEXAMPLE"
```

系统返回类似于以下内容的信息。

```
{
    "WindowExecutionId": "14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE",
    "TaskExecutionId": "c9b05aba-197f-4d8d-be34-e73fbEXAMPLE",
    "TaskArn": "AWS-RunShellScript",
    "ServiceRole": "arn:aws:iam::111122223333:role/MyMaintenanceWindowServiceRole",
    "Type": "RUN_COMMAND",
    "TaskParameters": [
        {
            "aws:InstanceId": {
                "Values": [
                    "i-02573cafcfEXAMPLE"
                ]
            },
```

```
            "commands": {
                "Values": [
                    "df"
                ]
            }
        }
    ],
    "Priority": 10,
    "MaxConcurrency": "1",
    "MaxErrors": "1",
    "Status": "SUCCESS",
    "StartTime": 1557593493.162,
    "EndTime": 1557593498.57
}
```

5. 运行以下命令，获取执行某个任务时执行的具体任务调用。

   Linux & macOS

   ```
   aws ssm describe-maintenance-window-execution-task-invocations \
       --window-execution-id "14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE" \
       --task-id "c9b05aba-197f-4d8d-be34-e73fbEXAMPLE"
   ```

   Windows

   ```
   aws ssm describe-maintenance-window-execution-task-invocations ^
       --window-execution-id "14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE" ^
       --task-id "c9b05aba-197f-4d8d-be34-e73fbEXAMPLE"
   ```

   系统返回类似于以下内容的信息。

   ```
   {
       "WindowExecutionTaskInvocationIdentities": [
           {
               "WindowExecutionId": "14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE",
               "TaskExecutionId": "c9b05aba-197f-4d8d-be34-e73fbEXAMPLE",
               "InvocationId": "c336d2ab-09de-44ba-8f6a-6136cEXAMPLE",
               "ExecutionId": "76a5a04f-caf6-490c-b448-92c02EXAMPLE",
               "TaskType": "RUN_COMMAND",
   ```

```
            "Parameters": "{\"documentName\":\"AWS-RunShellScript\",\"instanceIds
\":[\"i-02573cafcfEXAMPLE\"],\"maxConcurrency\":\"1\",\"maxErrors\":\"1\",
\"parameters\":{\"commands\":[\"df\"]}}",
            "Status": "SUCCESS",
            "StatusDetails": "Success",
            "StartTime": 1557593493.222,
            "EndTime": 1557593498.466
        }
    ]
}
```

教程：使用 AWS CLI 更新维护时段

本教程演示了如何使用 AWS Command Line Interface (AWS CLI) 更新维护时段。它还向您演示了如何更新不同的任务类型，包括 AWS Systems Manager Run Command、自动化、AWS Lambda 和 AWS Step Functions 任务类型。

本节中的示例使用以下 Systems Manager 操作来更新维护时段。

- UpdateMaintenanceWindow
- UpdateMaintenanceWindowTarget
- UpdateMaintenanceWindowTask
- DeregisterTargetFromMaintenanceWindow

有关使用 Systems Manager 控制台来更新维护时段的信息，请参阅 使用控制台更新或删除维护时段资源。

在按照本教程中的步骤操作时，请将以斜体*##*文本显示的值替换为您自己的选项和 ID。例如，请将维护时段 ID *mw-0c50858d01EXAMPLE* 和实例 ID *i-02573cafcfEXAMPLE* 替换为您创建的资源的 ID。

要使用 AWS CLI 更新维护时段

1. 打开 AWS CLI 并运行以下命令将目标更新为包括名称和描述。

   Linux & macOS

   ```
   aws ssm update-maintenance-window-target \
       --window-id "mw-0c50858d01EXAMPLE" \
   ```

```
    --window-target-id "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE" \
    --name "My-Maintenance-Window-Target" \
    --description "Description for my maintenance window target"
```

Windows

```
aws ssm update-maintenance-window-target ^
    --window-id "mw-0c50858d01EXAMPLE" ^
    --window-target-id "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE" ^
    --name "My-Maintenance-Window-Target" ^
    --description "Description for my maintenance window target"
```

系统返回类似于以下内容的信息。

```
{
    "WindowId": "mw-0c50858d01EXAMPLE",
    "WindowTargetId": "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE",
    "Targets": [
        {
            "Key": "InstanceIds",
            "Values": [
                "i-02573cafcfEXAMPLE"
            ]
        }
    ],
    "Name": "My-Maintenance-Window-Target",
    "Description": "Description for my maintenance window target"
}
```

2. 运行以下命令，使用 `replace` 选项来删除描述字段和添加更多目标。描述字段被删除，因为更新不包含该字段（空值）。请务必指定已配置为用于 Systems Manager 的其他节点。

Linux & macOS

```
aws ssm update-maintenance-window-target \
    --window-id "mw-0c50858d01EXAMPLE" \
    --window-target-id "d208dedf-3f6b-41ff-ace8-8e751EXAMPLE" \
    --targets "Key=InstanceIds,Values=i-02573cafcfEXAMPLE,i-0471e04240EXAMPLE" \
    --name "My-Maintenance-Window-Target" \
    --replace
```

Windows

```
aws ssm update-maintenance-window-target ^
    --window-id "mw-0c50858d01EXAMPLE" ^
    --window-target-id "d208dedf-3f6b-41ff-ace8-8e751EXAMPLE" ^
    --targets "Key=InstanceIds,Values=i-02573cafcfEXAMPLE,i-0471e04240EXAMPLE" ^
    --name "My-Maintenance-Window-Target" ^
    --replace
```

系统返回类似于以下内容的信息。

```
{
    "WindowId": "mw-0c50858d01EXAMPLE",
    "WindowTargetId": "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE",
    "Targets": [
        {
            "Key": "InstanceIds",
            "Values": [
                "i-02573cafcfEXAMPLE",
                "i-0471e04240EXAMPLE"
            ]
        }
    ],
    "Name": "My-Maintenance-Window-Target"
}
```

3. `start-date` 选项用于将维护时段的激活时间延迟到一个指定的未来日期。`end-date` 选项用于设置在某个未来的日期和时间后不再运行维护时段。请以 ISO-8601 扩展格式指定这些选项。

   运行以下命令可为定期执行的维护时段指定日期和时间范围。

   Linux & macOS

   ```
   aws ssm update-maintenance-window \
       --window-id "mw-0c50858d01EXAMPLE" \
       --start-date "2020-10-01T10:10:10Z" \
       --end-date "2020-11-01T10:10:10Z"
   ```

Windows

```
aws ssm update-maintenance-window ^
    --window-id "mw-0c50858d01EXAMPLE" ^
    --start-date "2020-10-01T10:10:10Z" ^
    --end-date "2020-11-01T10:10:10Z"
```

4. 运行以下命令可更新 Run Command 任务。

> **Tip**
>
> 如果您的目标是 Windows Server 的 Amazon Elastic Compute Cloud (Amazon EC2) 实例，请将以下命令中的 `df` 更改为 `ipconfig`，并将 `AWS-RunShellScript` 更改为 `AWS-RunPowerShellScript`。

Linux & macOS

```
aws ssm update-maintenance-window-task \
    --window-id "mw-0c50858d01EXAMPLE" \
    --window-task-id "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE" \
    --targets "Key=WindowTargetIds,Values=e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
 \
    --task-arn "AWS-RunShellScript" \
    --service-role-arn "arn:aws:iam::account-id:role/MaintenanceWindowsRole" \
    --task-invocation-parameters "RunCommand={Comment=Revising my Run Command
 task,Parameters={commands=df}}" \
    --priority 1 --max-concurrency 10 --max-errors 4 \
    --name "My-Task-Name" --description "A description for my Run Command task"
```

Windows

```
aws ssm update-maintenance-window-task ^
    --window-id "mw-0c50858d01EXAMPLE" ^
    --window-task-id "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE" ^
    --targets "Key=WindowTargetIds,Values=e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
 ^
    --task-arn "AWS-RunShellScript" ^
    --service-role-arn "arn:aws:iam::account-id:role/MaintenanceWindowsRole" ^
    --task-invocation-parameters "RunCommand={Comment=Revising my Run Command
 task,Parameters={commands=df}}" ^
```

```
    --priority 1 --max-concurrency 10 --max-errors 4 ^
    --name "My-Task-Name" --description "A description for my Run Command task"
```

系统返回类似于以下内容的信息。

```
{
    "WindowId": "mw-0c50858d01EXAMPLE",
    "WindowTaskId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE",
    "Targets": [
        {
            "Key": "WindowTargetIds",
            "Values": [
                "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
            ]
        }
    ],
    "TaskArn": "AWS-RunShellScript",
    "ServiceRoleArn": "arn:aws:iam::111122223333:role/MaintenanceWindowsRole",
    "TaskParameters": {},
    "TaskInvocationParameters": {
        "RunCommand": {
            "Comment": "Revising my Run Command task",
            "Parameters": {
                "commands": [
                    "df"
                ]
            }
        }
    },
    "Priority": 1,
    "MaxConcurrency": "10",
    "MaxErrors": "4",
    "Name": "My-Task-Name",
    "Description": "A description for my Run Command task"
}
```

5. 改写并运行以下命令以更新 Lambda 任务。

Linux & macOS

```
aws ssm update-maintenance-window-task \
    --window-id mw-0c50858d01EXAMPLE \
```

```
    --window-task-id 4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE \
    --targets "Key=WindowTargetIds,Values=e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
 \
    --task-arn "arn:aws:lambda:region:111122223333:function:SSMTestLambda" \
    --service-role-arn "arn:aws:iam:account-id:role/MaintenanceWindowsRole" \
    --task-invocation-parameters '{"Lambda":{"Payload":"{\"InstanceId\":
\"{{RESOURCE_ID}}\",\"targetType\":\"{{TARGET_TYPE}}\"}"}}' \
    --priority 1 --max-concurrency 10 --max-errors 5 \
    --name "New-Lambda-Task-Name" \
    --description "A description for my Lambda task"
```

Windows

```
aws ssm update-maintenance-window-task ^
    --window-id mw-0c50858d01EXAMPLE ^
    --window-task-id 4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE ^
    --targets "Key=WindowTargetIds,Values=e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
 ^
    --task-arn --task-arn
 "arn:aws:lambda:region:111122223333:function:SSMTestLambda" ^
    --service-role-arn "arn:aws:iam:account-id:role/MaintenanceWindowsRole" ^
    --task-invocation-parameters '{"Lambda":{"Payload":"{\"InstanceId\":
\"{{RESOURCE_ID}}\",\"targetType\":\"{{TARGET_TYPE}}\"}"}}' ^
    --priority 1 --max-concurrency 10 --max-errors 5 ^
    --name "New-Lambda-Task-Name" ^
    --description "A description for my Lambda task"
```

系统返回类似于以下内容的信息。

```
{
    "WindowId": "mw-0c50858d01EXAMPLE",
    "WindowTaskId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE",
    "Targets": [
        {
            "Key": "WindowTargetIds",
            "Values": "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
        }
    ],
    "TaskArn": "arn:aws:lambda:us-east-2:111122223333:function:SSMTestLambda",
    "ServiceRoleArn": "arn:aws:iam::111122223333:role/MaintenanceWindowsRole",
    "TaskParameters": {},
    "TaskInvocationParameters": {
```

```
        "Lambda": {
            "Payload": "e30="
        }
    },
    "Priority": 1,
    "MaxConcurrency": "10",
    "MaxErrors": "5",
    "Name": "New-Lambda-Task-Name",
    "Description": "A description for my Lambda task"
}
```

6. 如果您要更新 Step Functions 任务，请改写并运行以下命令，以更新其 task-invocation-parameters。

   Linux & macOS

```
aws ssm update-maintenance-window-task \
    --window-id "mw-0c50858d01EXAMPLE" \
    --window-task-id "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE" \
    --targets "Key=WindowTargetIds,Values=e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
 \
    --task-arn "arn:aws:states:region:execution:SSMStepFunctionTest" \
    --service-role-arn "arn:aws:iam:account-id:role/MaintenanceWindowsRole" \
    --task-invocation-parameters '{"StepFunctions":{"Input":"{\"InstanceId\":
\"{{RESOURCE_ID}}\"}"}}' \
    --priority 0 --max-concurrency 10 --max-errors 5 \
    --name "My-Step-Functions-Task" \
    --description "A description for my Step Functions task"
```

   Windows

```
aws ssm update-maintenance-window-task ^
    --window-id "mw-0c50858d01EXAMPLE" ^
    --window-task-id "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE" ^
    --targets "Key=WindowTargetIds,Values=e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
 ^
    --task-arn "arn:aws:states:region:execution:SSMStepFunctionTest" ^
    --service-role-arn "arn:aws:iam:account-id:role/MaintenanceWindowsRole" ^
    --task-invocation-parameters '{"StepFunctions":{"Input":"{\"InstanceId\":
\"{{RESOURCE_ID}}\"}"}}' ^
    --priority 0 --max-concurrency 10 --max-errors 5 ^
    --name "My-Step-Functions-Task" ^
```

```
    --description "A description for my Step Functions task"
```

系统返回类似于以下内容的信息。

```
{
    "WindowId": "mw-0c50858d01EXAMPLE",
    "WindowTaskId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE",
    "Targets": [
        {
            "Key": "WindowTargetIds",
            "Values": [
                "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
            ]
        }
    ],
    "TaskArn": "arn:aws:states:us-
east-2:111122223333:execution:SSMStepFunctionTest",
    "ServiceRoleArn": "arn:aws:iam::111122223333:role/MaintenanceWindowsRole",
    "TaskParameters": {},
    "TaskInvocationParameters": {
        "StepFunctions": {
            "Input": "{\"instanceId\":\"{{RESOURCE_ID}}\"}"
        }
    },
    "Priority": 0,
    "MaxConcurrency": "10",
    "MaxErrors": "5",
    "Name": "My-Step-Functions-Task",
    "Description": "A description for my Step Functions task"
}
```

7. 运行以下命令可从维护时段中注销目标。本示例使用 `safe` 参数确定目标是否被任何任务引用，从而确定注销是否安全。

   Linux & macOS

```
aws ssm deregister-target-from-maintenance-window \
    --window-id "mw-0c50858d01EXAMPLE" \
    --window-target-id "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE" \
    --safe
```

Windows

```
aws ssm deregister-target-from-maintenance-window ^
    --window-id "mw-0c50858d01EXAMPLE" ^
    --window-target-id "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE" ^
    --safe
```

系统返回类似于以下内容的信息。

```
An error occurred (TargetInUseException) when calling the
 DeregisterTargetFromMaintenanceWindow operation:
This Target cannot be deregistered because it is still referenced in Task:
 4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE
```

8. 运行以下命令可从维护时段注销目标，即使该目标已被某个任务引用。您可以使用 no-safe 参数强制执行注销操作。

Linux & macOS

```
aws ssm deregister-target-from-maintenance-window \
    --window-id "mw-0c50858d01EXAMPLE" \
    --window-target-id "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE" \
    --no-safe
```

Windows

```
aws ssm deregister-target-from-maintenance-window ^
    --window-id "mw-0c50858d01EXAMPLE" ^
    --window-target-id "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE" ^
    --no-safe
```

系统返回类似于以下内容的信息。

```
{
    "WindowId": "mw-0c50858d01EXAMPLE",
    "WindowTargetId": "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
}
```

9. 运行以下命令可更新 Run Command 任务。此示例使用名为 `UpdateLevel` 的 Systems Manager Parameter Store 参数，其格式如下所示：“`{{ssm:UpdateLevel}}`”

   Linux & macOS

   ```
   aws ssm update-maintenance-window-task \
       --window-id "mw-0c50858d01EXAMPLE" \
       --window-task-id "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE" \
       --targets "Key=InstanceIds,Values=i-02573cafcfEXAMPLE"  \
       --task-invocation-parameters "RunCommand={Comment=A comment for my task
    update,Parameters={UpdateLevel='{{ssm:UpdateLevel}}'}}"
   ```

   Windows

   ```
   aws ssm update-maintenance-window-task ^
       --window-id "mw-0c50858d01EXAMPLE" ^
       --window-task-id "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE" ^
       --targets "Key=InstanceIds,Values=i-02573cafcfEXAMPLE"  ^
       --task-invocation-parameters "RunCommand={Comment=A comment for my task
    update,Parameters={UpdateLevel='{{ssm:UpdateLevel}}'}}"
   ```

   系统返回类似于以下内容的信息。

   ```
   {
       "WindowId": "mw-0c50858d01EXAMPLE",
       "WindowTaskId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE",
       "Targets": [
           {
               "Key": "InstanceIds",
               "Values": [
                   "i-02573cafcfEXAMPLE"
               ]
           }
       ],
       "TaskArn": "AWS-RunShellScript",
       "ServiceRoleArn": "arn:aws:iam::111122223333:role/
   MyMaintenanceWindowServiceRole",
       "TaskParameters": {},
       "TaskInvocationParameters": {
           "RunCommand": {
               "Comment": "A comment for my task update",
   ```

```
            "Parameters": {
                "UpdateLevel": [
                    "{{ssm:UpdateLevel}}"
                ]
            }
        }
    },
    "Priority": 10,
    "MaxConcurrency": "1",
    "MaxErrors": "1"
}
```

10. 运行以下命令可更新 自动化 任务，为 task-invocation-parameters 参数指定 WINDOW_ID 和 WINDOW_TASK_ID 参数：

    Linux & macOS

    ```
    aws ssm update-maintenance-window-task \
        --window-id "mw-0c50858d01EXAMPLE" \
        --window-task-id "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE" \
        --targets "Key=WindowTargetIds,Values=e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE \
        --task-arn "AutoTestDoc" \
        --service-role-arn "arn:aws:iam:account-id:role/
    MyMaintenanceWindowServiceRole \
        --task-invocation-parameters
     "Automation={Parameters={InstanceId='{{RESOURCE_ID}}',initiator='{{WINDOW_ID}}.Task-
    {{WINDOW_TASK_ID}}'}}" \
        --priority 3 --max-concurrency 10 --max-errors 5
    ```

    Windows

    ```
    aws ssm update-maintenance-window-task ^
        --window-id "mw-0c50858d01EXAMPLE" ^
        --window-task-id "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE" ^
        --targets "Key=WindowTargetIds,Values=e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE ^
        --task-arn "AutoTestDoc" ^
        --service-role-arn "arn:aws:iam:account-id:role/
    MyMaintenanceWindowServiceRole ^
        --task-invocation-parameters
     "Automation={Parameters={InstanceId='{{RESOURCE_ID}}',initiator='{{WINDOW_ID}}.Task-
    {{WINDOW_TASK_ID}}'}}" ^
        --priority 3 --max-concurrency 10 --max-errors 5
    ```

系统返回类似于以下内容的信息。

```
{
    "WindowId": "mw-0c50858d01EXAMPLE",
    "WindowTaskId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE",
    "Targets": [
        {
            "Key": "WindowTargetIds",
            "Values": [
                "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
            ]
        }
    ],
    "TaskArn": "AutoTestDoc",
    "ServiceRoleArn": "arn:aws:iam::111122223333:role/
MyMaintenanceWindowServiceRole",
    "TaskParameters": {},
    "TaskInvocationParameters": {
        "Automation": {
            "Parameters": {
                "multi": [
                    "{{WINDOW_TASK_ID}}"
                ],
                "single": [
                    "{{WINDOW_ID}}"
                ]
            }
        }
    },
    "Priority": 0,
    "MaxConcurrency": "10",
    "MaxErrors": "5",
    "Name": "My-Automation-Task",
    "Description": "A description for my Automation task"
}
```

教程：使用 AWS CLI 删除维护时段

要删除您在这些教程中创建的维护时段，请运行以下命令。

```
aws ssm delete-maintenance-window --window-id "mw-0c50858d01EXAMPLE"
```

系统返回类似于以下内容的信息。

```
{
   "WindowId":"mw-0c50858d01EXAMPLE"
}
```

教程：使用控制台创建修补的维护时段

> **Important**
>
> 您可以继续使用此早期主题来创建用于修补的维护时段。但是，我们建议您使用补丁策略。有关更多信息，请参阅Quick Setup 中的补丁策略配置和使用 Quick Setup 补丁策略为组织中的实例配置修补。

为了最大程度减少对服务器可用性的影响，我们建议您将维护时段配置为在不中断业务运营的时间运行修补。

您必须先为 Maintenance Windows（AWS Systems Manager 中的一项工具）配置角色和权限，然后才能开始此过程。有关更多信息，请参阅 设置 Maintenance Windows。

创建维护时段以进行修补

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Maintenance Windows。
3. 选择 Create maintenance window。
4. 在名称字段中输入一个名称，将其指定为用于修补关键更新和重要更新的维护时段。
5. （可选）对于描述，输入描述。
6. 如果您希望即便没有将托管式节点注册为目标，也允许维护时段任务在这些节点上运行，则选择 Allow unregistered targets（允许未注册的目标）。

   如果选择了此选项，您在将任务注册到维护时段时便可以选择已注销节点（按节点 ID）。

   如果未选择此选项，您在将任务注册到维护时段时就必须选择之前注册的目标。

7. 在 Schedule (计划) 部分的顶部，通过使用三个计划选项之一来为维护时段指定计划。

   有关构建 cron/rate 表达式的信息，请参阅 参考：适用于 Systems Manager 的 Cron 和 Rate 表达式。

8. 对于 Duration (持续时间)，输入维护时段将运行的小时数。您指定的值根据维护时段的开始时间确定维护时段的具体结束时间。在最终结束时间减去您在下一步中为 Stop initiating tasks (停止启动任务) 指定的小时数后，不允许启动维护时段任务。

   例如，如果维护时段从下午 3 点开始，持续时间是 3 个小时，并且 Stop initiating tasks (停止启动任务) 值是 1 个小时，则下午 5 点之后将无法启动维护时段任务。

9. 在停止启动任务中，输入系统应该在维护时段结束前几小时停止计划要运行的新任务。

10. （可选）对于 Window start date（时段开始日期），请以 ISO-8601 扩展格式指定您希望维护时段变为活动状态的日期和时间。这让您可以将维护时段的激活时间推迟到指定的将来日期。

11. （可选）对于 Window end date（时段结束日期），请以 ISO-8601 扩展格式指定您希望维护时段变为不活动状态的日期和时间。这样可以设置在某个未来的日期和时间后不再运行维护时段。

12. （可选）对于计划时区，请以互联网编号分配机构（IANA）格式指定时区，计划的维护时段执行将基于该时区。例如：“America/Los_Angeles”、“etc/UTC”或“Asia/Seoul”。

    有关有效格式的更多信息，请参阅 IANA 网站上的 Time Zone Database。

13. （可选）在管理标签区域，将一个或多个标签键名称/值对应用到维护时段。

    标签是您分配给资源的可选元数据。标签可让您按不同的方式（如用途、拥有者或环境）对资源进行分类。例如，您可能需要标记此维护时段来标识它运行的任务的类型。在这种情况下，您可以指定以下键名/键值对：

    - `Key=TaskType,Value=Patching`

14. 选择 Create maintenance window。

15. 在维护时段列表中，请选择您刚刚创建的维护时段，然后选择操作和注册目标。

16. (可选) 在 Maintenance window target details 部分中，为此目标提供名称、描述和所有者信息 (您的姓名或别名)。

17. 对于目标选择，选择指定实例标签。

18. 对于指定实例标签，输入标签键和标签值来标识要注册到维护时段的节点，然后选择添加。

19. 选择注册目标。系统会创建维护时段目标。

20. 在创建的维护时段的详细信息页面中，选择 Actions (操作) 和 Register Run command task (注册 Run Command 任务)。

21. （可选）对于 Maintenance window task details (维护时段任务详细信息)，为此任务提供名称和描述。
22. 对于 Command document (命令文档)，选择 `AWS-RunPatchBaseline`。
23. 对于 Task priority (任务优先级)，请选择一个优先级。零 (0) 表示最高优先级。
24. 对于目标，在定位方式 下，选择在此过程中先前创建的维护时段目标。
25. 对于 Rate control（速率控制）：
    - 对于 Concurrency（并发），请指定要同时运行该命令的托管式节点的数量或百分比。

      > **Note**
      >
      > 如果您通过指定应用于托管式节点的标签或指定 AWS Resource Groups 来选择目标，但不确定有多少个托管式节点已被设为目标，则可通过指定百分比来限制可同时运行该文档的目标的数量。

    - 对于 Error threshold（错误阈值），请指定当命令在一定数量或百分比的节点上失败后，何时在其他托管式节点上停止运行该命令。例如，如果您指定三个错误，Systems Manager 将在收到第四个错误时停止发送该命令。仍在处理该命令的托管式节点也可能发送错误。
26. （可选）对于 IAM 服务角色，选择一个角色以向 Systems Manager 提供运行维护时段任务时所承担的权限。

    如果您未指定服务角色 ARN，Systems Manager 将使用您账户中的服务相关角色。如果您的账户中没有适用于 Systems Manager 的适当服务相关角色，则将在成功注册任务后创建该角色。

    > **Note**
    >
    > 为了改善安全状况，强烈建议您创建自定义策略和自定义服务角色来运行维护时段任务。可以精心设计该策略，只提供特定维护时段任务所需的权限。有关更多信息，请参阅 设置 Maintenance Windows。

27. （可选）对于 Output options (输出选项)，要将命令输出保存到文件，请选中 Enable writing output to S3 (启用将输出写入 S3) 方框。在方框中输入存储桶和前缀（文件夹）名称。

    > **Note**
    >
    > 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给托管式节点的实例配置文件的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems

Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确认与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

要将输出流式传输到 Amazon CloudWatch Logs 日志组，请选择 CloudWatch output (CloudWatch 输出) 框。在该框中输入日志组名称。

28. 在 SNS notifications (SNS 通知) 部分中，如果希望发送有关命令执行状态的通知，请选中 Enable SNS notifications (启用 SNS 通知) 复选框。

    有关为 Run Command 配置 Amazon SNS 通知的更多信息，请参阅 使用 Amazon SNS 通知监控 Systems Manager 状态更改。

29. 对于 Parameters (参数)：

    - 对于 Operation (操作)，选择 Scan (扫描) 扫描缺失的补丁，或选择 Install (安装) 扫描并安装缺失的补丁。
    - 您不需要在 Snapshot Id (快照 ID) 字段中输入任何值。此系统将自动生成并提供此参数。
    - 除非您希望 Patch Manager 使用与为补丁基准指定的补丁集不同的补丁集，否则无需在 Install Override List (安装覆盖列表) 字段中输入任何内容。有关信息，请参阅参数名称: `InstallOverrideList`。
    - 对于重启选项，请指定您是希望在 `Install` 操作期间安装补丁时，还是在 Patch Manager 检测到自上次重启节点以来安装的其他补丁时重启节点。有关信息，请参阅参数名称: `RebootOption`。
    - （可选）对于 Comment (注释)，输入有关此命令的跟踪备注或提醒。
    - 对于 Timeout (seconds) (超时（秒）)，输入系统在认为操作失败前等待操作完成的秒数。

30. 选择注册运行命令任务。

维护时段任务完成后，您可以通过 Systems Manager 控制台中的 Fleet Manager 工具查看补丁合规性详细信息。

您还可以在合规性报告选项卡上的Patch Manager工具中查看合规性信息。

您也可以使用 DescribePatchGroupState 和 DescribeInstancePatchStatesForPatchGroup API 来查看合规性详细信息。有关补丁合规性数据的信息，请参阅 关于补丁合规性。

使用维护时段修补计划

配置补丁基准（可以选择配置补丁组）后，您可以使用维护时段将补丁应用于您的节点。维护时段通过让您指定在不中断业务运营的时间执行修补流程，可以减少对服务器可用性的影响。维护时段的操作如下：

1. 创建带修补操作计划的维护时段。
2. 通过为标签名称指定 `Patch Group` 或 `PatchGroup` 标签，并指定已定义 Amazon Elastic Compute Cloud (Amazon EC2) 标签的任意值（例如，“web 服务器”或“US-EAST-PROD”），来选择维护时段的目标。（如果在 EC2 实例元数据中允许使用标签，则必须使用 `PatchGroup`，且不能使用空格。
3. 创建新的维护时段任务，并指定 `AWS-RunPatchBaseline` 文档。

在配置任务时，可以选择扫描节点，也可以选择扫描并安装节点上的补丁。如果选择扫描节点，则 Patch Manager（AWS Systems Manager 中的一项工具）会扫描每个节点，并生成缺失补丁的列表供您审核。

如果选择扫描和安装补丁，Patch Manager 将扫描每个节点，并将已安装补丁的列表与基准中已批准补丁的列表进行比较。Patch Manager 将标识缺失的补丁，然后下载并安装所有缺失和批准的补丁。

如果希望执行一次性扫描或安装来修复问题，可以使用 Run Command 直接调用 `AWS-RunPatchBaseline` 文档。

> **Important**
>
> 在安装补丁后，Systems Manager 会重启每个节点。需要重启才能确保正确安装补丁，并确保系统不会使节点处于潜在的不良状态。（例外：如果将 `AWS-RunPatchBaseline` 文档中的 `RebootOption` 参数设置为 `NoReboot`，则在 Patch Manager 运行后不会重启托管式节点。有关更多信息，请参阅 参数名称: `RebootOption`。）

注册维护时段任务时使用伪参数

当您在 Maintenance Windows（AWS Systems Manager 中的一项工具）中注册任务时，可以指定分别对于四种任务类型唯一的参数。（在 CLI 命令中，这些参数是使用 `--task-invocation-parameters` 选项提供的。）

您还可以使用伪参数 语法引用特定值，例如 {{RESOURCE_ID}}、{{TARGET_TYPE}} 和 {{WINDOW_TARGET_ID}}。维护时段任务运行时，它传递正确的值而不是伪参数占位符。本主题后面的 支持的伪参数 中提供了您可以使用的伪参数的完整列表。

> **Important**
>
> 对于目标类型 RESOURCE_GROUP，根据任务所需的 ID 格式，您可以在任务运行时选择是使用 {{TARGET_ID}} 还是使用 {{RESOURCE_ID}} 来引用资源。{{TARGET_ID}} 返回资源的完整 ARN。{{RESOURCE_ID}} 仅返回资源的较短名称或 ID，如以下示例所示。
>
> - {{TARGET_ID}} 格式：arn:aws:ec2:us-east-1:123456789012:instance/i-02573cafcfEXAMPLE
> - {{RESOURCE_ID}} 格式：i-02573cafcfEXAMPLE
>
> 对于目标类型 INSTANCE，{{TARGET_ID}} 和 {{RESOURCE_ID}} 参数两者仅生成实例 ID。有关更多信息，请参阅 支持的伪参数。
> {{TARGET_ID}} 和 {{RESOURCE_ID}} 可用于将 AWS 资源的 ID 仅传递到 自动化、Lambda 和 Step Functions 任务。这两个伪参数不能用于 Run Command 任务。

伪参数示例

假设 AWS Lambda 任务的有效负载需要通过其 ID 引用实例。

无论您使用的是 INSTANCE 维护时段目标，还是 RESOURCE_GROUP 维护时段目标，这都可以通过使用 {{RESOURCE_ID}} 虚拟参数来实现。例如：

```
"TaskArn": "arn:aws:lambda:us-east-2:111122223333:function:SSMTestFunction",
    "TaskType": "LAMBDA",
    "TaskInvocationParameters": {
        "Lambda": {
            "ClientContext": "ew0KICAi--truncated--0KIEXAMPLE",
            "Payload": "{ \"instanceId\": \"{{RESOURCE_ID}}\" }",
            "Qualifier": "$LATEST"
        }
    }
```

如果您的 Lambda 任务旨在针对 Amazon Elastic Compute Cloud (Amazon EC2) 实例之外的其他受支持的目标类型运行（如 Amazon DynamoDB 表），则可以使用相同的语法，并且 {{RESOURCE_ID}} 仅生成该表的名称。但是，如果您需要表的完整 ARN，请使用 {{TARGET_ID}}，如以下示例所示。

```
"TaskArn": "arn:aws:lambda:us-east-2:111122223333:function:SSMTestFunction",
    "TaskType": "LAMBDA",
    "TaskInvocationParameters": {
        "Lambda": {
            "ClientContext": "ew0KICAi--truncated--0KIEXAMPLE",
            "Payload": "{ \"tableArn\": \"{{TARGET_ID}}\" }",
            "Qualifier": "$LATEST"
        }
    }
```

相同的语法适用于目标实例或其他资源类型。将多个资源类型添加到资源组时，任务将针对每个适当的资源运行。

> **Important**
>
> 并非所有可能包含在资源组中的资源类型都为 {{RESOURCE_ID}} 参数生成值。有关受支持的资源类型的列表，请参阅 支持的伪参数。

另一个例子是，要运行停止您的 Amazon EC2 实例的 自动化 任务，请指定 AWS-StopEC2Instance Systems Manager 文档（SSM 文档）作为 TaskArn 值，并使用 {{RESOURCE_ID}} 伪参数：

```
"TaskArn": "AWS-StopEC2Instance",
    "TaskType": "AUTOMATION"
    "TaskInvocationParameters": {
        "Automation": {
            "DocumentVersion": "1",
            "Parameters": {
                "instanceId": [
                    "{{RESOURCE_ID}}"
                ]
            }
        }
    }
```

要运行复制 Amazon Elastic Block Store (Amazon EBS) 卷的快照的 自动化 任务，请将 AWS-CopySnapshot SSM 文档指定为 TaskArn 值，并使用 {{RESOURCE_ID}} 伪参数。

```
"TaskArn": "AWS-CopySnapshot",
    "TaskType": "AUTOMATION"
    "TaskInvocationParameters": {
        "Automation": {
            "DocumentVersion": "1",
            "Parameters": {
                "SourceRegion": "us-east-2",
                "targetType":"RESOURCE_GROUP",
                "SnapshotId": [
                    "{{RESOURCE_ID}}"
                ]
            }
        }
    }
```

支持的伪参数

以下列表描述了您可在 `--task-invocation-parameters` 选项中使用 {{*PSEUDO_PARAMETER*}} 语法指定的伪参数。

- **WINDOW_ID**：目标维护时段的 ID。
- **WINDOW_TASK_ID**：正在运行的时段任务的 ID。
- **WINDOW_TARGET_ID**：包含目标的时段目标的 ID (目标 ID)。
- **WINDOW_EXECUTION_ID**：当前时段执行的 ID。
- **TASK_EXECUTION_ID**：当前任务执行的 ID。
- **INVOCATION_ID**：当前调用的 ID。
- **TARGET_TYPE**：目标的类型。支持的类型包括 RESOURCE_GROUP 和 INSTANCE。
- **TARGET_ID**:

  如果您指定的目标类型为 INSTANCE，则 TARGET_ID 伪参数将被替换为实例的 ID。例如，i-078a280217EXAMPLE。

  如果您指定的目标类型为 RESOURCE_GROUP，则为任务执行引用的值将是资源的完整 ARN。例如：arn:aws:ec2:us-east-1:123456789012:instance/i-078a280217EXAMPLE。下表提供资源组中特定资源类型的示例 TARGET_ID 值。

> **Note**
>
> 对于 Run Command 任务，TARGET_ID 不受支持。

资源类型	示例 TARGET_ID	
AWS::CloudWatch::Alarm	arn:aws:cloudwatch:us-east-1:123456789012:alarm:MyCloudWatchAlarm i-078a280217EXAMPLE	
AWS::DynamoDB::Table	arn:aws:dynamodb:us-east-1:123456789012:table/MyTable	
AWS::EC2::Instance	arn:aws:ec2:us-east-1:123456789012:instance/ i-078a280217EXAMPLE	
AWS::EC2::Image	arn:aws:ec2:us-east-1:123456789012:image/ami-02250b3732EXAMPLE	
AWS::EC2::SecurityGroup	arn:aws:ec2:us-east-1:123456789012:security-group/sg-cEXAMPLE	
AWS::EC2::Snapshot	arn:aws:ec2:us-east-1:123456789012:snapshot/snap-03866bf003EXAMPLE	

资源类型	示例 TARGET_ID
AWS::EC2::Volume	arn:aws:ec2:us-east-1:123456789012:volume/vol-0912e04d78EXAMPLE
AWS::ECS::Service	arn:aws:ecs:us-east-1:123456789012:service/my-ecs-service
AWS::RDS::DBCluster	arn:aws:rds:us-east-2:123456789012:cluster:My-Cluster
AWS::RDS::DBInstance	arn:aws:rds:us-east-1:123456789012:db:My-SQL-Instance
AWS::S3::Bucket	arn:aws:s3:::amzn-s3-demo-bucket
AWS::SSM::ManagedInstance	arn:aws:ssm:us-east-1:123456789012:managed-instance/mi-0feadcf2d9EXAMPLE

- **RESOURCE_ID**：资源组中包含的资源类型的短 ID。下表提供资源组中特定资源类型的示例 RESOURCE_ID 值。

> **Note**
>
> 对于 Run Command 任务，RESOURCE_ID 不受支持。

资源类型	示例 RESOURCE_ID	
AWS::CloudWatch::Alarm	MyCloudWatchAlarm	
AWS::DynamoDB::Table	MyTable	
AWS::EC2::Instance	i-078a280217EXAMPLE	
AWS::EC2::Image	ami-02250b3732EXAMPLE	
AWS::EC2::SecurityGroup	sg-cEXAMPLE	
AWS::EC2::Snapshot	snap-03866bf003EXAMPLE	
AWS::EC2::Volume	vol-0912e04d78EXAMPLE	
AWS::ECS::Service	my-ecs-service	
AWS::RDS::DBCluster	My-Cluster	
AWS::RDS::DBInstance	My-SQL-Instance	
AWS::S3::Bucket	amzn-s3-demo-bucket	
AWS::SSM::ManagedInstance	mi-0feadcf2d9EXAMPLE	

Note

如果您指定的 AWS 资源组包含的资源类型不会生成 RESOURCE_ID 值，并且这些资源类型未在上表中列出，则不会填充 RESOURCE_ID 参数。对于该资源仍会发生执行调用。在这些情况下，请改用 TARGET_ID 伪参数，该参数将替换为资源的完整 ARN。

维护时段计划和活动期间选项

创建维护时段时，必须使用 Cron 或 Rate 表达式指定维护时段运行的频率。您可以选择性地指定维护时段依据其定期计划运行的日期范围，以及该定期计划所基于的时区。

但请注意，时区选项以及开始日期和结束日期选项彼此互不影响。指定的任何开始日期和结束日期时间（包含或不包含时区偏移量）仅确定维护时段定期运行的有效期间。时区选项确定维护时段计划在有效期间内 所基于的国际时区。

> **Note**
>
> 请以 ISO-8601 时间戳格式指定开始日期和结束日期。例如：`2021-04-07T14:29:00-08:00`
> 请以互联网编号分配机构 (IANA) 格式指定时区。例如：`America/Chicago`、`Europe/Berlin` 或 `Asia/Tokyo`。

示例

- 示例 1：指定维护时段的开始日期
- 示例 2：指定维护时段的开始日期和结束日期
- 示例 3：创建仅运行一次的维护时段
- 示例 4：指定维护时段的计划偏移天数

示例 1：指定维护时段的开始日期

假设您使用 AWS Command Line Interface (AWS CLI) 创建一个具有以下选项的维护时段：

- `--start-date 2021-01-01T00:00:00-08:00`
- `--schedule-timezone "America/Los_Angeles"`
- `--schedule "cron(0 09 ? * WED *)"`

例如：

Linux & macOS

```
aws ssm create-maintenance-window \
    --name "My-LAX-Maintenance-Window" \
    --allow-unassociated-targets \
```

```
    --duration 3 \
    --cutoff 1 \
    --start-date 2021-01-01T00:00:00-08:00 \
    --schedule-timezone "America/Los_Angeles" \
    --schedule "cron(0 09 ? * WED *)"
```

Windows

```
aws ssm create-maintenance-window ^
    --name "My-LAX-Maintenance-Window" ^
    --allow-unassociated-targets ^
    --duration 3 ^
    --cutoff 1 ^
    --start-date 2021-01-01T00:00:00-08:00 ^
    --schedule-timezone "America/Los_Angeles" ^
    --schedule "cron(0 09 ? * WED *)"
```

这意味着维护时段只有在到达其指定的开始日期和时间（即美国太平洋时间 2021 年 1 月 1 日星期五中午 12:00）之后 才会首次运行。（此时区比 UTC 时间晚 8 个小时。）在这种情况下，时段的开始日期和时间不表示维护时段首次运行的时间。总的来说，`--schedule-timezone` 和 `--schedule` 值意味着维护时段将在美国太平洋时区（以 IANA 格式“America/Los Angeles”表示）每个星期三的上午 9 点运行。允许时段内的首次执行时间将为美国太平洋时间 2021 年 1 月 4 日星期三上午 9 点。

示例 2：指定维护时段的开始日期和结束日期

假设您接下来创建一个具有以下选项的维护时段：

- `--start-date 2019-01-01T00:03:15+09:00`
- `--end-date 2019-06-30T00:06:15+09:00`
- `--schedule-timezone "Asia/Tokyo"`
- `--schedule "rate(7 days)"`

例如：

Linux & macOS

```
aws ssm create-maintenance-window \
    --name "My-NRT-Maintenance-Window" \
    --allow-unassociated-targets \
```

```
    --duration 3 \
    --cutoff 1 \
    --start-date 2019-01-01T00:03:15+09:00 \
    --end-date 2019-06-30T00:06:15+09:00 \
    --schedule-timezone "Asia/Tokyo" \
    --schedule "rate(7 days)"
```

Windows

```
aws ssm create-maintenance-window ^
    --name "My-NRT-Maintenance-Window" ^
    --allow-unassociated-targets ^
    --duration 3 ^
    --cutoff 1 ^
    --start-date 2019-01-01T00:03:15+09:00 ^
    --end-date 2019-06-30T00:06:15+09:00 ^
    --schedule-timezone "Asia/Tokyo" ^
    --schedule "rate(7 days)"
```

此维护时段的允许时段于日本标准时间 2019 年 1 月 1 日凌晨 3:15 开始。此维护时段的有效期间于日本标准时间 2019 年 6 月 30 日凌晨 6:15 结束。（此时区比 UTC 时间早 9 个小时。）总的来说，`--schedule-timezone` 和 `--schedule` 值意味着维护时段将在日本标准时区（以 IANA 格式“Asia/Tokyo”表示）每个星期二的凌晨 3:15 运行。这是因为维护时段每七天运行一次，并且于 1 月 1 日星期二凌晨 3:15 变为活动状态。最后一次执行时间为日本标准时间 2019 年 6 月 25 日星期二凌晨 3:15。这是允许的维护时段期限在五天后结束前的最后一个星期二。

示例 3：创建仅运行一次的维护时段

现在，您可以创建具有此选项的维护时段：

- `--schedule "at(2020-07-07T15:55:00)"`

例如：

Linux & macOS

```
aws ssm create-maintenance-window \
    --name "My-One-Time-Maintenance-Window" \
    --schedule "at(2020-07-07T15:55:00)" \
    --duration 5 \
```

```
    --cutoff 2 \
    --allow-unassociated-targets
```

Windows

```
aws ssm create-maintenance-window ^
    --name "My-One-Time-Maintenance-Window" ^
    --schedule "at(2020-07-07T15:55:00)" ^
    --duration 5 ^
    --cutoff 2 ^
    --allow-unassociated-targets
```

此维护时段仅运行一次，即在 2020 年 7 月 7 日下午 3:55（UTC 时间）运行。根据需要允许维护时段运行最多 5 小时，但在维护时段期限结束前的两小时禁止开始新任务。

示例 4：指定维护时段的计划偏移天数

现在，您可以创建具有此选项的维护时段：

```
--schedule-offset 2
```

例如：

Linux & macOS

```
aws ssm create-maintenance-window \
    --name "My-Cron-Offset-Maintenance-Window" \
    --schedule "cron(0 30 23 ? * TUE#3 *)" \
    --duration 4 \
    --cutoff 1 \
    --schedule-offset 2 \
    --allow-unassociated-targets
```

Windows

```
aws ssm create-maintenance-window ^
    --name "My-Cron-Offset-Maintenance-Window" ^
    --schedule "cron(0 30 23 ? * TUE#3 *)" ^
    --duration 4 ^
    --cutoff 1 ^
    --schedule-offset 2 ^
```

```
    --allow-unassociated-targets
```

计划偏移是在运行维护时段之前但在 CRON 表达式指定的日期和时间之后等待的天数。

在上面的示例中，CRON 表达式计划一个维护时段，在每月第三个星期二的晚上 11:30 运行：

```
--schedule "cron(0 30 23 ? * TUE#3 *)
```

但是，包括 `--schedule-offset 2` 表示维护时段要到每月第三个星期二之后 两天的晚上 11:30 才会运行。

仅针对 CLEN 表达式支持计划偏移。

更多信息

- 参考：适用于 Systems Manager 的 Cron 和 Rate 表达式
- 使用控制台创建维护时段
- 教程：使用 AWS CLI 创建和配置维护时段
- AWS Systems Manager API 参考中的 CreateMaintenanceWindow
- 《AWS CLI Command Reference》的 AWS Systems Manager 部分中的 create-maintenance-window
- IANA 网站上的时区数据库

注册不含目标的维护时段任务

对于您创建的每个维护时段，您可以指定要在维护时段运行时执行的一个或多个任务。在大多数情况下，您必须指定要在其上运行任务的资源或目标。但是，在某些情况下，无需在任务中明确指定目标。

必须为维护时段 Systems Manager Run Command 类型任务指定一个或多个目标。根据任务的性质，目标对于其他维护时段任务类型（Systems Manager 自动化、AWS Lambda 和 AWS Step Functions）是可选的。

对于 Lambda 和 Step Functions 任务类型，是否需要目标取决于您创建的函数或状态机的内容。

> **Note**
>
> 当任务注册了目标时，Automation、AWS Lambda 和 AWS Step Functions 任务会从资源组和标签解析目标，并为每个已解析的资源发送一次调用，这会导致多次任务调用。但是，举例来说，您只想对在包含多个实例的资源组中注册的 Lambda 任务进行一次调用。在这种情

况下，如果您正在使用AWS Management Console，请在注册 Lambda 任务或编辑 Lambda 任务页面中选择任务目标不是必需的选项。如果使用 AWS CLI 命令，则在运行 register-task-with-maintenance-window 命令或 update-maintenance-window-task 命令时，请勿使用 --targets 参数指定目标。

在许多情况下，您无需为自动化任务明确指定目标。例如，假设您要创建 自动化 类型的任务，以使用 AWS-UpdateLinuxAmi 运行手册为 Linux 更新 Amazon Machine Image (AMI)。在该任务运行时，已使用最新可用的 Linux 分发版本的程序包和 Amazon 软件更新了 AMI。从 AMI 创建的新实例已经安装了这些更新。由于要更新的 AMI 的 ID 是在运行手册的输入参数中指定的，因此无需在维护时段任务中再次指定目标。

同样，假设您正在使用 AWS Command Line Interface (AWS CLI) 注册使用 AWS-RestartEC2Instance 运行手册的维护时段 Automation 任务。由于要重启的节点是在 --task-invocation-parameters 参数中指定，您也无需指定 --targets 选项。

> **Note**
>
> 对于没有指定目标的维护时段任务，您无法为 --max-errors 和 --max-concurrency 提供值。作为替代方式，系统会插入一个占位符值 1，该值可能会在响应诸如 describe-maintenance-window-tasks 和 get-maintenance-window-task 等命令时发出报告。这些值不会影响任务的运行，并且可以忽略。

以下示例演示了为无目标维护时段任务省略 --targets、--max-errors 和 --max-concurrency 选项的情形。

Linux & macOS

```
aws ssm register-task-with-maintenance-window \
    --window-id "mw-ab12cd34eEXAMPLE" \
    --service-role-arn "arn:aws:iam::123456789012:role/
MaintenanceWindowAndAutomationRole" \
    --task-type "AUTOMATION" \
    --name "RestartInstanceWithoutTarget" \
    --task-arn "AWS-RestartEC2Instance" \
    --task-invocation-parameters "{\"Automation\":{\"Parameters\":{\"InstanceId\":
[\"i-02573cafcfEXAMPLE\"]}}}" \
    --priority 10
```

Windows

```
aws ssm register-task-with-maintenance-window ^
    --window-id "mw-ab12cd34eEXAMPLE" ^
    --service-role-arn "arn:aws:iam::123456789012:role/
MaintenanceWindowAndAutomationRole" ^
    --task-type "AUTOMATION" ^
    --name "RestartInstanceWithoutTarget" ^
    --task-arn "AWS-RestartEC2Instance" ^
    --task-invocation-parameters "{\"Automation\":{\"Parameters\":{\"InstanceId\":
[\"i-02573cafcfEXAMPLE\"]}}}" ^
    --priority 10
```

> **Note**
>
> 对于 2020 年 12 月 23 日之前注册的维护时段任务：如果您为任务指定了多个目标，但不再需要一个目标，则可以更新该任务，以使用 Systems Manager 控制台或 update-maintenance-window-task AWS CLI 命令删除目标。

更多信息

- 错误消息：“Maintenance window tasks without targets don't support MaxConcurrency values (没有目标的维护时段任务不支持 MaxConcurrency 值)”和“Maintenance window tasks without targets don't support MaxErrors values (没有目标的维护时段任务不支持 MaxErrors 值)”

对维护时段进行故障排除

可以使用以下信息来帮助您对维护时段的问题进行故障排除。

主题

- 编辑任务错误：在用于编辑维护时段任务的页面上，IAM 角色列表返回以下错误消息：“We couldn't find the IAM maintenance window role specified for this task。It might have been deleted, or it might not have been created yet. (找不到为此任务指定的 IAM 维护时段角色。它可能已被删除，也可能尚未创建。)”
- 并非所有维护时段目标都会被更新
- 任务失败并显示任务调用状态：“The provided role does not contain the correct SSM permissions”（提供的角色不包含正确的 SSM 权限）。

- 任务失败并显示错误消息：“Step fails when it is validating and resolving the step inputs (步骤在验证和解析步骤输入时失败)”
- 错误消息：“Maintenance window tasks without targets don't support MaxConcurrency values (没有目标的维护时段任务不支持 MaxConcurrency 值)”和“Maintenance window tasks without targets don't support MaxErrors values (没有目标的维护时段任务不支持 MaxErrors 值)”

编辑任务错误：在用于编辑维护时段任务的页面上，IAM 角色列表返回以下错误消息：“We couldn't find the IAM maintenance window role specified for this task。It might have been deleted, or it might not have been created yet. (找不到为此任务指定的 IAM 维护时段角色。它可能已被删除，也可能尚未创建。)”

问题 1：您最初指定的 AWS Identity and Access Management (IAM) 维护时段角色在您创建任务后被删除。

可能的修复措施：1) 选择其他 IAM 维护时段角色（如果您的账户中存在），或者新建一个角色并为任务选择该角色。

问题 2：如果任务是使用 AWS Command Line Interface (AWS CLI)、AWS Tools for Windows PowerShell 或 AWS 开发工具包创建的，则可能指定了不存在的 IAM 维护时段角色名称。例如，IAM 维护时段角色可能已在创建任务之前删除，或者键入的角色名称不正确，如 **`myrole`** 而不是 **`my-role`**。

可能的修复措施：为您要使用的 IAM 维护时段角色选择正确的名称，或者新建一个要为任务指定的角色。

并非所有维护时段目标都会被更新

问题：您注意到，维护时段任务未在维护时段设为目标的所有资源上运行。例如，在维护时段运行结果中，该资源的任务被标记为已失败或已超时。

解决方案：维护时段任务未在目标资源上运行的最常见原因涉及连接性和可用性。例如：

- Systems Manager 在维护时段操作之前或期间丢失与资源的连接。
- 在维护时段操作期间，资源处于离线或停止状态。

您可以等待下一个计划维护时段的时间在资源上运行任务。您可以在不可用或离线的资源上手动运行维护时段任务。

任务失败并显示任务调用状态：“The provided role does not contain the correct SSM permissions”（提供的角色不包含正确的 SSM 权限）。

问题：您已为任务指定维护时段服务角色，但该任务无法成功运行，并且任务调用状态报告“The provided role does not contain the correct SSM permissions”（提供的角色不包含正确的 SSM 权限）。

- 解决方案：在 任务 1：使用控制台为维护时段服务角色创建自定义策略 中，我们提供了一个基本策略，您可以将其附加到您的自定义维护时段服务角色。该策略包含许多任务方案所需的权限。但是，由于您可以运行的任务种类繁多，您可能需要在策略中为自己的维护时段角色提供其他权限。

  例如，一些 自动化 操作使用 AWS CloudFormation 堆栈。因此，您可能需要将附加权限 `cloudformation:CreateStack`、`cloudformation:DescribeStacks` 和 `cloudformation:DeleteStack` 添加到您的维护时段服务角色的策略中。

  另一个示例：自动化运行手册 `AWS-CopySnapshot` 需要权限来创建 Amazon Elastic Block Store（Amazon EBS）快照，因此，您可能需要添加权限 `ec2:CreateSnapshot`。

  有关 AWS 托管的自动化运行手册所需角色权限的信息，请参阅 AWS Systems Manager 自动化运行手册参考中的运行手册描述。

  有关 AWS 托管的 SSM 文档所需角色权限的信息，请查看文档部分 Systems Manager 控制台中的文档内容。

  有关 Step Functions 任务、Lambda 任务以及自定义自动化运行手册和 SSM 文档所需角色权限的信息，请向这些资源的作者验证权限要求。

任务失败并显示错误消息：“Step fails when it is validating and resolving the step inputs (步骤在验证和解析步骤输入时失败)”

问题：在任务中使用的 自动化 运行手册或 Systems Manager 命令文档要求您指定输入，例如 `InstanceId` 或 `SnapshotId`，但没有提供或未正确提供某个值。

- 解决方案 1：如果您的任务将单个资源（例如单个节点或单个快照）设为目标，请在任务的输入参数中输入其 ID。
- 解决方案 2：如果您的任务将多个资源设为目标，例如在使用运行手册 `AWS-CreateImage` 时从多个节点创建映像，您可以在输入参数中使用维护时段任务支持的伪参数之一来表示命令中的节点 ID。

以下命令使用 AWS CLI 将 Systems Manager 自动化 任务注册到维护时段。`--targets` 值指示维护时段目标 ID。此外，即使 `--targets` 参数指定时段目标 ID，自动化运行手册的参数也会要求提供节点 ID。在这种情况下，该命令使用伪参数 `{{RESOURCE_ID}}` 作为 `InstanceId` 值。

AWS CLI 命令：

Linux & macOS

以下示例命令会重新启动属于 ID 为 e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE 的维护时段目标组的 Amazon Elastic Compute Cloud（Amazon EC2）实例。

```
aws ssm register-task-with-maintenance-window \
    --window-id "mw-0c50858d01EXAMPLE" \
    --targets Key=WindowTargetIds,Values=e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE \
    --task-arn "AWS-RestartEC2Instance" \
    --service-role-arn arn:aws:iam::123456789012:role/
MyMaintenanceWindowServiceRole \
    --task-type AUTOMATION \
    --task-invocation-parameters
 "Automation={DocumentVersion=5,Parameters={InstanceId='{{RESOURCE_ID}}'}}" \
    --priority 0 --max-concurrency 10 --max-errors 5 --name "My-Restart-EC2-
Instances-Automation-Task" \
    --description "Automation task to restart EC2 instances"
```

Windows

```
aws ssm register-task-with-maintenance-window ^
    --window-id "mw-0c50858d01EXAMPLE" ^
    --targets Key=WindowTargetIds,Values=e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE ^
    --task-arn "AWS-RestartEC2Instance" ^
    --service-role-arn arn:aws:iam::123456789012:role/
MyMaintenanceWindowServiceRole ^
    --task-type AUTOMATION ^
    --task-invocation-parameters
 "Automation={DocumentVersion=5,Parameters={InstanceId='{{RESOURCE_ID}}'}}" ^
    --priority 0 --max-concurrency 10 --max-errors 5 --name "My-Restart-EC2-
Instances-Automation-Task" ^
    --description "Automation task to restart EC2 instances"
```

有关使用维护时段任务的伪参数的更多信息，请参阅 注册维护时段任务时使用伪参数 和 任务注册示例。

错误消息：“Maintenance window tasks without targets don't support MaxConcurrency values (没有目标的维护时段任务不支持 MaxConcurrency 值)”和“Maintenance window tasks without targets don't support MaxErrors values (没有目标的维护时段任务不支持 MaxErrors 值)”

问题：当您注册 Run Command 类型的任务时，必须指定至少一个目标，以便任务能在其上运行。对于其他任务类型（自动化、AWS Lambda 和 AWS Step Functions），根据任务的性质，目标是可选的。选项 `MaxConcurrency`（要同时运行某一任务的资源数量）和 `MaxErrors`（在任务失败之前在目标资源上运行该任务的失败次数）不是必需的，也不支持未指定目标的维护时段任务。当未指定任何任务目标时，如果为这些选项中的任何一个指定值，则系统将生成这些错误消息。

解决方案：如果您收到这些错误中的任何一个，请在继续注册或更新维护时段任务之前，删除并发值和错误阈值。

有关运行未指定目标的任务的更多信息，请参阅 AWS Systems Manager 用户指南中的 注册不含目标的维护时段任务。

AWS Systems Manager Quick Setup

使用Quick Setup（AWS Systems Manager 中的一项工具）可按照建议的最佳实践快速配置常用 Amazon Web Services 服务和功能。Quick Setup将通过自动执行常见或建议的任务，来简化 Systems Manager 等服务的设置。例如，这些任务包括创建必需的 AWS Identity and Access Management (IAM) 实例配置文件角色以及设置操作最佳实践，例如定期补丁扫描和清单收集。使用 Quick Setup 不会收取任何费用。但是，根据您设置的服务类型和使用限制，可能会产生成本，但不收取用于设置服务的服务费用。要开始使用 Quick Setup，请打开 Systems Manager 控制台。在导航窗格中，请选择 Quick Setup。

> **Note**
>
> 如果指示您 Quick Setup，以帮助您将实例配置为由 Systems Manager 托管，请完成 使用 Quick Setup 设置 Amazon EC2 主机管理 中的过程。

Quick Setup 的优势是什么？

Quick Setup 具有以下优势：

- 简化服务和功能配置

  Quick Setup 将指导您完成配置操作最佳实践并自动部署这些配置。Quick Setup 控制面板显示配置部署状态的实时视图。

- 跨多个账户自动部署配置

  可通过与 AWS Organizations 集成在单个 AWS 账户中或跨多个 AWS 账户和 AWS 区域使用 Quick Setup。跨多个账户使用 Quick Setup 有助于确保您的组织保持一致的配置。
- 消除配置漂移

  当用户对与通过 Quick Setup 进行的选择冲突的服务或功能进行更改时，会发生配置漂移。Quick Setup 会定期检查是否存在配置漂移并尝试进行修复。

谁应该使用 Quick Setup？

如果客户已经具有当前设置的服务和功能的一些使用经验并且希望简化其设置过程，建议这些客户使用 Quick Setup。如果不熟悉正在使用 Quick Setup 配置的 AWS 服务，我们建议您首先深入了解该服务。请先查看相关用户指南中的内容，然后再使用 Quick Setup 创建配置。

Quick Setup 在 AWS 区域 中的可用性

在下列 AWS 区域中，您可以为整个组织或者您选择的组织账户和区域使用所有 Quick Setup 配置类型（如 AWS Organizations 中所配置）。您也可以在这些区域中仅将 Quick Setup 用于一个账户。

- 美国东部（俄亥俄州）
- 美国东部（弗吉尼亚州北部）
- 美国西部（加利福尼亚北部）
- 美国西部（俄勒冈州）
- 亚太地区（孟买）
- 亚太地区（首尔）
- 亚太地区（新加坡）
- 亚太地区（悉尼）
- 亚太地区（东京）
- 加拿大（中部）
- 欧洲地区（法兰克福）
- 欧洲地区（斯德哥尔摩）
- 欧洲地区（爱尔兰）
- 欧洲地区（伦敦）
- 欧洲（巴黎）

- 南美洲（圣保罗）

在下列区域中，只有主机管理配置类型可用于单个账户：

- 欧洲地区（米兰）
- 亚太地区（香港）
- 中东（巴林）
- 中国（北京）
- China (Ningxia)
- AWS GovCloud（美国东部）
- AWS GovCloud（美国西部）

有关 Systems Manager 支持的所有区域的列表，请参阅《Amazon Web Services 一般参考》中 Systems Manager 服务端点 部分的区域列。

开始使用 Quick Setup

使用本主题中的信息帮助您准备使用 Quick Setup。

主题

- 注册 Quick Setup 的 IAM 角色和权限
- 以编程方式使用 Quick Setup API 时的手动初始设置

注册 Quick Setup 的 IAM 角色和权限

Quick Setup 推出了新的控制台体验和新的 API。现在，您可以使用控制台、AWS CLI、AWS CloudFormation 和 SDK 与此 API 进行交互。如果您选择使用新体验，则会使用新的 API 重新创建现有配置。根据您账户中的现有配置数量，此过程可能需要几分钟。

要使用新的 Quick Setup 控制台，您必须拥有以下操作的权限：

JSON

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

```
            "Effect": "Allow",
            "Action": [
                "ssm-quicksetup:*",
                "cloudformation:DescribeStackSetOperation",
                "cloudformation:ListStacks",
                "cloudformation:DescribeStacks",
                "cloudformation:DescribeStackResources",
                "cloudformation:ListStackSetOperations",
                "cloudformation:ListStackInstances",
                "cloudformation:DescribeStackSet",
                "cloudformation:ListStackSets",
                "cloudformation:DescribeStackInstance",
                "cloudformation:DescribeOrganizationsAccess",
                "cloudformation:ActivateOrganizationsAccess",
                "cloudformation:GetTemplate",
                "cloudformation:ListStackSetOperationResults",
                "cloudformation:DescribeStackEvents",
                "cloudformation:UntagResource",
                "ec2:DescribeInstances",
                "ssm:DescribeAutomationExecutions",
                "ssm:GetAutomationExecution",
                "ssm:ListAssociations",
                "ssm:DescribeAssociation",
                "ssm:GetDocument",
                "ssm:ListDocuments",
                "ssm:DescribeDocument",
                "ssm:ListResourceDataSync",
                "ssm:DescribePatchBaselines",
                "ssm:GetPatchBaseline",
                "ssm:DescribeMaintenanceWindows",
                "ssm:DescribeMaintenanceWindowTasks",
                "ssm:GetOpsSummary",
                "organizations:DeregisterDelegatedAdministrator",
                "organizations:DescribeAccount",
                "organizations:DescribeOrganization",
                "organizations:ListDelegatedAdministrators",
                "organizations:ListRoots",
                "organizations:ListParents",
                "organizations:ListOrganizationalUnitsForParent",
                "organizations:DescribeOrganizationalUnit",
                "organizations:ListAWSServiceAccessForOrganization",
                "s3:GetBucketLocation",
                "s3:ListAllMyBuckets",
                "s3:ListBucket",
```

```
                "resource-groups:ListGroups",
                "iam:ListRoles",
                "iam:ListRolePolicies",
                "iam:GetRole",
                "iam:CreatePolicy",
                "organizations:RegisterDelegatedAdministrator",
                "organizations:EnableAWSServiceAccess",
                "cloudformation:TagResource"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudformation:RollbackStack",
                "cloudformation:CreateStack",
                "cloudformation:UpdateStack",
                "cloudformation:DeleteStack"
            ],
            "Resource": [
                "arn:aws:cloudformation:*:*:stack/StackSet-AWS-QuickSetup-*",
                "arn:aws:cloudformation:*:*:stack/AWS-QuickSetup-*",
                "arn:aws:cloudformation:*:*:type/resource/*",
                "arn:aws:cloudformation:*:*:stack/StackSet-SSMQuickSetup"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudformation:CreateStackSet",
                "cloudformation:UpdateStackSet",
                "cloudformation:DeleteStackSet",
                "cloudformation:DeleteStackInstances",
                "cloudformation:CreateStackInstances",
                "cloudformation:StopStackSetOperation"
            ],
            "Resource": [
                "arn:aws:cloudformation:*:*:stackset/AWS-QuickSetup-*",
                "arn:aws:cloudformation:*:*:stackset/SSMQuickSetup",
                "arn:aws:cloudformation:*:*:type/resource/*",
                "arn:aws:cloudformation:*:*:stackset-target/AWS-QuickSetup-*:*"
            ]
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "iam:CreateRole",
                "iam:DeleteRole",
                "iam:AttachRolePolicy",
                "iam:DetachRolePolicy",
                "iam:GetRolePolicy",
                "iam:PutRolePolicy"
            ],
            "Resource": [
                "arn:aws:iam::*:role/AWS-QuickSetup-*",
                "arn:aws:iam::*:role/service-role/AWS-QuickSetup-*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::111122223333:role/AWS-QuickSetup-*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ssm-quicksetup.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:DeleteAssociation",
                "ssm:CreateAssociation",
                "ssm:StartAssociationsOnce"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ssm:StartAutomationExecution",
            "Resource": "arn:aws:ssm:*:*:automation-definition/AWS-
EnableExplorer:*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetOpsSummary",
                "ssm:CreateResourceDataSync",
```

```
                "ssm:UpdateResourceDataSync"
            ],
            "Resource": "arn:aws:ssm:*:*:resource-data-sync/AWS-QuickSetup-*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole"
            ],
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": [
                        "accountdiscovery.ssm.amazonaws.com",
                        "ssm.amazonaws.com",
                        "ssm-quicksetup.amazonaws.com",
                        "stacksets.cloudformation.amazonaws.com"
                    ]
                }
            },
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole"
            ],
            "Resource": "arn:aws:iam::*:role/aws-
service-role/stacksets.cloudformation.amazonaws.com/
AWSServiceRoleForCloudFormationStackSetsOrgAdmin"
        }
    ]
}
```

要将用户限制为只读权限，请仅允许对 Quick Setup API 执行 `ssm-quicksetup:List*` 和 `ssm-quicksetup:Get*` 操作。

在引导期间，Quick Setup 将代表您创建以下 AWS Identity and Access Management（IAM）角色：

- `AWS-QuickSetup-LocalExecutionRole` – 授予 AWS CloudFormation 使用任何模板（不包括补丁策略模板）和创建必要资源的权限。
- `AWS-QuickSetup-LocalAdministrationRole` - 授予 AWS CloudFormation 担任 `AWS-QuickSetup-LocalExecutionRole` 的权限。

- AWS-QuickSetup-PatchPolicy-LocalExecutionRole – 授予 AWS CloudFormation 使用补丁策略模板和创建必要资源的权限。
- AWS-QuickSetup-PatchPolicy-LocalAdministrationRole - 授予 AWS CloudFormation 担任 AWS-QuickSetup-PatchPolicy-LocalExecutionRole 的权限。

如果您正在注册管理账户（用于在 AWS Organizations 中创建组织的账户），Quick Setup 还会代表您创建以下角色：

- AWS-QuickSetup-SSM-RoleForEnablingExplorer - 授予 AWS-EnableExplorer 自动化 Runbook 权限。AWS-EnableExplorer 运行手册可配置 Explorer（Systems Manager 中的一项工具）来显示多个 AWS 账户和 AWS 区域的信息。
- AWSServiceRoleForAmazonSSM - 服务链接角色，用于授予对由 Systems Manager 管理和使用的 AWS 资源的访问权限。
- AWSServiceRoleForAmazonSSM_AccountDiscovery – 服务相关角色，用于向 Systems Manager 授予在同步数据时调用 AWS 服务来查找 AWS 账户信息的权限。有关更多信息，请参阅使用角色为 OpsCenter 和 Explorer 收集 AWS 账户 信息。

在引导管理账户时，Quick Setup 将在 AWS Organizations 和 CloudFormation 之间启用受信任的访问权限，来部署整个组织的 Quick Setup 配置。要启用受信任访问权限，您的管理账户必须具有管理员权限。在完成引导后，便不再需要管理员权限。有关更多信息，请参阅启用 Organizations 受信任的访问权限。

有关 AWS Organizations 账户类型的更多信息，请参阅《AWS Organizations 用户指南》中的 AWS Organizations 术语和概念。

> **Note**
>
> Quick Setup 使用 AWS CloudFormation StackSets 通过 AWS 账户 和区域来部署您的配置。如果目标账户数量乘以区域数超过 10000，则配置部署失败。我们建议检查您的用例并创建使用较少目标的配置，以适应组织的增长。堆栈实例不会部署到组织的管理账户。有关更多信息，请参阅创建具有服务托管权限的堆栈集时的注意事项。

以编程方式使用 Quick Setup API 时的手动初始设置

如果您使用控制台操作 Quick Setup，则该服务会为您处理开户步骤。如果您计划使用 SDK 或 AWS CLI 操作 Quick Setup API，则仍然可以使用控制台为您完成开户步骤，这样您就不必手动执行这些步

骤。但是，有些客户需要在不与控制台交互的情况下以编程方式完成 Quick Setup 的开户步骤。如果此方法适合您的使用案例，您必须完成以下步骤。所有这些步骤都必须通过您的 AWS Organizations 管理账户完成。

若要完成 Quick Setup 的手动开户

1. 使用 Organizations 激活 AWS CloudFormation 的可信访问权限。这为管理账户提供为您的组织创建和管理 StackSets 所需的权限。您可以使用 AWS CloudFormation 的 `ActivateOrganizationsAccess` API 操作来完成此步骤。有关更多信息，请参阅 AWS CloudFormation API 参考中的ActivateOrganizationsAccess。
2. 使用 Organizations 启用 Systems Manager 的集成。这允许 Systems Manager 在您组织的所有账户中创建服务相关角色。这也允许 Systems Manager 在您的组织和账户内代表您执行操作。您可以使用 AWS Organizations 的 `EnableAWSServiceAccess` API 操作来完成此步骤。Systems Manager 的服务主体是 *`ssm.amazonaws.com`*。有关更多信息，请参阅 AWS Organizations API 参考中的 EnableAWSServiceAccess。
3. 为 Explorer 创建所需的 IAM 角色。这就允许 Quick Setup 针对您的配置创建控制面板，以便您查看部署和关联状态。创建 IAM 角色并附加 `AWSSystemsManagerEnableExplorerExecutionPolicy` 托管策略。修改该角色的信任策略，使其符合以下内容。将每个*`## ID`* 替换为您的信息。

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ssm.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "account ID"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:*:ssm:*:111122223333:automation-
execution/*"
                }
```

```
            }
        }
    ]
}
```

4. 更新 Explorer 的 Quick Setup 服务设置。您可以使用 Quick Setup 的 `UpdateServiceSettings` API 操作来完成此步骤。针对 `ExplorerEnablingRoleArn` 请求参数指定您在上一步中创建的 IAM 角色的 ARN。有关更多信息，请参阅 Quick Setup API 参考中的 UpdateServiceSettings。
5. 创建必需的 IAM 角色以供 AWS CloudFormation StackSets 使用。您必须创建执行角色和管理角色。

   a. 创建执行角色。执行角色应至少附加一个 `AWSQuickSetupDeploymentRolePolicy` 或 `AWSQuickSetupPatchPolicyDeploymentRolePolicy` 托管策略。如果您只是在创建补丁策略配置，则可以使用 `AWSQuickSetupPatchPolicyDeploymentRolePolicy` 托管策略。所有其他配置都使用 `AWSQuickSetupDeploymentRolePolicy` 策略。修改该角色的信任策略，使其符合以下内容。使用您的信息替换每个*## ID* 和*######*

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/administration
 role name"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

   b. 创建管理角色。权限策略必须符合以下内容。使用您的信息替换每个*## ID* 和*######*

   JSON

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Action": [
                "sts:AssumeRole"
            ],
            "Resource": "arn:*:iam::111122223333:role/execution role
 name",
            "Effect": "Allow"
        }
    ]
}
```

修改该角色的信任策略，使其符合以下内容。将每个*## ID* 替换为您的信息。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "cloudformation.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333"
                },
                "ArnLike": {
                    "aws:SourceArn":
 "arn:aws:cloudformation:*:111122223333:stackset/AWS-QuickSetup-*"
                }
            }
        }
    ]
}
```

对Quick Setup使用委派管理员

在为 Quick Setup 注册委托管理员账户后，在该账户中拥有相应权限的用户将可以创建、更新、查看和删除以 AWS Organizations 结构中组织单位为目标的配置管理器。此委托管理员账户还可以管理之前由组织的管理账户创建的配置管理器。

组织中的管理账户可以将组织中的任何账户指定为委托管理员。将某个账户注册为 Quick Setup 的委托管理员后，该账户会自动成为 AWS CloudFormation StackSets 和 Systems Manager Explorer 的委托管理员，因为 Quick Setup 配置的部署和监控将需要这些服务。

> **Note**
>
> 目前，Quick Setup的委派管理员不支持补丁策略配置类型。必须在组织的管理账户中创建和维护组织的补丁策略配置。有关更多信息，请参阅 the section called “创建补丁策略”。

以下主题介绍了如何注册和取消注册Quick Setup的委派管理员。

主题

- 注册Quick Setup的委派管理员
- 取消注册Quick Setup的委派管理员

注册Quick Setup的委派管理员

使用以下过程注册Quick Setup的委派管理员。

注册Quick Setup委派管理员

1. 登录到您的 AWS Organizations 管理账户。
2. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
3. 在导航窗格中，请选择 Quick Setup。
4. 选择设置。
5. 在 Delegated administrator for Quick Setup (Explorer 的委派管理员) 部分，验证您是否已配置所需的服务链接角色和服务访问选项。如有必要，请选择 Create role (创建角色) 和 Enable access (启用访问权限) 按钮来配置这些选项。
6. 对于 Account ID (账户 ID)，输入 AWS 账户 ID。此账户必须是 AWS Organizations 中的成员账户。

7. 选择 Register delegated administrator (注册委派管理员)。

取消注册Quick Setup的委派管理员

使用以下过程来取消注册Quick Setup的委派管理员。

取消注册Quick Setup委派管理员

1. 登录到您的 AWS Organizations 管理账户。
2. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
3. 在导航窗格中，请选择 Quick Setup。
4. 选择设置。
5. 在Quick Setup的委派管理员部分中，从操作下拉菜单中选择取消注册。
6. 选择确认。

了解 Quick Setup 术语和详细信息

Quick Setup是 AWS Systems Manager 中的一项工具，其在Quick Setup主页的配置管理器表中显示了您在所有 AWS 区域创建的所有配置管理器的结果。在此页面上，您可以查看每个配置的详细信息，从 Actions（操作）下拉列表中删除配置，或 Create（创建）配置。配置管理器表包含以下信息：

- 名称 - 配置管理器的名称（如果在创建时提供）。
- Configuration type（配置类型）– 在创建配置时选择的配置类型。
- 版本 - 当前部署的配置类型的版本。
- Organizational units（组织单位）– 显示配置部署到的组织单位 (OU)（如果您选择了一组 Custom（自定义）目标。）组织单位和自定义目标仅适用于组织的管理账户。管理账户是您用于在 AWS Organizations 中创建组织的账户。
- 部署类型 – 指示该部署是适用于整个组织（`Organizational`）还是仅适用于您的账户（`Local`）。
- Regions（区域）– 配置部署到的区域，前提是您选择了一组 Custom（自定义）目标或 Current account（当前账户）内的目标。
- Deployment status（部署状态）– 部署状态指示 AWS CloudFormation 是否已成功部署目标或堆栈实例。目标实例和堆栈实例包含您在配置创建过程中选择的配置选项。

- Association status（关联状态）– 关联状态是由您创建的配置创建的所有关联的状态。必须成功运行所有目标的关联，否则状态为 Failed（失败）。

  Quick Setup创建并运行每个配置目标的State Manager关联。State Manager是 AWS Systems Manager 中的一项工具。

要查看部署到您当前正在浏览的区域的配置，请选择配置选项卡。

配置详细信息

Configuration details（配置详细信息）页面显示有关配置及其相关联的部署的信息。在此页面中，您可以编辑配置选项、更新目标或删除配置。您还可以查看每个配置部署的详细信息，以获取有关这些关联的更多信息。

根据配置类型，会显示以下一个或多个状态图：

配置部署状态

显示已成功、失败、正在运行或挂起的部署数。部署发生在包含受配置影响的节点的指定目标账户和区域中。

配置关联状态

显示已成功、失败或待处理的 State Manager 关联的数量。Quick Setup 在每个部署中为选定的配置选项创建关联。

设置状态

显示配置类型执行的操作数量及其当前状态。

资源合规性

显示符合配置指定策略的资源数量。

Configuration details（配置详细信息）表显示有关配置部署的信息。通过选择部署然后选择 View details（查看详细信息），您可以查看有关每个部署的更多详细信息。每个部署的详细信息页面显示部署到该部署中的节点的关联。

编辑和删除配置

您可以通过选择 Actions（操作）然后选择 Edit configuration options（编辑配置选项）编辑 Configuration details（配置详细信息）页面中的某个配置的配置选项。向配置添加新选项时，Quick

Setup 运行部署并创建新的关联。从配置中删除选项时，Quick Setup 运行部署并删除任何相关的关联。

> **Note**
>
> 您可随时编辑账户的 Quick Setup 配置。要编辑组织配置，配置状态必须为 Success（成功）或 Failed（失败）。

您还可以通过选择 Actions（操作）和 Add OUs（添加 OU）、Add Regions（添加区域）、Remove OUs（删除 OU）或 Remove Regions（删除区域）来更新包含在您的配置中的目标。如果账户未配置为管理账户，或者仅为当前账户创建了配置，则无法更新目标组织单位（OU）。删除区域或 OU 会从这些区域或 OU 中删除关联。

Quick Setup 会定期发布新版本的配置。您可以选择升级配置选项将您的配置升级到最新版本。

您可以通过选择配置，然后选择 Actions（操作）和 Delete configuration（删除配置）从 Quick Setup 中删除配置。您还可以在 Actions（操作）下拉列表下选择 Configuration details（配置详细信息）页面，然后选择 Delete configuration（删除配置），从而删除配置。然后，Quick Setup 将提示您 Remove all OUs and Regions（删除所有 OU 和区域），完成此操作可能需要一些时间。删除配置还会删除所有相关关联。此两步删除过程将从所有账户和区域中删除所有已部署的资源，然后删除配置。

配置合规性

您可以在 Explorer 或 Compliance（都是 AWS Systems Manager 中的工具）中查看实例是否符合配置创建的关联。要了解有关合规性的更多信息，请参阅 了解有关合规性的详细信息。要了解有关查看 Explorer 中的合规性的更多信息，请参阅 AWS Systems Manager Explorer。

使用 Quick Setup API 管理配置和部署

您可以使用 Quick Setup 提供的 API，通过 AWS CLI 或您的首选 SDK 创建和管理配置与部署。您还可以使用 AWS CloudFormation 创建用于部署配置的配置管理器资源。使用 API，您可以创建用于部署配置定义的配置管理器。配置定义包含部署特定配置类型的所有必要信息。有关使用 Quick Setup API 的更多信息，请参阅快速设置 API 参考。

以下示例演示如何使用 AWS CLI 和 AWS CloudFormation 创建配置管理器。

AWS CLI

```
aws ssm-quicksetup create-configuration-manager \
```

```
--name configuration manager name \
--description Description of your configuration manager
--configuration-definitions JSON string containing configuration defintion
```

以下是包含补丁策略配置定义的 JSON 字符串示例。

```
'{"Type":"AWSQuickSetupType-
PatchPolicy","LocalDeploymentAdministrationRoleArn":"arn:aws:iam::123456789012:role/
AWS-QuickSetup-StackSet-Local-
AdministrationRole","LocalDeploymentExecutionRoleName":"AWS-
QuickSetup-StackSet-Local-ExecutionRole","Parameters":
{"ConfigurationOptionsInstallNextInterval":"true","ConfigurationOptionsInstallValue":"cron(0
 2 ? * SAT#1
 *)","ConfigurationOptionsPatchOperation":"ScanAndInstall","ConfigurationOptionsScanNextInte
 1 * * ?
 *)","HasDeletedBaseline":"false","IsPolicyAttachAllowed":"true","OutputBucketRegion":"","Ou
east-1","PatchBaselineUseDefault":"custom","PatchPolicyName":"dev-patch-
policy","RateControlConcurrency":"5","RateControlErrorThreshold":"0%","RebootOption":"Reboot
\":{\"value\":\"arn:aws:ssm:us-east-1:123456789012:patchbaseline/
pb-0cb0c4966f86b059b\",\"label\":\"AWS-AlmaLinuxDefaultPatchBaseline\",\"description
\":\"Default Patch Baseline for Alma Linux Provided by AWS.\",\"disabled\":false},
\"AMAZON_LINUX\":{\"value\":\"arn:aws:ssm:us-east-1:123456789012:patchbaseline/
pb-0c10e657807c7a700\",\"label\":\"AWS-AmazonLinuxDefaultPatchBaseline\",
\"description\":\"Default Patch Baseline for Amazon Linux Provided by AWS.
\",\"disabled\":false},\"AMAZON_LINUX_2\":{\"value\":\"arn:aws:ssm:us-
east-1:123456789012:patchbaseline/pb-0be8c61cde3be63f3\",\"label\":\"AWS-
AmazonLinux2DefaultPatchBaseline\",\"description\":\"Baseline containing
 all Security and Bugfix updates approved for Amazon Linux 2 instances
\",\"disabled\":false},\"AMAZON_LINUX_2022\":{\"value\":\"arn:aws:ssm:us-
east-1:123456789012:patchbaseline/pb-0028ca011460d5eaf\",\"label\":\"AWS-
AmazonLinux2022DefaultPatchBaseline\",\"description\":\"Default Patch
 Baseline for Amazon Linux 2022 Provided by AWS.\",\"disabled\":false},
\"AMAZON_LINUX_2023\":{\"value\":\"arn:aws:ssm:us-east-1:123456789012:patchbaseline/
pb-05c9c9bf778d4c4d0\",\"label\":\"AWS-AmazonLinux2023DefaultPatchBaseline
\",\"description\":\"Default Patch Baseline for Amazon Linux 2023 Provided
 by AWS.\",\"disabled\":false},\"CENTOS\":{\"value\":\"arn:aws:ssm:us-
east-1:123456789012:patchbaseline/pb-03e3f588eec25344c\",\"label\":\"AWS-
CentOSDefaultPatchBaseline\",\"description\":\"Default Patch Baseline for CentOS
 Provided by AWS.\",\"disabled\":false},\"DEBIAN\":{\"value\":\"arn:aws:ssm:us-
east-1:123456789012:patchbaseline/pb-09a5f8eb62bde80b1\",\"label\":\"AWS-
DebianDefaultPatchBaseline\",\"description\":\"Default Patch Baseline for Debian
 Provided by AWS.\",\"disabled\":false},\"MACOS\":{\"value\":\"arn:aws:ssm:us-
east-1:123456789012:patchbaseline/pb-0ee4f94581368c0d4\",\"label\":\"AWS-
```

```
MacOSDefaultPatchBaseline\",\"description\":\"Default Patch Baseline for
 MacOS Provided by AWS.\",\"disabled\":false},\"ORACLE_LINUX\":{\"value\":
\"arn:aws:ssm:us-east-1:123456789012:patchbaseline/pb-06bff38e95fe85c02\",
\"label\":\"AWS-OracleLinuxDefaultPatchBaseline\",\"description\":\"Default
 Patch Baseline for Oracle Linux Server Provided by AWS.\",\"disabled\":false},
\"RASPBIAN\":{\"value\":\"arn:aws:ssm:us-east-1:123456789012:patchbaseline/
pb-0ec16280999c5c75e\",\"label\":\"AWS-RaspbianDefaultPatchBaseline\",
\"description\":\"Default Patch Baseline for Raspbian Provided by AWS.\",
\"disabled\":false},\"REDHAT_ENTERPRISE_LINUX\":{\"value\":\"arn:aws:ssm:us-
east-1:123456789012:patchbaseline/pb-0cbb3a633de00f07c\",\"label\":\"AWS-
RedHatDefaultPatchBaseline\",\"description\":\"Default Patch Baseline for Redhat
 Enterprise Linux Provided by AWS.\",\"disabled\":false},\"ROCKY_LINUX\":{\"value
\":\"arn:aws:ssm:us-east-1:123456789012:patchbaseline/pb-03ec98bc512aa3ac0\",\"label
\":\"AWS-RockyLinuxDefaultPatchBaseline\",\"description\":\"Default Patch Baseline
 for Rocky Linux Provided by AWS.\",\"disabled\":false},\"SUSE\":{\"value\":
\"arn:aws:ssm:us-east-1:123456789012:patchbaseline/pb-07d8884178197b66b\",\"label\":
\"AWS-SuseDefaultPatchBaseline\",\"description\":\"Default Patch Baseline for Suse
 Provided by AWS.\",\"disabled\":false},\"UBUNTU\":{\"value\":\"pb-06e3563bd35503f2b
\",\"label\":\"custom-UbuntuServer-Blog-Baseline\",\"description\":\"Default Patch
 Baseline for Ubuntu Provided by AWS.\",\"disabled\":false},\"WINDOWS\":{\"value
\":\"pb-016889927b2bb8542\",\"label\":\"custom-WindowsServer-Blog-Baseline\",
\"disabled\":false}}","TargetInstances":"","TargetOrganizationalUnits":"ou-9utf-
example","TargetRegions":"us-east-1,us-
east-2","TargetTagKey":"Patch","TargetTagValue":"true","TargetType":"Tags"}}' \
```

AWS CloudFormation

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
SSMQuickSetupTestConfigurationManager:
Type: "AWS::SSMQuickSetup::ConfigurationManager"
Properties:
    Name: "MyQuickSetup"
    Description: "Test configuration manager"
    ConfigurationDefinitions:
    - Type: "AWSQuickSetupType-CFGRecording"
      Parameters:
        TargetAccounts:
            Ref: AWS::AccountId
        TargetRegions:
            Ref: AWS::Region
        LocalDeploymentAdministrationRoleArn: !Sub "arn:aws:iam::
${AWS::AccountId}:role/AWS-QuickSetup-StackSet-ContractTest-AdministrationRole"
```

```
        LocalDeploymentExecutionRoleName: "AWS-QuickSetup-StackSet-ContractTest-
ExecutionRole"
    Tags:
        foo1: "bar1"
```

支持的 Quick Setup 配置类型

支持的配置类型

Quick Setup 将指导您为多个 Systems Manager 和其他 AWS 服务 配置操作最佳实践，并自动部署这些配置。Quick Setup 控制面板显示配置部署状态的实时视图。

您可以通过与 AWS Organizations 集成在单个 AWS 账户中或跨多个 AWS 账户 和区域中使用 Quick Setup。跨多个账户使用 Quick Setup 有助于确保您的组织保持一致的配置。

Quick Setup 支持以下配置类型。

- 使用 Quick Setup 设置 Amazon EC2 主机管理
- 使用 Quick Setup 为组织设置默认主机管理配置
- 使用 Quick Setup 创建 AWS Config 配置记录器
- 使用 Quick Setup 部署 AWS Config 一致性包
- 使用 Quick Setup 补丁策略为组织中的实例配置修补
- Change Manager 组织设置
- 使用 Quick Setup 设置 DevOps Guru
- 使用 Quick Setup 部署 Distributor 软件包
- 按计划使用 Quick Setup 自动停止和启动 EC2 实例
- OpsCenter 组织设置
- 使用 Quick Setup 配置 AWS 资源探索器

使用 Quick Setup 设置 Amazon EC2 主机管理

使用Quick Setup（AWS Systems Manager 中的一项工具）在 Amazon Elastic Compute Cloud（Amazon EC2）实例上快速配置必要的安全角色和常用的 Systems Manager 工具。可通过与 AWS Organizations 集成在单个账户中或跨多个账户和 AWS 区域使用 Quick Setup。这些工具帮助您管理和监控实例的运行状况，同时提供开始使用所需的最低权限。

如果您不熟悉 Systems Manager 的服务和功能，建议您先查看《AWS Systems Manager 用户指南》，然后再使用 Quick Setup 创建配置。有关 Systems Manager 的更多信息，请参阅 什么是 AWS Systems Manager？。

Important

如果以下任一项适用于您，则 Quick Setup 可能不是适合 EC2 管理使用的正确工具：

- 您第一次尝试创建 EC2 实例，以尝试 AWS 功能。
- 您仍然是 EC2 实例管理的新客户。

相反，建议您浏览以下内容：

- Amazon EC2 入门
- 《Amazon EC2 用户指南》中的使用新启动实例向导启动实例
- 《Amazon EC2 用户指南》中的教程：Amazon EC2 Linux 实例入门

如果您已经熟悉 EC2 实例管理，并且希望简化多个 EC2 实例的配置和管理，请使用 Quick Setup。无论您的组织有几十个、数千个还是数百万个 EC2 实例，都可以使用以下 Quick Setup 过程一次性为它们配置多个选项。

Note

借助此配置类型，您可以为 AWS Organizations 中定义的整个组织、仅部分组织账户和区域或单个账户设置多个选项。其中一个选项是每两周检查并应用 SSM Agent 更新。如果您是组织管理员，则还可以选择使用默认主机管理配置类型，每两周更新一次代理，从而更新组织中的所有 EC2 实例。有关信息，请参阅使用 Quick Setup 为组织设置默认主机管理配置。

配置 EC2 实例的主机管理选项

要设置主机管理，请在 AWS Systems Manager Quick Setup 控制台中执行以下任务。

打开“主机管理配置”页面

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。

2. 在导航窗格中，选择 Quick Setup。
3. 在主机管理卡片上，选择创建。

> **Tip**
>
> 如果您的账户中已经有一个或多个配置，请先选择库选项卡或配置部分的中的创建按钮以查看卡片。

配置 Systems Manager 主机管理选项

- 要配置 Systems Manager 功能，请在配置选项部分的 Systems Manager 组中选择要为配置启用的选项：

  每两周更新一次 Systems Manager（SSM）代理

  启用 Systems Manager，每两周检查一次是否存在新版本的代理。如果存在新版本，Systems Manager 会自动将托管式节点上的代理更新为发布的最新版本。Quick Setup 不会在尚不存在代理的实例上安装此代理。有关哪些 AMIs 预装了 SSM Agent 的信息，请参阅 查找预装了 SSM Agent 的 AMIs。

  建议您选择此选项，以确保您的节点始终运行最新版本的 SSM Agent。有关 SSM Agent 的更多信息（包括有关如何手动安装该代理的信息），请参阅 使用 SSM Agent。

  每 30 分钟从实例中收集一次清单

  启用 Quick Setup 配置以下类型的元数据收集：

  - AWS 组件 - EC2 驱动程序、代理和版本等。
  - 应用程序 - 应用程序名称、发布者和版本等。
  - 节点详细信息 - 系统名称、操作系统（OS）名称、操作系统版本、上次启动时间、DNS、域、工作组和操作系统架构等。
  - 网络配置 - IP 地址、MAC 地址、DNS、网关和子网掩码等。
  - 服务 - 名称、显示名称、状态、相关服务、服务类型和启动类型等（仅限 Windows Server 节点）。
  - Windows 角色 - 名称、显示名称、路径、功能类型和安装状态等（仅限 Windows Server 节点）。

- Windows 更新 - 补丁 ID、安装者和安装日期等（仅限 Windows Server 节点）。

有关 Inventory（AWS Systems Manager 中的一项工具）的更多信息，请参阅 AWS Systems Manager 清单。

> **Note**
>
> 即使您仅选择了几个节点，Inventory collection（清单收集）选项也可能需要长达 10 分钟才能完成。

每天扫描实例以查找缺少的补丁

启用Patch Manager（Systems Manager 中的一项工具），每天扫描节点并在合规性页面中生成报告。该报告根据默认补丁基准显示满足补丁合规性的节点数。该报告包含一份关于每个节点及其合规性状态的列表。

有关修补操作和补丁基准的信息，请参阅 AWS Systems Manager Patch Manager。

要查看补丁合规信息，请参阅 Systems Manager 合规性页面。

有关在一个配置中修补多个账户和区域中的托管节点的信息，请参阅 Quick Setup 中的补丁策略配置 和 使用 Quick Setup 补丁策略为组织中的实例配置修补。

> **Important**
>
> Systems Manager 支持多种方法来扫描托管节点，以满足补丁合规性。若同时执行上述方法中的多种方法，您看到的补丁合规信息将始终是最近一次扫描的结果。先前扫描的结果将被覆盖。如果扫描方法使用不同的补丁基准，有不同的批准规则，那么补丁合规信息可能会出现意外变更。有关更多信息，请参阅 避免意外覆盖补丁合规性数据。

配置 Amazon CloudWatch 主机管理选项

- 要配置 CloudWatch 功能，请在配置选项部分的 Amazon CloudWatch 组中选择要为配置启用的选项：

安装和配置 CloudWatch 代理

在 Amazon EC2 实例上安装统一的 CloudWatch 代理的基本配置。代理将从 Amazon CloudWatch 的实例中收集指标和日志文件。将合并此信息以快速确定实例的运行状况。有关 CloudWatch 代理基本配置的更多信息，请参阅 CloudWatch 代理预定义指标集。可能会增加成本。有关更多信息，请参阅 Amazon CloudWatch 定价。

每 30 天更新一次 CloudWatch 代理

启用 Systems Manager，每 30 天检查一次是否存在新版本的 CloudWatch 代理。如果存在新版本， Systems Manager 会自动在实例上更新代理。我们建议您选择该选项，以确保您的实例始终运行最新版本的 CloudWatch 代理。

配置 Amazon EC2 启动代理的主机管理选项

- 要配置 Amazon EC2 启动代理功能，请在配置选项部分的 Amazon EC2 启动代理组中选择要为配置启用的选项：

每 30 天更新一次 EC2 启动代理

启用 Systems Manager，每 30 天检查一次实例上安装的启动代理是否存在新版本。如果有新版本可用，则 Systems Manager 会在实例上更新代理。我们建议您选择该选项，以确保您的实例始终运行最新版本的适用启动代理。对于 Amazon EC2 Windows 实例，此选项支持 EC2Launch、EC2Launch v2 和 EC2Config。对于 Amazon EC2 Linux 实例，此选项支持 `cloud-init`。对于 Amazon EC2 Mac 实例，此选项支持 `ec2-macos-init`。Quick Setup 不支持更新安装在启动代理不支持的操作系统或 AL2023 上的启动代理。

有关这些初始化代理的更多信息，请参阅以下主题：

- 使用 EC2Launch v2 配置 Windows 实例
- 使用 EC2Launch 配置 Windows 实例
- 使用 EC2Config 服务配置 Windows 实例
- cloud-init 文档
- ec2-macos-init

选择要通过“主机管理配置”更新的 EC2 实例

- 在目标部分，选择用于确定要部署配置的账户和区域的方法：

> **Note**
>
> 无法创建定位同一 AWS 区域的多个 Quick Setup 主机管理配置。

Entire organization

配置已部署到所有组织单位（OU）和组织中的 AWS 区域。

> **Note**
>
> 仅当正在从组织的管理账户配置主机管理时，整个组织选项才可用。

Custom

1. 在目标 OU 部分，选择要部署“主机管理配置”的 OU。
2. 在目标区域部分，选择要部署“主机管理配置”的区域。

Current account

选择其中一个区域选项，然后按照该选项的步骤进行操作。

当前区域

选择如何仅将当前区域中的实例作为目标：

- 所有实例 –“主机管理配置”会自动将当前区域中每个 EC2 实例作为目标。
- 标签 – 选择添加，并输入添加到要作为目标的实例的键和可选值。
- 资源组 – 对于资源组，选择包含要作为目标的 EC2 实例的现有资源组。
- 手册 – 在实例部分中，选中要作为目标的每个 EC2 实例的复选框。

选择区域

通过选择以下选项之一，选择如何将指定区域中的实例作为目标：

- 所有实例 – 将指定区域中的所有实例作为目标。
- 标签 – 选择添加，并输入已添加到要作为目标的实例的键和可选值。

在目标区域部分，选择要部署“主机管理配置”的区域。

指定实例配置文件选项

- 仅限整个组织和自定义目标。

  在实例配置文件选项部分，选择是要将必需的 IAM 策略添加至已附加到实例的现有实例配置文件，还是允许 Quick Setup 使用所选配置所需权限创建 IAM 策略和实例配置文件。

指定好所有配置选项后，选择创建。

使用 Quick Setup 为组织设置默认主机管理配置

借助Quick Setup（AWS Systems Manager 中的一项工具），您可以为自己在 AWS Organizations 中添加到组织的所有账户和区域激活默认主机管理配置。这样可以确保组织中所有 Amazon Elastic Compute Cloud（EC2）实例上的 SSM Agent 保持更新，并且可以连接到 Systems Manager。

开始前的准备工作

您必须满足以下要求才能启用此设置。

- 组织中要管理的所有 EC2 实例上都已安装了最新版本的 SSM Agent。
- 要管理的 EC2 实例使用的是实例元数据服务版本 2（IMDSv2）。
- 您使用具有管理员权限的 AWS Identity and Access Management（IAM）身份（用户、角色或组）登录组织的管理账户（如在 AWS Organizations 中所指定）。

使用默认 EC2 实例管理角色

默认主机管理配置将使用 Systems Manager 的 `default-ec2-instance-management-role` 服务设置。该角色具有您希望提供给组织中所有账户的权限，从而便于在实例上的 SSM Agent 与云端 Systems Manager 服务之间进行通信。

如果您已经使用 update-service-setting CLI 命令设置了该角色，则默认主机管理配置将使用该角色。如果您尚未设置，Quick Setup 将为您创建并应用该角色。

要检查是否已经为您的组织指定了该角色，请使用 get-service-setting 命令。

启用每两周自动更新 SSM Agent

按照以下过程为整个 AWS Organizations 组织启用“默认主机管理配置”选项。

启用每两周自动更新 SSM Agent

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Quick Setup。
3. 在默认主机管理配置选项片上，选择创建。

> **Tip**
>
> 如果您的账户中已经有一个或多个配置，请先选择库选项卡或配置部分的中的创建按钮以查看卡片。

4. 在配置选项部分中，选择启用每两周自动更新 SSM Agent。
5. 选择 Create（创建）。

使用 Quick Setup 创建 AWS Config 配置记录器

借助Quick Setup（AWS Systems Manager 中的一项工具），您可以快速创建由 AWS Config 提供支持的配置记录器。使用配置记录器在您的资源配置中检测更改，并将这些更改捕获为配置项。如果您不熟悉 AWS Config，建议您在使用 Quick Setup 创建配置之前查看《AWS Config 开发人员指南》内容，了解有关该服务的更多信息 有关 AWS Config 的更多信息，请参阅《AWS Config 开发人员指南》中的什么是 AWS Config？。

预设情况下，配置记录器会记录正在运行 AWS Config 的 AWS 区域内所有受支持的资源。您可以自定义配置，以仅记录指定的资源类型。有关更多信息，请参阅《AWS Config 开发人员指南》中的选择 AWS Config 记录的资源。

当 AWS Config 开始记录配置时，我们就会向您收取服务使用费。有关定价信息，请参阅 AWS Config 定价。

> **Note**
>
> 如果您已经创建了配置记录器，Quick Setup 不会停止记录或对已记录资源类型进行任何更改。如果您选择使用 Quick Setup 记录其他资源类型，则该服务会将它们附加到您现有的记录器组中。删除Quick Setup Config 记录配置类型时，配置记录器不会停止工作。将继续记录更改，并且在停止配置记录器之前会一直收取服务使用费。要了解有关管理配置记录器的更多信息，请参阅《AWS Config 开发人员指南》中的管理配置记录器。

要设置 AWS Config 记录，请在 AWS Systems Manager 控制台中执行以下任务。

使用 Quick Setup 设置 AWS Config 记录

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Quick Setup。
3. 在配置记录卡片上，选择创建。

   > **Tip**
   >
   > 如果您的账户中已经有一个或多个配置，请先选择库选项卡或配置部分的中的创建按钮以查看卡片。

4. 在配置选项部分中，执行以下操作：

   a. 对于选择要记录的 AWS 资源类型，指定是记录所有支持的资源还是仅记录您选择的资源类型。

   b. 对于交付设置，指定是新建 Amazon Simple Storage Service（Amazon S3）存储桶，还是选择要向其发送配置快照的现有存储桶。

   c. 对于通知选项，选择您的首选通知选项。AWS Config 将使用 Amazon Simple Notification Service（Amazon SNS）通知您与您的资源有关的重要 AWS Config 事件。如果选择使用现有 SNS 主题选项，则必须在要使用的账户中提供现有 Amazon SNS 主题的 AWS 账户 ID 和名称。如果要定位多个 AWS 区域，则每个区域中的主题名称必须相同。

5. 在计划部分中，选择 Quick Setup 修复对资源所做的不同于配置的更改时的所需频率。默认值选项运行一次。如果您不希望 Quick Setup 修复对资源所做的不同于配置的更改，请在自定义下选择选择禁用修复。
6. 在目标部分中，选择以下选项之一以确定用于记录的账户和区域。

> **Note**
>
> 如果您使用单一账户，用于组织和组织单位 (OU) 的选项将不可用。您可以选择是将此配置应用于账户 AWS 区域 中的所有区域，还是仅将此配置应用于您选择的区域。

- Entire organization（整个组织）：组织中的所有账户和区域。
- Custom（自定义）：仅限您指定的 OU 和区域。
  - 在目标 OU 部分中，选择要允许进行记录的 OU。
  - 在目标区域部分中，选择要允许进行记录的区域。
- Current account（当前账户）：只有您在当前登录的账户中指定的区域才是目标区域。选择下列选项之一：
  - Current Region（当前区域）：仅以在控制台中选择的区域中的托管节点为目标。
  - 选择区域：选择要将记录配置应用到的各个区域。

7. 选择创建。

使用 Quick Setup 部署 AWS Config 一致性包

一致性包是 AWS Config 规则和修复操作的集合。可在 AWS Organizations 中通过 Quick Setup 将一致性包部署为账户和 AWS 区域或整个组织中的单个实体。这有助于使用通用框架和打包模型大规模管理 AWS 资源的配置合规性，即从策略定义一直到审计和汇总报告。

要部署一致性包，请在 AWS Systems Manager Quick Setup 控制台中执行以下任务。

> **Note**
>
> 必须先启用 AWS Config 记录，然后再部署此配置。有关更多信息，请参阅《AWS Config 开发人员指南》中的一致性包。

使用 Quick Setup 部署一致性包

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Quick Setup。
3. 在一致性包卡片上，选择创建。

> **Tip**
>
> 如果您的账户中已经有一个或多个配置，请先选择库选项卡或配置部分的中的创建按钮以查看卡片。

4. 在选择配置包部分中，选择要部署的一致性包。
5. 在计划部分中，选择 Quick Setup 修复对资源所做的不同于配置的更改时的所需频率。默认值选项运行一次。如果不希望 Quick Setup 修复对资源所做的不同于配置的更改，请在自定义下选择禁用。
6. 在目标部分中，选择是要将一致性包部署到整个组织、其中一些 AWS 区域 还是当前登录的账户。

   如果选择整个组织，继续执行步骤 8。

   如果选择自定义，继续执行步骤 7。
7. 在目标区域部分中，选中要部署一致性包的区域的复选框。
8. 选择创建。

使用 Quick Setup 补丁策略为组织中的实例配置修补

借助Quick Setup（AWS Systems Manager 中的一项工具），您可以创建由Patch Manager提供支持的补丁策略。补丁策略定义了自动修补 Amazon Elastic Compute Cloud (Amazon EC2) 实例和其他托管节点时要使用的计划和基准。使用单一补丁策略配置，您可以为贵组织中多个 AWS 区域 的所有账户定义修补、仅为所选的账户和区域定义修补，或为单个账户区域对定义修补。有关补丁策略的更多信息，请参阅 Quick Setup 中的补丁策略配置。

先决条件

要使用 Quick Setup 为节点定义补丁策略，该节点必须是托管节点。有关管理节点的更多信息，请参阅为组织设置 Systems Manager 统一控制台。

> **Important**
>
> 补丁合规性扫描方法 – Systems Manager 支持多种方法来扫描托管式节点，可以满足补丁合规性。若同时执行上述方法中的多种方法，您看到的补丁合规信息将始终是最近一次扫描的结果。先前扫描的结果将被覆盖。如果扫描方法使用不同的补丁基准，有不同的批准规则，那么补丁合规信息可能会出现意外变更。有关更多信息，请参阅 避免意外覆盖补丁合规性数据。

关联合规性状态和补丁策略 – Quick Setup 补丁策略下的托管式节点的修补状态与该节点的 State Manager 关联执行状态相匹配。如果关联执行状态为 `Compliant`，则还会将托管式节点的修补状态标记为 `Compliant`。如果关联执行状态为 `Non-Compliant`，则还会将托管式节点的修补状态标记为 `Non-Compliant`。

支持补丁策略配置的区域

以下区域当前支持 Quick Setup 中的补丁策略配置：

- 美国东部（俄亥俄州）(us-east-2)
- 美国东部（弗吉尼亚州北部）(us-east-1)
- 美国西部（北加利福尼亚）(us-west-1)
- 美国西部（俄勒冈州）(us-west-2)
- 亚太地区（孟买）(ap-south-1)
- 亚太地区（首尔）(ap-northeast-2)
- 亚太地区（新加坡）(ap-southeast-1)
- 亚太地区（悉尼）(ap-southeast-2)
- 亚太地区（东京）(ap-northeast-1)
- 加拿大（中部）(ca-central-1)
- 欧洲地区（法兰克福）(eu-central-1)
- 欧洲地区（爱尔兰）(eu-west-1)
- 欧洲（伦敦）(eu-west-2)
- 欧洲地区（巴黎）（eu-west-3）
- 欧洲地区（斯德哥尔摩）(eu-north-1)
- 南美洲（圣保罗）（sa-east-1）

补丁策略 S3 存储桶的权限

创建补丁策略时，Quick Setup 会创建一个 Amazon S3 存储桶，其中包含一个名为 `baseline_overrides.json` 的文件。此文件存储了有关为补丁策略指定的补丁基准的信息。

S3 存储桶以 `aws-quicksetup-patchpolicy-`*`account-id`*`-`*`quick-setup-configuration-id`* 格式命名。

例如：aws-quicksetup-patchpolicy-123456789012-abcde

如果您要为组织创建补丁策略，则存储桶将在组织的管理账户中创建。

对于以下两种使用案例，您必须向其他 AWS 资源提供使用 AWS Identity and Access Management（IAM）policy 访问此 S3 存储桶的权限：

- 情况 1：将您自己的实例配置文件或服务角色与托管式节点一起使用，而不是 Quick Setup 提供的实例配置文件或服务角色
- 情况 2：使用 VPC 端点连接到 Systems Manager

无论哪种情况，您需要的权限策略位于下面的 Quick Setup S3 存储桶的策略权限。

情况 1：将您自己的实例配置文件或服务角色与托管式节点一起使用，而不是 Quick Setup 提供的实例配置文件或服务角色

补丁策略配置包括一个选项，用于将所需的 IAM policy 添加到附加于实例的现有实例配置文件中。

如果您不选择此选项，但想要 Quick Setup 使用此补丁策略修补托管式节点，则必须确保实施以下内容：

- IAM 托管式策略 AmazonSSMManagedInstanceCore 必须附加到用于向托管式节点提供 Systems Manager 权限的 IAM 实例配置文件或 IAM 服务角色。
- 您必须将访问补丁策略存储桶的权限添加为 IAM 实例配置文件或 IAM 服务角色的内联策略。您可以提供对所有 aws-quicksetup-patchpolicy 存储桶的通配符访问权限，或仅对为组织或账户创建的特定存储桶提供通配符访问权限，如前面的代码示例所示。
- 您必须使用以下键值对标记 IAM 实例配置文件或 IAM 服务角色。

  Key: QSConfigId-*quick-setup-configuration-id*, Value: *quick-setup-configuration-id*

  quick-setup-configuration-id 表示应用于创建补丁策略配置时使用的 AWS CloudFormation 堆栈的参数值。要检索此 ID，请执行以下操作：

  1. 通过以下网址打开 AWS CloudFormation 控制台：https://console.aws.amazon.com/cloudformation。
  2. 选择用于创建补丁策略的堆栈名称。名称的格式，例 StackSet-AWS-QuickSetup-PatchPolicy-LA-q4bkg-52cd2f06-d0f9-499e-9818-d887cEXAMPLE。
  3. 选择参数选项卡。

4. 在参数列表的密钥列中，找到密钥 QSConfigurationId。在其行的值列中，找到配置 ID，例如 abcde。

   在此示例中，对于要应用于实例配置文件或服务角色的标签，密钥为 QSConfigId-abcde，值为 abcde。

有关向 IAM 角色添加标签的信息，请参阅《IAM 用户指南》中的标记 IAM 角色和管理实例配置文件（AWS CLI 或 AWS API）的标签。

情况 2：使用 VPC 端点连接到 Systems Manager

如果您使用 VPC 端点连接到 Systems Manager，则 S3 的 VPC 端点策略必须允许访问 Quick Setup 补丁策略 S3 存储桶。

有关为 S3 的 VPC 端点策略添加权限的信息，请参阅《Amazon S3 用户指南》中的使用桶策略控制从 VPC 端点的访问。

Quick Setup S3 存储桶的策略权限

您可以提供对所有 aws-quicksetup-patchpolicy 存储桶的通配符访问权限，或仅对为组织或账户创建的特定存储桶提供通配符访问权限。要为下述两种情况提供必要的权限，请使用任一格式。

All patch policy buckets

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccessToAllPatchPolicyRelatedBuckets",
      "Effect": "Allow",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::aws-quicksetup-patchpolicy-*"
    }
  ]
}
```

Specific patch policy bucket

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccessToMyPatchPolicyRelatedBucket",
      "Effect": "Allow",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::aws-quicksetup-patchpolicy-111122223333-quick-setup-configuration-id"
    }
  ]
}
```

[1] 创建补丁策略配置后，您可以在 S3 控制台中找到存储桶的全名。例如：aws-quicksetup-patchpolicy-123456789012-abcde

补丁策略操作中的随机补丁基准 ID

补丁策略的修补操作使用 AWS-RunPatchBaseline SSM Command 文档中的 BaselineOverride 参数。

您使用 AWS-RunPatchBaseline 在修补策略之外进行修补时，可以使用 BaselineOverride 指定要在操作期间使用的补丁基准列表，这些基准不同于指定的默认基准。您在名为 baseline_overrides.json 的文件中创建此列表，并将其手动添加到您拥有的 Amazon S3 存储桶，如 使用基准覆盖参数 中所述。

但是，对于基于补丁策略的修补操作，Systems Manager 会自动创建 S3 存储桶并向其中添加 baseline_overrides.json 文件。然后，Quick Setup每次运行修补操作（使用 Run Command 工具）时，系统都会为每个补丁基准生成一个随机 ID。对于每个补丁策略修补操作，此 ID 都是不同的，并且它代表的补丁基准不会在您的账户中存储或访问。

因此，您将不会在修补日志中看到配置中选择的补丁基准 ID。这适用于 AWS 托管式补丁基准和您可能选择的自定义补丁基准。日志中报告的基准 ID 是为此特定修补操作生成的基准 ID。

此外，如果您尝试在 Patch Manager 中查看有关使用随机 ID 生成的补丁基准的详细信息，系统会报告此补丁基准不存在。此为预期行为，可以安全忽略。

创建补丁策略

要创建补丁策略，请在 Systems Manager 控制台中执行以下任务。

使用 Quick Setup 创建补丁策略

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。

   如果您正在为组织设置修补，请确保您已登录到该组织的管理账户。您无法使用委派管理员账户或成员账户设置策略。
2. 在导航窗格中，请选择 Quick Setup。
3. 在 Patch Manager（补丁管理器）选项卡上，选择 Create（创建）。

   > **Tip**
   >
   > 如果您的账户中已经有一个或多个配置，请先选择库选项卡或配置部分的中的创建按钮以查看卡片。

4. 在 Configuration name（配置名称）中输入名称以帮助识别补丁策略。
5. 在 Scanning and installation（扫描和安装）部分的 Patch operation（补丁操作）下，选择补丁策略是 Scan（扫描）指定目标还是在指定目标 Scan and install（扫描并安装）补丁。
6. 在 Scanning schedule（扫描计划）下，选择 Use recommended defaults（使用推荐的默认值）或 Custom scan schedule（自定义扫描计划）。默认扫描计划将在世界标准时间每天凌晨 1 点扫描目标。

   - 如果选择 Custom scan schedule（自定义扫描计划），请选择 Scanning frequency（扫描频率）。
   - 如果您选择 Daily（每日），请以 UTC 为单位输入要扫描目标的时间。
   - 如果选择 Custom CRON Expression（自定义 CRON 表达式），请以 CRON 表达式的形式输入计划。有关为 Systems Manager 设置 CRON 表达式的更多信息，请参阅 参考：适用于 Systems Manager 的 Cron 和 Rate 表达式。

     此外，选择 Wait to scan targets until first CRON interval（等到第一个 CRON 间隔才扫描目标）。默认情况下，在节点成为目标时 Patch Manager 立即对其进行扫描。

7. 如果选择 Scan and install（扫描并安装），请选择向指定目标安装补丁时要使用的 Installation schedule（安装计划）。如果您选择 Use recommended defaults（使用推荐的默认值），则 Patch Manager 将在世界标准时间星期天凌晨 2 点安装补丁。

   - 如果选择 Custom install schedule（自定义安装计划），请选择 Installation frequency（安装频率）。
   - 如果您选择 Daily（每日），请输入要对目标更新安装的时间。
   - 如果选择 Custom CRON expression（自定义 CRON 表达式），请以 CRON 表达式的形式输入计划。有关为 Systems Manager 设置 CRON 表达式的更多信息，请参阅 参考：适用于 Systems Manager 的 Cron 和 Rate 表达式。

     此外，清除 Wait to install updates until first CRON interval（等到第一个 CRON 间隔才安装更新），以便在节点成为目标时立即在节点上安装更新。默认情况下，Patch Manager 等到第一个 CRON 间隔出现才会安装更新。
   - 安装补丁后，请选择 Reboot if needed（需要时重新启动）以重新启动节点。建议在安装后重新启动，但可能会导致可用性问题。

8. 在 Patch baseline（补丁基准）部分中，选择扫描和更新目标时要使用的补丁基准。

   默认情况下，Patch Manager 使用预定义的补丁基准。有关更多信息，请参阅 预定义基准。

   如果您选择 Custom patch baseline（自定义补丁基准），请为不想使用预定义 AWS 补丁基准的操作系统更改所选补丁基准。

> **Note**
>
> 如果您使用 VPC 端点连接到 Systems Manager，确保 S3 的 VPC 端点策略必须允许访问此 S3 存储桶。有关更多信息，请参阅 补丁策略 S3 存储桶的权限。

> **Important**
>
> 如果您在 Quick Setup 中使用补丁策略配置，则系统会每小时与 Quick Setup 同步一次您对自定义补丁基准所做的更新。
> 如果删除了补丁策略中引用的自定义补丁基准，那么补丁策略的 Quick Setup Configuration details（配置详细信息）页面上会显示一个横幅。横幅将通知您，补丁策略引用的补丁基准已不存在，且后续的修补操作将失败。在这种情况下，返回 Quick Setup Configurations（配置）页面，选择 Patch Manager 配置，然后选择 Actions（操

作），Edit configuration（编辑配置）。已删除的补丁基准名称将突出显示，您必须为受影响的操作系统选择新的补丁基准。

9. （可选）在 Patching log storage（修补日志存储）部分中，选择 Write output to S3 bucket（将输出写入 S3 存储桶），将补丁操作日志存储在 Amazon S3 存储桶中。

> **Note**
>
> 如果您要为组织设置补丁策略，则您的组织的管理账户必须至少具有此存储桶的只读权限。策略中包含的所有组织单位都必须具有存储桶的写入权限。有关向不同账户授予存储桶访问权限的信息，请参阅 Amazon Simple Storage Service 用户指南中的示例 2：存储桶所有者授予跨账户存储桶权限。

10. 选择浏览 S3，以选择要在其中存储补丁日志输出的存储桶。管理账户必须具有对此存储桶的读取权限。在 Targets（目标）部分配置的所有非管理账户和目标都必须具有对所提供的 S3 存储桶的写入权限才能进行记录。
11. 在 Targets（目标）部分中，选择以下选项之一以确定此补丁策略操作的账户和区域。

> **Note**
>
> 如果您使用单一账户，用于组织和组织单位 (OU) 的选项将不可用。您可以选择是将此配置应用于账户 AWS 区域 中的所有区域，还是仅将此配置应用于您选择的区域。
> 如果您之前为自己的账户指定了主区域，但尚未加入新的 Quick Setup 主机体验，则无法将该区域从目标配置中排除。

- Entire organization（整个组织）：组织中的所有账户和区域。
- Custom（自定义）：仅限您指定的 OU 和区域。
  - 在 Target OUs（目标组织单位）部分，选择要设置补丁策略的组织单位。
  - 在 Target Regions（目标区域）部分，选择要设置补丁策略的区域。
- Current account（当前账户）：只有您在当前登录的账户中指定的区域才是目标区域。选择下列选项之一：
  - Current Region（当前区域）：仅以在控制台中选择的区域中的托管节点为目标。
  - Choose Regions（选择区域）：选择要将补丁策略应用到的各个区域。

12. 对于 Choose how you want to target instances（选择您希望如何定位实例），请选择以下选项之一来标记要修补的节点：

- All managed nodes（所有托管节点）：选定 OU 和区域中的所有托管节点。
- Specify the resource group（指定资源组）：从列表中选择资源组的名称以将其关联资源作为目标。

> **Note**
>
> 目前，仅单个账户配置支持选择资源组。要修补多个账户中的资源，请选择不同的目标选项。

- Specify a node tag（指定节点标签）：只有使用您指定的键值对标记的节点才会在您的目标账户和区域中得到修补。
- Manual（手动）：从列表中手动选择所有指定账户和区域的托管节点。

> **Note**
>
> 此选项目前仅支持 Amazon EC2 实例。您可以在补丁策略配置中最多手动添加 25 个实例。

13. 在 Rate control（速率控制）部分，执行以下操作：

    - 对于 Concurrency（并发），输入要同时运行补丁策略的节点数量或百分比。
    - 对于 Error threshold（错误阈值），输入在补丁策略失败之前可能出现错误的节点数量或百分比。

14. 选择将所需的 IAM policy 添加到附加到实例的现有实例配置文件复选框。

    此选择会将此 Quick Setup 配置创建的 IAM policy 应用于已附加实例配置文件（EC2 实例）或服务角色（混合激活节点）的节点。若您的托管式节点已经附加实例配置文件或服务角色，但它不包含使用 Systems Manager 所需的所有权限，我们推荐选择该选项。

    此处的选择将应用于稍后在此补丁策略配置应用的账户和区域中创建的托管节点。

> **Important**
>
> 如果您不选中此复选框，但想要 Quick Setup 使用此补丁策略修补托管式节点，则必须确保实施以下内容：
> 向您的 IAM 实例配置文件或 IAM 服务角色添加权限，以访问为补丁策略创建的 S3 存储桶使用特定的键值对标记 IAM 实例配置文件或 IAM 服务角色。

有关信息，请参阅情况 1：将您自己的实例配置文件或服务角色与托管式节点一起使用，而不是 Quick Setup 提供的实例配置文件或服务角色。

15. 选择创建。

在创建补丁策略后要查看补丁状态，您可以从 Quick Setup 页面访问配置。

使用 Quick Setup 设置 DevOps Guru

可通过使用 Quick Setup 快速配置 DevOps Guru 选项。Amazon DevOps Guru 是由机器学习 (ML) 支持的服务，可轻松提高应用程序的运行性能和可用性。DevOps Guru 会检测到与正常操作模式不同的行为，因此可尽早识别运行问题，以免影响客户。DevOps Guru 会自动从 AWS 应用程序提取数据，并提供单一控制面板，用于可视化运行数据中的问题。无需手动设置或具备机器学习专业知识即可开始使用 DevOps Guru，以提高应用程序可用性和可靠性。

可以在以下 AWS 区域 中使用 Quick Setup 配置 DevOps Guru：

- 美国东部（弗吉尼亚州北部）
- 美国东部（俄亥俄州）
- 美国西部（俄勒冈州）
- 欧洲地区（法兰克福）
- 欧洲地区（爱尔兰）
- 欧洲地区（斯德哥尔摩）
- 亚太地区（新加坡）
- 亚太地区（悉尼）
- 亚太地区（东京）

有关定价信息，请参阅 Amazon DevOps Guru 定价。

要设置 DevOps Guru，请在 AWS Systems Manager Quick Setup 控制台中执行以下操作。

使用 Quick Setup 设置 DevOps Guru

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Quick Setup。

3. 在 DevOps Guru 卡片上，选择创建。

> **Tip**
>
> 如果您的账户中已经有一个或多个配置，请先选择库选项卡或配置部分的中的创建按钮以查看卡片。

4. 在配置选项部分，选择要分析的 AWS 资源类型以及通知首选项。

   如果未选择分析我的组织中所有账户内的所有 AWS 资源选项，则可以稍后在 DevOps Guru 控制台中选择要分析的 AWS 资源。DevOps Guru 会分析不同的 AWS 资源类型（例如 Amazon Simple Storage Service (Amazon S3) 存储桶和 Amazon Elastic Compute Cloud (Amazon EC2) 实例），这些资源类型将按两个定价组进行分类。您需要为每个活动资源支付已分析的 AWS 资源小时数。仅当资源在一小时内生成指标、事件或日志条目时，该资源才处于活动状态。为特定 AWS 资源类型支付的价格取决于价格组。

   如果选择了启用 SNS 通知选项，将在使用配置定位的组织单元 (OU) 的每个 AWS 账户中创建 Amazon Simple Notification Service (Amazon SNS) 主题。DevOps Guru 将使用该主题通知您重要的 DevOps Guru 事件，例如新建洞察。如果不启用此选项，则可以稍后在 DevOps Guru 控制台中添加主题。

   如果选择启用 AWS Systems Manager OpsItems 选项，则将为相关的 Amazon EventBridge 事件和 Amazon CloudWatch 警报创建操作工作项 (OpsItems)。

5. 在计划部分中，选择 Quick Setup 修复对资源所做的不同于配置的更改时的所需频率。默认值选项运行一次。如果不希望 Quick Setup 修复对资源所做的不同于配置的更改，请在自定义下选择禁用。

6. 在目标部分中，选择是允许 DevOps Guru 分析其中一些组织单位 (OU) 中的还是当前登录的账户中的资源。

   如果选择自定义，继续执行步骤 8。

   如果选择当前账户，继续执行步骤 9。

7. 在目标 OU 和目标区域部分中，选中要使用 DevOps Guru 的 OU 和区域的复选框。

8. 选择当前账户中要使用 DevOps Guru 的区域。

9. 选择创建。

使用 Quick Setup 部署 Distributor 软件包

Distributor 是 AWS Systems Manager 中的一项工具。Distributor 软件包是可以作为单个实体部署的可安装软件或资产的集合。可以在 AWS Organizations 中使用 Quick Setup 将 Distributor 软件包部署到 AWS 账户和 AWS 区域或整个组织中。目前，只能使用 Quick Setup 部署 EC2Launch v2 代理、Amazon Elastic File System（Amazon EFS）实用工具包和 Amazon CloudWatch 代理。有关 Distributor 的更多信息，请参阅 AWS Systems Manager Distributor。

要部署 Distributor 软件包，请在 AWS Systems Manager Quick Setup 控制台中执行以下任务。

使用Quick Setup 部署 Distributor 软件包

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Quick Setup。
3. 在分发卡片上，选择创建。

   > **Tip**
   >
   > 如果您的账户中已经有一个或多个配置，请先选择库选项卡或配置部分的中的创建按钮以查看卡片。

4. 在配置选项部分中，选择要部署的软件包。
5. 在目标部分中，选择是将软件包部署到整个组织、其中一些组织单位 (OU) 还是当前登录的账户。

   如果选择整个组织，继续执行步骤 8。

   如果选择自定义，继续执行步骤 7。
6. 在目标 OU 部分中，选中要部署软件包的 OU 和区域的复选框。
7. 选择 Create（创建）。

按计划使用 Quick Setup 自动停止和启动 EC2 实例

借助Quick Setup（AWS Systems Manager 中的一项工具），您可以配置资源调度器自动启动和停止 Amazon Elastic Compute Cloud（Amazon EC2）实例。

此 Quick Setup 配置可根据您指定的计划启动和停止实例，从而帮助您降低运营成本。该工具可帮助您避免在不需要时运行实例而产生的不必要成本。

例如，您目前可能让实例持续运行，即使它们每周运行 5 天、每天只运行 10 个小时。相反，您可以安排实例每天下班后停止运行。由于运行时间从 168 小时减少到 50 小时，实例运行成本将节省 70%。使用 Quick Setup 不会收取任何费用。但是，根据您设置的资源和使用限制，可能会产生成本，但不收取用于设置配置的服务费用。

使用资源调度器，您可以选择根据自己定义的调度在多个 AWS 区域和 AWS 账户自动停止和启动实例。该 Quick Setup 配置使用您指定的标签密钥和值以 Amazon EC2 实例为目标。资源调度程序只会停止或启动具有与配置中指定的值匹配的标记的实例。请注意，如果附加到实例的 Amazon EBS 卷已加密，则必须将 AWS KMS 密钥所需的权限添加到 IAM 角色中，资源调度器才能启动实例。

每种配置的最大实例数

单个配置每个区域最多可调度 5000 个实例。如果您的用例要求在给定区域中调度超过 5000 个实例，则必须创建多个配置。相应地标记实例，以便每个配置最多管理 5000 个实例。创建多个资源调度程序 Quick Setup 配置时，必须指定不同的标签密钥值。例如，一种配置可以使用 `Environment` 标签键和 `Production` 标签值，而另一种配置则使用 `Environment` 标签键和 `Development` 标签值。

调度行为

以下几点介绍了调度配置的某些行为：

- 只有当带标签的实例处于 `Stopped` 状态时，资源调度程序才会启动这些实例。同样，只有实例处于 `running` 状态时才会停止。资源调度程序在事件驱动的模型上运行，并且仅在您指定的时间启动或停止实例。例如，您创建了一个在上午 9 点启动实例的计划。资源调度程序在上午 9 点启动所有与您指定的标签关联且处于 `Stopped` 状态的实例。如果稍后手动停止实例，资源调度程序将不会再次启动它们以保持 `Running` 状态。同样，如果在实例按计划停止后进行手动启动，则资源调度程序不会再次停止该实例。
- 如果创建的调度开始时间晚于停止时间（24 小时制），资源调度器假定实例会过夜运行。例如，您创建了一个计划，上午 9 点启动实例，上午 7 点停止实例。资源调度程序在上午 9 点启动所有与您指定的标签关联且处于 `Stopped` 状态的实例，并在第二天上午 7 点停止实例。对于整夜运行计划，开始时间为您为计划选择的日期。但停止时间为计划的第二天。
- 创建调度配置时，为满足调度的要求，可能会更改实例的当前状态。

  例如，假设今天是星期三，您指定托管式实例的调度仅在周二和周四上午 9:00 开始，下午 5:00 停止。实例当前的运行时间已超出规定的运行时间，因此这些实例会在配置创建后停止。这些实例要等到下一个规定的时间，即周四上午 9:00 才会再次运行。

  如果实例当前处于 `Stopped` 状态，并且您指定了调度使实例在当前时间运行，则资源调度器会在创建配置后启动实例。

如果您删除配置，将不再根据先前定义的计划停止和启动实例。在极少数情况下，由于 API 操作失败，可能无法成功停止或启动实例。

要设置 Amazon EC2 实例调度。请在 AWS Systems Manager Quick Setup 控制台中执行以下任务。

使用 Quick Setup 设置实例调度

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Quick Setup。
3. 在资源调度器卡片上，选择创建。

> **Tip**
>
> 如果您的账户中已经有一个或多个配置，请先选择库选项卡或配置部分的中的创建按钮以查看卡片。

4. 在 Instance tag（实例标签）部分，指定要应用于与计划关联的实例的标签密钥和值。
5. 在 Schedule options（计划选项）部分，指定您想要启动和停止实例的时区、日期和时间。
6. 在 Targets（目标）部分，选择是为 Custom（自定义）组织单位 (OU) 还是您登录的 Current account（当前账户）设置调度：
   - Custom（自定义）：在 Target OUs（目标组织单位）部分，选择要设置主机管理的组织单位。然后，在 Target Regions（目标区域）部分，选择要设置计划的区域。
   - 当前账户 - 选择当前区域或选择区域。如果您选择了 Choose Regions（选择区域），请选择要在其中设置调度的 Target Regions（目标区域）。
7. 验证 Summary（摘要）部分中的计划信息。
8. 选择创建。

使用 Quick Setup 配置 AWS 资源探索器

您可以使用Quick Setup（AWS Systems Manager 中的一项工具）快速配置 AWS 资源探索器，进而搜索并发现 AWS 账户或整个 AWS 组织中的资源。您可以使用名称、标签和 ID 等元数据来搜索资源。AWS 资源探索器 使用索引来快速响应搜索查询。资源探索器使用各种数据来源创建并维护索引，进而收集 AWS 账户 中关于资源的信息。

资源探索器的 Quick Setup 可自动执行索引配置过程。有关 AWS 资源探索器 的更多信息，请参阅《AWS 资源探索器 用户指南》中什么是 AWS 资源探索器？的内容。

在 Quick Setup 期间，资源探索器执行以下操作：

- 在您的 AWS 账户 中的每一个 AWS 区域 中创建索引。
- 更新您指定为账户聚合器索引的区域中的索引。
- 在聚合器索引区域中创建一个默认视图。此视图没有筛选条件，因此它会返回在索引中找到的所有资源。

最小权限

要执行下列程序中的步骤，您必须具有以下权限：

- 操作：`resource-explorer-2:*` – 资源：无特定资源（*）
- 操作：`iam:CreateServiceLinkedRole` – 资源：无特定资源（*）

配置资源探索器

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Quick Setup。
3. 在资源探索器卡片上，选择创建。
4. 在聚合器索引区域部分，选择要包含聚合器索引的区域。您应该选择适合用户地理位置的区域。
5. （可选）选中替换除上面所选区域以外的区域中的现有聚合器索引复选框。
6. 在目标部分，选择包含要发现的资源的目标组织或特定组织单位（OU）。
7. 在区域部分，选择要包含在配置中的区域。
8. 查看配置摘要，然后选择创建。

支持在资源探索器页面上监控配置状态。

对 Quick Setup 结果进行故障排除

利用以下信息，帮助排查Quick Setup（AWS Systems Manager 中的一项工具）出现的问题。本主题介绍了依据 Quick Setup 问题类型来解决问题的特定任务。

问题：部署失败

如果 CloudFormation 堆栈集在创建过程中失败，则部署失败。使用以下步骤调查部署失败。

1. 导航至 AWS CloudFormation 控制台。
2. 请选择您的 Quick Setup 配置创建的堆栈。Stack name（堆栈名称）包括 `QuickSetup`，后跟您选择的配置类型，例如 `SSMHostMgmt`。

   > **Note**
   >
   > CloudFormation 有时会删除失败的堆栈部署。如果堆栈未提供在 Stacks（堆栈）表中，请从筛选条件列表中选择 Deleted（已删除）。

3. 请查看 Status（状态）和 Status reason（状态原因）。有关堆栈状态的更多信息，请参阅 AWS CloudFormation 用户指南中的堆栈状态代码。
4. 要了解失败的确切步骤，请查看 Events（事件）选项卡并查看每个事件的状态。
5. 请查看 AWS CloudFormation 用户指南中的故障排除。
6. 如果您无法使用 CloudFormation 故障排除步骤解决部署失败，请删除配置并重新配置它。

问题：关联失败

如果设置过程中有任何关联失败，则您的配置的 Configuration details（配置详细信息）页面上的 Configuration details（配置详细信息）表将会显示 Configuration status（配置状态）为 Failed（失败）。使用以下步骤对失败的关联进行故障排除。

1. 在 Configuration details（配置详细信息）表中，请选择失败的配置，然后选择 View Details（查看详细信息）。
2. 复制关联名称。
3. 导航到 State Manager，然后将关联名称粘贴到搜索字段中。
4. 选择关联，然后选择 Execution history（执行历史记录）选项卡。
5. 在执行 ID 下面，选择失败的关联执行。
6. Association execution targets（关联执行目标）页面列出运行关联的所有节点。为无法运行的执行选择输出按钮。
7. 在输出页面中，选择步骤 - 输出以在命令执行中查看该步骤的错误消息。每个步骤可能会显示不同的错误消息。请查看所有步骤的错误消息以帮助解决问题。

如果查看步骤输出无法解决问题，您可以尝试重新创建关联。要重新创建关联，请先在 State Manager 中删除失败的关联。删除关联后，编辑配置并选择您删除的选项，然后选择 Update（更新）。

> **Note**
>
> 要调查组织配置的失败关联，必须登录存在失败关联的账户，然后使用以下失败关联过程，如前所述。当查看来自管理账户的结果时，关联 ID 不是指向目标账户的超链接。

问题：偏差状态

在查看配置的详细信息页面时，您可以查看每个部署的偏差状态。当用户对与通过 Quick Setup 进行的选择冲突的服务或功能进行更改时，会发生配置偏差。如果在初始配置之后关联发生了更改，则表格将显示一个警告图标，指示已偏差的项目数量。您可以通过将鼠标悬停在图标上来确定造成偏差的原因。

在 State Manager 中删除关联时，相关部署会显示偏差警告。要修复此问题，请编辑配置并选择删除关联时删除的选项。请选择 Update（更新）并等待部署完成。

AWS Systems Manager 应用程序工具

应用程序工具是一套功能，可帮助您管理在 AWS 中运行的应用程序。

主题

- AWS AppConfig
- AWS Systems Manager Application Manager
- AWS Systems Manager Parameter Store

AWS AppConfig

有关 AWS AppConfig 的信息已移到单独的指南中。有关更多信息，请参阅下列内容：

- 《AWS AppConfig 用户指南》
- AWS AppConfig API 参考

AWS Systems Manager Application Manager

Application Manager 是 AWS Systems Manager 中的一项工具，让 DevOps 工程师能够结合其应用程序和集群的上下文调查和修复其 AWS 资源的问题。Application Manager 会将来自多个 AWS 服务和 Systems Manager 工具的操作信息聚合到单个 AWS Management Console。

在 Application Manager 中，应用程序是一个您希望其作为一个单元运行的 AWS 资源的逻辑组。此逻辑组可以表示应用程序的不同版本、操作员的所有权边界或开发人员环境等。对容器集群的 Application Manager 支持包括 Amazon Elastic Kubernetes Service (Amazon EKS) 和 Amazon Elastic Container Service (Amazon ECS) 集群。

首次打开 Application Manager 时，会显示Application Manager 能为您做什么页面。选择开始使用时，Application Manager 会自动导入有关在其他 AWS 服务或 Systems Manager 工具中创建的有关资源的元数据。然后，Application Manager 在按预定义类别分组的列表中显示这些资源。

对于应用程序，列表包含以下内容：

- AWS CloudFormation 堆栈
- 自定义应用程序
- AWS Launch Wizard 应用程序
- AppRegistry 应用程序
- AWS SAP 企业工作负载应用程序
- Amazon ECS 集群
- Amazon EKS 集群

在设置并配置 AWS 服务和 Systems Manager 工具后，Application Manager 会显示与您资源有关的以下类型的信息：

- 有关应用程序中的 Amazon Elastic Compute Cloud (Amazon EC2) 实例的当前状态、情况和 Amazon EC2 Auto Scaling 运行状况的信息
- Amazon CloudWatch 提供的警报
- AWS Config 和 State Manager（Systems Manager 的一个组件）提供的合规性信息
- Amazon EKS 提供的 Kubernetes 集群信息
- AWS CloudTrail 和 Amazon CloudWatch Logs 提供的日志数据
- Systems Manager OpsCenter 提供的 OpsItems
- 托管相关资源的 AWS 服务提供的资源详细信息。
- Amazon ECS 提供的容器集群信息。

为了帮助您修复与组件或资源有关的问题，Application Manager 还提供了可与应用程序关联的运行手册。要开始使用 Application Manager，请打开 Systems Manager 控制台。在导航窗格中，请选择 Application Manager。

使用 Application Manager 有哪些优势？

Application Manager 减少了 DevOps 工程师检测和调查 AWS 资源问题的时间。为实现此目的，Application Manager 在一个控制台中显示应用程序上下文中的许多类型的操作信息。Application Manager 还可以通过提供运行手册来执行常见的 AWS 资源修正任务，从而减少修复问题所需的时间。

Application Manager 具有哪些功能？

Application Manager 包括以下功能：

- 自动导入 AWS 资源

  在初始设置过程中，您可以选择让 Application Manager 在 AWS 账户 中自动导入和显示基于 CloudFormation 堆栈、AWS Resource Groups、Launch Wizard 部署、APpregistry 应用程序以及 Amazon ECS 和 Amazon EKS 集群的资源。系统将以预定义的应用程序或集群类别显示这些资源。此后，无论何时将这些类型的新资源添加到 AWS 账户，Application Manager 都会在预定义的应用程序和集群类别中自动显示新资源。

- 创建或编辑 CloudFormation 堆栈和模板

  Application Manager 可帮助您配置和管理应用程序的资源，方法是与 CloudFormation 集成。您可以在 Application Manager 中创建、编辑和删除 AWS CloudFormation 模板和堆栈。Application Manager 还包括一个模板库，您可以在其中克隆、创建和存储模板。Application Manager 和 CloudFormation 显示有关堆栈当前状态的相同信息。模板和模板更新存储在 Systems Manager 中，直到您配置堆栈为止，届时更改也会显示在 CloudFormation 中。

- 查看应用程序上下文中有关实例的信息

  Application Manager 与 Amazon Elastic Compute Cloud (Amazon EC2) 集成，以在应用程序上下文中显示有关实例的信息。Application Manager 以图形格式显示所选应用程序的实例状态、情况和 Amazon EC2 Auto Scaling 运行状况。Instances（实例）选项卡还包括一个表，其中包含应用程序中每个实例的以下信息：

  - 实例状态（待处理、正在停止、正在运行、已停止）
  - SSM Agent 的 Ping 状态
  - 在实例上处理的最新 Systems Manager Automation 运行手册的状态和名称
  - 每个州的 Amazon CloudWatch Logs 警报数量。
    - ALARM – 指标或表达式超出定义的阈值。
    - OK – 指标或表达式在定义的阈值范围内。

    - INSUFFICIENT_DATA（数据不足）– 告警刚刚启动，指标不可用，或者指标没有足够的数据以确定告警状态。
  - 父组和单个自动扩缩组的自动扩缩组运行状况
- 查看应用程序或集群的运行指标和警报

  Application Manager 与 Amazon CloudWatch 集成，为应用程序或集群提供实时操作指标和警报。您可以深入查看应用程序树以查看每个组件级别上的警报，或查看单个集群的警报。
- 查看应用程序的日志数据

  Application Manager 与 Amazon CloudWatch Logs 集成，在您的应用程序上下文中提供日志数据，而无需离开 Systems Manager。
- 查看和管理用于应用程序或集群的 OpsItems

  Application Manager 与 AWS Systems Manager OpsCenter 集成提供操作工作项目列表 (OpsItems)，用于您的应用程序和集群。该列表反映了自动生成和手动创建的 OpsItems。您可以查看有关创建 OpsItem 的资源的详细信息，以及 OpsItem 状态、源和严重性。
- 查看应用程序或集群的资源合规性数据

  Application Manager 与 AWS Config 集成，以根据您指定的规则提供有关您的 AWS 资源的合规和历史详细信息。Application Manager 也与 AWS Systems Manager State Manager 集成，以提供有关要维护的 Amazon Elastic Compute Cloud (Amazon EC2) 实例状态的合规性信息。
- 查看 Amazon ECS 和 Amazon EKS 集群基础设施信息

  Application Manager 与 Amazon ECS 和 Amazon EKS 集成，提供有关集群基础架构运行状况的信息，以及集群中计算、网络和存储资源的组件运行时视图。

  但是，您无法在 Application Manager 管理或查看有关 Amazon EKS 窗格或容器的操作信息。您只能管理和查看托管 Amazon EKS 资源的基础设施的运行信息。
- 查看应用程序的资源成本详细信息

  Application Manager 通过 Cost 小组件与 AWS 账单与成本管理 功能 AWS Cost Explorer 集成。在账单和成本管理控制台中启用 Cost Explorer 后，Application Manager 中的 Cost 小组件会显示特定非容器应用程序或应用程序组件的成本数据。您可以根据条形图或折线图中的不同时段、精细度和成本类型，使用小组件中的筛选条件查看成本数据。
- 在一个控制台中查看详细的资源信息

选择 Application Manager 列出的一个资源名称，并查看有关该资源的上下文信息和操作信息，而无需离开 Systems Manager。

- 接收应用程序的自动资源更新

  如果您对服务控制台中的资源进行了更改，并且该资源是 Application Manager 中某个应用程序的一部分，则 Systems Manager 会自动显示这些更改。例如，如果您在 AWS CloudFormation 控制台中更新了一个堆栈，并且该堆栈是 Application Manager 应用程序的一部分，则堆栈更新会自动反映在 Application Manager 中。

- 自动发现 Launch Wizard 应用程序

  Application Manager 已与 AWS Launch Wizard 集成。如果使用“Launch Wizard”部署应用程序的资源，则 Application Manager 可以将其自动导入“Launch Wizard”部分并显示显示。

- 使用 CloudWatch Application Insights 监控 Application Manager 中的应用程序资源

  Application Manager 与 Amazon CloudWatch Application Insights 集成。Application Insights 可在应用程序资源和技术堆栈中指定并设置关键指标、日志和告警。Application Insights 会持续监控指标和日志，以检测异常情况和错误并将它们关联起来。在系统检测到错误和异常情况时，Application Insights 生成 CloudWatch Events，可以使用这些事件来设置通知或执行操作。您可以在 Application Manager 的概览和监控选项卡中启用和查看 Application Insights。有关 Application Insights 的更多信息，请参阅 Amazon CloudWatch 用户指南中的什么是 Amazon CloudWatch Application Insights。

- 修正运行手册中的问题

  Application Manager 包括预定义的 Systems Manager 运行手册，用于修复 AWS 资源的问题。您可以对应用程序中的所有适用资源执行运行手册，而无需离开 Application Manager。

使用 Application Manager 是否需要收取费用？

Application Manager 不收取额外费用。

Application Manager 的资源配额是什么？

在《Amazon Web Services 一般参考》的 Systems Manager service quotas 中，您可以查看所有 Systems Manager 工具的限额。除非另有说明，否则，每个配额是区域特定的。

主题

- 设置相关服务

- 为 Systems Manager Application Manager 配置权限
- 将应用程序和容器集群添加到 Application Manager
- 使用 应用程序

设置相关服务

Application Manager（AWS Systems Manager 中的一项工具）会显示来自其他 AWS 服务和 Systems Manager 工具的资源及信息。为了最大限度增加在 Application Manager 中显示的操作信息数量，建议在使用 Application Manager 前设置和配置这些其他服务或工具。

主题

- 设置导入资源的任务
- 设置查看资源相关运行信息的任务

设置导入资源的任务

以下设置任务可帮助您查看 Application Manager 中的 AWS 资源。每个任务完成后，Systems Manager 都可以自动将资源导入到 Application Manager 中。导入资源后，您可以在 Application Manager 中创建应用程序，并将导入的资源移动到这些应用程序中。这有助于您查看应用程序上下文中的运行信息。

（可选）使用标签组织您的 AWS 资源

可以将自己的元数据以标签形式分配给 AWS 资源。每个标签都是由用户定义的键和值组成的标签。标签可帮助您管理、识别、组织、搜索和筛选资源。您可以创建标签，以按用途、所有者、环境或其他标准对资源进行分类。

（可选）使用 AWS Resource Groups 组织您的 AWS 资源

您可以使用资源组来组织 AWS 资源。资源组帮助您轻松地同时在大量资源上管理、监控和自动执行任务。

Application Manager 会自动导入您的所有资源组，并将它们在定制应用程序类别中列出。

（可选）使用 AWS CloudFormation 设置和部署您的 AWS 资源

AWS CloudFormation 可以帮助您以可预测、可重复的方式创建和预置 AWS 基础设施部署。它可以帮助您使用 AWS 服务，例如 Amazon EC2、Amazon Elastic Block Store（Amazon EBS）、Amazon Simple Notification Service（Amazon SNS）、Elastic Load Balancing 和 AWS

Auto Scaling。借助 CloudFormation，您可以在云中构建可靠、可扩展且经济高效的应用程序，而无需创建和配置底层 AWS 基础设施。

Application Manager 会自动导入您的所有 AWS CloudFormation 资源，并将其在AWS CloudFormation堆栈类别中列出。您可以在 Application Manager 中创建 CloudFormation 堆栈和模板。堆栈和模板更改会在 Application Manager 和 CloudFormation 之间自动同步。您也可以在 Application Manager 中创建应用程序并将堆栈移动到其中。这有助于您在应用程序上下文中查看堆栈中资源的运行信息。有关定价信息，请参阅 AWS CloudFormation 定价。

（可选）使用 AWS Launch Wizard 设置和部署您的应用程序

Launch Wizard 可以引导您完成调整大小、配置和部署用于第三方应用程序的 AWS 资源的过程，而无需手动识别和配置单个 AWS 资源。

Application Manager 会自动导入所有 Launch Wizard 资源，并将其在 Launch Wizard 类别中列出。有关 AWS Launch Wizard 的更多信息，请参阅 SQL Server AWS Launch Wizard 入门。Launch Wizard 不收取额外费用。您仅需为您配置用于运行解决方案的 AWS 资源付费。

（可选）使用 Amazon ECS 和 Amazon EKS 设置和部署容器化应用程序

Amazon Elastic Container Service (Amazon ECS) 是一项高度可扩展的快速容器管理服务，可帮助轻松运行、停止和管理集群上的容器。您的容器是在用于运行单个任务或服务内任务的任务定义中定义的。

Amazon EKS 是一种托管服务，可帮助您在 AWS 上运行 Kubernetes，无需安装、操作和维护您自己的 Kubernetes 控制平面或节点。Kubernetes 是一个开源系统，用于实现容器化应用程序的部署、扩展和管理的自动化。

Application Manager 自动导入您的所有 Amazon ECS 和 Amazon EKS 基础设施资源，并将其列在容器集群选项卡中。但是，您无法在 Application Manager 中管理或查看有关 Amazon EKS 窗格或容器的运行信息。您只能管理和查看托管 Amazon EKS 资源的基础设施的运行信息。有关定价信息，请参阅 Amazon ECS 定价和 Amazon EKS 定价。

设置查看资源相关运行信息的任务

以下设置任务可帮助您在 Application Manager 中查看有关您的 AWS 资源的信息。

（推荐）验证运行手册权限

您可以通过使用 Systems Manager 自动化运行手册验证来自 Application Manager 的 AWS 资源问题。要使用此修正工具，您必须配置或验证权限。有关定价信息，请参阅 AWS Systems Manager 定价。

（可选）启用 Cost Explorer

AWS Cost Explorer 是 AWS Cost Management 的一项功能，您可以用它来可视化成本数据以供进一步分析。启用 Cost Explorer 后，您可以在 Application Manager 控制台中查看应用程序资源的成本信息、成本历史记录以及成本优化。

（可选）设置和配置 Amazon CloudWatch 日志和警报

CloudWatch 是一项监控和管理服务，为 AWS、混合和多云应用程序以及基础架构资源提供数据和切实可行的见解。借助 CloudWatch，您可以从单个平台以日志和指标的形式收集和访问所有性能和操作数据。要查看您在 Application Manager 中资源的 CloudWatch 日志和警报，就必须设置和配置 CloudWatch。有关定价信息，请参阅 CloudWatch 定价。

> **Note**
>
> CloudWatch Logs 支持仅适用于应用程序，不适用于集群。

（可选）设置和配置 AWS Config

AWS Config 提供了与您的 AWS 账户 关联的资源的详细视图，包括它们是如何配置的、它们之间是如何相互关联的，以及配置及其关系是如何随时间变化的。您可以使用 AWS Config 来评估您的 AWS 资源的配置设置。您可以通过创建 AWS Config 规则执行此操作，这些规则代表您的理想配置设置。在 AWS Config 持续跟踪您的资源中出现的配置更改时，它会检查这些更改是否违反了规则中的任何条件。如果某个资源违反了规则，AWS Config 会将资源和规则标记为不合规。Application Manager 显示关于 AWS Config 规则的合规性信息。要在 Application Manager 中查看此数据，必须设置并配置 AWS Config。有关定价信息，请参阅 AWS Config 定价。

（可选）创建 State Manager 关联

您可以使用 Systems Manager State Manager 创建分配给托管式节点的配置。该配置称为关联，定义要在节点上保持的状态。要在 Application Manager 中查看关联合规性数据，您必须配置一个或多个 State Manager 关联。State Manager 不另外收取费用。

（可选）设置和配置 OpsCenter

您可以通过使用 OpsCenter 在 Application Manager 中查看关于您的资源的运行工作项 (OpsItems)。您可以配置 Amazon CloudWatch 和 Amazon EventBridge 以基于警报和事件将 OpsItems 自动发送到 OpsCenter。您也可以手动输入 OpsItems。有关定价信息，请参阅 AWS Systems Manager 定价。

为 Systems Manager Application Manager 配置权限

如果您的 AWS Identity and Access Management（IAM）实体（如用户、组或角色）有权访问本主题中列出的 API 操作，则您可以使用 Application Manager（AWS Systems Manager 中的一项工具）的所有功能。API 操作分为两个表，以帮助您理解它们执行的不同功能。

下表列出了 Systems Manager 您在 Application Manager 中选择资源时调用的 API 操作，因为您希望查看资源详细信息。例如，如果 Application Manager 列出了 Amazon EC2 Auto Scaling 组，而您选择该组查看其详细信息，Systems Manager 会调用 `autoscaling:DescribeAutoScalingGroups` API 操作。如果您的账户中没有任何 Auto Scaling 组，则不会从 Application Manager 中调用此 API。

仅资源详细信息

```
acm:DescribeCertificate
acm:ListTagsForCertificate
autoscaling:DescribeAutoScalingGroups
cloudfront:GetDistribution
cloudfront:ListTagsForResource
cloudtrail:DescribeTrails
cloudtrail:ListTags
cloudtrail:LookupEvents
codebuild:BatchGetProjects
codepipeline:GetPipeline
codepipeline:ListTagsForResource
dynamodb:DescribeTable
dynamodb:ListTagsOfResource
ec2:DescribeAddresses
ec2:DescribeCustomerGateways
ec2:DescribeHosts
ec2:DescribeInternetGateways
ec2:DescribeNetworkAcls
ec2:DescribeNetworkInterfaces
ec2:DescribeRouteTables
ec2:DescribeSecurityGroups
ec2:DescribeSubnets
ec2:DescribeVolumes
ec2:DescribeVpcs
ec2:DescribeVpnConnections
ec2:DescribeVpnGateways
elasticbeanstalk:DescribeApplications
elasticbeanstalk:ListTagsForResource
elasticloadbalancing:DescribeInstanceHealth
```

仅资源详细信息

```
elasticloadbalancing:DescribeListeners
elasticloadbalancing:DescribeLoadBalancers
elasticloadbalancing:DescribeTags
iam:GetGroup
iam:GetPolicy
iam:GetRole
iam:GetUser
lambda:GetFunction
rds:DescribeDBClusters
rds:DescribeDBInstances
rds:DescribeDBSecurityGroups
rds:DescribeDBSnapshots
rds:DescribeDBSubnetGroups
rds:DescribeEventSubscriptions
rds:ListTagsForResource
redshift:DescribeClusterParameters
redshift:DescribeClusterSecurityGroups
redshift:DescribeClusterSnapshots
redshift:DescribeClusterSubnetGroups
redshift:DescribeClusters
s3:GetBucketTagging
```

下表列出了 Systems Manager 用于更改 Application Manager 中所列应用程序和资源或查看所选应用程序或资源的操作信息的 API 操作。

申请操作和详细信息

```
applicationinsights:CreateApplication
applicationinsights:DescribeApplication
applicationinsights:ListProblems
ce:GetCostAndUsage
ce:GetTags
ce:ListCostAllocationTags
ce:UpdateCostAllocationTagsStatus
cloudformation:CreateStack
cloudformation:DeleteStack
cloudformation:DescribeStackDriftDetectionStatus
cloudformation:DescribeStackEvents
```

申请操作和详细信息

```
cloudformation:DescribeStacks
cloudformation:DetectStackDrift
cloudformation:GetTemplate
cloudformation:GetTemplateSummary
cloudformation:ListStacks
cloudformation:UpdateStack
cloudwatch:DescribeAlarms
cloudwatch:DescribeInsightRules
cloudwatch:DisableAlarmActions
cloudwatch:EnableAlarmActions
cloudwatch:GetMetricData
cloudwatch:ListTagsForResource
cloudwatch:PutMetricAlarm
config:DescribeComplianceByConfigRule
config:DescribeComplianceByResource
config:DescribeConfigRules
config:DescribeRemediationConfigurations
config:GetComplianceDetailsByConfigRule
config:GetComplianceDetailsByResource
config:GetResourceConfigHistory
config:ListDiscoveredResources
config:PutRemediationConfigurations
config:SelectResourceConfig
config:StartConfigRulesEvaluation
config:StartRemediationExecution
ec2:DescribeInstances
ecs:DescribeCapacityProviders
ecs:DescribeClusters
ecs:DescribeContainerInstances
ecs:ListClusters
ecs:ListContainerInstances
ecs:TagResource
eks:DescribeCluster
eks:DescribeFargateProfile
eks:DescribeNodegroup
eks:ListClusters
eks:ListFargateProfiles
eks:ListNodegroups
eks:TagResource
iam:CreateServiceLinkedRole
iam:ListRoles
logs:DescribeLogGroups
```

申请操作和详细信息

```
resource-groups:CreateGroup
resource-groups:DeleteGroup
resource-groups:GetGroup
resource-groups:GetGroupQuery
resource-groups:GetTags
resource-groups:ListGroupResources
resource-groups:ListGroups
resource-groups:Tag
resource-groups:Untag
resource-groups:UpdateGroup
s3:ListAllMyBuckets
s3:ListBucket
s3:ListBucketVersions
servicecatalog:GetApplication
servicecatalog:ListApplications
sns:CreateTopic
sns:ListSubscriptionsByTopic
sns:ListTopics
sns:Subscribe
ssm:AddTagsToResource
ssm:CreateDocument
ssm:CreateOpsMetadata
ssm:DeleteDocument
ssm:DeleteOpsMetadata
ssm:DescribeAssociation
ssm:DescribeAutomationExecutions
ssm:DescribeDocument
ssm:DescribeDocumentPermission
ssm:GetDocument
ssm:GetInventory
ssm:GetOpsMetadata
ssm:GetOpsSummary
ssm:GetServiceSetting
ssm:ListAssociations
ssm:ListComplianceItems
ssm:ListDocuments
ssm:ListDocumentVersions
ssm:ListOpsMetadata
ssm:ListResourceComplianceSummaries
ssm:ListTagsForResource
ssm:ModifyDocumentPermission
ssm:RemoveTagsFromResource
```

申请操作和详细信息

```
ssm:StartAssociationsOnce
ssm:StartAutomationExecution
ssm:UpdateDocument
ssm:UpdateDocumentDefaultVersion
ssm:UpdateOpsItem
ssm:UpdateOpsMetadata
ssm:UpdateServiceSetting
tag:GetTagKeys
tag:GetTagValues
tag:TagResources
tag:UntagResources
```

所有 Application Manager 权限的策略示例

要为 IAM 实体（如用户、组或角色）配置 Application Manager 权限，请使用以下示例创建 IAM policy。此策略示例包括 Application Manager 使用的所有 API 操作。

JSON

```
                {
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "acm:DescribeCertificate",
                "acm:ListTagsForCertificate",
                "applicationinsights:CreateApplication",
                "applicationinsights:DescribeApplication",
                "applicationinsights:ListProblems",
                "autoscaling:DescribeAutoScalingGroups",
                "ce:GetCostAndUsage",
                "ce:GetTags",
                "ce:ListCostAllocationTags",
                "ce:UpdateCostAllocationTagsStatus",
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack",
                "cloudformation:DescribeStackDriftDetectionStatus",
```

```
                "cloudformation:DescribeStackEvents",
                "cloudformation:DescribeStacks",
                "cloudformation:DetectStackDrift",
                "cloudformation:GetTemplate",
                "cloudformation:GetTemplateSummary",
                "cloudformation:ListStacks",
                "cloudformation:ListStackResources",
                "cloudformation:UpdateStack",
                "cloudfront:GetDistribution",
                "cloudfront:ListTagsForResource",
                "cloudtrail:DescribeTrails",
                "cloudtrail:ListTags",
                "cloudtrail:LookupEvents",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:DescribeInsightRules",
                "cloudwatch:DisableAlarmActions",
                "cloudwatch:EnableAlarmActions",
                "cloudwatch:GetMetricData",
                "cloudwatch:ListTagsForResource",
                "cloudwatch:PutMetricAlarm",
                "codebuild:BatchGetProjects",
                "codepipeline:GetPipeline",
                "codepipeline:ListTagsForResource",
                "config:DescribeComplianceByConfigRule",
                "config:DescribeComplianceByResource",
                "config:DescribeConfigRules",
                "config:DescribeRemediationConfigurations",
                "config:GetComplianceDetailsByConfigRule",
                "config:GetComplianceDetailsByResource",
                "config:GetResourceConfigHistory",
                "config:ListDiscoveredResources",
                "config:PutRemediationConfigurations",
                "config:SelectResourceConfig",
                "config:StartConfigRulesEvaluation",
                "config:StartRemediationExecution",
                "dynamodb:DescribeTable",
                "dynamodb:ListTagsOfResource",
                "ec2:DescribeAddresses",
                "ec2:DescribeCustomerGateways",
                "ec2:DescribeHosts",
                "ec2:DescribeInstances",
                "ec2:DescribeInternetGateways",
                "ec2:DescribeNetworkAcls",
                "ec2:DescribeNetworkInterfaces",
```

```
                "ec2:DescribeRouteTables",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSubnets",
                "ec2:DescribeVolumes",
                "ec2:DescribeVpcs",
                "ec2:DescribeVpnConnections",
                "ec2:DescribeVpnGateways",
                "ecs:DescribeCapacityProviders",
                "ecs:DescribeClusters",
                "ecs:DescribeContainerInstances",
                "ecs:ListClusters",
                "ecs:ListContainerInstances",
                "ecs:TagResource",
                "eks:DescribeCluster",
                "eks:DescribeFargateProfile",
                "eks:DescribeNodegroup",
                "eks:ListClusters",
                "eks:ListFargateProfiles",
                "eks:ListNodegroups",
                "eks:TagResource",
                "elasticbeanstalk:DescribeApplications",
                "elasticbeanstalk:ListTagsForResource",
                "elasticloadbalancing:DescribeInstanceHealth",
                "elasticloadbalancing:DescribeListeners",
                "elasticloadbalancing:DescribeLoadBalancers",
                "elasticloadbalancing:DescribeTags",
                "iam:CreateServiceLinkedRole",
                "iam:GetGroup",
                "iam:GetPolicy",
                "iam:GetRole",
                "iam:GetUser",
                "iam:ListRoles",
                "lambda:GetFunction",
                "logs:DescribeLogGroups",
                "rds:DescribeDBClusters",
                "rds:DescribeDBInstances",
                "rds:DescribeDBSecurityGroups",
                "rds:DescribeDBSnapshots",
                "rds:DescribeDBSubnetGroups",
                "rds:DescribeEventSubscriptions",
                "rds:ListTagsForResource",
                "redshift:DescribeClusterParameters",
                "redshift:DescribeClusters",
                "redshift:DescribeClusterSecurityGroups",
```

```
                "redshift:DescribeClusterSnapshots",
                "redshift:DescribeClusterSubnetGroups",
                "resource-groups:CreateGroup",
                "resource-groups:DeleteGroup",
                "resource-groups:GetGroup",
                "resource-groups:GetGroupQuery",
                "resource-groups:GetTags",
                "resource-groups:ListGroupResources",
                "resource-groups:ListGroups",
                "resource-groups:Tag",
                "resource-groups:Untag",
                "resource-groups:UpdateGroup",
                "s3:GetBucketTagging",
                "s3:ListAllMyBuckets",
                "s3:ListBucket",
                "s3:ListBucketVersions",
                "servicecatalog:GetApplication",
                "servicecatalog:ListApplications",
                "sns:CreateTopic",
                "sns:ListSubscriptionsByTopic",
                "sns:ListTopics",
                "sns:Subscribe",
                "ssm:AddTagsToResource",
                "ssm:CreateDocument",
                "ssm:CreateOpsMetadata",
                "ssm:DeleteDocument",
                "ssm:DeleteOpsMetadata",
                "ssm:DescribeAssociation",
                "ssm:DescribeAutomationExecutions",
                "ssm:DescribeDocument",
                "ssm:DescribeDocumentPermission",
                "ssm:GetDocument",
                "ssm:GetInventory",
                "ssm:GetOpsMetadata",
                "ssm:GetOpsSummary",
                "ssm:GetServiceSetting",
                "ssm:ListAssociations",
                "ssm:ListComplianceItems",
                "ssm:ListDocuments",
                "ssm:ListDocumentVersions",
                "ssm:ListOpsMetadata",
                "ssm:ListResourceComplianceSummaries",
                "ssm:ListTagsForResource",
                "ssm:ModifyDocumentPermission",
```

```
                "ssm:RemoveTagsFromResource",
                "ssm:StartAssociationsOnce",
                "ssm:StartAutomationExecution",
                "ssm:UpdateDocument",
                "ssm:UpdateDocumentDefaultVersion",
                "ssm:UpdateOpsMetadata",
                "ssm:UpdateOpsItem",
                "ssm:UpdateServiceSetting",
                "tag:GetResources",
                "tag:GetTagKeys",
                "tag:GetTagValues",
                "tag:TagResources",
                "tag:UntagResources"
            ],
            "Resource": "*"
        }
    ]
}
```

Note

您可以限制用户在 Application Manager 中对应用程序和资源的进行修改的能力，方法是从附加到其用户、组或角色的 IAM 权限策略中删除以下 API 操作。删除这些操作会在 Application Manager 中创建一个只读体验。以下是允许用户更改应用程序或其他相关资源的所有 API。

```
applicationinsights:CreateApplication
ce:UpdateCostAllocationTagsStatus
cloudformation:CreateStack
cloudformation:DeleteStack
cloudformation:UpdateStack
cloudwatch:DisableAlarmActions
cloudwatch:EnableAlarmActions
cloudwatch:PutMetricAlarm
config:PutRemediationConfigurations
config:StartConfigRulesEvaluation
config:StartRemediationExecution
ecs:TagResource
eks:TagResource
iam:CreateServiceLinkedRole
resource-groups:CreateGroup
resource-groups:DeleteGroup
```

```
resource-groups:Tag
resource-groups:Untag
resource-groups:UpdateGroup
sns:CreateTopic
sns:Subscribe
ssm:AddTagsToResource
ssm:CreateDocument
ssm:CreateOpsMetadata
ssm:DeleteDocument
ssm:DeleteOpsMetadata
ssm:ModifyDocumentPermission
ssm:RemoveTagsFromResource
ssm:StartAssociationsOnce
ssm:StartAutomationExecution
ssm:UpdateDocument
ssm:UpdateDocumentDefaultVersion
ssm:UpdateOpsMetadata
ssm:UpdateOpsItem
ssm:UpdateServiceSetting
tag:TagResources
tag:UntagResources
```

有关如何创建 IAM policy 的信息，请参阅 IAM 用户指南中的创建 IAM policy。有关如何将此策略分配给 IAM 实体（如用户、组或角色）的信息，请参阅添加和删除 IAM 身份权限。

将应用程序和容器集群添加到 Application Manager

Application Manager 是 AWS Systems Manager 的一个组件。在 Application Manager 中，应用程序是您希望作为一个单元运行的 AWS 资源的逻辑组。此逻辑组可以表示应用程序的不同版本、操作员的所有权边界或开发人员环境等。

首次打开 Application Manager 时，会显示Application Manager 能为您做什么页面。选择开始使用时，Application Manager 会自动导入有关在其他 AWS 服务或 Systems Manager 工具中创建的有关资源的元数据。然后，Application Manager 在按预定义类别分组的列表中显示这些资源。

对于应用程序，列表包含以下内容：

- AWS CloudFormation 堆栈
- 自定义应用程序
- AWS Launch Wizard 应用程序

- AppRegistry 应用程序
- AWS SAP 企业工作负载应用程序
- Amazon ECS 集群
- Amazon EKS 集群

导入完成后，您可以在这些预定义类别中查看应用程序或特定资源的操作信息。或者，如果要提供有关资源集合的更多上下文，则可以在 Application Manager 中手动创建一个应用程序。然后，您可以将资源或资源组添加到该应用程序中。在 Application Manager 中创建应用程序后，您可以在应用程序的上下文中查看有关资源的操作信息。

在 Application Manager 中创建应用程序

使用以下过程在 Application Manager 创建应用程序并向该应用程序添加资源。

在 Application Manager 中创建应用程序

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Application Manager。
3. 选择创建应用程序。
4. 从下拉列表中选择一个选项，然后填写显示的页面中的字段。

使用 应用程序

Application Manager 是 AWS Systems Manager 的一个组件。本部分包含帮助您使用 Application Manager 应用程序并查看有关 AWS 资源的运行信息的主题。

内容

- Application Manager 中的应用程序概述
- 管理应用程序 EC2 实例
- 查看与应用程序关联的资源
- 管理您的应用程序合规性
- 使用 CloudWatch Application Insights 监控应用程序
- 查看应用程序的 OpsItems
- 管理您的应用程序日志

- 使用自动化运行手册修复应用程序问题
- 为 Application Manager 中的资源添加标签
- 在 Application Manager 中使用 AWS CloudFormation 模板和堆栈
- 使用 Application Manager 中的数据库集群

Application Manager 中的应用程序概述

Application Manager 是 AWS Systems Manager 的一个组件，其中的概览选项卡显示 Amazon CloudWatch 警报、操作工作项目 (OpsItems)、CloudWatch Application Insights 和运行手册历史记录。选择查看全部以打开任意卡片，您可以在其中查看所有应用见解、警报、OpsItems 或运行手册历史记录。

关于 Application Insights

CloudWatch Application Insights 可在应用程序资源和技术堆栈中指定并设置关键指标、日志和告警。Application Insights 会持续监控指标和日志，以检测异常情况和错误并将它们关联起来。在系统检测到错误和异常情况时，Application Insights 生成 CloudWatch Events，您可以使用这些事件来设置通知或执行操作。如果选择监控选项卡上的编辑配置按钮，系统将打开 CloudWatch Application Insights 控制台。有关 Application Insights 的更多信息，请参阅 Amazon CloudWatch 用户指南中的什么是 Amazon CloudWatch Application Insights。

关于成本管理器

Application Manager 通过成本小组件和成本选项卡，与 AWS 成本管理的 AWS Cost Explorer 功能集成。在成本管理控制台中启用 Cost Explorer 后，Application Manager 中的成本小组件和成本选项卡会显示特定非容器应用程序或应用程序组件的成本数据。您可以根据条形图或折线图中的不同时段、精细度和成本类型，使用小组件或选项卡中的筛选条件查看成本数据。

您可以通过选择以下选项来启用此功能转到AWS成本管理控制台（按钮）。默认情况下，会筛选过去三个月内的数据。对于非容器应用程序，如果您选择 View all（查看全部）按钮，Application Manager 会打开 Resources（资源）选项卡。对于容器应用程序，View all（查看全部）按钮会打开 AWS Cost Explorer 控制台。

您可以在此页上执行的操作

您可以在此页面的 Overview（概览）选项卡上打开并访问有关以下控件的信息。启用控件后，选择其中的 View all（查看全部）以查看该区域相关应用程序的详细信息。

- 在 Insights and Alarms（见解和警报）部分，选择严重性数字以打开 Monitoring（监控）选项卡，您可以在其中查看有关所选严重性警报的更多详细信息。

- 在 Cost（成本）部分，选择 View all（查看全部）以打开 Resources（资源）选项卡，您可以在其中查看特定应用程序或应用程序组件的成本数据。
- 在 Compliance（合规性）部分中，选择 View all（查看全部）以打开 Compliance（合规性）选项卡，您可以在其中查看来自 AWS Config 和 State Manager 关联的合规信息。

> **Note**
>
> 要查看补丁合规性详细信息，请直接选择 Compliance（合规性）选项卡。然后，您可以查看所选应用程序使用的托管式节点的补丁合规性详细信息。

- 在运行手册部分中，选择一个运行手册以在 Systems Manager 文档页面中打开，您可以在其中查看有关文档的更多详细信息。
- 在 OpsItems 部分中，选择一个严重性以打开 OpsItems 选项卡，您可以在其中查看所有选定严重性的 OpsItems。
- 选择查看全部按钮打开相应的选项卡。您可以查看应用程序的所有警报、OpsItems 或运行手册历史记录条目。

要打开 Overview（概览）选项卡

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Application Manager。
3. 在应用程序部分中，选择一个类别。如果要在 Application Manager 中打开您手动创建的应用程序，则选择定制应用程序。
4. 在列表中选择应用程序。Application Manager 打开 概览选项卡。

管理应用程序 EC2 实例

Application Manager 与 Amazon Elastic Compute Cloud (Amazon EC2) 集成，以在应用程序上下文中显示有关实例的信息。Application Manager 以图形格式显示所选应用程序的实例状态、情况和 Amazon EC2 Auto Scaling 运行状况。Instances（实例）选项卡还包括一个表，其中包含应用程序中每个实例的以下信息：

- 实例状态（待处理、正在停止、正在运行、已停止）
- SSM Agent 的 Ping 状态
- 在实例上处理的最新 Systems Manager Automation 运行手册的状态和名称

- 每个州的 Amazon CloudWatch Logs 警报数量。
  - ALARM – 指标或表达式超出定义的阈值。
  - OK – 指标或表达式在定义的阈值范围内。
  - INSUFFICIENT_DATA（数据不足）– 告警刚刚启动，指标不可用，或者指标没有足够的数据以确定告警状态。
- 父组和单个自动扩缩组的自动扩缩组运行状况

如果您在 All instances（所有实例）表中选择一个实例，则 Application Manager 会在四个选项卡上显示有关该实例的信息：

- Details（详细信息）：来自 Amazon EC2 的所有实例详细信息，包括亚马逊机器映像（AMI）、DNS 信息、IP 地址信息等。
- Health（运行状况）：EC2 系统和实例状态检查提供的当前状态。
- Execution history（执行历史记录）：实例处理的 Systems Manager Automation 运行手册和 API 调用的执行日志。
- CloudWatch alarms（CloudWatch 警报）：实例引发的任何 CloudWatch 警报的名称、状态等。

您可以在此页上执行的操作

您可以在此页面上执行以下操作：

- 启动、停止和终止实例。
- 应用 Chef 配方。
- 将实例附加到自动扩缩组或从中分离实例。
- 为 SSM Agent 启用自动更新。

要打开 Instances（实例）选项卡

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Application Manager。
3. 在应用程序部分中，选择一个类别。如果要在 Application Manager 中打开您手动创建的应用程序，则选择 Custom applications（自定义应用程序）。
4. 在列表中选择应用程序。Application Manager 打开 概览选项卡。

5. 选择 Instances 选项卡。

查看应用程序实例详细信息

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Application Manager。
3. 在应用程序部分中，选择一个类别。如果要在 Application Manager 中打开您手动创建的应用程序，则选择 Custom applications（自定义应用程序）。
4. 在列表中选择应用程序。Application Manager 打开 概览选项卡。
5. 选择 Instances 选项卡。
6. 选择要查看其详细信息的实例旁边的按钮。
7. 检查页面底部的实例详细信息。

自动更新 SSM Agent

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Application Manager。
3. 在应用程序部分中，选择一个类别。如果要在 Application Manager 中打开您手动创建的应用程序，则选择 Custom applications（自定义应用程序）。
4. 在列表中选择应用程序。Application Manager 打开 概览选项卡。
5. 选择 Instances 选项卡。
6. 在代理操作下拉列表中，选择配置 SSM Agent 更新。
7. 选择所有实例，从而为所有托管式实例配置 SSM Agent 自动更新。也可选择实例，从而为应用程序中的单个实例配置 SSM Agent 自动更新。
8. 选择启用自动更新开关。
9. 在指定时间表下拉列表中，选择要用于 SSM Agent 更新的时间表。
10. 选择配置。

查看与应用程序关联的资源

Application Manager 是 AWS Systems Manager 的一个组件，其中资源选项卡显示 AWS 资源。如果选择顶层组件，则此页将显示该组件和任何子组件的所有资源。如果选择子组件，则此页仅显示分配给该子组件的资源。

您可以在此页上执行的操作

您可以在此页面上执行以下操作：

- 选择资源名称以查看相关信息，包括创建资源的控制台提供的详细信息、标签、Amazon CloudWatch 警报、AWS Config 详细信息以及 AWS CloudTrail 日志信息。
- 选择资源名称旁边的选项按钮。然后，选择资源时间线按钮打开 AWS Config 控制台，您可以在其中查看有关所选资源的合规性信息。
- 如果您启用 AWS Cost Explorer，Cost Explorer 部分会显示特定非容器应用程序或应用程序组件的成本数据。您可以通过选择以下选项来启用此功能转到AWS成本管理控制台（按钮）。使用此部分中的筛选条件查看有关应用程序的成本信息。

要打开 Resources（资源）选项卡

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Application Manager。
3. 在应用程序部分中，选择一个类别。如果要在 Application Manager 中打开您手动创建的应用程序，则选择定制应用程序。
4. 在列表中选择应用程序。Application Manager 打开概述选项卡。
5. 选择资源选项卡。

管理您的应用程序合规性

在 Application Manager（AWS Systems Manager 的一个组件）中，Configurations（配置）页面会显示 AWS Config 资源和配置规则合规性信息。此页面还显示 AWS Systems Manager State Manager 关联合规性信息。您可以选择资源、规则或关联来打开相应的控制台以获取详细信息。此页面显示最近 90 天的合规性信息。

您可以在此页上执行的操作

您可以在此页面上执行以下操作：

- 选择一个资源名称以打开 AWS Config 控制台，您可以在其中查看有关所选资源的合规性信息。
- 选择资源名称旁边的选项按钮。然后，选择资源时间线按钮打开 AWS Config 控制台，您可以在其中查看有关所选资源的合规性信息。
- 在配置规则合规性部分，您可以执行以下操作：
  - 选择规则名称以打开 AWS Config 控制台，您可以在其中查看有关该规则的信息。
  - 选择添加规则以打开 AWS Config 控制台，您可以在其中创建规则。
  - 选择规则名称旁边的选项按钮，选择操作，然后选择。管理修正以更改规则的修正操作。
  - 选择规则名称旁边的选项按钮，选择操作，然后选择重新评估，以便让 AWS Config 按照所选规则运行合规性检查。
- 在关联合规性部分，您可以执行以下操作：
  - 选择关联名称以打开关联页面，您可以在其中查看有关该关联的信息。
  - 选择创建关联以打开 Systems Manager State Manager，在这里您可以创建关联。
  - 选择关联名称旁边的选项按钮，然后选择应用关联以立即启动关联中指定的所有操作。

打开 Compliance（合规性）选项卡

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Application Manager。
3. 在应用程序部分中，选择一个类别。如果要在 Application Manager 中打开您手动创建的应用程序，则选择定制应用程序。
4. 在列表中选择应用程序。Application Manager 打开 概览选项卡。
5. 选择 Compliance（合规性）选项卡。

使用 CloudWatch Application Insights 监控应用程序

Application Manager 是 AWS Systems Manager 的一个组件，其中的监控选项卡显示应用程序中资源的 Amazon CloudWatch Application Insights 和警报详细信息。

关于 Application Insights

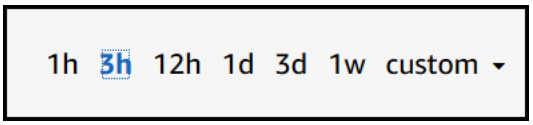
CloudWatch Application Insights 可在应用程序资源和技术堆栈中指定并设置关键指标、日志和告警。Application Insights 会持续监控指标和日志，以检测异常情况和错误并将它们关联起来。在系统检

测到错误和异常情况时，Application Insights 生成 CloudWatch Events，您可以使用这些事件来设置通知或执行操作。如果选择监控选项卡上的编辑配置按钮，系统将打开 CloudWatch Application Insights 控制台。有关 Application Insights 的更多信息，请参阅 Amazon CloudWatch 用户指南中的什么是 Amazon CloudWatch Application Insights。

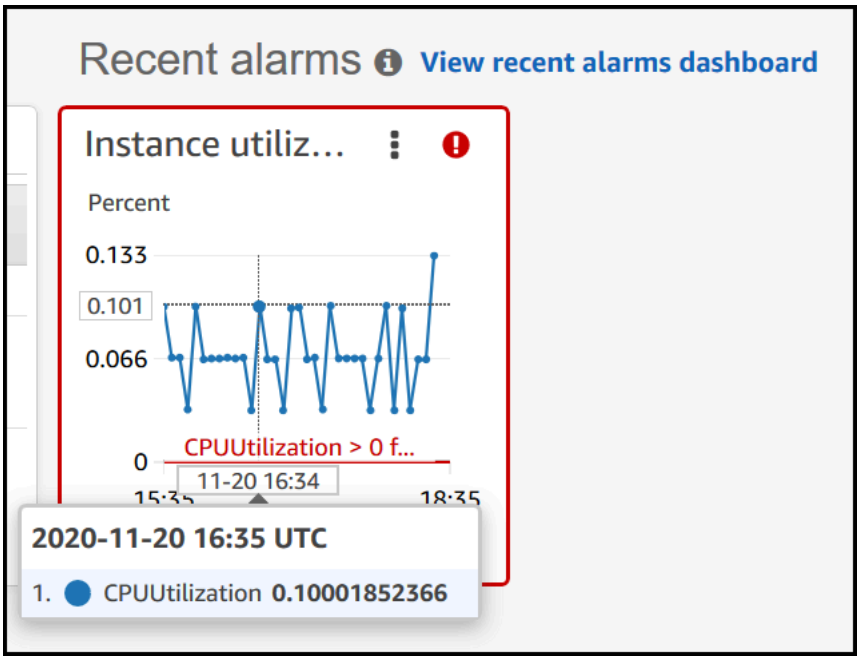
您可以在此页上执行的操作

您可以在此页面上执行以下操作：

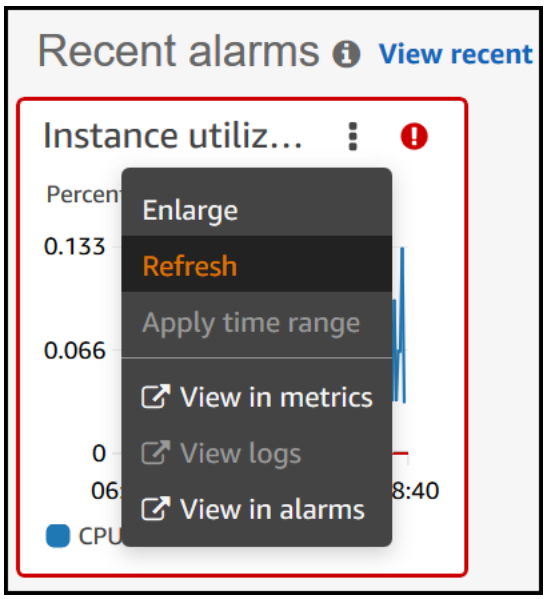
- 在按 AWS 服务分类的警报部分选择服务名称，打开 CloudWatch 到所选服务和警报。
- 通过选择一个预定义的时间段值，调整最近警报部分小组件中所显示数据的时间段。您可以选择自定义来定义您自己的时间段。



- 将光标悬停在最近警报部分的小组件上方，查看特定时间的数据弹出窗口。

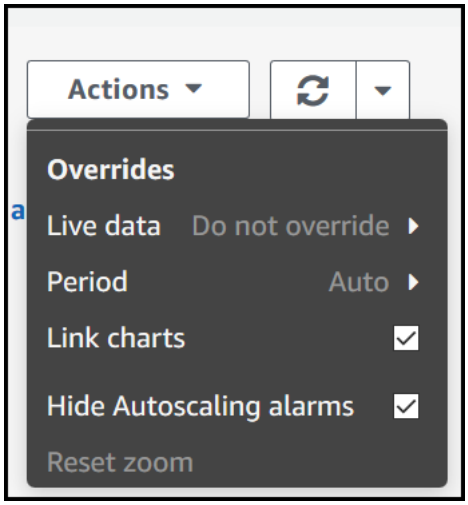


- 选择小组件中的选项菜单以查看显示选项。选择放大以展开小部件。选择刷新更新小部件中的数据。在小组件数据显示中单击并拖动光标以选择特定范围。然后，您可以选择应用时间范围。



- 选择操作菜单查看警报数据覆盖选项，其中包括以下内容：
  - 选择小组件是否显示实时数据。实时数据是将在最后一分钟内发布的、尚未完全聚合的数据。如果关闭实时数据，则仅显示过去至少一分钟的聚合期的数据点。例如，如果使用 5 分钟聚合期，12:35 的数据点将从 12:35 聚合到 12:40，并在 12:41 时显示。

    如果启用实时数据，则在相应聚合间隔内发布任何数据后，立即显示最新数据点。每次刷新显示时，最新数据点都可能会随着该聚合期内的新数据发布而发生变化。
  - 指定实时数据的时间段。
  - 将图表链接到最近警报部分，这样当您放大或缩小一个图表时，另一个图表将同时放大或缩小。您可以取消图表链接以仅允许缩放一个图表。
  - 隐藏 Auto Scaling 警报。



要打开 Monitoring（监控）选项卡

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Application Manager。
3. 在应用程序部分中，选择一个类别。如果要在 Application Manager 中打开您手动创建的应用程序，则选择定制应用程序。
4. 在列表中选择应用程序。Application Manager 打开概览选项卡。
5. 选择监控选项卡。

查看应用程序的 OpsItems

Application Manager 是 AWS Systems Manager d 一个组件，其中 OpsItems 选项卡显示所选应用程序中资源的操作工作项 (OpsItems）。您可以配置 Systems Manager OpsCenter 自动创建来自 Amazon CloudWatch 警报和 Amazon EventBridge 事件的 OpsItems。您还可以手动创建 OpsItems。

您可以在此选项卡上执行的操作

您可以在此页面上执行以下操作：

- 通过使用搜索字段筛选 OpsItems 的列表。您可以通过名称、ID、源 ID 或严重性筛选 OpsItem。您也可以根据状态筛选此列表。OpsItems 支持以下状态：“打开”、“正在进行中”、“打开”并“正在进行中”、“已解决”或“全部”。
- 更改 OpsItem 状态的方法是选择旁边的选项按钮，然后在设置状态菜单选择一个选项。
- 打开 Systems Manager OpsCenter，通过选择创建 OpsItem 创建 OpsItem。

要打开 OpsItems 选项卡

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Application Manager。
3. 在应用程序部分中，选择一个类别。如果要在 Application Manager 中打开您手动创建的应用程序，则选择定制应用程序。
4. 在列表中选择应用程序。Application Manager 打开概览选项卡。
5. 选择 OpsItems 选项卡。

管理您的应用程序日志

Application Manager 是 AWS Systems Manager 的一个组件，其中日志选项卡显示来自 Amazon CloudWatch Logs 的日志组列表。

您可以在此选项卡上执行的操作

您可以在此页面上执行以下操作：

- 选择一个日志组名称以在 CloudWatch Logs 中打开。然后，您可以选择日志流以查看应用程序上下文中资源的日志。
- 选择创建日志组以在 CloudWatch Logs 中创建日志组。

要打开 Logs（日志）选项卡

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Application Manager。
3. 在应用程序部分中，选择一个类别。如果要在 Application Manager 中打开您手动创建的应用程序，则选择定制应用程序。
4. 在列表中选择应用程序。Application Manager 打开概览选项卡。
5. 选择日志选项卡。

使用自动化运行手册修复应用程序问题

您可以通过使用 Automation 运行手册，从 Application Manager（AWS Systems Manager 中的一项工具）修复 AWS 资源的问题。自动化运行手册定义了自动化运行时 Systems Manager 在托管式实例和其他 AWS 资源上执行的操作。Automation 是 AWS Systems Manager 中的一项工具。运行手册包含一个或多个按顺序运行的步骤。每个步骤是根据单个操作生成的。可以将一个步骤的输出作为后面步骤的输入。

在从 Application Manager 应用程序或集群选择 Start runbook（启动运行手册）时，系统会根据应用程序或集群中的资源类型显示经过筛选的可用运行手册列表。当您选择要启动的运行手册时，Systems Manager 会打开执行自动化文档页。

Application Manager 包含以下针对使用运行手册的增强功能。

- 如果您在 Application Manager 中选择了某个资源的名称，然后选择执行运行手册，系统会显示经过筛选的该资源类型的运行手册列表。

- 您可以在列表中选择一个运行手册，然后选择为相同类型的资源运行，启动所有相同类型资源的自动化。

开始前的准备工作

在您开始从 Application Manager 中启动运行手册之前，执行以下操作：

- 验证您具有启动运行手册的权限。有关更多信息，请参阅 设置自动化。
- 查看有关启动运行手册的自动化过程文档。有关更多信息，请参阅 运行由 Systems Manager Automation 支持的自动化操作。

从 Application Manager 启动运行手册

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Application Manager。
3. 在应用程序部分中，选择一个类别。如果要在 Application Manager 中打开您手动创建的应用程序，则选择定制应用程序。
4. 在列表中选择应用程序。Application Manager 打开 概览选项卡。
5. 选择启动运行手册。Application Manager 会打开自动化小组件弹出窗口。有关自动化小组件中的选项的信息，请参阅 运行由 Systems Manager Automation 支持的自动化操作。

为 Application Manager 中的资源添加标签

在 Application Manager 中，您可以快速添加或删除应用程序和 AWS 资源上的标签。请使用以下过程将标签添加到应用程序或从应用程序中删除，并对该应用程序中的所有 AWS 资源实施相同的操作。

向应用程序和应用程序中的所有资源添加标签或从中删除标签

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Application Manager。
3. 在应用程序部分中，选择一个类别。如果要在 Application Manager 中打开您手动创建的应用程序，则选择定制应用程序。
4. 在列表中选择应用程序。Application Manager 打开概览选项卡。

5. 在应用信息部分，选择应用程序标记下方的数字。如果没有为应用程序分配任何标签，则数字为零。
6. 要添加标签，请选择添加新标签。指定键和可选值。要删除标签，请选择删除。
7. 选择保存。

请按照以下步骤将标记添加到 Application Manager 中的特定资源，或将标签从中删除。

向资源添加标签或从中删除标签

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Application Manager。
3. 在应用程序部分中，选择一个类别。如果要在 Application Manager 中打开您手动创建的应用程序，则选择定制应用程序。
4. 在列表中选择应用程序。Application Manager 打开概述选项卡。
5. 选择资源选项卡。
6. 选择一个资源名称。
7. 在标签部分中，选择编辑。
8. 要添加标签，请选择添加新标签。指定键和可选值。要删除标签，请选择删除。
9. 选择保存。

在 Application Manager 中使用 AWS CloudFormation 模板和堆栈

Application Manager 是 AWS Systems Manager 中的一项工具，通过与 AWS CloudFormation 集成，可帮助您配置和管理应用程序的资源。您可以在 Application Manager 中创建、编辑和删除 AWS CloudFormation 模板和堆栈。堆栈是可作为单个单元管理的一系列 AWS 资源。这意味着您可以通过使用 CloudFormation 堆栈，创建、更新或删除 AWS 资源。模板是 JSON 或 YAML 中的格式化文本文件，指定要在堆栈中配置的资源。

Application Manager 还包括一个模板库，您可以在其中克隆、创建和存储模板。Application Manager 和 CloudFormation 显示有关堆栈的当前状态的相同信息。模板和模板更新存储在 Systems Manager 中，直到您配置堆栈为止，届时更改也会显示在 CloudFormation 中。

在 Application Manager 中创建堆栈后，CloudFormation 堆栈页面显示有关此堆栈的有用信息。这包括用于创建它的模板、针对堆栈中资源的一系列 OpsItems、堆栈状态和偏差状态。

关于成本管理器

Application Manager 通过 Cost 小组件与AWS 成本管理功能 AWS Cost Explorer 集成。在成本管理控制台中启用 Cost Explorer 后，Application Manager 中的 Cost 小组件会显示特定非容器应用程序或应用程序组件的成本数据。您可以根据条形图或折线图中的不同时段、精细度和成本类型，使用小组件中的筛选条件查看成本数据。

您可以通过选择以下选项来启用此功能转到AWS成本管理控制台（按钮）。默认情况下，会筛选过去三个月内的数据。对于非容器应用程序，如果您选择 View all（查看全部）按钮，Application Manager 会打开 Resources（资源）选项卡。对于容器应用程序，View all（查看全部）按钮会打开 AWS Cost Explorer 控制台。

> **Note**
>
> Cost Explorer 使用标签来跟踪应用程序成本。如果 AWS CloudFormation 基于堆栈的应用程序未配置 `AppManagerCFNStackKey` 标签键，则 Cost Explorer 无法在 Application Manager 中准确显示成本数据。当未检测到 `AppManagerCFNStackKey` 标签键时，控制台中将提示您将标签添加到您的 CloudFormation 堆栈中以启用成本跟踪功能。添加时会将标签键映射到堆栈的 Amazon 资源名称 (ARN)，使 Cost 小组件能够准确显示成本数据。

> **Important**
>
> 添加 `AppManagerCFNStackKey` 标签的操作将触发堆栈更新。在最初部署堆栈后执行的任何手动配置都不会在添加用户标签后反映出来。有关资源更新行为的更多信息，请参阅《AWS CloudFormation User Guide》中的 Update behaviors of stack resources。

开始前的准备工作

在创建、编辑或删除 CloudFormation 模板和堆栈之前，请使用以下链接了解有关 CloudFormation 概念的信息，方法是使用 Application Manager。

- 什么是 AWS CloudFormation ？
- AWS CloudFormation 最佳实践
- 了解模板基础知识
- 使用 AWS CloudFormation 堆栈
- 使用 AWS CloudFormation 模板

- 示例模板

主题

- 使用 Application Manager 管理 AWS CloudFormation 模板
- 使用 Application Manager 管理 AWS CloudFormation 堆栈

使用 Application Manager 管理 AWS CloudFormation 模板

Application Manager 是 AWS Systems Manager 中的一项工具，包括一个模板库和其他工具，可帮助您管理 AWS CloudFormation 模板。本节包含以下信息：

主题

- 使用模板库
- 创建模板
- 编辑模板

使用模板库

这些 Application Manager 模板库提供了帮助您查看、创建、编辑、删除和克隆模板的工具。您也可以直接从模板库配置堆栈。模板存储为 `CloudFormation` 类型的 Systems Manager (SSM) 文档。通过将模板存储为 SSM 文档，您可以使用版本控件来操作不同版本的模板。您也可以设置权限和共享模板。成功配置堆栈后，将在 Application Manager 和 CloudFormation 中提供堆栈和模板。

开始前的准备工作

我们建议您阅读以下主题，以了解有关 SSM 文档的详细信息，然后再开始使用 Application Manager 中的 CloudFormation 模板。

- AWS Systems Manager 文档
- 共享 SSM 文档
- 共享 SSM 文档的最佳做法

查看 Application Manager 中的模板库

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。

2. 在导航窗格中，请选择 Application Manager。
3. 选择 CloudFormation 模板库。

创建模板

以下过程介绍如何在 Application Manager 中创建 CloudFormation 模板。创建模板时，可以在 JSON 或 YAML 中输入模板的堆栈详细信息。如果您不熟悉 JSON 或 YAML，可以使用 AWS 基础架构编辑器，此工具可用于通过可视化方式创建和修改模板。有关更多信息，请参阅《AWS CloudFormation 用户指南》中的使用基础设施编辑器直观地创建模板。有关模板结构和语法的更多信息，请参阅《AWS CloudFormation 用户指南》中的 AWS CloudFormation 模板部分。

您还可以利用多个模板片段构建模板。模板片段提供一些示例，以说明如何为特定资源编写模板。例如，您可以查看用于 Amazon Elastic Compute Cloud (Amazon EC2) 实例、Amazon Simple Compute Service (Amazon S3) 域、Amazon Simple Storage Service (Amazon S3) 域、AWS CloudFormation 映射等的片段。片段按资源分组。可以在AWS CloudFormation用户指南的通用模板片段部分中，可以找到通用 AWS CloudFormation 片段。

在 Application Manager（操作台）中创建 CloudFormation 模板

通过使用 AWS Management Console，使用以下过程在 Application Manager 中创建 CloudFormation 模板。

在 Application Manager 中创建 CloudFormation 模板

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Application Manager。
3. 选择 CloudFormation 模板库，然后选择创建模板或选择现有模板，然后再选择操作、克隆。
4. 对于名称，输入模板名称，以帮助您识别模板创建的资源或堆栈的用途。
5. （可选）对于版本名称，输入名称或编号以标识模板版本。
6. 在代码编辑器部分中，选择 YAML 或者 JSON，然后输入或复制并粘贴模板代码。
7. （可选）在文档标签部分，将一个或多个标签键名称/值对应用于文档。
8. （可选）在权限部分中，输入 AWS 账户 ID 并选择添加帐户。此操作提供对模板的读取权限。帐户所有者可以配置和克隆模板，但无法编辑或删除模板。
9. 选择创建。模板保存在 Systems Manager (SSM) 文档服务中。

在 Application Manager 中创建 CloudFormation 模板（命令行）

在 JSON 或 YAML 中创建 CloudFormation 模板的内容后，您可以使用AWS Command Line Interface (AWS CLI) 或 AWS Tools for PowerShell 将模板另存为 SSM 文档。将每个*#######*替换为您自己的信息。

开始前的准备工作

安装并配置 AWS CLI 或 AWS Tools for PowerShell（如果尚未执行该操作）。有关信息，请参阅安装或更新 AWS CLI 的最新版本以及安装 AWS Tools for PowerShell。

Linux & macOS

```
aws ssm create-document \
    --content file://path/to/template_in_json_or_yaml \
    --name "a_name_for_the_template" \
    --document-type "CloudFormation" \
    --document-format "JSON_or_YAML" \
    --tags "Key=tag-key,Value=tag-value"
```

Windows

```
aws ssm create-document ^
    --content file://C:\path\to\template_in_json_or_yaml ^
    --name "a_name_for_the_template" ^
    --document-type "CloudFormation" ^
    --document-format "JSON_or_YAML" ^
    --tags "Key=tag-key,Value=tag-value"
```

PowerShell

```
$json = Get-Content -Path "C:\path\to\template_in_json_or_yaml | Out-String
New-SSMDocument `
    -Content $json `
    -Name "a_name_for_the_template" `
    -DocumentType "CloudFormation" `
    -DocumentFormat "JSON_or_YAML" `
    -Tags "Key=tag-key,Value=tag-value"
```

如果成功，该命令将返回类似于以下内容的响应。

```
{
    "DocumentDescription": {
        "Hash": "c1d9640f15fbdba6deb41af6471d6ace0acc22f213bdd1449f03980358c2d4fb",
        "HashType": "Sha256",
        "Name": "MyTestCFTemplate",
        "Owner": "428427166869",
        "CreatedDate": "2021-06-04T09:44:18.931000-07:00",
        "Status": "Creating",
        "DocumentVersion": "1",
        "Description": "My test template",
        "PlatformTypes": [],
        "DocumentType": "CloudFormation",
        "SchemaVersion": "1.0",
        "LatestVersion": "1",
        "DefaultVersion": "1",
        "DocumentFormat": "YAML",
        "Tags": [
            {
                "Key": "Templates",
                "Value": "Test"
            }
        ]
    }
```


编辑模板

使用以下过程在 Application Manager 中编辑 CloudFormation 模板。配置使用更新后模板的堆栈后，将在 CloudFormation 中提供模板更改。

在 Application Manager 中编辑 CloudFormation 模板的步骤

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Application Manager。
3. 选择 CloudFormation 模板库
4. 选择模板，然后依次选择操作、编辑。您无法更改模板的名称，但您可以更改其他所有详细信息。
5. 选择保存。模板将保存在 Systems Manager 文档服务中。

使用 Application Manager 管理 AWS CloudFormation 堆栈

Application Manager 是 AWS Systems Manager 中的一项工具，通过与 AWS CloudFormation 集成，可帮助您配置和管理应用程序的资源。您可以在 Application Manager 中创建、编辑和删除 CloudFormation 模板和资源。堆栈是可作为单个单元管理的一系列 AWS 资源。这意味着您可以通过使用 CloudFormation 堆栈，创建、更新或删除 AWS 资源。模板是 JSON 或 YAML 中的格式化文本文件，指定要在堆栈中配置的资源。本节包含以下信息：

主题

- 创建堆栈
- 更新堆栈

创建堆栈

以下过程介绍了如何通过使用 Application Manager 创建 CloudFormation 堆栈。堆栈基于模板。创建堆栈时，您可以选择现有模板或创建新模板。创建堆栈后，系统会立即尝试创建堆栈中标识的资源。系统成功配置资源后，可以在 Application Manager 和 CloudFormation 中创建模板和堆栈。

> **Note**
>
> 使用 Application Manager 创建堆栈不会产生任何费用，但您需要您在堆栈中创建的 AWS 资源。

使用 Application Manager 创建 CloudFormation 堆栈（操作台）

执行以下过程，在 AWS Management Console 中使用 Application Manager 创建新的配置。

创建 CloudFormation 堆栈

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Application Manager。
3. 依次选择创建应用程序，CloudFormation 堆栈。
4. 在准备模板部分中，选择一个选项。如果选择使用现有模板，您可以使用选择模板部分中的选项卡查找所需的模板。（如果选择其他选项之一，请完成模板准备向导。）
5. 选择模板名称旁边的按钮，然后选择下一步。

6. 在存储库的指定模板详细信息页面上，验证模板的详细信息，以确保流程创建所需的资源。

   - （可选）在标签 部分，将一个或多个标签键名称/值对应用于文档。
   - 标签是您分配给资源的可选元数据。通过使用标签，您可以按各种标准（如用途、所有者或环境）对资源进行分类。
   - 选择下一步。

7. 在存储库的编辑堆栈详细信息页面上，为堆栈名称输入一个名称，该名称可帮助您识别堆栈创建的资源或其用途。

   - 参数部分包含模板中指定的所有可选参数和必需参数。在每个字段中输入一个或多个参数。
   - （可选）在标签区域，将一个或多个标签键名称/值对应用到堆栈。
   - （可选）在权限部分中，指定 AWS Identity and Access Management (IAM) 角色名称或 IAM Amazon Resource Name (ARN)。系统使用指定的服务角色创建堆栈中指定的所有资源。如果您不指定 IAM 角色，则 AWS CloudFormation 使用系统根据您的用户凭证生成的临时会话。有关此 IAM 角色的更多信息，请参阅AWS CloudFormation用户指南中的 AWS CloudFormation 服务角色。
   - 选择下一步。

8. 在存储库的审核和配置页面上，查看堆栈的所有详细信息。选择编辑按钮进行更改。

9. 选择配置堆栈。

Application Manager 显示 CloudFormation 堆栈页面以及堆栈创建和部署的状态。如果 CloudFormation 无法创建和配置堆栈，请参阅AWS CloudFormation用户指南。

- 堆栈状态代码
- AWS CloudFormation 故障排除

在堆栈资源得到配置并开始运行后，用户可以使用创建资源的基础服务直接编辑资源。例如，用户可以使用 Amazon Elastic Compute Cloud (Amazon EC2) 控制台来更新创建为 CloudFormation 堆栈一部分的服务器实例。有些更改可能是偶然的，而有些更改可能会故意响应对时间敏感的操作事件。无论如何，在 CloudFormation 之外进行的更改会导致堆栈更新或删除操作变得复杂。您可以使用偏差检测或偏差状态来识别在 CloudFormation 管理之外进行配置更改的堆栈资源。有关偏差状态的更多信息，请参阅检测堆栈和资源的非托管配置更改。

使用 Application Manager 创建 CloudFormation 堆栈（命令行）

使用以下 AWS Command Line Interface (AWS CLI) 过程，通过使用作为 SSM 文档存储在 Systems Manager 中的 CloudFormation 模板来配置堆栈。将每个*#######*替换为您自己的信息。有关创建堆栈的其他 AWS CLI 过程，请参阅AWS CloudFormation用户指南中的创建堆栈。

开始前的准备工作

安装并配置 AWS CLI 或 AWS Tools for PowerShell（如果尚未执行该操作）。有关信息，请参阅安装或更新 AWS CLI 的最新版本以及安装 AWS Tools for PowerShell。

Linux & macOS

```
aws cloudformation create-stack \
    --stack-name a_name_for_the_stack \
    --template-url "ssm-doc://arn:aws:ssm:Region:account_ID:document/template_name"
 \
```

Windows

```
aws cloudformation create-stack ^
     --stack-name a_name_for_the_stack ^
     --template-url "ssm-doc://arn:aws:ssm:Region:account_ID:document/template_name"
 ^
```

PowerShell

```
New-CFNStack `
    -StackName "a_name_for_the_stack" `
    -TemplateURL "ssm-doc://arn:aws:ssm:Region:account_ID:document/template_name" `
```

更新堆栈

通过直接在 Application Manager 中编辑堆栈，您可以将更新部署到 CloudFormation 堆栈。利用直接更新，您可以指定针对模板或输入参数的更新。保存并部署更改后，CloudFormation 会根据您指定的更改更新 AWS 资源。

您可以使用更改集预览 CloudFormation 在您更新之前将对堆栈进行的更改。有关更多信息，请参阅AWS CloudFormation用户指南中的使用更改集更新堆栈。

在中 Application Manager 更新 CloudFormation 堆栈

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Application Manager。
3. 选择应用程序名称旁边的单选按钮，然后依次选择操作、更新堆栈。
4. 在存储库的指定模板源页面上，选择以下选项之一，然后选择下一步。
   - 选择使用堆栈中当前配置的模板代码查看模板。在版本列表中选择一个模板版本，然后选择下一步。
   - 选择切换到其他模板，为堆栈选择或创建新模板。
5. 完成模板的更改后，选择下一步。
6. 在编辑堆栈详细信息页面上，您可以编辑参数、标签和权限。您无法更改堆栈名称。进行更改，然后选择保存。
7. 在存储库的审核和配置页面上，查看堆栈的所有详细信息，然后选择配置堆栈。

使用 Application Manager 中的数据库集群

本部分包括帮助您使用 Amazon Elastic Container Service (Amazon ECS) 和 Amazon Elastic Kubernetes Service (Amazon EKS) 容器集群的主题，可帮助您在 Application Manager 中使用这些集群，这是 AWS Systems Manager 的一个组件。

内容

- 在 Application Manager 中使用 Amazon ECS
- 在 Application Manager 中使用 Amazon EKS
- 使用集群的运行手册

在 Application Manager 中使用 Amazon ECS

Application Manager 是 AWS Systems Manager 中的一项工具，可用于查看和管理 Amazon Elastic Container Service（Amazon ECS）集群基础设施。Application Manager 使用集群的 Amazon 资源名称（ARN）作为标签值，将标签应用于 Amazon ECS 集群。Application Manager 可提供集群中计算、联网和存储资源的组件运行时视图。

> **Note**
>
> 您不能在 Application Manager 中管理或查看有关容器的运行信息。您只能管理和查看托管 Amazon ECS 资源的基础设施的运行信息。

您可以在此页上执行的操作

您可以在此页面上执行以下操作：

- 选择管理集群在 Amazon ECS 中打开集群。
- 选择查看全部，可查看集群中资源的列表
- 选择在 CloudWatch 中查看以查看 Amazon CloudWatch 中的资源警报。
- 选择 Manage nodes（管理节点）或者 Manage Fargate profiles（管理 Fargate 配置文件），以便在 Amazon ECS 中查看这些资源。
- 选择资源 ID 以在创建资源 ID 的控制台中查看有关该资源 ID 的详细信息。
- 查看与您的集群相关的 OpsItems 的列表。
- 查看已在集群上运行的运行手册历史记录。

要打开 ECS 集群

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Application Manager。
3. 在容器集群部分中，选择 ECS 集群。
4. 在列表中选择一个集群。Application Manager 打开 概览选项卡。

在 Application Manager 中使用 Amazon EKS

Application Manager 是 AWS Systems Manager 中的一项工具，可与 Amazon Elastic Kubernetes Service（Amazon EKS）集成，从而提供有关 Amazon EKS 集群基础设施运行状况的信息。Application Manager 使用集群的 Amazon 资源名称（ARN）作为标签值，将标签应用于 Amazon EKS 集群。Application Manager 可提供有关集群中计算、联网和存储资源的组件运行时视图。

> **Note**
>
> 您无法在 Application Manager 管理或查看有关 Amazon EKS 容器组或容器的操作信息。您只能管理和查看托管 Amazon EKS 资源的基础设施的运行信息。

您可以在此页上执行的操作

您可以在此页面上执行以下操作：

- 选择管理集群在 Amazon EKS 中打开集群。
- 选择查看全部，可查看集群中资源的列表。
- 选择在 CloudWatch 中查看以查看 Amazon CloudWatch 中的资源警报。
- 选择 Manage nodes（管理节点）或者 Manage Fargate profiles（管理 Fargate 配置文件），以便在 Amazon EKS 中查看这些资源。
- 选择资源 ID 以在创建资源 ID 的控制台中查看有关该资源 ID 的详细信息。
- 查看与您的集群相关的 OpsItems 的列表。
- 查看已在集群上运行的运行手册的历史记录。

要打开 EKS 集群应用程序

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Application Manager。
3. 在容器集群部分中，选择 EKS 集群。
4. 在列表中选择一个集群。Application Manager 打开概览选项卡。

使用集群的运行手册

您可以通过使用 Systems Manager Automation 运行手册验证来自 Application Manager（AWS Systems Manager 中的一项工具）的 AWS 资源问题。在从 Application Manager 集群中选择启动运行手册时，系统会根据集群中的资源类型显示经过筛选的运行手册列表。当您选择要启动的运行手册时，Systems Manager 会打开执行自动化文档页面。

开始前的准备工作

在您开始从 Application Manager 中启动运行手册之前，执行以下操作：

- 验证您具有启动运行手册的权限。有关更多信息，请参阅 设置自动化。
- 查看有关启动运行手册的自动化过程文档。有关更多信息，请参阅 运行由 Systems Manager Automation 支持的自动化操作。
- 如果您打算同时在多个资源上启动运行手册，请查看有关使用目标和速率控制的文档。有关更多信息，请参阅 大规模运行自动化操作。

要从 Application Manager 启动集群的运行手册

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Application Manager。
3. 在容器集群部分，选择容器类型。
4. 在列表中选择集群。Application Manager 打开概览选项卡。
5. 在 Runbooks（运行手册）选项卡上，选择 Start runbook（启动运行手册）。Application Manager 在新选项卡中打开 Execute automation document（执行自动化文档）页面。有关执行自动化文档页面中的信息，请参阅 运行由 Systems Manager Automation 支持的自动化操作。

AWS Systems Manager Parameter Store

Parameter Store 是 AWS Systems Manager 中的一项工具，可提供安全的分层存储用于配置数据管理和密钥管理。您可以将密码、数据库字符串、Amazon Machine Image (AMI) ID 和许可证代码等数据存储为参数值。可以将值存储为纯文本或加密数据。您可以使用创建 Systems Manager 参数时指定的唯一名称，在脚本、命令、SSM 文档以及配置和自动化工作流中引用该参数。要开始使用 Parameter Store，请打开 Systems Manager 控制台。在导航窗格中，请选择 Parameter Store。

Parameter Store 还与 Secrets Manager 进行了集成。您可以在使用其他已支持引用 Parameter Store 参数的 AWS 服务时检索 Secrets Manager 密钥。有关更多信息，请参阅 通过 Parameter Store 参数引用 AWS Secrets Manager 密钥。

> **Note**
>
> 要实施密码轮换生命周期，请使用 AWS Secrets Manager。您可以使用 Secrets Manager 在数据库凭证、API 密钥和其他密钥的整个生命周期内对其进行轮换、管理和检索。有关更多信息，请参阅 AWS Secrets Manager 用户指南中的什么是AWS Secrets Manager？。

我的组织如何从 Parameter Store 获益？

Parameter Store 具备下列优势：

- 使用安全、可扩展的托管密钥管理服务，无需管理服务器。
- 通过将数据与代码分离来改善安保状况。
- 分层存储配置数据和加密的字符串，而且可跟踪版本。
- 实现以细粒度控制和审核访问。
- 可靠地存储参数，因为 Parameter Store 托管在 AWS 区域的多个可用区中。

谁应该使用 Parameter Store？

- 任何希望集中管理配置数据的 AWS 客户。
- 希望存储不同登录和引用流的软件开发人员。
- 希望在密钥和密码发生更改或未更改时接收通知的管理员。

Parameter Store 具有哪些功能？

- 更改通知

  您可以为参数和参数策略配置更改通知并调用自动操作。有关更多信息，请参阅 基于 Parameter Store 事件设置通知或触发操作。
- 组织参数

  您可以单独标记参数，以便根据为其分配的标签识别一个或多个参数。例如，可以为特定环境或部门标记参数。
- 标签版本

  您可以通过创建标签将别名与参数版本关联。标签可帮助您在存在多个版本时记住参数版本的用途。
- 数据验证

  您可以创建指向 Amazon Elastic Compute Cloud (Amazon EC2) 实例的参数，Parameter Store 会验证这些参数，以确保其引用预期的资源类型，此类资源存在，并且客户有权使用此类资源。例如，您可以使用 Amazon Machine Image (AMI) ID 创建一个参数，作为 `aws:ec2:image` 数据类型的值。Parameter Store 会执行异步验证操作，以确保参数值满足 AMI ID 的格式设置要求，并且指定的 AMI 在您的 AWS 账户 中可用。

- 引用密钥

  Parameter Store 现已与 AWS Secrets Manager 集成，因此您可以在使用其他已支持引用 Parameter Store 参数的 AWS 服务时检索 Secrets Manager 密钥。

- 与其他账户共享参数

  您可以选择将配置数据集中保管在单个 AWS 账户 中，以便与其他需要访问参数的账户共享参数。

- 可从其他 AWS 服务访问

  您可以将 Parameter Store 参数与其他 Systems Manager 工具和 AWS 服务结合使用，从集中存储中检索密钥和配置数据。参数可与 Systems Manager 工具配合使用，例如 Run Command、Automation 和State Manager（都是 AWS Systems Manager 中工具）。您还可以在许多其他 AWS 服务中引用参数，其中包括：

  - Amazon Elastic Compute Cloud（Amazon EC2）
  - Amazon Elastic Container Service (Amazon ECS)
  - AWS Secrets Manager
  - AWS Lambda
  - AWS CloudFormation
  - AWS CodeBuild
  - AWS CodePipeline
  - AWS CodeDeploy

- 与其他 AWS 服务集成

  配置与以下 AWS 服务的集成以实现加密、通知、监控和审计：

  - AWS Key Management Service (AWS KMS)
  - Amazon Simple Notification Service (Amazon SNS)
  - Amazon CloudWatch：有关更多信息，请参阅 为参数和参数策略配置 Eventbridge 规则。
  - Amazon EventBridge：有关更多信息，请参阅 使用 Amazon SNS 通知监控 Systems Manager 状态更改 和 引用：Amazon EventBridge 事件模式和 Systems Manager 类型。
  - AWS CloudTrail：有关更多信息，请参阅 使用 AWS CloudTrail 记录 AWS Systems Manager API 调用。

什么是参数？

Parameter Store 参数是保存在 Parameter Store 中的任何数据片段，例如文本数据块、名称列表、密码、AMI ID、许可证密钥等。可以在脚本、命令和 SSM 文档中集中安全地引用此数据。

在引用参数时，使用以下惯例指定参数名称。

{{ssm:*parameter-name*}}

> **Note**
>
> 参数不能被引用或嵌套在其他参数的值中。参数值中不能包含 {{}} 或 {{ssm:*parameter-name*}}。

Parameter Store 支持三种类型的参数：String、StringList 和 SecureString。

有一个例外，当创建或更新参数时，您将以明文形式输入参数值，并且 Parameter Store 不对输入的文本执行验证。但是，对于 String 参数，您可以将数据类型指定为 aws:ec2:image，并且 Parameter Store 会验证您输入的值是否为 Amazon EC2 AMI 的正确格式；例如：ami-12345abcdeEXAMPLE。

参数类型：String

默认情况下，String 参数的值由您输入的任何文本数据块组成。例如：

- abc123
- Example Corp
-

参数类型：StringList

StringList 参数值包含以逗号分隔的值列表，如下面的示例所示。

```
Monday,Wednesday,Friday
```

```
CSV,TSV,CLF,ELF,JSON
```

参数类型：SecureString

SecureString 参数的值是需要以安全的方式存储和引用的任何敏感数据。如果您有不希望用户更改或以明文形式引用的数据（例如密码或许可证密钥），则应使用 SecureString 数据类型创建这些参数。

> **Important**
>
> 请勿将敏感数据存储在 String 或 StringList 的参数中。对于必须保持加密状态的所有敏感数据，请仅使用 SecureString 参数类型。
> 有关更多信息，请参阅 使用 AWS CLI 来创建 SecureString 参数。

建议您在以下情形中使用 SecureString 参数：

- 您想要跨 AWS 服务使用数据/参数，但又不想以明文形式在命令、函数、代理日志或 CloudTrail 日志中公开这些值。
- 您想要控制可以访问敏感数据的人。
- 您希望在有人访问敏感数据时进行审核 (CloudTrail)。
- 您希望对您的敏感数据进行加密，并想用自己的加密密钥管理访问。

> **Important**
>
> 只会加密 SecureString 参数的值。不会加密参数名称、描述和其他属性。

您可以将 SecureString 参数类型用于要加密的文本数据，例如密码、应用程序密钥、机密配置数据或需要保护的任何其他类型的数据。使用 AWS KMS 密钥对 SecureString 数据进行加密和解密。您可以使用 AWS 提供的默认 KMS 密钥，也可以创建和使用您自己的 AWS KMS key。（如果您想要限制用户对 SecureString 参数的访问，可以使用自己的 AWS KMS key。有关更多信息，请参阅 有关使用 AWS 默认密钥和客户托管密钥的 IAM 权限。）

您还可以将 SecureString 参数与其他 AWS 服务结合使用。在以下示例中，Lambda 函数使用 GetParameters API 检索 SecureString 参数。

```
import json
import boto3
```

```
ssm = boto3.client('ssm', 'us-east-2')
def get_parameters():
    response = ssm.get_parameters(
        Names=['LambdaSecureString'],WithDecryption=True
    )
    for parameter in response['Parameters']:
        return parameter['Value']

def lambda_handler(event, context):
    value = get_parameters()
    print("value1 = " + value)
    return value  # Echo back the first key value
```

AWS KMS 加密和定价

如果您在创建参数时选择了 SecureString 参数类型，Systems Manager 将使用 AWS KMS 加密参数值。

> **Important**
>
> Parameter Store 仅支持对称加密 KMS 密钥。不能使用非对称加密 KMS 密钥来加密您的参数。要获取确定 KMS 密钥是对称还是非对称密钥的帮助，请参阅 AWS Key Management Service 开发人员指南中的识别对称 KMS 密钥和非对称 KMS 密钥。

创建 SecureString 参数不收取 Parameter Store 任何费用，但使用 AWS KMS 加密的费用确实适用。有关更多信息，请参阅 AWS Key Management Service 定价。

有关 AWS 托管式密钥和客户管理型密钥的更多信息，请参阅《AWS Key Management Service 开发人员指南》中的 AWS Key Management Service 概念。有关 Parameter Store 和 AWS KMS 加密的更多信息，请参阅 AWS Systems ManagerParameter Store 如何使用 AWS KMS。

> **Note**
>
> 要查看 AWS 托管式密钥，请使用 AWS KMS DescribeKey 操作。此 AWS Command Line Interface（AWS CLI）示例使用 DescribeKey 查看 AWS 托管式密钥。
>
> ```
> aws kms describe-key --key-id alias/aws/ssm
> ```

更多信息

- 在 Parameter Store 中创建 SecureString 参数并将节点加入域（PowerShell）
- 使用 Parameter Store 安全地访问 CodeDeploy 中的密钥和配置数据
- 有关 Amazon EC2 Systems Manager Parameter Store 的有趣文章

参数大小限制

根据您使用的参数层，Parameter Store 对参数值有不同的大小限制：

- 标准参数：最大值大小为 4 KB
- 高级参数：最大值大小为 8 KB

如果需要存储大于 4 KB 的参数值，则必须使用高级参数层。高级参数提供了附加功能，但会在您的 AWS 账户中产生费用。有关参数层及其功能的更多信息，请参阅 管理参数层。

有关 Parameter Store 配额和限制的完整列表，请参阅《AWS 一般参考》中的 AWS Systems Manager endpoints and quotas。

设置 Parameter Store

在 Parameter Store（AWS Systems Manager 中的一项工具）中设置参数之前，请先配置 AWS Identity and Access Management（IAM）策略，为账户中的用户提供执行您指定操作的权限。

本部分包含有关如何使用 IAM 控制台手动配置这些策略，以及如何将它们分配给用户和用户组的信息。您还可以创建并分配策略，用于控制可在托管式节点上运行哪些参数操作。

本部分还包含有关如何创建 Amazon EventBridge 规则，以使您可以接收有关 Systems Manager 参数更改的通知的信息。您还可以使用 Eventbridge 规则，根据 Parameter Store 中的更改在 AWS 中调用其他操作。

内容

- 使用 IAM 策略限制对 Parameter Store 参数的访问
- 管理参数层
- 提高或重置 Parameter Store 吞吐量
- 基于 Parameter Store 事件设置通知或触发操作

使用 IAM 策略限制对 Parameter Store 参数的访问

您可以使用 AWS Identity and Access Management (IAM) 限制对 AWS Systems Manager 参数的访问。更具体地说，您可以创建 IAM policy 来限制对以下 API 操作的访问：

- DeleteParameter
- DeleteParameters
- DescribeParameters
- GetParameter
- GetParameters
- GetParameterHistory
- GetParametersByPath
- PutParameter

当使用 IAM policy 限制对 Systems Manager 参数的访问时，建议您创建和使用限制性 IAM policy。例如，以下策略允许用户为有限的一组资源调用 `DescribeParameters` 和 `GetParameters` API 操作。这意味着，用户可以获取有关以 `prod-*` 开头的所有参数的信息并使用这些参数。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeParameters"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameters"
            ],
            "Resource": "arn:aws:ssm:us-east-2:123456789012:parameter/prod-*"
        }
    ]
```

```
}
```

> **Important**
>
> 如果用户有权访问某个路径，则该用户可以访问该路径的所有级别。例如，如果某个用户有权访问路径 /a，则该用户也可以访问 /a/b。即使用户在 IAM 中已被显式拒绝访问参数 /a/b，他们仍能够以递归方式为 /a 调用 GetParametersByPath API 操作并查看 /a/b。

对于受信任的管理员，您可以通过使用类似以下示例的策略提供对所有 Systems Manager 参数 API 操作的访问权限。此策略将授予用户对所有以 dbserver-prod-* 开头的生产参数的完全访问权限。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:PutParameter",
                "ssm:DeleteParameter",
                "ssm:GetParameterHistory",
                "ssm:GetParametersByPath",
                "ssm:GetParameters",
                "ssm:GetParameter",
                "ssm:DeleteParameters"
            ],
            "Resource": "arn:aws:ssm:us-east-2:111122223333:parameter/dbserver-
prod-*"
        },
        {
            "Effect": "Allow",
            "Action": "ssm:DescribeParameters",
            "Resource": "*"
        }
    ]
}
```

拒绝权限

每个 API 都是唯一的，并且具有不同的操作和权限，您可以单独允许或拒绝这些操作和权限。任何策略中的显式拒绝将覆盖允许。

> **Note**
>
> 默认的 AWS Key Management Service (AWS KMS) 密钥具有对 AWS 账户中的所有 IAM 委托人的Decrypt权限。如果您希望对账户中的 SecureString 参数拥有不同的访问级别，不建议您使用默认密钥。

如果您希望所有检索参数值的 API 操作都具有相同的行为，则可以在策略中使用类似 GetParameter* 的模式。以下示例显示了如何拒绝所有以 prod-* 开头的参数的 GetParameter、GetParameters、GetParameterHistory 和 GetParametersByPath。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "ssm:GetParameter*"
            ],
            "Resource": "arn:aws:ssm:us-east-2:123456789012:parameter/prod-*"
        }
    ]
}
```

以下示例显示了如何拒绝一些命令，同时允许用户对以 prod-* 开头的所有参数执行其他命令。

JSON

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

```
        {
            "Effect": "Deny",
            "Action": [
                "ssm:PutParameter",
                "ssm:DeleteParameter",
                "ssm:DeleteParameters",
                "ssm:DescribeParameters"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParametersByPath",
                "ssm:GetParameters",
                "ssm:GetParameter",
                "ssm:GetParameterHistory"
            ],
            "Resource": "arn:aws:ssm:us-east-2:111122223333:parameter/prod-*"
        }
    ]
}
```

> **Note**
>
> 参数历史记录包括当前版本在内的所有参数版本。因此，如果拒绝用户使用 GetParameter、GetParameters 和 GetParameterByPath 的权限，但允许其使用 GetParameterHistory 的权限，则用户可以使用 GetParameterHistory 查看当前参数（包括 SecureString 参数）。

仅允许特定参数在节点上运行

您可以控制访问权限，以使托管式节点只能运行指定参数。

如果您在创建参数时选择 SecureString 参数类型，Systems Manager 将使用 AWS KMS 加密参数值。AWS KMS 使用 AWS 托管式密钥 或客户托管密钥来加密值。有关 AWS KMS 和 AWS KMS key 的更多信息，请参阅 AWS Key Management Service Developer Guide 。

您可以通过从 AWS CLI 运行以下命令来查看 AWS 托管式密钥：

```
aws kms describe-key --key-id alias/aws/ssm
```

在以下示例中，允许节点获取仅用于以 prod- 开头的参数的参数值。如果参数是 SecureString 参数，则节点会使用 AWS KMS 解密该字符串。

> **Note**
>
> 如以下示例所示，实例策略分配到 IAM 中的实例角色。有关配置对 Systems Manager 功能的访问权限的详细信息，包括如何将策略分配给用户和实例，请参阅 使用 Systems Manager 管理 EC2 实例。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameters"
            ],
            "Resource": [
                "arn:aws:ssm:us-east-2:111122223333:parameter/prod-*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-2:111122223333:key/4914ec06-e888-4ea5-
a371-5b88eEXAMPLE"
            ]
        }
    ]
}
```

有关使用 AWS 默认密钥和客户托管密钥的 IAM 权限

Parameter Store SecureString 参数使用 AWS KMS 密钥进行加密和解密。您可以选择使用 AWS KMS key 或 AWS 提供的默认 KMS 密钥对 SecureString 参数进行加密。

使用客户托管密钥时，授予用户访问参数或参数路径的权限的 IAM policy 必须提供对密钥的显式 kms:Encrypt 权限。例如，以下策略允许用户在指定的 AWS 区域和 AWS 账户中创建、更新和查看以 prod- 开头的 SecureString 参数。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:PutParameter",
                "ssm:GetParameter",
                "ssm:GetParameters"
            ],
            "Resource": [
                "arn:aws:ssm:us-east-2:111122223333:parameter/prod-*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:Encrypt",
                "kms:GenerateDataKey"
            ],
            "Resource": [
                "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-12345EXAMPLE"
            ]
        }
    ]
}
```

[1]使用指定的客户托管密钥创建加密的高级参数需要 kms:GenerateDataKey 权限。

相比之下，客户账户中的所有用户都可以访问默认 AWS 托管密钥。如果您使用此默认密钥加密 SecureString 参数，并且不希望用户使用 SecureString 参数，则其 IAM policy 必须显式拒绝对默认密钥的访问权限，如以下策略示例所示。

> **Note**
>
> 您可以在AWS 托管密钥页面的 AWS KMS 控制台中找到默认密钥的 Amazon Resource Name (ARN)。默认密钥是在 Alias (别名) 列中使用 aws/ssm 标识的密钥。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "kms:Decrypt",
                "kms:GenerateDataKey"
            ],
            "Resource": [
                "arn:aws:kms:us-east-2:111122223333:key/abcd1234-ab12-cd34-ef56-
abcdeEXAMPLE"
            ]
        }
    ]
}
```

如果您需要对账户中的 SecureString 参数进行精细访问控制，则应使用客户托管密钥来保护和限制对这些参数的访问。我们还建议使用 AWS CloudTrail 监视 SecureString 参数活动。

有关更多信息，请参阅以下主题：

- IAM 用户指南中的策略评估逻辑
- AWS Key Management Service Developer Guide 中的在 AWS KMS 中使用密钥策略
- AWS CloudTrail 用户指南中的使用 CloudTrail 事件历史记录查看事件

管理参数层

Parameter Store 是 AWS Systems Manager 中的一项工具，包含了标准参数和高级参数。需单独配置参数以使用标准参数层（默认层）或高级参数层。

您可以随时将标准参数更改为高级参数，但不能将高级参数恢复为标准参数。这是因为将高级参数恢复为标准参数会导致系统将参数的大小从 8KB 缩小到 4KB，从而导致数据丢失。恢复参数还会移除所有应用于参数的策略。此外，高级参数使用不同于标准参数的加密形式。有关更多信息，请参阅 AWS Key Management Service 开发人员指南中的 AWS Systems ManagerParameter Store 如何使用 AWS KMS。

如果您不再需要高级参数，或者如果您不想再为高级参数支付费用，请删除它并将它重新创建为新的标准参数。

下表介绍了两个层之间的区别。

	Standard	高级
允许的参数的总数 （每个 AWS 账户和 AWS 区域）	10000	100000
参数值的最大大小	4 KB	8 KB
提供参数策略	否	是 有关更多信息，请参阅 在 Parameter Store 中分配参数策略。
成本	无额外费用	需支付费用 有关更多信息，请参阅 Parameter Store 的 AWS Systems Manager 定价。

主题

- 指定默认参数层

- 将标准参数更改为高级参数

指定默认参数层

在创建或更新参数的请求（即 PutParameter 操作）中，您可以指定要在请求中使用的参数层。以下是使用 AWS Command Line Interface (AWS CLI) 的示例。

Linux & macOS

```
aws ssm put-parameter \
    --name "default-ami" \
    --type "String" \
    --value "t2.micro" \
    --tier "Standard"
```

Windows

```
aws ssm put-parameter ^
    --name "default-ami" ^
    --type "String" ^
    --value "t2.micro" ^
    --tier "Standard"
```

无论何时在请求中指定层，Parameter Store 都会根据您的请求创建或更新参数。不过，如果没有在请求中显式指定层，Parameter Store 默认层设置会确定在哪个层中创建参数。

在开始使用 Parameter Store 时，默认层是标准参数层。如果使用高级参数层，则可以将以下其中一项指定为默认值：

- 高级：在使用该选项时，Parameter Store 会将所有请求作为高级参数进行评估。
- 智能分层：在使用该选项时，Parameter Store 会评估每个请求以确定参数是标准还是高级参数。

  如果请求不包含任何需要使用高级参数的选项，则会在标准参数层中创建参数。如果在请求中包含一个或多个需要使用高级参数的选项，则 Parameter Store 会在高级参数层中创建参数。

智能分层的优点

以下是您可能选择智能分层作为默认层的原因。

成本控制 - 除非绝对需要使用高级参数，否则智能分层将始终创建标准参数以帮助控制参数相关成本。

自动升级到高级参数层 - 在对需要将标准参数升级到高级参数的代码进行更改时，智能分层将为您处理转换。您无需更改代码即可进行升级。

以下是一些自动升级的示例：

- 您的 AWS CloudFormation 模板在运行时预置了大量参数。在该过程导致您达到标准参数层中的 1 万个参数配额时，智能分层会自动将您升级到高级参数层，并且 AWS CloudFormation 过程不会中断。
- 您需将证书值存储在一个参数中，定期轮换证书值，并且内容小于标准参数层的 4 KB 配额。如果替换证书值超过 4 KB，智能分层会自动将参数升级到高级参数层。
- 您希望将大量现有标准参数与参数策略相关联，这需要使用高级参数层。并非必须在更新参数的所有调用中包含 `--tier Advanced` 选项，智能分层会自动将参数升级到高级参数层。无论何时引入高级参数层条件，智能分层选项都会将参数从标准升级到高级。

需要使用高级参数的选项包括以下几项：

- 参数的内容大小超过 4 KB。
- 参数使用参数策略。
- 您的 AWS 账户在当前 AWS 区域中已具有超过 10000 个参数。

默认层选项

您可以指定为默认值的层选项包括以下内容。

- Standard (标准) – 在开始使用 Parameter Store 时，默认层是标准参数层。通过使用标准参数层，您可以为 AWS 账户中的每个 AWS 区域创建 10000 个参数。每个参数的内容大小最大可等于 4 KB。标准参数不支持参数策略。使用标准参数层不会产生额外的费用。选择标准作为默认层表示 Parameter Store 将始终尝试为未指定层的请求创建标准参数。
- Advanced (高级) – 使用高级参数层最多可以为 AWS 账户中的每个 AWS 区域创建 100000 个参数。每个参数的内容大小最大可等于 8 KB。高级参数支持参数策略。要共享参数，该参数就必须在高级参数层中。使用高级参数层会产生相应的费用。有关更多信息，请参阅 Parameter Store 的 AWS Systems Manager 定价。选择高级作为默认层表示 Parameter Store 将始终尝试为未指定层的请求创建高级参数。

> **Note**
>
> 选择高级参数层时，您需要明确授权 AWS 就您创建的任何高级参数向您的账户收取相应费用。

- 智能分层 – 使用智能分层选项，Parameter Store 可以根据请求内容确定是使用标准参数层，还是使用高级参数层。例如，如果运行命令以创建内容小于 4KB 的参数，您的 AWS 账户的当前 AWS 区域中的参数少于 10000 个，并且未指定参数策略，则会创建标准参数。如果运行命令以创建内容超过 4KB 的参数，您的 AWS 账户的当前 AWS 区域中的参数超过 10000 个，或者您指定了参数策略，则会创建高级参数。

> **Note**
>
> 选择智能分层时，您需要明确授权 AWS 就您创建的任何高级参数向您的账户收取相应费用。

您可以随时更改 Parameter Store 默认层设置。

配置权限以指定 Parameter Store 默认层

您可以执行以下操作之一，以验证您是否在 AWS Identity and Access Management (IAM) 中具有相应权限，能够更改 Parameter Store 中的默认参数层：

- 确保将 `AdministratorAccess` 策略附加到您的 IAM 实体（例如用户、组或角色）。
- 确保您有权使用以下 API 操作更改默认层设置：
  - GetServiceSetting
  - UpdateServiceSetting
  - ResetServiceSetting

向 IAM 实体授予以下权限，以允许用户查看和更改 AWS 账户 内特定 AWS 区域 中参数的默认层设置

JSON

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetServiceSetting"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:UpdateServiceSetting"
            ],
            "Resource": "arn:aws:ssm:us-east-1:111122223333:servicesetting/ssm/
parameter-store/default-parameter-tier"
        }
    ]
}
```

管理员可以通过分配以下权限来指定只读权限。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetServiceSetting"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Deny",
            "Action": [
                "ssm:ResetServiceSetting",
                "ssm:UpdateServiceSetting"
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

要提供访问权限，请为您的用户、组或角色添加权限：

- AWS IAM Identity Center 中的用户和群组：

  创建权限集合。按照《AWS IAM Identity Center 用户指南》中创建权限集的说明进行操作。
- 通过身份提供商在 IAM 中托管的用户：

  创建适用于身份联合验证的角色。按照《IAM 用户指南》中针对第三方身份提供商创建角色（联合身份验证）的说明进行操作。
- IAM 用户：
  - 创建您的用户可以担任的角色。按照《IAM 用户指南》中为 IAM 用户创建角色的说明进行操作。
  - （不推荐使用）将策略直接附加到用户或将用户添加到用户组。按照《IAM 用户指南》中向用户添加权限（控制台）中的说明进行操作。

使用 控制台来指定或更改 Parameter Store 默认层

以下过程介绍了如何使用 Systems Manager 控制台指定或更改当前 AWS 账户和 AWS 区域的默认参数层。

> **Tip**
>
> 如果尚未创建参数，则可以使用 AWS Command Line Interface (AWS CLI) 或 AWS Tools for Windows PowerShell 更改默认参数层。有关更多信息，请参阅 使用 AWS CLI 指定或更改 Parameter Store 默认层 和 指定或更改 Parameter Store 默认层 (PowerShell)。

指定或更改 Parameter Store 默认层

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Parameter Store。
3. 选择设置选项卡。
4. 选择 Change default tier (更改默认层)。
5. 选择以下任一选项。

- Standard
- 高级
- 智能分层

有关这些选项的信息，请参阅 指定默认参数层。

6. 查看消息，然后选择确认。

如果要稍后更改默认层设置，请重复该过程并指定不同的默认层选项。

使用 AWS CLI 指定或更改 Parameter Store 默认层

以下过程介绍了如何使用 AWS CLI 更改当前 AWS 区域和 AWS 账户的默认参数层设置。

使用 AWS CLI 指定或更改 Parameter Store 默认层

1. 打开 AWS CLI 并运行以下命令，以更改 AWS 账户中特定 AWS 区域的默认参数层设置。

```
aws ssm update-service-setting --setting-id arn:aws:ssm:region:account-
id:servicesetting/ssm/parameter-store/default-parameter-tier --setting-value tier-
option
```

region 表示 AWS Systems Manager 支持的 AWS 区域 的标识符，例如美国东部（俄亥俄）区域的 us-east-2。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

tier-option 值包括 Standard、Advanced 和 Intelligent-Tiering。有关这些选项的信息，请参阅 指定默认参数层。

如果此命令成功，则无任何输出。

2. 运行以下命令以查看当前 AWS 账户 和 AWS 区域 中 Parameter Store 的当前默认参数层设置。

```
aws ssm get-service-setting --setting-id arn:aws:ssm:region:account-
id:servicesetting/ssm/parameter-store/default-parameter-tier
```

系统返回类似于以下内容的信息。

```
{
```

```
    "ServiceSetting": {
        "SettingId": "/ssm/parameter-store/default-parameter-tier",
        "SettingValue": "Advanced",
        "LastModifiedDate": 1556551683.923,
        "LastModifiedUser": "arn:aws:sts::123456789012:assumed-role/Administrator/
Jasper",
        "ARN": "arn:aws:ssm:us-east-2:123456789012:servicesetting/ssm/parameter-
store/default-parameter-tier",
        "Status": "Customized"
    }
}
```

如果要再次更改默认层设置，请重复该过程并指定不同的 `SettingValue` 选项。

指定或更改 Parameter Store 默认层 (PowerShell)

以下过程介绍了如何使用 Tools for Windows PowerShell 更改 Amazon Web Services 账户中特定 AWS 区域的默认参数层设置。

使用 PowerShell 指定或更改 Parameter Store 默认层

1. 使用 AWS Tools for PowerShell (Tools for PowerShell) 更改当前 AWS 账户和 AWS 区域中的 Parameter Store 默认层。

   ```
   Update-SSMServiceSetting -SettingId "arn:aws:ssm:region:account-id:servicesetting/
   ssm/parameter-store/default-parameter-tier" -SettingValue "tier-option" -
   Region region
   ```

   region 表示 AWS Systems Manager 支持的 AWS 区域 的标识符，例如美国东部（俄亥俄）区域的 `us-east-2`。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

   tier-option 值包括 `Standard`、`Advanced` 和 `Intelligent-Tiering`。有关这些选项的信息，请参阅 指定默认参数层。

   如果此命令成功，则无任何输出。
2. 运行以下命令以查看当前 AWS 账户 和 AWS 区域 中 Parameter Store 的当前默认参数层设置。

   ```
   Get-SSMServiceSetting -SettingId "arn:aws:ssm:region:account-id:servicesetting/ssm/
   parameter-store/default-parameter-tier" -Region region
   ```

region 表示 AWS Systems Manager 支持的 AWS 区域 的标识符，例如 us-east-2 对应美国东部（俄亥俄）区域。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

系统返回类似于以下内容的信息。

```
ARN : arn:aws:ssm:us-east-2:123456789012:servicesetting/ssm/parameter-store/
default-parameter-tier
LastModifiedDate : 4/29/2019 3:35:44 PM
LastModifiedUser : arn:aws:sts::123456789012:assumed-role/Administrator/Jasper
SettingId        : /ssm/parameter-store/default-parameter-tier
SettingValue     : Advanced
Status           : Customized
```

如果要再次更改默认层设置，请重复该过程并指定不同的 SettingValue 选项。

将标准参数更改为高级参数

使用以下过程可将现有的标准参数更改为高级参数。有关如何创建新的高级参数的信息，请参阅 在 Systems Manager 中创建 Parameter Store 参数。

将标准参数更改为高级参数

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Parameter Store。
3. 选择一个参数，然后选择 Edit (编辑)。
4. 对于 Description (说明)，输入有关此参数的信息。
5. 选择 Advanced (高级)。
6. 对于 Value (值)，输入此参数的值。高级参数有最大值限制 8 KB。
7. 选择保存更改。

提高或重置 Parameter Store 吞吐量

提高 Parameter Store 吞吐量将提高 Parameter Store（AWS Systems Manager 中的一项工具）每秒可以处理的事务（TPS）的最大数量。通过提高吞吐量，您能够以更大的量运行 Parameter Store 来支

持需要并发访问多个参数的应用程序和工作负载。您可以在 Settings（设置）选项卡上将配额提高到最大吞吐量。

Parameter Store 吞吐量设置适用于当前 AWS 账户 和 AWS 区域 中的所有 IAM 用户创建的所有事务。吞吐量设置适用于标准参数和高级参数。

> **Note**
>
> 通常情况下，更新会立即显示在服务配额中。只有在极少数情况下，可能需要长达 24 小时才能显示更新。

有关最大吞吐量默认值和最大值限制的更多信息，请参阅 AWS Systems Manager endpoints and quotas。

提高吞吐量配额将向您的 AWS 账户 收费。有关更多信息，请参阅 AWS Systems Manager 定价。

主题

- 配置权限来更改 Parameter Store 吞吐量
- 使用控制台来提高或重置吞吐量
- 使用 AWS CLI 来提高或重置吞吐量
- 提高或重置吞吐量（PowerShell）

配置权限来更改 Parameter Store 吞吐量

请确认您在 IAM 中有权通过执行以下操作之一来更改 Parameter Store 吞吐量：

- 确保将 `AdministratorAccess` 策略附加到您的 IAM 实体（用户、组或角色）。
- 确保您有权使用以下 API 操作更改吞吐量服务设置：
  - GetServiceSetting
  - UpdateServiceSetting
  - ResetServiceSetting

向 IAM 实体授予以下权限，以允许用户查看和更改 AWS 账户 内特定 AWS 区域 中参数的参数-吞吐量设置。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetServiceSetting"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:UpdateServiceSetting"
            ],
            "Resource": "arn:aws:ssm:us-east-1:111122223333:servicesetting/ssm/
parameter-store/high-throughput-enabled"
        }
    ]
}
```

管理员可以通过分配以下权限来指定只读权限。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetServiceSetting"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Deny",
            "Action": [
                "ssm:ResetServiceSetting",
```

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

要提供访问权限，请为您的用户、组或角色添加权限：

- AWS IAM Identity Center 中的用户和群组：

  创建权限集合。按照《AWS IAM Identity Center 用户指南》中创建权限集的说明进行操作。
- 通过身份提供商在 IAM 中托管的用户：

  创建适用于身份联合验证的角色。按照《IAM 用户指南》中针对第三方身份提供商创建角色（联合身份验证）的说明进行操作。
- IAM 用户：
  - 创建您的用户可以担任的角色。按照《IAM 用户指南》中为 IAM 用户创建角色的说明进行操作。
  - （不推荐使用）将策略直接附加到用户或将用户添加到用户组。按照《IAM 用户指南》中向用户添加权限（控制台）中的说明进行操作。

使用控制台来提高或重置吞吐量

以下过程介绍了如何使用 Systems Manager 控制台增加 Parameter Store 每秒可以为当前 AWS 账户和 AWS 区域处理的事务数。其中还显示了如果不再需要提高吞吐量，或者不想再产生费用，要如何恢复为标准设置。

使用控制台来提高或重置 Parameter Store 吞吐量

> **Tip**
>
> 如果尚未创建参数，则可以使用 AWS Command Line Interface (AWS CLI) 或 AWS Tools for Windows PowerShell 提高吞吐量。有关更多信息，请参阅 使用 AWS CLI 来提高或重置吞吐量 和 提高或重置吞吐量（PowerShell）。

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。

2. 在导航窗格中，请选择 Parameter Store。
3. 选择设置选项卡。
4. 要提高吞吐量，请选择设置限制。

   –或者–

   要恢复到默认限制，请选择重置限制。
5. 如果提高限制，请执行以下操作：

   - 选中我接受更改此设置会在我的 AWS 账户 中产生费用复选框。
   - 选择 Set limit (设置限制)。

   –或者–

   如果将限制重置为默认值，请执行以下操作：

   - 选中我接受重置为默认吞吐量限制会导致 Parameter Store 每秒处理的事务减少复选框。
   - 选择重置限制。

使用 AWS CLI 来提高或重置吞吐量

以下过程介绍了如何使用 AWS CLI 增加 Parameter Store 每秒可以为当前 AWS 账户和 AWS 区域处理的事务数。您也可以选择恢复为默认限制。

使用 AWS CLI 增加 Parameter Store 吞吐量

1. 打开 AWS CLI 并运行以下命令，以增加 Parameter Store 在当前 AWS 账户和 AWS 区域每秒可以处理的事务。

   ```
   aws ssm update-service-setting --setting-id arn:aws:ssm:region:account-
   id:servicesetting/ssm/parameter-store/high-throughput-enabled --setting-value true
   ```

   如果此命令成功，则无任何输出。
2. 运行以下命令以查看当前 AWS 账户和 AWS 区域中 Parameter Store 当前的吞吐量服务设置。

   ```
   aws ssm get-service-setting --setting-id arn:aws:ssm:region:account-
   id:servicesetting/ssm/parameter-store/high-throughput-enabled
   ```

系统返回类似于下文的信息：

```
{
    "ServiceSetting": {
        "SettingId": "/ssm/parameter-store/high-throughput-enabled",
        "SettingValue": "true",
        "LastModifiedDate": 1556551683.923,
        "LastModifiedUser": "arn:aws:sts::123456789012:assumed-role/Administrator/
Jasper",
        "ARN": "arn:aws:ssm:us-east-2:123456789012:servicesetting/ssm/parameter-
store/high-throughput-enabled",
        "Status": "Customized"
    }
}
```

如果您不再需要增加吞吐量，或者如果您不想再产生费用，您可以恢复为标准设置。要还原您的设置，请运行以下命令。

```
aws ssm reset-service-setting --setting-id arn:aws:ssm:region:account-
id:servicesetting/ssm/parameter-store/high-throughput-enabled
```

```
{
    "ServiceSetting": {
        "SettingId": "/ssm/parameter-store/high-throughput-enabled",
        "SettingValue": "false",
        "LastModifiedDate": 1555532818.578,
        "LastModifiedUser": "System",
        "ARN": "arn:aws:ssm:us-east-2:123456789012:servicesetting/ssm/parameter-store/
high-throughput-enabled",
        "Status": "Default"
    }
}
```

提高或重置吞吐量（PowerShell）

以下过程介绍了如何使用 Tools for Windows PowerShell 增加 Parameter Store 每秒可为当前 AWS 账户 和 AWS 区域 处理的事务数。您也可以选择恢复为默认限制。

使用 PowerShell 增加 Parameter Store 吞吐量

1. 使用 AWS Tools for PowerShell (Tools for PowerShell) 提高当前 AWS 账户和 AWS 区域的 Parameter Store 吞吐量。

   ```
   Update-SSMServiceSetting -SettingId "arn:aws:ssm:region:account-id:servicesetting/
   ssm/parameter-store/high-throughput-enabled" -SettingValue "true" -Region region
   ```

   如果此命令成功，则无任何输出。

2. 运行以下命令以查看当前 AWS 账户和 AWS 区域中 Parameter Store 当前的吞吐量服务设置。

   ```
   Get-SSMServiceSetting -SettingId "arn:aws:ssm:region:account-id:servicesetting/ssm/
   parameter-store/high-throughput-enabled" -Region region
   ```

   系统将返回与以下类似的信息：

   ```
   ARN              : arn:aws:ssm:us-east-2:123456789012:servicesetting/ssm/parameter-
   store/high-throughput-enabled
   LastModifiedDate : 4/29/2019 3:35:44 PM
   LastModifiedUser : arn:aws:sts::123456789012:assumed-role/Administrator/Jasper
   SettingId        : /ssm/parameter-store/high-throughput-enabled
   SettingValue     : true
   Status           : Customized
   ```

如果您不再需要增加吞吐量，或者如果您不想再产生费用，您可以恢复为标准设置。要还原您的设置，请运行以下命令。

```
Reset-SSMServiceSetting -SettingId "arn:aws:ssm:region:account-id:servicesetting/ssm/
parameter-store/high-throughput-enabled" -Region region
```

系统返回类似于下文的信息：

```
ARN              : arn:aws:ssm:us-east-2:123456789012:servicesetting/ssm/parameter-
store/high-throughput-enabled
LastModifiedDate : 4/17/2019 8:26:58 PM
LastModifiedUser : System
SettingId        : /ssm/parameter-store/high-throughput-enabled
SettingValue     : false
Status           : Default
```

基于 Parameter Store 事件设置通知或触发操作

本部分中的主题介绍了如何使用 Amazon EventBridge 和 Amazon Simple Notification Service (Amazon SNS) 通知您有关 AWS Systems Manager 参数的更改。您可以创建 Eventbridge 规则，以在创建、更新或删除参数或参数标签版本的情况下通知您。尽最大努力发出事件。您可以收到与参数策略相关的更改或状态通知，例如，参数过期、将过期或者在指定时间段内未更改。

> **Note**
>
> 参数策略可用于使用高级参数层的参数。需支付费用。有关更多信息，请参阅 在 Parameter Store 中分配参数策略 和 管理参数层。

本部分中的主题还介绍了如何针对特定参数事件在目标上启动其他操作。例如，您可以运行 AWS Lambda 函数来在参数过期或被删除后自动重新创建参数。您可以设置一个通知，以在数据库密码更新时调用 Lambda 函数。Lambda 函数可以强制您的数据库连接重置或使用新密码重新连接。Eventbridge 还支持在许多其他 AWS 服务中运行 Run Command 命令、自动化执行和操作。Run Command 和 Automation 都是 AWS Systems Manager 中的工具。有关更多信息，请参阅 Amazon EventBridge 用户指南。

开始前的准备工作

创建您需要的任何资源，以便为您创建的规则指定目标操作。例如，如果您创建的规则是关于发送通知，首先创建 Amazon SNS 主题。有关更多信息，请参阅 Amazon Simple Notification Service 开发人员指南中的 Amazon SNS 入门。

为参数和参数策略配置 Eventbridge 规则

本主题说明了以下内容：

- 如何创建将根据您的 AWS 账户中一个或多个参数发生的事件调用某个目标的 Eventbridge 规则。
- 如何创建将根据您的 AWS 账户中一个或多个参数策略发生的事件调用目标的 Eventbridge 规则。创建高级参数时，您需要指定参数到期时间、在参数到期前多长时间接收通知以及在参数保持不变时等待多长时间后发送通知。您可以按照以下过程为这些事件设置通知。有关更多信息，请参阅在 Parameter Store 中分配参数策略 和管理参数层。

为某个 Systems Manager 参数或参数策略配置 EventBridge 规则

1. 访问 https://console.aws.amazon.com/events/，打开 Amazon EventBridge 控制台。

2. 在导航窗格中，选择 Rules (规则)，然后选择 Create rule (创建规则)。

   –或者–

   如果 EventBridge 主页首先打开，请选择 Create rule (创建规则)。

3. 为规则输入名称和描述。

   规则不能与同一区域中的另一个规则和同一事件总线上的名称相同。

4. 对于 Event bus（事件总线），请选择要与此规则关联的事件总线。如果您希望此规则根据来自您自己的 AWS 账户的匹配事件启动，请选择 defaul（默认）。当您账户中的某个 AWS 服务发出一个事件时，它始终会发送到您账户的默认事件总线。

5. 对于 Rule type（规则类型），保留默认值 Rule with an event pattern（具有事件模式的规则）。

6. 选择下一步。

7. 对于事件源，选中默认值 AWS 事件或 EventBridge 合作伙伴事件。您可以跳过 Sample event（示例事件）部分。

8. 对于 Event pattern（事件模式），执行以下操作：

   - 选择 Custom patterns (JSON editor) [自定义模式（JSON 编辑器）]。
   - 对于 Event pattern（事件模式），根据您是要为参数还是为参数策略创建规则，将下面的任意一项内容粘贴到对话框中：

     Parameter

```
{
    "source": [
        "aws.ssm"
    ],
    "detail-type": [
        "Parameter Store Change"
    ],
    "detail": {
        "name": [
            "parameter-1-name",
            "/parameter-2-name/level-2",
            "/parameter-3-name/level-2/level-3"
        ],
        "operation": [
            "Create",
            "Update",
            "Delete",
```

```
            "LabelParameterVersion"
        ]
    }
}
```

Parameter policy

```
{
    "source": [
        "aws.ssm"
    ],
    "detail-type": [
        "Parameter Store Policy Action"
    ],
    "detail": {
        "parameter-name": [
            "parameter-1-name",
            "/parameter-2-name/level-2",
            "/parameter-3-name/level-2/level-3"
        ],
        "policy-type": [
            "Expiration",
            "ExpirationNotification",
            "NoChangeNotification"
        ]
    }
}
```

- 修改要对其执行操作的参数和操作的内容，如下面的示例所示。

  Parameter

  在此例中，在名为 /Oncall 和 /Project/Teamlead 的参数更新时将执行一项操作：

```
{
    "source": [
        "aws.ssm"
    ],
    "detail-type": [
        "Parameter Store Change"
    ],
    "detail": {
        "name": [
            "/Oncall",
```

```
            "/Project/Teamlead"
        ],
        "operation": [
            "Update"
        ]
    }
}
```

Parameter policy

在此例中，每当名为 /OncallDuties 的参数过期且被删除时都将执行一项操作：

```
{
    "source": [
        "aws.ssm"
    ],
    "detail-type": [
        "Parameter Store Policy Action"
    ],
    "detail": {
        "parameter-name": [
            "/OncallDuties"
        ],
        "policy-type": [
            "Expiration"
        ]
    }
}
```

9. 选择下一步。
10. 对于 Target 1（目标 1），选择一个目标类型和支持的资源。例如，如果选择 SNS topic (SNS 主题)，为 Topic (主题) 选择一个选项。如果您选择 CodePipeline，则对于 Pipeline ARN（管道 ARN）输入一个管道 ARN。根据需要提供其他配置值。

> **Tip**
>
> 如果您需要为此规则添加其他目标，请选择 Add another target（添加其他目标）。

11. 选择下一步。
12. （可选）为规则输入一个或多个标签。有关更多信息，请参阅《Amazon EventBridge 用户指南》中的 Amazon EventBridge 标签。

13. 选择下一步。
14. 选择创建规则。

更多信息

- 使用参数标签以实现跨环境轻松更新配置
- 《Amazon EventBridge 用户指南》中的 教程：使用 EventBridge 将事件转发到 AWS Systems ManagerRun Command
- Amazon EventBridge 用户指南中的教程：将 AWS Systems Manager 自动化设置为 EventBridge 目标

使用 Parameter Store

本部分介绍如何组织和创建标签参数以及如何创建不同版本的参数。

您可以使用 AWS Systems Manager 控制台、AWS Command Line Interface (AWS CLI)、AWS Tools for PowerShell 和 AWS SDK 来创建和使用参数。有关参数的更多信息，请参阅 什么是参数？。

主题

- 在 Systems Manager 中创建 Parameter Store 参数
- 在 Systems Manager 中搜索 Parameter Store 参数
- 在 Parameter Store 中分配参数策略
- 处理 Parameter Store 中的参数层次结构
- 防止访问 Parameter Store API 操作
- 处理 Parameter Store 中的参数标签
- 处理 Parameter Store 中的参数版本
- 使用 Parameter Store 中的共享参数
- 使用 Run Command 命令处理 Parameter Store 中的参数
- 在 Parameter Store 中为 Amazon Machine Image ID 使用本机参数支持
- 从 Parameter Store 中删除参数

在 Systems Manager 中创建 Parameter Store 参数

以下主题中的信息可帮助您使用 AWS Systems Manager 控制台、AWS Command Line Interface (AWS CLI) 或 AWS Tools for Windows PowerShell (Tools for Windows PowerShell) 创建 Systems Manager 参数。

本部分演示了如何在测试环境中使用 Parameter Store 创建、存储和运行参数。其中还演示了如何将 Parameter Store 与其他 Systems Manager 工具和 AWS 服务结合使用。有关更多信息，请参阅 什么是参数？

了解参数名称的要求和约束

使用本主题中的信息可在您创建参数时帮助您为参数名称指定有效值。

此信息对《AWS Systems Manager API 参考》中的主题 PutParameter（设置参数） 的详细信息进行了补充，此外，还提供了有关 AllowedPattern、Description、KeyId、Overwrite、Type 和 Value 数值的信息。

参数名称的要求和约束包括：

- 区分大小写：参数名称区分大小写。
- 空格：参数名称不能包含空格。
- 有效字符：参数名称只能包含以下符号和字母：`a-zA-Z0-9_.-`

  此外，斜杠字符 (/) 用于描述参数名称中的层次结构。例如：`/Dev/Production/East/Project-ABC/MyParameter`
- 有效 AMI 格式：选择 `aws:ec2:image` 作为 `String` 参数的数据类型时，您所输入的 ID 必须验证是否符合 AMI ID 格式 `ami-12345abcdeEXAMPLE`。
- 完全限定：在层次结构中创建或引用参数时，需要包含前导正斜杠字符 (/)。在引用属于层次结构一部分的参数时，需要指定包括初始斜杠 (/) 在内的整个层次结构路径。
  - 完全限定的参数名称：`MyParameter1`、`/MyParameter2`、`/Dev/Production/East/Project-ABC/MyParameter`
  - 未完全限定的参数名称：`MyParameter3/L1`
- 长度：您指定的参数名称最大长度为 1011 个字符。这 1011 个字符数包括所指定名称之前的 ARN 中的字符，例如 `arn:aws:ssm:us-east-2:111122223333:parameter/` 中的 45 个字符。
- 前缀：参数名称不得以“aws”或“ssm”（不区分大小写）作为前缀。例如，尝试使用以下名称创建参数会失败，并引发异常：

- awsTestParameter
- SSM-testparameter
- /aws/testparam1

> **Note**
>
> 在 SSM 文档、命令或脚本中指定参数时，应在句法中包含 ssm。例如，{{ssm:*parameter-name*}} 和 {{ ssm:*parameter-name* }}，比如 {{ssm:MyParameter}} 和 {{ ssm:MyParameter }}.

- 唯一性：参数名称必须在 AWS 区域中是唯一的。例如，Systems Manager 将以下参数视为不同的参数（如果它们存在于同一区域中）：
  - /Test/TestParam1
  - /TestParam1

  以下示例也是唯一的：
  - /Test/TestParam1/Logpath1
  - /Test/TestParam1

  不过，以下示例（如果在同一区域中）不是唯一的：
  - /TestParam1
  - TestParam1
- 层次结构深度：如果指定参数层次结构，则层次结构可以具有最多十五个级别的深度。可以在该层次结构的任何级别定义参数。以下两个示例的结构都是有效的：
  - /Level-1/L2/L3/L4/L5/L6/L7/L8/L9/L10/L11/L12/L13/L14/parameter-name
  - parameter-name

  尝试创建以下参数将失败并引发 HierarchyLevelLimitExceededException 异常：
  - /Level-1/L2/L3/L4/L5/L6/L7/L8/L9/L10/L11/L12/L13/L14/L15/L16/parameter-name

> **Important**
>
> 如果用户有权访问某个路径，则该用户可以访问该路径的所有级别。例如，如果某个用户有权访问路径 /a，则该用户也可以访问 /a/b。即使用户在 AWS Identity and Access

Management (IAM) 中已被显式拒绝访问参数 /a/b，他们仍能够以递归方式为 /a 调用 GetParametersByPath API 操作并查看 /a/b。

主题

- 使用控制台来创建 Parameter Store 参数
- 使用 AWS CLI 来创建 Parameter Store 参数
- 使用 Tools for Windows PowerShell 来创建 Parameter Store 参数

使用控制台来创建 Parameter Store 参数

您可以使用 AWS Systems Manager 控制台来创建并运行 String、StringList 和 SecureString 参数类型。删除参数后，至少等待 30 秒才能创建具有相同名称的参数。

> **Note**
>
> 参数只在创建它的 AWS 区域 可用。

以下过程将指导您完成在 Parameter Store 控制台中创建参数的过程。您可以从控制台创建 String、StringList和 SecureString 参数类型。

创建参数

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Parameter Store。
3. 选择创建参数。
4. 在 Name (名称) 框中，输入层次结构和名称。例如，输入 **/Test/helloWorld**。

   有关参数层次结构的更多信息，请参阅 处理 Parameter Store 中的参数层次结构。
5. 在 Description 框中，键入用于将此参数标识为测试参数的描述。
6. 对于 Parameter tier (参数层)，选择 Standard (标准) 或 Advanced (高级)。有关高级参数的更多信息，请参阅 管理参数层。
7. 对于 Type，选择 String、StringList 或 SecureString。

- 如果选择 String，则会显示 Data type (数据类型) 字段。如果要创建一个参数来保存 Amazon Machine Image (AMI) 的资源 ID，请选择 `aws:ec2:image`。否则，将保持选择默认 `text`。
- 如果选择 SecureString，则会显示 KMS Key ID (KMS 密钥 ID) 字段。如果您没有提供 AWS Key Management Service AWS KMS key ID、AWS KMS key Amazon Resource Name (ARN)、别名或别名 ARN，则系统将使用 `alias/aws/ssm`，这是 Systems Manager 的 AWS 托管式密钥。如果您不想使用此密钥，则可以使用客户托管密钥。有关 AWS 托管式密钥 和客户管理型密钥的更多信息，请参阅 AWS Key Management Service 开发人员指南中的 AWS Key Management Service 概念。有关 Parameter Store 和 AWS KMS 加密的更多信息，请参阅 AWS Systems ManagerParameter Store 如何使用 AWS KMS。

  > **Important**
  >
  > Parameter Store 仅支持对称加密 KMS 密钥。不能使用非对称加密 KMS 密钥来加密您的参数。要获取确定 KMS 密钥是对称还是非对称密钥的帮助，请参阅 AWS Key Management Service 开发人员指南中的识别对称 KMS 密钥和非对称 KMS 密钥。

- 在控制台中使用具有客户托管密钥别名或别名 ARN 的 `SecureString` 参数创建 `key-id` 参数时，需要在别名前面指定前缀 `alias/`。以下是一个 ARN 示例。

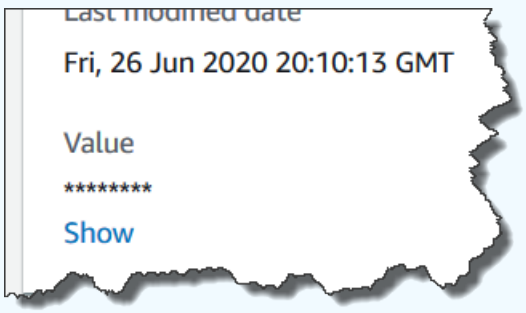
  ```
  arn:aws:kms:us-east-2:123456789012:alias/abcd1234-ab12-cd34-ef56-abcdeEXAMPLE
  ```

  以下是一个别名示例。

  ```
  alias/MyAliasName
  ```

8. 在 Value (值) 框中，键入一个值。例如，键入 **`This is my first parameter`** 或 **`ami-0dbf5ea29aEXAMPLE`**。

   > **Note**
   >
   > 参数不能被引用或嵌套在其他参数的值中。参数值中不能包含 `{{}}` 或 `{{ssm:parameter-name}}`。
   >
   > 如果选择 SecureString，则稍后在参数 Overview 选项卡上查看时，参数值默认情况下会被遮蔽 (“******”) ，如下图所示。选择 Show (显示) 以显示参数值。



9. （可选）在 Tags (标签) 区域，将一个或多个标签键/值对应用到参数。

   标签是您分配给资源的可选元数据。标签可让您按不同的方式（如用途、拥有者或环境）对资源进行分类。例如，您可能希望标记一个 Systems Manager 参数来标识其所应用到的资源的类型、环境或参数引用的配置数据的类型。在这种情况下，您可以指定以下键/值对：

   - `Key=Resource,Value=S3bucket`
   - `Key=OS,Value=Windows`
   - `Key=ParameterType,Value=LicenseKey`

10. 选择创建参数。
11. 在参数列表中，请选择刚才创建的参数的名称。验证 Overview 选项卡上的详细信息。如果您创建了 `SecureString` 参数，请选择 Show 查看未加密值。

> **Note**
>
> 您不能将高级参数更改为标准参数。如果您不再需要高级参数，或者如果您不想再为高级参数支付费用，请删除它并将它重新创建为新的标准参数。

使用 AWS CLI 来创建 Parameter Store 参数

您可以使用 AWS Command Line Interface (AWS CLI) 创建 `String`、`StringList` 和 `SecureString` 参数类型。删除参数后，至少等待 30 秒才能创建具有相同名称的参数。

参数不能被引用或嵌套在其他参数的值中。参数值中不能包含 `{{}}` 或 `{{ssm:parameter-name}}`。

> **Note**
>
> 参数只在创建它的 AWS 区域 可用。

主题

- 使用 AWS CLI 来创建 String 参数
- 使用 AWS CLI 来创建 StringList 参数
- 使用 AWS CLI 来创建 SecureString 参数
- 使用 AWS CLI 来创建多行参数

使用 AWS CLI 来创建 **`String`** 参数

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关更多信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 运行以下命令，创建 `String` 类型参数。将每个*#######*替换为您自己的信息。

   Linux & macOS

   ```
   aws ssm put-parameter \
       --name "parameter-name" \
       --value "parameter-value" \
       --type String \
       --tags "Key=tag-key,Value=tag-value"
   ```

   Windows

   ```
   aws ssm put-parameter ^
       --name "parameter-name" ^
       --value "parameter-value" ^
       --type String ^
       --tags "Key=tag-key,Value=tag-value"
   ```

   - 或者 -

   运行以下命令，创建一个包含 Amazon Machine Image (AMI) ID 作为参数值的参数。

Linux & macOS

```
aws ssm put-parameter \
    --name "parameter-name" \
    --value "an-AMI-id" \
    --type String \
    --data-type "aws:ec2:image" \
    --tags "Key=tag-key,Value=tag-value"
```

Windows

```
aws ssm put-parameter ^
    --name "parameter-name" ^
    --value "an-AMI-id" ^
    --type String ^
    --data-type "aws:ec2:image" ^
    --tags "Key=tag-key,Value=tag-value"
```

`--name` 选项支持层次结构。有关层次结构的更多信息，请参阅 处理 Parameter Store 中的参数层次结构。

仅当要创建包含 AMI ID 的参数时，才必须指定 `--data-type` 选项。它会验证您输入的参数值是否为格式正确的 Amazon Elastic Compute Cloud (Amazon EC2) AMI ID。对于所有其他参数，默认数据类型为 `text`，并且可以选择指定一个值。有关更多信息，请参阅 在 Parameter Store 中为 Amazon Machine Image ID 使用本机参数支持。

> **Important**
>
> 如果成功，则该命令返回参数的版本号。例外：如果您已将 `aws:ec2:image` 指定为数据类型，则响应中的新版本号并不意味着参数值已经过验证。有关更多信息，请参阅 在 Parameter Store 中为 Amazon Machine Image ID 使用本机参数支持。

以下示例将两个键/值对标签添加到参数。

Linux & macOS

```
aws ssm put-parameter \
```

```
    --name parameter-name \
    --value "parameter-value" \
    --type "String" \
    --tags '[{"Key":"Region","Value":"East"},{"Key":"Environment",
 "Value":"Production"}]'
```

Windows

```
aws ssm put-parameter ^
    --name parameter-name ^
    --value "parameter-value" ^
    --type "String" ^
    --tags [{\"Key\":\"Region1\",\"Value\":\"East1\"},{\"Key\":\"Environment1\",
\"Value\":\"Production1\"}]
```

以下是在名称中使用参数层次结构创建明文 String 参数的示例。它会返回参数的版本号。有关参数层次结构的更多信息，请参阅 处理 Parameter Store 中的参数层次结构。

Linux & macOS

不在层次结构中的参数

```
aws ssm put-parameter \
    --name "golden-ami" \
    --type "String" \
    --value "ami-12345abcdeEXAMPLE"
```

层次结构中的参数

```
aws ssm put-parameter \
    --name "/amis/linux/golden-ami" \
    --type "String" \
    --value "ami-12345abcdeEXAMPLE"
```

Windows

不在层次结构中的参数

```
aws ssm put-parameter ^
    --name "golden-ami" ^
```

```
    --type "String" ^
    --value "ami-12345abcdeEXAMPLE"
```

层次结构中的参数

```
aws ssm put-parameter ^
    --name "/amis/windows/golden-ami" ^
    --type "String" ^
    --value "ami-12345abcdeEXAMPLE"
```

3. 运行以下命令，查看最新参数值并验证新参数的详细信息。

```
aws ssm get-parameters --names "/Test/IAD/helloWorld"
```

系统将返回类似于以下内容的信息。

```
{
    "InvalidParameters": [],
    "Parameters": [
        {
            "Name": "/Test/IAD/helloWorld",
            "Type": "String",
            "Value": "My updated parameter value",
            "Version": 2,
            "LastModifiedDate": "2020-02-25T15:55:33.677000-08:00",
            "ARN": "arn:aws:ssm:us-east-2:123456789012:parameter/Test/IAD/
helloWorld"
        }
    ]
}
```

运行以下命令，更改参数值。它会返回参数的版本号。

```
aws ssm put-parameter --name "/Test/IAD/helloWorld" --value "My updated 1st parameter"
 --type String --overwrite
```

运行以下命令，查看参数值的历史记录。

```
aws ssm get-parameter-history --name "/Test/IAD/helloWorld"
```

运行以下命令，在命令中使用该参数。

```
aws ssm send-command --document-name "AWS-RunShellScript" --parameters '{"commands":
["echo {{ssm:/Test/IAD/helloWorld}}"]}' --targets "Key=instanceids,Values=instance-ids"
```

如果您只想检索参数值，请运行以下命令。

```
aws ssm get-parameter --name testDataTypeParameter --query "Parameter.Value"
```

如果您只想使用 get-parameters 检索参数值，请运行以下命令。

```
aws ssm get-parameters --names "testDataTypeParameter" --query "Parameters[*].Value"
```

运行以下命令，查看参数元数据。

```
aws ssm describe-parameters --filters "Key=Name,Values=/Test/IAD/helloWorld"
```

> **Note**
>
> 名称必须大写。

系统将返回类似于以下内容的信息。

```
{
    "Parameters": [
        {
            "Name": "helloworld",
            "Type": "String",
            "LastModifiedUser": "arn:aws:iam::123456789012:user/JohnDoe",
            "LastModifiedDate": 1494529763.156,
            "Version": 1,
            "Tier": "Standard",
            "Policies": []
        }
    ]
}
```

使用 AWS CLI 来创建 StringList 参数

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关更多信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 运行以下命令创建一个参数。将每个*#######*替换为您自己的信息。

   Linux & macOS

   ```
   aws ssm put-parameter \
       --name "parameter-name" \
       --value "a-comma-separated-list-of-values" \
       --type StringList \
       --tags "Key=tag-key,Value=tag-value"
   ```

   Windows

   ```
   aws ssm put-parameter ^
       --name "parameter-name" ^
       --value "a-comma-separated-list-of-values" ^
       --type StringList ^
       --tags "Key=tag-key,Value=tag-value"
   ```

   > **Note**
   >
   > 如果成功，则该命令返回参数的版本号。

   此示例将两个键/值对添加到参数。（根据本地计算机上的操作系统类型，运行以下命令之一。从本地 Windows 计算机运行的版本包含转义字符（“\”），您需要从命令行工具运行命令。）

   下面是一个使用参数层次结构的 `StringList` 示例。

   Linux & macOS

   ```
   aws ssm put-parameter \
       --name /IAD/ERP/Oracle/addUsers \
       --value "Milana,Mariana,Mark,Miguel" \
       --type StringList
   ```

Windows

```
aws ssm put-parameter ^
    --name /IAD/ERP/Oracle/addUsers ^
    --value "Milana,Mariana,Mark,Miguel" ^
    --type StringList
```

> **Note**
>
> StringList 中的项目必须用逗号 (,) 分隔。不能使用其他标点符号或特殊字符对列表中的项目进行转义。如果您有需要逗号的参数值，则使用 String 类型。

3. 运行 get-parameters 命令，验证该参数的详细信息。例如：

```
aws ssm get-parameters --name "/IAD/ERP/Oracle/addUsers"
```

使用 AWS CLI 来创建 SecureString 参数

使用以下过程创建 SecureString 参数。将每个#######替换为您自己的信息。

> **Important**
>
> 只会加密 SecureString 参数的值。不会加密参数名称、描述和其他属性。

> **Important**
>
> Parameter Store 仅支持对称加密 KMS 密钥。不能使用非对称加密 KMS 密钥来加密您的参数。要获取确定 KMS 密钥是对称还是非对称密钥的帮助，请参阅 AWS Key Management Service 开发人员指南中的识别对称 KMS 密钥和非对称 KMS 密钥。

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关更多信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 运行以下命令之一，创建使用 SecureString 数据类型的参数。

Linux & macOS

使用默认的 AWS 托管式密钥创建 **SecureString** 参数

```
aws ssm put-parameter \
    --name "parameter-name" \
    --value "parameter-value" \
    --type "SecureString"
```

创建使用客户托管密钥的 **SecureString** 参数

```
aws ssm put-parameter \
    --name "parameter-name" \
    --value "a-parameter-value, for example P@ssW%rd#1" \
    --type "SecureString"
    --tags "Key=tag-key,Value=tag-value"
```

创建使用自定义 AWS KMS 密钥的 **SecureString** 参数

```
aws ssm put-parameter \
    --name "parameter-name" \
    --value "a-parameter-value, for example P@ssW%rd#1" \
    --type "SecureString" \
    --key-id "your-account-ID/the-custom-AWS KMS-key" \
    --tags "Key=tag-key,Value=tag-value"
```

Windows

使用默认的 AWS 托管式密钥创建 **SecureString** 参数

```
aws ssm put-parameter ^
    --name "parameter-name" ^
    --value "parameter-value" ^
    --type "SecureString"
```

创建使用客户托管密钥的 **SecureString** 参数

```
aws ssm put-parameter ^
    --name "parameter-name" ^
    --value "a-parameter-value, for example P@ssW%rd#1" ^
```

```
    --type "SecureString" ^
    --tags "Key=tag-key,Value=tag-value"
```

创建使用自定义 AWS KMS 密钥的 **SecureString** 参数

```
aws ssm put-parameter ^
    --name "parameter-name" ^
    --value "a-parameter-value, for example P@ssW%rd#1" ^
    --type "SecureString" ^
    --key-id " ^
    --tags "Key=tag-key,Value=tag-value"account-ID/the-custom-AWS KMS-key"
```

如果您在自己的账户和区域中使用 AWS 托管式密钥创建 SecureString 参数，则无需提供 --key-id 参数的值。

> **Note**
>
> 要使用分配给您的 AWS 账户和 AWS 区域的 AWS KMS key，请从命令中删除 key-id 参数。有关 AWS KMS keys 的更多信息，请参阅 AWS Key Management Service 开发人员指南中的 AWS Key Management Service 概念。

要使用客户托管密钥而不是分配给您的账户的 AWS 托管式密钥，需要使用 --key-id 参数指定密钥。该参数支持以下 KMS 参数格式。

- 密钥 Amazon Resource Name (ARN) 示例：

  `arn:aws:kms:us-east-2:123456789012:key/key-id`

- 别名 ARN 示例:

  `arn:aws:kms:us-east-2:123456789012:alias/alias-name`

- 密钥 ID 示例：

  `12345678-1234-1234-1234-123456789012`

- 别名示例:

  `alias/MyAliasName`

您可以使用 AWS Management Console 或 AWS KMS API 创建客户托管密钥。以下 AWS CLI 命令在您的 AWS 账户的当前 AWS 区域创建客户托管密钥。

```
aws kms create-key
```

使用以下格式的命令，利用您刚刚创建的密钥创建 SecureString 参数。

以下示例对密码参数和 AWS KMS key 使用模糊名称 (3l3vat3131)。

Linux & macOS

```
aws ssm put-parameter \
    --name /Finance/Payroll/3l3vat3131 \
    --value "P@sSwW)rd" \
    --type SecureString \
    --key-id arn:aws:kms:us-
east-2:123456789012:key/1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d5e
```

Windows

```
aws ssm put-parameter ^
    --name /Finance/Payroll/3l3vat3131 ^
    --value "P@sSwW)rd" ^
    --type SecureString ^
    --key-id arn:aws:kms:us-
east-2:123456789012:key/1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d5e
```

3. 运行以下命令验证参数的详细信息。

如果您不指定 with-decryption 参数，或者如果您指定no-with-decryption 参数，命令会返回加密的 GUID。

Linux & macOS

```
aws ssm get-parameters \
    --name "the-parameter-name-you-specified" \
    --with-decryption
```

Windows

```
aws ssm get-parameters ^
    --name "the-parameter-name-you-specified" ^
    --with-decryption
```

4. 运行以下命令，查看参数元数据。

Linux & macOS

```
aws ssm describe-parameters \
    --filters "Key=Name,Values=the-name-that-you-specified"
```

Windows

```
aws ssm describe-parameters ^
    --filters "Key=Name,Values=the-name-that-you-specified"
```

5. 如果您使用的不是客户托管的 AWS KMS key，请运行以下命令更改参数值。

Linux & macOS

```
aws ssm put-parameter \
    --name "the-name-that-you-specified" \
    --value "a-new-parameter-value" \
    --type "SecureString" \
    --overwrite
```

Windows

```
aws ssm put-parameter ^
    --name "the-name-that-you-specified" ^
    --value "a-new-parameter-value" ^
    --type "SecureString" ^
    --overwrite
```

- 或者 -

如果您使用的是客户托管的 AWS KMS key，请运行以下命令之一更改参数值。

Linux & macOS

```
aws ssm put-parameter \
    --name "the-name-that-you-specified" \
    --value "a-new-parameter-value" \
    --type "SecureString" \
    --key-id "the-KMSkey-ID" \
    --overwrite
```

```
aws ssm put-parameter \
    --name "the-name-that-you-specified" \
    --value "a-new-parameter-value" \
    --type "SecureString" \
    --key-id "account-alias/the-KMSkey-ID" \
    --overwrite
```

Windows

```
aws ssm put-parameter ^
    --name "the-name-that-you-specified" ^
    --value "a-new-parameter-value" ^
    --type "SecureString" ^
    --key-id "the-KMSkey-ID" ^
    --overwrite
```

```
aws ssm put-parameter ^
    --name "the-name-that-you-specified" ^
    --value "a-new-parameter-value" ^
    --type "SecureString" ^
    --key-id "account-alias/the-KMSkey-ID" ^
    --overwrite
```

6. 运行以下命令，查看最新的参数值。

Linux & macOS

```
aws ssm get-parameters \
    --name "the-name-that-you-specified" \
    --with-decryption
```

Windows

```
aws ssm get-parameters ^
    --name "the-name-that-you-specified" ^
    --with-decryption
```

7. 运行以下命令，查看参数值的历史记录。

Linux & macOS

```
aws ssm get-parameter-history \
    --name "the-name-that-you-specified"
```

Windows

```
aws ssm get-parameter-history ^
    --name "the-name-that-you-specified"
```

> **Note**
>
> 您可以使用加密值手动创建参数。在本例中，由于值已经加密，因此您无需选择 SecureString 参数类型。如果您选择 SecureString，将对您的参数进行双重加密。

默认情况下，所有 SecureString 值均显示为密码文本。要解密 SecureString 值，用户必须有权调用 AWS KMS Decrypt API 操作。有关配置 AWS KMS 访问控制的信息，请参阅 AWS Key Management Service 开发人员指南中的 AWS KMS 的身份验证和访问控制。

> **Important**
>
> 如果更改用于加密参数的 KMS 密钥的别名，则还必须更新该参数用于引用 AWS KMS 的密钥别名。这仅适用于 KMS 密钥别名；除非删除整个密钥，否则别名附加到的密钥 ID 将保持不变。

使用 AWS CLI 来创建多行参数

您可以使用 AWS CLI 来创建带换行符的参数。使用换行符将文本分成更长的参数值，使其易于阅读，或者，例如，更新网页的多段参数内容。您可以将内容包含在 JSON 文件中并使用 `--cli-input-json` 选项，使用诸如 `\n` 等换行符，如以下示例所示。

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关更多信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 运行以下命令，创建多行参数。

   Linux & macOS

   ```
   aws ssm put-parameter \
       --name "MultiLineParameter" \
       --type String \
       --cli-input-json file://MultiLineParameter.json
   ```

   Windows

   ```
   aws ssm put-parameter ^
       --name "MultiLineParameter" ^
       --type String ^
       --cli-input-json file://MultiLineParameter.json
   ```

   以下示例显示 `MultiLineParameter.json` 文件的内容。

   ```
   {
       "Value": "<para>Paragraph One</para>\n<para>Paragraph Two</para>
   \n<para>Paragraph Three</para>"
   }
   ```

保存的参数值的存储方式如下。

```
<para>Paragraph One</para>
<para>Paragraph Two</para>
<para>Paragraph Three</para>
```

使用 Tools for Windows PowerShell 来创建 Parameter Store 参数

您可以使用 AWS Tools for Windows PowerShell 创建 `String`、`StringList` 和 `SecureString` 参数类型。删除参数后，至少等待 30 秒才能创建具有相同名称的参数。

参数不能被引用或嵌套在其他参数的值中。参数值中不能包含 `{{}}` 或 `{{ssm:parameter-name}}`。

> **Note**
>
> 参数只在创建它的 AWS 区域 可用。

主题

- 创建字符串参数 (Tools for Windows PowerShell)
- 创建 StringList 参数 (Tools for Windows PowerShell)
- 创建 SecureString 参数 (Tools for Windows PowerShell)

创建字符串参数 (Tools for Windows PowerShell)

1. 如果您尚未安装和配置 AWS Tools for PowerShell（适用于 Windows PowerShell 的工具），请执行这些操作。

   有关更多信息，请参阅安装 AWS Tools for PowerShell。
2. 运行以下命令，创建一个包含纯文本值的参数。将每个*#######*替换为您自己的信息。

   ```
   Write-SSMParameter `
       -Name "parameter-name" `
       -Value "parameter-value" `
       -Type "String"
   ```

   - 或者 -

   运行以下命令，创建一个包含 Amazon Machine Image (AMI) ID 作为参数值的参数。

   > **Note**
   >
   > 要创建带有标签的参数，请先创建 service.model.tag 作为变量。见下列。

```
$tag = New-Object Amazon.SimpleSystemsManagement.Model.Tag
$tag.Key = "tag-key"
$tag.Value = "tag-value"
```

```
Write-SSMParameter `
    -Name "parameter-name" `
    -Value "an-AMI-id" `
    -Type "String" `
    -DataType "aws:ec2:image" `
    -Tags $tag
```

仅当要创建包含 AMI ID 的参数时，才必须指定 `-DataType` 选项。对于所有其他参数，默认数据类型为 `text`。有关更多信息，请参阅 在 Parameter Store 中为 Amazon Machine Image ID 使用本机参数支持。

以下是使用参数层次结构的示例。

```
Write-SSMParameter `
    -Name "/IAD/Web/SQL/IPaddress" `
    -Value "99.99.99.999" `
    -Type "String" `
    -Tags $tag
```

3. 运行以下命令验证参数的详细信息。

```
(Get-SSMParameterValue -Name "the-parameter-name-you-specified").Parameters
```

创建 StringList 参数 (Tools for Windows PowerShell)

1. 如果您尚未安装和配置 AWS Tools for PowerShell（适用于 Windows PowerShell 的工具），请执行这些操作。

   有关更多信息，请参阅安装 AWS Tools for PowerShell。

2. 运行以下命令，创建 StringList 参数。将每个*#######*替换为您自己的信息。

> **Note**
>
> 要创建带有标签的参数，请先创建 service.model.tag 作为变量。见下列。
>
> ```
> $tag = New-Object Amazon.SimpleSystemsManagement.Model.Tag
> $tag.Key = "tag-key"
> $tag.Value = "tag-value"
> ```

```
Write-SSMParameter `
    -Name "parameter-name" `
    -Value "a-comma-separated-list-of-values" `
    -Type "StringList" `
    -Tags $tag
```

如果成功，则该命令返回参数的版本号。

见下列。

```
Write-SSMParameter `
    -Name "stringlist-parameter" `
    -Value "Milana,Mariana,Mark,Miguel" `
    -Type "StringList" `
    -Tags $tag
```

> **Note**
>
> StringList 中的项目必须用逗号 (,) 分隔。不能使用其他标点符号或特殊字符对列表中的项目进行转义。如果您有需要逗号的参数值，则使用 String 类型。

3. 运行以下命令验证参数的详细信息。

```
(Get-SSMParameterValue -Name "the-parameter-name-you-specified").Parameters
```

创建 SecureString 参数 (Tools for Windows PowerShell)

在创建 SecureString 参数前，请阅读关于此类型参数的要求。有关更多信息，请参阅 使用 AWS CLI 来创建 SecureString 参数。

> **Important**
>
> 只会加密 SecureString 参数的值。不会加密参数名称、描述和其他属性。

> **Important**
>
> Parameter Store 仅支持对称加密 KMS 密钥。不能使用非对称加密 KMS 密钥来加密您的参数。要获取确定 KMS 密钥是对称还是非对称密钥的帮助，请参阅 AWS Key Management Service 开发人员指南中的识别对称 KMS 密钥和非对称 KMS 密钥。

1. 如果您尚未安装和配置 AWS Tools for PowerShell（适用于 Windows PowerShell 的工具），请执行这些操作。

   有关更多信息，请参阅安装 AWS Tools for PowerShell。

2. 运行以下命令创建一个参数。将每个*#######*替换为您自己的信息。

   > **Note**
   >
   > 要创建带有标签的参数，请先创建 service.model.tag 作为变量。见下列。
   >
   > ```
   > $tag = New-Object Amazon.SimpleSystemsManagement.Model.Tag
   > $tag.Key = "tag-key"
   > $tag.Value = "tag-value"
   > ```

   ```
   Write-SSMParameter `
       -Name "parameter-name" `
       -Value "parameter-value" `
       -Type "SecureString"  `
       -KeyId "an AWS KMS key ID, an AWS KMS key ARN, an alias name, or an alias ARN"
    `
   ```

```
    -Tags $tag
```

如果成功，则该命令返回参数的版本号。

> **Note**
>
> 要使用分配给您的账户的 AWS 托管式密钥，请移除命令中的 `-KeyId` 参数。

以下示例使用了一个密码参数的模糊名称（3l3vat3131）和一个 AWS 托管式密钥。

```
Write-SSMParameter `
    -Name "/Finance/Payroll/3l3vat3131" `
    -Value "P@sSwW)rd" `
    -Type "SecureString"`
    -Tags $tag
```

3. 运行以下命令验证参数的详细信息。

```
(Get-SSMParameterValue -Name "the-parameter-name-you-specified" -WithDecryption
 $true).Parameters
```

默认情况下，所有 `SecureString` 值均显示为密码文本。要解密 `SecureString` 值，用户必须有权调用 AWS KMS Decrypt API 操作。有关配置 AWS KMS 访问控制的信息，请参阅 AWS Key Management Service 开发人员指南中的 AWS KMS 的身份验证和访问控制。

> **Important**
>
> 如果更改用于加密参数的 KMS 密钥的别名，则还必须更新该参数用于引用 AWS KMS 的密钥别名。这仅适用于 KMS 密钥别名；除非删除整个密钥，否则别名附加到的密钥 ID 将保持不变。

在 Systems Manager 中搜索 Parameter Store 参数

在您的账户中具有大量参数时，可能很难一次找到有关单个或几个参数的信息。在这种情况下，您可以使用筛选工具根据指定的搜索条件搜索需要查找信息的参数。您可以使用 AWS Systems Manager 控制台、AWS Command Line Interface (AWS CLI)、AWS Tools for PowerShell 或 DescribeParameters API 搜索参数。

主题

- 使用控制台来搜索参数
- 使用 AWS CLI 来搜索参数

使用控制台来搜索参数

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Parameter Store。
3. 在搜索框中进行选择，然后选择所需的搜索方式。例如，Type 或 Name。
4. 提供选定的搜索类型的信息。例如：
   - 如果按 Type (类型) 进行搜索，请从 String、StringList 或 SecureString 中进行选择。
   - 如果按 Name (名称) 进行搜索，请选择 contains、equals 或 begins-with，然后输入完整参数名称或其中的一部分。

   > **Note**
   >
   > 在控制台中，Name 的默认搜索类型为 contains。
5. 按 Enter 键。

将使用搜索结果更新参数列表。

> **Note**
>
> 您的搜索结果可能比第一页显示的结果还要多。使用参数列表主题处的右箭头 (>) (如有) 查看下一组结果。

使用 AWS CLI 来搜索参数

可以在 AWS CLI 中使用 describe-parameters 命令查看有关一个或多个参数的信息。

以下示例展示了多个选项，您可以使用这些选项查看有关您的 AWS 账户中的参数的信息。有关设置这些选项的更多信息，请参阅 AWS Command Line Interface 用户指南中的 describe-parameters。

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关更多信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 将以下命令中的示例值替换为反映在您的账户中创建的参数的值。

   Linux & macOS

   ```
   aws ssm describe-parameters \
       --parameter-filters "Key=Name,Values=MyParameterName"
   ```

   Windows

   ```
   aws ssm describe-parameters ^
       --parameter-filters "Key=Name,Values=MyParameterName"
   ```

   > **Note**
   >
   > 对于 `describe-parameters`，`Name` 的默认搜索类型为 `Equals`。在参数筛选条件中，指定 `"Key=Name,Values=MyParameterName"` 与指定 `"Key=Name,Option=Equals,Values=MyParameterName"` 相同。

```
aws ssm describe-parameters \
    --parameter-filters "Key=Name,Option=Contains,Values=Product"
```

```
aws ssm describe-parameters \
    --parameter-filters "Key=Type,Values=String"
```

```
aws ssm describe-parameters \
    --parameter-filters "Key=Path,Values=/Production/West"
```

```
aws ssm describe-parameters \
    --parameter-filters "Key=Tier,Values=Standard"
```

```
aws ssm describe-parameters \
    --parameter-filters "Key=tag:tag-key,Values=tag-value"
```

```
aws ssm describe-parameters \
    --parameter-filters "Key=KeyId,Values=key-id"
```

> **Note**
>
> 在最后一个示例中，*key-id* 表示 AWS Key Management Service (AWS KMS) 密钥的 ID，该密钥用于加密在您的账户中创建的 SecureString 参数。或者，您也可以输入 **alias/aws/ssm** 以使用您的账户的默认 AWS KMS 密钥。有关更多信息，请参阅 使用 AWS CLI 来创建 SecureString 参数。

如果成功，该命令将返回类似于以下内容的输出。

```
{
    "Parameters": [
        {
            "Name": "/Production/West/Manager",
            "Type": "String",
            "LastModifiedDate": 1573438580.703,
            "LastModifiedUser": "arn:aws:iam::111122223333:user/Mateo.Jackson",
            "Version": 1,
            "Tier": "Standard",
            "Policies": []
        },
        {
            "Name": "/Production/West/TeamLead",
            "Type": "String",
            "LastModifiedDate": 1572363610.175,
            "LastModifiedUser": "arn:aws:iam::111122223333:user/Mateo.Jackson",
            "Version": 1,
            "Tier": "Standard",
            "Policies": []
        },
        {
            "Name": "/Production/West/HR",
            "Type": "String",
            "LastModifiedDate": 1572363680.503,
            "LastModifiedUser": "arn:aws:iam::111122223333:user/Mateo.Jackson",
            "Version": 1,
            "Tier": "Standard",
```

```
            "Policies": []
        }
    ]
}
```

在 Parameter Store 中分配参数策略

参数策略让您可以将特定条件分配到参数（如到期日期或存活时间），通过这种方式来帮助您管理一组不断增加的参数。参数策略在强制您更新或删除 Parameter Store（AWS Systems Manager 中的一项工具）中存储的密码和配置数据时尤其有用。Parameter Store 提供以下策略类型：`Expiration`、`ExpirationNotification` 和 `NoChangeNotification`。

> **Note**
>
> 要实施密码轮换生命周期，请使用 AWS Secrets Manager。您可以使用 Secrets Manager 在数据库凭证、API 密钥和其他密钥的整个生命周期内对其进行轮换、管理和检索。有关更多信息，请参阅 AWS Secrets Manager 用户指南中的什么是AWS Secrets Manager？。

Parameter Store 通过使用异步的定期扫描强制实施参数策略。创建策略后，您无需执行其他操作来强制实施策略。Parameter Store 会根据您指定的条件独立执行策略定义的操作。

> **Note**
>
> 参数策略可用于使用高级参数层的参数。有关更多信息，请参阅 管理参数层。

参数策略是一个 JSON 数组，如下表所示。您可以在创建新的高级参数时分配策略，也可以通过更新参数应用策略。Parameter Store 支持以下类型的参数策略。

策略	详细信息	示例
过期	此策略将删除参数。您可以使用 `ISO_INSTANT` 格式或 `ISO_OFFSET_DATE_TIME` 格式指定特定日期和时间。要更改需要删除参数的时间，请更新策略。更新参数不会影响	`{ "Type": "Expiration", "Version": "1.0", "Attributes": {`

策略	详细信息	示例
	附加到参数的策略的到期日期或时间。当到达到期日期和时间时，Parameter Store 将删除参数。 **Note** 此示例使用 ISO_INSTANT 格式。您还可以使用 ISO_OFFSET_DATE_TIME 格式指定日期和时间。以下是一个示例：2019-11-01T22:13:48.87+10:30:00 。	"Timestamp": "2018-12-02T21:34:33.000Z" } }

ExpirationNotification	该策略在 Amazon EventBridge (EventBridge) 中启动一个事件，以通知您到期信息。使用此策略，您可以在到达到期时间之前收到通知（以天或小时为单位）。	{
		"Type": "ExpirationNotification",
		"Version": "1.0",
		"Attributes": {
		"Before": "15",
		"Unit": "Days"
		}
		}

策略	详细信息	示例
NoChangeNotification	如果参数在指定时间段内未被修改，该策略将在 EventBridge 中启动一个事件。例如，当在一段时间内需要更改密码时，此策略类型非常有用。 此策略通过读取参数的 `LastModifiedTime` 属性确定何时发送通知。如果您更改或编辑参数，系统会根据 `LastModifiedTime` 的新值重置通知时间段。	`{ "Type": "NoChangeNotification", "Version": "1.0", "Attributes": { "After": "20", "Unit": "Days" } }`

您可向参数分配多个策略。例如，您可以分配 `Expiration` 和 `ExpirationNotification` 策略，让系统启动 EventBridge 事件以通知您即将删除参数。您最多可向参数分配是十 (10) 个策略。

以下示例展示了一个 PutParameter API 请求的请求语法，该请求将四个策略分配到名为 `ProdDB3` 的新 `SecureString` 参数。

```
{
    "Name": "ProdDB3",
    "Description": "Parameter with policies",
    "Value": "P@ssW*rd21",
    "Type": "SecureString",
    "Overwrite": "True",
    "Policies": [
        {
            "Type": "Expiration",
            "Version": "1.0",
            "Attributes": {
                "Timestamp": "2018-12-02T21:34:33.000Z"
            }
        },
        {
            "Type": "ExpirationNotification",
            "Version": "1.0",
            "Attributes": {
                "Before": "30",
```

```
                "Unit": "Days"
            }
        },
        {
            "Type": "ExpirationNotification",
            "Version": "1.0",
            "Attributes": {
                "Before": "15",
                "Unit": "Days"
            }
        },
        {
            "Type": "NoChangeNotification",
            "Version": "1.0",
            "Attributes": {
                "After": "20",
                "Unit": "Days"
            }
        }
    ]
}
```

将策略添加到现有参数

本部分包括有关如何使用 AWS Systems Manager 控制台、AWS Command Line Interface (AWS CLI) 和 AWS Tools for Windows PowerShell 将策略添加到现有参数的信息。有关如何创建包含策略的新参数的信息，请参阅 在 Systems Manager 中创建 Parameter Store 参数。

主题

- 使用控制台将策略添加到现有参数
- 使用 AWS CLI 将策略添加到现有参数
- 将策略添加到现有参数 (Tools for Windows PowerShell)

使用控制台将策略添加到现有参数

按照以下过程，使用 Systems Manager 控制台将策略添加到现有参数。

将策略添加到现有参数

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。

2. 在导航窗格中，请选择 Parameter Store。
3. 选择要更新以包含策略的参数旁边的选项，然后选择编辑。
4. 选择 Advanced (高级)。
5. （可选）在 Parameter policies (参数策略) 部分中，选择 Enabled (已启用)。您可以为此参数指定到期日期以及一个或多个通知策略。
6. 选择保存更改。

> **Important**
>
> - Parameter Store 会在参数上保留策略，直至您使用新策略覆盖这些策略或删除这些策略。
> - 要从现有参数中删除所有策略，请编辑参数并使用括号和大括号应用空策略，如下所示：`[{}]`
> - 如果您向已有策略的参数添加新策略，Systems Manager 将覆盖附加到该参数的策略。现有策略将被删除。如果您希望向已有一个或多个策略的参数添加新策略，请复制并粘贴原始策略，键入新策略，然后保存所做的更改。

使用 AWS CLI 将策略添加到现有参数

使用以下过程，使用 AWS CLI 将策略添加到现有参数。

将策略添加到现有参数

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关更多信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 运行以下命令将策略添加到现有参数。将每个*#######*替换为您自己的信息。

   Linux & macOS

```
aws ssm put-parameter
    --name "parameter name" \
    --value 'parameter value' \
    --type parameter type \
    --overwrite \
    --policies "[{policies-enclosed-in-brackets-and-curly-braces}]"
```

Windows

```
aws ssm put-parameter
    --name "parameter name" ^
    --value 'parameter value' ^
    --type parameter type ^
    --overwrite ^
    --policies "[{policies-enclosed-in-brackets-and-curly-braces}]"
```

下面是一个包含在 15 天后删除参数的到期策略的示例。此示例还包含在参数被删除前五 (5) 天生成 Eventbridge 事件的通知策略。最后，它还在 60 天后未对此参数进行更改时包含了 NoChangeNotification 策略。此示例对密码和 AWS Key Management Service AWS KMS key使用模糊名称 (3l3vat3131)。有关 AWS KMS keys 的更多信息，请参阅 AWS Key Management Service Developer Guide 中的 AWS Key Management Service 概念。

Linux & macOS

```
aws ssm put-parameter \
    --name "/Finance/Payroll/3l3vat3131" \
    --value "P@sSwW)rd" \
    --type "SecureString" \
    --overwrite \
    --policies "[{\"Type\":\"Expiration\",\"Version\":\"1.0\",\"Attributes\":
{\"Timestamp\":\"2020-05-13T00:00:00.000Z\"}},{\"Type\":\"ExpirationNotification
\",\"Version\":\"1.0\",\"Attributes\":{\"Before\":\"5\",\"Unit\":\"Days\"}},
{\"Type\":\"NoChangeNotification\",\"Version\":\"1.0\",\"Attributes\":{\"After
\":\"60\",\"Unit\":\"Days\"}}]"
```

Windows

```
aws ssm put-parameter ^
    --name "/Finance/Payroll/3l3vat3131" ^
    --value "P@sSwW)rd" ^
    --type "SecureString" ^
    --overwrite ^
    --policies "[{\"Type\":\"Expiration\",\"Version\":\"1.0\",\"Attributes\":
{\"Timestamp\":\"2020-05-13T00:00:00.000Z\"}},{\"Type\":\"ExpirationNotification
\",\"Version\":\"1.0\",\"Attributes\":{\"Before\":\"5\",\"Unit\":\"Days\"}},
```

```
{\"Type\":\"NoChangeNotification\",\"Version\":\"1.0\",\"Attributes\":{\"After
\":\"60\",\"Unit\":\"Days\"}}]"
```

3. 运行以下命令验证参数的详细信息。将####替换为您自己的信息。

   Linux & macOS

   ```
   aws ssm describe-parameters  \
       --parameter-filters "Key=Name,Values=parameter name"
   ```

   Windows

   ```
   aws ssm describe-parameters  ^
       --parameter-filters "Key=Name,Values=parameter name"
   ```

Important

- Parameter Store 会为参数保留策略，直至您使用新策略覆盖这些策略或删除这些策略。
- 要从现有参数中删除所有策略，请编辑参数并使用括号和大括号应用空策略。将每个#####替换为您自己的信息。例如：

  Linux & macOS

  ```
  aws ssm put-parameter \
      --name parameter name \
      --type parameter type  \
      --value 'parameter value' \
      --policies "[{}]"
  ```

  Windows

  ```
  aws ssm put-parameter ^
      --name parameter name ^
      --type parameter type  ^
      --value 'parameter value' ^
      --policies "[{}]"
  ```

- 如果您向已有策略的参数添加新策略，Systems Manager 将覆盖附加到该参数的策略。现有策略将被删除。如果您希望向已有一个或多个策略的参数添加新策略，请复制并粘贴原始策略，键入新策略，然后保存所做的更改。

将策略添加到现有参数 (Tools for Windows PowerShell)

按照以下过程，使用 Tools for Windows PowerShell 将策略添加到现有参数。将每个*#######*替换为您自己的信息。

将策略添加到现有参数

1. 打开 Tools for Windows PowerShell 并运行以下命令以指定您的凭证。您必须在 Amazon Elastic Compute Cloud (Amazon EC2) 中具有管理员权限，或者 AWS Identity and Access Management (IAM) 中必须为您授予了相应的权限。

```
Set-AWSCredentials `
    -AccessKey access-key-name `
    -SecretKey secret-key-name
```

2. 运行以下命令，以便为 PowerShell 会话设置区域。此示例使用美国东部（俄亥俄）区域 (us-east-2)。

```
Set-DefaultAWSRegion `
    -Region us-east-2
```

3. 运行以下命令将策略添加到现有参数。将每个*#######*替换为您自己的信息。

```
Write-SSMParameter `
    -Name "parameter name" `
    -Value "parameter value" `
    -Type "parameter type" `
    -Policies "[{policies-enclosed-in-brackets-and-curly-braces}]" `
    -Overwrite
```

下面是一个包含在 2020 年 5 月 13 日午夜 12 点 (GMT) 删除参数的到期策略的示例。此示例还包含在参数被删除前五 (5) 天生成 Eventbridge 事件的通知策略。最后，它还在 60 天后未对此参数进行更改时包含了 NoChangeNotification 策略。此示例对密码和 AWS 托管式密钥使用模糊名称 (3l3vat3131)。

```
Write-SSMParameter `
    -Name "/Finance/Payroll/3l3vat3131" `
    -Value "P@sSwW)rd" `
    -Type "SecureString" `
    -Policies "[{\"Type\":\"Expiration\",\"Version\":\"1.0\",\"Attributes\":
{\"Timestamp\":\"2018-05-13T00:00:00.000Z\"}},{\"Type\":\"ExpirationNotification
\",\"Version\":\"1.0\",\"Attributes\":{\"Before\":\"5\",\"Unit\":\"Days\"}},{\"Type
\":\"NoChangeNotification\",\"Version\":\"1.0\",\"Attributes\":{\"After\":\"60\",
\"Unit\":\"Days\"}}]" `
    -Overwrite
```

4. 运行以下命令验证参数的详细信息。将####替换为您自己的信息。

```
(Get-SSMParameterValue -Name "parameter name").Parameters
```

Important

- Parameter Store 会在参数上保留策略，直至您使用新策略覆盖这些策略或删除这些策略。
- 要从现有参数中删除所有策略，请编辑参数并使用括号和大括号应用空策略。例如：

```
Write-SSMParameter `
    -Name "parameter name" `
    -Value "parameter value" `
    -Type "parameter type" `
    -Policies "[{}]"
```

- 如果您向已有策略的参数添加新策略，Systems Manager 将覆盖附加到该参数的策略。现有策略将被删除。如果您希望向已有一个或多个策略的参数添加新策略，请复制并粘贴原始策略，键入新策略，然后保存所做的更改。

处理 Parameter Store 中的参数层次结构

以平面列表的形式管理几十个或数百个参数十分耗时且容易出错。而且为任务确定正确参数也很难。这意味着，您可能意外地使用了错误的参数，或者您可能创建多个使用相同配置数据的参数。

您可以使用参数层次结构来帮助组织和管理参数。一个层次结构是一个参数名称，包括您使用正斜杠定义的路径。

主题

- 通过示例了解参数层次结构
- 查询层次结构中的参数
- 通过 AWS CLI 来使用层次结构管理参数

通过示例了解参数层次结构

下列示例在名称中使用三个层次结构级别来标识以下内容：

```
/Environment/Type of computer/Application/Data
```

```
/Dev/DBServer/MySQL/db-string13
```

您可以创建具有最多 15 个级别的层次结构。我们建议您创建反映环境中现有层次结构的层次结构，如以下示例所示：

- 您的持续集成和持续交付环境（CI/CD 工作流）

  ```
  /Dev/DBServer/MySQL/db-string
  ```

  ```
  /Staging/DBServer/MySQL/db-string
  ```

  ```
  /Prod/DBServer/MySQL/db-string
  ```
- 您的使用容器的应用程序

  ```
  /MyApp/.NET/Libraries/my-password
  ```
- 您的业务组织

  ```
  /Finance/Accountants/UserList
  ```

  ```
  /Finance/Analysts/UserList
  ```

  ```
  /HR/Employees/EU/UserList
  ```

参数层次结构规范了创建参数的方式，而且使得随时间的推移管理参数更为容易。参数层次结构还可帮助您为配置任务确定正确参数。这可帮助您避免使用相同的配置数据创建多个参数。

您可以创建一个层次结构，允许您在不同的环境中共享参数，如以下示例所示，这些示例在开发和暂存环境中使用密码。

/DevTest/MyApp/database/*my-password*

然后您可以为生产环境创建一个唯一密码，如以下示例所示：

/prod/MyApp/database/*my-password*

您无需指定参数层次结构。您可以在第一级创建参数。它们叫做根参数。考虑到向后兼容性，在发布层次结构之前在 Parameter Store 中创建的所有参数都是根参数。系统将以下两个参数视为根参数。

/parameter-name

parameter-name

查询层次结构中的参数

使用层次结构的另一个好处是，能够通过使用 GetParametersByPath API 操作查询层次结构中某个级别下的所有参数。例如，若从 AWS Command Line Interface（AWS CLI）运行以下命令，系统会返回 Oncall 级别下的所有参数：

```
aws ssm get-parameters-by-path --path /Dev/Web/Oncall
```

示例输出：

```
{
    "Parameters": [
        {
            "Name": "/Dev/Web/Oncall/Week1",
            "Type": "String",
            "Value": "John Doe",
            "Version": 1,
            "LastModifiedDate": "2024-11-22T07:18:53.510000-08:00",
            "ARN": "arn:aws:ssm:us-east-2:123456789012:parameter/Dev/Web/Oncall/Week1",
            "DataType": "text"
        },
        {
            "Name": "/Dev/Web/Oncall/Week2",
            "Type": "String",
            "Value": "Mary Major",
            "Version": 1,
            "LastModifiedDate": "2024-11-22T07:21:25.325000-08:00",
            "ARN": "arn:aws:ssm:us-east-2:123456789012:parameter/Dev/Web/Oncall/Week2",
            "DataType": "text"
        }
    ]
```

```
}
```

要查看层次结构中解密的 SecureString 参数，请指定路径和 --with-decryption 参数，如以下示例所示。

```
aws ssm get-parameters-by-path --path /Prod/ERP/SAP --with-decryption
```

通过 AWS CLI 来使用层次结构管理参数

此过程介绍了如何通过使用 AWS CLI 来使用参数和参数层次结构。

使用层次结构管理参数

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关更多信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 运行以下命令创建一个使用 allowedPattern 参数和 String 参数类型的参数。本示例中的允许模式表示参数的值必须为 1 到 4 个数字。

   Linux & macOS

   ```
   aws ssm put-parameter \
       --name "/MyService/Test/MaxConnections" \
       --value 100 --allowed-pattern "\d{1,4}" \
       --type String
   ```

   Windows

   ```
   aws ssm put-parameter ^
       --name "/MyService/Test/MaxConnections" ^
       --value 100 --allowed-pattern "\d{1,4}" ^
       --type String
   ```

   该命令返回参数的版本号。

3. 执行以下命令尝试用新值覆盖您刚刚创建的参数。

   Linux & macOS

   ```
   aws ssm put-parameter \
   ```

```
    --name "/MyService/Test/MaxConnections" \
    --value 10,000 \
    --type String \
    --overwrite
```

Windows

```
aws ssm put-parameter ^
    --name "/MyService/Test/MaxConnections" ^
    --value 10,000 ^
    --type String ^
    --overwrite
```

系统会返回以下错误，因为新值不满足您在上一步骤中指定的允许模式的要求。

```
An error occurred (ParameterPatternMismatchException) when calling the PutParameter
 operation: Parameter value, cannot be validated against allowedPattern: \d{1,4}
```

4. 运行以下命令，创建使用 AWS 托管式密钥的 SecureString 参数。本示例中的允许模式表示用户可以指定任意字符，并且值必须在 8 到 20 个字符之间。

Linux & macOS

```
aws ssm put-parameter \
    --name "/MyService/Test/my-password" \
    --value "p#sW*rd33" \
    --allowed-pattern ".{8,20}" \
    --type SecureString
```

Windows

```
aws ssm put-parameter ^
    --name "/MyService/Test/my-password" ^
    --value "p#sW*rd33" ^
    --allowed-pattern ".{8,20}" ^
    --type SecureString
```

5. 运行以下命令创建多个使用上一步骤中的层次结构的参数。

Linux & macOS

```
aws ssm put-parameter \
    --name "/MyService/Test/DBname" \
    --value "SQLDevDb" \
    --type String
```

```
aws ssm put-parameter \
    --name "/MyService/Test/user" \
    --value "SA" \
    --type String
```

```
aws ssm put-parameter \
    --name "/MyService/Test/userType" \
    --value "SQLuser" \
    --type String
```

Windows

```
aws ssm put-parameter ^
    --name "/MyService/Test/DBname" ^
    --value "SQLDevDb" ^
    --type String
```

```
aws ssm put-parameter ^
    --name "/MyService/Test/user" ^
    --value "SA" ^
    --type String
```

```
aws ssm put-parameter ^
    --name "/MyService/Test/userType" ^
    --value "SQLuser" ^
    --type String
```

6. 运行以下命令获取两个参数的值。

Linux & macOS

```
aws ssm get-parameters \
```

```
    --names "/MyService/Test/user" "/MyService/Test/userType"
```

Windows

```
aws ssm get-parameters ^
    --names "/MyService/Test/user" "/MyService/Test/userType"
```

7. 运行以下命令查询单个级别下的所有参数。

   Linux & macOS

   ```
   aws ssm get-parameters-by-path \
       --path "/MyService/Test"
   ```

   Windows

   ```
   aws ssm get-parameters-by-path ^
       --path "/MyService/Test"
   ```

8. 运行以下命令删除两个参数。

   Linux & macOS

   ```
   aws ssm delete-parameters \
       --names "/IADRegion/Dev/user" "/IADRegion/Dev/userType"
   ```

   Windows

   ```
   aws ssm delete-parameters ^
       --names "/IADRegion/Dev/user" "/IADRegion/Dev/userType"
   ```

防止访问 Parameter Store API 操作

使用 Systems Manager for AWS Identity and Access Management（IAM）策略支持的服务特定条件，您可以明确允许或拒绝对 Parameter Store API 操作和内容的访问。通过使用这些条件，您可仅允许组织中的某些 IAM 实体（用户和角色）调用某些 API 操作，或者防止某些 IAM 实体运行它们。这包括通过 Parameter Store 控制台、AWS Command Line Interface（AWS CLI）和 SDK 运行的操作。

Systems Manager 当前支持三个特定于 Parameter Store 的条件。

主题

- 通过 ssm:Overwrite 来防止更改现有参数
- 通过 ssm:Policies 来防止创建或更新使用参数策略的参数
- 通过 ssm:Recursive 来防止访问分层参数中的级别

通过 **ssm:Overwrite** 来防止更改现有参数

使用 ssm:Overwrite 条件控制 IAM 实体是否可以更新现有参数。

在以下示例策略中，"Allow" 语句授予创建参数的权限，创建方法是在美国东部（俄亥俄州）区域（us-east-2）的 AWS 账户 123456789012 中运行 PutParameter API 操作。

然而，"Deny" 语句防止实体更改现有参数的值，因为针对 PutParameter 操作明确拒绝 Overwrite 选项。因此，为其分配此策略的实体可以创建参数，但不能对现有参数进行更改。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:PutParameter"
            ],
            "Resource": "arn:aws:ssm:us-east-2:123456789012:parameter/*"
        },
        {
            "Effect": "Deny",
            "Action": [
                "ssm:PutParameter"
            ],
            "Condition": {
                "StringEquals": {
                    "ssm:Overwrite": [
                        "true"
                    ]
                }
            },
            "Resource": "arn:aws:ssm:us-east-2:123456789012:parameter/*"
```

```
        }
    ]
}
```

通过 **ssm:Policies** 来防止创建或更新使用参数策略的参数

使用 ssm:Policies 条件来控制实体是否可以创建包含参数策略的参数，以及是否可以更新包含参数策略的现有参数。

在以下策略示例中，"Allow" 语句将授予创建参数的一般权限，而 "Deny" 语句会阻止实体在美国东部（俄亥俄州）区域（us-east-2）的 AWS 账户 123456789012 中创建或更新包含某个参数策略的参数。实体仍然可以创建或更新未分配参数策略的参数。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:PutParameter"
            ],
            "Resource": "arn:aws:ssm:us-east-2:111122223333:parameter/*"
        },
        {
            "Effect": "Deny",
            "Action": [
                "ssm:PutParameter"
            ],
            "Condition": {
                "StringEquals": {
                    "ssm:Policies": [
                        "true"
                    ]
                }
            },
            "Resource": "arn:aws:ssm:us-east-2:111122223333:parameter/*"
        }
    ]
}
```

通过 **ssm:Recursive** 来防止访问分层参数中的级别

使用 ssm:Recursive 条件控制 IAM 实体是否可以查看或引用分层参数中的级别。您可以提供或限制对层次结构中特定级别之外的所有参数的访问权限。

在以下示例策略中，"Allow" 语句提供对美国东部（俄亥俄州）区域（us-east-2）中的 AWS 账户 123456789012 的路径 /Code/Departments/* 中所有参数的 Parameter Store 操作访问权限。

此后，"Deny" 语句会防止 IAM 实体查看或检索位于或低于 /Code/Departments/* 级别的参数数据。但是，实体仍然可以在该路径中创建或更新参数。本示例的构造是为了说明递归拒绝参数层次结构中低于某个级别的访问权限优先于同一策略中更宽松的访问权限。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:*"
            ],
            "Resource": "arn:aws:ssm:us-east-2:123456789012:parameter/*"
        },
        {
            "Effect": "Deny",
            "Action": [
                "ssm:GetParametersByPath"
            ],
            "Condition": {
                "StringEquals": {
                    "ssm:Recursive": [
                        "true"
                    ]
                }
            },
            "Resource": "arn:aws:ssm:us-east-2:123456789012:parameter/Code/
Departments/*"
        }
    ]
}
```

> **Important**
>
> 如果用户有权访问某个路径，则该用户可以访问该路径的所有级别。例如，如果某个用户有权访问路径 /a，则该用户也可以访问 /a/b。除非用户在 IAM 中被明确拒绝参数 /b 的访问权限（如上所示），否则就是如此情况。

处理 Parameter Store 中的参数标签

参数标签是用户定义的别名，帮助管理参数的不同版本。修改参数时，AWS Systems Manager 将自动保存新版本并将版本号增加 1。标签可帮助您在存在多个版本时记住参数版本的用途。

例如，假设您有一个名为 /MyApp/DB/ConnectionString 的参数。参数值在测试环境中是本地数据库中 MySQL 服务器的连接字符串。更新完应用程序之后，您希望此参数使用生产数据库的连接字符串。您应更改 /MyApp/DB/ConnectionString 的值。Systems Manager 将自动使用新连接字符串创建版本 2。为帮助您记住每个版本的用途，您为每个参数附加了一个标签。为版本 1 附加 Test 标签，并为版本 2 附加 Production 标签。

您可以将一个参数版本的标签移至另一个版本。例如，如果您使用新生产数据库的连接字符串为 /MyApp/DB/ConnectionString 参数创建版本 3，则可将 Production 标签从参数的版本 2 移至参数的版本 3。

参数标签是参数标记的轻量级替代。您的组织可能对必须应用于不同 AWS 资源的标记具有严格的准则。相比之下，标签只是参数特定版本的文本关联。

与标签类似，您可使用标签查询参数。如果使用 GetParametersByPath API 操作查询参数设置，则可查看使用相同标签的所有特定参数版本的列表，如本部分后文所述。

> **Note**
>
> 如果您运行的命令指定了不存在的参数版本，则该命令将失败。它不会回退到该参数的最新或默认值。

标签要求和限制

参数标签具有以下要求和限制：

- 一个参数版本最多可以具有 10 个标签。

- 您无法将同一标签附加到同一参数的不同版本。例如，如果参数的版本 1 具有标签 Production，则无法将 Production 附加到版本 2。
- 您可以将一个参数版本的标签移至另一个参数版本。
- 创建参数时，无法创建标签。必须将标签附加到参数的特定版本。
- 如果您不再想使用参数标签，可将其移至参数的其他版本或将其删除。
- 标签最多可包含 100 个字符。
- 标签可以包含字母（区分大小写）、数字、句点 (.)、连字符 (-) 或下划线 (_)。
- 标签不能以数字、“aws”或“ssm”（不区分大小写）开头。如果标签不满足这些要求，则不会将标签附加到参数版本，并且系统将在 `InvalidLabels` 的列表中显示它。

主题

- 使用控制台处理参数标签
- 使用 AWS CLI 处理参数标签

使用控制台处理参数标签

此部分介绍了如何使用 Systems Manager 控制台执行以下任务。

- 使用控制台来创建参数标签
- 使用控制台来查看附加到参数的标签
- 使用控制台来移动参数标签
- 使用控制台来删除参数标签

使用控制台来创建参数标签

以下过程介绍了如何使用 Systems Manager 控制台将标签附加到现有 参数的特定版本。创建参数时，无法附加标签。

要将标签附加到某个参数版本，请执行以下步骤：

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Parameter Store。
3. 选择参数的名称以打开该参数的详细信息页面。

4. 选择 History 选项卡。
5. 选择要为其附加标签的参数版本。
6. 选择 Manage labels (管理标签)。
7. 选择 Add new label (添加新标签)。
8. 在文本框中，输入标签名称。要添加更多标签，请选择 Add new label (添加新标签)。您最多可以附加 10 个标签。
9. 完成后，选择保存更改。

使用控制台来查看附加到参数的标签

一个参数版本最多可以具有 10 个标签。以下过程介绍了如何使用 Systems Manager 控制台查看附加到参数版本的所有标签。

要查看附加到参数版本的标签，请执行以下步骤：

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Parameter Store。
3. 选择参数的名称以打开该参数的详细信息页面。
4. 选择 History 选项卡。
5. 找到要查看其所有附加标签的参数版本。Labels (标签) 列显示附加到参数版本的所有标签。

使用控制台来移动参数标签

以下过程介绍了如何使用 Systems Manager 控制台将参数标签移至同一参数的其他版本。

要将标签移至其他参数版本，请执行以下步骤：

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Parameter Store。
3. 选择参数的名称以打开该参数的详细信息页面。
4. 选择 History 选项卡。
5. 选择要移动其标签的参数版本。
6. 选择 Manage labels (管理标签)。

7. 选择 Add new label (添加新标签)。
8. 在文本框中，输入标签名称。
9. 完成后，选择保存更改。

使用控制台来删除参数标签

以下过程介绍了如何使用 Systems Manager 控制台删除一个或多个参数标签。

要删除参数的标签，请执行以下步骤：

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Parameter Store。
3. 选择参数的名称以打开该参数的详细信息页面。
4. 选择 History 选项卡。
5. 选择要删除其标签的参数版本。
6. 选择 Manage labels (管理标签)。
7. 选择要删除的每个标签旁边的 Remove (删除)。
8. 完成后，选择保存更改。
9. 确认您的更改是正确的，在文本框中输入 `Confirm`，然后选择 Confirm (确认)。

使用 AWS CLI 处理参数标签

此部分介绍了如何使用 AWS Command Line Interface (AWS CLI) 执行以下任务。

- 使用 AWS CLI 来创建新的参数标签
- 使用 AWS CLI 来查看参数标签
- 使用 AWS CLI 来查看已分配标签的参数列表
- 使用 AWS CLI 来移动参数标签
- 使用 AWS CLI 来删除参数标签

使用 AWS CLI 来创建新的参数标签

以下过程介绍如何使用 将标签附加到现有AWS CLI 参数的特定版本。创建参数时，无法附加标签。

要创建参数标签，请执行以下步骤：

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关更多信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 运行以下命令以查看您有权为其附加标签的参数的列表。

   > **Note**
   >
   > 参数只在创建它的 AWS 区域 可用。如果看不到要为其附加标签的参数，请验证您的区域。

   ```
   aws ssm describe-parameters
   ```

   记下要为其附加标签的参数的名称。

3. 运行以下命令，查看该参数的所有版本。

   ```
   aws ssm get-parameter-history --name "parameter-name"
   ```

   记下要为其附加标签的参数版本。

4. 运行以下命令，按版本号检索有关参数的信息。

   ```
   aws ssm get-parameters --names "parameter-name:version-number"
   ```

   见下列。

   ```
   aws ssm get-parameters --names "/Production/SQLConnectionString:3"
   ```

5. 运行以下命令之一，以将标签附加到参数版本。如果附加多个标签，请用空格分隔标签名称。

   将标签附加到参数的最新版本

   ```
   aws ssm label-parameter-version --name parameter-name  --labels label-name
   ```

   将标签附加到参数的特定版本

```
aws ssm label-parameter-version --name parameter-name --parameter-version version-number --labels label-name
```

下面是一些示例。

```
aws ssm label-parameter-version --name /config/endpoint --labels production east-region finance
```

```
aws ssm label-parameter-version --name /config/endpoint --parameter-version 3 --labels MySQL-test
```

> **Note**
>
> 如果输出在 InvalidLabels 列表中显示您创建的标签，则表示此标签不符合本主题前面所述的要求。请查看要求并重试。如果 InvalidLabels 列表为空，则表示您的标签已成功应用于参数版本。

6. 可以使用版本号或标签名称查看参数的详细信息。运行以下命令并指定您在上一步中创建的标签。

```
aws ssm get-parameter --name parameter-name:label-name --with-decryption
```

此命令会返回如下信息。

```
{
    "Parameter": {
        "Version": version-number,
        "Type": "parameter-type",
        "Name": "parameter-name",
        "Value": "parameter-value",
        "Selector": ":label-name"
    }
}
```

> **Note**
>
> 输出中的 Selector 是您在 Name 输入字段中指定的版本号或标签。

使用 AWS CLI 来查看参数标签

您可以使用 GetParameterHistory API 操作查看完整历史记录和附加到指定参数的所有标签。或者，您可以使用 GetParametersByPath API 操作查看分配有特定标签的所有参数的列表。

要使用 GetParameterHistory API 操作查看参数的标签，请执行以下步骤：

1. 运行以下命令以查看您可以查看其标签的参数的列表。

   > **Note**
   >
   > 参数只在创建它的区域可用。如果看不到要移动其标签的参数，请验证您的区域。

   ```
   aws ssm describe-parameters
   ```

   请注意要查看其标签的参数的名称。

2. 运行以下命令，查看该参数的所有版本。

   ```
   aws ssm get-parameter-history --name parameter-name --with-decryption
   ```

   系统将返回类似于以下内容的信息。

   ```
   {
       "Parameters": [
           {
               "Name": "/Config/endpoint",
               "LastModifiedDate": 1528932105.382,
               "Labels": [
                   "Deprecated"
               ],
               "Value": "MyTestService-June-Release.example.com",
               "Version": 1,
               "LastModifiedUser": "arn:aws:iam::123456789012:user/test",
               "Type": "String"
           },
           {
               "Name": "/Config/endpoint",
               "LastModifiedDate": 1528932111.222,
               "Labels": [
   ```

```
                "Current"
            ],
            "Value": "MyTestService-July-Release.example.com",
            "Version": 2,
            "LastModifiedUser": "arn:aws:iam::123456789012:user/test",
            "Type": "String"
        }
    ]
}
```

使用 AWS CLI 来查看已分配标签的参数列表

您可以使用 GetParametersByPath API 操作查看路径中分配有特定标签的所有参数的列表。

运行以下命令以查看路径中分配有特定标签的参数的列表。将每个*#######*替换为您自己的信息。

```
aws ssm get-parameters-by-path \
    --path parameter-path \
    --parameter-filters Key=Label,Values=label-name,Option=Equals \
    --max-results a-number \
    --with-decryption --recursive
```

系统将返回类似于以下内容的信息。在此示例中，用户在 /Config 路径下进行了搜索。

```
{
    "Parameters": [
        {
            "Version": 3,
            "Type": "SecureString",
            "Name": "/Config/DBpwd",
            "Value": "MyS@perGr&pass33"
        },
        {
            "Version": 2,
            "Type": "String",
            "Name": "/Config/DBusername",
            "Value": "TestUserDB"
        },
        {
            "Version": 2,
            "Type": "String",
            "Name": "/Config/endpoint",
```

```
            "Value": "MyTestService-July-Release.example.com"
        }
    ]
}
```

使用 AWS CLI 来移动参数标签

以下过程介绍了如何将参数标签移至同一参数的其他版本。

移动参数标签

1. 运行以下命令，查看该参数的所有版本。将####替换为您自己的信息。

```
aws ssm get-parameter-history \
    --name "parameter name"
```

请注意要将标签移入和移出的参数的版本。

2. 运行以下命令以将现有标签分配给参数的其他版本。将每个#######替换为您自己的信息。

```
aws ssm label-parameter-version \
    --name parameter name \
    --parameter-version version number \
    --labels name-of-existing-label
```

> **Note**
>
> 如果要将现有标签移至参数的最新版本，则删除此命令中的 `--parameter-version`。

使用 AWS CLI 来删除参数标签

以下过程介绍了如何使用 AWS CLI 删除参数标签。

要删除参数标签，请执行以下操作：

1. 运行以下命令，查看该参数的所有版本。将####替换为您自己的信息。

```
aws ssm get-parameter-history \
    --name "parameter name"
```

系统将返回类似于以下内容的信息。

```
{
    "Parameters": [
        {
            "Name": "foo",
            "DataType": "text",
            "LastModifiedDate": 1607380761.11,
            "Labels": [
                "l3",
                "l2"
            ],
            "Value": "test",
            "Version": 1,
            "LastModifiedUser": "arn:aws:iam::123456789012:user/test",
            "Policies": [],
            "Tier": "Standard",
            "Type": "String"
        },
        {
            "Name": "foo",
            "DataType": "text",
            "LastModifiedDate": 1607380763.11,
            "Labels": [
                "l1"
            ],
            "Value": "test",
            "Version": 2,
            "LastModifiedUser": "arn:aws:iam::123456789012:user/test",
            "Policies": [],
            "Tier": "Standard",
            "Type": "String"
        }
    ]
}
```

请注意要删除其一个或多个标签的参数的版本。

2. 运行以下命令，删除您在参数中选择的标签。将每个*#######*替换为您自己的信息。

```
aws ssm unlabel-parameter-version \
    --name parameter name \
```

```
    --parameter-version version \
    --labels label 1,label 2,label 3
```

系统将返回类似于以下内容的信息。

```
{
    "InvalidLabels": ["invalid"],
    "DeletedLabels" : ["Prod"]
 }
```

处理 Parameter Store 中的参数版本

每次编辑参数的值时，Parameter Store（AWS Systems Manager 中的一项工具）都会创建参数的新版本，也会保留以前的版本。在最初创建一个参数时，Parameter Store 为该参数分配版本 1。更改参数的值后，Parameter Store 会自动将版本号增加 1。您可以在参数的历史记录中查看所有版本的详细信息（包括值）。

您还可以指定要在 API 命令和 SSM 文档中使用的参数的版本；例如：`ssm:MyParameter:3`。您可以在 API 调用和 SSM 文档中指定参数名和特定版本号。如果不指定版本号，系统自动使用最新版本。如果您为不存在的版本指定编号，则系统将返回错误，而不会回退到该参数的最新或默认版本。

您可以使用参数版本查看一段时间内更改参数的次数。此外，参数版本提供了一层保护，以防止参数值被意外更改。

您最多可以为一个参数创建和维护 100 个版本。在创建了 100 个参数版本后，每次创建新版本时，都会从历史记录中删除参数的最旧版本，以便为新版本腾出空间。

此情况的一个例外是历史记录中已经有 100 个参数版本，并且参数标签被分配给参数的最旧版本。在这种情况下，不会从历史记录中删除该版本，创建新参数版本的请求将失败。此保护措施旨在防止分配有任务关键型标签的参数版本被删除。要继续创建新参数，请首先将标签从参数的最旧版本移至较新版本，以便在操作中使用。有关移动参数标签的信息，请参阅 使用控制台来移动参数标签 和 使用 AWS CLI 来移动参数标签。

以下过程介绍了如何编辑参数，并验证是否已创建新版本。您可以使用 `get-parameter` 和 `get-parameters` 命令查看参数版本。有关使用这些命令的示例，请参阅《AWS Systems Manager API 参考》中的 GetParameter 和 GetParameters

主题

- 使用控制台创建参数的新版本

- 引用参数版本

使用控制台创建参数的新版本

您可以使用 Systems Manager 控制台创建参数的新版本并查看参数的版本历史记录。

创建参数的新版本

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Parameter Store。
3. 选择之前创建的参数的名称。有关创建新参数的信息，请参阅 在 Systems Manager 中创建 Parameter Store 参数。
4. 选择编辑。
5. 在 Value (值) 框中输入新的值，然后选择 Save changes (保存更改)。
6. 选择刚刚更新的参数的名称。在 Overview 选项卡上，验证版本号是否增加 1，并验证新值。
7. 要查看参数的所有版本的历史记录，请选择 History (历史记录) 选项卡。

引用参数版本

可以使用以下格式，在命令、API 调用和 SSM 文档中引用特定的参数版本：ssm:*parameter-name*:*version-number*。

在以下示例中，Amazon Elastic Compute Cloud (Amazon EC2) `run-instances command` 使用参数 `golden-ami` 的版本 3。

Linux & macOS

```
aws ec2 run-instances \
    --image-id resolve:ssm:/golden-ami:3 \
    --count 1 \
    --instance-type t2.micro \
    --key-name my-key-pair \
    --security-groups my-security-group
```

Windows

```
aws ec2 run-instances ^
```

```
    --image-id resolve:ssm:/golden-ami:3 ^
    --count 1 ^
    --instance-type t2.micro ^
    --key-name my-key-pair ^
    --security-groups my-security-group
```

> **Note**
>
> 使用 `resolve` 和参数值仅适用于 `--image-id` 选项和包含 Amazon Machine Image (AMI) 作为其值的参数。有关更多信息，请参阅 在 Parameter Store 中为 Amazon Machine Image ID 使用本机参数支持。

以下是有关在 SSM 文档中指定名为 `MyRunCommandParameter` 的参数的版本 2 的示例。

YAML

```
---
schemaVersion: '2.2'
description: Run a shell script or specify the commands to run.
parameters:
  commands:
    type: String
    description: "(Required) Specify a shell script or a command to run."
    displayType: textarea
    default: "{{ssm:MyRunCommandParameter:2}}"
mainSteps:
- action: aws:runShellScript
  name: RunScript
  inputs:
    runCommand:
    - "{{commands}}"
```

JSON

```
{
    "schemaVersion": "2.2",
    "description": "Run a shell script or specify the commands to run.",
    "parameters": {
        "commands": {
            "type": "String",
```

```
            "description": "(Required) Specify a shell script or a command to run.",
            "displayType": "textarea",
            "default": "{{ssm:MyRunCommandParameter:2}}"
        }
    },
    "mainSteps": [
        {
            "action": "aws:runShellScript",
            "name": "RunScript",
            "inputs": {
                "runCommand": [
                    "{{commands}}"
                ]
            }
        }
    ]
}
```

使用 Parameter Store 中的共享参数

共享高级参数可简化多账户环境中的配置数据管理操作。您可以集中存储和管理参数，与需要引用这些参数的其他 AWS 账户 共享参数。

Parameter Store 与 AWS Resource Access Manager（AWS RAM）集成，可实现高级参数共享。AWS RAM 是一项服务，让您能够与其他 AWS 账户 共享资源，或者通过 AWS Organizations 共享资源。

通过使用 AWS RAM，您可以创建资源共享以共享您拥有的资源。资源共享指定要共享的资源、要授予的权限以及与之共享资源的使用者。使用者可包括：

- AWS Organizations 中其组织内部或外部的特定 AWS 账户。
- AWS Organizations 中的所有者组织内部的组织单位
- 它在 AWS Organizations 中的整个组织

有关 AWS RAM 的更多信息，请参阅 AWS RAM 用户指南。

本主题说明如何共享您拥有的参数以及如何使用共享给您的参数。

内容

- 共享参数的先决条件

- 共享参数
- 停止共享共享参数
- 标识共享参数
- 访问共享参数
- 共享参数的权限集
- 共享参数的最大吞吐量
- 共享参数的定价
- 已关闭 AWS 账户 的跨账户访问权限

共享参数的先决条件

要共享账户中的参数，就必须先满足以下先决条件：

- 要共享参数，您必须在 AWS 账户 中拥有该参数。您无法共享已与您共享的参数。
- 要共享参数，该参数就必须在高级参数层中。有关参数层的信息，请参阅 管理参数层。有关将现有标准参数更改为高级参数的信息，请参阅 将标准参数更改为高级参数。
- 要共享 `SecureString` 参数，必须使用客户托管密钥对其进行加密，并且必须通过 AWS Key Management Service 单独共享该密钥。无法共享 AWS 托管式密钥。使用默认 AWS 托管式密钥 加密的参数可以更新为改用客户托管密钥。有关 AWS KMS 密钥的定义，请参阅《AWS Key Management Service Developer Guide》中的 AWS KMS concepts。
- 要与组织或 AWS Organizations 内的组织单位共享参数，您必须允许与 AWS Organizations 共享。有关更多信息，请参阅AWS RAM《用户指南》中的允许与 AWS Organizations 共享。

共享参数

要共享参数，您必须将其添加到资源共享。资源共享是一项 AWS RAM 资源，可让您跨 AWS 账户 共享资源。资源共享指定要共享的资源以及与之共享资源的使用者。

在与其他 AWS 账户 共享自己拥有的参数时，您可以从两个 AWS 托管权限中进行选择，授予使用者相应权限。有关更多信息，请参阅 共享参数的权限集。

如果您属于 AWS Organizations 内的某个组织，并启用了组织内共享，则您可以授予组织中的使用者从 AWS RAM 控制台访问共享参数的权限。否则，使用者会收到加入资源共享的邀请，并在接受邀请后获得对共享参数的访问权限。

您可以使用 AWS RAM 控制台或 AWS CLI 共享自己拥有的参数。

> **Note**
>
> 虽然可以使用 Systems Manager PutResourcePolicy API 操作共享参数，但建议使用 AWS Resource Access Manager（AWS RAM）。这是因为使用 PutResourcePolicy 需要执行额外步骤，即使用 AWS RAM PromoteResourceShareCreatedFromPolicy API 操作将参数提升为标准资源共享。否则，Systems Manager DescribeParameters API 操作不会使用 --shared 选项返回该参数。

使用 AWS RAM 控制台共享拥有的参数

请参阅《AWS RAM 用户指南》中的在 AWS RAM 中创建资源共享。

完成该过程时进行以下选择：

- 在“步骤 1”页面中，对于资源，选择 Parameter Store Advanced Parameter，然后选中要共享的高级参数层中每个参数的框。
- 在“步骤 2”页面中，对于托管式权限，选择要授予使用者的权限，如本主题后面部分的 共享参数的权限集 中所述。

根据参数共享目标选择其他选项。

使用 AWS CLI 共享拥有的参数

使用 create-resource-share 命令向新的资源共享添加参数。

使用 associate-resource-share 命令向现有资源共享添加参数。

以下示例创建了一个新的资源共享，以便与组织和个人账户中的使用者共享参数。

```
aws ram create-resource-share \
    --name "MyParameter" \
    --resource-arns "arn:aws:ssm:us-east-2:123456789012:parameter/MyParameter" \
    --principals "arn:aws:organizations::123456789012:ou/o-63bEXAMPLE/ou-46xi-rEXAMPLE"
 "987654321098"
```

停止共享共享参数

停止共享共享参数后，使用者账户无法再访问该参数。

要停止共享拥有的参数，必须从资源共享中将其移除。您可以使用 Systems Manager 控制台、AWS RAM 控制台或 AWS CLI 完成此操作。

使用 AWS RAM 控制台停止共享拥有的参数

请参阅《AWS RAM User Guide》中的 Update a resource share in AWS RAM。

使用 AWS CLI 停止共享拥有的参数

使用 disassociate-resource-share 命令。

标识共享参数

拥有者和使用者可以使用 AWS CLI 标识共享参数。

使用 AWS CLI 标识共享参数

要使用 AWS CLI 标识共享参数，可以选用 Systems Manager `describe-parameters` 命令或 AWS RAM `list-resources` 命令。

在将 `--shared` 选项与 `describe-parameters` 一起使用后，该命令会返回与您共享的参数。

以下是示例：

```
aws ssm describe-parameters --shared
```

访问共享参数

使用者可以使用 AWS 命令行工具和 AWS SDK 访问共享参数。对于使用者账户，与该类账户共享的参数未包含在我的参数页面中。

CLI 示例：使用 AWS CLI 访问共享参数的详细信息

要使用 AWS CLI 访问共享参数的详细信息，可以使用 get-parameter 或 get-parameters 命令。必须将完整参数 ARN 指定为 `--name`，才能从其他账户检索参数。

示例如下：

```
aws ssm get-parameter \
    --name arn:aws:ssm:us-east-2:123456789012:parameter/MySharedParameter
```

共享参数支持和不支持的集成

目前可在以下集成场景中使用共享参数：

- AWS CloudFormation 模板参数
- AWS 参数和密钥 Lambda 扩展
- Amazon Elastic Compute Cloud（EC2）启动模板
- 使用 EC2 RunInstances 命令从 Amazon Machine Image（AMI）创建实例的 `ImageID` 值
- 针对 Automation（Systems Manager 中的一项工具）在运行手册中检索参数值

以下场景和集成服务目前不支持使用共享参数：

- Run Command（Systems Manager 中的一项工具）中命令中的参数
- AWS CloudFormation 动态引用
- AWS CodeBuild 中环境变量的值
- AWS App Runner 中环境变量的值
- Amazon Elastic Container Service 中密钥的值

共享参数的权限集

使用者账户收到您与其共享参数的只读访问权限。使用者无法更新或删除该类参数。使用者无法与第三个账户共享该类参数。

在 AWS Resource Access Manager 中创建用于共享参数的资源共享时，可以从两个 AWS 托管式权限集中进行选择，以授予此只读访问权限：

AWSRAMDefaultPermissionSSMParameterReadOnly

允许的操作：`DescribeParameters`、`GetParameter`、`GetParameters`

AWSRAMPermissionSSMParameterReadOnlyWithHistory

允许的操作：`DescribeParameters`、`GetParameter`、`GetParameters`、`GetParameterHistory`

按照《AWS RAM 用户指南》中在 AWS RAM 中创建资源共享的步骤进行操作时，选择 `Parameter Store Advanced Parameters` 作为资源类型，并根据您是否希望用户查看参数历史记录选择其中一种托管式权限。

> **Note**
>
> 如果以编程方式（例如使用 AWS Lambda）检索共享参数，则可能需要向任何调用 AWS Resource Access Manager API 操作的 IAM 角色添加 `ssm:GetResourcePolicies` 和 `ssm:PutResourcePolicy` 权限。

共享参数的最大吞吐量

Systems Manager 限制了 GetParameter 和 GetParameters 操作的最大吞吐量（每秒事务数）。吞吐量在个人账户级别强制实施。因此，使用共享参数的每个账户都可以使用允许的最大吞吐量，不会受到其他账户的影响。有关参数最大吞吐量的更多信息，请参阅以下主题：

- 提高 Parameter Store 吞吐量
- 《Amazon Web Services 一般参考》中的 Systems Manager service quotas。

共享参数的定价

只有高级参数层中支持跨账户共享。对于高级参数，每个高级参数的存储和 API 使用量按当前价格计费。高级参数的拥有账户需要支付此类参数的存储费用。对共享的高级参数进行 API 调用的使用账户，则需支付此类参数的使用费。

例如，账户 A 创建了高级参数 `MyAdvancedParameter`，则该账户每月需支付 0.05 美元来存储该参数。

然后，账户 A 与账户 B 和账户 C 共享 `MyAdvancedParameter`。在一个月内，这三个账户都会调用 `MyAdvancedParameter`。下表说明了各账户因调用次数产生的费用。

> **Note**
>
> 下表列示的费用仅供说明之用。要验证当前定价，请参阅 Parameter Store 的 AWS Systems Manager 定价。

Account	调用次数	收费
账户 A（拥有账户）	一万次调用	• 存储高级参数一个月：0.05 美元

Account	调用次数	收费
		• 调用 MyAdvanceParameter 一万次：0.05 美元 • 总额：0.10 美元
账户 B（使用账户）	两万次调用	• 调用 MyAdvanceParameter 两万次：0.10 美元 • 总额：0.10 美元
账户 C（使用账户）	三万次调用	• 调用 MyAdvanceParameter 三万次：0.15 美元 • 总额：0.15 美元

已关闭 AWS 账户 的跨账户访问权限

如果拥有共享参数的 AWS 账户 已关闭，则所有使用账户都会失去对共享参数的访问权限。如果拥有账户在账户关闭后的 90 天内重新打开，则使用账户会重新获得对先前共享参数的访问权限。有关在关闭后期间重新打开账户的更多信息，请参阅《AWS 账户管理 Reference Guide》中的 Accessing your AWS 账户 after you close it。

使用 Run Command 命令处理 Parameter Store 中的参数

您可以使用 Run Command（AWS Systems Manager 中的一项工具）中的参数。有关更多信息，请参阅 AWS Systems Manager Run Command。

使用控制台来运行 String 参数

以下过程将指导您完成运行使用 String 参数的命令的过程。

要使用 Parameter Store 运行 String 参数，请执行以下步骤：

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Run Command。

3. 选择 Run command（运行命令）。
4. 在 Command document 列表中，请选择 `AWS-RunPowerShellScript` (Windows) 或 `AWS-RunShellScript` (Linux)。
5. 对于命令参数，请输入 **`echo {{ssm:parameter-name}}`**。例如：**`echo {{ssm:/Test/helloWorld}}`**。
6. 在 Targets（目标）部分中，通过指定标签、手动选择实例或边缘设备或指定资源组，选择要在其上运行此操作的托管式节点。

> **Tip**
>
> 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

7. 对于 Other parameters（其他参数）：
   - 对于 Comment（注释），请输入有关此命令的信息。
   - 对于 Timeout (seconds) (超时 (秒))，请指定在整个命令执行失败之前系统等待的秒数。
8. 对于 Rate control（速率控制）：
   - 对于 Concurrency（并发），请指定要同时运行该命令的托管式节点的数量或百分比。

> **Note**
>
> 如果您通过指定应用于托管式节点的标签或指定 AWS Resource Groups 来选择目标，但不确定有多少个托管式节点已被设为目标，则可通过指定百分比来限制可同时运行该文档的目标的数量。

   - 对于 Error threshold（错误阈值），请指定当命令在一定数量或百分比的节点上失败后，何时在其他托管式节点上停止运行该命令。例如，如果您指定三个错误，Systems Manager 将在收到第四个错误时停止发送该命令。仍在处理该命令的托管式节点也可能发送错误。
9. （可选）对于 输出选项，要将命令输出保存到文件，请选中 将命令输出写入 S3 存储桶 框。在输入框中输入存储桶和前缀（文件夹）名称。

> **Note**
>
> 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给实例的实例配置文件（适用于 EC2 实例）或 IAM 服务角色（混合激活的计算机）的权限，而不是执行此任务的 IAM 用

户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确保与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

10. 在 SNS 通知部分，如果需要发送有关命令执行状态的通知，请选中 Enable SNS notifications（启用 SNS 通知）复选框。

    有关为 Run Command 配置 Amazon SNS 通知的更多信息，请参阅 使用 Amazon SNS 通知监控 Systems Manager 状态更改。

11. 选择 Run（运行）。

12. 在 Command ID（命令 ID）页面上的 Targets and outputs（目标和输出）区域中，选择在其中运行命令的节点 ID 旁边的按钮，然后选择 View output（查看输出）。验证命令输出是否是您为参数提供的值，例如 **`This is my first parameter`**。

使用AWS CLI 来运行参数

示例 1：简单命令

以下示例命令包含一个名为 `DNS-IP` 的 Systems Manager 参数。此参数的值即为节点的 IP 地址。此示例使用 AWS Command Line Interface (AWS CLI) 命令重复参数值。

Linux & macOS

```
aws ssm send-command \
    --document-name "AWS-RunShellScript" \
    --document-version "1" \
    --targets "Key=instanceids,Values=i-02573cafcfEXAMPLE" \
    --parameters "commands='echo {{ssm:DNS-IP}}'" \
    --timeout-seconds 600 \
    --max-concurrency "50" \
    --max-errors "0" \
    --region us-east-2
```

Windows

```
aws ssm send-command ^
    --document-name "AWS-RunPowerShellScript" ^
    --document-version "1" ^
    --targets "Key=instanceids,Values=i-02573cafcfEXAMPLE" ^
    --parameters "commands='echo {{ssm:DNS-IP}}'" ^
```

```
    --timeout-seconds 600 ^
    --max-concurrency "50" ^
    --max-errors "0" ^
    --region us-east-2
```

此命令会返回如下信息。

```
{
    "Command": {
        "CommandId": "c70a4671-8098-42da-b885-89716EXAMPLE",
        "DocumentName": "AWS-RunShellScript",
        "DocumentVersion": "1",
        "Comment": "",
        "ExpiresAfter": "2023-12-26T15:19:17.771000-05:00",
        "Parameters": {
            "commands": [
                "echo {{ssm:DNS-IP}}"
            ]
        },
        "InstanceIds": [],
        "Targets": [
            {
                "Key": "instanceids",
                "Values": [
                    "i-02573cafcfEXAMPLE"
                ]
            }
        ],
        "RequestedDateTime": "2023-12-26T14:09:17.771000-05:00",
        "Status": "Pending",
        "StatusDetails": "Pending",
        "OutputS3Region": "us-east-2",
        "OutputS3BucketName": "",
        "OutputS3KeyPrefix": "",
        "MaxConcurrency": "50",
        "MaxErrors": "0",
        "TargetCount": 0,
        "CompletedCount": 0,
        "ErrorCount": 0,
        "DeliveryTimedOutCount": 0,
        "ServiceRole": "",
        "NotificationConfig": {
```

```
            "NotificationArn": "",
            "NotificationEvents": [],
            "NotificationType": ""
        },
        "CloudWatchOutputConfig": {
            "CloudWatchLogGroupName": "",
            "CloudWatchOutputEnabled": false
        },
        "TimeoutSeconds": 600,
        "AlarmConfiguration": {
            "IgnorePollAlarmFailure": false,
            "Alarms": []
        },
        "TriggeredAlarms": []
    }
}
```

命令执行完成后，您可以使用以下命令查看有关命令执行的更多信息：

- get-command-invocation – 查看有关命令执行的详细信息。
- list-command-invocations – 查看特定托管式节点上的命令执行状态。
- list-commands – 查看若干托管式节点的命令执行状态。

示例 2：解密 **SecureString** 参数值

下一个示例命令使用名为 SecurePassword 的 SecureString 参数。在 parameters 中使用的命令可检索和解密 SecureString 参数的值，然后重置本地管理员密码，而无需以明文形式传递密码。

Linux

```
aws ssm send-command \
        --document-name "AWS-RunShellScript" \
        --document-version "1" \
        --targets "Key=instanceids,Values=i-02573cafcfEXAMPLE" \
        --parameters '{"commands":["secure=$(aws ssm get-parameters --names
 SecurePassword --with-decryption --query Parameters[0].Value --output text --region
 us-east-2)","echo $secure | passwd myuser --stdin"]}' \
        --timeout-seconds 600 \
        --max-concurrency "50" \
        --max-errors "0" \
        --region us-east-2
```

Windows

```
aws ssm send-command ^
        --document-name "AWS-RunPowerShellScript" ^
        --document-version "1" ^
        --targets "Key=instanceids,Values=i-02573cafcfEXAMPLE" ^
        --parameters "commands=['$secure = (Get-SSMParameterValue -Names
 SecurePassword -WithDecryption $True).Parameters[0].Value','net user administrator
 $secure']" ^
        --timeout-seconds 600 ^
        --max-concurrency "50" ^
        --max-errors "0" ^
        --region us-east-2
```

示例 3：在 SSM 文档中引用参数

您还可以在 SSM 文档的 Parameters 部分引用 Systems Manager 参数，如以下示例所示。

```
{
   "schemaVersion":"2.0",
   "description":"Sample version 2.0 document v2",
   "parameters":{
      "commands" : {
        "type": "StringList",
        "default": ["{{ssm:parameter-name}}"]
      }
    },
    "mainSteps":[
       {
          "action":"aws:runShellScript",
          "name":"runShellScript",
          "inputs":{
             "runCommand": "{{commands}}"
          }
       }
    ]
}
```

不要将 SSM 文档的 `runtimeConfig` 部分中使用的本地参数的类似句法与 Parameter Store 参数混淆。本地参数不同于 Systems Manager 参数。您可以通过是否存在 `ssm:` 前缀区分本地参数和 Systems Manager 参数：

```
"runtimeConfig":{
        "aws:runShellScript":{
            "properties":[
                {
                    "id":"0.aws:runShellScript",
                    "runCommand":"{{ commands }}",
                    "workingDirectory":"{{ workingDirectory }}",
                    "timeoutSeconds":"{{ executionTimeout }}"
```

Note

SSM 文档不支持对 SecureString 参数的引用。因此，要通过 Run Command（举例来说）使用 SecureString 的参数，您必须在将参数值传递到 Run Command 之前，检索这些参数值，如以下示例所示。

Linux & macOS

```
value=$(aws ssm get-parameters --names parameter-name --with-decryption)
```

```
aws ssm send-command \
    --name AWS-JoinDomain \
    --parameters password=$value \
    --instance-id instance-id
```

Windows

```
aws ssm send-command ^
    --name AWS-JoinDomain ^
    --parameters password=$value ^
    --instance-id instance-id
```

Powershell

```
$secure = (Get-SSMParameterValue -Names parameter-name -WithDecryption
 $True).Parameters[0].Value | ConvertTo-SecureString -AsPlainText -Force
```

```
$cred = New-Object System.Management.Automation.PSCredential -
argumentlist user-name,$secure
```

在 Parameter Store 中为 Amazon Machine Image ID 使用本机参数支持

创建 `String` 参数时，您可以将数据类型指定为 `aws:ec2:image`，以确保所输入的参数值为有效的 Amazon Machine Image (AMI) ID 格式。

通过支持 AMI ID 格式，您可以避免每次要在流程中使用的 AMI 发生更改时都使用新 ID 来更新所有脚本和模板。您可以创建数据类型为 `aws:ec2:image` 的参数，并输入 AMI ID 作为其值。这是您要从中创建新实例的 AMI。然后在模板、命令和脚本中引用此参数。

例如，您可以在运行 Amazon Elastic Compute Cloud (Amazon EC2) `run-instances` 命令时指定包含首选 AMI ID 的参数。

> **Note**
>
> 运行此命令的用户必须具有包含 `ssm:GetParameters` API 操作的 AWS Identity and Access Management (IAM) 权限，才能验证参数值。否则，参数创建过程将失败。

Linux & macOS

```
aws ec2 run-instances \
    --image-id resolve:ssm:/golden-ami \
    --count 1 \
    --instance-type t2.micro \
    --key-name my-key-pair \
    --security-groups my-security-group
```

Windows

```
aws ec2 run-instances ^
    --image-id resolve:ssm:/golden-ami ^
    --count 1 ^
    --instance-type t2.micro ^
    --key-name my-key-pair ^
    --security-groups my-security-group
```

您也可以在使用 Amazon EC2 控制台创建实例时选择首选 AMI。有关更多信息，请参阅《Amazon EC2 用户指南》中的使用 Systems Manager 参数查找 AMI。

当在实例创建工作流中使用其他 AMI 时，您只需使用新的 AMI 值更新参数即可，然后 Parameter Store 会再次验证您所输入 ID 的格式是否正确。

授予创建 **aws:ec2:image** 数据类型参数的权限

使用 AWS Identity and Access Management (IAM) 策略，您可以提供或限制用户对 Parameter Store API 操作和内容的访问权限。

要创建 `aws:ec2:image` 数据类型参数，用户必须同时拥有 `ssm:PutParameter` 和 `ec2:DescribeImages` 权限。

以下示例策略授予用户为 `aws:ec2:image` 调用 `PutParameter` API 操作的权限。因此，用户可以向系统添加数据类型为 `aws:ec2:image` 的参数。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ssm:PutParameter",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeImages",
            "Resource": "*"
        }
    ]
}
```

AMI 格式验证的工作原理

在将 `aws:ec2:image` 指定为参数的数据类型时，Systems Manager 不会立即创建参数。而是执行异步验证操作，以确保参数值满足 AMI ID 的格式设置要求，并且指定的 AMI 在您的 AWS 账户中可用。

在验证操作完成之前，可能会生成参数版本号。即使生成了参数版本号，操作也可能无法完成。

要监控参数是否已成功创建，我们建议使用 Amazon EventBridge 向您发送有关 `create` 和 `update` 参数操作的通知。这些通知会报告参数操作是否成功。如果操作失败，通知会包含一条错误消息，指出失败的原因。

```
{
    "version": "0",
    "id": "eed4a719-0fa4-6a49-80d8-8ac65EXAMPLE",
    "detail-type": "Parameter Store Change",
    "source": "aws.ssm",
    "account": "111122223333",
    "time": "2020-05-26T22:04:42Z",
    "region": "us-east-2",
    "resources": [
        "arn:aws:ssm:us-east-2:111122223333:parameter/golden-ami"
    ],
    "detail": {
        "exception": "Unable to Describe Resource",
        "dataType": "aws:ec2:image",
        "name": "golden-ami",
        "type": "String",
        "operation": "Create"
    }
}
```

有关在 Eventbridge 中订阅 Parameter Store 事件的信息，请参阅 基于 Parameter Store 事件设置通知或触发操作。

从 Parameter Store 中删除参数

本主题介绍如何删除在 Parameter Store（AWS Systems Manager 中的一项工具）中创建的参数。

> **Warning**
>
> 删除参数会移除该参数的所有版本。删除后，该参数及其版本将无法还原。

若要使用控制台删除参数

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Parameter Store。

3. 在 My parameters（我的参数）选项卡中，选中要删除的各参数旁边的复选框。
4. 选择删除。
5. 在确认对话框中，选择 Delete parameters（删除参数）。

若要使用 AWS CLI 删除参数

- 运行以下命令：

```
aws ssm delete-parameter --name "my-parameter"
```

将 *my-parameter* 替换为待删除参数的名称。

有关可以与 `delete-parameter` 命令结合使用的所有选项的信息，请参阅《AWS CLI Command Reference》中 AWS Systems Manager 部分的 delete-parameter。

使用 Parameter Store 中的公有参数

一些 AWS 服务将发布有关常见构件的信息，并将其发布为 AWS Systems Manager 公有参数。例如，Amazon Elastic Compute Cloud (Amazon EC2) 服务将有关 Amazon Machine Images (AMIs) 的信息作为公有参数发布。

本指南中的主题

- 发现 Parameter Store 中的公有参数
- 调用 Parameter Store 中的 AMI 公有参数
- 调用 Parameter Store 中的 ECS 优化 AMI 公有参数
- 调用 Parameter Store 中的 EKS 优化 AMI 公有参数
- 调用 AWS 服务、区域、端点、可用区、本地区域以及 Parameter Store 中 Wavelength 区域的公有参数

相关 AWS 博客文章

- 使用 AWS Systems ManagerParameter Store 查询 AWS 区域、端点等
- 使用 AWS Systems ManagerParameter Store 查询最新的 Amazon Linux AMI ID
- 使用 AWS Systems ManagerParameter Store 查询最新的 Windows AMI

发现 Parameter Store 中的公有参数

您可以使用 Parameter Store 控制台或 AWS Command Line Interface 搜索公有参数。

公有参数名称以 `aws/service/list` 开头。名称中接下来的部分对应于拥有该参数的服务。

以表列出了一些提供公有参数的 AWS 服务和其他资源：

- `ami-amazon-linux-latest`
- `ami-windows-latest`
- `ec2-macos`
- `appmesh`
- `aws-for-fluent-bit`
- `aws-sdk-pandas`
- `bottlerocket`
- `canonical`
- `cloud9`
- `datasync`
- `debian`
- `deeplearning`
- `ecs`
- `eks`
- `fis`
- `freebsd`
- `global-infrastructure`
- `marketplace`
- `neuron`
- `powertools`
- `sagemaker-distribution`
- `storagegateway`

并非所有公有参数都会发布到每个 AWS 区域。

使用 Parameter Store 控制台查找公有参数

您必须在您的 AWS 账户和 AWS 区域中至少有一个参数，然后才能使用控制台搜索公有参数。

要使用控制台查找公有参数，请执行以下步骤：

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Parameter Store。
3. 选择 Public parameters (公有参数) 选项卡。
4. 选择 Select a service (选择服务) 下拉菜单。选择要使用其参数的服务。
5. （可选）在搜索栏中输入更多信息，以筛选所选服务拥有的参数。
6. 选择要使用的公有参数。

使用 AWS CLI 查找公有参数

使用 `describe-parameters` 发现公有参数。

使用 `get-parameters-by-path` 获取 `/aws/service/list` 下列出的服务的实际路径。要获取服务的路径，从路径中删除 `/list`。例如，`/aws/service/list/ecs` 改为 `/aws/service/ecs`。

要在 Parameter Store 中检索不同服务拥有的公有参数列表，请运行以下命令。

```
aws ssm get-parameters-by-path --path /aws/service/list
```

此命令会返回如下信息。此示例输出已由于空间问题截断。

```
{
    "Parameters": [
        {
            "Name": "/aws/service/list/ami-al-latest",
            "Type": "String",
            "Value": "/aws/service/ami-al-latest/",
            "Version": 1,
            "LastModifiedDate": "2021-01-29T10:25:10.902000-08:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/list/ami-al-latest",
            "DataType": "text"
        },
        {
            "Name": "/aws/service/list/ami-windows-latest",
```

```
            "Type": "String",
            "Value": "/aws/service/ami-windows-latest/",
            "Version": 1,
            "LastModifiedDate": "2021-01-29T10:25:12.567000-08:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/list/ami-windows-
latest",
            "DataType": "text"
        },
        {
            "Name": "/aws/service/list/aws-storage-gateway-latest",
            "Type": "String",
            "Value": "/aws/service/aws-storage-gateway-latest/",
            "Version": 1,
            "LastModifiedDate": "2021-01-29T10:25:09.903000-08:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/list/aws-storage-
gateway-latest",
            "DataType": "text"
        },
        {
            "Name": "/aws/service/list/global-infrastructure",
            "Type": "String",
            "Value": "/aws/service/global-infrastructure/",
            "Version": 1,
            "LastModifiedDate": "2021-01-29T10:25:11.901000-08:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/list/global-
infrastructure",
            "DataType": "text"
        }
    ]
}
```

如果要查看特定服务拥有的参数，请从运行之前的命令后生成的列表中选择该服务。然后使用所需服务的名称调用 `get-parameters-by-path`。

例如，`/aws/service/global-infrastructure`。路径可以只有一级（只调用与给定值完全一致的参数），也可以是递归路径（路径中包含超出给定值的元素）。

> **Note**
>
> 并非所有区域都支持查询 `/aws/service/global-infrastructure` 路径。有关信息，请参阅[调用 AWS 服务、区域、端点、可用区、本地区域以及 Parameter Store 中 Wavelength 区域的公有参数]。

如果没有返回指定服务的任何结果，请添加 --recursive 标记并再次运行命令。

```
aws ssm get-parameters-by-path --path /aws/service/global-infrastructure
```

这将返回 global-infrastructure 拥有的所有参数。示例如下：

```
{
    "Parameters": [
        {
            "Name": "/aws/service/global-infrastructure/current-region",
            "Type": "String",
            "LastModifiedDate": "2019-06-21T05:15:34.252000-07:00",
            "Version": 1,
            "Tier": "Standard",
            "Policies": [],
            "DataType": "text"
        },
        {
            "Name": "/aws/service/global-infrastructure/version",
            "Type": "String",
            "LastModifiedDate": "2019-02-04T06:59:32.875000-08:00",
            "Version": 1,
            "Tier": "Standard",
            "Policies": [],
            "DataType": "text"
        }
    ]
}
```

您还可以使用 Option:BeginsWith 筛选条件查看特定服务拥有的参数。

```
aws ssm describe-parameters --parameter-filters "Key=Name, Option=BeginsWith, Values=/
aws/service/ami-amazon-linux-latest"
```

此命令会返回如下信息。此示例输出已由于空间问题截断。

```
{
    "Parameters": [
        {
            "Name": "/aws/service/ami-amazon-linux-latest/amzn-ami-hvm-x86_64-ebs",
            "Type": "String",
            "LastModifiedDate": "2021-01-26T13:39:40.686000-08:00",
```

```
            "Version": 25,
            "Tier": "Standard",
            "Policies": [],
            "DataType": "text"
        },
        {
            "Name": "/aws/service/ami-amazon-linux-latest/amzn-ami-hvm-x86_64-gp2",
            "Type": "String",
            "LastModifiedDate": "2021-01-26T13:39:40.807000-08:00",
            "Version": 25,
            "Tier": "Standard",
            "Policies": [],
            "DataType": "text"
        },
        {
            "Name": "/aws/service/ami-amazon-linux-latest/amzn-ami-hvm-x86_64-s3",
            "Type": "String",
            "LastModifiedDate": "2021-01-26T13:39:40.920000-08:00",
            "Version": 25,
            "Tier": "Standard",
            "Policies": [],
            "DataType": "text"
        }
    ]
}
```

> **Note**
>
> 使用 `Option=BeginsWith` 时，返回的参数可能会有所不同，因为它使用不同的搜索模式。

调用 Parameter Store 中的 AMI 公有参数

Amazon Elastic Compute Cloud（Amazon EC2）Amazon Machine Image（AMI）公有参数在 Amazon Linux 1、Amazon Linux 2、Amazon Linux 2023（AL2023）、macOS 和 Windows Server 上可通过以下路径获取：

- Amazon Linux 1、Amazon Linux 2 和 Amazon Linux 2023：`/aws/service/ami-amazon-linux-latest`
- macOS: `/aws/service/ec2-macos`
- Windows Server: `/aws/service/ami-windows-latest`

为 Amazon Linux 1、Amazon Linux 2 和 Amazon Linux 2023 调用 AMI 公有参数

您可以在 AWS Command Line Interface（AWS CLI）中使用以下命令查看当前 AWS 区域 中所有 Amazon Linux 1、Amazon Linux 2 和 Amazon Linux 2023（AL2023）AMIs 的列表。

Linux & macOS

```
aws ssm get-parameters-by-path \
    --path /aws/service/ami-amazon-linux-latest \
    --query 'Parameters[].Name'
```

Windows

```
aws ssm get-parameters-by-path ^
    --path /aws/service/ami-amazon-linux-latest ^
    --query Parameters[].Name
```

此命令会返回如下信息。

```
[
    "/aws/service/ami-amazon-linux-latest/al2023-ami-kernel-6.1-arm64",
    "/aws/service/ami-amazon-linux-latest/al2023-ami-kernel-6.1-x86_64",
    "/aws/service/ami-amazon-linux-latest/al2023-ami-minimal-kernel-6.1-arm64",
    "/aws/service/ami-amazon-linux-latest/al2023-ami-minimal-kernel-6.1-x86_64",
    "/aws/service/ami-amazon-linux-latest/al2023-ami-minimal-kernel-default-arm64",
    "/aws/service/ami-amazon-linux-latest/amzn-ami-hvm-x86_64-gp2",
    "/aws/service/ami-amazon-linux-latest/amzn-ami-hvm-x86_64-s3",
    "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-ebs",
    "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
    "/aws/service/ami-amazon-linux-latest/amzn2-ami-kernel-5.10-hvm-x86_64-ebs",
    "/aws/service/ami-amazon-linux-latest/al2023-ami-kernel-default-arm64",
    "/aws/service/ami-amazon-linux-latest/al2023-ami-kernel-default-x86_64",
    "/aws/service/ami-amazon-linux-latest/al2023-ami-minimal-kernel-default-x86_64",
    "/aws/service/ami-amazon-linux-latest/amzn-ami-hvm-x86_64-ebs",
    "/aws/service/ami-amazon-linux-latest/amzn-ami-minimal-hvm-x86_64-ebs",
    "/aws/service/ami-amazon-linux-latest/amzn-ami-minimal-hvm-x86_64-s3",
    "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-arm64-gp2",
    "/aws/service/ami-amazon-linux-latest/amzn2-ami-kernel-5.10-hvm-arm64-gp2",
    "/aws/service/ami-amazon-linux-latest/amzn2-ami-kernel-5.10-hvm-x86_64-gp2",
    "/aws/service/ami-amazon-linux-latest/amzn2-ami-minimal-hvm-arm64-ebs",
    "/aws/service/ami-amazon-linux-latest/amzn2-ami-minimal-hvm-x86_64-ebs"
```

```
]
```

您可以使用以下命令查看有关这些 AMIs 的详细信息，包括 AMI ID 和 Amazon Resource Name (ARN)。

Linux & macOS

```
aws ssm get-parameters-by-path \
    --path "/aws/service/ami-amazon-linux-latest" \
    --region region
```

Windows

```
aws ssm get-parameters-by-path ^
    --path "/aws/service/ami-amazon-linux-latest" ^
    --region region
```

region 表示 AWS Systems Manager 支持的 AWS 区域 的标识符，例如 us-east-2 对应美国东部（俄亥俄）区域。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

此命令会返回如下信息。此示例输出已由于空间问题截断。

```
{
    "Parameters": [
         {
            "Name": "/aws/service/ami-amazon-linux-latest/al2023-ami-kernel-6.1-arm64",
            "Type": "String",
            "Value": "ami-0b1b8b24a6c8e5d8b",
            "Version": 69,
            "LastModifiedDate": "2024-03-13T14:05:09.583000-04:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ami-amazon-linux-
latest/al2023-ami-kernel-6.1-arm64",
            "DataType": "text"
        },
        {
            "Name": "/aws/service/ami-amazon-linux-latest/al2023-ami-kernel-6.1-
x86_64",
            "Type": "String",
            "Value": "ami-0e0bf53f6def86294",
            "Version": 69,
```

```
            "LastModifiedDate": "2024-03-13T14:05:09.890000-04:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ami-amazon-linux-
latest/al2023-ami-kernel-6.1-x86_64",
            "DataType": "text"
        },
        {
            "Name": "/aws/service/ami-amazon-linux-latest/al2023-ami-minimal-
kernel-6.1-arm64",
            "Type": "String",
            "Value": "ami-09951bb66f9e5b5a5",
            "Version": 69,
            "LastModifiedDate": "2024-03-13T14:05:10.197000-04:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ami-amazon-linux-
latest/al2023-ami-minimal-kernel-6.1-arm64",
            "DataType": "text"
        }
    ]
}
```

您可以使用具有完整 AMI 名称的 GetParameters API 操作查看特定 AMI 的详细信息，包括路径。以下是一个示例命令。

Linux & macOS

```
aws ssm get-parameters \
    --names /aws/service/ami-amazon-linux-latest/al2023-ami-kernel-6.1-arm64 \
    --region us-east-2
```

Windows

```
aws ssm get-parameters ^
    --names /aws/service/ami-amazon-linux-latest/al2023-ami-kernel-6.1-arm64 ^
    --region us-east-2
```

该命令将返回以下信息。

```
{
    "Parameters": [
        {
            "Name": "/aws/service/ami-amazon-linux-latest/al2023-ami-kernel-6.1-arm64",
            "Type": "String",
```

```
            "Value": "ami-0b1b8b24a6c8e5d8b",
            "Version": 69,
            "LastModifiedDate": "2024-03-13T14:05:09.583000-04:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ami-amazon-linux-
latest/al2023-ami-kernel-6.1-arm64",
            "DataType": "text"
        }
    ],
    "InvalidParameters": []
}
```

为 macOS 调用 AMI 公有参数

您可以在 AWS CLI 中使用以下命令查看当前 macOS中所有 AWS 区域 AMIs 的列表。

Linux & macOS

```
aws ssm get-parameters-by-path \
    --path /aws/service/ec2-macos\
    --query 'Parameters[].Name'
```

Windows

```
aws ssm get-parameters-by-path ^
    --path /aws/service/ec2-macos ^
    --query Parameters[].Name
```

此命令会返回如下信息。

```
[
    "/aws/service/ec2-macos/sonoma/x86_64_mac/latest/image_id",
    "/aws/service/ec2-macos/ventura/x86_64_mac/latest/image_id",
    "/aws/service/ec2-macos/monterey/x86_64_mac/latest/image_id",
    "/aws/service/ec2-macos/sonoma/arm64_mac/latest/image_id",
    "/aws/service/ec2-macos/ventura/arm64_mac/latest/image_id",
    "/aws/service/ec2-macos/monterey/arm64_mac/latest/image_id"
]
```

您可以使用以下命令查看有关这些 AMIs 的详细信息，包括 AMI ID 和 Amazon Resource Name (ARN)。

Linux & macOS

```
aws ssm get-parameters-by-path \
    --path "/aws/service/ec2-macos" \
    --region region
```

Windows

```
aws ssm get-parameters-by-path ^
    --path "/aws/service/ec2-macos" ^
    --region region
```

region 表示 AWS Systems Manager 支持的 AWS 区域 的标识符，例如 us-east-2 对应美国东部（俄亥俄）区域。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

此命令会返回如下信息。此示例输出已由于空间问题截断。

```
{
    "Parameters": [
        ...sample results pending...
    ]
}
```

您可以使用具有完整 AMI 名称的 GetParameters API 操作查看特定 AMI 的详细信息，包括路径。以下是一个示例命令。

Linux & macOS

```
aws ssm get-parameters \
    --names /aws/service/ec2-macos/...pending... \
    --region us-east-2
```

Windows

```
aws ssm get-parameters ^
     --names /aws/service/ec2-macos/...pending... ^
    --region us-east-2
```

该命令将返回以下信息。

```
{
    "Parameters": [
         ...sample results pending...
    ],
    "InvalidParameters": []
}
```

为 Windows Server 调用 AMI 公有参数

您可以在 AWS CLI 中使用以下命令查看当前 Windows Server中所有 AWS 区域 AMIs 的列表。

Linux & macOS

```
aws ssm get-parameters-by-path \
    --path /aws/service/ami-windows-latest \
    --query 'Parameters[].Name'
```

Windows

```
aws ssm get-parameters-by-path ^
    --path /aws/service/ami-windows-latest ^
    --query Parameters[].Name
```

此命令会返回如下信息。此示例输出已由于空间问题截断。

```
[
    "/aws/service/ami-windows-latest/EC2LaunchV2-Windows_Server-2016-English-Full-
Base",
    "/aws/service/ami-windows-latest/Windows_Server-2016-English-Full-
SQL_2014_SP3_Enterprise",
    "/aws/service/ami-windows-latest/Windows_Server-2016-German-Full-Base",
    "/aws/service/ami-windows-latest/Windows_Server-2016-Japanese-Full-
SQL_2016_SP3_Standard",
    "/aws/service/ami-windows-latest/Windows_Server-2016-Japanese-Full-SQL_2017_Web",
    "/aws/service/ami-windows-latest/Windows_Server-2019-English-Core-
EKS_Optimized-1.25",
    "/aws/service/ami-windows-latest/Windows_Server-2019-Italian-Full-Base",
    "/aws/service/ami-windows-latest/Windows_Server-2022-Japanese-Full-
SQL_2019_Enterprise",
```

```
    "/aws/service/ami-windows-latest/Windows_Server-2022-Portuguese_Brazil-Full-Base",
    "/aws/service/ami-windows-latest/amzn2-ami-hvm-2.0.20191217.0-x86_64-gp2-mono",
    "/aws/service/ami-windows-latest/Windows_Server-2016-English-Deep-Learning",
    "/aws/service/ami-windows-latest/Windows_Server-2016-Japanese-Full-
SQL_2016_SP3_Web",
    "/aws/service/ami-windows-latest/Windows_Server-2016-Korean-Full-Base",
    "/aws/service/ami-windows-latest/Windows_Server-2019-English-STIG-Core",
    "/aws/service/ami-windows-latest/Windows_Server-2019-French-Full-Base",
    "/aws/service/ami-windows-latest/Windows_Server-2019-Japanese-Full-
SQL_2017_Enterprise",
    "/aws/service/ami-windows-latest/Windows_Server-2019-Korean-Full-Base",
    "/aws/service/ami-windows-latest/Windows_Server-2022-English-Full-SQL_2022_Web",
    "/aws/service/ami-windows-latest/Windows_Server-2022-Italian-Full-Base",
    "/aws/service/ami-windows-latest/amzn2-x86_64-SQL_2019_Express",
    "/aws/service/ami-windows-latest/EC2LaunchV2-Windows_Server-2016-English-Core-
Base",
    "/aws/service/ami-windows-latest/Windows_Server-2016-English-Full-
SQL_2019_Enterprise",
    "/aws/service/ami-windows-latest/Windows_Server-2016-English-Full-
SQL_2019_Standard",
    "/aws/service/ami-windows-latest/Windows_Server-2016-Portuguese_Portugal-Full-
Base",
    "/aws/service/ami-windows-latest/Windows_Server-2019-English-Core-
EKS_Optimized-1.24",
    "/aws/service/ami-windows-latest/Windows_Server-2019-English-Deep-Learning",
    "/aws/service/ami-windows-latest/Windows_Server-2019-English-Full-SQL_2017_Web",
    "/aws/service/ami-windows-latest/Windows_Server-2019-Hungarian-Full-Base
]
```

您可以使用以下命令查看有关这些 AMIs 的详细信息，包括 AMI ID 和 Amazon Resource Name (ARN)。

Linux & macOS

```
aws ssm get-parameters-by-path \
    --path "/aws/service/ami-windows-latest" \
    --region region
```

Windows

```
aws ssm get-parameters-by-path ^
    --path "/aws/service/ami-windows-latest" ^
```

```
    --region region
```

region 表示 AWS Systems Manager 支持的 AWS 区域 的标识符，例如 us-east-2 对应美国东部（俄亥俄）区域。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

此命令会返回如下信息。此示例输出已由于空间问题截断。

```
{
    "Parameters": [
        {
            "Name": "/aws/service/ami-windows-latest/EC2LaunchV2-Windows_Server-2016-
English-Full-Base",
            "Type": "String",
            "Value": "ami-0a30b2e65863e2d16",
            "Version": 36,
            "LastModifiedDate": "2024-03-15T15:58:37.976000-04:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ami-windows-latest/
EC2LaunchV2-Windows_Server-2016-English-Full-Base",
            "DataType": "text"
        },
        {
            "Name": "/aws/service/ami-windows-latest/Windows_Server-2016-English-Full-
SQL_2014_SP3_Enterprise",
            "Type": "String",
            "Value": "ami-001f20c053dd120ce",
            "Version": 69,
            "LastModifiedDate": "2024-03-15T15:53:58.905000-04:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ami-windows-latest/
Windows_Server-2016-English-Full-SQL_2014_SP3_Enterprise",
            "DataType": "text"
        },
        {
            "Name": "/aws/service/ami-windows-latest/Windows_Server-2016-German-Full-
Base",
            "Type": "String",
            "Value": "ami-063be4935453e94e9",
            "Version": 102,
            "LastModifiedDate": "2024-03-15T15:51:12.003000-04:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ami-windows-latest/
Windows_Server-2016-German-Full-Base",
            "DataType": "text"
```

```
        }
    ]
}
```

您可以使用具有完整 AMI 名称的 GetParameters API 操作查看特定 AMI 的详细信息，包括路径。以下是一个示例命令。

Linux & macOS

```
aws ssm get-parameters \
    --names /aws/service/ami-windows-latest/EC2LaunchV2-Windows_Server-2016-English-
Full-Base \
    --region us-east-2
```

Windows

```
aws ssm get-parameters ^
    --names /aws/service/ami-windows-latest/EC2LaunchV2-Windows_Server-2016-English-
Full-Base ^
    --region us-east-2
```

该命令将返回以下信息。

```
{
    "Parameters": [
        {
            "Name": "/aws/service/ami-windows-latest/EC2LaunchV2-Windows_Server-2016-
English-Full-Base",
            "Type": "String",
            "Value": "ami-0a30b2e65863e2d16",
            "Version": 36,
            "LastModifiedDate": "2024-03-15T15:58:37.976000-04:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ami-windows-latest/
EC2LaunchV2-Windows_Server-2016-English-Full-Base",
            "DataType": "text"
        }
    ],
    "InvalidParameters": []
}
```

调用 Parameter Store 中的 ECS 优化 AMI 公有参数

Amazon Elastic Container Service (Amazon ECS) 服务将最新的 Amazon ECS 优化 Amazon Machine Images (AMIs) 的名称作为公有参数发布。建议用户在为 Amazon ECS 创建新 Amazon Elastic Compute Cloud（Amazon EC2）集群时使用此 AMI，因为优化的 AMIs 包括错误修复和功能更新。

使用以下命令查看适用于 Amazon Linux 2 的最新 Amazon ECS 优化 AMI 的名称。要查看其他操作系统的命令，请参阅 Amazon Elastic Container Service Developer Guide 中的检索 Amazon ECS 优化的 AMI 元数据。

Linux & macOS

```
aws ssm get-parameters \
    --names /aws/service/ecs/optimized-ami/amazon-linux-2/recommended
```

Windows

```
aws ssm get-parameters ^
    --names /aws/service/ecs/optimized-ami/amazon-linux-2/recommended
```

此命令会返回如下信息。

```
{
    "Parameters": [
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/recommended",
            "Type": "String",
            "Value": "{\"schema_version\":1,\"image_name\":\"amzn2-ami-ecs-
hvm-2.0.20210929-x86_64-ebs\",\"image_id\":\"ami-0c38a2329ed4dae9a\",\"os\":\"Amazon
 Linux 2\",\"ecs_runtime_version\":\"Docker version 20.10.7\",\"ecs_agent_version\":
\"1.55.4\"}",
            "Version": 73,
            "LastModifiedDate": "2021-10-06T16:35:10.004000-07:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/recommended",
            "DataType": "text"
        }
    ],
    "InvalidParameters": []
}
```

调用 Parameter Store 中的 EKS 优化 AMI 公有参数

Amazon Elastic Kubernetes Service (Amazon EKS) 服务将最新的 Amazon EKS 优化 Amazon Machine Image (AMI) 的名称作为公有参数发布。建议您在向 Amazon EKS 集群添加节点时使用此 AMI，因为新版本包括 Kubernetes 补丁和安全更新。以前，为了保证您使用最新的 AMI，您需要检查 Amazon EKS 文档，并使用新的 AMI ID 手动更新任何部署模板或资源。

使用以下命令查看适用于 Amazon Linux 2 的最新 Amazon EKS 优化 AMI 的名称。

Linux & macOS

```
aws ssm get-parameters \
    --names /aws/service/eks/optimized-ami/1.14/amazon-linux-2/recommended
```

Windows

```
aws ssm get-parameters ^
    --names /aws/service/eks/optimized-ami/1.14/amazon-linux-2/recommended
```

此命令会返回如下信息。

```
{
    "Parameters": [
        {
            "Name": "/aws/service/eks/optimized-ami/1.14/amazon-linux-2/recommended",
            "Type": "String",
            "Value": "{\"schema_version\":\"2\",\"image_id\":\"ami-08984d8491de17ca0\",
\"image_name\":\"amazon-eks-node-1.14-v20201007\",\"release_version\":
\"1.14.9-20201007\"}",
            "Version": 24,
            "LastModifiedDate": "2020-11-17T10:16:09.971000-08:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/eks/optimized-
ami/1.14/amazon-linux-2/recommended",
            "DataType": "text"
        }
    ],
    "InvalidParameters": []
}
```

调用 AWS 服务、区域、端点、可用区、本地区域以及 Parameter Store 中 Wavelength 区域的公有参数

您可以使用以下路径调用 AWS 区域、服务、端点、可用区和 Wavelength 区域的公有参数。

`/aws/service/global-infrastructure`

> **Note**
>
> 目前，仅在以下 AWS 区域 中支持查询路径 `/aws/service/global-infrastructure`：
>
> - 美国东部（弗吉尼亚州北部）(us-east-1)
> - 美国东部（俄亥俄州）(us-east-2)
> - 美国西部（加利福尼亚北部）(us-west-1)
> - 美国西部（俄勒冈州）(us-west-2)
> - 亚太地区（香港）(ap-east-1)
> - 亚太地区（孟买）(ap-south-1)
> - 亚太地区（首尔）(ap-northeast-2)
> - 亚太地区（新加坡）(ap-southeast-1)
> - 亚太地区（悉尼）(ap-southeast-2)
> - 亚太地区（东京）(ap-northeast-1)
> - 加拿大（中部）(ca-central-1)
> - 欧洲地区（法兰克福）(eu-central-1)
> - 欧洲地区（爱尔兰）(eu-west-1)
> - 欧洲（伦敦）(eu-west-2)
> - 欧洲地区（巴黎）（eu-west-3）
> - 欧洲地区（斯德哥尔摩）(eu-north-1)
> - 南美洲（圣保罗）（sa-east-1）
>
> 如果您在其他商业区域工作，则可以在查询中指定支持的区域以查看结果。例如，如果您在加拿大西部（卡尔加里）（ca-west-1）区域工作，则可以在查询中指定加拿大（中部）（ca-central-1）：
>
> ```
> aws ssm get-parameters-by-path \
>     --path /aws/service/global-infrastructure/regions \
> ```

```
    --region ca-central-1
```

查看处于活动状态的 AWS 区域

您可以在 AWS Command Line Interface (AWS CLI) 中使用以下命令查看所有活动 AWS 区域的列表。

Linux & macOS

```
aws ssm get-parameters-by-path \
    --path /aws/service/global-infrastructure/regions \
    --query 'Parameters[].Name'
```

Windows

```
aws ssm get-parameters-by-path ^
    --path /aws/service/global-infrastructure/regions ^
    --query Parameters[].Name
```

此命令会返回如下信息。

```
[
    "/aws/service/global-infrastructure/regions/af-south-1",
    "/aws/service/global-infrastructure/regions/ap-east-1",
    "/aws/service/global-infrastructure/regions/ap-northeast-3",
    "/aws/service/global-infrastructure/regions/ap-south-2",
    "/aws/service/global-infrastructure/regions/ca-central-1",
    "/aws/service/global-infrastructure/regions/eu-central-2",
    "/aws/service/global-infrastructure/regions/eu-west-2",
    "/aws/service/global-infrastructure/regions/eu-west-3",
    "/aws/service/global-infrastructure/regions/us-east-1",
    "/aws/service/global-infrastructure/regions/us-gov-west-1",
    "/aws/service/global-infrastructure/regions/ap-northeast-2",
    "/aws/service/global-infrastructure/regions/ap-southeast-1",
    "/aws/service/global-infrastructure/regions/ap-southeast-2",
    "/aws/service/global-infrastructure/regions/ap-southeast-3",
    "/aws/service/global-infrastructure/regions/cn-north-1",
    "/aws/service/global-infrastructure/regions/cn-northwest-1",
    "/aws/service/global-infrastructure/regions/eu-south-1",
    "/aws/service/global-infrastructure/regions/eu-south-2",
```

```
    "/aws/service/global-infrastructure/regions/us-east-2",
    "/aws/service/global-infrastructure/regions/us-west-1",
    "/aws/service/global-infrastructure/regions/ap-northeast-1",
    "/aws/service/global-infrastructure/regions/ap-south-1",
    "/aws/service/global-infrastructure/regions/ap-southeast-4",
    "/aws/service/global-infrastructure/regions/ca-west-1",
    "/aws/service/global-infrastructure/regions/eu-central-1",
    "/aws/service/global-infrastructure/regions/il-central-1",
    "/aws/service/global-infrastructure/regions/me-central-1",
    "/aws/service/global-infrastructure/regions/me-south-1",
    "/aws/service/global-infrastructure/regions/sa-east-1",
    "/aws/service/global-infrastructure/regions/us-gov-east-1",
    "/aws/service/global-infrastructure/regions/eu-north-1",
    "/aws/service/global-infrastructure/regions/eu-west-1",
    "/aws/service/global-infrastructure/regions/us-west-2"
]
```

查看可用的 AWS 服务

您可以使用以下命令查看所有可用 AWS 服务的完整列表，并将其按字母顺序排序。此示例输出已由于空间问题截断。

Linux & macOS

```
aws ssm get-parameters-by-path \
    --path /aws/service/global-infrastructure/services \
    --query 'Parameters[].Name | sort(@)'
```

Windows

```
aws ssm get-parameters-by-path ^
    --path /aws/service/global-infrastructure/services ^
    --query "Parameters[].Name | sort(@)"
```

此命令会返回如下信息。此示例输出已由于空间问题截断。

```
[
     "/aws/service/global-infrastructure/services/accessanalyzer",
    "/aws/service/global-infrastructure/services/account",
    "/aws/service/global-infrastructure/services/acm",
    "/aws/service/global-infrastructure/services/acm-pca",
```

```
    "/aws/service/global-infrastructure/services/ahl",
    "/aws/service/global-infrastructure/services/aiq",
    "/aws/service/global-infrastructure/services/amazonlocationservice",
    "/aws/service/global-infrastructure/services/amplify",
    "/aws/service/global-infrastructure/services/amplifybackend",
    "/aws/service/global-infrastructure/services/apigateway",
    "/aws/service/global-infrastructure/services/apigatewaymanagementapi",
    "/aws/service/global-infrastructure/services/apigatewayv2",
    "/aws/service/global-infrastructure/services/appconfig",
    "/aws/service/global-infrastructure/services/appconfigdata",
    "/aws/service/global-infrastructure/services/appflow",
    "/aws/service/global-infrastructure/services/appintegrations",
    "/aws/service/global-infrastructure/services/application-autoscaling",
    "/aws/service/global-infrastructure/services/application-insights",
    "/aws/service/global-infrastructure/services/applicationcostprofiler",
    "/aws/service/global-infrastructure/services/appmesh",
    "/aws/service/global-infrastructure/services/apprunner",
    "/aws/service/global-infrastructure/services/appstream",
    "/aws/service/global-infrastructure/services/appsync",
    "/aws/service/global-infrastructure/services/aps",
    "/aws/service/global-infrastructure/services/arc-zonal-shift",
    "/aws/service/global-infrastructure/services/artifact",
    "/aws/service/global-infrastructure/services/athena",
    "/aws/service/global-infrastructure/services/auditmanager",
    "/aws/service/global-infrastructure/services/augmentedairuntime",
    "/aws/service/global-infrastructure/services/aurora",
    "/aws/service/global-infrastructure/services/autoscaling",
    "/aws/service/global-infrastructure/services/aws-appfabric",
    "/aws/service/global-infrastructure/services/awshealthdashboard",
```

查看支持某个 AWS 服务的区域

您可以查看提供服务的 AWS 区域列表。此示例使用 AWS Systems Manager (ssm)。

Linux & macOS

```
aws ssm get-parameters-by-path \
    --path /aws/service/global-infrastructure/services/ssm/regions \
    --query 'Parameters[].Value'
```

Windows

```
aws ssm get-parameters-by-path ^
```

```
    --path /aws/service/global-infrastructure/services/ssm/regions ^
    --query Parameters[].Value
```

此命令会返回如下信息。

```
[
    "ap-south-1",
    "eu-central-1",
    "eu-central-2",
    "eu-west-1",
    "eu-west-2",
    "eu-west-3",
    "il-central-1",
    "me-south-1",
    "us-east-2",
    "us-gov-west-1",
    "af-south-1",
    "ap-northeast-3",
    "ap-southeast-1",
    "ap-southeast-4",
    "ca-central-1",
    "ca-west-1",
    "cn-north-1",
    "eu-north-1",
    "eu-south-2",
    "us-west-1",
    "ap-east-1",
    "ap-northeast-1",
    "ap-northeast-2",
    "ap-southeast-2",
    "ap-southeast-3",
    "cn-northwest-1",
    "eu-south-1",
    "me-central-1",
    "us-gov-east-1",
    "us-west-2",
    "ap-south-2",
    "sa-east-1",
    "us-east-1"
]
```

查看服务的区域端点

您可以使用以下命令查看服务的区域端点。此命令会查询美国东部（俄亥俄州）（us-east-2）区域。

Linux & macOS

```
aws ssm get-parameter \
    --name /aws/service/global-infrastructure/regions/us-east-2/services/ssm/
endpoint \
    --query 'Parameter.Value'
```

Windows

```
aws ssm get-parameter ^
    --name /aws/service/global-infrastructure/regions/us-east-2/services/ssm/
endpoint ^
    --query Parameter.Value
```

此命令会返回如下信息。

```
"ssm.us-east-2.amazonaws.com"
```

查看完整的可用区详细信息

您可以使用以下命令查看可用区。

Linux & macOS

```
aws ssm get-parameters-by-path \
    --path /aws/service/global-infrastructure/availability-zones/
```

Windows

```
aws ssm get-parameters-by-path ^
    --path /aws/service/global-infrastructure/availability-zones/
```

此命令会返回如下信息。此示例输出已由于空间问题截断。

```
{
    "Parameters": [
```

```
        {
            "Name": "/aws/service/global-infrastructure/availability-zones/afs1-az3",
            "Type": "String",
            "Value": "afs1-az3",
            "Version": 1,
            "LastModifiedDate": "2020-04-21T12:05:35.375000-04:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/
availability-zones/afs1-az3",
            "DataType": "text"
        },
        {
            "Name": "/aws/service/global-infrastructure/availability-zones/aps1-az2",
            "Type": "String",
            "Value": "aps1-az2",
            "Version": 1,
            "LastModifiedDate": "2020-04-03T16:13:57.351000-04:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/
availability-zones/aps1-az2",
            "DataType": "text"
        },
        {
            "Name": "/aws/service/global-infrastructure/availability-zones/apse3-az1",
            "Type": "String",
            "Value": "apse3-az1",
            "Version": 1,
            "LastModifiedDate": "2021-12-13T08:51:38.983000-05:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/
availability-zones/apse3-az1",
            "DataType": "text"
        }
    ]
}
```

仅查看可用区名称

您可以使用以下命令仅查看可用区的名称。

Linux & macOS

```
aws ssm get-parameters-by-path \
    --path /aws/service/global-infrastructure/availability-zones \
    --query 'Parameters[].Name | sort(@)'
```

Windows

```
aws ssm get-parameters-by-path ^
    --path /aws/service/global-infrastructure/availability-zones ^
    --query "Parameters[].Name | sort(@)"
```

此命令会返回如下信息。此示例输出已由于空间问题截断。

```
[
    "/aws/service/global-infrastructure/availability-zones/afs1-az1",
    "/aws/service/global-infrastructure/availability-zones/afs1-az2",
    "/aws/service/global-infrastructure/availability-zones/afs1-az3",
    "/aws/service/global-infrastructure/availability-zones/ape1-az1",
    "/aws/service/global-infrastructure/availability-zones/ape1-az2",
    "/aws/service/global-infrastructure/availability-zones/ape1-az3",
    "/aws/service/global-infrastructure/availability-zones/apne1-az1",
    "/aws/service/global-infrastructure/availability-zones/apne1-az2",
    "/aws/service/global-infrastructure/availability-zones/apne1-az3",
    "/aws/service/global-infrastructure/availability-zones/apne1-az4"
```

查看单个区域中的可用区的名称

您可以使用以下命令查看一个区域（在此示例中为 us-east-2）中的可用区的名称。

Linux & macOS

```
aws ssm get-parameters-by-path \
    --path /aws/service/global-infrastructure/regions/us-east-2/availability-zones \
    --query 'Parameters[].Name | sort(@)'
```

Windows

```
aws ssm get-parameters-by-path ^
    --path /aws/service/global-infrastructure/regions/us-east-2/availability-zones ^
    --query "Parameters[].Name | sort(@)"
```

此命令会返回如下信息。

```
[
```

```
    "/aws/service/global-infrastructure/regions/us-east-2/availability-zones/use2-az1",
    "/aws/service/global-infrastructure/regions/us-east-2/availability-zones/use2-az2",
    "/aws/service/global-infrastructure/regions/us-east-2/availability-zones/use2-az3"
```

仅查看可用区 ARN

您可以使用以下命令仅查看可用区的 Amazon Resource Name (ARN)。

Linux & macOS

```
aws ssm get-parameters-by-path \
    --path /aws/service/global-infrastructure/availability-zones \
    --query 'Parameters[].ARN | sort(@)'
```

Windows

```
aws ssm get-parameters-by-path ^
    --path /aws/service/global-infrastructure/availability-zones ^
    --query "Parameters[].ARN | sort(@)"
```

此命令会返回如下信息。此示例输出已由于空间问题截断。

```
[
    "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/availability-
zones/afs1-az1",
    "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/availability-
zones/afs1-az2",
    "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/availability-
zones/afs1-az3",
    "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/availability-
zones/ape1-az1",
    "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/availability-
zones/ape1-az2",
    "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/availability-
zones/ape1-az3",
    "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/availability-
zones/apne1-az1",
```

查看本地区域详细信息

您可以使用以下命令查看本地区域。

Linux & macOS

```
aws ssm get-parameters-by-path \
    --path /aws/service/global-infrastructure/local-zones
```

Windows

```
aws ssm get-parameters-by-path ^
    --path /aws/service/global-infrastructure/local-zones
```

此命令会返回如下信息。此示例输出已由于空间问题截断。

```
{
    "Parameters": [
        {
            "Name": "/aws/service/global-infrastructure/local-zones/afs1-los1-az1",
            "Type": "String",
            "Value": "afs1-los1-az1",
            "Version": 1,
            "LastModifiedDate": "2023-01-25T11:53:11.690000-05:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/
local-zones/afs1-los1-az1",
            "DataType": "text"
        },
        {
            "Name": "/aws/service/global-infrastructure/local-zones/apne1-tpe1-az1",
            "Type": "String",
            "Value": "apne1-tpe1-az1",
            "Version": 1,
            "LastModifiedDate": "2024-03-15T12:35:41.076000-04:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/
local-zones/apne1-tpe1-az1",
            "DataType": "text"
        },
        {
            "Name": "/aws/service/global-infrastructure/local-zones/aps1-ccu1-az1",
            "Type": "String",
            "Value": "aps1-ccu1-az1",
            "Version": 1,
            "LastModifiedDate": "2022-12-19T11:34:43.351000-05:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/
local-zones/aps1-ccu1-az1",
```

```
            "DataType": "text"
        }
    ]
}
```

查看 Wavelength 区域详细信息

您可以使用以下命令查看 Wavelength 区域。

Linux & macOS

```
aws ssm get-parameters-by-path \
    --path /aws/service/global-infrastructure/wavelength-zones
```

Windows

```
aws ssm get-parameters-by-path ^
    --path /aws/service/global-infrastructure/wavelength-zones
```

此命令会返回如下信息。此示例输出已由于空间问题截断。

```
{
    "Parameters": [
        {
            "Name": "/aws/service/global-infrastructure/wavelength-zones/apne1-wl1-nrt-
wlz1",
            "Type": "String",
            "Value": "apne1-wl1-nrt-wlz1",
            "Version": 3,
            "LastModifiedDate": "2020-12-15T17:16:04.715000-05:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/
wavelength-zones/apne1-wl1-nrt-wlz1",
            "DataType": "text"
        },
        {
            "Name": "/aws/service/global-infrastructure/wavelength-zones/apne2-wl1-sel-
wlz1",
            "Type": "String",
            "Value": "apne2-wl1-sel-wlz1",
            "Version": 1,
            "LastModifiedDate": "2022-05-25T12:29:13.862000-04:00",
```

```
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/
wavelength-zones/apne2-wl1-sel-wlz1",
            "DataType": "text"
        },
        {
            "Name": "/aws/service/global-infrastructure/wavelength-zones/cac1-wl1-yto-
wlz1",
            "Type": "String",
            "Value": "cac1-wl1-yto-wlz1",
            "Version": 1,
            "LastModifiedDate": "2022-04-26T09:57:44.495000-04:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/
wavelength-zones/cac1-wl1-yto-wlz1",
            "DataType": "text"
        }
    ]
}
```

查看本地区域下的所有参数和值

您可以使用以下命令查看本地区域的所有参数数据。

Linux & macOS

```
aws ssm get-parameters-by-path \
    --path "/aws/service/global-infrastructure/local-zones/usw2-lax1-az1/"
```

Windows

```
aws ssm get-parameters-by-path ^
    --path "/aws/service/global-infrastructure/local-zones/use1-bos1-az1"
```

此命令会返回如下信息。此示例输出已由于空间问题截断。

```
{
    "Parameters": [
        {
            "Name": "/aws/service/global-infrastructure/local-zones/use1-bos1-az1/
geolocationCountry",
            "Type": "String",
            "Value": "US",
            "Version": 3,
```

```
            "LastModifiedDate": "2020-12-15T14:16:17.641000-08:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/
local-zones/use1-bos1-az1/geolocationCountry",
            "DataType": "text"
        },
        {
            "Name": "/aws/service/global-infrastructure/local-zones/use1-bos1-az1/
geolocationRegion",
            "Type": "String",
            "Value": "US-MA",
            "Version": 3,
            "LastModifiedDate": "2020-12-15T14:16:17.794000-08:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/
local-zones/use1-bos1-az1/geolocationRegion",
            "DataType": "text"
        },
        {
            "Name": "/aws/service/global-infrastructure/local-zones/use1-bos1-az1/
location",
            "Type": "String",
            "Value": "US East (Boston)",
            "Version": 1,
            "LastModifiedDate": "2021-01-11T10:53:24.634000-08:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/
local-zones/use1-bos1-az1/location",
            "DataType": "text"
        },
        {
            "Name": "/aws/service/global-infrastructure/local-zones/use1-bos1-az1/
network-border-group",
            "Type": "String",
            "Value": "us-east-1-bos-1",
            "Version": 3,
            "LastModifiedDate": "2020-12-15T14:16:20.641000-08:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/
local-zones/use1-bos1-az1/network-border-group",
            "DataType": "text"
        },
        {
            "Name": "/aws/service/global-infrastructure/local-zones/use1-bos1-az1/
parent-availability-zone",
            "Type": "String",
            "Value": "use1-az4",
            "Version": 3,
```

```
            "LastModifiedDate": "2020-12-15T14:16:20.834000-08:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/
local-zones/use1-bos1-az1/parent-availability-zone",
            "DataType": "text"
        },
        {
            "Name": "/aws/service/global-infrastructure/local-zones/use1-bos1-az1/
parent-region",
            "Type": "String",
            "Value": "us-east-1",
            "Version": 3,
            "LastModifiedDate": "2020-12-15T14:16:20.721000-08:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/
local-zones/use1-bos1-az1/parent-region",
            "DataType": "text"
        },
        {
            "Name": "/aws/service/global-infrastructure/local-zones/use1-bos1-az1/zone-
group",
            "Type": "String",
            "Value": "us-east-1-bos-1",
            "Version": 3,
            "LastModifiedDate": "2020-12-15T14:16:17.983000-08:00",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/global-infrastructure/
local-zones/use1-bos1-az1/zone-group",
            "DataType": "text"
        }
    ]
}
```

仅查看本地区域参数名称

您可以使用以下命令仅查看本地区域参数的名称。

Linux & macOS

```
aws ssm get-parameters-by-path \
    --path /aws/service/global-infrastructure/local-zones/usw2-lax1-az1 \
    --query 'Parameters[].Name | sort(@)'
```

Windows

```
aws ssm get-parameters-by-path ^
```

```
    --path /aws/service/global-infrastructure/local-zones/use1-bos1-az1 ^
    --query "Parameters[].Name | sort(@)"
```

此命令会返回如下信息。

```
[
    "/aws/service/global-infrastructure/local-zones/use1-bos1-az1/geolocationCountry",
    "/aws/service/global-infrastructure/local-zones/use1-bos1-az1/geolocationRegion",
    "/aws/service/global-infrastructure/local-zones/use1-bos1-az1/location",
    "/aws/service/global-infrastructure/local-zones/use1-bos1-az1/network-border-
group",
    "/aws/service/global-infrastructure/local-zones/use1-bos1-az1/parent-availability-
zone",
    "/aws/service/global-infrastructure/local-zones/use1-bos1-az1/parent-region",
    "/aws/service/global-infrastructure/local-zones/use1-bos1-az1/zone-group"
]
```

Parameter Store 演练

此部分中的演练向您展示了如何使用 Parameter Store（AWS Systems Manager 中的一项工具）在测试环境中创建、存储和运行参数。本演练展示了如何将 Parameter Store 与其他 Systems Manager 工具结合使用。您还可以将 Parameter Store 与其他 AWS 服务结合使用。有关更多信息，请参阅 什么是参数？。

内容

- 在 Parameter Store 中创建 SecureString 参数并将节点加入域（PowerShell）

在 Parameter Store 中创建 SecureString 参数并将节点加入域（PowerShell）

本演练演示了如何使用 AWS Systems Manager `SecureString` 参数和 Run Command 将 Windows Server 节点加入域。演练中使用了典型的域参数，如域名和域用户名。这些值作为未加密的字符串值传递。域密码使用 AWS 托管式密钥加密，并作为加密字符串传递。

先决条件

本演练假定您已在与 Amazon VPC 关联的 DHCP 选项集中指定域名和 DNS 服务器 IP 地址。有关更多信息，请参阅 Amazon VPC 用户指南中的使用 DHCP 选项集。

创建 **SecureString** 参数并将节点加入到域

1. 使用 AWS Tools for Windows PowerShell 将参数输入到系统中。

   在以下示例中，将每个*#######*替换为您自己的信息。

```
Write-SSMParameter -Name "domainName" -Value "DOMAIN-NAME" -Type String
Write-SSMParameter -Name "domainJoinUserName" -Value "DOMAIN\USERNAME" -Type String
Write-SSMParameter -Name "domainJoinPassword" -Value "PASSWORD" -Type SecureString
```

   > **Important**
   >
   > 只会加密 SecureString 参数的值。不会加密参数名称、描述和其他属性。

2. 将以下 AWS Identity and Access Management (IAM) 策略附加到节点的 IAM 角色权限：

   - AmazonSSMManagedInstanceCore – 必需。此 AWS 托管式策略允许托管式节点使用 Systems Manager 服务核心功能。
   - AmazonSSMDirectoryServiceAccess – 必需。此 AWS 托管式策略允许 SSM Agent 代表您访问 AWS Directory Service，以处理托管式节点加入域的请求。
   - 用于 S3 存储桶访问的自定义策略 – 必需。SSM Agent 位于您的节点上并执行 Systems Manager 任务，它需要访问 Amazon 拥有的特定 Amazon Simple Storage Service（Amazon S3）存储桶。在您创建的自定义 S3 存储桶策略中，您还会提供对您自己的 S3 存储桶的访问权限，这是执行 Systems Manager 操作所必需的。

     示例：您可以将 Run Command 命令或 Session Manager 会话输出写入到一个 S3 存储桶中，以后将使用该输出进行审核或故障排除。您将访问脚本或自定义补丁基准列表存储在一个 S3 存储桶中，并在运行命令或应用补丁基准时引用该脚本或列表。

     有关为 Amazon S3 存储桶访问创建自定义策略的信息，请参阅为实例配置文件创建自定义 S3 存储桶策略

     > **Note**
     >
     > 在 S3 存储桶中保存输出日志数据是可选功能，但如果您决定使用该功能，建议在开始执行 Systems Manager 配置过程时对其进行设置。有关更多信息，请参阅 Amazon Simple Storage Service 用户指南中的创建存储桶。

- CloudWatchAgentServerPolicy – 可选。此 AWS 托管式策略允许您在托管式节点上运行 CloudWatch 代理。通过使用此策略，可以读取节点上的信息并将其写入 Amazon CloudWatch。只有在使用 Amazon EventBridge 或 CloudWatch Logs 等服务时，实例配置文件才需要使用该策略。

> **Note**
>
> 使用 CloudWatch 和 Eventbridge 功能是可选的，但如果您决定使用这些功能，建议您在开始执行 Systems Manager 配置过程时设置这些功能。有关更多信息，请参阅 Amazon EventBridge 用户指南和 Amazon CloudWatch Logs 用户指南。

3. 编辑附加到节点的 IAM 角色并添加以下策略。此策略授予节点调用 `kms:Decrypt` 和 `ssm:CreateDocument` API 的权限。

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "ssm:CreateDocument"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/kms-key-id"
            ]
        }
    ]
}
```

4. 复制以下 JSON 文本并粘贴到文本编辑器中，并将文件保存为以下位置中的 `JoinInstanceToDomain.json`：`c:\temp\JoinInstanceToDomain.json`。

```
{
    "schemaVersion": "2.2",
    "description": "Run a PowerShell script to securely join a Windows Server
 instance to a domain",
    "mainSteps": [
```

```
        {
            "action": "aws:runPowerShellScript",
            "name": "runPowerShellWithSecureString",
            "precondition": {
                "StringEquals": [
                    "platformType",
                    "Windows"
                ]
            },
            "inputs": {
                "runCommand": [
                    "$domain = (Get-SSMParameterValue -Name
 domainName).Parameters[0].Value",
                    "if ((gwmi Win32_ComputerSystem).domain -eq $domain){write-host
 \"Computer is part of $domain, exiting\"; exit 0}",
                    "$username = (Get-SSMParameterValue -Name
 domainJoinUserName).Parameters[0].Value",
                    "$password = (Get-SSMParameterValue -Name domainJoinPassword -
WithDecryption $True).Parameters[0].Value | ConvertTo-SecureString -asPlainText -
Force",
                    "$credential = New-Object
 System.Management.Automation.PSCredential($username,$password)",
                    "Add-Computer -DomainName $domain -Credential $credential -
ErrorAction SilentlyContinue -ErrorVariable domainjoinerror",
                    "if($?){Write-Host \"Instance joined to domain successfully.
 Restarting\"; exit 3010}else{Write-Host \"Instance failed to join domain with
 error:\" $domainjoinerror; exit 1 }"
                ]
            }
        }
    ]
}
```

5. 在 Tools for Windows PowerShell 中运行以下命令来创建新 SSM 文档。

```
$json = Get-Content C:\temp\JoinInstanceToDomain | Out-String
New-SSMDocument -Name JoinInstanceToDomain -Content $json -DocumentType Command
```

6. 在 Tools for Windows PowerShell 中运行以下节点，将节点加入到域中。

```
Send-SSMCommand -InstanceId instance-id -DocumentName JoinInstanceToDomain
```

如果此命令成功，系统将返回类似以下内容的信息。

```
WARNING: The changes will take effect after you restart the computer EC2ABCD-
EXAMPLE.
Domain join succeeded, restarting
Computer is part of example.local, exiting
```

如果此命令失败，系统将返回类似以下内容的信息。

```
Failed to join domain with error:
Computer 'EC2ABCD-EXAMPLE' failed to join domain 'example.local'
from its current workgroup 'WORKGROUP' with following error message:
The specified domain either does not exist or could not be contacted.
```

审计和记录 Parameter Store 活动

AWS CloudTrail 捕获在 AWS Systems Manager 控制台、AWS Command Line Interface (AWS CLI) 和 Systems Manager SDK 中进行的 API 调用。您可以在 CloudTrail 控制台或 Amazon Simple Storage Service (Amazon S3) 存储桶中查看信息。您的账户的所有 CloudTrail 日志都使用一个存储桶。有关查看和使用 Systems Manager 活动的 CloudTrail 日志的更多信息，请参阅 使用 AWS CloudTrail 记录 AWS Systems Manager API 调用。有关 Systems Manager 的审计和日志记录选项的更多信息，请参阅 AWS Systems Manager 中的日志记录和监控。

排查 Parameter Store 问题

利用以下信息，帮助排查Parameter Store（AWS Systems Manager 中的一项工具）出现的问题。

aws:ec2:image 参数创建问题排查

使用以下信息可帮助排查创建 `aws:ec2:image` 数据类型参数的问题。

没有创建实例的权限

问题：您尝试使用 `aws:ec2:image` 参数创建实例，但收到“You are not authorized to perform this operation.”这样的错误信息。

- 解决方案：您不具备使用参数值创建 EC2 实例所需的所有权限，例如 `ec2:RunInstances`、`ec2:DescribeImages`、`ssm:GetParameter` 等所需的权限。请联系贵组织中拥有管理员权限的用户，向其请求必要的权限。

EventBridge 报告失败消息“Unable to Describe Resource (无法描述资源)”

问题：您运行了一条命令来创建 `aws:ec2:image` 参数，但创建参数失败。您会从 Amazon EventBridge 收到一条通知，报告异常“Unable to Describe Resource (无法描述资源)”。

解决方案：此消息可指出以下问题：

- 您没有 `ec2:DescribeImages` API 操作所需的所有权限，或者您缺少访问参数中引用的特定映像的权限。请联系贵组织中拥有管理员权限的用户，以请求必要的权限。
- 作为参数值输入的 Amazon Machine Image (AMI) ID 无效。确保您所输入的 AMI ID 在您使用的当前 AWS 区域和账户中可用。

新 **aws:ec2:image** 参数不可用

问题：您刚刚运行了一条命令来创建 `aws:ec2:image` 参数，并且系统报告了版本号，但该参数不可用。

- 解决方案：当您运行命令来创建使用 `aws:ec2:image` 数据类型的参数时，将会立即为该参数生成版本号，但必须在该参数可用之前验证参数格式。此过程可能需要数分钟。要监控参数创建和验证过程，可以执行以下操作：
  - 使用 EventBridge 向您发送有关 `create` 和 `update` 参数操作的通知。这些通知会报告参数操作是否成功。有关在 Eventbridge 中订阅 Parameter Store 事件的信息，请参阅 基于 Parameter Store 事件设置通知或触发操作。
  - 在 Systems Manager 控制台的 Parameter Store 部分中，定期刷新参数列表以搜索新参数或更新的参数详细信息。
  - 使用 GetParameter 命令检查新参数或更新的参数。例如，可以使用 AWS Command Line Interface (AWS CLI) 执行以下操作：

    ```
    aws ssm get-parameter name MyParameter
    ```

    对于新参数，将返回一条 `ParameterNotFound` 消息，直到参数经过验证。对于要更新的现有参数，在参数经过验证之前不会包含有关新版本的信息。

  如果您在验证过程完成之前尝试再次创建或更新参数，系统将报告验证仍在进行中。如果未创建或更新参数，您可以在初始尝试结束 5 分钟后重试。

AWS Systems Manager 操作工具

操作工具是一套功能，可帮助您管理您的 AWS 资源。

- Systems Manager 托管的 Amazon CloudWatch 控制面板是 CloudWatch 控制台中的可自定义主页，可用于在整个 AWS 区域 单一视图中监控资源。
- Explorer 是一个可自定义的操作控制面板，用于报告有关 AWS 资源的信息。Explorer 可以显示您的 AWS 账户 和不同 AWS 区域 中的操作数据 (OpsData) 的聚合视图。
- Incident Manager 可帮助您减轻影响 AWS 上托管的应用程序的事件并从中恢复。
- OpsCenter 提供了一个中心位置，运营工程师和 IT 专业人员可以在此查看、调查和解决与 AWS 资源相关的运营工作项（OpsItems）。

主题

- AWS Systems Manager Incident Manager
- AWS Systems Manager Explorer
- AWS Systems Manager OpsCenter
- 使用 Systems Manager 托管的 Amazon CloudWatch 控制面板

AWS Systems Manager Incident Manager

使用 Incident Manager（AWS Systems Manager 中的一项工具）来管理 AWS 托管应用程序中发生的事件。Incident Manager 结合了用户参与、上报、运行手册、响应计划、聊天渠道和事件后分析，可帮助您的团队更快地分类事件并使应用程序恢复正常状态。要了解有关 Incident Manager 的更多信息，请参阅 Incident Manager 用户指南。

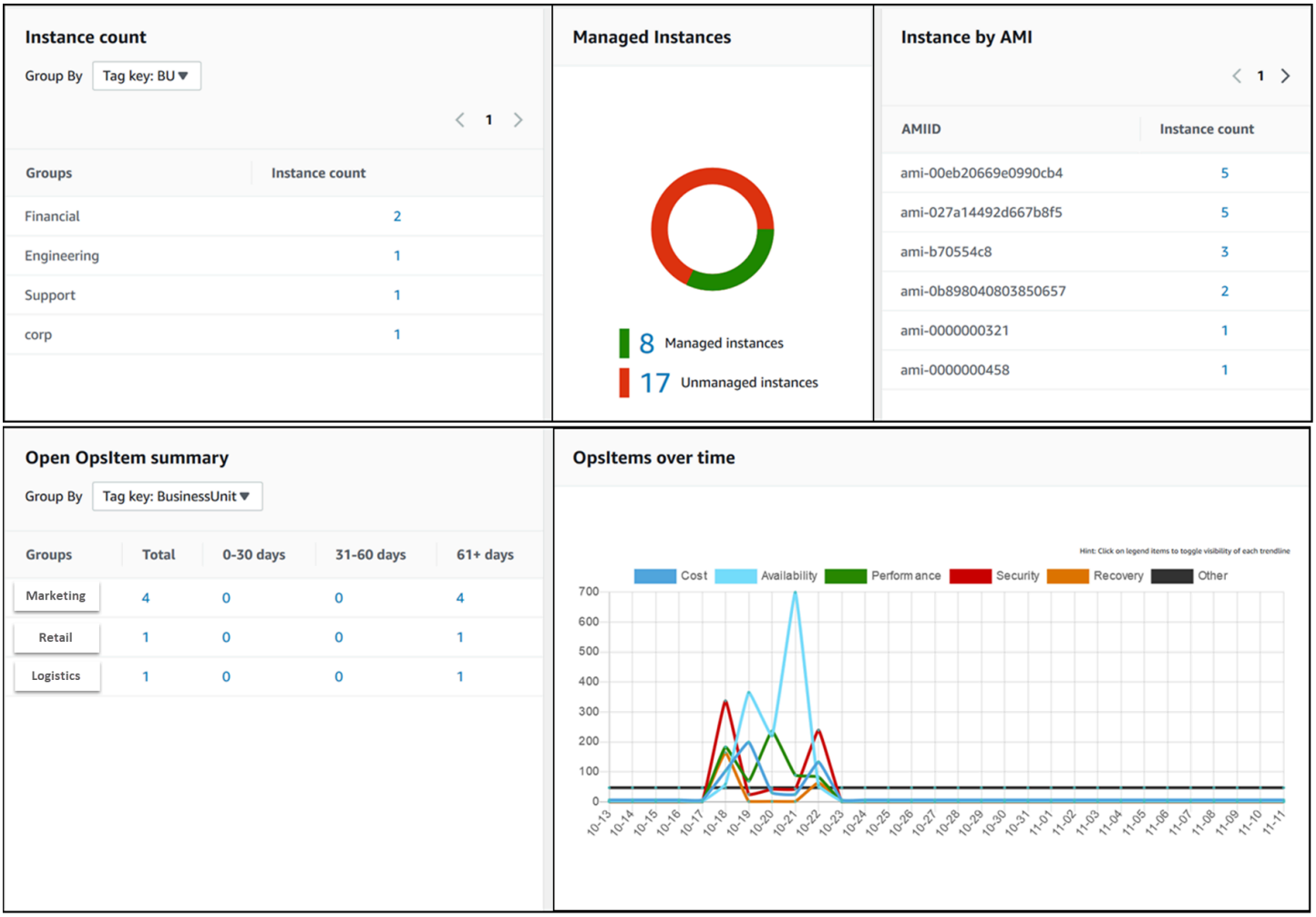
AWS Systems Manager Explorer

AWS Systems Manager Explorer 是一个可自定义的运营控制面板，用于报告有关 AWS 资源的信息。Explorer 会跨 AWS 区域 聚合显示您的 AWS 账户 的运营数据（OpsData）。在 Explorer 中，OpsData 包含有关混合和多云环境中托管式节点的元数据。OpsData 还包含其他 Systems Manager 工具提供的信息，包括Patch Manager补丁合规性和State Manager关联合规性详细信息。为了进一步简化您访问 OpsData 的过程，Explorer 会显示来自 AWS Config、AWS Trusted Advisor、AWS Compute Optimizer和（AWS 支持支持用例）等支持 AWS 服务的信息。

为提高操作感知能力，Explorer 还会显示操作工作项 (OpsItems)。Explorer 提供了有关 OpsItems 在业务单位或应用程序之间的分布方式、它们随时间变化的趋势，以及它们随类别变化的方式等上下

文。您可以在 Explorer 中对信息进行分组和筛选，以将重点放在与您相关的项目和需要采取措施的项目上。在查找高优先级问题时，您可以使用 Systems Manager OpsCenter 运行自动化运行手册并快速解决这些问题。要开始使用 Explorer，请打开 Systems Manager 控制台。在导航窗格中，选择 Explorer。

下图显示了一些单独的报告框（称为小部件），它们位于 Explorer 中。



Explorer 具有哪些功能？

Explorer 包括以下功能：

- 可操作信息的可自定义显示内容：Explorer 包括拖放式小组件，它们自动显示有关 AWS 资源的可操作信息。Explorer 在两种类型的小组件中显示信息。
  - 信息性小部件：这些小部件汇总了来自 Amazon EC2、Patch Manager、State Manager 和支持性 AWS 服务（如 AWS Trusted Advisor、AWS Compute Optimizer 和 支持）的数据。这些小组件提供了重要的上下文，以帮助您了解 AWS 资源的状态和操作风险。信息控件的示例包括 Instance

count（实例计数）、Instance by AMI（按 AMI 划分的实例）、Total noncompliant nodes（不合规的节点总数）（修补中）、Non-compliant associations（不合规的关联）和 Support Center cases（支持中心用例）。

  - OpsItem 小组件：Systems Manager OpsItem 是与一个或多个 AWS 资源相关的操作工作项。OpsItems 是 Systems Manager OpsCenter 的一项功能。OpsItems 可能要求 DevOps 工程师调查问题，并采取可能的纠正措施。可能的 OpsItems 示例包括：较高的 Amazon EC2 实例 CPU 使用率、分离的 Amazon Elastic Block Store (Amazon EBS) 卷、AWS CodeDeploy 部署失败，或 Systems Manager 自动化执行失败。OpsItem 小组件的示例包括：Open OpsItem summary (打开的 OpsItem 摘要)、OpsItem by status (按状态划分的 OpsItem)，以及 OpsItems over time (随时间变化的 OpsItem)。

- Filters（筛选条件）：每个小组件都提供了根据 AWS 账户、AWS 区域和标签来筛选信息的功能。筛选条件可以帮助您快速优化 Explorer 中显示的信息。
- Direct links to service screens (指向服务屏幕的直接链接)：为了帮助您调查 AWS 资源的问题，Explorer 小组件包含指向相关服务屏幕的直接链接。如果导航到相关的服务屏幕，应用于小部件的筛选条件仍然有效。
- Groups (组)：为了帮助您了解组织中的操作问题的类型，您可以使用一些小组件根据账户、区域和标签对数据进行分组。
- 报告标签键：在设置 Explorer 时，您可以指定最多 5 个标签键。这些键可以帮助您对 Explorer 中的数据进行分组和筛选。如果指定的键与生成 OpsItem 的资源上的键匹配，则该键和值将包含在 OpsItems 中。
- AWS 账户和 AWS 区域显示的三种模式：Explorer 包括以下显示模式，用于 AWS 账户和 AWS 区域中的 OpsData 及 OpsItems：
  - 单账户/单区域：这是默认视图。在该模式下，用户可以查看自己的账户和当前区域中的数据和 OpsItems。
  - 单账户/多区域：该模式要求您使用 Explorer Settings (设置) 页面创建一个或多个资源数据同步。资源数据同步聚合一个或多个区域中的 OpsData。在创建资源数据同步后，您可以在 Explorer 控制面板上切换要使用的同步。然后，您可以根据区域对数据进行筛选和分组。
  - Multiple-account/multiple-Region (多账户/多区域)：此模式要求您的组织或公司使用 AWS Organizations，并且打开 All features (所有功能)。在您的计算环境中配置 AWS Organizations 后，您可以在管理账户中聚合所有账户数据。然后，您可以创建资源数据同步，以便根据区域对数据进行筛选和分组。有关 Organizations 的 All features (所有功能) 模式的更多信息，请参阅在 Organizations 中启用所有功能。

- Reporting (报告)：您可以将 Explorer 报告以逗号分隔值 (.csv) 文件的形式导出到 Amazon Simple Storage Service (Amazon S3) 存储桶。导出完成后，您会收到来自 Amazon Simple Notification Service (Amazon SNS) 的提示。

Explorer 如何与 OpsCenter 相关？

Systems Manager OpsCenter 提供了一个中心位置，运营工程师和 IT 专业人员可以在其中查看、调查和解决与 AWS 资源相关的 OpsItems 问题。Explorer 是一个报告中心，DevOps 经理可以在其中查看不同 AWS 区域和账户中的操作数据的聚合摘要（包括 OpsItems）。Explorer 可以帮助用户发现趋势和模式，并在必要时使用 Systems Manager 自动化运行手册快速解决问题。

OpsCenter 设置现在与 Explorer 设置集成在一起。如果已设置 OpsCenter，则 Explorer 自动显示操作数据，包括有关 OpsItems 的聚合信息。如果尚未设置 OpsCenter，则可以使用 Explorer 设置来开始使用这两项工具。有关更多信息，请参阅 Systems Manager Explorer 和 OpsCenter 入门。

什么是 OpsData？

OpsData 是 Systems Manager Explorer 控制面板中显示的任何运营数据。Explorer 从以下源中检索 OpsData：

- Amazon Elastic Compute Cloud (Amazon EC2)

  Explorer 中显示的数据包括：节点总数、托管式和非托管式节点总数，以及使用特定 Amazon Machine Image (AMI) 的节点计数。
- Systems Manager OpsCenter

  Explorer 中显示的数据包括：OpsItems 按状态计数、OpsItems 按严重性计数、OpsItems 跨组和跨 30 天时段的未结计数以及 OpsItems 一段时间历史数据。
- Systems Manager Patch Manager

  Explorer 中显示的数据包括不合规节点和关键不合规节点的数量。
- AWS Trusted Advisor

  Explorer 中显示的数据包括：对 EC2 预留实例的成本优化、安全性、容错能力、性能和服务限制领域的最佳实践检查状态。
- AWS Compute Optimizer

  Explorer 中显示的数据包括：Under provisioned (预置不足) 和 Over provisioned (预置过度) EC2 实例的计数、优化发现结果、按需定价详情，以及关于实例类型和价格的建议。

- 支持 中心用例

  Explorer 中显示的数据包括：用例 ID、严重性、状态、创建时间、主题、服务和类别。

- AWS Config

  Explorer 中显示的数据包括：合规和不合规的 AWS Config 规则的总体摘要、合规和不合规资源的数量，以及有关每个资源的特定详细信息（当您深入了解某个不合规的规则或资源时）。

- AWS Security Hub

  Explorer 中显示的数据包括：Security Hub 结果的总体摘要、按严重性分组的每个结果的数量，以及有关结果的具体详细信息。

> **Note**
>
> 要在 Explorer 中查看 AWS Trusted Advisor 和 支持 中心工单，您必须借助 AWS 支持 支持设置企业账户或商业账户。

您可以从 Explorer Settings (设置) 页面中查看和管理 OpsData 源。有关设置和配置使用 OpsData 填充 Explorer 小部件的服务的信息，请参阅为 Explorer 设置相关服务。

使用 Explorer 是否需要收取费用？

是。在集成设置期间，当您打开创建 OpsItems 的默认规则时，将启动自动创建 OpsItems 的过程。将根据每月创建的 OpsItems 数量向您的账户收费。还会根据每月进行的 `GetOpsItem`、`DescribeOpsItem`、`UpdateOpsItem` 和 `GetOpsSummary` API 调用次数向您的账户收费。此外，可能还会针对其他服务的公有 API 调用（公开相关的诊断信息）向您收费。有关更多信息，请参阅 AWS Systems Manager定价。

主题

- Systems Manager Explorer 和 OpsCenter 入门
- 使用 Explorer
- 从 Systems Manager Explorer 中导出 OpsData
- Systems Manager Explorer 问题排查

Systems Manager Explorer 和 OpsCenter 入门

AWS Systems Manager 使用集成设置体验来帮助您开始使用 Systems Manager Explorer 和 Systems Manager OpsCenter。在本文档中，Explorer 和 OpsCenter 设置称为集成设置。如果已设置 OpsCenter，您仍然需要完成集成设置以验证设置和选项。如果尚未设置 OpsCenter，则可以使用集成设置来开始使用这两项工具。

> **Note**
>
> 集成设置仅在 Systems Manager 控制台中提供。您不能以编程方式设置 Explorer 或 OpsCenter。

集成设置执行以下任务：

- 配置角色和权限：集成设置将创建一个 AWS Identity and Access Management (IAM) 角色，允许 Amazon EventBridge 根据默认规则自动创建 OpsItems。在设置后，您必须为 OpsCenter 配置用户、组或角色权限，如本部分中所述。
- 启用用于创建 OpsItem 的默认规则：集成设置将在 EventBridge 中创建默认规则。这些规则自动创建 OpsItems 以响应事件。这些事件的示例包括：AWS 资源的状态变化、安全设置更改，或服务变得不可用。
- 激活 OpsData 源：集成设置可激活以下填充 Explorer 小组件的数据来源。
  - 支持 中心（您必须拥有业务或企业支持计划才能激活此源。）
  - AWS Compute Optimizer（您必须拥有业务或企业支持计划才能激活此源。）
  - Systems Manager State Manager 关联合规性
  - AWS Config Compliance
  - Systems Manager OpsCenter
  - Systems Manager Patch Manager 补丁合规性
  - Amazon Elastic Compute Cloud（Amazon EC2）
  - Systems Manager Inventory
  - AWS Trusted Advisor（您必须拥有业务或企业支持计划才能激活此源。）
  - AWS Security Hub

> **Note**
>
> 您可以随时在 Settings (设置) 页面上更改设置配置。

在完成集成设置后，我们建议您设置 Explorer 以显示多个区域和账户中的数据。Explorer 和 OpsCenter 将自动为您在完成集成设置时使用的 AWS 账户 和 AWS 区域 同步 OpsData 和 OpsItems。您可以创建资源数据同步以聚合其他账户和区域中的 OpsData 和 OpsItems。

为 Explorer 设置相关服务

AWS Systems Manager Explorer 和 AWS Systems Manager OpsCenter 会从其他 AWS 服务和 Systems Manager 工具收集信息，或与之交互。建议在使用集成设置之前设置并配置这些其他服务或工具。

下表包含的任务允许 Explorer 和 OpsCenter 从其他 AWS 服务和 Systems Manager 工具收集信息，或与之交互。

Task	信息
验证 Systems Manager 自动化的权限	Explorer 和 OpsCenter 允许您使用 Systems Manager 自动化运行手册修复 AWS 资源的问题。要使用此修复工具，您必须拥有运行 Systems Manager Automation 运行手册的权限。有关更多信息，请参阅 设置自动化。
设置并配置 Systems Manager Patch Manager	Explorer 包含一个小部件以提供有关补丁合规性的信息。要在 Explorer 中查看此数据，必须配置修补功能。有关更多信息，请参阅 AWS Systems Manager Patch Manager。
设置并配置 Systems Manager State Manager	Explorer 包含一个小组件，它可提供有关 Systems Manager State Manager 关联合规性的信息。要在 Explorer 中查看此数据，必须配置 State Manager。有关更多信息，请参阅 AWS Systems Manager State Manager。
打开 AWS Config 配置记录器	Explorer 使用 AWS Config 配置记录器提供的数据以在小部件中填充有关 EC2 实例的信息。要

Task	信息
	在 Explorer 中查看此数据，请打开 AWS Config 配置记录器。有关更多信息，请参阅管理配置记录器。 **Note** 在允许配置记录器后，Systems Manager 最多可能需要 6 小时才会在 Explorer 小组件中显示数据，这些小组件可以显示有关 Amazon EC2 实例的信息。
启用 AWS Trusted Advisor	Explorer 将使用 Trusted Advisor 提供的数据，显示对 Amazon EC2 预留实例在成本优化、安全性、容错能力、性能和服务限制领域进行最佳实践检查的状态。要在 Explorer 中查看此数据，您必须拥有商业或企业支持计划。有关更多信息，请参阅 支持。
启用 AWS Compute Optimizer	Explorer 使用 Compute Optimizer 提供的数据显示以下详细信息：Under provisioned (预置不足) 和 Over provisioned (预置过度) 的 Amazon EC2 实例的计数、优化结果、按需定价详细信息，以及关于实例类型和价格的建议。要在 Explorer 中查看此数据，请打开 Compute Optimizer。有关更多信息，请参阅开始使用 AWS Compute Optimizer。
启用 AWS Security Hub	Explorer 使用 Security Hub 提供的数据在小组件中填充有关安全结果的信息。要在 Explorer 中查看此数据，请打开 Security Hub 集成。有关更多信息，请参阅什么是 AWS Security Hub。

为 Systems Manager Explorer 配置角色和权限

集成设置可自动为 AWS Systems Manager Explorer 和 AWS Systems Manager OpsCenter 创建及配置 AWS Identity and Access Management（IAM）角色。如果您完成了集成设置，则不需要执行任何其他任务以便为 Explorer 配置角色和权限。不过，您必须为 OpsCenter 配置权限，如本主题后面所述。

集成设置创建和配置以下角色以使用 Explorer 和 OpsCenter。

- `AWSServiceRoleForAmazonSSM`：提供对由 Systems Manager 管理或使用的 AWS 资源的访问权限。
- `OpsItem-CWE-Role`：允许 CloudWatch Events 和 EventBridge 创建 OpsItems 以响应常见事件。
- `AWSServiceRoleForAmazonSSM_AccountDiscovery`：允许 Systems Manager 调用其他 AWS 服务，以便在同步数据时发现 AWS 账户 信息。有关该角色的更多信息，请参阅使用角色为 OpsCenter 和 Explorer 收集 AWS 账户 信息。
- `AmazonSSMExplorerExport`：允许 Explorer 将 OpsData 导出到逗号分隔值 (CSV) 文件。

如果将 Explorer 配置为使用 AWS Organizations 和资源数据同步显示多个账户和区域中的数据，则 Systems Manager 将创建 `AWSServiceRoleForAmazonSSM_AccountDiscovery` 服务相关角色。Systems Manager 使用此角色在 AWS Organizations 中获取有关 AWS 账户 的信息。该角色使用以下权限策略。

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "organizations:DescribeAccount",
            "organizations:DescribeOrganization",
            "organizations:ListAccounts",
            "organizations:ListAWSServiceAccessForOrganization",
            "organizations:ListChildren",
            "organizations:ListParents"
         ],
         "Resource":"*"
      }
```

```
    ]
}
```

有关 AWSServiceRoleForAmazonSSM_AccountDiscovery 角色的更多信息，请参阅 使用角色为 OpsCenter 和 Explorer 收集 AWS 账户 信息。

为 Systems Manager OpsCenter 配置权限

在完成集成设置后，您必须配置用户、组或角色权限，以便用户能在 OpsCenter 中执行操作。

开始前的准备工作

您可以将 OpsCenter 配置为仅为单个账户或跨多个账户创建和管理 OpsItems。如果您将 OpsCenter 配置为跨多个账户创建和管理 OpsItems，则您可以使用 Systems Manager 委派的管理员账户或 AWS Organizations 管理账户在其他账户中手动创建、查看或编辑 OpsItems。有关 Systems Manager 委派管理员账户的更多信息，请参阅 为 Explorer 配置委派管理员。

如果您为单个账户配置 OpsCenter，则您只能在创建 OpsItems 的账户中查看或编辑 OpsItems。您无法跨 AWS 账户共享或传输 OpsItems。因此，我们建议您在用于运行 AWS 工作负载的 AWS 账户中为 OpsCenter 配置权限。然后，您可以在该账户中创建用户或组。这样一来，多个运营工程师或 IT 专业人员便能在同一个 AWS 账户中创建、查看和编辑 OpsItems。

Explorer 和 OpsCenter 使用以下 API 操作。如果您的用户、组或角色拥有访问这些操作的权限，您可以使用 Explorer 和 OpsCenter 的所有功能。您还可以创建更严格的访问权限，如本节后面所述。

- CreateOpsItem
- CreateResourceDataSync
- DescribeOpsItems
- DeleteResourceDataSync
- GetOpsItem
- GetOpsSummary
- ListResourceDataSync
- UpdateOpsItem
- UpdateResourceDataSync

如果您愿意，可以通过将以下内联策略添加到您的账户、组或角色来指定只读权限。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:GetOpsItem",
        "ssm:GetOpsSummary",
        "ssm:DescribeOpsItems",
        "ssm:GetServiceSetting",
        "ssm:ListResourceDataSync"
      ],
      "Resource": "*"
    }
  ]
}
```

有关创建和编辑 IAM policy的更多信息，请参阅《IAM 用户指南》中的创建 IAM policy。有关如何将此策略分配给 IAM 组的信息，请参阅将策略附加到 IAM 组。

使用以下内容创建权限，并将该权限添加到您的用户、组或角色：

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:GetOpsItem",
        "ssm:UpdateOpsItem",
        "ssm:DescribeOpsItems",
        "ssm:CreateOpsItem",
        "ssm:CreateResourceDataSync",
        "ssm:DeleteResourceDataSync",
        "ssm:ListResourceDataSync",
        "ssm:UpdateResourceDataSync"
```

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

根据您在组织中使用的身份应用程序，您可选择以下任何选项来配置用户访问权限。

要提供访问权限，请为您的用户、组或角色添加权限：

- AWS IAM Identity Center 中的用户和群组：

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

- 通过身份提供商在 IAM 中托管的用户：

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

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

通过使用标签限制对 OpsItems 的访问

您还可以通过使用指定标签的内联 IAM policy 来限制对 OpsItems 的访问。以下是一个指定 Department 标签键和 Finance 标签值的示例。利用此策略，用户只能调用 GetOpsItem API 操作，来查看之前已用键 Department 和值 Finance 标记的 OpsItems。用户无法查看任何其他 OpsItems。

JSON

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

```
      "Resource": "*"
      ,
      "Condition": { "StringEquals": { "ssm:resourceTag/Department":
 "Finance" } }
    }
  ]
}
```

以下是一个指定用于查看和更新 OpsItems 的 API 操作的示例。此策略还指定了两组标签键/值对：Department-Finance 和 Project-Unity。

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "ssm:GetOpsItem",
            "ssm:UpdateOpsItem"
         ],
         "Resource":"*",
         "Condition":{
            "StringEquals":{
               "ssm:resourceTag/Department":"Finance",
               "ssm:resourceTag/Project":"Unity"
            }
         }
      }
   ]
}
```

有关向 OpsItem 添加标签的更多信息，请参阅 手动创建 OpsItems。

了解由集成设置创建的默认 EventBridge 规则

在 Explorer 和 OpsCenter 的集成设置过程中，您可以选择启用许多基于 Amazon EventBridge 检测到的事件的默认规则。当检测到这些事件时，系统会自动在 AWS Systems Manager OpsCenter 中创建 OpsItems。

例如，规则 `SSMOpsItems-Autoscaling-instance-termination-failure` 将在终止 EC2 自动扩缩实例失败时导致 OpsItem 被创建。

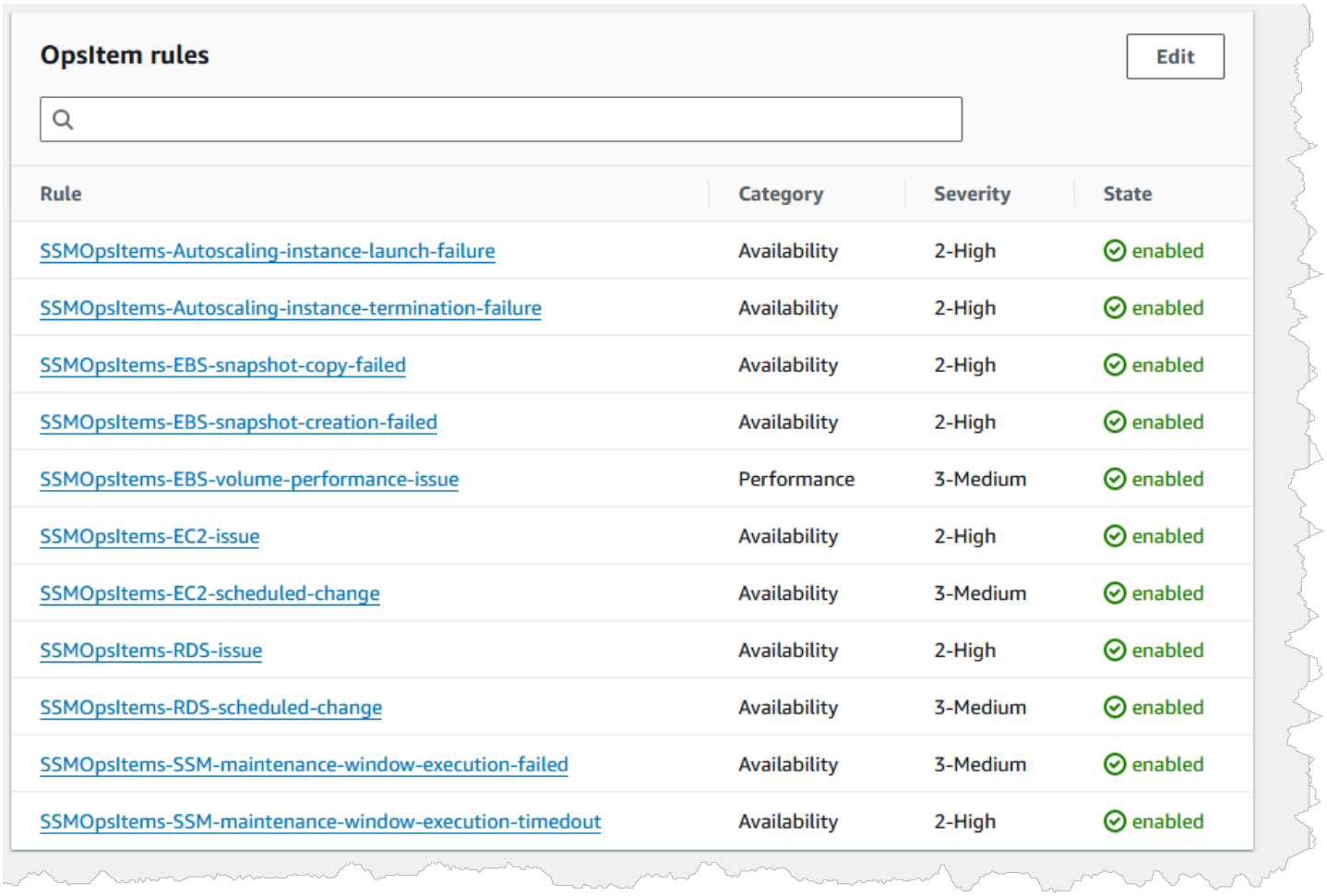
当 Systems Manager 维护时段未能成功完成时，规则 `SSMOpsItems-SSM-maintenance-window-execution-failed` 将导致 OpsItem 被创建。

有关您可以在设置过程中启用的所有 EventBridge 规则的设置说明和描述，请参阅 设置 OpsCenter。

如果您不希望 EventBridge 为这些事件创建 OpsItems，您可以在集成设置中选择不启用此选项。如果愿意，您可以将 OpsCenter 指定为特定 EventBridge 事件的目标。有关更多信息，请参阅 配置 EventBridge 规则以创建 OpsItems。

您可以在 OpsCenter 设置页面中禁用默认规则或更改其类别和严重性级别，方法是选择 OpsCenter，设置，然后在 OpsItem 规则区域中选择编辑。

您还可以在 Systems Manager 控制台中编辑分配给根据这些规则而创建的单个 OpsItem 的类别或严重性。有关信息，请参阅编辑 OpsItem。



为 Explorer 配置委派管理员

如果您通过将资源数据同步与 AWS Organizations 结合使用来聚合多个 AWS 区域和账户中的 AWS Systems Manager Explorer 数据，我们建议您为 Explorer 配置一个委派管理员。委派管理员可通过以下方式提高 Explorer 的安全性。

- 将可以创建或删除多账户和区域资源数据同步的 Explorer 管理员数量限制为一个 AWS 账户。
- 您不再需要登录到 AWS Organizations 管理账户即可管理 Explorer 中的资源数据同步。

委派管理员可以使用控制台、SDK、AWS Command Line Interface（AWS CLI）或 AWS Tools for Windows PowerShell，使用以下 Explorer 资源数据同步 API：

- CreateResourceDataSync
- DeleteResourceDataSync
- ListResourceDataSync
- UpdateResourceDataSync

委派管理员可以使用控制台或编程工具（如 SDK、AWS CLI 或 AWS Tools for Windows PowerShell）搜索、筛选和聚合 Explorer 数据。搜索、筛选和数据聚合使用 GetOpsSummary API 操作。

委派管理员可以为整个组织或组织单位的子集创建最多五个资源数据同步。委派管理员创建的资源数据同步仅在委派管理员账户中可用。您无法在 AWS Organizations 管理账户中查看同步数据或聚合数据。

> **Note**
>
> 您不能使用委派管理员账户在选择加入 AWS 区域中创建资源数据同步。您必须使用 AWS Organizations 管理账户。

有关资源数据同步的更多信息，请参阅 设置 Systems Manager Explorer 以显示来自多个账户和区域的数据。有关 AWS Organizations 的更多信息，请参阅《AWS Organizations 用户指南》中的什么是 AWS Organizations？。

主题

- 开始前的准备工作

- 配置 Explorer 委派管理员
- 取消注册 Explorer 委派管理员

开始前的准备工作

下表包含有关 Explorer 委派管理的重要信息。

- 您只能委派一个账户用于 Explorer 管理。
- 您指定为 Explorer 委派管理员的账户 ID 必须在 AWS Organizations 中列为成员账户。有关更多信息，请参阅 AWS Organizations 用户指南中的在您的组织中创建 AWS 账户。
- 委托管理员可在控制台中使用所有 Explorer 资源数据同步 API 操作，也可通过 SDK、AWS Command Line Interface (AWS CLI) 或 AWS Tools for Windows PowerShell 等编程工具来使用这些操作。资源数据同步 API 操作包括：CreateResourceDataSync、DeleteResourceDataSync、ListResourceDataSync 和 UpdateResourceDataSync。
- 委派管理员可以使用控制台或编程工具（如 SDK、AWS CLI 或 AWS Tools for Windows PowerShell）搜索、筛选和聚合 Explorer 数据。搜索、筛选和数据聚合使用 GetOpsSummary API 操作。
- 委派管理员创建的资源数据同步仅在委派管理员账户中可用。您无法在 AWS Organizations 管理账户中查看同步数据或聚合数据。
- 委派管理员最多可以创建五个资源数据同步。
- 委派管理员可以为 AWS Organizations 中的整个组织或组织单位的子集创建资源数据同步。

配置 Explorer 委派管理员

使用以下过程注册 Explorer 委派管理员。

注册 Explorer 委派管理员

1. 登录到您的 AWS Organizations 管理账户。
2. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
3. 在导航窗格中，请选择 Explorer。
4. 选择设置。

5. 在 Delegated administrator for Explorer (Explorer 的委派管理员) 部分，验证您是否已配置所需的服务链接角色和服务访问选项。如有必要，请选择 Create role (创建角色) 和 Enable access (启用访问权限) 按钮来配置这些选项。
6. 对于 Account ID (账户 ID)，输入 AWS 账户 ID。此账户必须是 AWS Organizations 中的成员账户。
7. 选择 Register delegated administrator (注册委派管理员)。

委托管理员现在拥有访问权限，可以访问 Create resource data sync (创建资源数据同步) 页面上的 Include all accounts from my AWS Organizations configuration (包括我的 Amazon Organizations 配置中的所有账户) 和 Select organization units in AWS Organizations (选择 Amazon Organizations 中的组织单位) 选项。

取消注册 Explorer 委派管理员

使用以下过程取消注册 Explorer 委派管理员。只有 AWS Organizations 管理账户才能取消注册委托管理员账户。取消注册委派管理员账户后，系统将删除该委派管理员创建的所有 AWS Organizations 资源数据同步。

取消注册 Explorer 委派管理员

1. 登录到您的 AWS Organizations 管理账户。
2. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
3. 在导航窗格中，请选择 Explorer。
4. 选择设置。
5. 在 Delegated administrator for Explorer (Explorer 的委派管理员) 部分，选择 Deregister (取消注册)。系统将显示一条警告。
6. 输入账户 ID，然后选择 Remove (删除)。

此账户不再拥有对 AWS Organizations 资源数据同步 API 操作的访问权限。系统将删除此账户创建的所有 AWS Organizations 资源数据同步。

设置 Systems Manager Explorer 以显示来自多个账户和区域的数据

AWS Systems Manager 使用集成设置体验来帮助您开始使用 AWS Systems Manager Explorer 和 AWS Systems Manager OpsCenter。完成集成设置后，Explorer 和 OpsCenter 将自动同步数据。更具体地说，在您完成集成设置时，这些工具会自动为使用的 AWS 账户和 AWS 区域同步 OpsData 及

OpsItems。如果要从其他账户和区域聚合 OpsData 和 OpsItems，您必须创建资源数据同步，如本主题中所述。

> **Note**
>
> 有关集成设置的更多信息，请参阅 Systems Manager Explorer 和 OpsCenter 入门。

主题

- 了解 Explorer 的资源数据同步
- 了解多个账户和区域资源数据同步
- 创建资源数据同步

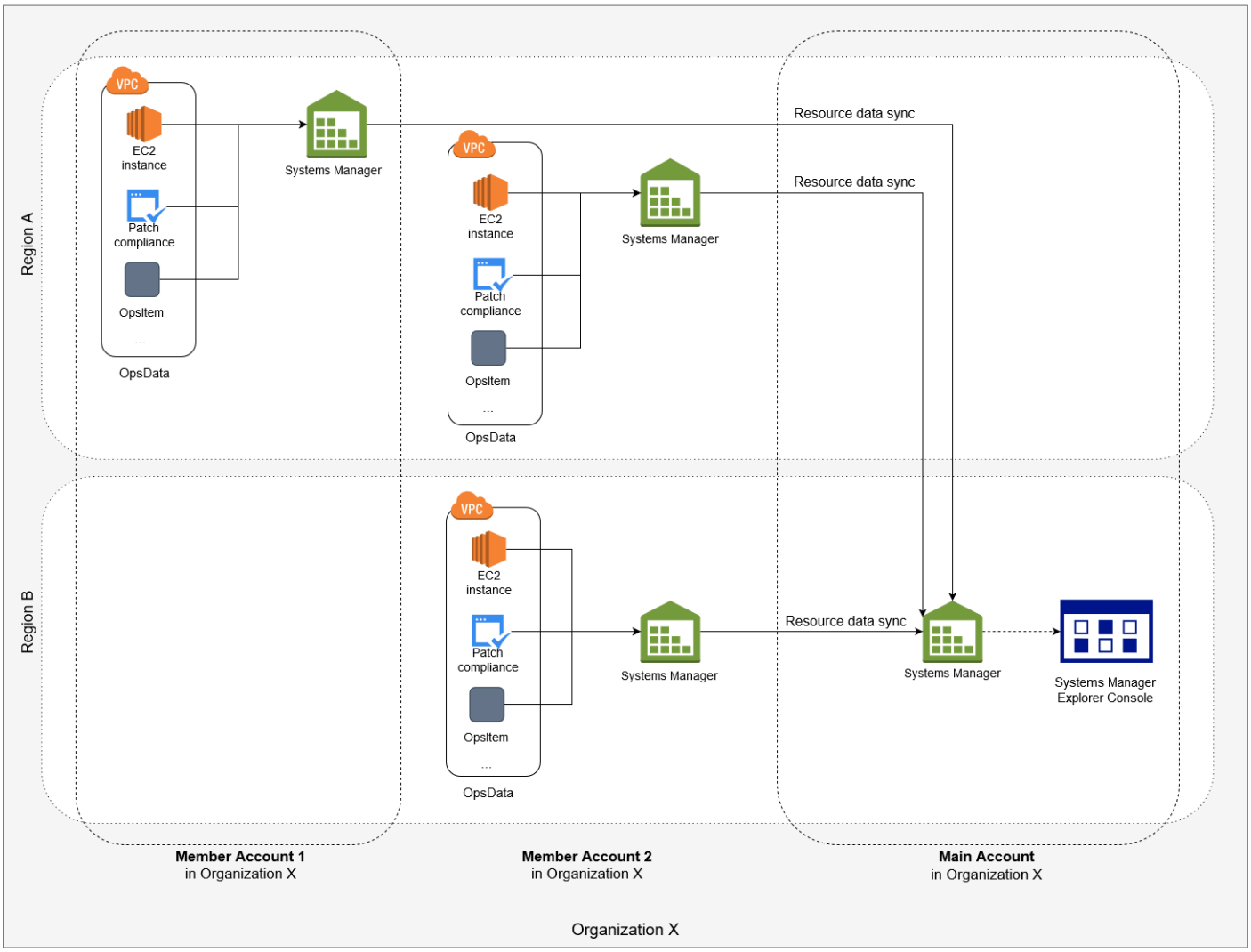
了解 Explorer 的资源数据同步

Explorer 的资源数据同步提供两个聚合选项：

- Single-account/Multiple-regions (单账户/多区域)：您可以将 Explorer 配置为聚合来自多个 AWS 区域的 OpsItems 和 OpsData 数据，但数据集仅限于当前 AWS 账户。
- Multiple-accounts/Multiple-regions (多账户/多区域)：您可以将 Explorer 配置为聚合来自多个 AWS 区域和账户的数据。此选项要求您设置和配置 AWS Organizations。在设置并配置 AWS Organizations 后，则可以按组织单位 (OU) 或为整个组织聚合 Explorer 中的数据。Systems Manager 将数据聚合到 AWS Organizations 管理账户中，然后在 Explorer 中显示这些数据。有关更多信息，请参阅AWS Organizations《用户指南》中的什么是AWS Organizations？。

> **Warning**
>
> 如果您将 Explorer 配置为在 AWS Organizations 中聚合来自企业的数据，则系统会在企业内的所有成员账户中启用 OpsData。在所有成员账户中启用 OpsData 源会增加对 OpsCenter API 的调用（比如 CreateOpsItem 和 GetOpsSummary）的数量。您需要为调用这些 API 操作付费。

下图显示了配置为使用 AWS Organizations 的资源数据同步。在此情况下，用户已在 AWS Organizations 中定义两个账户。资源数据同步将这两个账户和多个 AWS 区域中的数据聚合到 AWS Organizations 管理账户中，然后在 Explorer 中显示这些数据。



了解多个账户和区域资源数据同步

本节介绍有关使用 AWS Organizations 的多个账户和多个区域的资源数据同步的重要详细信息。具体来说，如果您在 Create resource data sync (创建资源数据同步) 页面中选择了以下选项之一，本节中的信息才适用：

- Include all accounts from my AWS Organizations configuration (包括我的 Amazon Organizations 配置中的所有账户)
- Select organization units in AWS Organizations (选择 Amazon Organizations 中的组织单位)

如果您不打算使用这些选项之一，则可以跳过本节。

在 SSM 控制台中创建资源数据同步时，如果您选择 AWS Organizations 选项之一，则 Systems Manager 会自动允许选定区域内的所有 OpsData 源用于您的组织（或选定组织单位）中的所有 AWS

账户。例如，即便您尚未在某一区域内打开 Explorer，如果您为资源数据同步选择 AWS Organizations 选项，则 Systems Manager 将自动从该区域收集 OpsData。要在不允许 OpsData 源的情况下创建资源数据同步，请在创建数据同步时将 EnableAllOpsDataSources 指定为 false。有关更多信息，请参阅 Amazon EC2 Systems Manager API 参考中的 ResourceDataSyncSource 数据类型的 `EnableAllOpsDataSources` 参数详细信息。

如果您没有为资源数据同步选择 AWS Organizations 选项之一，则您必须在希望 Explorer 访问数据的每个账户和区域中完成集成设置。否则，Explorer 不会为未完成集成设置的账户和区域显示 OpsData 和 OpsItems。

如果您将某一子账户添加到您的组织，Explorer 将自动允许所有 OpsData 源用于该账户。如果后来您从组织中删除了该子账户，Explorer 将继续从该账户中收集 OpsData。

如果您更新使用 AWS Organizations 选项之一的现有资源数据同步，系统会提示您批准收集受该更改影响的所有账户和区域的所有 OpsData 源。

如果您将新服务添加到 AWS 账户，并且如果 Explorer 收集该服务的 OpsData，Systems Manager 会自动配置 Explorer 以收集该 OpsData。例如，假设在您先前创建资源数据同步时，您的组织没有使用 AWS Trusted Advisor，但您的组织注册了此服务，Explorer 会自动更新您的资源数据同步，以收集此 OpsData。

Important

请注意以下有关多个账户和区域的资源数据同步的重要信息：

- 删除资源数据同步不会在 Explorer 中关闭 OpsData 源。
- 要查看多个账户中的 OpsData 和 OpsItems，您必须打开 AWS Organizations All features (所有功能) 模式，并且必须登录到 AWS Organizations 管理账户。
- 大多数 AWS 区域默认情况下对于您的 AWS 账户处于活动状态，但对于某些区域，仅当您手动选择它们时才会激活。这些区域称为选择加入区域。默认情况下，Explorer 跨账户/跨区域资源数据同步不支持选择加入区域中的数据聚合。2025 年 6 月 30 日，增加了对以下选择加入区域的支持。
  - 欧洲地区（米兰）
  - 非洲（开普敦）
  - 中东（巴林）
  - 亚太地区（香港）

请注意，您不能使用委派管理员账户在选择加入区域中创建资源数据同步。您必须使用 AWS Organizations 管理账户。

创建资源数据同步

在为 Explorer 配置资源数据同步前，请注意以下细节。

- Explorer 最多支持 5 个资源数据同步。
- 在为某一区域创建资源数据同步后，您将无法更改该同步的账户选项。例如，如果您在 us-east-2（俄亥俄）区域中创建一个同步，并选择 Include only the current account (仅包括当前账户) 选项，则您以后无法编辑该同步，并且无法选择 Include all accounts from my AWS Organizations configuration (包括我的 Amazon Organizations 配置中的所有账户) 选项。相反，您必须删除第一个资源数据同步，然后创建新的同步。
- 在 Explorer 中查看的 OpsData 是只读的。

可以使用以下过程为 Explorer 创建资源数据同步。

创建资源数据同步

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Explorer。
3. 选择设置。
4. 在 Configure resource data sync (配置资源数据同步) 部分中，选择 Create resource data sync (创建资源数据同步)。
5. 对于 Resource data sync name (资源数据同步名称)，请输入一个名称。
6. 在 Add accounts (添加账户) 部分中，选择一个选项。

   > **Note**
   >
   > 要使用 AWS Organizations 选项中的任何一个，您必须登录到 AWS Organizations 管理账户，或者必须登录到 Explorer 委托管理员账户。有关委派管理员账户的更多信息，请参阅 为 Explorer 配置委派管理员。

7. 在 Regions to include (要包括的区域) 部分中，选择以下选项之一。

- 选择 All current and future regions (所有当前和将来的区域) 可以自动同步所有当前 AWS 区域和将来上线的任何新区域中的数据。
- 选择 All regions (所有区域) 可以自动同步所有当前 AWS 区域中的数据。
- 单独选择要包括的区域。

8. 选择创建资源数据同步。

在创建资源数据同步后，系统可能需要几分钟的时间以在 Explorer 中填充数据。您可以查看同步，方法是从 Explorer 内的 Select a resource data sync (选择资源数据同步) 列表中选择该同步。

使用 Explorer

这一部分包含了有关如何更改小部件布局和更改将在控制面板中显示的数据，从而自定义 AWS Systems Manager Explorer 的信息。

内容

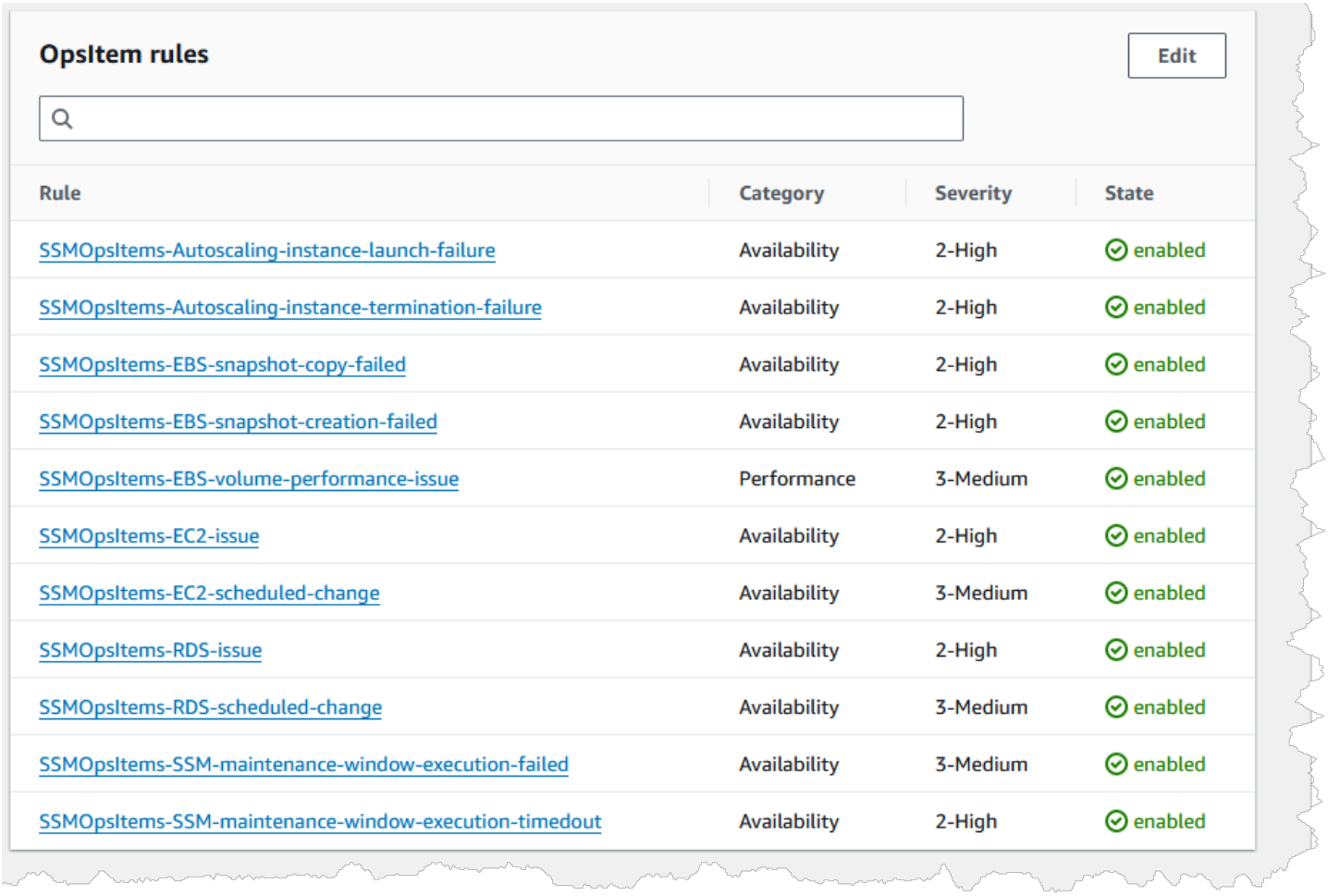
- 编辑为 Explorer 创建的 EventBridge 规则
- 编辑 Systems Manager Explorer 数据源
- 自定义 Explorer 显示视图
- 接收来自 Explorer 中的 AWS Security Hub 的调查结果
- 删除 Systems Manager Explorer 资源数据同步

编辑为 Explorer 创建的 EventBridge 规则

在完成集成设置时，系统将在 Amazon EventBridge 中允许十多个规则。这些规则在 AWS Systems Manager OpsCenter 中自动创建 OpsItems。然后，AWS Systems Manager Explorer 显示有关 OpsItems 的聚合信息。

每个规则包含预设的 Category (类别) 和 Severity (严重性) 值。在系统从事件中创建 OpsItems 时，它自动分配预设的 Category (类别) 和 Severity (严重性)。

> **Important**
>
> 您无法编辑默认规则的 Category (类别) 和 Severity (严重性) 值，但可以在根据默认规则创建的 OpsItems 上编辑这些值。



编辑用于创建 OpsItems 的默认规则

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Explorer。
3. 选择设置。
4. 在 OpsItems rules (OpsItem 规则) 部分中，选择 Edit (编辑)。
5. 展开 CWE rules (CWE 规则)。
6. 清除您不希望使用的规则旁边的复选框。
7. 使用 Category (类别) 和 Severity (严重性) 列表以更改规则的该信息。
8. 选择保存。

更改将在系统下次创建 OpsItem 时生效。

编辑 Systems Manager Explorer 数据源

对于默认可用的 AWS 区域，AWS Systems Manager Explorer 将显示来自以下来源的数据。您可以编辑 Explorer 设置以添加或删除数据源：

- Amazon Elastic Compute Cloud（Amazon EC2）
- AWS Systems Manager Inventory（清单）
- AWS Systems Manager OpsCenter OpsItems
- AWS Systems Manager Patch Manager 补丁合规性
- AWS Systems Manager State Manager 关联合规性
- AWS Trusted Advisor
- AWS Compute Optimizer
- AWS 支持 中心案例
- AWS Config 规则和资源合规性
- AWS Security Hub 检测结果

> **Note**
>
> 对于亚太地区（大阪）区域，Explorer 不会显示来自 AWS Compute Optimizer 和 AWS Security Hub 调查发现的数据。

对于 AWS 选择加入区域，Explorer 会显示来自以下来源的数据：

- Amazon Elastic Compute Cloud（Amazon EC2）
- AWS Systems Manager Inventory（清单）
- AWS Systems Manager OpsCenter OpsItems
- AWS Systems Manager Patch Manager 补丁合规性
- AWS Systems Manager State Manager 关联合规性
- AWS Trusted Advisor
- AWS 支持 中心案例
- AWS Config 规则和资源合规性

> **Note**
>
> - 要在 Explorer 中查看 支持 中心案例，必须借助 支持 设置企业账户或业务账户。
> - 您无法配置 Explorer 以停止显示 OpsCenter OpsItem 数据。

开始前的准备工作

验证并确保您已设置并配置了用数据填充 Explorer 小组件的服务。有关更多信息，请参阅 为 Explorer 设置相关服务。

要编辑数据来源

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Explorer。
3. 选择 Settings（设置），然后选择 Configure Dashboard（配置控制面板）选项卡。
4. 在 OpsData sources（OpsData 源）部分的 Status（状态）列中，根据要查看的数据打开或关闭数据源。

自定义 Explorer 显示视图

您可以在 AWS Systems Manager Explorer 中使用拖放功能自定义小部件布局。您还可以使用筛选条件自定义在 Explorer 中显示的 OpsData 和 OpsItems，如本主题中所述。

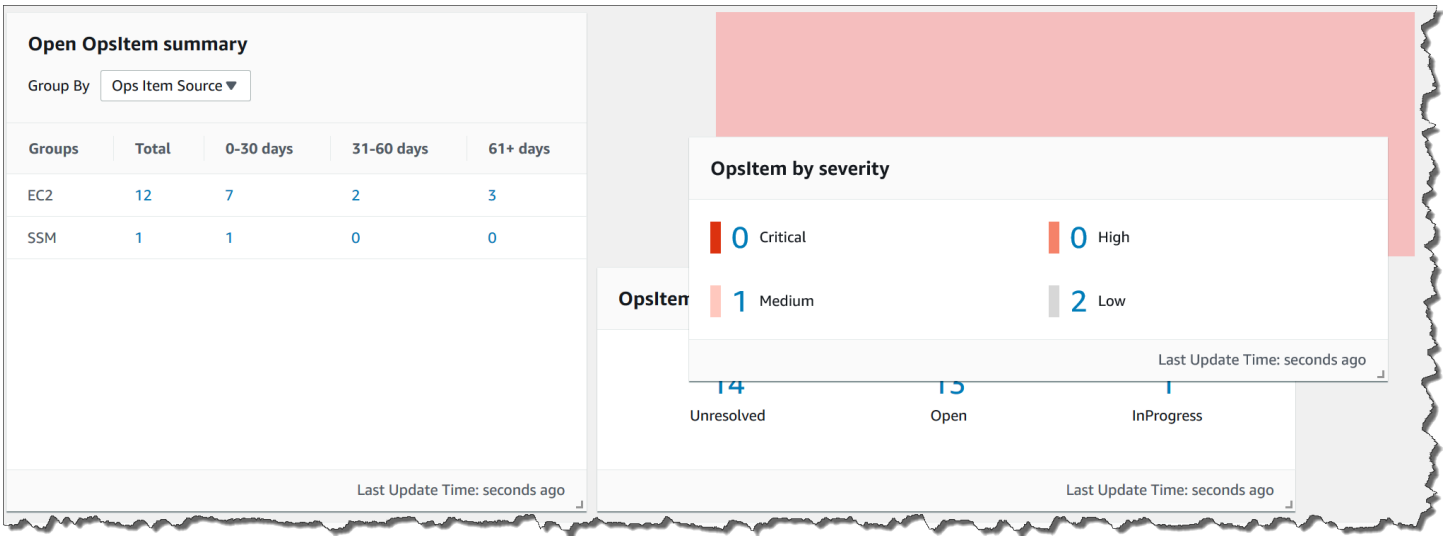
在自定义控件布局之前，请验证要查看的控件当前显示在 Explorer 中。要查看 Explorer 中的某些控件（例如 AWS Config 合规性控件），必须在 Configure dashboard（配置控制面板）页面上启用它们。

允许在 Explorer 中显示控件

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Explorer。
3. 选择 Dashboard actions（控制面板操作），Configure dashboard（配置控制面板）。
4. 选择 Configure Dashboard（配置控制面板）选项卡。
5. 选择 Enable all（全部启用）或打开单个控件或数据来源。

6. 选择 Explorer 查看您的更改。

要在 Explorer 中自定义小组件布局，请选择要移动的小组件。单击并按住小部件的名称，然后将其拖到新位置。



对于要移动的每个小部件，请重复该过程。

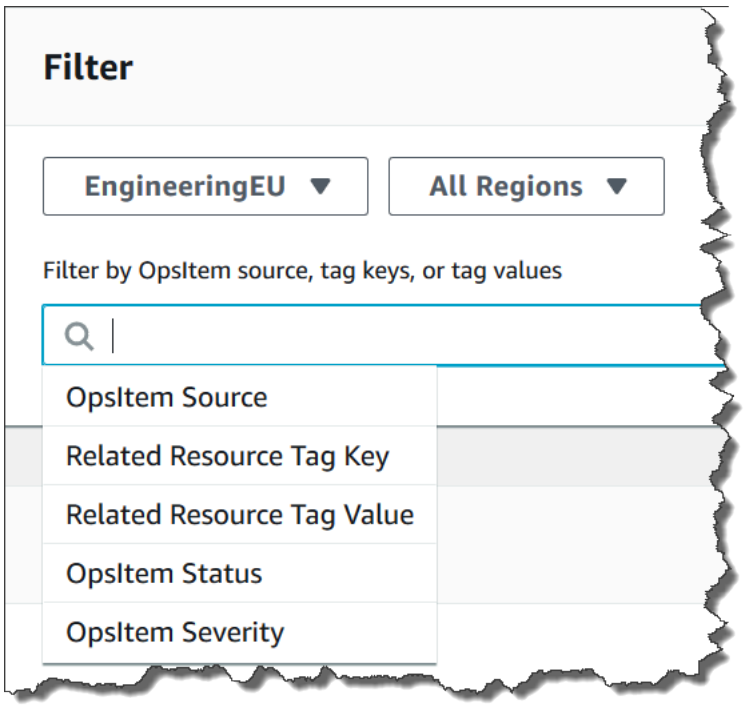
如果确定您不喜欢新布局，请选择 Reset layout (重置布局) 以将所有小部件移回到原始位置。

使用筛选条件更改 Explorer 中显示的数据

预设情况下，Explorer 显示当前 AWS 账户和当前区域的数据。如果创建一个或多个资源数据同步，您可以使用筛选条件更改哪个同步处于活动状态。然后，您可以选择显示特定区域或所有区域的数据。您也可以使用搜索栏筛选不同的 OpsItem 和键标签条件。

使用筛选条件更改在 Explorer 中显示的数据

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Explorer。
3. 在 Filter (筛选条件) 部分中，使用 Select a resource data sync (选择资源数据同步) 列表以选择一个同步。
4. 使用 Regions (区域) 列表选择特定的 AWS 区域，或者选择 All Regions (所有区域)。
5. 选择搜索栏，然后选择用于筛选数据的条件。



6. 按 Enter。

如果关闭并重新打开页面，Explorer 将保留您选择的筛选选项。

接收来自 Explorer 中的 AWS Security Hub 的调查结果

AWS Security Hub 向您提供在 AWS 中安全状态的全面视图。该服务从跨 AWS 账户、服务和支持的第三方产品收集安全数据，称为调查发现。Security Hub 的调查发现可帮助您检查您的环境是否符合安全行业标准和最佳实践、分析安全趋势并确定优先级最高的安全问题。

Security Hub 会将调查发现发送到 Amazon EventBridge，后者使用事件规则将这些调查发现发送到 Explorer。启用集成后（如此处所述），您可以在 Explorer 小组件中查看 Security Hub 的调查发现，并在 OpsCenter OpsItems 中查看调查发现的详细信息。根据严重性，小组件提供所有 Security Hub 调查发现的摘要。Security Hub 中的新调查发现通常在创建后几秒钟内即可在 Explorer 中看见。

Warning

请注意以下重要信息：

- Explorer 已与 OpsCenter（Systems Manager 中的一项工具）集成。启用与 Security Hub 的 Explorer 集成后，OpsCenter 会自动为 Security Hub 调查发现创建 OpsItems。根据您的 AWS 环境，启用集成可能会导致大量的 OpsItems，并收取一定费用。

在继续操作之前，请阅读有关与 Security Hub OpsCenter 集成的信息。该主题包含有关如何更改和更新调查发现，以及如何从您的账户中收取 OpsItems 费用的特定详细信息。有关更多信息，请参阅 了解 OpsCenter 与 AWS Security Hub 的集成。有关 OpsCenter 定价信息，请参阅 AWS Systems Manager 定价。

- 如果您在登录管理员账户时在 Explorer 中创建资源数据同步，则会自动为管理员和同步中的所有成员账户启用 Security Hub 集成。启用后，OpsCenter 会自动为 Security Hub 调查发现创建 OpsItems，并收取一定费用。有关创建资源数据同步的更多信息，请参阅 设置 Systems Manager Explorer 以显示来自多个账户和区域的数据。

Explorer 接收的结果的类型

Explorer 将接收来自 Security Hub 的所有结果。当您打开 Security Hub 默认设置时，可在 Explorer 小组件中查看基于严重性的所有结果。默认情况下，Explorer 会针对重大和高严重性的调查发现创建 OpsItems。您可以手动配置 Explorer，已针对中等和低严重性调查发现创建 OpsItems。

尽管 Explorer 不会针对信息性调查发现创建 OpsItems，但您可以在 Security Hub 调查发现摘要小组件中查看信息操作数据（OpsData）。无论严重程度如何，Explorer 都会为所有调查发现创建 OpsData。有关 Security Hub 严重性级别的更多信息，请参阅《AWS Security Hub API Reference》中的 Severity。

启用集成

本部分介绍如何启用和配置 Explorer 以开始接收 Security Hub 调查发现。

开始前的准备工作

请完成以下任务，然后才能配置 Explorer 开始接收 Security Hub 结果。

- 启用和配置 Security Hub。有关更多信息，请参阅 AWS Security Hub 用户指南中的设置 Security Hub。
- 登录到 AWS Organizations 管理账户。Systems Manager 需要对 AWS Organizations 的访问权限，才能从 Security Hub 结果创建 OpsItems。在您登录到管理账户后，系统将提示您选择 Explorer Configure dashboard (配置控制面板) 选项卡上的 Enable access (启用访问权限)，如以下过程中所述。如果您没有登录到 AWS Organizations 管理账户，则不能允许访问，并且 Explorer 无法从 Security Hub 结果创建 OpsItems。

开始接收 Security Hub 结果

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Explorer。
3. 选择设置。
4. 选择 Configure dashboard (配置控制面板) 选项卡。
5. 选择 AWS Security Hub。
6. 选择 Disabled (已禁用) 滑块，以打开 AWS Security Hub。

   默认情况下，会显示重大和高严重性的调查发现。要同时显示中等和低严重性安全性调查发现，请选择中等、低旁边的已禁用滑块。
7. 在 OpsItems created by Security Hub findings (通过 Security Hub 结果创建的 OpsItems) 部分中，选择 Enable access (启用访问权限)。如果您没有看到此按钮，请登录到 AWS Organizations 管理账户，然后返回到此页面以选择该按钮。

如何查看来自 Security Hub 的结果

以下过程介绍了如何查看 Security Hub 结果。

查看 Security Hub 结果

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Explorer。
3. 查找 AWS Security Hub 结果摘要小组件。这将显示 Security Hub 结果。您可以选择严重性级别，以查看相应 OpsItem 的详细描述。

如何停止接收结果

以下过程介绍了如何停止接收 Security Hub 结果。

停止接收 Security Hub 结果

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Explorer。

3. 选择设置。
4. 选择 Configure dashboard (配置控制面板) 选项卡。
5. 选择 Enabled (已启用) 滑块以关闭 AWS Security Hub。

> **Important**
>
> 如果控制台中禁用 Security Hub 调查发现的选项显示为灰色，则可以通过在 AWS CLI 中运行以下命令来禁用此设置。您必须在登录 AWS Organizations 管理账户或 Systems Manager 委派管理员账户时运行该命令。对于 `region` 参数，请指定您要停止在 Explorer 中接收 Security Hub 调查发现的 AWS 区域。
>
> ```
> aws ssm update-service-setting --setting-id /ssm/opsdata/SecurityHub --setting-value Disabled --region AWS ##
> ```
>
> 以下为示例。
>
> ```
> aws ssm update-service-setting --setting-id /ssm/opsdata/SecurityHub --setting-value Disabled --region us-east-1
> ```

删除 Systems Manager Explorer 资源数据同步

在 AWS Systems Manager Explorer 中，您可以创建资源数据同步以聚合其他账户和区域中的 OpsData 和 OpsItems。

您无法更改资源数据同步的账户选项。例如，如果您在 us-east-2（俄亥俄）区域中创建了一个同步，并选择了 Include only the current account (仅包括当前账户) 选项，则您以后无法编辑该同步，并且无法选择 Include all accounts from my AWS Organizations configuration (包括我的 Amazon Organizations 配置中的所有账户) 选项。相反，您必须删除该资源数据同步，然后创建新的同步，如以下过程中所述。

删除资源数据同步

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Explorer。

3. 选择设置。
4. 在 Configure resource data sync (配置资源数据同步) 部分中，选择要删除的资源数据同步。
5. 选择 Delete（删除）。

从 Systems Manager Explorer 中导出 OpsData

您可以从 AWS Systems Manager Explorer 将 5000 个 OpsData 项目以逗号分隔值 (.csv) 文件的形式，导出到 Amazon Simple Storage Service（Amazon S3）存储桶。Explorer 使用 AWS-ExportOpsDataToS3 自动化运行手册来导出 OpsData。导出 OpsData 时，系统会显示自动化运行手册页面，您可以在其中指定详细信息，例如 assumeRole、Amazon S3 存储桶名称、SNS 主题 ARN 和要导出的字段。

导出 OpsData：

- 步骤 1：指定 SNS 主题
- 步骤 2：（可选）配置数据导出
- 步骤 3：导出 OpsData

步骤 1：指定 SNS 主题

在您配置数据导出时，您必须指定一个 Amazon Simple Notification Service (Amazon SNS) 主题，该主题所在的 AWS 区域与您要导出数据的区域相同。导出完成后，Systems Manager 将向该 Amazon SNS 主题发送通知。有关创建 Amazon SNS 主题的信息，请参阅创建 Amazon SNS 主题。

步骤 2：（可选）配置数据导出

您可以从设置或将操作数据导出到 S3 存储桶页面配置数据导出设置。

从 Explorer 配置数据导出

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Explorer。
3. 选择设置。
4. 在 Configure data export (配置数据导出) 部分中，选择 Edit (编辑)。
5. 要将数据导出文件上传到现有的 Amazon S3 存储桶，请选择 选择现有的 S3 存储桶，然后从列表中选择该存储桶。

要将数据导出文件上传到新的 Amazon S3 存储桶，请选择创建新的 S3 存储桶，然后输入要用于新存储桶的名称。

> **Note**
>
> 您只能在 Explorer 中首次配置 Amazon S3 存储桶名称和 Amazon SNS 主题 ARN 的页面上编辑这些设置。如果您从设置页面设置了 Amazon S3 存储桶和 Amazon SNS 主题 ARN，则只能从设置页面修改这些设置。

6. 对于选择 Amazon SNS 主题 ARN，选择要在导出完成时通知的主题。
7. 选择创建。

步骤 3：导出 OpsData

在导出 Explorer 数据时，Systems Manager 将创建一个名为 `AmazonSSMExplorerExportRole` 的 AWS Identity and Access Management（IAM）角色。此角色使用以下 IAM policy。

JSON

```
{
      "Version": "2012-10-17",
      "Statement": [
          {
              "Sid": "OpsSummaryExportAutomationServiceRoleStatement1",
              "Effect": "Allow",
              "Action": [
                  "s3:PutObject"
              ],
              "Resource": [
                 "arn:aws:s3:::{{ExportDestinationS3BucketName}}/*"
              ]
          },
          {
              "Sid": "OpsSummaryExportAutomationServiceRoleStatement2",
              "Effect": "Allow",
              "Action": [
                  "s3:GetBucketAcl",
                  "s3:GetBucketLocation"
              ],
```

```
            "Resource": [
               "arn:aws:s3:::{{ExportDestinationS3BucketName}}"
            ]
        },
        {
            "Sid": "OpsSummaryExportAutomationServiceRoleStatement3",
            "Effect": "Allow",
            "Action": [
                "sns:Publish"
            ],
            "Resource": [
                "arn:aws:sns:*:*:{{SNSTopicName}}"
            ]
        },
        {
            "Sid": "OpsSummaryExportAutomationServiceRoleStatement4",
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups",
                "logs:DescribeLogStreams"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Sid": "OpsSummaryExportAutomationServiceRoleStatement5",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:PutLogEvents",
                "logs:CreateLogStream"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Sid": "OpsSummaryExportAutomationServiceRoleStatement6",
            "Effect": "Allow",
            "Action": [
                "ssm:GetOpsSummary"
            ],
            "Resource": [
```

```
                "*"
            ]
        }
    ]
}
```

该角色包含以下信任实体。

JSON

```
{
      "Version": "2012-10-17",
      "Statement": [
        {
          "Sid": "OpsSummaryExportAutomationServiceRoleTrustPolicy",
          "Effect": "Allow",
          "Principal": {
            "Service": "ssm.amazonaws.com"
          },
          "Action": "sts:AssumeRole"
        }
      ]
    }
```

从 Explorer 中导出 OpsData

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Explorer。
3. 在 Explorer 小组件中选择数据链接。例如，在按状态分类 OpsItems 小组件中选择未解决的问题或待解决的问题。或者，在按严重性分类 OpsItem 小组件中选择严重或高数据链接。Explorer 会打开所选数据集的 OpsData 页面。
4. 选择导出表。

> **Note**
>
> 首次导出 OpsData 时，系统会为导出创建代入角色。您无法修改默认的代入角色。

5. 对于 S3 存储桶名称，选择现有存储桶。您可以根据需要选择创建以创建 Amazon S3 存储桶。

   如果您无法更改 S3 存储桶名称，则表示您已从设置页面配置存储桶名称。您只能从设置页面更改存储桶名称。

   > **Note**
   >
   > 您只能在 Explorer 中首次配置 Amazon S3 存储桶名称和 Amazon SNS 主题 ARN 的页面上编辑这些设置。

6. 对于 SNS 主题 ARN，请选择现有的 Amazon SNS 主题 ARN，以便在下载完成时发出通知。

   如果您无法更改 Amazon SNS 主题 ARN，则表示您已从设置页面配置 Amazon SNS 主题 ARN。您只能从设置页面更改主题 ARN。

7. （可选）对于 SNS 成功消息，指定成功完成导出时要显示的成功消息。
8. 选择提交。系统导航到上一页并显示消息单击查看导出过程的状态。查看详细信息。

   您可以选择查看详细信息以在 Systems Manager Automation 中查看运行手册的状态和进度。

现在，您可以将 OpsData 从 Explorer 导出到指定的 Amazon S3 存储桶。

如果您无法使用此过程导出数据，请验证并确保您的用户、组或角色包括 `iam:CreatePolicyVersion` 和 `iam:DeletePolicyVersion` 操作。有关将这些操作添加到您的用户、组或角色的信息，请参阅《IAM 用户指南》中的编辑 IAM policy。

Systems Manager Explorer 问题排查

本主题包含有关如何排查 AWS Systems Manager Explorer 常见问题的信息。

资源数据同步停止工作

如果资源数据同步已停止用于选择加入 AWS 区域，请确认 AWS Organizations 管理账户和配置同步的成员账户都选择加入该区域。

大多数 AWS 区域默认情况下对于您的 AWS 账户处于活动状态，但对于某些区域，仅当您手动选择它们时才会激活。这些区域称为选择加入区域。默认情况下，Explorer 跨账户/跨区域资源数据同步不支持选择加入区域中的数据聚合。2025 年 6 月 30 日，增加了对以下选择加入区域的支持。

- 欧洲地区（米兰）
- 非洲（开普敦）

- 中东（巴林）
- 亚太地区（香港）

请注意，您不能使用委派管理员账户在选择加入区域中创建资源数据同步。您必须使用 AWS Organizations 管理账户。

在 Settings（设置）页面上更新标签后无法筛选 Explorer 中的 AWS 资源

如果您在 Explorer 中更新标签键或其他数据设置，系统可能需要最多 6 小时来根据您的更改同步数据。

Create resource data sync (创建资源数据同步) 页面上的 AWS Organizations 选项灰显

仅当您设置并配置了 AWS Organizations 后，Create resource data sync (创建资源数据同步) 页面上的 Include all accounts from my AWS Organizations configuration (包括我的 Amazon Organizations 配置中的所有账户) 和 Select organization units in AWS Organizations (选择 Amazon Organizations 中的组织单位) 选项才可用。如果设置并配置了 AWS Organizations，则 AWS Organizations 管理账户或 Explorer 委托管理员都可以创建使用这些选项的资源数据同步。

有关更多信息，请参阅 设置 Systems Manager Explorer 以显示来自多个账户和区域的数据 和 为 Explorer 配置委派管理员。

Explorer 根本不显示任何数据

- 确认在您希望 Explorer 访问和显示数据的每个账户和区域中完成了集成设置。否则，Explorer 不会为未完成集成设置的账户和区域显示 OpsData 和 OpsItems。有关更多信息，请参阅 Systems Manager Explorer 和 OpsCenter 入门。
- 在使用 Explorer 查看多个账户和区域中的数据时，请验证并确保您登录到 AWS Organizations 管理账户或 Explorer 委托管理员账户。要查看多个账户和区域中的 OpsData 和 OpsItems，您必须登录到该账户。

有关 Amazon EC2 实例的小组件不显示数据

如果有关 Amazon Elastic Compute Cloud (Amazon EC2) 实例，例如 Instance count (实例计数)、Managed instances (托管实例) 和 Instance by AMI (按 AMI 小组件划分的实例)的小组件不显示数据，则请验证并确保以下内容：

- 确认您等待了几分钟。在完成集成设置后，OpsData 可能需要几分钟的时间才会在 Explorer 中显示。

- 确认您配置了 AWS Config 配置记录器。Explorer 使用 AWS Config 配置记录器提供的数据在小部件中填充有关 EC2 实例的信息。有关更多信息，请参阅管理配置记录器。
- 验证并确保 Amazon EC2 OpsData 源在 Settings (设置) 页面上处于活动状态。此外，还要验证并确保自您激活配置记录器或对实例进行更改后已经过去 6 个小时。在您最初激活配置记录器或对您的实例进行更改后，Systems Manager 可能需要最多六个小时才能在 Explorer EC2 小组件中显示来自 AWS Config 的数据。
- 请注意，如果停止或终止实例，则 Explorer 在 24 小时后停止显示这些实例。
- 验证并确保您位于正确的 AWS 区域中，您在其中配置了 Amazon EC2 实例。Explorer 不会显示有关本地实例的数据。
- 如果您为多个账户和区域配置了一个资源数据同步，请验证并确保您登录到 Organizations 管理账户或 Explorer 委托管理员账户。

补丁小部件不显示数据

Non-compliant instances for patching (不合规的修补实例) 小组件仅显示有关不合规的补丁实例的数据。如果实例合规，则该小部件不显示任何数据。如果您怀疑具有不合规的实例，请验证并确保您设置并配置了 Systems Manager 修补，并使用 AWS Systems Manager Patch Manager 检查补丁合规性。有关更多信息，请参阅 AWS Systems Manager Patch Manager。

其他问题

Explorer 不允许您编辑或修复 OpsItems：跨账户或区域查看的 OpsItems 是只读的。只能从主账户或区域中更新和修复它们。

AWS Systems Manager OpsCenter

OpsCenter（AWS Systems Manager 中的一项工具）提供了一个中心位置，运营工程师和 IT 专业人员可以在此处查看、调查和解决与 AWS 资源相关的操作工作项（OpsItems）。OpsCenter 旨在缩短影响 AWS 资源的问题的平均解决时间。OpsCenter 聚合并标准化各种服务的 OpsItems，同时提供有关每个 OpsItem、相关 OpsItems 以及相关资源的上下文调查数据。OpsCenter 还提供 Systems Manager Automation 运行手册，可用于快速解决问题。您可以为每个 OpsItem 指定可搜索的自定义数据。您还可以按状态和源查看自动生成的 OpsItems 相关摘要报告。要开始使用 OpsCenter，请打开 Systems Manager 控制台。在导航窗格中，选择 OpsCenter。

OpsCenter 已与 Amazon EventBridge 和 Amazon CloudWatch 集成。所以，您可以配置这些服务，以便当 CloudWatch 告警进入 ALARM 状态时或者当 EventBridge 处理发布事件中的 AWS 服务的事件时，自动在 OpsCenter 中创建 OpsItem。通过配置 CloudWatch 告警和 EventBridge 事件以自动创建 OpsItems，使您可以在单个控制台中快速诊断和修正 AWS 资源的问题。

为了帮助您诊断问题，每个 OpsItem 均包含上下文相关信息，例如生成 OpsItem 的 AWS 资源名称和 ID、告警或事件详细信息、告警历史记录和告警时间线图。

对于 AWS 资源，OpsCenter 可汇集来自 AWS Config、AWS CloudTrail 日志和 Amazon CloudWatch Events 的信息，因此您在调查期间不必浏览多个控制台页面。

以下列表包含 AWS 资源和指标的类型，客户可为其配置创建 OpsItems 的 CloudWatch 告警。

- Amazon DynamoDB：数据库读取和写入操作达到阈值
- Amazon EC2：CPU 使用率达到阈值
- AWS 账单：预估费用达到阈值
- Amazon EC2：实例未能通过状态检查
- Amazon Elastic Block Store (EBS)：磁盘空间使用率达到阈值

以下列表包含客户配置用于创建 OpsItems 的 EventBridge 规则类型。

- AWS Security Hub：发出安全提示
- DynamoDB：节流事件
- Amazon EC2 Auto Scaling：启动实例失败
- Systems Manager：运行自动化失败
- AWS Health：计划维护提示
- EC2：实例状态从 `Running` 更改为 `Stopped`

OpsCenter 还与适用于 .NET 和 SQL Server 的 Amazon CloudWatch Application Insights 进行集成。因此，您可以为应用程序中检测到的问题自动创建 OpsItems。您也可以集成 OpsCenter 与 AWS Security Hub，以便在 Systems Manager 中汇集安全性、性能和操作问题并采取措施。

运营工程师和 IT 专业人员可以使用 AWS Systems Manager 控制台中的 OpsCenter 页面、公有 API 操作、AWS Command Line Interface(AWS CLI)、AWS Tools for Windows PowerShell或 AWS SDK，以创建、查看和编辑 OpsItems。OpsCenter 公有 API 操作还支持将 OpsCenter 与您的用例管理系统和运行状况控制面板集成。

我的组织如何从 OpsCenter 获益？

对于查看、处理和修正与 AWS 资源相关的问题,OpsCenter 提供了标准和统一的体验。标准和统一的体验可以缩短解决问题、调查相关问题以及培训新的运营工程师和 IT 专业人员所需的时间。标准和统一的体验还减少了输入管理和修复问题的系统中的手动错误的数量。

更具体地说，OpsCenter 为运营工程师和组织提供以下好处：

- 您不再需要浏览多个控制台页面来查看、调查和解决与 AWS 资源相关的 OpsItems。OpsItems 跨服务聚合在一个中心位置。
- 您可以查看与 OpsItems 相关的特定服务数据和上下文相关数据，这些数据由 CloudWatch 告警、EventBridge 事件以及适用于 .NET 和 SQL Server 的 CloudWatch Application Insights 自动生成。
- 您可以指定与 OpsItem 相关的资源的 Amazon Resource Name (ARN)。通过指定相关资源，OpsCenter 可使用内置逻辑来帮助您避免创建重复的 OpsItems。
- 您可以查看有关类似 OpsItems 的详细信息和分辨率信息。
- 您可以快速查看并运行有关 Systems Manager 自动化运行手册的信息，以解决问题。

OpsCenter 具有哪些功能？

- 自动化和手动 OpsItem 创建

  OpsCenter 已与 Amazon CloudWatch 集成。因此，您可以配置 CloudWatch，以便当告警进入 ALARM 状态时或者当 Amazon EventBridge 处理发布事件的 AWS 服务的事件时，可自动在 OpsCenter 中创建 OpsItem。您还可以手动创建 OpsItems。

  OpsCenter 还与适用于 .NET 和 SQL Server 的 Amazon CloudWatch Application Insights 进行集成。因此，您可以为应用程序中检测到的问题自动创建 OpsItems。

- 详细的且可搜索的 OpsItems

  每个 OpsItem 包含多个信息字段，包括标题、ID、优先级、描述、OpsItem 的源以及该源的上次更新日期/时间。每个 OpsItem 还包括以下可配置的功能：

  - 状态：“打开”、“正在进行中”、“已解决”或“打开且正在进行中”。
  - 相关资源：相关资源是指受影响的资源或已启动创建 OpsItem 的 EventBridge 事件的资源。每个 OpsItem 均包含一个相关资源部分，其中 OpsCenter 自动列出了相关资源的 Amazon Resource Name (ARN)。您也可以手动指定相关资源的 ARN。对于一些 ARN 类型，OpsCenter 会自动创建一个深层链接，您可通过该链接查看有关此资源的详细信息，而无需访问其他控制台页面来查看此类信息。例如，如果您指定 EC2 实例的 ARN，则可以在 OpsCenter 中查看 EC2 提供的有关该实例的所有详细信息。您可以手动添加额外的相关资源的 ARN。每个 OpsItem 可以列出最多 100 个相关资源 ARN。有关更多信息，请参阅 将相关资源添加到 OpsItem。

- 相关和相似 OpsItems：借助相关 OpsItems 功能，您可以指定在某些方面与当前 OpsItem 相关的 OpsItems 的 ID。相似 OpsItem 功能将自动审查 OpsItem 标题和描述，然后列出可能与您相关或您感兴趣的其他 OpsItems。
- 可搜索的私有操作数据：操作数据是一种自定义数据，该数据提供了有关 OpsItem 的有用参考详细信息。例如，您可以指定日志文件、错误字符串、许可密钥、故障排除提示或其他相关数据。请以键/值对的形式输入操作数据。密钥的最大长度为 128 个字符。值最大为 20 KB 大小。

  此自定义数据是可搜索的，但有限制。对于 Searchable operational data (可搜索的操作数据) 功能，所有有权访问 OpsItem概览页面（由 DescribeOpsItems API 操作提供）的用户都可以查看和搜索指定的数据。对于 Private operational data (私有操作数据) 功能，数据仅可供有权访问 OpsItem（由 GetOpsItem API 操作提供）的用户查看。
- 重复数据删除：通过指定相关资源，OpsCenter 可使用内置逻辑来帮助您避免创建重复的 OpsItems。OpsCenter 还包括一项名为 操作洞察 的功能，可显示有关 OpsItems 的重复信息。要进一步限制账户中重复的 OpsItems 的数量，您可以为 EventBridge 事件规则手动指定重复数据删除字符串。有关更多信息，请参阅 管理重复的 OpsItems。

- 批量编辑 OpsItems：您可以在 OpsCenter 中选择多个 OpsItems，然后编辑以下字段之一：Status（状态）、Priority（优先级）、Severity（严重性）、Category（类别）。
- 使用运行手册轻松修正

  每个 OpsItem 均包含一个 Runbooks (运行手册) 部分以及一个 Systems Manager 自动化运行手册列表，您可使用这些运行手册自动修正与 AWS 资源相关的常见问题。如果您打开 OpsItem，请为该 OpsItem 选择 AWS 资源，然后在控制台中选择 Run automation（运行自动化）按钮，则 OpsCenter 会提供一个自动化运行手册列表，以供您在生成 OpsItem 的 AWS 资源中运行这些运行手册。从 OpsItem 运行自动化运行手册后，运行手册将自动与该 OpsItem 的相关资源关联以供将来参考。此外，如果您通过使用 OpsCenter 在 EventBridge 中自动设置 OpsItem 规则，则 EventBridge 将自动关联常见事件的运行手册。OpsCenter 为特定的 OpsItem 保存 30 天的 自动化运行手册记录。有关更多信息，请参阅 修复 OpsItem 问题。
- 更改通知：您可以指定 Amazon Simple Notification Service (SNS) 主题的 ARN 并在更改或编辑 OpsItem 时随时发布通知。SNS 主题必须位于与 OpsItem 相同的 AWS 区域中。
- 全面的 OpsItem 搜索功能：OpsCenter 提供多个搜索选项来帮助您快速找到 OpsItems。以下是几个搜索方式的示例：OpsItem ID、标题、上次修改时间、操作数据值、源和运行手册执行的自动化 ID（仅举几例）。您可以使用状态筛选器进一步限制搜索结果。
- OpsItem 摘要报告

  OpsCenter 包含一个摘要报告页面，此页面显示以下部分：

  - 状态摘要：按状态（“打开”、“正在进行中”、“已解决”和“打开且正在进行中”）显示的 OpsItems 摘要。
  - 具有最常打开的 OpsItems源：最高 AWS 服务的细分，其具有打开的 OpsItems 。
  - OpsItems by source and age (按源和使用期限显示的 &OIS;)：按源和创建天数分组的 OpsItems 的计数。

  有关查看 OpsCenter 摘要报告的更多信息，请参阅 查看 OpsCenter 摘要报告。
- 支持日志记录和审计功能

  您可以通过与其他 AWS 服务集成来审计和记录 AWS 账户 中的 OpsCenter 用户操作。有关更多信息，请参阅 查看 OpsCenter 日志和报告。
- 支持通过控制台、CLI、PowerShell 和开发工具包等方式访问 OpsCenter 工具

  您可以通过 AWS Systems Manager 控制台、AWS Command Line Interface (AWS CLI)、AWS Tools for PowerShell 或您选择的 AWS SDK 来使用 OpsCenter。

OpsCenter 是否与我的现有用例管理系统集成？

OpsCenter 旨在补充您的现有用例管理系统。您可以使用公有 API 操作将 OpsItems 集成到您的现有用例管理系统中。您还可以维护当前系统中的手动生命周期工作流程，并使用 OpsCenter 作为调查和修正中心。

有关 OpsCenter 公有 API 操作的信息，请参阅《AWS Systems Manager API 参考》中的以下 API 操作。

- CreateOpsItem
- DescribeOpsItems
- GetOpsItem
- GetOpsSummary
- UpdateOpsItem

使用 OpsCenter 是否需要收取费用？

是。有关更多信息，请参阅 AWS Systems Manager 定价。

OpsCenter 是否使用我的本地和混合托管式节点？

是。若问题与为 Systems Manager 配置的本地托管式节点相关，您可以使用 OpsCenter 来调查和修正这些问题。有关为 Systems Manager 设置和配置本地服务器和虚拟机的更多信息，请参阅 使用 Systems Manager 管理混合和多云环境中的节点。

OpsCenter 的配额是多少？

在《Amazon Web Services 一般参考》的 Systems Manager service quotas 中，您可以查看所有 Systems Manager 工具的限额。除非另有说明，否则，每个配额是区域特定的。

设置 OpsCenter

AWS Systems Manager 可以使用集成设置体验来帮助您开始使用 OpsCenter 和 Explorer（都是 Systems Manager 中的工具）。Explorer 是一个可自定义的操作控制面板，用于报告有关您 AWS 资源的信息。在本文档中，Explorer 和 OpsCenter 设置称为集成设置。

必须使用“集成设置”借助 Explorer 设置 OpsCenter。集成设置仅在 AWS Systems Manager 控制台中可用。您无法以编程方式设置 Explorer 和 OpsCenter。有关更多信息，请参阅 Systems Manager Explorer 和 OpsCenter 入门。

开始前的准备工作

设置 OpsCenter 后，您将在 Amazon EventBridge 中启用会自动创建 OpsItems 的默认规则。下表描述了自动创建 OpsItems 的默认 EventBridge 规则。您可以在 OpsCenter 设置页面的 OpsItem 规则下禁用 EventBridge 规则。

> **Important**
>
> 您的账户需要为默认规则创建的 OpsItems 付费。有关更多信息，请参阅 AWS Systems Manager 定价。

Rule name（规则名称）	描述
SSMOpsItems-Autoscaling-instance-launch-failure	此规则将在启动 EC2 Auto Scaling 实例失败时创建 OpsItems。
SSMOpsItems-Autoscaling-instance-termination-failure	此规则将在终止 EC2 Auto Scaling 实例失败时创建 OpsItems。

Rule name（规则名称）	描述
SSMOpsItems-EBS-snapshot-copy-failed	此规则将在系统无法复制 Amazon Elastic Block Store（Amazon EBS）快照时创建 OpsItems。
SSMOpsItems-EBS-snapshot-creation-failed	此规则将在系统无法创建 Amazon EBS 快照时创建 OpsItems。
SSMOpsItems-EBS-volume-performance-issue	此规则对应于 AWS Health 跟踪规则。每当 Amazon EBS 卷出现性能问题（运行状况事件 = AWS_EBS_DEGRADED_EBS_VOLUME_PERFORMANCE ）时，此规则都会创建 OpsItems。
SSMOpsItems-EC2-issue	此规则对应于跟踪影响 AWS 服务或资源的意外事件的 AWS Health 跟踪规则。例如，当服务发出有关导致服务降级的运营问题，或者旨在提醒注意本地资源级别的问题的通信时，该规则就会创建 OpsItems。例如，此规则将为下列事件创建 OpsItem：AWS_EC2_OPERATIONAL_ISSUE 。
SSMOpsItems-EC2-scheduled-change	此规则对应于 AWS Health 跟踪规则。AWS 可为实例计划相关事件，例如重启、停止或启动实例。该规则将为 EC2 计划事件创建 OpsItems。有关计划事件的更多信息，请参阅《Amazon EC2 用户指南》中的 实例的计划事件。
SSMOpsItems-RDS-issue	此规则对应于跟踪影响 AWS 服务或资源的意外事件的 AWS Health 跟踪规则。例如，当服务发出有关导致服务降级的运营问题，或者旨在提醒注意本地资源级别的问题的通信时，该规则就会创建 OpsItems。例如，此规则将为下列事件创建 OpsItem：AWS_RDS_MYSQL_DATABASE_CRASHING_REPEATEDLY 、AWS_RDS_EXPORT_TASK_FAILED 和 AWS_RDS_CONNECTIVITY_ISSUE 。

Rule name（规则名称）	描述
SSMOpsItems-RDS-scheduled-change	此规则对应于 AWS Health 跟踪规则。该规则将为 Amazon RDS 计划事件创建 OpsItems。计划事件提供了有关 Amazon RDS 资源即将发生的更改的信息。某些事件可能会建议您执行操作以避免服务中断。其他事件则可能会自动发生，无需您执行任何操作。在执行计划的更改活动期间，您的资源可能暂时不可用。例如，此规则将为下列事件创建 OpsItem：`AWS_RDS_SYSTEM_UPGRADE_SCHEDULED` 和 `AWS_RDS_MAINTENANCE_SCHEDULED`。有关计划事件的更多信息，请参阅《AWS Health 用户指南》中的 事件类型类别。
SSMOpsItems-SSM-maintenance-window-execution-failed	此规则将在 Systems Manager 维护时段处理失败时创建 OpsItems。
SSMOpsItems-SSM-maintenance-window-execution-timedout	此规则在 Systems Manager 维护时段启动超时时创建 OpsItems。

请使用以下过程设置 OpsCenter。

要设置 OpsCenter，请按以下步骤操作：

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 OpsCenter。
3. 在 OpsCenter 主页上，选择开始使用。
4. 在 OpsCenter 设置页面上，选择启用此选项，以便让 Explorer 根据常用规则和事件配置 AWS Config 和 Amazon CloudWatch Events，以自动创建 OpsItems。如果您不选择此选项，OpsCenter 将保持禁用状态。

> **Note**
>
> Amazon EventBridge（前身为 Amazon CloudWatch Events）提供了 CloudWatch Events 的所有功能和一些新功能，如自定义事件总线、第三方事件源和架构注册表。

5. 请选择启用OpsCenter。

当您启用 OpsCenter 后，可以从设置中执行以下操作：

- 使用打开 CloudWatch 控制台按钮创建 CloudWatch 警报。有关更多信息，请参阅 将 CloudWatch 警报配置为创建 OpsItems。
- 启用运营洞察。有关更多信息，请参阅 分析运营洞察以减少 OpsItems。
- 启用 AWS Security Hub 结果警报。有关更多信息，请参阅 了解 OpsCenter 与 AWS Security Hub 的集成。

内容

- （可选）将 OpsCenter 设置为跨账户集中管理 OpsItems
- （可选）设置 Amazon SNS 以接收关于 OpsItems 的通知

（可选）将 OpsCenter 设置为跨账户集中管理 OpsItems

您可以使用 Systems Manager OpsCenter 在选定 AWS 区域 中的多个 AWS 账户 中集中管理 OpsItems。当您在 AWS Organizations 中设置组织后此功能可用。AWS Organizations 是一项账户管理服务，使您能够将多个 AWS 账户整合到您创建并集中管理的组织中。AWS Organizations 包含账户管理和整合账单功能，可利用这些功能更好地满足企业的预算、安全性和合规性需求。有关更多信息，请参阅《AWS Organizations 用户指南》中的什么是 AWS Organizations？

属于 AWS Organizations 管理账户的用户，可以设置 Systems Manager 的委派管理员账户。在 OpsCenter 的上下文中，委派管理员可以在成员账户中创建、编辑和查看 OpsItems。委派管理员还可以使用 Systems Manager 自动化运行手册批量解决 OpsItems 或修复与生成 OpsItems 的 AWS 资源相关的问题。

> **Note**
>
> 您只能指定一个账户作为 Systems Manager 的委派管理员。有关更多信息，请参阅 为 Systems Manager 创建 AWS Organizations 委派管理员。

Systems Manager 提供了以下 OpsCenter 设置方法，用于在多个 AWS 账户 之间集中管理 OpsItems。

- 快速设置功能：快速设置功能是 Systems Manager 中的一项工具，其简化了 Systems Manager 工具的设置和配置任务。有关更多信息，请参阅 AWS Systems Manager Quick Setup。

  OpsCenter 的快速设置功能可帮助您完成以下跨账户管理 OpsItems 的任务：

  - 将账户注册为委派管理员（如果尚未指定委派管理员）
  - 创建必需的 AWS Identity and Access Management（IAM）策略和角色
  - 指定一个 AWS Organizations 组织或组织单位（OU），委派管理员可以在其中跨账户管理 OpsItems

  有关更多信息，请参阅 （可选）使用 Quick Setup 配置 OpsCenter 以跨账户管理 OpsItems。

  > **Note**
  >
  > 并非所有当前 Systems Manager 可用的 AWS 区域 都提供快速设置功能。如果快速设置功能在某区域不可用，而您希望使用它在该区域配置 OpsCenter 从而跨多个账户集中管理 OpsItems，则必须使用手动方法。要查看快速设置功能可用的 AWS 区域 列表，请参阅 Quick Setup 在 AWS 区域 中的可用性。

- 手动设置：如果快速设置功能在某区域不可用，而您希望使用它在该区域配置 OpsCenter 从而跨账户集中管理 OpsItems，则您可以使用手动步骤来实现。有关更多信息，请参阅 （可选）将 OpsCenter 手动设置为跨账户集中管理 OpsItems。

（可选）使用 Quick Setup 配置 OpsCenter 以跨账户管理 OpsItems

Quick Setup是 AWS Systems Manager 中的一项工具，其简化了 Systems Manager 工具的设置和配置任务。OpsCenter 的Quick Setup可帮助您完成以下跨账户管理 OpsItems 的任务：

- 指定委派管理员账户
- 创建必需的 AWS Identity and Access Management（IAM）policy 和角色

- 指定一个 AWS Organizations 组织或成员账户的子集，委派管理员可以在其中跨账户管理 OpsItems

当您使用快速设置功能配置 OpsCenter 以跨账户管理 OpsItems 时，Quick Setup 将在指定账户中创建以下资源。这些资源授予指定账户权限以使用 OpsItems，并使用自动化运行手册来修复生成 OpsItems 的 AWS 资源的相关问题。

资源	账户
`AWSServiceRoleForAmazonSSM_AccountDiscovery` AWS Identity and Access Management（IAM）服务相关角色 有关该角色的更多信息，请参阅使用角色为 OpsCenter 和 Explorer 收集 AWS 账户 信息。	AWS Organizations 管理账户和委派管理员账户
`OpsItem-CrossAccountManagementRole` IAM 角色 `AWS-SystemsManager-AutomationAdministrationRole` IAM 角色	委托管理员帐户
`OpsItem-CrossAccountExecutionRole` IAM 角色 `AWS-SystemsManager-AutomationExecutionRole` IAM 角色 默认 OpsItem 组的 `AWS::SSM::ResourcePolicy` Systems Manager 资源策略（`OpsItemGroup`）	所有 AWS Organizations 成员账户

> **Note**
>
> 如果您之前配置了 OpsCenter 以便使用手动方法跨账户管理 OpsItems，则必须删除在该过程的步骤 4 和步骤 5 中创建的 AWS CloudFormation 堆栈或堆栈集。如果您完成以下步骤时您的账户中存在这些资源，则 Quick Setup 无法正确配置跨账户 OpsItem 管理。

使用快速设置功能配置 OpsCenter，以跨账户管理 OpsItems

1. 使用 AWS Organizations 管理账户登录 AWS Management Console。
2. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
3. 在导航窗格中，请选择 Quick Setup。
4. 选择库选项卡。
5. 滚动到底部并找到 OpsCenter 配置图块。选择创建。
6. 在 Quick SetupOpsCenter 页面上的委派管理员部分，输入账户 ID。如果您无法编辑此字段，则表示已经为 Systems Manager 指定了委派管理员账户。
7. 在 Targets 部分中，选择选项。如果选择自定义，则选择要在其中跨账户管理 OpsItems 的组织单位（OU）。
8. 选择创建。

Quick Setup 创建 OpsCenter 配置并将所需 AWS 资源部署到指定的 OU。

> **Note**
>
> 如果您不想跨多个账户管理 OpsItems，可以从 Quick Setup 中删除配置。删除配置时，Quick Setup 会删除最初部署配置时创建的以下 IAM policy 和角色：
>
> - 来自委派管理员账户的 `OpsItem-CrossAccountManagementRole`
> - 来自所有 Organizations 成员账户的 `OpsItem-CrossAccountExecutionRole` 和 `SSM::ResourcePolicy`
>
> Quick Setup 从所有组织单位和最初部署配置所在的 AWS 区域 移除配置。

排查 OpsCenter 的 Quick Setup 配置问题

本部分包含的信息可帮助您解决在使用 Quick Setup 配置跨账户 OpsItem 管理时遇到的问题。

主题

- 部署到这些堆栈集失败：delegatedAdmin
- Quick Setup 配置状态显示失败

部署到这些堆栈集失败：delegatedAdmin

创建 OpsCenter 配置时，Quick Setup 会在 Organizations 管理账户中部署两个 AWS CloudFormation 堆栈集。堆栈集使用以下前缀：`AWS-QuickSetup-SSMOpsCenter`。如果 Quick Setup 显示以下错误：`Deployment to these StackSets failed: delegatedAdmin`，请使用以下步骤修复此问题。

排查堆栈集失败的问题：delegatedAdmin 错误

1. 如果您在 Quick Setup 控制台的红色横幅中收到 `Deployment to these StackSets failed: delegatedAdmin` 错误，请登录到委派管理员账户和指定为 Quick Setup 主区域的 AWS 区域。
2. 通过以下网址打开 AWS CloudFormation 控制台：https://console.aws.amazon.com/cloudformation。
3. 请选择您的 Quick Setup 配置创建的堆栈。堆栈名称包括以下内容：AWS-QuickSetup-SSMOpsCenter。

   > **Note**
   >
   > CloudFormation 有时会删除失败的堆栈部署。如果堆栈未提供在 Stacks（堆栈）表中，请从筛选条件列表中选择 Deleted（已删除）。

4. 请查看 Status（状态）和 Status reason（状态原因）。有关堆栈状态的更多信息，请参阅 AWS CloudFormation 用户指南中的堆栈状态代码。
5. 要了解失败的确切步骤，请查看 Events（事件）选项卡并查看每个事件的状态。有关更多信息，请参阅《AWS CloudFormation User Guide》中的 Troubleshooting。

> **Note**
>
> 如果您无法使用 CloudFormation 问题排查步骤解决部署失败问题，请删除配置并再次尝试。

Quick Setup 配置状态显示失败

如果配置详细信息页面上的配置详细信息表显示配置状态为 `Failed`，请登录到失败的 AWS 账户 和区域。

排查创建 OpsCenter 配置时的 Quick Setup 失败问题

1. 登录到失败发生的 AWS 账户和 AWS 区域。
2. 通过以下网址打开 AWS CloudFormation 控制台：https://console.aws.amazon.com/cloudformation。
3. 请选择您的 Quick Setup 配置创建的堆栈。堆栈名称包括以下内容：AWS-QuickSetup-SSMOpsCenter。

   > **Note**
   >
   > CloudFormation 有时会删除失败的堆栈部署。如果堆栈未提供在 Stacks（堆栈）表中，请从筛选条件列表中选择 Deleted（已删除）。

4. 请查看 Status（状态）和 Status reason（状态原因）。有关堆栈状态的更多信息，请参阅 AWS CloudFormation 用户指南中的堆栈状态代码。
5. 要了解失败的确切步骤，请查看 Events（事件）选项卡并查看每个事件的状态。有关更多信息，请参阅《AWS CloudFormation User Guide》中的 Troubleshooting。

成员账户配置显示 ResourcePolicyLimitExceededException

如果堆栈状态显示 `ResourcePolicyLimitExceededException`，则表示该账户之前已使用手动方法载入了 OpsCenter 跨账户管理。要解决此问题，您必须删除在手动载入流程的步骤 4 和步骤 5 中创建的 AWS CloudFormation 堆栈或堆栈集。有关详细信息，请参阅《AWS CloudFormation User Guide》中的 Delete a stack set 和 Deleting a stack on the AWS CloudFormation console。

（可选）将 OpsCenter 手动设置为跨账户集中管理 OpsItems

本节介绍如何为跨账户 OpsItem 管理手动配置 OpsCenter。虽然仍支持此进程，但它已被使用 Systems Manager Quick Setup 的新进程所取代。有关更多信息，请参阅 （可选）使用 Quick Setup 配置 OpsCenter 以跨账户管理 OpsItems。

您可以设置一个中央账户为成员账户手动创建 OpsItems，并管理和修复这些 OpsItems。中央账户可以是 AWS Organizations 管理账户，也可以是 AWS Organizations 管理账户兼 Systems Manager 委派管理员账户。我们建议您使用 Systems Manager 委派管理员账户作为中央账户。您只能在配置 AWS Organizations 后使用此功能。

借助 AWS Organizations，您可以将多个 AWS 账户 整合到您创建并集中管理的组织中。中央账户用户可以同时为所有选定的成员账户创建 OpsItems 并管理这些 OpsItems。

使用本节中的过程在“组织”中启用 Systems Manager 服务主体，并配置 AWS Identity and Access Management（IAM）权限，以便跨账户使用 OpsItems。

主题

- 开始前的准备工作
- 步骤 1：创建资源数据同步
- 步骤 2：在 AWS Organizations 中启用 Systems Manager 服务主体
- 步骤 3：创建 AWSServiceRoleForAmazonSSM_AccountDiscovery 服务相关角色
- 步骤 4：配置用于跨账户使用 OpsItems 的权限
- 步骤 5：配置用于跨账户使用相关资源的权限

> **Note**
>
> 当在 OpsCenter 中跨账户工作时，仅支持 `/aws/issue` 类型的 OpsItems。

开始前的准备工作

在将 OpsCenter 设置为跨账户使用 OpsItems 之前，请确保已设置以下内容：

- Systems Manager 委托管理员账户。有关更多信息，请参阅 为 Explorer 配置委派管理员。
- 已在组织中设置和配置了一个组织。有关更多信息，请参阅 AWS Organizations 用户指南中的创建并管理组织。
- 您将 Systems Manager Automation 配置为跨多个 AWS 区域 和 AWS 账户运行自动化运行手册。有关更多信息，请参阅 在多个 AWS 区域 和账户中运行自动化。

步骤 1：创建资源数据同步

设置和配置 AWS Organizations 后，您可以通过创建资源数据同步在 OpsCenter 中为整个组织聚合 OpsItems。有关更多信息，请参阅 创建资源数据同步。创建同步时，请务必在添加账户部分中选择包括我的 AWS Organizations 配置中的所有账户选项。

步骤 2：在 AWS Organizations 中启用 Systems Manager 服务主体

要让用户能够跨账户使用 OpsItems，必须在 AWS Organizations 中启用 Systems Manager 服务主体。如果您以前使用其他工具为多账户方案配置了 Systems Manager，则 Organizations 中可能已经配置 Systems Manager 服务主体。从（AWS Command Line InterfaceAWS CLI）运行以下命令

以进行验证。如果您尚未为其他多账户方案配置 Systems Manager，请跳到下一个步骤，即在 AWS Organizations 中启用 Systems Manager 服务主体。

验证 AWS Organizations 中是否启用了 Systems Manager 服务主体

1. 在本地计算机上，下载最新版本的 AWS CLI。
2. 打开 AWS CLI，然后运行以下命令，以指定您的凭证和 AWS 区域。

```
aws configure
```

系统将提示您指定以下内容。在以下示例中，将每个*#######*替换为您自己的信息。

```
AWS Access Key ID [None]: key_name
AWS Secret Access Key [None]: key_name
Default region name [None]: region
Default output format [None]: ENTER
```

3. 运行以下命令以验证是否为 AWS Organizations 启用了 Systems Manager 服务主体。

```
aws organizations list-aws-service-access-for-organization
```

该命令会返回类似以下示例中显示的信息。

```
{
    "EnabledServicePrincipals": [
        {
            "ServicePrincipal":
 "member.org.stacksets.cloudformation.amazonaws.com",
            "DateEnabled": "2020-12-11T16:32:27.732000-08:00"
        },
        {
            "ServicePrincipal": "opsdatasync.ssm.amazonaws.com",
            "DateEnabled": "2022-01-19T12:30:48.352000-08:00"
        },
        {
            "ServicePrincipal": "ssm.amazonaws.com",
            "DateEnabled": "2020-12-11T16:32:26.599000-08:00"
        }
    ]
}
```

在 AWS Organizations 中启用 Systems Manager 服务主体

如果您以前未为“组织”配置 Systems Manager 服务主体，请按照以下步骤执行此操作。有关此命令的更多信息，请参阅AWS CLI 命令参考 中的 enable-aws-service-access 。

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。有关信息，请参阅安装 CLI 和配置 CLI。
2. 在本地计算机上，下载最新版本的 AWS CLI。
3. 打开 AWS CLI，然后运行以下命令，以指定您的凭证和 AWS 区域。

   ```
   aws configure
   ```

   系统将提示您指定以下内容。在以下示例中，将每个*******替换为您自己的信息。

   ```
   AWS Access Key ID [None]: key_name
   AWS Secret Access Key [None]: key_name
   Default region name [None]: region
   Default output format [None]: ENTER
   ```

4. 运行以下命令为 AWS Organizations 启用 Systems Manager 服务主体。

   ```
   aws organizations enable-aws-service-access --service-principal "ssm.amazonaws.com"
   ```

步骤 3：创建 **AWSServiceRoleForAmazonSSM_AccountDiscovery** 服务相关角色

服务相关角色（如 AWSServiceRoleForAmazonSSM_AccountDiscovery 角色）是一种与（AWS 服务如 Systems Manager）直接关联的独特类型的 IAM 角色。服务相关角色由服务预定义，具有服务代表您调用其他 AWS 服务 所需的所有权限。有关 AWSServiceRoleForAmazonSSM_AccountDiscovery 服务相关角色的更多信息，请参阅Systems Manager 账户发现的服务相关角色权限。

使用以下过程，通过利用 AWS CLI 创建 AWSServiceRoleForAmazonSSM_AccountDiscovery 服务相关角色。有关在此步骤中使用的命令的更多信息，请参阅AWS CLI 命令参考中的 create-service-linked-role。

创建 **AWSServiceRoleForAmazonSSM_AccountDiscovery** 服务相关角色

1. 登录 AWS Organizations 管理账户。
2. 登录“组织”管理账户时，运行以下命令。

```
aws iam create-service-linked-role \
    --aws-service-name accountdiscovery.ssm.amazonaws.com \
    --description "Systems Manager account discovery for AWS Organizations service-
linked role"
```

步骤 4：配置用于跨账户使用 OpsItems 的权限

使用 AWS CloudFormation 堆栈集创建 OpsItemGroup 资源策略和 IAM 执行角色，以授予用户跨账户使用 OpsItems 的权限。首先，请下载并解压 OpsCenterCrossAccountMembers.zip 文件。此文件包含 OpsCenterCrossAccountMembers.yaml AWS CloudFormation 模板文件。使用此模板创建堆栈集时，CloudFormation 会自动在账户中创建 OpsItemCrossAccountResourcePolicy 资源策略和 OpsItemCrossAccountExecutionRole 执行角色。有关创建堆栈集的更多信息，请参阅 AWS CloudFormation 用户指南中的 Create a stack set（创建堆栈集）。

> **Important**
>
> 请注意有关此任务的以下重要信息：
>
> - 您必须在登录到 AWS Organizations 管理账户时部署该堆栈集。
> - 您必须在登录到您要设置为目标的每个账户时重复此过程，以便跨账户（包括委托管理员账户）使用 OpsItems。
> - 如果您想在不同的 AWS 区域 中启用跨账户 OpsItems 管理，请在模板的 Specify regions（指定区域）部分中选择 Add all regions（添加所有区域）。选择加入区域不支持跨账户 OpsItem 管理。

步骤 5：配置用于跨账户使用相关资源的权限

OpsItem 可以包括有关受影响资源的详细信息，如 Amazon Elastic Compute Cloud（Amazon EC2）实例或 Amazon Simple Storage Service（Amazon S3）存储桶。您在“步骤 4”中创建的 OpsItemCrossAccountExecutionRole 执行角色将为 OpsCenter 提供只读权限，以便成员账户查看相关资源。您还必须创建 IAM 角色 以向管理账户提供查看相关资源和与之交互的权限，您将在本任务中完成此操作。

首先，请下载并解压 OpsCenterCrossAccountManagementRole.zip 文件。此文件包含 OpsCenterCrossAccountManagementRole.yaml AWS CloudFormation 模板文件。当您使用此模板创建堆栈时，CloudFormation 会自动在账户中创建

`OpsCenterCrossAccountManagementRole` IAM 角色。有关创建堆栈集的更多信息，请参阅《AWS CloudFormation 用户指南》中的在 AWS CloudFormation 控制台上创建堆栈。

> **Important**
>
> 请注意有关此任务的以下重要信息：
>
> - 如果您计划指定一个账户作为 OpsCenter 的委派管理员，请务必在创建堆栈时指定 AWS 账户。
> - 您必须在登录到 AWS Organizations 管理账户时执行此过程，并在登录到委托管理员账户时再次执行此过程。

（可选）设置 Amazon SNS 以接收关于 OpsItems 的通知

您可以将 OpsCenter 配置为在系统创建 OpsItem 或更新现有 OpsItem 时，向 Amazon Simple Notification Service（Amazon SNS）主题发送通知。

完成以下步骤以接收 OpsItems 的通知。

- 步骤 1：创建并订阅 Amazon SNS 主题
- 步骤 2：更新 Amazon SNS 访问策略
- 步骤 3：更新 AWS KMS 访问策略

> **Note**
>
> 如果您在“步骤 2”中打开了 AWS Key Management Service（AWS KMS）服务器端加密，则必须完成“步骤 3”。否则，可以跳过“步骤 3”。

- 步骤 4：打开默认 OpsItems 规则以针对新的 OpsItems 发送通知

步骤 1：创建并订阅 Amazon SNS 主题

要接收通知，您必须创建并订阅 Amazon SNS 主题。有关更多信息，请参阅 Amazon Simple Notification Service Developer Guide 中的创建 Amazon SNS 主题和订阅 Amazon SNS 主题。

> **Note**
>
> 如果您在多个 AWS 区域 或账户中使用 OpsCenter，则必须在每个要接收 OpsItem 通知的区域或账户中创建并订阅 Amazon SNS 主题。

步骤 2：更新 Amazon SNS 访问策略

您必须将 Amazon SNS 主题与 OpsItems 关联。使用以下过程设置 Amazon SNS 访问策略，以便 Systems Manager 能将 OpsItems 通知发布到您在步骤 1 中创建的 Amazon SNS 主题。

1. 访问 https://console.aws.amazon.com/sns/v3/home，登录 AWS Management Console 并打开 Amazon SNS 控制台。
2. 在导航窗格中，选择 Topics（主题）。
3. 选择您在“步骤 1”中创建的主题，然后选择编辑。
4. 展开 Access policy (访问策略)。
5. 将以下 `Sid` 数据块添加到现有策略。将每个*#######*替换为您自己的信息。

```
{
      "Sid": "Allow OpsCenter to publish to this topic",
      "Effect": "Allow",
      "Principal": {
        "Service": "ssm.amazonaws.com"
      },
      "Action": "SNS:Publish",
      "Resource": "arn:aws:sns:region:account ID:topic name", // Account ID of the
 SNS topic owner
      "Condition": {
      "StringEquals": {
        "AWS:SourceAccount": "account ID" //  Account ID of the OpsItem owner
      }
   }
}
```

> **Note**
>
> 使用 `aws:SourceAccount` 全局条件键来防止混淆代理场景。要使用此条件键，请将值设置为 OpsItem 所有者的账户 ID。有关更多信息，请参阅《IAM 用户指南》中的混淆代理。

6. 选择 Save changes（保存更改）。

现在，当创建或更新 OpsItems 时，系统会向 Amazon SNS 主题发送通知。

> **Important**
>
> 如果使用 AWS Key Management Service（AWS KMS）服务器端加密密钥配置 Amazon SNS 主题，则必须完成步骤 3。否则，可以跳过“步骤 3”。

步骤 3：更新 AWS KMS 访问策略

如果您为 Amazon SNS 主题启用了 AWS KMS 服务器端加密，则还必须更新您在配置主题时选择的 AWS KMS key 的访问策略。使用以下过程更新访问策略，以便 Systems Manager 能将 OpsItem 通知发布到您在“步骤 1”中创建的 Amazon SNS 主题。

> **Note**
>
> OpsCenter 不支持将 OpsItems 发布到使用 AWS 托管式密钥 配置的 Amazon SNS 主题。

1. 从 https://console.aws.amazon.com/kms 打开 AWS KMS 控制台。
2. 要更改 AWS 区域，请使用页面右上角的区域选择器。
3. 在导航窗格中，选择客户托管密钥。
4. 选择您在创建主题时选择的 KMS 密钥的 ID。
5. 在 Key Policy (密钥策略) 部分中，选择 Switch to policy view (切换到策略视图)。
6. 选择编辑。
7. 将以下 `Sid` 数据块添加到现有策略。将每个`#######`替换为您自己的信息。

```
{
```

```
    "Sid": "Allow OpsItems to decrypt the key",
    "Effect": "Allow",
    "Principal": {
      "Service": "ssm.amazonaws.com"
    },
    "Action": ["kms:Decrypt", "kms:GenerateDataKey*"],
     "Resource": "arn:aws:kms:region:account ID:key/key ID"
  }
```

在下面的示例中，在第 14 行输入新块。



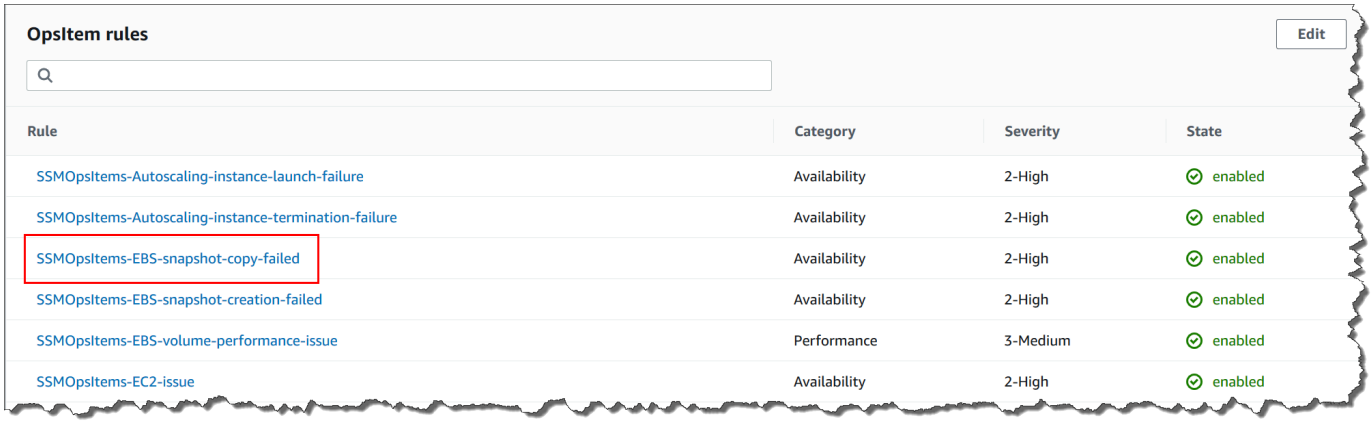
8. 选择 Save changes（保存更改）。

步骤 4：打开默认 OpsItems 规则以针对新的 OpsItems 发送通知

未使用 Amazon 资源名称（ARN）针对 Amazon SNS 通知配置 Amazon EventBridge 中的默认 OpsItems 规则。使用以下过程编辑 EventBridge 中的规则并输入 `notifications` 数据块。

将通知块添加到默认 OpsItem 规则

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 OpsCenter。
3. 选择 OpsItems 选项卡，然后选择 Configure sources (配置源)。
4. 选择您要使用 `notifications` 块配置的源规则的名称，如以下示例所示：



该规则将在 Amazon EventBridge 中打开。

5. 在规则详细信息页面的 Targets（目标）选项卡上，选择 Edit（编辑）。
6. 在 Additional settings（其他设置）部分，选择 Configure input transformer（配置输入转换器）。
7. 在模板框中，添加一个具有以下格式的 `notifications` 块：

```
"notifications":[{"arn":"arn:aws:sns:region:account ID:topic name"}],
```

以下为示例。

```
"notifications":[{"arn":"arn:aws:sns:us-west-2:1234567890:MySNSTopic"}],
```

在 `resources` 块之前输入通知块，如以下美国西部（俄勒冈州）（us-west-2）区域示例所示。

```
{
    "title": "EBS snapshot copy failed",
    "description": "CloudWatch Event Rule SSMOpsItems-EBS-snapshot-copy-failed was
 triggered. Your EBS snapshot copy has failed. See below for more details.",
    "category": "Availability",
    "severity": "2",
    "source": "EC2",
    "notifications": [{
        "arn": "arn:aws:sns:us-west-2:1234567890:MySNSTopic"
    }],
    "resources": <resources>,
    "operationalData": {
        "/aws/dedup": {
            "type": "SearchableString",
            "value": "{\"dedupString\":\"SSMOpsItems-EBS-snapshot-copy-failed\"}"
        },
```

```
        "/aws/automations": {
            "value": "[ { \"automationType\": \"AWS:SSM:Automation\",
 \"automationId\": \"AWS-CopySnapshot\" } ]"
        },
        "failure-cause": {
            "value": <failure - cause>
        },
        "source": {
            "value": <source>
        },
        "start-time": {
            "value": <start - time>
        },
        "end-time": {
            "value": <end - time>
        }
    }
}
```

8. 选择确认。
9. 选择下一步。
10. 选择下一步。
11. 选择更新规则。

当系统下次为默认规则创建 OpsItem 时，系统会向 Amazon SNS 主题发布通知。

将 OpsCenter 与其他 AWS 服务 集成

OpsCenter 是 AWS Systems Manager 中的一项工具，可以与多个 AWS 服务集成，用于诊断和修复 AWS 资源的问题。您必须先设置 AWS 服务，然后才能将其与 OpsCenter 集成。

默认情况下，以下 AWS 服务 将与 OpsCenter 集成，并可自动创建 OpsItems：

- Amazon CloudWatch
- Amazon CloudWatch Application Insights
- Amazon EventBridge
- AWS Config
- AWS Systems Manager Incident Manager

您必须将以下服务与 OpsCenter 集成，才能自动创建 OpsItems：

- Amazon DevOps Guru
- AWS Security Hub

当这些服务中的任何服务创建 OpsItem 时，您都可以从 OpsCenter 管理和修复 OpsItem。有关更多信息，请参阅管理 OpsItems和修复 OpsItem 问题。

有关每项 AWS 服务 及其如何与 OpsCenter 集成的更多信息，请参阅以下主题。

主题

- 了解 OpsCenter 与 Amazon CloudWatch 的集成
- 了解 OpsCenter 与 Amazon CloudWatch Application Insights 的集成
- 了解 OpsCenter 与 Amazon DevOps Guru 的集成
- 了解 OpsCenter 与 Amazon EventBridge 的集成
- 了解 OpsCenter 与 AWS Config 的集成
- 了解 OpsCenter 与 AWS Security Hub 的集成
- 了解 OpsCenter 与 Incident Manager 的集成

了解 OpsCenter 与 Amazon CloudWatch 的集成

Amazon CloudWatch 将监控您的 AWS 资源和服务，并显示有关您使用的每项 AWS 服务 的指标。当警报进入警报状态时，CloudWatch 将创建 OpsItem。例如，您可以配置告警，在 Application Load Balancer 生成的 HTTP 错误出现峰值时自动创建 OpsItem。

下面的列表中显示了一些警报，您可以在 CloudWatch 中配置它们以创建 OpsItems：

- Amazon DynamoDB：数据库读取和写入操作达到阈值
- Amazon EC2：CPU 使用率达到阈值
- AWS 账单：预估费用达到阈值
- Amazon EC2：实例未能通过状态检查
- Amazon Elastic Block Store (EBS)：磁盘空间使用率达到阈值

您可以创建警报或编辑现有警报，以创建 OpsItem。有关更多信息，请参阅 将 CloudWatch 警报配置为创建 OpsItems。

当您使用“集成设置”启用 OpsCenter 时，它会将 CloudWatch 与 OpsCenter 集成。

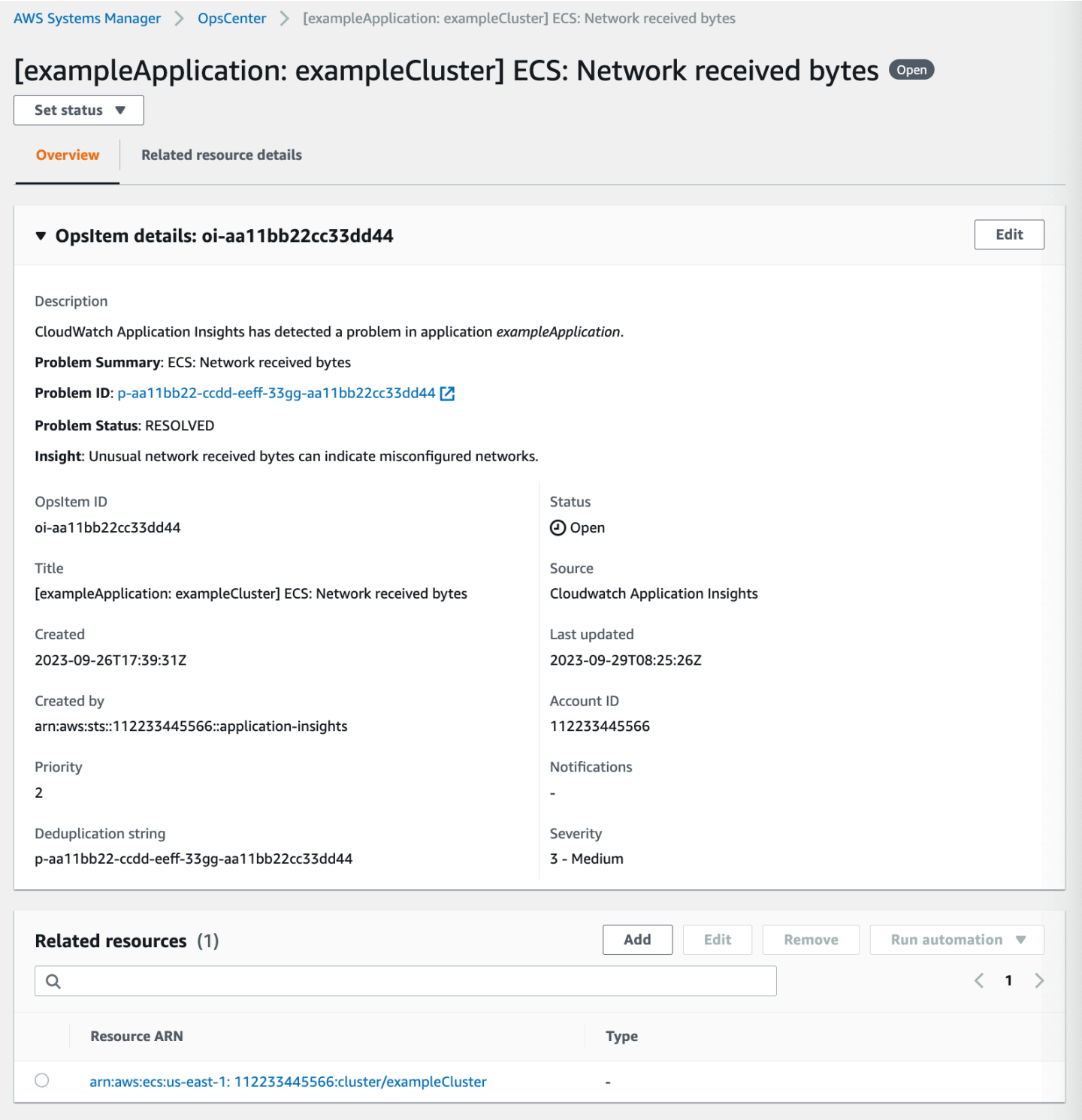
了解 OpsCenter 与 Amazon CloudWatch Application Insights 的集成

使用 Amazon CloudWatch Application Insights，您可以为应用程序资源设置最合适的监视器以持续分析数据，以便找出应用程序出现问题的迹象。在 CloudWatch Application Insights 中配置应用程序资源时，可以选择让系统在 OpsCenter 中创建 OpsItems。对于在应用程序中检测到的每个问题，都会在 OpsCenter 控制台上创建一个 OpsItem。有关信息，请参阅《Amazon CloudWatch 用户指南》中的设置、配置和管理用于监控的应用程序。

> **Note**
>
> 从 2023 年 10 月 16 日起，由 CloudWatch Application Insights 创建的 OpsItems 标题和描述现在使用以下经过改进的格式：

```
OpsItem title: [<APPLICATION NAME>: <RESOURCE ID>] <PROBLEM SUMMARY>

OpsItem description:

CloudWatch Application Insights has detected a problem in application <APPLICATION
 NAME>.
Problem summary: <PROBLEM SUMMARY>
Problem ID: <PROBLEM ID> (hyperlinks to the Application Insights problem summary page)
Problem Status: <PROBLEM STATUS>
Insight: <INSIGHT>
```

示例如下：



了解 OpsCenter 与 Amazon DevOps Guru 的集成

Amazon DevOps Guru 应用机器学习来分析运营数据、应用程序指标以及应用程序事件，以识别偏离正常运营模式的行为。如果您让 DevOps Guru 在 OpsCenter 中生成 OpsItem，则每个洞察都会生成一个新的 OpsItem。您可以使用 OpsCenter 来管理您的 OpsItems。

DevOps Guru 自动创建 OpsItems。借助 Systems Manager 中的Quick Setup工具，您可以启用 Amazon DevOps Guru 来创建 OpsItems。系统将通过使用 AWSServiceRoleForDevOpsGuru AWS Identity and Access Management（IAM）服务相关角色创建 OpsItems。

将 OpsCenter 与 DevOps Guru 集成

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Quick Setup。
3. 在自定义 DevOps Guru 配置选项页面上，选择库选项卡。
4. 在 DevOps Guru 窗格中，选择创建。
5. 对于配置选项，选择启用 AWS Systems Manager OpsItems。
6. 完成设置后，选择创建。

了解 OpsCenter 与 Amazon EventBridge 的集成

Amazon EventBridge 将提供事件流，这些事件描述了 AWS 资源中的更改。当您使用“集成设置”启用 OpsCenter 时，它会将 EventBridge 与 OpsCenter 集成，并启用默认 EventBridge 规则。根据这些规则，EventBridge 将创建 OpsItems。使用规则，您可以筛选事件并将其路由到 OpsCenter，以进行调查和修复。

> **Note**
>
> Amazon EventBridge（前身为 Amazon CloudWatch Events）提供了 CloudWatch Events 的所有功能和一些新功能，如自定义事件总线、第三方事件源和架构注册表。

以下是您可以在 EventBridge 中配置以创建 OpsItem 的一些规则：

- Security Hub：已发出安全提示
- Amazon DynamoDB 节流事件
- Amazon Elastic Compute Cloud Auto Scaling：启动实例失败
- Systems Manager：运行自动化失败
- AWS Health：计划维护提示
- Amazon EC2：实例状态已从运行更改为停止

根据您的要求，您可以创建规则或编辑现有规则来创建 OpsItems。有关如何编辑规则以创建 OpsItem 的说明，请参阅 配置 EventBridge 规则以创建 OpsItems。

了解 OpsCenter 与 AWS Config 的集成

AWS Config 可以提供关于您的 AWS 账户中 AWS 资源的配置的详细信息。

AWS Config 不直接与 OpsCenter 集成。相反，您可以创建一条向 Amazon EventBridge 发送事件的 AWS Config 规则，例如当 AWS Config 检测到不合规的实例时。然后，EventBridge 根据您创建的 EventBridge 规则评估该事件。如果规则匹配，EventBridge 会将事件转换为 OpsItem，并将其作为目标传输到 OpsCenter。

使用此 OpsItem，您可以追踪不合规资源的详细信息、记录调查操作，并提供对一致修复操作的访问权限。

相关信息

配置 EventBridge 规则以创建 OpsItems

使用 AWS Systems Manager OpsCenter 和 AWS Config 进行合规性监控

了解 OpsCenter 与 AWS Security Hub 的集成

AWS Security Hub 从跨 AWS 账户 和服务中收集为调查发现的安全数据。Security Hub 使用一组规则来检测和生成调查发现，可帮助您识别所管理资源的安全问题、确定其优先级并对其进行修复。按照本主题所述，配置集成后，Systems Manager 会为 OpsCenter 中的 Security Hub 调查发现创建 OpsItems。

> **Note**
>
> OpsCenter 与 Security Hub 进行双向集成。这意味着，如果您更新了与安全调查发现相关的 OpsItem 的状态或严重性字段，则系统会将更改与 Security Hub 同步。同样，对调查发现的任何更改都会自动在 OpsCenter 中的相应 OpsItems 中更新。
>
> 根据 Security Hub 调查发现创建 OpsItem 时，Security Hub 元数据会自动添加到 OpsItem 的操作数据字段中。如果删除此元数据，则双向更新将不再起作用。

默认情况下，Systems Manager 会针对重大和高严重性的调查发现创建 OpsItems。您可以手动配置 OpsCenter，以便针对中等和低严重性调查发现创建 OpsItems。OpsCenter 不会为信息性调查发现创建 OpsItems，因为这种结果无需修复。有关 Security Hub 严重性级别的更多信息，请参阅《AWS Security Hub API Reference》中的 Severity。

开始前的准备工作

在配置 OpsCenter 以便基于 Security Hub 的调查发现创建 OpsItems 之前，请确认您已完成 Security Hub 的设置任务。有关更多信息，请参阅 AWS Security Hub 用户指南中的设置 Security Hub。

当您将 Security Hub 与 OpsCenter 集成时，系统将使用 OpsItems IAM 服务相关角色创建 `AWSServiceRoleForSystemsManagerOpsDataSync`。有关该角色的更多信息，请参阅使用角色为 Explorer 创建 OpsData 和 OpsItems。

> **Warning**
>
> 请注意以下有关与 Security Hub 进行 OpsCenter 集成的定价的重要信息：
>
> - 如果您在配置 OpsCenter 和 Security Hub 集成时登录到 Security Hub 管理员账户，则系统会针对在管理员和所有成员账户中的调查发现创建 OpsItems。OpsItems 均在管理员账户中创建。根据各种因素，这可能会导致来自 AWS 的意外巨额账单。
>
>   如果您在配置集成时登录到成员账户，则系统仅针对该个人账户中的调查发现创建 OpsItems。有关 Security Hub 管理员账户、成员账户及其与 EventBridge 调查发现事件源关系的更多信息，请参阅《AWS Security Hub User Guide》中的 Types of Security Hub integration with EventBridge。
> - 对于创建 OpsItem 的每个调查发现，都将按常规价格向您收取创建 OpsItem 的费用。如果您编辑了 OpsItem 或在 Security Hub 中更新了相应的调查发现（这会触发 OpsItem 更新），则也会产生费用。
> - 通过与 AWS Security Hub 集成创建的 OpsItems 暂不受每个区域每个账户 50 万个 OpsItems 的最大配额限制。因此，Security Hub 警报可以在账户的每个区域中创建超过 50 万个可收费 OpsItems。
>
>   因此，对于高负载生产环境，建议将 Security Hub 调查发现范围限定为仅高严重性问题。

配置 OpsCenter 以便为 Security Hub 调查发现创建 OpsItems

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 OpsCenter。
3. 选择设置。
4. 在 Security Hub 调查发现部分，选择编辑。

5. 选择滑块将已禁用更改为已启用。
6. 如果您希望系统针对中等或低严重性调查发现创建 OpsItems，请切换这些选项。
7. 选择 Save (保存) 以保存您的配置。

如果您不再希望系统为 Security Hub 调查发现创建 OpsItems，请使用以下步骤。

停止接收 Security Hub 调查发现的 OpsItems

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 OpsCenter。
3. 选择设置。
4. 在 Security Hub 调查发现部分，选择编辑。
5. 选择滑块将已启用更改为已禁用。如果无法切换滑块，则说明您的 AWS 账户 尚未启用 Security Hub。
6. 选择保存以保存您的配置。OpsCenter 不再根据 Security Hub 的调查发现创建 OpsItems。

> **Important**
>
> Systems Manager 委派的管理员或 AWS Organizations 管理账户可以通过在 Explorer 中创建资源数据同步，从而在 OpsCenter 中为多个账户和 AWS 区域 启用 Security Hub 调查发现。如果在 Explorer 中启用了 Security Hub 源，并且存在以成员账户（禁用了 Security Hub）为目标的资源数据同步，则以管理员选择的设置优先。OpsCenter 继续为 Security Hub 调查发现创建 OpsItems。要停止在被资源数据同步视作目标的成员账户中为 Security Hub 调查发现创建 OpsItems，请联系管理员移除您在资源数据同步中的账户，或者在 Explorer 中关闭 Security Hub 源。有关在 Explorer 中更改设置的信息，请参阅 编辑 Systems Manager Explorer 数据源。

了解 OpsCenter 与 Incident Manager 的集成

Incident Manager（AWS Systems Manager 中的一项工具）提供了一个事件管理控制台，可帮助缓解影响您 AWS 托管应用程序的事件并从中恢复。事件是指任何计划外的服务中断或质量下降。在设置并配置 Incident Manager 后，系统将在 OpsCenter 中自动创建 OpsItems。

当系统在 Incident Manager 中创建事件时，它还会在 OpsCenter 中创建一个 OpsItem，并将该事件显示为相关项目。如果 OpsItem 已经存在，则 Incident Manager 不会创建 OpsItem。第一个 OpsItem 称为父 OpsItem。如果某一事件的规模和范围扩大，则您可以向现有 OpsItem 中添加多个事件。如果需要，您可以为 OpsItem 手动创建事件。在事件关闭后，您可以在 Incident Manager 中创建分析，以便查看和改进相似问题的修复过程。

默认情况下，OpsCenter 将与 Incident Manager 集成。如果未设置 Incident Manager，则 OpsCenter 页面将显示一条消息，以设置 Incident Manager。当 Incident Manager 创建 OpsItem 时，您可以从 OpsCenter 管理和修复 OpsItem。有关为 OpsItem 创建事件的说明，请参阅 为 OpsItem 创建事件。

创建 OpsItems

在设置好 OpsCenter（AWS Systems Manager 中的一项工具）并将其与 AWS 服务集成后，AWS 服务会根据默认规则、事件或警报自动创建 OpsItems。

您可以查看默认 Amazon EventBridge 规则的状态和严重性级别。如果需要，您可以从 Amazon EventBridge 创建或编辑这些规则。您还可以从 Amazon CloudWatch 查看警报，以及创建或编辑警报。使用规则和警报，您可以配置要为其自动生成 OpsItems 的事件。

当系统创建 OpsItem 时，它将处于打开状态。当您开始 OpsItem 的调查时，可将状态更改为正在进行；在修复 OpsItem 后，可将状态更改为已解决。关于如何在 AWS 服务 中配置警报和规则以创建 OpsItems，以及如何手动创建 OpsItem 的更多信息，请参阅以下主题。

主题

- 配置 EventBridge 规则以创建 OpsItems
- 将 CloudWatch 警报配置为创建 OpsItems
- 手动创建 OpsItems

配置 EventBridge 规则以创建 OpsItems

当 Amazon EventBridge 收到事件时，它将根据默认规则创建新的 OpsItem。您可以创建规则或编辑现有规则，以将 OpsCenter 设置为 EventBridge 事件的目标。有关如何创建事件规则的信息，请参阅《Amazon EventBridge 用户指南》中的为 AWS 服务 创建规则。

配置 EventBridge 规则以在 OpsCenter 中创建 OpsItems

1. 打开位于 https://console.aws.amazon.com/events/ 的 Amazon EventBridge 控制台。
2. 在导航窗格中，选择规则。

3. 在 Rule（规则）页面上，对于 Event bus（事件总线），选择 default（默认）。
4. 对于规则，通过选中规则名称旁边的复选框来选择规则。
5. 选择规则的名称以打开其详细信息页面。在规则详细信息中，验证是否已将状态设置为已启用。

> **Note**
>
> 如果需要，您可以使用该页面右上角的编辑来更新状态。

6. 选择目标选项卡。
7. 在 Targets (目标) 选项卡上，选择 Edit (编辑)。
8. 对于目标类型，选择 AWS 服务。
9. 对于 Select a target（选择一个目标），选择 Systems Manager OpsItem。
10. 对于许多目标类型，EventBridge 需要权限以便将事件发送到目标。在这些情况下，EventBridge 可以创建运行规则所需的 AWS Identity and Access Management (IAM) 角色：

    - 若要自动创建 IAM 角色，请选择 Create a new role for this specific resource (为此特定资源创建新角色)。
    - 要使用您创建的 IAM 角色向 Eventbridge 授予在 OpsCenter 中创建 OpsItems 的权限，请选择Use existing role（使用现有角色）。
11. 在其他设置中，对于配置目标输入，请选择输入转换器。

    您可以使用输入转换器选项为 OpsItems 指定重复数据删除字符串和其他重要信息，如标题和严重性。
12. 选择 Configure input transformer（配置输入转换器）。
13. 在目标输入转换器中，对于输入路径，请指定要从触发事件中解析的值。例如，要从触发该规则的事件中解析开始时间、结束时间和其他详细信息，请使用下面的 JSON。

```
{
    "end-time": "$.detail.EndTime",
    "failure-cause": "$.detail.cause",
    "resources": "$.resources[0]",
    "source": "$.detail.source",
    "start-time": "$.detail.StartTime"
}
```

14. 对于 Template（模板），请指定要发送到目标的信息。例如，使用下面的 JSON 将信息传递到 OpsCenter。此信息将用于创建 OpsItem。

> **Note**
>
> 如果输入模板采用 JSON 格式，则模板中的对象值不能包含引号。例如，资源、故障原因、来源、开始时间和结束时间的值不能以引号表示。

```
{
    "title": "EBS snapshot copy failed",
    "description": "CloudWatch Event Rule SSMOpsItems-EBS-snapshot-copy-failed was
 triggered. Your EBS snapshot copy has failed. See below for more details.",
    "category": "Availability",
    "severity": "2",
    "source": "EC2",
    "operationalData": {
        "/aws/dedup": {
            "type": "SearchableString",
            "value": "{\"dedupString\":\"SSMOpsItems-EBS-snapshot-copy-failed\"}"
        },
        "/aws/automations": {
            "value": "[ { \"automationType\": \"AWS:SSM:Automation\",
 \"automationId\": \"AWS-CopySnapshot\" } ]"
        },
        "failure-cause": {
            "value": <failure-cause>
        },
        "source": {
            "value": <source>
        },
        "start-time": {
            "value": <start-time>
        },
        "end-time": {
            "value": <end-time>
        },
         },
        "resources": {
            "value": <resources>
        }
    }
}
```

有关这些字段的更多信息，请参阅《Amazon EventBridge 用户指南》中的转换目标输入。

15. 选择确认。
16. 选择下一步。
17. 选择下一步。
18. 选择更新规则。

在从事件创建 OpsItem 后，您可以通过打开 OpsItem 并向下滚动到私有操作数据部分来查看事件详细信息。有关如何配置 OpsItem 中的选项的信息，请参阅 管理 OpsItems。

将 CloudWatch 警报配置为创建 OpsItems

在 OpsCenter（AWS Systems Manager 中的一项工具）的集成设置中，您可以启用 Amazon CloudWatch，根据常见警报自动创建 OpsItems。您可以创建警报或编辑现有警报，以在 OpsCenter 中创建 OpsItems。

当您将警报配置为创建 OpsItems 时，CloudWatch 将在 AWS Identity and Access Management（IAM）中创建新的服务相关角色。该新角色名为 `AWSServiceRoleForCloudWatchAlarms_ActionSSM`。有关 CloudWatch 服务相关角色的更多信息，请参阅《Amazon CloudWatch 用户指南》中的为 CloudWatch 使用服务相关角色。

当 CloudWatch 警报生成 OpsItem 时，OpsItem 将显示 CloudWatch 警报 -“***alarm_name***”处于警报状态。

要查看有关特定 OpsItem 的详细信息，请选择 OpsItem，然后选择相关资源详细信息 选项卡。您可以手动编辑 OpsItems，以更改诸如严重性或类别等详细信息。但是，当您编辑警报的严重性或类别时，Systems Manager 将无法更新已通过警报创建的 OpsItems 的严重性或类别。如果某个告警创建了 OpsItem，并且您指定了重复数据删除字符串，则即使您在 CloudWatch 中编辑此告警，它也不会创建额外的 OpsItems。如果已在 OpsCenter 中解决了 OpsItem 的问题，CloudWatch 将创建一个新的 OpsItem。

有关配置 CloudWatch 警报的更多信息，请参阅以下主题。

主题

- 配置 CloudWatch 警报以创建 OpsItems（控制台）
- 将现有 CloudWatch 警报配置为创建 OpsItems（以编程方式）

配置 CloudWatch 警报以创建 OpsItems（控制台）

您可以手动创建警报或更新现有警报，以从 Amazon CloudWatch 创建 OpsItems。

创建 CloudWatch 警报并将 Systems Manager 配置为该警报的目标

1. 完成《Amazon CloudWatch 用户指南》中根据静态阈值创建 CloudWatch 警报中指定的步骤 1-9。
2. 在 Systems Manager 操作部分中，选择添加 Systems Manager OpsCenter 操作。
3. 选择 OpsItems。
4. 对于严重性，请在 1 到 4 之间进行选择。
5. （可选）对于类别，为 OpsItem 选择一个类别。
6. 完成《Amazon CloudWatch 用户指南》中根据静态阈值创建 CloudWatch 警报中指定的步骤 11-13。
7. 选择 Next (下一步) 并完成向导。

要编辑现有告警，并将 Systems Manager 配置为此告警的目标，请执行以下步骤：

1. 通过 https://console.aws.amazon.com/cloudwatch/ 打开 CloudWatch 控制台。
2. 在导航窗格中，选择警报。
3. 选择告警，然后依次选择 Actions (操作) 和 Edit (编辑)。
4. （可选）更改 Metrics (指标) 和 Conditions (条件) 部分中的设置，然后选择 Next (下一步)。
5. 在 Systems Manager 部分，选择 Add Systems Manager OpsCenter 操作。
6. 对于 Severity (严重性)，选择一个数字。

> **Note**
>
> 严重性是用户定义的值。您或您的组织可以确定每个严重性级别的值的含义以及与每个严重性级别相关联的任何服务等级协议。

7. （可选）对于 Category (类别)，选择一个选项。
8. 选择 Next (下一步) 并完成向导。

将现有 CloudWatch 警报配置为创建 OpsItems（以编程方式）

您可以将 Amazon CloudWatch 警报配置为使用 AWS Command Line Interface（AWS CLI）、AWS CloudFormation 模板或 Java 代码段以编程方式创建 OpsItems。

主题

- 开始前的准备工作
- 将 CloudWatch 警报配置为创建 OpsItems（AWS CLI）
- 将 CloudWatch 警报配置为创建或更新 OpsItems（CloudFormation）
- 将 CloudWatch 警报配置为创建或更新 OpsItems（Java）

开始前的准备工作

如果您以编程方式编辑现有警报或创建用于创建 OpsItems 的警报，则必须指定 Amazon 资源名称（ARN）。此 ARN 将 Systems Manager OpsCenter 标识为通过告警创建的 OpsItems 目标。您可以自定义 ARN，使通过告警创建的 OpsItems 包含特定信息，如严重性或类别。每个 ARN 包含下表中描述的信息。

参数	详细信息
Region（必填）	告警所在的 AWS 区域。例如：us-west-2 。有关如何在 AWS 区域 中使用OpsCenter 的更多信息，请参阅 AWS Systems Manager 端点和配额。
account_ID （必填）	创建告警时使用的同一 AWS 账户 ID。例如：123456789012 。账户 ID 必须后跟冒号（:）和参数 opsitem，如以下示例所示。
severity（必填）	由用户定义的通过告警创建的 OpsItems 的严重性级别。有效值：1、2、3、4
Category（可选）	通过告警创建的 OpsItems 的类别。有效值：Availability 、Cost、Performance 、Recovery 和 Security。

使用以下句法创建 ARN。此 ARN 不包括可选的 Category 参数。

```
arn:aws:ssm:Region:account_ID:opsitem:severity
```

以下为示例。

```
arn:aws:ssm:us-west-2:123456789012:opsitem:3
```

要创建使用可选的 Category 参数的 ARN，请使用以下句法：

```
arn:aws:ssm:Region:account_ID:opsitem:severity#CATEGORY=category_name
```

以下为示例。

```
arn:aws:ssm:us-west-2:123456789012:opsitem:3#CATEGORY=Security
```

将 CloudWatch 警报配置为创建 OpsItems（AWS CLI）

此命令要求您为 alarm-actions 参数指定 ARN。有关如何创建 ARN 的信息，请参阅 开始前的准备工作。

将 CloudWatch 警报配置为创建 OpsItems（AWS CLI）

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本。
2. 运行以下命令以收集要配置的告警的信息。

   ```
   aws cloudwatch describe-alarms --alarm-names "alarm name"
   ```
3. 运行以下命令以更新告警。将每个#######替换为您自己的信息。

   ```
   aws cloudwatch put-metric-alarm --alarm-name name \
   --alarm-description "description" \
   --metric-name name --namespace namespace \
   --statistic statistic --period value --threshold value \
   --comparison-operator value \
   --dimensions "dimensions" --evaluation-periods value \
       --alarm-actions
    arn:aws:ssm:Region:account_ID:opsitem:severity#CATEGORY=category_name \
   --unit unit
   ```

以下为示例。

Linux & macOS

```
aws cloudwatch put-metric-alarm --alarm-name cpu-mon \
--alarm-description "Alarm when CPU exceeds 70 percent" \
--metric-name CPUUtilization --namespace AWS/EC2 \
--statistic Average --period 300 --threshold 70 \
--comparison-operator GreaterThanThreshold \
--dimensions "Name=InstanceId,Value=i-12345678" --evaluation-periods 2 \
--alarm-actions arn:aws:ssm:us-east-1:123456789012:opsitem:3#CATEGORY=Security \
--unit Percent
```

Windows

```
aws cloudwatch put-metric-alarm --alarm-name cpu-mon ^
--alarm-description "Alarm when CPU exceeds 70 percent" ^
--metric-name CPUUtilization --namespace AWS/EC2 ^
--statistic Average --period 300 --threshold 70 ^
--comparison-operator GreaterThanThreshold ^
--dimensions "Name=InstanceId,Value=i-12345678" --evaluation-periods 2 ^
--alarm-actions arn:aws:ssm:us-east-1:123456789012:opsitem:3#CATEGORY=Security ^
--unit Percent
```

将 CloudWatch 警报配置为创建或更新 OpsItems（CloudFormation）

本部分包括多个 AWS CloudFormation 模板，您可以使用这些模板将 CloudWatch 警报配置为自动创建或更新 OpsItems。每个模板均要求您为 `AlarmActions` 参数指定 ARN。有关如何创建 ARN 的信息，请参阅 开始前的准备工作。

指标警报 - 使用以下 CloudFormation 模板创建或更新 CloudWatch 指标警报。此模板中指定的警报用于监控 Amazon Elastic Compute Cloud（Amazon EC2）实例状态检查。如果告警进入 `ALARM` 状态，则在 OpsCenter 中创建 OpsItem。

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters" : {
    "RecoveryInstance" : {
```

```
          "Description" : "The EC2 instance ID to associate this alarm with.",
          "Type" : "AWS::EC2::Instance::Id"
        }
      },
      "Resources": {
        "RecoveryTestAlarm": {
          "Type": "AWS::CloudWatch::Alarm",
          "Properties": {
            "AlarmDescription": "Run a recovery action when instance status check fails
 for 15 consecutive minutes.",
            "Namespace": "AWS/EC2" ,
            "MetricName": "StatusCheckFailed_System",
            "Statistic": "Minimum",
            "Period": "60",
            "EvaluationPeriods": "15",
            "ComparisonOperator": "GreaterThanThreshold",
            "Threshold": "0",
            "AlarmActions": [ {"Fn::Join" : ["",
 ["arn:arn:aws:ssm:Region:account_ID:opsitem:severity#CATEGORY=category_name",
 { "Ref" : "AWS::Partition" }, ":ssm:", { "Ref" : "AWS::Region" }, { "Ref" : "AWS::
 AccountId" }, ":opsitem:3" ]]} ],
            "Dimensions": [{"Name": "InstanceId","Value": {"Ref": "RecoveryInstance"}}]
          }
        }
      }
    }
```

复合警报 - 使用以下 CloudFormation 模板创建或更新复合警报。复合告警由多个指标告警组成。如果告警进入 ALARM 状态，则在 OpsCenter 中创建 OpsItem。

```
"Resources":{
       "HighResourceUsage":{
          "Type":"AWS::CloudWatch::CompositeAlarm",
          "Properties":{
             "AlarmName":"HighResourceUsage",
             "AlarmRule":"(ALARM(HighCPUUsage) OR ALARM(HighMemoryUsage)) AND NOT
 ALARM(DeploymentInProgress)",

 "AlarmActions":"arn:aws:ssm:Region:account_ID:opsitem:severity#CATEGORY=category_name",
             "AlarmDescription":"Indicates that the system resource usage is high while
 no known deployment is in progress"
          },
          "DependsOn":[
```

```
            "DeploymentInProgress",
            "HighCPUUsage",
            "HighMemoryUsage"
         ]
      },
      "DeploymentInProgress":{
         "Type":"AWS::CloudWatch::CompositeAlarm",
         "Properties":{
            "AlarmName":"DeploymentInProgress",
            "AlarmRule":"FALSE",
            "AlarmDescription":"Manually updated to TRUE/FALSE to disable other
alarms"
         }
      },
      "HighCPUUsage":{
         "Type":"AWS::CloudWatch::Alarm",
         "Properties":{
            "AlarmDescription":"CPUusageishigh",
            "AlarmName":"HighCPUUsage",
            "ComparisonOperator":"GreaterThanThreshold",
            "EvaluationPeriods":1,
            "MetricName":"CPUUsage",
            "Namespace":"CustomNamespace",
            "Period":60,
            "Statistic":"Average",
            "Threshold":70,
            "TreatMissingData":"notBreaching"
         }
      },
      "HighMemoryUsage":{
         "Type":"AWS::CloudWatch::Alarm",
         "Properties":{
            "AlarmDescription":"Memoryusageishigh",
            "AlarmName":"HighMemoryUsage",
            "ComparisonOperator":"GreaterThanThreshold",
            "EvaluationPeriods":1,
            "MetricName":"MemoryUsage",
            "Namespace":"CustomNamespace",
            "Period":60,
            "Statistic":"Average",
            "Threshold":65,
            "TreatMissingData":"breaching"
         }
      }
```

```
    }
```

将 CloudWatch 警报配置为创建或更新 OpsItems（Java）

本部分包括多个 Java 代码段，您可以使用这些代码段将 CloudWatch 警报配置为自动创建或更新 OpsItems。每个代码段均要求您为 `validSsmActionStr` 参数指定 ARN。有关如何创建 ARN 的信息，请参阅 开始前的准备工作。

特定警报 - 使用以下 Java 代码段创建或更新 CloudWatch 警报。此模板中指定的告警用于监控 Amazon EC2 实例状态检查。如果告警进入 `ALARM` 状态，则在 OpsCenter 中创建 OpsItem。

```
import com.amazonaws.services.cloudwatch.AmazonCloudWatch;
    import com.amazonaws.services.cloudwatch.AmazonCloudWatchClientBuilder;
    import com.amazonaws.services.cloudwatch.model.ComparisonOperator;
    import com.amazonaws.services.cloudwatch.model.Dimension;
    import com.amazonaws.services.cloudwatch.model.PutMetricAlarmRequest;
    import com.amazonaws.services.cloudwatch.model.PutMetricAlarmResult;
    import com.amazonaws.services.cloudwatch.model.StandardUnit;
    import com.amazonaws.services.cloudwatch.model.Statistic;

    private void putMetricAlarmWithSsmAction() {
        final AmazonCloudWatch cw =
                AmazonCloudWatchClientBuilder.defaultClient();

        Dimension dimension = new Dimension()
                .withName("InstanceId")
                .withValue(instanceId);

        String validSsmActionStr =
 "arn:aws:ssm:Region:account_ID:opsitem:severity#CATEGORY=category_name";

        PutMetricAlarmRequest request = new PutMetricAlarmRequest()
                .withAlarmName(alarmName)
                .withComparisonOperator(
                        ComparisonOperator.GreaterThanThreshold)
                .withEvaluationPeriods(1)
                .withMetricName("CPUUtilization")
                .withNamespace("AWS/EC2")
                .withPeriod(60)
                .withStatistic(Statistic.Average)
                .withThreshold(70.0)
                .withActionsEnabled(false)
                .withAlarmDescription(
```

```
                        "Alarm when server CPU utilization exceeds 70%")
                .withUnit(StandardUnit.Seconds)
                .withDimensions(dimension)
                .withAlarmActions(validSsmActionStr);

        PutMetricAlarmResult response = cw.putMetricAlarm(request);
    }
```

更新所有警报 - 使用以下 Java 代码段更新您的 AWS 账户 中的所有 CloudWatch 警报，以便在警报进入 ALARM 状态时创建 OpsItems。

```
import com.amazonaws.services.cloudwatch.AmazonCloudWatch;
    import com.amazonaws.services.cloudwatch.AmazonCloudWatchClientBuilder;
    import com.amazonaws.services.cloudwatch.model.DescribeAlarmsRequest;
    import com.amazonaws.services.cloudwatch.model.DescribeAlarmsResult;
    import com.amazonaws.services.cloudwatch.model.MetricAlarm;

    private void listMetricAlarmsAndAddSsmAction() {
        final AmazonCloudWatch cw = AmazonCloudWatchClientBuilder.defaultClient();

        boolean done = false;
        DescribeAlarmsRequest request = new DescribeAlarmsRequest();

        String validSsmActionStr =
 "arn:aws:ssm:Region:account_ID:opsitem:severity#CATEGORY=category_name";

        while(!done) {

            DescribeAlarmsResult response = cw.describeAlarms(request);

            for(MetricAlarm alarm : response.getMetricAlarms()) {
                // assuming there are no alarm actions added for the metric alarm
                alarm.setAlarmActions(ImmutableList.of(validSsmActionStr));
            }

            request.setNextToken(response.getNextToken());

            if(response.getNextToken() == null) {
                done = true;
            }
        }
    }
```

手动创建 OpsItems

当您发现操作问题时，可以通过 OpsCenter（AWS Systems Manager 中的一项工具）手动创建 OpsItem 来管理和解决该问题。

如果您为跨账户管理设置了 OpsCenter，则 Systems Manager 委托管理员或 AWS Organizations 管理账户可以为成员账户创建 OpsItems。有关更多信息，请参阅（可选）将 OpsCenter 手动设置为跨账户集中管理 OpsItems。

您可以使用 AWS Systems Manager 控制台、AWS Command Line Interface（AWS CLI）或 AWS Tools for Windows PowerShell 创建 OpsItems。

主题

- 手动创建 OpsItems（控制台）
- 手动创建 OpsItems（AWS CLI）
- 手动创建 OpsItems（PowerShell）

手动创建 OpsItems（控制台）

您可以使用 AWS Systems Manager 控制台手动创建 OpsItems。当您创建 OpsItem 时，它将显示在您的 OpsCenter 账户中。如果您为跨账户管理设置了 OpsCenter，则 OpsCenter 将为委托管理员或管理账户提供为选定成员账户创建 OpsItems 的选项。有关更多信息，请参阅（可选）将 OpsCenter 手动设置为跨账户集中管理 OpsItems。

使用 AWS Systems Manager 控制台创建 OpsItem

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 OpsCenter。
3. 选择 Create (创建) OpsItem。如果您没有看到该按钮，请选择 OpsItems 选项卡，然后选择 Create OpsItem (创建 OpsItem)。
4. （可选）选择其他账户，然后选择要在其中创建 OpsItem 的账户。

   > **Note**
   >
   > 如果您要为成员账户创建 OpsItems，则需要执行此步骤。

5. 对于 Title (标题)，输入一个描述性名称以帮助您了解 OpsItem 的用途。

6. 对于 Source (源)，请输入受影响的 AWS 资源的类型或其他源信息，以帮助用户了解 OpsItem 的源。

   > **Note**
   >
   > 创建 OpsItem 后，您无法编辑 Source (源) 字段。

7. （可选）对于 Priority (优先级)，选择优先级级别。
8. （可选）对于 Severity (严重性)，选择严重性级别。
9. （可选）对于 Category (类别)，选择一个类别。
10. 对于 Description (描述)，输入有关此 OpsItem 的信息，包括（如果适用）再现此问题的步骤。

    > **Note**
    >
    > 控制台支持 OpsiteM 描述字段中的大多数 Markdown 格式。有关更多信息，请参阅《AWS Management Console 入门指南》中的在控制台中使用 Markdown。

11. 对于重复数据删除字符串，输入系统可以用于检查重复 OpsItems 的单词。有关重复数据删除字符串的更多信息，请参阅 管理重复的 OpsItems。
12. （可选）对于通知，指定在更新此 OpsItem 时要将通知发送到的 Amazon SNS 主题的 Amazon 资源名称（ARN）。您必须指定一个与 OpsItem 位于同一 AWS 区域 中的 Amazon SNS ARN。
13. （可选）对于相关资源，选择添加以指定受影响资源和任何相关资源的 ID 或 ARN。
14. 选择创建OpsItem。

如果成功，该页面将显示 OpsItem。当委托管理员或管理账户为选定成员账户创建 OpsItem 时，新的 OpsItems 将显示在管理员和成员账户的 OpsCenter 中。有关如何配置 OpsItem 中的选项的信息，请参阅 管理 OpsItems。

手动创建 OpsItems（AWS CLI）

以下过程介绍了使用 AWS Command Line Interface (AWS CLI) 创建 OpsItem 的方法。

使用 AWS CLI 创建 OpsItem

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 打开 AWS CLI 并运行以下命令以创建 OpsItem。将每个*******替换为您自己的信息。

```
aws ssm create-ops-item \
    --title "Descriptive_title" \
    --description "Information_about_the_issue" \
    --priority Number_between_1_and_5 \
    --source Source_of_the_issue \
    --operational-data Up_to_20_KB_of_data_or_path_to_JSON_file \
    --notifications Arn="SNS_ARN_in_same_Region" \
    --tags "Key=key_name,Value=a_value"
```

指定文件中的操作数据

在创建 OpsItem 时，您可以指定文件中的操作数据。该文件必须是 JSON 文件，并且该文件的内容必须使用以下格式。

```
{
  "key_name": {
    "Type": "SearchableString",
    "Value": "Up to 20 KB of data"
  }
}
```

见下列。

```
aws ssm create-ops-item ^
    --title "EC2 instance disk full" ^
    --description "Log clean up may have failed which caused the disk to be full" ^
    --priority 2 ^
    --source ec2 ^
    --operational-data file:///Users/TestUser1/Desktop/OpsItems/opsData.json ^
    --notifications Arn="arn:aws:sns:us-west-1:12345678:TestUser1" ^
    --tags "Key=EC2,Value=Production"
```

> **Note**
>
> 有关如何在不同本地操作系统的命令行中输入 JSON 格式参数的信息，请参阅 AWS Command Line Interface 用户指南中的在 AWS CLI 中将引号和字符串结合使用。

系统将返回类似于以下内容的信息。

```
{
    "OpsItemId": "oi-1a2b3c4d5e6f"
}
```

3. 运行以下命令以查看有关您创建的 OpsItem 的详细信息。

```
aws ssm get-ops-item --ops-item-id ID
```

系统将返回类似于以下内容的信息。

```
{
    "OpsItem": {
        "CreatedBy": "arn:aws:iam::12345678:user/TestUser",
        "CreatedTime": 1558386334.995,
        "Description": "Log clean up may have failed which caused the disk to be
 full",
        "LastModifiedBy": "arn:aws:iam::12345678:user/TestUser",
        "LastModifiedTime": 1558386334.995,
        "Notifications": [
            {
                "Arn": "arn:aws:sns:us-west-1:12345678:TestUser"
            }
        ],
        "Priority": 2,
        "RelatedOpsItems": [],
        "Status": "Open",
        "OpsItemId": "oi-1a2b3c4d5e6f",
        "Title": "EC2 instance disk full",
        "Source": "ec2",
        "OperationalData": {
            "EC2": {
                "Value": "12345",
                "Type": "SearchableString"
            }
        }
    }
}
```

4. 运行以下命令以更新 OpsItem。此命令将状态从 Open（默认值）更改为 InProgress。

```
aws ssm update-ops-item --ops-item-id ID --status InProgress
```

该命令没有输出。

5. 再次运行以下命令以验证状态是否已更改为 `InProgress`。

```
aws ssm get-ops-item --ops-item-id ID
```

创建 OpsItem 的示例

以下代码示例向您演示如何使用 Linux 管理门户、macOS 或 Windows 创建 OpsItem。

Linux 管理门户或 macOS

以下命令将在 Amazon Elastic Compute Cloud（Amazon EC2）实例磁盘已满时创建 OpsItem。

```
aws ssm create-ops-item \
    --title "EC2 instance disk full" \
    --description "Log clean up may have failed which caused the disk to be full" \
    --priority 2 \
    --source ec2 \
    --operational-data '{"EC2":{"Value":"12345","Type":"SearchableString"}}' \
    --notifications Arn="arn:aws:sns:us-west-1:12345678:TestUser1" \
    --tags "Key=EC2,Value=ProductionServers"
```

以下命令将使用 `OperationalData` 中的 `/aws/resources` 键创建包含 Amazon DynamoDB 相关资源的 OpsItem。

```
aws ssm create-ops-item \
    --title "EC2 instance disk full" \
    --description "Log clean up may have failed which caused the disk to be full" \
    --priority 2 \
    --source ec2 \
    --operational-data '{"/aws/resources":{"Value":"[{\"arn\": \"arn:aws:dynamodb:us-
west-2:12345678:table/OpsItems\"}]","Type":"SearchableString"}}' \
    --notifications Arn="arn:aws:sns:us-west-2:12345678:TestUser"
```

以下命令将使用 `OperationalData` 中的 `/aws/automations` 键创建将 `AWS-ASGEnterStandby` 文档指定为相关联的自动化运行手册的 OpsItem。

```
aws ssm create-ops-item \
    --title "EC2 instance disk full" \
```

```
    --description "Log clean up may have failed which caused the disk to be full" \
    --priority 2 \
    --source ec2 \
    --operational-data '{"/aws/automations":{"Value":"[{\"automationId
\": \"AWS-ASGEnterStandby\", \"automationType\": \"AWS::SSM::Automation
\"}]","Type":"SearchableString"}}' \
    --notifications Arn="arn:aws:sns:us-west-2:12345678:TestUser"
```

Windows

以下命令将在 Amazon Relational Database Service（Amazon RDS）实例不响应时创建 OpsItem。

```
aws ssm create-ops-item ^
    --title "RDS instance not responding" ^
    --description "RDS instance not responding to ping" ^
    --priority 1 ^
    --source RDS ^
    --operational-data={\"RDS\":{\"Value\":\"abcd\",\"Type\":\"SearchableString\"}} ^
    --notifications Arn="arn:aws:sns:us-west-1:12345678:TestUser1" ^
    --tags "Key=RDS,Value=ProductionServers"
```

以下命令将使用 OperationalData 中的 /aws/resources 键创建包含 Amazon EC2 实例相关资源的 OpsItem。

```
aws ssm create-ops-item ^
    --title "EC2 instance disk full" ^
    --description "Log clean up may have failed which caused the disk to be full" ^
    --priority 2 ^
    --source ec2 ^
    --operational-data={\"/aws/resources\":{\"Value\":\"[{\\\"""arn\\""":\
\"""arn:aws:ec2:us-east-1:123456789012:instance/i-1234567890abcdef0\\"""}]\",\"Type\":
\"SearchableString\"}}
```

以下命令将使用 OperationalData 中的 /aws/automations 键创建将 AWS-RestartEC2Instance 运行手册指定为相关联的自动化运行手册的 OpsItem。

```
aws ssm create-ops-item ^
    --title "EC2 instance disk full" ^
    --description "Log clean up may have failed which caused the disk to be full" ^
    --priority 2 ^
    --source ec2 ^
```

```
    --operational-data={\"/aws/automations\":{\"Value\":\"[{\\"""automationId\\""":
\\"""AWS-RestartEC2Instance\\”"",\\"""automationType\\""":\\"""AWS::SSM::Automation\
\"""}]\",\"Type\":\"SearchableString\"}}
```

手动创建 OpsItems（PowerShell）

以下过程介绍了如何使用 AWS Tools for Windows PowerShell 创建 OpsItem。

使用 AWS Tools for Windows PowerShell 创建 OpsItem

1. 打开 AWS Tools for Windows PowerShell，然后运行以下命令，以指定您的凭证。

   ```
   Set-AWSCredentials -AccessKey key-name -SecretKey key-name
   ```

2. 运行以下命令，以便为 PowerShell 会话设置 AWS 区域。

   ```
   Set-DefaultAWSRegion -Region Region
   ```

3. 运行以下命令以创建新的 OpsItem。将每个*#######*替换为您自己的信息。此命令指定用于修正此 OpsItem 的 Systems Manager 自动化运行手册。

   ```
   $opsItem = New-Object Amazon.SimpleSystemsManagement.Model.OpsItemDataValue
   $opsItem.Type = [Amazon.SimpleSystemsManagement.OpsItemDataType]::SearchableString
   $opsItem.Value = '[{\"automationId\":\"runbook_name\",\"automationType\":
   \"AWS::SSM::Automation\"}]'
   $newHash = @{" /aws/
   automations"=[Amazon.SimpleSystemsManagement.Model.OpsItemDataValue]$opsItem}

   New-SSMOpsItem `
       -Title "title" `
       -Description "description" `
       -Priority priority_number `
       -Source AWS_service `
       -OperationalData $newHash
   ```

   如果成功，此命令将输出新 OpsItem 的 ID。

下面的示例指定了受损的 Amazon Elastic Compute Cloud（Amazon EC2）实例的 Amazon 资源名称（ARN）。

```
$opsItem = New-Object Amazon.SimpleSystemsManagement.Model.OpsItemDataValue
```

```
$opsItem.Type = [Amazon.SimpleSystemsManagement.OpsItemDataType]::SearchableString
$opsItem.Value = '[{\"arn\":\"arn:aws:ec2:us-east-1:123456789012:instance/
i-1234567890abcdef0\"}]'
$newHash = @{" /aws/
resources"=[Amazon.SimpleSystemsManagement.Model.OpsItemDataValue]$opsItem}
New-SSMOpsItem -Title "EC2 instance disk full still" -Description "Log clean up may
 have failed which caused the disk to be full" -Priority 2 -Source ec2 -OperationalData
 $newHash
```

管理 OpsItems

OpsCenter 是 AWS Systems Manager 中的一项工具，可追踪 OpsItems 从创建到解决的过程。如果您为跨账户管理设置了 OpsCenter，则委托管理员或管理账户可以从 OpsItems 的账户中管理它们。有关更多信息，请参阅 （可选）将 OpsCenter 手动设置为跨账户集中管理 OpsItems。

您可以使用 Systems Manager 控制台中的以下页面查看和管理 OpsItems：

- 摘要 - 显示打开和正在进行的 OpsItems 的计数、按源和龄期划分的 OpsItems 的计数，以及运营洞察。您可以按源和 OpsItems 状态筛选 OpsItems。

- OpsItems - 显示 OpsItems 的列表，其中包含多个信息字段，如标题、ID、优先级、描述、OpsItem 的源，以及上次更新的日期和时间。使用此页面，您可以手动创建 OpsItems、配置源、更改 OpsItem 的状态，以及按新事件筛选 OpsItems。您可以选择某一 OpsItem，以显示其 OpsItems 详细信息页面。

- OpsItem 详细信息 - 提供可以用于管理 OpsItem 的详细洞察和工具。OpsItems 详细信息页面包含以下选项卡：
  - 概览 - 显示相关资源、过去 30 天内运行的运行手册，以及您可以运行的可用运行手册的列表。您还可以查看相似 OpsItems、添加操作数据，以及添加相关 OpsItems。
  - 相关资源详细信息 - 显示有关来自多个 AWS 服务的资源的信息。展开 Resource details (资源详细信息) 部分，查看由托管此资源的 AWS 服务提供的有关此资源的信息。您也可以使用 Related resources (相关资源) 列表以切换与此 OpsItem 相关的其他相关资源。

有关如何管理 OpsItems 的更多信息，请参阅以下主题。

主题

- 查看 OpsItem 的详细信息

- 编辑 OpsItem
- 将相关资源添加到 OpsItem
- 将相关 OpsItems 添加到 OpsItem
- 将操作数据添加到 OpsItem
- 为 OpsItem 创建事件
- 管理重复的 OpsItems
- 分析运营洞察以减少 OpsItems
- 查看 OpsCenter 日志和报告

查看 OpsItem 的详细信息

要获得 OpsItem 的全面视图，请使用 OpsCenter 控制台中的 OpsItem 详细信息页面。概览页面将显示以下信息：

- OpsItems 详细信息 - 显示所选 OpsItem 的一般信息。
- 相关资源 - 相关资源是指受影响的资源，或已启动创建 OpsItem 的事件的资源。
- 过去 30 天内的自动执行 - 过去 30 天内运行的运行手册列表。
- 运行手册 - 您可以从可用运行手册列表中选择一个运行手册。
- 相似 OpsItem - 这是系统生成的可能与您相关或您可能感兴趣的 OpsItems 的列表。为了生成此列表，系统将扫描所有 OpsItems 的标题和描述并返回使用相似单词的 OpsItems。
- 操作数据 - 操作数据是一种自定义数据，该数据提供有关 OpsItem 的有用参考详细信息。例如，您可以指定日志文件、错误字符串、许可密钥、故障排除提示或其他相关数据。
- 相关 OpsItem - 您可以指定以某种方式与当前 OpsItem 相关的 OpsItems 的 ID。
- 相关资源详细信息 - 显示数据提供程序，包括 Amazon CloudWatch 指标和警报、AWS CloudTrail 日志，以及来自 AWS Config 的详细信息。

查看OpsItem的详细信息

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 OpsCenter。
3. 选择 OpsItem 以查看其详细信息。

编辑 OpsItem

OpsItem 详细信息部分包括有关 OpsItem 的信息，其中包括描述、标题、源、OpsItem ID 和状态。您可以编辑单个 OpsItem，也可以选择多个 OpsItems，然后编辑以下字段：状态、优先级、严重性、类别。

当 Amazon EventBridge 创建 OpsItem 时，它将填充标题、源和描述字段。您可以编辑标题和描述字段，但无法编辑源字段。

> **Note**
>
> 控制台支持 OpsiteM 描述字段中的大多数 Markdown 格式。有关更多信息，请参阅《AWS Management Console 入门指南》中的在控制台中使用 Markdown。

通常，您可以为 OpsItem 编辑以下可配置数据：

- 标题 - OpsItem 的名称。源将创建 OpsItem 的标题。
- 描述 – 有关此 OpsItem 的信息，包括（如果适用）再现问题的步骤。
- 状态：OpsItem 的状态。有关有效状态值的列表，请参阅《AWS Systems ManagerAPI 参考》中的 OpsItem 状态。
- 优先级 - OpsItem 的优先级可以介于 1 到 5 之间。我们建议贵组织确定每个优先级的含义，以及每个优先级的相应服务级别协议。
- 严重性 - OpsItem 的严重性可以介于 1 到 4 之间，其中 1 表示严重，2 表示高，3 表示中等，4 表示低。
- 类别 - OpsItem 的类别可以是可用性、成本、性能、恢复能力或安全性。
- 通知 - 当您编辑 OpsItem 时，可在通知字段中指定 Amazon Simple Notification Service 主题的 Amazon 资源名称（ARN）。通过指定 ARN，您可以确保所有利益相关者在编辑 OpsItem 时收到通知，包括状态更改。有关更多信息，请参阅 Amazon Simple Notification Service 开发人员指南。

> **Important**
>
> Amazon SNS 主题必须位于与 OpsItem 相同的 AWS 区域中。如果主题和 OpsItem 位于不同的区域中，系统将返回错误。

OpsCenter 与 AWS Security Hub 具有双向集成。当您更新与安全结果相关的 OpsItem 状态和严重性时，这些更改将自动发送到 Security Hub，以确保您始终看到最新且正确的信息。

根据 Security Hub 调查发现创建 OpsItem 时，Security Hub 元数据会自动添加到 OpsItem 的操作数据字段中。如果删除此元数据，则双向更新将不再起作用。

编辑 OpsItem 详细信息

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 OpsCenter。
3. 选择 OpsItem ID 以打开详细信息页面，或选择多个 OpsItems。如果您选择多个 OpsItems，则只能编辑状态、优先级、严重性或类别。如果您编辑多个 OpsItems，一旦您选择了新的状态、优先级、严重性或类别，OpsCenter 会立即更新并保存更改。
4. 在 OpsItem details (OpsItem 详细信息) 部分中，选择 Edit (编辑)。
5. 根据组织指定的要求和指南编辑 OpsItem 的详细信息。
6. 完成后，选择 保存。

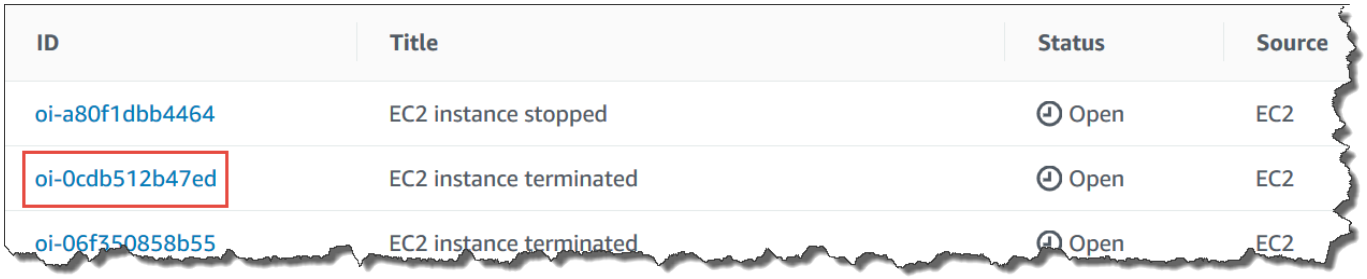
将相关资源添加到 OpsItem

每个 OpsItem 均包括一个相关资源部分，该部分将列出相关资源的 Amazon 资源名称（ARN）。相关资源是指需要调查的受影响的 AWS 资源。

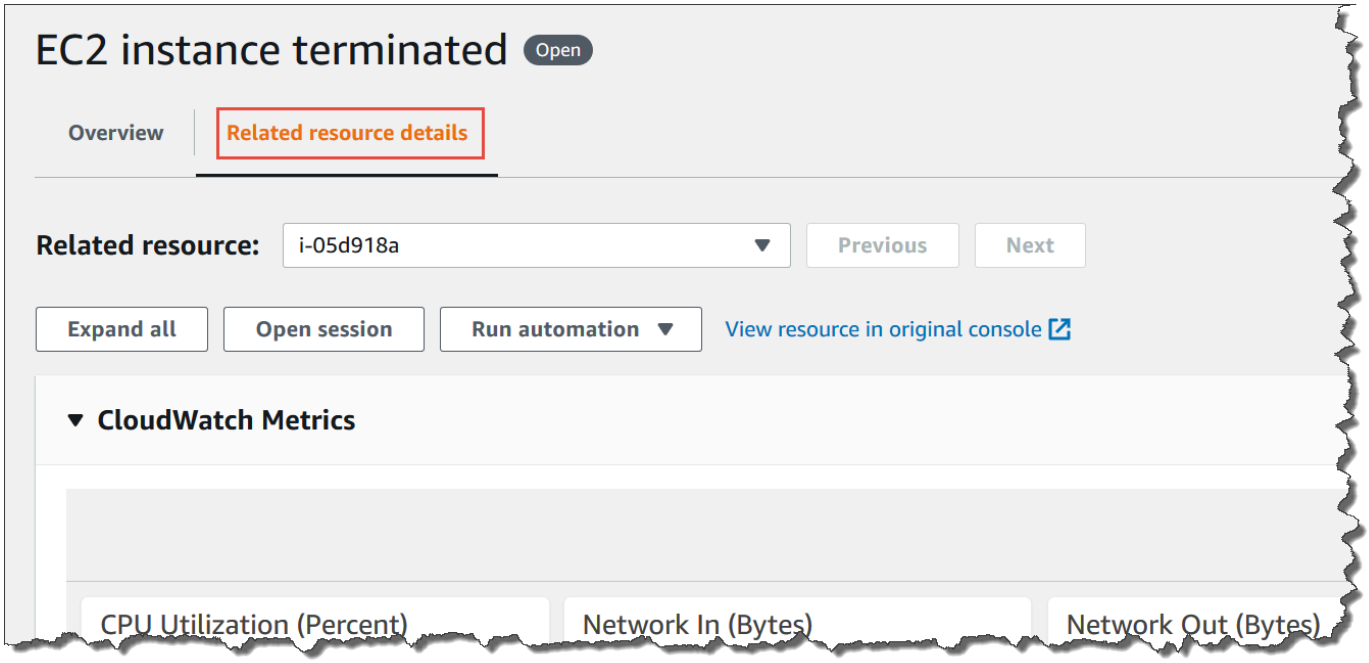
如果 Amazon EventBridge 创建 OpsItem，则系统将自动使用资源的 ARN 填充 OpsItem。您可以手动指定相关资源的 ARN。对于某些 ARN 类型，OpsCenter 将自动创建一个深层链接，它将直接在 OpsCenter 控制台中显示有关该资源的详细信息。例如，如果将 Amazon Elastic Compute Cloud（Amazon EC2）实例的 ARN 指定为相关资源，那么 OpsCenter 会提取与该 EC2 实例相关的详细信息。这使您能够查看有关受影响的 AWS 资源的详细信息，而无需离开 OpsCenter。

查看相关资源并将相关资源添加到 OpsItem

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 OpsCenter。
3. 选择 OpsItems 选项卡。
4. 选择一个 OpsItem ID。



5. 要查看有关受影响的资源的信息，请选择 Related resources details (相关资源详细信息)选项卡。



此选项卡将显示来自多个 AWS 服务的有关此资源的信息。展开 Resource details (资源详细信息) 部分，以查看由托管此资源的 AWS 服务提供的有关此资源的信息。您也可以使用 Related resources (相关资源) 列表以切换与此 OpsItem 相关的其他相关资源。

6. 要添加其他相关资源，请选择 Overview (概述) 选项卡。
7. 在 Related resources (相关资源) 部分中，选择 Add (添加)。
8. 对于 Resource type (资源类型)，从列表中选择一种资源。
9. 对于 Resource ID (资源 ID)，输入资源 ID 或 Amazon Resource Name (ARN)。所选信息类型取决于上一步中选择的资源。

> **Note**
>
> 您可以手动添加额外的相关资源的 ARN。每个 OpsItem 可以列出最多 100 个相关资源 ARN。

下表列出了自动创建指向相关资源的深层链接的资源类型。

支持的资源类型

资源名称	ARN 格式
AWS Certificate Manager 证书	`arn:aws:acm: region:account-id :certificate/ certificate-id`
Amazon EC2 Auto Scaling 组	`arn:aws:autoscaling: region:account-id :autoScalingGroup: groupid:autoScalingGroupName/ groupfriendlyname`
Amazon CloudFront 分配	`arn:aws:cloudfront:: account-id :*`
AWS CloudFormation 堆栈	`arn:aws:cloudformation: region:account-id :stack/stackname /additionalidentifier`
Amazon CloudWatch 告警	`arn:aws:cloudwatch: region:account-id :alarm:alarm-name`
AWS CloudTrail 跟踪	`arn:aws:cloudtrail: region:account-id :trail/trailname`
AWS CodeBuild 项目	`arn:aws:codebuild: region:account-id :resourcetype /resource`
AWS CodePipeline	`arn:aws:codepipeline: region:account-id :resource-specifier`
Amazon DevOps Guru 洞察	`arn:aws:devops-guru: region:account-id :insight/ proactive or reactive/resource-id`

资源名称	ARN 格式
Amazon DynamoDB 表	`arn:aws:dynamodb: region:account-id :table/tablename`
Amazon Elastic Compute Cloud (Amazon EC2) 客户网关	`arn:aws:ec2: region:account-id :customer-gateway/ cgw-id`
Amazon EC2 弹性 IP	`arn:aws:ec2: region:account-id :eip/eipalloc-id`
Amazon EC2 专用主机	`arn:aws:ec2: region:account-id :dedicated-host/ host-id`
Amazon EC2 实例	`arn:aws:ec2: region:account-id :instance/ instance-id`
Amazon EC2 互联网网关	`arn:aws:ec2: region:account-id :internet-gateway/ igw-id`
Amazon EC2 网络访问控制列表	`arn:aws:ec2: region:account-id :network-acl/ nacl-id`
Amazon EC2 网络接口	`arn:aws:ec2: region:account-id :network-interface/ eni-id`
Amazon EC2 路由表	`arn:aws:ec2: region:account-id :route-table/ route-table-id`
Amazon EC2 安全组	`arn:aws:ec2: region:account-id :security-group/ security-group-id`

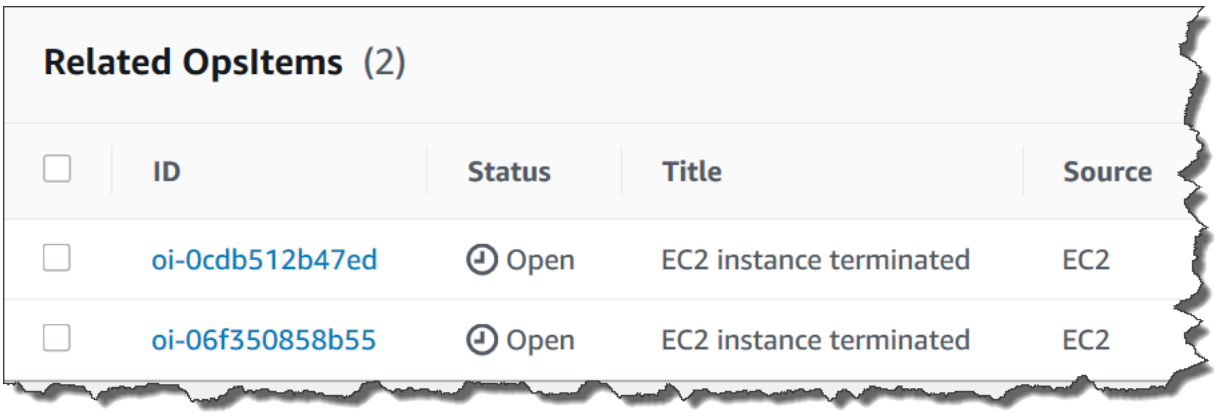
资源名称	ARN 格式
Amazon EC2 子网	`arn:aws:ec2: region:account-id :subnet/subnet-id`
Amazon EC2 卷	`arn:aws:ec2: region:account-id :volume/volume-id`
Amazon EC2 VPC	`arn:aws:ec2: region:account-id :vpc/vpc-id`
Amazon EC2 VPN 连接	`arn:aws:ec2: region:account-id :vpn-connection/ vpn-id`
Amazon EC2 VPN 网关	`arn:aws:ec2: region:account-id :vpn-gateway/ vgw-id`
AWS Elastic Beanstalk 应用程序	`arn:aws:elasticbeanstalk: region:account-id :application/ applicationname`
Elastic Load Balancing (经典负载均衡器)	`arn:aws:elasticloadbalancing: region:account-id :loadbalancer/ name`
Elastic Load Balancing (应用程序负载均衡器)	`arn:aws:elasticloadbalancing: region:account-id :loadbalancer/app/ load-balancer-name /load-balancer-id`
Elastic Load Balancing (Network Load Balancer)	`arn:aws:elasticloadbalancing: region:account-id :loadbalancer/net/ load-balancer-name /load-balancer-id`

资源名称	ARN 格式
AWS Identity and Access Management (IAM) 组	`arn:aws:iam:: account-id :group/group-name`
IAM policy	`arn:aws:iam:: account-id :policy/policy-name`
IAM 角色	`arn:aws:iam:: account-id :role/role-name`
IAM 用户	`arn:aws:iam:: account-id :user/user-name`
AWS Lambda 函数	`arn:aws:lambda: region:account-id :function: function-name`
Amazon Relational Database Service (Amazon RDS) 集群	`arn:aws:rds: region:account-id :cluster: db-cluster-name`
Amazon RDS 数据库实例	`arn:aws:rds: region:account-id :db:db-instance-name`
Amazon RDS 订阅	`arn:aws:rds: region:account-id :es:subscription-name`
Amazon RDS 安全组	`arn:aws:rds: region:account-id :secgrp:security-group-name`
Amazon RDS 集群快照	`arn:aws:rds: region:account-id :cluster-snapshot:  cluster-snapshot-name`

资源名称	ARN 格式
Amazon RDS 子网组	`arn:aws:rds:region:account-id:subgrp:subnet-group-name`
Amazon Redshift 集群	`arn:aws:redshift:region:account-id:cluster:cluster-name`
Amazon Redshift 参数组	`arn:aws:redshift:region:account-id:parametergroup:parameter-group-name`
Amazon Redshift 安全组	`arn:aws:redshift:region:account-id:securitygroup:security-group-name`
Amazon Redshift 集群快照	`arn:aws:redshift:region:account-id:snapshot:cluster-name/snapshot-name`
Amazon Redshift 子网组	`arn:aws:redshift:region:account-id:subnetgroup:subnet-group-name`
Amazon Simple Storage Service (Amazon S3) 桶	`arn:aws:s3:::bucket_name`
AWS Systems Manager 托管式节点清单的 AWS Config 记录	`arn:aws:ssm:region:account-id:managed-instance-inventory/node_id`
Systems Manager State Manager 关联	`arn:aws:ssm:region:account-id:association/association_ID`

将相关 OpsItems 添加到 OpsItem

通过使用 OpsItem 详细信息页面的相关 OpsItems，您可以调查操作问题，并提供问题的上下文。可以通过不同方式与 OpsItems 关联，包括 OpsItems 之间的父子关系、根本原因或重复。您可以将一个 OpsItem 与另一个关联起来，以将其显示在相关 OpsItem 部分中。您最多可以为与当前 OpsItem 相关的其他 OpsItems 指定 10 个 ID。

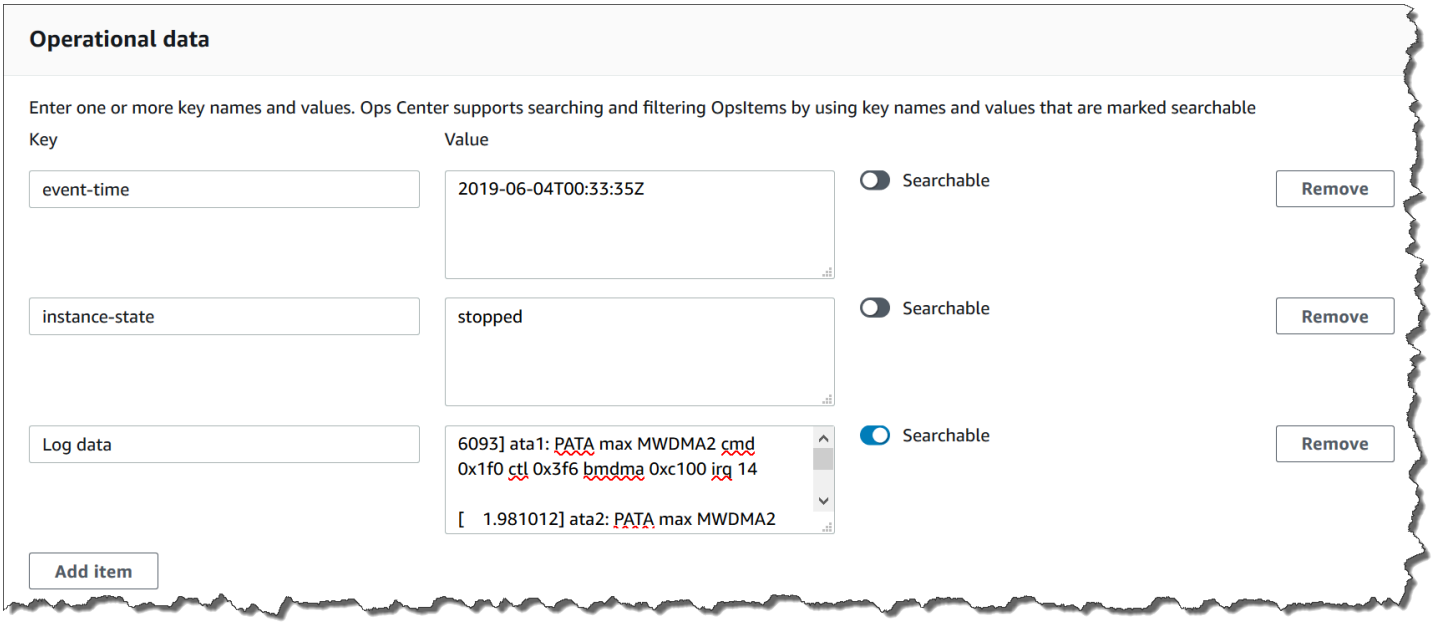


要添加相关的 OpsItem，请执行以下步骤：

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 OpsCenter。
3. 选择 OpsItem ID 以打开详细信息页面。
4. 在 Related OpsItem (相关的 &OI;) 部分中，选择 Add (添加)。
5. 对于 OpsItem ID，请指定一个 ID。
6. 选择添加。

将操作数据添加到 OpsItem

操作数据是一种自定义数据，该数据提供了有关 OpsItem 的有用参考详细信息。您可以输入操作数据的多个密钥/值对。例如，您可以指定日志文件、错误字符串、许可密钥、故障排除提示或其他相关数据。密钥的最大长度可以为 128 个字符，值的最大大小可以为 20 KB。



您可以使数据可供账户中的其他用户搜索，也可以限制搜索访问。可搜索数据是指所有有权访问 OpsItem 概览页面（由 DescribeOpsItems API 操作提供）的用户都可以查看和搜索指定数据。不可搜索的操作数据仅可供有权访问 OpsItem（由 GetOpsItem API 操作提供）的用户查看。

将操作数据添加到 OpsItem

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 OpsCenter。
3. 选择 OpsItem ID 以打开其详细信息页面。
4. 展开操作数据。
5. 如果 OpsItem 没有操作数据，则选择添加。如果 OpsItem 有操作数据，则选择 Manage (管理)。

   在创建操作数据后，您可以通过选择 Manage (管理) 来编辑密钥和值、删除操作数据或添加其他密钥值对。
6. 对于 Key (密钥)，指定一个或多个单词来帮助用户了解数据的用途。

   > **Important**
   >
   > 操作数据键不能以下列单词开头：`amazon`、`aws`、`amzn`、`ssm`、`/amazon`、`/aws`、`/amzn`、`/ssm`。
7. 对于 Value (值)，指定该数据。

8. 选择保存。

> **Note**
>
> 您可以在 OpsItems 页面上使用操作数据运算符筛选 OpsItems。在搜索框中，选择操作数据，然后输入 JSON 格式的键值对。您必须使用以下格式输入键/值对：`{"key":"key_name","value":"a_value"}`

为 OpsItem 创建事件

按照以下过程为 OpsItem 手动创建事件，以便在 AWS Systems Manager Incident Manager（AWS Systems Manager 中的一项工具）中跟踪和管理该 OpsItem。事件是指任何计划外的服务中断或质量下降。有关 Incident Manager 的更多信息，请参阅 the section called “将 OpsCenter 与其他 AWS 服务 集成”。

要为 OpsItem 手动创建事件，请执行以下步骤：

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 OpsCenter。
3. 如果 Incident Manager 为您创建了 OpsItem，请选择它，然后转到步骤 5。如果没有，请选择 Create OpsItem (创建 OpsItem) 并填写表单。如果您没有看到该按钮，请选择 OpsItems 选项卡，然后选择 Create OpsItem (创建 OpsItem)。
4. 如果您创建了 OpsItem，则打开它。
5. 选择 Start Incident (启动事件)。
6. 对于响应计划，选择要分配给此事件的 Incident Manager 响应计划。
7. （可选）对于 Title (标题)，输入一个有助于其他团队成员了解事件性质的描述性名称。如果您没有输入新的标题，OpsCenter 会使用响应计划中的标题在 Incident Manager 中创建 OpsItem 和相应的事件。
8. (可选）对于 Incident impact (事件影响)，选择该事件的影响级别。如果您没有选择影响级别，OpsCenter 会使用响应计划中的影响级别在 Incident Manager 中创建 OpsItem 和相应的事件。
9. 选择启动。

管理重复的 OpsItems

OpsCenter 可从多个 AWS 服务 获得某个源的多个重复 OpsItems。OpsCenter 使用内置逻辑和可配置的重复数据删除字符串的组合来避免创建重复的 OpsItems。在调用创建OpsItem API 操作时，AWS Systems Manager 将应用重复数据删除内置逻辑。

AWS Systems Manager 使用以下重复数据删除逻辑：

1. 创建 OpsItem 时，Systems Manager 会创建和存储基于重复数据删除字符串的哈希值，以及启动 OpsItem 的资源。
2. 当发出另一个创建 OpsItem 的请求时，系统将检查新请求的重复数据删除字符串。
3. 如果此重复数据删除字符串存在匹配的哈希值，则 Systems Manager 将检查现有 OpsItem 的状态。如果现有 OpsItem 的状态为打开或正在进行，则不会创建 OpsItem。如果现有 OpsItem 得到解决，Systems Manager 将创建一个新的 OpsItem。

创建 OpsItem 之后，您无法 编辑或更改该 OpsItem 中的重复数据删除字符串。

要管理重复的 OpsItems，可以执行以下操作：

- 编辑以 OpsCenter 为目标的 Amazon EventBridge 规则的重复数据删除字符串。有关更多信息，请参阅 在默认 EventBridge 规则中编辑重复数据删除字符串。
- 当您手动创建 OpsItem 时，指定重复数据删除字符串。有关更多信息，请参阅 使用 AWS CLI 指定重复数据删除字符串。
- 使用运营洞察查看和解决重复的 OpsItems。您可以使用运行手册来解决重复的 OpsItems。

  为了帮助您解决重复的 OpsItems，并减少由源创建的 OpsItems 的数量，Systems Manager 提供了自动化运行手册。有关信息，请参阅基于洞察解决重复的 OpsItems。

在默认 EventBridge 规则中编辑重复数据删除字符串

按照以下过程为将 OpsCenter 作为目标的 Eventbridge 规则指定重复数据删除字符串。

为 EventBridge 规则编辑重复数据删除字符串

1. 登录 AWS Management Console并打开 Amazon EventBridge 控制台 (https://console.aws.amazon.com/events/)。
2. 在导航窗格中，选择规则。
3. 选择一个规则，然后选择 Edit (编辑)。

4. 转至 Select target(s)（选择目标）页面。
5. 在 Additional settings（其他设置）部分，选择 Configure input transformer（配置输入转换器）。
6. 在 Template（模块）对话框中，找到该 `"operationalData": { "/aws/dedup"` JSON 条目以及您要编辑的重复数据删除字符串。

   Eventbridge 规则中的重复数据删除字符串条目使用以下 JSON 格式。

   ```
   "operationalData": { "/aws/dedup": {"type": "SearchableString","value":
    "{\"dedupString\":\"Words the system should use to check for duplicate
    OpsItems\"}"}}
   ```

   下面是一个例子。

   ```
   "operationalData": { "/aws/dedup": {"type": "SearchableString","value":
    "{\"dedupString\":\"SSMOpsCenter-EBS-volume-performance-issue\"}"}}
   ```
7. 编辑这些重复数据删除字符串，然后选择确认。
8. 选择下一步。
9. 选择下一步。
10. 选择更新规则。

使用 AWS CLI 指定重复数据删除字符串

当您使用 AWS Systems Manager 控制台或 AWS CLI 手动创建新的 OpsItem 时，可以指定重复数据删除字符串。有关在控制台中手动创建 OpsItem 时输入重复数据删除字符串的信息，请参阅 手动创建 OpsItems。如果您使用 AWS CLI，可为 `OperationalData` 参数输入重复数据删除字符串。参数语法使用 JSON，如下面的示例所示。

```
--operational-data '{"/aws/dedup":{"Value":"{\"dedupString\": \"Words the system should
 use to check for duplicate OpsItems\"}","Type":"SearchableString"}}'
```

此处的示例命令指定重复数据删除字符串 `disk full`。

Linux & macOS

```
aws ssm create-ops-item \
    --title "EC2 instance disk full" \
    --description "Log clean up may have failed which caused the disk to be full" \
```

```
    --priority 1 \
    --source ec2 \
    --operational-data '{"/aws/dedup":{"Value":"{\"dedupString\": \"disk full
\"}","Type":"SearchableString"}}' \
    --tags "Key=EC2,Value=ProductionServers" \
    --notifications Arn="arn:aws:sns:us-west-1:12345678:TestUser"
```

Windows

```
aws ssm create-ops-item ^
    --title "EC2 instance disk full" ^
    --description "Log clean up may have failed which caused the disk to be full" ^
    --priority 1 ^
    --source EC2 ^
    --operational-data={\"/aws/dedup\":{\"Value\":\"{\\\"""dedupString\\\""":\\\"""disk
 full\\\"""}\",\"Type\":\"SearchableString\"}} ^
    --tags "Key=EC2,Value=ProductionServers" --notifications Arn="arn:aws:sns:us-
west-1:12345678:TestUser"
```

分析运营洞察以减少 OpsItems

OpsCenter 运营洞察显示有关重复 OpsItems 的信息。OpsCenter 在您的账户中自动分析 OpsItems 并生成三种类型的洞察。您可以在 OpsCenter 摘要选项卡的运营洞察部分中查看此信息。

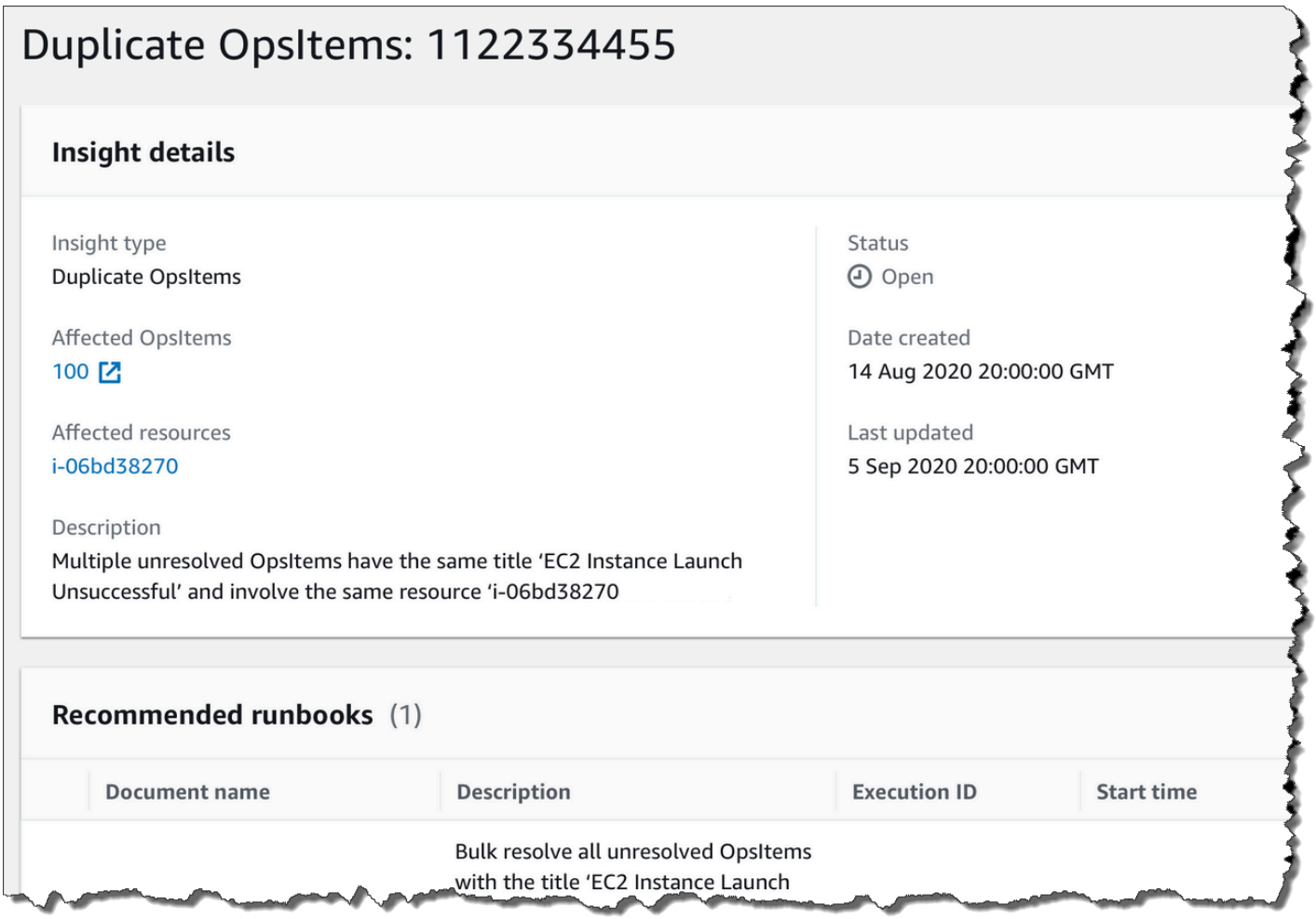
- 重复 OpsItems – 当八个或更多 OpsItems 对同一资源具有相同的标题时，将生成洞察。
- 最常见的标题 – 当超过 50 个 OpsItems 具有相同标题时，将生成洞察。
- 生成最多 OpsItems 的资源 – 当 AWS 资源具有超过 10 个打开的 OpsItems 时，将生成洞察。这些洞察及其对应的资源显示在 OpsCenter 摘要选项卡上的生成最多 OpsItem 的资源表中。资源按 OpsItem 计数的递减顺序列出。

> **Note**
>
> OpsCenter 为以下资源类型创建生成最多 OpsItems 的资源洞察：
>
> - Amazon Elastic Compute Cloud（Amazon EC2）实例
> - Amazon EC2 安全组
> - Amazon EC2 自动扩缩组
> - Amazon Relational Database Service（Amazon RDS）数据库

- Amazon RDS 集群
- AWS Lambda 函数
- Amazon DynamoDB 表
- Elastic Load Balancing 负载均衡器
- Amazon Redshift 集群
- AWS Certificate Manager 证书
- Amazon Elastic Block Store 卷

OpsCenter 强制实施每种类型 15 个洞察的限制。如果某个类型达到此限制，OpsCenter 会停止显示该类型的更多洞察。要查看其他洞察，您必须解决与该类型的 OpsInsight 相关的所有 OpsItems。如果由于 15 个洞察的限制而无法在控制台中显示待处理的洞察，则该洞察将在另一个洞察关闭后变为可见。

当您选择某个洞察后，OpsCenter 会显示受影响的 OpsItems 和资源的相关信息。以下屏幕截图展示了一个示例，其中包含重复 OpsItem 洞察的详细信息。



默认情况下，“运营洞察”处于关闭状态。有关使用运营洞察的更多信息，请参阅以下主题。

主题

- 启用运营洞察
- 基于洞察解决重复的 OpsItems
- 禁用运营洞察

启用运营洞察

您可以在 Systems Manager 控制台的 OpsCenter 页面上启用运营洞察。当您启用运营洞察后，Systems Manager 将创建名为 `AWSServiceRoleForAmazonSSM_OpsInsights` 的 AWS Identity and Access Management（IAM）服务相关角色。服务相关角色是一种与 Systems Manager 直接关联的独特类型的 IAM 角色。服务相关角色是预定义的角色，包括相应服务代表您调用其他 AWS 服务 所需的所有权限。有关 `AWSServiceRoleForAmazonSSM_OpsInsights` 服务相关角色的更多信息，请参阅在 Systems Manager OpsCenter 中使用角色创建运营洞察 OpsItem。

> **Note**
>
> 请注意以下重要信息：
>
> - 将向您的 AWS 账户 收取运营洞察的费用。有关更多信息，请参阅AWS Systems Manager 定价。
> - OpsCenter 使用批处理定期刷新洞察。这意味着 OpsCenter 中显示的洞察列表可能不同步。

按照以下过程在 OpsCenter 中启用和查看运营洞察。

启用和查看运营洞察

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 OpsCenter。
3. 在运营洞察可用消息框中，选择启用。如果您没有看到此消息，请向下滚动到运营洞察部分，然后选择启用。
4. 启用此功能后，在摘要选项卡上，向下滚动到运营洞察部分。
5. 要查看经过筛选的洞察列表，选择重复OpsItems、最常见的标题或生成最多 OpsItems 的资源旁边的链接。要查看所有洞察，选择 View all operational insights (查看所有运营洞察)。
6. 选择洞察 ID 以查看更多信息。

基于洞察解决重复的 OpsItems

要解决洞察问题，必须先解决与洞察关联的所有 OpsItems 的问题。您可以使用 `AWS-BulkResolveOpsItemsForInsight` 运行手册来解决与洞察关联的 OpsItems 的问题。

为了帮助您解决重复的 OpsItems，并减少由源创建的 OpsItems 的数量，Systems Manager 提供了以下自动化运行手册：

- `AWS-BulkResolveOpsItems` 运行手册解决与指定筛选条件匹配的 OpsItems。
- `AWS-AddOpsItemDedupStringToEventBridgeRule` 运行手册将为与特定 Amazon EventBridge 规则关联的所有 OpsItem 目标添加重复数据删除字符串。如果某一规则已经包含重复数据删除字符串，则此运行手册不会再添加该字符串。
- 如果 EventBridge 中的某一规则生成数十或数百个 OpsItems，则 `AWS-DisableEventBridgeRule` 将关闭该规则。

解决运营洞察问题

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 OpsCenter。
3. 在 Overview (概览) 选项卡上，向下滚动到 Operational insights (运营洞察)。
4. 选择查看所有运营洞察。
5. 选择洞察 ID 以查看更多信息。
6. 选择运行手册，然后选择执行。

禁用运营洞察

当您关闭运营洞察后，系统将停止创建新的洞察，并停止在控制台中显示洞察。任何处于活动状态的洞察在系统中都保持不变，但您不会在控制台中看到它们。如果您再次启用此功能，系统将显示之前未解决的洞察，并开始创建新的洞察。按照以下过程关闭运营洞察。

关闭运营洞察

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 OpsCenter。
3. 选择设置。
4. 在 Operational insights (运营洞察) 部分，选择 Edit (编辑)，然后切换 Disable (禁用) 选项。
5. 选择保存。

查看 OpsCenter 日志和报告

AWS CloudTrail 可以记录对控制台、AWS Command Line Interface（AWS CLI）和 SDK 的 AWS Systems Manager OpsCenter API 调用。您可以在 CloudTrail 控制台或 Amazon Simple Storage Service (Amazon S3) 存储桶中查看信息。Amazon S3 使用一个存储桶来存储您账户的所有 CloudTrail 日志。

OpsCenter 操作日志显示对 OpsItem 活动进行的创建、更新、获取和描述操作。有关查看和使用 Systems Manager 活动的 CloudTrail 日志的更多信息，请参阅 使用 AWS CloudTrail 记录 AWS Systems Manager API 调用。

AWS Systems Manager OpsCenter 可以为您提供有关 OpsItems 的以下信息：

- OpsItem 状态摘要 - 按状态（“打开且正在进行”、“打开”或“正在进行”、）提供 OpsItems 的摘要。
- 具有最多开放 OpsItems 的源 - 具有最多开放 OpsItems 的热门 AWS 服务 分类。
- 按源和使用期限显示的 OpsItems - 提供按源和创建天数分组的 OpsItems 的计数。

查看 OpsCenter 摘要报告

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 OpsCenter。
3. 在 OpsItems 概览页面上，选择摘要。
4. 在 OpsItems by source and age (按源和使用期限显示的 OpsItem) 下，选择搜索栏以根据 Source (源) 筛选 OpsItems。使用列表来根据状态筛选。

删除 OpsItems

您可以使用 AWS Command Line Interface 或 AWS SDK 来调用 DeleteOpsItem API 操作，从而删除单个 OpsItem。您无法 AWS Management Console在中删除 OpsItem。要删除 OpsItem，您的 AWS Identity and Access Management（IAM）用户、组或角色必须具有管理员权限，或者您必须已获得调用 `DeleteOpsItem` API 操作的权限。

> **Important**
>
> 对于此操作应注意以下重要的信息：
>
> - 删除 OpsItem 后将不可逆。您无法恢复已删除的 OpsItem。
> - 此操作使用的是最终一致性模型，这意味着系统可能需要几分钟才能完成操作。如果您删除 OpsItem 后立即调用（例如 GetOpsItem），则已删除的 OpsItem 可能仍会出现在响应中。
> - 此操作是幂等的。如果您为同一 OpsItem 反复调用此操作，系统不会发出异常。如果第一个调用成功，则所有其他调用都会返回与第一个调用相同的成功响应。
> - 此操作不支持跨账户调用。委派管理员或管理账户无法删除其他账户中的 OpsItems，即使针对跨账户管理设置了 OpsCenter 也不可行。有关跨账户管理的更多信息，请参阅（可选）将 OpsCenter 设置为跨账户集中管理 OpsItems。
> - 如果您收到 `OpsItemLimitExceededException`，则可以删除一个或多个 OpsItems，以使 OpsItems 总数低于限额。有关此异常的更多信息，请参阅 排查 OpsCenter 问题。

删除 OpsItem

按照以下过程删除 OpsItem。

删除 OpsItem

1. 安装并配置 AWS CLI（如果尚未执行该操作）。有关更多信息，请参阅安装或更新 AWS CLI 的最新版本。
2. 运行以下命令。将 *ID* 替换为您要删除的 OpsItem ID。

```
aws ssm delete-ops-item --ops-item-id ID
```

如果成功，该命令将不会返回任何数据。

修复 OpsItem 问题

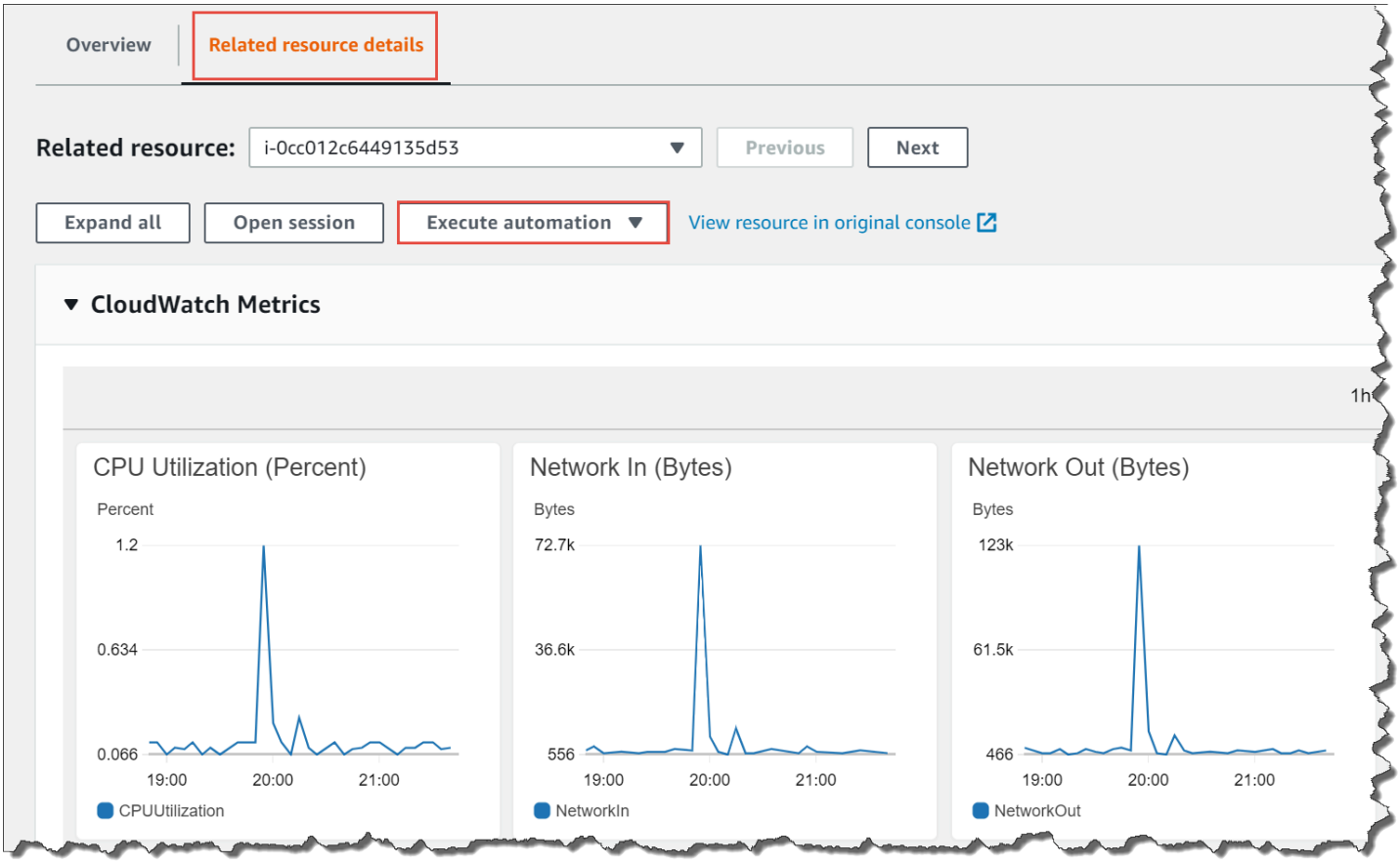
使用 AWS Systems Manager 自动化运行手册，您可以修复在 OpsItem 中发现的 AWS 资源的问题。自动化使用预定义的运行手册来修复 AWS 资源的常见问题。

每个 OpsItem 都包括运行手册部分，该部分将提供可用于修复的运行手册列表。当您从该列表中选择自动化运行手册时，OpsCenter 将自动显示运行该文档所需的一些字段。当您运行自动化运行手册时，系统会将该运行手册与 OpsItem 的相关资源关联起来。如果 Amazon EventBridge 创建了 OpsItem，它会将运行手册与 OpsItem 关联起来。OpsCenter 将为 OpsItem 保留自动化运行手册的 30 天记录。

您可以选择某一状态以查看有关运行手册的重要详细信息，如自动化失败的原因，以及失败发生时正在运行自动化运行手册的步骤，如下面的示例所示。



所选 OpsItem 的 Related resource details (相关资源详细信息) 页面包含 Run automation (运行自动化) 列表。您可以选择最近或特定于资源的自动化运行手册，并且可以运行它们以修复问题。此页面还包括多个数据提供程序，其中包括 Amazon CloudWatch 指标和警报、AWS CloudTrail 日志，以及来自 AWS Config 的详细信息。



您可以通过在控制台中选择自动化运行手册名称或使用 Systems Manager 自动化运行手册参考 来查看有关该运行手册的信息。

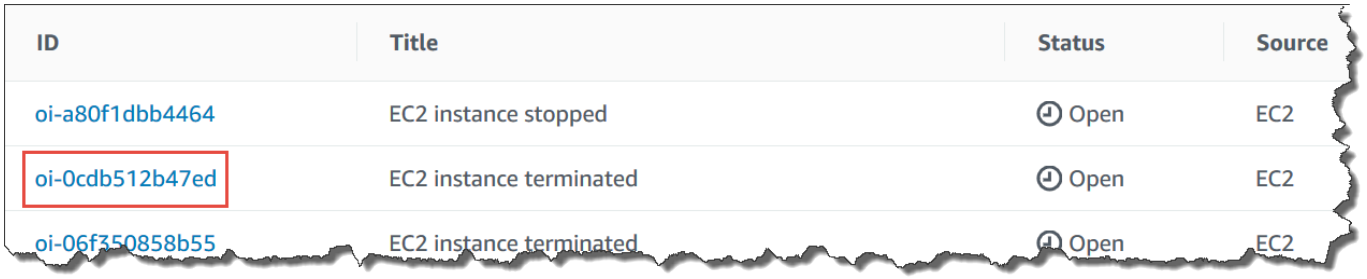
使用运行手册修复 OpsItem

在使用自动化运行手册修复 OpsItem 问题之前，请执行以下操作：

- 确认您有权运行 Systems Manager 自动化运行手册。有关更多信息，请参阅 设置自动化。
- 为要运行的自动化运行手册收集特定于资源的 ID 信息。例如，如果您要运行重新启动 EC2 实例的自动化，则必须指定要重新启动的 EC2 实例的 ID。

要运行自动化运行手册来修正 OpsItem 问题，请执行以下步骤：

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 OpsCenter。
3. 选择 OpsItem ID 以打开详细信息页面。



4. 滚动到 Runbooks (运行手册) 部分。
5. 使用搜索栏或右上角的数字查找您要运行的自动化运行手册。
6. 选择运行手册，然后选择 Execute (执行)。
7. 输入运行手册的所需信息，然后选择提交。

   启动运行手册后，系统会返回到上一个屏幕并显示状态。
8. 在过去 30 天内的自动化执行部分中，选择执行 ID 链接以查看执行步骤和状态。

使用关联的运行手册修复 OpsItem

在您从 OpsItem 中运行自动化运行手册后，OpsCenter 会将该运行手册与 OpsItem 关联起来。关联的运行手册在运行手册列表中的排名高于其他运行手册。

按照以下过程运行已与 OpsItem 中的相关资源关联的自动化运行手册。有关添加相关资源的信息，请参阅 管理 OpsItems。

要运行与资源关联的运行手册以修正 OpsItem 问题，请执行以下步骤：

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 OpsCenter。
3. 打开 OpsItem。
4. 在 Related resources (相关资源) 部分中，选择要运行自动化运行手册的资源。
5. 选择 Run automation (运行自动化)，然后选择要运行的关联自动化运行手册。
6. 输入运行手册的所需信息，然后选择 Execute (执行)。

   启动运行手册后，系统会返回到上一个屏幕并显示状态。
7. 在过去 30 天内的自动化执行部分中，选择执行 ID 链接以查看执行步骤和状态。

查看 OpsCenter 摘要报告

AWS Systems Manager OpsCenter 包括一个摘要页面，其中将自动显示以下信息：

- OpsItem 状态摘要 – 按状态列出的 OpsItems 摘要，例如 `Open` 和 `In progress`。
- 具有最多打开的 OpsItems 的源 – 具有最多打开的 OpsItems 的 AWS 服务 明细。
- 按源和使用期限显示的 OpsItems – 按源和创建天数分组的 OpsItems 的计数。

查看 OpsCenter 摘要报告

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 OpsCenter，然后选择摘要选项卡。
3. 在按源和使用期限显示的 OpsItems 部分中，执行以下操作：
   1. （可选）在筛选字段中，选择来源，选择 `Equal`、`Begin With` 或 `Not Equal`，然后输入搜索参数。
   2. 在相邻列表中，选择下列状态值之一：
      - `Open`
      - `In progress`
      - `Resolved`
      - `Open and in progress`
      - `All`

排查 OpsCenter 问题

本主题包含的信息有助于排查 OpsCenter 的常见错误和问题。

您收到了 OpsItemLimitExceededException

如果您的 AWS 账户 在调用 CreateOpsItem API 操作时已达到允许的最大 OpsItems 数量，您将会收到 `OpsItemLimitExceededException`。如果您的调用将会超过以下任一限额的最大 OpsItems 数量时，OpsCenter 将会返回异常：

- 每区域每 AWS 账户的 OpsItems 总数（包括 `Open` 和 `Resolved` OpsItem）：500,000
- 每月每 AWS 账户的最大 OpsItems 数：10000

这些限额适用于从任何来源创建的 OpsItems，但以下来源除外：

- 由 AWS Security Hub 调查发现创建的 OpsItems
- 在 Incident Manager 事件打开时自动生成的 OpsItems

从这些来源创建的 OpsItems 不会计入您的 OpsItem 限额，但您需要为每个 OpsItem 付费。

如果您收到了 `OpsItemLimitExceededException`，则可以手动删除 OpsItems，直到低于您的限额，不会妨碍您创建新的 OpsItem 为止。同样，删除为 Security Hub 调查发现或 Incident Manager 事件创建的 OpsItems 不会减少限额强制执行的 OpsItems 总数。您必须删除其他来源的 OpsItems。有关如何删除 OpsItem 的信息，请参阅 删除 OpsItems。

对于大量自动生成的 OpsItems，您会收到高额的 AWS 账单

如果您配置了与 AWS Security Hub 的集成，则 OpsCenter 会为 Security Hub 调查发现创建 OpsItems。根据 Security Hub 生成的调查发现数量以及您在配置集成时登录的账户，OpsCenter 可能会生成大量的 OpsItems 并且这会产生成本。以下是与 Security Hub 调查发现生成的 OpsItems 相关的更具体详细信息：

- 如果您在配置 OpsCenter 和 Security Hub 集成时登录到 Security Hub 管理员账户，则系统会针对在管理员和所有成员账户中的调查发现创建 OpsItems。OpsItems 均在管理员账户中创建。根据各种因素，这可能会导致来自 AWS 的意外巨额账单。

  如果您在配置集成时登录到成员账户，则系统仅针对该个人账户中的调查发现创建 OpsItems。有关 Security Hub 管理员账户、成员账户及其与 EventBridge 调查发现事件源关系的更多信息，请参阅《AWS Security Hub User Guide》中的 Types of Security Hub integration with EventBridge。
- 对于创建 OpsItem 的每个调查发现，都将按常规价格向您收取创建 OpsItem 的费用。如果您编辑了 OpsItem 或在 Security Hub 中更新了相应的调查发现（这会触发 OpsItem 更新），则也会产生费用。
- 通过与 AWS Security Hub 集成创建的 OpsItems 暂不受每个区域每个账户 50 万个 OpsItems 的最大配额限制。因此，Security Hub 警报可以在账户的每个区域中创建超过 50 万个可收费 OpsItems。

  因此，对于高负载生产环境，建议将 Security Hub 调查发现范围限定为仅高严重性问题。

> **Important**
>
> 如果您认为有大量 OpsItems 是错误创建的，并且您的 AWS 账单没有依据，请联系 支持。

如果您不再希望系统为 Security Hub 调查发现创建 OpsItems，请使用以下步骤。

停止接收 Security Hub 调查发现的 OpsItems

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 OpsCenter。
3. 选择设置。
4. 在 Security Hub 调查发现部分，选择编辑。
5. 选择滑块将已启用更改为已禁用。如果无法切换滑块，则说明您的 AWS 账户 尚未启用 Security Hub。
6. 选择保存以保存您的配置。OpsCenter 不再根据 Security Hub 的调查发现创建 OpsItems。

> **Important**
>
> 如果 OpsCenter 将设置切换回已启用并继续为调查发现创建 OpsItems，请登录 Systems Manager 委派管理员账户或 AWS Organizations 管理账户，然后重复此过程。如果您无权登录其中任何一个账户，请联系您的管理员并要求他们重复此步骤，以为您的账户禁用集成。

使用 Systems Manager 托管的 Amazon CloudWatch 控制面板

Amazon CloudWatch 控制面板是 CloudWatch 控制台中的可自定义主页，可用于在单一视图中监控资源，即便是分布到不同 AWS 区域的资源，也能对其进行监控。您可以使用 CloudWatch 控制面板为您的 AWS 资源创建指标和告警的自定义视图。利用控制面板，您可以创建以下各项：

- 所选指标和告警的单一视图，用于帮助您跨一个或多个 AWS 区域评估资源和应用程序的运行状况。您可以在每个图表上选择用于每个指标的颜色，以便您能跨多个图表跟踪同一指标。
- 一个操作手册，为团队成员提供有关如何对操作事件期间发生的特定事故做出响应的指南。
- 关键资源与应用程序测量的公共视图，团队成员可以共享该视图，以便在操作事件期间加快通信流。

您可以使用控制台、 AWS Command Line Interface (AWS CLI) 或使用 CloudWatch `PutDashboard` API 来创建控制面板。有关更多信息，请参阅《Amazon CloudWatch 用户指南》中的使用 Amazon CloudWatch 控制面板。

使用 SSM Agent

AWS Systems Manager Agent（SSM Agent）是一款 Amazon 软件，可在 Amazon Elastic Compute Cloud（Amazon EC2）实例、边缘设备、本地服务器和虚拟机（VM）上运行。SSM Agent 使 Systems Manager 可以更新、管理和配置这些资源。代理在 AWS Cloud 中处理来自 Systems Manager 服务的请求，然后按照请求中指定的方式运行它们。SSM Agent 之后使用 Amazon Message Gateway Service（ssmmessages）将状态和执行信息发送回 Systems Manager 服务。（在 2024 年之前推出的 AWS 区域 中，状态和执行信息也可能通过 Amazon Message Delivery Service（服务前缀:ec2messages）发回。）

如果您监控流量，则会发现托管式节点与 ssmmessages.* 端点以及可能与 ec2messages.* 端点进行通信。有关更多信息，请参阅 参考：ec2messages、ssmmessages 和其他 API 操作。有关将 SSM Agent 日志移植到 Amazon CloudWatch Logs 的更多信息，请参阅 AWS Systems Manager 中的日志记录和监控。

内容

- 了解有关 SSM Agent 的技术细节
- 查找预装了 SSM Agent 的 AMIs
- 在适用于 Linux 的 EC2 实例上使用 SSM Agent
- 在适用于 macOS 的 EC2 实例上使用 SSM Agent
- 在适用于 Windows Server 的 EC2 实例上使用 SSM Agent
- 正在检查 SSM Agent 状态并启动代理
- 正在检查 SSM Agent 版本号
- 查看 SSM Agent 日志
- 通过 SSM Agent 限制对根级别命令的访问
- 自动更新到 SSM Agent
- 订阅 SSM Agent 通知
- 排查 SSM Agent 问题

了解有关 SSM Agent 的技术细节

使用本主题中的信息有助于您实施 AWS Systems Manager Agent (SSM Agent) 并了解该代理的工作原理。

主题

- SSM Agent 版本 3.2.x.x 凭证行为
- SSM Agent 凭证优先级
- 配置 SSM Agent 以与美国联邦信息处理标准（FIPS）结合使用
- 关于本地 ssm-user 账户
- SSM Agent 和 Instance Metadata Service (IMDS)
- 保持 SSM Agent 的最新状态
- 确保 SSM Agent 安装目录未被修改、移动或删除
- 按 AWS 区域 的 SSM Agent 滚动更新
- SSM Agent 与 AWS 托管 S3 存储桶进行通信

SSM Agent 版本 3.2.x.x 凭证行为

使用 Quick Setup 中的默认主机管理配置加载实例时，SSM Agent 将一组临时凭证存储在 `/var/lib/amazon/ssm/credentials`（适用于 Linux 和 macOS）或 `%PROGRAMFILES%\Amazon\SSM\credentials`（适用于 Windows Server）。临时凭证具有您为默认主机管理配置选择的 IAM 角色指定的权限。在 Linux 上，只有 root 账户才能访问这些凭证。在 Windows Server 上，只有 SYSTEM 账户和本地管理员才能访问这些凭证。

SSM Agent 凭证优先级

本主题介绍有关如何向 SSM Agent 授予对资源执行操作的权限的重要信息。

> **Note**
>
> 对边缘设备的支持略有不同。必须将边缘设备配置为使用 AWS IoT Greengrass Core 软件，配置一个 AWS Identity and Access Management（IAM）服务角色，并使用 AWS IoT Greengrass 将 SSM Agent 部署到设备。有关更多信息，请参阅 使用 Systems Manager 管理边缘设备。

在计算机上安装 SSM Agent 后，需要适当的权限才能与 Systems Manager 服务进行通信。在 Amazon Elastic Compute Cloud (Amazon EC2) 实例上，将在附加到实例的实例配置文件中提供这些权限。在非 EC2 计算机上，SSM Agent 通常从共享的凭证文件中获取所需的权限，

该文件位于 `/root/.aws/credentials`（Linux 和 macOS）或 `%USERPROFILE%\.aws\credentials`（Windows Server）。所需的权限将在混合激活过程中添加至此文件。

但是，在极少数情况下，计算机的权限最终可能会添加至多个位置，SSM Agent 将检查这些位置是否有权运行其任务。

例如，您可能将 EC2 实例配置为由 Systems Manager 托管。该配置包括附加实例配置文件。但是，之后您决定还将该实例用于开发人员或终端用户任务，并在上面安装 AWS Command Line Interface（AWS CLI）。此类安装会导致将其他权限添加至实例上的凭证文件。

在实例上运行 Systems Manager 命令时，SSM Agent 可能会尝试使用与您期望使用的凭证不同的凭证，例如凭证文件而不是实例配置文件中的凭证。这是因为 SSM Agent 将按照默认凭证提供程序链的规定顺序查找凭证。

> **Note**
>
> 在 Linux 和 macOS 上，SSM Agent 以根用户身份运行。因此，SSM Agent 在此过程中查找的环境变量和凭证文件仅属于根用户（`/root/.aws/credentials`）。SSM Agent 在搜索凭证期间，不会查看实例上任何其他用户的环境变量或凭证文件。

默认提供程序链将按照以下顺序查找凭证：

1. 环境变量（`AWS_ACCESS_KEY_ID` 和 `AWS_SECRET_ACCESS_KEY`）（如果已配置）。
2. 共享的凭证文件 (对于 Linux 和 macOS，为 `$HOME/.aws/credentials`；对于 Windows Server，为 `%USERPROFILE%\.aws\credentials`）以及混合激活或 AWS CLI 安装等提供的权限。
3. 任务的 AWS Identity and Access Management (IAM) 角色，如果存在一个应用程序使用 Amazon Elastic Container Service (Amazon ECS) 任务定义或 RunTask API 操作。
4. 附加到 Amazon EC2 实例的实例配置文件
5. 为默认主机管理配置选择的 IAM 角色。

有关信息，请参阅以下主题：

- EC2 实例的实例配置文件 — 配置 Systems Manager 所需的实例权限
- 混合激活 — 创建混合激活以将节点注册到 Systems Manager
- AWS CLI 凭证 - AWS Command Line Interface 用户指南中的配置和凭证文件设置

- 默认凭证提供程序链 -适用于 Go 的 AWS SDK Developer Guide 中的指定凭证

> **Note**
>
> 适用于 Go 的 AWS SDK Developer Guide 中的本主题介绍了 SDK for Go 的默认提供程序链；但是，评估 SSM Agent 的凭证时也遵循同一原则。

配置 SSM Agent 以与美国联邦信息处理标准（FIPS）结合使用

如果您需要将 Systems Manager 与经美国联邦信息处理标准（FIPS）140-3 验证的加密模块结合使用，则可以将 AWS Systems Manager Agent（SSM Agent）配置为在支持的区域中使用 FIPS 端点。

若要配置 SSM Agent 以连接至 FIPS 140-3 端点

1. 连接到您的托管节点
2. 导航到包含 amazon-ssm-agent.json 文件的目录：
   - Linux：/etc/amazon/ssm/
   - macOS: /opt/aws/ssm/
   - Windows Server: C:\Program Files\Amazon\SSM\
3. 打开名为 amazon-ssm-agent.json 的文件进行编辑。

   > **Tip**
   >
   > 如果尚不存在任何 amazon-ssm-agent.json 文件，请将 amazon-ssm-agent.json.template 的内容复制到名为 amazon-ssm-agent.json 的新文件中。Saveamazon-ssm-agent.json在同一目录中，其中amazon-ssm-agent.json.template位于中。

4. 将以下内容添加到该文件。将 *region* 占位符值替换为分区的相应区域代码：

```
{
    ---Existing file content, if any---
    "Mds": {
        "Endpoint": "ec2messages-fips.region.amazonaws.com",
    },
    "Ssm": {
        "Endpoint": "ssm-fips.region.amazonaws.com",
```

```
    },
    "Mgs": {
        "Endpoint": "ssmmessages-fips.region.amazonaws.com",
        "Region": "region"
    },
    "S3": {
        "Endpoint": "s3-fips.dualstack.region.amazonaws.com",
        "Region": region"
    },
    "Kms": {
        "Endpoint": "kms-fips.region.amazonaws.com"
    }
}
```

支持的区域包括以下：

- 美国东部（弗吉尼亚州北部）区域为 us-east-1
- 美国东部（俄亥俄州）区域为 us-east-2
- 美国西部（北加利福尼亚）区域为 us-west-1
- 美国西部（俄勒冈州）区域为 us-west-2
- 加拿大西部（卡尔加里）区域为 ca-west-1

5. 保存文件并重新启动 SSM Agent。

每次更改配置时，请重新启动 SSM Agent。

您可以按照相同的过程自定义 SSM Agent 的其他功能。有关可用配置属性及其默认值的最新列表，请参阅 GitHub 的 amazon-ssm-agent 存储库中的配置属性定义。

有关 FIPS 的 AWS 支持的更多信息，请参阅美国联邦信息处理标准（FIPS）140-3。

关于本地 ssm-user 账户

从 SSM Agent 2.3.50.0 版开始，该代理会创建一个名为 ssm-user 的本地用户账户，并将其添加至 /etc/sudoers.d 目录（Linux 和 macOS）或管理员组 (Windows Server)。在 2.3.612.0 之前的代理版本上，账户会在 SSM Agent 安装后首次启动或重启时创建。在版本 2.3.612.0 及更高版本上，ssm-user 账户将在会话在实例上首次启动时创建。当会话在Session Manager（AWS Systems Manager 中的一项工具）中启动会话时，该 ssm-user 是默认操作系统用户。您可以通过将 ssm-user 移动到权限较低的组或更改 sudoers 文件来更改权限。在卸载 SSM Agent 后，不会从系统中移除 ssm-user 账户。

在 Windows Server 上，SSM Agent 处理每个会话启动时对 `ssm-user` 账户的新密码的设置。Linux 托管实例上没有为 `ssm-user` 设置密码。

从 SSM Agent 2.3.612.0 版开始，不会在用作域控制器的 Windows Server 计算机上自动创建 `ssm-user` 账户。要在 Windows Server 域控制器上使用 Session Manager，手动创建 `ssm-user` 账户（如果尚不存在），然后向用户分配域管理员权限。

> **Important**
>
> 为了能够创建 `ssm-user` 账户，附加到实例的实例配置文件必须提供必要的权限。有关信息，请参阅步骤 2：验证或添加 Session Manager 的实例权限。

SSM Agent 和 Instance Metadata Service (IMDS)

Systems Manager 需要依靠 EC2 实例元数据才能正常运行。Systems Manager 可以使用 Instance Metadata Service 1 或 2 版（IMDSv1和IMDSv2）访问实例元数据。您的实例必须能够访问实例元数据服务的 IPv4 地址：169.254.169.254。有关更多信息，请参阅《Amazon EC2 用户指南》中的实例元数据和用户数据。

保持 SSM Agent 的最新状态

如果有新工具添加至 Systems Manager 或者对现有工具进行了更新，则将发布 SSM Agent 的更新版本。无法使用代理的最新版本可能会阻止托管式节点使用 Systems Manager 的各项工具和功能。因此，我们建议您自动完成确保机器上的 SSM Agent 为最新的过程。有关更多信息，请参阅 自动更新到 SSM Agent。要获得有关 SSM Agent 更新的通知，请在 GitHub 上订阅 SSM Agent 发布说明页面。

> **Note**
>
> 如果有新工具添加至 Systems Manager 或者对现有工具进行了更新，则将发布 SSM Agent 的更新版本。无法使用代理的最新版本可能会阻止托管式节点使用 Systems Manager 的各项工具和功能。因此，我们建议您自动完成确保机器上的 SSM Agent 为最新的过程。有关更多信息，请参阅 自动更新到 SSM Agent。要获得有关 SSM Agent 更新的通知，请在 GitHub 上订阅 SSM Agent 发布说明页面。
>
> 默认包含 SSM Agent 的 Amazon Machine Images (AMIs) 可能需要长达两周时间更新为最新版本的 SSM Agent。我们建议您将 SSM Agent 的自动更新配置为更高的频率。

确保 SSM Agent 安装目录未被修改、移动或删除

SSM Agent 安装在 `/var/lib/amazon/ssm/`（Linux 和 macOS）和 `%PROGRAMFILES%\Amazon\SSM\`（Windows Server）上。这些安装目录包含 SSM Agent 使用的关键文件和文件夹，例如凭证文件、进程间通信（IPC）资源和编排文件夹。不得修改、移动或删除安装目录中的任何内容。否则，SSM Agent 可能会停止正常运行。

按 AWS 区域 的 SSM Agent 滚动更新

SSM Agent 更新在 GitHub 存储库中可用后，可能需要长达两周时间，更新的版本才会在不同的时间推广到所有 AWS 区域。为此，当您尝试在区域中部署 SSM Agent 的新版本时，可能会收到以下错误：“在当前平台上不受支持”或“正在将 amazon-ssm-agent 更新为早期版本，请启用‘允许降级’以继续”。

要确定您的可用 SSM Agent 版本，可以运行 `curl` 命令。

要查看全局下载存储桶中的可用代理版本，请运行以下命令。

```
curl https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/VERSION
```

要查看特定区域中的可用代理版本，请运行以下命令，将 *region* 替换为与您正在使用的区域，例如对于美国东部（俄亥俄州）区域使用 `us-east-2`。

```
curl https://s3.region.amazonaws.com/amazon-ssm-region/latest/VERSION
```

还可以直接在浏览器中打开 `VERSION` 文件，无需使用 `curl` 命令。

SSM Agent 与 AWS 托管 S3 存储桶进行通信

在执行各种 Systems Manager 操作过程中，AWS Systems Manager Agent (SSM Agent) 将访问许多 Amazon Simple Storage Service (Amazon S3) 存储桶。这些 S3 存储桶可以公开访问，默认情况下，SSM Agent 使用 HTTP 调用连接到这些存储桶。

但是，如果您正在 Systems Manager 操作中使用虚拟私有云（VPC）端点，则必须在 Systems Manager 的 Amazon Elastic Compute Cloud（Amazon EC2）实例配置文件中提供显式权限，或者在混合和多云环境中的非 EC2 计算机的服务角色中提供。否则，资源无法访问这些公有存储桶。

要在使用 VPC 端点时授予这些存储桶的托管式节点访问权限，您需要创建一个自定义 Amazon S3 权限策略，然后将其附加到您的实例配置文件（适用于 EC2 实例）或服务角色（适用于非 EC2 托管式节点）。

有关在 Systems Manager 操作中使用虚拟私有云（VPC）端点的信息，请参阅使用适用于 Systems Manager 的 VPC 端点提高 EC2 实例的安全性。

> **Note**
>
> 这些权限仅提供对 SSM Agent所需的 AWS 托管存储桶的访问。它们不提供其他 Amazon S3 操作所需的权限。它们也不提供您自己的 S3 存储桶的权限。

有关更多信息，请参阅以下主题：

- 配置 Systems Manager 所需的实例权限
- 在混合和多云环境中创建 Systems Manager 所需的 IAM 服务角色
- 参考：用于修补操作的 Amazon S3 存储桶

内容

- 所需的存储桶权限
- 示例
- 使用硬件指纹验证混合激活的计算机
- SSM Agent，发布时间：GitHub

所需的存储桶权限

下表介绍了对于 Systems Manager 操作 SSM Agent 可能需要访问的每个 S3 存储桶。

> **Note**
>
> *region* 表示 AWS Systems Manager 支持的 AWS 区域 的标识符，例如 us-east-2 对应美国东部（俄亥俄）区域。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

SSM Agent 所需的 Amazon S3 权限

S3 存储桶 ARN	描述
arn:aws:s3:::aws-windows-downloads- *region*/*	需要一些仅支持 Windows Server 操作系统的 SSM 文档，还需要一些用于跨平台支持，例如 AWSEC2-ConfigureSTIG 。
arn:aws:s3:::amazon-ssm- *region*/*	更新 SSM Agent安装所必需的。这些存储桶包含 SSM Agent安装软件包，以及 AWS-UpdateSSMAgent 文档和插件引用的安装清单。如果未提供这些权限，则 SSM Agent 会发出 HTTP 调用以下载更新。
arn:aws:s3:::aws-ssm- *region*/*	提供对 S3 存储桶的访问权限，该存储桶包含与非修补 Systems Manager 文档（SSM 命令文档）结合使用所需的模块。例如：arn:aws:s3:::aws-ssm-us-east-2/* 。 下面是存储在这些存储桶中的一些常用 SSM 文档。 • AWS-ConfigureWindowsUpdate • AWS-FindWindowsUpdates • AWS-UpdateSSMAgent • AWS-UpdateEC2Config
arn:aws:s3:::patch-baseline-snapshot- *region*/* –或者– arn:aws:s3:::patch-baseline-snapshot- *region-unique-suffix* /*	提供对包含补丁基准快照的 S3 存储桶的访问权限。如果您使用下列 SSM 命令文档，则这是必需的： • AWS-RunPatchBaseline • AWS-RunPatchBaselineAssociation • AWS-RunPatchBaselineWithHooks • AWS-ApplyPatchBaseline （原有 SSM 文档）

S3 存储桶 ARN	描述
	最支持 AWS 区域 的存储桶使用以下格式： `arn:aws:s3:::patch-baseline-snapshot-` *region* 对于某些区域，存储桶名称中还会包含额外的唯一后缀。例如，中东（巴林）区域（me-south-1）的存储桶名称如下： • `patch-baseline-snapshot-me-south-1-uduvl7q8` 有关补丁基准快照存储桶名称的完整列表，请参阅 包含 AWS 托管补丁基准快照的存储桶。 **Note** 如果您使用本地防火墙并计划使用 Patch Manager，还必须允许该防火墙访问适当的补丁基准终端节点。

S3 存储桶 ARN	描述
对于 Linux 和 Windows Server 托管式节点：`arn:aws:s3:::aws-patch-manager-` *region*`-`*unique-suffix*` /*` 对于 macOS 的 Amazon EC2 实例：`arn:aws:s3:::aws-patchmanager-macos-` *region*`-`*unique-suffix*` /*`	提供对 S3 存储桶的访问权限，该存储桶包含用于在 Patch Manager 中修补操作的 SSM 命令文档。每个存储桶名称都包含唯一后缀，例如美国东部（俄亥俄州）（us-east-2）区域的存储桶后缀为 552881074 ： • `arn:aws:s3:::aws-patch-manager-us-east-2-552881074/*` • `arn:aws:s3::: aws-patchmanager-macos-us-east-2-552881074/*` SSM 文档 下面是存储在这些存储桶中的一些常用 SSM 文档。 • `AWS-RunPatchBaseline` • `AWS-RunPatchBaselineAssociation` • `AWS-RunPatchBaselineWithHooks` • `AWS-InstanceRebootWithHooks` • `AWS-PatchAsgInstance` • `AWS-PatchInstanceWithRollback` 有关修补操作的 AWS 托管 S3 存储桶的完整列表，请参阅以下主题： • 包含用于修补操作的 SSM 命令文档的存储桶（Linux 和 Windows Server） • 包含用于修补操作的 SSM 命令文档的存储桶（macOS）

示例

以下示例演示了如何提供对美国东部（俄亥俄）区域 (us-east-2) 中 Systems Manager 操作所需的 S3 存储桶的访问权限。在大多数情况下，仅当使用 VPC 端点时，才需要在实例配置文件或服务角色中显式提供这些权限。

> **Important**
>
> 我们建议您避免在该策略中使用通配符 (*) 以替代特定区域。例如，使用 `arn:aws:s3:::aws-ssm-us-east-2/*` 而不使用 `arn:aws:s3:::aws-ssm-*/*`。使用通配符可能会提供对您不打算授予访问权限的 S3 存储桶的访问。如果要将实例配置文件用于多个区域，我们建议每个区域重复使用第一个 `Statement` 块。

JSON

```
{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": "s3:GetObject",
                "Resource": [
                    "arn:aws:s3:::aws-windows-downloads-us-east-2/*",
                    "arn:aws:s3:::amazon-ssm-us-east-2/*",
                    "arn:aws:s3:::aws-ssm-us-east-2/*",
                    "arn:aws:s3:::patch-baseline-snapshot-us-east-2/*",
                    "arn:aws:s3:::aws-patch-manager-us-east-2-552881074/*",
                    "arn:aws:s3:::aws-patchmanager-macos-us-east-2-552881074/*"
                ]
            }
        ]
    }
```

使用硬件指纹验证混合激活的计算机

当非 EC2 计算机处于混合和多云环境中时，SSM Agent 将收集许多系统属性（称为硬件哈希），并使用这些属性计算指纹。指纹是代理传递给某些 Systems Manager API 的不透明字符串。这种唯一的指

纹将调用方与特定混合激活的托管节点关联起来。代理将指纹和硬件哈希存储在本地磁盘上名为文件库的位置。

当计算机注册为与 Systems Manager 一起使用时，代理会计算硬件哈希和指纹。然后，在代理发送 `RegisterManagedInstance` 命令时，将指纹传递回 Systems Manager 服务。

稍后，当发送 `RequestManagedInstanceRoleToken` 命令时，代理将检查文件库中的指纹和硬件哈希，以确保当前计算机属性与存储的硬件哈希匹配。如果当前计算机属性确实与文件库中存储的硬件哈希匹配，则代理将在指纹中从文件库传递到 `RegisterManagedInstance`，从而成功调用。

如果当前计算机属性与存储的硬件哈希不匹配，SSM Agent 将计算新的指纹，将新的硬件哈希和指纹存储在文件库中，然后将新指纹传递给 `RequestManagedInstanceRoleToken`。这会导致 *`RequestManagedInstanceRoleToken`* 失败，并且代理将无法获取连接到 Systems Manager 服务的角色令牌。

此失败是特意设计的，将用作验证步骤，以防多个托管式节点作为同一托管式节点与 Systems Manager 服务进行通信。

将当前计算机属性与在文件库中存储的硬件哈希进行比较时，代理使用以下逻辑来确定旧哈希和新哈希是否匹配：

- 如果 SID（系统/计算机 ID）不同，则表示不匹配。
- 否则，如果 IP 地址相同，则表示匹配。
- 否则，将计算计算机属性的匹配百分比，并与用户配置的相似性阈值进行比较，以确定是否存在匹配。

相似性阈值作为硬件哈希的一部分存储在文件库中。

可以在注册实例后使用如下所示的命令设置相似性阈值。

在 Linux 计算机上：

```
sudo amazon-ssm-agent -fingerprint -similarityThreshold 1
```

在使用 PowerShell 的 Windows Server 计算机上：

```
cd "C:\Program Files\Amazon\SSM\" `
    .\amazon-ssm-agent.exe -fingerprint -similarityThreshold 1
```

> **Important**
>
> 如果用于计算指纹的组件之一发生了变化，则这可能会导致代理休眠。为了帮助避免出现这种休眠，请将相似性阈值设置为低值，例如 **1**。

SSM Agent，发布时间：GitHub

GitHub 上提供 SSM Agent 的源代码，便于您根据需要调整代理。我们鼓励您针对要包含的更改提交提取请求。但是，Amazon Web Services 不支持运行此软件的已修改副本。

查找预装了 SSM Agent 的 AMIs

AWS Systems Manager 代理（SSM Agent）已预先安装在由 AWS 和信任的第三方提供的某些 Amazon Machine Images（AMIs）上。

例如，使用以下操作系统之一启动从 AMI 创建的 Amazon Elastic Compute Cloud（Amazon EC2）实例时，您可能会发现 SSM Agent 已安装：

- AlmaLinux
- 2017 年 9 月及之后发布的 Amazon Linux 1 Base AMI
- Amazon Linux 2
- Amazon Linux 2 ECS 优化基础 AMIs
- Amazon Linux 2023（AL2023）
- Amazon EKS 优化版 Amazon Linux AMIs
- macOS 10.14.x（Mojave）、10.15.x（Catalina）、11.x（Big Sur）、12.x（Monterey）、13.x（Ventura）、14.x（Sonoma）和 15.x（Sequoia）
- SUSE Linux Enterprise Server (SLES) 12 和 15
- Ubuntu Server 16.04 LTS 64 位 (Snap)、18.04、20.04、22.04 LTS、23.10、24.04 LTS、24.0 和 25.04
- 2016 年 11 月或以后发布的 Windows Server 2008-2012 R2 AMIs
- Windows Server 2016、2019、2022（不包括 Nano 版本）和 2025

> **Note**
>
> 预装在 AMI 上的 SSM Agent 版本可能不是最新的可用版本。作为最佳实操，我们建议您始终在受管节点上使用最新可用的 SSM Agent 版本。有关自动 SSM Agent 更新的更多信息，请参阅 自动更新到 SSM Agent。
> SSM Agent 可能已预先安装在不在此列表中的 AWS 托管 AMIs 上。这通常表明并非所有 Systems Manager 工具都完全支持操作系统（OS）。
> SSM Agent 可能已预安装在 AWS Marketplace 或社区 AMIs 存储库中的 AMIs 上，但是，AWS 不支持这些 AMIs。

验证 SSM Agent 的状态

根据初始化的时间，从前面列表中的 AMI 创建的实例可能没有预安装 SSM Agent。也可能实例已预安装代理，但代理没有运行。因此，我们建议您在首次尝试在实例上使用 Systems Manager 之前先检查 SSM Agent 的状态。

使用以下过程验证您的实例上已安装并正在运行 SSM Agent。如果您发现未安装代理，可以在 Linux、macOS和 Windows Server 实例上手动安装。

要验证实例上是否已安装 SSM Agent

1. 启动新实例后，请等待几分钟，以便其完成初始化。
2. 使用您的首选方法连接到实例。例如，您可以使用 SSH 连接到 Linux 实例，或使用远程桌面连接到 Windows Server 实例。
3. 根据实例的操作系统类型运行命令，以检查 SSM Agent 的状态。

操作系统	命令
Amazon Linux 1	`sudo status amazon-ssm-agent`
Amazon Linux 2 和 Amazon Linux 2023	`sudo systemctl status amazon-ssm-agent`
macOS	macOS 上没有用于查看 SSM Agent 状态的命令。您可以通过查找和评估代理日志文件 `/var/log/amazon/ssm/amazon-ssm-agent.log` 来查看状态。

操作系统	命令
SUSE Linux Enterprise Server	`sudo systemctl status amazon-ssm-agent`
Ubuntu Server (32 位)	`sudo status amazon-ssm-agent`
Ubuntu Server (64 位 - Deb)	`sudo systemctl status amazon-ssm-agent`
Ubuntu Server (64 位 - Snap)	`sudo systemctl status snap.amazon-ssm-agent.amazon-ssm-agent.service`
Windows Server	`Get-Service AmazonSSMAgent`

> **Tip**
>
> 若要查看在 Systems Manager 支持的所有操作系统类型上查看 SSM Agent 状态的命令，请参阅 正在检查 SSM Agent 状态并启动代理。

4. 评估命令输出以了解的 SSM Agent 状态。

   状态：已安装且正在运行

   在大多数情况下，命令输出会表明代理已安装并正在运行。

   以下示例显示 Amazon Linux 2 实例上已安装且正在运行 SSM Agent。

```
amazon-ssm-agent.service - amazon-ssm-agent
Loaded: loaded (/usr/lib/systemd/system/amazon-ssm-agent.service; enabled; vendor
 preset: enabled)
Active: active (running) since Wed 2021-10-20 19:09:29 UTC; 4min 6s ago
--truncated--
```

   以下示例显示 Windows Server 实例上已安装且正在运行 SSM Agent。

```
Status   Name               DisplayName
```

```
------   ----               -----------
Running  AmazonSSMAgent     Amazon SSM Agent
```

状态：已安装但未运行

在某些情况下，命令输出会表明代理已安装但并未运行。

以下示例显示 Amazon Linux 2 实例上已安装 SSM Agent 但并未运行。

```
amazon-ssm-agent.service - amazon-ssm-agent
Loaded: loaded (/usr/lib/systemd/system/amazon-ssm-agent.service; enabled; vendor
 preset: enabled)
Active: inactive (dead) since Wed 2021-10-20 22:16:41 UTC; 18s ago
--truncated--
```

以下示例显示 Windows Server 实例上已安装 SSM Agent 但并未运行。

```
Status   Name               DisplayName
------   ----               -----------
Stopped  AmazonSSMAgent     Amazon SSM Agent
```

如果代理已安装但并未运行，您可以根据实例的操作系统类型使用命令手动激活代理。

操作系统	命令
Amazon Linux 1	`sudo start amazon-ssm-agent`

Amazon Linux 2 和 Amazon Linux 2023	`sudo systemctl enable amazon-ssm-agent`
	
	`sudo systemctl start amazon-ssm-agent`

操作系统	命令

macOS	`sudo launchctl load -w /Library/LaunchDaemons/com.amazon.aws.ssm.plist`
	
	`sudo launchctl start com.amazon.aws.ssm`

SUSE Linux Enterprise Server	`sudo systemctl enable amazon-ssm-agent`
	
	`sudo systemctl start amazon-ssm-agent`

Ubuntu Server (32 位)	`sudo start amazon-ssm-agent`

Ubuntu Server (64 位 - Deb)	`sudo systemctl enable amazon-ssm-agent`
	
	`sudo systemctl start amazon-ssm-agent`

Ubuntu Server (64 位 - Snap)	`sudo snap start amazon-ssm-agent`
Windows Server	在 PowerShell 中运行以下命令。 `Start-Service AmazonSSMAgent`

状态：未安装

在某些情况下，命令输出会表明座席未安装。

以下示例显示 Amazon Linux 2 实例上未安装 SSM Agent。

```
Unit amazon-ssm-agent.service could not be found.
```

以下示例显示 Windows Server 实例上未安装 SSM Agent。

```
Get-Service : Cannot find any service with service name 'AmazonSSMAgent'.
--truncated--
```

如果未安装代理，您可以根据操作系统类型使用程序手动安装：

- 在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent
- 在适用于 macOS 的 EC2 实例上手动安装和卸载 SSM Agent
- 在适用于 Windows Server 的 EC2 实例上手动安装和卸载 SSM Agent

在适用于 Linux 的 EC2 实例上使用 SSM Agent

AWS Systems Manager Agent (SSM Agent) 用于处理 Systems Manager 请求并按照请求中的规定配置计算机。使用以下主题中的程序在 Linux 操作系统上安装、配置或卸载 SSM Agent。

主题

- 验证 SSM Agent 签名
- 在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent
- 配置 SSM Agent 以在 Linux 节点上使用代理

验证 SSM Agent 签名

适用于 Linux 实例的AWS Systems Manager Agent (SSM Agent) deb 和 rpm 安装程序包是以加密方式签名的。您可以使用公钥验证代理软件包是否为未修改的原始包。如果文件有任何损坏或已更改，则验证将失败。您可以使用 RPM 或 GPG 验证安装程序包的签名。以下信息适用于 SSM Agent 版本 3.1.1141.0 或更高版本。

要查找您的实例架构和操作系统的正确签名文件，请参阅下表。

region 表示 AWS Systems Manager 支持的 AWS 区域 的标识符，例如 us-east-2 对应美国东部（俄亥俄）区域。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

架构	操作系统	签名文件 URL	代理下载文件名
x86_64	AlmaLinux、Amazon Linux 1、Amazon	https://s3. *region*.amazonaw	amazon-ssm-agent.rpm

架构	操作系统	签名文件 URL	代理下载文件名
	Linux 2、Amazon Linux 2023、CentOS、CentOS Stream、RHEL、Oracle Linux、Rocky Linux、SLES	s.com/amazon-ssm-*region*/latest/linux_amd64/amazon-ssm-agent.rpm.sig https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/linux_amd64/amazon-ssm-agent.rpm.sig	
x86_64	Debian Server, Ubuntu Server	https://s3. *region*.amazonaws.com/amazon-ssm-*region*/latest/debian_amd64/amazon-ssm-agent.deb.sig https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/debian_amd64/amazon-ssm-agent.deb.sig	amazon-ssm-agent.deb

架构	操作系统	签名文件 URL	代理下载文件名
x86	Amazon Linux 1、Amazon Linux 2、Amazon Linux 2023、CentOS、RHEL	https://s3.*region*.amazonaws.com/amazon-ssm-*region*/latest/linux_386/amazon-ssm-agent.rpm.sig https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/linux_386/amazon-ssm-agent.rpm.sig	amazon-ssm-agent.rpm

架构	操作系统	签名文件 URL	代理下载文件名
x86	Ubuntu Server	https://s3.*region*.amazonaws.com/amazon-ssm-*region*/latest/debian_386/amazon-ssm-agent.deb.sig https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/debian_386/amazon-ssm-agent.deb.sig	amazon-ssm-agent.deb

架构	操作系统	签名文件 URL	代理下载文件名

ARM64	Amazon Linux 1、Amazon Linux 2、Amazon Linux 2023、CentOS、RHEL	`https://s3.region.amazonaws.com/amazon-ssm-region/latest/linux_arm64/amazon-ssm-agent.rpm.sig`
		
		`https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/linux_arm64/amazon-ssm-agent.rpm.sig`
			`amazon-ssm-agent.rpm`

在 Linux 服务器上验证 SSM Agent 软件包（v3.3.1802.0 及更高版本）

开始前的准备工作

本部分中 GPG 和 RPM 的程序适用于 SSM Agent 版本 3.3.1802.0 及更高版本。在使用以下程序验证 SSM Agent 的签名之前，请确保已下载适用于操作系统的最新代理软件包。例如 `https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/linux_arm64/amazon-ssm-agent.rpm`。有关下载 SSM Agent 软件包的更多信息，请参阅 在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent。

如果您有理由继续使用代理版本 3.3.1611.0 或更早版本，则请改为按照 在 Linux 服务器上验证 SSM Agent 软件包（v3.3.1611.0 及更早版本） 中的说明进行操作。

GPG

要在 Linux 服务器上验证 SSM Agent 软件包（v3.3.1802.0 及更高版本）

1. 复制以下公钥并将其保存到名为 amazon-ssm-agent.gpg 的文件。

> **Important**
>
> 以下公有密钥将于 2026-07-15（2026 年 7 月 15 日）到期。在旧公有密钥失效之前，Systems Manager 将在本主题中发布新的公有密钥。我们建议您订阅此主题的 RSS 源，以便在新密钥发布时收到通知。

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2.0.22 (GNU/Linux)

mQINBGeRNq4BEACrlf5h6Pz+k+M+QCJJ2LfK7d2Tn9J8iJ9qBK2Vwvuxco1rpSO+
KEI3nTeysPuheximps8WOCADX4VlbsKxMZQLjQM4mA26m1Tiw9nAI4kod4bKjiuM
BMUTCD1wfnjH3zQi4kDUdbpfAEMiPgNLVLH85Wf+lhK+Zm+V38DYzLyVj03kX4wK
iG6RMoxzOBZa5gNsVq+j+oCUITGz/URxH713Rgo8WeoEegI0+7iCBLKg+PM0b7GV
2nzkwWJz796HdkqSg8BwXsYaLTrHxa2P1IpwPCisAkyO7gZaMd6Uj69dtMFO+V8a
Qee6b57qGuFKZw7h1Vvc85PbF1Gy/wNIpary57kUHBFUg1vYep/roJuEbJCq97r5
I2liLl4NAyrWb9r/TAVxlXvqM4iZUhxm8GAp0FywMdBr9ZECClKa5HxuVmlm0Wgl
TXoYTOZKeDg6ZoCvyhNxWneCNip74fohXymeFF5L/budhBwy5wuwSniOgTGLo/4C
VgZHWCcN+d0Q3bx/sl2QNqPg5/xzsxEtymXLdVdwLIsLdEQUnIvy8KTs5jol3Dwi
nnEEyhly6wdaw+qDOhkSOT/VnErrSMkYF8VJfa5GjhCBWKw9JVSkaP2CI/VHOgHM
MKROnulq0hRQBR7RmLYt98xu38BHJWMmF8Ga/HJuIxzD1VmkZOPvDDESUwARAQAB
tCdTU00gQWdlbnQgPHNzbS1hZ2VudC1zaWduZXJAYW1hem9uLmNvbT6JAj8EEwEC
ACkFAmeRNq4CGy8FCQLGmIAHCwkIBwMCAQYVCAIJCgsEFgIDAQIeAQIXgAAKCRBR
qOBQ0AUuXTdND/9qldQ1E3dYjBVXOnbhiUQL594bkS5VoEX7D4fZ5UMVZa5pGiz+
husnoRUS9rH1cSeq7aHJu9hSCMuMdvRpuoo0CwLB+7HtzJvAO2M01hcEkUYa6Qdj
njTzP0ZjnoenJmqF9SYmVqAI/VPa9mNQ1OJ+HQ3qh5i6w+FoWlVqEdXjZGrWijub
TqyN33i1Y26t7Os/x8I9fUeNx37y/7Kama8LTdtv9GhWiMVBg2IuVf27HCMYofrQ
m2uCGe61IhtsnhsYaYupmljl+6qgdiuCiS9BAsoIGtqTnu8lnKcGyGz6YnRszN+U
1bNE4w+UFpXWJF8ogpYcghJ06aW/LhjZnQSx3VliLdW8eOJzou41yWmiuL3ZY8eW
KAlD+7eYKS6N6fEJCeNO2VX2lcKtDfaOX+lqGIVyexKayMfpi+0frNzt/92YCpF5
3jkeS77vMMVqKIUiIp1OCGv3XsFpIr6Bt2c2throYPDoQL3zvq6vvG40BKeRQ4tT
Y+5vTc8MeNn3LdzTl9pusxTcKifrJq7f5FIsL2CpAX8uQ+Qz+XWsYQQ5PvyUDtOz
nU/MRZaP6HnqY42bzI9ZlKgXi9IE3MXIwoET9YyzFjkIDvat7SlB4uJCpeIqp/KM
OIrTMb7paGLYmBU6YqxNBkDWItNG7NeZzyhh/R/Qqb4vJaf4S+ZqD1RZXokCHAQQ
AQIABgUCZ5E2rwAKCRB90Jej2tf1/CdnD/46It+RNoE00TesZK5n2bijH5Eljw0E
4/UpMi1SV6t2zY7lIm7TcKNn18tynJNFqB6YXXOwSbBG/fbN2E9RaoUCZw23TmAv
```

```
amuHwrfsDqsHb7zzPF0bISYjqEDLQJj/gtEugUc6XY1dEpFSlWJIOvgryG04cFXI
uD2KY87ya4s1R+sEVAJ14K4RlUCiMmzJdR0NJNYJOwBi1gkLEp6jG86ttiG2U7fY
pE2ibV+c0GeIFq8PIzqqENsn9KBuRH5EcbdBwfnsj2XfM4aR3ZtRIdWXkKkdP9Rs
yU5dTF/Y7XPId5h8/gp00+DMlXFBinQ1jE7A7eDYviEFd1ba8P7dIom3Q3gzKiWu
KTGpnykShs5NvpQmvGUF6JqDHI4RK9s3kLqsNyZkhenJfRBrJ/45fQAuP4CRedkF
7PSfX0Xp7kDnKuyK6wEUEfXXrqmuLGDmigTXblO5qgdyMwkOLjiY9znBZbHoKs76
VplOoNgGnN19i3nuMcPf2npFICJv7kTIyn5Fh7pjWDCahl3U/PwoLjrrlEzpyStU
oXSZrK3kiAADEdSODXJl8KYU0Pb27JbRr1ZbWnxb+O39TOhtssstulkR0v+IDGDQ
rQE1b12sKgcNFSzInzWrNGu4S06WN8DYzlrTZ9aSHj+37ZqpXAevi8WOFXKPV3PA
E6+O8RI2451Dcg==
=aDkv
-----END PGP PUBLIC KEY BLOCK-----
```

2. 将公钥导入到您的密钥环中，并记下返回的键值。

```
gpg --import amazon-ssm-agent.gpg
```

3. 验证指纹。请务必将 *key-value* 替换为上一步中的值。我们建议您使用 GPG 来验证指纹，即使您使用 RPM 验证安装程序包也是如此。

```
gpg --fingerprint key-value
```

该命令会返回类似以下内容的输出：

```
pub   4096R/D0052E5D 2025-01-22 [expires: 2026-07-15]
      Key fingerprint = 4855 A9E6 8332 16D6 A77D  8FE4 51A8 E050 D005 2E5D
uid                  SSM Agent <ssm-agent-signer@amazon.com>
```

指纹应与以下内容匹配。

```
4855 A9E6 8332 16D6 A77D 8FE4 51A8 E050 D005 2E5D
```

否则，请勿安装该代理。联系 AWS 支持。

4. 根据您的实例架构和操作系统下载签名文件（如果您尚未执行此操作）。
5. 验证安装程序包签名。请务必将 *signature-filename* 和 *agent-download-filename* 替换为下载签名文件和代理时指定的值，如本主题前面的表中所列。

```
gpg --verify signature-filename agent-download-filename
```

例如，对于 Amazon Linux 2 上的 x86_64 架构：

```
gpg --verify amazon-ssm-agent.rpm.sig amazon-ssm-agent.rpm
```

该命令会返回类似以下内容的输出：

```
gpg: Signature made Sat 08 Feb 2025 12:05:08 AM UTC using RSA key ID D0052E5D
gpg: Good signature from "SSM Agent <ssm-agent-signer@amazon.com>"
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: 4855 A9E6 8332 16D6 A77D  8FE4 51A8 E050 D005 2E5D
```

如果输出包含短语 BAD signature，则检查是否正确执行了此过程。如果您继续获得该响应，请联系 支持 且不要安装该代理。有关信任的警告消息并不意味着签名无效，只是您尚未验证公有密钥而已。只有当您或您信任的某个人对密钥进行了签名，密钥才是可信的。如果输出包含短语 Can't check signature: No public key，则验证您下载的是 SSM Agent 版本 3.1.1141.0 还是更高版本。

RPM

要在 Linux 服务器上验证 SSM Agent 软件包（v3.3.1802.0 及更高版本）

1. 复制以下公有密钥并将其保存到名为 amazon-ssm-agent.gpg 的文件。

> **Important**
>
> 以下公有密钥将于 2026-07-15（2026 年 7 月 15 日）到期。在旧公有密钥失效之前，Systems Manager 将在本主题中发布新的公有密钥。我们建议您订阅此主题的 RSS 源，以便在新密钥发布时收到通知。

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2.0.22 (GNU/Linux)

mQINBGeRNq4BEACrlf5h6Pz+k+M+QCJJ2LfK7d2Tn9J8iJ9qBK2Vwvuxco1rpSO+
KEI3nTeysPuheximps8WOCADX4VlbsKxMZQLjQM4mA26m1Tiw9nAI4kod4bKjiuM
BMUTCD1wfnjH3zQi4kDUdbpfAEMiPgNLVLH85Wf+lhK+Zm+V38DYzLyVj03kX4wK
iG6RMoxzOBZa5gNsVq+j+oCUITGz/URxH713Rgo8WeoEegI0+7iCBLKg+PM0b7GV
2nzkwWJz796HdkqSg8BwXsYaLTrHxa2P1IpwPCisAkyO7gZaMd6Uj69dtMFO+V8a
Qee6b57qGuFKZw7h1Vvc85PbF1Gy/wNIpary57kUHBFUg1vYep/roJuEbJCq97r5
```

```
I2liLl4NAyrWb9r/TAVxlXvqM4iZUhxm8GAp0FywMdBr9ZECClKa5HxuVmlm0Wgl
TXoYTOZKeDg6ZoCvyhNxWneCNip74fohXymeFF5L/budhBwy5wuwSniOgTGLo/4C
VgZHWCcN+d0Q3bx/sl2QNqPg5/xzsxEtymXLdVdwLIsLdEQUnIvy8KTs5jol3Dwi
nnEEyhly6wdaw+qDOhkSOT/VnErrSMkYF8VJfa5GjhCBWKw9JVSkaP2CI/VHOgHM
MKROnulq0hRQBR7RmLYt98xu38BHJWMmF8Ga/HJuIxzD1VmkZOPvDDESUwARAQAB
tCdTU00gQWdlbnQgPHNzbS1hZ2VudC1zaWduZXJAYW1hem9uLmNvbT6JAj8EEwEC
ACkFAmeRNq4CGy8FCQLGmIAHCwkIBwMCAQYVCAIJCgsEFgIDAQIeAQIXgAAKCRBR
qOBQ0AUuXTdND/9qldQ1E3dYjBVXOnbhiUQL594bkS5VoEX7D4fZ5UMVZa5pGiz+
husnoRUS9rH1cSeq7aHJu9hSCMuMdvRpuoo0CwLB+7HtzJvAO2M01hcEkUYa6Qdj
njTzP0ZjnoenJmqF9SYmVqAI/VPa9mNQ1OJ+HQ3qh5i6w+FoWlVqEdXjZGrWijub
TqyN33i1Y26t7Os/x8I9fUeNx37y/7Kama8LTdtv9GhWiMVBg2IuVf27HCMYofrQ
m2uCGe61IhtsnhsYaYupmljl+6qgdiuCiS9BAsoIGtqTnu8lnKcGyGz6YnRszN+U
1bNE4w+UFpXWJF8ogpYcghJ06aW/LhjZnQSx3VliLdW8eOJzou41yWmiuL3ZY8eW
KAlD+7eYKS6N6fEJCeNO2VX2lcKtDfaOX+lqGIVyexKayMfpi+0frNzt/92YCpF5
3jkeS77vMMVqKIUiIp1OCGv3XsFpIr6Bt2c2throYPDoQL3zvq6vvG40BKeRQ4tT
Y+5vTc8MeNn3LdzTl9pusxTcKifrJq7f5FIsL2CpAX8uQ+Qz+XWsYQQ5PvyUDtOz
nU/MRZaP6HnqY42bzI9ZlKgXi9IE3MXIwoET9YyzFjkIDvat7SlB4uJCpeIqp/KM
OIrTMb7paGLYmBU6YqxNBkDWItNG7NeZzyhh/R/Qqb4vJaf4S+ZqD1RZXokCHAQQ
AQIABgUCZ5E2rwAKCRB90Jej2tf1/CdnD/46It+RNoE00TesZK5n2bijH5Eljw0E
4/UpMi1SV6t2zY7lIm7TcKNn18tynJNFqB6YXXOwSbBG/fbN2E9RaoUCZw23TmAv
amuHwrfsDqsHb7zzPF0bISYjqEDLQJj/gtEugUc6XY1dEpFSlWJIOvgryG04cFXI
uD2KY87ya4s1R+sEVAJ14K4RlUCiMmzJdR0NJNYJOwBi1gkLEp6jG86ttiG2U7fY
pE2ibV+c0GeIFq8PIzqqENsn9KBuRH5EcbdBwfnsj2XfM4aR3ZtRIdWXkKkdP9Rs
yU5dTF/Y7XPId5h8/gp00+DMlXFBinQ1jE7A7eDYviEFd1ba8P7dIom3Q3gzKiWu
KTGpnykShs5NvpQmvGUF6JqDHI4RK9s3kLqsNyZkhenJfRBrJ/45fQAuP4CRedkF
7PSfX0Xp7kDnKuyK6wEUEfXXrqmuLGDmigTXblO5qgdyMwkOLjiY9znBZbHoKs76
VplOoNgGnN19i3nuMcPf2npFICJv7kTIyn5Fh7pjWDCahl3U/PwoLjrrlEzpyStU
oXSZrK3kiAADEdSODXJl8KYU0Pb27JbRr1ZbWnxb+O39TOhtssstulkR0v+IDGDQ
rQE1b12sKgcNFSzInzWrNGu4S06WN8DYzlrTZ9aSHj+37ZqpXAevi8WOFXKPV3PA
E6+O8RI2451Dcg==
=aDkv
-----END PGP PUBLIC KEY BLOCK-----
```

2. 将公钥导入到您的密钥环中，并记下返回的键值。

```
rpm --import amazon-ssm-agent.gpg
```

3. 验证指纹。我们建议您使用 GPG 来验证指纹，即使您使用 RPM 验证安装程序包也是如此。

```
rpm -qa gpg-pubkey --qf '%{Description}' | gpg --with-fingerprint | grep -A 1
 "ssm-agent-signer@amazon.com"
```

该命令会返回类似以下内容的输出：

```
pub  4096R/D0052E5D 2025-01-22 SSM Agent <ssm-agent-signer@amazon.com>
      Key fingerprint = 4855 A9E6 8332 16D6 A77D  8FE4 51A8 E050 D005 2E5D
```

指纹应与以下内容匹配。

```
4855 A9E6 8332 16D6 A77D 8FE4 51A8 E050 D005 2E5D
```

否则，请勿安装该代理。联系 AWS 支持。

4. 验证安装程序包签名。请务必将 *agent-download-filename* 替换为下载代理时指定的值，如本主题前面的表中所列。

```
rpm --checksig agent-download-filename
```

例如，对于 Amazon Linux 2 上的 x86_64 架构：

```
rpm --checksig amazon-ssm-agent.rpm
```

该命令会返回类似以下内容的输出。

```
amazon-ssm-agent.rpm: rsa sha1 md5 OK
```

如果输出中缺少 pgp，并且您已导入公钥，则不会对代理进行签名。如果输出包含短语 NOT OK (MISSING KEYS: (MD5) *key-id*)，则检查是否正确执行了此过程，并验证您下载的是 SSM Agent 版本 3.1.1141.0 还是更高版本。如果您继续获得该响应，请联系 支持 且不要安装该代理。

在 Linux 服务器上验证 SSM Agent 软件包（v3.3.1611.0 及更早版本）

开始前的准备工作

本部分中 GPG 和 RPM 的程序适用于 SSM Agent 版本 3.3.1611.0 及更早版本。我们始终建议使用最新版本的代理。有关信息，请参阅在 Linux 服务器上验证 SSM Agent 软件包（v3.3.1802.0 及更高版本）。但是，如果您有特定理由继续使用代理版本 3.3.1611.0 或更早版本，请改为按照以下程序之一中的说明进行操作。

GPG

要在 Linux 服务器上验证 SSM Agent 软件包（v3.3.1611.0 及更早版本）

1. 复制以下公有密钥并将其保存到名为 amazon-ssm-agent.gpg 的文件。

> **Important**
>
> 下面显示的公有密钥已于 2025-02-17（2025 年 2 月 17 日）过期，适用于版本 3.3.1611.0 和 3.2.1542.0 之前的早期版本，并且前提是该公有密钥之前用于验证代理的签名。在旧公有密钥失效之前，Systems Manager 将在本主题中发布新的公有密钥。我们建议您订阅此主题的 RSS 源，以便在新密钥发布时收到通知。

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2.0.22 (GNU/Linux)

mQENBGTtIoIBCAD2M1aoGIE0FXynAHM/jtuvdAVVaX3Q4ZejTqrX+Jq8ElAMhxyO
GzHu2CDtCYxtVxXK3unptLVt2kGgJwNbhYC393jDeZx5dCda4Nk2YXX1UK3P461i
axuuXRzMYvfM4RZn+7bJTu635tA07q9Xm6MGD4TCTvsjBfViOxbrxOg5ozWbJdSw
fSR8MwUrRfmFpAefRlYfCEuZ8FHywa9U6jLeWt2O/kqrZliJOAGjGzXtB7EZkqKb
faCCxikjjvhF1awdEqSK4DQorC/OvQc4I5kP5y2CJbtXvXO73QH2yE75JMDIIx9x
rOsIRUoSfK3UrWaOVuAnEEn5ueKzZNqGG1J1ABEBAAG0J1NTTSBBZ2VudCA8c3Nt
LWFnZW50LXNpZ25lckBhbWF6b24uY29tPokBPwQTAQIAKQUCZO0iggIbLwUJAsaY
gAcLCQgHAwIBBhUIAgkKCwQWAgMBAh4BAheAAAoJELwfSVyX3QTt+icH/A//tJsW
I+7Ay8FGJh8dJPNy++HIBjVSFdGNJFWNbw1Z8uZcazHEcUCH3FhW4CLQLTZ3OVPz
qvFwzDtRDVIN/Y9EGDhLMFvimrE+/z4olWsJ5DANf6BnX8I5UNIcRt5d8SWH1BEJ
2FWIBZFgKyTDI6XzRC5x4ahtgpOVAGeeKDehs+wh6Ga4W0/K4GsviP1Kyr+Ic2br
NAIq0q0IHyN1q9zam3Y0+jKwEuNmTj+Bjyzshyv/X8S0JWWoXJhkexkOvWeBYNNt
5wI4QcSteyfIzp6KlQF8q11Hzz9D9WaPfcBEYyhq7vLEARobkbQMBzpkmaZua241
0RaWG50HRvrgm4aJAhwEEAECAAYFAmTtIoMACgkQfdCXo9rX9fwwqBAAzkTgYJ38
sWgxpn7Ux/81F2BWR1sVkmP79i++fXyJlKI8xtcJFQZhzeUos69KBUCy7mgx5bYU
P7NA5o9DUbwz/QS0i1Cqm4+jtFlX0MXe4FikXcqfDPnnzN8mVB2H+fa43iHR1PuH
GgUWuNdxzSoIYRmLZXWmeN5YXPcmixlhLzcE2TOQn1mOKcu2fKdLtBQ8KiEkmjiu
naoLxnUcyk1zMhaha+LzEkQdOyasix0ggylN2ViWVnlmfy0niuXDxW0qZWPdLStF
OODiX3iqGmkH3rDfy6nvxxBR4GIs+MGD72fpWzzrINDgkGI2i2t1+0AX/mps3aTy
+ftlgrim8stYWB58XXDAb0vad06sNye5/zDzfr0I9HupJrTzFhaYJQjWPaSlINto
LDJnBXohiUIPRYRcy/k012oFHDWZHT3H6CyjK9UD5UlxA9H7dsJurANs6FOVRe+7
34uJyxDZ/W7zLG4AVG0zxibrUSoaJxwcOjVPVsQAlrwG/GTs7tcAccsJqbJ1Py/w
9AgJl8VU2qc8POsHNXk348gjP7C8PDnGMpZFzr9f5INctRushpiv7onX+aWJVX7T
n2uX/TP3LCyH/MsrNJrJOQnMYFRLQitciP0E+F+eA3v9CY6mDuyb8JSx5HuGGUsG
S4bKBOcA8vimEpwPoT8CE7fdsZ3Qkwdu+pw=
```

```
=zr5w
-----END PGP PUBLIC KEY BLOCK-----
```

2. 将公钥导入到您的密钥环中，并记下返回的键值。

```
gpg --import amazon-ssm-agent.gpg
```

3. 验证指纹。请务必将 *key-value* 替换为上一步中的值。我们建议您使用 GPG 来验证指纹，即使您使用 RPM 验证安装程序包也是如此。

```
gpg --fingerprint key-value
```

该命令会返回类似以下内容的输出。

```
pub   2048R/97DD04ED 2023-08-28 [expired: 2025-02-17]
      Key fingerprint = DE92 C7DA 3E56 E923 31D6 2A36 BC1F 495C 97DD 04ED
uid                  SSM Agent <ssm-agent-signer@amazon.com>
```

指纹应与以下内容匹配。

```
DE92 C7DA 3E56 E923 31D6 2A36 BC1F 495C 97DD 04ED
```

否则，请勿安装该代理。联系 AWS 支持。

4. 根据您的实例架构和操作系统下载签名文件（如果您尚未执行此操作）。
5. 验证安装程序包签名。请务必将 *signature-filename* 和 *agent-download-filename* 替换为下载签名文件和代理时指定的值，如本主题前面的表中所列。

```
gpg --verify signature-filename agent-download-filename
```

例如，对于 Amazon Linux 2 上的 x86_64 架构：

```
gpg --verify amazon-ssm-agent.rpm.sig amazon-ssm-agent.rpm
```

该命令会返回类似以下内容的输出：

```
gpg: Signature made Fri 10 Jan 2025 01:54:18 AM UTC using RSA key ID 97DD04ED
gpg: Good signature from "SSM Agent <ssm-agent-signer@amazon.com>"
gpg: Note: This key has expired!
Primary key fingerprint: DE92 C7DA 3E56 E923 31D6  2A36 BC1F 495C 97DD 04ED
```

如果输出包含短语 BAD signature，则检查是否正确执行了此过程。如果您继续获得该响应，请联系 支持 且不要安装该代理。有关信任的警告消息并不意味着签名无效，只是您尚未验证公有密钥而已。只有当您或您信任的某个人对密钥进行了签名，密钥才是可信的。如果输出包含短语 Can't check signature: No public key，则验证您下载的是 SSM Agent 版本 3.1.1141.0 还是更高版本。

RPM

要在 Linux 服务器上验证 SSM Agent 软件包（v3.3.1611.0 及更早版本）

1. 复制以下公有密钥并将其保存到名为 amazon-ssm-agent.gpg 的文件。

> **Important**
>
> 下面显示的公有密钥已于 2025-02-17（2025 年 2 月 17 日）过期，适用于版本 3.3.1611.0 和 3.2.1542.0 之前的早期版本，并且前提是该公有密钥之前用于验证代理的签名。在旧公有密钥失效之前，Systems Manager 将在本主题中发布新的公有密钥。我们建议您订阅此主题的 RSS 源，以便在新密钥发布时收到通知。

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2.0.22 (GNU/Linux)

mQENBGTtIoIBCAD2M1aoGIE0FXynAHM/jtuvdAVVaX3Q4ZejTqrX+Jq8ElAMhxyO
GzHu2CDtCYxtVxXK3unptLVt2kGgJwNbhYC393jDeZx5dCda4Nk2YXX1UK3P461i
axuuXRzMYvfM4RZn+7bJTu635tA07q9Xm6MGD4TCTvsjBfViOxbrxOg5ozWbJdSw
fSR8MwUrRfmFpAefRlYfCEuZ8FHywa9U6jLeWt2O/kqrZliJOAGjGzXtB7EZkqKb
faCCxikjjvhF1awdEqSK4DQorC/OvQc4I5kP5y2CJbtXvXO73QH2yE75JMDIIx9x
rOsIRUoSfK3UrWaOVuAnEEn5ueKzZNqGG1J1ABEBAAG0J1NTTSBBZ2VudCA8c3Nt
LWFnZW50LXNpZ25lckBhbWF6b24uY29tPokBPwQTAQIAKQUCZO0iggIbLwUJAsaY
gAcLCQgHAwIBBhUIAgkKCwQWAgMBAh4BAheAAAoJELwfSVyX3QTt+icH/A//tJsW
I+7Ay8FGJh8dJPNy++HIBjVSFdGNJFWNbw1Z8uZcazHEcUCH3FhW4CLQLTZ3OVPz
qvFwzDtRDVIN/Y9EGDhLMFvimrE+/z4olWsJ5DANf6BnX8I5UNIcRt5d8SWH1BEJ
2FWIBZFgKyTDI6XzRC5x4ahtgpOVAGeeKDehs+wh6Ga4W0/K4GsviP1Kyr+Ic2br
NAIq0q0IHyN1q9zam3Y0+jKwEuNmTj+Bjyzshyv/X8S0JWWoXJhkexkOvWeBYNNt
5wI4QcSteyfIzp6KlQF8q11Hzz9D9WaPfcBEYyhq7vLEARobkbQMBzpkmaZua241
0RaWG50HRvrgm4aJAhwEEAECAAYFAmTtIoMACgkQfdCXo9rX9fwwqBAAzkTgYJ38
sWgxpn7Ux/81F2BWR1sVkmP79i++fXyJlKI8xtcJFQZhzeUos69KBUCy7mgx5bYU
P7NA5o9DUbwz/QS0i1Cqm4+jtFlX0MXe4FikXcqfDPnnzN8mVB2H+fa43iHR1PuH
```

```
GgUWuNdxzSoIYRmLZXWmeN5YXPcmixlhLzcE2TOQn1mOKcu2fKdLtBQ8KiEkmjiu
naoLxnUcyk1zMhaha+LzEkQdOyasix0ggylN2ViWVnlmfy0niuXDxW0qZWPdLStF
OODiX3iqGmkH3rDfy6nvxxBR4GIs+MGD72fpWzzrINDgkGI2i2t1+0AX/mps3aTy
+ftlgrim8stYWB58XXDAb0vad06sNye5/zDzfr0I9HupJrTzFhaYJQjWPaSlINto
LDJnBXohiUIPRYRcy/k012oFHDWZHT3H6CyjK9UD5UlxA9H7dsJurANs6FOVRe+7
34uJyxDZ/W7zLG4AVG0zxibrUSoaJxwcOjVPVsQAlrwG/GTs7tcAccsJqbJ1Py/w
9AgJl8VU2qc8POsHNXk348gjP7C8PDnGMpZFzr9f5INctRushpiv7onX+aWJVX7T
n2uX/TP3LCyH/MsrNJrJOQnMYFRLQitciP0E+F+eA3v9CY6mDuyb8JSx5HuGGUsG
S4bKBOcA8vimEpwPoT8CE7fdsZ3Qkwdu+pw=
=zr5w
-----END PGP PUBLIC KEY BLOCK-----
```

2. 将公钥导入到您的密钥环中，并记下返回的键值。

```
rpm --import amazon-ssm-agent.gpg
```

3. 验证指纹。我们建议您使用 GPG 来验证指纹，即使您使用 RPM 验证安装程序包也是如此。

```
rpm -qa gpg-pubkey --qf '%{Description}' | gpg --with-fingerprint | grep -A 1
 "ssm-agent-signer@amazon.com"
```

该命令会返回类似以下内容的输出：

```
pub  2048R/97DD04ED 2023-08-28 SSM Agent <ssm-agent-signer@amazon.com>
      Key fingerprint = DE92 C7DA 3E56 E923 31D6 2A36 BC1F 495C 97DD 04ED
```

指纹应与以下内容匹配。

```
DE92 C7DA 3E56 E923 31D6 2A36 BC1F 495C 97DD 04ED
```

否则，请勿安装该代理。联系 AWS 支持。

4. 验证安装程序包签名。请务必将 *agent-download-filename* 替换为下载代理时指定的值，如本主题前面的表中所列。

```
rpm --checksig agent-download-filename
```

例如，对于 Amazon Linux 2 上的 x86_64 架构：

```
rpm --checksig amazon-ssm-agent.rpm
```

该命令会返回类似以下内容的输出。

```
amazon-ssm-agent.rpm: rsa sha1 md5 OK
```

如果输出中缺少 pgp，并且您已导入公钥，则不会对代理进行签名。如果输出包含短语 NOT OK (MISSING KEYS: (MD5) *key-id*)，则检查是否正确执行了此过程，并验证您下载的是 SSM Agent 版本 3.1.1141.0 还是更高版本。如果您继续获得该响应，请联系 支持 且不要安装该代理。

在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent

在 Amazon Elastic Compute Cloud (Amazon EC2) Linux 操作系统上手动安装 AWS Systems Manager Agent (SSM Agent) 之前，查看以下信息。

SSM Agent 安装文件 URL

您可以访问存储在任何商业 AWS 区域 中的 SSM Agent 的安装文件。我们还在全球可用的 Amazon Simple Storage Service（Amazon S3）存储桶中提供安装文件，您可以将其用作文件的备用源或备份源。

如果您要在一个或两个实例上手动安装代理，则可以使用我们提供的快速安装程序中的命令以节省时间。这些过程中提供的命令也可以通过用户数据作为脚本传递给 Amazon EC2 实例。

如果您正在创建脚本或模板以用于在多个实例上安装代理，我们建议您在您所在 AWS 区域 地理位置或附近使用安装文件。对于批量安装，这可以提高下载速度并减少延迟。在这些情况下，我们建议使用安装主题中的创建自定义安装命令程序。

已预安装代理的 Amazon Machine Images

SSM Agent 预安装在 AWS 提供的一些 Amazon Machine Images（AMIs）上。有关信息，请参阅 查找预装了 SSM Agent 的 AMIs。

在其他机器类型上安装

如果需要在本地服务器或虚拟机（VM）上安装代理，以与 Systems Manager 配合使用，请参阅如何在混合 Linux 节点上安装 SSM Agent。有关在边缘设备上安装代理的信息，请参阅 使用 Systems Manager 管理边缘设备。

将代理保持最新状态

如果有新工具添加至 Systems Manager 或者对现有工具进行了更新，则将发布 SSM Agent 的更新版本。无法使用代理的最新版本可能会阻止托管式节点使用 Systems Manager 的各项工具和功能。因此，我们建议您自动完成确保机器上的 SSM Agent 为最新的过程。有关更多信息，请参阅 自动更新到 SSM Agent。要获得有关 SSM Agent 更新的通知，请在 GitHub 上订阅 SSM Agent 发布说明页面。

选择操作系统

要查看在指定操作系统上手动安装 SSM Agent 的过程，请从以下列表中选择一个链接：

> **Note**
>
> 有关下列每个操作系统的支持版本列表，请参阅 Systems Manager 支持的操作系统。

- AlmaLinux
- Amazon Linux 2 和 Amazon Linux 2023
- Amazon Linux 1
- CentOS
- CentOS Stream
- Debian Server
- Oracle Linux
- Red Hat Enterprise Linux
- Rocky Linux
- SUSE Linux Enterprise Server
- Ubuntu Server

从 Linux 实例卸载 SSM Agent

使用适用于操作系统的程序包管理器从 Linux 实例卸载 SSM Agent。根据不同的操作系统，卸载命令将与以下示例命令类似：

```
sudo dpkg -r amazon-ssm-agent
```

在 AlmaLinux 实例上手动安装 SSM Agent

使用本节中的信息可帮助您在 AlmaLinux 实例上手动安装或重新安装 SSM Agent。

开始前的准备工作

在 AlmaLinux 实例上安装 SSM Agent 之前，请注意以下事项：

- 确保在您的 AlmaLinux 实例上安装了 Python 3。这是 SSM Agent 正常工作所必需的。
- 有关适用于在所有基于 Linux 的操作系统上安装 SSM Agent 的重要信息，请参阅 在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent。

主题

- AlmaLinux 上 SSM Agent 的快速安装命令
- 为您所在区域的 AlmaLinux 创建自定义代理安装命令

AlmaLinux 上 SSM Agent 的快速安装命令

使用以下步骤在单个实例上手动安装 SSM Agent。此过程使用全局可用的安装文件。

开始前的准备工作

在 AlmaLinux 实例上安装 SSM Agent 之前，请注意以下事项：

- 确保在您的 AlmaLinux 实例上安装了 Python 3。这是 SSM Agent 正常工作所必需的。

在 AlmaLinux 上安装 SSM Agent

1. 使用首选方法（例如 SSH）连接到您的 AlmaLinux 实例。
2. 复制实例架构的命令并在实例上运行它。

> **Note**
>
> 即使以下命令中的 URL 包含 `ec2-downloads-windows` 目录，这些也是 AlmaLinux 的正确全局安装文件。

x86_64 实例

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

ARM64 实例

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. （建议）请使用以下命令验证代理是否正在运行。

```
sudo systemctl status amazon-ssm-agent
```

在大多数情况下，命令会报告代理正在运行，如下面的示例所示。

```
# amazon-ssm-agent.service - amazon-ssm-agent
   Loaded: loaded (/etc/systemd/system/amazon-ssm-agent.service; enabled; vendo>
   Active: active (running) since Tue 2022-04-19 16:40:41 UTC; 9s ago
 Main PID: 4898 (amazon-ssm-agen)
    Tasks: 14 (limit: 4821)
   Memory: 34.6M
   CGroup: /system.slice/amazon-ssm-agent.service
           ##4898 /usr/bin/amazon-ssm-agent
           ##4954 /usr/bin/ssm-agent-worker
            --truncated--
```

在极少数情况下，命令会报告代理已安装但未运行，如下面的示例所示。

```
# amazon-ssm-agent.service - amazon-ssm-agent
   Loaded: loaded (/etc/systemd/system/amazon-ssm-agent.service; enabled; vendo>
   Active: inactive (dead) since Tue 2022-04-19 16:42:05 UTC; 2s ago
            --truncated--
```

要在这些情况下激活代理，请运行以下命令。

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

为您所在区域的 AlmaLinux 创建自定义代理安装命令

在使用脚本或模板在多个实例上安装 SSM Agent 时，建议使用存储在工作所在的 AWS 区域中的安装文件。

对于以下命令，我们提供了在美国东部（俄亥俄州）区域（us-east-2）使用可公开访问 S3 存储桶的示例。

> **Tip**
>
> 您还可以使用您构造的自定义区域 URL 替换本主题前面的 AlmaLinux 上 SSM Agent 的快速安装命令 程序中的全局 URL。

在下面的命令中，将 *region* 替换为您自己的信息。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

x86_64

```
sudo dnf install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

请参阅以下示例。

```
sudo dnf install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_amd64/amazon-ssm-agent.rpm
```

ARM64

```
sudo dnf install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_arm64/amazon-ssm-agent.rpm
```

请参阅以下 示例。

```
sudo dnf install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_arm64/amazon-ssm-agent.rpm
```

在 Amazon Linux 2 和 Amazon Linux 2023 实例上手动安装 SSM Agent

> **Important**
>
> 本主题提供了用于在 Amazon Linux 2 和 Amazon Linux 2023 实例上使用 SSM Agent 的命令。其中一些命令不适用于 Amazon Linux 1 实例。在继续操作之前，请验证您正在查看的主题是否与您的实例类型对应。有关要在 Amazon Linux 1 实例上运行的命令，请参阅 在 Amazon Linux 1 实例上手动安装 SSM Agent。

在大多数情况下，AWS 提供的适用于 Amazon Linux 2 和 Amazon Linux 2023 的 Amazon Machine Images（AMIs）附带默认情况下预安装的 AWS Systems Manager 代理（SSM Agent）。有关更多信息，请参阅 查找预装了 SSM Agent 的 AMIs。

如果 SSM Agent 未预安装在新的 Amazon Linux 2 或 Amazon Linux 2023 实例上，或者如果您需要手动重新安装代理，请使用此页面上的信息来帮助您。

开始前的准备工作

在 Amazon Linux 2 或 Amazon Linux 2023 实例上安装 SSM Agent 之前，请注意以下事项：

- 有关适用于在所有基于 Linux 的操作系统上安装 SSM Agent 的重要信息，请参阅 在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent。
- 如果您在使用 SSM 文档 `AWS-UpdateSSMAgent` 成功安装或更新代理之后，使用 `yum` 命令来更新托管节点上的 SSM Agent，系统可能会显示以下消息：`"Warning: RPMDB altered outside of yum."`。此消息是预期中的，您可以安全地将其忽略。

主题

- Amazon Linux 2 或 Amazon Linux 2023 上的 SSM Agent 的快速安装命令
- 为您所在区域的 Amazon Linux 2 或 Amazon Linux 2023 创建自定义代理安装命令

Amazon Linux 2 或 Amazon Linux 2023 上的 SSM Agent 的快速安装命令

使用以下步骤在单个实例上手动安装 SSM Agent。此过程使用全局可用的安装文件。

使用快速复制和粘贴命令在 Amazon Linux 2 或 Amazon Linux 2023 上安装 SSM Agent

1. 使用首选方法（例如 SSH）连接到您的 Amazon Linux 2 或 Amazon Linux 2023 实例。

2. 复制实例架构的命令并在实例上运行它。

> **Note**
>
> 即使以下命令中的 URL 包含 `ec2-downloads-windows` 目录，这些也是 Amazon Linux 2 和 Amazon Linux 2023 的正确全局安装文件。

x86_64

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

ARM64

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. （建议）请使用以下命令验证代理是否正在运行。

```
sudo systemctl status amazon-ssm-agent
```

在大多数情况下，命令会报告代理正在运行，如下面的示例所示。

```
amazon-ssm-agent.service - amazon-ssm-agent
Loaded: loaded (/usr/lib/systemd/system/amazon-ssm-agent.service; enabled; vendor
 preset: enabled)
Active: active (running) since Wed 2021-10-20 19:09:29 UTC; 4min 6s ago
            --truncated--
```

在极少数情况下，命令会报告代理已安装但未运行，如下面的示例所示。

```
amazon-ssm-agent.service - amazon-ssm-agent
Loaded: loaded (/usr/lib/systemd/system/amazon-ssm-agent.service; enabled; vendor
 preset: enabled)
Active: inactive (dead) since Wed 2021-10-20 22:16:41 UTC; 18s ago
            --truncated--
```

要在这些情况下激活代理，请运行以下命令。

```
sudo systemctl start amazon-ssm-agent
```

为您所在区域的 Amazon Linux 2 或 Amazon Linux 2023 创建自定义代理安装命令

在使用脚本或模板在多个实例上安装 SSM Agent 时，建议使用存储在工作所在的 AWS 区域中的安装文件。

对于以下命令，我们提供了在美国东部（俄亥俄州）区域（us-east-2）使用可公开访问 S3 存储桶的示例。

> **Tip**
>
> 您还可以使用您构造的自定义区域 URL 替换本主题前面的 Amazon Linux 1 上 SSM Agent 的快速安装命令 程序中的全局 URL。

在下面的命令中，将 *region* 替换为您自己的信息。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

x86_64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

请参阅以下示例。

```
sudo yum install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_amd64/amazon-ssm-agent.rpm
```

ARM64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_arm64/amazon-ssm-agent.rpm
```

请参阅以下 示例。

```
sudo yum install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_arm64/amazon-ssm-agent.rpm
```

在 Amazon Linux 1 实例上手动安装 SSM Agent

> **Important**
>
> 正如 AWS News Blog 上 Update on Amazon Linux AMI end-of-life 一文宣布的那样，Amazon Linux 1 于 2020 年 12 月 31 日结束了标准支持，于 2023 年 12 月 31 日结束了生命周期。AWS 不再为该操作系统提供 Amazon Machine Images（AMIs）。不过，AWS Systems Manager 会继续为现有的 Amazon Linux 1 实例提供支持。
>
> 本主题提供了用于在 Amazon Linux 1 实例上使用 SSM Agent 的命令。其中一些命令不适用于 Amazon Linux 2 和 Amazon Linux 2023 实例。在继续操作之前，请验证您正在查看的主题是否与您的实例类型对应。有关要在 Amazon Linux 2 或 Amazon Linux 2023 实例上运行的命令，请参阅 在 Amazon Linux 2 和 Amazon Linux 2023 实例上手动安装 SSM Agent。

在大多数情况下，AWS 提供的适用于 Amazon Linux 1 的 Amazon Machine Images（AMIs）附带默认情况下预安装的 AWS Systems Manager 代理（SSM Agent）。有关更多信息，请参阅 查找预装了 SSM Agent 的 AMIs。

如果需要在 Amazon Linux 1 上手动重新安装代理，请参照此页面上的信息操作。

开始前的准备工作

在 Amazon Linux 1 实例上安装 SSM Agent 之前，请注意以下事项：

- 有关适用于在所有基于 Linux 的操作系统上安装 SSM Agent 的重要信息，请参阅 在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent。
- 如果您在使用 SSM 文档 `AWS-UpdateSSMAgent` 成功安装或更新代理之后，使用 `yum` 命令来更新托管节点上的 SSM Agent，系统可能会显示以下消息：`"Warning: RPMDB altered outside of yum."`。此消息是预期中的，您可以安全地将其忽略。

主题

- Amazon Linux 1 上 SSM Agent 的快速安装命令
- 为所在区域的 Amazon Linux 1 创建自定义代理安装命令

Amazon Linux 1 上 SSM Agent 的快速安装命令

使用以下步骤在单个实例上手动安装 SSM Agent。此过程使用全局可用的安装文件。

使用快速复制和粘贴命令在 Amazon Linux 1 上安装 SSM Agent

1. 使用首选方法（例如 SSH）连接到 Amazon Linux 1 实例。
2. 复制实例架构的命令并在实例上运行它。

> **Note**
>
> 即使以下命令中的 URL 包含 `ec2-downloads-windows` 目录，这些也是 Amazon Linux 1 的正确全局安装文件。

x86_64

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

x86

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_386/amazon-ssm-agent.rpm
```

ARM64

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. （推荐）为实例架构运行命令，以验证代理是否正在运行。

x86_64 和 x86

```
sudo status amazon-ssm-agent
```

ARM64

```
sudo systemctl status amazon-ssm-agent
```

在大多数情况下，命令会报告代理正在运行，如下面的示例所示。

x86_64 和 x86

```
amazon-ssm-agent start/running, process 12345
```

ARM64

```
amazon-ssm-agent.service - amazon-ssm-agent
Loaded: loaded (/usr/lib/systemd/system/amazon-ssm-agent.service; enabled;
 vendor preset: enabled)
Active: active (running) since Wed 2021-10-20 19:09:29 UTC; 4min 6s ago
            --truncated--
```

在极少数情况下，命令会报告代理已安装但未运行，如下面的示例所示。

x86_64 和 x86

```
amazon-ssm-agent stop/waiting
```

ARM64

```
amazon-ssm-agent.service - amazon-ssm-agent
Loaded: loaded (/usr/lib/systemd/system/amazon-ssm-agent.service; enabled;
 vendor preset: enabled)
Active: inactive (dead) since Wed 2021-10-20 22:16:41 UTC; 18s ago
            --truncated--
```

要在这些情况下激活代理，请运行实例架构的命令。

x86_64 和 x86

```
sudo start amazon-ssm-agent
```

ARM64

```
sudo systemctl start amazon-ssm-agent
```

为所在区域的 Amazon Linux 1 创建自定义代理安装命令

在使用脚本或模板在多个实例上安装 SSM Agent 时，建议使用存储在工作所在的 AWS 区域中的安装文件。

对于以下命令，我们提供了在美国东部（俄亥俄州）区域（us-east-2）使用可公开访问 S3 存储桶的示例。

> **Tip**
>
> 您还可以使用您构造的自定义区域 URL 替换本主题前面的 Amazon Linux 1 上 SSM Agent 的快速安装命令 程序中的全局 URL。

在下面的命令中，将 *region* 替换为您自己的信息。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

x86_64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

请参阅以下示例。

```
sudo yum install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_amd64/amazon-ssm-agent.rpm
```

x86

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_386/amazon-ssm-agent.rpm
```

请参阅以下示例。

```
sudo yum install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_386/amazon-ssm-agent.rpm
```

ARM64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_arm64/amazon-ssm-agent.rpm
```

请参阅以下 示例。

```
sudo yum install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_arm64/amazon-ssm-agent.rpm
```

在 CentOS 实例上手动安装 SSM Agent

AWS 提供的适用于 CentOS 的 Amazon Machine Images（AMIs）不附带默认情况下预安装的 AWS Systems Manager 代理（SSM Agent）。有关可能在其上预安装代理的 AWS 托管式 AMIs 的列表，请参阅 查找预装了 SSM Agent 的 AMIs。

使用本节中的信息可帮助您在 CentOS 实例上手动安装或重新安装 SSM Agent。

开始前的准备工作

在 CentOS 实例上安装 SSM Agent 之前，请注意以下事项：

- 有关适用于在所有基于 Linux 的操作系统上安装 SSM Agent 的重要信息，请参阅 在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent。
- 如果您在使用 SSM 文档 `AWS-UpdateSSMAgent` 成功安装或更新代理之后，使用 `yum` 命令来更新托管节点上的 SSM Agent，系统可能会显示以下消息：`"Warning: RPMDB altered outside of yum."`。此消息是预期中的，您可以安全地将其忽略。

主题

- 在 CentOS 8.x 上安装 SSM Agent
- 在 CentOS 7.x 上安装 SSM Agent
- 在 CentOS 6.x 上安装 SSM Agent

在 CentOS 8.x 上安装 SSM Agent

Amazon Machine Images 提供的适用于 CentOS 8 的 AMIs（AWS）不附带默认情况下预安装的 AWS Systems Manager 代理（SSM Agent）。使用本页中的信息可帮助您在 CentOS 8 实例上安装或重新安装代理。

开始前的准备工作

在 CentOS 8 实例上安装 SSM Agent 之前，请注意以下事项：

- 确保在您的 CentOS 8 实例上安装了 Python 2 或 Python 3。这是 SSM Agent 正常工作所必需的。

主题

- CentOS 8 上的 SSM Agent 的快速安装命令
- 为您所在区域的 CentOS 8 创建自定义代理安装命令

CentOS 8 上的 SSM Agent 的快速安装命令

使用以下步骤在单个实例上手动安装 SSM Agent。此过程使用全局可用的安装文件。

在 CentOS 8.x 上安装 SSM Agent

1. 使用首选方法（例如 SSH）连接到您的 CentOS 8 实例。
2. 复制实例架构的命令并在实例上运行它。

> **Note**
>
> 即使以下命令中的 URL 包含 `ec2-downloads-windows` 目录，这些也是 CentOS 8 的正确全局安装文件。

x86_64 实例

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

ARM64 实例

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. （建议）请使用以下命令验证代理是否正在运行。

```
sudo systemctl status amazon-ssm-agent
```

在大多数情况下，命令会报告代理正在运行，如下面的示例所示。

```
# amazon-ssm-agent.service - amazon-ssm-agent
   Loaded: loaded (/etc/systemd/system/amazon-ssm-agent.service; enabled; vendo>
   Active: active (running) since Tue 2022-04-19 15:48:54 UTC; 19s ago
            --truncated--
```

在极少数情况下，命令会报告代理已安装但未运行，如下面的示例所示。

```
# amazon-ssm-agent.service - amazon-ssm-agent
   Loaded: loaded (/etc/systemd/system/amazon-ssm-agent.service; disabled; vend>
   Active: inactive (dead)
            --truncated--
```

要在这些情况下激活代理，请运行以下命令。

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

为您所在区域的 CentOS 8 创建自定义代理安装命令

在使用脚本或模板在多个实例上安装 SSM Agent 时，建议使用存储在工作所在的 AWS 区域中的安装文件。

对于以下命令，我们提供了在美国东部（俄亥俄州）区域（us-east-2）使用可公开访问 S3 存储桶的示例。

> **Tip**
>
> 您还可以使用您构造的自定义区域 URL 替换本主题前面的 CentOS 8 上的 SSM Agent 的快速安装命令 程序中的全局 URL。

在下面的命令中，将 *region* 替换为您自己的信息。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

x86_64

```
sudo dnf install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

请参阅以下示例。

```
sudo dnf install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_amd64/amazon-ssm-agent.rpm
```

ARM64

```
sudo dnf install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_arm64/amazon-ssm-agent.rpm
```

请参阅以下 示例。

```
sudo dnf install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_arm64/amazon-ssm-agent.rpm
```

在 CentOS 7.x 上安装 SSM Agent

Amazon Machine Images 提供的适用于 CentOS 7 的 AMIs（AWS）不附带默认情况下预安装的 AWS Systems Manager 代理（SSM Agent）。使用本页中的信息可帮助您在 CentOS 7 实例上安装或重新安装代理。

主题

- CentOS 7 上的 SSM Agent 的快速安装命令
- 为您所在区域的 CentOS 7 创建自定义代理安装命令

CentOS 7 上的 SSM Agent 的快速安装命令

使用以下步骤在单个实例上手动安装 SSM Agent。此过程使用全局可用的安装文件。

在 CentOS 7.x 上安装 SSM Agent

1. 使用首选方法（例如 SSH）连接到您的 CentOS 7 实例。

2. 复制实例架构的命令并在实例上运行它。

> **Note**
>
> 即使以下命令中的 URL 包含 `ec2-downloads-windows` 目录，这些也是 CentOS 7 的正确全局安装文件。

x86_64 实例

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

ARM64 实例

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. （建议）请使用以下命令验证代理是否正在运行。

```
sudo systemctl status amazon-ssm-agent
```

在大多数情况下，命令会报告代理正在运行，如下面的示例所示。

```
# amazon-ssm-agent.service - amazon-ssm-agent
   Loaded: loaded (/etc/systemd/system/amazon-ssm-agent.service; enabled; vendor
 preset: disabled)
   Active: active (running) since Tue 2022-04-19 15:57:27 UTC; 6s ago
           --truncated--
```

在极少数情况下，命令会报告代理已安装但未运行，如下面的示例所示。

```
# amazon-ssm-agent.service - amazon-ssm-agent
   Loaded: loaded (/etc/systemd/system/amazon-ssm-agent.service; enabled; vendor
 preset: disabled)
   Active: inactive (dead) since Tue 2022-04-19 15:58:44 UTC; 2s ago
           --truncated--
```

要在这些情况下激活代理，请运行以下命令。

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

为您所在区域的 CentOS 7 创建自定义代理安装命令

在使用脚本或模板在多个实例上安装 SSM Agent 时，建议使用存储在工作所在的 AWS 区域中的安装文件。

对于以下命令，我们提供了在美国东部（俄亥俄州）区域（us-east-2）使用可公开访问 S3 存储桶的示例。

> **Tip**
>
> 您还可以使用您构造的自定义区域 URL 替换本主题前面的 CentOS 7 上的 SSM Agent 的快速安装命令 程序中的全局 URL。

在下面的命令中，将 *region* 替换为您自己的信息。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

x86_64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

请参阅以下示例。

```
sudo yum install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_amd64/amazon-ssm-agent.rpm
```

ARM64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_arm64/amazon-ssm-agent.rpm
```

请参阅以下 示例。

```
sudo yum install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_arm64/amazon-ssm-agent.rpm
```

在 CentOS 6.x 上安装 SSM Agent

Amazon Machine Images 提供的适用于 CentOS 6 的 AMIs（AWS）不附带默认情况下预安装的 AWS Systems Manager 代理（SSM Agent）。使用本页中的信息可帮助您在 CentOS 6 实例上安装或重新安装代理。

主题

- CentOS 6 上的 SSM Agent 的快速安装命令
- 为您所在区域的 CentOS 6 创建自定义代理安装命令

CentOS 6 上的 SSM Agent 的快速安装命令

使用以下步骤在单个实例上手动安装 SSM Agent。此过程使用全局可用的安装文件。

在 CentOS 6.x 上安装 SSM Agent

1. 使用首选方法（例如 SSH）连接到您的 CentOS 6 实例。
2. 复制实例架构的命令并在实例上运行它。

> **Note**
>
> 即使以下命令中的 URL 包含 `ec2-downloads-windows` 目录，这些也是 CentOS 6 的正确全局安装文件。
> 以下命令指定版本目录 `3.0.1479.0`，而不是 `latest` 目录。这是因为 CentOS 6 不支持 SSM Agent 版本 3.1 及更高版本。

x86_64 实例

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/
SSMAgent/3.0.1479.0/linux_amd64/amazon-ssm-agent.rpm
```

x86 实例

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/
SSMAgent/3.0.1479.0/linux_386/amazon-ssm-agent.rpm
```

3. （建议）请使用以下命令验证代理是否正在运行。

```
sudo status amazon-ssm-agent
```

在大多数情况下，命令会报告代理正在运行，如下面的示例所示。

```
amazon-ssm-agent start/running, process 1744
```

在极少数情况下，命令会报告代理已安装但未运行，如下面的示例所示。

```
amazon-ssm-agent stop/waiting
```

要在这些情况下激活代理，请运行以下命令。

```
sudo start amazon-ssm-agent
```

为您所在区域的 CentOS 6 创建自定义代理安装命令

在使用脚本或模板在多个实例上安装 SSM Agent 时，建议使用存储在工作所在的 AWS 区域中的安装文件。

对于以下命令，我们提供了在美国东部（俄亥俄州）区域（`us-east-2`）使用可公开访问 S3 存储桶的示例。

> **Tip**
>
> 您还可以使用您构造的自定义区域 URL 替换本主题前面的 CentOS 6 上的 SSM Agent 的快速安装命令 程序中的全局 URL。

在下面的命令中，将 *`region`* 替换为您自己的信息。有关支持的 *`region`* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

> **Note**
>
> 以下命令指定版本目录 `3.0.1479.0`，而不是 `latest` 目录。这是因为 CentOS 6 不支持 SSM Agent 版本 3.1 及更高版本。

x86_64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/3.0.1479.0/linux_amd64/amazon-ssm-agent.rpm
```

请参阅以下示例。

```
sudo yum install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/3.0.1479.0/linux_amd64/amazon-ssm-agent.rpm
```

x86

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/3.0.1479.0/linux_386/amazon-ssm-agent.rpm
```

请参阅以下 示例。

```
sudo yum install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/3.0.1479.0/linux_386/amazon-ssm-agent.rpm
```

在 CentOS Stream 实例上手动安装 SSM Agent

AWS 提供的适用于 CentOS Stream 的 Amazon Machine Images（AMIs）不附带默认情况下预安装的 AWS Systems Manager 代理（SSM Agent）。有关可能在其上预安装代理的 AWS 托管式 AMIs 的列表，请参阅 查找预装了 SSM Agent 的 AMIs。

使用本节中的信息可帮助您在 CentOS Stream 实例上手动安装或重新安装 SSM Agent。

开始前的准备工作

在 CentOS Stream 实例上安装 SSM Agent 之前，请注意以下事项：

- 有关适用于在所有基于 Linux 的操作系统上安装 SSM Agent 的重要信息，请参阅 在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent。

主题

- CentOS Stream 上的 SSM Agent 的快速安装命令
- 为您所在区域的 CentOS Stream 创建自定义代理安装命令

CentOS Stream 上的 SSM Agent 的快速安装命令

使用以下步骤在单个实例上手动安装 SSM Agent。此过程使用全局可用的安装文件。

开始前的准备工作

在 CentOS Stream 实例上安装 SSM Agent 之前，请注意以下事项：

- 确保在您的 CentOS Stream 8 实例上安装了 Python 2 或 Python 3。这是 SSM Agent 正常工作所必需的。

在 CentOS Stream 上安装 SSM Agent

1. 使用首选方法（例如 SSH）连接到您的 CentOS Stream 实例。
2. 复制实例架构的命令并在实例上运行它。

> **Note**
>
> 即使以下命令中的 URL 包含 `ec2-downloads-windows` 目录，这些也是 CentOS Stream 的正确全局安装文件。

x86_64 实例

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

ARM64 实例

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. （建议）请使用以下命令验证代理是否正在运行。

```
sudo systemctl status amazon-ssm-agent
```

在大多数情况下，命令会报告代理正在运行，如下面的示例所示。

```
# amazon-ssm-agent.service - amazon-ssm-agent
   Loaded: loaded (/etc/systemd/system/amazon-ssm-agent.service; enabled; vendo>
   Active: active (running) since Tue 2022-04-19 16:40:41 UTC; 9s ago
 Main PID: 4898 (amazon-ssm-agen)
    Tasks: 14 (limit: 4821)
   Memory: 34.6M
   CGroup: /system.slice/amazon-ssm-agent.service
           ##4898 /usr/bin/amazon-ssm-agent
           ##4954 /usr/bin/ssm-agent-worker
            --truncated--
```

在极少数情况下，命令会报告代理已安装但未运行，如下面的示例所示。

```
# amazon-ssm-agent.service - amazon-ssm-agent
   Loaded: loaded (/etc/systemd/system/amazon-ssm-agent.service; enabled; vendo>
   Active: inactive (dead) since Tue 2022-04-19 16:42:05 UTC; 2s ago
            --truncated--
```

要在这些情况下激活代理，请运行以下命令。

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

为您所在区域的 CentOS Stream 创建自定义代理安装命令

在使用脚本或模板在多个实例上安装 SSM Agent 时，建议使用存储在工作所在的 AWS 区域中的安装文件。

对于以下命令，我们提供了在美国东部（俄亥俄州）区域（us-east-2）使用可公开访问 S3 存储桶的示例。

> **Tip**
>
> 您还可以使用您构造的自定义区域 URL 替换本主题前面的 CentOS Stream 上的 SSM Agent 的快速安装命令 程序中的全局 URL。

在下面的命令中，将 *region* 替换为您自己的信息。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

x86_64

```
sudo dnf install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

请参阅以下示例。

```
sudo dnf install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_amd64/amazon-ssm-agent.rpm
```

ARM64

```
sudo dnf install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_arm64/amazon-ssm-agent.rpm
```

请参阅以下 示例。

```
sudo dnf install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_arm64/amazon-ssm-agent.rpm
```

在 Debian Server 实例上手动安装 SSM Agent

AWS 提供的适用于 Debian Server 的 Amazon Machine Images（AMIs）不附带默认情况下预安装的 AWS Systems Manager 代理（SSM Agent）。有关可能在其上预安装代理的 AWS 托管式 AMIs 的列表，请参阅 查找预装了 SSM Agent 的 AMIs。

使用本节中的信息可帮助您在 Debian Server 实例上手动安装或重新安装 SSM Agent。

开始前的准备工作

在 Debian Server 实例上安装 SSM Agent 之前，请注意以下事项：

- 有关适用于在所有基于 Linux 的操作系统上安装 SSM Agent 的重要信息，请参阅 在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent。

主题

- Debian Server 上的 SSM Agent 的快速安装命令
- 为您所在区域的 Debian Server 创建自定义代理安装命令

Debian Server 上的 SSM Agent 的快速安装命令

使用以下步骤在单个实例上手动安装 SSM Agent。此过程使用全局可用的安装文件。

在 Debian Server 上安装 SSM Agent

1. 使用首选方法（例如 SSH）连接到您的 Debian Server 实例。
2. 输入以下命令以在实例上创建临时目录。

```
mkdir /tmp/ssm
```

3. 运行以下命令以更改到临时目录。

```
cd /tmp/ssm
```

4. 复制实例架构的命令并在实例上运行它。

> **Note**
>
> 即使以下命令中的 URL 包含 `ec2-downloads-windows` 目录，这些也是 Debian Server 的正确全局安装文件。
> 对于 Debian Server 8，仅支持 x86_64 架构。

x86_64 实例

```
wget https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/
debian_amd64/amazon-ssm-agent.deb
```

ARM64 实例

```
wget https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/
debian_arm64/amazon-ssm-agent.deb
```

5. 运行以下命令。

```
sudo dpkg -i amazon-ssm-agent.deb
```

6. （建议）请使用以下命令验证代理是否正在运行。

```
sudo systemctl status amazon-ssm-agent
```

在大多数情况下，命令会报告代理正在运行，如下面的示例所示。

```
# amazon-ssm-agent.service - amazon-ssm-agent
   Loaded: loaded (/lib/systemd/system/amazon-ssm-agent.service; enabled; vendor
   Active: active (running) since Tue 2022-04-19 16:25:03 UTC; 4s ago
 Main PID: 628 (amazon-ssm-agen)
   CGroup: /system.slice/amazon-ssm-agent.service
           ##628 /usr/bin/amazon-ssm-agent
           ##650 /usr/bin/ssm-agent-worker
            --truncated--
```

在极少数情况下，命令会报告代理已安装但未运行，如下面的示例所示。

```
# amazon-ssm-agent.service - amazon-ssm-agent
   Loaded: loaded (/lib/systemd/system/amazon-ssm-agent.service; enabled; vendor
   Active: inactive (dead) since Tue 2022-04-19 16:26:30 UTC; 5s ago
 Main PID: 628 (code=exited, status=0/SUCCESS)
            --truncated--
```

要在这些情况下激活代理，请运行以下命令。

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

为您所在区域的 Debian Server 创建自定义代理安装命令

在使用脚本或模板在多个实例上安装 SSM Agent 时，建议使用存储在工作所在的 AWS 区域中的安装文件。

对于以下命令，我们提供了在美国东部（俄亥俄州）区域（us-east-2）使用可公开访问 S3 存储桶的示例。

> **Tip**
>
> 您还可以使用您构造的自定义区域 URL 替换本主题前面的 Debian Server 上的 SSM Agent 的快速安装命令 程序中的全局 URL。

在下面的命令中，将 *region* 替换为您自己的信息。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

> **Note**
>
> 对于 Debian Server 8，仅支持 x86_64 架构。

x86_64

```
wget https://s3.region.amazonaws.com/amazon-ssm-region/latest/debian_amd64/amazon-ssm-agent.deb
```

```
sudo dpkg -i amazon-ssm-agent.deb
```

请参阅以下示例。

```
wget https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/debian_amd64/amazon-ssm-agent.deb
```

```
sudo dpkg -i amazon-ssm-agent.deb
```

ARM64

```
wget https://s3.region.amazonaws.com/amazon-ssm-region/latest/debian_arm64/amazon-ssm-agent.deb
```

```
sudo dpkg -i amazon-ssm-agent.deb
```

请参阅以下 示例。

```
wget https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/debian_arm64/amazon-ssm-agent.deb
```

```
sudo dpkg -i amazon-ssm-agent.deb
```

在 Oracle Linux 实例上手动安装 SSM Agent

AWS 提供的适用于 Oracle Linux 的 Amazon Machine Images（AMIs）不附带默认情况下预安装的 AWS Systems Manager 代理（SSM Agent）。有关可能在其上预安装代理的 AWS 托管式 AMIs 的列表，请参阅 查找预装了 SSM Agent 的 AMIs。

使用本节中的信息可帮助您在 Oracle Linux 实例上手动安装或重新安装 SSM Agent。

开始前的准备工作

在 Oracle Linux 实例上安装 SSM Agent 之前，请注意以下事项：

- 有关适用于在所有基于 Linux 的操作系统上安装 SSM Agent 的重要信息，请参阅 在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent。
- 如果您在使用 SSM 文档 `AWS-UpdateSSMAgent` 成功安装或更新代理之后，使用 `yum` 命令来更新托管节点上的 SSM Agent，系统可能会显示以下消息：`"Warning: RPMDB altered outside of yum."`。此消息是预期中的，您可以安全地将其忽略。

主题

- Oracle Linux 上的 SSM Agent 的快速安装命令
- 为您所在区域的 Oracle Linux 创建自定义代理安装命令

Oracle Linux 上的 SSM Agent 的快速安装命令

使用以下步骤在单个实例上手动安装 SSM Agent。此过程使用全局可用的安装文件。

使用快速复制和粘贴命令在 Oracle Linux 上安装 SSM Agent

1. 使用首选方法（例如 SSH）连接到您的 Oracle Linux 实例。
2. 复制以下命令并在实例上运行该命令。

> **Note**
>
> 即使以下命令中的 URL 包含 `ec2-downloads-windows` 目录，这些也是 Oracle Linux 的正确全局安装文件。

x86_64

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

3. （建议）请使用以下命令验证代理是否正在运行。

```
sudo systemctl status amazon-ssm-agent
```

在大多数情况下，命令会报告代理正在运行，如下面的示例所示。

```
amazon-ssm-agent.service - amazon-ssm-agent
Loaded: loaded (/usr/lib/systemd/system/amazon-ssm-agent.service; enabled; vendor
 preset: enabled)
Active: active (running) since Wed 2021-10-20 19:09:29 UTC; 4min 6s ago
            --truncated--
```

在极少数情况下，命令会报告代理已安装但未运行，如下面的示例所示。

```
amazon-ssm-agent.service - amazon-ssm-agent
Loaded: loaded (/usr/lib/systemd/system/amazon-ssm-agent.service; enabled; vendor
 preset: enabled)
Active: inactive (dead) since Wed 2021-10-20 22:16:41 UTC; 18s ago
            --truncated--
```

要在这些情况下激活代理，请运行以下命令。

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

为您所在区域的 Oracle Linux 创建自定义代理安装命令

在使用脚本或模板在多个实例上安装 SSM Agent 时，建议使用存储在工作所在的 AWS 区域中的安装文件。

对于以下命令，我们提供了在美国东部（俄亥俄州）区域（us-east-2）使用可公开访问 S3 存储桶的示例。

> **Tip**
>
> 您还可以使用您构造的自定义区域 URL 替换本主题前面的 Oracle Linux 上的 SSM Agent 的快速安装命令 程序中的全局 URL。

在下面的命令中，将 *region* 替换为您自己的信息。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

x86_64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

请参阅以下 示例。

```
sudo yum install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_amd64/amazon-ssm-agent.rpm
```

在 Red Hat Enterprise Linux 实例上手动安装 SSM Agent

AWS 提供的适用于 Red Hat Enterprise Linux（RHEL）的 Amazon Machine Images（AMIs）不附带默认情况下预安装的 AWS Systems Manager 代理（SSM Agent）。有关可能在其上预安装代理的 AWS 托管式 AMIs 的列表，请参阅 查找预装了 SSM Agent 的 AMIs。

使用本节中的信息可帮助您在 RHEL 实例上手动安装或重新安装 SSM Agent。

开始前的准备工作

在 RHEL 实例上安装 SSM Agent 之前，请注意以下事项：

- 有关适用于在所有基于 Linux 的操作系统上安装 SSM Agent 的重要信息，请参阅 在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent。
- 如果您在使用 SSM 文档 `AWS-UpdateSSMAgent` 成功安装或更新代理之后，使用 `yum` 命令来更新托管节点上的 SSM Agent，系统可能会显示以下消息：`"Warning: RPMDB altered outside of yum."`。此消息是预期中的，您可以安全地将其忽略。

主题

- 在 RHEL 8.x、9.x 和 10.x 上安装 SSM Agent
- 在 RHEL 7.x 上安装 SSM Agent
- 在 RHEL 6.x 上安装 SSM Agent

在 RHEL 8.x、9.x 和 10.x 上安装 SSM Agent

AWS 提供的适用于 RHEL 8 和 9 的 Amazon Machine Images（AMIs）不附带默认情况下预安装的 AWS Systems Manager 代理（SSM Agent）。使用本页中的信息可帮助您在 RHEL 8 和 9 实例上安装或重新安装代理。

开始前的准备工作

在 RHEL 8、9 或 10 实例上安装 SSM Agent 之前，请注意以下事项：

- 确保在 RHEL 8、9 或 10 实例上安装了 Python 2 或 Python 3。这是 SSM Agent 正常工作所必需的。

主题

- RHEL 8、9 和 10 上 SSM Agent 的快速安装命令

- 为您所在区域的 RHEL 8、9 和 10 创建自定义代理安装命令

RHEL 8、9 和 10 上 SSM Agent 的快速安装命令

使用以下步骤在单个实例上手动安装 SSM Agent。此过程使用全局可用的安装文件。

在 RHEL 8.x、9.x 或 10.x 上安装 SSM Agent

1. 使用首选方法（例如 SSH）连接到 RHEL 8、9 或 10 实例。
2. 复制实例架构的命令并在实例上运行它。

> **Note**
>
> 即使以下命令中的 URL 包含 `ec2-downloads-windows` 目录，这些也是 RHEL 8、9 和 10 的正确全局安装文件。

x86_64 实例

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

ARM64 实例

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. （建议）请使用以下命令验证代理是否正在运行。

```
sudo systemctl status amazon-ssm-agent
```

在大多数情况下，命令会报告代理正在运行，如下面的示例所示。

```
# amazon-ssm-agent.service - amazon-ssm-agent
   Loaded: loaded (/etc/systemd/system/amazon-ssm-agent.service; enabled; vendo>
   Active: active (running) since Tue 2022-04-19 16:40:41 UTC; 9s ago
 Main PID: 4898 (amazon-ssm-agen)
    Tasks: 14 (limit: 4821)
   Memory: 34.6M
   CGroup: /system.slice/amazon-ssm-agent.service
```

```
            ##4898 /usr/bin/amazon-ssm-agent
            ##4954 /usr/bin/ssm-agent-worker
             --truncated--
```

在极少数情况下，命令会报告代理已安装但未运行，如下面的示例所示。

```
# amazon-ssm-agent.service - amazon-ssm-agent
   Loaded: loaded (/etc/systemd/system/amazon-ssm-agent.service; enabled; vendo>
   Active: inactive (dead) since Tue 2022-04-19 16:42:05 UTC; 2s ago
            --truncated--
```

要在这些情况下激活代理，请运行以下命令。

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

为您所在区域的 RHEL 8、9 和 10 创建自定义代理安装命令

在使用脚本或模板在多个实例上安装 SSM Agent 时，建议使用存储在工作所在的 AWS 区域中的安装文件。

对于以下命令，我们提供了在美国东部（俄亥俄州）区域（us-east-2）使用可公开访问 S3 存储桶的示例。

> **Tip**
>
> 您还可以使用您构造的自定义区域 URL 替换本主题前面的 RHEL 8、9 和 10 上 SSM Agent 的快速安装命令 程序中的全局 URL。

在下面的命令中，将 *region* 替换为您自己的信息。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

x86_64

```
sudo dnf install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

请参阅以下示例。

```
sudo dnf install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_amd64/amazon-ssm-agent.rpm
```

ARM64

```
sudo dnf install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_arm64/amazon-ssm-agent.rpm
```

请参阅以下 示例。

```
sudo dnf install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_arm64/amazon-ssm-agent.rpm
```

在 RHEL 7.x 上安装 SSM Agent

AWS 提供的适用于 RHEL 7 的 Amazon Machine Images（AMIs）不附带默认情况下预安装的 AWS Systems Manager 代理（SSM Agent）。使用本页中的信息可帮助您在 RHEL 7 实例上安装或重新安装代理。

主题

- RHEL 7 上 SSM Agent 的快速安装命令
- 为您所在区域的 RHEL 7 创建自定义代理安装命令

RHEL 7 上 SSM Agent 的快速安装命令

使用以下步骤在单个实例上手动安装 SSM Agent。此过程使用全局可用的安装文件。

在 RHEL 7.x 上安装 SSM Agent

1. 使用首选方法（例如 SSH）连接到您的 RHEL 7 实例。
2. 复制实例架构的命令并在实例上运行它。

> **Note**
>
> 即使以下命令中的 URL 包含 `ec2-downloads-windows` 目录，这些也是 RHEL 7 的正确全局安装文件。

x86_64 实例

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

ARM64 实例

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. （建议）请使用以下命令验证代理是否正在运行。

```
sudo systemctl status amazon-ssm-agent
```

在大多数情况下，命令会报告代理正在运行，如下面的示例所示。

```
# amazon-ssm-agent.service - amazon-ssm-agent
   Loaded: loaded (/etc/systemd/system/amazon-ssm-agent.service; enabled; vendor
 preset: disabled)
   Active: active (running) since Tue 2022-04-19 16:47:36 UTC; 22s ago
 Main PID: 1342 (amazon-ssm-agen)
   CGroup: /system.slice/amazon-ssm-agent.service
           ##1342 /usr/bin/amazon-ssm-agent
           ##1362 /usr/bin/ssm-agent-worker
            --truncated--
```

在极少数情况下，命令会报告代理已安装但未运行，如下面的示例所示。

```
# amazon-ssm-agent.service - amazon-ssm-agent
   Loaded: loaded (/etc/systemd/system/amazon-ssm-agent.service; enabled; vendor
 preset: disabled)
   Active: inactive (dead) since Tue 2022-04-19 16:48:56 UTC; 5s ago
  Process: 1342 ExecStart=/usr/bin/amazon-ssm-agent (code=exited, status=0/SUCCESS)
 Main PID: 1342 (code=exited, status=0/SUCCESS)
            --truncated--
```

要在这些情况下激活代理，请运行以下命令。

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

为您所在区域的 RHEL 7 创建自定义代理安装命令

在使用脚本或模板在多个实例上安装 SSM Agent 时，建议使用存储在工作所在的 AWS 区域中的安装文件。

对于以下命令，我们提供了在美国东部（俄亥俄州）区域（us-east-2）使用可公开访问 S3 存储桶的示例。

> **Tip**
>
> 您还可以使用您构造的自定义区域 URL 替换本主题前面的 RHEL 7 上 SSM Agent 的快速安装命令 程序中的全局 URL。

在下面的命令中，将 *region* 替换为您自己的信息。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

x86_64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

请参阅以下示例。

```
sudo yum install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_amd64/amazon-ssm-agent.rpm
```

ARM64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_arm64/amazon-ssm-agent.rpm
```

请参阅以下 示例。

```
sudo yum install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_arm64/amazon-ssm-agent.rpm
```

在 RHEL 6.x 上安装 SSM Agent

AWS 提供的适用于 RHEL 6 的 Amazon Machine Images（AMIs）不附带默认情况下预安装的 AWS Systems Manager 代理（SSM Agent）。使用本页中的信息可帮助您在 RHEL 6 实例上安装或重新安装代理。

主题

- RHEL 6 上 SSM Agent 的快速安装命令
- 为您所在区域的 RHEL 6 创建自定义代理安装命令

RHEL 6 上 SSM Agent 的快速安装命令

使用以下步骤在单个实例上手动安装 SSM Agent。此过程使用全局可用的安装文件。

在 RHEL 6.x 上安装 SSM Agent

1. 使用首选方法（例如 SSH）连接到您的 RHEL 6 实例。
2. 复制实例架构的命令并在实例上运行它。

> **Note**
>
> 即使以下命令中的 URL 包含 `ec2-downloads-windows` 目录，这些也是 RHEL 6 的正确全局安装文件。
> 以下命令指定版本目录 `3.0.1479.0`，而不是 `latest` 目录。这是因为 RHEL 6 不支持 SSM Agent 版本 3.1 及更高版本。

x86_64 实例

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/
SSMAgent/3.0.1479.0/linux_amd64/amazon-ssm-agent.rpm
```

x86 实例

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/
SSMAgent/3.0.1479.0/linux_386/amazon-ssm-agent.rpm
```

3. （建议）请使用以下命令验证代理是否正在运行。

```
sudo status amazon-ssm-agent
```

在大多数情况下，命令会报告代理正在运行，如下面的示例所示。

```
amazon-ssm-agent start/running, process 1788
```

在极少数情况下，命令会报告代理已安装但未运行，如下面的示例所示。

```
amazon-ssm-agent stop/waiting
```

要在这些情况下激活代理，请运行以下命令。

```
sudo start amazon-ssm-agent
```

为您所在区域的 RHEL 6 创建自定义代理安装命令

在使用脚本或模板在多个实例上安装 SSM Agent 时，建议使用存储在工作所在的 AWS 区域中的安装文件。

对于以下命令，我们提供了在美国东部（俄亥俄州）区域（`us-east-2`）使用可公开访问 S3 存储桶的示例。

> **Tip**
>
> 您还可以使用您构造的自定义区域 URL 替换本主题前面的 RHEL 6 上 SSM Agent 的快速安装命令 程序中的全局 URL。

在下面的命令中，将 *`region`* 替换为您自己的信息。有关支持的 *`region`* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

> **Note**
>
> 以下命令指定版本目录 `3.0.1479.0`，而不是 `latest` 目录。这是因为 RHEL 6 不支持 SSM Agent 版本 3.1 及更高版本。

x86_64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/3.0.1479.0/linux_amd64/amazon-ssm-agent.rpm
```

请参阅以下示例。

```
sudo yum install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/3.0.1479.0/linux_amd64/amazon-ssm-agent.rpm
```

x86

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/3.0.1479.0/linux_386/amazon-ssm-agent.rpm
```

请参阅以下 示例。

```
sudo yum install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/3.0.1479.0/linux_386/amazon-ssm-agent.rpm
```

在 Rocky Linux 实例上手动安装 SSM Agent

AWS 提供的适用于 Rocky Linux 的 Amazon Machine Images（AMIs）不附带默认情况下预安装的 AWS Systems Manager 代理（SSM Agent）。有关可能在其上预安装代理的 AWS 托管式 AMIs 的列表，请参阅 查找预装了 SSM Agent 的 AMIs。

使用本节中的信息可帮助您在 Rocky Linux 实例上手动安装或重新安装 SSM Agent。

开始前的准备工作

在 Rocky Linux 实例上安装 SSM Agent 之前，请注意以下事项：

- 有关适用于在所有基于 Linux 的操作系统上安装 SSM Agent 的重要信息，请参阅 在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent。

主题

- Rocky Linux 上的 SSM Agent 的快速安装命令
- 为您所在区域的 Rocky Linux 创建自定义代理安装命令

Rocky Linux 上的 SSM Agent 的快速安装命令

使用以下步骤在单个实例上手动安装 SSM Agent。此过程使用全局可用的安装文件。

开始前的准备工作

在 Rocky Linux 实例上安装 SSM Agent 之前，请注意以下事项：

- 确保在您的 Rocky Linux 实例上安装了 Python 2 或 Python 3。这是 SSM Agent 正常工作所必需的。

在 Rocky Linux 上安装 SSM Agent

1. 使用首选方法（例如 SSH）连接到您的 Rocky Linux 实例。
2. 复制实例架构的命令并在实例上运行它。

> **Note**
>
> 即使以下命令中的 URL 包含 `ec2-downloads-windows` 目录，这些也是 Rocky Linux 的正确全局安装文件。

x86_64 实例

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

ARM64 实例

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. （建议）请使用以下命令验证代理是否正在运行。

```
sudo systemctl status amazon-ssm-agent
```

在大多数情况下，命令会报告代理正在运行，如下面的示例所示。

```
# amazon-ssm-agent.service - amazon-ssm-agent
   Loaded: loaded (/etc/systemd/system/amazon-ssm-agent.service; enabled; vendo>
   Active: active (running) since Tue 2022-04-19 16:40:41 UTC; 9s ago
 Main PID: 4898 (amazon-ssm-agen)
    Tasks: 14 (limit: 4821)
   Memory: 34.6M
   CGroup: /system.slice/amazon-ssm-agent.service
           ##4898 /usr/bin/amazon-ssm-agent
           ##4954 /usr/bin/ssm-agent-worker
            --truncated--
```

在极少数情况下，命令会报告代理已安装但未运行，如下面的示例所示。

```
# amazon-ssm-agent.service - amazon-ssm-agent
   Loaded: loaded (/etc/systemd/system/amazon-ssm-agent.service; enabled; vendo>
   Active: inactive (dead) since Tue 2022-04-19 16:42:05 UTC; 2s ago
            --truncated--
```

要在这些情况下激活代理，请运行以下命令。

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

为您所在区域的 Rocky Linux 创建自定义代理安装命令

在使用脚本或模板在多个实例上安装 SSM Agent 时，建议使用存储在工作所在的 AWS 区域中的安装文件。

对于以下命令，我们提供了在美国东部（俄亥俄州）区域（us-east-2）使用可公开访问 S3 存储桶的示例。

> **Tip**
>
> 您还可以使用您构造的自定义区域 URL 替换本主题前面的 Rocky Linux 上的 SSM Agent 的快速安装命令 程序中的全局 URL。

在下面的命令中，将 *region* 替换为您自己的信息。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

x86_64

```
sudo dnf install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

请参阅以下示例。

```
sudo dnf install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_amd64/amazon-ssm-agent.rpm
```

ARM64

```
sudo dnf install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_arm64/amazon-ssm-agent.rpm
```

请参阅以下 示例。

```
sudo dnf install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_arm64/amazon-ssm-agent.rpm
```

在 SUSE Linux Enterprise Server 实例上手动安装 SSM Agent

在大多数情况下，AWS 提供的适用于 SUSE Linux Enterprise Server（SLES）的 Amazon Machine Images（AMIs）附带默认情况下预安装的 AWS Systems Manager 代理（SSM Agent）。有关更多信息，请参阅 查找预装了 SSM Agent 的 AMIs。

如果 SSM Agent 未预装在新的 SLES 实例上，或者如果您需要手动重新安装代理，请使用此页面上的信息来帮助您。

开始前的准备工作

在 SLES 实例上安装 SSM Agent 之前，请注意以下事项：

- 有关适用于在所有基于 Linux 的操作系统上安装 SSM Agent 的重要信息，请参阅 在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent。

主题

- SLES 上的 SSM Agent 的快速安装命令
- 为您所在区域的 SLES 创建自定义代理安装命令

SLES 上的 SSM Agent 的快速安装命令

使用以下步骤在单个实例上手动安装 SSM Agent。此过程使用全局可用的安装文件。

使用快速复制和粘贴命令在 SLES 上安装 SSM Agent

1. 使用首选方法（例如 SSH）连接到您的 SLES 实例。
2. 选择 1：使用 `zypper` 命令：
   - 运行以下命令：

     ```
     sudo zypper install amazon-ssm-agent
     ```

   - 输入 `y` 以对任何提示作出响应。

   选择 2：使用 `rpm` 命令。

   - 在实例上创建临时目录。

     ```
     mkdir /tmp/ssm
     ```

   - 更改为临时目录。

     ```
     cd /tmp/ssm
     ```

   - 运行以下命令（一次一个）下载和运行 SSM Agent安装程序。

> **Note**
>
> 即使以下命令中的 URL 包含 ec2-downloads-windows 目录，这些也是 SLES 的正确全局安装文件。

x86_64 实例：

```
wget https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/linux_amd64/
amazon-ssm-agent.rpm
```

ARM64 实例：

```
wget https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/linux_arm64/
amazon-ssm-agent.rpm
```

- 运行以下命令。

```
sudo rpm --install amazon-ssm-agent.rpm
```

- （建议）请使用以下命令验证代理是否正在运行。

```
sudo systemctl status amazon-ssm-agent
```

在大多数情况下，命令会报告代理正在运行，如下面的示例所示。

```
# amazon-ssm-agent.service - amazon-ssm-agent
 Loaded: loaded (/usr/lib/systemd/system/amazon-ssm-agent.service; enabled;
 vendor preset: disabled)
 Active: active (running) since Mon 2022-02-21 23:13:28 UTC; 7s ago
 Main PID: 2102 (amazon-ssm-agen)
 Tasks: 15 (limit: 512)
 CGroup: /system.slice/amazon-ssm-agent.service
 ##2102 /usr/sbin/amazon-ssm-agent
 ##2107 /usr/sbin/ssm-agent-worker
           --truncated--
```

在极少数情况下，命令会报告代理已安装但未运行，如下面的示例所示。

```
# amazon-ssm-agent.service - amazon-ssm-agent
   Loaded: loaded (/usr/lib/systemd/system/amazon-ssm-agent.service; disabled;
 vendor preset: disabled)
   Active: inactive (dead)
           --truncated--
```

要在这些情况下激活代理，请运行以下命令。

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

为您所在区域的 SLES 创建自定义代理安装命令

在使用脚本或模板在多个实例上安装 SSM Agent 时，建议使用存储在工作所在的 AWS 区域中的安装文件。

对于以下命令，我们提供了在美国东部（俄亥俄州）区域（us-east-2）使用可公开访问 S3 存储桶的示例。

> **Tip**
>
> 您还可以使用您构造的自定义区域 URL 替换本主题前面的 Amazon Linux 1 上 SSM Agent 的快速安装命令 程序中的全局 URL。

在下面的命令中，将 *region* 替换为您自己的信息。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

x86_64

```
wget https://s3.region.amazonaws.com/amazon-ssm-region/latest/linux_amd64/amazon-
ssm-agent.rpm
```

```
sudo rpm --install amazon-ssm-agent.rpm
```

请参阅以下示例。

```
wget https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/linux_amd64/
amazon-ssm-agent.rpm
```

```
sudo rpm --install amazon-ssm-agent.rpm
```

ARM64

```
wget https://s3.region.amazonaws.com/amazon-ssm-region/latest/linux_arm64/amazon-
ssm-agent.rpm
```

```
sudo rpm --install amazon-ssm-agent.rpm
```

请参阅以下 示例。

```
wget https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/linux_arm64/
amazon-ssm-agent.rpm
```

```
sudo rpm --install amazon-ssm-agent.rpm
```

在 Ubuntu Server 实例上手动安装 SSM Agent

> **Important**
>
> 在 64 位版 Ubuntu Server 上安装 SSM Agent 之前，请确保您使用的安装工具正确。从使用 20180627 标识的亚马逊机器映像（AMI）开始，SSM Agent 已使用 Snap 安装包预安装到版本 16.04 上。在通过以前的 AMI 创建的实例上，必须使用 deb 安装程序包安装 SSM Agent。有关更多信息，请参阅 确定要安装在 64 位 Ubuntu Server 16.04 实例上的正确 SSM Agent 版本。

在大多数情况下，AWS 提供的适用于 Ubuntu Server 的 Amazon Machine Images（AMIs）附带默认情况下预安装的 AWS Systems Manager 代理（SSM Agent）。有关更多信息，请参阅 查找预装了 SSM Agent 的 AMIs。

如果 SSM Agent 未预安装在新的 Ubuntu Server 实例上，或者如果您需要手动重新安装代理，请使用此部分中的信息来帮助您。

开始前的准备工作

在 Ubuntu Server 实例上安装 SSM Agent 之前，请注意以下事项：

- 有关适用于在所有基于 Linux 的操作系统上安装 SSM Agent 的重要信息，请参阅 在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent。

主题

- 在 Ubuntu Server 16.04 LTS 64 位 (Snap)、18.04、20.04、20.10 STR、22.04 LTS、23.10、24.04 LTS、24.0 和 25.04 上安装 SSM Agent
- 在 Ubuntu Server 16.04 和 14.04 64 位（deb）上安装 SSM Agent
- 在 Ubuntu Server 16.04 和 14.04 32 位上安装 SSM Agent
- 确定要安装在 64 位 Ubuntu Server 16.04 实例上的正确 SSM Agent 版本

在 Ubuntu Server 16.04 LTS 64 位 (Snap)、18.04、20.04、20.10 STR、22.04 LTS、23.10、24.04 LTS、24.0 和 25.04 上安装 SSM Agent

开始前的准备工作

在 Ubuntu Server 16.04 LTS 64 位 (Snap)、18.04、20.04、20.10 STR、22.04 LTS、23.10、24.04 LTS、24.0 和 25.04 上安装 SSM Agent 之前，请注意以下事项：

由 Snap 或 deb 安装程序进行的版本 16.04 安装

在 Ubuntu Server 16.04 上，使用 Snap 或 deb 安装软件包安装 SSM Agent，具体取决于 16.04 AMI 的版本。

SSM Agent 安装程序文件位置

在 Ubuntu Server 16.04 LTS 64 位 (Snap)、18.04、20.04、20.10 STR、22.04 LTS、23.10、24.04 LTS、24.0 和 25.04 上，SSM Agent 安装程序文件（包括代理二进制文件和配置文件）均存储在以下目录中：`/snap/amazon-ssm-agent/current/`。如果要对此目录中的任何配置文件进行更改，则必须将这些文件从 `/snap` 目录复制到 `/etc/amazon/ssm/` 目录。日志和二进制文件尚未更改（`/var/lib/amazon/ssm`、`/var/log/amazon/ssm`）。

使用 Snap `candidate` 通道

Snap 存储中的候选通道包含最新版本的 SSM Agent（包括所有最新的错误修复）；而不是稳定频道。要了解有关候选通道与稳定通道间差异的更多信息，请参阅风险级别（网址为 https://snapcraft.io/docs/channels）。

如果您想跟踪有关候选通道的 SSM Agent 版本信息，请在 Ubuntu Server 20.10 STR & 20.04、18.04 和 16.04 LTS 64 位实例上运行以下命令。

```
sudo snap switch --channel=candidate amazon-ssm-agent
```

建议在版本 18.04 及更高版本上使用 Snap

在 Ubuntu Server 18.04、20.04、20.10 STR、22.04 LTS、23.10、24.04 LTS、24.0 和 25.04 上，建议仅使用 Snap。另外请确保实例上只安装并运行了代理的一个实例。如果不想 SSM Agent 与 Snap 一起使用，请卸载 SSM Agent。然后按照有关在 Ubuntu Server16.04 和 14.04 64 位（deb）上安装 SSM Agent 的说明操作，将 SSM Agent 作为 debian 软件包安装。在安装之前，请确保未安装与要作为 Debian 软件包管理的软件包列表重叠的任何 Snap。

`Maximum timeout exceeded` 错误消息

由于 Snap 的已知问题，使用 `snap` 命令时您可能会看到 `Maximum timeout exceeded` 错误。如果您收到此错误，请运行以下命令（一次运行一条命令）来启动代理、停止它并检查其状态：

```
sudo systemctl start snap.amazon-ssm-agent.amazon-ssm-agent.service
```

```
sudo systemctl stop snap.amazon-ssm-agent.amazon-ssm-agent.service
```

```
sudo systemctl status snap.amazon-ssm-agent.amazon-ssm-agent.service
```

在 Ubuntu Server 16.04 LTS 64 位 (Snap)、18.04、20.04、20.10 STR、22.04 LTS、23.10、24.04 LTS、24.0 和 25.04（带有 Snap 包）上安装 SSM Agent

1. 默认情况下，SSM Agent 安装在 Ubuntu Server 16.04 LTS 64 位 (Snap)、18.04、20.04、20.10 STR、22.04 LTS、23.10、24.04 LTS、24.0 和 25.04 AMIs 上，标识符为 `20180627` 或更高版本。

   如果您需要在本地服务器上安装 SSM Agent或者需要重新安装代理，您可以使用以下脚本。您无需为下载指定 URL，因为 `snap` 命令会自动从 Snap 应用商店 https://snapcraft.io 下载代理。

   ```
   sudo snap install amazon-ssm-agent --classic
   ```

2. 运行以下命令确定 SSM Agent是否在运行。

```
sudo snap list amazon-ssm-agent
```

3. 如果上一条命令返回 `amazon-ssm-agent is stopped`、`inactive` 或 `disabled`，则运行以下命令将启动服务。

```
sudo snap start amazon-ssm-agent
```

4. 检查代理的状态。

```
sudo snap services amazon-ssm-agent
```

在 Ubuntu Server 16.04 和 14.04 64 位（deb）上安装 SSM Agent

> **Important**
>
> 在 64 位版 Ubuntu Server 上安装 SSM Agent 之前，请确保您使用的安装工具正确。从使用 20180627 标识的 Amazon Machine Image (AMI) 开始，SSM Agent 已使用 Snap 安装包预安装到版本 16.04 上。在通过以前的 AMI 创建的实例上，必须使用 deb 安装程序包安装 SSM Agent。有关更多信息，请参阅 确定要安装在 64 位 Ubuntu Server 16.04 实例上的正确 SSM Agent 版本。如果 SSM Agent 与 Snap 一起安装到实例上，并且使用 deb 安装程序软件包安装或更新 SSM Agent，则安装或 SSM Agent 操作可能失败。

在大多数情况下，AWS 提供的 Amazon Machine Images（AMIs）Ubuntu Server 16.04 附带默认情况下预安装的 AWS Systems Manager 代理（SSM Agent）。有关更多信息，请参阅 查找预装了 SSM Agent 的 AMIs。

如果 SSM Agent 未在版本 20180627 之前预安装在新的 Ubuntu Server 16.04 实例上，您在 Ubuntu Server 14.04 上进行安装或者如果您需要手动重新安装代理，请使用此页面上的信息来帮助您。

Ubuntu Server 16.04 和 14.04 64 位（deb）上的 SSM Agent 的快速安装命令

使用以下步骤在单个实例上手动安装 SSM Agent。此过程使用全局可用的安装文件。

使用快速复制和粘贴命令在 Ubuntu Server 16.04 和 14.04 64 位（deb）上安装 SSM Agent

1. 使用首选方法（例如 SSH）连接到您的 Ubuntu Server 实例。
2. 输入以下命令以在实例上创建临时目录。

```
mkdir /tmp/ssm
```

3. 更改为临时目录。

```
cd /tmp/ssm
```

4. 运行以下命令。

> **Note**
>
> 即使以下命令中的 URL 包含 `ec2-downloads-windows` 目录，这些也是 Ubuntu Server 16.04 和 14.04 64 位的正确全局安装文件。

```
wget https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/debian_amd64/
amazon-ssm-agent.deb
```

```
sudo dpkg -i amazon-ssm-agent.deb
```

5. （建议）运行以下命令之一以确定 SSM Agent 是否在运行。

   Ubuntu Server 16.04

   ```
   sudo systemctl status amazon-ssm-agent
   ```

   Ubuntu Server 14.04

   ```
   sudo status amazon-ssm-agent
   ```

   在大多数情况下，命令会报告代理正在运行。

   在极少数情况下，命令会报告代理已安装但未运行，如下面的示例所示。

6. 如果上一条命令返回 `amazon-ssm-agent is stopped`、`inactive` 或 `disabled`，则运行以下命令之一将启动服务。

   Ubuntu Server 16.04：

```
sudo systemctl enable amazon-ssm-agent
```

Ubuntu Server 14.04：

```
sudo start amazon-ssm-agent
```

为您所在区域的 Ubuntu Server 16.04 和 14.04 64 位（deb）上的 SSM Agent 创建自定义安装命令

在使用脚本或模板在多个实例上安装 SSM Agent 时，建议使用存储在工作所在的 AWS 区域中的安装文件。

对于以下命令，我们提供了在美国东部（俄亥俄州）区域（us-east-2）使用可公开访问 S3 存储桶的示例。

> **Tip**
>
> 您还可以使用您构造的自定义区域 URL 替换本主题前面的 Ubuntu Server 16.04 和 14.04 64 位（deb）上的 SSM Agent 的快速安装命令 程序中的全局 URL。

在下面的命令中，将 *region* 替换为您自己的信息。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

```
wget https://s3.region.amazonaws.com/amazon-ssm-region/latest/debian_amd64/amazon-ssm-agent.deb
```

```
sudo dpkg -i amazon-ssm-agent.deb
```

请参阅以下 示例。

```
wget https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/debian_amd64/amazon-ssm-agent.deb
```

```
sudo dpkg -i amazon-ssm-agent.deb
```

在 Ubuntu Server 16.04 和 14.04 32 位上安装 SSM Agent

在大多数情况下，AWS 提供的 Amazon Machine Images（AMIs）Ubuntu Server 16.04 附带默认情况下预安装的 AWS Systems Manager 代理（SSM Agent）。有关更多信息，请参阅 查找预装了 SSM Agent 的 AMIs。

如果 SSM Agent 未在新的 Ubuntu Server 16.04 实例上预安装，则您在 Ubuntu Server 14.04 上进行安装或者如果您需要手动重新安装代理，请使用此页面上的信息来帮助您。

Ubuntu Server 16.04 和 14.04 32 位 (deb) 上 SSM Agent 的快速安装命令

使用以下步骤在单个实例上手动安装 SSM Agent。此过程使用全局可用的安装文件。

使用快速复制和粘贴命令在 Ubuntu Server 16.04 和 14.04 32 位（deb）上安装 SSM Agent

1. 使用首选方法（例如 SSH）连接到您的 Ubuntu Server 实例。
2. 输入以下命令以在实例上创建临时目录。

```
mkdir /tmp/ssm
```

3. 更改为临时目录。

```
cd /tmp/ssm
```

4. 运行以下命令。

> **Note**
>
> 即使以下命令中的 URL 包含 `ec2-downloads-windows` 目录，这些也是 Ubuntu Server 16.04 和 14.04 32 位的正确全局安装文件。

```
wget https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/debian_386/
amazon-ssm-agent.deb
```

```
sudo dpkg -i amazon-ssm-agent.deb
```

5. （建议）运行以下命令之一以确定 SSM Agent 是否在运行。

Ubuntu Server 16.04

```
sudo systemctl status amazon-ssm-agent
```

Ubuntu Server 14.04

```
sudo status amazon-ssm-agent
```

在大多数情况下，命令会报告代理正在运行。

在极少数情况下，命令会报告代理已安装但未运行，如下面的示例所示。

6. 如果上一条命令返回 `amazon-ssm-agent is stopped`、`inactive` 或 `disabled`，则运行以下命令之一将启动服务。

Ubuntu Server 16.04：

```
sudo systemctl enable amazon-ssm-agent
```

Ubuntu Server 14.04：

```
sudo start amazon-ssm-agent
```

为您所在区域的 Ubuntu Server 16.04 和 14.04 32 位（deb）上的 SSM Agent 创建自定义安装命令

在使用脚本或模板在多个实例上安装 SSM Agent 时，建议使用存储在工作所在的 AWS 区域中的安装文件。

对于以下命令，我们提供了在美国东部（俄亥俄州）区域（`us-east-2`）使用可公开访问 S3 存储桶的示例。

> **Tip**
>
> 您还可以使用您构造的自定义区域 URL 替换本主题前面的 Ubuntu Server 16.04 和 14.04 32 位 (deb) 上 SSM Agent 的快速安装命令 程序中的全局 URL。

在下面的命令中，将 *region* 替换为您自己的信息。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

```
wget https://s3.region.amazonaws.com/amazon-ssm-region/latest/debian_386/amazon-ssm-agent.deb
```

```
sudo dpkg -i amazon-ssm-agent.deb
```

请参阅以下 示例。

```
wget https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/debian_386/amazon-ssm-agent.deb
```

```
sudo dpkg -i amazon-ssm-agent.deb
```

确定要安装在 64 位 Ubuntu Server 16.04 实例上的正确 SSM Agent 版本

> **Important**
>
> 在 64 位版 Ubuntu Server 上安装 SSM Agent 之前，请确保您使用的安装工具正确。从使用 20180627 标识的亚马逊机器映像（AMI）开始，SSM Agent 已使用 Snap 安装包预安装到版本 16.04 上。在通过以前的 AMI 创建的实例上，必须使用 deb 安装程序包安装 SSM Agent。有关更多信息，请参阅 确定要安装在 64 位 Ubuntu Server 16.04 实例上的正确 SSM Agent 版本
>
> 请注意，如果实例上已安装多个 SSM Agent（例如，一个是使用 Snap 安装的，一个是使用 deb 安装程序安装的），则代理将无法正常运行。

您可以使用以下任一方法对验证实例的源 AMI ID 创建日期。这些过程仅适用于 AWS 托管的 AMIs。

验证源 AMI ID 创建日期（控制台）

1. 通过以下网址打开 Amazon EC2 控制台：https://console.aws.amazon.com/ec2/。
2. 在左侧导航窗格中，选择 Instances (实例)。
3. 选择一个实例。
4. 在详细信息选项卡上，检查 AMI 名称字段下的值中是否存在 YYYYMMDD 标识符。例如：ubuntu/images/hvm-ssd/ubuntu-xenial-16.04-amd64-server-20180627。

验证源 AMI ID 创建日期 (AWS CLI)

- 运行以下命令。

```
aws ec2 describe-images --image-ids ami-id
```

ami-id 表示 AWS 提供的 AMI 的 ID (例如 ami-07c8bc5c1ce9598c3) 。

如果成功，该命令将返回类似以下内容的信息，您可以在其中检查 CreationDate 和 Name 字段以获取信息。

```
{
    "Images": [
        {
            "Architecture": "x86_64",
            "CreationDate": "2020-07-24T20:40:27.000Z",
            "ImageId": "ami-07c8bc5c1ce9598c3",
-- truncated --
            "ImageOwnerAlias": "amazon",
            "Name": "amzn2-ami-hvm-2.0.20200722.0-x86_64-gp2",
            "RootDeviceName": "/dev/xvda",
            "RootDeviceType": "ebs",
            "SriovNetSupport": "simple",
            "VirtualizationType": "hvm"
        }
    ]
}
```

配置 SSM Agent 以在 Linux 节点上使用代理

可通过创建覆盖配置文件并向其添加 http_proxy、https_proxy 和 no_proxy 设置将 AWS Systems Manager Agent (SSM Agent) 配置为通过 HTTP 代理进行通信。如果您安装 SSM Agent的较新或较早版本，覆盖文件还会保留代理设置。本节包含用于在 upstart 和 systemd 环境创建覆盖文件的过程。请注意，如果打算使用 Session Manager，HTTPS 代理服务器不受支持。

主题

- 配置 SSM Agent以使用代理 (upstart)
- 配置 SSM Agent以使用代理 (systemd)

配置 SSM Agent以使用代理 (upstart)

请按照以下过程为 upstart 环境创建覆盖配置文件。

将 SSM Agent 配置为使用代理 (upstart)

1. 连接到已安装 SSM Agent 的托管实例。
2. 打开 VIM 等简单编辑器，然后根据您正在使用的是 HTTP 还是 HTTPS 代理服务器，添加以下配置之一。

   对于 HTTP 代理服务器：

   ```
   env http_proxy=http://hostname:port
   env https_proxy=http://hostname:port
   env no_proxy=IP address for instance metadata services (IMDS)
   ```

   对于 HTTPS 代理服务器：

   ```
   env http_proxy=http://hostname:port
   env https_proxy=https://hostname:port
   env no_proxy=IP address for instance metadata services (IMDS)
   ```

   Important

   将 no_proxy 设置添加至文件，然后指定 IP 地址。no_proxy 的 IP 地址是 Systems Manager 的实例元数据服务（IMDS）端点。如果您未指定 no_proxy，则对 Systems Manager 的调用将使用代理服务的身份（如果已启用 IMDSv1 回退），或者对 Systems Manager 的调用失败（如果强制执行 IMDSv2）。

   - 对于 IPv4，请指定 no_proxy=169.254.169.254。
   - 对于 IPv6，请指定 no_proxy=[fd00:ec2::254]。实例元数据服务的 IPv6 地址与 IMDSv2 命令兼容。IPv6 地址只能在基于 AWS Nitro System 构建的实例上访问。有关更多信息，请参阅《Amazon EC2 用户指南》中的实例元数据服务版本 2 的工作原理。
3. 使用名称 amazon-ssm-agent.override 将文件保存在以下位置：/etc/init/
4. 使用以下命令停止和重新启动 SSM Agent。

   ```
   sudo service stop amazon-ssm-agent
   ```

```
sudo service start amazon-ssm-agent
```

> **Note**
>
> 有关在 Upstart 环境中使用 `.override` 文件的更多信息，请参阅 init: Upstart init daemon job configuration。

配置 SSM Agent以使用代理 (systemd)

使用以下过程将 SSM Agent 配置为在 `systemd` 环境中使用代理。

> **Note**
>
> 此过程中的某些步骤包含适用于 Ubuntu Server 实例（使用 Snap 安装 SSM Agent）的明确说明。

1. 连接到已安装 SSM Agent的实例。
2. 根据操作系统类型，运行以下命令之一。

   - 在已使用 Snap 安装 SSM Agent 的 Ubuntu Server 实例上：

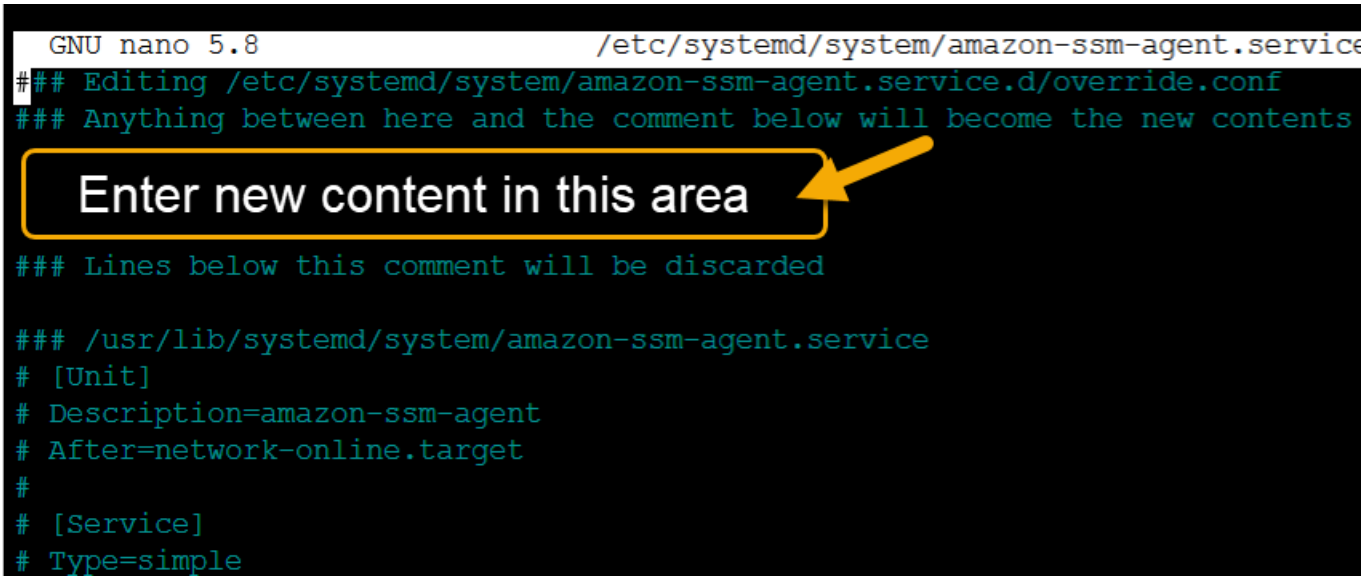
     ```
     sudo systemctl edit snap.amazon-ssm-agent.amazon-ssm-agent
     ```

     在其他操作系统上：

     ```
     sudo systemctl edit amazon-ssm-agent
     ```

3. 打开 VIM 等简单编辑器，然后根据您正在使用的是 HTTP 还是 HTTPS 代理服务器，添加以下配置之一。

   务必在显示“`### Lines below this comment will be discarded`”的注释上方输入信息，如下图所示。



对于 HTTP 代理服务器：

```
[Service]
Environment="http_proxy=http://hostname:port"
Environment="https_proxy=http://hostname:port"
Environment="no_proxy=IP address for instance metadata services (IMDS)"
```

对于 HTTPS 代理服务器：

```
[Service]
Environment="http_proxy=http://hostname:port"
Environment="https_proxy=https://hostname:port"
Environment="no_proxy=IP address for instance metadata services (IMDS)"
```

Important

将 no_proxy 设置添加至文件，然后指定 IP 地址。no_proxy 的 IP 地址是 Systems Manager 的实例元数据服务（IMDS）端点。如果您未指定 no_proxy，则对 Systems Manager 的调用将使用代理服务的身份（如果已启用 IMDSv1 回退），或者对 Systems Manager 的调用失败（如果强制执行 IMDSv2）。

- 对于 IPv4，请指定 no_proxy=169.254.169.254。

- 对于 IPv6，请指定 no_proxy=[fd00:ec2::254]。实例元数据服务的 IPv6 地址与 IMDSv2 命令兼容。IPv6 地址只能在基于 AWS Nitro System 构建的实例上访问。有关更多信息，请参阅《Amazon EC2 用户指南》中的实例元数据服务版本 2 的工作原理。

4. 保存您的更改。系统将根据操作系统类型自动创建下列文件之一。

   - 在已使用 Snap 安装 SSM Agent 的 Ubuntu Server 实例上：

     `/etc/systemd/system/snap.amazon-ssm-agent.amazon-ssm-agent.service.d/override.conf`

   - 在 Amazon Linux 2 和 Amazon Linux 2023 实例上：

     `/etc/systemd/system/amazon-ssm-agent.service.d/override.conf`

   - 在其他操作系统上：

     `/etc/systemd/system/amazon-ssm-agent.service.d/amazon-ssm-agent.override`

5. 通过使用以下命令之一重新启动 SSM Agent，具体取决于操作系统类型。

   - 在已使用 Snap 安装的 Ubuntu Server 实例上：

     ```
     sudo systemctl daemon-reload && sudo systemctl restart snap.amazon-ssm-agent.amazon-ssm-agent
     ```

   - 在其他操作系统上：

     ```
     sudo systemctl daemon-reload && sudo systemctl restart amazon-ssm-agent
     ```

> **Note**
>
> 有关在 systemd 环境中使用 .override 文件的详细信息，请参阅 Red Hat Enterprise Linux 7 系统管理员指南中的修改现有单元文件。

在适用于 macOS 的 EC2 实例上使用 SSM Agent

AWS Systems Manager (SSM Agent) 用于处理 Systems Manager 请求并按照请求中指定的方式配置计算机。按照以下过程为 macOS 安装、配置或卸载 SSM Agent。

> **Note**
>
> 默认在 Amazon Machine Images (AMIs) 上为 macOS 预先安装了 SSM Agent。无需在 Amazon Elastic Compute Cloud (Amazon EC2) 实例上为 macOS 安装 SSM Agent，除非已将其卸载。

GitHub 上提供 SSM Agent 的源代码，便于您根据需要调整代理。我们鼓励您针对要包含的更改提交提取请求。但是，AWS 不支持运行此软件的已修改副本。

> **Note**
>
> 要查看有关不同版本 SSM Agent 的详细信息，请参阅发行说明。

在 macOS 操作系统上手动安装 SSM Agent 之前，请查看以下信息。

- SSM Agent 默认安装在 EC2 实例上，并且 Systems Manager 支持 Amazon Machine Images。有关更多信息，请参阅 macOS 支持的操作系统列表。

  无需在 macOS EC2 实例上手动安装 SSM Agent，除非已将其卸载。
- 并非所有 AWS 区域 都支持 macOS 的 EC2 实例。有关支持 macOS 的基于 x86 和 M1 EC2 的实例的区域列表，请参阅 Amazon EC2 常见问题解答中的macOS 工作负载。
- 如果有新工具添加至 Systems Manager 或者对现有工具进行了更新，则将发布 SSM Agent 的更新版本。无法使用代理的最新版本可能会阻止托管式节点使用 Systems Manager 的各项工具和功能。因此，我们建议您自动完成确保机器上的 SSM Agent 为最新的过程。有关更多信息，请参阅 自动更新到 SSM Agent。要获得有关 SSM Agent 更新的通知，请在 GitHub 上订阅 SSM Agent 发布说明页面。

主题

- 在适用于 macOS 的 EC2 实例上手动安装和卸载 SSM Agent

在适用于 macOS 的 EC2 实例上手动安装和卸载 SSM Agent

连接到您的 macOS 实例，然后执行下列步骤来安装 AWS Systems Manager Agent (SSM Agent)。对将使用 Systems Manager 运行命令的每个实例执行这些步骤。此过程中提供的命令也可以通过用户数据作为脚本传递给 Amazon EC2 实例。

开始前的准备工作

使用 Homebrew 安装 `wget`。

在 macOS 上安装 SSM Agent

1. 使用以下命令下载适用于 x86_64 实例的代理安装程序文件。

   在下面的命令中，将 *region* 替换为您自己的信息。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

   ```
   sudo wget https://s3.region.amazonaws.com/amazon-ssm-region/latest/darwin_amd64/
   amazon-ssm-agent.pkg
   ```

   对于 Apple silicon 实例，使用以下命令。

   ```
   sudo wget https://s3.region.amazonaws.com/amazon-ssm-region/latest/darwin_arm64/
   amazon-ssm-agent.pkg
   ```

   见下列。

   ```
   sudo wget https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
   darwin_amd64/amazon-ssm-agent.pkg
   ```

2. 使用以下命令运行 SSM Agent 安装程序。

   x86_64:

   ```
   sudo installer -pkg amazon-ssm-agent.pkg -target /
   ```

3. 检查代理的状态。

   要确定 SSM Agent 代理是否正在运行，请检查位于以下位置的代理日志：`/var/log/amazon/ssm/amazon-ssm-agent.log`。

4. 如果代理日志指示 “amazon-ssm-agent 已停止”，则运行以下命令启动服务。

```
sudo launchctl load -w /Library/LaunchDaemons/com.amazon.aws.ssm.plist && sudo
 launchctl start com.amazon.aws.ssm
```

> **Important**
>
> 如果有新工具添加至 Systems Manager 或者对现有工具进行了更新，则将发布 SSM Agent 的更新版本。无法使用代理的最新版本可能会阻止托管式节点使用 Systems Manager 的各项工具和功能。因此，我们建议您自动完成确保机器上的 SSM Agent 为最新的过程。有关更多信息，请参阅 自动更新到 SSM Agent。要获得有关 SSM Agent 更新的通知，请在 GitHub 上订阅 SSM Agent 发布说明页面。

从 macOS 实例卸载 SSM Agent

macOS 本地不支持卸载 PKG 文件。要从 macOS 的 Amazon Elastic Compute Cloud (Amazon EC2) 实例卸载 AWS Systems Manager Agent (SSM Agent)，可以使用以下位置中的 AWS 托管脚本。

https://github.com/aws/amazon-ssm-agent/blob/mainline/Tools/src/update/darwin/uninstall.sh

在适用于 Windows Server 的 EC2 实例上使用 SSM Agent

默认情况下，AWS Systems Manager 代理（SSM Agent）已预安装在由 AWS 提供的 Windows Server 的 Amazon Machine Images（AMIs）上。支持以下操作系统（OS）版本。

- 2016 年 11 月或以后发布的 Windows Server 2008-2012 R2 AMIs
- Windows Server 2016、2019、2022（不包括 Nano 版本）和 2025

先前版本的支持说明

2016 年 11 月之前发布的 Windows Server AMIs 使用 EC2Config 服务处理请求并配置实例。

除非您出于特定原因需要使用 EC2Config 服务或早期版本的 SSM Agent 来处理 Systems Manager 请求，否则建议您在混合和多云环境中为 Systems Manager 配置的每个 Amazon Elastic Compute Cloud（Amazon EC2）实例或非 EC2 计算机上下载并安装最新版本的 SSM Agent。

从 2020 年 1 月 14 日开始，Microsoft 不再为 Windows Server 2008 的功能或安全性更新提供支持。原有 Amazon Machine Images (AMIs) for Windows Server 2008 和 2008 R2 仍包含预安装的 SSM

Agent 2 版，但 Systems Manager 不再正式支持 2008 版，并且不再针对更新这些 Windows Server 版本更新代理。此外，SSM Agent 版本 3 可能不兼容 Windows Server 2008 和 2008 R2 上的所有操作。适用于 Windows Server 2008 版的最后正式支持的 SSM Agent 版本为 2.3.1644.0。

保持 SSM Agent 的最新状态

如果有新工具添加至 Systems Manager 或者对现有工具进行了更新，则将发布 SSM Agent 的更新版本。无法使用代理的最新版本可能会阻止托管式节点使用 Systems Manager 的各项工具和功能。因此，我们建议您自动完成确保机器上的 SSM Agent 为最新的过程。有关更多信息，请参阅 自动更新到 SSM Agent。要获得有关 SSM Agent 更新的通知，请在 GitHub 上订阅 SSM Agent 发布说明页面。

要查看有关不同版本 SSM Agent 的详细信息，请参阅发行说明。

主题

- 在适用于 Windows Server 的 EC2 实例上手动安装和卸载 SSM Agent
- 配置 SSM Agent以使用 Windows Server 实例的代理

在适用于 Windows Server 的 EC2 实例上手动安装和卸载 SSM Agent

默认情况下，AWS Systems Manager 代理（SSM Agent）已预安装在由 Amazon 提供的以下 Windows Server 的 Amazon Machine Images（AMIs）上：

- 2016 年 11 月或以后发布的 Windows Server 2008-2012 R2 AMIs
- Windows Server 2016、2019、2022（不包括 Nano 版本）和 2025

在适用于 Windows Server 的 EC2 实例上安装 SSM Agent

如有必要，可以使用以下过程在 Amazon Elastic Compute Cloud (Amazon EC2) 实例 Windows Server 上手动下载并安装最新版本的 SSM Agent。此过程中提供的命令也可以通过用户数据作为脚本传递给 Amazon EC2 实例。

SSM Agent 需要使用 Windows PowerShell 3.0 或更高版本才能在 Windows Server 实例上运行 AWS Systems Manager 某些 SSM 文档（例如，原有 `AWS-ApplyPatchBaseline` 文档）。验证您的 Windows Server 实例是否在 Windows Management Framework 3.0 或更高版本上运行。此框架包括 Windows PowerShell。有关更多信息，请参阅 Windows 管理框架 3.0。

> **Note**
>
> 此过程适用于在 Windows Server 的 EC2 实例上安装或重新安装 SSM Agent。如果需要在本地服务器或虚拟机（VM）上安装代理，以与 Systems Manager 配合使用，请参阅如何在混合 Windows 节点上安装 SSM Agent。

在 Windows Server 的 EC2 实例上手动安装最新版本的 SSM Agent

1. 使用远程桌面或 Windows PowerShell 连接到您的实例。有关详细信息，请参阅《Amazon EC2 用户指南》中的连接到您的实例。
2. 将最新版本的 SSM Agent下载到您的实例。您可以使用 PowerShell 命令或直接下载链接来进行下载。

> **Note**
>
> 本步骤中的 URL 用于从任何 AWS 区域 下载 SSM Agent 如果您要从特定区域下载代理，请改为使用特定于区域的 URL：
> `https://amazon-ssm-region.s3.region.amazonaws.com/latest/windows_amd64/AmazonSSMAgentSetup.exe`
> *region* 表示 AWS Systems Manager 支持的 AWS 区域 的标识符，例如 `us-east-2` 对应美国东部（俄亥俄）区域。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

PowerShell

按顺序运行以下三条 PowerShell 命令。可通过这些命令下载 SSM Agent，无需调整 Internet Explorer (IE) 增强的安全设置，然后安装代理并移除安装文件。

64-bit

```
[System.Net.ServicePointManager]::SecurityProtocol = 'TLS12'
$progressPreference = 'silentlyContinue'
Invoke-WebRequest `
    https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/windows_amd64/AmazonSSMAgentSetup.exe `
    -OutFile $env:USERPROFILE\Desktop\SSMAgent_latest.exe
```

32-bit

```
[System.Net.ServicePointManager]::SecurityProtocol = 'TLS12'
$progressPreference = 'silentlyContinue'
Invoke-WebRequest `
    https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/
windows_386/AmazonSSMAgentSetup.exe `
    -OutFile $env:USERPROFILE\Desktop\SSMAgent_latest.exe
```

```
Start-Process `
    -FilePath $env:USERPROFILE\Desktop\SSMAgent_latest.exe `
    -ArgumentList "/S" `
    -Wait
```

```
rm -Force $env:USERPROFILE\Desktop\SSMAgent_latest.exe
```

直接下载

使用以下链接将最新版本的 SSM Agent 下载到您的实例。如果需要，请使用特定于 AWS 区域的 URL 更新此 URL。

https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/windows_amd64/AmazonSSMAgentSetup.exe

运行下载的 `AmazonSSMAgentSetup.exe` 文件以安装 SSM Agent。

3. 在 PowerShell 中发送以下命令，以启动或重新启动 SSM Agent：

```
Restart-Service AmazonSSMAgent
```

> **Note**
>
> 要从 Windows Server 实例卸载 SSM Agent，请打开控制面板、程序。选择 Uninstall a program（卸载程序）选项。打开 Amazon SSM Agent 的上下文（右键单击）菜单，选择 Uninstall（卸载）。

配置 SSM Agent以使用 Windows Server 实例的代理

本主题中的信息适用于在 2016 年 11 月或之后创建但不使用 Nano 安装选项的 Windows Server 实例。请注意，如果打算使用 Session Manager，HTTPS 代理服务器不受支持。

> **Note**
>
> 从 2020 年 1 月 14 日开始，Microsoft 不再为 Windows Server 2008 的功能或安全性更新提供支持。原有 Amazon Machine Images (AMIs) for Windows Server 2008 和 2008 R2 仍包含预安装的 SSM Agent 2 版，但 Systems Manager 不再正式支持 2008 版，并且不再针对更新这些 Windows Server 版本更新代理。此外，SSM Agent 版本 3 可能不兼容 Windows Server 2008 和 2008 R2 上的所有操作。适用于 Windows Server 2008 版的最后正式支持的 SSM Agent 版本为 2.3.1644.0。

开始前的准备工作

在将 SSM Agent 配置为使用代理之前，请注意以下重要信息。

在以下过程中，您将运行命令，以将 SSM Agent 配置为使用代理。该命令包含带有 IP 地址的 `no_proxy` 设置。该 IP 地址是 Systems Manager 的实例元数据服务（IMDS）端点。如果您未指定 `no_proxy`，则对 Systems Manager 的调用将使用代理服务的身份（如果已启用 IMDSv1 回退），或者对 Systems Manager 的调用失败（如果强制执行 IMDSv2）。

- 对于 IPv4，请指定 `no_proxy=169.254.169.254`。
- 对于 IPv6，请指定 `no_proxy=[fd00:ec2::254]`。实例元数据服务的 IPv6 地址与 IMDSv2 命令兼容。IPv6 地址只能在基于 AWS Nitro System 构建的实例上访问。有关更多信息，请参阅《Amazon EC2 用户指南》中的实例元数据服务版本 2 的工作原理。

配置 SSM Agent以使用代理

1. 使用远程桌面或 Windows PowerShell，连接到您希望配置的实例，以使用代理。
2. 在 PowerShell 中运行以下命令块。将 *hostname* 和 *port* 替换为有关代理的信息。

```
$serviceKey = "HKLM:\SYSTEM\CurrentControlSet\Services\AmazonSSMAgent"
$keyInfo = (Get-Item -Path $serviceKey).GetValue("Environment")
$proxyVariables = @("http_proxy=hostname:port", "https_proxy=hostname:port",
 "no_proxy=IP address for instance metadata services (IMDS)")
```

```
if ($keyInfo -eq $null) {
    New-ItemProperty -Path $serviceKey -Name Environment -Value $proxyVariables -
PropertyType MultiString -Force
} else {
    Set-ItemProperty -Path $serviceKey -Name Environment -Value $proxyVariables
}

Restart-Service AmazonSSMAgent
```

运行上述命令后，您可以查看 SSM Agent 日志以确认已应用代理设置。日志中的条目类似于以下内容。有关 SSM Agent 日志的更多信息，请参阅 查看 SSM Agent 日志。

```
2020-02-24 15:31:54 INFO Getting IE proxy configuration for current user: The operation
 completed successfully.
2020-02-24 15:31:54 INFO Getting WinHTTP proxy default configuration: The operation
 completed successfully.
2020-02-24 15:31:54 INFO Proxy environment variables:
2020-02-24 15:31:54 INFO http_proxy: hostname:port
2020-02-24 15:31:54 INFO https_proxy: hostname:port
2020-02-24 15:31:54 INFO no_proxy: IP address for instance metadata services (IMDS)
2020-02-24 15:31:54 INFO Starting Agent: amazon-ssm-agent - v2.3.871.0
2020-02-24 15:31:54 INFO OS: windows, Arch: amd64
```

重置 SSM Agent的代理配置

1. 使用远程桌面或 Windows PowerShell，连接到要配置的实例。
2. 如果使用远程桌面连接，则以管理员的身份启动 PowerShell。
3. 在 PowerShell 中运行以下命令块。

```
Remove-ItemProperty -Path HKLM:\SYSTEM\CurrentControlSet\Services\AmazonSSMAgent -
Name Environment
Restart-Service AmazonSSMAgent
```

SSM Agent代理设置优先级

在 Windows Server 实例上为 SSM Agent配置代理设置时，务必了解在启动 SSM Agent 时会评估这些设置并将它们应用于代理配置。您为 Windows Server 实例配置代理设置的方式可以确定其他设置是否可以取代您的预期设置。座席使用其找到的第一个代理设置。

> **Important**
>
> SSM Agent 使用 HTTPS 协议进行通信。因此，您必须使用下面的一个设置选项来配置 HTTPS proxy 参数。

SSM Agent 代理设置的评估顺序如下。

1. AmazonSSMAgent 注册表设置 (HKLM:\SYSTEM\CurrentControlSet\Services\AmazonSSMAgent)
2. System 环境变量（http_proxy、https_proxy、no_proxy）
3. LocalSystem 用户帐户环境变量（http_proxy、https_proxy、no_proxy）
4. 浏览器设置（HTTP、secure、exceptions）
5. WinHTTP 代理设置（http=、https=、bypass-list=）

SSM Agent 代理设置和 Systems Manager 服务

如果已将 SSM Agent 配置为使用代理，并且正在使用 AWS Systems Manager 工具（例如 Run Command 和Patch Manager），而这些工具在 Windows Server 实例上执行期间将使用 PowerShell 或 Windows Update 客户端，请配置其他代理设置。否则，操作可能会失败，因为 PowerShell 和 Windows Update 客户端使用的代理设置不是从 SSM Agent 代理配置继承的。

对于 Run Command，在 Windows Server 实例上配置 WinINet 代理设置。将按会话提供 [System.Net.WebRequest] 命令。要将这些配置应用于在 Run Command 中运行的后续网络命令，这些命令必须位于同一 aws:runPowershellScript 插件输入中其他 Powershell 命令之前。

以下 PowerShell 命令返回当前 WinINet 代理设置并将您的代理设置应用于 WinINet。

```
[System.Net.WebRequest]::DefaultWebProxy

$proxyServer = "http://hostname:port"
$proxyBypass = "169.254.169.254"
$WebProxy = New-Object System.Net.WebProxy($proxyServer,$true,$proxyBypass)

[System.Net.WebRequest]::DefaultWebProxy = $WebProxy
```

对于 Patch Manager，配置系统范围的代理设置，以便 Windows Update 客户端能够扫描和下载更新。建议您使用 Run Command 运行以下命令，因为这些命令会在系统账户上运行，并且这些设置会应用于整个系统范围。以下 `netsh` 命令返回当前代理设置，并将您的代理设置应用于本地系统。

```
netsh winhttp show proxy

netsh winhttp set proxy proxy-server="hostname:port" bypass-list="169.254.169.254"
```

有关使用 Run Command 的更多信息，请参阅AWS Systems Manager Run Command。

正在检查 SSM Agent 状态并启动代理

本主题列出了用于检查 AWS Systems Manager Agent (SSM Agent) 是否正在每个受支持的操作系统上运行的命令。它还提供了用于启动代理（如果未在运行）的命令。

操作系统	用于检查 SSM Agent 状态的命令	用于启动 SSM Agent 的命令
Amazon Linux 1	`sudo status amazon-ssm-agent`	`sudo start amazon-ssm-agent`
Amazon Linux 2、Amazon Linux 2022 和 Amazon Linux 2023	`sudo systemctl status amazon-ssm-agent`	`sudo systemctl enable amazon-ssm-agent` `sudo systemctl start amazon-ssm-agent`
CentOS 6.x	`sudo status amazon-ssm-agent`	`sudo start amazon-ssm-agent`
CentOS 7.x 和 CentOS 8.x	`sudo systemctl status amazon-ssm-agent`	`sudo systemctl enable amazon-ssm-agent` `sudo systemctl start amazon-ssm-agent`
Debian Server 8、9 和 10	`sudo systemctl status amazon-ssm-agent`	`sudo systemctl enable amazon-ssm-agent`

操作系统	用于检查 SSM Agent 状态的命令	用于启动 SSM Agent 的命令
		`sudo systemctl start amazon-ssm-agent`
macOS	检查位于 `/var/log/amazon/ssm/amazon-ssm-agent.log` 中的代理日志文件	`sudo launchctl load -w /Library/LaunchDaemons/com.amazon.aws.ssm.plist` `sudo launchctl start com.amazon.aws.ssm`
Oracle Linux	`sudo systemctl status amazon-ssm-agent`	`sudo systemctl enable amazon-ssm-agent` `sudo systemctl start amazon-ssm-agent`
Red Hat Enterprise Linux(RHEL) 6.x	`sudo status amazon-ssm-agent`	`sudo start amazon-ssm-agent`
Red Hat Enterprise Linux (RHEL) 7.x、8.x、9.x 和 10.x	`sudo systemctl status amazon-ssm-agent`	`sudo systemctl enable amazon-ssm-agent` `sudo systemctl start amazon-ssm-agent`
SUSE Linux Enterprise Server (SLES)	`sudo systemctl status amazon-ssm-agent`	`sudo systemctl enable amazon-ssm-agent` `sudo systemctl start amazon-ssm-agent`
Ubuntu Server 14.04（全部）和 16.04（32 位）	`sudo status amazon-ssm-agent`	`sudo start amazon-ssm-agent`

操作系统	用于检查 SSM Agent 状态的命令	用于启动 SSM Agent 的命令

Ubuntu Server 16.04 64 位实例（Deb 程序包安装）	`sudo systemctl status amazon-ssm-agent`	`sudo systemctl enable amazon-ssm-agent`
		
		`sudo systemctl start amazon-ssm-agent`

Ubuntu Server 16.04 LTS、18.04、20.04、20.10 STR、22.04 LTS、23.10 、24.04 LTS、24.0 和 25.04	`sudo systemctl status snap.amazon-ssm-agent.amazon-ssm-agent.service`	`sudo snap start amazon-ssm-agent`
Windows Server	在 PowerShell 中运行： `Get-Service AmazonSSMAgent`	在 PowerShell 管理员模式下运行： `Start-Service AmazonSSMAgent`

更多信息

- 在适用于 Linux 的 EC2 实例上使用 SSM Agent
- 在适用于 Windows Server 的 EC2 实例上使用 SSM Agent
- 正在检查 SSM Agent 版本号

正在检查 SSM Agent 版本号

某些 AWS Systems Manager 功能具有先决条件，包括托管式节点上所安装的最低 Systems Manager Agent (SSM Agent) 版本。您可以使用 Systems Manager 控制台或登录托管式节点，获取托管式节点上当前安装的 SSM Agent 版本。

> **Note**
>
> 要查看有关早期版本的详细信息，请参阅 GitHub 上的 SSM Agent 发行说明。

以下过程介绍如何在托管式节点上获取当前安装的 SSM Agent 版本。

检查在托管式节点上安装的 SSM Agent 的版本号

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Fleet Manager。
3. 在 SSM Agent 版本列中，记住代理版本编号。

从操作系统中获取当前安装的 SSM Agent 版本

选择以下选项卡之一获取操作系统中当前安装的 SSM Agent 版本。

Amazon Linux 1, Amazon Linux 2, and Amazon Linux 2023

> **Note**
>
> 此命令因操作系统的软件包管理器而异。

1. 登录您的托管式节点。
2. 运行以下命令。

```
yum info amazon-ssm-agent
```

该命令会返回类似以下内容的输出。

```
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Installed Packages
Name        : amazon-ssm-agent
Arch        : x86_64
Version     : 3.0.655.0
```

CentOS

1. 登录您的托管式节点。
2. 对于 CentOS 6 和 7，运行以下命令。

```
yum info amazon-ssm-agent
```

该命令会返回类似以下内容的输出。

```
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Installed Packages
Name        : amazon-ssm-agent
Arch        : x86_64
Version     : 3.0.655.0
```

Debian 服务器

1. 登录您的托管式节点。
2. 运行以下命令。

```
apt list amazon-ssm-agent
```

该命令会返回类似以下内容的输出。

```
apt list amazon-ssm-agent
Listing... Done
amazon-ssm-agent/now 3.0.655.0-1 amd64 [installed,local]

3.0.655.0 is the version of SSM agent
```

macOS

1. 登录您的托管式节点。
2. 运行以下命令。

```
pkgutil --pkg-info com.amazon.aws.ssm
```

RHEL

1. 登录您的托管式节点。
2. 对于 RHEL 6、7、8 和 9，运行以下命令。

```
yum info amazon-ssm-agent
```

该命令会返回类似以下内容的输出。

```
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Installed Packages
Name        : amazon-ssm-agent
Arch        : x86_64
Version     : 3.0.655.0
```

对于 DNF 程序包实用工具，运行以下命令。

```
dnf info amazon-ssm-agent
```

SLES

1. 登录您的托管式节点。
2. 对于 SLES 12 和 15，运行以下命令。

```
zypper info amazon-ssm-agent
```

该命令会返回类似以下内容的输出。

```
Loading repository data...
Reading installed packages...
Information for package amazon-ssm-agent:
----------------------------------------
Repository : @System
Name : amazon-ssm-agent
Version : 3.0.655.0-1
```

Ubuntu Server

> **Note**
>
> 要检查您的 Ubuntu Server 16.04 实例是使用 Deb 软件包还是 Snap 软件包，请参阅 在 Ubuntu Server 实例上手动安装 SSM Agent。

1. 登录您的托管式节点。
2. 对于 Ubuntu Server 16.04 和 14.04 64 位（使用 Deb 安装程序包），运行以下命令。

```
apt list amazon-ssm-agent
```

该命令会返回类似以下内容的输出。

```
apt list amazon-ssm-agent
Listing... Done
amazon-ssm-agent/now 3.0.655.0-1 amd64 [installed,local]

3.0.655.0 is the version of SSM agent
```

对于 Ubuntu Server 16.04 LTS 64 位 (Snap)、18.04、20.04、20.10 STR、22.04 LTS、23.10、24.04 LTS、24.0 和 25.04，运行以下命令。

```
sudo snap list amazon-ssm-agent
```

该命令会返回类似以下内容的输出。

```
snap list amazon-ssm-agent
Name Version Rev Tracking Publisher Notes
amazon-ssm-agent 3.0.529.0 3552 latest/stable/… aws# classic-

3.0.529.0 is the version of SSM agent
```

Windows

1. 登录您的托管式节点。
2. 运行以下 PowerShell 命令。

```
& "C:\Program Files\Amazon\SSM\amazon-ssm-agent.exe" -version
```

该命令会返回类似以下内容的输出。

```
SSM Agent version: 3.1.804.0
```

我们建议您使用最新版本的 SSM Agent，以便您可以受益于新的或更新的功能。为确保您的托管实例始终运行最新版本的 SSM Agent，您可以自动处理更新 SSM Agent 的过程。有关更多信息，请参阅自动更新到 SSM Agent。

查看 SSM Agent 日志

AWS Systems Manager Agent (SSM Agent) 将有关执行、命令、计划操作、错误和运行状况的信息写入每个托管式节点上的日志文件中。您可以通过手动连接到托管式节点来查看日志文件，也可以将日志自动发送到 Amazon CloudWatch Logs。有关将事件发送到 CloudWatch Logs 的更多信息，请参阅 AWS Systems Manager 中的日志记录和监控。

您可以在以下位置中查看托管式节点上的 SSM Agent 日志。

Linux and macOS

```
/var/log/amazon/ssm/
```

Windows

```
%PROGRAMDATA%\Amazon\SSM\Logs\
```

对于 Linux 托管式节点，SSM Agent `stderr` 和 `stdout` 文件将写入以下目录：`/var/lib/amazon/ssm/`。

对于 Windows 托管式节点，SSM Agent `stderr` 和 `stdout` 文件将写入以下目录：`%PROGRAMDATA%\Amazon\SSM\InstanceData\`。

有关启用 SSM Agent 调试日志记录的信息，请参阅 允许 SSM Agent 调试日志记录。

有关 `cihub/seelog` 配置的更多信息，请参阅 GitHub 上的 Seelog Wiki。有关 `cihub/seelog` 配置的示例，请参阅 GitHub 上的 cihub/seelog examples 存储库。

允许 SSM Agent 调试日志记录

按照以下过程在托管式节点上允许 SSM Agent 调试日志记录。

Linux and macOS

在 Linux 和 macOS 托管式节点上允许 SSM Agent 调试日志记录

1. 使用Session Manager（AWS Systems Manager 中的一项工具）连接到要启用调试日志记录的托管式节点，或登录托管式节点。有关更多信息，请参阅 使用 Session Manager。
2. 找到 seelog.xml.template 文件。

   Linux：

   在大多数 Linux 托管式节点类型上，该文件位于 `/etc/amazon/ssm/seelog.xml.template` 目录中。

   在 Ubuntu Server 20.10 STR & 20.04、18.04 和 16.04 LTS 上，该文件位于目录 `/snap/amazon-ssm-agent/current/seelog.xml.template` 中。将此文件从 `/snap/amazon-ssm-agent/current/` 目录复制到 `/etc/amazon/ssm/` 目录，然后再进行任何更改。

   macOS:

   在 macOS 实例类型上，该文件位于 `/opt/aws/ssm/seelog.xml.template` 目录中。
3. 将文件名从 `seelog.xml.template` 更改为 `seelog.xml`。

> **Note**
>
> 在 Ubuntu Server 20.10 STR 和 20.04、18.04 及 16.04 LTS 上，必须在目录 `/etc/amazon/ssm/` 中创建文件 `seelog.xml`。可以通过运行以下命令来创建此目录和文件。
>
> ```
> sudo mkdir -p /etc/amazon/ssm
> ```
>
> ```
> sudo cp -p /snap/amazon-ssm-agent/current/seelog.xml.template /etc/amazon/ssm/seelog.xml
> ```

4. 编辑 seelog.xml 文件以更改默认日志记录行为。将 minlevel 的值从 info 更改为 debug，如下面的示例所示。

   ```
   <seelog type="adaptive" mininterval="2000000" maxinterval="100000000"
   critmsgcount="500" minlevel="debug">
   ```

5. （可选）使用以下命令重新启动 SSM Agent。

   Linux：

   ```
   sudo service amazon-ssm-agent restart
   ```

   macOS:

   ```
   sudo /opt/aws/ssm/bin/amazon-ssm-agent restart
   ```

Windows

在 Windows Server 托管式节点上允许 SSM Agent 调试日志记录

1. 使用 Session Manager 连接到要启用调试日志记录的托管式节点，或登录托管式节点。有关更多信息，请参阅 使用 Session Manager。
2. 生成 seelog.xml.template 文件的副本。将副本名称更改为 seelog.xml。该文件位于以下目录中。

   ```
   %PROGRAMFILES%\Amazon\SSM\seelog.xml.template
   ```

3. 编辑 seelog.xml 文件以更改默认日志记录行为。将 minlevel 的值从 info 更改为 debug，如下面的示例所示。

   ```
   <seelog type="adaptive" mininterval="2000000" maxinterval="100000000"
   critmsgcount="500" minlevel="debug">
   ```

4. 找到以下条目。

   ```
   filename="{{LOCALAPPDATA}}\Amazon\SSM\Logs\{{EXECUTABLENAME}}.log"
   ```

   将此条目更改为使用以下路径。

   ```
   filename="C:\ProgramData\Amazon\SSM\Logs\{{EXECUTABLENAME}}.log"
   ```

5. 找到以下条目。

```
filename="{{LOCALAPPDATA}}\Amazon\SSM\Logs\errors.log"
```

将此条目更改为使用以下路径。

```
filename="C:\ProgramData\Amazon\SSM\Logs\errors.log"
```

6. 在管理员模式下使用以下 PowerShell 命令重新启动 SSM Agent。

```
Restart-Service AmazonSSMAgent
```

通过 SSM Agent 限制对根级别命令的访问

AWS Systems Manager 代理（SSM Agent）使用 root 权限（Linux）或 SYSTEM 权限（Windows Server）在混合和多云环境中的 Amazon Elastic Compute Cloud（Amazon EC2）实例以及其他计算机类型上运行。由于存在最高级别的系统访问权限，有权向 SSM Agent 发送命令的任何可信实例均具有根或系统权限。（在 AWS 中，可以在 AWS 中执行操作并访问资源的可信实体称为主体。主体可以是 AWS 账户根用户、用户或角色。）

主体需要此访问级别才能向 SSM Agent 发送授权的 Systems Manager 命令，主体还可以利用 SSM Agent 中的任何潜在漏洞来运行恶意代码。

特别是，应小心地限制运行命令 SendCommand 和 StartSession 的权限。第一步应为每个命令授予仅选择组织中的委托人的权限。但是，建议通过限制委托人可在其上运行这些命令的托管式节点来进一步巩固安保状况。可以在分配给主体的 IAM policy 中执行此操作。在 IAM policy 中，您可以包括一个条件，用于将用户限制为仅在使用特定标签或标签组合进行标记的托管式节点上运行命令。

例如，假设您有两队服务器，一个用于测试，一个用于生产。在应用于初级工程师的 IAM policy 中，可以指定他们只能在使用 `ssm:resourceTag/testServer` 标记的实例上运行命令。但对于应有权访问所有实例的一小群首席工程师，可授予对使用 `ssm:resourceTag/testServer` 和 `ssm:resourceTag/productionServer` 标记的实例的访问权限。

使用此方法，如果初级工程师尝试在生产实例上运行命令，则其访问将遭拒，因为其分配的 IAM policy 不提供对使用 `ssm:resourceTag/productionServer` 标记的实例的显式访问权限。

有关更多信息及示例，请参阅以下主题：

- 根据标签限制 Run Command 访问
- 基于实例标签限制会话访问

自动更新到 SSM Agent

在添加或更新 Systems Manager 工具时，AWS 会发布一个新版本的 AWS Systems Manager Agent（SSM Agent）。如果托管式节点使用旧版本的代理，则您无法使用全新的工具或从更新后的工具中受益。出于这些原因，我们建议您使用以下任意一种方法自动完成托管式节点上的 SSM Agent 更新过程。

Bottlerocket 操作系统上的代理更新

Bottlerocket 操作系统上的 SSM Agent 无法使用 Systems Manager 命令文档 `AWS-UpdateSSMAgent` 进行更新。更新在 Bottlerocket 控制容器中托管。有关更多信息，请参阅 GitHub 上的 Bottlerocket 控制容器 和 Bottlerocket 更新操作员。

macOS 版本要求

如果实例运行 macOS 版本 11.0（Big Sur）或更高版本，则该实例必须具有 SSM Agent 版本 3.1.941.0 或更高版本才能运行 AWS-UpdateSSMAgent 文档。如果实例运行 3.1.941.0 之前发布的 SSM Agent 版本，则可以通过运行 `brew update` 和 `brew upgrade amazon-ssm-agent` 命令来更新 SSM Agent，从而运行 AWS-UpdateSSMAgent。

方法	详细信息
所有托管式节点上的一键式自动更新（建议）	可以将您的 AWS 账户 中的所有托管式节点配置为自动检查并下载新版本的 SSM Agent。为此，请在的 Fleet Manager 中的设置选项卡上选择 自动更新 SSM Agent，如本主题后面所述。
全局或选择性更新	您可以使用State Manager（AWS Systems Manager 中的一项工具）创建一个关联，在托管式节点上自动下载并安装 SSM Agent。如果要限制对工作负载的中断，您可以创建 Systems Manager 维护时段在指定的时间段内执行安装。通过这两种方法，您可以为所有托管式节点创建全局更新配置，或者有选择地选择要更新的实例。有关创建 State Manager 关联的信息，请参阅 演练：使用 AWS CLI 自动更新 SSM Agent。有关创建维护时段的信息，请参阅 教程：使用控制台创建修补的维护时段。

方法	详细信息
针对新环境的全局或选择性更新	如果要开始使用 Systems Manager，建议使用 Quick Setup（AWS Systems Manager 中的一项工具）中的每两周更新 Systems Manager（SSM）Agent选项。使用Quick Setup，您可以为所有托管式节点创建全局更新配置，或者有选择地选择要更新的托管式节点。有关更多信息，请参阅 使用 Quick Setup 设置 Amazon EC2 主机管理。

如果您希望手动更新托管式节点上的 SSM Agent，您可以订阅新代理版本发布时 AWS 发布的通知。有关信息，请参阅订阅 SSM Agent 通知。在订阅通知后，您可以使用 Run Command 手动将一个或多个托管式节点更新为最新版本。有关更多信息，请参阅 使用 Run Command 更新 SSM Agent。

自动更新 SSM Agent

您可以将 Systems Manager 配置为自动更新 AWS 账户 中所有基于 Linux 和基于 Windows 的托管式节点上的 SSM Agent。如果启用该选项，Systems Manager 每两周自动检查一次是否存在新版本的代理。如果存在新版本，则 Systems Manager 将自使用 `AWS-UpdateSSMAgent` 中的 SSM 文档自动将代理更新为发布的最新版本。建议您选择该选项，以确保您的托管式节点始终运行最新版本的 SSM Agent。

> **Note**
>
> 如果您在使用 SSM 文档 `AWS-UpdateSSMAgent` 成功安装或更新代理之后，使用 `yum` 命令来更新托管节点上的 SSM Agent，系统可能会显示以下消息：`"Warning: RPMDB altered outside of yum."`。此消息是预期中的，您可以安全地将其忽略。

自动更新 SSM Agent

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Fleet Manager。
3. 选择设置选项卡。
4. 在代理自动更新区域中，选择自动更新 SSM Agent。

要更改机群要更新到的 SSM Agent 版本，请在设置选项卡上的代理自动更新下选择编辑。然后，在参数下的版本中输入要更新到的 SSM Agent 版本号。如果未指定，代理将更新为最新版本。

要停止将更新后的 SSM Agent 版本自动部署到账户中的所有托管式节点，请在设置选项卡上的代理自动更新下选择删除。此操作将删除在托管式节点上自动更新 SSM Agent 的 State Manager 关联。

订阅 SSM Agent 通知

Amazon Simple Notification Service (Amazon SNS) 可以在发布 AWS Systems Manager Agent (SSM Agent) 新版本时通知您。使用以下过程订阅这些通知。

> **Tip**
>
> 您还可以通过在 GitHub 上查看 SSM Agent 发布说明页面来订阅通知。

订阅 SSM Agent 通知

1. 通过 https://console.aws.amazon.com/sns/v3/home 打开 Amazon SNS 控制台。
2. 从导航栏的区域选择器中，选择美国东部（弗吉尼亚北部）（如果尚未选择）。您必须选择此 AWS 区域，因为您订阅的 SSM Agent Amazon SNS 通知仅在此区域中生成。
3. 在导航窗格中，选择订阅。
4. 选择创建订阅。
5. 对于创建订阅，请执行以下操作：
   a. 对于 Topic ARN，请使用以下 Amazon Resource Name (ARN)：

      ```
      arn:aws:sns:us-east-1:720620558202:SSM-Agent-Update
      ```

   b. 对于协议，选择 Email 或 SMS。
   c. 对于 Endpoint（端点），根据您在上一步中选择的是 Email 还是 SMS，输入用于接收通知的电子邮件地址或区号及电话号码。
   d. 选择创建订阅。
6. 如果您选择了 Email，则会收到要求您确认订阅的电子邮件消息。打开该消息，然后按照指示完成订阅。

当 SSM Agent的新版本发布时，我们会向订户发送通知。如果您不希望再收到这些通知，请通过以下步骤取消订阅。

取消订阅 SSM Agent 通知

1. 打开 Amazon SNS 控制台。
2. 在导航窗格中，选择 Subscriptions。
3. 选择该订阅，然后选择 Delete（删除）。当系统提示进行确认时，选择 Delete (删除)。

排查 SSM Agent 问题

如果您在托管式节点上运行操作时遇到问题，可能是 AWS Systems Manager Agent (SSM Agent) 出现了问题。使用以下信息可帮助您查看 SSM Agent日志文件和排查该代理的问题。

主题

- SSM Agent 已过时
- 使用 SSM Agent 日志文件进行故障排除
- 代理日志文件不会滚动 (Windows)
- 无法连接到 SSM 端点
- 验证您的 VPC 配置
- 验证 VPC DNS 相关属性
- 验证端点安全组的入口规则
- 使用 ssm-cli 排查托管节点可用性的问题

SSM Agent 已过时

如果有新工具添加至 Systems Manager 或者对现有工具进行了更新，则将发布 SSM Agent 的更新版本。无法使用代理的最新版本可能会阻止托管式节点使用 Systems Manager 的各项工具和功能。因此，我们建议您自动完成确保机器上的 SSM Agent 为最新的过程。有关更多信息，请参阅 自动更新到 SSM Agent。要获得有关 SSM Agent 更新的通知，请在 GitHub 上订阅 SSM Agent 发布说明页面。

使用 SSM Agent 日志文件进行故障排除

SSM Agent在下列日志文件中记录信息。这些文件中的信息还可以帮助您排查问题。有关 SSM Agent 日志文件的更多信息（包括如何打开调试日志记录），请参阅 查看 SSM Agent 日志。

> **Note**
>
> 如果您选择使用 Windows 文件资源管理器查看这些日志，请务必在“文件夹选项”中允许查看隐藏文件和系统文件。

在 Windows 上

- `%PROGRAMDATA%\Amazon\SSM\Logs\amazon-ssm-agent.log`
- `%PROGRAMDATA%\Amazon\SSM\Logs\errors.log`

在 Linux 和 macOS 上

- `/var/log/amazon/ssm/amazon-ssm-agent.log`
- `/var/log/amazon/ssm/errors.log`

对于 Linux 托管式节点，您可以在写入以下目录的 `messages` 文件中查找更多信息：`/var/log`。

有关使用代理日志进行故障排除的更多信息，请参阅《AWS re:Post 知识中心》中的如何使用 SSM Agent 日志排查托管实例中的 SSM Agent 问题？。

代理日志文件不会滚动 (Windows)

如果您在 seelog.xml 文件（在 Windows Server 托管式节点上）中指定基于日期的日志文件轮换，并且日志不轮换，请指定 `fullname=true` 参数。下面是指定了 `fullname=true` 参数的 seelog.xml 配置文件的示例。

```
<seelog type="adaptive" mininterval="2000000" maxinterval="100000000"
 critmsgcount="500" minlevel="debug">
   <exceptions>
      <exception filepattern="test*" minlevel="error" />
   </exceptions>
   <outputs formatid="fmtinfo">
      <console formatid="fmtinfo" />
      <rollingfile type="date" datepattern="200601021504" maxrolls="4" filename="C:
\ProgramData\Amazon\SSM\Logs\amazon-ssm-agent.log" fullname=true />
      <filter levels="error,critical" formatid="fmterror">
```

```
         <rollingfile type="date" datepattern="200601021504" maxrolls="4" filename="C:
\ProgramData\Amazon\SSM\Logs\errors.log" fullname=true />
      </filter>
   </outputs>
   <formats>
      <format id="fmterror" format="%Date %Time %LEVEL [%FuncShort @ %File.%Line] %Msg
%n" />
      <format id="fmtdebug" format="%Date %Time %LEVEL [%FuncShort @ %File.%Line] %Msg
%n" />
      <format id="fmtinfo" format="%Date %Time %LEVEL %Msg%n" />
   </formats>
</seelog>
```

无法连接到 SSM 端点

SSM Agent 必须允许到以下端点的 HTTPS（端口 443）出站流量：

- ssm.*region*.amazonaws.com
- ssmmessages.*region*.amazonaws.com

region 表示 AWS Systems Manager 支持的 AWS 区域 的标识符，例如 us-east-2 对应美国东部（俄亥俄）区域。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

> **Note**
>
> 在 2024 年之前，还需要 ec2messages.*region*.amazonaws.com。对于在 2024 年之前推出的 AWS 区域，仍然需要允许流向 ssmmessages.*region*.amazonaws.com 的流量，但流向 ec2messages.*region*.amazonaws.com 的流量是可选的。
> 对于 2024 年及之后推出的区域，需要允许流向 ssmmessages.*region*.amazonaws.com 的流量，但这些区域不支持 ec2messages.*region*.amazonaws.com 端点。

如果 SSM Agent 无法与前面的端点进行通信（如上所述），则其无法正常运行，即使您使用 AWS 提供的 Amazon Machine Images（AMIs），例如 Amazon Linux 2 或 Amazon Linux 2023。您的网络配置必须具有开放的互联网访问权限，或者您必须配置自定义 Virtual Private Cloud (VPC) 端点。如果您不打算创建自定义 VPC 端点，请检查您的互联网网关或 NAT 网关。有关如何管理 VPC 端点的更多信息，请参阅 使用适用于 Systems Manager 的 VPC 端点提高 EC2 实例的安全性。

验证您的 VPC 配置

如果使用了虚拟私有云 (VPC)，则要使用 Systems Manager 管理 EC2 实例，就必须为 `ssm.region.amazonaws.com`、`ssmmessages.region.amazonaws.com` 正确配置 VPC 端点，在某些情况下，本主题前面的 无法连接到 SSM 端点 和 `ec2messages.region.amazonaws.com` 中对此进行了说明。

> **Note**
>
> 使用 VPC 端点的替代方法是，在托管实例上允许出站 Internet 访问。在这种情况下，托管实例还必须允许以下终端节点的 HTTPS（端口 443）出站流量：
>
> - `ssm.region.amazonaws.com`
> - `ssmmessages.region.amazonaws.com`
> - `ec2messages.region.amazonaws.com`
>
> SSM Agent 将在云中启动所有与 Systems Manager 服务的连接。因此，您无需为 Systems Manager 配置防火墙以允许入站流量到达您的实例。
> 有关对这些终端节点的调用的更多信息，请参阅 参考：ec2messages、ssmmessages 和其他 API 操作。

要排查 VPC 端点的问题，请执行以下操作：

- 确保在 VPC 级别包含 VPC 端点。如果在 VPC 上找不到具有特定服务名称的 VPC 端点，请先验证是否已在 VPC 级别启用 DNS 支持。接下来，创建一个新的 VPC 端点，将其与每个可用区中的一个子网关联。
- 确保在 VPC 端点级别启用私有 DNS 名称。私有 DNS 名称默认处于启用状态，但可能在某些时候被手动禁用。
- 确保现有 VPC 端点与适当的子网关联。此外，请确保 VPCE 已与该可用区中的子网关联。

有关更多信息，请参阅以下主题：

- 《AWS PrivateLink 指南》中的使用接口 VPC 端点访问 AWS 服务
- 《AWS PrivateLink 指南》中的关联私有 DNS 名称
- 使用适用于 Systems Manager 的 VPC 端点提高 EC2 实例的安全性

验证 VPC DNS 相关属性

如果使用了虚拟私有云 (VPC)，则作为验证 VPC 配置的一部分，需确保属性 `enableDnsSupport` 和 `enableDnsHostnames` 已启用。

您可以使用 Amazon EC2 ModifyVPCAttribute API 操作或 AWS CLI 命令 modify-vpc-attribute 启用这些属性。

有关在 Amazon VPC 控制台启用这些属性的信息，请参阅《Amazon VPC 用户指南》中的查看和更新 VPC 的 DNS 属性。

> **Note**
>
> 使用 VPC 端点的替代方法是，在托管实例上允许出站 Internet 访问。在这种情况下，托管实例还必须允许以下终端节点的 HTTPS（端口 443）出站流量：
>
> - `ssm.region.amazonaws.com`
> - `ssmmessages.region.amazonaws.com`
> - `ec2messages.region.amazonaws.com`
>
> SSM Agent 将在云中启动所有与 Systems Manager 服务的连接。因此，您无需为 Systems Manager 配置防火墙以允许入站流量到达您的实例。
> 有关对这些终端节点的调用的更多信息，请参阅 参考：ec2messages、ssmmessages 和其他 API 操作。

验证端点安全组的入口规则

确保您配置的任何 VPC 端节（`ssm`、`ssmmessages` 和 `ec2messages`）在其安全组中包含入口规则，以允许流量在端口 443 上进入。如有必要，您可以在 VPC 中创建一个新的安全组，并设置一条入口规则来允许端口 443 上的流量通过 VPC 的无类别域间路由 (CIDR) 块。创建安全组后，将其附加到每个 VPC 端点。

有关更多信息，请参阅以下主题：

- 如何创建 VPC 端点，以便在没有互联网接入的情况下使用 System Manager 管理私有 EC2 实例？AWS re:Post
- 《Amazon VPC 用户指南》中的 VPC CIDR 块

使用 ssm-cli 排查托管节点可用性的问题

从 SSM Agent 版本 3.1.501.0 开始，可以使用 ssm-cli 来确定托管式节点是否满足由 Systems Manager 管理的主要要求，并出现在 Fleet Manager 中的托管式节点列表。ssm-cli 是包含在 SSM Agent 安装中的独立命令行工具。包含用于收集所需信息的预配置命令，以帮助您诊断您已确认正在运行的 Amazon EC2 实例或非 EC2 计算机未包含在 Systems Manager 的托管式节点列表中的原因。这些命令会在您指定 get-diagnostics 选项时运行。

有关更多信息，请参阅 使用 ssm-cli 排除托管节点可用性的故障。

AWS Systems Manager 中的安全性

云安全性一直是 Amazon Web Services 的重中之重。作为 AWS 客户，您将从专为满足大多数安全敏感型组织的要求而打造的数据中心和网络架构中受益。

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

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

此文档将帮助您了解如何在使用 AWS Systems Manager 时应用责任共担模型。以下主题说明如何配置 Systems Manager 以实现您的安全性和合规性目标。您还将了解如何使用其他 AWS 服务以帮助您监控和保护 Systems Manager 资源。

主题

- AWS Systems Manager 中的数据保护
- 对 AWS Systems Manager 进行身份和访问管理
- 将服务相关角色用于 Systems Manager
- AWS Systems Manager 中的日志记录和监控
- AWS Systems Manager 的合规性验证
- AWS Systems Manager 中的故障恢复能力
- AWS Systems Manager 中的基础设施安全性
- AWS Systems Manager 中的配置和漏洞分析
- Systems Manager 的安全最佳实践

AWS Systems Manager 中的数据保护

数据保护指在数据传输（发往和离开 Systems Manager 时）和处于静态（存储 AWS 数据中心内）期间保护数据。

AWS 责任共担模式适用于 AWS Systems Manager 中的数据保护。如该模式中所述，AWS 负责保护运行所有 AWS Cloud 的全球基础架构。您负责维护对托管在此基础结构上的内容的控制。您还负

责您所使用的 AWS 服务 的安全配置和管理任务。有关数据隐私的更多信息，请参阅数据隐私常见问题。有关欧洲数据保护的信息，请参阅 AWS Security Blog 上的 AWS Shared Responsibility Model and GDPR 博客文章。

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

- 对每个账户使用多重身份验证（MFA）。
- 使用 SSL/TLS 与 AWS 资源进行通信。我们要求使用 TLS 1.2，建议使用 TLS 1.3。
- 使用 AWS CloudTrail 设置 API 和用户活动日记账记录。有关使用 CloudTrail 跟踪来捕获 AWS 活动的信息，请参阅《AWS CloudTrail 用户指南》中的使用 CloudTrail 跟踪。
- 使用 AWS 加密解决方案以及 AWS 服务中的所有默认安全控制。
- 使用高级托管安全服务（例如 Amazon Macie），它有助于发现和保护存储在 Amazon S3 中的敏感数据。
- 如果在通过命令行界面或 API 访问 AWS 时需要经过 FIPS 140-3 验证的加密模块，请使用 FIPS 端点。有关可用的 FIPS 端点的更多信息，请参阅《美国联邦信息处理标准（FIPS）第 140-3 版》。

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

数据加密

静态加密

Parameter Store 参数

您可以在 Parameter Store（AWS Systems Manager 中的一项工具）中创建的参数类型包括 `String`、`StringList` 和 `SecureString`。

所有参数，无论其类型如何，传输过程中和静止状态下都会加密。在传输过程中，使用传输层安全性协议（TLS）对参数进行加密，以便为 API 请求创建安全的 HTTPS 连接。在静态状态下，使用 AWS Key Management Service（AWS KMS）中的 AWS 拥有的密钥 对参数进行加密。有关 AWS 拥有的密钥 加密的更多信息，请参阅《AWS Key Management Service Developer Guide》中的 AWS 拥有的密钥。

SecureString 类型支持其他加密选项，建议用于所有敏感数据。您可以从以下类型的 AWS KMS 密钥中进行选择，用于加密和解密 SecureString 参数的值：

- 您账户的 AWS 托管式密钥
- 您在账户中创建的客户自主管理型密钥（CMK）
- 其他 AWS 账户 与您共享的 CMK

有关 AWS KMS 加密的更多信息，请参阅《AWS Key Management Service Developer Guide》https://docs.aws.amazon.com/kms/latest/developerguide/。

S3 存储桶中的内容

作为 Systems Manager 操作的一部分，您可以选择将数据上传或存储到一个或多个 Amazon Simple Storage Service (Amazon S3) 存储桶中。

有关 S3 存储桶加密的信息，请参阅 Amazon Simple Storage Service 用户指南中的使用加密保护数据和 Amazon S3 中的数据保护。

以下是在 Systems Manager 活动中，您可以上传或存储到 S3 存储桶中的数据类型。

- Run Command（AWS Systems Manager 中的一项工具）中命令的输出结果
- Distributor（AWS Systems Manager 中的一项工具）中的软件包
- Patch Manager（AWS Systems Manager 中的一项工具）中的修补操作日志
- Patch Manager 补丁覆盖列表
- 要在 Automation（AWS Systems Manager 中的一项工具）的运行手册工作流中运行的脚本或 Ansible Playbook
- 要与 Compliance（AWS Systems Manager 中的一项工具）中的扫描结合使用的 Chef InSpec 配置文件
- AWS CloudTrail 日志
- Session Manager（AWS Systems Manager 中的一项工具）中的会话历史记录日志
- 来自 Explorer（AWS Systems Manager 中的一项工具）的报告
- 来自 OpsCenter（AWS Systems Manager 中的一项工具）的 OpsData
- 用于自动化工作流的 AWS CloudFormation 模板
- 来自资源数据同步扫描的合规性数据

- 在托管式节点上的State Manager（AWS Systems Manager 中的一项工具）中创建或编辑关联的请求的输出
- 可使用 AWS 托管 SSM 文档 `AWS-RunDocument` 运行的自定义 Systems Manager 文档（SSM 文档）

CloudWatch Logs 日志组

作为 Systems Manager 操作的一部分，您可以选择将数据流式传输到一个或多个 Amazon CloudWatch Logs 日志组。

有关 CloudWatch Logs 日志组加密的信息，请参阅 Amazon CloudWatch Logs 用户指南中的使用 AWS Key Management Service 加密 CloudWatch Logs 中的日志数据。

以下是在 Systems Manager 活动中，您可以流式传输到 CloudWatch Logs 日志组中的数据类型。

- Run Command 命令的输出
- 在自动化运行手册中使用 `aws:executeScript` 操作的脚本运行的输出
- Session Manager 会话历史记录日志
- 托管式节点上 SSM Agent 中的日志

传输中加密

建议您使用传输层安全性 (TLS) 等加密协议加密在客户端和节点之间传输的敏感数据。

Systems Manager 为传输中的数据加密提供以下支持。

与 Systems Manager API 端点的连接

Systems Manager API 端点仅支持基于 HTTPS 的安全连接。使用 AWS Management Console、AWS SDK 或 Systems Manager API 管理 Systems Manager 资源时，所有通信都使用传输层安全性 (TLS) 进行加密。有关 API 端点的完整列表，请参阅《Amazon Web Services 一般参考》中的 AWS 服务 endpoints。

托管式实例

AWS 在 Amazon Elastic Compute Cloud (Amazon EC2) 实例之间提供安全的私有连接。此外，我们使用带 256 位加密的 AEAD 算法，自动对同一 Virtual Private Cloud (VPC) 或对等 VPC 中支持实例之间的传输中流量进行加密。此加密功能将使用基础硬件的分载功能，对网络性能不会造成影响。支持的实例包括：C5n、G4、I3en、M5dn、M5n、P3dn、R5dn 和 R5n。

Session Manager 会话

默认情况下，Session Manager 使用 TLS 1.3 来加密您的账户中用户的本地计算机与您的 EC2 实例之间传输的会话数据。您还可以选择使用已经在 AWS KMS 中创建的 AWS KMS key 来进一步加密传输中的数据。AWS KMS 加密支持 `Standard_Stream`、`InteractiveCommands` 和 `NonInteractiveCommands` 会话类型。

Run Command访问

默认情况下，使用 Run Command 对节点进行远程访问所用的加密协议为 TLS 1.3，连接创建请求所用的签名版本为 SigV4。

互联网络流量隐私

您可以使用 Amazon Virtual Private Cloud (Amazon VPC) 在托管式节点中的资源之间创建边界，并控制它们、本地网络和互联网之间的流量。有关详细信息，请参阅使用适用于 Systems Manager 的 VPC 端点提高 EC2 实例的安全性。

有关 Amazon Virtual Private Cloud 安全性的更多信息，请参阅 Amazon VPC 用户指南中的 Amazon VPC 中的互联网络流量隐私。

对 AWS Systems Manager 进行身份和访问管理

AWS Identity and Access Management（IAM）是一项 AWS 服务，可以帮助管理员安全地控制对 AWS 资源的访问。IAM 管理员控制谁可以通过身份验证（登录）和授权（具有权限）来使用 Systems Manager 资源。IAM 是一项无需额外费用即可使用的 AWS 服务。

主题

- 受众
- 使用身份进行身份验证
- 使用策略管理访问
- AWS Systems Manager 如何与 IAM 协同工作
- AWS Systems Manager 基于身份的策略示例
- 适用于 AWS Systems Manager 的 AWS 托管式策略
- 对 AWS Systems Manager 身份和访问进行故障排除

受众

使用 AWS Identity and Access Management（IAM）的方式因您可以在 Systems Manager 中执行的操作而异。

服务用户：如果使用 Systems Manager 服务来完成作业，则您的管理员会为您提供所需的凭证和权限。当您使用更多 Systems Manager 特征来完成工作时，您可能需要额外权限。了解如何管理访问权限有助于您向管理员请求适合的权限。如果您无法访问 Systems Manager 中的特征，请参阅 对 AWS Systems Manager 身份和访问进行故障排除。

服务管理员：如果您在公司负责管理 Systems Manager 资源，则您可能具有 Systems Manager 的完全访问权限。您有责任确定您的服务用户应访问哪些 Systems Manager 特征和资源。然后，您必须向 IAM 管理员提交请求以更改服务用户的权限。请查看该页面上的信息以了解 IAM 的基本概念。要了解有关您的公司如何将 IAM 与 Systems Manager 搭配使用的更多信息，请参阅AWS Systems Manager 如何与 IAM 协同工作。

IAM 管理员：如果您是 IAM 管理员，您可能希望了解如何编写策略以管理对 Systems Manager 的访问权限的详细信息。要查看您可在 IAM 中使用的 Systems Manager 基于身份的策略示例，请参阅 AWS Systems Manager 基于身份的策略示例。

使用身份进行身份验证

身份验证是您使用身份凭证登录 AWS 的方法。您必须作为 AWS 账户根用户、IAM 用户或通过代入 IAM 角色进行身份验证（登录到 AWS）。

您可以使用通过身份源提供的凭证以联合身份登录到 AWS。AWS IAM Identity Center（IAM Identity Center）用户、您公司的单点登录身份验证以及您的 Google 或 Facebook 凭证都是联合身份的示例。当您以联合身份登录时，您的管理员以前使用 IAM 角色设置了身份联合验证。当您使用联合身份验证访问 AWS 时，您就是在间接代入角色。

根据您的用户类型，您可以登录 AWS Management Console 或 AWS 访问门户。有关登录到 AWS 的更多信息，请参阅《AWS 登录 User Guide》中的 How to sign in to your AWS 账户。

如果您以编程方式访问 AWS，则 AWS 将提供软件开发工具包（SDK）和命令行界面（CLI），以便使用您的凭证以加密方式签署您的请求。如果您不使用 AWS 工具，则必须自行对请求签名。有关使用推荐的方法自行签署请求的更多信息，请参阅《IAM 用户指南》中的用于签署 API 请求的 AWS 签名版本 4。

无论使用何种身份验证方法，您都可能需要提供其他安全信息。例如，AWS 建议您使用多重身份验证（MFA）来提高账户的安全性。要了解更多信息，请参阅《AWS IAM Identity Center 用户指南》中的多重身份验证和《IAM 用户指南》中的 IAM 中的 AWS 多重身份验证。

AWS 账户根用户

当您创建 AWS 账户时，最初使用的是一个对账户中所有 AWS 服务和资源拥有完全访问权限的登录身份。此身份称为 AWS 账户根用户，使用您创建账户时所用的电子邮件地址和密码登录，即可获得该身份。强烈建议您不要使用根用户执行日常任务。保护好根用户凭证，并使用这些凭证来执行仅根用户可以执行的任务。有关需要您以根用户身份登录的任务的完整列表，请参阅《IAM 用户指南》中的需要根用户凭证的任务。

IAM 用户和群组

IAM 用户是 AWS 账户内对某个人员或应用程序具有特定权限的一个身份。在可能的情况下，我们建议使用临时凭证，而不是创建具有长期凭证（如密码和访问密钥）的 IAM 用户。但是，如果您有一些特定的使用场景需要长期凭证以及 IAM 用户，建议您轮换访问密钥。有关更多信息，请参阅《IAM 用户指南》中的对于需要长期凭证的用例，应在需要时更新访问密钥。

IAM 组是一个指定一组 IAM 用户的身份。您不能使用组的身份登录。您可以使用组来一次性为多个用户指定权限。如果有大量用户，使用组可以更轻松地管理用户权限。例如，您可能具有一个名为 IAMAdmins 的组，并为该组授予权限以管理 IAM 资源。

用户与角色不同。用户唯一地与某个人员或应用程序关联，而角色旨在让需要它的任何人代入。用户具有永久的长期凭证，而角色提供临时凭证。要了解更多信息，请参阅《IAM 用户指南》中的 IAM 用户的使用案例。

IAM 角色

IAM 角色是 AWS 账户中具有特定权限的身份。它类似于 IAM 用户，但与特定人员不关联。要在 AWS Management Console中临时代入 IAM 角色，可以从用户切换到 IAM 角色（控制台）。您可以调用 AWS CLI 或 AWS API 操作或使用自定义网址以代入角色。有关使用角色的方法的更多信息，请参阅《IAM 用户指南》中的代入角色的方法。

具有临时凭证的 IAM 角色在以下情况下很有用：

- 联合用户访问：要向联合身份分配权限，请创建角色并为角色定义权限。当联合身份进行身份验证时，该身份将与角色相关联并被授予由此角色定义的权限。有关用于联合身份验证的角色的信息，请参阅《IAM 用户指南》中的针对第三方身份提供商创建角色（联合身份验证）。如果您使用

IAM Identity Center，则需要配置权限集。为控制您的身份在进行身份验证后可以访问的内容，IAM Identity Center 将权限集与 IAM 中的角色相关联。有关权限集的信息，请参阅《AWS IAM Identity Center 用户指南》中的权限集。

- 临时 IAM 用户权限：IAM 用户可代入 IAM 用户或角色，以暂时获得针对特定任务的不同权限。
- 跨账户存取：您可以使用 IAM 角色以允许不同账户中的某个人（可信主体）访问您的账户中的资源。角色是授予跨账户访问权限的主要方式。但是，对于某些 AWS 服务，您可以将策略直接附加到资源（而不是使用角色作为代理）。要了解用于跨账户访问的角色和基于资源的策略之间的差别，请参阅 IAM 用户指南中的 IAM 中的跨账户资源访问。
- 跨服务访问：某些 AWS 服务使用其他 AWS 服务中的特征。例如，当您在某个服务中进行调用时，该服务通常会在 Amazon EC2 中运行应用程序或在 Simple Storage Service（Amazon S3）中存储对象。服务可能会使用发出调用的主体的权限、使用服务角色或使用服务相关角色来执行此操作。
  - 转发访问会话（FAS）：当您使用 IAM 用户或角色在 AWS 中执行操作时，您将被视为主体。使用某些服务时，您可能会执行一个操作，然后此操作在其他服务中启动另一个操作。FAS 使用主体调用 AWS 服务的权限，结合请求的 AWS 服务，向下游服务发出请求。只有在服务收到需要与其他 AWS 服务或资源交互才能完成的请求时，才会发出 FAS 请求。在这种情况下，您必须具有执行这两项操作的权限。有关发出 FAS 请求时的策略详情，请参阅转发访问会话。
  - 服务角色 - 服务角色是服务代表您在您的账户中执行操作而分派的 IAM 角色。IAM 管理员可以在 IAM 中创建、修改和删除服务角色。有关更多信息，请参阅《IAM 用户指南》中的创建向 AWS 服务委派权限的角色。
  - 服务相关角色：服务相关角色是与 AWS 服务 关联的一种服务角色。服务可以代入代表您执行操作的角色。服务相关角色显示在您的 AWS 账户 中，并由该服务拥有。IAM 管理员可以查看但不能编辑服务相关角色的权限。
- 在 Amazon EC2 上运行的应用程序：您可以使用 IAM 角色管理在 EC2 实例上运行并发出 AWS CLI 或 AWS API 请求的应用程序的临时凭证。这优先于在 EC2 实例中存储访问密钥。要将 AWS 角色分配给 EC2 实例并使其对该实例的所有应用程序可用，您可以创建一个附加到实例的实例配置文件。实例配置文件包含角色，并使 EC2 实例上运行的程序能够获得临时凭证。有关更多信息，请参阅《IAM 用户指南》中的使用 IAM 角色向在 Amazon EC2 实例上运行的应用程序授予权限。

使用策略管理访问

您将创建策略并将其附加到 AWS 身份或资源，以控制 AWS 中的访问。策略是 AWS 中的对象；在与身份或资源相关联时，策略定义它们的权限。在主体（用户、根用户或角色会话）发出请求时，AWS 将评估这些策略。策略中的权限确定是允许还是拒绝请求。大多数策略在 AWS 中存储为 JSON 文档。有关 JSON 策略文档的结构和内容的更多信息，请参阅 IAM 用户指南中的 JSON 策略概览。

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

默认情况下，用户和角色没有权限。要授予用户对所需资源执行操作的权限，IAM 管理员可以创建 IAM 策略。管理员随后可以向角色添加 IAM 策略，用户可以代入角色。

IAM 策略定义操作的权限，无关乎您使用哪种方法执行操作。例如，假设您有一个允许 `iam:GetRole` 操作的策略。具有该策略的用户可以从 AWS Management Console、AWS CLI 或 AWS API 获取角色信息。

基于身份的策略

基于身份的策略是可附加到身份（如 IAM 用户、用户组或角色）的 JSON 权限策略文档。这些策略控制用户和角色可在何种条件下对哪些资源执行哪些操作。要了解如何创建基于身份的策略，请参阅《IAM 用户指南》中的使用客户托管策略定义自定义 IAM 权限。

基于身份的策略可以进一步归类为内联策略或托管式策略。内联策略直接嵌入单个用户、组或角色中。托管策略是可以附加到 AWS 账户 中的多个用户、组和角色的独立策略。托管式策略包括 AWS 托管式策略和客户托管策略。要了解如何在托管策略和内联策略之间进行选择，请参阅《IAM 用户指南》中的在托管策略与内联策略之间进行选择。

有关适用于 Systems Manager 的 AWS 托管式策略的信息，请参阅 AWS Systems Manager 托管式策略。

基于资源的策略

基于资源的策略是附加到资源的 JSON 策略文档。基于资源的策略的示例包括 IAM 角色信任策略和 Amazon S3 存储桶策略。在支持基于资源的策略的服务中，服务管理员可以使用它们来控制对特定资源的访问。对于在其中附加策略的资源，策略定义指定主体可以对该资源执行哪些操作以及在什么条件下执行。您必须在基于资源的策略中指定主体。主体可以包括账户、用户、角色、联合用户或 AWS 服务。

基于资源的策略是位于该服务中的内联策略。您不能在基于资源的策略中使用来自 IAM 的 AWS 托管式策略。

策略条件键

用户和角色可以执行的操作以及可以执行这些操作的资源，可能会受到特定条件的进一步限制。

在 JSON 策略文档中，`Condition` 元素（或 `Condition` 块）可以指定语句生效的条件。`Condition` 元素是可选的。您可以创建使用条件运算符（例如 `StringEquals` 或 `StringNotLike`）的条件表达式，让策略中的条件与请求中的值相匹配。

如果您在一个语句中指定多个 Condition 元素，或在单个 Condition 元素中指定多个键，则 AWS 使用逻辑 AND 运算评估它们。如果您为单个条件键指定多个值，则 AWS 使用逻辑 OR 运算来评估条件。在授予语句的权限之前必须满足所有的条件。

在指定条件时，您也可以使用占位符变量。例如，只有在使用 IAM 用户名标记 IAM 用户时，您才能为其授予访问资源的权限。有关更多信息，请参阅《IAM 用户指南》中的 IAM 策略元素：变量和标签。

AWS 支持全局条件键和特定于服务的条件键。有关更多信息，请参阅《IAM 用户指南》中的 AWS 全局条件上下文键。

> **Important**
>
> 如果使用 Systems Manager Automation，我们建议您不要在策略中使用 aws:SourceIp 条件键。此条件键的行为取决于多种因素，包括是否提供了用于 Automation 运行手册执行的 IAM 角色以及运行手册中使用的 Automation 操作。因此，条件键可能会产生意外的行为。因此，我们建议您不要使用它。

Systems Manager 支持众多自有的条件键。有关更多信息，请参阅《Service Authorization Reference》中的 Condition keys for AWS Systems Manager。《Service Authorization Reference》中的 Resource types defined by AWS Systems Manager 列出了可用于 Systems Manager 特定条件键的操作和资源。

如果策略必须依赖于 Systems Manager 服务拥有的服务主体名称，建议使用 aws:PrincipalServiceNamesList 多值条件键而不是 aws:PrincipalServiceName 条件键来检查其是否存在。aws:PrincipalServiceName 条件键仅包含服务主体名称列表中的一个条目，并且该条目不一定是您预期的服务主体名称。以下 Condition 块演示了检查 ssm.amazonaws.com 是否存在的过程。

```
{
    "Condition": {
        "ForAnyValue:StringEquals": {
            "aws:PrincipalServiceNamesList": "ssm.amazonaws.com"
        }
    }
}
```

要查看 Systems Manager 基于身份的策略的示例，请参阅 AWS Systems Manager 基于身份的策略示例。

访问控制列表（ACL）

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

Amazon S3、AWS WAF 和 Amazon VPC 是支持 ACL 的服务示例。要了解有关 ACL 的更多信息，请参阅《Amazon Simple Storage Service 开发人员指南》中的访问控制列表（ACL）概览。

其他策略类型

AWS 支持额外的、不太常用的策略类型。这些策略类型可以设置更常用的策略类型向您授予的最大权限。

- 权限边界：权限边界是一个高级特征，用于设置基于身份的策略可以为 IAM 实体（IAM 用户或角色）授予的最大权限。您可为实体设置权限边界。这些结果权限是实体基于身份的策略及其权限边界的交集。在 `Principal` 中指定用户或角色的基于资源的策略不受权限边界限制。任一项策略中的显式拒绝将覆盖允许。有关权限边界的更多信息，请参阅IAM 用户指南中的 IAM 实体的权限边界。
- 服务控制策略（SCP）– SCP 是 JSON 策略，指定了组织或组织单元（OU）在 AWS Organizations 中的最大权限。AWS Organizations 服务可以分组和集中管理您的企业拥有的多个 AWS 账户 账户。如果在组织内启用了所有特征，则可对任意或全部账户应用服务控制策略（SCP）。SCP 限制成员账户中实体（包括每个 AWS 账户根用户）的权限。有关组织和 SCP 的更多信息，请参阅《AWS Organizations 用户指南》中的服务控制策略。
- 资源控制策略（RCP）– RCP 是 JSON 策略，您可以使用它们设置账户中资源的最大可用权限，而无需更新附加到您拥有的每个资源的 IAM 策略。RCP 限制了成员账户中资源的权限，并可能影响身份（包括 AWS 账户根用户）的有效权限，无论这些身份是否属于您的组织。有关 Organizations 和 RCP（包括支持 RCP 的 AWS 服务列表）的更多信息，请参阅《AWS Organizations User Guide》中的 Resource control policies (RCPs)。
- 会话策略：会话策略是当您以编程方式为角色或联合用户创建临时会话时作为参数传递的高级策略。结果会话的权限是用户或角色的基于身份的策略和会话策略的交集。权限也可以来自基于资源的策略。任一项策略中的显式拒绝将覆盖允许。有关更多信息，请参阅 IAM 用户指南中的会话策略。

多个策略类型

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

AWS Systems Manager 如何与 IAM 协同工作

在使用 AWS Identity and Access Management (IAM) 管理对 AWS Systems Manager 的访问权限之前，您应该了解哪些 IAM 功能可用于 Systems Manager。要大致了解 Systems Manager 和其他 AWS 服务如何与 IAM 一起使用，请参阅《IAM 用户指南》中的与 IAM 一起使用的 AWS 服务。

主题

- Systems Manager 基于身份的策略
- Systems Manager 基于资源的策略
- 基于 Systems Manager 标签的授权
- Systems Manager IAM 角色

Systems Manager 基于身份的策略

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

操作

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

JSON 策略的 `Action` 元素描述可用于在策略中允许或拒绝访问的操作。策略操作通常与关联的 AWS API 操作同名。有一些例外情况，例如没有匹配 API 操作的仅限权限 操作。还有一些操作需要在策略中执行多个操作。这些附加操作称为相关操作。

在策略中包含操作以授予执行关联操作的权限。

Systems Manager中的策略操作在操作前使用以下前缀：`ssm:`。例如，要授予某人使用 Systems Manager `PutParameter` API 操作创建 Systems Manager 参数（SSM 参数）的权限，您应将 `ssm:PutParameter` 操作纳入其策略中。策略语句必须包括 `Action` 或 `NotAction` 元素。Systems Manager 定义了自己的一组操作，这些操作描述了可使用该服务执行的任务。

要在单个语句中指定多项操作，请使用逗号将它们隔开，如下所示：

```
"Action": [
      "ssm:action1",
      "ssm:action2"
```

```
]
```

> **Note**
>
> AWS Systems Manager 中以下工具在操作之前使用不同的前缀。
>
> - AWS AppConfig 在操作之前使用前缀 `appconfig:`。
> - Incident Manager 在操作之前使用前缀 `ssm-incidents:` 或 `ssm-contacts:`。
> - Systems Manager GUI Connect 在操作之前使用前缀 `ssm-guiconnect:`。
> - Quick Setup 在操作之前使用前缀 `ssm-quicksetup:`。

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

```
"Action": "ssm:Describe*"
```

要查看 Systems Manager 操作的列表，请参阅《服务授权参考》中的 AWS Systems Manager 定义的操作。

资源

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

`Resource` JSON 策略元素指定要向其应用操作的一个或多个对象。语句必须包含 `Resource` 或 `NotResource` 元素。作为最佳实践，请使用其 Amazon 资源名称（ARN）指定资源。对于支持特定资源类型（称为资源级权限）的操作，您可以执行此操作。

对于不支持资源级权限的操作（如列出操作），请使用通配符（*）指示语句应用于所有资源。

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

例如，Systems Manager 维护时段资源具有以下 ARN 格式。

```
arn:aws:ssm:region:account-id:maintenancewindow/window-id
```

要在美国东部（俄亥俄）区域的语句中指定 mw-0c50858d01EXAMPLE 维护时段，请使用与以下类似的 ARN。

```
"Resource": "arn:aws:ssm:us-east-2:123456789012:maintenancewindow/mw-0c50858d01EXAMPLE"
```

要指定属于特定账户的所有维护时段，请使用通配符 (*)。

```
"Resource": "arn:aws:ssm:region:123456789012:maintenancewindow/*"
```

对于 Parameter Store API 操作，可以使用层次结构名称和 AWS Identity and Access Management (IAM) 策略提供或限制对层次结构的一个级别中所有参数的访问权限，如下所示。

```
"Resource": "arn:aws:ssm:region:123456789012:parameter/Dev/ERP/Oracle/*"
```

某些 Systems Manager 操作（例如用于创建资源的那些操作）不能在特定资源上执行。在这些情况下，您必须使用通配符 （*)。

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

有些 Systems Manager API 操作接受多个资源。要在单个语句中指定多个资源，请使用逗号分隔其 ARN，如下所示。

```
"Resource": [
      "resource1",
      "resource2"
```

> **Note**
>
> 大多数 AWS 服务将 ARN 中的冒号（:）或正斜杠（/）视为同一个字符。不过，Systems Manager 在资源模式和规则中要求精确匹配。创建事件模式时，请务必使用正确的 ARN 字符，以使其与资源的 ARN 匹配。

下表介绍了 Systems Manager 支持的资源类型的 ARN 格式。

> **Note**
>
> 请注意以下 ARN 格式的例外情况。

- AWS Systems Manager 中以下工具在操作之前使用不同的前缀。
  - AWS AppConfig 在操作之前使用前缀 `appconfig:`。
  - Incident Manager 在操作之前使用前缀 `ssm-incidents:` 或 `ssm-contacts:`。
  - Systems Manager GUI Connect 在操作之前使用前缀 `ssm-guiconnect`。
- Amazon 拥有的文档和自动化定义资源，以及 Amazon 和第三方来源提供的公有参数，都未在 ARN 格式中包含账户 ID。例如：
  - SSM 文档 `AWS-RunPatchBaseline`：

    ```
    arn:aws:ssm:us-east-2::document/AWS-RunPatchBaseline
    ```
  - 自动化运行手册 `AWS-ConfigureMaintenanceWindows`：

    ```
    arn:aws:ssm:us-east-2::automation-definition/AWS-
    ConfigureMaintenanceWindows
    ```
  - 公有参数 `/aws/service/bottlerocket/aws-ecs-1-nvidia/x86_64/1.13.4/image_version`：

    ```
    arn:aws:ssm:us-east-2::parameter/aws/service/bottlerocket/aws-
    ecs-1-nvidia/x86_64/1.13.4/image_version
    ```

  有关这些第三方资源类型的更多信息，请参阅以下主题：
  - 使用文档
  - 运行由 Systems Manager Automation 支持的自动化操作
  - 使用 Parameter Store 中的公有参数
- Quick Setup 在操作之前使用前缀 `ssm-quicksetup:`。

资源类型	ARN 格式
应用程序 (AWS AppConfig)	arn:aws:appconfig:*region*:*account-id* :application/*application-id*
关联	arn:aws:ssm:*region*:*account-id* :association/*association-id*
自动化执行	arn:aws:ssm:*region*:*account-id* :automation-execution/*automation-execution-id*

资源类型	ARN 格式
自动化定义 (使用版本子资源)	arn:aws:ssm:*region*:*account-id* :automation-definition/*automation-definition-id* :*version-id* [1]
配置文件 (AWS AppConfig)	arn:aws:appconfig:*region*:*account-id* :application/*application-id* /configurationprofile/*configurationprofile-id*
联系人 (Incident Manager)	arn:aws:ssm-contacts:*region*:*account-id* :contact/*contact-alias*
部署策略 (AWS AppConfig)	arn:aws:appconfig:*region*:*account-id* :deploymentstrategy/*deploymentstrategy-id*
文档	arn:aws:ssm:*region*:*account-id* :document/*document-name*
环境 (AWS AppConfig)	arn:aws:appconfig:*region*:*account-id* :application/*application-id* /environment/*environment-id*
事件	arn:aws:ssm-incidents:*region*:*account-id* :incident-record/*response-plan-name* /*incident-id*
维护时段	arn:aws:ssm:*region*:*account-id* :maintenancewindow/*window-id*
托管式节点	arn:aws:ssm:*region*:*account-id* :managed-instance/*managed-node-id*
托管式节点清单	arn:aws:ssm:*region*:*account-id* :managed-instance-inventory /*managed-node-id*
OpsItem	arn:aws:ssm:*region*:*account-id* :opsitem/*OpsItem-id*

资源类型	ARN 格式
参数	一级参数： • arn:aws:ssm:*region*:*account-id* :parameter/*parameter-name*/ 使用分层结构命名的参数： • arn:aws:ssm:*region*:*account-id* :parameter/*parameter-name-root* /*level-2*/*level-3*/*level-4*/*level-5* [2]
补丁基准	arn:aws:ssm:*region*:*account-id* :patchbaseline/*patch-baseline-id*
响应计划	arn:aws:ssm-incidents:*region*:*account-id* :response-plan/*response-plan-name*
会话	arn:aws:ssm:*region*:*account-id* :session/*session-id* [3]
所有 Systems Manager 资源	arn:aws:ssm:*
指定 AWS 账户在指定 AWS 区域拥有的所有 Systems Manager 资源	arn:aws:ssm:*region*:*account-id* :*

[1]
对于自动化定义，Systems Manager 支持二级资源版本 ID。在 AWS 中，这些二级资源称作子资源。通过指定自动化定义资源的版本子资源，您可以提供对自动化定义的特定版本的访问权限。例如，您可能需要确保在节点管理中只使用最新版本的自动化定义。

[2]
要组织和管理参数，可以使用分层结构为参数创建名称。在分层结构中，参数名称可以包含使用正斜杠定义的路径。您可以命名最多包含十五个级别的参数资源。建议创建反映环境中现有层次结构的层次结构。有关更多信息，请参阅 在 Systems Manager 中创建 Parameter Store 参数。

3

在大多数情况下，会话 ID 是用启动会话的账户用户的 ID 构造的，外加字母数字后缀。例如：

```
arn:aws:us-east-2:111122223333:session/JohnDoe-1a2b3c4sEXAMPLE
```

但是，如果用户 ID 不可用，则改为通过以下方式构造 ARN：

```
arn:aws:us-east-2:111122223333:session/session-1a2b3c4sEXAMPLE
```

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

有关 Systems Manager 资源类型及其 ARN 的列表，请参阅《服务授权参考》中的 AWS Systems Manager 定义的资源。要了解您可以在哪些操作中指定每个资源的 ARN，请参阅 AWS Systems Manager 定义的操作。

Systems Manager 的条件键

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

在 Condition 元素（或 Condition 块）中，可以指定语句生效的条件。Condition 元素是可选的。您可以创建使用条件运算符（例如，等于或小于）的条件表达式，以使策略中的条件与请求中的值相匹配。

如果您在一个语句中指定多个 Condition 元素，或在单个 Condition 元素中指定多个键，则 AWS 使用逻辑 AND 运算评估它们。如果您为单个条件键指定多个值，则 AWS 使用逻辑 OR 运算来评估条件。在授予语句的权限之前必须满足所有的条件。

在指定条件时，您也可以使用占位符变量。例如，只有在使用 IAM 用户名标记 IAM 用户时，您才能为其授予访问资源的权限。有关更多信息，请参阅《IAM 用户指南》中的 IAM 策略元素：变量和标签。

AWS 支持全局条件键和特定于服务的条件键。要查看所有 AWS 全局条件键，请参阅《IAM 用户指南》中的 AWS 全局条件上下文键。

有关 Systems Manager 条件密钥的列表，请参阅《服务授权参考》中的 AWS Systems Manager 的条件密钥。要了解您可以对哪些操作和资源使用条件键，请参阅 AWS Systems Manager 定义的操作。

有关使用 ssm:resourceTag/* 条件键的信息，请参阅以下主题：

- 通过 SSM Agent 限制对根级别命令的访问

- 根据标签限制 Run Command 访问
- 基于实例标签限制会话访问

有关使用 ssm:Recursive、ssm:Policies 和 ssm:Overwrite 条件键的信息，请参阅 防止访问 Parameter Store API 操作。

示例

要查看 Systems Manager 基于身份的策略的示例，请参阅 AWS Systems Manager 基于身份的策略示例。

Systems Manager 基于资源的策略

其他 AWS 服务 [如 Amazon Simple Storage Service（Amazon S3）] 支持基于资源的权限策略。例如，您可以将权限策略挂载到 S3 存储桶以管理对该存储桶的访问权限。

Systems Manager 不支持基于资源的策略。

基于 Systems Manager 标签的授权

您可以将标签附加到 Systems Manager 资源或将请求中的标签传递到 Systems Manager。要基于标签控制访问，您需要使用 ssm:resourceTag/*key-name*、aws:ResourceTag/*key-name*、aws:RequestTag/*key-name* 或 aws:TagKeys 条件密钥在策略的条件元素中提供标签信息。在创建或更新下列资源类型时，您可以向其中添加标签：

- 文档
- 托管式节点
- 维护时段
- 参数
- 补丁基准
- OpsItem

要查看基于身份的策略（用于根据资源上的标签来限制对该资源的访问）的示例，请参阅基于标签查看 Systems Manager 文档。

Systems Manager IAM 角色

IAM 角色是 AWS 账户中具有特定权限的实体。

将临时凭证用于 Systems Manager

可以使用临时凭证进行联合身份验证登录，分派 IAM 角色或分派跨账户角色。您可以调用 AWS Security Token Service (AWS STS) API 操作（如 AssumeRole 或 GetFederationToken）以获得临时安全凭证。

Systems Manager 支持使用临时凭证。

服务相关角色

服务相关角色允许 AWS 服务访问其它服务中的资源以代表您完成操作。服务相关角色列在您的 IAM 账户中，并归相应的服务所有。管理员可以查看但不能编辑服务相关角色的权限。

Systems Manager 支持服务相关角色。有关创建或管理 Systems Manager 服务相关角色的详细信息，请参阅 将服务相关角色用于 Systems Manager。

服务角色

此功能允许服务代表您担任服务角色。此角色允许服务访问其他服务中的资源以代表您完成操作。服务角色显示在您的 IAM 账户中，并归相应的账户所有。这意味着管理员可以更改此角色的权限。但是，这样做可能会中断服务的功能。

Systems Manager 支持服务角色。

在 Systems Manager 中选择 IAM 角色

要让 Systems Manager 与您的托管式节点交互，您必须选择一个角色以允许 Systems Manager 代表您访问节点。如果您之前已经创建了一个服务角色或服务相关角色，则 Systems Manager 会为您提供一个角色列表供您选择。选择一个允许访问以启动和停止托管式节点的角色很重要。

要访问 EC2 实例，您必须配置实例权限。有关信息，请参阅配置 Systems Manager 所需的实例权限。

要访问混合和多云中的非 EC2 节点，AWS 账户 需要的角色是 IAM 服务角色。有关信息，请参阅在混合和多云环境中创建 Systems Manager 所需的 IAM 服务角色。

自动化工作流可以在服务角色（或代入角色）的上下文下启动。这样服务就能够代表您执行操作。如果未指定代入角色，则自动化将使用已调用执行的用户的上下文。不过，在特定情况下需要您为自动化指定服务角色。有关更多信息，请参阅 为自动化配置服务角色（担任角色）访问权限。

AWS Systems Manager 托管式策略

AWS通过提供由创建和管理的独立 IAM policy 来满足许多常用案例的要求。AWS这些 AWS 托管策略授予常见使用情形的必要权限，这样您不必调查需要哪些权限。（此外，您还可以创建您自己的自定义 IAM policy ，以授予 Systems Manager 操作和资源的相关权限。）

有关 Systems Manager 托管策略的更多信息，请参阅 适用于 AWS Systems Manager 的 AWS 托管式策略。

有关托管策略的一般信息，请参阅《IAM 用户指南》中的 AWS 托管策略。

AWS Systems Manager 基于身份的策略示例

默认情况下，AWS Identity and Access Management（IAM）实体（用户和角色）没有创建或修改 AWS Systems Manager 资源的权限。他们还无法使用 Systems Manager 控制台、AWS Command Line Interface (AWS CLI) 或 AWS API 执行任务。管理员必须创建 IAM policy，以便为用户和角色授予权限以对所需的指定资源执行特定的 API 操作。然后，管理员必须将这些策略附加到需要这些权限的用户或组。

以下示例权限策略允许用户删除位于美国东部（俄亥俄）（us-east-2）AWS 区域的名称以 **`MyDocument-`** 开头的文档。

JSON

```
{
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "ssm:DeleteDocument"
      ],
      "Resource" : [
        "arn:aws:ssm:us-east-2:111122223333:document/MyDocument-*"
      ]
    }
  ]
}
```

要了解如何使用这些示例 JSON 策略文档创建基于 IAM 身份的策略，请参阅《IAM 用户指南》中的创建 IAM policy。

主题

- 策略最佳实践
- 示例：使用 Systems Manager 控制台的权限
- 示例：允许用户查看其自有权限的权限
- 示例：读取和描述单个参数的权限
- 防止跨服务混淆代理
- 客户管理型策略示例
- 基于标签查看 Systems Manager 文档

策略最佳实践

基于身份的策略确定某个人是否可以创建、访问或删除您账户中的 Systems Manager 资源。这些操作可能会使 AWS 账户产生成本。创建或编辑基于身份的策略时，请遵循以下指南和建议：

- AWS 托管式策略及转向最低权限许可入门 – 要开始向用户和工作负载授予权限，请使用 AWS 托管式策略来为许多常见使用场景授予权限。您可以在 AWS 账户 中找到这些策略。我们建议通过定义特定于您的使用场景的 AWS 客户托管式策略来进一步减少权限。有关更多信息，请参阅《IAM 用户指南》中的 AWS 托管式策略或工作职能的 AWS 托管式策略。
- 应用最低权限：在使用 IAM 策略设置权限时，请仅授予执行任务所需的权限。为此，您可以定义在特定条件下可以对特定资源执行的操作，也称为最低权限许可。有关使用 IAM 应用权限的更多信息，请参阅《IAM 用户指南》中的 IAM 中的策略和权限。
- 使用 IAM 策略中的条件进一步限制访问权限：您可以向策略添加条件来限制对操作和资源的访问。例如，您可以编写策略条件来指定必须使用 SSL 发送所有请求。如果通过特定 AWS 服务（例如 AWS CloudFormation）使用服务操作，您还可以使用条件来授予对服务操作的访问权限。有关更多信息，请参阅《IAM 用户指南》中的 IAM JSON 策略元素：条件。
- 使用 IAM Access Analyzer 验证您的 IAM 策略，以确保权限的安全性和功能性 – IAM Access Analyzer 会验证新策略和现有策略，以确保策略符合 IAM 策略语言（JSON）和 IAM 最佳实践。IAM Access Analyzer 提供 100 多项策略检查和可操作的建议，以帮助您制定安全且功能性强的策略。有关更多信息，请参阅《IAM 用户指南》中的使用 IAM Access Analyzer 验证策略。
- 需要多重身份验证（MFA）：如果您所处的场景要求您的 AWS 账户 中有 IAM 用户或根用户，请启用 MFA 来提高安全性。若要在调用 API 操作时需要 MFA，请将 MFA 条件添加到您的策略中。有关更多信息，请参阅《IAM 用户指南》中的使用 MFA 保护 API 访问。

有关 IAM 中的最佳实操的更多信息，请参阅《IAM 用户指南》中的 IAM 中的安全最佳实践。

示例：使用 Systems Manager 控制台的权限

要访问 Systems Manager 控制台，您必须具有一组最低的权限。这些权限必须允许您列出和查看有关您 AWS 账户中的 Systems Manager 资源和其他资源的详细信息。

如果您创建的基于身份的策略比所需的最低权限更严格，则无法为具有该策略的 IAM 实体（用户或角色）按预期运行该控制台。

对于只需要调用 AWS CLI 或 AWS API 的用户，无需为其提供最低控制台权限。相反，只允许访问与您尝试执行的 API 操作相匹配的操作。

为确保用户和角色仍然可以使用 Systems Manager 控制台，还请将 AmazonSSMFullAccess 或 AmazonSSMReadOnlyAccess AWS 托管策略附加到实体。有关更多信息，请参阅《IAM 用户指南》中的为用户添加权限。

示例：允许用户查看其自有权限的权限

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

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

```
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

示例：读取和描述单个参数的权限

Example 读取并描述参数

通过将以下策略附加到身份，您可以授予参数的访问权限。

JSON

```
{
"Version": "2012-10-17",
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "ssm:GetParameter",
      "ssm:DescribeParameters"
      ],
    "Resource": "arn:aws:ssm:us-east-1:111122223333:parameter/parameter-name"
  }
]
}
```

防止跨服务混淆代理

混淆代理问题是一个安全性问题，即不具有某操作执行权限的实体可能会迫使具有更高权限的实体执行该操作。在 AWS 中，跨服务模拟可能会导致混淆代理问题。一个服务（呼叫服务）调用另一项服务（所谓的服务）时，可能会发生跨服务模拟。可以操纵调用服务，使用其权限以在其他情况下该服务不

应有访问权限的方式对另一个客户的资源进行操作。为防止这种情况，AWS 提供可帮助您保护所有服务的数据的工具，而这些服务中的服务主体有权限访问账户中的资源。

我们建议在资源策略中使用 aws:SourceArn 和 aws:SourceAccount 全局条件上下文键，以限制 AWS Systems Manager 为其他服务提供的资源访问权限。如果 aws:SourceArn 值不包含账户 ID，例如 S3 存储桶的 Amazon 资源名称（ARN），则您必须使用两个全局条件上下文键来限制权限。如果同时使用全局条件上下文密钥和包含账户 ID 的 aws:SourceArn 值，则 aws:SourceAccount 值和 aws:SourceArn 值中的账户在同一策略语句中使用时，必须使用相同的账户 ID。如果您只希望将一个资源与跨服务访问相关联，请使用 aws:SourceArn。如果您想允许该账户中的任何资源与跨服务使用操作相关联，请使用 aws:SourceAccount。

以下几节提供 AWS Systems Manager 工具的策略示例。

混合激活策略示例

对于在混合激活中使用的服务角色，aws:SourceArn 的值必须是 AWS 账户 的 ARN。请务必指定您在其中创建混合激活的 ARN 中的 AWS 区域。如果您不知道资源的完整 ARN，或正在指定多个资源，请针对 ARN 未知部分使用带有通配符 (*) 的 aws:SourceArn 全局上下文条件键。例如 arn:aws:ssm:*:*region*:*123456789012*:*。

以下示例演示了如何通过为自动化使用 aws:SourceArn 和 aws:SourceAccount 全局条件上下文键，以防止在美国东部（俄亥俄）区域（us-east-2）出现混淆代理问题。

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"",
         "Effect":"Allow",
         "Principal":{
            "Service":"ssm.amazonaws.com"
         },
         "Action":"sts:AssumeRole",
         "Condition":{
            "StringEquals":{
               "aws:SourceAccount":"123456789012"
            },
            "ArnEquals":{
               "aws:SourceArn":"arn:aws:ssm:us-east-2:123456789012:*"
```

```
            }
         }
      }
   ]
}
```

资源数据同步策略示例

利用 Systems Manager Inventory、Explorer 和 Compliance，您能够创建资源数据同步，将运营数据 (OpsData) 集中存储在 Amazon Simple Storage Service 中央存储桶中。如果希望使用 AWS Key Management Service (AWS KMS) 来加密资源数据同步，则必须创建包含以下策略的新键，或更新现有键并向其添加此策略。此策略中的 `aws:SourceArn` 和 `aws:SourceAccount` 条件键可防止混淆代理问题。以下为策略示例。

JSON

```
{
    "Version": "2012-10-17",
    "Id": "ssm-access-policy",
    "Statement": [
        {
            "Sid": "ssm-access-policy-statement",
            "Action": [
                "kms:GenerateDataKey"
            ],
            "Effect": "Allow",
            "Principal": {
                "Service": "ssm.amazonaws.com"
            },
            "Resource": "arn:aws:kms:us-east-2:123456789012:key/KMS_key_id",
            "Condition": {
                "StringLike": {
                    "aws:SourceAccount": "123456789012"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:ssm:*:123456789012:role/aws-
service-role/ssm.amazonaws.com/AWSServiceRoleForAmazonSSM"
                }
            }
        }
    ]
```

```
}
```

> **Note**
>
> 策略示例中的 ARN 让系统能够加密来自除 AWS Security Hub 之外所有来源的 OpsData。如果您需要加密 Security Hub 数据，例如使用 Explorer 收集 Security Hub 数据，则必须附加指定以下 ARN 的额外策略：
> `"aws:SourceArn": "arn:aws:ssm:*:account-id:role/aws-service-role/opsdatasync.ssm.amazonaws.com/AWSServiceRoleForSystemsManagerOpsDataSync"`

客户管理型策略示例

您可以创建在自己的 AWS 账户中管理的独立策略。我们将它们称作客户托管策略。随后可以将这些策略附加到您 AWS 账户中的多个主要委托人实体。将策略附加到主体实体时，便向实体授予了策略中定义的权限。有关更多信息，请参阅 IAM 用户指南中的客户托管式策略示例。

以下用户策略示例授予执行各种 Systems Manager 操作的权限。可以使用它们限制对您的 IAM 实体（用户和角色）的 Systems Manager 访问。在 Systems Manager API、AWS 开发工具包或 AWS CLI 中执行操作时，这些策略将会发挥作用。对于使用控制台的用户，需要授予特定于控制台的其他权限。有关更多信息，请参阅 示例：使用 Systems Manager 控制台的权限。

> **Note**
>
> 所有示例都使用 美国西部（俄勒冈）区域 (us-west-2) 和虚构的账户 ID。不应在 AWS 公有文档（以 `AWS-*` 开头的文档）的 Amazon Resource Name (ARN) 中指定账户 ID。

示例

- 示例 1：允许用户在单个区域中执行 Systems Manager 操作
- 示例 2：允许用户列出某个区域的文档

示例 1：允许用户在单个区域中执行 Systems Manager 操作

以下示例授予在美国东部（俄亥俄）区域（us-east-2）中执行 Systems Manager 操作的权限。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:*"
            ],
            "Resource": [
                "arn:aws:ssm:us-east-2:111122223333:*"
            ]
        }
    ]
}
```

示例 2：允许用户列出某个区域的文档

以下示例授予列出美国东部（俄亥俄）区域（us-east-2）中所有以 **Update** 开头的文档名称的权限。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:ListDocuments"
            ],
            "Resource": [
                "arn:aws:ssm:us-east-2:111122223333:document/Update*"
            ]
        }
    ]
}
```

示例 3：允许用户使用特定 SSM 文档在特定节点上运行命令

以下示例 IAM policy 允许用户在美国东部（俄亥俄州）区域（us-east-2）执行以下操作：

- 列出 Systems Manager 文档（SSM 文档）和文档版本。
- 查看有关文档的详细信息。
- 使用策略中指定的文档发送命令。文档名称由以下条目确定。

```
arn:aws:ssm:us-east-2:aws-account-ID:document/Systems-Manager-document-name
```

- 将命令发送到三个节点。节点由第二个 Resource 部分中的以下条目确定。

```
"arn:aws:ec2:us-east-2:aws-account-ID:instance/i-02573cafcfEXAMPLE",
"arn:aws:ec2:us-east-2:aws-account-ID:instance/i-0471e04240EXAMPLE",
"arn:aws:ec2:us-east-2:aws-account-ID:instance/i-07782c72faEXAMPLE"
```

- 发送命令后查看有关命令的详细信息。
- 在 Automation（AWS Systems Manager 中的一项工具）中启动和停止工作流。
- 获取有关自动化工作流的信息。

如果您要授予某个用户使用此文档向该用户有权访问的任何节点发送命令的权限，则您可以在 Resource 部分指定与以下类似的条目并删除其他节点条目。以下示例使用美国东部（俄亥俄）区域（us-east-2）。

```
"arn:aws:ec2:us-east-2:*:instance/*"
```

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ssm:ListDocuments",
                "ssm:ListDocumentVersions",
                "ssm:DescribeDocument",
                "ssm:GetDocument",
                "ssm:DescribeInstanceInformation",
                "ssm:DescribeDocumentParameters",
```

```
                "ssm:DescribeInstanceProperties"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": "ssm:SendCommand",
            "Effect": "Allow",
            "Resource": [
                "arn:aws:ec2:us-east-2:111122223333:instance/
i-02573cafcfEXAMPLE",
                "arn:aws:ec2:us-east-2:111122223333:instance/
i-0471e04240EXAMPLE",
                "arn:aws:ec2:us-east-2:111122223333:instance/
i-07782c72faEXAMPLE",

                "arn:aws:ssm:us-east-2:111122223333:document/Systems-Manager-
document-name"
            ]
        },
        {
            "Action": [
                "ssm:CancelCommand",
                "ssm:ListCommands",
                "ssm:ListCommandInvocations"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": "ec2:DescribeInstanceStatus",
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": "ssm:StartAutomationExecution",
            "Effect": "Allow",
            "Resource": [
                "arn:aws:ssm:us-east-2:111122223333:automation-definition/*"
            ]
        },
        {
            "Action": "ssm:DescribeAutomationExecutions",
            "Effect": "Allow",
```

```
            "Resource": [
                "*"
            ]
        },
        {
            "Action": [
                "ssm:StopAutomationExecution",
                "ssm:GetAutomationExecution"
            ],
            "Effect": "Allow",
            "Resource": [
                "*"
            ]
        }
    ]
}
```

基于标签查看 Systems Manager 文档

您可以在基于身份的策略中使用条件，以便基于标签控制对 Systems Manager 资源的访问。此示例显示如何创建策略以允许查看 SSM 文档。但是，仅当文档标签 `Owner` 的值为该用户的用户名时，才能授予此权限。此策略还授予在控制台上完成此操作的必要权限。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListDocumentsInConsole",
            "Effect": "Allow",
            "Action": "ssm:ListDocuments",
            "Resource": "*"
        },
        {
            "Sid": "ViewDocumentIfOwner",
            "Effect": "Allow",
            "Action": "ssm:GetDocument",
            "Resource": "arn:aws:ssm:*:*:document/*",
            "Condition": {
```

```
                "StringEquals": {"ssm:ResourceTag/Owner": "${aws:username}"}
            }
        }
    ]
}
```

您可以将此策略附加到您账户中的用户。如果名为 `richard-roe` 的用户尝试查看 Systems Manager 文档，则必须将该文档标记为 `Owner=richard-roe` 或 `owner=richard-roe`。否则，他们将被拒绝访问。条件标签密钥 `Owner` 与 `Owner` 和 `owner` 匹配，因为条件密钥名称不区分大小写。有关更多信息，请参阅《IAM 用户指南》中的 IAM JSON 策略元素：条件。

适用于 AWS Systems Manager 的 AWS 托管式策略

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

请记住，AWS 托管策略可能不会为您的特定使用场景授予最低权限，因为它们可供所有 AWS 客户使用。我们建议通过定义特定于您的使用场景的客户管理型策略来进一步减少权限。

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

有关更多信息，请参阅《IAM 用户指南》中的 AWS 托管式策略。

主题

- AWS 托管式策略：AmazonSSMServiceRolePolicy
- AWS 托管策略：AmazonSSMAutomationRole
- AWS 托管式策略：AmazonSSMReadOnlyAccess
- AWS 托管式策略：AWSSystemsManagerOpsDataSyncServiceRolePolicy
- AWS 托管式策略：AmazonSSMManagedEC2InstanceDefaultPolicy
- AWS 托管式策略：SSMQuickSetupRolePolicy
- AWS 托管式策略：AWSQuickSetupDeploymentRolePolicy
- AWS 托管式策略：AWSQuickSetupPatchPolicyDeploymentRolePolicy

- AWS 托管式策略：AWSQuickSetupPatchPolicyBaselineAccess
- AWS 托管式策略：AWSSystemsManagerEnableExplorerExecutionPolicy
- AWS 托管式策略：AWSSystemsManagerEnableConfigRecordingExecutionPolicy
- AWS 托管式策略：AWSQuickSetupDevOpsGuruPermissionsBoundary
- AWS 托管式策略：AWSQuickSetupDistributorPermissionsBoundary
- AWS 托管式策略：AWSQuickSetupSSMHostMgmtPermissionsBoundary
- AWS 托管式策略：AWSQuickSetupPatchPolicyPermissionsBoundary
- AWS 托管式策略：AWSQuickSetupSchedulerPermissionsBoundary
- AWS 托管式策略：AWSQuickSetupCFGCPacksPermissionsBoundary
- AWS 托管式策略：AWS-SSM-DiagnosisAutomation-AdministrationRolePolicy
- AWS 托管式策略：AWS-SSM-DiagnosisAutomation-ExecutionRolePolicy
- AWS 托管式策略：AWS-SSM-RemediationAutomation-AdministrationRolePolicy
- AWS 托管式策略：AWS-SSM-RemediationAutomation-ExecutionRolePolicy
- AWS 托管式策略：AWSQuickSetupSSMManageResourcesExecutionPolicy
- AWS 托管式策略：AWSQuickSetupSSMLifecycleManagementExecutionPolicy
- AWS 托管式策略：AWSQuickSetupSSMDeploymentRolePolicy
- AWS 托管式策略：AWSQuickSetupSSMDeploymentS3BucketRolePolicy
- AWS 托管式策略：AWSQuickSetupEnableDHMCExecutionPolicy
- AWS 托管式策略：AWSQuickSetupEnableAREXExecutionPolicy
- AWS 托管式策略：AWSQuickSetupManagedInstanceProfileExecutionPolicy
- AWS 托管式策略：AWSQuickSetupFullAccess
- AWS 托管式策略：AWSQuickSetupReadOnlyAccess
- AWS 托管策略：AWSQuickSetupManageJITNAResourcesExecutionPolicy
- AWS 托管策略：AWSQuickSetupJITNADeploymentRolePolicy
- AWS 托管策略：AWSSystemsManagerJustInTimeAccessServicePolicy
- AWS 托管策略：AWSSystemsManagerJustInTimeAccessTokenPolicy
- AWS 托管策略：AWSSystemsManagerJustInTimeAccessTokenSessionPolicy
- AWS 托管策略：AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy
- AWS 托管策略：AWSSystemsManagerNotificationsServicePolicy
- AWS 托管式策略：AWS-SSM-Automation-DiagnosisBucketPolicy

- AWS 托管式策略：AWS-SSM-RemediationAutomation-OperationalAccountAdministrationRolePolicy
- AWS 托管式策略：AWS-SSM-DiagnosisAutomation-OperationalAccountAdministrationRolePolicy
- Systems Manager 更新了 AWS 托管式策略
- Systems Manager 的其他托管策略

AWS 托管式策略：AmazonSSMServiceRolePolicy

此策略提供对由 AWS Systems Manager 管理或在 Systems Manager 操作中使用的许多 AWS 资源的访问权限。

您不能将 AmazonSSMServiceRolePolicy 附加到自己的 AWS Identity and Access Management（IAM）实体。此附加到服务相关角色的策略允许 AWS Systems Manager 代表您执行操作。有关更多信息，请参阅 使用角色来收集清单并查看 OpsData。

权限详细信息

该策略包含以下权限。

- ssm：允许主体启动和分步执行 Run Command 和自动化；检索有关 Run Command 和自动化操作的信息；检索有关 Parameter Store 参数 Change Calendar 日历的信息；更新和检索有关 OpsCenter 资源的 Systems Manager 服务设置的信息；读取有关已应用于资源的标签的信息。
- cloudformation：允许主体检索有关堆栈集操作和堆栈集实例的信息，并删除资源 arn:aws:cloudformation:*:*:stackset/AWS-QuickSetup-SSM*:* 上的堆栈集。允许主体删除与以下资源关联的堆栈实例：

```
arn:aws:cloudformation:*:*:stackset/AWS-QuickSetup-SSM*:*
arn:aws:cloudformation:*:*:stackset-target/AWS-QuickSetup-SSM*:*
arn:aws:cloudformation:*:*:type/resource/*
```

- cloudwatch：允许主体检索有关 Amazon CloudWatch 警报的信息。
- compute-optimizer：允许主体检索账户向 AWS Compute Optimizer 服务的注册（选择加入）状态，并检索满足特定一组规定要求的 Amazon EC2 实例的建议。
- config：允许主体检索 AWS Config 中的信息修复配置和配置记录器，并确定指定的 AWS Config 规则和 AWS 资源是否合规。
- events：允许主体检索有关 EventBridge 规则的信息；专门为 Systems Manager 服务 (ssm.amazonaws.com) 创建 EventBridge 规则和目标；以及删除资源 arn:aws:events:*:*:rule/SSMExplorerManagedRule 的规则和目标。

- ec2：允许主体检索有关 Amazon EC2 实例的信息。
- iam：允许主体传递 Systems Manager 服务 (ssm.amazonaws.com) 的角色权限。
- lambda：允许主体调用专门配置为供 Systems Manager 使用的 Lambda 函数。
- resource-explorer-2：允许主体检索有关 EC2 实例的数据，以确定每个实例当前是否由 Systems Manager 管理。

  允许对 arn:aws:resource-explorer-2:*:*:managed-view/AWSManagedViewForSSM* 资源执行操作 resource-explorer-2:CreateManagedView。
- resource-groups：允许主体从属于某资源组的资源的 AWS Resource Groups中检索列表资源组及其成员。
- securityhub：允许主体检索有关当前账户中的 AWS Security Hub 中心资源的信息。
- states：允许主体启动和检索专门配置为供 Systems Manager 使用的 AWS Step Functions 的信息。
- support：允许主体检索有关 AWS Trusted Advisor 中的检查和案例的信息。
- tag：允许主体检索有关位于账户的指定 AWS 区域的所有已标记或之前已标记的资源的信息。

要查看有关策略（包括 JSON 策略文档的最新版本）的更多信息，请参阅《AWS 托管式策略参考指南》中的 AmazonSSMServiceRolePolicy。

AWS 托管策略：AmazonSSMAutomationRole

您可以将 AmazonSSMAutomationRole 策略附加到 IAM 身份。本策略为 AWS Systems Manager 自动化服务提供权限，以运行在自动化运行手册中定义的活动。

权限详细信息

该策略包含以下权限。

- lambda：允许主体调用名字以“Automation”开头的 Lambda 函数。这是自动化运行手册在其工作流程中执行 Lambda 函数所必需的。
- ec2：允许主体执行各种 Amazon EC2 操作，包括创建、复制和注销映像；管理快照；启动、运行、停止和终止实例；管理实例状态；以及创建、删除和描述标签。这些权限使自动化运行手册能够在执行期间管理 Amazon EC2 资源。
- cloudformation：允许主体创建、描述、更新和删除 AWS CloudFormation 堆栈。这使自动化运行手册能够通过 CloudFormation 管理基础设施即代码。

- ssm：允许主体使用所有 Systems Manager 操作。这种全面的访问权限是自动化运行手册与所有 Systems Manager 功能进行交互所必需的。
- sns：允许主体向名字以“Automation”开头的 Amazon SNS 主题发布消息。这使自动化运行手册能够在执行期间发送通知。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS 托管式策略参考指南》中的 AmazonSSMAutomationRole。

AWS 托管式策略：AmazonSSMReadOnlyAccess

您可以将 AmazonSSMReadOnlyAccess 策略附加到 IAM 身份。此策略授予对 AWS Systems Manager API 操作的只读访问权限，包括 Describe*、Get* 和 List*。

要查看有关策略（包括 JSON 策略文档的最新版本）的更多信息，请参阅《AWS 托管式策略参考指南》中的 AmazonSSMReadOnlyAccess。

AWS 托管式策略：AWSSystemsManagerOpsDataSyncServiceRolePolicy

您不能将 AWSSystemsManagerOpsDataSyncServiceRolePolicy 附加到自己的 IAM 实体。此附加到服务相关角色的策略允许 Systems Manager 代表您执行操作。有关更多信息，请参阅 使用角色为 Explorer 创建 OpsData 和 OpsItems。

AWSSystemsManagerOpsDataSyncServiceRolePolicy 允许 AWSServiceRoleForSystemsManagerOpsDataSync 服务相关角色创建和更新来自 AWS Security Hub 调查结果的 OpsItems 及 OpsData。

除非另有说明，否则策略允许 Systems Manager 对所有相关资源（"Resource": "*"）完成以下操作：

- ssm:GetOpsItem [1]
- ssm:UpdateOpsItem [1]
- ssm:CreateOpsItem
- ssm:AddTagsToResource [2]
- ssm:UpdateServiceSetting [3]
- ssm:GetServiceSetting [3]
- securityhub:GetFindings
- securityhub:GetFindings

- securityhub:BatchUpdateFindings [4]

[1] 根据 Systems Manager 服务的以下条件，仅允许执行 ssm:GetOpsItem 和 ssm:UpdateOpsItem 操作的权限。

```
"Condition": {
    "StringEquals": {
        "aws:ResourceTag/ExplorerSecurityHubOpsItem": "true"
    }
}
```

[2] 仅允许对以下资源执行 ssm:AddTagsToResource 操作的权限。

```
arn:aws:ssm:*:*:opsitem/*
```

[3] 仅允许对以下资源执行 ssm:UpdateServiceSetting 和 ssm:GetServiceSetting 操作的权限。

```
arn:aws:ssm:*:*:servicesetting/ssm/opsitem/*
arn:aws:ssm:*:*:servicesetting/ssm/opsdata/*
```

[4] 根据 Systems Manager 服务的以下条件，仅拒绝执行 securityhub:BatchUpdateFindings 的权限。

```
{
   "Effect": "Deny",
   "Action": "securityhub:BatchUpdateFindings",
   "Resource": "*",
   "Condition": {
    "StringEquals": {
     "securityhub:ASFFSyntaxPath/Workflow.Status": "SUPPRESSED"
    }
   }
  },
  {
   "Effect": "Deny",
   "Action": "securityhub:BatchUpdateFindings",
   "Resource": "*",
   "Condition": {
    "Null": {
     "securityhub:ASFFSyntaxPath/Confidence": false
```

```
    }
   }
  },
  {
   "Effect": "Deny",
   "Action": "securityhub:BatchUpdateFindings",
   "Resource": "*",
   "Condition": {
    "Null": {
     "securityhub:ASFFSyntaxPath/Criticality": false
    }
   }
  },
  {
   "Effect": "Deny",
   "Action": "securityhub:BatchUpdateFindings",
   "Resource": "*",
   "Condition": {
    "Null": {
     "securityhub:ASFFSyntaxPath/Note.Text": false
    }
   }
  },
  {
   "Effect": "Deny",
   "Action": "securityhub:BatchUpdateFindings",
   "Resource": "*",
   "Condition": {
    "Null": {
     "securityhub:ASFFSyntaxPath/Note.UpdatedBy": false
    }
   }
  },
  {
   "Effect": "Deny",
   "Action": "securityhub:BatchUpdateFindings",
   "Resource": "*",
   "Condition": {
    "Null": {
     "securityhub:ASFFSyntaxPath/RelatedFindings": false
    }
   }
  },
  {
```

```
    "Effect": "Deny",
    "Action": "securityhub:BatchUpdateFindings",
    "Resource": "*",
    "Condition": {
     "Null": {
      "securityhub:ASFFSyntaxPath/Types": false
     }
    }
   },
   {
    "Effect": "Deny",
    "Action": "securityhub:BatchUpdateFindings",
    "Resource": "*",
    "Condition": {
     "Null": {
      "securityhub:ASFFSyntaxPath/UserDefinedFields.key": false
     }
    }
   },
   {
    "Effect": "Deny",
    "Action": "securityhub:BatchUpdateFindings",
    "Resource": "*",
    "Condition": {
     "Null": {
      "securityhub:ASFFSyntaxPath/UserDefinedFields.value": false
     }
    }
   },
   {
    "Effect": "Deny",
    "Action": "securityhub:BatchUpdateFindings",
    "Resource": "*",
    "Condition": {
     "Null": {
      "securityhub:ASFFSyntaxPath/VerificationState": false
     }
    }
```

要查看有关策略（包括 JSON 策略文档的最新版本）的更多信息，请参阅《AWS 托管式策略参考指南》中的 AWSSystemsManagerOpsDataSyncServiceRolePolicy。

AWS 托管式策略：AmazonSSMManagedEC2InstanceDefaultPolicy

对于想要获得 Systems Manager 功能使用权限的 Amazon EC2 实例，您只应将 `AmazonSSMManagedEC2InstanceDefaultPolicy` 附加到 IAM 角色。您不应将该角色附加到其他 IAM 实体（例如 IAM 用户和 IAM 组）或用于其他目的的 IAM 角色。有关更多信息，请参阅 使用默认主机管理配置自动管理 EC2 实例。

此策略授予的权限允许您 Amazon EC2 实例上的 SSM Agent 与云中的 Systems Manager 服务进行通信以执行各种任务。它还为提供授权令牌的两个服务授予权限，以确保在正确的实例上执行操作。

权限详细信息

该策略包含以下权限。

- `ssm` – 允许主体检索文档、使用 Run Command 执行命令、使用 Session Manager 建立会话、收集实例清单以及使用 Patch Manager 扫描补丁和补丁合规性的权限。
- `ssmmessages` – 允许主体针对每个实例访问由 Amazon Message Gateway Service 创建的个性化授权令牌。Systems Manager 根据 API 操作中提供的实例 Amazon 资源名称（ARN）验证个性化授权令牌。此访问权限是确保 SSM Agent 在正确的实例上执行 API 操作的必要条件。
- `ec2messages` – 允许主体针对每个实例访问由 Amazon Message Delivery Service 创建的个性化授权令牌。Systems Manager 根据 API 操作中提供的实例 Amazon 资源名称（ARN）验证个性化授权令牌。此访问权限是确保 SSM Agent 在正确的实例上执行 API 操作的必要条件。

有关 `ssmmessages` 和 `ec2messages` 端点的相关信息，包括两者之间的区别，请参阅 与代理相关的 API 操作（`ssmmessages` 和 `ec2messages` 端点）。

要查看有关策略（包括 JSON 策略文档的最新版本）的更多信息，请参阅《AWS 托管式策略参考指南》中的 AmazonSSMManagedEC2InstanceDefaultPolicy。

AWS 托管式策略：SSMQuickSetupRolePolicy

您无法将 SSMQuickSetupRolePolicy 附加至您的 IAM 实体。此策略附加至服务相关角色，允许 Systems Manager 代表您执行操作。有关更多信息，请参阅 使用角色来维护已配置 Quick Setup 的资源运行状况和一致性。

此策略授予只读权限，允许 Systems Manager 检查配置运行状况，确保参数和已配置资源的一致使用，并在检测到偏差时修复资源。它还授予创建服务相关角色的管理权限。

权限详细信息

该策略包含以下权限。

- ssm：允许主体读取 Systems Manager 中的资源数据同步和 SSM 文档信息，包括在委派管理员账户中。这是必需的，这样 Quick Setup 才能确定已配置资源的预期状态。
- organizations – 允许主体读取属于组织的成员账户的信息，如 AWS Organizations 中所配置。这是必需的，这样 Quick Setup 才能识别组织中要执行资源运行状况检查的所有账户。
- cloudformation – 允许主体从 AWS CloudFormation 中读取信息。这是必需的，这样 Quick Setup 才能收集有关用于管理资源状态和 CloudFormation 堆栈集操作的 AWS CloudFormation 堆栈的数据。

要查看有关策略（包括 JSON 策略文档的最新版本）的更多信息，请参阅《AWS 托管式策略参考指南》中的 SSMQuickSetupRolePolicy。

AWS 托管式策略：AWSQuickSetupDeploymentRolePolicy

托管式策略 AWSQuickSetupDeploymentRolePolicy 支持多种 Quick Setup 配置类型。这些配置类型创建 IAM 角色和自动化，以配置常用 Amazon Web Services 服务和功能，并提供建议的最佳实践。

您可以将 AWSQuickSetupDeploymentRolePolicy 附加到 IAM 实体。

此策略授予创建与以下 Quick Setup 配置关联的资源所需的管理权限：

- 使用 Quick Setup 设置 Amazon EC2 主机管理
- 使用 Quick Setup 创建 AWS Config 配置记录器
- 使用 Quick Setup 部署 AWS Config 一致性包
- 使用 Quick Setup 设置 DevOps Guru
- 使用 Quick Setup 部署 Distributor 软件包
- 按计划使用 Quick Setup 自动停止和启动 EC2 实例

权限详细信息

该策略包含以下权限。

- iam – 允许主体管理和删除自动化配置任务所需的 IAM 角色；以及管理自动化角色策略。
- cloudformation – 允许主体创建和管理堆栈集。
- config – 允许主体创建、托管和删除一致性包。

- `events` – 允许主体创建、更新和删除计划操作的事件规则。
- `resource-groups` – 允许主体检索与 Quick Setup 配置所针对的资源组关联的资源查询。
- `ssm` – 允许主体创建应用 Quick Setup 配置的自动化运行手册和关联。

要查看有关策略（包括 JSON 策略文档的最新版本）的更多信息，请参阅《AWS 托管式策略参考指南》中的 AWSQuickSetupDeploymentRolePolicy。

AWS 托管式策略：AWSQuickSetupPatchPolicyDeploymentRolePolicy

托管式策略 `AWSQuickSetupPatchPolicyDeploymentRolePolicy` 支持 使用 Quick Setup 补丁策略为组织中的实例配置修补 Quick Setup 类型。此配置类型有助于自动修补单个账户或整个组织中的应用程序和节点。

您可以将 `AWSQuickSetupPatchPolicyDeploymentRolePolicy` 附加到 IAM 实体。Systems Manager 还会将此策略附加到服务角色，以允许 Systems Manager 代表您执行操作。

此策略授予管理权限，允许 Quick Setup 创建与补丁策略配置关联的资源。

权限详细信息

该策略包含以下权限。

- `iam` – 允许主体管理和删除自动化配置任务所需的 IAM 角色；以及管理自动化角色策略。
- `cloudformation` – 允许主体读取 AWS CloudFormation 堆栈信息；并控制由 Quick Setup 使用 AWS CloudFormation 堆栈集创建的 AWS CloudFormation 堆栈。
- `ssm` – 允许主体创建、更新、读取和删除配置任务所需的自动化运行手册；以及创建、更新和删除 State Manager 关联。

- `resource-groups` – 允许主体检索与 Quick Setup 配置所针对的资源组关联的资源查询。

- `s3` – 允许主体列出 Amazon S3 存储桶；以及管理用于存储补丁策略访问日志的存储桶。
- `lambda` – 允许主体管理 AWS Lambda 修复功能，将配置保持在正确的状态。
- `logs` – 允许主体为 Lambda 配置资源描述和管理日志组。

要查看有关策略（包括 JSON 策略文档的最新版本）的更多信息，请参阅《AWS 托管式策略参考指南》中的 AWSQuickSetupPatchPolicyDeploymentRolePolicy。

AWS 托管式策略：AWSQuickSetupPatchPolicyBaselineAccess

托管式策略 AWSQuickSetupPatchPolicyBaselineAccess 支持 使用 Quick Setup 补丁策略为组织中的实例配置修补 Quick Setup 类型。此配置类型有助于自动修补单个账户或整个组织中的应用程序和节点。

您可以将 AWSQuickSetupPatchPolicyBaselineAccess 附加到 IAM 实体。Systems Manager 还会将此策略附加到服务角色，以允许 Systems Manager 代表您执行操作。

此策略提供只读权限，允许访问由当前 AWS 账户 或组织中的管理员使用 Quick Setup 配置的补丁基准。补丁基准存储在 Amazon S3 存储桶中，可用于修补单个账户或整个组织中的实例。

权限详细信息

此策略包含以下权限。

- s3 – 允许主体读取 Amazon S3 存储桶中存储的补丁基准覆盖。

要查看有关策略（包括 JSON 策略文档的最新版本）的更多信息，请参阅《AWS 托管式策略参考指南》中的 AWSQuickSetupPatchPolicyBaselineAccess。

AWS 托管式策略：**AWSSystemsManagerEnableExplorerExecutionPolicy**

托管式策略 AWSSystemsManagerEnableExplorerExecutionPolicy 支持启用 Explorer（AWS Systems Manager 中的一项工具）。

您可以将 AWSSystemsManagerEnableExplorerExecutionPolicy 附加到 IAM 实体。Systems Manager 还会将此策略附加到服务角色，以允许 Systems Manager 代表您执行操作。

此策略授予管理权限，以便启用 Explorer。这包括更新相关 Systems Manager 服务设置和为 Systems Manager 创建服务相关角色的权限。

权限详细信息

该策略包含以下权限。

- config – 允许主体通过提供对配置记录器详细信息的只读访问权限来帮助启用 Explorer。
- iam – 允许主体帮助启用 Explorer。
- ssm – 允许主体启动启用 Explorer 的自动化工作流。

要查看有关策略（包括 JSON 策略文档的最新版本）的更多信息，请参阅《AWS 托管是策略参考指南》中的 AWSSystemsManagerEnableExplorerExecutionPolicy。

AWS 托管式策略：AWSSystemsManagerEnableConfigRecordingExecutionPolicy

托管式策略 AWSSystemsManagerEnableConfigRecordingExecutionPolicy 支持 使用 Quick Setup 创建 AWS Config 配置记录器 Quick Setup 配置类型。使用此配置类型，Quick Setup 可以跟踪和记录对为 AWS Config 选择的 AWS 资源类型的更改。它还可让 Quick Setup 为记录的数据配置传送和通知选项。

您可以将 AWSSystemsManagerEnableConfigRecordingExecutionPolicy 附加到 IAM 实体。Systems Manager 还会将此策略附加到服务角色，以允许 Systems Manager 代表您执行操作。

此策略授予管理权限，允许Quick Setup 启用和配置 AWS Config 配置记录。

权限详细信息

该策略包含以下权限。

- s3 – 允许主体创建和配置 Amazon S3 存储桶以交付配置记录。
- sns - 允许主体列出并创建 Amazon SNS 主题。
- config – 允许主体配置和启动配置记录器；并帮助启用 Explorer。
- iam – 允许主体为 AWS Config 创建、获取和传递服务相关角色；为 Systems Manager 创建服务相关角色；以及帮助启用 Explorer。
- ssm – 允许主体启动启用 Explorer 的自动化工作流。
- compute-optimizer – 允许主体通过提供只读访问权限来帮助启用 Explorer，以确定资源是否注册到了 AWS Compute Optimizer。
- support – 允许主体通过提供只读访问权限来帮助启用 Explorer，以确定资源是否注册到了 AWS Compute Optimizer。

要查看有关策略（包括 JSON 策略文档的最新版本）的更多信息，请参阅《AWS 托管式策略参考指南》中的 AWSSystemsManagerEnableConfigRecordingExecutionPolicy。

AWS 托管式策略：AWSQuickSetupDevOpsGuruPermissionsBoundary

> **Note**
>
> 此策略是权限边界。权限边界设置基于身份的策略可以授予 IAM 实体的最大权限。您不应自行使用和附加 Quick Setup 权限边界策略。Quick Setup 权限边界策略只能附加到 Quick Setup 托管角色。有关权限边界的更多信息，请参阅《IAM 用户指南》中的 IAM 实体的权限边界。

托管式策略 `AWSQuickSetupDevOpsGuruPermissionsBoundary` 支持 使用 Quick Setup 设置 DevOps Guru 类型。配置类型支持由机器学习助力的 Amazon DevOps Guru。DevOps Guru 服务可以帮助提高应用程序的运行性能和可用性。

当您使用 Quick Setup 创建 `AWSQuickSetupDevOpsGuruPermissionsBoundary` 配置时，系统会将此权限边界应用于部署配置时创建的 IAM 角色。权限边界限制 Quick Setup 创建的角色的范围。

此策略授予管理权限，允许 Quick Setup 启用和配置 Amazon DevOps Guru。

权限详细信息

该策略包含以下权限。

- `iam` – 允许主体为 DevOps Guru 和 Systems Manager 创建服务相关角色；以及列出帮助启用 Explorer 的角色。
- `cloudformation` – 允许主体列出并描述 AWS CloudFormation 堆栈。
- `sns` – 允许主体列出并创建 Amazon SNS 主题。
- `devops-guru` – 允许主体配置 DevOps Guru；以及添加通知渠道。
- `config` – 允许主体通过提供对配置记录器详细信息的只读访问权限来帮助启用 Explorer。
- `ssm` – 允许主体启动启用 Explorer 的自动化工作流；以及读取和更新 Explorer 服务设置。
- `compute-optimizer` – 允许主体通过提供只读访问权限来帮助启用 Explorer，以确定资源是否注册到了 AWS Compute Optimizer。
- `support` – 允许主体通过提供只读访问权限来帮助启用 Explorer，以确定资源是否注册到了 AWS Compute Optimizer。

要查看有关策略（包括 JSON 策略文档的最新版本）的更多信息，请参阅《AWS 托管式策略参考指南》中的 AWSQuickSetupDevOpsGuruPermissionsBoundary。

AWS 托管式策略：AWSQuickSetupDistributorPermissionsBoundary

> **Note**
>
> 此策略是权限边界。权限边界设置基于身份的策略可以授予 IAM 实体的最大权限。您不应自行使用和附加 Quick Setup 权限边界策略。Quick Setup 权限边界策略只能附加到 Quick Setup 托管角色。有关权限边界的更多信息，请参阅《IAM 用户指南》中的 IAM 实体的权限边界。

托管式策略 `AWSQuickSetupDistributorPermissionsBoundary` 支持 使用 Quick Setup 部署 Distributor 软件包 Quick Setup 配置类型。此配置类型有助于使用 Distributor（AWS Systems Manager 中的一项工具）将软件包（例如代理）分发至您的 Amazon Elastic Compute Cloud（Amazon EC2）实例。

当您使用 Quick Setup 创建 `AWSQuickSetupDistributorPermissionsBoundary` 配置时，系统会将此权限边界应用于部署配置时创建的 IAM 角色。权限边界限制 Quick Setup 创建的角色的范围。

该策略授予管理权限，允许 Quick Setup 使用 Distributor 将软件包（例如代理）分发至您的 Amazon EC2 实例。

权限详细信息

该策略包含以下权限。

- `iam` – 允许主体获取和传递 Distributor 自动化角色；创建、读取、更新和删除默认实例角色；将默认实例角色传递给 Amazon EC2 和 Systems Manager；将实例管理策略附加到实例角色；为 Systems Manager 创建服务相关角色；向实例配置文件添加默认实例角色；读取有关 IAM 角色和实例配置文件的信息；以及创建默认实例配置文件。
- `ec2` – 允许主体将默认实例配置文件与 EC2 实例关联起来；以及帮助启用 Explorer。
- `ssm` – 允许主体启动配置实例和安装软件包的自动化工作流；帮助启动启用 Explorer 的自动化工作流；以及读取和更新 Explorer 服务设置。
- `config` – 允许主体通过提供对配置记录器详细信息的只读访问权限来帮助启用 Explorer。
- `compute-optimizer` – 允许主体通过提供只读访问权限来帮助启用 Explorer，以确定资源是否注册到了 AWS Compute Optimizer。
- `support` – 允许主体通过提供只读访问权限来帮助启用 Explorer，以确定资源是否注册到了 AWS Compute Optimizer。

要查看有关策略（包括 JSON 策略文档的最新版本）的更多信息，请参阅《AWS 托管式策略参考指南》中的 AWSQuickSetupDistributorPermissionsBoundary。

AWS 托管式策略：AWSQuickSetupSSMHostMgmtPermissionsBoundary

> **Note**
>
> 此策略是权限边界。权限边界设置基于身份的策略可以授予 IAM 实体的最大权限。您不应自行使用和附加 Quick Setup 权限边界策略。Quick Setup 权限边界策略只能附加到 Quick Setup 托管角色。有关权限边界的更多信息，请参阅《IAM 用户指南》中的 IAM 实体的权限边界。

托管式策略 `AWSQuickSetupSSMHostMgmtPermissionsBoundary` 支持 使用 Quick Setup 设置 Amazon EC2 主机管理 Quick Setup 配置类型。此配置类型用于配置 IAM 角色并启用常用的 Systems Manager 工具来安全地管理您的 Amazon EC2 实例。

当您使用 Quick Setup 创建 `AWSQuickSetupSSMHostMgmtPermissionsBoundary` 配置时，系统会将此权限边界应用于部署配置时创建的 IAM 角色。权限边界限制 Quick Setup 创建的角色的范围。

此策略授予管理权限，允许Quick Setup启用和配置安全管理 EC2 实例所需的 Systems Manager 工具。

权限详细信息

该策略包含以下权限。

- `iam` – 允许主体获取服务角色并将其传递给自动化。允许主体创建、读取、更新和删除默认实例角色；将默认实例角色传递给 Amazon EC2 和 Systems Manager；将实例管理策略附加到实例角色；为 Systems Manager 创建服务相关角色；向实例配置文件添加默认实例角色；读取有关 IAM 角色和实例配置文件的信息；以及创建默认实例配置文件。
- `ec2` – 允许主体将默认实例配置文件与 EC2 实例关联和取消关联。
- `ssm`：允许主体启动会启用 Explorer 的自动化工作流；读取和更新 Explorer 服务设置；配置实例；以及在实例上启用 Systems Manager 工具。
- `compute-optimizer` – 允许主体通过提供只读访问权限来帮助启用 Explorer，以确定资源是否注册到了 AWS Compute Optimizer。
- `support` – 允许主体通过提供只读访问权限来帮助启用 Explorer，以确定资源是否注册到了 AWS Compute Optimizer。

要查看有关策略（包括 JSON 策略文档的最新版本）的更多信息，请参阅《AWS 托管式策略参考指南》中的 AWSQuickSetupSSMHostMgmtPermissionsBoundary。

AWS 托管式策略：AWSQuickSetupPatchPolicyPermissionsBoundary

> **Note**
>
> 此策略是权限边界。权限边界设置基于身份的策略可以授予 IAM 实体的最大权限。您不应自行使用和附加 Quick Setup 权限边界策略。Quick Setup 权限边界策略只能附加到 Quick Setup 托管角色。有关权限边界的更多信息，请参阅《IAM 用户指南》中的 IAM 实体的权限边界。

托管式策略 `AWSQuickSetupPatchPolicyPermissionsBoundary` 支持 使用 Quick Setup 补丁策略为组织中的实例配置修补 Quick Setup 类型。此配置类型有助于自动修补单个账户或整个组织中的应用程序和节点。

当您使用 Quick Setup 创建 `AWSQuickSetupPatchPolicyPermissionsBoundary` 配置时，系统会将此权限边界应用于部署配置时创建的 IAM 角色。权限边界限制 Quick Setup 创建的角色的范围。

此策略授予管理权限，允许Quick Setup在Patch Manager（AWS Systems Manager 中的一项工具）中启用和配置补丁策略。

权限详细信息

该策略包含以下权限。

- `iam` – 允许主体获得 Patch Manager 自动化角色；将自动化角色传递给 Patch Manager 修补操作；创建默认实例角色；将默认实例角色传递给 Amazon EC2 和 Systems Manager；`AmazonSSMRoleForInstancesQuickSetup`；将选定的 AWS 托管式策略附加到实例角色；为 Systems Manager 创建服务相关角色；向实例配置文件添加默认实例角色；读取有关实例配置文件和角色的信息；创建默认实例配置文件；以及标记有权读取补丁基准覆盖的角色。
- `ssm` – 允许主体更新由 Systems Manager 管理的实例角色；管理由 Quick Setup 中创建的 Patch Manager 补丁策略创建的关联；标记补丁策略配置所针对的实例；读取有关实例和修补状态的信息；启动配置、启用和修复实例补丁的自动化工作流；启动启用 Explorer 的自动化工作流；帮助启用 Explorer；以及读取和更新 Explorer 服务设置。
- `ec2` – 允许主体将默认实例配置文件与 EC2 实例关联和取消关联；标记补丁策略配置所针对的实例；标记补丁策略配置所针对的实例；以及帮助启用 Explorer。
- `s3` – 允许主体创建和配置 S3 存储桶以存储补丁基准覆盖。

- `lambda` – 允许主体调用配置补丁的 AWS Lambda 函数，并在删除 Quick Setup 补丁策略配置后执行清理操作。
- `logs` – 允许主体为 Patch Manager Quick Setup AWS Lambda 函数配置日志记录。
- `config` – 允许主体通过提供对配置记录器详细信息的只读访问权限来帮助启用 Explorer。
- `compute-optimizer` – 允许主体通过提供只读访问权限来帮助启用 Explorer，以确定资源是否注册到了 AWS Compute Optimizer。
- `support` – 允许主体通过提供只读访问权限来帮助启用 Explorer，以确定资源是否注册到了 AWS Compute Optimizer。

要查看有关策略（包括 JSON 策略文档的最新版本）的更多信息，请参阅《AWS 托管式策略参考指南》中的 AWSQuickSetupPatchPolicyPermissionsBoundary。

AWS 托管式策略：AWSQuickSetupSchedulerPermissionsBoundary

> **Note**
>
> 此策略是权限边界。权限边界设置基于身份的策略可以授予 IAM 实体的最大权限。您不应自行使用和附加 Quick Setup 权限边界策略。Quick Setup 权限边界策略只能附加到 Quick Setup 托管角色。有关权限边界的更多信息，请参阅《IAM 用户指南》中的 IAM 实体的权限边界。

托管式策略 `AWSQuickSetupSchedulerPermissionsBoundary` 支持 按计划使用 Quick Setup 自动停止和启动 EC2 实例 Quick Setup 配置类型。此配置类型允许您在指定的时间停止和启动 EC2 实例和其他资源。

当您使用 Quick Setup 创建 `AWSQuickSetupSchedulerPermissionsBoundary` 配置时，系统会将此权限边界应用于部署配置时创建的 IAM 角色。权限边界限制 Quick Setup 创建的角色的范围。

此策略授予管理权限，允许 Quick Setup 在 EC2 实例和其他资源上启用和配置计划操作。

权限详细信息

该策略包含以下权限。

- `iam`：允许主体检索和传递实例管理自动化操作的角色；管理、传递和附加用于 EC2 实例管理的默认实例角色；创建默认实例配置文件；向实例配置文件添加默认实例角色；为 Systems Manager 创建服务相关角色；读取有关 IAM 角色和实例配置文件的信息；将默认实例配置文件与 EC2 实例关联；以及启动自动化工作流以配置实例并在实例上启用 Systems Manager 工具。

- ssm – 允许主体启动启用 Explorer 的自动化工作流；以及读取和更新 Explorer 服务设置。
- ec2 – 允许主体找到目标实例并按计划启动和停止它们。
- config – 允许主体通过提供对配置记录器详细信息的只读访问权限来帮助启用 Explorer。
- compute-optimizer – 允许主体通过提供只读访问权限来帮助启用 Explorer，以确定资源是否注册到了 AWS Compute Optimizer。
- support — 允许主体通过提供对账户 AWS Trusted Advisor 检查的只读访问权限来帮助启用 Explorer。

要查看有关策略（包括 JSON 策略文档的最新版本）的更多信息，请参阅《AWS 托管式策略参考指南》中的 AWSQuickSetupSchedulerPermissionsBoundary。

AWS 托管式策略：AWSQuickSetupCFGCPacksPermissionsBoundary

> **Note**
>
> 此策略是权限边界。权限边界设置基于身份的策略可以授予 IAM 实体的最大权限。您不应自行使用和附加 Quick Setup 权限边界策略。Quick Setup 权限边界策略只能附加到 Quick Setup 托管角色。有关权限边界的更多信息，请参阅《IAM 用户指南》中的 IAM 实体的权限边界。

托管式策略 AWSQuickSetupCFGCPacksPermissionsBoundary 支持 使用 Quick Setup 部署 AWS Config 一致性包 Quick Setup 配置类型。此配置类型部署 AWS Config 一致性包。一致性包是一系列 AWS Config 规则和补救措施，这些规则和补救措施可以作为单个实体进行部署。

当您使用 Quick Setup 创建 AWSQuickSetupCFGCPacksPermissionsBoundary 配置时，系统会将此权限边界应用于部署配置时创建的 IAM 角色。权限边界限制 Quick Setup 创建的角色的范围。

此策略授予管理权限，允许 Quick Setup 部署 AWS Config 一致性包。

权限详细信息

该策略包含以下权限。

- iam – 允许主体为 AWS Config 创建、获取和传递服务相关角色。
- sns – 允许主体在 Amazon SNS 中列出平台应用程序。
- config – 允许主体部署 AWS Config 一致性包；获取一致性包的状态；以及获取有关配置记录器的信息。

- ssm – 允许主体获取有关 SSM 文档和自动化工作流的信息；获取有关资源标签的信息；以及获取相关信息和更新服务设置。
- compute-optimizer – 允许主体获取账户的选择加入状态。
- support – 允许主体获取有关 AWS Trusted Advisor 检查的信息。

要查看有关策略（包括 JSON 策略文档的最新版本）的更多信息，请参阅《AWS 托管式策略参考指南》中的 AWSQuickSetupCFGCPacksPermissionsBoundary。

AWS 托管式策略：AWS-SSM-DiagnosisAutomation-AdministrationRolePolicy

策略 AWS-SSM-DiagnosisAutomation-AdministrationRolePolicy 通过在管理节点的账户和区域中启动自动化工作流，提供诊断与 Systems Manager 服务交互的节点中问题的权限。

您可以将 AWS-SSM-DiagnosisAutomation-AdministrationRolePolicy 附加到您的 IAM 实体。Systems Manager 还会将此策略附加到一个服务角色，以允许 Systems Manager 代表您执行诊断操作。

权限详细信息

该策略包含以下权限。

- ssm：允许主体运行特定的自动化运行手册来诊断节点问题，访问工作流的执行状态，并检索自动化执行详细信息。此策略授予描述自动化执行、描述自动化步骤执行、获取自动化执行详细信息以及开始对诊断相关文档进行自动化执行的权限。
- kms：允许主体在访问 Amazon S3 存储桶中用于诊断操作的加密对象时，使用客户指定的 AWS Key Management Service 密钥进行解密和生成数据密钥。这些权限仅限于标有 SystemsManagerManaged，并通过 Amazon S3 服务按照特定加密环境要求使用的密钥。
- sts：允许主体代入诊断执行角色，以在同一账户中运行自动化运行手册。该权限仅限于具有 AWS-SSM-DiagnosisExecutionRole 命名模式的角色，且包含确保资源账户与主体账户相匹配的条件。
- iam：允许主体将诊断管理角色传递给 Systems Manager 以运行自动化运行手册。该权限仅限于具有 AWS-SSM-DiagnosisAdminRole 命名模式的角色，并且只能传递给 Systems Manager 服务。
- s3：允许主体访问、读取、写入和删除 Amazon S3 存储桶中用于诊断操作的对象。这些权限仅限于具有 do-not-delete-ssm-diagnosis- 命名模式的存储桶，且包含确保操作在相同账户内执行的条件。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS 托管式策略参考指南》中的 AWS-SSM-DiagnosisAutomation-AdministrationRolePolicy。

AWS 托管式策略：AWS-SSM-DiagnosisAutomation-ExecutionRolePolicy

托管式策略 `AWS-SSM-DiagnosisAutomation-ExecutionRolePolicy` 提供在目标 AWS 账户和区域中运行自动化运行手册的管理权限，以诊断与 Systems Manager 服务交互的托管节点的问题。

您可以将 `AWS-SSM-DiagnosisAutomation-ExecutionRolePolicy` 附加到 IAM 实体。Systems Manager 还会将此策略附加到服务角色，以允许 Systems Manager 代表您执行操作。

权限详细信息

该策略包含以下权限。

- `ec2`：允许主体描述 Amazon EC2 和 Amazon VPC 资源及其配置，以诊断 Systems Manager 服务的问题。这包括描述 VPC、VPC 属性、VPC 端点、子网、安全组、实例和互联网网关的权限。
- `ssm`：允许主体运行特定于诊断的自动化运行手册并访问自动化工作流状态和执行元数据。这包括描述自动化步骤执行、描述实例信息、描述自动化执行、获取自动化执行详细信息以及开始对特定的 AWS 非托管 EC2 诊断文档进行自动化执行的权限。
- `kms`：允许主体在访问 Amazon S3 存储桶中用于诊断操作的加密对象时，使用客户指定的 AWS Key Management Service 密钥进行解密和生成数据密钥。这些权限仅限于标有 `SystemsManagerManaged`，并通过 Amazon S3 服务按照特定加密环境要求用于诊断存储桶的密钥。
- `iam`：允许主体将诊断执行角色传递给 Systems Manager 以运行自动化文档。该权限仅限于具有 `AWS-SSM-DiagnosisExecutionRole` 命名模式的角色，并且只能传递给 Systems Manager 服务。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS 托管式策略参考指南》中的 AWS-SSM-DiagnosisAutomation-ExecutionRolePolicy。

AWS 托管式策略：AWS-SSM-RemediationAutomation-AdministrationRolePolicy

此策略 `AWS-SSM-RemediationAutomation-AdministrationRolePolicy` 提供的权限是通过执行自动化文档中定义的活动来修复 Systems Manager 服务问题，这主要用于运行自动化文档。此政策允许在进行节点管理的账户和区域中启动自动化工作流程解决连接和配置问题。

您可以将 AWS-SSM-RemediationAutomation-AdministrationRolePolicy 附加到 IAM 实体。Systems Manager 还会将此策略附加到服务角色，以允许 Systems Manager 代表您执行修复操作。

权限详细信息

该策略包含以下权限。

- ssm：允许主体运行特定的自动化运行手册来修复节点问题，访问工作流的执行状态，并检索自动化执行详细信息。此策略授予描述自动化执行、描述自动化步骤执行、获取自动化执行详细信息以及开始对修复相关文档进行自动化执行的权限。
- kms：允许主体在访问 Amazon S3 存储桶中用于修复操作的加密对象时，使用客户指定的 AWS Key Management Service 密钥进行解密和生成数据密钥。这些权限仅限于标有 SystemsManagerManaged，并通过 Amazon S3 服务按照特定加密环境要求使用的密钥。
- sts：允许主体代入修复执行角色，以在同一账户中运行自动化运行手册。该权限仅限于具有 AWS-SSM-RemediationExecutionRole 命名模式的角色，且包含确保资源账户与主体账户相匹配的条件。
- iam：允许主体将修复管理角色传递给 Systems Manager 以运行自动化运行手册。该权限仅限于具有 AWS-SSM-RemediationAdminRole 命名模式的角色，并且只能传递给 Systems Manager 服务。
- s3：允许主体访问、读取、写入和删除 Amazon S3 存储桶中用于修复操作的对象。这些权限仅限于具有 do-not-delete-ssm-diagnosis- 命名模式的存储桶，且包含确保操作在相同账户内执行的条件。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS 托管式策略参考指南》中的 AWS-SSM-RemediationAutomation-AdministrationRolePolicy。

AWS 托管式策略：AWS-SSM-RemediationAutomation-ExecutionRolePolicy

托管式策略 AWS-SSM-RemediationAutomation-ExecutionRolePolicy 提供在特定目标账户和区域中运行自动化运行手册的权限，以修复与 Systems Manager 服务交互的托管节点的网络和连接问题。此策略支持自动化文档中定义的修复活动，主要用于运行自动化文档来解决连接和配置问题。

可以将 策略附加到您的 IAM 实体。Systems Manager 还会将此策略附加到服务角色，以允许 Systems Manager 代表您执行修复操作。

权限详细信息

该策略包含以下权限。

- ssm：允许主体检索有关自动化执行及其步骤执行的信息，并启动特定的修复自动化运行手册，包括 AWS-OrchestrateUnmanagedEC2Actions 和 AWS-RemediateSSMAgent 文档。此策略授予描述自动化执行、描述自动化步骤执行、获取自动化执行详细信息以及开始对修复相关文档进行自动化执行的权限。
- ec2：允许主体描述和修改 Amazon VPC 网络资源以修复连接问题。这包括：
  - 描述 Amazon VPC 属性、子网、Amazon VPC 端点和安全组。
  - 使用所需标签，为 Systems Manager 服务（ssm、ssmmessages 和 ec2messages）创建 Amazon VPC 端点。
  - 修改 Amazon VPC 属性以启用 DNS 支持和主机名。
  - 创建和管理带有特定标签的安全组，以用于访问 Amazon VPC 端点。
  - 使用相应标签，授权和撤消对 HTTPS 访问的安全组规则。
  - 在资源创建期间，在 Amazon VPC 端点、安全组和安全组规则上创建标签。
- kms：允许主体在访问 Amazon S3 存储桶中用于修复操作的加密对象时，使用客户指定的 AWS Key Management Service 密钥进行解密和生成数据密钥。这些权限仅限于标有 SystemsManagerManaged，并通过 Amazon S3 服务按照特定加密环境要求使用的密钥。
- iam：允许主体将修复执行角色传递给 Systems Manager 以运行自动化运行手册。该权限仅限于具有 AWS-SSM-RemediationExecutionRole 命名模式的角色，并且只能传递给 Systems Manager 服务。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS 托管式策略参考指南》中的 AWS-SSM-RemediationAutomation-ExecutionRolePolicy。

AWS 托管式策略：AWSQuickSetupSSMManageResourcesExecutionPolicy

此策略授予允许Quick Setup运行 AWSQuickSetupType-SSM-SetupResources 自动化运行手册的权限。此运行手册会为Quick Setup关联创建 IAM 角色，而这些角色又由 AWSQuickSetupType-SSM 部署创建。它还授予在Quick Setup删除操作期间清理关联的 Amazon S3 存储桶的权限。

您可以将该策略附加到您的 IAM 实体。Systems Manager 还会将此策略附加到一个服务角色，以允许 Systems Manager 代表您执行操作。

权限详细信息

该策略包含以下权限。

- iam：允许主体列出和管理用于Quick Setup Systems Manager Explorer 操作的 IAM 角色；查看、附加和分离用于Quick Setup和 Systems Manager Explorer 的 IAM 策略。需要这些权限，Quick Setup才能创建其某些配置操作所需的角色。
- s3：允许主体在主体账户中检索有关 Amazon S3 存储桶中对象的信息，并从中删除专门用于Quick Setup配置操作的对象。这是必需的，以便可以移除配置后不再需要的 S3 对象。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS 托管式策略参考指南》中的 AWSQuickSetupSSMManageResourcesExecutionPolicy。

AWS 托管式策略：AWSQuickSetupSSMLifecycleManagementExecutionPolicy

AWSQuickSetupSSMLifecycleManagementExecutionPolicy 策略授予管理权限，允许Quick Setup在 Systems Manager 中的Quick Setup部署期间对生命周期事件运行 AWS CloudFormation 自定义资源。

您可以将此策略附加到您的 IAM 实体。Systems Manager 还会将此策略附加到一个服务角色，以允许 Systems Manager 代表您执行操作。

权限详细信息

该策略包含以下权限。

- ssm：允许主体获取有关自动化执行的信息并启动自动化执行以设置某些Quick Setup操作。
- iam：允许主体从 IAM 传递角色来设置某些Quick Setup资源。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS 托管式策略参考指南》中的 AWSQuickSetupSSMLifecycleManagementExecutionPolicy。

AWS 托管式策略：AWSQuickSetupSSMDeploymentRolePolicy

托管式策略 AWSQuickSetupSSMDeploymentRolePolicy 授予管理权限，允许Quick Setup创建在 Systems Manager 加入过程中使用的资源。

虽然您可以手动将此策略附加到您的 IAM 实体，但不建议这样做。Quick Setup会创建实体来将此策略附加到一个服务角色，从而允许 Systems Manager 代表您执行操作。

此策略与 SSMQuickSetupRolePolicy 策略无关，后者用于为 AWSServiceRoleForSSMQuickSetup 服务相关角色提供权限。

权限详细信息

该策略包含以下权限。

- ssm：允许主体管理使用 AWS CloudFormation 模板和一组特定的 SSM 文档创建的某些资源的关联；管理用于通过 AWS CloudFormation 模板诊断和修复托管节点的角色和角色策略；以及附加和删除Quick Setup生命周期事件的策略
- iam：允许主体标记角色和传递 Systems Manager 服务和 Lambda 服务的角色权限；并传递诊断操作的角色权限。
- lambda：允许主体使用 AWS CloudFormation 模板标记和管理主体账户中的 Quick Setup 生命周期的功能。
- cloudformation – 允许主体从 AWS CloudFormation 中读取信息。这是必需的，这样 Quick Setup 才能收集有关用于管理资源状态和 CloudFormation 堆栈集操作的 AWS CloudFormation 堆栈的数据。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS 托管式策略参考指南》中的 AWSQuickSetupSSMDeploymentRolePolicy。

AWS 托管式策略：AWSQuickSetupSSMDeploymentS3BucketRolePolicy

AWSQuickSetupSSMDeploymentS3BucketRolePolicy 策略授予以下权限：列出账户中的所有 S3 存储桶；以及管理和检索有关通过 AWS CloudFormation 模板管理的主体账户中特定存储桶的信息。

您可以将 AWSQuickSetupSSMDeploymentS3BucketRolePolicy 附加到 IAM 实体。Systems Manager 还会将此策略附加到服务角色，以允许 Systems Manager 代表您执行操作。

权限详细信息

该策略包含以下权限。

- s3：允许主体列出账户中的所有 S3 存储桶；以及管理和检索有关通过 AWS CloudFormation 模板管理的主体账户中特定存储桶的信息。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS 托管式策略参考指南》中的 AWSQuickSetupSSMDeploymentS3BucketRolePolicy。

AWS 托管式策略：AWSQuickSetupEnableDHMCExecutionPolicy

此策略授予管理权限，允许主体运行 AWSQuickSetupType-EnableDHMC 自动化运行手册，从而启用默认主机管理配置。默认主机管理配置设置允许 Systems Manager 自动将 Amazon EC2 实例作

为托管实例进行管理。托管实例是一个配置为与 Systems Manager 一起使用的 EC2 实例。此策略还授予创建 IAM 角色的权限，这些角色在 Systems Manager 服务设置中指定为 SSM Agent 的默认角色。

您可以将 AWSQuickSetupEnableDHMCExecutionPolicy 附加到 IAM 实体。Systems Manager 还会将此策略附加到服务角色，以允许 Systems Manager 代表您执行操作。

权限详细信息

该策略包含以下权限。

- ssm：允许主体更新并获取有关 Systems Manager 服务设置的信息。
- iam：允许主体创建和检索有关Quick Setup操作的 IAM 角色的信息。

要查看有关策略（包括 JSON 策略文档的最新版本）的更多信息，请参阅《AWS 托管式策略参考指南》中的 AWSQuickSetupEnableDHMCExecutionPolicy。

AWS 托管式策略：AWSQuickSetupEnableAREXExecutionPolicy

此策略授予管理权限来允许 Systems Manager 运行 AWSQuickSetupType-EnableAREX 自动化运行手册，从而使 AWS 资源探索器能够与 Systems Manager 一起使用。资源探索器让您能够通过类似于 Internet 搜索引擎的搜索体验来查看您账户中的资源。该策略还会授予管理资源探索器索引和视图的权限。

您可以将 AWSQuickSetupEnableAREXExecutionPolicy 附加到 IAM 实体。Systems Manager 还会将此策略附加到服务角色，以允许 Systems Manager 代表您执行操作。

权限详细信息

该策略包含以下权限。

- iam：允许主体在 AWS Identity and Access Management (IAM) 服务中创建服务相关角色。
- resource-explorer-2：允许主体检索有关资源探索器视图和索引的信息；创建资源探索器视图和索引；更改Quick Setup中显示的索引的索引类型。

要查看有关策略（包括 JSON 策略文档的最新版本）的更多信息，请参阅《AWS 托管式策略参考指南》中的 AWSQuickSetupEnableAREXExecutionPolicy。

AWS 托管式策略：AWSQuickSetupManagedInstanceProfileExecutionPolicy

此策略授予管理权限来允许 Systems Manager 为Quick Setup工具创建默认 IAM 实例配置文件，并将其附加到尚未附加实例配置文件的 Amazon EC2 实例。该策略还授予 Systems Manager 将权限附加到现有实例配置文件的能力。这样做是为了确保 Systems Manager 与 EC2 实例上的 SSM Agent 通信所需的权限就绪。

您可以将 `AWSQuickSetupManagedInstanceProfileExecutionPolicy` 附加到 IAM 实体。Systems Manager 还会将此策略附加到服务角色，以允许 Systems Manager 代表您执行操作。

权限详细信息

该策略包含以下权限。

- `ssm`：允许主体启动与Quick Setup过程关联的自动化工作流。
- `ec2`：允许主体将 IAM 实例配置文件附加到由Quick Setup管理的 EC2 实例。
- `iam`：允许主体从 IAM 创建、更新和检索有关在Quick Setup过程中使用的角色的信息；创建 IAM 实例配置文件；将 `AmazonSSMManagedInstanceCore` 托管式策略附加到 IAM 实例配置文件。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS 托管式策略参考指南》中的 AWSQuickSetupManagedInstanceProfileExecutionPolicy。

AWS 托管式策略：AWSQuickSetupFullAccess

此策略授予管理权限来允许完全访问 AWS Management Console和 AWS SDK 中的 AWS Systems Manager Quick Setup API 操作和数据，以及有限访问Quick Setup操作所需的其他 AWS 服务资源。

您可以将 `AWSQuickSetupFullAccess` 策略附加到 IAM 身份。

权限详细信息

该策略包含以下权限。

- `ssm`：允许主体启用 Explorer；在State Manager中执行资源数据同步操作；以及使用 SSM 命令文档和自动化运行手册执行操作。

  Explorer、State Manager、Documents 和 Automation 都是 Systems Manager 中的工具。
- `cloudformation`：允许主体执行跨 AWS 区域和 AWS 账户预置资源所需的 AWS CloudFormation 操作。
- `ec2`：允许主体选择给定配置的必要参数，并在 AWS Management Console中提供验证。

- `iam`：允许主体创建Quick Setup操作所需的服务角色和服务相关角色。
- `organizations`：允许主体读取 AWS Organizations 组织中的账户状态；检索组织的结构；启用可信访问；以及从管理账户注册委派管理员账户。
- `resource-groups`：允许主体选择给定配置的必要参数，并在 AWS Management Console中提供验证。
- `s3`：允许主体选择给定配置的必要参数，并在 AWS Management Console中提供验证。
- `ssm-quicksetup`：允许主体在Quick Setup中执行只读操作。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS 托管式策略参考指南》中的 AWSQuickSetupFullAccess。

AWS 托管式策略：AWSQuickSetupReadOnlyAccess

此策略授予只读权限，允许主体查看 AWS Systems Manager Quick Setup数据和报告，包括Quick Setup操作所需的其他 AWS 服务资源的信息。

您可以将 `AWSQuickSetupReadOnlyAccess` 策略附加到 IAM 身份。

权限详细信息

该策略包含以下权限。

- `ssm`：允许主体读取 SSM 命令文档和自动化运行手册；以及检索State Manager关联执行的状态。
- `cloudformation`：允许主体启动检索 AWS CloudFormation 部署状态所需的操作。
- `organizations`：允许主体读取 AWS Organizations 组织中的账户状态。
- `ssm-quicksetup`：允许主体在Quick Setup中执行只读操作。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS 托管式策略参考指南》中的 AWSQuickSetupReadOnlyAccess。

AWS 托管策略：AWSQuickSetupManageJITNAResourcesExecutionPolicy

托管策略 `AWSQuickSetupManageJITNAResourcesExecutionPolicy` 使 Quick Setup（Systems Manager 中的一个工具）能够设置即时节点访问。

您可以将 `AWSQuickSetupManageJITNAResourcesExecutionPolicy` 附加到 IAM 实体。Systems Manager 还会将此策略附加到服务角色，以允许 Systems Manager 代表您执行操作。

此策略授予管理权限，允许 Systems Manager 创建与即时节点访问关联的资源。

权限详细信息

该策略包含以下权限。

- ssm：允许主体获取和更新服务设置，该设置为即时节点访问指定身份提供者。
- iam：允许主体创建、标记和获取角色，为即时节点访问托管策略附加角色策略，并为即时节点访问和通知创建与服务相关的角色。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS Managed Policy Reference Guide》中的 AWSQuickSetupManageJITNAResourcesExecutionPolicy。

AWS 托管策略：AWSQuickSetupJITNADeploymentRolePolicy

托管策略 AWSQuickSetupJITNADeploymentRolePolicy 允许 Quick Setup 部署必要的配置类型，用于设置即时节点访问。

您可以将 AWSQuickSetupJITNADeploymentRolePolicy 附加到 IAM 实体。Systems Manager 还会将此策略附加到服务角色，以允许 Systems Manager 代表您执行操作。

此策略授予管理权限，允许 Systems Manager 创建与即时节点访问关联的资源。

权限详细信息

该策略包含以下权限。

- cloudformation：允许主体创建、更新、删除和读取 AWS CloudFormation 堆栈。
- ssm：允许主体创建、删除、更新和读取由 AWS CloudFormation 调用的 State Manager 关联。
- iam：允许主体创建、删除、读取和标记由 AWS CloudFormation 调用的 IAM 角色。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS Managed Policy Reference Guide》中的 AWSQuickSetupJITNADeploymentRolePolicy。

AWS 托管策略：AWSSystemsManagerJustInTimeAccessServicePolicy

托管策略 AWSSystemsManagerJustInTimeAccessServicePolicy 为由 Systems Manager 即时节点访问功能管理或使用的 AWS 资源提供权限。

您不能将 AWSSystemsManagerJustInTimeAccessServicePolicy 附加到自己的 IAM 实体。此附加到服务相关角色的策略允许 Systems Manager 代表您执行操作。有关更多信息，请参阅 使用角色启用即时节点访问。

此策略授予允许访问与即时节点访问关联的资源的管理权限。

权限详细信息

该策略包含以下权限。

- ssm：允许主体创建、描述和更新 OpsItems、描述会话和列出托管节点的标签。
- ssm-guiconnect：允许主体列出连接。
- identitystore：允许获取用户和群组 ID、描述用户和列出群组成员资格。
- sso-directory：允许主体描述用户并确定用户是否为组的成员。
- sso：允许主体描述注册的区域并列出实例和目录关联。
- cloudwatch：允许主体将指标数据放入 AWS/SSM/JustInTimeAccess 命名空间。
- ec2：允许主体描述标签。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS Managed Policy Reference Guide》中的 AWSSystemsManagerJustInTimeAccessServicePolicy。

AWS 托管策略：AWSSystemsManagerJustInTimeAccessTokenPolicy

托管策略 AWSSystemsManagerJustInTimeAccessTokenPolicy 允许 Systems Manager 生成用于即时节点访问的访问令牌。

您可以将 AWSSystemsManagerJustInTimeAccessTokenPolicy 附加到 IAM 实体。

此策略授予管理权限，允许 Systems Manager 启动会话并为即时节点访问生成访问令牌。

权限详细信息

该策略包含以下权限。

- ssm：允许主体发送命令、列出命令调用以及启动、恢复和结束 Session Manager 会话。
- ssm-guiconnect：允许主体启动、描述和结束 Systems Manager GUI Connect RDP 连接。
- kms：允许主体创建授权和生成密钥数据。
- sso：允许主体列出目录关联。
- identitystore：允许主体描述用户。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS Managed Policy Reference Guide》中的 AWSSystemsManagerJustInTimeAccessTokenPolicy。

AWS 托管策略：AWSSystemsManagerJustInTimeAccessTokenSessionPolicy

托管策略 AWSSystemsManagerJustInTimeAccessTokenSessionPolicy 允许 Systems Manager 将缩小范围的权限应用于即时节点访问令牌。

您可以将 AWSSystemsManagerJustInTimeAccessTokenSessionPolicy 附加到 IAM 实体。

此策略授予管理权限，允许 Systems Manager 缩小即时节点访问令牌的权限范围。

权限详细信息

该策略包含以下权限。

- ssm：允许主体使用 SSM-SessionManagerRunShell 文档开始 Session Manager 会话。同样，当首先通过 ssm-guiconnect 调用时，使用 AWS-StartPortForwardingSession 文档启动会话，列出命令调用，并使用 AWSSSO-CreateSSOUser 文档发送命令。
- ssm-guiconnect：允许主体取消、获取和启动所有资源的连接。
- kms：当使用 AWS 服务通过 ssm-guiconnect 调用时，允许主体为使用 SystemsManagerJustInTimeNodeAccessManaged 标记的密钥创建授权和生成数据密钥。
- sso：允许主体在通过 ssm-guiconnect 调用时列出目录关联。
- identitystore：允许主体在通过 ssm-guiconnect 调用时描述用户。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS Managed Policy Reference Guide》中的 AWSSystemsManagerJustInTimeAccessTokenSessionPolicy。

AWS 托管策略：AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy

托管策略 AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy 允许 Systems Manager 将来自委派管理员账户的拒绝访问策略共享给成员账户，并在多个 AWS 区域 账户之间复制这些策略。

您可以将 AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy 附加到 IAM 实体。

此策略提供 Systems Manager 共享和创建拒绝访问策略必要的管理权限。这样可以确保拒绝访问策略适用于 AWS Organizations 组织中的所有账户和针对即时节点访问配置的区域。

权限详细信息

该策略包含以下权限。

- ssm：允许主体管理 SSM 文档和资源策略。
- ssm-quicksetup：允许主体读取 Quick Setup 配置管理器。
- organizations：允许主体列出 AWS Organizations 组织的详细信息和委派管理员。
- ram：允许主体创建、标记和描述资源共享。
- iam：允许主体描述服务角色。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS Managed Policy Reference Guide》中的 AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy。

AWS 托管策略：AWSSystemsManagerNotificationsServicePolicy

托管策略 AWSSystemsManagerNotificationsServicePolicy 允许 Systems Manager 向访问请求审批者发送有关即时节点访问请求的电子邮件通知。

您不能将 AWSSystemsManagerJustInTimeAccessServicePolicy 附加到自己的 IAM 实体。此附加到服务相关角色的策略允许 Systems Manager 代表您执行操作。有关更多信息，请参阅 使用角色发送即时节点访问请求通知。

此策略授予管理权限，允许 Systems Manager 向访问请求审批者发送有关即时节点访问请求的电子邮件通知。

权限详细信息

该策略包含以下权限。

- identitystore：允许主体列出和描述用户和组成员资格。
- sso：允许主体列出实例、目录和描述注册区域。
- sso-directory：允许主体描述用户和列出组中的成员。
- iam：允许主体获取有关角色的信息。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS Managed Policy Reference Guide》中的 AWSSystemsManagerNotificationsServicePolicy。

AWS 托管式策略：AWS-SSM-Automation-DiagnosisBucketPolicy

托管式策略 AWS-SSM-Automation-DiagnosisBucketPolicy 通过允许访问用于诊断和修复问题的 S3 存储桶，提供诊断与 AWS Systems Manager 服务交互的节点的问题的权限。

您可以将 AWS-SSM-Automation-DiagnosisBucketPolicy 策略附加到 IAM 身份。Systems Manager 还会将此策略附加到 IAM 角色，以允许 Systems Manager 代表您执行诊断操作。

权限详细信息

该策略包含以下权限。

- s3：允许主体访问并将对象写入 Amazon S3 存储桶。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS 托管式策略参考指南》中的 AWS-SSM-Automation-DiagnosisBucketPolicy。

AWS 托管式策略：AWS-SSM-RemediationAutomation-OperationalAccountAdministrationRolePolicy

托管式策略 AWS-SSM-RemediationAutomation-OperationalAccountAdministrationRolePolicy 通过提供组织特定的权限，为操作账户提供诊断节点问题的权限。

您可以将 AWS-SSM-RemediationAutomation-OperationalAccountAdministrationRolePolicy 附加到您的 IAM 身份。Systems Manager 还会将此策略附加到 IAM 角色，以允许 Systems Manager 代表您执行诊断操作。

权限详细信息

该策略包含以下权限。

- organizations：允许主体列出组织的根，并获取成员账户以确定目标账户。
- sts：允许主体代入修复执行角色，以在同一组织内跨账户和区域运行 SSM 自动化文档。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS 托管式策略参考指南》中的 AWS-SSM-RemediationAutomation-OperationalAccountAdministrationRolePolicy。

AWS 托管式策略：AWS-SSM-DiagnosisAutomation-OperationalAccountAdministrationRolePolicy

托管式策略 AWS-SSM-DiagnosisAutomation-OperationalAccountAdministrationRolePolicy 通过提供组织特定的权限来为操作账户提供诊断节点问题的权限。

您可以将 AWS-SSM-DiagnosisAutomation-OperationalAccountAdministrationRolePolicy 策略附加到 IAM 身份。Systems Manager 还会将此策略附加到 IAM 角色，以允许 Systems Manager 代表您执行诊断操作。

权限详细信息

该策略包含以下权限。

- organizations：允许主体列出组织的根，并获取成员账户以确定目标账户。
- sts：允许主体代入诊断执行角色，以在同一组织内跨账户和区域运行 SSM 自动化文档。

要查看有关策略的更多详细信息（包括 JSON 策略文档的最新版本），请参阅《AWS 托管式策略参考指南》中的 AWS-SSM-DiagnosisAutomation-OperationalAccountAdministrationRolePolicy。

Systems Manager 更新了 AWS 托管式策略

在下表中，查看自该服务于 2021 年 3 月 12 日开始跟踪这些更改以来，有关 Systems Manager 的 AWS 托管策略更新的详细信息。有关 Systems Manager 服务的其他托管策略的信息，请参阅本主题后面的 Systems Manager 的其他托管策略。要获得有关此页面更改的自动提示，请订阅 Systems Manager 文档历史记录 页面上的 RSS 源。

更改	描述	日期
AmazonssAutomationRole：更新文档	Systems Manager 为现有 AmazonSSMAutomationRole 策略添加了全面的文档，该策略为 Systems Manager 自动化服务提供权限，以运行自动化运行手册中定义的活动。	2025 年 7 月 15 日
AWS-SSM-RemediationAutomation-ExecutionRolePolicy：更新现有策略	Systems Manager 更新了托管策略 ，要求同时拥有“文档”和“自动执行”资源类型的权限，从而改善 ssm:StartAutomationExecution API 的	2025 年 7 月 16 日

更改	描述	日期
	安全状况。更新后的策略为修复自动化执行提供了更加全面和详细的权限，包括改善对网络修复功能的描述，提供更具体的 Amazon VPC 端点创建权限、详细的安全组管理权限，以及增强对修复操作的资源标记控制。	
AWS-SSM-RemediationAutomation-AdministrationRolePolicy：更新现有策略	Systems Manager 更新了托管策略 ，以支持 API 授权改进实现修复自动化操作。更新后的策略强化了执行自动化文档中定义活动的权限，并改进了修复工作流程的安全控制和资源访问模式。	2025 年 7 月 16 日
AWS-SSM-DiagnosisAutomation-ExecutionRolePolicy：更新现有策略	Systems Manager 更新了托管策略 ，从而为诊断自动化执行提供更加详细和准确的权限。更新后的策略包括改善对 Amazon EC2 和 Amazon VPC 资源访问的描述，提供更加具体的 SSM 自动化权限，以及通过适当资源限制改善对 AWS KMS 和 IAM 权限的描述。	2025 年 7 月 16 日
AWS-SSM-DiagnosisAutomation-AdministrationRolePolicy：更新现有策略	Systems Manager 更新了托管策略 ，从而为诊断自动化操作提供更加具体的权限和安全条件。更新后的策略增强了对 AWS KMS 密钥使用、Amazon S3 存储桶访问和角色代入的安全控制，并采用更加严格的资源条件和账户级别限制。	2025 年 7 月 16 日

更改	描述	日期
AWSQuickSetupDeploymentRolePolicy：策略更新	Systems Manager 增加了此 AWSQuickSetupDeploymentRolePolicy 托管式策略的权限，用于访问 Amazon 拥有的运行手册 AWSQuickSetupType-ManageInstanceProfile。借助此权限，Quick Setup 将可以使用托管式策略而不是内联策略创建关联。	2025 年 7 月 14 日
AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy：策略更新	Systems Manager 增加了允许 Systems Manager 标记由 AWS Resource Access Manager 共享的资源的权限，从而支持即时节点访问。	2025 年 4 月 30 日
AAWSQuickSetupManageJITNAResourcesExecutionPolicy：策略更新	Systems Manager 添加了权限，允许 Systems Manager 标记为即时节点访问而创建的 IAM 角色。	2025 年 4 月 30 日
AWSSystemsManagerJustInTimeAccessTokenSessionPolicy：新策略	Systems Manager 添加了一项新策略，允许 Systems Manager 将缩小范围的权限应用于即时节点访问令牌。	2025 年 4 月 30 日
AWSSystemsManagerNotificationsServicePolicy：新策略	Systems Manager 添加了一项新策略，允许 Systems Manager 向访问请求审批者发送有关即时节点访问请求的电子邮件通知。	2025 年 4 月 30 日

更改	描述	日期
AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy：新策略	Systems Manager 添加了一项新策略，允许 Systems Manager 将审批策略复制到不同的区域。	2025 年 4 月 30 日
AWSSystemsManagerJustInTimeAccessTokenPolicy：新策略	Systems Manager 添加了一项新策略，允许 Systems Manager 生成用于即时节点访问的访问令牌。	2025 年 4 月 30 日
AWSSystemsManagerJustInTimeAccessServicePolicy：新策略	Systems Manager 添加了一项新策略，为由 Systems Manager 即时节点访问功能管理或使用的 AWS 资源提供权限。	2025 年 4 月 30 日
AWSQuickSetupManageJITNAResourcesExecutionPolicy：新策略	Systems Manager 添加了一项新策略，以允许 Quick Setup（Systems Manager 中的一项工具）创建即时节点访问所需的 IAM 角色。	2025 年 4 月 30 日
AWSQuickSetupJITNADeploymentRolePolicy：新策略	Systems Manager 添加了一个新策略，该策略提供了权限以允许 Quick Setup 部署设置即时节点访问所需的配置类型。	2025 年 4 月 30 日
AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy：策略更新	Systems Manager 增加了允许 Systems Manager 标记由 AWS Resource Access Manager 共享的资源的权限，从而支持即时节点访问。	2025 年 4 月 30 日

更改	描述	日期
AWSQuickSetupManageJITNAResourcesExecutionPolicy：策略更新	Systems Manager 添加了权限，允许 Systems Manager 标记为即时节点访问而创建的 IAM 角色。	2025 年 4 月 30 日
AWSSystemsManagerJustInTimeAccessTokenSessionPolicy：新策略	Systems Manager 添加了一项新策略，允许 Systems Manager 将缩小范围的权限应用于即时节点访问令牌。	2025 年 4 月 30 日
AWSSystemsManagerNotificationsServicePolicy：新策略	Systems Manager 添加了一项新策略，允许 Systems Manager 向访问请求审批者发送有关即时节点访问请求的电子邮件通知。	2025 年 4 月 30 日
AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy：新策略	Systems Manager 添加了一项新策略，允许 Systems Manager 将审批策略复制到不同的区域。	2025 年 4 月 30 日
AWSSystemsManagerJustInTimeAccessTokenPolicy：新策略	Systems Manager 添加了一项新策略，允许 Systems Manager 生成用于即时节点访问的访问令牌。	2025 年 4 月 30 日
AWSSystemsManagerJustInTimeAccessServicePolicy：新策略	Systems Manager 添加了一项新策略，为由 Systems Manager 即时节点访问功能管理或使用的 AWS 资源提供权限。	2025 年 4 月 30 日

更改	描述	日期
AWSQuickSetupManageJITNAResourcesExecutionPolicy：新策略	Systems Manager 添加了一项新策略，以允许 Quick Setup（Systems Manager 中的一项工具）创建即时节点访问所需的 IAM 角色。	2025 年 4 月 30 日
AWSQuickSetupJITNADeploymentRolePolicy：新策略	Systems Manager 添加了一个新策略，该策略提供了权限以允许 Quick Setup 部署设置即时节点访问所需的配置类型。	2025 年 4 月 30 日
AWS-SSM-DiagnosisAutomation-OperationalAccountAdministrationRolePolicy：新策略	Systems Manager 添加了一项新策略，通过提供组织特定的权限来为操作账户提供诊断节点问题的权限。	2024 年 11 月 21 日
AWS-SSM-RemediationAutomation-OperationalAccountAdministrationRolePolicy：新策略	Systems Manager 添加了一项新策略，通过提供组织特定的权限来为操作账户提供诊断节点问题的权限。	2024 年 11 月 21 日
AWS-SSM-Automation-DiagnosisBucketPolicy：新策略	Systems Manager 添加了一项新策略来支持启动自动化工作流，从而诊断目标账户和区域中的托管节点的问题。	2024 年 11 月 21 日
AmazonSSMServiceRolePolicy – 对现有策略的更新	Systems Manager 添加了新的权限，允许 AWS 资源探索器收集有关 Amazon EC2 实例的详细信息并在新 Systems Manager 控制面板的小组件中显示结果。	2024 年 11 月 21 日

更改	描述	日期
SSMQuickSetupRolePolicy – 对现有策略的更新	Systems Manager 已更新托管式策略 SSMQuickSetupRolePolicy 。此更新允许关联的服务相关角色 AWSServiceRoleForSSMQuickSetup 管理资源数据同步。	2024 年 11 月 21 日
AWS-SSM-DiagnosisAutomation-AdministrationRolePolicy：新策略	Systems Manager 添加了一项新策略来支持启动自动化工作流，从而诊断目标账户和区域中的托管节点的问题。	2024 年 11 月 21 日
AWS-SSM-DiagnosisAutomation-ExecutionRolePolicy：新策略	Systems Manager 添加了一项新策略来支持启动自动化工作流，从而诊断目标账户和区域中的托管节点的问题。	2024 年 11 月 21 日
AWS-SSM-RemediationAutomation-AdministrationRolePolicy：新策略	Systems Manager 添加了一项新策略来支持启动自动化工作流，从而修复目标账户和区域中的托管节点的问题。	2024 年 11 月 21 日
AWS-SSM-RemediationAutomation-ExecutionRolePolicy：新策略	Systems Manager 添加了一项新策略来支持启动自动化工作流，从而修复目标账户和区域中的托管节点的问题。	2024 年 11 月 21 日
AWSQuickSetupSSMDeploymentRolePolicy：策略更新	Systems Manager 添加了权限以允许 Systems Manager 标记出为统一控制台创建的 IAM 角色和 Lambda。	2025 年 5 月 7 日

更改	描述	日期
AWSQuickSetupSSMManageResourcesExecutionPolicy：新策略	Systems Manager 添加了一项新策略来支持在Quick Setup中运行操作，该操作会为Quick Setup关联创建 IAM 角色，而这些角色又由 AWSQuickSetupType-SSM 部署创建。	2024 年 11 月 21 日
AWSQuickSetupSSMLifecycleManagementExecutionPolicy：新策略	ystems Manager 添加了一项新策略来支持Quick Setup在Quick Setup部署期间对生命周期事件运行 AWS CloudFormation 自定义资源。	2024 年 11 月 21 日
AWSQuickSetupSSMDeploymentRolePolicy：新策略	Systems Manager 添加了一项新策略来支持授予管理权限，允许Quick Setup创建为 Systems Manager 加入过程中使用的资源。	2024 年 11 月 21 日
AWSQuickSetupSSMDeploymentS3BucketRolePolicy：新策略	Systems Manager 添加了一项新策略来支持管理和检索有关通过 AWS CloudFormation 模板管理的主体账户中特定存储桶的信息	2024 年 11 月 21 日
AWSQuickSetupEnableDHMCExecutionPolicy：新策略	Systems Manager 正在推出一项新策略来允许Quick Setup创建一个 IAM 角色，该角色本身使用现有的 AmazonSSMManagedEC2InstanceDefaultPolicy。此策略包含 SSM Agent 与 Systems Manager 服务通信所需的所有权限。该新策略还允许修改 Systems Manager 服务设置。	2024 年 11 月 21 日

更改	描述	日期
AWSQuickSetupEnableAREXExecutionPolicy：新策略	Systems Manager 添加了一项新策略来允许Quick Setup为 AWS 资源探索器创建服务相关角色，用于访问资源探索器视图和聚合器索引。	2024 年 11 月 21 日
AWSQuickSetupManagedInstanceProfileExecutionPolicy：新策略	Systems Manager 添加了一项新策略来允许Quick Setup创建默认Quick Setup实例配置文件，并将其附加到任何缺少关联实例配置文件的 Amazon EC2 实例。此新策略还允许Quick Setup将权限附加到现有配置文件，以确保已授予所有必需的 Systems Manager 权限。	2024 年 11 月 21 日
AWSQuickSetupFullAccess：新策略	Systems Manager 添加了一项新策略，允许实体完全访问 AWS Systems Manager Quick Setup和 AWS SDK 中的 AWS Management Console API 操作和数据，以及有限访问Quick Setup操作所需的其他 AWS 服务资源。	2024 年 11 月 21 日
AWSQuickSetupReadOnlyAccess：新策略	Systems Manager 添加了一项新策略以授予只读权限，允许主体查看 AWS Systems Manager Quick Setup数据和报告，包括Quick Setup操作所需的其他 AWS 服务资源的信息。	2024 年 11 月 21 日

更改	描述	日期
SSMQuickSetupRolePolicy – 对现有策略的更新	Systems Manager 增加了新的权限，以允许 Quick Setup 检查其创建的其他 AWS CloudFormation 堆栈集的运行状况。	2024 年 8 月 13 日
AmazonSSMManagedEC2InstanceDefaultPolicy – 对现有策略的更新	Systems Manager 已针对 AmazonSSMManagedEC2InstanceDefaultPolicy 向 JSON 策略添加语句 ID（Sid）。这些 Sid 针对每条策略语句的目的提供内联描述。	2024 年 7 月 18 日
SSMQuickSetupRolePolicy：新策略	Systems Manager 添加了新策略，允许 Quick Setup 检查已部署的资源的运行状况并修复偏离原始配置的实例。	2024 年 7 月 3 日
AWSQuickSetupDeploymentRolePolicy：新策略	Systems Manager 添加了新策略以支持多种快速设置配置类型，这些配置类型可创建 IAM 角色和自动化，进而使用推荐的最佳实践配置常用的 Amazon Web Services 服务和功能。	2024 年 7 月 3 日
AWSQuickSetupPatchPolicyDeploymentRolePolicy – 新策略	Systems Manager 添加了新策略，允许 Quick Setup 创建与 Patch Manager 补丁策略 Quick Setup 配置关联的资源。	2024 年 7 月 3 日

更改	描述	日期
AWSQuickSetupPatchPolicyBaselineAccess – 新策略	Systems Manager 添加了新策略，允许 Quick Setup 以只读权限访问 Patch Manager 中的补丁基准。	2024 年 7 月 3 日
AWSSystemsManagerEnableExplorerExecutionPolicy – 新策略	Systems Manager 添加了新策略，允许 Quick Setup 授予启用 Explorer 的管理权限。	2024 年 7 月 3 日
AWSSystemsManagerEnableConfigRecordingExecutionPolicy – 新策略	Systems Manager 添加了允许 Quick Setup 启用和配置 AWS Config 配置记录的新策略。	2024 年 7 月 3 日
AWSQuickSetupDevOpsGuruPermissionsBoundary – 新策略	Systems Manager 添加了新策略，允许 Quick Setup 启用和配置 Amazon DevOps Guru。	2024 年 7 月 3 日
AWSQuickSetupDistributorPermissionsBoundary – 新策略	Systems Manager 添加了新策略，允许Quick Setup启用和配置 Distributor（AWS Systems Manager 中的一项工具）。	2024 年 7 月 3 日
AWSQuickSetupSSMHostMgmtPermissionsBoundary – 新策略	Systems Manager 添加了新策略，允许Quick Setup启用和配置 Systems Manager 工具，从而安全地管理 Amazon EC2 实例。	2024 年 7 月 3 日
AWSQuickSetupPatchPolicyPermissionsBoundary – 新策略	Systems Manager 添加了新策略，允许Quick Setup在Patch Manager（AWS Systems Manager 中的一项工具）中启用和配置补丁策略。	2024 年 7 月 3 日

更改	描述	日期
AWSQuickSetupSchedulerPermissionsBoundary – 新策略	Systems Manager 添加了新策略，允许 Quick Setup 在 Amazon EC2 实例和其他资源上启用和配置计划操作。	2024 年 7 月 3 日
AWSQuickSetupCFGCPacksPermissionsBoundary – 新策略	Systems Manager 添加了允许 Quick Setup 部署 AWS Config 一致性包的新策略。	2024 年 7 月 3 日
AWSSystemsManagerOpsDataSyncServiceRolePolicy – 对现有策略的更新	OpsCenter 更新了策略，以提高 Explorer 管理 OpsData 相关操作的服务相关角色内服务代码的安全性。	2023 年 7 月 3 日
AmazonSSMManagedEC2InstanceDefaultPolicy：新策略	Systems Manager 添加了新策略，允许 Amazon EC2 实例上的 Systems Manager 功能，而不使用 IAM 实例配置文件。	2022 年 8 月 18 日
AmazonSSMServiceRolePolicy – 更新到现有策略	Systems Manager 添加了新的权限，允许 Explorer 在您从 Explorer 或 OpsCenter 启用 Security Hub 时创建托管规则。添加了新的权限，以便在允许 OpsData 之前检查 config 和 compute-optimizer 是否满足必需的要求。	2021 年 4 月 27 日
AWSSystemsManagerOpsDataSyncServiceRolePolicy：新策略	Systems Manager 添加了新策略，用于从 Explorer 和 OpsCenter 中的 Security Hub 调查结果创建并更新 OpsItems 及 OpsData。	2021 年 4 月 27 日

更改	描述	日期
AmazonSSMServiceRolePolicy – 更新到现有策略	Systems Manager 添加了新的权限，允许在 Explorer 中查看多个账户和 AWS 区域 的聚合 OpsData 及 OpsItems 详细信息。	2021 年 3 月 24 日
Systems Manager 已开启跟踪更改	Systems Manager 为其 AWS 托管式策略开启了跟踪更改。	2021 年 3 月 12 日

Systems Manager 的其他托管策略

除了本主题前面介绍的托管策略外，Systems Manager 还支持以下策略。

- AmazonSSMAutomationApproverAccess – 允许访问以查看自动化执行并将批准决策发送到等待批准的自动化的 AWS 托管策略。
- AmazonSSMDirectoryServiceAccess – 允许 SSM Agent 代表用户访问 AWS Directory Service 以处理托管式节点加入域的请求的 AWS 托管策略。
- AmazonSSMFullAccess – 授予 Systems Manager API 和文档完全访问权限的 AWS 托管策略。
- AmazonSSMMaintenanceWindowRole – 为维护时段提供 Systems Manager API 权限的 AWS 托管策略。
- AmazonSSMManagedInstanceCore – 允许节点使用 Systems Manager 服务核心功能的 AWS 托管式策略。
- AmazonSSMPatchAssociation – 为补丁关联操作提供对子实例的访问权限的 AWS 托管式策略。
- AmazonSSMReadOnlyAccess – 授予 Systems Manager 只读 API 操作（例如 `Get*` 和 `List*`）访问权限的 AWS 托管策略。
- AWSSSMOpsInsightsServiceRolePolicy – 为在 Systems Manager 中创建和更新运维洞察 OpsItems 提供权限的 AWS 托管策略。其借助服务相关角色 AWSServiceRoleForAmazonSSM_OpsInsights 提供权限。
- AWSSystemsManagerAccountDiscoveryServicePolicy – 授予 Systems Manager 发现 AWS 账户 信息的权限的 AWS 托管策略。
- AWSSystemsManagerChangeManagementServicePolicy – 提供对由 Systems Manager 更改管理框架管理或使用的以及由服务相关角色

AWSServiceRoleForSystemsManagerChangeManagement 使用的 AWS 资源的访问权限的 AWS 托管策略。

- AmazonEC2RoleforSSM – 此策略不再受支持，不应使用。在相应的位置，使用 AmazonSSMManagedInstanceCore 策略在 EC2 实例上允许 Systems Manager 服务的核心功能。有关信息，请参阅配置 Systems Manager 所需的实例权限。

对 AWS Systems Manager 身份和访问进行故障排除

使用以下信息可帮助您诊断和修复在使用 AWS Systems Manager 和 AWS Identity and Access Management (IAM) 时可能会遇到的常见问题。

主题

- 我无权在 Systems Manager 中执行操作
- 我无权执行 iam:PassRole
- 我希望允许我的 AWS 账户以外的人访问我的 Systems Manager 资源

我无权在 Systems Manager 中执行操作

如果 AWS Management Console 告诉您，您无权执行某个操作，则必须联系您的管理员寻求帮助。管理员是向您提供登录凭证的人。

当 mateojackson 用户尝试使用控制台查看有关文档的详细信息，但没有 ssm:GetDocument 权限时，将发生以下示例错误。

```
User: arn:aws:ssm::123456789012:user/mateojackson isn't authorized to perform:
 ssm:GetDocument on resource: MyExampleDocument
```

在这种情况下，Mateo 请求他的管理员更新其策略，以允许他使用 ssm:GetDocument 操作访问 MyExampleDocument 资源。

我无权执行 iam:PassRole

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

有些 AWS 服务允许您将现有角色传递到该服务，而不是创建新服务角色或服务相关角色。为此，您必须具有将角色传递到服务的权限。

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

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

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

如果您需要帮助，请联系 AWS 管理员。您的管理员是提供登录凭证的人。

我希望允许我的 AWS 账户以外的人访问我的 Systems Manager 资源

您可以创建一个角色，以便其他账户中的用户或您组织外的人员可以使用该角色来访问您的资源。您可以指定谁值得信赖，可以担任角色。对于支持基于资源的策略或访问控制列表（ACL）的服务，您可以使用这些策略向人员授予对您的资源的访问权。

要了解更多信息，请参阅以下内容：

- 要了解 Systems Manager 是否支持这些特征，请参阅 AWS Systems Manager 如何与 IAM 协同工作。
- 要了解如何为您拥有的 AWS 账户 中的资源提供访问权限，请参阅《IAM 用户指南》中的为您拥有的另一个 AWS 账户中的 IAM 用户提供访问权限。
- 要了解如何为第三方 AWS 账户 提供您的资源的访问权限，请参阅 IAM 用户指南中的为第三方拥有的 AWS 账户 提供访问权限。
- 要了解如何通过联合身份验证提供访问权限，请参阅《IAM 用户指南》中的为经过外部身份验证的用户（联合身份验证）提供访问权限。
- 要了解使用角色和基于资源的策略进行跨账户访问之间的差别，请参阅《IAM 用户指南》中的 IAM 中的跨账户资源访问。

将服务相关角色用于 Systems Manager

AWS Systems Manager 使用 AWS Identity and Access Management（IAM）服务相关角色。服务相关角色是一种独特类型的 IAM 角色，它与 Systems Manager 直接相关。服务相关角色由 Systems Manager 预定义，并包含相关服务代表您调用其他 AWS 服务所需的所有权限。

> **Note**
>
> 服务角色不同于服务相关角色。服务角色是一种AWS Identity and Access Management（IAM）角色，用于向某个 AWS 服务授予相应的权限，以便该服务可以访问 AWS 资源。只有几个 Systems Manager 场景需要服务角色。当您创建 Systems Manager 的服务角色时，您可以选择要授予的权限，以便它可以访问其他 AWS 资源或与之交互。

服务相关角色使 Systems Manager 的设置更轻松，因为您不必手动添加必要的权限。Systems Manager 定义其服务相关角色的权限，除非另行定义，否则仅 Systems Manager 可以担任其角色。定义的权限包括信任策略和权限策略，以及不能附加到任何其他 IAM 实体的权限策略。

只有在首先删除相关资源后，您才能删除服务相关角色。这将保护您的 Systems Manager 资源，因为您不会无意中删除对资源的访问权限。

> **Note**
>
> 对于混合和多云环境中的非 EC2 节点，您需要额外的 IAM 角色来允许这些计算机与 Systems Manager 服务通信。这就是 Systems Manager 的 IAM 服务角色。此角色向 AWS Security Token Service (AWS STS) AssumeRole 授予对 Systems Manager 服务的信任。`AssumeRole` 操作返回一组临时安全凭证 (由访问密钥 ID、秘密访问密钥和安全令牌组成)。您使用这些临时凭证访问通常可能无法访问的 AWS 资源。有关更多信息，请参阅《AWS Security Token Service API 参考》中的在混合和多云环境中创建 Systems Manager 所需的 IAM 服务角色和 AssumeRole。

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

主题

- 使用角色来收集清单并查看 OpsData
- 使用角色为 OpsCenter 和 Explorer 收集 AWS 账户 信息
- 使用角色为 Explorer 创建 OpsData 和 OpsItems
- 在 Systems Manager OpsCenter 中使用角色创建运营洞察 OpsItem
- 使用角色来维护已配置 Quick Setup 的资源运行状况和一致性

- 使用角色导出 Explorer OpsData
- 使用角色启用即时节点访问
- 使用角色发送即时节点访问请求通知

使用角色来收集清单并查看 OpsData

Systems Manager 使用名为 **AWSServiceRoleForAmazonSSM** 的服务相关角色。AWS Systems Manager 使用此 IAM 服务角色代表您管理 AWS 资源。

清单、OpsData 和 OpsItem 的服务相关角色权限

AWSServiceRoleForAmazonSSM 服务相关角色仅信任 ssm.amazonaws.com 服务担任此角色。

您可以使用 Systems Manager 服务相关角色 AWSServiceRoleForAmazonSSM 执行以下操作：

- Systems Manager Inventory 工具使用服务相关角色 AWSServiceRoleForAmazonSSM 从标签和资源组采集清单元数据。
- Explorer 工具使用服务相关角色 AWSServiceRoleForAmazonSSM 来启用查看多个账户的 OpsData 和 OpsItems 的功能。当您将 Security Hub 启用为 Explorer 或者 OpsCenter 的数据源时，此服务相关角色还允许 Explorer 创建托管规则。

> **Important**
>
> 以前，Systems Manager 控制台允许您选择 AWS 托管式 IAM 服务相关角色 AWSServiceRoleForAmazonSSM，以用作任务的维护角色。现在不再建议将此角色及其相关策略 AmazonSSMServiceRolePolicy 用于维护时段任务。如果您目前在将此角色用于维护时段任务，我们建议您停止使用它。而应创建您自己的 IAM 角色，以便您的维护时段任务运行时在 Systems Manager 与其他 AWS 服务之间进行通信。
> 有关更多信息，请参阅 设置 Maintenance Windows。

用于为 AWSServiceRoleForAmazonSSM 角色提供权限的托管策略是 AmazonSSMServiceRolePolicy。有关该策略授予的权限的详细信息，请参阅 AWS 托管式策略：AmazonSSMServiceRolePolicy。

创建 Systems Manager 的 **AWSServiceRoleForAmazonSSM** 服务相关角色

您可以使用 IAM 控制台为 EC2 使用案例创建服务相关角色。在 AWS Command Line Interface (AWS CLI) 中使用 IAM 的命令或使用 IAM API，创建服务名称为 `ssm.amazonaws.com` 的服务相关角色。有关更多信息，请参阅 IAM 用户指南 中的创建服务相关角色。

如果您删除该服务相关角色，然后需要再次创建，您可以使用相同流程在账户中重新创建此角色。

编辑 Systems Manager 的 **AWSServiceRoleForAmazonSSM** 服务相关角色

Systems Manager 不允许您编辑 `AWSServiceRoleForAmazonSSM` 服务相关角色。在创建服务相关角色后，您将无法更改角色的名称，因为可能有多种实体引用该角色。不过，您可以使用 IAM 编辑角色的说明。有关更多信息，请参阅《IAM 用户指南》中的编辑服务相关角色。

删除 Systems Manager 的 **AWSServiceRoleForAmazonSSM** 服务相关角色

如果您不再需要使用某个需要服务相关角色的功能或服务，建议您删除该角色。这样您就没有未被主动监控或维护的未使用实体。您可以使用 IAM 控制台、AWS CLI 或 IAM API 手动删除此服务相关角色。为此，您必须先手动清除服务相关角色的资源，然后才能手动删除它。

多项工具都可使用 `AWSServiceRoleForAmazonSSM` 服务相关角色，因此在尝试删除该角色之前，请确保没有任何工具在使用该角色。

- Inventory：如果删除 Inventory 工具使用的服务相关角色，则标签和资源组的清单数据将不再同步。您必须先清除服务相关角色的资源，然后才能手动删除它。
- Explorer：如果删除 Explorer 工具使用的服务相关角色，则无法再查看跨账户和跨区域的 OpsData 及 OpsItems。

> **Note**
>
> 如果在您尝试删除标签或资源组时 Systems Manager 服务正在使用该角色，则删除操作可能会失败。如果发生这种情况，请等待几分钟后重试。

删除 **AWSServiceRoleForAmazonSSM** 所用的 Systems Manager 资源

1. 要删除标签，请参阅为单个资源添加和删除标签。
2. 要删除资源组，请参阅从 AWS Resource Groups 中删除组。

使用 IAM 手动删除 **AWSServiceRoleForAmazonSSM** 服务相关角色

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

支持 Systems Manager**AWSServiceRoleForAmazonSSM** 服务相关角色的区域

Systems Manager 支持提供该服务的所有 AWS 区域中使用 AWSServiceRoleForAmazonSSM 服务相关角色。有关更多信息，请参阅 AWS Systems Manager 终端节点和限额。

使用角色为 OpsCenter 和 Explorer 收集 AWS 账户 信息

Systems Manager 使用名为 **AWSServiceRoleForAmazonSSM_AccountDiscovery** 的服务相关角色。AWS Systems Manager 使用此 IAM 服务角色调用其他 AWS 服务来发现 AWS 账户 信息。

Systems Manager 账户发现的服务相关角色权限

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

- accountdiscovery.ssm.amazonaws.com

角色权限策略允许 Systems Manager 对指定资源完成以下操作：

- organizations:DescribeAccount
- organizations:DescribeOrganizationalUnit
- organizations:DescribeOrganization
- organizations:ListAccounts
- organizations:ListAWSServiceAccessForOrganization
- organizations:ListChildren
- organizations:ListParents
- organizations:ListDelegatedServicesForAccount
- organizations:ListDelegatedAdministrators
- organizations:ListRoots

您必须配置权限，允许 IAM 实体（如用户、组或角色）创建、编辑或删除服务相关角色。有关更多信息，请参阅《IAM 用户指南》中的服务相关角色权限。

创建 Systems Manager 的 **AWSServiceRoleForAmazonSSM_AccountDiscovery** 服务相关角色

如果要跨多个 AWS 账户使用 Explorer 和 OpsCenter（都是 Systems Manager 中的工具），则必须创建服务相关角色。对于 OpsCenter，您必须手动创建服务相关角色。有关更多信息，请参阅（可选）将 OpsCenter 手动设置为跨账户集中管理 OpsItems。

对于 Explorer，如果您使用 AWS Management Console 中的 Systems Manager 创建资源数据同步，则可以通过选择 Create role(创建角色)按钮创建服务相关角色。如果要以编程方式创建资源数据同步，则必须在创建资源数据同步之前创建角色。您可以使用 CreateServiceLinkedRole API 操作创建角色。

编辑 Systems Manager 的 **AWSServiceRoleForAmazonSSM_AccountDiscovery** 服务相关角色

Systems Manager 不允许您编辑 AWSServiceRoleForAmazonSSM_AccountDiscovery 服务相关角色。在创建服务相关角色后，您将无法更改角色的名称，因为可能有多种实体引用该角色。不过，您可以使用 IAM 编辑角色的说明。有关更多信息，请参阅《IAM 用户指南》中的编辑服务相关角色。

删除 Systems Manager 的 **AWSServiceRoleForAmazonSSM_AccountDiscovery** 服务相关角色

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

清除 **AWSServiceRoleForAmazonSSM_AccountDiscovery** 服务相关角色

您必须首先删除所有 Explorer 资源数据同步，然后才能使用 IAM 删除 AWSServiceRoleForAmazonSSM_AccountDiscovery 服务相关角色。

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

手动删除 **AWSServiceRoleForAmazonSSM_AccountDiscovery** 服务相关角色

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

支持 Systems Manager**AWSServiceRoleForAmazonSSM_AccountDiscovery** 服务相关角色的区域

Systems Manager 支持在服务可用的所有区域中使用服务相关角色。有关更多信息，请参阅 AWS Systems Manager 终端节点和限额。

对 AWSServiceRoleForAmazonSSM_AccountDiscovery 服务相关角色的更新

查看有关 AWSServiceRoleForAmazonSSM_AccountDiscovery 服务相关角色更新（从该服务开始跟踪这些变更开始）的详细信息。要获得有关此页面更改的自动提示，请订阅 Systems Manager 文档历史记录 页面上的 RSS 源。

更改	描述	日期
已添加新权限	此服务相关角色现在包括 `organizations:DescribeOrganizationalUnit` 和 `organizations:ListRoots` 权限。这些权限让 AWS Organizations 管理账户或 Systems Manager 委派管理员账户可以跨账户使用 OpsItems。有关更多信息，请参阅 （可选）将 OpsCenter 手动设置为跨账户集中管理 OpsItems。	2022 年 10 月 17 日

使用角色为 Explorer 创建 OpsData 和 OpsItems

Systems Manager 使用名为 **AWSServiceRoleForSystemsManagerOpsDataSync** 的服务相关角色。AWS Systems Manager 使用此 IAM 服务角色为 Explorer 创建 OpsData 及 OpsItems。

Systems Manager OpsData 同步的服务相关角色权限

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

- `opsdatasync.ssm.amazonaws.com`

角色权限策略允许 Systems Manager 对指定资源完成以下操作：

- Systems Manager Explorer 需要使用服务相关角色授予相应的权限，以便在更新 OpsItem 时更新安全结果，创建和更新 OpsItem，以及在客户删除 SSM 托管式规则时关闭 Security Hub 数据源。

用于为 `AWSServiceRoleForSystemsManagerOpsDataSync` 角色提供权限的托管策略是 `AWSSystemsManagerOpsDataSyncServiceRolePolicy`。有关该策略授予的权限的详细信息，请参阅 AWS 托管式策略：AWSSystemsManagerOpsDataSyncServiceRolePolicy。

您必须配置权限，允许 IAM 实体（如用户、组或角色）创建、编辑或删除服务相关角色。有关更多信息，请参阅《IAM 用户指南》中的服务相关角色权限。

创建 Systems Manager 的 **AWSServiceRoleForSystemsManagerOpsDataSync** 服务相关角色

您无需手动创建服务相关角色。当您在 AWS Management Console 中启用 Explorer 时，Systems Manager 将为您创建服务相关角色。

> **Important**
>
> 如果您在使用此服务相关角色支持的功能的其他服务中完成某个操作，该角色可以显示在您的账户中。此外，如果您在 2017 年 1 月 1 日之前使用 Systems Manager 服务，当它开始支持服务相关角色时，则 Systems Manager 会在您的账户中创建 `AWSServiceRoleForSystemsManagerOpsDataSync` 角色。要了解更多信息，请参阅我的 IAM 账户中出现新角色。

如果您删除该服务相关角色，然后需要再次创建，您可以使用相同流程在账户中重新创建此角色。当您在 AWS Management Console 中启用 Explorer 时，Systems Manager 将再次为您创建服务相关角色。

您也可以使用 IAM 控制台创建服务相关角色，用允许 Explorer 创建 OpsData 和 OpsItems 的 AWS 服务角色用例来进行创建。在 AWS CLI 或 AWS API 中，使用 `opsdatasync.ssm.amazonaws.com` 服务名称创建服务相关角色。有关更多信息，请参阅《IAM 用户指南》中的创建服务相关角色。如果您删除了此服务相关角色，可以使用同样的过程再次创建角色。

编辑 Systems Manager 的 **AWSServiceRoleForSystemsManagerOpsDataSync** 服务相关角色

Systems Manager 不允许您编辑 AWSServiceRoleForSystemsManagerOpsDataSync 服务相关角色。在创建服务相关角色后，您将无法更改角色的名称，因为可能有多种实体引用该角色。不过，您可以使用 IAM 编辑角色的说明。有关更多信息，请参阅《IAM 用户指南》中的编辑服务相关角色。

删除 Systems Manager 的 **AWSServiceRoleForSystemsManagerOpsDataSync** 服务相关角色

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

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

删除 AWSServiceRoleForSystemsManagerOpsDataSync 角色使用的 Systems Manager 资源的过程取决于您是否已将 Explorer 或 OpsCenter 配置为与 Security Hub 集成。

要删除 **AWSServiceRoleForSystemsManagerOpsDataSync** 角色使用的 Systems Manager 资源，请参阅以下信息：

- 要停止 Explorer 为 Security Hub 调查结果创建新的 OpsItems，请参阅 如何停止接收结果。
- 要停止 OpsCenter 为 Security Hub 结果创建新的 OpsItems，请参阅

使用 IAM 手动删除 **AWSServiceRoleForSystemsManagerOpsDataSync** 服务相关角色

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

支持 Systems Manager**AWSServiceRoleForSystemsManagerOpsDataSync** 服务相关角色的区域

Systems Manager 支持在服务可用的所有区域中使用服务相关角色。有关更多信息，请参阅 AWS Systems Manager 终端节点和限额。

Systems Manager 并非在提供服务的每个区域中都支持使用服务相关角色。您可以在以下区域中使用 `AWSServiceRoleForSystemsManagerOpsDataSync` 角色。

AWS 区域 名称	区域标识	在 Systems Manager 中支持
美国东部（弗吉尼亚州北部）	us-east-1	是
美国东部（俄亥俄州）	us-east-2	是
美国西部（加利福尼亚北部）	us-west-1	是
美国西部（俄勒冈州）	us-west-2	是
亚太地区（孟买）	ap-south-1	是
Asia Pacific (Osaka)	ap-northeast-3	是
Asia Pacific (Seoul)	ap-northeast-2	是
亚太地区（新加坡）	ap-southeast-1	是
亚太地区（悉尼）	ap-southeast-2	是
Asia Pacific (Tokyo)	ap-northeast-1	是
加拿大（中部）	ca-central-1	是
欧洲（法兰克福）	eu-central-1	是
欧洲地区（爱尔兰）	eu-west-1	是
欧洲地区（伦敦）	eu-west-2	是
欧洲（巴黎）	eu-west-3	是
欧洲地区（斯德哥尔摩）	eu-north-1	是
南美洲（圣保罗）	sa-east-1	是
AWS GovCloud (US)	us-gov-west-1	否

在 Systems Manager OpsCenter 中使用角色创建运营洞察 OpsItem

Systems Manager 使用名为 **AWSServiceRoleForAmazonSSM_OpsInsights** 的服务相关角色。AWS Systems Manager 使用此 IAM 服务角色在 Systems Manager OpsCenter 中创建和更新运营洞察 OpsItem。

Systems Manager 运营洞察 OpsItem 的 **AWSServiceRoleForAmazonSSM_OpsInsights** 服务相关角色权限

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

- opsinsights.ssm.amazonaws.com

角色权限策略允许 Systems Manager 对指定资源完成以下操作：

JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "AllowCreateOpsItem",
   "Effect": "Allow",
   "Action": [
    "ssm:CreateOpsItem",
    "ssm:AddTagsToResource"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AllowAccessOpsItem",
   "Effect": "Allow",
   "Action": [
    "ssm:UpdateOpsItem",
    "ssm:GetOpsItem"
   ],
   "Resource": "*",
   "Condition": {
    "StringEquals": {
     "aws:ResourceTag/SsmOperationalInsight": "true"
    }
```

```
    }
   }
  ]
}
```

您必须配置权限，允许 IAM 实体（如用户、组或角色）创建、编辑或删除服务相关角色。有关更多信息，请参阅《IAM 用户指南》中的服务相关角色权限。

创建 Systems Manager 的 **AWSServiceRoleForAmazonSSM_OpsInsights** 服务相关角色

您必须创建服务相关角色。如果您在 AWS Management Console中使用 Systems Manager 启用运营洞察，则可以通过选择 Enable (启用) 按钮创建服务相关角色。

编辑 Systems Manager 的 **AWSServiceRoleForAmazonSSM_OpsInsights** 服务相关角色

Systems Manager 不允许您编辑 AWSServiceRoleForAmazonSSM_OpsInsights 服务相关角色。创建服务相关角色后，您将无法更改角色的名称，因为可能有多种实体引用该角色。但是可以使用 IAM 编辑角色描述。有关更多信息，请参阅《IAM 用户指南》中的编辑服务相关角色。

删除 Systems Manager 的 **AWSServiceRoleForAmazonSSM_OpsInsights** 服务相关角色

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

清除 **AWSServiceRoleForAmazonSSM_OpsInsights** 服务相关角色

必须首先在 Systems Manager OpsCenter 中停用运营洞察，才能使用 IAM 删除 AWSServiceRoleForAmazonSSM_OpsInsights 服务相关角色。有关更多信息，请参阅 分析运营洞察以减少 OpsItems。

手动删除 **AWSServiceRoleForAmazonSSM_OpsInsights** 服务相关角色

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

支持 Systems Manager**AWSServiceRoleForAmazonSSM_OpsInsights** 服务相关角色的区域

Systems Manager 并非在提供该服务的每个区域中都支持使用服务相关角色。您可以在下列区域使用 AWSServiceRoleForAmazonSSM_OpsInsights 角色。

区域名称	区域标识	在 Systems Manager 中支持
美国东部（弗吉尼亚州北部）	us-east-1	是
美国东部（俄亥俄州）	us-east-2	是
美国西部（加利福尼亚北部）	us-west-1	是
美国西部（俄勒冈州）	us-west-2	是
亚太地区（孟买）	ap-south-1	是
亚太地区（东京）	ap-northeast-1	是
亚太地区（首尔）	ap-northeast-2	是
亚太地区（新加坡）	ap-southeast-1	是
亚太地区（悉尼）	ap-southeast-2	是
亚太地区（香港）	ap-east-1	是
加拿大（中部）	ca-central-1	是
欧洲（法兰克福）	eu-central-1	是
欧洲地区（爱尔兰）	eu-west-1	是
欧洲地区（伦敦）	eu-west-2	是
欧洲（巴黎）	eu-west-3	是
欧洲地区（斯德哥尔摩）	eu-north-1	是

区域名称	区域标识	在 Systems Manager 中支持
欧洲（米兰）	eu-south-1	是
南美洲（圣保罗）	sa-east-1	是
中东（巴林）	me-south-1	是
非洲（开普敦）	af-south-1	是
AWS GovCloud (US)	us-gov-west-1	是
AWS GovCloud (US)	us-gov-east-1	是

使用角色来维护已配置 Quick Setup 的资源运行状况和一致性

Systems Manager 使用名为 **AWSServiceRoleForSSMQuickSetup** 的服务相关角色。

Systems Manager 的 **AWSServiceRoleForSSMQuickSetup** 服务相关角色权限

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

- ssm-quicksetup.amazonaws.com

AWS Systems Manager 使用此 IAM 服务角色来检查配置运行状况，确保参数和已配置资源的一致使用，并在检测到偏差时修复资源。

角色权限策略允许 Systems Manager 对指定资源完成以下操作：

- ssm (Systems Manager)：读取有关已配置资源预期处于的状态的信息，包括在委派管理员账户中。
- iam (AWS Identity and Access Management)：这是为了在 AWS Organizations 中的整个组织都可访问资源数据同步所必需的。
- organizations（AWS Organizations）– 读取属于组织的成员账户的信息，如“组织”中所配置。
- cloudformation（AWS CloudFormation）– 读取有关用于管理资源状态和 CloudFormation 堆栈集操作的 CloudFormation 堆栈的信息。

用于为 AWSServiceRoleForSSMQuickSetup 角色提供权限的托管策略是 SSMQuickSetupRolePolicy。有关该策略授予的权限的详细信息，请参阅 AWS 托管式策略：SSMQuickSetupRolePolicy。

您必须配置权限，允许 IAM 实体（如用户、组或角色）创建、编辑或删除服务相关角色。有关更多信息，请参阅《IAM 用户指南》中的服务相关角色权限。

创建 Systems Manager 的 **AWSServiceRoleForSSMQuickSetup** 服务相关角色

您无需手动创建 AWSServiceRoleForSSMQuickSetup 服务相关角色。当您在 AWS Management Console 中创建 Quick Setup 配置时，Systems Manager 会为您创建服务相关角色。

编辑 Systems Manager 的 **AWSServiceRoleForSSMQuickSetup** 服务相关角色

Systems Manager 不允许您编辑 AWSServiceRoleForSSMQuickSetup 服务相关角色。创建服务相关角色后，您将无法更改角色的名称，因为可能有多种实体引用该角色。但是可以使用 IAM 编辑角色描述。有关更多信息，请参阅《IAM 用户指南》中的编辑服务相关角色。

删除 Systems Manager 的 **AWSServiceRoleForSSMQuickSetup** 服务相关角色

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

清除 **AWSServiceRoleForSSMQuickSetup** 服务相关角色

在使用 IAM 删除 AWSServiceRoleForSSMQuickSetup 服务相关角色之前，必须先删除使用该角色的 Quick Setup 配置。有关更多信息，请参阅 编辑和删除配置。

手动删除 **AWSServiceRoleForSSMQuickSetup** 服务相关角色

使用 IAM 控制台、AWS CLI 或 AWS API 删除 AWSServiceRoleForSSMQuickSetup 服务相关角色。有关更多信息，请参阅以下主题：

- 《IAM 用户指南》中的删除服务相关角色
- 《AWS CLI 参考》的 Quick Setup 部分中的 delete-configuration-manager
- 《Quick Setup API 参考》中的 DeleteConfigurationManager

支持 Systems Manager**AWSServiceRoleForSSMQuickSetup** 服务相关角色的区域

Systems Manager 并非在提供该服务的每个区域中都支持使用服务相关角色。您可以在以下区域中使用 AWSServiceRoleForSSMQuickSetup 角色。

- 美国东部（俄亥俄州）
- 美国东部（弗吉尼亚州北部）
- 美国西部（加利福尼亚北部）
- 美国西部（俄勒冈州）
- 亚太地区（孟买）
- 亚太地区（首尔）
- 亚太地区（新加坡）
- 亚太地区（悉尼）
- 亚太地区（东京）
- 加拿大（中部）
- 欧洲地区（法兰克福）
- 欧洲地区（斯德哥尔摩）
- 欧洲地区（爱尔兰）
- 欧洲地区（伦敦）
- 欧洲（巴黎）
- 南美洲（圣保罗）

使用角色导出 Explorer OpsData

AWS Systems Manager Explorer 使用 AmazonsmExplorerExportRole 服务角色，通过使用 AWS-ExportOpsDataToS3 自动化运行手册导出操作数据（opsData）。

Explorer 的服务相关角色权限

AmazonSSMExplorerExportRole 服务相关角色仅信任 ssm.amazonaws.com 服务担任此角色。

您可以使用 AmazonSSMExplorerExportRole 服务相关角色，通过使用 AWS-ExportOpsDataToS3 自动化运行手册导出操作数据（opsData）。您可以从 Explorer 将 5000 个 OpsData 项目以逗号分隔值（.csv）文件的形式导出到 Amazon Simple Storage Service（Amazon S3）存储桶。

角色权限策略允许 Systems Manager 对指定资源完成以下操作：

- s3:PutObject
- s3:GetBucketAcl

- s3:GetBucketLocation
- sns:Publish
- logs:DescribeLogGroups
- logs:DescribeLogStreams
- logs:CreateLogGroup
- logs:PutLogEvents
- logs:CreateLogStream
- ssm:GetOpsSummary

您必须配置权限，允许 IAM 实体（如用户、组或角色）创建、编辑或删除服务相关角色。有关更多信息，请参阅《IAM 用户指南》中的服务相关角色权限。

创建 Systems Manager 的 **AmazonSSMExplorerExportRole** 服务相关角色

当您在 Systems Manager 控制台中使用 Explorer 导出 OpsData 时，Systems Manager 会创建 AmazonSSMExplorerExportRole 服务相关角色。有关更多信息，请参阅 从 Systems Manager Explorer 中导出 OpsData。

如果您删除该服务相关角色，然后需要再次创建，您可以使用相同流程在账户中重新创建此角色。

编辑 Systems Manager 的 **AmazonSSMExplorerExportRole** 服务相关角色

Systems Manager 不允许您编辑 AmazonSSMExplorerExportRole 服务相关角色。在创建服务相关角色后，您将无法更改角色的名称，因为可能有多种实体引用该角色。不过，您可以使用 IAM 编辑角色的说明。有关更多信息，请参阅《IAM 用户指南》中的编辑服务相关角色。

删除 Systems Manager 的 **AmazonSSMExplorerExportRole** 服务相关角色

如果您不再需要使用某个需要服务相关角色的功能或服务，建议您删除该角色。这样您就没有未被主动监控或维护的未使用实体。您可以使用 IAM 控制台、AWS CLI 或 IAM API 手动删除此服务相关角色。为此，您必须先手动清除服务相关角色的资源，然后才能手动删除它。

> **Note**
>
> 如果在您尝试删除标签或资源组时 Systems Manager 服务正在使用该角色，则删除操作可能会失败。如果发生这种情况，请等待几分钟后重试。

删除 **AmazonSSMExplorerExportRole** 所用的 Systems Manager 资源

1. 要删除标签，请参阅为单个资源添加和删除标签。
2. 要删除资源组，请参阅从 AWS Resource Groups 中删除组。

使用 IAM 手动删除 **AmazonSSMExplorerExportRole** 服务相关角色

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

支持 Systems Manager**AmazonSSMExplorerExportRole** 服务相关角色的区域

Systems Manager 支持提供该服务的所有 AWS 区域中使用 AmazonSSMExplorerExportRole 服务相关角色。有关更多信息，请参阅 AWS Systems Manager 终端节点和限额。

使用角色启用即时节点访问

Systems Manager 使用名为 **AWSServiceRoleForSystemsManagerJustInTimeAccess** 的服务相关角色。AWS Systems Manager 使用此 IAM 服务角色启用即时节点访问。

Systems Manager 即时节点访问的服务相关角色权限

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

- ssm.amazonaws.com

角色权限策略允许 Systems Manager 对指定资源完成以下操作：

- ssm:CreateOpsItem
- ssm:GetOpsItem
- ssm:UpdateOpsItem
- ssm:DescribeOpsItems
- ssm:DescribeSessions
- ssm:ListTagsForResource
- ssm-guiconnect:ListConnections
- identitystore:ListGroupMembershipsForMember

- identitystore:DescribeUser
- identitystore:GetGroupId
- identitystore:GetUserId
- sso-directory:DescribeUsers
- sso-directory:IsMemberInGroup
- sso:ListInstances
- sso:DescribeRegisteredRegions
- sso:ListDirectoryAssociations
- ec2:DescribeTags

用于为 AWSServiceRoleForSystemsManagerJustInTimeAccess 角色提供权限的托管策略是 AWSSystemsManagerEnableJustInTimeAccessPolicy。有关该策略授予的权限的详细信息，请参阅 AWS 托管策略：AWSSystemsManagerJustInTimeAccessServicePolicy。

您必须配置权限，允许 IAM 实体（如用户、组或角色）创建、编辑或删除服务相关角色。有关更多信息，请参阅《IAM 用户指南》中的服务相关角色权限。

创建 Systems Manager 的 **AWSServiceRoleForSystemsManagerJustInTimeAccess** 服务相关角色

您无需手动创建服务相关角色。您在 AWS Management Console中启用即时节点访问时，Systems Manager 会为您创建服务相关角色。

> **Important**
>
> 如果您在使用此服务相关角色支持的功能的其他服务中完成某个操作，该角色可以显示在您的账户中。此外，如果您在 2024 年 11 月 19 日之前使用 Systems Manager 服务，当它开始支持服务相关角色时，Systems Manager 会在您的账户中创建 AWSServiceRoleForSystemsManagerJustInTimeAccess 角色。要了解更多信息，请参阅我的 IAM 账户中出现新角色。

如果您删除该服务相关角色，然后需要再次创建，您可以使用相同流程在账户中重新创建此角色。您在 AWS Management Console中启用即时节点访问时，Systems Manager 将再次为您创建服务相关角色。

您也可以使用 IAM 控制台来创建服务相关角色，以便 AWS 服务角色允许 Systems Manager 启用即时节点访问。应用场景。在 AWS CLI 或 AWS API 中，使用 `ssm.amazonaws.com` 服务名称创建服务相关角色。有关更多信息，请参阅《IAM 用户指南》中的创建服务相关角色。如果您删除了此服务相关角色，可以使用同样的过程再次创建角色。

编辑 Systems Manager 的 **AWSServiceRoleForSystemsManagerJustInTimeAccess** 服务相关角色

Systems Manager 不允许您编辑 `AWSServiceRoleForSystemsManagerJustInTimeAccess` 服务相关角色。在创建服务相关角色后，您将无法更改角色的名称，因为可能有多种实体引用该角色。不过，您可以使用 IAM 编辑角色的说明。有关更多信息，请参阅《IAM 用户指南》中的编辑服务相关角色。

删除 Systems Manager 的 **AWSServiceRoleForSystemsManagerJustInTimeAccess** 服务相关角色

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

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

使用 IAM 手动删除 **AWSServiceRoleForSystemsManagerJustInTimeAccess** 服务相关角色

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

支持 Systems Manager**AWSServiceRoleForSystemsManagerJustInTimeAccess** 服务相关角色的区域

AWS 区域 名称	区域标识	在 Systems Manager 中支持
美国东部（弗吉尼亚州北部）	us-east-1	是

AWS 区域 名称	区域标识	在 Systems Manager 中支持
美国东部（俄亥俄州）	us-east-2	是
美国西部（加利福尼亚北部）	us-west-1	是
美国西部（俄勒冈州）	us-west-2	是
亚太地区（孟买）	ap-south-1	是
亚太地区（大阪）	ap-northeast-3	是
亚太地区（首尔）	ap-northeast-2	是
亚太地区（新加坡）	ap-southeast-1	是
亚太地区（悉尼）	ap-southeast-2	是
亚太地区（东京）	ap-northeast-1	是
加拿大（中部）	ca-central-1	是
欧洲（法兰克福）	eu-central-1	是
欧洲地区（爱尔兰）	eu-west-1	是
欧洲地区（伦敦）	eu-west-2	是
欧洲（巴黎）	eu-west-3	是
欧洲地区（斯德哥尔摩）	eu-north-1	是
南美洲（圣保罗）	sa-east-1	是
AWS GovCloud (US)	us-gov-west-1	否

使用角色发送即时节点访问请求通知

Systems Manager 使用名为 **`AWSServiceRoleForSystemsManagerNotifications`** 的服务相关角色。AWS Systems Manager 使用此 IAM 服务角色向访问请求审批者发送通知。

Systems Manager 即时节点访问通知的服务相关角色权限

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

- ssm.amazonaws.com

角色权限策略允许 Systems Manager 对指定资源完成以下操作：

- identitystore:ListGroupMembershipsForMember
- identitystore:ListGroupMemberships
- identitystore:DescribeUser
- sso:ListInstances
- sso:DescribeRegisteredRegions
- sso:ListDirectoryAssociations
- sso-directory:DescribeUser
- sso-directory:ListMembersInGroup
- iam:GetRole

用于为 AWSServiceRoleForSystemsManagerNotifications 角色提供权限的托管策略是 AWSSystemsManagerNotificationsServicePolicy。有关该策略授予的权限的详细信息，请参阅 AWS 托管策略：AWSSystemsManagerNotificationsServicePolicy。

您必须配置权限，允许 IAM 实体（如用户、组或角色）创建、编辑或删除服务相关角色。有关更多信息，请参阅《IAM 用户指南》中的服务相关角色权限。

创建 Systems Manager 的 **AWSServiceRoleForSystemsManagerNotifications** 服务相关角色

您无需手动创建服务相关角色。您在 AWS Management Console中启用即时节点访问时，Systems Manager 会为您创建服务相关角色。

> **Important**
>
> 如果您在使用此服务相关角色支持的功能的其他服务中完成某个操作，该角色可以显示在您的账户中。此外，如果您在 2024 年 11 月 19 日之前使用 Systems Manager

服务，当它开始支持服务相关角色时，Systems Manager 会在您的账户中创建 AWSServiceRoleForSystemsManagerNotifications 角色。要了解更多信息，请参阅我的 IAM 账户中出现新角色。

如果您删除该服务相关角色，然后需要再次创建，您可以使用相同流程在账户中重新创建此角色。您在 AWS Management Console中启用即时节点访问时，Systems Manager 将再次为您创建服务相关角色。

您也可以使用 IAM 控制台来创建服务相关角色，以便 AWS 服务角色允许 Systems Manager 向访问请求审批者发送通知。应用场景。在 AWS CLI 或 AWS API 中，使用 ssm.amazonaws.com 服务名称创建服务相关角色。有关更多信息，请参阅《IAM 用户指南》中的创建服务相关角色。如果您删除了此服务相关角色，可以使用同样的过程再次创建角色。

编辑 Systems Manager 的 **AWSServiceRoleForSystemsManagerNotifications** 服务相关角色

Systems Manager 不允许您编辑 AWSServiceRoleForSystemsManagerNotifications 服务相关角色。在创建服务相关角色后，您将无法更改角色的名称，因为可能有多种实体引用该角色。不过，您可以使用 IAM 编辑角色的说明。有关更多信息，请参阅《IAM 用户指南》中的编辑服务相关角色。

删除 Systems Manager 的 **AWSServiceRoleForSystemsManagerNotifications** 服务相关角色

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

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

使用 IAM 手动删除 **AWSServiceRoleForSystemsManagerNotifications** 服务相关角色

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

支持 Systems Manager**AWSServiceRoleForSystemsManagerNotifications** 服务相关角色的区域

AWS 区域 名称	区域标识	在 Systems Manager 中支持
美国东部（弗吉尼亚州北部）	us-east-1	是
美国东部（俄亥俄州）	us-east-2	是
美国西部（加利福尼亚北部）	us-west-1	是
美国西部（俄勒冈州）	us-west-2	是
亚太地区（孟买）	ap-south-1	是
亚太地区（大阪）	ap-northeast-3	是
亚太地区（首尔）	ap-northeast-2	是
亚太地区（新加坡）	ap-southeast-1	是
亚太地区（悉尼）	ap-southeast-2	是
亚太地区（东京）	ap-northeast-1	是
加拿大（中部）	ca-central-1	是
欧洲（法兰克福）	eu-central-1	是
欧洲地区（爱尔兰）	eu-west-1	是
欧洲地区（伦敦）	eu-west-2	是
欧洲（巴黎）	eu-west-3	是
欧洲地区（斯德哥尔摩）	eu-north-1	是
南美洲（圣保罗）	sa-east-1	是
AWS GovCloud (US)	us-gov-west-1	否

AWS Systems Manager 中的日志记录和监控

监控是保持 AWS Systems Manager 和您的 AWS 解决方案的可靠性、可用性和性能的重要方面。您应该从 AWS 解决方案的各个部分收集监控数据，以便在发生多点故障时进行更多的调试。AWS 提供了多种工具来监控您的 Systems Manager 和其他资源，并对潜在事件做出响应。

AWS CloudTrail 日志

CloudTrail 提供了用户、角色或 AWS 服务在 Systems Manager 中所执行操作的记录。使用 CloudTrail 收集的信息，您可以确定向 Systems Manager 发出了什么请求、发出请求的 IP 地址、何人发出的请求、请求的发出时间以及其他详细信息。有关更多信息，请参阅 使用 AWS CloudTrail 记录 AWS Systems Manager API 调用。

Amazon CloudWatch 警报

使用 Amazon CloudWatch 告警，您可以在为 Amazon Elastic Compute Cloud (Amazon EC2) 实例和其他资源指定的时间段内监控某个指标。如果指标超过给定阈值，则会向 Amazon Simple Notification Service (Amazon SNS) 主题或 AWS Auto Scaling 策略发送通知。CloudWatch 告警将不会调用操作，因为这些操作处于特定状态。而是必须在状态已改变并在指定的若干个时间段内保持不变后才调用。有关更多信息，请参阅 Amazon CloudWatch 用户指南中的使用 Amazon CloudWatch 警报。

Amazon CloudWatch 控制面板

CloudWatch 控制面板是 CloudWatch 控制台中的可自定义主页，可用于在单一视图中监控资源，即便是分布到不同 AWS 区域的资源，也能对其进行监控。您可以使用 CloudWatch 控制面板创建 AWS 资源的指标和告警的自定义视图。有关更多信息，请参阅 使用 Systems Manager 托管的 Amazon CloudWatch 控制面板。

Amazon EventBridge

使用 Amazon EventBridge，您可以配置规则以提示您 Systems Manager 资源中的更改，并指示 EventBridge 根据这些事件的内容执行操作。EventBridge 提供对由各项 Systems Manager 工具发出的大量事件的支持。有关更多信息，请参阅 使用 Amazon EventBridge 监控 Systems Manager 事件。

Amazon CloudWatch Logs 和 SSM Agent 日志

SSM Agent 将有关执行、计划操作、错误和运行状况的信息写入每个节点上的日志文件。您可以通过手动连接到节点来查看日志文件。我们建议将代理日志数据自动发送到 CloudWatch Logs 中的日志组以进行分析。有关更多信息，请参阅将节点日志发送到统一的 CloudWatch Logs（CloudWatch 代理）和查看 SSM Agent 日志。

AWS Systems Manager Compliance

您可以使用 Compliance（AWS Systems Manager 中的一项工具）扫描托管式节点实例集，了解补丁合规性和配置不一致性。您可以从多个 AWS 账户 和 AWS 区域 中收集并聚合数据，然后深入了解不合规的特定资源。默认情况下，Compliance 会显示关于Patch Manager（AWS Systems Manager 中的一项工具）中的修补以及State Manager（AWS Systems Manager 中的一项工具）中的关联的当前合规性数据。有关更多信息，请参阅 AWS Systems Manager Compliance。

AWS Systems Manager Explorer

Explorer（AWS Systems Manager 中的一项工具）是一个可自定义的操作控制面板，用于报告有关 AWS 资源的信息。Explorer 可以显示您 AWS 账户和不同 AWS 区域中的操作数据（OpsData）的聚合视图。在 Explorer 中，OpsData 包含有关 EC2 实例、补丁合规性详细信息和操作工作项 (OpsItems) 的元数据。Explorer 提供有关如何在业务单位或应用程序之间分配 OpsItems、它们随时间的变化趋势以及它们如何随类别变化的上下文。您可以在 Explorer 中对信息进行分组和筛选，以将重点放在与您相关的项目和需要采取措施的项目上。有关更多信息，请参阅 AWS Systems Manager Explorer。

AWS Systems Manager OpsCenter

OpsCenter（AWS Systems Manager 中的一项工具）提供了一个中心位置，运营工程师和 IT 专业人员可以在此处查看、调查和解决与 AWS 资源相关的操作工作项（OpsItems）。OpsCenter 聚合并标准化各种服务的 OpsItems，同时提供有关每个 OpsItem、相关 OpsItems 以及相关资源的上下文调查数据。OpsCenter 还在 Automation（AWS Systems Manager 中的一项工具）中提供运行手册，可用于快速解决问题。OpsCenter 已与 Amazon EventBridge 集成。因此，您可以创建 EventBridge 规则，从而为发布事件到 EventBridge 的任何 AWS 服务自动创建 OpsItems。有关更多信息，请参阅 AWS Systems Manager OpsCenter。

Amazon Simple Notification Service

您可以配置 Amazon Simple Notification Service（Amazon SNS），令其发送与使用 Run Command 或 Maintenance Windows（都是 AWS Systems Manager 中的工具）发送的命令的状态有关的通知。Amazon SNS 协调并管理向订阅 Amazon SNS 主题的客户端或端点发送和传输通知。您可以在命令更改为新状态或特定状态（例如 `Failed` 或 `Timed Out`）时收到通知。如果您将一条命令发送给多个节点，则对于发送给特定节点的命令的每个副本，您都可以收到通知。有关更多信息，请参阅 使用 Amazon SNS 通知监控 Systems Manager 状态更改。

AWS Trusted Advisor 和 AWS Health Dashboard

Trusted Advisor 凝聚了从为数十万 AWS 客户提供服务中总结的最佳实践。Trusted Advisor 可检查您的 AWS 环境，然后在有可能节省开支、提高系统可用性和性能或弥补安全漏洞时为您提供建议。所有 AWS 客户均有权访问五个 Trusted Advisor 检查。使用 AWS 支持 商业或企业计划的客

户可以查看所有 Trusted Advisor 检查。有关更多信息，请参阅《AWS 支持 用户指南》和《AWS Health 用户指南》中的 AWS Trusted Advisor。

更多信息

- AWS Systems Manager 中的日志记录和监控

AWS Systems Manager 的合规性验证

本主题介绍如何确保第三方保证计划的 AWS Systems Manager 合规性。有关查看托管式节点的合规性数据的信息，请参阅 AWS Systems Manager Compliance。

作为多个 AWS 合规性计划的一部分，第三方审计员将评估 Systems Manager 的安全性和合规性。其中包括 SOC、PCI、FedRAMP、HIPAA 及其他。

有关特定合规性计划范围内的 AWS 服务列表，请参阅合规性计划范围内的 AWS 服务。有关常规信息，请参阅 AWS 合规性计划 。

您可以使用 AWS Artifact 下载第三方审计报告。有关更多信息，请参阅在 AWS Artifact 中下载报告。

您使用 Systems Manager 的合规性责任取决于您数据的敏感度、贵公司的合规性目标以及适用的法律法规。AWS 提供以下资源来帮助满足合规性：

- 安全性与合规性 Quick Start 指南 - 这些部署指南讨论了架构注意事项，并提供了在AWS上部署基于安全性和合规性的基准环境的步骤。
- 设计符合 HIPAA 安全性和合规性要求的架构白皮书：此白皮书介绍公司如何使用 AWS 创建符合 HIPAA 标准的应用程序。
- AWS 合规性资源 – 此业务手册和指南集合可能适用于您的行业和位置。
- AWS Config 开发人员指南中的使用规则评估资源 – 此 AWS Config 服务评估您的资源配置对内部实践、行业指南和法规的遵循情况。
- AWS Security Hub：此 AWS 服务 提供了 AWS 中安全状态的全面视图，可帮助您检查是否符合安全行业标准和最佳实践规范。

AWS Systems Manager 中的故障恢复能力

AWS 全球基础设施围绕 AWS 区域和可用区构建。AWS 区域提供多个在物理上独立且隔离的可用区，这些可用区与延迟率低、吞吐量高且冗余性高的网络连接在一起。利用可用区，您可以设计和操作在可用区之间无中断地自动实现失效转移的应用程序和数据库。与传统的单个或多个数据中心基础设施相比，可用区具有更高的可用性、容错性和可扩展性。

有关AWS 区域和可用区的更多信息，请参阅AWS全球基础设施。

AWS Systems Manager 中的基础设施安全性

作为一项托管式服务，AWS Systems Manager 受 AWS 全球网络安全保护。有关 AWS 安全服务以及 AWS 如何保护基础设施的信息，请参阅 AWS 云安全。要按照基础架构安全最佳实践设计您的 AWS 环境，请参阅《安全性支柱 AWS Well-Architected Framework》中的 基础架构保护。

您可以使用 AWS 发布的 API 调用通过网络访问 Systems Manager。客户端必须支持以下内容：

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

此外，必须使用访问密钥 ID 和与 IAM 主体关联的秘密访问密钥来对请求进行签名。或者，您可以使用 AWS Security Token Service（AWS STS）生成临时安全凭证来对请求进行签名。

AWS Systems Manager 中的配置和漏洞分析

AWS 负责处理防火墙配置和灾难恢复等基本安全任务。这些流程已通过相应第三方审核和认证。有关更多详细信息，请参阅以下资源：

- AWS Systems Manager 的合规性验证
- 责任共担模式
- 安全性、身份和合规性最佳实践

Systems Manager 的安全最佳实践

AWS Systems Manager 提供了在您开发和实施自己的安全策略时需要考虑的大量安全功能。以下最佳实践是一般指导原则，并不代表完整安全解决方案。这些最佳实践可能不适合环境或不满足环境要求，请将其视为有用的考虑因素而不是惯例。

主题

- Systems Manager 预防性安全最佳实践
- Systems Manager 监控和审计最佳实践

Systems Manager 预防性安全最佳实践

Systems Manager 的以下最佳实践可以帮助防止安全事故。

实施最低权限访问

在授予权限时，您可以决定谁获得哪些 Systems Manager 资源的哪些权限。您可以允许对这些资源启用希望允许的特定操作。因此，您应仅授予执行任务所需的权限。实施最低权限访问对于减小安全风险以及可能由错误或恶意意图造成的影响至关重要。

为实现最低权限访问，可以使用以下工具：

- IAM 策略和 IAM 实体的权限边界
- 服务控制策略

配置为使用代理时，请使用 SSM Agent 的建议设置

如果将 SSM Agent 配置为使用代理，请将 no_proxy 变量与 Systems Manager 实例元数据服务的 IP 地址配合使用，以确保对 Systems Manager 的调用不会采用代理服务的标识。

有关更多信息，请参阅 配置 SSM Agent 以在 Linux 节点上使用代理 和 配置 SSM Agent以使用 Windows Server 实例的代理。

使用 SecureString 参数加密和保护密钥数据

在 Parameter Store（AWS Systems Manager 中的一项工具）中，SecureString 参数是需要以安全的方式存储和引用的任何敏感数据。如果您有不希望用户更改或以明文形式引用的数据（例如密码或许可证密钥），则应使用 SecureString 数据类型创建这些参数。Parameter Store 使用 AWS Key Management Service（AWS KMS）中的 AWS KMS key 加密参数值。在加密参数值时，AWS KMS 使用客户托管密钥或 AWS 托管式密钥。为了获得最大的安全性，我们建议您使用自己的 KMS 密钥。如果您使用 AWS 托管式密钥，则有权在您的账户中运行 GetParameter 和 GetParameters 操作的任何用户均有权查看或检索所有 SecureString 参数的内容。如果您使用客户托管密钥加密 SecureString 值，则可使用 IAM 策略和密钥策略来管理加密和解密参数的权限。

使用 AWS 托管式密钥 时，为这些操作建立访问控制策略更加困难。例如，如果您使用 AWS 托管式密钥 来加密 SecureString 参数，并且不希望用户使用 SecureString 参数，那么用户的 IAM 策略必须明确拒绝对默认密钥的访问权限。

有关更多信息，请参阅《AWS Key Management Service 开发人员指南》中的 使用 IAM 策略限制对 Parameter Store 参数的访问 和 AWS Systems ManagerParameter Store 如何使用 AWS KMS。

为文档参数定义 allowedValues 和 allowedPattern

您可以通过定义 `allowedValues` 和 `allowedPattern` 来验证用户在 Systems Manager 文档（SSM 文档）中的参数输入。对于 `allowedValues`，需要定义参数允许的值数组。如果用户输入了不允许的值，则执行将无法启动。对于 `allowedPattern`，需要定义一个正则表达式，用于验证用户输入是否与参数的定义模式匹配。如果用户输入与允许的模式不匹配，则执行无法启动。

有关 `allowedValues` 和 `allowedPattern` 的更多信息，请参阅 数据元素和参数。

阻止文档公开分享

除非您的应用场景需要允许公开共享，否则我们建议在 Systems Manager 文档控制台的首选项部分，为您的 SSM 文档启用阻止公开共享设置。

使用 Amazon Virtual Private Cloud（Amazon VPC）和 VPC 端点

您可以使用 Amazon VPC 在已定义的虚拟网络内启动 AWS 资源。这个虚拟网络与您在数据中心中运行的传统网络极其相似，并会为您提供使用 AWS 的可扩展基础设施的优势。

通过实施 VPC 端点，您可以通过私有方式将 VPC 连接到支持的 AWS 服务 和 VPC 端点服务（由 AWS PrivateLink 提供支持），而无需互联网网关、NAT 设备、VPN 连接或 AWS Direct Connect 连接。VPC 中的实例无需公有 IP 地址便可与服务中的资源进行通信。VPC 和其他服务之间的流量不会脱离 Amazon 网络。

有关 Amazon VPC 安全性的更多信息，请参阅《Amazon VPC 用户指南》中的使用适用于 Systems Manager 的 VPC 端点提高 EC2 实例的安全性和 Amazon VPC 中的互联网络流量隐私。

使用交互式命令和特定的 SSM 会话文档限制 Session Manager 用户访问会话

Session Manager（AWS Systems Manager 中的一项工具）可向托管式节点提供多种方法来启动会话。为了实现最安全的连接，您可以要求用户使用交互式命令 方法进行连接，以将用户的交互限制为特定命令或命令序列。这有助于管理用户可以执行的交互操作。有关更多信息，请参阅 启动会话（交互式和非交互式命令）。

为了提高安全性，您可以限制 Session Manager 对特定 Amazon EC2 实例和特定 Session Manager 会话文档的访问权限。您可以使用 AWS Identity and Access Management（IAM）policy 以这种方式授予或撤消 Session Manager 访问权限。有关更多信息，请参阅 步骤 3：控制会话对托管式节点的访问。

为自动化工作流提供临时节点权限

在 Automation（AWS Systems Manager 中的一项工具）中执行工作流期间，节点可能只需要执行该工作流所需的权限，而无需其他 Systems Manager 操作的权限。例如，Automation 工作流可能需要节点在工作流期间调用特定 API 操作或访问特定 AWS 资源。如果您希望限制访问这些调用

或资源，则可以在自动化运行手册本身中为您的节点提供临时补充权限，而不是将权限添加到您的 IAM 实例配置文件中。在自动化工作流结束时，临时权限将删除。有关更多信息，请参阅《AWS 管理和治理博客》中的为 AWS Systems Manager 自动化提供临时实例权限。

使 AWS 和 Systems Manager 工具保持最新

AWS 定期发布您可在 AWS 和 Systems Manager 操作中使用的工具和插件的更新版本。将这些资源保持为最新，可确保您账户中的用户和节点能够访问这些工具中的最新功能和安全功能。

- SSM Agent – AWS Systems Manager Agent（SSM Agent）是一个 Amazon 软件，可以在 Amazon Elastic Compute Cloud（Amazon EC2）实例、本地服务器或虚拟机（VM）上安装和配置。SSM Agent 让 Systems Manager 可以更新、管理和配置这些资源。我们建议至少每两周检查一次新版本，或者应用对代理的自动更新。有关信息，请参阅 自动更新到 SSM Agent。我们还建议在更新过程中验证 SSM Agent 的签名。有关信息，请参阅验证 SSM Agent 签名。
- AWS CLI – AWS Command Line Interface（AWS CLI）是一种开源工具，让您能够在命令行 Shell 中使用命令与 AWS 服务 进行交互。要更新 AWS CLI，请运行安装 AWS CLI 时使用的相同命令。我们建议您在本地计算机上创建计划任务，根据您的操作系统来相应运行命令，至少每两周一次。有关安装命令的信息，请参阅《AWS Command Line Interface 用户指南》中的安装 AWS CLI 版本 2。
- AWS Tools for Windows PowerShell – Tools for Windows PowerShell 是一组 PowerShell 模块，根据适用于 .NET 的 AWS SDK 公开的功能构建。AWS Tools for Windows PowerShell 使您可以从 PowerShell 命令行在 AWS 资源上为操作编写脚本。随着 Tools for Windows PowerShell 的更新版本定期发布，您应更新在本地运行的版本。有关信息，请参阅《IAM policy simulator 用户指南》中的在 Windows 上更新 AWS Tools for Windows PowerShell 或在 Linux 或 macOS 上更新 AWS Tools for Windows PowerShell。
- Session Manager 插件 – 如果贵企业中具有 Session Manager 使用权限的用户想要使用 AWS CLI 连接到节点，则他们必须先在其本地计算机上安装 Session Manager 插件。若要更新插件，请运行安装插件时使用的相同命令。我们建议您在本地计算机上创建计划任务，根据您的操作系统来相应运行命令，至少每两周一次。有关信息，请参阅为 AWS CLI 安装 Session Manager 插件。
- CloudWatch 代理 – 您可以配置和使用 CloudWatch 代理，从 EC2 实例、本地实例和虚拟机 (VM) 收集指标及日志。这些日志可以发送到 Amazon CloudWatch Logs 以进行监控和分析。我们建议至少每两周检查一次新版本，或者应用对代理的自动更新。对于最简单的更新，请使用 AWS Systems Manager 快速设置。有关信息，请参阅AWS Systems Manager Quick Setup。

Systems Manager 监控和审计最佳实践

Systems Manager 的以下实践可以帮助检测潜在的安全弱点和事故。

识别和审计您的所有 Systems Manager 资源

确定您的 IT 资产是监管和安全性的一个至关重要的方面。您需要标识所有 Systems Manager 资源，以评估它们的安保状况并对潜在的薄弱领域采取措施。

使用标签编辑器确定安全性敏感或审计敏感资源，然后在您需要搜索这些资源时使用这些标签。有关更多信息，请参阅《AWS Resource Groups 用户指南》中的查找要标记的资源。

为您的 Systems Manager 资源创建资源组。有关更多信息，请参阅什么是资源组？

使用 Amazon CloudWatch 监控工具实施监控

监控是保持 Systems Manager 和您的 AWS 解决方案的可靠性、安全性、可用性和性能的重要环节。Amazon CloudWatch 提供了多种工具和服务来帮助您监控 Systems Manager 和其他 AWS 服务。有关更多信息，请参阅将节点日志发送到统一的 CloudWatch Logs（CloudWatch 代理）和使用 Amazon EventBridge 监控 Systems Manager 事件。

使用 CloudTrail

AWS CloudTrail 提供了用户、角色或 AWS 服务 在 Systems Manager 中所执行操作的记录。使用 CloudTrail 收集的信息，您可以确定向 Systems Manager 发出了什么请求、发出请求的 IP 地址、何人发出的请求、请求的发出时间以及其他详细信息。有关更多信息，请参阅 使用 AWS CloudTrail 记录 AWS Systems Manager API 调用。

启用 AWS Config

使用 AWS Config，您能够评测、审计和评估您的 AWS 资源的配置。AWS Config 监控资源配置，让您能够针对所需的安全配置评估所记录的配置。使用 AWS Config，您可以查看配置更改以及 AWS 资源之间的关系，调查详细的资源配置历史记录，并判断您的配置在整体上是否符合内部指南中所指定的配置要求。这可以帮助您简化合规性审核、安全性分析、变更管理和操作故障排除。有关更多信息，请参阅《AWS Config 开发人员指南》中的使用控制台设置 AWS Config。当指定要记录的资源类型时，确保您包括了 Systems Manager 资源。

监控 AWS 安全公告

您应该经常为您的 AWS 账户 检查在 Trusted Advisor 中发布的安全公告。您可以使用 describe-trusted-advisor-checks 以编程方式完成此操作。

此外，积极地监控向您的每一个 AWS 账户 注册的原始邮件地址。AWS 将使用该邮箱地址就可能影响您的紧急安全事件与您联系。

具有广泛影响的 AWS 操作性问题将在 AWS Service Health Dashboard 上发布。操作性问题也会通过 Personal Health Dashboard 发布给个人账户。有关更多信息，请参阅 AWS Health 文档。

更多信息

- 安全性、身份和合规性最佳实践
- 入门：在配置AWS资源时遵循最佳实践（AWS安全博客）
- IAM 中的安全最佳实践
- AWS CloudTrail 中的安全最佳实践
- Simple Storage Service（Amazon S3）的安全最佳实践
- AWS Key Management Service 的安全最佳实践

使用 SDK 的 AWS Systems Manager 的代码示例

以下代码示例显示如何将 Systems Manager 与 AWS 软件开发工具包（SDK）一起使用。

基础知识是向您展示如何在服务中执行基本操作的代码示例。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景的上下文查看操作。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

开始使用

Hello Systems Manager

以下代码示例展示了如何开始使用 Systems Manager。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.ssm.SsmClient;
import software.amazon.awssdk.services.ssm.model.DocumentFilter;
import software.amazon.awssdk.services.ssm.model.ListDocumentsRequest;
import software.amazon.awssdk.services.ssm.model.ListDocumentsResponse;

public class HelloSSM {

    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <awsAccount>
```

```
                Where:
                    awsAccount - Your AWS Account number.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String awsAccount = args[0] ;
        Region region = Region.US_EAST_1;
        SsmClient ssmClient = SsmClient.builder()
            .region(region)
            .build();

        listDocuments(ssmClient, awsAccount);
    }

    /*
    This code automatically fetches the next set of results using the `nextToken`
and
    stops once the desired maxResults (20 in this case) have been reached.
    */
    public static void listDocuments(SsmClient ssmClient, String awsAccount) {
        String nextToken = null;
        int totalDocumentsReturned = 0;
        int maxResults = 20;
        do {
            ListDocumentsRequest request = ListDocumentsRequest.builder()
                .documentFilterList(
                    DocumentFilter.builder()
                        .key("Owner")
                        .value(awsAccount)
                        .build()
                    )
                .maxResults(maxResults)
                .nextToken(nextToken)
                .build();

            ListDocumentsResponse response = ssmClient.listDocuments(request);
            response.documentIdentifiers().forEach(identifier ->
System.out.println("Document Name: " + identifier.name()));
            nextToken = response.nextToken();
            totalDocumentsReturned += response.documentIdentifiers().size();
```

```
        } while (nextToken != null && totalDocumentsReturned < maxResults);
    }
}
```

- 有关 API 详细信息，请参阅《AWS SDK for Java 2.x API 参考》中的 ListDocuments。

JavaScript

适用于 JavaScript 的 SDK（v3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import { paginateListDocuments, SSMClient } from "@aws-sdk/client-ssm";

// Call ListDocuments and display the result.
export const main = async () => {
  const client = new SSMClient();
  const listDocumentsPaginated = [];
  console.log(
    "Hello, AWS Systems Manager! Let's list some of your documents:\n",
  );
  try {
    // The paginate function is a wrapper around the base command.
    const paginator = paginateListDocuments({ client }, { MaxResults: 5 });
    for await (const page of paginator) {
      listDocumentsPaginated.push(...page.DocumentIdentifiers);
    }
  } catch (caught) {
    console.error(`There was a problem saying hello: ${caught.message}`);
    throw caught;
  }

  for (const { Name, DocumentFormat, CreatedDate } of listDocumentsPaginated) {
    console.log(`${Name} - ${DocumentFormat} - ${CreatedDate}`);
  }
};
```

```
// Call function if run directly.
import { fileURLToPath } from "node:url";
if (process.argv[1] === fileURLToPath(import.meta.url)) {
  main();
}
```

- 有关 API 详细信息，请参阅《适用于 JavaScript 的 AWS SDK API 参考》中的 ListDocuments。

Python

适用于 Python 的 SDK (Boto3)

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import boto3
from botocore.exceptions import ClientError


def hello_systems_manager(ssm_client):
    """
    Use the AWS SDK for Python (Boto3) to create an AWS Systems Manager
    client and list the first 5 documents in your account.
    This example uses the default settings specified in your shared credentials
    and config files.

    :param ssm_client: A Boto3 AWS Systems Manager Client object. This object
 wraps
                           the low-level AWS Systems Manager service API.
    """
    print("Hello, AWS Systems Manager! Let's list some of your documents:\n")

    paginator = ssm_client.get_paginator("list_documents")
    page_iterator = paginator.paginate(PaginationConfig={"MaxItems": 5})
    for page in page_iterator:
        for document in page["DocumentIdentifiers"]:
```

```
            print(f"  {document['Name']}")


if __name__ == "__main__":
    try:
        hello_systems_manager(boto3.client("ssm"))
    except ClientError as err:
        print("Hello systems manager had an error.")
        print(err.response["Error"]["Code"])
        print(err.response["Error"]["Message"])
```

- 有关 API 详细信息，请参阅《AWS SDK for Python (Boto3) API 参考》中的 ListDocuments。

代码示例

- 使用 AWS SDK 的 Systems Manager 的基本示例
  - Hello Systems Manager
  - 了解将 Systems Manager 与 AWS SDK 结合使用的基础知识
  - 使用 AWS SDK 对 Systems Manager 执行的操作
    - 将 AddTagsToResource 与 CLI 配合使用
    - 将 CancelCommand 与 CLI 配合使用
    - 将 CreateActivation 与 CLI 配合使用
    - 将 CreateAssociation 与 CLI 配合使用
    - 将 CreateAssociationBatch 与 CLI 配合使用
    - 将 CreateDocument 与 AWS SDK 或 CLI 配合使用
    - 将 CreateMaintenanceWindow 与 AWS SDK 或 CLI 配合使用
    - 将 CreateOpsItem 与 AWS SDK 或 CLI 配合使用
    - 将 CreatePatchBaseline 与 CLI 配合使用
    - 将 DeleteActivation 与 CLI 配合使用
    - 将 DeleteAssociation 与 CLI 配合使用
    - 将 DeleteDocument 与 AWS SDK 或 CLI 配合使用
    - 将 DeleteMaintenanceWindow 与 AWS SDK 或 CLI 配合使用
    - 将 DeleteOpsItem 和 AWS SDK 搭配使用

- 将 DeleteParameter 与 CLI 配合使用
- 将 DeletePatchBaseline 与 CLI 配合使用
- 将 DeregisterManagedInstance 与 CLI 配合使用
- 将 DeregisterPatchBaselineForPatchGroup 与 CLI 配合使用
- 将 DeregisterTargetFromMaintenanceWindow 与 CLI 配合使用
- 将 DeregisterTaskFromMaintenanceWindow 与 CLI 配合使用
- 将 DescribeActivations 与 CLI 配合使用
- 将 DescribeAssociation 与 CLI 配合使用
- 将 DescribeAssociationExecutionTargets 与 CLI 配合使用
- 将 DescribeAssociationExecutions 与 CLI 配合使用
- 将 DescribeAutomationExecutions 与 CLI 配合使用
- 将 DescribeAutomationStepExecutions 与 CLI 配合使用
- 将 DescribeAvailablePatches 与 CLI 配合使用
- 将 DescribeDocument 与 CLI 配合使用
- 将 DescribeDocumentPermission 与 CLI 配合使用
- 将 DescribeEffectiveInstanceAssociations 与 CLI 配合使用
- 将 DescribeEffectivePatchesForPatchBaseline 与 CLI 配合使用
- 将 DescribeInstanceAssociationsStatus 与 CLI 配合使用
- 将 DescribeInstanceInformation 与 CLI 配合使用
- 将 DescribeInstancePatchStates 与 CLI 配合使用
- 将 DescribeInstancePatchStatesForPatchGroup 与 CLI 配合使用
- 将 DescribeInstancePatches 与 CLI 配合使用
- 将 DescribeMaintenanceWindowExecutionTaskInvocations 与 CLI 配合使用
- 将 DescribeMaintenanceWindowExecutionTasks 与 CLI 配合使用
- 将 DescribeMaintenanceWindowExecutions 与 CLI 配合使用
- 将 DescribeMaintenanceWindowTargets 与 CLI 配合使用
- 将 DescribeMaintenanceWindowTasks 与 CLI 配合使用
- 将 DescribeMaintenanceWindows 与 CLI 配合使用
- 将 DescribeOpsItems 与 AWS SDK 或 CLI 配合使用
- 将 DescribeParameters 与 AWS SDK 或 CLI 配合使用

- 将 DescribePatchBaselines 与 CLI 配合使用
- 将 DescribePatchGroupState 与 CLI 配合使用
- 将 DescribePatchGroups 与 CLI 配合使用
- 将 GetAutomationExecution 与 CLI 配合使用
- 将 GetCommandInvocation 与 CLI 配合使用
- 将 GetConnectionStatus 与 CLI 配合使用
- 将 GetDefaultPatchBaseline 与 CLI 配合使用
- 将 GetDeployablePatchSnapshotForInstance 与 CLI 配合使用
- 将 GetDocument 与 CLI 配合使用
- 将 GetInventory 与 CLI 配合使用
- 将 GetInventorySchema 与 CLI 配合使用
- 将 GetMaintenanceWindow 与 CLI 配合使用
- 将 GetMaintenanceWindowExecution 与 CLI 配合使用
- 将 GetMaintenanceWindowExecutionTask 与 CLI 配合使用
- 将 GetParameter 与 AWS SDK 或 CLI 配合使用
- 将 GetParameterHistory 与 CLI 配合使用
- 将 GetParameters 与 CLI 配合使用
- 将 GetPatchBaseline 与 CLI 配合使用
- 将 GetPatchBaselineForPatchGroup 与 CLI 配合使用
- 将 ListAssociationVersions 与 CLI 配合使用
- 将 ListAssociations 与 CLI 配合使用
- 将 ListCommandInvocations 与 AWS SDK 或 CLI 配合使用
- 将 ListCommands 与 CLI 配合使用
- 将 ListComplianceItems 与 CLI 配合使用
- 将 ListComplianceSummaries 与 CLI 配合使用
- 将 ListDocumentVersions 与 CLI 配合使用
- 将 ListDocuments 与 CLI 配合使用
- 将 ListInventoryEntries 与 CLI 配合使用
- 将 ListResourceComplianceSummaries 与 CLI 配合使用
- 将 ListTagsForResource 与 CLI 配合使用

- 将 ModifyDocumentPermission 与 CLI 配合使用
- 将 PutComplianceItems 与 CLI 配合使用
- 将 PutInventory 与 CLI 配合使用
- 将 PutParameter 与 AWS SDK 或 CLI 配合使用
- 将 RegisterDefaultPatchBaseline 与 CLI 配合使用
- 将 RegisterPatchBaselineForPatchGroup 与 CLI 配合使用
- 将 RegisterTargetWithMaintenanceWindow 与 CLI 配合使用
- 将 RegisterTaskWithMaintenanceWindow 与 CLI 配合使用
- 将 RemoveTagsFromResource 与 CLI 配合使用
- 将 SendCommand 与 AWS SDK 或 CLI 配合使用
- 将 StartAutomationExecution 与 CLI 配合使用
- 将 StartSession 与 CLI 配合使用
- 将 StopAutomationExecution 与 CLI 配合使用
- 将 UpdateAssociation 与 CLI 配合使用
- 将 UpdateAssociationStatus 与 CLI 配合使用
- 将 UpdateDocument 与 CLI 配合使用
- 将 UpdateDocumentDefaultVersion 与 CLI 配合使用
- 将 UpdateMaintenanceWindow 与 AWS SDK 或 CLI 配合使用
- 将 UpdateManagedInstanceRole 与 CLI 配合使用
- 将 UpdateOpsItem 与 AWS SDK 或 CLI 配合使用
- 将 UpdatePatchBaseline 与 CLI 配合使用

使用 AWS SDK 的 Systems Manager 的基本示例

以下代码示例展示了如何将 AWS Systems Manager 的基础知识与 AWS SDK 结合使用。

示例

- Hello Systems Manager
- 了解将 Systems Manager 与 AWS SDK 结合使用的基础知识
- 使用 AWS SDK 对 Systems Manager 执行的操作
  - 将 AddTagsToResource 与 CLI 配合使用

- 将 CancelCommand 与 CLI 配合使用
- 将 CreateActivation 与 CLI 配合使用
- 将 CreateAssociation 与 CLI 配合使用
- 将 CreateAssociationBatch 与 CLI 配合使用
- 将 CreateDocument 与 AWS SDK 或 CLI 配合使用
- 将 CreateMaintenanceWindow 与 AWS SDK 或 CLI 配合使用
- 将 CreateOpsItem 与 AWS SDK 或 CLI 配合使用
- 将 CreatePatchBaseline 与 CLI 配合使用
- 将 DeleteActivation 与 CLI 配合使用
- 将 DeleteAssociation 与 CLI 配合使用
- 将 DeleteDocument 与 AWS SDK 或 CLI 配合使用
- 将 DeleteMaintenanceWindow 与 AWS SDK 或 CLI 配合使用
- 将 DeleteOpsItem 和 AWS SDK 搭配使用
- 将 DeleteParameter 与 CLI 配合使用
- 将 DeletePatchBaseline 与 CLI 配合使用
- 将 DeregisterManagedInstance 与 CLI 配合使用
- 将 DeregisterPatchBaselineForPatchGroup 与 CLI 配合使用
- 将 DeregisterTargetFromMaintenanceWindow 与 CLI 配合使用
- 将 DeregisterTaskFromMaintenanceWindow 与 CLI 配合使用
- 将 DescribeActivations 与 CLI 配合使用
- 将 DescribeAssociation 与 CLI 配合使用
- 将 DescribeAssociationExecutionTargets 与 CLI 配合使用
- 将 DescribeAssociationExecutions 与 CLI 配合使用
- 将 DescribeAutomationExecutions 与 CLI 配合使用
- 将 DescribeAutomationStepExecutions 与 CLI 配合使用
- 将 DescribeAvailablePatches 与 CLI 配合使用
- 将 DescribeDocument 与 CLI 配合使用
- 将 DescribeDocumentPermission 与 CLI 配合使用
- 将 DescribeEffectiveInstanceAssociations 与 CLI 配合使用
- 将 DescribeEffectivePatchesForPatchBaseline 与 CLI 配合使用

- 将 DescribeInstanceAssociationsStatus 与 CLI 配合使用
- 将 DescribeInstanceInformation 与 CLI 配合使用
- 将 DescribeInstancePatchStates 与 CLI 配合使用
- 将 DescribeInstancePatchStatesForPatchGroup 与 CLI 配合使用
- 将 DescribeInstancePatches 与 CLI 配合使用
- 将 DescribeMaintenanceWindowExecutionTaskInvocations 与 CLI 配合使用
- 将 DescribeMaintenanceWindowExecutionTasks 与 CLI 配合使用
- 将 DescribeMaintenanceWindowExecutions 与 CLI 配合使用
- 将 DescribeMaintenanceWindowTargets 与 CLI 配合使用
- 将 DescribeMaintenanceWindowTasks 与 CLI 配合使用
- 将 DescribeMaintenanceWindows 与 CLI 配合使用
- 将 DescribeOpsItems 与 AWS SDK 或 CLI 配合使用
- 将 DescribeParameters 与 AWS SDK 或 CLI 配合使用
- 将 DescribePatchBaselines 与 CLI 配合使用
- 将 DescribePatchGroupState 与 CLI 配合使用
- 将 DescribePatchGroups 与 CLI 配合使用
- 将 GetAutomationExecution 与 CLI 配合使用
- 将 GetCommandInvocation 与 CLI 配合使用
- 将 GetConnectionStatus 与 CLI 配合使用
- 将 GetDefaultPatchBaseline 与 CLI 配合使用
- 将 GetDeployablePatchSnapshotForInstance 与 CLI 配合使用
- 将 GetDocument 与 CLI 配合使用
- 将 GetInventory 与 CLI 配合使用
- 将 GetInventorySchema 与 CLI 配合使用
- 将 GetMaintenanceWindow 与 CLI 配合使用
- 将 GetMaintenanceWindowExecution 与 CLI 配合使用
- 将 GetMaintenanceWindowExecutionTask 与 CLI 配合使用
- 将 GetParameter 与 AWS SDK 或 CLI 配合使用
- 将 GetParameterHistory 与 CLI 配合使用
- 将 GetParameters 与 CLI 配合使用

- 将 GetPatchBaseline 与 CLI 配合使用
- 将 GetPatchBaselineForPatchGroup 与 CLI 配合使用
- 将 ListAssociationVersions 与 CLI 配合使用
- 将 ListAssociations 与 CLI 配合使用
- 将 ListCommandInvocations 与 AWS SDK 或 CLI 配合使用
- 将 ListCommands 与 CLI 配合使用
- 将 ListComplianceItems 与 CLI 配合使用
- 将 ListComplianceSummaries 与 CLI 配合使用
- 将 ListDocumentVersions 与 CLI 配合使用
- 将 ListDocuments 与 CLI 配合使用
- 将 ListInventoryEntries 与 CLI 配合使用
- 将 ListResourceComplianceSummaries 与 CLI 配合使用
- 将 ListTagsForResource 与 CLI 配合使用
- 将 ModifyDocumentPermission 与 CLI 配合使用
- 将 PutComplianceItems 与 CLI 配合使用
- 将 PutInventory 与 CLI 配合使用
- 将 PutParameter 与 AWS SDK 或 CLI 配合使用
- 将 RegisterDefaultPatchBaseline 与 CLI 配合使用
- 将 RegisterPatchBaselineForPatchGroup 与 CLI 配合使用
- 将 RegisterTargetWithMaintenanceWindow 与 CLI 配合使用
- 将 RegisterTaskWithMaintenanceWindow 与 CLI 配合使用
- 将 RemoveTagsFromResource 与 CLI 配合使用
- 将 SendCommand 与 AWS SDK 或 CLI 配合使用
- 将 StartAutomationExecution 与 CLI 配合使用
- 将 StartSession 与 CLI 配合使用
- 将 StopAutomationExecution 与 CLI 配合使用
- 将 UpdateAssociation 与 CLI 配合使用
- 将 UpdateAssociationStatus 与 CLI 配合使用
- 将 UpdateDocument 与 CLI 配合使用
- 将 UpdateDocumentDefaultVersion 与 CLI 配合使用

- 将 UpdateMaintenanceWindow 与 AWS SDK 或 CLI 配合使用
- 将 UpdateManagedInstanceRole 与 CLI 配合使用
- 将 UpdateOpsItem 与 AWS SDK 或 CLI 配合使用
- 将 UpdatePatchBaseline 与 CLI 配合使用

Hello Systems Manager

以下代码示例展示了如何开始使用 Systems Manager。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.ssm.SsmClient;
import software.amazon.awssdk.services.ssm.model.DocumentFilter;
import software.amazon.awssdk.services.ssm.model.ListDocumentsRequest;
import software.amazon.awssdk.services.ssm.model.ListDocumentsResponse;

public class HelloSSM {

    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <awsAccount>

                Where:
                    awsAccount - Your AWS Account number.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
```

```
        }

        String awsAccount = args[0] ;
        Region region = Region.US_EAST_1;
        SsmClient ssmClient = SsmClient.builder()
            .region(region)
            .build();

        listDocuments(ssmClient, awsAccount);
    }

    /*
    This code automatically fetches the next set of results using the `nextToken`
 and
    stops once the desired maxResults (20 in this case) have been reached.
    */
    public static void listDocuments(SsmClient ssmClient, String awsAccount) {
        String nextToken = null;
        int totalDocumentsReturned = 0;
        int maxResults = 20;
        do {
            ListDocumentsRequest request = ListDocumentsRequest.builder()
                .documentFilterList(
                    DocumentFilter.builder()
                        .key("Owner")
                        .value(awsAccount)
                        .build()
                    )
                .maxResults(maxResults)
                .nextToken(nextToken)
                .build();

            ListDocumentsResponse response = ssmClient.listDocuments(request);
            response.documentIdentifiers().forEach(identifier ->
 System.out.println("Document Name: " + identifier.name()));
            nextToken = response.nextToken();
            totalDocumentsReturned += response.documentIdentifiers().size();
        } while (nextToken != null && totalDocumentsReturned < maxResults);
    }
}
```

- 有关 API 详细信息，请参阅《AWS SDK for Java 2.x API 参考》中的 ListDocuments。

JavaScript

适用于 JavaScript 的 SDK（v3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import { paginateListDocuments, SSMClient } from "@aws-sdk/client-ssm";

// Call ListDocuments and display the result.
export const main = async () => {
  const client = new SSMClient();
  const listDocumentsPaginated = [];
  console.log(
    "Hello, AWS Systems Manager! Let's list some of your documents:\n",
  );
  try {
    // The paginate function is a wrapper around the base command.
    const paginator = paginateListDocuments({ client }, { MaxResults: 5 });
    for await (const page of paginator) {
      listDocumentsPaginated.push(...page.DocumentIdentifiers);
    }
  } catch (caught) {
    console.error(`There was a problem saying hello: ${caught.message}`);
    throw caught;
  }

  for (const { Name, DocumentFormat, CreatedDate } of listDocumentsPaginated) {
    console.log(`${Name} - ${DocumentFormat} - ${CreatedDate}`);
  }
};

// Call function if run directly.
import { fileURLToPath } from "node:url";
if (process.argv[1] === fileURLToPath(import.meta.url)) {
  main();
}
```

- 有关 API 详细信息，请参阅《适用于 JavaScript 的 AWS SDK API 参考》中的 ListDocuments。

Python

适用于 Python 的 SDK (Boto3)

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import boto3
from botocore.exceptions import ClientError


def hello_systems_manager(ssm_client):
    """
    Use the AWS SDK for Python (Boto3) to create an AWS Systems Manager
    client and list the first 5 documents in your account.
    This example uses the default settings specified in your shared credentials
    and config files.

    :param ssm_client: A Boto3 AWS Systems Manager Client object. This object
 wraps
                           the low-level AWS Systems Manager service API.
    """
    print("Hello, AWS Systems Manager! Let's list some of your documents:\n")

    paginator = ssm_client.get_paginator("list_documents")
    page_iterator = paginator.paginate(PaginationConfig={"MaxItems": 5})
    for page in page_iterator:
        for document in page["DocumentIdentifiers"]:
            print(f"  {document['Name']}")


if __name__ == "__main__":
    try:
        hello_systems_manager(boto3.client("ssm"))
    except ClientError as err:
```

```
        print("Hello systems manager had an error.")
        print(err.response["Error"]["Code"])
        print(err.response["Error"]["Message"])
```

- 有关 API 详细信息，请参阅《AWS SDK for Python (Boto3) API 参考》中的 ListDocuments。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

了解将 Systems Manager 与 AWS SDK 结合使用的基础知识

以下代码示例演示了如何：

- 创建维护时段。
- 修改维护时段计划。
- 创建文档。
- 向指定的 EC2 实例发送命令。
- 创建 OpsItem。
- 更新并解析 OpsItem。
- 删除维护时段、OpsItem 和文档。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import software.amazon.awssdk.services.ssm.model.DocumentAlreadyExistsException;
import software.amazon.awssdk.services.ssm.model.SsmException;

import java.util.Scanner;
public class SSMScenario {
```

```
   public static final String DASHES = new String(new char[80]).replace("\0",
"-");

   public static void main(String[] args) {
       String usage = """
           Usage:
             <instanceId> <title> <source> <category> <severity>

           Where:
               instanceId - The Amazon EC2 Linux/UNIX instance Id that AWS
Systems Manager uses (ie, i-0149338494ed95f06).
               title - The title of the parameter (default is Disk Space Alert).
               source - The source of the parameter (default is EC2).
               category - The category of the parameter. Valid values are
'Availability', 'Cost', 'Performance', 'Recovery', 'Security' (default is
Performance).
               severity - The severity of the parameter. Severity should be a
number from 1 to 4 (default is 2).
       """;

       if (args.length != 1) {
           System.out.println(usage);
           System.exit(1);
       }

       Scanner scanner = new Scanner(System.in);
       SSMActions actions = new SSMActions();
       String documentName;
       String windowName;
       String instanceId = args[0];
       String title = "Disk Space Alert" ;
       String source = "EC2" ;
       String category = "Availability" ;
       String severity = "2" ;

       System.out.println(DASHES);
       System.out.println("""
               Welcome to the AWS Systems Manager SDK Basics scenario.
               This Java program demonstrates how to interact with AWS Systems
Manager using the AWS SDK for Java (v2).
               AWS Systems Manager is the operations hub for your AWS
applications and resources and a secure end-to-end management solution.
               The program's primary functionalities include creating a
maintenance window, creating a document, sending a command to a document,
```

```
                listing documents, listing commands, creating an OpsItem,
modifying an OpsItem, and deleting AWS SSM resources.
                Upon completion of the program, all AWS resources are cleaned up.
                Let's get started...

                """);
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println("1. Create an SSM maintenance window.");
        System.out.println("Please enter the maintenance window name (default is
ssm-maintenance-window):");
        String win = scanner.nextLine();
        windowName = win.isEmpty() ? "ssm-maintenance-window" : win;
        String winId = null;
        try {
            winId = actions.createMaintenanceWindow(windowName);
            waitForInputToContinue(scanner);
            System.out.println("The maintenance window ID is: " + winId);
        } catch (DocumentAlreadyExistsException e) {
            System.err.println("The SSM maintenance window already exists.
Retrieving existing window ID...");
            String existingWinId = actions.createMaintenanceWindow(windowName);
            System.out.println("Existing window ID: " + existingWinId);
        } catch (SsmException e) {
            System.err.println("SSM error: " + e.getMessage());
            return;
        } catch (RuntimeException e) {
            System.err.println("Unexpected error: " + e.getMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println("2. Modify the maintenance window by changing the
schedule");
        waitForInputToContinue(scanner);
        try {
            actions.updateSSMMaintenanceWindow(winId, windowName);
            waitForInputToContinue(scanner);
            System.out.println("The SSM maintenance window was successfully
updated");
        } catch (SsmException e) {
            System.err.println("SSM error: " + e.getMessage());
```

```
            return;
        } catch (RuntimeException e) {
            System.err.println("Unexpected error: " + e.getMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println("3. Create an SSM document that defines the actions
that Systems Manager performs on your managed nodes.");
        System.out.println("Please enter the document name (default is
ssmdocument):");
        String doc = scanner.nextLine();
        documentName = doc.isEmpty() ? "ssmdocument" : doc;
        try {
            actions.createSSMDoc(documentName);
            waitForInputToContinue(scanner);
            System.out.println("The SSM document was successfully created");
        } catch (DocumentAlreadyExistsException e) {
            System.err.println("The SSM document already exists. Moving on");
        } catch (SsmException e) {
            System.err.println("SSM error: " + e.getMessage());
            return;
        } catch (RuntimeException e) {
            System.err.println("Unexpected error: " + e.getMessage());
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println("4. Now we are going to run a command on an EC2
instance");
        waitForInputToContinue(scanner);
        String commandId="";
        try {
            commandId = actions.sendSSMCommand(documentName, instanceId);
            waitForInputToContinue(scanner);
            System.out.println("The command was successfully sent. Command ID: "
+ commandId);
        } catch (SsmException e) {
            System.err.println("SSM error: " + e.getMessage());
        } catch (InterruptedException e) {
            System.err.println("Thread was interrupted: " + e.getMessage());
        } catch (RuntimeException e) {
            System.err.println("Unexpected error: " + e.getMessage());
```

```
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println("5. Lets get the time when the specific command was
sent to the specific managed node");
        waitForInputToContinue(scanner);
        try {
            actions.displayCommands(commandId);
            System.out.println("The command invocations were successfully
displayed.");
        } catch (SsmException e) {
            System.err.println("SSM error: " + e.getMessage());
            return;
        } catch (RuntimeException e) {
            System.err.println("Unexpected error: " + e.getMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
             6. Now we will create an SSM OpsItem.
             A SSM OpsItem is a feature provided by Amazon's Systems Manager
(SSM) service.
             It is a type of operational data item that allows you to manage and
track various operational issues,
             events, or tasks within your AWS environment.

             You can create OpsItems to track and manage operational issues as
they arise.
             For example, you could create an OpsItem whenever your application
detects a critical error
             or an anomaly in your infrastructure.
            """);

        waitForInputToContinue(scanner);
        String opsItemId;
        try {
            opsItemId = actions.createSSMOpsItem(title, source, category,
severity);
            System.out.println(opsItemId + " was created");
        } catch (SsmException e) {
```

```
            System.err.println("SSM error: " + e.getMessage());
            return;
        } catch (RuntimeException e) {
            System.err.println("Unexpected error: " + e.getMessage());
            return;
        }
        waitForInputToContinue(scanner);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("7. Now we will update the SSM OpsItem "+opsItemId);
        waitForInputToContinue(scanner);
        String description = "An update to "+opsItemId ;
        try {
            actions.updateOpsItem(opsItemId, title, description);
        } catch (SsmException e) {
            System.err.println("SSM error: " + e.getMessage());
            return;
        } catch (RuntimeException e) {
            System.err.println("Unexpected error: " + e.getMessage());
            return;
        }

        System.out.println(DASHES);
        System.out.println("8. Now we will get the status of the SSM OpsItem
"+opsItemId);
        waitForInputToContinue(scanner);
        try {
            actions.describeOpsItems(opsItemId);
        } catch (SsmException e) {
            System.err.println("SSM error: " + e.getMessage());
            return;
        } catch (RuntimeException e) {
            System.err.println("Unexpected error: " + e.getMessage());
            return;
        }

        System.out.println(DASHES);
        System.out.println("9. Now we will resolve the SSM OpsItem "+opsItemId);
        waitForInputToContinue(scanner);
        try {
            actions.resolveOpsItem(opsItemId);
        } catch (SsmException e) {
            System.err.println("SSM error: " + e.getMessage());
```

```
            return;
        } catch (RuntimeException e) {
            System.err.println("Unexpected error: " + e.getMessage());
            return;
        }

        System.out.println(DASHES);
        System.out.println("10. Would you like to delete the AWS Systems Manager
resources? (y/n)");
        String delAns = scanner.nextLine().trim();
        if (delAns.equalsIgnoreCase("y")) {
            System.out.println("You selected to delete the resources.");
            waitForInputToContinue(scanner);
            try {
                actions.deleteMaintenanceWindow(winId);
                actions.deleteDoc(documentName);
            } catch (SsmException e) {
                System.err.println("SSM error: " + e.getMessage());
                return;
            } catch (RuntimeException e) {
                System.err.println("Unexpected error: " + e.getMessage());
                return;
            }
        } else {
            System.out.println("The AWS Systems Manager resources will not be
deleted");
        }
        System.out.println(DASHES);

        System.out.println("This concludes the AWS Systems Manager SDK Basics
scenario.");
        System.out.println(DASHES);
    }

    private static void waitForInputToContinue(Scanner scanner) {
        while (true) {
            System.out.println("");
            System.out.println("Enter 'c' followed by <ENTER> to continue:");
            String input = scanner.nextLine();

            if (input.trim().equalsIgnoreCase("c")) {
                System.out.println("Continuing with the program...");
                System.out.println("");
                break;
```

```
            } else {
                // Handle invalid input.
                System.out.println("Invalid input. Please try again.");
            }
        }
    }
}
```

Systems Manager SDK 方法的包装器类。

```
public class SSMActions {

    private static SsmAsyncClient ssmAsyncClient;

    private static SsmAsyncClient getAsyncClient() {
        if (ssmAsyncClient == null) {
            SdkAsyncHttpClient httpClient = NettyNioAsyncHttpClient.builder()
                    .maxConcurrency(100)
                    .connectionTimeout(Duration.ofSeconds(60))
                    .readTimeout(Duration.ofSeconds(60))
                    .writeTimeout(Duration.ofSeconds(60))
                    .build();

            ClientOverrideConfiguration overrideConfig =
 ClientOverrideConfiguration.builder()
                    .apiCallTimeout(Duration.ofMinutes(2))
                    .apiCallAttemptTimeout(Duration.ofSeconds(90))
                    .retryPolicy(RetryPolicy.builder()
                            .numRetries(3)
                            .build())
                    .build();

            ssmAsyncClient = SsmAsyncClient.builder()
                    .region(Region.US_EAST_1)
                    .httpClient(httpClient)
                    .overrideConfiguration(overrideConfig)
                    .build();
        }
        return ssmAsyncClient;
    }

    /**
```

```
     * Deletes an AWS SSM document asynchronously.
     *
     * @param documentName The name of the document to delete.
     * <p>
     * This method initiates an asynchronous request to delete an SSM document.
     * If an exception occurs, it handles the error appropriately.
     */
    public void deleteDoc(String documentName) {
        DeleteDocumentRequest documentRequest = DeleteDocumentRequest.builder()
                .name(documentName)
                .build();

        CompletableFuture<Void> future = CompletableFuture.runAsync(() -> {
            getAsyncClient().deleteDocument(documentRequest)
                    .thenAccept(response -> {
                        System.out.println("The SSM document was successfully
deleted.");
                    })
                    .exceptionally(ex -> {
                        throw new CompletionException(ex);
                    }).join();
        }).exceptionally(ex -> {
            Throwable cause = (ex instanceof CompletionException) ?
ex.getCause() : ex;
            if (cause instanceof SsmException) {
                throw new RuntimeException("SSM error: " + cause.getMessage(),
cause);
            } else {
                throw new RuntimeException("Unexpected error: " +
cause.getMessage(), cause);
            }
        });

        try {
            future.join();
        } catch (CompletionException ex) {
            throw ex.getCause() instanceof RuntimeException ? (RuntimeException)
ex.getCause() : ex;
        }
    }

    /**
     * Deletes an AWS SSM Maintenance Window asynchronously.
     *
```

```
     * @param winId The ID of the Maintenance Window to delete.
     * <p>
     * This method initiates an asynchronous request to delete an SSM Maintenance
Window.
     * If an exception occurs, it handles the error appropriately.
     */
    public void deleteMaintenanceWindow(String winId) {
        DeleteMaintenanceWindowRequest windowRequest =
DeleteMaintenanceWindowRequest.builder()
                .windowId(winId)
                .build();

        CompletableFuture<Void> future = CompletableFuture.runAsync(() -> {
            getAsyncClient().deleteMaintenanceWindow(windowRequest)
                    .thenAccept(response -> {
                        System.out.println("The maintenance window was
successfully deleted.");
                    })
                    .exceptionally(ex -> {
                        throw new CompletionException(ex);
                    }).join();
        }).exceptionally(ex -> {
            Throwable cause = (ex instanceof CompletionException) ?
ex.getCause() : ex;
            if (cause instanceof SsmException) {
                throw new RuntimeException("SSM error: " + cause.getMessage(),
cause);
            } else {
                throw new RuntimeException("Unexpected error: " +
cause.getMessage(), cause);
            }
        });

        try {
            future.join();
        } catch (CompletionException ex) {
            throw ex.getCause() instanceof RuntimeException ? (RuntimeException)
ex.getCause() : ex;
        }
    }

    /**
     * Resolves an AWS SSM OpsItem asynchronously.
     *
```

```
     * @param opsID The ID of the OpsItem to resolve.
     * <p>
     * This method initiates an asynchronous request to resolve an SSM OpsItem.
     * If an exception occurs, it handles the error appropriately.
     */
    public void resolveOpsItem(String opsID) {
        UpdateOpsItemRequest opsItemRequest = UpdateOpsItemRequest.builder()
                .opsItemId(opsID)
                .status(OpsItemStatus.RESOLVED)
                .build();

        CompletableFuture<Void> future = CompletableFuture.runAsync(() -> {
            getAsyncClient().updateOpsItem(opsItemRequest)
                    .thenAccept(response -> {
                        System.out.println("OpsItem resolved successfully.");
                    })
                    .exceptionally(ex -> {
                        throw new CompletionException(ex);
                    }).join();
        }).exceptionally(ex -> {
            Throwable cause = (ex instanceof CompletionException) ?
ex.getCause() : ex;
            if (cause instanceof SsmException) {
                throw new RuntimeException("SSM error: " + cause.getMessage(),
cause);
            } else {
                throw new RuntimeException("Unexpected error: " +
cause.getMessage(), cause);
            }
        });

        try {
            future.join();
        } catch (CompletionException ex) {
            throw ex.getCause() instanceof RuntimeException ? (RuntimeException)
ex.getCause() : ex;
        }
    }

    /**
     * Describes AWS SSM OpsItems asynchronously.
     *
     * @param key The key to filter OpsItems by (e.g., OPS_ITEM_ID).
     *
```

```
    * This method initiates an asynchronous request to describe SSM OpsItems.
    * If the request is successful, it prints the title and status of each
OpsItem.
    * If an exception occurs, it handles the error appropriately.
    */
   public void describeOpsItems(String key) {
       OpsItemFilter filter = OpsItemFilter.builder()
               .key(OpsItemFilterKey.OPS_ITEM_ID)
               .values(key)
               .operator(OpsItemFilterOperator.EQUAL)
               .build();

       DescribeOpsItemsRequest itemsRequest = DescribeOpsItemsRequest.builder()
               .maxResults(10)
               .opsItemFilters(filter)
               .build();

       CompletableFuture<Void> future = CompletableFuture.runAsync(() -> {
           getAsyncClient().describeOpsItems(itemsRequest)
                   .thenAccept(itemsResponse -> {
                       List<OpsItemSummary> items =
itemsResponse.opsItemSummaries();
                       for (OpsItemSummary item : items) {
                           System.out.println("The item title is " +
item.title() + " and the status is " + item.status().toString());
                       }
                   })
                   .exceptionally(ex -> {
                       throw new CompletionException(ex);
                   }).join();
       }).exceptionally(ex -> {
           Throwable cause = (ex instanceof CompletionException) ?
ex.getCause() : ex;
           if (cause instanceof SsmException) {
               throw new RuntimeException("SSM error: " + cause.getMessage(),
cause);
           } else {
               throw new RuntimeException("Unexpected error: " +
cause.getMessage(), cause);
           }
       });

       try {
           future.join();
```

```
        } catch (CompletionException ex) {
            throw ex.getCause() instanceof RuntimeException ? (RuntimeException)
ex.getCause() : ex;
        }
    }

    /**
     * Updates the AWS SSM OpsItem asynchronously.
     *
     * @param opsItemId The ID of the OpsItem to update.
     * @param title The new title of the OpsItem.
     * @param description The new description of the OpsItem.
     * <p>
     * This method initiates an asynchronous request to update an SSM OpsItem.
     * If the request is successful, it completes without returning a value.
     * If an exception occurs, it handles the error appropriately.
     */
    public void updateOpsItem(String opsItemId, String title, String description)
{
        Map<String, OpsItemDataValue> operationalData = new HashMap<>();
        operationalData.put("key1",
OpsItemDataValue.builder().value("value1").build());
        operationalData.put("key2",
OpsItemDataValue.builder().value("value2").build());

        CompletableFuture<Void> future =
getOpsItem(opsItemId).thenCompose(opsItem -> {
            UpdateOpsItemRequest request = UpdateOpsItemRequest.builder()
                    .opsItemId(opsItemId)
                    .title(title)
                    .operationalData(operationalData)
                    .status(opsItem.statusAsString())
                    .description(description)
                    .build();

            return getAsyncClient().updateOpsItem(request).thenAccept(response ->
{
                System.out.println(opsItemId + " updated successfully.");
            }).exceptionally(ex -> {
                throw new CompletionException(ex);
            });
        }).exceptionally(ex -> {
            Throwable cause = (ex instanceof CompletionException) ?
ex.getCause() : ex;
```

```
            if (cause instanceof SsmException) {
                throw new RuntimeException("SSM error: " + cause.getMessage(),
cause);
            } else {
                throw new RuntimeException("Unexpected error: " +
cause.getMessage(), cause);
            }
        });

        try {
            future.join();
        } catch (CompletionException ex) {
            throw ex.getCause() instanceof RuntimeException ? (RuntimeException)
ex.getCause() : ex;
        }
    }


    private static CompletableFuture<OpsItem> getOpsItem(String opsItemId) {
        GetOpsItemRequest request =
GetOpsItemRequest.builder().opsItemId(opsItemId).build();
        return
getAsyncClient().getOpsItem(request).thenApply(GetOpsItemResponse::opsItem);
    }

    /**
     * Creates an SSM OpsItem asynchronously.
     *
     * @param title The title of the OpsItem.
     * @param source The source of the OpsItem.
     * @param category The category of the OpsItem.
     * @param severity The severity of the OpsItem.
     * @return The ID of the created OpsItem.
     * <p>
     * This method initiates an asynchronous request to create an SSM OpsItem.
     * If the request is successful, it returns the OpsItem ID.
     * If an exception occurs, it handles the error appropriately.
     */
    public String createSSMOpsItem(String title, String source, String category,
String severity) {
        CreateOpsItemRequest opsItemRequest = CreateOpsItemRequest.builder()
                .description("Created by the SSM Java API")
                .title(title)
                .source(source)
```

```
                .category(category)
                .severity(severity)
                .build();

        CompletableFuture<CreateOpsItemResponse> future =
getAsyncClient().createOpsItem(opsItemRequest);

        try {
            CreateOpsItemResponse response = future.join();
            return response.opsItemId();
        } catch (CompletionException e) {
            Throwable cause = e.getCause();
            if (cause instanceof SsmException) {
                throw (SsmException) cause;
            } else {
                throw new RuntimeException(cause);
            }
        }
    }

    /**
     * Displays the date and time when the specific command was invoked.
     *
     * @param commandId The ID of the command to describe.
     * <p>
     * This method initiates an asynchronous request to list command invocations
and prints the date and time of each command invocation.
     * If an exception occurs, it handles the error appropriately.
     */
    public void displayCommands(String commandId) {
        ListCommandInvocationsRequest commandInvocationsRequest =
ListCommandInvocationsRequest.builder()
                .commandId(commandId)
                .build();

        CompletableFuture<ListCommandInvocationsResponse> future =
getAsyncClient().listCommandInvocations(commandInvocationsRequest);
        future.thenAccept(response -> {
            List<CommandInvocation> commandList = response.commandInvocations();
            DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd
HH:mm:ss").withZone(ZoneId.systemDefault());
            for (CommandInvocation invocation : commandList) {
                System.out.println("The time of the command invocation is " +
formatter.format(invocation.requestedDateTime()));
```

```
            }
        }).exceptionally(ex -> {
            Throwable cause = (ex instanceof CompletionException) ?
ex.getCause() : ex;
            if (cause instanceof SsmException) {
                throw (SsmException) cause;
            } else {
                throw new RuntimeException(cause);
            }
        }).join();
    }

    /**
     * Sends a SSM command to a managed node asynchronously.
     *
     * @param documentName The name of the document to use.
     * @param instanceId The ID of the instance to send the command to.
     * @return The command ID.
     * <p>
     * This method initiates asynchronous requests to send a SSM command to a
managed node.
     * It waits until the document is active, sends the command, and checks the
command execution status.
     */
    public String sendSSMCommand(String documentName, String instanceId) throws
InterruptedException, SsmException {
        // Before we use Document to send a command - make sure it is active.
        CompletableFuture<Void> documentActiveFuture =
CompletableFuture.runAsync(() -> {
            boolean isDocumentActive = false;
            DescribeDocumentRequest request = DescribeDocumentRequest.builder()
                    .name(documentName)
                    .build();

            while (!isDocumentActive) {
                CompletableFuture<DescribeDocumentResponse> response =
getAsyncClient().describeDocument(request);
                String documentStatus =
response.join().document().statusAsString();
                if (documentStatus.equals("Active")) {
                    System.out.println("The SSM document is active and ready to
use.");
                    isDocumentActive = true;
                } else {
```

```
                    System.out.println("The SSM document is not active. Status: "
+ documentStatus);
                    try {
                        Thread.sleep(5000);
                    } catch (InterruptedException e) {
                        throw new RuntimeException(e);
                    }
                }
            }
        });

        documentActiveFuture.join();

        // Create the SendCommandRequest.
        SendCommandRequest commandRequest = SendCommandRequest.builder()
                .documentName(documentName)
                .instanceIds(instanceId)
                .build();

        // Send the command.
        CompletableFuture<SendCommandResponse> commandFuture =
getAsyncClient().sendCommand(commandRequest);
        final String[] commandId = {null};

        commandFuture.whenComplete((commandResponse, ex) -> {
            if (commandResponse != null) {
                commandId[0] = commandResponse.command().commandId();
                System.out.println("Command ID: " + commandId[0]);

                // Wait for the command execution to complete.
                GetCommandInvocationRequest invocationRequest =
GetCommandInvocationRequest.builder()
                        .commandId(commandId[0])
                        .instanceId(instanceId)
                        .build();

                try {
                    System.out.println("Wait 5 secs");
                    TimeUnit.SECONDS.sleep(5);

                    // Retrieve the command execution details.
                    CompletableFuture<GetCommandInvocationResponse>
invocationFuture = getAsyncClient().getCommandInvocation(invocationRequest);
```

```
                    invocationFuture.whenComplete((commandInvocationResponse,
invocationEx) -> {
                        if (commandInvocationResponse != null) {
                            // Check the status of the command execution.
                            CommandInvocationStatus status =
commandInvocationResponse.status();
                            if (status == CommandInvocationStatus.SUCCESS) {
                                System.out.println("Command execution
successful");
                            } else {
                                System.out.println("Command execution failed.
Status: " + status);
                            }
                        } else {
                            Throwable invocationCause = (invocationEx instanceof
CompletionException) ? invocationEx.getCause() : invocationEx;
                            throw new CompletionException(invocationCause);
                        }
                    }).join();
                } catch (InterruptedException e) {
                    throw new RuntimeException(e);
                }
            } else {
                Throwable cause = (ex instanceof CompletionException) ?
ex.getCause() : ex;
                if (cause instanceof SsmException) {
                    throw (SsmException) cause;
                } else {
                    throw new RuntimeException(cause);
                }
            }
        }).join();

        return commandId[0];
    }

    /**
     * Creates an AWS SSM document asynchronously.
     *
     * @param docName The name of the document to create.
     * <p>
     * This method initiates an asynchronous request to create an SSM document.
     * If the request is successful, it prints the document status.
     * If an exception occurs, it handles the error appropriately.
```

```
     */
    public void createSSMDoc(String docName) throws SsmException {
        String jsonData = """
        {
        "schemaVersion": "2.2",
        "description": "Run a simple shell command",
        "mainSteps": [
            {
                "action": "aws:runShellScript",
                "name": "runEchoCommand",
                "inputs": {
                  "runCommand": [
                    "echo 'Hello, world!'"
                  ]
                }
              }
            ]
        }
        """;

        CreateDocumentRequest request = CreateDocumentRequest.builder()
                .content(jsonData)
                .name(docName)
                .documentType(DocumentType.COMMAND)
                .build();

        CompletableFuture<CreateDocumentResponse> future =
getAsyncClient().createDocument(request);
        future.thenAccept(response -> {
            System.out.println("The status of the SSM document is " +
response.documentDescription().status());
        }).exceptionally(ex -> {
            Throwable cause = (ex instanceof CompletionException) ?
ex.getCause() : ex;
            if (cause instanceof DocumentAlreadyExistsException) {
                throw new CompletionException(cause);
            } else if (cause instanceof SsmException) {
                throw new CompletionException(cause);
            } else {
                throw new RuntimeException(cause);
            }
        }).join();
    }
```

```
   /**
    * Updates an SSM maintenance window asynchronously.
    *
    * @param id The ID of the maintenance window to update.
    * @param name The new name for the maintenance window.
    * <p>
    * This method initiates an asynchronous request to update an SSM maintenance
window.
    * If the request is successful, it prints a success message.
    * If an exception occurs, it handles the error appropriately.
    */
   public void updateSSMMaintenanceWindow(String id, String name) throws
SsmException {
       UpdateMaintenanceWindowRequest updateRequest =
UpdateMaintenanceWindowRequest.builder()
               .windowId(id)
               .allowUnassociatedTargets(true)
               .duration(24)
               .enabled(true)
               .name(name)
               .schedule("cron(0 0 ? * MON *)")
               .build();

       CompletableFuture<UpdateMaintenanceWindowResponse> future =
getAsyncClient().updateMaintenanceWindow(updateRequest);
       future.whenComplete((response, ex) -> {
           if (response != null) {
               System.out.println("The SSM maintenance window was successfully
updated");
           } else {
               Throwable cause = (ex instanceof CompletionException) ?
ex.getCause() : ex;
               if (cause instanceof SsmException) {
                   throw new CompletionException(cause);
               } else {
                   throw new RuntimeException(cause);
               }
           }
       }).join();
   }

   /**
    * Creates an SSM maintenance window asynchronously.
    *
```

```
    * @param winName The name of the maintenance window.
    * @return The ID of the created or existing maintenance window.
    * <p>
    * This method initiates an asynchronous request to create an SSM maintenance
window.
    * If the request is successful, it prints the maintenance window ID.
    * If an exception occurs, it handles the error appropriately.
    */
   public String createMaintenanceWindow(String winName) throws SsmException,
DocumentAlreadyExistsException {
       CreateMaintenanceWindowRequest request =
CreateMaintenanceWindowRequest.builder()
               .name(winName)
               .description("This is my maintenance window")
               .allowUnassociatedTargets(true)
               .duration(2)
               .cutoff(1)
               .schedule("cron(0 10 ? * MON-FRI *)")
               .build();

       CompletableFuture<CreateMaintenanceWindowResponse> future =
getAsyncClient().createMaintenanceWindow(request);
       final String[] windowId = {null};
       future.whenComplete((response, ex) -> {
           if (response != null) {
               String maintenanceWindowId = response.windowId();
               System.out.println("The maintenance window id is " +
maintenanceWindowId);
               windowId[0] = maintenanceWindowId;
           } else {
               Throwable cause = (ex instanceof CompletionException) ?
ex.getCause() : ex;
               if (cause instanceof DocumentAlreadyExistsException) {
                   throw new CompletionException(cause);
               } else if (cause instanceof SsmException) {
                   throw new CompletionException(cause);
               } else {
                   throw new RuntimeException(cause);
               }
           }
       }).join();

       if (windowId[0] == null) {
           MaintenanceWindowFilter filter = MaintenanceWindowFilter.builder()
```

```
                    .key("name")
                    .values(winName)
                    .build();

            DescribeMaintenanceWindowsRequest winRequest =
 DescribeMaintenanceWindowsRequest.builder()
                    .filters(filter)
                    .build();

            CompletableFuture<DescribeMaintenanceWindowsResponse> describeFuture
 = getAsyncClient().describeMaintenanceWindows(winRequest);
            describeFuture.whenComplete((describeResponse, describeEx) -> {
                if (describeResponse != null) {
                    List<MaintenanceWindowIdentity> windows =
 describeResponse.windowIdentities();
                    if (!windows.isEmpty()) {
                        windowId[0] = windows.get(0).windowId();
                        System.out.println("Window ID: " + windowId[0]);
                    } else {
                        System.out.println("Window not found.");
                        windowId[0] = "";
                    }
                } else {
                    Throwable describeCause = (describeEx instanceof
 CompletionException) ? describeEx.getCause() : describeEx;
                    throw new RuntimeException("Error describing maintenance
 windows: " + describeCause.getMessage(), describeCause);
                }
            }).join();
        }

        return windowId[0];
    }
}
```

- 有关 API 详细信息，请参阅《AWS SDK for Java 2.x API 参考》中的以下主题。
  - CreateDocument
  - CreateMaintenanceWindow
  - CreateOpsItem
  - DeleteMaintenanceWindow

- ListCommandInvocations
- SendCommand
- UpdateOpsItem

JavaScript

适用于 JavaScript 的 SDK（v3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库 中查找完整示例，了解如何进行设置和运行。

```
import {
  Scenario,
  ScenarioAction,
  ScenarioInput,
  ScenarioOutput,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { fileURLToPath } from "node:url";
import {
  CreateDocumentCommand,
  CreateMaintenanceWindowCommand,
  CreateOpsItemCommand,
  DeleteDocumentCommand,
  DeleteMaintenanceWindowCommand,
  DeleteOpsItemCommand,
  DescribeOpsItemsCommand,
  DocumentAlreadyExists,
  OpsItemStatus,
  waitUntilCommandExecuted,
  CancelCommandCommand,
  paginateListCommandInvocations,
  SendCommandCommand,
  UpdateMaintenanceWindowCommand,
  UpdateOpsItemCommand,
  SSMClient,
} from "@aws-sdk/client-ssm";
import { parseArgs } from "node:util";
```

```
/**
 * @typedef {{
 *   ssmClient: import('@aws-sdk/client-ssm').SSMClient,
 *   documentName?: string
 *   maintenanceWindow?: string
 *   winId?: int
 *   ec2InstanceId?: string
 *   requestedDateTime?: Date
 *   opsItemId?: string
 *   askToDeleteResources?: boolean
 * }} State
 */

const defaultMaintenanceWindow = "ssm-maintenance-window";
const defaultDocumentName = "ssmdocument";
// The timeout duration is highly dependent on the specific setup and environment
 necessary. This example handles only the most common error cases, and uses a
 much shorter duration than most productions systems would use.
const COMMAND_TIMEOUT_DURATION_SECONDS = 30; // 30 seconds

const pressEnter = new ScenarioInput("continue", "Press Enter to continue", {
  type: "confirm",
});

const greet = new ScenarioOutput(
  "greet",
  `Welcome to the AWS Systems Manager SDK Getting Started scenario.
    This program demonstrates how to interact with Systems Manager using the AWS
 SDK for JavaScript V3.
    Systems Manager is the operations hub for your AWS applications and resources
 and a secure end-to-end management solution.
    The program's primary functions include creating a maintenance window,
 creating a document, sending a command to a document,
    listing documents, listing commands, creating an OpsItem, modifying an
 OpsItem, and deleting Systems Manager resources.
    Upon completion of the program, all AWS resources are cleaned up.
    Let's get started...`,
  { header: true },
);

const createMaintenanceWindow = new ScenarioOutput(
  "createMaintenanceWindow",
  "Step 1: Create a Systems Manager maintenance window.",
);
```

```
const getMaintenanceWindow = new ScenarioInput(
  "maintenanceWindow",
  "Please enter the maintenance window name:",
  { type: "input", default: defaultMaintenanceWindow },
);

export const sdkCreateMaintenanceWindow = new ScenarioAction(
  "sdkCreateMaintenanceWindow",
  async (/** @type {State} */ state) => {
    try {
      const response = await state.ssmClient.send(
        new CreateMaintenanceWindowCommand({
          Name: state.maintenanceWindow,
          Schedule: "cron(0 10 ? * MON-FRI *)", //The schedule of the maintenance
 window in the form of a cron or rate expression.
          Duration: 2, //The duration of the maintenance window in hours.
          Cutoff: 1, //The number of hours before the end of the maintenance
 window that Amazon Web Services Systems Manager stops scheduling new tasks for
 execution.
          AllowUnassociatedTargets: true, //Allow the maintenance window to run
 on managed nodes, even if you haven't registered those nodes as targets.
        }),
      );
      state.winId = response.WindowId;
    } catch (caught) {
      console.error(caught.message);
      console.log(
        `An error occurred while creating the maintenance window. Please fix the
 error and try again. Error message: ${caught.message}`,
      );
      throw caught;
    }
  },
);

const modifyMaintenanceWindow = new ScenarioOutput(
  "modifyMaintenanceWindow",
  "Modify the maintenance window by changing the schedule.",
);

const sdkModifyMaintenanceWindow = new ScenarioAction(
  "sdkModifyMaintenanceWindow",
  async (/** @type {State} */ state) => {
```

```
    try {
      await state.ssmClient.send(
        new UpdateMaintenanceWindowCommand({
          WindowId: state.winId,
          Schedule: "cron(0 0 ? * MON *)",
        }),
      );
    } catch (caught) {
      console.error(caught.message);
      console.log(
        `An error occurred while modifying the maintenance window. Please fix the
 error and try again. Error message: ${caught.message}`,
      );
      throw caught;
    }
  },
);

const createSystemsManagerActions = new ScenarioOutput(
  "createSystemsManagerActions",
  "Create a document that defines the actions that Systems Manager performs on
 your EC2 instance.",
);

const getDocumentName = new ScenarioInput(
  "documentName",
  "Please enter the document: ",
  { type: "input", default: defaultDocumentName },
);

const sdkCreateSSMDoc = new ScenarioAction(
  "sdkCreateSSMDoc",
  async (/** @type {State} */ state) => {
    const contentData = `{
                "schemaVersion": "2.2",
                "description": "Run a simple shell command",
                "mainSteps": [
                    {
                        "action": "aws:runShellScript",
                        "name": "runEchoCommand",
                        "inputs": {
                          "runCommand": [
                            "echo 'Hello, world!'"
                          ]
```

```
                        }
                    }
                ]
            }`;
    try {
      await state.ssmClient.send(
        new CreateDocumentCommand({
          Content: contentData,
          Name: state.documentName,
          DocumentType: "Command",
        }),
      );
    } catch (caught) {
      console.log(`Exception type: (${typeof caught})`);
      if (caught instanceof DocumentAlreadyExists) {
        console.log("Document already exists. Continuing...\n");
      } else {
        console.error(caught.message);
        console.log(
          `An error occurred while creating the document. Please fix the error
 and try again. Error message: ${caught.message}`,
        );
        throw caught;
      }
    }
  },
);

const ec2HelloWorld = new ScenarioOutput(
  "ec2HelloWorld",
  `Now you have the option of running a command on an EC2 instance that echoes
 'Hello, world!'. In order to run this command, you must provide the instance ID
 of a Linux EC2 instance. If you do not already have a running Linux EC2 instance
 in your account, you can create one using the AWS console. For information
 about creating an EC2 instance, see https://docs.aws.amazon.com/AWSEC2/latest/
UserGuide/ec2-launch-instance-wizard.html.`,
);

const enterIdOrSkipEC2HelloWorld = new ScenarioInput(
  "enterIdOrSkipEC2HelloWorld",
  "Enter your EC2 InstanceId or press enter to skip this step: ",
  { type: "input", default: "" },
);
```

```
const sdkEC2HelloWorld = new ScenarioAction(
  "sdkEC2HelloWorld",
  async (/** @type {State} */ state) => {
    try {
      const response = await state.ssmClient.send(
        new SendCommandCommand({
          DocumentName: state.documentName,
          InstanceIds: [state.ec2InstanceId],
          TimeoutSeconds: COMMAND_TIMEOUT_DURATION_SECONDS,
        }),
      );
      state.CommandId = response.Command.CommandId;
    } catch (caught) {
      console.error(caught.message);
      console.log(
        `An error occurred while sending the command. Please fix the error and
 try again. Error message: ${caught.message}`,
      );
      throw caught;
    }
  },
  {
    skipWhen: (/** @type {State} */ state) =>
      state.enterIdOrSkipEC2HelloWorld === "",
  },
);

const sdkGetCommandTime = new ScenarioAction(
  "sdkGetCommandTime",
  async (/** @type {State} */ state) => {
    const listInvocationsPaginated = [];
    console.log(
      "Let's get the time when the specific command was sent to the specific
 managed node.",
    );

    console.log(
      `First, we'll wait for the command to finish executing. This may take up to
 ${COMMAND_TIMEOUT_DURATION_SECONDS} seconds.`,
    );
    const commandExecutedResult = waitUntilCommandExecuted(
      { client: state.ssmClient },
      {
        CommandId: state.CommandId,
```

```
        InstanceId: state.ec2InstanceId,
      },
    );
    // This is necessary because the TimeoutSeconds of SendCommandCommand is only
for the delivery, not execution.
    try {
      await new Promise((_, reject) =>
        setTimeout(
          reject,
          COMMAND_TIMEOUT_DURATION_SECONDS * 1000,
          new Error("Command Timed Out"),
        ),
      );
    } catch (caught) {
      if (caught.message === "Command Timed Out") {
        commandExecutedResult.state = "TIMED_OUT";
      } else {
        throw caught;
      }
    }

    if (commandExecutedResult.state !== "SUCCESS") {
      console.log(
        `The command with id: ${state.CommandId} did not execute in the allotted
time. Canceling command.`,
      );
      state.ssmClient.send(
        new CancelCommandCommand({
          CommandId: state.CommandId,
        }),
      );
      state.enterIdOrSkipEC2HelloWorld === "";
      return;
    }

    for await (const page of paginateListCommandInvocations(
      { client: state.ssmClient },
      { CommandId: state.CommandId },
    )) {
      listInvocationsPaginated.push(...page.CommandInvocations);
    }
    /**
     * @type {import('@aws-sdk/client-ssm').CommandInvocation}
     */
```

```
    const commandInvocation = listInvocationsPaginated.shift(); // Because the
 call was made with CommandId, there's only one result, so shift it off.
    state.requestedDateTime = commandInvocation.RequestedDateTime;

    console.log(
      `The command invocation happened at: ${state.requestedDateTime}.`,
    );
  },
  {
    skipWhen: (/** @type {State} */ state) =>
      state.enterIdOrSkipEC2HelloWorld === "",
  },
);

const createSSMOpsItem = new ScenarioOutput(
  "createSSMOpsItem",
  `Now we will create a Systems Manager OpsItem. An OpsItem is a feature provided
 by the Systems Manager service. It is a type of operational data item that
 allows you to manage and track various operational issues, events, or tasks
 within your AWS environment.
You can create OpsItems to track and manage operational issues as they arise.
 For example, you could create an OpsItem whenever your application detects a
 critical error or an anomaly in your infrastructure.`,
);

const sdkCreateSSMOpsItem = new ScenarioAction(
  "sdkCreateSSMOpsItem",
  async (/** @type {State} */ state) => {
    try {
      const response = await state.ssmClient.send(
        new CreateOpsItemCommand({
          Description: "Created by the System Manager Javascript API",
          Title: "Disk Space Alert",
          Source: "EC2",
          Category: "Performance",
          Severity: "2",
        }),
      );
      state.opsItemId = response.OpsItemId;
    } catch (caught) {
      console.error(caught.message);
      console.log(
        `An error occurred while creating the ops item. Please fix the error and
 try again. Error message: ${caught.message}`,
```

```
      );
      throw caught;
    }
  },
);

const updateOpsItem = new ScenarioOutput(
  "updateOpsItem",
  (/** @type {State} */ state) =>
    `Now we will update the OpsItem: ${state.opsItemId}`,
);

const sdkUpdateOpsItem = new ScenarioAction(
  "sdkUpdateOpsItem",
  async (/** @type {State} */ state) => {
    try {
      const _response = await state.ssmClient.send(
        new UpdateOpsItemCommand({
          OpsItemId: state.opsItemId,
          Description: `An update to ${state.opsItemId}`,
        }),
      );
    } catch (caught) {
      console.error(caught.message);
      console.log(
        `An error occurred while updating the ops item. Please fix the error and
 try again. Error message: ${caught.message}`,
      );
      throw caught;
    }
  },
);

const getOpsItemStatus = new ScenarioOutput(
  "getOpsItemStatus",
  (/** @type {State} */ state) =>
    `Now we will get the status of the OpsItem: ${state.opsItemId}`,
);

const sdkOpsItemStatus = new ScenarioAction(
  "sdkGetOpsItemStatus",
  async (/** @type {State} */ state) => {
    try {
      const response = await state.ssmClient.send(
```

```
        new DescribeOpsItemsCommand({
          OpsItemId: state.opsItemId,
        }),
      );
      state.opsItemStatus = response.OpsItemStatus;
    } catch (caught) {
      console.error(caught.message);
      console.log(
        `An error occurred while describing the ops item. Please fix the error
 and try again. Error message: ${caught.message}`,
      );
      throw caught;
    }
  },
);

const resolveOpsItem = new ScenarioOutput(
  "resolveOpsItem",
  (/** @type {State} */ state) =>
    `Now we will resolve the OpsItem: ${state.opsItemId}`,
);

const sdkResolveOpsItem = new ScenarioAction(
  "sdkResolveOpsItem",
  async (/** @type {State} */ state) => {
    try {
      const _response = await state.ssmClient.send(
        new UpdateOpsItemCommand({
          OpsItemId: state.opsItemId,
          Status: OpsItemStatus.RESOLVED,
        }),
      );
    } catch (caught) {
      console.error(caught.message);
      console.log(
        `An error occurred while updating the ops item. Please fix the error and
 try again. Error message: ${caught.message}`,
      );
      throw caught;
    }
  },
);

const askToDeleteResources = new ScenarioInput(
```

```
  "askToDeleteResources",
  "Would you like to delete the Systems Manager resources created during this
 example run?",
  { type: "confirm" },
);

const confirmDeleteChoice = new ScenarioOutput(
  "confirmDeleteChoice",
  (/** @type {State} */ state) => {
    if (state.askToDeleteResources) {
      return "You chose to delete the resources.";
    }
    return "The Systems Manager resources will not be deleted. Please delete them
 manually to avoid charges.";
  },
);

export const sdkDeleteResources = new ScenarioAction(
  "sdkDeleteResources",
  async (/** @type {State} */ state) => {
    try {
      await state.ssmClient.send(
        new DeleteOpsItemCommand({
          OpsItemId: state.opsItemId,
        }),
      );
      console.log(`The ops item: ${state.opsItemId} was successfully deleted.`);
    } catch (caught) {
      console.log(
        `There was a problem deleting the ops item: ${state.opsItemId}. Please
 delete it manually. Error: ${caught.message}`,
      );
    }

    try {
      await state.ssmClient.send(
        new DeleteMaintenanceWindowCommand({
          Name: state.maintenanceWindow,
          WindowId: state.winId,
        }),
      );
      console.log(
        `The maintenance window: ${state.maintenanceWindow} was successfully
 deleted.`,
```

```
      );
    } catch (caught) {
      console.log(
        `There was a problem deleting the maintenance window: ${state.opsItemId}.
 Please delete it manually. Error: ${caught.message}`,
      );
    }

    try {
      await state.ssmClient.send(
        new DeleteDocumentCommand({
          Name: state.documentName,
        }),
      );
      console.log(
        `The document: ${state.documentName} was successfully deleted.`,
      );
    } catch (caught) {
      console.log(
        `There was a problem deleting the document: ${state.documentName}. Please
 delete it manually. Error: ${caught.message}`,
      );
    }
  },
  { skipWhen: (/** @type {{}} */ state) => !state.askToDeleteResources },
);

const goodbye = new ScenarioOutput(
  "goodbye",
  "This concludes the Systems Manager Basics scenario for the AWS Javascript SDK
 v3. Thank you!",
);

const myScenario = new Scenario(
  "SSM Basics",
  [
    greet,
    pressEnter,
    createMaintenanceWindow,
    getMaintenanceWindow,
    sdkCreateMaintenanceWindow,
    modifyMaintenanceWindow,
    pressEnter,
    sdkModifyMaintenanceWindow,
```

```
    createSystemsManagerActions,
    getDocumentName,
    sdkCreateSSMDoc,
    ec2HelloWorld,
    enterIdOrSkipEC2HelloWorld,
    sdkEC2HelloWorld,
    sdkGetCommandTime,
    pressEnter,
    createSSMOpsItem,
    pressEnter,
    sdkCreateSSMOpsItem,
    updateOpsItem,
    pressEnter,
    sdkUpdateOpsItem,
    getOpsItemStatus,
    pressEnter,
    sdkOpsItemStatus,
    resolveOpsItem,
    pressEnter,
    sdkResolveOpsItem,
    askToDeleteResources,
    confirmDeleteChoice,
    sdkDeleteResources,
    goodbye,
  ],
  { ssmClient: new SSMClient({}) },
);

/** @type {{ stepHandlerOptions: StepHandlerOptions }} */
export const main = async (stepHandlerOptions) => {
  await myScenario.run(stepHandlerOptions);
};

// Invoke main function if this file was run directly.
if (process.argv[1] === fileURLToPath(import.meta.url)) {
  const { values } = parseArgs({
    options: {
      yes: {
        type: "boolean",
        short: "y",
      },
    },
  });
  main({ confirmAll: values.yes });
```

```
}
```

- 有关 API 详细信息，请参阅《适用于 JavaScript 的 AWS SDK API 参考》中的以下主题。
  - CreateDocument
  - CreateMaintenanceWindow
  - CreateOpsItem
  - DeleteMaintenanceWindow
  - ListCommandInvocations
  - SendCommand
  - UpdateOpsItem

Python

适用于 Python 的 SDK (Boto3)

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

在命令提示符中运行交互式场景。

```
class SystemsManagerScenario:
    """Runs an interactive scenario that shows how to get started using Amazon
 Systems Manager."""

    def __init__(self, document_wrapper, maintenance_window_wrapper,
 ops_item_wrapper):
        """
        :param document_wrapper: An object that wraps Systems Manager document
 functions.
        :param maintenance_window_wrapper: An object that wraps Systems Manager
 maintenance window functions.
        :param ops_item_wrapper: An object that wraps Systems Manager OpsItem
 functions.
        """
        self.document_wrapper = document_wrapper
```

```
        self.maintenance_window_wrapper = maintenance_window_wrapper
        self.ops_item_wrapper = ops_item_wrapper

    def run(self):
        """Demonstrates how to use the AWS SDK for Python (Boto3) to get started
 with Systems Manager."""
        try:
            print("-" * 88)
            print(
                """
Welcome to the AWS Systems Manager SDK Getting Started scenario.
This program demonstrates how to interact with Systems Manager using the AWS SDK
 for Python (Boto3).
Systems Manager is the operations hub for your AWS applications and resources and
 a secure end-to-end management
solution. The program's primary functions include creating a maintenance window,
 creating a document, sending a
command to a document, listing documents, listing commands, creating an OpsItem,
 modifying an OpsItem, and deleting
Systems Manager resources. Upon completion of the program, all AWS resources are
 cleaned up.
Let's get started..."""
            )
            q.ask("Please hit Enter")

            print("-" * 88)
            print("Create a Systems Manager maintenance window.")
            maintenance_window_name = q.ask(
                "Please enter the maintenance window name (default is ssm-
maintenance-window):",
            )
            if not maintenance_window_name:
                maintenance_window_name = "ssm-maintenance-window"

            self.maintenance_window_wrapper.create(
                name=maintenance_window_name,
                schedule="cron(0 10 ? * MON-FRI *)",
                duration=2,
                cutoff=1,
                allow_unassociated_targets=True,
            )

            print("-" * 88)
            print("Modify the maintenance window by changing the schedule")
```

```
            q.ask("Please hit Enter")

            self.maintenance_window_wrapper.update(
                name=maintenance_window_name,
                schedule="cron(0 0 ? * MON *)",
                duration=24,
                cutoff=1,
                allow_unassociated_targets=True,
                enabled=True,
            )

            print("-" * 88)
            print(
                "Create a document that defines the actions that Systems Manager
 performs on your EC2 instance."
            )
            document_name = q.ask(
                "Please enter the document name (default is ssmdocument):"
            )

            if not document_name:
                document_name = "ssmdocument"

            self.document_wrapper.create(
                name=document_name,
                content="""
{
    "schemaVersion": "2.2",
    "description": "Run a simple shell command",
    "mainSteps": [
        {
            "action": "aws:runShellScript",
            "name": "runEchoCommand",
            "inputs": {
              "runCommand": [
                "echo 'Hello, world!'"
              ]
            }
        }
    ]
}
            """,
            )
```

```
            self.document_wrapper.wait_until_active()

            print(
                """
Now you have the option of running a command on an EC2 instance that echoes
 'Hello, world!'.
In order to run this command, you must provide the instance ID of a Linux EC2
 instance. If you do
not already have a running Linux EC2 instance in your account, you can create one
 using the AWS console.
For information about creating an EC2 instance, see
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-launch-instance-
wizard.html.
            """
            )

            if q.ask(
                "Would you like to run a command on an EC2 instance? (y/n)",
                q.is_yesno,
            ):
                instance_id = q.ask(
                    "Please enter the instance ID of the EC2 instance:",
 q.non_empty
                )
                command_id = self.document_wrapper.send_command(
                    instance_ids=[instance_id]
                )

                self.document_wrapper.wait_command_executed(
                    command_id=command_id, instance_id=instance_id
                )

                print("-" * 88)
                print(
                    "Lets get the time when the specific command was sent to the
 specific managed node"
                )
                q.ask("Please hit Enter")


 self.document_wrapper.list_command_invocations(instance_id=instance_id)

            print("-" * 88)
            print("-" * 88)
```

```
            print(
                """
Now we will create a  Systems Manager OpsItem.
An OpsItem is a feature provided by the Systems Manager service.
It is a type of operational data item that allows you to manage and track various
 operational issues,
events, or tasks within your AWS environment.

You can create OpsItems to track and manage operational issues as they arise.
For example, you could create an OpsItem whenever your application detects a
 critical error
or an anomaly in your infrastructure.
            """
            )
            q.ask("Please hit Enter")

            self.ops_item_wrapper.create(
                title="Disk Space Alert",
                description="Created by the Systems Manager Python (Boto3) API",
                source="EC2",
                category="Performance",
                severity="2",
            )

            print("-" * 88)
            print("-" * 88)
            print(f"Now we will update  the OpsItem {self.ops_item_wrapper.id}")
            q.ask("Please hit Enter")

            self.ops_item_wrapper.update(
                title="Disk Space Alert",
                description=f"An update to {self.ops_item_wrapper.id}",
            )

            print(
                f"Now we will get the status of the OpsItem
 {self.ops_item_wrapper.id}"
            )
            q.ask("Please hit Enter")

            # It may take a second for the ops item to be available
            counter = 0
            while not self.ops_item_wrapper.describe() and counter < 5:
                counter += 1
```

```
                time.sleep(1)

            print(f"Now we will resolve the OpsItem {self.ops_item_wrapper.id}")
            q.ask("Please hit Enter")

            self.ops_item_wrapper.update(status="Resolved")

            print("-" * 88)
            print("-" * 88)
            if q.ask(
                "Would you like to delete the Systems Manager resources? (y/n)",
                q.is_yesno,
            ):
                print("You selected to delete the resources.")
                self.cleanup()
            else:
                print("The Systems Manager resources will not be deleted")

            print("-" * 88)
            print("This concludes the Systems Manager SDK Getting Started
 scenario.")
            print("-" * 88)

        except Exception:
            self.cleanup()
            raise

    def cleanup(self):
        self.maintenance_window_wrapper.delete()
        self.ops_item_wrapper.delete()
        self.document_wrapper.delete()


if __name__ == "__main__":
    try:
        scenario = SystemsManagerScenario(
            DocumentWrapper.from_client(),
            MaintenanceWindowWrapper.from_client(),
            OpsItemWrapper.from_client(),
        )
        scenario.run()
    except Exception:
        logging.exception("Something went wrong with the demo.")
```

定义一个包装文档操作和命令操作的类。

```
class DocumentWrapper:
    """Encapsulates AWS Systems Manager Document actions."""

    def __init__(self, ssm_client):
        """
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.name = None

    @classmethod
    def from_client(cls):
        ssm_client = boto3.client("ssm")
        return cls(ssm_client)


    def create(self, content, name):
        """
        Creates a document.

        :param content: The content of the document.
        :param name: The name of the document.
        """
        try:
            self.ssm_client.create_document(
                Name=name, Content=content, DocumentType="Command"
            )
            self.name = name
        except self.ssm_client.exceptions.DocumentAlreadyExists:
            print(f"Document {name} already exists.")
            self.name = name
        except ClientError as err:
            logger.error(
                "Couldn't create %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

```
    def delete(self):
        """
        Deletes an AWS Systems Manager document.
        """
        if self.name is None:
            return

        try:
            self.ssm_client.delete_document(Name=self.name)
            print(f"Deleted document {self.name}.")
            self.name = None
        except ClientError as err:
            logger.error(
                "Couldn't delete %s. Here's why: %s: %s",
                self.name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def send_command(self, instance_ids):
        """
        Sends a command to one or more instances.

        :param instance_ids: The IDs of the instances to send the command to.
        :return: The ID of the command.
        """
        try:
            response = self.ssm_client.send_command(
                InstanceIds=instance_ids, DocumentName=self.name,
TimeoutSeconds=3600
            )
            return response["Command"]["CommandId"]
        except ClientError as err:
            logger.error(
                "Couldn't send command to %s. Here's why: %s: %s",
                self.name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

```
    def describe(self):
        """
        Describes the document.

        :return: Document status.
        """
        try:
            response = self.ssm_client.describe_document(Name=self.name)
            return response["Document"]["Status"]
        except ClientError as err:
            logger.error(
                "Couldn't get %s. Here's why: %s: %s",
                self.name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def wait_until_active(self, max_attempts=20, delay=5):
        """
        Waits until the document is active.

        :param max_attempts: The maximum number of attempts for checking the
status.
        :param delay: The delay in seconds between each check.
        """
        attempt = 0
        status = ""
        while attempt <= max_attempts:
            status = self.describe()
            if status == "Active":
                break
            attempt += 1
            time.sleep(delay)

        if status != "Active":
            logger.error("Document is not active.")
        else:
            logger.info("Document is active.")

    def wait_command_executed(self, command_id, instance_id):
```

```
        """
        Waits until the command is executed on the instance.

        :param command_id: The ID of the command.
        :param instance_id: The ID of the instance.
        """

        waiter = self.ssm_client.get_waiter("command_executed")
        waiter.wait(CommandId=command_id, InstanceId=instance_id)

    def list_command_invocations(self, instance_id):
        """
        Lists the commands for an instance.

        :param instance_id: The ID of the instance.
        :return: The list of commands.
        """
        try:
            paginator = self.ssm_client.get_paginator("list_command_invocations")
            command_invocations = []
            for page in paginator.paginate(InstanceId=instance_id):
                command_invocations.extend(page["CommandInvocations"])
            num_of_commands = len(command_invocations)
            print(
                f"{num_of_commands} command invocation(s) found for instance
{instance_id}."
            )

            if num_of_commands > 10:
                print("Displaying the first 10 commands:")
                num_of_commands = 10
            date_format = "%A, %d %B %Y %I:%M%p"
            for command in command_invocations[:num_of_commands]:
                print(
                    f"   The time of command invocation is
{command['RequestedDateTime'].strftime(date_format)}"
                )
        except ClientError as err:
            logger.error(
                "Couldn't list commands for %s. Here's why: %s: %s",
                instance_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
```

```
            raise
```

定义一个包装运营项目操作的类。

```
class OpsItemWrapper:
    """Encapsulates AWS Systems Manager OpsItem actions."""

    def __init__(self, ssm_client):
        """
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.id = None

    @classmethod
    def from_client(cls):
        """
        :return: A OpsItemWrapper instance.
        """
        ssm_client = boto3.client("ssm")
        return cls(ssm_client)


    def create(self, title, source, category, severity, description):
        """
        Create an OpsItem

        :param title: The OpsItem title.
        :param source: The OpsItem source.
        :param category: The OpsItem category.
        :param severity: The OpsItem severity.
        :param description: The OpsItem description.

        """
        try:
            response = self.ssm_client.create_ops_item(
                Title=title,
                Source=source,
                Category=category,
```

```
                Severity=severity,
                Description=description,
            )
            self.id = response["OpsItemId"]
        except self.ssm_client.exceptions.OpsItemLimitExceededException as err:
            logger.error(
                "Couldn't create ops item because you have exceeded your open
OpsItem limit. "
                "Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        except ClientError as err:
            logger.error(
                "Couldn't create ops item %s. Here's why: %s: %s",
                title,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise

    def delete(self):
        """
        Delete the OpsItem.
        """
        if self.id is None:
            return
        try:
            self.ssm_client.delete_ops_item(OpsItemId=self.id)
            print(f"Deleted ops item with id {self.id}")
            self.id = None
        except ClientError as err:
            logger.error(
                "Couldn't delete ops item %s. Here's why: %s: %s",
                self.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def describe(self):
        """
```

```
        Describe an OpsItem.
        """
        try:
            paginator = self.ssm_client.get_paginator("describe_ops_items")
            ops_items = []
            for page in paginator.paginate(
                OpsItemFilters=[
                    {"Key": "OpsItemId", "Values": [self.id], "Operator":
"Equal"}
                ]
            ):
                ops_items.extend(page["OpsItemSummaries"])

            for item in ops_items:
                print(
                    f"The item title is {item['Title']} and the status is
{item['Status']}"
                )
            return len(ops_items) > 0
        except ClientError as err:
            logger.error(
                "Couldn't describe ops item %s. Here's why: %s: %s",
                self.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def update(self, title=None, description=None, status=None):
        """
        Update an OpsItem.

        :param title: The new OpsItem title.
        :param description: The new OpsItem description.
        :param status: The new OpsItem status.
        :return:
        """
        args = dict(OpsItemId=self.id)
        if title is not None:
            args["Title"] = title
        if description is not None:
            args["Description"] = description
        if status is not None:
```

```
            args["Status"] = status
        try:
            self.ssm_client.update_ops_item(**args)
        except ClientError as err:
            logger.error(
                "Couldn't update ops item %s. Here's why: %s: %s",
                self.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

定义一个包装维护窗口操作的类。

```
class MaintenanceWindowWrapper:
    """Encapsulates AWS Systems Manager maintenance window actions."""

    def __init__(self, ssm_client):
        """
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.window_id = None
        self.name = None

    @classmethod
    def from_client(cls):
        ssm_client = boto3.client("ssm")
        return cls(ssm_client)


    def create(self, name, schedule, duration, cutoff,
 allow_unassociated_targets):
        """
        Create an AWS Systems Manager maintenance window.

        :param name: The name of the maintenance window.
        :param schedule: The schedule of the maintenance window.
        :param duration: The duration of the maintenance window.
```

```
        :param cutoff: The cutoff time of the maintenance window.
        :param allow_unassociated_targets: Allow the maintenance window to run on
managed nodes, even
                                           if you haven't registered those nodes
as targets.
        """
        try:
            response = self.ssm_client.create_maintenance_window(
                Name=name,
                Schedule=schedule,
                Duration=duration,
                Cutoff=cutoff,
                AllowUnassociatedTargets=allow_unassociated_targets,
            )
            self.window_id = response["WindowId"]
            self.name = name
            logger.info("Created maintenance window %s.", self.window_id)
        except ParamValidationError as error:
            logger.error(
                "Parameter validation error when trying to create maintenance
window %s. Here's why: %s",
                self.window_id,
                error,
            )
            raise
        except ClientError as err:
            logger.error(
                "Couldn't create maintenance window %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def delete(self):
        """
        Delete the associated AWS Systems Manager maintenance window.
        """
        if self.window_id is None:
            return

        try:
            self.ssm_client.delete_maintenance_window(WindowId=self.window_id)
```

```
            logger.info("Deleted maintenance window %s.", self.window_id)
            print(f"Deleted maintenance window {self.name}")
            self.window_id = None
        except ClientError as err:
            logger.error(
                "Couldn't delete maintenance window %s. Here's why: %s: %s",
                self.window_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def update(
        self, name, enabled, schedule, duration, cutoff,
allow_unassociated_targets
    ):
        """
        Update an AWS Systems Manager maintenance window.

        :param name: The name of the maintenance window.
        :param enabled: Whether the maintenance window is enabled to run on
managed nodes.
        :param schedule: The schedule of the maintenance window.
        :param duration: The duration of the maintenance window.
        :param cutoff: The cutoff time of the maintenance window.
        :param allow_unassociated_targets: Allow the maintenance window to run on
managed nodes, even
                                           if you haven't registered those nodes
as targets.
        """
        try:
            self.ssm_client.update_maintenance_window(
                WindowId=self.window_id,
                Name=name,
                Enabled=enabled,
                Schedule=schedule,
                Duration=duration,
                Cutoff=cutoff,
                AllowUnassociatedTargets=allow_unassociated_targets,
            )
            self.name = name
            logger.info("Updated maintenance window %s.", self.window_id)
        except ParamValidationError as error:
```

```
            logger.error(
                "Parameter validation error when trying to update maintenance
window %s. Here's why: %s",
                self.window_id,
                error,
            )
            raise
        except ClientError as err:
            logger.error(
                "Couldn't update maintenance window %s. Here's why: %s: %s",
                self.name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 有关 API 详细信息，请参阅《AWS SDK for Python (Boto3) API Reference》中的以下主题。
  - CreateDocument
  - CreateMaintenanceWindow
  - CreateOpsItem
  - DeleteMaintenanceWindow
  - ListCommandInvocations
  - SendCommand
  - UpdateOpsItem

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

使用 AWS SDK 对 Systems Manager 执行的操作

以下代码示例演示了如何使用 AWS SDK 来执行各个 Systems Manager 操作。每个示例都包含一个指向 GitHub 的链接，您可以在其中找到有关设置和运行代码的说明。

以下示例仅包括最常用的操作。有关完整列表，请参阅 AWS Systems Manager API 参考。

示例

- 将 AddTagsToResource 与 CLI 配合使用

- 将 CancelCommand 与 CLI 配合使用
- 将 CreateActivation 与 CLI 配合使用
- 将 CreateAssociation 与 CLI 配合使用
- 将 CreateAssociationBatch 与 CLI 配合使用
- 将 CreateDocument 与 AWS SDK 或 CLI 配合使用
- 将 CreateMaintenanceWindow 与 AWS SDK 或 CLI 配合使用
- 将 CreateOpsItem 与 AWS SDK 或 CLI 配合使用
- 将 CreatePatchBaseline 与 CLI 配合使用
- 将 DeleteActivation 与 CLI 配合使用
- 将 DeleteAssociation 与 CLI 配合使用
- 将 DeleteDocument 与 AWS SDK 或 CLI 配合使用
- 将 DeleteMaintenanceWindow 与 AWS SDK 或 CLI 配合使用
- 将 DeleteOpsItem 和 AWS SDK 搭配使用
- 将 DeleteParameter 与 CLI 配合使用
- 将 DeletePatchBaseline 与 CLI 配合使用
- 将 DeregisterManagedInstance 与 CLI 配合使用
- 将 DeregisterPatchBaselineForPatchGroup 与 CLI 配合使用
- 将 DeregisterTargetFromMaintenanceWindow 与 CLI 配合使用
- 将 DeregisterTaskFromMaintenanceWindow 与 CLI 配合使用
- 将 DescribeActivations 与 CLI 配合使用
- 将 DescribeAssociation 与 CLI 配合使用
- 将 DescribeAssociationExecutionTargets 与 CLI 配合使用
- 将 DescribeAssociationExecutions 与 CLI 配合使用
- 将 DescribeAutomationExecutions 与 CLI 配合使用
- 将 DescribeAutomationStepExecutions 与 CLI 配合使用
- 将 DescribeAvailablePatches 与 CLI 配合使用
- 将 DescribeDocument 与 CLI 配合使用
- 将 DescribeDocumentPermission 与 CLI 配合使用
- 将 DescribeEffectiveInstanceAssociations 与 CLI 配合使用
- 将 DescribeEffectivePatchesForPatchBaseline 与 CLI 配合使用

- 将 DescribeInstanceAssociationsStatus 与 CLI 配合使用
- 将 DescribeInstanceInformation 与 CLI 配合使用
- 将 DescribeInstancePatchStates 与 CLI 配合使用
- 将 DescribeInstancePatchStatesForPatchGroup 与 CLI 配合使用
- 将 DescribeInstancePatches 与 CLI 配合使用
- 将 DescribeMaintenanceWindowExecutionTaskInvocations 与 CLI 配合使用
- 将 DescribeMaintenanceWindowExecutionTasks 与 CLI 配合使用
- 将 DescribeMaintenanceWindowExecutions 与 CLI 配合使用
- 将 DescribeMaintenanceWindowTargets 与 CLI 配合使用
- 将 DescribeMaintenanceWindowTasks 与 CLI 配合使用
- 将 DescribeMaintenanceWindows 与 CLI 配合使用
- 将 DescribeOpsItems 与 AWS SDK 或 CLI 配合使用
- 将 DescribeParameters 与 AWS SDK 或 CLI 配合使用
- 将 DescribePatchBaselines 与 CLI 配合使用
- 将 DescribePatchGroupState 与 CLI 配合使用
- 将 DescribePatchGroups 与 CLI 配合使用
- 将 GetAutomationExecution 与 CLI 配合使用
- 将 GetCommandInvocation 与 CLI 配合使用
- 将 GetConnectionStatus 与 CLI 配合使用
- 将 GetDefaultPatchBaseline 与 CLI 配合使用
- 将 GetDeployablePatchSnapshotForInstance 与 CLI 配合使用
- 将 GetDocument 与 CLI 配合使用
- 将 GetInventory 与 CLI 配合使用
- 将 GetInventorySchema 与 CLI 配合使用
- 将 GetMaintenanceWindow 与 CLI 配合使用
- 将 GetMaintenanceWindowExecution 与 CLI 配合使用
- 将 GetMaintenanceWindowExecutionTask 与 CLI 配合使用
- 将 GetParameter 与 AWS SDK 或 CLI 配合使用
- 将 GetParameterHistory 与 CLI 配合使用
- 将 GetParameters 与 CLI 配合使用

- 将 GetPatchBaseline 与 CLI 配合使用
- 将 GetPatchBaselineForPatchGroup 与 CLI 配合使用
- 将 ListAssociationVersions 与 CLI 配合使用
- 将 ListAssociations 与 CLI 配合使用
- 将 ListCommandInvocations 与 AWS SDK 或 CLI 配合使用
- 将 ListCommands 与 CLI 配合使用
- 将 ListComplianceItems 与 CLI 配合使用
- 将 ListComplianceSummaries 与 CLI 配合使用
- 将 ListDocumentVersions 与 CLI 配合使用
- 将 ListDocuments 与 CLI 配合使用
- 将 ListInventoryEntries 与 CLI 配合使用
- 将 ListResourceComplianceSummaries 与 CLI 配合使用
- 将 ListTagsForResource 与 CLI 配合使用
- 将 ModifyDocumentPermission 与 CLI 配合使用
- 将 PutComplianceItems 与 CLI 配合使用
- 将 PutInventory 与 CLI 配合使用
- 将 PutParameter 与 AWS SDK 或 CLI 配合使用
- 将 RegisterDefaultPatchBaseline 与 CLI 配合使用
- 将 RegisterPatchBaselineForPatchGroup 与 CLI 配合使用
- 将 RegisterTargetWithMaintenanceWindow 与 CLI 配合使用
- 将 RegisterTaskWithMaintenanceWindow 与 CLI 配合使用
- 将 RemoveTagsFromResource 与 CLI 配合使用
- 将 SendCommand 与 AWS SDK 或 CLI 配合使用
- 将 StartAutomationExecution 与 CLI 配合使用
- 将 StartSession 与 CLI 配合使用
- 将 StopAutomationExecution 与 CLI 配合使用
- 将 UpdateAssociation 与 CLI 配合使用
- 将 UpdateAssociationStatus 与 CLI 配合使用
- 将 UpdateDocument 与 CLI 配合使用
- 将 UpdateDocumentDefaultVersion 与 CLI 配合使用

- 将 UpdateMaintenanceWindow 与 AWS SDK 或 CLI 配合使用
- 将 UpdateManagedInstanceRole 与 CLI 配合使用
- 将 UpdateOpsItem 与 AWS SDK 或 CLI 配合使用
- 将 UpdatePatchBaseline 与 CLI 配合使用

将 **AddTagsToResource** 与 CLI 配合使用

以下代码示例演示如何使用 AddTagsToResource。

CLI

AWS CLI

示例 1：向维护时段添加标签

以下 add-tags-to-resource 示例向指定的维护时段添加标签。

```
aws ssm add-tags-to-resource \
    --resource-type "MaintenanceWindow" \
    --resource-id "mw-03eb9db428EXAMPLE" \
    --tags "Key=Stack,Value=Production"
```

此命令不生成任何输出。

示例 2：向参数添加标签

以下 add-tags-to-resource 示例向指定参数添加两个标签。

```
aws ssm add-tags-to-resource \
    --resource-type "Parameter" \
    --resource-id "My-Parameter" \
    --tags '[{"Key":"Region","Value":"East"},{"Key":"Environment",
 "Value":"Production"}]'
```

此命令不生成任何输出。

示例 3：向 SSM 文档添加标签

以下 add-tags-to-resource 示例向指定文档添加标签。

```
aws ssm add-tags-to-resource \
```

```
    --resource-type "Document" \
    --resource-id "My-Document" \
    --tags "Key=Quarter,Value=Q322"
```

此命令不生成任何输出。

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的标记 Systems Manager 资源。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 AddTagsToResource。

PowerShell

Tools for PowerShell V4

示例 1：此示例使用新标签更新维护时段。如果此命令成功，则无任何输出。此示例使用的语法需要 PowerShell 版本 3 或更高版本。

```
$option1 = @{Key="Stack";Value=@("Production")}
Add-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
 "MaintenanceWindow" -Tag $option1
```

示例 2：对于 PowerShell 版本 2，必须使用 New-Object 创建每个标签。如果此命令成功，则无任何输出。

```
$tag1 = New-Object Amazon.SimpleSystemsManagement.Model.Tag
$tag1.Key = "Stack"
$tag1.Value = "Production"

Add-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
 "MaintenanceWindow" -Tag $tag1
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 AddTagsToResource。

Tools for PowerShell V5

示例 1：此示例使用新标签更新维护时段。如果此命令成功，则无任何输出。此示例使用的语法需要 PowerShell 版本 3 或更高版本。

```
$option1 = @{Key="Stack";Value=@("Production")}
```

```
Add-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
 "MaintenanceWindow" -Tag $option1
```

示例 2：对于 PowerShell 版本 2，必须使用 New-Object 创建每个标签。如果此命令成功，则无任何输出。

```
$tag1 = New-Object Amazon.SimpleSystemsManagement.Model.Tag
$tag1.Key = "Stack"
$tag1.Value = "Production"

Add-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
 "MaintenanceWindow" -Tag $tag1
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 AddTagsToResource。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **CancelCommand** 与 CLI 配合使用

以下代码示例演示如何使用 CancelCommand。

CLI

AWS CLI

示例 1：取消所有实例的命令

以下 cancel-command 示例尝试取消已对所有实例运行的指定命令。

```
aws ssm cancel-command \
    --command-id "662add3d-5831-4a10-b64a-f2ff3EXAMPLE"
```

此命令不生成任何输出。

示例 2：取消特定实例的命令

以下 cancel-command 示例仅尝试取消指定实例的命令。

```
aws ssm cancel-command \
```

```
    --command-id "662add3d-5831-4a10-b64a-f2ff3EXAMPLE"
    --instance-ids "i-02573cafcfEXAMPLE"
```

此命令不生成任何输出。

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的标记 Systems Manager 参数。

- 有关 API 的详细信息，请参阅《AWS CLI Command Reference》中的 CancelCommand。

PowerShell

Tools for PowerShell V4

示例 1：此示例尝试取消命令。如果操作成功，则无任何输出。

```
Stop-SSMCommand -CommandId "9ded293e-e792-4440-8e3e-7b8ec5feaa38"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 CancelCommand。

Tools for PowerShell V5

示例 1：此示例尝试取消命令。如果操作成功，则无任何输出。

```
Stop-SSMCommand -CommandId "9ded293e-e792-4440-8e3e-7b8ec5feaa38"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 CancelCommand。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **CreateActivation** 与 CLI 配合使用

以下代码示例演示如何使用 `CreateActivation`。

CLI

AWS CLI

创建托管式实例激活

以下 create-activation 示例创建托管式实例激活。

```
aws ssm create-activation \
    --default-instance-name "HybridWebServers" \
    --iam-role "HybridWebServersRole" \
    --registration-limit 5
```

输出：

```
{
    "ActivationId": "5743558d-563b-4457-8682-d16c3EXAMPLE",
    "ActivationCode": "dRmgnYaFv567vEXAMPLE"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的步骤 4：为混合环境创建托管式实例激活。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 CreateActivation。

PowerShell

Tools for PowerShell V4

示例 1：此示例创建托管式实例。

```
New-SSMActivation -DefaultInstanceName "MyWebServers" -IamRole
 "SSMAutomationRole" -RegistrationLimit 10
```

输出：

```
ActivationCode       ActivationId
--------------       ------------
KWChhOxBTiwDcKE9BlKC 08e51e79-1e36-446c-8e63-9458569c1363
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 CreateActivation。

Tools for PowerShell V5

示例 1：此示例创建托管式实例。

```
New-SSMActivation -DefaultInstanceName "MyWebServers" -IamRole
 "SSMAutomationRole" -RegistrationLimit 10
```

输出：

```
ActivationCode       ActivationId
--------------       ------------
KWChhOxBTiwDcKE9BlKC 08e51e79-1e36-446c-8e63-9458569c1363
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 CreateActivation。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **CreateAssociation** 与 CLI 配合使用

以下代码示例演示如何使用 CreateAssociation。

CLI

AWS CLI

示例 1：使用实例 ID 关联文档

此示例使用实例 ID 将配置文档与实例关联起来。

```
aws ssm create-association \
    --instance-id "i-0cb2b964d3e14fd9f" \
    --name "AWS-UpdateSSMAgent"
```

输出：

```
{
    "AssociationDescription": {
        "Status": {
            "Date": 1487875500.33,
            "Message": "Associated with AWS-UpdateSSMAgent",
            "Name": "Associated"
        },
        "Name": "AWS-UpdateSSMAgent",
```

```
        "InstanceId": "i-0cb2b964d3e14fd9f",
        "Overview": {
            "Status": "Pending",
            "DetailedStatus": "Creating"
        },
        "AssociationId": "b7c3266e-a544-44db-877e-b20d3a108189",
        "DocumentVersion": "$DEFAULT",
        "LastUpdateAssociationDate": 1487875500.33,
        "Date": 1487875500.33,
        "Targets": [
            {
                "Values": [
                    "i-0cb2b964d3e14fd9f"
                ],
                "Key": "InstanceIds"
            }
        ]
    }
}
```

有关更多信息，请参阅《AWS Systems Manager API Reference》中的 CreateAssociation。

示例 2：使用目标关联文档

此示例使用目标将配置文档与实例关联起来。

```
aws ssm create-association \
    --name "AWS-UpdateSSMAgent" \
    --targets "Key=instanceids,Values=i-0cb2b964d3e14fd9f"
```

输出：

```
{
    "AssociationDescription": {
        "Status": {
            "Date": 1487875500.33,
            "Message": "Associated with AWS-UpdateSSMAgent",
            "Name": "Associated"
        },
        "Name": "AWS-UpdateSSMAgent",
        "InstanceId": "i-0cb2b964d3e14fd9f",
        "Overview": {
            "Status": "Pending",
```

```
            "DetailedStatus": "Creating"
        },
        "AssociationId": "b7c3266e-a544-44db-877e-b20d3a108189",
        "DocumentVersion": "$DEFAULT",
        "LastUpdateAssociationDate": 1487875500.33,
        "Date": 1487875500.33,
        "Targets": [
            {
                "Values": [
                    "i-0cb2b964d3e14fd9f"
                ],
                "Key": "InstanceIds"
            }
        ]
    }
}
```

有关更多信息，请参阅《AWS Systems Manager API Reference》中的 CreateAssociation。

示例 3：创建仅运行一次的关联

此示例创建一个仅在指定日期和时间运行一次的新关联。使用过去或现在的日期创建的关联（处理关联时该日期已过去）会立即运行。

```
aws ssm create-association \
    --name "AWS-UpdateSSMAgent" \
    --targets "Key=instanceids,Values=i-0cb2b964d3e14fd9f" \
    --schedule-expression "at(2020-05-14T15:55:00)"  \
    --apply-only-at-cron-interval
```

输出：

```
{
    "AssociationDescription": {
        "Status": {
            "Date": 1487875500.33,
            "Message": "Associated with AWS-UpdateSSMAgent",
            "Name": "Associated"
        },
        "Name": "AWS-UpdateSSMAgent",
        "InstanceId": "i-0cb2b964d3e14fd9f",
        "Overview": {
            "Status": "Pending",
```

```
            "DetailedStatus": "Creating"
        },
        "AssociationId": "b7c3266e-a544-44db-877e-b20d3a108189",
        "DocumentVersion": "$DEFAULT",
        "LastUpdateAssociationDate": 1487875500.33,
        "Date": 1487875500.33,
        "Targets": [
            {
                "Values": [
                    "i-0cb2b964d3e14fd9f"
                ],
                "Key": "InstanceIds"
            }
        ]
    }
}
```

有关更多信息，请参阅《AWS Systems Manager API Reference》中的 CreateAssociation 或《AWS Systems Manager 用户指南》中的参考：适用于 Systems Manager 的 Cron 和 Rate 表达式。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 CreateAssociation。

PowerShell

Tools for PowerShell V4

示例 1：此示例使用实例 ID 将配置文档与实例关联起来。

```
New-SSMAssociation -InstanceId "i-0cb2b964d3e14fd9f" -Name "AWS-UpdateSSMAgent"
```

输出：

```
Name                  : AWS-UpdateSSMAgent
InstanceId            : i-0000293ffd8c57862
Date                  : 2/23/2017 6:55:22 PM
Status.Name           : Associated
Status.Date           : 2/20/2015 8:31:11 AM
Status.Message        : Associated with AWS-UpdateSSMAgent
Status.AdditionalInfo :
```

示例 2：此示例使用目标将配置文档与实例关联起来。

```
$target = @{Key="instanceids";Values=@("i-0cb2b964d3e14fd9f")}
New-SSMAssociation -Name "AWS-UpdateSSMAgent" -Target $target
```

输出：

```
Name                  : AWS-UpdateSSMAgent
InstanceId            :
Date                  : 3/1/2017 6:22:21 PM
Status.Name           :
Status.Date           :
Status.Message        :
Status.AdditionalInfo :
```

示例 3：此示例使用目标和参数将配置文档与实例关联起来。

```
$target = @{Key="instanceids";Values=@("i-0cb2b964d3e14fd9f")}
$params = @{
  "action"="configure"
  "mode"="ec2"
  "optionalConfigurationSource"="ssm"
  "optionalConfigurationLocation"=""
  "optionalRestart"="yes"
}
New-SSMAssociation -Name "Configure-CloudWatch" -AssociationName
 "CWConfiguration" -Target $target -Parameter $params
```

输出：

```
Name                  : Configure-CloudWatch
InstanceId            :
Date                  : 5/17/2018 3:17:44 PM
Status.Name           :
Status.Date           :
Status.Message        :
Status.AdditionalInfo :
```

示例 4：此示例使用 **AWS-GatherSoftwareInventory** 创建了与该区域中所有实例的关联。它还在要收集的参数中提供自定义文件和注册表位置

```
$params =
 [Collections.Generic.Dictionary[String,Collections.Generic.List[String]]]::new()
```

```
$params["windowsRegistry"] ='[{"Path":"HKEY_LOCAL_MACHINE\SOFTWARE\Amazon
\MachineImage","Recursive":false,"ValueNames":["AMIName"]}]'
$params["files"] = '[{"Path":"C:\Program Files","Pattern":
["*.exe"],"Recursive":true}, {"Path":"C:\ProgramData","Pattern":
["*.log"],"Recursive":true}]'
New-SSMAssociation -AssociationName new-in-mum -Name AWS-GatherSoftwareInventory
 -Target @{Key="instanceids";Values="*"} -Parameter $params -region ap-south-1 -
ScheduleExpression "rate(720 minutes)"
```

输出：

```
Name                  : AWS-GatherSoftwareInventory
InstanceId            :
Date                  : 6/9/2019 8:57:56 AM
Status.Name           :
Status.Date           :
Status.Message        :
Status.AdditionalInfo :
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 CreateAssociation。

Tools for PowerShell V5

示例 1：此示例使用实例 ID 将配置文档与实例关联起来。

```
New-SSMAssociation -InstanceId "i-0cb2b964d3e14fd9f" -Name "AWS-UpdateSSMAgent"
```

输出：

```
Name                  : AWS-UpdateSSMAgent
InstanceId            : i-0000293ffd8c57862
Date                  : 2/23/2017 6:55:22 PM
Status.Name           : Associated
Status.Date           : 2/20/2015 8:31:11 AM
Status.Message        : Associated with AWS-UpdateSSMAgent
Status.AdditionalInfo :
```

示例 2：此示例使用目标将配置文档与实例关联起来。

```
$target = @{Key="instanceids";Values=@("i-0cb2b964d3e14fd9f")}
```

```
New-SSMAssociation -Name "AWS-UpdateSSMAgent" -Target $target
```

输出：

```
Name                  : AWS-UpdateSSMAgent
InstanceId            :
Date                  : 3/1/2017 6:22:21 PM
Status.Name           :
Status.Date           :
Status.Message        :
Status.AdditionalInfo :
```

示例 3：此示例使用目标和参数将配置文档与实例关联起来。

```
$target = @{Key="instanceids";Values=@("i-0cb2b964d3e14fd9f")}
$params = @{
  "action"="configure"
  "mode"="ec2"
  "optionalConfigurationSource"="ssm"
  "optionalConfigurationLocation"=""
  "optionalRestart"="yes"
}
New-SSMAssociation -Name "Configure-CloudWatch" -AssociationName
 "CWConfiguration" -Target $target -Parameter $params
```

输出：

```
Name                  : Configure-CloudWatch
InstanceId            :
Date                  : 5/17/2018 3:17:44 PM
Status.Name           :
Status.Date           :
Status.Message        :
Status.AdditionalInfo :
```

示例 4：此示例使用 **AWS-GatherSoftwareInventory** 创建了与该区域中所有实例的关联。它还在要收集的参数中提供自定义文件和注册表位置

```
$params =
 [Collections.Generic.Dictionary[String,Collections.Generic.List[String]]]::new()
$params["windowsRegistry"] ='[{"Path":"HKEY_LOCAL_MACHINE\SOFTWARE\Amazon
\MachineImage","Recursive":false,"ValueNames":["AMIName"]}]'
```

```
$params["files"] = '[{"Path":"C:\Program Files","Pattern":
["*.exe"],"Recursive":true}, {"Path":"C:\ProgramData","Pattern":
["*.log"],"Recursive":true}]'
New-SSMAssociation -AssociationName new-in-mum -Name AWS-GatherSoftwareInventory
 -Target @{Key="instanceids";Values="*"} -Parameter $params -region ap-south-1 -
ScheduleExpression "rate(720 minutes)"
```

输出：

```
Name                  : AWS-GatherSoftwareInventory
InstanceId            :
Date                  : 6/9/2019 8:57:56 AM
Status.Name           :
Status.Date           :
Status.Message        :
Status.AdditionalInfo :
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 CreateAssociation。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **CreateAssociationBatch** 与 CLI 配合使用

以下代码示例演示如何使用 CreateAssociationBatch。

CLI

AWS CLI

创建多个关联

此示例将一个配置文档与多个实例相关联。如果适用，输出将返回成功和失败操作的列表。

命令:

```
aws ssm create-association-batch --entries "Name=AWS-
UpdateSSMAgent,InstanceId=i-1234567890abcdef0" "Name=AWS-
UpdateSSMAgent,InstanceId=i-9876543210abcdef0"
```

输出：


```
{
  "Successful": [
      {
          "Name": "AWS-UpdateSSMAgent",
          "InstanceId": "i-1234567890abcdef0",
          "AssociationVersion": "1",
          "Date": 1550504725.007,
          "LastUpdateAssociationDate": 1550504725.007,
          "Status": {
              "Date": 1550504725.007,
              "Name": "Associated",
              "Message": "Associated with AWS-UpdateSSMAgent"
          },
          "Overview": {
              "Status": "Pending",
              "DetailedStatus": "Creating"
          },
          "DocumentVersion": "$DEFAULT",
          "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
          "Targets": [
              {
                  "Key": "InstanceIds",
                  "Values": [
                      "i-1234567890abcdef0"
                  ]
              }
          ]
      },
      {
          "Name": "AWS-UpdateSSMAgent",
          "InstanceId": "i-9876543210abcdef0",
          "AssociationVersion": "1",
          "Date": 1550504725.057,
          "LastUpdateAssociationDate": 1550504725.057,
          "Status": {
              "Date": 1550504725.057,
              "Name": "Associated",
              "Message": "Associated with AWS-UpdateSSMAgent"
          },
          "Overview": {
              "Status": "Pending",
              "DetailedStatus": "Creating"
```

```
            },
            "DocumentVersion": "$DEFAULT",
            "AssociationId": "9c9f7f20-5154-4fed-a83e-0123456789ab",
            "Targets": [
                {
                    "Key": "InstanceIds",
                    "Values": [
                        "i-9876543210abcdef0"
                    ]
                }
            ]
        }
    ],
    "Failed": []
}
```

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 CreateAssociationBatch。

PowerShell

Tools for PowerShell V4

示例 1：此示例将一个配置文档与多个实例相关联。如果适用，输出将返回成功和失败操作的列表。

```
$option1 = @{InstanceId="i-0cb2b964d3e14fd9f";Name=@("AWS-UpdateSSMAgent")}
$option2 = @{InstanceId="i-0000293ffd8c57862";Name=@("AWS-UpdateSSMAgent")}
New-SSMAssociationFromBatch -Entry $option1,$option2
```

输出：

```
Failed  Successful
------  ----------
{}      {Amazon.SimpleSystemsManagement.Model.FailedCreateAssociation,
 Amazon.SimpleSystemsManagement.Model.FailedCreateAsso...
```

示例 2：此示例将显示成功操作的完整详细信息。

```
$option1 = @{InstanceId="i-0cb2b964d3e14fd9f";Name=@("AWS-UpdateSSMAgent")}
$option2 = @{InstanceId="i-0000293ffd8c57862";Name=@("AWS-UpdateSSMAgent")}
```

```
(New-SSMAssociationFromBatch -Entry $option1,$option2).Successful
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 CreateAssociationBatch。

Tools for PowerShell V5

示例 1：此示例将一个配置文档与多个实例相关联。如果适用，输出将返回成功和失败操作的列表。

```
$option1 = @{InstanceId="i-0cb2b964d3e14fd9f";Name=@("AWS-UpdateSSMAgent")}
$option2 = @{InstanceId="i-0000293ffd8c57862";Name=@("AWS-UpdateSSMAgent")}
New-SSMAssociationFromBatch -Entry $option1,$option2
```

输出：

```
Failed  Successful
------  ----------
{}      {Amazon.SimpleSystemsManagement.Model.FailedCreateAssociation,
 Amazon.SimpleSystemsManagement.Model.FailedCreateAsso...
```

示例 2：此示例将显示成功操作的完整详细信息。

```
$option1 = @{InstanceId="i-0cb2b964d3e14fd9f";Name=@("AWS-UpdateSSMAgent")}
$option2 = @{InstanceId="i-0000293ffd8c57862";Name=@("AWS-UpdateSSMAgent")}
(New-SSMAssociationFromBatch -Entry $option1,$option2).Successful
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 CreateAssociationBatch。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **CreateDocument** 与 AWS SDK 或 CLI 配合使用

以下代码示例演示如何使用 CreateDocument。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 了解基础知识

CLI

AWS CLI

创建文档

以下 `create-document` 示例创建一个 Systems Manager 文档。

```
aws ssm create-document \
    --content file://exampleDocument.yml \
    --name "Example" \
    --document-type "Automation" \
    --document-format YAML
```

输出：

```
{
    "DocumentDescription": {
        "Hash":
 "fc2410281f40779e694a8b95975d0f9f316da8a153daa94e3d9921102EXAMPLE",
        "HashType": "Sha256",
        "Name": "Example",
        "Owner": "29884EXAMPLE",
        "CreatedDate": 1583256349.452,
        "Status": "Creating",
        "DocumentVersion": "1",
        "Description": "Document Example",
        "Parameters": [
            {
                "Name": "AutomationAssumeRole",
                "Type": "String",
                "Description": "(Required) The ARN of the role that allows
 Automation to perform the actions on your behalf. If no role is specified,
 Systems Manager Automation uses your IAM permissions to execute this document.",
                "DefaultValue": ""
            },
            {
                "Name": "InstanceId",
                "Type": "String",
                "Description": "(Required) The ID of the Amazon EC2 instance.",
                "DefaultValue": ""
```

```
            }
        ],
        "PlatformTypes": [
            "Windows",
            "Linux"
        ],
        "DocumentType": "Automation",
        "SchemaVersion": "0.3",
        "LatestVersion": "1",
        "DefaultVersion": "1",
        "DocumentFormat": "YAML",
        "Tags": []
    }
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的创建 Systems Manager 文档。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 CreateDocument。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    /**
     * Creates an AWS SSM document asynchronously.
     *
     * @param docName The name of the document to create.
     * <p>
     * This method initiates an asynchronous request to create an SSM document.
     * If the request is successful, it prints the document status.
     * If an exception occurs, it handles the error appropriately.
     */
    public void createSSMDoc(String docName) throws SsmException {
        String jsonData = """
        {
```

```
        "schemaVersion": "2.2",
        "description": "Run a simple shell command",
        "mainSteps": [
            {
                "action": "aws:runShellScript",
                "name": "runEchoCommand",
                "inputs": {
                  "runCommand": [
                    "echo 'Hello, world!'"
                  ]
                }
              }
            ]
        }
        """;

        CreateDocumentRequest request = CreateDocumentRequest.builder()
                .content(jsonData)
                .name(docName)
                .documentType(DocumentType.COMMAND)
                .build();

        CompletableFuture<CreateDocumentResponse> future =
getAsyncClient().createDocument(request);
        future.thenAccept(response -> {
            System.out.println("The status of the SSM document is " +
response.documentDescription().status());
        }).exceptionally(ex -> {
            Throwable cause = (ex instanceof CompletionException) ?
ex.getCause() : ex;
            if (cause instanceof DocumentAlreadyExistsException) {
                throw new CompletionException(cause);
            } else if (cause instanceof SsmException) {
                throw new CompletionException(cause);
            } else {
                throw new RuntimeException(cause);
            }
        }).join();
    }
```

- 有关 API 详细信息，请参阅《AWS SDK for Java 2.x API Reference》中的 CreateDocument。

JavaScript

适用于 JavaScript 的 SDK（v3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import { CreateDocumentCommand, SSMClient } from "@aws-sdk/client-ssm";
import { parseArgs } from "node:util";

/**
 * Create an SSM document.
 * @param {{ content: string, name: string, documentType?: DocumentType }}
 */
export const main = async ({ content, name, documentType }) => {
  const client = new SSMClient({});
  try {
    const { documentDescription } = await client.send(
      new CreateDocumentCommand({
        Content: content, // The content for the new SSM document. The content
 must not exceed 64KB.
        Name: name,
        DocumentType: documentType, // Document format type can be JSON, YAML, or
 TEXT. The default format is JSON.
      }),
    );
    console.log("Document created successfully.");
    return { DocumentDescription: documentDescription };
  } catch (caught) {
    if (caught instanceof Error && caught.name === "DocumentAlreadyExists") {
      console.warn(`${caught.message}. Did you provide a new document name?`);
    } else {
      throw caught;
    }
  }
};
```

- 有关 API 详细信息，请参阅《适用于 JavaScript 的 AWS SDK API Reference》中的 CreateDocument。

PowerShell

Tools for PowerShell V4

示例 1：此示例在您的账户中创建一个文档。该文档必须采用 JSON 格式。有关编写配置文档的更多信息，请参阅《SSM API 参考》中的配置文档。

```
New-SSMDocument -Content (Get-Content -Raw "c:\temp\RunShellScript.json") -Name
 "RunShellScript" -DocumentType "Command"
```

输出：

```
CreatedDate     : 3/1/2017 1:21:33 AM
DefaultVersion  : 1
Description     : Run an updated script
DocumentType    : Command
DocumentVersion : 1
Hash            :
 1d5ce820e999ff051eb4841ed887593daf77120fd76cae0d18a53cc42e4e22c1
HashType        : Sha256
LatestVersion   : 1
Name            : RunShellScript
Owner           : 809632081692
Parameters      : {commands}
PlatformTypes   : {Linux}
SchemaVersion   : 2.0
Sha1            :
Status          : Creating
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 CreateDocument。

Tools for PowerShell V5

示例 1：此示例在您的账户中创建一个文档。该文档必须采用 JSON 格式。有关编写配置文档的更多信息，请参阅《SSM API 参考》中的配置文档。

```
New-SSMDocument -Content (Get-Content -Raw "c:\temp\RunShellScript.json") -Name
 "RunShellScript" -DocumentType "Command"
```

输出：

```
CreatedDate     : 3/1/2017 1:21:33 AM
DefaultVersion  : 1
Description     : Run an updated script
DocumentType    : Command
DocumentVersion : 1
Hash            :
 1d5ce820e999ff051eb4841ed887593daf77120fd76cae0d18a53cc42e4e22c1
HashType        : Sha256
LatestVersion   : 1
Name            : RunShellScript
Owner           : 809632081692
Parameters      : {commands}
PlatformTypes   : {Linux}
SchemaVersion   : 2.0
Sha1            :
Status          : Creating
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 CreateDocument。

Python

适用于 Python 的 SDK (Boto3)

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
class DocumentWrapper:
    """Encapsulates AWS Systems Manager Document actions."""

    def __init__(self, ssm_client):
        """
```

```
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.name = None

    @classmethod
    def from_client(cls):
        ssm_client = boto3.client("ssm")
        return cls(ssm_client)


    def create(self, content, name):
        """
        Creates a document.

        :param content: The content of the document.
        :param name: The name of the document.
        """
        try:
            self.ssm_client.create_document(
                Name=name, Content=content, DocumentType="Command"
            )
            self.name = name
        except self.ssm_client.exceptions.DocumentAlreadyExists:
            print(f"Document {name} already exists.")
            self.name = name
        except ClientError as err:
            logger.error(
                "Couldn't create %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 有关 API 的详细信息，请参阅《适用于 Python 的 AWS SDK (Boto3) API 参考》中的 CreateDocument。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **CreateMaintenanceWindow** 与 AWS SDK 或 CLI 配合使用

以下代码示例演示如何使用 CreateMaintenanceWindow。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 了解基础知识

CLI

AWS CLI

示例 1：创建维护时段

以下 create-maintenance-window 示例创建一个新的维护时段，每五分钟执行一次，最多持续两个小时（根据需要），防止新任务在维护时段执行结束后的一小时内启动，允许未关联的目标（您尚未向维护时段注册的实例），并通过使用自定义标签表明其创建者打算在教程中进行使用。

```
aws ssm create-maintenance-window \
    --name "My-Tutorial-Maintenance-Window" \
    --schedule "rate(5 minutes)" \
    --duration 2 --cutoff 1 \
    --allow-unassociated-targets \
    --tags "Key=Purpose,Value=Tutorial"
```

输出：

```
{
    "WindowId": "mw-0c50858d01EXAMPLE"
}
```

示例 2：创建仅运行一次的维护时段

以下 create-maintenance-window 示例创建了一个仅在指定日期和时间运行一次的新维护时段。

```
aws ssm create-maintenance-window \
    --name My-One-Time-Maintenance-Window \
    --schedule "at(2020-05-14T15:55:00)" \
```

```
    --duration 5 \
    --cutoff 2 \
    --allow-unassociated-targets \
    --tags "Key=Environment,Value=Production"
```

输出：

```
{
    "WindowId": "mw-01234567890abcdef"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的维护时段。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 CreateMaintenanceWindow。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    /**
     * Creates an SSM maintenance window asynchronously.
     *
     * @param winName The name of the maintenance window.
     * @return The ID of the created or existing maintenance window.
     * <p>
     * This method initiates an asynchronous request to create an SSM maintenance
window.
     * If the request is successful, it prints the maintenance window ID.
     * If an exception occurs, it handles the error appropriately.
     */
    public String createMaintenanceWindow(String winName) throws SsmException,
DocumentAlreadyExistsException {
        CreateMaintenanceWindowRequest request =
CreateMaintenanceWindowRequest.builder()
```

```
                .name(winName)
                .description("This is my maintenance window")
                .allowUnassociatedTargets(true)
                .duration(2)
                .cutoff(1)
                .schedule("cron(0 10 ? * MON-FRI *)")
                .build();

        CompletableFuture<CreateMaintenanceWindowResponse> future =
getAsyncClient().createMaintenanceWindow(request);
        final String[] windowId = {null};
        future.whenComplete((response, ex) -> {
            if (response != null) {
                String maintenanceWindowId = response.windowId();
                System.out.println("The maintenance window id is " +
maintenanceWindowId);
                windowId[0] = maintenanceWindowId;
            } else {
                Throwable cause = (ex instanceof CompletionException) ?
ex.getCause() : ex;
                if (cause instanceof DocumentAlreadyExistsException) {
                    throw new CompletionException(cause);
                } else if (cause instanceof SsmException) {
                    throw new CompletionException(cause);
                } else {
                    throw new RuntimeException(cause);
                }
            }
        }).join();

        if (windowId[0] == null) {
            MaintenanceWindowFilter filter = MaintenanceWindowFilter.builder()
                    .key("name")
                    .values(winName)
                    .build();

            DescribeMaintenanceWindowsRequest winRequest =
DescribeMaintenanceWindowsRequest.builder()
                    .filters(filter)
                    .build();

            CompletableFuture<DescribeMaintenanceWindowsResponse> describeFuture
= getAsyncClient().describeMaintenanceWindows(winRequest);
            describeFuture.whenComplete((describeResponse, describeEx) -> {
```

```
                if (describeResponse != null) {
                    List<MaintenanceWindowIdentity> windows =
describeResponse.windowIdentities();
                    if (!windows.isEmpty()) {
                        windowId[0] = windows.get(0).windowId();
                        System.out.println("Window ID: " + windowId[0]);
                    } else {
                        System.out.println("Window not found.");
                        windowId[0] = "";
                    }
                } else {
                    Throwable describeCause = (describeEx instanceof
CompletionException) ? describeEx.getCause() : describeEx;
                    throw new RuntimeException("Error describing maintenance
windows: " + describeCause.getMessage(), describeCause);
                }
            }).join();
        }

        return windowId[0];
    }
```

- 有关 API 详细信息，请参阅《AWS SDK for Java 2.x API Reference》中的 CreateMaintenanceWindow。

JavaScript

适用于 JavaScript 的 SDK（v3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import { CreateMaintenanceWindowCommand, SSMClient } from "@aws-sdk/client-ssm";
import { parseArgs } from "node:util";

/**
 * Create an SSM maintenance window.
```

```
 * @param {{ name: string, allowUnassociatedTargets: boolean, duration: number,
 cutoff: number, schedule: string, description?: string }}
 */
export const main = async ({
  name,
  allowUnassociatedTargets, // Allow the maintenance window to run on managed
 nodes, even if you haven't registered those nodes as targets.
  duration, // The duration of the maintenance window in hours.
  cutoff, // The number of hours before the end of the maintenance window that
 Amazon Web Services Systems Manager stops scheduling new tasks for execution.
  schedule, // The schedule of the maintenance window in the form of a cron or
 rate expression.
  description = undefined,
}) => {
  const client = new SSMClient({});

  try {
    const { windowId } = await client.send(
      new CreateMaintenanceWindowCommand({
        Name: name,
        Description: description,
        AllowUnassociatedTargets: allowUnassociatedTargets, // Allow the
 maintenance window to run on managed nodes, even if you haven't registered those
 nodes as targets.
        Duration: duration, // The duration of the maintenance window in hours.
        Cutoff: cutoff, // The number of hours before the end of the maintenance
 window that Amazon Web Services Systems Manager stops scheduling new tasks for
 execution.
        Schedule: schedule, // The schedule of the maintenance window in the form
 of a cron or rate expression.
      }),
    );
    console.log(`Maintenance window created with Id: ${windowId}`);
    return { WindowId: windowId };
  } catch (caught) {
    if (caught instanceof Error && caught.name === "MissingParameter") {
      console.warn(`${caught.message}. Did you provide these values?`);
    } else {
      throw caught;
    }
  }
};
```

- 有关 API 详细信息，请参阅《适用于 JavaScript 的 AWS SDK API Reference》中的 CreateMaintenanceWindow。

PowerShell

Tools for PowerShell V4

示例 1：此示例使用指定名称创建了一个新的维护时段，该时段在每周二下午 4 点运行 4 小时，中断 1 小时，且允许未关联的目标。

```
New-SSMMaintenanceWindow -Name "MyMaintenanceWindow" -Duration 4 -Cutoff 1 -
AllowUnassociatedTarget $true -Schedule "cron(0 16 ? * TUE *)"
```

输出：

```
mw-03eb53e1ea7383998
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 CreateMaintenanceWindow。

Tools for PowerShell V5

示例 1：此示例使用指定名称创建了一个新的维护时段，该时段在每周二下午 4 点运行 4 小时，中断 1 小时，且允许未关联的目标。

```
New-SSMMaintenanceWindow -Name "MyMaintenanceWindow" -Duration 4 -Cutoff 1 -
AllowUnassociatedTarget $true -Schedule "cron(0 16 ? * TUE *)"
```

输出：

```
mw-03eb53e1ea7383998
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 CreateMaintenanceWindow。

Python

适用于 Python 的 SDK (Boto3)

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
class MaintenanceWindowWrapper:
    """Encapsulates AWS Systems Manager maintenance window actions."""

    def __init__(self, ssm_client):
        """
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.window_id = None
        self.name = None

    @classmethod
    def from_client(cls):
        ssm_client = boto3.client("ssm")
        return cls(ssm_client)


    def create(self, name, schedule, duration, cutoff,
 allow_unassociated_targets):
        """
        Create an AWS Systems Manager maintenance window.

        :param name: The name of the maintenance window.
        :param schedule: The schedule of the maintenance window.
        :param duration: The duration of the maintenance window.
        :param cutoff: The cutoff time of the maintenance window.
        :param allow_unassociated_targets: Allow the maintenance window to run on
 managed nodes, even
                                           if you haven't registered those nodes
 as targets.
        """
        try:
```

```
            response = self.ssm_client.create_maintenance_window(
                Name=name,
                Schedule=schedule,
                Duration=duration,
                Cutoff=cutoff,
                AllowUnassociatedTargets=allow_unassociated_targets,
            )
            self.window_id = response["WindowId"]
            self.name = name
            logger.info("Created maintenance window %s.", self.window_id)
        except ParamValidationError as error:
            logger.error(
                "Parameter validation error when trying to create maintenance
window %s. Here's why: %s",
                self.window_id,
                error,
            )
            raise
        except ClientError as err:
            logger.error(
                "Couldn't create maintenance window %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 有关 API 详细信息，请参阅《AWS SDK for Python (Boto3) API 参考》中的 CreateMaintenanceWindow。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **CreateOpsItem** 与 AWS SDK 或 CLI 配合使用

以下代码示例演示如何使用 CreateOpsItem。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 了解基础知识

CLI

AWS CLI

创建 OpsItem

以下 `create-ops-item` 示例在 OperationalData 中使用 /aws/resources 键创建具有 Amazon DynamoDB 相关资源的 OpsItem。

```
aws ssm create-ops-item \
    --title "EC2 instance disk full" \
    --description "Log clean up may have failed which caused the disk to be full"
 \
    --priority 2 \
    --source ec2 \
    --operational-data '{"/aws/resources":{"Value":"[{\"arn
\": \"arn:aws:dynamodb:us-west-2:12345678:table/OpsItems
\"}]","Type":"SearchableString"}}' \
    --notifications Arn="arn:aws:sns:us-west-2:12345678:TestUser"
```

输出：

```
{
    "OpsItemId": "oi-1a2b3c4d5e6f"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的创建 OpsItem。

- 有关 API 详细信息，请参阅《AWS CLI 命令参考》中的 CreateOpsItem。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    /**
     * Creates an SSM OpsItem asynchronously.
     *
     * @param title The title of the OpsItem.
     * @param source The source of the OpsItem.
     * @param category The category of the OpsItem.
     * @param severity The severity of the OpsItem.
     * @return The ID of the created OpsItem.
     * <p>
     * This method initiates an asynchronous request to create an SSM OpsItem.
     * If the request is successful, it returns the OpsItem ID.
     * If an exception occurs, it handles the error appropriately.
     */
    public String createSSMOpsItem(String title, String source, String category,
String severity) {
        CreateOpsItemRequest opsItemRequest = CreateOpsItemRequest.builder()
                .description("Created by the SSM Java API")
                .title(title)
                .source(source)
                .category(category)
                .severity(severity)
                .build();

        CompletableFuture<CreateOpsItemResponse> future =
getAsyncClient().createOpsItem(opsItemRequest);

        try {
            CreateOpsItemResponse response = future.join();
            return response.opsItemId();
        } catch (CompletionException e) {
            Throwable cause = e.getCause();
            if (cause instanceof SsmException) {
                throw (SsmException) cause;
            } else {
                throw new RuntimeException(cause);
            }
        }
    }
```

- 有关 API 详细信息，请参阅《AWS SDK for Java 2.x API 参考》中的 CreateOpsItem。

JavaScript

适用于 JavaScript 的 SDK（v3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import { CreateOpsItemCommand, SSMClient } from "@aws-sdk/client-ssm";
import { parseArgs } from "node:util";

/**
 * Create an SSM OpsItem.
 * @param {{ title: string, source: string, category?: string, severity?:
 string }}
 */
export const main = async ({
  title,
  source,
  category = undefined,
  severity = undefined,
}) => {
  const client = new SSMClient({});
  try {
    const { opsItemArn, opsItemId } = await client.send(
      new CreateOpsItemCommand({
        Title: title,
        Source: source, // The origin of the OpsItem, such as Amazon EC2 or
 Systems Manager.
        Category: category,
        Severity: severity,
      }),
    );
    console.log(`Ops item created with id: ${opsItemId}`);
    return { OpsItemArn: opsItemArn, OpsItemId: opsItemId };
  } catch (caught) {
    if (caught instanceof Error && caught.name === "MissingParameter") {
      console.warn(`${caught.message}. Did you provide these values?`);
    } else {
      throw caught;
```

```
    }
  }
};
```

- 有关 API 详细信息，请参阅《适用于 JavaScript 的 AWS SDK API 参考》中的 CreateOpsItem。

Python

适用于 Python 的 SDK (Boto3)

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
class OpsItemWrapper:
    """Encapsulates AWS Systems Manager OpsItem actions."""

    def __init__(self, ssm_client):
        """
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.id = None

    @classmethod
    def from_client(cls):
        """
        :return: A OpsItemWrapper instance.
        """
        ssm_client = boto3.client("ssm")
        return cls(ssm_client)


    def create(self, title, source, category, severity, description):
        """
        Create an OpsItem
```

```
        :param title: The OpsItem title.
        :param source: The OpsItem source.
        :param category: The OpsItem category.
        :param severity: The OpsItem severity.
        :param description: The OpsItem description.

        """
        try:
            response = self.ssm_client.create_ops_item(
                Title=title,
                Source=source,
                Category=category,
                Severity=severity,
                Description=description,
            )
            self.id = response["OpsItemId"]
        except self.ssm_client.exceptions.OpsItemLimitExceededException as err:
            logger.error(
                "Couldn't create ops item because you have exceeded your open
OpsItem limit. "
                "Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        except ClientError as err:
            logger.error(
                "Couldn't create ops item %s. Here's why: %s: %s",
                title,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 有关 API 的详细信息，请参阅《AWS SDK for Python（Boto3）API 参考》中的 CreateOpsItem。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **CreatePatchBaseline** 与 CLI 配合使用

以下代码示例演示如何使用 CreatePatchBaseline。

CLI

AWS CLI

示例 1：创建具有自动批准功能的补丁基准

以下 create-patch-baseline 示例创建一个 Windows Server 的补丁基准，该基准在 Microsoft 发布补丁 7 天后批准生产环境的补丁。

```
aws ssm create-patch-baseline \
    --name "Windows-Production-Baseline-AutoApproval" \
    --operating-system "WINDOWS" \
    --approval-
rules "PatchRules=[{PatchFilterGroup={PatchFilters=[{Key=MSRC_SEVERITY,Values=[Critical,1
{Key=CLASSIFICATION,Values=[SecurityUpdates,Updates,UpdateRollups,CriticalUpdates]}]},App
 \
    --description "Baseline containing all updates approved for Windows Server
 production systems"
```

输出：

```
{
    "BaselineId": "pb-045f10b4f3EXAMPLE"
}
```

示例 2：创建带有批准截止日期的补丁基准

以下 create-patch-baseline 示例为 Windows Server 创建补丁基准，其批准 2020 年 7 月 7 日或之前在生产环境中发布的所有补丁。

```
aws ssm create-patch-baseline \
    --name "Windows-Production-Baseline-AutoApproval" \
    --operating-system "WINDOWS" \
    --approval-
rules "PatchRules=[{PatchFilterGroup={PatchFilters=[{Key=MSRC_SEVERITY,Values=[Critical,1
{Key=CLASSIFICATION,Values=[SecurityUpdates,Updates,UpdateRollups,CriticalUpdates]}]},App
 \
```

```
    --description "Baseline containing all updates approved for Windows Server
 production systems"
```

输出：

```
{
    "BaselineId": "pb-045f10b4f3EXAMPLE"
}
```

示例 3：创建批准规则存储在 JSON 文件中的补丁基准

以下 create-patch-baseline 示例为 Amazon Linux 2017.09 创建补丁基准，其将在补丁发布 7 天后批准生产环境的补丁，指定补丁基准的批准规则，并指定补丁的自定义存储库。

```
aws ssm create-patch-baseline \
    --cli-input-json file://my-amazon-linux-approval-rules-and-repo.json
```

my-amazon-linux-approval-rules-and-repo.json 的内容：

```
{
    "Name": "Amazon-Linux-2017.09-Production-Baseline",
    "Description": "My approval rules patch baseline for Amazon Linux 2017.09
 instances",
    "OperatingSystem": "AMAZON_LINUX",
    "Tags": [
        {
            "Key": "Environment",
            "Value": "Production"
        }
    ],
    "ApprovalRules": {
        "PatchRules": [
            {
                "ApproveAfterDays": 7,
                "EnableNonSecurity": true,
                "PatchFilterGroup": {
                    "PatchFilters": [
                        {
                            "Key": "SEVERITY",
                            "Values": [
                                "Important",
                                "Critical"
```

```
                            ]
                        },
                        {
                            "Key": "CLASSIFICATION",
                            "Values": [
                                "Security",
                                "Bugfix"
                            ]
                        },
                        {
                            "Key": "PRODUCT",
                            "Values": [
                                "AmazonLinux2017.09"
                            ]
                        }
                    ]
                }
            }
        ]
    },
    "Sources": [
        {
            "Name": "My-AL2017.09",
            "Products": [
                "AmazonLinux2017.09"
            ],
            "Configuration": "[amzn-main] \nname=amzn-main-Base
\nmirrorlist=http://repo./$awsregion./$awsdomain//$releasever/main/
mirror.list //nmirrorlist_expire=300//nmetadata_expire=300 \npriority=10
 \nfailovermethod=priority \nfastestmirror_enabled=0 \ngpgcheck=1
 \ngpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-amazon-ga \nenabled=1 \nretries=3
 \ntimeout=5\nreport_instanceid=yes"
        }
    ]
}
```

示例 4：创建指定已批准和已拒绝补丁的补丁基准

以下 create-patch-baseline 示例明确指定要批准和拒绝的补丁，作为默认批准规则的例外情况。

```
aws ssm create-patch-baseline \
    --name "Amazon-Linux-2017.09-Alpha-Baseline" \
```

```
    --description "My custom approve/reject patch baseline for Amazon Linux
 2017.09 instances" \
    --operating-system "AMAZON_LINUX" \
    --approved-patches "CVE-2018-1234567,example-pkg-EE-2018*.amzn1.noarch" \
    --approved-patches-compliance-level "HIGH" \
    --approved-patches-enable-non-security \
    --tags "Key=Environment,Value=Alpha"
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的创建自定义补丁基准。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 CreatePatchBaseline。

PowerShell

Tools for PowerShell V4

示例 1：此示例创建补丁基准，用于在 Microsoft 发布补丁 7 天后为在生产环境中运行 Windows Server 2019 的托管式实例批准补丁。

```
$rule = New-Object Amazon.SimpleSystemsManagement.Model.PatchRule
$rule.ApproveAfterDays = 7

$ruleFilters = New-Object Amazon.SimpleSystemsManagement.Model.PatchFilterGroup

$patchFilter = New-Object Amazon.SimpleSystemsManagement.Model.PatchFilter
$patchFilter.Key="PRODUCT"
$patchFilter.Values="WindowsServer2019"

$severityFilter = New-Object Amazon.SimpleSystemsManagement.Model.PatchFilter
$severityFilter.Key="MSRC_SEVERITY"
$severityFilter.Values.Add("Critical")
$severityFilter.Values.Add("Important")
$severityFilter.Values.Add("Moderate")

$classificationFilter = New-Object
 Amazon.SimpleSystemsManagement.Model.PatchFilter
$classificationFilter.Key = "CLASSIFICATION"
$classificationFilter.Values.Add( "SecurityUpdates" )
$classificationFilter.Values.Add( "Updates" )
$classificationFilter.Values.Add( "UpdateRollups" )
$classificationFilter.Values.Add( "CriticalUpdates" )

$ruleFilters.PatchFilters.Add($severityFilter)
```

```
$ruleFilters.PatchFilters.Add($classificationFilter)
$ruleFilters.PatchFilters.Add($patchFilter)
$rule.PatchFilterGroup = $ruleFilters

New-SSMPatchBaseline -Name "Production-Baseline-Windows2019" -Description
 "Baseline containing all updates approved for production systems" -
ApprovalRules_PatchRule $rule
```

输出：

```
pb-0z4z6221c4296b23z
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 CreatePatchBaseline。

Tools for PowerShell V5

示例 1：此示例创建补丁基准，用于在 Microsoft 发布补丁 7 天后为在生产环境中运行 Windows Server 2019 的托管式实例批准补丁。

```
$rule = New-Object Amazon.SimpleSystemsManagement.Model.PatchRule
$rule.ApproveAfterDays = 7

$ruleFilters = New-Object Amazon.SimpleSystemsManagement.Model.PatchFilterGroup

$patchFilter = New-Object Amazon.SimpleSystemsManagement.Model.PatchFilter
$patchFilter.Key="PRODUCT"
$patchFilter.Values="WindowsServer2019"

$severityFilter = New-Object Amazon.SimpleSystemsManagement.Model.PatchFilter
$severityFilter.Key="MSRC_SEVERITY"
$severityFilter.Values.Add("Critical")
$severityFilter.Values.Add("Important")
$severityFilter.Values.Add("Moderate")

$classificationFilter = New-Object
 Amazon.SimpleSystemsManagement.Model.PatchFilter
$classificationFilter.Key = "CLASSIFICATION"
$classificationFilter.Values.Add( "SecurityUpdates" )
$classificationFilter.Values.Add( "Updates" )
$classificationFilter.Values.Add( "UpdateRollups" )
$classificationFilter.Values.Add( "CriticalUpdates" )
```

```
$ruleFilters.PatchFilters.Add($severityFilter)
$ruleFilters.PatchFilters.Add($classificationFilter)
$ruleFilters.PatchFilters.Add($patchFilter)
$rule.PatchFilterGroup = $ruleFilters

New-SSMPatchBaseline -Name "Production-Baseline-Windows2019" -Description
 "Baseline containing all updates approved for production systems" -
ApprovalRules_PatchRule $rule
```

输出：

```
pb-0z4z6221c4296b23z
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 CreatePatchBaseline。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DeleteActivation** 与 CLI 配合使用

以下代码示例演示如何使用 DeleteActivation。

CLI

AWS CLI

删除托管式实例激活

以下 delete-activation 示例删除托管式实例激活。

```
aws ssm delete-activation \
    --activation-id "aa673477-d926-42c1-8757-1358cEXAMPLE"
```

此命令不生成任何输出。

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的为混合环境设置 AWS Systems Manager。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DeleteActivation。

PowerShell

Tools for PowerShell V4

示例 1：此示例删除激活。如果此命令成功，则无任何输出。

```
Remove-SSMActivation -ActivationId "08e51e79-1e36-446c-8e63-9458569c1363"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DeleteActivation。

Tools for PowerShell V5

示例 1：此示例删除激活。如果此命令成功，则无任何输出。

```
Remove-SSMActivation -ActivationId "08e51e79-1e36-446c-8e63-9458569c1363"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DeleteActivation。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DeleteAssociation** 与 CLI 配合使用

以下代码示例演示如何使用 DeleteAssociation。

CLI

AWS CLI

示例 1：使用关联 ID 删除关联

以下 delete-association 示例删除指定关联 ID 的关联。如果此命令成功，则无任何输出。

```
aws ssm delete-association \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

此命令不生成任何输出。

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的编辑和创建关联的新版本。

示例 2：删除关联

以下 delete-association 示例删除实例和文档之间的关联。如果此命令成功，则无任何输出。

```
aws ssm delete-association \
    --instance-id "i-1234567890abcdef0" \
    --name "AWS-UpdateSSMAgent"
```

此命令不生成任何输出。

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的在 Systems Manager 中使用关联。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DeleteAssociation。

PowerShell

Tools for PowerShell V4

示例 1：此示例删除实例和文档之间的关联。如果此命令成功，则无任何输出。

```
Remove-SSMAssociation -InstanceId "i-0cb2b964d3e14fd9f" -Name "AWS-
UpdateSSMAgent"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DeleteAssociation。

Tools for PowerShell V5

示例 1：此示例删除实例和文档之间的关联。如果此命令成功，则无任何输出。

```
Remove-SSMAssociation -InstanceId "i-0cb2b964d3e14fd9f" -Name "AWS-
UpdateSSMAgent"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DeleteAssociation。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DeleteDocument** 与 AWS SDK 或 CLI 配合使用

以下代码示例演示如何使用 DeleteDocument。

CLI

AWS CLI

删除文档

以下 `delete-document` 示例删除一个 Systems Manager 文档。

```
aws ssm delete-document \
    --name "Example"
```

此命令不生成任何输出。

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的创建 Systems Manager 文档。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DeleteDocument。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    /**
     * Deletes an AWS SSM document asynchronously.
     *
     * @param documentName The name of the document to delete.
     * <p>
     * This method initiates an asynchronous request to delete an SSM document.
     * If an exception occurs, it handles the error appropriately.
     */
    public void deleteDoc(String documentName) {
        DeleteDocumentRequest documentRequest = DeleteDocumentRequest.builder()
                .name(documentName)
                .build();
```

```
        CompletableFuture<Void> future = CompletableFuture.runAsync(() -> {
            getAsyncClient().deleteDocument(documentRequest)
                    .thenAccept(response -> {
                        System.out.println("The SSM document was successfully
deleted.");
                    })
                    .exceptionally(ex -> {
                        throw new CompletionException(ex);
                    }).join();
        }).exceptionally(ex -> {
            Throwable cause = (ex instanceof CompletionException) ?
ex.getCause() : ex;
            if (cause instanceof SsmException) {
                throw new RuntimeException("SSM error: " + cause.getMessage(),
cause);
            } else {
                throw new RuntimeException("Unexpected error: " +
cause.getMessage(), cause);
            }
        });

        try {
            future.join();
        } catch (CompletionException ex) {
            throw ex.getCause() instanceof RuntimeException ? (RuntimeException)
ex.getCause() : ex;
        }
    }
```

- 有关 API 详细信息，请参阅《AWS SDK for Java 2.x API Reference》中的 DeleteDocument。

JavaScript

适用于 JavaScript 的 SDK（v3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import { DeleteDocumentCommand, SSMClient } from "@aws-sdk/client-ssm";
import { parseArgs } from "node:util";

/**
 * Delete an SSM document.
 * @param {{ documentName: string }}
 */
export const main = async ({ documentName }) => {
  const client = new SSMClient({});
  try {
    await client.send(new DeleteDocumentCommand({ Name: documentName }));
    console.log(`Document '${documentName}' deleted.`);
    return { Deleted: true };
  } catch (caught) {
    if (caught instanceof Error && caught.name === "MissingParameter") {
      console.warn(`${caught.message}. Did you provide this value?`);
    } else {
      throw caught;
    }
  }
};
```

- 有关 API 详细信息，请参阅《适用于 JavaScript 的 AWS SDK API Reference》中的 DeleteDocument。

PowerShell

Tools for PowerShell V4

示例 1：此示例删除文档。如果此命令成功，则无任何输出。

```
Remove-SSMDocument -Name "RunShellScript"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DeleteDocument。

Tools for PowerShell V5

示例 1：此示例删除文档。如果此命令成功，则无任何输出。

```
Remove-SSMDocument -Name "RunShellScript"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DeleteDocument。

Python

适用于 Python 的 SDK (Boto3)

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
class DocumentWrapper:
    """Encapsulates AWS Systems Manager Document actions."""

    def __init__(self, ssm_client):
        """
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.name = None

    @classmethod
    def from_client(cls):
        ssm_client = boto3.client("ssm")
        return cls(ssm_client)


    def delete(self):
        """
        Deletes an AWS Systems Manager document.
        """
        if self.name is None:
            return

        try:
            self.ssm_client.delete_document(Name=self.name)
            print(f"Deleted document {self.name}.")
            self.name = None
        except ClientError as err:
```

```
            logger.error(
                "Couldn't delete %s. Here's why: %s: %s",
                self.name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 有关 API 的详细信息，请参阅《适用于 Python 的 AWS SDK (Boto3) API 参考》中的 DeleteDocument。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DeleteMaintenanceWindow** 与 AWS SDK 或 CLI 配合使用

以下代码示例演示如何使用 DeleteMaintenanceWindow。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 了解基础知识

CLI

AWS CLI

删除维护时段

此 delete-maintenance-window 示例删除指定的维护时段。

```
aws ssm delete-maintenance-window \
    --window-id "mw-1a2b3c4d5e6f7g8h9"
```

输出：

```
{
    "WindowId":"mw-1a2b3c4d5e6f7g8h9"
```

```
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的删除维护时段（AWS CLI）。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DeleteMaintenanceWindow。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    /**
     * Deletes an AWS SSM Maintenance Window asynchronously.
     *
     * @param winId The ID of the Maintenance Window to delete.
     * <p>
     * This method initiates an asynchronous request to delete an SSM Maintenance
Window.
     * If an exception occurs, it handles the error appropriately.
     */
    public void deleteMaintenanceWindow(String winId) {
        DeleteMaintenanceWindowRequest windowRequest =
DeleteMaintenanceWindowRequest.builder()
                .windowId(winId)
                .build();

        CompletableFuture<Void> future = CompletableFuture.runAsync(() -> {
            getAsyncClient().deleteMaintenanceWindow(windowRequest)
                    .thenAccept(response -> {
                        System.out.println("The maintenance window was
successfully deleted.");
                    })
                    .exceptionally(ex -> {
                        throw new CompletionException(ex);
                    }).join();
        }).exceptionally(ex -> {
```

```
            Throwable cause = (ex instanceof CompletionException) ?
ex.getCause() : ex;
            if (cause instanceof SsmException) {
                throw new RuntimeException("SSM error: " + cause.getMessage(),
cause);
            } else {
                throw new RuntimeException("Unexpected error: " +
cause.getMessage(), cause);
            }
        });

        try {
            future.join();
        } catch (CompletionException ex) {
            throw ex.getCause() instanceof RuntimeException ? (RuntimeException)
ex.getCause() : ex;
        }
    }
```

- 有关 API 详细信息，请参阅《AWS SDK for Java 2.x API Reference》中的 DeleteMaintenanceWindow。

JavaScript

适用于 JavaScript 的 SDK (v3)

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import { DeleteMaintenanceWindowCommand, SSMClient } from "@aws-sdk/client-ssm";
import { parseArgs } from "node:util";

/**
 * Delete an SSM maintenance window.
 * @param {{ windowId: string }}
 */
export const main = async ({ windowId }) => {
```

```
  const client = new SSMClient({});
  try {
    await client.send(
      new DeleteMaintenanceWindowCommand({ WindowId: windowId }),
    );
    console.log(`Maintenance window '${windowId}' deleted.`);
    return { Deleted: true };
  } catch (caught) {
    if (caught instanceof Error && caught.name === "MissingParameter") {
      console.warn(`${caught.message}. Did you provide this value?`);
    } else {
      throw caught;
    }
  }
};
```

- 有关 API 详细信息，请参阅《适用于 JavaScript 的 AWS SDK API Reference》中的 DeleteMaintenanceWindow。

PowerShell

Tools for PowerShell V4

示例 1：此示例删除维护时段。

```
Remove-SSMMaintenanceWindow -WindowId "mw-06d59c1a07c022145"
```

输出：

```
mw-06d59c1a07c022145
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DeleteMaintenanceWindow。

Tools for PowerShell V5

示例 1：此示例删除维护时段。

```
Remove-SSMMaintenanceWindow -WindowId "mw-06d59c1a07c022145"
```

输出：

```
mw-06d59c1a07c022145
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DeleteMaintenanceWindow。

Python

适用于 Python 的 SDK (Boto3)

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
class MaintenanceWindowWrapper:
    """Encapsulates AWS Systems Manager maintenance window actions."""

    def __init__(self, ssm_client):
        """
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.window_id = None
        self.name = None

    @classmethod
    def from_client(cls):
        ssm_client = boto3.client("ssm")
        return cls(ssm_client)


    def delete(self):
        """
        Delete the associated AWS Systems Manager maintenance window.
        """
        if self.window_id is None:
            return

        try:
            self.ssm_client.delete_maintenance_window(WindowId=self.window_id)
```

```
            logger.info("Deleted maintenance window %s.", self.window_id)
            print(f"Deleted maintenance window {self.name}")
            self.window_id = None
        except ClientError as err:
            logger.error(
                "Couldn't delete maintenance window %s. Here's why: %s: %s",
                self.window_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 有关 API 详细信息，请参阅《适用于 Python 的 AWS SDK (Boto3) API 参考》中的 DeleteMaintenanceWindow 。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 `DeleteOpsItem` 和 AWS SDK 搭配使用

以下代码示例演示了如何使用 `DeleteOpsItem`。

Python

适用于 Python 的 SDK (Boto3)

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
class OpsItemWrapper:
    """Encapsulates AWS Systems Manager OpsItem actions."""

    def __init__(self, ssm_client):
        """
        :param ssm_client: A Boto3 Systems Manager client.
        """
```

```
        self.ssm_client = ssm_client
        self.id = None

    @classmethod
    def from_client(cls):
        """
        :return: A OpsItemWrapper instance.
        """
        ssm_client = boto3.client("ssm")
        return cls(ssm_client)


    def delete(self):
        """
        Delete the OpsItem.
        """
        if self.id is None:
            return
        try:
            self.ssm_client.delete_ops_item(OpsItemId=self.id)
            print(f"Deleted ops item with id {self.id}")
            self.id = None
        except ClientError as err:
            logger.error(
                "Couldn't delete ops item %s. Here's why: %s: %s",
                self.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 有关 API 的详细信息，请参阅《AWS SDK for Python（Boto3）API 参考》中的 DeleteOpsItem。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DeleteParameter** 与 CLI 配合使用

以下代码示例演示如何使用 DeleteParameter。

CLI

AWS CLI

删除参数

以下 delete-parameter 示例将删除指定的一个参数。

```
aws ssm delete-parameter \
    --name "MyParameter"
```

此命令不生成任何输出。

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的使用 Parameter Store。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DeleteParameter。

PowerShell

Tools for PowerShell V4

示例 1：此示例删除一个参数。如果此命令成功，则无任何输出。

```
Remove-SSMParameter -Name "helloWorld"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DeleteParameter。

Tools for PowerShell V5

示例 1：此示例删除一个参数。如果此命令成功，则无任何输出。

```
Remove-SSMParameter -Name "helloWorld"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DeleteParameter。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DeletePatchBaseline** 与 CLI 配合使用

以下代码示例演示如何使用 DeletePatchBaseline。

CLI

AWS CLI

删除补丁基准

以下 delete-patch-baseline 示例将删除指定的补丁基准。

```
aws ssm delete-patch-baseline \
    --baseline-id "pb-045f10b4f382baeda"
```

输出：

```
{
    "BaselineId": "pb-045f10b4f382baeda"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的更新或删除补丁基准（控制台）。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DeletePatchBaseline。

PowerShell

Tools for PowerShell V4

示例 1：此示例删除补丁基准。

```
Remove-SSMPatchBaseline -BaselineId "pb-045f10b4f382baeda"
```

输出：

```
pb-045f10b4f382baeda
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DeletePatchBaseline。

Tools for PowerShell V5

示例 1：此示例删除补丁基准。

```
Remove-SSMPatchBaseline -BaselineId "pb-045f10b4f382baeda"
```

输出：

```
pb-045f10b4f382baeda
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DeletePatchBaseline。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DeregisterManagedInstance** 与 CLI 配合使用

以下代码示例演示如何使用 DeregisterManagedInstance。

CLI

AWS CLI

取消注册托管式实例

以下 deregister-managed-instance 示例取消注册指定的托管式实例。

```
aws ssm deregister-managed-instance \
    --instance-id 'mi-08ab247cdfEXAMPLE'
```

此命令不生成任何输出。

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的在混合和多云环境中取消注册托管式节点。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DeregisterManagedInstance。

PowerShell

Tools for PowerShell V4

示例 1：此示例取消注册托管式实例。如果此命令成功，则无任何输出。

```
Unregister-SSMManagedInstance -InstanceId "mi-08ab247cdf1046573"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DeregisterManagedInstance。

Tools for PowerShell V5

示例 1：此示例取消注册托管式实例。如果此命令成功，则无任何输出。

```
Unregister-SSMManagedInstance -InstanceId "mi-08ab247cdf1046573"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DeregisterManagedInstance。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DeregisterPatchBaselineForPatchGroup** 与 CLI 配合使用

以下代码示例演示如何使用 DeregisterPatchBaselineForPatchGroup。

CLI

AWS CLI

从补丁基准取消注册补丁组

以下 deregister-patch-baseline-for-patch-group 示例从指定的补丁基准中取消注册指定的补丁组。

```
aws ssm deregister-patch-baseline-for-patch-group \
    --patch-group "Production" \
    --baseline-id "pb-0ca44a362fEXAMPLE"
```

输出：

```
{
  "PatchGroup":"Production",
  "BaselineId":"pb-0ca44a362fEXAMPLE"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的将补丁组添加到补丁基准。

- 有关 API 的详细信息，请参阅《AWS CLI Command Reference》中的 DeregisterPatchBaselineForPatchGroup。

PowerShell

Tools for PowerShell V4

示例 1：此示例从补丁基准中取消注册补丁组。

```
Unregister-SSMPatchBaselineForPatchGroup -BaselineId "pb-045f10b4f382baeda" -
PatchGroup "Production"
```

输出：

```
BaselineId           PatchGroup
----------           ----------
pb-045f10b4f382baeda Production
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DeregisterPatchBaselineForPatchGroup。

Tools for PowerShell V5

示例 1：此示例从补丁基准中取消注册补丁组。

```
Unregister-SSMPatchBaselineForPatchGroup -BaselineId "pb-045f10b4f382baeda" -
PatchGroup "Production"
```

输出：

```
BaselineId           PatchGroup
----------           ----------
pb-045f10b4f382baeda Production
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DeregisterPatchBaselineForPatchGroup。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DeregisterTargetFromMaintenanceWindow** 与 CLI 配合使用

以下代码示例演示如何使用 DeregisterTargetFromMaintenanceWindow。

CLI

AWS CLI

从维护时段删除目标

以下 deregister-target-from-maintenance-window 示例从指定的维护时段中删除指定的目标。

```
aws ssm deregister-target-from-maintenance-window \
    --window-id "mw-ab12cd34ef56gh78" \
    --window-target-id "1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
```

输出：

```
{
    "WindowId":"mw-ab12cd34ef56gh78",
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的更新维护时段（AWS CLI）。

- 有关 API 的详细信息，请参阅《AWS CLI Command Reference》中的 DeregisterTargetFromMaintenanceWindow。

PowerShell

Tools for PowerShell V4

示例 1：此示例从维护时段中删除目标。

```
Unregister-SSMTargetFromMaintenanceWindow -WindowTargetId
 "6ab5c208-9fc4-4697-84b7-b02a6cc25f7d" -WindowId "mw-06cf17cbefcb4bf4f"
```

输出：

```
WindowId             WindowTargetId
--------             --------------
mw-06cf17cbefcb4bf4f 6ab5c208-9fc4-4697-84b7-b02a6cc25f7d
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DeregisterTargetFromMaintenanceWindow。

Tools for PowerShell V5

示例 1：此示例从维护时段中删除目标。

```
Unregister-SSMTargetFromMaintenanceWindow -WindowTargetId
 "6ab5c208-9fc4-4697-84b7-b02a6cc25f7d" -WindowId "mw-06cf17cbefcb4bf4f"
```

输出：

```
WindowId             WindowTargetId
--------             --------------
mw-06cf17cbefcb4bf4f 6ab5c208-9fc4-4697-84b7-b02a6cc25f7d
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DeregisterTargetFromMaintenanceWindow。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DeregisterTaskFromMaintenanceWindow** 与 CLI 配合使用

以下代码示例演示如何使用 `DeregisterTaskFromMaintenanceWindow`。

CLI

AWS CLI

从维护时段删除任务

以下 deregister-task-from-maintenance-window 示例从指定的维护时段中删除指定的任务。

```
aws ssm deregister-task-from-maintenance-window \
    --window-id "mw-ab12cd34ef56gh78" \
    --window-task-id "1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d5e6c"
```

输出：

```
{
    "WindowTaskId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d5e6c",
    "WindowId":"mw-ab12cd34ef56gh78"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的 Systems Manager 维护时段教程（AWS CLI）。

- 有关 API 的详细信息，请参阅《AWS CLI Command Reference》中的 DeregisterTaskFromMaintenanceWindow。

PowerShell

Tools for PowerShell V4

示例 1：此示例从维护时段中删除任务。

```
Unregister-SSMTaskFromMaintenanceWindow -WindowTaskId "f34a2c47-ddfd-4c85-
a88d-72366b69af1b" -WindowId "mw-03a342e62c96d31b0"
```

输出：

```
WindowId             WindowTaskId
--------             ------------
mw-03a342e62c96d31b0 f34a2c47-ddfd-4c85-a88d-72366b69af1b
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DeregisterTaskFromMaintenanceWindow。

Tools for PowerShell V5

示例 1：此示例从维护时段中删除任务。

```
Unregister-SSMTaskFromMaintenanceWindow -WindowTaskId "f34a2c47-ddfd-4c85-
a88d-72366b69af1b" -WindowId "mw-03a342e62c96d31b0"
```

输出：

```
WindowId             WindowTaskId
--------             ------------
mw-03a342e62c96d31b0 f34a2c47-ddfd-4c85-a88d-72366b69af1b
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DeregisterTaskFromMaintenanceWindow。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeActivations** 与 CLI 配合使用

以下代码示例演示如何使用 DescribeActivations。

CLI

AWS CLI

描述激活

以下 describe-activations 示例列出有关您 AWS 账户中激活的详细信息。

```
aws ssm describe-activations
```

输出：

```
{
    "ActivationList": [
        {
            "ActivationId": "5743558d-563b-4457-8682-d16c3EXAMPLE",
            "Description": "Example1",
            "IamRole": "HybridWebServersRole,
            "RegistrationLimit": 5,
            "RegistrationsCount": 5,
```

```
            "ExpirationDate": 1584316800.0,
            "Expired": false,
            "CreatedDate": 1581954699.792
        },
        {
            "ActivationId": "3ee0322b-f62d-40eb-b672-13ebfEXAMPLE",
            "Description": "Example2",
            "IamRole": "HybridDatabaseServersRole",
            "RegistrationLimit": 5,
            "RegistrationsCount": 5,
            "ExpirationDate": 1580515200.0,
            "Expired": true,
            "CreatedDate": 1578064132.002
        },
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的步骤 4：为混合环境创建托管式实例激活。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeActivations。

PowerShell

Tools for PowerShell V4

示例 1：此示例提供有关您账户激活的详细信息。

```
Get-SSMActivation
```

输出：

```
ActivationId        : 08e51e79-1e36-446c-8e63-9458569c1363
CreatedDate         : 3/1/2017 12:01:51 AM
DefaultInstanceName : MyWebServers
Description         :
ExpirationDate      : 3/2/2017 12:01:51 AM
Expired             : False
IamRole             : AutomationRole
RegistrationLimit   : 10
RegistrationsCount  : 0
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeActivations。

Tools for PowerShell V5

示例 1：此示例提供有关您账户激活的详细信息。

```
Get-SSMActivation
```

输出：

```
ActivationId        : 08e51e79-1e36-446c-8e63-9458569c1363
CreatedDate         : 3/1/2017 12:01:51 AM
DefaultInstanceName : MyWebServers
Description         :
ExpirationDate      : 3/2/2017 12:01:51 AM
Expired             : False
IamRole             : AutomationRole
RegistrationLimit   : 10
RegistrationsCount  : 0
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeActivations。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeAssociation** 与 CLI 配合使用

以下代码示例演示如何使用 DescribeAssociation。

CLI

AWS CLI

示例 1：获取关联的详细信息

以下 describe-association 示例描述指定关联 ID 的关联。

```
aws ssm describe-association \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

输出：

```
{
    "AssociationDescription": {
        "Name": "AWS-GatherSoftwareInventory",
        "AssociationVersion": "1",
        "Date": 1534864780.995,
        "LastUpdateAssociationDate": 1543235759.81,
        "Overview": {
            "Status": "Success",
            "AssociationStatusAggregatedCount": {
                "Success": 2
            }
        },
        "DocumentVersion": "$DEFAULT",
        "Parameters": {
            "applications": [
                "Enabled"
            ],
            "awsComponents": [
                "Enabled"
            ],
            "customInventory": [
                "Enabled"
            ],
            "files": [
                ""
            ],
            "instanceDetailedInformation": [
                "Enabled"
            ],
            "networkConfig": [
                "Enabled"
            ],
            "services": [
                "Enabled"
            ],
            "windowsRegistry": [
                ""
            ],
            "windowsRoles": [
                "Enabled"
            ],
            "windowsUpdates": [
```

```
                "Enabled"
            ]
        },
        "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
        "Targets": [
            {
                "Key": "InstanceIds",
                "Values": [
                    "*"
                ]
            }
        ],
        "ScheduleExpression": "rate(24 hours)",
        "LastExecutionDate": 1550501886.0,
        "LastSuccessfulExecutionDate": 1550501886.0,
        "AssociationName": "Inventory-Association"
    }
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的编辑和创建关联的新版本。

示例 2：获取特定实例和文档的关联的详细信息

以下 describe-association 示例描述实例和文档之间的关联。

```
aws ssm describe-association \
    --instance-id "i-1234567890abcdef0" \
    --name "AWS-UpdateSSMAgent"
```

输出：

```
{
    "AssociationDescription": {
        "Status": {
            "Date": 1487876122.564,
            "Message": "Associated with AWS-UpdateSSMAgent",
            "Name": "Associated"
        },
        "Name": "AWS-UpdateSSMAgent",
        "InstanceId": "i-1234567890abcdef0",
        "Overview": {
            "Status": "Pending",
```

```
            "DetailedStatus": "Associated",
            "AssociationStatusAggregatedCount": {
                "Pending": 1
            }
        },
        "AssociationId": "d8617c07-2079-4c18-9847-1234567890ab",
        "DocumentVersion": "$DEFAULT",
        "LastUpdateAssociationDate": 1487876122.564,
        "Date": 1487876122.564,
        "Targets": [
            {
                "Values": [
                    "i-1234567890abcdef0"
                ],
                "Key": "InstanceIds"
            }
        ]
    }
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的编辑和创建关联的新版本。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeAssociation。

PowerShell

Tools for PowerShell V4

示例 1：此示例描述实例和文档之间的关联。

```
Get-SSMAssociation -InstanceId "i-0000293ffd8c57862" -Name "AWS-UpdateSSMAgent"
```

输出：

```
Name                  : AWS-UpdateSSMAgent
InstanceId            : i-0000293ffd8c57862
Date                  : 2/23/2017 6:55:22 PM
Status.Name           : Pending
Status.Date           : 2/20/2015 8:31:11 AM
Status.Message        : temp_status_change
Status.AdditionalInfo : Additional-Config-Needed
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeAssociation。

Tools for PowerShell V5

示例 1：此示例描述实例和文档之间的关联。

```
Get-SSMAssociation -InstanceId "i-0000293ffd8c57862" -Name "AWS-UpdateSSMAgent"
```

输出：

```
Name                  : AWS-UpdateSSMAgent
InstanceId            : i-0000293ffd8c57862
Date                  : 2/23/2017 6:55:22 PM
Status.Name           : Pending
Status.Date           : 2/20/2015 8:31:11 AM
Status.Message        : temp_status_change
Status.AdditionalInfo : Additional-Config-Needed
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeAssociation。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeAssociationExecutionTargets** 与 CLI 配合使用

以下代码示例演示如何使用 DescribeAssociationExecutionTargets。

CLI

AWS CLI

获取关联执行的详细信息

以下 describe-association-execution-targets 示例描述指定的关联执行。

```
aws ssm describe-association-execution-targets \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab" \
    --execution-id "7abb6378-a4a5-4f10-8312-0123456789ab"
```

输出：

```
{
    "AssociationExecutionTargets": [
        {
            "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
            "AssociationVersion": "1",
            "ExecutionId": "7abb6378-a4a5-4f10-8312-0123456789ab",
            "ResourceId": "i-1234567890abcdef0",
            "ResourceType": "ManagedInstance",
            "Status": "Success",
            "DetailedStatus": "Success",
            "LastExecutionDate": 1550505538.497,
            "OutputSource": {
                "OutputSourceId": "97fff367-fc5a-4299-aed8-0123456789ab",
                "OutputSourceType": "RunCommand"
            }
        }
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的查看关联历史记录。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeAssociationExecutionTargets。

PowerShell

Tools for PowerShell V4

示例 1：此示例显示作为关联执行目标组成部分的资源 ID 及其执行状态

```
Get-SSMAssociationExecutionTarget -AssociationId 123a45a0-
c678-9012-3456-78901234db5e -ExecutionId 123a45a0-c678-9012-3456-78901234db5e |
 Select-Object ResourceId, Status
```

输出：

```
ResourceId           Status
----------           ------
i-0b1b2a3456f7a890b  Success
```

```
i-01c12a45d6fc7a89f  Success
i-0a1caf234f56d7dc8  Success
i-012a3fd45af6dbcfe  Failed
i-0ddc1df23c4a5fb67  Success
```

示例 2：此命令检查自昨天以来与命令文档关联的特定自动化的特定执行。它会进一步检查关联执行是否失败；如果失败，它将显示执行的命令调用详细信息以及实例 ID

```
$AssociationExecution= Get-SSMAssociationExecutionTarget -
AssociationId 1c234567-890f-1aca-a234-5a678d901cb0 -ExecutionId
 12345ca12-3456-2345-2b45-23456789012 |
    Where-Object {$_.LastExecutionDate -gt (Get-Date -Hour 00 -Minute
 00).AddDays(-1)}

foreach ($execution in $AssociationExecution) {
    if($execution.Status -ne 'Success'){
        Write-Output "There was an issue executing the association
 $($execution.AssociationId) on $($execution.ResourceId)"
        Get-SSMCommandInvocation -CommandId
 $execution.OutputSource.OutputSourceId -Detail:$true | Select-Object -
ExpandProperty CommandPlugins
    }
}
```

输出：

```
There was an issue executing the association 1c234567-890f-1aca-a234-5a678d901cb0
 on i-0a1caf234f56d7dc8


Name                   : aws:runPowerShellScript
Output                 :
                           ----------ERROR-------
                           failed to run commands: exit status 1
OutputS3BucketName     :
OutputS3KeyPrefix      :
OutputS3Region         : eu-west-1
ResponseCode           : 1
ResponseFinishDateTime : 5/29/2019 11:04:49 AM
ResponseStartDateTime  : 5/29/2019 11:04:49 AM
StandardErrorUrl       :
StandardOutputUrl      :
Status                 : Failed
```

```
StatusDetails          : Failed
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeAssociationExecutionTargets。

Tools for PowerShell V5

示例 1：此示例显示作为关联执行目标组成部分的资源 ID 及其执行状态

```
Get-SSMAssociationExecutionTarget -AssociationId 123a45a0-
c678-9012-3456-78901234db5e -ExecutionId 123a45a0-c678-9012-3456-78901234db5e |
 Select-Object ResourceId, Status
```

输出：

```
ResourceId           Status
----------           ------
i-0b1b2a3456f7a890b  Success
i-01c12a45d6fc7a89f  Success
i-0a1caf234f56d7dc8  Success
i-012a3fd45af6dbcfe  Failed
i-0ddc1df23c4a5fb67  Success
```

示例 2：此命令检查自昨天以来与命令文档关联的特定自动化的特定执行。它会进一步检查关联执行是否失败；如果失败，它将显示执行的命令调用详细信息以及实例 ID

```
$AssociationExecution= Get-SSMAssociationExecutionTarget -
AssociationId 1c234567-890f-1aca-a234-5a678d901cb0 -ExecutionId
 12345ca12-3456-2345-2b45-23456789012 |
    Where-Object {$_.LastExecutionDate -gt (Get-Date -Hour 00 -Minute
 00).AddDays(-1)}

foreach ($execution in $AssociationExecution) {
    if($execution.Status -ne 'Success'){
        Write-Output "There was an issue executing the association
 $($execution.AssociationId) on $($execution.ResourceId)"
        Get-SSMCommandInvocation -CommandId
 $execution.OutputSource.OutputSourceId -Detail:$true | Select-Object -
ExpandProperty CommandPlugins
    }
}
```

输出：

```
There was an issue executing the association 1c234567-890f-1aca-a234-5a678d901cb0
 on i-0a1caf234f56d7dc8


Name                   : aws:runPowerShellScript
Output                 :
                         ----------ERROR-------
                         failed to run commands: exit status 1
OutputS3BucketName     :
OutputS3KeyPrefix      :
OutputS3Region         : eu-west-1
ResponseCode           : 1
ResponseFinishDateTime : 5/29/2019 11:04:49 AM
ResponseStartDateTime  : 5/29/2019 11:04:49 AM
StandardErrorUrl       :
StandardOutputUrl      :
Status                 : Failed
StatusDetails          : Failed
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeAssociationExecutionTargets。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeAssociationExecutions** 与 CLI 配合使用

以下代码示例演示如何使用 DescribeAssociationExecutions。

CLI

AWS CLI

示例 1：获取关联所有执行的详细信息

以下 describe-association-executions 示例描述指定关联的所有执行。

```
aws ssm describe-association-executions \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

输出：

```
{
    "AssociationExecutions": [
        {
            "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
            "AssociationVersion": "1",
            "ExecutionId": "474925ef-1249-45a2-b93d-0123456789ab",
            "Status": "Success",
            "DetailedStatus": "Success",
            "CreatedTime": 1550505827.119,
            "ResourceCountByStatus": "{Success=1}"
        },
        {
            "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
            "AssociationVersion": "1",
            "ExecutionId": "7abb6378-a4a5-4f10-8312-0123456789ab",
            "Status": "Success",
            "DetailedStatus": "Success",
            "CreatedTime": 1550505536.843,
            "ResourceCountByStatus": "{Success=1}"
        },
        ...
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的查看关联历史记录。

示例 2：获取特定日期和时间之后关联的所有执行的详细信息

以下 describe-association-executions 示例描述指定日期和时间之后关联的所有执行。

```
aws ssm describe-association-executions \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab" \
    --filters "Key=CreatedTime,Value=2019-02-18T16:00:00Z,Type=GREATER_THAN"
```

输出：

```
{
    "AssociationExecutions": [
        {
```

```
            "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
            "AssociationVersion": "1",
            "ExecutionId": "474925ef-1249-45a2-b93d-0123456789ab",
            "Status": "Success",
            "DetailedStatus": "Success",
            "CreatedTime": 1550505827.119,
            "ResourceCountByStatus": "{Success=1}"
        },
        {
            "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
            "AssociationVersion": "1",
            "ExecutionId": "7abb6378-a4a5-4f10-8312-0123456789ab",
            "Status": "Success",
            "DetailedStatus": "Success",
            "CreatedTime": 1550505536.843,
            "ResourceCountByStatus": "{Success=1}"
        },
        ...
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的查看关联历史记录。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeAssociationExecutions。

PowerShell

Tools for PowerShell V4

示例 1：此示例返回所提供关联 ID 的执行

```
Get-SSMAssociationExecution -AssociationId 123a45a0-c678-9012-3456-78901234db5e
```

输出：

```
AssociationId         : 123a45a0-c678-9012-3456-78901234db5e
AssociationVersion    : 2
CreatedTime           : 3/2/2019 8:53:29 AM
DetailedStatus        :
ExecutionId           : 123a45a0-c678-9012-3456-78901234db5e
LastExecutionDate     : 1/1/0001 12:00:00 AM
```

```
ResourceCountByStatus : {Success=4}
Status                : Success
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeAssociationExecutions。

Tools for PowerShell V5

示例 1：此示例返回所提供关联 ID 的执行

```
Get-SSMAssociationExecution -AssociationId 123a45a0-c678-9012-3456-78901234db5e
```

输出：

```
AssociationId         : 123a45a0-c678-9012-3456-78901234db5e
AssociationVersion    : 2
CreatedTime           : 3/2/2019 8:53:29 AM
DetailedStatus        :
ExecutionId           : 123a45a0-c678-9012-3456-78901234db5e
LastExecutionDate     : 1/1/0001 12:00:00 AM
ResourceCountByStatus : {Success=4}
Status                : Success
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeAssociationExecutions。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeAutomationExecutions** 与 CLI 配合使用

以下代码示例演示如何使用 DescribeAutomationExecutions。

CLI

AWS CLI

描述自动化执行

以下 describe-automation-executions 示例显示了有关自动化执行的详细信息。

```
aws ssm describe-automation-executions \
```

```
    --filters Key=ExecutionId,Values=73c8eef8-f4ee-4a05-820c-e354fEXAMPLE
```

输出：

```
{
    "AutomationExecutionMetadataList": [
        {
            "AutomationExecutionId": "73c8eef8-f4ee-4a05-820c-e354fEXAMPLE",
            "DocumentName": "AWS-StartEC2Instance",
            "DocumentVersion": "1",
            "AutomationExecutionStatus": "Success",
            "ExecutionStartTime": 1583737233.748,
            "ExecutionEndTime": 1583737234.719,
            "ExecutedBy": "arn:aws:sts::29884EXAMPLE:assumed-role/
mw_service_role/OrchestrationService",
            "LogFile": "",
            "Outputs": {},
            "Mode": "Auto",
            "Targets": [],
            "ResolvedTargets": {
                "ParameterValues": [],
                "Truncated": false
            },
            "AutomationType": "Local"
        }
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的运行简单的自动化工作流。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeAutomationExecutions。

PowerShell

Tools for PowerShell V4

示例 1：此示例描述与您的账户关联的所有活动和已终止的自动化执行。

```
Get-SSMAutomationExecutionList
```

输出：

```
AutomationExecutionId     : 4105a4fc-f944-11e6-9d32-8fb2db27a909
AutomationExecutionStatus : Failed
DocumentName              : AWS-UpdateLinuxAmi
DocumentVersion           : 1
ExecutedBy                : admin
ExecutionEndTime          : 2/22/2017 9:17:08 PM
ExecutionStartTime        : 2/22/2017 9:17:02 PM
LogFile                   :
Outputs                   : {[createImage.ImageId,
 Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
```

示例 2：此示例显示 AutomationExecutionStatus 并非“成功”的执行的 ExecutionID、文档、执行开始/结束时间戳

```
Get-SSMAutomationExecutionList | Where-Object AutomationExecutionStatus
 -ne "Success" | Select-Object AutomationExecutionId, DocumentName,
 AutomationExecutionStatus, ExecutionStartTime, ExecutionEndTime | Format-Table -
AutoSize
```

输出：

```
AutomationExecutionId                DocumentName
 AutomationExecutionStatus ExecutionStartTime   ExecutionEndTime
--------------------                 ------------
 ------------------------- ------------------   ----------------
e1d2bad3-4567-8901-ae23-456c7c8901be AWS-UpdateWindowsAmi
 Cancelled                 4/16/2019 5:37:04 AM 4/16/2019 5:47:29 AM
61234567-a7f8-90e1-2b34-567b8bf9012c Fixed-UpdateAmi
 Cancelled                 4/16/2019 5:33:04 AM 4/16/2019 5:40:15 AM
91234d56-7e89-0ac1-2aee-34ea5d6a7c89 AWS-UpdateWindowsAmi
 Failed                    4/16/2019 5:22:46 AM 4/16/2019 5:27:29 AM
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeAutomationExecutions。

Tools for PowerShell V5

示例 1：此示例描述与您的账户关联的所有活动和已终止的自动化执行。

```
Get-SSMAutomationExecutionList
```

输出：

```
AutomationExecutionId     : 4105a4fc-f944-11e6-9d32-8fb2db27a909
AutomationExecutionStatus : Failed
DocumentName              : AWS-UpdateLinuxAmi
DocumentVersion           : 1
ExecutedBy                : admin
ExecutionEndTime          : 2/22/2017 9:17:08 PM
ExecutionStartTime        : 2/22/2017 9:17:02 PM
LogFile                   :
Outputs                   : {[createImage.ImageId,
 Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
```

示例 2：此示例显示 AutomationExecutionStatus 并非“成功”的执行的 ExecutionID、文档、执行开始/结束时间戳

```
Get-SSMAutomationExecutionList | Where-Object AutomationExecutionStatus
 -ne "Success" | Select-Object AutomationExecutionId, DocumentName,
 AutomationExecutionStatus, ExecutionStartTime, ExecutionEndTime | Format-Table -
AutoSize
```

输出：

```
AutomationExecutionId                DocumentName
 AutomationExecutionStatus ExecutionStartTime   ExecutionEndTime
---------------------                ------------
 ------------------------- ------------------   ----------------
e1d2bad3-4567-8901-ae23-456c7c8901be AWS-UpdateWindowsAmi
 Cancelled                 4/16/2019 5:37:04 AM 4/16/2019 5:47:29 AM
61234567-a7f8-90e1-2b34-567b8bf9012c Fixed-UpdateAmi
 Cancelled                 4/16/2019 5:33:04 AM 4/16/2019 5:40:15 AM
91234d56-7e89-0ac1-2aee-34ea5d6a7c89 AWS-UpdateWindowsAmi
 Failed                    4/16/2019 5:22:46 AM 4/16/2019 5:27:29 AM
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeAutomationExecutions。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeAutomationStepExecutions** 与 CLI 配合使用

以下代码示例演示如何使用 DescribeAutomationStepExecutions。

CLI

AWS CLI

示例 1：描述自动化执行的所有步骤

以下 describe-automation-step-executions 示例显示有关自动化执行步骤的详细信息。

```
aws ssm describe-automation-step-executions \
    --automation-execution-id 73c8eef8-f4ee-4a05-820c-e354fEXAMPLE
```

输出：

```
{
    "StepExecutions": [
        {
            "StepName": "startInstances",
            "Action": "aws:changeInstanceState",
            "ExecutionStartTime": 1583737234.134,
            "ExecutionEndTime": 1583737234.672,
            "StepStatus": "Success",
            "Inputs": {
                "DesiredState": "\"running\"",
                "InstanceIds": "[\"i-0cb99161f6EXAMPLE\"]"
            },
            "Outputs": {
                "InstanceStates": [
                    "running"
                ]
            },
            "StepExecutionId": "95e70479-cf20-4d80-8018-7e4e2EXAMPLE",
            "OverriddenParameters": {}
        }
    ]
}
```

示例 2：描述自动化执行的特定步骤

以下 describe-automation-step-executions 示例显示了有关自动化执行中特定步骤的详细信息。

```
aws ssm describe-automation-step-executions \
```

```
    --automation-execution-id 73c8eef8-f4ee-4a05-820c-e354fEXAMPLE \
    --filters Key=StepExecutionId,Values=95e70479-cf20-4d80-8018-7e4e2EXAMPLE
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的分步运行自动化工作流（命令行）。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeAutomationStepExecutions。

PowerShell

Tools for PowerShell V4

示例 1：此示例显示有关自动化工作流中所有活动和已终止步骤执行的信息。

```
Get-SSMAutomationStepExecution -AutomationExecutionId e1d2bad3-4567-8901-
ae23-456c7c8901be | Select-Object StepName, Action, StepStatus
```

输出：

```
StepName                  Action                   StepStatus
--------                  ------                   ----------
LaunchInstance            aws:runInstances         Success
OSCompatibilityCheck      aws:runCommand           Success
RunPreUpdateScript        aws:runCommand           Success
UpdateEC2Config           aws:runCommand           Cancelled
UpdateSSMAgent            aws:runCommand           Pending
UpdateAWSPVDriver         aws:runCommand           Pending
UpdateAWSEnaNetworkDriver aws:runCommand           Pending
UpdateAWSNVMe             aws:runCommand           Pending
InstallWindowsUpdates     aws:runCommand           Pending
RunPostUpdateScript       aws:runCommand           Pending
RunSysprepGeneralize      aws:runCommand           Pending
StopInstance              aws:changeInstanceState Pending
CreateImage               aws:createImage          Pending
TerminateInstance         aws:changeInstanceState Pending
```

- 有关 API 详细信息，请参阅《《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeAutomationStepExecutions。

Tools for PowerShell V5

示例 1：此示例显示有关自动化工作流中所有活动和已终止步骤执行的信息。

```
Get-SSMAutomationStepExecution -AutomationExecutionId e1d2bad3-4567-8901-
ae23-456c7c8901be | Select-Object StepName, Action, StepStatus
```

输出：

```
StepName                  Action                   StepStatus
--------                  ------                   ----------
LaunchInstance            aws:runInstances         Success
OSCompatibilityCheck      aws:runCommand           Success
RunPreUpdateScript        aws:runCommand           Success
UpdateEC2Config           aws:runCommand           Cancelled
UpdateSSMAgent            aws:runCommand           Pending
UpdateAWSPVDriver         aws:runCommand           Pending
UpdateAWSEnaNetworkDriver aws:runCommand           Pending
UpdateAWSNVMe             aws:runCommand           Pending
InstallWindowsUpdates     aws:runCommand           Pending
RunPostUpdateScript       aws:runCommand           Pending
RunSysprepGeneralize      aws:runCommand           Pending
StopInstance              aws:changeInstanceState Pending
CreateImage               aws:createImage          Pending
TerminateInstance         aws:changeInstanceState Pending
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeAutomationStepExecutions。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeAvailablePatches** 与 CLI 配合使用

以下代码示例演示如何使用 DescribeAvailablePatches。

CLI

AWS CLI

获取可用补丁

以下 describe-available-patches 示例检索有关 MSRC 严重性为“严重”的所有 Windows Server 2019 可用补丁的详细信息。

```
aws ssm describe-available-patches \
    --
filters "Key=PRODUCT,Values=WindowsServer2019" "Key=MSRC_SEVERITY,Values=Critical"
```

输出：

```
{
    "Patches": [
        {
            "Id": "fe6bd8c2-3752-4c8b-ab3e-1a7ed08767ba",
            "ReleaseDate": 1544047205.0,
            "Title": "2018-11 Update for Windows Server 2019 for x64-based
 Systems (KB4470788)",
            "Description": "Install this update to resolve issues in Windows.
 For a complete listing of the issues that are included in this update, see the
 associated Microsoft Knowledge Base article for more information. After you
 install this item, you may have to restart your computer.",
            "ContentUrl": "https://support.microsoft.com/en-us/kb/4470788",
            "Vendor": "Microsoft",
            "ProductFamily": "Windows",
            "Product": "WindowsServer2019",
            "Classification": "SecurityUpdates",
            "MsrcSeverity": "Critical",
            "KbNumber": "KB4470788",
            "MsrcNumber": "",
            "Language": "All"
        },
        {
            "Id": "c96115e1-5587-4115-b851-22baa46a3f11",
            "ReleaseDate": 1549994410.0,
            "Title": "2019-02 Security Update for Adobe Flash Player for Windows
 Server 2019 for x64-based Systems (KB4487038)",
            "Description": "A security issue has been identified in a Microsoft
 software product that could affect your system. You can help protect your system
 by installing this update from Microsoft. For a complete listing of the issues
 that are included in this update, see the associated Microsoft Knowledge Base
 article. After you install this update, you may have to restart your system.",
            "ContentUrl": "https://support.microsoft.com/en-us/kb/4487038",
            "Vendor": "Microsoft",
            "ProductFamily": "Windows",
            "Product": "WindowsServer2019",
            "Classification": "SecurityUpdates",
            "MsrcSeverity": "Critical",
```

```
            "KbNumber": "KB4487038",
            "MsrcNumber": "",
            "Language": "All"
        },
        ...
    ]
}
```

获取特定补丁的详细信息

以下 describe-available-patches 示例将检索有关指定补丁的详细信息。

```
aws ssm describe-available-patches \
    --filters "Key=PATCH_ID,Values=KB4480979"
```

输出：

```
{
    "Patches": [
        {
            "Id": "680861e3-fb75-432e-818e-d72e5f2be719",
            "ReleaseDate": 1546970408.0,
            "Title": "2019-01 Security Update for Adobe Flash Player for Windows
 Server 2016 for x64-based Systems (KB4480979)",
            "Description": "A security issue has been identified in a Microsoft
 software product that could affect your system. You can help protect your system
 by installing this update from Microsoft. For a complete listing of the issues
 that are included in this update, see the associated Microsoft Knowledge Base
 article. After you install this update, you may have to restart your system.",
            "ContentUrl": "https://support.microsoft.com/en-us/kb/4480979",
            "Vendor": "Microsoft",
            "ProductFamily": "Windows",
            "Product": "WindowsServer2016",
            "Classification": "SecurityUpdates",
            "MsrcSeverity": "Critical",
            "KbNumber": "KB4480979",
            "MsrcNumber": "",
            "Language": "All"
        }
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的 Patch Manager 工作原理。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeAvailablePatches。

PowerShell

Tools for PowerShell V4

示例 1：此示例获取 MSRC 严重性为“严重”的、适用于 Windows Server 2012 的所有可用补丁。此示例使用的语法需要 PowerShell 版本 3 或更高版本。

```
$filter1 = @{Key="PRODUCT";Values=@("WindowsServer2012")}
$filter2 = @{Key="MSRC_SEVERITY";Values=@("Critical")}

Get-SSMAvailablePatch -Filter $filter1,$filter2
```

输出：

```
Classification : SecurityUpdates
ContentUrl     : https://support.microsoft.com/en-us/kb/2727528
Description    : A security issue has been identified that could allow an
 unauthenticated remote attacker to compromise your system and gain control
                  over it. You can help protect your system by installing this
 update from Microsoft. After you install this update, you may have to
                  restart your system.
Id             : 1eb507be-2040-4eeb-803d-abc55700b715
KbNumber       : KB2727528
Language       : All
MsrcNumber     : MS12-072
MsrcSeverity   : Critical
Product        : WindowsServer2012
ProductFamily  : Windows
ReleaseDate    : 11/13/2012 6:00:00 PM
Title          : Security Update for Windows Server 2012 (KB2727528)
Vendor         : Microsoft
...
```

示例 2：对于 PowerShell 版本 2，必须使用 New-Object 创建每个筛选器。

```
$filter1 = New-Object
 Amazon.SimpleSystemsManagement.Model.PatchOrchestratorFilter
$filter1.Key = "PRODUCT"
```

```
$filter1.Values = "WindowsServer2012"
$filter2 = New-Object
 Amazon.SimpleSystemsManagement.Model.PatchOrchestratorFilter
$filter2.Key = "MSRC_SEVERITY"
$filter2.Values = "Critical"

Get-SSMAvailablePatch -Filter $filter1,$filter2
```

示例 3：此示例获取过去 20 天内发布且适用于与 WindowsServer2019 匹配产品的所有更新

```
Get-SSMAvailablePatch | Where-Object ReleaseDate -ge (Get-Date).AddDays(-20)
 | Where-Object Product -eq "WindowsServer2019" | Select-Object ReleaseDate,
 Product, Title
```

输出：

```
ReleaseDate         Product           Title
-----------         -------           -----
4/9/2019 5:00:12 PM WindowsServer2019 2019-04 Security Update for Adobe Flash
 Player for Windows Server 2019 for x64-based Systems (KB4493478)
4/9/2019 5:00:06 PM WindowsServer2019 2019-04 Cumulative Update for Windows
 Server 2019 for x64-based Systems (KB4493509)
4/2/2019 5:00:06 PM WindowsServer2019 2019-03 Servicing Stack Update for Windows
 Server 2019 for x64-based Systems (KB4493510)
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeAvailablePatches。

Tools for PowerShell V5

示例 1：此示例获取 MSRC 严重性为“严重”的、适用于 Windows Server 2012 的所有可用补丁。此示例使用的语法需要 PowerShell 版本 3 或更高版本。

```
$filter1 = @{Key="PRODUCT";Values=@("WindowsServer2012")}
$filter2 = @{Key="MSRC_SEVERITY";Values=@("Critical")}

Get-SSMAvailablePatch -Filter $filter1,$filter2
```

输出：

```
Classification : SecurityUpdates
ContentUrl     : https://support.microsoft.com/en-us/kb/2727528
```

```
Description    : A security issue has been identified that could allow an
 unauthenticated remote attacker to compromise your system and gain control
                 over it. You can help protect your system by installing this
 update from Microsoft. After you install this update, you may have to
                 restart your system.
Id             : 1eb507be-2040-4eeb-803d-abc55700b715
KbNumber       : KB2727528
Language       : All
MsrcNumber     : MS12-072
MsrcSeverity   : Critical
Product        : WindowsServer2012
ProductFamily  : Windows
ReleaseDate    : 11/13/2012 6:00:00 PM
Title          : Security Update for Windows Server 2012 (KB2727528)
Vendor         : Microsoft
...
```

示例 2：对于 PowerShell 版本 2，必须使用 New-Object 创建每个筛选器。

```
$filter1 = New-Object
 Amazon.SimpleSystemsManagement.Model.PatchOrchestratorFilter
$filter1.Key = "PRODUCT"
$filter1.Values = "WindowsServer2012"
$filter2 = New-Object
 Amazon.SimpleSystemsManagement.Model.PatchOrchestratorFilter
$filter2.Key = "MSRC_SEVERITY"
$filter2.Values = "Critical"

Get-SSMAvailablePatch -Filter $filter1,$filter2
```

示例 3：此示例获取过去 20 天内发布且适用于与 WindowsServer2019 匹配产品的所有更新

```
Get-SSMAvailablePatch | Where-Object ReleaseDate -ge (Get-Date).AddDays(-20)
 | Where-Object Product -eq "WindowsServer2019" | Select-Object ReleaseDate,
 Product, Title
```

输出：

```
ReleaseDate         Product           Title
-----------         -------           -----
4/9/2019 5:00:12 PM WindowsServer2019 2019-04 Security Update for Adobe Flash
 Player for Windows Server 2019 for x64-based Systems (KB4493478)
```

```
4/9/2019 5:00:06 PM WindowsServer2019 2019-04 Cumulative Update for Windows
 Server 2019 for x64-based Systems (KB4493509)
4/2/2019 5:00:06 PM WindowsServer2019 2019-03 Servicing Stack Update for Windows
 Server 2019 for x64-based Systems (KB4493510)
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeAvailablePatches。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeDocument** 与 CLI 配合使用

以下代码示例演示如何使用 DescribeDocument。

CLI

AWS CLI

显示文档的详细信息

以下 describe-document 示例显示有关您 AWS 账户中 Systems Manager 文档的详细信息。

```
aws ssm describe-document \
    --name "Example"
```

输出：

```
{
    "Document": {
        "Hash":
 "fc2410281f40779e694a8b95975d0f9f316da8a153daa94e3d9921102EXAMPLE",
        "HashType": "Sha256",
        "Name": "Example",
        "Owner": "29884EXAMPLE",
        "CreatedDate": 1583257938.266,
        "Status": "Active",
        "DocumentVersion": "1",
        "Description": "Document Example",
        "Parameters": [
```

```
            {
                "Name": "AutomationAssumeRole",
                "Type": "String",
                "Description": "(Required) The ARN of the role that allows
 Automation to perform the actions on your behalf. If no role is specified,
 Systems Manager Automation uses your IAM permissions to execute this document.",
                "DefaultValue": ""
            },
            {
                "Name": "InstanceId",
                "Type": "String",
                "Description": "(Required) The ID of the Amazon EC2 instance.",
                "DefaultValue": ""
            }
        ],
        "PlatformTypes": [
            "Windows",
            "Linux"
        ],
        "DocumentType": "Automation",
        "SchemaVersion": "0.3",
        "LatestVersion": "1",
        "DefaultVersion": "1",
        "DocumentFormat": "YAML",
        "Tags": []
    }
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的创建 Systems Manager 文档。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeDocument。

PowerShell

Tools for PowerShell V4

示例 1：此示例返回有关文档的信息。

```
Get-SSMDocumentDescription -Name "RunShellScript"
```

输出：

```
CreatedDate     : 2/24/2017 5:25:13 AM
DefaultVersion  : 1
Description     : Run an updated script
DocumentType    : Command
DocumentVersion : 1
Hash            :
 f775e5df4904c6fa46686c4722fae9de1950dace25cd9608ff8d622046b68d9b
HashType        : Sha256
LatestVersion   : 1
Name            : RunShellScript
Owner           : 123456789012
Parameters      : {commands}
PlatformTypes   : {Linux}
SchemaVersion   : 2.0
Sha1            :
Status          : Active
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeDocument。

Tools for PowerShell V5

示例 1：此示例返回有关文档的信息。

```
Get-SSMDocumentDescription -Name "RunShellScript"
```

输出：

```
CreatedDate     : 2/24/2017 5:25:13 AM
DefaultVersion  : 1
Description     : Run an updated script
DocumentType    : Command
DocumentVersion : 1
Hash            :
 f775e5df4904c6fa46686c4722fae9de1950dace25cd9608ff8d622046b68d9b
HashType        : Sha256
LatestVersion   : 1
Name            : RunShellScript
Owner           : 123456789012
Parameters      : {commands}
PlatformTypes   : {Linux}
SchemaVersion   : 2.0
Sha1            :
```

```
Status          : Active
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeDocument。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeDocumentPermission** 与 CLI 配合使用

以下代码示例演示如何使用 DescribeDocumentPermission。

CLI

AWS CLI

描述文档权限

以下 describe-document-permission 示例显示有关公开共享 Systems Manager 文档的权限详细信息。

```
aws ssm describe-document-permission \
    --name "Example" \
    --permission-type "Share"
```

输出：

```
{
    "AccountIds": [
        "all"
    ],
    "AccountSharingInfoList": [
        {
            "AccountId": "all",
            "SharedDocumentVersion": "$DEFAULT"
        }
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的共享 Systems Manager 文档。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeDocumentPermission。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出文档的所有版本。

```
Get-SSMDocumentVersionList -Name "RunShellScript"
```

输出：

```
CreatedDate          DocumentVersion IsDefaultVersion Name
-----------          --------------- ---------------- ----
2/24/2017 5:25:13 AM 1               True             RunShellScript
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeDocumentPermission。

Tools for PowerShell V5

示例 1：此示例列出文档的所有版本。

```
Get-SSMDocumentVersionList -Name "RunShellScript"
```

输出：

```
CreatedDate          DocumentVersion IsDefaultVersion Name
-----------          --------------- ---------------- ----
2/24/2017 5:25:13 AM 1               True             RunShellScript
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeDocumentPermission。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 `DescribeEffectiveInstanceAssociations` 与 CLI 配合使用

以下代码示例演示如何使用 `DescribeEffectiveInstanceAssociations`。

CLI

AWS CLI

获取实例有效关联的详细信息

以下 describe-effective-instance-associations 示例检索有关实例有效关联的详细信息。

命令:

```
aws ssm describe-effective-instance-associations --instance-
id "i-1234567890abcdef0"
```

输出：

```
{
    "Associations": [
        {
            "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
            "InstanceId": "i-1234567890abcdef0",
            "Content": "{\n    \"schemaVersion\": \"1.2\",\n    \"description\":
 \"Update the Amazon SSM Agent to the latest version or specified version.\",\n
  \"parameters\": {\n        \"version\": {\n            \"default\": \"\",\n
       \"description\": \"(Optional) A specific version of the Amazon SSM Agent
 to install. If not specified, the agent will be updated to the latest version.
\",\n            \"type\": \"String\"\n        },\n        \"allowDowngrade
\": {\n            \"default\": \"false\",\n            \"description\":
 \"(Optional) Allow the Amazon SSM Agent service to be downgraded to an earlier
 version. If set to false, the service can be upgraded to newer versions only
 (default). If set to true, specify the earlier version.\",\n            \"type
\": \"String\",\n            \"allowedValues\": [\n                \"true\",\n
               \"false\"\n            ]\n        }\n    },\n    \"runtimeConfig
\": {\n        \"aws:updateSsmAgent\": {\n            \"properties\": [\n
         {\n                \"agentName\": \"amazon-ssm-agent\",\n
  \"source\": \"https://s3.{Region}.amazonaws.com/amazon-ssm-{Region}/ssm-agent-
manifest.json\",\n                \"allowDowngrade\": \"{{ allowDowngrade }}\",\n
                \"targetVersion\": \"{{ version }}\"\n                }\n
    ]\n        }\n    }\n}\n",
            "AssociationVersion": "1"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeEffectiveInstanceAssociations。

PowerShell

Tools for PowerShell V4

示例 1：此示例描述实例的有效关联。

```
Get-SSMEffectiveInstanceAssociationList -InstanceId "i-0000293ffd8c57862" -
MaxResult 5
```

输出：

```
AssociationId                        Content
-------------                        -------
d8617c07-2079-4c18-9847-1655fc2698b0 {...
```

示例 2：此示例显示实例有效关联的内容。

```
(Get-SSMEffectiveInstanceAssociationList -InstanceId "i-0000293ffd8c57862" -
MaxResult 5).Content
```

输出：

```
{
    "schemaVersion": "1.2",
    "description": "Update the Amazon SSM Agent to the latest version or
 specified version.",
    "parameters": {
        "version": {
            "default": "",
            "description": "(Optional) A specific version of the Amazon SSM Agent
 to install. If not specified, the agen
t will be updated to the latest version.",
            "type": "String"
        },
        "allowDowngrade": {
            "default": "false",
            "description": "(Optional) Allow the Amazon SSM Agent service to be
 downgraded to an earlier version. If set
```

```
 to false, the service can be upgraded to newer versions only (default). If set
 to true, specify the earlier version.",
            "type": "String",
            "allowedValues": [
                "true",
                "false"
            ]
        }
    },
    "runtimeConfig": {
        "aws:updateSsmAgent": {
            "properties": [
                {
                "agentName": "amazon-ssm-agent",
                "source": "https://s3.{Region}.amazonaws.com/amazon-ssm-{Region}/
ssm-agent-manifest.json",
                "allowDowngrade": "{{ allowDowngrade }}",
                "targetVersion": "{{ version }}"
                }
            ]
        }
    }
}
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeEffectiveInstanceAssociations。

Tools for PowerShell V5

示例 1：此示例描述实例的有效关联。

```
Get-SSMEffectiveInstanceAssociationList -InstanceId "i-0000293ffd8c57862" -
MaxResult 5
```

输出：

```
AssociationId                        Content
-------------                        -------
d8617c07-2079-4c18-9847-1655fc2698b0 {...
```

示例 2：此示例显示实例有效关联的内容。

```
(Get-SSMEffectiveInstanceAssociationList -InstanceId "i-0000293ffd8c57862" -
MaxResult 5).Content
```

输出：

```
{
    "schemaVersion": "1.2",
    "description": "Update the Amazon SSM Agent to the latest version or
 specified version.",
    "parameters": {
        "version": {
            "default": "",
            "description": "(Optional) A specific version of the Amazon SSM Agent
 to install. If not specified, the agen
t will be updated to the latest version.",
            "type": "String"
        },
        "allowDowngrade": {
            "default": "false",
            "description": "(Optional) Allow the Amazon SSM Agent service to be
 downgraded to an earlier version. If set
 to false, the service can be upgraded to newer versions only (default). If set
 to true, specify the earlier version.",
            "type": "String",
            "allowedValues": [
                "true",
                "false"
            ]
        }
    },
    "runtimeConfig": {
        "aws:updateSsmAgent": {
            "properties": [
                {
                "agentName": "amazon-ssm-agent",
                "source": "https://s3.{Region}.amazonaws.com/amazon-ssm-{Region}/
ssm-agent-manifest.json",
                "allowDowngrade": "{{ allowDowngrade }}",
                "targetVersion": "{{ version }}"
                }
            ]
        }
    }
```

```
}
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeEffectiveInstanceAssociations。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 DescribeEffectivePatchesForPatchBaseline 与 CLI 配合使用

以下代码示例演示如何使用 DescribeEffectivePatchesForPatchBaseline。

CLI

AWS CLI

示例 1：获取自定义补丁基准定义的所有补丁

以下 describe-effective-patches-for-patch-baseline 示例返回当前 AWS 账户中由自定义补丁基准定义的补丁。请注意，对于自定义基准，--baseline-id 只需要 ID 。

```
aws ssm describe-effective-patches-for-patch-baseline \
    --baseline-id "pb-08b654cf9b9681f04"
```

输出：

```
{
    "EffectivePatches": [
        {
            "Patch": {
                "Id": "fe6bd8c2-3752-4c8b-ab3e-1a7ed08767ba",
                "ReleaseDate": 1544047205.0,
                "Title": "2018-11 Update for Windows Server 2019 for x64-based
 Systems (KB4470788)",
                "Description": "Install this update to resolve issues in Windows.
 For a complete listing of the issues that are included in this update, see the
 associated Microsoft Knowledge Base article for more information. After you
 install this item, you may have to restart your computer.",
                "ContentUrl": "https://support.microsoft.com/en-us/kb/4470788",
                "Vendor": "Microsoft",
                "ProductFamily": "Windows",
                "Product": "WindowsServer2019",
```

```
                "Classification": "SecurityUpdates",
                "MsrcSeverity": "Critical",
                "KbNumber": "KB4470788",
                "MsrcNumber": "",
                "Language": "All"
            },
            "PatchStatus": {
                "DeploymentStatus": "APPROVED",
                "ComplianceLevel": "CRITICAL",
                "ApprovalDate": 1544047205.0
            }
        },
        {
            "Patch": {
                "Id": "915a6b1a-f556-4d83-8f50-b2e75a9a7e58",
                "ReleaseDate": 1549994400.0,
                "Title": "2019-02 Cumulative Update for .NET Framework 3.5 and
4.7.2 for Windows Server 2019 for x64 (KB4483452)",
                "Description": "A security issue has been identified in a
Microsoft software product that could affect your system. You can help protect
your system by installing this update from Microsoft. For a complete listing
of the issues that are included in this update, see the associated Microsoft
Knowledge Base article. After you install this update, you may have to restart
your system.",
                "ContentUrl": "https://support.microsoft.com/en-us/kb/4483452",
                "Vendor": "Microsoft",
                "ProductFamily": "Windows",
                "Product": "WindowsServer2019",
                "Classification": "SecurityUpdates",
                "MsrcSeverity": "Important",
                "KbNumber": "KB4483452",
                "MsrcNumber": "",
                "Language": "All"
            },
            "PatchStatus": {
                "DeploymentStatus": "APPROVED",
                "ComplianceLevel": "CRITICAL",
                "ApprovalDate": 1549994400.0
            }
        },
        ...
    ],
    "NextToken": "--token string truncated--"
```

```
}
```

示例 2：获取由 AWS 托管式补丁基准定义的所有补丁

以下 describe-effective-patches-for-patch-baseline 示例返回由 AWS 托管式补丁基准定义的补丁。请注意，对于 AWS 托管式基准，--baseline-id 需要完整的基准 ARN

```
aws ssm describe-effective-patches-for-patch-baseline \
    --baseline-id "arn:aws:ssm:us-east-2:733109147000:patchbaseline/
pb-020d361a05defe4ed"
```

有关输出示例，请参阅示例 1。

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的如何选择安全补丁。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeEffectivePatchesForPatchBaseline。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出所有补丁基准，最大结果列表为 1。

```
Get-SSMEffectivePatchesForPatchBaseline -BaselineId "pb-0a2f1059b670ebd31" -
MaxResult 1
```

输出：

```
Patch                                       PatchStatus
-----                                       -----------
Amazon.SimpleSystemsManagement.Model.Patch
 Amazon.SimpleSystemsManagement.Model.PatchStatus
```

示例 2：此示例显示所有补丁基准的补丁状态，最大结果列表为 1。

```
(Get-SSMEffectivePatchesForPatchBaseline -BaselineId "pb-0a2f1059b670ebd31" -
MaxResult 1).PatchStatus
```

输出：

```
ApprovalDate          DeploymentStatus
------------          ----------------
12/21/2010 6:00:00 PM APPROVED
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeEffectivePatchesForPatchBaseline。

Tools for PowerShell V5

示例 1：此示例列出所有补丁基准，最大结果列表为 1。

```
Get-SSMEffectivePatchesForPatchBaseline -BaselineId "pb-0a2f1059b670ebd31" -
MaxResult 1
```

输出：

```
Patch                                       PatchStatus
-----                                       -----------
Amazon.SimpleSystemsManagement.Model.Patch
 Amazon.SimpleSystemsManagement.Model.PatchStatus
```

示例 2：此示例显示所有补丁基准的补丁状态，最大结果列表为 1。

```
(Get-SSMEffectivePatchesForPatchBaseline -BaselineId "pb-0a2f1059b670ebd31" -
MaxResult 1).PatchStatus
```

输出：

```
ApprovalDate          DeploymentStatus
------------          ----------------
12/21/2010 6:00:00 PM APPROVED
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeEffectivePatchesForPatchBaseline。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeInstanceAssociationsStatus** 与 CLI 配合使用

以下代码示例演示如何使用 DescribeInstanceAssociationsStatus。

CLI

AWS CLI

描述实例关联的状态

此示例显示实例关联的详细信息。

命令:

```
aws ssm describe-instance-associations-status --instance-id "i-1234567890abcdef0"
```

输出：

```
{
  "InstanceAssociationStatusInfos": [
      {
          "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
          "Name": "AWS-GatherSoftwareInventory",
          "DocumentVersion": "1",
          "AssociationVersion": "1",
          "InstanceId": "i-1234567890abcdef0",
          "ExecutionDate": 1550501886.0,
          "Status": "Success",
          "ExecutionSummary": "1 out of 1 plugin processed, 1 success, 0 failed,
 0 timedout, 0 skipped. ",
          "AssociationName": "Inventory-Association"
      },
      {
          "AssociationId": "5c5a31f6-6dae-46f9-944c-0123456789ab",
          "Name": "AWS-UpdateSSMAgent",
          "DocumentVersion": "1",
          "AssociationVersion": "1",
          "InstanceId": "i-1234567890abcdef0",
          "ExecutionDate": 1550505828.548,
          "Status": "Success",
          "DetailedStatus": "Success",
          "AssociationName": "UpdateSSMAgent"
      }
  ]
}
```

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeInstanceAssociationsStatus。

PowerShell

Tools for PowerShell V4

示例 1：此示例显示实例关联的详细信息。

```
Get-SSMInstanceAssociationsStatus -InstanceId "i-0000293ffd8c57862"
```

输出：

```
AssociationId    : d8617c07-2079-4c18-9847-1655fc2698b0
DetailedStatus   : Pending
DocumentVersion  : 1
ErrorCode        :
ExecutionDate    : 2/20/2015 8:31:11 AM
ExecutionSummary : temp_status_change
InstanceId       : i-0000293ffd8c57862
Name             : AWS-UpdateSSMAgent
OutputUrl        :
Status           : Pending
```

示例 2：此示例检查给定实例 ID 的实例关联状态，并进一步显示这些关联的执行状态

```
Get-SSMInstanceAssociationsStatus -InstanceId i-012e3cb4df567e8aa | ForEach-
Object {Get-SSMAssociationExecution -AssociationId .AssociationId}
```

输出：

```
AssociationId         : 512a34a5-c678-1234-1234-12345678db9e
AssociationVersion    : 2
CreatedTime           : 3/2/2019 8:53:29 AM
DetailedStatus        :
ExecutionId           : 512a34a5-c678-1234-1234-12345678db9e
LastExecutionDate     : 1/1/0001 12:00:00 AM
ResourceCountByStatus : {Success=9}
Status                : Success
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeInstanceAssociationsStatus。

Tools for PowerShell V5

示例 1：此示例显示实例关联的详细信息。

```
Get-SSMInstanceAssociationsStatus -InstanceId "i-0000293ffd8c57862"
```

输出：

```
AssociationId    : d8617c07-2079-4c18-9847-1655fc2698b0
DetailedStatus   : Pending
DocumentVersion  : 1
ErrorCode        :
ExecutionDate    : 2/20/2015 8:31:11 AM
ExecutionSummary : temp_status_change
InstanceId       : i-0000293ffd8c57862
Name             : AWS-UpdateSSMAgent
OutputUrl        :
Status           : Pending
```

示例 2：此示例检查给定实例 ID 的实例关联状态，并进一步显示这些关联的执行状态

```
Get-SSMInstanceAssociationsStatus -InstanceId i-012e3cb4df567e8aa | ForEach-
Object {Get-SSMAssociationExecution -AssociationId .AssociationId}
```

输出：

```
AssociationId         : 512a34a5-c678-1234-1234-12345678db9e
AssociationVersion    : 2
CreatedTime           : 3/2/2019 8:53:29 AM
DetailedStatus        :
ExecutionId           : 512a34a5-c678-1234-1234-12345678db9e
LastExecutionDate     : 1/1/0001 12:00:00 AM
ResourceCountByStatus : {Success=9}
Status                : Success
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeInstanceAssociationsStatus。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeInstanceInformation** 与 CLI 配合使用

以下代码示例演示如何使用 DescribeInstanceInformation。

CLI

AWS CLI

示例 1：描述托管式实例信息

以下 describe-instance-information 示例检索每个托管式实例的详细信息。

```
aws ssm describe-instance-information
```

示例 2：描述有关特定托管式实例的信息

以下 describe-instance-information 示例显示托管式实例 i-028ea792daEXAMPLE 的详细信息。

```
aws ssm describe-instance-information \
    --filters "Key=InstanceIds,Values=i-028ea792daEXAMPLE"
```

示例 3：描述有关具有特定标签键的托管式实例的信息

以下 describe-instance-information 示例显示具有标签键 DEV 的托管式实例的详细信息。

```
aws ssm describe-instance-information \
    --filters "Key=tag-key,Values=DEV"
```

输出：

```
{
    "InstanceInformationList": [
        {
            "InstanceId": "i-028ea792daEXAMPLE",
            "PingStatus": "Online",
            "LastPingDateTime": 1582221233.421,
            "AgentVersion": "2.3.842.0",
```

```
            "IsLatestVersion": true,
            "PlatformType": "Linux",
            "PlatformName": "SLES",
            "PlatformVersion": "15.1",
            "ResourceType": "EC2Instance",
            "IPAddress": "192.0.2.0",
            "ComputerName": "ip-198.51.100.0.us-east-2.compute.internal",
            "AssociationStatus": "Success",
            "LastAssociationExecutionDate": 1582220806.0,
            "LastSuccessfulAssociationExecutionDate": 1582220806.0,
            "AssociationOverview": {
                "DetailedStatus": "Success",
                "InstanceAssociationStatusAggregatedCount": {
                    "Success": 2
                }
            }
        }
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的托管式实例。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeInstanceInformation。

PowerShell

Tools for PowerShell V4

示例 1：此示例显示每个实例的详细信息。

```
Get-SSMInstanceInformation
```

输出：

```
ActivationId                         :
AgentVersion                         : 2.0.672.0
AssociationOverview                  :
 Amazon.SimpleSystemsManagement.Model.InstanceAggregatedAssociationOverview
AssociationStatus                    : Success
ComputerName                         : ip-172-31-44-222.us-
west-2.compute.internal
IamRole                              :
```

```
InstanceId                              : i-0cb2b964d3e14fd9f
IPAddress                               : 172.31.44.222
IsLatestVersion                         : True
LastAssociationExecutionDate            : 2/24/2017 3:18:09 AM
LastPingDateTime                        : 2/24/2017 3:35:03 AM
LastSuccessfulAssociationExecutionDate : 2/24/2017 3:18:09 AM
Name                                    :
PingStatus                              : ConnectionLost
PlatformName                            : Amazon Linux AMI
PlatformType                            : Linux
PlatformVersion                         : 2016.09
RegistrationDate                        : 1/1/0001 12:00:00 AM
ResourceType                            : EC2Instance
```

示例 2：此示例展示如何使用 -Filter 参数将结果筛选为仅显示区域 **us-east-1** 中 **AgentVersion** 为 **2.2.800.0** 的 AWS Systems Manager 实例。您可以在 InstanceInformation API 参考主题（https://docs.aws.amazon.com/systems-manager/latest/APIReference/API_InstanceInformation.html#systemsmanager-Type-InstanceInformation-ActivationId）中找到有效的 -Filter 键值列表。

```
$Filters = @{
    Key="AgentVersion"
    Values="2.2.800.0"
}
Get-SSMInstanceInformation -Region us-east-1 -Filter $Filters
```

输出：

```
ActivationId                            :
AgentVersion                            : 2.2.800.0
AssociationOverview                     :
 Amazon.SimpleSystemsManagement.Model.InstanceAggregatedAssociationOverview
AssociationStatus                       : Success
ComputerName                            : EXAMPLE-EXAMPLE.WORKGROUP
IamRole                                 :
InstanceId                              : i-EXAMPLEb0792d98ce
IPAddress                               : 10.0.0.01
IsLatestVersion                         : False
LastAssociationExecutionDate            : 8/16/2018 12:02:50 AM
LastPingDateTime                        : 8/16/2018 7:40:27 PM
LastSuccessfulAssociationExecutionDate : 8/16/2018 12:02:50 AM
Name                                    :
```

```
PingStatus                                : Online
PlatformName                              : Microsoft Windows Server 2016 Datacenter
PlatformType                              : Windows
PlatformVersion                           : 10.0.14393
RegistrationDate                          : 1/1/0001 12:00:00 AM
ResourceType                              : EC2Instance

ActivationId                              :
AgentVersion                              : 2.2.800.0
AssociationOverview                       :
 Amazon.SimpleSystemsManagement.Model.InstanceAggregatedAssociationOverview
AssociationStatus                         : Success
ComputerName                              : EXAMPLE-EXAMPLE.WORKGROUP
IamRole                                   :
InstanceId                                : i-EXAMPLEac7501d023
IPAddress                                 : 10.0.0.02
IsLatestVersion                           : False
LastAssociationExecutionDate              : 8/16/2018 12:00:20 AM
LastPingDateTime                          : 8/16/2018 7:40:35 PM
LastSuccessfulAssociationExecutionDate    : 8/16/2018 12:00:20 AM
Name                                      :
PingStatus                                : Online
PlatformName                              : Microsoft Windows Server 2016 Datacenter
PlatformType                              : Windows
PlatformVersion                           : 10.0.14393
RegistrationDate                          : 1/1/0001 12:00:00 AM
ResourceType                              : EC2Instance
```

示例 3：此示例展示如何使用 -InstanceInformationFilterList 参数将结果筛选为仅显示区域 **us-east-1** 中 **PlatformTypes** 为 **Windows** 或 **Linux** 的 AWS Systems Manager 实例。您可以在 InstanceInformationFilter API 参考主题（https://docs.aws.amazon.com/systems-manager/latest/APIReference/API_InstanceInformationFilter.html）中找到有效的 -InstanceInformationFilterList 键值列表。

```
$Filters = @{
   Key="PlatformTypes"
   ValueSet=("Windows","Linux")
}
Get-SSMInstanceInformation -Region us-east-1 -InstanceInformationFilterList
 $Filters
```

输出：

```
ActivationId                            :
AgentVersion                            : 2.2.800.0
AssociationOverview                     :
 Amazon.SimpleSystemsManagement.Model.InstanceAggregatedAssociationOverview
AssociationStatus                       : Success
ComputerName                            : EXAMPLE-EXAMPLE.WORKGROUP
IamRole                                 :
InstanceId                              : i-EXAMPLEb0792d98ce
IPAddress                               : 10.0.0.27
IsLatestVersion                         : False
LastAssociationExecutionDate            : 8/16/2018 12:02:50 AM
LastPingDateTime                        : 8/16/2018 7:40:27 PM
LastSuccessfulAssociationExecutionDate : 8/16/2018 12:02:50 AM
Name                                    :
PingStatus                              : Online
PlatformName                            : Ubuntu Server 18.04 LTS
PlatformType                            : Linux
PlatformVersion                         : 18.04
RegistrationDate                        : 1/1/0001 12:00:00 AM
ResourceType                            : EC2Instance

ActivationId                            :
AgentVersion                            : 2.2.800.0
AssociationOverview                     :
 Amazon.SimpleSystemsManagement.Model.InstanceAggregatedAssociationOverview
AssociationStatus                       : Success
ComputerName                            : EXAMPLE-EXAMPLE.WORKGROUP
IamRole                                 :
InstanceId                              : i-EXAMPLEac7501d023
IPAddress                               : 10.0.0.100
IsLatestVersion                         : False
LastAssociationExecutionDate            : 8/16/2018 12:00:20 AM
LastPingDateTime                        : 8/16/2018 7:40:35 PM
LastSuccessfulAssociationExecutionDate : 8/16/2018 12:00:20 AM
Name                                    :
PingStatus                              : Online
PlatformName                            : Microsoft Windows Server 2016 Datacenter
PlatformType                            : Windows
PlatformVersion                         : 10.0.14393
RegistrationDate                        : 1/1/0001 12:00:00 AM
ResourceType                            : EC2Instance
```

示例 4：此示例列出 ssm 托管式实例，并将 InstanceId、PingStatus、LastPingDateTime 和 PlatformName 导出到 csv 文件中。

```
Get-SSMInstanceInformation | Select-Object InstanceId, PingStatus,
 LastPingDateTime, PlatformName | Export-Csv Instance-details.csv -
NoTypeInformation
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeInstanceInformation。

Tools for PowerShell V5

示例 1：此示例显示每个实例的详细信息。

```
Get-SSMInstanceInformation
```

输出：

```
ActivationId                          :
AgentVersion                          : 2.0.672.0
AssociationOverview                   :
 Amazon.SimpleSystemsManagement.Model.InstanceAggregatedAssociationOverview
AssociationStatus                     : Success
ComputerName                          : ip-172-31-44-222.us-
west-2.compute.internal
IamRole                               :
InstanceId                            : i-0cb2b964d3e14fd9f
IPAddress                             : 172.31.44.222
IsLatestVersion                       : True
LastAssociationExecutionDate          : 2/24/2017 3:18:09 AM
LastPingDateTime                      : 2/24/2017 3:35:03 AM
LastSuccessfulAssociationExecutionDate : 2/24/2017 3:18:09 AM
Name                                  :
PingStatus                            : ConnectionLost
PlatformName                          : Amazon Linux AMI
PlatformType                          : Linux
PlatformVersion                       : 2016.09
RegistrationDate                      : 1/1/0001 12:00:00 AM
ResourceType                          : EC2Instance
```

示例 2：此示例展示如何使用 -Filter 参数将结果筛选为仅显示区域 **us-east-1** 中 **AgentVersion** 为 **2.2.800.0** 的 AWS Systems Manager 实例。您可以在

InstanceInformation API 参考主题（https://docs.aws.amazon.com/systems-manager/latest/APIReference/API_InstanceInformation.html#systemsmanager-Type-InstanceInformation-ActivationId）中找到有效的 -Filter 键值列表。

```
$Filters = @{
    Key="AgentVersion"
    Values="2.2.800.0"
}
Get-SSMInstanceInformation -Region us-east-1 -Filter $Filters
```

输出：

```
ActivationId                          :
AgentVersion                          : 2.2.800.0
AssociationOverview                   :
 Amazon.SimpleSystemsManagement.Model.InstanceAggregatedAssociationOverview
AssociationStatus                     : Success
ComputerName                          : EXAMPLE-EXAMPLE.WORKGROUP
IamRole                               :
InstanceId                            : i-EXAMPLEb0792d98ce
IPAddress                             : 10.0.0.01
IsLatestVersion                       : False
LastAssociationExecutionDate          : 8/16/2018 12:02:50 AM
LastPingDateTime                      : 8/16/2018 7:40:27 PM
LastSuccessfulAssociationExecutionDate : 8/16/2018 12:02:50 AM
Name                                  :
PingStatus                            : Online
PlatformName                          : Microsoft Windows Server 2016 Datacenter
PlatformType                          : Windows
PlatformVersion                       : 10.0.14393
RegistrationDate                      : 1/1/0001 12:00:00 AM
ResourceType                          : EC2Instance

ActivationId                          :
AgentVersion                          : 2.2.800.0
AssociationOverview                   :
 Amazon.SimpleSystemsManagement.Model.InstanceAggregatedAssociationOverview
AssociationStatus                     : Success
ComputerName                          : EXAMPLE-EXAMPLE.WORKGROUP
IamRole                               :
InstanceId                            : i-EXAMPLEac7501d023
IPAddress                             : 10.0.0.02
IsLatestVersion                       : False
```

```
LastAssociationExecutionDate           : 8/16/2018 12:00:20 AM
LastPingDateTime                       : 8/16/2018 7:40:35 PM
LastSuccessfulAssociationExecutionDate : 8/16/2018 12:00:20 AM
Name                                   :
PingStatus                             : Online
PlatformName                           : Microsoft Windows Server 2016 Datacenter
PlatformType                           : Windows
PlatformVersion                        : 10.0.14393
RegistrationDate                       : 1/1/0001 12:00:00 AM
ResourceType                           : EC2Instance
```

示例 3：此示例展示如何使用 -InstanceInformationFilterList 参数将结果筛选为仅显示区域 **us-east-1** 中 **PlatformTypes** 为 **Windows** 或 **Linux** 的 AWS Systems Manager 实例。您可以在 InstanceInformationFilter API 参考主题（https://docs.aws.amazon.com/systems-manager/latest/APIReference/API_InstanceInformationFilter.html）中找到有效的 -InstanceInformationFilterList 键值列表。

```
$Filters = @{
   Key="PlatformTypes"
   ValueSet=("Windows","Linux")
}
Get-SSMInstanceInformation -Region us-east-1 -InstanceInformationFilterList
 $Filters
```

输出：

```
ActivationId                           :
AgentVersion                           : 2.2.800.0
AssociationOverview                    :
 Amazon.SimpleSystemsManagement.Model.InstanceAggregatedAssociationOverview
AssociationStatus                      : Success
ComputerName                           : EXAMPLE-EXAMPLE.WORKGROUP
IamRole                                :
InstanceId                             : i-EXAMPLEb0792d98ce
IPAddress                              : 10.0.0.27
IsLatestVersion                        : False
LastAssociationExecutionDate           : 8/16/2018 12:02:50 AM
LastPingDateTime                       : 8/16/2018 7:40:27 PM
LastSuccessfulAssociationExecutionDate : 8/16/2018 12:02:50 AM
Name                                   :
PingStatus                             : Online
PlatformName                           : Ubuntu Server 18.04 LTS
```

```
PlatformType                          : Linux
PlatformVersion                       : 18.04
RegistrationDate                      : 1/1/0001 12:00:00 AM
ResourceType                          : EC2Instance

ActivationId                          :
AgentVersion                          : 2.2.800.0
AssociationOverview                   :
 Amazon.SimpleSystemsManagement.Model.InstanceAggregatedAssociationOverview
AssociationStatus                     : Success
ComputerName                          : EXAMPLE-EXAMPLE.WORKGROUP
IamRole                               :
InstanceId                            : i-EXAMPLEac7501d023
IPAddress                             : 10.0.0.100
IsLatestVersion                       : False
LastAssociationExecutionDate          : 8/16/2018 12:00:20 AM
LastPingDateTime                      : 8/16/2018 7:40:35 PM
LastSuccessfulAssociationExecutionDate : 8/16/2018 12:00:20 AM
Name                                  :
PingStatus                            : Online
PlatformName                          : Microsoft Windows Server 2016 Datacenter
PlatformType                          : Windows
PlatformVersion                       : 10.0.14393
RegistrationDate                      : 1/1/0001 12:00:00 AM
ResourceType                          : EC2Instance
```

示例 4：此示例列出 ssm 托管式实例，并将 InstanceId、PingStatus、LastPingDateTime 和 PlatformName 导出到 csv 文件中。

```
Get-SSMInstanceInformation | Select-Object InstanceId, PingStatus,
 LastPingDateTime, PlatformName | Export-Csv Instance-details.csv -
NoTypeInformation
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeInstanceInformation。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeInstancePatchStates** 与 CLI 配合使用

以下代码示例演示如何使用 DescribeInstancePatchStates。

CLI

AWS CLI

获取实例的补丁摘要状态

此 describe-instance-patch-states 示例获取实例的补丁摘要状态。

```
aws ssm describe-instance-patch-states \
    --instance-ids "i-1234567890abcdef0"
```

输出：

```
{
    "InstancePatchStates": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "PatchGroup": "my-patch-group",
            "BaselineId": "pb-0713accee01234567",
            "SnapshotId": "521c3536-930c-4aa9-950e-01234567abcd",
            "CriticalNonCompliantCount": 2,
            "SecurityNonCompliantCount": 2,
            "OtherNonCompliantCount": 1,
            "InstalledCount": 123,
            "InstalledOtherCount": 334,
            "InstalledPendingRebootCount": 0,
            "InstalledRejectedCount": 0,
            "MissingCount": 1,
            "FailedCount": 2,
            "UnreportedNotApplicableCount": 11,
            "NotApplicableCount": 2063,
            "OperationStartTime": "2021-05-03T11:00:56-07:00",
            "OperationEndTime": "2021-05-03T11:01:09-07:00",
            "Operation": "Scan",
            "LastNoRebootInstallOperationTime": "2020-06-14T12:17:41-07:00",
            "RebootOption": "RebootIfNeeded"
        }
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的关于补丁合规性。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeInstancePatchStates。

PowerShell

Tools for PowerShell V4

示例 1：此示例获取实例的补丁摘要状态。

```
Get-SSMInstancePatchState -InstanceId "i-08ee91c0b17045407"
```

示例 2：此示例获取两个实例的补丁摘要状态。

```
Get-SSMInstancePatchState -InstanceId "i-08ee91c0b17045407","i-09a618aec652973a9"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeInstancePatchStates。

Tools for PowerShell V5

示例 1：此示例获取实例的补丁摘要状态。

```
Get-SSMInstancePatchState -InstanceId "i-08ee91c0b17045407"
```

示例 2：此示例获取两个实例的补丁摘要状态。

```
Get-SSMInstancePatchState -InstanceId "i-08ee91c0b17045407","i-09a618aec652973a9"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeInstancePatchStates。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeInstancePatchStatesForPatchGroup** 与 CLI 配合使用

以下代码示例演示如何使用 DescribeInstancePatchStatesForPatchGroup。

CLI

AWS CLI

示例 1：获取补丁组的实例状态

以下 `describe-instance-patch-states-for-patch-group` 示例检索有关指定补丁组每个实例的补丁摘要状态的详细信息。

```
aws ssm describe-instance-patch-states-for-patch-group \
    --patch-group "Production"
```

输出：

```
{
    "InstancePatchStates": [
        {
            "InstanceId": "i-02573cafcfEXAMPLE",
            "PatchGroup": "Production",
            "BaselineId": "pb-0c10e65780EXAMPLE",
            "SnapshotId": "a3f5ff34-9bc4-4d2c-a665-4d1c1EXAMPLE",
            "OwnerInformation": "",
            "InstalledCount": 32,
            "InstalledOtherCount": 1,
            "InstalledPendingRebootCount": 0,
            "InstalledRejectedCount": 0,
            "MissingCount": 2,
            "FailedCount": 0,
            "UnreportedNotApplicableCount": 2671,
            "NotApplicableCount": 400,
            "OperationStartTime": "2021-08-04T11:03:50.590000-07:00",
            "OperationEndTime": "2021-08-04T11:04:21.555000-07:00",
            "Operation": "Scan",
            "RebootOption": "NoReboot",
            "CriticalNonCompliantCount": 0,
            "SecurityNonCompliantCount": 1,
            "OtherNonCompliantCount": 0
        },
        {
            "InstanceId": "i-0471e04240EXAMPLE",
            "PatchGroup": "Production",
            "BaselineId": "pb-09ca3fb51fEXAMPLE",
            "SnapshotId": "05d8ffb0-1bbe-4812-ba2d-d9b7bEXAMPLE",
```

```
            "OwnerInformation": "",
            "InstalledCount": 32,
            "InstalledOtherCount": 1,
            "InstalledPendingRebootCount": 0,
            "InstalledRejectedCount": 0,
            "MissingCount": 2,
            "FailedCount": 0,
            "UnreportedNotApplicableCount": 2671,
            "NotApplicableCount": 400,
            "OperationStartTime": "2021-08-04T22:06:20.340000-07:00",
            "OperationEndTime": "2021-08-04T22:07:11.220000-07:00",
            "Operation": "Scan",
            "RebootOption": "NoReboot",
            "CriticalNonCompliantCount": 0,
            "SecurityNonCompliantCount": 1,
            "OtherNonCompliantCount": 0
        }
    ]
}
```

示例 2：获取缺失五个补丁以上的补丁组的实例状态

以下 describe-instance-patch-states-for-patch-group 示例针对缺失五个补丁以上的实例的指定补丁组，检索补丁摘要状态详细信息。

```
aws ssm describe-instance-patch-states-for-patch-group \
    --filters Key=MissingCount,Type=GreaterThan,Values=5 \
    --patch-group "Production"
```

输出：

```
{
    "InstancePatchStates": [
        {
            "InstanceId": "i-02573cafcfEXAMPLE",
            "PatchGroup": "Production",
            "BaselineId": "pb-0c10e65780EXAMPLE",
            "SnapshotId": "a3f5ff34-9bc4-4d2c-a665-4d1c1EXAMPLE",
            "OwnerInformation": "",
            "InstalledCount": 46,
            "InstalledOtherCount": 4,
            "InstalledPendingRebootCount": 1,
            "InstalledRejectedCount": 1,
```

```
            "MissingCount": 7,
            "FailedCount": 0,
            "UnreportedNotApplicableCount": 232,
            "NotApplicableCount": 654,
            "OperationStartTime": "2021-08-04T11:03:50.590000-07:00",
            "OperationEndTime": "2021-08-04T11:04:21.555000-07:00",
            "Operation": "Scan",
            "RebootOption": "NoReboot",
            "CriticalNonCompliantCount": 0,
            "SecurityNonCompliantCount": 1,
            "OtherNonCompliantCount": 1
        }
    ]
}
```

示例 3：获取需要重启的实例少于 10 个的补丁组的实例状态

以下 describe-instance-patch-states-for-patch-group 示例针对需要重启的实例少于 10 个实例的指定补丁组，检索补丁摘要状态的详细信息。

```
aws ssm describe-instance-patch-states-for-patch-group \
    --filters Key=InstalledPendingRebootCount,Type=LessThan,Values=10 \
    --patch-group "Production"
```

输出：

```
{
    "InstancePatchStates": [
        {
            "InstanceId": "i-02573cafcfEXAMPLE",
            "BaselineId": "pb-0c10e65780EXAMPLE",
            "SnapshotId": "a3f5ff34-9bc4-4d2c-a665-4d1c1EXAMPLE",
            "PatchGroup": "Production",
            "OwnerInformation": "",
            "InstalledCount": 32,
            "InstalledOtherCount": 1,
            "InstalledPendingRebootCount": 4,
            "InstalledRejectedCount": 0,
            "MissingCount": 2,
            "FailedCount": 0,
            "UnreportedNotApplicableCount": 846,
            "NotApplicableCount": 212,
            "OperationStartTime": "2021-08-046T11:03:50.590000-07:00",
```

```
            "OperationEndTime": "2021-08-06T11:04:21.555000-07:00",
            "Operation": "Scan",
            "RebootOption": "NoReboot",
            "CriticalNonCompliantCount": 0,
            "SecurityNonCompliantCount": 1,
            "OtherNonCompliantCount": 0
        }
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的了解补丁合规性状态值。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeInstancePatchStatesForPatchGroup。

PowerShell

Tools for PowerShell V4

示例 1：此示例获取补丁组每个实例的补丁摘要状态。

```
Get-SSMInstancePatchStatesForPatchGroup -PatchGroup "Production"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeInstancePatchStatesForPatchGroup。

Tools for PowerShell V5

示例 1：此示例获取补丁组每个实例的补丁摘要状态。

```
Get-SSMInstancePatchStatesForPatchGroup -PatchGroup "Production"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeInstancePatchStatesForPatchGroup。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeInstancePatches** 与 CLI 配合使用

以下代码示例演示如何使用 DescribeInstancePatches。

CLI

AWS CLI

示例 1：获取实例的补丁状态详细信息

以下 describe-instance-patches 示例将检索有关指定实例补丁的详细信息。

```
aws ssm describe-instance-patches \
    --instance-id "i-1234567890abcdef0"
```

输出：

```
{
    "Patches": [
        {
            "Title": "2019-01 Security Update for Adobe Flash Player for Windows
 Server 2016 for x64-based Systems (KB4480979)",
            "KBId": "KB4480979",
            "Classification": "SecurityUpdates",
            "Severity": "Critical",
            "State": "Installed",
            "InstalledTime": "2019-01-09T00:00:00+00:00"
        },
        {
            "Title": "",
            "KBId": "KB4481031",
            "Classification": "",
            "Severity": "",
            "State": "InstalledOther",
            "InstalledTime": "2019-02-08T00:00:00+00:00"
        },
        ...
    ],
    "NextToken": "--token string truncated--"
}
```

示例 2：获取实例的处于“缺失”状态的补丁列表

以下 describe-instance-patches 示例检索有关指定实例处于“缺失”状态的补丁的信息。

```
aws ssm describe-instance-patches \
    --instance-id "i-1234567890abcdef0" \
```

```
    --filters Key=State,Values=Missing
```

输出：

```
{
    "Patches": [
        {
            "Title": "Windows Malicious Software Removal Tool x64 - February 2019
 (KB890830)",
            "KBId": "KB890830",
            "Classification": "UpdateRollups",
            "Severity": "Unspecified",
            "State": "Missing",
            "InstalledTime": "1970-01-01T00:00:00+00:00"
        },
        ...
    ],
    "NextToken": "--token string truncated--"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的关于补丁合规性状态。

示例 3：获取实例自指定 InstalledTime 以来所安装补丁的列表

以下 `describe-instance-patches` 示例通过组合使用 `--filters` 和 `--query`，检索指定实例自指定时间以来所安装补丁的信息。

```
aws ssm describe-instance-patches \
    --instance-id "i-1234567890abcdef0" \
    --filters Key=State,Values=Installed \
    --query "Patches[?InstalledTime >= `2023-01-01T16:00:00`]"
```

输出：

```
{
    "Patches": [
        {
            "Title": "2023-03 Cumulative Update for Windows Server 2019 (1809)
 for x64-based Systems (KB5023702)",
            "KBId": "KB5023702",
            "Classification": "SecurityUpdates",
            "Severity": "Critical",
            "State": "Installed",
```

```
            "InstalledTime": "2023-03-16T11:00:00+00:00"
        },
        ...
    ],
    "NextToken": "--token string truncated--"
}
```

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeInstancePatches。

PowerShell

Tools for PowerShell V4

示例 1：此示例获取实例的补丁合规性详细信息。

```
Get-SSMInstancePatch -InstanceId "i-08ee91c0b17045407"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeInstancePatches。

Tools for PowerShell V5

示例 1：此示例获取实例的补丁合规性详细信息。

```
Get-SSMInstancePatch -InstanceId "i-08ee91c0b17045407"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeInstancePatches。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeMaintenanceWindowExecutionTaskInvocations** 与 CLI 配合使用

以下代码示例演示如何使用 DescribeMaintenanceWindowExecutionTaskInvocations。

CLI

AWS CLI

获取为执行维护时段任务而执行的特定任务调用

以下 describe-maintenance-window-execution-task-invocations 示例列出作为指定维护时段执行组成部分来执行的指定任务的调用。

```
aws ssm describe-maintenance-window-execution-task-invocations \
    --window-execution-id "518d5565-5969-4cca-8f0e-da3b2a638355" \
    --task-id "ac0c6ae1-daa3-4a89-832e-d384503b6586"
```

输出：

```
{
    "WindowExecutionTaskInvocationIdentities": [
        {
            "Status": "SUCCESS",
            "Parameters": "{\"documentName\":\"AWS-RunShellScript\",
\"instanceIds\":[\"i-0000293ffd8c57862\"],\"parameters\":{\"commands\":[\"df\"]},
\"maxConcurrency\":\"1\",\"maxErrors\":\"1\"}",
            "InvocationId": "e274b6e1-fe56-4e32-bd2a-8073c6381d8b",
            "StartTime": 1487692834.723,
            "EndTime": 1487692834.871,
            "WindowExecutionId": "518d5565-5969-4cca-8f0e-da3b2a638355",
            "TaskExecutionId": "ac0c6ae1-daa3-4a89-832e-d384503b6586"
        }
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的查看有关任务和任务执行的信息（AWS CLI）。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeMaintenanceWindowExecutionTaskInvocations。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出作为维护时段执行组成部分来执行的任务的调用。

```
Get-SSMMaintenanceWindowExecutionTaskInvocationList -TaskId "ac0c6ae1-
daa3-4a89-832e-d384503b6586" -WindowExecutionId "518d5565-5969-4cca-8f0e-
da3b2a638355"
```

输出：

```
EndTime           : 2/21/2017 4:00:34 PM
ExecutionId       :
InvocationId      : e274b6e1-fe56-4e32-bd2a-8073c6381d8b
OwnerInformation  :
Parameters        : {"documentName":"AWS-RunShellScript","instanceIds":
["i-0000293ffd8c57862"],"parameters":{"commands":["df"]},"maxConcurrency":"1",
                    "maxErrors":"1"}
StartTime         : 2/21/2017 4:00:34 PM
Status            : FAILED
StatusDetails     : The instance IDs list contains an invalid entry.
TaskExecutionId   : ac0c6ae1-daa3-4a89-832e-d384503b6586
WindowExecutionId : 518d5565-5969-4cca-8f0e-da3b2a638355
WindowTargetId    :
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeMaintenanceWindowExecutionTaskInvocations。

Tools for PowerShell V5

示例 1：此示例列出作为维护时段执行组成部分来执行的任务的调用。

```
Get-SSMMaintenanceWindowExecutionTaskInvocationList -TaskId "ac0c6ae1-
daa3-4a89-832e-d384503b6586" -WindowExecutionId "518d5565-5969-4cca-8f0e-
da3b2a638355"
```

输出：

```
EndTime           : 2/21/2017 4:00:34 PM
ExecutionId       :
InvocationId      : e274b6e1-fe56-4e32-bd2a-8073c6381d8b
OwnerInformation  :
Parameters        : {"documentName":"AWS-RunShellScript","instanceIds":
["i-0000293ffd8c57862"],"parameters":{"commands":["df"]},"maxConcurrency":"1",
                    "maxErrors":"1"}
StartTime         : 2/21/2017 4:00:34 PM
Status            : FAILED
StatusDetails     : The instance IDs list contains an invalid entry.
TaskExecutionId   : ac0c6ae1-daa3-4a89-832e-d384503b6586
WindowExecutionId : 518d5565-5969-4cca-8f0e-da3b2a638355
WindowTargetId    :
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeMaintenanceWindowExecutionTaskInvocations。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeMaintenanceWindowExecutionTasks** 与 CLI 配合使用

以下代码示例演示如何使用 DescribeMaintenanceWindowExecutionTasks。

CLI

AWS CLI

列出与维护时段执行相关的所有任务

以下 ssm describe-maintenance-window-execution-tasks 示例列出与指定维护时段执行相关的任务。

```
aws ssm describe-maintenance-window-execution-tasks \
    --window-execution-id "518d5565-5969-4cca-8f0e-da3b2EXAMPLE"
```

输出：

```
{
    "WindowExecutionTaskIdentities": [
        {
            "Status": "SUCCESS",
            "TaskArn": "AWS-RunShellScript",
            "StartTime": 1487692834.684,
            "TaskType": "RUN_COMMAND",
            "EndTime": 1487692835.005,
            "WindowExecutionId": "518d5565-5969-4cca-8f0e-da3b2EXAMPLE",
            "TaskExecutionId": "ac0c6ae1-daa3-4a89-832e-d3845EXAMPLE"
        }
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的查看有关任务和任务执行的信息（AWS CLI）。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeMaintenanceWindowExecutionTasks。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出与维护时段执行相关的任务。

```
Get-SSMMaintenanceWindowExecutionTaskList -WindowExecutionId
 "518d5565-5969-4cca-8f0e-da3b2a638355"
```

输出：

```
EndTime           : 2/21/2017 4:00:35 PM
StartTime         : 2/21/2017 4:00:34 PM
Status            : SUCCESS
TaskArn           : AWS-RunShellScript
TaskExecutionId   : ac0c6ae1-daa3-4a89-832e-d384503b6586
TaskType          : RUN_COMMAND
WindowExecutionId : 518d5565-5969-4cca-8f0e-da3b2a638355
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeMaintenanceWindowExecutionTasks。

Tools for PowerShell V5

示例 1：此示例列出与维护时段执行相关的任务。

```
Get-SSMMaintenanceWindowExecutionTaskList -WindowExecutionId
 "518d5565-5969-4cca-8f0e-da3b2a638355"
```

输出：

```
EndTime           : 2/21/2017 4:00:35 PM
StartTime         : 2/21/2017 4:00:34 PM
Status            : SUCCESS
TaskArn           : AWS-RunShellScript
TaskExecutionId   : ac0c6ae1-daa3-4a89-832e-d384503b6586
TaskType          : RUN_COMMAND
WindowExecutionId : 518d5565-5969-4cca-8f0e-da3b2a638355
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeMaintenanceWindowExecutionTasks。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeMaintenanceWindowExecutions** 与 CLI 配合使用

以下代码示例演示如何使用 DescribeMaintenanceWindowExecutions。

CLI

AWS CLI

示例 1：列出维护时段内的所有执行

以下 describe-maintenance-window-executions 示例列出指定维护时段的所有执行。

```
aws ssm describe-maintenance-window-executions \
    --window-id "mw-ab12cd34eEXAMPLE"
```

输出：

```
{
    "WindowExecutions": [
        {
            "WindowId": "mw-ab12cd34eEXAMPLE",
            "WindowExecutionId": "6027b513-64fe-4cf0-be7d-1191aEXAMPLE",
            "Status": "IN_PROGRESS",
            "StartTime": "2021-08-04T11:00:00.000000-07:00"

        },
        {
            "WindowId": "mw-ab12cd34eEXAMPLE",
            "WindowExecutionId": "ff75b750-4834-4377-8f61-b3cadEXAMPLE",
            "Status": "SUCCESS",
            "StartTime": "2021-08-03T11:00:00.000000-07:00",
            "EndTime": "2021-08-03T11:37:21.450000-07:00"
        },
        {
            "WindowId": "mw-ab12cd34eEXAMPLE",
            "WindowExecutionId": "9fac7dd9-ff21-42a5-96ad-bbc4bEXAMPLE",
```

```
            "Status": "FAILED",
            "StatusDetails": "One or more tasks in the orchestration failed.",
            "StartTime": "2021-08-02T11:00:00.000000-07:00",
            "EndTime": "2021-08-02T11:22:36.190000-07:00"
        }
    ]
}
```

示例 2：列出指定日期之前维护时段内的所有执行

以下 describe-maintenance-window-executions 示例列出指定日期之前指定维护时段内的所有执行。

```
aws ssm describe-maintenance-window-executions \
    --window-id "mw-ab12cd34eEXAMPLE" \
    --filters "Key=ExecutedBefore,Values=2021-08-03T00:00:00Z"
```

输出：

```
{
    "WindowExecutions": [
        {
        "WindowId": "mw-ab12cd34eEXAMPLE",
        "WindowExecutionId": "9fac7dd9-ff21-42a5-96ad-bbc4bEXAMPLE",
        "Status": "FAILED",
        "StatusDetails": "One or more tasks in the orchestration failed.",
        "StartTime": "2021-08-02T11:00:00.000000-07:00",
        "EndTime": "2021-08-02T11:22:36.190000-07:00"
    }
    ]
}
```

示例 3：列出指定日期之后维护时段内的所有执行

以下 describe-maintenance-window-executions 示例列出指定日期之后指定维护时段内的所有执行。

```
aws ssm describe-maintenance-window-executions \
    --window-id "mw-ab12cd34eEXAMPLE" \
    --filters "Key=ExecutedAfter,Values=2021-08-04T00:00:00Z"
```

输出：

```
{
    "WindowExecutions": [
        {
        "WindowId": "mw-ab12cd34eEXAMPLE",
        "WindowExecutionId": "6027b513-64fe-4cf0-be7d-1191aEXAMPLE",
        "Status": "IN_PROGRESS",
        "StartTime": "2021-08-04T11:00:00.000000-07:00"
        }
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的查看有关任务和任务执行的信息（AWS CLI）。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeMaintenanceWindowExecutions。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出维护时段的所有执行。

```
Get-SSMMaintenanceWindowExecutionList -WindowId "mw-03eb9db42890fb82d"
```

输出：

```
EndTime           : 2/20/2017 6:30:17 PM
StartTime         : 2/20/2017 6:30:16 PM
Status            : FAILED
StatusDetails     : One or more tasks in the orchestration failed.
WindowExecutionId : 6f3215cf-4101-4fa0-9b7b-9523269599c7
WindowId          : mw-03eb9db42890fb82d
```

示例 2：此示例列出在指定日期之前的维护时段内的所有执行。

```
$option1 = @{Key="ExecutedBefore";Values=@("2016-11-04T05:00:00Z")}
Get-SSMMaintenanceWindowExecutionList -WindowId "mw-03eb9db42890fb82d" -Filter
 $option1
```

示例 3：此示例列出指定日期之后维护时段内的所有执行。

```
$option1 = @{Key="ExecutedAfter";Values=@("2016-11-04T05:00:00Z")}
Get-SSMMaintenanceWindowExecutionList -WindowId "mw-03eb9db42890fb82d" -Filter
 $option1
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeMaintenanceWindowExecutions。

Tools for PowerShell V5

示例 1：此示例列出维护时段的所有执行。

```
Get-SSMMaintenanceWindowExecutionList -WindowId "mw-03eb9db42890fb82d"
```

输出：

```
EndTime           : 2/20/2017 6:30:17 PM
StartTime         : 2/20/2017 6:30:16 PM
Status            : FAILED
StatusDetails     : One or more tasks in the orchestration failed.
WindowExecutionId : 6f3215cf-4101-4fa0-9b7b-9523269599c7
WindowId          : mw-03eb9db42890fb82d
```

示例 2：此示例列出在指定日期之前的维护时段内的所有执行。

```
$option1 = @{Key="ExecutedBefore";Values=@("2016-11-04T05:00:00Z")}
Get-SSMMaintenanceWindowExecutionList -WindowId "mw-03eb9db42890fb82d" -Filter
 $option1
```

示例 3：此示例列出指定日期之后维护时段内的所有执行。

```
$option1 = @{Key="ExecutedAfter";Values=@("2016-11-04T05:00:00Z")}
Get-SSMMaintenanceWindowExecutionList -WindowId "mw-03eb9db42890fb82d" -Filter
 $option1
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeMaintenanceWindowExecutions。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeMaintenanceWindowTargets** 与 CLI 配合使用

以下代码示例演示如何使用 DescribeMaintenanceWindowTargets。

CLI

AWS CLI

示例 1：列出维护时段内的所有目标

以下 describe-maintenance-window-targets 示例列出维护时段内的所有目标。

```
aws ssm describe-maintenance-window-targets \
    --window-id "mw-06cf17cbefEXAMPLE"
```

输出：

```
{
    "Targets": [
        {
            "ResourceType": "INSTANCE",
            "OwnerInformation": "Single instance",
            "WindowId": "mw-06cf17cbefEXAMPLE",
            "Targets": [
                {
                    "Values": [
                        "i-0000293ffdEXAMPLE"
                    ],
                    "Key": "InstanceIds"
                }
            ],
            "WindowTargetId": "350d44e6-28cc-44e2-951f-4b2c9EXAMPLE"
        },
        {
            "ResourceType": "INSTANCE",
            "OwnerInformation": "Two instances in a list",
            "WindowId": "mw-06cf17cbefEXAMPLE",
            "Targets": [
                {
                    "Values": [
                        "i-0000293ffdEXAMPLE",
                        "i-0cb2b964d3EXAMPLE"
                    ],
```

```
                    "Key": "InstanceIds"
                }
            ],
            "WindowTargetId": "e078a987-2866-47be-bedd-d9cf4EXAMPLE"
        }
    ]
}
```

示例 2：列出匹配特定所有者信息值的维护时段的所有目标

此 describe-maintenance-window-targets 示例列出具有特定值的维护时段的所有目标。

```
aws ssm describe-maintenance-window-targets \
    --window-id "mw-0ecb1226ddEXAMPLE" \
    --filters "Key=OwnerInformation,Values=CostCenter1"
```

输出：

```
{
    "Targets": [
        {
            "WindowId": "mw-0ecb1226ddEXAMPLE",
            "WindowTargetId": "da89dcc3-7f9c-481d-ba2b-edcb7d0057f9",
            "ResourceType": "INSTANCE",
            "Targets": [
                {
                    "Key": "tag:Environment",
                    "Values": [
                        "Prod"
                    ]
                }
            ],
            "OwnerInformation": "CostCenter1",
            "Name": "ProdTarget1"
        }
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的查看有关维护时段的信息（AWS CLI）。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeMaintenanceWindowTargets。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出维护时段的所有目标。

```
Get-SSMMaintenanceWindowTarget -WindowId "mw-06cf17cbefcb4bf4f"
```

输出：

```
OwnerInformation : Single instance
ResourceType     : INSTANCE
Targets          : {InstanceIds}
WindowId         : mw-06cf17cbefcb4bf4f
WindowTargetId   : 350d44e6-28cc-44e2-951f-4b2c985838f6

OwnerInformation : Two instances in a list
ResourceType     : INSTANCE
Targets          : {InstanceIds}
WindowId         : mw-06cf17cbefcb4bf4f
WindowTargetId   : e078a987-2866-47be-bedd-d9cf49177d3a
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeMaintenanceWindowTargets。

Tools for PowerShell V5

示例 1：此示例列出维护时段的所有目标。

```
Get-SSMMaintenanceWindowTarget -WindowId "mw-06cf17cbefcb4bf4f"
```

输出：

```
OwnerInformation : Single instance
ResourceType     : INSTANCE
Targets          : {InstanceIds}
WindowId         : mw-06cf17cbefcb4bf4f
WindowTargetId   : 350d44e6-28cc-44e2-951f-4b2c985838f6
```

```
OwnerInformation : Two instances in a list
ResourceType     : INSTANCE
Targets          : {InstanceIds}
WindowId         : mw-06cf17cbefcb4bf4f
WindowTargetId   : e078a987-2866-47be-bedd-d9cf49177d3a
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeMaintenanceWindowTargets。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeMaintenanceWindowTasks** 与 CLI 配合使用

以下代码示例演示如何使用 DescribeMaintenanceWindowTasks。

CLI

AWS CLI

示例 1：列出维护时段内的所有任务

以下 describe-maintenance-window-tasks 示例列出指定维护时段内的所有任务。

```
aws ssm describe-maintenance-window-tasks \
    --window-id "mw-06cf17cbefEXAMPLE"
```

输出：

```
{
    "Tasks": [
        {
            "WindowId": "mw-06cf17cbefEXAMPLE",
            "WindowTaskId": "018b31c3-2d77-4b9e-bd48-c91edEXAMPLE",
            "TaskArn": "AWS-RestartEC2Instance",
            "TaskParameters": {},
            "Type": "AUTOMATION",
            "Description": "Restarting EC2 Instance for maintenance",
            "MaxConcurrency": "1",
            "MaxErrors": "1",
            "Name": "My-Automation-Example-Task",
            "Priority": 0,
```

```
            "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
            "Targets": [
                {
                    "Key": "WindowTargetIds",
                    "Values": [
                        "da89dcc3-7f9c-481d-ba2b-edcb7EXAMPLE"
                    ]
                }
            ]
        },
        {
            "WindowId": "mw-06cf17cbefEXAMPLE",
            "WindowTaskId": "1943dee0-0a17-4978-9bf4-3cc2fEXAMPLE",
            "TaskArn": "AWS-DisableS3BucketPublicReadWrite",
            "TaskParameters": {},
            "Type": "AUTOMATION",
            "Description": "Automation task to disable read/write access on
 public S3 buckets",
            "MaxConcurrency": "10",
            "MaxErrors": "5",
            "Name": "My-Disable-S3-Public-Read-Write-Access-Automation-Task",
            "Priority": 0,
            "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
            "Targets": [
                {
                    "Key": "WindowTargetIds",
                    "Values": [
                        "da89dcc3-7f9c-481d-ba2b-edcb7EXAMPLE"
                    ]
                }
            ]
        }
    ]
}
```

示例 2：列出调用 AWS-RunPowerShellScript 命令文档的维护时段内的所有任务

以下 `describe-maintenance-window-tasks` 示例列出在调用 `AWS-RunPowerShellScript` 命令文档的指定维护时段内的所有任务。

```
aws ssm describe-maintenance-window-tasks \
```

```
    --window-id "mw-ab12cd34eEXAMPLE" \
    --filters "Key=TaskArn,Values=AWS-RunPowerShellScript"
```

输出：

```
{
    "Tasks": [
        {
            "WindowId": "mw-ab12cd34eEXAMPLE",
            "WindowTaskId": "0d36e6b4-3a4f-411e-adcb-3558eEXAMPLE",
            "TaskArn": "AWS-RunPowerShellScript",
            "Type": "RUN_COMMAND",
            "Targets": [
                {
                    "Key": "WindowTargetIds",
                    "Values": [
                        "da89dcc3-7f9c-481d-ba2b-edcb7EXAMPLE"
                    ]
                }
            ],
            "TaskParameters": {},
            "Priority": 1,
            "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
            "MaxConcurrency": "1",
            "MaxErrors": "1",
            "Name": "MyTask"
        }
    ]
}
```

示例 3：列出优先级为 3 的维护时段内的所有任务

以下 describe-maintenance-window-tasks 示例列出指定维护时段内 Priority 为 3 的所有任务。

```
aws ssm describe-maintenance-window-tasks \
    --window-id "mw-ab12cd34eEXAMPLE" \
    --filters "Key=Priority,Values=3"
```

输出：

```
{
    "Tasks": [
        {
            "WindowId": "mw-ab12cd34eEXAMPLE",
            "WindowTaskId": "0d36e6b4-3a4f-411e-adcb-3558eEXAMPLE",
            "TaskArn": "AWS-RunPowerShellScript",
            "Type": "RUN_COMMAND",
            "Targets": [
                {
                    "Key": "WindowTargetIds",
                    "Values": [
                        "da89dcc3-7f9c-481d-ba2b-edcb7EXAMPLE"
                    ]
                }
            ],
            "TaskParameters": {},
            "Priority": 3,
            "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
            "MaxConcurrency": "1",
            "MaxErrors": "1",
            "Name": "MyRunCommandTask"
        },
        {
            "WindowId": "mw-ab12cd34eEXAMPLE",
            "WindowTaskId": "ee45feff-ad65-4a6c-b478-5cab8EXAMPLE",
            "TaskArn": "AWS-RestartEC2Instance",
            "Type": "AUTOMATION",
            "Targets": [
                {
                    "Key": "WindowTargetIds",
                    "Values": [
                        "da89dcc3-7f9c-481d-ba2b-edcb7EXAMPLE"
                    ]
                }
            ],
            "TaskParameters": {},
            "Priority": 3,
            "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
            "MaxConcurrency": "10",
            "MaxErrors": "5",
            "Name": "My-Automation-Task",
```

```
            "Description": "A description for my Automation task"
        }
    ]
}
```

示例 4：列出优先级为 1 并使用 Run Command 的维护时段内的所有任务

此 describe-maintenance-window-tasks 示例列出指定维护时段内 Priority 为 1 并使用 Run Command 的所有任务。

```
aws ssm describe-maintenance-window-tasks \
    --window-id "mw-ab12cd34eEXAMPLE" \
    --filters "Key=Priority,Values=1" "Key=TaskType,Values=RUN_COMMAND"
```

输出：

```
{
    "Tasks": [
        {
            "WindowId": "mw-ab12cd34eEXAMPLE",
            "WindowTaskId": "0d36e6b4-3a4f-411e-adcb-3558eEXAMPLE",
            "TaskArn": "AWS-RunPowerShellScript",
            "Type": "RUN_COMMAND",
            "Targets": [
                {
                    "Key": "WindowTargetIds",
                    "Values": [
                        "da89dcc3-7f9c-481d-ba2b-edcb7EXAMPLE"
                    ]
                }
            ],
            "TaskParameters": {},
            "Priority": 1,
            "ServiceRoleArn": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
            "MaxConcurrency": "1",
            "MaxErrors": "1",
            "Name": "MyRunCommandTask"
        }
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的查看有关维护时段的信息（AWS CLI）。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeMaintenanceWindowTasks。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出维护时段的所有任务。

```
Get-SSMMaintenanceWindowTaskList -WindowId "mw-06cf17cbefcb4bf4f"
```

输出：

```
LoggingInfo    :
MaxConcurrency : 1
MaxErrors      : 1
Priority       : 10
ServiceRoleArn : arn:aws:iam::123456789012:role/MaintenanceWindowsRole
Targets        : {InstanceIds}
TaskArn        : AWS-RunShellScript
TaskParameters : {[commands,
 Amazon.SimpleSystemsManagement.Model.MaintenanceWindowTaskParameterValueExpression]}
Type           : RUN_COMMAND
WindowId       : mw-06cf17cbefcb4bf4f
WindowTaskId   : a23e338d-ff30-4398-8aa3-09cd052ebf17
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeMaintenanceWindowTasks。

Tools for PowerShell V5

示例 1：此示例列出维护时段的所有任务。

```
Get-SSMMaintenanceWindowTaskList -WindowId "mw-06cf17cbefcb4bf4f"
```

输出：

```
LoggingInfo    :
MaxConcurrency : 1
```

```
MaxErrors      : 1
Priority       : 10
ServiceRoleArn : arn:aws:iam::123456789012:role/MaintenanceWindowsRole
Targets        : {InstanceIds}
TaskArn        : AWS-RunShellScript
TaskParameters : {[commands,
 Amazon.SimpleSystemsManagement.Model.MaintenanceWindowTaskParameterValueExpression]}
Type           : RUN_COMMAND
WindowId       : mw-06cf17cbefcb4bf4f
WindowTaskId   : a23e338d-ff30-4398-8aa3-09cd052ebf17
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeMaintenanceWindowTasks。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeMaintenanceWindows** 与 CLI 配合使用

以下代码示例演示如何使用 DescribeMaintenanceWindows。

CLI

AWS CLI

示例 1：列出所有维护时段

以下 describe-maintenance-windows 示例列出当前区域中您 AWS 账户的所有维护时段。

```
aws ssm describe-maintenance-windows
```

输出：

```
{
    "WindowIdentities": [
        {
            "WindowId": "mw-0ecb1226ddEXAMPLE",
            "Name": "MyMaintenanceWindow-1",
            "Enabled": true,
            "Duration": 2,
            "Cutoff": 1,
```

```
            "Schedule": "rate(180 minutes)",
            "NextExecutionTime": "2020-02-12T23:19:20.596Z"
        },
        {
            "WindowId": "mw-03eb9db428EXAMPLE",
            "Name": "MyMaintenanceWindow-2",
            "Enabled": true,
            "Duration": 3,
            "Cutoff": 1,
            "Schedule": "rate(7 days)",
            "NextExecutionTime": "2020-02-17T23:22:00.956Z"
        },
    ]
}
```

示例 2：列出所有已启用的维护时段

以下 describe-maintenance-windows 示例列出所有已启用的维护时段。

```
aws ssm describe-maintenance-windows \
    --filters "Key=Enabled,Values=true"
```

示例 3：列出与特定名称匹配的维护时段

此 describe-maintenance-windows 示例列出具有指定名称的所有维护时段。

```
aws ssm describe-maintenance-windows \
    --filters "Key=Name,Values=MyMaintenanceWindow"
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的查看有关维护时段的信息（AWS CLI）。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribeMaintenanceWindows。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出您账户中的所有维护时段。

```
Get-SSMMaintenanceWindowList
```

输出：

```
Cutoff   : 1
Duration : 4
Enabled  : True
Name     : My-First-Maintenance-Window
WindowId : mw-06d59c1a07c022145
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeMaintenanceWindows。

Tools for PowerShell V5

示例 1：此示例列出您账户中的所有维护时段。

```
Get-SSMMaintenanceWindowList
```

输出：

```
Cutoff   : 1
Duration : 4
Enabled  : True
Name     : My-First-Maintenance-Window
WindowId : mw-06d59c1a07c022145
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeMaintenanceWindows。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeOpsItems** 与 AWS SDK 或 CLI 配合使用

以下代码示例演示如何使用 DescribeOpsItems。

CLI

AWS CLI

列出一组 OpsItem

以下 describe-ops-items 示例显示您 AWS 账户中所有打开的 OpsItem 列表。

```
aws ssm describe-ops-items \
    --ops-item-filters "Key=Status,Values=Open,Operator=Equal"
```

输出：

```
{
    "OpsItemSummaries": [
        {
            "CreatedBy": "arn:aws:sts::111222333444:assumed-role/OpsItem-CWE-
Role/fbf77cbe264a33509569f23e4EXAMPLE",
            "CreatedTime": "2020-03-14T17:02:46.375000-07:00",
            "LastModifiedBy": "arn:aws:sts::111222333444:assumed-role/OpsItem-
CWE-Role/fbf77cbe264a33509569f23e4EXAMPLE",
            "LastModifiedTime": "2020-03-14T17:02:46.375000-07:00",
            "Source": "SSM",
            "Status": "Open",
            "OpsItemId": "oi-7cfc5EXAMPLE",
            "Title": "SSM Maintenance Window execution failed",
            "OperationalData": {
                "/aws/dedup": {
                    "Value": "{\"dedupString\":\"SSMOpsItems-SSM-maintenance-
window-execution-failed\"}",
                    "Type": "SearchableString"
                },
                "/aws/resources": {
                    "Value": "[{\"arn\":\"arn:aws:ssm:us-
east-2:111222333444:maintenancewindow/mw-034093d322EXAMPLE\"}]",
                    "Type": "SearchableString"
                }
            },
            "Category": "Availability",
            "Severity": "3"
        },
        {
            "CreatedBy": "arn:aws:sts::1112223233444:assumed-role/OpsItem-CWE-
Role/fbf77cbe264a33509569f23e4EXAMPLE",
            "CreatedTime": "2020-02-26T11:43:15.426000-08:00",
            "LastModifiedBy": "arn:aws:sts::111222333444:assumed-role/OpsItem-
CWE-Role/fbf77cbe264a33509569f23e4EXAMPLE",
            "LastModifiedTime": "2020-02-26T11:43:15.426000-08:00",
            "Source": "EC2",
            "Status": "Open",
            "OpsItemId": "oi-6f966EXAMPLE",
```

```
            "Title": "EC2 instance stopped",
            "OperationalData": {
                "/aws/automations": {
                    "Value": "[ { \"automationType\": \"AWS:SSM:Automation\",
 \"automationId\": \"AWS-RestartEC2Instance\" } ]",
                    "Type": "SearchableString"
                },
                "/aws/dedup": {
                    "Value": "{\"dedupString\":\"SSMOpsItems-EC2-instance-stopped
\"}",
                    "Type": "SearchableString"
                },
                "/aws/resources": {
                    "Value": "[{\"arn\":\"arn:aws:ec2:us-
east-2:111222333444:instance/i-0beccfbc02EXAMPLE\"}]",
                    "Type": "SearchableString"
                }
            },
            "Category": "Availability",
            "Severity": "3"
        }
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的使用 OpsItem。

- 有关 API 详细信息，请参阅《AWS CLI 命令参考》中的 DescribeOpsItems。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    /**
     * Describes AWS SSM OpsItems asynchronously.
     *
     * @param key The key to filter OpsItems by (e.g., OPS_ITEM_ID).
```

```
     *
     * This method initiates an asynchronous request to describe SSM OpsItems.
     * If the request is successful, it prints the title and status of each
OpsItem.
     * If an exception occurs, it handles the error appropriately.
     */
    public void describeOpsItems(String key) {
        OpsItemFilter filter = OpsItemFilter.builder()
                .key(OpsItemFilterKey.OPS_ITEM_ID)
                .values(key)
                .operator(OpsItemFilterOperator.EQUAL)
                .build();

        DescribeOpsItemsRequest itemsRequest = DescribeOpsItemsRequest.builder()
                .maxResults(10)
                .opsItemFilters(filter)
                .build();

        CompletableFuture<Void> future = CompletableFuture.runAsync(() -> {
            getAsyncClient().describeOpsItems(itemsRequest)
                    .thenAccept(itemsResponse -> {
                        List<OpsItemSummary> items =
itemsResponse.opsItemSummaries();
                        for (OpsItemSummary item : items) {
                            System.out.println("The item title is " +
item.title() + " and the status is " + item.status().toString());
                        }
                    })
                    .exceptionally(ex -> {
                        throw new CompletionException(ex);
                    }).join();
        }).exceptionally(ex -> {
            Throwable cause = (ex instanceof CompletionException) ?
ex.getCause() : ex;
            if (cause instanceof SsmException) {
                throw new RuntimeException("SSM error: " + cause.getMessage(),
cause);
            } else {
                throw new RuntimeException("Unexpected error: " +
cause.getMessage(), cause);
            }
        });

        try {
```

```
            future.join();
        } catch (CompletionException ex) {
            throw ex.getCause() instanceof RuntimeException ? (RuntimeException)
 ex.getCause() : ex;
        }
    }
```

- 有关 API 详细信息，请参阅《AWS SDK for Java 2.x API 参考》中的 DescribeOpsItems。

JavaScript

适用于 JavaScript 的 SDK（v3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import {
  OpsItemFilterOperator,
  OpsItemFilterKey,
  paginateDescribeOpsItems,
  SSMClient,
} from "@aws-sdk/client-ssm";
import { parseArgs } from "node:util";

/**
 * Describe SSM OpsItems.
 * @param {{ opsItemId: string }}
 */
export const main = async ({ opsItemId }) => {
  const client = new SSMClient({});
  try {
    const describeOpsItemsPaginated = [];
    for await (const page of paginateDescribeOpsItems(
      { client },
      {
        OpsItemFilters: {
          Key: OpsItemFilterKey.OPSITEM_ID,
          Operator: OpsItemFilterOperator.EQUAL,
```

```
          Values: opsItemId,
        },
      },
    )) {
      describeOpsItemsPaginated.push(...page.OpsItemSummaries);
    }
    console.log("Here are the ops items:");
    console.log(describeOpsItemsPaginated);
    return { OpsItemSummaries: describeOpsItemsPaginated };
  } catch (caught) {
    if (caught instanceof Error && caught.name === "MissingParameter") {
      console.warn(`${caught.message}. Did you provide this value?`);
    }
    throw caught;
  }
};
```

- 有关 API 详细信息，请参阅《适用于 JavaScript 的 AWS SDK API 参考》中的 DescribeOpsItems。

Python

适用于 Python 的 SDK (Boto3)

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
class OpsItemWrapper:
    """Encapsulates AWS Systems Manager OpsItem actions."""

    def __init__(self, ssm_client):
        """
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.id = None
```

```
    @classmethod
    def from_client(cls):
        """
        :return: A OpsItemWrapper instance.
        """
        ssm_client = boto3.client("ssm")
        return cls(ssm_client)


    def describe(self):
        """
        Describe an OpsItem.
        """
        try:
            paginator = self.ssm_client.get_paginator("describe_ops_items")
            ops_items = []
            for page in paginator.paginate(
                OpsItemFilters=[
                    {"Key": "OpsItemId", "Values": [self.id], "Operator":
"Equal"}
                ]
            ):
                ops_items.extend(page["OpsItemSummaries"])

            for item in ops_items:
                print(
                    f"The item title is {item['Title']} and the status is
{item['Status']}"
                )
            return len(ops_items) > 0
        except ClientError as err:
            logger.error(
                "Couldn't describe ops item %s. Here's why: %s: %s",
                self.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 有关 API 的详细信息，请参阅《AWS SDK for Python（Boto3）API 参考》中的 DescribeOpsItems。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribeParameters** 与 AWS SDK 或 CLI 配合使用

以下代码示例演示如何使用 DescribeParameters。

CLI

AWS CLI

示例 1：列出所有参数

以下 describe-parameters 示例列出了当前 AWS 账户和区域中的所有参数。

```
aws ssm describe-parameters
```

输出：

```
{
    "Parameters": [
        {
            "Name": "MySecureStringParameter",
            "Type": "SecureString",
            "KeyId": "alias/aws/ssm",
            "LastModifiedDate": 1582155479.205,
            "LastModifiedUser": "arn:aws:sts::111222333444:assumed-role/Admin/
Richard-Roe-Managed",
            "Description": "This is a SecureString parameter",
            "Version": 2,
            "Tier": "Advanced",
            "Policies": [
                {
                    "PolicyText": "{\"Type\":\"Expiration\",\"Version\":\"1.0\",
\"Attributes\":{\"Timestamp\":\"2020-07-07T22:30:00Z\"}}",
                    "PolicyType": "Expiration",
                    "PolicyStatus": "Pending"
                },
                {
                    "PolicyText": "{\"Type\":\"ExpirationNotification\",\"Version
\":\"1.0\",\"Attributes\":{\"Before\":\"12\",\"Unit\":\"Hours\"}}",
                    "PolicyType": "ExpirationNotification",
                    "PolicyStatus": "Pending"
```

```
                }
            ]
        },
        {
            "Name": "MyStringListParameter",
            "Type": "StringList",
            "LastModifiedDate": 1582154764.222,
            "LastModifiedUser": "arn:aws:iam::111222333444:user/Mary-Major",
            "Description": "This is a StringList parameter",
            "Version": 1,
            "Tier": "Standard",
            "Policies": []
        },
        {
            "Name": "MyStringParameter",
            "Type": "String",
            "LastModifiedDate": 1582154711.976,
            "LastModifiedUser": "arn:aws:iam::111222333444:user/Alejandro-
Rosalez",
            "Description": "This is a String parameter",
            "Version": 1,
            "Tier": "Standard",
            "Policies": []
        },
        {
            "Name": "latestAmi",
            "Type": "String",
            "LastModifiedDate": 1580862415.521,
            "LastModifiedUser": "arn:aws:sts::111222333444:assumed-role/lambda-
ssm-role/Automation-UpdateSSM-Param",
            "Version": 3,
            "Tier": "Standard",
            "Policies": []
        }
    ]
}
```

示例 2：列出与特定元数据匹配的所有参数

以下 `describe-parameters` 示例列出了与筛选器匹配的所有参数。

```
aws ssm describe-parameters --filters "Key=Type,Values=StringList"
```

输出：

```
{
    "Parameters": [
        {
            "Name": "MyStringListParameter",
            "Type": "StringList",
            "LastModifiedDate": 1582154764.222,
            "LastModifiedUser": "arn:aws:iam::111222333444:user/Mary-Major",
            "Description": "This is a StringList parameter",
            "Version": 1,
            "Tier": "Standard",
            "Policies": []
        }
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的搜索 Systems Manager 参数。

- 有关 API 详细信息，请参阅《AWS CLI 命令参考》中的 DescribeParameters。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.ssm.SsmClient;
import software.amazon.awssdk.services.ssm.model.GetParameterRequest;
import software.amazon.awssdk.services.ssm.model.GetParameterResponse;
import software.amazon.awssdk.services.ssm.model.SsmException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
```

```
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class GetParameter {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <paraName>

                Where:
                    paraName - The name of the parameter.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String paraName = args[0];
        Region region = Region.US_EAST_1;
        SsmClient ssmClient = SsmClient.builder()
                .region(region)
                .build();

        getParaValue(ssmClient, paraName);
        ssmClient.close();
    }

    public static void getParaValue(SsmClient ssmClient, String paraName) {
        try {
            GetParameterRequest parameterRequest = GetParameterRequest.builder()
                    .name(paraName)
                    .build();

            GetParameterResponse parameterResponse =
 ssmClient.getParameter(parameterRequest);
            System.out.println("The parameter value is " +
 parameterResponse.parameter().value());

        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
```

```
        }
    }
}
```

- 有关 API 详细信息，请参阅《AWS SDK for Java 2.x API 参考》中的 DescribeParameters。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出所有参数。

```
Get-SSMParameterList
```

输出：

```
Description      :
KeyId            :
LastModifiedDate : 3/3/2017 6:58:23 PM
LastModifiedUser : arn:aws:iam::123456789012:user/admin
Name             : Welcome
Type             : String
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribeParameters。

Tools for PowerShell V5

示例 1：此示例列出所有参数。

```
Get-SSMParameterList
```

输出：

```
Description      :
KeyId            :
LastModifiedDate : 3/3/2017 6:58:23 PM
LastModifiedUser : arn:aws:iam::123456789012:user/admin
Name             : Welcome
```

```
Type             : String
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribeParameters。

Rust

适用于 Rust 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
async fn show_parameters(client: &Client) -> Result<(), Error> {
    let resp = client.describe_parameters().send().await?;

    for param in resp.parameters() {
        println!("  {}", param.name().unwrap_or_default());
    }

    Ok(())
}
```

- 有关 API 详细信息，请参阅《AWS SDK for Rust API 参考》中的 DescribeParameters。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 `DescribePatchBaselines` 与 CLI 配合使用

以下代码示例演示如何使用 `DescribePatchBaselines`。

CLI

AWS CLI

示例 1：列出所有补丁基准

以下 describe-patch-baselines 示例检索您账户中当前区域所有补丁基准的详细信息。

```
aws ssm describe-patch-baselines
```

输出：

```
{
    "BaselineIdentities": [
        {
            "BaselineName": "AWS-SuseDefaultPatchBaseline",
            "DefaultBaseline": true,
            "BaselineDescription": "Default Patch Baseline for Suse Provided by
 AWS.",
            "BaselineId": "arn:aws:ssm:us-east-2:733109147000:patchbaseline/
pb-0123fdb36e334a3b2",
            "OperatingSystem": "SUSE"
        },
        {
            "BaselineName": "AWS-DefaultPatchBaseline",
            "DefaultBaseline": false,
            "BaselineDescription": "Default Patch Baseline Provided by AWS.",
            "BaselineId": "arn:aws:ssm:us-east-2:733109147000:patchbaseline/
pb-020d361a05defe4ed",
            "OperatingSystem": "WINDOWS"
        },
        ...
        {
            "BaselineName": "MyWindowsPatchBaseline",
            "DefaultBaseline": true,
            "BaselineDescription": "My patch baseline for EC2 instances for
 Windows Server",
            "BaselineId": "pb-0ad00e0dd7EXAMPLE",
            "OperatingSystem": "WINDOWS"
        }
    ]
}
```

示例 2：列出 AWS 提供的所有补丁基准

以下 describe-patch-baselines 示例列出 AWS 提供的所有补丁基准。

```
aws ssm describe-patch-baselines \
```

```
    --filters "Key=OWNER,Values=[AWS]"
```

示例 3：列出您拥有的所有补丁基准

以下 describe-patch-baselines 示例列出当前区域在您的账户中创建的所有自定义补丁基准。

```
aws ssm describe-patch-baselines \
    --filters "Key=OWNER,Values=[Self]"
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的关于预定义和自定义补丁基准。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribePatchBaselines。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出所有补丁基准。

```
Get-SSMPatchBaseline
```

输出：

```
BaselineDescription                                                BaselineId
                                                  BaselineName
------------------                                                ----------
                                                  ------------
Default Patch Baseline Provided by AWS.                           arn:aws:ssm:us-
west-2:123456789012:patchbaseline/pb-04fb4ae6142167966 AWS-DefaultP...
Baseline containing all updates approved for production systems
 pb-045f10b4f382baeda
 Production-B...
Baseline containing all updates approved for production systems
 pb-0a2f1059b670ebd31
 Production-B...
```

示例 2：此示例列出 AWS 提供的所有补丁基准。此示例使用的语法需要 PowerShell 版本 3 或更高版本。

```
$filter1 = @{Key="OWNER";Values=@("AWS")}
```

输出：

```
Get-SSMPatchBaseline -Filter $filter1
```

示例 3：此示例列出您作为所有者的所有补丁基准。此示例使用的语法需要 PowerShell 版本 3 或更高版本。

```
$filter1 = @{Key="OWNER";Values=@("Self")}
```

输出：

```
Get-SSMPatchBaseline -Filter $filter1
```

示例 4：对于 PowerShell 版本 2，必须使用 New-Object 创建每个标签。

```
$filter1 = New-Object
 Amazon.SimpleSystemsManagement.Model.PatchOrchestratorFilter
$filter1.Key = "OWNER"
$filter1.Values = "AWS"

Get-SSMPatchBaseline -Filter $filter1
```

输出：

```
BaselineDescription                     BaselineId
                            BaselineName            DefaultBaselin

                                                    e
------------------                      ----------
                            ------------            --------------
Default Patch Baseline Provided by AWS. arn:aws:ssm:us-
west-2:123456789012:patchbaseline/pb-04fb4ae6142167966 AWS-DefaultPatchBaseline
 True
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribePatchBaselines。

Tools for PowerShell V5

示例 1：此示例列出所有补丁基准。

```
Get-SSMPatchBaseline
```

输出：

```
BaselineDescription                                                BaselineId
                                                  BaselineName
------------------                                                ----------
                                                  ------------
Default Patch Baseline Provided by AWS.                           arn:aws:ssm:us-
west-2:123456789012:patchbaseline/pb-04fb4ae6142167966 AWS-DefaultP...
Baseline containing all updates approved for production systems
 pb-045f10b4f382baeda
 Production-B...
Baseline containing all updates approved for production systems
 pb-0a2f1059b670ebd31
 Production-B...
```

示例 2：此示例列出 AWS 提供的所有补丁基准。此示例使用的语法需要 PowerShell 版本 3 或更高版本。

```
$filter1 = @{Key="OWNER";Values=@("AWS")}
```

输出：

```
Get-SSMPatchBaseline -Filter $filter1
```

示例 3：此示例列出您作为所有者的所有补丁基准。此示例使用的语法需要 PowerShell 版本 3 或更高版本。

```
$filter1 = @{Key="OWNER";Values=@("Self")}
```

输出：

```
Get-SSMPatchBaseline -Filter $filter1
```

示例 4：对于 PowerShell 版本 2，必须使用 New-Object 创建每个标签。

```
$filter1 = New-Object
 Amazon.SimpleSystemsManagement.Model.PatchOrchestratorFilter
$filter1.Key = "OWNER"
$filter1.Values = "AWS"

Get-SSMPatchBaseline -Filter $filter1
```

输出：

```
BaselineDescription                     BaselineId
                            BaselineName             DefaultBaselin

                                                     e
-------------------                     ----------
                            ------------             --------------
Default Patch Baseline Provided by AWS. arn:aws:ssm:us-
west-2:123456789012:patchbaseline/pb-04fb4ae6142167966 AWS-DefaultPatchBaseline
 True
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribePatchBaselines。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribePatchGroupState** 与 CLI 配合使用

以下代码示例演示如何使用 DescribePatchGroupState。

CLI

AWS CLI

获取补丁组的状态

以下 describe-patch-group-state 示例检索补丁组的高级补丁合规性摘要。

```
aws ssm describe-patch-group-state \
```

```
    --patch-group "Production"
```

输出：

```
{
    "Instances": 21,
    "InstancesWithCriticalNonCompliantPatches": 1,
    "InstancesWithFailedPatches": 2,
    "InstancesWithInstalledOtherPatches": 3,
    "InstancesWithInstalledPatches": 21,
    "InstancesWithInstalledPendingRebootPatches": 2,
    "InstancesWithInstalledRejectedPatches": 1,
    "InstancesWithMissingPatches": 3,
    "InstancesWithNotApplicablePatches": 4,
    "InstancesWithOtherNonCompliantPatches": 1,
    "InstancesWithSecurityNonCompliantPatches": 1,
    "InstancesWithUnreportedNotApplicablePatches": 2
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的关于补丁组 <https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-patchgroups.html>__ 和了解补丁合规性状态值。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribePatchGroupState。

PowerShell

Tools for PowerShell V4

示例 1：此示例获取补丁组的高级补丁合规性摘要。

```
Get-SSMPatchGroupState -PatchGroup "Production"
```

输出：

```
Instances                       : 4
InstancesWithFailedPatches      : 1
InstancesWithInstalledOtherPatches : 4
InstancesWithInstalledPatches   : 3
InstancesWithMissingPatches     : 0
```

```
InstancesWithNotApplicablePatches  : 0
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribePatchGroupState。

Tools for PowerShell V5

示例 1：此示例获取补丁组的高级补丁合规性摘要。

```
Get-SSMPatchGroupState -PatchGroup "Production"
```

输出：

```
Instances                          : 4
InstancesWithFailedPatches         : 1
InstancesWithInstalledOtherPatches : 4
InstancesWithInstalledPatches      : 3
InstancesWithMissingPatches        : 0
InstancesWithNotApplicablePatches  : 0
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribePatchGroupState。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **DescribePatchGroups** 与 CLI 配合使用

以下代码示例演示如何使用 DescribePatchGroups。

CLI

AWS CLI

显示补丁组注册

以下 describe-patch-groups 示例列出补丁组注册。

```
aws ssm describe-patch-groups
```

输出：

```
{
    "Mappings": [
        {
            "PatchGroup": "Production",
            "BaselineIdentity": {
                "BaselineId": "pb-0123456789abcdef0",
                "BaselineName": "ProdPatching",
                "OperatingSystem": "WINDOWS",
                "BaselineDescription": "Patches for Production",
                "DefaultBaseline": false
            }
        },
        {
            "PatchGroup": "Development",
            "BaselineIdentity": {
                "BaselineId": "pb-0713accee01234567",
                "BaselineName": "DevPatching",
                "OperatingSystem": "WINDOWS",
                "BaselineDescription": "Patches for Development",
                "DefaultBaseline": true
            }
        },
        ...
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的创建补丁组 <https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-group-tagging.html>__ 和将补丁组添加到补丁基准。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 DescribePatchGroups。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出补丁组注册。

```
Get-SSMPatchGroup
```

输出：

```
BaselineIdentity                                                PatchGroup
----------------                                                ----------
Amazon.SimpleSystemsManagement.Model.PatchBaselineIdentity Production
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 DescribePatchGroups。

Tools for PowerShell V5

示例 1：此示例列出补丁组注册。

```
Get-SSMPatchGroup
```

输出：

```
BaselineIdentity                                                PatchGroup
----------------                                                ----------
Amazon.SimpleSystemsManagement.Model.PatchBaselineIdentity Production
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 DescribePatchGroups。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **GetAutomationExecution** 与 CLI 配合使用

以下代码示例演示如何使用 GetAutomationExecution。

CLI

AWS CLI

显示有关自动化执行的详细信息

以下 get-automation-execution 示例显示有关自动化执行的详细信息。

```
aws ssm get-automation-execution \
    --automation-execution-id 73c8eef8-f4ee-4a05-820c-e354fEXAMPLE
```

输出：


```
{
    "AutomationExecution": {
        "AutomationExecutionId": "73c8eef8-f4ee-4a05-820c-e354fEXAMPLE",
        "DocumentName": "AWS-StartEC2Instance",
        "DocumentVersion": "1",
        "ExecutionStartTime": 1583737233.748,
        "ExecutionEndTime": 1583737234.719,
        "AutomationExecutionStatus": "Success",
        "StepExecutions": [
            {
                "StepName": "startInstances",
                "Action": "aws:changeInstanceState",
                "ExecutionStartTime": 1583737234.134,
                "ExecutionEndTime": 1583737234.672,
                "StepStatus": "Success",
                "Inputs": {
                    "DesiredState": "\"running\"",
                    "InstanceIds": "[\"i-0cb99161f6EXAMPLE\"]"
                },
                "Outputs": {
                    "InstanceStates": [
                        "running"
                    ]
                },
                "StepExecutionId": "95e70479-cf20-4d80-8018-7e4e2EXAMPLE",
                "OverriddenParameters": {}
            }
        ],
        "StepExecutionsTruncated": false,
        "Parameters": {
            "AutomationAssumeRole": [
                ""
            ],
            "InstanceId": [
                "i-0cb99161f6EXAMPLE"
            ]
        },
        "Outputs": {},
        "Mode": "Auto",
        "ExecutedBy": "arn:aws:sts::29884EXAMPLE:assumed-role/mw_service_role/
OrchestrationService",
        "Targets": [],
```

```
        "ResolvedTargets": {
            "ParameterValues": [],
            "Truncated": false
        }
    }
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的演练：修补 Linux AMI（AWS CLI）。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 GetAutomationExecution。

PowerShell

Tools for PowerShell V4

示例 1：此示例显示自动化执行的详细信息。

```
Get-SSMAutomationExecution -AutomationExecutionId "4105a4fc-
f944-11e6-9d32-8fb2db27a909"
```

输出：

```
AutomationExecutionId     : 4105a4fc-f944-11e6-9d32-8fb2db27a909
AutomationExecutionStatus : Failed
DocumentName              : AWS-UpdateLinuxAmi
DocumentVersion           : 1
ExecutionEndTime          : 2/22/2017 9:17:08 PM
ExecutionStartTime        : 2/22/2017 9:17:02 PM
FailureMessage            : Step launchInstance failed maximum allowed times. You
 are not authorized to perform this operation. Encoded
                            authorization failure message:
 B_V2QyyN7NhSZQYpmVzpEc4oSnj2GLTNYnXUHsTbqJkNMoDgubmbtthLmZyaiUYekORIrA42-
fv1x-04q5Fjff6glh
                            Yb6TI5b0GQeeNrpwNvpDzmO-
PSR1swlAbg9fdM9BcNjyrznspUkWpuKu9EC1Ou6v3OXU1KC9nZ7mPlWMFZNkSioQqpwWEvMw-
GZktsQzm67qOhUhBNOLWYhbS
                            pkfiqzY-5nw3S0obx30fhd3EJa5O_-
GjV_a0nFXQJa70ik40bFOrEh3MtCSbrQT6--DvFy_FQ8TKvkIXadyVskeJI84XOF5WmA60f1pi5GI08i-
nRfZS6oDeU
```

```
 gELBjjoFKD8s3L2aI0B6umWVxnQOjqhQRxwJ53b54sZJ2PW3v_mtg9-q0CK0ezS3xfh_y0ilaUGOAZG-
xjQFuvU_JZedWpla3xi-MZsmblAifBI
                            (Service: AmazonEC2; Status Code: 403; Error Code:
 UnauthorizedOperation; Request ID:
                            6a002f94-ba37-43fd-99e6-39517715fce5)
Outputs                   : {[createImage.ImageId,
 Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
Parameters                : {[AutomationAssumeRole,
 Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]], [InstanceIamRole,

 Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]], [SourceAmiId,

 Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
StepExecutions            : {launchInstance, updateOSSoftware, stopInstance,
 createImage...}
```

示例 2：此示例列出给定自动化执行 ID 的步骤详细信息

```
Get-SSMAutomationExecution -AutomationExecutionId e1d2bad3-4567-8901-
ae23-456c7c8901be | Select-Object -ExpandProperty StepExecutions | Select-Object
 StepName, Action, StepStatus, ValidNextSteps
```

输出：

```
StepName                  Action                  StepStatus ValidNextSteps
--------                  ------                  ---------- --------------
LaunchInstance            aws:runInstances        Success
 {OSCompatibilityCheck}
OSCompatibilityCheck      aws:runCommand          Success    {RunPreUpdateScript}
RunPreUpdateScript        aws:runCommand          Success    {UpdateEC2Config}
UpdateEC2Config           aws:runCommand          Cancelled  {}
UpdateSSMAgent            aws:runCommand          Pending    {}
UpdateAWSPVDriver         aws:runCommand          Pending    {}
UpdateAWSEnaNetworkDriver aws:runCommand          Pending    {}
UpdateAWSNVMe             aws:runCommand          Pending    {}
InstallWindowsUpdates     aws:runCommand          Pending    {}
RunPostUpdateScript       aws:runCommand          Pending    {}
RunSysprepGeneralize      aws:runCommand          Pending    {}
StopInstance              aws:changeInstanceState Pending    {}
CreateImage               aws:createImage         Pending    {}
TerminateInstance         aws:changeInstanceState Pending    {}
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 GetAutomationExecution。

Tools for PowerShell V5

示例 1：此示例显示自动化执行的详细信息。

```
Get-SSMAutomationExecution -AutomationExecutionId "4105a4fc-
f944-11e6-9d32-8fb2db27a909"
```

输出：

```
AutomationExecutionId     : 4105a4fc-f944-11e6-9d32-8fb2db27a909
AutomationExecutionStatus : Failed
DocumentName              : AWS-UpdateLinuxAmi
DocumentVersion           : 1
ExecutionEndTime          : 2/22/2017 9:17:08 PM
ExecutionStartTime        : 2/22/2017 9:17:02 PM
FailureMessage            : Step launchInstance failed maximum allowed times. You
 are not authorized to perform this operation. Encoded
                            authorization failure message:
 B_V2QyyN7NhSZQYpmVzpEc4oSnj2GLTNYnXUHsTbqJkNMoDgubmbtthLmZyaiUYekORIrA42-
fv1x-04q5Fjff6glh
                            Yb6TI5b0GQeeNrpwNvpDzmO-
PSR1swlAbg9fdM9BcNjyrznspUkWpuKu9EC1Ou6v3OXU1KC9nZ7mPlWMFZNkSioQqpwWEvMw-
GZktsQzm67qOhUhBNOLWYhbS
                            pkfiqzY-5nw3S0obx30fhd3EJa5O_-
GjV_a0nFXQJa70ik40bFOrEh3MtCSbrQT6--DvFy_FQ8TKvkIXadyVskeJI84XOF5WmA60f1pi5GI08i-
nRfZS6oDeU

 gELBjjoFKD8s3L2aI0B6umWVxnQOjqhQRxwJ53b54sZJ2PW3v_mtg9-q0CK0ezS3xfh_y0ilaUGOAZG-
xjQFuvU_JZedWpla3xi-MZsmblAifBI
                            (Service: AmazonEC2; Status Code: 403; Error Code:
 UnauthorizedOperation; Request ID:
                            6a002f94-ba37-43fd-99e6-39517715fce5)
Outputs                   : {[createImage.ImageId,
 Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
Parameters                : {[AutomationAssumeRole,
 Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]], [InstanceIamRole,

 Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]], [SourceAmiId,

 Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
```

```
StepExecutions             : {launchInstance, updateOSSoftware, stopInstance,
 createImage...}
```

示例 2：此示例列出给定自动化执行 ID 的步骤详细信息

```
Get-SSMAutomationExecution -AutomationExecutionId e1d2bad3-4567-8901-
ae23-456c7c8901be | Select-Object -ExpandProperty StepExecutions | Select-Object
 StepName, Action, StepStatus, ValidNextSteps
```

输出：

```
StepName                  Action                  StepStatus ValidNextSteps
--------                  ------                  ---------- --------------
LaunchInstance            aws:runInstances        Success
 {OSCompatibilityCheck}
OSCompatibilityCheck      aws:runCommand          Success    {RunPreUpdateScript}
RunPreUpdateScript        aws:runCommand          Success    {UpdateEC2Config}
UpdateEC2Config           aws:runCommand          Cancelled  {}
UpdateSSMAgent            aws:runCommand          Pending    {}
UpdateAWSPVDriver         aws:runCommand          Pending    {}
UpdateAWSEnaNetworkDriver aws:runCommand          Pending    {}
UpdateAWSNVMe             aws:runCommand          Pending    {}
InstallWindowsUpdates     aws:runCommand          Pending    {}
RunPostUpdateScript       aws:runCommand          Pending    {}
RunSysprepGeneralize      aws:runCommand          Pending    {}
StopInstance              aws:changeInstanceState Pending    {}
CreateImage               aws:createImage         Pending    {}
TerminateInstance         aws:changeInstanceState Pending    {}
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 GetAutomationExecution。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **GetCommandInvocation** 与 CLI 配合使用

以下代码示例演示如何使用 GetCommandInvocation。

CLI

AWS CLI

显示命令调用的详细信息

以下 get-command-invocation 示例列出对指定实例上指定命令的所有调用。

```
aws ssm get-command-invocation \
    --command-id "ef7fdfd8-9b57-4151-a15c-db9a12345678" \
    --instance-id "i-1234567890abcdef0"
```

输出：

```
{
    "CommandId": "ef7fdfd8-9b57-4151-a15c-db9a12345678",
    "InstanceId": "i-1234567890abcdef0",
    "Comment": "b48291dd-ba76-43e0-b9df-13e11ddaac26:6960febb-2907-4b59-8e1a-
d6ce8EXAMPLE",
    "DocumentName": "AWS-UpdateSSMAgent",
    "DocumentVersion": "",
    "PluginName": "aws:updateSsmAgent",
    "ResponseCode": 0,
    "ExecutionStartDateTime": "2020-02-19T18:18:03.419Z",
    "ExecutionElapsedTime": "PT0.091S",
    "ExecutionEndDateTime": "2020-02-19T18:18:03.419Z",
    "Status": "Success",
    "StatusDetails": "Success",
    "StandardOutputContent": "Updating amazon-ssm-agent from 2.3.842.0 to latest
\nSuccessfully downloaded https://s3.us-east-2.amazonaws.com/amazon-ssm-us-
east-2/ssm-agent-manifest.json\namazon-ssm-agent 2.3.842.0 has already been
 installed, update skipped\n",
    "StandardOutputUrl": "",
    "StandardErrorContent": "",
    "StandardErrorUrl": "",
    "CloudWatchOutputConfig": {
        "CloudWatchLogGroupName": "",
        "CloudWatchOutputEnabled": false
    }
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的了解命令状态。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 GetCommandInvocation。

PowerShell

Tools for PowerShell V4

示例 1：此示例显示在实例上执行的命令的详细信息。

```
Get-SSMCommandInvocationDetail -InstanceId "i-0cb2b964d3e14fd9f" -CommandId
 "b8eac879-0541-439d-94ec-47a80d554f44"
```

输出：

```
CommandId              : b8eac879-0541-439d-94ec-47a80d554f44
Comment                : IP config
DocumentName           : AWS-RunShellScript
ExecutionElapsedTime   : PT0.004S
ExecutionEndDateTime   : 2017-02-22T20:13:16.651Z
ExecutionStartDateTime : 2017-02-22T20:13:16.651Z
InstanceId             : i-0cb2b964d3e14fd9f
PluginName             : aws:runShellScript
ResponseCode           : 0
StandardErrorContent   :
StandardErrorUrl       :
StandardOutputContent  :
StandardOutputUrl      :
Status                 : Success
StatusDetails          : Success
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 GetCommandInvocation。

Tools for PowerShell V5

示例 1：此示例显示在实例上执行的命令的详细信息。

```
Get-SSMCommandInvocationDetail -InstanceId "i-0cb2b964d3e14fd9f" -CommandId
 "b8eac879-0541-439d-94ec-47a80d554f44"
```

输出：

```
CommandId              : b8eac879-0541-439d-94ec-47a80d554f44
Comment                : IP config
DocumentName           : AWS-RunShellScript
ExecutionElapsedTime   : PT0.004S
ExecutionEndDateTime   : 2017-02-22T20:13:16.651Z
ExecutionStartDateTime : 2017-02-22T20:13:16.651Z
InstanceId             : i-0cb2b964d3e14fd9f
PluginName             : aws:runShellScript
ResponseCode           : 0
StandardErrorContent   :
StandardErrorUrl       :
StandardOutputContent  :
StandardOutputUrl      :
Status                 : Success
StatusDetails          : Success
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 GetCommandInvocation。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **GetConnectionStatus** 与 CLI 配合使用

以下代码示例演示如何使用 GetConnectionStatus。

CLI

AWS CLI

显示托管式实例的连接状态

此 get-connection-status 示例返回指定托管式实例的连接状态。

```
aws ssm get-connection-status \
    --target i-1234567890abcdef0
```

输出：

```
{
    "Target": "i-1234567890abcdef0",
    "Status": "connected"
```

```
}
```

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 GetConnectionStatus。

PowerShell

Tools for PowerShell V4

示例 1：此示例检索实例的 Session Manager 连接状态，以确定其是否已连接并准备好接收 Session Manager 连接。

```
Get-SSMConnectionStatus -Target i-0a1caf234f12d3dc4
```

输出：

```
Status    Target
------    ------
Connected i-0a1caf234f12d3dc4
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 GetConnectionStatus。

Tools for PowerShell V5

示例 1：此示例检索实例的 Session Manager 连接状态，以确定其是否已连接并准备好接收 Session Manager 连接。

```
Get-SSMConnectionStatus -Target i-0a1caf234f12d3dc4
```

输出：

```
Status    Target
------    ------
Connected i-0a1caf234f12d3dc4
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 GetConnectionStatus。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **GetDefaultPatchBaseline** 与 CLI 配合使用

以下代码示例演示如何使用 GetDefaultPatchBaseline。

CLI

AWS CLI

示例 1：显示默认 Windows 补丁基准

以下 get-default-patch-baseline 示例检索 Windows Server 默认补丁基准的详细信息。

```
aws ssm get-default-patch-baseline
```

输出：

```
{
  "BaselineId": "pb-0713accee01612345",
  "OperatingSystem": "WINDOWS"
}
```

示例 2：显示 Amazon Linux 的默认补丁基准

以下 get-default-patch-baseline 示例检索 Amazon Linux 默认补丁基准的详细信息。

```
aws ssm get-default-patch-baseline \
    --operating-system AMAZON_LINUX
```

输出：

```
{
    "BaselineId": "pb-047c6eb9c8fc12345",
    "OperatingSystem": "AMAZON_LINUX"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的关于预定义和自定义补丁基准 <https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-baselines.html>__ 和将现有补丁基准设置为默认项。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 GetDefaultPatchBaseline。

PowerShell

Tools for PowerShell V4

示例 1：此示例显示默认补丁基准。

```
Get-SSMDefaultPatchBaseline
```

输出：

```
arn:aws:ssm:us-west-2:123456789012:patchbaseline/pb-04fb4ae6142167966
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 GetDefaultPatchBaseline。

Tools for PowerShell V5

示例 1：此示例显示默认补丁基准。

```
Get-SSMDefaultPatchBaseline
```

输出：

```
arn:aws:ssm:us-west-2:123456789012:patchbaseline/pb-04fb4ae6142167966
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 GetDefaultPatchBaseline。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **GetDeployablePatchSnapshotForInstance** 与 CLI 配合使用

以下代码示例演示如何使用 GetDeployablePatchSnapshotForInstance。

CLI

AWS CLI

检索实例使用的补丁基准的当前快照

以下 get-deployable-patch-snapshot-for-instance 示例检索实例使用的指定补丁基准当前快照的详细信息。此命令必须使用实例凭证从实例运行。为确保其使用实例凭证，请运行 aws configure 并仅指定您的实例的区域。将 Access Key 和 Secret Key 字段留空。

提示：使用 uuidgen 生成 snapshot-id。

```
aws ssm get-deployable-patch-snapshot-for-instance \
    --instance-id "i-1234567890abcdef0" \
    --snapshot-id "521c3536-930c-4aa9-950e-01234567abcd"
```

输出：

```
{
    "InstanceId": "i-1234567890abcdef0",
    "SnapshotId": "521c3536-930c-4aa9-950e-01234567abcd",
    "Product": "AmazonLinux2018.03",
    "SnapshotDownloadUrl": "https://patch-baseline-snapshot-us-
east-1.s3.amazonaws.com/
ed85194ef27214f5984f28b4d664d14f7313568fea7d4b6ac6c10ad1f729d7e7-773304212436/
AMAZON_LINUX-521c3536-930c-4aa9-950e-01234567abcd?X-Amz-
Algorithm=AWS4-HMAC-SHA256&X-Amz-Date=20190215T164031Z&X-Amz-
SignedHeaders=host&X-Amz-Expires=86400&X-Amz-Credential=AKIAJ5C56P35AEBRX2QQ
%2F20190215%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-
Signature=efaaaf6e3878e77f48a6697e015efdbda9c426b09c5822055075c062f6ad2149"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的参数名称：快照 ID。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 GetDeployablePatchSnapshotForInstance。

PowerShell

Tools for PowerShell V4

示例 1：此示例显示实例使用的补丁基准的当前快照。此命令必须使用实例凭证从实例运行。为确保其使用实例证书，该示例将 **Amazon.Runtime.InstanceProfileAWSCredentials** 对象传递给 Credentials 参数。

```
$credentials = [Amazon.Runtime.InstanceProfileAWSCredentials]::new()
```

```
Get-SSMDeployablePatchSnapshotForInstance -SnapshotId "4681775b-098f-4435-
a956-0ef33373ac11" -InstanceId "i-0cb2b964d3e14fd9f" -Credentials $credentials
```

输出：

```
InstanceId          SnapshotDownloadUrl
----------          -------------------
i-0cb2b964d3e14fd9f https://patch-baseline-snapshot-us-west-2.s3-us-
west-2.amazonaws.com/853d0d3db0f0cafe...1692/4681775b-098f-4435...
```

示例 2：此示例展示如何获取完整的 SnapshotDownloadUrl。此命令必须使用实例凭证从实例运行。为确保其使用实例凭证，该示例将 PowerShell 会话配置为使用 **Amazon.Runtime.InstanceProfileAWSCredentials** 对象。

```
Set-AWSCredential -Credential
 ([Amazon.Runtime.InstanceProfileAWSCredentials]::new())
(Get-SSMDeployablePatchSnapshotForInstance -SnapshotId "4681775b-098f-4435-
a956-0ef33373ac11" -InstanceId "i-0cb2b964d3e14fd9f").SnapshotDownloadUrl
```

输出：

```
https://patch-baseline-snapshot-us-west-2.s3-us-
west-2.amazonaws.com/853d0d3db0f0cafe...
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 GetDeployablePatchSnapshotForInstance。

Tools for PowerShell V5

示例 1：此示例显示实例使用的补丁基准的当前快照。此命令必须使用实例凭证从实例运行。为确保其使用实例证书，该示例将 **Amazon.Runtime.InstanceProfileAWSCredentials** 对象传递给 Credentials 参数。

```
$credentials = [Amazon.Runtime.InstanceProfileAWSCredentials]::new()
Get-SSMDeployablePatchSnapshotForInstance -SnapshotId "4681775b-098f-4435-
a956-0ef33373ac11" -InstanceId "i-0cb2b964d3e14fd9f" -Credentials $credentials
```

输出：

```
InstanceId          SnapshotDownloadUrl
----------          -------------------
```

```
i-0cb2b964d3e14fd9f https://patch-baseline-snapshot-us-west-2.s3-us-
west-2.amazonaws.com/853d0d3db0f0cafe...1692/4681775b-098f-4435...
```

示例 2：此示例展示如何获取完整的 SnapshotDownloadUrl。此命令必须使用实例凭证从实例运行。为确保其使用实例凭证，该示例将 PowerShell 会话配置为使用 **Amazon.Runtime.InstanceProfileAWSCredentials** 对象。

```
Set-AWSCredential -Credential
 ([Amazon.Runtime.InstanceProfileAWSCredentials]::new())
(Get-SSMDeployablePatchSnapshotForInstance -SnapshotId "4681775b-098f-4435-
a956-0ef33373ac11" -InstanceId "i-0cb2b964d3e14fd9f").SnapshotDownloadUrl
```

输出：

```
https://patch-baseline-snapshot-us-west-2.s3-us-
west-2.amazonaws.com/853d0d3db0f0cafe...
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 GetDeployablePatchSnapshotForInstance。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 GetDocument 与 CLI 配合使用

以下代码示例演示如何使用 GetDocument。

CLI

AWS CLI

获取文档内容

以下 get-document 示例显示 Systems Manager 文档的内容。

```
aws ssm get-document \
    --name "AWS-RunShellScript"
```

输出：

```
{
```

```
    "Name": "AWS-RunShellScript",
    "DocumentVersion": "1",
    "Status": "Active",
    "Content": "{\n    \"schemaVersion\":\"1.2\",\n    \"description\":\"Run
 a shell script or specify the commands to run.\",\n    \"parameters\":{\n
       \"commands\":{\n            \"type\":\"StringList\",\n
 \"description\":\"(Required) Specify a shell script or a command to run.\",
\n            \"minItems\":1,\n            \"displayType\":\"textarea\"\n
      },\n        \"workingDirectory\":{\n            \"type\":\"String\",\n
            \"default\":\"\",\n            \"description\":\"(Optional) The
 path to the working directory on your instance.\",\n            \"maxChars
\":4096\n        },\n        \"executionTimeout\":{\n            \"type\":
\"String\",\n            \"default\":\"3600\",\n            \"description
\":\"(Optional) The time in seconds for a command to complete before it is
 considered to have failed. Default is 3600 (1 hour). Maximum is 172800 (48
 hours).\",\n            \"allowedPattern\":\"([1-9][0-9]{0,4})|(1[0-6][0-9]
{4})|(17[0-1][0-9]{3})|(172[0-7][0-9]{2})|(172800)\"\n        }\n    },\n
 \"runtimeConfig\":{\n        \"aws:runShellScript\":{\n            \"properties
\":[\n                {\n                    \"id\":\"0.aws:runShellScript
\",\n                    \"runCommand\":\"{{ commands }}\",\n
      \"workingDirectory\":\"{{ workingDirectory }}\",\n
 \"timeoutSeconds\":\"{{ executionTimeout }}\"\n                }\n
  ]\n        }\n    }\n}\n",
    "DocumentType": "Command",
    "DocumentFormat": "JSON"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的 AWS Systems Manager 文档。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 GetDocument。

PowerShell

Tools for PowerShell V4

示例 1：此示例返回文档的内容。

```
Get-SSMDocument -Name "RunShellScript"
```

输出：

```
Content
```

```
-------
{...
```

示例 2：此示例显示文档的完整内容。

```
(Get-SSMDocument -Name "RunShellScript").Content
{
   "schemaVersion":"2.0",
   "description":"Run an updated script",
   "parameters":{
      "commands":{
         "type":"StringList",
         "description":"(Required) Specify a shell script or a command to run.",
         "minItems":1,
         "displayType":"textarea"
      }
   },
   "mainSteps":[
      {
         "action":"aws:runShellScript",
         "name":"runShellScript",
         "inputs":{
            "commands":"{{ commands }}"
         }
      },
      {
         "action":"aws:runPowerShellScript",
         "name":"runPowerShellScript",
         "inputs":{
            "commands":"{{ commands }}"
         }
      }
   ]
}
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 GetDocument。

Tools for PowerShell V5

示例 1：此示例返回文档的内容。

```
Get-SSMDocument -Name "RunShellScript"
```

输出：

```
Content
-------
{...
```

示例 2：此示例显示文档的完整内容。

```
(Get-SSMDocument -Name "RunShellScript").Content
{
   "schemaVersion":"2.0",
   "description":"Run an updated script",
   "parameters":{
      "commands":{
         "type":"StringList",
         "description":"(Required) Specify a shell script or a command to run.",
         "minItems":1,
         "displayType":"textarea"
      }
   },
   "mainSteps":[
      {
         "action":"aws:runShellScript",
         "name":"runShellScript",
         "inputs":{
            "commands":"{{ commands }}"
         }
      },
      {
         "action":"aws:runPowerShellScript",
         "name":"runPowerShellScript",
         "inputs":{
            "commands":"{{ commands }}"
         }
      }
   ]
}
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 GetDocument。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **GetInventory** 与 CLI 配合使用

以下代码示例演示如何使用 GetInventory。

CLI

AWS CLI

查看您的清单

此示例获取清单的自定义元数据。

命令:

```
aws ssm get-inventory
```

输出：

```
{
  "Entities": [
      {
          "Data": {
              "AWS:InstanceInformation": {
                  "Content": [
                      {
                          "ComputerName": "ip-172-31-44-222.us-
west-2.compute.internal",
                          "InstanceId": "i-0cb2b964d3e14fd9f",
                          "IpAddress": "172.31.44.222",
                          "AgentType": "amazon-ssm-agent",
                          "ResourceType": "EC2Instance",
                          "AgentVersion": "2.0.672.0",
                          "PlatformVersion": "2016.09",
                          "PlatformName": "Amazon Linux AMI",
                          "PlatformType": "Linux"
                      }
                  ],
                  "TypeName": "AWS:InstanceInformation",
                  "SchemaVersion": "1.0",
                  "CaptureTime": "2017-02-20T18:03:58Z"
```

```
                }
            },
            "Id": "i-0cb2b964d3e14fd9f"
        }
    ]
}
```

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 GetInventory。

PowerShell

Tools for PowerShell V4

示例 1：此示例获取清单的自定义元数据。

```
Get-SSMInventory
```

输出：

```
Data
     Id
----
     --
{[AWS:InstanceInformation,
 Amazon.SimpleSystemsManagement.Model.InventoryResultItem]} i-0cb2b964d3e14fd9f
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 GetInventory。

Tools for PowerShell V5

示例 1：此示例获取清单的自定义元数据。

```
Get-SSMInventory
```

输出：

```
Data
     Id
----
     --
```

```
{[AWS:InstanceInformation,
 Amazon.SimpleSystemsManagement.Model.InventoryResultItem]} i-0cb2b964d3e14fd9f
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 GetInventory。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **GetInventorySchema** 与 CLI 配合使用

以下代码示例演示如何使用 GetInventorySchema。

CLI

AWS CLI

查看您的清单架构

此示例返回账户清单类型名称列表。

命令:

```
aws ssm get-inventory-schema
```

输出：

```
{
  "Schemas": [
      {
          "TypeName": "AWS:AWSComponent",
          "Version": "1.0",
          "Attributes": [
              {
                  "Name": "Name",
                  "DataType": "STRING"
              },
              {
                  "Name": "ApplicationType",
                  "DataType": "STRING"
              },
```

```
                {
                    "Name": "Publisher",
                    "DataType": "STRING"
                },
                {
                    "Name": "Version",
                    "DataType": "STRING"
                },
                {
                    "Name": "InstalledTime",
                    "DataType": "STRING"
                },
                {
                    "Name": "Architecture",
                    "DataType": "STRING"
                },
                {
                    "Name": "URL",
                    "DataType": "STRING"
                }
            ]
        },
        ...
    ],
    "NextToken": "--token string truncated--"
}
```

查看特定清单类型的清单架构

此示例返回 AWS:AWSComponent 清单类型的清单架构。

命令:

```
aws ssm get-inventory-schema --type-name "AWS:AWSComponent"
```

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 GetInventorySchema。

PowerShell

Tools for PowerShell V4

示例 1：此示例返回账户清单类型名称列表。

```
Get-SSMInventorySchema
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 GetInventorySchema。

Tools for PowerShell V5

示例 1：此示例返回账户清单类型名称列表。

```
Get-SSMInventorySchema
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 GetInventorySchema。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 GetMaintenanceWindow 与 CLI 配合使用

以下代码示例演示如何使用 GetMaintenanceWindow。

CLI

AWS CLI

查看有关维护时段的信息

以下 get-maintenance-window 示例将检索指定维护时段的详细信息。

```
aws ssm get-maintenance-window \
    --window-id "mw-03eb9db428EXAMPLE"
```

输出：

```
{
    "AllowUnassociatedTargets": true,
    "CreatedDate": 1515006912.957,
    "Cutoff": 1,
    "Duration": 6,
    "Enabled": true,
    "ModifiedDate": 2020-01-01T10:04:04.099Z,
```

```
    "Name": "My-Maintenance-Window",
    "Schedule": "rate(3 days)",
    "WindowId": "mw-03eb9db428EXAMPLE",
    "NextExecutionTime": "2020-02-25T00:08:15.099Z"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的查看有关维护时段的信息（AWS CLI）。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 GetMaintenanceWindow。

PowerShell

Tools for PowerShell V4

示例 1：此示例获取有关维护时段的详细信息。

```
Get-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d"
```

输出：

```
AllowUnassociatedTargets : False
CreatedDate              : 2/20/2017 6:14:05 PM
Cutoff                   : 1
Duration                 : 2
Enabled                  : True
ModifiedDate             : 2/20/2017 6:14:05 PM
Name                     : TestMaintWin
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 GetMaintenanceWindow。

Tools for PowerShell V5

示例 1：此示例获取有关维护时段的详细信息。

```
Get-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d"
```

输出：

```
AllowUnassociatedTargets : False
CreatedDate              : 2/20/2017 6:14:05 PM
Cutoff                   : 1
Duration                 : 2
Enabled                  : True
ModifiedDate             : 2/20/2017 6:14:05 PM
Name                     : TestMaintWin
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 GetMaintenanceWindow。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 GetMaintenanceWindowExecution 与 CLI 配合使用

以下代码示例演示如何使用 GetMaintenanceWindowExecution。

CLI

AWS CLI

获取有关维护时段任务执行的信息

以下 get-maintenance-window-execution 示例列出有关指定维护时段执行组成部分来执行的任务的信息。

```
aws ssm get-maintenance-window-execution \
    --window-execution-id "518d5565-5969-4cca-8f0e-da3b2EXAMPLE"
```

输出：

```
{
    "Status": "SUCCESS",
    "TaskIds": [
        "ac0c6ae1-daa3-4a89-832e-d3845EXAMPLE"
    ],
    "StartTime": 1487692834.595,
    "EndTime": 1487692835.051,
```

```
    "WindowExecutionId": "518d5565-5969-4cca-8f0e-da3b2EXAMPLE",
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的查看有关任务和任务执行的信息（AWS CLI）。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 GetMaintenanceWindowExecution。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出有关作为维护时段执行组成部分来执行的任务的信息。

```
Get-SSMMaintenanceWindowExecution -WindowExecutionId "518d5565-5969-4cca-8f0e-
da3b2a638355"
```

输出：

```
EndTime           : 2/21/2017 4:00:35 PM
StartTime         : 2/21/2017 4:00:34 PM
Status            : FAILED
StatusDetails     : One or more tasks in the orchestration failed.
TaskIds           : {ac0c6ae1-daa3-4a89-832e-d384503b6586}
WindowExecutionId : 518d5565-5969-4cca-8f0e-da3b2a638355
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 GetMaintenanceWindowExecution。

Tools for PowerShell V5

示例 1：此示例列出有关作为维护时段执行组成部分来执行的任务的信息。

```
Get-SSMMaintenanceWindowExecution -WindowExecutionId "518d5565-5969-4cca-8f0e-
da3b2a638355"
```

输出：

```
EndTime           : 2/21/2017 4:00:35 PM
StartTime         : 2/21/2017 4:00:34 PM
```

```
Status            : FAILED
StatusDetails     : One or more tasks in the orchestration failed.
TaskIds           : {ac0c6ae1-daa3-4a89-832e-d384503b6586}
WindowExecutionId : 518d5565-5969-4cca-8f0e-da3b2a638355
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 GetMaintenanceWindowExecution。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **GetMaintenanceWindowExecutionTask** 与 CLI 配合使用

以下代码示例演示如何使用 GetMaintenanceWindowExecutionTask。

CLI

AWS CLI

获取有关维护时段任务执行的信息

以下 get-maintenance-window-execution-task 示例列出有关作为指定维护时段执行组成部分的任务的信息。

```
aws ssm get-maintenance-window-execution-task \
    --window-execution-id "518d5565-5969-4cca-8f0e-da3b2EXAMPLE" \
    --task-id "ac0c6ae1-daa3-4a89-832e-d3845EXAMPLE"
```

输出：

```
{
    "WindowExecutionId": "518d5565-5969-4cca-8f0e-da3b2EXAMPLE",
    "TaskExecutionId": "ac0c6ae1-daa3-4a89-832e-d3845EXAMPLE",
    "TaskArn": "AWS-RunPatchBaseline",
    "ServiceRole": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
    "Type": "RUN_COMMAND",
    "TaskParameters": [
        {
            "BaselineOverride": {
                "Values": [
                    ""
```

```
                ]
            },
            "InstallOverrideList": {
                "Values": [
                    ""
                ]
            },
            "Operation": {
                "Values": [
                    "Scan"
                ]
            },
            "RebootOption": {
                "Values": [
                    "RebootIfNeeded"
                ]
            },
            "SnapshotId": {
                "Values": [
                    "{{ aws:ORCHESTRATION_ID }}"
                ]
            },
            "aws:InstanceId": {
                "Values": [
                    "i-02573cafcfEXAMPLE",
                    "i-0471e04240EXAMPLE",
                    "i-07782c72faEXAMPLE"
                ]
            }
        }
    ],
    "Priority": 1,
    "MaxConcurrency": "1",
    "MaxErrors": "3",
    "Status": "SUCCESS",
    "StartTime": "2021-08-04T11:45:35.088000-07:00",
    "EndTime": "2021-08-04T11:53:09.079000-07:00"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的查看有关任务和任务执行的信息（AWS CLI）。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 GetMaintenanceWindowExecutionTask。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出有关维护时段执行组成部分的任务的信息。

```
Get-SSMMaintenanceWindowExecutionTask -TaskId "ac0c6ae1-daa3-4a89-832e-
d384503b6586" -WindowExecutionId "518d5565-5969-4cca-8f0e-da3b2a638355"
```

输出：

```
EndTime           : 2/21/2017 4:00:35 PM
MaxConcurrency    : 1
MaxErrors         : 1
Priority          : 10
ServiceRole       : arn:aws:iam::123456789012:role/MaintenanceWindowsRole
StartTime         : 2/21/2017 4:00:34 PM
Status            : FAILED
StatusDetails     : The maximum error count was exceeded.
TaskArn           : AWS-RunShellScript
TaskExecutionId   : ac0c6ae1-daa3-4a89-832e-d384503b6586
TaskParameters    :
 {Amazon.Runtime.Internal.Util.AlwaysSendDictionary`2[System.String,Amazon.SimpleSystemsM
                   meterValueExpression]}
Type              : RUN_COMMAND
WindowExecutionId : 518d5565-5969-4cca-8f0e-da3b2a638355
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 GetMaintenanceWindowExecutionTask。

Tools for PowerShell V5

示例 1：此示例列出有关维护时段执行组成部分的任务的信息。

```
Get-SSMMaintenanceWindowExecutionTask -TaskId "ac0c6ae1-daa3-4a89-832e-
d384503b6586" -WindowExecutionId "518d5565-5969-4cca-8f0e-da3b2a638355"
```

输出：

```
EndTime           : 2/21/2017 4:00:35 PM
MaxConcurrency    : 1
```

```
MaxErrors         : 1
Priority          : 10
ServiceRole       : arn:aws:iam::123456789012:role/MaintenanceWindowsRole
StartTime         : 2/21/2017 4:00:34 PM
Status            : FAILED
StatusDetails     : The maximum error count was exceeded.
TaskArn           : AWS-RunShellScript
TaskExecutionId   : ac0c6ae1-daa3-4a89-832e-d384503b6586
TaskParameters    :
 {Amazon.Runtime.Internal.Util.AlwaysSendDictionary`2[System.String,Amazon.SimpleSystemsM
                     meterValueExpression]}
Type              : RUN_COMMAND
WindowExecutionId : 518d5565-5969-4cca-8f0e-da3b2a638355
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 GetMaintenanceWindowExecutionTask。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **GetParameter** 与 AWS SDK 或 CLI 配合使用

以下代码示例演示如何使用 GetParameter。

CLI

AWS CLI

示例 1：显示参数的值

以下 get-parameter 示例列出指定单个参数的值。

```
aws ssm get-parameter \
    --name "MyStringParameter"
```

输出：

```
{
    "Parameter": {
        "Name": "MyStringParameter",
        "Type": "String",
```

```
        "Value": "Veni",
        "Version": 1,
        "LastModifiedDate": 1530018761.888,
        "ARN": "arn:aws:ssm:us-east-2:111222333444:parameter/MyStringParameter"
        "DataType": "text"
    }
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的使用 Parameter Store。

示例 2：解密 SecureString 参数的值

以下 get-parameter 示例解密指定 SecureString 参数的值。

```
aws ssm get-parameter \
    --name "MySecureStringParameter" \
    --with-decryption
```

输出：

```
{
    "Parameter": {
        "Name": "MySecureStringParameter",
        "Type": "SecureString",
        "Value": "16679b88-310b-4895-a943-e0764EXAMPLE",
        "Version": 2,
        "LastModifiedDate": 1582155479.205,
        "ARN": "arn:aws:ssm:us-east-2:111222333444:parameter/
MySecureStringParameter"
        "DataType": "text"
    }
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的使用 Parameter Store。

示例 3：使用标签显示参数的值

以下 get-parameter 示例列出具有指定标签的指定单个参数的值。

```
aws ssm get-parameter \
    --name "MyParameter:label"
```

输出：

```
{
    "Parameter": {
        "Name": "MyParameter",
        "Type": "String",
        "Value": "parameter version 2",
        "Version": 2,
        "Selector": ":label",
        "LastModifiedDate": "2021-07-12T09:49:15.865000-07:00",
        "ARN": "arn:aws:ssm:us-west-2:786973925828:parameter/MyParameter",
        "DataType": "text"
    }
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的使用参数标签。

示例 4：使用版本显示参数的值

以下 get-parameter 示例列出指定单个参数版本的值。

```
aws ssm get-parameter \
    --name "MyParameter:2"
```

输出：

```
{
    "Parameter": {
        "Name": "MyParameter",
        "Type": "String",
        "Value": "parameter version 2",
        "Version": 2,
        "Selector": ":2",
        "LastModifiedDate": "2021-07-12T09:49:15.865000-07:00",
        "ARN": "arn:aws:ssm:us-west-2:786973925828:parameter/MyParameter",
        "DataType": "text"
    }
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的使用参数标签。

- 有关 API 详细信息，请参阅 AWS CLI 命令参考中的 GetParameter。

Rust

适用于 Rust 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    pub async fn list_path(&self, path: &str) -> Result<Vec<Parameter>, EC2Error>
{
        let maybe_params: Vec<Result<Parameter, _>> = TryFlatMap::new(
            self.inner
                .get_parameters_by_path()
                .path(path)
                .into_paginator()
                .send(),
        )
        .flat_map(|item| item.parameters.unwrap_or_default())
        .collect()
        .await;
        // Fail on the first error
        let params = maybe_params
            .into_iter()
            .collect::<Result<Vec<Parameter>, _>>()?;
        Ok(params)
    }
```

- 有关 API 详细信息，请参阅 AWS SDK for Rust API 参考中的 GetParameter。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **GetParameterHistory** 与 CLI 配合使用

以下代码示例演示如何使用 GetParameterHistory。

CLI

AWS CLI

获取参数的值历史记录

以下 get-parameter-history 示例列出指定参数的更改历史记录，包括其值。

```
aws ssm get-parameter-history \
    --name "MyStringParameter"
```

输出：

```
{
    "Parameters": [
        {
            "Name": "MyStringParameter",
            "Type": "String",
            "LastModifiedDate": 1582154711.976,
            "LastModifiedUser": "arn:aws:iam::111222333444:user/Mary-Major",
            "Description": "This is the first version of my String parameter",
            "Value": "Veni",
            "Version": 1,
            "Labels": [],
            "Tier": "Standard",
            "Policies": []
        },
        {
            "Name": "MyStringParameter",
            "Type": "String",
            "LastModifiedDate": 1582156093.471,
            "LastModifiedUser": "arn:aws:iam::111222333444:user/Mary-Major",
            "Description": "This is the second version of my String parameter",
            "Value": "Vidi",
            "Version": 2,
            "Labels": [],
            "Tier": "Standard",
            "Policies": []
        },
        {
            "Name": "MyStringParameter",
            "Type": "String",
            "LastModifiedDate": 1582156117.545,
```

```
            "LastModifiedUser": "arn:aws:iam::111222333444:user/Mary-Major",
            "Description": "This is the third version of my String parameter",
            "Value": "Vici",
            "Version": 3,
            "Labels": [],
            "Tier": "Standard",
            "Policies": []
        }
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的使用参数版本。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 GetParameterHistory。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出参数的值历史记录。

```
Get-SSMParameterHistory -Name "Welcome"
```

输出：

```
Description      :
KeyId            :
LastModifiedDate : 3/3/2017 6:55:25 PM
LastModifiedUser : arn:aws:iam::123456789012:user/admin
Name             : Welcome
Type             : String
Value            : helloWorld
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 GetParameterHistory。

Tools for PowerShell V5

示例 1：此示例列出参数的值历史记录。

```
Get-SSMParameterHistory -Name "Welcome"
```

输出：

```
Description      :
KeyId            :
LastModifiedDate : 3/3/2017 6:55:25 PM
LastModifiedUser : arn:aws:iam::123456789012:user/admin
Name             : Welcome
Type             : String
Value            : helloWorld
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 GetParameterHistory。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **GetParameters** 与 CLI 配合使用

以下代码示例演示如何使用 GetParameters。

CLI

AWS CLI

示例 1：列出参数的值

以下 get-parameters 示例列出三个指定参数的值。

```
aws ssm get-parameters \
    --names "MyStringParameter" "MyStringListParameter" "MyInvalidParameterName"
```

输出：

```
{
    "Parameters": [
        {
            "Name": "MyStringListParameter",
            "Type": "StringList",
            "Value": "alpha,beta,gamma",
            "Version": 1,
            "LastModifiedDate": 1582154764.222,
            "ARN": "arn:aws:ssm:us-east-2:111222333444:parameter/
MyStringListParameter"
            "DataType": "text"
```

```
        },
        {
            "Name": "MyStringParameter",
            "Type": "String",
            "Value": "Vici",
            "Version": 3,
            "LastModifiedDate": 1582156117.545,
            "ARN": "arn:aws:ssm:us-east-2:111222333444:parameter/
MyStringParameter"
            "DataType": "text"
        }
    ],
    "InvalidParameters": [
        "MyInvalidParameterName"
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的使用 Parameter Store。

示例 2：使用 ``--query`` 选项列出多个参数的名称和值

以下 `get-parameters` 示例列出指定参数的名称和值。

```
aws ssm get-parameters \
    --names MyStringParameter MyStringListParameter \
    --query "Parameters[*].{Name:Name,Value:Value}"
```

输出：

```
[
    {
        "Name": "MyStringListParameter",
        "Value": "alpha,beta,gamma"
    },
    {
        "Name": "MyStringParameter",
        "Value": "Vidi"
    }
]
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的使用 Parameter Store。

示例 3：使用标签显示参数的值

以下 get-parameter 示例列出具有指定标签的指定单个参数的值。

```
aws ssm get-parameter \
    --name "MyParameter:label"
```

输出：

```
{
    "Parameters": [
        {
            "Name": "MyLabelParameter",
            "Type": "String",
            "Value": "parameter by label",
            "Version": 1,
            "Selector": ":label",
            "LastModifiedDate": "2021-07-12T09:49:15.865000-07:00",
            "ARN": "arn:aws:ssm:us-west-2:786973925828:parameter/MyParameter",
            "DataType": "text"
        },
        {
            "Name": "MyVersionParameter",
            "Type": "String",
            "Value": "parameter by version",
            "Version": 2,
            "Selector": ":2",
            "LastModifiedDate": "2021-03-24T16:20:28.236000-07:00",
            "ARN": "arn:aws:ssm:us-west-2:786973925828:parameter/unlabel-param",
            "DataType": "text"
        }
    ],
    "InvalidParameters": []
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的使用参数标签。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 GetParameters。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出参数的值。

```
Get-SSMParameterValue -Name "Welcome"
```

输出：

```
InvalidParameters Parameters
----------------- ----------
{}                {Welcome}
```

示例 2：此示例列出值的详细信息。

```
(Get-SSMParameterValue -Name "Welcome").Parameters
```

输出：

```
Name    Type   Value
----    ----   -----
Welcome String Good day, Sunshine!
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 GetParameters。

Tools for PowerShell V5

示例 1：此示例列出参数的值。

```
Get-SSMParameterValue -Name "Welcome"
```

输出：

```
InvalidParameters Parameters
----------------- ----------
{}                {Welcome}
```

示例 2：此示例列出值的详细信息。

```
(Get-SSMParameterValue -Name "Welcome").Parameters
```

输出：

```
Name    Type   Value
----    ----   -----
Welcome String Good day, Sunshine!
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 GetParameters。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **GetPatchBaseline** 与 CLI 配合使用

以下代码示例演示如何使用 GetPatchBaseline。

CLI

AWS CLI

显示补丁基准

以下 get-patch-baseline 示例检索指定补丁基准的详细信息。

```
aws ssm get-patch-baseline \
    --baseline-id "pb-0123456789abcdef0"
```

输出：

```
{
    "BaselineId": "pb-0123456789abcdef0",
    "Name": "WindowsPatching",
    "OperatingSystem": "WINDOWS",
    "GlobalFilters": {
        "PatchFilters": []
    },
    "ApprovalRules": {
        "PatchRules": [
            {
                "PatchFilterGroup": {
                    "PatchFilters": [
                        {
                            "Key": "PRODUCT",
```

```
                            "Values": [
                                "WindowsServer2016"
                            ]
                        }
                    ]
                },
                "ComplianceLevel": "CRITICAL",
                "ApproveAfterDays": 0,
                "EnableNonSecurity": false
            }
        ]
    },
    "ApprovedPatches": [],
    "ApprovedPatchesComplianceLevel": "UNSPECIFIED",
    "ApprovedPatchesEnableNonSecurity": false,
    "RejectedPatches": [],
    "RejectedPatchesAction": "ALLOW_AS_DEPENDENCY",
    "PatchGroups": [
        "QA",
        "DEV"
    ],
    "CreatedDate": 1550244180.465,
    "ModifiedDate": 1550244180.465,
    "Description": "Patches for Windows Servers",
    "Sources": []
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的关于补丁基准。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 GetPatchBaseline。

PowerShell

Tools for PowerShell V4

示例 1：此示例显示补丁基准的详细信息。

```
Get-SSMPatchBaselineDetail -BaselineId "pb-03da896ca3b68b639"
```

输出：

```
ApprovalRules    : Amazon.SimpleSystemsManagement.Model.PatchRuleGroup
```

```
ApprovedPatches : {}
BaselineId      : pb-03da896ca3b68b639
CreatedDate     : 3/3/2017 5:02:19 PM
Description     : Baseline containing all updates approved for production systems
GlobalFilters   : Amazon.SimpleSystemsManagement.Model.PatchFilterGroup
ModifiedDate    : 3/3/2017 5:02:19 PM
Name            : Production-Baseline
PatchGroups     : {}
RejectedPatches : {}
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 GetPatchBaseline。

Tools for PowerShell V5

示例 1：此示例显示补丁基准的详细信息。

```
Get-SSMPatchBaselineDetail -BaselineId "pb-03da896ca3b68b639"
```

输出：

```
ApprovalRules   : Amazon.SimpleSystemsManagement.Model.PatchRuleGroup
ApprovedPatches : {}
BaselineId      : pb-03da896ca3b68b639
CreatedDate     : 3/3/2017 5:02:19 PM
Description     : Baseline containing all updates approved for production systems
GlobalFilters   : Amazon.SimpleSystemsManagement.Model.PatchFilterGroup
ModifiedDate    : 3/3/2017 5:02:19 PM
Name            : Production-Baseline
PatchGroups     : {}
RejectedPatches : {}
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 GetPatchBaseline。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **GetPatchBaselineForPatchGroup** 与 CLI 配合使用

以下代码示例演示如何使用 GetPatchBaselineForPatchGroup。

CLI

AWS CLI

显示补丁组的补丁基准

以下 get-patch-baseline-for-patch-group 示例检索有关指定补丁组补丁基准的详细信息。

```
aws ssm get-patch-baseline-for-patch-group \
    --patch-group "DEV"
```

输出：

```
{
    "PatchGroup": "DEV",
    "BaselineId": "pb-0123456789abcdef0",
    "OperatingSystem": "WINDOWS"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的创建补丁组 <https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-group-tagging.html>__ 和将补丁组添加到补丁基准。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 GetPatchBaselineForPatchGroup。

PowerShell

Tools for PowerShell V4

示例 1：此示例显示补丁组的补丁基准。

```
Get-SSMPatchBaselineForPatchGroup -PatchGroup "Production"
```

输出：

```
BaselineId           PatchGroup
----------           ----------
pb-045f10b4f382baeda Production
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 GetPatchBaselineForPatchGroup。

Tools for PowerShell V5

示例 1：此示例显示补丁组的补丁基准。

```
Get-SSMPatchBaselineForPatchGroup -PatchGroup "Production"
```

输出：

```
BaselineId           PatchGroup
----------           ----------
pb-045f10b4f382baeda Production
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 GetPatchBaselineForPatchGroup。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **ListAssociationVersions** 与 CLI 配合使用

以下代码示例演示如何使用 ListAssociationVersions。

CLI

AWS CLI

列出特定关联 ID 的关联的所有版本

以下 list-association-versions 示例列出指定关联的所有版本。

```
aws ssm list-association-versions \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

输出：

```
{
"AssociationVersions": [
```

```
        {
            "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
            "AssociationVersion": "1",
            "CreatedDate": 1550505536.726,
            "Name": "AWS-UpdateSSMAgent",
            "Parameters": {
                "allowDowngrade": [
                    "false"
                ],
                "version": [
                    ""
                ]
            },
            "Targets": [
                {
                    "Key": "InstanceIds",
                    "Values": [
                        "i-1234567890abcdef0"
                    ]
                }
            ],
            "ScheduleExpression": "cron(0 00 12 ? * SUN *)",
            "AssociationName": "UpdateSSMAgent"
        }
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的在 Systems Manager 中使用关联。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 ListAssociationVersions。

PowerShell

Tools for PowerShell V4

示例 1：此示例检索所提供关联的所有版本。

```
Get-SSMAssociationVersionList -AssociationId 123a45a0-c678-9012-3456-78901234db5e
```

输出：

```
AssociationId      : 123a45a0-c678-9012-3456-78901234db5e
AssociationName    :
AssociationVersion : 2
ComplianceSeverity :
CreatedDate        : 3/12/2019 9:21:01 AM
DocumentVersion    :
MaxConcurrency     :
MaxErrors          :
Name               : AWS-GatherSoftwareInventory
OutputLocation     :
Parameters         : {}
ScheduleExpression :
Targets            : {InstanceIds}

AssociationId      : 123a45a0-c678-9012-3456-78901234db5e
AssociationName    : test-case-1234567890
AssociationVersion : 1
ComplianceSeverity :
CreatedDate        : 3/2/2019 8:53:29 AM
DocumentVersion    :
MaxConcurrency     :
MaxErrors          :
Name               : AWS-GatherSoftwareInventory
OutputLocation     :
Parameters         : {}
ScheduleExpression : rate(30minutes)
Targets            : {InstanceIds}
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 ListAssociationVersions。

Tools for PowerShell V5

示例 1：此示例检索所提供关联的所有版本。

```
Get-SSMAssociationVersionList -AssociationId 123a45a0-c678-9012-3456-78901234db5e
```

输出：

```
AssociationId      : 123a45a0-c678-9012-3456-78901234db5e
AssociationName    :
AssociationVersion : 2
```

```
ComplianceSeverity :
CreatedDate        : 3/12/2019 9:21:01 AM
DocumentVersion    :
MaxConcurrency     :
MaxErrors          :
Name               : AWS-GatherSoftwareInventory
OutputLocation     :
Parameters         : {}
ScheduleExpression :
Targets            : {InstanceIds}

AssociationId      : 123a45a0-c678-9012-3456-78901234db5e
AssociationName    : test-case-1234567890
AssociationVersion : 1
ComplianceSeverity :
CreatedDate        : 3/2/2019 8:53:29 AM
DocumentVersion    :
MaxConcurrency     :
MaxErrors          :
Name               : AWS-GatherSoftwareInventory
OutputLocation     :
Parameters         : {}
ScheduleExpression : rate(30minutes)
Targets            : {InstanceIds}
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 ListAssociationVersions。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **ListAssociations** 与 CLI 配合使用

以下代码示例演示如何使用 `ListAssociations`。

CLI

AWS CLI

示例 1：列出特定实例的关联

以下 list-associations 示例列出具有 AssociationName、UpdateSSMAgent 的所有关联。

```
aws ssm list-associations /
    --association-filter-list "key=AssociationName,value=UpdateSSMAgent"
```

输出：

```
{
    "Associations": [
        {
            "Name": "AWS-UpdateSSMAgent",
            "InstanceId": "i-1234567890abcdef0",
            "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
            "AssociationVersion": "1",
            "Targets": [
                {
                    "Key": "InstanceIds",
                    "Values": [
                        "i-016648b75dd622dab"
                    ]
                }
            ],
            "Overview": {
                "Status": "Pending",
                "DetailedStatus": "Associated",
                "AssociationStatusAggregatedCount": {
                    "Pending": 1
                }
            },
            "ScheduleExpression": "cron(0 00 12 ? * SUN *)",
            "AssociationName": "UpdateSSMAgent"
        }
    ]
}
```

有关更多信息，请参阅《Systems Manager 用户指南》中的在 Systems Manager 中使用关联。

示例 2：列出特定文档的关联

以下 list-associations 示例列出指定文档的所有关联。

```
aws ssm list-associations /
    --association-filter-list "key=Name,value=AWS-UpdateSSMAgent"
```

输出：

```
{
    "Associations": [
        {
            "Name": "AWS-UpdateSSMAgent",
            "InstanceId": "i-1234567890abcdef0",
            "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
            "AssociationVersion": "1",
            "Targets": [
                {
                    "Key": "InstanceIds",
                    "Values": [
                        "i-1234567890abcdef0"
                    ]
                }
            ],
            "LastExecutionDate": 1550505828.548,
            "Overview": {
                "Status": "Success",
                "DetailedStatus": "Success",
                "AssociationStatusAggregatedCount": {
                    "Success": 1
                }
            },
            "ScheduleExpression": "cron(0 00 12 ? * SUN *)",
            "AssociationName": "UpdateSSMAgent"
        },
    {
            "Name": "AWS-UpdateSSMAgent",
            "InstanceId": "i-9876543210abcdef0",
            "AssociationId": "fbc07ef7-b985-4684-b82b-0123456789ab",
            "AssociationVersion": "1",
            "Targets": [
                {
                    "Key": "InstanceIds",
                    "Values": [
                        "i-9876543210abcdef0"
                    ]
                }
            ],
            "LastExecutionDate": 1550507531.0,
            "Overview": {
                "Status": "Success",
```

```
                "AssociationStatusAggregatedCount": {
                    "Success": 1
                }
            }
        }
    ]
}
```

有关更多信息，请参阅《Systems Manager 用户指南》中的在 Systems Manager 中使用关联。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 ListAssociations。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出实例的所有关联。此示例使用的语法需要 PowerShell 版本 3 或更高版本。

```
$filter1 = @{Key="InstanceId";Value=@("i-0000293ffd8c57862")}
Get-SSMAssociationList -AssociationFilterList $filter1
```

输出：

```
AssociationId      : d8617c07-2079-4c18-9847-1655fc2698b0
DocumentVersion    :
InstanceId         : i-0000293ffd8c57862
LastExecutionDate  : 2/20/2015 8:31:11 AM
Name               : AWS-UpdateSSMAgent
Overview           : Amazon.SimpleSystemsManagement.Model.AssociationOverview
ScheduleExpression :
Targets            : {InstanceIds}
```

示例 2：此示例列出配置文档的所有关联。此示例使用的语法需要 PowerShell 版本 3 或更高版本。

```
$filter2 = @{Key="Name";Value=@("AWS-UpdateSSMAgent")}
Get-SSMAssociationList -AssociationFilterList $filter2
```

输出：

```
AssociationId      : d8617c07-2079-4c18-9847-1655fc2698b0
```

```
DocumentVersion    :
InstanceId         : i-0000293ffd8c57862
LastExecutionDate  : 2/20/2015 8:31:11 AM
Name               : AWS-UpdateSSMAgent
Overview           : Amazon.SimpleSystemsManagement.Model.AssociationOverview
ScheduleExpression :
Targets            : {InstanceIds}
```

示例 3：对于 PowerShell 版本 2，必须使用 New-Object 创建每个筛选器。

```
$filter1 = New-Object Amazon.SimpleSystemsManagement.Model.AssociationFilter
$filter1.Key = "InstanceId"
$filter1.Value = "i-0000293ffd8c57862"

Get-SSMAssociationList -AssociationFilterList $filter1
```

输出：

```
AssociationId      : d8617c07-2079-4c18-9847-1655fc2698b0
DocumentVersion    :
InstanceId         : i-0000293ffd8c57862
LastExecutionDate  : 2/20/2015 8:31:11 AM
Name               : AWS-UpdateSSMAgent
Overview           : Amazon.SimpleSystemsManagement.Model.AssociationOverview
ScheduleExpression :
Targets            : {InstanceIds}
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 ListAssociations。

Tools for PowerShell V5

示例 1：此示例列出实例的所有关联。此示例使用的语法需要 PowerShell 版本 3 或更高版本。

```
$filter1 = @{Key="InstanceId";Value=@("i-0000293ffd8c57862")}
Get-SSMAssociationList -AssociationFilterList $filter1
```

输出：

```
AssociationId      : d8617c07-2079-4c18-9847-1655fc2698b0
DocumentVersion    :
InstanceId         : i-0000293ffd8c57862
```

```
LastExecutionDate  : 2/20/2015 8:31:11 AM
Name               : AWS-UpdateSSMAgent
Overview           : Amazon.SimpleSystemsManagement.Model.AssociationOverview
ScheduleExpression :
Targets            : {InstanceIds}
```

示例 2：此示例列出配置文档的所有关联。此示例使用的语法需要 PowerShell 版本 3 或更高版本。

```
$filter2 = @{Key="Name";Value=@("AWS-UpdateSSMAgent")}
Get-SSMAssociationList -AssociationFilterList $filter2
```

输出：

```
AssociationId      : d8617c07-2079-4c18-9847-1655fc2698b0
DocumentVersion    :
InstanceId         : i-0000293ffd8c57862
LastExecutionDate  : 2/20/2015 8:31:11 AM
Name               : AWS-UpdateSSMAgent
Overview           : Amazon.SimpleSystemsManagement.Model.AssociationOverview
ScheduleExpression :
Targets            : {InstanceIds}
```

示例 3：对于 PowerShell 版本 2，必须使用 New-Object 创建每个筛选器。

```
$filter1 = New-Object Amazon.SimpleSystemsManagement.Model.AssociationFilter
$filter1.Key = "InstanceId"
$filter1.Value = "i-0000293ffd8c57862"

Get-SSMAssociationList -AssociationFilterList $filter1
```

输出：

```
AssociationId      : d8617c07-2079-4c18-9847-1655fc2698b0
DocumentVersion    :
InstanceId         : i-0000293ffd8c57862
LastExecutionDate  : 2/20/2015 8:31:11 AM
Name               : AWS-UpdateSSMAgent
Overview           : Amazon.SimpleSystemsManagement.Model.AssociationOverview
ScheduleExpression :
Targets            : {InstanceIds}
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 ListAssociations。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 `ListCommandInvocations` 与 AWS SDK 或 CLI 配合使用

以下代码示例演示如何使用 `ListCommandInvocations`。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 了解基础知识

CLI

AWS CLI

列出特定命令的调用

以下 `list-command-invocations` 示例列出命令的所有调用。

```
aws ssm list-command-invocations \
    --command-id "ef7fdfd8-9b57-4151-a15c-db9a12345678" \
    --details
```

输出：

```
{
    "CommandInvocations": [
        {
            "CommandId": "ef7fdfd8-9b57-4151-a15c-db9a12345678",
            "InstanceId": "i-02573cafcfEXAMPLE",
            "InstanceName": "",
            "Comment": "b48291dd-ba76-43e0-
b9df-13e11ddaac26:6960febb-2907-4b59-8e1a-d6ce8EXAMPLE",
            "DocumentName": "AWS-UpdateSSMAgent",
            "DocumentVersion": "",
            "RequestedDateTime": 1582136283.089,
            "Status": "Success",
```

```
            "StatusDetails": "Success",
            "StandardOutputUrl": "",
            "StandardErrorUrl": "",
            "CommandPlugins": [
                {
                    "Name": "aws:updateSsmAgent",
                    "Status": "Success",
                    "StatusDetails": "Success",
                    "ResponseCode": 0,
                    "ResponseStartDateTime": 1582136283.419,
                    "ResponseFinishDateTime": 1582136283.51,
                    "Output": "Updating amazon-ssm-agent from 2.3.842.0 to latest
\nSuccessfully downloaded https://s3.us-east-2.amazonaws.com/amazon-ssm-us-
east-2/ssm-agent-manifest.json\namazon-ssm-agent 2.3.842.0 has already been
 installed, update skipped\n",
                    "StandardOutputUrl": "",
                    "StandardErrorUrl": "",
                    "OutputS3Region": "us-east-2",
                    "OutputS3BucketName": "",
                    "OutputS3KeyPrefix": ""
                }
            ],
            "ServiceRole": "",
            "NotificationConfig": {
                "NotificationArn": "",
                "NotificationEvents": [],
                "NotificationType": ""
            },
            "CloudWatchOutputConfig": {
                "CloudWatchLogGroupName": "",
                "CloudWatchOutputEnabled": false
            }
        },
        {
            "CommandId": "ef7fdfd8-9b57-4151-a15c-db9a12345678",
            "InstanceId": "i-0471e04240EXAMPLE",
            "InstanceName": "",
            "Comment": "b48291dd-ba76-43e0-
b9df-13e11ddaac26:6960febb-2907-4b59-8e1a-d6ce8EXAMPLE",
            "DocumentName": "AWS-UpdateSSMAgent",
            "DocumentVersion": "",
            "RequestedDateTime": 1582136283.02,
            "Status": "Success",
            "StatusDetails": "Success",
```

```
            "StandardOutputUrl": "",
            "StandardErrorUrl": "",
            "CommandPlugins": [
                {
                    "Name": "aws:updateSsmAgent",
                    "Status": "Success",
                    "StatusDetails": "Success",
                    "ResponseCode": 0,
                    "ResponseStartDateTime": 1582136283.812,
                    "ResponseFinishDateTime": 1582136295.031,
                    "Output": "Updating amazon-ssm-agent from 2.3.672.0 to
 latest\nSuccessfully downloaded https://s3.us-east-2.amazonaws.com/amazon-
ssm-us-east-2/ssm-agent-manifest.json\nSuccessfully downloaded https://s3.us-
east-2.amazonaws.com/amazon-ssm-us-east-2/amazon-ssm-agent-updater/2.3.842.0/
amazon-ssm-agent-updater-snap-amd64.tar.gz\nSuccessfully downloaded https://
s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/amazon-ssm-agent/2.3.672.0/
amazon-ssm-agent-snap-amd64.tar.gz\nSuccessfully downloaded https://s3.us-
east-2.amazonaws.com/amazon-ssm-us-east-2/amazon-ssm-agent/2.3.842.0/amazon-ssm-
agent-snap-amd64.tar.gz\nInitiating amazon-ssm-agent update to 2.3.842.0\namazon-
ssm-agent updated successfully to 2.3.842.0",
                    "StandardOutputUrl": "",
                    "StandardErrorUrl": "",
                    "OutputS3Region": "us-east-2",
                    "OutputS3BucketName": "",
                    "OutputS3KeyPrefix": "8bee3135-398c-4d31-99b6-e42d2EXAMPLE/
i-0471e04240EXAMPLE/awsupdateSsmAgent"
                }
            ],
            "ServiceRole": "",
            "NotificationConfig": {
                "NotificationArn": "",
                "NotificationEvents": [],
                "NotificationType": ""
            },
            "CloudWatchOutputConfig": {
                "CloudWatchLogGroupName": "",
                "CloudWatchOutputEnabled": false
            }
        }
    ]
}
```


有关更多信息，请参阅《AWS Systems Manager 用户指南》中的了解命令状态。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 ListCommandInvocations。

JavaScript

适用于 JavaScript 的 SDK (v3)

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import { paginateListCommandInvocations, SSMClient } from "@aws-sdk/client-ssm";
import { parseArgs } from "node:util";

/**
 * List SSM command invocations on an instance.
 * @param {{ instanceId: string }}
 */
export const main = async ({ instanceId }) => {
  const client = new SSMClient({});
  try {
    const listCommandInvocationsPaginated = [];
    // The paginate function is a wrapper around the base command.
    const paginator = paginateListCommandInvocations(
      { client },
      {
        InstanceId: instanceId,
      },
    );
    for await (const page of paginator) {
      listCommandInvocationsPaginated.push(...page.CommandInvocations);
    }
    console.log("Here is the list of command invocations:");
    console.log(listCommandInvocationsPaginated);
    return { CommandInvocations: listCommandInvocationsPaginated };
  } catch (caught) {
    if (caught instanceof Error && caught.name === "ValidationError") {
      console.warn(`${caught.message}. Did you provide a valid instance ID?`);
    }
```

```
    throw caught;
  }
};
```

- 有关 API 详细信息，请参阅《适用于 JavaScript 的 AWS SDK API 参考》中的 ListCommandInvocations。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出命令的所有调用。

```
Get-SSMCommandInvocation -CommandId "b8eac879-0541-439d-94ec-47a80d554f44" -
Detail $true
```

输出：

```
CommandId          : b8eac879-0541-439d-94ec-47a80d554f44
CommandPlugins     : {aws:runShellScript}
Comment            : IP config
DocumentName       : AWS-RunShellScript
InstanceId         : i-0cb2b964d3e14fd9f
InstanceName       :
NotificationConfig : Amazon.SimpleSystemsManagement.Model.NotificationConfig
RequestedDateTime  : 2/22/2017 8:13:16 PM
ServiceRole        :
StandardErrorUrl   :
StandardOutputUrl  :
Status             : Success
StatusDetails      : Success
TraceOutput        :
```

示例 2：此示例列出用于调用命令 ID e1eb2e3c-ed4c-5123-45c1-234f5612345f 的 CommandPlugins

```
Get-SSMCommandInvocation -CommandId e1eb2e3c-ed4c-5123-45c1-234f5612345f -Detail:
$true | Select-Object -ExpandProperty CommandPlugins
```

输出：

```
Name                   : aws:runPowerShellScript
Output                 : Completed 17.7 KiB/17.7 KiB (40.1 KiB/s) with 1 file(s)
 remainingdownload: s3://dd-aess-r-ctmer/KUMO.png to ..\..\programdata\KUMO.png
                          kumo available

OutputS3BucketName     :
OutputS3KeyPrefix      :
OutputS3Region         : eu-west-1
ResponseCode           : 0
ResponseFinishDateTime : 4/3/2019 11:53:23 AM
ResponseStartDateTime  : 4/3/2019 11:53:21 AM
StandardErrorUrl       :
StandardOutputUrl      :
Status                 : Success
StatusDetails          : Success
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 ListCommandInvocations。

Tools for PowerShell V5

示例 1：此示例列出命令的所有调用。

```
Get-SSMCommandInvocation -CommandId "b8eac879-0541-439d-94ec-47a80d554f44" -
Detail $true
```

输出：

```
CommandId          : b8eac879-0541-439d-94ec-47a80d554f44
CommandPlugins     : {aws:runShellScript}
Comment            : IP config
DocumentName       : AWS-RunShellScript
InstanceId         : i-0cb2b964d3e14fd9f
InstanceName       :
NotificationConfig : Amazon.SimpleSystemsManagement.Model.NotificationConfig
RequestedDateTime  : 2/22/2017 8:13:16 PM
ServiceRole        :
StandardErrorUrl   :
StandardOutputUrl  :
Status             : Success
StatusDetails      : Success
TraceOutput        :
```

示例 2：此示例列出用于调用命令 ID e1eb2e3c-ed4c-5123-45c1-234f5612345f 的 CommandPlugins

```
Get-SSMCommandInvocation -CommandId e1eb2e3c-ed4c-5123-45c1-234f5612345f -Detail:
$true | Select-Object -ExpandProperty CommandPlugins
```

输出：

```
Name                   : aws:runPowerShellScript
Output                 : Completed 17.7 KiB/17.7 KiB (40.1 KiB/s) with 1 file(s)
 remainingdownload: s3://dd-aess-r-ctmer/KUMO.png to ..\..\programdata\KUMO.png
                        kumo available

OutputS3BucketName     :
OutputS3KeyPrefix      :
OutputS3Region         : eu-west-1
ResponseCode           : 0
ResponseFinishDateTime : 4/3/2019 11:53:23 AM
ResponseStartDateTime  : 4/3/2019 11:53:21 AM
StandardErrorUrl       :
StandardOutputUrl      :
Status                 : Success
StatusDetails          : Success
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 ListCommandInvocations。

Python

适用于 Python 的 SDK (Boto3)

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
class DocumentWrapper:
    """Encapsulates AWS Systems Manager Document actions."""
```

```
    def __init__(self, ssm_client):
        """
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.name = None

    @classmethod
    def from_client(cls):
        ssm_client = boto3.client("ssm")
        return cls(ssm_client)


    def list_command_invocations(self, instance_id):
        """
        Lists the commands for an instance.

        :param instance_id: The ID of the instance.
        :return: The list of commands.
        """
        try:
            paginator = self.ssm_client.get_paginator("list_command_invocations")
            command_invocations = []
            for page in paginator.paginate(InstanceId=instance_id):
                command_invocations.extend(page["CommandInvocations"])
            num_of_commands = len(command_invocations)
            print(
                f"{num_of_commands} command invocation(s) found for instance
{instance_id}."
            )

            if num_of_commands > 10:
                print("Displaying the first 10 commands:")
                num_of_commands = 10
            date_format = "%A, %d %B %Y %I:%M%p"
            for command in command_invocations[:num_of_commands]:
                print(
                    f"   The time of command invocation is
{command['RequestedDateTime'].strftime(date_format)}"
                )
        except ClientError as err:
            logger.error(
                "Couldn't list commands for %s. Here's why: %s: %s",
                instance_id,
```

```
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 有关 API 详细信息，请参阅《AWS SDK for Python (Boto3) API 参考》中的 ListCommandInvocations。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **ListCommands** 与 CLI 配合使用

以下代码示例演示如何使用 ListCommands。

CLI

AWS CLI

示例 1：获取特定命令的状态

以下 list-commands 示例检索并显示指定命令的状态。

```
aws ssm list-commands \
    --command-id "0831e1a8-a1ac-4257-a1fd-c831bEXAMPLE"
```

示例 2：获取特定日期之后请求的命令的状态

以下 list-commands 示例检索在指定日期之后请求的命令的详细信息。

```
aws ssm list-commands \
    --filter "key=InvokedAfter,value=2020-02-01T00:00:00Z"
```

示例 3：列出 AWS 账户中请求的所有命令

以下 list-commands 示例列出了当前 AWS 账户和区域中用户请求的所有命令。

```
aws ssm list-commands
```

输出：


```
{
    "Commands": [
        {
            "CommandId": "8bee3135-398c-4d31-99b6-e42d2EXAMPLE",
            "DocumentName": "AWS-UpdateSSMAgent",
            "DocumentVersion": "",
            "Comment": "b48291dd-ba76-43e0-
b9df-13e11ddaac26:6960febb-2907-4b59-8e1a-d6ce8EXAMPLE",
            "ExpiresAfter": "2020-02-19T11:28:02.500000-08:00",
            "Parameters": {},
            "InstanceIds": [
                "i-028ea792daEXAMPLE",
                "i-02feef8c46EXAMPLE",
                "i-038613f3f0EXAMPLE",
                "i-03a530a2d4EXAMPLE",
                "i-083b678d37EXAMPLE",
                "i-0dee81debaEXAMPLE"
            ],
            "Targets": [],
            "RequestedDateTime": "2020-02-19T10:18:02.500000-08:00",
            "Status": "Success",
            "StatusDetails": "Success",
            "OutputS3BucketName": "",
            "OutputS3KeyPrefix": "",
            "MaxConcurrency": "50",
            "MaxErrors": "100%",
            "TargetCount": 6,
            "CompletedCount": 6,
            "ErrorCount": 0,
            "DeliveryTimedOutCount": 0,
            "ServiceRole": "",
            "NotificationConfig": {
                "NotificationArn": "",
                "NotificationEvents": [],
                "NotificationType": ""
            },
            "CloudWatchOutputConfig": {
                "CloudWatchLogGroupName": "",
                "CloudWatchOutputEnabled": false
            }
        }
        {
```

```
            "CommandId": "e9ade581-c03d-476b-9b07-26667EXAMPLE",
            "DocumentName": "AWS-FindWindowsUpdates",
            "DocumentVersion": "1",
            "Comment": "",
            "ExpiresAfter": "2020-01-24T12:37:31.874000-08:00",
            "Parameters": {
                "KbArticleIds": [
                    ""
                ],
                "UpdateLevel": [
                    "All"
                ]
            },
            "InstanceIds": [],
            "Targets": [
                {
                    "Key": "InstanceIds",
                    "Values": [
                        "i-00ec29b21eEXAMPLE",
                        "i-09911ddd90EXAMPLE"
                    ]
                }
            ],
            "RequestedDateTime": "2020-01-24T11:27:31.874000-08:00",
            "Status": "Success",
            "StatusDetails": "Success",
            "OutputS3BucketName": "my-us-east-2-bucket",
            "OutputS3KeyPrefix": "my-rc-output",
            "MaxConcurrency": "50",
            "MaxErrors": "0",
            "TargetCount": 2,
            "CompletedCount": 2,
            "ErrorCount": 0,
            "DeliveryTimedOutCount": 0,
            "ServiceRole": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
            "NotificationConfig": {
                "NotificationArn": "arn:aws:sns:us-east-2:111222333444:my-us-
east-2-notification-arn",
                "NotificationEvents": [
                    "All"
                ],
                "NotificationType": "Invocation"
            },
```

```
            "CloudWatchOutputConfig": {
                "CloudWatchLogGroupName": "",
                "CloudWatchOutputEnabled": false
            }
        }
        {
            "CommandId": "d539b6c3-70e8-4853-80e5-0ce4fEXAMPLE",
            "DocumentName": "AWS-RunPatchBaseline",
            "DocumentVersion": "1",
            "Comment": "",
            "ExpiresAfter": "2020-01-24T12:21:04.350000-08:00",
            "Parameters": {
                "InstallOverrideList": [
                    ""
                ],
                "Operation": [
                    "Install"
                ],
                "RebootOption": [
                    "RebootIfNeeded"
                ],
                "SnapshotId": [
                    ""
                ]
            },
            "InstanceIds": [],
            "Targets": [
                {
                    "Key": "InstanceIds",
                    "Values": [
                        "i-00ec29b21eEXAMPLE",
                        "i-09911ddd90EXAMPLE"
                    ]
                }
            ],
            "RequestedDateTime": "2020-01-24T11:11:04.350000-08:00",
            "Status": "Success",
            "StatusDetails": "Success",
            "OutputS3BucketName": "my-us-east-2-bucket",
            "OutputS3KeyPrefix": "my-rc-output",
            "MaxConcurrency": "50",
            "MaxErrors": "0",
            "TargetCount": 2,
            "CompletedCount": 2,
```

```
            "ErrorCount": 0,
            "DeliveryTimedOutCount": 0,
            "ServiceRole": "arn:aws:iam::111222333444:role/aws-service-role/
ssm.amazonaws.com/AWSServiceRoleForAmazonSSM",
            "NotificationConfig": {
                "NotificationArn": "arn:aws:sns:us-east-2:111222333444:my-us-
east-2-notification-arn",
                "NotificationEvents": [
                    "All"
                ],
                "NotificationType": "Invocation"
            },
            "CloudWatchOutputConfig": {
                "CloudWatchLogGroupName": "",
                "CloudWatchOutputEnabled": false
            }
        }
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的使用 Systems Manager Run Command 运行命令。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 ListCommands。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出请求的所有命令。

```
Get-SSMCommand
```

输出：

```
CommandId          : 4b75a163-d39a-4d97-87c9-98ae52c6be35
Comment            : Apply association with id at update time: 4cc73e42-
d5ae-4879-84f8-57e09c0efcd0
CompletedCount     : 1
DocumentName       : AWS-RefreshAssociation
ErrorCount         : 0
ExpiresAfter       : 2/24/2017 3:19:08 AM
```

```
InstanceIds        : {i-0cb2b964d3e14fd9f}
MaxConcurrency     : 50
MaxErrors          : 0
NotificationConfig : Amazon.SimpleSystemsManagement.Model.NotificationConfig
OutputS3BucketName :
OutputS3KeyPrefix  :
OutputS3Region     :
Parameters         : {[associationIds,
 Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
RequestedDateTime  : 2/24/2017 3:18:08 AM
ServiceRole        :
Status             : Success
StatusDetails      : Success
TargetCount        : 1
Targets            : {}
```

示例 2：此示例获取特定命令的状态。

```
Get-SSMCommand -CommandId "4b75a163-d39a-4d97-87c9-98ae52c6be35"
```

示例 3：此示例检索 2019-04-01T00:00:00Z 之后调用的所有 SSM 命令

```
Get-SSMCommand -Filter @{Key="InvokedAfter";Value="2019-04-01T00:00:00Z"} |
 Select-Object CommandId, DocumentName, Status, RequestedDateTime | Sort-Object -
Property RequestedDateTime -Descending
```

输出：

```
CommandId                            DocumentName               Status
 RequestedDateTime
---------                            ------------               ------
 -----------------
edb1b23e-456a-7adb-aef8-90e-012ac34f AWS-RunPowerShellScript    Cancelled
 4/16/2019 5:45:23 AM
1a2dc3fb-4567-890d-a1ad-234b5d6bc7d9 AWS-ConfigureAWSPackage    Success
 4/6/2019 9:19:42 AM
12c3456c-7e90-4f12-1232-1234f5b67893 KT-Retrieve-Cloud-Type-Win Failed
 4/2/2019 4:13:07 AM
fe123b45-240c-4123-a2b3-234bdd567ecf AWS-RunInspecChecks        Failed
 4/1/2019 2:27:31 PM
1eb23aa4-567d-4123-12a3-4c1c2ab34561 AWS-RunPowerShellScript    Success
 4/1/2019 1:05:55 PM
```

```
1c2f3bb4-ee12-4bc1-1a23-12345eea123e AWS-RunInspecChecks        Failed
 4/1/2019 11:13:09 AM
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 ListCommands。

Tools for PowerShell V5

示例 1：此示例列出请求的所有命令。

```
Get-SSMCommand
```

输出：

```
CommandId          : 4b75a163-d39a-4d97-87c9-98ae52c6be35
Comment            : Apply association with id at update time: 4cc73e42-
d5ae-4879-84f8-57e09c0efcd0
CompletedCount     : 1
DocumentName       : AWS-RefreshAssociation
ErrorCount         : 0
ExpiresAfter       : 2/24/2017 3:19:08 AM
InstanceIds        : {i-0cb2b964d3e14fd9f}
MaxConcurrency     : 50
MaxErrors          : 0
NotificationConfig : Amazon.SimpleSystemsManagement.Model.NotificationConfig
OutputS3BucketName :
OutputS3KeyPrefix  :
OutputS3Region     :
Parameters         : {[associationIds,
 Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
RequestedDateTime  : 2/24/2017 3:18:08 AM
ServiceRole        :
Status             : Success
StatusDetails      : Success
TargetCount        : 1
Targets            : {}
```

示例 2：此示例获取特定命令的状态。

```
Get-SSMCommand -CommandId "4b75a163-d39a-4d97-87c9-98ae52c6be35"
```

示例 3：此示例检索 2019-04-01T00:00:00Z 之后调用的所有 SSM 命令

```
Get-SSMCommand -Filter @{Key="InvokedAfter";Value="2019-04-01T00:00:00Z"} |
 Select-Object CommandId, DocumentName, Status, RequestedDateTime | Sort-Object -
Property RequestedDateTime -Descending
```

输出：

```
CommandId                            DocumentName               Status
 RequestedDateTime
---------                            ------------               ------
 -----------------
edb1b23e-456a-7adb-aef8-90e-012ac34f AWS-RunPowerShellScript    Cancelled
 4/16/2019 5:45:23 AM
1a2dc3fb-4567-890d-a1ad-234b5d6bc7d9 AWS-ConfigureAWSPackage    Success
 4/6/2019 9:19:42 AM
12c3456c-7e90-4f12-1232-1234f5b67893 KT-Retrieve-Cloud-Type-Win Failed
 4/2/2019 4:13:07 AM
fe123b45-240c-4123-a2b3-234bdd567ecf AWS-RunInspecChecks        Failed
 4/1/2019 2:27:31 PM
1eb23aa4-567d-4123-12a3-4c1c2ab34561 AWS-RunPowerShellScript    Success
 4/1/2019 1:05:55 PM
1c2f3bb4-ee12-4bc1-1a23-12345eea123e AWS-RunInspecChecks        Failed
 4/1/2019 11:13:09 AM
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 ListCommands。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **ListComplianceItems** 与 CLI 配合使用

以下代码示例演示如何使用 `ListComplianceItems`。

CLI

AWS CLI

列出特定实例的合规性项目

此示例列出指定实例的所有合规性项目。

命令:

```
aws ssm list-compliance-items --resource-ids "i-1234567890abcdef0" --resource-types "ManagedInstance"
```

输出：

```
{
  "ComplianceItems": [
      {
          "ComplianceType": "Association",
          "ResourceType": "ManagedInstance",
          "ResourceId": "i-1234567890abcdef0",
          "Id": "8dfe3659-4309-493a-8755-0123456789ab",
          "Title": "",
          "Status": "COMPLIANT",
          "Severity": "UNSPECIFIED",
          "ExecutionSummary": {
              "ExecutionTime": 1550408470.0
          },
          "Details": {
              "DocumentName": "AWS-GatherSoftwareInventory",
              "DocumentVersion": "1"
          }
      },
      {
          "ComplianceType": "Association",
          "ResourceType": "ManagedInstance",
          "ResourceId": "i-1234567890abcdef0",
          "Id": "e4c2ed6d-516f-41aa-aa2a-0123456789ab",
          "Title": "",
          "Status": "COMPLIANT",
          "Severity": "UNSPECIFIED",
          "ExecutionSummary": {
              "ExecutionTime": 1550508475.0
          },
          "Details": {
              "DocumentName": "AWS-UpdateSSMAgent",
              "DocumentVersion": "1"
          }
      },
              ...
  ],
  "NextToken": "--token string truncated--"
```

```
}
```

列出特定实例和关联 ID 的合规性项目

此示例列出指定实例和关联 ID 的所有合规性项目。

命令:

```
aws ssm list-compliance-items --resource-ids "i-1234567890abcdef0" --resource-
types "ManagedInstance" --
filters "Key=ComplianceType,Values=Association,Type=EQUAL" "Key=Id,Values=e4c2ed6d-516f-4
aa2a-0123456789ab,Type=EQUAL"
```

列出特定日期和时间之后实例的合规性项目

此示例列出指定日期和时间之后实例的所有合规性项目。

命令:

```
aws ssm list-compliance-items --resource-ids "i-1234567890abcdef0" --resource-
types "ManagedInstance" --
filters "Key=ExecutionTime,Values=2019-02-18T16:00:00Z,Type=GREATER_THAN"
```

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 ListComplianceItems。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出给定资源 ID 和类型的合规性项目列表，筛选合规性类型为“关联”

```
Get-SSMComplianceItemList -ResourceId i-1a2caf345f67d0dc2 -ResourceType
 ManagedInstance -Filter @{Key="ComplianceType";Values="Association"}
```

输出：

```
ComplianceType   : Association
Details          : {[DocumentName, AWS-GatherSoftwareInventory],
 [DocumentVersion, 1]}
```

```
ExecutionSummary :
 Amazon.SimpleSystemsManagement.Model.ComplianceExecutionSummary
Id                : 123a45a1-c234-1234-1245-67891236db4e
ResourceId        : i-1a2caf345f67d0dc2
ResourceType      : ManagedInstance
Severity          : UNSPECIFIED
Status            : COMPLIANT
Title             :
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 ListComplianceItems。

Tools for PowerShell V5

示例 1：此示例列出给定资源 ID 和类型的合规性项目列表，筛选合规性类型为“关联”

```
Get-SSMComplianceItemList -ResourceId i-1a2caf345f67d0dc2 -ResourceType
 ManagedInstance -Filter @{Key="ComplianceType";Values="Association"}
```

输出：

```
ComplianceType    : Association
Details           : {[DocumentName, AWS-GatherSoftwareInventory],
 [DocumentVersion, 1]}
ExecutionSummary :
 Amazon.SimpleSystemsManagement.Model.ComplianceExecutionSummary
Id                : 123a45a1-c234-1234-1245-67891236db4e
ResourceId        : i-1a2caf345f67d0dc2
ResourceType      : ManagedInstance
Severity          : UNSPECIFIED
Status            : COMPLIANT
Title             :
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 ListComplianceItems。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **ListComplianceSummaries** 与 CLI 配合使用

以下代码示例演示如何使用 ListComplianceSummaries。

CLI

AWS CLI

列出所有合规性类型的合规性摘要

此示例列出您账户中所有合规性类型的合规性摘要。

命令:

```
aws ssm list-compliance-summaries
```

输出：

```
{
  "ComplianceSummaryItems": [
      {
          "ComplianceType": "Association",
          "CompliantSummary": {
              "CompliantCount": 2,
              "SeveritySummary": {
                  "CriticalCount": 0,
                  "HighCount": 0,
                  "MediumCount": 0,
                  "LowCount": 0,
                  "InformationalCount": 0,
                  "UnspecifiedCount": 2
              }
          },
          "NonCompliantSummary": {
              "NonCompliantCount": 0,
              "SeveritySummary": {
                  "CriticalCount": 0,
                  "HighCount": 0,
                  "MediumCount": 0,
                  "LowCount": 0,
                  "InformationalCount": 0,
                  "UnspecifiedCount": 0
              }
          }
      },
      {
          "ComplianceType": "Patch",
```

```
            "CompliantSummary": {
                "CompliantCount": 1,
                "SeveritySummary": {
                    "CriticalCount": 0,
                    "HighCount": 0,
                    "MediumCount": 0,
                    "LowCount": 0,
                    "InformationalCount": 0,
                    "UnspecifiedCount": 1
                }
            },
            "NonCompliantSummary": {
                "NonCompliantCount": 1,
                "SeveritySummary": {
                    "CriticalCount": 1,
                    "HighCount": 0,
                    "MediumCount": 0,
                    "LowCount": 0,
                    "InformationalCount": 0,
                    "UnspecifiedCount": 0
                }
            }
        },
                ...
    ],
    "NextToken": "eyJOZXh0VG9rZW4iOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiAyfQ=="
}
```

列出特定合规性类型的合规性摘要

此示例列出补丁合规性类型的合规性摘要。

命令:

```
aws ssm list-compliance-summaries --
filters "Key=ComplianceType,Values=Patch,Type=EQUAL"
```

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 ListComplianceSummaries。

PowerShell

Tools for PowerShell V4

示例 1：此示例返回所有合规性类型的合规和不合规资源的摘要数。

```
Get-SSMComplianceSummaryList
```

输出：

```
ComplianceType CompliantSummary
 NonCompliantSummary
-------------- ---------------
 ------------------
FleetTotal     Amazon.SimpleSystemsManagement.Model.CompliantSummary
 Amazon.SimpleSystemsManagement.Model.NonCompliantSummary
Association    Amazon.SimpleSystemsManagement.Model.CompliantSummary
 Amazon.SimpleSystemsManagement.Model.NonCompliantSummary
Custom:InSpec  Amazon.SimpleSystemsManagement.Model.CompliantSummary
 Amazon.SimpleSystemsManagement.Model.NonCompliantSummary
Patch          Amazon.SimpleSystemsManagement.Model.CompliantSummary
 Amazon.SimpleSystemsManagement.Model.NonCompliantSummary
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 ListComplianceSummaries。

Tools for PowerShell V5

示例 1：此示例返回所有合规性类型的合规和不合规资源的摘要数。

```
Get-SSMComplianceSummaryList
```

输出：

```
ComplianceType CompliantSummary
 NonCompliantSummary
-------------- ---------------
 ------------------
FleetTotal     Amazon.SimpleSystemsManagement.Model.CompliantSummary
 Amazon.SimpleSystemsManagement.Model.NonCompliantSummary
Association    Amazon.SimpleSystemsManagement.Model.CompliantSummary
 Amazon.SimpleSystemsManagement.Model.NonCompliantSummary
```

```
Custom:InSpec  Amazon.SimpleSystemsManagement.Model.CompliantSummary
 Amazon.SimpleSystemsManagement.Model.NonCompliantSummary
Patch          Amazon.SimpleSystemsManagement.Model.CompliantSummary
 Amazon.SimpleSystemsManagement.Model.NonCompliantSummary
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 ListComplianceSummaries。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **ListDocumentVersions** 与 CLI 配合使用

以下代码示例演示如何使用 `ListDocumentVersions`。

CLI

AWS CLI

列出文档版本

以下 `list-document-versions` 示例列出 Systems Manager 文档的所有版本。

```
aws ssm list-document-versions \
    --name "Example"
```

输出：

```
{
    "DocumentVersions": [
        {
            "Name": "Example",
            "DocumentVersion": "1",
            "CreatedDate": 1583257938.266,
            "IsDefaultVersion": true,
            "DocumentFormat": "YAML",
            "Status": "Active"
        }
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的发送使用文档版本参数的命令。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 ListDocumentVersions。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出文档的所有版本。

```
Get-SSMDocumentVersionList -Name "AWS-UpdateSSMAgent"
```

输出：

```
CreatedDate       : 6/1/2021 5:19:10 PM
DocumentFormat    : JSON
DocumentVersion   : 1
IsDefaultVersion  : True
Name              : AWS-UpdateSSMAgent
Status            : Active
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 ListDocumentVersions。

Tools for PowerShell V5

示例 1：此示例列出文档的所有版本。

```
Get-SSMDocumentVersionList -Name "AWS-UpdateSSMAgent"
```

输出：

```
CreatedDate       : 6/1/2021 5:19:10 PM
DocumentFormat    : JSON
DocumentVersion   : 1
IsDefaultVersion  : True
Name              : AWS-UpdateSSMAgent
Status            : Active
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 ListDocumentVersions。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **ListDocuments** 与 CLI 配合使用

以下代码示例演示如何使用 ListDocuments。

CLI

AWS CLI

示例 1：列出文档

以下 list-documents 示例列出标有自定义标签的请求账户拥有的文档。

```
aws ssm list-documents \
    --filters Key=Owner,Values=Self Key=tag:DocUse,Values=Testing
```

输出：

```
{
    "DocumentIdentifiers": [
        {
            "Name": "Example",
            "Owner": "29884EXAMPLE",
            "PlatformTypes": [
                "Windows",
                "Linux"
            ],
            "DocumentVersion": "1",
            "DocumentType": "Automation",
            "SchemaVersion": "0.3",
            "DocumentFormat": "YAML",
            "Tags": [
                {
                    "Key": "DocUse",
                    "Value": "Testing"
                }
```

```
            ]
        }
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的 AWS Systems Manager 文档。

示例 2：列出共享文档

以下 `list-documents` 示例列出共享文档，包括不属于 AWS 的私有共享文档。

```
aws ssm list-documents \
    --filters Key=Name,Values=sharedDocNamePrefix  Key=Owner,Values=Private
```

输出：

```
{
    "DocumentIdentifiers": [
        {
            "Name": "Example",
            "Owner": "12345EXAMPLE",
            "PlatformTypes": [
                "Windows",
                "Linux"
            ],
            "DocumentVersion": "1",
            "DocumentType": "Command",
            "SchemaVersion": "0.3",
            "DocumentFormat": "YAML",
            "Tags": []
        }
    ]
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的 AWS Systems Manager 文档。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 ListDocuments。

PowerShell

Tools for PowerShell V4

示例 1：列出您账户中的所有配置文档。

```
Get-SSMDocumentList
```

输出：

```
DocumentType    : Command
DocumentVersion : 1
Name            : AWS-ApplyPatchBaseline
Owner           : Amazon
PlatformTypes   : {Windows}
SchemaVersion   : 1.2

DocumentType    : Command
DocumentVersion : 1
Name            : AWS-ConfigureAWSPackage
Owner           : Amazon
PlatformTypes   : {Windows, Linux}
SchemaVersion   : 2.0

DocumentType    : Command
DocumentVersion : 1
Name            : AWS-ConfigureCloudWatch
Owner           : Amazon
PlatformTypes   : {Windows}
SchemaVersion   : 1.2
...
```

示例 2：此示例检索名称与“Platform”匹配的所有自动化文档

```
Get-SSMDocumentList -DocumentFilterList @{Key="DocumentType";Value="Automation"}
 | Where-Object Name -Match "Platform"
```

输出：

```
DocumentFormat  : JSON
DocumentType    : Automation
```

```
DocumentVersion : 7
Name            : KT-Get-Platform
Owner           : 987654123456
PlatformTypes   : {Windows, Linux}
SchemaVersion   : 0.3
Tags            : {}
TargetType      :
VersionName     :
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 ListDocuments。

Tools for PowerShell V5

示例 1：列出您账户中的所有配置文档。

```
Get-SSMDocumentList
```

输出：

```
DocumentType    : Command
DocumentVersion : 1
Name            : AWS-ApplyPatchBaseline
Owner           : Amazon
PlatformTypes   : {Windows}
SchemaVersion   : 1.2

DocumentType    : Command
DocumentVersion : 1
Name            : AWS-ConfigureAWSPackage
Owner           : Amazon
PlatformTypes   : {Windows, Linux}
SchemaVersion   : 2.0

DocumentType    : Command
DocumentVersion : 1
Name            : AWS-ConfigureCloudWatch
Owner           : Amazon
PlatformTypes   : {Windows}
SchemaVersion   : 1.2
...
```

示例 2：此示例检索名称与“Platform”匹配的所有自动化文档

```
Get-SSMDocumentList -DocumentFilterList @{Key="DocumentType";Value="Automation"}
 | Where-Object Name -Match "Platform"
```

输出：

```
DocumentFormat  : JSON
DocumentType    : Automation
DocumentVersion : 7
Name            : KT-Get-Platform
Owner           : 987654123456
PlatformTypes   : {Windows, Linux}
SchemaVersion   : 0.3
Tags            : {}
TargetType      :
VersionName     :
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 ListDocuments。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **ListInventoryEntries** 与 CLI 配合使用

以下代码示例演示如何使用 ListInventoryEntries。

CLI

AWS CLI

示例 1：查看实例的特定清单类型条目

以下 list-inventory-entries 示例列出特定实例上 AWS:Application 清单类型的清单条目。

```
aws ssm list-inventory-entries \
    --instance-id "i-1234567890abcdef0" \
    --type-name "AWS:Application"
```

输出：

```
{
  "TypeName": "AWS:Application",
  "InstanceId": "i-1234567890abcdef0",
  "SchemaVersion": "1.1",
  "CaptureTime": "2019-02-15T12:17:55Z",
  "Entries": [
    {
      "Architecture": "i386",
      "Name": "Amazon SSM Agent",
      "PackageId": "{88a60be2-89a1-4df8-812a-80863c2a2b68}",
      "Publisher": "Amazon Web Services",
      "Version": "2.3.274.0"
    },
    {
      "Architecture": "x86_64",
      "InstalledTime": "2018-05-03T13:42:34Z",
      "Name": "AmazonCloudWatchAgent",
      "Publisher": "",
      "Version": "1.200442.0"
    }
  ]
}
```

示例 2：查看分配给实例的自定义清单条目

以下 `list-inventory-entries` 示例列出分配给实例的自定义清单条目。

```
aws ssm list-inventory-entries \
    --instance-id "i-1234567890abcdef0" \
    --type-name "Custom:RackInfo"
```

输出：

```
{
  "TypeName": "Custom:RackInfo",
  "InstanceId": "i-1234567890abcdef0",
  "SchemaVersion": "1.0",
  "CaptureTime": "2021-05-22T10:01:01Z",
  "Entries": [
    {
      "RackLocation": "Bay B/Row C/Rack D/Shelf E"
    }
```

```
  ]
}
```

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 ListInventoryEntries。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出实例的所有自定义清单条目。

```
Get-SSMInventoryEntriesList -InstanceId "i-0cb2b964d3e14fd9f" -TypeName
 "Custom:RackInfo"
```

输出：

```
CaptureTime   : 2016-08-22T10:01:01Z
Entries       :
 {Amazon.Runtime.Internal.Util.AlwaysSendDictionary`2[System.String,System.String]}
InstanceId    : i-0cb2b964d3e14fd9f
NextToken     :
SchemaVersion : 1.0
TypeName      : Custom:RackInfo
```

示例 2：此示例列出详细信息。

```
(Get-SSMInventoryEntriesList -InstanceId "i-0cb2b964d3e14fd9f" -TypeName
 "Custom:RackInfo").Entries
```

输出：

```
Key          Value
---          -----
RackLocation Bay B/Row C/Rack D/Shelf E
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 ListInventoryEntries。

Tools for PowerShell V5

示例 1：此示例列出实例的所有自定义清单条目。

```
Get-SSMInventoryEntriesList -InstanceId "i-0cb2b964d3e14fd9f" -TypeName
 "Custom:RackInfo"
```

输出：

```
CaptureTime   : 2016-08-22T10:01:01Z
Entries       :
 {Amazon.Runtime.Internal.Util.AlwaysSendDictionary`2[System.String,System.String]}
InstanceId    : i-0cb2b964d3e14fd9f
NextToken     :
SchemaVersion : 1.0
TypeName      : Custom:RackInfo
```

示例 2：此示例列出详细信息。

```
(Get-SSMInventoryEntriesList -InstanceId "i-0cb2b964d3e14fd9f" -TypeName
 "Custom:RackInfo").Entries
```

输出：

```
Key          Value
---          -----
RackLocation Bay B/Row C/Rack D/Shelf E
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 ListInventoryEntries。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **ListResourceComplianceSummaries** 与 CLI 配合使用

以下代码示例演示如何使用 `ListResourceComplianceSummaries`。

CLI

AWS CLI

列出资源级合规性摘要计数

此示例列出资源级合规性摘要计数。

命令:

```
aws ssm list-resource-compliance-summaries
```

输出：

```
{
  "ResourceComplianceSummaryItems": [
      {
          "ComplianceType": "Association",
          "ResourceType": "ManagedInstance",
          "ResourceId": "i-1234567890abcdef0",
          "Status": "COMPLIANT",
          "OverallSeverity": "UNSPECIFIED",
          "ExecutionSummary": {
              "ExecutionTime": 1550509273.0
          },
          "CompliantSummary": {
              "CompliantCount": 2,
              "SeveritySummary": {
                  "CriticalCount": 0,
                  "HighCount": 0,
                  "MediumCount": 0,
                  "LowCount": 0,
                  "InformationalCount": 0,
                  "UnspecifiedCount": 2
              }
          },
          "NonCompliantSummary": {
              "NonCompliantCount": 0,
              "SeveritySummary": {
                  "CriticalCount": 0,
                  "HighCount": 0,
                  "MediumCount": 0,
                  "LowCount": 0,
                  "InformationalCount": 0,
                  "UnspecifiedCount": 0
              }
          }
      },
      {
          "ComplianceType": "Patch",
          "ResourceType": "ManagedInstance",
```

```
            "ResourceId": "i-9876543210abcdef0",
            "Status": "COMPLIANT",
            "OverallSeverity": "UNSPECIFIED",
            "ExecutionSummary": {
                "ExecutionTime": 1550248550.0,
                "ExecutionId": "7abb6378-a4a5-4f10-8312-0123456789ab",
                "ExecutionType": "Command"
            },
            "CompliantSummary": {
                "CompliantCount": 397,
                "SeveritySummary": {
                    "CriticalCount": 0,
                    "HighCount": 0,
                    "MediumCount": 0,
                    "LowCount": 0,
                    "InformationalCount": 0,
                    "UnspecifiedCount": 397
                }
            },
            "NonCompliantSummary": {
                "NonCompliantCount": 0,
                "SeveritySummary": {
                    "CriticalCount": 0,
                    "HighCount": 0,
                    "MediumCount": 0,
                    "LowCount": 0,
                    "InformationalCount": 0,
                    "UnspecifiedCount": 0
                }
            }
        }
    ],
    "NextToken": "--token string truncated--"
}
```

列出特定合性规类型的资源级合规性摘要

此示例列出补丁合规性类型的资源级合规性摘要。

命令:

```
aws ssm list-resource-compliance-summaries --
filters "Key=ComplianceType,Values=Patch,Type=EQUAL"
```

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 ListResourceComplianceSummaries。

PowerShell

Tools for PowerShell V4

示例 1：此示例获取资源级摘要计数。摘要包含有关合规和不合规状态的信息以及与“Windows10”匹配产品的详细合规性项目严重性计数。由于参数未指定时 MaxResult 的默认值为 100，且该值无效，因此添加了 MaxResult 参数，并将该值设置为 50。

```
$FilterValues = @{
  "Key"="Product"
        "Type"="EQUAL"
        "Values"="Windows10"
}
        Get-SSMResourceComplianceSummaryList -Filter $FilterValues -MaxResult 50
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 ListResourceComplianceSummaries。

Tools for PowerShell V5

示例 1：此示例获取资源级摘要计数。摘要包含有关合规和不合规状态的信息以及与“Windows10”匹配产品的详细合规性项目严重性计数。由于参数未指定时 MaxResult 的默认值为 100，且该值无效，因此添加了 MaxResult 参数，并将该值设置为 50。

```
$FilterValues = @{
  "Key"="Product"
        "Type"="EQUAL"
        "Values"="Windows10"
}
        Get-SSMResourceComplianceSummaryList -Filter $FilterValues -MaxResult 50
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 ListResourceComplianceSummaries。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **ListTagsForResource** 与 CLI 配合使用

以下代码示例演示如何使用 ListTagsForResource。

CLI

AWS CLI

列出应用于补丁基准的标签

以下 list-tags-for-resource 示例列出了补丁基准的标签。

```
aws ssm list-tags-for-resource \
    --resource-type "PatchBaseline" \
    --resource-id "pb-0123456789abcdef0"
```

输出：

```
{
    "TagList": [
        {
            "Key": "Environment",
            "Value": "Production"
        },
        {
            "Key": "Region",
            "Value": "EMEA"
        }
    ]
}
```

有关更多信息，请参阅《AWS 一般参考》中的标记 AWS 资源。

- 有关 API 详细信息，请参阅《AWS CLI 命令参考》中的 ListTagsForResource。

PowerShell

Tools for PowerShell V4

示例 1：此示例列出维护时段的标签。

```
Get-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
 "MaintenanceWindow"
```

输出：

```
Key   Value
---   -----
Stack Production
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 ListTagsForResource。

Tools for PowerShell V5

示例 1：此示例列出维护时段的标签。

```
Get-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
 "MaintenanceWindow"
```

输出：

```
Key   Value
---   -----
Stack Production
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 ListTagsForResource。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **ModifyDocumentPermission** 与 CLI 配合使用

以下代码示例演示如何使用 `ModifyDocumentPermission`。

CLI

AWS CLI

修改文档权限

以下 modify-document-permission 示例公开共享一个 Systems Manager 文档。

```
aws ssm modify-document-permission \
    --name "Example" \
    --permission-type "Share" \
    --account-ids-to-add "All"
```

此命令不生成任何输出。

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的共享 Systems Manager 文档。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 ModifyDocumentPermission。

PowerShell

Tools for PowerShell V4

示例 1：此示例为文档的所有账户添加“共享”权限。如果此命令成功，则无任何输出。

```
Edit-SSMDocumentPermission -Name "RunShellScript" -PermissionType "Share" -
AccountIdsToAdd all
```

示例 2：此示例为文档的特定账户添加“共享”权限。如果此命令成功，则无任何输出。

```
Edit-SSMDocumentPermission -Name "RunShellScriptNew" -PermissionType "Share" -
AccountIdsToAdd "123456789012"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 ModifyDocumentPermission。

Tools for PowerShell V5

示例 1：此示例为文档的所有账户添加“共享”权限。如果此命令成功，则无任何输出。

```
Edit-SSMDocumentPermission -Name "RunShellScript" -PermissionType "Share" -
AccountIdsToAdd all
```

示例 2：此示例为文档的特定账户添加“共享”权限。如果此命令成功，则无任何输出。

```
Edit-SSMDocumentPermission -Name "RunShellScriptNew" -PermissionType "Share" -
AccountIdsToAdd "123456789012"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 ModifyDocumentPermission。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **PutComplianceItems** 与 CLI 配合使用

以下代码示例演示如何使用 PutComplianceItems。

CLI

AWS CLI

向指定实例注册合规性类型和合规性详细信息

此示例将合规性类型 Custom:AVCheck 注册到指定的托管式实例。如果此命令成功，则无任何输出。

命令:

```
aws ssm put-compliance-items --resource-id "i-1234567890abcdef0" --
resource-type "ManagedInstance" --compliance-type "Custom:AVCheck"
 --execution-summary "ExecutionTime=2019-02-18T16:00:00Z" --
items "Id=Version2.0,Title=ScanHost,Severity=CRITICAL,Status=COMPLIANT"
```

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 PutComplianceItems。

PowerShell

Tools for PowerShell V4

示例 1：此示例为给定的托管式实例编写自定义合规性项目

```
$item = [Amazon.SimpleSystemsManagement.Model.ComplianceItemEntry]::new()
$item.Id = "07Jun2019-3"
$item.Severity="LOW"
$item.Status="COMPLIANT"
```

```
$item.Title="Fin-test-1 - custom"
Write-SSMComplianceItem -ResourceId mi-012dcb3ecea45b678 -ComplianceType
 Custom:VSSCompliant2 -ResourceType ManagedInstance -Item $item -
ExecutionSummary_ExecutionTime "07-Jun-2019"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 PutComplianceItems。

Tools for PowerShell V5

示例 1：此示例为给定的托管式实例编写自定义合规性项目

```
$item = [Amazon.SimpleSystemsManagement.Model.ComplianceItemEntry]::new()
$item.Id = "07Jun2019-3"
$item.Severity="LOW"
$item.Status="COMPLIANT"
$item.Title="Fin-test-1 - custom"
Write-SSMComplianceItem -ResourceId mi-012dcb3ecea45b678 -ComplianceType
 Custom:VSSCompliant2 -ResourceType ManagedInstance -Item $item -
ExecutionSummary_ExecutionTime "07-Jun-2019"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 PutComplianceItems。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **PutInventory** 与 CLI 配合使用

以下代码示例演示如何使用 PutInventory。

CLI

AWS CLI

将客户元数据分配给实例

此示例将机架位置信息分配给某个实例。如果此命令成功，则无任何输出。

命令（Linux）：

```
aws ssm put-inventory --instance-id "i-016648b75dd622dab" --items
 '[{"TypeName": "Custom:RackInfo","SchemaVersion": "1.0","CaptureTime":
```

```
 "2019-01-22T10:01:01Z","Content":[{"RackLocation": "Bay B/Row C/Rack D/Shelf
 E"}]}]'
```

命令（Windows）：

```
aws ssm put-inventory --instance-id "i-016648b75dd622dab" --
items "TypeName=Custom:RackInfo,SchemaVersion=1.0,CaptureTime=2019-01-22T10:01:01Z,Conten
 B/Row C/Rack D/Shelf F'}]"
```

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 PutInventory。

PowerShell

Tools for PowerShell V4

示例 1：此示例将机架位置信息分配给某个实例。如果此命令成功，则无任何输出。

```
$data = New-Object
 "System.Collections.Generic.Dictionary[System.String,System.String]"
$data.Add("RackLocation", "Bay B/Row C/Rack D/Shelf F")

$items = New-Object
 "System.Collections.Generic.List[System.Collections.Generic.Dictionary[System.String,
 System.String]]"
$items.Add($data)

$customInventoryItem = New-Object
 Amazon.SimpleSystemsManagement.Model.InventoryItem
$customInventoryItem.CaptureTime = "2016-08-22T10:01:01Z"
$customInventoryItem.Content = $items
$customInventoryItem.TypeName = "Custom:TestRackInfo2"
$customInventoryItem.SchemaVersion = "1.0"

$inventoryItems = @($customInventoryItem)

Write-SSMInventory -InstanceId "i-0cb2b964d3e14fd9f" -Item $inventoryItems
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 PutInventory。

Tools for PowerShell V5

示例 1：此示例将机架位置信息分配给某个实例。如果此命令成功，则无任何输出。

```
$data = New-Object
 "System.Collections.Generic.Dictionary[System.String,System.String]"
$data.Add("RackLocation", "Bay B/Row C/Rack D/Shelf F")

$items = New-Object
 "System.Collections.Generic.List[System.Collections.Generic.Dictionary[System.String,
 System.String]]"
$items.Add($data)

$customInventoryItem = New-Object
 Amazon.SimpleSystemsManagement.Model.InventoryItem
$customInventoryItem.CaptureTime = "2016-08-22T10:01:01Z"
$customInventoryItem.Content = $items
$customInventoryItem.TypeName = "Custom:TestRackInfo2"
$customInventoryItem.SchemaVersion = "1.0"

$inventoryItems = @($customInventoryItem)

Write-SSMInventory -InstanceId "i-0cb2b964d3e14fd9f" -Item $inventoryItems
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 PutInventory。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **PutParameter** 与 AWS SDK 或 CLI 配合使用

以下代码示例演示如何使用 PutParameter。

CLI

AWS CLI

示例 1：更改参数值

以下 put-parameter 示例更改了指定参数的值。

```
aws ssm put-parameter \
    --name "MyStringParameter" \
    --type "String" \
    --value "Vici" \
```

```
    --overwrite
```

输出：

```
{
    "Version": 2,
    "Tier": "Standard"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的创建 Systems Manager 参数（AWS CLI）、管理参数层和处理参数策略。

示例 2：创建高级参数

以下 put-parameter 示例将创建高级参数。

```
aws ssm put-parameter \
    --name "MyAdvancedParameter" \
    --description "This is an advanced parameter" \
    --value "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do
 eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim
 veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo
 consequat [truncated]" \
    --type "String" \
    --tier Advanced
```

输出：

```
{
    "Version": 1,
    "Tier": "Advanced"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的创建 Systems Manager 参数（AWS CLI）、管理参数层和处理参数策略。

示例 3：将标准参数转换为高级参数

以下 put-parameter 示例将现有标准参数转换为高级参数。

```
aws ssm put-parameter \
```

```
    --name "MyConvertedParameter" \
    --value "abc123" \
    --type "String" \
    --tier Advanced \
    --overwrite
```

输出：

```
{
    "Version": 2,
    "Tier": "Advanced"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的创建 Systems Manager 参数（AWS CLI）、管理参数层和处理参数策略。

示例 4：创建附加有策略的参数

以下 put-parameter 示例创建了一个附加参数策略的高级参数。

```
aws ssm put-parameter \
    --name "/Finance/Payroll/q2accesskey" \
    --value "P@sSwW)rd" \
    --type "SecureString" \
    --tier Advanced \
    --policies "[{\"Type\":\"Expiration\",\"Version\":\"1.0\",\"Attributes\":
{\"Timestamp\":\"2020-06-30T00:00:00.000Z\"}},{\"Type\":\"ExpirationNotification
\",\"Version\":\"1.0\",\"Attributes\":{\"Before\":\"5\",\"Unit\":\"Days\"}},
{\"Type\":\"NoChangeNotification\",\"Version\":\"1.0\",\"Attributes\":{\"After\":
\"60\",\"Unit\":\"Days\"}}]"
```

输出：

```
{
    "Version": 1,
    "Tier": "Advanced"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的创建 Systems Manager 参数（AWS CLI）、管理参数层和处理参数策略。

示例 5：向现有参数添加策略

以下 put-parameter 示例将策略附加到现有高级参数。

```
aws ssm put-parameter \
    --name "/Finance/Payroll/q2accesskey" \
    --value "N3wP@sSwW)rd" \
    --type "SecureString" \
    --tier Advanced \
    --policies "[{\"Type\":\"Expiration\",\"Version\":\"1.0\",\"Attributes\":
{\"Timestamp\":\"2020-06-30T00:00:00.000Z\"}},{\"Type\":\"ExpirationNotification
\",\"Version\":\"1.0\",\"Attributes\":{\"Before\":\"5\",\"Unit\":\"Days\"}},
{\"Type\":\"NoChangeNotification\",\"Version\":\"1.0\",\"Attributes\":{\"After\":
\"60\",\"Unit\":\"Days\"}}]"
    --overwrite
```

输出：

```
{
    "Version": 2,
    "Tier": "Advanced"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的创建 Systems Manager 参数（AWS CLI）、管理参数层和处理参数策略。

- 有关 API 详细信息，请参阅《AWS CLI 命令参考》中的 PutParameter。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.ssm.SsmClient;
import software.amazon.awssdk.services.ssm.model.ParameterType;
import software.amazon.awssdk.services.ssm.model.PutParameterRequest;
import software.amazon.awssdk.services.ssm.model.SsmException;
```

```
public class PutParameter {

    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <paraName>

                Where:
                    paraName - The name of the parameter.
                    paraValue - The value of the parameter.
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String paraName = args[0];
        String paraValue = args[1];
        Region region = Region.US_EAST_1;
        SsmClient ssmClient = SsmClient.builder()
                .region(region)
                .build();

        putParaValue(ssmClient, paraName, paraValue);
        ssmClient.close();
    }

    public static void putParaValue(SsmClient ssmClient, String paraName, String
 value) {
        try {
            PutParameterRequest parameterRequest = PutParameterRequest.builder()
                    .name(paraName)
                    .type(ParameterType.STRING)
                    .value(value)
                    .build();

            ssmClient.putParameter(parameterRequest);
            System.out.println("The parameter was successfully added.");

        } catch (SsmException e) {
            System.err.println(e.getMessage());
```

```
            System.exit(1);
        }
    }
}
```

- 有关 API 详细信息，请参阅《AWS SDK for Java 2.x API 参考》中的 PutParameter。

PowerShell

Tools for PowerShell V4

示例 1：此示例创建一个参数。如果此命令成功，则无任何输出。

```
Write-SSMParameter -Name "Welcome" -Type "String" -Value "helloWorld"
```

示例 2：此示例更改了参数。如果此命令成功，则无任何输出。

```
Write-SSMParameter -Name "Welcome" -Type "String" -Value "Good day, Sunshine!" -
Overwrite $true
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 PutParameter。

Tools for PowerShell V5

示例 1：此示例创建一个参数。如果此命令成功，则无任何输出。

```
Write-SSMParameter -Name "Welcome" -Type "String" -Value "helloWorld"
```

示例 2：此示例更改了参数。如果此命令成功，则无任何输出。

```
Write-SSMParameter -Name "Welcome" -Type "String" -Value "Good day, Sunshine!" -
Overwrite $true
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 PutParameter。

Rust

适用于 Rust 的 SDK

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
async fn make_parameter(
    client: &Client,
    name: &str,
    value: &str,
    description: &str,
) -> Result<(), Error> {
    let resp = client
        .put_parameter()
        .overwrite(true)
        .r#type(ParameterType::String)
        .name(name)
        .value(value)
        .description(description)
        .send()
        .await?;

    println!("Success! Parameter now has version: {}", resp.version());

    Ok(())
}
```

- 有关 API 详细信息，请参阅《AWS SDK for Rust API 参考》中的 PutParameter。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **RegisterDefaultPatchBaseline** 与 CLI 配合使用

以下代码示例演示如何使用 RegisterDefaultPatchBaseline。

CLI

AWS CLI

设置默认补丁基准

以下 register-default-patch-baseline 示例将指定的自定义补丁基准注册为其支持的操作系统类型的默认补丁基准。

```
aws ssm register-default-patch-baseline \
    --baseline-id "pb-abc123cf9bEXAMPLE"
```

输出：

```
{
    "BaselineId":"pb-abc123cf9bEXAMPLE"
}
```

以下 register-default-patch-baseline 示例将 AWS 为 CentOS 提供的默认补丁基准注册为默认补丁基准。

```
aws ssm register-default-patch-baseline \
    --baseline-id "arn:aws:ssm:us-east-2:733109147000:patchbaseline/
pb-0574b43a65ea646ed"
```

输出：

```
{
    "BaselineId":"pb-abc123cf9bEXAMPLE"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的关于预定义和自定义补丁基准。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 RegisterDefaultPatchBaseline。

PowerShell

Tools for PowerShell V4

示例 1：此示例将补丁基准注册为默认补丁基准。

```
Register-SSMDefaultPatchBaseline -BaselineId "pb-03da896ca3b68b639"
```

输出：

```
pb-03da896ca3b68b639
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 RegisterDefaultPatchBaseline。

Tools for PowerShell V5

示例 1：此示例将补丁基准注册为默认补丁基准。

```
Register-SSMDefaultPatchBaseline -BaselineId "pb-03da896ca3b68b639"
```

输出：

```
pb-03da896ca3b68b639
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 RegisterDefaultPatchBaseline。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **RegisterPatchBaselineForPatchGroup** 与 CLI 配合使用

以下代码示例演示如何使用 RegisterPatchBaselineForPatchGroup。

CLI

AWS CLI

为补丁组注册补丁基准

以下 register-patch-baseline-for-patch-group 示例为补丁组注册补丁基准。

```
aws ssm register-patch-baseline-for-patch-group \
    --baseline-id "pb-045f10b4f382baeda" \
    --patch-group "Production"
```

输出：

```
{
    "BaselineId": "pb-045f10b4f382baeda",
    "PatchGroup": "Production"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的创建补丁组 <https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-group-tagging.html>__ 和将补丁组添加到补丁基准。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 RegisterPatchBaselineForPatchGroup。

PowerShell

Tools for PowerShell V4

示例 1：此示例为补丁组注册补丁基准。

```
Register-SSMPatchBaselineForPatchGroup -BaselineId "pb-03da896ca3b68b639" -
PatchGroup "Production"
```

输出：

```
BaselineId           PatchGroup
----------           ----------
pb-03da896ca3b68b639 Production
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 RegisterPatchBaselineForPatchGroup。

Tools for PowerShell V5

示例 1：此示例为补丁组注册补丁基准。

```
Register-SSMPatchBaselineForPatchGroup -BaselineId "pb-03da896ca3b68b639" -
PatchGroup "Production"
```

输出：

```
BaselineId           PatchGroup
----------           ----------
pb-03da896ca3b68b639 Production
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 RegisterPatchBaselineForPatchGroup。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **RegisterTargetWithMaintenanceWindow** 与 CLI 配合使用

以下代码示例演示如何使用 RegisterTargetWithMaintenanceWindow。

CLI

AWS CLI

示例 1：向维护时段注册单个目标

以下 register-target-with-maintenance-window 示例向维护时段注册实例。

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-ab12cd34ef56gh78" \
    --target "Key=InstanceIds,Values=i-0000293ffd8c57862" \
    --owner-information "Single instance" \
    --resource-type "INSTANCE"
```

输出：

```
{
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

示例 2：使用实例 ID 向维护时段注册多个目标

以下 register-target-with-maintenance-window 示例通过指定其实例 ID 向维护时段注册两个实例。

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-ab12cd34ef56gh78" \
    --target "Key=InstanceIds,Values=i-0000293ffd8c57862,i-0cb2b964d3e14fd9f" \
    --owner-information "Two instances in a list" \
    --resource-type "INSTANCE"
```

输出：

```
{
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

示例 3：使用资源标签向维护时段注册目标

以下 register-target-with-maintenance-window 示例通过指定已应用于实例的资源标签，向维护时段注册实例。

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-06cf17cbefcb4bf4f" \
    --targets "Key=tag:Environment,Values=Prod" "Key=Role,Values=Web" \
    --owner-information "Production Web Servers" \
    --resource-type "INSTANCE"
```

输出：

```
{
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

示例 4：使用一组标签键注册目标

以下 register-target-with-maintenance-window 示例注册所有被分配了一个或多个标签键的实例（不考虑其键值）。

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-0c50858d01EXAMPLE" \
    --resource-type "INSTANCE" \
```

```
    --target "Key=tag-key,Values=Name,Instance-Type,CostCenter"
```

输出：

```
{
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

示例 5：使用资源组名称注册目标

以下 `register-target-with-maintenance-window` 示例注册指定的资源组，无论其包含的资源类型如何。

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-0c50858d01EXAMPLE" \
    --resource-type "RESOURCE_GROUP" \
    --target "Key=resource-groups:Name,Values=MyResourceGroup"
```

输出：

```
{
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的向维护时段注册目标实例（AWS CLI）。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 RegisterTargetWithMaintenanceWindow。

PowerShell

Tools for PowerShell V4

示例 1：此示例向维护时段注册实例。

```
$option1 = @{Key="InstanceIds";Values=@("i-0000293ffd8c57862")}
Register-SSMTargetWithMaintenanceWindow -WindowId "mw-06cf17cbefcb4bf4f" -Target
 $option1 -OwnerInformation "Single instance" -ResourceType "INSTANCE"
```

输出：

```
d8e47760-23ed-46a5-9f28-927337725398
```

示例 2：此示例向维护时段注册多个实例。

```
$option1 =
 @{Key="InstanceIds";Values=@("i-0000293ffd8c57862","i-0cb2b964d3e14fd9f")}
Register-SSMTargetWithMaintenanceWindow -WindowId "mw-06cf17cbefcb4bf4f" -Target
 $option1 -OwnerInformation "Single instance" -ResourceType "INSTANCE"
```

输出：

```
6ab5c208-9fc4-4697-84b7-b02a6cc25f7d
```

示例 3：此示例使用 EC2 标签向维护时段注册实例。

```
$option1 = @{Key="tag:Environment";Values=@("Production")}
Register-SSMTargetWithMaintenanceWindow -WindowId "mw-06cf17cbefcb4bf4f" -Target
 $option1 -OwnerInformation "Production Web Servers" -ResourceType "INSTANCE"
```

输出：

```
2994977e-aefb-4a71-beac-df620352f184
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 RegisterTargetWithMaintenanceWindow。

Tools for PowerShell V5

示例 1：此示例向维护时段注册实例。

```
$option1 = @{Key="InstanceIds";Values=@("i-0000293ffd8c57862")}
Register-SSMTargetWithMaintenanceWindow -WindowId "mw-06cf17cbefcb4bf4f" -Target
 $option1 -OwnerInformation "Single instance" -ResourceType "INSTANCE"
```

输出：

```
d8e47760-23ed-46a5-9f28-927337725398
```

示例 2：此示例向维护时段注册多个实例。

```
$option1 =
 @{Key="InstanceIds";Values=@("i-0000293ffd8c57862","i-0cb2b964d3e14fd9f")}
Register-SSMTargetWithMaintenanceWindow -WindowId "mw-06cf17cbefcb4bf4f" -Target
 $option1 -OwnerInformation "Single instance" -ResourceType "INSTANCE"
```

输出：

```
6ab5c208-9fc4-4697-84b7-b02a6cc25f7d
```

示例 3：此示例使用 EC2 标签向维护时段注册实例。

```
$option1 = @{Key="tag:Environment";Values=@("Production")}
Register-SSMTargetWithMaintenanceWindow -WindowId "mw-06cf17cbefcb4bf4f" -Target
 $option1 -OwnerInformation "Production Web Servers" -ResourceType "INSTANCE"
```

输出：

```
2994977e-aefb-4a71-beac-df620352f184
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 RegisterTargetWithMaintenanceWindow。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **RegisterTaskWithMaintenanceWindow** 与 CLI 配合使用

以下代码示例演示如何使用 `RegisterTaskWithMaintenanceWindow`。

CLI

AWS CLI

示例 1：向维护时段注册 Automation 任务

以下 `register-task-with-maintenance-window` 示例向针对实例的维护时段注册 Automation 任务。

```
aws ssm register-task-with-maintenance-window \
    --window-id "mw-082dcd7649EXAMPLE" \
    --targets Key=InstanceIds,Values=i-1234520122EXAMPLE \
    --task-arn AWS-RestartEC2Instance \
    --service-role-arn arn:aws:iam::111222333444:role/SSM --task-type AUTOMATION
 \
    --task-invocation-parameters "{\"Automation\":{\"DocumentVersion\":\"\$LATEST
\",\"Parameters\":{\"InstanceId\":[\"{{RESOURCE_ID}}\"]}}}" \
    --priority 0 \
    --max-concurrency 1 \
    --max-errors 1 \
    --name "AutomationExample" \
    --description "Restarting EC2 Instance for maintenance"
```

输出：

```
{
    "WindowTaskId":"11144444-5555-6666-7777-88888888"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的向维护时段注册任务（AWS CLI）。

示例 2：向维护时段注册 Lambda 任务

以下 `register-task-with-maintenance-window` 示例向针对实例的维护时段注册 Lambda 任务。

```
aws ssm register-task-with-maintenance-window \
    --window-id "mw-082dcd7649dee04e4" \
    --targets Key=InstanceIds,Values=i-12344d305eEXAMPLE \
    --task-arn arn:aws:lambda:us-east-1:111222333444:function:SSMTestLAMBDA \
    --service-role-arn arn:aws:iam::111222333444:role/SSM \
    --task-type LAMBDA \
    --task-invocation-parameters '{"Lambda":{"Payload":"{\"InstanceId\":
\"{{RESOURCE_ID}}\",\"targetType\":\"{{TARGET_TYPE}}\"}","Qualifier":"$LATEST"}}'
 \
    --priority 0 \
    --max-concurrency 10 \
    --max-errors 5 \
    --name "Lambda_Example" \
    --description "My Lambda Example"
```

输出：

```
{
    "WindowTaskId":"22244444-5555-6666-7777-88888888"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的向维护时段注册任务（AWS CLI）。

示例 3：向维护时段注册 Run Command 任务

以下 `register-task-with-maintenance-window` 示例向针对实例的维护时段注册 Run Command 任务。

```
aws ssm register-task-with-maintenance-window \
    --window-id "mw-082dcd7649dee04e4" \
    --targets "Key=InstanceIds,Values=i-12344d305eEXAMPLE" \
    --service-role-arn "arn:aws:iam::111222333444:role/SSM" \
    --task-type "RUN_COMMAND" \
    --name "SSMInstallPowerShellModule" \
    --task-arn "AWS-InstallPowerShellModule" \
    --task-invocation-parameters "{\"RunCommand\":{\"Comment\":\"\",
\"OutputS3BucketName\":\"runcommandlogs\",\"Parameters\":{\"commands\":[\"Get-
Module -ListAvailable\"],\"executionTimeout\":[\"3600\"],\"source\":[\"https:\/
\/gallery.technet.microsoft.com\/EZOut-33ae0fb7\/file\/110351\/1\/EZOut.zip\"],
\"workingDirectory\":[\"\\\\\"]},\"TimeoutSeconds\":600}}" \
    --max-concurrency 1 \
    --max-errors 1 \
    --priority 10
```

输出：

```
{
    "WindowTaskId":"33344444-5555-6666-7777-88888888"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的向维护时段注册任务（AWS CLI）。

示例 4：向维护时段注册 Step Functions 任务

以下 register-task-with-maintenance-window 示例向针对实例的维护时段注册 Step Functions 任务。

```
aws ssm register-task-with-maintenance-window \
    --window-id "mw-1234d787d6EXAMPLE" \
    --targets Key=WindowTargetIds,Values=12347414-69c3-49f8-95b8-ed2dcEXAMPLE \
    --task-arn arn:aws:states:us-
east-1:111222333444:stateMachine:SSMTestStateMachine \
    --service-role-arn arn:aws:iam::111222333444:role/MaintenanceWindows \
    --task-type STEP_FUNCTIONS \
    --task-invocation-parameters '{"StepFunctions":{"Input":"{\"InstanceId\":
\"{{RESOURCE_ID}}\"}"}}' \
    --priority 0 \
    --max-concurrency 10 \
    --max-errors 5 \
    --name "Step_Functions_Example" \
    --description "My Step Functions Example"
```

输出：

```
{
    "WindowTaskId":"44444444-5555-6666-7777-88888888"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的向维护时段注册任务（AWS CLI）。

示例 5：使用维护时段目标 ID 注册任务

以下 register-task-with-maintenance-window 示例使用维护时段目标 ID 注册任务。维护时段目标 ID 位于 aws ssm register-target-with-maintenance-window 命令的输出中。您也可以从 aws ssm describe-maintenance-window-targets 命令输出中进行检索。

```
aws ssm register-task-with-maintenance-window \
    --targets "Key=WindowTargetIds,Values=350d44e6-28cc-44e2-951f-4b2c9EXAMPLE" \
    --task-arn "AWS-RunShellScript" \
    --service-role-arn "arn:aws:iam::111222333444:role/MaintenanceWindowsRole" \
    --window-id "mw-ab12cd34eEXAMPLE" \
    --task-type "RUN_COMMAND" \
    --task-parameters  "{\"commands\":{\"Values\":[\"df\"]}}" \
```

```
    --max-concurrency 1 \
    --max-errors 1 \
    --priority 10
```

输出：

```
{
    "WindowTaskId":"33344444-5555-6666-7777-88888888"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的向维护时段注册任务（AWS CLI）。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 RegisterTaskWithMaintenanceWindow。

PowerShell

Tools for PowerShell V4

示例 1：此示例使用实例 ID 向维护时段注册任务。输出为任务 ID。

```
$parameters = @{}
$parameterValues = New-Object
 Amazon.SimpleSystemsManagement.Model.MaintenanceWindowTaskParameterValueExpression
$parameterValues.Values = @("Install")
$parameters.Add("Operation", $parameterValues)

Register-SSMTaskWithMaintenanceWindow -WindowId "mw-03a342e62c96d31b0"
 -ServiceRoleArn "arn:aws:iam::123456789012:role/MaintenanceWindowsRole"
 -MaxConcurrency 1 -MaxError 1 -TaskArn "AWS-RunShellScript" -Target
 @{ Key="InstanceIds";Values="i-0000293ffd8c57862" } -TaskType "RUN_COMMAND" -
Priority 10 -TaskParameter $parameters
```

输出：

```
f34a2c47-ddfd-4c85-a88d-72366b69af1b
```

示例 2：此示例使用目标 ID 向维护时段注册任务。输出为任务 ID。

```
$parameters = @{}
```

```
$parameterValues = New-Object
 Amazon.SimpleSystemsManagement.Model.MaintenanceWindowTaskParameterValueExpression
$parameterValues.Values = @("Install")
$parameters.Add("Operation", $parameterValues)

register-ssmtaskwithmaintenancewindow -WindowId "mw-03a342e62c96d31b0"
 -ServiceRoleArn "arn:aws:iam::123456789012:role/MaintenanceWindowsRole"
 -MaxConcurrency 1 -MaxError 1 -TaskArn "AWS-RunShellScript" -Target
 @{ Key="WindowTargetIds";Values="350d44e6-28cc-44e2-951f-4b2c985838f6" } -
TaskType "RUN_COMMAND" -Priority 10 -TaskParameter $parameters
```

输出：

```
f34a2c47-ddfd-4c85-a88d-72366b69af1b
```

示例 3：此示例为运行命令文档 **AWS-RunPowerShellScript** 创建参数对象，并使用目标 ID 创建具有给定维护时段的任务。返回的输出是任务 ID。

```
$parameters =
 [Collections.Generic.Dictionary[String,Collections.Generic.List[String]]]::new()
$parameters.Add("commands",@("ipconfig","dir env:\computername"))
$parameters.Add("executionTimeout",@(3600))

$props = @{
    WindowId = "mw-0123e4cce56ff78ae"
    ServiceRoleArn = "arn:aws:iam::123456789012:role/MaintenanceWindowsRole"
    MaxConcurrency = 1
    MaxError = 1
    TaskType = "RUN_COMMAND"
    TaskArn = "AWS-RunPowerShellScript"
    Target =
 @{Key="WindowTargetIds";Values="fe1234ea-56d7-890b-12f3-456b789bee0f"}
    Priority = 1
    RunCommand_Parameter = $parameters
    Name = "set-via-cmdlet"
}

Register-SSMTaskWithMaintenanceWindow @props
```

输出：

```
f1e2ef34-5678-12e3-456a-12334c5c6cbe
```

示例 4：此示例使用名为 **Create-Snapshots** 的文档注册 AWS Systems Manager Automation 任务。

```
$automationParameters = @{}
$automationParameters.Add( "instanceId", @("{{ TARGET_ID }}") )
$automationParameters.Add( "AutomationAssumeRole",
 @("{arn:aws:iam::111111111111:role/AutomationRole}") )
$automationParameters.Add( "SnapshotTimeout", @("PT20M") )
Register-SSMTaskWithMaintenanceWindow -WindowId mw-123EXAMPLE456`
    -ServiceRoleArn "arn:aws:iam::123456789012:role/MW-Role"`
    -MaxConcurrency 1 -MaxError 1 -TaskArn "CreateVolumeSnapshots"`
    -Target @{ Key="WindowTargetIds";Values="4b5acdf4-946c-4355-
bd68-4329a43a5fd1" }`
    -TaskType "AUTOMATION"`
    -Priority 4`
    -Automation_DocumentVersion '$DEFAULT' -Automation_Parameter
 $automationParameters -Name "Create-Snapshots"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 RegisterTaskWithMaintenanceWindow。

Tools for PowerShell V5

示例 1：此示例使用实例 ID 向维护时段注册任务。输出为任务 ID。

```
$parameters = @{}
$parameterValues = New-Object
 Amazon.SimpleSystemsManagement.Model.MaintenanceWindowTaskParameterValueExpression
$parameterValues.Values = @("Install")
$parameters.Add("Operation", $parameterValues)

Register-SSMTaskWithMaintenanceWindow -WindowId "mw-03a342e62c96d31b0"
 -ServiceRoleArn "arn:aws:iam::123456789012:role/MaintenanceWindowsRole"
 -MaxConcurrency 1 -MaxError 1 -TaskArn "AWS-RunShellScript" -Target
 @{ Key="InstanceIds";Values="i-0000293ffd8c57862" } -TaskType "RUN_COMMAND" -
Priority 10 -TaskParameter $parameters
```

输出：

```
f34a2c47-ddfd-4c85-a88d-72366b69af1b
```

示例 2：此示例使用目标 ID 向维护时段注册任务。输出为任务 ID。

```
$parameters = @{}
$parameterValues = New-Object
 Amazon.SimpleSystemsManagement.Model.MaintenanceWindowTaskParameterValueExpression
$parameterValues.Values = @("Install")
$parameters.Add("Operation", $parameterValues)

register-ssmtaskwithmaintenancewindow -WindowId "mw-03a342e62c96d31b0"
 -ServiceRoleArn "arn:aws:iam::123456789012:role/MaintenanceWindowsRole"
 -MaxConcurrency 1 -MaxError 1 -TaskArn "AWS-RunShellScript" -Target
 @{ Key="WindowTargetIds";Values="350d44e6-28cc-44e2-951f-4b2c985838f6" } -
TaskType "RUN_COMMAND" -Priority 10 -TaskParameter $parameters
```

输出：

```
f34a2c47-ddfd-4c85-a88d-72366b69af1b
```

示例 3：此示例为运行命令文档 **AWS-RunPowerShellScript** 创建参数对象，并使用目标 ID 创建具有给定维护时段的任务。返回的输出是任务 ID。

```
$parameters =
 [Collections.Generic.Dictionary[String,Collections.Generic.List[String]]]::new()
$parameters.Add("commands",@("ipconfig","dir env:\computername"))
$parameters.Add("executionTimeout",@(3600))

$props = @{
    WindowId = "mw-0123e4cce56ff78ae"
    ServiceRoleArn = "arn:aws:iam::123456789012:role/MaintenanceWindowsRole"
    MaxConcurrency = 1
    MaxError = 1
    TaskType = "RUN_COMMAND"
    TaskArn = "AWS-RunPowerShellScript"
    Target =
 @{Key="WindowTargetIds";Values="fe1234ea-56d7-890b-12f3-456b789bee0f"}
    Priority = 1
    RunCommand_Parameter = $parameters
    Name = "set-via-cmdlet"
}

Register-SSMTaskWithMaintenanceWindow @props
```

输出：

```
f1e2ef34-5678-12e3-456a-12334c5c6cbe
```

示例 4：此示例使用名为 **Create-Snapshots** 的文档注册 AWS Systems Manager Automation 任务。

```
$automationParameters = @{}
$automationParameters.Add( "instanceId", @("{{ TARGET_ID }}") )
$automationParameters.Add( "AutomationAssumeRole",
 @("{arn:aws:iam::111111111111:role/AutomationRole}") )
$automationParameters.Add( "SnapshotTimeout", @("PT20M") )
Register-SSMTaskWithMaintenanceWindow -WindowId mw-123EXAMPLE456`
    -ServiceRoleArn "arn:aws:iam::123456789012:role/MW-Role"`
    -MaxConcurrency 1 -MaxError 1 -TaskArn "CreateVolumeSnapshots"`
    -Target @{ Key="WindowTargetIds";Values="4b5acdf4-946c-4355-
bd68-4329a43a5fd1" }`
    -TaskType "AUTOMATION"`
    -Priority 4`
    -Automation_DocumentVersion '$DEFAULT' -Automation_Parameter
 $automationParameters -Name "Create-Snapshots"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 RegisterTaskWithMaintenanceWindow。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **RemoveTagsFromResource** 与 CLI 配合使用

以下代码示例演示如何使用 RemoveTagsFromResource。

CLI

AWS CLI

从补丁基准删除标签

以下 remove-tags-from-resource 示例将从补丁基准中删除标签。

```
aws ssm remove-tags-from-resource \
    --resource-type "PatchBaseline" \
    --resource-id "pb-0123456789abcdef0" \
```

```
    --tag-keys "Region"
```

此命令不生成任何输出。

有关更多信息，请参阅《AWS 一般参考》中的标记 AWS 资源。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 RemoveTagsFromResource。

PowerShell

Tools for PowerShell V4

示例 1：此示例从维护时段中删除标签。如果此命令成功，则无任何输出。

```
Remove-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
 "MaintenanceWindow" -TagKey "Production"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 RemoveTagsFromResource。

Tools for PowerShell V5

示例 1：此示例从维护时段中删除标签。如果此命令成功，则无任何输出。

```
Remove-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
 "MaintenanceWindow" -TagKey "Production"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 RemoveTagsFromResource。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 SendCommand 与 AWS SDK 或 CLI 配合使用

以下代码示例演示如何使用 SendCommand。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 了解基础知识

CLI

AWS CLI

示例 1：在一个或多个远程实例上运行命令

以下 send-command 示例在目标实例上运行 echo 命令。

```
aws ssm send-command \
    --document-name "AWS-RunShellScript" \
    --parameters 'commands=["echo HelloWorld"]' \
    --targets "Key=instanceids,Values=i-1234567890abcdef0" \
    --comment "echo HelloWorld"
```

输出：

```
{
    "Command": {
        "CommandId": "92853adf-ba41-4cd6-9a88-142d1EXAMPLE",
        "DocumentName": "AWS-RunShellScript",
        "DocumentVersion": "",
        "Comment": "echo HelloWorld",
        "ExpiresAfter": 1550181014.717,
        "Parameters": {
            "commands": [
                "echo HelloWorld"
            ]
        },
        "InstanceIds": [
            "i-0f00f008a2dcbefe2"
        ],
        "Targets": [],
        "RequestedDateTime": 1550173814.717,
        "Status": "Pending",
        "StatusDetails": "Pending",
        "OutputS3BucketName": "",
        "OutputS3KeyPrefix": "",
        "MaxConcurrency": "50",
        "MaxErrors": "0",
        "TargetCount": 1,
        "CompletedCount": 0,
        "ErrorCount": 0,
        "DeliveryTimedOutCount": 0,
```

```
        "ServiceRole": "",
        "NotificationConfig": {
            "NotificationArn": "",
            "NotificationEvents": [],
            "NotificationType": ""
        },
        "CloudWatchOutputConfig": {
            "CloudWatchLogGroupName": "",
            "CloudWatchOutputEnabled": false
        }
    }
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的使用 Systems Manager Run Command 运行命令。

示例 2：获取有关实例的 IP 信息

以下 send-command 示例检索关于实例的 IP 信息。

```
aws ssm send-command \
    --instance-ids "i-1234567890abcdef0" \
    --document-name "AWS-RunShellScript" \
    --comment "IP config" \
    --parameters "commands=ifconfig"
```

有关输出示例，请参阅示例 1。

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的使用 Systems Manager Run Command 运行命令。

示例 3：在具有特定标签的实例上运行命令

以下 send-command 示例在标签键为“ENV”且值为“Dev”的实例上运行命令。

```
aws ssm send-command \
    --targets "Key=tag:ENV,Values=Dev" \
    --document-name "AWS-RunShellScript" \
    --parameters "commands=ifconfig"
```

有关输出示例，请参阅示例 1。

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的使用 Systems Manager Run Command 运行命令。

示例 4：运行发送 SNS 通知的命令

以下 send-command 示例运行一条命令，发送所有通知事件和 Command 通知类型的 SNS 通知。

```
aws ssm send-command \
    --instance-ids "i-1234567890abcdef0" \
    --document-name "AWS-RunShellScript" \
    --comment "IP config" \
    --parameters "commands=ifconfig" \
    --service-role-arn "arn:aws:iam::123456789012:role/SNS_Role" \
    --notification-config "NotificationArn=arn:aws:sns:us-
east-1:123456789012:SNSTopicName,NotificationEvents=All,NotificationType=Command"
```

有关输出示例，请参阅示例 1。

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的使用 Systems Manager Run Command 运行命令。

示例 5：运行输出到 S3 和 CloudWatch 的命令

以下 send-command 示例运行一条命令，将命令详细信息输出到 S3 存储桶和 CloudWatch Logs 日志组。

```
aws ssm send-command \
    --instance-ids "i-1234567890abcdef0" \
    --document-name "AWS-RunShellScript" \
    --comment "IP config" \
    --parameters "commands=ifconfig" \
    --output-s3-bucket-name "s3-bucket-name" \
    --output-s3-key-prefix "runcommand" \
    --cloud-watch-output-
config "CloudWatchOutputEnabled=true,CloudWatchLogGroupName=CWLGroupName"
```

有关输出示例，请参阅示例 1。

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的使用 Systems Manager Run Command 运行命令。

示例 6：在具有不同标签的多个实例上运行命令

以下 send-command 示例对具有两个不同标签键和值的实例运行命令。

```
aws ssm send-command \
    --document-name "AWS-RunPowerShellScript" \
    --parameters commands=["echo helloWorld"] \
    --targets Key=tag:Env,Values=Dev Key=tag:Role,Values=WebServers
```

有关输出示例，请参阅示例 1。

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的使用 Systems Manager Run Command 运行命令。

示例 7：将具有相同标签键的多个实例设为目标

以下 send-command 示例在具有相同标签键但不同值的实例上运行命令。

```
aws ssm send-command \
    --document-name "AWS-RunPowerShellScript" \
    --parameters commands=["echo helloWorld"] \
    --targets Key=tag:Env,Values=Dev,Test
```

有关输出示例，请参阅示例 1。

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的使用 Systems Manager Run Command 运行命令。

示例 8：运行使用共享文档的命令

以下 send-command 示例在目标实例上运行共享文档。

```
aws ssm send-command \
    --document-name "arn:aws:ssm:us-east-1:123456789012:document/ExampleDocument"
 \
    --targets "Key=instanceids,Values=i-1234567890abcdef0"
```

有关输出示例，请参阅示例 1。

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的使用共享 SSM 文档。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 SendCommand。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
   /**
    * Sends a SSM command to a managed node asynchronously.
    *
    * @param documentName The name of the document to use.
    * @param instanceId The ID of the instance to send the command to.
    * @return The command ID.
    * <p>
    * This method initiates asynchronous requests to send a SSM command to a
managed node.
    * It waits until the document is active, sends the command, and checks the
command execution status.
    */
   public String sendSSMCommand(String documentName, String instanceId) throws
InterruptedException, SsmException {
       // Before we use Document to send a command - make sure it is active.
       CompletableFuture<Void> documentActiveFuture =
CompletableFuture.runAsync(() -> {
           boolean isDocumentActive = false;
           DescribeDocumentRequest request = DescribeDocumentRequest.builder()
                   .name(documentName)
                   .build();

           while (!isDocumentActive) {
               CompletableFuture<DescribeDocumentResponse> response =
getAsyncClient().describeDocument(request);
               String documentStatus =
response.join().document().statusAsString();
               if (documentStatus.equals("Active")) {
                   System.out.println("The SSM document is active and ready to
use.");
                   isDocumentActive = true;
               } else {
```

```
                    System.out.println("The SSM document is not active. Status: "
+ documentStatus);
                    try {
                        Thread.sleep(5000);
                    } catch (InterruptedException e) {
                        throw new RuntimeException(e);
                    }
                }
            }
        });

        documentActiveFuture.join();

        // Create the SendCommandRequest.
        SendCommandRequest commandRequest = SendCommandRequest.builder()
                .documentName(documentName)
                .instanceIds(instanceId)
                .build();

        // Send the command.
        CompletableFuture<SendCommandResponse> commandFuture =
getAsyncClient().sendCommand(commandRequest);
        final String[] commandId = {null};

        commandFuture.whenComplete((commandResponse, ex) -> {
            if (commandResponse != null) {
                commandId[0] = commandResponse.command().commandId();
                System.out.println("Command ID: " + commandId[0]);

                // Wait for the command execution to complete.
                GetCommandInvocationRequest invocationRequest =
GetCommandInvocationRequest.builder()
                        .commandId(commandId[0])
                        .instanceId(instanceId)
                        .build();

                try {
                    System.out.println("Wait 5 secs");
                    TimeUnit.SECONDS.sleep(5);

                    // Retrieve the command execution details.
                    CompletableFuture<GetCommandInvocationResponse>
invocationFuture = getAsyncClient().getCommandInvocation(invocationRequest);
```

```
                    invocationFuture.whenComplete((commandInvocationResponse,
invocationEx) -> {
                        if (commandInvocationResponse != null) {
                            // Check the status of the command execution.
                            CommandInvocationStatus status =
commandInvocationResponse.status();
                            if (status == CommandInvocationStatus.SUCCESS) {
                                System.out.println("Command execution
successful");
                            } else {
                                System.out.println("Command execution failed.
Status: " + status);
                            }
                        } else {
                            Throwable invocationCause = (invocationEx instanceof
CompletionException) ? invocationEx.getCause() : invocationEx;
                            throw new CompletionException(invocationCause);
                        }
                    }).join();
                } catch (InterruptedException e) {
                    throw new RuntimeException(e);
                }
            } else {
                Throwable cause = (ex instanceof CompletionException) ?
ex.getCause() : ex;
                if (cause instanceof SsmException) {
                    throw (SsmException) cause;
                } else {
                    throw new RuntimeException(cause);
                }
            }
        }).join();

        return commandId[0];
    }
```

- 有关 API 详细信息，请参阅《AWS SDK for Java 2.x API Reference》中的 SendCommand。

JavaScript

适用于 JavaScript 的 SDK（v3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import { SendCommandCommand, SSMClient } from "@aws-sdk/client-ssm";
import { parseArgs } from "node:util";

/**
 * Send an SSM command to a managed node.
 * @param {{ documentName: string }}
 */
export const main = async ({ documentName }) => {
  const client = new SSMClient({});
  try {
    await client.send(
      new SendCommandCommand({
        DocumentName: documentName,
      }),
    );
    console.log("Command sent successfully.");
    return { Success: true };
  } catch (caught) {
    if (caught instanceof Error && caught.name === "ValidationError") {
      console.warn(`${caught.message}. Did you provide a valid document name?`);
    } else {
      throw caught;
    }
  }
};
```

- 有关 API 详细信息，请参阅《适用于 JavaScript 的 AWS SDK API Reference》中的 SendCommand。

PowerShell

Tools for PowerShell V4

示例 1：此示例在目标实例上运行 echo 命令。

```
Send-SSMCommand -DocumentName "AWS-RunPowerShellScript" -Parameter @{commands =
 "echo helloWorld"} -Target @{Key="instanceids";Values=@("i-0cb2b964d3e14fd9f")}
```

输出：

```
CommandId          : d8d190fc-32c1-4d65-a0df-ff5ff3965524
Comment            :
CompletedCount     : 0
DocumentName       : AWS-RunPowerShellScript
ErrorCount         : 0
ExpiresAfter       : 3/7/2017 10:48:37 PM
InstanceIds        : {}
MaxConcurrency     : 50
MaxErrors          : 0
NotificationConfig : Amazon.SimpleSystemsManagement.Model.NotificationConfig
OutputS3BucketName :
OutputS3KeyPrefix  :
OutputS3Region     :
Parameters         : {[commands,
 Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
RequestedDateTime  : 3/7/2017 9:48:37 PM
ServiceRole        :
Status             : Pending
StatusDetails      : Pending
TargetCount        : 0
Targets            : {instanceids}
```

示例 2：此示例展示如何运行接受嵌套参数的命令。

```
Send-SSMCommand -DocumentName "AWS-RunRemoteScript" -Parameter
 @{ sourceType="GitHub";sourceInfo='{"owner": "me","repository": "amazon-
ssm","path": "Examples/Install-Win32OpenSSH"}'; "commandLine"=".\Install-
Win32OpenSSH.ps1"} -InstanceId i-0cb2b964d3e14fd9f
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 SendCommand。

Tools for PowerShell V5

示例 1：此示例在目标实例上运行 echo 命令。

```
Send-SSMCommand -DocumentName "AWS-RunPowerShellScript" -Parameter @{commands =
 "echo helloWorld"} -Target @{Key="instanceids";Values=@("i-0cb2b964d3e14fd9f")}
```

输出：

```
CommandId          : d8d190fc-32c1-4d65-a0df-ff5ff3965524
Comment            :
CompletedCount     : 0
DocumentName       : AWS-RunPowerShellScript
ErrorCount         : 0
ExpiresAfter       : 3/7/2017 10:48:37 PM
InstanceIds        : {}
MaxConcurrency     : 50
MaxErrors          : 0
NotificationConfig : Amazon.SimpleSystemsManagement.Model.NotificationConfig
OutputS3BucketName :
OutputS3KeyPrefix  :
OutputS3Region     :
Parameters         : {[commands,
 Amazon.Runtime.Internal.Util.AlwaysSendList`1[System.String]]}
RequestedDateTime  : 3/7/2017 9:48:37 PM
ServiceRole        :
Status             : Pending
StatusDetails      : Pending
TargetCount        : 0
Targets            : {instanceids}
```

示例 2：此示例展示如何运行接受嵌套参数的命令。

```
Send-SSMCommand -DocumentName "AWS-RunRemoteScript" -Parameter
 @{ sourceType="GitHub";sourceInfo='{"owner": "me","repository": "amazon-
ssm","path": "Examples/Install-Win32OpenSSH"}'; "commandLine"=".\Install-
Win32OpenSSH.ps1"} -InstanceId i-0cb2b964d3e14fd9f
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 SendCommand。

Python

适用于 Python 的 SDK (Boto3)

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
class DocumentWrapper:
    """Encapsulates AWS Systems Manager Document actions."""

    def __init__(self, ssm_client):
        """
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.name = None

    @classmethod
    def from_client(cls):
        ssm_client = boto3.client("ssm")
        return cls(ssm_client)


    def send_command(self, instance_ids):
        """
        Sends a command to one or more instances.

        :param instance_ids: The IDs of the instances to send the command to.
        :return: The ID of the command.
        """
        try:
            response = self.ssm_client.send_command(
                InstanceIds=instance_ids, DocumentName=self.name,
 TimeoutSeconds=3600
            )
            return response["Command"]["CommandId"]
        except ClientError as err:
            logger.error(
                "Couldn't send command to %s. Here's why: %s: %s",
```

```
                self.name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 有关 API 详细信息，请参阅《AWS SDK for Python (Boto3) API 参考》中的 SendCommand。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **StartAutomationExecution** 与 CLI 配合使用

以下代码示例演示如何使用 StartAutomationExecution。

CLI

AWS CLI

示例 1：执行自动化文档

以下 start-automation-execution 示例运行自动化文档。

```
aws ssm start-automation-execution \
    --document-name "AWS-UpdateLinuxAmi" \
    --parameters "AutomationAssumeRole=arn:aws:iam::123456789012:role/
SSMAutomationRole,SourceAmiId=ami-EXAMPLE,IamInstanceProfileName=EC2InstanceRole"
```

输出：

```
{
  "AutomationExecutionId": "4105a4fc-f944-11e6-9d32-0a1b2EXAMPLE"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的手动运行自动化工作流。

示例 2：运行共享自动化文档

以下 start-automation-execution 示例运行一个共享的自动化文档。

```
aws ssm start-automation-execution \
    --document-name "arn:aws:ssm:us-east-1:123456789012:document/ExampleDocument"
```

输出：

```
{
  "AutomationExecutionId": "4105a4fc-f944-11e6-9d32-0a1b2EXAMPLE"
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的使用共享 SSM 文档。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 StartAutomationExecution。

PowerShell

Tools for PowerShell V4

示例 1：此示例运行一个文档，指定自动化角色、AMI 源 ID 和 Amazon EC2 实例角色。

```
Start-SSMAutomationExecution -DocumentName AWS-UpdateLinuxAmi -
Parameter @{'AutomationAssumeRole'='arn:aws:iam::123456789012:role/
SSMAutomationRole';'SourceAmiId'='ami-
f173cc91';'InstanceIamRole'='EC2InstanceRole'}
```

输出：

```
3a532a4f-0382-11e7-9df7-6f11185f6dd1
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 StartAutomationExecution。

Tools for PowerShell V5

示例 1：此示例运行一个文档，指定自动化角色、AMI 源 ID 和 Amazon EC2 实例角色。

```
Start-SSMAutomationExecution -DocumentName AWS-UpdateLinuxAmi -
Parameter @{'AutomationAssumeRole'='arn:aws:iam::123456789012:role/
```

```
SSMAutomationRole';'SourceAmiId'='ami-
f173cc91';'InstanceIamRole'='EC2InstanceRole'}
```

输出：

```
3a532a4f-0382-11e7-9df7-6f11185f6dd1
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 StartAutomationExecution。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **StartSession** 与 CLI 配合使用

以下代码示例演示如何使用 StartSession。

CLI

AWS CLI

示例 1：启动会话管理器会话

此 start-session 示例为会话管理器会话建立与实例的连接。请注意，此交互式命令要求在进行调用的客户端计算机上安装会话管理器插件。

```
aws ssm start-session \
    --target "i-1234567890abcdef0"
```

输出：

```
Starting session with SessionId: Jane-Roe-07a16060613c408b5
```

示例 2：使用 SSH 启动会话管理器会话

此 start-session 示例为使用 SSH 的会话管理器会话建立与实例的连接。请注意，此交互式命令要求在进行调用的客户端计算机上安装会话管理器插件，并且该命令在实例上使用默认用户，例如为 Linux 的 EC2 实例使用 ec2-user。

```
ssh -i /path/my-key-pair.pem ec2-user@i-02573cafcfEXAMPLE
```

输出：

```
Starting session with SessionId: ec2-user-07a16060613c408b5
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的启动会话和为 AWS CLI 安装会话管理器插件。

- 有关 API 详细信息，请参阅《AWS CLI 命令参考》中的 StartSession。

PowerShell

Tools for PowerShell V4

示例 1：此示例为会话管理器会话启动与目标的连接，从而启用端口转发。

```
Start-SSMSession -Target 'i-064578e5e7454488f' -DocumentName 'AWS-
StartPortForwardingSession' -Parameter @{ localPortNumber = '8080'; portNumber =
 '80' }
```

输出：

```
SessionId    StreamUrl
----------    ----------
random-id0    wss://ssmmessages.amazonaws.com/v1/data-channel/random-id
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 StartSession。

Tools for PowerShell V5

示例 1：此示例为会话管理器会话启动与目标的连接，从而启用端口转发。注意：如果尚未使用 AWSCredentials 进行设置，则需要添加 Region 参数。

```
Start-SSMSession -Target 'i-064578e5e7454488f' -DocumentName 'AWS-
StartPortForwardingSession' -Parameter @{ localPortNumber = '8080'; portNumber =
 '80' } -Region 'us-west-1'
```

输出：

```
Starting session with SessionId: testUser-xi4glew849asyeryde34u4dfsdfy
Port 8080 opened for sessionId testUser-xi4glew849asyeryde34u4dfsdfy.
```

```
Waiting for connections...
```

示例 2：此示例为会话管理器会话创建与指定实例的交互式会话。

```
Start-SSMSession -Target 'i-1234567890abcdef0' -Region 'us-west-1'
```

输出：

```
Starting session with SessionId : testUser-xi4glew849asyeryde34u4dfsdfy
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.



Install the latest PowerShell for new features and improvements!


PS C:\Windows\system32> whoami
ec2amaz-fnsdrwv\ec2-test-user
PS C:\Windows\system32>
```

示例 3：此示例创建一个会话但不连接到它，并返回连接到该会话所需的 SessionId、StreamUrl 和 TokenValue 属性。

```
Start-SSMSession -Target 'i-1234567890abcdef0' -Region 'us-west-1' -
DisablePluginInvocation
```

输出：

```
SessionId        : testUser-xi4glew849asyeryde34u4dfsdfy
StreamUrl        : {StreamUrl value redacted}
TokenValue       : {Token value redacted}
ContentLength    : 1207
HttpStatusCode   : OK
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 StartSession。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **StopAutomationExecution** 与 CLI 配合使用

以下代码示例演示如何使用 StopAutomationExecution。

CLI

AWS CLI

停止自动化执行

以下 stop-automation-execution 示例停止自动化文档。

```
aws ssm stop-automation-execution
    --automation-execution-id "4105a4fc-f944-11e6-9d32-0a1b2EXAMPLE"
```

此命令不生成任何输出。

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的手动运行自动化工作流。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 StopAutomationExecution。

PowerShell

Tools for PowerShell V4

示例 1：此示例停止自动化执行。如果此命令成功，则无任何输出。

```
Stop-SSMAutomationExecution -AutomationExecutionId "4105a4fc-
f944-11e6-9d32-8fb2db27a909"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 StopAutomationExecution。

Tools for PowerShell V5

示例 1：此示例停止自动化执行。如果此命令成功，则无任何输出。

```
Stop-SSMAutomationExecution -AutomationExecutionId "4105a4fc-
f944-11e6-9d32-8fb2db27a909"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 StopAutomationExecution。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **UpdateAssociation** 与 CLI 配合使用

以下代码示例演示如何使用 UpdateAssociation。

CLI

AWS CLI

示例 1：更新文档关联

以下 update-association 示例使用新文档版本更新关联。

```
aws ssm update-association \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab" \
    --document-version "\$LATEST"
```

输出：

```
{
    "AssociationDescription": {
        "Name": "AWS-UpdateSSMAgent",
        "AssociationVersion": "2",
        "Date": 1550508093.293,
        "LastUpdateAssociationDate": 1550508106.596,
        "Overview": {
            "Status": "Pending",
            "DetailedStatus": "Creating"
        },
        "DocumentVersion": "$LATEST",
        "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
        "Targets": [
            {
                "Key": "tag:Name",
                "Values": [
                    "Linux"
                ]
```

```
            }
        ],
        "LastExecutionDate": 1550508094.879,
        "LastSuccessfulExecutionDate": 1550508094.879
    }
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的编辑和创建关联的新版本。

示例 2：更新关联的计划表达式

以下 update-association 示例更新了指定关联的计划表达式。

```
aws ssm update-association \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab" \
    --schedule-expression "cron(0 0 0/4 1/1 * ? *)"
```

输出：

```
{
    "AssociationDescription": {
        "Name": "AWS-HelloWorld",
        "AssociationVersion": "2",
        "Date": "2021-02-08T13:54:19.203000-08:00",
        "LastUpdateAssociationDate": "2021-06-29T11:51:07.933000-07:00",
        "Overview": {
            "Status": "Pending",
            "DetailedStatus": "Creating"
        },
        "DocumentVersion": "$DEFAULT",
        "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
        "Targets": [
            {
                "Key": "aws:NoOpAutomationTag",
                "Values": [
                    "AWS-NoOpAutomationTarget-Value"
                ]
            }
        ],
        "ScheduleExpression": "cron(0 0 0/4 1/1 * ? *)",
        "LastExecutionDate": "2021-06-26T19:00:48.110000-07:00",
        "ApplyOnlyAtCronInterval": false
    }
```

```
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的编辑和创建关联的新版本。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 UpdateAssociation。

PowerShell

Tools for PowerShell V4

示例 1：此示例使用新文档版本更新关联。

```
Update-SSMAssociation -AssociationId "93285663-92df-44cb-9f26-2292d4ecc439" -
DocumentVersion "1"
```

输出：

```
Name                  : AWS-UpdateSSMAgent
InstanceId            :
Date                  : 3/1/2017 6:22:21 PM
Status.Name           :
Status.Date           :
Status.Message        :
Status.AdditionalInfo :
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 UpdateAssociation。

Tools for PowerShell V5

示例 1：此示例使用新文档版本更新关联。

```
Update-SSMAssociation -AssociationId "93285663-92df-44cb-9f26-2292d4ecc439" -
DocumentVersion "1"
```

输出：

```
Name                  : AWS-UpdateSSMAgent
InstanceId            :
Date                  : 3/1/2017 6:22:21 PM
Status.Name           :
```

```
Status.Date           :
Status.Message        :
Status.AdditionalInfo :
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 UpdateAssociation。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **UpdateAssociationStatus** 与 CLI 配合使用

以下代码示例演示如何使用 UpdateAssociationStatus。

CLI

AWS CLI

更新关联状态

以下 update-association-status 示例更新了实例和文档之间关联的关联状态。

```
aws ssm update-association-status \
    --name "AWS-UpdateSSMAgent" \
    --instance-id "i-1234567890abcdef0" \
    --association-
status "Date=1424421071.939,Name=Pending,Message=temp_status_change,AdditionalInfo=Additi
Config-Needed"
```

输出：

```
{
    "AssociationDescription": {
        "Name": "AWS-UpdateSSMAgent",
        "InstanceId": "i-1234567890abcdef0",
        "AssociationVersion": "1",
        "Date": 1550507529.604,
        "LastUpdateAssociationDate": 1550507806.974,
        "Status": {
            "Date": 1424421071.0,
            "Name": "Pending",
            "Message": "temp_status_change",
```

```
            "AdditionalInfo": "Additional-Config-Needed"
        },
        "Overview": {
            "Status": "Success",
            "AssociationStatusAggregatedCount": {
                "Success": 1
            }
        },
        "DocumentVersion": "$DEFAULT",
        "AssociationId": "8dfe3659-4309-493a-8755-0123456789ab",
        "Targets": [
            {
                "Key": "InstanceIds",
                "Values": [
                    "i-1234567890abcdef0"
                ]
            }
        ],
        "LastExecutionDate": 1550507808.0,
        "LastSuccessfulExecutionDate": 1550507808.0
    }
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的在 Systems Manager 中使用关联。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 UpdateAssociationStatus。

PowerShell

Tools for PowerShell V4

示例 1：此示例更新实例与配置文档之间关联的关联状态。

```
Update-SSMAssociationStatus -Name "AWS-UpdateSSMAgent" -InstanceId
 "i-0000293ffd8c57862" -AssociationStatus_Date "2015-02-20T08:31:11Z"
 -AssociationStatus_Name "Pending" -AssociationStatus_Message
 "temporary_status_change" -AssociationStatus_AdditionalInfo "Additional-Config-
Needed"
```

输出：

```
Name                   : AWS-UpdateSSMAgent
InstanceId             : i-0000293ffd8c57862
Date                   : 2/23/2017 6:55:22 PM
Status.Name            : Pending
Status.Date            : 2/20/2015 8:31:11 AM
Status.Message         : temporary_status_change
Status.AdditionalInfo : Additional-Config-Needed
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 UpdateAssociationStatus。

Tools for PowerShell V5

示例 1：此示例更新实例与配置文档之间关联的关联状态。

```
Update-SSMAssociationStatus -Name "AWS-UpdateSSMAgent" -InstanceId
 "i-0000293ffd8c57862" -AssociationStatus_Date "2015-02-20T08:31:11Z"
 -AssociationStatus_Name "Pending" -AssociationStatus_Message
 "temporary_status_change" -AssociationStatus_AdditionalInfo "Additional-Config-
Needed"
```

输出：

```
Name                   : AWS-UpdateSSMAgent
InstanceId             : i-0000293ffd8c57862
Date                   : 2/23/2017 6:55:22 PM
Status.Name            : Pending
Status.Date            : 2/20/2015 8:31:11 AM
Status.Message         : temporary_status_change
Status.AdditionalInfo : Additional-Config-Needed
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 UpdateAssociationStatus。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **UpdateDocument** 与 CLI 配合使用

以下代码示例演示如何使用 UpdateDocument。

CLI

AWS CLI

创建文档的新版本

以下 update-document 示例在 Windows 计算机上运行时创建文档的新版本。--document 指定的文档必须采用 JSON 格式。请注意，必须先引用 file://，后跟内容文件的路径。由于 --document-version 参数的开头有 $，因此在 Windows 上，必须用双引号将该值括起来。在 Linux、MacOS 或 PowerShell 提示符下，必须用单引号将该值括起来。

Windows 版本：

```
aws ssm update-document \
    --name "RunShellScript" \
    --content "file://RunShellScript.json" \
    --document-version "$LATEST"
```

Linux/Mac 版本：

```
aws ssm update-document \
    --name "RunShellScript" \
    --content "file://RunShellScript.json" \
    --document-version '$LATEST'
```

输出：

```
{
  "DocumentDescription": {
      "Status": "Updating",
      "Hash": "f775e5df4904c6fa46686c4722fae9de1950dace25cd9608ff8d622046b68d9b",
      "Name": "RunShellScript",
      "Parameters": [
          {
              "Type": "StringList",
              "Name": "commands",
              "Description": "(Required) Specify a shell script or a command to
 run."
          }
      ],
      "DocumentType": "Command",
      "PlatformTypes": [
```

```
            "Linux"
        ],
        "DocumentVersion": "2",
        "HashType": "Sha256",
        "CreatedDate": 1487899655.152,
        "Owner": "809632081692",
        "SchemaVersion": "2.0",
        "DefaultVersion": "1",
        "LatestVersion": "2",
        "Description": "Run an updated script"
    }
}
```

- 有关 API 详细信息，请参阅《AWS CLI 命令参考》中的 UpdateDocument。

PowerShell

Tools for PowerShell V4

示例 1：此示例将使用您指定的 json 文件更新内容创建文档的新版本。该文档必须采用 JSON 格式。您可以使用“Get-SSMDocumentVersionList”cmdlet 获取文档版本。

```
Update-SSMDocument -Name RunShellScript -DocumentVersion "1" -Content (Get-
Content -Raw "c:\temp\RunShellScript.json")
```

输出：

```
CreatedDate     : 3/1/2017 2:59:17 AM
DefaultVersion  : 1
Description     : Run an updated script
DocumentType    : Command
DocumentVersion : 2
Hash            :
 1d5ce820e999ff051eb4841ed887593daf77120fd76cae0d18a53cc42e4e22c1
HashType        : Sha256
LatestVersion   : 2
Name            : RunShellScript
Owner           : 809632081692
Parameters      : {commands}
PlatformTypes   : {Linux}
SchemaVersion   : 2.0
Sha1            :
```

```
Status          : Updating
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 UpdateDocument。

Tools for PowerShell V5

示例 1：此示例将使用您指定的 json 文件更新内容创建文档的新版本。该文档必须采用 JSON 格式。您可以使用“Get-SSMDocumentVersionList”cmdlet 获取文档版本。

```
Update-SSMDocument -Name RunShellScript -DocumentVersion "1" -Content (Get-
Content -Raw "c:\temp\RunShellScript.json")
```

输出：

```
CreatedDate     : 3/1/2017 2:59:17 AM
DefaultVersion  : 1
Description     : Run an updated script
DocumentType    : Command
DocumentVersion : 2
Hash            :
 1d5ce820e999ff051eb4841ed887593daf77120fd76cae0d18a53cc42e4e22c1
HashType        : Sha256
LatestVersion   : 2
Name            : RunShellScript
Owner           : 809632081692
Parameters      : {commands}
PlatformTypes   : {Linux}
SchemaVersion   : 2.0
Sha1            :
Status          : Updating
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 UpdateDocument。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **UpdateDocumentDefaultVersion** 与 CLI 配合使用

以下代码示例演示如何使用 UpdateDocumentDefaultVersion。

CLI

AWS CLI

更新文档的默认版本

以下 update-document-default-version 示例更新了 Systems Manager 文档的默认版本。

```
aws ssm update-document-default-version \
    --name "Example" \
    --document-version "2"
```

输出：

```
{
    "Description": {
        "Name": "Example",
        "DefaultVersion": "2"
    }
}
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的编写 SSM 文档内容。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 UpdateDocumentDefaultVersion。

PowerShell

Tools for PowerShell V4

示例 1：此实例更新文档的默认版本。您可以使用“Get-SSMDocumentVersionList”cmdlet 获取可用的文档版本。

```
Update-SSMDocumentDefaultVersion -Name "RunShellScript" -DocumentVersion "2"
```

输出：

```
DefaultVersion Name
-------------- ----
2              RunShellScript
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 UpdateDocumentDefaultVersion。

Tools for PowerShell V5

示例 1：此实例更新文档的默认版本。您可以使用“Get-SSMDocumentVersionList”cmdlet 获取可用的文档版本。

```
Update-SSMDocumentDefaultVersion -Name "RunShellScript" -DocumentVersion "2"
```

输出：

```
DefaultVersion Name
-------------- ----
2              RunShellScript
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 UpdateDocumentDefaultVersion。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **UpdateMaintenanceWindow** 与 AWS SDK 或 CLI 配合使用

以下代码示例演示如何使用 UpdateMaintenanceWindow。

CLI

AWS CLI

示例 1：更新维护时段

以下 update-maintenance-window 示例更新了维护时段的名称。

```
aws ssm update-maintenance-window \
    --window-id "mw-1a2b3c4d5e6f7g8h9" \
    --name "My-Renamed-MW"
```

输出：

```
{
```

```
    "Cutoff": 1,
    "Name": "My-Renamed-MW",
    "Schedule": "cron(0 16 ? * TUE *)",
    "Enabled": true,
    "AllowUnassociatedTargets": true,
    "WindowId": "mw-1a2b3c4d5e6f7g8h9",
    "Duration": 4
}
```

示例 2：禁用维护时段

以下 update-maintenance-window 示例禁用维护时段。

```
aws ssm update-maintenance-window \
    --window-id "mw-1a2b3c4d5e6f7g8h9" \
    --no-enabled
```

示例 3：启用维护时段

以下 update-maintenance-window 示例启用维护时段。

```
aws ssm update-maintenance-window \
    --window-id "mw-1a2b3c4d5e6f7g8h9" \
    --enabled
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的更新维护时段（AWS CLI）。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 UpdateMaintenanceWindow。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    /**
```

```
    * Updates an SSM maintenance window asynchronously.
    *
    * @param id The ID of the maintenance window to update.
    * @param name The new name for the maintenance window.
    * <p>
    * This method initiates an asynchronous request to update an SSM maintenance
window.
    * If the request is successful, it prints a success message.
    * If an exception occurs, it handles the error appropriately.
    */
   public void updateSSMMaintenanceWindow(String id, String name) throws
SsmException {
       UpdateMaintenanceWindowRequest updateRequest =
UpdateMaintenanceWindowRequest.builder()
               .windowId(id)
               .allowUnassociatedTargets(true)
               .duration(24)
               .enabled(true)
               .name(name)
               .schedule("cron(0 0 ? * MON *)")
               .build();

       CompletableFuture<UpdateMaintenanceWindowResponse> future =
getAsyncClient().updateMaintenanceWindow(updateRequest);
       future.whenComplete((response, ex) -> {
           if (response != null) {
               System.out.println("The SSM maintenance window was successfully
updated");
           } else {
               Throwable cause = (ex instanceof CompletionException) ?
ex.getCause() : ex;
               if (cause instanceof SsmException) {
                   throw new CompletionException(cause);
               } else {
                   throw new RuntimeException(cause);
               }
           }
       }).join();
   }
```

- 有关 API 详细信息，请参阅《AWS SDK for Java 2.x API Reference》中的 UpdateMaintenanceWindow。

JavaScript

适用于 JavaScript 的 SDK（v3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import { UpdateMaintenanceWindowCommand, SSMClient } from "@aws-sdk/client-ssm";
import { parseArgs } from "node:util";

/**
 * Update an SSM maintenance window.
 * @param {{ windowId: string, allowUnassociatedTargets?: boolean, duration?:
 number, enabled?: boolean, name?: string, schedule?: string }}
 */
export const main = async ({
  windowId,
  allowUnassociatedTargets = undefined, //Allow the maintenance window to run on
 managed nodes, even if you haven't registered those nodes as targets.
  duration = undefined, //The duration of the maintenance window in hours.
  enabled = undefined,
  name = undefined,
  schedule = undefined, //The schedule of the maintenance window in the form of a
 cron or rate expression.
}) => {
  const client = new SSMClient({});
  try {
    const { opsItemArn, opsItemId } = await client.send(
      new UpdateMaintenanceWindowCommand({
        WindowId: windowId,
        AllowUnassociatedTargets: allowUnassociatedTargets,
        Duration: duration,
        Enabled: enabled,
        Name: name,
        Schedule: schedule,
      }),
    );
    console.log("Maintenance window updated.");
    return { OpsItemArn: opsItemArn, OpsItemId: opsItemId };
```

```
  } catch (caught) {
    if (caught instanceof Error && caught.name === "ValidationError") {
      console.warn(`${caught.message}. Are these values correct?`);
    } else {
      throw caught;
    }
  }
};
```

- 有关 API 详细信息，请参阅《适用于 JavaScript 的 AWS SDK API Reference》中的 UpdateMaintenanceWindow。

PowerShell

Tools for PowerShell V4

示例 1：此示例更新维护时段的名称。

```
Update-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d" -Name "My-Renamed-
MW"
```

输出：

```
AllowUnassociatedTargets : False
Cutoff                   : 1
Duration                 : 2
Enabled                  : True
Name                     : My-Renamed-MW
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d
```

示例 2：此示例启用维护时段。

```
Update-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d" -Enabled $true
```

输出：

```
AllowUnassociatedTargets : False
Cutoff                   : 1
Duration                 : 2
```

```
Enabled                   : True
Name                      : My-Renamed-MW
Schedule                  : cron(0 */30 * * * ? *)
WindowId                  : mw-03eb9db42890fb82d
```

示例 3：此示例禁用维护时段。

```
Update-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d" -Enabled $false
```

输出：

```
AllowUnassociatedTargets : False
Cutoff                   : 1
Duration                 : 2
Enabled                  : False
Name                     : My-Renamed-MW
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 UpdateMaintenanceWindow。

Tools for PowerShell V5

示例 1：此示例更新维护时段的名称。

```
Update-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d" -Name "My-Renamed-
MW"
```

输出：

```
AllowUnassociatedTargets : False
Cutoff                   : 1
Duration                 : 2
Enabled                  : True
Name                     : My-Renamed-MW
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d
```

示例 2：此示例启用维护时段。

```
Update-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d" -Enabled $true
```

输出：

```
AllowUnassociatedTargets : False
Cutoff                   : 1
Duration                 : 2
Enabled                  : True
Name                     : My-Renamed-MW
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d
```

示例 3：此示例禁用维护时段。

```
Update-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d" -Enabled $false
```

输出：

```
AllowUnassociatedTargets : False
Cutoff                   : 1
Duration                 : 2
Enabled                  : False
Name                     : My-Renamed-MW
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 UpdateMaintenanceWindow。

Python

适用于 Python 的 SDK (Boto3)

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
class MaintenanceWindowWrapper:
    """Encapsulates AWS Systems Manager maintenance window actions."""
```

```
    def __init__(self, ssm_client):
        """
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.window_id = None
        self.name = None

    @classmethod
    def from_client(cls):
        ssm_client = boto3.client("ssm")
        return cls(ssm_client)


    def update(
        self, name, enabled, schedule, duration, cutoff,
allow_unassociated_targets
    ):
        """
        Update an AWS Systems Manager maintenance window.

        :param name: The name of the maintenance window.
        :param enabled: Whether the maintenance window is enabled to run on
managed nodes.
        :param schedule: The schedule of the maintenance window.
        :param duration: The duration of the maintenance window.
        :param cutoff: The cutoff time of the maintenance window.
        :param allow_unassociated_targets: Allow the maintenance window to run on
managed nodes, even
                                           if you haven't registered those nodes
as targets.
        """
        try:
            self.ssm_client.update_maintenance_window(
                WindowId=self.window_id,
                Name=name,
                Enabled=enabled,
                Schedule=schedule,
                Duration=duration,
                Cutoff=cutoff,
                AllowUnassociatedTargets=allow_unassociated_targets,
            )
            self.name = name
            logger.info("Updated maintenance window %s.", self.window_id)
```

```
        except ParamValidationError as error:
            logger.error(
                "Parameter validation error when trying to update maintenance
window %s. Here's why: %s",
                self.window_id,
                error,
            )
            raise
        except ClientError as err:
            logger.error(
                "Couldn't update maintenance window %s. Here's why: %s: %s",
                self.name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 有关 API 详细信息，请参阅《AWS SDK for Python (Boto3) API 参考》中的 UpdateMaintenanceWindow。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **UpdateManagedInstanceRole** 与 CLI 配合使用

以下代码示例演示如何使用 UpdateManagedInstanceRole。

CLI

AWS CLI

更新托管式实例的 IAM 角色

以下 update-managed-instance-role 示例更新了托管式实例的 IAM 实例配置文件。

```
aws ssm update-managed-instance-role \
    --instance-id "mi-08ab247cdfEXAMPLE" \
    --iam-role "ExampleRole"
```

此命令不生成任何输出。

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的步骤 4：为 Systems Manager 创建 IAM 实例配置文件。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 UpdateManagedInstanceRole。

PowerShell

Tools for PowerShell V4

示例 1：此示例更新托管式实例的角色。如果此命令成功，则无任何输出。

```
Update-SSMManagedInstanceRole -InstanceId "mi-08ab247cdf1046573" -IamRole
 "AutomationRole"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 UpdateManagedInstanceRole。

Tools for PowerShell V5

示例 1：此示例更新托管式实例的角色。如果此命令成功，则无任何输出。

```
Update-SSMManagedInstanceRole -InstanceId "mi-08ab247cdf1046573" -IamRole
 "AutomationRole"
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 UpdateManagedInstanceRole。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **UpdateOpsItem** 与 AWS SDK 或 CLI 配合使用

以下代码示例演示如何使用 UpdateOpsItem。

操作示例是大型程序的代码摘录，必须在上下文中运行。在以下代码示例中，您可以查看此操作的上下文：

- 了解基础知识

CLI

AWS CLI

更新 OpsItem

以下 `update-ops-item` 示例更新 OpsItem 的描述、优先级和类别。此外，该命令还指定一个 SNS 主题，即，当编辑或更改此 OpsItem 时，将发送通知。

```
aws ssm update-ops-item \
    --ops-item-id "oi-287b5EXAMPLE" \
    --description "Primary OpsItem for failover event 2020-01-01-fh398yf" \
    --priority 2 \
    --category "Security" \
    --notifications "Arn=arn:aws:sns:us-east-2:111222333444:my-us-east-2-topic"
```

输出：

```
This command produces no output.
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的使用 OpsItem。

- 有关 API 详细信息，请参阅《AWS CLI 命令参考》中的 UpdateOpsItem。

Java

适用于 Java 的 SDK 2.x

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
    /**
     * Resolves an AWS SSM OpsItem asynchronously.
     *
     * @param opsID The ID of the OpsItem to resolve.
     * <p>
     * This method initiates an asynchronous request to resolve an SSM OpsItem.
```

```
     * If an exception occurs, it handles the error appropriately.
     */
    public void resolveOpsItem(String opsID) {
        UpdateOpsItemRequest opsItemRequest = UpdateOpsItemRequest.builder()
                .opsItemId(opsID)
                .status(OpsItemStatus.RESOLVED)
                .build();

        CompletableFuture<Void> future = CompletableFuture.runAsync(() -> {
            getAsyncClient().updateOpsItem(opsItemRequest)
                    .thenAccept(response -> {
                        System.out.println("OpsItem resolved successfully.");
                    })
                    .exceptionally(ex -> {
                        throw new CompletionException(ex);
                    }).join();
        }).exceptionally(ex -> {
            Throwable cause = (ex instanceof CompletionException) ?
ex.getCause() : ex;
            if (cause instanceof SsmException) {
                throw new RuntimeException("SSM error: " + cause.getMessage(),
cause);
            } else {
                throw new RuntimeException("Unexpected error: " +
cause.getMessage(), cause);
            }
        });

        try {
            future.join();
        } catch (CompletionException ex) {
            throw ex.getCause() instanceof RuntimeException ? (RuntimeException)
ex.getCause() : ex;
        }
    }
```

- 有关 API 详细信息，请参阅《AWS SDK for Java 2.x API 参考》中的 UpdateOpsItem。

JavaScript

适用于 JavaScript 的 SDK（v3）

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
import { UpdateOpsItemCommand, SSMClient } from "@aws-sdk/client-ssm";
import { parseArgs } from "node:util";

/**
 * Update an SSM OpsItem.
 * @param {{ opsItemId: string, status?: OpsItemStatus }}
 */
export const main = async ({
  opsItemId,
  status = undefined, // The OpsItem status. Status can be Open, In Progress, or
 Resolved
}) => {
  const client = new SSMClient({});
  try {
    await client.send(
      new UpdateOpsItemCommand({
        OpsItemId: opsItemId,
        Status: status,
      }),
    );
    console.log("Ops item updated.");
    return { Success: true };
  } catch (caught) {
    if (
      caught instanceof Error &&
      caught.name === "OpsItemLimitExceededException"
    ) {
      console.warn(
        `Couldn't create ops item because you have exceeded your open OpsItem
 limit. ${caught.message}.`,
      );
    } else {
```

```
      throw caught;
    }
  }
};
```

- 有关 API 详细信息，请参阅《适用于 JavaScript 的 AWS SDK API 参考》中的 UpdateOpsItem。

Python

适用于 Python 的 SDK (Boto3)

> **Note**
>
> 查看 GitHub，了解更多信息。在 AWS 代码示例存储库中查找完整示例，了解如何进行设置和运行。

```
class OpsItemWrapper:
    """Encapsulates AWS Systems Manager OpsItem actions."""

    def __init__(self, ssm_client):
        """
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.id = None

    @classmethod
    def from_client(cls):
        """
        :return: A OpsItemWrapper instance.
        """
        ssm_client = boto3.client("ssm")
        return cls(ssm_client)


    def update(self, title=None, description=None, status=None):
        """
        Update an OpsItem.
```

```
        :param title: The new OpsItem title.
        :param description: The new OpsItem description.
        :param status: The new OpsItem status.
        :return:
        """
        args = dict(OpsItemId=self.id)
        if title is not None:
            args["Title"] = title
        if description is not None:
            args["Description"] = description
        if status is not None:
            args["Status"] = status
        try:
            self.ssm_client.update_ops_item(**args)
        except ClientError as err:
            logger.error(
                "Couldn't update ops item %s. Here's why: %s: %s",
                self.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- 有关 API 的详细信息，请参阅《AWS SDK for Python（Boto3）API 参考》中的 UpdateOpsItem。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

将 **UpdatePatchBaseline** 与 CLI 配合使用

以下代码示例演示如何使用 UpdatePatchBaseline。

CLI

AWS CLI

示例 1：更新补丁基准

以下 update-patch-baseline 示例将指定的两个补丁（作为已拒绝的补丁）和一个补丁（作为已批准的补丁）添加到指定的补丁基准。

```
aws ssm update-patch-baseline \
        --baseline-id "pb-0123456789abcdef0" \
        --rejected-patches "KB2032276" "MS10-048" \
        --approved-patches "KB2124261"
```

输出：

```
{
    "BaselineId": "pb-0123456789abcdef0",
    "Name": "WindowsPatching",
    "OperatingSystem": "WINDOWS",
    "GlobalFilters": {
        "PatchFilters": []
    },
    "ApprovalRules": {
        "PatchRules": [
            {
                "PatchFilterGroup": {
                    "PatchFilters": [
                        {
                            "Key": "PRODUCT",
                            "Values": [
                                "WindowsServer2016"
                            ]
                        }
                    ]
                },
                "ComplianceLevel": "CRITICAL",
                "ApproveAfterDays": 0,
                "EnableNonSecurity": false
            }
        ]
    },
    "ApprovedPatches": [
        "KB2124261"
    ],
    "ApprovedPatchesComplianceLevel": "UNSPECIFIED",
    "ApprovedPatchesEnableNonSecurity": false,
    "RejectedPatches": [
        "KB2032276",
```

```
        "MS10-048"
    ],
    "RejectedPatchesAction": "ALLOW_AS_DEPENDENCY",
    "CreatedDate": 1550244180.465,
    "ModifiedDate": 1550244180.465,
    "Description": "Patches for Windows Servers",
    "Sources": []
}
```

示例 2：重命名补丁基准

以下 update-patch-baseline 示例重命名指定的补丁基准。

```
aws ssm update-patch-baseline \
    --baseline-id "pb-0713accee01234567" \
    --name "Windows-Server-2012-R2-Important-and-Critical-Security-Updates"
```

有关更多信息，请参阅《AWS Systems Manager 用户指南》中的更新或删除补丁基准 <https://docs.aws.amazon.com/systems-manager/latest/userguide/patch-baseline-update-or-delete.html>`__。

- 有关 API 详细信息，请参阅《AWS CLI Command Reference》中的 UpdatePatchBaseline。

PowerShell

Tools for PowerShell V4

示例 1：此示例将两个补丁（作为已拒绝的补丁）和一个补丁（作为已批准的补丁）添加到现有补丁基准。

```
Update-SSMPatchBaseline -BaselineId "pb-03da896ca3b68b639" -RejectedPatch
 "KB2032276","MS10-048" -ApprovedPatch "KB2124261"
```

输出：

```
ApprovalRules   : Amazon.SimpleSystemsManagement.Model.PatchRuleGroup
ApprovedPatches : {KB2124261}
BaselineId      : pb-03da896ca3b68b639
CreatedDate     : 3/3/2017 5:02:19 PM
Description     : Baseline containing all updates approved for production systems
GlobalFilters   : Amazon.SimpleSystemsManagement.Model.PatchFilterGroup
```

```
ModifiedDate    : 3/3/2017 5:22:10 PM
Name            : Production-Baseline
RejectedPatches : {KB2032276, MS10-048}
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考（V4）》中的 UpdatePatchBaseline。

Tools for PowerShell V5

示例 1：此示例将两个补丁（作为已拒绝的补丁）和一个补丁（作为已批准的补丁）添加到现有补丁基准。

```
Update-SSMPatchBaseline -BaselineId "pb-03da896ca3b68b639" -RejectedPatch
 "KB2032276","MS10-048" -ApprovedPatch "KB2124261"
```

输出：

```
ApprovalRules   : Amazon.SimpleSystemsManagement.Model.PatchRuleGroup
ApprovedPatches : {KB2124261}
BaselineId      : pb-03da896ca3b68b639
CreatedDate     : 3/3/2017 5:02:19 PM
Description     : Baseline containing all updates approved for production systems
GlobalFilters   : Amazon.SimpleSystemsManagement.Model.PatchFilterGroup
ModifiedDate    : 3/3/2017 5:22:10 PM
Name            : Production-Baseline
RejectedPatches : {KB2032276, MS10-048}
```

- 有关 API 详细信息，请参阅《AWS Tools for PowerShell Cmdlet 参考 (V5)》中的 UpdatePatchBaseline。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 将此服务与 AWS SDK 结合使用。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

AWS Systems Manager 中的日志记录和监控

监控是保持 AWS Systems Manager 和您的 AWS 解决方案的可靠性、可用性和性能的重要方面。您应从 AWS 解决方案的所有部分收集监控数据，以便调试出现的多点故障。在开始监控 Systems Manager 之前，您应创建一个监控计划，解决以下问题：

- 监控目的是什么？
- 您将监控哪些资源？
- 监控这些资源的频率如何？
- 您将使用哪些监控工具？
- 谁执行监控任务？
- 出现错误时应通知谁？

在定义监控目标并创建监控计划后，下一步是在您的环境中建立正常 Systems Manager 性能的基准。您应该在不同时间和不同负载条件下测量 Systems Manager 的性能。监控 Systems Manager 时，您应存储所收集的监控数据的历史记录。您可将当前 Systems Manager 性能与这些历史数据进行比较，这样可帮助您确定性能的正常模式和异常模式，找出解决问题的方法。

例如，您可以监控自动化工作流、补丁基准的应用、维护时段事件和配置合规性等操作的成功或失败。Automation 是 AWS Systems Manager 中的一项工具。

您还可以监控托管式节点的 CPU 利用率、磁盘 I/O 和网络利用率。如果性能低于您所建立的基准，则您可能需要重新配置或优化节点，以降低 CPU 利用率、改进磁盘 I/O 或减少网络流量。有关监控 EC2 实例的更多信息，请参阅《Amazon EC2 用户指南》中的监控 Amazon EC2。

主题

- 监控工具
- 将节点日志发送到统一的 CloudWatch Logs（CloudWatch 代理）
- 将 SSM Agent 日志发送到 CloudWatch Logs
- 监控您的变更请求事件
- 监控您的自动化
- 使用 Amazon CloudWatch 监控 Run Command 指标
- 使用 AWS CloudTrail 记录 AWS Systems Manager API 调用
- 使用 CloudWatch Logs 记录自动化操作输出

- 为 Run Command 配置 Amazon CloudWatch Logs
- 使用 Amazon EventBridge 监控 Systems Manager 事件
- 使用 Amazon SNS 通知监控 Systems Manager 状态更改

监控工具

本章中的内容提供了与使用可用于监控 Systems Manager 和其他 AWS 资源的工具有关的信息。有关工具的更完整的列表，请参阅 AWS Systems Manager 中的日志记录和监控。

将节点日志发送到统一的 CloudWatch Logs（CloudWatch 代理）

您可以配置和使用 Amazon CloudWatch 代理，以从节点收集指标和日志，而不是使用 AWS Systems Manager 代理 (SSM Agent) 完成此类任务。CloudWatch 代理使您能够收集到比使用 SSM Agent 收集时更多的 Amazon EC2 实例相关指标。此外，您还可以使用 CloudWatch 代理从本地服务器收集指标。

您也可以将代理配置设置存储在 Systems Manager Parameter Store 中，以便与 CloudWatch 代理一起使用。Parameter Store 是 AWS Systems Manager 中的一项工具。

> **Note**
>
> AWS Systems Manager 支持从 SSM Agent 迁移到统一的 CloudWatch 代理，以便仅在 64 位版本的 Windows 上收集日志和指标。有关在其他操作系统中设置统一 CloudWatch 代理的信息，以及有关使用 CloudWatch 代理的完整信息，请参阅 Amazon CloudWatch 用户指南中的使用 CloudWatch 代理从 Amazon EC2 实例和本地服务器收集指标和日志。
> 您可以在其他受支持的操作系统上使用 CloudWatch 代理，但无法使用 Systems Manager 执行工具迁移。

SSM Agent 将有关执行、计划操作、错误和运行状况的信息写入每个节点上的日志文件。手动连接到节点以查看日志文件并对 SSM Agent 的问题进行故障排除是非常耗时的工作。为更高效地监控节点，可以配置 SSM Agent 本身或 CloudWatch 代理，以将此日志数据发送到 Amazon CloudWatch Logs。

> **Important**
>
> 统一的 CloudWatch 代理已取代 SSM Agent，作为将日志数据发送到 Amazon CloudWatch Logs 的工具。不支持 SSM Agent aws:cloudWatch 插件。我们建议仅将统一的 CloudWatch 代理用于您的日志收集过程。有关更多信息，请参阅以下主题：
>
> - 将节点日志发送到统一的 CloudWatch Logs（CloudWatch 代理）
> - 将 Windows Server 节点日志收集迁移到 CloudWatch 代理
> - 《Amazon CloudWatch 用户指南》中的使用 CloudWatch 代理收集指标、日志和跟踪信息。

借助 CloudWatch Logs，可以实时监控日志数据，可以通过创建一个或多个指标筛选器来搜索和筛选日志数据，可以归档历史数据并在需要时进行检索。有关 CloudWatch Logs 的更多信息，请参阅 Amazon CloudWatch Logs 用户指南。

配置代理将日志数据发送到 Amazon CloudWatch Logs 有以下好处：

- 适用于所有 SSM Agent 日志文件的集中日志文件存储。
- 实现对文件的更快访问，以调查错误。
- 无限日志文件保留（可配置）。
- 无论节点的状态如何，都可以维护和访问日志。
- 可以访问其他 CloudWatch 功能（例如指标和告警）。

有关监控 Session Manager 活动的信息，请参阅 记录会话活动 和 启用和禁用会话日志记录。

将 Windows Server 节点日志收集迁移到 CloudWatch 代理

如果在受支持的 Windows Server 节点上使用 SSM Agent 将 SSM Agent 日志文件发送到 Amazon CloudWatch Logs，则可以使用 Systems Manager 从 SSM Agent 迁移到作为日志收集工具的 CloudWatch 代理，并迁移配置设置。

CloudWatch 代理在 32 位版本的 Windows Server 上不受支持。

对于 Windows Server 的 64 位 Amazon EC2 实例，您可以自动或手动执行向 CloudWatch 代理的迁移。对于本地服务器和虚拟机，必须手动执行该过程。

> **Note**
>
> 迁移期间，发送到 CloudWatch 的数据可能会中断或重复发送。迁移完成后，指标和日志数据会再次准确记录到 CloudWatch 中。

建议先在有限数量的节点上测试迁移，再将整个机群迁移到 CloudWatch 代理。迁移后，如果更倾向于使用 SSM Agent 进行日志收集，可以改为使用它。

> **Important**
>
> 在以下情况下，不能使用本主题中介绍的步骤向 CloudWatch 代理迁移：
>
> - SSM Agent 的现有配置指定多个区域。
> - SSM Agent 的现有配置指定多组访问/私密密钥凭证。
>
> 在上述情况下，必须在 SSM Agent 中关闭日志收集并安装 CloudWatch 代理，而不执行迁移过程。有关更多信息，请参阅 Amazon CloudWatch 用户指南中的以下主题：
>
> - 安装 CloudWatch 代理
> - 在本地部署服务器上安装 CloudWatch 代理

开始前的准备工作

在开始迁移到 CloudWatch 代理进行日志收集前，请确保将对其执行迁移的节点满足以下要求：

- 操作系统为 64 位版本的 Windows Server。
- 节点上安装了 SSM Agent 2.2.93.0 或更高版本。
- 已在节点上配置 SSM Agent 进行监控。

主题

- 自动迁移到 CloudWatch 代理
- 手动迁移到 CloudWatch 代理

自动迁移到 CloudWatch 代理

仅对于 Windows Server 的 Amazon EC2 实例，您可以使用 AWS Systems Manager 控制台或 AWS Command Line Interface (AWS CLI) 自动迁移到 CloudWatch 代理以作为日志收集工具。

> **Note**
>
> AWS Systems Manager 支持从 SSM Agent 迁移到统一的 CloudWatch 代理，以便仅在 64 位版本的 Windows 上收集日志和指标。有关在其他操作系统中设置统一 CloudWatch 代理的信息，以及有关使用 CloudWatch 代理的完整信息，请参阅 Amazon CloudWatch 用户指南中的使用 CloudWatch 代理从 Amazon EC2 实例和本地服务器收集指标和日志。
> 您可以在其他受支持的操作系统上使用 CloudWatch 代理，但无法使用 Systems Manager 执行工具迁移。

迁移成功后，请在 CloudWatch 中检查结果，以确保正在接收预期的指标、日志或 Windows 事件日志。如果对结果满意，可以选择将 CloudWatch 代理配置设置存储到 Parameter Store 中。如果迁移不成功或者结果不符合预期，可以尝试回滚到使用 SSM Agent 进行日志收集。

> **Note**
>
> 如果需要迁移包含 {hostname} 条目的源配置文件，则请注意，在迁移完成后，{hostname} 条目会更改字段的值。例如，假设下面的 "LogStream": "{hostname}" 条目映射到名为 MyLogServer001 的服务器。
>
> ```
> {
> "Id": "CloudWatchIISLogs",
> "FullName":
>  "AWS.EC2.Windows.CloudWatch.CloudWatchLogsOutput,AWS.EC2.Windows.CloudWatch",
> "Parameters": {
> "AccessKey": "",
> "SecretKey": "",
> "Region": "us-east-1",
> "LogGroup": "Production-Windows-IIS",
> "LogStream": "{hostname}"
>      }
> }
> ```

迁移完成后，该条目将映射到一个域，例如 ip-11-1-1-11.production。ExampleCompany.com。要保留本地主机名值，请指定 `{local_hostname}` 而不是 `{hostname}`。

自动迁移到 CloudWatch 代理（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Run Command，然后选择 Run command（运行命令）。
3. 在 Command document（命令文档）列表中，请选择 `AmazonCloudWatch-MigrateCloudWatchAgent`。
4. 对于 Status（状态），选择 Enabled（已启用）。
5. 在 Targets（目标）部分中，通过指定标签、手动选择实例或边缘设备或指定资源组，选择要在其上运行此操作的托管式节点。

   > **Tip**
   >
   > 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

6. 对于 Rate control（速率控制）：

   - 对于 Concurrency（并发），请指定要同时运行该命令的托管式节点的数量或百分比。

     > **Note**
     >
     > 如果您通过指定应用于托管式节点的标签或指定 AWS Resource Groups 来选择目标，但不确定有多少个托管式节点已被设为目标，则可通过指定百分比来限制可同时运行该文档的目标的数量。

   - 对于 Error threshold（错误阈值），请指定当命令在一定数量或百分比的节点上失败后，何时在其他托管式节点上停止运行该命令。例如，如果您指定三个错误，Systems Manager 将在收到第四个错误时停止发送该命令。仍在处理该命令的托管式节点也可能发送错误。

7. （可选）对于 输出选项，要将命令输出保存到文件，请选中 将命令输出写入 S3 存储桶 框。在输入框中输入存储桶和前缀（文件夹）名称。

> **Note**
>
> 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给实例的实例配置文件（适用于 EC2 实例）或 IAM 服务角色（混合激活的计算机）的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确保与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

8. 在 SNS 通知部分，如果需要发送有关命令执行状态的通知，请选中 Enable SNS notifications（启用 SNS 通知）复选框。

   有关为 Run Command 配置 Amazon SNS 通知的更多信息，请参阅 使用 Amazon SNS 通知监控 Systems Manager 状态更改。

9. 选择 Run（运行）。

自动迁移到 CloudWatch 代理 (AWS CLI)

- 运行以下命令。

```
aws ssm send-command --document-name AmazonCloudWatch-MigrateCloudWatchAgent --
targets Key=instanceids,Values=ID1,ID2,ID3
```

ID1、*ID2* 和 *ID3* 表示要更新的节点的 ID，例如 i-02573cafcfEXAMPLE。

手动迁移到 CloudWatch 代理

对于本地 Windows Server 节点或适用于 Windows Server 的 Amazon EC2 实例，请按照以下步骤将日志收集手动迁移到 Amazon CloudWatch 代理。

> **Note**
>
> 如果需要迁移包含 {hostname} 条目的源配置文件，则请注意，在迁移完成后，{hostname} 条目会更改字段的值。例如，假设下面的 "LogStream": "{hostname}" 条目映射到名为 MyLogServer001 的服务器。

```
{
"Id": "CloudWatchIISLogs",
```

```
"FullName":
 "AWS.EC2.Windows.CloudWatch.CloudWatchLogsOutput,AWS.EC2.Windows.CloudWatch",
"Parameters": {
"AccessKey": "",
"SecretKey": "",
"Region": "us-east-1",
"LogGroup": "Production-Windows-IIS",
"LogStream": "{hostname}"
     }
}
```

迁移完成后，该条目将映射到一个域，例如 ip-11-1-1-11.production.ExampleCompany.com。要保留本地主机名值，请指定 `{local_hostname}` 而不是 `{hostname}`。

第一步：安装 CloudWatch 代理（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Run Command，然后选择 Run command（运行命令）。
3. 在 Command document（命令文档）列表中，请选择 `AWS-ConfigureAWSPackage`。
4. 对于 Action（操作），选择 `Install`。
5. 对于名称，请输入 **`AmazonCloudWatchAgent`**。
6. 对于 Version（版本），输入 **`latest`**（如果在预设情况下未提供）。
7. 在 Targets（目标）部分中，通过指定标签、手动选择实例或边缘设备或指定资源组，选择要在其上运行此操作的托管式节点。

   > **Tip**
   >
   > 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

8. 对于 Rate control（速率控制）：
   - 对于 Concurrency（并发），请指定要同时运行该命令的托管式节点的数量或百分比。

> **Note**
>
> 如果您通过指定应用于托管式节点的标签或指定 AWS Resource Groups 来选择目标，但不确定有多少个托管式节点已被设为目标，则可通过指定百分比来限制可同时运行该文档的目标的数量。

- 对于 Error threshold（错误阈值），请指定当命令在一定数量或百分比的节点上失败后，何时在其他托管式节点上停止运行该命令。例如，如果您指定三个错误，Systems Manager 将在收到第四个错误时停止发送该命令。仍在处理该命令的托管式节点也可能发送错误。

9. （可选）对于 输出选项，要将命令输出保存到文件，请选中 将命令输出写入 S3 存储桶 框。在输入框中输入存储桶和前缀（文件夹）名称。

> **Note**
>
> 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给实例的实例配置文件（适用于 EC2 实例）或 IAM 服务角色（混合激活的计算机）的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确保与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

10. 在 SNS 通知部分，如果需要发送有关命令执行状态的通知，请选中 Enable SNS notifications（启用 SNS 通知）复选框。

    有关为 Run Command 配置 Amazon SNS 通知的更多信息，请参阅 使用 Amazon SNS 通知监控 Systems Manager 状态更改。

11. 选择 Run（运行）。

第二步：更新配置数据 JSON 格式

- 要更新 CloudWatch 代理的现有配置设置的 JSON 格式，请使用 Run Command（AWS Systems Manager 中的一项工具），或直接使用 RDP 连接登录节点，然后在该节点上运行以下 Windows PowerShell 命令（一次运行一条命令）。

```
cd ${Env:ProgramFiles}\\Amazon\\AmazonCloudWatchAgent
```

```
.\\amazon-cloudwatch-agent-config-wizard.exe --isNonInteractiveWindowsMigration
```

{Env:ProgramFiles} 表示可找到包含 CloudWatch 代理的 Amazon 目录的位置，通常为 C:\Program Files。

第三步：配置并启动 CloudWatch 代理（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Run Command，然后选择 Run command（运行命令）。
3. 在 Command document（命令文档）列表中，请选择 AWS-RunPowerShellScript。
4. 对于 Commands（命令），输入以下两条命令。

```
cd ${Env:ProgramFiles}\Amazon\AmazonCloudWatchAgent
```

```
.\amazon-cloudwatch-agent-ctl.ps1 -a fetch-config -m ec2 -c file:config.json -s
```

{Env:ProgramFiles} 表示可找到包含 CloudWatch 代理的 Amazon 目录的位置，通常为 C:\Program Files。

5. 在 Targets（目标）部分中，通过指定标签、手动选择实例或边缘设备或指定资源组，选择要在其上运行此操作的托管式节点。

> **Tip**
>
> 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

6. 对于 Rate control（速率控制）：

   - 对于 Concurrency（并发），请指定要同时运行该命令的托管式节点的数量或百分比。

> **Note**
>
> 如果您通过指定应用于托管式节点的标签或指定 AWS Resource Groups 来选择目标，但不确定有多少个托管式节点已被设为目标，则可通过指定百分比来限制可同时运行该文档的目标的数量。

- 对于 Error threshold（错误阈值），请指定当命令在一定数量或百分比的节点上失败后，何时在其他托管式节点上停止运行该命令。例如，如果您指定三个错误，Systems Manager 将在收到第四个错误时停止发送该命令。仍在处理该命令的托管式节点也可能发送错误。

7. （可选）对于 输出选项，要将命令输出保存到文件，请选中 将命令输出写入 S3 存储桶 框。在输入框中输入存储桶和前缀（文件夹）名称。

> **Note**
>
> 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给实例的实例配置文件（适用于 EC2 实例）或 IAM 服务角色（混合激活的计算机）的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确保与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

8. 在 SNS 通知部分，如果需要发送有关命令执行状态的通知，请选中 Enable SNS notifications（启用 SNS 通知）复选框。

   有关为 Run Command 配置 Amazon SNS 通知的更多信息，请参阅 使用 Amazon SNS 通知监控 Systems Manager 状态更改。

9. 选择 Run（运行）。

第四步：在 SSM Agent（控制台）中关闭日志收集

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Run Command，然后选择 Run command（运行命令）。
3. 在 Command document（命令文档） 列表中，请选择 `AWS-ConfigureCloudWatch`。
4. 对于 Status（状态），请选择 Disabled（已禁用）。
5. 在 Targets（目标）部分中，通过指定标签、手动选择实例或边缘设备或指定资源组，选择要在其上运行此操作的托管式节点。

> **Tip**
>
> 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

6. 对于 Status（状态），请选择 Disabled。
7. 对于 Rate control（速率控制）：
   - 对于 Concurrency（并发），请指定要同时运行该命令的托管式节点的数量或百分比。

     > **Note**
     >
     > 如果您通过指定应用于托管式节点的标签或指定 AWS Resource Groups 来选择目标，但不确定有多少个托管式节点已被设为目标，则可通过指定百分比来限制可同时运行该文档的目标的数量。

   - 对于 Error threshold（错误阈值），请指定当命令在一定数量或百分比的节点上失败后，何时在其他托管式节点上停止运行该命令。例如，如果您指定三个错误，Systems Manager 将在收到第四个错误时停止发送该命令。仍在处理该命令的托管式节点也可能发送错误。
8. （可选）对于 输出选项，要将命令输出保存到文件，请选中 将命令输出写入 S3 存储桶 框。在输入框中输入存储桶和前缀（文件夹）名称。

   > **Note**
   >
   > 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给实例的实例配置文件（适用于 EC2 实例）或 IAM 服务角色（混合激活的计算机）的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确保与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

9. 在 SNS 通知部分，如果需要发送有关命令执行状态的通知，请选中 Enable SNS notifications（启用 SNS 通知）复选框。

   有关为 Run Command 配置 Amazon SNS 通知的更多信息，请参阅 使用 Amazon SNS 通知监控 Systems Manager 状态更改。
10. 选择 Run（运行）。

    完成上述步骤后，请在 CloudWatch 中检查日志，以验证并确保正在接收预期的指标、日志或 Windows 事件日志。如果对结果满意，可以选择将 CloudWatch 代理配置设置存储到 Parameter Store 中。如果迁移不成功或者结果不符合预期，可以回滚到使用 SSM Agent 进行日志收集。

将 CloudWatch 代理配置设置存储到 Parameter Store 中

您可以将 CloudWatch 代理配置文件的内容存储到 Parameter Store 中。通过将此配置数据保存在一个参数中，多个节点可以从该参数获取配置设置，您不必再在节点上创建或手动更新配置文件。例如，您可以使用 Run Command 将该参数的内容写入多个节点上的配置文件，或使用State Manager（AWS Systems Manager 中的一项工具）帮助避免一群节点间的 CloudWatch 代理配置设置出现配置偏差。

运行 CloudWatch 代理配置向导时，可以选择让该向导将配置设置保存为 Parameter Store 中的新参数。有关运行 CloudWatch 代理配置向导的信息，请参阅 Amazon CloudWatch 用户指南中的使用向导创建 CloudWatch 代理配置文件。

如果在运行该向导时没有选择将设置保存为参数的选项，或者手动创建了 CloudWatch 代理配置文件，则可以在以下文件中检索要在节点上保存为参数的数据。

```
${Env:ProgramFiles}\Amazon\AmazonCloudWatchAgent\config.json
```

{Env:ProgramFiles} 表示可找到包含 CloudWatch 代理的 Amazon 目录的位置，通常为 `C:\Program Files`。

建议在该节点以外的位置保留此文件中 JSON 的备份。

有关创建参数的信息，请参阅 在 Systems Manager 中创建 Parameter Store 参数。

有关 CloudWatch 代理的更多信息，请参阅 Amazon CloudWatch 用户指南中的使用 CloudWatch 代理从 Amazon EC2 实例和本地服务器收集指标和日志。

回滚到使用 SSM Agent 进行日志收集

如果需要恢复为使用 SSM Agent 进行日志收集，请执行以下步骤。

第一步：从 SSM Agent 检索配置数据

1. 在需要恢复为使用 SSM Agent 收集日志的节点上，找到 SSM Agent 配置文件的内容。此 JSON 文件通常位于以下位置：

   ```
   ${Env:ProgramFiles}\\Amazon\\SSM\\Plugins\\awsCloudWatch\
   \AWS.EC2.Windows.CloudWatch.json
   ```

   {Env:ProgramFiles} 表示可找到 Amazon 目录的位置，通常为 `C:\Program Files`。
2. 将这些数据复制到一个文本文件中，以便在后面的步骤中使用。

建议在该节点以外的位置存储 JSON 的备份。

第二步：卸载 CloudWatch 代理（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Run Command，然后选择 Run command（运行命令）。
3. 在 Command document（命令文档）列表中，请选择 `AWS-ConfigureAWSPackage`。
4. 对于 Action（操作），选择 Uninstall（卸载）。
5. 对于 Name（名称），请输入 **`AmazonCloudWatchAgent`**。
6. 在 Targets（目标）部分中，通过指定标签、手动选择实例或边缘设备或指定资源组，选择要在其上运行此操作的托管式节点。

   > **Tip**
   >
   > 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

7. 对于 Rate control（速率控制）：

   - 对于 Concurrency（并发），请指定要同时运行该命令的托管式节点的数量或百分比。

     > **Note**
     >
     > 如果您通过指定应用于托管式节点的标签或指定 AWS Resource Groups 来选择目标，但不确定有多少个托管式节点已被设为目标，则可通过指定百分比来限制可同时运行该文档的目标的数量。

   - 对于 Error threshold（错误阈值），请指定当命令在一定数量或百分比的节点上失败后，何时在其他托管式节点上停止运行该命令。例如，如果您指定三个错误，Systems Manager 将在收到第四个错误时停止发送该命令。仍在处理该命令的托管式节点也可能发送错误。

8. （可选）对于 输出选项，要将命令输出保存到文件，请选中 将命令输出写入 S3 存储桶 框。在输入框中输入存储桶和前缀（文件夹）名称。

> **Note**
>
> 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给实例的实例配置文件（适用于 EC2 实例）或 IAM 服务角色（混合激活的计算机）的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确保与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

9. 在 SNS 通知部分，如果需要发送有关命令执行状态的通知，请选中 Enable SNS notifications（启用 SNS 通知）复选框。

   有关为 Run Command 配置 Amazon SNS 通知的更多信息，请参阅 使用 Amazon SNS 通知监控 Systems Manager 状态更改。

10. 选择 Run（运行）。

第三步：在 SSM Agent（控制台）中重新打开日志收集

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Run Command，然后选择 Run command（运行命令）。
3. 在 Command document（命令文档）列表中，请选择 `AWS-ConfigureCloudWatch`。
4. 对于 Status（状态），请选择 `Enabled`。
5. 对于 Properties（属性），将保存的旧配置数据的内容粘贴到文本文件中。
6. 在 Targets（目标）部分中，通过指定标签、手动选择实例或边缘设备或指定资源组，选择要在其上运行此操作的托管式节点。

> **Tip**
>
> 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

7. 对于 Rate control（速率控制）：

   - 对于 Concurrency（并发），请指定要同时运行该命令的托管式节点的数量或百分比。

> **Note**
>
> 如果您通过指定应用于托管式节点的标签或指定 AWS Resource Groups 来选择目标，但不确定有多少个托管式节点已被设为目标，则可通过指定百分比来限制可同时运行该文档的目标的数量。

- 对于 Error threshold（错误阈值），请指定当命令在一定数量或百分比的节点上失败后，何时在其他托管式节点上停止运行该命令。例如，如果您指定三个错误，Systems Manager 将在收到第四个错误时停止发送该命令。仍在处理该命令的托管式节点也可能发送错误。

8. （可选）对于 输出选项，要将命令输出保存到文件，请选中 将命令输出写入 S3 存储桶 框。在输入框中输入存储桶和前缀（文件夹）名称。

> **Note**
>
> 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给实例的实例配置文件（适用于 EC2 实例）或 IAM 服务角色（混合激活的计算机）的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确保与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

9. 在 SNS 通知部分，如果需要发送有关命令执行状态的通知，请选中 Enable SNS notifications（启用 SNS 通知）复选框。

   有关为 Run Command 配置 Amazon SNS 通知的更多信息，请参阅 使用 Amazon SNS 通知监控 Systems Manager 状态更改。

10. 选择 Run（运行）。

将 SSM Agent 日志发送到 CloudWatch Logs

AWS Systems Manager Agent（SSM Agent）是在为 Systems Manager 配置的 EC2 实例、边缘设备、本地服务器和虚拟机（VM）上运行的 Amazon 软件。SSM Agent 可在云中处理来自 Systems Manager 服务的请求，并按请求中指定的方式配置计算机。有关 SSM Agent 的更多信息，请参阅 使用 SSM Agent。

此外，使用以下步骤，您可以配置 SSM Agent 将日志数据发送到 Amazon CloudWatch Logs。

开始前的准备工作

在 CloudWatch Logs 中创建日志组。有关更多信息，请参阅 Amazon CloudWatch Logs 用户指南中的 CloudWatch Logs 入门。

配置 SSM Agent 将日志发送到 CloudWatch

1. 登录节点并找到以下文件：

   Linux

   在大多数 Linux 节点类型上：`/etc/amazon/ssm/seelog.xml.template`。

   在 Ubuntu Server 20.10 STR & 20.04、18.04 和 16.04 LTS 上：`/snap/amazon-ssm-agent/current/seelog.xml.template`

   macOS

   `/opt/aws/ssm/seelog.xml.template`

   Windows

   `%ProgramFiles%\Amazon\SSM\seelog.xml.template`

2. 将文件名从 `seelog.xml.template` 更改为 `seelog.xml`

   > **Note**
   >
   > 在 Ubuntu Server 20.10 STR 和 20.04、18.04 及 16.04 LTS 上，必须在目录 `/etc/amazon/ssm/` 中创建文件 `seelog.xml`。可以通过运行以下命令来创建此目录和文件。
   >
   > ```
   > sudo mkdir -p /etc/amazon/ssm
   > ```
   >
   > ```
   > sudo cp -pr /snap/amazon-ssm-agent/current/* /etc/amazon/ssm
   > ```
   >
   > ```
   > sudo cp -p /etc/amazon/ssm/seelog.xml.template /etc/amazon/ssm/seelog.xml
   > ```

3. 用文本编辑器打开 `seelog.xml` 文件并找到以下部分。

   Linux and macOS

   ```
   <outputs formatid="fmtinfo">
   ```

```
   <console formatid="fmtinfo"/>
   <rollingfile type="size" filename="/var/log/amazon/ssm/amazon-ssm-agent.log"
 maxsize="30000000" maxrolls="5"/>
   <filter levels="error,critical" formatid="fmterror">
      <rollingfile type="size" filename="/var/log/amazon/ssm/errors.log"
 maxsize="10000000" maxrolls="5"/>
   </filter>
</outputs>
```

Windows

```
<outputs formatid="fmtinfo">
   <console formatid="fmtinfo"/>
   <rollingfile type="size" maxrolls="5" maxsize="30000000"
 filename="{{LOCALAPPDATA}}\Amazon\SSM\Logs\amazon-ssm-agent.log"/>
   <filter formatid="fmterror" levels="error,critical">
      <rollingfile type="size" maxrolls="5" maxsize="10000000"
 filename="{{LOCALAPPDATA}}\Amazon\SSM\Logs\errors.log"/>
   </filter>
</outputs>
```

4. 编辑文件，并在结束 </filter> 标签后添加 custom name 元素。在以下示例中，指定的 custom name 名称为 `cloudwatch_receiver`。

   Linux and macOS

```
<outputs formatid="fmtinfo">
   <console formatid="fmtinfo"/>
   <rollingfile type="size" filename="/var/log/amazon/ssm/amazon-ssm-agent.log"
 maxsize="30000000" maxrolls="5"/>
   <filter levels="error,critical" formatid="fmterror">
      <rollingfile type="size" filename="/var/log/amazon/ssm/errors.log"
 maxsize="10000000" maxrolls="5"/>
   </filter>
   <custom name="cloudwatch_receiver" formatid="fmtdebug" data-log-group="your-
CloudWatch-log-group-name"/>
</outputs>
```

   Windows

```
<outputs formatid="fmtinfo">
   <console formatid="fmtinfo"/>
```

```
    <rollingfile type="size" maxrolls="5" maxsize="30000000"
 filename="{{LOCALAPPDATA}}\Amazon\SSM\Logs\amazon-ssm-agent.log"/>
    <filter formatid="fmterror" levels="error,critical">
       <rollingfile type="size" maxrolls="5" maxsize="10000000"
 filename="{{LOCALAPPDATA}}\Amazon\SSM\Logs\errors.log"/>
    </filter>
    <custom name="cloudwatch_receiver" formatid="fmtdebug" data-log-group="your-
CloudWatch-log-group-name"/>
</outputs>
```

5. 保存您的更改，然后重新启动 SSM Agent 或节点。
6. 访问 https://console.aws.amazon.com/cloudwatch/，打开 CloudWatch 控制台。
7. 在导航窗格中，选择 Log groups (日志组)，然后选择日志组的名称。

> **Tip**
>
> SSM Agent 日志文件数据的日志流按节点 ID 进行整理。

监控您的变更请求事件

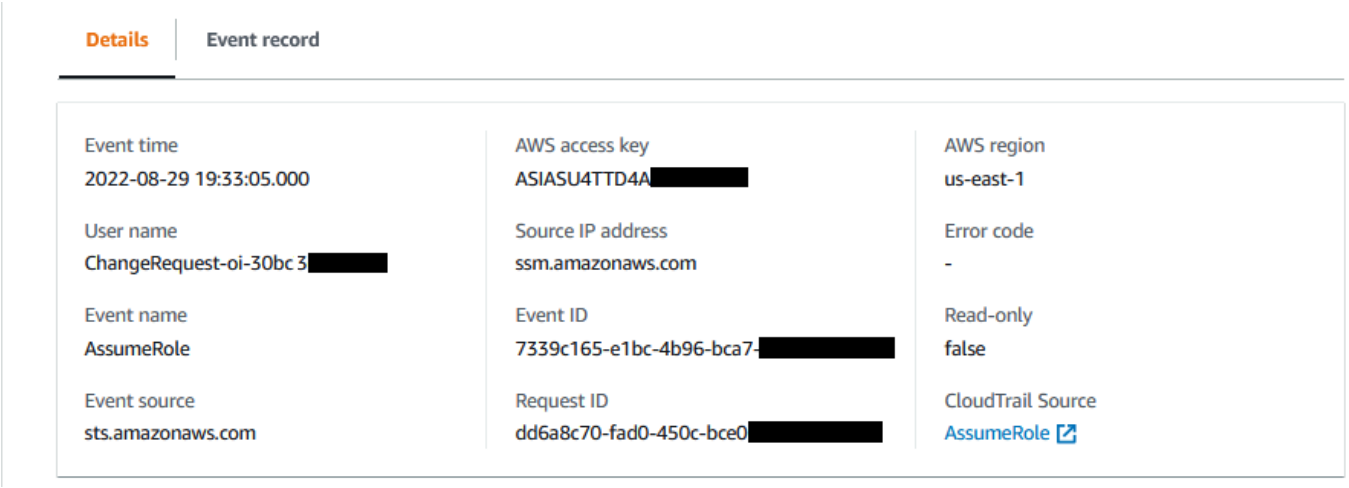
打开与 AWS CloudTrail Lake 的集成并创建事件数据存储后，您可以查看有关在您的账户或组织中运行的变更请求的可审批详细信息。它包括诸如以下详细信息：

- 发起变更请求的用户的身份
- 在其中进行更改的 AWS 区域
- 请求的源 IP 地址
- 用于请求的 AWS 访问密钥
- 针对变更请求运行 API 操作
- 这些操作所包含的请求参数
- 在此过程中更新的资源

以下是在 AWS CloudTrail Lake 中创建事件数据存储后可以查看的变更请求事件详细信息示例。

Details

下图显示了有关 Details（详细信息）选项卡上可用的变更请求的高级别信息。这些详细信息包括诸如变更请求操作开始的时间、发起变更请求的用户的 ID、受影响的 AWS 区域 以及与请求关联的事件 ID 和请求 ID 等信息。



Event record

下图显示了 CloudTrail Lake 为变更请求事件提供的 JSON 内容的结构。在变更请求的 Event record（事件记录）选项卡上提供此数据。



> **Important**
>
> 如果您将 Change Manager 用于组织，则可以在登录到 Change Manager 的管理账户或委派管理员账户时完成以下过程。
> 但是，要使用委派管理员账户完成这些步骤，必须为 CloudTrail 和 Change Manager 指定相同的委派管理员账户。
> 登录 Change Manager 的管理账户时，可以在 CloudTrail 设置页面上添加或更改 CloudTrail 的委派管理员账户。必须先完成此操作，然后委派管理员才能创建事件数据存储以供整个组织使用。

从 Change Manager 打开 CloudTrail Lake 事件跟踪

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 Change Manager。
3. 选择 Requests (请求) 选项卡。
4. 选择任何现有的变更请求，然后选择 Associated events（关联的事件）选项卡。
5. 选择 Enable CloudTrail Lake（启用 CloudTrail Lake）。
6. 按照《AWS CloudTrail User Guide》中的 Create an event data store for CloudTrail events 中的步骤进行操作。

   要确保存储变更请求的事件数据，请在完成步骤时进行以下选择：

   - 对于事件类型，请将默认的 AWS 事件和 CloudTrail 事件保持选中状态。
   - 如果将 Change Manager 用于组织，请选择为我组织中的所有账户启用。
   - 对于管理事件，请勿清除写入复选框。

   创建事件数据存储时选择的其他选项不会影响变更请求的事件数据存储。

监控您的自动化

指标是 Amazon CloudWatch 中的基本概念。指标表示一个发布到 CloudWatch 并且按时间排序的数据点集。可将指标视为要监控的变量，而数据点代表该变量随时间变化的值。

Automation 是 AWS Systems Manager 中的一项工具。Systems Manager 向 CloudWatch 发布了有关 Automation 使用情况的指标。这使您能够根据这些指标设置告警。

要在 CloudWatch 控制台中查看 Automation 指标

1. 通过 https://console.aws.amazon.com/cloudwatch/ 打开 CloudWatch 控制台。
2. 在导航窗格中，选择指标。
3. 请选择 SSM。
4. 在指标选项卡上，选择使用情况，然后选择按 AWS 资源。
5. 请在指标列表附近的搜索框中，输入 SSM。

要使用 AWS CLI 查看 Automation 指标

打开命令提示符窗口并使用以下命令。

```
aws cloudwatch list-metrics \
    --namespace "AWS/Usage"
```

Automation 指标

Systems Manager 将以下 Automation 指标发送到 CloudWatch。

指标	描述
ConcurrentAutomationUsage	在当前 AWS 账户 和 AWS 区域 中同时运行的自动化数量。
QueuedAutomationUsage	当前排队但尚未启动且状态为 Pending 的自动化数量。

有关使用 CloudWatch 指标的更多信息，请参阅《Amazon CloudWatch 用户指南》中的以下主题：

- Metrics
- 使用 Amazon CloudWatch 指标
- 使用 Amazon CloudWatch 告警

使用 Amazon CloudWatch 监控 Run Command 指标

指标是 Amazon CloudWatch 中的基本概念。指标表示一个发布到 CloudWatch 并且按时间排序的数据点集。可将指标视为要监控的变量，而数据点代表该变量随时间变化的值。

AWS Systems Manager 会将与 Run Command 命令的状态有关的指标发布到 CloudWatch，让您能够根据这些指标设置警报。Run Command 是 AWS Systems Manager 中的一项工具。这些统计数据会被长时间记录，以便您可以访问历史信息并更好地了解 AWS 账户中命令运行的成功率。

可以跟踪其指标的命令的终端状态值包括 `Success`、`Failed` 和 `Delivery Timed Out`。例如，对于设置为每小时运行一次的 SSM Command 文档，您可以配置告警，以便在其中任何一小时未报告 `Success` 的状态时向您发送通知。有关命令状态值的更多信息，请参阅 了解命令状态。

在 CloudWatch 控制台中查看指标

1. 通过 https://console.aws.amazon.com/cloudwatch/ 打开 CloudWatch 控制台。
2. 在导航窗格中，选择指标。
3. 在按 AWS 服务告警区域，对于服务，选择 SSM-Run Command。

使用 AWS CLI 查看指标

打开命令提示符窗口并使用以下命令。

```
aws cloudwatch list-metrics --namespace "AWS/SSM-RunCommand"
```

要列出所有可用的指标，请使用以下命令。

```
aws cloudwatch list-metrics
```

Systems Manager Run Command 指标和维度

Systems Manager 每分钟向 CloudWatch 发送一次 Run Command 命令指标。

Systems Manager 将以下命令指标发送到 CloudWatch。

> **Note**
>
> 这些指标使用 `Count` 作为单位，因此 `Sum` 和 `SampleCount` 是最有用的统计数据。

指标	描述
CommandsDeliveryTimedOut	终端状态为 Delivery Timed Out 的命令的数量。
CommandsFailed	终端状态为 Failed 的命令的数量。
CommandsSucceeded	终端状态为 Success 的命令的数量。

有关使用 CloudWatch 指标的更多信息，请参阅 Amazon CloudWatch 用户指南中的以下主题：

- Metrics
- 使用 Amazon CloudWatch 指标
- 使用 Amazon CloudWatch 告警

使用 AWS CloudTrail 记录 AWS Systems Manager API 调用

AWS Systems Manager 与 AWS CloudTrail 集成，后者是提供用户、角色或 AWS 服务 所执行操作记录的服务。CloudTrail 将 Systems Manager 的所有 API 调用作为事件捕获。捕获的调用包含来自 Systems Manager 控制台和代码的 Systems Manager API 操作调用。借助通过 CloudTrail 收集的信息，您可以确定向 Systems Manager 发出哪些请求、发出请求的 IP 地址、请求的发出时间以及其他详细信息。

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

- 请求是使用根用户凭证还是用户凭证发出的。
- 请求是否代表 IAM Identity Center 用户发出。
- 请求是使用角色还是联合用户的临时安全凭证发出的。
- 请求是否由其他 AWS 服务 发出。

当您创建账户并可以自动访问 CloudTrail 事件历史记录时，CloudTrail 在您的 AWS 账户 中处于活动状态。CloudTrail 事件历史记录提供对 AWS 区域 中过去 90 天的已记录管理事件的可查看、可搜索、可下载和不可变记录。有关更多信息，请参见《AWS CloudTrail 用户指南》的 使用 CloudTrail 事件历史记录。查看事件历史记录不会收取 CloudTrail 费用。

要持续记录您的 AWS 账户 过去 90 天的事件，请创建跟踪或 CloudTrail Lake 事件数据存储。

CloudTrail 跟踪

通过跟踪记录，CloudTrail 可将日志文件传送至 Simple Storage Service (Amazon S3) 存储桶。使用 AWS Management Console 创建的所有跟踪均具有多区域属性。您可以通过使用 AWS CLI 创建单区域或多区域跟踪。建议创建多区域跟踪，因为您可记录您账户中的所有 AWS 区域 的活动。如果您创建单区域跟踪，则只能查看跟踪的 AWS 区域 中记录的事件。有关跟踪的更多信息，请参阅《AWS CloudTrail 用户指南》中的为您的 AWS 账户 创建跟踪和为组织创建跟踪。

通过创建跟踪，您可以从 CloudTrail 免费向您的 Amazon S3 存储桶传送一份正在进行的管理事件的副本，但会收取 Amazon S3 存储费用。有关 CloudTrail 定价的更多信息，请参阅 AWS CloudTrail 定价。有关 Amazon S3 定价的信息，请参阅 Amazon S3 定价。

CloudTrail Lake 事件数据存储

CloudTrail Lake 允许您对事件运行基于 SQL 的查询。CloudTrail Lake 可将基于行的 JSON 格式的现有事件转换为 Apache ORC 格式。ORC 是一种针对快速检索数据进行优化的列式存储格式。事件将被聚合到事件数据存储中，它是基于您通过应用高级事件选择器选择的条件的不可变的事件集合。应用于事件数据存储的选择器用于控制哪些事件持续存在并可供您查询。有关 CloudTrail Lake 的更多信息，请参阅《AWS CloudTrail 用户指南》中的使用 AWS CloudTrail Lake。

CloudTrail Lake 事件数据存储和查询会产生费用。创建事件数据存储时，您可以选择要用于事件数据存储的定价选项。定价选项决定了摄取和存储事件的成本，以及事件数据存储的默认和最长保留期。有关 CloudTrail 定价的更多信息，请参阅 AWS CloudTrail 定价。

CloudTrail 中的 Systems Manager 数据事件

数据事件可提供对资源或在资源中所执行资源操作（例如，创建或打开控制通道）的相关信息。这些也称为数据层面操作。数据事件通常是高容量活动。默认情况下，CloudTrail 不记录数据事件。CloudTrail 事件历史记录不记录数据事件。

记录数据事件将收取额外费用。有关 CloudTrail 定价的更多信息，请参阅 AWS CloudTrail 定价。

您可以使用 CloudTrail 控制台、AWS CLI 或 CloudTrail API 操作来记录 Systems Manager 资源类型的数据事件。有关如何记录数据事件的更多信息，请参阅《AWS CloudTrail 用户指南》中的使用 AWS Management Console 记录数据事件和使用 AWS Command Line Interface 记录数据事件。

下表列出了您可以为其记录数据事件的 Systems Manager 资源类型。数据事件类型（控制台）列显示可从 CloudTrail 控制台上的数据事件类型列表中选择的值。resources.type 值列显示了您在使用 AWS CLI 或 CloudTrail API 配置高级事件选择器时需要指定的 `resources.type` 值。记录到 CloudTrail 的数据 API 列显示了针对该资源类型记录到 CloudTrail 的 API 调用。

数据事件类型（控制台）	resources.type 值	记录至 CloudTrail 的数据 API
Systems Manager (系统管理员)	`AWS::SSMMessages::ControlChannel`	• `CreateControlChannel` • `OpenControlChannel` 有关更多信息，请参阅《Service Authorization Reference》中的 Actions defined by Amazon Message Gateway Service。
Systems Manager 托管式节点	`AWS::SSM::ManagedNode`	• `RequestManagedInstanceRoleToken` – 当在由 Systems Manager 托管的节点上运行的 AWS Systems Manager Agent（SSM Agent）向 Systems Manager 凭证服务请求凭证时，会生成此事件。 有关 `RequestManagedInstanceRoleToken` 操作的更多信息，请参阅 使用硬件指纹验证混合激活的计算机

您可以将高级事件选择器配置为在 `eventName`、`readOnly` 和 `resources.ARN` 字段上进行筛选，从而仅记录那些对您很重要的事件。有关这些字段的更多信息，请参阅《AWS CloudTrail API 参考》中的 AdvancedFieldSelector。

CloudTrail 中的 Systems Manager 管理事件

管理事件提供对您的 AWS 账户 内资源所执行管理操作的相关信息。这些也称为控制面板操作。默认情况下，CloudTrail 会记录管理事件。

Systems Manager 将所有控制面板操作将作为管理事件记录到 CloudTrail。Systems Manager API 操作记录在 AWS Systems Manager API 参考中。例如，对 `CreateMaintenanceWindows`、`PutInventory`、`SendCommand` 和 `StartSession` 操作的调用将在 CloudTrail 日志文件中生成条目。有关设置 CloudTrail 来监控 Systems Manager API 调用的示例，请参阅 使用 Amazon EventBridge 监控会话活动（控制台）。

Systems Manager 事件示例

一个事件表示一个来自任何源的请求，包括有关所请求的 API 操作、操作的日期和时间、请求参数等方面的信息。CloudTrail 日志文件不是公用 API 调用的有序堆栈跟踪，因此事件不会按任何特定顺序显示。

示例：

- 管理事件示例
- 数据事件示例

管理事件示例

示例 1：**`DeleteDocument`**

以下示例显示了一个 CloudTrail 事件，用于演示对美国东部（俄亥俄州）区域（us-east-2）中名为 `example-Document` 的文档执行的 `DeleteDocument` 操作。

```
{
    "eventVersion": "1.04",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AKIAI44QH8DHBEXAMPLE:203.0.113.11",
        "arn": "arn:aws:sts::123456789012:assumed-role/example-role/203.0.113.11",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-03-06T20:19:16Z"
            },
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AKIAI44QH8DHBEXAMPLE",
                "arn": "arn:aws:iam::123456789012:role/example-role",
                "accountId": "123456789012",
```

```
                "userName": "example-role"
            }
        }
    },
    "eventTime": "2018-03-06T20:30:12Z",
    "eventSource": "ssm.amazonaws.com",
    "eventName": "DeleteDocument",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "203.0.113.11",
    "userAgent": "example-user-agent-string",
    "requestParameters": {
        "name": "example-Document"
    },
    "responseElements": null,
    "requestID": "86168559-75e9-11e4-8cf8-75d18EXAMPLE",
    "eventID": "832b82d5-d474-44e8-a51d-093ccEXAMPLE",
    "resources": [
        {
            "ARN": "arn:aws:ssm:us-east-2:123456789012:document/example-Document",
            "accountId": "123456789012"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012",
    "eventCategory": "Management"
}
```

示例 2：**StartConnection**

以下示例显示了一个在美国东部（俄亥俄州）区域（us-east-2）使用 Fleet Manager 启动 RDP 连接的用户的 CloudTrail 事件。底层 API 操作是 `StartConnection`。

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:sts::123456789012:assumed-role/exampleRole",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
```

```
                "principalId": "AKIAI44QH8DHBEXAMPLE",
                "arn": "arn:aws:sts::123456789012:assumed-role/exampleRole",
                "accountId": "123456789012",
                "userName": "exampleRole"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2021-12-13T14:57:05Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2021-12-13T16:50:41Z",
    "eventSource": "ssm-guiconnect.amazonaws.com",
    "eventName": "StartConnection",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "34.230.45.60",
    "userAgent": "example-user-agent-string",
    "requestParameters": {
        "AuthType": "Credentials",
        "Protocol": "RDP",
        "ConnectionType": "SessionManager",
        "InstanceId": "i-02573cafcfEXAMPLE"
    },
    "responseElements": {
        "ConnectionArn": "arn:aws:ssm-guiconnect:us-east-2:123456789012:connection/
fcb810cd-241f-4aae-9ee4-02d59EXAMPLE",
        "ConnectionKey": "71f9629f-0f9a-4b35-92f2-2d253EXAMPLE",
        "ClientToken": "49af0f92-d637-4d47-9c54-ea51aEXAMPLE",
        "requestId": "d466710f-2adf-4e87-9464-055b2EXAMPLE"
    },
    "requestID": "d466710f-2adf-4e87-9464-055b2EXAMPLE",
    "eventID": "fc514f57-ba19-4e8b-9079-c2913EXAMPLE",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management"
}
```


数据事件示例

示例 1：**CreateControlChannel**

以下示例显示了一个 CloudTrail 事件，该事件演示了 CreateControlChannel 操作。


```
{
  "eventVersion":"1.08",
  "userIdentity":{
    "type":"AssumedRole",
    "principalId":"AKIAI44QH8DHBEXAMPLE",
    "arn":"arn:aws:sts::123456789012:assumed-role/exampleRole",
    "accountId":"123456789012",
    "accessKeyId":"AKIAI44QH8DHBEXAMPLE",
    "sessionContext":{
      "sessionIssuer":{
        "type":"Role",
        "principalId":"AKIAI44QH8DHBEXAMPLE",
        "arn":"arn:aws:iam::123456789012:role/exampleRole",
        "accountId":"123456789012",
        "userName":"exampleRole"
      },
      "attributes":{
        "creationDate":"2023-05-04T23:14:50Z",
        "mfaAuthenticated":"false"
      }
    }
  },
  "eventTime":"2023-05-04T23:53:55Z",
  "eventSource":"ssm.amazonaws.com",
  "eventName":"CreateControlChannel",
  "awsRegion":"us-east-1",
  "sourceIPAddress":"192.0.2.0",
  "userAgent":"example-agent",
  "requestParameters":{
    "channelId":"44295c1f-49d2-48b6-b218-96823EXAMPLE",
    "messageSchemaVersion":"1.0",
    "requestId":"54993150-0e8f-4142-aa54-3438EXAMPLE",
    "userAgent":"example-agent"
  },
  "responseElements":{
    "messageSchemaVersion":"1.0",
    "tokenValue":"Value hidden due to security reasons.",
    "url":"example-url"
  },
  "requestID":"54993150-0e8f-4142-aa54-3438EXAMPLE",
  "eventID":"a48a28de-7996-4ca1-a3a0-a51fEXAMPLE",
  "readOnly":false,
```

```
  "resources":[
    {
      "accountId":"123456789012",
      "type":"AWS::SSMMessages::ControlChannel",
      "ARN":"arn:aws:ssmmessages:us-east-1:123456789012:control-
channel/44295c1f-49d2-48b6-b218-96823EXAMPLE"
    }
  ],
  "eventType":"AwsApiCall",
  "managementEvent":false,
  "recipientAccountId":"123456789012",
  "eventCategory":"Data"
}
```

示例 2：**RequestManagedInstanceRoleToken**

以下示例显示了一个 CloudTrail 事件，该事件演示了 RequestManagedInstanceRoleToken 操作。

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "123456789012:aws:ec2-instance:i-02854e4bEXAMPLE",
        "arn": "arn:aws:sts::123456789012:assumed-role/aws:ec2-instance/
i-02854e4bEXAMPLE",
        "accountId": "123456789012",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "123456789012:aws:ec2-instance",
                "arn": "arn:aws:iam::123456789012:role/aws:ec2-instance",
                "accountId": "123456789012",
                "userName": "aws:ec2-instance"
            },
            "attributes": {
                "creationDate": "2023-08-27T03:34:46Z",
                "mfaAuthenticated": "false"
            },
            "ec2RoleDelivery": "2.0"
        }
    },
```

```
    "eventTime": "2023-08-27T03:37:15Z",
    "eventSource": "ssm.amazonaws.com",
    "eventName": "RequestManagedInstanceRoleToken",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "Apache-HttpClient/UNAVAILABLE (Java/1.8.0_362)",
    "requestParameters": {
        "fingerprint": "i-02854e4bf85EXAMPLE"
    },
    "responseElements": null,
    "requestID": "2582cced-455b-4189-9b82-7b48EXAMPLE",
    "eventID": "7f200508-e547-4c27-982d-4da0EXAMLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "123456789012",
            "type": "AWS::SSM::ManagedNode",
            "ARN": "arn:aws:ec2:us-east-1:123456789012:instance/i-02854e4bEXAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "123456789012",
    "eventCategory": "Data"
}
```


有关 CloudTrail 记录内容的信息，请参阅《AWS CloudTrail 用户指南》中的 CloudTrail 记录内容。

使用 CloudWatch Logs 记录自动化操作输出

Automation 是 AWS Systems Manager 中的一项工具，与 Amazon CloudWatch Logs 集成在一起。您可以将运行手册中 `aws:executeScript` 操作的输出发送到您指定的日志组。Systems Manager 不会为不使用 `aws:executeScript` 操作的文档创建日志组或任何日志流。如果文档使用 `aws:executeScript`，则发送到 CloudWatch Logs 的输出仅与这些操作有关。您可以将存储在 CloudWatch Logs 日志组中的 `aws:executeScript` 操作输出用于调试和故障排除目的。如果您选择已加密的日志组，则 `aws:executeScript` 操作输出也会被加密。记录 `aws:executeScript` 操作的输出是一项账户级别的设置。

要向 CloudWatch Logs for Amazon 拥有的运行手册发送操作输出，运行自动化的用户或角色必须拥有执行以下操作的权限：

- `logs:CreateLogGroup`

- `logs:CreateLogStream`
- `logs:DescribeLogGroups`
- `logs:DescribeLogStreams`
- `logs:PutLogEvents`

对于您拥有的运行手册，必须将相同权限添加到您用于运行运行手册的 IAM 服务角色（或 AssumeRole）。

将操作输出发送到 CloudWatch Logs（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，选择 自动化。
3. 选择首选项选项卡，然后选择编辑。
4. 选中 Send output to CloudWatch Logs (将输出发送到 CloudWatch Logs) 旁边的复选框。
5. （推荐）选中 Encrypt log data (对日志数据加密) 旁边的复选框。在打开此选项的情况下，将使用为日志组指定的服务器端加密密钥对日志数据加密。如果不需要加密发送到 CloudWatch Logs 的日志数据，请清除该复选框。如果日志组不允许加密，请清除该复选框。
6. 对于 CloudWatch Logs 日志组，要指定您要向其发送操作输出的 AWS 账户中的现有 CloudWatch Logs 日志组，请选择以下选项之一：
   - Send output to the default log group (将输出发送到默认日志组) – 如果默认日志组不存在 (`/aws/ssm/automation/executeScript`)，自动化将为您创建它。
   - Choose from a list of log groups (从日志组列表中选择) - 选择已在您的账户中创建的日志组来存储操作输出。
   - Enter a log group name (输入日志组名称) - 在文本框中输入已在您的账户中创建的用于存储操作输出的日志组的名称。
7. 选择 Save (保存)。

将操作输出发送到 CloudWatch Logs（命令行）

1. 打开首选命令行工具并运行以下命令，以更新操作输出目标。

Linux & macOS

```
aws ssm update-service-setting \
    --setting-id arn:aws:ssm:region:account-id:servicesetting/ssm/automation/
customer-script-log-destination \
    --setting-value CloudWatch
```

Windows

```
aws ssm update-service-setting ^
    --setting-id arn:aws:ssm:region:account-id:servicesetting/ssm/automation/
customer-script-log-destination ^
    --setting-value CloudWatch
```

PowerShell

```
Update-SSMServiceSetting `
    -SettingId "arn:aws:ssm:region:account-id:servicesetting/ssm/automation/
customer-script-log-destination" `
    -SettingValue "CloudWatch"
```

如果此命令成功，则无任何输出。

2. 运行以下命令，以指定要向其发送操作输出的日志组。

Linux & macOS

```
aws ssm update-service-setting \
    --setting-id arn:aws:ssm:region:account-id:servicesetting/ssm/automation/
customer-script-log-group-name \
    --setting-value my-log-group
```

Windows

```
aws ssm update-service-setting ^
    --setting-id arn:aws:ssm:region:account-id:servicesetting/ssm/automation/
customer-script-log-group-name ^
    --setting-value my-log-group
```

PowerShell

```
Update-SSMServiceSetting `
    -SettingId "arn:aws:ssm:region:account-id:servicesetting/ssm/automation/
customer-script-log-group-name" `
    -SettingValue "my-log-group"
```

如果此命令成功，则无任何输出。

3. 运行以下命令，以查看当前 AWS 账户 和 AWS 区域 中自动化操作日志记录首选项的当前服务设置。

Linux & macOS

```
aws ssm get-service-setting \
    --setting-id arn:aws:ssm:region:account-id:servicesetting/ssm/automation/
customer-script-log-destination
```

Windows

```
aws ssm get-service-setting ^
    --setting-id arn:aws:ssm:region:account-id:servicesetting/ssm/automation/
customer-script-log-destination
```

PowerShell

```
Get-SSMServiceSetting `
    -SettingId "arn:aws:ssm:region:account-id:servicesetting/ssm/automation/
customer-script-log-destination"
```

此命令会返回如下信息。

```
{
    "ServiceSetting": {
        "Status": "Customized",
        "LastModifiedDate": 1613758617.036,
        "SettingId": "/ssm/automation/customer-script-log-destination",
        "LastModifiedUser": "arn:aws:sts::123456789012:assumed-role/Administrator/
User_1",
```

```
        "SettingValue": "CloudWatch",
        "ARN": "arn:aws:ssm:us-east-2:123456789012:servicesetting/ssm/automation/
customer-script-log-destination"
    }
}
```

为 Run Command 配置 Amazon CloudWatch Logs

使用 Run Command（AWS Systems Manager 中的一项工具）发送命令时，可以指定要发送命令输出的位置。预设情况下，Systems Manager 仅返回命令输出的前 2.4 万个字符。如果您要查看命令输出的完整详细信息，可以指定 Amazon Simple Storage Service (Amazon S3) 存储桶。或者可以指定 Amazon CloudWatch Logs。如果指定 CloudWatch Logs，则 Run Command 会向 CloudWatch Logs 定期发送所有命令输出和错误日志。您可以近乎实时地监控输出日志，搜索特定短语、值或模式，以及基于搜索创建告警。

如果将托管节点配置为使用 AWS Identity and Access Management（IAM）托管策略 `AmazonSSMManagedInstanceCore` 和 `CloudWatchAgentServerPolicy`，则您的节点不需要进行额外配置即可将输出发送到 CloudWatch Logs。如果从控制台发送命令，请选择此选项；如果使用 AWS Command Line Interface (AWS CLI)、AWS Tools for Windows PowerShell 或 API 操作，则请添加 `cloud-watch-output-config` 部分和 `CloudWatchOutputEnabled` 参数。我们将在本主题的稍后内容中详细介绍 `cloud-watch-output-config` 部分和 `CloudWatchOutputEnabled` 参数。

有关将策略添加到 EC2 实例的实例配置文件的信息，请参阅配置 Systems Manager 所需的实例权限。有关将策略添加到打算用作托管式节点的本地服务器和虚拟机的服务角色的信息，请参阅在混合和多云环境中创建 Systems Manager 所需的 IAM 服务角色。

如果在多个节点上使用一个自定义策略，请在每个节点上更新该策略，以允许 Systems Manager 向 CloudWatch Logs 发送输出和日志。将以下策略对象添加到您的自定义策略。有关更新 IAM policy 的更多信息，请参阅《IAM 用户指南》中的编辑 IAM policy。

```
{
    "Effect": "Allow",
    "Action": "logs:DescribeLogGroups",
    "Resource": "*"
},
{
   "Effect":"Allow",
   "Action":[
```

```
      "logs:CreateLogGroup",
      "logs:CreateLogStream",
      "logs:DescribeLogStreams",
      "logs:PutLogEvents"
   ],
   "Resource":"arn:aws:logs:*:*:log-group:/aws/ssm/*"
},
```

在发送命令时指定 CloudWatch Logs

要在您从AWS Management Console发送命令时将 CloudWatch Logs 指定为输出，请在 Output options (输出选项) 部分选择 CloudWatch Output (CloudWatch 输出)。您可以选择在您要发送命令输出的位置指定 CloudWatch Logs 组的名称。如果您未指定组名称，则 Systems Manager 会自动为您创建一个日志组。该日志组使用以下命名格式：`/aws/ssm/`*`SystemsManagerDocumentName`*

如果使用 AWS CLI 运行命令，请在命令中指定 `cloud-watch-output-config` 部分。此部分使您能够指定 `CloudWatchOutputEnabled` 参数，并可选择指定 `CloudWatchLogGroupName` 参数。下面是一个例子。

Linux & macOS

```
aws ssm send-command \
    --instance-ids "instance ID" \
    --document-name "AWS-RunShellScript" \
    --parameters "commands=echo helloWorld" \
    --cloud-watch-output-config
 "CloudWatchOutputEnabled=true,CloudWatchLogGroupName=log group name"
```

Windows

```
aws ssm send-command ^
    --document-name "AWS-RunPowerShellScript" ^
    --parameters commands=["echo helloWorld"] ^
    --targets "Key=instanceids,Values=an instance ID” ^
    --cloud-watch-output-config '{"CloudWatchLogGroupName":"log group
 name","CloudWatchOutputEnabled":true}'
```

在 CloudWatch Logs 中查看命令输出

一旦命令开始运行，Systems Manager 便会近乎实时地向 CloudWatch Logs 发送输出。CloudWatch Logs 中的输出使用以下格式：

CommandID/*InstanceID*/*PluginID*/stdout

CommandID/*InstanceID*/*PluginID*/stderr

执行的输出每 30 秒或当缓冲区超过 200KB 时（以先发生者为准）上载一次。

> **Note**
>
> 日志流仅在输出数据可用时创建。例如，如果执行不存在错误数据，则不会创建 stderr 流。

此处是命令输出在 CloudWatch Logs 中显示时的示例。

```
Group - /aws/ssm/AWS-RunShellScript
Streams –
1234-567-8910/i-abcd-efg-hijk/AWS-RunPowerShellScript/stdout
24/1234-567-8910/i-abcd-efg-hijk/AWS-RunPowerShellScript/stderr
```

使用 Amazon EventBridge 监控 Systems Manager 事件

Amazon EventBridge 是一种无服务器事件总线服务，使您能够将应用程序与来自多种来源的数据连接起来。EventBridge 可以从您自己的应用程序、软件即服务（SaaS）应用程序和 AWS 服务传输实时数据流，然后将该数据路由到诸如 AWS Lambda 之类的目标。您可以设置路由规则来确定发送数据的目的地，以便构建能够实时响应所有数据源的应用程序架构。借助 EventBridge，您可以构建松耦合和分布式的事件驱动型架构。

EventBridge 以前称为 Amazon CloudWatch Events。EventBridge 包含新的功能，让您能够从 SaaS 合作伙伴和您自己的应用程序接收事件。现有 CloudWatch Events 用户可以在新的 EventBridge 控制台和 CloudWatch Events 控制台中访问其现有的默认总线、规则和事件。EventBridge 使用相同的 CloudWatch Events API，因此您现有的所有 CloudWatch Events API 使用方式保持不变。

EventBridge 可以将来自数十种 AWS 服务的事件添加到您的规则中，并可将 20 多种 AWS 服务添加为目标。

EventBridge 提供对 AWS Systems Manager 事件和 Systems Manager 目标的支持。

受支持的 Systems Manager 事件类型

EventBridge 可以检测到多种类型的 Systems Manager 事件，其中包括：

- 用于手动审批的即时节点访问请求状态更新。
- 即时节点访问请求失败。
- 维护时段结束。
- 自动化工作流成功完成。Automation 是 AWS Systems Manager 中的一项工具。
- 托管式节点不符合补丁合规性要求。
- 正在更新参数值。

EventBridge 支持来自以下 AWS Systems Manager 工具的事件：

- 即时节点访问（尽最大努力发出事件。）
- 自动化（尽最大努力发出事件。）
- Change Calendar（尽最大努力发出事件。）
- 合规性
- Inventory（尽最大努力发出事件。）
- Maintenance Windows（尽最大努力发出事件。）
- Parameter Store（尽最大努力发出事件。）
- Run Command（尽最大努力发出事件。）
- State Manager（尽最大努力发出事件。）

有关受支持的 Systems Manager 事件类型的完整详细信息，请参阅引用：Amazon EventBridge 事件模式和 Systems Manager 类型和适用于 Systems Manager 的 Amazon EventBridge 事件示例。

受支持的 Systems Manager 目标类型

EventBridge 支持以下三项 Systems Manager 工具作为事件规则的目标：

- 运行自动化工作流
- 运行 Run Command 命令文档（尽最大努力发出事件。）

- 创建 OpsCenter OpsItem

有关使用这些目标的建议方法，请参阅 示例场景：Amazon EventBridge 规则中的 Systems Manager 目标。

有关如何开始使用 EventBridge 并设置规则的更多信息，请参阅 Amazon EventBridge 用户指南中的 Amazon EventBridge 入门。有关使用 EventBridge 的完整信息，请参阅 Amazon EventBridge 用户指南。

主题

- 为 Systems Manager 事件配置 EventBridge
- 适用于 Systems Manager 的 Amazon EventBridge 事件示例
- 示例场景：Amazon EventBridge 规则中的 Systems Manager 目标

为 Systems Manager 事件配置 EventBridge

当支持的 AWS Systems Manager 状态变更或其他情况发生时，您可以使用 Amazon EventBridge 执行目标事件。您可以创建一个规则，只要状态发生转换或者在转换到一个或多个所关注的目标状态时，就运行此规则。

以下过程提供了创建 EventBridge 规则的一般步骤，当 Systems Manager 发出指定事件时，此规则便会生效。有关本用户指南中解决特定场景的过程列表，请参阅本主题末尾的更多信息。

> **Note**
>
> 当您的 AWS 账户 中的某个服务发出一个事件时，它始终会发送到您账户的默认事件总线。要编写一个规则来响应来自您账户中的 AWS 服务的事件，请将此规则与默认事件总线相关联。您可以在自定义事件总线上创建规则，以查找来自 AWS 服务的事件，但只有当您通过跨账户事件传输从其他账户收到此类事件时，此规则才会生效。有关更多信息，请参阅 Amazon EventBridge 用户指南中的在 AWS 账户 之间发送和接收 Amazon EventBridge 事件。

为 Systems Manager 事件配置 EventBridge

1. 打开位于 https://console.aws.amazon.com/events/ 的 Amazon EventBridge 控制台。
2. 在导航窗格中，选择规则。
3. 选择创建规则。

4. 为规则输入名称和描述。

   规则不能与同一 AWS 区域中和同一事件总线上的另一条规则的名称相同。

5. 对于事件总线，请选择要与此规则关联的事件总线。如果您希望此规则响应来自您自己的 AWS 账户的匹配事件，请选择 defaul（默认）。当您账户中的某个 AWS 服务发出一个事件时，它始终会发送到您账户的默认事件总线。
6. 对于规则类型，选择具有事件模式的规则。
7. 选择下一步。
8. 对于事件源，选择AWS 事件或 EventBridge 合作伙伴事件。
9. 在 Event pattern（事件模式）部分，选择 Event pattern form（事件模式表单）。
10. 对于事件源，选择AWS 服务。
11. 对于 service（AWS 服务），选择 Systems Manager。
12. 对于 Event type (事件类型)，执行以下操作之一：

    - 选择 All Events (所有事件)。

      如果您选择 All Events (所有事件)，则 Systems Manager 发出的所有事件都将匹配此规则。请注意，此选项可能会导致许多事件目标操作。
    - 选择要用于此规则的 Systems Manager 事件类型。EventBridge 支持来自以下 AWS Systems Manager 工具的事件：
      - 自动化
      - Change Calendar
      - 合规性
      - Inventory（清单）
      - Maintenance Windows
      - Parameter Store
      - Run Command
      - State Manager

      > **Note**
      >
      > 对于 EventBridge 不支持的 Systems Manager 操作，您可以通过 CloudTrail 选择 AWS API 调用，以创建基于 API 调用的事件规则，此类调用由 CloudTrail 记录。有关示例，请参阅 使用 Amazon EventBridge 监控会话活动（控制台）。

13. （可选）要使规则更具体，请添加筛选条件值。例如，假设 您选择了State Manager 并希望将规则限定为某个关联所针对单个托管式实例的状态，则对于 Specific type(s)（特定类型），请选择 EC2 State Manager Instance Association State Change（EC2 State Manager 实例关联状态更改）。

    有关受支持的详细类型的完整详细信息，请参阅 引用：Amazon EventBridge 事件模式和 Systems Manager 类型。

    某些详情类型还有其他受支持的选项，例如状态。可用选项取决于您选择的具体工具。

14. 选择下一步。
15. 对于目标类型，选择AWS 服务。
16. 对于 Select a target（选择一个目标），请选择一个目标，例如某个 Amazon SNS 主题或 AWS Lambda 函数。在收到与规则中定义的事件模式匹配的事件时将触发目标。
17. 对于许多目标类型，EventBridge 需要权限以便将事件发送到目标。在这些情况下，EventBridge 可以创建运行规则所需的 AWS Identity and Access Management (IAM) 角色：

    - 若要自动创建 IAM 角色，请选择 Create a new role for this specific resource (为此特定资源创建新角色)。
    - 要使用您之前创建的 IAM 角色，请选择 Use existing role (使用现有角色)。

18. （可选）选择 Add another target（添加其他目标），以为此规则添加其他目标。
19. 选择下一步。
20. （可选）为规则输入一个或多个标签。有关更多信息，请参阅《Amazon EventBridge 用户指南》中的 Amazon EventBridge 标签。
21. 选择下一步。
22. 查看规则详细信息并选择创建规则。

更多信息

- 创建使用运行手册（控制台）的 EventBridge 事件
- 使用输入变压器将数据传递到 Automation
- 使用 EventBridge 修复合规性问题
- 在 EventBridge 中查看清单删除操作
- 配置 EventBridge 规则以创建 OpsItems
- 为参数和参数策略配置 Eventbridge 规则

适用于 Systems Manager 的 Amazon EventBridge 事件示例

以下是 AWS Systems Manager 支持的 EventBridge 事件的 JSON 格式示例。

Systems Manager 事件类型

- AWS Systems Manager自动化事件
- AWS Systems Manager事件Change Calendar
- AWS Systems Manager事件Change Manager
- AWS Systems Manager 合规性事件
- AWS Systems Manager事件Maintenance Windows
- AWS Systems Manager事件Parameter Store
- AWS Systems Manager事件OpsCenter
- AWS Systems Manager事件Run Command
- AWS Systems Manager事件State Manager

AWS Systems Manager自动化事件

自动化步骤状态更改通知

```
{
  "version": "0",
  "id": "eeca120b-a321-433e-9635-dab369006a6b",
  "detail-type": "EC2 Automation Step Status-change Notification",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2024-11-29T19:43:35Z",
  "region": "us-east-1",
  "resources": ["arn:aws:ssm:us-east-2:123456789012:automation-
execution/333ba70b-2333-48db-b17e-a5e69c6f4d1c",
    "arn:aws:ssm:us-east-2:123456789012:automation-definition/runcommand1:1"],
  "detail": {
    "ExecutionId": "333ba70b-2333-48db-b17e-a5e69c6f4d1c",
    "Definition": "runcommand1",
    "DefinitionVersion": 1.0,
    "Status": "Success",
    "EndTime": "Nov 29, 2024 7:43:25 PM",
    "StartTime": "Nov 29, 2024 7:43:23 PM",
```

```
    "Time": 2630.0,
    "StepName": "runFixedCmds",
    "Action": "aws:runCommand"
  }
}
```

自动化执行状态更改通知

```
{
  "version": "0",
  "id": "d290ece9-1088-4383-9df6-cd5b4ac42b99",
  "detail-type": "EC2 Automation Execution Status-change Notification",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2024-11-29T19:43:35Z",
  "region": "us-east-2",
  "resources": ["arn:aws:ssm:us-east-2:123456789012:automation-
execution/333ba70b-2333-48db-b17e-a5e69c6f4d1c",
    "arn:aws:ssm:us-east-2:123456789012:automation-definition/runcommand1:1"],
  "detail": {
    "ExecutionId": "333ba70b-2333-48db-b17e-a5e69c6f4d1c",
    "Definition": "runcommand1",
    "DefinitionVersion": 1.0,
    "Status": "Success",
    "StartTime": "Nov 29, 2024 7:43:20 PM",
    "EndTime": "Nov 29, 2024 7:43:26 PM",
    "Time": 5753.0,
    "ExecutedBy": "arn:aws:iam::123456789012:user/userName"
  }
}
```

AWS Systems Manager事件Change Calendar

使用本主题中的信息来规划和了解 AWS Systems Manager Change Calendar 的 EventBridge 事件的行为。

> **Note**
>
> 目前不支持更改从其他 AWS 账户共享的日历的状态。

Change Calendar 与 Amazon EventBridge 集成

AWS Systems Manager Change Calendar 与 Amazon EventBridge 集成，用于通知您日历状态的变化。请注意以下与底层调度架构相关的行为：

事件计时和可靠性

- EventBridge 会尽力发送通知，调度容差最多 15 分钟。
- 状态变更事件反映整体日历状态转换，而不是单个日历事件。
- 当多个日历事件同时发生时，EventBridge 只会为每个实际日历状态变更生成一个事件。
- EventBridge 仅在日历的整体状态转换（例如，从“关闭”变为“打开”）时触发事件，而不会针对未导致状态变更的单个日历事件触发事件。
- 不会修改日历状态的咨询事件不会触发 EventBridge 通知。

事件修改和计时注意事项

- 如果在日历事件预定开始或结束时间的 15 分钟内修改日历事件，EventBridge 可能会生成重复通知或错过通知。
- 出现此行为的原因是调度系统可能没有足够的时间正确更新或取消之前安排的通知。
- 对于重复事件，此行为通常仅影响修改后的第一次事件。

相邻和重叠事件

- 当日历事件的安排间隔不超过 5 分钟时，状态转换事件可能会发生，也可能不会发生，具体取决于实际状态变更。
- 即使没有发生实际状态变更，按特定顺序创建重叠事件也可能会生成额外 EventBridge 事件。
- 为确保行为的可预测性，请避免在接近执行时间的位置创建或修改日历事件。

最佳实践

- 设计 EventBridge 规则和下游自动化，以处理潜在的重复事件。
- 在自动化工作流中实现幂等性，以防止重复通知导致的问题。
- 创建或修改日历事件时，请留出充足的准备时间（至少 15 分钟）。
- 使用特定的日历事件模式全面测试 EventBridge 集成。

日历开放

```
{
    "version": "0",
    "id": "47a3f03a-f30d-1011-ac9a-du3bdEXAMPLE",
    "detail-type": "Calendar State Change",
```

```
    "source": "aws.ssm",
    "account": "123456789012",
    "time": "2024-09-19T18:00:07Z",
    "region": "us-east-2",
    "resources": [
        "arn:aws:ssm:us-east-2:123456789012:document/MyCalendar"
    ],
    "detail": {
        "state": "OPEN",
        "atTime": "2024-09-19T18:00:07Z",
        "nextTransitionTime": "2024-10-11T18:00:07Z"
    }
}
```

日历关闭

```
{
    "version": "0",
    "id": "f30df03a-1011-ac9a-47a3-f761eEXAMPLE",
    "detail-type": "Calendar State Change",
    "source": "aws.ssm",
    "account": "123456789012",
    "time": "2024-09-17T21:40:02Z",
    "region": "us-east-2",
    "resources": [
        "arn:aws:ssm:us-east-2:123456789012:document/MyCalendar"
    ],
    "detail": {
        "state": "CLOSED",
        "atTime": "2024-08-17T21:40:00Z",
        "nextTransitionTime": "2024-09-19T18:00:07Z"
    }
}
```

AWS Systems Manager事件Change Manager

更改请求状态更新通知 – 示例 1

```
{
  "version": "0",
  "id": "feab80c1-a8ff-c721-b8b1-96ce70939696",
  "detail-type": "Change Request Status Update",
  "source": "aws.ssm",
```

```
  "account": "123456789012",
  "time": "2024-10-24T10:51:52Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ssm:us-west-2:123456789012:opsitem/oi-12345abcdef",
    "arn:aws:ssm:us-west-2:123456789012:document/MyRunbook1"
  ],
  "detail": {
    "change-request-id": "d0585556-80f6-4522-8dad-dada6d45b67d",
    "change-request-title": "A change request title",
    "ops-item-id": "oi-12345abcdef",
    "ops-item-created-by": "arn:aws:iam::123456789012:user/JohnDoe",
    "ops-item-created-time": "2024-10-24T10:50:33.180334Z",
    "ops-item-modified-by": "arn:aws:iam::123456789012:user/JohnDoe",
    "ops-item-modified-time": "2024-10-24T10:50:33.180340Z",
    "ops-item-status": "InProgress",
    "change-template-document-name": "MyChangeTemplate",
    "runbook-document-arn": "arn:aws:ssm:us-west-2:123456789012:document/MyRunbook1",
    "runbook-document-version": "1",
    "auto-approve": true,
    "approvers": [
      "arn:aws:iam::123456789012:user/JaneDoe"
    ]
  }
}
```

更改请求状态更新通知 – 示例 2

```
{
  "version": "0",
  "id": "25ce6b03-2e4e-1a2b-2a8f-6c9de8d278d2",
  "detail-type": "Change Request Status Update",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2024-10-24T10:51:52Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ssm:us-west-2:123456789012:opsitem/oi-abcdef12345",
    "arn:aws:ssm:us-west-2:123456789012:document/MyRunbook1"
  ],
  "detail": {
    "change-request-id": "d0585556-80f6-4522-8dad-dada6d45b67d",
    "change-request-title": "A change request title",
```

```
    "ops-item-id": "oi-abcdef12345",
    "ops-item-created-by": "arn:aws:iam::123456789012:user/JohnDoe",
    "ops-item-created-time": "2024-10-24T10:50:33.180334Z",
    "ops-item-modified-by": "arn:aws:iam::123456789012:user/JohnDoe",
    "ops-item-modified-time": "2024-10-24T10:50:33.997163Z",
    "ops-item-status": "Rejected",
    "change-template-document-name": "MyChangeTemplate",
    "runbook-document-arn": "arn:aws:ssm:us-west-2:123456789012:document/MyRunbook1",
    "runbook-document-version": "1",
    "auto-approve": true,
    "approvers": [
      "arn:aws:iam::123456789012:user/JaneDoe"
    ]
  }
}
```

AWS Systems Manager 合规性事件

以下是 AWS Systems Manager 合规性事件的示例。

关联合规

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "Configuration Compliance State Change",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2024-07-17T19:03:26Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:ssm:us-east-2:123456789012:managed-instance/i-01234567890abcdef"
  ],
  "detail": {
    "last-runtime": "2024-01-01T10:10:10Z",
    "compliance-status": "compliant",
    "resource-type": "managed-instance",
    "resource-id": "i-01234567890abcdef",
    "compliance-type": "Association"
  }
}
```

关联不合规

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "Configuration Compliance State Change",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2024-07-17T19:02:31Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:ssm:us-east-2:123456789012:managed-instance/i-01234567890abcdef"
  ],
  "detail": {
    "last-runtime": "2024-01-01T10:10:10Z",
    "compliance-status": "non_compliant",
    "resource-type": "managed-instance",
    "resource-id": "i-01234567890abcdef",
    "compliance-type": "Association"
  }
}
```

补丁合规

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "Configuration Compliance State Change",
  "source": "aws.123456789012",
  "account": "123456789012",
  "time": "2024-07-17T19:03:26Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:ssm:us-east-2:123456789012:managed-instance/i-01234567890abcdef"
  ],
  "detail": {
    "resource-type": "managed-instance",
    "resource-id": "i-01234567890abcdef",
    "compliance-status": "compliant",
    "compliance-type": "Patch",
    "patch-baseline-id": "PB789",
    "severity": "critical"
  }
}
```

补丁不合规

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "Configuration Compliance State Change",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2024-07-17T19:02:31Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:ssm:us-east-2:123456789012:managed-instance/i-01234567890abcdef"
  ],
  "detail": {
    "resource-type": "managed-instance",
    "resource-id": "i-01234567890abcdef",
    "compliance-status": "non_compliant",
    "compliance-type": "Patch",
    "patch-baseline-id": "PB789",
    "severity": "critical"
  }
}
```

AWS Systems Manager事件Maintenance Windows

以下是 Systems Manager Maintenance Windows 事件的示例。

注册目标

有效状态值包括 REGISTERED 和 DEREGISTERED。

```
{
   "version":"0",
   "id":"01234567-0123-0123-0123-0123456789ab",
   "detail-type":"Maintenance Window Target Registration Notification",
   "source":"aws.ssm",
   "account":"123456789012",
   "time":"2024-11-16T00:58:37Z",
   "region":"us-east-2",
   "resources":[
      "arn:aws:ssm:us-east-2:123456789012:maintenancewindow/mw-0ed7251d3fcf6e0c2",
      "arn:aws:ssm:us-east-2:123456789012:windowtarget/
e7265f13-3cc5-4f2f-97a9-7d3ca86c32a6"
```

```
   ],
   "detail":{
      "window-target-id":"e7265f13-3cc5-4f2f-97a9-7d3ca86c32a6",
      "window-id":"mw-0ed7251d3fcf6e0c2",
      "status":"REGISTERED"
   }
}
```

时段执行类型

有效状态值如下所示：

- CANCELLED
- CANCELLING
- FAILED
- IN_PROGRESS
- PENDING
- SKIPPED_OVERLAPPING
- SUCCESS TIMED_OUT

```
{
   "version":"0",
   "id":"01234567-0123-0123-0123-0123456789ab",
   "detail-type":"Maintenance Window Execution State-change Notification",
   "source":"aws.ssm",
   "account":"123456789012",
   "time":"2025-06-02T14:52:18Z",
   "region":"us-east-2",
   "resources":[
      "arn:aws:ssm:us-east-2:123456789012:maintenancewindow/mw-0c50858d01EXAMPLE"
   ],
   "detail":{
      "start-time":"2025-06-02T14:48:28.039273Z",
      "end-time":"2025-06-02T14:52:18.083773Z",
      "window-id":"mw-0ed7251d3fcf6e0c2",
      "window-execution-id":"14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE",
      "status":"SUCCESS"
   }
}
```

任务执行类型

有效状态值包括 IN_PROGRESS、SUCCESS、FAILED 和 TIMED_OUT。

```
{
   "version":"0",
   "id":"01234567-0123-0123-0123-0123456789ab",
   "detail-type":"Maintenance Window Task Execution State-change Notification",
   "source":"aws.ssm",
   "account":"123456789012",
   "time":"2025-06-02T14:52:18Z",
   "region":"us-east-2",
   "resources":[
      "arn:aws:ssm:us-east-2:123456789012:maintenancewindow/mw-0c50858d01EXAMPLE"
   ],
   "detail":{
      "start-time":"2025-06-02T14:48:28.039273Z",
      "task-execution-id":"6417e808-7f35-4d1a-843f-123456789012",
      "end-time":"2025-06-02T14:52:18.083773Z",
      "window-id":"mw-0ed7251d3fcf6e0c2",
      "window-execution-id":"14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE",
      "status":"SUCCESS"
   }
}
```

已处理的任务目标

有效状态值包括 IN_PROGRESS、SUCCESS、FAILED 和 TIMED_OUT。

```
{
   "version":"0",
   "id":"01234567-0123-0123-0123-0123456789ab",
   "detail-type":"Maintenance Window Task Target Invocation State-change Notification",
   "source":"aws.ssm",
   "account":"123456789012",
   "time":"2025-06-02T14:52:18Z",
   "region":"us-east-2",
   "resources":[
      "arn:aws:ssm:us-east-2:123456789012:maintenancewindow/mw-123456789012345678"
   ],
   "detail":{
      "start-time":"2025-06-02T14:48:28.039273Z",
      "end-time":"2025-06-02T14:52:18.083773Z",
```

```
      "window-id":"mw-0ed7251d3fcf6e0c2",
      "window-execution-id":"791b72e0-f0da-4021-8b35-f95dfEXAMPLE",
      "task-execution-id":"c9b05aba-197f-4d8d-be34-e73fbEXAMPLE",
      "window-target-id":"e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE",
      "status":"SUCCESS",
      "owner-information":"Owner"
   }
}
```

时段状态更改

有效状态值包括 ENABLED 和 DISABLED。

```
{
   "version":"0",
   "id":"01234567-0123-0123-0123-0123456789ab",
   "detail-type":"Maintenance Window State-change Notification",
   "source":"aws.ssm",
   "account":"123456789012",
   "time":"2024-11-16T00:58:37Z",
   "region":"us-east-2",
   "resources":[
      "arn:aws:ssm:us-east-2:123456789012:maintenancewindow/mw-0c50858d01EXAMPLE"
   ],
   "detail":{
      "window-id":"mw-0c50858d01EXAMPLE",
      "status":"DISABLED"
   }
}
```

AWS Systems Manager事件Parameter Store

以下是 Systems Manager Parameter Store 事件的示例。

创建参数

```
{
  "version": "0",
  "id": "6a7e4feb-b491-4cf7-a9f1-bf3703497718",
  "detail-type": "Parameter Store Change",
  "source": "aws.ssm",
  "account": "123456789012",
```

```
  "time": "2024-05-22T16:43:48Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:ssm:us-east-2:123456789012:parameter/MyExampleParameter"
  ],
  "detail": {
    "operation": "Create",
    "name": "MyExampleParameter",
    "type": "String",
    "description": "Sample Parameter"
  }
}
```

更新参数

```
{
  "version": "0",
  "id": "9547ef2d-3b7e-4057-b6cb-5fdf09ee7c8f",
  "detail-type": "Parameter Store Change",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2024-05-22T16:44:48Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:ssm:us-east-2:123456789012:parameter/MyExampleParameter"
  ],
  "detail": {
    "operation": "Update",
    "name": "MyExampleParameter",
    "type": "String",
    "description": "Sample Parameter"
  }
}
```

删除参数

```
{
  "version": "0",
  "id": "80e9b391-6a9b-413c-839a-453b528053af",
  "detail-type": "Parameter Store Change",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2024-05-22T16:45:48Z",
```

```
  "region": "us-east-2",
  "resources": [
    "arn:aws:ssm:us-east-2:123456789012:parameter/MyExampleParameter"
  ],
  "detail": {
    "operation": "Delete",
    "name": "MyExampleParameter",
    "type": "String",
    "description": "Sample Parameter"
  }
}
```

AWS Systems Manager事件OpsCenter

OpsCenter OpsItem 创建通知

```
{
  "version": "0",
  "id": "aae66adc-7aac-f0c0-7854-7691e8c079b8",
  "detail-type": "OpsItem Create",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2024-10-19T02:48:11Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ssm:us-west-2:123456789012:opsitem/oi-123456abcdef"
  ],
  "detail": {
    "created-by": "arn:aws:iam::123456789012:user/JohnDoe",
    "created-time": "2024-10-19T02:46:53.629361Z",
    "source": "aws.ssm",
    "status": "Open",
    "ops-item-id": "oi-123456abcdef",
    "title": "An issue title",
    "ops-item-type": "/aws/issue",
    "description": "A long description may appear here"
  }
}
```

OpsCenter OpsItem 更新通知

```
{
  "version": "0",
```

```
  "id": "2fb5b168-b725-41dd-a890-29311200089c",
  "detail-type": "OpsItem Update",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2024-10-19T02:48:11Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ssm:us-west-2:123456789012:opsitem/oi-123456abcdef"
  ],
  "detail": {
    "created-by": "arn:aws:iam::123456789012:user/JohnDoe",
    "created-time": "2024-10-19T02:46:54.049271Z",
    "modified-by": "arn:aws:iam::123456789012:user/JohnDoe",
    "modified-time": "2024-10-19T02:46:54.337354Z",
    "source": "aws.ssm",
    "status": "Open",
    "ops-item-id": "oi-123456abcdef",
    "title": "An issue title",
    "ops-item-type": "/aws/issue",
    "description": "A long description may appear here"
  }
}
```

AWS Systems Manager事件Run Command

Run Command 状态更改通知

```
{
    "version": "0",
    "id": "51c0891d-0e34-45b1-83d6-95db273d1602",
    "detail-type": "EC2 Command Status-change Notification",
    "source": "aws.ssm",
    "account": "123456789012",
    "time": "2024-07-10T21:51:32Z",
    "region": "us-east-2",
    "resources": ["arn:aws:ec2:us-east-2:123456789012:instance/i-02573cafcfEXAMPLE"],
    "detail": {
        "command-id": "e8d3c0e4-71f7-4491-898f-c9b35bee5f3b",
        "document-name": "AWS-RunPowerShellScript",
        "expire-after": "2024-07-14T22:01:30.049Z",
        "parameters": {
            "executionTimeout": ["3600"],
            "commands": ["date"]
```

```
        },
        "requested-date-time": "2024-07-10T21:51:30.049Z",
        "status": "Success"
    }
}
```

Run Command 调用状态更改通知

```
{
    "version": "0",
    "id": "4780e1b8-f56b-4de5-95f2-95dbEXAMPLE",
    "detail-type": "EC2 Command Invocation Status-change Notification",
    "source": "aws.ssm",
    "account": "123456789012",
    "time": "2024-07-10T21:51:32Z",
    "region": "us-east-2",
    "resources": ["arn:aws:ec2:us-east-2:123456789012:instance/i-02573cafcfEXAMPLE"],
    "detail": {
        "command-id": "e8d3c0e4-71f7-4491-898f-c9b35bee5f3b",
        "document-name": "AWS-RunPowerShellScript",
        "instance-id": "i-02573cafcfEXAMPLE",
        "requested-date-time": "2024-07-10T21:51:30.049Z",
        "status": "Success"
    }
}
```

AWS Systems Manager事件State Manager

State Manager 关联状态更改

```
{
   "version":"0",
   "id":"db839caf-6f6c-40af-9a48-25b2ae2b7774",
   "detail-type":"EC2 State Manager Association State Change",
   "source":"aws.ssm",
   "account":"123456789012",
   "time":"2024-05-16T23:01:10Z",
   "region":"us-east-2",
   "resources":[
      "arn:aws:ssm:us-east-2::document/AWS-RunPowerShellScript"
   ],
   "detail":{
      "association-id":"6e37940a-23ba-4ab0-9b96-5d0a1a05464f",
```

```
      "document-name":"AWS-RunPowerShellScript",
      "association-version":"1",
      "document-version":"Optional.empty",
      "targets":"[{\"key\":\"InstanceIds\",\"values\":[\"i-12345678\"]}]",
      "creation-date":"2024-02-13T17:22:54.458Z",
      "last-successful-execution-date":"2024-05-16T23:00:01Z",
      "last-execution-date":"2024-05-16T23:00:01Z",
      "last-updated-date":"2024-02-13T17:22:54.458Z",
      "status":"Success",
      "association-status-aggregated-count":"{\"Success\":1}",
      "schedule-expression":"cron(0 */30 * * * ? *)",
      "association-cwe-version":"1.0"
   }
}
```

State Manager 实例关联状态更改

```
{
   "version":"0",
   "id":"6a7e8feb-b491-4cf7-a9f1-bf3703467718",
   "detail-type":"EC2 State Manager Instance Association State Change",
   "source":"aws.ssm",
   "account":"123456789012",
   "time":"2024-02-23T15:23:48Z",
   "region":"us-east-2",
   "resources":[
      "arn:aws:ec2:us-east-2:123456789012:instance/i-12345678",
      "arn:aws:ssm:us-east-2:123456789012:document/my-custom-document"
   ],
   "detail":{
      "association-id":"34fcb7e0-9a14-4984-9989-0e04e3f60bd8",
      "instance-id":"i-02573cafcfEXAMPLE",
      "document-name":"my-custom-document",
      "document-version":"1",
      "targets":"[{\"key\":\"instanceids\",\"values\":[\"i-02573cafcfEXAMPLE\"]}]",
      "creation-date":"2024-02-23T15:23:48Z",
      "last-successful-execution-date":"2024-02-23T16:23:48Z",
      "last-execution-date":"2024-02-23T16:23:48Z",
      "status":"Success",
      "detailed-status":"",
      "error-code":"testErrorCode",
      "execution-summary":"testExecutionSummary",
      "output-url":"sampleurl",
```

```
      "instance-association-cwe-version":"1"
   }
}
```

示例场景：Amazon EventBridge 规则中的 Systems Manager 目标

在 Amazon EventBridge 规则中指定要调用的目标时，您可以从 20 多种目标类型中进行选择，并向每个规则添加最多 5 个目标。

在各种目标中，您可以选择 Automation、OpsCenter 和 Run Command（都是 AWS Systems Manager 中的工具）作为 EventBridge 事件发生时的目标操作。

以下是将这些工具用作 EventBridge 规则目标的几个示例。

自动化示例

您可以配置 EventBridge 规则，以便在发生以下事件时启动自动化工作流：

- 当 Amazon CloudWatch 告警报告托管式节点的状态检查失败时 (`StatusCheckFailed_Instance=1`)，请运行节点上的 `AWSSupport-ExecuteEC2Rescue` 自动化运行手册。
- 当 `EC2 Instance State-change Notification`事件发生时，因为新的 Amazon Elastic Compute Cloud (Amazon EC2) 实例正在运行，请运行实例上的 `AWS-AttachEBSVolume` 自动化运行手册。
- 当 Amazon Elastic Block Store (Amazon EBS) 卷创建并可用时，请运行卷上的 `AWS-CreateSnapshot`自动化运行手册。

OpsCenter 示例

您可以配置一个 EventBridge 规则，以便在发生以下事件时创建新的 OpsItem：

- 发生 Amazon DynamoDB 节流事件，或者 Amazon EBS 卷性能下降。
- Amazon EC2 Auto Scaling 组未能启动节点，或者 Systems Manager 自动化工作流失败。
- EC2 实例的状态从`Running`变为`Stopped`。

Run Command 示例

您可以配置 EventBridge 规则，以便在发生以下事件时在 Run Command 中运行 Systems Manager 命令文档：

- 当 Auto Scaling 组即将结束时，Run Command 脚本可以在节点结束之前捕获节点中的日志文件。
- 当在 Auto Scaling 组中创建新节点时，Run Command 目标操作可以打开 Web 服务器角色或者在节点上安装软件。
- 当发现托管式节点不合规时，Run Command 目标操作可以运行 `AWS-RunPatchBaseline` 文档，以更新节点上的补丁。

使用 Amazon SNS 通知监控 Systems Manager 状态更改

您可以配置 Amazon Simple Notification Service（Amazon SNS），令其发送与使用 Run Command 或 Maintenance Windows（都是 AWS Systems Manager 中的工具）发送的命令的状态有关的通知。Amazon SNS 协调并管理向订阅 Amazon SNS 主题的客户端或端点发送和传输通知。您可以在命令更改为新状态或特定状态（例如 Failed (已失败) 或 Timed Out (已超时)）时收到通知。如果您将一条命令发送给多个节点，则对于发送给特定节点的命令的每个副本，您都可以收到通知。每个副本称为一个调用。

Amazon SNS 能够以 HTTP 或 HTTPS POST 以及电子邮件（SMTP，纯文本或 JSON 格式）的形式传输通知，或将通知作为消息发布到 Amazon Simple Queue Service (Amazon SQS) 队列。有关更多信息，请参阅 Amazon Simple Notification Service Developer Guide 中的什么是 Amazon SNS。有关由 Run Command 和 Maintenance Windows 提供的 Amazon SNS 通知中包括的 JSON 数据的结构示例，请参阅 适用于 AWS Systems Manager 的 Amazon SNS 通知示例。

> **Important**
>
> 请注意以下重要信息。
>
> - 不支持 Amazon Simple Notification Service FIFO 主题。
> - 聊天应用程序中的 Amazon Q 开发者版不支持使用 Amazon SNS 监控 Systems Manager。若想使用聊天应用程序中的 Amazon Q 开发者版来监控 Systems Manager，则必须将其与 Amazon EventBridge 一起使用。有关使用 EventBridge 监控 Systems Manager 的信息，请参阅使用 Amazon EventBridge 监控 Systems Manager 事件。有关聊天应用程序中的 Amazon EventBridge 和 Amazon Q 开发者版的信息，请参阅《聊天应用程序中的 Amazon Q 开发者版管理员指南》中的教程：创建向聊天应用程序中的 Amazon Q 开发者版发送通知的 EventBridge 规则。

为 AWS Systems Manager 配置 Amazon SNS 通知

注册到维护时段的 Run Command 和 Maintenance Windows 任务可为进入以下状态的命令任务发送 Amazon SNS 通知：

- 正在进行
- 成功
- 已失败
- 已超时
- 已取消

有关导致命令进入以下状态之一的条件的信息，请参阅 了解命令状态。

> **Note**
>
> 使用 Run Command 发送的命令还会报告 Canceling (正在取消) 和 Pending (待处理) 状态。Amazon SNS 通知不会捕获这些状态。

命令摘要 Amazon SNS 通知

如果您在维护时段中为 Amazon SNS 通知配置 Run Command 或 Run Command 任务，Amazon SNS 将发送摘要消息，其中包含以下信息。

字段	类型	描述
eventTime	字符串	启动事件的时间。由于 Amazon SNS 不保证消息传输顺序，因此时间戳很重要。示例：2016-04-26T13:15:30Z
documentName	字符串	用于运行此命令的 SSM 文档的名称。
commandId	字符串	在发送命令后由 Run Command 生成的 ID。

字段	类型	描述
expiresAfter	日期	如果达到此时间但命令尚未开始执行，则它将不会运行。
outputS3BucketName	字符串	应该存储对命令执行的响应的 Amazon Simple Storage Service (Amazon S3) 存储桶。
outputS3KeyPrefix	字符串	应该存储对命令执行的响应的存储桶中的 Amazon S3 目录路径。
requestedDateTime	字符串	将请求发送到此特定节点的日期和时间。
instanceIds	StringList	通过命令设为目标的节点。 **Note** 如果 Run Command 任务直接将实例 ID 设为目标，则实例 ID 仅包含在摘要消息中。如果使用基于标签的目标设定发出 Run Command 任务，则实例 ID 不会包含在摘要消息中。
状态	字符串	命令的命令状态。

基于调用的 Amazon SNS 通知

如果您将一条命令发送给多个节点，则 Amazon SNS 可以发送有关该命令的每个副本或调用的消息。此类消息包含以下信息。

字段	类型	描述
eventTime	字符串	启动事件的时间。由于 Amazon SNS 不保证消息传输顺序，因此时间戳很重要。示例：2016-04-26T13:15:30Z
documentName	字符串	用于运行此命令的 Systems Manager 文档（SSM 文档）的名称。
requestedDateTime	字符串	将请求发送到此特定节点的日期和时间。
commandId	字符串	在发送命令后由 Run Command 生成的 ID。
instanceId	字符串	由命令设为目标的实例。
状态	字符串	此调用的命令状态。

要设置命令更改状态时的 Amazon SNS 通知，必须完成以下任务。

> **Note**
>
> 如果您没有为维护时段配置 Amazon SNS 通知，则可跳过本主题后面的“任务 5”。

主题

- 任务 1：创建并订阅 Amazon SNS 主题
- 任务 2：为 Amazon SNS 通知创建 IAM policy
- 任务 3：为 Amazon SNS 通知创建 IAM 角色
- 任务 4：配置用户访问权限
- 任务 5：将 iam:PassRole 策略附加到您的维护时段角色

任务 1：创建并订阅 Amazon SNS 主题

Amazon SNS 主题是一个通信渠道，注册到维护时段的 Run Command 和 Run Command 任务使用该渠道发送与命令的状态有关的通知。Amazon SNS 支持多种通信协议，包括 HTTP/S、电子邮件和其他 AWS 服务，如 Amazon Simple Queue Service（Amazon SQS）。要入门，我们建议您先从电子邮件协议开始。有关如何创建主题的信息，请参阅 Amazon Simple Notification Service 开发人员指南中的创建 Amazon SNS 主题。

> **Note**
>
> 创建主题后，复制或记下 Topic ARN (主题 ARN)。在发送配置为返回状态通知的命令时将指定此 ARN。

创建主题后，通过指定终端节点来订阅该主题。如果您选择了电子邮件协议，则终端节点即为您希望接收通知的电子邮件地址。有关如何订阅主题的更多信息，请参阅 Amazon Simple Notification Service 开发人员指南中的订阅 Amazon SNS 主题。

Amazon SNS 将从 AWS Notifications 向您指定的电子邮件地址发送确认电子邮件。打开这封电子邮件，然后选择 Confirm subscription (确认订阅) 链接。

您将收到来自 AWS 的确认消息。Amazon SNS 现已配置为接收通知并以电子邮件的形式将通知发送到指定的电子邮件地址。

任务 2：为 Amazon SNS 通知创建 IAM policy

使用以下过程创建自定义 AWS Identity and Access Management (IAM) policy，该策略提供了启动 Amazon SNS 通知的权限。

为 Amazon SNS 通知创建自定义 IAM policy

1. 访问：https://console.aws.amazon.com/iam/，打开 IAM 控制台。
2. 在导航窗格中选择 Policies，然后选择 Create Policy。（如果显示 Get Started (入门) 按钮，请选择此按钮，然后选择 Create Policy (创建策略)。）
3. 选择 JSON 选项卡。
4. 根据 Amazon SNS 主题是否使用 AWS KMS 加密，将默认内容替换为以下内容之一：

SNS topic not encrypted

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sns:Publish"
            ],
            "Resource": "arn:aws:sns:us-east-1:111122223333:sns-topic-
name"
        }
    ]
}
```

region 表示 AWS Systems Manager 支持的 AWS 区域 的标识符，例如 us-east-2 对应美国东部（俄亥俄）区域。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

account-id 表示您的 AWS 账户的 12 位标识符，格式为 123456789012。

sns-topic-name 表示要用于发布通知的 Amazon SNS 主题的名称。

SNS topic encrypted

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sns:Publish"
            ],
            "Resource": "arn:aws:sns:us-east-1:111122223333:sns-topic-
name"
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/kms-key-
id"
        }
    ]
}
```

region 表示 AWS Systems Manager 支持的 AWS 区域 的标识符，例如 us-east-2 对应美国东部（俄亥俄）区域。有关支持的 *region* 值的列表，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints 的 Region 列。

account-id 表示您的 AWS 账户的 12 位标识符，格式为 123456789012。

sns-topic-name 表示要用于发布通知的 Amazon SNS 主题的名称。

kms-key-id 表示 AWS KMS 中用于加密和解密主题的对称加密 KMS 密钥的 ID，格式为 1234abcd-12ab-34cd-56ef-12345EXAMPLE。

> **Note**
>
> 使用 AWS KMS 加密需要支付相应的费用。有关更多信息，请参阅《AWS Key Management Service Developer Guide》中的 Managing Amazon SNS encryption keys and costs。

5. 选择下一步：标签。
6. （可选）添加一个或多个标签键值对，以组织、跟踪或控制此策略的访问权限。
7. 选择下一步：审核。
8. 在 Review policy (审核策略) 页面上，对于 Name (名称)，输入内联策略的名称。例如：**my-sns-publish-permissions**。
9. （可选）对于 Description (描述)，输入策略的描述。
10. 选择 Create policy (创建策略)。

任务 3：为 Amazon SNS 通知创建 IAM 角色

使用以下过程为 Amazon SNS 通知创建 IAM 角色。Systems Manager 使用此服务角色启动 Amazon SNS 通知。在后续的所有过程中，此角色都称为 Amazon SNS IAM 角色。

为 Amazon SNS 通知创建 IAM 服务角色

1. 登录 AWS Management Console，然后通过以下网址打开 IAM 控制台：https://console.aws.amazon.com/iam/。
2. 在 IAM 控制台的导航窗格中，选择角色，然后选择创建角色。
3. 选择 AWS 服务 角色类型，然后选择 Systems Manager。
4. 选择 Systems Manager 应用场景。然后选择下一步。
5. 在 Attach permissions policies (附加权限策略) 页面上，选中您在“任务 2”中创建的自定义策略名称左侧的方框。例如：**`my-sns-publish-permissions`**。
6. （可选）设置权限边界。这是一项高级特征，可用于服务角色，但不可用于服务相关角色。

   展开 Permissions boundary（权限边界）部分，然后选择 Use a permissions boundary to control the maximum role permissions（使用权限边界控制最大角色权限）。IAM 包括您的账户中的 AWS 托管策略和客户托管策略的列表。选择要用于权限边界的策略，或选择创建策略以打开新的浏览器选项卡并从头开始创建新策略。有关更多信息，请参阅 IAM 用户指南 中的创建 IAM policy。在您创建策略后，关闭该选项卡并返回到您的原始选项卡，以选择要用于权限边界的策略。
7. 选择 Next（下一步）。
8. 如果可能，输入有助于识别该角色的作用的角色名称或角色名称后缀。角色名称在您的 AWS 账户内必须是唯一的。名称不区分大小写。例如，您无法同时创建名为 **`PRODROLE`** 和 **`prodrole`** 的角色。由于多个单位可能引用该角色，角色创建完毕后无法编辑角色名称。
9. （可选）对于 Description（描述），输入新角色的描述。
10. 在 Step 1: Select trusted entities（步骤 1：选择可信实体）或 Step 2: Select permissions（步骤 2：选择权限）部分中的 Edit（编辑），以编辑角色的用户案例和权限。
11. （可选）通过以键值对的形式附加标签来向用户添加元数据。有关在 IAM 中使用标签的更多信息，请参阅 《IAM 用户指南》 中的标记 IAM 资源。
12. 检查角色，然后选择 Create role。
13. 选择此角色的名称，然后复制或记下 Role ARN（角色 ARN）的值。在发送配置为返回 Amazon SNS 通知的命令时，将使用此角色的 Amazon Resource Name（ARN）。

14. 使 Summary (摘要) 页面保持打开状态。

任务 4：配置用户访问权限

如果为 IAM 实体（用户、角色或组）分配了管理员权限，则该用户或角色将有权访问 Run Command 和 Maintenance Windows（都是 AWS Systems Manager 中的工具）。

对于没有管理员权限的实体，管理员必须向 IAM 实体授予以下权限：

- `AmazonSSMFullAccess` 托管策略，或提供类似权限的策略。
- 已在 任务 3：为 Amazon SNS 通知创建 IAM 角色 中为角色创建的 `iam:PassRole` 权限。例如：

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::111122223333:role/sns-role-name",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ssm.amazonaws.com"
                }
            }
        }
    ]
}
```

要提供访问权限，请为您的用户、组或角色添加权限：

- AWS IAM Identity Center 中的用户和群组：

  创建权限集合。按照《AWS IAM Identity Center 用户指南》中创建权限集的说明进行操作。
- 通过身份提供商在 IAM 中托管的用户：

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

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

配置用户访问权限并将 **iam:PassRole** 策略附加到用户账户

1. 在 IAM 导航窗格中，选择 Users (用户)，然后选择您要配置的用户账户。
2. 在 Permissions (权限) 选项卡上的策略列表中，验证并确保其中列出了 **AmazonSSMFullAccess** 策略，或者存在授予账户访问 Systems Manager 权限的类似策略。
3. 选择 Add inline policy (添加内联策略)。
4. 在 Create policy (创建策略) 页面上，选择 Visual editor (可视化编辑器) 选项卡。
5. 选择 Choose a service (选择服务)，然后选择 IAM。
6. 对于 Actions（操作），在 Filter actions（筛选操作）文本框中，输入 **PassRole**，然后选中 PassRole 旁的复选框。
7. 对于 Resources（资源），验证已选择 Specific（特定），然后选择 Add ARN（添加 ARN）。
8. 在 Specify ARN for role（为角色指定 ARN）字段中，粘贴在“任务 3”结束时复制的 Amazon SNS IAM 角色 ARN。系统会自动填充 Account (账户) 和 Role name with path (具有路径的角色名称) 字段。
9. 选择 Add (添加)。
10. 选择查看策略。
11. 在 Review Policy（检查策略）页面上输入一个名称，然后选择 Create Policy（创建策略）。

任务 5：将 iam:PassRole 策略附加到您的维护时段角色

当您向某一维护时段注册 Run Command 任务时，将指定服务角色 Amazon Resource Name (ARN)。Systems Manager 将使用此服务角色来运行注册到该维护时段的任务。要为已注册的 Run Command 任务配置 Amazon SNS 通知，请将 `iam:PassRole` 策略附加到指定的维护时段服务角色。如果您不打算为 Amazon SNS 通知配置已注册的任务，则可以跳过此任务。

`iam:PassRole` 策略允许 Maintenance Windows 服务角色将在“任务 3”中创建的 Amazon SNS IAM 角色传递到 Amazon SNS 服务。以下过程显示了如何将 `iam:PassRole` 策略附加到 Maintenance Windows 服务角色。

> **Note**
>
> 使用您的维护时段的自定义服务角色发送与已注册的 Run Command 任务有关的通知。有关信息，请参阅设置 Maintenance Windows。
> 如果您需要为维护时段任务创建自定义服务角色，请参阅设置 Maintenance Windows。

将 **`iam:PassRole`** 策略附加到 Maintenance Windows 角色

1. 通过 https://console.aws.amazon.com/iam/ 打开 IAM 控制台。
2. 在导航窗格中，选择 Roles (角色)，然后选择在“任务 3”中创建的 Amazon SNS IAM 角色。
3. 复制或记下 Role ARN (角色 ARN)，然后返回到 IAM 控制台的 Roles (角色) 部分。
4. 从 Role name（角色名称）列表中选择您创建的自定义 Maintenance Windows 服务角色。
5. 在 Permissions（权限）选项卡中，验证是否列出了 `AmazonSSMMaintenanceWindowRole` 策略，或者存在可向 Systems Manager API 授予维护时段权限的类似策略。如果没有，则依次选择添加权限、附加策略，以附加此策略。
6. 选择 Add permissions, Create inline policy（添加权限、创建内联策略）。
7. 选择 Visual editor (可视化编辑器) 选项卡。
8. 对于 Service (服务)，请选择 IAM。
9. 对于 Actions（操作），在 Filter actions（筛选操作）文本框中，输入 **`PassRole`**，然后选中 PassRole 旁的复选框。
10. 对于 Resources (资源)，请选择 Specific (特定)，然后选择 Add ARN (添加 ARN)。
11. 在 Specify ARN for role (为角色指定 ARN) 框中，粘贴在“任务 3”中创建的 Amazon SNS IAM 角色的 ARN，然后选择 Add (添加)。
12. 选择查看策略。
13. 在审核策略页面上，为 `PassRole` 策略指定名称，然后选择创建策略。

适用于 AWS Systems Manager 的 Amazon SNS 通知示例

您可以配置 Amazon Simple Notification Service（Amazon SNS），令其发送与使用 Run Command 或 Maintenance Windows（都是 AWS Systems Manager 中的工具）发送的命令的状态有关的通知。

> **Note**
>
> 本指南不解决如何为 Run Command 或 Maintenance Windows 配置通知的问题。有关将 Run Command 或 Maintenance Windows 配置为发送与命令的状态有关的 Amazon SNS 通知的信息，请参阅 为 AWS Systems Manager 配置 Amazon SNS 通知。

以下示例显示了为 Run Command 或 Maintenance Windows 配置后，Amazon SNS 通知返回的 JSON 输出的结构。

使用实例 ID 目标设定的命令摘要消息的 JSON 输出示例

```
{
    "commandId": "a8c7e76f-15f1-4c33-9052-0123456789ab",
    "documentName": "AWS-RunPowerShellScript",
    "instanceIds": [
        "i-1234567890abcdef0",
        "i-9876543210abcdef0"
    ],
    "requestedDateTime": "2019-04-25T17:57:09.17Z",
    "expiresAfter": "2019-04-25T19:07:09.17Z",
    "outputS3BucketName": "amzn-s3-demo-bucket",
    "outputS3KeyPrefix": "runcommand",
    "status": "InProgress",
    "eventTime": "2019-04-25T17:57:09.236Z"
}
```

使用基于标签的目标设定的命令摘要消息的 JSON 输出示例

```
{
    "commandId": "9e92c686-ddc7-4827-b040-0123456789ab",
    "documentName": "AWS-RunPowerShellScript",
    "instanceIds": [],
    "requestedDateTime": "2019-04-25T18:01:03.888Z",
    "expiresAfter": "2019-04-25T19:11:03.888Z",
    "outputS3BucketName": "",
    "outputS3KeyPrefix": "",
    "status": "InProgress",
    "eventTime": "2019-04-25T18:01:05.825Z"
}
```

调用消息的 JSON 输出示例

```
{
    "commandId": "ceb96b84-16aa-4540-91e3-925a9a278b8c",
    "documentName": "AWS-RunPowerShellScript",
    "instanceId": "i-1234567890abcdef0",
    "requestedDateTime": "2019-04-25T18:06:05.032Z",
    "status": "InProgress",
    "eventTime": "2019-04-25T18:06:05.099Z"
}
```

使用 Run Command 发送返回状态通知的命令

以下过程显示了如何使用 AWS Command Line Interface（AWS CLI）或 AWS Systems Manager 控制台通过配置为返回状态通知的 Run Command（AWS Systems Manager 中的一项工具）来发送命令。

发送返回通知的 Run Command（控制台）

可以使用以下过程，通过配置为使用 Systems Manager 控制台返回状态通知的 Run Command 来发送命令。

发送返回通知的命令（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Run Command。
3. 选择 Run command（运行命令）。
4. 在命令文档列表中，请选择 Systems Manager 文档。
5. 在 Command parameters (命令参数) 部分中，为必需的参数指定值。
6. 在 Targets（目标）部分中，通过指定标签、手动选择实例或边缘设备或指定资源组，选择要在其上运行此操作的托管式节点。

   > **Tip**
   >
   > 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

7. 对于 Other parameters（其他参数）：

- 对于 Comment（注释），请输入有关此命令的信息。
- 对于 Timeout (seconds) (超时 (秒))，请指定在整个命令执行失败之前系统等待的秒数。

8. 对于 Rate control（速率控制）：

   - 对于 Concurrency（并发），请指定要同时运行该命令的托管式节点的数量或百分比。

   > **Note**
   >
   > 如果您通过指定应用于托管式节点的标签或指定 AWS Resource Groups 来选择目标，但不确定有多少个托管式节点已被设为目标，则可通过指定百分比来限制可同时运行该文档的目标的数量。

   - 对于 Error threshold（错误阈值），请指定当命令在一定数量或百分比的节点上失败后，何时在其他托管式节点上停止运行该命令。例如，如果您指定三个错误，Systems Manager 将在收到第四个错误时停止发送该命令。仍在处理该命令的托管式节点也可能发送错误。

9. （可选）对于 输出选项，要将命令输出保存到文件，请选中 将命令输出写入 S3 存储桶 框。在输入框中输入存储桶和前缀（文件夹）名称。

   > **Note**
   >
   > 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给实例的实例配置文件（适用于 EC2 实例）或 IAM 服务角色（混合激活的计算机）的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确保与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

10. 在 SNS Notifications (SNS 通知) 部分中，选择 Enable SNS notifications (启用 SNS 通知)。
11. 对于 IAM role（IAM 角色），选择您在使用 Amazon SNS 通知监控 Systems Manager 状态更改的“任务 3”中创建的 Amazon SNS IAM 角色 ARN。
12. 对于 SNS topic (SNS 主题)，请输入要使用的 Amazon SNS 主题 ARN。
13. 对于 Event notifications (事件通知)，请选择要针对其接收通知的事件。
14. 对于 Change notifications（更改通知），选择 Command status changes（命令状态更改）以仅接收命令摘要的通知，或者 Command status on each instance changes（每个实例的命令状态更改）以接收发送到多个节点的命令的每个副本。
15. 选择运行。

16. 检查您的电子邮件以查找来自 Amazon SNS 的邮件，然后打开该电子邮件。Amazon SNS 可能需要几分钟来发送该电子邮件。

发送返回通知的 Run Command (CLI)

可以使用以下过程，通过配置为使用 AWS CLI 返回状态通知的 Run Command 来发送命令。

发送返回通知的命令 (CLI)

1. 打开 AWS CLI。
2. 在以下命令中指定参数，以基于托管式节点 ID 设定目标。

```
aws ssm send-command --instance-ids "ID-1, ID-2" --document-name "Name"
 --parameters '{"commands":["input"]}' --service-role "SNSRoleARN" --
notification-config '{"NotificationArn":"SNSTopicName","NotificationEvents":
["All"],"NotificationType":"Command"}'
```

以下为示例。

```
aws ssm send-command --instance-ids "i-02573cafcfEXAMPLE, i-0471e04240EXAMPLE"
 --document-name "AWS-RunPowerShellScript" --parameters '{"commands":
["Get-Process"]}' --service-role "arn:aws:iam::111122223333:role/
SNS_Role" --notification-config '{"NotificationArn":"arn:aws:sns:us-
east-1:111122223333:SNSTopic","NotificationEvents":
["All"],"NotificationType":"Command"}'
```

替代命令

在以下命令中指定参数，以使用标签设定托管实例目标。

```
aws ssm send-command --targets "Key=tag:TagName,Values=TagKey" --document-name
 "Name" --parameters '{"commands":["input"]}' --service-role "SNSRoleARN" --
notification-config '{"NotificationArn":"SNSTopicName","NotificationEvents":
["All"],"NotificationType":"Command"}'
```

以下为示例。

```
aws ssm send-command --targets "Key=tag:Environment,Values=Dev" --
document-name "AWS-RunPowerShellScript" --parameters '{"commands":
["Get-Process"]}' --service-role "arn:aws:iam::111122223333:role/
```

```
SNS_Role" --notification-config '{"NotificationArn":"arn:aws:sns:us-
east-1:111122223333:SNSTopic","NotificationEvents":
["All"],"NotificationType":"Command"}'
```

3. 按 Enter 键。
4. 检查您的电子邮件以查找来自 Amazon SNS 的邮件，然后打开该电子邮件。Amazon SNS 可能需要几分钟来发送该电子邮件。

有关更多信息，请参阅《AWS CLI 命令参考》中的 send-command。

使用维护时段发送返回状态通知的命令

以下过程显示了如何使用 AWS Systems Manager 控制台或 AWS Command Line Interface（AWS CLI）将 Run Command 任务注册到维护时段。Run Command 是 AWS Systems Manager 中的一项工具。这些过程还介绍了如何将 Run Command 任务配置为返回状态通知。

开始前的准备工作

如果您尚未创建维护时段或注册目标，请参阅 使用控制台创建和管理维护时段，了解有关如何创建维护时段和注册目标的步骤。

要接收来自 Amazon Simple Notification Service (Amazon SNS) 服务的通知，请将 `iam:PassRole` 策略附加到在已注册的任务中指定的 Maintenance Windows 服务角色。如果您尚未向 Maintenance Windows 服务角色添加 `iam:PassRole` 权限，请参阅 任务 5：将 iam:PassRole 策略附加到您的维护时段角色。

将返回通知的 Run Command 任务注册到维护时段（控制台）

可以使用以下过程，借助 Systems Manager 控制台将配置为返回状态通知的 Run Command 任务注册到您的维护时段。

将返回通知的 Run Command 任务注册到维护时段（控制台）

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Maintenance Windows。
3. 选择您要为其注册配置为发送 Amazon Simple Notification Service (Amazon SNS) 通知的 Run Command 任务的维护时段。

4. 选择 Actions (操作)，然后选择 Register Run command task (注册运行命令任务)。
5. （可选）在 Name (名称) 字段中，输入任务的名称。
6. （可选）在 Description (描述) 字段中，输入描述。
7. 对于 Command document（命令文档），选择一个命令文档。
8. 对于 Task priority (任务优先级)，指定此任务的优先级。零 (0) 表示最高优先级。维护时段内的任务按优先级顺序计划。具有相同优先级的任务则并行计划。
9. 在 Targets (目标) 部分中，选择已注册的目标组或选择未注册的目标。
10. 对于 Rate control（速率控制）：
    - 对于 Concurrency（并发），请指定要同时运行该命令的托管式节点的数量或百分比。

      > **Note**
      >
      > 如果您通过指定应用于托管式节点的标签或指定 AWS Resource Groups 来选择目标，但不确定有多少个托管式节点已被设为目标，则可通过指定百分比来限制可同时运行该文档的目标的数量。

    - 对于 Error threshold（错误阈值），请指定当命令在一定数量或百分比的节点上失败后，何时在其他托管式节点上停止运行该命令。例如，如果您指定三个错误，Systems Manager 将在收到第四个错误时停止发送该命令。仍在处理该命令的托管式节点也可能发送错误。
11. 在 IAM service role (IAM 服务角色) 区域中，选择对 SNS 角色具有 `iam:PassRole` 权限的 Maintenance Windows 服务角色。

    > **Note**
    >
    > 将 `iam:PassRole` 权限添加到 Maintenance Windows 角色，以允许 Systems Manager 将 SNS 角色传递给 Amazon SNS。如果您尚未添加 `iam:PassRole` 权限，请参阅主题使用 Amazon SNS 通知监控 Systems Manager 状态更改中的“任务 5”。

12. （可选）对于 Output options (输出选项)，要将命令输出保存到文件，请选中 Enable writing output to S3 (启用将输出写入 S3) 方框。在方框中输入存储桶和前缀（文件夹）名称。

    > **Note**
    >
    > 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给托管式节点的实例配置文件的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶

位于不同的 AWS 账户 中，请确认与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

13. 在 SNS notifications (SNS 通知) 部分中，执行以下操作：
    - 选择 Enable SNS Notifications (启用 SNS 通知)。
    - 对于 IAM role (IAM 角色)，请选择您在使用 Amazon SNS 通知监控 Systems Manager 状态更改的“任务 3”中创建的 Amazon SNS IAM 角色 Amazon Resource Name (ARN)，以启动 Amazon SNS。
    - 对于 SNS topic (SNS 主题)，请输入要使用的 Amazon SNS 主题 ARN。
    - 对于 Event type (事件类型)，请选择要针对其接收通知的事件。
    - 对于 Notification type（通知类型），请选择针对发送给多个节点（调用）或命令摘要的命令的每个副本接收通知。
14. 在 Parameters (参数) 部分中，根据您选择的命令文档输入所需的参数。
15. 选择注册运行命令任务。
16. 在下次运行维护时段后，检查您的电子邮件以查找来自 Amazon SNS 的邮件，然后打开该电子邮件。Amazon SNS 可能需要几分钟来发送该电子邮件。

将返回通知的 Run Command 任务注册到维护时段 (CLI)

可以使用以下过程，借助 AWS CLI 将配置为返回状态通知的 Run Command 任务注册到维护时段。

将返回通知的 Run Command 任务注册到维护时段 (CLI)

> **Note**
>
> 为了更好地管理您的任务选项，此过程使用命令选项 `--cli-input-json`，其选项值存储在 JSON 文件中。

1. 在您的本地计算机上，创建一个名为 `RunCommandTask.json` 的文件。
2. 将以下内容粘贴到该文件中。

```
{
    "Name": "Name",
    "Description": "Description",
```

```
    "WindowId": "mw-0c50858d01EXAMPLE",
    "ServiceRoleArn": "arn:aws:iam::account-id:role/MaintenanceWindowIAMRole",
    "MaxConcurrency": "1",
    "MaxErrors": "1",
    "Priority": 3,
    "Targets": [
        {
            "Key": "WindowTargetIds",
            "Values": [
                "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
            ]
        }
    ],
    "TaskType": "RUN_COMMAND",
    "TaskArn": "CommandDocumentName",
    "TaskInvocationParameters": {
        "RunCommand": {
            "Comment": "Comment",
            "TimeoutSeconds": 3600,
            "NotificationConfig": {
                "NotificationArn": "arn:aws:sns:region:account-id:SNSTopicName",
                "NotificationEvents": [
                    "All"
                ],
                "NotificationType": "Command"
            },
            "ServiceRoleArn": "arn:aws:iam::account-id:role/SNSIAMRole"
        }
    }
}
```

3. 将示例值替换为有关您自己的资源的信息。

   如果您想使用本示例中略去的选项，您也可以恢复它们。例如，您可以将命令输出保存到 S3 存储桶。

   有关更多信息，请参阅 AWS CLI 命令参考 中的 register-task-with-maintenance-window。

4. 保存该文件。

5. 在您的本地计算机上保存该文件的目录中，运行以下命令。

```
aws ssm register-task-with-maintenance-window --cli-input-json file://
RunCommandTask.json
```

> **Important**
>
> 务必在文件名前包含 `file://`。此命令中需要该项。

如果成功，该命令将返回类似以下内容的信息。

```
{
    "WindowTaskId": "j2l8d5b5c-mw66-tk4d-r3g9-1d4d1EXAMPLE"
}
```

6. 在下次执行维护时段后，检查您的电子邮件以查找来自 Amazon SNS 的邮件，然后打开该电子邮件。Amazon SNS 可能需要几分钟来发送该电子邮件。

有关从命令行为维护时段注册任务的更多信息，请参阅将任务注册到维护时段。

与 Systems Manager 的产品和服务集成

默认情况下，AWS Systems Manager 会与 AWS 服务以及其他产品和服务集成。以下信息可帮助您配置 Systems Manager，以与您使用的产品和服务集成。

- 与 AWS 服务 集成
- 与其他产品和服务的集成

与 AWS 服务 集成

通过使用 Systems Manager Command 文档（SSM 文档）和自动化运行手册，您可以使用 AWS Systems Manager 以与 AWS 服务集成。有关这些资源的标签的更多信息，请参阅 AWS Systems Manager 文档。

Systems Manager 已与以下 AWS 服务集成。

计算

Amazon Elastic Compute Cloud（Amazon EC2）	Amazon EC2 在 AWS Cloud中提供了可扩展的计算容量。使用 Amazon EC2 可避免前期的硬件投入，使您能够快速开发和部署应用程序。您可以使用 Amazon EC2 启动所需数量的虚拟服务器，配置安全性和联网以及管理存储。 Systems Manager 使您可以在 Amazon EC2 实例上执行多种任务。例如，您可以启动、配置、管理、维护、故障排除以及安全地连接到 Amazon EC2 实例。您还可以使用 Systems Manager 部署软件、确定合规性状态，以及从 Amazon EC2 实例收集清单。 了解更多 • 使用托管式节点 • AWS Systems Manager State Manager • AWS Systems Manager Run Command

	• AWS Systems Manager Patch Manager • AWS Systems Manager Session Manager • AWS Systems Manager Distributor • AWS Systems Manager Compliance • AWS Systems Manager 清单
Amazon EC2 Auto Scaling	Auto Scaling 可帮助确保您拥有正确数量的 Amazon EC2 实例，这些实例可用于处理应用程序的负载。您可创建 EC2 实例的集合，称为自动扩缩组。 Systems Manager 使您可以自动执行常见过程，如适用于您的 Auto Scaling 组的 Auto Scaling 模板中所使用的修补 Amazon Machine Image (AMI)。 了解更多 更新自动扩缩组的 AMIs
Amazon Elastic Container Service (Amazon ECS)	Amazon ECS 是一项高度可扩展的快速容器管理服务，它使您可以轻松运行、停止和管理集群上的 Docker 容器。 Systems Manager 使您可以远程管理容器实例，并向容器中注入敏感数据，方法是将敏感数据存储在 Parameter Store（Systems Manager 中的一项工具）的参数中，然后在容器定义中引用这些参数。 了解更多 • 使用 AWS Systems Manager 远程管理容器实例 • 使用 Systems Manager Parameter Store 指定敏感数据

AWS Lambda	Lambda 是一项计算服务，使您无需预置或管理服务器即可运行代码。只有在需要时 Lambda 才运行您的代码，并且能自动扩缩，从每天几个请求扩展到每秒数千个请求。 Systems Manager 使您可以通过使用 `aws:invokeLambdaFunction` 操作，在自动化运行手册内容中使用 Lambda 函数。 若要在 AWS Lambda 函数中使用 Parameter Store 中的参数，您可以使用 AWS 参数和密钥 Lambda 扩展来检索参数值并将其缓存，以供将来使用。 了解更多 使用 Automation、AWS Lambda 和 Parameter Store 来更新黄金 AMI 在 AWS Lambda 函数中使用 Parameter Store 参数

物联网 (IoT)

AWS IoT Greengrass 核心设备	AWS IoT Greengrass 是一项开源 IoT 边缘运行时和云服务，可帮助您在设备上构建、部署和管理 IoT 应用程序。Systems Manager 为 AWS IoT Greengrass 核心设备提供本机支持。 了解更多 使用 Systems Manager 管理边缘设备
AWS IoT 核心设备	AWS IoT 提供云服务将 IoT 设备连接到其他设备和 AWS 云服务。AWS IoT 提供设备软件以帮助您将 IoT 设备集成到基于 AWS IoT 的解决方案。如果您的设备可以连接到 AWS IoT，则 AWS IoT 可以将它们连接到 AWS 提供的云服

	务。Systems Manager 支持 AWS IoT 核心设备，前提是这些设备需在混合和多云环境中配置为托管式节点。 了解更多 使用 Systems Manager 管理混合和多云环境中的节点

存储

Amazon Simple Storage Service（Amazon S3）	Amazon S3 是一种面向互联网的存储服务。该服务旨在降低开发人员进行 Web 级计算的难度。Amazon S3 提供了一个简单 Web 服务接口，可用于随时在 Web 上的任何位置存储和检索任何数量的数据。 Systems Manager 使您可以运行存储在 Amazon S3 中的远程脚本和 SSM 文档。Distributor（AWS Systems Manager 中的一项工具）使用 Amazon S3 来存储软件包。您还可以将输出发送到 Amazon S3，用于 Run Command 和Session Manager（都是 AWS Systems Manager 中的工具）。 了解更多 • 从 Amazon S3 运行脚本 • 从远程位置运行文档 • AWS Systems Manager Distributor • 使用 Amazon S3 记录会话数据（控制台）

开发工具

AWS CodeBuild	CodeBuild 是一项在云中完全托管的生成服务。CodeBuild 可编译源代码，运行单元测试，并生成可供部署的构件。使用 CodeBuild，您无需预配置、管理和扩展自己的构建服务器。 Parameter Store 使您可以存储用于构建规范和项目的敏感信息。 了解更多 • 适用于 CodeBuild 的构建规范参考 • 在 AWS CodeBuild 中创建构建项目
AWS CDK	AWS Cloud Development Kit (AWS CDK) 是一个框架，用于使用编程语言将云基础架构定义为代码，并通过 AWS CloudFormation 进行部署。 Application Manager 允许您在 Application Manager 控制台中查看分组为应用程序的 CDK 结构，查看包括底层资源在内的应用程序结构，查看警报、调查和修复操作问题以及跟踪成本。 了解更多 • 查看有关应用程序的概述信息 • 查看应用程序资源

安全、身份和合规性

AWS Identity and Access Management（IAM）	IAM 是一种 Web 服务，可以帮助您安全地控制对 AWS 资源的访问。可以使用 IAM 来控制谁通过了身份验证（准许登录）并获得授权（具有z权限）来使用资源。

	Systems Manager 使您可以使用 IAM 控制对服务的访问。 了解更多 • AWS Systems Manager 如何与 IAM 协同工作 • AWS Systems Manager 的操作、资源和条件键 • 配置 Systems Manager 所需的实例权限
AWS Secrets Manager	Secrets Manager 提供了更轻松的秘密管理。密钥可以是数据库凭证、密码、第三方 API 密钥，甚至是任意文本。 Parameter Store 使您可以在使用其他已支持引用 Parameter Store 参数的 AWS 服务时检索 Secrets Manager 密钥。 了解更多 通过 Parameter Store 参数引用 AWS Secrets Manager 密钥

AWS Security Hub	Security Hub 可以让您全面了解多个 AWS 账户间的高优先级安全告警和合规性状态。Security Hub 可以聚合、组织来自多个 AWS 服务的安全提醒或检查结果并确定优先级。 在启用 Security Hub 与Patch Manager（AWS Systems Manager 中的一项工具）之间的集成时，Security Hub 会从安全角度监控实例集的修补状态。补丁合规性详细信息将被自动导出到 Security Hub。这使您可以使用单个视图集中监控补丁合规性状态，以及跟踪其他安全结果。您可以在机群中的节点不符合补丁合规性时收到告警，还可以在 Security Hub 控制台中审核补丁合规性结果。 您还可以将 Security Hub 与 Explorer 和 OpsCenter（都是 AWS Systems Manager 中的工具）集成。与 Security Hub 的集成使您能够接收来自 Explorer 和 OpsCenter 中的 Security Hub 的结果。Security Hub 结果可以提供安全信息，您可以在 Explorer 和 OpsCenter 中使用这些信息来聚合 AWS Systems Manager 中的安全性、性能和操作问题，并对它们采取相应措施。 使用 Security Hub 需支付费用。有关更多信息，请参阅 Security Hub 定价。 了解更多 • 接收来自 Explorer 中的 AWS Security Hub 的调查结果 • 了解 OpsCenter 与 AWS Security Hub 的集成 • 将 Patch Manager 与 AWS Security Hub 集成

加密和 PKI

AWS Key Management Service (AWS KMS)	AWS KMS 是一项托管式服务，使您可以创建和控制客户托管密钥，这是用于加密您的数据的加密密钥。 Systems Manager 使您可以使用 AWS KMS 创建 `SecureString` 参数，以及加密 Session Manager 会话数据。 了解更多 • AWS Systems Manager Parameter Store SecureString 参数的 AWS KMS 加密 • 启用会话数据的 KMS 密钥加密（控制台）

管理和治理

AWS CloudFormation	AWS CloudFormation 是一项服务，可帮助您对 Amazon Web Services 资源进行建模和设置，以便能花较少的时间管理这些资源，而花更多时间专注于 AWS 中运行的应用程序上。 Parameter Store 是动态引用的源。动态引用提供了一种简明、强大的方法，用于指定在 AWS CloudFormation 堆栈模板内的其他服务中存储和管理的外部值。 了解更多 使用动态引用以指定模板值
AWS CloudTrail	CloudTrail 是一项 AWS 服务，可帮助您授权对您的 AWS 账户执行监管、合规性以及操作和风险审计。用户、角色或 AWS 服务执行的操作将以事件的形式记录在 CloudTrail 中。事件包括在 AWS Management Console、AWS Command

	Line Interface (AWS CLI) 和 AWS 开发工具包及 API 中执行的操作。 Systems Manager 与 CloudTrail 集成，后者将大多数 Systems Manager API 调用捕获为事件。其中包括来自 Systems Manager 控制台的 API 调用和对 Systems Manager API 的调用。 了解更多 使用 AWS CloudTrail 记录 AWS Systems Manager API 调用
Amazon CloudWatch Logs	Amazon CloudWatch Logs 使您可以集中记录来自您使用的所有系统、应用程序和 AWS 服务的日志。随后您就可以查看日志、在日志中搜索特定错误代码或模式、根据特定字段筛选日志，或者安全地将这些日志归档以供将来分析。 Systems Manager 支持将 SSM Agent、Run Command 和 Session Manager 的日志发送到 CloudWatch Logs。 了解更多 • 将节点日志发送到统一的 CloudWatch Logs（CloudWatch 代理） • 为 Run Command 配置 Amazon CloudWatch Logs • 使用 Amazon CloudWatch Logs 记录会话数据（控制台）

Amazon EventBridge	EventBridge 提供近乎实时的系统事件流，这些系统事件可以描述 Amazon Web Services 资源中的更改。通过使用可快速设置的简单规则，您可以匹配事件并将事件路由到一个或多个目标函数或流。EventBridge 可在发生操作更改时感知到这些更改。EventBridge 可以响应这些操作更改，并在必要时采取纠正措施。这些操作包括发送消息以响应环境、激活函数和捕获状态信息。 Systems Manager 具有 EventBridge 支持的多个事件，使您可以根据这些事件的内容进行操作。 了解更多 使用 Amazon EventBridge 监控 Systems Manager 事件 **Note** Amazon EventBridge 是管理事件的首选方式。CloudWatch Events 和 EventBridge 是相同的底层服务和 API，但 EventBridge 提供了更多功能。您在 CloudWatch 或 EventBridge 中所做的更改将显示在每个控制台中。有关更多信息，请参阅 Amazon EventBridge 用户指南。

AWS Config	AWS Config 可以提供关于您的 AWS 账户中 AWS 资源的配置的详细信息。这些信息包括资源之间的关联方式以及资源以前的配置方式。这使您可以了解配置和关系如何随着的时间的推移而变化。 Systems Manager 与 AWS Config 集成，提供了多个规则，可帮助您了解 Amazon EC2 实例。这些规则可帮助您确定哪些 Amazon EC2 实例由 Systems Manager 管理、操作系统配置、系统级更新、已安装的应用程序、网络配置等。 了解更多 • 支持 AWS Config 的资源类型 • 记录托管实例的软件配置 • 查看清单历史记录和变更跟踪
AWS Trusted Advisor	Trusted Advisor 是一个在线工具，可提供实时指导，帮助您按照 AWS 最佳实践预置资源。 Systems Manager 可以托管 Trusted Advisor，您可以在 Explorer 中查看 Trusted Advisor 数据。 了解更多 • AWS Systems Manager Explorer • AWS Trusted Advisor 入门

AWS Organizations	Organizations 是一项账户管理服务，使您可以将多个 AWS 账户整合到您创建组织中并进行集中管理。Organizations 包含账户管理和整合账单功能，这些功能使您能够更好地满足业务的预算、安全性和合规性需求。 借助 Change Manager（AWS Systems Manager 中的一项工具）与 Organizations 之间的集成，您可以使用委托管理员账户来管理更改请求、更改模板，并可通过这一单个账户管理针对整个企业的批准。 Organizations 与 Inventory（AWS Systems Manager 中的一项工具）和 Explorer 的集成，让您可以聚合多个 AWS 区域和 AWS 账户中的清单及操作数据（OpsData）。 借助Quick Setup（AWS Systems Manager 中的一项工具）与 Organizations 之间的集成，您可以自动完成常见服务设置任务，以及根据最佳实践在各个组织单位（OU）中部署服务配置。

联网和内容分发

AWS PrivateLink	借助 AWS PrivateLink，您可以通过私有方式将您的虚拟私有云（VPC）连接到支持的 AWS 服务和 VPC 端点服务，而无需互联网网关、NAT 设备、VPN 连接或 AWS Direct Connect 连接。 Systems Manager 支持使用 AWS PrivateLink 连接到 Systems Manager API 的托管式节点。这可以改善托管式节点的安保状况，因为 AWS PrivateLink 可将托管式节点、Systems Manager 和 Amazon EC2 之间的所有网络流量限制在 Amazon 网络内。这意味着托管式节点无需访问互联网。

	了解更多 使用适用于 Systems Manager 的 VPC 端点提高 EC2 实例的安全性

Analytics

Amazon Athena	Athena 是一种交互式查询服务，使您可以使用标准 SQL 直接分析 Amazon Simple Storage Service (Amazon S3) 中的数据。只需在 AWS Management Console 中执行几项操作，即可将 Athena 指向 Amazon S3 中存储的数据，并开始使用标准 SQL 运行一次性查询，然后在几秒钟内获得结果。 Systems Manager Inventory 可与 Athena 集成，以帮助您查询来自多个 AWS 区域和 AWS 账户的清单数据。Athena 集成使用资源数据同步，以便您可以在 Systems Manager Inventory 控制台中的 Detailed View（详细视图）页面上查看来自所有托管式节点的清单数据。 了解更多 • 查询多个区域和账户的清单数据 • 演练：使用资源数据同步聚合清单数据
AWS Glue	AWS Glue 是一项完全托管的 ETL（提取、转换和加载）服务，使您能够轻松而经济高效地对数据进行分类、清理和扩充，并在各种数据存储和数据流之间可靠地移动数据。 Systems Manager 使用 AWS Glue 来爬取 Amazon S3 存储桶中的 Inventory 数据。 了解更多

	查询多个区域和账户的清单数据
Amazon QuickSight	QuickSight 是一项业务分析服务，可用于构建可视化内容、执行一次性分析，并从数据中获得业务见解。它可以自动发现 AWS 数据源，还可以使用您的数据源。 Systems Manager 资源数据同步可将从所有托管节点收集到的清单数据发送到单个 S3 存桶。您可以使用 QuickSight 查询和分析聚合数据。 了解更多 • 为 Inventory 创建资源数据同步 • 演练：使用资源数据同步聚合清单数据

应用程序集成

Amazon Simple Notification Service (Amazon SNS)	Amazon SNS 是一项 Web 服务，用于协调和管理向订阅端点或客户传输或发送消息的过程。 Systems Manager 可以生成多个服务的状态，Amazon SNS 通知可以捕获这些状态。 了解更多 • 使用 Amazon SNS 通知监控 Systems Manager 状态更改 • 基于 Parameter Store 事件设置通知或触发操作

AWS Management Console

AWS Resource Groups	Resource Groups 可将您的 AWS 资源组织起来。资源组使得同时在大量资源上管理、监控和自动化任务变得容易。

	Systems Manager 资源类型，如托管式节点、SSM 文档、维护时段、Parameter Store 参数和补丁基准，可以添加到资源组中。 了解更多 什么是 AWS Resource Groups？

主题

- 从 Amazon S3 运行脚本
- 通过 Parameter Store 参数引用 AWS Secrets Manager 密钥
- AWS Systems Manager Parameter Store SecureString 参数的 AWS KMS 加密
- 在 Amazon Elastic Kubernetes Service 中使用 Parameter Store 参数
- 在 AWS Lambda 函数中使用 Parameter Store 参数

从 Amazon S3 运行脚本

本节介绍如何从 Amazon Simple Storage Service (Amazon S3) 下载和运行脚本。以下主题包含与 Amazon S3 相关的信息和术语。要了解有关 Amazon S3 的更多信息，请参阅什么是 Amazon S3？ 您可以运行不同类型的脚本，包括 Ansible Playbooks、Python、Ruby、Shell 和 PowerShell。

还可以下载包括多个脚本的目录。在运行目录中的主脚本时，AWS Systems Manager 还会运行该目录中包含的任何引用的脚本。

请注意有关从 Amazon S3 运行脚本的以下重要详细信息：

- Systems Manager 不会验证您的脚本是否能够在节点上运行。在下载和运行脚本之前，请确认相应节点上已经安装所需软件。您也可以创建一个复合文档，能够使用 Run Command 或State Manager（都是 AWS Systems Manager 中的工具）安装软件，然后再下载并运行脚本。
- 验证并确保您的用户、角色或组已经获得读取 S3 存储桶所需的 AWS Identity and Access Management（IAM）权限。
- 确保 Amazon Elastic Compute Cloud (Amazon EC2) 实例上的实例配置文件拥有 `s3:ListBucket` 和 `s3:GetObject` 权限。如果实例配置文件没有这些权限，系统将无法从 S3 存储桶下载脚本。有关更多信息，请参阅 IAM 用户指南中的使用实例配置文件。

从 Amazon S3 运行 Shell 脚本

以下信息包含帮助您使用 AWS Systems Manager 控制台或 AWS Command Line Interface（AWS CLI）从 Amazon Simple Storage Service（Amazon S3）运行脚本的过程。尽管示例中使用了 Shell 脚本，但可以替换其他类型的脚本。

从 Amazon S3 运行 Shell 脚本（控制台）

从 Amazon S3 运行 Shell 脚本

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Run Command。
3. 选择 Run command（运行命令）。
4. 在 Command document (命令文档) 列表中，选择 **AWS-RunRemoteScript**。
5. 在命令参数中，执行以下操作：

   - 在源类型中，选择 S3。
   - 在 Source Info (源信息) 文本框中，按以下格式输入访问源所需的信息。将每个*#######*替换为您自己的信息。

     > **Note**
     >
     > 将 https://s3.*aws-api-domain* 替换为您的存储桶 URL。您可以在 Objects（对象）选项卡上复制 Amazon S3 中的存储桶 URL。

     ```
     {"path":"https://s3.aws-api-domain/path to script"}
     ```

     示例如下：

     ```
     {"path":"https://amzn-s3-demo-bucket.s3.us-east-2.amazonaws.com/scripts/shell/
     helloWorld.sh"}
     ```

   - 在 Command Line (命令行) 字段中，输入用于脚本执行的参数。见下列。

     ```
     helloWorld.sh argument-1 argument-2
     ```

- （可选）在 Working Directory（工作目录）字段中，输入节点（要在其中下载和运行脚本）上的目录的名称。
- (可选) 在执行超时中，指定脚本命令执行失败之前系统要等待的秒数。

6. 在 Targets（目标）部分中，通过指定标签、手动选择实例或边缘设备或指定资源组，选择要在其上运行此操作的托管式节点。

> **Tip**
>
> 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

7. 对于 Other parameters（其他参数）：

- 对于 Comment（注释），请输入有关此命令的信息。
- 对于 Timeout (seconds) (超时 (秒))，请指定在整个命令执行失败之前系统等待的秒数。

8. 对于 Rate control（速率控制）：

- 对于 Concurrency（并发），请指定要同时运行该命令的托管式节点的数量或百分比。

> **Note**
>
> 如果您通过指定应用于托管式节点的标签或指定 AWS Resource Groups 来选择目标，但不确定有多少个托管式节点已被设为目标，则可通过指定百分比来限制可同时运行该文档的目标的数量。

- 对于 Error threshold（错误阈值），请指定当命令在一定数量或百分比的节点上失败后，何时在其他托管式节点上停止运行该命令。例如，如果您指定三个错误，Systems Manager 将在收到第四个错误时停止发送该命令。仍在处理该命令的托管式节点也可能发送错误。

9. （可选）对于 输出选项，要将命令输出保存到文件，请选中 将命令输出写入 S3 存储桶 框。在输入框中输入存储桶和前缀（文件夹）名称。

> **Note**
>
> 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给实例的实例配置文件（适用于 EC2 实例）或 IAM 服务角色（混合激活的计算机）的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境

创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确保与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

10. 在 SNS 通知部分，如果需要发送有关命令执行状态的通知，请选中 Enable SNS notifications（启用 SNS 通知）复选框。

    有关为 Run Command 配置 Amazon SNS 通知的更多信息，请参阅 使用 Amazon SNS 通知监控 Systems Manager 状态更改。

11. 选择 Run（运行）。

从 Amazon S3 运行 Shell 脚本（命令行）

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 运行以下命令。将每个*#######*替换为您自己的信息。

> **Note**
>
> 将 https://s3.*aws-api-domain*/*script path* 替换为您的存储桶 URL。您可以在 Objects（对象）选项卡上复制 Amazon S3 中的存储桶 URL。

Linux & macOS

```
aws ssm send-command \
    --document-name "AWS-RunRemoteScript" \
    --output-s3-bucket-name "amzn-s3-demo-bucket" \
    --output-s3-key-prefix "key-prefix" \
    --targets "Key=InstanceIds,Values=instance-id" \
    --parameters '{"sourceType":["S3"],"sourceInfo":["{\"path\":\"https://
s3.aws-api-domain/script path\"}"],"commandLine":["script name and arguments"]}'
```

Windows

```
aws ssm send-command ^
    --document-name "AWS-RunRemoteScript" ^
    --output-s3-bucket-name "amzn-s3-demo-bucket" ^
    --output-s3-key-prefix "key-prefix" ^
```

```
    --targets "Key=InstanceIds,Values=instance-id" ^
    --parameters "sourceType"="S3",sourceInfo='{\"path\":\"https://s3.aws-api-
domain/script path\"}',"commandLine"="script name and arguments"
```

PowerShell

```
Send-SSMCommand `
    -DocumentName "AWS-RunRemoteScript" `
    -OutputS3BucketName "amzn-s3-demo-bucket" `
    -OutputS3KeyPrefix "key-prefix" `
    -Target @{Key="InstanceIds";Values=@("instance-id")} `
    -Parameter @{
        sourceType = "S3";
        sourceInfo = '{"path": "s3://bucket-name/path/to/script"}';
        commandLine = "script name and arguments"
    }
```

通过 Parameter Store 参数引用 AWS Secrets Manager 密钥

AWS Secrets Manager 可以帮助您组织和管理重要的配置数据（例如凭证、密码和许可证密钥）。Parameter Store（AWS Systems Manager 中的一项工具）与 Secrets Manager 集成，以便您在使用其他已支持引用 Parameter Store 参数的 AWS 服务时，可以检索 Secrets Manager 密钥。这些服务包括 Amazon Elastic Compute Cloud（Amazon EC2）、Amazon Elastic Container Service（Amazon ECS）、AWS Lambda、AWS CloudFormation、AWS CodeBuild、AWS CodeDeploy 和其他 Systems Manager 工具。通过使用 Parameter Store 引用 Secrets Manager 密钥，可创建一致且安全的过程来调用和使用代码，以及配置脚本中的密钥和引用数据。

有关 Secrets Manager 的更多信息，请参阅 AWS Secrets Manager 用户指南中的什么是 AWS Secrets Manager？。

限制

使用 Parameter Store 引用 Secrets Manager 秘密时，请注意以下限制：

- 只能通过使用 GetParameter 和 GetParameters API 操作来检索 Secrets Manager 秘密。Secrets Manager 不支持修改操作和高级查询 API 操作（例如 DescribeParameters 和 GetParametersByPath）。
- 可以使用 AWS Command Line Interface (AWS CLI)、AWS Tools for Windows PowerShell 和开发工具包通过 Parameter Store 检索秘密。

- Parameter Store 中的 Secrets Manager 秘密的前缀必须是 `/aws/reference/secretsmanager`。示例如下：

  ```
  /aws/reference/secretsmanager/CFCreds1
  ```

  ```
  /aws/reference/secretsmanager/myapp/db/password
  ```

- Parameter Store 将遵循附加到 Secrets Manager 秘密的 AWS Identity and Access Management (IAM) 策略。例如，如果用户 1 没有密钥 A 的访问权限，则用户 1 无法使用 Parameter Store 检索密钥 A。
- 引用 Secrets Manager 秘密的参数无法使用 Parameter Store 版本控制或历史记录功能。
- Parameter Store 将遵循 Secrets Manager 版本阶段。如果您引用某一版本阶段，它将使用字母、数字、句点 (.)、连字符 (-) 或下划线 (_)。在版本阶段中指定的所有其他符号都会导致引用失败。

如何使用 Parameter Store 引用 Secrets Manager 秘密

以下过程介绍如何使用 Parameter Store API 引用 Secrets Manager 秘密。该过程可以引用 AWS Secrets Manager 用户指南中的其他过程。

> **Note**
>
> 在开始之前，请验证并确保您拥有在 Parameter Store 参数中引用 Secrets Manager 秘密的权限。如果您在 Secrets Manager 和 Systems Manager 中拥有管理员权限，则可以使用 Parameter Store API 引用或检索秘密。如果您在某一 Parameter Store 参数中引用 Secrets Manager 秘密，但您没有访问该秘密的权限，则该引用将失败。有关更多信息，请参阅 AWS Secrets Manager 用户指南中的 AWS Secrets Manager 的身份验证和访问控制。

> **Important**
>
> Parameter Store 将充当对 Secrets Manager 秘密的引用的直通服务。Parameter Store 不会保留有关秘密的数据或元数据。引用是无状态的。

使用 Parameter Store 引用 Secrets Manager 秘密

1. 可以在 Secrets Manager 中创建秘密。有关更多信息，请参阅使用 AWS Secrets Manager 创建和管理密钥。

2. 使用 AWS CLI、AWS Tools for Windows PowerShell 或开发工具包引用密钥。当您引用 Secrets Manager 秘密时，名称必须以以下预留路径开头：`/aws/reference/secretsmanager/`。指定此路径后，Systems Manager 知道从 Secrets Manager 而不是 Parameter Store 检索秘密。以下是一些使用 Parameter Store 正确引用 Secrets Manager 秘密 `CFCreds1` 和 `DBPass` 的示例名称。

   - `/aws/reference/secretsmanager/CFCreds1`
   - `/aws/reference/secretsmanager/DBPass`

   以下是引用存储在 Secrets Manager 中的访问密钥和秘密密钥的 Java 代码示例。此代码示例设置 Amazon DynamoDB 客户端。代码通过 Parameter Store 检索配置数据和凭证。配置数据以字符串参数的形式存储在 Parameter Store 中，凭证存储在 Secrets Manager 中。即使配置数据和凭证存储在不同的服务中，但这两组数据均可通过 Parameter Store 使用 `GetParameter` API 访问。

```
/**
* Initialize Systems Manager client with default credentials
*/
AWSSimpleSystemsManagement ssm =
 AWSSimpleSystemsManagementClientBuilder.defaultClient();

...

/**
* Example method to launch DynamoDB client with credentials different from default
* @return DynamoDB client
*/
AmazonDynamoDB getDynamoDbClient() {
    //Getting AWS credentials from Secrets Manager using GetParameter
    BasicAWSCredentials differentAWSCreds = new BasicAWSCredentials(
            getParameter("/aws/reference/secretsmanager/access-key"),
            getParameter("/aws/reference/secretsmanager/secret-key"));

    //Initialize the DynamoDB client with different credentials
    final AmazonDynamoDB client = AmazonDynamoDBClient.builder()
            .withCredentials(new AWSStaticCredentialsProvider(differentAWSCreds))
            .withRegion(getParameter("region")) //Getting configuration from
 Parameter Store
            .build();
    return client;
}
```

```
/**
* Helper method to retrieve parameter value
* @param parameterName identifier of the parameter
* @return decrypted parameter value
*/
public GetParameterResult getParameter(String parameterName) {
    GetParameterRequest request = new GetParameterRequest();
    request.setName(parameterName);
    request.setWithDecryption(true);
    return ssm.newGetParameterCall().call(request).getParameter().getValue();
}
```

下面是一些 AWS CLI 示例。使用 `aws secretsmanager list-secrets` 命令查找秘密的名称。

AWS CLI 示例 1：使用密钥名称引用

Linux & macOS

```
aws ssm get-parameter \
    --name /aws/reference/secretsmanager/s1-secret \
    --with-decryption
```

Windows

```
aws ssm get-parameter ^
    --name /aws/reference/secretsmanager/s1-secret ^
    --with-decryption
```

此命令会返回如下信息。

```
{
    "Parameter": {
        "Name": "/aws/reference/secretsmanager/s1-secret",
        "Type": "SecureString",
        "Value": "Fl*MEishm!al875",
        "Version": 0,
        "SourceResult":
              "{
                   \"CreatedDate\": 1526334434.743,
```

```
                    \"Name\": \"s1-secret\",
                    \"VersionId\": \"aaabbbccc-1111-222-333-123456789\",
                    \"SecretString\": \"Fl*MEishm!al875\",
                    \"VersionStages\": [\"AWSCURRENT\"],
                    \"ARN\": \"arn:aws:secretsmanager:us-
east-2:123456789012:secret:s1-secret-E18LRP\"
                }"
        "LastModifiedDate": 2018-05-14T21:47:14.743Z,
        "ARN": "arn:aws:secretsmanager:us-east-2:123456789012:secret:s1-secret-
               E18LRP",
      }
}
```

AWS CLI 示例 2：包括版本 ID 的引用

Linux & macOS

```
aws ssm get-parameter \
    --name /aws/reference/secretsmanager/s1-secret:11111-aaa-bbb-ccc-123456789 \
    --with-decryption
```

Windows

```
aws ssm get-parameter ^
    --name /aws/reference/secretsmanager/s1-secret:11111-aaa-bbb-ccc-123456789 ^
    --with-decryption
```

此命令会返回如下信息。

```
{
    "Parameter": {
        "Name": "/aws/reference/secretsmanager/s1-secret",
        "Type": "SecureString",
        "Value": "Fl*MEishm!al875",
        "Version": 0,
        "SourceResult":
              "{
                    \"CreatedDate\": 1526334434.743,
                    \"Name\": \"s1-secret\",
                    \"VersionId\": \"11111-aaa-bbb-ccc-123456789\",
                    \"SecretString\": \"Fl*MEishm!al875\",
```

```
                    \"VersionStages\": [\"AWSCURRENT\"],
                    \"ARN\": \"arn:aws:secretsmanager:us-
east-2:123456789012:secret:s1-secret-E18LRP\"
                }"
          "Selector": ":11111-aaa-bbb-ccc-123456789"
       }
          "LastModifiedDate": 2018-05-14T21:47:14.743Z,
          "ARN": "arn:aws:secretsmanager:us-east-2:123456789012:secret:s1-secret-
                 E18LRP",
}
```

AWS CLI 示例 3：包括版本阶段的引用

Linux & macOS

```
aws ssm get-parameter \
    --name /aws/reference/secretsmanager/s1-secret:AWSCURRENT \
    --with-decryption
```

Windows

```
aws ssm get-parameter ^
    --name /aws/reference/secretsmanager/s1-secret:AWSCURRENT ^
    --with-decryption
```

此命令会返回如下信息。

```
{
    "Parameter": {
        "Name": "/aws/reference/secretsmanager/s1-secret",
        "Type": "SecureString",
        "Value": "Fl*MEishm!al875",
        "Version": 0,
        "SourceResult":
              "{
                    \"CreatedDate\": 1526334434.743,
                    \"Name\": \"s1-secret\",
                    \"VersionId\": \"11111-aaa-bbb-ccc-123456789\",
                    \"SecretString\": \"Fl*MEishm!al875\",
                    \"VersionStages\": [\"AWSCURRENT\"],
```

```
                    \"ARN\": \"arn:aws:secretsmanager:us-
east-2:123456789012:secret:s1-secret-E18LRP\"
                }"
          "Selector": ":AWSCURRENT"
       }
       "LastModifiedDate": 2018-05-14T21:47:14.743Z,
       "ARN": "arn:aws:secretsmanager:us-east-2:123456789012:secret:s1-secret-
                E18LRP",
}
```

AWS Systems Manager Parameter Store SecureString 参数的 AWS KMS 加密

借助 AWS Systems Manager Parameter Store，您可以创建 SecureString 参数，这些参数具有明文参数名称和加密的参数值。Parameter Store 使用 AWS KMS 加密和解密 `SecureString` 参数的参数值。

借助 Parameter Store，您可以创建、存储和管理数据作为具有值的参数。您可以在 Parameter Store 中创建一个参数，然后在多个受您设计的策略和权限限制的应用程序和服务中使用该参数。在需要更改参数值时，您可以更改一个实例，而不是管理对众多源进行的更改，后者很容易出错。Parameter Store 支持参数名称采用分层结构，因此，您可以将某个参数限制为用于特定的用途。

要管理敏感数据，您可以创建 `SecureString` 参数。在创建或更改 `SecureString` 参数的参数值时，Parameter Store 使用 AWS KMS keys 来加密这些参数值。在访问这些参数值时，它也使用 KMS 密钥解密这些值。您可以使用 Parameter Store 为您的账户创建的 AWS 托管式密钥，或者指定您自己的客户自主管理型密钥。

> **Important**
>
> Parameter Store 仅支持对称 KMS 密钥。不能使用非对称 KMS 密钥来加密您的参数。如需帮助确定 KMS 密钥是对称的还是非对称的，请参阅《AWS Key Management Service 开发人员指南》中的识别不同的密钥类型。

Parameter Store 支持两个级别的`SecureString`参数：标准 和高级，不能超过 4096 字节的标准参数直接在指定的 KMS 密钥下进行加密和解密。为了加密和解密高级`SecureString`参数，Parameter Store 将信封加密与 AWS Encryption SDK结合使用。您可以将标准`SecureString`参数转换为高级参

数，但不能将高级参数转换为标准参数。有关标准和高级 SecureString 参数之间的差异的更多信息，请参阅管理参数层。

主题

- 保护标准 SecureString 参数
- 保护高级 SecureString 参数
- 设置权限以加密和解密参数值
- Parameter Store 加密上下文
- 排查 Parameter Store 中的 KMS 密钥问题

保护标准 SecureString 参数

Parameter Store 不执行任何加密操作。相反，它依赖于 AWS KMS 加密和解密SecureString参数值。在创建或更改标准SecureString参数值时，Parameter Store 会调用 AWS KMS Encrypt 操作。该操作直接使用对称加密 KMS 密钥加密参数值，而不是使用 KMS 密钥生成数据密钥。

您可以选择 Parameter Store 用于加密参数值的 KMS 密钥。如果未指定 KMS 密钥，Parameter Store 会使用 Systems Manager 在您的账户中自动创建的 AWS 托管式密钥。此 KMS 密钥具有 aws/ssm 别名。

要查看您的账户的默认 aws/ssm KMS 密钥，请使用 AWS KMS API 中的 DescribeKey 操作。以下示例在别名为 describe-key 的 AWS Command Line Interface (AWS CLI) 中使用 aws/ssm 命令。

```
aws kms describe-key \
    --key-id alias/aws/ssm
```

要创建标准 SecureString 参数，请使用 Systems Manager API 中的 PutParameter 操作。省略 Tier 参数或指定 Standard 的值（它是默认值）。请包含具有 SecureString 值的 Type 参数。要指定 KMS 密钥，请使用 KeyId 参数。默认值为您的账户的 AWS 托管式密钥，即 aws/ssm。

然后，Parameter Store 使用 KMS 密钥和明文参数值调用 AWS KMS Encrypt 操作。AWS KMS 返回加密的参数值，Parameter Store 会将该值与参数名称一起存储。

以下示例在 AWS CLI 中使用 Systems Manager put-parameter 命令及其 --type 参数来创建 SecureString 参数。由于该命令省略了可选的 --tier 和 --key-id 参数，因此，Parameter Store 将创建标准 SecureString 参数，并在 AWS 托管式密钥下对其进行加密。

```
aws ssm put-parameter \
    --name MyParameter \
    --value "secret_value" \
    --type SecureString
```

以下类似的示例使用 `--key-id` 参数指定客户托管密钥。该示例使用 KMS 密钥 ID 标识 KMS 密钥，但您可以使用任何有效的 KMS 密钥标识符。由于该命令省略 `Tier` 参数 (`--tier`)，因此，Parameter Store 将创建标准`SecureString`参数而非高级参数。

```
aws ssm put-parameter \
    --name param1 \
    --value "secret" \
    --type SecureString \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

在从 Parameter Store 获取`SecureString`参数时，将对其值进行加密。要获取参数，请使用 Systems Manager API 中的 GetParameter 操作。

以下示例在 AWS CLI 中使用 Systems Manager get-parameter 命令从 Parameter Store 中获取 `MyParameter` 参数，而不解密其值。

```
aws ssm get-parameter --name MyParameter
```

```
{
    "Parameter": {
        "Type": "SecureString",
        "Name": "MyParameter",
        "Value": "AQECAHgnOkMROh5LaLXkA4j0+vYi6tmM17Lg"
    }
}
```

要在返回之前解密参数值，请将 `GetParameter` 的 `WithDecryption` 参数设置为 `true`。在使用 `WithDecryption` 时，Parameter Store 代表您调用 AWS KMS Decrypt 操作以解密该参数值。因此，`GetParameter` 请求返回参数以及明文参数值，如以下示例中所示。

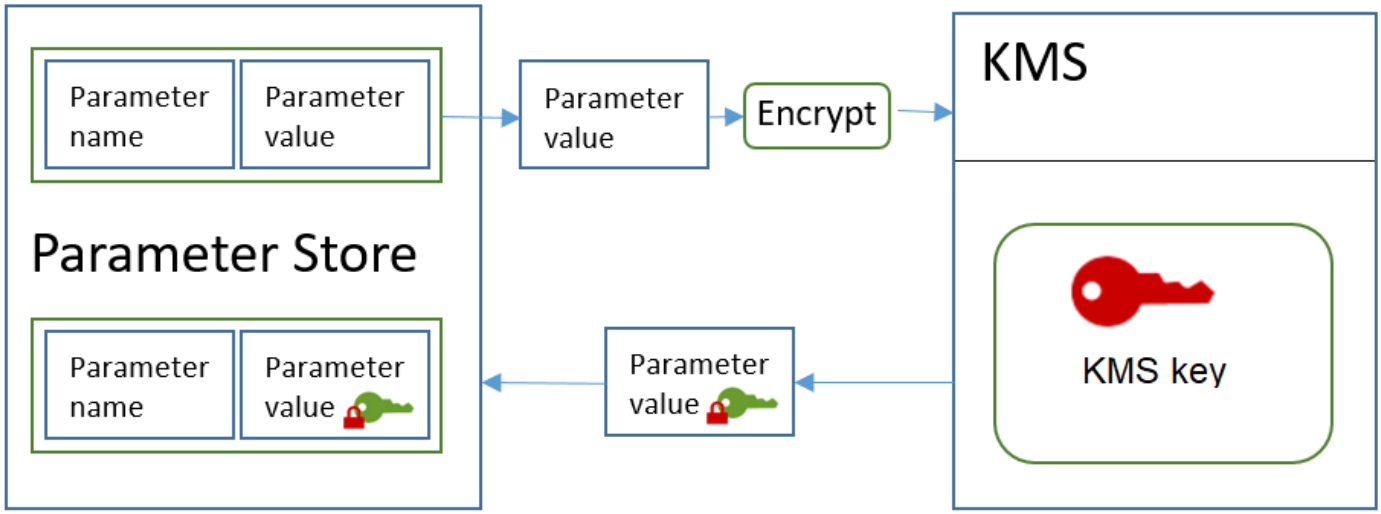
```
aws ssm get-parameter \
    --name MyParameter \
    --with-decryption
```

```
{
    "Parameter": {
        "Type": "SecureString",
        "Name": "MyParameter",
        "Value": "secret_value"
    }
}
```

以下工作流演示 Parameter Store 如何使用 KMS 密钥来加密和解密标准 SecureString 参数。

加密标准参数

1. 在使用 PutParameter 创建 SecureString 参数时，Parameter Store 将向 AWS KMS 发送 Encrypt 请求。该请求包含明文参数值、您选择的 KMS 密钥和 Parameter Store 加密上下文。在传输到 AWS KMS 期间，将通过传输层安全性 (TLS) 保护SecureString参数中的明文值。
2. AWS KMS 使用指定的 KMS 密钥和加密上下文来加密参数值。它将密文返回到 Parameter Store，后者将存储参数名称及其加密值。



解密标准参数

1. 当您在 GetParameter 请求中包含 WithDecryption 参数时，Parameter Store 会向 AWS KMS 发送包含加密的 SecureString 参数值和 Parameter Store 加密上下文的 Decrypt 请求。
2. AWS KMS 使用相同的 KMS 密钥和提供的加密上下文来解密加密的值。它向 Parameter Store 返回明文（解密的）参数值。在传输期间，将通过 TLS 保护明文数据。
3. Parameter Store 在 GetParameter 响应中向您返回明文参数值。

保护高级 SecureString 参数

当您使用 PutParameter 创建高级 SecureString 参数时，Parameter Store 会使用信封加密与 AWS Encryption SDK 和对称加密 AWS KMS key来保护参数值。每个高级参数值都使用唯一数据密钥加密，数据密钥使用 KMS 密钥加密。您可以将 AWS 托管式密钥 用于账户 (aws/ssm) 或任何客户托管密钥。

AWS Encryption SDK 是一个开源客户端库，可帮助您使用行业标准和最佳实践来加密和解密数据。它在多个平台上受支持并支持多种编程语言，包括命令行界面。您可以在 GitHub 中查看源代码并加入源代码的开发工作。

对于每个SecureString参数值，Parameter Store 调用 AWS Encryption SDK 以使用 AWS KMS 生成的唯一数据密钥 (GenerateDataKey) 来加密参数值。AWS Encryption SDK 向 Parameter Store 返回一条加密消息（其中包含加密参数值）和唯一数据密钥的加密副本。Parameter Store 将整个加密消息存储在SecureString参数值中。之后，在您获取高级SecureString参数值时，Parameter Store 使用 AWS Encryption SDK 解密参数值。这需要调用 AWS KMS 以解密加密的数据密钥。

要创建高级 SecureString 参数，请使用 Systems Manager API 中的 PutParameter 操作。将 Tier 参数的值设置为 Advanced。请包含具有 SecureString 值的 Type 参数。要指定 KMS 密钥，请使用 KeyId 参数。默认值为您的账户的 AWS 托管式密钥，即 aws/ssm。

```
aws ssm put-parameter \
    --name MyParameter \
    --value "secret_value" \
    --type SecureString \
    --tier Advanced
```

以下类似的示例使用 --key-id 参数指定客户托管密钥。此示例使用 KMS 密钥的 Amazon Resource Name (ARN)，但您可以使用任何有效的 KMS 密钥标识符。

```
aws ssm put-parameter \
    --name MyParameter \
    --value "secret_value" \
    --type SecureString \
    --tier Advanced \
    --key-id arn:aws:kms:us-
east-2:987654321098:key/1234abcd-12ab-34cd-56ef-1234567890ab
```

在从 Parameter Store 获取SecureString参数时，其值为 AWS Encryption SDK 所返回的加密消息。要获取参数，请使用 Systems Manager API 中的 GetParameter 操作。

以下示例使用 Systems Manager GetParameter 操作从 Parameter Store 获取 MyParameter 参数，而不解密其值。

```
aws ssm get-parameter --name MyParameter
```

```
{
    "Parameter": {
        "Type": "SecureString",
        "Name": "MyParameter",
        "Value": "AQECAHgnOkMROh5LaLXkA4j0+vYi6tmM17Lg"
    }
}
```

要在返回之前解密参数值，请将 GetParameter 的 WithDecryption 参数设置为 true。在使用 WithDecryption 时，Parameter Store 代表您调用 AWS KMS Decrypt 操作以解密该参数值。因此，GetParameter 请求返回参数以及明文参数值，如以下示例中所示。

```
aws ssm get-parameter \
    --name MyParameter \
    --with-decryption
```

```
{
    "Parameter": {
        "Type": "SecureString",
        "Name": "MyParameter",
        "Value": "secret_value"
    }
}
```

您无法将高级SecureString参数转换为标准参数，但可以将标准SecureString参数转换为高级参数。要将标准SecureString参数转换为高级SecureString参数，请将 PutParameter 操作与 Overwrite 参数结合使用。Type 必须为 SecureString，并且 Tier 值必须为 Advanced。KeyId 参数（它标识客户托管密钥）是可选的。如果省略此参数，Parameter Store 将使用该账户的 AWS 托管式密钥。您可以指定委托人有权使用的任何 KMS 密钥，即使您使用不同的 KMS 密钥来加密标准参数也是如此。

当您使用 Overwrite 参数时，Parameter Store 使用 AWS Encryption SDK 加密参数值。然后，它会将新加密的消息存储在 Parameter Store 中。

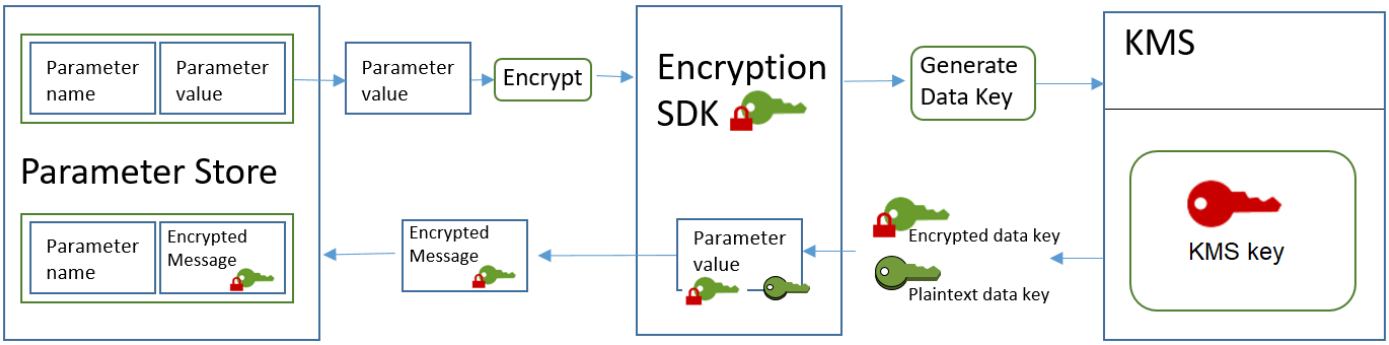
```
aws ssm put-parameter \
```

```
    --name myStdParameter \
    --value "secret_value"  \
    --type SecureString \
    --tier Advanced \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --overwrite
```

以下工作流介绍 Parameter Store 如何使用 KMS 密钥来加密和解密高级 `SecureString` 参数。

加密高级参数

1. 当您使用 `PutParameter` 创建高级 `SecureString` 参数时，Parameter Store 会使用 AWS Encryption SDK 和 AWS KMS 加密参数值。Parameter Store 调用带参数值的 AWS Encryption SDK、您指定的 KMS 密钥以及 Parameter Store 加密上下文。
2. AWS Encryption SDK 使用您指定的 KMS 密钥的标识符和 Parameter Store 加密上下文向 AWS KMS 发送 GenerateDataKey。AWS KMS 返回唯一数据密钥的两个副本：一个是明文形式，一个使用 KMS 密钥进行了加密。（加密数据密钥时将使用加密上下文。）
3. AWS Encryption SDK 使用明文数据密钥对参数值进行加密。它返回一条加密消息，其中包含加密的参数值、加密的数据密钥和其他数据（包括 Parameter Store 加密上下文）。
4. Parameter Store 将加密消息存储为参数值。



解密高级参数

1. 您可以将 `WithDecryption` 参数包含在 `GetParameter` 请求中以获取高级`SecureString`参数。在执行此操作时，Parameter Store 将加密消息从参数值传递到 AWS Encryption SDK 的解密方法。
2. AWS Encryption SDK 调用 AWS KMS Decrypt 操作。它传入加密消息中的加密的数据密钥和 Parameter Store 加密上下文。

3. AWS KMS 使用 KMS 密钥和 Parameter Store 加密上下文来解密加密的数据密钥。然后，它向 AWS Encryption SDK 返回明文（解密的）数据密钥 。
4. AWS Encryption SDK 使用明文数据密钥对参数值进行解密。它向 Parameter Store 返回明文参数值。
5. Parameter Store 验证您的加密上下文，并在 `GetParameter` 响应中向您返回明文参数值。

设置权限以加密和解密参数值

要加密标准`SecureString`参数值，用户需要 `kms:Encrypt` 权限。要加密高级`SecureString`参数值，用户需要 `kms:GenerateDataKey` 权限。要解密任一类型的`SecureString`参数值，用户需要 `kms:Decrypt` 权限。

您可以使用 AWS Identity and Access Management (IAM) 策略来允许或拒绝用户调用 Systems Manager `PutParameter` 和 `GetParameter` 操作的权限。

如果使用客户自主管理型密钥来加密您的 `SecureString` 参数值，则可以使用 IAM 策略和密钥政策来管理加密和解密权限。不过，您无法为默认的 `aws/ssm` KMS 密钥制定访问控制策略。有关如何控制对客户自主管理型密钥的访问权限的详细信息，请参阅《AWS Key Management Service 开发人员指南》中的KMS密钥访问和权限。

以下示例展示了为标准 `SecureString` 参数设计的 IAM 策略。此策略允许用户对 `FinancialParameters` 路径中的所有参数调用 Systems Manager `PutParameter` 操作。此策略还允许用户对示例客户托管密钥调用 AWS KMS `Encrypt` 操作。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:PutParameter"
            ],
            "Resource": "arn:aws:ssm:us-east-2:111122223333:parameter/
FinancialParameters/*"
        },
        {
            "Effect": "Allow",
```

```
            "Action": [
                "kms:Encrypt"
            ],
            "Resource": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ]
}
```

下一个示例展示了为高级 SecureString 参数设计的 IAM 策略。此策略允许用户对 ReservedParameters 路径中的所有参数调用 Systems Manager PutParameter 操作。此策略还允许用户对示例客户托管密钥调用 AWS KMS GenerateDataKey 操作。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:PutParameter"
            ],
            "Resource": "arn:aws:ssm:us-east-2:111122223333:parameter/
ReservedParameters/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey"
            ],
            "Resource": "arn:aws:kms:us-east-2:111122223333:key/key-id"
        }
    ]
}
```

最后一个示例还显示了一个 IAM 策略，该策略可用于标准或高级SecureString参数。此策略允许用户对 ITParameters 路径中的所有参数调用 Systems Manager GetParameter 操作（以及相关操作）。此策略还允许用户对示例客户托管密钥调用 AWS KMS Decrypt 操作。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameter*"
            ],
            "Resource": "arn:aws:ssm:us-east-2:111122223333:parameter/
ITParameters/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ]
}
```

Parameter Store 加密上下文

加密上下文 是一组包含任意非机密数据的键值对。在请求中包含加密上下文以加密数据时，AWS KMS 以加密方式将加密上下文绑定到加密的数据。要解密数据，您必须传入相同的加密上下文。

您还可以使用加密上下文在审核记录和日志中识别加密操作。加密上下文在日志中以明文形式显示，例如，AWS CloudTrail 日志。

虽然 AWS Encryption SDK 也接受加密上下文，但它以不同方式处理该上下文。Parameter Store 向加密方法提供加密上下文。AWS Encryption SDK 以加密方式将加密上下文绑定到加密数据。它还以明文形式将加密上下文包含在其返回的加密消息的标头中。但与 AWS KMS 不同，AWS Encryption SDK 解密方法不将加密上下文作为输入。相反，在解密数据时，AWS Encryption SDK 将获取加密消息中的加密上下文。Parameter Store 验证加密上下文是否包含在将明文参数值返回给您之前所期望的值。

Parameter Store 在加密操作中使用以下加密上下文：

- 键：PARAMETER_ARN

- 值：要加密的参数的 Amazon Resource Name (ARN)。

加密上下文的格式如下所示：

```
"PARAMETER_ARN":"arn:aws:ssm:region-id:account-id:parameter/parameter-name"
```

例如，Parameter Store 在调用中包含此加密上下文，以在示例 AWS 账户和区域中加密和解密 `MyParameter` 参数。

```
"PARAMETER_ARN":"arn:aws:ssm:us-east-2:111122223333:parameter/MyParameter"
```

如果该参数位于 Parameter Store 分层路径中，则在加密上下文中包含路径和名称。例如，在示例 AWS 账户 和区域中加密和解密 `/ReadableParameters` 路径中的 `MyParameter` 参数时，将使用此加密上下文。

```
"PARAMETER_ARN":"arn:aws:ssm:us-east-2:111122223333:parameter/ReadableParameters/
MyParameter"
```

您可以通过使用正确的加密上下文和 Systems Manager `GetParameter` 操作返回的加密参数值调用 AWS KMS `Decrypt` 操作来解密加密的 `SecureString` 参数值。不过，我们建议您将 `GetParameter` 操作与 `WithDecryption` 参数一起使用以解密 Parameter Store 参数值。

您还可以在 IAM policy 中包含加密上下文。例如，您可以允许用户仅解密某个特定参数值或一组参数值。

以下示例 IAM policy 语句允许用户获取 `MyParameter` 参数值，以及使用指定的 KMS 密钥解密该参数值。不过，只有在加密上下文与指定的字符串匹配时，这些权限才适用。这些权限不适用于任何其他参数或 KMS 密钥；如果加密上下文与该字符串不匹配，对 `GetParameter` 的调用将失败。

在使用类似于此的策略语句之前，请将*## ARN* 替换为有效值。

JSON

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

```
            "Action": [
                "ssm:GetParameter*"
            ],
            "Resource": "arn:aws:ssm:us-
east-2:111122223333:parameter/MyParameter"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "arn:aws:kms:us-east-2:111122223333:key/key-id",
            "Condition": {
                "StringEquals": {
                    "kms:EncryptionContext:PARAMETER_ARN":"arn:aws:ssm:us-
east-2:111122223333:parameter/MyParameter"
                }
            }
        }
    ]
}
```

排查 Parameter Store 中的 KMS 密钥问题

要对 `SecureString` 参数执行任何操作，Parameter Store 必须能够使用您为预期操作指定的 AWS KMS KMS 密钥。与 KMS 密钥相关的大多数 Parameter Store 失败都是由以下问题造成的：

- 应用程序使用的凭证无权对 KMS 密钥执行指定的操作。

  要修复该错误，请使用不同的凭证运行应用程序，或者修改妨碍执行该操作的 IAM 或密钥策略。有关 AWS KMS IAM 和密钥政策的帮助，请参阅《AWS Key Management Service 开发人员指南》中的 KMS 密钥访问和权限。

- 找不到 KMS 密钥。

  在 KMS 密钥使用不正确的标识符时，通常会发生这种情况。请查找 KMS 密钥的正确标识符，然后重试该命令。

- KMS 密钥未启用。在发生这种情况时，Parameter Store 返回 InvalidKeyId 异常，并包含来自 AWS KMS 的详细错误消息。如果 KMS 密钥状态为 `Disabled`，请启用它。如果状态为 `Pending Import`，请完成导入过程。如果密钥状态为 `Pending Deletion`，请取消密钥删除或使用不同的 KMS 密钥。

要查找 KMS 密钥的密钥状态，请使用 DescribeKey 操作。

在 Amazon Elastic Kubernetes Service 中使用 Parameter Store 参数

要将 Parameter Store（AWS Systems Manager 的工具）中的参数显示为挂载在 Amazon EKS 容器组（pod）中的文件，您可以使用 Kubernetes Secrets Store CSI 驱动程序的 AWS Secrets and Configuration Provider。ASCP 与运行 Amazon EC2 节点组的 Amazon Elastic Kubernetes Service 1.17+ 结合使用。不支持 AWS Fargate 节点组。

借助 ASCP，您可以在 Parameter Store 中存储并管理参数，然后通过在 Amazon EKS 上运行的工作负载检索参数。如果参数包含多个 JSON 格式的键/值对，您可以选择要挂载到 Amazon EKS 中的键/值对。ASCP 可使用 JMESPath 语法来查询密钥中的键/值对。ASCP 也适用于 AWS Secrets Manager 密钥。

ASCP 提供两种通过 Amazon EKS 进行身份验证的方法。第一种方法是使用服务账户的 IAM 角色（IRSA），第二种方法是使用容器组身份。每种方法都有其优势和用例。

基于服务账户的 IAM 角色（IRSA）的 ASCP

基于服务账户的 IAM 角色（IRSA）的 ASCP 允许您将 Parameter Store 中的参数作为文件挂载到 Amazon EKS 容器组（pod）中。这种方法适用于以下情况：

- 需要将参数作为文件挂载到容器组（pod）中时。
- 将 Amazon EKS 版本 1.17 或更高版本与 Amazon EC2 节点组结合使用时。
- 希望从 JSON 格式的参数中检索特定的键/值对时。

有关更多信息，请参阅 the section called “将 ASCP 与适用于 Amazon EKS 的 IRSA 集成”。

基于容器组身份的 ASCP

基于 EKS 容器组身份的 ASCP

基于容器组身份的 ASCP 方法增强了安全性，简化了访问 Parameter Store 中参数的配置。这种方法在以下情况下非常有用：

- 需要在容器组（pod）级别进行更精细的权限管理时。
- 使用的是 Amazon EKS 版本 1.24 或更高版本时。
- 需要提高性能和可扩展性时。

有关更多信息，请参阅 the section called “将 ASCP 与适用于 Amazon EKS 的容器组身份集成”。

选择正确的方法

在基于 IRSA 的 ASCP 和基于容器组身份的 ASCP 之间做选择时，需考虑以下因素：

- Amazon EKS 版本：容器组身份适用于 Amazon EKS 1.24 或更高版本，CSI 驱动程序则适用于 Amazon EKS 1.17 或更高版本。
- 安全要求：容器组身份可在容器组（pod）级别提供更精细的控制。
- 性能：容器组身份通常会在大规模环境中展现更出色的性能。
- 复杂性：容器组身份无需单独的服务账户，可以简化设置。

选择最符合您的具体要求和 Amazon EKS 环境的方法。

安装适用于 Amazon EKS 的 ASCP

本节介绍了如何安装适用于 Amazon EKS 的 AWS Secrets and Configuration Provider。借助 ASCP，您可以将 Parameter Store 的参数和 AWS Secrets Manager 的密钥作为文件挂载到 Amazon EKS 容器组（pod）中。

先决条件

- Amazon EKS 集群
  - 容器组身份版本 1.24 或更高版本
  - IRSA 版本 1.17 或更高版本
- 已安装并配置 AWS CLI
- 已为 Amazon EKS 集群安装并配置 kubectl
- Helm（版本 3.0 或更高版本）

安装和配置 ASCP

ASCP 在 secrets-store-csi-provider-aws 存储库中的 GitHub 可用。此存储库还包含 YAML 文件示例，该文件可通过将 `objectType` 值从 `secretsmanager` 更改为 `ssmparameter` 来创建和挂载密钥。

安装过程中，您可以将 ASCP 配置为使用 FIPS 端点。有关 Systems Manager 端点列表，请参阅《Amazon Web Services 一般参考》 中的 Systems Manager service endpoints。

使用 Helm 安装 ASCP

1. 为确保存储库指向最新图表，请使用 `helm repo update.`
2. 添加 Secrets Store CSI Driver 图表。

```
helm repo add secrets-store-csi-driver https://kubernetes-sigs.github.io/secrets-store-csi-driver/charts
```

3. 安装图表。要配置节流，请添加以下标志：`--set-json 'k8sThrottlingParams={"qps": "number of queries per second", "burst": "number of queries per second"}'`

```
helm install -n kube-system csi-secrets-store secrets-store-csi-driver/secrets-store-csi-driver
```

4. 添加 ASCP 图表。

```
helm repo add aws-secrets-manager https://aws.github.io/secrets-store-csi-driver-provider-aws
```

5. 安装图表。要使用 FIPS 端点，请添加以下标志：`--set useFipsEndpoint=true`

```
helm install -n kube-system secrets-provider-aws aws-secrets-manager/secrets-store-csi-driver-provider-aws
```

在存储库中使用 YAML 进行安装

- 使用以下命令。

```
helm repo add secrets-store-csi-driver https://kubernetes-sigs.github.io/secrets-store-csi-driver/charts
helm install -n kube-system csi-secrets-store secrets-store-csi-driver/secrets-store-csi-driver
kubectl apply -f https://raw.githubusercontent.com/aws/secrets-store-csi-driver-provider-aws/main/deployment/aws-provider-installer.yaml
```

验证安装情况

要验证 EKS 集群、Secrets Store CSI 驱动程序和 ASCP 插件的安装情况，请按照以下步骤操作：

1. 验证 EKS 集群的安装情况：

```
eksctl get cluster --name clusterName
```

该命令应返回有关集群的信息。

2. 验证 Secrets Store CSI 驱动程序的安装情况：

```
kubectl get pods -n kube-system -l app=secrets-store-csi-driver
```

您应该看到正在运行的容器组（pod），其名称如下：`csi-secrets-store-secrets-store-csi-driver-xxx`。

3. 验证 ASCP 插件的安装情况：

YAML installation

```
$ kubectl get pods -n kube-system -l app=csi-secrets-store-provider-aws
```

输出示例：

```
NAME                                     READY   STATUS    RESTARTS   AGE
csi-secrets-store-provider-aws-12345      1/1     Running   0          2m
```

Helm installation

```
$  kubectl get pods -n kube-system -l app=secrets-store-csi-driver-provider-aws
```

输出示例：

```
NAME                                                READY   STATUS    RESTARTS
 AGE
secrets-provider-aws-secrets-store-csi-driver-provider-67890       1/1
 Running   0          2m
```

您应该看到处于 Running 状态的容器组（pod）。

运行这些命令后，如果一切设置正确，您应该会看到所有组件都在运行，且没有任何错误。如果遇到任何问题，可能需要通过查看出现问题的特定容器组（pod）的日志来进行故障排除。

故障排除

1. 要查看 ASCP 提供者的日志，请运行：

```
kubectl logs -n kube-system -l app=csi-secrets-store-provider-aws
```

2. 检查 kube-system 命名空间中所有容器组（pod）的状态。

   将*#######*替换为自己的容器组（pod）ID：

```
kubectl -n kube-system get pods
```

```
kubectl -n kube-system logs pod/pod-id
```

   所有与 CSI 驱动程序和 ASCP 相关的容器组（pod）都应处于“正在运行”状态。

3. 检查 CSI 驱动程序版本：

```
kubectl get csidriver secrets-store.csi.k8s.io -o yaml
```

   该命令应返回有关已安装的 CSI 驱动程序的信息。

其他资源

有关将 ASCP 与 Amazon EKS 结合使用的更多信息，请参阅以下资源：

- 将容器组身份与 Amazon EKS 结合使用
- GitHub 上的 AWS Secrets Store CSI 驱动程序

将 AWS Secrets and Configuration Provider CSI 与适用于 Amazon EKS 的容器组身份结合使用

AWS Secrets and Configuration Provider 与适用于 Amazon Elastic Kubernetes Service 的容器组身份代理集成，为在 Amazon EKS 上运行的应用程序提供了增强的安全性、简化的配置以及更高的性能。容器组身份简化了从 AWS Systems Manager Parameter Store 检索参数或从 Secrets Manager 检索密钥时的适用于 Amazon EKS 的 AWS Identity and Access Management（IAM）身份验证流程。

Amazon EKS 容器组身份通过直接在 Amazon EKS 接口设置权限，减少了操作步骤，且无需在 Amazon EKS 和 IAM 服务之间切换，从而简化了为 Kubernetes 应用程序配置 IAM 权限的过程。容器组身份允许在多个集群中共用一个 IAM 角色而无需更新信任策略，并支持角色会话标签以实现更精细的访问控制。这种方法不仅允许跨角色重复使用权限策略，简化了策略管理，而且还允许根据匹配的标签访问 AWS 资源，增强了安全性。

工作方式

1. 容器组身份会为容器组（pod）分配 IAM 角色。
2. ASCP 会使用此角色通过 AWS 服务 进行身份验证。
3. 如果获得授权，ASCP 会检索请求的参数并将其提供给容器组（pod）。

有关更多信息，请参阅《Amazon EKS 用户指南》中的了解 Amazon EKS 容器组身份的工作原理。

先决条件

> **Important**
>
> 仅云中的 Amazon EKS 支持容器组身份。Amazon EKS Anywhere、AWS 云端 Red Hat OpenShift 服务 或 Amazon EC2 实例上自行管理的 Kubernetes 集群不支持容器组身份。

- Amazon EKS 集群（版本 1.24 或更高版本）
- 通过 `kubectl` 访问 AWS CLI 和 Amazon EKS 集群
- （可选）访问两个 AWS 账户以实现跨账户访问

安装 Amazon EKS 容器组身份代理

要将容器组身份与集群结合使用，必须安装 Amazon EKS 容器组身份代理附加组件。

安装容器组身份代理

- 在集群上安装容器组身份代理附加组件。

  将*******替换为自己的值：

```
eksctl create addon \
  --name eks-pod-identity-agent \
```

```
  --cluster clusterName \
  --region region
```

通过容器组身份设置 ASCP

1. 创建权限策略，授予对容器组（pod）需要访问的参数的 `ssm:GetParameters` 和 `ssm:DescribeParameters` 权限。
2. 创建可由容器组身份的 Amazon EKS 服务主体担任的 IAM 角色：

   JSON

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
         {
           "Effect": "Allow",
           "Principal": {
             "Service": "pods.eks.amazonaws.com"
           },
           "Action": [
             "sts:AssumeRole",
             "sts:TagSession"
           ],
           "Resource": "arn:aws:iam::123456789012:role/EKSPodIdentityRole"
         }
       ]
     }
   ```

   向角色附加 IAM 策略。

   将*#######*替换为自己的值：

   ```
   aws iam attach-role-policy \
     --role-name MY_ROLE \
     --policy-arn POLICY_ARN
   ```

3. 创建容器组身份关联。有关示例，请参阅《Amazon EKS 用户指南》中的创建容器组身份关联
4. 创建 `SecretProviderClass`，用于指定要挂载到容器组（pod）中的参数或密钥：

```
kubectl apply -f kubectl apply -f https://raw.githubusercontent.com/aws/
secrets-store-csi-driver-provider-aws/main/examples/ExampleSecretProviderClass-
PodIdentity.yaml
```

IRSA 和容器组身份在 SecretProviderClass 中的关键区别在于可选参数 usePodIdentity。这是一个可选字段，用于确定身份验证方法。如果未指定，则默认对服务账户（IRSA）使用 IAM 角色。

- 要使用 EKS 容器组身份，请使用以下任意值："true", "True", "TRUE", "t", "T"。
- 要明确使用 IRSA，请将其设置为以下任意值："false", "False", "FALSE", "f", or "F"。

5. 部署挂载 /mnt/secrets-store 下的参数或密钥的容器组（pod）：

```
kubectl apply -f kubectl apply -f https://raw.githubusercontent.com/aws/secrets-
store-csi-driver-provider-aws/main/examples/ExampleDeployment-PodIdentity.yaml
```

6. 如果使用私有 Amazon EKS 集群，请确保该集群所在的 VPC 具有 AWS STS 端点。有关创建端点的信息，请参阅《AWS Identity and Access Management 用户指南》中的接口 VPC 端点。

验证密钥的挂载情况

要验证参数或密钥是否正确挂载，请运行以下命令。

将*#######*替换为自己的值：

```
kubectl exec -it $(kubectl get pods | awk '/pod-identity-deployment/{print $1}' | head
 -1) -- cat /mnt/secrets-store/MyParameter
```

设置 Amazon EKS 容器组身份以访问 Parameter Store 中的参数

1. 创建权限策略，授予对容器组（pod）需要访问的参数的 ssm:GetParameters 和 ssm:DescribeParameters 权限。
2. 如果您还没有参数，请在 Parameter Store 中创建一个。有关信息，请参阅在 Systems Manager 中创建 Parameter Store 参数。

故障排除

您可以通过描述容器组（pod）部署来查看大多数错误。

查看容器的错误消息

1. 使用以下命令获取容器组（pod）名称列表。如果您没有使用默认命名空间，请使用 `-n` *`namespace`*。

```
kubectl get pods
```

2. 要描述容器组（pod），请在以下命令中为 *`pod-id`* 使用在上一步中找到的容器组（pod）的容器组（pod）ID。如果没有使用默认命名空间，请使用 `-n` *`NAMESPACE`*。

```
kubectl describe pod/pod-id
```

查看 ASCP 的错误

- 要在提供者日志中查找更多信息，请在以下命令中为 *`PODID`* 使用 csi-secrets-store-provider-aws 容器组（pod）的 ID。

```
kubectl -n kube-system get pods
kubectl -n kube-system logs pod/pod-id
```

将 AWS Secrets and Configuration Provider CSI 与服务账户的 IAM 角色（IRSA）结合使用

主题

- 先决条件
- 设置访问控制
- 确定要挂载的参数
- 故障排除

先决条件

- Amazon EKS 集群（版本 1.17 或更高版本）
- 通过 `kubectl` 访问 AWS CLI 和 Amazon EKS 集群

设置访问控制

ASCP 会检索 Amazon EKS 容器组身份并将其交换为 IAM 角色。您可以在 IAM 策略中为该 IAM 角色设置权限。当 ASCP 担任 IAM 角色时，它可以访问您授权的参数。除非将这些参数与 IAM 角色关联，否则其他容器无法访问它们。

授予 Amazon EKS 容器组（Pod）对 Parameter Store 中参数的访问权限

1. 创建权限策略，授予对容器组（pod）需要访问的参数的 `ssm:GetParameters` 和 `ssm:DescribeParameters` 权限。
2. 为集群创建 IAM OpenID Connect (OIDC) 提供商（如果还没有）。有关更多信息，请参阅《Amazon EKS 用户指南》中的为集群创建 IAM OIDC 提供商。
3. 为服务账户创建一个 IAM 角色并将策略附加到该角色。有关更多信息，请参阅《Amazon EKS 用户指南》中的为服务账户创建 IAM 角色。
4. 如果使用私有 Amazon EKS 集群，请确保该集群所在的 VPC 具有 AWS STS 端点。有关创建端点的信息，请参阅《AWS Identity and Access Management 用户指南》中的接口 VPC 端点。

确定要挂载的参数

要确定 ASCP 将哪些参数作为文件系统上的文件挂载到 Amazon EKS，需要创建 the section called “SecretProviderClass” YAML 文件。`SecretProviderClass` 列出了要挂载的参数以及要挂载这些参数的文件名。`SecretProviderClass` 必须与该文件引用的 Amazon EKS 容器组（pod）位于同一命名空间。

将参数作为文件进行挂载

以下说明展示了如何使用示例 YAML 文件 ExampleSecretProviderClass.yaml 和 ExampleDeployment.yaml 将参数作为文件进行挂载。

将参数挂载到 Amazon EKS 中

1. 将 `SecretProviderClass` 应用于容器组（pod）：

```
kubectl apply -f ExampleSecretProviderClass.yaml
```

2. 部署容器组（pod）：

```
kubectl apply -f ExampleDeployment.yaml
```

3. ASCP 会挂载文件。

故障排除

您可以通过描述容器组（pod）部署来查看大多数错误。

查看容器的错误消息

1. 使用以下命令获取容器组（pod）名称列表。如果您没有使用默认命名空间，请使用 `-n` *name-space*。

```
kubectl get pods
```

2. 要描述容器组（pod），请在以下命令中为 *pod-id* 使用在上一步中找到的容器组（pod）的容器组（pod）ID。如果没有使用默认命名空间，请使用 `-n` *nameSpace*。

```
kubectl describe pod/pod-id
```

查看 ASCP 的错误

- 要在提供者日志中查找更多信息，请在以下命令中为 *pod-id* 使用 csi-secrets-store-provider-aws 容器组（pod）的 ID。

```
kubectl -n kube-system get pods
kubectl -n kube-system logs Pod/pod-id
```

- 验证是否已安装 **SecretProviderClass** CRD：

```
kubectl get crd secretproviderclasses.secrets-store.csi.x-k8s.io
```

该命令应返回有关 SecretProviderClass 自定义资源定义的信息。

- 验证是否已创建 SecretProviderClass 对象。

```
kubectl get secretproviderclass SecretProviderClassName -o yaml
```

AWS Secrets and Configuration Provider 代码示例

ASCP 身份验证和访问控制示例

示例：允许 Amazon EKS 容器组身份服务（pods.eks.amazonaws.com）担任角色并标记会话的 IAM 策略：

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "pods.eks.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ],
      "Resource": "arn:aws:iam::111122223333:role/EKSPodIdentityRole"
    }
  ]
}
```

SecretProviderClass

您可以使用 YAML 描述要通过 ASCP 挂载到 Amazon EKS 中的参数。有关示例，请参阅 the section called “SecretProviderClass 用法”。

SecretProviderClass YAML 结构

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
   name: name
spec:
  provider: aws
  parameters:
    region:
```

```
    failoverRegion:
    pathTranslation:
    usePodIdentity:
    preferredAddressType:
    objects:
```

参数字段包含挂载请求的详细信息：

区域

（可选）参数的 AWS 区域。如果不使用此字段，ASCP 将从节点上的注释中查找 “区域”。查找会增加挂载请求的开销，因此我们建议为使用大量容器组（pod）的集群提供区域。

如果您还指定了 `failoverRegion`，ASCP 会尝试从两个区域中检索参数。如果任一区域返回 `4xx` 错误（例如身份验证问题），ASCP 将不会挂载任何一个参数。如果成功从 `region` 中检索到参数，则 ASCP 会挂载该参数值。如果未成功从 `region` 中检索到参数，但成功从 `failoverRegion` 中检索到参数，则 ASCP 会挂载该参数值。

`failoverRegion`

（可选）如果包含此字段，则 ASCP 会尝试从 `region` 中定义的区域和此字段中检索参数。如果任一区域返回 `4xx` 错误（例如身份验证问题），ASCP 将不会挂载任何一个参数。如果成功从 `region` 中检索到参数，则 ASCP 会挂载该参数值。如果未成功从 `region` 中检索到参数，但成功从 `failoverRegion` 中检索到参数，则 ASCP 会挂载该参数值。有关如何使用此字段的示例，请参阅 多区域参数失效转移。

pathTranslation

（可选）如果 Amazon EKS 中的文件名包含路径分隔符则要使用的单个替换字符，例如 Linux 上的斜杠 (/)。ASCP 无法创建包含路径分隔符的挂载文件。相反，ASCP 使用不同的字符替换路径分隔符。如果不使用此字段，替换字符为下划线 (_)，因此，例如 `My/Path/Parameter` 挂载为 `My_Path_Parameter`。

要防止字符替换，请输入字符串 `False`。

`usePodIdentity`

（可选）确定身份验证方法。如果未指定，则默认为服务账户（IRSA）的 IAM 角色。

- 要使用 EKS 容器组身份，请使用以下任意值：`"true"`、`"True"`、`"TRUE"`、`"t"` 或 `"T"`。
- 要明确使用 IRSA，请设置为以下任意值：`"false"`、`"False"`、`"FALSE"`、`"f"` 或 `"F"`。

preferredAddressType

（可选）指定容器组身份代理端点通信的首选 IP 地址类型。该字段仅在使用 EKS 容器组身份功能时适用，使用服务账户的 IAM 角色时将忽略。值不区分大小写。有效值为：

- "ipv4"、"IPv4" 或 "IPV4"：强制使用容器组身份代理 IPv4 端点
- "ipv6"、"IPv6" 或 "IPV6"：强制使用容器组身份代理 IPv6 端点
- 未指定：使用自动端点选择，首先尝试 IPv4 端点，如果 IPv4 失败则回退到 IPv6 端点

objects

包含要挂载密钥的 YAML 声明字符串。我们建议使用 YAML 多行字符串或竖线 (|) 字符。

objectName

必需。指定要获取的参数或密钥的名称。对于 Parameter Store，这是参数的 Name，可以是参数名称，也可以是完整 ARN。对于 Secrets Manager，这是 SecretId 参数，可以是密钥的友好名称，也可以是完整 ARN。

objectType

如果不将 Secrets Manager ARN 用于 objectName，需要这个操作 对于 Parameter Store，请使用 ssmparameter。对于 Secrets Manager，请使用 secretsmanager。

objectAlias

（可选）Amazon EKS 容器组（pod）中密钥的文件名。如果不指定此字段，则 objectName 作为文件名显示。

ObjectVersion

（可选）参数的版本 ID。不推荐，因为每次更新参数时都必须更新版本 ID。默认情况下，使用最新版本。如果包括 failoverRegion，则此字段表示主 objectVersion。

objectVersionLabel

（可选）版本的别名。默认为最新版本 AWSCURRENT。如果包括 failoverRegion，则此字段表示主 objectVersionLabel。

JMESPath

（可选）参数中的键映射到要挂载到 Amazon EKS 中的文件。要使用此字段，参数值必须采用 JSON 格式。

以下示例展示了 JSON 编码的参数。

```
{
    "username" : "myusername",
    "password" : "mypassword"
}
```

键是 username 和 password。与 username 关联的值是 myusername，与 password 关联的值是 mypassword。

如果使用此字段，必须包含子字段 path 和 objectAlias。

path

参数值 JSON 中的键/值对中的键。如果该字段包含连字符，请使用单引号对其进行转义，例如：path: '"hyphenated-path"'

objectAlias

要挂载到 Amazon EKS 容器组（pod）中的文件名。如果该字段包含连字符，请使用单引号对其进行转义，例如：objectAlias: '"hyphenated-alias"'

failoverObject

（可选）如果您指定此字段，ASCP 会尝试检索主 objectName 中指定的参数和 failoverObject objectName 子字段中指定的参数。如果任何一个参数返回 4xx 错误（例如身份验证问题），ASCP 将不会挂载任何一个参数。如果成功从主 objectName 中检索到参数，则 ASCP 会挂载该参数值。如果未成功从主 objectName 中检索到参数，但已成功从失效转移 objectName 中检索到参数，则 ASCP 会挂载该参数值。如果包含此字段，责必须包含字段 objectAlias。有关如何使用此字段的示例，请参阅 失效转移到其他参数。

当失效转移参数不是副本时，通常会使用此字段。有关如何指定副本的示例，请参阅 多区域参数失效转移。

objectName

失效转移参数的名称或完整 ARN。如果使用 ARN，则 ARN 中的区域必须与字段 failoverRegion 匹配。

ObjectVersion

（可选）参数的版本 ID。必须与主 objectVersion 匹配。不推荐，因为每次更新参数时都必须更新版本 ID。默认情况下，使用最新版本。

objectVersionLabel

（可选）版本的别名。默认为最新版本 AWSCURRENT。

创建基本的 SecretProviderClass 配置，以将参数挂载到 Amazon EKS 容器组（pod）中。

Pod Identity

SecretProviderClass 在同一 Amazon EKS 集群中使用的参数：

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: aws-parameter-store
spec:
  provider: aws
  parameters:
    objects: |
      - objectName: "MyParameter"
        objectType: "ssmparameter"
    usePodIdentity: "true"
```

IRSA

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: deployment-aws-parameter
spec:
  provider: aws
  parameters:
    objects: |
        - objectName: "MyParameter"
          objectType: "ssmparameter"
```

SecretProviderClass 用法

借助这些示例为不同场景创建 SecretProviderClass 配置。

示例：按名称或 ARN 挂载参数

此示例说明了如何挂载三种不同类型的参数：

- 由完整 ARN 指定的参数
- 由名称指定的参数
- 密钥的参数版本

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: aws-parameters
spec:
  provider: aws
  parameters:
    objects: |
      - objectName: "arn:aws:ssm:us-east-2:777788889999:parameter:MyParameter2-d4e5f6"
      - objectName: "MyParameter3"
        objectType: "ssmparameter"
      - objectName: "MyParameter4"
        objectType: "ssmparameter"
        objectVersionLabel: "AWSCURRENT"
```

示例：从参数挂载键/值对

此示例说明了如何从 JSON 格式的参数挂载特定的键/值对：

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: aws-parameters
spec:
  provider: aws
  parameters:
    objects: |
      - objectName: "arn:aws:ssm:us-east-2:777788889999:parameter:MyParameter-a1b2c3"
        jmesPath:
            - path: username
              objectAlias: dbusername
            - path: password
              objectAlias: dbpassword
```

示例：失效转移配置示例

此示例说明了如何为参数配置失效转移。

多区域参数失效转移

此示例说明了如何为跨多个区域复制的参数配置自动失效转移：

```
apiVersion: secrets-store.csi.x-k8s.io/v1
```

```
kind: SecretProviderClass
metadata:
  name: aws-parameters
spec:
  provider: aws
  parameters:
    region: us-east-1
    failoverRegion: us-east-2
    objects: |
      - objectName: "MyParameter"
```

失效转移到其他参数

此示例说明了如何将失效转移配置为其他参数（并非副本）：

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: aws-parameters
spec:
  provider: aws
  parameters:
    region: us-east-1
    failoverRegion: us-east-2
    objects: |
      - objectName: "arn:aws:ssm:us-east-1:777788889999:parameter:MyParameter-a1b2c3"
        objectAlias: "MyMountedParameter"
        failoverObject:
          - objectName: "arn:aws:ssm:us-
east-2:777788889999:parameter:MyFailoverParameter-d4e5f6"
```

其他资源

有关将 ASCP 与 Amazon EKS 结合使用的更多信息，请参阅以下资源：

- 将容器组身份与 Amazon EKS 结合使用
- GitHub 上的 AWS Secrets Store CSI 驱动程序

在 AWS Lambda 函数中使用 Parameter Store 参数

Parameter Store 是 AWS Systems Manager 中的一项工具，可提供安全的分层存储用于配置数据管理和密钥管理。您可以将密码、数据库字符串、Amazon Machine Image (AMI) ID 和许可证代码等数据存储为参数值。

若要在不使用 SDK 的情况下在 AWS Lambda 函数中使用 Parameter Store 中的参数，您可以使用 AWS 参数和密钥 Lambda 扩展。此扩展会检索参数值，并将其缓存以供将来使用。使用 Lambda 扩展可以通过减少对 Parameter Store 的 API 调用次数来降低成本。使用扩展还可以降低延迟，因为检索缓存参数比从 Parameter Store 中检索参数更快。

Lambda 扩展是一个配套进程，增加了 Lambda 函数的功能。扩展类似于与 Lambda 调用并行的客户端。此并行客户端可以在其生命周期中的任何时刻与您的函数交互。有关更多信息，请参阅《AWS Lambda 开发人员指南》中的Lambda 扩展 API。

AWS 参数和密钥 Lambda 扩展同时适用于 Parameter Store 和 AWS Secrets Manager。了解如何将 Lambda 扩展与 Secrets Manager 中的密钥一起使用，请参阅《AWS Secrets Manager 用户指南》中的在 AWS Lambda 函数中使用 AWS Secrets Manager 密钥。

相关信息

使用 AWS 参数和密钥 Lambda 扩展来缓存参数和密钥（AWS 计算博客）

扩展的工作原理

在不使用 Lambda 扩展的情况下，若要在 Lambda 函数中使用参数，您必须通过与 Parameter Store 的 `GetParameter` API 操作集成来配置您的 Lambda 函数，以接收配置更新。

使用 AWS 参数和密钥 Lambda 扩展时，扩展程序会从 Parameter Store 中检索参数值并将其存储在本地缓存中。然后，缓存值将用于进一步的调用，直到过期。缓存值会在超过其生存时间（TTL）后过期。您可以使用 `SSM_PARAMETER_STORE_TTL` 环境变量配置 TTL 值，如本主题下文所述。

如果配置的缓存 TTL 尚未到期，则使用缓存的参数值。如果已到期，则缓存的值将失效，并从 Parameter Store 中检索参数值。

此外，系统会检测常用的参数值并将其保存在缓存中，同时清除已过期或未使用的参数值。

> **Important**
>
> 该扩展只能在 Lambda 操作的 `INVOKE` 阶段调用，不能在 `INIT` 阶段调用。

实施详情

使用以下详细信息来帮助您配置 AWS 参数和密钥 Lambda 扩展。

身份验证

为了对 Parameter Store 请求进行授权和身份验证，扩展程序将使用与运行 Lambda 函数本身相同的凭证。因此，用于运行函数的 AWS Identity and Access Management（IAM）角色必须具有以下权限才能与 Parameter Store 交互：

- `ssm:GetParameter` – 从 Parameter Store 中检索参数时需要
- `kms:Decrypt` – 从 Parameter Store 中检索 `SecureString` 参数时需要

有关更多信息，请参阅《AWS Lambda 开发人员指南》中的 AWS Lambda 执行角色。

实例化

Lambda 会实例化与函数所需的并发级别相对应的单独实例。每个实例都是独立的，并维护自己的配置数据本地缓存。有关 Lambda 实例和并发的更多信息，请参阅《AWS Lambda 开发人员指南》中的配置预留并发。

无 SDK 依赖项

AWS 参数和密钥 Lambda 扩展程序独立于任何 AWS SDK 语言库工作。无需 AWS SDK 即可向 Parameter Store 发出 GET 请求。

Localhost 端口

在您的 GET 请求中使用 `localhost`。该扩展向 localhost 端口 2773 发出请求。无需指定外部或内部端点即可使用扩展。您可以通过设置环境变量 `PARAMETERS_SECRETS_EXTENSION_HTTP_PORT` 来配置端口。

例如，在 Python 中，您的 GET URL 可能与以下示例类似。

```
parameter_url = ('http://localhost:' + port + '/systemsmanager/parameters/get/?
name=' + ssm_parameter_path)
```

TTL 到期之前参数值的更改

扩展程序不会检测对参数值所做的更改，也不会在 TTL 到期之前执行自动刷新操作。如果您更改参数值，则在缓存下次刷新之前，使用缓存参数值的操作可能会失败。如果您预计参数值会频繁更改，我们建议设置较短的 TTL 值。

标头要求

若要从扩展缓存中检索参数，则 GET 请求的标头必须包含 X-Aws-Parameters-Secrets-Token 引用。将令牌设置为 AWS_SESSION_TOKEN，Lambda 为所有正在运行的函数提供此令牌。使用此标头表示调用方在 Lambda 环境中。

示例

以下 Python 示例演示了检索缓存参数值的基本请求。

```
import urllib.request
import os
import json

aws_session_token = os.environ.get('AWS_SESSION_TOKEN')

def lambda_handler(event, context):
    # Retrieve /my/parameter from Parameter Store using extension cache
    req = urllib.request.Request('http://localhost:2773/systemsmanager/parameters/
get?name=%2Fmy%2Fparameter')
    req.add_header('X-Aws-Parameters-Secrets-Token', aws_session_token)
    config = urllib.request.urlopen(req).read()

    return json.loads(config)
```

ARM 支持

在支持 x86_64 和 x86 架构的大多数 AWS 区域中，该扩展不支持 ARM 架构。如果您使用的是 ARM 架构，我们建议您验证您的架构是否受支持。有关扩展 ARN 的完整列表，请参阅 AWS 参数和密钥 Lambda 扩展 ARN。

日志记录

Lambda 使用 Amazon CloudWatch Logs 记录有关扩展的执行信息以及函数。默认情况下，扩展将最少量的信息记录到 CloudWatch。若要记录更多详细信息，请将环境变量 PARAMETERS_SECRETS_EXTENSION_LOG_LEVEL 设置为 DEBUG。

将扩展添加到 Lambda 函数

若要使用 AWS 参数和密钥 Lambda 扩展，您需要将扩展作为层添加到 Lambda 函数。

使用以下方法之一将扩展添加到函数。

AWS Management Console（添加层选项）

1. 通过 https://console.aws.amazon.com/lambda/ 打开 AWS Lambda 控制台。
2. 选择您的函数。在 Layers（层）区域中，选择 Add a layer（添加层）。
3. 在选择层区域中，选择 AWS 层选项。
4. 在 AWS 层中，选择 AWS-Parameters-and-Secrets-Lambda-Extension，选择版本，然后选择添加。

AWS Management Console（指定 ARN 选项）

1. 通过 https://console.aws.amazon.com/lambda/ 打开 AWS Lambda 控制台。
2. 选择您的函数。在 Layers（层）区域中，选择 Add a layer（添加层）。
3. 在 Choose a layer（选择层）区域中，选择 Specify an ARN（指定 ARN）选项。
4. 对于 Specify an ARN（指定 ARN），输入 AWS 区域 和架构的扩展 ARN，然后选择 Add（添加）。

AWS Command Line Interface

在 AWS CLI 中运行以下命令。将每个*#######*替换为您自己的信息。

```
aws lambda update-function-configuration \
    --function-name function-name \
    --layers layer-ARN
```

相关信息

将层与 Lambda 函数结合使用

配置扩展（.zip 文件存档）

AWS 参数和密钥 Lambda 扩展环境变量

您可以通过更改以下环境变量配置扩展。若要查看当前设置，请将 `PARAMETERS_SECRETS_EXTENSION_LOG_LEVEL` 设置为 `DEBUG`。有关更多信息，请参阅《AWS Lambda 开发者指南》中的使用 AWS Lambda 环境变量。

> **Note**
>
> AWS Lambda 将关于 Lambda 扩展和 Lambda 函数的操作详细信息记录在 Amazon CloudWatch Logs 中。

环境变量	详细信息	必需	有效值	默认值
SSM_PARAMETER_STORE_TIMEOUT_MILLIS	对 Parameter Store 的请求超时（以毫秒为单位）。 值为 0（零）表示没有超时。	否	所有整数	0（零）
SECRETS_MANAGER_TIMEOUT_MILLIS	对 Secrets Manager 的请求超时（以毫秒为单位）。 值为 0（零）表示没有超时。	否	所有整数	0（零）
SSM_PARAMETER_STORE_TTL	缓存中参数在失效前的最长有效生命周期（以秒为单位）。值为 0（零）表示应绕过缓存。如果 PARAMETERS_SECRETS_EXTENSION_CACHE_SIZE 的值为 0（零），则忽略此变量。	否	0（零）到 300 秒（五分钟）	300 秒（五分钟）
SECRETS_MANAGER_TTL	缓存中密钥在失效前的最长有效生命周期（以秒为单位）。值	否	0（零）到 300 秒（五分钟）	300 秒（5 分钟）

环境变量	详细信息	必需	有效值	默认值
	为 0（零）表示已绕过缓存。如果 PARAMETERS_SECRETS_EXTENSION_CACHE_SIZE 的值为 0（零），则忽略此变量。			
PARAMETERS_SECRETS_EXTENSION_CACHE_ENABLED	确定是否已启用扩展缓存。有效值：TRUE | FALSE	否	TRUE | FALSE	TRUE
PARAMETERS_SECRETS_EXTENSION_CACHE_SIZE	缓存的最大大小（以项目数为单位）。值为 0（零）表示已绕过缓存。如果两个缓存 TTL 值都为 0（零），则忽略此变量。	否	0（零）到 1000	1000
PARAMETERS_SECRETS_EXTENSION_HTTP_PORT	本地 HTTP 服务器的端口。	否	1-65535	2773

环境变量	详细信息	必需	有效值	默认值
PARAMETERS_SECRETS_EXTENSION_MAX_CONNECTIONS	扩展用于向 Parameter Store 或 Secrets Manager 发出请求的 HTTP 客户端的最大连接数。这是 Secrets Manager 客户端和 Parameter Store 客户端与后端服务建立的连接数的各客户端配置。	否	最小值为 1；无最大值限制。	3
PARAMETERS_SECRETS_EXTENSION_LOG_LEVEL	扩展日志中报告的详细信息级别。 我们建议您在设置和测试扩展时使用 DEBUG 以获取有关缓存配置最详细的信息。 Lambda 操作的日志会自动推送到关联的 Logs 日志组。	否	DEBUG | WARN | ERROR | NONE | INFO	INFO

使用 AWS Systems ManagerParameter Store 和 AWS Secrets Manager 扩展的示例命令

本节中的示例演示了与 AWS Systems Manager Parameter Store 和 AWS Secrets Manager 扩展搭配使用的 API 操作。

Parameter Store 的示例命令

Lambda 扩展对 GetParameter API 操作使用只读访问权限。

若要调用此操作，请进行类似于以下示例的 HTTP GET 调用。此命令格式提供对标准参数层中参数的访问权限。

```
GET http://localhost:port/systemsmanager/parameters/get?name=parameter-name&version=version&label=label&withDecryption={true|false}
```

在此示例中，，*parameter-name* 表示完整的参数名称，例如 MyParameter（对于不在层次结构中的参数）或 %2FDev%2FProduction%2FEast%2FProject-ABC%2FMyParameter（对于作为层次结构一部分的名为 /Dev/Production/East/Project-ABC/MyParameter 的参数）。

> **Note**
>
> 使用 GET 调用时，必须针对 HTTP 对参数值进行编码，以保留特殊字符。例如，对可转换为 URL 一部分的字符进行编码（如 %2Fa%2Fb%2Fc），而不要将层次结构路径进行格式化（如 /a/b/c）。

##和##是可用于 GetParameter 操作的选择器。

```
GET http://localhost:port/systemsmanager/parameters/get/?name=MyParameter&version=5
```

若要调用层次结构中的参数，请进行类似于以下示例的 HTTP GET 调用。

```
GET http://localhost:port/systemsmanager/parameters/get?name=%2Fa%2Fb%2F&label=release
```

若要调用公用（全局）参数，请进行类似于以下示例的 HTTP GET 调用。

```
GET http://localhost:port/systemsmanager/parameters/get/?name=%2Faws%2Fservice%20list%2F…
```

若要使用 Parameter Store 引用对 Secrets Manager 密钥进行 HTTP GET 调用，请进行类似于以下示例的 HTTP GET 调用。

```
GET http://localhost:port/systemsmanager/parameters/get?name=%2Faws%2Freference%2Fsecretsmanager%2F…
```

若要使用参数的 Amazon 资源名称（ARN）进行调用，请进行类似于以下示例的 HTTP GET 调用。

```
GET http://localhost:port/systemsmanager/parameters/get?name=arn:aws:ssm:us-
east-1:123456789012:parameter/MyParameter
```

若要进行的调用将访问已解密的 SecureString 参数，请进行类似于以下示例的 HTTP GET 调用。

```
GET http://localhost:port/systemsmanager/parameters/get?
name=MyParameter&withDecryption=true
```

您可以通过省略 withDecryption 或将其明确设置为 false 来指定未解密的参数。您还可以指定版本或标签，但不能同时指定两者。如果同时指定版本和标签，则仅使用置于 URL 中问号（？）之后的第一个字符。

AWS 参数和密钥 Lambda 扩展 ARN

下表提供了支持架构和区域的扩展 ARN。

主题

- x86_64 和 x86 架构的扩展 ARN
- ARM64 和 Mac with Apple silicon 架构的扩展 ARN

x86_64 和 x86 架构的扩展 ARN

上次更新时间：2025 年 6 月 26 日

区域	ARN
美国东部（俄亥俄州）	arn:aws:lambda:us-east-2:590474943231:layer:AWS-Parameters-and-Secrets-Lambda-Extension:22
美国东部（弗吉尼亚州北部）	arn:aws:lambda:us-east-1:177933569100:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18

区域	ARN
美国西部（加利福尼亚北部）	arn:aws:lambda:us-west-1:997803712105:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
美国西部（俄勒冈州）	arn:aws:lambda:us-west-2:345057560386:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
非洲（开普敦）	arn:aws:lambda:af-south-1:317013901791:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
亚太地区（香港）	arn:aws:lambda:ap-east-1:768336418462:layer:AWS-Parameters-and-Secrets-Lambda-Extension:20
亚太地区（台北）	arn:aws:lambda:ap-east-2:890742577149:layer:AWS-Parameters-and-Secrets-Lambda-Extension:6
亚太地区（海得拉巴）区域	arn:aws:lambda:ap-south-2:070087711984:layer:AWS-Parameters-and-Secrets-Lambda-Extension:15
亚太地区（雅加达）	arn:aws:lambda:ap-southeast-3:490737872127:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18

区域	ARN
亚太地区（墨尔本）	arn:aws:lambda:ap-southeast-4:090732460067:layer:AWS-Parameters-and-Secrets-Lambda-Extension:8
亚太地区（马来西亚）	arn:aws:lambda:ap-southeast-5:381492012281:layer:AWS-Parameters-and-Secrets-Lambda-Extension:7
亚太地区（孟买）	arn:aws:lambda:ap-south-1:176022468876:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
亚太地区（大阪）	arn:aws:lambda:ap-northeast-3:576959938190:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
亚太地区（首尔）	arn:aws:lambda:ap-northeast-2:738900069198:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
亚太地区（新加坡）	arn:aws:lambda:ap-southeast-1:044395824272:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
亚太地区（悉尼）	arn:aws:lambda:ap-southeast-2:665172237481:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18

区域	ARN
亚太地区（泰国）	arn:aws:lambda:ap-southeast-7:941377119484:layer:AWS-Parameters-and-Secrets-Lambda-Extension:6
亚太地区（东京）	arn:aws:lambda:ap-northeast-1:133490724326:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
加拿大（中部）	arn:aws:lambda:ca-central-1:200266452380:layer:AWS-Parameters-and-Secrets-Lambda-Extension:20
加拿大西部（卡尔加里）	arn:aws:lambda:ca-west-1:243964427225:layer:AWS-Parameters-and-Secrets-Lambda-Extension:8
中国（北京）	arn:aws-cn:lambda:cn-north-1:287114880934:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18arn:aws-cn:lambda:cn-north-1:287114880934:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
中国（宁夏）	arn:aws-cn:lambda:cn-northwest-1:287310001119:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18arn:aws-cn:lambda:cn-northwest-1:287310001119:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18

区域	ARN
欧洲地区（法兰克福）	arn:aws:lambda:eu-central-1:187925254637:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
欧洲地区（爱尔兰）	arn:aws:lambda:eu-west-1:015030872274:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
欧洲地区（伦敦）	arn:aws:lambda:eu-west-2:133256977650:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
欧洲地区（米兰）	arn:aws:lambda:eu-south-1:325218067255:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
欧洲地区（巴黎）	arn:aws:lambda:eu-west-3:780235371811:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
欧洲地区（西班牙）区域	arn:aws:lambda:eu-south-2:524103009944:layer:AWS-Parameters-and-Secrets-Lambda-Extension:17
欧洲地区（斯德哥尔摩）	arn:aws:lambda:eu-north-1:427196147048:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18

区域	ARN
以色列（特拉维夫）	arn:aws:lambda:il-central-1:148806536434:layer:AWS-Parameters-and-Secrets-Lambda-Extension:8
欧洲（苏黎世）	arn:aws:lambda:eu-central-2:772501565639:layer:AWS-Parameters-and-Secrets-Lambda-Extension:15
墨西哥（中部）区域	arn:aws:lambda:mx-central-1:241533131596:layer:AWS-Parameters-and-Secrets-Lambda-Extension:5
中东（巴林）	arn:aws:lambda:me-south-1:832021897121:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
中东（阿联酋）	arn:aws:lambda:me-central-1:858974508948:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
南美洲（圣保罗）	arn:aws:lambda:sa-east-1:933737806257:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18

区域	ARN
AWS GovCloud（美国东部）	arn:aws-us-gov:lambda:us-gov-east-1:129776340158:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18arn:aws-us-gov:lambda:us-gov-east-1:129776340158:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
AWS GovCloud（美国西部）	arn:aws-us-gov:lambda:us-gov-west-1:127562683043:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18arn:aws-us-gov:lambda:us-gov-west-1:127562683043:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18

ARM64 和 Mac with Apple silicon 架构的扩展 ARN

上次更新时间：2025 年 6 月 26 日

区域	ARN
美国东部（俄亥俄州）	arn:aws:lambda:us-east-2:590474943231:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:22
美国东部（弗吉尼亚州北部）	arn:aws:lambda:us-east-1:177933569100:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:18
美国西部（北加利福尼亚）区域	arn:aws:lambda:us-west-1:997803712105:layer:AWS-Parame

区域	ARN
	ters-and-Secrets-Lambda-Extension-Arm64:15
美国西部（俄勒冈州）	arn:aws:lambda:us-west-2:345057560386:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:18
非洲（开普敦）区域	arn:aws:lambda:af-south-1:317013901791:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:15
亚太地区（香港）区域	arn:aws:lambda:ap-east-1:768336418462:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:17
亚太地区（台北）	arn:aws:lambda:ap-east-2:890742577149:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:2
亚太地区（海得拉巴）区域	arn:aws:lambda:ap-south-2:070087711984:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7
亚太地区（雅加达）区域	arn:aws:lambda:ap-southeast-3:490737872127:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:15

区域	ARN
亚太地区（墨尔本）	arn:aws:lambda:ap-southeast-4:090732460067:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7
亚太地区（马来西亚）	arn:aws:lambda:ap-southeast-5:381492012281:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7
亚太地区（孟买）	arn:aws:lambda:ap-south-1:176022468876:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:18
亚太地区（大阪）	arn:aws:lambda:ap-northeast-3:576959938190:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:15
亚太地区（首尔）区域	arn:aws:lambda:ap-northeast-2:738900069198:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:15
亚太地区（新加坡）	arn:aws:lambda:ap-southeast-1:044395824272:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:18
亚太地区（悉尼）	arn:aws:lambda:ap-southeast-2:665172237481:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:18

区域	ARN
亚太地区（泰国）	arn:aws:lambda:ap-southeast-7:941377119484:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:5
亚太地区（东京）	arn:aws:lambda:ap-northeast-1:133490724326:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:18
加拿大（中部）区域	arn:aws:lambda:ca-central-1:200266452380:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:17
加拿大西部（卡尔加里）	arn:aws:lambda:ca-west-1:243964427225:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7
中国（北京）	arn:aws-cn:lambda:cn-north-1:287114880934:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7arn:aws-cn:lambda:cn-north-1:287114880934:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7

区域	ARN
中国（宁夏）	arn:aws-cn:lambda:cn-northwest-1:287310001119:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7arn:aws-cn:lambda:cn-northwest-1:287310001119:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7
欧洲地区（法兰克福）	arn:aws:lambda:eu-central-1:187925254637:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:18
欧洲地区（爱尔兰）	arn:aws:lambda:eu-west-1:015030872274:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:18
欧洲地区（伦敦）	arn:aws:lambda:eu-west-2:133256977650:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:18
欧洲地区（米兰）	arn:aws:lambda:eu-south-1:325218067255:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:15
欧洲地区（巴黎）区域	arn:aws:lambda:eu-west-3:780235371811:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:15

区域	ARN
欧洲地区（西班牙）区域	arn:aws:lambda:eu-south-2:524103009944:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:9
欧洲地区（斯德哥尔摩）区域	arn:aws:lambda:eu-north-1:427196147048:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:15
以色列（特拉维夫）	arn:aws:lambda:il-central-1:148806536434:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7
欧洲（苏黎世）	arn:aws:lambda:eu-central-2:772501565639:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7
墨西哥（中部）区域	arn:aws:lambda:mx-central-1:241533131596:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:4
中东（巴林）区域	arn:aws:lambda:me-south-1:832021897121:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:15
中东（阿联酋）	arn:aws:lambda:me-central-1:858974508948:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7

区域	ARN
南美洲（圣保罗）区域	`arn:aws:lambda:sa-east-1:933737806257:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:15`
AWS GovCloud（美国东部）	`arn:aws-us-gov:lambda:us-gov-east-1:129776340158:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7arn:aws-us-gov:lambda:us-gov-east-1:129776340158:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7`
AWS GovCloud（美国西部）	`arn:aws-us-gov:lambda:us-gov-west-1:127562683043:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7arn:aws-us-gov:lambda:us-gov-west-1:127562683043:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7`

与其他产品和服务的集成

AWS Systems Manager 内置集成了下表中所示的产品和服务。

Ansible	Ansible 是一个 IT 自动化平台，可使您的应用程序和系统更易于部署。 Systems Manager 提供 Systems Manager 文档（SSM 文档）`AWS-ApplyAnsiblePlaybooks` ，让您可以创建运行 Ansible Playbook 的 State Manager 关联。

	了解更多 创建运行 Ansible Playbook 的关联
Chef	Chef 是一种 IT 自动化工具，可使您的应用程序和系统更易于部署。 Systems Manager 提供 `AWS-ApplyChefRecipes` SSM 文档，让您可以在运行 Chef 配方的 State Manager（AWS Systems Manager 中的一项工具）中创建关联。 了解更多 创建运行 Chef 配方的关联 Systems Manager 还与 Chef InSpec 配置文件集成，让您可以运行合规性扫描，并查看合规和不合规的节点。 了解更多 将 Chef InSpec 配置文件与 Systems Manager Compliance 结合使用
GitHub	GitHub 可为软件开发版本控制和协作提供托管。 Systems Manager 提供 SSM 文档 `AWS-RunDocument`，让您可以运行存储在 GitHub 中的其他 SSM 文档；其也提供 SSM 文档 `AWS-RunRemoteScript`，让您可以运行存储在 GitHub 中的脚本。 了解更多 • 从远程位置运行文档 • 从 GitHub 运行脚本

Jenkins	Jenkins 是一款开源自动化服务器，让开发人员可以可靠地构建、测试和部署软件。 Automation（Systems Manager 中的一项工具）可以用作构建后步骤，以便将应用程序版本预安装到Amazon Machine Images（AMIs）中。 了解更多 使用 Automation 和 Jenkins 更新 AMIs
ServiceNow	ServiceNow 是一个企业服务管理系统，让您可以管理 IT 服务和运维。 Systems Manager 中的所有工具（包括 Automation、Change Manager、Incident Manager 和 OpsCenter）都可使用 AWS 服务管理连接器与 ServiceNow 进行集成。通过此集成，您可以查看、创建、更新、添加通信和解决 ServiceNow 中的 AWS 支持 问题。 了解更多 集成 ServiceNow

Terraform	HashiCorp Terraform 是一款开源基础设施即代码 (IaC) 软件工具，它提供命令行界面 (CLI) 工作流程来管理各种云服务。对于 Systems Manager，您可以使用 Terraform 来管理或提供以下内容： 资源 • aws_ssm_activation • aws_ssm_association • aws_ssm_default_patch_baseline • aws_ssm_document • aws_ssm_maintenance_window • aws_ssm_maintenance_window_target • aws_ssm_maintenance_window_task • aws_ssm_parameter • aws_ssm_patch_baseline • aws_ssm_patch_group • aws_ssm_resource_data_sync • aws_ssm_service_setting 数据源 • aws_ssm_document • aws_ssm_instances • ssm_maintenance_windows • aws_ssm_parameter • aws_ssm_parameters_by_path • aws_ssm_patch_baseline

主题

- 从 GitHub 运行脚本
- 将 Chef InSpec 配置文件与 Systems Manager Compliance 结合使用

- 集成 ServiceNow

从 GitHub 运行脚本

本节介绍如何使用预定义 Systems Manager 文档（SSM 文档）`AWS-RunRemoteScript` 从 GitHub 下载脚本，包括 Ansible Playbook、Python、Ruby 和 PowerShell 脚本。通过使用此 SSM 文档，您不再需要将脚本手动移植到 Amazon Elastic Compute Cloud（Amazon EC2）中，也不需要将这些脚本封装在 SSM 文档中。AWS Systems Manager 与 GitHub 的集成可以促进实现基础设施即代码，这既可缩短管理节点所需的时间，又可在整个实例集中实现标准化配置。

您还可以创建自定义 SSM 文档，这使您可以从远程位置下载并运行脚本或其他 SSM 文档。有关更多信息，请参阅 创建复合文档。

还可以下载包括多个脚本的目录。在运行目录中的主脚本时，Systems Manager 还会运行该目录中包含的任何引用的脚本。

请注意关于从 GitHub 运行脚本的下列重要详细信息。

- Systems Manager 不会验证您的脚本是否能够在节点上运行。在下载和运行脚本之前，请确认相应节点上已经安装所需软件。您也可以创建一个复合文档，能够使用 Run Command 或State Manager（都是 AWS Systems Manager 中的工具）安装软件，然后再下载并运行脚本。
- 您应负责确保满足所有 GitHub 要求。包括根据需要刷新访问令牌。确保不要超过已进行身份验证或未进行身份验证的请求的数量。有关更多信息，请参阅GitHub 文档。
- 不支持 GitHub Enterprise 存储库。

主题

- 从 GitHub 运行 Ansible Playbook
- 从 GitHub 运行 Python 脚本

从 GitHub 运行 Ansible Playbook

本节包含帮助您使用控制台或 AWS Command Line Interface（AWS CLI）从 GitHub 运行 Ansible Playbook 的过程。

开始前的准备工作

如果您计划运行私有 GitHub 存储库中存储的脚本，请为 GitHub 安全访问令牌创建 AWS Systems Manager `SecureString` 参数。通过 SSH 手动传递令牌无法访问私有 GitHub 存储库中的脚本。该访

问令牌必须作为 Systems Manager SecureString 参数传递。有关创建 SecureString 参数的更多信息，请参阅 在 Systems Manager 中创建 Parameter Store 参数。

从 GitHub 运行 Ansible Playbook（控制台）

从 GitHub 运行 Ansible Playbook

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Run Command。
3. 选择 Run command（运行命令）。
4. 在 Command document (命令文档) 列表中，选择 **AWS-RunRemoteScript**。
5. 在命令参数中，执行以下操作：
   - 在源类型中，选择 GitHub。
   - 在 Source Info (源信息) 框中，按以下格式输入访问源所需的信息。

```
{
  "owner": "owner_name",
  "repository": "repository_name",
  "getOptions": "branch:branch_name",
  "path": "path_to_scripts_or_directory",
  "tokenInfo": "{{ssm-secure:SecureString_parameter_name}}"
}
```

此示例下载名为 webserver.yml 的文件。

```
{
    "owner": "TestUser1",
    "repository": "GitHubPrivateTest",
    "getOptions": "branch:myBranch",
    "path": "scripts/webserver.yml",
    "tokenInfo": "{{ssm-secure:mySecureStringParameter}}"
}
```

> **Note**
>
> 仅当您的 SSM 文档存储在 master 以外的分支中时， "branch" 才是必需的。

要使用存储库中特定提交 中的脚本版本，请使用 `commitID` 和 `getOptions`，而不是 `branch`。例如：
`"getOptions": "commitID:bbc1ddb94...b76d3bEXAMPLE",`

- 在 Command Line (命令行) 字段中，输入用于脚本执行的参数。见下列。

  `ansible-playbook -i “localhost,” --check -c local webserver.yml`

- （可选）在 Working Directory（工作目录）字段中，输入节点（要在其中下载和运行脚本）上的目录的名称。
- (可选) 在执行超时中，指定脚本命令执行失败之前系统要等待的秒数。

6. 在 Targets（目标）部分中，通过指定标签、手动选择实例或边缘设备或指定资源组，选择要在其上运行此操作的托管式节点。

> **Tip**
>
> 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

7. 对于 Other parameters（其他参数）：
   - 对于 Comment（注释），请输入有关此命令的信息。
   - 对于 Timeout (seconds) (超时 (秒))，请指定在整个命令执行失败之前系统等待的秒数。
8. 对于 Rate control（速率控制）：
   - 对于 Concurrency（并发），请指定要同时运行该命令的托管式节点的数量或百分比。

> **Note**
>
> 如果您通过指定应用于托管式节点的标签或指定 AWS Resource Groups 来选择目标，但不确定有多少个托管式节点已被设为目标，则可通过指定百分比来限制可同时运行该文档的目标的数量。

   - 对于 Error threshold（错误阈值），请指定当命令在一定数量或百分比的节点上失败后，何时在其他托管式节点上停止运行该命令。例如，如果您指定三个错误，Systems Manager 将在收到第四个错误时停止发送该命令。仍在处理该命令的托管式节点也可能发送错误。
9. （可选）对于 输出选项，要将命令输出保存到文件，请选中 将命令输出写入 S3 存储桶 框。在输入框中输入存储桶和前缀（文件夹）名称。

> **Note**
>
> 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给实例的实例配置文件（适用于 EC2 实例）或 IAM 服务角色（混合激活的计算机）的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确保与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

10. 在 SNS 通知部分，如果需要发送有关命令执行状态的通知，请选中 Enable SNS notifications（启用 SNS 通知）复选框。

    有关为 Run Command 配置 Amazon SNS 通知的更多信息，请参阅 使用 Amazon SNS 通知监控 Systems Manager 状态更改。

11. 选择 Run（运行）。

使用 AWS CLI 从 GitHub 运行 Ansible Playbook

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关更多信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 运行以下命令可从 GitHub 下载并运行脚本。

```
aws ssm send-command \
    --document-name "AWS-RunRemoteScript" \
    --instance-ids "instance-IDs"\
    --parameters '{"sourceType":["GitHub"],"sourceInfo":["{\"owner
\":\"owner_name\", \"repository\": \"repository_name\", \"path
\": \"path_to_file_or_directory\", \"tokenInfo\":\"{{ssm-
secure:name_of_your_SecureString_parameter}}\" }"],"commandLine":
["commands_to_run"]}'
```

   以下是一个要在本地 Linux 计算机上运行的示例命令。

```
aws ssm send-command \
    --document-name "AWS-RunRemoteScript" \
    --instance-ids "i-02573cafcfEXAMPLE" \
    --parameters '{"sourceType":["GitHub"],"sourceInfo":["{\"owner\":\"TestUser1\",
 \"repository\": \"GitHubPrivateTest\", \"path\": \"scripts/webserver.yml\",
```

```
 \"tokenInfo\":\"{{ssm-secure:mySecureStringParameter}}\" }"],"commandLine":
["ansible-playbook -i “localhost,” --check -c local webserver.yml"]}'
```

从 GitHub 运行 Python 脚本

本节包含帮助您使用 AWS Systems Manager 控制台或 AWS Command Line Interface（AWS CLI）从 GitHub 运行 Python 脚本的过程。

从 GitHub 运行 Python 脚本（控制台）

从 GitHub 运行 Python 脚本

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 Run Command。
3. 选择 Run command（运行命令）。
4. 在 Command document (命令文档) 列表中，选择 **AWS-RunRemoteScript**。
5. 对于命令参数，执行以下操作：
   - 在源类型中，选择 GitHub。
   - 在 Source Info (源信息) 框中，按以下格式输入访问源所需的信息：

```
{
    "owner": "owner_name",
    "repository": "repository_name",
    "getOptions": "branch:branch_name",
    "path": "path_to_document",
    "tokenInfo": "{{ssm-secure:SecureString_parameter_name}}"
}
```

以下示例会下载一个名为 complex-script 的脚本目录。

```
{
    "owner": "TestUser1",
    "repository": "SSMTestDocsRepo",
    "getOptions": "branch:myBranch",
    "path": "scripts/python/complex-script",
    "tokenInfo": "{{ssm-secure:myAccessTokenParam}}"
```

```
}
```

> **Note**
>
> 仅当您的脚本存储在 master 以外的分支中时，"branch" 才是必需的。
> 要使用存储库中特定提交 中的脚本版本，请使用 commitID 和 getOptions，而不是 branch。例如：
> "getOptions": "commitID:bbc1ddb94...b76d3bEXAMPLE",

- 对于 Command Line (命令行)，输入脚本执行的参数。见下列。

```
mainFile.py argument-1 argument-2
```

此示例运行 mainFile.py，它可在随后运行 complex-script 目录中的其他脚本。

- （可选）对于 Working Directory（工作目录），输入节点上要在其中下载和运行脚本的目录的名称。
- （可选）对于 Execution Timeout（执行超时），指定脚本命令执行失败之前系统要等待的秒数。

6. 在 Targets（目标）部分中，通过指定标签、手动选择实例或边缘设备或指定资源组，选择要在其上运行此操作的托管式节点。

> **Tip**
>
> 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

7. 对于 Other parameters（其他参数）：
   - 对于 Comment（注释），请输入有关此命令的信息。
   - 对于 Timeout (seconds) (超时 (秒))，请指定在整个命令执行失败之前系统等待的秒数。
8. 对于 Rate control（速率控制）：
   - 对于 Concurrency（并发），请指定要同时运行该命令的托管式节点的数量或百分比。

> **Note**
>
> 如果您通过指定应用于托管式节点的标签或指定 AWS Resource Groups 来选择目标，但不确定有多少个托管式节点已被设为目标，则可通过指定百分比来限制可同时运行该文档的目标的数量。

- 对于 Error threshold（错误阈值），请指定当命令在一定数量或百分比的节点上失败后，何时在其他托管式节点上停止运行该命令。例如，如果您指定三个错误，Systems Manager 将在收到第四个错误时停止发送该命令。仍在处理该命令的托管式节点也可能发送错误。

9. （可选）对于 输出选项，要将命令输出保存到文件，请选中 将命令输出写入 S3 存储桶 框。在输入框中输入存储桶和前缀（文件夹）名称。

> **Note**
>
> 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给实例的实例配置文件（适用于 EC2 实例）或 IAM 服务角色（混合激活的计算机）的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确保与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

10. 在 SNS 通知部分，如果需要发送有关命令执行状态的通知，请选中 Enable SNS notifications（启用 SNS 通知）复选框。

    有关为 Run Command 配置 Amazon SNS 通知的更多信息，请参阅 使用 Amazon SNS 通知监控 Systems Manager 状态更改。

11. 选择 Run（运行）。

使用 AWS CLI 从 GitHub 运行 Python 脚本

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关更多信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 运行以下命令可从 GitHub 下载并运行脚本。

```
aws ssm send-command --document-name "AWS-RunRemoteScript" --instance-
ids "instance-IDs" --parameters '{"sourceType":["GitHub"],"sourceInfo":
```

```
["{\"owner\":\"owner_name\", \"repository\":\"repository_name\", \"path\":
 \"path_to_script_or_directory\"}"],"commandLine":["commands_to_run"]}'
```

见下列。

```
aws ssm send-command --document-name "AWS-RunRemoteScript" --instance-ids
 "i-02573cafcfEXAMPLE" --parameters '{"sourceType":["GitHub"],"sourceInfo":
["{\"owner\":\"TestUser1\", \"repository\":\"GitHubTestPublic\", \"path\":
 \"scripts/python/complex-script\"}"],"commandLine":["mainFile.py argument-1
 argument-2 "]}'
```

此示例将下载名为 `complex-script` 的脚本目录。`commandLine` 条目运行 `mainFile.py`，它可在随后运行 `complex-script` 目录中的其他脚本。

将 Chef InSpec 配置文件与 Systems Manager Compliance 结合使用

AWS Systems Manager 与 Chef InSpec 集成。Chef InSpec 是一个开源的测试框架，使您可以创建要存储在 GitHub 或 Amazon Simple Storage Service（Amazon S3）中的人类可读配置文件。然后，您可以使用 Systems Manager 运行合规性扫描，并查看合规和不合规的节点。配置文件列出您的计算环境的安全性、合规性或策略要求。例如，您可以使用 Compliance（AWS Systems Manager 中的一项工具），创建将在您扫描节点时执行以下检查的配置文件：

- 检查特定端口是否处于打开或关闭状态。
- 检查特定应用程序是否正在运行。
- 检查某些软件包是否已安装。
- 检查 Windows 注册表项中的特定属性。

您可以仅为使用 Systems Manager 管理的 Amazon Elastic Compute Cloud (Amazon EC2) 实例创建 InSpec 配置文件。不支持本地服务器和虚拟机 (VM)。以下示例 Chef InSpec 配置文件将检查端口 22 是否处于打开状态。

```
control 'Scan Port' do
impact 10.0
title 'Server: Configure the service port'
desc 'Always specify which port the SSH server should listen to.
Prevent unexpected settings.'
describe sshd_config do
its('Port') { should eq('22') }
```

```
end
end
```

InSpec 包含一组资源，可帮助您快速编写检查和审计控制。InSpec 使用 InSpec 域特定语言 (DSL) 在 Ruby 中编写这些控件。您还可以使用由大型 InSpec 用户社区创建的配置文件。例如，GitHub 上的 DevSec chef-os-hardening 项目包含许多配置文件，可帮助保护节点。您可以在 GitHub 或 Amazon S3 中创作和存储配置文件。

工作方式

下面演示了 InSpec 配置文件与 Compliance 结合使用的过程：

1. 要么识别要使用的预定义 InSpec 配置文件，要么创建您自己的配置文件。您可以使用 GitHub 上的预定义配置文件来开始操作。有关如何创建自己的 InSpec 配置文件的信息，请参阅 ChefChef InSpec 配置文件。
2. 将配置文件存储在公有或私有 GitHub 存储库或者 S3 存储桶中。
3. 使用 Systems Manager 文档（SSM 文档）`AWS-RunInspecChecks`，借助您的 InSpec 配置文件运行 Compliance。您可以使用 Run Command（AWS Systems Manager 中的一项工具）开始按需进行合规性扫描，也可以使用State Manager（AWS Systems Manager 中的一项工具）按计划定期进行合规性扫描。
4. 使用 Compliance API 或 Compliance 控制台识别不合规的节点。

> **Note**
>
> 请注意以下信息。
>
> - Chef 使用节点上的客户端来处理配置文件。您不需要安装客户端。当 Systems Manager 运行 SSM 文档 `AWS-RunInspecChecks` 时，系统将检查是否已安装客户端。如果未安装，Systems Manager 会在扫描期间安装 Chef 客户端，然后在扫描完成后卸载该客户端。
> - 运行 SSM 文档 `AWS-RunInspecChecks`（如本主题中所述）会将类型为 `Custom:Inspec` 的合规性条目分配到每个目标节点。要分配此合规性类型，该文档将调用 PutComplianceItems API 操作。

运行 InSpec 合规性扫描

本节包含有关如何使用 Systems Manager 控制台和 AWS Command Line Interface (AWS CLI) 运行 InSpec 合规性扫描的信息。控制台过程介绍了如何配置 State Manager 来运行该扫描。AWS CLI 过程介绍了如何配置 Run Command 来运行该扫描。

使用 State Manager 运行 InSpec 合规性扫描（控制台）

使用 AWS Systems Manager 控制台通过 State Manager 运行 InSpec 合规性扫描

1. 访问 https://console.aws.amazon.com/systems-manager/，打开 AWS Systems Manager 控制台。
2. 在导航窗格中，请选择 State Manager。
3. 选择 Create association（创建关联）。
4. 在提供关联详细信息部分，输入一个名称。
5. 在 Document (文档) 列表中，选择 **AWS-RunInspecChecks**。
6. 在文档版本列表中，选择运行时的最新版本。
7. 在参数部分的源类型列表中，请选择 GitHub 或 S3。

   如果选择 GitHub，则在源信息字段中输入公有或私有 GitHub 存储库中 InSpec 配置文件的路径。以下是由 Systems Manager 团队从以下位置提供的公有配置文件的示例路径：https://github.com/awslabs/amazon-ssm/tree/master/Compliance/InSpec/PortCheck。

   ```
   {"owner":"awslabs","repository":"amazon-ssm","path":"Compliance/InSpec/
   PortCheck","getOptions":"branch:master"}
   ```

   如果您选择 S3，请在 Source Info (源信息) 字段中输入 S3 存储桶中 InSpec 配置文件的有效 URL。

   有关 Systems Manager 如何与 GitHub 和 Amazon S3 集成的更多信息，请参阅 从 GitHub 运行脚本。
8. 在 Targets（目标）部分中，通过指定标签、手动选择实例或边缘设备或指定资源组，选择要在其上运行此操作的托管式节点。

> **Tip**
>
> 如果未列出您希望看到的托管式节点，请参阅 排除托管式节点可用性的问题 以获取故障排除技巧。

9. 在 Specify schedule (指定计划) 部分中，使用计划生成器选项创建计划，该计划将指定您希望运行合规性扫描的时间。
10. 对于 Rate control（速率控制）：
    - 对于 Concurrency（并发），请指定要同时运行该命令的托管式节点的数量或百分比。

      > **Note**
      >
      > 如果您通过指定应用于托管式节点的标签或指定 AWS Resource Groups 来选择目标，但不确定有多少个托管式节点已被设为目标，则可通过指定百分比来限制可同时运行该文档的目标的数量。

    - 对于 Error threshold（错误阈值），请指定当命令在一定数量或百分比的节点上失败后，何时在其他托管式节点上停止运行该命令。例如，如果您指定三个错误，Systems Manager 将在收到第四个错误时停止发送该命令。仍在处理该命令的托管式节点也可能发送错误。
11. （可选）对于 输出选项，要将命令输出保存到文件，请选中 将命令输出写入 S3 存储桶 框。在输入框中输入存储桶和前缀（文件夹）名称。

> **Note**
>
> 授予将数据写入 S3 存储桶的能力的 S3 权限，是分配给实例的实例配置文件（适用于 EC2 实例）或 IAM 服务角色（混合激活的计算机）的权限，而不是执行此任务的 IAM 用户的权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限或为混合环境创建 IAM 服务角色。此外，如果指定的 S3 存储桶位于不同的 AWS 账户 中，请确保与该托管式节点关联的实例配置文件或 IAM 服务角色具有写入该存储桶的所需权限。

12. 选择创建关联。系统将创建关联并自动运行合规性扫描。
13. 等待几分钟，以完成扫描，然后在导航窗格中选择合规性。
14. 在对应的托管式实例中，找到 Compliance Type（合规性类型）列为 Custom:Inspec 的节点。
15. 选择一个节点 ID 来查看不合规状态的详细信息。

使用 Run Command (AWS CLI) 运行 InSpec 合规性扫描

1. 安装并配置 AWS Command Line Interface（AWS CLI）（如果尚未执行该操作）。

   有关更多信息，请参阅安装或更新 AWS CLI 的最新版本。

2. 运行以下命令之一，从 GitHub 或 Amazon S3 运行 InSpec 配置文件。

   命令使用以下参数：

   - sourceType：GitHub 或 Amazon S3
   - sourceInfo：GitHub 或 S3 存储桶中 InSpec 配置文件文件夹的 URL。该文件夹必须包含基本 InSpec 文件 (*.yml) 和所有相关控件 (*.rb)。

   GitHub

```
aws ssm send-command --document-name "AWS-RunInspecChecks" --targets
 '[{"Key":"tag:tag_name","Values":["tag_value"]}]' --parameters '{"sourceType":
["GitHub"],"sourceInfo":["{\"owner\":\"owner_name\", \"repository\":
\"repository_name\", \"path\": \"Inspec.yml_file"}"]}'
```

   见下列。

```
aws ssm send-command --document-name "AWS-RunInspecChecks" --targets
 '[{"Key":"tag:testEnvironment","Values":["webServers"]}]' --parameters
 '{"sourceType":["GitHub"],"getOptions":"branch:master","sourceInfo":["{\"owner\":
\"awslabs\", \"repository\":\"amazon-ssm\", \"path\": \"Compliance/InSpec/PortCheck
\"}"]}'
```

   Amazon S3

```
aws ssm send-command --document-name "AWS-RunInspecChecks" --targets
 '[{"Key":"tag:tag_name","Values":["tag_value"]}]' --parameters'{"sourceType":
["S3"],"sourceInfo":["{\"path\":\"https://s3.aws-api-domain/amzn-s3-demo-
bucket/Inspec.yml_file\"}"]}'
```

   见下列。

```
aws ssm send-command --document-name "AWS-RunInspecChecks" --targets
 '[{"Key":"tag:testEnvironment","Values":["webServers"]}]' --
```

```
parameters'{"sourceType":["S3"],"sourceInfo":["{\"path\":\"https://s3.aws-api-
domain/amzn-s3-demo-bucket/InSpec/PortCheck.yml\"}"]}'
```

3. 运行以下命令以查看合规性扫描的摘要。

```
aws ssm list-resource-compliance-summaries --filters
 Key=ComplianceType,Values=Custom:Inspec
```

4. 运行以下命令以查看不合规的节点的详细信息。

```
aws ssm list-compliance-items --resource-ids node_ID --resource-type
 ManagedInstance --filters Key=DocumentName,Values=AWS-RunInspecChecks
```

集成 ServiceNow

ServiceNow 提供了一个基于云的服务管理系统，用于创建和管理组织级工作流，例如 IT 服务、工单系统和支持。AWS Service Management Connector 将 ServiceNow 与 Systems Manager 集成，以便从 ServiceNow 预置、管理和操作 AWS 资源。您可以使用 AWS 服务管理连接器，将 ServiceNow 与 AWS Systems Manager 中的所有工具（包括 Automation、Change Manager、Incident Manager 和 OpsCenter）集成。

您可以使用 ServiceNow 执行以下任务：

- 从 Systems Manager 运行自动化 Playbook。
- 从 Systems Manager OpsItems 查看、更新和解决事件。
- 通过 Systems Manager OpsCenter 查看和管理操作项，如事件。
- 从预先批准的更改模板的精选列表中查看和运行 Systems Manager 更改请求。
- 通过与 Incident Manager 集成，管理和解决涉及 AWS 托管应用程序的事件。

> **Note**
>
> 有关如何与 ServiceNow 集成的信息，请参阅《AWS Service Management Connector Administrator Guide》中的 Configuring AWS service integrations。

AWS Systems Manager 引用

以下信息和主题可以帮助您更好地实施 AWS Systems Manager 解决方案。

委托人

在 AWS Identity and Access Management (IAM) 中，您可使用委托人策略元素授予或拒绝授予针对资源的服务访问权限。Systems Manager 的主要策略元素的值是 `ssm.amazonaws.com`。

支持的 AWS 区域 和端点

请参阅《Amazon Web Services 一般参考》中的 Systems Manager service endpoints。

服务配额

请参阅《Amazon Web Services 一般参考》中的 Systems Manager service quotas。

API 引用

请参阅 AWS Systems Manager API 参考。

AWS CLI 命令引用

请参阅 AWS CLI 命令引用的 AWS Systems Manager 部分。

AWS Tools for PowerShell Cmdlet 引用

请参阅 AWS Tools for PowerShell Cmdlet 引用的 AWS Systems Manager 部分。

GitHub 上的 SSM Agent 存储库

请参阅 aws/amazon-ssm-agent。

提问

AWSre:Post 中的 Systems Manager 问题

AWS 新闻博客

管理工具

更多引用主题

- 将此服务与 AWS SDK 结合使用
- 参考：用于修补操作的 Amazon S3 存储桶

- 引用：Amazon EventBridge 事件模式和 Systems Manager 类型
- 参考：适用于 Systems Manager 的 Cron 和 Rate 表达式
- 参考：ec2messages、ssmmessages 和其他 API 操作
- 参考：Systems Manager 的日期和时间字符串格式

将此服务与 AWS SDK 结合使用

AWS 软件开发工具包（SDK）适用于许多常用编程语言。每个软件开发工具包都提供 API、代码示例和文档，使开发人员能够更轻松地以其首选语言构建应用程序。

SDK 文档	代码示例
适用于 C++ 的 AWS SDK	适用于 C++ 的 AWS SDK 代码示例
AWS CLI	AWS CLI 代码示例
适用于 Go 的 AWS SDK	适用于 Go 的 AWS SDK 代码示例
适用于 Java 的 AWS SDK	适用于 Java 的 AWS SDK 代码示例
适用于 JavaScript 的 AWS SDK	适用于 JavaScript 的 AWS SDK 代码示例
适用于 Kotlin 的 AWS SDK	适用于 Kotlin 的 AWS SDK 代码示例
适用于 .NET 的 AWS SDK	适用于 .NET 的 AWS SDK 代码示例
适用于 PHP 的 AWS SDK	适用于 PHP 的 AWS SDK 代码示例
AWS Tools for PowerShell	AWS Tools for PowerShell 代码示例
适用于 Python (Boto3) 的 AWS SDK	适用于 Python (Boto3) 的 AWS SDK 代码示例
适用于 Ruby 的 AWS SDK	适用于 Ruby 的 AWS SDK 代码示例
AWS SDK for Rust	AWS SDK for Rust 代码示例
适用于 SAP ABAP 的 AWS SDK	适用于 SAP ABAP 的 AWS SDK 代码示例
AWS SDK for Swift	AWS SDK for Swift 代码示例

> **示例可用性**
>
> 找不到所需的内容？ 通过使用此页面底部的提供反馈链接请求代码示例。

参考：用于修补操作的 Amazon S3 存储桶

在执行各种 Patch Manager 修补操作的过程中，AWS Systems Manager Agent（SSM Agent）访问 Amazon Web Services（AWS）拥有和维护的某些 Amazon Simple Storage Service（Amazon S3）存储桶。这些 S3 存储桶可以公开访问，默认情况下，SSM Agent 使用 HTTP 调用连接到这些存储桶。

但是，如果您正在 Systems Manager 操作中使用虚拟私有云（VPC）端点，则必须在 Systems Manager 的 Amazon Elastic Compute Cloud（Amazon EC2）实例配置文件中提供显式权限，或者在混合和多云环境中的非 EC2 计算机的服务角色中提供。否则，资源无法访问这些公有存储桶。

本参考主题列出了每个受支持 AWS 区域 的修补相关存储桶。

有关在 EC2 实例配置文件中指定这些 S3 存储桶的信息，请参阅 SSM Agent 与 AWS 托管 S3 存储桶进行通信。

有关将 VPC 端点与 Systems Manager 结合使用的信息，请参阅 使用适用于 Systems Manager 的 VPC 端点提高 EC2 实例的安全性。

主题

- 包含用于修补操作的 SSM 命令文档的存储桶（Linux 和 Windows Server）
- 包含用于修补操作的 SSM 命令文档的存储桶 （macOS）
- 包含 AWS 托管补丁基准快照的存储桶

包含用于修补操作的 SSM 命令文档的存储桶（Linux 和 Windows Server）

格式为 `aws-patch-manager-`*`region`*`-`*`unique-suffix`* 的存储桶包含在 Linux 和 Windows Server 操作系统上进行 Patch Manager 修补操作时使用的以下文档：

- `AWS-RunPatchBaseline`
- `AWS-RunPatchBaselineAssociation`
- `AWS-RunPatchBaselineWithHooks`
- `AWS-InstanceRebootWithHooks`

- `AWS-PatchAsgInstance`
- `AWS-PatchInstanceWithRollback`

区域名称	区域代码	**`aws-patch-manager-`*`region-suffix`*** 存储桶
美国东部（俄亥俄州）	us-east-2	aws-patch-manager-us-east-2-552881074
美国东部（弗吉尼亚州北部）	us-east-1	aws-patch-manager-us-east-1-1970c647d
美国西部（加利福尼亚北部）	us-west-1	aws-patch-manager-us-west-1-8badb4304
美国西部（俄勒冈州）	us-west-2	aws-patch-manager-us-west-2-34d7f99f8
非洲（开普敦）	af-south-1	aws-patch-manager-af-south-1-bdd5f65a9
亚太地区（香港）	ap-east-1	aws-patch-manager-ap-east-1-632356271
亚太地区（海得拉巴）	ap-south-2	aws-patch-manager-ap-south-2-32f4b4128
亚太地区（雅加达）	ap-southeast-3	aws-patch-manager-ap-southeast-3-aa48fc462
亚太地区（墨尔本）	ap-southeast-4	aws-patch-manager-ap-southeast-4-01e2c40d3
亚太地区（孟买）	ap-south-1	aws-patch-manager-ap-south-1-cb7c62ff9
Asia Pacific (Osaka)	ap-northeast-3	aws-patch-manager-ap-northeast-3-67373598a

区域名称	区域代码	**aws-patch-manager-*region*-*suffix*** 存储桶
Asia Pacific (Seoul)	ap-northeast-2	aws-patch-manager-ap-northeast-2-10467995c
亚太地区（新加坡）	ap-southeast-1	aws-patch-manager-ap-southeast-1-7fdfd9ef7
亚太地区（悉尼）	ap-southeast-2	aws-patch-manager-ap-southeast-2-17283a275
Asia Pacific (Tokyo)	ap-northeast-1	aws-patch-manager-ap-northeast-1-4849fa78f
加拿大（中部）	ca-central-1	aws-patch-manager-ca-central-1-3148e69e3
加拿大西部（卡尔加里）	ca-west-1	aws-patch-manager-ca-west-1-9e3a4b2f9
欧洲（法兰克福）	eu-central-1	aws-patch-manager-eu-central-1-9163fdaaf
欧洲地区（爱尔兰）	eu-west-1	aws-patch-manager-eu-west-1-5522fb710
欧洲（伦敦）	eu-west-2	aws-patch-manager-eu-west-2-902a2bc74
欧洲（米兰）	eu-south-1	aws-patch-manager-eu-south-1-c52f3f594
欧洲（巴黎）	eu-west-3	aws-patch-manager-eu-west-3-29bf85721
欧洲（西班牙）	eu-south-2	aws-patch-manager-eu-south-2-a4cf248b1

区域名称	区域代码	**aws-patch-manager-*region*-*suffix*** 存储桶
欧洲地区（斯德哥尔摩）	eu-north-1	aws-patch-manager-eu-north-1-795879e9b
欧洲（苏黎世）	eu-central-2	aws-patch-manager-eu-central-2-184ce43c8
以色列（特拉维夫）	il-central-1	aws-patch-manager-il-central-1-e221cb57b
中东（巴林）	me-south-1	aws-patch-manager-me-south-1-a53fc9dce
中东（阿联酋）	me-central-1	aws-patch-manager-me-central-1-2932f2f80
南美洲（圣保罗）	sa-east-1	aws-patch-manager-sa-east-1-ddf4b6a09

包含用于修补操作的 SSM 命令文档的存储桶（macOS）

格式为 `aws-patchmanager-macos-`*`region`*`-`*`unique-suffix* 的存储桶包含在 macOS 操作系统上进行 Patch Manager 修补操作时使用的以下文档：

- `AWS-RunPatchBaseline`
- `AWS-RunPatchBaselineAssociation`
- `AWS-RunPatchBaselineWithHooks`
- `AWS-InstanceRebootWithHooks`
- `AWS-PatchAsgInstance`
- `AWS-PatchInstanceWithRollback`

区域名称	区域代码	**aws-patchmanager-macos-*region-suffix*** 存储桶
美国东部（俄亥俄州）	us-east-2	aws-patchmanager-macos-us-east-2-552881074
美国东部（弗吉尼亚州北部）	us-east-1	aws-patchmanager-macos-us-east-1-1970c647d
美国西部（加利福尼亚北部）	us-west-1	aws-patchmanager-macos-us-west-1-8badb4304
美国西部（俄勒冈州）	us-west-2	aws-patchmanager-macos-us-west-2-34d7f99f8
非洲（开普敦）	af-south-1	aws-patchmanager-macos-af-south-1-bdd5f65a9
亚太地区（香港）	ap-east-1	aws-patchmanager-macos-ap-east-1-632356271
亚太地区（海得拉巴）	ap-south-2	aws-patchmanager-macos-ap-south-2-32f4b4128
亚太地区（雅加达）	ap-southeast-3	aws-patchmanager-macos-ap-southeast-3-aa48fc462
亚太地区（墨尔本）	ap-southeast-4	aws-patchmanager-macos-ap-southeast-4-01e2c40d3
亚太地区（孟买）	ap-south-1	aws-patchmanager-macos-ap-south-1-cb7c62ff9
Asia Pacific (Osaka)	ap-northeast-3	aws-patchmanager-macos-ap-northeast-3-67373598a
Asia Pacific (Seoul)	ap-northeast-2	aws-patchmanager-macos-ap-northeast-2-10467995c

区域名称	区域代码	**aws-patchmanager-macos-*region*-*suffix*** 存储桶
亚太地区（新加坡）	ap-southeast-1	aws-patchmanager-macos-ap-southeast-1-7fdfd9ef7
亚太地区（悉尼）	ap-southeast-2	aws-patchmanager-macos-ap-southeast-2-17283a275
Asia Pacific (Tokyo)	ap-northeast-1	aws-patchmanager-macos-ap-northeast-1-4849fa78f
加拿大（中部）	ca-central-1	aws-patchmanager-macos-ca-central-1-3148e69e3
加拿大西部（卡尔加里）	ca-west-1	aws-patchmanager-macos-ca-west-1-9e3a4b2f9
欧洲（法兰克福）	eu-central-1	aws-patchmanager-macos-eu-central-1-9163fdaaf
欧洲地区（爱尔兰）	eu-west-1	aws-patchmanager-macos-eu-west-1-5522fb710
欧洲（伦敦）	eu-west-2	aws-patchmanager-macos-eu-west-2-902a2bc74
欧洲（米兰）	eu-south-1	aws-patchmanager-macos-eu-south-1-c52f3f594
欧洲（巴黎）	eu-west-3	aws-patchmanager-macos-eu-west-3-29bf85721
欧洲（西班牙）	eu-south-2	aws-patchmanager-macos-eu-south-2-a4cf248b1
欧洲地区（斯德哥尔摩）	eu-north-1	aws-patchmanager-macos-eu-north-1-795879e9b

区域名称	区域代码	**aws-patchmanager-macos-*region*-*suffix*** 存储桶
欧洲（苏黎世）	eu-central-2	aws-patchmanager-macos-eu-central-2-184ce43c8
以色列（特拉维夫）	il-central-1	aws-patchmanager-macos-il-central-1-e221cb57b
中东（巴林）	me-south-1	aws-patchmanager-macos-me-south-1-a53fc9dce
中东（阿联酋）	me-central-1	aws-patchmanager-macos-me-central-1-2932f2f80
南美洲（圣保罗）	sa-east-1	aws-patchmanager-macos-sa-east-1-ddf4b6a09

包含 AWS 托管补丁基准快照的存储桶

格式为 `patch-baseline-snapshot-`*`region`* 或 `patch-baseline-snapshot-`*`region`*`-`*`unique-suffix`* 的存储桶包含 AWS 托管补丁基准快照。如果您使用下列 SSM 文档，则必须具有此 S3 存储桶的访问权限：

- `AWS-RunPatchBaseline`
- `AWS-RunPatchBaselineAssociation`
- `AWS-RunPatchBaselineWithHooks`
- `AWS-ApplyPatchBaseline`（原有 SSM 文档）

区域名称	区域代码	**patch-baseline-snapshot-*** 存储桶
美国东部（俄亥俄州）	us-east-2	patch-baseline-snapshot-us-east-2

区域名称	区域代码	**patch-baseline-snapshot-*** 存储桶
美国东部（弗吉尼亚州北部）	us-east-1	patch-baseline-snapshot-us-east-1
美国西部（加利福尼亚北部）	us-west-1	patch-baseline-snapshot-us-west-1
美国西部（俄勒冈州）	us-west-2	patch-baseline-snapshot-us-west-2
非洲（开普敦）	af-south-1	patch-baseline-snapshot-af-south-1-tbxdb5b9
亚太地区（香港）	ap-east-1	patch-baseline-snapshot-ap-east-1
亚太地区（海得拉巴）	ap-south-2	patch-baseline-snapshot-ap-south-2-50209442
亚太地区（雅加达）	ap-southeast-3	patch-baseline-snapshot-ap-southeast-3-be0a3174
亚太地区（墨尔本）	ap-southeast-4	patch-baseline-snapshot-ap-southeast-4-dc6f76ce
亚太地区（孟买）	ap-south-1	patch-baseline-snapshot-ap-south-1
Asia Pacific (Osaka)	ap-northeast-3	patch-baseline-snapshot-ap-northeast-3
Asia Pacific (Seoul)	ap-northeast-2	patch-baseline-snapshot-ap-northeast-2
亚太地区（新加坡）	ap-southeast-1	patch-baseline-snapshot-ap-southeast-1

区域名称	区域代码	**patch-baseline-snapshot-*** 存储桶
亚太地区（悉尼）	ap-southeast-2	patch-baseline-snapshot-ap-southeast-2
Asia Pacific (Tokyo)	ap-northeast-1	patch-baseline-snapshot-ap-northeast-1
加拿大（中部）	ca-central-1	patch-baseline-snapshot-ca-central-1
加拿大西部（卡尔加里）	ca-west-1	patch-baseline-snapshot-ca-west-1
欧洲（法兰克福）	eu-central-1	patch-baseline-snapshot-eu-central-1
欧洲地区（爱尔兰）	eu-west-1	patch-baseline-snapshot-eu-west-1
欧洲（伦敦）	eu-west-2	patch-baseline-snapshot-eu-west-2
欧洲（米兰）	eu-south-1	patch-baseline-snapshot-eu-south-1
欧洲（巴黎）	eu-west-3	patch-baseline-snapshot-eu-west-3
欧洲（西班牙）	eu-south-2	patch-baseline-snapshot-eu-south-2-df2c9d70
欧洲地区（斯德哥尔摩）	eu-north-1	patch-baseline-snapshot-eu-north-1
欧洲（苏黎世）	eu-central-2	patch-baseline-snapshot-eu-central-2

区域名称	区域代码	**patch-baseline-snapshot-*** 存储桶
以色列（特拉维夫）	il-central-1	patch-baseline-snapshot-il-central-1
中东（巴林）	me-south-1	patch-baseline-snapshot-me-south-1-uduvl7q8
中东（阿联酋）	me-central-1	patch-baseline-snapshot-me-central-1
南美洲（圣保罗）	sa-east-1	patch-baseline-snapshot-sa-east-1

引用：Amazon EventBridge 事件模式和 Systems Manager 类型

> **Note**
>
> Amazon EventBridge 是管理事件的首选方式。CloudWatch Events 和 EventBridge 是相同的底层服务和 API，但 EventBridge 提供了更多功能。您在 CloudWatch 或 EventBridge 中所做的更改将显示在每个控制台中。有关更多信息，请参阅 Amazon EventBridge 用户指南。

使用 Amazon EventBridge，您可以创建匹配传入事件的规则并将它们路由到目标进行处理。

一个事件表示您自己应用程序、软件即服务（SaaS）应用程序或 AWS 服务中的某个环境发生的一个变化。事件会尽最大努力发出。检测到规则中指定的事件类型后，EventBridge 将其路由到指定的目标进行处理。目标可以包括 Amazon Elastic Compute Cloud (Amazon EC2) 实例、AWS Lambda 函数、Amazon Kinesis streams、Amazon Elastic Container Service (Amazon ECS) 任务、AWS Step Functions 状态计算机、Amazon Simple Notification Service (Amazon SNS) 主题、Amazon Simple Queue Service (Amazon SQS) 队列、内置目标等。

有关创建 EventBridge 规则的信息，请参阅以下主题：

- 使用 Amazon EventBridge 监控 Systems Manager 事件

- 适用于 Systems Manager 的 Amazon EventBridge 事件示例
- Amazon EventBridge 用户指南中的 Amazon EventBridge 入门

本主题的其余部分介绍了可以包含在 EventBridge 规则中的 Systems Manager 事件类型。

事件类型：自动化

事件类型名称	可以添加到规则中的事件的描述
EC2 自动化执行状态更改通知	自动化工作流的总体状态将发生更改。您可以向事件规则添加下列一个或多个状态更改： • 已批准 • 已取消 • 已失败 • 待批准 • 待处理更改日历覆盖 • 已拒绝 • 已安排 • 成功 • 超时
EC2 自动化步骤状态更改通知	自动化工作流中特定步骤的状态将发生更改。您可以向事件规则添加下列一个或多个状态更改： • 已取消 • 已失败 • 成功 • 超时

事件类型：Change Calendar

事件类型名称	可以添加到规则中的事件的描述
日历状态更改	Change Calendar 的状态更改。您可以向事件规则添加下列一个或两个状态更改： • 打开 • 已关闭 目前不支持从其他 AWS 账户 日历的状态更改。

事件类型：Change Manager

事件类型名称	可以添加到规则中的事件的描述
更改请求状态更新	Change Manager 更改请求的状态。您可以在事件规则中使用以下状态： • 已批准 • 已拒绝 • InProgress

事件类型：配置合规性

事件类型名称	可以添加到规则中的事件的描述
配置合规性状态更改	托管式节点的状态针对关联合规性或补丁合规性发生更改。您可以向事件规则添加下列一个或多个状态更改： • 合规 • 不兼容

事件类型：库存

事件类型名称	可以添加到规则中的事件的描述
库存资源状态更改	删除自定义清单和使用旧架构版本的 PutInventory 调用。您可以向事件规则添加下列一个或多个状态更改： • 特定节点上的自定义清单类型删除事件。EventBridge 将针对每个自定义清单类型的每个节点发送一个事件。 • 所有节点的自定义清单类型删除事件。 • 使用旧架构版本事件的 PutInventory 调用。当架构版本小于当前架构时，EventBridge 将发送此事件。此事件适用于所有清单类型。 有关更多信息，请参阅 使用 EventBridge 监控 Inventory 事件。

事件类型：维护窗口

事件类型名称	可以添加到规则中的事件的描述
维护时间段状态更改通知	一个或多个维护时段的总体状态发生更改。您可以向事件规则添加下列一个或多个状态更改： • 已禁用 • 已启用
维护窗口目标登记通知	一个或多个维护窗口目标的状态发生更改。您可以向事件规则添加下列一个或多个状态更改： • 已取消注册 • 已注册 • 已更新

事件类型名称	可以添加到规则中的事件的描述
维护时段执行状态更改通知	维护窗口在运行时的整体状态会发生变化。您可以向事件规则添加下列一个或多个状态更改： • 已取消 • 正在取消 • 已失败 • 正在进行中 • 待处理 • 跳过重叠 • 成功 • 超时
维护窗口任务执行状态更改通知	维护窗口中任务的状态在运行时发生更改。您可以向事件规则添加下列一个或多个状态更改： • 已取消 • 正在取消 • 已失败 • 正在进行中 • 成功 • 超时

事件类型名称	可以添加到规则中的事件的描述
维护窗口任务目标调用状态更改通知	特定目标上的维护窗口任务的状态发生更改。 只有 Run Command 任务才支持此通知。对于此类型的任务，您可以向事件规则添加下列一个或多个状态更改： • 已取消 • 正在取消 • 已失败 • 正在进行中 • 成功 • 超时 对于自动化、AWS Lambda 和 AWS Step Functions 任务，EventBridge 仅报告状态 `IN_PROGRESS` 和 `COMPLETE`。无论任务成功与否，都会报告 `COMPLETE`。
维护窗口任务登记通知	一个或多个维护时段任务的状态发生更改。您可以向事件规则添加下列一个或多个状态更改： • 已取消注册 • 已注册 • 已更新

事件类型：OpsCenter

事件类型名称	可以添加到规则中的事件的描述
OpsItem 创建	在创建 OpsItem 时发生。您可以为以下 OpsItem 类型之一添加规则： • /aws/issue

事件类型名称	可以添加到规则中的事件的描述
	• /aws/task • /aws/insight • /aws/actionitem
OpsItem 更新	更新 OpsItem 时发生。您可以为以下 OpsItem 类型之一添加规则： • /aws/issue • /aws/task • /aws/insight • /aws/actionitem

事件类型：Parameter Store

事件类型名称	可以添加到规则中的事件的描述
参数存储更改	参数的状态发生更改。您可以向事件规则添加下列一个或多个状态更改： • 创建 • 更新 • 删除 • 标签参数版本 有关更多信息，请参阅 为参数和参数策略配置 Eventbridge 规则。
参数存储策略操作	满足高级参数策略更改的条件。您可以向事件规则添加下列一个或多个状态更改： • 过期 • 过期通知 • 无更改通知

事件类型名称	可以添加到规则中的事件的描述
	有关更多信息，请参阅 为参数和参数策略配置 Eventbridge 规则。

事件类型：Run Command

事件类型名称	可以添加到规则中的事件的描述
EC2 命令调用状态更改通知	发送到单个托管实例的命令的状态发生更改。您可以向事件规则添加下列一个或多个状态更改： • 成功 • 正在进行中 • 超时 • 已取消 • 已失败
EC2 命令状态更改通知	命令的总体状态将发生更改。您可以向事件规则添加下列一个或多个状态更改： • 成功 • 正在进行中 • 超时 • 已取消 • 已失败

事件类型：State Manager

事件类型名称	可以添加到规则中的事件的描述
EC2 State Manager 关联状态更改	关联的总体状态随着应用而发生变化。您可以向事件规则添加下列一个或多个状态更改： • 失败

事件类型名称	可以添加到规则中的事件的描述
	• 待处理 • 成功
EC2 State Manager 实例关联状态更改	关联所针对单个托管实例的状态发生更改。您可以向事件规则添加下列一个或多个状态更改： • 失败 • 待处理 • 成功

参考：适用于 Systems Manager 的 Cron 和 Rate 表达式

在 AWS Systems Manager 中创建 State Manager 关联或维护时段时，您需要指定一个说明该时段或关联应在何时运行的计划。您能够以基于时间的条目（称为 cron 表达式）或基于频率的条目（称为 rate 表达式）的形式指定计划。

有关 Cron 和 Rate 表达式的一般信息

以下信息适用于维护时段和关联的 cron 和 rate 表达式。

单次运行计划

当您创建关联或维护时段时，可以按协调世界时 (UTC) 格式指定时间戳，使其在指定的时间运行一次。例如：`"at(2020-07-07T15:55:00)"`

计划偏移天数

关联和维护时段仅支持 cron 表达式的计划偏移天数。计划偏移是在运行关联或维护时段之前但在 cron 表达式指定的日期和时间之后等待的天数。

Maintenance window example

在以下命令中，cron 表达式将维护时段计划在每月第三个星期二的晚上 11:30 运行。但是，由于计划偏移为 2，因此维护时段将在两天后的晚上 11:30 才运行。

```
aws ssm create-maintenance-window \
    --name "My-Cron-Offset-Maintenance-Window" \
```

```
    --allow-unassociated-targets \
    --schedule "cron(30 23 ? * TUE#3 *)" \
    --duration 4 \
    --cutoff 1 \
    --schedule-offset 2
```

Association example

在以下命令中，cron 表达式将关联计划在每个月的第二个星期四运行。但是，由于计划偏移量为 3，因此该关联将在三天后的下周日才会运行。

```
aws ssm create-association \
    --name "AWS-UpdateSSMAgent" \
    --targets "Key=instanceids,Values=i-0cb2b964d3e14fd9f" \
    --schedule-expression "cron(0 0 ? * THU#2 *)"  \
    --schedule-offset 3
    --apply-only-at-cron-interval
```

> **Note**
>
> 要将偏移与关联一起使用，必须指定 `--apply-only-at-cron-interval` 选项。此选项将告知系统不要在创建关联后立即运行该关联。

如果您使用 cron 表达式创建一个关联或维护时段，该维护时段指向当前期间中已经过去的一天，但添加了一个将来的计划偏移日期，则关联或维护时段将不会在此期间内运行。它将在下一个期间生效。例如，如果您指定了一个 cron 表达式（它应在昨天运行维护时段）并添加两天的计划偏移，则该维护时段明天不会运行。

必填字段

维护时段的 Cron 表达式有六个必填字段。关联的 Cron 表达式有五个必填字段。（State Manager 目前不支持在关联的 cron 表达式中指定月数。）另一个字段 Seconds（cron 表达式中的第一个字段）是可选的。这些字段用空格分隔。

Cron 表达式示例

分钟	小时	日期	月份	星期几	年	含义
0	10	*	*	?	*	每天上午的 10:00

分钟	小时	日期	月份	星期几	年	含义
						(UTC) 运行
15	12	*	*	?	*	每天下午 12:15 (UTC) 运行
0	18	?	*	MON-FRI	*	每星期一到星期五下午 6:00 (UTC) 运行
0	8	1	*	?	*	每月第 1 天上午 8:00 (UTC) 运行

支持的值

下表列出了必需的 cron 条目支持的值。

Cron 表达式支持的值

字段	值	通配符
Minutes	0-59	, - * /
小时	0-23	, - * /
Day-of-month	1-31	, - * ? / L W
月（仅限维护时段）	1-12 或 JAN-DEC	, - * /
Day-of-week	1-7 或 SUN-SAT	, - * ? / L #

字段	值	通配符
Year	1970-2199	, - * /

> **Note**
>
> 您不能在同一 cron 表达式中的 day-of-month（日期）和 day-of-week（星期几）字段中指定值。如果您在其中一个字段中指定了值，则必须在另一个字段中使用？（问号）。

Cron 表达式的通配符

下表列出了 cron 表达式支持的通配符值。

> **Note**
>
> 不支持产生的速率快于五 (5) 分钟的 cron 表达式。对同时指定星期几值和日期值的支持不完整。在其中一个字段中使用问号 (?) 字符。

Cron 表达式支持的通配符

通配符	描述
,	,（逗号）通配符包含其他值。在“月份”字段中，JAN、FEB 和 MAR 将包含 January、February 和 March。
-	-（破折号）通配符用于指定范围。在“日”字段中，1-15 将包含指定月份的 1 - 15 日。
*	*（星号）通配符包含该字段中的所有值。在“小时”字段中，* 将包含每个小时。
/	/（正斜杠）通配符用于指定增量。在“分钟”字段中，您可以输入 1/10 以指定从一个小时的第一分钟开始的每个第十分钟。因此，1/10 指

通配符	描述
	定第 1、第 11、第 21 和第 31 分钟，依此类推。
?	?（问号）通配符用于指定一个或另一个。在“日期”字段中，您可以输入 7，如果您不介意 7 日是星期几，则可以在“星期几”字段中输入“?”。
L	Day-of-month（日期）和 Day-of-week（星期几）字段字段中的 L 通配符用于指定该月或该周最后一天。
W	“Day-of-month”字段中的 W 通配符用于指定工作日。在“日期”字段中，3W 用于指定最靠近当月的第三周的日。
#	Day-of-week（星期几）字段中的 # 通配符后接一个介于一到五之间的数值，用于指定该月的特定日期。例如，5#3 表示该月第三个星期四（以星期日作为一星期中的第一天计算）。

Rate 表达式

Rate 表达式有以下两个必需字段。这些字段采用空格分隔。

Rate 表达式的必需字段

字段	值
值	正数，如 1 或 15
单位	minute minutes hour hours

字段	值
	day days

如果值等于 1，则单位必须为单数。同样，对于大于 1 的值，单位必须为复数。例如，`rate(1 hours)` 和 `rate(5 hour)` 无效，而 `rate(1 hour)` 和 `rate(5 hours)` 有效。

主题

- 适用于关联的 Cron 和 Rate 表达式
- 适用于维护时段的 Cron 和 Rate 表达式

适用于关联的 Cron 和 Rate 表达式

此部分包括适用于 State Manager 关联的 cron 和 rate 表达式的示例。在您创建这些表达式之前，请注意以下信息：

- 关联支持以下 cron 表达式：每 1/2、1、2、4、8 或 12 个小时；每天、每周、每周的特定日期和时间；某月中特定某周的特定日期，或某月最后 x 天的特定时间。
- 关联支持以下 rate 表达式：30 分钟的时间间隔或大于和小于 31 天。
- 如果指定可选的 `Seconds` 字段，其值可以为 0（零）。例如：`cron(0 */30 * * * ? *)`
- 对于收集 Inventory（AWS Systems Manager 中的一项工具）元数据的关联，建议使用 Rate 表达式。
- State Manager 目前不支持在 cron 表达式中为关联指定月数。

关联支持 cron 表达式，其中包括一周中的某一天，数字符号 (#) 用于指定一个月的第 n 天来运行关联。以下是一个在每月的第三个星期二 23:30 UTC 运行 cron 计划的示例：

```
cron(30 23 ? * TUE#3 *)
```

以下是一个在每月的第二个星期四在 UTC 午夜运行的示例：

```
cron(0 0 ? * THU#2 *)
```

关联还支持 (L) 符号来表示一个月的最后一个 X 天。以下是一个在每月的最后一个星期二 UTC 午夜运行 cron 计划的示例：

```
cron(0 0 ? * 3L *)
```

要进一步控制关联运行的时间，例如，如果您想在星期二补丁后两天运行关联，可以指定偏移量。偏移量定义了在计划日期之后等待多少天再运行关联。例如，如果指定的 cron 计划为 `cron(0 0 ? * THU#2 *)`，则可以在 Schedule offset（计划偏移量）字段中指定数字 3，以便在每月的第二个星期四之后的每个星期天运行关联。

要使用偏移量，必须在控制台中选择 Apply association only at the next specified Cron interval（仅在下一个指定的 Cron 周期应用关联）选项，或者必须从命令行指定使用 `--apply-only-at-cron-interval` 参数。此选项将告知 State Manager 不要在创建关联后立即运行关联。

下表提供了适用于关联的 cron 示例。

适用于关联的 Cron 示例

示例	详细信息
cron(0/30 * * * ? *)	每 30 分钟
cron(0 0/1 * * ? *)	每小时
cron(0 0/2 * * ? *)	每 2 小时
cron(0 0/4 * * ? *)	每 4 小时
cron(0 0/8 * * ? *)	每 8 小时
cron(0 0/12 * * ? *)	每 12 小时
cron(15 13 ? * * *)	每天下午 1:15
cron(15 13 ? * MON *)	每星期一下午 1:15
cron(30 23 ? * TUE#3 *)	每月第三个星期二晚上 11:30

以下是一些适用于关联的 rate 示例。

适用于关联的 Rate 示例

示例	详细信息
rate(30 minutes)	每 30 分钟
rate(1 hour)	每小时
rate(5 hours)	每 5 小时
rate(15 days)	每 15 天

适用于关联的 AWS CLI 示例

要使用 AWS CLI 创建State Manager关联，您可以使用 cron 或 rate 表达式包含 `--schedule-expression` 参数。以下是在本地 Linux 计算机上使用 AWS CLI 的示例。

> **Note**
>
> 默认情况下，创建新关联时，系统会在创建关联后立即运行它，然后根据您指定的计划运行它。指定 `--apply-only-at-cron-interval`，这样便不会在创建关联后立即运行它。Rate 表达式不支持此参数。

```
aws ssm create-association \
    --association-name "My-Cron-Association" \
    --schedule-expression "cron(0 2 ? * SUN *)" \
    --targets Key=tag:ServerRole,Values=WebServer \
    --name AWS-UpdateSSMAgent
```

```
aws ssm create-association \
    --association-name "My-Rate-Association" \
    --schedule-expression "rate(7 days)" \
    --targets Key=tag:ServerRole,Values=WebServer \
    --name AWS-UpdateSSMAgent
```

```
aws ssm create-association \
    --association-name "My-Rate-Association" \
    --schedule-expression "at(2020-07-07T15:55:00)" \
```

```
    --targets Key=tag:ServerRole,Values=WebServer \
    --name AWS-UpdateSSMAgent \
    --apply-only-at-cron-interval
```

适用于维护时段的 Cron 和 Rate 表达式

本节包括适用于维护时段的 cron 和 rate 表达式的示例。

与 State Manager 关联不同，维护时段支持所有 cron 和 rate 表达式。这包括支持秒数字段中的值。

例如，下面的 6 字段 cron 表达式在每天上午 9:30 运行维护时段。

```
cron(30 09 ? * * *)
```

通过向 Seconds 字段添加一个值，下面的 7 字段 cron 表达式在每天上午 9:30:24 运行维护时段。

```
cron(24 30 09 ? * * *)
```

下表提供更多适用于维护时段的 6 字段 cron 示例。

维护时段的 Cron 示例

示例	详细信息
cron(0 2 ? * THU#3 *)	每月第三个星期四凌晨 2:00
cron(15 10 ? * * *)	每天上午 10:15
cron(15 10 ? * MON-FRI *)	星期一到星期五每天上午 10:15
cron(0 2 L * ? *)	每月最后一天凌晨 2:00
cron(15 10 ? * 6L *)	每月最后一个星期五上午 10:15

下表提供维护时段的 rate 示例。

维护时段的 Rate 示例

示例	详细信息
rate(30 minutes)	每 30 分钟

示例	详细信息
rate(1 hour)	每小时
rate(5 hours)	每 5 小时
rate(25 days)	每 25 天

适用于维护时段的 AWS CLI 示例

要使用 AWS CLI 创建维护时段，您可以使用 cron 或 rate 表达式或者时间戳包含 `--schedule` 参数。以下是在本地 Linux 计算机上使用 AWS CLI 的示例。

```
aws ssm create-maintenance-window \
    --name "My-Cron-Maintenance-Window" \
    --allow-unassociated-targets \
    --schedule "cron(0 16 ? * TUE *)" \
    --schedule-timezone "America/Los_Angeles" \
    --start-date 2021-01-01T00:00:00-08:00 \
    --end-date 2021-06-30T00:00:00-08:00 \
    --duration 4 \
    --cutoff 1
```

```
aws ssm create-maintenance-window \
    --name "My-Rate-Maintenance-Window" \
    --allow-unassociated-targets \
    --schedule "rate(7 days)" \
    --duration 4 \
    --schedule-timezone "America/Los_Angeles" \
    --cutoff 1
```

```
aws ssm create-maintenance-window \
    --name "My-TimeStamp-Maintenance-Window" \
    --allow-unassociated-targets \
    --schedule "at(2021-07-07T13:15:30)" \
    --duration 4 \
    --schedule-timezone "America/Los_Angeles" \
    --cutoff 1
```

更多信息

维基百科网站中的 CRON 表达式

参考：ec2messages、ssmmessages 和其他 API 操作

如果您监控 API 操作，可能会看到对以下操作的调用：

- ec2messages:AcknowledgeMessage
- ec2messages:DeleteMessage
- ec2messages:FailMessage
- ec2messages:GetEndpoint
- ec2messages:GetMessages
- ec2messages:SendReply
- ssmmessages:CreateControlChannel
- ssmmessages:CreateDataChannel
- ssmmessages:OpenControlChannel
- ssmmessages:OpenDataChannel
- ssm:DescribeDocumentParameters
- ssm:DescribeInstanceProperties
- ssm:GetCalendar
- ssm:GetManifest
- ssm:ListInstanceAssociations
- ssm:PutCalendar
- ssm:PutConfigurePackageResult
- ssm:RegisterManagedInstance
- ssm:RequestManagedInstanceRoleToken
- ssm:UpdateInstanceAssociationStatus
- ssm:UpdateInstanceInformation
- ssm:UpdateManagedInstancePublicKey

这些是 AWS Systems Manager 使用的特殊操作，如本主题的其余部分中所述。

与代理相关的 API 操作（ssmmessages 和 ec2messages 端点）

ssmmessages API 操作

Systems Manager 使用 ssmmessages 端点进行以下类型的 API 操作：

- 从 Systems Manager 代理（SSM Agent）到云中 Systems Manager 服务的操作。
- 云中从 SSM Agent 到Session Manager（AWS Systems Manager 中的一项工具）的操作。使用云中的Session Manager服务创建和删除会话通道需要此终端节点。此外，如果允许连接，则 SSM Agent 通过此 Amazon Message Gateway Service 接收 Command 文档。如果不允许连接，则 SSM Agent 通过 Amazon Message Delivery Service 接收 Command 文档。有关更多信息，请参阅用于 Amazon Message Gateway Service 的操作、资源和条件键。
- 操作来自 Run Command。

ec2messages API 操作

对 Amazon Message Delivery Service 端点进行 ec2messages:* API 操作。Systems Manager 将此端点用于从 Systems Manager Agent (SSM Agent) 到云中 Systems Manager 服务的 API 操作。

> **Important**
>
> ec2messages:* API 操作仅在 2024 年之前推出的 AWS 区域 中受支持。在 2024 年及之后推出的区域中，仅支持 ssmmessages:* API 操作。

端点连接优先级

从 SSM Agent 的版本 3.3.40.0 开始，只要可用，Systems Manager 就会使用 ssmmessages:* 端点（Amazon Message Gateway Service）而非 ec2messages:* 端点（Amazon Message Delivery Service）。

如果在 AWS Identity and Access Management（IAM）权限策略中提供对 ssmmessages:* 的访问权限，即便 IAM 实例配置文件配置为允许两个端点，SSM Agent 也会连接到 ssmmessages:* 端点。这包括您自己创建的 IAM 实例配置文件的策略和 IAM 服务角色的策略，以及由 Quick Setup 主机管理配置和默认主机管理配置创建的 IAM 实例配置文件的策略。

如果已经为两个端点提供权限，并使用 CloudWatch Metrics 等监控 API 操作，则不会看到对 ec2messages:* 的调用。

对于 2024 年之前推出的 AWS 区域：此时，您可以放心地删除策略中的 ec2messages:* 权限。

端点连接失效转移

仅适用于 2024 年之前推出的 AWS 区域：如果 IAM 实例配置文件在代理启动时没有为 ssmmessages:* 提供权限，而仅为 ec2messages:* 提供权限，则 SSM Agent 会连接到 ec2messages:* 端点。如果在 SSM Agent 启动时同时拥有 ssmmessages:* 和 ec2messages:*，但在代理启动后移除了 ssmmessages:*，则 SSM Agent 很快就会将连接切换到 ec2messages:* 端点。对于 2024 年及之后推出的区域，仅支持 ssmmessages:* 端点。

有关 ssmmessages 和 ec2messages:* 端点的详细信息，请参阅《AWS Service Authorization Reference》中的以下主题。

- Amazon Message Gateway Service 的操作、资源和条件键 (ssmmessages)。
- Amazon Message Delivery Service 的操作、资源和条件键 (ec2messages:*)

与 ssm:* 命名空间实例相关的 API 操作

DescribeDocumentParameters

Systems Manager 运行此 API 操作以在 Amazon EC2 控制台中呈现特定节点。DescribeDocumentParameters 操作的结果将在文档节点中显示。

DescribeInstanceProperties

Systems Manager 运行此 API 操作以在 Amazon EC2 控制台中呈现特定节点。DescribeInstanceProperties 操作的结果将显示在 Fleet Manager 节点中。

GetCalendar

Systems Manager 运行此 API 操作，在 Change Calendar 控制台中呈现 Change Calendar 类型文档。

GetManifest

SSM Agent 运行此 API 操作以确定安装或更新指定版本 AWS Systems Manager Distributor 软件包的系统要求。这是旧版 API 操作，在 2017 年之后推出的 AWS 区域 中不可用。

ListInstanceAssociations

SSM Agent 运行此 API 操作以了解新的 State Manager 关联是否可用。State Manager 需要此 API 操作才能正常运行。

`PutCalendar`

Systems Manager 运行此 API 操作，在 Change Calendar 控制台中更新 Change Calendar 类型文档。

`PutConfigurePackageResult`

SSM Agent 运行此 API 操作以将公共 Distributor 软件包的安装错误和延迟指标发布到软件包所有者的账户。

`RegisterManagedInstance`

SSM Agent 对以下场景运行此 API 操作：

- 使用激活码和 ID 向 Systems Manager 注册本地服务器或虚拟机（VM）作为托管式实例。
- 注册 AWS IoT Greengrass Version 2 凭证。

运行 SSM Agent 版本 3.1.x 或更高版本的 Amazon EC2 实例也会调用此操作。

`RequestManagedInstanceRoleToken`

SSM Agent 运行此 API 操作以检索访问托管式节点的临时凭证。

`UpdateInstanceAssociationStatus`

SSM Agent 运行此 API 操作以更新关联。State Manager（AWS Systems Manager 中的一项工具）需要此 API 操作才能正常运行。

`UpdateInstanceInformation`

SSM Agent 每 5 分钟调用一次云中的 Systems Manager 服务，以提供检测信号信息。需要此调用来保持代理的检测信号，以便服务了解代理按照预期工作。

`UpdateManagedInstancePublicKey`

在托管式节点上轮换密钥对后，SSM Agent 会运行此 API 操作以提供公有密钥。公有密钥用于对使用私有密钥签名的请求进行身份验证，以从 Systems Manager 获取临时凭证。

ssm: * 命名空间其他 API 操作

`ExecuteApi`

在 OpsCenter 中管理 OpsItems 的 Systems Manager 委托管理员需要访问此 API 操作，这样他们才能跨多个 AWS 账户 查看 OpsItems 的相关资源详细信息。具体而言，该 API 赋予委托管理员在

AWS Management Console 中查看以下 OpsItem 详细信息的权限：OpsItem 描述、标签、AWS CloudFormation 模板、AWS Config 更改、CloudWatch 日志警报和 AWS CloudTrail 事件。有关跨账户使用 OpsItems 的更多信息，请参阅（可选）将 OpsCenter 手动设置为跨账户集中管理 OpsItems。有关 OpsItems 相关资源详情的更多信息，请参阅 将相关资源添加到 OpsItem。

参考：Systems Manager 的日期和时间字符串格式

AWS Systems Manager API 操作接受使用筛选条件来限制请求所返回的结果数量。其中一些 API 操作接受要求使用格式化字符串来表示特定日期和时间的筛选条件。例如，`DescribeSessions` API 操作接受 `InvokedAfter` 和 `InvokedBefore` 键作为 `SessionFilter` 对象的一些有效值。再举个例子，`DescribeAutomationExecutions` API 操作接受 `StartTimeBefore` 和 `StartTimeAfter` 键作为 `AutomationExecutionFilter` 对象的一些有效值。您在筛选请求时为这些键提供的值必须符合 ISO 8601 标准。有关 ISO 8601 的信息，请参阅 ISO 8601。

这些格式化的日期和时间字符串不限于筛选条件。在为请求参数提供值时，还有一些 API 操作要求使用 ISO 8601 格式化的字符串来表示特定的日期和时间。例如，`GetCalendarState` 操作的 `AtTime` 请求参数。这些字符串很难创建。使用本主题中的示例创建要用于 Systems Manager API 操作的格式化日期和时间字符串。

Systems Manager 的格式化日期和时间字符串

以下是 ISO 8601 格式化的日期和时间字符串的示例。

```
2024-05-08T15:16:43Z
```

这表示 2024 年 5 月 8 日 15 时 16 分（协调世界时（UTC））。字符串的日历日期部分由四位数年份、两位数月份和两位数日期表示（以连字符分隔）。这可以用以下格式表示。

```
YYYY-MM-DD
```

字符串的时间部分以字母“T”作为分隔符开头，然后用两位数小时、两位数分钟和两位数秒数表示（以冒号分隔）。这可以用以下格式表示。

```
hh:mm:ss
```

字符串的时间部分以字母“Z”结尾，表示 UTC 标准。

为 Systems Manager 创建自定义日期和时间字符串

您可以使用首选的命令行工具从本地计算机创建自定义日期和时间字符串。用于创建 ISO 8601 格式化的日期和时间字符串的语法因本地计算机的操作系统而异。以下示例说明如何从 Linux 上的 GNU coreutils（或 Windows 上的 PowerShell）使用 `date` 创建 ISO 8601 格式化的日期和时间字符串。

coreutils

```
date '+%Y-%m-%dT%H:%M:%SZ'
```

PowerShell

```
(Get-Date).ToString("yyyy-MM-ddTH:mm:ssZ")
```

使用 Systems Manager API 操作时，您可能需要创建历史日期和时间字符串，以进行报告或故障排除。以下是如何为 AWS Tools for PowerShell 和 AWS Command Line Interface (AWS CLI) 创建并使用自定义历史 ISO 8601 格式化的日期和时间字符串的示例。

AWS CLI

- 检索 SSM 文档的最后一周命令历史记录。

```
lastWeekStamp=$(date '+%Y-%m-%dT%H:%M:%SZ' -d '7 days ago')

docFilter='{"key":"DocumentName","value":"AWS-RunPatchBaseline"}'
timeFilter='{"key":"InvokedAfter","value":'\"$lastWeekStamp\"'}'

commandFilters=[$docFilter,$timeFilter]

aws ssm list-commands \
    --filters $commandFilters
```

- 检索最后一周的自动化执行历史记录。

```
lastWeekStamp=$(date '+%Y-%m-%dT%H:%M:%SZ' -d '7 days ago')

aws ssm describe-automation-executions \
    --filters Key=StartTimeAfter,Values=$lastWeekStamp
```

- 检索最后一个月的会话历史记录。

```
lastWeekStamp=$(date '+%Y-%m-%dT%H:%M:%SZ' -d '30 days ago')

aws ssm describe-sessions \
    --state History \
    --filters key=InvokedAfter,value=$lastWeekStamp
```

AWS Tools for PowerShell

- 检索 SSM 文档的最后一周命令历史记录。

```
$lastWeekStamp = (Get-Date).AddDays(-7).ToString("yyyy-MM-ddTH:mm:ssZ")

$docFilter = @{
    Key="DocumentName"
    Value="AWS-InstallWindowsUpdates"
}
$timeFilter = @{
    Key="InvokedAfter"
    Value=$lastWeekStamp
}

$commandFilters = $docFilter,$timeFilter

Get-SSMCommand `
    -Filters $commandFilters
```

- 检索最后一周的自动化执行历史记录。

```
$lastWeekStamp = (Get-Date).AddDays(-7).ToString("yyyy-MM-ddTH:mm:ssZ")

Get-SSMAutomationExecutionList `
    -Filters @{Key="StartTimeAfter";Values=$lastWeekStamp}
```

- 检索最后一个月的会话历史记录。

```
$lastWeekStamp = (Get-Date).AddDays(-30).ToString("yyyy-MM-ddTH:mm:ssZ")

Get-SSMSession `
    -State History `
    -Filters @{Key="InvokedAfter";Value=$lastWeekStamp}
```

应用场景和最佳实践

本主题列出了 AWS Systems Manager 工具常见应用场景和最佳实践。如果可用，本主题还包括指向相关博客文章和技术文档的链接。

> **Note**
>
> 此处每个章节的标题都是一个指向技术文档中相应章节的有效链接。

自动化

- 为基础设施创建自助服务自动化运行手册。
- 使用 Automation（AWS Systems Manager 中的一项工具），简化利用公有 Systems Manager 文档（SSM 文档）或通过创作自己的工作流从 AWS Marketplace 或自定义 AMIs 创建Amazon Machine Images（AMIs）的过程。
- 使用 `AWS-UpdateLinuxAmi` 和 `AWS-UpdateWindowsAmi` 自动化运行手册，或使用您创建的自定义自动化运行手册，构建和维护 AMIs。

合规

- 作为最佳安全实践，建议您更新托管式节点使用的 AWS Identity and Access Management（IAM）角色，以限制该节点使用 PutComplianceItems API 操作的功能。该 API 操作会在指定资源（例如 Amazon EC2 实例或托管式节点）上注册合规性类型及其他合规性详细信息。有关更多信息，请参阅 配置合规性权限。

清单

- 将 Inventory（AWS Systems Manager 中的一项工具）与 AWS Config 结合使用，便可随着时间推移审核应用程序配置。

Maintenance Windows

- 制定计划，以在节点上执行可能造成中断的操作，例如操作系统 (OS) 修补、驱动程序更新或软件安装。

- 有关State Manager与 Maintenance Windows（都是 AWS Systems Manager 中的工具）的差异的更多信息，请参阅在 State Manager 和 Maintenance Windows 之间选择。

Parameter Store

- 使用 Parameter Store（AWS Systems Manager 中的一项工具）集中管理全局配置设置。
- AWS Systems ManagerParameter Store 如何使用 AWS KMS。
- 通过 Parameter Store 参数引用 AWS Secrets Manager 密钥。

Patch Manager

- 使用Patch Manager（AWS Systems Manager 中的一项工具）可大规模推出补丁，在节点中提高实例集合规可见性。
- 将 Patch Manager 与 AWS Security Hub 集成在一起，以便在机群中的节点不合规时收到提醒，并从安全角度监控机群的修补状态。使用 Security Hub 需支付费用。有关更多信息，请参阅 定价。
- 为避免意外覆盖合规性数据，每次仅使用一种方法来扫描托管节点的补丁合规性。

Run Command

- 使用 EC2 Run Command 在不使用 SSH 访问的情况下大规模地管理实例。
- 使用 AWS CloudTrail 审核由 Run Command（AWS Systems Manager 中的一项工具）进行或代表其进行的所有 API 调用。
- 当您使用 Run Command 发送命令时，不要包含纯文本格式的敏感信息，例如密码、配置数据或其他密钥。您账户中的所有 Systems Manager API 活动都将记录在 AWS CloudTrail 日志的 S3 存储桶中。这意味着任何有权访问该 Amazon S3 存储桶的用户都能够查看这些密钥的纯文本值。因此，我们建议您创建和使用 `SecureString` 参数，以便加密您在 Systems Manager 操作中使用的敏感数据。

  有关更多信息，请参阅 使用 IAM 策略限制对 Parameter Store 参数的访问。

> **Note**
>
> 预设情况下，CloudTrail 提交到您存储桶的日志文件是用 Amazon S3 托管加密密钥 (SSE-S3) 通过 Amazon 服务器端加密进行加密的。要提供可直接管理的安全层，您可以改为将服务器端加密与 AWS KMS 托管密钥 (SSE-KMS) 结合起来用于 CloudTrail 日志文件。

有关更多信息，请参阅《AWS CloudTrail 用户指南》中的使用 AWS KMS 托管式密钥（SSE-KMS）加密 CloudTrail 日志文件。

- 使用 Run Command 中的目标和速率控制功能执行暂存的命令操作。
- 借助 AWS Identity and Access Management（IAM）策略，将精细访问权限用于 Run Command（以及其余 Systems Manager 工具）。

Session Manager

- 使用 AWS CloudTrail 记录您的 AWS 账户中的会话活动。
- 使用 Amazon CloudWatch Logs 或 Amazon S3 记录 AWS 账户 中的会话数据。
- 控制用户会话对实例的访问。
- 限制对会话中命令的访问。
- 关闭或打开 SSM 用户账户管理权限。

State Manager

- 使用预配置的 `AWS-UpdateSSMAgent` 文档，至少每月更新 SSM Agent 一次。
- (Windows) 将 PowerShell 或 DSC 模块上载到 Amazon Simple Storage Service (Amazon S3)，并使用 `AWS-InstallPowerShellModule`。
- 使用标签为节点创建应用程序组。然后使用 `Targets` 参数而不是指定各个节点 ID 来将节点设为目标。
- 使用 Systems Manager 自动修复 Amazon Inspector 生成的结果。
- 对 SSM 文档使用集中式配置存储库，并在整个组织中共享文档。
- 有关 State Manager 与 Maintenance Windows 的差异的更多信息，请参阅 在 State Manager 和 Maintenance Windows 之间选择。

托管式节点

- Systems Manager 需要准确的时间参考以执行其操作。如果节点的日期和时间设置不正确，它们可能与 API 请求的签名日期不匹配。这可能会导致错误或功能不完整。例如，时间设置不正确的节点不会包含在您的托管式节点列表中。

  有关在节点上设置时间的更多信息，请参阅为 Amazon EC2 实例设置时间。
- 在 Linux 托管节点上，请验证 SSM Agent 的签名。

更多信息

- Systems Manager 的安全最佳实践

删除 Systems Manager 资源和构件

作为最佳实践，如果您不再需要查看有关这些资源的数据，或者不再需要以任何方式使用构件，我们建议您删除 Systems Manager 资源和构件。下表列出了 Systems Manager 的各个工具或构件，以及指向有关删除 Systems Manager 创建的资源或构件的详细信息的链接。

功能或构件	详细信息
Application Manager	您无法在 Application Manager 中删除应用程序，但您可以通过删除底层标签、Resource Groups 或 AWS CloudFormation 堆栈，从服务中删除应用程序。
自动化	如果您使用 Systems Manager 自动化创建 AWS 资源，则必须使用相应的 AWS Management Console 手动删除这些资源。如果您创建了自定义运行手册，则可删除底层 SSM 文档。有关更多信息，请参阅 删除自定义 SSM 文档。
Change Calendar	您可以删除更改日历和更改日历事件。有关更多信息，请参阅 删除更改日历 和 删除 Change Calendar 事件。
Change Manager	您可以删除更改模板。有关更多信息，请参阅 删除更改模板。
Compliance	Systems Manager Compliance 自动显示有关 Patch Manager 修补和 State Manager 关联的合规性数据。您无法删除此数据。如果您将资源数据同步配置为将合规性数据集中在 S3 存储桶中，则可删除该同步。有关更多信息，请参阅 删除用于 Compliance 的资源数据同步。

功能或构件	详细信息
Distributor	您可以在 Distributor 中删除程序包。有关更多信息，请参阅 删除 Distributor 软件包。
Explorer	您可以断开与 Explorer 从中收集 OpsData 的源的连接。有关更多信息，请参阅 编辑 Systems Manager Explorer 数据源。 您还可以删除 Explorer 用于将来自多个 AWS 区域和账户的 OpsData 及 OpsItems 聚合到单个 Amazon Simple Storage Service（Amazon S3）存储桶的资源数据同步。有关更多信息，请参阅 删除 Systems Manager Explorer 资源数据同步。有关删除 S3 存储桶的信息，请参阅《Amazon Simple Email Service 开发人员指南》中的删除存储桶。
Fleet Manager	您无法使用 Fleet Manager 删除托管式节点。而必须使用 Amazon Elastic Compute Cloud（Amazon EC2）。有关更多信息，请参阅终止您的实例 (Linux) 和终止您的实例 (Windows)。
清单	您可以通过删除用于定义时间表和从中收集元数据的资源的 State Manager 关联，来停止 Inventory 数据收集。有关更多信息，请参阅 停止数据收集和删除清单数据。 如果您不再需要使用 AWS Systems Manager Inventory 来查看有关您的 AWS 资源的元数据，那么我们还建议删除用于库存数据收集的资源数据同步。有关更多信息，请参阅 删除 Inventory 资源数据同步。
Maintenance Windows	您可以删除维护时段、维护时段目标和维护时段任务。有关更多信息，请参阅 使用控制台更新或删除维护时段资源。

功能或构件	详细信息
OpsCenter	您可以使用 AWS Command Line Interface 或 AWS SDK 来调用 DeleteOpsItem API 操作，从而删除单个 OpsItem。您无法在 AWS Management Console 中删除 OpsItem。有关更多信息，请参阅 删除 OpsItems。
Parameter Store	您可以删除已创建的参数。有关更多信息，请参阅 从 Parameter Store 中删除参数。
Patch Manager	您可以删除自定义补丁基准。有关更多信息，请参阅 更新或删除自定义补丁基准。
快速设置	您可以删除快速设置创建的关联。关联由 State Manager 存储和处理。有关更多信息，请参阅 删除关联。
Run Command	在某一命令完成处理后，有关该命令的信息将存储在命令历史记录选项卡中。您无法从命令历史记录选项卡中删除信息。
自动化	在某一自动化完成处理后，有关该自动化的信息将存储在执行选项卡中。您无法从执行选项卡中删除信息。
服务相关角色	Systems Manager 可为某些工具自动创建服务相关角色。您可以删除这些角色。有关更多信息，请参阅 删除 Systems Manager 的 AWSServiceRoleForAmazonSSM 服务相关角色。
Session Manager	Session Manager 不会在您终止会话后保留有关您的资源的数据。要终止会话，请参阅 结束会话。

功能或构件	详细信息
SSM Agent	您可以从节点中手动卸载 SSM Agent。有关更多信息，请参阅以下主题。 • Linux：在适用于 Linux 的 EC2 实例上手动安装和卸载 SSM Agent • macOS: 在适用于 macOS 的 EC2 实例上手动安装和卸载 SSM Agent • Windows Server：打开控制面板，然后选择添加/删除程序。
State Manager	您可以删除关联。有关更多信息，请参阅 删除关联。
Systems Manager 文档服务	您不能删除由 AWS 或 AWS 支持 提供的运行手册，但可以删除自定义运行手册。有关更多信息，请参阅 删除自定义 SSM 文档。

在 State Manager 和 Maintenance Windows 之间选择

State Manager和 Maintenance Windows（都是 AWS Systems Manager 中的工具）可在托管式节点上执行一些相似类型的更新。您选择哪一项取决于您是需要自动执行系统合规性，还是在指定的时间段内执行高优先级、时效性强的任务。

State Manager 和 Maintenance Windows：关键使用案例

State Manager（AWS Systems Manager 中的一项工具）可为 AWS 账户中的托管式节点和 AWS 资源设置目标状态配置，并加以保持。您可以将配置和目标的组合定义为关联对象。如果您希望使账户中的所有托管式节点保持一致状态，使用 Amazon EC2 Auto Scaling 生成新节点，或者对账户中的托管式节点具有严格的合规性报告要求，则建议使用State Manager工具。

State Manager 的主要使用案例如下：

- Auto Scaling 方案：State Manager 可以手动或通过 Auto Scaling 组监控账户内启动的所有新节点。如果账户中存在以该新节点为目标（通过标签或所有节点设为目标）的任何关联，则该特定关联将自动应用于该新节点。

- 合规性报告：State Manager可为账户中的资源推动所需状态的合规性报告。
- 支持所有节点：State Manager 可将给定账户内的所有节点设为目标。

维护时段可在给定时段内对 AWS 资源执行一项或多项操作。您可以定义具有开始时间和结束时间的单个维护时段。您可以指定多个任务在此维护时段内运行。如果您的高优先级操作包括修补托管式节点、在更新期间对节点运行多种类型的任务，或者控制何时可在节点上运行更新操作，则使用 Maintenance Windows（AWS Systems Manager 中的一项工具）。

Maintenance Windows 的主要使用案例如下：

- 运行多个文档：维护时段可以运行多个任务。每个任务均可使用不同的文档类型。因此，您可以在一个维护时段内使用不同的任务构建复杂的工作流。
- 修补：维护时段可为用特定标签或资源组标记的单个区域内的所有托管式节点提供修补支持。由于修补通常涉及关闭节点（例如，从负载均衡器中删除节点）、修补和后期处理（将节点重新投入生产），因此可在给定补丁时段内以一系列任务的形式实现修补。

  > **Note**
  >
  > 使用维护时段，您的修补操作仅限于单个账户中的单个区域。使用在Quick Setup（Systems Manager 中的一项工具）中创建的补丁策略，您可以为在 AWS Organizations 中创建的组织中的部分或全部账户和区域配置补丁。有关更多信息，请参阅 Quick Setup 中的补丁策略配置。

- 时段操作：维护时段可使一组或多组操作在特定时段内开始。维护时段不会在该时段之外开始。已开始的操作将一直持续到完成，即便其完成时间超出该时段。

下表对 State Manager 和 Maintenance Windows 的主要功能进行了比较。

功能	State Manager	Maintenance Windows
AWS CloudFormation 集成	AWS CloudFormation 模板支持 State Manager 关联。	AWS CloudFormation 模板支持维护时段、时段目标和时段任务。
Compliance	每个 State Manager 关联都会报告与目标资源的所需状态相关的合规性。可以使用	不适用。

功能	State Manager	Maintenance Windows
	Compliance Dashboard (合规性控制面板) 聚合和查看报告的合规性。	
配置管理集成	State Manager 支持外部目标状态解决方案，如 Microsoft PowerShell Desired State Configuration（DSC）、Ansible Playbook 和 Chef Recipe。可以使用 State Manager 关联来测试配置管理解决方案是否有效，并在准备就绪时将其配置更改应用于您的节点。	不适用。
文档	对于 AWS 资源，例如 Amazon Simple Storage Service (Amazon S3) 存储桶，State Manager 配置可定义为策略文档（用于收集库存信息）、自动化运行手册；而对于托管式节点，则可定义为 Systems Manager 命令文档（SSM 文档）。	Maintenance Windows 配置可以定义为自动化文档（使用可选批准工作流的多步操作）或 SSM 文档（托管式节点的所需状态）。
监控	State Manager 可以监控节点的配置、关联或状态的更改（例如，新节点联机）。当 State Manager 检测到这些更改时，给定关联将重新应用于最初通过该关联设为目标的节点。	不适用。

功能	State Manager	Maintenance Windows
任务中的优先级	不适用。	可以为维护时段内的任务分配优先级。优先级相同的所有任务都将并行运行。优先级较低的任务将在优先级较高的任务达到最终状态后运行。无法有条件地运行任务。优先级较高的任务达到其最终状态后，下一优先级的任务开始运行，而不考虑上一个任务的状态。
安全控制	在大型机群中部署配置时，State Manager 支持两种安全控制。可以使用最大并发数来定义应用的配置应该包含的并发节点或资源的数量。您可以定义一个最大错误率，如果在整个机群中发生一定数量或百分比的错误，该错误率可用于暂停 State Manager 关联。	在大型机群中部署配置时，维护时段支持两种安全控制。可以使用最大并发数来定义应用的配置应该包含的并发节点或资源的数量。您可以定义一个最大错误率，如果在整个机群中发生一定数量或百分比的错误，该错误率可用于暂停维护时段中的操作。

功能	State Manager	Maintenance Windows
计划编制	您可以按需、以特定 Cron 间隔、以给定速率或在创建后运行 State Manager 关联。如果您希望以一致和及时的方式保持资源的所需状态，这将非常有用。 **Important** 适用于 State Manager 关联的 Cron 表达式不支持 months（月份）字段，例如表示三月的 `03` 或 `MAR`。如果您需要每月或每季度进行一次配置更新，则维护时段最适合满足您的需求。有关更多信息，请参阅 参考：适用于 Systems Manager 的 Cron 和 Rate 表达式。	维护时段支持多种计划编制选项，包括 `at` 表达式（例如 `"at(2021-07-07T13:15:30)"` ）、Cron 和速率表达式、带偏移的 Cron、指定维护时段何时应该运行的开始时间和结束时间，以及指定何时在给定时段内停止计划编制的截止时间。
设置目标	State Manager 关联可以使用节点 ID、标签或资源组将一个或多个节点设为目标。State Manager 可将给定账户内的所有托管式节点设为目标。	维护时段可以使用节点 ID、标签或资源组将一个或多个节点设为目标。

功能	State Manager	Maintenance Windows
维护时段内的任务	不适用。	维护时段可以支持一个或多个任务，其中每个任务都将特定的 自动化运行手册或 Command 文档操设为目标。除非为不同任务设置了不同的优先级，否则一个维护时段内的所有任务都将并行运行。 总的来说，维护时段支持四种类型的任务： • AWS Systems Manager Run Command 命令 • AWS Systems Manager 自动化工作流 • AWS Lambda 函数 • AWS Step Functions 任务

相关信息

下列相关资源在您使用此服务的过程中会有所帮助。

定价

使用某些 Systems Manager 工具会产生费用。有关更多信息，请参阅AWS Systems Manager 定价。

AWS Systems Manager 文档库

AWS Systems Manager 文档：访问 Systems Manager（包括 AWS AppConfig、Incident Manager 和 SAP 的 AWS Systems Manager）的所有用户文档。

AWS re:Post

AWS re:Post – AWS 托管式问答（Q & A）服务，为您的技术问题提供众包、经专家审查的答案。

AWS 博客和播客

阅读 AWS 管理工具类别中有关 Systems Manager 的博客文章，以及标记有 #Systems Manager 的其他文章。

服务配额

查看《Amazon Web Services 一般参考》中的 Systems Manager service quotas。除非另有说明，每个配额适用于 AWS 账户 中的单个区域。

Systems Manager 服务授权参考

在《AWS 服务授权参考》中，查看有关您可以在 Systems Manager AWS Identity and Access Management（IAM）policy 中使用的操作、资源和全局条件上下文键的信息。

AWS Systems Manager 服务水平协议

AWS Systems Manager 服务水平协议（SLA）是管理 Systems Manager 使用的策略，并单独适用于每个使用 Systems Manager 的 AWS 账户。

一般 AWS 资源

以下一般资源在您使用 AWS 的过程中会有所帮助。

- 课程和研讨会 – 指向基于角色的专业课程和自主进度动手实验室的链接，这些课程和实验室旨在帮助您增强 AWS 技能并获得实践经验。

- AWS 开发人员中心 – 浏览教程、下载工具并了解 AWS 开发人员活动。
- AWS 开发人员工具 – 指向开发人员工具、开发工具包、IDE 工具包和命令行工具的链接，这些资源用于开发和管理 AWS 应用程序。
- 入门资源中心 – 了解如何设置 AWS 账户、加入 AWS 社区和启动您的第一个应用程序。
- 动手实践教程 – 按照分步教程在 AWS 上启动您的第一个应用程序。
- AWS 白皮书 – 指向 AWS 技术白皮书的完整列表的链接，这些资料涵盖了架构、安全性、经济性等主题，由 AWS 解决方案架构师或其他技术专家编写。
- AWS 支持 中心 – 用于创建和管理 AWS 支持 案例的中心。还提供指向其他有用资源的链接，如论坛、技术常见问题、服务运行状况以及AWS Trusted Advisor。
- 支持 – 提供有关 支持 的信息的主要网页，这是一个一对一的快速响应支持渠道，可以帮助您在云中构建和运行应用程序。
- 联系我们 – 用于查询有关 AWS 账单、账户、事件、滥用和其他问题的中央联系点。
- AWS 网站条款 – 有关我们的版权和商标、您的账户、许可、网站访问和其他主题的详细信息。

文档历史记录

下表列出了自 AWS Systems Manager 上一次发布以来对文档所做的重要更改。如需获取对此文档的更新的通知，您可以订阅 RSS 源。

变更	说明	日期
针对 SLES 托管节点上 zypper 软件包锁定依赖项故障的新疑难解答指南	添加了全面的疑难解答指南，用于解决在 SUSE Linux Enterprise Server 托管节点上进行 Patch Manager `Install` 操作期间发生的 zypper 软件包锁定依赖项故障问题。新增的“问题：SLES 托管节点上的 Zypper 软件包锁定依赖项故障”章节提供了有关识别、诊断和解决阻碍成功安装修补程序的软件包锁定冲突问题的详细信息。	2025 年 7 月 28 日
针对已跳过清单关联的增强版疑难解答指南	更新了“已跳过”的疑难解答部分，从而为清单关联优先级的处理提供全方面的指导。增强版文档现在包括有关 Quick Setup 清单关联的详细信息、明确目标设置场景以及对冲突的清单关联进行管理的分步解决程序。 重要改进包括： • 说明清单关联优先顺序（Quick Setup、显式和全局关联） • 常见场景附有具体示例和预期结果	2025 年 7 月 24 日

	• 提供使用自定义清单关联时删除 Quick Setup 配置的详细解决步骤 • 提供有关数据收集行为和关联优先级的重要须知	
更新了托管策略：AWS-SSM-RemediationAutomation-ExecutionRolePolicy	Systems Manager 更新了托管策略 `AWS-SSM-RemediationAutomation-ExecutionRolePolicy` ，要求同时拥有“文档”和“自动执行”资源类型的权限，从而改善 ssm:StartAutomationExecution API 的安全状况。此更新旨在通过要求同时拥有“文档”和“自动执行”资源类型的权限来改善 ssm:StartAutomationExecution API 的安全状况，从而为修复自动化执行提供更加全面和详细的权限。有关更多信息，请参阅 Systems Manager 对 AWS 托管式策略的更新。	2025 年 7 月 16 日
更新了托管策略：AWS-SSM-RemediationAutomation-AdministrationRolePolicy	Systems Manager 更新了托管策略 `AWS-SSM-RemediationAutomation-AdministrationRolePolicy` ，以支持 API 授权改进实现修复自动化操作。更新后的策略强化了执行自动化文档中定义活动的权限，并改进了修复工作流程的安全控制和资源访问模式。有关更多信息，请参阅 Systems Manager 对 AWS 托管式策略的更新。	2025 年 7 月 16 日

更新了托管策略：AWS-SSM-DiagnosisAutomation-ExecutionRolePolicy	Systems Manager 更新了托管策略 `AWS-SSM-DiagnosisAutomation-ExecutionRolePolicy` ，从而为诊断自动化执行提供更加详细和准确的权限。更新后的策略包括改善对 Amazon EC2 和 Amazon VPC 资源访问的描述，提供更加具体的 SSM 自动化权限，以及通过适当资源限制改善对 AWS KMS 和 IAM 权限的描述。有关更多信息，请参阅 Systems Manager 对 AWS 托管式策略的更新。	2025 年 7 月 16 日
更新了托管策略：AWS-SSM-DiagnosisAutomation-AdministrationRolePolicy	Systems Manager 更新了托管策略 `AWS-SSM-DiagnosisAutomation-AdministrationRolePolicy` ，从而为诊断自动化操作提供更加具体的权限和安全条件。更新后的策略增强了对 AWS KMS 密钥使用、Amazon S3 存储桶访问和角色代入的安全控制，并采用更加严格的资源条件和账户级别限制。有关更多信息，请参阅 Systems Manager 对 AWS 托管式策略的更新。	2025 年 7 月 16 日
添加了 AmazonSSMAutomationRole 托管策略文档	Systems Manager 更新了现有托管策略 `AmazonSSMAutomationRole` 的文档。有关更多信息，请参阅 Systems Manager 对 AWS 托管式策略的更新。	2025 年 7 月 15 日

使用环境变量插值增强了 SSM 文档的安全性	AWS Systems Manager 命令文档现在支持在处理参数时使用环境变量插值。此功能在 2.2 版架构和 3.3.2746.0 或更高版本的 SSM Agent 中可用，可帮助防范命令注入攻击，做法是将参数值视为文字字符串，而非可能的可执行命令。设置为 ENV_VAR 时，代理会创建一个包含参数值的环境变量，格式为 SSM_*parameter-name* 。 有关更多信息，请参阅以下主题： • 文档组件 • 安全处理参数示例 • 数据元素和参数 • 命令文档插件参考	2025 年 7 月 14 日

Explorer 资源数据同步已在四个选择加入 AWS 区域推出	默认情况下，Explorer 跨账户/跨区域资源数据同步不支持选择加入区域中的数据聚合。2025 年 6 月 30 日，增加了对以下选择加入区域的支持。 • 欧洲地区（米兰） • 非洲（开普敦） • 中东（巴林） • 亚太地区（香港） 请注意，您不能使用委派管理员账户在选择加入区域中创建资源数据同步。您必须使用 AWS Organizations 管理账户。有关更多信息，请参阅了解多个账户和区域资源数据同步。	2025 年 6 月 30 日
AWS 参数和密钥 Lambda 扩展的新版本	新版 AWS Parameters and Secrets Lambda 扩展现在适用于所有支持的区域和架构。此外，还增加了对亚太地区（台北）区域 (ap-east-2) 的支持。	2025 年 6 月 26 日
新主题：Change Calendar 与 Amazon EventBridge 集成	主题 Change Calendar 与 Amazon EventBridge 集成按以下类别提供有关此集成的信息： • 事件计时和可靠性 • 事件修改和计时注意事项 • 相邻和重叠事件 • 最佳实践	2025 年 6 月 25 日

自动化操作“aws:executeScript”的运行时支持更新	自动化操作 aws:executeScript 中的 `Runtime` 参数现在支持 PowerShell 7.4 (`dotnet8`) 脚本。	2025 年 6 月 24 日
SSM Agent 和 Patch Manager 现在支持 RHEL 10 和 Ubuntu Server 25.04	现在，您可以使用 Systems Manager 管理 Red Hat Enterprise Linux 10 和 Ubuntu Server 25.04 计算机，包括使用Patch Manager修补托管节点。有关支持的操作系统和版本的完整列表，请参阅以下主题： • Systems Manager 支持的操作系统 • Patch Manager 支持的操作系统	2025 年 6 月 23 日

即时节点访问主题的更新	我们对使用 Systems Manager 进行即时节点访问一章中的内容进行了以下更新： • 阐明了对即时节点访问的身份验证支持。有关信息，请参阅使用 Systems Manager 设置即时访问。 • 阐明了使用标签键 `SystemsManagerJustInTimeNodeAccessManaged` 和标签值 `true` 来标记用于即时节点访问中的 Session Manager 加密和 RDP 记录的 AWS KMS 密钥。有关信息，请参阅更新即时节点访问会话首选项和配置用于 RDP 录制的 S3 存储桶加密。 • 更新了为即时访问请求配置通知并添加了其他配置详细信息。	2025 年 6 月 17 日

增强了 RDP 录制功能，以支持即时节点访问	AWS Systems Manager 即时节点访问功能现在可提供有关 RDP 录制配置的更多信息。在存储录制的 S3 存储桶上启用 AWS KMS 加密时，必须向 `ssm-guiconnect` 服务主体提供存储桶加密所用客户自主管理型密钥的访问权限。本次发布还引入了新的录制状态 `ProcessingError` ，用于表示视频处理过程中可能因服务依赖项故障或权限缺失导致的中间故障。有关更多信息，请参阅录制 RDP 连接。	2025 年 6 月 13 日
更新了中有关在 Patch Manager 中选择和安装补丁的信息	增加了有关 Patch Manager 在补丁选择和安装过程中会如何处理已过时软件包的说明信息。有关更多信息，请参阅如何选择安全性补丁和如何安装补丁。	2025 年 6 月 12 日
更新了 Quick Setup 的托管式策略：AWSQuickSetupDeploymentRolePolicy	Systems Manager 更新了托管策略 `AWSQuickSetupDeploymentRolePolicy` ，从而让 Quick Setup 可以使用此托管式策略而不是内联策略来创建关联。有关更多信息，请参阅 Systems Manager 对 AWS 托管式策略的更新。	2025 年 6 月 10 日

完善了对 Patch Manager 的介绍	为更好地向潜在客户和用户介绍 Patch Manager，我们在 Patch Manager 章节中增加了以下三个部分： • Patch Manager 对我的组织有哪些益处？ • 谁该使用 Patch Manager？ • Patch Manager 的主要功能有哪些？	2025 年 5 月 21 日
更新了 Quick Setup 的托管式策略：AWSQuickSetupSSMDeploymentRolePolicy	Systems Manager 更新了托管式策略 `AWSQuickSetupSSMDeploymentRolePolicy` ，以提供标记 IAM 角色和为统一控制台创建的 Lambda 资源的权限。有关更多信息，请参阅 Systems Manager 对 AWS 托管式策略的更新。	2025 年 5 月 7 日
统一控制台的功能配置选项	现在，在设置统一控制台时，Systems Manager 提供了功能配置选项。这些功能配置允许您选择是否设置默认主机管理配置（DHMC）、清单元数据收集和自动 SSM Agent 更新。有关更多信息，请参阅为组织设置 Systems Manager 统一控制台和为单个账户和区域设置 Systems Manager 统一控制台。	2025 年 5 月 7 日

更新了 Quick Setup 的托管式策略：AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy	Systems Manager 更新了托管式策略 `AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy` ，以提供标记 AWS Resource Access Manager 共享之资源的权限，以便进行即时节点访问。有关更多信息，请参阅 Systems Manager 对 AWS 托管式策略的更新。	2025 年 4 月 29 日
更新了 Quick Setup 的托管式策略：AWSQuickSetupManageJITNAResourcesExecutionPolicy	Systems Manager 更新了托管式策略 `AWSQuickSetupManageJITNAResourcesExecutionPolicy` ，以提供标记为即时节点访问而创建的 IAM 角色的权限。有关更多信息，请参阅 Systems Manager 对 AWS 托管式策略的更新。	2025 年 4 月 29 日
Systems Manager 引入即时节点访问	AWS Systems Manager 现在支持连接到托管式节点的即时访问方法。这种连接到托管式节点的新方法仅在需要时才提供对节点的临时、有时限的访问权限，无需长期权限，从而帮助提高安全性。此外，Systems Manager 还为 Windows Server 节点的 RDP 会话提供会话录制，以帮助您满足合规性要求、执行根本原因分析等。有关更多信息，请参阅使用 Systems Manager 进行即时节点访问。	2025 年 4 月 29 日

用于即时节点访问的新托管式策略	Systems Manager 发布了几项新的托管式策略，以支持即时节点访问。有关更多信息，请参阅 Systems Manager 对 AWS 托管式策略的更新。	2025 年 4 月 29 日
AWS 参数和密钥 Lambda 扩展的新版本	新版 AWS Parameters and Secrets Lambda 扩展现在适用于所有支持的区域和架构。	2025 年 4 月 23 日
新主题：“Patch Manager中的合规性是什么？”	AWS 或其他第三方未定义您的队列中托管节点的补丁合规性。新主题Patch Manager 中的合规性是什么？阐明了 Systems Manager 管理的节点的补丁合规性意味着已在节点上安装所有符合相关补丁基准中指定的批准标准的补丁。合规性由您定义，位于补丁基准中您配置的规则中，或位于 AWS 提供的、您为修补操作选择的预定义补丁基准中。	2025 年 4 月 11 日
AWS 参数和密钥 Lambda 扩展的新版本	现已推出 AWS 参数和密钥 Lambda 扩展的新版本。此外，还为以下区域添加了扩展支持： • 亚太地区（泰国）（ap-southeast-7） • 墨西哥（中部）（mx-central-1）	2025 年 4 月 3 日

新主题：配置合规性权限	作为最佳安全实践，建议您更新托管式节点使用的 AWS Identity and Access Management（IAM）角色，以限制该节点使用 PutComplianceItems API 操作的功能。该 API 操作会在指定资源（例如 Amazon EC2 实例或托管式节点）上注册合规性类型及其他合规性详细信息。有关更多信息，请参阅 Configuring permissions for Compliance。	2025 年 3 月 11 日
更新的主题：查看节点见解	在 Systems Manager 的统一控制台中，查看节点见解页面会报告组织或账户中的托管式及非托管式 EC2 实例数。我们已经阐明当 EC2 实例处于 `Stopped` 状态时，Systems Manager 会将其纳入“非托管”计数。有关更多信息，请参阅什么是非托管式实例？。	2025 年 2 月 25 日
更新的主题：验证 SSM Agent 的签名	适用于 Linux 实例的 AWS Systems Manager Agent（SSM Agent）RPM 和 Debian 安装程序包采用加密签名。您可以使用公钥验证代理软件包是否为未修改的原始包。如果文件有任何损坏或已更改，则验证将失败。您可以使用 RPM 或 GPG 验证安装程序包的签名。该主题包括 SSM Agent 的一个新的公有密钥。有关更多信息，请参阅验证 SSM Agent 签名。	2025 年 2 月 14 日

新主题：验证 Session Manager 插件的签名	适用于 Linux 实例的 Session Manager 插件 RPM 和 Debian 安装包是以加密方式签名的。您可以使用公有密钥验证插件二进制文件和软件包是否为未修改的原始包。如果文件有任何更改或损坏，则验证将失败。您可以使用 GNU Privacy Guard（GPG）工具验证安装包的签名。有关更多信息，请参阅验证 Session Manager 插件的签名。	2025 年 2 月 13 日
Session Manager 插件已更新	Session Manager 插件最近进行了更新，增加了以下增强功能：在 Dockerfile 中将 Go 版本升级到 1.23。更新了 README 中的版本配置步骤。有关 Session Manager 插件的更多信息，请参阅安装 AWS CLI 的 Session Manager 插件。	2025 年 2 月 6 日
已发布 Automation API 操作的 TPS 配额	我们已经发布了多个与 Systems Manager Automation 相关的 API 操作的每秒事务数（TPS）配额（旧称限制）。有关信息，请参阅《Amazon Web Services 一般参考》中的 Systems Manager service quotas。	2025 年 1 月 21 日

术语变更：Systems Manager 功能更名为 Systems Manager 工具	此前，构成 Systems Manager 的功能单元被称为功能，例如 Automation 功能、Parameter Store 功能。这些功能单元现在被称为工具，例如 Automation 工具和 Parameter Store 工具。本用户指南已就该变更进行了更新。	2025 年 1 月 10 日
Patch Manager支持其他 macOS 版本（Amazon EC2 AMI）	Patch Manager现在支持 macOS 版本 12.7、13.6、13.7 以及 15.x。有关Patch Manager支持的操作系统和版本的完整列表，请参阅Patch Manager支持的操作系统。	2025 年 1 月 8 日
SSM Agent 和Patch Manager 支持其他版本：CentOS Stream、Ubuntu Server 和 Windows Server	Patch Manager现在支持 CentOS Stream 9、Ubuntu Server 24.10 和 Windows Server 2025。SSM Agent 现在支持 Ubuntu Server 24.10 和 Windows Server 2025。（之前已发布对 CentOS Stream 9 的代理支持。）有关支持的操作系统和版本的完整列表，请参阅以下主题： • Systems Manager 支持的操作系统 • Patch Manager 支持的操作系统	2024 年 11 月 22 日

新主题：Parameter Store SecureString 参数的 AWS KMS 加密	请阅读以下主题，了解 AWS Systems Manager Parameter Store 如何使用 AWS Key Management Service 加密 Parameter Store 中 `SecureString` 参数的值： • Parameter Store 中 SecureString 参数的 AWS KMS 加密	2024 年 11 月 22 日
Systems Manager 的新托管式策略和更新的托管式策略	为了支持 Systems Manager 的新功能，我们发布了多个新的托管式策略来支持新的 Systems Manager 配置和操作，并更新其他托管式策略。有关更多信息，请参阅 Systems Manager 更新了 AWS 托管式策略。	2024 年 11 月 21 日

Systems Manager 简化的新节点管理体验	AWS Systems Manager 发布了一种新的统一控制台体验，用于跨账户和区域大规模管理节点。您现在可以从单个位置查看您组织的 AWS 账户和区域内的所有托管和非托管节点。您还可以识别、诊断并修复非托管节点。Systems Manager 现在还与 Amazon Q 开发者版（Amazon Q）集成，通过输入自然语言提示，您可以从 AWS Management Console中的任意位置查看和控制您的节点。在此版本中，您现在还可以使用 AWS Organizations 让委派管理员账户能够从中心角度管理整个组织的节点。有关更多信息，请参阅以下主题： • 什么是 AWS Systems Manager ？ • 设置 AWS Systems Manager • 使用 AWS Systems Manager 完成节点任务	2024 年 11 月 21 日
Session Manager插件错误修复	Session Manager插件最近进行了更新，修复了以下错误：回滚了向 OpenDataChannel 请求添加凭证的更改。	2024 年 11 月 20 日

Session Manager插件增强	此版本已于 2024 年 11 月 20 日弃用。 Session Manager插件最近进行了更新，包含以下增强功能。 • 为 OpenDataChannel 请求添加了凭证。 • 升级了 `testify` 和 `objx` 依赖的软件包。	2024 年 11 月 6 日
macOS 的其他操作系统版本支持	Systems Manager 现在支持 macOS 操作系统版本 15.x (Sequoia)（仅限 EC2 实例）。有关所有受支持的操作系统和版本的列表，请参阅 Systems Manager 支持的操作系统。	2024 年 11 月 6 日
Patch Manager：已批准补丁和已拒绝补丁列表的程序包名称格式	对于多个操作系统，我们更新并扩展了您可以在补丁基准中的已批准的补丁和已拒绝的补丁列表中指定的软件包名称的格式列表。有关信息，请参阅 Amazon Linux 1、Amazon Linux 2、Amazon Linux 2022、Amazon Linux 2023、CentOS、Oracle Linux 和 Red Hat Enterprise Linux (RHEL) 的程序包名称格式。	2024 年 11 月 1 日

Patch Manager 的其他操作版本支持	Patch Manager现在支持其他版本的 Oracle Linux（8.10 和 9.4）和 Ubuntu Server（23.10 和 24.04）。有关所有受支持的操作系统和版本的列表，请参阅以下主题： • Systems Manager 支持的操作系统 • Patch Manager 支持的操作系统	2024 年 11 月 1 日
SSM Agent支持其他版本：CentOS StreamOracle Linux、和 Ubuntu Server	除了之前支持的版本外，SSM Agent 现在还支持 CentOS Stream 9、Oracle Linux 8.10 和 9.4 以及 Ubuntu Server 24.04 LTS。有关 Systems Manager 支持的操作系统和版本的完整列表，请参阅 Systems Manager 支持的操作系统。	2024 年 10 月 25 日
Session Manager 插件增强	增加了对通过 OpenDataChannel 请求传递插件版本的支持。	2024 年 10 月 10 日
新增：查看通过 Fleet Manager 建立的 RDP 连接的详细信息	现在，您可以在 AWS 账户 中查看您的用户所建立的远程桌面协议连接的相关信息。有关更多信息，请参阅查看有关当前和已完成连接的信息。	2024 年 10 月 10 日

Patch Manager 现支持 SLES 版本 15.6	针对 SUSE Linux Enterprise Server（SLES）15.6 的支持已发布。您现在可以使用 Patch Manager 修补 SLES 15.6 计算机。有关 Patch Manager 支持的操作系统和版本的完整列表，请参阅Patch Manager 支持的操作系统。	2024 年 9 月 29 日

AWS 参数和密钥 Lambda 扩展的新版本	现已推出 AWS 参数和密钥 Lambda 扩展的新版本。亚太地区（马来西亚）区域（ap-southeast-5）引入了对所有架构的支持。此外，还为以下区域添加了 ARM64 和 Mac with Apple silicon 架构扩展支持： • 亚太地区（海得拉巴）(ap-south-2) • 亚太地区（墨尔本）(ap-southeast-4) • 加拿大西部（卡尔加里）（ca-west-1） • 欧洲（苏黎世）(eu-central-2) • 欧洲（西班牙）(eu-south-2) • 中东（阿联酋）(me-central-) • 中国（北京）（cn-north-1） • 中国（宁夏）（cn-northwest-1） • 以色列（特拉维夫）（il-central-1） • AWS GovCloud（美国东部）(us-gov-east-1) • AWS GovCloud（美国西部）(us-gov-west-1)	2024 年 9 月 19 日
新主题：使用 AWS CLI 配置维护时段的权限	使用 AWS CLI 配置维护时段权限主题提供了有关创建自定义服务角色（及其策略）以代表用户运行维护时段任务的说明。	2024 年 8 月 19 日

SSM Agent 和 Patch Manager 支持其他版本：AlmaLinux、Oracle Linux 和 Rocky Linux	除了早期支持的版本，SSM Agent 和 Patch Manager 现在还支持 AlmaLinux 和 Rocky Linux 的版本 8.10、9.3 和 9.4，以及 Oracle Linux 的版本 9.3。有关支持的操作系统和版本的完整列表，请参阅以下主题： • Systems Manager 支持的操作系统 • Patch Manager 支持的操作系统	2024 年 8 月 14 日
Parameter Store 支持的新的 IAM 策略条件：ssm:Policies	使用 IAM 策略新支持的条件 `ssm:Policies` ，您可以阻止实体创建或更新包含参数策略的参数。有关更多信息，请参阅以下主题： • ssm:Policies：阻止创建或更新使用参数策略的参数 • 《服务授权参考》中的 AWS Systems Manager 的条件键。	2024 年 8 月 14 日
更新了 Quick Setup 的托管策略：SSMQuickSetupRolePolicy	Systems Manager 已更新托管策略 `SSMQuickSetupRolePolicy` ，以提供对其他 AWS CloudFormation 堆栈集的访问权限。有关更多信息，请参阅 Systems Manager 对 AWS 托管式策略的更新。	2024 年 8 月 13 日

支持使用 Terraform 预置和管理 Systems Manager 资源	我们已将 HashiCorp Terraform 添加到面向 Systems Manager 的受支持第三方集成列表中。Terraform 是一款开源基础设施即代码（IaC）软件工具，它提供命令行界面（CLI）工作流来管理各种云服务。您可以使用 Terraform 来预置和管理许多常用的 Systems Manager 资源和数据来源。有关此集成以及其他面向 Systems Manager 的第三方集成的信息，请参阅与其他产品和服务的集成。	2024 年 8 月 1 日
全新的 Quick Setup 控制台体验和 API	Systems Manager Quick Setup 功能已发布新的控制台体验和 API。现在，您可以使用控制台、AWS CLI、AWS CloudFormation 和 SDK 与此 API 进行交互。您可以使用 Quick Setup 控制台选择使用新控制台。有关注册全新 Quick Setup 体验的更多信息，请参阅快速设置功能入门。有关通过 Quick Setup API 提供的 API 操作的更多详细信息，请参阅《Quick Setup API 参考》https://docs.aws.amazon.com/quick-setup/latest/APIReference/Welcome.html。	2024 年 8 月 1 日

新主题：自定义补丁基准中已拒绝的补丁列表选项	对于 Patch Manager 中使用自定义补丁基准的修补操作，我们已经阐明向已拒绝的补丁列表中添加的补丁分配操作 `Allow as dependency` 时的行为。由于 Windows Server 不支持补丁依赖项的概念，因此会跳过尚未安装在托管节点上的补丁。已安装在节点上的补丁会被分配状态 `INSTALLED_REJECTED` 。有关更多信息，请参阅自定义补丁基准中的“已拒绝的补丁”列表选项和其他操作系统的补丁合规性值。	2024 年 7 月 23 日
新主题：配置 SSM Agent 以与美国联邦信息处理标准（FIPS）结合使用	我们已提供配置 SSM Agent 以与美国联邦信息处理标准（FIPS）结合使用的说明。有关更多信息，请参阅配置 SSM Agent 以与美国联邦信息处理标准（FIPS）结合使用。	2024 年 7 月 22 日
更新：阐明对 Fleet Manager 用户名中 @ 符号的支持	如果 IAM Identity Center 用户名包含一个或多个 @ 符号，则 Fleet Manager RDP 会忽略第一个 @ 符号及其后面的所有字符，无论 @ 是否引入电子邮件地址的域部分。有关 Fleet Manager RDP 连接中支持的用户名字符的更多信息，请参阅对远程桌面连接进行身份验证。	2024 年 7 月 21 日

更新的托管式策略：AmazonSSMManagedEC2InstanceDefaultPolicy	Systems Manager 通过提供内联语句 ID（Sid）来阐明每条策略语句的用途，以此更新托管式策略 `AmazonSSMManagedEC2InstanceDefaultPolicy` 。有关更多信息，请参阅 Systems Manager 对 AWS 托管式策略的更新。	2024 年 7 月 18 日

对用于 Patch Manager 修补操作的 AWS 托管存储桶进行名称更改	AWS 拥有并维护许多 Amazon S3 存储桶，SSM Agent 在执行各种 Patch Manager 修补操作的过程中访问这些存储桶。这些 S3 存储桶可以公开访问，默认情况下，SSM Agent 使用 HTTP 调用连接到这些存储桶。但是，如果您正在 Systems Manager 操作中使用虚拟私有云（VPC）端点，则必须在 Systems Manager 的 Amazon EC2 实例配置文件中提供显式权限，或者在混合和多云环境中的非 EC2 计算机的服务角色中提供。否则，资源无法访问这些公有存储桶。大多数情况下，我们会更改这些存储桶的名称。例如，对于修补操作，存储桶 `aws-patchmanager-macos-us-east-2` 替换为 `aws-patchmanager-macos-us-east-2-552881074` ，存储桶 `aws-patch-manager-us-east-2-552881074` 替换为 `aws-ssm-us-east-2`。有关更多信息，请参阅以下主题： • SSM Agent 与 AWS 托管 S3 存储桶进行通信 • 参考：用于修补操作的 Amazon S3 存储桶	2024 年 7 月 18 日

Quick Setup 的新服务相关角色	Systems Manager 已发布新的服务相关角色 `AWSServiceRoleForSSMQuickSetup` 。Systems Manager 使用此角色来检查使用 Quick Setup 设置的资源的配置运行状况，确保参数和已配置资源的一致使用，并在检测到偏差时修复资源。与此角色关联的托管式策略是 `SSMQuickSetupRolePolicy` 。有关更多信息，请参阅适用于 Systems Manager 的 AWSServiceRoleForSSMQuickSetup 服务相关角色权限。	2024 年 7 月 3 日
Quick Setup 配置类型的新托管式策略	Systems Manager 又发布了 12 个新的托管式策略，以支持各种 Quick Setup 配置类型和流程。有关更多信息，请参阅 Systems Manager 对 AWS 托管式策略的更新。	2024 年 7 月 3 日
支持 RHEL 8.10 和 9.4	Systems Manager 和 Patch Manager 现在支持 Red Hat Enterprise Linux 版本 8.10 和 9.4。有关更多信息，请参阅支持的操作系统和计算机类型和 Patch Manager 支持的操作系统。	2024 年 6 月 26 日

Patch Manager 支持 8.8 和 8.9 版本：AlmaLinux、Oracle Linux 和 Rocky Linux	除了早期的 8.x 版本，Patch Manager 现在还支持 8.8 和 8.9 版本的 AlmaLinux、Oracle Linux 和 Rocky Linux。有关 Patch Manager 支持的操作系统和版本的完整列表，请参阅 Patch Manager 支持的操作系统。	2024 年 6 月 17 日
macOS Amazon EC2 AMIs 的新公有参数	已发布公有参数以支持 macOS Amazon Elastic Compute Cloud 实例的 Amazon Machine Images 有关更多信息，请参阅以下主题。 • 查找公有参数 • 为 macOS 调用 AMI 公有参数 • 《Amazon EC2 用户指南》中的启动 Mac 实例 • 《Amazon EC2 用户指南》中的 Amazon EC2 实例 • 《Amazon EC2 Auto Scaling 用户指南》中的在启动模板中使用 AWS Systems Manager 参数代替 AMI ID	2024 年 6 月 17 日

更新：/aws/service/global-infrastructure 参数路径的区域可用性	我们已经阐明了可以从哪些商业区域查询 /aws/service/global-infrastructure 公共参数路径，以及如果您在其他商业 AWS 区域 工作时该如何对该路径进行查询。有关更多信息，请参阅调用 AWS 服务、区域、端点、可用区、本地区域和 Wavelength 区域的公有参数。	2024 年 6 月 12 日
新增：代码示例章节	新章节使用 AWS SDK 的 Systems Manager 的代码示例提供了使用不同 SDK 语言的示例，说明如何使用 Systems Manager 服务。	2024 年 5 月 8 日
对 ec2messages:* 端点支持的变更	对于在 2024 年或之后推出的 AWS 区域，SSM Agent 不支持 ec2messages:* 端点将状态和执行信息发送回 Systems Manager 服务。以下区域中的账户必须使用 ssmmessages:* 。在 2024 年之前推出的区域中，仍然同时支持 ssmmessages:* 和 ec2messages:* ，但建议现在仅使用 ssmmessages:* 端点（Amazon Message Gateway Service）。此时，您可以放心地删除策略中的 ec2messages:* 权限。有关更多信息，请参阅使用 SSM Agent 和与代理相关的 API 操作（ssmmessages 和 ec2messages 端点）。	2024 年 5 月 3 日

可用于在自动化运行手册中运行脚本的其他运行时	`aws:executeScript` 操作现在支持 Python 3.9、3.10 和 3.11 运行时。有关如何使用此操作的更多信息，请参阅 `aws:executeScript` 。	2024 年 4 月 23 日
支持 8.8 和 8.9 版本：AlmaLinux、Oracle Linux 和 Rocky Linux	除了早期的 8.x 版本，Systems Manager 现在还支持 AlmaLinux、Oracle Linux 和 Rocky Linux 的 8.8 和 8.9 版本。有关支持的操作系统和版本的完整列表，请参阅 Systems Manager 支持的操作系统。	2024 年 4 月 22 日
Patch Manager：更改为修补状态“INSTALLED_PENDING_REBOOT”	以前，只能将 Patch Manager 安装的补丁标记为 `INSTALLED_PENDING_REBOOT` 。在 Patch Manager 之外安装的补丁永远不会具备该状态。现在，`INSTALLED_PENDING_REBOOT` 可以应用于自上次重启托管式节点以来应用于该节点的任何补丁。这包括通过选定 `NoReboot` 选项由 Patch Manager 安装的补丁，以及节点最近一次重启后在 Patch Manager 之外安装的补丁。有关所有 Patch Manager 修补状态值的描述，请参阅了解补丁合规性状态值。	2024 年 4 月 16 日

支持 RHEL 8.9 和 9.3	除了早期的 8.x 和 9.x 版本，Systems Manager（包括 Patch Manager）现在还支持 Red Hat Enterprise Linux（RHEL）版本 8.9 和 9.3。	2024 年 3 月 26 日
主题更新：适用于 AWS Systems Manager 的 AWS 托管策略	主题适用于 AWS Systems Manager 的 AWS 托管策略提供了有关自 2021 年 3 月 12 日以来推出或更新的四个 Systems Manager 托管策略的信息。我们在本主题中添加了一个小节，其中包含有关在该日期之前创建或上次更新的其他 12 个用于 Systems Manager 的托管策略的信息。有关详细信息，请参阅 Systems Manager 的其他托管策略。	2024 年 3 月 1 日
Parameter Store 现在支持跨账户共享	现在，您可以通过设置资源共享，在 AWS 账户 之间或您的 AWS 组织内安全高效地共享高级参数。通过资源共享，您能够集中管理应用程序配置，减少与您拥有的每个账户共享参数的运维开销。可以使用 Parameter Store 控制台、AWS RAM 控制台或 AWS CLI 跨账户共享参数。有关更多信息，请参阅使用共享参数。	2024 年 2 月 21 日

自动化操作增强功能	现在，您可以在 `aws:approve` 操作中使用 `onFailure` 和 `isCritical` 属性。有关 `aws:approve` 操作的更多信息，请参阅 aws:approve – 暂停自动化以进行手动批准。	2024 年 2 月 12 日
Patch Manager 的其他操作版本支持	已添加到 Patch Manager 支持的操作系统 列表中。添加了对以下版本的支持： • Debian Server 11.x 和 12.x • macOS 14.x（Sonoma） • SUSE Linux Enterprise Server（SLES）15.5 • Ubuntu Server 23.04	2024 年 1 月 4 日
使用 Application Manager 控制台配置 SSM Agent 自动更新	现在，您可以使用 Application Manager 控制台来为应用程序实例自动更新 SSM Agent。有关更多信息，请参阅 使用应用程序实例。	2023 年 12 月 21 日
更新了在混合环境和多云环境中注册非 Amazon EC2 计算机的流程	Systems Manager 现在提供了 `ssm-setup-cli` ，可助您在混合环境和多云环境中注册非 Amazon Elastic Compute Cloud（Amazon EC2）计算机。有关更多信息，请参阅 如何在混合 Linux 节点上安装 SSM Agent 和如何在混合 Windows 节点上安装 SSM Agent。	2023 年 12 月 20 日

使用 Fleet Manager 管理 Amazon EBS 卷	现在，您可以使用 Fleet Manager（AWS Systems Manager 中的一项工具）来管理托管式实例上的 Amazon Elastic Block Store 卷。例如，您可以初始化 EBS 卷，格式化分区，然后挂载该卷以使其可供使用。有关更多信息，请参阅 EBS 卷管理。	2023 年 12 月 14 日
Session Manager 插件增强	增加了对将 StartSession API 响应作为环境变量传递给会话管理器插件的支持。	2023 年 12 月 4 日
自动化运行手册的全新视觉对象设计体验	现在，您可以使用 Systems Manager Automation 开发的全新视觉对象设计体验来创建和编辑运行手册。视觉对象设计体验提供低代码的拖放界面，因此您可以更轻松地创建和编辑运行手册。有关更多信息，请参阅自动化运行手册的视觉对象设计体验。	2023 年 11 月 26 日

运行手册的全新 Systems Manager Automation 操作、数据元素和功能增强	现在，您可以使用 aws:loop 操作在运行手册中循环执行多个操作。该新操作支持 do while 和 for each 样式循环。此外，使用新的变量数据元素，您可以在运行手册的上下文中动态定义、引用和更新值。要更新运行手册中变量的值，请使用新 aws:updateVariable 操作。自动化还增加了对输出的动态数据类型转换的支持。这意味着，如果输出的值与您指定的数据类型不匹配，自动化会尝试转换该数据类型。例如，如果返回的值是 Integer，但指定的 Type 是 String，则最终输出值为 String 值。最后，自动化现在支持选择器的 JSONPath 筛选条件表达式。有关更多信息，请参阅以下主题： • aws:loop – 迭代自动化中的步骤 • aws:updateVariable – 更新运行手册变量的值 • 数据元素和参数 – 顶级数据元素 • 使用操作输出作为输入。 • 在运行手册中使用 JSONPath。	2023 年 11 月 17 日

更新了对 Remote Desktop Protocol（RDP）连接的区域支持	Fleet Manager 远程桌面由 Amazon DCV 提供支持，可让您直接从 Systems Manager 控制台安全连接到 Windows Server 实例。已为 Fleet Manager 远程桌面连接启用了以下三个其他区域： • 非洲（开普敦）(af-south-1) • 亚太地区（雅加达）（ap-southeast-3） • 以色列（特拉维夫）（il-central-1）	2023 年 11 月 15 日
Patch Manager：扩展了对 RHEL 和 macOS 的操作系统版本支持	Patch Manager 现在增加支持以下操作系统版本： • Red Hat Enterprise Linux：版本 8.8 • macOS：11.5–11:7（Big Sur） • macOS：12.0–12.6（Monterey） • macOS：13.0–13.5（Ventura）	2023 年 10 月 23 日
新增 OpsCenter API – DeleteOpsItem	OpsCenter 现在提供了可用于删除单个 OpsItems 的 DeleteOpsItem API。有关更多信息，请参阅《AWS Systems Manager API 参考》中的 DeleteOpsItem。	2023 年 10 月 20 日

新增 Quick Setup 配置类型：整个组织的 SSM Agent 更新	借助新推出的配置类型默认主机管理配置，组织管理员（如中所 AWS Organizations 定义）可以提示自动检查和更新组织账户和区域中所有 EC2 实例的 SSM Agent。有关更多信息，请参阅 Default Host Management for an organization。	2023 年 10 月 16 日
由 CloudWatch Application Insights 为 OpsItems 创建的新标题和描述格式	由 CloudWatch Application Insights 为 OpsItems 创建的标题和描述将于 2023 年 10 月 16 日更改为经过改进的格式。要查看新格式，请参阅 Amazon CloudWatch Application Insights。	2023 年 9 月 29 日
支持 Fleet Manager RDP 连接中的多种显示分辨率	使用 Fleet Manager 中的远程桌面协议（RDP）选项连接到 Windows Server 托管式节点时，现在可以选择显示分辨率。此前，所有连接都使用固定的 720P（1366 x 768）分辨率。现在，您可以为每个连接从以下选项中进行选择： • 自动适应（根据检测到的屏幕尺寸确定最佳分辨率） • 1920 x 1080 • 1400 x 900 • 1366 x 768 • 800 x 600 有关更多信息，请参阅使用远程桌面连接到托管节点。	2023 年 9 月 22 日

新主题：补丁策略操作中的随机补丁基准 ID	我们添加了内容来描述 Quick Setup 补丁策略如何在每次运行补丁策略操作时使用 AWS-RunPatchBaseline SSM 命令文档中的 BaselineOverride 参数为补丁基准生成随机 ID。有关更多信息，请参阅补丁策略操作中的随机补丁基准 ID。	2023 年 9 月 22 日
管理 OpsItems 的全新运营洞察	OpsCenter 现在包含一个名为生成最多 OpsItem 资源的操作洞察。当一个 AWS 资源具有超过 10 个开放 OpsItems 时，就会生成这种类型的洞察。利用此洞察来查找有问题的资源。使用洞察中的 AWS-BulkResolveOpsItems 运行手册快速解决与资源关联的 OpsItems。有关更多信息，请参阅分析运营洞察以减少 OpsItems。	2023 年 9 月 22 日
GPG 公有密钥已更新	已创建新的公有密钥验证 SSM Agent 签名。有关更多信息，请参阅验证 SSM Agent 签名。	2023 年 9 月 5 日

添加了对 AlmaLinux、Oracle Linux、RHEL 和 Rocky Linux 其他版本的支持	支持 AWS Systems Manager 和 Patch Manager 的操作系统列表已更新，以反映对以下其他操作系统版本的支持： • AlmaLinux：9.2 • Oracle Linux：8.7 和 9.2 • Red Hat Enterprise Linux（RHEL）：8.7、9.1 和 9.2 • Rocky Linux：8.6 和 8.7、9.0-9.2	2023 年 8 月 30 日
OpsCenter 在 OpsItem 描述字段中添加了对 Markdown 格式的支持。	OpsCenter 现在支持 OpsItem 描述字段中的 Markdown 格式。支持以下类型的 Markdown 格式： • 段落 • 行距 • 水平线 • 标题 • 文本格式设置 • 链接 • Lists 有关更多信息，请参阅《AWS Management Console 入门指南》中的在控制台中使用 Markdown。	2023 年 8 月 18 日

AWS 参数和密钥 Lambda 扩展的新版本	现已推出 AWS 参数和密钥 Lambda 扩展的新版本。此外，还对亚太地区（墨尔本）（ap-southeast-4）和以色列（特拉维夫）（il-central-1）区域（仅限 x86_64 和 x86 架构）添加了扩展支持。有关更多信息，请参阅在 AWS Lambda 函数中使用 Parameter Store 参数。	2023 年 8 月 16 日
更新：添加了有关 Quick Setup 补丁策略存储桶所需权限的信息	创建补丁策略时，Quick Setup 会创建一个 Amazon S3 存储桶，其中包含一个名为 `baseline_overrides.json` 的文件。此文件存储了有关为补丁策略指定的补丁基准的信息。配置补丁策略时，您可以选择将所需的 IAM policy 添加到附加于实例的现有实例配置文件中复选框。如果您选择不选择此选项，则必须手动向某些资源提供访问此存储桶的权限，否则您的策略操作可能会失败。有关更多信息，请参阅以下主题： • 补丁策略 S3 存储桶的权限 • 问题：`baseline_overrides.json` 出现“Invoke-PatchBaselineOperation：Access Denied”错误或“Unable to download file from S3”错误	2023 年 7 月 6 日

使用 Quick Setup 配置 OpsCenter，实现多账户 OpsItem 管理	OpsCenter 的 Quick Setup 可帮助您完成以下跨账户管理 OpsItems 的任务： • 指定委派管理员账户 • 创建必需的 AWS Identity and Access Management（IAM）policy 和角色 • 指定一个 AWS Organizations 组织或成员账户的子集，委派管理员可以在其中跨账户管理 OpsItems 有关更多信息，请参阅（可选）使用 Quick Setup 配置 OpsCenter，以跨账户管理 OpsItems。	2023 年 6 月 19 日
使用 Quick Setup 更新 Amazon EC2 启动代理	现在可以允许 Systems Manager 每 30 天检查实例上安装的启动代理的新版本。如果有新版本可用，则 Systems Manager 会在实例上更新代理。有关更多信息，请参阅 Quick Setup 主机管理。	2023 年 6 月 19 日
Patch Manager 现在支持 Ubuntu Server 22.04 LTS	现在可以使用 Patch Manager 来修补 Ubuntu Server 22.04 LTS 节点。与其他受支持的 Ubuntu Server 版本一样，版本 22.04 LTS 使用 AWS 托管 `AWS-UbuntuDefaultPatchBaseline` 补丁基准。	2023 年 5 月 15 日

Systems Manager 现在支持 AlmaLinux，包括 Patch Manager	现在可以使用 Systems Manager 来管理 AlmaLinux 8.3-8.7；9.0-9.1 节点。许多适用于 RHEL 8 的修补规则也适用于 AlmaLinux。AlmaLinux 使用了新的 `AWS-DefaultAlmaLinuxPatchBaseline` 。有关更多信息，请参阅以下主题： • 在 AlmaLinux 实例上手动安装 SSM Agent • 如何选择安全性补丁 • 如何安装补丁 • 补丁基准规则在 AlmaLinux、RHEL 和 Rocky Linux 上的工作原理。	2023 年 5 月 8 日
使用 Quick Setup 部署 EC2Launch v2 代理	现在可以使用 Quick Setup 部署 EC2Launch v2 代理。有关更多信息，请参阅使用 Quick Setup 部署 Distributor 软件包。	2023 年 4 月 13 日

Systems Manager 现在支持 Amazon Linux 2023	Systems Manager 现在支持新的 Amazon Linux 2023（AL2023）EC2 实例类型，包括对 Patch Manager 操作的支持。适用于 Amazon Linux 2 的许多补丁规则也适用于 Amazon Linux 2023。（Patch Manager 还继续支持 Amazon Linux 2022 的预览版。）有关更多信息，请参阅以下主题： • 如何选择安全性补丁 • 如何安装补丁 • 补丁基准规则在 Amazon Linux 1、Amazon Linux 2、Amazon Linux 2022 和 Amazon Linux 2023 上的工作原理	2023 年 3 月 23 日
修订了 Amazon EC2 实例的设置内容	我们已经修订了 Amazon EC2 实例的设置内容。现在建议使用新发布的“默认主机管理配置”以获得实例权限。有关更多信息，请参阅配置 Systems Manager 所需的实例权限。	2023 年 2 月 15 日
使用“默认主机管理配置”实现自动实例管理	现在，您可以使用 Systems Manager 自动管理整个 AWS 区域 中的 Amazon EC2 实例。有关更多信息，请参阅默认主机管理配置。	2023 年 2 月 15 日

将 SSM 文档添加到您的收藏夹	为帮助您查找常用 SSM 文档，现在可将文档添加到您的收藏夹。每种文档类型、每种 AWS 账户 和 AWS 区域 最多可以收藏 20 份文档。您可以从 Systems Manager 文档控制台中选择、修改和查看您的收藏夹。有关更多信息，请参阅将文档添加到您的收藏夹。	2023 年 2 月 7 日
使用 Change Calendar 为自动化实施更改控制	通过将自动化与 Change Calendar 集成，现在您可以为 AWS 账户 中的所有自动化实施更改控制。有关更多信息，请参阅为自动化实施更改控制。	2023 年 1 月 24 日
新的 Change Manager 批准工作流	Change Manager 批准工作流现在支持逐级批准，而不是逐行批准。以前，您添加到批准级别的每个审批者都必须批准更改请求。否则，该级别不会获得批准。现在，您可以指定该级别需要多少次批准，并可添加相应数量或更多的审批者。例如，对于某一级别，您可能需要三次批准，但最多可以指定五个审批者。其中任何三个审批者的批准都足以批准该级别。有关更多信息，请参阅关于更改模板中的批准。	2023 年 1 月 23 日

新增：使用 Quick Setup 中的补丁策略为整个组织配置修补	借助Quick Setup（Systems Manager 中的一项工具），现在您可以创建由Patch Manager提供支持的补丁策略。补丁策略定义了在自动修补托管式节点时使用的计划和补丁基准。使用单一补丁策略配置，您可以为贵组织中所有区域的所有账户定义修补、仅为所选的账户和区域定义修补，或为单个账户区域对定义修补。有关更多信息，请参阅以下主题。 • 使用 Quick Setup 补丁策略 • 使用 Quick Setup 补丁策略自动进行组织范围的修补	2022 年 12 月 22 日

Application Manager 与 Amazon EC2 集成，可在应用程序上下文中显示您的实例信息。	Application Manager 以图形格式显示所选应用程序的实例状态、情况和 Amazon EC2 Auto Scaling 运行状况。Instances（实例）选项卡还包括一个表，其中包含应用程序中每个实例的以下信息：
	• 实例状态（待处理、正在停止、正在运行、已停止）
	• SSM Agent 的 Ping 状态
	• 在实例上处理的最新 Systems Manager Automation 运行手册的状态和名称
	• 每个州的 Amazon CloudWatch Logs 警报数量。
	  • ALARM – 指标或表达式超出定义的阈值。
	  • OK – 指标或表达式在定义的阈值范围内。
	  • INSUFFICIENT_DATA （数据不足）– 告警刚刚启动，指标不可用，或者指标没有足够的数据以确定告警状态。
	• 父组和单个自动扩缩组的自动扩缩组运行状况
		2022 年 12 月 22 日

使用 Quick Setup 调度 Amazon EC2 实例的启动和停止	您现在可以部署资源调度程序解决方案，使用 Quick Setup 自动启动和停止 Amazon EC2 实例。有关更多信息，请参阅资源调度程序。	2022 年 12 月 19 日
OpsCenter 现在支持跨账户使用 OpsItems	OpsCenter 支持在会话期间从管理账户（AWS Organizations 管理账户或 Systems Manager 委派管理员账户）和成员账户与 OpsItems 合作。配置后，用户可以执行以下类型的操作： • 在成员账户中创建、查看并更新 OpsItems • 在成员账户中查看 OpsItems 中指定的 AWS 资源的详细信息。 • 启动 Systems Manager Automation 运行手册以修复成员账户中的 AWS 资源问题 有关更多信息，请参阅设置 OpsCenter 以便跨账户使用 OpsItems。	2022 年 11 月 16 日

使用 AWS CloudTrail Lake 跟踪 Change Manager 变更请求的详细信息	现在，您可以使用 AWS CloudTrail Lake 中的事件数据存储来捕获和查看有关在 Change Manager 中为您的组织或账户变更请求的详细信息。此信息包括有关创建变更请求的用户身份、发出请求的 IP 地址、进行请求的 AWS 区域、目标资源等供审批的详细信息。有关更多信息，请参阅 Monitoring your change request events（监控您的变更请求事件）和查看变更请求的详细信息、任务和时间表。	2022 年 11 月 11 日
使用 CloudWatch 警报进行其他 Systems Manager 自动化任务	现在，您可以使用 CloudWatch 警报在多个账户和区域运行自动化时进行其他控制。通过将指标或复合 CloudWatch 警报应用于自动化，您可以根据定义的指标控制自动化何时停止。有关将 CloudWatch 警报应用于在多个账户和地区上运行的自动化的详细信息，请参阅 Run an automation in multiple Regions and accounts (console) [在多个区域和账户中运行自动化（控制台）]	2022 年 11 月 9 日

更新：“在 AWS Lambda 函数中使用 Parameter Store 参数”	我们提供了其他信息，以帮助您使用 AWS 参数和密钥 Lambda 扩展来检索参数值并将其缓存以供将来在 Lambda 函数中使用。使用 Lambda 扩展可以通过减少对 Parameter Store 的 API 调用次数来降低成本。有关更多信息，请参阅在 AWS Lambda 函数中使用 Parameter Store 参数。	2022 年 10 月 25 日
使用 CloudWatch 警报进行其他 Systems Manager 任务控制	现在，您可以使用 CloudWatch 警报在运行自动化和命令时实施额外的控制。当 CloudWatch 警报注册到 State Manager 关联或维护窗口任务时，还可以将其添加到自动化或命令中。通过将复合 CloudWatch 警报应用于自动化或命令，您可以根据定义的指标控制自动化或命令何时停止。有关将 CloudWatch 警报应用于自动化或命令的更多信息，请参阅以下步骤： • 如何选择安全性补丁 • 如何安装补丁 • 补丁基准规则在 Amazon Linux 1、Amazon Linux 2 和 Amazon Linux 2022 上的工作原理。	2022 年 9 月 26 日

使用 CloudWatch 警报进行其他 Systems Manager 任务控制	现在，您可以使用 CloudWatch 警报在运行自动化和命令时实施额外的控制。当 CloudWatch 警报注册到 State Manager 关联或维护窗口任务时，还可以将其添加到自动化或命令中。通过将复合 CloudWatch 警报应用于自动化或命令，您可以根据定义的指标控制自动化或命令何时停止。有关将 CloudWatch 警报应用于自动化或命令的更多信息，请参阅以下步骤： • 运行简单的自动化 • 从控制台运行命令 • 创建关联 • 为维护时段分配任务	2022 年 9 月 26 日
明确高级实例套餐要求	根据客户反馈，我们已在配置实例套餐中明确了需要激活高级实例套餐的情况。	2022 年 9 月 21 日
使用 Quick Setup 部署 Amazon CloudWatch 代理	现在可以使用 Quick Setup 部署 Amazon CloudWatch 代理。有关更多信息，请参阅使用 Quick Setup 部署 Distributor 软件包。	2022 年 9 月 20 日

对于允许 EC2 实例元数据的情况，补丁组现在支持“PatchGroup”键	对于允许在 EC2 实例元数据中使用标签的情况，创建的标签键不能包含任何空格。以前，这会阻止客户将他们的一些 EC2 实例添加到 Patch Manager 中的补丁组，因为标签键 `Patch Group` 必须应用到实例。Patch Manager 现在支持 `Patch Group`（带空格）和 `PatchGroup`（不带空格）作为标签键，用于标识补丁组的实例。允许在实例元数据中使用标签的 EC2 实例现在可以添加到 Patch Manager 中的补丁组中。有关更多信息，请参阅关于补丁组。	2022 年 8 月 31 日
新主题：“如何计算软件包发布日期和更新日期”	在由 AWS 管理的补丁基准中，新补丁将在发布或更新后 7 天自动批准。在您创建的自定义补丁基准中，您可以选择指定发布或更新后等待多少天再自动批准安装。对于 Amazon Linux 1 和 Amazon Linux 2，有多种因素会影响最新发布日期和更新日期的计算方式。为了帮助您避免在选择自动批准延迟时出现意外结果，在主题如何计算软件包发布日期和更新日期中对这些因素进行了解释。	2022 年 8 月 24 日

更新内容：修补 AMI 并更新自动扩缩组	我们更新了为自动扩缩组更新 AMIs 演练，以使用启动模板而不是启动配置。此外，我们还在运行手册内容中实施了最新的自动化操作和运行时。	2022 年 6 月 22 日
Change Manager：阻止用户创建可自动批准的请求	您可以在 Change Manager 中配置更改模板以支持自动批准，这意味着具有必要 IAM 权限的用户可以选择启动更改请求，而无需额外批准。现在，您还可以限制个别用户、组或 IAM 角色提交自动批准请求，即使更改模板支持此类请求。通过使用新的 IAM 条件密钥，`ssm:AutoApprove` 可以实现这一点。有关更多信息，请参阅控制对自动批准运行手册工作流的访问	2022 年 6 月 15 日
更新了有关维护时段任务角色的指南	以前，Systems Manager 控制台允许您选择 AWS 托管式 IAM 服务相关角色 `AWSServiceRoleForAmazonSSM` ，以用作任务的维护角色。现在不再建议将此角色及其相关策略 `AmazonSSMServiceRolePolicy` 用于维护时段任务。而应为维护时段任务创建一个自定义策略和角色。有关更多信息，请参阅设置 Maintenance Windows。	2022 年 6 月 9 日

端口转发到 Session Manager 的远程主机支持	Session Manager 现在支持端口转发到远程主机的会话。远程主机不需要由 Systems Manager 进行管理。有关更多信息，请参阅启动会话（端口转发到远程主机）。	2022 年 5 月 25 日
更新内容：在 Amazon EC2 Linux 实例上手动安装 SSM Agent 的说明	为了响应客户反馈，我们对提供在 Amazon EC2 实例上手动安装 SSM Agent 说明的主题进行了全面修改。这些主题现在提供了使用全局可用文件的命令，您可以复制和粘贴这些文件，以便在任何 AWS 区域中的 EC2 实例上快速安装。这些主题还提供了帮助您创建安装命令的信息，这些命令使用您自己的工作区域中的可用文件。当您使用脚本或模板在多个实例上安装代理时，建议使用后一种方法。有关更多信息，请参阅在适用于 Linux 的 EC2 实例上手动安装 SSM Agent 章节中的 Linux 操作系统的说明。	2022 年 5 月 9 日

新主题：预安装了 SSM Agent 的 Amazon Machine Images（AMIs）	为了响应客户反馈，我们集中了关于哪些 AWS 托管式 AMIs 包括预安装 SSM Agent 的信息。本主题还提供了有关如何验证从这些 AMIs 创建的 Amazon EC2 实例是否已成功安装并运行的说明。对于代理可能无法成功安装或已安装但未启动的极少数情况，我们还提供有关在这些实例上启动或手动安装代理的信息。有关详细信息，请参阅预安装了 SSM Agent 的 Amazon Machine Images（AMIs）。	2022 年 5 月 8 日
新的 State Manager 章节	添加了描述 State Manager 何时运行关联的详细信息的新章节。有关更多信息，请参阅关于关联计划。	2022 年 4 月 27 日
Patch Manager 现在支持 Rocky Linux	现在可以使用 Patch Manager 来修补 Rocky Linux 节点。许多适用于 RHEL 8 的修补规则也适用于 Rocky Linux。Rocky Linux 8 使用了新的 `AWS-DefaultRockyLinuxPatchBaseline` 。有关更多信息，请参阅以下主题： • 如何选择安全性补丁 • 如何安装补丁 • 补丁基准规则在 RHEL、CentOS Stream 和 Rocky Linux 上的工作原理。	2022 年 4 月 14 日

Patch Manager 现在支持 CentOS Stream 8	您现在可以使用 Patch Manager 修补 CentOS Stream 8 实例和 Red Hat Enterprise Linux (RHEL) 4.4-4.5 实例。许多适用于 RHEL 8 的修补规则也适用于 CentOS Stream 8。CentOS Stream 8 使用了 `AWS-DefaultCentOSPatchBaseline` 。有关更多信息，请参阅以下主题： • 如何选择安全性补丁 • 如何安装补丁 • 补丁基准规则在 RHEL 和 CentOS Stream 上的工作原理	2022 年 4 月 4 日
创建 Change Manager 的担任角色	新章节明确了为 Change Manager 创建和实施担任角色的要求。担任角色是一个 AWS Identity and Access Management (IAM) 服务角色，该角色可以让 Change Manager 代表您安全地运行在批准的更改请求中指定的运行手册工作流。该角色为 Change Manager 授予 AWS Systems Manager (AWS STS) `AssumeRole` 信任关系。有关更多信息，请参阅为 Change Manager 配置角色和权限。	2022 年 3 月 18 日

批量批准或拒绝 Change Manager 更改请求	在 Systems Manager 控制台中，您现在可以在单个操作中选择多个要批准或拒绝的更改请求。有关更多信息，请参阅审核和批准或拒绝更改请求（控制台）。	2022 年 3 月 8 日
对 Rocky Linux 和 Windows Server 2022 托管式节点的支持	Systems Manager 支持 Rocky Linux 和 Windows Server 2022 托管式节点，包括位于本地或与其他云提供商共享的边缘设备和混合计算机。要将 Systems Manager 与这些操作系统配合使用，必须完成所有必要的 Systems Manager 设置过程，包括混合环境或边缘设备的过程（如果适用）。有关更多信息，请参阅设置 Systems Manager。对于 Rocky Linux 计算机，还必须手动安装 SSM Agent。有关更多信息，请参阅在在 Rocky Linux 实例上手动安装 SSM Agent。对于 Windows Server 2022 Amazon Elastic Compute Cloud (Amazon EC2) 实例，默认情况下会安装 SSM Agent。	2022 年 3 月 1 日

允许 Automation 适应您的并发需求并查看 Automation 使用情况指标	您现在可以允许 Automation 自动调整并发自动化配额，并查看发布到 CloudWatch 的 Automation 使用情况指标。有关自适应并发的更多信息，请参阅允许 Automation 适应您的并发需求。有关如何查看 Automation 使用情况指标的更多信息，请参阅使用 Amazon CloudWatch 监控 Automation 指标。	2022 年 1 月 27 日
允许 Automation 适应您的并发需求并查看 Automation 使用情况指标	您现在可以允许 Automation 自动调整并发自动化配额，并查看发布到 CloudWatch 的 Automation 使用情况指标。有关自适应并发的更多信息，请参阅允许 Automation 适应您的并发需求。有关如何查看 Automation 使用情况指标的更多信息，请参阅使用 Amazon CloudWatch 监控 Automation 指标。	2022 年 1 月 27 日
按类别组织的 Systems Manager 文档	Amazon 拥有的 Systems Manager 文档现在按类型和类别进行组织，以帮助您找到所需的文档。	2022 年 1 月 13 日

创建并调用 Automation 的集成	现在，您可以通过创建集成在自动化期间使用 Webhooks 发送消息。可以在自动化过程中使用新的运行手册中的新 `aws:invokeWebhook` 操作调用集成。有关创建集成的更多信息，请参阅为 Automation 创建 Webhook 集成。要了解有关 `aws:invokeWebhook` 操作的更多信息，请参阅 aws:invokeWebhook – 调用 Automation Webhook 集成。	2022 年 1 月 13 日
在新的 AWS 区域 中不可用的功能	以下 Systems Manager 工具目前在新的亚太地区（雅加达）区域不可用。 • Application Manager • Change Calendar • Change Manager • Explorer • Fleet Manager • Incident Manager • Quick Setup	2021 年 12 月 13 日

查看应用程序的资源成本详细信息	Application Manager 通过 Cost Explorer 小组件与 AWS 账单与成本管理 集成。在账单和成本管理控制台中启用 Cost Explorer 后，Application Manager 中的 Cost Explorer 小组件会显示特定非容器应用程序或应用程序组件的成本数据。您可以根据条形图或折线图中的不同时段、精细度和成本类型，使用小组件中的筛选条件查看成本数据。有关更多信息，请参阅查看有关应用程序的概览信息。	2021 年 12 月 7 日
使用 Fleet Manager 管理进程	现在，您可以使用 Fleet Manager 管理节点上的进程。有关更多信息，请参阅使用进程。	2021 年 12 月 6 日
术语变更：托管式实例现在为托管式节点	由于对 AWS IoT Greengrass 核心设备的支持，在大多数 Systems Manager 文档中，托管式实例一词已更改为托管式节点。Systems Manager 控制台、API 调用、错误消息和 SSM 文档仍使用实例一词。	2021 年 11 月 29 日

对边缘设备的支持	Systems Manager 支持以下边缘设备配置。 • AWS IoT Greengrass：Systems Manager 现在支持为 AWS IoT Greengrass 配置以及运行 AWS IoT Greengrass Core 软件的任何设备。要注册 AWS IoT Greengrass 核心设备，您必须创建一个 AWS Identity and Access Management (IAM) 服务角色。您还必须使用 AWS IoT Greengrass 控制台将 SSM Agent 部署为设备上的 AWS IoT Greengrass 组件。有关更多信息，请参阅为边缘设备设置 AWS Systems Manager。 • 混合环境中的边缘设备：将 AWS IoT 核心设备和非 AWS IoT 设备配置为本地计算机后，Systems Manager 还支持这两种设备。要注册设备，您必须创建 IAM 服务角色、为混合环境创建托管式节点激活并在设备上手动安装 SSM Agent。有关更多信息，请参阅为混合环境设置 AWS Systems Manager	2021 年 11 月 29 日

使用远程桌面连接到托管式实例	现在，您可以通过远程桌面协议 (RDP) 使用 Fleet Manager 连接到托管式 Windows 实例。这些由 Amazon DCV 提供支持的远程桌面会话可以直接从浏览器连接到实例。有关更多信息，请参阅使用远程桌面进行连接。	2021 年 11 月 23 日
指定最长会话持续时间并提供会话原因	现在，您可以在 AWS 账户中为 AWS 区域 内的所有 Session Manager 会话指定最长会话持续时间。会话到达指定的持续时间时会终止。现在，您还可以选择在开始会话时添加原因。有关更多信息，请参阅指定最长会话持续时间。	2021 年 11 月 16 日
Patch Manager 现在支持 Raspberry Pi OS 操作系统	现在，您可以使用 Patch Manager 修补 Raspberry Pi OS 实例。Patch Manager 支持修补 Raspberry Pi OS 9 (Stretch) 和 10 (Buster)。由于 Raspberry Pi OS 是基于 Debian 的操作系统，其适用的许多修补规则与 Debian Server 相同。有关更多信息，请参阅以下主题： • 如何选择安全性补丁 • 如何安装补丁 • 补丁基准规则在 Debian Server 和 Raspberry Pi OS 上的工作原理	2021 年 11 月 16 日

访问 Red Hat 知识库门户	使用 Fleet Manager 访问 RHEL 知识库门户，查找有关使用 Red Hat 产品的解决方案、文章、文档和视频。有关更多信息，请参阅访问 Red Hat 知识库门户。	2021 年 11 月 3 日
批量编辑 OpsItems	OpsCenter 现在支持批量编辑 OpsItems。您可以选择多个 OpsItems，然后编辑以下字段之一：Status（状态）、Priority（优先级）、Severity（严重性）、Category（类别）。有关更多信息，请参阅编辑 OpsItems。	2021 年 10 月 15 日
创建填充 AWS 资源的输入参数	您现在可以在自动化运行手册中创建输入参数，该手册会在 AWS Management Console 中填充 AWS 资源。有关更多信息，请参阅创建填充 AWS 资源的输入参数。	2021 年 10 月 14 日
维护时段的新任务调用截止选项	现在，在到达为维护时段指定的截止时间后，您可以选择阻止启动任何新任务调用。有关更多信息，请参阅为维护时段分配任务（控制台）。	2021 年 10 月 13 日

Patch Manager 对 macOS 11.3.1 和 11.4 (Big Sur) 的支持	适用于 macOS 11.3.1 和 11.4 (Big Sur) 的 Amazon Elastic Compute Cloud (Amazon EC2) 实例现在可以使用 Patch Manager 进行修补。这是对 macOS 10.14.x (Mojave) 和 10.15.x (Catalina) 的现有支持的补充。有关使用 Patch Manager 的信息，请参阅 AWS Systems ManagerPatch Manager。	2021 年 10 月 1 日
Application Manager 中的 Application Insights	Application Manager 与 Amazon CloudWatch Application Insights 进行集成。Application Insights 可在应用程序资源和技术堆栈中指定并设置关键指标、日志和告警。Application Insights 会持续监控指标和日志，以检测异常情况和错误，并将它们关联起来。在系统检测到错误和异常情况时，Application Insights 生成 CloudWatch Events，可以使用这些事件来设置通知或执行操作。您可以在 Application Manager 的概览和监控选项卡中启用和查看 Application Insights。有关 Application Insights 的更多信息，请参阅 Amazon CloudWatch 用户指南中的什么是 Amazon CloudWatch Application Insights。	2021 年 9 月 21 日

将其他日历中的事件导入 Change Calendar	现在，您可以将第三方日历中的事件导入 Change Calendar 中的日历。以往，必须将每个事件手动输入到日历中。将日历从受支持的第三方日历提供程序导出到 iCalendar (`.ics`) 文件后，将其导入到Change Calendar 中，日历中的事件便会包括在 Systems Manager 的打开或关闭的日历规则中。受支持的提供程序包括 iCloud 日历、Google 日历和 Microsoft Outlook。有关更多信息，请参阅导入和管理来自第三方日历的事件。	2021 年 9 月 8 日
Application Manager 中的新的标记和运行手册功能	标记增强功能包括能够向 Application Manager 应用程序中的特定资源或所有资源添加标记或从中删除标记。运行手册增强功能包括能够查看特定资源类型的经筛选的运行手册列表，或启动相同类型的所有资源上的运行手册。有关更多信息，请参阅在 Application Manager 中使用标签和在 Application Manager 中使用运行手册。	2021 年 8 月 31 日

新示例：使用 AWS CLI 创建更改请求	使用 AWS CLI 创建更改请求的示例已添加到 Change Manager 章节。该示例使用示例 `AWS-HelloWorldChangeTemplate` 更改模板和 `AWS-HelloWorld` runbook： • 创建更改请求 (AWS CLI)	2021 年 8 月 20 日
新部分：在 Amazon EKS 中使用参数	Parameter Store 章节中增加了一个新的部分。本主题是介绍如何在 Amazon EKS 集群中使用参数的演练。有关更多信息，请参阅在 Amazon Elastic Kubernetes Service 中使用 Parameter Store 参数。	2021 年 8 月 19 日
更新了 Patch Manager 生命周期钩子	现在，Patch Manager 在 Patch now (立即修补) 修补操作期间为其他点提供了生命周期钩子（能够运行 Systems Manager 命令文档）。如果您计划在运行立即修补之后重启实例，则您可以指定一个生命周期挂钩在重启完成之后运行。有关更多信息，请参阅使用“立即修补”生命周期钩子和关于 `AWS-RunPatchBaselineWithHooks` SSM 文档。	2021 年 8 月 9 日

Change Manager 请求现在支持自动批准	您现在可以在 Change Manager 中配置更改模板以支持自动批准，这意味着具有必要 IAM 权限的用户可以选择启动更改请求，而无需额外批准。有权访问自动批准模板的用户仍然可以选择指定审批人。为了帮助您控制 Change Manager 流程，在更改冻结期间，所有请求仍需要经过批准。有关更多信息，请参阅以下主题： • 创建更改模板 • 创建更改请求 • 试用 AWS 托管 `Hello World` 更改模板	2021 年 7 月 30 日
OpsCenter Operational Insights (运营洞察)	OpsCenter 会自动分析您账户中的 OpsItems 并生成洞察。洞察所包含的信息有助于您了解账户中有多少重复的 OpsItems，以及它们是由哪些源创建的。洞察还提供建议的最佳实践和自动化运行手册，以帮助您解决重复的 OpsItems。有关更多信息，请参阅使用运营洞察。	2021 年 7 月 13 日
在 Fleet Manager 中查看停止的实例	现在，您可以从 Fleet Manager 控制台查看哪些实例的状态为 `running` (正在运行)，哪些实例的状态为 `stopped` (已停止)。有关更多信息，请参阅 AWS Systems ManagerFleet Manager。	2021 年 7 月 12 日

新主题：编写自动化运行手册	新主题编写自动化运行手册提供了指导和叙述性示例，介绍如何编写自定义自动化运行手册的内容。	2021 年 7 月 8 日
Application Manager 中的 AWS CloudFormation 堆栈和模板创建	通过与 CloudFormation 进行集成，Application Manager 可帮助您预置和管理应用程序的资源。您可以在 Application Manager 中创建、编辑和删除 AWS CloudFormation 模板和堆栈。Application Manager 还包括一个模板库，您可以在其中克隆、创建和存储模板。Application Manager 和 CloudFormation 显示有关堆栈当前状态的相同信息。模板和模板更新将存储在 Systems Manager 中，直到您预置堆栈，届时更改也会显示在 CloudFormation 中。有关更多信息，请参阅在 Application Manager 中使用 AWS CloudFormation 堆栈。	2021 年 7 月 8 日
新主题：在混合实例上自动轮换 SSM Agent 的私有密钥	新主题设置私有密钥自动轮换提供了有关如何通过将 SSM Agent 配置为自动轮换混合环境私有密钥来加强安保状况的说明。	2021 年 6 月 15 日
适用于 AWS CLI 版本 1.2.205.0 的 Session Manager 插件	适用于 AWS CLI 的 Session Manager 插件新版本已发布。有关更多信息，请参阅 Session Manager 插件的最新版本和发行版历史记录。	2021 年 6 月 10 日

新的 IAM 服务相关角色	启用 OpsCenter 运营洞察时，Systems Manager 将创建名为 AWSSSMOps InsightsServiceRolePolicy 的新 AWS Identity and Access Management (IAM) 服务相关角色。有关此角色的更多信息，请参阅在 Systems Manager OpsCenter 中使用角色创建运营洞察 OpsItem：AWSSSMOps InsightsServiceRolePolicy。	2021 年 6 月 9 日
适用于 Linux 的新的 Patch Manager 故障排除内容	新主题在 Linux 上运行 AWS-RunPatchBaseline 时出现错误提供了修补 Linux 操作系统的托管式实例时可能遇到的一些问题的说明和解决方案。	2021 年 6 月 8 日

改进了对不需要指定目标的维护时段任务的支持（控制台）	现在，您可以在控制台中创建维护时段任务，而无需在任务中指定目标（如果不需要）。此选项之前仅在使用 AWS CLI 或 API 时才可用。此选项适用于自动化、AWS Lambda 和 AWS Step Functions 任务类型。例如，如果您创建自动化任务并在自动化文档参数中指定了要更新的资源，则不再需要在任务本身中指定一个目标。有关更多信息，请参阅注册不含目标的维护时段任务、为维护时段分配任务（控制台），以及使用维护时段计划自动化。	2021 年 5 月 28 日
重新定位自动化运行手册参考	自动化运行手册参考已移至新位置。有关更多信息，请参阅 Systems Manager 自动化运行手册参考。	2021 年 5 月 10 日
推出 AWS Systems Manager Incident Manager	Incident Manager 是一个事件管理控制台，旨在帮助用户缓解影响其 AWS 托管应用程序的事件并从中恢复。有关更多信息，请参阅《AWS Systems Manager Incident Manager 用户指南》。	2021 年 5 月 10 日

State Manager 支持 Change Calendar	现在，您可以在创建或更新 State Manager 关联时指定Change Calendar名称或 Amazon Resource Name (ARN)。State Manager 仅在更改日历打开时应用关联，而不会在其关闭时应用关联。有关更多信息，请参阅创建关联以及编辑和创建新的关联版本。	2021 年 5 月 6 日
克隆 Systems Manager 文档	使用 Systems Manager 文档控制台，您现在可以将现有文档中的内容复制到可以修改的新文档。要了解详情，请参阅克隆 SSM 文档。	2021 年 5 月 4 日
将 Security Hub 与 Explorer 和 OpsCenter 集成	现在可将 Explorer 和 OpsCenter 与 AWS Security Hub 集成。Security Hub 提供了 AWS 中安全状态的全面视图，可帮助您检查您的环境是否符合安全行业标准和最佳实践。与 Explorer 集成后，您可以在 Explorer 控制面板上的 Security Hub 小组件中查看安全性调查结果。与 OpsCenter 集成后，您可以为 Security Hub 调查结果创建 OpsItems。有关更多信息，请参阅在 Explorer 中接收来自 AWS Security Hub 的调查结果和在 OpsCenter 中接收来自 AWS Security Hub 的调查结果。	2021 年 4 月 27 日

新主题：文档惯例	我们增加了一个新主题，以帮助用户了解《AWS Systems Manager 用户指南》的常见印刷惯例。有关更多信息，请参阅文档惯例。	2021 年 4 月 21 日
更新的主题：关于在 Windows Server 上修补 Microsoft 发布的应用程序	主题关于在 Windows Server 上修补 Microsoft 发布的应用程序现在已经明确，为了让 Patch Manager 能够在 Windows Server 托管式实例上为 Microsoft 发布的应用程序安装补丁，必须在实例上启用 Windows 更新选项 Give me updates for other Microsoft products when I update Windows（更新 Windows 时向我提供其他 Microsoft 产品的更新）。	2021 年 4 月 12 日
重新组织自动化运行手册参考	为了帮助您找到所需的运行手册并更有效地浏览参考内容，我们按照相关的 AWS 服务重新组织了自动化运行手册参考中的内容。要查看这些更改，请参阅 Systems Manager 自动化运行手册参考。	2021 年 4 月 12 日

Patch Manager：生成 .csv 补丁合规性报告	Patch Manager 现在支持为实例生成补丁合规性报告，并将报告以 .csv 格式保存在您选择的 S3 存储桶中。然后，您可以使用 Amazon QuickSight 等工具分析补丁合规性报告数据。您可以为 AWS 账户 中的单个实例或所有实例生成补丁合规性报告。您可以按需生成一次性报告，也可以设置自动创建报告的计划。此外，您还可以指定 Amazon Simple Notification Service 主题，以便在生成报告时提供通知。有关更多信息，请参阅生成 CSV 补丁合规性报告。	2021 年 4 月 9 日
删除 Parameter Store 参数标签	现在，您可以使用 Systems Manager 控制台或 AWS CLI 删除 Parameter Store 参数标签。有关更多信息，请参阅使用参数标签。	2021 年 4 月 6 日
计划在使用 Patch now (立即修补) 后重启实例	Patch Manager 现在支持使用 Patch now (立即修补) 功能计划在安装补丁后重启实例的时间。这补充了现有选项，即仅在需要完成补丁安装或修补操作后跳过所有重启时才重启实例。有关更多信息，请参阅按需修补实例。	2021 年 4 月 1 日
新主题：发现公有参数	现在可以使用 AWS CLI 或 Systems Manager 控制台找到 Parameter Store 公有参数。有关更多信息，请参阅查找公有参数。	2021 年 4 月 1 日

Patch now (立即修补) 更新：将日志存储在 S3 中并运行生命周期钩子	运行 Patch Manager Patch now (立即修补) 操作时，您可以选择一个 S3 存储桶，在其中自动存储修补日志。此外，您可以选择在操作期间的以下三个时间点运行 Systems Manager 命令文档（SSM 文档）作为生命周期钩子：安装前、安装后和退出时。有关更多信息，请参阅按需修补实例。	2021 年 3 月 31 日
Systems Manager 现在报告 AWS 托管式策略的更改	从 2021 年 3 月 24 日开始，对托管式策略的更改将在主题 AWS 托管式策略的 Systems Manager 更新中报告。列出的第一项更改是添加了对 Explorer 工具的支持，用来报告来自多个账户和区域的 OpsData 及 OpsItems。	2021 年 3 月 24 日

Explorer 会自动允许所有 OpsData 源根据 AWS Organizations 中的账户进行资源数据同步	在创建资源数据同步时，如果您选择 AWS Organizations 中的任一选项，Systems Manager 会自动允许您的组织（或选定的组织部门）中的所有 AWS 账户 使用选定 AWS 区域 中的所有 OpsData 源。也就是说，例如，即使您没有允许在某个 AWS 区域中使用 Explorer，但如果您为资源数据同步选择了 AWS Organizations 选项，Systems Manager 也会自动从该区域收集 OpsData。有关更多信息，请参阅关于多个账户和区域资源数据同步。	2021 年 3 月 24 日
Systems Manager 自动化为运行手册提供了新的系统变量	使用新的 `global:AWS_PARTITION` 系统变量，您可以在编写运行手册时指定资源所在的 AWS 分区。有关更多信息，请参阅自动化系统变量。	2021 年 3 月 18 日
允许对 Change Manager 更改请求进行多级批准	在创建 Change Manager 更改模板时，现在您可以要求多个级别的审批人授予运行更改请求的权限。例如，您可能要求技术审核人员首先批准根据更改模板创建的变更请求，然后需要一个或多个经理的第二级批准。有关更多信息，请参阅创建更改模板。	2021 年 3 月 4 日

Patch Manager 现在支持 Oracle Linux 8.x	现在，您可以使用 Patch Manager 修补版本 8.3 之前的 Oracle Linux 8.x 实例。有关更多信息，请参阅以下主题： • 如何选择安全性补丁 • 如何安装补丁 • 补丁基准规则在 Oracle Linux 上的工作原理	2021 年 3 月 1 日
OpsCenter 显示所选资源的其他 OpsItems	为了帮助您调查问题并提供问题的上下文，您可以查看特定 AWS 资源的 OpsItems 列表。该列表会显示每个 OpsItem 的状态、严重性级别和标题。该列表还包括指向每个 OpsItem 的深层链接。有关更多信息，请参阅查看特定资源的其他 OpsItems。	2021 年 3 月 1 日
在运行时定义修补首选项	现在，您可以使用基准覆盖功能在运行时定义修补首选项。有关更多信息，请参阅使用 BaselineOverride 参数。	2021 年 2 月 25 日
新的 Systems Manager 文档类型	AWS CloudFormation 模板现在可以存储为 Systems Manager 文档。通过将 CloudFormation 模板存储为 Systems Manager 文档，您可以受益于版本控制、比较版本内容以及与账户共享等 Systems Manager 文档功能。有关更多信息，请参阅 AWS Systems Manager 文档。	2021 年 2 月 9 日

使用可选钩子修补实例	新的 SSM 文档 AWS-RunPatchBaselineWithHooks 提供了可用于在实例修补周期的三个时间点运行 SSM 文档的钩子。有关 AWS-RunPatchBaselineWithHooks 的信息，请参阅关于 AWS-RunPatchBaselineWithHooks SSM 文档。有关使用所有三个钩子的修补操作的示例演练，请参阅演练：更新应用程序依赖项、修补实例以及执行特定于应用程序的运行状况检查。	2021 年 2 月 2 日
新主题：使用硬件指纹验证本地服务器和虚拟机	SSM Agent 使用计算出的指纹来验证您在该服务中注册的本地服务器和 VM 的身份。此类指纹是存储在文件库中的不透明字符串，代理会将其传递给某些 Systems Manager API。有关硬件指纹的信息以及配置相似性阈值以协助计算机验证的说明，请参阅使用硬件指纹验证本地服务器和虚拟机。	2021 年 1 月 25 日
新主题：SSM Agent 技术参考	主题 SSM Agent 技术参考汇集了可帮助您实施 AWS Systems Manager SSM Agent 并了解其工作原理的信息。本主题包括一个全新的部分，即 SSM Agent 按 AWS 区域 滚动更新。	2021 年 1 月 21 日

Windows Server 2008 上的 SSM Agent	从 2020 年 1 月 14 日开始，Microsoft 不再为 Windows Server 2008 的功能或安全性更新提供支持。Windows Server 2008 AMIs 确实包含 SSM Agent，但不再针对此操作系统更新代理。	2021 年 1 月 5 日
改进了对不需要指定目标的维护时段任务的支持（仅限 AWS CLI 和 API）	现在，您可以创建维护时段任务，而无需在任务中指定目标（如果不需要）（仅限 AWS CLI 和 API）。这适用于自动化、AWS Lambda 和 AWS Step Functions 任务类型。例如，如果您创建自动化任务并在自动化运行手册参数中指定了要更新的资源，则不再需要在任务本身中指定一个目标。有关更多信息，请参阅注册不含目标的维护时段任务和使用维护时段计划行自动化。	2020 年 12 月 23 日
新的自动化功能	Systems Manager 自动化运行手册中添加了一个新的共享属性。使用 `onCancel` 属性，您可以指定在用户取消自动化时将自动化转到哪个步骤。有关更多信息，请参阅所有操作共享的属性。	2020 年 12 月 21 日
新主题：使用 IAM 处理关联	Systems Manager State Manager 章节中添加了一个新主题，其中介绍了使用 IAM 创建关联的最佳实践。有关更多信息，请参阅使用 IAM 处理关联。	2020 年 12 月 18 日

State Manager 现在支持多个区域和多个账户	现在可以创建或更新多个区域或账户的关联。有关更多信息，请参阅创建关联。	2020 年 12 月 15 日
新工具：Fleet Manager	Fleet Manager（AWS Systems Manager 中的一项工具）是一种统一的用户界面（UI）体验，可帮助您远程管理在 AWS 上或在本地运行的服务器实例集。借助 Fleet Manager，您可以从一个控制台查看整个服务器机群的运行状况和性能状态。您还可以从单个实例收集数据，以便从控制台执行常见的故障排除和管理任务。有关更多信息，请参阅 AWS Systems ManagerFleet Manager。	2020 年 12 月 15 日
新工具：Change Manager	Amazon Web Services 发布了 Change Manager，这是一个企业更改管理框架，用于请求、批准、实施和报告对应用程序配置和基础设施的操作更改。如果您使用 AWS Organizations，可从单个委托管理员账户中，管理多个 AWS 区域 中多个 AWS 账户 的变更。或者，通过使用本地账户，您可以管理单个 AWS 账户 的变更。使用 Change Manager 可管理对 AWS 资源和本地部署资源的变更。有关更多信息，请参阅 AWS Systems ManagerChange Manager。	2020 年 12 月 15 日

新工具：Application Manager	Application Manager 可帮助您结合应用程序的上下文调查和修复 AWS 资源的问题。Application Manager 会将来自多个 AWS 服务和 Systems Manager 工具的操作信息聚合到单个 AWS Management Console。有关更多信息，请参阅 AWS Systems ManagerApplication Manager。	2020 年 12 月 15 日
AWS Systems Manager 支持适用于 macOS 的 Amazon EC2 实例	发布了 Amazon Elastic Compute Cloud (Amazon EC2) 对 macOS 实例的支持后，Systems Manager 现在也支持适用于 macOS 的 EC2 实例的众多操作。支持的版本包括 macOS 10.14.x (Mojave) 和 10.15.x (Catalina)。有关更多信息，请参阅以下主题。 • 有关在 EC2 实例上为 macOS 安装 SSM Agent 的信息，请参阅在 EC2 实例上为 macOS 安装和配置 SSM Agent。 • 有关为 macOS 修补 EC2 实例的信息，请参阅如何安装补丁和创建自定义补丁基准 (macOS) 。 • 有关对适用于 macOS 的 EC2 实例的支持的一般信息，请参阅《Amazon EC2 用户指南》中的 Amazon EC2 Mac 实例。	2020 年 11 月 30 日

维护时段虚拟参数：{{TARGET_ID}} 和 {{RESOURCE_ID}} 支持的新资源类型	现在，另一种资源类型可与虚拟参数 {{TARGET_ID}} 和 {{RESOURCE_ID}} 一起使用。您现在可以将资源类型 AWS::RDS::DBCluster 与这两个虚拟参数一起使用。有关维护时段伪参数的信息，请参阅注册维护时段任务时使用伪参数。	2020 年 11 月 27 日
适用于 AWS CLI 版本 1.2.30.0 的 Session Manager 插件	适用于 AWS CLI 的 Session Manager 插件新版本已发布。有关更多信息，请参阅 Session Manager 插件的最新版本和发行版历史记录。	2020 年 11 月 24 日
新主题：比较 SSM 文档版本	现在，您可以在 Systems Manager 文档控制台中比较不同 SSM 文档版本之间的内容差异。有关更多信息，请参阅比较 SSM 文档版本。	2020 年 11 月 24 日
Systems Manager 现在支持 VPC 端点策略	现在，您可以为 Systems Manager 的接口 VPC 端点创建策略。有关更多信息，请参阅创建接口 VPC 端点策略。	2020 年 11 月 18 日
新主题：指定空闲会话超时值	现在，您可以指定在 Session Manager 结束会话之前允许用户处于非活动状态的时间长度。有关更多信息，请参阅指定空闲会话超时值。	2020 年 11 月 18 日

新的 Session Manager 日志记录功能	现在，您可以向 Amazon CloudWatch Logs 持续发送 JSON 格式的会话数据日志流。有关更多信息，请参阅使用 Amazon CloudWatch Logs 流式传输会话数据。	2020 年 11 月 18 日
新主题：验证 SSM Agent 的签名	现在，您可以验证 Linux 实例上 SSM Agent 的安装程序包的加密签名。有关更多信息，请参阅 SSM 文档架构和功能。	2020 年 11 月 17 日
新主题：了解自动化状态	Systems Manager 自动化章节中添加了一个新主题，其中介绍了操作和自动化的状态。有关更多信息，请参阅了解自动化状态。	2020 年 11 月 17 日
`aws:downloadContent` 插件的新源类型	现在支持将 Git 和 HTTP 作为 `aws:downloadContent` 插件的源类型。有关更多信息，请参阅 `aws:downloadContent`。	2020 年 11 月 17 日
新的 Systems Manager 文档（SSM 文档）架构功能	在使用架构版本 2.2 或更高版本的 SSM 文档中，`precondition` 参数现在支持引用文档的输入参数。有关更多信息，请参阅 SSM 文档架构和功能。	2020 年 11 月 17 日

Explorer 中的新数据源：AWS Config	Explorer 现在显示有关 AWS Config 合规性的信息，包括合规和不合规的 AWS Config 规则的总体摘要、合规和不合规资源的数量，以及每个资源的详细信息（当您深入了解某个不合规的规则或资源时）。有关更多信息，请参阅编辑 Systems Manager Explorer 数据源。	2020 年 11 月 11 日
新主题：使用关联来运行 Auto Scaling 组	State Manager 中增加了一个新的部分，其中介绍了创建关联来运行 Auto Scaling 组的最佳实践。有关更多信息，请参阅使用关联来运行 Auto Scaling 组。	2020 年 11 月 10 日
Quick Setup (快速设置) 现在支持将资源组设为目标	Quick Setup (快速设置) 现在支持选择资源组作为本地设置类型的目标。有关更多信息，请参阅选择 Quick Setup 目标。	2020 年 11 月 5 日

Patch Manager 增加了对 Debian Server 10 LTS、Oracle Linux 7.9 LTS 和 Ubuntu Server 20.10 STR 的支持	现在可以使用 Patch Manager 修补 Debian Server 10 LTS、Oracle Linux 7.9 LTS 和 Ubuntu Server 20.10 STR 实例。有关更多信息，请参阅以下主题： • Patch Manager 先决条件 • 如何选择安全性补丁 • 如何安装补丁 • 补丁基准规则在 Debian Server 上的工作原理 • 补丁基准规则在 Oracle Linux 上的工作原理 • 补丁基准规则在 Ubuntu Server 上的工作原理	2020 年 11 月 4 日
EventBridge 对 AWS Systems ManagerChange Calendar的新支持	现在，Amazon EventBridge 在事件规则中提供对 Change Calendar 事件的支持。当日历状态更改时，EventBridge 可以启动您定义的 EventBridge 规则中的目标操作。有关使用 EventBridge 和 Systems Manager 事件的信息，请参阅以下主题。 • 为 Systems Manager 事件配置 EventBridge • 参考：适用于 Systems Manager 的 Amazon EventBridge 事件模式和类型	2020 年 11 月 4 日

配置 CloudWatch 以通过告警创建 OpsItems	您可以将 Amazon CloudWatch 配置为当告警进入 ALARM (告警) 状态时，自动在 Systems Manager OpsCenter 中创建 OpsItem。这样做可以让您在单个控制台中快速诊断和修正 AWS 资源的问题。有关更多信息，请参阅配置 CloudWatch 以通过告警创建 OpsItems。	2020 年 11 月 4 日
对 Ubuntu Server 20.10 的支持	AWS Systems Manager 现在支持 Ubuntu Server 20.10 短期发行版 (STR)。有关更多信息，请参阅以下主题： • 支持的操作系统 • 为混合环境 (Linux) 安装 SSM Agent • 在 Ubuntu Server 实例上手动安装 SSM Agent • 检查 SSM Agent 状态并启动代理	2020 年 10 月 22 日
新主题：允许可配置的 Shell 配置文件	您现在可以通过 Session Manager 允许可配置的 Shell 配置文件。通过允许可配置的 Shell 配置文件，您可以自定义会话中的首选项，例如 Shell 首选项、环境变量、工作目录以及在启动会话时运行多个命令。有关更多信息，请参阅允许可配置的 Shell 配置文件。	2020 年 10 月 21 日

补丁合规性结果现在会报告哪些补丁解决了哪些 CVE	对于大多数受支持的 Linux 系统，在查看托管式实例的补丁合规性结果时，现在可以查看的详细信息将报告哪些可用补丁解决了哪些常见漏洞和风险 (CVE) 公告问题。此信息可帮助您确定需要安装丢失或失败的补丁的紧急程度。有关更多信息，请参阅查看补丁合规性结果。	2020 年 10 月 20 日
扩展了对 Linux 补丁元数据的支持	现在，您可以在 Patch Manager 中查看有关可用的 Linux 补丁的许多详细信息。您可以选择查看补丁数据，例如架构、epoch、版本、CVE ID、建议 ID、Bugzilla ID、存储库等。此外，DescribeAvailablePatches API 操作已更新，以支持 Linux 操作系统，并根据这些新提供的补丁元数据类型进行筛选。有关更多信息，请参阅以下主题： • 查看可用补丁 • 《AWS Systems Manager API Reference》中的 DescribeAvailablePatches 和 Patch • 《AWS CLI Command Reference》的 AWS Systems Manager 部分中的 describe-available-patches	2020 年 10 月 16 日

适用于 AWS CLI 版本 1.2.7.0 的 Session Manager 插件	适用于 AWS CLI 的 Session Manager 插件新版本已发布。有关更多信息，请参阅 Session Manager 插件的最新版本和发行版历史记录。	2020 年 10 月 15 日
新主题：会话文档架构	新主题会话文档架构介绍了会话文档的架构元素。这些信息可帮助您创建自定义会话文档，您可以在其中指定与 Session Manager 一起使用的会话类型的首选项。	2020 年 10 月 15 日
新主题：SSM 文档的自由文本搜索	Systems Manager Documents (文档) 页面上的搜索框现在支持自由文本搜索。自由文本搜索会将您输入的一个或多个搜索词与每个 SSM 文档中的文档名称进行比较。有关更多信息，请参阅使用自由文本搜索。	2020 年 10 月 15 日
新主题：Amazon EC2 托管式实例可用性故障排除	新主题 Amazon EC2 托管式实例可用性故障排除可帮助您调查为什么您已确认正在运行的 Amazon EC2 实例未列在 Systems Manager 的可用托管式实例列表中。	2020 年 10 月 6 日

Parameter Store 章节内容重新组织	为了帮助您更有效地查找所需信息，我们重新组织了《AWS Systems Manager 用户指南》中 Parameter Store 章节的内容。现在，大多数内容都组织在设置 Parameter Store 和使用 Parameter Store 部分中。此外，主题 AWS Systems ManagerParameter Store 已扩展，包括以下部分： • 我的组织如何从 Parameter Store 获益？ • 谁应该使用 Parameter Store？ • Parameter Store 具有哪些功能？ • 什么是参数？	2020 年 10 月 1 日
与补丁合规性相关的新主题	添加了以下主题，以帮助您识别不符合补丁合规性的托管式实例，了解不同类型的补丁合规性扫描，并采取适当的措施以使您的实例满足合规性要求。 • 识别不合规实例 • 修补不合规实例 • 查看补丁合规性结果	2020 年 9 月 24 日
SSM Agent 版本 3.0	Systems Manager 推出了新版本的 SSM Agent。	2020 年 9 月 21 日

新增和更新的主题：Amazon EventBridge 取代 CloudWatch Events 用于事件管理	CloudWatch Events 和 EventBridge 是相同的底层服务和 API，但 EventBridge 提供了更多功能，现在是管理 AWS 中事件的首选方式。（您在 CloudWatch 或 EventBridge 中所做的更改会反映在每个控制台中。）《AWS Systems Manager 用户指南》中对 CloudWatch Events 和现有过程的引用已更新，以反映 EventBridge 支持。此外，还添加了以下新主题。 • 监控 Systems Manager 事件 • 为 Systems Manager 事件配置 EventBridge • Systems Manager 目标类型示例 • 参考：适用于 Systems Manager 的 Amazon EventBridge 事件模式和类型	2020 年 9 月 18 日
集成 AWS Security Hub 和 Patch Manager	现在可将 Patch Manager 与 AWS Security Hub 集成。Security Hub 提供了 AWS 中安全状态的全面视图，可帮助您检查您的环境是否符合安全行业标准和最佳实践。与 Patch Manager 集成后，Security Hub 可从安全角度监控队列的修补状态。有关更多信息，请参阅将 Patch Manager 与 AWS Security Hub 集成。	2020 年 9 月 17 日

维护时段虚拟参数：{{TARGET_ID}} 和 {{RESOURCE_ID}} 支持的新资源类型	在您注册维护时段任务时，可以使用 `--task-invocation-parameters` 选项指定分别对于四种任务类型唯一的参数。您还可以使用虚拟参数句法引用特定值，例如 `{{TARGET_ID}}` 和 `{{RESOURCE_ID}}` 。维护时段任务运行时，它传递正确的值而不是伪参数占位符。现在，另两种资源类型可与虚拟参数 `{{TARGET_ID}}` 和 `{{RESOURCE_ID}}` 一起使用。您现在可以将资源类型 `AWS::RDS::DBInstance` 和 `AWS::SSM::ManagedInstance` 与这两个虚拟参数一起使用。有关维护时段伪参数的信息，请参阅注册维护时段任务时使用伪参数。	2020 年 9 月 14 日
使用新的 Patch now (立即修补) 选项按需修补实例	现在，您可以随时使用 Systems Manager 控制台修补实例，或扫描缺少的补丁。您无需创建或修改计划，也无需指定完整的修补配置选项即可做到这一点，以满足即时修补需求。您只需指定是扫描还是安装补丁，并确定操作的目标实例。Patch Manager自动为您的实例类型应用当前默认补丁基准，并应用最佳实践选项，以确定一次修补多少个实例，以及操作失败之前允许多少错误。有关更多信息，请参阅按需修补实例。	2020 年 9 月 9 日

新主题：检查 SSM Agent 状态并启动代理	新主题检查 SSM Agent 状态并启动代理提供了命令，以检查 SSM Agent 是否在每个支持的操作系统上运行。此外，其中还提供了在 SSM Agent 未运行时启动该代理的命令。	2020 年 9 月 7 日
Patch Manager 现在支持 Ubuntu Server 20.04 LTS	现在，您可以使用 Patch Manager 修补 Ubuntu Server 20.04 LTS 实例。有关更多信息，请参阅以下主题： • 如何选择安全性补丁 • 如何安装补丁 • 补丁基准规则在 Ubuntu Server 上的工作原理	2020 年 8 月 31 日
应用场景和最佳实践的新主题	我们添加了一个新主题，旨在帮助用户快速了解 Maintenance Windows 和 State Manager 之间的区别。有关更多信息，请参阅在 State Manager 和 Maintenance Windows之间进行选择。	2020 年 8 月 28 日
新 OpsCenter 功能	OpsCenter 包含新的功能，可以帮助您快速找到和运行自动化运行手册来修正问题。有关更多信息，请参阅 OpsCenter 中的自动化运行手册功能。	2020 年 8 月 19 日
Explorer 中的新数据源：AWS 支持 用例	Explorer 现在显示有关 支持 用例的信息。您必须在 支持 中设置一个企业账户。有关更多信息，请参阅编辑 Systems Manager Explorer 数据源。	2020 年 8 月 13 日

Distributor 现在提供来自 Trend Micro 的第三方软件包。	Distributor 现在包括来自 Trend Micro 的第三方软件包。您可以使用 Distributor 在托管式实例上安装 Trend Micro Cloud One 代理。Trend Micro Cloud One 可帮助您保护云中的工作负载。有关更多信息，请参阅 AWSDistributor。	2020 年 8 月 12 日
aws:configurePackage 文档插件现在包含 additionalArguments 参数。	Systems Manager 命令文档插件 aws:configurePackage 现在支持通过新的 additionalArguments 参数为您的脚本（安装、卸载和更新）提供其他参数。有关更多信息，请参阅主题 aws:configurePackage 。	2020 年 8 月 11 日
AppConfig 内容移至单独的用户指南	有关 AWS AppConfig 的信息已移至单独的用户指南中。有关更多信息，请参阅什么是 AWSAppConfig？ AppConfig 还有一个单独的文档登录页面，其中包含指向用户指南、AppConfig API 参考以及新的 AppConfig 研讨会的链接。	2020 年 8 月 3 日
Quick Setup 现在支持 AWS Organizations	Quick Setup现在支持 AWS Organizations，使您可以跨多个账户和区域快速配置所需的安全角色和常用的 Systems Manager 工具。有关更多信息，请参阅 AWS Systems ManagerQuick Setup。	2020 年 7 月 23 日

Explorer 中的新数据源：关联合规性	Explorer 现在显示来自 State Manager 的关联合规性数据。有关更多信息，请参阅编辑 Systems Manager Explorer 数据源。	2020 年 7 月 23 日
用于开启和关闭Kernel Live Patching的新 Systems Manager 命令文档	现在，如果您希望在 Amazon Linux 2 实例上开启或关闭Kernel Live Patching，可以将文档 `AWS-ConfigureKernelLivePatching` 与 Run Command 一起使用。此文档取代了为这些任务创建自己的自定义命令文档的需求。有关更多信息，请参阅在 Amazon Linux 2 实例上使用内核实时修补	2020 年 7 月 22 日
更新了自动化配额	自动化的服务配额已更新，包括用于速率控制自动化的单独队列。有关更多信息，请参阅 AWS Systems Manager 自动化。	2020 年 7 月 20 日

使用控制台指定维护时段的计划偏移天数	使用 Systems Manager 控制台，您现在可以指定在运行维护时段之前但在 CRON 表达式指定的日期和时间之后等待的天数。（此选项之前仅在使用 AWS SDK 或命令行工具时才可用。）例如，如果您的 CRON 表达式将维护时段计划在每月第三个星期二的晚上 23:30 运行 (`cron(0 30 23 ? * TUE#3 *)`)，并且您指定计划偏移为 2，则该时段要等到两天后的晚上 23:30 才会运行。有关更多信息，请参阅适用于 Systems Manager 的 cron 和 rate 表达式以及指定维护时段的计划偏移天数。	2020 年 7 月 17 日
使用 Run Command 更新 PowerShell	为了帮助您在 Windows Server 2012 和 2012 R2 实例上将 PowerShell 更新到 5.1 版本，我们在 AWS Systems Manager 用户指南中添加了一个演练。有关更多信息，请参阅使用 Run Command 更新 PowerShell。	2020 年 6 月 30 日

Patch Manager 现在支持 CentOS 8.0 和 8.1	您现在可以使用 Patch Manager 修补 CentOS 8.0 和 8.1 实例。有关更多信息，请参阅以下主题： • 如何选择安全性补丁 • 如何安装补丁 • 补丁基准规则在 CentOS 上的工作原理 • 在 CentOS 实例上手动安装 SSM Agent • 如何在混合 Linux 节点上安装 SSM Agent	2020 年 6 月 27 日

AppConfig 与 AWS CodePipeline 集成	AppConfig 是 AWS CodePipeline (CodePipeline) 的集成部署操作。CodePipeline 是一种完全托管式持续交付服务，可帮助您自动化发布管道，以实现快速可靠的应用程序和基础设施更新。根据您定义的发布模型，只要代码发生变化，CodePipeline 便会自动执行发布流程中的构建、测试和部署阶段。AppConfig 与 CodePipeline 集成可提供以下好处。有关更多信息，请参阅 AppConfig 与 CodePipeline 集成。 • 使用 CodePipeline 来管理编排的客户现在可以采用一种轻量级的方法，将配置更改部署到其应用程序，而不必部署整个代码库。 • 对于希望使用 AppConfig 来管理配置部署但由于 AppConfig 不支持其当前代码或配置存储而受到限制的客户，现在有了更多的选择。CodePipeline 支持 AWS CodeCommit、GitHub 和 BitBucket（仅举几例）。	2020 年 6 月 25 日

新章节：产品和服务集成	为了帮助您了解 Systems Manager 如何与 AWS 服务以及其他产品和服务集成，我们在《AWS Systems Manager 用户指南》中增加了一个新章节。有关更多信息，请参阅产品和服务与 Systems Manager 集成。	2020 年 6 月 23 日
自动化章节内容重新组织	为了帮助您找到所需内容，我们重新组织了《AWS Systems Manager 用户指南》中自动化章节中的主题。例如，自动化操作和自动化运行手册参考现在是本章中的顶级部分。有关更多信息，请参阅 AWS Systems Manager 自动化。	2020 年 6 月 23 日
指定维护时段的计划偏移天数	使用命令行工具或 AWS SDK，您现在可以指定在运行维护时段之前但在 CRON 表达式指定的日期和时间之后等待的天数。例如，如果您的 CRON 表达式将维护时段计划在每月第三个星期二的晚上 23:30 运行 (`cron(0 30 23 ? * TUE#3 *)`)，并且您指定计划偏移为 2，则该时段要等到两天后的晚上 23:30 才会运行。有关更多信息，请参阅适用于 Systems Manager 的 cron 和 rate 表达式以及指定维护时段的计划偏移天数。	2020 年 6 月 19 日

Patch Manager 支持 Amazon Linux 2 实例上的内核实时修补	适用于 Amazon Linux 2 的内核实时修补使您能够将安全漏洞和严重错误补丁应用于正在运行的 Linux 内核，而无需重启或中断正在运行的应用程序。现在，您可以使用 Patch Manager 允许该功能并应用内核实时补丁。有关更多信息，请参阅在 Amazon Linux 2 实例上使用内核实时修补。	2020 年 6 月 16 日
Patch Manager 增加了 Oracle Linux 版本支持	以前，Patch Manager 仅支持版本 7.6 的 Oracle Linux。如 Patch Manager 先决条件中所列，现在支持涵盖了版本 7.5-7.8。	2020 年 6 月 16 日
在修补操作中使用 InstallOverrideList 参数的示例方案	新主题 InstallOverrideList 参数的使用示例方案介绍使用 AWS-RunPatchBaseline 文档中的 InstallOverrideList 参数在不同维护时段计划中将不同类型的补丁应用到目标组的策略，同时仍然使用单个补丁基准。	2020 年 6 月 11 日
AppConfig 的预定义部署策略	AppConfig 现在提供预定义的部署策略。有关更多信息，请参阅创建部署策略。	2020 年 6 月 10 日

Patch Manager 现在支持 Red Hat Enterprise Linux (RHEL) 7.8-8.2	现在可以使用 Patch Manager 修补 RHEL 7.8–8.2 实例。有关更多信息，请参阅以下主题： • 如何选择安全性补丁 • 如何安装补丁 • 补丁基准规则在 RHEL 上的工作原理 • 在 Red Hat Enterprise Linux 实例上手动安装 SSM Agent • 如何在混合 Linux 节点上安装 SSM Agent	2020 年 6 月 9 日
Explorer 支持委派管理	如果您通过 AWS Organizations 使用资源数据同步来聚合多个 AWS 区域和 AWS 账户中的 Explorer 数据，我们建议您为 Explorer 配置委派管理员。委派管理员可以通过将 Explorer 管理员的数量限制为只有一个来提高 Explorer 安全性，只有该管理员可以创建或删除多账户和区域资源数据同步。您也不再需要登录到 AWS Organizations 管理账户即可管理 Explorer 中的资源数据同步。有关更多信息，请参阅配置委派管理员。	2020 年 6 月 3 日

仅在下一个指定的 Cron 周期应用 State Manager 关联	如果您不希望 State Manager 关联在创建后立即运行，可以在 Systems Manager 控制台中选择 Apply association only at the next specified Cron interval（仅在下一个指定的 Cron 周期应用关联）选项。有关更多信息，请参阅创建关联。	2020 年 6 月 3 日
Explorer 中的新数据源：AWS Compute Optimizer	Explorer 现在显示来自 AWS Compute Optimizer 的数据。这包括 Under provisioned（预置不足）和 Over provisioned（预置过度）EC2 实例的计数、优化发现结果、按需定价详情，以及关于实例类型和价格的建议。有关更多信息，请参阅设置相关服务中的设置 AWS Compute Optimizer 的详细信息。	2020 年 5 月 26 日
使用 Patch Manager 安装 Windows Service Pack 和 Linux 次要版本升级	新主题教程：创建用于安装 Windows Service Pack 的补丁基准（控制台）演示如何创建专门用于安装 Windows Service Pack 的补丁基准。主题创建自定义补丁基准 (Linux) 已更新，其中包含有关在补丁基准中包括 Linux 操作系统的次要版本升级的信息。	2020 年 5 月 21 日

Parameter Store 章节内容重新组织	所有有关为 Parameter Store 操作配置或设置选项的主题都已合并到设置 Parameter Store 部分中。其中包括管理参数层和增加 Parameter Store 吞吐量主题，这些内容已从本章的其他部分转走。	2020 年 5 月 18 日
创建用于与 Systems Manager API 操作交互的日期和时间字符串的新主题。	新主题为 Systems Manager 创建格式化的日期和时间字符串介绍如何创建格式化的日期和时间字符串，以便与 Systems Manager API 操作进行交互。	2020 年 5 月 13 日
关于加密 SecureString 参数的权限	新主题使用 IAM policy 限制访问 Systems Manager 参数解释了使用 AWS KMS key 和 AWS 提供的 AWS 托管式密钥 加密 `SecureString` 参数之间的区别。	2020 年 5 月 13 日
Patch Manager 现在支持 Debian Server 和 Oracle Linux 7.6 操作系统	现在，您可以使用 Patch Manager 修补 Debian Server 和 Oracle Linux 实例。Patch Manager 支持修补 Debian Server 8.x 和 9.x 及 Oracle Linux 7.6 版本。有关更多信息，请参阅以下主题： • 如何选择安全性补丁 • 如何安装补丁 • 补丁基准规则在 Debian Server 上的工作原理 • 补丁基准规则在 Oracle Linux 上的工作原理	2020 年 5 月 7 日

创建将 AWS Resource Groups 设为目标的 State Manager 关联	除了将 AWS 账户中的标签、单个实例和所有实例设为目标之外，您现在还可以创建将 AWS Resource Groups 中的实例设为目标的 State Manager 关联。有关更多信息，请参阅关于 State Manager 关联中的目标和速率控制	2020 年 5 月 7 日
用于验证 AMI ID 的 Parameter Store 中的新 `aws:ec2:image` 数据类型	创建 `String` 参数时，您可以将数据类型指定为 `aws:ec2:image` ，以确保所输入的参数值为有效的 Amazon Machine Image (AMI) ID 格式。通过支持 AMI ID 格式，您不必每次要在流程中使用的 AMI 发生更改时都使用新 ID 来更新所有脚本和模板。您可以创建数据类型为 `aws:ec2:image` 的参数，并输入 AMI ID 作为其值。这是您要从中创建新实例的 AMI。然后在模板、命令中引用此参数。当您准备使用其他 AMI 时，请更新参数值。Parameter Store 会验证新 AMI ID，您无需更新脚本和模板。有关更多信息，请参阅对 Amazon Machine Image ID 的本机参数支持。	2020 年 5 月 5 日

通过 Run Command 命令管理退出代码	利用 Run Command，您可以定义如何在脚本中处理退出代码。默认情况下，脚本中运行的最后一个命令的退出代码将报告为整个脚本的退出代码。但是，如果最后一个命令之前的任何命令使用以下方法失败，则可以包含 shell 条件语句来退出脚本。有关示例，请参阅新主题通过 Run Command 命令管理退出代码。	2020 年 5 月 5 日
发布了可用区和本地区域新的公有参数	已发布一些公有参数，用于以编程方式提供有关 AWS 可用区和 Local Zones（本地区域）的信息。这些是 AWS 服务和 AWS 区域现有全球基础设施公有参数的补充。有关更多信息，请参阅调用 AWS 服务、区域、端点、可用区、Local Zone 和 Wavelength Zone 的公有参数。	2020 年 5 月 4 日
Explorer 中的新数据源：AWS Trusted Advisor	Explorer 现在显示来自 AWS Trusted Advisor 的数据。这包括以下方面的最佳实践检查和建议的状态：成本优化、安全性、容错能力、性能和服务配额。有关更多信息，请参阅设置相关服务中的设置 Trusted Advisor 的详细信息。	2020 年 5 月 4 日

创建运行 Chef 配方的 State Manager 关联	您可以使用 `AWS-ApplyChefRecipes` 文档创建运行 Chef 说明书和配方的 State Manager 关联。本文档为运行 Chef 配方提供了以下好处： • 支持多个版本的 Chef（Chef 11 到 Chef 14）。 • 在目标实例上自动安装 Chef 客户端软件。 • （可选）在目标实例上运行 Systems Manager 合规性检查，并将合规性检查的结果存储在 S3 存储桶中。 • 在文档的单次运行中运行多个说明书和配方。 • （可选）在 `why-run` 模式下运行配方，以显示哪些配方会在未进行更改的情况下在目标实例上发生更改。 • （可选）将自定义 JSON 属性应用于 `chef-client` 运行。 有关更多信息，请参阅创建自动运行 Chef 配方的关联	2020 年 3 月 19 日
将多个 AWS 账户中的清单数据同步到中央 Amazon S3 存储桶	您可以将多个 AWS 账户中的 Systems Manager 清单数据同步到中央 S3 存储桶。必须在 AWS Organizations 中定义账户。有关更多信息，请参阅为 AWS Organizations 中定义的多个账户创建 Inventory 资源数据同步。	2020 年 3 月 16 日

在 Amazon S3 中存储 AppConfig 配置	以前，AppConfig 仅支持存储在 Systems Manager (SSM) 文档或 Parameter Store 参数中的应用程序配置。除了这些选项之外，AppConfig 现在还支持在 Amazon S3 中存储配置。有关更多信息，请参阅关于存储在 Amazon S3 中的配置。	2020 年 3 月 13 日
SSM Agent 默认安装在优化的 Amazon ECS 的 AMIs 上	现在，SSM Agent 默认安装在优化的 Amazon ECS 的 AMIs 上。有关更多信息，请参阅使用 SSM Agent。	2020 年 2 月 25 日
在控制台中创建 AppConfig 配置	现在，您可以使用 AppConfig 在创建配置文件时在控制台中创建应用程序配置。有关更多信息，请参阅创建配置和配置文件。	2020 年 2 月 13 日
仅自动批准在指定日期之前发布的补丁	除了在发布补丁指定天数后自动批准安装补丁的选项外，Patch Manager 现在还支持仅自动批准在您指定的日期或之前发布的补丁。例如，如果将 2020 年 7 月 7 日指定为补丁基准的截止日期，则不会自动安装在 2020 年 7 月 8 日或之后发布的任何补丁。有关更多信息，请参阅关于自定义基准和使用自定义补丁基准（控制台）。	2020 年 2 月 12 日

在维护时段任务中使用 {{RESOURCE_ID}} 虚拟参数	注册维护时段任务时，您可以指定对任务类型唯一的参数。您可以使用伪参数语法引用特定值，例如 {{TARGET_ID}} 、{{TARGET_TYPE}} 和 {{WINDOW_TARGET_ID}} 。维护时段任务运行时，它传递正确的值而不是伪参数占位符。要支持将属于某个资源组的资源作为目标，您可以使用 {{RESOURCE_ID}} 虚拟参数传递资源的值，例如 DynamoDB 表、S3 存储桶和其他受支持的类型。有关更多信息，请参阅教程：创建和配置维护时段 (AWS CLI) 中的以下主题： • 注册维护时段任务时使用伪参数 • 示例：向维护时段注册任务	2020 年 2 月 6 日
快速重新运行命令	Systems Manager 包含两个选项，可帮助您从 AWS Systems Manager 控制台中的 Run Command 页面重新运行命令。Rerun (重新运行)：利用此按钮，您可以运行同一个命令而不对其进行更改。复制到新项目：此按钮将一个命令的设置复制到一个新命令，并为您提供在运行该命令之前编辑这些设置的选项。有关更多信息，请参阅重新运行命令。	2020 年 2 月 5 日

从高级实例套餐还原到标准实例套餐	如果以前您已将混合环境中运行的所有本地实例配置为使用高级实例套餐，现在您即可快速配置这些实例以使用标准实例套餐。恢复到标准实例套餐适用于 AWS 账户和单个 AWS 区域中的所有混合实例。恢复到标准实例套餐会影响某些 Systems Manager 工具的可用性。有关更多信息，请参阅从高级实例套餐还原到标准实例套餐。	2020 年 1 月 16 日
用于在安装补丁后跳过实例重启的新选项	在过去，Patch Manager 在托管式实例上安装补丁后始终会重启这些实例。SSM 文档 `AWS-RunPatchBaseline` 中的新 `RebootOption` 参数允许您指定是否希望在安装新补丁后自动重启实例。有关更多信息，请参阅主题关于 SSM 文档 `AWS-RunPatchBaseline` 中的参数名称：RebootOption。	2020 年 1 月 15 日
新主题：“在 Linux 实例上运行 PowerShell 脚本”	新主题介绍如何使用 Run Command 在 Linux 实例上运行 PowerShell 脚本。有关更多信息，请参阅在 Linux 实例上运行 PowerShell 脚本。	2020 年 1 月 10 日
“配置 SSM Agent 以使用代理”的更新	配置 SSM Agent 以使用代理时要指定的值已更新，以反映适用于 HTTP 代理服务器和 HTTPS 代理服务器的选项。有关更多信息，请参阅配置 SSM Agent 以使用代理。	2020 年 1 月 9 日

新的“安全”章节概述了用于保护 Systems Manager 资源的实践	《AWS Systems Manager 用户指南》中新增了一个安全性章节，可帮助您了解如何在使用 Systems Manager 时应用责任共担模式。本章中的主题说明如何配置 Systems Manager 以实现您的安全性和合规性目标。您还将了解如何使用其他 AWS 服务以帮助您监控和保护 Systems Manager 资源。 **Note** 作为此更新的一部分，已将用户指南的“身份验证和访问控制”一章替换为一个新的、更简明的部分，即 AWS Systems Manager 的身份和访问管理。	2019 年 12 月 24 日
新的示例自定义自动化运行手册	已在用户指南中添加一组示例自定义自动化运行手册。这些示例说明如何使用各种自动化操作来简化部署、故障排除和维护任务，并且旨在帮助您编写自己的自定义自动化运行手册。有关更多信息，请参阅自定义自动化运行手册示例。您还可以在 Systems Manager 控制台中查看 Amazon 托管自动化运行手册内容。有关更多信息，请参阅 Systems Manager 自动化运行手册参考。	2019 年 12 月 23 日

对 Oracle Linux 的支持	Systems Manager 现在支持 Oracle Linux 7.5 和 7.7。有关在适用于 Oracle Linux 实例的 EC2 实例上手动安装 SSM Agent 的信息，请参阅 Oracle Linux。有关在混合环境中的 Oracle Linux 服务器上安装 SSM Agent 的信息，请参阅如何在混合 Linux 节点上安装 SSM Agent。	2019 年 12 月 19 日
从 Amazon EC2 控制台启动 Session Manager 会话	现在，您可以从 Amazon Elastic Compute Cloud (Amazon EC2) 控制台启动 Session Manager 会话。从 Amazon EC2 控制台处理与会话相关的任务需要用户和管理员的不同的 IAM 权限。您可以提供仅使用 Session Manager 控制台和 AWS CLI、仅使用 Amazon EC2 控制台或使用这三项工具所需的权限。有关更多信息，请参阅以下主题。 • Session Manager 默认 IAM policy 快速入门 • 启动会话（Amazon EC2 控制台）	2019 年 12 月 18 日

CloudWatch 支持 Run Command 指标和告警	现在，AWS Systems Manager 会将与 Run Command 命令的状态有关的指标发布到 CloudWatch，使您能够根据这些指标设置告警。可以跟踪其指标的命令的终端状态值包括 `Success`、`Failed` 和 `Delivery Timed Out`。有关更多信息，请参阅使用 Amazon CloudWatch 监控 Run Command 指标。	2019 年 12 月 17 日
Systems Manager 新工具：Change Calendar	使用 Systems Manager Change Calendar 可指定要限制或阻止对资源进行代码更改（例如源自 Systems Manager 自动化运行手册或 AWS Lambda 函数）的时间段（事件）。更改日历是一种新的 Systems Manager 文档类型，以明文格式存储 iCalendar 2.0 数据。有关更多信息，请参阅 AWS Systems Manager 更改日历。	2019 年 12 月 11 日

Systems Manager 新工具：AWSAppConfig	可以使用 AppConfig 创建、管理以及快速部署应用程序配置。AppConfig 支持以受控方式部署到任意大小的应用程序。您可以将 AppConfig 与 EC2 实例上托管的应用程序、AWS Lambda、容器、移动应用程序或 IoT 设备一起使用。为了防止在部署应用程序配置时出现意外错误，AppConfig 包含验证程序。验证程序提供语法或语义检查，以确保要部署的配置正常工作。在配置部署期间，AppConfig 监控应用程序以确保部署成功。如果系统遇到错误或部署启动告警，AppConfig 将回滚更改以最大限度减少对应用程序用户的影响。有关更多信息，请参阅 AWSAppConfig。	2019 年 11 月 25 日

Systems Manager 新工具：Systems Manager Explorer	AWS Systems Manager 是一个可自定义的运营控制面板，用于报告有关 AWS 资源的信息。Explorer 会跨 AWS 区域聚合显示您的 AWS 账户的运营数据（OpsData）。在 Explorer 中，OpsData 包含有关 EC2 实例、补丁合规性详细信息和操作工作项 (OpsItems) 的元数据。Explorer 提供有关如何在业务单位或应用程序之间分配 OpsItems、它们随时间的变化趋势以及它们如何随类别变化的上下文。您可以在 Explorer 中对信息进行分组和筛选，以将重点放在与您相关的项目和需要采取措施的项目上。在查找高优先级问题时，您可以使用 Systems Manager OpsCenter 运行自动化运行手册并快速解决这些问题。有关更多信息，请参阅 AWS Systems ManagerExplorer。 **Note** 针对 Systems Manager OpsCenter 的设置与针对 Explorer 的设置集成在一起。如果已设置 OpsCenter，您仍然需要完成集成设置以验证设置和选项。如果尚未设置 OpsCenter，则可以使用集成设置来开始使用这两项工具。有关更多	2019 年 11 月 18 日

信息，请参阅 Explorer 和 OpsCenter 入门。

改进的参数搜索功能	现在，如果在您的账户中具有大量参数或者您忘记了参数的确切名称，搜索参数的工具可以轻松查找参数。您可以使用搜索工具按 `contains` 进行筛选。以前，搜索工具仅支持按 `equals` 和 `begins-with` 搜索参数名称。有关更多信息，请参阅搜索 Systems Manager 参数。	2019 年 11 月 15 日

基于控制台的新自动化文档生成器 | 支持在自动化步骤中运行脚本	现在，您可以使用 Systems Manager 自动化构建和共享标准化的操作手册，以确保用户、AWS 账户 和 AWS 区域 之间的一致性。通过使用该功能运行脚本以及使用 Markdown 将内联文档添加到自动化运行手册中，您可以减少错误并消除手动步骤，例如，在 Wiki 中导航编写的过程以及运行终端命令。 有关更多信息，请参阅以下主题。 • 演练：使用文档生成器创建自定义自动化运行手册 • aws:executeScript（自动化操作参考） • 使用文档生成器创建自动化运行手册 • AWS 新闻博客上 Systems Manager 中的新自动化功能	2019 年 11 月 14 日

使用 Distributor 执行就地软件包更新	以前，如果要使用 Distributor 安装软件包更新，唯一的办法是卸载整个软件包并重新安装新版本。现在，您可以选择执行就地更新。在就地更新期间，Distributor 根据您在软件包中包含的更新脚本仅安装自上次安装以来的新文件或更改的文件。在使用该选项时，软件包应用程序在更新期间保持可用状态而不会脱机。有关更多信息，请参阅以下主题。 • 创建软件包 • 安装或更新软件包	2019 年 11 月 11 日
新的 SSM Agent 自动更新功能	只需单击一下，就可以将您的 AWS 账户中的所有实例配置为自动检查并下载新版本的 SSM Agent。为此，请在 AWS Systems Manager 控制台中的 Managed instances (托管式实例) 页面上选择 Agent auto update (代理自动更新)。有关更多信息，请参阅自动实施 SSM Agent的更新。	2019 年 11 月 5 日

使用 AWS 提供的标签限制 Session Manager 访问	现在可以使用第二种方法来控制用户对会话操作的访问。通过这种新方法，您可以使用 AWS 提供的会话标签而不是使用 `{aws:username}` 变量来创建 IAM 访问策略。通过使用这些由 AWS 提供的会话标签，使用联合 ID 的组织可以控制用户对会话的访问。有关更多信息，请参阅允许用户仅终止其启动的会话。	2019 年 10 月 2 日
应用 Ansible Playbook 的新 SSM 命令文档	您可以使用 `AWS-ApplyAnsiblePlaybooks` 文档创建运行 Ansible Playbook 的 State Manager 关联。本文档为运行 Playbook 提供了以下好处： • 支持运行复杂的 Playbook • 支持从 GitHub 和 Amazon Simple Storage Service（Amazon S3）下载 Playbook • 支持压缩的 PlayBook 结构 • 增强的日志记录 • 捆绑 Playbook 时可以指定要运行的 Playbook 有关更多信息，请参阅创建自动运行 Ansible Playbook 的关联	2019 年 9 月 24 日

Session Manager 的端口转发支持	Session Manager 现在支持端口转发会话。通过使用端口转发，您可以在私有子网中部署的实例之间安全地创建隧道，而无需在服务器上启动 SSH 服务，在安全组中打开 SSH 端口或使用堡垒主机。与 SSH 隧道类似，您可以通过端口转发在笔记本电脑和实例上的打开端口之间转发流量。在配置端口转发后，您可以连接到本地端口，并访问在实例中运行的服务器应用程序。有关更多信息，请参阅以下主题： • AWS 新闻博客上的使用 AWS Systems ManagerSession Manager 进行端口转发 • 启动会话（端口转发）	2019 年 8 月 29 日
指定原定设置参数层或自动选择层	您现在可以指定默认参数层，以用于创建或更新未指定层的参数的请求。您可以将默认层设置为标准参数、高级参数或新的选项“智能分层”。智能分层评估每个 PutParameter 请求，并仅在需要时创建高级参数。（如果参数值大小超过 4KB，参数策略与参数关联，或者已创建标准层最多支持的 1 万个参数，则需要使用高级参数。）有关指定默认层和使用智能分层的更多信息，请参阅指定默认参数层。	2019 年 8 月 27 日

使用 CLI 和 PowerShell 过程更新了“使用关联”部分	更新了“使用关联”部分，以包含使用 AWS CLI 或 AWS Tools for PowerShell 管理关联的过程文档。有关更多信息，请参阅在 Systems Manager 中使用关联。	2019 年 8 月 26 日
使用 CLI 和 PowerShell 过程更新了“使用自动化执行”部分	更新了“使用自动化执行”部分，以包含使用 AWS CLI 或 AWS Tools for PowerShell 运行自动化工作流的过程文档。有关更多信息，请参阅使用自动化执行。	2019 年 8 月 20 日
OpsCenter 与 Application Insights 集成	OpsCenter 与适用于 .NET 和 SQL Server 的 Amazon CloudWatch Application Insights 进行集成。因此，您可以为应用程序中检测到的问题自动创建 OpsItems。有关如何配置 Application Insights 以创建 OpsItems 的信息，请参阅 Amazon CloudWatch 用户指南中的设置、配置和管理用于监控的应用程序。	2019 年 8 月 7 日

新的控制台功能：AWS Systems ManagerQuick Setup	Quick Setup是 Systems Manager 控制台中的一项新功能，可以帮助您在 EC2 实例上快速配置一些 Systems Manager 组件。具体来说，快速设置帮助您使用标签在所选实例或目标实例上配置以下组件： • Systems Manager 的 AWS Identity and Access Management (IAM) 实例配置文件角色。 • 每两个月一次的计划 SSM Agent 更新。 • 每 30 分钟一次的计划清单元数据收集。 • 每天扫描一次实例以查找缺少的补丁。 • 一次性的 Amazon CloudWatch 代理安装和配置。 • 每月一次的计划 CloudWatch 代理更新。 有关更多信息，请参阅 AWS Systems Manager 快速设置。	2019 年 8 月 7 日

将资源组注册为维护时段目标	除了注册托管式实例作为维护时段的目标之外，您现在还可以注册资源组作为维护时段目标。Maintenance Windows支持受到 AWS Resource Groups 支持的所有 AWS 资源类型，包括 `AWS::EC2::Instance` 、`AWS::DynamoDB::Table` 、`AWS::OpsWorks::Instance` 、`AWS::Redshift::Cluster` 等。在此版本中，您还可以将命令发送到资源组，例如，使用 Run Command 控制台或 AWS CLI send-command 命令。有关更多信息，请参阅以下主题： • 为维护时段分配目标（控制台） • 示例：向维护时段注册目标 • 使用目标和速率控制将命令发送到队列	2019 年 7 月 23 日
使用 AWS Systems ManagerDistributor 简化了软件包创建和版本控制	Distributor 有一个新的、简化的软件包创建工作流，可为您生成软件包清单、脚本和文件哈希值。在将版本添加到现有软件包时，也可以使用简化的工作流。	2019 年 7 月 22 日
适用于 Systems Manager 自动化的新的“文档类别”窗格	当您在控制台中运行自动化时，Systems Manager 包括一个新的“文档类别”窗格。使用此窗格根据用途来筛选自动化运行手册。	2019 年 7 月 18 日

支持使用操作系统用户凭证开启 Session Manager 会话	默认情况下，Session Manager 会话是使用在托管实例上创建的系统生成的`ssm-user`账户的证书启动的。在 Linux 计算机上，您现在可以使用操作系统账户的凭证启动会话。有关更多信息，请参阅为 Linux 实例启用“运行身份”支持。	2019 年 7 月 9 日
支持使用 SSH 开启 Session Manager 会话	现在，您可以使用 AWS CLI，通过 Session Manager 在托管实例上启动 SSH 会话。有关允许使用 Session Manager 进行 SSH 会话的信息，请参阅（可选）启用 SSH Session Manager 会话。有关使用 Session Manager 启动 SSH 会话的信息，请参阅启动会话 (SSH)。	2019 年 7 月 9 日
支持在托管式实例上更改密码	您现在可以使用 Systems Manager 在您管理的计算机上重置密码（托管式实例）。您可以使用 Systems Manager 控制台或 AWS CLI 重置密码。有关更多信息，请参阅在托管实例上重置密码。	2019 年 7 月 9 日
修订了“什么是 AWS Systems Manager？”	什么是 AWS Systems Manager？中的介绍内容已扩展，以对服务进行更广泛的介绍，也反映了最近发布的 Systems Manager 工具。此外，该部分中的其他内容已移到各个主题，以便于进行查找。	2019 年 6 月 10 日

Systems Manager 新工具：OpsCenter	OpsCenter 提供了一个中心位置，运营工程师和 IT 专业人员可在该位置查看、调查并解决与 AWS 资源相关的操作工作项 (OpsItems)。OpsCenter 旨在缩短影响 AWS 资源的问题的平均解决时间。此 Systems Manager 工具跨服务聚合和标准化 OpsItems，同时提供有关每个 OpsItem、相关 OpsItems 和相关资源的上下文调查数据。OpsCenter 还提供了可用于快速解决问题的 Systems Manager Automation 运行手册。您可以为每个 OpsItem 指定可搜索的自定义数据。您还可以按状态和源查看自动生成的 OpsItems 相关摘要报告。有关更多信息，请参阅 AWS Systems ManagerOpsCenter。	2019 年 6 月 6 日
对 AWS Management Console 中 Systems Manager 左侧导航窗格的更改	AWS Management Console中的 Systems Manager 左导航窗格包括新的标题，其中有面向 OpsCenter 的新标题，提供对 Systems Manager 工具的更多逻辑分组。	2019 年 6 月 6 日

修订了使用 AWS CLI 创建和配置维护时段的教程	教程：创建和配置维护时段 (AWS CLI) 经过全面修订，以通过实践步骤提供一个简单路径。您可以创建单个维护时段，确定单个目标，并为要运行的维护时段建立一个简单任务。在该过程中，我们提供了您可用来创建自己的任务注册命令的信息和示例，包括使用伪参数（例如 {{TARGET_ID}} ）的信息。有关更多信息及示例，请参阅以下主题： • 示例：向维护时段注册目标 • 示例：向维护时段注册任务 • 关于 register-task-with-maintenance-windows 选项 • 注册维护时段任务时使用伪参数	2019 年 5 月 31 日
有关 SSM Agent 更新的通知	要获得有关 SSM Agent 更新的通知，请在 GitHub 上订阅 SSM Agent 发布说明页面。	2019 年 5 月 24 日

基于 Parameter Store 中的更改接收通知或触发操作	基于 Parameter Store 事件设置通知或触发操作这一主题现在可帮助您设置 Amazon EventBridge 规则，以响应 Parameter Store 中的更改。您可以在发生以下任一情况时接收通知或触发其他操作： • 创建、更新或删除一个参数。 • 创建、更新或删除一个参数标签版本。 • 一个参数过期、即将过期或者在指定时间内未始终未更改。	2019 年 5 月 22 日

设置和开启使用内容的主要修订	我们扩展并重新组织了《AWS Systems Manager 用户指南》中的设置和入门内容。设置内容已被分为两个部分。一个部分重点介绍设置 Systems Manager 以配置和管理您的 EC2 实例的任务。另一个部分重点介绍设置 Systems Manager 以配置并在混合环境中管理您的本地服务器和虚拟机 (VM) 的任务。现在，这两个部分中都按照建议的完成顺序、使用带编号的主要步骤的形式列出所有设置主题。新增的入门章节重点介绍如何在完成账户和服务配置任务后帮助终端用户完成 Systems Manager 入门。 • 设置 AWS Systems Manager • 为混合环境设置 AWS Systems Manager • AWS Systems Manager 入门	2019 年 5 月 15 日

在补丁基准中纳入 Microsoft 发布的应用程序的补丁 (Windows)	现在，Patch Manager 在 Windows Server 实例上支持 Microsoft 发布的应用程序的补丁更新。以前，仅支持 Windows Server 操作系统的补丁。Patch Manager 为 Windows Server 实例提供了两个预定义的补丁基准。补丁基准 `AWS-WindowsPredefinedPatchBaseline-OS` 仅适用于操作系统补丁。`AWS-WindowsPredefinedPatchBaseline-OS-Applications` 同时适用于 Windows Server 操作系统和 Microsoft 在 Windows 上发布的应用程序。有关创建包括 Microsoft 发布的应用程序的补丁的自定义补丁基准的信息，请参阅创建自定义补丁基准中的第一个过程。另外，作为此更新的一部分，AWS 提供的预定义补丁基准的名称也在更改。有关更多信息，请参阅预定义基准。	2019 年 5 月 7 日
使用 AWS CLI 注册维护时段目标的示例	新增主题示例：使用维护时段注册目标提供三个示例命令，以演示在使用 AWS CLI 时您可以用于为维护时段指定目标的不同方式。本主题还介绍了每个示例命令的最佳应用场景。	2019 年 5 月 3 日

补丁组主题更新	主题关于补丁组已更新，加入了有关托管实例如何确定修补操作期间要使用的合适补丁基准的部分。此外，还增加了使用 AWS CLI 或 Systems Manager 控制台向托管式实例添加补丁组或补丁组标签，以及如何向补丁基准添加补丁组或 PatchGroup 的说明。（如果在 EC2 实例元数据中允许使用标签，则必须使用 **PatchGroup** ，且不能使用空格。）有关更多信息，请参阅创建补丁组和向补丁基准添加补丁组。	2019 年 5 月 1 日

新 Parameter Store 功能	Parameter Store 提供以下新功能： • 高级参数：Parameter Store 现在允许您单独配置参数来使用标准参数层（默认层）或高级参数层。高级参数提供更大的参数值大小配额，为每个 AWS 账户 和 AWS 区域 可以创建的参数提供更高的数量配额，并提供使用参数策略的能力。有关高级参数的更多信息，请参阅关于 Systems Manager 高级参数。 • 参数策略：参数策略让您可以将特定条件分配到参数（如到期日期或存活时间），通过这种方式来帮助您管理一组不断增加的参数。参数策略在强制您更新或删除 Parameter Store 中存储的密码和配置数据时尤其有用。参数策略仅可用于使用高级参数层的参数。有关更多信息，请参阅使用参数策略。 • 更高的吞吐量：您现在可以将 Parameter Store 吞吐量配额提高到最大 1000 个事务/秒。有关更多信息，请参阅增加 Parameter Store 吞吐量。	2019 年 4 月 25 日

自动化部分更新	自动化 部分已针对提高的可发现性进行了更新。此外，自动化部分还增加了三个新主题： • 要按步骤运行自动化 • 运行需要审批的自动化 • 使用 State Manager 关联调度自动化	2019 年 4 月 17 日
使用 AWS KMS 密钥加密会话数据	默认情况下，Session Manager 使用 TLS 1.2 来加密您的账户中用户的本地计算机与您的 EC2 实例之间传输的会话数据。现在，您可以选择使用 AWS Key Management Service 中已创建的 AWS KMS key进一步加密该数据。您可以使用您的 AWS 账户中创建的 KMS 密钥，或来自另一个账户的与您共享的密钥。有关指定用于加密会话数据的 KMS 密钥的信息，请参阅启用会话数据的 AWS KMS 密钥加密（控制台）、创建 Session Manager 首选项 (AWS CLI) 或更新 Session Manager 首选项 (AWS CLI)。	2019 年 4 月 4 日

为 AWS Systems Manager 配置 Amazon SNS 通知	增加了使用 AWS CLI 或 Systems Manager 控制台的说明，以便为注册到维护时段的 Run Command 和 Run Command 任务配置 Amazon SNS 通知。有关更多信息，请参阅为 AWS Systems Manager 配置 Amazon SNS 通知。	2019 年 3 月 6 日
混合环境中的服务器和 VM 的高级实例	AWS Systems Manager 为混合环境中的服务器和 VM 提供了标准实例套餐和高级实例套餐。通过标准实例套餐，每个 AWS 区域 内每个 AWS 账户最多可以注册 1000 个服务器或 VM。如果您需要在一个账户和区域中注册超过 1000 个服务器或 VM，则使用高级实例套餐。您可以根据需要在高级实例套餐中创建任意数量的实例，但为 Systems Manager 配置的所有实例均按使用量付费。通过高级实例，您还可以使用 AWS Systems Manager Session Manager 连接到混合计算机。Session Manager 提供了访问实例的交互式 Shell。有关允许高级实例的更多信息，请参阅使用高级实例套餐。	2019 年 3 月 4 日

创建使用共享 SSM 文档的 State Manager 关联	您可以创建使用从其他 AWS 账户共享的 SSM 命令和自动化运行手册的 State Manager 关联。使用共享 SSM 文档创建关联可以帮助您将 Amazon EC2 和混合基础设施保持在一致状态，即使实例不在同一账户内。有关共享 SSM 文档的信息，请参阅 AWS Systems Manager 文档。有关创建 State Manager 关联的信息，请参阅创建关联。	2019 年 2 月 28 日
查看 Amazon EventBridge 规则支持的 Systems Manager 事件的列表	新主题使用 Amazon EventBridge 监控 Systems Manager 事件为您在 Eventbridge 中为其设置事件监控规则的 Systems Manager 发出的各个事件提供了概要介绍。	2019 年 2 月 25 日
在创建 Systems Manager 资源时添加标签	Systems Manager 现在支持在创建标签时将标签添加到某些资源类型。在使用 AWS CLI 或开发工具包创建资源时可以加以标记的资源包括维护时段、补丁基准、Parameter Store 参数和 SSM 文档。在为托管实例创建激活时，您还可以向托管实例分配标签。在使用 Systems Manager 控制台时，您可以为维护时段、补丁基准和参数添加标签。	2019 年 2 月 24 日

为 Systems Manager Inventory 自动创建 IAM 角色	以前，您必须创建 AWS Identity and Access Management (IAM) 角色并向此角色附加单独的策略，才能在控制台中的 Inventory Detail View (Inventory 详细信息视图) 页面上查看清单数据。您不再需要创建此角色或向其附加策略。如果您在 Inventory Detail View（清单详细信息视图）页面上选择了远程数据同步，则 Systems Manager 会自动创建 `Amazon-GlueService PolicyForSSM` 角色，并向其分配 Amazon-GlueService PolicyForSSM-{S3 bucket name} 策略和 AWSGlueServiceRole 策略。有关更多信息，请参阅查询多个区域和账户的清单数据。	2019 年 2 月 14 日
更新 SSM Agent 的 Maintenance Windows 演练	向 Maintenance Windows 文档添加了两个新的演练。这些演练详细介绍如何使用 Systems Manager 控制台或 AWS CLI 创建自动保持 SSM Agent 的最新状态的维护时段。有关更多信息，请参阅 Maintenance Windows 演练。	2019 年 2 月 11 日
使用 Parameter Store 公有参数	增加了介绍 Parameter Store 公有参数的简短内容。有关更多信息，请参阅使用 Systems Manager 公有参数。	2019 年 1 月 31 日

使用 AWS CLI 创建 Session Manager 首选项	增加了使用 AWS CLI 创建 Session Manager 首选项（如 CloudWatch Logs、S3 存储桶日志记录选项和会话加密设置）的说明。有关更多信息，请参阅使用 AWS CLI 创建 Session Manager 首选项。	2019 年 1 月 22 日
使用 State Manager 执行 Systems Manager 自动化工作流	AWS Systems Manager State Manager 现在支持创建使用 SSM 自动化运行手册的关联。State Manager 以前仅支持 `command` (命令) 和 `policy` (策略) 文档，这意味着您只能创建将托管式实例指定为目标的关联。有了对 SSM 自动化运行手册的支持，您现在可以创建将不同类型的 AWS 资源指定为目标的关联。有关更多信息，请参阅使用 State Manager 执行 Systems Manager 自动化工作流。	2019 年 1 月 22 日
Cron 和 Rate 表达式以及维护时段计划选项的参考更新	修订了参考主题适用于 Systems Manager 的 cron 和 rate 表达式。新版本提供更多示例以及有关如何使用 cron 和 rate 表达式计划维护时段和 State Manager 关联的改进说明。此外，新主题 Maintenance Windows 计划和活动期间选项介绍维护时段的各种与计划相关的选项（开始日期、结束日期、时区和计划频率）之间的关联方式。	2018 年 12 月 6 日

启用 SSM Agent 调试日志记录	您可以通过编辑托管式实例上的 seelog.xml.template 文件来启用 SSM Agent 调试日志记录。有关更多信息，请参阅启用 SSM Agent 调试日志记录。	2018 年 11 月 30 日
支持 ARM64 处理器架构	AWS Systems Manager 现在支持 ARM64 版本的 Amazon Linux 2、Red Hat Enterprise Linux 7.6 和 Ubuntu Server（18.04 LTS 和 16.04 LTS）操作系统。有关更多信息，请参阅安装 Amazon Linux 2、RHEL 及 Ubuntu Server 18.04 和 16.04 LTS（带 Snap 软件包）的说明。有关 A1 实例类型的更多信息，请参阅《Amazon EC2 用户指南》中的通用实例。	2018 年 11 月 26 日

使用 AWS Systems ManagerDistributor 创建和部署软件包	使用 AWS Systems Manager Distributor，您可以将自己的软件打包或查找 AWS 提供的代理软件包（如 `AmazonCloudWatchAgent`），以便在 AWS Systems Manager 托管式实例上安装。Distributor 会将软件包等资源发布到 AWS Systems Manager 托管式实例。发布软件包会将软件包文档（在 AWS 账户 中添加软件包时创建的 Systems Manager 文档）的特定版本发布到通过托管式实例 ID、Distributor ID、标签或 AWS 区域标识的托管式实例。有关更多信息，请参阅 AWS Systems ManagerDistributor。	2018 年 11 月 20 日
从中央账户跨多个 AWS 区域和 AWS 账户并发运行 AWS Systems Manager 自动化工作流	您可以从自动化管理账户跨多个 AWS 区域和 AWS 账户或 AWS 组织部门 (OU) 并发运行 AWS Systems Manager自动化工作流。在多个区域和账户或 OU 中并发执行自动化可减少管理 AWS 资源所需的时间，同时提高计算环境的安全性。有关更多信息，请参阅在多个 AWS 区域 和 AWS 账户 中执行自动化工作流。	2018 年 11 月 19 日

查询多个 AWS 区域和 AWS 账户的清单数据	Systems Manager Inventory 与 Amazon Athena 集成，可帮助您查询多个 AWS 区域和 AWS 账户的清单数据。Athena 集成使用资源数据同步，以便您可以在 AWS Systems Manager 控制台的 Inventory Detail View (Inventory 详细信息视图) 页面上查看所有托管式实例的清单数据。有关更多信息，请参阅查询多个区域和账户的清单数据。	2018 年 11 月 15 日
创建运行 MOF 文件的 State Manager 关联	您可以使用 `AWS-ApplyDSCMofs` SSM 文档运行托管对象格式（MOF）文件，以使用 State Manager 在 Windows Server 托管实例上强制实现目标状态。`AWS-ApplyDSCMofs` 文档有两种执行模式。在第一种模式下，您可以配置关联来扫描并报告托管实例当前是否处于指定 MOF 文件中定义的目标状态。在第二种模式下，您可以运行 MOF 文件并根据 MOF 文件中定义的资源及其值更改实例的配置。`AWS-ApplyDSCMofs` 文档让您能够从 Amazon Simple Storage Service (Amazon S3)、本地共享或具有 HTTPS 域的安全网站下载和运行 MOF 配置文件。有关更多信息，请参阅创建运行 MOF 文件的关联。	2018 年 11 月 15 日

在 Session Manager 会话中限制管理访问权限	Session Manager 会话是使用名为 ssm-user 的用户账户（使用默认根或管理员权限创建）的凭证启动的。现在，您可以在开启或关闭 ssm-user 账户管理权限主题中找到有关限制此账户的管理控制权限的信息。	2018 年 11 月 13 日
自动化操作参考中的 YAML 示例	自动化操作参考现在为包含 JSON 示例的每个操作包含一个 YAML 示例。	2018 年 10 月 31 日
为关联分配合规性严重性级别	现在，您可以为 State Manager 关联分配合规性严重性级别。这些严重性级别在合规性控制面板中报告，也可用于筛选合规性报告。您可以分配的严重性级别包括 Critical、High、Medium、Low 和 Unspecified。有关更多信息，请参阅创建关联（控制台）。	2018 年 10 月 26 日
将目标和速率控制与自动化和 State Manager 结合使用	通过使用目标、并发和错误阈值可以控制自动化和 State Manager 关联跨一系列资源的执行。有关更多信息，请参阅使用目标和速率控制在队列上执行自动化工作流和将目标和速率控制与 State Manager 关联结合使用。	2018 年 10 月 23 日

为维护时段指定活动时间范围和国际时区	您还可以指定维护时段在其之前或之后不应运行的日期（开始日期和结束日期），以及维护时段计划所基于的国际时区。有关更多信息，请参阅创建维护时段（控制台）和更新维护时段 (AWS CLI)。	2018 年 10 月 9 日
在 S3 存储桶中为补丁基准维护一个自定义补丁列表	使用 SSM 命令文档 `AWS-RunPatchBaseline` 中的新“InstallOverrideList”参数，您可以指定一个到要安装的补丁列表的 https URL 或 Amazon Simple Storage Service (Amazon S3) 路径类型 URL。此补丁安装列表（以 YAML 格式在 S3 存储桶中维护）会覆盖默认补丁基准指定的补丁。有关更多信息，请参阅参数名称：InstallOverrideList。	2018 年 10 月 5 日
扩展了对是否已安装补丁依赖项的控制	以前，如果已拒绝的补丁列表中的某个补丁被识别为另一个补丁的依赖项，则仍将安装此补丁。现在，您可以选择是安装还是阻止安装这些依赖项。有关更多信息，请参阅创建补丁基准。	2018 年 10 月 5 日

使用条件分支创建动态自动化工作流	`aws:branch` 自动化操作让您能够创建在一个步骤中评估多个选择，然后根据评估结果跳转到自动化运行手册中的不同步骤的动态自动化工作流。有关更多信息，请参阅 Using conditional statements in runbooks（在运行手册中使用条件语句）。	2018 年 9 月 26 日
使用 AWS CLI 更新 Session Manager 首选项	使用 CLI 更新 Session Manager 首选项（如 CloudWatch Logs 和 S3 存储桶日志记录选项）的说明已添加到《AWS Systems Manager 用户指南》中。有关更多信息，请参阅使用 AWS CLI 更新 Session Manager 首选项。	2018 年 9 月 25 日
更新了 Session Manager 的 SSM Agent 要求	Session Manager 现在需要 2.3.68.0 版或更高版本的 SSM Agent。有关 Session Manager 先决条件的更多信息，请参阅满足 Session Manager 先决条件。	2018 年 9 月 17 日

使用 Session Manager 管理实例而无需打开入站端口或维护堡垒主机	使用Session Manager（AWS Systems Manager 中一项完全托管的工具），您可以通过基于浏览器的一键式交互 Shell 或 AWS CLI 来管理 EC2 实例。Session Manager提供安全且可审核的实例管理，无需打开入站端口、维护堡垒主机或管理 SSH 密钥。Session Manager还让您可以遵守需要对实例进行受控访问的公司策略，严格的安全实践以及具有实例访问详细信息的完全可审核日志，同时仍然为终端用户提供对 EC2 实例的简单一键式跨平台访问。有关更多信息，请参阅了解有关 Session Manager 的更多信息。	2018 年 9 月 11 日
从 Systems Manager 自动化工作流调用其他 AWS 服务	您可以在 Automation 运行手册中使用三个新的 Automation 操作（或插件），从而在 Automation 工作流中调用其他 AWS 服务和其他 Systems Manager 工具。有关更多信息，请参阅 Using action outputs as inputs（使用操作输出作为输入）。	2018 年 8 月 28 日
在 IAM policy 中使用特定 Systems Manager 的条件键	主题在策略中指定条件已更新，其中列出了可在策略中包含的 Systems Manager 的 IAM 条件密钥。可以使用这些键指定策略生效所依据的条件。本主题还包括指向示例策略和其他相关主题的链接。	2018 年 8 月 18 日

使用组聚合清单数据以查看配置为和未配置为收集清单类型的实例	利用组，您可以快速查看已配置为以及未配置为收集一个或多个清单类型的托管式实例的计数。利用组，可指定一个或多个清单类型和使用 `exists` 运算符的筛选条件。有关更多信息，请参阅聚合清单数据。	2018 年 8 月 16 日
查看清单和配置合规性的历史记录和变更跟踪	现在可以查看通过托管实例收集的清单的历史记录和变更跟踪。您还可以查看通过配置合规性报告的 Patch Manager 修补和 State Manager 关联的历史记录及更改跟踪。有关更多信息，请参阅查看清单历史记录和变更跟踪。	2018 年 8 月 9 日

Parameter Store 与 Secrets Manager 集成	Parameter Store 现已与 AWS Secrets Manager 集成，您可以在使用其他已支持引用 Parameter Store 参数的 AWS 服务时检索 Secrets Manager 密钥。这些服务包括 Amazon EC2、Amazon Elastic Container Service、AWS Lambda、AWS CloudFormation、AWS CodeBuild、AWS CodeDeploy 和其他 Systems Manager 工具。通过使用 Parameter Store 引用 Secrets Manager 密钥，可创建一致且安全的过程来调用和使用代码，以及配置脚本中的密钥和引用数据。有关更多信息，请参阅引用 Parameter Store 参数中的 AWS Secrets Manager 密钥。	2018 年 7 月 26 日
将标注附上 Parameter Store 参数	参数标签是用户定义的别名，帮助管理参数的不同版本。修改参数时，Systems Manager 将自动保存新版本并将版本号增加 1。标签可帮助您在存在多个版本时记住参数版本的用途。有关更多信息，请参阅标记参数。	2018 年 7 月 26 日

创建动态自动化工作流	默认情况下，您在自动化运行手册的 mainSteps 部分中定义的步骤（或操作）将按先后顺序执行。在一个操作完成后，mainSteps 部分中指定的下一个操作将开始。利用此版本，您现在可以创建执行条件分支的自动化工作流。因此，您可以创建动态响应条件更改并跳转至指定步骤的自动化工作流。有关更多信息，请参阅 Using conditional statements in runbooks（在运行手册中使用条件语句）。	2018 年 7 月 18 日
SSM Agent 现已通过 Snap 预安装到 Ubuntu Server 16.04 AMIs 上	从通过使用 20180627 标识的 Ubuntu Server 16.04 AMIs 创建的实例开始，已使用 Snap 软件包预安装 SSM Agent。在通过较早 AMIs 创建的实例上，您应继续使用 deb 安装软件包。有关更多信息，请参阅关于 64 位 Ubuntu Server 16.04 实例上的 SSM Agent安装。	2018 年 7 月 7 日

查看 SSM Agent 所需的最低 S3 权限	新主题 SSM Agent 的最低 S3 存储桶权限提供了有关资源为执行 Systems Manager 操作可能需要访问的 Amazon Simple Storage Service (Amazon S3) 存储桶的信息。如果要将实例配置文件或 VPC 端点的 S3 存储桶访问权限制为使用 Systems Manager 所需的最小权限，则可在自定义策略中指定这些存储桶。	2018 年 7 月 5 日
查看特定 State Manager 关联 ID 的完整执行历史记录	新主题查看关联历史记录介绍如何查看特定关联 ID 的所有执行，然后查看一个或多个资源的执行详细信息。	2018 年 7 月 2 日
Patch Manager 引入了对 Amazon Linux 2 的支持	现在，您可以使用 Patch Manager 将补丁应用于 Amazon Linux 2 实例。有关 Patch Manager 操作系统支持的一般信息，请参阅 Patch Manager 先决条件。有关在定义补丁筛选条件时 Amazon Linux 2 支持的键/值对的信息，请参阅《AWS Systems Manager API 参考》中的 PatchFilter。	2018 年 6 月 26 日
向 Amazon CloudWatch Logs 发送命令输出	新主题为 Run Command 配置 Amazon CloudWatch Logs 介绍了如何将 Run Command 输出发送到 CloudWatch Logs。	2018 年 18 月 6 日

使用 AWS CloudFormation，快速为 Inventory 创建资源数据同步	您可以使用 AWS CloudFormation 为 Systems Manager Inventory 创建或删除资源数据同步。要使用 AWS CloudFormation，请将 AWS::SSM::ResourceDataSync 资源添加到 AWS CloudFormation 模板。有关更多信息，请参阅《AWS CloudFormation 用户指南》中的使用 AWS CloudFormation 模板。您还可以手动为 Inventory 创建资源数据同步，如为 Inventory 创建资源数据同步中所述。	2018 年 6 月 11 日
现在可通过 RSS 获得 AWS Systems Manager 用户指南更新通知	HTML 版本的 Systems Manager 用户指南现在支持更新的 RSS 馈送，此类更新记载在 Systems Manager Documentation update history (Systems Manager 文档更新历史记录) 页面上。RSS 源包括 2018 年 6 月及此日期之后发布的更新。此前宣布的更新在 Systems Manager Documentation update history (Systems Manager 文档更新历史记录) 页面中仍可用。使用顶部菜单面板中的 RSS 按钮，订阅此源。	2018 年 6 月 6 日

在脚本中指定一个退出代码，以重启托管式实例	新主题通过脚本重启托管式实例介绍了如何通过在您使用 Run Command 运行的脚本中指定退出代码，指示 Systems Manager 重启托管式实例。	2018 年 6 月 3 日
删除自定义清单时在 Amazon EventBridge 中创建事件	新主题在 EventBridge 中查看清单删除操作介绍了如何配置 Amazon EventBridge 以在用户删除自定义清单时创建事件。	2018 年 6 月 1 日

2018 年 6 月之前的更新

下表描述了 2018 年 6 月之前每次发布 AWS Systems Manager 用户指南 时进行的重要更改。

更改	描述	发行日期
在 AWS 账户 中创建所有托管式实例的清单	您可以通过创建全局清单关联，轻松创建您的 AWS 账户 中所有托管式实例的清单。有关更多信息，请参阅 清点 AWS 账户 中的所有托管式节点。 **Note** 全局清单关联在 SSM Agent版本 2.0.790.0 或更高版本中可用。有关如何在实例上更新 SSM Agent 的信息，请参阅使用 Run Command 更新 SSM Agent。	2018 年 5 月 3 日
默认情况下在 Ubuntu Server 18 上安装了 SSM Agent	默认情况下，SSM Agent 安装在 Ubuntu Server 18.04 LTS 64 位和 32 位 AMIs 上。	2018 年 5 月 2 日
新主题	新主题使用指令文档版本运行命令介绍如何使用 document-version 参数指定在运行命令时使用 SSM 文档的哪个版本。	2018 年 5 月 1 日

更改	描述	发行日期
新主题	新主题删除自定义清单介绍了如何使用 AWS CLI 从 Amazon S3 中删除自定义清单数据。此主题还介绍了如何使用 SchemaDeleteOption 通过关闭或删除自定义清单类型来管理自定义清单。此新功能使用 DeleteInventory API 操作。	2018 年 4 月 19 日
适用于 SSM Agent 的 Amazon SNS 通知	您可以订阅 Amazon SNS 主题，以在新版本的 SSM Agent 可用时收到通知。有关更多信息，请参阅 订阅 SSM Agent 通知。	2018 年 4 月 9 日
CentOS 修补支持	Systems Manager 现在支持修补 CentOS 实例。有关支持的 CentOS 版本的信息，请参阅 Patch Manager 先决条件。有关修补工作方式的更多信息，请参阅 Patch Manager 操作是如何工作的。	2018 年 3 月 29 日
新的章节	为了在 AWS Systems Manager 用户指南中提供单个参考信息来源，增加了一个新的章节 AWS Systems Manager 引用。其他内容在推出后也将添加到此部分中。	2018 年 3 月 15 日
新主题	新主题已批准补丁和已拒绝补丁列表的程序包名称格式详细说明了您可以在自定义补丁基准的已批准补丁和已拒绝补丁列表中输入的软件包名称格式。为 Patch Manager 支持的每种操作系统类型提供了示例格式。	2018 年 3 月 9 日
新主题	Systems Manager 现在与 ChefChef InSpec 进行了集成。InSpec 是一个开源的运行时框架，您可以使用该框架在 GitHub 或 Amazon S3 上创建人类可读的配置文件。然后，您可以使用 Systems Manager 运行合规性扫描并查看合规和不合规的实例。有关更多信息，请参阅 将 Chef InSpec 配置文件与 Systems Manager Compliance 结合使用。	2018 年 3 月 7 日
新主题	新主题将服务相关角色用于 Systems Manager介绍了如何将 AWS Identity and Access Management (IAM) 服务相关角色用于 Systems Manager。目前，仅在使用 Systems Manager Inventory 收集有关标签和 Resource Groups 的元数据时，才需要服务相关角色。	2018 年 2 月 27 日

更改	描述	发行日期
新增和更新的主题	您现在可以使用 Patch Manager 来安装与实例上配置的默认源存储库不同的源存储库中的补丁。这可用于使用与安全性无关的更新、Ubuntu Server 的个人软件包存档 (PPA) 的内容、企业内部应用程序的更新等来修补实例。您在创建自定义补丁基准时指定备用补丁源存储库。有关更多信息，请参阅以下主题： • 如何指定备用补丁源存储库 (Linux) • 使用自定义补丁基准 • 创建对不同操作系统版本使用自定义存储库的补丁基准 此外，您现在可以使用 Patch Manager 修补 SUSE Linux Enterprise Server 实例。Patch Manager 支持修补 SLES 12.* 版本（仅限 64 位）。有关更多信息，请参阅以下主题中特定于 SLES 的信息： • 如何选择安全性补丁 • 如何安装补丁 • 补丁基准规则在 SUSE Linux Enterprise Server 上的工作原理	2018 年 2 月 6 日
新主题	新主题适用于修补托管式节点的 SSM 命令文档介绍了可用的七个 SSM 文档，有助于您使托管式实例始终获得最新的安全相关更新补丁。	2018 年 1 月 10 日
有关 Linux 支持的重要更新	在各主题中更新了以下信息： • 在 2017.09 及以后日期版本的 Amazon Linux 1 基本 AMIs 上，默认情况下会安装 SSM Agent。 • 对于其他版本的 Linux（包括经 Amazon ECS 优化的 AMIs 等非基本映像），需要手动安装 SSM Agent。	2018 年 1 月 9 日

更改	描述	发行日期
新主题	新主题用于修补的 SSM 命令文档：AWS-RunPatchBaseline 详细介绍如何在 Windows 和 Linux 系统上使用此 SSM 文档。此外，还介绍了 AWS-RunPatchBaseline 文档中的两个可用参数：Operation 和 Snapshot ID。	2018 年 1 月 5 日
新主题	新增一个部分Patch Manager 操作是如何工作的，其中介绍了一些技术详细信息，说明 Patch Manager 如何确定安装哪些安全补丁，以及如何在每个支持的操作系统上安装这些补丁。此外，还介绍补丁基准规则在不同的 Linux 操作系统分发版上的工作原理	2018 年 1 月 2 日
重新命名并改动了 Systems Manager 自动化操作参考的位置	根据客户反馈，“自动化操作参考”现更名为“Systems Manager 自动化运行手册参考”。此外，我们还将此参考移到了“共享资源”>“文档”节点中，使其更靠近命令文档插件参考。有关更多信息，请参阅 Systems Manager 自动化操作参考。	2017 年 12 月 20 日
新增监控章节和内容	新章节AWS Systems Manager 中的日志记录和监控介绍了如何将指标和日志数据发送到 Amazon CloudWatch Logs。新主题将节点日志发送到统一的 CloudWatch Logs（CloudWatch 代理）介绍了如何将实例（仅限 64 位 Windows Server 实例）上的监控任务从 SSM Agent 迁移到 CloudWatch 代理。	2017 年 12 月 14 日
新增章节	新章节对 AWS Systems Manager 进行身份和访问管理全面介绍了如何使用 AWS Identity and Access Management (IAM) 和 AWS Systems Manager，以通过使用凭证帮助保护对资源的访问。这些凭证提供访问 AWS 资源所需的权限，如访问存储在 S3 存储桶中的数据、向 EC2 实例发送命令和读取 EC2 实例上的标签。	2017 年 12 月 11 日
左侧导航窗格的改动	本用户指南左侧导航窗格中的标题更改为与新 AWS Systems Manager 控制台中的标题一致。	2017 年 12 月 8 日

更改	描述	发行日期
因 re:Invent 2017 而产生的多次更改	• 正式发布的 AWS Systems Manager：AWS Systems Manager（以前称作 Amazon EC2 Systems Manager）是一个统一接口，您可通过该接口轻松地将操作数据集中到一起，并跨 AWS 资源自动执行任务。您可以在此处访问新的 AWS Systems Manager 控制台。有关更多信息，请参阅 什么是 AWS Systems Manager？。 • YAML 支持：您可以创建 YAML 格式的 SSM 文档。有关更多信息，请参阅 AWS Systems Manager 文档。	2017 年 11 月 29 日
使用 Run Command 为 EBS 卷拍摄启用 VSS 的快照	使用 Run Command，您可以为附加到 Amazon EC2 Windows 实例上的所有 Amazon Elastic Block Store (Amazon EBS) 卷拍摄应用程序一致性快照。快照过程使用 Windows 卷影复制服务（VSS）为 VSS 感知应用程序拍摄映像级备份，包括这些应用程序和磁盘之间的待处理事务中的数据。此外，当需要备份所有已连接的卷时，无需关闭实例或断开与它们的连接。有关更多信息，请参阅《Amazon EC2 用户指南》中的使用 AWS Systems Manager 拍摄启用 Microsoft VSS 的快照。	2017 年 11 月 20 日
通过使用 VPC 端点增强可用的 Systems Manager 安全性	您可以配置 Systems Manager 以使用接口 VPC 端点，改善托管式实例（包括混合环境中的托管式实例）的安保状况。接口端点由 PrivateLink 提供支持，该技术使您能够通过使用私有 IP 地址私下访问 Amazon EC2 和 Systems Manager API。PrivateLink 将托管式实例、Systems Manager 和 EC2 之间的所有网络流量限制在 Amazon 网络内（托管式实例无法访问 Internet）。而且，您无需 Internet 网关、NAT 设备或虚拟专用网关。有关更多信息，请参阅使用适用于 Systems Manager 的 VPC 端点提高 EC2 实例的安全性。	2017 年 11 月 7 日

更改	描述	发行日期
针对文件、服务、Windows 角色和 Windows 注册表的清单支持	SSM Inventory 现在支持从您的托管式实例中收集以下信息。 • 文件：名称、大小、版本、安装日期、修改时间和上次访问时间等。 • 服务：名称、显示名称、状态、相关服务、服务类型、启动类型等。 • Windows 注册表：注册表项路径、值名称、值类型和值。 • Windows 角色：名称、显示名称、路径、功能类型、安装状态等。 在尝试为这些清单类型收集信息之前，请更新需要清点的实例上的 SSM Agent。通过运行最新版本的 SSM Agent，可确保您能够收集所有支持的清单类型的元数据。有关如何使用 State Manager 更新 SSM Agent 的信息，请参阅 演练：使用 AWS CLI 自动更新 SSM Agent。 有关清单的更多信息，请参阅 了解有关 Systems Manager Inventory 的更多信息。	2017 年 11 月 6 日
自动化文档更新	修复了 Systems Manager 自动化访问权限设置和配置相关信息中的几个问题。有关更多信息，请参阅 设置自动化。	2017 年 10 月 31 日

更改	描述	发行日期
GitHub 与 Amazon S3 集成	运行远程脚本：Systems Manager 现在支持从私有或公有 GitHub 存储库以及从 Amazon S3 下载并运行脚本。使用 `AWS-RunRemoteScript` 预定义 SSM 文档或自定义 SSM 文档中的 `aws:downloadContent` 插件，可以运行 Ansible Playbook 以及 Python、Ruby 或 PowerShell 等（仅举几例）中的脚本。当您使用 Systems Manager 在混合环境中自动配置和部署 EC2 实例和本地托管式实例时，这些更改将进一步增强基础设施即代码。有关更多信息，请参阅 从 GitHub 运行脚本 和 从 Amazon S3 运行脚本。 创建复合 SSM 文档：Systems Manager 现在支持从一个主要 SSM 文档运行一个或多个次要 SSM 文档。这些运行其他文档的主要文档称为复合 文档。通过使用复合文档，您可以为多个 AWS 账户创建并共享一组标准的次要 SSM 文档，从而完成引导启动防病毒软件或域连接实例等常见任务。您可以运行 Systems Manager、GitHub 或 Amazon S3 中存储的复合或次要文档。在创建复合文档后，可以使用 `AWS-RunDocument` 预定义 SSM 文档运行它。有关更多信息，请参阅创建复合文档和从远程位置运行文档。 SSM 文档插件参考：为了更便于访问，我们将 SSM 文档的 SSM 插件参考从 Systems Manager API 参考中移到用户指南中。有关更多信息，请参阅 命令文档插件参考。	2017 年 10 月 26 日
Parameter Store 中的参数版本支持	在编辑参数时，Parameter Store 现在会自动将其版本号增加 1。您可以在 API 调用和 SSM 文档中指定参数名和特定版本号。如果不指定版本号，系统自动使用最新版本。 参数版本针对意外更改参数的情况提供额外保护。您可以查看所有版本的值，并在必要时引用旧版本。还可以使用参数版本查看一段时间内更改参数的次数。有关更多信息，请参阅 处理 Parameter Store 中的参数版本。	2017 年 10 月 24 日

更改	描述	发行日期
支持标记 Systems Manager 文档	现在，您可以使用 AddTagsToResource API、AWS CLI 或 AWS Tools for PowerShell 通过键/值对标记 Systems Manager 文档。通过添加标签，可帮助您根据为特定资源分配的标签快速确定它们。这是对现有的对托管实例、维护时段、Parameter Store 参数和补丁基准的标签支持的补充。	2017 年 10 月 3 日
用于根据反馈修复错误或更新内容的各种文档更新	• 使用适用于 Raspbian Linux 的信息更新了使用 Systems Manager 管理混合和多云环境中的节点。 • 使用适用于 Windows Server 实例的新要求更新了使用 Systems Manager 管理 EC2 实例。SSM Agent 需要 Windows PowerShell 3.0 或更高版本才能在 Windows Server 实例上运行特定 SSM 文档（例如，早期 `AWS-ApplyPatchBaseline` 文档）。验证您的 Windows Server 实例是否在 Windows Management Framework 3.0 或更高版本上运行。该框架包括 PowerShell。有关更多信息，请参阅 Windows 管理框架 3.0。	2017 年 10 月 2 日
使用 EC2Rescue 自动化工作流对无法访问的 Windows 实例进行故障排除	EC2Rescue 可以帮助您诊断并解决有关 Amazon EC2 Windows Server 实例的问题。您可以使用 AWSSupport-ExecuteEC2Rescue 文档将该工具作为 Systems Manager 自动化工作流运行。AWSSupport-ExecuteEC2Rescue 文档旨在执行 Systems Manager 操作、AWS CloudFormation 操作和 Lambda 函数的组合，从而自动执行使用 EC2Rescue 通常所需的步骤。有关更多信息，请参阅 在无法访问的实例上运行 EC2Rescue 工具。	2017 年 9 月 29 日
SSM Agent 默认安装在 Amazon Linux 上。	在 2017.09 及以后日期版本的 Amazon Linux AMIs 上，默认情况下会安装 SSM Agent。需要在其他版本的 Linux 上手动安装 SSM Agent，如在适用于 Linux 的 EC2 实例上使用 SSM Agent中所述。	2017 年 9 月 27 日

更改	描述	发行日期
Run Command 增强功能	Run Command 包含以下增强功能。 • 您可以通过创建和分配 IAM policy 将命令执行限制为针对特定实例，该策略包括一个条件，即用户只能对使用特定 Amazon EC2 标签标记的实例运行命令。有关更多信息，请参阅 根据标签限制 Run Command 访问。 • 使用 Amazon EC2 标签，您可以有更多的选择来将实例设为目标。现在，您可以在发送命令时指定多个标签键和多个标签值。有关更多信息，请参阅 大规模运行命令。	2017 年 9 月 12 日
Raspbian 上支持 Systems Manager	Systems Manager 现在可以在 Raspbian Jessie 和 Raspbian Stretch 设备（包括 Raspberry Pi（32 位））上运行。	2017 年 9 月 7 日
自动将 SSM Agent 日志发送到 Amazon CloudWatch Logs	您现在可以在实例上进行简单的配置更改，使 SSM Agent 将日志文件发送到 CloudWatch。有关更多信息，请参阅 将 SSM Agent 日志发送到 CloudWatch Logs。	2017 年 9 月 7 日
加密资源数据同步	借助 Systems Manager 资源数据同步，您可以将在几十个或几百个托管式实例上收集的清单数据聚合到一个中央 S3 存储桶。您现在可以使用 AWS Key Management Service 密钥加密资源数据同步。有关更多信息，请参阅 演练：使用资源数据同步聚合清单数据。	2017 年 9 月 1 日
新的 State Manager 演练	向 State Manager 文档添加了两个新的演练： 演练：使用 AWS CLI 自动更新 SSM Agent 演练：在适用于 Windows Server 的 EC2 实例上自动更新半虚拟化驱动程序	2017 年 8 月 31 日

更改	描述	发行日期
Systems Manager 配置合规性	使用配置合规性扫描托管实例队列，了解补丁合规性和配置不一致性。您可以从多个 AWS 账户 和 AWS 区域 中收集并聚合数据，然后深入了解不合规的特定资源。默认情况下，配置合规性将显示有关 Patch Manager 修补和 State Manager 关联的合规性数据。您也可以根据 IT 或业务要求自定义服务并创建自己的合规性类型。有关更多信息，请参阅 AWS Systems Manager Compliance。	2017 年 8 月 28 日
新的自动化操作：`aws:executeAutomation`	通过调用辅助自动化运行手册运行辅助自动化工作流。借助此操作，您可以为最常见的工作流创建自动化运行手册，并在自动化执行期间引用这些文档。因为无需跨类似运行手册复制步骤，此操作可以简化您的自动化运行手册。有关更多信息，请参阅 `aws:executeAutomation` - 运行另一个自动化。	2017 年 8 月 22 日
自动化作为 CloudWatch 事件的目标	您可通过指定自动化运行手册作为 Amazon CloudWatch 事件的目标来启动自动化工作流。您可以按计划或者在特定 AWS 系统事件发生时启动工作流。有关更多信息，请参阅基于 EventBridge 事件运行自动化。	2017 年 8 月 21 日
State Manager 关联版本控制和常规更新	您现在可以创建不同的 State Manager 关联版本。每个关联具有 1000 个版本的配额。您还可为您的关联指定名称。此外，State Manager 文档已更新，处理了过时信息和不一致之处。有关更多信息，请参阅 AWS Systems Manager State Manager。	2017 年 8 月 21 日

更改	描述	发行日期
对Maintenance Windows的更改	Maintenance Windows包括以下更改或增强： • 以前，Maintenance Windows只能通过使用 Run Command 执行任务。现在，您可以使用 Systems Manager、自动化、AWS Lambda 和 AWS Step Functions 来执行任务。 • 您可编辑维护时段的目标，指定目标名称、描述和所有者。 • 您可以编辑维护时段中的任务，包括为 Run Command 和自动化任务指定新的 SSM 文档。 • 现在支持所有 Run Command 参数，包括 DocumentHash、DocumentHashType、TimeoutSeconds、Comment 和 NotificationConfig。 • 您现在可在尝试取消注册目标时使用 `safe` 标记。如果启用，系统将在任何任务引用目标时返回错误。 有关更多信息，请参阅 AWS Systems Manager Maintenance Windows。	2017 年 8 月 16 日
新的自动化操作：`aws:approve`	自动化运行手册的这一新操作会临时暂停自动化执行，直到指定主体批准或拒绝操作。在达到所需批准数后，自动化执行将恢复。 有关更多信息，请参阅 Systems Manager 自动化操作参考。	2017 年 8 月 10 日

更改	描述	发行日期
不再需要自动化担任角色	自动化 之前需要您指定服务角色 (或担任 角色)，以便服务有权代表您执行操作。自动化 不再需要此角色，因为服务现在通过使用已调用执行的用户的上下文来操作。 不过，以下情况仍需要您为 自动化 指定服务角色： • 当您想限制用户对资源的权限，但希望用户运行需要提升权限的 自动化 工作流时。在此方案中，您可以创建具有提升权限的服务角色并允许用户运行此工作流。 • 运行时长预计将超过 12 小时的操作需要一个服务角色。 有关更多信息，请参阅 设置自动化。	2017 年 8 月 3 日
配置合规性	使用 Amazon EC2 Systems Manager 配置合规性扫描托管式实例队列，了解补丁合规性和配置不一致性。您可以从多个 AWS 账户 和 AWS 区域 中收集并聚合数据，然后深入了解不合规的特定资源。有关更多信息，请参阅 AWS Systems Manager Compliance。	2017 年 8 月 8 日
SSM 文档增强功能	SSM 命令和策略文档现在提供跨平台支持。因此，单个 SSM 文档可以处理 Windows 和 Linux 操作系统的插件。通过使用跨平台支持，您可以合并管理的文档数量。使用 2.2 版或更高版本架构的 SSM 文档中提供跨平台支持。 使用 2.0 版或更高版本架构的 SSM 命令文档现在包含相同类型的多个插件。例如，您可以创建多次调用 `aws:runRunShellScript` 插件的命令文档。 有关 2.2 版架构变更的更多信息，请参阅 AWS Systems Manager 文档。有关 SSM 插件的更多信息，请参阅命令文档插件参考。	2017 年 7 月 12 日

更改	描述	发行日期
Linux 修补	Patch Manager 现在可以修补以下 Linux 发行版： 64 位和 32 位系统 • Amazon Linux 2014.03、2014.09 或更高版本 • Ubuntu Server 16.04 LTS、14.04 LTS 或 12.04 LTS • Red Hat Enterprise Linux (RHEL) 6.5 或更高版本 仅 64 位系统 • Amazon Linux 2015.03、2015.09 或更高版本 • Red Hat Enterprise Linux (RHEL) 7.x 或更高版本 有关更多信息，请参阅 AWS Systems Manager Patch Manager。 **Note** • 要修补 Linux 实例，您的实例必须运行 SSM Agent 2.0.834.0 版或更高版本。有关更新此代理的信息，请参阅中标题为 SSM Agent 示例：更新从控制台运行命令的部分。 • `AWS-ApplyPatchBaseline` SSM 文档将替换为 `AWS-RunPatchBaseline` 文档。	2017 年 6 月 7 日
资源数据同步	您可以使用 Systems Manager 资源数据同步，将从所有托管式实例收集的清单数据发送到单个 Amazon S3 存储桶。在收集新的清单数据时，资源数据同步自动更新集中式数据。所有清单数据存储在目标 S3 存储桶中后，您可以使用 Amazon Athena 和 Amazon QuickSight 等服务查询和分析聚合数据。有关更多信息，请参阅为 Inventory 创建资源数据同步。有关如何使用资源数据同步的示例，请参阅 演练：使用资源数据同步聚合清单数据。	2017 年 6 月 29 日

更改	描述	发行日期
Systems Manager 参数层次结构	以平面列表的形式管理几十个或数百个 Systems Manager 参数十分耗时且容易出错。您可以使用参数层次结构来帮助组织和管理 Systems Manager 参数。一个层次结构是一个参数名称，包括您使用正斜杠定义的路径。下面是一个示例，它在名称中使用三个层次结构级别来标识以下内容： /环境/计算机类型/应用程序/数据 `/Dev/DBServer/MySQL/db-string13` 有关更多信息，请参阅 处理 Parameter Store 中的参数层次结构。	2017 年 6 月 22 日
SSM Agent 对于 SUSE Linux Enterprise Server的支持	您可以在 64 位 SUSE Linux Enterprise Server (SLES) 上安装 SSM Agent。有关更多信息，请参阅 在适用于 Linux 的 EC2 实例上使用 SSM Agent。	2017 年 6 月 14 日

文档惯例

以下是 AWS Systems Manager 用户指南的通用印刷惯例。

针对本地操作系统或命令行语言的差异化示例

我们使用选项卡来根据用户的本地操作系统类型显示命令的不同示例。对于 Linux 和 macOS 示例，我们使用反斜杠 (\) 字符将长命令分解为多行。对于 Windows Server 示例，我们使用脱字号 (^) 字符将命令分解为多行。

例如：

Linux & macOS

```
aws ssm update-service-setting \
    --setting-id arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-
instance/activation-tier \
    --setting-value advanced
```

Windows

```
aws ssm update-service-setting ^
    --setting-id arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-
instance/activation-tier ^
    --setting-value advanced
```

用户界面中的元素

格式：粗体文本

示例：选择文件、属性。

用户输入 (用户键入的文本)

格式：文本采用等宽字体

示例：对于名称，键入 **my-new-resource**。

必需值的占位符文本

格式：*##* 文本

例如：

```
aws ec2 register-image --image-location amzn-s3-demo-bucket/image.manifest.xml
```