使用者指南

AWS Systems Manager

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

AWS Systems Manager: 使用者指南

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

Amazon 的商標和商業外觀不得用於任何非 Amazon 的產品或服務，也不能以任何可能造成客戶混淆、任何貶低或使 Amazon 名譽受損的方式使用 Amazon 的商標和商業外觀。所有其他非 Amazon 擁有的商標均為其各自擁有者的財產，這些擁有者可能隸屬於 Amazon，或與 Amazon 有合作關係，或由 Amazon 贊助。

Table of Contents

什麼是 AWS Systems Manager？ 1
Systems Manager 如何讓我的營運獲得益處？ 1
哪些人員應該使用 Systems Manager？ 2
Systems Manager 有哪些主要功能？ 2
支援的 AWS 區域 3
存取 Systems Manager 4
Systems Manager 服務名稱歷程記錄 5
支援的作業系統和機器類型 5
Systems Manager 支援的作業系統 5
混合多雲端環境中支援的機器類型 12
什麼是統一主控台？ 12
設定 AWS Systems Manager的受管節點 15
使用 Systems Manager 管理 EC2 執行個體 15
設定 Systems Manager 所需的執行個體許可 16
使用適用於 Systems Manager 的 VPC 端點來改善 EC2 執行個體的安全性 26
使用 Systems Manager 在混合多雲端環境中管理節點 30
在混合多雲端環境中建立 Systems Manager 所需的 IAM 服務角色 31
建立混合啟用，以向 Systems Manager 註冊節點 39
在混合 Linux 節點上安裝 SSM Agent 44
在混合 Windows Server 節點上安裝 SSM Agent 50
使用 Systems Manager 管理邊緣裝置 55
為邊緣裝置建立 IAM 服務角色 56
設定 的邊緣裝置 AWS IoT Greengrass 61
更新 AWS IoT Greengrass 權杖交換角色，並在SSM Agent您的邊緣裝置上安裝 61
為 Systems Manager 建立 AWS Organizations 委派管理員 62
將委派管理員與 Change Manager 搭配使用 62
將委派管理員與 Explorer 搭配使用 63
將委派管理員與 OpsCenter 搭配使用 63
將委派管理員與 Quick Setup 搭配使用 63
一般設定 64
註冊 AWS 帳戶 64
建立具有管理存取權的使用者 64
設定 AWS Systems Manager 66
設定 Systems Manager 主控台存取 66

Systems Manager 加入政策 66
AWS Systems Manager 主控台運算子政策 72
AWS Systems Manager 主控台運算子唯讀政策 76
為組織設定 Systems Manager 統一主控台 78
先決條件 78
統一主控台資源 79
設定統一主控台 81
針對單一帳戶和區域設定 Systems Manager 統一主控台 82
統一主控台資源 83
設定統一主控台 84
停用 Systems Manager 統一主控台 85
執行節點管理任務 87
檢閱節點洞察 87
新增或移除小工具 90
重新排列小工具 90
探索節點 91
使用主控台篩選條件探索節點 92
在 Amazon Q 中使用文字提示探索節點 96
檢視個別節點詳細資訊並對節點採取動作 103
下載或匯出受管節點報告 104
管理節點報告內容和外觀 108
使用 Systems Manager Just-in-time節點存取 108
使用 Systems Manager just-in-time存取 110
啟動just-in-time節點存取工作階段 154
管理just-in-time存取請求 156
從 移至just-in-time節點存取 Session Manager 157
使用 Systems Manager 停用just-in-time存取 161
常見問答集 161
診斷和修復 163
診斷和修復失敗部署 164
診斷和修復漂移組態 164
診斷和修復未受管的 Amazon EC2 執行個體 165
執行手冊動作的修復影響類型 171
檢視修復的詳細執行歷史記錄 173
調整 Systems Manager 設定 174
帳戶設定 175

組織設定 175
功能組態 175
診斷和修復設定 179
使用 Systems Manager 的 Amazon S3 儲存貯體和儲存貯體政策 180
使用 AWS Systems Manager 工具 189
節點工具 189
變更管理工具 191
應用程式工具 192
操作工具 192
使用 Systems Manager 執行節點任務 193
先決條件 193
使用預先安裝有 SSM Agent 的 AMI 啟動執行個體 193
使用 Systems Manager 連線至受管執行個體 195
清理您的執行個體 195
節點工具 195
合規 196
Distributor 212
Fleet Manager 260
混合啟用 337
庫存 339
Patch Manager 425
Run Command 697
Session Manager 757
State Manager 890
變更管理工具 977
自動化 977
Change Calendar 1420
Change Manager 1437
Documents 1511
Maintenance Windows 1633
Quick Setup 1754
應用程式工具 192
AWS AppConfig 1797
Application Manager 1797
Parameter Store 1839
操作工具 1981

Incident Manager 1982
Explorer 1982
OpsCenter 2018
CloudWatch 儀表板 2095
使用 SSM Agent 2097
了解 SSM Agent 的技術詳細資訊 2097
SSM Agent 3.2.x.x 版憑證行為 2098
SSM Agent 憑證優先順序 2098
設定 SSM Agent 以與聯邦資訊處理標準 (FIPS) 搭配使用 2100
關於本機 ssm 使用者帳戶 2101
SSM Agent 與 Instance Metadata Service (IMDS) 2102
讓 SSM Agent 保持最新狀態 2102
確定未修改、移動或刪除 SSM Agent 安裝目錄 2103
SSM Agent 滾動更新 AWS 區域 2103
SSM Agent 與 AWS 受管 S3 儲存貯體通訊 2103
尋找預先安裝了 SSM Agent的 AMIs 2110
驗證 SSM Agent 的狀態 2111
在 Linux 的 EC2 執行個體使用 SSM Agent 2115
驗證 SSM Agent 的簽章 2115
在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent 2129
將 SSM Agent 設定為使用 Linux 節點上的代理 2183
在 macOS 專用 EC2 執行個體使用 SSM Agent 2187
在 macOS EC2 執行個體上手動安裝和解除安裝 SSM Agent 2188
在 Windows Server 專用 EC2 執行個體使用 SSM Agent 2190
在 Windows Server EC2 執行個體上手動安裝和解除安裝 SSM Agent 2191
將 SSM Agent 設定為使用 Windows Server 執行個體的代理 2193
檢查 SSM Agent 狀態並啟動代理程式 2197
檢查 SSM Agent 版本編號 2199
檢視 SSM Agent 日誌 2204
允許 SSM Agent 偵錯記錄 2204
限制透過 SSM Agent 存取根層級命令 2207
自動化 SSM Agent 更新 2207
自動更新 SSM Agent 2209
訂閱 SSM Agent 通知 2209
SSM Agent 疑難排解 2210
SSM Agent 過期 2211

使用 SSM Agent 日誌檔案診斷並解決問題 2211
代理程式日誌檔案不會輪換 (Windows) 2212
無法連線至 SSM 端點 2212
驗證 VPC 組態 2213
驗證 VPC DNS 相關屬性 2214
驗證端點安全群組的相關輸入規則 2214
使用 ssm-cli 診斷並解決受管節點的可用性問題 2214
安全 2215
資料保護 2215
資料加密 2216
網際網路流量隱私權 2219
身分與存取管理 2219
目標對象 2219
使用身分驗證 2220
使用政策管理存取權 2222
AWS Systems Manager 搭配 IAM 的運作方式 2225
身分型政策範例 2235
AWS 受管政策 2244
疑難排解 2288
使用服務連結角色 2289
庫存和 Explorer 資料角色 2290
OpsCenter 和 Explorer 帳戶探索角色 2293
OpsData 和 OpsItems 建立角色 2295
操作洞察建立角色 2299
Quick Setup 部署運作狀態檢查角色 2302
匯出 OpsData 服務角色 2304
Just-in-time節點存取服務角色 2306
Just-in-time節點存取請求通知服務角色 2309
日誌記錄和監控 2313
法規遵循驗證 2315
恢復能力 2315
基礎設施安全性 2316
組態與漏洞分析 2316
安全最佳實務 2316
Systems Manager 預防性安全最佳實務 2317
Systems Manager 監控和稽核最佳實務 2320

程式碼範例 2322
基本概念 2329
嗨，Systems Manager 2333
了解基本概念 2337
動作 2388
日誌記錄和監控 2717
監控工具 2718
將節點日誌傳送至統一 CloudWatch Logs (CloudWatch 代理程式) 2718
將 Windows Server 節點日誌收集遷移到 CloudWatch 代理程式 2719
將 CloudWatch 代理程式組態設定存放在 Parameter Store 2729
回復為使用 SSM Agent收集日誌 2729
將 SSM Agent 日誌傳送至 CloudWatch Logs 2732
監控您的變更請求事件 2735
監控自動化 2737
Automation 指標 2738
使用 Amazon CloudWatch 監控 Run Command 指標 2739
Systems Manager Run Command 指標與維度 2739
使用 記錄 AWS Systems Manager API 呼叫 AWS CloudTrail 2740
CloudTrail 中的 Systems Manager 資料事件 2741
CloudTrail 中的 Systems Manager 管理事件 2743
Systems Manager 事件範例 2743
使用 CloudWatch Logs 記錄自動化動作輸出 2748
設定 Run Command 的 Amazon CloudWatch Logs 2752
傳送命令時指定 CloudWatch Logs 2753
在 CloudWatch Logs 中檢視命令輸出 2754
使用 Amazon EventBridge 進行監控 2754
為 Systems Manager 事件設定 EventBridge 2756
Systems Manager 的 Amazon EventBridge 事件範例 2759
範例方案：Amazon EventBridge 規則的 Systems Manager 目標 2774
使用 Amazon SNS 通知監控 Systems Manager 狀態變更 2776
設定 AWS Systems Manager的 Amazon SNS 通知 2776
AWS Systems Manager的 Amazon SNS 通知範例 2786
使用Run Command傳送命令以傳回狀態通知 2787
使用維護時段傳送命令以傳回狀態通知 2790
產品和服務整合 2795
與 整合 AWS 服務 2795

運算 2795
物聯網 (IoT) 2797
儲存 2798
開發人員工具 2799
安全性、身分與合規 2799
密碼編譯和 PKI 2802
管理與管控 2802
聯網與內容交付 2806
分析 2807
應用程式整合 2808
AWS Management Console 2808
從 Amazon Simple Storage Service (Amazon S3) 執行指令碼 2809
從Parameter Store參數參考 AWS Secrets Manager 秘密 2813
AWS KMS Parameter Store SecureString 參數的加密 2819
在 Amazon Elastic Kubernetes Service 中使用 AWS Secrets Manager 秘密 2830
在 AWS Lambda 函數中使用 Parameter Store 參數 2848
與其他產品及服務整合 2869
從 GitHub 執行指令碼 2872
將 Chef InSpec 設定檔與 Systems Manager 合規搭配使用 2879
與 ServiceNow 整合 2884
AWS Systems Manager 參考 2885
使用 AWS SDKs 2886
用於修補操作的 Amazon S3 儲存貯體 2887
包含用於修補操作之 SSM Command 文件的儲存貯體 (Linux 和 Windows Server) 2887
包含用於修補操作之 SSM Command 文件的儲存貯體 (macOS) 2890
包含 AWS 受管修補基準快照的儲存貯體 2893
Systems Manager 的 Amazon EventBridge 事件模式和類型 2896
事件類型：自動化 2897
事件類型：Change Calendar 2898
事件類型：Change Manager 2898
事件類型：組態合規 2898
事件類型：庫存 2899
事件類型：維護時段 2899
事件類型：OpsCenter 2901
事件類型：Parameter Store 2902
事件類型：Run Command 2903

事件類型：State Manager 2903
Cron 與 Rate 運算式 2904
有關 Cron 和 Rate 運算式的一般資訊 2904
關聯的 Cron 與 Rate 運算式 2909
維護時段的 Cron 與 Rate 運算式 2912
ec2messages、ssmmessages 和其他 API 操作 2914
代理程式相關的 API 操作 (ssmmessages 和 ec2messages 端點) 2914
ssm:* 命名空間執行個體相關的 API 操作 2916
ssm:* 命名空間其他 API 操作 2917
Systems Manager 的日期和時間字串格式 2918
為 Systems Manager 格式化日期和時間字串 2918
為 Systems Manager 建立自訂的日期和時間字串 2918
使用案例與最佳實務 2921
刪除 Systems Manager 資源和成品 2924
在 State Manager 與 Maintenance Windows 之間進行選擇 2927
State Manager 和 Maintenance Windows：關鍵使用案例 2927
相關資訊 2933
文件歷史紀錄 2935
2018 年 6 月前的更新 3075
文件慣用形式 3089
..... mmmxci

什麼是 AWS Systems Manager？

AWS Systems Manager 可協助您集中檢視、管理和大規模操作節點 AWS、內部部署和多雲端環境。隨著統一主控台體驗的推出，Systems Manager 會合併各種工具，協助您跨 AWS 帳戶 和 完成常見的節點任務 AWS 區域。

若要使用 Systems Manager，節點必須為受管節點，這表示機器上已安裝 SSM Agent，且代理程式可以與 Systems Manager 服務進行通訊。為了協助您識別節點未回報為受管的原因，Systems Manager 提供一鍵式代理程式問題診斷和修復執行手冊，讓您可以根據定義的排程設定此手冊來自動執行。此功能有助於識別節點無法連線至 Systems Manager 的原因，包括聯網組態錯誤。此外，此功能提供建議的執行手冊，可修復聯網問題和其他會防止將節點設為受管節點的問題。

統一主控台體驗也包含儀表板，可提供節點的高層級概觀。您可以深入了解更具體的節點洞察，例如哪些節點正在執行過時的作業系統 (OS) 軟體。您也可以根據OSs和作業系統版本、 AWS 帳戶和 SSM Agent版本等執行個體中繼資料 AWS 區域，使用篩選條件進行精細檢視。這些篩選條件可協助您擷取整個組織中特定帳戶層級或應用程式層級的相關資訊。

主題

- Systems Manager 如何讓我的營運獲得益處？
- 哪些人員應該使用 Systems Manager？
- Systems Manager 有哪些主要功能？
- 支援的 AWS 區域
- 存取 Systems Manager
- Systems Manager 服務名稱歷程記錄
- 支援的作業系統和機器類型
- 什麼是統一主控台？

Systems Manager 如何讓我的營運獲得益處？

Systems Manager 的優勢包含下方項目：

- 深入了解整個基礎設施

  Systems Manager 支援集中檢視整個組織帳戶和區域的節點。快速存取 ID、名稱、作業系統詳細資訊和已安裝的代理程式等執行個體資訊。使用 Amazon Q Developer 以自然語言查詢執行個體中繼資料，協助您識別問題並以更快的速度採取行動。

- 透過自動化提升營運效率

  將常見的操作任務自動化，減少維護系統所需的時間和工作。Systems Manager 可以大規模地對節點進行安全且可靠的遠端管理，無需登入您的伺服器。您不再需要使用堡壘主機、SSH 或遠端 PowerShell。此外，Systems Manager 可讓您輕鬆自動化節點群組的常見管理任務，如登錄檔編輯、使用者管理以及軟體和修補程式安裝。

- 在任何環境中大規模地簡化節點管理作業

  Systems Manager 可協助您管理跨 AWS、內部部署和多雲端環境的節點。排程自動診斷來識別 SSM Agent 問題，並使用一鍵式執行手冊修復這些問題。將節點設定為受管節點之後，您可以執行套用安全修補程式、啟動記錄的工作階段以及遠端執行命令等重要操作任務。

哪些人員應該使用 Systems Manager？

IT 營運經理和操作人員、DevOps 工程師、安全與合規經理、IT 主管以及 CIO 應該使用 Systems Manager。一般而言，Systems Manager 適用於下列對象：

- 想要大規模改善節點管理和安全性的組織。
- 想要在管理基礎設施時深入了解狀況並提高營運敏捷性的組織。
- 想要大規模增加營運效率的組織。

Systems Manager 有哪些主要功能？

Systems Manager 的主要功能是在統一主控台與 Systems Manager 提供的個別工具之間共用，以協助您大規模管理節點。

統一主控台

統一主控台提供集中式體驗，讓您檢視和管理節點。此主控台利用數個 Systems Manager 工具等，為您提供下列項目：

- 集中檢視節點
- 詳細的節點洞見
- 自動化診斷和修復常見節點問題

如需統一主控台的詳細資訊，請參閱 什麼是統一主控台？。

工具

工具包含 Systems Manager 的個別功能及其功能Session Manager，例如 Run Command、、自動化和 Parameter Store。使用 Systems Manager 工具，您可以執行下列動作：

- 大規模修補節點
- 安全地連接到節點，無需開啟傳入連接埠
- 在節點上遠端執行命令
- 安全地存放應用程式參考的資料
- 自動化常見的系統管理任務

如需 Systems Manager 工具的詳細資訊，請參閱 使用 AWS Systems Manager 工具。

支援的 AWS 區域

如需支援 Systems Manager 工具 AWS 區域 的 清單，請參閱《》中的 Systems Manager 服務端點Amazon Web Services 一般參考。

於 2024 年 11 月 21 日發行的統一 Systems Manager 主控台可在下列位置取得 AWS 區域：

- 美國東部 (維吉尼亞北部) 區域
- 美國東部 (俄亥俄) 區域
- 美國西部 (加利佛尼亞北部) 區域
- 美國西部 (奧勒岡) 區域
- 加拿大 (中部) 區域
- 南美洲 (聖保羅) 區域
- 亞太 (孟買) 區域
- 亞太區域 (東京) 區域
- 亞太區域 (首爾) 區域
- 亞太區域 (新加坡) 區域
- 亞太區域 (雪梨) 區域
- 歐洲 (法蘭克福) 區域
- 歐洲 (斯德哥爾摩) 區域
- 歐洲 (愛爾蘭) 區域

- 歐洲 (倫敦) 區域
- 歐洲 (巴黎) 區域

存取 Systems Manager

您可以透過以下方式使用 Systems Manager：

Systems Manager 主控台：

Systems Manager 主控台是一種瀏覽器界面，可存取及使用 Systems Manager。

AWS IoT Greengrass V2 主控台

您可以在 Greengrass 主控台 AWS IoT Greengrass 中檢視和管理為 設定的邊緣裝置。

AWS 命令列工具

透過使用 AWS 命令列工具，您可以在系統的命令列發出命令，以執行 Systems Manager 和其他 AWS 任務。Linux、macOS 和 Windows 支援這些工具。與使用主控台相較，使用 AWS Command Line Interface (AWS CLI) 更快速也更便利。若您想要建構執行 AWS 任務的指令碼，命令列工具也非常實用。

AWS 提供兩組命令列工具： AWS Command Line Interface和 AWS Tools for Windows PowerShell。如需安裝和使用 的詳細資訊 AWS CLI，請參閱AWS Command Line Interface 《使用者指南》。如需安裝和使用 Tools for Windows PowerShell 的詳細資訊，請參閱 AWS Tools for PowerShell 使用者指南。

> **Note**
>
> 在您的 Windows Server 執行個體上，需要 Windows PowerShell 3.0 或更新版本，才能執行特定 SSM 文件 (例如舊版 `AWS-ApplyPatchBaseline` 文件)。驗證您的 Windows Server 執行個體正在執行 Windows Management Framework 3.0 或更新版本。這個架構包含 Windows PowerShell。

AWS SDKs

AWS 提供軟體開發套件 (SDKs)，其中包含適用於各種程式設計語言和平台 （例如 Java、Python、Ruby、.NET、iOS 和 Android 等) 的程式庫和範本程式碼。SDK 提供便捷方法來授予對 Systems Manager 的程式設計方式存取。如需 AWS SDKs 的相關資訊，包括如何下載和安裝，請參閱適用於 Amazon Web Services 的工具。

Systems Manager 服務名稱歷程記錄

AWS Systems Manager (Systems Manager) 先前稱為「Amazon Simple Systems Manager (SSM)」和「Amazon EC2 Systems Manager (SSM)」。服務的原始縮寫名稱「SSM」仍會反映在各種 AWS 資源中，包括一些其他服務主控台。一些範例：

- Systems ManagerAgent：SSM Agent
- Systems Manager 參數：SSM 參數
- Systems Manager 服務端點：`ssm.region.amazonaws.com`
- AWS CloudFormation 資源類型： `AWS::SSM::Document`
- AWS Config 規則識別符： `EC2_INSTANCE_MANAGED_BY_SSM`
- AWS Command Line Interface (AWS CLI) 命令： `aws ssm describe-patch-baselines`
- AWS Identity and Access Management (IAM) 受管政策名稱： `AmazonSSMReadOnlyAccess`
- Systems Manager 資源 ARN：`arn:aws:ssm:region:account-id:patchbaseline/pb-07d8884178EXAMPLE`

支援的作業系統和機器類型

在使用 Systems Manager 之前，請確認您的作業系統 (OS)、作業系統版本和機器類型均符合受管節點的要求。

主題

- Systems Manager 支援的作業系統
- 混合多雲端環境中支援的機器類型

Systems Manager 支援的作業系統

以下各節列出 Systems Manager 支援的作業系統和作業系統版本。

> **Note**
>
> 如果您計劃使用 Systems Manager 管理和設定 AWS IoT Greengrass 核心裝置，這些裝置必須符合 的要求 AWS IoT Greengrass。如需詳細資訊，請參閱《 AWS IoT Greengrass Version 2 開發人員指南》中的設定 AWS IoT Greengrass 核心裝置。

如果您計劃管理和設定 AWS IoT 和非AWS 邊緣裝置，這些裝置必須符合此處列出的要求，並設定為 Systems Manager 的內部部署受管節點。如需詳細資訊，請參閱使用 Systems Manager 管理邊緣裝置。

Important

Patch ManagerSystems Manager 中的工具 可能不支援本主題中列出的所有作業系統版本。如需 Patch Manager 支援的作業系統版本的完整清單，請參閱 Patch Manager 先決條件。

作業系統類型

- Linux
- macOS (僅限 Amazon EC2 執行個體)
- Raspberry Pi OS (先前為 Raspbian)
- Windows Server

選取作業系統平台以查看支援的主要和次要版本。

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**
>
> 從 2015.03 版開始，Amazon Linux 1 已發行 x86_64 版。
> Amazon Linux 1 已於 2020 年 12 月 31 日終止標準支援，並於 2023 年 12 月 31 日終止生命週期，如AWS 新聞部落格上的 Amazon Linux 生命週期AMI結束更新 end-of-life中所宣布。 AWS 不再為此作業系統提供 Amazon Machine Images(AMIs)。不過， AWS Systems Manager 繼續為現有的 Amazon Linux 1 執行個體提供支援。

Amazon Linux 2

版本	x86	x86_64	ARM64
2.0 和所有更新版本		✓	✓

Amazon Linux 2023

版本	x86	x86_64	ARM64
2023.0.20230315.0 和所有更新版本		✓	✓

Bottlerocket

版本	x86_64	ARM64
1.0.0 和所有更新版本	✓	✓

CentOS

版本	x86	x86_64	ARM64
6.x[1]	✓	✓	
7.1 和更新的 7.x 版本		✓	✓
8.x		✓	✓

[1] 若要使用這些版本，必須使用 SSM Agent 的 3.0.x 版。我們建議使用 SSM Agent 的最新可用 3.0.x 版。後續不支援 SSM Agent 版本 (3.1 或更新版本)。

CentOS Stream

版本	x86	x86_64	ARM64
8		✓	✓
9		✓	✓

Debian Server

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] 若要使用這些版本，必須使用 SSM Agent 的 3.0.x 版。我們建議使用 SSM Agent 的最新可用 3.0.x 版。後續不支援 SSM Agent 版本 (3.1 或更新版本)。

Rocky Linux

版本	x86	x86_64	ARM64
8.x		✓	✓
9.x		✓	✓

SUSE Linux Enterprise Server (SLES)

版本	x86	x86_64	ARM64
12 和更新的 12.x 版本		✓	
15.x		✓	✓

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 STR		✓	✓
22.04 LTS		✓	✓
23.04、23.10		✓	✓
24.04 LTS		✓	✓
24.10		✓	✓
25.04		✓	✓

macOS (僅限 Amazon EC2 執行個體)

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**
>
> macOS 不支援全部 AWS 區域。如需有關 macOS Amazon EC2 支援的詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Amazon EC2 Mac instances。

Raspberry Pi OS (先前為 Raspbian)

版本	ARM32
8 (Jessie)	✓
9 (Stretch)	✓

詳細資訊

- 使用 AWS Systems Manager管理 Raspberry Pi 裝置

Windows Server

SSM Agent 需要 Windows PowerShell 3.0或更新版本，才能在Windows Server執行個體 AWS Systems Manager（例如舊版文件）上執行特定文件 (SSM `AWS-ApplyPatchBaseline` 文件）。驗證您的 Windows Server 執行個體正在執行 Windows Management Framework 3.0 或更新版本。這個架構包含 Windows PowerShell。如需詳細資訊，請參閱Windows Management Framework 3.0。

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

版本	x86	x86_64	ARM64
2025		✓	

[1] Windows Server 2008 和 2008 R2 支援：截至 2020 年 1 月 14 日，Microsoft 不再支援 Windows Server 2008 的功能或安全性更新。Windows Server 2008 和 2008 R2 的舊版 Amazon Machine Images (AMIs) 仍包含預先安裝的 SSM Agent 的版本 2，但 Systems Manager 不再正式支援 2008 版本，並且不再更新這些 Windows Server 版本的代理程式。除此之外，SSM Agent 第 3 版可能無法與 Windows Server 2008 和 2008 R2 上的所有操作相容。Windows Server 2008 版本的 SSM Agent 的最終的正式支援版本是 2.3.1644.0。

[2] Windows Server 2012 和 2012 R2 支援：Windows Server 2012 和 2012 R2 已於 2023 年 10 月 10 日終止支援。若要將 SSM Agent 與這些版本搭配使用，我們建議使用 Microsoft 的延伸安全性更新 (ESU)。如需詳細資訊，請參閱 Microsoft 網站上的 Windows Server 2012 和 2012 R2 即將終止支援。

混合多雲端環境中支援的機器類型

Systems Manager 支援多種機器類型做為受管節點。受管節點是指針對 Systems Manager 設定可與之搭配使用的任何機器。

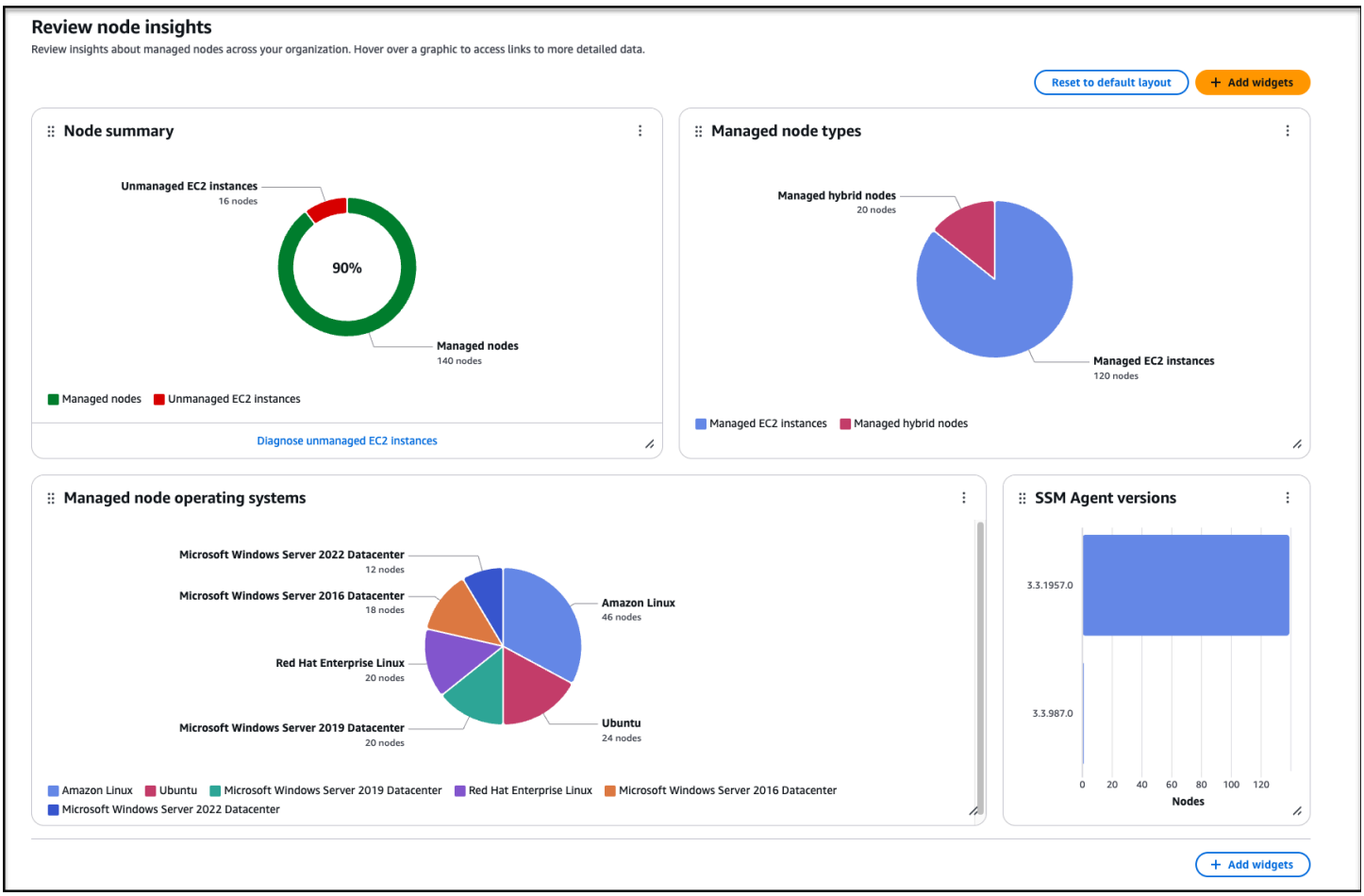
本使用者指南使用混合多雲端一詞來表示包含下列機器類型之任意組合的環境：

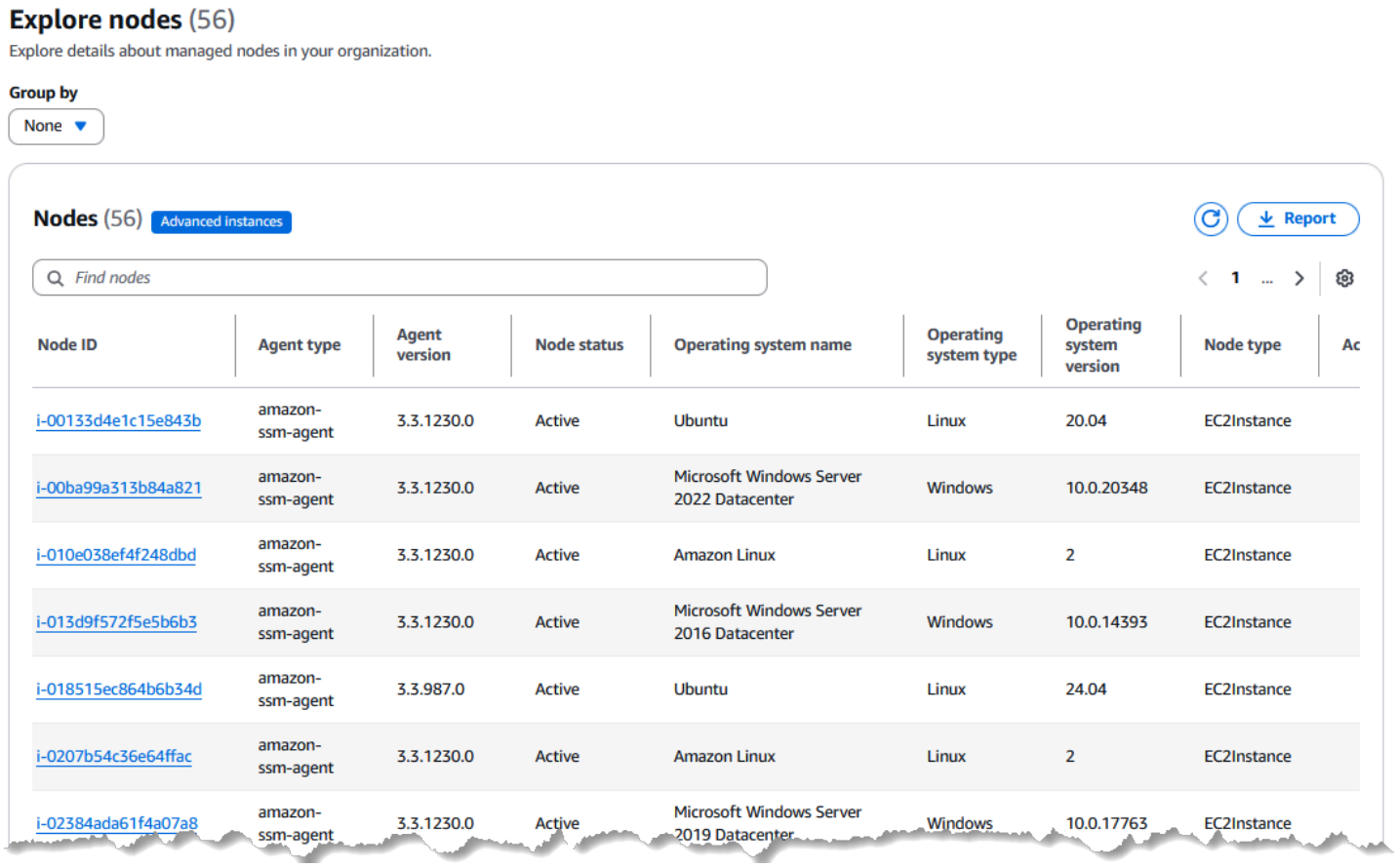
- Amazon Elastic Compute Cloud (Amazon EC2) 執行個體
- 您內部部署的伺服器 (內部部署伺服器)
- AWS IoT Greengrass 核心裝置
- AWS IoT 和非AWS 邊緣裝置
- 虛擬機器 (VM)，包含其他雲端環境中的 VM

如需混合多雲端環境 AWS 支援的相關資訊，請參閱AWS 混合多雲端解決方案。

什麼是統一主控台？

統一 Systems Manager 主控台是合併的體驗，結合各種工具，協助您完成 AWS 區域 AWS Organizations 組織中多個 AWS 帳戶 和 ，或單一帳戶和區域的常見節點任務。節點可以是 EC2 執行個體、混合伺服器，或在多雲端環境中執行的伺服器。在統一主控台中，您會獲得節點的詳細洞見。您可以為節點產生報告，診斷和修復常見問題，以防止節點報告由 Systems Manager 管理，例如連線問題。除了節點的摘要之外，您還可以檢視節點的特定詳細資訊，例如軟體庫存和修補狀態。


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



無論您在組織中的多個帳戶和區域中有節點，還是單一帳戶和區域中有節點，我們建議使用統一主控台。Systems Manager 會繼續將完成更多節點任務的功能新增至統一主控台。若要了解您現在可以使用統一主控台執行的節點任務，請參閱 使用 執行節點管理任務 AWS Systems Manager。

繼續進行下一個主題，以設定要由 Systems Manager 管理的節點。若要進一步了解如何設定 Systems Manager 的節點，請參閱 設定 AWS Systems Manager的受管節點。設定節點之後，您可以設定 Systems Manager 和統一主控台。若要進一步了解如何設定 Systems Manager，請參閱 設定 AWS Systems Manager。

設定 AWS Systems Manager的受管節點

完成本節中的任務，以設定角色、使用者帳戶、許可和初始資源，以使用 AWS Systems Manager 工具。本節所述的任務通常由 AWS 帳戶 和系統管理員執行。完成這些步驟之後，組織中的使用者即可使用 Systems Manager 設定、管理及存取受管節點。受管節點是設定為在混合多雲端環境中搭配 Systems Manager 使用的任何機器。

> **Note**
>
> 如果您計劃在混合多雲端環境中同時使用 Amazon EC2 執行個體和您自己的運算資源，請先依照 使用 Systems Manager 管理 EC2 執行個體 中的步驟進行操作。該主題以完成 EC2 執行個體和非 EC2 機器之 Systems Manager 設定的最佳順序呈現步驟。

如果您已使用其他 AWS 服務，則已完成其中一些步驟。不過，其他步驟則是 Systems Manager 特有的。因此，我們建議您檢閱整個區段，以確保您已準備好使用所有 Systems Manager 工具。

主題

- 使用 Systems Manager 管理 EC2 執行個體
- 使用 Systems Manager 在混合多雲端環境中管理節點
- 使用 Systems Manager 管理邊緣裝置
- 為 Systems Manager 建立 AWS Organizations 委派管理員
- 的一般設定 AWS Systems Manager

使用 Systems Manager 管理 EC2 執行個體

完成本節中的任務，以設定 的角色、許可和初始資源 AWS Systems Manager。本節所述的任務通常由 AWS 帳戶 和系統管理員執行。完成這些步驟之後，您組織中的使用者即可使用 Systems Manager 設定、管理和存取 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。

> **Note**
>
> 如果您計劃使用 Systems Manager 來管理及設定內部部署機器，請依照 使用 Systems Manager 在混合多雲端環境中管理節點 中的步驟設定。如果您計劃在混合多雲端環境中同時使

用 Amazon EC2 執行個體和非 EC2 機器，請先依照這裡的步驟進行操作。本節提供建議順序的步驟，來設定角色、使用者、許可和初始資源以用於您的 Systems Manager 操作。

如果您已使用其他 AWS 服務，則已完成其中一些步驟。不過，其他步驟則是 Systems Manager 特有的。因此，我們建議您檢閱整個區段，以確保您已準備好使用所有 Systems Manager 工具。

目錄

- 設定 Systems Manager 所需的執行個體許可
- 使用適用於 Systems Manager 的 VPC 端點來改善 EC2 執行個體的安全性

設定 Systems Manager 所需的執行個體許可

根據預設， AWS Systems Manager 沒有在您的執行個體上執行動作的許可。您可以使用 AWS Identity and Access Management (IAM) 角色在帳戶層級提供執行個體許可，或使用執行個體描述檔在執行個體層級提供執行個體許可。如果您的使用案例允許，建議您使用預設主機管理組態在帳戶層級授予存取權。

> **Note**
>
> 您可以略過此步驟，允許 Systems Manager 在設定統一主控台時將必要的許可套用至執行個體。如需詳細資訊，請參閱設定 AWS Systems Manager。

EC2 執行個體許可的建議組態

預設主機管理組態允許 Systems Manager 自動管理 Amazon EC2 執行個體。開啟此設定後，使用中執行個體中繼資料服務第 2 版 (IMDSv2) 的所有執行個體， AWS 區域 以及安裝SSM Agent版本 3.2.582.0 或更新版本 AWS 帳戶 的執行個體，都會自動成為受管執行個體。預設主機管理組態不支援執行個體中繼資料服務第 1 版。如需有關轉換至 IMDSv2 的資訊，請參閱《Amazon EC2 使用者指南》中的 Transition to using Instance Metadata Service Version 2。如需有關檢查執行個體上已安裝之 SSM Agent 版本的詳細資訊，請參閱檢查 SSM Agent 版本編號。如需有關更新 SSM Agent 的資訊，請參閱自動更新 SSM Agent。受管執行個體的優點包含：

- 使用 Session Manager 安全地連線至執行個體。
- 使用 Patch Manager 執行自動修補程式掃描。
- 使用 Systems Manager 庫存檢視執行個體的詳細資訊。

- 使用 Fleet Manager 追蹤和管理執行個體。
- 自動將 SSM Agent 保持在最新狀態。

Fleet Manager、庫存Patch Manager、 和 Session Manager是 中的工具 AWS Systems Manager。

預設主機管理組態可無需使用執行個體設定檔就實現執行個體管理，並確保 Systems Manager 擁有管理區域和帳戶中所有執行個體的許可。如果提供的許可不足以滿足使用案例，您也可以將政策新增至預設主機管理組態建立的預設 IAM 角色。如果您不需要預設 IAM 角色提供的所有功能的許可，則可以建立自己的自訂角色和政策。對為預設主機管理組態選擇的 IAM 角色所做的任何變更都會套用到該區域和帳戶中的所有受管 Amazon EC2 執行個體。如需有關預設主機管理組態所使用之政策的詳細資訊，請參閱 AWS 管理的政策：AmazonSSMManagedEC2InstanceDefaultPolicy。如需有關預設主機管理組態的詳細資訊，請參閱使用預設主機管理組態來自動管理 EC2 執行個體。

> **Important**
>
> 使用預設主機管理組態註冊的執行個體，會將註冊資訊儲存在本機 `/lib/amazon/ssm` 或 `C:\ProgramData\Amazon` 目錄中。移除這些目錄或其檔案可防止執行個體取得使用預設主機管理組態連線至 Systems Manager 的必要憑證。在這些情況下，您必須使用執行個體設定檔來提供執行個體所需的許可，或重新建立執行個體。

> **Note**
>
> 此程序僅供管理員執行。允許個人設定或修改預設主機管理組態時，實作最低權限存取。您必須在 AWS 區域 想要自動管理 Amazon EC2 執行個體的每個 中開啟預設主機管理組態。

開啟預設主機管理組態設定

您可以從 Fleet Manager 主控台開啟預設主機管理組態。若要使用 AWS Management Console 或偏好的命令列工具順利完成此程序，您必須擁有 GetServiceSetting、ResetServiceSetting 和 UpdateServiceSetting API 操作的許可。此外，您必須擁有 `AWSSystemsManagerDefaultEC2InstanceManagementRole` IAM 角色 `iam:PassRole` 許可的許可。政策範例如下。將每個`########`取代為您自己的資訊。

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

```
 "Statement": [
  {
   "Effect": "Allow",
   "Action": [
    "ssm:GetServiceSetting",
    "ssm:ResetServiceSetting",
    "ssm:UpdateServiceSetting"
   ],
   "Resource": "arn:aws:ssm:region:account-id:servicesetting/ssm/managed-instance/
default-ec2-instance-management-role"
  },
  {
   "Effect": "Allow",
   "Action": [
    "iam:PassRole"
   ],
   "Resource": "arn:aws:iam::account-id:role/service-role/
AWSSystemsManagerDefaultEC2InstanceManagementRole",
   "Condition": {
    "StringEquals": {
     "iam:PassedToService": [
      "ssm.amazonaws.com"
     ]
    }
   }
  }
 ]
}
```

在開始之前，如果執行個體設定檔已連接至 Amazon EC2 執行個體，請移除允許該 `ssm:UpdateInstanceInformation` 操作的所有許可。在使用預設主機管理組態許可之前，SSM Agent 會嘗試使用執行個體設定檔許可。如果您允許執行個體設定檔中的 `ssm:UpdateInstanceInformation` 操作，執行個體將不會使用預設主機管理組態許可。

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 在帳戶管理下拉式清單中，選擇設定預設主機管理組態。
4. 開啟開啟預設主機管理組態。

5. 選擇用於為您的執行個體啟用 Systems Manager 工具的 IAM 角色。建議使用預設主機管理組態提供的預設角色。其中包括使用 Systems Manager 管理 Amazon EC2 執行個體所需的最小許可集。如果您偏好使用自訂角色，角色的信任政策必須允許 Systems Manager 作為受信任實體。
6. 選擇設定以完成設定。

開啟預設主機管理組態之後，執行個體可能需要 30 分鐘的時間才能使用您所選角色的憑證。您必須在想要實現 Amazon EC2 執行個體自動管理的每個區域中開啟預設主機管理組態。

EC2 執行個體許可的替代組態

您可以使用 AWS Identity and Access Management (IAM) 執行個體設定檔，在個別執行個體層級授予存取權。執行個體設定檔是在啟動時將 IAM 角色資訊傳遞到 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體的容器。透過將定義所需許可的一個或多個 IAM 政策連接到新角色或您已建立的角色，可建立 Systems Manager 的執行個體設定檔。

> **Note**
>
> 您可以使用 中的Quick Setup工具 AWS Systems Manager，快速在 中的所有執行個體上設定執行個體描述檔 AWS 帳戶。 Quick Setup也會建立 IAM 服務角色 （或擔任角色），讓 Systems Manager 可以代表您安全地為您的執行個體執行命令。透過使用 Quick Setup，您可以跳過此步驟 (步驟 3) 和步驟 4。如需詳細資訊，請參閱AWS Systems Manager Quick Setup。

請記下建立 IAM 執行個體設定檔的下列詳細資訊：

- 如果您正在 Systems Manager 混合多雲端環境中設定非 EC2 機器，您不需要為它們建立執行個體設定檔。反之，將伺服器和虛擬機器設定為使用 IAM 服務角色。如需詳細資訊，請參閱在混合多雲端環境中建立 Systems Manager 所需的 IAM 服務角色。
- 如果您變更 IAM 執行個體設定檔，執行個體憑證重新整理可能需要花費一些時間。SSM Agent 在此狀況發生前都不會處理請求。若要加速重新整理程序，您可以重新啟動 SSM Agent 或重新啟動執行個體。

根據您是否將為執行個體設定檔建立新角色或將所需的許可新增到現有角色，來使用下列其中一個程序。

為 Systems Manager 受管執行個體建立執行個體設定檔 (主控台)

1. 在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。
2. 在導覽窗格中，選擇角色，然後選擇建立角色。
3. 對於 Trusted entity type (信任的實體類型)，請選擇 AWS 服務。
4. 在 Use case (使用案例) 下，隨即選擇 EC2，然後選擇 Next (下一步)。
5. 在 Add permissions (新增許可) 頁面上，執行以下作業：

   - 使用 Search (搜尋) 欄位，找出 AmazonSSMManagedInstanceCore 政策。選取其名稱旁的核取方塊，如下圖所示。

     即使您搜尋其他政策，主控台仍會保留您的選取項目。

   - 如果您在先前程序中已建立自訂 S3 儲存貯體政策，(選用) 建立 S3 儲存貯體存取的自訂政策，請搜尋它並選取名稱旁的核取方塊。
   - 如果您計劃將執行個體加入由 管理的 Active Directory AWS Directory Service，請搜尋 AmazonSSMDirectoryServiceAccess，然後選取其名稱旁的核取方塊。
   - 若您計劃使用 Eventbridge 或 CloudWatch Logs 來管理或監控您的執行個體，請搜尋 CloudWatchAgentServerPolicy，然後選取其名稱旁邊的核取方塊。
6. 選擇 Next (下一步)。
7. 在 Role name (角色名稱) 中，輸入新執行個體設定檔的名稱，如 **`SSMInstanceProfile`**。

   > **Note**
   >
   > 請記下角色名稱。當您建立您想要使用 Systems Manager 管理的新執行個體時，將選擇此角色。

8. (選用) 在 Description (說明)，更新此執行個體設定檔的說明。
9. (選用) 對於 Tags (標籤)，新增一個或多個標籤鍵值組來整理、追蹤或控制存取此角色的存取權，然後選擇 Create role (建立角色)。系統會讓您回到 Roles (角色) 頁面。

若要將 Systems Manager 的執行個體設定檔許可新增到現有角色 (主控台)

1. 開啟位於 https://console.aws.amazon.com/iam/ 的 IAM 主控台。
2. 在導覽窗格中，選擇 Roles (角色)，然後選擇您想要針對 Systems Manager 操作與執行個體設定檔建立關聯的現有角色。
3. 在 Permissions 索引標籤上，依序選擇 Add permissions, Attach policies (新增許可、連接政策)。
4. 在 Attach policy (連接政策) 頁面上，執行下列動作：
   - 使用 Search (搜尋) 欄位，找出 AmazonSSMManagedInstanceCore 政策。選取其名稱旁的核取方塊。
   - 如果您已建立自訂的 S3 儲存貯體政策，請加以搜尋並選取名稱旁的核取方塊。如需為執行個體設定檔自訂 S3 儲存貯體政策的相關資訊，請參閱 (選用) 建立 S3 儲存貯體存取的自訂政策。
   - 如果您計劃將執行個體加入由 管理的 Active Directory AWS Directory Service，請搜尋 AmazonSSMDirectoryServiceAccess，然後選取其名稱旁的核取方塊。
   - 若您計劃使用 Eventbridge 或 CloudWatch Logs 來管理或監控您的執行個體，請搜尋 CloudWatchAgentServerPolicy，然後選取其名稱旁邊的核取方塊。
5. 選擇連接政策。

如需有關如何更新角色以包含信任實體或進一步限制存取的詳細資訊，請參閱 IAM 使用者指南中的修改角色。

(選用) 建立 S3 儲存貯體存取的自訂政策

唯有在 Systems Manager 操作中使用 VPC 端點或自有 S3 儲存貯體時，才需要為 Amazon Simple Storage Service (Amazon S3) 存取建立自訂政策。您可以將此政策連接至由預設主機管理組態所建立的預設 IAM 角色，或是您在上一個程序中建立的執行個體設定檔。

如需您在下列政策中提供存取權的 AWS 受管 S3 儲存貯體相關資訊，請參閱 SSM Agent 與 AWS 受管 S3 儲存貯體通訊。

1. 在 https://console.aws.amazon.com/iam/ 中開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Policies (政策)，然後選擇 Create policy (建立政策)。
3. 選擇 JSON 標籤，並預設文字取代為下列內容：

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

```
{ (1)
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": [
                "arn:aws:s3:::aws-ssm-region/*",
                "arn:aws:s3:::aws-windows-downloads-region/*",
                "arn:aws:s3:::amazon-ssm-region/*",
                "arn:aws:s3:::amazon-ssm-packages-region/*",
                "arn:aws:s3:::region-birdwatcher-prod/*",
                "arn:aws:s3:::aws-ssm-distributor-file-region/*",
                "arn:aws:s3:::aws-ssm-document-attachments-region/*",
                "arn:aws:s3:::patch-baseline-snapshot-region/*"
            ]
        },

{ (2)
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject",

 "s3:PutObjectAcl", (3)

 "s3:GetEncryptionConfiguration" (4)
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/*",
                "arn:aws:s3:::amzn-s3-demo-
bucket" (5)
            ]
        }
    ]
}
```

[1] 唯有在您使用 VPC 端點時，才需要第一個 Statement 元素。

[2] 唯有在 Systems Manager 操作中使用您建立的 S3 儲存貯體時，才需要第二個 Statement 元素。

[3] 唯有在您計劃在其他帳戶中支援跨帳戶存取 S3 儲存貯體時，才需要 PutObjectAcl 存取控制清單許可。

[4] 如果您的 S3 儲存貯體設定為使用加密，則需要 GetEncryptionConfiguration 元素。

[5] 如果您的 S3 儲存貯體設定為使用加密，則 S3 儲存貯體根 (例如 arn:aws:s3:::amzn-s3-demo-bucket) 必須列在 Resource (資源) 區段中。您的使用者、群組或角色必須設定為可存取根儲存貯體。

4. 如果您在操作中使用 VPC 端點，請執行下列動作：

   在第一個 Statement 元素中，將每個*##*預留位置取代為將使用此政策的 AWS 區域 的識別符。例如，對於美國東部 (俄亥俄) 區域，使用 us-east-2。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

   > **Important**
   >
   > 我們建議您避免在這個政策中的特定區域使用萬用字元 (*)。例如，使用 arn:aws:s3:::aws-ssm-us-east-2/* 而不使用 arn:aws:s3:::aws-ssm-*/*。使用萬用字元可能允許存取您不想授與存取權的 S3 儲存貯體。如果您要將執行個體設定檔用於多個區域，建議您為每個區域重複第一個 Statement 元素。

   -或-

   如果您在操作中不是使用 VPC 端點，您可以刪除第一個 Statement 元素。

5. 如果您在 Systems Manager 操作中使用自有的 S3 儲存貯體，請執行下列動作：

   在第二個 Statement 元素中，將 *amzn-s3-demo-bucket* 取代為帳戶中的 S3 儲存貯體名稱。您將使用此儲存貯體來進行 Systems Manager 操作。它將使用 "arn:aws:s3:::my-bucket-name/*" 作為資源，為儲存貯體中的物件提供許可。如需有關對儲存貯體或儲存貯體中的物件提供許可的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的 Amazon Simple Storage Service (Amazon S3) 動作主題與 AWS 部落格文章 IAM 政策和儲存貯體政策以及 ACL！天啊！(控制 S3 資源的存取權)。

> **Note**
>
> 如果您使用多個儲存貯體，請為每個儲存貯體提供 ARN。請參閱下列範例，了解儲存貯體的許可。
>
> ```
> "Resource": [
> "arn:aws:s3:::amzn-s3-demo-bucket1/*",
> "arn:aws:s3:::amzn-s3-demo-bucket2/*"
>               ]
> ```

-或-

如果您在 Systems Manager 操作中不是使用自有的 S3 儲存貯體，您可以刪除第二個 `Statement` 元素。

6. 選擇下一步：標籤。
7. (選用) 透過選擇 Add tag (新增標籤)，然後輸入政策的首選標籤來新增標籤。
8. 選擇下一步：檢閱。
9. 對於 Name (名稱)，輸入識別此政策的名稱，例如 **`SSMInstanceProfileS3Policy`**。
10. 選擇 建立政策。

受管執行個體的其他政策考量

本節介紹一些政策，您可以將這些政策新增至由預設主機管理組態所建立的預設 IAM 角色，或新增至 AWS Systems Manager的執行個體設定檔。若要提供執行個體和 Systems Manager API 間通訊的許可，建議考慮您系統需求和安全需求的自訂政策。視您的操作計劃而定，您可能需要其他一或多個政策所呈現的許可。

政策：**`AmazonSSMDirectoryServiceAccess`**

唯有在您計劃將 Windows Server 的 Amazon EC2 執行個體加入 Microsoft AD 目錄時，才需要這項政策。

此 AWS 受管政策允許 AWS Directory Service 代表您SSM Agent存取 ，以請求由受管執行個體加入網域。如需詳細資訊，請參閱《AWS Directory Service 管理指南》中的無縫加入 Windows EC2 執行個體。

政策：**CloudWatchAgentServerPolicy**

唯有在您計劃在執行個體上安裝並執行 CloudWatch 代理程式，以讀取執行個體上的指標和日誌資料，並將其寫入 Amazon CloudWatch 時，才需要這項政策。這些可協助您監控、分析和快速回應 AWS 資源的問題或變更。

唯有在您將使用諸如 Amazon EventBridge 或 Amazon CloudWatch Logs 等功能時，由預設主機管理組態建立的預設 IAM 角色或執行個體設定檔才需要此政策。(您也可以建立更嚴格的政策，例如，限制特定 CloudWatch Logs 日誌串流的寫入存取)。

> **Note**
>
> 使用 EventBridge 和 CloudWatch Logs 功能為選用。然後，如果您已決定使用，則建議在 Systems Manager 組態程序開始時進行設定。如需詳細資訊，請參閱《Amazon EventBridge 使用者指南》和《Amazon CloudWatch Logs 使用者指南》。

若要建立具有其他 Systems Manager 工具許可的 IAM 政策，請參閱下列資源：

- 使用 IAM 政策限制對 Parameter Store 參數的存取
- 設定自動化
- 步驟 2：為 Session Manager 確認或新增執行個體許可

將 Systems Manager 執行個體設定檔連接至執行個體 (主控台)

下列程序說明如何使用 Amazon EC2 主控台，將 IAM 執行個體設定檔連接至 Amazon EC2 執行個體。

1. 登入 AWS Management Console ，並在 https://console.aws.amazon.com/ec2/ ：// 開啟 Amazon EC2 主控台。
2. 在導覽窗格的 Instances (執行個體) 下方，選擇 Instances (執行個體)。
3. 導覽並從清單選擇您的 EC2 執行個體。
4. 在 Actions (動作) 選單中，選擇 Security (安全性)，然後選擇 Modify IAM role (修改 IAM 角色)。
5. 針對 IAM role (IAM 角色)，選取您使用EC2 執行個體許可的替代組態中程序所建立的執行個體設定檔。
6. 選擇 Update IAM role (更新 IAM 角色)。

如需有關將 IAM 角色連接至執行個體的詳細資訊，請根據您所選的作業系統類型，選擇下列其中一項：

- 《Amazon EC2 使用者指南》中的 Attach an IAM role to an instance
- 《Amazon EC2 使用者指南》中的 Attach an IAM role to an instance

繼續進行使用適用於 Systems Manager 的 VPC 端點來改善 EC2 執行個體的安全性。

使用適用於 Systems Manager 的 VPC 端點來改善 EC2 執行個體的安全性

您可以設定 AWS Systems Manager 以使用 Amazon Virtual Private Cloud (Amazon VPC) 中的介面 VPC 端點，藉此改善受管節點（包括混合多雲端環境中non-EC2 機器）的安全性狀態。透過使用介面 VPC 端點（介面端點），您可以連線至由 提供支援的服務 AWS PrivateLink。 AWS PrivateLink 是一種技術，可讓您使用私有 IP 地址私下存取 Amazon Elastic Compute Cloud (Amazon EC2) 和 Systems Manager APIs。

AWS PrivateLink 會將受管執行個體、Systems Manager 和 Amazon EC2 之間的所有網路流量限制在 Amazon 網路。這意味著受管執行個體無法存取網際網路。如果您使用 AWS PrivateLink，則不需要網際網路閘道、NAT 裝置或虛擬私有閘道。

您不需要設定 AWS PrivateLink，但建議使用。如需 AWS PrivateLink 和 VPC 端點的詳細資訊，請參閱 AWS PrivateLink 和 VPC 端點。

> **Note**
>
> 使用 VPC 端點的替代方案是在您的受管執行個體上啟用對外網際網路存取。在此情況下，受管執行個體也必須允許 HTTPS (連接埠 443) 傳出流量至下列端點：
>
> - `ssm.region.amazonaws.com`
> - `ssmmessages.region.amazonaws.com`
> - `ec2messages.region.amazonaws.com`
>
> SSM Agent 會啟動到雲端中 Systems Manager 服務的所有連線。因此，您不需要將防火牆設定為允許 Systems Manager 將流量傳入到您的執行個體。
> 如需呼叫這些端點的詳細資訊，請參閱參考：ec2messages、ssmmessages 和其他 API 操作。

Amazon VPC 簡介

您可以使用 Amazon Virtual Private Cloud (Amazon VPC)，在 內您自己的邏輯隔離區域中定義虛擬網路 AWS 雲端，稱為虛擬私有雲端 (VPC)。您可以在您的 VPC 中啟動 AWS 資源 (例如執行個體)。VPC 近似於您在自有資料中心內運作的傳統網路，卻能提供 AWS可擴展性基礎設施的效益。您可以設定您的 VPC；您可以選取其 IP 地址範圍、建立子網，以及設定路由表、網路閘道與安全設定。您可以將 VPC 中的執行個體連線至網際網路。您可以將 VPC 連接到自己的公司資料中心，讓資料中心 AWS 雲端 成為延伸。為了保護各個子網路的資源，您可使用多個安全性層級，包括安全群組及網路存取控制清單。如需詳細資訊，請參閱 Amazon VPC 使用者指南。

主題

- VPC 端點的限制與局限
- 若要建立 Systems Manager 的 VPC 端點
- 建立介面 VPC 端點政策

VPC 端點的限制與局限

在您設定 Systems Manager 的 VPC 端點之前，請注意以下的約束與限制。

VPC 對等連線

您可以透過區域內和區域間 VPC 對等連線來存取 VPC 界面端點。如需 VPC 介面端點 VPC 對等互連連線請求的詳細資訊，請參閱《Amazon Virtual Private Cloud 使用者指南》中的 VPC 對等互連連線 (配額)。

VPC 閘道端點連線不能延伸出 VPC。VPC 中 VPC 對等互連另一側的資源無法使用閘道端點與閘道端點服務中的資源通訊。如需 VPC 閘道端點 VPC 對等互連連線請求的詳細資訊，請參閱《Amazon Virtual Private Cloud 使用者指南》中的 VPC 端點 (配額)。

傳入連線

連接到 VPC 端點的安全群組必須允許從受管執行個體的私有子網路透過 443 通訊埠傳入的連線。如果不允許傳入的連線，則受管執行個體無法連線到 SSM 和 EC2 端點。

DNS 解析

如果您使用自訂 DNS 伺服器，則必須為 VPC 的 Amazon DNS 伺服器的 `amazonaws.com` 網域查詢新增條件式轉寄站。

S3 儲存貯體

VPC 端點政策必須允許對至少 SSM Agent 與 AWS 受管 S3 儲存貯體通訊 中所列 Amazon S3 儲存貯體的存取權。

> **Note**
>
> 如果您使用內部部署防火牆並計劃使用 Patch Manager，則該防火牆也必須允許存取適當的修補基準端點。

Amazon CloudWatch Logs

如果您不允許執行個體存取網際網路，請為 CloudWatch Logs 建立 VPC 端點，以使用將日誌傳送至 CloudWatch Logs 的功能。如需有關建立 CloudWatch Logs 端點的詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的建立 CloudWatch Logs 的 VPC 端點。

混合多雲端環境中的 DNS

如需有關設定 DNS 以在混合和多雲端環境中使用 AWS PrivateLink 端點的資訊，請參閱《Amazon VPC 使用者指南》中的介面端點的私有 DNS。如果您想要使用自己的 DNS，可以使用 Route 53 Resolver。如需詳細資訊，請參閱《Amazon Route 53 開發人員指南》中的在 VPC 和網路之間解析 DNS 查詢。

若要建立 Systems Manager 的 VPC 端點

使用下列資訊建立 VPC 介面端點 AWS Systems Manager。本主題連結至《Amazon VPC 使用者指南》中的程序。

> **Note**
>
> *region* 代表 支援的 AWS 區域 識別符 AWS Systems Manager，例如us-east-2美國東部（俄亥俄）區域。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

請遵循建立介面端點中的步驟來建立下列介面端點：

- **com.amazonaws.*region*.ssm** – Systems Manager 服務的端點。

- **com.amazonaws.*region*.ec2messages** – Systems Manager 使用此端點，從 SSM Agent 到 Systems Manager 服務進行呼叫。從 SSM Agent 3.3.40.0 版開始，Systems Manager 會在可用時開始使用 `ssmmessages:*` 端點 (Amazon Message Gateway Service)，而不是 `ec2messages:*` 端點 (Amazon Message Delivery Service)。
- **com.amazonaws.*region*.ec2** – 如果您使用 Systems Manager 來建立具備 VSS 功能的快照，則必須確定您擁有至 EC2 服務的端點。如果沒有定義 EC2 端點，則用來列舉所連接 Amazon EBS 磁碟區的呼叫會失敗，進而導致 Amazon Systems Manager 命令失敗。
- **com.amazonaws.*region*.s3**：Systems Manager 會使用此端點更新 SSM Agent。如果您可以選擇擷取在儲存貯體中存放的指令碼或其他檔案，或將輸出日誌上傳到儲存貯體，則 Systems Manager 也會使用此端點。如果與執行個體關聯的安全群組限制傳出流量，您必須新增規則，以允許 Amazon Simple Storage Service (Amazon S3) 的字首清單的流量。如需詳細資訊，請參閱 AWS PrivateLink 指南中的修改安全群組。
- **com.amazonaws.*region*.ssmmessages**：SSM Agent 要與 Systems Manager 服務通訊、進行Run Command，以及使用 Session Manager 透過安全資料管道連線至執行個體時，都需要此端點。如需詳細資訊，請參閱 AWS Systems Manager Session Manager 和 參考：ec2messages、ssmmessages 和其他 API 操作。
- （選用）**com.amazonaws.*region*.kms** – 如果您想要對 Session Manager或 Parameter Store 參數使用 AWS Key Management Service (AWS KMS) 加密，請建立此端點。
- (選用) **com.amazonaws.*region*.logs**：如果想使用適用於 Session Manager、Run Command 或 SSM Agent 日誌的 Amazon CloudWatch Logs (CloudWatch Logs)，則請建立此端點。

如需SSM Agent必須能夠存取的 AWS 受管 S3 儲存貯體相關資訊，請參閱 SSM Agent 與 AWS 受管 S3 儲存貯體通訊。如果您在 Systems Manager 操作中使用虛擬私有雲端 (VPC) 端點，則必須在適用於 Systems Manager 的 EC2 執行個體設定檔中提供明確許可，或在混合多雲端環境中的非 EC2 受管節點服務角色中提供明確許可。

建立介面 VPC 端點政策

您可以為 VPC 介面端點建立政策，您可以在 AWS Systems Manager 其中指定：

- 可執行動作的委託人
- 可執行的動作
- 可對其執行動作的資源

如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的使用 VPC 端點控制對服務的存取。

使用 Systems Manager 在混合多雲端環境中管理節點

您可以使用 AWS Systems Manager 來管理 Amazon Elastic Compute Cloud (EC2) 執行個體和多種 non-EC2 機器類型。本節說明帳戶和系統管理員會執行的設定任務，以使用 Systems Manager 管理混合多雲端環境中的非 EC2 機器。完成這些步驟後，管理員授予許可的使用者 AWS 帳戶 可以使用 Systems Manager 來設定和管理其組織的non-EC2 機器。

任何已設定為搭配 Systems Manager 使用的機器都稱為受管節點。

> **Note**
>
> - 您可以使用用於其他非 EC2 機器的混合啟用步驟，將邊緣裝置登錄為受管節點。這些類型的邊緣裝置包括 AWS IoT 裝置以外的裝置和 AWS IoT 裝置。使用本節所述程序來設定這些類型的邊緣裝置。
>
>   Systems Manager 也支援使用 AWS IoT Greengrass Core 軟體的邊緣裝置。 AWS IoT Greengrass 核心裝置的設定程序和需求與邊緣裝置以外的 AWS IoT 和 AWS 邊緣裝置不同。如需註冊 AWS IoT Greengrass 裝置以與 Systems Manager 搭配使用的相關資訊，請參閱 使用 Systems Manager 管理邊緣裝置。
> - Systems Manager 混合多雲端環境不支援非 EC2 macOS 機器。

如果您計劃使用 Systems Manager 來管理 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體，或在混合多雲端環境中同時使用 Amazon EC2 執行個體和非 EC2 機器，請先遵照 使用 Systems Manager 管理 EC2 執行個體 中的步驟操作。

設定 Systems Manager 的混合多雲端環境後，您可執行以下操作：

- 建立一致且安全的方式，使用相同的工具或指令碼，從相同的位置遠端管理您的混合多雲端環境中的工作負載。
- 使用 AWS Identity and Access Management (IAM) 集中控制可在機器上執行的動作的存取控制。
- 透過檢視 AWS CloudTrail中記錄的 API 活動，集中稽核在機器中執行的操作。

  如需使用 CloudTrail 監控 Systems Manager 動作的相關資訊，請參閱使用 記錄 AWS Systems Manager API 呼叫 AWS CloudTrail。
- 透過設定 Amazon EventBridge 和 Amazon Simple Notification Service (Amazon SNS) 以傳送有關服務執行成功的通知，進而集中管理監控。

如需使用 EventBridge 監控 Systems Manager 事件的相關資訊，請參閱使用 Amazon EventBridge 監控 Systems Manager。

關於受管節點

完成如本節所述為 Systems Manager 設定non-EC2 機器後，混合啟用的機器會列在 中 AWS Management Console ，並描述為受管節點。在主控台中，混合模式受管節點的 ID 字首為 "mi-"，有別於 Amazon EC2 執行個體。Amazon EC2 執行個體 ID 使用字首「i-」。

受管理節點是指針對 Systems Manager 設定的任何機器。以前，受管節點都稱為受管執行個體。術語執行個體現在僅指 EC2 執行個體。在此術語變更之前，命令命名為 deregister-managed-instance。

如需詳細資訊，請參閱使用受管節點。

關於執行個體方案

Systems Manager 為混合多雲端環境中的非 EC2 受管節點提供 standard-instances 方案和 advanced-instances 方案。standard-instances 方案可讓您在每個 AWS 區域的每個 AWS 帳戶 中最多登錄 1,000 部啟用混合模式的機器。如果您需要在單一帳戶和區域中登錄 1,000 部以上的非 EC2 機器，則請使用 advanced-instances 方案。進階執行個體也可讓您使用 連線到non-EC2 機器 AWS Systems Manager Session Manager。 Session Manager提供受管節點的互動式 Shell 存取。

如需詳細資訊，請參閱設定執行個體方案。

主題

- 在混合多雲端環境中建立 Systems Manager 所需的 IAM 服務角色
- 建立混合啟用，以向 Systems Manager 註冊節點
- 在混合 Linux 節點上安裝 SSM Agent
- 在混合 Windows Server 節點上安裝 SSM Agent

在混合多雲端環境中建立 Systems Manager 所需的 IAM 服務角色

混合多雲端環境中Non-EC2 (Amazon Elastic Compute Cloud) 機器需要 AWS Identity and Access Management (IAM) 服務角色才能與服務通訊 AWS Systems Manager 。該角色將 AWS Security Token Service (AWS STS) `AssumeRole` 信任授予給 Systems Manager 服務。您只需要為每個 AWS 帳戶建立一次適用於混合多雲端環境的服務角色。不過，如果混合多雲端環境中的機器需要不同的許可，則您可以選擇為不同的混合式啟用建立多個服務角色。

下列處理程序說明如何使用 Systems Manager 主控台或您慣用的命令列工具來建立所需的服務角色。

使用 AWS Management Console 為 Systems Manager 混合啟用建立 IAM 服務角色

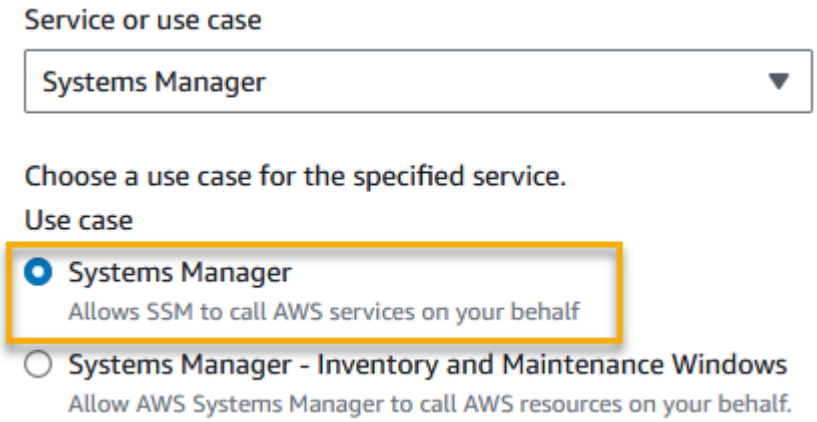
按照下列程序為混合式啟用建立服務角色。此處理程序針對 Systems Manager 核心功能使用 `AmazonSSMManagedInstanceCore` 政策。根據您的使用案例，您可能需要將其他政策新增至您的服務角色，您的內部部署機器才能存取其他 Systems Manager 工具或 AWS 服務。例如，如果無法存取所需的 AWS 受管 Amazon Simple Storage Service (Amazon S3) 儲存貯體，Patch Manager修補操作會失敗。

建立 服務角色 (主控台)

1. 在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。
2. 在導覽窗格中，選擇角色，然後選擇建立角色。
3. 對於 Select trusted entity (選擇信任的實體)，請執行以下選項：

   1. 對於 Trusted entity type (信任的實體類型)，請選擇 AWS 服務。
   2. 針對其他 的使用案例 AWS 服務，選擇 Systems Manager。
   3. 選擇 Systems Manager。

   下圖反白顯示了 Systems Manager 選項的位置。



4. 選擇下一步。
5. 在 Add permissions (新增許可) 頁面上，執行以下作業：

   - 使用 Search (搜尋) 欄位，找出 AmazonSSMManagedInstanceCore 政策。選取其名稱旁的核取方塊，如下圖所示。

> **Note**
>
> 即使您搜尋其他政策，主控台仍會保留您的選取項目。

- 如果您在程序 (選用) 建立 S3 儲存貯體存取的自訂政策 中已建立自訂 S3 儲存貯體政策，請搜尋它並選取名稱旁的核取方塊。
- 如果您計劃將non-EC2 機器加入由 管理的 Active Directory AWS Directory Service，請搜尋 AmazonSSMDirectoryServiceAccess，然後選取其名稱旁的核取方塊。
- 若您計劃使用 Eventbridge 或 CloudWatch Logs 來管理或監控您的受管節點，請搜尋 CloudWatchAgentServerPolicy，然後選取其名稱旁邊的核取方塊。

6. 選擇下一步。
7. 針對角色名稱，為您的新 IAM 角色輸入名稱 (例如 **SSMServerRole**)。

> **Note**
>
> 請記下角色名稱。當您註冊您想要使用 Systems Manager 管理的新機器時，將選擇此角色。

8. (選用) 對於描述，更新此 IAM 服務角色的描述。
9. (選用) 針對 Tags (標籤)，新增一個或多個標籤鍵值組來組織、追蹤或控制對此角色的存取。
10. 選擇 Create role (建立角色)。系統會讓您回到 Roles (角色) 頁面。

使用 AWS CLI 為 Systems Manager 混合啟用建立 IAM 服務角色

按照下列程序為混合式啟用建立服務角色。此處理程序針對 Systems Manager 核心功能使用 `AmazonSSMManagedInstanceCore` 政策。視您的使用案例而定，您可能需要為混合多雲端環境中 non-EC2 機器新增其他政策至服務角色，才能存取其他工具或 AWS 服務。

S3 儲存貯體政策要求

在下列任一案例中，您必須先為 Amazon Simple Storage Service (Amazon S3) 儲存貯體建立自訂 IAM 許可政策，才能完成此程序：

- 案例 1 – 您使用 VPC 端點將 VPC 私下連線至支援的 AWS 服務 和 支援的 VPC 端點服務 AWS PrivateLink。
- 案例 2 – 您計劃在 Systems Manager 操作過程中使用您建立的 Amazon S3 儲存貯體，例如將 Run Command 命令或 Session Manager 工作階段的輸出存放到 S3 儲存貯體。在繼續進行之前，請先遵循為執行個體設定檔建立一個自訂 S3 儲存貯體政策中的步驟。該主題中的 S3 儲存貯體政策相關資訊也適用於您的服務角色。

AWS CLI

建立適用於混合多雲端環境的 IAM 服務角色 (AWS CLI)

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

2. 在您的本機機器上，使用下列信任政策，建立名稱為 SSMService-Trust.json 的文字檔案。請務必將檔案儲存為 .json 副檔名。請務必在您建立混合啟用的 ARN AWS 區域 中指定 AWS 帳戶 和 。將帳戶 ID 和區域的*#####*取代為您自己的資訊。

```
{
   "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. 開啟 AWS CLI，並在您建立 JSON 檔案的目錄中執行 create-role 命令來建立服務角色。此範例會建立名稱為 SSMServiceRole 的角色。如果您想要的話，可以選擇其他名稱。

   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. 如下執行 attach-role-policy 命令，以允許您剛建立的服務角色建立工作階段字符。工作階段字符可讓受管節點具有使用 Systems Manager 執行命令的許可。

   > **Note**
   >
   > 您在混合多雲端環境中為受管節點的服務描述檔新增的政策，與用於為 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體建立執行個體設定檔的政策相同。如需有關下列命令中所用 AWS 政策的詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可。

   （必要）執行下列命令，以允許受管節點使用 AWS Systems Manager 服務核心功能。

   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
```

如果您為服務角色建立自訂 S3 儲存貯體政策，請執行下列命令，以允許 AWS Systems Manager 客服人員 (SSM Agent) 存取您在政策中指定的儲存貯體。使用您的 AWS 帳戶 ID 和儲存貯體名稱取代 *account-id* 和 *amzn-s3-demo-bucket*。

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
```

（選用）執行下列命令，SSM Agent以允許 AWS Directory Service 代表您存取 ，請求由受管節點加入網域。只有在您將節點加入 Microsoft AD 目錄時，服務角色才需要此政策。

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
```

(選用) 執行以下命令，以允許 CloudWatch 代理程式在您的受管節點上執行。此命令可讓您讀取節點的資訊，並將資訊寫入 CloudWatch。只有在使用諸如 Amazon EventBridge 或 Amazon CloudWatch Logs 等服務時，您的服務設定檔才需要此政策。

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

Tools for PowerShell

建立適用於混合多雲端環境的 IAM 服務角色 (AWS Tools for Windows PowerShell)

1. 如果您尚未安裝並設定 AWS Tools for PowerShell (Tools for Windows PowerShell)。

   如需相關資訊，請參閱安裝 AWS Tools for PowerShell。

2. 在您的本機機器上，使用下列信任政策，建立名稱為 SSMService-Trust.json 的文字檔案。請務必將檔案儲存為 .json 副檔名。請務必在您建立混合啟用的 ARN AWS 區域 中指定 AWS 帳戶 和 。

```
{
   "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:region:123456789012:*"
            }
         }
      }
   ]
}
```

3. 在管理模式下開啟 PowerShell，並在您建立 JSON 檔案的目錄中如下所示執行 New-IAMRole，以建立服務角色。此範例會建立名稱為 SSMServiceRole 的角色。如果您想要的話，可以選擇其他名稱。

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

4. 如下使用 Register-IAMRolePolicy，以允許您建立的服務角色建立工作階段字符。工作階段字符可讓受管節點具有使用 Systems Manager 執行命令的許可。

> **Note**
>
> 您在混合多雲端環境中為受管節點的服務描述檔新增的政策，與用於為 EC2 執行個體建立執行個體設定檔的政策相同。如需下列命令中使用的 AWS 政策的詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可。

（必要）執行下列命令，以允許受管節點使用 AWS Systems Manager 服務核心功能。

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

若您為服務角色建立自訂 S3 儲存貯體政策，請執行下列命令以允許 SSM Agent 存取您在政策中指定的儲存貯體。將 *account-id* 和 *my-bucket-policy-name* 取代為您的 AWS 帳戶 ID 和儲存貯體名稱。

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

（選用）執行下列命令，SSM Agent以允許 AWS Directory Service 代表您存取 ，請求由受管節點加入網域。只有在您將節點加入 Microsoft AD 目錄時，服務角色才需要此政策。

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

(選用) 執行以下命令，以允許 CloudWatch 代理程式在您的受管節點上執行。此命令可讓您讀取節點的資訊，並將資訊寫入 CloudWatch。只有在使用諸如 Amazon EventBridge 或 Amazon CloudWatch Logs 等服務時，您的服務設定檔才需要此政策。

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

繼續進行建立混合啟用，以向 Systems Manager 註冊節點。

建立混合啟用，以向 Systems Manager 註冊節點

若要在混合多雲端環境中將 Amazon Elastic Compute Cloud (EC2) 執行個體以外的機器設定為的受管節點，您可以建立並套用混合啟用。成功完成啟用之後，您會在主控台頁面頂部的立即收到啟用代碼和啟用 ID。當您在混合多雲端環境non-EC2 機器 AWS Systems Manager SSM Agent上安裝 時，您可以指定此程式碼和 ID 組合。代碼和 ID 可讓您從受管節點中安全地存取 Systems Manager 服務。

> **Important**
>
> Systems Manager 會立即將啟用代碼和 ID 傳回主控台或命令視窗，視您如何建立啟用而定。複製此資訊，並將其存放在安全的地方。如果您離開主控台或關閉命令視窗，您可能會遺失此資訊。如果您遺失此資訊，您必須建立新的啟用。

關於啟用過期

啟用過期是一個時段，您可以在這個時段中向 Systems Manager 註冊內部部署機器。過期的啟用不會對您先前向 Systems Manager 註冊的伺服器或虛擬機器產生任何影響。若啟用過期，您便無法使用該特定啟用向 Systems Manager 註冊更多伺服器或虛擬機器。您只能建立新的環境。

您之前註冊的每個內部部署伺服器和虛擬機器仍會註冊為 Systems Manager 受管節點，直到您明確將其取消註冊為止。您可以透過以下方式取消註冊non-EC2 受管節點：

- 在 Systems Manager 主控台中使用 Fleet Manager中的受管節點索引標籤
- 使用 AWS CLI 命令 deregister-managed-instance
- 使用 API 動作 DeregisterManagedInstance。

如需詳細資訊，請參閱下列主題

- 取消註冊並重新註冊受管節點 (Linux)

- 取消註冊並重新註冊受管節點 (Windows Server)

關於受管節點

受管節點是針對 設定的任何機器 AWS Systems Manager。 AWS Systems Manager 支援 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、邊緣裝置和內部部署伺服器或 VMs，包括其他雲端環境中VMs。以前，受管節點都稱為受管執行個體。術語執行個體現在僅指 EC2 執行個體。在此術語變更之前，命令命名為 deregister-managed-instance。

關於啟用標籤

如果您使用 AWS Command Line Interface (AWS CLI) 或 建立啟用 AWS Tools for Windows PowerShell，則可以指定標籤。標籤是您指派給資源的選用性中繼資料。標籤允許您以不同的方式 (例如用途、擁有者或環境) 將資源分類。以下是在美國東部 （俄亥俄） 區域，在包含選用標籤的本機 Linux 機器上執行 AWS CLI 的範例命令。

```
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"
```

若您在建立啟用時指定標籤，那些標籤便會在您啟用它們時自動指派給您的受管節點。

您無法將標籤新增到現有的啟用，或是從現有的啟用刪除標籤。若您不希望使用啟用將標籤自動指派給您的現場部署伺服器和 VM，您可以稍後再為他們新增標籤。具體而言，您可以在內部部署伺服器和虛擬機器首次連線到 Systems Manager 後，為它們新增標籤。在它們連線後，便會被指派受管節點 ID，並在 Systems Manager 主控台中列出，其 ID 也會加上 "mi-" 字首。

> **Note**
>
> 若您使用 Systems Manager 主控台建立啟用，則無法為其指派標籤。您必須使用 AWS CLI 或 Tools for Windows PowerShell 來建立它。

如果您不想再使用 Systems Manager 來管理內部部署伺服器或虛擬機器 (VM)，您可以將其取消註冊。如需相關資訊，請參閱 取消註冊混合多雲端環境中的受管節點。

主題

- 使用 AWS Management Console 建立啟用，以向 Systems Manager 註冊受管節點
- 使用命令列建立啟用，以向 Systems Manager 註冊受管節點

使用 AWS Management Console 建立啟用，以向 Systems Manager 註冊受管節點

建立受管節點啟用

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Hybrid Activations (混合啟用)。
3. 選擇建立啟用。

   -或-

   如果您第一次在目前 中存取混合啟用 AWS 區域，請選擇建立啟用。
4. (選用) 在 Activation description (啟用描述)，輸入此啟用的描述。如果您計劃啟用大量伺服器和虛擬機器，則建議您輸入描述。
5. 針對執行個體限制，指定您要在 註冊的節點總數， AWS 做為此啟用的一部分。預設值為 1 個執行個體。
6. 對於 IAM 角色，請選擇服務角色選項，讓您的伺服器和 VMs 在雲端 AWS Systems Manager 中與 通訊：

   - 選項 1：選擇 Use the default role created by the system (使用系統建立的預設角色) 來使用AWS提供的角色及受管理政策。
   - 選項 2：選擇 Select an existing custom IAM role that has the required permissions (選取具有所需許可的現有自訂 IAM 角色)，以使用您先前建立的選用自訂角色。此角色必須擁有可指定 `"Service": "ssm.amazonaws.com"` 的信任關係政策。如果您的 IAM 角色未在信任關係政策中指定此準則，您會收到下列錯誤：

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

     如需建立此角色的詳細資訊，請參閱在混合多雲端環境中建立 Systems Manager 所需的 IAM 服務角色。

7. 在 Activation expiry date (啟用過期日)，指定啟用的過期日期。過期日期必須為未來的日期，不能超過 30 天。預設值為 24 小時。

> **Note**
>
> 如果想要在過期日期之後註冊其他的受管節點，您必須建立新的啟用。該過期日期不會影響已經註冊與正在執行的節點。

8. (選用)對於 Default instance name (預設執行個體) 欄位，針對此啟用的所有相關受管節點指定要顯示的識別名稱值。
9. 選擇建立啟用。Systems Manager 會立即將啟用代碼和 ID 傳回主控台。

使用命令列建立啟用，以向 Systems Manager 註冊受管節點

下列程序說明如何使用 AWS Command Line Interface (AWS CLI) （在 Linux 或 上Windows Server) 或 AWS Tools for PowerShell 來建立受管節點啟用。

建立啟用

1. AWS Tools for PowerShell如果您尚未安裝和設定 AWS CLI 或 。

   如需相關資訊，請參閱安裝或更新 AWS CLI的最新版本和安裝 AWS Tools for PowerShell。
2. 執行以下命令來建立啟用。

> **Note**
>
> - 在下列命令中，用您自己的資訊取代*##*。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域欄位。
> - 您為 *iam-role* 參數指定的角色必須擁有可指定 `"Service": "ssm.amazonaws.com"` 的信任關係政策。如果您的 AWS Identity and Access Management (IAM) 角色未在信任關係政策中指定此原則，您會收到下列錯誤：
>
>   ```
>   An error occurred (ValidationException) when calling the CreateActivation
>                                           operation: Not existing role:
>    arn:aws:iam::<accountid>:role/SSMRole
>   ```
>
>   如需建立此角色的詳細資訊，請參閱在混合多雲端環境中建立 Systems Manager 所需的 IAM 服務角色。

- 對於 `--expiration-date`，以時間戳記格式提供日期，例如 `"2021-07-07T00:00:00"`，適用於啟用代碼到期時。您可以在 30 天前指定日期。如果您未提供過期日期，啟用代碼將在 24 小時內過期。

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"}
```

請見此處範例。

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"}
```

若成功建立啟用，系統會立即傳回啟用代碼和 ID。

在混合 Linux 節點上安裝 SSM Agent

本主題說明如何在混合多雲端環境中non-EC2 (Amazon Elastic Compute Cloud) Linux 機器上安裝 AWS Systems Manager SSM Agent 。如需在 Linux 的 EC2 執行個體SSM Agent上安裝 的資訊，請參閱 在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent。

開始之前，請先找到混合啟用程序期間產生的啟用碼和啟用 ID，如 中所述建立混合啟用，以向 Systems Manager 註冊節點。您需在以下程序中指定代碼和 ID。

在混合多雲端環境中的非 EC2 機器上安裝 SSM Agent

1. 登入混合多雲端環境中的伺服器或虛擬機器。
2. 如果您使用 HTTP 或 HTTPS 代理伺服器，則必須在目前的 Shell 工作階段中設定 `http_proxy` 或 `https_proxy` 環境變數。如果您不使用代理伺服器，則可以略過此步驟。

   對於 HTTP 代理伺服器，請在命令列輸入下列命令：

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

   對於 HTTPS 代理伺服器，請在命令列輸入下列命令：

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

3. 複製以下其中一個命令區塊並貼到 SSH。將預留位置值取代為混合啟用程序期間產生的啟用碼和啟用 ID，並將 取代為 AWS 區域 您要SSM Agent從中下載的識別符，然後按 `Enter`。

   > **Important**
   >
   > 請注意以下重要詳細資訊：
   >
   > - 使用 `ssm-setup-cli`進行non-EC2 安裝可最大化 Systems Manager 安裝和組態的安全性。
   > - 如果您是根使用者，則不需要 `sudo`。
   > - `ssm-setup-cli` 從與建立混合啟用的 AWS 區域 相同位置下載。
   > - `ssm-setup-cli` 支援用於確定代理程式下載來源的 `manifest-url` 選項。除非您的組織需要，否則請勿為此選項指定值。
   > - 註冊執行個體時，請僅使用為 `ssm-setup-cli` 提供的下載連結。`ssm-setup-cli` 不應單獨存放以供日後使用。
   > - 您可以使用此處提供的指令碼來驗證 `ssm-setup-cli` 的簽章。

region 代表 支援的 AWS 區域 識別符 AWS Systems Manager，例如us-east-2美國東部（俄亥俄）區域。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域欄位。

此外，ssm-setup-cli 還包括下列選項：

- version：有效值為 latest 和 stable。
- downgrade：允許 SSM Agent 降級至較早的版本。指定 true 以安裝較早版本的代理程式。
- skip-signature-validation：在下載和安裝代理程式期間略過簽章驗證。

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 (先前為 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 套件

  ```
  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 套件

  您不需要指定 URL 以供下載，因為 snap 命令會自動從 Snap 應用程式商店下載代理程式，網址為 https://snapcraft.io。

在 Ubuntu Server 20.10 STR & 20.04、18.04 和 16.04 LTS 上，SSM Agent 安裝程式檔案 (包括代理程式二進位程式碼和組態檔案) 存放在以下目錄中：/snap/amazon-ssm-agent/current/。如果您變更此目錄中的任何組態檔案，則必須將這些檔案從 /snap 目錄複製到 /etc/amazon/ssm/ 目錄。日誌和程式庫檔案未變更 (/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 商店中的候選頻道包含最新版本的 SSM Agent，而不是穩定的頻道。如果您想追蹤候選頻道上的 SSM Agent 版本資訊，請在 Ubuntu Server 18.04 和 16.04 LTS 64 位元受管節點上執行下列命令。

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

該命令會下載並安裝 SSM Agent 到混合多雲端環境中的啟用混合模式機器上。該命令會停止 SSM Agent，然後使用 Systems Manager 服務來註冊此機器。此機器現在是受管節點。為 Systems Manager 設定的 Amazon EC2 執行個體也是受管節點。不過，在 Systems Manager 主控台中，啟用混合模式節點的字首為 "mi-"，有別於 Amazon EC2 執行個體。

繼續進行在混合 Windows Server 節點上安裝 SSM Agent。

設定私有金鑰自動輪換

若要強化您的安全狀態，您可以設定 AWS Systems Manager Agent (SSM Agent) 自動輪換混合多雲端環境的私有金鑰。您可以使用 SSM Agent 3.0.1031.0 版或更新版本來存取此功能。使用下列程序開啟此功能。

設定 SSM Agent 以輪換混合多雲端環境的私有金鑰

1. 在 Linux 機器中導覽至 /etc/amazon/ssm/，或在 Windows 機器中導覽至 C:\Program Files\Amazon\SSM。

2. 將 amazon-ssm-agent.json.template 的內容複製到名為 amazon-ssm-agent.json 的新檔案。將 amazon-ssm-agent.json 儲存在 amazon-ssm-agent.json.template 所在的相同目錄中。
3. 查找 Profile、KeyAutoRotateDays。輸入您想要的自動私有金鑰輪換之間的天數。
4. 重新啟動 SSM Agent。

每次變更組態時，請重新啟動 SSM Agent。

您可以使用相同的程序來自訂 SSM Agent 的其他功能。如需可用組態屬性及其預設值的最新清單，請參閱組態屬性定義。

取消註冊並重新註冊受管節點 (Linux)

您可以從 AWS CLI 或 Tools for Windows PowerShell 呼叫 DeregisterManagedInstance API 操作，取消註冊混合啟用的受管節點。以下是範例 CLI 命令：

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

若要移除代理程式的剩餘註冊資訊，請移除 amazon-ssm-agent.json 檔案中的 IdentityConsumptionOrder 索引鍵。然後，根據您的安裝類型，執行下列其中一個命令。

在使用 Snap 套件安裝 SSM Agent的Ubuntu Server節點上：

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

在所有其他 Linux 安裝上：

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

在非 EC2 Linux 機器上重新註冊受管節點

您可以在取消註冊機器之後重新註冊它。不過，您必須使用與先前註冊機器時不同的啟用代碼和啟用 ID。

1. 連線至您的機器。
2. 執行下列命令。確保將預留位置值取代為您在建立受管節點啟用時產生的啟用碼和啟用 ID，以及取代為您想要從中下載 SSM Agent 的區域識別符。

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

診斷並解決在 EC2 Linux 機器上安裝 SSM Agent 的問題

使用以下資訊，解決在混合多雲端環境中的啟用混合模式 Linux 機器上安裝 SSM Agent 的問題。

您會收到 DeliveryTimedOut 錯誤

問題：將機器設定為個別的 AWS 帳戶 受管節點時 AWS 帳戶，您會在執行命令以在目標機器SSM Agent上安裝 `DeliveryTimedOut`後收到 。

解決方案：`DeliveryTimedOut` 是此情況的預期回應代碼。在目標節點上安裝 SSM Agent 的命令會變更來源節點的節點 ID。因為節點 ID 已變更，所以來源節點無法回覆在執行時命令已失敗、已完成或已逾時的目標節點。

無法載入節點關聯

問題：執行安裝命令之後，您會在 SSM Agent 錯誤日誌中看到下列錯誤：

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

如果機器 ID 在重新開機後未持續存在，則會看到此錯誤。

解決方案：若要解決此問題，請執行下列命令。此命令會強制機器 ID 在重新開機後持續存在。

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

在混合 Windows Server 節點上安裝 SSM Agent

本主題說明如何在混合多雲端環境中的Windows Server機器上安裝 AWS Systems Manager SSM Agent 。如需在 EC2 執行個體上針對 Windows Server 安裝 SSM Agent，請參閱 在 Windows Server EC2 執行個體上手動安裝和解除安裝 SSM Agent。

開始之前，請找到混合啟用程序期間產生的啟用碼和啟用 ID，如 中所述建立混合啟用，以向 Systems Manager 註冊節點。您需在以下程序中指定代碼和 ID。

在混合多雲端環境中的非 EC2 Windows Server 機器上安裝 SSM Agent

1. 登入混合多雲端環境中的伺服器或虛擬機器。
2. 如果您使用 HTTP 或 HTTPS 代理伺服器，則必須在目前的 Shell 工作階段中設定 `http_proxy` 或 `https_proxy` 環境變數。如果您不使用代理伺服器，則可以略過此步驟。

   對於 HTTP 代理伺服器，請設定此變數：

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

   對於 HTTPS 代理伺服器，請設定此變數：

   ```
   http_proxy=http://hostname:port
   https_proxy=https://hostname:port
   ```
3. 以高階 (管理) 模式開啟 Windows PowerShell。
4. 複製以下命令區塊並貼到 Windows PowerShell。將每個########取代為您自己的資訊。例如，當您建立混合啟用時產生的啟用碼和啟用 ID，以及 AWS 區域 您要SSM Agent從中下載的識別符。

   > **Important**
   >
   > 請注意以下重要詳細資訊：
   >
   > - 使用 `ssm-setup-cli`進行non-EC2 安裝可將 Systems Manager 安裝和組態的安全性最大化。
   > - `ssm-setup-cli` 支援用於確定代理程式下載來源的 `manifest-url` 選項。除非您的組織需要，否則請勿為此選項指定值。
   > - 您可以使用此處提供的指令碼來驗證 `ssm-setup-cli` 的簽章。
   > - 註冊執行個體時，請僅使用為 `ssm-setup-cli` 提供的下載連結。`ssm-setup-cli` 不應單獨存放以供日後使用。

   region 代表 支援的 AWS 區域 識別符 AWS Systems Manager，例如`us-east-2`美國東部（俄亥俄）區域。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域欄位。

此外，ssm-setup-cli 還包括下列選項：

- version：有效值為 latest 和 stable。
- downgrade：將代理程式還原為較早版本。
- skip-signature-validation：在下載和安裝代理程式期間略過簽章驗證。

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 鍵。

> **Note**
>
> 如果命令失敗，請確認您正在執行最新版本的 AWS Tools for PowerShell。

命令會執行下列動作：

- 將 SSM Agent 下載並安裝到機器上。
- 向 Systems Manager 服務註冊機器。
- 傳回類似如下的請求回應：

```
    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
```

此機器現在是受管節點。這些受管節點現在會使用字首 "mi-" 進行標識。您可以使用 AWS CLI 命令 Fleet Manager或使用 API 命令 describe-instance-information，在 的受管節點頁面上檢視受管節點DescribeInstanceInformation。

設定私有金鑰自動輪換

若要強化您的安全狀態，您可以設定 AWS Systems Manager Agent (SSM Agent) 自動輪換混合多雲端環境的私有金鑰。您可以使用 SSM Agent 3.0.1031.0 版或更新版本來存取此功能。使用下列程序開啟此功能。

設定 SSM Agent 以輪換混合多雲端環境的私有金鑰

1. 在 Linux 機器中導覽至 `/etc/amazon/ssm/`，或在 Windows Server 機器中導覽至 `C:\Program Files\Amazon\SSM`。
2. 將 `amazon-ssm-agent.json.template` 的內容複製到名為 `amazon-ssm-agent.json` 的新檔案。將 `amazon-ssm-agent.json` 儲存在 `amazon-ssm-agent.json.template` 所在的相同目錄中。
3. 查找 `Profile`、`KeyAutoRotateDays`。輸入您想要的自動私有金鑰輪換之間的天數。
4. 重新啟動 SSM Agent。

每次變更組態時，請重新啟動 SSM Agent。

您可以使用相同的程序來自訂 SSM Agent 的其他功能。如需可用組態屬性及其預設值的最新清單，請參閱組態屬性定義。

取消註冊並重新註冊受管節點 (Windows Server)

您可以從 AWS CLI 或 Tools for Windows PowerShell 呼叫 DeregisterManagedInstance API 操作來取消註冊受管節點。以下是範例 CLI 命令：

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

若要移除代理程式的剩餘註冊資訊，請移除 `amazon-ssm-agent.json` 檔案中的 `IdentityConsumptionOrder` 索引鍵。然後執行以下命令：

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

若要在 Windows Server 混合機器中重新註冊受管節點

您可以在取消註冊機器之後重新註冊它。不過，您必須使用與先前註冊機器時不同的啟用代碼和啟用 ID。

1. 連線至您的機器。
2. 執行下列命令。確保將預留位置值替換為您在建立混合啟用時產生的啟用碼和啟用 ID，以及替換為您想要從中下載 SSM Agent 的區域識別符。

```
'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 管理邊緣裝置

本節說明帳戶和系統管理員為了啟用 AWS IoT Greengrass 核心裝置的組態和管理而執行的設定任務。完成這些任務後，管理員授予許可的使用者 AWS 帳戶 可以使用 AWS Systems Manager 來設定和管理其組織 AWS IoT Greengrass 的核心裝置。

> **Note**
>
> - SSM Agent macOS和 Windows 10 AWS IoT Greengrass 不支援 的 。您無法使用 Systems Manager 工具來管理和設定使用這些作業系統的邊緣裝置。
> - Systems Manager 也支援未設定為 AWS IoT Greengrass 核心裝置的邊緣裝置。若要使用 Systems Manager 管理 AWS IoT 核心裝置和非AWS 邊緣裝置，您必須使用混合式啟用來設定它們。如需詳細資訊，請參閱使用 Systems Manager 在混合多雲端環境中管理節點。
> - 使用 Session Manager 和 Microsoft 應用程式修補您的邊緣裝置，您必須啟用進階執行個體層。如需詳細資訊，請參閱開啟 advanced-instances 方案。

開始之前

確認邊緣裝置符合下列需求。

- 您的邊緣裝置必須符合要求，才能設定為 AWS IoT Greengrass 核心裝置。如需詳細資訊，請參閱《 AWS IoT Greengrass Version 2 開發人員指南》中的設定 AWS IoT Greengrass 核心裝置。
- 您的邊緣裝置必須與 AWS Systems Manager 代理程式 () 相容SSM Agent。如需詳細資訊，請參閱Systems Manager 支援的作業系統。
- 邊緣裝置必須能與雲端的 Systems Manager 服務進行通訊。Systems Manager 不支援中斷連線的邊緣裝置。

關於設定邊緣裝置

為 Systems Manager 設定 AWS IoT Greengrass 裝置涉及下列程序。

> **Note**
>
> 如需有關SSM Agent從邊緣裝置解除安裝的資訊，請參閱《 AWS IoT Greengrass Version 2 開發人員指南》中的解除安裝 AWS Systems Manager 代理程式。

為邊緣裝置建立 IAM 服務角色

AWS IoT Greengrass 核心裝置需要 AWS Identity and Access Management (IAM) 服務角色才能與 通訊 AWS Systems Manager。角色會將AssumeRole信任授予 Systems Manager 服務 AWS Security Token Service (AWS STS)。您只需要為每個 AWS 帳戶建立一次服務角色。當您設定和部署SSM Agent元件至 AWS IoT Greengrass 裝置時，您將為 `RegistrationRole` 參數指定此角色。如果您在針對混合多雲端環境設定非 EC2 節點時已建立此角色，則可以略過此步驟。

> **Note**
>
> 必須在 IAM 中為將在邊緣裝置上使用 Systems Manager 的公司或組織使用者授予呼叫 Systems Manager API 的許可。

S3 儲存貯體政策要求

在下列任一案例中，您必須先為 Amazon Simple Storage Service (Amazon S3) 儲存貯體建立自訂 IAM 許可政策，才能完成此程序：

- 案例 1：您使用 VPC 端點將 VPC 私下連線至支援的 AWS 服務 和 支援的 VPC 端點服務 AWS PrivateLink。
- 案例 2：您計劃在 Systems Manager 操作過程中使用您建立的 S3 儲存貯體，例如將 Run Command 命令或 Session Manager 工作階段的輸出儲存到 S3 儲存貯體。在繼續進行之前，請先遵循為執行個體設定檔建立一個自訂 S3 儲存貯體政策中的步驟。該主題中的 S3 儲存貯體政策相關資訊也適用於您的服務角色。

> **Note**
>
> 如果您的裝置受到防火牆保護，且您計劃使用 Patch Manager，則防火牆必須允許存取修補基準端點 `arn:aws:s3:::patch-baseline-snapshot-`*`region`*`/*`。*`region`* 代表 支援的 AWS 區域 識別符 AWS Systems Manager，例如`us-east-2`美國東部（俄亥俄）區域。如需支援的 *`region`* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域欄位。

AWS CLI

為 AWS IoT Greengrass 環境建立 IAM 服務角色 (AWS CLI)

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

2. 在您的本機機器上，使用下列信任政策，建立名稱為 SSMService-Trust.json 的文字檔案。請務必將檔案儲存為 .json 副檔名。

   > **Note**
   >
   > 記下名稱。您將在部署SSM Agent到 AWS IoT Greengrass 核心裝置時指定它。

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

3. 開啟 AWS CLI，並在您建立 JSON 檔案的目錄中執行 create-role 命令來建立服務角色。將每個########取代為您自己的資訊。

   Linux 與 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. 如下執行 attach-role-policy 命令，以允許您剛建立的服務角色建立工作階段字符。工作階段字符可讓邊緣裝置具有使用 Systems Manager 執行命令的許可。

> **Note**
>
> 您為邊緣裝置的服務設定檔新增的政策，與用於為 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體建立執行個體設定檔的政策相同。如需有關下列命令中所用 IAM 政策的詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可。

（必要）執行下列命令，以允許邊緣裝置使用 AWS Systems Manager 服務核心功能。

Linux 與 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
```

如果您為服務角色建立自訂 S3 儲存貯體政策，請執行下列命令，以允許 AWS Systems Manager 客服人員 (SSM Agent) 存取您在政策中指定的儲存貯體。將 *account-ID* 和 *my-bucket-policy-name* 取代為您的 AWS 帳戶 ID 和儲存貯體名稱。

Linux 與 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
```

(選用) 執行下列命令以允許 SSM Agent 代表您存取 AWS Directory Service ，以請求從邊緣設置加入網域。只有在您將邊緣裝置加入 Microsoft AD 目錄時，服務角色才需要此政策。

Linux 與 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
```

(選用) 執行以下命令，以允許 CloudWatch 代理程式在您的邊緣裝置上執行。此命令可讓您讀取裝置的資訊，並將資訊寫入 CloudWatch。只有在使用諸如 Amazon EventBridge 或 Amazon CloudWatch Logs 等服務時，您的服務角色才需要此政策。

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

Tools for PowerShell

為 AWS IoT Greengrass 環境建立 IAM 服務角色 (AWS Tools for Windows PowerShell)

1. 如果您尚未安裝並設定 AWS Tools for PowerShell (Tools for Windows PowerShell)。

   如需相關資訊，請參閱安裝 AWS Tools for PowerShell。

2. 在您的本機機器上，使用下列信任政策，建立名稱為 `SSMService-Trust.json` 的文字檔案。請務必將檔案儲存為 `.json` 副檔名。

> **Note**
>
> 記下名稱。您將在部署SSM Agent到 AWS IoT Greengrass 核心裝置時指定它。

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

3. 在管理模式下開啟 PowerShell，並在您建立 JSON 檔案的目錄中如下所示執行 New-IAMRole，以建立服務角色。

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

4. 如下使用 Register-IAMRolePolicy，以允許您建立的服務角色建立工作階段字符。工作階段字符可讓邊緣裝置具有使用 Systems Manager 執行命令的許可。

> **Note**
>
> 您在 AWS IoT Greengrass 環境中為邊緣裝置之服務角色新增的政策，與用於為 EC2 執行個體建立執行個體設定檔的政策相同。如需下列命令中使用的 AWS 政策的詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可。

（必要）執行下列命令，以允許邊緣裝置使用 AWS Systems Manager 服務核心功能。

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

若您為服務角色建立自訂 S3 儲存貯體政策，請執行下列命令以允許 SSM Agent 存取您在政策中指定的儲存貯體。將 *account-ID* 和 *my-bucket-policy-name* 取代為您的 AWS 帳戶 ID 和儲存貯體名稱。

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

(選用) 執行下列命令以允許 SSM Agent 代表您存取 AWS Directory Service ，以請求從邊緣設置加入網域。只有在您將邊緣裝置加入 Microsoft AD 目錄時，服務角色才需要此政策。

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

(選用) 執行以下命令，以允許 CloudWatch 代理程式在您的邊緣裝置上執行。此命令可讓您讀取裝置的資訊，並將資訊寫入 CloudWatch。只有在使用諸如 Amazon EventBridge 或 Amazon CloudWatch Logs 等服務時，您的服務角色才需要此政策。

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

設定 的邊緣裝置 AWS IoT Greengrass

將邊緣裝置設定為 AWS IoT Greengrass 核心裝置。設定程序涉及驗證支援的作業系統和系統需求，以及在您的裝置上安裝和設定 AWS IoT Greengrass 核心軟體。如需詳細資訊，請參閱《AWS IoT Greengrass Version 2 開發人員指南》中的設定 AWS IoT Greengrass 核心裝置。

更新 AWS IoT Greengrass 權杖交換角色，並在SSM Agent您的邊緣裝置上安裝

設定和設定 Systems Manager AWS IoT Greengrass 核心裝置的最後步驟，需要您更新 AWS IoT Greengrass AWS Identity and Access Management (IAM) 裝置服務角色，也稱為字符交換角色，並將 AWS Systems Manager 代理程式 (SSM Agent) 部署至 AWS IoT Greengrass 您的裝置。如需有關這些程序的詳細資訊，請參閱《AWS IoT Greengrass Version 2 開發人員指南》中的安裝 AWS Systems Manager Agent 一節。

部署SSM Agent至裝置後， AWS IoT Greengrass 會自動向 Systems Manager 註冊您的裝置。不需要額外註冊。您可以開始使用 Systems Manager 工具來存取、管理和設定 AWS IoT Greengrass 您的裝置。

> **Note**
>
> 邊緣裝置必須能與雲端的 Systems Manager 服務進行通訊。Systems Manager 不支援中斷連線的邊緣裝置。

為 Systems Manager 建立 AWS Organizations 委派管理員

當您在 中設定組織時 AWS Organizations，您會指派管理帳戶以執行所有管理任務 AWS 服務。管理帳戶使用者只能為 Systems Manager 指派委派的管理員帳戶，以執行 Change Manager、 Explorer 和 的管理任務OpsCenter。 AWS Organizations 是可用來建立組織並指派 AWS 帳戶 以集中管理這些帳戶的帳戶管理服務。如需詳細資訊 AWS Organizations，請參閱AWS Organizations 《 使用者指南AWS Organizations》中的 。

Change ManagerExplorer、 和 OpsCenter中的工具 AWS Systems Manager，使用 AWS Organizations 來執行組織所有成員帳戶的任務。您只能為所有 Systems Manager 工具指派一個委派管理員。委派管理員帳戶必須是指派給組織的成員。

主題

- 將委派管理員與 Change Manager 搭配使用
- 將委派管理員與 Explorer 搭配使用
- 將委派管理員與 OpsCenter 搭配使用
- 將委派管理員與 Quick Setup 搭配使用

將委派管理員與 Change Manager 搭配使用

Change Manager 是一個企業變更管理架構，用於請求、核准、實作和報告應用程式組態和基礎設施的操作變更。

如果您在整個組織中使用 Change Manager，請指派受委派管理員帳戶來管理所有成員帳戶的變更範本、核准和報告。藉助Quick Setup，您可以設定 Change Manager 與組織搭配使用，以及選取委派管理員帳戶。如果您Change Manager搭配單一 使用 AWS 帳戶，則不需要委派的管理員帳戶。

依預設，Change Manager 會在受委派管理員帳戶中顯示所有變更相關任務。如需在為組織設定 Change Manager 時設定受委派管理員的指示，請參閱 設定適用於組織的 Change Manager (管理帳戶)。

> **Important**
>
> 如果您在整個組織中使用 Change Manager，我們建議始終從委派管理員帳戶進行變更。雖然您可以從組織中的其他帳戶進行變更，但這些變更將不會在受委派管理員帳戶中報告，也不可在其中檢視。

將委派管理員與 Explorer 搭配使用

Explorer 是可自訂的操作儀表板 AWS 帳戶，可報告您跨 的操作資料 (OpsData) 的彙總檢視 AWS 區域。

您可以設定 Systems Manager 的委派管理員帳戶，透過使用資源Explorer資料同步來彙總來自多個區域和帳戶的資料 AWS Organizations。委派管理員可以使用 、 AWS Command Line Interface (AWS CLI) 或 來搜尋 AWS Management Console、篩選和彙總Explorer資料 AWS Tools for Windows PowerShell。

當您將受委派管理員帳戶用於 Explorer 時，您會限制可建立或刪除與個別 AWS 帳戶的多帳戶和區域資源資料同步的管理員數目。

您可以使用 來同步 AWS 帳戶 組織中所有 的操作資料Explorer。如需如何從 Explorer 指派受委派管理員的資訊，請參閱 設定 Explorer 的委派管理員。

將委派管理員與 OpsCenter 搭配使用

OpsCenter 提供操作工程師和 IT 專業人員可以管理與 AWS 資源相關的操作工作項目 (OpsItems) 的集中位置。如果您想要使用 OpsCenter 跨帳戶集中管理 OpsItems，則必須在 AWS Organizations中設定組織。

為 OpsCenter 使用 Quick Setup 時，您可以指派受委派管理員帳戶，並設定 OpsCenter 以集中管理 OpsItems。如需詳細資訊，請參閱(選用) 使用 Quick Setup 設定 OpsCenter 以跨帳戶管理 OpsItems。

將委派管理員與 Quick Setup 搭配使用

Quick Setup 是 Systems Manager 中的工具，可協助您使用建議的最佳實務快速設定常用 AWS 服務和功能。您可以為 設定委派管理員帳戶Quick Setup，以協助您使用 在帳戶和區域之間部署和管理組態 AWS Organizations。Quick Setup 的委派管理員可以建立、更新、檢視和刪除組織中的組態管理員資

源。在整合式主控台體驗設定程序中，Systems Manager 會為 Quick Setup 註冊委派管理員。如需詳細資訊，請參閱為組織設定 Systems Manager 統一主控台。

的一般設定 AWS Systems Manager

如果您尚未這麼做，請註冊 AWS 帳戶 並建立管理使用者。

註冊 AWS 帳戶

如果您沒有 AWS 帳戶，請完成下列步驟來建立一個。

註冊 AWS 帳戶

1. 開啟 https://portal.aws.amazon.com/billing/signup。
2. 請遵循線上指示進行。

   部分註冊程序需接收來電或簡訊，並在電話鍵盤輸入驗證碼。

   當您註冊 時 AWS 帳戶，AWS 帳戶根使用者會建立 。根使用者有權存取該帳戶中的所有 AWS 服務 和資源。作為安全最佳實務，請將管理存取權指派給使用者，並且僅使用根使用者來執行需要根使用者存取權的任務。

AWS 會在註冊程序完成後傳送確認電子郵件給您。您可以隨時登錄 https://aws.amazon.com/ 並選擇我的帳戶，以檢視您目前的帳戶活動並管理帳戶。

建立具有管理存取權的使用者

註冊 後 AWS 帳戶，請保護 AWS 帳戶根使用者、啟用 AWS IAM Identity Center和建立 管理使用者，以免將根使用者用於日常任務。

保護您的 AWS 帳戶根使用者

1. 選擇根使用者並輸入 AWS 帳戶 您的電子郵件地址，以帳戶擁有者AWS Management Console身分登入 。在下一頁中，輸入您的密碼。

   如需使用根使用者登入的說明，請參閱 AWS 登入 使用者指南中的以根使用者身分登入。
2. 若要在您的根使用者帳戶上啟用多重要素驗證 (MFA)。

   如需說明，請參閱《IAM 使用者指南》中的為您的 AWS 帳戶 根使用者 （主控台） 啟用虛擬 MFA 裝置。

建立具有管理存取權的使用者

1. 啟用 IAM Identity Center。

   如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的啟用 AWS IAM Identity Center。
2. 在 IAM Identity Center 中，將管理存取權授予使用者。

   如需使用 IAM Identity Center 目錄 做為身分來源的教學課程，請參閱AWS IAM Identity Center《 使用者指南》中的使用預設值設定使用者存取 IAM Identity Center 目錄。

以具有管理存取權的使用者身分登入

- 若要使用您的 IAM Identity Center 使用者簽署，請使用建立 IAM Identity Center 使用者時傳送至您電子郵件地址的簽署 URL。

  如需使用 IAM Identity Center 使用者登入的說明，請參閱AWS 登入 《 使用者指南》中的登入 AWS 存取入口網站。

指派存取權給其他使用者

1. 在 IAM Identity Center 中，建立一個許可集來遵循套用最低權限的最佳實務。

   如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的建立許可集。
2. 將使用者指派至群組，然後對該群組指派單一登入存取權。

   如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的新增群組。

設定 AWS Systems Manager

下列主題說明如何為 AWS Organizations 組織和單一 設定統一 AWS Systems Manager 主控台 AWS 帳戶。

主題

- 設定 Systems Manager 主控台存取
- 為組織設定 Systems Manager 統一主控台
- 針對單一帳戶和區域設定 Systems Manager 統一主控台
- 停用 Systems Manager 統一主控台

設定 Systems Manager 主控台存取

若要 AWS Systems Manager 在 中使用 AWS Management Console，您必須設定正確的許可。

如需如何建立 AWS Identity and Access Management 政策並將其連接至 IAM 身分的詳細資訊，請參閱《IAM 使用者指南》中的建立 IAM 政策

Systems Manager 加入政策

您可以建立類似下列範例所示的 IAM 政策，並將政策連接至 IAM 身分。此政策授予加入和設定 Systems Manager 所需的完整存取權。

許可詳細資訊

此政策包含以下許可。

- `ssm-quicksetup`：允許主體存取所有 AWS Systems Manager 快速設定 動作。
- `ssm`：允許主體存取 Systems Manager Automation 和 Resource Explorer。
- `organizations` – 允許主體在 中讀取組織的結構 AWS Organizations，並在委派管理員以組織身分加入 Systems Manager 時對其進行管理。
- `cloudformation`：允許主體管理其 Quick Setup 堆疊。
- `iam`：允許主體管理 Systems Manager 加入所需的 IAM 角色和政策。

```
{
   "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 主控台運算子政策

您可以建立類似下列範例所示的 IAM 政策，並將政策連接至 IAM 身分。此政策授予操作 Systems Manager 所需的完整存取權，以及允許 Systems Manager 執行 Automation 文件進行診斷和修復。

許可詳細資訊

此政策包含以下許可。

- `ssm`：允許主體存取所有 Systems Manager API。
- `ssm-quicksetup`：允許主體管理其 Quick Setup 組態。
- `ec2` – 允許 Systems Manager 判斷您的已啟用 AWS 區域 和 Amazon EC2 執行個體狀態。
- `cloudformation`：允許主體讀取其 Quick Setup 堆疊。
- `organizations` – 允許主體在 中讀取組織的結構 AWS Organizations，並在委派管理員以組織身分加入 Systems Manager 時對其進行管理。
- `s3`：允許主體列出並取得 Amazon S3 儲存貯體中的物件進行診斷，該診斷是在 Systems Manager 加入程序期間建立的。

- iam:PassRole：允許主體在執行自動化來診斷和修復非受管節點時，將要擔任的角色傳遞給 Systems Manager。
- iam:GetRole：允許主體在 Systems Manager 中作業時取得 Quick Setup 角色的特定角色資訊。

```
{
  "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 主控台運算子唯讀政策

您可以建立類似下列範例所示的 IAM 政策，並將政策連接至 IAM 身分。此政策會授予使用 Systems Manager 所需的唯讀存取權。

- `ssm`：允許主體存取 Systems Manager 唯讀 API。
- `ssm-quicksetup`：允許主體讀取其 Quick Setup 組態。
- `cloudformation`：允許主體讀取其 Quick Setup 堆疊。
- `iam:GetRole`：允許主體在使用 Systems Manager 時取得 Quick Setup 角色的特定角色資訊。
- `ec2:DescribeRegions`：允許 Systems Manager 判斷已啟用的 AWS 區域。
- `organizations` – 允許主體在加入 Systems Manager 做為組織 AWS Organizations 時讀取 中的組織結構。
- `s3`：允許主體列出及取得在 Systems Manager 加入程序期間建立的 Amazon S3 儲存貯體中的物件。

許可詳細資訊

此政策包含以下許可。

```
{
  "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}"
        }
      }
    }
  ]
}
```

為組織設定 Systems Manager 統一主控台

只需按幾下滑鼠，即可從 AWS Management Console 完成 Systems Manager 統一主控台體驗的設定程序。若要為 AWS Organizations 組織設定 Systems Manager，您必須能夠存取您組織的管理帳戶，以及您組織中的另一個帳戶，以做為委派管理員使用。只有在啟用或停用 Systems Manager 時，才需要管理帳戶的存取權。若要管理節點，您應使用委派管理員帳戶。

先決條件

管理整個組織的節點時，Systems Manager 會使用各種相依服務來設定和增強統一主控台的功能。因此，Systems Manager 必須啟用受信任的存取，以及註冊下列服務的委派管理員帳戶：

- AWS CloudFormation - 將 Systems Manager 所需的資源部署至您的帳戶。
- AWS 資源總管 - 搜尋和篩選您帳戶中的 EC2 執行個體。
- AWS Systems ManagerExplorer：監控帳戶中為 Systems Manager 部署的資源的運作狀態以及進行故障診斷。
- AWS Systems ManagerQuick Setup：將 Systems Manager 所需的 Quick Setup 組態部署至帳戶。

開始之前，請確定您尚未超過任何這些相依服務的委派管理員配額。否則，您無法註冊在啟用 Systems Manager 時所需的委派管理員帳戶。在為組織啟用 Systems Manager 時，組織中的每個帳戶都包含在內。目前沒有將帳戶從設定程序排除的佈建。啟用 Systems Manager 時，您可以選擇 AWS

區域 要包含的 。只能選取目前支援 Systems Manager 統一主控台的區域。若要進一步了解推出主控台體驗的區域，請參閱支援的 AWS 區域。

統一主控台資源

Systems Manager 統一主控台的設定程序會為您完成許多先決條件任務。根據您選擇設定的功能，這包括啟用預設主機管理組態，為您的節點提供必要的 IAM 許可等等。以下是 Systems Manager 為統一主控台建立的資源的詳細清單。根據您選擇設定的功能，可能不會建立某些資源。

AWS 資源總管 受管檢視

- `AWSManagedViewForSSM` – 允許 Systems Manager 存取由 Resource Explorer 為您的組織編製索引的資源資訊。這些受管檢視只能由 Systems Manager 更新或刪除。這表示如果您要刪除受管檢視或關閉 Resource Explorer，您必須停用統一主控台。如需停用統一主控台的詳細資訊，請參閱 停用 Systems Manager 統一主控台。如需受管檢視的詳細資訊，請參閱 Resource Explorer 使用者指南中的AWS 受管檢視。

> **Note**
>
> 如果您已在與主區域不同的區域中建立適用於 Resource Explorer 的彙總工具索引，則 Systems Manager 會降級目前的索引。然後，Systems Manager 會將主區域中的本機索引提高為新的彙總工具索引。在此期間，系統僅會顯示主區域的節點。此程序最長需要 24 小時才能完成。

IAM 角色

- `RoleForOnboardingAutomation`：允許 Systems Manager 在設定程序期間管理資源。如需有關政策的詳細資訊，請參閱 AWSQuickSetupSSMManageResourcesExecutionPolicy。
- `RoleForLifecycleManagement`：允許 Lambda 管理設定程序所建立的資源生命週期。如需有關政策的詳細資訊，請參閱 AWSQuickSetupSSMLifecycleManagementExecutionPolicy。
- `RoleForAutomation`：Systems Manager Automation 在執行執行手冊時所擔任的服務角色。如需詳細資訊，請參閱使用主控台建立用於自動化的服務角色。
- `AWSSSMDiagnosisAdminRole`：管理角色，用來啟動可使用診斷執行手冊的自動化。如需有關政策的詳細資訊，請參閱 AWS-SSM-DiagnosisAutomation-AdministrationRolePolicy、AWS-SSM-Automation-DiagnosisBucketPolicy 和 AWS-SSM-DiagnosisAutomation-OperationalAccountAdministrationRolePolicy。

- AWSSSMDiagnosisExecutionRole：診斷執行手冊的自動化執行角色。如需有關政策的詳細資訊，請參閱 AWS-SSM-DiagnosisAutomation-ExecutionRolePolicy 和 AWS-SSM-Automation-DiagnosisBucketPolicy。
- AWSSSMRemediationAdminRole：管理角色，用來啟動可使用修復執行手冊的自動化。如需有關政策的詳細資訊，請參閱 AWS-SSM-RemediationAutomation-AdministrationRolePolicy、AWS-SSM-Automation-DiagnosisBucketPolicy 和 AWS-SSM-RemediationAutomation-OperationalAccountAdministrationRolePolicy。
- AWSSSMRemediationExecutionRole：修復執行手冊的自動化執行角色。如需有關政策的詳細資訊，請參閱 AWS-SSM-RemediationAutomation-ExecutionRolePolicy 和 AWS-SSM-Automation-DiagnosisBucketPolicy。
- ManagedInstanceCrossAccountManagementRole：允許 Systems Manager 跨帳戶收集受管節點資訊。

State Manager 關聯

- EnableDHMCAssociation：每天執行一次，確保預設主機管理組態已啟用。
- SystemAssociationForEnablingExplorer：每天執行一次，確保 Explorer 已啟用。Explorer 會用於同步受管節點中的資料。
- EnableAREXAssociation – 每天執行 ，並確保 AWS 資源總管 已啟用。Resource Explorer 會用於判斷組織中的哪些 Amazon EC2 執行個體不是由 Systems Manager 管理。
- SSMAgentUpdateAssociation：每 14 天執行一次，確保最新的 SSM Agent 可用版本已安裝在受管節點。
- SystemAssociationForInventoryCollection：每 12 小時執行一次，從受管節點收集庫存資料。

S3 儲存貯體

- DiagnosisBucket：存放從診斷執行手冊執行所收集的資料。

Lambda 函數

- SSMLifecycleOperatorLambda：允許主體存取所有 AWS Systems Manager 快速設定 動作。
- SSMLifecycleResource：自訂資源，協助管理設定程序所建立的資源的生命週期。

此外，在設定程序完成後，您可以選取診斷和修復節點任務，自動將修正套用至節點 (這些節點未報告為由 Systems Manager 管理)。這可能包括識別問題，例如 Systems Manager 端點的網路連線問題。如需詳細資訊，請參閱診斷和修復。

設定統一主控台

設定組織的 Systems Manager

1. 登入組織的管理帳戶。
2. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
3. 輸入要註冊為委派管理員的帳戶的 ID。
4. 委派管理員帳戶成功註冊後，請登入剛剛註冊的委派管理員帳戶，再返回 Systems Manager 主控台完成 Systems Manager 的設定。
5. 選取啟用 Systems Manager。
6. 在主區域區段中，您可以決定希望 Systems Manager 彙總節點資料的區域。根據預設，Systems Manager 會選取您目前使用的區域。若要選擇不同的主區域，請在設定 Systems Manager 之前，將主控台變更為您要使用的區域。節點資料會複寫到組織的帳戶和區域，並存放在主區域中。設定 Systems Manager 後，就無法變更所選擇的區域。若要使用不同的區域作為組織的主區域，您必須停用統一主控台，再次完成設定程序。如果組織使用的是 IAM Identity Center，則必須選取您將 IAM Identity Center 設為主區域的同一個區域。
7. 在區域區段中，選取要啟用 Systems Manager 的區域。
8. 在功能組態區段中，選擇您要為組態啟用的選項：

   啟用預設主機管理組態 (DHMC)

   允許 Systems Manager 設定 DHMC。此功能可讓 Systems Manager 使用 IAM 角色，以確保帳戶和區域中的所有 Amazon EC2 執行個體都具有由 Systems Manager 管理的必要許可。您也可以指定漂移修復的頻率。每當使用者對服務或功能進行任何變更，而該變更與透過您的組態所做的選擇衝突時，就會發生組態偏離。Systems Manager 會檢查組態偏離，並嘗試根據您指定的頻率進行修復。您必須指定介於 1 到 31 天之間的值。如果您已在區域中設定 DHMC，Systems Manager 不會變更您先前選取的 IAM 角色。如需 DHMC 的詳細資訊，請參閱 使用預設主機管理組態來自動管理 EC2 執行個體。

   DHMC 可讓您管理 Amazon EC2 執行個體，而無需手動建立 AWS Identity and Access Management (IAM) 執行個體描述檔。建議您選擇此選項，以確保您的 EC2 執行個體具有由 Systems Manager 管理的必要許可。

啟用庫存中繼資料收集

可讓 Systems Manager 從節點設定下列中繼資料類型的集合：

- AWS 元件 – EC2 驅動程式、代理程式、版本等。
- 應用程式 – 應用程式名稱、發佈者、版本等。
- 節點詳細資訊 – 系統名稱、作業系統 (OS) 名稱、作業系統版本、上次啟動時間、DNS、網域、工作團隊、作業系統架構等。
- 網路組態 – IP 地址、MAC 地址、DNS、閘道、子網路遮罩等。
- 服務 – 名稱、顯示名稱、狀態、相依服務、服務類型、啟動類型等 (僅限 Windows Server 節點)。
- Windows 角色 – 名稱、顯示名稱、路徑、功能類型、安裝狀態等 (僅限 Windows Server 受管節點)。
- Windows 更新 – Hotfix ID、安裝人員、安裝日期等 (僅限 Windows Server 節點)。

指定收集庫存的頻率。您必須指定介於 1 到 744 小時之間的值。如需 中的工具庫存的詳細資訊 AWS Systems Manager，請參閱 AWS Systems Manager庫存。

啟用自動 Systems Manager (SSM) 代理程式更新

可讓 Systems Manager 依您指定的頻率檢查新版本的代理程式。頻率的值必須介於 1 到 31 天之間。如果有新版本，則 Systems Manager 會自動將受管節點上的代理程式更新為最新發行版本。Systems Manager 不會在尚未存在的執行個體上安裝代理程式。如需有關哪些 AMIs 已預先安裝 SSM Agent 的資訊，請參閱 尋找預先安裝了 SSM Agent的 AMIs。

建議您選擇此選項，以確保節點始終執行最新版本的 SSM Agent。如需 SSM Agent 的詳細資訊，包括如何手動安裝代理程式的資訊。請參閱 使用 SSM Agent。

9. 選擇提交。

如果組織規模較大，設定 Systems Manager 統一主控台體驗可能需要較長的時間。

針對單一帳戶和區域設定 Systems Manager 統一主控台

若要為單一 AWS 區域 設定 Systems Manager 統一主控台體驗 AWS 帳戶 ，您不需要使用 Organizations 或註冊委派管理員帳戶。Systems Manager 主控台體驗的設定程序會為您完成許多先決條件任務。根據您選擇設定的功能，這包括啟用預設主機管理組態，為您的節點提供必要的 IAM 許可等等。以下是 Systems Manager 為統一主控台建立的資源的詳細清單。

統一主控台資源

根據您選擇設定的功能，可能不會建立某些資源。

IAM 角色

- `RoleForOnboardingAutomation`：允許 Systems Manager 在設定程序期間管理資源。如需有關政策的詳細資訊，請參閱 AWSQuickSetupSSMManageResourcesExecutionPolicy。
- `RoleForLifecycleManagement`：允許 Lambda 管理設定程序所建立的資源生命週期。如需有關政策的詳細資訊，請參閱 AWSQuickSetupSSMLifecycleManagementExecutionPolicy。
- `RoleForAutomation`：Systems Manager Automation 在執行執行手冊時所擔任的服務角色。如需詳細資訊，請參閱使用主控台建立用於自動化的服務角色。
- `AWSSSMDiagnosisAdminRole`：診斷執行手冊的自動化執行角色。如需有關政策的詳細資訊，請參閱 AWS-SSM-DiagnosisAutomation-AdministrationRolePolicy、AWS-SSM-Automation-DiagnosisBucketPolicy 和 AWS-SSM-DiagnosisAutomation-OperationalAccountAdministrationRolePolicy。
- `AWSSSMRemediationAdminRole`：修復執行手冊的自動化執行角色。如需有關政策的詳細資訊，請參閱 AWS-SSM-RemediationAutomation-AdministrationRolePolicy、AWS-SSM-Automation-DiagnosisBucketPolicy 和 AWS-SSM-RemediationAutomation-OperationalAccountAdministrationRolePolicy。
- `ManagedInstanceCrossAccountManagementRole`：允許 Systems Manager 跨帳戶收集受管節點資訊。

State Manager 關聯

- `EnableDHMCAssociation`：每天執行一次，確保預設主機管理組態已啟用。
- `SystemAssociationForEnablingExplorer`：每天執行一次，確保 Explorer 已啟用。Explorer 會用於同步受管節點中的資料。
- `EnableAREXAssociation` – 每天執行 ，並確保 AWS 資源總管 已啟用。Resource Explorer 會用於判斷組織中的哪些 Amazon EC2 執行個體不是由 Systems Manager 管理。
- `SSMAgentUpdateAssociation`：每 14 天執行一次，確保最新的 SSM Agent 可用版本已安裝在受管節點。
- `SystemAssociationForInventoryCollection`：每 12 小時執行一次，從受管節點收集庫存資料。

S3 儲存貯體

- `DiagnosisBucket`：存放從診斷執行手冊執行所收集的資料。

Lambda 函數

- `SSMLifecycleOperatorLambda`：允許主體存取所有 AWS Systems Manager 快速設定 動作。
- `SSMLifecycleResource`：自訂資源，協助管理設定程序所建立的資源的生命週期。

此外，在設定程序完成後，您可以選取診斷和修復節點任務，自動將修正套用至節點 (這些節點未報告為由 Systems Manager 管理)。這可能包括識別問題，例如 Systems Manager 端點的網路連線問題。

設定統一主控台

針對單一帳戶和區域設定 Systems Manager

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 選取啟用 Systems Manager。
3. 在功能組態區段中，選擇您要為組態啟用的選項：

   啟用預設主機管理組態 (DHMC)

   允許 Systems Manager 設定 DHMC。此功能可讓 Systems Manager 使用 IAM 角色，以確保帳戶和區域中的所有 Amazon EC2 執行個體都具有由 Systems Manager 管理的必要許可。您也可以指定漂移修復的頻率。每當使用者對服務或功能進行任何變更，而該變更與透過您的組態所做的選擇衝突時，就會發生組態偏離。Systems Manager 會檢查組態偏離，並嘗試根據您指定的頻率進行修復。您必須指定介於 1 到 31 天之間的值。如果您已在區域中設定 DHMC，Systems Manager 不會變更您先前選取的 IAM 角色。如需 DHMC 的詳細資訊，請參閱 使用預設主機管理組態來自動管理 EC2 執行個體。

   DHMC 可讓您管理 Amazon EC2 執行個體，而無需手動建立 AWS Identity and Access Management (IAM) 執行個體描述檔。建議您選擇此選項，以確保您的 EC2 執行個體具有由 Systems Manager 管理的必要許可。

啟用庫存中繼資料收集

可讓 Systems Manager 從節點設定下列中繼資料類型的集合：

- AWS 元件 – EC2 驅動程式、代理程式、版本等。
- 應用程式 – 應用程式名稱、發佈者、版本等。
- 節點詳細資訊 – 系統名稱、作業系統 (OS) 名稱、作業系統版本、上次啟動時間、DNS、網域、工作團隊、作業系統架構等。
- 網路組態 – IP 地址、MAC 地址、DNS、閘道、子網路遮罩等。
- 服務 – 名稱、顯示名稱、狀態、相依服務、服務類型、啟動類型等 (僅限 Windows Server 節點)。
- Windows 角色 – 名稱、顯示名稱、路徑、功能類型、安裝狀態等 (僅限 Windows Server 受管節點)。
- Windows 更新 – Hotfix ID、安裝人員、安裝日期等 (僅限 Windows Server 節點)。

指定收集庫存的頻率。您必須指定介於 1 到 744 小時之間的值。如需 中的工具庫存的詳細資訊 AWS Systems Manager，請參閱 AWS Systems Manager庫存。

啟用自動 Systems Manager (SSM) 代理程式更新

可讓 Systems Manager 依您指定的頻率檢查新版本的代理程式。頻率的值必須介於 1 到 31 天之間。如果有新版本，則 Systems Manager 會自動將受管節點上的代理程式更新為最新發行版本。Systems Manager 不會在尚未存在的執行個體上安裝代理程式。如需有關哪些 AMIs 已預先安裝 SSM Agent 的資訊，請參閱 尋找預先安裝了 SSM Agent的 AMIs。

建議您選擇此選項，以確保節點始終執行最新版本的 SSM Agent。如需 SSM Agent 的詳細資訊，包括如何手動安裝代理程式的資訊。請參閱 使用 SSM Agent。

4. 選擇提交。

停用 Systems Manager 統一主控台

若要停用組織的 Systems Manager 統一主控台，您需要存取註冊為 Systems Manager 委派管理員的帳戶。登入組織的委派管理員帳戶後，您就可以在統一主控台的設定區段中停用組織的設定。當您停用組織的統一主控台設定時，Systems Manager 會在設定程序期間刪除建立的資源，包括 Resource Explorer 受管檢視。停用組織的設定不會撤銷受信任的存取，也不會取消註冊相依服務的委派管理員帳戶。下列程序說明如何停用統一主控台的設定。

停用 Systems Manager 統一主控台的設定

1. 開啟 AWS Systems Manager 主控台，網址為 https：//https://console.aws.amazon.com/systems-manager/。
2. 在導覽窗格中選取設定。
3. 選取停用。您必須確認是否要停用組織的設定。此動作會刪除為統一主控台建立的資源，且無法復原。

如果您無法存取 Systems Manager 的委派管理員帳戶，而且想要停用統一主控台的設定，您也可以從組織的管理帳戶執行此操作。使用 AWS CLI 或 SDK，呼叫 `DeleteConfigurationManager` API 操作並傳遞您帳戶中組織設定`ManagerArn`的值。用於設定統一主控台的管理員 ARN 格式如下：

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

> **Note**
>
> 如果您不知道 *`configuration-manager-id`* 的值，請呼叫 `ListConfigurationManagers` API 動作並使用 `AWSQuickSetupType-SSM`類型篩選結果。

使用 執行節點管理任務 AWS Systems Manager

下列主題說明如何使用 AWS Organizations 組織和單一的統一 AWS Systems Manager 主控台來完成常見節點任務 AWS 帳戶。

主題

- 檢閱節點洞察
- 探索節點
- 使用 Systems Manager Just-in-time節點存取
- 診斷和修復
- 調整 Systems Manager 設定

檢閱節點洞察

您可以使用統一 Systems Manager 主控台，深入了解組織或帳戶中受管節點和未受管 EC2 執行個體的整體狀態。

Systems Manager 可為您的受管節點和尚未由 Systems Manager 管理的 EC2 執行個體提供視覺化概觀。(受管節點是設定為在混合多雲端環境中搭配 Systems Manager 使用的任何機器。如需有關支援機器類型的資訊，請參閱混合多雲端環境中支援的機器類型。)

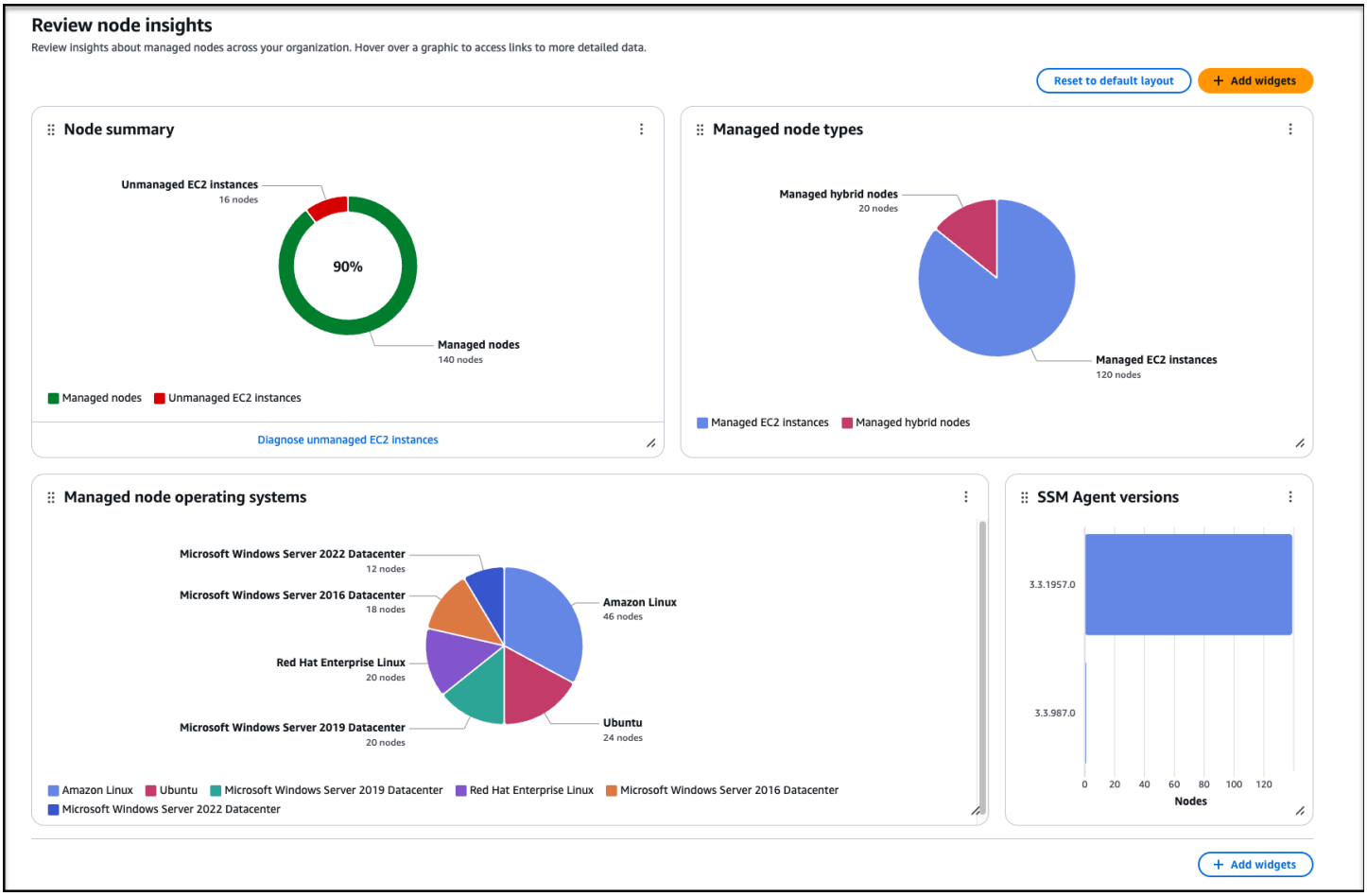
此概觀透過稱為小工具的個別報告方塊提供，具有互動式圓餅圖和其他圖形。

開始之前

若要檢閱節點洞見，您必須先將組織或帳戶加入統一的 Systems Manager 主控台。如需詳細資訊，請參閱設定 AWS Systems Manager。

加入後，開啟 Systems Manager 主控台，然後選擇檢閱節點洞察。

下列影像顯示一些個別的報告方塊，稱為小組件，可在檢閱節點洞察頁面中使用。



顯示介面支援的小工具可提供下列資訊。

節點摘要

指出組織或帳戶中有多少個 EC2 執行個體目前不是受管節點，以及組織或帳戶機群中有多少個受管節點。

什麼是未受管執行個體？

當您停止受管 EC2 執行個體時，它會在 Systems Manager 主控台中報告為「未受管」。這是預期的行為，因為 SSM Agent沒有與 服務的作用中連線。

> **Note**
>
> 這與 如何將執行個體 AWS Config 定義為未受管不同。如果執行個體目前停止， 會 AWS Config 報告執行個體與 Systems Manager 服務SSM Agent上一次在 之間進行「heartbeat」連線的狀態。

當執行個體重新啟動時，它會自動重新連線至 Systems Manager 服務，並在五分鐘內將其在統一主控台中的狀態還原至「受管」。不需要手動介入，且執行個體的所有 Systems Manager 組態都會在停止/啟動週期期間保留。

不過，如果執行個體在啟動幾分鐘後仍未回報為「受管」，則執行個體可能無法正確設定 Systems Manager 管理。在此情況下，建議您執行診斷，以識別執行個體為何保持未受管狀態。如需詳細資訊，請參閱在 Systems Manager 中診斷和修復未受管的 Amazon EC2 執行個體。

如果診斷掃描無法判斷問題，請參閱下列主題，以確認符合 SSM Agent、 AWS Identity and Access Management (IAM) 角色和 Systems Manager 先決條件的要求：

- SSM Agent 疑難排解
- 設定 Systems Manager 所需的執行個體許可
- 疑難排解受管節點的可用性

受管節點類型

指出機群中有多少受管節點是 EC2 執行個體，以及有多少其他伺服器類型，包括您自己的內部部署（內部部署伺服器）、 AWS IoT Greengrass 核心裝置 AWS IoT 和非AWS 邊緣裝置，以及虛擬機器 (VMs)，包括其他雲端環境中VMs。將滑鼠暫留在節點類型圖形上，即可存取探索節點頁面中更多詳細資訊的連結。

如需混合多雲端環境 AWS 支援的詳細資訊，請參閱AWS 混合多雲端解決方案。

SSM Agent 版本

提供有關在機群中安裝 AWS Systems Manager 代理程式 (SSM Agent) 的資訊。 SSM Agent是在受管節點上執行的 Amazon 軟體。 SSM Agent 可讓 Systems Manager 更新、管理和設定這些資源。代理程式會在 中處理來自 Systems Manager 服務的請求 AWS 雲端，然後依照請求中的指定執行它們。

對於機群中的受管節點，此小工具會報告機群中從最新到最舊的 SSM Agent版本。將滑鼠暫留在 SSM Agent 版本圖形上，即可存取探索節點頁面中更多詳細資訊的連結。

如需 SSM Agent 的相關資訊，請參閱 使用 SSM Agent。

受管節點作業系統

提供機群中受管節點上每個作業系統的百分比明細。將滑鼠暫留在依照作業系統分組的受管節點圖形上，即可存取探索節點頁面中更多詳細資訊的連結。

在顯示介面上使用拖放功能來移除和新增小工具，即可在檢閱節點洞察頁面上自訂小工具配置。

參照下列主題中的資訊，協助自己使用 Systems Manager 節點洞察小工具。

主題

- 在檢閱節點洞察頁面中新增或移除小工具
- 在檢閱節點洞察頁面中重新排列小工具

在檢閱節點洞察頁面中新增或移除小工具

透過新增和移除小工具，即可在 Systems Manager 檢閱節點洞察頁面中自訂配置。

> **Note**
>
> 依預設，頁面會顯示所有可用的小工具。

在檢閱節點洞察頁面中新增或移除小工具

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在左側導覽列中，選擇檢閱節點洞察。
3. 若要從顯示介面中移除小工具，請執行下列動作：
   a. 選擇小工具的更多選項選單
      (⋮)。
   b. 選擇 Remove widget (移除 Widget)。
4. 若要將小工具新增至顯示介面，請執行下列動作：
   a. 選擇新增小工具。
   b. 在新增小工具窗格中，按住小工具的拖曳控點
      (⠿)
      將其新增至顯示介面。
   c. 將小工具拖放到主窗格中。

在檢閱節點洞察頁面中重新排列小工具

您可以透過重新排列小工具，在檢閱節點洞察頁面中自訂配置。

在檢閱節點洞察頁面中重新排列小工具

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇檢閱節點洞察。
3. 若要自訂小工具配置，請選擇要移動的小工具。按住小工具的拖曳控點 (⠿)，將其拖曳到新位置。
4. 對您要重新定位的每個 Widget 重複此程序。

如果決定不要新的配置，請選擇重設為預設配置，將所有小組件移回原始位置。

探索節點

您可以使用 Systems Manager 中的探索節點頁面，根據您在篩選條件中指定的條件檢閱組織或帳戶中受管節點的詳細資訊。您也可以使用 Systems Manager 與 AWS 生成式 AI 解決方案 Amazon Q Developer (Amazon Q) 的整合，使用文字提示進行搜尋。

開始之前

若要使用探索節點功能，您必須先將組織或帳戶加入統一的 Systems Manager 主控台。如需詳細資訊，請參閱為組織設定 Systems Manager 統一主控台。

在加入後，請開啟 Systems Manager 主控台，並選擇探索節點。

> **Note**
>
> 如果您已在與主區域不同的區域中建立適用於 Resource Explorer 的彙總工具索引，則 Systems Manager 會降級目前的索引。然後，Systems Manager 會將主區域中的本機索引提高為新的彙總工具索引。在此期間，系統僅會顯示主區域的節點。此程序最長需要 24 小時才能完成。

主題

- 使用主控台篩選條件探索節點
- 在 Amazon Q 中使用文字提示探索節點

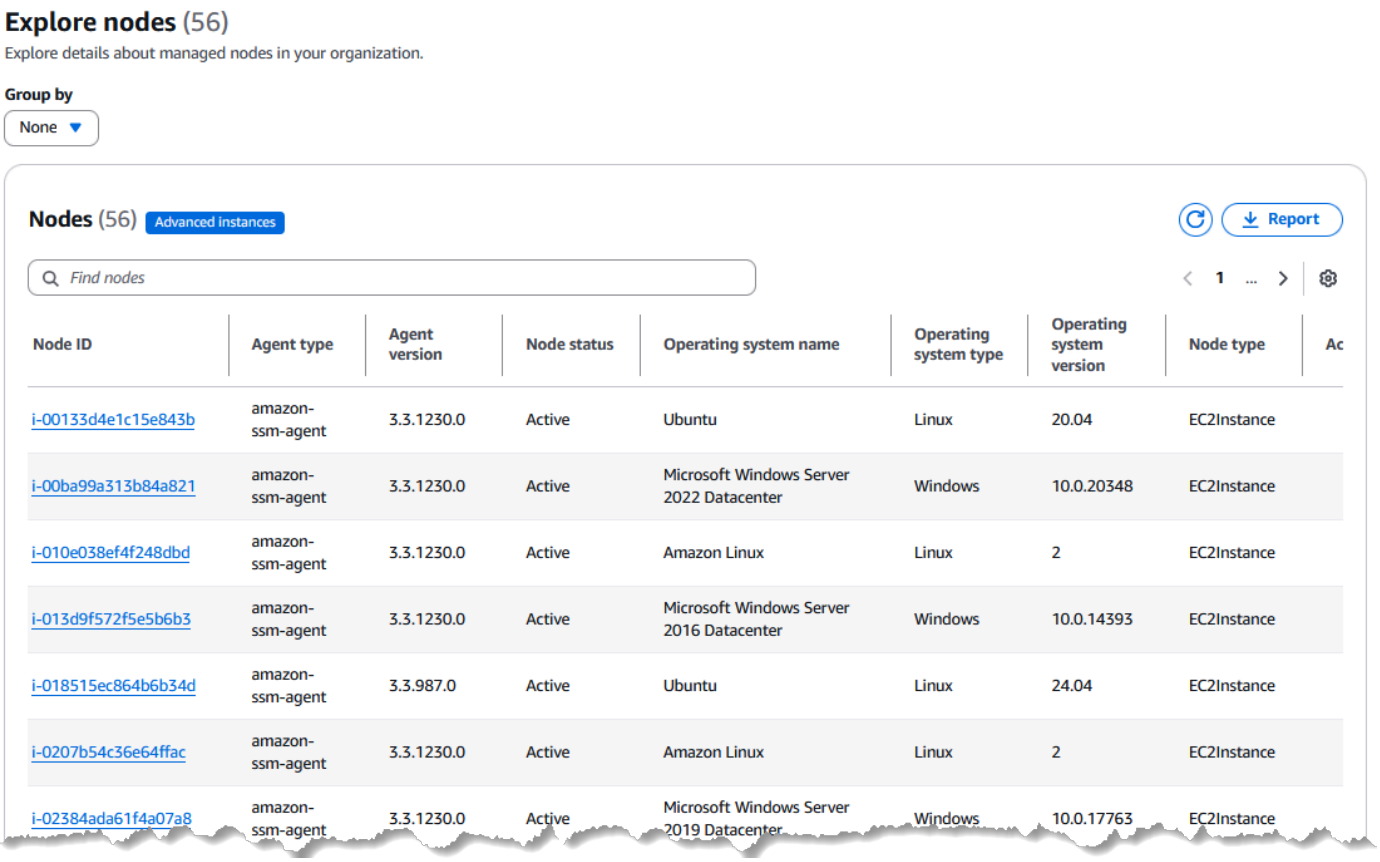
- 檢視個別節點詳細資訊並對節點採取動作
- 下載或匯出受管節點報告
- 管理節點報告內容和外觀

使用主控台篩選條件探索節點

在 Systems Manager 主控台中，您可以根據下方檢視對受管節點進行分組：

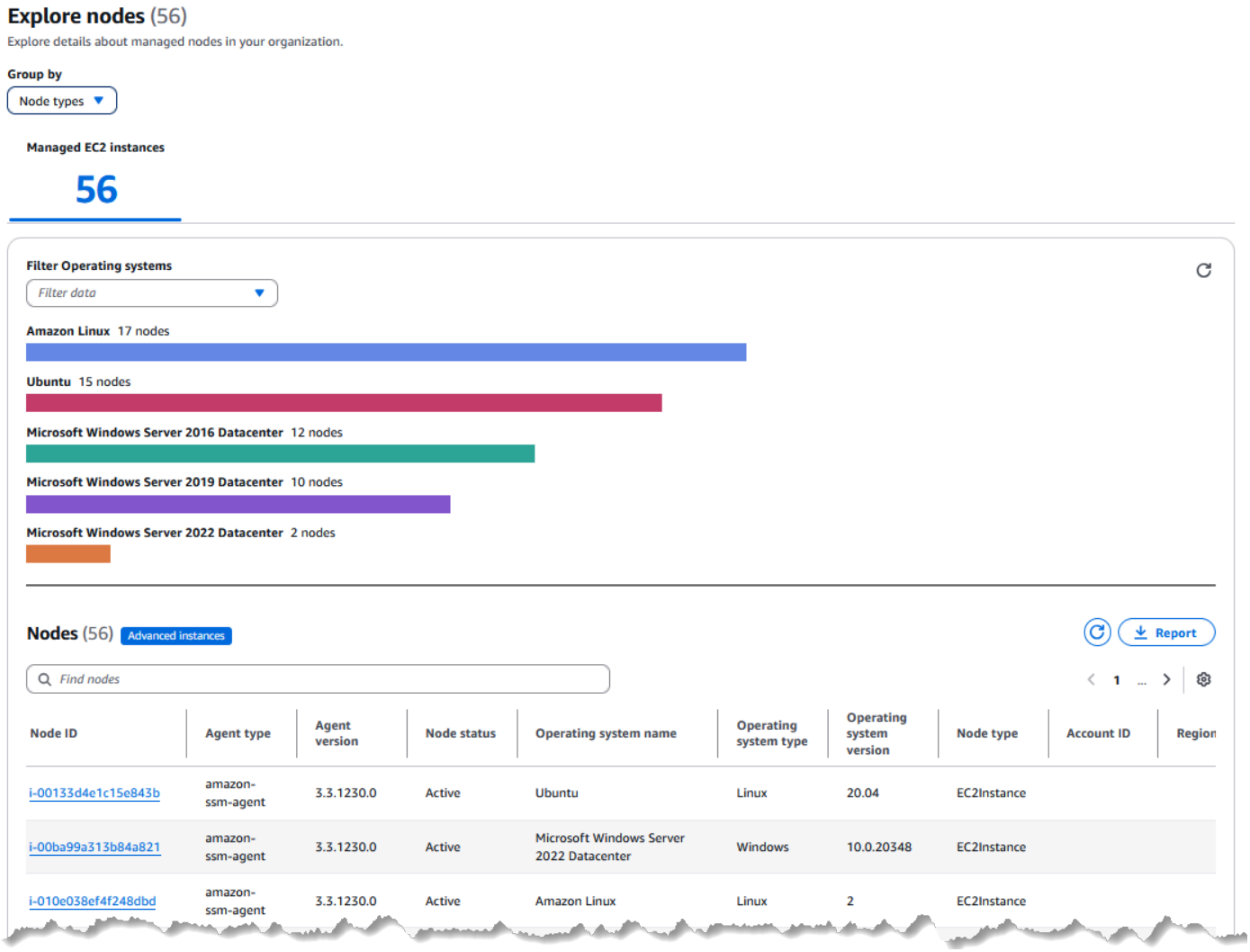
All nodes (No filter)

列出組織或帳戶中的所有受管節點。



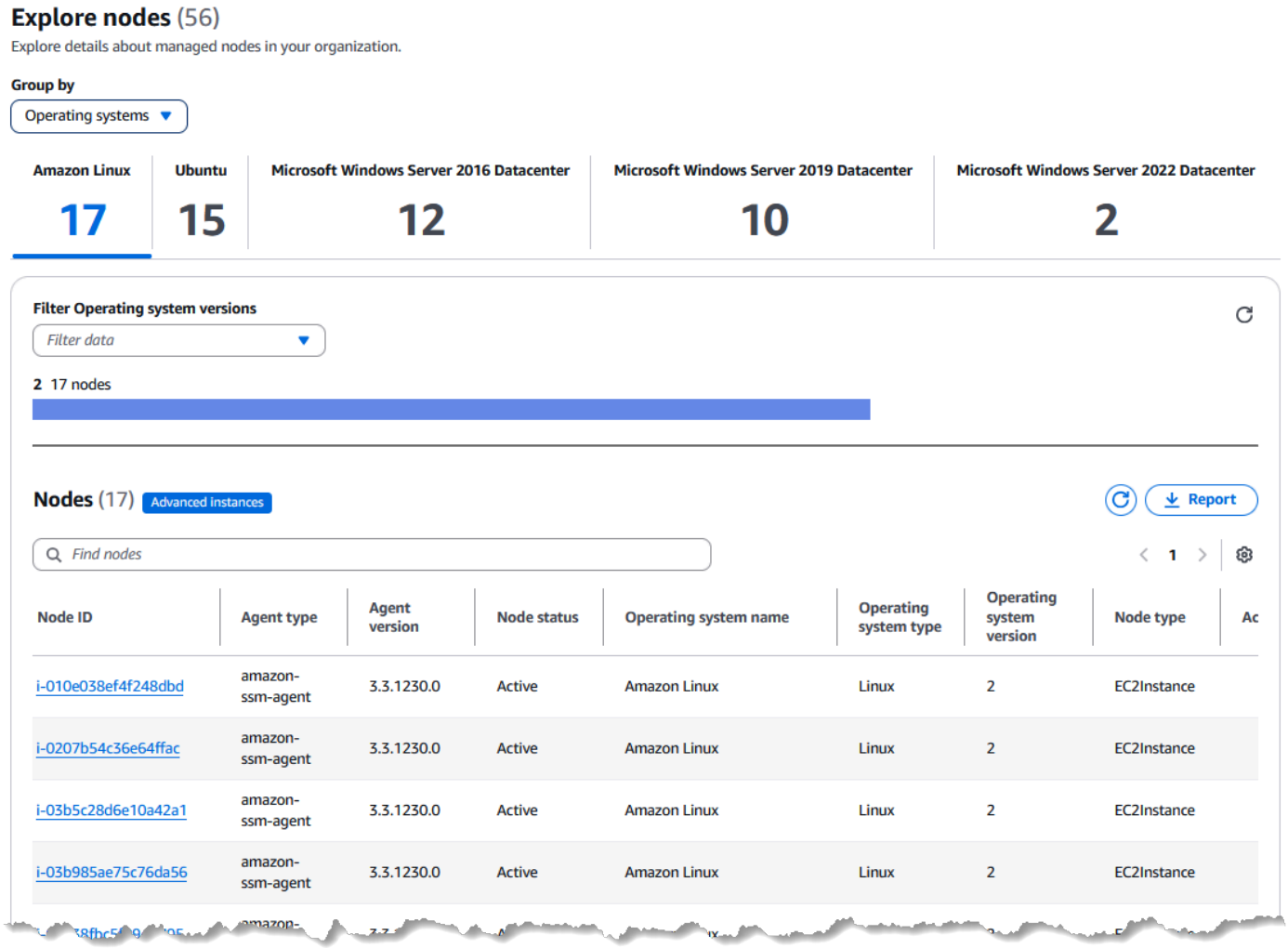
Node types

提供標籤，分別檢視 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體和其他機器類型的資料，包括您內部部署 （內部部署伺服器） 上的伺服器、 AWS IoT Greengrass 核心裝置 AWS IoT 和非AWS 邊緣裝置，以及虛擬機器 (VMs)，包括其他雲端環境中VMs。



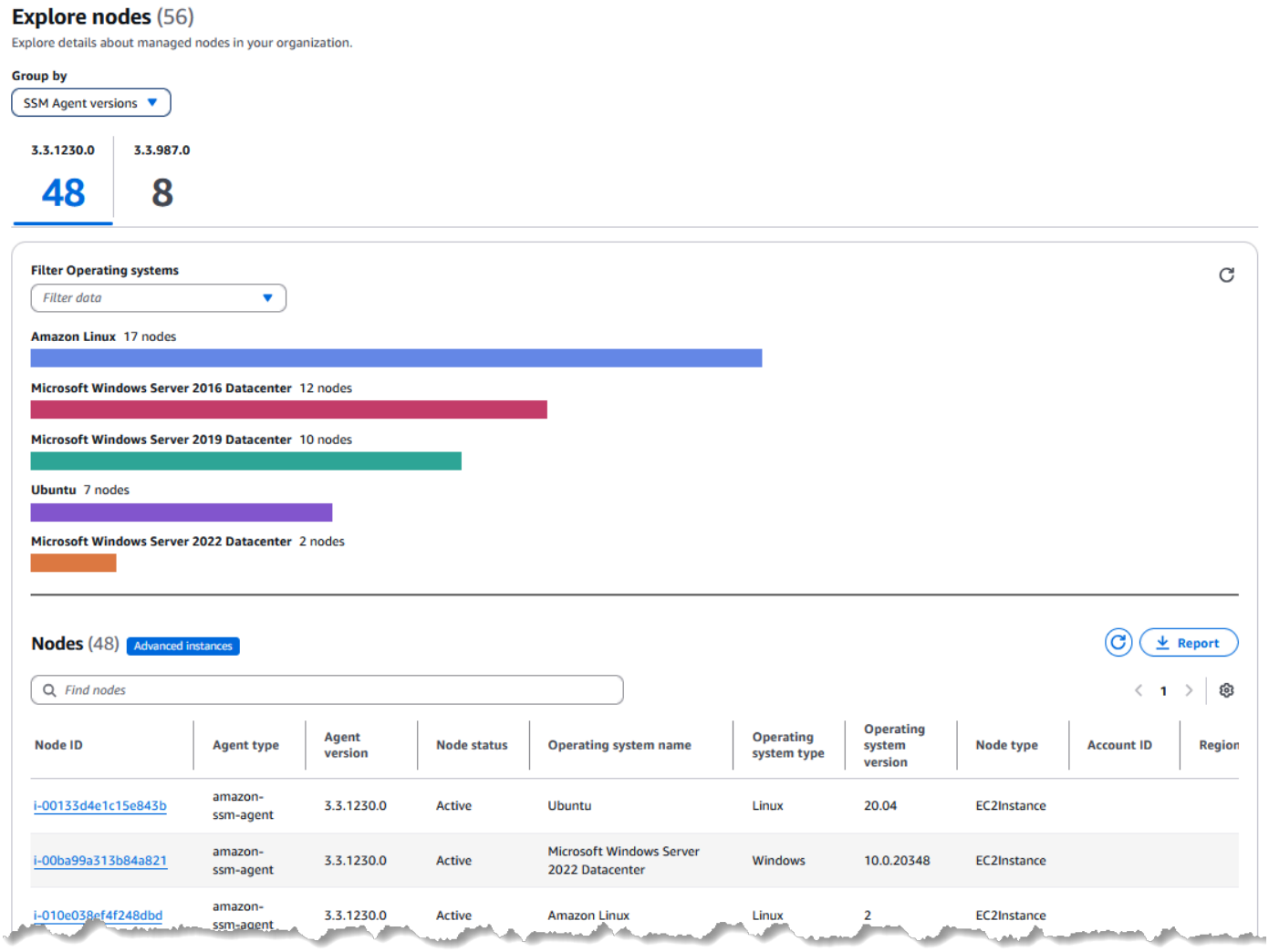
Operating systems

提供組織或帳戶中每個作業系統類型的索引標籤，例如 Amazon Linux 和 Microsoft Windows Server 2022 Datacenter。在每個索引標籤上，您可以僅選取 Amazon Linux 2 和 Amazon Linux 2023 等特定版本的作業系統，進一步篩選清單。



SSM Agent versions

為機群中受管節點上安裝的每個 SSM Agent 版本提供索引標籤。在每個索引標籤上，您可以僅選取 Amazon Linux 和 Microsoft Windows Server 2022 Datacenter 等特定作業系統，進一步篩選清單。



此外，對於每個檢視，您可以選擇僅檢視節點狀態、 AWS 帳戶 ID、組織單位 ID 等特定屬性的節點，進一步精簡報告的節點清單。

您可以選擇在探索節點頁面顯示哪些可用的欄位來自訂報告顯示。您也可以下載 CSV 或 JSON 格式的報告，或以 CSV 格式將報告匯出至 Amazon S3。

主題

- 選擇受管節點摘要的篩選條件檢視

選擇受管節點摘要的篩選條件檢視

您可以在 Systems Manager 中的探索節點頁面上，根據多種可用的篩選條件檢視來檢視機群的相關彙總資料。

選擇受管節點摘要的篩選條件檢視

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇探索節點。
3. 對於篩選條件檢視，選取其中一個篩選條件選項，並選擇性地進一步精簡報告：

   - 受管節點：您可以在搜尋 ( ) 方塊中選取屬性和分隔符號，例如 `Node type = Managed EC2 instances`。
   - 作業系統：在篩選作業系統版本清單中，您可以選取作業系統版本號。在搜尋 ( ) 方塊中，您可以選取屬性和分隔符號，例如 `Node type = Managed EC2 instances`。
   - SSM Agent 版本：在篩選作業系統清單中，您可以選取作業系統名稱。在搜尋 ( ) 方塊中，您可以選取屬性和分隔符號，例如 `Node type = Managed EC2 instances`。
   - 節點類型：在篩選作業系統清單中，您可以選取作業系統名稱。在搜尋 ( ) 方塊中，您可以選取屬性和分隔符號，例如 `Node type = Managed EC2 instances`。

在選擇性篩選清單之後，您可以在節點 ID 欄位中選擇特定受管節點的 ID，以檢視該節點的相關詳細資訊。從該詳細檢視中，您可以對節點執行許多動作。

在 Amazon Q 中使用文字提示探索節點

藉助 Systems Manager 與 Amazon Q Developer 的整合，您可以使用文字提示檢視由生成式 AI 建立的受管節點相關資訊。

Amazon Q Developer 是採用 AI 技術的生成式對話助理，可協助您了解、建置、延伸和操作 AWS 應用程式。為了加速您的建置 AWS，支援 Amazon Q 的模型會擴增為高品質的 AWS 內容，以產生更完整、可行且參考的答案。如需詳細資訊，請參閱《Amazon Q Developer 使用者指南》中的 What is Amazon Q Developer? 部分。

Systems Manager 和 Amazon Q 之間的整合可讓您快速取得跨多個 和 區域的大型分散式環境的可見 AWS 帳戶 性和控制。您可以使用自然語言查詢快速搜尋節點資料，然後識別問題並以更快的速度採取行動。

當您詢問有關受管節點或受管執行個體的自然語言問題時，Amazon Q 會使用 Systems Manager `ListNodes` 動作，並根據文字輸入建立篩選條件來擷取結果。

例如，假設您對 Amazon Q 提供下方提示：

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

Amazon Q 會判斷要包含在請求中的篩選條件，然後執行如下所示的查詢：

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

之後，Amazon Q 會產生帳戶中關於 Red Hat Enterprise Linux 執行個體的報告，列出執行個體的數量、ID 以及區域等資訊。

您可以檢視每個執行個體詳細資訊的 JSON 摘要，也可以開啟連結，在 Systems Manager 探索節點頁面中檢視 EC2 執行個體或受管節點的完整清單。探索節點會顯示符合提示所包含篩選條件的結果。您可以在該處變更或精簡請求的篩選條件，如在探索節點中所述。

主題

- 學習制定有效的提示向 Amazon Q 詢問有關機群的問題
- 使用 Amazon Q 探索受管節點

學習制定有效的提示向 Amazon Q 詢問有關機群的問題

您提供給 Amazon Q 品質越好的問題或提示，Amazon Q 提供的結果就越好。

查詢提示的提示

在向 Amazon Q 查詢有關機群的問題時，請記住下列提示：

1. 為了協助提高結果的準確性，請在提示中使用「受管節點」和「受管執行個體」等字詞，而不是只使用「節點」和「執行個體」。
2. 若要在屬於組織一部分的多個帳戶中查詢結果，如 中所設定 AWS Organizations，您必須登入指定主區域中的委派管理員帳戶。
3. 在委派管理員帳戶中，使用字詞來協助 Amazon Q 了解您詢問的是有關整個組織的節點和執行個體，具體使用「在我的組織中」或「在我的帳戶 123456789012 中」等字詞。

主題

- Amazon Q 的範例問題
- 提示支援的作業系統名稱和版本

Amazon Q 的範例問題

在下表中，我們提供數個範例問題，示範了您可以建立 Amazon Q 查詢來產生較佳結果的一些方式。

我們也提供 Amazon Q 會在執行 ListNodes 命令時套用的篩選條件範例，而這些篩選條件是根據提示內容產生的。

自然語言問題範例	Amazon Q 套用的篩選條件
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`

自然語言問題範例	Amazon Q 套用的篩選條件
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`

自然語言問題範例	Amazon Q 套用的篩選條件
List all managed instances that are not running a Linux operating system.	`PlatformType != Linux`

提示支援的作業系統名稱和版本

當您向 Amazon Q 詢問帳戶中的受管節點時，提供如 Systems Manager 中標示的作業系統名稱會很有幫助。您也可以提供版本編號，進一步縮小結果範圍。例如，如下表所示，您可以要求具體關於 **macOS 14.5**、**Microsoft Windows Server 2019 Datacenter** 和 **AlmaLinux 9.2 through 9.4** 等的結果。

這些清單並非詳盡無遺，僅作為範例提供。

macOS

平台名稱	版本編號
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

推出	版本編號
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

推出	版本編號
Microsoft Windows Server 2016 Datacenter	無
Microsoft Windows Server 2016 Standard	10.0.14393
Microsoft Windows Server 2019 Datacenter	無
Microsoft Windows Server 2019 Standard	無
Microsoft Windows Server 2022 Datacenter	無
Microsoft Windows Server 2022 Standard	20348 年 00 月 10 日

Linux

平台名稱	版本編號
AlmaLinux	8.10、9.2、9.3、9.4
Amazon Linux 2	2.0 及更新版本
Amazon Linux 2023	2023.0.20230315.0 及更新版本
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 (最終)、6.10、6.10 (最終)
CentOS Linux	7、7.2.1511、7.2.1511 (Core)、7.3.1611、7.3.1611 (Core)、7.4.1708、7.5.1804、7.5.1804 (Core)、7.6.1810、7.7.1908、7.8.2003、7.9.2009、7.9.2009 (Core)、8.2.2004、8.5.2111
CentOS Stream	8、9

平台名稱	版本編號
Debian GNU/Linux	8-12
Oracle Linux 伺服器	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 伺服器	6.9 (Santiago)、6.10 (Santiago)、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 探索受管節點

Systems Manager 與 Amazon Q Developer 的整合可讓您從 Amazon Q 介面可用的任何 AWS Management Console 位置，針對機群中的受管節點提出問題。

如需有關與 Amazon Q 互動的詳細資訊，請參閱《Amazon Q Developer 使用者指南》中的 Chatting with Amazon Q Developer about AWS。

使用 Amazon Q 探索受管節點

1. 從 中的任何位置 AWS Management Console，選擇 Amazon Q 圖示 ( )。
2. 在 Amazon Q 窗格底部的提示欄位中，詢問有關帳戶或組織中受管節點的問題。

   > **Tip**
   >
   > 如需建立有效提示的秘訣，請檢閱 學習制定有效的提示向 Amazon Q 詢問有關機群的問題 中的資訊。

3. 檢查有關特定節點的資訊，或選擇開啟 AWS Systems Manager 主控台以繼續探索。

檢視個別節點詳細資訊並對節點採取動作

從 Systems Manager 中的探索節點頁面清單中，可以選取個別節點，以檢視機器的完整詳細資訊，或對節點執行各種動作。詳細資訊頁面中的一般頁面提供有關節點的完整資訊。

檢視個別節點詳細資訊並對節點採取動作

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇探索節點。
3. (選用) 請依照選擇受管節點摘要的篩選條件檢視中的步驟，精簡為組織或帳戶顯示的受管節點清單。
4. 在節點 ID 欄位中，選擇節點的連結 ID。
5. 若要檢視有關節點的更多詳細資訊，請在左側導覽的屬性清單中，選擇屬性以檢視有關下列項目的詳細資訊：

   - 標籤：檢視套用至節點的標籤清單。您也可以新增或移除標籤。
   - 庫存：選擇庫存類型，如 AWS:Application 或 AWS:Network，以檢視節點的庫存詳細資訊。
   - 關聯：檢視有關套用至節點的所有 State Manager 關聯的詳細資訊，包括狀態和關聯 SSM 文件名稱等詳細資訊。
   - 修補程式：檢視有關節點修補程式和修補程式狀態的摘要資訊。
   - 組態合規：檢視節點的合規詳細資訊，如合規狀態和合規問題嚴重性。
6. 若要對節點採取動作，請使用節點動作選單中的下列選項：

   > **Note**
   >
   > 這些動作僅適用於您目前工作所在 AWS 帳戶 和區域中的受管節點。針對您可能在其他帳戶或區域中存取的受管節點，您可以改為存取屬性清單。

   - 連線、啟動終端工作階段：使用 AWS Systems Manager Session Manager 連線至節點。
   - 工具
     - 檢視檔案系統：瀏覽節點目錄結構的內容。新增、重新命名和移除目錄。剪下或複製並貼上檔案。
     - 檢視效能計數器：檢視有關節點的效能資訊，例如 CPU 使用率、網路流量和其他使用率類型。

- 受管程序：檢視有關節點上資源用量的資訊。在節點上啟動或停止程序。
- 管理使用者和群組：檢視、新增或刪除節點上的使用者帳戶和使用者群組。
- 執行 Run Command：使用 AWS Systems Manager Run Command 管理節點的組態。Run Command 使用 Systems Manager 文件執行隨需變更，例如更新應用程式或執行 Linux Shell 指令碼和 Windows PowerShell 命令。
- 修補程式節點：使用 AWS Systems Manager Patch Manager 中的立即修補功能，從主控台對節點執行隨需修補操作。

> **Note**
>
> 上述任務也可以從左側導覽的工具選單中啟動。

- 節點設定
  - 新增標籤：將其他標籤鍵值對套用至節點。
  - 重設節點使用者密碼：為節點上指定的使用者設定新密碼。
  - 修改 IAM 角色：變更與節點相關聯的 IAM 角色。建立新的 IAM 角色以連接至節點。

下載或匯出受管節點報告

您可以使用 Systems Manager 探索節點功能，在 Systems Manager 主控台中檢視 AWS 組織或帳戶的受管節點篩選或未篩選清單。對於您想要離線檢視資料或在另一個應用程式中處理資料的情況，您可以將報告儲存為 CSV 或 JSON 檔案。

根據報告的大小，系統會提示您將報告下載到本機電腦，或將其匯出至 Amazon S3 儲存貯體。報告只會以 CSV 格式儲存至 S3 儲存貯體。

下載或匯出受管節點報告

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇探索節點。
3. (選用) 請依照選擇受管節點摘要的篩選條件檢視中的步驟，精簡為組織或帳戶顯示的受管節點清單。
4. 選擇報告
   (⬇)。

5. 如果顯示下載報告對話方塊，請執行下列動作：

   a. 對於檔案名稱，輸入檔案的名稱。我們建議指定代表報告範圍的名稱，例如 `all-organization-nodes` 或 `ec2-instances-out-of-date-agent`。

   b. 對於包含的欄位，指定要包含所有可用節點詳細資訊的欄位，還是只包含您為目前顯示選取的欄位。

   > **Tip**
   >
   > 如需有關管理報告顯示中欄位的資訊，請參閱管理節點報告內容和外觀。

   c. 對於檔案格式，根據檔案的用途選取 CSV 或 JSON。

   d. 對於試算表標題，若要在 CSV 檔案中包含欄位標題列，請選取包含欄位名稱的資料列。

   e. 選擇 Download (下載)。

   報告會根據瀏覽器的設定儲存至預設下載位置。

6. 如果顯示匯出至 Amazon S3 對話方塊，請執行下列動作：

   a. 在 S3 URI 欄位中，輸入匯出報告所在儲存貯體的 URI。

   > **Tip**
   >
   > 若要在 Amazon S3 主控台中檢視儲存貯體的清單，請選擇檢視。若要從帳戶的儲存貯體清單中選取，選擇瀏覽 S3。

   b. 對於授權方法，指定要用來提供將報告匯出至儲存貯體之許可的服務角色。

   如果您選擇讓 Systems Manager 為您建立角色，則其會提供操作所需的全部必要許可和信任陳述式。

   如果您想要使用或建立自己的角色，該角色必須包含必要的許可和信任陳述式。如需建立此角色的詳細資訊，請參閱建立自訂服務角色以將診斷報告匯出到 S3。

   c. 選擇提交。

建立自訂服務角色以將診斷報告匯出到 S3

在 Systems Manager 的探索節點頁面中檢視 AWS 組織或帳戶的已篩選或未篩選的受管節點清單時，可以將清單匯出為報告，以 CSV 檔案形式傳送至 Amazon S3 儲存貯體。

為此，您必須指定具有操作所需的必要許可和信任政策的服務角色。可以選擇讓 Systems Manager 在下載報告的過程中為您建立角色。也可以自行建立角色及其所需的政策。

建立自訂服務角色以將診斷報告匯出到 S3

1. 依照《IAM 使用者指南》中的 Creating policies using the JSON editor 中的步驟。

   - 使用以下內容作為政策內容，並務必使用自己的資訊取代*#####*。

```
{
  "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": "*"
    }
  ]
}
```

- 為政策命名，以協助您在下一個步驟中輕鬆識別該政策。

2. 依照《IAM 使用者指南》中的 Creating an IAM role using a custom trust policy (console) 中的步驟。

   - 對於步驟 4，輸入下列信任政策，並務必使用自己的資訊取代*#####*。

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

3. 針對步驟 10，選擇步驟 2：新增許可，然後選取在上一步中建立的政策名稱。

建立角色之後，可以在依照下載或匯出受管節點報告中的步驟操作時選取該角色。

管理節點報告內容和外觀

您可以使用 Systems Manager 探索節點功能，在 Systems Manager 主控台中檢視 AWS 組織或帳戶的受管節點篩選或未篩選清單。您可以從十幾個欄位中選擇要包含在節點清單中的欄位，例如節點 ID、作業系統名稱、區域等。您也可以重新排序清單和報告中的欄位，以及變更清單在主控台中的顯示方式。

管理節點報告內容和外觀

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇探索節點。
3. 在節點區域中選擇偏好設定齒輪圖示 ( )。
4. 在偏好設定對話方塊中，執行下列動作：
   a. 對於頁面大小，選擇每個主控台檢視顯示中包含的資料列數，即 10、25 或 50。
   b. 對於換行，選取方塊以在可用的欄位寬度中顯示儲存格的所有內容。
   c. 對於條紋效果，選取方塊來以透明和陰影背景資料列交替顯示。
   d. 對於選取可見內容，請執行下列動作：
      - 開啟或關閉清單顯示和報告的個別欄位。
      - 若要變更欄位的順序，請按住欄位名稱的拖曳控點 ( )，然後在清單中向上或向下拖曳。
5. 選擇確認。

使用 Systems Manager Just-in-time節點存取

Systems Manager 支援just-in-time存取，協助您改善節點的安全性。Just-in-time節點存取允許使用者請求暫時、有時間限制的節點存取，只有在真正需要存取時才能核准。這樣就不需要長期存取由 IAM 政策管理的節點。此外，Systems Manager 會將 RDP 工作階段的工作階段記錄提供給Windows Server節點，協助您符合合規要求、執行根本原因分析等。若要使用just-in-time節點存取，您必須設定統一的 Systems Manager 主控台。

透過just-in-time節點存取，您可以建立精細的 IAM 政策，以確保只有您允許的使用者才能向節點提交存取請求。然後，您可以建立核准政策，定義連線到節點所需的核准。對於just-in-time節點存取，有自動核准政策和手動核准政策。自動核准政策會定義使用者可以自動連線的節點。手動核准政策會定義您必須提供的手動核准數量和層級，才能存取您指定的節點。此外，您也可以建立拒絕存取政策。拒絕存取政策明確防止自動核准對您指定節點的存取請求。拒絕存取政策適用於 AWS Organizations 組織中的所有帳戶。自動核准和手動核准政策僅適用於建立它們 AWS 區域 的 AWS 帳戶 和 。

當使用者嘗試連線到節點時，系統會提示他們輸入存取節點的原因。然後評估您的核准政策。根據您的政策，使用者會自動連線到目標節點，或者 Systems Manager 會自動代表申請者建立手動核准請求。然後，套用至節點的手動核准政策中指定的核准者會收到存取請求的通知，並且可以核准或拒絕請求。在與 Slack 或 Microsoft Teams 整合的聊天應用程式中，核准者和申請者可以透過電子郵件，或透過 Amazon Q Developer 收到通知。Systems Manager 只會在指定的核准者提供所有必要的核准時，將存取權授予請求的節點。收到所有必要的核准後，使用者可以在核准政策中指定的存取時段期間，視需要對節點啟動任意數量的工作階段。Systems Manager 不會自動終止just-in-time節點存取工作階段。最佳實務是指定工作階段持續時間上限和閒置工作階段逾時工作階段偏好設定的值。這些偏好設定可防止使用者與其核准的存取時段以外的節點保持連線。

我們建議您使用核准政策的組合，以協助您保護具有更關鍵資料的節點，同時允許使用者連線到較不重要的節點，而無需介入。例如，您可以要求手動核准資料庫節點的存取請求，並自動核准工作階段至非持久性簡報層節點。

Systems Manager 支援與 IAM Identity Center 或 IAM 聯合的使用者just-in-time節點存取。當聯合身分使用者提交存取請求時，他們會指定目標節點，以及需要連線到節點的原因。Systems Manager 會將使用者的身分與您組織的核准政策中定義的參數進行比較。當符合自動核准政策條件，或核准者手動提供核准時，申請者可以連線到目標節點。當使用者嘗試連線到核准的節點時，Systems Manager 會建立並使用臨時字符來建立工作階段。

由於 Systems Manager 服務會處理存取請求和建立工作階段的身分驗證，因此您不需要使用 IAM 政策來管理對節點的存取。透過使用just-in-time節點存取，Systems Manager 可協助您的組織更接近零常設權限，因為您只需要允許使用者建立存取請求，而不是允許他們啟動對節點具有持久許可的工作階段。為了協助您符合合規要求，Systems Manager 會保留所有存取請求 1 年。Systems Manager 也會針對失敗存取請求的just-in-time節點存取發出 EventBridge 事件，以及針對手動核准存取請求的狀態更新。如需詳細資訊，請參閱 使用 Amazon EventBridge 監控 Systems Manager。

主題

- 使用 Systems Manager just-in-time存取
- 啟動just-in-time節點存取工作階段
- 管理just-in-time存取請求

- 從 移至just-in-time節點存取 Session Manager
- 使用 Systems Manager 停用just-in-time存取
- Just-in-time節點存取常見問答集

使用 Systems Manager just-in-time存取

使用 Systems Manager just-in-time節點存取涉及多個步驟。首先，您可以選擇要設定just-in-time節點存取的目標。目標由 AWS Organizations 組織單位 (OUs和 組成 AWS 區域。根據預設，系統會選取您在設定統一 Systems Manager 主控台時所選擇的相同目標，以進行just-in-time節點存取。您可以選擇為所有相同的目標或您在設定統一 Systems Manager 主控台時指定的目標子集設定just-in-time節點存取。不支援新增設定統一 Systems Manager 主控台時未選取的新目標。

接下來，您將建立核准政策，以判斷節點連線何時需要手動核准並自動核准。核准政策是由組織中的每個帳戶管理。您也可以從委派的管理員帳戶共用政策，以明確拒絕自動核准與特定節點的連線。

> **Note**
>
> 設定just-in-time節點存取不會影響您為 設定的現有 IAM 政策或偏好設定Session Manager。您必須`StartSession`從 IAM 政策中移除 等Session Manager動作的許可，以確保當使用者嘗試連線到節點時，只會使用just-in-time節點存取。在您設定just-in-time節點存取之後，建議您使用使用者和節點子集來測試核准政策，以驗證政策是否視需要運作，然後再移除 的許可 Session Manager。

身分驗證支援

請注意以下有關用於just-in-time節點存取的身分驗證支援的詳細資訊：

- 使用遠端桌面連線至Windows Server執行個體時，Just-in-time節點存取不支援單一登入身分驗證類型。
- 僅支援 AWS Security Token Service (AWS STS) `AssumeRole`臨時安全登入資料。如需詳細資訊，請參閱《IAM 使用者指南》中的以下主題：
- • IAM 中的臨時安全登入資料
  - 比較 AWS STS 登入資料
  - 請求臨時安全登入資料

下列 IAM 政策概述了管理 所需的許可，並允許使用者使用 Systems Manager 建立節點just-in-time節點存取請求。驗證您擁有使用 Systems Manager just-in-time節點存取的必要許可後，您可以繼續設定程序。將每個########取代為您自己的資訊。

啟用just-in-time政策

```
{
    "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:region:account id:automation-definition/AWS-
EnableExplorer:*"
        },
        {
            "Sid": "SSMAssociationPermissions",
            "Effect": "Allow",
            "Action": [
            "ssm:DeleteAssociation",
            "ssm:CreateAssociation",
            "ssm:StartAssociationsOnce"
            ],
            "Resource": "arn:aws:ssm:region:account id:association/*"
        },
        {
            "Sid": "SSMResourceDataSync",
            "Effect": "Allow",
            "Action": [
                "ssm:CreateResourceDataSync",
                "ssm:UpdateResourceDataSync"
            ],
            "Resource": "arn:aws:ssm:region:account-id: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:region:account id: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": "*"
        }
    ]
}
```

用於設定just-in-time政策

```
{
    "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:region:account id: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::account id: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:region:account id: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:region:account id: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:region:account id:SSM-JITNA*"
```

```
            ]
        },
        {
            "Sid": "SNSListTopics",
            "Effect": "Allow",
            "Action": [
                "sns:ListTopics"
            ],
            "Resource": "*"
        },
        {
            "Sid": "EventBridgeRuleManagement",
            "Effect": "Allow",
            "Action": [
                "events:PutRule",
                "events:PutTargets"
            ],
            "Resource": [
                "arn:aws:events:region:account id: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:region:account id:servicesetting/ssm/access-request/email-
role-mapping",
                "arn:aws:ssm:region:account id: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::account id: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::account id:role/role name",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "chatbot.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Sid": "AllowIamServiceRoleForChat",
```

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

存取請求核准者的 IAM 政策

```
{
    "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:region:account id: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:region:account id: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": "*"
        }
```

```
    ]
}
```

即時just-in-time政策

```
{
    "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:*:account id: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**
>
> 若要限制對建立、更新或刪除核准政策之 API 操作的存取，請使用 `AutoApprovalPolicy`和 `ManualApprovalPolicy` 文件類型的 `ssm:DocumentType` 條件索引鍵。`StartAccessRequest` 和 `GetAccessToken` API 操作不支援下列全域內容金鑰：
>
> - aws：ViaAwsService
> - aws:MultiFactorAuthPresent
> - aws:SourceVpce
> - aws:UserAgent

如需 Systems Manager 條件內容索引鍵的詳細資訊，請參閱《服務授權參考》中的 [的條件索引鍵 AWS Systems Manager]。

下列程序說明如何完成just-in-time節點存取的第一個設定步驟。

設定just-in-time節點存取

1. 登入組織的 Systems Manager 委派管理員帳戶。
2. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
3. 在導覽窗格中選取Just-in-time節點存取。
4. 選取啟用新體驗。
5. 選擇您要啟用just-in-time節點存取的區域。預設會選取您在設定統一 Systems Manager 主控台時選擇的相同區域，以進行just-in-time節點存取。不支援在設定統一 Systems Manager 主控台時選擇未選取的新區域。
6. 選取啟用just-in-time節點存取。

啟用此功能後 30 天內使用just-in-time節點存取無需付費。在 30 天的試用期之後，使用just-in-time節點存取需要付費。如需詳細資訊，請參閱 AWS Systems Manager 定價。

為您的節點建立核准政策

核准政策定義使用者存取節點所需的核准。由於just-in-time節點存取不需要透過 IAM 政策對節點提供長期許可，因此您必須建立核准政策，以允許存取您的節點。如果沒有套用至節點的核准政策，使用者就無法請求存取節點。

在just-in-time節點存取中，有三種類型的政策。政策類型為自動核准、拒絕存取和手動核准。

Just-in-time節點存取政策類型

- 自動核准政策會定義使用者可以自動連線的節點。
- 手動核准政策會定義您必須提供的手動核准數量和層級，才能存取您指定的節點。
- 拒絕存取政策明確防止對您指定的節點進行存取請求的自動核准。

拒絕存取政策適用於 AWS Organizations 組織中的所有帳戶。例如，您可以明確拒絕將 `Intern`群組自動核准到以 `Production`金鑰標記的節點。自動核准和手動核准政策僅適用於建立它們 AWS 區域的 AWS 帳戶 和 。組織中的每個成員帳戶都會管理自己的核准政策。核准政策的評估順序如下：

1. 拒絕存取
2. 自動核准
3. 手動

雖然每個組織只能有一個拒絕存取政策，以及每個帳戶和區域有一個自動核准政策，但您可能會在帳戶中有數個手動核准政策。評估手動核准政策時，just-in-time節點存取一律有利於節點更具體的政策。手動核准政策的評估順序如下：

1. 標記特定目標
2. 所有節點目標

例如，您有一個以 Demo金鑰標記的節點。在同一帳戶中，您有一個手動核准政策，以所有節點為目標，並且需要一個層級的核准。您也有手動核准政策，需要兩個層級的兩個核准，才能使用 Demo金鑰標記的節點。Systems Manager 會將以Demo標籤為目標的政策套用到節點，因為它比以所有節點為目標的政策更為具體。這可讓您為帳戶中的所有節點建立一般政策，確保使用者可以提交存取請求，同時讓您視需要建立更精細的政策。

根據您的組織，可能有多個標籤套用至您的節點。在此案例中，如果多個手動核准政策適用於節點，則存取請求會失敗。例如，節點會加上 Production和 Database金鑰的標籤。在同一帳戶中，您有一個手動核准政策，適用於以 Production 金鑰標記的節點，以及另一個手動核准政策，適用於以 Database金鑰標記的節點。這會導致標記金鑰和存取請求的節點發生衝突。Systems Manager 會將使用者重新導向至失敗的請求。在那裡，他們可以檢視衝突政策和標籤的詳細資訊，以便在他們擁有必要的許可時進行必要的調整。否則，他們可以通知組織中的同事具有修改政策所需的許可。導致存取請求失敗的政策衝突會發出 EventBridge 事件，讓您靈活地建置自己的回應工作流程。此外，Systems Manager 會針對政策衝突，向您指定的收件人傳送導致存取請求失敗的電子郵件通知。如需設定政策衝突電子郵件通知的詳細資訊，請參閱 設定just-in-time存取請求的通知。

在拒絕存取政策中，您可以使用 Cedar 政策語言來定義使用者明確無法自動連線到組織中哪些節點。此政策是從組織的委派管理員帳戶建立和共用。拒絕存取政策會取代所有自動核准政策。每個組織只能有一個拒絕存取政策。

在自動核准政策中，您可以使用 Cedar 政策語言來定義哪些使用者可以自動連線到指定的節點，而無需手動核准。自動核准的存取請求的存取持續時間為 1 小時。此值無法變更。每個帳戶和區域只能有一個自動核准政策。

在手動核准政策中，您可以指定存取持續時間、需要多少層級的核准、每個層級所需的核准者數目，以及他們可以核准just-in-time存取請求的節點。手動核准政策的存取持續時間必須介於 1 到 336 小時之間。如果您指定多個層級的核准，存取請求的核准會一次處理一個層級。這表示您必須先提供某個層級所需的所有核准，才能將核准程序移至後續層級。如果您在手動核准政策中指定多個標籤，則會將其評估為or陳述式而非and陳述式。例如，如果您建立包含標籤 Application、 Web和 的手動核准政策Test，該政策會套用至任何以其中一個金鑰標記的節點。此政策不僅適用於標記全部三個金鑰的節點。

我們建議您將手動政策與自動核准政策結合使用，以協助您保護具有更關鍵資料的節點，同時允許使用者在不介入的情況下連接到較不關鍵的節點。例如，您可以要求手動核准資料庫節點的存取請求，並自動核准工作階段到非持久性簡報層節點。

下列程序說明如何建立just-in-time節點存取的核准政策。

主題

- 建立just-in-time節點存取的手動核准政策
- 自動核准和拒絕存取政策的陳述式結構和內建運算子
- 建立just-in-time節點存取的自動核准政策
- 建立拒絕存取政策以just-in-time存取節點
- 使用 Amazon Q 建立just-in-time節點存取的核准政策

建立just-in-time節點存取的手動核准政策

下列程序說明如何建立手動核准政策。Systems Manager 可讓您為每個 AWS 帳戶 和 建立最多 50 個手動核准政策 AWS 區域。

建立手動核准政策

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中選取管理節點存取。
3. 在建立手動核准政策步驟的政策詳細資訊區段中，輸入核准政策的名稱和描述。
4. 輸入存取持續時間的值。這是核准存取請求後，使用者可以開始對節點進行工作階段的時間上限。值必須介於 1 到 336 小時之間。
5. 在節點目標區段中，輸入與您希望政策套用之節點相關聯的標籤鍵/值對。如果政策中指定的標籤都未與節點相關聯，則政策不會套用至節點。
6. 在存取請求核准者區段中，輸入您要在政策中核准節點目標存取請求的使用者或群組。存取請求核准者可以是 IAM Identity Center 使用者和群組或 IAM 使用者。每個層級最多可以指定 5 個核准者，以及最多 5 個核准者層級。
7. 選取建立手動核准政策。

自動核准和拒絕存取政策的陳述式結構和內建運算子

下表顯示自動核准和拒絕存取政策的結構。

元件	語法
效用	`permit | forbid`
scope	`(principal, action, resource)`
條件子句	`when { principal or resource has attribute name };`

政策元件

自動核准或拒絕存取政策包含下列元件：

- 效果 – `permit`（允許）或 `forbid`（拒絕）存取。
- 範圍 – 套用效果的主體、動作和資源。您可以透過無法識別特定委託人、動作或資源，讓 Cedar 中的範圍保持未定義。在此情況下，政策適用於所有可能的委託人、動作和資源。對於just-in-time節點存取， 一律action為 `AWS::SSM::Action::"getTokenForInstanceAccess"`。
- 條件子句 – 套用效果的內容。

說明

您可以在政策中包含註解。註解定義為以換行字元開頭//和結尾的行。

下列範例顯示政策中的註解。

```
// 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"
};
```

多個子句

您可以使用 &&運算子在政策陳述式中使用多個條件子句。

```
// 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"
};
```

預留字元

下列範例示範如何在內容屬性使用：（分號） 時撰寫政策，這是政策語言的預留字元。

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

如需額外的範例，請參閱範例政策陳述式。

Just-in-time節點存取結構描述

以下是just-in-time節點存取的 Cedar 結構描述。

```
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
        }
    };
}
```

內建運算子

使用各種條件建立自動核准或拒絕存取政策的內容時，您可以使用 &&運算子來新增其他條件。您也可以使用許多其他內建運算子，為您的政策條件新增額外的表達式能力。下表包含所有內建運算子以供參考。

運算子	類型和過載	描述
!	布林值 → 布林值	邏輯不是。

運算子	類型和過載	描述
==	任何 → 任何	平等。適用於任何類型的引數，即使類型不相符。不同類型的值永遠不會彼此相等。
!=	任何 → 任何	不等式；完全相同的反向（請參閱上述）。
<	（長、長）→ 布林值	小於 的長整數。
<=	（長、長）→ 布林值	less-than-or-equal-to的長整數。
>	（長、長）→ 布林值	大於 的長整數。
>=	（長、長）→ 布林值	greater-than-or-equal-to。
in	（實體、實體）→ 布林值	階層成員資格（反射：A 中的 A 一律為 true)。
	（實體，set（實體））→ 布林值	階層成員資格：如果 (A 和 B) || (C 中的 A) || ... 如果集合包含非實體，則 【B、C、...】 中的 A 為 true。
&&	（布林值、布林值）→ 布林值	邏輯 和（短路）。
||	（布林值、布林值）→ 布林值	邏輯 或（短路）。
.exists()	實體 → 布林值	實體存在。
具有	（實體、屬性）→ 布林值	Infix Operator。 會e has f測試記錄或實體是否e具有屬性的繫結f。false 如果 e 不存在或 e存在但 沒有 屬性 ，則傳回 f。屬性可以表示為識別符或字串常值。

運算子	類型和過載	描述
like	(字串、字串) → 布林值	Infix Operator。會t like p檢查文字是否t與模式 相符p，其中可能包含*符合 0 或多個任何字元的萬用字元。為了符合 中的常值星星字元t，您可以使用 *中的特殊逸出字元序列p。
.hasTag()	(實體、字串) → 布林值	檢查實體是否已套用指定的標籤。
.getTag()	(實體、字串) → 布林值	傳回指定標籤索引鍵的值。
.contains()	(集合，任何) → 布林值	設定成員資格 (B 為 A 元素)。
.containsAll()	(設定、設定) → 布林值	測試集合 A 是否包含集合 B 中的所有元素。
.containsAny()	(設定、設定) → 布林值	測試集合 A 是否包含集合 B 中的任何元素。

範例政策陳述式

以下是政策陳述式範例。

```
// 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"
};
```

建立just-in-time節點存取的自動核准政策

自動核准政策使用 Cedar 政策語言來定義哪些使用者可以自動連線到指定的節點，而無需手動核准。自動核准政策包含多個指定 `principal`和 的`permit`陳述式`resource`。每個陳述式都包含定義自動核准條件的`when`子句。

以下是自動核准政策的範例。

```
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"
};
```

下列程序說明如何為just-in-time節點存取建立自動核准政策。自動核准的存取請求的存取持續時間為 1 小時。此值無法變更。每個 AWS 帳戶 和 只能有一個自動核准政策 AWS 區域。如需如何建構政策陳述式的詳細資訊，請參閱 自動核准和拒絕存取政策的陳述式結構和內建運算子。

建立自動核准政策

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中選取管理節點存取。
3. 在核准政策索引標籤中，選取建立自動核准政策。
4. 在政策陳述式區段中輸入自動核准政策的政策陳述式。您可以使用提供的範例陳述式來協助您建立政策。
5. 選取建立自動核准政策。

建立拒絕存取政策以just-in-time存取節點

拒絕存取政策使用 Cedar 政策語言來定義使用者無法在未經手動核准的情況下自動連線到哪些節點。拒絕存取政策包含多個指定 `principal`和 的`forbid`陳述式`resource`。每個陳述式都包含一個子`when`句，定義明確拒絕自動核准的條件。

以下是拒絕存取政策的範例。

```
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"
};
```

下列程序說明如何為just-in-time節點存取建立拒絕存取政策。如需如何建構政策陳述式的詳細資訊，請參閱 自動核准和拒絕存取政策的陳述式結構和內建運算子。

> **Note**
>
> 記下以下資訊。
>
> - 您可以在登入 AWS 管理帳戶或委派管理員帳戶時建立拒絕存取政策。您的 AWS Organizations 組織只能有一個拒絕存取政策。

- Just-in-time節點存取使用 AWS Resource Access Manager (AWS RAM) 與組織中的成員帳戶共用拒絕存取政策。如果您想要與組織中的成員帳戶共用拒絕存取政策，則必須從組織的管理帳戶啟用資源共用。如需詳細資訊，請參閱《AWS RAM 使用者指南》中的透過 AWS Organizations啟用共用。

建立拒絕存取政策

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中選取管理節點存取。
3. 在核准政策索引標籤中，選取建立拒絕存取政策。
4. 在政策陳述式區段中輸入拒絕存取政策的政策陳述式。您可以使用提供的範例陳述式來協助您建立政策。
5. 選取建立拒絕存取政策。

使用 Amazon Q 建立just-in-time節點存取的核准政策

將 Amazon Q Developer 用於命令列，可在軟體開發的各個層面提供指導和支援。對於just-in-time節點存取，Amazon Q 透過產生和更新政策的程式碼、分析政策陳述式等，協助您建立核准政策。以下資訊說明如何使用 Amazon Q for 命令列建立核准政策。

識別您的使用案例

建立核准政策的第一步是清楚定義您的使用案例。例如，在您的組織中，您可能想要使用 `Environment:Testing`標籤自動核准對節點的存取請求。如果員工 ID 以 開頭，您可能也想要明確拒絕對具有 `Environment:Production`標籤的節點進行自動核准TEMP。對於具有`Tier:Database`標籤的節點，您可能需要兩個層級的手動核准。

在任何特定案例中，您可能偏好一個政策或條件，而不是另一個政策或條件。因此，我們建議您明確定義要決定哪些陳述式最適合您的使用案例和偏好設定的政策行為。

設定開發環境

為您要開發核准政策的命令列安裝 Amazon Q。如需為命令列安裝 Amazon Q 的詳細資訊，請參閱《Amazon Q 開發人員使用者指南》中的為命令列安裝 Amazon Q。

我們也建議您為 AWS 文件安裝 MCP 伺服器。此 MCP 伺服器會將 Amazon Q for 命令列連線至最新的文件資源。如需有關在命令列使用 MCP 搭配 Amazon Q 的資訊，請參閱《Amazon Q 開發人員使用者指南》中的搭配使用 MCP 搭配 Amazon Q 開發人員。

如需 AWS 文件 MCP 伺服器的詳細資訊，請參閱 AWS 文件 MCP 伺服器。

如果您尚未安裝和設定 AWS CLI。如需詳細資訊，請參閱安裝或更新最新版本的 AWS CLI。

開發核准政策內容

識別您的使用案例並設定環境後，您就可以開發政策的內容。您的使用案例和偏好設定將主要決定您使用的核准政策和陳述式類型。

如果您不確定如何使用特定政策，或需要有關政策結構描述的詳細資訊，請參閱 為您的節點建立核准政策和後續主題。這些主題詳細說明如何評估政策，並提供有效的範例陳述式。

下列程序說明如何使用 Amazon Q for 命令列建立核准政策。

> **Note**
>
> Amazon Q Developer 使用生成式 AI。您可能需要驗證回應。請參閱AWS 負責任的 AI 政策。

使用 Amazon Q for 命令列建立核准政策

1. 開啟終端機並使用 Amazon Q 啟動工作階段。

```
$ q chat
```

2. 輸入提示並描述您要建立哪種政策陳述式。例如，針對核准政策使用先前的案例，您可以輸入下列提示：

```
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. 輸入此範例提示會產生下列輸出：

```
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. 將政策陳述式貼到新的 Cedar 檔案中，以驗證語法。您也可以使用 Cedar 遊樂場驗證陳述式。
5. 驗證陳述式之後，請儲存 Cedar 檔案。
6. 從終端機執行下列 AWS CLI 命令，以使用您的政策內容建立 SSM 文件。在生產環境中使用核准政策之前，請使用 AWS 帳戶 和 中的身分和節點子集來測試核准政策 AWS 區域。

> **Note**
>
> 對於自動核准政策，文件名稱必須是 SSM-JustInTimeAccessAutoApprovalPolicy。每個 AWS 帳戶 和 只能有一個自動核准政策 AWS 區域。對於拒絕存取政策，文件名稱必須是 SSM-JustInTimeAccessDenyAccessOrgPolicy。每個 AWS Organizations 組織只能有一個拒絕存取政策，且必須在 Systems Manager 的委派管理員帳戶中建立政策。手動核准政策的命名限制與其他 SSM 文件相同。如需詳細資訊，請參閱 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"
```

just-in-time節點存取工作階段偏好設定

透過just-in-time節點存取，您可以在 AWS 區域 組織中的每個 和 中指定一般工作階段 AWS 帳戶 和記錄偏好設定。或者，您可以使用 AWS CloudFormation StackSets 在多個帳戶和區域中建立工作階段偏

好設定文件，協助您擁有一致的工作階段偏好設定。如需工作階段偏好設定文件結構描述的資訊，請參閱 工作階段文件結構描述。

為了記錄，我們建議您將串流選項與 Amazon CloudWatch Logs 搭配使用。此功能可讓您將工作階段資料日誌的持續串流傳送至 CloudWatch Logs。串流工作階段資料時會包含基本詳細資料，例如使用者在工作階段中執行的命令、執行命令的使用者 ID，以及工作階段資料串流至 CloudWatch Logs 時的時間戳記。串流工作階段資料時，日誌會採用 JSON 格式，以協助您與現有的日誌解決方案進行整合。

Systems Manager 不會自動終止just-in-time節點存取工作階段。最佳實務是指定工作階段持續時間上限和閒置工作階段逾時設定的值。使用這些設定可協助您防止使用者保持與節點的連線超過存取請求中核准的時段。下列程序說明如何更新just-in-time節點存取的工作階段偏好設定。

> **Important**
>
> 您必須使用標籤 AWS KMS 金鑰 和標籤值 ，在just-in-time節點存取中標記用於Session Manager加密`SystemsManagerJustInTimeNodeAccessManaged`和 RDP 記錄的金鑰`true`。
> 如需標記 KMS 金鑰的詳細資訊，請參閱《 AWS Key Management Service 開發人員指南》中的 中的標籤 AWS KMS。

更新工作階段偏好設定

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中選取設定。
3. 選取Just-in-time節點存取索引標籤。
4. 在工作階段偏好設定區段中，選取編輯。
5. 視需要更新您的一般和記錄偏好設定，然後選取儲存。

設定just-in-time存取請求的通知

您可以設定 Systems Manager 在使用者為核准者和申請者建立just-in-time節點存取請求時傳送通知給電子郵件地址或聊天用戶端。通知包含請求者提供的存取請求的原因 AWS 帳戶、 AWS 區域、請求的狀態，以及目標節點的 ID。目前，Systems Manager 透過與聊天應用程式中的 Amazon Q Developer

整合來支援 Slack 和 Microsoft Teams 用戶端。透過聊天用戶端使用通知時，存取請求核准者可以直接與存取請求互動。這不需要登入 主控台，即可對存取請求採取動作。

開始之前

設定just-in-time節點存取通知的聊天用戶端之前，請注意下列需求：

- 如果您使用 IAM 角色來管理帳戶中的使用者身分，則必須手動將要傳送通知的核准者或申請者的電子郵件地址與相關聯的角色建立關聯。否則，無法透過電子郵件通知預期的收件人。

下列程序說明如何設定just-in-time節點存取請求的通知。

設定just-in-time節點存取通知的聊天用戶端

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中選取設定。
3. 選取Just-in-time節點存取索引標籤。
4. 在聊天區段中，選取設定新用戶端。
5. 在選取用戶端類型下拉式清單中，選擇您要設定的聊天用戶端類型，然後選取下一步。
6. 系統會提示您允許聊天應用程式中的 Amazon Q Developer 存取您的聊天用戶端。選取允許。
7. 在設定頻道區段中，輸入聊天用戶端頻道的資訊，然後選取您要接收的通知類型。
8. 如果您要設定 Slack 通知，請邀請 "@Amazon Q" 加入正在設定通知的每個 Slack 頻道。
9. 選取設定頻道。

> **Note**
>
> 若要允許直接從 Slack 頻道核准/拒絕存取請求，請確定使用 Slack 頻道設定的 IAM 角色具有`ssm:SendAutomationSignal`許可，且具有包含 chatbot 的信任政策：

```
{
            "Effect": "Allow",
            "Principal": {
                "Service": "chatbot.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
```

```
}
```

為just-in-time節點存取通知設定電子郵件通知

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中選取設定。
3. 選取Just-in-time節點存取索引標籤。
4. 在電子郵件區段中，選取編輯。
5. 選取新增電子郵件，選擇您要手動建立電子郵件地址關聯的 IAM 角色。
6. 在電子郵件地址欄位中輸入電子郵件地址。每當建立需要您指定之 IAM 角色核准的存取請求時，您與該角色建立關聯的電子郵件地址都會收到通知。
7. 選取新增電子郵件地址。

記錄 RDP 連線

Just-in-time節點存取包括記錄對Windows Server節點進行的 RDP 連線的能力。記錄 RDP 連線需要 S3 儲存貯體和 AWS Key Management Service (AWS KMS) 客戶受管金鑰。 AWS KMS key 用於暫時加密在 Systems Manager 資源上產生和存放的錄製資料。客戶受管金鑰必須是具有加密和解密金鑰使用情況的對稱金鑰。您可以為您的組織使用多區域金鑰，或者您必須在已啟用just-in-time節點存取的每個區域中建立客戶受管金鑰。

如果您已在儲存記錄的 S3 儲存貯體上啟用 KMS 加密，您必須將用於儲存貯體加密的客戶受管金鑰存取權提供給`ssm-guiconnect`服務主體。此客戶受管金鑰可以是不同於您在記錄設定中指定的金鑰，其中必須包含建立連線所需的`kms:CreateGrant`許可。

設定 RDP 記錄的 S3 儲存貯體加密

您的連線記錄會存放在您在啟用 RDP 記錄時指定的 S3 儲存貯體中。

如果您使用 KMS 金鑰做為 S3 儲存貯體 (SSE-KMS) 的預設加密機制，您必須允許`ssm-guiconnect`服務主體存取金鑰上的`kms:GenerateDataKey`動作。建議在搭配 S3 儲存貯體使用 SSE-KMS 加密時，使用客戶受管金鑰。這是因為您可以更新客戶受管金鑰的相關金鑰政策。您無法更新 的金鑰政策 AWS 受管金鑰。

Important

您必須使用標籤 AWS KMS 金鑰 和標籤值 ，在just-in-time節點存取中標記用於Session Manager加密`SystemsManagerJustInTimeNodeAccessManaged`和 RDP 記錄的金鑰`true`。
如需有關標記 KMS 金鑰的資訊，請參閱《 AWS Key Management Service 開發人員指南》中的 中的標籤 AWS KMS。

使用下列客戶受管金鑰政策，允許`ssm-guiconnect`服務存取 S3 儲存的 KMS 金鑰。如需有關更新客戶受管金鑰的資訊，請參閱《 AWS Key Management Service 開發人員指南》中的變更金鑰政策。

將每個`########`取代為您自己的資訊：

- *`account-id`* 代表啟動連線 AWS 帳戶 的 ID。
- *`region`* 代表 S3 AWS 區域 儲存貯體所在的 。(*如果儲存貯體將收到來自多個區域的錄音，您可以使用 。 範例：`s3.*.amazonaws.com`。)

Note

您可以在政策`aws:SourceOrgID`中使用 ，而不是`aws:SourceAccount`如果帳戶屬於 組織 AWS Organizations。

```
{
    "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"
        }
    }
}
```

設定記錄 RDP 連線的 IAM 許可

除了just-in-time節點存取所需的 IAM 許可之外，還必須根據您需要執行的任務，允許您使用的使用者或角色下列許可。

設定連線錄製的許可

若要設定 RDP 連線錄製，需要下列許可：

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

啟動連線的許可

若要使用just-in-time節點存取進行 RDP 連線，需要下列許可：

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

開始之前

若要存放連線錄音，您必須先建立 S3 儲存貯體，並新增下列儲存貯體政策。將每個########取代為您自己的資訊。

（如需新增儲存貯體政策的相關資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的使用 Amazon S3 主控台新增儲存貯體政策。）

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ConnectionRecording",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ssm-guiconnect.amazonaws.com"
                ]
            },
            "Action": "s3:PutObject",
            "Resource": [
                "arn:aws:s3:::bucket name",
                "arn:aws:s3:::bucket name/*"
            ],
            "Condition":{
            "StringEquals":{
                "aws:SourceAccount":"123456789012"
                }
            }
        }
    ]
}
```

啟用和設定 RDP 連線記錄

下列程序說明如何啟用和設定 RDP 連線記錄。

啟用和設定 RDP 連線記錄

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中選取設定。
3. 選取Just-in-time節點存取索引標籤。
4. 在 RDP 錄製區段中，選取啟用 RDP 錄製。
5. 選擇您要上傳工作階段錄製的 S3 儲存貯體。
6. 選擇您要使用的客戶受管金鑰，在記錄資料產生並存放在 Systems Manager 資源時暫時加密記錄資料。（這可以是不同於您用來加密儲存貯體的客戶受管金鑰。）
7. 選取 Save (儲存)。

RDP 連線記錄狀態值

RPD 連線記錄的有效狀態值包括下列項目：

- `Recording` - 連線正在進行記錄
- `Processing` - 連線終止後正在處理影片。
- `Finished` - 成功的終端狀態：連線錄製影片已成功處理並上傳至指定的儲存貯體。
- `Failed` - 失敗的終端機狀態。未成功記錄連線。
- `ProcessingError` - 影片處理期間發生一或多個中繼失敗/錯誤。可能的原因包括服務相依性失敗，或由於在用於儲存記錄的 S3 儲存貯體上設定錯誤而缺少許可。當錄製處於此狀態時，服務會繼續嘗試處理。

> **Note**
>
> `ProcessingError` 可能是建立連線後`ssm-guiconnect`，服務主體沒有許可將物件上傳至 S3 儲存貯體的結果。另一個潛在原因是缺少用於 S3 儲存貯體加密之 KMS 金鑰的 KMS 許可。

修改目標

當您設定just-in-time節點存取時，您可以選擇要設定just-in-time節點存取的目標。目標由 AWS Organizations 組織單位 (OUs和 組成 AWS 區域。根據預設，系統會選取您在設定統一 Systems Manager 主控台時所選擇的相同目標，以進行just-in-time節點存取。您可以選擇為所有相同的目標或您在設定統一 Systems Manager 主控台時指定的目標子集設定just-in-time節點存取。不支援新增設定統一 Systems Manager 主控台時未選取的新目標。您可以在設定just-in-time節點存取後變更您選取的目標。

下列程序說明如何修改just-in-time節點存取的目標。

修改目標

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中選取設定。
3. 選取Just-in-time節點存取索引標籤。
4. 在目標區段中，選取編輯。

5. 選取您要使用just-in-time節點存取的組織單位和區域。
6. 選取 Save (儲存)。

變更身分提供者

根據預設，just-in-time節點存取會使用身分提供者的 IAM。啟用just-in-time節點存取後，搭配 組織使用統一主控台的客戶可以修改此設定，以使用 IAM Identity Center。設定單一帳戶和區域時，Just-in-time節點存取不支援 IAM Identity Center 做為身分提供者。

下列程序說明如何修改身分提供者以進行just-in-time節點存取。

修改身分提供者

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中選取設定。
3. 選取Just-in-time節點存取索引標籤。
4. 在使用者身分區段中，選取編輯。
5. 選擇 AWS IAM Identity Center。
6. 選取 Save (儲存)。

啟動just-in-time節點存取工作階段

在啟用和設定just-in-time節點存取，以及設定工作階段和通知偏好設定之後，使用者就可以開始just-in-time節點存取工作階段。您可以使用 Systems Manager just-in-time節點存取，或使用 AWS Command Line Interface Session Manager外掛程式從 啟動工作階段。可以在相同帳戶和區域中的節點上啟動 Just-in-time節點存取工作階段。下列程序說明如何使用just-in-time節點存取來啟動工作階段。

> **Note**
>
> 如果您的使用者先前用於Session Manager連線至節點，您必須從其 IAM 政策中移除Session Manager許可，例如 `ssm:StartSession`，以使用just-in-time節點存取來啟動工作階段。否則，當連接到節點時，它們將繼續使用 Session Manager。

使用主控台啟動具有just-in-time節點存取權的工作階段

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中選取探索節點。
3. 選取您要連線的節點。
4. 在動作下拉式清單中，選取連線。

如果您組織的核准政策不允許您自動連線到節點，系統會提示您提交存取請求。在您填寫請求的資訊並提交存取請求後，您就可以在收到所有必要的核准後開始對節點的工作階段。

使用 just-in-time節點存取的工作階段 AWS CLI

1. 執行下列命令來啟動存取請求工作流程，請務必將*#####*取代為您自己的資訊。

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

   視組織的核准政策而定，您會自動連線到節點，或啟動手動核准程序。對於需要手動核准的請求，請注意回應中傳回的存取請求 ID。
2. 等待提供所有必要的核准。
3. 提供所有必要的核准之後，請執行下列命令，以取得包含臨時登入資料的存取字符。將*#####*取代為您自己的資訊。

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

   請注意回應中傳回的存取字符。
4. 執行下列命令以在 中使用臨時登入資料 AWS CLI，請務必將*#####*取代為您自己的資訊。

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

5. 執行下列命令來啟動節點的工作階段，請務必將*#####*取代為您自己的資訊。

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

管理just-in-time存取請求

為了提高整個組織的可見性，Systems Manager 會將存取請求複寫到組織的委派管理員帳戶。為了協助您符合合規要求，Systems Manager 會保留所有存取請求 1 年。下列主題說明如何管理just-in-time節點存取請求。此資訊適用於存取請求核准者。開始之前，我們建議您檢閱 IAM 政策，並確保您擁有管理just-in-time節點存取所需的許可。如需詳細資訊，請參閱 使用 Systems Manager just-in-time存取。

主題

- 核准和拒絕just-in-time節點存取請求

核准和拒絕just-in-time節點存取請求

存取請求核准者可以核准或拒絕來自統一 Systems Manager 主控台或使用您偏好的命令列工具的just-in-time節點存取請求。此資訊適用於存取請求核准者。如果您沒有核准或拒絕存取請求所需的許可，請聯絡您的管理員。下列程序說明如何核准或拒絕just-in-time節點存取請求。

使用主控台核准或拒絕just-in-time節點存取請求

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中選取管理節點存取。
3. 選取存取請求索引標籤。
4. 選取我的請求切換。
5. 選取您要核准或拒絕之存取請求旁的核取方塊。
6. 選取核准或拒絕。

核准存取請求之後，您可以隨時選取撤銷來撤銷您的核准。

使用命令列核准或拒絕just-in-time節點存取請求

1. 請注意來自 通知的存取請求 ID。例如，*oi-12345abcdef*。
2. 執行下列命令以傳回存取請求核准工作流程的詳細資訊，請務必使用您自己的資訊取代*#####*。

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

請注意 /aws/accessrequest欄位中automationExecutionId的值OperationalData。例如，*9231944f-61c6-40be-8bce-8ee2bEXAMPLE*。

3. 執行下列命令來核准或拒絕存取請求。使用 Approve 訊號類型來核准請求，以及Deny拒絕請求。請務必將*#####*取代為您自己的資訊。

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

從 移至just-in-time節點存取 Session Manager

當您啟用just-in-time節點存取時，Systems Manager 不會對 的現有資源進行任何變更Session Manager。這可確保現有環境不會中斷，而且使用者可以在您建立和驗證核准政策時繼續啟動工作階段。準備好測試核准政策後，您必須修改現有的 IAM 政策，才能完成轉換為just-in-time節點存取。這包括新增just-in-time節點存取身分所需的許可，以及移除 的 StartSession API 操作許可Session Manager。我們建議使用 AWS 帳戶 和 中的身分和節點子集來測試核准政策 AWS 區域。

如需just-in-time節點存取所需許可的詳細資訊，請參閱 使用 Systems Manager just-in-time存取。

如需修改 和身分的 IAM 許可的詳細資訊，請參閱《IAM 使用者指南》中的新增和移除 IAM 身分許可。

以下說明如何從 just-in-time節點存取的詳細方法Session Manager。

從 Session Manager 移至just-in-time節點存取需要仔細規劃和測試，以確保順利轉換，而不會中斷您的操作。下列各節說明如何完成此程序。

先決條件

開始之前，請確定您已完成下列任務：

- 設定 Systems Manager 統一主控台。
- 已驗證您具有修改帳戶中 IAM 政策的許可。
- 識別目前授予Session Manager許可的所有 IAM 政策和角色。
- 已記錄您目前的Session Manager組態，包括工作階段偏好設定和記錄設定。

評估

完成下列任務，評估您目前的環境並概述所需的核准行為：

1. 清查您的節點 - 識別使用者目前透過 存取的所有節點Session Manager。
2. 識別使用者存取模式 - 記錄哪些使用者或角色需要存取哪些節點，以及在何種情況下。
3. 映射核准工作流程 - 決定誰應該核准不同類型的節點的存取請求。
4. 檢閱標記策略 - 確保您的節點已正確標記，以支援您計劃的核准政策。
5. 稽核現有的 IAM 政策 - 識別包含Session Manager許可的所有政策。

規劃

分階段策略

從 移至Session Managerjust-in-time節點存取時，我們建議使用如下的分階段方法：

1. 階段 1：設定和組態 - 啟用just-in-time節點存取，而不修改現有的Session Manager許可。
2. 階段 2：政策開發 - 建立和測試節點的核准政策。
3. 階段 3：試行遷移 - 將一小群非關鍵節點和使用者或角色從 修改Session Manager為just-in-time節點存取。
4. 階段 4：完全遷移 - 逐漸遷移所有剩餘的節點和使用者或角色。

時間軸考量事項

建立要從 移至Session Managerjust-in-time節點存取的時間軸時，請考慮下列因素：

- 允許使用者訓練和調整新核准工作流程的時間。
- 在較低的操作活動期間排程遷移。
- 包含用於疑難排解和調整的緩衝時間。
- 規劃一段時間的平行操作，其中兩個系統皆可使用。

實作步驟

階段 1：設定和組態

1. 在 Systems Manager just-in-time節點存取。如需詳細步驟，請參閱使用 Systems Manager just-in-time存取。

2. 設定just-in-time節點存取的工作階段偏好設定，以符合您目前的Session Manager設定。如需詳細資訊，請參閱just-in-time節點存取工作階段偏好設定。
3. 設定存取請求的通知偏好設定。如需詳細資訊，請參閱設定just-in-time存取請求的通知。
4. 如果您使用Windows Server節點的 RDP 連線，請設定 RDP 記錄。如需詳細資訊，請參閱記錄 RDP 連線。

階段 2：政策開發

1. 為just-in-time節點存取管理員和使用者建立 IAM 政策。
2. 根據您的安全需求和使用案例制定核准政策。
3. 在非生產環境中測試您的政策，以確保它們如預期般運作。

階段 3：試行遷移

1. 為試行選取一小群使用者和非關鍵節點。
2. 為包含just-in-time政策。
3. 從試用使用者的 IAM 政策中移除Session Manager許可 (`ssm:StartSession`)。
4. 在新的存取請求工作流程上訓練試行使用者。
5. 監控試行方案是否有問題並收集意見回饋。
6. 根據試行結果調整政策和程序。

試用使用者的 IAM 政策修改範例

具有Session Manager許可的原始政策：

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

```
                    }
                ]
            }
```

修改just-in-time節點存取的政策：

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

階段 4：完整遷移

制定排程，以批次遷移剩餘的使用者和節點。

測試方法

在整個遷移過程中，執行下列測試：

- 政策驗證 - 驗證核准政策是否正確套用至預期的節點和使用者。
- 存取請求工作流程 - 針對自動核准和手動核准案例，測試從存取請求到工作階段建立的完整工作流程。
- 通知 - 驗證核准者是否透過設定的管道（電子郵件、Slack、Microsoft Teams) 接收通知。
- 記錄和監控 - 確認工作階段日誌和存取請求已正確擷取和儲存。

成功遷移的最佳實務

- 及早且經常地溝通 - 通知使用者有關遷移時間表和just-in-time節點存取的優勢。
- 從非關鍵系統開始 - 從開發或測試環境開始遷移，然後再移至生產環境。

- 記錄一切 - 維護核准政策、IAM 政策變更和組態設定的詳細記錄。
- 監控和調整 - 持續監控存取請求和核准工作流程，並視需要調整政策。
- 建立控管 - 建立在您的環境變更時定期檢閱和更新核准政策的程序。

使用 Systems Manager 停用just-in-time存取

下列程序說明如何停用just-in-time節點存取。停用just-in-time節點存取後，您的組織中的使用者可能無法連線到節點，除非您已實作其他連線方法。

停用just-in-time節點存取

1. 登入組織的 Systems Manager 委派管理員帳戶。
2. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
3. 在導覽窗格中選取設定。
4. 在Just-in-time節點存取索引標籤中，選取停用。

Just-in-time節點存取常見問答集

如何從 移至Session Manager即時節點存取？ just-in-time

設定統一主控台並啟用just-in-time節點存取後，您必須修改現有的 IAM 政策，以完成移至just-in-time節點存取。這包括新增just-in-time節點存取所需的許可，以及移除 的 `StartSession` API 操作許可 Session Manager。如需just-in-time節點存取的 IAM 政策詳細資訊，請參閱使用 Systems Manager just-in-time存取。

我是否必須設定統一主控台才能使用just-in-time節點存取？

是，設定統一主控台是just-in-time節點存取的先決條件。不過，在您設定統一主控台並啟用just-in-time節點存取之後，有數種方法可以連線至節點。例如，您可以從 Amazon EC2 主控台和 just-in-time節點存取工作階段 AWS CLI。如需設定統一主控台的詳細資訊，請參閱 為組織設定 Systems Manager 統一主控台。

just-in-time節點存取是否有相關成本？

Systems Manager 提供 30 天的免費試用，讓您just-in-time存取節點。試用之後，just-in-time節點存取會產生成本。如需詳細資訊，請參閱AWS Systems Manager 定價。

just-in-time節點存取核准政策的優先順序為何？

核准政策的評估順序如下：

1. 拒絕存取
2. 自動核准
3. 手動

如何評估手動核准政策？

Just-in-time節點存取一律有利於節點更具體的政策。手動核准政策的評估順序如下：

1. 標記特定目標
2. 所有節點目標

如果沒有適用於節點的核准政策，會發生什麼情況？

若要使用just-in-time節點存取連線到節點，核准政策必須套用至節點。如果沒有適用於節點的核准政策，則使用者無法請求存取節點。

多個核准政策是否可以以標籤為目標？

在您的核准政策中，標籤只能以一次為目標。

如果多個手動核准政策因標籤重疊而套用至節點，會發生什麼情況？

當多個手動核准政策套用到節點時，這會導致衝突，使用者無法請求存取節點。建立手動核准政策時，請謹記這一點，因為某些執行個體可能會有多個標籤，視您的案例而定。

我可以使用just-in-time節點存取來請求跨帳戶和區域的節點存取和啟動工作階段嗎？

Just-in-time節點存取支援請求存取和啟動與請求者相同帳戶和區域中節點的工作階段。

我可以使用just-in-time節點存取，在註冊混合啟用的節點上請求存取和啟動工作階段嗎？

是，just-in-time節點存取支援在註冊混合啟用的節點上請求存取和啟動工作階段。節點必須與申請者在相同的帳戶和區域中註冊。

診斷和修復

使用統一的 Systems Manager 主控台，您可以在單一診斷操作中識別整個機群的問題。如果是多個組織，則可以接著使用單一自動化操作，嘗試修復所有或僅選取的目標。如果是一個組織，身為委派帳戶管理員，您可以選取所有帳戶和區域中的目標。如果您使用的是單一帳戶，則可以一次選取單一區域中的目標。

Systems Manager 可以診斷並協助您修復多種類型的部署失敗和漂移組態。Systems Manager 也可以識別帳戶或組織中 Systems Manager 無法視為受管節點的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。EC2 執行個體診斷程序可以識別與虛擬私有雲端 (VPC)、網域名稱服務 (DNS) 設定或 Amazon Elastic Compute Cloud (Amazon EC2) 安全群組的錯誤組態相關的問題。

> **Note**
>
> Systems Manager 支援混合多雲端環境中作為受管節點的 EC2 執行個體和其他機器類型。若要成為受管節點，必須在機器上安裝 AWS Systems Manager Agent (SSM Agent)，且 Systems Manager 必須具有在機器上執行動作的許可。
> 對於 EC2 執行個體，可以使用 AWS Identity and Access Management (IAM) 角色在帳戶層級提供此許可，或使用執行個體設定檔在執行個體層級提供此許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可。
> 對於非 EC2 機器，此許可是使用 IAM 服務角色提供。如需詳細資訊，請參閱在混合多雲端環境中建立 Systems Manager 所需的 IAM 服務角色。

開始之前

若要使用診斷和修復功能來偵測未受管的 EC2 執行個體，您必須先將組織或帳戶加入統一的 Systems Manager 主控台。在此流程中，必須選擇選項以建立這些操作所需的 IAM 角色和受管政策。如需詳細資訊，請參閱為組織設定 Systems Manager 統一主控台。

使用以下主題來協助您識別和修正某些常見類型的失敗部署、漂移組態和未受管 EC2 執行個體。

主題

- 診斷和修復失敗部署
- 診斷和修復漂移組態
- 在 Systems Manager 中診斷和修復未受管的 Amazon EC2 執行個體
- 執行手冊動作的修復影響類型

- 檢視 Systems Manager 中修復的執行進度和歷史記錄

診斷和修復失敗部署

Systems Manager 可以診斷失敗部署，協助修復下列類型的失敗部署：

- 組織成員帳戶的核心設定
- 委派管理員帳戶的核心設定
- 您帳戶的核心設定

按照下列程序嘗試修復這些類型的問題。

診斷和修復失敗部署

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇診斷和修復。
3. 選擇部署問題索引標籤。
4. 在失敗部署區段中，檢閱失敗部署的調查結果清單。
5. 在設定步驟欄位中，選擇調查結果的名稱，檢閱有關問題的其他詳細資訊。例如：組織成員帳戶的核心設定。
6. 在該失敗部署的詳細資訊頁面中，您可以檢視帳戶清單，以及每個帳戶中有多少區域出現過部署失敗。
7. 選取帳戶 ID 來檢視該帳戶中失敗原因的相關資訊。
8. 在失敗區域區域中，檢查針對狀態原因提供的資訊。這些資訊可以指出失敗部署的原因，可能會提供對需要進行的組態變更的深入分析。
9. 如果想要重試部署而不進行組態變更，請選擇重新部署。

診斷和修復漂移組態

Systems Manager 可以診斷漂移組態，協助修復下列類型的漂移組態：

- 組織成員帳戶的核心設定
- 委派管理員帳戶的核心設定
- 您帳戶的核心設定

按照下列程序嘗試修復這些類型的漂移組態。

診斷和修復漂移組態

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇診斷和修復。
3. 選擇部署問題索引標籤。
4. 在漂移部署區段中，檢閱失敗部署的調查結果清單。

   -或-

   若要執行新的診斷，請選擇偵測漂移。
5. 在設定步驟欄位中，選擇調查結果的名稱，檢閱有關問題的其他詳細資訊。例如：組織成員帳戶的核心設定。
6. 在該失敗部署的詳細資訊頁面中，您可以檢視帳戶清單，以及每個帳戶中有多少區域出現過組態漂移。
7. 選取帳戶 ID 來檢視該帳戶中組態漂移原因的相關資訊。
8. 在漂移資源區域中，資源欄位會報告出現過漂移的資源名稱。漂移類型欄位會報告資源是否已遭修改或刪除。
9. 若要重新部署預期的組態，則選擇重新部署。

在 Systems Manager 中診斷和修復未受管的 Amazon EC2 執行個體

為了協助您使用 Systems Manager 管理 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體，您可以使用統一的 Systems Manager 主控台來執行下列動作：

1. 執行手動或排定的診斷程序，確定帳戶或組織中的哪些 EC2 執行個體目前未由 Systems Manager 管理。
2. 確定會導致 Systems Manager 無法接管執行個體管理的網路問題或其他問題。
3. 執行自動化執行來自動修復問題，或存取資訊來協助自己手動解決問題。

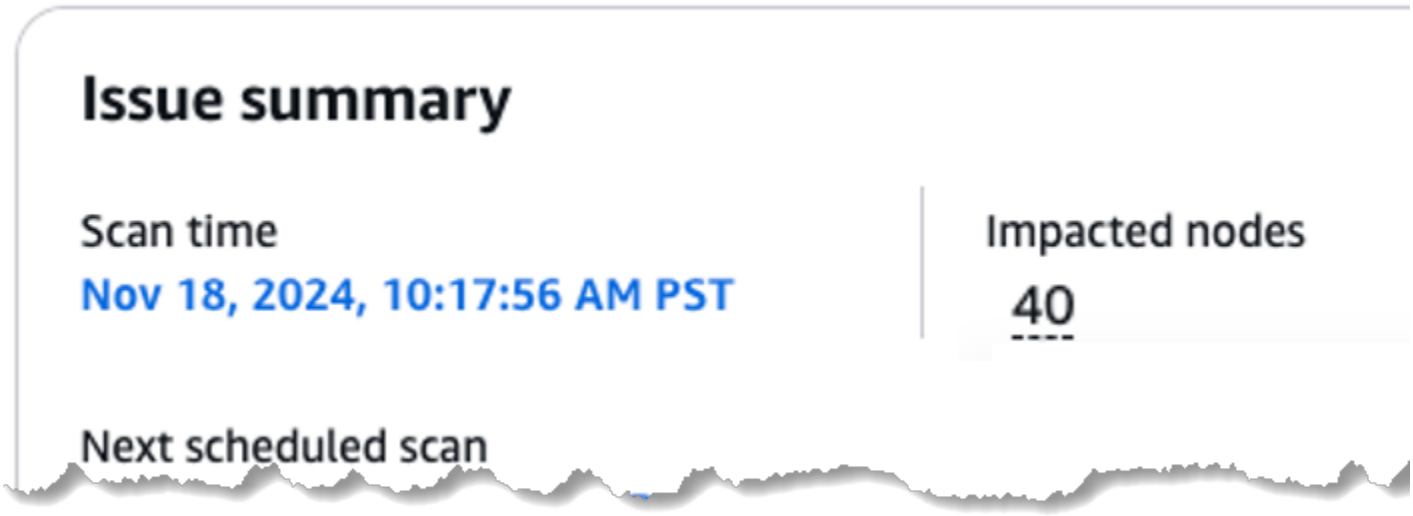
使用下列主題中的資訊，協助您診斷和修復會導致 Systems Manager 無法管理 EC2 執行個體的問題。

Systems Manager 如何計算「未受管 EC2 執行個體問題」清單受影響的節點數

在未受管 EC2 執行個體問題索引標籤上報告為未受管的節點數量，代表診斷掃描時具有下列任何狀態值的執行個體總數：

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

此數量在問題摘要區域中報告為受影響的節點數。在下列影像中，目前不受 Systems Manager 管理的受影響節點數量為 `40`。



與檢閱節點洞察頁面上未受管 EC2 執行個體的報告不同，此 EC2 執行個體計數並非動態。這代表在上次報告的診斷掃描期間獲得的調查結果，顯示為掃描時間值。因此，我們建議針對未受管 EC2 執行個體定期執行診斷掃描，讓受影響節點的此報告數量保持在最新狀態。

如需有關檢閱節點洞察頁面上未受管執行個體計數的資訊，請參閱主題 中的[什麼是未受管執行個體？檢閱節點洞察]。

主題

- 可診斷的未受管 EC2 執行個體問題的類別
- 為未受管 EC2 執行個體執行診斷和選用修復
- 排程未受管 EC2 執行個體的定期掃描

可診斷的未受管 EC2 執行個體問題的類別

本主題列出了 EC2 管理問題的主要類別，以及 Systems Manager 可協助您診斷和修復的每個類別的特定問題。請注意，對於某些問題，Systems Manager 可以發現問題，但不提供自動修復。在這些情況下，Systems Manager 主控台會將您引導至相關資訊，以協助您手動解決問題。

診斷程序會根據每個 EC2 執行個體群組所屬的虛擬私有雲端 (VPC)，一次檢查每組 EC2 執行個體。

問題類型

- 問題類別：安全群組組態和 HTTPS 通訊
- 問題類別：DNS 或 DNS 主機名稱組態
- 問題類別：VPC 端點組態

問題類別：安全群組組態和 HTTPS 通訊

診斷操作可能發現 SSM Agent 無法透過 HTTPS 與 Systems Manager 服務通訊。在這些情況下，可以選擇執行 Automation 執行手冊，嘗試更新連接至執行個體的安全群組。

> **Note**
>
> Systems Manager 有時可能無法自動修復這些問題，但您可以手動編輯受影響的安全群組。

支援的問題類型

- 執行個體安全群組：連接埠 443 上不允許傳出流量
- **ssm** VPC 端點的安全群組：連接埠 443 上不允許傳入流量
- **ssmmessages** VPC 端點的安全群組：連接埠 443 上不允許傳入流量
- **ec2messages** VPC 端點的安全群組：連接埠 443 上不允許傳入流量

如需詳細資訊，請參閱 SSM Agent 疑難排解 主題中的 驗證端點安全群組的相關輸入規則。

問題類別：DNS 或 DNS 主機名稱組態

診斷操作可能會發現未正確設定 VPC 的網域名稱系統 (DNS) 或 DNS 主機名稱。在這些情況下，可以選擇執行 Automation 執行手冊，嘗試啟用受影響 VPC 的 `enableDnsSupport` 和 `enableDnsHostnames` 屬性。

支援的問題類型

- VPC 中停用了 DNS 支援。
- VPC 中停用了 DNS 主機名稱。

如需詳細資訊，請參閱 SSM Agent 疑難排解 主題中的 驗證 VPC DNS 相關屬性。

問題類別：VPC 端點組態

診斷操作可能會發現未正確設定 VPC 的 VPC 端點。

如果 SSM Agent 所需的 VPC 端點不存在，Systems Manager 會嘗試執行 Automation 執行手冊來建立 VPC 端點，並將它們與各個相關區域可用區域 (AZ) 中的一個子網路進行關聯。如果存在所需的 VPC 端點，但端點未與問題所在的子網路關聯，則執行手冊會將 VPC 端點與受影響的子網路進行關聯。

> **Note**
>
> Systems Manager 不支援修復所有設定錯誤的 VPC 端點問題。在這些情況下，Systems Manager 會為您提供手動修復指示，而非執行 Automation 執行手冊。

支援的問題類型

- 找不到 PrivateLink 的 `ssm.region.amazonaws.com` 端點。
- 找不到 PrivateLink 的 `ssmmessages.region.amazonaws.com` 端點。
- 找不到 PrivateLink 的 `ec2messages.region.amazonaws.com` 端點。

可診斷的問題類型

Systems Manager 可以診斷下列問題類型，但目前沒有可用於修復這些問題的執行手冊。您可以針對這些問題手動編輯組態。

- 執行個體的子網路未連接至 `ssm.region.amazonaws.com` 端點。
- 執行個體的子網路未連接至 `ssmmessages.region.amazonaws.com` 端點。
- 執行個體的子網路未連接至 `ec2messages.region.amazonaws.com` 端點。

如需詳細資訊，請參閱 SSM Agent 疑難排解 主題中的 驗證 VPC 組態。

為未受管 EC2 執行個體執行診斷和選用修復

按照下列程序診斷網路相關問題和 VPC 相關問題，這些問題可能導致 Systems Manager 無法管理 EC2 執行個體。

該診斷操作可以偵測下列類型的問題，並將其分組：

- 網路組態問題：可能導致 EC2 執行個體無法與雲端中的 Systems Manager 服務通訊的網路問題類型。這些問題或許可以使用修復操作。如需有關網路組態問題的詳細資訊，請參閱可診斷的未受管 EC2 執行個體問題的類別。
- 無法辨識的問題：案例調查結果清單，其中列明無法透過診斷操作來判斷 EC2 執行個體為何無法與雲端中的 Systems Manager 服務通訊。

為未受管 EC2 執行個體執行診斷和修復

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇診斷和修復。
3. 選擇未受管 EC2 執行個體問題索引標籤。
4. 在問題摘要區段中，選擇執行新診斷。

   -或-

   如果這是第一次診斷未受管 EC2 問題，請在診斷未受管 EC2 執行個體區段中選擇執行。

   > **Tip**
   >
   > 在診斷正在執行時，選擇檢視進度或檢視執行來監控執行的目前狀態。如需詳細資訊，請參閱檢視 Systems Manager 中修復的執行進度和歷史記錄。

5. 診斷完成後，請執行下列動作：
   - 對於無法辨識的問題區段中報告的任何問題，請選擇進一步了解連結取得解決問題的相關資訊。
   - 對於網路組態問題區段中報告的問題，請繼續下一個步驟。
6. 在調查結果類型清單的建議欄位中，針對特定問題選擇連結，例如 2 條建議。
7. 在開啟的建議窗格中，從可用的緩解措施中進行選擇：
   - 進一步了解：打開一個主題，其中包含如何手動解決問題的相關資訊。

- 檢視執行手冊：打開一個窗格，其中包含有關可執行以解決 EC2 執行個體問題的 Automation 執行手冊的資訊，以及產生執行手冊將採取之動作預覽的選項。繼續下一個步驟。

8. 在執行手冊窗格中，執行下列動作：

   a. 對於文件描述，請檢閱內容，其中概述了執行手冊可以採取哪些動作來修復未受管 EC2 執行個體問題。選擇檢視步驟，預覽執行手冊會採取的個別動作。

   b. 對於 Targets (目標)，執行下列動作：

      - 如果正在管理組織的修復，請在帳戶欄位中指定此執行手冊是針對所有帳戶，還是只針對所選帳戶子集。
      - 對於區域，指定此 Runbook 是否將 AWS 區域 鎖定您帳戶或組織中的所有 ，或僅鎖定您選擇的區域子集。

   c. 對於執行手冊預覽，請仔細檢閱資訊。這些資訊會說明，如果選擇執行執行手冊，範圍和影響會如何。

      > **Note**
      >
      > 選擇執行執行手冊會產生費用。請仔細檢閱預覽資訊，再決定是否繼續。

      執行手冊預覽內容提供下列資訊：

      - 執行手冊操作將在多少個區域中發生。
      - (僅限組織) 操作將在多少個組織單位 (OU) 中執行。
      - 要採取的動作類型，以及每種動作的數量。

        動作類型如下：

        - 變更：執行手冊步驟會透過建立、修改或刪除資源的動作來變更目標。
        - 不變更：執行手冊步驟會擷取有關資源的資料，但不會變更資料。此類別通常包含 `Describe*`、`List*`、`Get*` 和類似的唯讀 API 動作。
        - 未確定：未確定的步驟會叫用由另一個協調服務執行的執行 AWS Lambda，例如 AWS Step Functions或 AWS Systems Manager Run Command。未確定的步驟也可能會呼叫第三方 API。Systems Manager Automation 不知道協同運作程序或第三方 API 執行的結果，因此步驟的結果未確定。

   d. 此時可以執行以下其中一個動作：

- 停止且不要執行執行手冊。
- 選擇執行，使用已選取的選項執行執行手冊。

如果選擇執行該操作，請選擇檢視進度或檢視執行來監控執行的目前狀態。如需詳細資訊，請參閱檢視 Systems Manager 中修復的執行進度和歷史記錄。

排程未受管 EC2 執行個體的定期掃描

在帳戶或組織中，可以針對 Systems Manager 因各種組態問題而無法管理的 Amazon EC2 執行個體執行隨需掃描。此掃描也可排程為定期自動執行。

排程未受管 EC2 執行個體的定期掃描

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇診斷和修復。
3. 選擇未受管 EC2 執行個體問題索引標籤。
4. 在診斷未受管 EC2 執行個體區段中，開啟排程定期診斷。
5. 在診斷頻率欄位中，選取是每天還是每週執行一次診斷。
6. (選用) 在開始時間欄位中，為要開始的診斷輸入 24 小時格式的時間。例如，若是晚上 8:15，則輸入 **`20:15`**。

   輸入的時間適用於目前的當地時區。

   如果未指定時間，診斷掃描會立即執行。Systems Manager 也可在目前時間排程要在未來執行的掃描。如果指定時間，Systems Manager 會等待在指定時間執行診斷掃描。
7. 選擇 Execute (執行)。診斷會立即執行，但也會依指定的排程執行。

執行手冊動作的修復影響類型

Systems Manager 可以執行診斷操作，來探索某些類型的失敗部署和漂移組態，以及會導致 Systems Manager 無法管理 EC2 執行個體的某些類型的組態問題。診斷結果可能包括對 Automation 執行手冊的建議，您可以執行這些建議來嘗試修復問題。如需有關這些診斷操作的詳細資訊，請參閱下列主題：

- 診斷和修復失敗部署

- 診斷和修復漂移組態
- 在 Systems Manager 中診斷和修復未受管的 Amazon EC2 執行個體

如果 Systems Manager 發現可透過在受影響的資源上執行 Automation 執行手冊來修正的問題，即會為您提供執行預覽。執行預覽提供執行手冊執行會對目標所做的變更類型的相關資訊。此資訊包含診斷發現的三種變更類型中每種變更的數量。

這些變更類型所示如下：

- `Mutating`：執行手冊步驟會透過建立、修改或刪除資源的動作來變更目標。
- `Non-Mutating`：執行手冊步驟會擷取有關資源的資料，但不會變更資料。此類別通常包含 `Describe*`、`List*`、`Get*` 和類似的唯讀 API 動作。
- `Undetermined`：未決定的步驟會叫用由另一個協調服務執行的執行 AWS Lambda AWS Step Functions，例如 或 Run Command，也就是 中的工具 AWS Systems Manager。未確定的步驟也可能呼叫第三方 API 或執行 Python 或 PowerShell 指令碼。Systems Manager Automation 無法偵測協同運作程序或第三方 API 執行的結果，因此無法進行評估。必須手動檢閱這些步驟的結果，才能判斷其影響。

  如需有關支援的自動化動作影響類型的資訊，請參閱下列資料表。

支援的修復動作的影響類型

該資料表顯示了修復執行手冊中可以包含的各種動作的影響類型：變更、不變更和未確定。

動作[1]	影響類型
aws:approve	不變更
aws:assertAwsResourceProperty	不變更
aws:branch	不變更
aws:changeInstanceState	變動
aws:copyImage	變動
aws:createImage	變動

動作[1]	影響類型
aws:createStack	變動
aws:createTags	變動
aws:deleteImage	變動
aws:deleteStack	變動
aws:executeAutomation	Undetermined (未決定)
aws:executeAwsApi	Undetermined (未決定)
aws:executeScript	Undetermined (未決定)
aws:executeStateMachine	Undetermined (未決定)
aws:invokeLambdaFunction	Undetermined (未決定)
aws:invokeWebhook	Undetermined (未決定)
aws:loop	各有不同。取決於迴圈中的動作。
aws:pause	不變更
aws:runCommand	Undetermined (未決定)
aws:runInstances	變動
aws:sleep	不變更
aws:updateVariable	變動
aws:waitForAwsResourceProperty	不變更

[1] 如需有關自動化動作的詳細資訊，請參閱 Systems Manager Automation 動作參考。

檢視 Systems Manager 中修復的執行進度和歷史記錄

您可以檢視使用 Systems Manager 中的診斷和修復功能進行的所有進行中和已完成修復操作的清單。

執行歷史記錄清單中的資料會報告下列類型的資訊：

- 執行的類型：`Diagnosis` 或 `Remediation`。
- 執行狀態，例如 `Success` 或 `Failed`。
- 執行開始和結束的時間。

檢視修復的執行進度和歷史記錄

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇診斷和修復。
3. 選擇檢視執行。

   > **Tip**
   >
   > 執行正在執行中時，也可以選擇檢視進度來開啟執行歷史記錄頁面。

4. (選用) 在搜尋
   ( )
   方塊中，輸入片語以協助縮小執行清單範圍，例如 **EC2** 或 **VPC**。
5. (選用) 若要檢視執行的其他詳細資訊，請在執行名稱欄位中選擇操作名稱，例如 AWS-DiagnoseUnmanagedEC2NetworkIssues。

   在詳細資訊窗格中，可以檢視有關操作期間嘗試的所有步驟的資訊，以及執行的所有輸入和輸出的資訊。

調整 Systems Manager 設定

設定頁面上的選項會在 Systems Manager 統一主控台中啟用和設定功能。顯示的選項取決於您登入的帳戶，以及您是否已經完成 Systems Manager 的設定。

> **Note**
>
> 設定頁面上的選項不會影響 Systems Manager 工具 (先前稱為功能)。

帳戶設定

如果已啟用 Systems Manager，且您登入的帳戶不是 Organizations 的成員，或委派管理員尚未將 Organizations 帳戶新增至 Systems Manager，則帳戶設定頁面會顯示停用 Systems Manager 的選項。停用 Systems Manager 表示 Systems Manager 不會顯示統一主控台。所有 Systems Manager 工具仍可運作。

組織設定

在組織設定索引標籤上，主要區域區段會顯示 AWS 區域 選擇為設定期間主要區域的 。在使用 的多帳戶和多區域環境中 AWS Organizations，Systems Manager 會自動將所有帳戶和區域的節點資料彙總至主要區域。以這種方式彙總資料，可讓您在單一位置跨帳戶和區域檢視節點資料。

> **Note**
>
> 若想變更主區域，則必須停用 Systems Manager，然後再次將其啟用。若要停用 Systems Manager，請選擇停用。

組織設定區段會顯示組織單位， AWS 並在設定期間 AWS 區域 選擇。若要變更哪些組織單位和區域會在 Systems Manager 中顯示節點資料，請選擇編輯。如需有關為組織設定 Systems Manager 的詳細資訊，請參閱設定 AWS Systems Manager。

功能組態

功能組態區段可讓您啟用和設定關鍵 Systems Manager 功能，以增強整個組織的節點管理。這些功能可共同運作，為您的受管節點提供自動化管理、合規監控和維護。

您可以在初始 Systems Manager 設定期間設定這些功能，或稍後透過設定頁面進行修改。您可以根據您組織的需求，獨立啟用或停用每個功能。

預設主機管理組態

預設主機管理組態 (DHMC) 會自動將組織中的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體設定為由 Systems Manager 管理。啟用時，DHMC 會確保新的和現有的 EC2 執行個體具有與 Systems Manager 服務通訊所需的 AWS Identity and Access Management (IAM) 許可和組態。

DHMC 提供下列優點：

- 自動 IAM 角色指派 - 確保 EC2 執行個體具有必要的 IAM 角色和政策，以做為受管節點運作
- 偏離修復 - 當執行個體失去其受管節點狀態時，自動更正組態偏離
- 簡化加入 - 減少新執行個體的手動組態步驟
- 一致的組態 - 維持 EC2 機群的統一設定

設定偏離修復頻率

當 EC2 執行個體遺失其受管節點組態時，漂移修復會自動偵測和更正。您可以設定 Systems Manager 檢查和修復組態偏離的頻率。

設定預設主機管理組態

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇設定。
3. 在功能組態區段中，找到預設主機管理組態。
4. 若要啟用 DHMC，請開啟切換開關。
5. 針對偏離修復頻率，選擇您希望 Systems Manager 檢查和修復組態偏離的頻率：
   - 每日 - 每天檢查並修復漂移一次
   - 每週 - 每週檢查並修復漂移一次
   - 每月 - 每月檢查並修復漂移一次
6. 選擇儲存。

> **Note**
>
> 當您啟用 DHMC 時，Systems Manager 會在您的帳戶中建立必要的 IAM 角色和政策。這些角色允許 EC2 執行個體與 Systems Manager 服務通訊。如需 DHMC 建立之 IAM 角色的詳細資訊，請參閱 使用 Systems Manager 管理 EC2 執行個體。

庫存中繼資料收集

清查中繼資料收集會自動收集受管節點的詳細資訊，包括已安裝的應用程式、網路組態、系統更新和其他系統中繼資料。此資訊可協助您維持合規性、執行安全分析，以及了解您的基礎設施組成。

庫存集合提供下列優點：

- 合規監控 - 追蹤已安裝的軟體和組態以進行合規報告
- 安全分析 - 識別過期的軟體和潛在的安全漏洞
- 資產管理 - 維護基礎設施up-to-date庫存
- 查詢功能 - 將收集的資料與 Amazon Q Developer 搭配使用以進行自然語言查詢

收集的庫存資料類型

啟用清查中繼資料收集時，Systems Manager 會從受管節點收集下列類型的資訊：

- 應用程式 - 已安裝的軟體套件和應用程式
- 網路組態 - 網路介面、IP 地址和網路設定
- 系統更新 - 已安裝的修補程式和可用的更新
- 系統屬性 - 硬體規格、作業系統詳細資訊和系統組態
- 服務 - 執行服務及其組態

設定庫存收集頻率

您可以設定 Systems Manager 從受管節點收集庫存中繼資料的頻率。較頻繁的集合提供up-to-date資訊，但可能會增加 AWS 服務用量。

設定庫存中繼資料收集

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇設定。
3. 在功能組態區段中，尋找庫存中繼資料集合。
4. 若要啟用庫存收集，請開啟切換開關。
5. 針對收集頻率，選擇您希望 Systems Manager 收集庫存資料的頻率：
   - 每日 - 每天收集一次庫存資料
   - 每週 - 每週收集一次庫存資料
   - 每月 - 每月收集一次庫存資料
6. 選擇儲存。

> **Important**
>
> 庫存收集需要受管節點具備必要的許可，才能收集系統資訊。確保您的受管節點具有適當的 IAM 角色和政策。如需所需許可的相關資訊，請參閱AWS Systems Manager庫存。

SSM Agent 更新

自動SSM Agent更新可確保您的受管節點執行的是最新版本的 SSM Agent。讓代理程式保持在up-to-date，可讓您存取最新的功能、安全性改善和錯誤修正。

SSM Agent 自動更新提供下列優點：

- 最新功能 - 存取新的 Systems Manager 功能和改進
- 安全性更新 - 自動安裝安全性修補程式和修正
- 改善可靠性 - 錯誤修正和穩定性改善
- 減少維護 - 無需手動更新代理程式

設定自動代理程式更新

您可以設定 Systems Manager 在受管節點上檢查和安裝SSM Agent更新的頻率。定期更新有助於確保最佳效能和安全性。

設定SSM Agent更新

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇設定。
3. 在功能組態區段中，尋找SSM Agent更新。
4. 若要啟用自動更新，請開啟切換開關。
5. 針對更新頻率，選擇您希望 Systems Manager 檢查並安裝代理程式更新的頻率：
   - 每日 - 每天檢查更新一次
   - 每週 - 每週檢查更新一次
   - 每月 - 每月檢查更新一次
6. 選擇儲存。

診斷和修復設定

診斷和修復設定會判斷 Systems Manager 是否自動掃描節點，確保節點可以與 Systems Manager 通訊。如果已啟用，則此功能會根據您定義的排程自動執行。該功能會識別哪些節點無法連線至 Systems Manager 及其原因。此外，此功能提供建議的執行手冊，可修復聯網問題和其他會防止將節點設為受管節點的問題。

安排定期診斷掃描

Systems Manager 可以診斷並協助您修復多種類型的部署失敗和漂移組態。Systems Manager 也可以識別帳戶或組織中 Systems Manager 無法視為受管節點的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。EC2 執行個體診斷程序可以識別與虛擬私有雲端 (VPC)、網域名稱服務 (DNS) 設定或 Amazon Elastic Compute Cloud (Amazon EC2) 安全群組的錯誤組態相關的問題。

若要僅識別無法連線至 Systems Manager 的節點，排程定期診斷功能可讓您將定期診斷掃描自動化。這些掃描有助於識別哪些節點無法連線至 Systems Manager 及其原因。按照下列程序啟用和設定節點的定期診斷掃描。

安排定期診斷掃描

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中選擇設定，然後選擇診斷和修復索引標籤。
3. 開啟排程定期診斷選項。
4. 在掃描期間欄位中選擇掃描的執行頻率。
5. (選用) 在開始時間欄位中，為要開始的診斷輸入 24 小時格式的時間。例如，若是晚上 8:15，則輸入 **`20:15`**。

   輸入的時間適用於目前的當地時區。

   如果未指定時間，診斷掃描會立即執行。Systems Manager 也可在目前時間排程要在未來執行的掃描。如果指定時間，Systems Manager 會等待在指定時間執行診斷掃描。
6. 選擇儲存。
7. 掃描完成後，請在左側導覽中選擇診斷和修復來檢視詳細資訊。

如需有關診斷和修復功能的詳細資訊，請參閱診斷和修復。

更新 S3 儲存貯體加密

當您加入 Systems Manager 時，快速設定會在委派的管理員帳戶中建立 Amazon Simple Storage Service (Amazon S3) 儲存貯體以進行 AWS Organizations 設定。對於單一帳戶設定，系統會將此儲存貯體存放在要設定的帳戶中。此儲存貯體會用於存放診斷掃描期間產生的中繼資料。

如需設定統一 Systems Manager 主控台的詳細資訊，請參閱 設定 AWS Systems Manager。

根據預設，您儲存貯體中的資料會使用為您 AWS 擁有和管理的 AWS Key Management Service (AWS KMS) 金鑰進行加密。

您可以選擇使用不同的 AWS KMS 金鑰進行儲存貯體加密。或者，您可以使用客戶受管金鑰 AWS KMS keys (CMK) 搭配 (SSE-KMS) 使用伺服器端加密。如需相關資訊，請參閱使用 Systems Manager 的 Amazon S3 儲存貯體和儲存貯體政策。

使用不同的 AWS KMS 金鑰進行 S3 儲存貯體加密

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中選擇設定，然後選擇診斷和修復索引標籤。
3. 在更新 S3 儲存貯體加密區域中選擇編輯。
4. 選取自訂加密設定 (進階) 核取方塊。
5. 針對選擇 AWS KMS 金鑰，選擇或輸入金鑰的 Amazon Resource Name (ARN)。

   > **Tip**
   >
   > 若要建立新的金鑰，請選擇建立 AWS KMS 金鑰。

6. 選擇儲存。

使用 Systems Manager 的 Amazon S3 儲存貯體和儲存貯體政策

在 的加入程序中 AWS Systems Manager， 會在委派管理員帳戶中Quick Setup建立 Amazon Simple Storage Service (Amazon S3) 儲存貯體，以進行組織設定。對於單一帳戶設定，系統會將此儲存貯體存放在要設定的帳戶中。

您可以使用 Systems Manager 在機群上執行診斷操作，識別失敗部署和漂移組態的案例。Systems Manager 也可以偵測以下情況：因組態問題，導致 Systems Manager 無法管理帳戶或組織中的 EC2

執行個體。這些診斷操作的結果會存放在此 Amazon S3 儲存貯體中，同時受到加密方法和 S3 儲存貯體政策的保護。如需有關將資料輸出到此儲存貯體的診斷操作的資訊，請參閱診斷和修復。

變更儲存貯體加密方法

依預設，S3 儲存貯體會將伺服器端加密與 Amazon S3 受管金鑰 (SSE-S3) 搭配使用。

您可以改為使用伺服器端加密搭配 AWS KMS keys (SSE-KMS)，使用客戶受管金鑰 (CMK) 做為 Amazon S3 受管金鑰的替代方案，如 所述變更為 AWS KMS 客戶受管金鑰以加密 S3 資源。

儲存貯體政策的內容

此儲存貯體政策可防止組織中的成員帳戶彼此探索。只有為 Systems Manager 建立的診斷和修復角色，才擁有對儲存貯體的讀取和寫入許可。這些系統產生的政策內容會顯示在 統一 Systems Manager 主控台的 S3 儲存貯體政策 中。

> **Warning**
>
> 修改預設儲存貯體政策，可能會允許組織中的成員帳戶彼此探索，或讀取其他帳戶中執行個體的診斷輸出。如果選擇修改此政策，建議要謹慎。

主題

- 變更為 AWS KMS 客戶受管金鑰以加密 S3 資源
- 統一 Systems Manager 主控台的 S3 儲存貯體政策

變更為 AWS KMS 客戶受管金鑰以加密 S3 資源

在統一 Systems Manager 主控台的加入程序中， 會在委派管理員帳戶中Quick Setup建立 Amazon Simple Storage Service (Amazon S3) 儲存貯體。此儲存貯體用於存放修復執行手冊執行期間產生的診斷輸出資料。依預設，此儲存貯體將伺服器端加密與 Amazon S3 受管金鑰 (SSE-S3) 搭配使用。

您可以在統一 Systems Manager 主控台的 S3 儲存貯體政策中檢閱這些政策的內容。

不過，您可以使用客戶受管金鑰 AWS KMS keys (CMK) 做為 的替代方案，將伺服器端加密與 (SSE-KMS) 搭配使用 AWS KMS key。

要設定 Systems Manager 來使用 CMK，請完成下列任務。

任務 1：將標籤新增至現有 CMK

AWS Systems Manager 只有在您的 CMK 加上下列鍵值對的標籤時， 才會使用您的 CMK：

- 索引鍵：`SystemsManagerManaged`
- 值：`true`

按照下列程序提供使用 CMK 加密 S3 儲存貯體的存取權。

將標籤新增至現有 CMK

1. 開啟 AWS KMS 主控台，網址為 https：//https://console.aws.amazon.com/kms。
2. 在左側導覽列中，選擇客戶自管金鑰。
3. 選取要 AWS KMS key 與 搭配使用的 AWS Systems Manager。
4. 選擇標籤索引標籤，然後選擇編輯。
5. 選擇 Add tag (新增標籤)。
6. 請執行下列操作：
   a. 對於 Tag key (標籤索引鍵)，輸入 **`SystemsManagerManaged`**。
   b. 在標籤值欄位中輸入 **`true`**。
7. 選擇儲存。

任務 2：修改現有 CMK 金鑰政策

使用下列程序來更新 CMK 的 KMS 金鑰政策，以允許 AWS Systems Manager 角色代表您加密 S3 儲存貯體。

修改現有 CMK 金鑰政策

1. 開啟 AWS KMS 主控台，網址為 https：//https://console.aws.amazon.com/kms。
2. 在左側導覽列中，選擇客戶自管金鑰。
3. 選取要 AWS KMS key 與 搭配使用的 AWS Systems Manager。
4. 在金鑰政策標籤中，選擇編輯。
5. 將下列 JSON 陳述式新增至 `Statement` 欄位，再使用自己的資訊取代`#####`。

   請確定您在 AWS Systems Manager `Principal` 欄位中將組織中加入的所有 AWS 帳戶 IDs 新增至 。

若要在 Amazon S3 主控台中找到正確的儲存貯體名稱，請在委派管理員帳戶中找到如下格式的儲存貯體：do-not-delete-ssm-*operational-account-id*-*home-region*-*disambiguator*。

```
{
     "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 條件金鑰來更新 CMK 金鑰政策，以授予 CMK 的 AWS Systems Manager 存取權。

任務 3：在 Systems Manager 設定中指定 CMK

完成前兩個任務後，按照下列程序變更 S3 儲存貯體加密。此變更可確保相關聯的 Quick Setup 組態程序可以新增許可，讓 Systems Manager 接受 CMK。

1. 開啟 AWS Systems Manager 主控台，網址為 https：//https://console.aws.amazon.com/systems-manager/。
2. 在導覽窗格中，選擇設定。
3. 在診斷和修復索引標籤上，從更新 S3 儲存貯體加密區段中選擇編輯。
4. 選取自訂加密設定 (進階) 核取方塊。
5. 在搜尋
   ( )
   方塊中，選擇現有金鑰的 ID，或貼上現有金鑰的 ARN。
6. 選擇儲存。

統一 Systems Manager 主控台的 S3 儲存貯體政策

本主題包含當您將組織或單一帳戶加入統一的 Systems Manager 主控台時，由 Systems Manager 建立的 Amazon S3 儲存貯體政策。

> **Warning**
>
> 修改預設儲存貯體政策，可能會允許組織中的成員帳戶彼此探索，或讀取其他帳戶中執行個體的診斷輸出。如果選擇修改此政策，建議要謹慎。

適用於組織的 Amazon S3 儲存貯體政策

將組織加入 Systems Manager 時，系統會使用下列預設儲存貯體政策建立診斷儲存貯體。

```
{
    "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"
                }
            }
        }
    ]
}
```

適用於單一帳戶的 Amazon S3 儲存貯體政策

將單一帳戶加入 Systems Manager 時，系統會使用下列預設儲存貯體政策建立診斷儲存貯體。

```
{
  "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 會將工具分組為四個類別。下列文件說明 的各種工具， AWS Systems Manager 以及如何設定和使用這些工具。選擇各類別下方的索引標籤，深入解每個工具。

節點工具

受管節點是設定為在混合多雲端環境中搭配 Systems Manager 使用的任何機器。

Compliance

您可以使用合規功能來掃描受管節點機群，以檢查修補程式合規與組態的不一致。您可以從多個 AWS 帳戶 和 收集和彙總資料 AWS 區域，然後深入探討不合規的特定資源。根據預設，合規會顯示有關 Patch Manager 修補和 State Manager 關聯的合規資料。您也可以根據 IT 或業務的需求，來自訂服務和建立自己的合規類型。

Distributor

使用 Distributor，以建立套件並將其部署到受管節點。使用 Distributor，您可以封裝自己的軟體，或尋找 AWS提供的代理程式軟體套件，例如 AmazonCloudWatchAgent，以在 Systems Manager 受管節點上安裝 。第一次安裝套件之後，您可以使用 Distributor 解除安裝並重新安裝新的套件版本，或執行只會加入新增或變更檔案的就地更新。Distributor 會將資源 (例如軟體套件) 發佈至 Systems Manager 受管理的節點。

Fleet Manager

Fleet Manager 是統一的使用者介面 (UI) 體驗，可助您遠端管理節點。利用 Fleet Manager，您可以從單一主控台檢視整個機群的運作狀態和效能狀態。您也可以從個別裝置和執行個體收集資料，進而從主控台執行常見的故障診斷和管理任務。這包括檢視目錄和檔案內容、Windows 登錄管理、作業系統使用者管理等。

Hybrid Activations

若要將混合多雲端環境中的非 EC2 機器設定為受管節點，請建立混合啟用。完成啟用後，您會收到一組啟用代碼和 ID。這個程式碼/ID 組合的函數就像 Amazon Elastic Compute Cloud (Amazon EC2) 存取 ID 和秘密金鑰，可讓您從受管執行個體安全存取 Systems Manager 服務。

如果您想要使用 Systems Manager 來管理邊緣裝置，則也可以建立邊緣裝置的啟用。

Inventory

清查功能會自動化從受管節點收集軟體庫存的程序。您可以使用庫存在受管執行個體上蒐集有關應用程式、檔案、元件、修補程式等。

修補程式管理員

使用 Patch Manager 以透過安全相關和其他類型的更新，來自動化修補您受管節點的程序。您可以使用 Patch Manager 以套用適用於作業系統和應用程式的修補程式。(在 Windows Server 上，應用程式支援僅限於由 Microsoft 發行的應用程式更新。)

此工具可讓您掃描受管節點是否有遺失的修補程式，並使用標籤將個別遺失的修補程式套用至大型受管節點群組。 Patch Manager使用修補程式基準，其中包括在修補程式發行後 天內自動核准修補程式的規則，以及已核准和遭拒的修補程式清單。您可以藉由將修補排定為以 Systems Manager 維護時段任務執行，定期安裝安全修補程式，或者您可以隨時隨需修補您的受管節點。

至於 Linux 作業系統，您可以在修補基準中，定義要用於修補操作的儲存庫。這樣便能確定不管受管節點上設定了哪些儲存庫，都只會從信任的儲存庫安裝更新。至於 Linux，您也能夠更新受管節點上的任何套件，而不只是歸類為作業系統安全更新的套件。您也可以產生可傳送至所選 S3 儲存貯體的修補程式報告。對於單一受管節點，報告包括機器所有修補程式的詳細資訊。對於所有受管節點的報告，只會提供缺少修補程式數量的摘要。

Run Command

使用 Run Command，以從遠端安全地大規模管理受管節點的組態。使用 Run Command 執行隨需變更，例如，在擁有數十或數百個受管節點的目標集上，更新應用程式或執行 Linux shell 指令碼和 Windows PowerShell 命令。

工作階段管理員

使用 透過互動式一鍵式瀏覽器型 shell 或透過 Session Manager來管理您的邊緣裝置和 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體 AWS CLI。 Session Manager提供安全且可稽核的邊緣裝置和執行個體管理，而無需開啟傳入連接埠。 維護堡壘主機、 或 管理 SSH 金鑰。 Session Manager也可讓您遵守需要受控制存取邊緣裝置和執行個體的公司政策， 嚴格的安全實務， 可完整稽核的日誌，其中包含邊緣裝置和執行個體存取詳細資訊， 同時仍為最終使用者提供對邊緣裝置和 EC2 執行個體的簡單一鍵式跨平台存取。若要使用 Session Manager，您必須啟用進階執行個體層。如需詳細資訊，請參閱開啟 advanced-instances 方案。

State Manager

使用 State Manager，以自動化讓受管節點維持在定義狀態的程序。您可以使用 State Manager 來保證受管節點在啟動時由特定軟體引導、加入 Windows 網域 (僅限 Windows Server 節點)，或使用特定軟體更新來進行修補。

變更管理工具

Automation

使用 Automation 來自動化一般維護與部署任務。您可以使用自動化來建立和更新 Amazon Machine Images (AMIs)、套用驅動程式和代理程式更新、重設 Windows Server 執行個體的密碼、重設 Linux 執行個體的 SSH 金鑰，以及套用 OS 修補程式或應用程式更新。

變更行事曆

Change Calendar 可協助您為指定的動作 (例如在 Systems Manager Automation Runbook 中) 設定要或不要在 AWS 帳戶中執行的日期與時間範圍。在 Change Calendar 中，這些範圍稱為「事件」。當您建立了 Change Calendar 項目，也會建立類型 `ChangeCalendar` 的 Systems Manager 文件。在 Change Calendar 中，這些文件會以純文字格式儲存 iCalendar 2.0 資料。您新增到 Change Calendar 項目的事件將成為文件的一部份。您可以在 Change Calendar 界面中手動新增事件，或使用 `.ics` 檔案從支援的第三方行事曆中匯入事件。

Change Manager

Change Manager 是一個企業變更管理架構，用於請求、核准、實作和報告應用程式組態和基礎設施的操作變更。從單一委派管理員帳戶，如果您使用 AWS Organizations，則可以管理多個 AWS 帳戶 中多個 的變更 AWS 區域。或使用本機帳戶，您可以管理單一 AWS 帳戶的變更。使用 Change Manager 來管理 AWS 資源和內部部署資源的變更。

Documents

Systems Manager 文件 (SSM 文件) 定義 Systems Manager 在受管執行個體上執行的動作。SSM 文件類型包含命令文件 (由 State Manager 和Run Command 使用) 和自動化 Runbook (由 Systems Manager 自動化使用)。Systems Manager 包含數十種預先設定的文件，可讓您用來在執行時間時指定參數。文件可以使用 JSON 或 YAML 格式表示，並包含您指定的步驟和參數。

維護時段

使用 Maintenance Windows，為受管執行個體安排各種管理任務的定期執行排程，例如安裝修補程式和更新，而不會中斷業務關鍵操作。

快速設定

使用 Quick Setup 設定常用 AWS 服務 和具有建議最佳實務的功能。您可以透過與 AWS 區域 整合，Quick Setup在個別 AWS 帳戶 或跨多個 AWS 帳戶 和 中使用 AWS Organizations。 透過自動化常見或建議的任務，Quick Setup簡化包括 Systems Manager 在內的服務設定。這些任務包括建立必要的 AWS Identity and Access Management (IAM) 執行個體描述檔角色和設定操作最佳實務，例如定期修補程式掃描和庫存收集。

應用程式工具

AppConfig

AppConfig 可協助您建立、管理及部署應用程式組態與功能標記。AppConfig 支援對任何大小應用程式的受控制部署。您可以將 AppConfig與託管在 Amazon EC2 執行個體、 AWS Lambda 容器、行動應用程式或邊緣裝置上的應用程式搭配使用。為了防止部署應用程式組態時發生錯誤，AppConfig 包括了驗證器。驗證器提供了一個語法或語義檢查，以確認您要部署的組態可如預期運作。AppConfig 會在組態部署期間監視應用程式，以確認部署成功。如果系統遇到錯誤或部署叫用警示，AppConfig 會復原變更，以將對應用程式使用者的影響降到最低。

Application Manager

Application Manager 協助 DevOps 工程師在其應用程式和叢集的內容中調查和修復其 AWS 資源的問題。在 中Application Manager，應用程式是您想要做為單位運作的邏輯 AWS 資源群組。此邏輯群組可以代表應用程式的不同版本、運算子或開發人員環境的擁有權界限等等。 對容器叢集的Application Manager支援包括 Amazon Elastic Kubernetes Service (Amazon EKS) 和 Amazon Elastic Container Service (Amazon ECS) 叢集。 會將多個 AWS 服務 和 Systems Manager 工具的操作資訊Application Manager彙總到單一 AWS Management Console叢集。

參數存放區

Parameter Store 會提供安全的階層式儲存空間，以供組態資料管理和秘密管理。您可以將密碼、資料庫字串、Amazon Elastic Compute Cloud (Amazon EC2) 執行個體 ID 和 Amazon Machine Image (AMI) ID，以及授權碼之類的資料存放為參數值。您存放的值可以是純文字或加密資料。然後，您可以使用建立參數時指定的唯一名稱來參考各個值。

操作工具

CloudWatch Dashboards

Amazon CloudWatch 儀表板是 CloudWatch 主控台中可自訂的頁面，可讓您在單一檢視中監控資源，甚至是分散在不同區域的那些資源。您可以使用 CloudWatch 儀表板來為 AWS 資源的指標和警示建立自訂檢視。

Explorer

Explorer 是可自訂的操作儀表板，可報告 AWS 資源的相關資訊。 Explorer會顯示您 AWS 帳戶和 之間的操作資料 (OpsData) 彙總檢視 AWS 區域。在 Explorer 中，OpsData 包含有關 Amazon EC2 執行個體、修補程式合規詳細資訊和操作工作項目 (OpsItems) 的中繼資料。Explorer 提供脈

絡指出 OpsItems 如何分散於業務單位或應用程式、如何隨著時間而演變趨勢，以及如何隨類別而變化。您可以在 Explorer 中群組和篩選資訊，以專注於與您相關且需要採取動作的項目。當您識別高優先順序問題時，您可以使用 Systems Manager 中的OpsCenter工具 來執行 Automation Runbook 並解決這些問題。

Incident Manager

Incident Manager 是一種事件管理主控台，可協助使用者減輕影響其 AWS 託管應用程式的事件並從中復原。

Incident Manager 可通知回應方相關影響、反白相關的故障診斷資料，並提供協同合作工具來備份和執行服務，藉此增強事件解決方案。Incident Manager 也會自動化回應計劃，並允許回應方團隊上報。

OpsCenter

OpsCenter 提供中央位置，讓營運工程師和IT專業人員可以檢視、調查和解決與 AWS 資源相關的操作工作項目 (OpsItems)。 OpsCenter 旨在減少對影響 AWS 資源的問題進行解決的平均時間。此 Systems Manager 工具會OpsItems跨服務彙總和標準化OpsItems，同時提供每個 OpsItem、相關和相關資源的相關內容調查資料。 OpsCenter也提供 Systems Manager Automation Runbook，可讓您用來解決問題。您可以為每個 OpsItem 指定可搜尋的自訂資料。您也可以依狀態和來源，檢視自動產生的 OpsItems 摘要報告。

使用 Systems Manager 執行節點任務

使用此教學課程來開始使用 AWS Systems Manager。您將學到如何啟動由 Systems Manager 管理的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體，以及如何連線到受管執行個體。

由於 Systems Manager 是多個工具的集合，因此沒有任何單一演練或教學課程可以介紹整個服務。本教學課程提供一些工具的簡介。

先決條件

開始之前，請務必先完成 使用 Systems Manager 管理 EC2 執行個體 中的步驟。

使用預先安裝有 SSM Agent 的 AMI 啟動執行個體

您可以使用 啟動 Amazon EC2 執行個體 AWS Management Console ，如下列程序所述。本教學課程旨在協助您快速啟動第一個受管執行個體，所以未涵蓋全部的可能選項。

啟動執行個體

1. 在 https://console.aws.amazon.com/ec2/ 開啟 Amazon EC2 主控台。
2. 在 EC2 主控台儀表板的 Launch instance (啟動執行個體) 方塊中，選擇 Launch instance (啟動執行個體)，然後從出現的選項中選擇 Launch instance (啟動執行個體)。
3. 在名稱和標籤下的名稱中，輸入執行個體的描述性名稱。
4. 在應用程式和作業系統映像 (Amazon Machine Image) 下，執行下列動作：
   a. 選擇快速入門索引標籤，然後選擇 Amazon Linux。這是您的執行個體運行所在的作業系統 (OS)。
   b. 在 Amazon Machine Image (AMI) 中，選擇 Amazon Linux 2 HVM 版本。
5. 在執行個體類型中，從執行個體類型清單中，為執行個體選擇硬體組態。選擇 `t2.micro` 執行個體類型 (預設為選取)。`t2.micro` 執行個體類型符合 AWS 免費方案的資格。在無法使用 `t2.micro` 的 AWS 區域 中，您可以根據免費方案使用 `t3.micro` 執行個體。如需詳細資訊，請參閱 AWS 免費方案。
6. 在金鑰對 (登入) 的金鑰對名稱中，選擇金鑰對。
7. 在網路設定中，選擇編輯。在安全群組名稱中，您會看到為您建立的精靈並選取的安全群組。您可使用此安全群組，或者使用下列步驟選取您先前建立的安全群組：
   a. 選擇 Select existing security group (選取現有的安全群組)。
   b. 在 Common security groups (常見安全群組) 中，從現有的安全群組清單中選擇您的安全群組。
8. 如果您沒有使用預設主機管理組態，請展開進階詳細資料區段，然後在 IAM 執行個體設定檔中，選擇您在 設定 Systems Manager 所需的執行個體許可 中進行設定時建立的執行個體設定檔。
9. 保留執行個體其他組態設定的預設選擇。
10. 在摘要窗格中檢閱執行個體組態的摘要。就緒後，選擇啟動執行個體。
11. 此時會出現確認頁面，指出執行個體正在啟動。選擇 View all instances (檢視所有執行個體)，以關閉確認頁面並返回主控台。
12. 您可以在 Instances (執行個體) 畫面中檢視啟動狀態。啟動執行個體無須費時。
13. 執行個體可能需要幾分鐘的時間才會顯示為受管並準備就緒讓您連線。若要檢查執行個體是否已通過狀態檢查，可以在狀態檢查欄檢視此資訊。

使用 Systems Manager 連線至受管執行個體

連線至受管執行個體

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇想要連線之執行個體旁邊的按鈕。
4. 在節點動作功能表中，選擇啟動終端工作階段。
5. 選取連線。

清理您的執行個體

如果您不再使用為此教學課程建立的受管執行個體，請將其終止。終止執行個體可有效將其刪除。

終止您的執行個體

1. 在 https://console.aws.amazon.com/ec2/ 開啟 Amazon EC2 主控台。
2. 在導覽窗格中，選擇執行個體。在執行個體清單中，選取執行個體。
3. 選擇 Instance state (執行個體狀態)、Terminate instance (終止執行個體)。
4. 出現確認提示時，請選擇 Terminate (終止)。

   Amazon EC2 關閉並終止您的執行個體。在執行個體終止後，其仍會短暫顯示於主控台，然後項目才會自動刪除。您無法自行從主控台顯示畫面中移除已終止的執行個體。

AWS Systems Manager 節點工具

AWS Systems Manager 提供下列工具來存取、管理和設定受管節點。受管節點是設定為在混合多雲端環境中搭配 Systems Manager 使用的任何機器。

主題

- AWS Systems Manager合規
- AWS Systems Manager Distributor
- AWS Systems Manager Fleet Manager
- AWS Systems Manager 混合啟用

- AWS Systems Manager庫存
- AWS Systems Manager Patch Manager
- AWS Systems Manager Run Command
- AWS Systems Manager Session Manager
- AWS Systems Manager State Manager

AWS Systems Manager合規

您可以使用 中的工具 Compliance AWS Systems Manager，掃描受管節點機群的修補程式合規性和組態不一致。您可以從多個 AWS 帳戶 和 區域收集和彙總資料，然後深入探索不合規的特定資源。在預設情況下，合規會顯示有關在 中修補Patch Manager和在 中關聯的目前合規資料State Manager。(Patch Manager 和 State Manager 也是 中的工具 AWS Systems Manager。) 若要開始使用合規，請開啟 Systems Manager 主控台。在導覽窗格中，選擇 Compliance (合規)。

來自 的修補程式合規資料Patch Manager可以傳送到 AWS Security Hub。Security Hub 可為您提供高優先級安全性警示和合規性狀態的全方位檢視。它還會監控您的機群的修補狀態。如需詳細資訊，請參閱將 AWS Security Hub 與 Patch Manager 整合。

Compliance 還提供下列的優點和功能：

- 使用 AWS Config查看合規歷程記錄，以及 Patch Manager 修補資料和 State Manager 關聯的變更追蹤。
- 自訂 Compliance 以根據您的 IT 或業務需求建立自訂的合規類型。
- 使用 Run Command、、 State Manager或 Amazon EventBridge AWS Systems Manager中的另一個工具來修復問題。
- 將資料連接至 Amazon Athena 和 Amazon QuickSight 以產生全機群的報告。

支援 EventBridge

Amazon EventBridge 規則中支援此 Systems Manager 工具做為事件類型。如需詳細資訊，請參閱 使用 Amazon EventBridge 監控 Systems Manager 及 參考：Systems Manager 的 Amazon EventBridge 事件模式和類型。

Chef InSpec 整合

Systems Manager 與 Chef InSpec 整合。InSpec 是一種開放原始碼的執行時期架構，讓您能夠在 GitHub 或 Amazon Simple Storage Service (Amazon S3) 建立人類可讀的設定檔。您可以使用

Systems Manager 執行合規掃描，檢視合規與不合規的受管節點。如需詳細資訊，請參閱將 Chef InSpec 設定檔與 Systems Manager 合規搭配使用。

定價

合規無須額外付費。您只需為所使用的 AWS 資源付費。

目錄

- 開始使用合規
- 設定合規許可
- 建立合規的資源資料同步
- 了解有關合規性的詳細資訊
- 刪除合規的資源資料同步
- 使用 EventBridge 修正合規問題
- 使用 指派自訂合規中繼資料 AWS CLI

開始使用合規

若要開始使用 中的工具 Compliance AWS Systems Manager，請完成下列任務。

任務	如需詳細資訊
合規適用於 中的修補程式資料Patch Manager 和 中的關聯State Manager。(Patch Manager 和 State Manager 也是 中的工具 AWS Systems Manager。) 合規也可與使用 Systems Manager 管理的受管節點上的自訂合規類型搭配使用。確認您已完成混合多雲端環境中 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體和非 EC2 機器的設定要求。	為組織設定 Systems Manager 統一主控台
更新受管節點使用的 AWS Identity and Access Management (IAM) 角色，以限制合規許可。	設定合規許可
如果您打算監控修補程式合規，請確認您已設定 Patch Manager。您必須先使用 Patch Manager	AWS Systems Manager Patch Manager

任務	如需詳細資訊
執行修補操作，合規才能顯示修補程式合規資料。	
如果您打算監控關聯合規，請確認您已建立 State Manager 關聯。您必須先建立關聯，合規才能顯示關聯合規資料。	AWS Systems Manager State Manager
(選用) 設定系統以查看合規歷程記錄和變更追蹤。	檢視合規組態歷程記錄和變更追蹤
(選用) 建立自訂的合規類型。	使用 指派自訂合規中繼資料 AWS CLI
(選用) 建立資源資料同步，將所有合規資料彙總至目標 Amazon Simple Storage Service (Amazon S3) 儲存貯體。	建立合規的資源資料同步

設定合規許可

作為安全最佳實務，建議您使用下列許可來更新受管節點使用的 AWS Identity and Access Management (IAM) 角色，以限制節點使用 PutComplianceItems API 動作的能力。此 API 動作會在指定的資源上註冊合規類型和其他合規詳細資訊，例如 Amazon EC2 執行個體或受管節點。

如果您的節點是 Amazon EC2 執行個體，您必須使用下列許可更新執行個體所使用的 IAM 執行個體描述檔。如需 Systems Manager 管理之 EC2 執行個體的執行個體描述檔詳細資訊，請參閱 設定 Systems Manager 所需的執行個體許可。對於其他類型的受管節點，請使用下列許可更新節點使用的 IAM 角色。如需詳細資訊，請參閱《IAM 使用者指南》中的更新角色的許可。

```
{
    "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}"
                }
            }
        }
    ]
}
```

建立合規的資源資料同步

您可以使用 中的資源資料同步功能 AWS Systems Manager ，將合規資料從所有受管節點傳送至目標 Amazon Simple Storage Service (Amazon S3) 儲存貯體。建立同步時，您可以從多個 AWS 區域 AWS 帳戶以及混合多雲端環境指定受管節點。然後，資源資料同步會在系統收集新的合規資料時，自動更新集中的資料。當所有合規資料存放至目標 S3 儲存貯體後，您就能利用 Amazon Athena 和 Amazon QuickSight 等服務來查詢及分析彙總的資料。設定合規的資源資料同步是一次性操作。

請使用下列程序，藉由使用 AWS Management Console建立合規的資源資料同步。

建立和設定用於資源資料同步的 S3 儲存貯體 (主控台)

1. 開啟位於 https://console.aws.amazon.com/s3/ 的 Amazon S3 主控台。
2. 建立儲存貯體以存放您的彙整合規資料。如需詳細資訊，請參閱 Amazon Simple Storage Service 主控台使用者指南中的建立儲存貯體。請記下儲存貯體名稱和您建立儲存貯體 AWS 區域 所在的 。
3. 開啟儲存貯體，選擇 Permissions (許可) 標籤，接著選擇 Bucket Policy (儲存貯體政策)。
4. 複製下列儲存貯體政策並貼至政策編輯器。將 amzn-s3-demo-bucket 和 *Account-ID* 取代為您建立的 S3 儲存貯體名稱和有效的 AWS 帳戶 ID。或者，您也可以使用 Amazon S3 字首 (子目錄) 取代 *Bucket-Prefix*。如果您沒有建立字首，請將 *Bucket-Prefix*/ 從政策中的 ARN 移除。

```
{
    "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"
                }
            }
        }
    ]
}
```

建立資源資料同步

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇 Account management (帳戶管理)、Resource Data Syncs (資源資料同步)，然後選擇 Create resource data sync (建立資源資料同步)。
4. 在 Sync name (同步名稱) 欄位中，輸入同步組態的名稱。
5. 在 Bucket name (儲存貯體名稱) 欄位中，輸入此程序開始時建立的 Amazon S3 儲存貯體名稱。

6. (選用) 在儲存貯體字首欄位中，輸入 S3 儲存貯體字首 (子目錄) 的名稱。
7. 如果您建立的 S3 儲存貯體位於目前的 AWS 區域，請在儲存貯體區域欄位中選擇此區域。如果儲存貯體位於不同的 AWS 區域，請選擇另一個區域，然後輸入區域的名稱。

> **Note**
>
> 如果同步與目標 S3 儲存貯體位於不同區域，您可能需要支付資料傳輸費用。如需詳細資訊，請參閱 Amazon S3 定價。

8. 選擇建立。

了解有關合規性的詳細資訊

Compliance 是 中的工具 AWS Systems Manager，會收集和報告 中修補Patch Manager和關聯狀態的資料State Manager。 (Patch Manager 和 State Manager 也是 中的工具 AWS Systems Manager。) 合規也會報告您為受管節點指定的自訂合規類型。此部分包含每個合規類型以及如何檢視 Systems Manager 合規資料的詳細資訊。這部分還包含如何查看合規歷程記錄和變更追蹤的相關資訊。

> **Note**
>
> Systems Manager 與 Chef InSpec 整合。InSpec 是一種開放原始碼的執行時期架構，讓您能夠在 GitHub 或 Amazon Simple Storage Service (Amazon S3) 建立人類可讀的設定檔。然後，您可以使用 Systems Manager 執行合規掃描，檢視合規與不合規的執行個體。如需詳細資訊，請參閱將 Chef InSpec 設定檔與 Systems Manager 合規搭配使用。

關於修補程式合規

使用 Patch Manager 在執行個體安裝修補程式後，合規狀態資訊會立即於主控台、回應 AWS Command Line Interface (AWS CLI) 命令或對應的 Systems Manager API 操作時提供給您。

如需有關修補程式合規狀態值的詳細資訊，請參閱 修補程式合規狀態值。

關於State Manager關聯合規

建立一或多個State Manager關聯後，您立即可以在主控台中或回應 AWS CLI 命令或對應的 Systems Manager API 操作，取得合規狀態資訊。對於關聯，合規會顯示 `Compliant` 或 `Non-compliant` 狀態，以及指派至關聯的嚴重程度，例如 `Critical` 或 `Medium`。

關於自訂合規

您可以將合規中繼資料指派至受管節點。然後，此中繼資料可與其他合規資料彙整，用於合規報告。例如，假設您的企業在受管節點上執行版本 2.0、3.0 和 4.0 的軟體 X。公司希望標準化為版本 4.0，這表示執行版本 2.0 和 3.0 的執行個體不合規。您可以使用 PutComplianceItems API 操作來明確記下哪些受管節點正在執行舊版軟體 X。您只能使用 AWS CLI AWS Tools for Windows PowerShell或 SDKs 來指派合規中繼資料。以下 CLI 範例命令會指派合規中繼資料到受管執行個體，並以規定的格式 `Custom:` 指定合規類型。將每個`########`取代為您自己的資訊。

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` 參數僅支援 `ManagedInstance`。如果將自訂合規新增至受管 AWS IoT Greengrass 核心裝置，則您必須指定 `ManagedInstance` 的 `ResourceType`。

合規經理便可以查看哪些受管節點合規或不合規的摘要，或建立報告。您最多可以將 10 個不同的自訂合規類型指派至受管節點。

如需如何建立自訂合規類型和檢視合規資料的範例，請參閱 使用 指派自訂合規中繼資料 AWS CLI。

檢視目前的合規資料

本部分說明如何使用 Systems Manager 主控台和 AWS CLI檢視合規資料。如需有關檢視修補程式和關聯合規歷程記錄以及變更追蹤的資訊，請參閱 檢視合規組態歷程記錄和變更追蹤。

主題

- 檢視目前的合規資料 (主控台)
- 檢視目前的合規資料 (AWS CLI)

檢視目前的合規資料 (主控台)

使用以下程序在 Systems Manager 主控台中檢視合規資料。

若要在 Systems Manager 主控台檢視目前的合規報告

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Compliance (合規)。
3. 在 Compliance dashboard filtering (合規儀表板篩選) 區段中，選擇篩選合規資料的選項。Compliance resources summary (合規資源摘要) 區段會根據您選擇的篩選條件，顯示合規資料的計數。
4. 若要向下鑽研資源以取得詳細資訊，請向下捲動至 Details overview for resources (資源的詳細資料概觀) 區域，然後選擇受管節點的 ID。
5. 在 Instance ID (執行個體 ID) 或 Name (名稱) 詳細資訊頁面上，選取 Configuration compliance (組態合規) 索引標籤，以檢視受管節點的詳細組態合規報告。

> **Note**
>
> 如需有關修復合規問題的資訊，請參閱 使用 EventBridge 修正合規問題。

檢視目前的合規資料 (AWS CLI)

您可以使用 AWS CLI 下列 AWS CLI 命令，在 中檢視修補、關聯和自訂合規類型的合規資料摘要。

list-compliance-summaries

根據您指定的篩選條件，傳回合規與不合規關聯狀態的計數摘要。(API：ListComplianceSummaries)

list-resource-compliance-summaries

傳回資源層級的計數摘要。根據您指定的篩選條件標準，摘要包括有關合規與不合規狀態的資訊，以及詳細的合規項目嚴重程度計數。(API：ListResourceComplianceSummaries)

您可以使用下列 AWS CLI 命令，檢視修補的其他合規資料。

describe-patch-group-state

傳回修補程式群組彙總的高層級修補程式合規狀態。(API：DescribePatchGroupState)

describe-instance-patch-states-for-patch-group

傳回指定修補程式群組執行個體的高層級修補程式狀態。(API：DescribeInstancePatchStatesForPatchGroup)

> **Note**
>
> 如需如何使用 設定修補和檢視修補程式合規詳細資訊的圖例 AWS CLI，請參閱 教學課程：使用 修補伺服器環境 AWS CLI。

檢視合規組態歷程記錄和變更追蹤

Systems Manager Configuration Compliance 會顯示有關您受管節點目前的修補和關聯合規資料。您可以使用 來檢視修補和關聯合規歷史記錄，以及變更追蹤AWS Config。 AWS Config 提供 資源組態 AWS 的詳細檢視 AWS 帳戶。這包含資源彼此之間的關係和之前的組態方式，所以您可以看到一段時間中組態和關係的變化。若要檢視修補和關聯合規歷程記錄和變更追蹤，您必須在 AWS Config中開啟以下資源：

- `SSM:PatchCompliance`
- `SSM:AssociationCompliance`

如需如何在 AWS Config中選擇和設定這些特定資源的資訊，請參閱《AWS Config 開發人員指南》中的選取哪些資源 AWS Config 記錄。

> **Note**
>
> 如需 AWS Config 定價的資訊，請參閱定價。

刪除合規的資源資料同步

如果您不想再使用 AWS Systems Manager 合規來檢視合規資料，我們也建議您刪除用於合規資料收集的資源資料同步。

若要刪除合規資源資料同步

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇 Account management (帳戶管理)、Resource data syncs (資源資料同步)。
4. 在清單中選擇同步。

   > **Important**
   >
   > 請確定您選擇了用於合規的同步。Systems Manager 支援多個工具的資源資料同步。如果選擇了錯誤的同步，則您可能會中斷 Systems Manager Explorer 或 Systems Manager Inventory 的資料彙總。

5. 選擇 刪除。
6. 刪除存放資料的 Amazon Simple Storage Service (Amazon S3) 儲存貯體。如需有關刪除 S3 儲存貯體的資訊，請參閱刪除儲存貯體。

使用 EventBridge 修正合規問題

您可以使用 中的工具Run Command，快速修復修補程式和關聯合規問題 AWS Systems Manager。您可以鎖定執行個體或 AWS IoT Greengrass 核心裝置 IDs或標籤，並執行`AWS-RunPatchBaseline`文件或`AWS-RefreshAssociation`文件。如果重新整理關聯或重新執行修補基準無法解決合規問題，則您需要調查您的關聯、修補基準或執行個體組態，以了解為什麼 Run Command 操作未能解決問題。

如需有關修補的詳細資訊，請參閱 AWS Systems Manager Patch Manager 與 用於修補的 SSM 命令文件：`AWS-RunPatchBaseline`。

如需關聯的相關詳細資訊，請參閱 在 Systems Manager 中使用關聯。

如需有關執行命令的詳細資訊，請參閱 AWS Systems Manager Run Command。

指定合規為 Eventbridge 事件的目標

您也可以設定 Amazon EventBridge 執行動作，以回應 Systems Manager 合規事件。例如，如果一個以上的受管節點無法安裝重大修補程式更新，或執行安裝防毒軟體的關聯，則您可以設定 Eventbridge 在合規事件發生時，執行 `AWS-RunPatchBaseline` 文件或 `AWS-RefreshAssocation` 文件。

使用以下程序設定合規作為 Eventbridge 事件的目標。

若要設定合規作為 Eventbridge 事件的目標 (主控台)

1. 前往 https://console.aws.amazon.com/events/ 開啟 Amazon EventBridge 主控台。
2. 在導覽窗格中，選擇規則。
3. 選擇建立規則。
4. 輸入規則的名稱和描述。

   規則不能與相同 AWS 區域 和相同事件匯流排上的另一個規則具有相同的名稱。
5. 針對事件匯流排，選擇要與此規則建立關聯的事件匯流排。如果您希望此規則回應來自您自己的相符事件 AWS 帳戶，請選取預設值。當您 AWS 服務 帳戶中的 發出事件時，一律會傳送至您帳戶的預設事件匯流排。
6. 針對規則類型，選擇具有事件模式的規則。
7. 選擇下一步。
8. 在事件來源欄位中，選擇 AWS 事件或 EventBridge 合作夥伴事件。
9. 在 Event pattern (事件模式) 區段中，選擇 Event pattern form (事件模式表單)。
10. 在事件來源欄位中，選擇 AWS 服務。
11. 針對 AWS service (服務)，請選擇 Systems Manager。
12. 在 Event Type (事件類型) 欄位中，選擇 Configuration Compliance (組態合規)。
13. 針對 Specific detail type(s) (特定詳細資訊類型)，請選擇 Configuration Compliance State Change (組態合規狀態變更)。
14. 選擇 Next (下一步)。
15. 在目標類型欄位中，選擇 AWS 服務。
16. 針對 Select a target (選取目標)，請選擇 Systems Manager Run Command。

17. 在 Document (文件) 清單中，選擇叫用目標時要執行的 Systems Manager 文件 (SSM 文件)。例如，選擇 `AWS-RunPatchBaseline` 作為不合規修補程式事件，或選擇 `AWS-RefreshAssociation` 作為不合規關聯事件。
18. 指定其餘欄位和參數的資訊。

> **Note**
>
> 必填欄位和參數在名稱旁會有星號 (*)。若要建立目標，您必須為每個必填參數或欄位指定值。如果不這麼做，系統會建立規則，但規則不會執行。

19. 選擇 Next (下一步)。
20. (選用) 為規則輸入一或多個標籤。如需詳細資訊，請參閱《Amazon EventBridge 使用者指南》中的標記您的 Amazon EventBridge 資源。
21. 選擇 Next (下一步)。
22. 檢閱規則的詳細資訊，然後選擇建立規則。

使用 指派自訂合規中繼資料 AWS CLI

下列程序會逐步引導您使用 AWS Command Line Interface (AWS CLI) 呼叫 AWS Systems Manager PutComplianceItems API 操作，將自訂合規中繼資料指派給資源。您也可以使用此 API 操作，手動將修補程式或關聯合規中繼資料指派至受管節點，如以下演練中所示。如需自訂合規的詳細資訊，請參閱關於自訂合規。

將自訂合規中繼資料指派給受管執行個體 (AWS CLI)

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。
2. 執行以下命令，將自訂合規中繼資料指派給受管節點。將每個*########*取代為您自己的資訊。`ResourceType` 參數僅支援 `ManagedInstance` 的值。即使您將自訂合規中繼資料指派給受管 AWS IoT Greengrass 核心裝置，也請指定此值。

   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. 重複之前的步驟，以指派更多自訂合規中繼資料到一個以上的節點。您也可以使用下列命令，手動指派修補程式或關聯合規中繼資料到受管節點：

   關聯合規中繼資料

   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
```

修補程式合規中繼資料

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. 執行以下命令來檢視特定受管節點的合規項目清單。使用篩選條件來深入檢視特定的合規資料。

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
```

以下範例說明如何搭配篩選條件使用此命令。

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. 執行下列命令以檢視合規狀態摘要。使用篩選條件來深入檢視特定的合規資料。

```
aws ssm list-resource-compliance-summaries --filters One or more filters.
```

以下範例說明如何搭配篩選條件使用此命令。

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. 執行以下命令以檢視合規類型的合規與不合規資源計數摘要。使用篩選條件來深入檢視特定的合規資料。

```
aws ssm list-compliance-summaries --filters One or more filters.
```

以下範例說明如何搭配篩選條件使用此命令。

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 中的工具 AWS Systems Manager，可協助您封裝軟體並將其發佈至 AWS Systems Manager 受管節點。您可以封裝並發佈自己的軟體，或使用 Distributor 尋找和發佈 AWS提供的代理程

式軟體套件，例如 AmazonCloudWatchAgent 或第三方套件，例如 Trend Micro。發佈套件會將套件文件的特定版本公告至您使用節點 IDs、 AWS 帳戶 IDs、標籤或 識別的受管節點 AWS 區域。若要開始使用 Distributor，請開啟 Systems Manager 主控台。在導覽窗格中，選擇 Distributor。

在 Distributor 建立套件後，您可以用下列其中一種方法安裝套件：

- 使用 AWS Systems Manager Run Command 安裝一次
- 使用 AWS Systems Manager State Manager 根據排程進行

> **Important**
>
> 由第三方賣方分發的套件不受 管理 AWS ，並由套件的廠商發佈。我們建議您開展額外的盡職調查，以確保符合您的內部安全控制項。安全性是 AWS 與您之間的共同責任。這被描述為共同的責任模式。如需進一步了解，請參閱共同的責任模型。

Distributor 如何為我的組織帶來益處？

Distributor 提供這些好處：

- 一個套件，許多平台

  當您在 Distributor 中建立套件時，系統會建立 AWS Systems Manager 文件 (SSM 文件)。您可以將 .zip 檔案連接至此文件。當您執行 Distributor 時，系統會處理 SSM 文件中的指示，並在指定的目標上將軟體套件安裝在 .zip 檔案中。Distributor 支援多種作業系統，包括 Windows、Ubuntu Server、Debian Server 和 Red Hat Enterprise Linux。如需支援的平台的詳細資訊，請參閱支援的套件平台和架構。

- 跨受管執行個體群組控制套件存取權

  您可以使用 Run Command或 State Manager 來控制哪些受管節點取得套件，以及該套件的哪個版本。 Run Command和 State Manager是 中的工具 AWS Systems Manager。受管節點可以透過執行個體或裝置 ID、 AWS 帳戶 號碼、標籤或 AWS 區域進行分組。您可以使用 State Manager 關聯，提供不同版本的不同執行個體群組。

- 包含許多 AWS 代理程式套件，隨時可供使用

  Distributor 包含許多可供您部署到受管節點的 AWS 代理程式套件。尋找在 Amazon 發佈的 Distributor Packages 清單頁面上的套件。範例包括 AmazonCloudWatchAgent 和 AWSPVDriver。

- 自動化部署

  為了將您的環境保持在最新狀態，請使用 State Manager 在那些機器首次啟動時，排程目標受管節點的自動化部署套件。

誰應該使用Distributor？

- 任何想要建立或部署現有軟體套件 AWS 的客戶，包括 AWS 已發佈的套件，一次到多個 Systems Manager 受管節點。
- 建立軟體套件的軟體開發人員。
- 負責用最新軟體套件將 Systems Manager 受管節點保持在最新的管理員。

Distributor 有哪些功能？

- 部署到 Windows 和 Linux 執行個體的套件

  使用 Distributor，您可以將軟體套件部署到適用於 Linux 和 的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體和 AWS IoT Greengrass 核心裝置Windows Server。如需支援的執行個體作業系統類型清單，請參閱 the section called “支援的套件平台和架構”。

  > **Note**
  >
  > macOS 作業系統不支援 Distributor。

- 一次部署套件，或在自動化排程

  您可以選擇一次、定期或是預設套件版本變更時部署套件。

- 完全重新安裝套件或執行就地更新

  若要安裝新套件版本，您可以根據您提供的更新指令碼，完全解除安裝目前版本，並在其位置安裝新版本，或僅以新的和更新的元件更新目前版本。您的套件應用程式在重新安裝期間無法使用，但在就地更新期間仍可使用。就地更新對於安全性監視應用程式或其他需要避免應用程式停機的案例特別有用。

- 主控台、CLI、PowerShell 和 Distributor 功能的軟體開發存取權

  您可以使用 Distributor Systems Manager 主控台、 AWS Command Line Interface (AWS CLI) AWS Tools for PowerShell、 或您選擇的 AWS SDK 來使用 。

- IAM 存取控制

  透過使用 AWS Identity and Access Management (IAM) 政策，您可以控制組織哪些成員可以建立、更新、部署或刪除套件或套件版本。例如，您可能想授予管理員許可以部署套件，但不會變更套件或建立新的套件版本。

- 記錄和稽核功能支援

  您可以透過與其他 整合 AWS 帳戶 ，在 中稽核和記錄Distributor使用者動作 AWS 服務。如需詳細資訊，請參閱稽核和記錄 Distributor 活動。

什麼是 Distributor 中的套件？

套件是可安裝型軟體或包含以下的資產。

- 每個目標作業系統平台的軟體 .zip 檔案。每個 .zip 檔案必須包含以下。
  - install 和 uninstall 指令碼。Windows Server 型受管節點需要 PowerShell 指令碼 (名為 `install.ps1` 和 `uninstall.ps1` 的指令碼)。Linux 型受管節點需要 shell 指令碼 （ 名為 `install.sh`和 的指令碼`uninstall.sh`) AWS Systems Manager SSM Agent。 會讀取並執行 install和 uninstall指令碼中的指示。
  - 一個可執行檔。SSM Agent 必須找到這個可執行檔，並在目標受管節點上安裝套件。
- 描述封裝內容的 JSON 格式的資訊清單檔案。此資訊清單不包含在 .zip 檔內，但與 .zip 檔案存放在相同的 Amazon Simple Storage Service (Amazon S3) 儲存貯體形成套件。此資訊清單識別套件版本，並將套件裡的 .zip 檔案對應至目標受管節點屬性，例如作業系統版本或架構。如需如何建立資訊清單的資訊，請參閱步驟 2：建立 JSON 套件資訊清單。

當您在 Distributor 主控台選擇建立 Simple (簡便) 套件，Distributor 會根據軟體可執行檔案名稱和目標平台及架構，為您產生安裝和解除安裝指令碼、檔案雜湊和 JSON 套件資訊清單。

支援的套件平台和架構

您可以使用 Distributor 將套件發佈至下列 Systems Manager 受管節點平台。版本數值必須符合您設為目標的作業系統 Amazon Machine Image (AMI) 確切版本。如需決定此版本的詳細資訊，請參閱步驟 2：建立 JSON 套件資訊清單的步驟 4。

> **Note**
>
> Systems Manager 不支援 AWS IoT Greengrass 核心裝置的所有下列作業系統。如需詳細資訊，請參閱《 AWS IoT Greengrass Version 2 開發人員指南》中的設定 AWS IoT Greengrass 核心裝置。

平台	資訊清單檔案中的代碼值	支援的架構
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 執行個體類型)
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
Oracle Linux	`oracle`	x86_64
Red Hat Enterprise Linux (RHEL)	`redhat`	x86_64 或 x86 ARM64 (RHEL 7.6 及更新版本、A1 執行個體類型)
Rocky Linux	`rocky`	x86_64 ARM64
SUSE Linux Enterprise Server (SLES)	`suse`	x86_64 或 x86
Ubuntu Server	`ubuntu`	x86_64 或 x86

平台	資訊清單檔案中的代碼值	支援的架構
		ARM64 (Ubuntu Server 16 及更新版本、A1 執行個體類型)
Windows Server	`windows`	x86_64 或 x86 *

主題

- 設定 Distributor
- 使用 Distributor 套件
- 稽核和記錄 Distributor 活動
- 疑難排解 AWS Systems ManagerDistributor

設定 Distributor

在您使用 中的Distributor工具 AWS Systems Manager來建立、管理和部署軟體套件之前，請遵循下列步驟。

完成 Distributor 事前準備

使用 中的Distributor工具之前 AWS Systems Manager，請確定您的環境符合下列要求。

Distributor 先決條件

需求	描述
SSM Agent	AWS Systems Manager SSM Agent 2.3.274.0 版或更新版本必須安裝在您要部署或移除套件的受管節點上。 若要安裝或更新 SSM Agent，請參閱 使用 SSM Agent。
AWS CLI	（選用） 若要使用 AWS Command Line Interface (AWS CLI) 而非 Systems Manager 主控台來建立和管理套件，請在本機 AWS CLI 電腦上安裝最新版本的 。

需求	描述
	更多如何安裝或升級 CLI 的資訊，請參閱《AWS Command Line Interface 使用者指南》的安裝 AWS Command Line Interface。
AWS Tools for PowerShell	(選用) 若要使用 Tools for PowerShell 而非 Systems Manager 主控台來建立和管理套件，請安裝 Tools for PowerShell 的最新版本到您的本機電腦。 如需如何安裝或升級 Tools for PowerShell 的詳細資訊，請參閱 AWS Tools for PowerShell 使用者指南中的設定 AWS Tools for Windows PowerShell 或 AWS Tools for PowerShell Core。

> **Note**
>
> Systems Manager 不支援使用 Distributor 將套件分配至 Oracle Linux 受管節點。

驗證或建立具有 Distributor 許可的 IAM 執行個體設定檔

根據預設， AWS Systems Manager 沒有在您的執行個體上執行動作的許可。您必須使用 AWS Identity and Access Management (IAM) 執行個體描述檔授予存取權。執行個體設定檔是在啟動時將 IAM 角色資訊傳遞到 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體的容器。此要求適用於所有 Systems Manager 工具的許可，而不只是 Distributor。

> **Note**
>
> 當您將邊緣裝置設定為執行 AWS IoT Greengrass 核心軟體和 時SSM Agent，您可以指定 IAM 服務角色，讓 Systems Manager 能夠對其執行 動作。您不需要使用執行個體設定檔來設定受管邊緣裝置。

如果您已經使用其他 Systems Manager 工具，例如 Run Command和 State Manager，具有 必要許可的執行個體描述檔Distributor已連接至您的執行個體。確保您有許可執行 Distributor 任務最簡單的方法

是將 AmazonSSMManagedInstanceCore 政策連接到您的執行個體設定檔。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可。

控制使用者對套件的存取權

使用 AWS Identity and Access Management (IAM) 政策，您可以控制誰可以建立、部署和管理套件。您也可以控制他們在受管節點上可執行的 Run Command 或 State Manager API 操作。和 一樣 Distributor， Run Command和 State Manager都是 中的工具 AWS Systems Manager。

ARN 格式

使用者定義的套件與文件 Amazon 資源名稱 (ARN) 相關，具有下列格式。

```
arn:aws:ssm:region:account-id:document/document-name
```

以下是範例。

```
arn:aws:ssm:us-west-1:123456789012:document/ExampleDocumentName
```

您可以使用一對 AWS 提供的預設 IAM 政策，一個用於最終使用者，另一個用於管理員，以授予Distributor活動的許可。您也可以建立適用於您許可要求的自訂 IAM 政策。

如需有關在 IAM 政策中使用變數的詳細資訊，請參閱 IAM 政策元素：變數。

如需有關如何建立政策並將其連接至使用者或群組的相關資訊，請參閱《IAM 使用者指南》中的建立 IAM 政策和新增和移除 IAM 政策。

建立或選擇要存放 Distributor 套件的 Amazon S3 儲存貯體

當您使用 AWS Systems Manager 主控台中的簡易工作流程建立套件時，您可以選擇Distributor上傳軟體的現有 Amazon Simple Storage Service (Amazon S3) 儲存貯體。 Distributor 是一種工具 AWS Systems Manager。在 Advanced (進階) 工作流程中，您必須在開始前將軟體或資產的 .zip 檔案上傳到 Amazon Simple Storage Service (Amazon S3) 儲存貯體。無論您是透過在主控台中使用 Simple (簡便) 或 Advanced (進階) 工作流程或透過使用 API 來建立套件，您必須擁有 Amazon Simple Storage Service (Amazon S3) 儲存貯體，才能開始建立套件。在套件建立程序的過程中，Distributor 會將安裝型軟體和資產從這個儲存貯體複製到內部 Systems Manager 存放區。由於資產會複製到內部存放區，您可以在套件建立完成時刪除或重新利用您的 Amazon Simple Storage Service (Amazon S3) 儲存貯體。

若要如何建立儲存貯體的相關資訊，請參閱《Amazon Simple Storage Service 入門指南》中的建立儲存貯體。如需如何執行 AWS CLI 命令以建立儲存貯體的詳細資訊，請參閱《 AWS CLI 命令參考mb》中的 。

使用 Distributor 套件

您可以使用 AWS Systems Manager 主控台、 AWS 命令列工具 (AWS CLI 和 AWS Tools for PowerShell) 和 AWS SDKs 在 中新增、管理或部署套件Distributor。 Distributor 是 中的工具 AWS Systems Manager。在您新增套件至 Distributor 前：

- 建立和 zip 安裝型資產。
- (選用) 建立套件的 JSON 資訊清單檔案。在 Distributor 主控台使用 Simple (簡便) 套件建立程序，不需要此項。簡單套件建立會為您產生 JSON 資訊清單檔案。

  您可以使用 AWS Systems Manager 主控台或文字或 JSON 編輯器來建立資訊清單檔案。
- 讓 Amazon Simple Storage Service (Amazon S3) 儲存貯體準備就緒，以便存放可安裝的資產或軟體。如果您使用的是 Advanced (進階) 套件建立程序，請在開始前將資產上傳到 Amazon Simple Storage Service (Amazon S3) 儲存貯體。

> **Note**
>
> 您可以在完成套件的建立後刪除或重新利用此儲存貯體，因為 Distributor 會在套件建立程序過程將套件內容移動到內部 Systems Manager 儲存貯體。

AWS 已發佈的套件已封裝並準備好進行部署。若要將 AWS發佈的套件部署至受管節點，請參閱 安裝或更新 Distributor 套件。

您可以在 AWS 帳戶之間共用 Distributor 套件。在 AWS CLI 命令中使用從另一個帳戶共用的套件時，請使用套件 Amazon Resource Name (ARN)，而非套件名稱。

主題

- 檢視 Distributor 中的套件
- 在 Distributor 中建立套件
- 在主控台中編輯 Distributor 套件許可
- 在主控台中編輯 Distributor 套件標籤
- 將版本新增至 Distributor 套件

- 安裝或更新 Distributor 套件
- 解除安裝 Distributor 套件
- 刪除 Distributor 套件

檢視 Distributor 中的套件

若要檢視可供安裝的套件，您可以使用 AWS Systems Manager 主控台或偏好的 AWS 命令列工具。 Distributor 是 中的工具 AWS Systems Manager。若要存取 Distributor，請開啟 AWS Systems Manager 主控台，然後在Distributor左側導覽窗格中選擇 。您會看到所有可用的套件。

以下章節描述了您可以如何使用您偏好的命令列工具檢視 Distributor 套件。

使用命令列檢視套件

本節包含有關如何透過提供的命令使用您偏好的命令列工具來檢視 Distributor 套件。

Linux & macOS

在 Linux AWS CLI 上使用 檢視套件

- 若要檢視所有套件 (不包含共用套件)，請執行以下命令。

```
aws ssm list-documents \
    --filters Key=DocumentType,Values=Package
```

- 若要檢視 Amazon 擁有的所有套件，請執行以下命令。

```
aws ssm list-documents \
    --filters Key=DocumentType,Values=Package Key=Owner,Values=Amazon
```

- 若要檢視第三方擁有的所有套件，請執行以下命令。

```
aws ssm list-documents \
    --filters Key=DocumentType,Values=Package Key=Owner,Values=ThirdParty
```

Windows

在 Windows AWS CLI 上使用 檢視套件

- 若要檢視所有套件 (不包含共用套件)，請執行以下命令。

```
aws ssm list-documents ^
    --filters Key=DocumentType,Values=Package
```

- 若要檢視 Amazon 擁有的所有套件，請執行以下命令。

```
aws ssm list-documents ^
    --filters Key=DocumentType,Values=Package Key=Owner,Values=Amazon
```

- 若要檢視第三方擁有的所有套件，請執行以下命令。

```
aws ssm list-documents ^
    --filters Key=DocumentType,Values=Package Key=Owner,Values=ThirdParty
```

PowerShell

使用 Tools for PowerShell 檢視套件

- 若要檢視所有套件 (不包含共用套件)，請執行以下命令。

```
$filter = New-Object Amazon.SimpleSystemsManagement.Model.DocumentKeyValuesFilter
$filter.Key = "DocumentType"
$filter.Values = "Package"

Get-SSMDocumentList `
    -Filters @($filter)
```

- 若要檢視 Amazon 擁有的所有套件，請執行以下命令。

```
$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)
```

- 若要檢視第三方擁有的所有套件，請執行以下命令。

```
$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 中建立套件

若要建立套件，請準備安裝型軟體或資產，每個作業系統平台一個檔案。至少需要一個檔案才能建立套件。

不同平台可能有時會使用相同的檔案，但連接到您套件的所有檔案必須列在資訊清單中的 `Files` 區段。如果您透過在主控台中使用簡便工作流程來建立套件，系統會為您產生資訊清單。您可以連接到單一文件的檔案數量上限為 20。每個檔案大小的上限為 1 GB。如需支援的平台的詳細資訊，請參閱支援的套件平台和架構。

當您建立套件時，系統會建立新的 SSM 文件。此文件允許您將套件部署到受管節點。

您可從我們的網站下載僅供示範用途的範例套件 ExamplePackage.zip。此範例套件包含已完成的 JSON 資訊清單和三個包含 PowerShell v7.0.0 安裝程式的 .zip 檔案。安裝和解除安裝指令碼不包含有效的命令。雖然您必須將每個軟體安裝型檔案和指令碼壓縮為 .zip 檔，來在 Advanced (進階) 工作流程中建立套件，您不會在 Simple (簡易) 工作流程壓縮安裝型資產。

主題

- 使用簡單工作流程建立套件
- 使用進階工作流程建立套件

使用簡單工作流程建立套件

本節說明如何Distributor在 Distributor主控台中選擇簡易套件建立工作流程，在 中建立套件。Distributor 是 中的工具 AWS Systems Manager。若要建立套件，請準備安裝型資產，每個作業系統

平台一個檔案。至少需要一個檔案才能建立套件。Simple (簡單) 套件建立程序會為您產生安裝和解除安裝指令碼、檔案雜湊和 JSON 格式的資訊清單。Simple (簡單) 工作流程負責上傳和壓縮可安裝的檔案，以及建立新的套件和相關聯的 SSM 文件。如需支援的平台的詳細資訊，請參閱支援的套件平台和架構。

當您使用 Simple 方法建立套件時，Distributor 會為您建立 `install` 和 `uninstall` 指令碼。不過，建立就地更新的套件時，您必須在 Update script (更新指令碼) 索引標籤上提供您自己的 `update` 指令碼內容。當您為 `update` 指令碼新增輸入命令時，Distributor 會在它為您建立的 .zip 套件中包含此指令碼，以及 `install` 和 `uninstall` 指令碼。

> **Note**
>
> 使用 `In-place` 更新選項將新的或更新的檔案新增至現有的套件安裝，而不需讓關聯的應用程式離線。

使用簡單工作流程建立套件

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Distributor。
3. 在 Distributor 首頁上，選擇 Create package (建立套件)，然後選擇 Simple (簡單)。
4. 在 Create package (建立套件) 頁面，輸入套件名稱。套件名稱可包含字母、數字、句點、破折號和底線。名稱應該一般到足以適用於所有版本的套件附件，但也特別到足以識別套件目的。
5. (選用) 對於 Version name (版本名稱)，輸入版本名稱。版本名稱最多可包含 512 個字元，而且不得包含特殊字元。
6. 對於 Location (位置)，請使用儲存貯體名稱和前置詞，或使用儲存貯體 URL 來選擇儲存貯體。
7. 對於 Upload software (上傳軟體)，請選擇 Add software (新增軟體)，然後瀏覽具有 `.rpm`、`.msi` 或 `.deb` 副檔名的安裝型軟體檔案。如果檔案名稱包含空格，則上傳會失敗。您可以在單一動作中上傳多個軟體檔案。
8. 對於 Target platform (目標平台)，確認為每個安裝型檔案顯示的目標作業系統平台是否正確。如果顯示的作業系統不正確，請從下拉式清單中選擇正確的作業系統。

   對於 Simple (簡單) 套件建立工作流程中，因為您每個安裝型檔案僅上傳一次，需要額外的步驟來指示 Distributor 以多個作業系統的單一檔案為目標。例如，如果您上傳名為 `Logtool_v1.1.1.rpm` 的安裝型軟體檔案，您必須在 Simple (簡單) 工作流程中變更一些預設

值，來同時以 Amazon Linux 和 Ubuntu 作業系統的相同軟體為目標。當鎖定多個平台時，請執行下列其中一個動作。

- 在開始前，改用 Advanced (進階) 工作流程來將每個安裝型檔案壓縮為 .zip 檔案，以及手動編寫資訊清單，讓系統能夠以多個作業系統平台或版本的安裝型檔案為目標。如需詳細資訊，請參閱使用進階工作流程建立套件。
- 在 Simple (簡單) 工作流程中手動編輯資訊清單檔案，讓系統以多個作業系統平台或版本的 .zip 檔案為目標。如需如何進行此動作的詳細資訊，請參閱 步驟 2：建立 JSON 套件資訊清單中的步驟 4 結尾。

9. 對於 Platform version (平台版本)，請確認顯示的作業系統平台版本是 **_any** (後接萬用字元的主要發行版本 (7.*))，或正是您希望軟體套用的確切作業系統發行版本。如需指定作業系統平台版本的詳細資訊，請參閱 步驟 2：建立 JSON 套件資訊清單中的步驟 4。
10. 對於 Architecture (架構)，從下拉式清單中為每個安裝型檔案選擇正確的處理器架構。如需支援處理器架構的詳細資訊，請參閱 支援的套件平台和架構。
11. (選用) 展開 Scripts (指令碼)，並檢閱 Distributor 為安裝型軟體產生的指令碼。
12. (選用) 若要提供更新指令碼以與就地更新搭配使用，請展開 Scripts (指令碼)，選擇 Update script (更新指令碼) 索引標籤，然後輸入您的更新指令碼命令。

    Systems Manager 不會代表您產生更新指令碼。
13. 若要新增更多安裝型軟體檔案，選擇 Add software (新增軟體)。否則，請進行下一個步驟。
14. (選用) 展開 Manifest (資訊清單)，並檢閱 Distributor 為安裝型軟體產生的 JSON 套件資訊清單。如果在您開始此程序後變更了與您軟體相關的任何資訊，例如平台版本或目標平台，請選擇 Generate manifest (產生資訊清單) 以顯示更新的套件資訊清單。

    如果您想要以多個作業系統的軟體安裝型檔案為目標，您可以手動編輯資訊清單，如步驟 8 所述。如需編輯資訊清單的相關資訊，請參閱 步驟 2：建立 JSON 套件資訊清單。
15. 選擇 Create package (建立套件)。

等待 Distributor 完成您軟體的上傳與套件的建立。Distributor 會為每個安裝型檔案顯示上傳狀態。根據您所新增的套件數量和大小，這可能需要幾分鐘的時間。Distributor 會自動將您重新導向至新套件的 Package details (套件詳細資訊) 頁面，但您可以選擇在上傳軟體後自行開啟此頁面。Package details (套件詳細資訊) 頁面在 Distributor 完成套件的建立程序後，才會顯示您套件的所有相關資訊。若要停止上傳和套件建立程序，請選擇 Cancel (取消)。

如果 Distributor 無法上傳任何軟體安裝型檔案，就會顯示 Upload failed (上傳失敗) 訊息。若要重試上傳，請選擇 Retry upload (重試上傳)。如需如何對套件建立失敗進行疑難排解的更多資訊，請參閱 疑難排解 AWS Systems ManagerDistributor。

使用進階工作流程建立套件

在本節中，了解進階使用者將已壓縮的可安裝資產 (連同安裝和解除安裝指令碼) 和 JSON 資訊清單檔案上傳至 Amazon Simple Storage Service (Amazon S3) 儲存貯體後，如何在 Distributor 中建立套件。

若要建立套件，請準備安裝型資產的 .zip 檔案，每個作業系統平台一個 .zip 檔案。至少需要一個 .zip 檔案才能建立套件。下一步，建立 JSON 資訊清單。資訊清單包括您套件程式碼檔案的指標。將所需的程式碼檔案新增至資料夾或目錄，而且資訊清單中已填入正確的值之後，請將套件上傳到 S3 儲存貯體。

範例套件 ExamplePackage.zip 可用於從我們的網站下載。此範例套件包含已完成的 JSON 資訊清單和 3 個 .zip 檔案。

主題

- 步驟 1：建立 ZIP 檔案
- 步驟 2：建立 JSON 套件資訊清單
- 步驟 3：將套件和資訊清單上傳至 Amazon Simple Storage Service (Amazon S3) 儲存貯體
- 步驟 4：將套件新增至 Distributor

步驟 1：建立 ZIP 檔案

您的套件基礎至少要有一個軟體或安裝型資產的 .zip 檔案。套件包含每個您想支援的作業系統一個 .zip 檔案，除非 .zip 檔案可以安裝在多個作業系統。例如，Red Hat Enterprise Linux 和 Amazon Linux 執行個體通常可執行相同的 .RPM 可執行檔，所以您只需要將一個 .zip 檔案連接到套件就能支援這兩種作業系統。

必要檔案

下列是每個 .zip 檔案的必要項目：

- install 和 uninstall 指令碼。Windows Server 型受管節點需要 PowerShell 指令碼 (名為 `install.ps1` 和 `uninstall.ps1` 的指令碼)。Linux 類型的受管節點需要 shell 指令碼 (指令碼名為 `install.sh` 和 `uninstall.sh`)。SSM Agent 會執行 install 和 uninstall 指令碼中的指示。

  例如，您的安裝指令碼可能會執行安裝程式 (例如 .rpm 或 .msi)、可能會複製檔案或設定組態。

- 可執行檔案、安裝程式套件 (.rpm、.deb、.msi 等)、其他指令碼或組態檔案等。

選用檔案

下列項目是每個 .zip 檔案中的選用項目：

- update 指令碼。提供更新指令碼可讓您使用 `In-place update` 選項來安裝套件。當您想要將新的或更新的檔案新增至現有的套件安裝時，`In-place update` 選項可在執行更新時避免套件應用程式離線。Windows Server 受管節點需要 PowerShell 指令碼 (名為 `update.ps1` 的指令碼)。Linux 型受管節點需要 shell 指令碼 (指令碼名為 `update.sh`)。SSM Agent 會執行 update 指令碼中的指示。

如需安裝或更新套件的詳細資訊，請參閱安裝或更新 Distributor 套件。

如需 .zip 檔案範例，包括 install 和 uninstall 指令碼、下載範例套件、ExamplePackage.zip。

步驟 2：建立 JSON 套件資訊清單

在您準備和壓縮安裝型檔案後，請建立 JSON 資訊清單。下列為範本。資訊清單範本的部分將在此程序的此區段所述。您可以使用 JSON 編輯器，在個別檔案建立此資訊清單。或者，您可以在建立套件時，在 AWS Systems Manager 主控台中撰寫資訊清單。

```
{
  "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"
      }
    }
  }
}
```

若要建立 JSON 套件資訊清單

1. 新增結構描述版本到您的資訊清單。在此版本中，結構描述版本一律為 2.0。

   ```
   { "schemaVersion": "2.0",
   ```

2. 新增使用者定義的套件版本到您的資訊清單。這也是 Version name (版本名稱) 的數值，當您將套件新增至 Distributor 時指定。它會成為您建立的套件時建立 Distributor 的 AWS Systems Manager 文件的一部分。您也提供這個值做為 AWS-ConfigureAWSPackage 文件中的輸入，以安裝最新版本以外的套件版本。version 值可以包含字母、數字、底線、連字號和句點，長度上限為 128 字元。我們建議您使用便於讀取的套件版本，讓您和其他管理員部署時能輕鬆指定的確切套件版本。以下是範例。

   ```
   "version": "1.0.1",
   ```

3. (選用) 新增發佈者名稱。以下是範例。

   ```
   "publisher": "MyOrganization",
   ```

4. 新增套件。"packages" 區段敘述您套件中 .zip 檔案支援的平台、發行版本和架構。如需詳細資訊，請參閱支援的套件平台和架構。

   platform_version 可為萬用字元值 _any。使用它表示 .zip 檔案支援平台所有版本。您也可以指定後接萬用字元的主要發行版本，以便支援所有次要版本，例如 7.*。如果您選擇為指定作業系統版本指定 *platform-version* 值，請確定其符合您目標作業系統 AMI 的確切發行版本。以下是取得作業系統正確值的建議資源。

   - 在 Windows Server 型受管節點上，發行版本是 Windows Management Instrumentation (WMI) 資料形式。您可以執行以下來自「命令提示字元」的命令，以取得版本資訊，然後剖析 version 的結果。

     ```
     wmic OS get /format:list
     ```

   - 在以 Linux 為基礎的受管節點上，先透過掃描作業系統發行版本 (下列命令) 來取得版本。尋找 VERSION_ID 的數值。

     ```
     cat /etc/os-release
     ```

     如果這麼做沒有傳回您需要的結果，請執行下列命令，從 /etc/lsb-release 檔案取得 LSB 發行版本資訊並尋找 DISTRIB_RELEASE 的值。

     ```
     lsb_release -a
     ```

     若這些方法失敗，您可以根據分發找到版本。例如，您可以在 Debian Server 掃描 /etc/debian_version 檔案，或者在 Red Hat Enterprise Linux 掃描 /etc/redhat-release 檔案。

     ```
     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"
        }
      }
    }
  }
```

以下是範例。在這個範例中，作業系統平台是 amazon，支援的更新版本 2016.09，架構是 x86_64，支援此平台的 .zip 檔案則是 test.zip。

```
{
    "amazon": {
        "2016.09": {
            "x86_64": {
                "file": "test.zip"
            }
        }
    }
},
```

您可以新增萬用字元值 _any，表示套件支援所有版本的父元素。例如，若要表示所有 Amazon Linux 更新版本都支援此套件，您的套件陳述式應該與下列相似。您可以在版本或架構層級使用 _any 萬用字元，支援平台所有版本，或版本的所有架構，或平台所有架構的所有版本。

```
{
    "amazon": {
        "_any": {
            "x86_64": {
                "file": "test.zip"
            }
        }
    }
```

```
},
```

以下範例新增 _any 以顯示第一個套件，Amazon Linux 2016.09 的所有架構都支援 data1.zip。Amazon Linux 的所有版本都支援第二個套件 data2.zip，但只支援有 x86_64 架構的受管節點。2016.09 和 _any 都是 amazon 下的項目。有一個平台 (Amazon Linux)，但有不同的支援版本、架構和相關 .zip 檔案。

```
{
    "amazon": {
        "2016.09": {
            "_any": {
                "file": "data1.zip"
            }
        },
        "_any": {
            "x86_64": {
                "file": "data2.zip"
            }
        }
    }
}
```

若 .zip 檔案支援多個平台，您可以參閱資訊清單 "packages" 區段的 .zip 檔案一次以上。例如，如果您有一個同時支援 Red Hat Enterprise Linux 7.x 版本和 Amazon Linux 的 .zip 檔，您在 "packages" 區段會有兩個指向相同 .zip 檔的項目，如下範例所示。

```
{
    "amazon": {
        "2018.03": {
            "x86_64": {
                "file": "test.zip"
            }
        }
    },
    "redhat": {
        "7.*": {
            "x86_64": {
                "file": "test.zip"
            }
        }
    }
```

```
},
```

5. 新增步驟 4 中，做為此套件的部分的 .zip 檔案清單。每個檔案項目需要檔案名稱和 sha256 雜湊值的檢查總和。資訊清單中的檢查總和值必須符合壓縮資產中的 sha256 雜湊值，避免套件的安裝失敗。

   若要從您可安裝型取得確切檢查總和，您可以執行以下命令。在 Linux 上執行 `shasum -a 256 file-name.zip` 或 `openssl dgst -sha256 file-name.zip`。在 Windows 上，執行 PowerShell 中的 `Get-FileHash -Path path-to-.zip-file` cmdlet。

   資訊清單的 `"files"` 區段包含您套件中每個 .zip 檔案的參考。

```
"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. 在您新增套件資訊後，儲存並關閉資訊清單檔案。

以下是已完成的資訊清單範例：在這個範例中，您有一個 .zip 檔案 NewPackage_LINUX.zip，支援多個平台，但曾在 "files" 區段中參考一次。

```
{
    "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"
            }
        }
    }
}
```

套件範例

範例套件 ExamplePackage.zip 可用於從我們的網站下載。此範例套件包含已完成的 JSON 資訊清單和 3 個 .zip 檔案。

步驟 3：將套件和資訊清單上傳至 Amazon Simple Storage Service (Amazon S3) 儲存貯體

將所有 .zip 檔案複製或移動到資料夾或目錄，以準備您的套件。有效套件需要您在 步驟 2：建立 JSON 套件資訊清單 建立的資訊清單，以及資訊清單檔案清單中找到的所有 .zip 檔案。

上傳套件和資訊清單至 Amazon Simple Storage Service (Amazon S3)

1. 將資訊清單檔案中您指定的所有 .zip 封存檔案複製或移動到資料夾或目錄。請勿壓縮將 .zip 封存檔案和資訊清單檔案所移至的資料夾或目錄。
2. 建立儲存貯體或選擇現有的儲存貯體。如需詳細資訊，請參閱《Amazon Simple Storage Service 入門指南》中的建立儲存貯體。如需如何執行 AWS CLI 命令以建立儲存貯體的詳細資訊，請參閱《 AWS CLI 命令參考mb》中的 。

3. 將資料夾或目錄上傳至儲存貯體。如需詳細資訊，請參閱《Amazon Simple Storage Service 入門指南》中的將物件新增到儲存貯體。如果您打算將 JSON 資訊清單貼到 AWS Systems Manager 主控台，請勿上傳資訊清單。如需如何執行 AWS CLI 命令以將檔案上傳至儲存貯體的詳細資訊，請參閱《 AWS CLI 命令參考mv》中的 。
4. 在儲存貯體的首頁，選擇您上傳的資料夾或目錄。如果您將檔案上傳到儲存貯體中的子資料夾，請務必記下子資料夾 (也稱為前綴)。您需要此前綴來將套件新增至 Distributor。

步驟 4：將套件新增至 Distributor

您可以使用 AWS Systems Manager 主控台、 AWS 命令列工具 (AWS CLI 和 AWS Tools for PowerShell) 或 AWS SDKs將新套件新增至 Distributor。新增套件時，您會新增新的 SSM 文件。文件允許您將套件部署到受管節點。

主題

- 使用主控台新增套件
- 使用 新增套件 AWS CLI

使用主控台新增套件

您可以使用 AWS Systems Manager 主控台來建立套件。將您在 步驟 3：將套件和資訊清單上傳至 Amazon Simple Storage Service (Amazon S3) 儲存貯體 中的套件上傳目標的儲存貯體名稱準備好。

將套件新增至 Distributor (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Distributor。
3. 在 Distributor 首頁上，選擇 Create package (建立套件)，然後選擇 Advanced (進階)。
4. 在 Create package (建立套件) 頁面，輸入套件名稱。套件名稱可包含字母、數字、句點、破折號和底線。名稱應該一般到足以適用於所有版本的套件附件，但也特別到足以識別套件目的。
5. 對於 Version name (版本名稱)，輸入您資訊清單檔案中 `version` 項目的確切值。
6. 對於 S3 bucket name (S3 儲存貯體名稱)，選擇您在 the section called “步驟 3：將套件和資訊清單上傳至 Amazon Simple Storage Service (Amazon S3) 儲存貯體”中將 .zip 檔案和資訊清單上傳至其中的儲存貯體名稱。
7. 對於 S3 key prefix (S3 金鑰前綴)，輸入 .zip 檔案和資訊清單存放所在的儲存貯體子資料夾。

8. 對於 Manifest (資訊清單)，選擇 Extract from package (從套件中擷取) 來使用您透過 .zip 檔案上傳到 Amazon Simple Storage Service (Amazon S3) 儲存貯體的資訊清單。

   (選用) 如果您未將 JSON 資訊清單上傳到 .zip 檔案存放所在的 S3 儲存貯體，請選擇 New manifest (新增資訊清單)。您可以在 JSON 編輯器欄位編寫或貼上整個資訊清單。更多如何建立 JSON 資訊清單的資訊，請參閱步驟 2：建立 JSON 套件資訊清單。
9. 當您完成資訊清單，請選擇 Create package (建立套件)。
10. 等待 Distributor 透過 .zip 檔案和資訊清單來建立套件。根據您所新增的套件數量和大小，這可能需要幾分鐘的時間。Distributor 會自動將您重新導向至新套件的 Package details (套件詳細資訊) 頁面，但您可以選擇在上傳軟體後自行開啟此頁面。Package details (套件詳細資訊) 頁面在 Distributor 完成套件的建立程序後，才會顯示您套件的所有相關資訊。若要停止上傳和套件建立程序，請選擇 Cancel (取消)。

使用 新增套件 AWS CLI

您可以使用 AWS CLI 來建立套件。將您在 步驟 3：將套件和資訊清單上傳至 Amazon Simple Storage Service (Amazon S3) 儲存貯體 中的套件上傳的儲存貯體的 URL 準備好。

使用 將套件新增至 Amazon S3 AWS CLI

1. 若要使用 AWS CLI 建立套件，請執行下列命令，將 *package-name* 取代為套件的名稱，並將 *path-to-manifest-file* 取代為 JSON 資訊清單檔案的檔案路徑。amzn-s3-demo-bucket 是存放整個套件的 Amazon S3 儲存貯體 URL。當您在 Distributor 中執行 create-document 命令時，請為 `--document-type` 指定 `Package` 的值。

   若您沒有新增資訊清單檔案到 Amazon Simple Storage Service (Amazon S3) 儲存貯體，`--content` 參數值就是 JSON 資訊清單檔案的檔案路徑。

   ```
   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
   ```

   以下是範例。

   ```
   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. 驗證您的套件已新增，並用您的套件名稱取代 *package_name*，而執行以下命令，以顯示套件資訊清單。若要取得特定版本的文件 (而非與套件版本相同)，您可以新增 `--document-version` 參數。

```
aws ssm get-document \
    --name "package-name"
```

如需可以與 create-document 命令搭配使用之其他選項的相關資訊，請參閱《AWS CLI 命令參考》中 AWS Systems Manager 章節的 create-document。如需可搭配 get-document 命令使用之其他選項的相關資訊，請參閱get-document。

在主控台中編輯 Distributor 套件許可

將套件新增至 中的Distributor工具後 AWS Systems Manager，您可以在 Systems Manager 主控台中編輯套件的許可。您可以將其他 AWS 帳戶 新增至套件的許可。套件的共用對象只限於相同 AWS 區域中的其他帳戶。不支援跨區域共用。根據預設，套件會設為私有，這表示只有可存取套件建立者之 的 AWS 帳戶 才能檢視套件資訊，以及更新或刪除套件。如果能夠接受 Private (私有) 許可，則可以略過此程序。

> **Note**
>
> 您可以更新與 20 個或更少帳戶共用的套件的許可。

在主控台中編輯套件許可

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Distributor。
3. 在 Packages (套件) 頁面選擇您要編輯許可的套件。
4. 在 Package details (套件詳細資訊) 標籤選擇 Edit permissions (編輯許可) 變更許可。
5. 對於 Edit permissions (編輯許可)，選擇 Shared with specific accounts (與特定帳戶共享)。

6. 在 Shared with specific accounts (與特定帳戶共享) 下新增 AWS 帳戶 號碼，一次一個。完成後，請選擇儲存。

在主控台中編輯 Distributor 套件標籤

將套件新增至 中的Distributor工具後 AWS Systems Manager，您可以在 Systems Manager 主控台中編輯套件的標籤。這些標籤會套用到套件，不會連接您部署套件的受管節點標籤上。標籤有區分大小寫的金鑰和值組，可協助您透過條件分組及篩選適用於您組織的套件。如果您不想新增標籤，就可以準備安裝套件或新增新版本了。

在主控台中編輯套件標籤

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Distributor。
3. 在 Packages (套件) 頁面選擇您要編輯標籤的套件。
4. 在 Details (詳細資訊) 索引標籤上，選擇 Tags (標籤) > Edit (編輯)。
5. 對於 Add tags (新增標籤)，輸入標籤鍵或標籤鍵和值組，然後選擇 Add (新增)。若您想新增更多標籤，請再重複一次。若要刪除標籤中，請選擇視窗底部標籤上的 X。
6. 完成新增標籤至套件時，請選擇 Save (儲存)。

將版本新增至 Distributor 套件

若要新增套件版本，請建立套件，然後使用 新增項目至已存在於較舊版本的 AWS Systems Manager (SSM) 文件Distributor，以新增套件版本。 Distributor 是 中的工具 AWS Systems Manager。若要節省時間，請更新較舊版本套件的資訊清單，變更資訊清單中 `version` 項目的值 (例如，從 `Test_1.0` 到 `Test_2.0`)，並儲存為新版本的資訊清單。Distributor 主控台中的簡便 Add version (新增版本) 工作流程會為您更新資訊清單檔案。

新套件版本能：

- 至少替換一個連接到目前版本的安裝型檔案。
- 新增新的安裝型檔案，來支援額外的平台。
- 刪除檔案，不再支援特定平台。

較新版本可以使用相同的 Amazon Simple Storage Service (Amazon S3) 儲存貯體，但 URL 尾端顯示的檔案名稱必須不同。您可以使用 Systems Manager 主控台或 AWS Command Line Interface (AWS CLI) 以新增新版本。上傳與 Amazon Simple Storage Service (Amazon S3) 儲存貯體中現有安裝型檔案相同名稱的安裝型檔案，會覆寫現有的檔案。系統不會將安裝型檔案從舊版複製到新版，您必須從舊版上傳安裝型檔案來讓這些檔案成為新版本的一部分。在 Distributor 完成新套件版本的建立後，您就可以刪除或重新利用 Amazon Simple Storage Service (Amazon S3) 儲存貯體，因為 Distributor 會在版本控制程序期間將您的軟體複製到內部 Systems Manager 儲存貯體。

> **Note**
>
> 每個套件最多能保留 25 個版本。您可以刪除不再需要的版本。

主題

- 使用主控台新增套件版本
- 使用 AWS CLI新增套件版本

使用主控台新增套件版本

執行這些步驟之前，請按照在 Distributor 中建立套件中的指示建立版本的新套件。接著使用 Systems Manager 主控台，將套件版本新增到 Distributor。

使用簡單工作流程新增套件版本

若要透過使用 Simple (簡單) 工作流程來新增套件版本，請準備更新的安裝型檔案，或新增安裝型檔案來支援更多平台和架構。然後，使用 Distributor 來上傳新的和更新的安裝型檔案和新增套件版本。Distributor 主控台中簡化的 Add version (新增版本) 工作流程會為您更新資訊清單檔案和相關的 SSM 文件。

使用簡單工作流程新增套件版本

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Distributor。
3. 在 Distributor 首頁，選擇套件到您想新增其他版本的地方。
4. 在 Add version (新增版本) 頁面上，選擇 Simple (簡單)。
5. 對於 Version name (版本名稱)，輸入版本名稱。新版本的版本名稱必須與舊版本名稱不同。版本名稱最多可包含 512 個字元，而且不得包含特殊字元。

6. 對於 S3 bucket name (S3 儲存貯體名稱)，從清單中選擇現有的 S3 儲存貯體。這可以與您存放舊版安裝型檔案使用的儲存貯體相同，但安裝型檔案名稱必須不同，來避免在儲存貯體中覆寫現有的安裝型檔案。
7. 對於 S3 key prefix (S3 金鑰前綴)，輸入安裝型資產存放所在的儲存貯體子資料夾。
8. 對於 Upload software (上傳軟體) 中，瀏覽您想要連接到新版本的安裝型軟體檔案。系統不會將現有版本中的安裝型檔案自動複製到新版本，如果您希望任何相同的安裝型檔案成為新版本的一部分，您必須從舊版套件上傳任何安裝型檔案。您可以在單一動作中上傳多個軟體檔案。
9. 對於 Target platform (目標平台)，確認為每個安裝型檔案顯示的目標作業系統平台是否正確。如果顯示的作業系統不正確，請從下拉式清單中選擇正確的作業系統。

   在 Simple (簡單) 版本控制工作流程中，因為您每個安裝型檔案僅上傳一次，需要額外的步驟來以多個作業系統的單一檔案為目標。例如，如果您上傳名為 `Logtool_v1.1.1.rpm` 的安裝型軟體檔案，您必須在 Simple (簡單) 工作流程中變更一些預設值，來指示 Distributor 同時以 Amazon Linux 和 Ubuntu 作業系統的相同軟體為目標。您可以執行以下其中一項操作來解決這個限制。

   - 在開始前，改用 Advanced (進階) 版本控制工作流程來將每個安裝型檔案壓縮為 .zip 檔案，以及手動編寫資訊清單，讓系統能夠以多個作業系統平台或版本的安裝型檔案為目標。如需詳細資訊，請參閱使用進階工作流程新增套件版本。
   - 在 Simple (簡單) 工作流程中手動編輯資訊清單檔案，讓系統以多個作業系統平台或版本的 .zip 檔案為目標。如需如何進行此動作的詳細資訊，請參閱 步驟 2：建立 JSON 套件資訊清單中的步驟 4 結尾。
10. 對於 Platform version (平台版本)，請確認顯示的作業系統平台版本是 **_any** (後接萬用字元的主要發行版本 (7.*))，或正是您希望軟體套用的確切作業系統發行版本。如需指定平台版本的詳細資訊，請參閱 步驟 2：建立 JSON 套件資訊清單中的步驟 4。
11. 對於 Architecture (架構)，從下拉式清單中為每個安裝型檔案選擇正確的處理器架構。如需支援架構的詳細資訊，請參閱 支援的套件平台和架構。
12. (選用) 展開 Scripts (指令碼)，並檢閱 Distributor 為安裝型軟體產生的安裝和解除安裝指令碼。
13. 若要將更多安裝型軟體檔案新增至新版本，選擇 Add software (新增軟體)。否則，請進行下一個步驟。
14. (選用) 展開 Manifest (資訊清單)，並檢閱 Distributor 為安裝型軟體產生的 JSON 套件資訊清單。如果在您開始此程序後變更了與安裝型軟體相關的任何資訊，例如平台版本或目標平台，請選擇 Generate manifest (產生資訊清單) 以顯示更新的套件資訊清單。

    如果您想要以多個作業系統的軟體安裝型檔案為目標，您可以手動編輯資訊清單，如步驟 9 所述。如需編輯資訊清單的相關資訊，請參閱 步驟 2：建立 JSON 套件資訊清單。

15. 當您完成軟體的新增和目標平台、版本和架構資料的檢閱後，請選擇 Add version (新增版本)。
16. 等待 Distributor 完成您軟體的上傳與新套件版本的建立。Distributor 會為每個安裝型檔案顯示上傳狀態。根據您所新增的套件數量和大小，這可能需要幾分鐘的時間。Distributor 會自動將您重新導向至套件的 Package details (套件詳細資訊) 頁面，但您可以選擇在上傳軟體後自行開啟此頁面。此 Package details (套件詳細資訊) 頁面在 Distributor 完成新套件版本的建立後，才會顯示您套件的所有相關資訊。若要停止上傳和套件版本建立，請選擇 Stop upload (停止上傳)。
17. 如果 Distributor 無法上傳任何軟體安裝型檔案，就會顯示 Upload failed (上傳失敗) 訊息。若要重試上傳，請選擇 Retry upload (重試上傳)。如需如何對套件版本建立失敗進行疑難排解的更多資訊，請參閱 疑難排解 AWS Systems ManagerDistributor。
18. 在 Distributor 完成新套件版本的建立時，可在套件 Details (詳細資訊) 頁面的 Versions (版本) 標籤上，在可用套件版本的清單中檢視新版本。選擇版本來設定預設的套件版本，然後選擇 Set default version (設定預設版本)。

    如果您不設定預設版本，則最新套件版本就是預設版本。

使用進階工作流程新增套件版本

若要新增套件版本，請建立套件，然後透過將項目新增至已存在於較舊版本的 SSM 文件，來使用 Distributor 新增套件版本。若要節省時間，請更新較舊版本套件的資訊清單，變更資訊清單中 `version` 項目的值 (例如，從 `Test_1.0` 到 `Test_2.0`)，並儲存為新版本的資訊清單。您必須擁有更新的資訊清單，透過使用 Advanced (進階) 工作流程來新增新的套件版本。

使用進階工作流程新增套件版本

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Distributor。
3. 在 Distributor 首頁，選擇您想將其他版本新增至其中的套件，然後選擇 Add version (新增版本)。
4. 對於 Version name (版本名稱)，輸入您資訊清單檔案中 `version` 項目的確切值。
5. 對於 S3 bucket name (S3 儲存貯體名稱)，從清單中選擇現有的 S3 儲存貯體。這可以與您存放舊版安裝型檔案使用的儲存貯體相同，但安裝型檔案名稱必須不同，來避免在儲存貯體中覆寫現有的安裝型檔案。
6. 對於 S3 key prefix (S3 金鑰前綴)，輸入安裝型資產存放所在的儲存貯體子資料夾。
7. 對於 Manifest (資訊清單)，選擇 Extract from package (從套件中擷取) 來使用您透過 .zip 檔案上傳到 S3 儲存貯體的資訊清單。

(選用) 如果您未將修訂的 JSON 資訊清單上傳到 .zip 檔存放所在的 Amazon Simple Storage Service (Amazon S3) 儲存貯體，請選擇 New manifest (新增資訊清單)。您可以在 JSON 編輯器欄位編寫或貼上整個資訊清單。更多如何建立 JSON 資訊清單的資訊，請參閱步驟 2：建立 JSON 套件資訊清單。

8. 當您完成資訊清單時，請選擇 Add package version (新增套件版本)。
9. 在套件的 Details (詳細資訊) 頁面的 Versions (版本) 標籤上，可查看可用套件版本清單的新版本。選擇版本來設定預設的套件版本，然後選擇 Set default version (設定預設版本)。

   如果您不設定預設版本，則最新套件版本就是預設版本。

使用 AWS CLI新增套件版本

您可以使用 AWS CLI 將新的套件版本新增至 Distributor。如本主題開始所述，執行這些命令前，您必須建立新套件版本並上傳到 S3。

使用 新增套件版本 AWS CLI

1. 執行下列命令，使用新套件版本的項目編輯 AWS Systems Manager 文件。用您的文件的名稱取代 *document_name*。使用在 步驟 3：將套件和資訊清單上傳至 Amazon Simple Storage Service (Amazon S3) 儲存貯體 中複製的 JSON 資訊清單 URL 取代 *amzn-s3-demo-bucket*。*S3-bucket-URL-of-package* 是儲存整個套件的 Amazon Simple Storage Service (Amazon S3) 儲存貯體的 URL。使用資訊清單中的 `version` 值取代 *version-name-from-updated-manifest*。將 `--document-version` 參數設定為 `$LATEST`，讓與此套件版本相關聯的文件變成最新文件版本。

   ```
   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
   ```

   以下是範例。

   ```
   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. 執行以下命令來驗證您的套件已更新，並顯示套件資訊清單。請用您的套件名稱取代 *package_name*，並可選擇用您更新的文件版本編號 (與套件版本不同) 取代 *document_version*。若此套件版本與最新文件版本相關，您可以將選用 --document-version 參數的值指定為 $LATEST。

```
aws ssm get-document \
    --name "package-name" \
    --document-version "document-version"
```

如需有關可與 update-document 命令搭配使用的其他選項的資訊，請參閱 AWS CLI 命令參考的 update-document AWS Systems Manager 一節。

安裝或更新 Distributor 套件

您可以使用 中的工具Distributor，將套件部署到受 AWS Systems Manager 管節點 AWS Systems Manager。若要部署套件，請使用 AWS Management Console 或 AWS Command Line Interface (AWS CLI)。您可以在每個命令部署一個套件版本。您可以安裝新套件或就地更新現有的安裝。您可以選擇部署特定版本，或選擇一律用最新部署套件版本來部署。我們建議您使用 中的State Manager工具 AWS Systems Manager來安裝套件。使用 State Manager 協助確保受管節點始終執行最新版本的套件。

> **Important**
>
> 您只能使用 Distributor 解除安裝您使用 Distributor 安裝的套件。否則，Systems Manager 仍然可以將應用程式註冊為 INSTALLED，並導致其他非預期的結果。

Preference	AWS Systems Manager 動作	詳細資訊
立即安裝或更新套件。	Run Command	• 使用主控台一次性安裝或更新套件 • 使用 安裝套件一次 AWS CLI • 使用 更新套件一次 AWS CLI

Preference	AWS Systems Manager 動作	詳細資訊
依排程安裝或更新的套件，以便安裝一律會包含預設版本。	State Manager	• 使用主控台排程套件安裝或更新 • 使用 排程套件安裝 AWS CLI • 使用 排程套件更新 AWS CLI
自動安裝套件到具有特定標籤或標籤集合的新受管節點上。例如，在新執行個體上安裝 Amazon CloudWatch 代理程式。	State Manager	有一種方法，可以套用標籤到新的受管節點，然後在您的 State Manager 關聯指定標籤為目標。State Manager 會在符合關聯標籤的受管節點關聯中安裝套件。請參閱 了解 State Manager 關聯中的目標和速率控制。

主題

- 使用主控台一次性安裝或更新套件
- 使用主控台排程套件安裝或更新
- 使用 安裝套件一次 AWS CLI
- 使用 更新套件一次 AWS CLI
- 使用 排程套件安裝 AWS CLI
- 使用 排程套件更新 AWS CLI

使用主控台一次性安裝或更新套件

您可以使用 AWS Systems Manager 主控台安裝或更新套件一次。當您設定一次性安裝時， Distributor 會使用 中的AWS Systems Manager Run Command工具 AWS Systems Manager來執行安裝。

使用主控台一次性安裝或更新套件

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Distributor。
3. 在 Distributor 首頁上，選擇您想安裝的套件。

4. 選擇 Install one time (一次性安裝)。

   這個命令會用命令文件 AWS-ConfigureAWSPackage 和您選擇的 Distributor 套件開啟 Run Command。

5. 對於 Document version (文件版本)，選取您要執行的 AWS-ConfigureAWSPackage 文件的版本。

6. 針對 Action (動作)，選擇 Install (安裝)。

7. 對於 Installation type (安裝類型)，選擇以下其中一項：

   - Uninstall and reinstall (解除安裝並重新安裝)：套件會完全解除安裝，然後重新安裝。在重新安裝完成之前，應用程式無法使用。
   - In-place update (就地更新)：根據您在 update 指令碼中提供的指示，只將新的或變更的檔案新增至現有的安裝。應用程式在整個更新程序中仍然可用。已 AWS 發佈的套件不支援此選項，但AWSEC2Launch-Agent套件除外。

8. 對於 Name (名稱)，確認已輸入您所選套件的名稱。

9. (選用) 對於 Version (版本)，輸入套件的版本名稱值。如果您將此欄位留空，Run Command 會安裝您在 Distributor 選的預設版本。

10. 在 Targets (目標) 區段中，透過手動指定標籤、選取執行個體或裝置，或指定資源群組，選擇您要執行這項操作的受管節點。

    > **Note**
    >
    > 如果在清單中沒看到受管節點，則請參閱 疑難排解受管節點的可用性。

11. 對於其他參數：

    - 在 Comment (註解) 中，輸入此命令的相關資訊。
    - 在逾時 (秒) 中，指定在命令執行全面失敗之前，系統要等候的秒數。

12. 對於 Rate Control (速率控制)：

    - 在 Concurrency (並行) 中，指定可同時執行命令的節點數目或百分比。

      > **Note**
      >
      > 如果透過指定標籤或資源群組選取了目標，且您不確定會以多少個受管節點為目標，則透過指定百分比限制可以同時執行文件之目標的數量。

- 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比之受管節點上的命令失敗之後，停止在其他目標上執行命令。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止傳送命令。仍在處理命令的受管節點也可能會傳送錯誤。

13. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取將命令輸出寫入至 S3 儲存貯體方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

    > **Note**
    >
    > 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給執行個體之執行個體設定檔 (適用於 EC2 執行個體) 或 IAM 服務角色 (啟用混合模式的機器) 的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確定與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

14. 在SNS 通知區段中，如果您要傳送有關命令執行狀態的通知，請選取啟用 SNS 通知核取方塊。

    如需為 Run Command 設定 Amazon SNS 通知的詳細資訊，請參閱 使用 Amazon SNS 通知監控 Systems Manager 狀態變更。

15. 當您準備好安裝套件時，請選擇 Run (執行)。

16. Command status (命令狀態) 區域會報告執行的進度。如果命令仍在進行中，請選擇主控台左上角的重新整理圖示，直到 Overall status (整體狀態) 或 Detailed status (詳細狀態) 欄顯示 Success (成功) 或 Failed (失敗) 顯示為止。

17. 在 Targets and outputs (目標和輸出) 區域中，選擇受管節點名稱旁邊的按鈕，然後選擇 View output (檢視輸出)。

    此指令輸出頁面會顯示指令執行的結果。

18. (選用) 如果您選擇將命令輸出寫入 Amazon Simple Storage Service (Amazon S3) 儲存貯體，請選擇 Amazon Simple Storage Service (Amazon S3) 以檢視輸出日誌資料。

使用主控台排程套件安裝或更新

您可以使用 AWS Systems Manager 主控台來排程套件的安裝或更新。排定套件安裝或更新時，Distributor 會使用 AWS Systems Manager State Manager來安裝或更新。

使用主控台排程套件安裝

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Distributor。
3. 在 Distributor 首頁上，選擇您想安裝或安裝的套件。
4. 對於 Package (套件)，選擇 Install on a schedule (依排程安裝)。

   此命令會開啟 State Manager 至為您建立的新關聯。
5. 對於 Name (名稱)，輸入名稱 (例如，**Deploy-test-agent-package**)。此為選用操作，但建議您採用。名稱中不得使用空格。
6. 在 Document (文件) 清單中，已選取文件名稱 `AWS-ConfigureAWSPackage`。
7. 對於 Action (動作)，請確認已選取 Install (安裝)。
8. 對於 Installation type (安裝類型)，選擇以下其中一項：
   - Uninstall and reinstall (解除安裝並重新安裝)：套件會完全解除安裝，然後重新安裝。在重新安裝完成之前，應用程式無法使用。
   - In-place update (就地更新)：根據您在 `update` 指令碼中提供的指示，只將新的或變更的檔案新增至現有的安裝。應用程式在整個更新程序中仍然可用。
9. 對於 Name (名稱)，確認已輸入您套件的名稱。
10. 對於 Version (版本)，如果您要安裝最新發佈版本以外的套件版本，請輸入版本識別碼。
11. 對於 Targets (目標)，請選擇 Selecting all managed instances in this account (選擇此帳戶中的所有受管執行個體)、Specifying tags (指定標籤) 或 Manually Selecting Instance (手動選擇執行個體)。如果您選擇使用標籤來指向資源，請在提供的欄位中輸入標籤索引鍵和標籤值。

    > **Note**
    >
    > 您可以選擇選取此帳戶中的所有受管執行個體或手動選取執行個體，以選擇受管 AWS IoT Greengrass 核心裝置。
12. 對於 Specify schedule (指定排程)，請選擇 On Schedule (依排程) 定期執行關聯，或 No Schedule (不排程) 來執行關聯一次。如需關於這些選項的詳細資訊，請參閱 在 Systems Manager 中使用關聯。使用控制項來建立 `cron` 或關聯速率排程。
13. 選擇 Create Association (建立關聯)。

14. 在 Association (關聯) 頁面上，選擇您建立的關聯旁邊的按鈕，然後選擇 Apply association now (立即套用關聯)。

    State Manager會在指定的目標上建立並立即執行關聯。如需執行關聯結果的詳細資訊，請參閱此指南中的在 Systems Manager 中使用關聯。

如需在 Advanced options (進階選項)、Rate control (速率控制) 以及 Output options (輸出選項) 中使用選項的詳細資訊，請參閱在 Systems Manager 中使用關聯。

使用 安裝套件一次 AWS CLI

您可以在 send-command中執行 AWS CLI ，以安裝Distributor套件一次。如果套件已經安裝，解除安裝套件時會讓應用程式離線，並將新版本安裝就位。

使用 安裝套件一次 AWS CLI

- 在 AWS CLI中執行以下命令。

```
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 的預設行為是 Uninstall and reinstall。當您安裝完整套件時，您可以在此命令中忽略 "installationType":["Uninstall and reinstall"]。

以下是範例。

```
aws ssm send-command \
    --document-name "AWS-ConfigureAWSPackage" \
    --instance-ids "i-00000000000000" \
    --parameters '{"action":["Install"],"installationType":["Uninstall and
 reinstall"],"name":["ExamplePackage"]}'
```

如需有關可與 send-command 命令搭配使用的其他選項的資訊，請參閱 AWS CLI 命令參考的 send-command AWS Systems Manager 一節。

使用 更新套件一次 AWS CLI

您可以在 send-command中執行 AWS CLI 來更新Distributor套件，而不需讓相關聯的應用程式離線。只會取代套件中的新檔案或更新檔案。

使用 更新套件一次 AWS CLI

- 在 AWS CLI中執行以下命令。

```
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**
>
> 當您新增新的或變更的檔案時，必須在命令中包含 `"installationType":["In-place update"]`。

以下是範例。

```
aws ssm send-command \
    --document-name "AWS-ConfigureAWSPackage" \
    --instance-ids "i-02573cafcfEXAMPLE" \
    --parameters '{"action":["Install"],"installationType":["In-place
 update"],"name":["ExamplePackage"]}'
```

如需有關可與 send-command 命令搭配使用的其他選項的資訊，請參閱 AWS CLI 命令參考的 send-command AWS Systems Manager 一節。

使用 排程套件安裝 AWS CLI

您可以在 create-association中執行 AWS CLI ，以按照排程安裝Distributor套件。--name 值 (文件名稱) 一律為 AWS-ConfigureAWSPackage。以下命令會使用索引鍵 InstanceIds 來指定目標受管節點。如果套件已經安裝，解除安裝套件時會讓應用程式離線，並將新版本安裝就位。

```
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 的預設行為是 Uninstall and reinstall。當您安裝完整套件時，您可以在此命令中忽略 "installationType":["Uninstall and reinstall"]。

以下是範例。

```
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\"]}]
```

如需有關可與 create-association 命令搭配使用的其他選項的資訊，請參閱 AWS CLI 命令參考的 create-association AWS Systems Manager 一節。

使用 排程套件更新 AWS CLI

您可以在 create-association中執行 AWS CLI ，以按照排程更新Distributor套件，而無需讓相關聯的應用程式離線。只會取代套件中的新檔案或更新檔案。--name 值 (文件名稱) 一律為 AWS-ConfigureAWSPackage。以下命令會使用索引鍵 InstanceIds 來指定目標執行個體。

```
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**
>
> 當您新增新的或變更的檔案時，必須在命令中包含 "installationType":["In-place update"]。

以下是範例。

```
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\"]}]
```

如需有關可與 create-association 命令搭配使用的其他選項的資訊，請參閱 AWS CLI 命令參考的 create-association AWS Systems Manager 一節。

解除安裝 Distributor 套件

您可以使用 AWS Management Console 或 AWS Command Line Interface (AWS CLI)，透過使用將Distributor套件從 AWS Systems Manager 受管節點解除安裝Run Command。 Distributor和 Run Command是 中的工具 AWS Systems Manager。在此版本中，您可以解除安裝一個套件版本。您可以解除安裝特定版本或預設版本。

> **Important**
>
> 您只能使用 Distributor 解除安裝您使用 Distributor 安裝的套件。否則，Systems Manager 仍然可以將應用程式註冊為 INSTALLED，並導致其他非預期的結果。

主題

- 使用主控台解除安裝套件
- 使用 AWS CLI解除安裝套件

使用主控台解除安裝套件

您可以在 Systems Manager 主控台中使用 Run Command 解除安裝套件一次。Distributor 使用 AWS Systems Manager Run Command 來解除安裝套件。

使用主控台解除安裝套件

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Run Command。
3. 在 Run Command 首頁選擇 Run command (執行命令)。
4. 選擇 `AWS-ConfigureAWSPackage` 命令文件。
5. 從 Action (動作)，選擇 Uninstall (解除安裝)
6. 在 Name (名稱) 輸入您想解除安裝的套件名稱。
7. 對於 Targets (目標)，選擇您想要如何鎖定受管節點。您可以指定目標共用的標籤索引鍵和值。您也可以選擇屬性來指定目標，屬性包括 ID、平台和 SSM Agent 版本等。
8. 您可以使用進階選項來新增有關作業的註解、變更 Rate control (比率控制) 中的 Concurrency (並行) 和 Error threshold (錯誤閾值) 值、指定輸出選項，或設定 Amazon Simple Notification Service (Amazon SNS) 通知。如需詳細資訊，請參閱此指南中的從主控台執行命令。
9. 準備好解除安裝套件時，請選擇 Run (執行)，然後選擇 View results (檢視結果)。
10. 在命令清單中，選擇您剛剛執行的 `AWS-ConfigureAWSPackage` 命令。如果命令仍在進行中，選擇主控台右上角的重新整理圖示。
11. Status (狀態) 列顯示 Success (成功) 或 Failed (失敗) 時，請選擇 Output (輸出) 標籤。
12. 選擇 View output (檢視輸出)。此指令輸出頁面會顯示指令執行的結果。

使用 AWS CLI解除安裝套件

您可以使用 從受管節點 AWS CLI 解除安裝Distributor套件，方法是使用 Run Command。

使用 解除安裝套件 AWS CLI

- 在 AWS CLI中執行以下命令。

```
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)"]}'
```

以下是範例。

```
aws ssm send-command \
    --document-name "AWS-ConfigureAWSPackage" \
    --instance-ids "i-02573cafcfEXAMPLE" \
    --parameters '{"action":["Uninstall"],"name":["Test-ConfigureAWSPackage"]}'
```

如需有關可與 send-command 命令搭配使用的其他選項的資訊，請參閱 AWS CLI 命令參考的 send-command AWS Systems Manager 一節。

刪除 Distributor 套件

本節說明如何刪除套件。您無法刪除套件的版本，只能刪除整個套件。

使用主控台刪除套件

您可以使用 AWS Systems Manager 主控台，從 中的Distributor工具 中刪除套件或套件版本 AWS Systems Manager。刪除套件會從 Distributor 刪除所有套件版本。

使用主控台刪除套件

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Distributor。
3. 在 Distributor 首頁上選擇您想刪除的套件。
4. 在套件詳細資訊頁面上，選擇 Delete package (刪除套件)。
5. 當系統提示您確認刪除時，選擇 Delete package (刪除套件)。

使用主控台刪除套件版本

您可以使用 Systems Manager 主控台從 Distributor 刪除套件版本。

使用主控台刪除套件版本

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。

2. 在導覽窗格中，選擇 Distributor。
3. 在 Distributor 首頁上選擇您想刪除其版本的套件。
4. 在套件的版本頁面上，選擇要刪除的版本，然後選擇 Delete version (刪除版本)。
5. 當系統提示您確認刪除時，選擇 Delete package version (刪除套件版本)。

使用命令列刪除套件

您可以使用偏好的命令列工具，從 Distributor 刪除套件。

Linux & macOS

使用 刪除套件 AWS CLI

1. 執行以下命令，列出特定套件的文件。在這個命令的結果中，尋找您要刪除的套件。

```
aws ssm list-documents \
    --filters Key=Name,Values=package-name
```

2. 執行以下命令以刪除套件。用套件名稱取代 *package_name*。

```
aws ssm delete-document \
    --name "package-name"
```

3. 再次執行 list-documents 命令，確認套件已刪除。您刪除的套件不應包含在清單中。

```
aws ssm list-documents \
    --filters Key=Name,Values=package-name
```

Windows

使用 刪除套件 AWS CLI

1. 執行以下命令，列出特定套件的文件。在這個命令的結果中，尋找您要刪除的套件。

```
aws ssm list-documents ^
    --filters Key=Name,Values=package-name
```

2. 執行以下命令以刪除套件。用套件名稱取代 *package_name*。

```
aws ssm delete-document ^
    --name "package-name"
```

3. 再次執行 list-documents 命令，確認套件已刪除。您刪除的套件不應包含在清單中。

```
aws ssm list-documents ^
    --filters Key=Name,Values=package-name
```

PowerShell

使用 Tools for PowerShell 刪除套件

1. 執行以下命令，列出特定套件的文件。在這個命令的結果中，尋找您要刪除的套件。

```
$filter = New-Object
 Amazon.SimpleSystemsManagement.Model.DocumentKeyValuesFilter
$filter.Key = "Name"
$filter.Values = "package-name"

Get-SSMDocumentList `
    -Filters @($filter)
```

2. 執行以下命令以刪除套件。用套件名稱取代 *package_name*。

```
Remove-SSMDocument `
    -Name "package-name"
```

3. 再次執行 Get-SSMDocumentList 命令，確認套件已刪除。您刪除的套件不應包含在清單中。

```
$filter = New-Object
 Amazon.SimpleSystemsManagement.Model.DocumentKeyValuesFilter
$filter.Key = "Name"
$filter.Values = "package-name"

Get-SSMDocumentList `
    -Filters @($filter)
```

使用命令列刪除套件版本

您可以使用偏好的命令列工具，從 Distributor 刪除套件版本。

Linux & macOS

使用 刪除套件版本 AWS CLI

1. 執行下列命令以列出套件的版本。在這個命令的結果中，尋找您要刪除的套件版本。

```
aws ssm list-document-versions \
    --name "package-name"
```

2. 執行以下命令以刪除套件版本。用套件名稱取代 *package-name*，以及用版本號碼取代 *version*。

```
aws ssm delete-document \
    --name "package-name" \
    --document-version version
```

3. 執行 list-document-versions 命令，確認套件版本已刪除。應該找不到您刪除的套件版本。

```
aws ssm list-document-versions \
    --name "package-name"
```

Windows

使用 刪除套件版本 AWS CLI

1. 執行下列命令以列出套件的版本。在這個命令的結果中，尋找您要刪除的套件版本。

```
aws ssm list-document-versions ^
    --name "package-name"
```

2. 執行以下命令以刪除套件版本。用套件名稱取代 *package-name*，以及用版本號碼取代 *version*。

```
aws ssm delete-document ^
    --name "package-name" ^
    --document-version version
```

3. 執行 list-document-versions 命令，確認套件版本已刪除。應該找不到您刪除的套件版本。

```
aws ssm list-document-versions ^
    --name "package-name"
```

PowerShell

使用 Tools for PowerShell 刪除套件版本

1. 執行下列命令以列出套件的版本。在這個命令的結果中，尋找您要刪除的套件版本。

```
Get-SSMDocumentVersionList `
    -Name "package-name"
```

2. 執行以下命令以刪除套件版本。用套件名稱取代 *package-name*，以及用版本號碼取代 *version*。

```
Remove-SSMDocument `
    -Name "package-name" `
    -DocumentVersion version
```

3. 執行 Get-SSMDocumentVersionList 命令，確認套件版本已刪除。應該找不到您刪除的套件版本。

```
Get-SSMDocumentVersionList `
    -Name "package-name"
```

如需有關可與 list-documents 命令搭配使用的其他選項的資訊，請參閱 AWS CLI 命令參考的 list-documents AWS Systems Manager 一節。如需可搭配 delete-document 命令使用之其他選項的相關資訊，請參閱delete-document。

稽核和記錄 Distributor 活動

您可以使用 AWS CloudTrail 來稽核與 中的Distributor工具 相關的活動 AWS Systems Manager。如需稽核和記錄 Systems Manager 選項的詳細資訊，請參閱 在 中記錄和監控 AWS Systems Manager。

使用 CloudTrail 稽核 Distributor 活動

CloudTrail 會擷取在 AWS Systems Manager 主控台、 AWS Command Line Interface (AWS CLI) 和 Systems Manager SDK 中發出的 API 呼叫。這些資訊可以在 CloudTrail 主控台中檢視，或存放在 Amazon Simple Storage Service (Amazon S3) 儲存貯體中。一個儲存貯體可用於您帳戶中所有 CloudTrail 日誌。

Run Command 和 State Manager動作的日誌會顯示文件建立、套件安裝和套件解除安裝活動。Run Command和 State Manager是 中的工具 AWS Systems Manager。如需檢視和使用 Systems Manager 活動的 CloudTrail 日誌的詳細資訊，請參閱 使用 記錄 AWS Systems Manager API 呼叫 AWS CloudTrail。

疑難排解 AWS Systems ManagerDistributor

以下資訊可協助您針對使用 工具時可能發生的問題Distributor進行故障診斷 AWS Systems Manager。

主題

- 安裝了相同名稱的錯誤套件
- 錯誤：無法擷取資訊清單：找不到套件的最新版本
- 錯誤：無法擷取資訊清單：驗證異常
- 不支援套件 (套件缺少安裝動作)
- 錯誤：無法下載資訊清單：具有名稱的文件不存在
- 上傳失敗。
- 錯誤：找不到平台：找不到平台的資訊清單：oracle，8.9 版，架構 x86_64

安裝了相同名稱的錯誤套件

問題：您已安裝套件，但 Distributor 改為安裝不同的套件。

原因：安裝時，Systems Manager 會尋找 AWS 發佈的套件做為結果，優先於使用者定義的外部套件。如果您的使用者定義套件名稱與 AWS 已發佈的套件名稱相同， AWS 則會安裝套件，而非您的套件。

解決方案：為避免此問題，請將套件命名為與 AWS 已發佈套件名稱不同的名稱。

錯誤：無法擷取資訊清單：找不到套件的最新版本

問題：您收到了以下錯誤：

```
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 搭配 Distributor 版本早於 2.3.274.0 版本。

解決方案：將 SSM Agent 的版本更新到 2.3.274.0 版本或更新版本。如需詳細資訊，請參閱 使用 Run Command 更新 SSM Agent 或 逐步解說：SSM Agent使用 自動更新 AWS CLI 。

錯誤：無法擷取資訊清單：驗證異常

問題：您收到了以下錯誤：

```
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 搭配 Distributor 版本早於 2.3.274.0 版本。

解決方案：將 SSM Agent 的版本更新到 2.3.274.0 版本或更新版本。如需詳細資訊，請參閱 使用 Run Command 更新 SSM Agent 或 逐步解說：SSM Agent使用 自動更新 AWS CLI 。

不支援套件 (套件缺少安裝動作)

問題：您收到了以下錯誤：

```
Package is not supported (package is missing install action)
```

原因：套件目錄結構不正確。

解決方案：不要壓縮包含軟件和所需指令碼的父目錄。相反地，直接在絕對路徑中建立全部所需內容的 .zip 檔案。驗證是否正確建立 .zip 檔案，請解壓縮目標平台目錄並檢閱目錄結構。例如，安裝指令碼絕對路徑應該是 /*ExamplePackage_targetPlatform*/install.sh。

錯誤：無法下載資訊清單：具有名稱的文件不存在

問題：您收到了以下錯誤：

```
Failed to download manifest - failed to retrieve package document description:
 InvalidDocument: Document with name filename does not exist.
```

原因 1：共用來自另一個帳戶的 Distributor 套件時，Distributor 無法透過套件名稱找到套件。

解決方案 1：共用另一個帳戶的套件時，使用套件的完整 Amazon Resource Name (ARN)，而不是只使用其名稱。

原因 2：使用 VPC 時，您尚未提供 IAM 執行個體描述檔存取權給 AWS 受管 S3 儲存貯體，其中包含 AWS 區域 目標 `AWS-ConfigureAWSPackage`的文件。

解決方案 2：確保您的 IAM 執行個體描述檔SSM Agent提供受 AWS 管 S3 儲存貯體的存取權，其中包含 AWS 區域 您要鎖定`AWS-ConfigureAWSPackage`的 文件，如 中所述SSM Agent 與 AWS 受管 S3 儲存貯體通訊。

上傳失敗。

問題：您收到了以下錯誤：

```
Upload failed. At least one of your files was not successfully uploaded to your S3
 bucket.
```

原因：軟體套件的名稱包含空格。例如，`Hello World.msi` 將無法上傳。

錯誤：找不到平台：找不到平台的資訊清單：oracle，8.9 版，架構 x86_64

問題：您收到了以下錯誤：

```
Failed to find platform: no manifest found for platform: oracle, version 8.9,
 architecture x86_64
```

原因：錯誤表示 JSON 套件資訊清單對該平台沒有任何定義，在此情況下為 Oracle Linux。

解決方案：從 Trend Micro Deep Security Software 網站下載您要分發的套件。使用 Distributor Simple 工作流程程序建立`.rpm`軟體套件。為套件設定下列值，並使用 Distributor 完成上傳軟體套件：

```
Platform version: _any
Target platform: oracle
Architecture: x86_64
```

AWS Systems Manager Fleet Manager

Fleet Manager是 中的工具 AWS Systems Manager，是一種統一的使用者介面 (UI) 體驗，可協助您遠端管理在內部部署 AWS 或內部部署上執行的節點。利用 Fleet Manager，您可以從單一主控台檢視整個伺服器機群的運作狀態和效能狀態。您也可以從個別節點收集資料，進而從主控台執行常見的故障診

斷和管理任務。其中包含使用遠端桌面通訊協定 (RDP) 連線至 Windows 執行個體、檢視資料夾和檔案內容、Windows 登錄管理、作業系統使用者管理等。

若要開始使用 Fleet Manager，請開啟 Systems Manager 主控台。在導覽窗格中，選擇 Fleet Manager。

誰應該使用Fleet Manager？

任何想要集中管理節點機群 AWS 的客戶都應該使用 Fleet Manager。

Fleet Manager 如何為我的組織帶來益處？

Fleet Manager 提供這些好處：

- 執行各種常見的系統管理任務，而不必手動連線到受管節點。
- 從單一的統一主控台管理在多個平台上執行的節點。
- 從單一的統一主控台管理執行不同作業系統的節點。
- 提高系統管理的效率。

Fleet Manager 有哪些功能？

Fleet Manager 的重要功能如下所示：

- 存取 Red Hat 知識庫入口網站

  透過您的 Red Hat Enterprise Linux (RHEL) 執行個體，在 Red Hat 知識庫入口網站上存取二進位檔、知識分享和討論區。
- 受管節點狀態

  檢視哪些受管執行個體是 `running` 以及哪些是 `stopped`。如需有關已停止執行個體的詳細資訊，請參閱《Amazon EC2 使用者指南》中的停止和啟動執行個體。對於 AWS IoT Greengrass 核心裝置，您可以檢視哪些是 `online`、 `offline`或 顯示 的狀態`Connection lost`。

  > **Note**
  >
  > 如果您在 2021 年 7 月 12 日之前停止受管執行個體，則不會顯示 `stopped` 標記。若要顯示標記，請啟動和停止執行個體。

- 檢視執行個體資訊

檢視存放在連接至受管執行個體之磁碟區上的資料夾和檔案資料的相關資訊、即時執行個體的效能資料，以及存放在執行個體上的記錄資料。

- 檢視邊緣裝置資訊

  檢視裝置的 AWS IoT Greengrass 物件名稱、SSM Agentping 狀態和版本等。

- 管理帳戶和登錄

  在您的 Windows 執行個體的執行個體和登錄上管理作業系統 (OS) 使用者帳戶。

- 控制對功能的存取

  使用 AWS Identity and Access Management (IAM) 政策控制對 Fleet Manager 功能的存取。透過這些政策，您可以控制組織中的哪些個別使用者或群組可以使用各種 Fleet Manager 功能，以及其可管理哪些受管節點。

主題

- 設定 Fleet Manager
- 使用受管節點
- 使用預設主機管理組態來自動管理 EC2 執行個體
- 使用 Remote Desktop 連線至 Windows Server 受管執行個體
- 在受管執行個體上管理 Amazon EBS 磁碟區
- 存取 Red Hat 知識庫入口網站
- 疑難排解受管節點的可用性

設定 Fleet Manager

在 中的使用者 AWS 帳戶 可以使用 Fleet Manager中的工具 AWS Systems Manager來監控和管理受管節點之前，必須先授予他們必要的許可。此外，任何 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體， AWS IoT Greengrass 核心裝置，以及要使用 Fleet Manager 監控和管理的內部部署伺服器、邊緣裝置和虛擬機器 (VM)，都必須是 Systems Manager 受管節點。受管節點是設定為在混合多雲端環境中搭配 Systems Manager 使用的任何機器。

這表示您的節點必須符合特定先決條件，並使用 AWS Systems Manager Agent (SSM Agent) 進行設定。

根據機器類型參閱下列其中一個主題，確保機器符合受管節點的需求。

- Amazon EC2 執行個體：使用 Systems Manager 管理 EC2 執行個體

> **Tip**
>
> 您也可以使用 中的Quick Setup工具 AWS Systems Manager，協助您快速將 Amazon EC2 執行個體設定為個別帳戶中的受管執行個體。如果您的企業或組織使用 AWS Organizations，您也可以跨多個組織單位 OUs) 和 設定執行個體 AWS 區域。如需使用 Quick Setup 設定受管執行個體的詳細資訊，請參閱 使用 Quick Setup 設定 Amazon EC2 主機管理。

- 雲端中的內部部署和其他伺服器類型：使用 Systems Manager 在混合多雲端環境中管理節點
- AWS IoT Greengrass （邊緣） 裝置： 使用 Systems Manager 管理邊緣裝置

範例政策

- 控制 Fleet Manager 存取

控制 Fleet Manager 存取

若要使用 Fleet Manager中的工具 AWS Systems Manager，您的 AWS Identity and Access Management (IAM) 使用者或角色必須具有必要的許可。您可以建立可存取所有 Fleet Manager 功能的 IAM 政策，或修改您的政策以授予您選擇的功能存取權。然後，將這些許可授予帳戶中的使用者或身分。

任務 1：建立 IAM 政策以定義存取許可

依照《IAM 使用者指南》中的下列主題提供的其中一種方法來建立 IAM，向身分 (使用者、角色或使用者群組) 提供對 Fleet Manager 的存取權：

- 使用客戶管理政策定義自訂 IAM 許可

可以使用下方提供的其中一個範例政策，或根據想要授予的許可修改這些政策。我們提供完整存取和唯讀存取 Fleet Manager 的範例政策。

任務 2：將 IAM 政策連接至使用者以授予許可

建立定義 Fleet Manager 存取許可的 IAM 政策或政策後，請使用《IAM 使用者指南》中的下列其中一種程序，將這些許可授予帳戶中的身分：

- 新增 IAM 身分許可 (主控台)
- 新增 IAM 身分許可 (AWS CLI)

- 新增 IAM 身分許可 (AWS API)

主題

- Fleet Manager 管理員存取權的範例政策
- Fleet Manager 唯讀存取權的範例政策

Fleet Manager 管理員存取權的範例政策

下列政策可提供所有 Fleet Manager 功能的許可。這表示使用者可以建立和刪除本機使用者和群組、修改任何本機群組的群組成員資格，以及修改 Windows Server 登錄索引鍵或值。將每個########取代為您自己的資訊。

```
{
   "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:region:account-id:servicesetting/ssm/managed-
instance/default-ec2-instance-management-role"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::account-id: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:*:account-id:instance/*",
            "arn:aws:ssm:*:account-id:managed-instance/*",
            "arn:aws:ssm:*:account-id: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 唯讀存取權的範例政策

下列政策可提供唯讀 Fleet Manager 功能的許可。將每個########取代為您自己的資訊。

```
{
   "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:*:account-id:instance/*",
            "arn:aws:ssm:*:account-id:managed-instance/*",
            "arn:aws:ssm:*:account-id: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}"
               ]
            }
         }
      }
   ]
}
```

使用受管節點

受管節點是針對 設定的任何機器 AWS Systems Manager。您可以將下列機器類型設定為受管節點：

- Amazon Elastic Compute Cloud (Amazon EC2) 執行個體
- 您內部部署的伺服器 (內部部署伺服器)
- AWS IoT Greengrass 核心裝置

- AWS IoT 和非AWS 邊緣裝置
- 虛擬機器 (VM)，包含其他雲端環境中的 VM

在 Systems Manager 主控台，任何字首為 "mi-" 的機器都已使用混合啟動設定為受管節點。Edge 裝置會顯示其 AWS IoT 物件名稱。

> **Note**
>
> macOS 執行個體唯一支援的功能是檢視檔案系統。

關於 Systems Manager 執行個體方案

AWS Systems Manager 提供 standard-instances 層和 advanced-instances 層。兩者都支援混合多雲端環境中的受管節點。Standard-instances 方案可讓您 AWS 帳戶 為每個機器註冊最多 1，000 個機器 AWS 區域。如果您需要在單一帳戶和區域中登錄 1,000 部以上的機器，則使用進階執行個體層。您可以在進階執行個體層中，視需要建立多個受管節點。所有針對 Systems Manager 設定的受管節點，按使用次數計價。如需啟用進階執行個體方案的詳細資訊，請參閱 開啟 advanced-instances 方案。如需定價的詳細資訊，請參閱AWS Systems Manager 定價。

請注意下列有關標準執行個體方案和進階執行個體方案的其他資訊：

- 進階執行個體也可讓您使用 連線到混合多雲端環境中non-EC2 節點 AWS Systems Manager Session Manager。 Session Manager 提供執行個體的互動式 shell 存取。如需詳細資訊，請參閱AWS Systems Manager Session Manager。
- 此 standard-instances 配額也適用於使用 Systems Manager 內部部署啟用的 EC2 執行個體 (這不是常見案例)。
- 若要修補 Microsoft 在虛擬機器 (VM) 內部部署執行個體上發行的應用程式，請啟用進階執行個體層。使用進階執行個體層會產生費用。修補由 Microsoft 在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上發行的應用程式無須另外付費。如需詳細資訊，請參閱在 Windows Server 上由 Microsoft 發行的修補應用程式。

顯示受管節點

如果您沒有看到您的受管節點列在主控台中，請執行以下各項：

1. 確認您 AWS 區域 建立受管節點的 中，主控台已開啟。您可以使用主控台頂端右上角的清單切換區域。

2. 確認受管節點的設定步驟滿足 Systems Manager 的要求。如需相關資訊，請參閱 設定 AWS Systems Manager的受管節點。
3. 對於非 EC2 機器，請確認您是否已完成混合啟用程序。如需詳細資訊，請參閱使用 Systems Manager 在混合多雲端環境中管理節點。

請注意以下更多資訊：

- Fleet Manager 主控台不會顯示已終止的 Amazon EC2 節點。
- Systems Manager 需要準確的時間參考才能執行在機器上的操作。如果您受管節點上的日期和時間未正確設定，則機器可能會與您 API 請求的簽章日期不符。如需詳細資訊，請參閱使用案例與最佳實務。
- 建立或編輯標籤時，系統最多可能需要一小時才能在資料表篩選條件中顯示變更。
- 在受管節點的狀態成為 `Connection Lost` 的時間至少 30 天之後，該節點可能不會再列於 Fleet Manager 主控台中。若要將其還原至清單，必須解決造成連線中斷的問題。如需疑難排解秘訣，請參閱 疑難排解受管節點的可用性。

確認受管節點上的 Systems Manager 支援

AWS Config 提供 AWS 受管規則，這是預先定義的可自訂規則， AWS Config 用於評估 AWS 資源組態是否符合常見的最佳實務。 AWS Config 受管規則包含 ec2-instance-managed-by-systems-manager 規則。此規則會檢查您帳戶中的 Amazon EC2 執行個體是否由 Systems Manager 管理。如需詳細資訊，請參閱 AWS Config 受管服務。

提升受管節點的安全狀態

如需提升安全狀態，防範受管節點上未獲授權的根層級命令的詳細資訊，請參閱 限制透過 SSM Agent 存取根層級命令

取消註冊受管節點

您可隨時取消註冊受管節點。例如，如果您使用相同的 AWS Identity and Access Management (IAM) 角色管理多個節點，並且注意到任何類型的惡意行為，您可以隨時取消註冊任意數量的機器。（若要重新註冊相同的機器，您必須使用與先前註冊相同的混合啟用碼和啟用 ID。）如需取消註冊受管節點的詳細資訊，請參閱 取消註冊混合多雲端環境中的受管節點。

主題

- 設定執行個體方案
- 在受管節點上重設密碼

- 取消註冊混合多雲端環境中的受管節點
- 透過 Fleet Manager 使用作業系統檔案系統
- 監控受管節點效能
- 處理程序
- 檢視受管節點上的日誌
- 使用 Fleet Manager 管理受管節點上的作業系統使用者帳戶和群組
- 管理受管節點上的 Windows 登錄檔

設定執行個體方案

本主題說明您必須啟用進階執行個體層的案例。

AWS Systems Manager 為混合多雲端環境中non-EC2 機器提供標準執行個體層和進階執行個體層。???

每個帳戶最多可註冊 1，000 個標準混合啟用節點 AWS 區域 ，無需額外費用。不過，登錄超過 1,000 個混合節點需要啟用進階執行個體層。使用 advanced-instance 方案會產生費用。如需詳細資訊，請參閱 AWS Systems Manager 定價。

即使登錄的啟用混合模式節點少於 1,000 個，出現其他兩個情況時依然需要使用 advanced-instances 方案：

- 您想要使用 Session Manager 連接到非 EC2 節點。
- 您想要修補 Microsoft 在非 EC2 節點上發行的應用程式 (而非作業系統)。

> **Note**
>
> 修補由 Microsoft 在 Amazon EC2 執行個體上發行的應用程式無需另外付費。

進階執行個體層詳細案例

下列資訊提供您必須啟用進階執行個體層的三種案例的詳細資訊。

情況 1：您想要登錄超過 1,000 個啟用混合模式節點

使用 Standard-instances 層，您可以在特定帳戶中的每個混合多雲端環境中註冊最多 1，000 個 non-EC2 AWS 區域 節點，無需額外費用。如果您需要在區域中登錄超過 1,000 個非 EC2 節點，

則必須使用進階執行個體層。然後，您可以於混合多雲端環境中啟用所需數目的機器。advanced-instances 方案會根據作為 Systems Manager 受管節點啟用的進階節點數目，以及這些節點執行的時數計費。

如果在特定帳戶中某個區域中有超過 1,000 個內部部署節點，則根據建立混合啟用以向 Systems Manager 註冊節點中的描述，使用啟用程序的所有 Systems Manager 受管節點需付費。

> **Note**
>
> 您也可以使用 Systems Manager 混合啟用來啟用現有的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體，並將其用作非 EC2 執行個體，例如為了進行測試。這些也符合作為混合節點的資格。這不是一個常見的案例。

案例 2：修補混合啟用節點上 Microsoft 發行的應用程式

如果您想要在混合多雲端環境中的非 EC2 節點上修補 Microsoft 發行的應用程式，則也需要 advanced-instances 方案。如果您啟用進階執行個體層來修補非 EC2 節點上的 Microsoft 應用程式，則即使您的數量少於 1,000 個，所有內部部署節點仍會產生費用。

修補由 Microsoft 在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上發行的應用程式無須另外付費。如需詳細資訊，請參閱在 Windows Server 上由 Microsoft 發行的修補應用程式。

案例 3：使用 Session Manager 連線到混合啟用節點

Session Manager 提供對執行個體的互動式 shell 存取。若要使用 Session Manager 連線到啟用混合模式的受管節點，您必須先啟用 advanced-instances 方案。然後，即使您的數量少於 1,000 個，所有混合啟用節點也會產生費用。

摘要：何時需要進階執行個體層？

請使用下表檢閱何時必須使用進階執行個體層，以及哪些案例需額外收費。

案例	是否需要進階執行個體層？	需支付額外費用？
特定帳戶中我的區域的混合啟用節點數量超過 1,000 個。	是	是
我想使用 Patch Manager 在任意數量的混合啟用節點 (即使少	是	是

案例	是否需要進階執行個體層？	需支付額外費用？
於 1,000 個) 上修補 Microsoft 發行的應用程式。		
我想使用 Session Manager 連線任意數量的混合啟用節點 (即使少於 1,000 個)。	是	是
1. 特定帳戶中某個區域中混合啟用節點的數量為 1,000 或以下；以及 2. 我沒有在任何混合啟用節點上修補 Microsoft 應用程式；以及 3. 我沒有使用 Session Manager 連線到任何混合啟用節點。	否	否

主題

- 開啟 advanced-instances 方案
- 從進階執行個體層還原至標準執行個體層

開啟 advanced-instances 方案

AWS Systems Manager 為混合多雲端環境中non-EC2 機器提供標準執行個體層和進階執行個體層。???Standard-instances 方案可讓您 AWS 帳戶 為每個 註冊最多 1，000 個混合啟用機器 AWS 區域。進階執行個體層也必須使用 Patch Manager 在非 EC2 節點上修補 Microsoft 發行的應用程式，並使用 Session Manager 連線到非 EC2 節點。如需詳細資訊，請參閱開啟 advanced-instances 方案。

此部分說明如何設定您的混合多雲端環境，以使用 advanced-instances 方案。

開始之前

查看進階執行個體的定價詳細資訊。進階執行個體以計次付費的形式提供。如需詳細資訊，請參閱 AWS Systems Manager 定價。

設定許可來開啟 advanced-instances 方案

確認您擁有在 AWS Identity and Access Management (IAM) 中將環境從 standard-instances 層變更為 advanced-instances 層的許可。您必須將 `AdministratorAccess` IAM 政策連接到使用者、群組或角色，或者您必須擁有變更 Systems Manager 啟用層服務設定的許可。啟用方案設定會使用以下 API 操作：

- GetServiceSetting
- UpdateServiceSetting
- ResetServiceSetting

請使用下列程序，將內嵌 IAM 政策新增至使用者帳戶。此政策允許使用者檢視目前的受管執行個體方案設定。此政策也允許使用者變更或重設指定 AWS 帳戶 和 中的目前設定 AWS 區域。

1. 登入 AWS Management Console ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇使用者 。
3. 在清單中，選擇要內嵌政策的使用者名稱。
4. 選擇許可索引標籤標籤。
5. 在頁面右邊的 Permission policies (許可政策) 下選擇 Add inline policy (新增內嵌政策)。
6. 選擇 JSON 標籤。
7. 將預設內容取代為以下內容：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetServiceSetting"

            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:ResetServiceSetting",
```

```
                "ssm:UpdateServiceSetting"
            ],
            "Resource": "arn:aws:ssm:region:aws-account-id:servicesetting/ssm/
managed-instance/activation-tier"
        }
    ]
}
```

8. 選擇檢閱政策。
9. 在檢閱政策頁面上名稱中，輸入該內嵌政策的名稱。例如：**Managed-Instances-Tier**。
10. 選擇 建立政策。

管理員可以將下列內嵌政策指派給使用者，以指定唯讀許可。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetServiceSetting"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Deny",
            "Action": [
                "ssm:ResetServiceSetting",
                "ssm:UpdateServiceSetting"
            ],
            "Resource": "*"
        }
    ]
}
```

如需有關建立和編輯 IAM 政策的詳細資訊，請參閱《IAM 使用者指南》中的建立 IAM 政策。

開啟 advanced-instances 方案 (主控台)

下列程序說明如何使用 Systems Manager 主控台，在指定的 AWS 帳戶 和 中，變更使用受管執行個體啟用新增的所有non-EC2 節點 AWS 區域，以使用 advanced-instances 層。

開始之前

確認您 AWS 區域 建立受管執行個體的 中，主控台已開啟。您可以使用主控台頂端右上角的清單切換區域。

確認您已完成混合多雲端環境中 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體和非 EC2 機器的設定要求。如需相關資訊，請參閱 設定 AWS Systems Manager的受管節點。

> **Important**
>
> 下列程序說明如何變更帳戶層級的設定。此變更會向您的帳戶收取費用。

若要開啟 advanced-instances 方案 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 依次選擇設定、變更執行個體方案設定。
4. 檢閱對話方塊中有關變更帳戶設定的資訊。
5. 如果您核准，請選擇要接受的選項，然後選擇變更設定。

系統可能需要幾分鐘的時間才能完成將所有執行個體從 standard-instances 方案移動到 advanced-instances 方案的程序。

> **Note**
>
> 如需變更回 standard-instances 方案的相關資訊，請參閱從進階執行個體層還原至標準執行個體層。

開啟 advanced-instances 方案 (AWS CLI)

下列程序說明如何使用 AWS Command Line Interface 來變更使用指定 和 中的受管執行個體啟用新增的所有現場部署伺服器 AWS 帳戶 和 VMs AWS 區域，以使用 advanced-instances 層。

> **Important**
>
> 下列程序說明如何變更帳戶層級的設定。此變更會向您的帳戶收取費用。

使用 開啟 advanced-instances 方案 AWS CLI

1. 開啟 AWS CLI 並執行下列命令。將每個########取代為您自己的資訊。

   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
   ```

   如果命令成功，則無輸出訊息。

2. 執行下列命令，以檢視目前 AWS 帳戶 和 中受管節點的目前服務設定 AWS 區域。

   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
   ```

該命令會傳回相關資訊，如以下所示。

```
{
    "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"
    }
}
```

開啟 advanced-instances 方案 (PowerShell)

下列程序說明如何使用 AWS Tools for Windows PowerShell 來變更使用指定 和 中的受管執行個體啟用新增的所有現場部署伺服器 AWS 帳戶 和 VMs AWS 區域，以使用 advanced-instances 層。

> **Important**
>
> 下列程序說明如何變更帳戶層級的設定。此變更會向您的帳戶收取費用。

若要使用 PowerShell 開啟 advanced-instances 方案

1. 開啟 AWS Tools for Windows PowerShell 並執行下列命令。將每個########取代為您自己的資訊。

   ```
   Update-SSMServiceSetting `
       -SettingId "arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-
   instance/activation-tier" `
       -SettingValue "advanced"
   ```

   如果命令成功，則無輸出訊息。

2. 執行下列命令，以檢視目前 AWS 帳戶 和 中受管節點的目前服務設定 AWS 區域。

```
Get-SSMServiceSetting `
    -SettingId "arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-
instance/activation-tier"
```

該命令會傳回相關資訊，如以下所示。

```
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
```

系統可能需要幾分鐘的時間才能完成將所有節點從 standard-instances 方案移動到 advanced-instances 方案的程序。

> **Note**
>
> 如需變更回 standard-instances 方案的相關資訊，請參閱從進階執行個體層還原至標準執行個體層。

從進階執行個體層還原至標準執行個體層

本節說明如何將以 advanced-instances 方案執行的啟用混合模式的節點變更回以 standard-instances 方案執行。此組態適用於 AWS 帳戶 和單一 中的所有混合啟用節點 AWS 區域。

開始之前

檢閱下列重要詳細資訊。

> **Note**
>
> - 如果您在帳戶和區域中執行超過 1,000 個啟用混合模式的節點，則無法還原至 standard-instances 方案。您必須先取消註冊節點，直到 1,000 個或更少。這也適用於使用 Systems Manager 混合啟用的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體 (這不是常見案例)。如需詳細資訊，請參閱取消註冊混合多雲端環境中的受管節點。

- 還原之後，您將無法使用 中的Session Manager工具 AWS Systems Manager，以互動方式存取混合啟用節點。
- 還原後，您將無法使用 中的Patch Manager工具 AWS Systems Manager來修補 Microsoft 在混合啟用節點上發行的應用程式。
- 將所有啟用混合模式的節點還原至 standard-instance 方案的程序，可能需要 30 分鐘以上才能完成。

本節說明如何將 AWS 帳戶 和 中的所有混合啟用節點 AWS 區域 從 advanced-instances 層還原為 standard-instances 層。

還原至 Standard-Instances 方案 (主控台)

下列程序說明如何使用 Systems Manager 主控台，將混合和多雲端環境中所有混合啟用節點變更為使用指定 AWS 帳戶 和 中的標準執行個體層 AWS 區域。

還原至 standard-instances 方案 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選取 Account settings (帳戶設定) 下拉式選單，並選擇 Instance tier settings (執行個體方案設定)。
4. 選擇 Change account setting (變更帳戶設定)。
5. 在有關變更帳戶設定的彈出式視窗中檢閱資訊，接下來如果您同意的話，選擇要接受的選項，並繼續進行。

還原至 Standard-Instances 方案 (AWS CLI)

下列程序說明如何使用 AWS Command Line Interface 來變更混合和多雲端環境中所有混合啟用節點，以使用指定 AWS 帳戶 和 中的標準執行個體層 AWS 區域。

使用 還原至標準執行個體層 AWS CLI

1. 開啟 AWS CLI 並執行下列命令。將每個########取代為您自己的資訊。

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
```

如果命令成功，則無輸出訊息。

2. 30 分鐘後執行下列命令，以檢視目前 AWS 帳戶 和 中受管執行個體的設定 AWS 區域。

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
```

該命令會傳回相關資訊，如以下所示。

```
{
    "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"
    }
}
```

核准請求後，狀態會變更為 Default。

還原至 Standard-Instances 方案 (PowerShell)

下列程序說明如何使用 AWS Tools for Windows PowerShell 來變更混合和多雲端環境中的混合啟用節點，以使用指定 AWS 帳戶 和 中的標準執行個體層 AWS 區域。

使用 PowerShell 還原至 standard-instances 方案

1. 開啟 AWS Tools for Windows PowerShell 並執行下列命令。

```
Update-SSMServiceSetting `
    -SettingId "arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-
instance/activation-tier" `
    -SettingValue "standard"
```

如果命令成功，則無輸出訊息。

2. 30 分鐘後執行下列命令，以檢視目前 AWS 帳戶 和 中受管執行個體的設定 AWS 區域。

```
Get-SSMServiceSetting `
    -SettingId "arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-
instance/activation-tier"
```

該命令會傳回相關資訊，如以下所示。

```
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
```

核准請求後，狀態會變更為 Default。

在受管節點上重設密碼

您可以在受管節點上為任何使用者重設密碼。這包括 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體； AWS IoT Greengrass 核心裝置；以及由 管理的內部部署伺服器、邊緣裝置和虛擬機器 VMs) AWS Systems Manager。密碼重設功能建立在 上Session Manager，這是 中的工具 AWS Systems Manager。您可以使用此功能，不需開啟傳入連接埠、維持堡壘主機或管理 SSH 金鑰，即可連接至受管節點。

密碼重設在使用者忘記密碼或想要快速更新密碼，而無需連接至受管節點的 RDP 或 SSH 連線時很有用。

先決條件

在能夠在受管節點上重設密碼前，必須符合下列需求：

- 您想要在其中變更密碼的受管節點必須是 Systems Manager 受管節點。還有，必須在受管節點上安裝 SSM Agent 2.3.668.0 版或更新版本。如需安裝或更新 SSM Agent 的詳細資訊，請參閱 使用 SSM Agent。
- 密碼重設功能會使用 Session Manager 組態，此組態已設定為讓您的帳戶連接到受管節點。因此，必須為目前 AWS 區域中您的帳戶完成 Session Manager 的使用事前準備。如需詳細資訊，請參閱設定 Session Manager。

  > **Note**
  >
  > 針對內部部署節點所提供的 Session Manager 支援僅適用於 advanced-instances 方案。如需詳細資訊，請參閱開啟 advanced-instances 方案。

- 變更密碼 AWS 的使用者必須擁有受管節點的 `ssm:SendCommand` 許可。如需詳細資訊，請參閱根據標籤限制 Run Command 存取。

限制存取

您可以限制使用者將密碼重設為特定受管節點的能力。您可使用 Session Manager `ssm:StartSession` 操作的以身分為基礎的政策搭配 `AWS-PasswordReset` SSM 文件來這麼做。如需詳細資訊，請參閱控制使用者工作階段存取執行個體。

加密資料

開啟 Session Manager AWS Key Management Service (AWS KMS) 資料的完整加密，以使用受管節點的密碼重設選項。如需詳細資訊，請參閱開啟工作階段資料的 KMS 金鑰加密 (主控台)。

在受管節點上重設密碼

您可以使用 Systems Manager Fleet Manager主控台或 AWS Command Line Interface () 在 Systems Manager 受管節點上重設密碼AWS CLI。

在受管節點上變更密碼 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇需要新密碼之節點旁的按鈕。
4. 依次選擇執行個體動作、重設密碼。
5. 對於 User name (使用者名稱)，輸入您要變更密碼的使用者名稱。這可以是在節點上擁有帳戶的任何使用者名稱。
6. 選擇提交。
7. 遵循輸入新密碼命令視窗中的指示，來指定新的密碼。

> **Note**
>
> 如果受管節點SSM Agent上的 版本不支援重設密碼，系統會提示您使用 Run Command中的工具安裝支援的版本 AWS Systems Manager。

若要在受管節點上重設密碼 (AWS CLI)

1. 若要在受管節點上為使用者重設密碼，請執行下列命令。將每個########取代為您自己的資訊。

> **Note**
>
> 若要使用 AWS CLI 重設密碼，外掛程式Session Manager必須安裝在本機機器上。如需相關資訊，請參閱 安裝 的Session Manager外掛程式 AWS CLI。

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. 遵循輸入新密碼命令視窗中的指示，來指定新的密碼。

對受管節點上的密碼重設進行疑難排解

您可透過確保已完成密碼重設事前準備來解決許多密碼重設的問題。對於其他問題，使用以下資訊以協助您對密碼重設的問題進行疑難排解。

主題

- 受管節點無法使用
- SSM Agent 不是最新狀態 (主控台)
- 不提供密碼重設選項 (AWS CLI)
- 沒有 ssm:SendCommand 的執行授權
- Session Manager 錯誤訊息

受管節點無法使用

問題：您想要在 Managed instances (受管執行個體) 主控台頁面上，為受管節點重設密碼，但該節點不在清單中。

- 解決方案：您想要連線的受管節點可能尚未設定用於 Systems Manager。若要將 EC2 執行個體與 Systems Manager 搭配使用，必須將授予 Systems Manager 許可在執行個體上執行動作的 AWS Identity and Access Management (IAM) 執行個體設定檔連接至執行個體。如需相關資訊，請參閱設定 Systems Manager 所需的執行個體許可。

  若要將非 EC2 機器與 Systems Manager 搭配使用，則須建立 IAM 服務角色，為 Systems Manager 提供在您機器上執行動作的許可。如需相關資訊，請參閱在混合多雲端環境中建立 Systems Manager 所需的 IAM 服務角色 (針對內部部署伺服器和虛擬機器所提供的 Session Manager 支援僅適用於進階執行個體方案。如需相關資訊，請參閱 開啟 advanced-instances 方案)。

SSM Agent 不是最新狀態 (主控台)

問題：一則訊息報告 SSM Agent 的版本不支援密碼重設功能。

- 解決方案：需要 SSM Agent 2.3.668.0 版或更新版本來執行密碼重設。在主控台中選擇 Update SSM Agent (更新)，即可更新受管節點上的代理程式。

  每當將新工具新增至 Systems Manager 或對現有工具進行更新時，SSM Agent就會發行更新的 版本。未使用最新版本的代理程式可能會使您的受管節點無法使用各種 Systems Manager 工具和功能。因此，我們建議您讓機器的 SSM Agent 自動保持最新狀態。如需相關資訊，請參閱 自動化 SSM Agent 更新。請訂閱 GitHub 上的 SSM Agent 版本備註頁面，以便接收有關 SSM Agent 更新的通知。

不提供密碼重設選項 (AWS CLI)

問題：您使用 命令成功連線到受管節點 AWS CLI `start-session`。您已指定 SSM 文件 `AWS-PasswordReset` 並提供有效的使用者名稱，但變更密碼的提示沒有顯示。

- 解決方案：受管節點上的 SSM Agent 版本不是最新的。需要 2.3.668.0 版或更新版本來執行密碼重設。

  每當將新工具新增至 Systems Manager 或對現有工具進行更新時，SSM Agent就會發行更新的 版本。未使用最新版本的代理程式可能會使您的受管節點無法使用各種 Systems Manager 工具和功能。因此，我們建議您讓機器的 SSM Agent 自動保持最新狀態。如需相關資訊，請參閱 自動化 SSM Agent 更新。請訂閱 GitHub 上的 SSM Agent 版本備註頁面，以便接收有關 SSM Agent 更新的通知。

沒有 **`ssm:SendCommand`** 的執行授權

問題：您嘗試連線至受管節點來變更密碼，但收到錯誤訊息，告知您沒有在受管節點上執行 `ssm:SendCommand` 的授權。

- 解決方案：您的 IAM 政策必須包含 `ssm:SendCommand` 命令的執行許可。如需相關資訊，請參閱 根據標籤限制 Run Command 存取。

Session Manager 錯誤訊息

問題：您收到與 Session Manager 相關的錯誤訊息。

- 解決方案：密碼重設支援要求 Session Manager 的設定需正確無誤。如需詳細資訊，請參閱 設定 Session Manager 及 Session Manager 疑難排解。

取消註冊混合多雲端環境中的受管節點

如果您不想再使用 管理內部部署伺服器、邊緣裝置或虛擬機器 (VM) AWS Systems Manager，則可以取消註冊。取消註冊混合啟用節點會將其從 Systems Manager 中的受管節點清單中移除。 AWS Systems Manager 在啟用混合模式的節點上執行的代理程式 (SSM Agent) 將無法重新整理其授權字符，因為已取消註冊。SSM Agent 會進入休眠，並將其對雲端中 Systems Manager 的 Ping 頻率降低為每小時一次。Systems Manager 會存放已取消註冊之受管節點的命令歷史記錄 30 天。

如果您想要重新註冊現場部署伺服器、邊緣裝置或 VM，您必須使用與先前註冊機器時不同的啟用碼和啟用 ID。啟用碼和啟用 ID 不可用於建立時指定的啟用數量上限。

以下程序說明如何使用 Systems Manager 主控台來取消註冊啟用混合模式的節點。如需如何使用 AWS Command Line Interface 來執行這項操作的資訊，請參閱 deregister-managed-instance。

如需相關資訊，請參閱下列主題：

- 取消註冊並重新註冊受管節點 (Linux) (Linux)
- 取消註冊並重新註冊受管節點 (Windows Server) (Windows Server)

取消註冊啟用混合模式的節點 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選取想要取消註冊之受管節點旁邊的核取方塊。
4. 依次選擇節點動作、工具、取消註冊此受管節點。
5. 檢閱取消註冊此受管節點對話方塊中的資訊。如果您核准，請選擇取消註冊。

透過 Fleet Manager 使用作業系統檔案系統

您可以使用 中的Fleet Manager工具 AWS Systems Manager，在受管節點上使用 檔案系統。使用 Fleet Manager，您可以檢視存放在連接至受管節點之磁碟區上的目錄和檔案資料相關資訊。例如，您可以檢視目錄和檔案的名稱、大小、副檔名、擁有者和許可。從 Fleet Manager 主控台，最多可以預覽 10,000 行文字形式的檔案資料。您也可以將此功能用於 `tail` 檔案。當您使用 `tail` 來檢視檔案資料

時，最初會顯示檔案的最後 10 行。隨著新的資料行寫入檔案中，即會即時更新視圖。因此，您可以從主控台檢閱日誌資料，這樣可以改善故障診斷和系統管理的效率。此外，您可以建立目錄，並複製、剪下、貼上、重新命名或刪除檔案和目錄。

建議建立定期備份，或建立連接至受管節點之 Amazon Elastic Block Store (Amazon EBS) 磁碟區的快照。複製或剪下和貼上檔案時，會取代目標路徑中與新檔案或目錄名稱相同的現有檔案和目錄。如果您取代或修改系統檔案和目錄，可能會發生嚴重問題。 AWS 不保證可以解決這些問題。修改系統檔案的風險由您自行承擔。您必須負責所有檔案和目錄的變更，並確保您有備份。刪除或取代檔案和目錄無法復原。

> **Note**
>
> Fleet Manager 使用 中的Session Manager工具 AWS Systems Manager來檢視文字預覽和`tail`檔案。對於 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體，連接至受管執行個體的執行個體設定檔必須向 Session Manager 提供許可，以使用此功能。如需將 Session Manager 許可新增至執行個體設定檔的詳細資訊，請參閱 將 Session Manager 許可新增至現有 IAM 角色。

主題

- 使用 Fleet Manager 檢視作業系統檔案系統
- 使用 Fleet Manager 預覽作業系統檔案
- 使用 Fleet Manager 追踪作業系統檔案
- 使用 Fleet Manager 複製、剪下和貼上作業系統檔案或目錄
- 使用 Fleet Manager 重新命名作業系統檔案和目錄
- 使用 Fleet Manager 刪除作業系統檔案和目錄
- 使用 Fleet Manager 建立作業系統目錄
- 使用 Fleet Manager 剪下、複製和貼上作業系統目錄

使用 Fleet Manager 檢視作業系統檔案系統

您可以使用 Fleet Manager 在 Systems Manager 受管節點上檢視作業系統檔案系統。

使用 Fleet Manager 檢視作業系統檔案系統

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。

2. 在導覽窗格中，選擇 Fleet Manager。
3. 選取您想要檢視之檔案系統的受管節點連結。
4. 依次選擇工具、檔案系統。

使用 Fleet Manager 預覽作業系統檔案

您可以使用 Fleet Manager 在作業系統上預覽文字檔案。

使用 Fleet Manager 檢視檔案的文字預覽

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選取您想要預覽之檔案的受管節點連結。
4. 依次選擇工具、檔案系統。
5. 選取資料夾的 File name (檔案名稱)，其中該目錄包含您要預覽的檔案。
6. 選擇您要預覽其內容之檔案旁的按鈕。
7. 依次選擇動作、以文字預覽。

使用 Fleet Manager 追踪作業系統檔案

您可以使用 Fleet Manager 在受管節點上追蹤檔案。

使用 Fleet Manager 追蹤作業系統檔案

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選取您想要追蹤之檔案的受管節點連結。
4. 依次選擇工具、檔案系統。
5. 選取目錄的 File name (檔案名稱)，該目錄包含您想要追蹤的檔案。
6. 選擇您要結尾其內容之檔案旁的按鈕。
7. 依次選擇動作、結尾檔案。

使用 Fleet Manager 複製、剪下和貼上作業系統檔案或目錄

可以使用 Fleet Manager 在受管節點上複製、剪下和貼上作業系統檔案。

使用 Fleet Manager 複製或剪下並貼上檔案或目錄

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選取具有您想要複製、或剪下並貼上之檔案的受管節點連結。
4. 依次選擇工具、檔案系統。
5. 若要複製或剪下檔案，請選取目錄的 File name (檔案名稱)，該目錄包含您想要複製或剪下的檔案。若要複製或剪下目錄，請選擇您想要複製或剪下之目錄旁的按鈕，然後繼續步驟 8。
6. 選擇您想要複製或剪下之檔案旁的按鈕。
7. 在動作選單上，選擇複製或剪下。
8. 在 File system (檔案系統) 檢視中，選擇您想要貼入檔案之目錄旁的按鈕。
9. 在動作選單上，選擇貼上。

使用 Fleet Manager 重新命名作業系統檔案和目錄

您可以使用 Fleet Manager 在帳戶中的受管節點上重新命名檔案和目錄。

使用 Fleet Manager 重新命名檔案或目錄

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選取具有您想要重新命名之檔案或目錄的受管節點連結。
4. 依次選擇工具、檔案系統。
5. 若要重新命名檔案，請選取目錄的 File name (檔案名稱)，該目錄包含您想要重新命名的檔案。若要重新命名目錄，請選擇您想要重新命名之目錄旁的按鈕，然後繼續步驟 8。
6. 選擇您要重新命名其內容之檔案旁的按鈕。
7. 依次選擇動作、重新命名。
8. 對於檔案名稱，輸入檔案的新名稱，然後選取重新命名。

使用 Fleet Manager 刪除作業系統檔案和目錄

可以使用 Fleet Manager 在帳戶中的受管節點上刪除檔案和目錄。

使用 Fleet Manager 刪除檔案或目錄

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選取具有您想要刪除之檔案或目錄的受管節點連結。
4. 依次選擇工具、檔案系統。
5. 若要刪除檔案，請選取目錄的 File name (檔案名稱)，該目錄包含您想要刪除的檔案。若要刪除目錄，請選擇您想要刪除之目錄旁的按鈕，然後繼續步驟 7。
6. 選擇具有您想要刪除之內容的檔案旁的按鈕。
7. 依次選擇動作、刪除。

使用 Fleet Manager 建立作業系統目錄

可以使用 Fleet Manager 在帳戶中的受管節點上建立目錄。

使用 Fleet Manager 建立目錄

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選取您想要在其中建立目錄之受管節點的連結。
4. 依次選擇工具、檔案系統。
5. 選取您想要在其中建立新目錄之目錄的 File name (檔案名稱)。
6. 選取 Create directory (建立目錄)。
7. 對於目錄名稱，輸入新目錄的名稱，然後選取建立目錄。

使用 Fleet Manager 剪下、複製和貼上作業系統目錄

可以使用 Fleet Manager 在帳戶中的受管節點上剪下、複製和貼上目錄。

使用 Fleet Manager 複製或剪下並貼上目錄

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選取具有您想要複製、或剪下並貼上之檔案的受管節點連結。
4. 依次選擇工具、檔案系統。
5. 選擇要複製或剪下之目錄旁的按鈕，然後繼續步驟 8。
6. 在動作選單上，選擇複製或剪下。
7. 在 File system (檔案系統) 檢視中，選擇您想要貼入檔案之目錄旁的按鈕。
8. 在動作選單上，選擇貼上。

監控受管節點效能

您可以使用 中的Fleet Manager工具 AWS Systems Manager，即時檢視受管節點的效能資料。可從效能計數器擷取效能資料。

以下效能計數器在 Fleet Manager 中可用。

- CPU 使用率
- 磁碟輸入/輸出 (I/O) 使用率
- 網路流量
- 記憶體用量

> **Note**
>
> Fleet Manager 使用 中的Session Manager工具 AWS Systems Manager來擷取效能資料。對於 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體，連接至受管執行個體的執行個體設定檔必須向 Session Manager 提供許可，以使用此功能。如需將 Session Manager 許可新增至執行個體設定檔的詳細資訊，請參閱 將 Session Manager 許可新增至現有 IAM 角色。

若要使用 Fleet Manager 檢視效能資料

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。

2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇您要監控其效能之受管節點旁的按鈕。
4. 請選擇檢視詳細資料。
5. 依次選擇工具、效能計數器。

處理程序

您可以使用 中的Fleet Manager工具 AWS Systems Manager來處理受管節點上的程序。使用 Fleet Manager，您可以檢視程序的相關資訊。例如，除了程序的控制代碼和執行緒之外，您還可以查看程序的 CPU 使用率和記憶體使用情況。使用 Fleet Manager，您可以從主控台啟動和終止程序。

> **Note**
>
> Fleet Manager 使用 中的Session Manager工具 AWS Systems Manager來擷取程序資料。對於 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體，連接至受管執行個體的執行個體設定檔必須向 Session Manager 提供許可，以使用此功能。如需將 Session Manager 許可新增至執行個體設定檔的詳細資訊，請參閱 將 Session Manager 許可新增至現有 IAM 角色。

主題

- 使用 Fleet Manager 檢視有關作業系統程序的詳細資訊
- 使用 Fleet Manager 在受管節點上啟動作業系統程序
- 使用 Fleet Manager 終止作業系統程序

使用 Fleet Manager 檢視有關作業系統程序的詳細資訊

您可以使用 Fleet Manager 檢視有關受管節點上程序的詳細資訊。

檢視具有 Fleet Manager 之程序的詳細資訊

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選取要檢視其程序之節點的連結。
4. 依次選擇工具、程序。

使用 Fleet Manager 在受管節點上啟動作業系統程序

您可以使用 Fleet Manager 在受管節點上啟動程序。

使用 Fleet Manager 啟動程序

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選取要在其中啟動程序的受管節點的連結。
4. 依次選擇工具、程序。
5. 選取 Start new process (啟動新程序)。
6. 對於程序名稱或完整路徑，輸入程序的名稱或執行檔的完整路徑。
7. (選用) 對於工作目錄，輸入您要執行程序的目錄路徑。

使用 Fleet Manager 終止作業系統程序

使用 Fleet Manager 終止作業系統程序

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選取要在其中啟動程序的受管節點的連結。
4. 依次選擇工具、程序。
5. 選擇您想要終止之程序旁的按鈕。
6. 依次選擇動作、終止程序或動作、終止程序樹。

> **Note**
>
> 終止程序樹狀結構也會終止使用該程序的所有程序和應用程式。

檢視受管節點上的日誌

您可以使用 中的Fleet Manager工具 AWS Systems Manager來檢視存放在受管節點上的日誌資料。如果是 Windows 受管節點，您可以從主控台檢視 Windows 事件日誌，並複製其詳細資訊。若要協助您搜尋事件，請依 Event level (事件層級)、Event ID (事件 ID)、Event source (事件來源) 以及 Time

created (建立時間) 篩選 Windows 事件日誌。您也可以使用可檢視檔案系統的程序來檢視其他日誌資料。如需使用 Fleet Manager 檢視檔案系統的詳細資訊，請參閱 透過 Fleet Manager 使用作業系統檔案系統。

若要使用 Fleet Manager 檢視 Windows 事件日誌

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇您要檢視其事件日誌檔案之受管節點旁的按鈕。
4. 請選擇檢視詳細資料。
5. 依次選擇工具、Windows 事件日誌。
6. 選擇包含您要檢視的事件的 Log name (日誌名稱)。
7. 選擇您要檢視的 Log name (日誌名稱) 旁的按鈕，然後選擇 View events (檢視事件)。
8. 選擇您要檢視的事件旁的按鈕，然後選擇 View event details (檢視事件詳細資訊)。
9. (選用) 選取 Copy as JSON (複製為 JSON) 將事件詳細資訊複製到剪貼簿。

使用 Fleet Manager 管理受管節點上的作業系統使用者帳戶和群組

您可以使用 中的Fleet Manager工具 AWS Systems Manager來管理受管節點上的作業系統 (OS) 使用者帳戶和群組。例如，您可以建立和刪除使用者和群組。此外，您可以檢視群組成員資格、使用者角色和狀態等詳細資訊。

> **Important**
>
> Fleet Manager 使用 Run Command和 Session Manager中的工具 AWS Systems Manager，進行各種使用者管理操作。因此，使用者可以將許可授予作業系統使用者帳戶，否則他們將無法執行。這是因為 AWS Systems Manager Agent (SSM Agent) 使用根許可 (Linux) 或 SYSTEM 許可 (Windows Server) 在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上執行。如需透過 SSM Agent 限制存取根層級命令的詳細資訊，請參閱 限制透過 SSM Agent 存取根層級命令。若要限制對此功能的存取，建議您為使用者建立 AWS Identity and Access Management (IAM) 政策，僅允許存取您定義的動作。如需建立 Fleet Manager 的 IAM 政策的詳細資訊，請參閱 控制 Fleet Manager 存取。

主題

- 使用 Fleet Manager 建立作業系統使用者或群組
- 使用 Fleet Manager 更新使用者或群組成員資格
- 使用 Fleet Manager 刪除作業系統使用者或群組

使用 Fleet Manager 建立作業系統使用者或群組

> **Note**
>
> Fleet Manager 會使用 Session Manager 來為新使用者設定密碼。對於 Amazon EC2 執行個體，連接至受管理執行個體的執行個體設定檔必須向 Session Manager 提供許可，以使用此功能。如需將 Session Manager 許可新增至執行個體設定檔的詳細資訊，請參閱 將 Session Manager 許可新增至現有 IAM 角色。

您可以使用 Fleet Manager 主控台執行相同的任務，而不是直接登入伺服器來建立使用者帳戶或群組。

使用 Fleet Manager 建立作業系統使用者帳戶

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇您要在其上建立新使用者之受管節點旁的按鈕。
4. 請選擇檢視詳細資料。
5. 依次選擇工具、使用者和群組。
6. 選擇 Users (使用者) 標籤，然後選擇 Create Users (建立使用者)。
7. 為新使用者的 Name (名稱) 輸入數值。
8. (建議) 選取 Set password (設定密碼) 旁的核取方塊。在程序結束時，系統會提示您為新使用者提供密碼。
9. 選取 Create user (建立使用者)。如果您選取核取方塊來為新使用者建立密碼，則系統會提示您輸入密碼值，然後選取 Done (完成)。如果您指定的密碼不符合受管節點本機或網域政策指定的需求，則會傳回錯誤。

使用 Fleet Manager 建立作業系統群組

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇您要在其中建立群組之受管節點旁的按鈕。
4. 請選擇檢視詳細資料。
5. 依次選擇工具、使用者和群組。
6. 選擇 Groups (群組) 標籤，然後選擇 Create group (建立群組)。
7. 為新群組的 Name (名稱) 輸入數值。
8. (選用) 為新群組的 Description (描述) 輸入數值。
9. (選用) 選取要新增至新群組的 Group members (群組成員) 的使用者。
10. 選取 Create group (建立群組)。

使用 Fleet Manager 更新使用者或群組成員資格

您可以使用 Fleet Manager 主控台執行相同的任務，而不是直接登入伺服器來更新使用者帳戶或群組。

使用 Fleet Manager 將作業系統使用者帳戶新增至新群組

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇您要更新的使用者帳戶所在受管節點旁的按鈕。
4. 請選擇檢視詳細資料。
5. 依次選擇工具、使用者和群組。
6. 選擇使用者索引標籤。
7. 選擇您要更新的使用者旁的按鈕。
8. 依次選擇動作、將使用者新增至群組。
9. 在 Add to group (新增至群組) 下，選擇您要新增使用者的群組。
10. 選取 Add user to group (新增使用者至群組)。

使用 Fleet Manager 編輯作業系統群組的成員資格

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇您要更新的群組所在受管節點旁的按鈕。
4. 請選擇檢視詳細資料。
5. 依次選擇工具、使用者和群組。
6. 選擇 Groups (群組) 標籤。
7. 選擇您要更新的群組旁的按鈕。
8. 依次選擇動作、修改群組。
9. 在 Group members (群組成員) 下，選擇您要新增或移除的使用者。
10. 選取 Modify group (修改群組)。

使用 Fleet Manager 刪除作業系統使用者或群組

您可以使用 Fleet Manager 主控台執行相同的任務，而不是直接登入伺服器來刪除使用者帳戶或群組。

使用 Fleet Manager 刪除作業系統使用者帳戶

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇您要刪除的使用者帳戶所在受管節點旁的按鈕。
4. 請選擇檢視詳細資料。
5. 選擇使用者和群組。
6. 選擇使用者索引標籤。
7. 選擇您要刪除的使用者旁的按鈕。
8. 依次選擇動作、刪除本機使用者。

使用 Fleet Manager 刪除作業系統群組

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。

2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇您要更新的群組所在受管節點旁的按鈕。
4. 請選擇檢視詳細資料。
5. 依次選擇工具、使用者和群組。
6. 選擇 Group (群組) 標籤。
7. 選擇您要更新的群組旁的按鈕。
8. 依次選擇動作、刪除本機群組。

管理受管節點上的 Windows 登錄檔

您可以使用 中的Fleet Manager工具 AWS Systems Manager來管理受Windows Server管節點上的登錄檔。從 Fleet Manager 主控台，您可以建立、複製、更新和刪除登錄項目與數值。

> **Important**
>
> 建議您在修改登錄之前，建立登錄備份，或建立連接至受管節點的根 Amazon Elastic Block Store (Amazon EBS) 磁碟區的快照。如果您不正確地修改登錄，可能會發生嚴重的問題。這些問題可能需要您重新安裝作業系統，或從快照還原節點的根磁碟區。 AWS 不保證可以解決這些問題。請自行承擔修改登錄的風險。您必須負責所有登錄變更，並確保您有備份。

建立 Windows 登錄金鑰或項目

若要使用 Fleet Manager 建立 Windows 登錄金鑰

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇您要在其上建立登錄金鑰的受管節點旁的按鈕。
4. 請選擇檢視詳細資料。
5. 依次選擇工具、Windows 登錄檔。
6. 選取 Registry name (登錄名稱)，選擇您要在其中建立新登錄金鑰的 Hive。
7. 依次選擇建立、建立登錄機碼。
8. 選擇您要在其中建立新金鑰的登錄項目旁的按鈕。

9. 選擇 Create registry key (建立登錄金鑰)。
10. 為新登錄金鑰的 Name (名稱) 輸入數值，然後選取 Submit (提交)。

若要使用 Fleet Manager 建立 Windows 登錄項目

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇您要在其上建立登錄項目的執行個體旁的按鈕。
4. 請選擇檢視詳細資料。
5. 依次選擇工具、Windows 登錄檔。
6. 選取 Registry name (登錄名稱)，選擇您要在其中建立新登錄項目的 Hive 以及後續登錄金鑰。
7. 依次選擇建立、建立登錄項目。
8. 為新登錄項目的 Name (名稱) 輸入數值。
9. 選擇要為登錄項目建立的數值類型。如需登錄值類型的詳細資訊，請參閱登錄值類型。
10. 為新登錄項目的 Value (數值) 輸入數值。

更新 Windows 登錄項目

若要使用 Fleet Manager 更新 Windows 登錄項目

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇您要在其上更新登錄項目的受管節點旁的按鈕。
4. 請選擇檢視詳細資料。
5. 依次選擇工具、Windows 登錄檔。
6. 選取 Registry name (登錄名稱)，選擇您要更新的 Hive 以及後續登錄金鑰。
7. 選擇您要更新的登錄項目旁的按鈕。
8. 依次選擇動作、更新登錄項目。
9. 為登錄項目的 Value (數值) 輸入新值。
10. 選擇更新。

刪除 Windows 登錄項目或金鑰

若要使用 Fleet Manager 刪除 Windows 登錄金鑰

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇您要在其上刪除登錄金鑰的受管節點旁的按鈕。
4. 依次選擇工具、Windows 登錄檔。
5. 選取 Registry name (登錄名稱)，選擇您要刪除的 Hive 以及後續登錄金鑰。
6. 選擇您要刪除的登錄金鑰旁的按鈕。
7. 依次選擇動作、刪除登錄機碼。

若要使用 Fleet Manager 刪除 Windows 登錄項目

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇您要在其上刪除登錄項目的受管節點旁的按鈕。
4. 請選擇檢視詳細資料。
5. 依次選擇工具、Windows 登錄檔。
6. 選取 Registry name (登錄名稱)，選擇您要刪除的包含項目的 Hive 以及後續登錄金鑰。
7. 選擇您要刪除的登錄項目旁的按鈕。
8. 依次選擇動作、刪除登錄項目。

使用預設主機管理組態來自動管理 EC2 執行個體

預設主機管理組態設定允許 AWS Systems Manager 以受管執行個體的形式自動管理您的 Amazon EC2 執行個體。 受管執行個體是指一種設定為搭配 Systems Manager 使用的 EC2 執行個體。

使用 Systems Manager 管理執行個體的好處包含：

- 使用 Session Manager 安全地連線至 EC2 執行個體。
- 使用 Patch Manager 執行自動修補程式掃描。

- 使用 Systems Manager 庫存檢視執行個體的詳細資訊。
- 使用 Fleet Manager 追蹤和管理執行個體。
- 自動將 SSM Agent 保持在最新狀態。

Fleet Manager、庫存Patch Manager、 和 Session Manager是 Systems Manager 中的工具。

使用預設主機管理組態，您可以管理 EC2 執行個體，而無需手動建立 AWS Identity and Access Management (IAM) 執行個體描述檔。反之，預設主機管理組態會建立並套用預設 IAM 角色，以確保 Systems Manager 具有管理 AWS 帳戶 及其啟用 AWS 區域 位置中所有執行個體的許可。

如果提供的許可不足以滿足使用案例，您也可以將政策新增至預設主機管理組態建立的預設 IAM 角色。如果您不需要預設 IAM 角色提供的所有功能的許可，則可以建立自己的自訂角色和政策。對為預設主機管理組態選擇的 IAM 角色所做的任何變更都會套用到該區域和帳戶中的所有受管 Amazon EC2 執行個體。

如需有關預設主機管理組態所使用之政策的詳細資訊，請參閱 AWS 管理的政策：AmazonSSMManagedEC2InstanceDefaultPolicy。

實作最低權限存取

本主題中的程序僅供管理員執行。因此，建議您實作最低權限存取原則，以防止非管理使用者設定或修改預設主機管理組態。若要檢視限制存取預設主機管理組態的範例政策，請參閱本主題稍後的 預設主機管理組態的最低權限政策範例 一節。

> **Important**
>
> 使用預設主機管理組態註冊的執行個體，會將註冊資訊存放在本機 `var/lib/amazon/ssm` 或 `C:\ProgramData\Amazon` 目錄中。移除這些目錄或其檔案可防止執行個體取得使用預設主機管理組態連線至 Systems Manager 的必要憑證。在這些情況下，您必須使用 IAM 執行個體設定檔來提供執行個體所需的許可，或重新建立執行個體。

主題

- 先決條件
- 啟用預設主機管理組態設定
- 停用預設主機管理組態設定
- 預設主機管理組態的最低權限政策範例

先決條件

若要在 AWS 區域 和您啟用設定 AWS 帳戶 的位置使用預設主機管理組態，必須符合下列要求。

- 要管理的執行個體必須使用 Instance Metadata Service Version 2 (IMDSv2)。

  預設主機管理組態不支援執行個體中繼資料服務第 1 版。如需有關轉換至 IMDSv2 的資訊，請參閱《Amazon EC2 使用者指南》中的 Transition to using Instance Metadata Service Version 2
- 要管理的執行個體上必須安裝 SSM Agent 3.2.582.0 版或更新版本。

  如需有關檢查執行個體上已安裝之 SSM Agent 版本的詳細資訊，請參閱 檢查 SSM Agent 版本編號。

  如需有關更新 SSM Agent 的資訊，請參閱 自動更新 SSM Agent。
- 作為管理員，在執行本主題所述任務時，您必須擁有執行 GetServiceSetting、ResetServiceSetting 和 UpdateServiceSetting API 操作的許可。此外，您必須擁有 `AWSSystemsManagerDefaultEC2InstanceManagementRole` IAM 角色 `iam:PassRole` 許可的許可。以下是提供這些許可的範例政策。將每個*########*取代為您自己的資訊。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetServiceSetting",
                "ssm:ResetServiceSetting",
                "ssm:UpdateServiceSetting"
            ],
            "Resource": "arn:aws:ssm:region:account-id:servicesetting/ssm/managed-
instance/default-ec2-instance-management-role"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::account-id:role/service-role/
AWSSystemsManagerDefaultEC2InstanceManagementRole",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
```

```
                    "ssm.amazonaws.com"
                ]
            }
        }
    ]
}
```

- 如果 IAM 執行個體設定檔已連接至要使用 Systems Manager 管理的 EC2 執行個體，則必須從中移除允許 `ssm:UpdateInstanceInformation` 操作的任何許可。SSM Agent 在使用預設主機管理組態許可之前，會嘗試使用執行個體設定檔許可。如果您允許自己的 IAM 執行個體設定檔中的 `ssm:UpdateInstanceInformation` 操作，執行個體將不會使用預設主機管理組態許可。

啟用預設主機管理組態設定

您可以從 Fleet Manager主控台或使用 AWS Command Line Interface 或 來啟用預設主機管理組態 AWS Tools for Windows PowerShell。

您必須在希望透過此設定管理 Amazon EC2 執行個體的每個區域中逐一開啟預設主機管理組態。

開啟預設主機管理組態後，執行個體可能需要 30 分鐘的時間才能使用您在下列程序的步驟 5 中所選角色的憑證。

啟用預設主機管理組態 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 依次選擇帳戶管理、設定預設主機管理組態。
4. 開啟開啟預設主機管理組態。
5. 選擇用於為您的執行個體啟用 Systems Manager 工具的 AWS Identity and Access Management (IAM) 角色。建議使用預設主機管理組態提供的預設角色。其中包括使用 Systems Manager 管理 Amazon EC2 執行個體所需的最小許可集。如果您偏好使用自訂角色，角色的信任政策必須允許 Systems Manager 作為受信任實體。
6. 選擇設定以完成設定。

啟用預設主機管理組態 (命令列)

1. 在本機電腦建立 JSON 檔案，內含以下信任關係政策。

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

2. 開啟 AWS CLI 或 Tools for Windows PowerShell，並根據本機電腦的作業系統類型執行下列其中一個命令，在您的帳戶中建立服務角色。將每個*########*取代為您自己的資訊。

   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. 執行以下命令以將 AmazonSSMManagedEC2InstanceDefaultPolicy 受管政策連接到您新建立的角色。將每個*########*取代為您自己的資訊。

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. 開啟 AWS CLI 或 Tools for Windows PowerShell，並執行下列命令。將每個########取代為您自己的資訊。

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"
```

如果命令成功，則無輸出訊息。

5. 執行下列命令，以檢視目前 AWS 帳戶 和 中預設主機管理組態的目前服務設定 AWS 區域。

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"
```

該命令會傳回相關資訊，如以下所示。

```
{
    "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"
    }
}
```

停用預設主機管理組態設定

您可以從 Fleet Manager主控台或使用 AWS Command Line Interface 或 來停用預設主機管理組態 AWS Tools for Windows PowerShell。

您必須在不再需要 Amazon EC2 執行個體由此組態管理的每個區域中逐一關閉預設主機管理組態設定。在一個區域中停用並不會導致在所有區域中停用。

如果停用預設主機管理組態，且尚未將執行個體設定檔連接到允許存取 Systems Manager 的 Amazon EC2 執行個體，則 Systems Manager 將不再管理這些執行個體。

停用預設主機管理組態 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 依次選擇帳戶管理、預設主機管理組態。
4. 關閉啟用預設主機管理組態。
5. 選擇設定以停用預設主機管理組態。

停用預設主機管理組態 (命令列)

- 開啟 AWS CLI 或 Tools for Windows PowerShell，並執行下列命令。將每個########取代為您自己的資訊。

  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"
```

預設主機管理組態的最低權限政策範例

下列範例政策示範如何防止您的組織成員變更您帳戶中的[預設主機管理組態設定。

的服務控制政策 AWS Organizations

下列政策示範如何防止 中的非管理成員 AWS Organizations 更新您的預設主機管理組態設定。將每個########取代為您自己的資訊。

```
{
    "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"
                    ]
                }
            }
        }
    ]
}
```

IAM 主體的政策

下列政策示範如何防止 中的 IAM 群組、角色或使用者 AWS Organizations 更新您的預設主機管理組態設定。將每個########取代為您自己的資訊。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "ssm:UpdateServiceSetting",
                "ssm:ResetServiceSetting"
            ],
```

```
            "Resource": "arn:aws:ssm:region:account-id:servicesetting/ssm/managed-
instance/default-ec2-instance-management-role"
        },
        {
            "Effect": "Deny",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:DeleteRole",
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::account-id:role/service-role/
AWSSystemsManagerDefaultEC2InstanceManagementRole"
        }
    ]
}
```

使用 Remote Desktop 連線至 Windows Server 受管執行個體

您可以使用 中的Fleet Manager工具 AWS Systems Manager，使用 Windows Server (RDP) 連線至 Amazon Elastic Compute Cloud Remote Desktop Protocol(Amazon EC2) 執行個體。 Fleet Manager 由 Amazon DCV 提供支援的遠端桌面，可讓您直接從 Systems Manager 主控台安全連線至 Windows Server 執行個體。在單一瀏覽器視窗中，您最多可同時建立四條連線。

目前您只能在執行 Windows Server 2012 RTM 或更新版本的執行個體上使用遠端桌面。遠端桌面僅支援英文語言輸入。

Fleet Manager 遠端桌面是僅限主控台的服務，不支援與受管執行個體的命令列連線。若要透過 shell 連線至Windows Server受管執行個體，您可以使用 Session Manager中的另一個工具 AWS Systems Manager。如需詳細資訊，請參閱AWS Systems Manager Session Manager。

> **Note**
>
> RDP 連線的持續時間並非由 AWS Identity and Access Management (IAM) 登入資料持續時間決定。相反地，連線會持續存在，直到達到連線持續時間上限或閒置時間限制為止，以先到者為準。如需詳細資訊，請參閱遠端連線持續時間與並行。

如需設定 AWS Identity and Access Management (IAM) 許可以允許執行個體與 Systems Manager 互動的詳細資訊，請參閱設定 Systems Manager 的執行個體許可。

主題

- 設定您的環境
- 為遠端桌面設定 IAM 許可
- 驗證遠端桌面連線
- 遠端連線持續時間與並行
- 使用遠端桌面連線至受管節點
- 檢視有關目前和已完成連線的資訊

設定您的環境

在您使用遠端桌面之前，請確定您的環境符合下列要求：

- 受管節點組態

  確保您的 Amazon EC2 執行個體在 Systems Manager 中設定為受管節點。
- SSM Agent 最低版本

  確認節點執行的是 SSM Agent 3.0.222.0 或更新版本。如需有關如何檢查節點執行之代理程式版本的詳細資訊，請參閱 檢查 SSM Agent 版本編號。如需安裝或更新 SSM Agent 的詳細資訊，請參閱使用 SSM Agent。
- RDP 連接埠組態

  若要接受遠端連線，Windows Server 節點上的 Remote Desktop Services 服務必須使用預設的 RDP 連接埠 3389。這是 (Amazon Machine ImagesAMIs) 提供的預設組態 AWS。您不需要明確開啟任何傳入連接埠即可使用遠端桌面。
- 鍵盤功能所需的 PSReadLine 模組版本

  若要確保您的鍵盤在 PowerShell 中正常運作，請確認執行 Windows Server 2022 的節點已安裝 PSReadLine 模組 2.2.2 或更新版本。如果執行的是較舊的版本，您可以使用以下命令安裝所需的版本。

  ```
  Install-PackageProvider -Name NuGet -MinimumVersion 2.8.5.201 -Force
  ```

  安裝 NuGet 套件供應程式之後，請執行下列命令。

  ```
  Install-Module `
   -Name PSReadLine `
   -Repository PSGallery `
  ```

```
-MinimumVersion 2.2.2 -Force
```

- Session Manager 組態

  您必須先完成 Session Manager 設定的先決條件，才能使用遠端桌面。當您使用遠端桌面連線至執行個體時， AWS 區域 會套用為 AWS 帳戶 和 定義的任何工作階段偏好設定。如需詳細資訊，請參閱設定 Session Manager。

  > **Note**
  >
  > 如果使用 Amazon Simple Storage Service (Amazon S3) 記錄 Session Manager 的活動，則遠端桌面連線會在 `bucket_name/Port/stderr` 中產生以下錯誤。此錯誤是預期會出現的行為，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>
  > ```

為遠端桌面設定 IAM 許可

除了 Systems Manager 和 Session Manager 所需的 IAM 許可之外，您使用的使用者或角色還必須擁有啟動連線的許可。

啟動連線的許可

若要在主控台中建立與 EC2 執行個體的 RDP 連線，則需要下列許可：

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

列出連線的許可

若要在主控台中檢視連線清單，則需要下列許可：

`ssm-guiconnect:ListConnections`

以下是您可以連接至使用者或角色的 IAM 政策範例，這些政策可允許您與遠端桌面進行不同類型的互動。將每個########取代為您自己的資訊。

連線至 EC2 執行個體的標準政策

```
{
    "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:*:account-id:instance/*",
                "arn:aws:ssm:*:account-id: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": "*"
        }
    ]
}
```

用於將 EC2 執行個體與特定標籤連線的政策

> **Note**
>
> 在下列 IAM 政策中，SSMStartSession 區段需要用於 ssm:StartSession 動作的 Amazon Resource Name (ARN)。如所示，您指定的 ARN 不需要 AWS 帳戶 ID。如果您指定帳戶 ID，Fleet Manager 會傳回 AccessDeniedException。AccessTaggedInstances 區段位於範例政策的下方，也需要用於 ssm:StartSession 的 ARN。對於這些 ARN，您需要指定 AWS 帳戶 ID。

```
{
    "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:*:account-id:instance/*",
                "arn:aws:ssm:*:account-id: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": "*"
        }
    ]
}
```

AWS IAM Identity Center 使用者連線至 EC2 執行個體的政策

```
{
    "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": "*"
        }
    ]
}
```

驗證遠端桌面連線

建立遠端連線時，您可以使用 Windows 憑證或與執行個體相關聯的 Amazon EC2 金鑰對 (.pem 檔案) 進行驗證。如需有關使用金鑰對的資訊，請參閱《Amazon EC2 使用者指南》中的 Amazon EC2 key pairs and Windows instances。

或者，如果您 AWS Management Console 使用 進行身分驗證 AWS IAM Identity Center，則可以連接到執行個體，而無需提供額外的登入資料。如需允許使用 IAM Identity Center 進行遠端連線身份驗證的政策範例，請參閱 為遠端桌面設定 IAM 許可。

開始之前

開始使用遠端桌面連線之前，請留意下列使用 IAM Identity Center 驗證的條件。

- 遠端桌面支援在您啟用 IAM Identity Center 所在 AWS 區域 的節點的 IAM Identity Center 身分驗證。
- 遠端桌面支援最多 16 個字元的 IAM Identity Center 使用者名稱。
- 遠端桌面支援的 IAM Identity Center 使用者名稱可包含英數字元和下列特殊字元：.、- 和 _

> **Important**
>
> 使用包含下列字元的 IAM Identity Center 使用者名稱無法成功連線：+ = ,
> IAM Identity Center 的使用者名稱支援這些字元，但 Fleet Manager RDP 連線則不支援。此外，如果 IAM Identity Center 使用者名稱包含一個或多個 @ 符號，Fleet Manager 會忽略第一個 @ 符號及其後面的所有字元，無論 @ 是否引入電子郵件地址的網域部分。例如，對於 IAM Identity Center 使用者名稱 `diego_ramirez@example.com`，`@example.com` 部分將被忽略，且 Fleet Manager 的使用者名稱會變成 `diego_ramirez`。對於 `diego_r@mirez@example.com`，Fleet Manager 會忽略 `@mirez@example.com`，且 Fleet Manager 的使用者名稱會變成 `diego_r`。

- 使用 IAM Identity Center 驗證連線時，遠端桌面會在執行個體的 Local Administrators 群組中建立一個本機 Windows 使用者。遠端連線結束後，此使用者仍將存在。
- 遠端桌面不允許屬於 Microsoft Active Directory 網域控制站的節點使用 IAM Identity Center 驗證。
- 雖然遠端桌面允許您針對已加入 Active Directory 網域的節點使用 IAM Identity Center 驗證，但我們不建議您這樣做。此驗證方法會將系統管理許可授予使用者，從而導致使用者可能會覆寫網域授予的更嚴格許可。

IAM Identity Center 驗證支援的區域

使用 IAM Identity Center 驗證的 Remote Desktop 連線在下列 AWS 區域中獲得支援：

- 美國東部 (俄亥俄) (us-east-2)
- 美國東部 (維吉尼亞北部) (us-east-1)
- 美國西部 (加利佛尼亞北部) (us-west-1)
- 美國西部 (奧勒岡) (us-west-2)
- 非洲 (開普敦) (af-south-1)
- 亞太區域 (香港) (ap-east-1)
- 亞太區域 (孟買) (ap-south-1)
- 亞太區域 (東京) (ap-northeast-1)

- 亞太區域 (首爾) (ap-northeast-2)
- 亞太區域 (大阪) (ap-northeast-3)
- 亞太區域 (新加坡) (ap-southeast-1)
- 亞太區域 (雪梨) (ap-southeast-2)
- 亞太區域 (雅加達) (ap-southeast-3)
- 加拿大 (中部) (ca-central-1)
- 歐洲 (法蘭克福) (eu-central-1)
- 歐洲 (斯德哥爾摩) (eu-north-1)
- 歐洲 (愛爾蘭) (eu-west-1)
- 歐洲 (倫敦) (eu-west-2)
- 歐洲 (巴黎) (eu-west-3)
- 以色列 (特拉維夫) (il-central-1)
- 南美洲 (聖保羅) (sa-east-1)
- 歐洲 (米蘭) (eu-south-1)
- 中東 (巴林) (me-south-1)
- AWS GovCloud （美國東部）(us-gov-east-1)
- AWS GovCloud （美國西部）(us-gov-west-1)

遠端連線持續時間與並行

下列條件適用於作用中的遠端桌面連線：

- 連線持續時間

  根據預設，遠端桌面連線會在 60 分鐘後中斷。若要避免中斷連線，您可以在中斷連線前選擇續約工作階段來重設持續時間計時器。

- 連線逾時

  遠端桌面連線會在閒置超過 10 分鐘後中斷。

- 連線持續性

  Windows Server 使用遠端桌面連線至 後，連線會持續存在，直到達到連線持續時間上限 (60 分鐘）或閒置逾時限制 (10 分鐘）。連線持續時間不是由 AWS Identity and Access Management (IAM) 登

入資料的持續時間決定。如果不符合連線持續時間限制，則 IAM 登入資料過期後，連線會持續存在。使用遠端桌面時，您應該在 IAM 登入資料過期後，離開瀏覽器頁面來終止連線。

- 並行連線

  根據預設，您一次最多可以有 5 個作用中的遠端桌面連線，用於相同的 AWS 帳戶 和 AWS 區域。若要請求將服務配額提高到最高 25 個並行連線，請參閱《Service Quotas 使用者指南》中的請求提高配額。

  > **Note**
  >
  > 的標準授權Windows Server允許兩個並行 RDP 連線。若要支援更多連線，您必須從 Microsoft 或 Microsoft 遠端桌面服務授權購買額外的用戶端存取授權 CALs) AWS。如需補充授權的詳細資訊，請參閱下列主題：
  >
  > - Microsoft 網站上的用戶端存取授權和管理授權
  > - 《License Manager 使用者指南》中的針對支援的軟體產品使用 License Manager 使用者型訂閱

使用遠端桌面連線至受管節點

瀏覽器複製/貼上文字支援

若使用 Google Chrome 和 Microsoft Edge 瀏覽器，可以將文字從受管節點複製並貼到本機電腦，以及從本機電腦複製並貼到連線的受管節點。

若使用 Mozilla Firefox 瀏覽器，您只能將文字從受管節點複製並貼到本機電腦。不支援從本機電腦複製到受管節點。

使用 Fleet Manager 遠端桌面連線至受管節點

1. 開啟 AWS Systems Manager 主控台，網址為 https://console.aws.amazon.com/systems-manager/：//。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇您要連線到的節點。您可以選取核取方塊或節點名稱。
4. 在節點動作選單中，選擇使用遠端桌面連線。
5. 選擇您偏好的 Authentication type (身分驗證類型)。如果您選擇使用者憑證，請在連線的節點上輸入 Windows 使用者帳戶的使用者名稱和密碼。如果選擇金鑰對，您可以使用以下其中一種方法提供身份驗證憑證：

a. 如果要從本機檔案系統選取與執行個體相關聯的 PEM 金鑰，請選擇瀏覽本機電腦。

- 或 -

b. 如果您要複製 PEM 檔案的內容並將其貼到提供的欄位中，請選擇貼上金鑰對內容。

6. 選取連線。
7. 若要選擇您偏好的顯示解析度，請在動作選單中選擇解析度，然後從下列選項中選取：

- 自動調整
- 1920 x 1080
- 1400 x 900
- 1366 x 768
- 800 x 600

自動調整選項會根據偵測到的螢幕大小自動設定解析度。

檢視有關目前和已完成連線的資訊

您可以使用 Systems Manager 主控台的 Fleet Manager 區段來檢視帳戶中已進行 RDP 連線的相關資訊。可以使用一組篩選條件，將顯示的連線清單限制為時間範圍、特定執行個體、建立連線的使用者，以及特定狀態的連線。主控台也提供索引標籤，顯示有關所有目前作用中連線和所有過去連線的資訊。

檢視有關目前和已完成連線的資訊

1. 開啟 AWS Systems Manager 主控台，網址為 https：//https://console.aws.amazon.com/systems-manager/。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇帳戶管理、與遠端桌面連線。
4. 選擇下列其中一個索引標籤：

- 作用中連線
- 連線歷史記錄

5. 若要進一步縮小顯示的連線結果清單範圍，請在搜尋
( )
方塊中指定一個或多個篩選條件。也可以輸入任意文字搜尋詞彙。

在受管執行個體上管理 Amazon EBS 磁碟區

Amazon Elastic Block Store (Amazon EBS) 提供區塊層級儲存磁碟區，可搭配使用 Amazon Elastic Compute Cloud (EC2) 執行個體。EBS 磁碟區的行為與未格式化的原始區塊型儲存設備相似。您可以將這些磁碟區做為裝置，掛載在您的執行個體上。

您可以使用 中的Fleet Manager工具 AWS Systems Manager來管理受管執行個體上的 Amazon EBS 磁碟區。例如，您可以初始化 EBS 磁碟區、格式化分割區，然後掛載磁碟區以供使用。

> **Note**
>
> Fleet Manager 目前僅支援 Windows Server 執行個體的 Amazon EBS 磁碟區管理。

檢視 EBS 磁碟區詳細資訊

使用 Fleet Manager 檢視 EBS 磁碟區的詳細資訊

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇您要檢視 EBS 磁碟區詳細資訊之受管執行個體旁的按鈕。
4. 請選擇檢視詳細資料。
5. 依次選擇工具、EBS 磁碟區。
6. 若要檢視 EBS 磁碟區的詳細資訊，請在磁碟區 ID 資料欄中選擇其 ID。

初始化和格式化 EBS 磁碟區

使用 Fleet Manager 初始化和格式化 EBS 磁碟區

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇要為其初始化、格式化和掛載 EBS 磁碟區的受管執行個體旁的按鈕。僅當 EBS 磁碟區的磁碟為空時，您才能對磁碟區進行初始化。
4. 請選擇檢視詳細資料。
5. 在工具選單中，選擇 EBS 磁碟區。

6. 選擇要初始化和格式化的 EBS 磁碟區旁的按鈕。
7. 選擇初始化和格式化。
8. 在分割區樣式中，選擇您要用於 EBS 磁碟區的分割區樣式。
9. (選用) 選擇分割區的磁碟機代號。
10. (選用) 輸入分割區名稱以識別分割區。
11. 選擇要用來整理分割區中所存放之檔案和資料的檔案系統。
12. 選擇確認以使 EBS 磁碟區可供使用。在確認 AWS Management Console 之後，您無法從 變更分割區組態，但您可以使用 SSH 或 RDP 登入執行個體來變更分割區組態。

存取 Red Hat 知識庫入口網站

如果您是 Red Hat 客戶 AWS Systems Manager，您可以使用 中的Fleet Manager工具 來存取知識庫入口網站。如果您執行 Red Hat Enterprise Linux (RHEL) 執行個體或使用 AWS上的 RHEL 服務，則會將您視為 Red Hat 客戶。知識庫入口網站包含二進位檔案、知識分享和社群支援的討論論壇，這些論壇僅供 Red Hat 授權客戶使用。

除了 Systems Manager 和 的必要 AWS Identity and Access Management (IAM) 許可之外Fleet Manager，您用來存取主控台的使用者或角色還必須允許 `rhelkb:GetRhelURL`動作存取知識庫入口網站。

存取 Red Hat 知識庫入口網站

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇您想要用於連線至 Red Hat 知識庫入口網站的 RHEL 執行個體。
4. 選擇帳戶管理、存取 Red Hat 知識庫以開啟 Red Hat 知識庫頁面。

如果您在 RHEL上使用 AWS 來執行完全支援的RHEL工作負載，您也可以使用您的 AWS 登入資料，透過 Red Hat 的網站存取 Red Hat 知識庫。

疑難排解受管節點的可用性

對於 Run Command、 Distributor和 等多種 AWS Systems Manager 工具Session Manager，您可以選擇手動選取要在其中執行操作的受管節點。在這類情況下，在您指定要手動選擇節點之後，系統會顯示您可以在其中執行操作的受管節點清單。

該主題提供資訊，以協助您診斷為何您確認正在執行的受管節點未包含在 Systems Manager 的受管節點清單中。

若要讓節點由 Systems Manager 管理，並在受管節點清單中可用，其必須符合三個需求：

- SSM Agent 必須在具有支援作業系統的節點上安裝且執行。

  > **Note**
  >
  > 有些 AWS 受管 Amazon Machine Images(AMIs) 設定為啟動SSM Agent預先安裝 的執行個體。(您還可以設定自訂 AMI 以預先安裝 SSM Agent。) 如需詳細資訊，請參閱 尋找預先安裝了 SSM Agent的 AMIs。

- 對於 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體，您必須將 AWS Identity and Access Management (IAM) 執行個體設定檔連接至執行個體。執行個體設定檔可讓執行個體與 Systems Manager 服務進行通訊。如果您沒有將執行個體設定檔指派給執行個體，則請使用混合式啟用進行註冊，這不是常見案例。
- SSM Agent 必須能夠連線到 Systems Manager 端點，才能自行註冊服務。此後，該受管節點必須可用於服務，該服務可由服務每五分鐘傳送一次訊號以檢查執行個體的運作狀態，來予以確認。
- 在受管節點的狀態成為 `Connection Lost` 的時間至少 30 天之後，該節點可能不會再列於 Fleet Manager 主控台中。若要將其還原至清單，必須解決造成連線中斷的問題。

在驗證受管節點正在執行之後，您可以使用下列命令來檢查 SSM Agent 是否已成功註冊 Systems Manager 服務。在成功註冊之前，此命令不會傳回結果。

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
```

如果註冊成功，且受管節點現在可供 Systems Manager 操作使用，命令會傳回類似如下的結果。

```
{
    "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"
        }
    ]
}
```

如果註冊尚未完成或失敗，此命令會傳回類似如下的結果：

```
{
    "InstanceAssociationStatusInfos": []
}
```

如果命令在 5 分鐘左右後仍未傳回結果，請使用下列資訊協助您故障診斷受管節點的問題。

主題

- 解決方案 1：確認受管節點上已安裝和執行 SSM Agent。
- 解決方案 2：確認已為執行個體 (僅 EC2 執行個體) 指定 IAM 執行個體設定檔
- 解決方案 3：確認服務端點連線能力
- 解決方案 4：確認目標作業系統支援

- 解決方案 5：確認您使用的 與 Amazon EC2 執行個體 AWS 區域 相同
- 解決方案 6：確認套用至受管節點上的 SSM Agent 的代理組態
- 解決方案 7：在受管執行個體上安裝 TLS 憑證
- 使用 ssm-cli 診斷並解決受管節點的可用性問題

解決方案 1：確認受管節點上已安裝和執行 SSM Agent。

確定受管節點上已安裝和執行 SSM Agent 的最新版本。

若要判斷受管節點是否上已安裝和執行 SSM Agent，請參閱 檢查 SSM Agent 狀態並啟動代理程式。

若要在受管節點上安裝或重新安裝 SSM Agent，請參閱下列主題：

- 在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent
- 如何在混合 Linux 節點上安裝 SSM Agent
- 在 Windows Server EC2 執行個體上手動安裝和解除安裝 SSM Agent
- 如何在混合 Windows 節點上安裝 SSM Agent

解決方案 2：確認已為執行個體 (僅 EC2 執行個體) 指定 IAM 執行個體設定檔

對於 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體，請確認該執行個體已使用 AWS Identity and Access Management (IAM) 執行個體設定檔進行設定，該設定檔可讓執行個體與 Systems Manager API 進行通訊。另外，確認您的使用者是否有 IAM 信任政策，其允許您的使用者與 Systems Manager API 通訊。

> **Note**
>
> 內部部署伺服器、邊緣裝置和虛擬機器 (VM) 使用 IAM 服務角色，而非執行個體設定檔。如需詳細資訊，請參閱在混合多雲端環境中建立 Systems Manager 所需的 IAM 服務角色。

若要判斷具有必要許可的執行個體設定檔是否已連接至 EC2 執行個體

1. 在 https://console.aws.amazon.com/ec2/ 開啟 Amazon EC2 主控台。
2. 在導覽窗格中，選擇執行個體。
3. 選擇要檢查執行個體設定檔的執行個體。

4. 在底部窗格的 Description (描述) 標籤上，找出 IAM role (IAM 角色)，然後選擇角色的名稱。
5. 在執行個體設定檔的角色 Summary (摘要) 頁面，在 Permissions (許可) 標籤上，確保 Permissions policies (許可政策) 下已列出 `AmazonSSMManagedInstanceCore`。

   如果改用自訂政策，確保它提供的許可與 `AmazonSSMManagedInstanceCore` 相同。

   在主控台中開啟 `AmazonSSMManagedInstanceCore`

   如需有關可連接到 Systems Manager 的執行個體設定檔的其他政策的資訊，請參閱 設定 Systems Manager 所需的執行個體許可。

解決方案 3：確認服務端點連線能力

請確認執行個體已連線到 Systems Manager 服務端點。提供此連線能力的方式是針對 Systems Manager 建立及設定 VPC 端點，或允許 HTTPS (連接埠 443) 輸出流量傳送至服務端點。

對於 Amazon EC2 執行個體，如果您的虛擬私有雲端 (VPC) 組態允許傳出流量，則 的 Systems Manager 服務端點 AWS 區域 會用來註冊執行個體。但是，如果在其中啟動執行個體的 VPC 組態不允許傳出流量，而且您無法變更此組態以允許連線至公有服務端點，則必須改為為 VPC 設定介面端點。

如需詳細資訊，請參閱使用適用於 Systems Manager 的 VPC 端點來改善 EC2 執行個體的安全性。

解決方案 4：確認目標作業系統支援

確認您選擇的操作可以在預期列出的受管節點類型上執行。某些 Systems Manager 操作可以僅以 Windows 執行個體或 Linux 執行個體為目標。例如，Systems Manager (SSM) 文件 `AWS-InstallPowerShellModule` 和 `AWS-ConfigureCloudWatch` 只能在 Windows 執行個體上執行。在 Run a command (執行命令) 頁面上，如果您選擇其中一個文件，並選取 Choose instances manually (手動選擇執行個體)，則只會列出您的 Windows 執行個體並可供選取。

解決方案 5：確認您使用的 與 Amazon EC2 執行個體 AWS 區域 相同

Amazon EC2 執行個體會建立並在特定區域提供 AWS 區域，例如美國東部 （俄亥俄） 區域 (us-east-2) 或歐洲 （愛爾蘭） 區域 (eu-west-1)。確保您使用的 Amazon EC2 AWS 區域 執行個體與您想要使用的 Amazon EC2 執行個體相同。如需詳細資訊，請參閱《 AWS Management Console入門》中的選擇區域。

解決方案 6：確認套用至受管節點上的 SSM Agent 的代理組態

確認套用至受管節點上的 SSM Agent 的代理組態正確無誤。如果代理組態不正確，節點就無法連線到必要的服務端點，或 Systems Manager 可能會不正確地識別受管節點的作業系統。如需詳細資訊，請

參閱 將 SSM Agent 設定為使用 Linux 節點上的代理 和 將 SSM Agent 設定為使用 Windows Server 執行個體的代理。

解決方案 7：在受管執行個體上安裝 TLS 憑證

Transport Layer Security (TLS) 憑證必須安裝在您使用的每個受管執行個體上 AWS Systems Manager。 AWS 服務 使用這些憑證來加密對其他執行個體的呼叫 AWS 服務。

依預設，從任何 Amazon Machine Image (AMI) 建立的每個 Amazon EC2 執行個體上都已安裝 TLS 憑證。大多數現代作業系統在其信任存放區中都包含來自 Amazon Trust Services CA 的必要 TLS 憑證。

若要確認執行個體是否已安裝所需的憑證，請根據執行個體的作業系統執行下列命令。請務必使用 AWS 區域 受管執行個體所在的 取代 URL *###*部分。

Linux & macOS

```
curl -L https://ssm.region.amazonaws.com
```

Windows

```
Invoke-WebRequest -Uri https://ssm.region.amazonaws.com
```

命令應該會傳回 `UnknownOperationException` 錯誤。如果您收到 SSL/TLS 錯誤訊息，則可能未安裝所需的憑證。

如果您發現基本作業系統上未安裝必要的 Amazon Trust Services CA 憑證、Amazon AMIs 未提供的執行個體上，或是您自己的內部部署伺服器和 VMs上，您必須安裝並允許 Amazon Trust Services 的憑證，或使用 AWS Certificate Manager (ACM) 來建立和管理受支援整合服務的憑證。

您的每個受管執行個體必須安裝下列其中一個 Transport Layer Security (TLS) 憑證。

- Amazon 根 CA 1
- Starfield Services 根憑證授權機構：G2
- Starfield Class 2 憑證授權機構

如需有關使用 ACM 的資訊，請參閱《AWS Certificate Manager 使用者指南》。

如果您運算環境中的憑證是由群組政策物件 (GPO) 管理，則您可能需要將群組政策設定為包含其中一個憑證。

如需 Amazon 根憑證和 Starfield 憑證的詳細資訊，請參閱部落格文章如何準備 AWS移至其自己的憑證授權單位。

使用 **ssm-cli** 診斷並解決受管節點的可用性問題

ssm-cli 是獨立的命令列工具，包含在 SSM Agent 安裝中。當您在機器上安裝 SSM Agent 3.1.501.0 或更新版本時，可以在該機器上執行 ssm-cli 命令。這些命令的輸出可協助您判斷機器是否符合由 管理的 Amazon EC2 執行個體non-EC2 機器的最低需求 AWS Systems Manager，因此已新增至 Systems Manager 中的受管節點清單。(SSM Agent 3.1.501.0 版已於 2021 年 11 月發行。)

最低需求

若要讓 Amazon EC2 執行個體non-EC2 機器由 管理 AWS Systems Manager，並可在受管節點清單中使用，必須符合三個主要需求：

- SSM Agent 必須在執行受支援作業系統的機器上安裝且執行。

  某些 EC2 的 AWS 受管 Amazon Machine Images(AMIs) 設定為啟動SSM Agent預先安裝 的執行個體。(您還可以設定自訂 AMI 以預先安裝 SSM Agent。) 如需詳細資訊，請參閱 尋找預先安裝了 SSM Agent的 AMIs。
- 必須將提供與 Systems Manager 服務通訊所需許可的 AWS Identity and Access Management (IAM) 執行個體描述檔 （適用於 EC2 執行個體） 或 IAM 服務角色 （適用於non-EC2 機器） 連接至機器。
- SSM Agent 必須能夠連線到 Systems Manager 端點，才能自行註冊服務。此後，該受管節點必須可用於服務，該服務可由服務每五分鐘傳送一次訊號以檢查受管節點的運作狀態，來予以確認。

ssm-cli 中的預先設定命令

包含了預先設定的命令，收集所需資訊，協助您診斷為何您確認正在執行的機器未包含在 Systems Manager 的受管節點清單中。這些命令會在您指定 get-diagnostics 選項時執行。

在機器上執行以下命令來使用 ssm-cli 協助您解決受管節點的可用性問題。

Linux & macOS

```
ssm-cli get-diagnostics --output table
```

Windows

在 Windows Server 機器上，您必須在執行命令之前瀏覽至 C:\Program Files\Amazon\SSM 目錄。

```
ssm-cli.exe get-diagnostics --output table
```

PowerShell

在 Windows Server 機器上，您必須在執行命令之前瀏覽至 C:\Program Files\Amazon\SSM 目錄。

```
.\ssm-cli.exe get-diagnostics --output table
```

此命令會傳回與以下表格類似的表格作為輸出：

> **Note**
>
> 對 ssmmessages、s3、kms、 logs和 monitoring端點的連線檢查適用於其他選用功能 Session Manager，例如 可記錄到 Amazon Simple Storage Service (Amazon S3) 或 Amazon CloudWatch Logs 和 use AWS Key Management Service (AWS KMS) 加密的功能。

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 ssm-cli 所執行之每項檢查的其他詳細資訊。

ssm-cli 診斷檢查

Check	詳細資訊
Amazon EC2 執行個體中繼資料服務	表示受管節點是否能夠存取中繼資料服務。失敗的測試表示 http://169.254.169.254 發生連線問題，這可能是由本機路由、代理或作業系統 (OS) 防火牆和代理組態所造成的。
混合式執行個體註冊	表示 SSM Agent 是否使用混合啟用註冊。
連線至 ssm 端點	表示此節點是否可以在 TCP 連接埠 443 上連線 Systems Manager 的服務端點。失敗的測試https://ssm. *region*.amazonaws.com 會根據節點所在的 AWS 區域 ，將連線問題指示為 。連線問題可能是由 VPC 組態造成，包括安全群組、網路存取控制清單、路由表或作業系統防火牆和代理。
連線至 ec2messages 端點	表示此節點是否可以在 TCP 連接埠 443 上連線 Systems Manager 的服務端點。失敗的測試https://ec2messages. *region*.amazonaws.com 會根據節點所在的 AWS 區域 ，將連線問題指示為 。連線問題可能是由 VPC 組態造成，包括安全群組、網路存取控制清單、路由表或作業系統防火牆和代理。

Check	詳細資訊
連線至 ssmmessages 端點	表示此節點是否可以在 TCP 連接埠 443 上連線 Systems Manager 的服務端點。失敗的測試https://ssmmessages. *region*.amazonaws.com 會根據節點所在的 AWS 區域 ，將連線問題指示為 。連線問題可能是由 VPC 組態造成，包括安全群組、網路存取控制清單、路由表或作業系統防火牆和代理。
連線至 s3 端點	表示此節點是否可以在 TCP 連接埠 443 上連線 Amazon Simple Storage Service 的服務端點。失敗的測試https://s3. *region*.amazonaws.com 會根據節點所在的 AWS 區域 ，將連線問題指示為 。節點無需與此端點連線，即可出現在受管節點清單中。
連線至 kms 端點	指出節點是否能夠在 TCP 連接埠 443 AWS Key Management Service 上達到 的服務端點。失敗的測試https://kms. *region*.amazonaws.com 會根據節點所在的 AWS 區域 ，將連線問題指示為 。節點無需與此端點連線，即可出現在受管節點清單中。
連線至 logs 端點	表示此節點是否可以在 TCP 連接埠 443 上連線 Amazon CloudWatch Logs 的服務端點。失敗的測試https://logs. *region*.amazonaws.com 會根據節點所在的 AWS 區域 ，將連線問題指示為 。節點無需與此端點連線，即可出現在受管節點清單中。

Check	詳細資訊
連線至 monitoring 端點	表示此節點是否可以在 TCP 連接埠 443 上連線 Amazon CloudWatch 的服務端點。失敗的測試https://monitoring. *region*.amazonaws.com 會根據節點所在的 AWS 區域 ，將連線問題指示為 。節點無需與此端點連線，即可出現在受管節點清單中。
AWS 登入資料	表示 SSM Agent 基於連接到機器的 IAM 執行個體設定檔 (適用於 EC2 執行個體) 或 IAM 服務角色 (適用於非 EC2 機器) 是否具有必要的憑證。失敗的測試表示沒有 IAM 執行個體設定檔或 IAM 服務角色連接至機器，或不包含 Systems Manager 所需的許可。
代理程式服務	表示 SSM Agent 服務是否正在執行，以及服務是否以 root (適用於 Linux 或 macOS) 或 SYSTEM 的身分執行 (適用於 Windows Server)。失敗的測試表示 SSM Agent 服務未執行，或未以 root 或 SYSTEM 身分執行。
代理組態	表示 SSM Agent 是否設定為使用代理。
Sysprep 映像狀態 (僅限 Windows)	指示節點上 Sysprep 的狀態。如果 Sysprep 狀態為 IMAGE_STATE_COMPLETE 以外的值，則不會在節點上啟動 SSM Agent。
SSM Agent 版本	表示是否已安裝 SSM Agent 的最新可用版本。

AWS Systems Manager 混合啟用

若要設定non-EC2 機器以在 AWS Systems Manager 混合和多雲端環境中搭配 使用，您可以建立混合啟用。作為受管節點得到支援的非 EC2 機器類型包含：

- 您內部部署的伺服器 (內部部署伺服器)
- AWS IoT Greengrass 核心裝置

- AWS IoT 和非AWS 邊緣裝置
- 虛擬機器 (VM)，包含其他雲端環境中的 VM

執行 create-activation 命令以啟動混合啟用程序後，命令回應中會顯示啟用碼和 ID。然後，您可以使用該啟用碼和 ID 以及相關命令在機器上安裝 SSM Agent，如在混合 Linux 節點上安裝 SSM Agent中的步驟 3 和在混合 Windows Server 節點上安裝 SSM Agent中的步驟 4 所述。

此啟用程序適用於所有non-EC2 機器類型，核心裝置除外 AWS IoT Greengrass 。如需有關為 Systems Manager 設定 AWS IoT Greengrass 核心裝置的詳細資訊，請參閱 使用 Systems Manager 管理邊緣裝置。

> **Note**
>
> 目前不支援非 EC2 macOS 機器。

關於 Systems Manager 執行個體方案

AWS Systems Manager 提供 standard-instances 層和 advanced-instances 層。兩者都支援混合多雲端環境中的受管節點。Standard-instances 方案可讓您 AWS 帳戶 為每個機器註冊最多 1，000 個機器 AWS 區域。如果您需要在單一帳戶和區域中登錄 1,000 部以上的機器，則使用進階執行個體層。您可以在進階執行個體層中，視需要建立多個受管節點。所有針對 Systems Manager 設定的受管節點，按使用次數計價。如需啟用進階執行個體方案的詳細資訊，請參閱 開啟 advanced-instances 方案。如需定價的詳細資訊，請參閱AWS Systems Manager 定價。

請注意下列有關標準執行個體方案和進階執行個體方案的其他資訊：

- 進階執行個體也可讓您使用 連線到混合多雲端環境中non-EC2 節點 AWS Systems Manager Session Manager。 Session Manager 提供執行個體的互動式 shell 存取。如需詳細資訊，請參閱AWS Systems Manager Session Manager。
- 此 standard-instances 配額也適用於使用 Systems Manager 內部部署啟用的 EC2 執行個體 (這不是常見案例)。
- 若要修補 Microsoft 在虛擬機器 (VM) 內部部署執行個體上發行的應用程式，請啟用進階執行個體層。使用進階執行個體層會產生費用。修補由 Microsoft 在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上發行的應用程式無須另外付費。如需詳細資訊，請參閱在 Windows Server 上由 Microsoft 發行的修補應用程式。

AWS Systems Manager庫存

AWS Systems Manager 庫存可讓您清楚了解 AWS 運算環境。您可以使用庫存，從受管的受管節點收集中繼資料。您可以將此中繼資料存放在一個集中的 Amazon Simple Storage Service (Amazon S3) 的儲存貯體，然後使用內建工具來查詢資料，並快速判斷執行軟體政策所需的軟體和組態是哪些節點以及需要更新的是哪些節點。您可以透過使用一鍵式程序來設定所有受管節點上的庫存。您也可以 AWS 帳戶 使用 Amazon Athena 設定和檢視來自多個 AWS 區域 和 的庫存資料。若要開始使用庫存，請開啟 Systems Manager 主控台。在導覽窗格中，選擇庫存。

如果 Systems Manager 庫存收集的預先設定中繼資料類型不符合您的需求，您可以建立自訂庫存。自訂庫存只是一個 JSON 檔案，內含您提供並新增到特定目錄中的受管節點。Systems Manager 庫存收集資料時，它會擷取此自訂庫存資料。例如，如果您執行大型資料中心，您可以將每個伺服器的機架位置指定為自訂庫存。您就可以在檢視其他庫存資料時檢視機架空間資料。

> **Important**
>
> Systems Manager 庫存只收集受管節點的中繼資料。庫存不會存取專屬資訊或資料。

下表列出您可以使用 Systems Manager 清查功能收集的資料類型。此表還介紹了針對目標節點的不同產品，以及您可以指定的收集間隔。

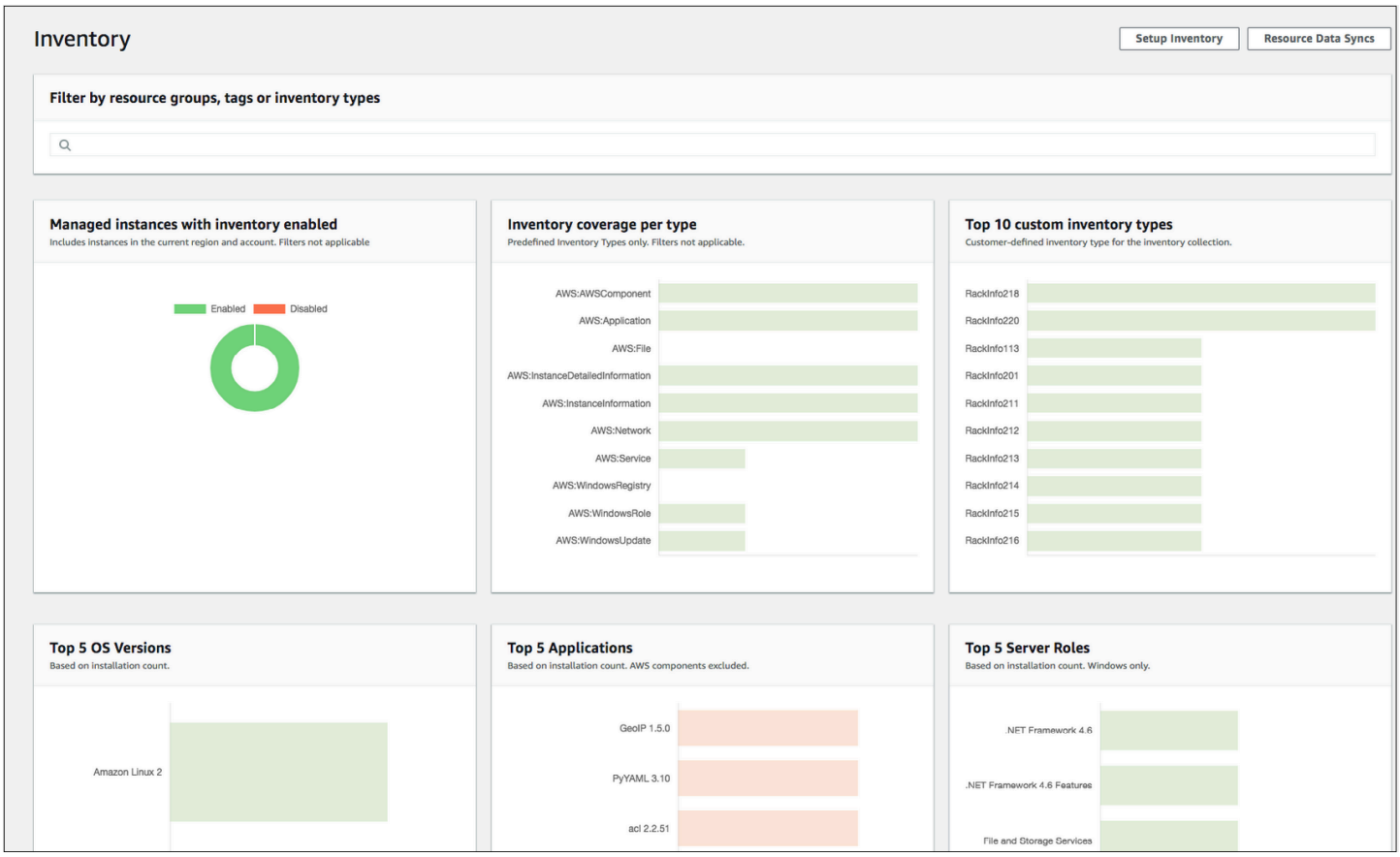
組態	詳細資訊
中繼資料類型	您能夠設定清查功能來收集下列類型的資料： • 應用程式：應用程式名稱、發佈者、版本等。 • AWS 元件：EC2 驅動程式、代理程式、版本等。 • 檔案：名稱、大小、版本、安裝日期、修改時間和上次存取時間等。 • 網路組態：IP 地址、MAC 地址、DNS、閘道、子網路遮罩等。 • Windows 更新：Hotfix ID、安裝人員、安裝日期等。 • 執行個體詳細資訊：CPUModel、CPUCores CPUs、CPU、CPUSpeedMHz、CPUSoc

組態	詳細資訊
	kets、CPUHyperThreadEnabled、OSServicePacketc。 • 服務：名稱、顯示名稱、狀態、相依服務、服務類型、開啟類型等。 • 標籤：指派到節點的標籤。 • Windows 登錄檔：登錄機碼路徑、數值名稱、數值類型和數值。 • Windows 角色：名稱、顯示名稱、路徑、功能類型、安裝狀態等。 • 自訂庫存：指派給受管節點的中繼資料，如使用自訂庫存 中所述。 **Note** 如需檢視庫存收集的所有中繼資料清單，請參閱按庫存收集的中繼資料。
目標節點	您可以選擇使用標籤 AWS 帳戶來清查 中所有受管節點、個別選取節點或節點目標群組。如需從所有受管節點收集清查資料的詳細資訊，請參閱清查 中的所有受管節點 AWS 帳戶。
收集資訊的時間	您能以分鐘、小時和天來指定收集程序的間隔時間。收集程序的最短間隔時間為 30 分鐘。

Note

依據所收集的資料量而定，系統可能要花費幾分鐘的時間，才能將資料回報至您指定的輸出結果。收集資訊後，資料會透過安全的 HTTPS 管道傳送至純文字 AWS 存放區，該存放區只能從您的 存取 AWS 帳戶。

您可以在 Inventory (庫存) 頁面的 Systems Manager 主控台中檢視資料，其中包含數個預先定義的卡，可協助您查詢資料。



> **Note**
>
> 庫存卡會自動篩選掉狀態為已終止和已停止的 Amazon EC2 受管執行個體。對於內部部署和 AWS IoT Greengrass 核心裝置受管節點，庫存卡會自動篩選出狀態為已終止的節點。

如果您建立資源資料同步，在單一 Amazon S3 儲存貯體中同步和存放所有資料，則可以在 Inventory Detailed View (庫存詳細檢視) 頁面上深入了解資料。如需詳細資訊，請參閱查詢來自多個區域和帳戶的清查資料。

支援 EventBridge

Amazon EventBridge 規則中支援此 Systems Manager 工具做為事件類型。如需詳細資訊，請參閱 使用 Amazon EventBridge 監控 Systems Manager 及 參考：Systems Manager 的 Amazon EventBridge 事件模式和類型。

目錄

- 進一步了解 Systems Manager 庫存
- 設定 Systems Manager 庫存
- 設定清查收集
- 查詢來自多個區域和帳戶的清查資料
- 使用篩選條件查詢清查收集
- 彙總庫存資料
- 使用自訂庫存
- 檢視清查歷程記錄和變更追蹤
- 停用資料收集和刪除庫存資料
- 將自訂庫存中繼資料指派給受管節點
- 使用 AWS CLI 設定庫存資料收集
- 演練：使用資源資料同步來彙總庫存資料
- 對 Systems Manager 庫存的問題進行故障診斷

進一步了解 Systems Manager 庫存

當您設定 AWS Systems Manager 庫存時，您可以指定要收集的中繼資料類型、應收集中繼資料的受管節點，以及中繼資料收集的排程。這些組態會與 一起儲存 AWS 帳戶 為 AWS Systems Manager State Manager關聯。簡單來說，關聯就是一種組態。

> **Note**
>
> 庫存只會收集中繼資料，該服務不會收集任何個人資料或私有資料。

主題

- 按庫存收集的中繼資料
- 使用檔案與 Windows 登錄檔清查

按庫存收集的中繼資料

下方範例將顯示每個 AWS Systems Manager 庫存外掛程式收集的中繼資料完整清單。

```
{
    "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**
>
> - 對於 `"typeName": "AWS:InstanceInformation"`，`InstanceStatus` 可以是下列其中一項：作用中、連線遺失、已停止、已終止。
> - 隨著 2.5 版的發行，RPM 套件管理員將以 Epoch 來取代 Serial 屬性。與 Serial 相同，Epoch 屬性也屬於依序遞增的整數。當您利用 `AWS:Application` 類型建立庫存時，請注意：Epoch 的值越大，表示版本越新。如果 Epoch 值相同或空白，請使用 Version 值或 Release 屬性來判斷是否為較新版本。
> - 某些中繼資料在 Linux 執行個體中不可用。具體而言，對於 "typeName": "AWS:Network"，Linux 執行個體尚不支援下列中繼資料類型。而 Windows 則支援這些類型。
>   - { "name": "SubnetMask", "dataType": "STRING"}
>   - { "name": "DHCPServer", "dataType": "STRING"}
>   - { "name": "DNSServer", "dataType": "STRING"}
>   - { "name": "Gateway", "dataType": "STRING"}

使用檔案與 Windows 登錄檔清查

AWS Systems Manager 清查可讓您搜尋和清查 Windows、Linux 和macOS作業系統上的檔案。同時，您還能搜尋 Windows 登錄檔並建立庫存。

檔案：您能夠收集與檔案相關的中繼資料資訊，包括檔案名稱、檔案建立時間、上次修改和存取檔案的時間，以及檔案大小等。若要開始收集檔案庫存資料，您必須指定要執行庫存建立作業的檔案路徑、能定義欲建立庫存檔案類型的一個或多個模式，以及是否要以遞迴方式周遊該路徑。Systems Manager 會為符合模式之指定路徑中檔案的所有檔案中繼資料建立庫存。檔案庫存資料會採用以下參數輸入。

```
{
"Path": string,
"Pattern": array[string],
"Recursive": true,
```

```
"DirScanLimit" : number // Optional
}
```

- Path：要建立檔案庫存的目錄路徑。在 Windows 作業系統中，您可以使用 %PROGRAMFILES% 等環境變數，前提是該變數必須對應至單一目錄路徑。舉例來說，如果您使用的 %PATH% 變數對應至多個目錄路徑，庫存便會擲出錯誤訊息。
- Pattern：用來辨識檔案的模式陣列。
- Recursive：指出庫存是否應以遞迴方式周遊目錄的布林值。
- DirScanLimit：此為選用值，可用來指定要掃描的目錄數量。只要使用此參數，就能徹底降低對受管節點效能的影響。庫存掃描最多 5，000 個目錄。

> **Note**
>
> 在所有指定路徑中，庫存最多能收集 500 個檔案的中繼資料。

下方範例會說明如何在執行檔案庫存時指定參數。

- 在 Linux 和 macOS 作業系統上，在 `/home/ec2-user` 目錄 (不包含所有子目錄) 中收集 .sh 檔案的中繼資料。

```
[{"Path":"/home/ec2-user","Pattern":["*.sh", "*.sh"],"Recursive":false}]
```

- 在 Windows 作業系統上，讓庫存以遞迴方式在 Program Files 資料夾 (包含子目錄) 中收集所有「.exe」檔案的中繼資料。

```
[{"Path":"C:\Program Files","Pattern":["*.exe"],"Recursive":true}]
```

- 在 Windows 作業系統上，讓庫存收集特定日誌模式的中繼資料。

```
[{"Path":"C:\ProgramData\Amazon","Pattern":["*amazon*.log"],"Recursive":true}]
```

- 執行遞迴收集時，限制目錄數量。

```
[{"Path":"C:\Users","Pattern":["*.ps1"],"Recursive":true, "DirScanLimit": 1000}]
```

Windows 登錄檔：您可以收集 Windows 登錄機碼和值。而且，您還能選擇機碼路徑，並以遞迴方式收集所有機碼和值。此外，您也可以收集指定路徑的特定登錄機碼及其值。庫存將收集機碼路徑、名稱、類型與值。

```
{
"Path": string,
"Recursive": true,
"ValueNames": array[string] // optional
}
```

- Path：登錄機碼的路徑。
- Recursive：指出庫存是否應以遞迴方式周遊登錄檔路徑的布林值。
- ValueNames：此為數值名稱陣列，可用來執行登錄機碼庫存。如果使用該參數，則 Systems Manager 僅會為指定路徑的特定數值名稱建立庫存。

> **Note**
>
> 在所有指定路徑中，庫存最多能收集 250 個登錄機碼值。

下方範例會說明如何在執行 Windows 登錄檔庫存時指定參數。

- 以遞迴方式收集指定路徑的所有機碼與值。

  ```
  [{"Path":"HKEY_LOCAL_MACHINE\SOFTWARE\Amazon","Recursive": true}]
  ```

- 收集指定路徑的所有機碼與值 (關閉遞迴搜尋)。

  ```
  [{"Path":"HKEY_LOCAL_MACHINE\SOFTWARE\Intel\PSIS\PSIS_DECODER", "Recursive": false}]
  ```

- 使用 `ValueNames` 選項來收集特定機碼。

  ```
  {"Path":"HKEY_LOCAL_MACHINE\SOFTWARE\Amazon\MachineImage","ValueNames":["AMIName"]}
  ```

設定 Systems Manager 庫存

在您使用 AWS Systems Manager 庫存收集受管節點上執行的應用程式、服務、 AWS 元件等中繼資料之前，建議您設定資源資料同步，將庫存資料的儲存集中在單一 Amazon Simple Storage Service

(Amazon S3) 儲存貯體中。同時建議您設定庫存事件的 Amazon EventBridge 監控。這些程序可讓您更輕鬆地檢視和管理庫存資料和收集。

主題

- 建立庫存的資源資料同步
- 使用 EventBridge 監控庫存事件

建立庫存的資源資料同步

本主題說明如何設定 AWS Systems Manager 清查的資源資料同步。如需 Systems Manager Explorer 資源資料同步的相關資訊，請參閱《設定 Systems Manager Explorer 以顯示來自多個帳戶和區域的資料》。

關於資源資料同步

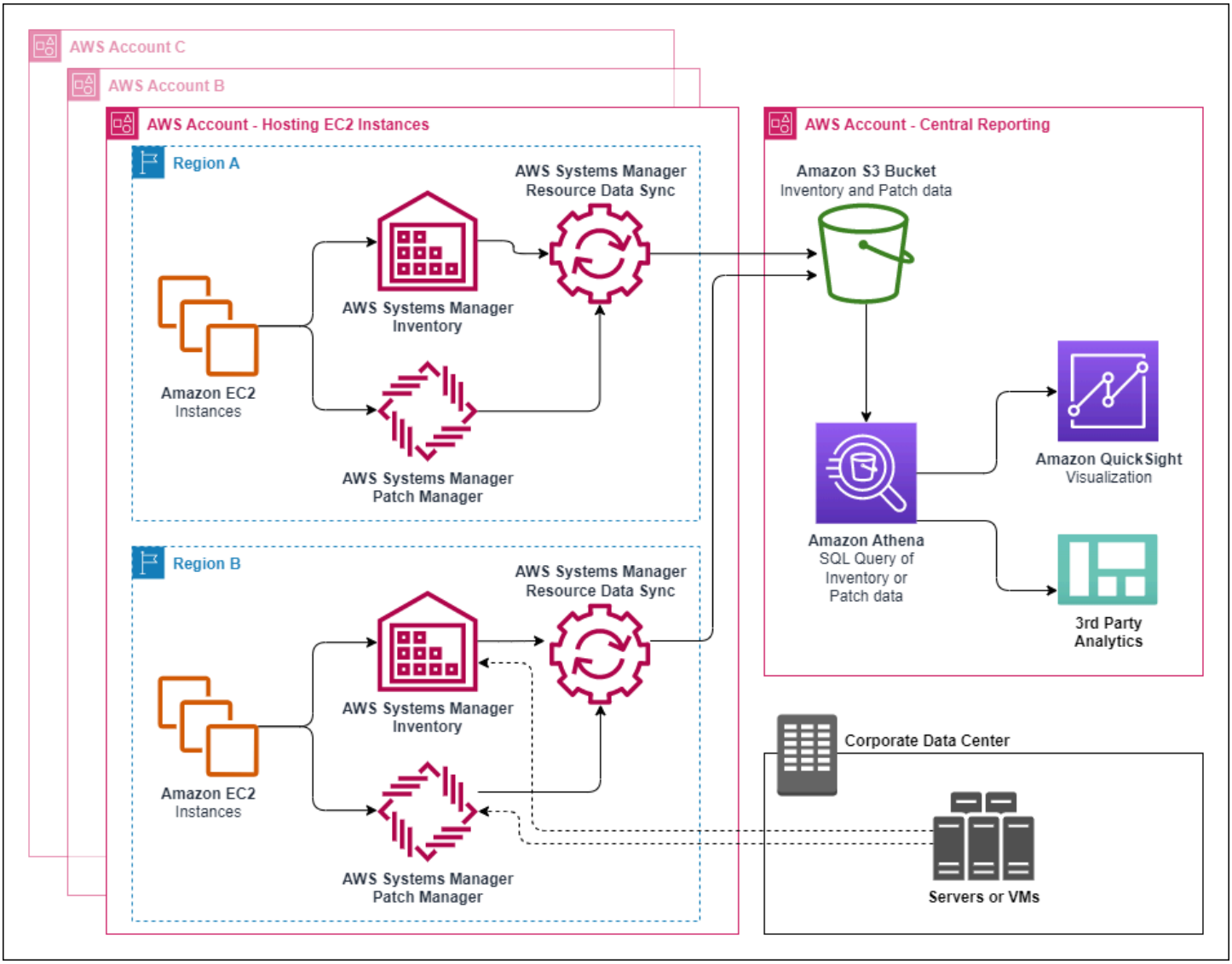
您可以使用 Systems Manager 資源資料同步，將從所有受管節點收集到的庫存資料傳送至單一 Amazon Simple Storage Service (Amazon S3) 儲存貯體。然後，資源資料同步會在系統收集新的清查資料時自動更新集中的資料。當所有庫存資料存放至目標 Amazon S3 儲存貯體後，您就能利用 Amazon Athena 和 Amazon QuickSight 等服務來查詢及分析彙總的資料。

例如，假設您將清查設定成在 150 個受管節點機群上，收集正在執行作業系統 (OS) 和應用程式的相關資料。其中部分執行個體位於內部部署資料中心，而其他節點則在多個 AWS 區域的 Amazon Elastic Compute Cloud (Amazon EC2) 中運作。如果您「沒有」設定清查的資源資料同步，便需要手動收集每個受管節點的清查資料集合，或是建立指令碼以收集這些資訊。接著，您還要將資料傳輸至應用程式，才能執行查詢和分析作業。

透過資源資料同步，您只要執行一次性操作，即可同步來自所有受管節點的任何清查資料。成功建立同步後，Systems Manager 會建立所有庫存資料的基準，並將存放至目標 Amazon S3 儲存貯體。收集到新的庫存資料時，Systems Manager 會自動更新 Amazon S3 儲存貯體中的資料。如此一來，您就能利用快速且具成本效益的方式，將資料連接至 Amazon Athena 和 Amazon QuickSight。

圖 1 顯示資源資料同步如何將混合多雲端環境中的 Amazon EC2 和其他機器類型的庫存資料，彙總至目標 Amazon S3 儲存貯體。此圖表也顯示資源資料同步如何與多個 AWS 帳戶 和 搭配使用 AWS 區域。

圖表 1：具有多個 AWS 帳戶 和 的資源資料同步 AWS 區域



如果您刪除受管節點，資源資料同步仍會保留已刪除節點的庫存檔案。不過，對於執行中的節點，當有新的檔案建立並寫入 Amazon S3 儲存貯體時，資源資料同步會自動覆寫舊的庫存檔案。如果您想要追蹤一段時間內的庫存變更，您可以使用 AWS Config 服務來追蹤`SSM:ManagedInstanceInventory`資源類型。如需詳細資訊，請參閱 入門 AWS Config。

使用本節中的程序，透過 Amazon S3 和 AWS Systems Manager 主控台建立庫存的資源資料同步。您也可以使用 AWS CloudFormation 來建立或刪除資源資料同步。若要使用 AWS CloudFormation，請將 `AWS::SSM::ResourceDataSync` 資源新增至您的 AWS CloudFormation 範本。如需相關資訊，請參閱下列任一文件資源：

- 在 AWS CloudFormation 中用於資源資料同步的 資源 AWS Systems Manager （部落格）
- 《AWS CloudFormation 使用者指南》中的使用 AWS CloudFormation 範本

> **Note**
>
> 您可以使用 AWS Key Management Service (AWS KMS) 來加密 Amazon S3 儲存貯體中的庫存資料。如需如何使用 AWS Command Line Interface (AWS CLI) 建立加密同步，以及如何在 Amazon Athena 和 Amazon QuickSight 中使用集中式資料的範例，請參閱 演練：使用資源資料同步來彙總庫存資料。

開始之前

建立資源資料同步前，請使用以下程序建立中央 Amazon S3 儲存貯體，以存放彙總的庫存資料。程序會說明如何指派儲存貯體政策，讓 Systems Manager 將庫存資料寫入多個帳戶的儲存貯體。如果您要使用已有的 Amazon S3 儲存貯體來彙總資源資料同步的庫存資料，則必須在以下程序中設定儲存貯體來使用政策。

> **Note**
>
> 如果該儲存貯體設定為使用 Object Lock，則 Systems Manager 庫存無法將資料新增至指定的 Amazon S3 儲存貯體。確認您為資源資料同步建立或選擇的 Amazon S3 儲存貯體未設定為使用 Amazon S3 Object Lock。如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的 Amazon S3 Object Lock 如何運作。

建立和設定資源資料同步的 Amazon S3 儲存貯體

1. 開啟位於 https://console.aws.amazon.com/s3/ 的 Amazon S3 主控台。
2. 建立儲存貯體以存放您彙整的清查資料。如需詳細資訊，請參閱 Amazon Simple Storage Service 主控台使用者指南中的建立儲存貯體。請記下儲存貯體名稱和您建立儲存貯體 AWS 區域 所在的 。
3. 選擇許可索引標籤，然後選擇儲存貯體政策。
4. 複製下列儲存貯體政策並貼至政策編輯器。將 *amzn-s3-demo-bucket* 取代為您建立的 S3 儲存貯體名稱。以有效的 AWS 帳戶 ID 號碼取代 *account_ID_number*。

```
{
    "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:*:account_ID_number:resource-data-
sync/*"
            }
        }
      }
    ]
}
```

5. 儲存您的變更。

建立庫存的資源資料同步

遵循以下程序，使用 Systems Manager 主控台建立 Systems Manager 庫存的資源資料同步。如需如何使用 建立資源資料同步的詳細資訊 AWS CLI，請參閱 使用 AWS CLI 設定庫存資料收集。

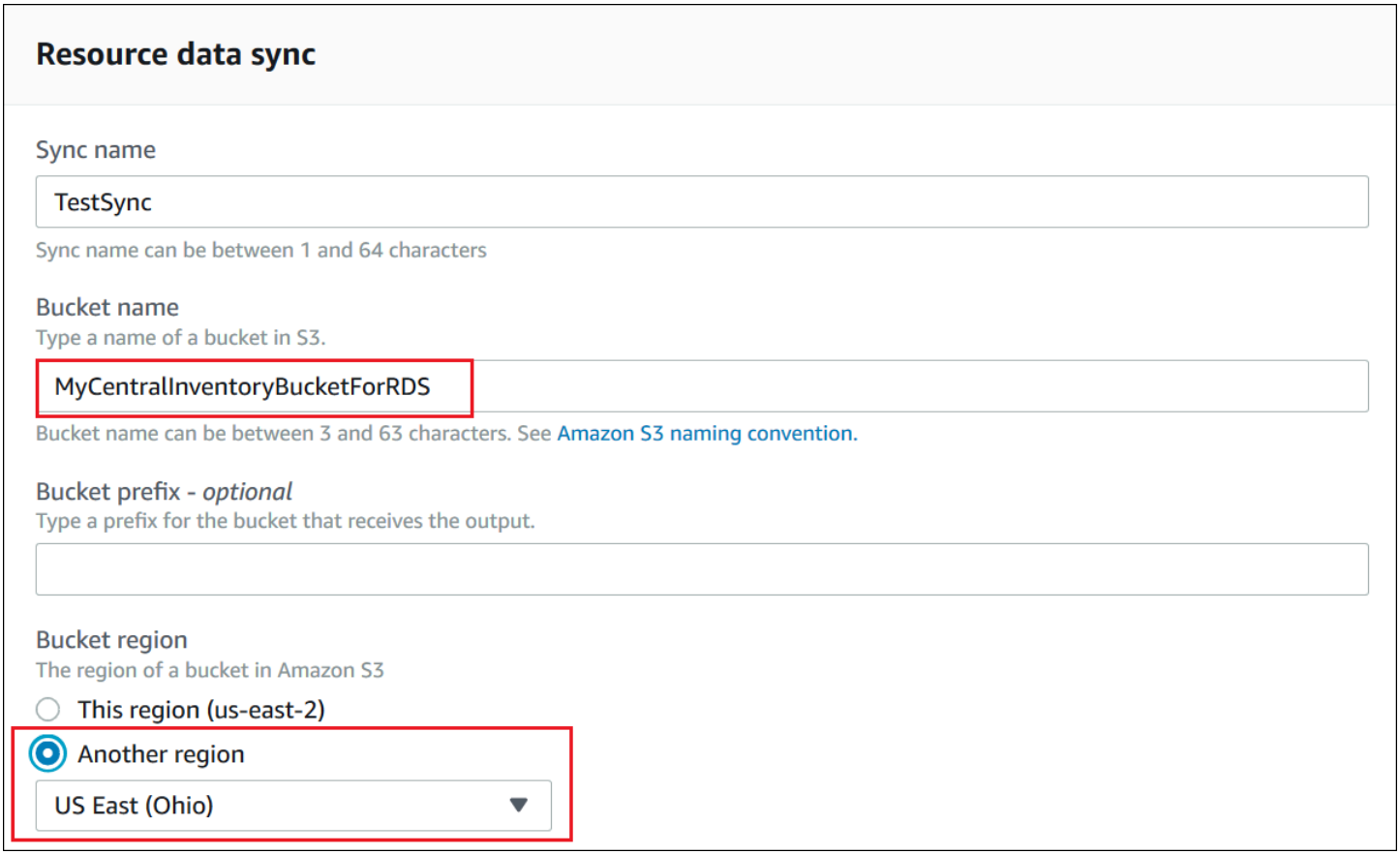
建立資源資料同步

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 在帳戶管理選單中，選擇資源資料同步。
4. 選擇建立資源資料同步。
5. 在 Sync name (同步名稱) 欄位中，輸入同步組態的名稱。
6. 在 Bucket name (儲存貯體名稱) 欄位中，輸入您使用為資源資料同步建立和設定 Amazon S3 儲存貯體處理程序建立的 Amazon S3 儲存貯體名稱。
7. (選用) 在 Bucket prefix (儲存貯體字首) 欄位中，輸入 Amazon S3 儲存貯體字首 (子目錄) 的名稱。
8. 如果您建立的 Amazon S3 儲存貯體位於目前的 AWS 區域，請在 Bucket region (儲存貯體區域) 欄位中選擇 This region (此區域)。如果儲存貯體位於不同的 AWS 區域，請選擇 Another region (其他區域)，然後輸入區域的名稱。

   > **Note**
   >
   > 如果同步與目標 Amazon S3 儲存貯體位於不同區域，您可能需要支付資料傳輸費用。如需詳細資訊，請參閱 Amazon S3 定價。

9. (選用) 在 KMS Key ARN (KMS 金鑰 ARN) 欄位中，輸入或貼上 KMS 金鑰 ARN，其可用來加密 Amazon S3 中的庫存資料。
10. 選擇建立。

若要同步來自多個 的庫存資料 AWS 區域，您必須在每個區域中建立資源資料同步。在您要收集庫存資料的每個 AWS 區域 中重複此程序，並將其傳送至中央 Amazon S3 儲存貯體。在每個區域中建立同步時，請在 Bucket name (儲存貯體名稱) 欄位中指定中央 Amazon S3 儲存貯體。接著，透過 Bucket region (儲存貯體區域) 選項，即可選擇建立中央 Amazon S3 儲存貯體的區域，如下方螢幕擷取畫面所示。下次執行關聯以收集清查資料時，Systems Manager 便會將資料存放至中央 Amazon S3 儲存貯體。



為 AWS Organizations中定義的帳戶建立庫存資源資料同步

您可以將 中 AWS 帳戶 定義的庫存資料同步 AWS Organizations 到中央 Amazon S3 儲存貯體。完成以下處理程序後，庫存資料便會同步到中央儲存貯體中的個別 Amazon S3 金鑰字首。每個金鑰字首代表不同的 AWS 帳戶 ID。

開始之前

開始之前，請確認您已在 AWS 帳戶 中設定 和 AWS Organizations。如需詳細資訊，請參閱《AWS Organizations 使用者指南》。

此外，請注意，您必須為 中 AWS 帳戶 定義的每個 AWS 區域 建立組織型資源資料同步 AWS Organizations。

建立中央 Amazon S3 儲存貯體

使用以下程序建立中央 Amazon S3 儲存貯體，以存放彙總的庫存資料。程序會說明如何指派儲存貯體政策，讓 Systems Manager 將庫存資料寫入您 AWS Organizations 帳戶 ID 中的儲存貯體。如果您要使用已有的 Amazon S3 儲存貯體來彙總資源資料同步的庫存資料，則必須在以下程序中設定儲存貯體來使用政策。

為 中定義的多個帳戶建立和設定資源資料同步的 Amazon S3 儲存貯體 AWS Organizations

1. 開啟位於 https://console.aws.amazon.com/s3/ 的 Amazon S3 主控台。
2. 建立儲存貯體以存放您彙整的清查資料。如需詳細資訊，請參閱 Amazon Simple Storage Service 主控台使用者指南中的建立儲存貯體。請記下儲存貯體名稱和您建立儲存貯體 AWS 區域 所在的 。
3. 選擇許可索引標籤，然後選擇儲存貯體政策。
4. 複製下列儲存貯體政策並貼至政策編輯器。將 *amzn-s3-demo-bucket* 和 *organization-id* 取代為您建立的 Amazon S3 儲存貯體名稱和有效的 AWS Organizations 帳戶 ID。

   或者，您也可以使用 Amazon S3 字首 (子目錄) 取代 *bucket-prefix*。如果您沒有建立字首，請在以下政策中，從 ARN 移除 *bucket-prefix/*。

```
{
  "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=*/*"
      ]
    }
  ]
}
```

為 AWS Organizations中定義的帳戶建立庫存資源資料同步

下列程序說明如何使用 AWS CLI 為 中定義的帳戶建立資源資料同步 AWS Organizations。您必須使用 AWS CLI 來執行此任務。您也必須為 中 AWS 帳戶 定義的每個 AWS 區域 和 執行此程序 AWS Organizations。

為 AWS Organizations (AWS CLI) 中定義的帳戶建立資源資料同步

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請安裝並設定 。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

2. 執行以下命令，以驗證您是否沒有任何其他以 AWS Organizations為基礎的資源資料同步。您可以擁有多個標準同步，包括多個標準同步和以 Organizations 為基礎的同步。但是，您只能有一個以 Organizations 為基礎的資源資料同步。

   ```
   aws ssm list-resource-data-sync
   ```

   如果命令傳回其他以 Organizations 為基礎的資源資料同步，則必須刪除該資源資料同步，或選擇不建立新的資源資料同步。

3. 執行以下命令來為 AWS Organizations中定義的帳戶建立資源資料同步。針對 amzn-s3-demo-bucket，指定您在本主題中稍早時候建立的 Amazon S3 儲存貯體名稱。如果您為儲存貯體建立了字首 (子目錄)，請針對 *prefix_name* 指定此資訊。

```
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. 針對您要將資料同步至中央 Amazon S3 儲存貯體的每個 AWS 區域 和 AWS 帳戶 ，重複步驟 2 和 3。 Amazon S3

管理資源資料同步

每個 AWS 帳戶 可以有 5 個資源資料同步 AWS 區域。您可以使用 AWS Systems ManagerFleet Manager 主控台來管理資源資料同步。

檢視資源資料同步

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 在帳戶管理下拉式選單中，選擇資源資料同步。
4. 從表格中選取一個資源資料同步，然後選擇檢視詳細資訊以檢視有關資源資料同步的資訊。

刪除資源資料同步

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 在帳戶管理下拉式選單中，選擇資源資料同步。
4. 從表格中選取一個資源資料同步，然後選擇刪除。

使用 EventBridge 監控庫存事件

您可以在 Amazon EventBridge 中設定規則以建立回應 AWS Systems Manager 庫存資源狀態變更的事件。EventBridge 支援下列庫存狀態變更的事件。盡可能傳送所有事件。

已刪除特定執行個體的自訂庫存類型：如果規則設定為監控此事件，則 EventBridge 會在刪除特定受管執行個體上的自訂庫存類型時建立事件。EventBridge 會針對每個自訂庫存類型的每個節點傳送一個事件。以下是範例事件模式。

```
{
    "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"
    }
}
```

已刪除所有執行個體的自訂庫存類型：如果規則設定為監控此事件，則 EventBridge 會在刪除所有受管節點上的自訂庫存類型時建立事件。以下是範例事件模式。

```
{
    "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 使用舊的結構描述版本事件進行呼叫：如果規則設定為監控此事件時，則 EventBridge 會在 PutInventory 進行呼叫時建立事件，使用低於目前結構描述之版本的結構描述版本。此事件適用於所有庫存類型。以下是範例事件模式。

```
{
    "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"
    }
}
```

如需如何將 EventBridge 設定為監控這些事件的詳細資訊，請參閱 為 Systems Manager 事件設定 EventBridge。

設定清查收集

本節說明如何使用 Systems Manager 主控台在一或多個受管節點上設定 AWS Systems Manager 庫存集合。如需如何使用 AWS Command Line Interface (AWS CLI) 設定庫存集合的範例，請參閱 使用 AWS CLI 設定庫存資料收集。

當您設定清查收集時，請先建立 AWS Systems Manager State Manager關聯。Systems Manager 會在執行關聯時收集庫存資料。如果您未先建立關聯，並嘗試使用 叫用`aws:softwareInventory`外掛程式 AWS Systems Manager Run Command，則系統會傳回下列錯誤： `The aws:softwareInventory plugin can only be invoked via ssm-associate.`

> **Note**
>
> 如果您為受管節點建立多個庫存關聯，請注意下列行為。
>
> - 可以為每個節點指派以所有節點為目標的庫存關聯 (--targets "Key=InstanceIds,Values=*")。
> - 每個節點也可以指派使用標籤鍵/值對或 AWS 資源群組的特定關聯。
> - 如果為節點指派多個庫存關聯，則尚未執行之關聯的狀態會顯示為略過。最近執行的關聯會顯示庫存關聯的實際狀態。
> - 如果為節點指派多個庫存關聯，且每個都使用標籤索引鍵/值對，則由於標籤衝突，這些庫存關聯無法在節點上執行。該關聯仍然在沒有標籤索引鍵/值衝突的節點上執行。

開始之前

設定庫存集合前，請務必完成以下任務。

- 更新您要清查 AWS Systems Manager SSM Agent的節點。透過執行最新版本的 SSM Agent，就能確保您可以收集所有受支援庫存類型的中繼資料。如需使用 SSM Agent 更新 State Manager 的相關資訊，請參閱 逐步解說：SSM Agent使用 自動更新 AWS CLI。
- 確認您已完成混合多雲端環境中 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體和非 EC2 機器的設定要求。如需相關資訊，請參閱設定 AWS Systems Manager的受管節點。
- 對於 Microsoft Windows 節點，請驗證您的受管節點是否已設定為使用 Windows PowerShell 3.0 (或更新版本)。SSM Agent 會使用 PowerShell 中的 `ConvertTo-Json` cmdlet，將 Windows 更新庫存資料轉換為所需的格式。
- (選用) 建立資源資料同步，將庫存資料集中存放在 Amazon S3 儲存貯體中。然後，資源資料同步會在系統收集新的清查資料時自動更新集中的資料。如需詳細資訊，請參閱演練：使用資源資料同步來彙總庫存資料。
- (選用) 建立 JSON 檔案以收集自訂庫存。如需詳細資訊，請參閱使用自訂庫存。

清查 中的所有受管節點 AWS 帳戶

您可以透過 AWS 帳戶 建立全域庫存關聯來清查 中的所有受管節點。全域庫存關聯會執行下列動作：

- 自動將全域庫存組態 （ 關聯 ） 套用至您 中的所有現有受管節點 AWS 帳戶。當系統套用並執行全域庫存關聯時，會略過已擁有庫存關聯的受管節點。一旦略過某個節點，詳細狀態資訊即會出現 `Overridden By Explicit Inventory Association` 訊息。儘管全域關聯會略過這類型的節點，但在執行指派的庫存關聯時，這些執行個體仍會回報庫存。

- 自動 AWS 帳戶 將 中建立的新節點新增至全域庫存關聯。

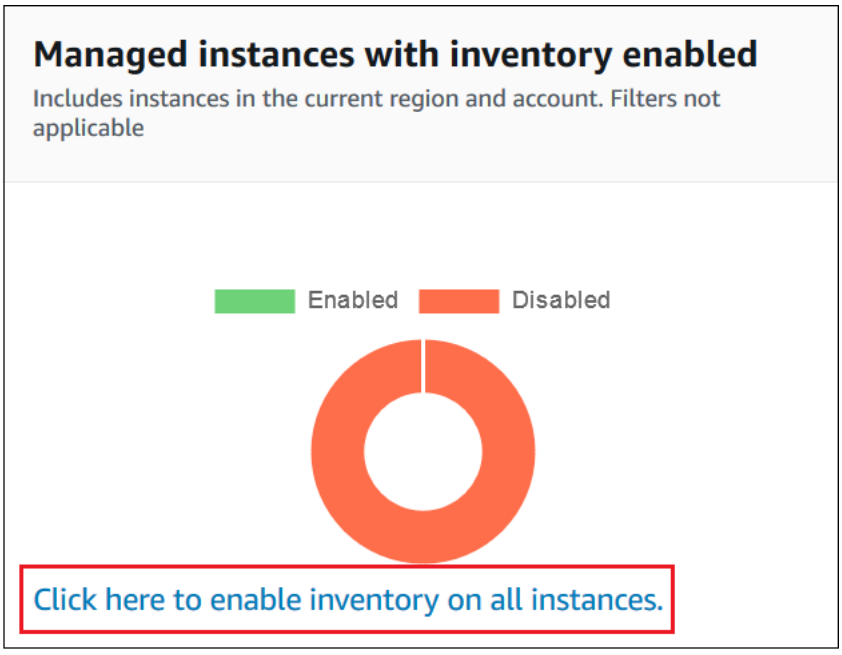
> **Note**
>
> - 如果全域庫存關聯已設定受管節點，且您指派了特定關聯給該節點，則 Systems Manager 庫存會降低全域關聯的優先順序，並套用特定關聯。
> - SSM Agent 2.0.790.0 版或更新版本皆能使用全域庫存關聯。如需如何在節點上更新 SSM Agent的相關資訊，請參閱 使用 Run Command 更新 SSM Agent。

按一下即可設定清查收集 (主控台)

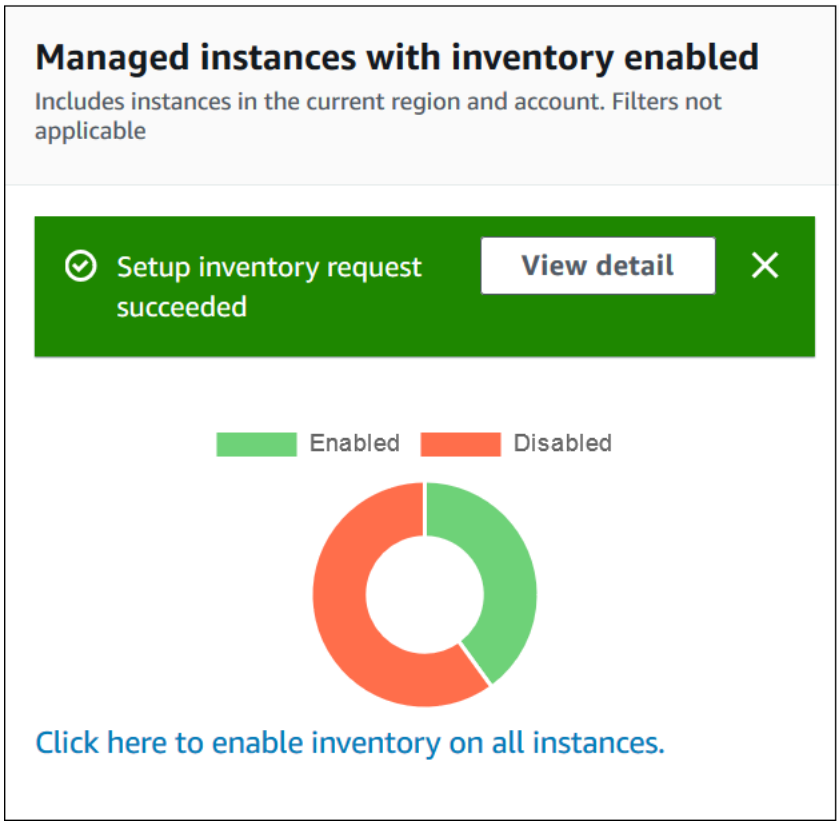
使用下列程序，在單一 AWS 帳戶 和 中設定所有受管節點的 Systems Manager 庫存 AWS 區域。

將目前區域中的所有受管節點設定為啟用 Systems Manager 庫存

1. 開啟 AWS Systems Manager 主控台，網址為 https：//https://console.aws.amazon.com/systems-manager/。
2. 在導覽窗格中，選擇庫存。
3. 在 Managed instances with inventory enabled (已啟用庫存的受管執行個體) 卡中，請選擇 Click here to enable inventory on all instances (按一下此處以啟用所有執行個體上的庫存)。



如果成功，主控台會顯示以下訊息。



根據您帳戶中的受管節點數量，需要幾分鐘的時間才能套用全域庫存關聯。請稍候幾分鐘，然後重新整理頁面。確認圖形的變更，以反映所有受管節點上設定的庫存。

使用主控台來設定收集

本節涵蓋的資訊將說明如何利用 Systems Manager 主控台來設定 Systems Manager 庫存，以收集來自受管節點的中繼資料。您可以快速收集特定 AWS 帳戶 （以及可能在該帳戶中建立的任何未來節點） 中所有節點的中繼資料，也可以使用標籤或節點 IDs 選擇性地收集庫存資料。

> **Note**
>
> 在完成此程序之前，請檢查全域庫存關聯是否存在。如果全域庫存關聯已經存在，則每當您啟動新執行個體時，系統都會套用該關聯，並清查此新執行個體。

設定庫存集合

1. 開啟 AWS Systems Manager 主控台，網址為 https：//https://console.aws.amazon.com/systems-manager/。

2. 在導覽窗格中，選擇庫存。
3. 選擇設定庫存。
4. 在 Targets (目標) 區段中，請選擇下列其中一個選項，藉此識別要執行這項操作的節點。

   - Selecting all managed instances in this account (選取此帳戶中的所有受管執行個體) - 此選項會選取沒有庫存關聯的所有受管節點。一旦選擇此選項，系統便會在收集庫存資料期間略過已擁有庫存關聯的節點，且庫存結果會顯示 Skipped (略過) 狀態。如需詳細資訊，請參閱清查 中的所有受管節點 AWS 帳戶。
   - Specifying a tag (指定標記) - 使用此選項指定單一標籤，以便在帳戶中識別要收集庫存的節點。在您使用標籤後，任何未來建立的節點若具備相同標籤，也都會回報庫存。如果現有庫存與所有節點相關聯，則使用標籤將特定節點選取為不同庫存的目標時，系統就會覆寫 All managed instances (所有受管執行個體) 目標群組中的節點成員資格。未來透過 All managed instances (所有受管執行個體) 收集庫存時，具備指定標籤的受管節點皆會遭略過。
   - Manually selecting instances (手動選取執行個體) - 使用此選項選擇帳戶中的特定受管節點。一旦透過此選項明確地選擇特定節點，系統將覆寫 All managed instances (所有受管執行個體) 目標上的庫存關聯。未來透過 All managed instances (所有受管執行個體) 收集庫存時，該節點便會遭略過。

   > **Note**
   >
   > 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

5. 在 Schedule (排程) 區段中，選擇系統從節點收集庫存中繼資料的頻率。
6. 在 Parameters (參數) 區段中，使用清單來開啟或關閉不同類型的庫存集合。如需收集檔案和 Windows 登錄檔庫存的詳細資訊，請參閱使用檔案與 Windows 登錄檔清查。
7. 如果您要將關聯執行狀態存放到 Amazon S3 儲存貯體，請在 Advanced (進階) 區段中選擇 Sync inventory execution logs to an Amazon S3 bucket (將清查執行日誌同步到 S3 儲存貯體)。
8. 選擇設定庫存。Systems Manager 會建立 State Manager 關聯，並立即在節點上執行庫存。
9. 在導覽窗格中，選擇 State Manager。接著，驗證系統是否使用 **`AWS-GatherSoftwareInventory`** 文件來建立新關聯。關聯排程會使用 Rate 運算式。此外，您還需確定 Status (狀態) 欄位已顯示 Success (成功)。如果選擇 Sync inventory execution logs to an Amazon S3 bucket (將庫存執行日誌同步到 Amazon S3 儲存貯體) 選項，則您可在幾分鐘後前往 Amazon S3 檢視日誌資料。在導覽窗格中選擇 Managed Instances (受管執行個體)，即可檢視特定節點的庫存資料。

10. 選擇一個節點，然後選擇 View details (檢視詳細資訊)。
11. 在節點詳細資訊頁面上，選擇 Inventory (庫存)。您能夠使用 Inventory type (庫存類型) 清單來篩選庫存。

查詢來自多個區域和帳戶的清查資料

AWS Systems Manager 庫存與 Amazon Athena 整合，以協助您查詢來自多個 AWS 區域 和 的庫存資料 AWS 帳戶。Athena 整合使用資源資料同步，因此您可以在 AWS Systems Manager 主控台中的詳細檢視頁面上檢視所有受管節點的庫存資料。

> **Important**
>
> 此功能會使用 AWS Glue 來編目 Amazon Simple Storage Service (Amazon S3) 儲存貯體中的資料，並使用 Amazon Athena 來查詢資料。視您所爬取和查詢的資料量而定，這些服務可能會向您收取使用費。使用 AWS Glue時，您需要為爬蟲程式 （ 探索資料 ） 和 ETL 任務 （ 處理和載入資料 ） 按秒計費的每小時費率。Athena 會按照每個查詢掃描的資料量向您收費。建議您先查看這些服務的定價準則，再使用 Amazon Athena 與 Systems Manager 庫存的整合。如需詳細資訊，請參閱 Amazon Athena 定價和 AWS Glue 定價。

您可以檢視所有 Amazon Athena 可用之 AWS 區域 中 Detailed View (詳細檢視) 頁面上的庫存資料。如需支援區域的清單，請參閱《Amazon Web Services 一般參考》中的 Amazon Athena 服務端點一節。

開始之前

Athena 整合能夠使用資源資料同步。您必須設定和配置資源資料同步，才能使用這項功能。如需詳細資訊，請參閱演練：使用資源資料同步來彙總庫存資料。

此外，請注意，針對資源資料同步使用的中央 Amazon S3 儲存貯體，Detailed View (詳細檢視) 頁面會顯示其擁有者的庫存資料。如果您不是中央 Amazon S3 儲存貯體的擁有者，則無法在 Detail View (詳細檢視) 頁面上查看庫存資料。

設定存取權

您必須先為 IAM 實體設定檢視資料的許可，才可以在 Systems Manager 主控台的詳細檢視頁面上查詢和檢視來自多個帳戶和區域的資料。

如果庫存資料存放在使用 AWS Key Management Service (AWS KMS) 加密的 Amazon S3 儲存貯體中，您也必須設定 IAM 實體和服務Amazon-GlueServiceRoleForSSM角色以進行 AWS KMS 加密。

主題

- 設定您的 IAM 實體以存取「詳細檢視」頁面
- （選用）設定檢視 AWS KMS 加密資料的許可

設定您的 IAM 實體以存取「詳細檢視」頁面

以下說明在詳細檢視頁面上檢視庫存資料所需的最低許可。

AWSQuicksightAthenaAccess 受管政策

以下是 PassRole 和其他所需的許可區塊

```
{
    "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::account_ID:role/*"
        },
        {
            "Sid": "iamPolicyCreation",
            "Effect": "Allow",
            "Action": "iam:CreatePolicy",
            "Resource": "arn:aws:iam::account_ID:policy/*"
        }
    ]
}
```

（選用）如果用於存放庫存資料的 Amazon S3 儲存貯體是使用 加密的 AWS KMS，您還必須將下列區塊新增至政策。

```
{
    "Effect": "Allow",
    "Action": [
        "kms:Decrypt"
    ],
    "Resource": [
        "arn:aws:kms:Region:account_ID:key/key_ARN"
    ]
}
```

若要提供存取權，請新增權限至您的使用者、群組或角色：

- 中的使用者和群組 AWS IAM Identity Center：

  建立權限合集。請按照 AWS IAM Identity Center 使用者指南 中的 建立權限合集 說明進行操作。

- 透過身分提供者在 IAM 中管理的使用者：

  建立聯合身分的角色。遵循「IAM 使用者指南」的為第三方身分提供者 (聯合) 建立角色中的指示。

- IAM 使用者：
  - 建立您的使用者可擔任的角色。請按照「IAM 使用者指南」的為 IAM 使用者建立角色中的指示。
  - (不建議) 將政策直接附加至使用者，或將使用者新增至使用者群組。請遵循 IAM 使用者指南的新增許可到使用者 (主控台) 中的指示。

（選用）設定檢視 AWS KMS 加密資料的許可

如果用於存放庫存資料的 Amazon S3 儲存貯體是使用 AWS Key Management Service (AWS KMS) 加密的，您必須使用 AWS KMS 金鑰的`kms:Decrypt`許可來設定 IAM 實體和 Amazon-GlueServiceRoleForSSM 角色。

開始之前

若要提供 AWS KMS 金鑰的`kms:Decrypt`許可，請將下列政策區塊新增至您的 IAM 實體：

```
{
    "Effect": "Allow",
    "Action": [
        "kms:Decrypt"
    ],
    "Resource": [
        "arn:aws:kms:Region:account_ID:key/key_ARN"
    ]
}
```

如果您尚未這麼做，請完成該程序並新增 AWS KMS 金鑰的`kms:Decrypt`許可。

使用下列程序來設定具有 AWS KMS 金鑰`kms:Decrypt`許可的 Amazon-GlueServiceRoleForSSM 角色。

設定具有 **`kms:Decrypt`** 許可的 Amazon-GlueServiceRoleForSSM 角色

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Roles (角色)，然後使用搜尋欄位來尋找 Amazon-GlueServiceRoleForSSM 角色。Summary (摘要) 頁面隨即開啟。
3. 使用搜尋欄位來尋找 Amazon-GlueServiceRoleForSSM 角色。選擇角色名稱 。Summary (摘要) 頁面隨即開啟。
4. 選擇角色名稱 。Summary (摘要) 頁面隨即開啟。

5. 選擇 Add inline policy (新增內嵌政策)。Create policy (建立政策) 頁面隨即開啟。
6. 選擇 JSON 標籤。
7. 刪除編輯器中的現有 JSON 文字，然後複製下列政策並貼至 JSON 編輯器。

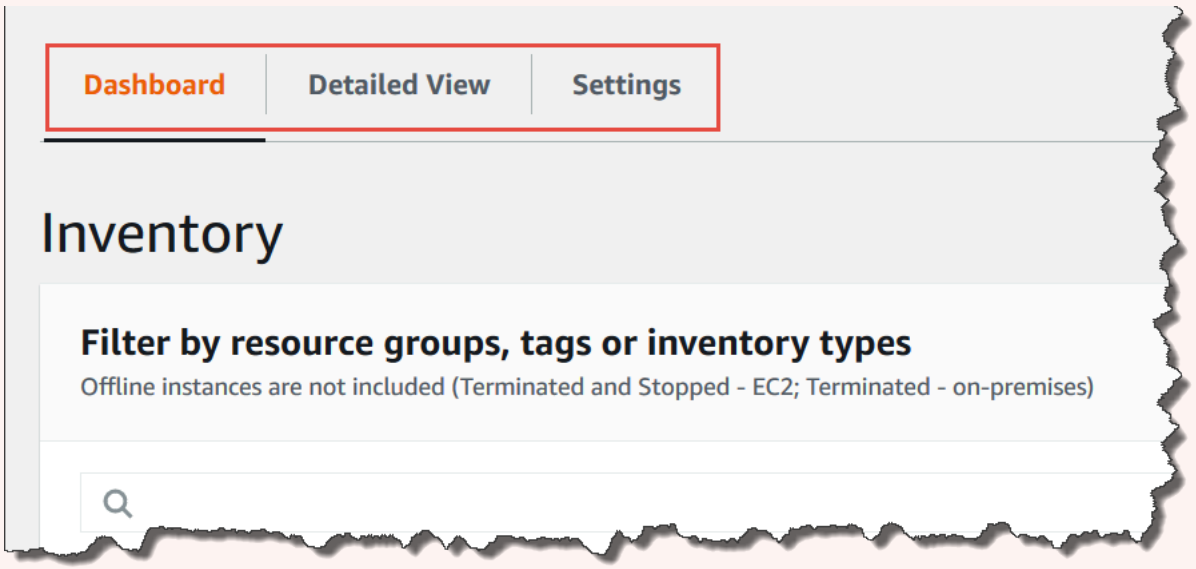
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:Region:account_ID:key/key_ARN"
            ]
        }
    ]
}
```

8. 選擇 Review policy (檢閱政策)
9. 在 Review Policy (檢閱政策) 頁面的 Name (名稱) 欄位中，輸入一個名稱。
10. 選擇建立政策。

在清查詳細檢視頁面上查詢資料

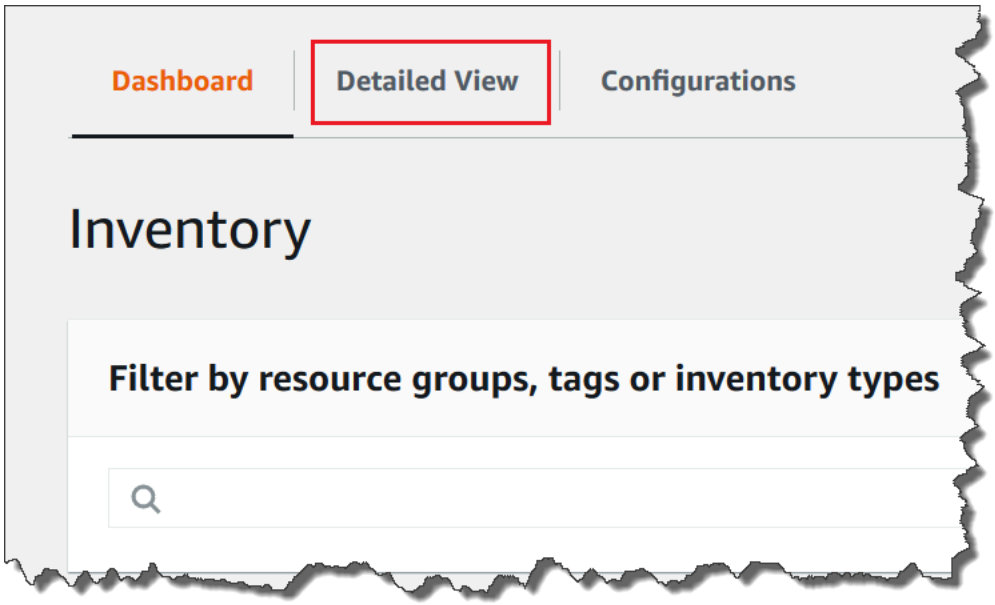
使用下列程序，在 Systems Manager 清查詳細檢視頁面上檢視來自多個 AWS 區域 和 AWS 帳戶 的清查資料。

> **Important**
>
> 庫存 Detailed View (詳細檢視) 頁面僅在提供 Amazon Athena 的 AWS 區域 中可用。若在 Systems Manager 庫存頁面上並未顯示下列標籤，這表示 Athena 並未在該區域中提供使用，且您無法使用 Detailed View (詳細檢視) 來查詢資料。

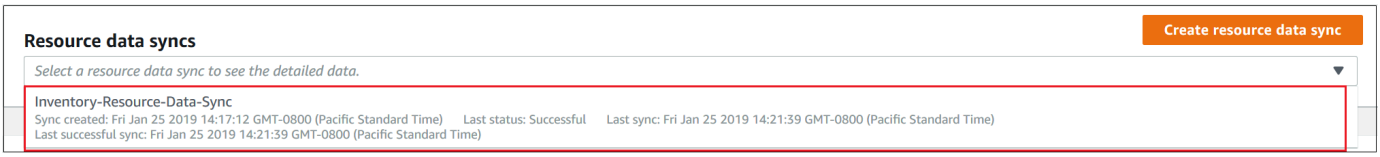


在 AWS Systems Manager 主控台中檢視來自多個區域及帳戶的清查資料

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇庫存。
3. 選擇 Detailed View (詳細檢視) 索引標籤。

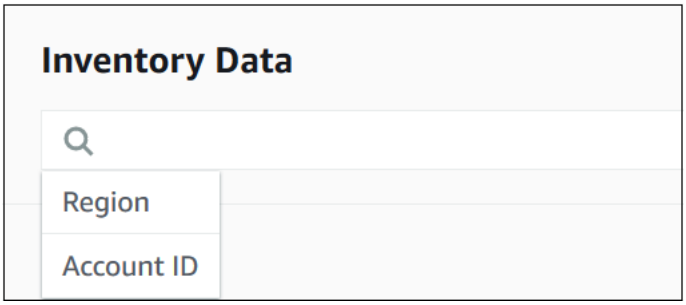


4. 選擇您要查詢資料的資源資料同步。



5. 在 Inventory Type (庫存類型) 清單中，選擇欲查詢的庫存資料類型，接著按 Enter 鍵。

6. 若要篩選資料，請選取篩選條件列，並選擇篩選條件選項。



您能夠善用 Export to CSV (匯出至 CSV) 按鈕，以便在 Microsoft Excel 等試算表應用程式中檢視目前的查詢集。除此之外，您還可以使用 Query History (查詢歷程記錄) 和 Run Advanced Queries (執行進階查詢) 按鈕，藉此檢視歷程記錄詳細資訊，並與 Amazon Athena 中的資料互動。

編輯 AWS Glue 爬蟲程式排程

AWS Glue 根據預設， 會每天編目中央 Amazon S3 儲存貯體中的庫存資料兩次。如果您經常變更節點上要收集的資料類型，可能會需要更頻繁地抓取資料，如以下程序所述。

> **Important**
>
> AWS Glue 對於爬蟲程式 （ 探索資料 ） 和 ETL 任務 （ 處理和載入資料 ） ， AWS 帳戶 會根據以秒計費的每小時費率向您的 收費。在您變更爬蟲程式排程前，請查看 AWS Glue 定價頁面。

變更庫存資料的爬蟲程式排程

1. 在 https：//https://console.aws.amazon.com/glue/ 開啟 AWS Glue 主控台。
2. 在導覽窗格中，選擇 Crawlers (爬蟲程式)。

3. 在爬蟲程式清單中，選擇 Systems Manager 庫存資料爬蟲程式旁的選項。爬蟲程式名稱需採用以下格式：

   `AWSSystemsManager-s3-bucket-name-Region-account_ID`

4. 選擇 Action (動作)，然後選擇 Edit crawler (編輯爬蟲程式)。
5. 在導覽窗格中，選擇 Schedule (排程)。
6. 在 Cron expression (Cron 運算式) 欄位中，使用 Cron 格式來指定新排程。如需 cron 格式的詳細資訊，請參閱《AWS Glue 開發人員指南》中的任務和爬蟲程式的時間排程。

> **Important**
>
> 您可以暫停爬蟲程式以停止產生費用 AWS Glue。若您暫停爬蟲程式或變更執行頻率，使系統減少爬取資料的次數，則 Detailed View (詳細檢視) 所顯示的資料可能不是目前的內容。

使用篩選條件查詢清查收集

收集清查資料後，您可以使用 中的篩選功能 AWS Systems Manager 來查詢符合特定篩選條件的受管節點清單。

根據庫存篩選條件查詢節點

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇庫存。
3. 在 Filter by resource groups, tags or inventory types (以資源群組、標記或庫存類型進行篩選) 區段中，選擇篩選條件方塊。螢幕上會顯示預先定義的篩選條件清單。
4. 選擇要篩選的屬性。例如，選擇 **`AWS:Application`**。如果出現提示，請選擇要篩選的次要屬性。例如，選擇 **`AWS:Application.Name`**。
5. 從清單中選擇分隔符號。舉例來說，您可以選擇 Begin with (開頭)。篩選條件中會顯示文字方塊。
6. 在文字方塊中輸入數值。例如，輸入 Amazon (SSM Agent 的名稱為 SSM Agent)。
7. 按 Enter 鍵。系統會傳回受管節點清單，其中包含開頭為 Amazon 字串的應用程式名稱。

> **Note**
>
> 您可以結合多個篩選條件，藉此縮小搜尋範圍。

彙總庫存資料

為 AWS Systems Manager 庫存設定受管節點後，您可以檢視庫存資料的彙總計數。例如，假設您配置了數十或數百個受管節點來收集 `AWS:Application` 庫存類型。透過本節涵蓋的資訊，您能夠查看經過設定用於收集此資料的節點確切計數。

您也可以彙整資料類型，以查看特定庫存詳細資訊。例如：`AWS:InstanceInformation` 庫存類型會收集包含 `Platform` 資料類型的作業系統平台資訊。只要彙總 `Platform` 資料類型的相關資料，您就能快速查看執行 Windows、執行 Linux 和執行 macOS 的節點數量。

本節中的程序說明如何使用 AWS Command Line Interface () 檢視庫存資料的彙總計數AWS CLI。您也可以在庫存頁面的 AWS Systems Manager 主控台中檢視預先設定的彙總計數。這些預先設定的儀表板通稱為「庫存詳情」。只需按一下，即可修正庫存組態問題。

檢視庫存資料彙整計數時，請注意以下重要詳細資訊：

- 如果您終止設定為收集庫存資料的受管節點，Systems Manager 會保留庫存資料 30 天，然後將其刪除。對於執行中的節點，系統會刪除超過 30 天的庫存資料。如果您需要存放庫存資料超過 30 天，您可以使用 AWS Config 記錄歷史記錄或定期查詢，並將資料上傳至 Amazon Simple Storage Service (Amazon S3) 儲存貯體。
- 如果您先前將節點設定成報告特定庫存資料類型 (如 `AWS:Network`)，之後為了停止收集該類型而變更組態，則彙整計數仍會顯示 `AWS:Network` 資料，直到節點終止且 30 天過去。

如需如何快速設定和收集特定 AWS 帳戶 （以及可能在該帳戶中建立的任何未來節點） 中所有節點的庫存資料的詳細資訊，請參閱 清查 中的所有受管節點 AWS 帳戶。

主題

- 彙總庫存資料以查看收集特定資料類型的節點計數
- 使用群組彙總庫存資料，以查看哪些節點已設定和未設定成收集某個庫存類型

彙總庫存資料以查看收集特定資料類型的節點計數

您可以使用 AWS Systems Manager GetInventory API 操作來檢視收集一或多個庫存類型和資料類型的節點彙總計數。例如，AWS:InstanceInformation 庫存類型可讓您將 GetInventory API 操作與 AWS:InstanceInformation.PlatformType 資料類型搭配使用，檢視作業系統的彙總。下方為 AWS CLI 命令和輸出的範例。

```
aws ssm get-inventory --aggregators "Expression=AWS:InstanceInformation.PlatformType"
```

系統會傳回相關資訊，如下所示。

```
{
   "Entities":[
      {
         "Data":{
            "AWS:InstanceInformation":{
               "Content":[
                  {
                     "Count":"7",
                     "PlatformType":"windows"
                  },
                  {
                     "Count":"5",
                     "PlatformType":"linux"
                  }
               ]
            }
         }
      }
   ]
}
```

開始使用

決定要檢視其計數的庫存類型和資料類型。如需檢視支援彙整功能的庫存類型和資料類型，請在 AWS CLI中執行以下命令。

```
aws ssm get-inventory-schema --aggregator
```

該命令會針對支援彙整功能的庫存類型和資料類型，傳回一份 JSON 清單。TypeName 欄位會顯示支援的庫存類型。Name (名稱) 欄位則會顯示每個資料類型。以以下清單為例，AWS:Application 庫存類型包含 Name 和 Version 的資料類型。

```
{
    "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"
                }
            ]
        }
    ]
}
```

使用以下語法建立命令，即可彙整任何列出的庫存類型資料。

```
aws ssm get-inventory --aggregators "Expression=InventoryType.DataType"
```

請見下方範例。

範例 1

此範例彙總了節點使用的 Windows 角色計數。

```
aws ssm get-inventory --aggregators "Expression=AWS:WindowsRole.Name"
```

範例 2

此範例彙總了節點上安裝的應用程式計數。

```
aws ssm get-inventory --aggregators "Expression=AWS:Application.Name"
```

結合多個彙總工具

您還能在一個命令中結合多個庫存類型與資料類型，有助於更全面了解相關資料。請見下方範例。

範例 1

此範例彙總了節點使用的作業系統類型計數。此命令亦可傳回特定的作業系統名稱。

```
aws ssm get-inventory --aggregators '[{"Expression":
 "AWS:InstanceInformation.PlatformType", "Aggregators":[{"Expression":
 "AWS:InstanceInformation.PlatformName"}]}]'
```

範例 2

此範例彙總了節點上執行的應用程式計數，以及每個應用程式的特定版本。

```
aws ssm get-inventory --aggregators '[{"Expression": "AWS:Application.Name",
 "Aggregators":[{"Expression": "AWS:Application.Version"}]}]'
```

如果您願意，還可以在 JSON 檔案中利用一個或多個庫存類型和資料類型來建立彙總表達式，並從 AWS CLI呼叫該檔案。檔案中的 JSON 物件必須採用以下語法。

```
[
       {
           "Expression": "string",
           "Aggregators": [
               {
                  "Expression": "string"
               }
           ]
       }
]
```

請務必使用 .json 副檔名儲存檔案。

以下是運用多個庫存類型和資料類型的範例。

```
[
       {
           "Expression": "AWS:Application.Name",
           "Aggregators": [
               {
                   "Expression": "AWS:Application.Version",
                   "Aggregators": [
                     {
                     "Expression": "AWS:InstanceInformation.PlatformType"
                     }
```

```
                    ]
                }
            ]
        }
]
```

使用下列命令，即可從 AWS CLI呼叫檔案。

```
aws ssm get-inventory --aggregators file://file_name.json
```

該命令會傳回相關資訊，如以下所示。

```
{"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"}
 ]
}
```

使用群組彙總庫存資料，以查看哪些節點已設定和未設定成收集某個庫存類型

Systems Manager 庫存中的群組可供您快速查看設定成收集一個或多個庫存類型的受管節點計數，以及未進行設定的執行個體數。透過群組功能，您可以指定一個或多個庫存類型，並使用 exists 運算子做為篩選條件。

例如，假設您分別設定四個受管節點，以收集下列庫存類型：

- 節點 1：AWS:Application
- 節點 2：AWS:File
- 節點 3：AWS:Application、AWS:File
- 節點 4：AWS:Network

您可以從 執行下列命令 AWS CLI ，以查看有多少節點設定為同時收集 AWS:Application和 AWS:File inventory類型。同時，回應還會傳回未設定成收集這兩種庫存類型的節點數。

```
aws ssm get-inventory --aggregators
 'Groups=[{Name=ApplicationAndFile,Filters=[{Key=TypeName,Values=[AWS:Application],Type=Exists}
{Key=TypeName,Values=[AWS:File],Type=Exists}]}]'
```

該命令回應顯示僅有一個受管節點經過設定，會同時收集 AWS:Application 與 AWS:File 庫存類型。

```
{
   "Entities":[
      {
         "Data":{
            "ApplicationAndFile":{
               "Content":[
                  {
                     "notMatchingCount":"3"
                  },
                  {
```

```
                    "matchingCount":"1"
                }
            ]
        }
    }
  ]
}
```

> **Note**
>
> 群組不會傳回資料類型計數。您也無法深入檢視結果，以查看已設定或未設定成收集庫存類型的節點 ID。

如果您願意，還可以在 JSON 檔案中利用一或多個庫存類型來建立彙整表達式，並從 AWS CLI呼叫該檔案。檔案中的 JSON 物件必須採用以下語法：

```
{
   "Aggregators":[
      {
         "Groups":[
            {
               "Name":"Name",
               "Filters":[
                  {
                     "Key":"TypeName",
                     "Values":[
                        "Inventory_type"
                     ],
                     "Type":"Exists"
                  },
                  {
                     "Key":"TypeName",
                     "Values":[
                        "Inventory_type"
                     ],
                     "Type":"Exists"
                  }
               ]
            }
         ]
```

```
      }
   ]
}
```

請務必使用 .json 副檔名儲存檔案。

使用下列命令，即可從 AWS CLI呼叫檔案。

```
aws ssm get-inventory --cli-input-json file://file_name.json
```

其他範例

下列範例會示範彙整庫存資料的方式，使您能查看設定成收集特定庫存類型的受管節點，以及未設定的受管節點。這些範例皆是採用 AWS CLI。每個範例均包含具備篩選功能的完整命令。如此一來，當您想在檔案中輸入資訊時，便能從命令列和範例 input.json 檔案中執行該命令。

範例 1

此範例會彙整設定成收集 AWS:Application 或 AWS:File 庫存類型的節點計數，以及未經過設定的執行個體數。

從 AWS CLI執行下列命令。

```
aws ssm get-inventory --aggregators
 'Groups=[{Name=ApplicationORFile,Filters=[{Key=TypeName,Values=[AWS:Application,
 AWS:File],Type=Exists}]}]'
```

若您偏好使用檔案，則可複製下方範例並貼至檔案，接著將其儲存為 input.json。

```
{
   "Aggregators":[
      {
         "Groups":[
            {
               "Name":"ApplicationORFile",
               "Filters":[
                  {
                     "Key":"TypeName",
                     "Values":[
                        "AWS:Application",
```

```
                        "AWS:File"
                     ],
                     "Type":"Exists"
                  }
               ]
            }
         ]
      }
   ]
}
```

從 AWS CLI執行下列命令。

```
aws ssm get-inventory --cli-input-json file://input.json
```

該命令會傳回相關資訊，如以下所示。

```
{
   "Entities":[
      {
         "Data":{
            "ApplicationORFile":{
               "Content":[
                  {
                     "notMatchingCount":"1"
                  },
                  {
                     "matchingCount":"3"
                  }
               ]
            }
         }
      }
   ]
}
```

範例 2

此範例會彙總設定成收集 AWS:Application、AWS:File 或 AWS:Network 庫存類型的節點計數，以及未經過設定的節點數。

從 AWS CLI執行下列命令。

```
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}]}]'
```

若您偏好使用檔案，則可複製下方範例並貼至檔案，接著將其儲存為 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"
                }
            ]
        }
    ]
  }
]
}
```

從 AWS CLI執行下列命令。

```
aws ssm get-inventory --cli-input-json file://input.json
```

該命令會傳回相關資訊，如以下所示。

```
{
   "Entities":[
      {
         "Data":{
            "Application":{
               "Content":[
                  {
                     "notMatchingCount":"2"
                  },
                  {
                     "matchingCount":"2"
                  }
               ]
            },
            "File":{
               "Content":[
                  {
                     "notMatchingCount":"2"
                  },
                  {
                     "matchingCount":"2"
                  }
               ]
            },
            "Network":{
               "Content":[
                  {
                     "notMatchingCount":"3"
```

```
                },
                {
                   "matchingCount":"1"
                }
             ]
          }
       }
    }
 ]
}
```

使用自訂庫存

您可以透過建立 AWS Systems Manager 庫存自訂庫存，將您想要的任何中繼資料指派給節點。舉例來說，假設您負責管理資料中心內的大量伺服器，而這些伺服器皆已設定為 Systems Manager 受管節點。目前，您都是將伺服器機架位置的相關資訊存放在試算表中。透過自訂庫存，即可在節點上將每個節點的機架位置指定為中繼資料。當您使用 Systems Manager 收集庫存時，系統將一併收集中繼資料和其他庫存中繼資料。然後，您可以透過使用資源資料同步將所有庫存中繼資料連接到中央 Amazon S3 儲存貯體，並查詢資料。

> **Note**
>
> Systems Manager 支援每個 AWS 帳戶最多 20 個自訂庫存類型。

若要指派自訂庫存給節點，則可使用 Systems Manager PutInventory API 操作，如中 將自訂庫存中繼資料指派給受管節點 所述。或者，您也能選擇建立自訂庫存 JSON 檔案，並將其上傳至節點。本節將說明如何建立 JSON 檔案。

以下範例 JSON 檔案與自訂庫存會指定與現場部署伺服器相關的機架資訊。此範例會指定一種的自訂庫存資料 (`"TypeName": "Custom:RackInformation"`)，並在 `Content` 下提供多個描述資料的項目。

```
{
    "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 區段中指定不同項目，如下範例所示。

```
{
"SchemaVersion": "1.0",
"TypeName": "Custom:PuppetModuleInfo",
    "Content": [{
        "Name": "puppetlabs/aws",
        "Version": "1.0"
      },
      {
        "Name": "puppetlabs/dsc",
        "Version": "2.0"
      }
    ]
}
```

自訂庫存的 JSON 結構描述需要 SchemaVersion、TypeName 和 Content 區段，而您可以自行定義這些區段的資訊。

```
{
    "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 的值限制為 100 個字元。此外，該 TypeName 值必須以大寫字母 Custom 開頭。例如：Custom:PuppetModuleInfo。因此，下列範例會導致例外狀況：CUSTOM:PuppetModuleInfo、custom:PuppetModuleInfo。

Content 區段包含屬性和##。這些項目不需區分大小寫。然而，若您有定義屬性 (如 "Vendor": "DELL")，自訂庫存檔案便必須持續參考此屬性。換而言之，如果您在某個檔案中指定 "Vendor":

"DELL" (vendor 使用大寫「V」)，卻在另一個檔案中指定 "vendor": "DELL" (vendor 使用小寫「v」)，系統就會傳回錯誤。

> **Note**
>
> 您必須以 .json 副檔名儲存檔案，且您定義的清查必須只包含字串值。

建立檔案之後，請務必將其儲存在節點上。下表顯示自訂庫存 JSON 檔案在節點上的存放位置：

作業系統	路徑
Linux	/var/lib/amazon/ssm/*node-id*/inventory/custom
macOS	/opt/aws/ssm/data/ *node-id*/ inventory/custom
Windows	%SystemDrive%\ProgramData\Amazon\SSM \InstanceData*node-id*\inventory\custom

如需使用自訂庫存的範例，請參閱部落格文章 Get Disk Utilization of Your Fleet Using EC2 Systems Manager Custom Inventory Types。

刪除自訂清查

透過 DeleteInventory API 操作，即可刪除自訂庫存類型，以及與該類型相關聯的資料。您可以使用 AWS Command Line Interface (AWS CLI) 來呼叫 delete-inventory 命令，進而刪除庫存類型的所有資料。此外，您也能利用 SchemaDeleteOption 來呼叫 delete-inventory 命令，藉此刪除自訂庫存類型。

> **Note**
>
> 庫存類型也稱為庫存結構描述。

SchemaDeleteOption 參數包含下列選項：

- DeleteSchema：此選項可刪除指定的自訂類型，以及與其相關聯的所有資料。您稍後可以視需求重新建立該結構描述。

- DisableSchema：如果您選擇此選項，系統即會關閉目前版本，然後刪除所有相關資料。當版本低於或等於遭關閉版本時，系統會忽略所有新資料。在版本比遭關閉版本更新的情況下，您能夠呼叫 PutInventory 動作，重新啟用這個庫存類型。

使用 刪除或關閉自訂庫存 AWS CLI

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

2. 執行下列命令，即可利用 `dry-run` 選項來查看系統要刪除的資料。此命令不會刪除任何資料。

   ```
   aws ssm delete-inventory --type-name "Custom:custom_type_name" --dry-run
   ```

   系統會傳回相關資訊，如下所示。

   ```
   {
      "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"
   }
   ```

   如需了解如何刪除庫存摘要的相關資訊，請參閱了解刪除清查摘要。

3. 執行下列命令，藉此刪除自訂庫存類型的所有資料。

   ```
   aws ssm delete-inventory --type-name "Custom:custom_type_name"
   ```

> **Note**
>
> 此命令輸出並不會顯示刪除進度。因此，TotalCount 與 Remaining Count 的結果都是相同的，因為系統尚未刪除任何項目。您能使用 describe-inventory-deletions 命令來顯示刪除進度，本主題稍後會予以說明。

系統會傳回相關資訊，如下所示。

```
{
   "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"
}
```

系統將從 Systems Manager 庫存服務刪除指定自訂庫存類型的所有資料。

4. 執行下列命令。該命令會針對目前的庫存類型版本執行以下動作：關閉目前版本，然後刪除所有相關資料。當版本低於或等於遭關閉版本時，系統會忽略所有新資料。

```
aws ssm delete-inventory --type-name "Custom:custom_type_name" --schema-delete-
option "DisableSchema"
```

系統會傳回相關資訊，如下所示。

```
{
   "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"
}
```

若要檢視遭關閉的庫存類型，則可使用下列命令。

```
aws ssm get-inventory-schema --type-name Custom:custom_type_name
```

5. 執行下列命令，以便刪除庫存類型。

```
aws ssm delete-inventory --type-name "Custom:custom_type_name" --schema-delete-
option "DeleteSchema"
```

系統會刪除指定自訂類型的結構描述及所有庫存資料。

系統會傳回相關資訊，如下所示。

```
{
   "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"
}
```

檢視刪除狀態

您可以使用 `describe-inventory-deletions` AWS CLI 命令來檢查刪除操作的狀態。若要檢視特定刪除操作的狀態，請指定刪除 ID。或者，您能夠省略刪除 ID，以檢視過去 30 天內執行的所有刪除操作清單。

1. 執行下列命令，藉此檢視刪除操作狀態。系統將在 delete-inventory 摘要中傳回刪除 ID。

   ```
   aws ssm describe-inventory-deletions --deletion-id system_generated_deletion_ID
   ```

   系統會傳回最新狀態。刪除操作可能尚未完成。系統會傳回相關資訊，如下所示。

   ```
   {"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"}
  ]
}
```

如果刪除操作成功，LastStatusMessage 會顯示以下狀態：Deletion is successful (刪除成功)。

```
{"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. 執行下列命令，即可檢視過去 30 天內執行的所有刪除操作清單。

```
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"
```

了解刪除清查摘要

為了協助您更充分了解庫存摘要刪除操作的內容，請參考以下範例。使用者將 Custom:RackSpace 庫存指派給三個節點。庫存項目 1 和 2 皆使用自訂類型 1.0 版 ("SchemaVersion":"1.0")。另一方面，庫存項目 3 則是使用自訂類型 2.0 版 ("SchemaVersion":"2.0")。

RackSpace 自訂庫存 1

```
{
   "CaptureTime":"2018-02-19T10:48:55Z",
   "TypeName":"CustomType:RackSpace",
   "InstanceId":"i-1234567890",
   "SchemaVersion":"1.0"   "Content":[
      {
         content of custom type omitted
      }
   ]
}
```

RackSpace 自訂庫存 2

```
{
   "CaptureTime":"2018-02-19T10:48:55Z",
   "TypeName":"CustomType:RackSpace",
   "InstanceId":"i-1234567891",
   "SchemaVersion":"1.0"   "Content":[
      {
         content of custom type omitted
      }
   ]
}
```

RackSpace 自訂庫存 3

```
{
   "CaptureTime":"2018-02-19T10:48:55Z",
   "TypeName":"CustomType:RackSpace",
   "InstanceId":"i-1234567892",
   "SchemaVersion":"2.0"   "Content":[
      {
         content of custom type omitted
```

```
      }
   ]
}
```

使用者可執行下列命令，以預覽即將刪除的資料。

```
aws ssm delete-inventory --type-name "Custom:RackSpace" --dry-run
```

系統會傳回相關資訊，如下所示。

```
{
   "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"
}
```

使用者可執行下列命令，以刪除 Custom:RackSpace 庫存。

> **Note**
>
> 此命令輸出並不會顯示刪除進度。因此，TotalCount 與 RemainingCount 都是相同的，因為系統尚未刪除任何項目。您可以使用 describe-inventory-deletions 命令來顯示刪除進度。

```
aws ssm delete-inventory --type-name "Custom:RackSpace"
```

系統會傳回相關資訊，如下所示。

```
{
   "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 中檢視清查刪除操作

您可以設定 Amazon EventBridge，在每次使用者刪除自訂庫存時建立事件。EventBridge 提供了三種事件類型，皆適用於自訂庫存刪除操作：

- 執行個體的刪除動作：指出特定受管節點的自訂庫存是否已成功刪除。
- 刪除動作摘要：刪除動作的摘要。

- 關閉自訂庫存類型的警告：當使用者對先前關閉的自訂庫存類型版本呼叫 PutInventory API 操作時，就會出現這個警告事件。

下方是每個事件的範例。

執行個體的刪除動作

```
{
   "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"
   }
}
```

刪除動作摘要

```
{
   "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"
   }
}
```

關閉自訂庫存類型的警告

```
{
   "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"
   }
}
```

遵循以下處理程序，即可建立適用自訂庫存刪除操作的 EventBridge 規則。這項處理程序會說明如何建立規則，使其可傳送自訂庫存刪除操作通知至 Amazon SNS 主題。開始操作前，請確認您擁有 Amazon SNS 主題，或是建立一個新的主題。如需詳細資訊，請參閱 Amazon Simple Notification Service 開發人員指南中的入門。

設定適用庫存刪除操作的 EventBridge

1. 前往 https://console.aws.amazon.com/events/ 開啟 Amazon EventBridge 主控台。
2. 在導覽窗格中，選擇規則。
3. 選擇建立規則。
4. 輸入規則的名稱和描述。

   在同一個區域和同一個事件匯流排上，規則不能與另一個規則同名。
5. 針對事件匯流排，選擇要與此規則建立關聯的事件匯流排。如果您希望此規則回應來自您自己的相符事件 AWS 帳戶，請選取預設值。當您 AWS 服務 帳戶中的 發出事件時，一律會傳送至您帳戶的預設事件匯流排。
6. 針對規則類型，選擇具有事件模式的規則。
7. 選擇下一步。
8. 在事件來源欄位中，選擇 AWS 事件或 EventBridge 合作夥伴事件。
9. 在 Event pattern (事件模式) 區段中，選擇 Event pattern form (事件模式表單)。
10. 在事件來源欄位中，選擇 AWS 服務。
11. 針對 AWS service (服務)，請選擇 Systems Manager。
12. 在 Event type (事件類型) 中，選擇 Inventory (庫存)。
13. 針對 Specific detail type(s) (特定詳細資訊類型)，請選擇 Inventory Resource State Change (庫存資源狀態變更)。
14. 選擇 Next (下一步)。
15. 在目標類型欄位中，選擇 AWS 服務。
16. 針對 Select a target (選取目標)，選擇 SNS topic (SNS 主題)，然後針對 Topic (主題)，選擇您的主題。
17. 在 Additional settings (其他設定) 區段中，針對 Configure target input (設定目標輸入)，確認已選取 Matched event (相符的事件)。
18. 選擇 Next (下一步)。
19. (選用) 為規則輸入一或多個標籤。如需詳細資訊，請參閱《Amazon EventBridge 使用者指南》中的標記您的 Amazon EventBridge 資源。
20. 選擇 Next (下一步)。
21. 檢閱規則的詳細資訊，然後選擇建立規則。

檢視清查歷程記錄和變更追蹤

您可以使用 來檢視所有受管節點的 AWS Systems Manager 庫存歷史記錄和變更追蹤AWS Config。AWS Config 提供 AWS 資源組態的詳細檢視 AWS 帳戶。這包含資源彼此之間的關係和之前的組態方式，所以您可以看到一段時間中組態和關係的變化。如要檢視清查歷史記錄和變更追蹤，您必須在AWS Config中開啟下列資源：

- SSM:ManagedInstanceInventory
- SSM:PatchCompliance
- SSM:AssociationCompliance
- SSM:FileData

> **Note**
>
> 請注意以下有關庫存歷史記錄和變更追蹤的重要詳細資訊：
>
> - 如果您使用 AWS Config 來追蹤系統中的變更，則必須設定 Systems Manager 庫存來收集`AWS:File`中繼資料，以便檢視 AWS Config () 中的檔案變更`SSM:FileData`。如果未如此設定，則 AWS Config 不會追蹤系統上的檔案變更。
> - 一旦開啟 SSM:PatchCompliance 與 SSM:AssociationCompliance，您就能檢視 Systems Manager Patch Manager 修補和 Systems Manager State Manager 關聯合規歷史記錄及變更追蹤。如需這些資源的合規管理詳細資訊，請參閱了解有關合規性的詳細資訊。

下列程序說明如何 AWS Config 使用 AWS Command Line Interface () 在 中開啟庫存歷史記錄和變更追蹤記錄AWS CLI。如需如何在 中選擇和設定這些資源的詳細資訊 AWS Config，請參閱《 AWS Config 開發人員指南》中的選取哪些資源 AWS Config 記錄。如需 AWS Config 定價的資訊，請參閱定價。

開始之前

AWS Config require AWS Identity and Access Management (IAM) 取得 Systems Manager 資源組態詳細資訊的許可。在下列程序中，您必須為授予 Systems Manager 資源 AWS Config 許可的 IAM 角色指定 Amazon Resource Name (ARN)。您能夠將 `AWS_ConfigRole` 受管政策連接至指派給 AWS Config的 IAM 角色。如需此角色的詳細資訊，請參閱《 AWS Config 開發人員指南》中的 AWS 受管政策： AWS_ConfigRole。如需如何建立 IAM 角色和指派 `AWS_ConfigRole` 受管政策給該角色的詳細資訊，請參閱 IAM 使用者指南中的建立角色以將許可指派給 AWS 服務。

在 中開啟庫存歷史記錄和變更追蹤記錄 AWS Config

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

2. 複製下列 JSON 範例並貼至簡單的文字檔案，然後將其儲存為 recordingGroup.json。

```
{
   "allSupported":false,
   "includeGlobalResourceTypes":false,
   "resourceTypes":[
      "AWS::SSM::AssociationCompliance",
      "AWS::SSM::PatchCompliance",
      "AWS::SSM::ManagedInstanceInventory",
      "AWS::SSM::FileData"
   ]
}
```

3. 請執行下列命令，藉此將 recordingGroup.json 檔案載入至 AWS Config。

```
aws configservice put-configuration-recorder --configuration-recorder
 name=myRecorder,roleARN=arn:aws:iam::123456789012:role/myConfigRole --recording-
group file://recordingGroup.json
```

4. 執行以下命令，即可開始記錄清查歷史記錄和變更追蹤。

```
aws configservice start-configuration-recorder --configuration-recorder-
name myRecorder
```

設定歷史記錄和變更追蹤之後，您可以透過選擇 Systems Manager 主控台中的 AWS Config 按鈕向下切入至特定受管節點的歷史記錄。您可以從 Managed Instances (受管執行個體) 頁面或 Inventory (庫存) 頁面存取 AWS Config 按鈕。視螢幕大小而定，您可能需要捲動至頁面右側，才能看見該按鈕。

停用資料收集和刪除庫存資料

如果您不想再使用 AWS Systems Manager 庫存來檢視資源 AWS 的中繼資料，您可以停止資料收集並刪除已收集的資料。此區段包含下列資訊：

主題

- 停用資料收集

- 刪除庫存資源資料同步

停用資料收集

當您最初設定 Systems Manager 來收集庫存資料時，系統會建立 State Manager 關聯，定義排程和要從中收集中繼資料的資源。您可以透過刪除任何使用 `AWS-GatherSoftwareInventory` 文件的 State Manager 關聯，停用資料收集。

刪除庫存關聯

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 State Manager。
3. 選擇使用 `AWS-GatherSoftwareInventory` 文件的關聯，然後選擇 Delete (刪除)。
4. 對任何使用 `AWS-GatherSoftwareInventory` 文件的剩餘關聯重複步驟 3。

刪除庫存資源資料同步

如果您不想再使用 AWS Systems Manager 庫存來檢視資源 AWS 的中繼資料，我們也建議您刪除用於庫存資料收集的資源資料同步。

刪除庫存資源資料同步

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇庫存。
3. 選擇 Resource Data Syncs (資源資料同步)。
4. 在清單中選擇同步。

> **Important**
>
> 請確定您選擇了用於庫存的同步。Systems Manager 支援多個工具的資源資料同步。如果選擇了錯誤的同步，則您可能會中斷 Systems Manager Explorer 或 Systems Manager Compliance 的資料彙總。

5. 選擇 Delete (刪除)
6. 針對您要刪除的任何剩餘資源資料同步重複這些步驟。

7. 刪除存放資料的 Amazon Simple Storage Service (Amazon S3) 儲存貯體。如需刪除 Amazon S3 儲存貯體的資訊，請參閱刪除儲存貯體。

將自訂庫存中繼資料指派給受管節點

下列程序會逐步引導您使用 AWS Systems Manager PutInventory API 操作，將自訂庫存中繼資料指派給受管節點。此範例會將機架位置資訊指派給節點。如需自訂清查的詳細資訊，請參閱 使用自訂庫存。

指派自訂庫存中繼資料給節點

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。
2. 請執行下列命令，將機架位置資訊指派給節點。

   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. 執行下列命令，即可檢視此節點的自訂庫存項目。

   ```
   aws ssm list-inventory-entries --instance-id ID --type-name "Custom:RackInfo"
   ```

   系統會回應相關資訊，如下所示。

   ```
   {
       "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. 執行以下命令來檢視自訂清查結構描述。

```
aws ssm get-inventory-schema --type-name Custom:RackInfo
```

系統會回應相關資訊，如下所示。

```
{
    "Schemas": [
        {
            "TypeName": "Custom:RackInfo",
            "Version": "1.0",
            "Attributes": [
                {
                    "DataType": "STRING",
                    "Name": "RackLocation"
                }
            ]
        }
    ]
}
```

使用 AWS CLI 設定庫存資料收集

以下程序會逐步引導您將 AWS Systems Manager 庫存設定為從受管節點中收集中繼資料的程序。設定庫存集合時，首先要建立 Systems Manager State Manager 關聯。Systems Manager 會在執行關聯時收集庫存資料。如果沒有先建立關聯，則當您試圖使用 Systems Manager Run Command 等呼叫 `aws:softwareInventory` 外掛程式時，系統即會傳回以下錯誤：

`The aws:softwareInventory plugin can only be invoked via ssm-associate.`

> **Note**
>
> 一個節點一次只能設定一個庫存關聯。若您為節點設定兩個以上的庫存關聯，關聯便不會執行，並且也不會收集任何庫存資料。

快速將所有受管節點設定為啟用庫存 (CLI)

您可以快速設定 AWS 帳戶 和目前區域中的所有受管節點，以收集庫存資料。這是過程稱為全域庫存關聯的建立。若要使用 建立全域庫存關聯 AWS CLI，請使用 `instanceIds`值的萬用字元選項，如下列程序所示。

為目前區域 (CLI) 中 AWS 帳戶 和 中的所有受管節點設定庫存

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。
2. 執行下列命令。

   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**
>
> 此命令不允許庫存收集 Windows 登錄檔或檔案的中繼資料。若要庫存這些資料類型，請使用下一個程序。

在受管節點上手動設定庫存 (CLI)

使用下列程序，使用節點 IDs或標籤，在受管節點上手動設定 AWS Systems Manager 庫存。

將受管節點手動設定為供庫存使用 (CLI)

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

2. 執行下列命令，以建立能在節點上執行 Systems Manager 庫存的 State Manager 關聯。將每個*########*取代為您自己的資訊。此命令會將該服務設定為每六小時執行一次，並收集來自節點的網路組態、Windows 更新與應用程式中繼資料。

   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"
   ```

   系統會回應相關資訊，如下所示。

   ```
   {
       "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"
            }
        ]
    }
}
```

透過 Targets 參數，即可使用 EC2 標記來鎖定大型目標節點群組。請參閱以下範例。

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"
```

您也可以在運算式中使用 `files` 和 `windowsRegistry` 庫存類型，以庫存 Windows Server 節點上的檔案和 Windows 登錄機碼。如需這些庫存類型的詳細資訊，請參閱[使用檔案與 Windows 登錄檔清查](#)。

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. 執行下列命令，以檢視關聯狀態。

```
aws ssm describe-instance-associations-status --instance-id an_instance_ID
```

系統會回應相關資訊，如下所示。

```
{
"InstanceAssociationStatusInfos": [
         {
            "Status": "Pending",
            "DetailedStatus": "Associated",
            "Name": "reInvent2016PolicyDocumentTest",
            "InstanceId": "i-1a2b3c4d5e6f7g",
            "AssociationId": "1a2b3c4d5e6f7g-1a2b3c-1a2b3c-1a2b3c-1a2b3c4d5e6f7g",
            "DocumentVersion": "1"
        }
]
}
```

演練：使用資源資料同步來彙總庫存資料

下列逐步解說說明如何使用 AWS Command Line Interface () 為 AWS Systems Manager 庫存建立資源資料同步組態AWS CLI。資源資料同步會自動將所有受管節點的庫存資料移至中央 Amazon Simple Storage Service (Amazon S3) 儲存貯體。每當搜索到新的庫存資料時，同步就會自動更新中央 Amazon S3 儲存貯體內的資料。

同時，此演練也會介紹如何使用 Amazon Athena 和 Amazon QuickSight 來查詢彙總資料，並予以分析。如需在 中使用 Systems Manager 建立資源資料同步的詳細資訊 AWS Management Console，請參閱 演練：使用資源資料同步來彙總庫存資料。如需有關使用 中的 Systems Manager 從多個 AWS 區域 和 帳戶查詢清查的資訊 AWS Management Console，請參閱 查詢來自多個區域和帳戶的清查資料。

> **Note**
>
> 此逐步教學涵蓋的資訊能協助您使用 AWS Key Management Service (AWS KMS) 加密同步作業。庫存並不會收集任何使用者專用、私有或敏感資料，因此加密為選用功能。如需 的詳細資訊 AWS KMS，請參閱 AWS Key Management Service 開發人員指南。

開始之前

在開始本節中的演練之前，請先檢閱或完成以下任務：

- 收集受管節點的庫存資料。為了完成此演練中的 Amazon Athena 和 Amazon QuickSight 章節，建議您收集應用程式資料。如需如何收集庫存資料的詳細資訊，請參閱 設定清查收集 或 使用 AWS CLI 設定庫存資料收集。
- （選用）如果庫存資料存放在使用 AWS Key Management Service () 加密的 Amazon Simple Storage Service (Amazon S3 AWS KMS) 儲存貯體中，您也必須設定 IAM 帳戶和服務`Amazon-GlueServiceRoleForSSM`角色以進行 AWS KMS 加密。如果您未設定 IAM 帳戶和此角色，Systems Manager 會在您選擇主控台上的 Detailed View (詳細檢視) 索引標籤時顯示 `Cannot load Glue tables`。如需詳細資訊，請參閱（選用）設定檢視 AWS KMS 加密資料的許可。
- （選用）如果您想要使用 加密資源資料同步 AWS KMS，則必須建立新的金鑰，其中包含下列政策，或者您必須更新現有的金鑰並將此政策新增至其中。

```
{
    "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/
*"
                }
            }
        }
    ]
}
```

建立庫存的資源資料同步

1. 開啟位於 https://console.aws.amazon.com/s3/ 的 Amazon S3 主控台。
2. 建立儲存貯體以存放您彙整的清查資料。如需詳細資訊，請參閱 Amazon Simple Storage Service 主控台使用者指南中的建立儲存貯體。請記下儲存貯體名稱和您建立儲存貯體 AWS 區域 所在的 。
3. 建立儲存貯體後，請選擇 Permissions (許可) 索引標籤，接著選擇 Bucket Policy (儲存貯體政策)。
4. 複製下列儲存貯體政策並貼至政策編輯器。將 amzn-s3-demo-bucket 和 *account-id* 取代為您建立的 Amazon S3 儲存貯體名稱和有效的 AWS 帳戶 ID。新增多個帳戶時，請為每個帳戶新增額外的條件字串和 ARN。新增帳戶時，請移除範例中額外的預留位置。或者，您也可以使用 Amazon S3 字首 (子目錄) 取代 *bucket-prefix*。如果您沒有建立字首，請將 *bucket-prefix/* 從政策中的 ARN 移除。

```
{
  "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:*:account-id1:resource-data-sync/*",
            "arn:aws:ssm:*:account-id2:resource-data-sync/*",
```

```
            "arn:aws:ssm:*:account-id3:resource-data-sync/*",
            "arn:aws:ssm:*:account-id4:resource-data-sync/*"
          ]
        }
      }
    }
  ]
}
```

5. (選用) 如果您想要加密同步作業，您必須將以下條件新增至前一步驟中所列的政策。將這些新增至 StringEquals 章節中。

```
"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"
```

請見此處範例：

```
"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. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請安裝並設定 。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

7. （選用）如果您想要加密同步，請執行下列命令來驗證儲存貯體政策是否強制執行 AWS KMS 金鑰需求。將每個########取代為您自己的資訊。

   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. 請執行下列命令，以您在此程序開頭建立的 Amazon S3 儲存貯體，建立資源資料同步組態。此命令會從 AWS 區域 您登入的 建立同步。

> **Note**
>
> 如果同步與目標 Amazon S3 儲存貯體位於不同區域，您可能需要支付資料傳輸費用。如需詳細資訊，請參閱 Amazon S3 定價。

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"
```

您可以透過 `region` 參數，指定應建立同步組態的位置。在以下範例中，系統會將來自 us-west-1 區域的庫存資料同步至 us-west-2 區域內的 Amazon S3 儲存貯體。

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
```

（選用）如果您想要使用 加密同步 AWS KMS，請執行下列命令來建立同步。如果您加密同步，則 AWS KMS 金鑰和 Amazon S3 儲存貯體必須位於相同的區域。

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. 執行下列命令，藉此檢視同步組態狀態。

```
aws ssm list-resource-data-sync
```

若您在不同區域中建立同步組態，請務必指定 `region` 參數，如下方範例所示。

```
aws ssm list-resource-data-sync --region us-west-1
```

10. 成功建立同步組態後，請檢查 Amazon S3 中的目標儲存貯體。庫存資料應該會在幾分鐘內顯示。

使用 Amazon Athena 的資料

下節會介紹在 Amazon Athena 中檢視及查詢資料的方法。開始操作前，建議您了解 Athena。如需詳細資訊，請參閱《Amazon Athena 使用者指南》中的什麼是 Amazon Athena？和使用資料。

檢視和查詢 Amazon Athena 中的資料

1. 前往 https://console.aws.amazon.com/athena/ 開啟 Athena 主控台。
2. 複製下列陳述式並貼至查詢編輯器，然後選擇執行查詢。

```
CREATE DATABASE ssminventory
```

   系統會建立名為 ssminventory 的資料庫。

3. 複製下列陳述式並貼至查詢編輯器，然後選擇執行查詢。使用 Amazon S3 目標的名稱和字首取代 amzn-s3-demo-bucket 和 *bucket_prefix*。

```
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. 複製下列陳述式並貼至查詢編輯器，然後選擇執行查詢。

```
MSCK REPAIR TABLE ssminventory.AWS_Application
```

   系統將分割資料表。

> **Note**
>
> 如果您從其他 AWS 區域 或 建立資源資料同步 AWS 帳戶，則必須再次執行此命令來更新分割區。另外，您可能也需要更新 Amazon S3 儲存貯體政策。

5. 選擇 AWS_Application 資料表旁的檢視圖示，即可預覽資料。

6. 複製下列陳述式並貼至查詢編輯器，然後選擇執行查詢。

```
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
```

查詢會傳回不同版本的 計數aws-cfn-bootstrap，這是 LinuxmacOS、 和 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上存在 AWS 的應用程式Windows Server。

7. 分別複製下列陳述式並貼至查詢編輯器，使用 Amazon S3 的相關資訊取代 amzn-s3-demo-bucket 和 *bucket-prefix*，然後選擇執行查詢。這些陳述式能夠設定 Athena 中的其他庫存資料表。

```
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 中的資料

下節將概要說明可用來在 Amazon QuickSight 中建置視覺效果的連結。

在 Amazon QuickSight 中建置視覺效果

1. 註冊 Amazon QuickSight，然後登入 QuickSight 主控台。
2. 建立來自您所建立之 AWS_Application 資料表和任何其他資料表的資料集。如需詳細資訊，請參閱使用 Amazon Athena 資料建立資料集。
3. 聯結資料表。舉例而言，您可以聯結 AWS_InstanceInformation 中的 instanceid 欄位，因為該欄位與其他庫存資料表中的 resourceid 欄位相符。如需聯結資料表的詳細資訊，請參閱聯結資料表。
4. 建置視覺效果。如需詳細資訊，請參閱使用 Amazon QuickSight 視覺效果。

對 Systems Manager 庫存的問題進行故障診斷

本主題包含有關如何對 AWS Systems Manager 庫存的常見錯誤或問題進行故障診斷的資訊。如果在 Systems Manager 中檢視節點時發生問題，請參閱 疑難排解受管節點的可用性。

主題

- 不支援具有文件 'AWS-GatherSoftwareInventory' 的多個應用全部關聯。
- 庫存執行狀態永遠不會結束擱置
- AWS-ListWindowsInventory 文件無法執行
- 主控台沒有顯示 Inventory (庫存) 儀表板 | Detailed View (詳細檢視) | Settings (設定) 標籤
- UnsupportedAgent
- 略過
- 失敗
- Amazon EC2 執行個體的庫存合規失敗
- S3 儲存貯體物件包含舊資料

不支援具有文件 '**AWS-GatherSoftwareInventory**' 的多個應用全部關聯。

`Multiple apply all associations with document 'AWS-GatherSoftwareInventory' are not supported` 表示嘗試設定所有節點之庫存關聯的一個或多個 AWS 區域 已經使用所有節點的庫存關聯進行設定的錯誤。必要時，您可以刪除所有節點的現有庫存關聯，然後建立新的庫存關聯。若要檢視現有的庫存關聯，請選擇 Systems Manager 主控台中的 State Manager，然後找出使用

AWS-GatherSoftwareInventory SSM 文件的關聯。如果所有節點的現有庫存關聯是跨多個區域建立的，而您想要建立新的關聯，則必須從每個存在現有關聯的區域中刪除現有關聯。

庫存執行狀態永遠不會結束擱置

庫存收集永遠不會結束 Pending 狀態的原因有兩個：

- 選取的節點中沒有 AWS 區域：

  如果您使用 Systems Manager Quick Setup 建立全域庫存關聯，庫存關聯的狀態 (AWS-GatherSoftwareInventory 文件) 會顯示 Pending (如果選取的區域中沒有可用的節點)。

- 許可不足：

  如果一或多個節點沒有執行 Systems Manager 庫存的許可，則庫存關聯會顯示 Pending。確定 AWS Identity and Access Management (IAM) 執行個體設定檔包含 AmazonSSMManagedInstanceCore 受管政策。如需如何將此政策新增到執行個體設定檔的資訊，請參閱 EC2 執行個體許可的替代組態。

  至少，執行個體設定檔必須具有以下 IAM 許可。

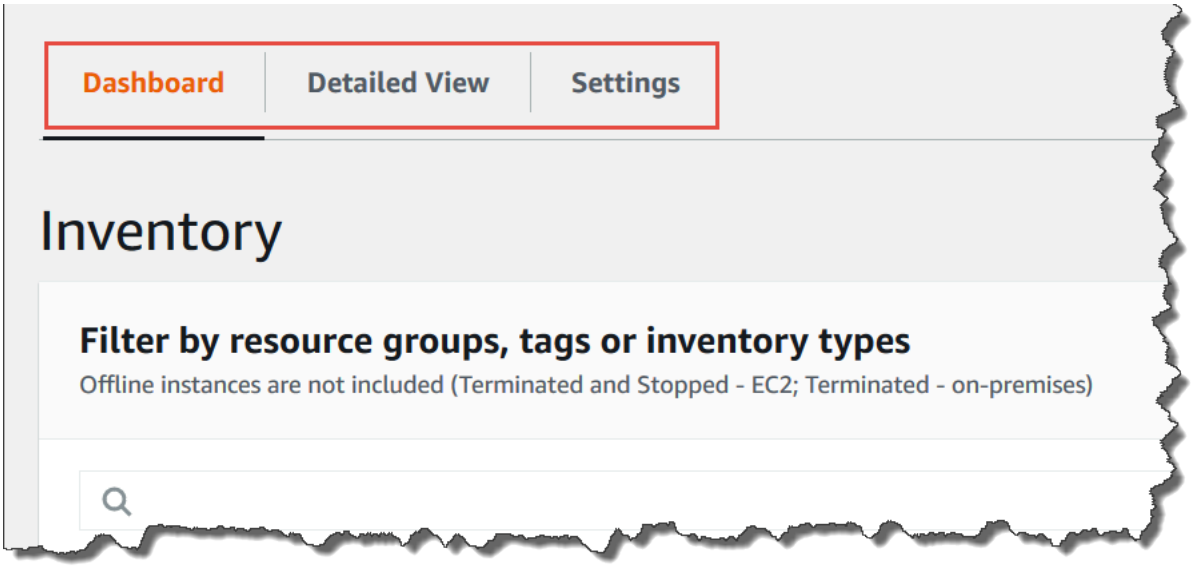
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeAssociation",
                "ssm:ListAssociations",
                "ssm:ListInstanceAssociations",
                "ssm:PutInventory",
                "ssm:PutComplianceItems",
                "ssm:UpdateAssociationStatus",
                "ssm:UpdateInstanceAssociationStatus",
                "ssm:UpdateInstanceInformation",
                "ssm:GetDocument",
                "ssm:DescribeDocument"
            ],
            "Resource": "*"
        }
    ]
}
```

`AWS-ListWindowsInventory` 文件無法執行

`AWS-ListWindowsInventory` 文件已棄用。請勿使用此文件來收集庫存。請改為使用設定清查收集中所述的其中一個處理程序。

主控台沒有顯示 Inventory (庫存) 儀表板 | Detailed View (詳細檢視) | Settings (設定) 標籤

庫存 Detailed View (詳細檢視) 頁面僅在提供 Amazon Athena 的 AWS 區域 中可用。若在庫存頁面上並未顯示下列索引標籤，則表示 Athena 並未在該區域中提供使用，且您無法使用 Detailed View (詳細檢視) 來查詢資料。



UnsupportedAgent

如果庫存關聯的詳細狀態顯示 UnsupportedAgent，且關聯狀態顯示失敗，則受管節點 AWS Systems Manager SSM Agent上的 版本不正確。若要建立全域清查關聯 （ 清查您 中的所有節點 AWS 帳戶），您必須使用 2SSM Agent.0.790.0 版或更新版本。您可以前往 Managed Instances (受管執行個體) 頁面的 Agent version (代理程式版本) 欄位，檢視每個節點上執行的代理程式版本。如需如何在節點上更新 SSM Agent的相關資訊，請參閱 使用 Run Command 更新 SSM Agent。

略過

如果節點的庫存關聯狀態顯示 Skipped (略過)，代表您已建立全域庫存關聯 (收集所有節點的庫存)，但遭略過的節點已擁有指派的庫存關聯。系統沒有將全域庫存關聯指派給此節點，因此全域庫存關聯也不會收集任何庫存資料。不過，當系統執行現有的庫存關聯時，該節點仍會回報庫存資料。

如果您不想讓全域庫存關聯略過節點，則必須刪除現有的庫存關聯。若要檢視現有的庫存關聯，請選擇 Systems Manager 主控台中的 State Manager，然後找出使用 `AWS-GatherSoftwareInventory` SSM 文件的關聯。

失敗

如果節點的庫存關聯狀態顯示 Failed (失敗)，這可能表示節點有多個指派的庫存關聯。一次只能指派一個庫存關聯給節點。庫存關聯使用 `AWS-GatherSoftwareInventory` AWS Systems Manager 文件 (SSM 文件) 。您可以使用 AWS Command Line Interface (AWS CLI) 檢視節點的關聯清單，以執行下列命令。

```
aws ssm describe-instance-associations-status
                --instance-id instance ID
```

Amazon EC2 執行個體的庫存合規失敗

如果您將多個庫存關聯指派給執行個體，則 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體的庫存合規可能會失敗。

若要解決此問題，請刪除指派給執行個體的一或多個庫存關聯。如需詳細資訊，請參閱刪除關聯。

> **Note**
>
> 如果您為受管節點建立多個庫存關聯，請注意下列行為。
>
> - 可以為每個節點指派以所有節點為目標的庫存關聯 (--targets "Key=InstanceIds,Values=*")。
> - 每個節點也可以指派使用標籤鍵值對或 AWS 資源群組的特定關聯。
> - 如果為節點指派多個庫存關聯，則尚未執行之關聯的狀態會顯示為略過。最近執行的關聯會顯示庫存關聯的實際狀態。
> - 如果為節點指派多個庫存關聯，且每個都使用標籤索引鍵/值對，則由於標籤衝突，這些庫存關聯無法在節點上執行。該關聯仍然在沒有標籤索引鍵/值衝突的節點上執行。

S3 儲存貯體物件包含舊資料

當庫存關聯成功並發現新資料時，Amazon S3 儲存貯體物件內的資料會進行更新。當關聯執行但失敗時，每個節點的 Amazon S3 儲存貯體物件都會更新，但在此情況下，物件內部的資料不會得到更新。只有在關聯成功執行時，Amazon S3 儲存貯體物件內的資料才會進行更新。庫存關聯失敗時，Amazon S3 儲存貯體中的資料為舊資料。

AWS Systems Manager Patch Manager

Patch Manager中的工具 會使用安全相關更新和其他類型的更新 AWS Systems Manager，自動修補受管節點的程序。

> **Note**
>
> Systems Manager 支援 中的修補程式政策Quick Setup，這是 中的工具 AWS Systems Manager。使用修補程式政策，是設定修補操作的建議方法。使用單一修補程式政策組態，您可以定義為組織中所有區域的所有帳戶、僅您選擇的帳戶和區域或者單一帳戶-區域對進行修補。如需詳細資訊，請參閱Quick Setup中的修補程式政策組態。

您可以使用 Patch Manager 以套用適用於作業系統和應用程式的修補程式。(在 Windows Server 上，應用程式支援僅限於由 Microsoft 發行的應用程式更新。) 您可以使用 Patch Manager 在 Windows 節點上安裝 Service Pack，並在 Linux 節點上執行次要版本升級。您可以依據作業系統類型，修補 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、邊緣裝置或內部部署伺服器與虛擬機器 (VM)。這包括數個作業系統的支援版本，如 Patch Manager 先決條件 中所列。您可以掃描執行個體，僅查看遺漏的修補程式報告，或者掃描並自動安裝所有遺漏的修補程式。若要開始使用 Patch Manager，請開啟 Systems Manager 主控台。在導覽窗格中，選擇 Patch Manager。

AWS 在 中提供修補程式之前， 不會對其進行測試Patch Manager。此外，Patch Manager 不支援將主要版本的作業系統升級，例如將 Windows Server 2016 升級至 Windows Server 2019，或 SUSE Linux Enterprise Server (SLES) 12.0 升級至 SLES 15.0。

對於報告修補程式嚴重性級別的 Linux 作業系統類型，Patch Manager 使用軟體發布者為更新通知或個別修補程式報告的嚴重性級別。Patch Manager 不會從第三方來源推導出嚴重性級別，例如通用漏洞評分系統 (CVSS)，或者美國國家漏洞資料庫 (NVD) 發行的指標。

Patch Manager 如何為我的組織帶來益處？

Patch Manager 會使用安全相關更新和其他類型的更新，自動修補受管節點的程序。它提供幾個主要優點：

- 集中式修補控制 – 使用修補程式政策，您可以為組織中的所有區域、特定帳戶和區域或單一帳戶區域對中的所有帳戶設定重複修補操作。
- 彈性修補操作 – 您可以選擇掃描執行個體，只查看遺失修補程式的報告，或掃描並自動安裝所有遺失的修補程式。
- 全面合規報告 – 掃描操作後，您可以檢視哪些受管節點不符合修補程式合規，以及哪些修補程式遺失的詳細資訊。
- 跨平台支援 – Patch Manager支援多個作業系統，包括各種 Linux 發行版本macOS、 和 Windows Server。

- 自訂修補程式基準 – 您可以透過自訂修補程式基準，指定哪些修補程式已核准安裝，來定義組織修補程式合規的構成因素。
- 與其他 AWS 服務的整合 – Patch Manager 整合 AWS Organizations AWS Security Hub、 AWS CloudTrail和 AWS Config ，以增強管理和安全性。
- 確定性升級 – 透過適用於 Amazon Linux 2023 作業系統的版本控制儲存庫支援確定性升級。

誰應該使用Patch Manager？

Patch Manager 專為下列項目而設計：

- 需要維持其受管節點機群修補程式合規性的 IT 管理員
- 需要了解其基礎設施中修補程式合規狀態的營運經理
- 想要大規模實作自動化修補解決方案的雲端架構師
- 需要將修補整合至其操作工作流程的 DevOps 工程師
- 多帳戶/多區域部署需要集中式修補程式管理的組織
- 負責維護 AWS 受管節點、邊緣裝置、內部部署伺服器和虛擬機器的安全狀態和運作狀態的任何人

Patch Manager有哪些主要功能？

Patch Manager 提供多種主要功能：

- 修補程式政策 – 透過與 整合，使用單一政策設定跨多個 AWS 帳戶 和 區域的修補操作 AWS Organizations。
- 自訂修補程式基準 – 定義在修補程式發佈後的 天內自動核准修補程式的規則，以及已核准和拒絕的修補程式清單。
- 多種修補方法 – 從修補政策、維護時段或隨需「立即修補」操作中進行選擇，以滿足您的特定需求。
- 合規報告 – 產生修補程式合規狀態的詳細報告，以 CSV 格式傳送至 Amazon S3 儲存貯體。
- 跨平台支援 – 跨 Windows Server、各種 Linux 發行版本和 修補作業系統和應用程式macOS。
- 排程彈性 – 使用自訂 CRON 或 Rate 表達式設定掃描和安裝修補程式的不同排程。
- 生命週期掛鉤 – 使用 Systems Manager 文件在修補操作前後執行自訂指令碼。
- 安全重點 – 預設情況下， Patch Manager 專注於與安全相關的更新，而不是安裝所有可用的修補程式。

- 速率控制 – 設定修補操作的並行和錯誤閾值，以將操作影響降至最低。

什麼是 中的合規Patch Manager？

Systems Manager 機群中構成受管節點修補程式合規性的基準，並非由作業系統 AWS(OS) 供應商或第三方定義，例如安全諮詢公司。

反之，您可以定義修補程式合規對組織或帳戶中受管節點的意義，也就是修補程式基準。修補程式基準是一種組態，可指定必須在受管節點上安裝修補程式的規則。受管節點在與您在修補程式基準中指定的核准條件相符的所有修補程式處於最新狀態時，即符合修補程式合規。

請注意，符合修補程式基準並不表示受管節點一定是安全的。合規表示由修補程式基準定義的可用和核准的修補程式都已安裝在節點上。受管節點的整體安全性取決於 範圍之外的許多因素Patch Manager。如需詳細資訊，請參閱安全。

每個修補基準都是特定支援的作業系統 (OS) 類型的組態，例如 Red Hat Enterprise Linux(RHEL)macOS、 或 Windows Server。修補基準可以為所有支援的作業系統版本定義修補規則，或僅限於您指定的修補規則，例如 RHEL 6.10、7.8 RHEL 和 RHEL 9.3。

在修補程式基準中，您可以指定特定分類和嚴重性等級的所有修補程式都已核准用於安裝。例如，您可以包含分類為 `Security` 但排除其他分類的所有修補程式，例如 `Bugfix`或 `Enhancement`。您也可以包含嚴重性為 的所有修補程式`Critical`，並排除其他修補程式，例如 `Important`和 `Moderate`。

您也可以在修補程式基準中明確定義修補程式，方法是將其 IDs 新增至要核准或拒絕的特定修補程式清單，例如 KB2736693 Amazon Linux 2023 `dbus.x86_64:1:1.12.28-1.amzn2023.0.1` (AL2023) 的 Windows Server或 。您可以選擇性地指定修補程式可用後等待修補的特定天數。對於 Linux 和 macOS，您可以選擇為合規指定修補程式的外部清單 （安裝覆寫清單），而非修補程式基準規則定義的修補程式清單。

執行修補操作時， 會根據修補程式基準或安裝覆寫清單中設定的規則，將目前套用至受管節點的修補程式與應套用的修補程式Patch Manager進行比較。您可以選擇讓 Patch Manager 僅顯示遺失修補程式的報告 (`Scan` 操作)，亦可選擇讓 Patch Manager 自動安裝所找到的受管節點中遺失的全部修補程式 (`Scan and install` 操作)。

Patch Manager 提供預先定義的修補基準，可用於修補操作；不過，這些預先定義的組態會以範例的形式提供，而非建議的最佳實務。我們建議您建立自己的自訂修補程式基準，以更好地控制哪些項目構成機群的修補程式合規。

如需修補程式基準的詳細資訊，請參閱下列主題：

- 預先定義和自訂的修補基準
- 已核准與遭拒的修補程式清單的套件名稱格式
- 查看 AWS 預先定義的修補基準
- 使用自訂修補基準
- 使用修補程式合規報告

主要元件

開始使用Patch Manager工具之前，您應該先熟悉工具修補操作的一些主要元件和功能。

修補基準

Patch Manager 會使用修補基準，其中包含在修補程式發行後的數日內自動核准修補程式等規則，以及已核准和拒絕的修補程式可選清單。執行修補操作時，Patch Manager 會將當前套用至受管節點的修補程式與應當根據修補基準中設定之規則來進行套用的修補程式進行比較。您可以選擇讓 Patch Manager 僅顯示遺失修補程式的報告 (`Scan` 操作)，亦可選擇讓 Patch Manager 自動安裝所找到的受管節點中遺失的全部修補程式 (`Scan and install` 操作)。

修補操作方法

Patch Manager 目前提供了四種執行 `Scan` 和 `Scan and install` 操作的方法：

- （建議）在 中設定的修補程式政策 Quick Setup – 根據與 的整合 AWS Organizations，單一修補程式政策可以定義整個組織的修補排程和修補程式基準，包括多個 AWS 帳戶 和所有 AWS 區域 這些帳戶在其中操作。修補程式政策也可以只針對組織中的某些組織單位 (OU)。您可以使用單一修補程式政策依照不同的排程進行掃描和安裝。如需詳細資訊，請參閱使用修補程式政策設定組織中執行個體的Quick Setup修補及Quick Setup中的修補程式政策組態。
- 在 中設定的主機管理選項 Quick Setup – 與 整合也支援主機管理組態 AWS Organizations，因此可以對整個 Organization 執行修補操作。不過，此選項僅限於使用目前預設修補基準掃描遺失的修補程式，並在合規報告中提供結果。此操作方法無法安裝修補程式。如需詳細資訊，請參閱使用 Quick Setup 設定 Amazon EC2 主機管理。
- 執行修補程式**`Scan`**或**`Install`**任務的維護時段 – 您在 Systems Manager 工具中設定的維護時段，稱為 Maintenance Windows，可設定為依照您定義的排程執行不同類型的任務。Run Command 類型的任務可用來在您選擇的一組受管節點上執行 `Scan` 或 `Scan and install` 任務。每個維護時段任務只能以 AWS 帳戶單一AWS 區域 配對中的受管節點為目標。如需詳細資訊，請參閱教學課程：使用主控台建立修補維護時段。

- Patch Manager 中的隨需立即修補操作：立即修補選項可讓您在需要盡快修補受管節點時略過排程設定。使用 Patch now (立即修補)，可指定是否執行 `Scan` 或 `Scan and install` 操作，以及要在哪些受管節點上執行操作。您也可以選擇在修補操作期間，將 Systems Manager 文件 (SSM 文件) 作為 lifecycle hook 執行。每個修補程式現在只能以 AWS 帳戶單一AWS 區域 配對中的受管節點為目標。如需詳細資訊，請參閱隨需修補受管節點。

合規報告

`Scan` 操作完成後，您可以使用 Systems Manager 主控台來檢視哪些受管節點不符合修補程式規範，以及每個節點遺失了哪些修補程式。您也可以產生可傳送至所選 Amazon Simple Storage Service (Amazon S3) 儲存貯體的修補程式合規報告，格式為 .csv。您可以產生一次性報告，或定期產生報告。對於單一受管節點，報告包含節點之所有修補程式的詳細資訊。對於所有受管節點的報告，只會提供缺少修補程式數量的摘要。產生報告後，您可以使用 Amazon QuickSight 等工具來匯入和分析資料。如需詳細資訊，請參閱使用修補程式合規報告。

> **Note**
>
> 透過使用修補程式政策產生之合規項的執行類型為 `PatchPolicy`。不是在修補程式政策操作中產生的合規項的執行類型為 `Command`。

整合

Patch Manager 與下列其他 整合 AWS 服務：

- AWS Identity and Access Management (IAM) – 使用 IAM 控制哪些使用者、群組和角色可以存取 Patch Manager 操作。如需詳細資訊，請參閱 AWS Systems Manager 搭配 IAM 的運作方式 和設定 Systems Manager 所需的執行個體許可。
- AWS CloudTrail – 使用 CloudTrail 記錄由使用者、角色或群組啟動之修補操作事件的可稽核歷史記錄。如需詳細資訊，請參閱使用 記錄 AWS Systems Manager API 呼叫 AWS CloudTrail。
- AWS Security Hub – 來自 的修補程式合規資料Patch Manager可以傳送到 AWS Security Hub。Security Hub 可為您提供高優先級安全性警示和合規性狀態的全方位檢視。它還會監控您的機群的修補狀態。如需詳細資訊，請參閱將 AWS Security Hub 與 Patch Manager 整合。
- AWS Config – 在 中設定記錄 AWS Config ，以在Patch Manager儀表板中檢視 Amazon EC2 執行個體管理資料。如需詳細資訊，請參閱檢視修補程式儀表板摘要。

主題

- Quick Setup中的修補程式政策組態
- Patch Manager 先決條件
- Patch Manager 操作的運作方式
- 用於修補受管節點的 SSM 命令文件
- 修補基準
- 在 Amazon Linux 2 受管節點上使用 Kernel Live Patching
- 透過主控台使用 Patch Manager 資源與合規
- 透過AWS CLI使用 Patch Manager 資源
- AWS Systems ManagerPatch Manager 教學課程
- Patch Manager 疑難排解

Quick Setup中的修補程式政策組態

AWS 建議使用修補程式政策來設定組織的修補 AWS 帳戶。修補程式政策於 2022 年 12 月在 Patch Manager 中推出。

修補程式政策是您使用 中的Quick Setup工具 所設定的組態 AWS Systems Manager。與先前設定修補的方法相比，修補程式政策可提供更廣泛且更集中的修補操作控制。修補程式政策可用於 Patch Manager 支援的所有作業系統，包括 Linux、macOS 和 Windows Server 的支援版本。如需建立修補程式政策的說明，請參閱使用修補程式政策設定組織中執行個體的Quick Setup修補。

修補程式政策的主要功能

請不要使用其他修補節點的方法，而是使用修補程式政策來利用這些主要功能：

- 單一設定 – 使用維護時段或 State Manager 關聯來設定修補操作可能需要在 Systems Manager 主控台的不同部分中執行多項任務。使用修補程式政策，您可以在單一精靈中設定所有修補操作。
- 多帳戶/多區域支援 - 現在使用 中的維護時段、State Manager關聯或修補程式功能Patch Manager，您僅限於以 AWS 帳戶單一AWS 區域 對中的受管節點為目標。如果您使用多個帳戶和多個區域，則設定與維護任務可能需要大量的時間，因為您必須在每個帳戶-區域對中執行設定任務。不過，如果您使用 AWS Organizations，您可以設定一個修補程式政策，該政策適用於所有 中所有 AWS 區域受管節點 AWS 帳戶。或者，如果您進行了選擇，則修補程式政策只能套用至您選擇的帳戶和區域中的某些組織單位 (OU)。如果您進行了選擇，則修補程式政策也可套用至單一本機帳戶。
- 組織層級的安裝支援 – Quick Setup 中的現有主機管理組態選項可支援受管節點的每日掃描，以確保修補程式的合規性。不過，此掃描會在預定時間完成，並且只會產生修補程式合規資訊。不會執行修

補程式安裝。使用修補程式政策時，您可以指定不同的掃描和安裝排程。您也可以使用自訂 CRON 或 Rate 運算式來選擇這些操作的頻率和時間。例如，您可以每天掃描遺失的修補程式，以提供定期更新的合規資訊。但是，您的安裝排程可能每週只有一次，以避免不必要的停機時間。

- 簡化的修補基準選取 – 修補程式政策仍包含修補基準，而且修補基準的設定方式沒有任何變更。不過，當您建立或更新修補程式政策時，您可以在單一清單中選取您要用於每個作業系統 (OS) 類型的 AWS 受管或自訂基準。您不需要在單獨的任務中為每個作業系統類型指定預設基準。

> **Note**
>
> 當修補操作基於修補程式政策執行時，其會使用 `AWS-RunPatchBaseline` SSM 文件。如需詳細資訊，請參閱用於修補的 SSM 命令文件：AWS-RunPatchBaseline。

相關資訊

使用 Systems Manager 在您的 AWS 組織中集中部署修補操作 Quick Setup(AWS 雲端操作和遷移部落格）

修補程式政策的其他差異

以下是使用修補程式政策而非先前設定修補的方法時需注意的一些其他差異：

- 不需要修補程式群組 – 在先前的修補操作中，您可以標記屬於一個修補程式群組的多個節點，然後指定要用於該修補程式群組的修補基準。如果沒有定義修補程式群組，則 Patch Manager 會使用作業系統類型的當前預設修補基準來修補執行個體。使用修補程式政策，不再需要設定和維護修補程式群組。

  > **Note**
  >
  > 在 2022 年 12 月 22 日發行修補程式政策支援之前，尚未使用修補程式群組的 account-Region 對主控台不支援修補程式群組功能。在此日期之前開始使用修補程式群組的帳戶區域對中，修補程式群組功能仍然可用。

- 「設定修補程式」頁面已移除 – 在發行修補程式政策之前，您可以在 Configure patching (設定修補程式) 頁面中指定要修補的節點、修補排程以及修補操作的預設值。此頁面已從 Patch Manager 中移除。這些選項現在已在修補程式政策中指定。
- 沒有「立即修補」支援 – 隨需修補節點的功能仍僅限於一次單一 AWS 帳戶AWS 區域 配對。如需相關資訊，請參閱 隨需修補受管節點。

- 修補程式政策和合規資訊 – 根據修補程式政策組態掃描受管節點是否合規時，即可提供合規資料供您使用。您可以使用與其他合規掃描方法相同的方式來檢視和使用資料。雖然您可以為整個組織或多個組織單位設定修補程式政策，但每個 AWS 帳戶AWS 區域 對的合規資訊都會個別報告。如需詳細資訊，請參閱使用修補程式合規報告。
- 關聯合規狀態和修補程式政策：Quick Setup 修補程式政策下受管節點的修補狀態符合該節點的 State Manager 關聯執行狀態。如果關聯執行狀態為 `Compliant`，受管節點的修補程式狀態也會標示為 `Compliant`。如果關聯執行狀態為 `Non-Compliant`，受管節點的修補程式狀態也會標示為 `Non-Compliant`。

AWS 區域 支援修補程式政策

下列區域目前支援 Quick Setup 中的修補程式政策組態：

- 美國東部 (俄亥俄) (us-east-2)
- 美國東部 (維吉尼亞北部) (us-east-1)
- 美國西部 (加利佛尼亞北部) (us-west-1)
- 美國西部 (奧勒岡) (us-west-2)
- 亞太區域 (孟買) (ap-south-1)
- 亞太區域 (首爾) (ap-northeast-2)
- 亞太區域 (新加坡) (ap-southeast-1)
- 亞太區域 (雪梨) (ap-southeast-2)
- 亞太區域 (東京) (ap-northeast-1)
- 加拿大 (中部) (ca-central-1)
- 歐洲 (法蘭克福) (eu-central-1)
- 歐洲 (愛爾蘭) (eu-west-1)
- 歐洲 (倫敦) (eu-west-2)
- 歐洲 (巴黎) (eu-west-3)
- 歐洲 (斯德哥爾摩) (eu-north-1)
- 南美洲 (聖保羅) (sa-east-1)

Patch Manager 先決條件

在 中使用 工具之前Patch Manager，請確定您已符合必要的先決條件 AWS Systems Manager。

主題

- SSM Agent 版本
- Python 版本
- 與修補程式來源的連線
- S3 端點存取
- 在本機安裝修補程式的許可
- Patch Manager 支援的作業系統

SSM Agent 版本

執行於您要以 Patch Manager 進行管理之受管節點的 SSM Agent 版本 2.0.834.0 或更新版本。

> **Note**
>
> 每當將新工具新增至 Systems Manager 或對現有工具進行更新時，SSM Agent就會發佈 的更新版本。若未使用最新版本的代理程式，您的受管節點可能會無法使用各種 Systems Manager 工具和功能。因此，我們建議您讓機器的 SSM Agent 自動保持最新狀態。如需相關資訊，請參閱自動化 SSM Agent 更新。請訂閱 GitHub 上的 SSM Agent 版本備註頁面，以便接收有關 SSM Agent 更新的通知。

Python 版本

針對 macOS 以及大多數 Linux 作業系統 (OS)，Patch Manager 目前支援 Python 2.6 - 3.10 版。AlmaLinux、Debian Server、Raspberry Pi OS 及 Ubuntu Server 作業系統需要支援的 Python 3 版 (3.0 - 3.10)。

與修補程式來源的連線

如果您的受管節點沒有直接連線至網際網路，而且您使用具有 VPC 端點的 Amazon Virtual Private Cloud (Amazon VPC)，則必須確保節點能夠存取來源修補程式儲存庫 (repos)。在 Linux 節點上，修補程式更新通常會從節點上設定的遠端儲存庫下載。因此，節點必須能夠連接至儲存庫，以便執行修補。如需詳細資訊，請參閱如何選取安全性修補程式。

CentOS 和 CentOS Stream：啟用 **`EnableNonSecurity`** 旗標

CentOS 6 和 7 受管節點使用 Yum 作為套件管理工具。CentOS 8 和 CentOS Stream 節點使用 DNF 作為套件管理工具。這兩個套件管理工具都使用更新通知的概念。更新通知僅只是修復特定問題的套件集合。

不過，CentOS 和 CentOS Stream 預設儲存庫並未設定更新通知。這表示 Patch Manager 不會偵測預設 CentOS 和 CentOS Stream 儲存庫上的套件。若要啟用 Patch Manager 來處理不包含在更新通知中的套件，您必須在修補基準規則中啟用 `EnableNonSecurity` 旗標。

Windows Server：確保連線至 Windows Update 目錄或 Windows Server 更新服務 (WSUS)

Windows Server 受管節點必須能夠連線至 Windows 更新目錄或 Windows 伺服器更新服務 (WSUS)。確認您的節點已透過網際網路閘道、NAT 閘道或 NAT 執行個體連線至 Microsoft 更新目錄。如果使用 WSUS，請確認節點已連線至您環境中的 WSUS 伺服器。如需詳細資訊，請參閱問題：受管節點無法存取 Windows 更新目錄或 WSUS。

S3 端點存取

無論您的受管節點是在私有或公有網路中運作，而無法存取所需的 AWS 受管 Amazon Simple Storage Service (Amazon S3) 儲存貯體，修補操作都會失敗。如需受管節點必須能夠存取之 S3 儲存貯體的相關資訊，請參閱 SSM Agent 與 AWS 受管 S3 儲存貯體通訊 和 使用適用於 Systems Manager 的 VPC 端點來改善 EC2 執行個體的安全性。

在本機安裝修補程式的許可

在 Windows Server 和 Linux 作業系統上，Patch Manager 會分別擔任管理員和根使用者帳戶來安裝修補程式。

然而，在 macOS 上，對於 Brew 和 Brew Cask，Homebrew 不支援其在根使用者帳戶下執行的命令。因此，Patch Manager 以 Homebrew 目錄的擁有者或屬於 Homebrew 目錄之擁有者群組的有效使用者身分查詢和執行 Homebrew 命令。因此，為了安裝修補程式，`homebrew` 目錄的擁有者也需要 `/usr/local` 目錄的遞迴擁有者許可。

> **Tip**
>
> 下列命令為指定使用者提供此許可：
>
> ```
> sudo chown -R $USER:admin /usr/local
> ```

Patch Manager 支援的作業系統

此Patch Manager工具不支援其他 Systems Manager 工具支援的所有相同作業系統版本。例如：Patch Manager 不支援 CentOS 6.3 或 Raspberry Pi OS 8 (Jessie)。(如需 Systems Manager 支援的作業系統完整清單，請參閱 Systems Manager 支援的作業系統。) 因此，確保您要使用 Patch Manager 的受管節點執行於下表所列的作業系統之一。

> **Note**
>
> Patch Manager 依賴在受管節點上設定的修補程式儲存庫，例如 Windows Update 目錄和 Windows 的 Windows Server 更新服務，來擷取可用的修補程式以進行安裝。因此，在終止服務 (EOL) 作業系統版本中，如果沒有可用的新更新，Patch Manager 可能無法報告新的更新。這可能是因為 Linux 發行版本維護者、Microsoft 或 Apple 未發行新的更新，或因為受管節點沒有存取新更新的適當授權。
>
> Patch Manager 針對受管節點上的可用修補程式，報告合規狀態。因此，如果執行個體正在執行 EOL 作業系統，而且沒有可用的更新，Patch Manager 可能會根據針對修補操作設定的修補基準，將節點報告為合規。

作業系統	詳細資訊
Linux	• AlmaLinux 8.x、9.x • Amazon Linux 2012.03–2018.03 • Amazon Linux 2 2.0 和所有更新版本 • Amazon Linux 2022 • 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 (先前稱為 Raspbian) 9 (Stretch) • Red Hat Enterprise Linux (RHEL) 6.5–8.x、9.x、10.x • Rocky Linux 8.x、9.x

作業系統	詳細資訊
	• SUSE Linux Enterprise Server (SLES) 12.0 和 12.x 之後的版本；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

作業系統	詳細資訊
macOS	macOS 僅支援 Amazon EC2 執行個體。 11.3.1；11.4–11.7 (Big Sur) 12.0–12.7（蒙特雷） 13.0–13.7（文圖拉） 14.x (Sonoma) 15.x (Sequoia) macOS OS 更新 Patch Manager 不支援 macOS 的作業系統 (OS) 更新或升級，例如從 12.x 到 13.x 或 13.1 至 13.2。若要在 macOS 上執行 OS 版本更新，我們建議您使用 Apple 內建的 OS 升級機制。如需詳細資訊，請參閱 Apple Developer Documentation 網站上的 Device Management。 Homebrew 支援 Patch Manager 需要將開放原始碼軟體套件管理系統 Homebrew 安裝到下列其中一個預設安裝位置： • `/usr/local/bin` • `/opt/homebrew/bin` 未安裝 Homebrew 時，使用 的修補操作Patch Manager無法正常運作。 Homebrew 已停止支援 macOS 10.14.x (Mojave) 和 10.15.x (Catalina)。因此，這些版本Patch Manager不支援使用 的修補操作。

作業系統	詳細資訊
	區域支援 macOS 不支援全部 AWS 區域。如需有關 macOS Amazon EC2 支援的詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Amazon EC2 Mac instances。 macOS 邊緣裝置 SSM Agent 不支援 AWS IoT Greengrass 核心裝置的 macOS。您無法使用 Patch Manager 修補 macOS 邊緣裝置。

作業系統	詳細資訊
Windows	Windows Server 2008 至 Windows Server 2025，包括 R2 版本。 **Note** SSM Agent Windows 10 不支援 AWS IoT Greengrass 核心裝置的 。您無法使用 Patch Manager 修補 Windows 10 邊緣裝置。 Windows Server 2008 支援 從 2020 年 1 月 14 日起，Microsoft 不再支援 Windows Server 2008 的功能或安全性更新。Windows Server 2008 和 2008 R2 的舊版 Amazon Machine Images (AMIs) 仍包含預先安裝的 SSM Agent 的版本 2，但 Systems Manager 不再正式支援 2008 版本，並且不再更新這些 Windows Server 版本的代理程式。除此之外，SSM Agent 第 3 版可能無法與 Windows Server 2008 和 2008 R2 上的所有操作相容。Windows Server 2008 版本的 SSM Agent 的最終的正式支援版本是 2.3.1644.0。 Windows Server 2012 和 2012 R2 支援 Windows Server 2012 和 2012 R2 已於 2023 年 10 月 10 日終止支援。若要將 Patch Manager 與這些版本搭配使用，我們亦建議使用 Microsoft 的延伸安全性更新 (ESU)。如需詳細資訊，請參閱 Microsoft 網站上的 Windows Server 2012 和 2012 R2 即將終止支援。

Patch Manager 操作的運作方式

本節提供技術詳細資訊Patch Manager，說明 如何 AWS Systems Manager決定要安裝哪些修補程式，以及如何在每個支援的作業系統上安裝這些修補程式。對於 Linux 作業系統，它也提供有關在自訂修補基準中指定來源儲存庫受管節點上預設以外的其他修補程式。本節亦提供有關修補基準規則用於不同 Linux 作業系統發行版的詳細資訊。

> **Note**
>
> 無論您使用哪種方法或組態類型進行修補操作，下列主題中的資訊都適用：
>
> - 在 Quick Setup 中設定的修補程式政策
> - 在 Quick Setup 中設定的主機管理選項
> - 用來執行修補程式 `Scan` 或 `Install` 任務的維護時段
> - 隨需 Patch now (立即修補) 操作

主題

- 套件發行日期和更新日期的計算方式
- 如何選取安全性修補程式
- 如何指定替代修補程式來源儲存庫 (Linux)
- 如何安裝修補程式
- 修補基準規則在 Linux 系統上的運作方式
- Linux 和 Windows Server 之間的修補操作差異

套件發行日期和更新日期的計算方式

> **Important**
>
> 本頁面上的資訊針對適用於 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體的 Amazon Linux 1、Amazon Linux 2、Amazon Linux 2022 和 Amazon Linux 2023 作業系統 (OS)。這些 OS 類型的套件由 Amazon Web Services 建立和維護。其他作業系統的製造商管理其套件和儲存庫的方式，會影響其發行日期和更新日期的計算方式。針對 Amazon Linux、Amazon Linux 2、Amazon Linux 2022 和 Amazon Linux 2023 之外的作業系統，例如

Red Hat Enterprise Linux (RHEL) 及 SUSE Linux Enterprise Server (SLES)，請參閱製造商的說明文件，了解有關如何更新和維護套件的資訊。

在您建立的自訂修補程式基準設定中，對於大多數作業系統類型，您可以指定在一定天數後自動核准安裝修補程式。 AWS 提供數個預先定義的修補程式基準，其中包含 7 天的自動核准日期。

自動核准延遲是修補程式發行之後，在自動核准修補程式進行修補之前的等待天數。例如，您可以使用 `CriticalUpdates` 分類建立規則，並將其設定為 7 天的自動核准延遲。因此，發行日期或最後更新日期為 7 月 7 日的新關鍵修補程式會在 7 月 14 日自動核准。

為了避免在 Amazon Linux 1、Amazon Linux 2、Amazon Linux 2022 和 Amazon Linux 2023 上因自動核准延遲導致意外結果，請務必了解其發行日期和更新日期的計算方式。

在大多數情況下，安裝修補程式之前的自動核准等待時間從 `updateinfo.xml` 中的 `Updated Date` 值開始計算，而不是從 `Release Date` 值開始。以下是有關這些日期計算的重要詳細資訊：

- `Release Date` 是發佈通知的日期。這並不意味著該套件必定在關聯的儲存庫中可用。
- `Update Date` 是更新通知的最後日期。通知的更新可以表示小到一段文字或描述更新的內容。這並不意味著該套件於該日期發行，或必定在關聯的儲存庫中可用。

  這意味著一個套件的 `Update Date` 值可能為 7 月 7 日，但直到 7 月 13 日 (例如) 才可以安裝。假設在此案例中，指定 7 天自動核准延遲的修補基準會在 7 月 14 日的 `Install` 操作中執行。由於 `Update Date` 值為執行日期前 7 天，套件中的修補程式和更新會在 7 月 14 日安裝。即使自套件可用於實際安裝以來僅過去 1 天，也會進行安裝。
- 包含作業系統或應用程式修補程式的套件可在初始版本發行後多次更新。
- 套件可以釋放到 AWS 受管儲存庫，但如果稍後發現問題，則會復原。

在某些修補操作中，這些因素可能並不重要。例如，如果將修補基準設定為安裝嚴重性值 `Low` 和 `Medium`、分類 `Recommended` 的修補程式，任何自動核准延遲可能對您的操作影響很小。

但是，如果嚴重或高嚴重性修補程式的時機更為重要，則您可能會想要對安裝修補程式的時機進行更多控制。建議的方法是使用替代的修補程式來源儲存庫，而非預設儲存庫來進行受管節點上的修補操作。

當您建立自訂修補基準時，可以指定替代的修補程式來源儲存庫。在每個自訂修補基準中，您可以指定修補程式來源組態，最多可達 20 個支援 Linux 作業系統的版本。如需詳細資訊，請參閱如何指定替代修補程式來源儲存庫 (Linux)。

如何選取安全性修補程式

中Patch Manager工具 的主要重點 AWS Systems Manager是在受管節點上安裝作業系統安全相關更新。在預設情況下，Patch Manager 不會安裝所有可用的修補程式，而是安裝以安全性為主的部分修補程式。

根據預設， Patch Manager 不會以任何不同名稱的替換套件取代在套件儲存庫中標示為已淘汰的套件，除非安裝另一個套件更新需要此替換。相反地，對於更新套件的命令， Patch Manager 只會報告和安裝已安裝但已淘汰之套件的遺失更新。這是因為取代過時的套件通常需要解除安裝現有的套件並安裝其取代。取代過時的套件可能會帶來重大變更或您不想要的其他功能。

此行為與 YUM 和 DNF 的`update-minimal`命令一致，其著重於安全性更新，而不是功能升級。如需詳細資訊，請參閱the section called “如何安裝修補程式”。

對於報告修補程式嚴重性級別的 Linux 作業系統類型，Patch Manager 使用軟體發佈者為更新通知或個別修補程式報告的嚴重性級別。Patch Manager 不會從第三方來源推導出嚴重性級別，例如通用漏洞評分系統 (CVSS)，或者美國國家漏洞資料庫 (NVD) 發佈的指標。

> **Note**
>
> 在 Patch Manager 支援的所有 Linux 為基礎的系統上，您可以選擇為受管節點設定不同的來源儲存庫，通常用於安裝非安全性更新。如需相關資訊，請參閱如何指定替代修補程式來源儲存庫 (Linux)。

本節的其他部分將說明 Patch Manager 如何為不同的支援作業系統選擇安全性修補程式。

Amazon Linux 1, Amazon Linux 2, Amazon Linux 2022, and Amazon Linux 2023

預先設定的儲存庫在 Amazon Linux 1 和 Amazon Linux 2 上的處理方式與在 Amazon Linux 2022 和 Amazon Linux 2023 上不同。

在 Amazon Linux 1 和 Amazon Linux 2 上，Systems Manager 修補基準服務會使用受管節點上預先設定的儲存庫。節點上通常會有兩個預先設定的儲存庫 (repos)：

在 Amazon Linux 1 上

- 儲存庫 ID：`amzn-main/latest`

  儲存庫名稱：`amzn-main-Base`
- 儲存庫 ID：`amzn-updates/latest`

儲存庫名稱：amzn-updates-Base

在 Amazon Linux 2 上

- 儲存庫 ID：amzn2-core/2/*architecture*

  儲存庫名稱：Amazon Linux 2 core repository
- 儲存庫 ID：amzn2extra-docker/2/*architecture*

  儲存庫名稱：Amazon Extras repo for docker

> **Note**
>
> *##*可以是 x86_64 或 aarch64。

當您建立 Amazon Linux 2023 (AL2023) 執行個體時，它包含 AL2023 版本中可用的更新，以及 AMI您選擇的特定更新。您的 AL2023 執行個體在啟動時不會自動接收其他重要和重要的安全性更新。相反地，透過 AL2023 預設開啟的版本控制儲存庫功能進行確定性升級，您可以根據符合您特定需求的排程來套用更新。如需詳細資訊，請參閱《Amazon Linux 2023 使用者指南》中的透過版本化儲存庫使用決定性升級一節。

在 Amazon Linux 2022 上，預先設定的儲存庫會繫結至套件更新的鎖定版本。當新的 Amazon Machine Images (AMIs) for Amazon Linux 2022 發布時，會鎖定至特定版本。針對修補程式更新，Patch Manager 會擷取修補程式更新儲存庫的最新鎖定版本，然後根據該鎖定版本的內容，更新受管節點上的套件。

在 AL2023 上，預先設定的儲存庫如下：

- 儲存庫 ID：amazonlinux

  儲存庫名稱：Amazon Linux 2023 儲存庫

在 Amazon Linux 2022 (預覽版本) 上，預先設定的儲存庫會繫結至套件更新的鎖定版本。當新的 Amazon Machine Images (AMIs) for Amazon Linux 2022 發布時，會鎖定至特定版本。針對修補程式更新，Patch Manager 會擷取修補程式更新儲存庫的最新鎖定版本，然後根據該鎖定版本的內容，更新受管節點上的套件。

在 Amazon Linux 2022 上，預先設定的存儲庫如下：

- 儲存庫 ID：`amazonlinux`

  儲存庫名稱：Amazon Linux 2022 儲存庫

> **Note**
>
> 所有更新會從受管節點上設定的遠端儲存庫下載。因此，此節點必須具有傳出至網際網路的存取權，以便連線至儲存庫以執行修補。

Amazon Linux 1 和 Amazon Linux 2 受管節點使用 Yum 作為套件管理工具。Amazon Linux 2022 和 Amazon Linux 2023 使用 Yum 做為套件管理器。

這兩個套件管理工具都使用更新通知的概念做為一個名為 `updateinfo.xml` 的檔案。更新通知僅只是修復特定問題的套件集合。更新通知中的所有套件皆被 Patch Manager 視為是安全的。個別套件不會被指派分類或嚴重性等級。因此，Patch Manager 會指派更新通知的屬性給相關的套件。

> **Note**
>
> 如果您在建立修補基準頁面中選取包含非安全性更新核取方塊，則在 `updateinfo.xml` 檔案中未分類的套件 (或包含檔案但未正確格式化分類、嚴重性和日期值的套件) 可包含在預先篩選的修補程式清單中。但是，若要套用修補程式，修補程式仍必須符合使用者指定的修補基準規則。
>
> 如需包含非安全性更新選項的詳細資訊，請參閱 如何安裝修補程式和 修補基準規則在 Linux 系統上的運作方式。

CentOS and CentOS Stream

在 CentOS 和 CentOS Stream 上，Systems Manager 修補基準服務會使用受管節點上預先設定的儲存庫 (repos)。下列清單提供虛構 CentOS 8.2 Amazon Machine Image (AMI) 的範例：

- 儲存庫 ID：`example-centos-8.2-base`

  儲存庫名稱：`Example CentOS-8.2 - Base`

- 儲存庫 ID：`example-centos-8.2-extras`

儲存庫名稱：`Example CentOS-8.2 - Extras`

- 儲存庫 ID：`example-centos-8.2-updates`

  儲存庫名稱：`Example CentOS-8.2 - Updates`

- 儲存庫 ID：`example-centos-8.x-examplerepo`

  儲存庫名稱：`Example CentOS-8.x – Example Repo Packages`

> **Note**
>
> 所有更新會從受管節點上設定的遠端儲存庫下載。因此，此節點必須具有傳出至網際網路的存取權，以便連線至儲存庫以執行修補。

CentOS 6 和 7 受管節點使用 Yum 作為套件管理工具。CentOS 8 和 CentOS Stream 節點使用 DNF 作為套件管理工具。這兩個套件管理工具都使用更新通知的概念。更新通知僅只是修復特定問題的套件集合。

不過，CentOS 和 CentOS Stream 預設儲存庫並未設定更新通知。這表示 Patch Manager 不會偵測預設 CentOS 和 CentOS Stream 儲存庫上的套件。若要啟用 Patch Manager 來處理不包含在更新通知中的套件，您必須在修補基準規則中啟用 `EnableNonSecurity` 旗標。

> **Note**
>
> 支援 CentOS 和 CentOS Stream 更新通知。啟動後，可下載含有更新通知的儲存區。

Debian Server and Raspberry Pi OS

在 Debian Server 和 Raspberry Pi OS (先前為 Raspbian) 上，Systems Manager 修補基準服務會使用執行個體上預先設定的儲存庫 (repos)。這些預先設定的儲存庫可用於提取更新的可用套件升級清單。因此，Systems Manager 執行相當於 `sudo apt-get update` 命令。

然後套件會從 `debian-security` *`codename`* 儲存庫進行篩選。這表示，在每個 Debian Server 版本上，Patch Manager 只會識別屬於該版本關聯儲存庫的升級，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**
>
> 僅限於 Debian Server 8 上：由於某些 Debian Server 8.* 受管節點會參考已淘汰的套件庫 (`jessie-backports`)，因此 Patch Manager 會執行額外的步驟以確保修補操作成功。如需詳細資訊，請參閱如何安裝修補程式。

Oracle Linux

在 Oracle Linux 上，Systems Manager 修補基準服務會使用受管節點上預先設定的儲存庫 (repos)。節點上通常會有兩個預先設定的儲存庫。

Oracle Linux 7：

- 儲存庫 ID：`ol7_UEKR5/x86_64`

  儲存庫名稱：`Latest Unbreakable Enterprise Kernel Release 5 for Oracle Linux 7Server (x86_64)`
- 儲存庫 ID：`ol7_latest/x86_64`

  儲存庫名稱：`Oracle Linux 7Server Latest (x86_64)`

Oracle Linux 8：

- 儲存庫 ID：`ol8_baseos_latest`

  儲存庫名稱：`Oracle Linux 8 BaseOS Latest (x86_64)`
- 儲存庫 ID：`ol8_appstream`

  儲存庫名稱：`Oracle Linux 8 Application Stream (x86_64)`
- 儲存庫 ID：`ol8_UEKR6`

  儲存庫名稱：`Latest Unbreakable Enterprise Kernel Release 6 for Oracle Linux 8 (x86_64)`

Oracle Linux 9：

- 儲存庫 ID：`ol9_baseos_latest`

  儲存庫名稱：`Oracle Linux 9 BaseOS Latest (x86_64)`
- 儲存庫 ID：`ol9_appstream`

  儲存庫名稱：`Oracle Linux 9 Application Stream Packages(x86_64)`
- 儲存庫 ID：`ol9_UEKR7`

  儲存庫名稱：`Oracle Linux UEK Release 7 (x86_64)`

> **Note**
>
> 所有更新會從受管節點上設定的遠端儲存庫下載。因此，此節點必須具有傳出至網際網路的存取權，以便連線至儲存庫以執行修補。

Oracle Linux 受管節點使用 Yum 做為套件管理工具，且 Yum 使用更新通知的概念，以做為名為 `updateinfo.xml` 的檔案。更新通知僅只是修復特定問題的套件集合。個別套件不會被指派分類或嚴重性等級。出於此原因，Patch Manager 會為相關套件指派更新通知的屬性，並根據修補基準中指定的分類篩選條件安裝套件。

> **Note**
>
> 如果您在建立修補基準頁面中選取包含非安全性更新核取方塊，則在 `updateinfo.xml` 檔案中未分類的套件 (或包含檔案但未正確格式化分類、嚴重性和日期值的套件) 可包含在預先篩選的修補程式清單中。但是，若要套用修補程式，修補程式仍必須符合使用者指定的修補基準規則。

AlmaLinux, RHEL, and Rocky Linux

在 AlmaLinux、Red Hat Enterprise Linux 和 Rocky Linux 上，Systems Manager 修補基準服務會使用受管節點上預先設定的儲存庫 (repos)。節點上通常會有三個預先設定的儲存庫。

所有更新會從受管節點上設定的遠端儲存庫下載。因此，此節點必須具有傳出至網際網路的存取權，以便連線至儲存庫以執行修補。

> **Note**
>
> 如果您在建立修補基準頁面中選取包含非安全性更新核取方塊，則在 `updateinfo.xml` 檔案中未分類的套件 (或包含檔案但未正確格式化分類、嚴重性和日期值的套件) 可包含在預先篩選的修補程式清單中。但是，若要套用修補程式，修補程式仍必須符合使用者指定的修補基準規則。

Red Hat Enterprise Linux 7 受管節點使用 Yum 作為套件管理工具。AlmaLinux、Red Hat Enterprise Linux 8 和 Rocky Linux 受管節點使用 DNF 作為套件管理工具。這兩個套件管理工具都使用更新通知的概念做為一個名為 `updateinfo.xml` 的檔案。更新通知僅只是修復特定問題的套件集合。個別套件不會被指派分類或嚴重性等級。出於此原因，Patch Manager 會為相關套件指派更新通知的屬性，並根據修補基準中指定的分類篩選條件安裝套件。

RHEL 7

> **Note**
>
> 下列儲存庫 ID 與 RHUI 2 相關聯。RHUI 3 於 2019 年 12 月推出，並為 Yum 儲存庫 ID 引入了不同的命名方式。根據您建立受管節點的 RHEL-7 AMI，您可能需要更新命令。如需詳細資訊，請參閱 Red Hat RHEL 客戶入口網站上 AWS 變更的 中 7 的儲存庫 IDs。

- 儲存庫 ID：`rhui-REGION-client-config-server-7/x86_64`

  儲存庫名稱：`Red Hat Update Infrastructure 2.0 Client Configuration Server 7`
- 儲存庫 ID：`rhui-REGION-rhel-server-releases/7Server/x86_64`

  儲存庫名稱：`Red Hat Enterprise Linux Server 7 (RPMs)`
- 儲存庫 ID：`rhui-REGION-rhel-server-rh-common/7Server/x86_64`

  儲存庫名稱：`Red Hat Enterprise Linux Server 7 RH Common (RPMs)`

AlmaLinux 8、RHEL 8 和 Rocky Linux 8

- 儲存庫 ID：`rhel-8-appstream-rhui-rpms`

儲存庫名稱：Red Hat Enterprise Linux 8 for x86_64 - AppStream from RHUI (RPMs)

- 儲存庫 ID：rhel-8-baseos-rhui-rpms

  儲存庫名稱：Red Hat Enterprise Linux 8 for x86_64 - BaseOS from RHUI (RPMs)

- 儲存庫 ID：rhui-client-config-server-8

  儲存庫名稱：Red Hat Update Infrastructure 3 Client Configuration Server 8

AlmaLinux 9、RHEL 9 和 Rocky Linux 9

- 儲存庫 ID：rhel-9-appstream-rhui-rpms

  儲存庫名稱：Red Hat Enterprise Linux 9 for x86_64 - AppStream from RHUI (RPMs)

- 儲存庫 ID：rhel-9-baseos-rhui-rpms

  儲存庫名稱：Red Hat Enterprise Linux 9 for x86_64 - BaseOS from RHUI (RPMs)

- 儲存庫 ID：rhui-client-config-server-9

  儲存庫名稱：Red Hat Enterprise Linux 9 Client Configuration

SLES

在 SUSE Linux Enterprise Server (SLES) 受管節點上，ZYPP 程式庫從下列位置取得可用修補程式的集合 (套件的集合)：

- 儲存庫的清單：etc/zypp/repos.d/*
- 套件資訊：/var/cache/zypp/raw/*

SLES 受管節點使用 Zypper 作為套件管理工具，Zypper 使用修補程式的概念。修補程式只是修復特定問題的套件集合。Patch Manager 處理安全性相關修補程式中參考的所有套件。由於個別套件並未指定分類或嚴重性，因此 Patch Manager 會將修補程式所屬的屬性指派給套件。

Ubuntu Server

在 Ubuntu Server 上，Systems Manager 修補基準服務會使用受管節點上預先設定的儲存庫 (repos)。這些預先設定的儲存庫可用於提取更新的可用套件升級清單。因此，Systems Manager 執行相當於 `sudo apt-get update` 命令。

然後套件會從 *`codename`*`-security` 儲存庫中進行篩選，其中的代號名稱為發行版本專屬，例如 Ubuntu Server 14 的 `trusty`。Patch Manager 僅識別屬於這些儲存庫之一部分的升級：

- 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 作業系統上，Patch Manager 會擷取 Microsoft 透過更新服務發佈的可用更新清單，並自動對 Windows Server Update Services (WSUS) 可用。

> **Note**
>
> Patch Manager 僅為 Patch Manager 支援的 Windows Server 作業系統版本提供可用的修補程式。例如，Patch Manager 無法用於修補 Windows RT。

Patch Manager 持續監控每個 AWS 區域中的新更新。各個區域中可用的更新清單將至少每天重新整理一次。Microsoft 的修補程式資訊經處理後，Patch Manager 會從其修補程式清單中刪除已被更新版本取代的更新。因此，只有最新的更新會顯示出來並提供安裝。例如，如果 `KB4012214` 取代了 `KB3135456`，Patch Manager 中只會有 `KB4012214` 可供更新。

同樣地，Patch Manager 只能在修補操作期間安裝受管節點上提供的修補程式。依預設，Windows Server 2019 和 Windows Server 2022 會移除更新，而這些更新會由稍後的更新取代。因此，如果您在 Windows Server 修補基準中使用 `ApproveUntilDate` 參數，但 `ApproveUntilDate` 參數中選取的日期早於最新修補程式的日期，則會發生下列情況：

- 取代的修補程式會從節點中移除，因此無法使用 Patch Manager 進行安裝。
- 節點上存在最新的替代修補程式，但尚未根據 `ApproveUntilDate` 參數指定的日期核准安裝。

這表示即使可能未安裝上個月的關鍵修補程式，受管節點在 Systems Manager 操作方面也符合規範。使用 `ApproveAfterDays` 參數時，可能會發生相同的情況。由於 Microsoft 取代了修補程式行為，可以設定數字 (通常大於 30 天)，這樣，若在 `ApproveAfterDays` 的天數之前發行 Microsoft 最新的可用修補程式，則永遠不會安裝 Windows Server 的修補程式。請注意，如果您已修改 Windows 群組政策物件 (GPO) 設定，以在受管節點上提供取代的修補程式，則此系統行為不適用。

> **Note**
>
> 在某些情況下，Microsoft 會針對未指定更新日期和時間的應用程式發行修補程式。在這些情況下，預設會提供 `01/01/1970` 的更新日期和時間。

如何指定替代修補程式來源儲存庫 (Linux)

當您使用受管節點上設定的預設儲存庫進行修補操作時， Patch Manager會是 中的工具 AWS Systems Manager，會掃描或安裝與安全相關的修補程式。這是 Patch Manager 的預設行為。如需有關 Patch Manager 如何選擇和安裝安全性修補程式的完整資訊，請參閱 如何選取安全性修補程式。

但是在 Linux 系統上，您也可以使用 Patch Manager 安裝與安全無關的修補程式，或安裝位於與設定於受管節點上的預設儲存庫不同來源儲存庫上的修補程式。當您建立自訂修補基準時，可以指定替代的修補程式來源儲存庫。在每個自訂修補基準中，您可以指定修補程式來源組態，最多可達 20 個支援 Linux 作業系統的版本。

例如，假設您的 Ubuntu Server 機群包含 Ubuntu Server 14.04 和 Ubuntu Server 16.04 受管節點。在這種情況下，您可以為同一自訂修補基準中的每個版本指定備用儲存庫。您為每個版本提供名稱，指定作業系統版本類型 (產品)，然後提供儲存庫組態。您也可以指定單一替代來源儲存庫，適用於所有支援的作業系統版本。

> **Note**
>
> 執行為受管節點指定替代修補程式儲存庫的自訂修補基準並不會將這些儲存庫設為作業系統上的新預設儲存庫。修補操作完成後，先前設定為節點作業系統之預設值的儲存庫會保留為預設儲存庫。

如需使用此選項的範例案例清單，請參閱本主題後面的替代修補程式來源儲存庫使用範例。

如需有關預設和自訂修補基準的詳細資訊，請參閱預先定義和自訂的修補基準。

範例：使用主控台

若要在 Systems Manager 主控台中工作時指定替代的修補程式來源儲存庫，請使用 Create patch baseline (建立修補基準) 頁面上的 Patch sources (修補程式來源) 區段。如需使用 Patch sources (修補程式來源) 選項的詳細資訊，請參閱建立適用於 Linux 的自訂修補基準。

範例：使用 AWS CLI

如需在 AWS Command Line Interface (AWS CLI) 中使用 `--sources` 選項的範例，請參閱 建立包含不同作業系統版本之自訂儲存庫的修補基準。

主題

- 替代儲存庫的重要考量
- 替代修補程式來源儲存庫使用範例

替代儲存庫的重要考量

在您使用替代修補程式儲存庫規劃修補策略時，請注意以下重點。

只有指定的儲存庫會用於修補

指定替代儲存庫不表示指定額外的儲存庫。您可以選擇受管節點上設定為預設儲存庫以外的儲存庫。不過，如果您希望套用預設儲存庫的更新，您也必須指定預設儲存庫做為替代修補程式來源組態的一部分。

例如，在 Amazon Linux 2 受管節點上，預設的儲存庫是 `amzn2-core` 和 `amzn2extra-docker`。若您希望將 Extra Packages for Enterprise Linux (EPEL) 儲存庫包含在您的修補操作中，您必須將這三個儲存庫全部指定為替代儲存庫。

> **Note**
>
> 執行為受管節點指定替代修補程式儲存庫的自訂修補基準並不會將這些儲存庫設為作業系統上的新預設儲存庫。修補操作完成後，先前設定為節點作業系統之預設值的儲存庫會保留為預設儲存庫。

以 YUM 為基礎之分發的修補行為取決於 updateinfo.xml 資訊清單

為以 YUM 為基礎之發行版本 (例如 Amazon Linux 1 或 Amazon Linux 2、Red Hat Enterprise Linux 或 CentOS) 指定替代修補程式儲存庫時，修補行為取決於儲存庫是否包含更新資訊清單，其形式為完整且格式正確的 `updateinfo.xml` 檔案。此檔案指定各種套件的發行日期、分類和嚴重性。以下項目將會影響修補行為：

- 如果您篩選 Classification (分類) 與 Severity (嚴重性)，但是在 `updateinfo.xml` 中並未指定這些項目，則該套件將不會被篩選條件所包含。這也表示沒有 `updateinfo.xml` 檔案的套件不會包含在修補中。
- 如果您篩選 ApprovalAfterDays，但套件發行日期並非 Unix Epoch 格式 (或沒有指定發行日期)，該套件將不會被篩選條件所包含。
- 如果您選取建立修補基準頁面中的包含非安全性更新核取方塊，則會有例外狀況。在這種情況下，沒有 `updateinfo.xml` 檔案的套件 (或包含此檔案但未正確格式化分類、嚴重性和日期值的套件)，會包含在預先篩選的修補程式清單中。(它們必須仍符合其他修補基準規則的要求，才能進行安裝。)

替代修補程式來源儲存庫使用範例

範例 1 – Ubuntu Server 的非安全性更新

您已使用 Patch Manager，使用 AWS提供的預先定義修補程式基準，在Ubuntu Server受管節點的機群上安裝安全修補程式`AWS-UbuntuDefaultPatchBaseline`。您可以建立以此預設為基礎的新修補基準，但在核准規則中指定，您希望也安裝預設分發中非安全性相關的更新。當此修補基準在您的節點上執行時，將會套用安全性與非安全性問題的修補程式。您也可以選擇在您為基準指定的修補程式例外狀況中，核准非安全性修補程式。

範例 2 - Ubuntu Server 的個人套件存檔 (PPA)

您的 Ubuntu Server 受管節點執行透過 Ubuntu 個人套件封存 (PPA) 分發的軟體。在此案例下，您建立一個修補基準，指定您已在受管節點上設定的 PPA 儲存庫做為修補操作的來源儲存庫。然後，使用 Run Command 執行在節點上的修補基準文件。

範例 3 - Amazon Linux 上的內部公司應用程式

您需要在您的 Amazon Linux 受管節點上執行一些應用程式，以符合產業法律合規需求。您可以在節點上為這些應用程式設定儲存庫，使用 YUM 初步安裝應用程式，然後更新或建立新的修補基準，以包含這個新的公司儲存庫。在此之後，您可以使用 Run Command，執行具有 `Scan` 選項的 `AWS-RunPatchBaseline` 文件，查看公司套件是否列在已安裝的套件中，並且在受管節點上是否為最新狀態。如果不是最新的，您可以使用 `Install` 選項更新應用程式以再次執行該文件。

如何安裝修補程式

Patch Manager是 中的工具 AWS Systems Manager，針對作業系統類型使用適當的內建機制，在受管節點上安裝更新。例如，在 Windows Server 上使用 Windows Update API，在 Amazon Linux 2 上使用 `yum` 套件管理員。

Patch Manager 不會安裝取代目前安裝之淘汰套件的新套件。（例外：新套件是正在安裝的另一個套件更新的相依性，或者新套件的名稱與已淘汰的套件相同。）反之， 會Patch Manager報告並安裝可用的更新至已安裝的套件。此方法有助於防止在某個套件取代另一個套件時，系統功能可能發生的意外變更。

如果您需要解除安裝已淘汰的套件並安裝其取代，您可能需要使用自訂指令碼，或在 Patch Manager 的標準操作之外使用套件管理員命令。

本節的其餘部分說明 Patch Manager 如何在作業系統上安裝修補程式。

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 受管節點上，修補程式安裝工作流程如下：

1. 如果使用 https URL 或 Amazon Simple Storage Service (Amazon S3) 路徑樣式 URL (使用 `AWS-RunPatchBaseline` 或 `AWS-RunPatchBaselineAssociation` 文件的 `InstallOverrideList` 參數) 指定修補程式清單，系統會安裝列出的修補程式，並略過步驟 2-7。
2. 套用修補程式基準中指定的 GlobalFilters，以便僅進一步處理合格的套件。
3. 套用修補程式基準中指定的 ApprovalRules。每個核准規則皆可將套件定義為已核准。

然而，核准規則也受限於建立或最後更新修補基準時，是否已選取 Include nonsecurity updates (包含非安全性更新) 核取方塊。

如果非安全更新被排除，將會套用隱含規則，以僅選擇安全儲存庫中包含升級的套件。對於每個套件，套件的候選版本 (通常是最新版本) 必須是安全儲存庫的一部分。

如果包含非安全性更新，則也會考慮來自其他儲存庫的修補程式。

4. 套用修補程式基準中指定的 ApprovedPatches。已核准的修補程式即使被 GlobalFilters 捨棄或 ApprovalRules 中沒有指定核准規則授予其核准，仍將會核准用於更新。
5. 套用修補程式基準中指定的 RejectedPatches。已遭拒的修補程式會從核准的修補程式清單中移除，而且將不會套用。
6. 如果核准修補程式的多個版本，將會套用最新版本。
7. YUM 更新 API (Amazon Linux 1、Amazon Linux 2) 或 DNF 更新 API (Amazon Linux 2022 和 Amazon Linux 2023) 會套用至已核准的修補程式，如下所示：
   - 對於由 AWS提供的預先定義預設修補基準，僅會套用 `updateinfo.xml` 中指定的修補程式 (僅限安全性更新)。這是因為未選取包含非安全性更新核取方塊。預先定義的基準等同於具有下列項目的自訂基準：
     - 未選取包含非安全性更新核取方塊
     - `[Critical, Important]` 嚴重性清單
     - `[Security, Bugfix]` 分類清單

     針對 Amazon Linux 1 和 Amazon Linux 2，此工作流程的同等 yum 命令為：

     ```
     sudo yum update-minimal --sec-severity=Critical,Important --bugfix -y
     ```

     針對 Amazon Linux 2022 和 Amazon Linux 2023，此工作流程的同等 yum 命令為：

     ```
     sudo dnf upgrade-minimal --sec-severity=Critical --sec-severity=Important --
     bugfix -y
     ```

     如果選取包含非安全性更新核取方塊，則位於 `updateinfo.xml` 中的修補程式和不位於 `updateinfo.xml` 中的修補程式皆會套用 (安全性和非安全性更新)。

     針對 Amazon Linux 1 和 Amazon Linux 2，如果選取包含非安全性更新的基準，且具有 `[Critical, Important]` 嚴重性清單和 `[Security, Bugfix]` 分類清單，則等效 yum 命令為：

```
sudo yum update --security --sec-severity=Critical,Important --bugfix -y
```

針對 Amazon Linux 2022 和 Amazon Linux 2023，等效 dnf 命令為：

```
sudo dnf upgrade --security --sec-severity=Critical --sec-severity=Important --
bugfix -y
```

> **Note**
>
> 如果您在 外部執行這些 yum或 dnf命令，則會安裝以不同名稱取代現已淘汰套件的新套件Patch Manager。不過，對等Patch Manager操作不會安裝它們。

Amazon Linux 2022 和 Amazon Linux 2023 的其他修補詳細資訊

支援嚴重性等級 '無'

Amazon Linux 2022 和 Amazon Linux 2023 還支援修補程式嚴重性等級 None，透過 DNF 套件管理器識別。

支援嚴重性等級 'Medium'

針對 Amazon Linux 2022 和 Amazon Linux 2023，修補程式嚴重性等級 Medium 相當於某些外部儲存庫中可能定義的嚴重性等級 Moderate。如果在修補基準中包含 Medium 嚴重性修補程式，則外部修補程式的 Moderate 嚴重性修補程式也會安裝在執行個體上。

當您使用 API 動作 DescribeInstancePatches 查詢合規資料時，嚴重性等級 Medium 篩選會報告嚴重性等級為 Medium 和 Moderate 的修補程式。

Amazon Linux 2022 和 Amazon Linux 2023 的暫時性相依性處理

對於 Amazon Linux 2022 和 Amazon Linux 2023， Patch Manager可能會安裝與同等dnf命令安裝不同的傳輸相依性版本。暫時性相依性是會自動安裝的套件，以滿足其他套件的需求（相依性）。

例如， dnf upgrade-minimal --security 安裝解決已知安全問題所需的最小傳輸相依性版本，而 Patch Manager 安裝相同傳輸相依性的最新可用版本。

8. 如有安裝任何更新，受管節點將會重新啟動。(例外：如果 AWS-RunPatchBaseline 文件中的 RebootOption 參數設為 NoReboot，受管節點不會在 Patch Manager 執行之後重新啟動。如需詳細資訊，請參閱 參數名稱：RebootOption。)

CentOS and CentOS Stream

在 CentOS 和 CentOS Stream 受管節點上，修補程式安裝工作流程如下：

1. 如果使用 https URL 或 Amazon Simple Storage Service (Amazon S3) 路徑樣式 URL (使用 AWS-RunPatchBaseline 或 AWS-RunPatchBaselineAssociation 文件的 InstallOverrideList 參數) 指定修補程式清單，系統會安裝列出的修補程式，並略過步驟 2-7。

   套用修補程式基準中指定的 GlobalFilters，以便僅進一步處理合格的套件。
2. 套用修補程式基準中指定的 ApprovalRules。每個核准規則皆可將套件定義為已核准。

   然而，核准規則也受限於建立或最後更新修補基準時，是否已選取 Include nonsecurity updates (包含非安全性更新) 核取方塊。

   如果非安全更新被排除，將會套用隱含規則，以僅選擇安全儲存庫中包含升級的套件。對於每個套件，套件的候選版本 (通常是最新版本) 必須是安全儲存庫的一部分。

   如果包含非安全性更新，則也會考慮來自其他儲存庫的修補程式。
3. 套用修補程式基準中指定的 ApprovedPatches。已核准的修補程式即使被 GlobalFilters 捨棄或 ApprovalRules 中沒有指定核准規則授予其核准，仍將會核准用於更新。
4. 套用修補程式基準中指定的 RejectedPatches。已遭拒的修補程式會從核准的修補程式清單中移除，而且將不會套用。
5. 如果核准修補程式的多個版本，將會套用最新版本。
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) 會套用至核准的修補程式。

> **Note**
>
> 對於 CentOS， Patch Manager可能會安裝與同等dnf命令安裝不同的傳輸相依性版本。暫時性相依性是會自動安裝的套件，以滿足其他套件的需求 （相依性）。例如， dnf upgrade-minimal --security 安裝解決已知安全問題所需的最低傳輸相依性版本，同時 Patch Manager 安裝相同傳輸相依性的最新可用版本。

7. 如有安裝任何更新，受管節點將會重新啟動。(例外：如果 AWS-RunPatchBaseline 文件中的 RebootOption 參數設為 NoReboot，受管節點不會在 Patch Manager 執行之後重新啟動。如需詳細資訊，請參閱 參數名稱：RebootOption。)

Debian Server and Raspberry Pi OS

在 Debian Server 和 Raspberry Pi OS (之前為 Raspbian) 執行個體上，修補程式安裝工作流程如下：

1. 如果使用 https URL 或 Amazon Simple Storage Service (Amazon S3) 路徑樣式 URL (使用 AWS-RunPatchBaseline 或 AWS-RunPatchBaselineAssociation 文件的 InstallOverrideList 參數) 指定修補程式清單，系統會安裝列出的修補程式，並略過步驟 2-7。
2. 若有可用於 python3-apt (libapt 的 Python 程式庫界面) 的更新，它會升級到最新版本。(這個非安全性套件會升級，即使您未選取 Include nonsecurity updates (包含非安全性更新) 選項)。

   > **Important**
   >
   > 僅限於 Debian Server 8 上：由於某些 Debian Server 8.* 受管節點會參考已淘汰的套件庫 (jessie-backports)，因此 Patch Manager 會執行下列額外的步驟以確保修補操作成功：
   >
   > a. 在您的受管節點上，對 jessie-backports 儲存庫的參考從來源位置清單 (/etc/apt/sources.list.d/jessie-backports) 會變更為註解。因此，不會嘗試從該位置下載修補程式，
   >
   > b. 並會匯入延展安全性更新簽署金鑰。此金鑰提供了 Debian Server 8.* 發行版本的更新和安裝操作所需的許可。
   >
   > c. 系統此時會執行 apt-get 操作以確保在修補程式開始之前已安裝最新版本的 python3-apt。
   >
   > d. 安裝程序完成後，會還原對 jessie-backports 儲存庫的參考，並從 apt 來源金鑰環中移除簽署金鑰。這麼做是為了讓系統組態保持在修補操作之前的狀態。
   >
   > Patch Manager 下次更新系統時，會重複執行相同的程序。

3. 套用修補程式基準中指定的 GlobalFilters，以便僅進一步處理合格的套件。
4. 套用修補程式基準中指定的 ApprovalRules。每個核准規則皆可將套件定義為已核准。

> **Note**
>
> 因為無法可靠地判斷 Debian Server 更新套件的發行日期，此作業系統不支援自動核准選項。

然而，核准規則也受限於建立或最後更新修補基準時，是否已選取 Include nonsecurity updates (包含非安全性更新) 核取方塊。

如果非安全更新被排除，將會套用隱含規則，以僅選擇安全儲存庫中包含升級的套件。對於每個套件，套件的候選版本 (通常是最新版本) 必須是安全儲存庫的一部分。

如果包含非安全性更新，則也會考慮來自其他儲存庫的修補程式。

> **Note**
>
> 對於 Debian Server 和 Raspberry Pi OS 來說，修補候選版本僅限於 `debian-security` 中包含的修補程式。

5. 套用修補程式基準中指定的 ApprovedPatches。已核准的修補程式即使被 GlobalFilters 捨棄或 ApprovalRules 中沒有指定核准規則授予其核准，仍將會核准用於更新。
6. 套用修補程式基準中指定的 RejectedPatches。已遭拒的修補程式會從核准的修補程式清單中移除，而且將不會套用。
7. APT 程式庫用於升級套件。

> **Note**
>
> Patch Manager 不支援使用 APT `Pin-Priority` 選項，將優先順序指派給套件。Patch Manager 會從所有已啟用的儲存庫彙總可用的更新，並選取符合每個已安裝套件基準的最新更新。

8. 如有安裝任何更新，受管節點將會重新啟動。(例外：如果 `AWS-RunPatchBaseline` 文件中的 `RebootOption` 參數設為 `NoReboot`，受管節點不會在 Patch Manager 執行之後重新啟動。如需詳細資訊，請參閱 參數名稱：`RebootOption`。)

macOS

在 macOS 受管節點上，修補程式安裝工作流程如下：

1. /Library/Receipts/InstallHistory.plist 屬性清單是使用 softwareupdate 和 installer 套件管理工具的已安裝和已升級軟體記錄。使用 pkgutil 命令列工具 (用於 installer) 以及 softwareupdate 套件管理工具時，會執行 CLI 命令來剖析此清單。

   對於 installer，對 CLI 命令的回應包括 package name、version、volume、location 和 install-time 詳細資訊，但 Patch Manager 僅使用 package name 和 version。

   對於 softwareupdate，對 CLI 命令的回應會包含套件名稱 (display name)、version 和 date，但 Patch Manager 只會使用套件名稱和版本。

   對於 Brew 和 Brew Cask，Homebrew 不支援其在根使用者下執行的命令。因此，Patch Manager 以 Homebrew 目錄的擁有者或屬於 Homebrew 目錄之擁有者群組的有效使用者身分查詢和執行 Homebrew 命令。這些命令類似於 softwareupdate 和 installer 並透過 Python 子程序執行命令，以收集套件資料，並解析輸出以識別套件名稱和版本。
2. 套用修補程式基準中指定的 GlobalFilters，以便僅進一步處理合格的套件。
3. 套用修補程式基準中指定的 ApprovalRules。每個核准規則皆可將套件定義為已核准。
4. 套用修補程式基準中指定的 ApprovedPatches。已核准的修補程式即使被 GlobalFilters 捨棄或 ApprovalRules 中沒有指定核准規則授予其核准，仍將會核准用於更新。
5. 套用修補程式基準中指定的 RejectedPatches。已遭拒的修補程式會從核准的修補程式清單中移除，而且將不會套用。
6. 如果核准修補程式的多個版本，將會套用最新版本。
7. 在受管節點上叫用適當的套件 CLI，以處理核准的修補程式，如下所示：

   > **Note**
   >
   > installer 缺乏檢查和安裝更新的功能。因此，對於 installer，Patch Manager 只會報告已安裝的套件。因此，installer 套件永遠不會報告為 Missing。

   - 針對 AWS提供的預先定義預設修補基準，以及未選取包含非安全性更新核取方塊的自訂修補基準，則只會套用安全性更新。
   - 針對已選取包含非安全性更新核取方塊的自訂修補基準，會套用安全性和非安全性更新。

8. 如有安裝任何更新，受管節點將會重新啟動。(例外：如果 AWS-RunPatchBaseline 文件中的 RebootOption 參數設為 NoReboot，受管節點不會在 Patch Manager 執行之後重新啟動。如需詳細資訊，請參閱 參數名稱：RebootOption。)

Oracle Linux

在 Oracle Linux 受管節點上，修補程式安裝工作流程如下：

1. 如果使用 https URL 或 Amazon Simple Storage Service (Amazon S3) 路徑樣式 URL (使用 AWS-RunPatchBaseline 或 AWS-RunPatchBaselineAssociation 文件的 InstallOverrideList 參數) 指定修補程式清單，系統會安裝列出的修補程式，並略過步驟 2-7。
2. 套用修補程式基準中指定的 GlobalFilters，以便僅進一步處理合格的套件。
3. 套用修補程式基準中指定的 ApprovalRules。每個核准規則皆可將套件定義為已核准。

   然而，核准規則也受限於建立或最後更新修補基準時，是否已選取 Include nonsecurity updates (包含非安全性更新) 核取方塊。

   如果非安全更新被排除，將會套用隱含規則，以僅選擇安全儲存庫中包含升級的套件。對於每個套件，套件的候選版本 (通常是最新版本) 必須是安全儲存庫的一部分。

   如果包含非安全性更新，則也會考慮來自其他儲存庫的修補程式。
4. 套用修補程式基準中指定的 ApprovedPatches。已核准的修補程式即使被 GlobalFilters 捨棄或 ApprovalRules 中沒有指定核准規則授予其核准，仍將會核准用於更新。
5. 套用修補程式基準中指定的 RejectedPatches。已遭拒的修補程式會從核准的修補程式清單中移除，而且將不會套用。
6. 如果核准修補程式的多個版本，將會套用最新版本。
7. 在版本 7 受管節點上，YUM 更新 API 會套用至核准的修補程式，如下所示：
   - 針對 AWS提供的預先定義預設修補基準，以及未選取包含非安全性更新核取方塊的自訂修補基準，則只會套用 updateinfo.xml 中指定的修補程式 (僅限安全性更新)。

     此工作流程的同等 yum 命令為：

     ```
     sudo yum update-minimal --sec-severity=Important,Moderate --bugfix -y
     ```
   - 針對已選取包含非安全性更新核取方塊的修補基準，位於 updateinfo.xml 中的修補程式和不位於 updateinfo.xml 中的修補程式皆會套用 (安全性和非安全性更新)。

此工作流程的同等 yum 命令為：

```
sudo yum update --security --bugfix -y
```

在版本 8 和 9 的受管節點上，DNF 更新 API 會套用至核准的修補程式，如下所示：

- 對於預先定義的預設修補程式基準 AWS，以及未選取包含非安全性更新核取方塊的自訂修補程式基準，只會套用 中指定的修補程式 `updateinfo.xml` （僅安全性更新）。

  此工作流程的同等 yum 命令為：

  ```
  sudo dnf upgrade-minimal --security --sec-severity=Moderate --sec-severity=Important
  ```

  > **Note**
  >
  > 對於 Oracle Linux， Patch Manager可能會安裝與同等`dnf`命令安裝不同的傳輸相依性版本。暫時性相依性是會自動安裝的套件，以滿足其他套件的需求 （相依性）。例如， `dnf upgrade-minimal --security` 安裝解決已知安全問題所需的最低傳輸相依性版本，而 Patch Manager 安裝相同傳輸相依性的最新可用版本。

- 針對已選取包含非安全性更新核取方塊的修補基準，位於 `updateinfo.xml` 中的修補程式和不位於 `updateinfo.xml` 中的修補程式皆會套用 (安全性和非安全性更新)。

  此工作流程的同等 yum 命令為：

  ```
  sudo dnf upgrade --security --bugfix
  ```

> **Note**
>
> 如果您在 外部執行這些 `yum`或 `dnf`命令，則會安裝以不同名稱取代現已淘汰套件的新套件Patch Manager。不過，對等Patch Manager操作不會安裝它們。

8. 如有安裝任何更新，受管節點將會重新啟動。(例外：如果 `AWS-RunPatchBaseline` 文件中的 `RebootOption` 參數設為 `NoReboot`，受管節點不會在 Patch Manager 執行之後重新啟動。如需詳細資訊，請參閱 參數名稱：RebootOption。)

AlmaLinux, RHEL, and Rocky Linux

在 AlmaLinux、Red Hat Enterprise Linux 和 Rocky Linux 受管節點上，修補程式安裝工作流程如下：

1. 如果使用 https URL 或 Amazon Simple Storage Service (Amazon S3) 路徑樣式 URL (使用 `AWS-RunPatchBaseline` 或 `AWS-RunPatchBaselineAssociation` 文件的 `InstallOverrideList` 參數) 指定修補程式清單，系統會安裝列出的修補程式，並略過步驟 2-7。
2. 套用修補程式基準中指定的 GlobalFilters，以便僅進一步處理合格的套件。
3. 套用修補程式基準中指定的 ApprovalRules。每個核准規則皆可將套件定義為已核准。

   然而，核准規則也受限於建立或最後更新修補基準時，是否已選取 Include nonsecurity updates (包含非安全性更新) 核取方塊。

   如果非安全更新被排除，將會套用隱含規則，以僅選擇安全儲存庫中包含升級的套件。對於每個套件，套件的候選版本 (通常是最新版本) 必須是安全儲存庫的一部分。

   如果包含非安全性更新，則也會考慮來自其他儲存庫的修補程式。
4. 套用修補程式基準中指定的 ApprovedPatches。已核准的修補程式即使被 GlobalFilters 捨棄或 ApprovalRules 中沒有指定核准規則授予其核准，仍將會核准用於更新。
5. 套用修補程式基準中指定的 RejectedPatches。已遭拒的修補程式會從核准的修補程式清單中移除，而且將不會套用。
6. 如果核准修補程式的多個版本，將會套用最新版本。
7. YUM 更新 API （在 7 RHEL 上） 或 DNF 更新 API （在 AlmaLinux 8 和 9、RHEL8、9 和 10 以及 Rocky Linux 8 和 9 上） 會根據下列規則套用至核准的修補程式：

   案例 1：排除非安全性更新

   - 適用於： 提供的預先定義預設修補程式基準 AWS 和自訂修補程式基準。
   - 包含非安全性更新核取方塊：未選取。
   - 套用的修補程式：只有在 `updateinfo.xml` （僅限安全性更新） 中指定的修補程式都符合修補程式基準組態，且在設定的儲存庫中找到時，才會套用這些修補程式。

在某些情況下，在 中指定的修補程式updateinfo.xml可能無法再在設定的儲存庫中使用。設定的儲存庫通常只有最新版本的修補程式，這是所有先前更新的累積彙總，但最新版本可能不符合修補程式基準規則，並從修補操作中省略。

- 命令：對於 RHEL 7，此工作流程的同等 yum 命令為：

```
sudo yum update-minimal --sec-severity=Critical,Important --bugfix -y
```

對於 AlmaLinux、RHEL 8 和 Rocky Linux，此工作流程的同等 dnf 命令為：

```
sudo dnf update-minimal --sec-severity=Critical --bugfix -y ; \
sudo dnf update-minimal --sec-severity=Important --bugfix -y
```

> **Note**
>
> 對於 AlmaLinux、 RHEL和 Rocky LinuxRocky Linux， Patch Manager可能會安裝與同等dnf命令安裝不同的傳輸相依性版本。暫時性相依性是會自動安裝的套件，以滿足其他套件的需求（相依性）。
> 例如， `dnf upgrade-minimal --security` 安裝解決已知安全問題所需的最低傳輸相依性版本，同時 Patch Manager 安裝相同傳輸相依性的最新可用版本。

案例 2：包含非安全性更新

- Apples to：自訂修補基準。
- 包含非安全性更新核取方塊：已選取。
- 套用的修補程式：套用 中的修補程式updateinfo.xml，以及未套用 中的修補程式updateinfo.xml（安全性和非安全性更新）。
- 命令：對於 RHEL 7，此工作流程的同等 yum 命令為：

```
sudo yum update --security --bugfix -y
```

對於 AlmaLinux 8 和 9、RHEL 8 和 9 以及 Rocky Linux 8 和 9，此工作流程的同等 dnf 命令為：

```
sudo dnf update --security --bugfix -y
```

> **Note**
>
> 如果您在 外部執行這些 yum或 dnf命令，則會安裝以不同名稱取代現已淘汰套件的新套件Patch Manager。不過，對等Patch Manager操作不會安裝它們。

8. 如有安裝任何更新，受管節點將會重新啟動。(例外：如果 AWS-RunPatchBaseline 文件中的 RebootOption 參數設為 NoReboot，受管節點不會在 Patch Manager 執行之後重新啟動。如需詳細資訊，請參閱 參數名稱：RebootOption。)

SLES

在 SUSE Linux Enterprise Server (SLES) 受管節點上，修補程式安裝工作流程如下：

1. 如果使用 https URL 或 Amazon Simple Storage Service (Amazon S3) 路徑樣式 URL (使用 AWS-RunPatchBaseline 或 AWS-RunPatchBaselineAssociation 文件的 InstallOverrideList 參數) 指定修補程式清單，系統會安裝列出的修補程式，並略過步驟 2-7。
2. 套用修補程式基準中指定的 GlobalFilters，以便僅進一步處理合格的套件。
3. 套用修補程式基準中指定的 ApprovalRules。每個核准規則皆可將套件定義為已核准。

   然而，核准規則也受限於建立或最後更新修補基準時，是否已選取 Include nonsecurity updates (包含非安全性更新) 核取方塊。

   如果非安全更新被排除，將會套用隱含規則，以僅選擇安全儲存庫中包含升級的套件。對於每個套件，套件的候選版本 (通常是最新版本) 必須是安全儲存庫的一部分。

   如果包含非安全性更新，則也會考慮來自其他儲存庫的修補程式。
4. 套用修補程式基準中指定的 ApprovedPatches。已核准的修補程式即使被 GlobalFilters 捨棄或 ApprovalRules 中沒有指定核准規則授予其核准，仍將會核准用於更新。
5. 套用修補程式基準中指定的 RejectedPatches。已遭拒的修補程式會從核准的修補程式清單中移除，而且將不會套用。
6. 如果核准修補程式的多個版本，將會套用最新版本。
7. Zypper 更新 API 將套用至核准的修補程式。
8. 如有安裝任何更新，受管節點將會重新啟動。(例外：如果 AWS-RunPatchBaseline 文件中的 RebootOption 參數設為 NoReboot，受管節點不會在 Patch Manager 執行之後重新啟動。如需詳細資訊，請參閱 參數名稱：RebootOption。)

Ubuntu Server

在 Ubuntu Server 受管節點上，修補程式安裝工作流程如下：

1. 如果使用 https URL 或 Amazon Simple Storage Service (Amazon S3) 路徑樣式 URL (使用 `AWS-RunPatchBaseline` 或 `AWS-RunPatchBaselineAssociation` 文件的 `InstallOverrideList` 參數) 指定修補程式清單，系統會安裝列出的修補程式，並略過步驟 2-7。
2. 若有可用於 `python3-apt` (`libapt` 的 Python 程式庫界面) 的更新，它會升級到最新版本。(這個非安全性套件會升級，即使您未選取 Include nonsecurity updates (包含非安全性更新) 選項)。
3. 套用修補程式基準中指定的 GlobalFilters，以便僅進一步處理合格的套件。
4. 套用修補程式基準中指定的 ApprovalRules。每個核准規則皆可將套件定義為已核准。

> **Note**
>
> 因為無法可靠地判斷 Ubuntu Server 更新套件的發行日期，此作業系統不支援自動核准選項。

然而，核准規則也受限於建立或最後更新修補基準時，是否已選取 Include nonsecurity updates (包含非安全性更新) 核取方塊。

如果非安全更新被排除，將會套用隱含規則，以僅選擇安全儲存庫中包含升級的套件。對於每個套件，套件的候選版本 (通常是最新版本) 必須是安全儲存庫的一部分。

如果包含非安全性更新，則也會考慮來自其他儲存庫的修補程式。

然而，核准規則也受限於建立或最後更新修補基準時，是否已選取 Include nonsecurity updates (包含非安全性更新) 核取方塊。

> **Note**
>
> 對於每個 Ubuntu Server 版本，修補程式候選版本僅限於屬於該版本關聯儲存庫的修補程式，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. 套用修補程式基準中指定的 ApprovedPatches。已核准的修補程式即使被 GlobalFilters 捨棄或 ApprovalRules 中沒有指定核准規則授予其核准，仍將會核准用於更新。
6. 套用修補程式基準中指定的 RejectedPatches。已遭拒的修補程式會從核准的修補程式清單中移除，而且將不會套用。
7. APT 程式庫用於升級套件。

> **Note**
>
> Patch Manager 不支援使用 APT `Pin-Priority` 選項，將優先順序指派給套件。Patch Manager 會從所有已啟用的儲存庫彙總可用的更新，並選取符合每個已安裝套件基準的最新更新。

8. 如有安裝任何更新，受管節點將會重新啟動。(例外：如果 `AWS-RunPatchBaseline` 文件中的 `RebootOption` 參數設為 `NoReboot`，受管節點不會在 Patch Manager 執行之後重新啟動。如需詳細資訊，請參閱 參數名稱：`RebootOption`。)

Windows Server

在 Windows Server 受管節點上執行修補操作時，節點會從 Systems Manager 請求適當修補基準的快照。此快照包含已核准部署之修補基準中所有可用更新的清單。此更新清單會傳送至 Windows Update API，決定哪些更新適用於受管節點並視需要安裝這些更新。Windows 只允許安裝最新版本的 KB。如果最新版本的 KB 或任何舊版 KB 符合套用的修補基準，Patch Manager 會安裝最新版本的 KB。如有安裝任何更新，受管節點將會重新啟動，重新啟動的次數依據完成所有必要修補的需要而定。(例外：如果 `AWS-RunPatchBaseline` 文件中的 `RebootOption` 參數設為 `NoReboot`，受管節點不會在 Patch Manager 執行之後重新啟動。如需詳細資訊，請參閱 參數名稱：`RebootOption`。) 在 Run Command 請求的輸出中，可以找到修補操作的摘要。在

%PROGRAMDATA%\Amazon\PatchBaselineOperations\Logs 資料夾的受管節點上，可以找到額外的日誌。

由於 Windows Update API 用於下載和安裝 KB，因此會遵守適用於 Windows Update 的所有群組政策設定。使用 Patch Manager 無需任何群組政策設定，但您已定義的所有設定都將會套用，例如將受管節點導向至 Windows Server Update Services (WSUS) 伺服器。

> **Note**
>
> 依預設，Windows 會從 Microsoft 的 Windows Update 網站中下載所有 KB，因為 Patch Manager 使用 Windows Update API 來推動下載和安裝 KB。因此，受管節點必須能夠連接至 Microsoft Windows Update 網站，否則修補將會失敗。或者，您可以設定 WSUS 伺服器作為 KB 儲存庫，並設定您的受管節點以 WSUS 伺服器為目標使用群組政策。
> Patch Manager 可能會在建立 的自訂修補程式基準時參考 KB IDsWindows Server，例如當基準組態中包含已核准修補程式清單或拒絕修補程式清單時。只有 Microsoft Windows Update 或 WSUS 伺服器中指派 KB ID 的更新，才會由 安裝Patch Manager。缺少 KB ID 的更新不會包含在修補操作中。
> 如需建立自訂修補程式基準的資訊，請參閱下列主題：
>
> - 建立 Windows Server 的自訂修補基準
> - 建立修補程式基準 (CLI)
> - 適用於 Windows 作業系統的套件名稱格式

修補基準規則在 Linux 系統上的運作方式

Linux 分發的修補基準中的規則，運作方式依據分發類型而有不同。與Windows Server受管節點上的修補程式更新不同，每個節點都會評估規則，以考量執行個體上設定的儲存庫。 Patch Manager是 中的工具 AWS Systems Manager，使用原生套件管理員來推動修補程式基準核准的修補程式安裝。

對於報告修補程式嚴重性級別的 Linux 作業系統類型，Patch Manager 使用軟體發布者為更新通知或個別修補程式報告的嚴重性級別。Patch Manager 不會從第三方來源推導出嚴重性級別，例如通用漏洞評分系統 (CVSS)，或者美國國家漏洞資料庫 (NVD) 發行的指標。

主題

- 修補基準規則在 Amazon Linux 1、Amazon Linux 2、Amazon Linux 2022 和 Amazon Linux 2023 上的運作方式
- 修補基準規則在 CentOS 和 CentOS Stream 上的運作方式

- 修補基準規則在 Debian Server 和 Raspberry Pi OS 上的運作方式
- 修補基準規則在 macOS 上的運作方式
- 修補基準規則在 Oracle Linux 上的運作方式
- 修補基準規則在 AlmaLinux、RHEL 和 Rocky Linux 上的運作方式
- 修補基準規則在 SUSE Linux Enterprise Server 上的運作方式
- 修補基準規則在 Ubuntu Server 上的運作方式

修補基準規則在 Amazon Linux 1、Amazon Linux 2、Amazon Linux 2022 和 Amazon Linux 2023 上的運作方式

> **Note**
>
> Amazon Linux 2023 (AL2023) 使用可透過一或多個系統設定鎖定至特定版本的版本控制儲存庫。對於 AL2023 EC2 執行個體上的所有修補操作， Patch Manager會使用與系統組態無關的最新儲存庫版本。如需詳細資訊，請參閱《Amazon Linux 2023 使用者指南》中的透過版本化儲存庫使用決定性升級一節。

在 Amazon Linux 1、Amazon Linux 2、Amazon Linux 2022 和 Amazon Linux 2023 上，修補程式選取程序如下：

1. 在受管節點上，YUM 程式庫 (Amazon Linux 1 和 Amazon Linux 2) 或 DNF 程式庫 (Amazon Linux 2022 和 Amazon Linux 2023) 會存取每個已設定儲存庫的 `updateinfo.xml` 檔案。

   如果沒有找到 `updateinfo.xml` 檔案，是否安裝修補程式取決於包含非安全性更新和自動核准的設定。例如，如果允許非安全性更新，則會在自動核准時間到達時進行安裝。
2. `updateinfo.xml` 中的每個更新通知皆包含數個屬性，以表示通知中的套件的屬性，如下表所述。

   更新通知屬性

屬性	描述
type	對應至修補程式基準之 PatchFilter 資料類型中的分類金鑰屬性的值。表示包含在更新通知中的套件類型。

屬性	描述
	您可以使用 AWS CLI 命令describe-patch-properties或 API 操作 來檢視支援的值清單DescribePatchProperties。您也可以在 Systems Manager 主控台之 Create patch baseline (建立修補基準) 頁面或 Edit patch baseline (編輯修補基準) 頁面的 Approval rules (核准規則) 區域中檢視清單。
severity	對應至修補程式基準之 PatchFilter 資料類型中的嚴重性金鑰屬性的值。表示包含在更新通知中的套件嚴重性。通常僅適用於安全性更新通知。 您可以使用 AWS CLI 命令describe-patch-properties或 API 操作 來檢視支援的值清單DescribePatchProperties。您也可以在 Systems Manager 主控台之 Create patch baseline (建立修補基準) 頁面或 Edit patch baseline (編輯修補基準) 頁面的 Approval rules (核准規則) 區域中檢視清單。
update_id	表示諮詢 ID，例如 ALAS-2017-867。諮詢 ID 可用於修補程式基準中的 ApprovedPatches 或 RejectedPatches 屬性。
參考	包含有關更新通知的額外資訊，例如 CVE ID (格式：CVE-2017-1234567)。CVE ID 可用於修補程式基準中的 ApprovedPatches 或 RejectedPatches 屬性。
已更新	對應至修補程式基準中的 ApproveAfterDays。表示包含在更新通知中的發行日期 (更新日期)。目前時間戳記與此屬性值加上 `ApproveAfterDays` 之間的比較，可用於判斷修補程式是否已核准部署。

如需已核准修補程式和遭拒的修補程式清單之可接受格式的相關資訊，請參閱 已核准與遭拒的修補程式清單的套件名稱格式。

3. 受管節點的產物取決於 SSM Agent。此屬性對應至修補程式基準之 PatchFilter 資料類型中的產品金鑰屬性的值。
4. 已根據以下指導方針選取欲更新之套件：

安全性選項	修補程式選擇

AWS 提供的預先定義預設修補基準和自訂修補基準，其中未選取包含非安全性更新核取方塊	對於 `updateinfo.xml` 中的每個更新通知，修補基準做為篩選條件使用，只允許更新中包含合格的套件。如果在套用修補基準定義後，有多個套件可以適用，將使用最新的版本。
	
	針對 Amazon Linux 1 和 Amazon Linux 2，此工作流程的同等 yum 命令為：
	
	`sudo yum update-minimal --sec-severity=Critical,Important --bugfix -y`
	
	針對 Amazon Linux 2022 和 Amazon Linux 2023，此工作流程的同等 yum 命令為：
	
	`sudo dnf upgrade-minimal --sec-severity=Critical --sec-severity=Important --bugfix -y`

自訂修補程式基準，其中已選取包含非安全性更新核取方塊，且嚴重性清單為 `[Critical, Important]` ，分類清單為 `[Security, Bugfix]`	除了套用從 `updateinfo.xml` 中選擇的安全性更新外，Patch Manager 也將套用符合修補程式篩選規則的非安全性更新。 針對 Amazon Linux 和 Amazon Linux 2，此工作流程的同等 yum 命令為：

安全性選項	修補程式選擇

	`sudo yum update --security --sec-severity=Critical,Important --bugfix -y`
	
	針對 Amazon Linux 2022 和 Amazon Linux 2023，此工作流程的同等 yum 命令為：
	
	`sudo dnf upgrade --security --sec-severity=Critical --sec-severity=Important --bugfix -y`
	
	Note
	如果您在 外部執行這些 yum或 dnf命令，則會安裝以不同名稱取代現已淘汰套件的新套件Patch Manager。不過，對等Patch Manager操作不會安裝它們。

如需有關修補程式合規狀態值的詳細資訊，請參閱 修補程式合規狀態值。

修補基準規則在 CentOS 和 CentOS Stream 上的運作方式

CentOS 和 CentOS Stream 預設儲存庫不包含 `updateinfo.xml` 檔案。不過，您建立或使用的自訂儲存庫可能包含此檔案。在本主題中，`updateinfo.xml` 的參考僅適用於這些自訂儲存庫。

在 CentOS 和 CentOS Stream 上，修補程式選擇程序如下：

1. 在受管節點上，YUM 程式庫 (在 CentOS 6.x 和 7.x 版本上) 或 DNF 程式庫 (在 CentOS 8.x 和 CentOS Stream 上) 會存取每個已設定儲存庫的 `updateinfo.xml` 檔案 (如果自訂儲存庫中存在該檔案)。

   如果沒有找到一律包含預設儲存庫的 `updateinfo.xml`，是否安裝修補程式取決於包含非安全性更新和自動核准的設定。例如，如果允許非安全性更新，則會在自動核准時間到達時進行安裝。

2. 如果存在 `updateinfo.xml`，則在檔案中的每個更新通知皆包含數個屬性，以表示通知中的套件的屬性，如下表所述。

   更新通知屬性

屬性	描述
type	對應至修補程式基準之 PatchFilter 資料類型中的分類金鑰屬性的值。表示包含在更新通知中的套件類型。 您可以使用 AWS CLI 命令describe-patch-properties或 API 操作 來檢視支援的值清單DescribePatchProperties。您也可以在 Systems Manager 主控台之 Create patch baseline (建立修補基準) 頁面或 Edit patch baseline (編輯修補基準) 頁面的 Approval rules (核准規則) 區域中檢視清單。
severity	對應至修補程式基準之 PatchFilter 資料類型中的嚴重性金鑰屬性的值。表示包含在更新通知中的套件嚴重性。通常僅適用於安全性更新通知。 您可以使用 AWS CLI 命令describe-patch-properties或 API 操作 來檢視支援的值清單DescribePatchProperties。您也可以在 Systems Manager 主控台之 Create patch baseline (建立修補基準) 頁面或 Edit patch baseline (編輯修補基準) 頁面的 Approval rules (核准規則) 區域中檢視清單。
update_id	表示諮詢 ID，例如 CVE-2019-17055。諮詢 ID 可用於修補程式基準中的 ApprovedPatches 或 RejectedPatches 屬性。

屬性	描述
參考	包含有關更新通知的額外資訊，例如 CVE ID (格式：CVE-2019-17055) 或 Bugzilla ID (格式：1463241)。CVE ID 與 Bugzilla ID 可用於修補程式基準中的 ApprovedPatches 或 RejectedPatches 屬性。
已更新	對應至修補程式基準中的 ApproveAfterDays。表示包含在更新通知中的發行日期 (更新日期)。目前時間戳記與此屬性值加上 `ApproveAfterDays` 之間的比較，可用於判斷修補程式是否已核准部署。

如需已核准修補程式和遭拒的修補程式清單之可接受格式的相關資訊，請參閱 已核准與遭拒的修補程式清單的套件名稱格式。

3. 在任何情況下，受管節點的產物取決於 SSM Agent。此屬性對應至修補程式基準之 PatchFilter 資料類型中的產品金鑰屬性的值。
4. 已根據以下指導方針選取欲更新之套件：

安全性選項	修補程式選擇
AWS 提供的預先定義預設修補基準和自訂修補基準，其中未選取包含非安全性更新核取方塊	對於 `updateinfo.xml` 中的每個更新通知，如果它存在於自訂儲存庫中，則修補基準將用作篩選條件，僅允許更新中包含合格的套件。如果在套用修補基準定義後，有多個套件可以適用，將使用最新的版本。 對於存在 `updateinfo.xml` 的 CentOS 6 和 7，此工作流程的等效 yum 命令為： `sudo yum update-minimal --sec-severity=Critical,Important --bugfix -y`

安全性選項	修補程式選擇
	對於存在 `updateinfo.xml` 的 CentOS 8 和 CentOS Stream，此工作流程的等效 yum 命令為： `sudo dnf upgrade-minimal --sec-severity=Critical --sec-severity=Important --bugfix -y`

安全性選項	修補程式選擇

自訂修補程式基準，其中已選取包含非安全性更新核取方塊，且嚴重性清單為 [Critical, Important] ，分類清單為 [Security, Bugfix]	除了套用從 updateinfo.xml 中選取的安全性更新外 (如果它存在於自訂儲存庫中)，Patch Manager 也將套用符合修補程式篩選規則的非安全性更新。
	
	對於存在 updateinfo.xml 的 CentOS 6 和 7，此工作流程的等效 yum 命令為：
	
	`sudo yum update --sec-severity=Critical,Important --bugfix -y`
	
	對於存在 updateinfo.xml 的 CentOS 8 和 CentOS Stream，此工作流程的等效 yum 命令為：
	
	`sudo dnf upgrade --security --sec-severity=Critical --sec-severity=Important --bugfix -y`
	
	對於沒有 updateinfo.xml 的預設儲存庫和自訂儲存庫，您必須選取包含非安全性更新核取方塊，以更新作業系統 (OS) 套件。
	
	Note
	如果您在 外部執行這些 yum或 dnf命令，則會安裝以不同名稱取代現已淘汰套件的新套件Patch Manager。不過，對等Patch Manager操作不會安裝它們。

如需有關修補程式合規狀態值的詳細資訊，請參閱 修補程式合規狀態值。

修補基準規則在 Debian Server 和 Raspberry Pi OS 上的運作方式

在 Debian Server 和 Raspberry Pi OS 上 (之前為 Raspbian)，修補基準服務提供篩選 Priority (優先順序) 與 Section (區段) 欄位操作。這些欄位通常會顯示所有 Debian Server 和 Raspberry Pi OS 套件。為了判斷修補程式是否已被修補基準選取，Patch Manager 會執行下列動作：

1. 在 Debian Server 和 Raspberry Pi OS 系統上，會執行與 `sudo apt-get update` 相當的命令以重新整理可用套件清單。儲存區未設定，資料從設定於 `sources` 清單中的儲存區提取。
2. 若有可用於 `python3-apt` (`libapt` 的 Python 程式庫界面) 的更新，它會升級到最新版本。(這個非安全性套件會升級，即使您未選取 Include nonsecurity updates (包含非安全性更新) 選項)。

> **Important**
>
> 僅限於 Debian Server 8 上：由於 Debian Server 8.* 作業系統會參考已淘汰的套件庫 (`jessie-backports`)，因此 Patch Manager 會執行下列額外的步驟以確保修補操作成功。
>
> a. 在您的受管節點上，對 `jessie-backports` 儲存庫的參考從來源位置清單 (`/etc/apt/sources.list.d/jessie-backports`) 會變更為註解。因此，不會嘗試從該位置下載修補程式，
>
> b. 並會匯入延展安全性更新簽署金鑰。此金鑰提供了 Debian Server 8.* 發行版本的更新和安裝操作所需的許可。
>
> c. 系統此時會執行 `apt-get` 操作以確保在修補程式開始之前已安裝最新版本的 `python3-apt`。
>
> d. 安裝程序完成後，會還原對 `jessie-backports` 儲存庫的參考，並從 apt 來源金鑰環中移除簽署金鑰。這麼做是為了讓系統組態保持在修補操作之前的狀態。

3. 接著，將套用 GlobalFilters、ApprovalRules、ApprovedPatches 和 RejectedPatches 清單。

> **Note**
>
> 因為無法可靠地判斷 Debian Server 更新套件的發行日期，此作業系統不支援自動核准選項。

然而，核准規則也受限於建立或最後更新修補基準時，是否已選取 Include nonsecurity updates (包含非安全性更新) 核取方塊。

如果非安全更新被排除，將會套用隱含規則，以僅選擇安全儲存庫中包含升級的套件。對於每個套件，套件的候選版本 (通常是最新版本) 必須是安全儲存庫的一部分。在這種情形下，對於 Debian Server 來說，修補候選版本僅限於以下儲存庫中包含的修補程式：

這些儲存庫的命名如下：

- 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`

如果包含非安全性更新，則也會考慮來自其他儲存庫的修補程式。

如需已核准修補程式和已拒絕修補程式清單之可接受格式的相關資訊，請參閱 已核准與遭拒的修補程式清單的套件名稱格式。

若要檢視 Priority (優先順序) 和 Section (區段) 欄位的內容，請執行下列 `aptitude` 命令：

> **Note**
>
> 您可能必須先在 Debian Server 系統上安裝 Aptitude。

```
aptitude search -F '%p %P %s %t %V#' '~U'
```

在此命令的回應中，所有可升級的套裝將以此格式回報：

```
name, priority, section, archive, candidate version
```

如需有關修補程式合規狀態值的詳細資訊，請參閱 修補程式合規狀態值。

修補基準規則在 macOS 上的運作方式

在 macOS 上，修補程式選擇程序如下：

1. 在受管節點上，Patch Manager 會存取 `InstallHistory.plist` 檔案已剖析的內容，並識別套件名稱和版本。

如需剖析程序的詳細資訊，請參閱 如何安裝修補程式 中的 macOS 區段。

2. 受管節點的產物取決於 SSM Agent。此屬性對應至修補程式基準之 PatchFilter 資料類型中的產品金鑰屬性的值。
3. 已根據以下指導方針選取欲更新之套件：

安全性選項	修補程式選擇
AWS 提供的預先定義預設修補基準和自訂修補基準，其中未選取包含非安全性更新核取方塊	對於每個可用套件更新，修補基準做為篩選條件使用，只允許更新中包含合格的套件。如果在套用修補基準定義後，有多個套件可以適用，將使用最新的版本。
已選取包含非安全性更新核取方塊的自訂修補基準。	除了套用使用 `InstallHistory.plist` 進行識別的安全性更新外，修補程式管理員也將套用符合修補程式篩選規則的非安全性更新。

如需有關修補程式合規狀態值的詳細資訊，請參閱 修補程式合規狀態值。

修補基準規則在 Oracle Linux 上的運作方式

在 Oracle Linux 上，修補程式選擇程序如下：

1. 在受管節點上，YUM 程式庫存取每個已設定之儲存庫的 `updateinfo.xml` 檔案。

> **Note**
>
> 如果儲存庫不是由 Oracle 管理的，則可能沒有 `updateinfo.xml` 檔案。如果沒有找到 `updateinfo.xml`，是否安裝修補程式取決於包含非安全性更新和自動核准的設定。例如，如果允許非安全性更新，則會在自動核准時間到達時進行安裝。

2. `updateinfo.xml` 中的每個更新通知皆包含數個屬性，以表示通知中的套件的屬性，如下表所述。

更新通知屬性

屬性	描述
type	對應至修補程式基準之 PatchFilter 資料類型中的分類金鑰屬性的值。表示包含在更新通知中的套件類型。 您可以使用 AWS CLI 命令describe-patch-properties或 API 操作 來檢視支援的值清單DescribePatchProperties。您也可以在 Systems Manager 主控台之 Create patch baseline (建立修補基準) 頁面或 Edit patch baseline (編輯修補基準) 頁面的 Approval rules (核准規則) 區域中檢視清單。
severity	對應至修補程式基準之 PatchFilter 資料類型中的嚴重性金鑰屬性的值。表示包含在更新通知中的套件嚴重性。通常僅適用於安全性更新通知。 您可以使用 AWS CLI 命令describe-patch-properties或 API 操作 來檢視支援的值清單DescribePatchProperties。您也可以在 Systems Manager 主控台之 Create patch baseline (建立修補基準) 頁面或 Edit patch baseline (編輯修補基準) 頁面的 Approval rules (核准規則) 區域中檢視清單。
update_id	表示諮詢 ID，例如 CVE-2019-17055。諮詢 ID 可用於修補程式基準中的 ApprovedPatches 或 RejectedPatches 屬性。
參考	包含有關更新通知的額外資訊，例如 CVE ID (格式：CVE-2019-17055) 或 Bugzilla ID (格式：1463241)。CVE ID 與 Bugzilla ID 可用於修補程式基準中的 ApprovedPatches 或 RejectedPatches 屬性。

屬性	描述
已更新	對應至修補程式基準中的 ApproveAfterDays。表示包含在更新通知中的發行日期 (更新日期)。目前時間戳記與此屬性值加上 ApproveAfterDays 之間的比較，可用於判斷修補程式是否已核准部署。

如需已核准修補程式和遭拒的修補程式清單之可接受格式的相關資訊，請參閱 已核准與遭拒的修補程式清單的套件名稱格式。

3. 受管節點的產物取決於 SSM Agent。此屬性對應至修補程式基準之 PatchFilter 資料類型中的產品金鑰屬性的值。
4. 已根據以下指導方針選取欲更新之套件：

安全性選項	修補程式選擇

AWS 提供的預先定義預設修補基準和自訂修補基準，其中未選取包含非安全性更新核取方塊	對於 updateinfo.xml 中的每個更新通知，修補基準做為篩選條件使用，只允許更新中包含合格的套件。如果在套用修補基準定義後，有多個套件可以適用，將使用最新的版本。
	
	對於版本 7 受管節點，此工作流程的同等 yum 命令為：
	
	`sudo yum update-minimal --sec-severity=Important,Moderate --bugfix -y`
	
	對於版本 8 和 9 受管節點，此工作流程的同等 dnf 命令為：
	
	`sudo dnf upgrade-minimal --security --sec-severity=Moderate --sec-severity=Important`

安全性選項	修補程式選擇

自訂修補程式基準，其中已選取包含非安全性更新核取方塊，且嚴重性清單為 [Critical, Important] ，分類清單為 [Security, Bugfix]	除了套用從 updateinfo.xml 中選擇的安全性更新外，Patch Manager 也將套用符合修補程式篩選規則的非安全性更新。
	
	對於版本 7 受管節點，此工作流程的同等 yum 命令為：
	
	`sudo yum update --security --sec-severity=Critical,Important --bugfix -y`
	
	對於版本 8 和 9 受管節點，此工作流程的同等 dnf 命令為：
	
	`sudo dnf upgrade --security --sec-severity=Critical, --sec-severity=Important --bugfix y`
	
	Note
	如果您在 外部執行這些 yum或 dnf命令，則會安裝以不同名稱取代現已淘汰套件的新套件Patch Manager。不過，對等Patch Manager操作不會安裝它們。

如需有關修補程式合規狀態值的詳細資訊，請參閱 修補程式合規狀態值。

修補基準規則在 AlmaLinux、RHEL 和 Rocky Linux 上的運作方式

在 AlmaLinux、Red Hat Enterprise Linux (RHEL) 和 Rocky Linux 上，修補程式選擇程序如下：

1. 在受管節點上，YUM 程式庫 (RHEL 7) 或 DNF 程式庫 RHEL (AlmaLinux 8 和 9、8、9 和 10，以及 Rocky Linux 8 和 9) 會存取每個設定儲存庫updateinfo.xml的檔案。

> **Note**
>
> 如果儲存庫不是由 Red Hat 管理的，則可能沒有 `updateinfo.xml` 檔案。如果沒有 `updateinfo.xml`，將不會套用任何修補程式。

2. `updateinfo.xml` 中的每個更新通知皆包含數個屬性，以表示通知中的套件的屬性，如下表所述。

更新通知屬性

屬性	描述
type	對應至修補程式基準之 PatchFilter 資料類型中的分類金鑰屬性的值。表示包含在更新通知中的套件類型。 您可以使用 AWS CLI 命令describe-patch-properties或 API 操作 來檢視支援的值清單DescribePatchProperties。您也可以在 Systems Manager 主控台之 Create patch baseline (建立修補基準) 頁面或 Edit patch baseline (編輯修補基準) 頁面的 Approval rules (核准規則) 區域中檢視清單。
severity	對應至修補程式基準之 PatchFilter 資料類型中的嚴重性金鑰屬性的值。表示包含在更新通知中的套件嚴重性。通常僅適用於安全性更新通知。 您可以使用 AWS CLI 命令describe-patch-properties或 API 操作 來檢視支援的值清單DescribePatchProperties。您也可以在 Systems Manager 主控台之 Create patch baseline (建立修補基準) 頁面或 Edit patch baseline (編輯修補基準) 頁面的 Approval rules (核准規則) 區域中檢視清單。

屬性	描述
update_id	表示諮詢 ID，例如 RHSA-2017:0864。諮詢 ID 可用於修補程式基準中的 ApprovedPatches 或 RejectedPatches 屬性。
參考	包含有關更新通知的額外資訊，例如 CVE ID (格式：CVE-2017-1000371) 或 Bugzilla ID (格式：1463241)。CVE ID 與 Bugzilla ID 可用於修補程式基準中的 ApprovedPatches 或 RejectedPatches 屬性。
已更新	對應至修補程式基準中的 ApproveAfterDays。表示包含在更新通知中的發行日期 (更新日期)。目前時間戳記與此屬性值加上 `ApproveAfterDays` 之間的比較，可用於判斷修補程式是否已核准部署。

如需已核准修補程式和遭拒的修補程式清單之可接受格式的相關資訊，請參閱 已核准與遭拒的修補程式清單的套件名稱格式。

3. 受管節點的產物取決於 SSM Agent。此屬性對應至修補程式基準之 PatchFilter 資料類型中的產品金鑰屬性的值。
4. 已根據以下指導方針選取欲更新之套件：

安全性選項	修補程式選擇
AWS 提供的預先定義預設修補基準和自訂修補基準，其中未在任何規則中選取包含非安全性更新核取方塊	對於 `updateinfo.xml` 中的每個更新通知，修補基準做為篩選條件使用，只允許更新中包含合格的套件。如果在套用修補基準定義後，有多個套件可以適用，將使用最新的版本。 針對 RHEL 7，此工作流程的同等 yum 命令為：

安全性選項	修補程式選擇

	`sudo yum update-minimal --sec-severity=Critical,Important --bugfix -y`
	
	對於 AlmaLinux 8 和 9、RHEL8、9 和 10，以及 Rocky Linux 8 和 9，此工作流程的同等 dnf 命令為：
	
	`sudo dnf upgrade-minimal --sec-severity=Critical --sec-severity=Important --bugfix -y`

安全性選項	修補程式選擇

自訂修補程式基準，其中已選取包含非安全性更新核取方塊，且嚴重性清單為 [Critical, Important] ，分類清單為 [Security, Bugfix]	除了套用從 updateinfo.xml 中選擇的安全性更新外，Patch Manager 也將套用符合修補程式篩選規則的非安全性更新。
	
	針對 RHEL 7，此工作流程的同等 yum 命令為：
	
	`sudo yum update --security --sec-severity=Critical,Important --bugfix -y`
	
	對於 AlmaLinux 8 和 9、RHEL8、9 和 10，以及 Rocky Linux 8 和 9，此工作流程的同等 dnf 命令為：
	
	`sudo dnf upgrade --sec-severity=Critical --sec-severity=Important --bugfix -y`
	
	Note
	如果您在 外部執行這些 yum或 dnf命令，則會安裝以不同名稱取代現已淘汰套件的新套件Patch Manager。不過，對等Patch Manager操作不會安裝它們。

如需有關修補程式合規狀態值的詳細資訊，請參閱 修補程式合規狀態值。

修補基準規則在 SUSE Linux Enterprise Server 上的運作方式

在 SLES 上，每個修補程式皆包含下列屬性，以表示修補程式中的套件的屬性：

- 分類：對應至修補基準之 PatchFilter 資料類型中的分類金鑰屬性的值。表示包含在更新通知中的修補程式類型。

您可以使用 AWS CLI 命令describe-patch-properties或 API 操作 來檢視支援的值清單DescribePatchProperties。您也可以在 Systems Manager 主控台之 Create patch baseline (建立修補基準) 頁面或 Edit patch baseline (編輯修補基準) 頁面的 Approval rules (核准規則) 區域中檢視清單。

- 嚴重性：對應至修補基準之 PatchFilter 資料類型中的嚴重性金鑰屬性的值。表示修補程式的嚴重性。

  您可以使用 AWS CLI 命令describe-patch-properties或 API 操作 來檢視支援的值清單DescribePatchProperties。您也可以在 Systems Manager 主控台之 Create patch baseline (建立修補基準) 頁面或 Edit patch baseline (編輯修補基準) 頁面的 Approval rules (核准規則) 區域中檢視清單。

受管節點的產物取決於 SSM Agent。此屬性對應至修補基準之 PatchFilter 資料類型中的產品金鑰屬性的值。

對於每個修補程式，修補基準做為篩選條件使用，只允許更新中包含合格的套件。如果在套用修補基準定義後，有多個套件可以適用，將使用最新的版本。

如需已核准修補程式和已拒絕修補程式清單之可接受格式的相關資訊，請參閱 已核准與遭拒的修補程式清單的套件名稱格式。

修補基準規則在 Ubuntu Server 上的運作方式

在 Ubuntu Server 上，修補基準服務提供篩選 Priority (優先順序) 與 Section (區段) 欄位。這些欄位通常會顯示所有 Ubuntu Server 套件。為了判斷修補程式是否已被修補基準選取，Patch Manager 會執行下列動作：

1. 在 Ubuntu Server 系統上，會執行與 `sudo apt-get update` 相當的命令以重新整理可用套件清單。儲存區未設定，資料從設定於 `sources` 清單中的儲存區提取。
2. 若有可用於 `python3-apt` (`libapt` 的 Python 程式庫界面) 的更新，它會升級到最新版本。(這個非安全性套件會升級，即使您未選取 Include nonsecurity updates (包含非安全性更新) 選項)。
3. 接著，將套用 GlobalFilters、ApprovalRules、ApprovedPatches 和 RejectedPatches 清單。

> **Note**
>
> 因為無法可靠地判斷 Ubuntu Server 更新套件的發行日期，此作業系統不支援自動核准選項。

然而，核准規則也受限於建立或最後更新修補基準時，是否已選取 Include nonsecurity updates (包含非安全性更新) 核取方塊。

如果非安全更新被排除，將會套用隱含規則，以僅選擇安全儲存庫中包含升級的套件。對於每個套件，套件的候選版本 (通常是最新版本) 必須是安全儲存庫的一部分。在這種情形下，對於 Ubuntu Server 來說，修補候選版本僅限於以下儲存庫中包含的修補程式：

- 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`)

如果包含非安全性更新，則也會考慮來自其他儲存庫的修補程式。

如需已核准修補程式和已拒絕修補程式清單之可接受格式的相關資訊，請參閱 已核准與遭拒的修補程式清單的套件名稱格式。

若要檢視 Priority (優先順序) 和 Section (區段) 欄位的內容，請執行下列 `aptitude` 命令：

> **Note**
>
> 您可能必須先在 Ubuntu Server 16 系統上安裝 Aptitude。

```
aptitude search -F '%p %P %s %t %V#' '~U'
```

在此命令的回應中，所有可升級的套裝將以此格式回報：

```
name, priority, section, archive, candidate version
```

如需有關修補程式合規狀態值的詳細資訊，請參閱 修補程式合規狀態值。

Linux 和 Windows Server 之間的修補操作差異

本主題說明 Linux 與 中的Windows Server修補之間的重要差異Patch Manager，這是 中的工具 AWS Systems Manager。

> **Note**
>
> 若要修補 Linux 受管節點，您的節點必須執行 SSM Agent 2.0.834.0 版或更新的版本。每當將新工具新增至 Systems Manager 或對現有工具進行更新時，SSM Agent就會發行更新的 版本。未使用最新版本的代理程式可能會使您的受管節點無法使用各種 Systems Manager 工具和功能。因此，我們建議您讓機器的 SSM Agent 自動保持最新狀態。如需相關資訊，請參閱 自動化 SSM Agent 更新。請訂閱 GitHub 上的 SSM Agent 版本備註頁面，以便接收有關 SSM Agent 更新的通知。

差異 1：修補程式評估

Patch Manager 會在 Windows 受管節點以及 Linux 受管節點上使用不同的程序，藉此評估哪些修補程式應該出現。

Linux

對於 Linux 修補，Systems Manager 會評估修補基準規則，以及每個受管節點上已核准與已拒絕修補程式的清單。Systems Manager 必須評估各個節點上的修補，因為服務會從受管節點上已設定的儲存庫擷取已知修補程式與更新的清單。

Windows

對於 Windows 修補，Systems Manager 會直接在服務中評估修補基準規則，以及已核准與已拒絕修補程式的清單。它可以這麼做，因為 Windows 修補程式皆來自於單一儲存庫 (Windows Update)。

差異 2：**`Not Applicable`** 修補程式

由於 Linux 作業系統有大量的可用套件，Systems Manager 不會報告狀態為不適用之修補程式的詳細資訊。例如，執行個體未安裝 Apache 時，`Not Applicable` 修補程式是 Apache 軟體的修補程式。Systems Manager 不會在摘要中報告 `Not Applicable` 修補程式的數目，但是如果您為受管節

點呼叫 DescribeInstancePatches API，則傳回的資料不會包含狀態為 `Not Applicable` 的修補程式。此行為不同於 Windows。

差異 3：SSM 文件支援

`AWS-ApplyPatchBaseline` Systems Manager 文件 (SSM 文件) 不支援 Linux 受管節點。若要將修補基準套用至 Linux、macOS 和 Windows Server 受管節點，建議的 SSM 文件是 `AWS-RunPatchBaseline`。如需詳細資訊，請參閱 用於修補受管節點的 SSM 命令文件 和 用於修補的 SSM 命令文件：`AWS-RunPatchBaseline`。

差異 4：應用程式修補程式

Patch Manager 的主要重點是將修補程式套用到作業系統。不過，您也可以使用 Patch Manager 在您受管節點的一些應用程式上套用修補程式。

Linux

在 Linux 作業系統上，Patch Manager 會使用已設定儲存庫進行更新，且不會區分作業系統和應用程式的修補程式。您可以使用 Patch Manager 來定義要擷取更新的儲存庫。如需詳細資訊，請參閱如何指定替代修補程式來源儲存庫 (Linux)。

Windows

在 Windows Server 受管節點上，您可以為 Microsoft 發佈的應用程式 (如 Microsoft Word 2016 和 Microsoft Exchange Server 2016) 套用核准規則以及已核准和已拒絕的修補程式例外狀況。如需詳細資訊，請參閱使用自訂修補基準。

差異 5：自訂修補基準中遭拒的修補程式清單選項

建立自訂修補基準時，可以為遭拒的修補程式清單指定一個或多個修補程式。對於 Linux 受管節點，如果它們是基準允許的其他修補程式的相依性，您也可以選擇允許安裝它們。

然而，Windows Server 不支援修補程式相依性的概念。您可以在 Windows Server 自訂基準中的遭拒的修補程式清單中新增修補程式，但結果取決於 (1) 是否已在受管節點上安裝拒絕的修補程式，以及 (2) 您為拒絕的修補程式動作選擇哪個選項。

如需有關 Windows Server 上遭拒的修補程式選項的詳細資訊，請參閱下表。

安裝狀態	選項：「當做相依性允許」	選項：「區塊」
修補程式已安裝	報告狀態：`INSTALLED_OTHER`	報告狀態：`INSTALLED_REJECTED`

安裝狀態	選項：「當做相依性允許」	選項：「區塊」
修補程式未安裝	修補程式已略過	修補程式已略過

Microsoft 發行的每個 Windows Server 修補程式通常都包含成功安裝所需的全部資訊。不過，有時可能需要您必須手動安裝的先決條件套件。Patch Manager 不會報告這些先決條件的相關資訊。如需相關資訊，請參閱 Microsoft 網站上的 Windows Update 問題疑難解答。

用於修補受管節點的 SSM 命令文件

此主題描述九個目前可用的 Systems Manager 文件 (SSM 文件)，協助您確保受管節點以最新的安全相關更新進行修補。

我們建議在您的修補操作中僅使用其中五個文件。這五個 SSM 文件可共同在您使用 AWS Systems Manager時提供完整的修補選項。其中四個文件的發佈晚於四個舊有 SSM 文件，它們取代並代表功能的擴充或合併。

適用於修補的建議 SSM 文件

我們建議在您的修補操作中使用下方的五個 SSM 文件。

- `AWS-ConfigureWindowsUpdate`
- `AWS-InstallWindowsUpdates`
- `AWS-RunPatchBaseline`
- `AWS-RunPatchBaselineAssociation`
- `AWS-RunPatchBaselineWithHooks`

適用於修補的舊版 SSM 文件

下列四個舊版 SSM 文件仍然可用於某些 AWS 區域 ，但這些文件不會再更新，因此無法保證可在所有案例中運作，而且未來可能不再受到支援。我們建議您不要在修補操作中使用它們。

- `AWS-ApplyPatchBaseline`
- `AWS-FindWindowsUpdates`
- `AWS-InstallMissingWindowsUpdates`
- `AWS-InstallSpecificWindowsUpdates`

如需有關在修補操作中使用這些 SSM 文件的詳細資訊，請參閱以下部分。

主題

- 建議用於修補受管節點的 SSM 文件
- 用於修補受管節點的舊版 SSM 文件
- 用於修補的 SSM 命令文件：AWS-RunPatchBaseline
- 用於修補的 SSM 命令文件：AWS-RunPatchBaselineAssociation
- 用於修補的 SSM 命令文件：AWS-RunPatchBaselineWithHooks
- 使用 AWS-RunPatchBaseline 或 AWS-RunPatchBaselineAssociation 中 InstallOverrideList 參數的範例案例
- 使用 BaselineOverride 參數

建議用於修補受管節點的 SSM 文件

建議在您的受管節點修補操作中，使用以下五個 SSM 文件。

建議的 SSM 文件

- AWS-ConfigureWindowsUpdate
- AWS-InstallWindowsUpdates
- AWS-RunPatchBaseline
- AWS-RunPatchBaselineAssociation
- AWS-RunPatchBaselineWithHooks

`AWS-ConfigureWindowsUpdate`

支援設定基本 Windows Update 功能以及使用它們自動安裝更新 (或關閉自動更新)。在全部 AWS 區域中提供。

此 SSM 文件提示 Windows Update 下載並安裝指定的更新，然後視需要重新啟動受管節點。將本文件與 中的工具 State Manager搭配使用 AWS Systems Manager，以確保 Windows Update 維持其組態。您也可以使用 中的Run Command工具手動執行它 AWS Systems Manager，以變更 Windows Update 組態。

此文件中可用的參數支援指定要安裝的更新類別 (或是否停用自動更新)，以及指定要在星期幾的什麼時間執行修補操作。如果您不需要嚴格控制 Windows 更新，也不需要收集合規資訊，那麼此 SSM 文件是最有用的。

取代舊有 SSM 文件：

- 無

AWS-InstallWindowsUpdates

在 Windows Server 受管節點上安裝更新。在全部 AWS 區域中提供。

在您希望安裝特定更新 (使用 `Include Kbs` 參數)，或希望安裝特定分類或類別的更新，但不需要修補程式合規資訊時，此 SSM 文件可提供基本修補功能。

取代舊有 SSM 文件：

- `AWS-FindWindowsUpdates`
- `AWS-InstallMissingWindowsUpdates`
- `AWS-InstallSpecificWindowsUpdates`

三個舊有文件執行不同的功能，但您可以使用更新的 SSM 文件 `AWS-InstallWindowsUpdates` 與不同的參數設定達到相同的結果。這些參數設定如用於修補受管節點的舊版 SSM 文件中所述。

AWS-RunPatchBaseline

在您的受管節點上安裝修補程式，或掃描節點以判斷是否有遺漏任何合格的修補程式。在全部 AWS 區域中提供。

`AWS-RunPatchBaseline` 允許您使用指定為作業系統類型之「預設」的修補基準來控制修補程式核准。報告修補程式合規資訊，您可以使用 Systems Manager 合規工具檢視。這些工具提供您受管節點修補程式合規狀態的洞見分析，例如哪些節點遺漏修補程式，以及遺漏的是哪些修補程式。當您使用 `AWS-RunPatchBaseline` 時，修補程式合規資訊會使用 `PutInventory` API 命令進行記錄。對於 Linux 作業系統，提供給修補程式的合規資訊來自受管節點上設定的預設來源儲存庫，以及您在自訂修補基準中指定的任何替代來源儲存庫。如需有關替代來源儲存庫的詳細資訊，請參閱 如何指定替代修補程式來源儲存庫 (Linux)。如需有關 Systems Manager 合規工具的詳細資訊，請參閱 AWS Systems Manager合規。

取代舊有文件：

- `AWS-ApplyPatchBaseline`

舊版文件 AWS-ApplyPatchBaseline 僅適用於 Windows Server 受管節點，不支援應用程式修補。較新的 AWS-RunPatchBaseline 為 Windows 和 Linux 系統提供相同的支援。SSM Agent 的 2.0.834.0 版或更新版本，才能使用 AWS-RunPatchBaseline 文件。

如需 AWS-RunPatchBaseline SSM 文件的詳細資訊，請參閱 用於修補的 SSM 命令文件：AWS-RunPatchBaseline。

AWS-RunPatchBaselineAssociation

在您的執行個體上安裝修補程式，或掃描執行個體以判斷是否有遺漏任何合格的修補程式。已在所有商業 AWS 區域提供。

AWS-RunPatchBaselineAssociation 在以下幾個重要方面不同於 AWS-RunPatchBaseline：

- AWS-RunPatchBaselineAssociation 主要用於使用 中Quick Setup工具 建立的State Manager 關聯 AWS Systems Manager。尤其是，當您使用 Quick Setup 主機管理組態類型時，如果您選擇 Scan instances for missing patches daily (每天掃描執行個體查看是否遺漏修補程式)，則系統會使用 AWS-RunPatchBaselineAssociation 進行操作。

  不過，在大多數情況下，當設定自己的修補操作時，您應該選擇 AWS-RunPatchBaseline 或 AWS-RunPatchBaselineWithHooks，而不是 AWS-RunPatchBaselineAssociation。

  如需詳細資訊，請參閱下列主題：

  - AWS Systems Manager Quick Setup
  - 用於修補的 SSM 命令文件：AWS-RunPatchBaselineAssociation
- AWS-RunPatchBaselineAssociation 支援使用標籤來識別執行一組目標時要與哪個修補基準搭配使用。
- 對於使用 AWS-RunPatchBaselineAssociation 的修補操作，修補程式合規資料會根據特定 State Manager 關聯進行編譯。AWS-RunPatchBaselineAssociation 執行時收集的修補程式合規資料會使用 PutComplianceItems API 命令，而不是 PutInventory 命令進行記錄。這樣可以防止覆寫不與此特定關聯相關聯的合規資料。

  對於 Linux 作業系統，提供給修補程式的合規資訊來自執行個體上設定的預設來源儲存庫，以及您在自訂修補基準中指定的任何替代來源儲存庫。如需有關替代來源儲存庫的詳細資訊，請參閱 如何指定替代修補程式來源儲存庫 (Linux)。如需有關 Systems Manager 合規工具的詳細資訊，請參閱 AWS Systems Manager合規。

取代舊有文件：

- 無

如需 AWS-RunPatchBaselineAssociation SSM 文件的詳細資訊，請參閱 用於修補的 SSM 命令文件：AWS-RunPatchBaselineAssociation。

AWS-RunPatchBaselineWithHooks

使用可在修補週期期間的三個點執行 SSM 文件的選用掛鉤，在您的受管節點上安裝修補程式，或掃描節點，以判斷是否遺漏任何合格的修補程式。已在所有商業 AWS 區域提供。macOS 上不支援。

AWS-RunPatchBaselineWithHooks 不同於其 Install 操作中的 AWS-RunPatchBaseline。

AWS-RunPatchBaselineWithHooks 支援在受管節點修補期間在指定點執行的生命週期掛鉤。由於修補程式安裝有時需要受管節點重新啟動，因此修補操作會分為兩個事件，總共有三個支援自訂功能的掛鉤。第一個掛鉤是在 Install with NoReboot 操作之前。第二個掛鉤是在 Install with NoReboot 操作之後。節點重新啟動後，第三個掛鉤可用。

取代舊有文件：

- 無

如需 AWS-RunPatchBaselineWithHooks SSM 文件的詳細資訊，請參閱 用於修補的 SSM 命令文件：AWS-RunPatchBaselineWithHooks。

用於修補受管節點的舊版 SSM 文件

以下四個 SSM 文件仍可於某些 AWS 區域中取得。但是，這些文件不會再更新而且可能在未來不再受到支援，因此不建議使用。反之，請使用建議用於修補受管節點的 SSM 文件中所述的文件。

舊有 SSM 文件

- AWS-ApplyPatchBaseline
- AWS-FindWindowsUpdates
- AWS-InstallMissingWindowsUpdates
- AWS-InstallSpecificWindowsUpdates

AWS-ApplyPatchBaseline

僅支援 Windows Server 受管節點，但不支援修補其替換項 AWS-RunPatchBaseline 中找到的應用程式。不適用於 2017 年 8 月之後 AWS 區域 推出的 。

> **Note**
>
> 此 SSM 文件的替代文件 AWS-RunPatchBaseline，需要 SSM Agent 的 2.0.834.0 版本或更新版。您可以使用 AWS-UpdateSSMAgent 文件將您的受管節點更新為最新版本的代理程式。

AWS-FindWindowsUpdates

由 AWS-InstallWindowsUpdates 取代，可執行所有相同的動作。在 2017 年 4 月之後 AWS 區域啟動的 中無法使用。

為了達到與此舊有 SSM 文件相同的結果，請使用下列參數組態與建議的替代文件 AWS-InstallWindowsUpdates：

- Action = Scan
- Allow Reboot = False

AWS-InstallMissingWindowsUpdates

由 AWS-InstallWindowsUpdates 取代，可執行所有相同的動作。不適用於 2017 年 4 月之後 AWS 區域 啟動的任何 。

為了達到與此舊有 SSM 文件相同的結果，請使用下列參數組態與建議的替代文件 AWS-InstallWindowsUpdates：

- Action = Install
- Allow Reboot = True

AWS-InstallSpecificWindowsUpdates

由 AWS-InstallWindowsUpdates 取代，可執行所有相同的動作。不適用於 2017 年 4 月之後 AWS 區域 啟動的任何 。

為了達到與此舊有 SSM 文件相同的結果，請使用下列參數組態與建議的替代文件 AWS-InstallWindowsUpdates：

- Action = Install
- Allow Reboot = True

- `Include Kbs = ##### KB ####`

用於修補的 SSM 命令文件：**AWS-RunPatchBaseline**

AWS Systems Manager 支援 `AWS-RunPatchBaseline`、適用於 的 Systems Manager 文件 (SSM 文件) Patch Manager、 中的工具 AWS Systems Manager。此 SSM 文件在受管節點上執行與安全相關和其他更新類型的修補操作。執行文件時，如果未指定修補程式群組，則會使用指定為作業系統類型之「預設」的修補基準。否則，它會使用與修補程式群組相關聯的修補基準。如需有關修補程式群組的資訊，請參閱 修補程式群組。

您可以使用 `AWS-RunPatchBaseline` 文件以套用適用於作業系統和應用程式的修補程式。(在 Windows Server 上，應用程式支援僅限於由 Microsoft 發行的應用程式更新。)

本文件支援 Linux、macOS，以及 Windows Server 受管節點。此文件會為各個平台執行適當的動作。

> **Note**
>
> Patch Manager 也支援舊版 SSM 文件 `AWS-ApplyPatchBaseline`。不過，此文件僅支援 Windows 受管節點上的修補。建議您改為使用 `AWS-RunPatchBaseline`，因為它支援在 Linux、macOS 和 Windows Server 受管節點上修補。SSM Agent 的 2.0.834.0 版或更新版本，才能使用 `AWS-RunPatchBaseline` 文件。

Windows Server

在 Windows Server 受管節點上，`AWS-RunPatchBaseline` 文件會下載並叫用 PowerShell 模組，進而下載套用至受管節點的修補基準快照。此修補基準快照集包含已核准修補程式清單，這些修補程式是藉由查詢修補基準針對 Windows Server 更新服務 (WSUS) 伺服器進行編譯。此清單會傳遞至 Windows Update API，視需要控制下載和安裝核准的修補程式。

Linux

在 Linux 受管節點上，`AWS-RunPatchBaseline` 文件會叫用 Python 模組，進而下載套用至受管節點的修補基準快照。此修補基準快照使用已定義的規則，以及已核准與已封鎖的修補程式清單，為各個節點類型推動適當的套件管理員：

- Amazon Linux 1、Amazon Linux 2、CentOS、Oracle Linux 和 RHEL 7 受管節點使用 YUM。針對 YUM 操作，Patch Manager 需要 `Python 2.6` 或更新版本的支援版本 (2.6 - 3.10)。
- RHEL 8 受管節點使用 DNF。針對 DNF 操作，Patch Manager 需要 `Python 2` 或 `Python 3` 的支援版本 (2.6 - 3.10)。(預設不會在 RHEL 8 上安裝兩個版本。您必須手動安裝其中一個。)

- Debian Server、Raspberry Pi OS 及 Ubuntu Server 執行個體使用 APT。針對 APT 操作，Patch Manager 需要 Python 3 的支援版本 (3.0 - 3.10)。
- SUSE Linux Enterprise Server 受管節點使用 Zypper。針對 Zypper 操作，Patch Manager 需要 Python 2.6 或更新版本的支援版本 (2.6 - 3.10)。

macOS

在 macOS 受管節點上，AWS-RunPatchBaseline 文件會叫用 Python 模組，進而下載套用至受管節點的修補基準快照。接著，Python 子程序會叫用節點上的 AWS Command Line Interface (AWS CLI)，以擷取指定套件管理員的安裝和更新資訊，並為每個更新套件驅動適當的套件管理員。

每個快照都專屬於 AWS 帳戶、修補程式群組、作業系統和快照 ID。快照是透過預先簽署的 Amazon Simple Storage Service (Amazon S3) URL 交付，快照會在建立後 24 小時過期。不過，URL 過期後，如果想要將相同的快照內容套用到其他受管節點，則您可以在建立快照後最多 3 天內產生新的預先簽署 Amazon Simple Storage Service (Amazon S3) URL。若要這麼做，請使用 get-deployable-patch-snapshot-for-instance 命令。

在安裝所有已核准且適用的更新，並視需要重新啟動之後，會在受管節點上產生修補程式合規資訊，並回報 Patch Manager。

> **Note**
>
> 如果在 AWS-RunPatchBaseline 文件中將 RebootOption 參數設定為 NoReboot，則在執行 Patch Manager 後不會重新啟動受管節點。如需詳細資訊，請參閱參數名稱：RebootOption。

如需有關檢視修補程式合規資料的詳細資訊，請參閱關於修補程式合規。

AWS-RunPatchBaseline 參數

AWS-RunPatchBaseline 支援六個參數。Operation 參數是必要參數。InstallOverrideList、BaselineOverride 和 RebootOption 參數是選用的。Snapshot-ID 技術上是選用的，但我們建議您在維護時段之外執行 AWS-RunPatchBaseline 時提供自訂值，並讓 Patch Manager 在該文件做為維護時段操作的一部分執行時自動提供自訂值。

參數

- 參數名稱：Operation
- 參數名稱：AssociationId
- 參數名稱：Snapshot ID
- 參數名稱：InstallOverrideList
- 參數名稱：RebootOption
- 參數名稱：BaselineOverride
- 參數名稱：StepTimeoutSeconds

參數名稱：**Operation**

用量：必要。

選項：Scan | Install。

Scan

當您選擇 Scan 選項時，AWS-RunPatchBaseline 會判斷受管節點的修補程式合規狀態，並將此資訊回報至 Patch Manager。Scan 不會提示要安裝的更新或需要重新啟動的受管節點。反之，此操作會識別遺漏了哪些已核准且適用於節點的更新。

安裝

當您選擇 Install 選項，AWS-RunPatchBaseline 會嘗試安裝受管節點上遺漏的已核准且適用的更新。在 Install 操作中產生的修補程式合規資訊不會列出任何遺失的更新，但如果因為任何原因導致未成功安裝更新，則可能會報告狀態為失敗的更新。每當更新安裝於受管節點時，節點將重新啟動，以確保安裝並啟動更新。(例外：如果 AWS-RunPatchBaseline 文件中的 RebootOption 參數設為 NoReboot，受管節點不會在 Patch Manager 執行之後重新啟動。如需詳細資訊，請參閱 參數名稱：RebootOption。)

> **Note**
>
> 如果在 Patch Manager 更新受管節點之前已安裝基準規則指定的修補程式，系統可能無法如預期重新啟動。當使用者手動安裝修補程式，或由其他程式自動安裝 (例如 Ubuntu Server 上的 unattended-upgrades 套件) 時，就會發生這種情況。

參數名稱：**AssociationId**

用量：選用。

AssociationId 是 中現有關聯的 IDState Manager，這是 中的工具 AWS Systems Manager。它由 Patch Manager 使用，將合規資料新增至指定的關聯。此關聯與 Quick Setup 中的修補程式政策中設定的修補操作有關。

> **Note**
>
> 使用 AWS-RunPatchBaseline 時，如果提供 AssociationId 值時發生修補程式政策基準覆寫，則修補會以 PatchPolicy 操作的形式完成，且 AWS:ComplianceItem 中報告的 ExecutionType 值也為 PatchPolicy。如果未提供任何 AssociationId 值，則會以 Command 操作形式完成修補，而且提交的 AWS:ComplianceItem 中報告的 ExecutionType 值也為 Command。

如果您還沒有想要使用的關聯，則可以透過執行 create-association 命令建立關聯。

參數名稱：**Snapshot ID**

用量：選用。

Snapshot ID 是 Patch Manager 使用的唯一 ID (GUID)，確保在單一操作中修補的一組受管節點皆有一組完全相同的核准修補程式。雖然此參數定義為選用，但我們的最佳實務建議取決於您是否會在維護時段中執行 AWS-RunPatchBaseline，如下表所述。

AWS-RunPatchBaseline 最佳實務

Mode	最佳實務	詳細資訊
在維護時段內執行 AWS-RunPatchBaseline	請勿提供快照 ID。Patch Manager 將會為您提供。	若您使用維護時段執行 AWS-RunPatchBaseline ，您不應提供自己產生的快照 ID。在此案例中，Systems Manager 會根據維護時段執行 ID 提供 GUID 值。這可確保維護時段中所有 AWS-RunPatchBaseline 呼叫皆使用正確的 ID。 如果您在此情況下指定值，則請注意修補基準的快照可能不

Mode	最佳實務	詳細資訊
		會保留超過 3 天。之後，即使您在快照過期後指定相同的 ID，仍將會產生新的快照。
在維護時段外執行 AWS-RunPatchBaseline	為快照 ID[1] 產生及指定自訂 GUID 值。	如果您不是使用維護時段執行 AWS-RunPatchBaseline ，我們建議您為每個修補基準產生並指定唯一的快照 ID，特別是如果您在相同的操作中，在多個受管節點上執行 AWS-RunPatchBaseline 文件時。如果您在此情況下沒有指定 ID，Systems Manager 將為命令傳送至的每個受管節點產生不同的快照 ID。這可能會導致在受管節點間指定不同的修補程式集合。 例如，假設您直接透過 、Run Command中的工具 AWS Systems Manager以及以 50 個受管節點群組為目標來執行AWS-RunPatchBaseline 文件。指定自訂快照 ID 會產生單一基準快照，用於評估和修補所有節點，以確保它們最終處於一致的狀態。

[1]您可以使用任何能產生 GUID 的工具來為快照 ID 參數產生一個值。例如，在 PowerShell，您可以使用 New-Guid cmdlet 產生 12345699-9405-4f69-bc5e-9315aEXAMPLE 格式的 GUID。

參數名稱：**InstallOverrideList**

用量：選用。

使用 InstallOverrideList，您可以將 https URL 或 Amazon Simple Storage Service (Amazon S3) 路徑樣式 URL 指定至要安裝的修補程式清單。此修補程式安裝清單 (以 YAML 格式維護) 會覆寫目前預設修補基準指定的修補程式。這可讓您更精密地控制哪些修補程式將安裝於您的受管節點。

> **Important**
>
> InstallOverrideList 檔案名稱不能包含下列字元：反引號 (`)、單引號 (')、雙引號 (") 和美元符號 ($)。

使用 InstallOverrideList 參數時的修補操作行為在 Linux 和 macOS 受管節點與 Windows Server 受管節點之間有所不同。在 Linux 與 macOS 上，Patch Manager 會嘗試套用 InstallOverrideList 修補程式清單中包含的修補程式，這些修補程式存在於節點上啟用的任何儲存庫中，無論修補程式是否符合修補基準規則。不過，只有在 InstallOverrideList 修補程式清單中的修補程式也符合修補基準規則時，才會在 Windows Server 節點上套用修補程式。

請注意，合規報告根據修補基準中的指定來反映修補程式狀態，而非您在 InstallOverrideList 合規清單中的指定。換言之，掃描操作會忽略 InstallOverrideList 參數。這是為了確保合規報告根據政策而非已核准用於特定修補操作的內容，來持續反映修補程式狀態。

如需如何使用 InstallOverrideList 參數依不同的維護時段排程將不同類型的修補程式套用至目標群組的說明，同時繼續單一修補基準，請參閱使用 AWS-RunPatchBaseline 或 AWS-RunPatchBaselineAssociation 中 InstallOverrideList 參數的範例案例。

有效的 URL 格式

> **Note**
>
> 如果您的檔案存放在公開可用的儲存貯體中，則可以指定 https URL 格式或 Amazon Simple Storage Service (Amazon S3) 路徑樣式的 URL。如果您的檔案存放在私有儲存貯體中，則必須指定 Amazon Simple Storage Service (Amazon S3) 路徑樣式的 URL。

- https URL 格式：

```
https://s3.aws-api-domain/amzn-s3-demo-bucket/my-windows-override-list.yaml
```

- Amazon Simple Storage Service (Amazon S3) 路徑樣式的 URL：

```
s3://amzn-s3-demo-bucket/my-windows-override-list.yaml
```

有效的 YAML 內容格式

您用於在清單中指定修補程式的格式，取決於您受管節點使用的作業系統。一般格式如下：

```
patches:
    -
        id: '{patch-d}'
        title: '{patch-title}'
        {additional-fields}:{values}
```

雖然您可以在 YAML 檔案中提供額外的欄位，但是在修補程式操作過程中會略過這些欄位。

此外，我們建議您在 S3 儲存貯體中新增或更新清單之前，確認您的 YAML 檔案格式是有效的。如需 YAML 格式的詳細資訊，請參閱 yaml.org。有關驗證工具的選項，請執行 Web 搜尋「yaml 格式驗證工具」。

Linux

id

id 欄位是必要的。利用它來使用套件名稱和架構以指定修補程式。例如：`'dhclient.x86_64'`。您可以在 id 中使用萬用字元以指示多個套件。例如：`'dhcp*'` 和 `'dhcp*1.*'`。

Title

標題欄位是選用的，但是在 Linux 系統上，它提供額外的篩選功能。如果您使用標題，它應包含套件版本資訊，並使用以下其中一種格式：

YUM/SUSE Linux Enterprise Server (SLES):

```
{name}.{architecture}:{epoch}:{version}-{release}
```

APT

```
{name}.{architecture}:{version}
```

對於 Linux 修補程式標題，您可以在任何位置使用一或多個萬用字元以擴大相符套件的數量。例如：`'*32:9.8.2-0.*.rc1.57.amzn1'`。

例如：

- apt 套件版本 1.2.25 目前已安裝於您的受管節點上，但現在有 1.2.27 可用。
- 您將 apt.amd64 版本 1.2.27 新增至修補程式清單。它取決於 apt utils.amd64 版本 1.2.27，但清單中指定的是 apt-utils.amd64 版本 1.2.25。

在此情況下，將會阻擋安裝 apt 版本 1.2.27，並回報為「Failed-NonCompliant」。

Windows Server

id

id 欄位是必要的。利用它來使用 Microsoft 知識庫 ID (例如 KB2736693) 和 Microsoft 資訊安全佈告欄 ID (例如 MS17-023) 指定修補程式。

您想在 Windows 修補程式清單中提供的任何其他欄位都是選用的，並僅供自己參考。您可以使用其他欄位，例如標題、分類、嚴重性等，提供有關指定的修補程式的更多詳細資訊。

macOS

id

id 欄位是必要的。id 欄位的值可以使用 `{package-name}.{package-version}` 格式或 {package_name} 格式。

範例修補程式清單

- 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'
```

參數名稱：**RebootOption**

用量：選用。

選項: RebootIfNeeded | NoReboot

預設：RebootIfNeeded

> **Warning**
>
> 預設選項為 RebootIfNeeded。請務必選取適用於您使用案例的正確選項。例如，如果您的受管節點必須立即重新啟動才能完成組態程序，則請選擇 RebootIfNeeded。或者，如果您需要維持受管節點的可用性，直到排定的重新啟動時間，則請選擇 NoReboot。

> **Important**
>
> 我們不建議使用 Patch Manager 修補 Amazon EMR (以前稱為 Amazon Elastic MapReduce) 中的叢集執行個體。特別是，請勿為 RebootOption 參數選取 RebootIfNeeded 選項。(此選項在用於修補 AWS-RunPatchBaseline、AWS-RunPatchBaselineAssociation 和 AWS-RunPatchBaselineWithHooks 的 SSM 命令文件中可用。)
>
> 使用 Patch Manager 進行修補時所使用的基礎命令使用 yum 和 dnf 命令。因此，相關操作會因套件的安裝方式而導致不相容。如需有關在 Amazon EMR 叢集上更新軟體的慣用方法的詳細資訊，請參閱《Amazon EMR 管理指南》中的使用 Amazon EMR 的預設 AMI 一節。

RebootIfNeeded

當您選擇 RebootIfNeeded 選項時，受管節點在下列情況下會重新啟動：

- Patch Manager 已安裝一或多個修補程式。

  Patch Manager 不會評估修補程式是否需要重新開機。即使修補程式不需要重新開機，系統也會重新開機。

- Patch Manager 偵測到一或多個修補程式在 Install 操作期間狀態為 INSTALLED_PENDING_REBOOT。

  INSTALLED_PENDING_REBOOT 狀態可能表示上次執行 Install 操作時已選取 NoReboot 選項，或者自上次重新啟動受管節點後，已在 Patch Manager 外部安裝修補程式。

在這兩種情況下重新啟動受管節點，可確保更新的套件會從記憶體中清除，並在所有作業系統中保持修補和重新啟動行為一致。

NoReboot

當您選擇 NoReboot 選項時，即使受管節點在 Install 作業期間安裝了修補程式，Patch Manager 也不會重新啟動受管節點。如果您知道您的受管節點在套用修補程式之後不需要重新啟動，或是您在節點上執行的應用程式或程序不應因於修補操作重新啟動而中斷，則此選項非常有用。當您想要進一步控制受管節點重新啟動的時間時 (例如使用維護時段)，此選項也很有用。

如果您選擇 NoReboot 選項並安裝修補程式，則會為修補程式指派 InstalledPendingReboot 的狀態。但是，受管節點本身會標示為 Non-Compliant。重新啟動並執行 Scan 操作之後，受管節點狀態會更新為 Compliant。

> **Important**
>
> NoReboot 選項只會防止作業系統層級重新啟動。服務層級重新啟動仍然可以作為修補程序的一部分發生。例如，當 Docker 更新時，即使NoReboot啟用 ，Amazon Elastic Container Service 等相依服務也可能自動重新啟動。如果您有不能中斷的重要服務，請考慮採取其他措施，例如暫時從服務中移除執行個體或在維護時段期間排程修補。

修補程式安裝追蹤檔案：若要追蹤修補程式安裝，特別是自上次系統重新啟動後已安裝的修補程式，Systems Manager 會在受管節點上維護檔案。

> **Important**
>
> 請勿刪除或修改追蹤檔案。如果此檔案已刪除或損毀，則受管節點的修補程式合規報告會不正確。如果發生此情況，請重新啟動節點並執行修補程式掃描作業以還原檔案。

此追蹤檔案存放於受管節點的下列位置：

- Linux 作業系統：
  - `/var/log/amazon/ssm/patch-configuration/patch-states-configuration.json`
  - `/var/log/amazon/ssm/patch-configuration/patch-inventory-from-last-operation.json`
- Windows Server 作業系統：
  - `C:\ProgramData\Amazon\PatchBaselineOperations\State\PatchStatesConfiguration.json`
  - `C:\ProgramData\Amazon\PatchBaselineOperations\State\PatchInventoryFromLastOperation.json`

參數名稱：**`BaselineOverride`**

用量：選用。

您可以使用 `BaselineOverride` 參數在執行時間定義修補偏好設定。此基準覆寫會作為 S3 儲存貯體中的 JSON 物件進行維護。它可確保修補操作使用符合主機作業系統所提供的基準，而不是從預設修補基準套用規則

> **Important**
>
> `BaselineOverride` 檔案名稱不能包含下列字元：反引號 (`)、單引號 (')、雙引號 (") 和美元符號 ($)。

如需使用 `BaselineOverride` 參數的詳細資訊，請參閱 使用 BaselineOverride 參數。

參數名稱：**`StepTimeoutSeconds`**

用量：選用。

命令在被視為失敗之前完成的秒數，介於 1 到 36000 秒 (10 小時） 之間。

用於修補的 SSM 命令文件：**AWS-RunPatchBaselineAssociation**

像 AWS-RunPatchBaseline 文件，AWS-RunPatchBaselineAssociation 在執行個體上執行與安全相關和其他更新類型的修補操作。您還可以使用 AWS-RunPatchBaselineAssociation 文件以套用適用於作業系統和應用程式的修補程式。(在 Windows Server 上，應用程式支援僅限於由 Microsoft 發佈的應用程式更新。)

本文件支援適用於 Linux、macOS 和 Windows Server 的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。其不支援混合多雲端環境中的非 EC2 節點。文件將在各個平台執行適當的動作，在 Linux 和 macOS 執行個體上叫用 Python 模組，在 Windows 執行個體上叫用 PowerShell 模組。

但是，AWS-RunPatchBaselineAssociation 在以下方面不同於 AWS-RunPatchBaseline：

- AWS-RunPatchBaselineAssociation 主要用於使用 中Quick Setup工具 建立的State Manager關聯 AWS Systems Manager。尤其是，當您使用 Quick Setup 主機管理組態類型時，如果您選擇 Scan instances for missing patches daily (每天掃描執行個體查看是否遺漏修補程式)，則系統會使用 AWS-RunPatchBaselineAssociation 進行操作。

  不過，在大多數情況下，當設定自己的修補操作時，您應該選擇 AWS-RunPatchBaseline 或 AWS-RunPatchBaselineWithHooks，而不是 AWS-RunPatchBaselineAssociation。
- 使用 AWS-RunPatchBaselineAssociation 文件時，您可以在文件的 BaselineTags 參數欄位指定標籤金鑰對。如果您的 中的自訂修補程式基準 AWS 帳戶 共用這些標籤， Patch Manager中的工具會在目標執行個體上執行時使用該標記的基準 AWS Systems Manager，而不是目前為作業系統類型指定的「預設」修補程式基準。

  > **Note**
  >
  > 如果您選擇使用修補操作中的 AWS-RunPatchBaselineAssociation，而不是使用 Quick Setup 的設定，並且您想要使用其選用 BaselineTags 參數，則必須提供部分額外的許可給執行個體設定檔，用於 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。如需詳細資訊，請參閱參數名稱：BaselineTags。

  以下兩種格式對 BaselineTags 參數都是有效的：

  `Key=tag-key,Values=tag-value`

  `Key=tag-key,Values=tag-value1,tag-value2,tag-value3`

> **Important**
>
> 標籤索引鍵和值不能包含下列字元：反引號 (`)、單引號 (')、雙引號 (") 和美元符號 ($)。

- AWS-RunPatchBaselineAssociation 執行時收集的修補程式合規資料會使用 PutComplianceItems API 命令，而不是 PutInventory 命令 (由 AWS-RunPatchBaseline 使用) 進行記錄。這種差異表示根據特定關聯存放和報告的修補程式合規資訊。不會覆寫在此關聯之外產生的修補程式合規資料。
- 執行 AWS-RunPatchBaselineAssociation 後報告的修補程式合規資訊指出執行個體是否合規。它不包含修補程式層級詳細資訊，如 following AWS Command Line Interface (AWS CLI) 命令的輸出所示。命令會在 Association 上進行篩選，作為合規類型：

```
aws ssm list-compliance-items \
    --resource-ids "i-02573cafcfEXAMPLE" \
    --resource-types "ManagedInstance" \
    --filters "Key=ComplianceType,Values=Association,Type=EQUAL" \
    --region us-east-2
```

系統會傳回如下資訊。

```
{
    "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"
        }
    ]
```

```
}
```

如果已將標籤金鑰對值指定為 AWS-RunPatchBaselineAssociation 文件的參數，則 Patch Manager 會搜尋符合作業系統類型且已使用該相同標籤金鑰對進行標記的自訂修補基準。此搜尋不限於目前指定的預設修補基準或指派給修補程式群組的基準。如果找不到具有指定標籤的基準線，Patch Manager 接下來會尋找修補程式群組 (如果在執行 AWS-RunPatchBaselineAssociation 的命令中指定了一個)。如果沒有相符的修補程式群組，則 Patch Manager 會回溯至目前作業系統帳戶的預設修補基準。

如果找到一個以上的修補基準，其中包含 AWS-RunPatchBaselineAssociation 文件中指定的標籤，則 Patch Manager 會傳回錯誤訊息，指出只有一個修補基準可以使用該鍵值對標記，以便繼續操作。

> **Note**
>
> 在 Linux 執行個體上，會使用每個執行個體類型的適當套件管理工具來安裝套件：
>
> - Amazon Linux 1、Amazon Linux 2、CentOS、Oracle Linux 和 RHEL 執行個體使用 YUM。針對 YUM 操作，Patch Manager 需要 Python 2.6 或更新版本的支援版本 (2.6 - 3.10)。
> - Debian Server、Raspberry Pi OS 及 Ubuntu Server 執行個體使用 APT。針對 APT 操作，Patch Manager 需要 Python 3 的支援版本 (3.0 - 3.10)。
> - SUSE Linux Enterprise Server 執行個體使用 Zypper。針對 Zypper 操作，Patch Manager 需要 Python 2.6 或更新版本的支援版本 (2.6 - 3.10)。

在掃描完成後或在所有已核准且適用的更新已安裝後，並視需要重新啟動之後，會在執行個體上產生修補程式合規資訊，並回報修補程式管理員。

> **Note**
>
> 如果在 AWS-RunPatchBaselineAssociation 文件中將 RebootOption 參數設定為 NoReboot，則在執行 Patch Manager 後不會重新啟動執行個體。如需詳細資訊，請參閱參數名稱：RebootOption。

如需有關檢視修補程式合規資料的詳細資訊，請參閱關於修補程式合規。

AWS-RunPatchBaselineAssociation 參數

AWS-RunPatchBaselineAssociation 支援五個參數。Operation 和 AssociationId 是必要參數。InstallOverrideList、RebootOption 和 BaselineTags 是選用參數。

參數

- 參數名稱：Operation
- 參數名稱：BaselineTags
- 參數名稱：AssociationId
- 參數名稱：InstallOverrideList
- 參數名稱：RebootOption

參數名稱：Operation

用量：必要。

選項：Scan | Install。

Scan

當您選擇 Scan 選項時，AWS-RunPatchBaselineAssociation 會判斷執行個體的修補程式合規狀態，並將此資訊回報至 Patch Manager。Scan 不會提示要安裝的更新或需要重新啟動的執行個體。反之，此操作會識別遺漏哪些已核准且適用於執行個體的更新。

安裝

當您選擇 Install 選項，AWS-RunPatchBaselineAssociation 會嘗試安裝執行個體上遺漏的已核准且適用的更新。在 Install 操作中產生的修補程式合規資訊不會列出任何遺失的更新，但如果因為任何原因導致未成功安裝更新，則可能會報告狀態為失敗的更新。每當更新安裝於執行個體時，執行個體將重新啟動，以確保安裝並啟動更新。(例外：如果 AWS-RunPatchBaselineAssociation 文件中的 RebootOption 參數設為 NoReboot，執行個體不會在 Patch Manager 執行之後重新啟動。如需詳細資訊，請參閱 參數名稱：RebootOption。)

> **Note**
>
> 如果在 Patch Manager 更新執行個體之前已安裝基準規則指定的修補程式，系統可能無法如預期重新開機。當使用者手動安裝修補程式，或由其他程式自動安裝 (例如 Ubuntu Server 上的 unattended-upgrades 套件) 時，就會發生這種情況。

參數名稱：**BaselineTags**

用量：選用。

BaselineTags 是唯一的標籤鍵值對，您可以選擇並指派給個別自訂修補基準。您可以為此參數指定一或多個值。以下兩種格式都是有效的：

Key=*tag-key*,Values=*tag-value*

Key=*tag-key*,Values=*tag-value1*,*tag-value2*,*tag-value3*

> **Important**
>
> 標籤索引鍵和值不能包含下列字元：反引號 (`)、單引號 (')、雙引號 (") 和美元符號 ($)。

BaselineTags 值是 Patch Manager 使用的唯一 ID (GUID)，確保在單一操作中修補的一組執行個體皆有一組完全相同的核准修補程式。修補操作執行時，Patch Manager 會檢查作業系統類型的修補基準是否已使用您為 BaselineTags 指定的相同鍵值對進行標記。如果有相符項目，則會使用此自訂修補基準。如果沒有相符項目，則會根據針對修補操作指定的任何修補程式群組來識別修補基準。如果沒有，則會使用該作業系統的 AWS 受管預先定義修補程式基準。

額外的許可要求

如果您使用修補操作中的 AWS-RunPatchBaselineAssociation，而不是使用 Quick Setup 的設定，並且您想要使用其選用 BaselineTags 參數，則必須新增以下許可給執行個體設定檔，用於 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。

> **Note**
>
> Quick Setup 和 AWS-RunPatchBaselineAssociation 不支援內部部署伺服器和虛擬機器 (VM)。

```
{
    "Effect": "Allow",
    "Action": [
        "ssm:DescribePatchBaselines",
        "tag:GetResources"
    ],
    "Resource": "*"
```

```
},
{
    "Effect": "Allow",
    "Action": [
        "ssm:GetPatchBaseline",
        "ssm:DescribeEffectivePatchesForPatchBaseline"
    ],
    "Resource": "patch-baseline-arn"
}
```

使用您要為其提供存取權之修補基準的 Amazon Resource Name (ARN) 取代 *patch-baseline-arn*，格式為 `arn:aws:ssm:us-east-2:123456789012:patchbaseline/pb-0c10e65780EXAMPLE`。

參數名稱：**AssociationId**

用量：必要。

`AssociationId` 是 中現有關聯的 IDState Manager，這是 中的工具 AWS Systems Manager。它由 Patch Manager 使用，將合規資料新增至指定的關聯。此關聯與在 Quick Setup 中建立的主機管理組態中啟用的修補程式 `Scan` 操作有關。藉由傳送修補結果作為關聯合規資料而非庫存合規資料，不會在修補操作之後覆寫執行個體的現有庫存合規資訊，也不會複寫其他關聯 ID。如果您還沒有想要使用的關聯，則可以透過執行 create-association 命令建立關聯。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
```

參數名稱：**InstallOverrideList**

用量：選用。

使用 InstallOverrideList，您可以將 https URL 或 Amazon Simple Storage Service (Amazon S3) 路徑樣式 URL 指定至要安裝的修補程式清單。此修補程式安裝清單 (以 YAML 格式維護) 會覆寫目前預設修補基準指定的修補程式。這可讓您更精密地控制哪些修補程式將安裝於您的執行個體。

> **Important**
>
> InstallOverrideList 檔案名稱不能包含下列字元：反引號 (`)、單引號 (')、雙引號 (") 和美元符號 ($)。

使用 InstallOverrideList 參數時的修補操作行為在 Linux 和 macOS 受管節點與 Windows Server 受管節點之間有所不同。在 Linux 與 macOS 上，Patch Manager 會嘗試套用 InstallOverrideList 修補程式清單中包含的修補程式，這些修補程式存在於節點上啟用的任何儲存庫中，無論修補程式是否符合修補基準規則。不過，只有在 InstallOverrideList 修補程式清單中的修補程式也符合修補基準規則時，才會在 Windows Server 節點上套用修補程式。

請注意，合規報告根據修補基準中的指定來反映修補程式狀態，而非您在 InstallOverrideList 合規清單中的指定。換言之，掃描操作會忽略 InstallOverrideList 參數。這是為了確保合規報告根據政策而非已核准用於特定修補操作的內容，來持續反映修補程式狀態。

有效的 URL 格式

> **Note**
>
> 如果您的檔案存放在公開可用的儲存貯體中，則可以指定 https URL 格式或 Amazon Simple Storage Service (Amazon S3) 路徑樣式的 URL。如果您的檔案存放在私有儲存貯體中，則必須指定 Amazon Simple Storage Service (Amazon S3) 路徑樣式的 URL。

- https URL 格式範例：

```
https://s3.amazonaws.com/amzn-s3-demo-bucket/my-windows-override-list.yaml
```

- Amazon S3 路徑樣式 URL 範例：

```
s3://amzn-s3-demo-bucket/my-windows-override-list.yaml
```

有效的 YAML 內容格式

您用於在清單中指定修補程式的格式，取決於您執行個體使用的作業系統。一般格式如下：

```
patches:
    -
        id: '{patch-d}'
        title: '{patch-title}'
        {additional-fields}:{values}
```

雖然您可以在 YAML 檔案中提供額外的欄位，但是在修補程式操作過程中會略過這些欄位。

此外，我們建議您在 S3 儲存貯體中新增或更新清單之前，確認您的 YAML 檔案格式是有效的。如需 YAML 格式的詳細資訊，請參閱 yaml.org。有關驗證工具的選項，請執行 Web 搜尋「yaml 格式驗證工具」。

- Microsoft Windows

  id

  id 欄位是必要的。利用它來使用 Microsoft 知識庫 ID (例如 KB2736693) 和 Microsoft 資訊安全佈告欄 ID (例如 MS17-023) 指定修補程式。

  您想在 Windows 修補程式清單中提供的任何其他欄位都是選用的，並僅供自己參考。您可以使用其他欄位，例如標題、分類、嚴重性等，提供有關指定的修補程式的更多詳細資訊。

- Linux

  id

  id 欄位是必要的。利用它來使用套件名稱和架構以指定修補程式。例如：`'dhclient.x86_64'`。您可以在 id 中使用萬用字元以指示多個套件。例如：`'dhcp*'` 和 `'dhcp*1.*'`。

標題

標題欄位是選用的，但是在 Linux 系統上，它提供額外的篩選功能。如果您使用標題，它應包含套件版本資訊，並使用以下其中一種格式：

YUM/SUSE Linux Enterprise Server (SLES):

```
{name}.{architecture}:{epoch}:{version}-{release}
```

APT

```
{name}.{architecture}:{version}
```

對於 Linux 修補程式標題，您可以在任何位置使用一或多個萬用字元以擴大相符套件的數量。例如：`'*32:9.8.2-0.*.rc1.57.amzn1'`。

例如：

- apt 套件版本 1.2.25 目前已安裝於您的執行個體上，但現在有 1.2.27 可用。
- 您將 apt.amd64 版本 1.2.27 新增至修補程式清單。它取決於 apt utils.amd64 版本 1.2.27，但清單中指定的是 apt-utils.amd64 版本 1.2.25。

在此情況下，將會阻擋安裝 apt 版本 1.2.27，並回報為「Failed-NonCompliant」。

其他欄位

您想在 Linux 修補程式清單中提供的任何其他欄位都是選用的，並僅供自己參考。您可以使用其他欄位，例如分類、嚴重性等，提供有關指定的修補程式的更多詳細資訊。

範例修補程式清單

- 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'
```

參數名稱：**RebootOption**

用量：選用。

選項: RebootIfNeeded | NoReboot

預設：RebootIfNeeded

Warning

預設選項為 RebootIfNeeded。請務必選取適用於您使用案例的正確選項。例如，如果您的執行個體必須立即重新啟動才能完成組態程序，則請選擇 RebootIfNeeded。或者，如果您需要維持執行個體的可用性，直到排定的重新啟動時間，則請選擇 NoReboot。

Important

NoReboot 選項只會防止作業系統層級重新啟動。服務層級重新啟動仍然可以作為修補程序的一部分發生。例如，當 Docker 更新時，即使NoReboot啟用 ，Amazon Elastic Container Service 等相依服務也可能自動重新啟動。如果您有不能中斷的重要服務，請考慮採取其他措施，例如暫時從服務中移除執行個體或在維護時段期間排程修補。

Important

我們不建議使用 Patch Manager 修補 Amazon EMR (以前稱為 Amazon Elastic MapReduce) 中的叢集執行個體。特別是，請勿為 RebootOption 參數選取 RebootIfNeeded 選項。(此選項在用於修補 AWS-RunPatchBaseline、AWS-RunPatchBaselineAssociation 和 AWS-RunPatchBaselineWithHooks 的 SSM 命令文件中可用。)
使用 Patch Manager 進行修補時所使用的基礎命令使用 yum 和 dnf 命令。因此，相關操作會因套件的安裝方式而導致不相容。如需有關在 Amazon EMR 叢集上更新軟體的慣用方法的詳細資訊，請參閱《Amazon EMR 管理指南》中的使用 Amazon EMR 的預設 AMI 一節。

RebootIfNeeded

當您選擇 RebootIfNeeded 選項時，執行個體在下列情況下會重新開機：

- Patch Manager 已安裝一或多個修補程式。

  Patch Manager 不會評估修補程式是否需要重新開機。即使修補程式不需要重新開機，系統也會重新開機。

- Patch Manager 偵測到一或多個修補程式在 Install 操作期間狀態為 INSTALLED_PENDING_REBOOT。

`INSTALLED_PENDING_REBOOT` 狀態可能表示上次執行 `Install` 操作時已選取 `NoReboot` 選項，或者自上次重新啟動受管節點後，已在 Patch Manager 外部安裝修補程式。

在這兩種情況下重新開機執行個體，可確保更新的套件會從記憶體中清除，並在所有作業系統中保持修補和重新開機行為一致。

NoReboot

當您選擇 `NoReboot` 選項時，即使執行個體在 `Install` 作業期間安裝了修補程式，Patch Manager 也不會重新啟動執行個體。如果您知道您的執行個體在套用修補程式之後不需要重新啟動，或是您在執行個體上執行的應用程式或程序不應因於修補操作重新啟動而中斷，則此選項非常有用。當您想要進一步控制執行個體重新啟動的時間時 (例如使用維護時段)，此選項也很有用。

修補程式安裝追蹤檔案：若要追蹤修補程式安裝，特別是自上次系統重新啟動後已安裝的修補程式，Systems Manager 會在受管執行個體上維護檔案。

> **Important**
>
> 請勿刪除或修改追蹤檔案。如果此檔案已刪除或損毀，則執行個體的修補程式合規報告會不正確。如果發生此情況，請重新啟動執行個體並執行修補程式掃描作業以還原檔案。

此追蹤檔案存放於受管執行個體的下列位置：

- Linux 作業系統：
  - `/var/log/amazon/ssm/patch-configuration/patch-states-configuration.json`
  - `/var/log/amazon/ssm/patch-configuration/patch-inventory-from-last-operation.json`
- Windows Server 作業系統：
  - `C:\ProgramData\Amazon\PatchBaselineOperations\State\PatchStatesConfiguration.json`
  - `C:\ProgramData\Amazon\PatchBaselineOperations\State\PatchInventoryFromLastOperation.json`

用於修補的 SSM 命令文件：**AWS-RunPatchBaselineWithHooks**

AWS Systems Manager 支援 AWS-RunPatchBaselineWithHooks、適用於 的 Systems Manager 文件 (SSM 文件) Patch Manager、 中的工具 AWS Systems Manager。此 SSM 文件在受管節點上執行與安全相關和其他更新類型的修補操作。

AWS-RunPatchBaselineWithHooks 在以下方面不同於 AWS-RunPatchBaseline：

- 包裝文件–AWS-RunPatchBaselineWithHooks 是 AWS-RunPatchBaseline 的包裝，並依賴 AWS-RunPatchBaseline 進行其操作。
- **Install** 操作 – AWS-RunPatchBaselineWithHooks 支援在受管節點修補期間在指定點執行的生命週期掛鉤。由於修補程式安裝有時需要受管節點重新啟動，因此修補操作會分為兩個事件，總共有三個支援自訂功能的掛鉤。第一個掛鉤是在 Install with NoReboot 操作之前。第二個掛鉤是在 Install with NoReboot 操作之後。受管節點重新啟動後，第三個掛鉤可用。
- 不支援自訂修補程式清單 – AWS-RunPatchBaselineWithHooks 不支援 InstallOverrideList 參數。
- SSM Agent 支援 – AWS-RunPatchBaselineWithHooks 需要 SSM Agent 3.0.502 或更新版本安裝在要修補的受管節點上。

執行文件時，如果未指定修補程式群組，則會使用當前指定為作業系統類型之「預設」的修補基準。否則，它會使用與修補程式群組相關聯的修補基準。如需有關修補程式群組的資訊，請參閱 修補程式群組。

您可以使用 AWS-RunPatchBaselineWithHooks 文件以套用適用於作業系統和應用程式的修補程式。(在 Windows Server 上，應用程式支援僅限於由 Microsoft 發行的應用程式更新。)

本文件支援 Linux 以及 Windows Server 受管節點。此文件會為各個平台執行適當的動作。

> **Note**
>
> macOS 不支援 AWS-RunPatchBaselineWithHooks。

Linux

在 Linux 受管節點上，AWS-RunPatchBaselineWithHooks 文件會叫用 Python 模組，進而下載套用至受管節點的修補基準快照。此修補基準快照使用已定義的規則，以及已核准與已封鎖的修補程式清單，為各個節點類型推動適當的套件管理員：

- Amazon Linux 1、Amazon Linux 2、CentOS、Oracle Linux 和 RHEL 7 受管節點使用 YUM。針對 YUM 操作，Patch Manager 需要 `Python 2.6` 或更新版本的支援版本 (2.6 - 3.10)。
- RHEL 8 受管節點使用 DNF。針對 DNF 操作，Patch Manager 需要 `Python 2` 或 `Python 3` 的支援版本 (2.6 - 3.10)。(預設不會在 RHEL 8 上安裝兩個版本。您必須手動安裝其中一個。)
- Debian Server、Raspberry Pi OS 及 Ubuntu Server 執行個體使用 APT。針對 APT 操作，Patch Manager 需要 `Python 3` 的支援版本 (3.0 - 3.10)。
- SUSE Linux Enterprise Server 受管節點使用 Zypper。針對 Zypper 操作，Patch Manager 需要 `Python 2.6` 或更新版本的支援版本 (2.6 - 3.10)。

Windows Server

在 Windows Server 受管節點上，`AWS-RunPatchBaselineWithHooks` 文件會下載並叫用 PowerShell 模組，進而下載套用至受管節點的修補基準快照。此修補基準快照集包含已核准修補程式清單，這些修補程式是藉由查詢修補基準針對 Windows Server 更新服務 (WSUS) 伺服器進行編譯。此清單會傳遞至 Windows Update API，視需要控制下載和安裝核准的修補程式。

每個快照都專屬於 AWS 帳戶、修補程式群組、作業系統和快照 ID。快照是透過預先簽署的 Amazon Simple Storage Service (Amazon S3) URL 交付，快照會在建立後 24 小時過期。不過，URL 過期後，如果想要將相同的快照內容套用到其他受管節點，則您可以在建立快照後最多三天內產生新的預先簽署 Amazon Simple Storage Service (Amazon S3) URL。若要這麼做，請使用 get-deployable-patch-snapshot-for-instance 命令。

在安裝所有已核准且適用的更新，並視需要重新啟動之後，會在受管節點上產生修補程式合規資訊，並回報 Patch Manager。

如果在 `AWS-RunPatchBaselineWithHooks` 文件中將 `RebootOption` 參數設定為 `NoReboot`，則在執行 Patch Manager 後不會重新啟動受管節點。如需詳細資訊，請參閱參數名稱：`RebootOption`。

> **Important**
>
> 雖然 `NoReboot`選項可防止作業系統重新啟動，但無法防止更新特定套件時可能發生的服務層級重新啟動。例如，更新 Docker 之類的套件可能會觸發相依服務 （例如容器協同運作服務）的自動重新啟動，即使 `NoReboot` 已指定。

如需有關檢視修補程式合規資料的詳細資訊，請參閱關於修補程式合規。

AWS-RunPatchBaselineWithHooks 操作步驟

當 AWS-RunPatchBaselineWithHooks 執行時，會進行下列步驟：

1. 掃描 – 在受管節點上執行使用 AWS-RunPatchBaseline 的 Scan 操作，並產生並上傳合規報告。
2. 確認本機修補程式狀態 – 執行指令碼，以確定將根據所選的操作和步驟 1 的 Scan 結果執行什麼步驟。
   a. 如果選取的操作為 Scan，則操作會標示為完成。操作結束。
   b. 如果選取的操作為 Install，則 Patch Manager 會評估步驟 1 的 Scan 結果，以確定接下來要執行的內容：
      i. 如果未偵測到遺漏的修補程式，且不需要擱置中的重新開機，則操作會直接進行到最後一個步驟 (步驟 8)，其中包括您提供的掛鉤。兩者之間的任何步驟都會略過。
      ii. 如果未偵測到遺漏的修補程式，但有所需的擱置中重新開機，且所選重新開啟選項為 NoReboot，則操作會直接進行到最後一個步驟 (步驟 8)，其中包括您提供的掛鉤。兩者之間的任何步驟都會略過。
      iii. 否則，操作會繼續至下一個步驟。
3. 修補程式前掛鉤操作 - 您為第一個lifecycle hook PreInstallHookDocName 提供的 SSM 文件在受管節點上執行。
4. 使用 NoReboot 安裝 – 使用 AWS-RunPatchBaseline 且具有 NoReboot 重啟啟動選項的 Install 操作在受管節點上執行，還會產生並上傳合規報告。
5. 安裝後掛鉤操作 - 您為第二個lifecycle hook PostInstallHookDocName 提供的 SSM 文件在受管節點上執行。
6. 確認重新啟動 – 執行指令碼以判斷受管節點是否需要重新開機，以及需要執行哪些步驟：
   a. 如果選取的重新開機選項為 NoReboot，則操作會直接進行到最後一個步驟 (步驟 8)，其中包含您提供的掛鉤。兩者之間的任何步驟都會略過。
   b. 如果選取的重新開機選項為 RebootIfNeeded，則 Patch Manager 會檢查步驟 4 中所收集的庫存是否有所需之任何擱置中的重新開機。這意味著在下列任一情況下，作業會繼續執行步驟 7，且受管節點會重新啟動：
      i. Patch Manager 已安裝一個或多個修補程式。(Patch Manager 不評估修補程式是否需要重新啟動。即使修補程式不需要重新啟動，系統也會重新啟動。)
      ii. Patch Manager 偵測到一或多個修補程式在安裝操作期間狀態為 INSTALLED_PENDING_REBOOT。INSTALLED_PENDING_REBOOT 狀態可能表示上次 Install 操作時已選取 NoReboot 選項，或者自上次重新啟動受管節點後，已在 Patch Manager 外部安裝修補程式。

如果找不到需要符合這些條件的修補程式，則受管節點修補操作即完成，且操作會直接進行到最後一個步驟 (步驟 8)，其中包括您提供的掛鉤。兩者之間的任何步驟都會略過。

7. 重新啟動和報告 – 具有 RebootIfNeeded 重啟啟動選項的安裝操作在使用 AWS-RunPatchBaseline 的受管節點上執行，還會產生並上傳合規報告。
8. 重新啟動後掛鉤操作 – 您為第三個lifecycle hook OnExitHookDocName 提供的 SSM 文件在受管節點上執行。

對於 Scan 操作，如果步驟 1 失敗，則執行文件的程序會停止並報告為失敗，儘管後續步驟會報告為成功。

對於 Install 操作，如果有任何 aws:runDocument 步驟在操作期間失敗，則這些步驟會報告為失敗，而且操作會直接進行到最後一個步驟 (步驟 8)，其中包含您提供的掛鉤。兩者之間的任何步驟都會略過。此步驟會報告為失敗，最後一個步驟會報告其操作結果的狀態，而兩者之間的所有步驟都會報告為成功。

AWS-RunPatchBaselineWithHooks 參數

AWS-RunPatchBaselineWithHooks 支援六個參數。

Operation 參數是必要參數。

RebootOption、PreInstallHookDocName、PostInstallHookDocName 和 OnExitHookDocName 是選用參數。

Snapshot-ID 在技術上是選用的，但建議您在維護時段以外執行 AWS-RunPatchBaselineWithHooks 時，為其提供自訂值。當文件以維護時段操作的一部分執行時，讓 Patch Manager 自動提供值。

參數

- 參數名稱：Operation
- 參數名稱：Snapshot ID
- 參數名稱：RebootOption
- 參數名稱：PreInstallHookDocName
- 參數名稱：PostInstallHookDocName
- 參數名稱：OnExitHookDocName

參數名稱：**Operation**

用量：必要。

選項：Scan | Install。

Scan

當您選擇 Scan 選項時，系統會使用 AWS-RunPatchBaseline 文件判斷受管節點的修補程式合規狀態，並將此資訊回報至 Patch Manager。Scan 不會提示要安裝的更新或需要重新啟動的受管節點。反之，此操作會識別遺漏了哪些已核准且適用於節點的更新。

安裝

當您選擇 Install 選項，AWS-RunPatchBaselineWithHooks 會嘗試安裝受管節點上遺漏的已核准且適用的更新。在 Install 操作中產生的修補程式合規資訊不會列出任何遺失的更新，但如果因為任何原因導致未成功安裝更新，則可能會報告狀態為失敗的更新。每當更新安裝於受管節點時，節點將重新啟動，以確保安裝並啟動更新。(例外：如果 AWS-RunPatchBaselineWithHooks 文件中的 RebootOption 參數設為 NoReboot，受管節點不會在 Patch Manager 執行之後重新啟動。如需詳細資訊，請參閱 參數名稱：RebootOption。)

> **Note**
>
> 如果在 Patch Manager 更新受管節點之前已安裝基準規則指定的修補程式，系統可能無法如預期重新啟動。當使用者手動安裝修補程式，或由其他程式自動安裝 (例如 Ubuntu Server 上的 unattended-upgrades 套件) 時，就會發生這種情況。

參數名稱：**Snapshot ID**

用量：選用。

Snapshot ID 是 Patch Manager 使用的唯一 ID (GUID)，確保在單一操作中修補的一組受管節點皆有一組完全相同的核准修補程式。雖然此參數定義為選用，但我們的最佳實務建議取決於您是否會在維護時段中執行 AWS-RunPatchBaselineWithHooks，如下表所述。

AWS-RunPatchBaselineWithHooks 最佳實務

Mode	最佳實務	詳細資訊
在維護時段內執行 AWS-RunPatchBaselineWithHooks	請勿提供快照 ID。Patch Manager 將會為您提供。	若您使用維護時段執行 AWS-RunPatchBaselineWithHooks ，您不應提供自己產生的快照 ID。在此案例中，Systems Manager 會根據維護時段執行 ID 提供 GUID 值。這可確保維護時段中所有 AWS-RunPatchBaselineWithHooks 呼叫皆使用正確的 ID。 如果您在此情況下指定值，則請注意修補基準的快照可能不會保留超過 3 天。之後，即使您在快照過期後指定相同的 ID，仍將會產生新的快照。
在維護時段外執行 AWS-RunPatchBaselineWithHooks	為快照 ID[1] 產生及指定自訂 GUID 值。	如果您不是使用維護時段執行 AWS-RunPatchBaselineWithHooks ，我們建議您為每個修補基準產生並指定唯一的快照 ID，特別是如果您在相同的操作中，在多個受管節點上執行 AWS-RunPatchBaselineWithHooks 文件時。如果您在此情況下沒有指定 ID，Systems Manager 將為命令傳送至的每個受管節點產生不同的快照 ID。這可能會導致在節點間指定不同的修補程式集合。 例如，假設您直接透過 、Run Command中的工具AWS

Mode	最佳實務	詳細資訊
		Systems Manager以及以 50 個受管節點群組為目標來執行AWS-RunPatchBaselineWithHooks 文件。指定自訂快照 ID 會產生單一基準快照，用於評估和修補所有受管節點，以確保它們最終處於一致的狀態。

[1]您可以使用任何能產生 GUID 的工具來為快照 ID 參數產生一個值。例如，在 PowerShell，您可以使用 New-Guid cmdlet 產生 12345699-9405-4f69-bc5e-9315aEXAMPLE 格式的 GUID。

參數名稱：**RebootOption**

用量：選用。

選項: RebootIfNeeded | NoReboot

預設：RebootIfNeeded

> **Warning**
>
> 預設選項為 RebootIfNeeded。請務必選取適用於您使用案例的正確選項。例如，如果您的受管節點必須立即重新啟動才能完成組態程序，則請選擇 RebootIfNeeded。或者，如果您需要維持受管節點的可用性，直到排定的重新啟動時間，則請選擇 NoReboot。

> **Important**
>
> 我們不建議使用 Patch Manager 修補 Amazon EMR (以前稱為 Amazon Elastic MapReduce) 中的叢集執行個體。特別是，請勿為 RebootOption 參數選取 RebootIfNeeded 選項。(此選項在用於修補 AWS-RunPatchBaseline、AWS-RunPatchBaselineAssociation 和 AWS-RunPatchBaselineWithHooks 的 SSM 命令文件中可用。)

使用 Patch Manager 進行修補時所使用的基礎命令使用 yum 和 dnf 命令。因此，相關操作會因套件的安裝方式而導致不相容。如需有關在 Amazon EMR 叢集上更新軟體的慣用方法的詳細資訊，請參閱《Amazon EMR 管理指南》中的使用 Amazon EMR 的預設 AMI 一節。

RebootIfNeeded

當您選擇 RebootIfNeeded 選項時，受管節點在下列情況下會重新啟動：

- Patch Manager 已安裝一或多個修補程式。

  Patch Manager 不會評估修補程式是否需要重新開機。即使修補程式不需要重新開機，系統也會重新開機。

- Patch Manager 偵測到一或多個修補程式在 Install 操作期間狀態為 INSTALLED_PENDING_REBOOT。

  INSTALLED_PENDING_REBOOT 狀態可能表示上次執行 Install 操作時已選取 NoReboot 選項，或者自上次重新啟動受管節點後，已在 Patch Manager 外部安裝修補程式。

在這兩種情況下重新啟動受管節點，可確保更新的套件會從記憶體中清除，並在所有作業系統中保持修補和重新啟動行為一致。

NoReboot

當您選擇 NoReboot 選項時，即使受管節點在 Install 作業期間安裝了修補程式，Patch Manager 也不會重新啟動受管節點。如果您知道您的受管節點在套用修補程式之後不需要重新啟動，或是您在節點上執行的應用程式或程序不應因於修補操作重新啟動而中斷，則此選項非常有用。當您想要進一步控制受管節點重新啟動的時間時 (例如使用維護時段)，此選項也很有用。

> **Note**
>
> 如果您選擇 NoReboot 選項並安裝修補程式，則會為修補程式指派 InstalledPendingReboot 的狀態。但是，受管節點本身會標示為 Non-Compliant。重新啟動並執行 Scan 操作之後，節點狀態會更新為 Compliant。

修補程式安裝追蹤檔案：若要追蹤修補程式安裝，特別是自上次系統重新啟動後已安裝的修補程式，Systems Manager 會在受管節點上維護檔案。

> **Important**
>
> 請勿刪除或修改追蹤檔案。如果此檔案已刪除或損毀，則受管節點的修補程式合規報告會不正確。如果發生此情況，請重新啟動節點並執行修補程式掃描作業以還原檔案。

此追蹤檔案存放於受管節點的下列位置：

- Linux 作業系統：
  - `/var/log/amazon/ssm/patch-configuration/patch-states-configuration.json`
  - `/var/log/amazon/ssm/patch-configuration/patch-inventory-from-last-operation.json`
- Windows Server 作業系統：
  - `C:\ProgramData\Amazon\PatchBaselineOperations\State\PatchStatesConfiguration.json`
  - `C:\ProgramData\Amazon\PatchBaselineOperations\State\PatchInventoryFromLastOperation.json`

參數名稱：**`PreInstallHookDocName`**

用量：選用。

預設：`AWS-Noop`。

提供給 `PreInstallHookDocName` 參數的值是您所選 SSM 文件的名稱或 Amazon Resource Name (ARN)。您可以提供 AWS 受管文件的名稱，或已建立或已與您共用的自訂 SSM 文件的名稱或 ARN。（對於從不同 與您共用的 SSM 文件 AWS 帳戶，您必須指定完整的資源 ARN，例如 `arn:aws:ssm:us-east-2:123456789012:document/MySharedDocument`。）

您指定的 SSM 文件會在 `Install` 操作之前執行，並執行 SSM Agent 支援的任何動作，例如用於在受管節點上執行修補之前進行應用程式運作狀態檢查的 shell 指令碼。(如需動作清單，請參閱 命令文件外掛程式參考)。預設 SSM 文件名稱為 `AWS-Noop`，它不會對受管節點執行任何操作。

如需建立自訂 SSM 文件的詳細資訊，請參閱 建立 SSM 文件內容。

參數名稱：**`PostInstallHookDocName`**

用量：選用。

預設：AWS-Noop。

提供給 PostInstallHookDocName 參數的值是您所選 SSM 文件的名稱或 Amazon Resource Name (ARN)。您可以提供 AWS 受管文件的名稱，或已建立或已與您共用的自訂 SSM 文件的名稱或 ARN。（對於從不同 與您共用的 SSM 文件 AWS 帳戶，您必須指定完整的資源 ARN，例如 arn:aws:ssm:us-east-2:123456789012:document/MySharedDocument。）

在 Install with NoReboot 操作之後執行您指定的 SSM 文件，還會執行 SSM Agent 支援的任何操作，例如重新開機前安裝第三方更新的 shell 指令碼。(如需動作清單，請參閱 命令文件外掛程式參考)。預設 SSM 文件名稱為 AWS-Noop，它不會對受管節點執行任何操作。

如需建立自訂 SSM 文件的詳細資訊，請參閱 建立 SSM 文件內容。

參數名稱：**OnExitHookDocName**

用量：選用。

預設：AWS-Noop。

提供給 OnExitHookDocName 參數的值是您所選 SSM 文件的名稱或 Amazon Resource Name (ARN)。您可以提供 AWS 受管文件的名稱，或已建立或已與您共用的自訂 SSM 文件的名稱或 ARN。(若是來自不同 AWS 帳戶且已與您共用的 SSM 文件，您必須指定完整資源 ARN，例如 arn:aws:ssm:us-east-2:123456789012:document/MySharedDocument。)

您指定的 SSM 文件會在受管節點重新啟動操作之後執行，並執行 SSM Agent 支援的任何動作，例如用於在修補操作完成之後確認節點運作狀態的 shell 指令碼。(如需動作清單，請參閱 命令文件外掛程式參考)。預設 SSM 文件名稱為 AWS-Noop，它不會對受管節點執行任何操作。

如需建立自訂 SSM 文件的詳細資訊，請參閱 建立 SSM 文件內容。

使用 **AWS-RunPatchBaseline** 或 **AWS-RunPatchBaselineAssociation** 中 InstallOverrideList 參數的範例案例

當您想要覆寫 中目前預設修補程式基準所指定的修補程式時，您可以使用 InstallOverrideList 參數Patch Manager，這是 中的工具 AWS Systems Manager。本主題提供示範如何使用此參數來達成下列目標的範例：

- 將不同的修補程式集套用至目標受管節點群組。
- 依不同的頻率套用這些修補程式集。
- 兩項操作都使用相同的修補基準。

假設您要在 Amazon Linux 2 受管節點上安裝兩種不同類別的修補程式。您想要使用維護時段，依不同的排程上安裝這些修補程式。您希望每週執行一個維護時段，並安裝所有 Security 修補程式。您希望另一個維護時段每月執行一次，並安裝所有可用的修補程式或 Security 以外的修補程式類別。

不過，一次只能將一個修補基準定義為作業系統的預設值。此要求有助於避免某個修補基準核准修補程式，而另一個修補程式封鎖該修補程式，這可能在衝突的版本之間導致問題。

下列策略可讓您使用 InstallOverrideList 參數，依不同的排程將不同類型的修補程式套用至目標群組，同時繼續使用相同的修補基準。

1. 在預設修補基準中，確定僅指定 Security 更新。
2. 建立維護時段，在每週執行 AWS-RunPatchBaseline 或 AWS-RunPatchBaselineAssociation。請勿指定覆寫清單。
3. 建立您要每月套用之所有修補程式類型的覆寫清單，並將其存放在 Amazon Simple Storage Service (Amazon S3) 儲存貯體中。
4. 建立第二個維護時段，在每月執行一次。不過，針對您為此維護時段註冊的 Run Command 任務，請指定覆寫清單的位置。

結果：每週只會安裝預設修補基準中定義的 Security 修補程式。每月都會安裝所有可用的修補程式，或您定義的任何修補程式子集。

如需詳細資訊和範例清單，請參閱 參數名稱：InstallOverrideList。

使用 BaselineOverride 參數

您可以使用 中的基準覆寫功能，在執行時間定義修補偏好設定Patch Manager，該功能是 中的工具 AWS Systems Manager。透過指定 Amazon Simple Storage Service (Amazon S3) 儲存貯體完成此操作，其中包含具有修補基準清單的 JSON 物件。修補操作使用符合主機作業系統之 JSON 物件中所提供的基準，而不是從預設修補基準套用規則。

> **Important**
>
> BaselineOverride 檔案名稱不能包含下列字元：反引號 (`)、單引號 (')、雙引號 (") 和美元符號 ($)。

除非修補操作使用修補程式政策，否則使用 BaselineOverride 參數不會覆寫參數中所提供之基準的修補程式合規。輸出結果會記錄在 的 Stdout 日誌中Run Command，這是 中的工具 AWS

Systems Manager。結果只會列印標示為 NON_COMPLIANT 的套件。這意味著該套件會標記為 Missing、Failed、InstalledRejected 或 InstalledPendingReboot。

然而，當修補程式操作使用修補程式政策時，系統會從關聯的 S3 儲存貯體傳遞覆寫參數，並更新受管節點的合規值。如需有關修補程式政策行為的詳細資訊，請參閱 Quick Setup中的修補程式政策組態。

使用修補基準覆寫 Snapshot Id 或 Install Override List 參數

在兩種情況下，修補基準覆寫具有值得注意的行為。

同時使用基準線覆寫和 Snapshot Id

Snapshot Id 可確保特定修補命令中的所有受管節點都套用相同的項目。例如，如果您一次修補 1,000 個節點，則修補程式將會相同。

同時使用 Snapshot Id 和修補基準覆寫時，Snapshot Id 的優先順序高於修補基準覆寫。仍會使用基準線覆寫規則，但只會評估一次。在先前的範例中，1,000 個受管節點的修補程式仍會永遠相同。如果在修補操作中間，您將參考之 S3 儲存貯體中的 JSON 檔案變更為不同的內容，則套用的修補程式仍然相同。這是因為已提供 Snapshot Id。

同時使用基準覆寫和 Install Override List

您無法同時使用這兩個參數。如果提供這兩個參數，則修補文件就會失敗，而且不會在受管節點上執行任何掃描或安裝。

程式碼範例

下列 Python 程式碼範例顯示如何產生修補基準覆寫。

```
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)
```

這會產生修補基準覆寫，如以下所示。

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

修補基準

本節中的主題提供在受管節點上執行 `Scan`或 `Install`操作 AWS Systems Manager時Patch Manager，修補基準如何在 中的工具中運作的相關資訊。

主題

- 預先定義和自訂的修補基準
- 已核准與遭拒的修補程式清單的套件名稱格式
- 修補程式群組
- 在 Windows Server 上由 Microsoft 發行的修補應用程式

預先定義和自訂的修補基準

Patch Manager是 中的工具 AWS Systems Manager，可為 支援的每個作業系統提供預先定義的修補基準Patch Manager。您可以依照目前設定的方式使用這些基準 (您無法自訂)，也可以建立自己的自訂修補基準。自訂修補基準可讓您進一步控制為您的環境核准或拒絕哪些修補程式。此外，預先定義的基準會將 `Unspecified` 的合規層級指派給使用這些基準安裝的所有修補程式。針對要指派的合規值，您可以建立預先定義基準的複本，並指定要指派給修補程式的合規值。如需詳細資訊，請參閱自訂基準及使用自訂修補基準。

> **Note**
>
> 無論您使用哪種方法或組態類型進行修補操作，此主題中的資訊都適用：
>
> - 在 Quick Setup 中設定的修補程式政策
> - 在 Quick Setup 中設定的主機管理選項
> - 用來執行修補程式 `Scan` 或 `Install` 任務的維護時段

- 隨需 Patch now (立即修補) 操作

主題

- 預先定義的基準
- 自訂基準

預先定義的基準

下表說明 Patch Manager 提供的預先定義修補基準。

如需有關 Patch Manager 支援各作業系統的哪些版本的詳細資訊，請參閱 Patch Manager 先決條件。

名稱	支援的作業系統	詳細資訊
AWS-AlmaLinuxDefaultPatchBaseline	AlmaLinux	核准所有被歸類為「安全性」以及嚴重性等級為「關鍵」或「重要」的作業系統修補程式。同時核准被歸類為「Bugfix」的所有修補程式。在發行或更新 7 天後，修補程式將自動核准。[1]
AWS-AmazonLinuxDefaultPatchBaseline	Amazon Linux 1	核准所有被歸類為「安全性」以及嚴重性等級為「關鍵」或「重要」的作業系統修補程式。同時自動核准所有被歸類為「Bugfix」的修補程式。在發行或更新 7 天後，修補程式將自動核准。[1]
AWS-AmazonLinux2DefaultPatchBaseline	Amazon Linux 2	核准所有被歸類為「安全性」以及嚴重性等級為「關鍵」或「重要」的作業系統修補程式。同時核准所有被歸類為「Bugfix」的修補程式。在發

名稱	支援的作業系統	詳細資訊
		行後 7 日，修補程式將自動核准。[1]
AWS-AmazonLinux2022DefaultPatchBaseline	Amazon Linux 2022	核准所有被歸類為「安全性」以及嚴重性等級為「關鍵」或「重要」的作業系統修補程式。在發行後七日，修補程式將自動核准。同時在修補程式發布七天之後，核准被歸類「Bugfix」的所有修補程式。
AWS-AmazonLinux2023DefaultPatchBaseline	Amazon Linux 2023	核准所有被歸類為「安全性」以及嚴重性等級為「關鍵」或「重要」的作業系統修補程式。在發行後七日，修補程式將自動核准。同時在修補程式發布七天之後，核准被歸類「Bugfix」的所有修補程式。
AWS-CentOSDefaultPatchBaseline	CentOS 和 CentOS Stream	在更新可用 7 天之後核准所有更新，包括非安全性更新。
AWS-DebianDefaultPatchBaseline	Debian Server	立即核准所有作業系統安全性相關且優先順序為「必要」、「重要」、「標準」、「選用」或「Extra」的修補程式。立刻核准是因為儲存庫未提供可靠的發行日期。
AWS-MacOSDefaultPatchBaseline	macOS	核准所有被歸類為「安全」的作業系統修補程式。同時核准包含目前更新的所有套件。

名稱	支援的作業系統	詳細資訊
AWS-OracleLinuxDefaultPatchBaseline	Oracle Linux	核准所有被歸類為「安全性」以及嚴重性等級為「重要」或「中等」的作業系統修補程式。同時在修補程式發行 7 天之後，核准被歸類為 "Bugfix" 的所有修補程式。在發行或更新 7 天後，修補程式將自動核准。[1]
AWS-DefaultRaspbianPatchBaseline	Raspberry Pi OS	立即核准所有作業系統安全性相關且優先順序為「必要」、「重要」、「標準」、「選用」或「Extra」的修補程式。立刻核准是因為儲存庫未提供可靠的發行日期。
AWS-RedHatDefaultPatchBaseline	Red Hat Enterprise Linux (RHEL)	核准所有被歸類為「安全性」以及嚴重性等級為「關鍵」或「重要」的作業系統修補程式。同時核准被歸類為「Bugfix」的所有修補程式。在發行或更新 7 天後，修補程式將自動核准。[1]
AWS-RockyLinuxDefaultPatchBaseline	Rocky Linux	核准所有被歸類為「安全性」以及嚴重性等級為「關鍵」或「重要」的作業系統修補程式。同時核准被歸類為「Bugfix」的所有修補程式。在發行或更新 7 天後，修補程式將自動核准。[1]

名稱	支援的作業系統	詳細資訊
AWS-SuseDefaultPatchBaseline	SUSE Linux Enterprise Server (SLES)	核准所有在被歸類為「安全性」以及嚴重性為「關鍵」或「重要」的作業系統修補程式。在發行或更新 7 天後，修補程式將自動核准。[1]
AWS-UbuntuDefaultPatchBaseline	Ubuntu Server	立即核准所有作業系統安全性相關且優先順序為「必要」、「重要」、「標準」、「選用」或「Extra」的修補程式。立刻核准是因為儲存庫未提供可靠的發行日期。
AWS-DefaultPatchBaseline	Windows Server	核准所有被歸類為「CriticalUpdates」或「SecurityUpdates」以及 MSRC 嚴重性為「關鍵」或「重要」的 Windows Server 作業系統修補程式。在發佈或更新 7 天後，修補程式將自動核准。[2]
AWS-WindowsPredefinedPatchBaseline-OS	Windows Server	核准所有被歸類為「CriticalUpdates」或「SecurityUpdates」以及 MSRC 嚴重性為「關鍵」或「重要」的 Windows Server 作業系統修補程式。在發佈或更新 7 天後，修補程式將自動核准。[2]

名稱	支援的作業系統	詳細資訊
AWS-WindowsPredefinedPatchBaseline-OS-Applications	Windows Server	在修補程式發佈七天之後，核准所有被歸類為「CriticalUpdates」或「SecurityUpdates」以及 MSRC 嚴重性為「關鍵」或「重要」的 Windows Server 作業系統修補程式。對 Microsoft 發行的應用程式核准所有的修補程式。作業系統和應用程式的修補程式會在發佈或更新 7 天後自動核准。[2]

[1] 對於 Amazon Linux 1 和 Amazon Linux 2，修補程式經自動核准前的 7 天等待時間從 `updateinfo.xml` 中的 `Updated Date` 值開始計算，而不是從 `Release Date` 值計算。各種因素會影響 `Updated Date` 值。其他作業系統會以不同的方式處理發行和更新日期。如需詳細資訊來協助您避免因自動核准延遲而導致非預期結果，請參閱套件發行日期和更新日期的計算方式。

[2] 對於 Windows Server，預設基準包括 7 天自動核准延遲。若要在發佈後 7 天內安裝修補程式，必須建立自訂基準。

自訂基準

使用下列資訊，協助您建立自訂修補基準，以符合您的修補目標。

主題

- 在自訂基準中使用自動核准
- 建立修補基準的其他資訊

在自訂基準中使用自動核准

如果您建立自己的修補基準，您可以使用以下類別來選擇自動核准哪些修補程式。

- 作業系統：Windows Server、Amazon Linux、Ubuntu Server 等。
- 產品名稱 (適用於作業系統)：例如，RHEL 6.5、Amazon Linux 2014.09、Windows Server 2012、Windows Server 2012 R2 等。

- 產品名稱 (僅適用於在 Windows Server 由 Microsoft 發行的應用程式)：例如，Word 2016、BizTalk Server 等。
- 分類：例如，重要更新、安全性更新等。
- 嚴重性：例如關鍵、重要等。

對於您建立的每個核准規則，您可以選擇指定自動核准延遲，或指定修補程式核准截止日期。

> **Note**
>
> 因為無法可靠地判斷 Ubuntu Server 更新套件的發行日期，此作業系統不支援自動核准選項。

自動核准延遲是修補程式發行或最後更新之後，在自動核准修補程式進行修補之前的等待天數。例如，如果您使用 `CriticalUpdates` 分類來建立規則，並將自動核准延遲設定為 7 天，則 7 月 7 日發行的新的關鍵修補程式將在 7 月 14 日自動核准。

如果 Linux 儲存庫未提供套件的發行日期資訊， Patch Manager會使用套件的建置時間做為 Amazon Linux 1、Amazon Linux 2、 SUSE Linux Enterprise Server(SLES)、 Red Hat Enterprise Linux(RHEL) 和 CentOS 自動核准日期規格的日期。如果無法判斷套件的建置時間， Patch Manager會使用預設日期 1970 年 1 月 1 日。這會導致Patch Manager略過修補程式基準中設定為在 1970 年 1 月 1 日之後核准修補程式的任何日期的任何自動核准日期規格。

當您指定自動核准截止日期時，Patch Manager 會自動套用在該日期當天或之前發行或最後更新的所有修補程式。例如，如果您指定 2023 年 7 月 7 日作為截止日期，則不會自動安裝在 2023 年 7 月 8 日或之後發行或最後更新的修補程式。

當您建立自訂修補基準時，您可以指定此修補基準所核准之修補程式的合規嚴重性等級，例如 `Critical` 或 `High`。如果任何核准之修補程式的修補程式狀態回報為 `Missing`，則修補基準的整體報告合規嚴重性是您指定的嚴重性等級。

建立修補基準的其他資訊

建立修補基準時，請謹記下列事項：

- Patch Manager 為每個支援的作業系統提供一個預設修補基準。除非您建立自己的修補基準，並將其指定為相應作業系統類型的預設基準，否則這些預先定義的修補基準會用於每種作業系統類型的預設修補基準。

> **Note**
>
> 對於 Windows Server，則會提供三個預先定義的修補基準。修補基準 `AWS-DefaultPatchBaseline` 和 `AWS-WindowsPredefinedPatchBaseline-OS` 僅支援 Windows 作業系統本身的作業系統更新。`AWS-DefaultPatchBaseline` 會用作 Windows Server 受管節點的預設修補基準，除非您指定了不同的修補基準。這兩個修補基準中的組態設定是相同的。兩者中較新的 `AWS-WindowsPredefinedPatchBaseline-OS` 是為了區分它與 Windows Server 第三方預先定義修補基準而建立的。修補基準 `AWS-WindowsPredefinedPatchBaseline-OS-Applications` 可用來將修補程式套用至 Windows Server 作業系統和 Microsoft 發行的支援應用程式。

- 依預設，Windows Server 2019 和 Windows Server 2022 會移除更新，而這些更新會由稍後的更新取代。因此，如果您在 Windows Server 修補基準中使用 `ApproveUntilDate` 參數，但 `ApproveUntilDate` 參數中選取的日期早於最新修補程式的日期，則修補操作執行時不會安裝新的修補程式。如需有關 Windows Server 修補規則的詳細資訊，請參閱 如何選取安全性修補程式 中的 Windows Server 索引標籤。

  這表示即使可能未安裝上個月的關鍵修補程式，受管節點在 Systems Manager 操作方面也符合規範。使用 `ApproveAfterDays` 參數時，可能會發生相同的情況。由於 Microsoft 取代了修補程式行為，可以設定數字 (通常大於 30 天)，這樣，若在 `ApproveAfterDays` 的天數之前發行 Microsoft 最新的可用修補程式，則永遠不會安裝 Windows Server 的修補程式。

- 僅限 Windows Server ，未經修補程式基準核准的可用安全更新修補程式可以具有合規值 或 `Compliant`或 `Non-Compliant`，如自訂修補程式基準中所定義。

  當您建立或更新修補程式基準時，您可以選擇要指派給可用但未核准的安全修補程式狀態，因為這些修補程式不符合修補程式基準中指定的安裝條件。例如，如果您已指定在安裝修補程式發行之前等待很長的時間，則可以略過您可能想要安裝的安全修補程式。如果在指定的等待期間發行修補程式的更新，安裝修補程式的等待期間會重新開始。如果等待期間過長，可以釋出多個版本的修補程式，但永遠不會安裝。

  使用主控台建立或更新修補程式基準，您可以在可用安全性更新合規狀態欄位中指定此選項。使用 AWS CLI 執行 create-patch-baseline或 update-patch-baseline命令，您可以在 `available-security-updates-compliance-status` 參數中指定此選項。

- 對於內部部署伺服器和虛擬機器 (VM)，Patch Manager 會嘗試使用您的自訂預設修補基準。如果不存在自訂預設修補基準，系統將使用為對應作業系統預先定義的修補基準。
- 如果修補程式已在相同的修補基準中同時列為核准與拒絕，修補程式將遭到拒絕。

- 一個受管節點只能有一個為其定義的修補基準。
- 可新增至修補基準之核准與拒絕修補程式清單的套件名稱格式，取決於您要修補之作業系統的類型。

  如需已核准修補程式和已拒絕修補程式清單之可接受格式的相關資訊，請參閱 已核准與遭拒的修補程式清單的套件名稱格式。
- 如果您在 Quick Setup 中使用修補程式政策組態，則您對自訂修補基準所做的更新會每小時與 Quick Setup 同步一次。

  如果刪除修補程式政策中參照的自訂修補基準，則修補程式政策的 Quick Setup Configuration details (組態詳細資訊) 頁面上會顯示橫幅。此橫幅會通知您修補程式政策參照修補基準不再存在，而後續的修補操作將會失敗。在此情況下，請返回到 Quick Setup Configurations (組態) 頁面，選取 Patch Manager 組態，然後選擇 Actions (動作)、Edit configuration (編輯組態)。刪除的修補基準名稱會反白顯示，您必須為受影響的作業系統選取新的修補基準。

如需有關建立修補基準的詳細資訊，請參閱使用自訂修補基準與教學課程：使用 修補伺服器環境 AWS CLI。

已核准與遭拒的修補程式清單的套件名稱格式

可新增至核准與拒絕修補程式清單的套件名稱格式，取決於您要修補之作業系統的類型。

適用於 Linux 作業系統的套件名稱格式

您可為修補基準中的核准與拒絕修補程式指定的格式，依據 Linux 類型而有不同。具體來說，支援的格式取決於 Linux 作業系統類型所使用的套件管理工具。

主題

- Amazon Linux 1、Amazon Linux 2、Amazon Linux 2022、Amazon Linux 2023、CentOS、Oracle Linux 及 Red Hat Enterprise Linux (RHEL)
- Debian Server、Raspberry Pi OS (先前為 Raspbian) 和 Ubuntu Server
- SUSE Linux Enterprise Server (SLES)

Amazon Linux 1、Amazon Linux 2、Amazon Linux 2022、Amazon Linux 2023、CentOS、Oracle Linux 及 Red Hat Enterprise Linux (RHEL)

套件管理工具：YUM、Amazon Linux 2022、Amazon Linux 2023 和 RHEL 8 除外，其使用 DNF 做為套件管理工具

核准的修補程式：對於核准的修補程式，您可以指定以下任何項目：

- Bugzilla ID，格式為 1234567 (系統將僅有數字的字串處理為 Bugzilla ID。)
- CVE ID，格式為 CVE-2018-1234567
- 諮詢 ID，格式範例為 RHSA-2017:0864 及 ALAS-2018-123
- 使用一個或多個可用於套件命名的元件建構的套件名稱。為了說明，對於名為 dbus.x86_64:1:1.12.28-1.amzn2023.0.1 的套件，元件如下所示：
  - name: dbus
  - architecture: x86_64
  - epoch: 1
  - version: 1.12.28
  - release: 1.amzn2023.0.1

  支援具有下列結構的套件名稱：
  - 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

  一些範例：
  - 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
- 我們也支援使用上述格式之單一萬用字元的套件名稱元件，例如下列所示：
  - dbus*

- dbus-1.12.2*
- dbus-*:1.12.28-1.amzn2023.0.1.x86_64

拒絕的修補程式：對於拒絕的修補程式，您可以指定以下任何項目：

- 使用一個或多個可用於套件命名的元件建構的套件名稱。為了說明，對於名為 dbus.x86_64:1:1.12.28-1.amzn2023.0.1 的套件，元件如下所示：
  - name: dbus
  - architecture; x86_64
  - epoch: 1
  - version: 1.12.28
  - release: 1.amzn2023.0.1

  支援具有下列結構的套件名稱：
  - 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

  一些範例：
  - 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

- 我們也支援使用上述格式之單一萬用字元的套件名稱元件，例如下列所示：
  - dbus*
  - dbus-1.12.2*
  - dbus-*:1.12.28-1.amzn2023.0.1.x86_64

Debian Server、Raspberry Pi OS (先前為 Raspbian) 和 Ubuntu Server

套件管理工具：APT

核准的修補程式與拒絕修補程式：對於核准與拒絕的修補程式，請指定以下項目：

- 套件名稱，格式為 ExamplePkg33

> **Note**
>
> 對於 Debian Server 清單、Raspberry Pi OS 清單和 Ubuntu Server 清單，請勿包含諸如架構或版本之類的元素。例如，您指定套件名稱 ExamplePkg33 將下列所有項目包含在修補程式清單中：
>
> - ExamplePkg33.x86.1
> - ExamplePkg33.x86.2
> - ExamplePkg33.x64.1
> - ExamplePkg33.3.2.5-364.noarch

SUSE Linux Enterprise Server (SLES)

套件管理工具：Zypper

核准的修補程式 與 拒絕修補程式：對於核准與拒絕的修補程式清單，可指定以下任何項目：

- 完整套件名稱，格式範例：
  - SUSE-SLE-Example-Package-12-2018-123
  - example-pkg-2018.11.4-46.17.1.x86_64.rpm
- 套件名稱含一個萬用字元，例如：
  - SUSE-SLE-Example-Package-12-2018-*
  - example-pkg-2018.11.4-46.17.1.*.rpm

macOS 的套件名稱格式

支援的套件管理工具：softwareupdate、installer、Brew、Brew Cask

核准的修補程式 與 拒絕修補程式：對於核准與拒絕的修補程式清單，您可以使用以下格式指定完整套件名稱：

- `XProtectPlistConfigData`
- `MRTConfigData`

macOS 的已核准和已拒絕修補程式清單不支援萬用字元。

適用於 Windows 作業系統的套件名稱格式

對於 Windows 作業系統，請使用修補程式的 Microsoft 知識庫 ID 和 Microsoft 資訊安全佈告欄 ID，例如：

```
KB2032276,KB2124261,MS10-048
```

修補程式群組

> **Note**
>
> 修補程式群組不會用於基於修補程式政策的修補操作。如需有關使用修補程式政策的資訊，請參閱 Quick Setup中的修補程式政策組態。
> 在 2022 年 12 月 22 日發行修補程式政策支援之前，尚未使用修補程式群組的帳戶區域對不支援修補程式群組功能。在此日期之前開始使用修補程式群組的帳戶區域對中，修補程式群組功能仍然可用。

您可以使用修補程式群組，將受管節點與 中的Patch Manager工具 中的特定修補程式基準建立關聯 AWS Systems Manager。修補程式群組會協助您根據相關的修補基準規則，確保您部署適當的修補程式至正確的節點。修補程式群組也能協助您避免在充分測試之前即部署修補程式。例如，您可以為不同的環境建立修補程式群組 (例如，開發、測試和生產) 並將每個修補程式群組註冊到適當的修補基準。

當您執行 `AWS-RunPatchBaseline` 或其他 SSM 命令文件進行修補時，您可以使用受管節點的 ID 或標籤將目標設為目標。 SSM Agent Patch Manager 然後，根據您新增至受管節點的修補程式群組值，評估要使用的修補程式基準。

使用標籤定義修補程式群組

您可以使用套用至混合多雲端環境中 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體non-EC2 節點的標籤來建立修補程式群組。請注意下列有關使用修補程式群組標籤的詳細資訊：

- 必須使用標籤索引鍵Patch Group或PatchGroup套用至受管節點來定義修補程式群組。為修補程式基準註冊修補程式群組時，為這兩個金鑰指定的任何相同值都會解譯為相同群組的一部分。例如，假設您已使用下列第一個索引鍵/值對標記五個節點，並使用第二個標記五個節點：
  - key=PatchGroup,value=DEV
  - key=Patch Group,value=DEV

  建立基準的 Patch Manager命令會根據值 ，將這 10 個受管節點合併為單一群組DEV。為修補程式群組建立修補程式基準的 命令 AWS CLI 同等項目如下：

  ```
  aws ssm register-patch-baseline-for-patch-group \
      --baseline-id pb-0c10e65780EXAMPLE \
      --patch-group DEV
  ```

  將來自不同索引鍵的值合併為單一目標，對於此Patch Manager命令是唯一的，用於建立新的修補程式群組，且不受其他 API 動作支援。例如，如果您使用具有相同值的 PatchGroup和 Patch Group金鑰執行send-command動作，則您鎖定的節點集會完全不同：

  ```
  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"
  ```
- 標籤型目標有限制。的每個目標陣列最多可SendCommand包含五個鍵值對。
- 我們建議您僅選擇其中一個標籤金鑰慣例，可以是 PatchGroup（不含空格） 或 Patch Group（含空格）。不過，如果您在執行個體的 EC2 執行個體中繼資料中允許標籤，則必須使用 PatchGroup。
- 該金鑰區分大小寫。您可以指定任何值，以協助您識別該群組中的資源並將其設為目標，例如「Web 伺服器」或「US-EAST-PROD」，但金鑰必須是 Patch Group或 PatchGroup。

在您建立修補程式群組並標記受管節點之後，您可以使用修補基準註冊修補程式群組。使用修補基準註冊修補程式群組，可確保修補程式群組中的節點使用相關修補基準中定義的規則。

有關如何建立修補程式群組，以及將修補程式群組與修補基準建立關聯的詳細資訊，請參閱 建立和管理修補程式群組 和 Add a patch group to a patch baseline(新增修補程式群組至修補基準)。

若要檢視使用 AWS Command Line Interface (AWS CLI) 建立修補基準與修補程式群組的範例，請參閱 教學課程：使用 修補伺服器環境 AWS CLI。如需有關 Amazon EC2 標籤的詳細資訊，請參閱《Amazon EC2 使用者指南》https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/Using_Tags.html中的 Tag your Amazon EC2 resources。

運作方式

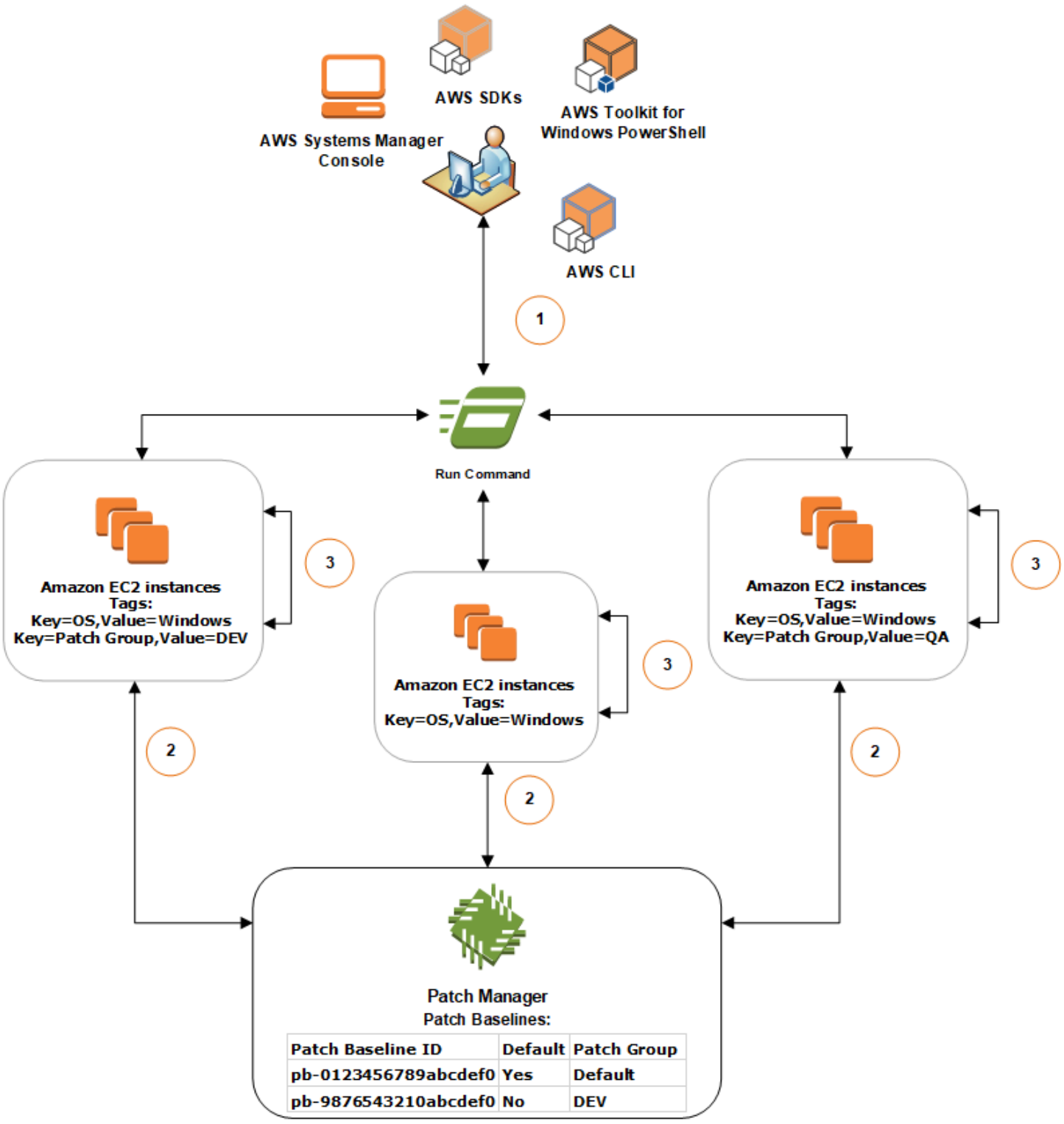
當系統執行將修補基準套用於受管節點的任務時，SSM Agent 會驗證是否為該節點定義了修補程式群組值。如果節點已指派給修補程式群組，則 Patch Manager 會驗證該群組註冊了哪一個修補基準。如果找到了該群組的修補基準，則 Patch Manager 會通知 SSM Agent 使用關聯的修補基準。如果節點未針對修補程式群組進行設定，則 Patch Manager 會自動通知 SSM Agent 已使用目前設定的預設修補基準。

> **Important**
>
> 受管節點只能存在於一個修補程式群組中。
> 一個修補程式群組只能為每個作業系統類型註冊一個修補基準。
> 如果在執行個體上啟用 Allow tags in instance metadata (允許執行個體中繼資料中的標籤) 選項，則您不得將 `Patch Group` 標籤 (有空格) 套用至 Amazon EC2 執行個體。允許執行個體中繼資料中的標籤可防止標籤鍵名稱含空格。如果您已在 EC2 執行個體中繼資料中允許標籤，則必須使用標籤索引鍵 `PatchGroup` (不留空格)。

圖 1：修補操作處理流程一般範例

下圖顯示 Systems Manager 將 Run Command 任務傳送到您的伺服器機群，以使用 Patch Manager 進行修補時執行程序的一般範例。這些程序會確定要在修補操作中使用的修補基準。(當維護時段設定為使用 Patch Manager 向修補程式傳送命令時，將使用類似的程序。)

完整程序見圖例下方的說明。



在此範例中，我們有三個 Windows Server EC2 執行個體群組已套用以下標籤：

EC2 執行個體群組	標籤
Group 1	key=OS,value=Windows key=PatchGroup,value=DEV
Group 2	key=OS,value=Windows
Group 3	key=OS,value=Windows key=PatchGroup,value=QA

在這個範例中，我們也有這兩個 Windows Server 修補基準：

修補基準 ID	預設	關聯的修補程式群組
pb-0123456789abcdef0	是	Default
pb-9876543210abcdef0	否	DEV

使用 中的Run Command工具 AWS Systems Manager和 掃描或安裝修補程式的一般程序Patch Manager如下：

1. 傳送命令至修補程式：使用 Systems Manager 主控台、SDK、 AWS Command Line Interface (AWS CLI)，或使用文件 AWS Tools for Windows PowerShell 傳送Run Command任務AWS-RunPatchBaseline。該圖顯示了透過定位標籤 key=OS,value=Windows 來修補受管執行個體的 Run Command 任務。
2. 修補基準判定：SSM Agent 會驗證套用於 EC2 執行個體的修補程式群組標籤，並查詢 Patch Manager 以取得對應的修補基準。
   - 符合與修補基準關聯的修補程式群組值：
     1. 安裝在群組 1 中的 EC2 執行個體上的 SSM Agent 會接收步驟 1 中發出的命令以開始修補操作。SSM Agent 會驗證 EC2 執行個體是否已套用修補程式群組標籤值 DEV，並向 Patch Manager 查詢關聯的修補基準。
     2. Patch Manager 會確認修補基準 pb-9876543210abcdef0 已關聯修補程式群組 DEV，並通知 SSM Agent。

3. SSM Agent 會根據在 pb-9876543210abcdef0 中設定的核准規則和例外狀況從 Patch Manager 擷取修補基準快照，然後繼續執行下一步驟。

- 無新增至執行個體的修補程式群組標籤：

1. 安裝在群組 2 中 EC2 執行個體上的 SSM Agent 會接收步驟 1 中發出的命令，以開始修補操作。SSM Agent 會驗證 EC2 執行個體未套用的 Patch Group 或 PatchGroup 標籤，因此，SSM Agent 會查詢 Patch Manager 以取得預設 Windows 修補基準。
2. Patch Manager 會確認預設的 Windows Server 修補基準是 pb-0123456789abcdef0 並通知 SSM Agent。
3. SSM Agent 會根據在預設修補程式基線 pb-0123456789abcdef0 中設定的核准規則和例外狀況，從 Patch Manager 擷取修補程式基線快照，然後繼續執行下一步驟。

- 沒有符合與修補基準關聯的修補程式群組值：

1. 安裝在群組 3 中的 EC2 執行個體上的 SSM Agent 會接收步驟 1 中發出的命令以開始修補操作。SSM Agent 會驗證 EC2 執行個體是否已套用修補程式群組標籤值 QA，並向 Patch Manager 查詢關聯的修補基準。
2. Patch Manager 未找到與修補程式群組 QA 相關聯的修補基準。
3. Patch Manager 會通知 SSM Agent 以使用預設的 Windows 修補基準 pb-0123456789abcdef0。
4. SSM Agent 會根據在預設修補基準 pb-0123456789abcdef0 中設定的核准規則和例外狀況，從 Patch Manager 擷取修補基準快照，然後繼續執行下一步驟。

3. 修補程式掃描或安裝：在決定欲使用的適當修補基準後，SSM Agent 會根據步驟 1 中指定的作業值開始掃描或安裝修補程式。掃描或安裝的修補程式，是由 Patch Manager 提供的修補基準快照中所定義的核准規則和修補程式的例外狀況所判斷。

詳細資訊

- 修補程式合規狀態值

在 Windows Server 上由 Microsoft 發行的修補應用程式

使用本主題中的資訊，協助您準備Windows Server使用 Patch Manager中的工具 修補應用程式 AWS Systems Manager。

Microsoft 應用程式修補

對 Windows Server 受管節點上應用程式的修補支援僅限於 Microsoft 發行的應用程式。

> **Note**
>
> 在某些情況下，Microsoft 會針對未指定更新日期和時間的應用程式發行修補程式。在這些情況下，預設會提供 01/01/1970 的更新日期和時間。

修補由 Microsoft 發行之應用程式的修補基準

對於 Windows Server，則會提供三個預先定義的修補基準。修補基準 AWS-DefaultPatchBaseline 和 AWS-WindowsPredefinedPatchBaseline-OS 僅支援 Windows 作業系統本身的作業系統更新。AWS-DefaultPatchBaseline 會用作 Windows Server 受管節點的預設修補基準，除非您指定了不同的修補基準。這兩個修補基準中的組態設定是相同的。兩者中較新的 AWS-WindowsPredefinedPatchBaseline-OS 是為了區分它與 Windows Server 第三方預先定義修補基準而建立的。修補基準 AWS-WindowsPredefinedPatchBaseline-OS-Applications 可用來將修補程式套用至 Windows Server 作業系統和 Microsoft 發行的支援應用程式。

您也可以建立自訂修補基準以在 Windows Server 電腦上更新 Microsoft 發行的應用程式。

對於 Microsoft 在內部部署伺服器、邊緣裝置、VM 和其他非 EC2 節點上發行之應用程式的修補支援

若要修補 Microsoft 在虛擬機器 (VM) 和其他非 EC2 受管節點上發行的應用程式，您必須開啟 advanced-instances 方案。使用 advanced-instance 方案會產生費用。但是，修補由 Microsoft 在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上發行的應用程式無須另外付費。如需詳細資訊，請參閱設定執行個體方案。

「其他 Microsoft 產品」的 Windows 更新選項

為了讓 Patch Manager 能夠在您的 Windows Server 受管節點上修補 Microsoft 發行的應用程式，必須在受管節點上啟用 Windows 更新選項 Give me updates for other Microsoft products when I update Windows (當我更新 Windows 時，為我提供其他 Microsoft 產品的更新)。

如需在單一受管節點上允許此選項的相關資訊，請參閱使用 Microsoft Update 更新 Office (位於 Microsoft Support 網站)。

針對執行 Windows Server 2016 及更新版本的受管節點機群，您可以使用群組政策物件 (GPO) 來開啟設定。在群組政策管理編輯器中，移至 Computer Configuration (電腦組態)、Administrative Templates (管理範本)、Windows Components (Windows 元件)、Windows Updates (Windows 更新)，然後選擇 Install updates for other Microsoft products (為其他 Microsoft 產品安裝更新)。我們還建議您使用其他參數來設定 GPO，以防止 Patch Manager 之外的計劃外自動更新和重新開機。如需詳細資訊，請參閱 Microsoft 技術文件網站上的在非作用中目錄環境設定自動更新。

針對執行 Windows Server 2012 或 2012 R2 的受管節點機群，您可以使用指令碼來開啟選項，如 Microsoft Docs 部落格網站上的透過指令碼啟用和停用 Windows 7 的 Microsoft 更新所述。例如，您可以執行下列操作：

1. 將部落格文章中的指令碼儲存在檔案中。
2. 將檔案上傳至 Amazon Simple Storage Service (Amazon S3) 儲存貯體或其他可存取的位置。
3. 使用 中的Run Command工具 AWS Systems Manager，使用 Systems Manager 文件 (SSM 文件) `AWS-RunPowerShellScript` 搭配類似以下的命令，在您的受管節點上執行指令碼。

```
Invoke-WebRequest `
    -Uri "https://s3.aws-api-domain/amzn-s3-demo-bucket/script.vbs" `
    -Outfile "C:\script.vbs" cscript c:\script.vbs
```

最低參數要求

若要在自定修補基準中包含 Microsoft 發行的應用程式，您至少必須指定要修補的產品。following AWS Command Line Interface (AWS CLI) 命令示範修補產品的最低需求，例如 Microsoft Office 2016。

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 應用程式產品系列，則您指定的每個產品都必須是所選產品系列的受支援成員。例如，若要修補產品「Active Directory Rights Management Services Client 2.0」，您必須指定其

產品系列為「Active Directory」而非「Office」或「SQL Server」。下列 AWS CLI 命令示範 產品系列和 產品的配對。

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**
>
> 如果您收到有關不相符產品與家庭配對的錯誤訊息，請參閱 問題：產品系列/產品配對不相符，以取得解決問題的協助。

在 Amazon Linux 2 受管節點上使用 Kernel Live Patching

適用於 Amazon Linux 2 的 Kernel Live Patching 可讓您將安全性漏洞和重大錯誤修補程式套用至執行中的 Linux 核心，而不需要重新啟動或中斷執行中的應用程式。這可讓您從改善的服務和應用程式可用性中受益，同時保持基礎設施的安全和最新狀態。在 Amazon EC2 執行個體、 AWS IoT Greengrass 核心裝置和執行 Amazon Linux 2 的內部部署虛擬機器上支援 Kernel Live Patching。

如需 的一般資訊Kernel Live Patching，請參閱《Amazon Linux Kernel Live Patching AL22。

Kernel Live Patching 開啟 Amazon Linux 2 受管節點後，您可以使用 中的Patch Manager工具 AWS Systems Manager，將核心即時修補程式套用至受管節點。使用 Patch Manager 是在節點上使用現有 yum 工作流程來套用更新的替代方法。

開始之前

若要使用 Patch Manager 將核心即時修補程式套用至您的 Amazon Linux 2 受管節點，請確定您的節點是以正確的架構和核心版本為基礎。如需相關資訊，請參閱《Amazon EC2 使用者指南》中的 Supported configurations and prerequisites。

主題

- 使用 Patch Manager 的 Kernel Live Patching
- 使用 Patch Manager 的 Kernel Live Patching 如何運作
- 使用 Run Command 開啟 Kernel Live Patching
- 使用 Run Command 套用核心即時修補程式
- 使用 Run Command 關閉 Kernel Live Patching

使用 Patch Manager 的 Kernel Live Patching

更新核心版本

套用核心即時修補程式更新後，您不需要重新啟動受管節點。不過， AWS 會在 Amazon Linux 2 核心版本發行後提供最多三個月的核心即時修補程式。在 3 個月期間過後，您必須更新至較新的核心版本，才能繼續接收核心即時修補程式。我們建議您使用維護時段，排程至少每三個月重新啟動節點一次，以提示核心版本更新。

解除安裝核心即時修補程式

無法使用 Patch Manager 解除安裝核心即時修補程式。您可以改為關閉 Kernel Live Patching，這會為已套用核心即時修補程式移除 RPM 套件。如需詳細資訊，請參閱使用 Run Command 關閉 Kernel Live Patching。

核心合規

在某些情況下，從目前核心版本的即時修補程式安裝所有 CVE 修正程式，可能會使該核心達到與較新核心版本相同的合規狀態。發生這種情況時，會將較新的版本報告為 `Installed`，並將受管節點報告為 `Compliant`。但是，對於較新的核心版本，不會報告任何安裝時間。

一個核心即時修補程式，多個 CVE

如果核心即時修補程式可處理多個 CVE，且這些 CVE 具有各種分類和嚴重性值，則只會報告 CVE 中最高的分類和嚴重性。

本節的其餘部分說明如何使用 Patch Manager，將核心即時修補程式套用至符合這些需求的受管節點。

使用 Patch Manager 的 Kernel Live Patching 如何運作

AWS 為 Amazon Linux 2 發行兩種類型的核心即時修補程式：安全性更新和錯誤修正。若要套用這些類型的修補程式，您可以使用修補基準文件，僅針對下表所列分類與嚴重性。

分類	嚴重性
`Security`	`Critical`, `Important`
`Bugfix`	`All`

您可以建立自訂修補基準，僅針對這些修補程式，或使用預先定義的 `AWS-AmazonLinux2DefaultPatchBaseline` 修補基準。換句話說，您可以使用 `AWS-AmazonLinux2DefaultPatchBaseline` 搭配啟用 Kernel Live Patching 的 Amazon Linux 2 受管節點，如此將套用核心即時更新。

> **Note**
>
> `AWS-AmazonLinux2DefaultPatchBaseline` 組態會指定在自動安裝修補程式之前、發行或最後更新修補程式之後的 7 天等待期。如果不想等待七天，讓核心即時修補程式自動核准，則您可以建立並使用自訂修補基準。在修補基準中，您可以指定非自動核准等待期間，或指定較短或較長的等待期間。如需詳細資訊，請參閱使用自訂修補基準。

我們建議您採用下列策略，以核心即時更新修補您的受管節點：

1. 在您的 Amazon Linux 2 受管節點上開啟 Kernel Live Patching。
2. 使用 中的Run Command工具 AWS Systems Manager，使用預先定義`AWS-AmazonLinux2DefaultPatchBaseline`或自訂修補程式基準，在您的受管節點上執行`Scan`操作，該基準也僅針對嚴重性分類為 `Critical`和 的更新`Important`，以及`Bugfix`嚴重性為的`Security`更新`All`。
3. 使用 中的工具 Compliance AWS Systems Manager來檢閱是否報告任何已掃描的受管節點的修補不合規。若是如此，請檢視節點合規詳細資訊，以判斷受管節點是否遺漏任何核心即時修補程式。

4. 若要安裝遺漏的核心即時修補程式，請使用 Run Command 搭配您之前指定的相同修補基準，但這次請執行 `Install` 操作而非 `Scan` 操作。

   由於安裝核心即時修補程式並不需要重新開機，因此您可以為此操作選擇 `NoReboot` 重新開機選項。

   > **Note**
   >
   > 如果安裝在受管節點上的其他類型修補程式有需要重新啟動，或者您想要更新至較新的核心，仍然可以重新啟動受管節點。在這些情況下，請改選 `RebootIfNeeded` 重新開機選項。

5. 返回 合規以確認已安裝核心即時修補程式。

使用 Run Command 開啟 Kernel Live Patching

若要開啟 Kernel Live Patching，您可以在受管節點上執行 yum 命令，或使用 Run Command 和您建立的自訂 Systems Manager 文件 (SSM 文件)。

如需有關透過直接在受管節點上執行 yum 命令來關閉 Kernel Live Patching 的資訊，請參閱《Amazon EC2 使用者指南》中的啟用 Kernel Live Patching。

> **Note**
>
> 當您開啟「核心即時修補」時，如果受管節點上已經執行的核心早於 `kernel-4.14.165-131.185.amzn2.x86_64` (最低支援的版本)，則程序會安裝最新的可用核心版本並重新啟動受管節點。如果節點已在執行 `kernel-4.14.165-131.185.amzn2.x86_64` 或更新版本，則程序不會安裝較新的版本，也不會重新啟動節點。

使用 Run Command 開啟 Kernel Live Patching (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Run Command。
3. 選擇執行命令。

4. 在 Command document (命令文件) 清單中，選擇 SSM 文件 `AWS-ConfigureKernelLivePatching`。
5. 在 Command parameters (命令參數) 區段中，指定是否要在此操作中重新啟動受管節點。
6. 如需使用此頁面上其餘控制項的詳細資訊，請參閱從主控台執行命令。
7. 選擇執行。

開啟 Kernel Live Patching (AWS CLI)

- 在本機機器上執行以下命令。

  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"
  ```

  將 i*instance-id* 取代為您要啟用該功能的 Amazon Linux 2 受管節點 ID，例如 i-02573cafcfEXAMPLE。若要在多個受管節點上開啟此功能，您可以使用下列其中一種格式。

  - `--targets "Key=instanceids,Values=instance-id1,instance-id2"`
  - `--targets "Key=tag:tag-key,Values=tag-value"`

  如需可以在命令中使用之其他選項的相關資訊，請參閱《AWS CLI 命令參考》中的 send-command 一節。

使用 Run Command 套用核心即時修補程式

若要套用核心即時修補程式，您可以在受管節點上執行 `yum` 命令，或使用 Run Command 和 SSM 文件 `AWS-RunPatchBaseline`。

如需有關透過直接在受管節點上執行 yum 命令來套用核心即時修補程式的資訊，請參閱《Amazon EC2 使用者指南》中的套用核心即時修補程式。

使用 Run Command 套用核心即時修補程式 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Run Command。
3. 選擇執行命令。
4. 在 Command document (命令文件) 清單中，選擇 SSM 文件 `AWS-RunPatchBaseline`。
5. 在 Command parameters (命令參數) 區段中，執行以下其中一項：
   - 如果您要檢查是否有新的核心即時修補程式，請在 Operaions (操作) 中選擇 `Scan`。對於 Reboot Option (重新開機選項)，如果不希望受管節點在此操作之後重新啟動，請選擇 `NoReboot`。操作完成後，您可以在「合規」中檢查新的修補程式和合規狀態。
   - 如果您已檢查修補程式合規性，並準備好套用可用的核心即時修補程式，請在 Operation (操作) 中選擇 `Install`。對於 Reboot Option (重新開機選項)，如果不希望受管節點在此操作之後重新啟動，請選擇 `NoReboot`。
6. 如需使用此頁面上其餘控制項的詳細資訊，請參閱從主控台執行命令。
7. 選擇執行。

使用 Run Command 套用核心即時修補程式 (AWS CLI)

1. 若要檢查合規中的結果之前執行 `Scan` 操作，請從本機電腦執行下列命令。

   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
\"]}
```

如需可以在命令中使用之其他選項的相關資訊，請參閱《AWS CLI 命令參考》中的 send-command 一節。

2. 若要檢查 合規中的結果之後執行 `Install` 操作，請從本機電腦執行下列命令。

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\"]}
```

在上述兩個命令中，將 *instance-id* 取代為您要套用核心即時修補程式的 Amazon Linux 2 受管節點 ID，例如 i-02573cafcfEXAMPLE。若要在多個受管節點上開啟此功能，您可以使用下列其中一種格式。

- `--targets "Key=instanceids,Values=instance-id1,instance-id2"`
- `--targets "Key=tag:tag-key,Values=tag-value"`

如需可以在命令中使用之其他選項的相關資訊，請參閱《AWS CLI 命令參考》中的 send-command 一節。

使用 Run Command 關閉 Kernel Live Patching

若要關閉 Kernel Live Patching，您可以在受管節點上執行 yum 命令，或使用 Run Command 和自訂 SSM 文件 `AWS-ConfigureKernelLivePatching`。

> **Note**
>
> 如果您不再需要使用 Kernel Live Patching，您可以隨時停用它。在大多數情況下，不需要關閉功能。

如需有關透過直接在受管節點上執行 yum 命令來關閉 Kernel Live Patching 的資訊，請參閱《Amazon EC2 使用者指南》中的啟用 Kernel Live Patching。

> **Note**
>
> 當您關閉 Kernel Live Patching 時，程序會解除安裝 Kernel Live Patching 外掛程式，然後重新啟動受管節點。

使用 Run Command 關閉 Kernel Live Patching (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Run Command。
3. 選擇執行命令。
4. 在 Command document (命令文件) 清單中，選擇 SSM 文件 `AWS-ConfigureKernelLivePatching`。
5. 在 Command parameters (命令參數) 區段，指定所需的參數值。
6. 如需使用此頁面上其餘控制項的詳細資訊，請參閱從主控台執行命令。
7. 選擇執行。

關閉 Kernel Live Patching (AWS CLI)

- 使用類似以下的命令：

  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"
```

將 i*instance-id* 取代為您要啟用該功能的 Amazon Linux 2 受管節點 ID，例如 i-02573cafcfEXAMPLE。若要在多個受管節點上關閉此功能，您可以使用下列其中一種格式。

- `--targets "Key=instanceids,Values=instance-id1,instance-id2"`
- `--targets "Key=tag:tag-key,Values=tag-value"`

如需可以在命令中使用之其他選項的相關資訊，請參閱《AWS CLI 命令參考》中的 send-command 一節。

透過主控台使用 Patch Manager 資源與合規

若要使用 中的Patch Manager工具 AWS Systems Manager，請完成下列任務。本節將詳細說明這些任務。

1. 確認您使用的每個作業系統類型的 AWS 預先定義修補程式基準符合您的需求。如果沒有，請建立一個修補基準，為該受管節點類型定義一組標準修補程式，並將其設定為預設值。
2. 使用 Amazon Elastic Compute Cloud (Amazon EC2) 標籤，將受管節點組織到修補程式群組中 (選用，但建議使用)。
3. 執行以下任意一項：
   - （建議）在 Systems Manager 中的Quick Setup工具 中設定修補程式政策，可讓您根據整個組織的排程、組織單位子集或單一 來安裝缺少的修補程式 AWS 帳戶。如需詳細資訊，請參閱使用修補程式政策設定組織中執行個體的Quick Setup修補。
   - 在 Run Command 任務類型中建立使用 Systems Manager 文件 (SSM 文件) `AWS-RunPatchBaseline` 的維護時段。如需詳細資訊，請參閱教學課程：使用主控台建立修補維護時段。
   - 手動執行 Run Command 操作中的 `AWS-RunPatchBaseline`。如需詳細資訊，請參閱從主控台執行命令。

- 使用 Patch now (立即修補) 功能，依需求手動修補節點。如需詳細資訊，請參閱隨需修補受管節點。

4. 監控修補以確認合規並調查失敗。

主題

- 建立修補程式政策
- 檢視修補程式儀表板摘要
- 使用修補程式合規報告
- 隨需修補受管節點
- 使用修補基準
- 檢視可用修補程式
- 建立和管理修補程式群組
- 將 AWS Security Hub 與 Patch Manager 整合

建立修補程式政策

修補程式政策是您使用 中的Quick Setup工具 所設定的組態 AWS Systems Manager。與其他設定修補的方法相比，修補程式政策可提供更廣泛且更集中的修補操作控制。修補程式政策會定義自動修補節點和應用程式時要使用的排程和基準。

如需詳細資訊，請參閱下列主題：

- Quick Setup中的修補程式政策組態
- 使用修補程式政策設定組織中執行個體的Quick Setup修補

檢視修補程式儀表板摘要

中的儀表板索引標籤Patch Manager提供主控台中的摘要檢視，可讓您用來在合併檢視中監控修補操作。 Patch Manager 是 中的工具 AWS Systems Manager。您可以在 Dashboard (儀表板) 索引標籤上檢視下列內容：

- 有多少受管節點符合和不符合修補規則的快照。
- 受管節點之修補程式合規結果存留期的快照。
- 每個最常見的不合規原因中有多少個不合規受管節點的連結計數。

- 最近修補操作的連結清單。
- 已設定之週期性修補任務的連結清單。

檢視修補程式儀表板摘要

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Patch Manager。
3. 選擇 Dashboard (儀表板) 標籤。
4. 捲動至包含您要檢視之摘要資料的區段：
   - Amazon EC2 instance management (Amazon EC2 執行個體管理)
   - Compliance summary (合規摘要)
   - Noncompliance counts (不合規計數)
   - Compliance reports (合規報告)
   - Non-patch policy-based operations (非修補程式政策型操作)
   - Non-patch policy-based recurring tasks (非修補程式政策型週期性任務)

使用修補程式合規報告

使用下列主題中的資訊，協助您在 中的Patch Manager工具產生和使用修補程式合規報告 AWS Systems Manager。

無論您使用哪種方法或組態類型進行修補操作，下列主題中的資訊都適用：

- 在 Quick Setup 中設定的修補程式政策
- 在 Quick Setup 中設定的主機管理選項
- 用來執行修補程式 `Scan` 或 `Install` 任務的維護時段
- 隨需 Patch now (立即修補) 操作

> **Important**
>
> 如果您有多種類型的操作來掃描執行個體以檢查修補程式合規性，則請注意每次掃描都會覆寫先前掃描的修補程式合規資料。因此，最終可能會在修補程式合規資料中產生非預期的結果。如需詳細資訊，請參閱避免意外覆寫修補程式合規資料。

若要驗證使用哪個修補基準來產生最新的合規資訊，請導覽至 Patch Manager 中的合規報告索引標籤，找到您想要其相關資訊的受管節點的資料列，然後在使用的基準 ID 資料欄中選擇基準 ID。

主題

- 檢視修補程式合規結果
- 產生 .csv 修補程式合規報告
- 使用 Patch Manager 修復不符合標準的受管節點
- 避免意外覆寫修補程式合規資料

檢視修補程式合規結果

使用這些處理程序來檢視有關受管節點的修補程式合規資訊

此處理程序適用於使用 AWS-RunPatchBaseline 文件的修補程式操作。如需檢視使用 AWS-RunPatchBaselineAssociation 文件之修補程式操作的修補程式合規資訊，請參閱 識別不合規的受管節點。

> **Note**
>
> Quick Setup 和 Explorer使用的修補程式掃描操作AWS-RunPatchBaselineAssociation。Quick Setup和 Explorer 都是 中的工具 AWS Systems Manager。

識別特定 CVE 問題的修補程式解決方案 (Linux)

對於許多 Linux 作業系統，修補程式合規結果會指出哪些常見弱點和暴露 (CVE) 重要問題可以透過哪些修補程式來解決。此資訊可協助您判斷安裝遺漏或失效修補程式的迫切性。

下列作業系統類型的支援版本包含 CVE 詳細資訊：

- 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**
>
> 根據預設，CentOS 和 CentOS Stream 不會提供有關更新的 CVE 資訊。但是，您可以使用第三方儲存庫，例如 Fedora 發行的 Extra Packages for Enterprise Linux (EPEL) 儲存庫，以允許這項支援。如需相關資訊，請參閱 Fedora Wiki 上的 EPEL。
> 目前，僅針對狀態為 `Missing` 或 `Failed` 的修補程式報告 CVE ID 值。

您也可以將 CVE ID 新增至修補基準中已核准或已拒絕的修補程式清單，視情形和您的修補目標而定。

如需使用已核准和已拒絕修補程式清單的相關資訊，請參閱下列主題：

- 使用自訂修補基準
- 已核准與遭拒的修補程式清單的套件名稱格式
- 修補基準規則在 Linux 系統上的運作方式
- 如何安裝修補程式

> **Note**
>
> 在某些情況下，Microsoft 會針對未指定更新日期和時間的應用程式發行修補程式。在這些情況下，預設會提供 `01/01/1970` 的更新日期和時間。

檢視修補程式合規結果

使用以下程序在 AWS Systems Manager 主控台中檢視修補合規資料。

> **Note**
>
> 如需產生已下載至 Amazon Simple Storage Service (Amazon S3) 儲存貯體之修補程式合規報告的相關資訊，請參閱 產生 .csv 修補程式合規報告。

檢視修補程式合規結果

1. 執行下列其中一項操作。

   選項 1（建議）– 從 導覽Patch Manager，這是 中的工具 AWS Systems Manager：

   - 在導覽窗格中，選擇 Patch Manager。
   - 選擇 Compliance reporting (合規報告) 索引標籤。
   - 在節點修補詳細資訊中，請選擇要檢閱修補程式合規結果之受管節點的節點 ID。
   - 在詳細資訊區域的屬性清單中選擇修補程式。

   選項 2 - 從 Compliance 導覽，這是一個工具 AWS Systems Manager：

   - 在導覽窗格中，選擇 Compliance (合規)。
   - 對於 Compliance resources summary (合規資源摘要)，請在您要檢閱的修補程式資源類型欄中選擇一個數字，例如 Non-Compliant resources (不合規資源)。
   - 在下方的資源清單中，請選擇要檢閱修補程式合規結果的受管節點 ID。
   - 在詳細資訊區域的屬性清單中選擇修補程式。

   選項 3 – 從 導覽Fleet Manager，這是 中的工具 AWS Systems Manager。

   - 在導覽窗格中，選擇 Fleet Manager。
   - 在受管執行個體區域中，請選擇要檢閱修補程式合規結果的受管節點 ID。
   - 在詳細資訊區域的屬性清單中選擇修補程式。

2. (選用) 在搜尋方塊
   ()
   中，請選擇可用的篩選條件。

   例如，對於 Red Hat Enterprise Linux (RHEL)，請從下列選項中選擇：

   - 名稱
   - 分類
   - State
   - 嚴重性

針對 Windows Server，請選擇下列項目：

- KB
- 分類
- State
- 嚴重性

3. 為您選擇的篩選條件類型選擇其中一個可用的值。例如，如果您選擇了 State (狀態)，則現在選擇合規狀態，例如 InstalledPendingReboot、Failed 或 Missing。

> **Note**
>
> 目前，僅針對狀態為 `Missing` 或 `Failed` 的修補程式報告 CVE ID 值。

4. 根據受管節點的合規狀態，您可以選擇要採取的動作來修復任何不合規節點。

   例如，您可以選擇立即修補不合規受管節點。如需隨需修補受管節點的詳細資訊，請參閱 隨需修補受管節點。

   如需有關修補程式合規資料的詳細資訊，請參閱 修補程式合規狀態值。

產生 .csv 修補程式合規報告

您可以使用 AWS Systems Manager 主控台產生修補程式合規報告，這些報告會儲存為您選擇的 Amazon Simple Storage Service (Amazon S3) 儲存貯體的 .csv 檔案。您可以產生單一隨需報告，或指定自動產生報告的排程。

您可以為單一受管節點或所選 AWS 帳戶 和 中的所有受管節點產生報告 AWS 區域。對於單一節點，報告包含完整的詳細資訊，包括與不合規節點相關的修補程式 ID。針對所有受管節點的報告，只會提供摘要資訊和不合規節點的修補程式計數。

產生報告後，您可以使用 Amazon QuickSight 等工具來匯入和分析資料。QuickSight 是一種商業智慧 (BI) 服務，可用於探索和解譯互動式視覺化環境中的資訊。如需詳細資訊，請參閱《Amazon QuickSight 使用者指南》。

> **Note**
>
> 當您建立自訂修補基準時，您可以指定此修補基準所核准之修補程式的合規嚴重性等級，例如 `Critical` 或 `High`。如果任何核准之修補程式的修補程式狀態回報為 `Missing`，則修補基準的整體報告合規嚴重性是您指定的嚴重性等級。

您也可以指定 Amazon Simple Notification Service (Amazon SNS) 主題，在產生報告時傳送通知。

產生修補程式合規報告的服務角色

第一次產生報告時，Systems Manager 會建立名為 `AWS-SystemsManager-PatchSummaryExportRole` 的 Automation 擔任角色以用於匯出至 S3 的程序。

> **Note**
>
> 如果您要將合規資料匯出至加密的 S3 儲存貯體，您必須更新其相關聯的 AWS KMS 金鑰政策，以提供 所需的許可`AWS-SystemsManager-PatchSummaryExportRole`。例如，新增類似於 S3 儲存貯體 AWS KMS 政策的許可：
>
> ```
> {
>     "Effect": "Allow",
>     "Action": [
>         "kms:GenerateDataKey"
>     ],
>     "Resource": "role-arn"
> }
> ```
>
> 將 *`role-arn`* 替換為您帳戶中建立的資源的 Amazon Resource Name (ARN)，並遵循格式 `arn:aws:iam::111222333444:role/service-role/AWS-SystemsManager-PatchSummaryExportRole`。
> 如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的在 AWS KMS中使用金鑰政策。

當您第一次依排程產生報告時，Systems Manager 會建立另一個名為 `AWS-EventBridge-Start-SSMAutomationRole` 的服務角色，以及 `AWS-SystemsManager-PatchSummaryExportRole` 服務角色 (如果尚未建立) 以用於匯出程序。`AWS-EventBridge-Start-SSMAutomationRole` 可讓 Amazon EventBridge 使用 Runbook AWS-ExportPatchReportToS3 啟動自動化。

建議您不要嘗試修改這些政策和角色。這樣做可能會導致修補程式合規報告產生失敗。如需詳細資訊，請參閱對修補程式合規報告產生進行故障診斷。

主題

- 產生的修補程式合規報告中有哪些內容？
- 為單一受管節點產生修補程式合規報告
- 為所有受管節點產生修補程式合規報告
- 檢視修補程式合規報告歷史
- 檢視修補程式合規報告排程
- 對修補程式合規報告產生進行故障診斷

產生的修補程式合規報告中有哪些內容？

本主題提供產生並下載至指定 S3 儲存貯體之修補程式合規報告中所包含之內容類型的相關資訊。

單一受管節點的報告格式

針對單一受管節點產生的報告會提供摘要和詳細資訊。

下載範例報告 (單一節點)

單一受管節點的摘要資訊包括下列各項：

- 索引
- 執行個體 ID
- 執行個體名稱
- 執行個體 IP
- 平台名稱
- 平台版本
- SSM Agent 版本
- 修補基準
- 修補程式群組
- 合規狀態
- 合規嚴重性
- 不合規關鍵嚴重性修補程式計數

- 不合規高嚴重性修補程式計數
- 不合規中嚴重性修補程式計數
- 不合規低嚴重性修補程式計數
- 不合規資訊嚴重性修補程式計數
- 不合規未指定嚴重性修補程式計數

單一受管節點的詳細資訊包括下列各項：

- 索引
- 執行個體 ID
- 執行個體名稱
- 修補程式名稱
- KB ID /修補程式 ID
- 修補程式狀態
- 上次報告時間
- 合規層級
- 修補程式嚴重性
- 修補程式分類
- CVE ID
- 修補基準
- 日誌 URL
- 執行個體 IP
- 平台名稱
- 平台版本
- SSM Agent 版本

> **Note**
>
> 當您建立自訂修補基準時，您可以指定此修補基準所核准之修補程式的合規嚴重性等級，例如 `Critical` 或 `High`。如果任何核准之修補程式的修補程式狀態回報為 `Missing`，則修補基準的整體報告合規嚴重性是您指定的嚴重性等級。

所有受管節點的報告格式

針對所有受管節點產生的報告僅提供摘要資訊。

下載範例報告 (所有受管節點)

所有受管節點的摘要資訊包括下列各項：

- 索引
- 執行個體 ID
- 執行個體名稱
- 執行個體 IP
- 平台名稱
- 平台版本
- SSM Agent 版本
- 修補基準
- 修補程式群組
- 合規狀態
- 合規嚴重性
- 不合規關鍵嚴重性修補程式計數
- 不合規高嚴重性修補程式計數
- 不合規中嚴重性修補程式計數
- 不合規低嚴重性修補程式計數
- 不合規資訊嚴重性修補程式計數
- 不合規未指定嚴重性修補程式計數

為單一受管節點產生修補程式合規報告

請使用下列處理程序來產生 AWS 帳戶中單一受管節點的修補程式摘要報告。單一受管節點的報告提供有關每個不合規之修補程式的詳細資訊，包括修補程式名稱和 ID。

為單一受管節點產生修補程式合規報告

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。

2. 在導覽窗格中，選擇 Patch Manager。
3. 選擇 Compliance reporting (合規報告) 索引標籤。
4. 選擇您要產生報告之受管節點資料列的按鈕，然後選擇 View detail (檢視詳細資訊)。
5. 在 Patch summary (修補程式摘要) 區段中，選擇 Export to S3 (匯出至 S3)。
6. 對於 Report name (報告名稱)，輸入一個名稱以協助您稍後識別報告。
7. 對於 Reporting frequency (報告頻率)，選擇下列其中一項：
   - On demand (隨需) – 建立一次性報告。跳至步驟 9。
   - On a schedule (排程) – 指定自動產生報告的週期性排程。繼續步驟 8。
8. 對於 Schedule type (排程類型)，請指定 Rate 表達式 (例如每 3 天)，或提供 Cron 表達式來設定報告頻率。

   如需 cron 表達式的相關資訊，請參閱 參考：Systems Manager 的 Cron 和 Rate 運算式。
9. 對於 Bucket name (儲存貯體名稱)，請選擇您要存放 .csv 報告檔案的 S3 儲存貯體名稱。

   > **Important**
   >
   > 如果您在 2019 年 3 月 20 日之後 AWS 區域 啟動的 中工作，則必須選取相同區域中的 S3 儲存貯體。依預設，在該日期之後啟動的區域會處於關閉狀態。如需詳細資訊和這些區域的清單，請參閱《Amazon Web Services 一般參考》中的啟用區域一節。
10. (選用) 若要在報告產生時傳送通知，請展開 SNS topic (SNS 主題) 區段，然後從 SNS topic Amazon Resource Name (ARN) (SNS 主題 Amazon Resource Name (ARN)) 中選擇現有的 Amazon SNS 主題。
11. 選擇提交。

如需檢視產生報告之歷史記錄的相關資訊，請參閱 檢視修補程式合規報告歷史。

如需檢視已建立之報告排程詳細資訊的相關資訊，請參閱 檢視修補程式合規報告排程。

為所有受管節點產生修補程式合規報告

請使用下列處理程序來產生 AWS 帳戶中所有受管節點的修補程式摘要報告。所有受管節點的報告會指出哪些節點不合規，以及不合規修補程式的數目。它不會提供修補程式的名稱或其他識別符。如需這些其他詳細資訊，您可以為單一受管節點產生修補程式合規報告。如需相關資訊，請參閱本主題中稍早的為單一受管節點產生修補程式合規報告。

為所有受管節點產生修補程式合規報告

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Patch Manager。
3. 選擇 Compliance reporting (合規報告) 索引標籤。
4. 選擇 Export to S3 (匯出至 S3)。(請勿先選取節點 ID。)
5. 對於 Report name (報告名稱)，輸入一個名稱以協助您稍後識別報告。
6. 對於 Reporting frequency (報告頻率)，選擇下列其中一項：
   - On demand (隨需) – 建立一次性報告。跳至步驟 8。
   - On a schedule (排程) – 指定自動產生報告的週期性排程。繼續步驟 7。
7. 對於 Schedule type (排程類型)，請指定 Rate 表達式 (例如每 3 天)，或提供 Cron 表達式來設定報告頻率。

   如需 cron 表達式的相關資訊，請參閱 參考：Systems Manager 的 Cron 和 Rate 運算式。
8. 對於 Bucket name (儲存貯體名稱)，請選擇您要存放 .csv 報告檔案的 S3 儲存貯體名稱。

   > **Important**
   >
   > 如果您在 2019 年 3 月 20 日之後 AWS 區域 啟動的 中工作，則必須選取相同區域中的 S3 儲存貯體。依預設，在該日期之後啟動的區域會處於關閉狀態。如需詳細資訊和這些區域的清單，請參閱《Amazon Web Services 一般參考》中的啟用區域一節。
9. (選用) 若要在報告產生時傳送通知，請展開 SNS topic (SNS 主題) 區段，然後從 SNS topic Amazon Resource Name (ARN) (SNS 主題 Amazon Resource Name (ARN)) 中選擇現有的 Amazon SNS 主題。
10. 選擇提交。

如需檢視產生報告之歷史記錄的相關資訊，請參閱 檢視修補程式合規報告歷史。

如需檢視已建立之報告排程詳細資訊的相關資訊，請參閱 檢視修補程式合規報告排程。

檢視修補程式合規報告歷史

使用本主題中的資訊，協助您檢視 中產生之修補程式合規報告的詳細資訊 AWS 帳戶。

檢視修補程式合規報告歷史

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Patch Manager。
3. 選擇 Compliance reporting (合規報告) 索引標籤。
4. 選擇 View all S3 exports (檢視所有 S3 匯出)，然後選擇 Export history (匯出歷史記錄) 索引標籤。

檢視修補程式合規報告排程

使用本主題中的資訊來協助您檢視在 中建立的修補程式合規報告排程的詳細資訊 AWS 帳戶。

檢視修補程式合規報告歷史

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Patch Manager。
3. 選擇 Compliance reporting (合規報告) 索引標籤。
4. 選擇 View all S3 exports (檢視所有 S3 匯出)，然後選擇 Report schedule rules (報告排程規則) 索引標籤。

對修補程式合規報告產生進行故障診斷

使用下列資訊，協助您針對在 中產生修補程式合規報告的問題進行疑難排解Patch Manager，這是 中的工具 AWS Systems Manager。

主題

- 報告 AWS-SystemsManager-PatchManagerExportRolePolicy 政策已損毀的訊息
- 刪除修補程式合規政策或角色後，無法成功產生排定的報告

報告 **AWS-SystemsManager-PatchManagerExportRolePolicy** 政策已損毀的訊息

問題：您會收到類似下列的錯誤訊息，指出 AWS-SystemsManager-PatchManagerExportRolePolicy 已損毀：

```
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.
```

- 解決方案：使用 Patch Manager主控台或 在產生新的修補程式合規報告之前 AWS CLI 刪除受影響的角色和政策。

  使用主控台刪除損毀的政策

  1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
  2. 執行以下任意一項：

     隨需報告 – 如果在產生一次性隨需報告時發生問題，則請在左側導覽中選擇 Policies (政策)，搜尋 AWS-SystemsManager-PatchManagerExportRolePolicy，然後刪除政策。接下來，選擇 Roles (角色)，搜尋 AWS-SystemsManager-PatchSummaryExportRole，然後刪除該角色。

     排定的報告 – 如果問題在依排程產生報告時發生，則請在左側導覽中選擇政策，對於 AWS-EventBridge-Start-SSMAutomationRolePolicy 和 AWS-SystemsManager-PatchManagerExportRolePolicy，一次搜尋一個，然後刪除每個政策。接下來，選擇 Roles (角色)，一次搜尋一個 AWS-EventBridge-Start-SSMAutomationRole 和 AWS-SystemsManager-PatchSummaryExportRole，然後刪除每個角色。

  使用 刪除損毀的政策 AWS CLI

  使用您的帳戶 ID 取代*######*。

  - 如果在產生一次性隨需報告時發生問題，請執行下列命令：

    ```
    aws iam delete-policy --policy-arn arn:aws:iam::account-id:policy/AWS-
    SystemsManager-PatchManagerExportRolePolicy
    ```

    ```
    aws iam delete-role --role-name AWS-SystemsManager-PatchSummaryExportRole
    ```

    如果在產生排程報告時發生問題，請執行下列命令：

    ```
    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
```

在完成上述任一程序後，依照步驟產生或排程新的修補程式合規報告。

刪除修補程式合規政策或角色後，無法成功產生排定的報告

問題：第一次產生報告時，Systems Manager 會建立服務角色和政策以用於匯出程序 (`AWS-SystemsManager-PatchSummaryExportRole` 和 `AWS-SystemsManager-PatchManagerExportRolePolicy`)。當您第一次依排程產生報告時，Systems Manager 會建立另一個服務角色和政策 (`AWS-EventBridge-Start-SSMAutomationRole` 和 `AWS-EventBridge-Start-SSMAutomationRolePolicy`)。這些可讓 Amazon EventBridge 使用 Runbook AWS-ExportPatchReportToS3 啟動自動化。

如果您刪除這些政策或角色中的任何一個，則排程與指定 S3 儲存貯體和 Amazon SNS 主題之間的連線可能會遺失。

- 解決方案：若要解決這個問題，建議刪除先前的排程，並建立新的排程，以取代發生問題的排程。

使用 Patch Manager 修復不符合標準的受管節點

本節中的主題提供如何識別修補程式不合規的受管節點，以及如何讓節點合規的概觀。

主題

- 識別不合規的受管節點
- 修補程式合規狀態值
- 修補不相容的受管節點

識別不合規的受管節點

系統會在兩個 AWS Systems Manager 文件 (SSM 文件) 中的任何一個文件執行時，識別不合規受管節點。這些 SSM 文件會參考 中的工具 Patch Manager中每個受管節點的適當修補程式基準 AWS Systems Manager。然後，他們會評估受管節點的修補程式狀態，然後將合規結果提供給您。

有兩個 SSM 文件可用來識別或更新不合規受管節點：`AWS-RunPatchBaseline` 和 `AWS-RunPatchBaselineAssociation`。每個都會由不同的程序使用，而其合規結果則可透過不同的通道取得。下表概述了這些文件之間的差異。

> **Note**
>
> 來自 的修補程式合規資料Patch Manager可以傳送到 AWS Security Hub。Security Hub 可為您提供高優先級安全性警示和合規性狀態的全方位檢視。它還會監控您的機群的修補狀態。如需詳細資訊，請參閱將 AWS Security Hub 與 Patch Manager 整合。

	AWS-RunPatchBaseline	**AWS-RunPatchBaselineAssociation**
使用文件的程序	隨需修補程式 – 您可以使用 Patch now (立即修補) 選項隨需掃描或修補受管節點。如需相關資訊，請參閱 隨需修補受管節點。 Systems Manager Quick Setup修補程式政策 – 您可以在 中建立修補組態Quick Setup，也就是 中的工具 AWS Systems Manager，可以掃描整個組織、組織單位子集或單一 的個別排程，或安裝缺少的修補程式 AWS 帳戶。如需相關資訊，請參閱 使用修補程式政策設定組織中執行個體的Quick Setup修補。	Systems Manager Quick Setup 主機管理 – 您可以在 Quick Setup 中啟用主機管理組態選項，每天掃描受管執行個體，檢查修補程式是否合規。如需相關資訊，請參閱 使用 Quick Setup 設定 Amazon EC2 主機管理。 Systems Manager Explorer – 當您允許 中的Explorer工具時 AWS Systems Manager，它會定期掃描您的受管執行個體是否有修補程式合規，並在Explorer儀表板中報告結果。

	AWS-RunPatchBaseline	AWS-RunPatchBaselineAssociation
	執行命令 – 您可以在 中的 操作AWS-RunPatchBaseline 中手動執行 Run Command，這是 中的工具 AWS Systems Manager。如需相關資訊，請參閱 從主控台執行命令。 Maintenance window (維護時段) – 您可以在 Run Command 任務類型中建立使用 SSM 文件 AWS-RunPatchBaseline 的維護時段。如需相關資訊，請參閱 教學課程：使用主控台建立修補維護時段。	
修補程式掃描結果資料的格式	AWS-RunPatchBaseline 執行後， Patch Manager 會將AWS:PatchSummary 物件傳送至 中的工具 Inventory AWS Systems Manager。	AWS-RunPatchBaselineAssociation 執行後，Patch Manager 會傳送 AWS:ComplianceItem 物件至 Systems Manager 庫存。

	AWS-RunPatchBaseline	AWS-RunPatchBaselineAssociation
在主控台檢視修補的合規報告	您可以在 Systems Manager 組態合規和 使用受管節點 中檢視使用 AWS-RunPatchBaseline 之程序的修補程式合規資訊。如需詳細資訊，請參閱檢視修補程式合規結果。	如果使用Quick Setup掃描受管執行個體的修補程式是否合規，則您可以在 Systems Manager Fleet Manager 中查看合規報告。在 Fleet Manager 主控台中，選擇受管節點的節點 ID。在一般選單中，選擇組態合規。 如果使用 Explorer 掃描受管執行個體的修補程式是否合規，則您可以在 Explorer 和 Systems Manager OpsCenter 中查看合規報告。
AWS CLI 檢視修補程式合規結果的 命令	對於使用 的程序AWS-RunPatchBaseline ，您可以使用下列 AWS CLI 命令來檢視受管節點上修補程式的摘要資訊。 • describe-instance-patch-states • describe-instance-patch-states-for-patch-group • describe-patch-group-state	對於使用 的程序AWS-RunPatchBaselineAssociation ，您可以使用下列 AWS CLI 命令來檢視執行個體上修補程式的摘要資訊。 • list-compliance-items

	AWS-RunPatchBaseline	AWS-RunPatchBaselineAssociation
修補操作	對於使用 AWS-RunPatchBaseline 的程序，您可以指定是否讓操作僅執行 Scan 操作，或者 Scan and install 操作。 如果您的目標是識別不合規受管節點，而不是進行修復，請僅執行 Scan 操作。	使用 AWS-RunPatchBaselineAssociation 的 Quick Setup 和 Explorer 程序僅執行 Scan 操作。
詳細資訊	用於修補的 SSM 命令文件：AWS-RunPatchBaseline	用於修補的 SSM 命令文件：AWS-RunPatchBaselineAssociation

如需您可能會看到報告各種修補程式合規狀態的相關資訊，請參閱 修補程式合規狀態值

如需修復修補程式不合規之受管節點的相關資訊，請參閱 修補不相容的受管節點。

修補程式合規狀態值

受管節點修補程式的相關資訊包括每個個別修補程式的狀態報告或狀態。

> **Tip**
>
> 如果您想要將特定修補程式合規狀態指派給受管節點，您可以使用 put-compliance-items AWS Command Line Interface (AWS CLI) 命令或 PutComplianceItems API 操作。主控台中不支援指派合規狀態。

使用下表中的資訊，協助您識別受管節點可能不符合修補程式規範的原因。

Debian Server、Raspberry Pi OS 和 Ubuntu Server 修補程式的合規值

對於 Debian Server、Raspberry Pi OS 和 Ubuntu Server，不同合規狀態的套件分類規則如以下資料表所示：

> **Note**
>
> 當您評估 INSTALLED、 INSTALLED_OTHER和 MISSING 狀態值時，請記住下列事項：如果您未在建立或更新修補程式基準時選取包含非安全性更新核取方塊，則修補程式候選版本僅限於下列儲存庫中的修補程式：。
>
> - 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)
>
> 如果選取 Include nonsecurity updates (包含非安全性更新) 核取方塊，則也會考慮來自其他儲存庫的修補程式。

修補程式狀態	描述	合規狀態
INSTALLED	修補程式列在修補基準中，而且已安裝在受管節點上。可能已經由個人手動安裝，或在受管節點上執行 AWS-RunPatchBaseline 文件時由 Patch Manager 自動安裝。	合規
INSTALLED_OTHER	修補程式未包含在基準中，或未經基準核准，但已安裝在	合規

修補程式狀態	描述	合規狀態
	受管節點上。修補程式可能是手動安裝的，套件可能是另一個已核准修補程式的必要相依性，或者修補程式可能已包含在 InstallOverrideList 操作中。如果您不指定 `Block` 作為 Rejected patches (已拒絕的修補程式) 動作，則 `INSTALLED_OTHER` 修補程式也包含已安裝但拒絕的修補程式。	
`INSTALLED_PENDING_REBOOT`	`INSTALLED_PENDING_REBOOT` 可表示下列兩種情況之一： • Patch Manager `Install` 操作已將修補程式套用至受管節點，但節點尚未在套用修補程式後重新啟動。這通常表示 `NoReboot` 文件上次在受管節點上執行時，已為 `RebootOption` 參數選取了 `AWS-RunPatchBaseline` 選項。 如需詳細資訊，請參閱參數名稱：`RebootOption` 。 • 自上次重新啟動受管節點以來，已在 Patch Manager 外部安裝修補程式。 在這兩種情況下，這都不表示具有此狀態的修補程式需要重新啟動，只表示自安裝修補程式以來尚未重新啟動節點。	不合規

修補程式狀態	描述	合規狀態
INSTALLED_REJECTED	修補程式已安裝在受管節點，但是列於 Rejected patches (已拒絕修補程式) 清單。這通常表示修補程式在加到遭拒的修補程式清單中前就已安裝。	不合規
MISSING	透過基準篩選且尚未安裝的套件。	不合規
FAILED	在修補程式操作過程中安裝失敗的套件。	不合規

適用於其他作業系統的修補程式合規值

對於除 Debian Server、Raspberry Pi OS 和 Ubuntu Server 之外的所有作業系統，不同合規狀態的套件分類規則如以下資料表所示：

修補程式狀態	描述	合規值
INSTALLED	修補程式列在修補基準中，而且已安裝在受管節點上。可能已經由個人手動安裝，或在節點上執行 `AWS-RunPatchBaseline` 文件時由 Patch Manager 自動安裝。	合規
INSTALLED_OTHER 1	修補程式不在基準範圍中，但安裝了受管節點。這有兩個可能的原因： 1. 可能已手動安裝修補程式。 2. 僅限 Linux：可能已安裝該套件，作為不同的已核准修補程式的必要相依性。如果將 `Allow as dependenc`	合規

修補程式狀態	描述	合規值
	y 指定為遭拒的修補程式動作，則安裝為相依性的修補程式會取得報告狀態 INSTALLED_OTHER 。 **Note** Windows Server 不支援修補程式相依性的概念。如需有關 Patch Manager 如何處理 Windows Server 上遭拒的修補程式清單中之修補程式的資訊，請參閱自訂修補基準中遭拒的修補程式清單選項。	
INSTALLED_REJECTED	修補程式已安裝在受管節點，但是列於 Rejected patches (已拒絕修補程式) 清單。這通常表示修補程式在加到遭拒的修補程式清單中前就已安裝。	不合規

修補程式狀態	描述	合規值
INSTALLED_PENDING_REBOOT	INSTALLED_PENDING_REBOOT 可表示下列兩種情況之一： • Patch Manager Install 操作已將修補程式套用至受管節點，但節點尚未在套用修補程式後重新啟動。這通常表示 NoReboot 文件上次在受管節點上執行時，已為 RebootOption 參數選取了 AWS-RunPatchBaseline 選項。 如需詳細資訊，請參閱參數名稱：RebootOption 。 • 自上次重新啟動受管節點以來，已在 Patch Manager 外部安裝修補程式。 在這兩種情況下，這都不表示具有此狀態的修補程式需要重新啟動，只表示自安裝修補程式以來尚未重新啟動節點。	不合規
MISSING	修補程式在基準中已核准，但未安裝在受管節點。如果您設定 AWS-RunPatchBaseline 文件任務掃描 (而不是安裝)，系統會為在掃描期間找到，但尚未安裝的修補程式報告此狀態。	不合規

修補程式狀態	描述	合規值
FAILED	修補程式在基準中已核准，但無法安裝在執行個體。若要排除這種情況，請檢視命令輸出的資訊，或許能幫助您了解問題。	不合規
NOT_APPLICABLE 1	只有Windows Server作業系統才會報告此合規狀態。 修補程式在基準中已核准，但使用該修補程式的服務或功能尚未安裝在受管節點上。例如，若已在基準中核准，但尚未在受管節點上安裝 Web 服務，則 Internet Information Services (IIS) 等 Web 伺服器服務的修補程式會顯示 NOT_APPLICABLE 。如果修補程式已由後續更新所取代，也可能標示為 NOT_APPLICABLE 。這表示已安裝較新的更新，不再需要 NOT_APPLICABLE 更新。	不適用

修補程式狀態	描述	合規值
AVAILABLE_SECURITY_UPDATES	只有Windows Server作業系統才會報告此合規狀態。 未經修補程式基準核准的可用安全更新修補程式可以具有合規值 或 Compliant 或 Non-Compliant ，如自訂修補程式基準中所定義。 當您建立或更新修補程式基準時，您可以選擇要指派給可用但未核准的安全修補程式狀態，因為這些修補程式不符合修補程式基準中指定的安裝條件。例如，如果您已指定在安裝修補程式發行之前等待很長的時間，則可以略過您可能想要安裝的安全修補程式。如果在指定的等待期間發行修補程式的更新，安裝修補程式的等待期間會重新開始。如果等待期間過長，可以釋出多個版本的修補程式，但永遠不會安裝。 對於修補程式摘要計數，當修補程式報告為 時AvailableSecurityUpdate ，一律會包含在 中AvailableSecurityUpdateCount 。如果基準設定為將這些修補程式報告為 NonCompliant ，它也會包含在中SecurityNonCompliantCount 。如果基準設定	合規或不合規，取決於為可用安全性更新選取的選項。 **Note** 使用主控台建立或更新修補程式基準，您可以在可用安全性更新合規狀態欄位中指定此選項。使用 AWS CLI 執行 create-patch-baseline或 update-patch-baseline命令，您可以在 available-security-updates-compliance-status 參數中指定此選項。

修補程式狀態	描述	合規值
	為將這些修補程式報告為 Compliant ，則不會包含在中SecurityNonCompliantCount 。這些修補程式一律會以未指定的嚴重性進行報告，絕不會包含在中CriticalNonCompliantCount 。	

[1] 對於具有 INSTALLED_OTHER 與 NOT_APPLICABLE 狀態的修補程式，Patch Manager 根據 describe-instance-patches 命令忽略了查詢結果中的一些資料，例如 Classification 和 Severity 的值。這是為了協助防止超過庫存中個別節點的資料限制，這是 中的工具 AWS Systems Manager。若要檢視所有修補程式的詳細資訊，您可以使用 describe-available-patches 命令。

修補不相容的受管節點

您可以使用許多相同的 AWS Systems Manager 工具和程序來檢查受管節點的修補程式合規性，以讓節點符合目前適用於它們的修補程式規則。若要將受管節點納入修補程式合規， Patch Manager中的工具 AWS Systems Manager必須執行 Scan and install操作。(如果您的目標僅是識別不合規受管節點，而不是進行修復，請改為執行 Scan 操作。如需詳細資訊，請參閱 識別不合規的受管節點。)

使用 Systems Manager 安裝修補

您可以從數種工具中選擇以執行 Scan and install 操作：

- （建議）在 Systems Manager 中的Quick Setup工具 中設定修補程式政策，可讓您根據整個組織的排程、組織單位子集或單一 來安裝缺少的修補程式 AWS 帳戶。如需詳細資訊，請參閱使用修補程式政策設定組織中執行個體的Quick Setup修補。
- 在 Run Command 任務類型中建立使用 Systems Manager 文件 (SSM 文件) AWS-RunPatchBaseline 的維護時段。如需相關資訊，請參閱 教學課程：使用主控台建立修補維護時段。
- 手動執行 Run Command 操作中的 AWS-RunPatchBaseline。如需相關資訊，請參閱 從主控台執行命令。

- 使用 Patch now (立即修補) 選項隨需安裝修補程式。如需相關資訊，請參閱 隨需修補受管節點。

避免意外覆寫修補程式合規資料

如果您有多種類型的操作來掃描執行個體以檢查修補程式合規性，則每次掃描都會覆寫先前掃描的修補程式合規資料。因此，最終可能會在修補程式合規資料中產生非預期的結果。

例如，假設您建立的修補程式政策會在當地時間每天凌晨 2 點掃描修補程式合規性。該修補程式政策使用修補基準，該基準以嚴重性標記為 `Critical`、`Important` 和 `Moderate` 的修補程式為目標。此修補基準也會指定幾個特別拒絕的修補程式。

此外，假設您已經設定了維護時段，以便每天在當地時間凌晨 4 點掃描同一組受管節點，而您不會刪除或停用這些節點。該維護時段的任務會使用不同的修補基準，該基準僅針對嚴重性為 `Critical` 的修補程式，且不會排除任何特定修補程式。

當維護時段執行第二次掃描時，會刪除第一次掃描中的修補程式合規資料，並取代為第二次掃描中的修補程式合規性。

因此，強烈建議您只使用一種自動化方法在修補操作中進行掃描和安裝。如果您正在設定修補程式政策，則應刪除或停用其他掃描方法，以確保修補程式合規性。如需詳細資訊，請參閱下列主題：

- 移除維護時段中的修補程式操作 - 使用主控台更新或取消註冊維護時段任務
- 刪除 State Manager 關聯 – 刪除關聯。

若要停用主機管理組態中的每日修補程式合規掃描，請在 Quick Setup 中執行下列動作：

1. 在導覽窗格中，選擇 Quick Setup。
2. 選取要更新的主機管理組態。
3. 選擇 Actions, Edit configuration (動作、編輯組態)。
4. 清除 Scan instances for missing patches daily (每日掃描遺失修補程式的執行個體) 核取方塊。
5. 選擇更新。

> **Note**
>
> 使用 Patch now (立即修補) 選項掃描受管節點是否合規，會導致覆寫修補程式合規資料。

隨需修補受管節點

使用 中的立即修補選項Patch Manager， AWS Systems Manager即可從 Systems Manager 主控台執行隨需修補操作。這表示您不需要建立排程，即可更新受管節點的合規狀態或在不合規節點上安裝修補程式。您也不需要在 中的Maintenance Windows工具 Patch Manager和 之間切換 Systems Manager 主控台 AWS Systems Manager，以設定或修改排定的修補時段。

Patch now (立即修補) 在您必須儘快在受管節點上套用零時差更新或安裝其他重要修補程式時，特別實用。

> **Note**
>
> 一次單一 AWS 帳戶AWS 區域 配對支援隨需修補。其無法用於以修補程式政策為基礎的修補操作。建議您使用修補程式政策來維持所有受管節點的合規性。如需有關使用修補程式政策的詳細資訊，請參閱 Quick Setup中的修補程式政策組態。

主題

- 「立即修補」的運作方式
- 執行「立即修補」

「立即修補」的運作方式

若要執行 Patch now (立即修補)，您只需指定兩個必要設定：

- 是否只掃描缺少的修補程式，或掃描和安裝受管節點上的修補程式
- 要在哪些受管節點上執行操作

當 Patch now (立即修補) 操作執行時，它會決定要使用哪個修補基準，以與為其他修補操作所選取的相同方式使用。如果受管節點與修補程式群組相關聯，則會使用為該群組指定的修補基準。如果受管節點未與修補程式群組關聯，則操作會使用目前設定為受管節點之操作系統類型預設值的修補基準。這可以是預先定義基準，也可以是您已設定為預設值的自訂基準。如需修補基準選取項目的詳細資訊，請參閱 修補程式群組。

您可以為 Patch now (立即修補) 指定的選項包括在修補後選擇何時或是否重新啟動受管節點、指定 Amazon Simple Storage Service (Amazon S3) 儲存貯體來存放修補操作的日誌資料，以及在修補期間將 Systems Manager 文件 (SSM 文件) 作為生命週期掛鉤執行。

「立即修補」的並行和錯誤閾值

對於 Patch now (立即修補) 操作，並行和錯誤閾值選項由 Patch Manager 處理。您不需要指定要一次修補的受管節點數目，也不需要指定操作失敗之前允許的錯誤數目。當您進行隨需修補時，Patch Manager 會套用下表所述的並行和錯誤閾值設定。

> **Important**
>
> 以下閾值僅適用於 `Scan and install` 操作。對於 `Scan` 操作，Patch Manager 會嘗試並行掃描最多 1,000 個節點，並繼續掃描，直至遇到最多 1,000 個錯誤。

並行：安裝操作

Patch now (立即修補) 操作中受管節點的總數	一次掃描或修補的受管節點數目
低於 25 個	1
25-100	5%
101 到 1,000 個	8%
1,000 個以上	10%

錯誤閾值：安裝操作

Patch now (立即修補) 操作中受管節點的總數	操作失敗前允許的錯誤數目
低於 25 個	1
25-100	5
101 到 1,000 個	10
1,000 個以上	10

使用「立即修補」生命週期掛鉤

Patch now (立即修補) 為您提供在 `Install` 修補操作期間能夠將 SSM Command 文件作為生命週期掛鉤執行的功能。您可以將這些掛鉤用於任務，例如在修補之前關閉應用程式，或在修補後或重新開機後對應用程式執行運作狀態檢查。

如需有關生命週期關聯的詳細資訊，請參閱 [用於修補的 SSM 命令文件：AWS-RunPatchBaselineWithHooks](#)。

除了每個掛鉤的使用範例，下表還為三個 Patch now (立即修補) 重新開機選項中的每個選項列出可用的生命週期掛鉤。

生命週期掛鉤和使用範例

重新開機選項	掛鉤：安裝前	掛鉤：安裝後	掛鉤：結束時	掛鉤：排定的重新開機後
Reboot if needed (必要時重新開機)	開始修補之前，請先執行 SSM 文件。 使用範例：在修補程序開始之前，安全地關閉應用程式。	在修補操作結束時和受管節點重新開機之前，執行 SSM 文件。 使用範例：執行操作，例如在潛在重新開機前安裝第三方應用程式。	修補操作完成並且執行個體重新開機後，執行 SSM 文件。 使用範例：確定應用程式在修補後如預期般執行。	不適用
Do not reboot my instances (請勿重新開機執行個體)	同上。	在修補操作結束時，執行 SSM 文件。 使用範例：確定應用程式在修補後如預期般執行。	不適用	不適用
Schedule a reboot time (排程重新開機時間)	同上。	與 Do not reboot my instances (請	不適用	在排定的重新開機完成後，立

重新開機選項	掛鉤：安裝前	掛鉤：安裝後	掛鉤：結束時	掛鉤：排定的重新開機後
		勿重新開機執行個體) 相同。		即執行 SSM 文件。 使用範例：確定應用程式在重新開機後如預期般執行。

執行「立即修補」

使用下列處理程序，隨需修補受管節點。

執行「立即修補」

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Patch Manager。
3. 選擇 Patch now (立即修補)。
4. 對於 Patching operation (修補操作)，選擇下列其中一項：
   - Scan (掃描)：Patch Manager 會發現受管節點中缺少哪些修補程式，但不會進行安裝。您可以在 Compliance (合規) 儀表板或其他用於檢視修補程式合規的工具中檢視結果。
   - Scan and install (掃描和安裝)：Patch Manager 會發現受管節點中缺少哪些修補程式，但不會進行安裝。
5. 只有在上一個步驟中選擇 Scan and install (掃描和安裝) 時，才使用該步驟。針對 Regions option (區域選項)，選擇以下其中一個選項：
   - Reboot if needed (依需要重新啟動)：安裝完成後，Patch Manager 僅在需要完成修補程式安裝時，才會重新啟動受管節點。
   - Don't reboot my instances (請勿重新啟動執行個體)：安裝完成後，Patch Manager 不會重新啟動受管節點。您可以在選擇或管理 Patch Manager 以外的重新開機時，自動重新開機節點。
   - Schedule a reboot time (排程重新開機時間)：指定 Patch Manager 的日期、時間和 UTC 時區重新開機您的受管節點。在您執行 Patch now (立即修補) 操作時，排定的重新開機會列為具有 `AWS-PatchRebootAssociation` 名稱之 State Manager 中的關聯。

6. 在 Instances to patch (要修補的執行個體)，選擇以下其中一項：

   - 修補所有執行個體： Patch Manager 會在目前 中 AWS 帳戶 的所有受管節點上執行指定的操作 AWS 區域。
   - Patch only the target instances I specify (只修補我指定的目標執行個體)：您可以在下一個步驟中指定要鎖定的受管節點。

7. 僅當您在上一個步驟中選擇 Patch only the target instances I specify (只修補我指定的目標執行個體) 時，使用該步驟。在 Target selection (目標選擇範圍) 區段中，指定標籤、手動選取節點或指定資源群組，以識別您要執行這項操作的節點。

   > **Note**
   >
   > 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。
   > 如果您選擇以資源群組為目標，請注意，以 AWS CloudFormation 堆疊為基礎的資源群組仍必須加上預設標籤`aws:cloudformation:stack-id`。如果已經移除，Patch Manager 可能無法判斷哪些受管節點屬於資源群組。

8. (選用) 對於 Patching log storage (修補日誌儲存)，如果您要從此修補操作建立並儲存日誌，請選取 S3 儲存貯體來存放日誌。

   > **Note**
   >
   > 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給執行個體之執行個體設定檔 (適用於 EC2 執行個體) 或 IAM 服務角色 (啟用混合模式的機器) 的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或在混合多雲端環境中建立 Systems Manager 所需的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確定與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

9. (選用) 如果您要在修補操作的特定時間點執行 SSM 文件作為生命週期掛鉤，則請執行下列動作：

   - 選擇 Use lifecycle hooks (使用生命週期掛鉤)。
   - 針對每個可用的掛鉤，選取要在操作之指定時間點執行的 SSM 文件：
     - 安裝前
     - 安裝後
     - 結束時

- 排定的重新開機後

> **Note**
>
> 預設文件 AWS-Noop 不會執行任何操作。

10. 選擇 Patch now (立即修補)。

    Association execution summary (關聯執行摘要) 頁面隨即開啟。(Patch 現在會在 中使用關聯 State Manager，這是 中的工具 AWS Systems Manager，用於其操作。) 在 Operation summary (操作摘要) 區域中，您可以監控指定受管節點上的掃描或修補狀態。

使用修補基準

中的修補程式基準Patch Manager，是 中的工具 AWS Systems Manager，定義哪些修補程式已核准安裝在受管節點上。您可以逐一指定核准或拒絕修補程式。您也可以建立自動核准規則，以指定應自動核准某些類型的更新 (例如，關鍵的更新)。拒絕清單會覆寫規則與核准清單。若要使用核准修補程式清單安裝特定套件，首先要移除所有自動核准規則。如果您明確將修補程式識別為拒絕，將不會核准或安裝該修補程式，即使它符合自動核准規則中的所有條件。此外，即使修補程式已核准用於受管節點，但只有在修補程式適用於節點上的軟體時，修補程式才會安裝於受管節點。

主題

- 查看 AWS 預先定義的修補基準
- 使用自訂修補基準
- 將現有的修補基準設為預設值 (主控台)

詳細資訊

- 修補基準

查看 AWS 預先定義的修補基準

Patch Manager中的工具 AWS Systems Manager包含 支援的每個作業系統的預先定義修補程式基準 Patch Manager。您可以使用這些修補基準 (您無法自訂它們)，或建立自己的修補基準。下列程序說明如何查看預先定義的修補基準是否符合您的需求。若要進一步了解修補基準，請參閱預先定義和自訂的修補基準。

檢視 AWS 預先定義的修補程式基準

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Patch Manager。
3. 在修補基準清單中，請選擇其中一個預先定義的修補基準的基準 ID。

   -或-

   如果您是第一次在目前的 AWS 區域存取 Patch Manager，請選擇從概觀開始，然後選擇修補基準索引標籤，再選擇其中一個預先定義的修補基準的基準 ID。

   > **Note**
   >
   > 對於 Windows Server，則會提供三個預先定義的修補基準。修補基準 `AWS-DefaultPatchBaseline` 和 `AWS-WindowsPredefinedPatchBaseline-OS` 僅支援 Windows 作業系統本身的作業系統更新。`AWS-DefaultPatchBaseline` 會用作 Windows Server 受管節點的預設修補基準，除非您指定了不同的修補基準。這兩個修補基準中的組態設定是相同的。兩者中較新的 `AWS-WindowsPredefinedPatchBaseline-OS` 是為了區分它與 Windows Server 第三方預先定義修補基準而建立的。修補基準 `AWS-WindowsPredefinedPatchBaseline-OS-Applications` 可用來將修補程式套用至 Windows Server 作業系統和 Microsoft 發行的支援應用程式。
   > 如需詳細資訊，請參閱將現有的修補基準設為預設值 (主控台)。

4. 在核准規則區段中，檢閱修補基準組態。
5. 如果該設定適用於您的受管節點，您可以請直接跳到程序 建立和管理修補程式群組。

   -或-

   若要建立自己的預設修補基準，請繼續前往主題 使用自訂修補基準。

使用自訂修補基準

Patch Manager中的工具 AWS Systems Manager包含 支援的每個作業系統的預先定義修補程式基準 Patch Manager。您可以使用這些修補基準 (您無法自訂它們)，或建立自己的修補基準。

下列程序說明如何建立、更新並刪除自己的自訂修補基準。若要進一步了解修補基準，請參閱預先定義和自訂的修補基準。

主題

- 建立適用於 Linux 的自訂修補基準
- 建立 macOS 的自訂修補基準
- 建立 Windows Server 的自訂修補基準
- 更新或刪除自訂修補基準

建立適用於 Linux 的自訂修補基準

使用下列程序為 中的Patch Manager工具 Linux 受管節點建立自訂修補程式基準 AWS Systems Manager。

如需為 macOS 受管節點建立修補基準的資訊，請參閱 建立 macOS 的自訂修補基準。如需為 Windows 受管節點建立修補基準的資訊，請參閱 建立 Windows Server 的自訂修補基準。

為 Linux 受管節點建立自訂修補基準

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Patch Manager。
3. 選擇修補基準索引標籤，然後選擇建立修補基準。

   -或-

   如果您是第一次在目前的 AWS 區域存取 Patch Manager，請選擇從概觀開始，然後選擇修補基準索引標籤，接著再選擇建立修補基準。
4. 在 Name (名稱) 中，為新的修補基準輸入一個名稱，例如 `MyRHELPatchBaseline`。
5. (選用) 在 Description (描述) 中，輸入此修補基準的描述。
6. 在 Operating system (作業系統) 選擇作業系統，例如 `Red Hat Enterprise Linux`。
7. 如果要在建立所選作業系統時，立即開始將此修補基準做為所選作業系統的預設設定，請核取 Set this patch baseline as the default patch baseline for ***`operating system name`*** instances (將此修補基準設定為「作業系統名稱」執行個體的預設修補基準) 旁的方塊。

   > **Note**
   >
   > 唯有當您在 2022 年 12 月 22 日修補程式政策發行前第一次存取 Patch Manager，才能使用此選項。

如需有關設定現有修補基準為預設的更多資訊，請參閱 [將現有的修補基準設為預設值 (主控台)](#)。

8. 在 Approval rules for operating-system (作業系統核准規則) 部分中，使用欄位來建立一或多個自動核准規則。

   - 產品：此核准規則適用的作業系統版本，例如 `RedhatEnterpriseLinux7.4`。預設的選取為 `All`。
   - Classification (分類)：此核准規則適用的修補程式類型，例如 `Security` 或 `Enhancement`。預設的選取為 `All`。

     Tip

     您可以設定修補基準，以控制是否安裝 Linux 的次要版本升級，例如 RHEL 7.8。Patch Manager 可以自動安裝次要版本升級，前提是適當的存放庫中有可用的更新。
     對於 Linux 作業系統，次要版本升級分類並不一致。即使在相同的核心版本中，它們可能被歸類為錯誤修正或安全更新，或者未歸類。以下是控制是否要安裝修補基準的幾個選項。

     - 選項 1：確保在次要版本升級可用時進行安裝的最廣泛核准規則是將 Classification (分類) 指定為 `All` (*)，然後選擇 Include nonsecurity updates (包含非安全更新) 選項。
     - 選項 2：若要確保安裝作業系統版本的修補程式，您可以使用萬用字元 (*)，在基準的 Patch exceptions (修補程式例外狀況) 區段中指定其核心格式。例如，RHEL 7.* 的核心格式為 `kernel-3.10.0-*.el7.x86_64`。

       在修補基準的 Approved patches (已核准的修補程式) 清單中輸入 `kernel-3.10.0-*.el7.x86_64`，確保所有修補程式 (包括次要版本升級) 已套用至 RHEL 7.* 受管節點。(如果您知道次要版本修補程式的確切套件名稱，可以改輸入該名稱。)
     - 選項 3：您可以對哪些修補程式套用至受管節點 (包括次要版本升級) 進行最大控制，方法是使用 `AWS-RunPatchBaseline` 文件中的 InstallOverrideList 參數。如需詳細資訊，請參閱[用於修補的 SSM 命令文件：AWS-RunPatchBaseline](#)。

   - 核准規則 (嚴重性)：此規則適用的修補程式嚴重性值，例如 `Critical`。預設的選取為 `All`。
   - Auto-approval (自動核准)：選取修補程式以進行自動核准的方法。

> **Note**
>
> 因為無法可靠地判斷 Ubuntu Server 更新套件的發行日期，此作業系統不支援自動核准選項。

- Approve patches after a specified number of days (指定天數之後核准修補程式)：修補程式發行或最後更新之後，Patch Manager 自動核准修補程式之前的等待天數。您可以輸入零 (0) 到 360 的任何整數。在大部分情形下，建議等候不要超過 100 天。
- Approve patches released up to a specific date (核准特定日期前發布的修補程式)：Patch Manager 會自動套用在此發行或更新日期當天或之前發行的所有修補程式。例如，如果您指定 2023 年 7 月 7 日，則不會自動安裝在 2023 年 7 月 8 日或之後發行或最後更新的修補程式。

- (選用) 合規報告：您要指派給基準所核准之修補程式的嚴重性等級，例如 `Critical` 或 `High`。

> **Note**
>
> 如果您指定合規報告等級，且任何核准之修補程式的修補程式狀態回報為 `Missing`，則修補基準的整體報告合規嚴重性是您指定的嚴重性等級。

- Include non-security updates (包含非安全性更新)：除了安裝安全性相關修補程式之外，選取此核取方塊可安裝來源儲存庫中可用的非安全性修補程式。

> **Note**
>
> 如果是 SUSE Linux Enterprise Server，(SLES)則無需選取此核取方塊，因為安全性與非安全性問題的修補程式預設都會安裝於 SLES 受管節點。如需詳細資訊，請參閱如何選取安全性修補程式中的有關 SLES 的內容。

如需有關在自訂修補基準中使用核准規則的詳細資訊，請參閱自訂基準。

9. 如果您要明確核准符合核准規則之修補程式以外的任何修補程式，請在 Patch exceptions (修補程式例外狀況) 部分執行下列動作：

   - 在 Approved patches (核准的修補程式)，輸入您要核准之修補程式的逗號分隔清單。

如需已核准修補程式和已拒絕修補程式清單之可接受格式的相關資訊，請參閱 已核准與遭拒的修補程式清單的套件名稱格式。

- (選用) 在 Approved patches compliance level (核准的修補程式合規層級)中，將合規層級指派至清單中的修補程式。
- 如果您指定的任何核准修補程式都與安全性無關，請選取包含非安全性更新核取方塊，以便在您的 Linux 作業系統中也安裝這些修補程式。

10. 如果您要明確拒絕任何符合核准規則之修補程式，請在 Patch exceptions (修補程式例外狀況) 部分執行下列動作：

    - 在 Rejected patches (拒絕的修補程式) 中，輸入您要拒絕之修補程式的逗號分隔清單。

      如需已核准修補程式和已拒絕修補程式清單之可接受格式的相關資訊，請參閱 已核准與遭拒的修補程式清單的套件名稱格式。

    - 在 Rejected patches action (已拒絕修補程式動作) 中，選擇要讓 Patch Manager在 Rejected patches (已拒絕修補程式) 清單所包含之修補程式上執行的動作。
      - Allow as dependency (當做相依性允許)：只有當套件是其他套件的相依性時，才會安裝 Rejected patches (已拒絕修補程式) 清單中的套件。這被視為符合修補基準，其狀態回報為 InstalledOther。若未指定選項，此為預設動作。
      - 封鎖：遭拒的修補程式清單中的套件，以及將這些套件作為相依性而包含在內的套件，Patch Manager 在任何情況下都無法安裝。如果在遭拒的修補程式清單中新增套件之前即安裝該套件，或在之後於 Patch Manager 外部安裝該套件，則會被視為不符合修補基準，並且會將其狀態回報為 InstalledRejected。

    > **Note**
    >
    > Patch Manager 會以遞迴方式搜尋修補程式相依性。

11. (選用) 如果您要為不同版本的作業系統指定替代的修補程式儲存庫，例如 AmazonLinux2016.03 與 AmazonLinux2017.09，請為 Patch sources (修補程式來源) 部分中的每個產品執行以下動作：

    - 在 Name (名稱) 中輸入一個名稱以協助您識別來源組態。
    - 在 Product (產品) 中，請選取修補程式來源儲存庫適用的作業系統版本，例如 `RedhatEnterpriseLinux7.4`。
    - 在 Configuration (組態) 中輸入要以下列格式使用的 yum 儲存庫組態的值。

```
[main]
name=MyCustomRepository
baseurl=https://my-custom-repository
enabled=1
```

> **Tip**
>
> 如需有關 yum 儲存庫組態可用選項的詳細資訊，請參閱 dnf.conf(5)。

選擇 Add another source (新增其他來源)，為每個額外的作業系統版本指定來源儲存庫，最多 20 個。

如需有關替代來源修補程式儲存庫的詳細資訊，請參閱如何指定替代修補程式來源儲存庫 (Linux)。

12. (選用) 對於 Manage tags (管理標籤)，請套用一或多個索引鍵名稱/值對至修補基準。

    標籤是您指派給資源的選用性中繼資料。標籤允許您以不同的方式 (例如用途、擁有者或環境) 將資源分類。例如，您可能希望標記修補基準，以識別其指定的修補程式的嚴重性等級、其適用的作業系統系列，以及環境類型。在這種情況下，您可以指定類似以下索引鍵名稱/值對的標籤：

    - `Key=PatchSeverity,Value=Critical`
    - `Key=OS,Value=RHEL`
    - `Key=Environment,Value=Production`

13. 選擇 Create patch baseline (建立修補基準)。

建立 macOS 的自訂修補基準

使用下列程序為 中的Patch Manager工具 中的macOS受管節點建立自訂修補程式基準 AWS Systems Manager。

如需為 Windows Server 受管節點建立修補基準的資訊，請參閱 建立 Windows Server 的自訂修補基準。如需為 Linux 受管節點建立修補基準的資訊，請參閱 建立適用於 Linux 的自訂修補基準。

> **Note**
>
> macOS 不支援全部 AWS 區域。如需有關 macOS Amazon EC2 支援的詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Amazon EC2 Mac instances。

為 macOS 受管節點建立自訂修補基準

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Patch Manager。
3. 選擇修補基準索引標籤，然後選擇建立修補基準。

   -或-

   如果您是第一次在目前的 AWS 區域存取 Patch Manager，請選擇從概觀開始，然後選擇修補基準索引標籤，接著再選擇建立修補基準。
4. 在 Name (名稱) 中，為新的修補基準輸入一個名稱，例如 `MymacOSPatchBaseline`。
5. (選用) 在 Description (描述) 中，輸入此修補基準的描述。
6. 在 Operating system (作業系統) 中，選擇 macOS。
7. 如果要在建立 macOS 時，立即開始將此修補基準做為所選作業系統的預設設定，請核取 Set this patch baseline as the default patch baseline for macOS instances (將此修補基準設定為 MacOS 執行個體的預設修補基準) 旁的方塊。

   > **Note**
   >
   > 唯有當您在 2022 年 12 月 22 日修補程式政策發行前第一次存取 Patch Manager，才能使用此選項。
   > 如需有關設定現有修補基準為預設的更多資訊，請參閱 將現有的修補基準設為預設值 (主控台)。
8. 在 Approval rules for operating-system (作業系統核准規則) 部分中，使用欄位來建立一或多個自動核准規則。

   - 產品：此核准規則適用的作業系統版本，例如 `Mojave10.14.1` 或 `Catalina10.15.1`。預設的選取為 `All`。

> **Note**
>
> Homebrew 開放原始碼軟體套件管理系統已停止支援 macOS 10.14.x (Mojave) 和 10.15.x (Catalina)。因此，這些版本Patch Manager不支援使用 的修補操作。

- Classification (分類)：您想要在修補程式期間套用套件的套件管理工具。您可以選擇下列項目：
  - softwareupdate
  - Installer (安裝程式)
  - brew
  - brew cask

  預設的選取為 `All`。
- (選用) 合規報告：您要指派給基準所核准之修補程式的嚴重性等級，例如 `Critical` 或 `High`。

> **Note**
>
> 如果您指定合規報告等級，且任何核准之修補程式的修補程式狀態回報為 `Missing`，則修補基準的整體報告合規嚴重性是您指定的嚴重性等級。

如需有關在自訂修補基準中使用核准規則的詳細資訊，請參閱自訂基準。

9. 如果您要明確核准符合核准規則之修補程式以外的任何修補程式，請在 Patch exceptions (修補程式例外狀況) 部分執行下列動作：
   - 在 Approved patches (核准的修補程式)，輸入您要核准之修補程式的逗號分隔清單。

     如需已核准修補程式和已拒絕修補程式清單之可接受格式的相關資訊，請參閱 已核准與遭拒的修補程式清單的套件名稱格式。
   - (選用) 在 Approved patches compliance level (核准的修補程式合規層級)中，將合規層級指派至清單中的修補程式。
10. 如果您要明確拒絕任何符合核准規則之修補程式，請在 Patch exceptions (修補程式例外狀況) 部分執行下列動作：
    - 在 Rejected patches (拒絕的修補程式) 中，輸入您要拒絕之修補程式的逗號分隔清單。

如需已核准修補程式和已拒絕修補程式清單之可接受格式的相關資訊，請參閱 已核准與遭拒的修補程式清單的套件名稱格式。

- 在 Rejected patches action (已拒絕修補程式動作) 中，選擇要讓 Patch Manager在 Rejected patches (已拒絕修補程式) 清單所包含之修補程式上執行的動作。
  - Allow as dependency (當做相依性允許)：只有當套件是其他套件的相依性時，才會安裝 Rejected patches (已拒絕修補程式) 清單中的套件。這被視為符合修補基準，其狀態回報為 InstalledOther。若未指定選項，此為預設動作。
  - 封鎖：遭拒的修補程式清單中的套件，以及將這些套件作為相依性而包含在內的套件，Patch Manager 在任何情況下都無法安裝。如果在遭拒的修補程式清單中新增套件之前即安裝該套件，或在之後於 Patch Manager 外部安裝該套件，則會被視為不符合修補基準，並且會將其狀態回報為 InstalledRejected。

11. (選用) 對於 Manage tags (管理標籤)，請套用一或多個索引鍵名稱/值對至修補基準。

    標籤是您指派給資源的選用性中繼資料。標籤允許您以不同的方式 (例如用途、擁有者或環境) 將資源分類。例如，您可能希望標記修補基準，以識別其指定的修補程式的嚴重性等級、其適用的套件管理工具，以及環境類型。在這種情況下，您可以指定類似以下索引鍵名稱/值對的標籤：

    - `Key=PatchSeverity,Value=Critical`
    - `Key=PackageManager,Value=softwareupdate`
    - `Key=Environment,Value=Production`

12. 選擇 Create patch baseline (建立修補基準)。

建立 Windows Server 的自訂修補基準

使用下列程序為 中的Patch Manager工具 Windows 受管節點建立自訂修補基準 AWS Systems Manager。

如需為 Linux 受管節點建立修補基準的資訊，請參閱 建立適用於 Linux 的自訂修補基準。如需有關建立 macOS 受管節點的修補基準的詳細資訊，請參閱建立 macOS 的自訂修補基準。

如需建立僅限於安裝 Windows Service Pack 之修補基準的範例，請參閱教學課程：使用主控台建立安裝 Windows Service Pack 的修補基準。

建立自訂修補基準 (Windows)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。

2. 在導覽窗格中，選擇 Patch Manager。
3. 選擇修補基準索引標籤，然後選擇建立修補基準。

   -或-

   如果您是第一次在目前的 AWS 區域存取 Patch Manager，請選擇從概觀開始，然後選擇修補基準索引標籤，接著再選擇建立修補基準。
4. 在 Name (名稱) 中，為新的修補基準輸入一個名稱，例如 `MyWindowsPatchBaseline`。
5. (選用) 在 Description (描述) 中，輸入此修補基準的描述。
6. 在 Operating system (作業系統) 中，選擇 `Windows`。
7. 針對可用安全更新合規狀態，選擇您要指派給可用但未核准的安全修補程式狀態，因為這些修補程式不符合修補程式基準、不合規或合規中指定的安裝條件。

   範例案例：如果您已指定在安裝修補程式發行後等待很長的時間，則可以略過您可能想要安裝的安全修補程式。如果在指定的等待期間發行修補程式的更新，安裝修補程式的等待期間會重新開始。如果等待期間過長，可以釋出多個版本的修補程式，但永遠不會安裝。
8. 如果要在建立 Windows 時立即開始將此修補基準做為 Windows 的預設設定，請選擇 Set this patch baseline as the default patch baseline for Windows Server instances (將此修補基準設為 Windows Server 執行個體的預設修補基準)。

   > **Note**
   >
   > 唯有當您在 2022 年 12 月 22 日修補程式政策發行前第一次存取 Patch Manager，才能使用此選項。
   > 如需有關設定現有修補基準為預設的更多資訊，請參閱 將現有的修補基準設為預設值 (主控台)。
9. 在 Approval rules for operating system (作業系統核准規則) 部分中，使用欄位來建立一或多個自動核准規則。

   - 產品：此核准規則適用的作業系統版本，例如 `WindowsServer2012`。預設的選取為 `All`。
   - Classification (分類)：此核准規則適用的修補程式類型，例如 `CriticalUpdates`、`Drivers` 和 `Tools`。預設的選取為 `All`。

> **Tip**
>
> 透過包含 `ServicePacks` 或在 Classification (分類) 清單中選擇 `All`，您可以在核准規則中包含 Windows Service Pack 安裝。如需範例，請參閱「教學課程：使用主控台建立安裝 Windows Service Pack 的修補基準」。

- 核准規則 (嚴重性)：此規則適用的修補程式嚴重性值，例如 `Critical`。預設的選取為 `All`。
- Auto-approval (自動核准)：選取修補程式以進行自動核准的方法。
  - Approve patches after a specified number of days (指定天數之後核准修補程式)：修補程式發行或更新之後，Patch Manager 自動核准修補程式之前的等待天數。您可以輸入零 (0) 到 360 的任何整數。在大部分情形下，建議等候不要超過 100 天。
  - Approve patches released up to a specific date (核准特定日期前發布的修補程式)：Patch Manager 會自動套用在此發行或更新日期當天或之前發行的所有修補程式。例如，如果您指定 2023 年 7 月 7 日，則不會自動安裝在 2023 年 7 月 8 日或之後發行或最後更新的修補程式。
- (選用) Compliance reporting (合規報告)：您要指派給基準所核准之修補程式的嚴重性等級，例如 `High`。

> **Note**
>
> 如果您指定合規報告等級，且任何核准之修補程式的修補程式狀態回報為 `Missing`，則修補基準的整體報告合規嚴重性是您指定的嚴重性等級。

10. (選用) 在 Approval rules for applications (應用程式的核准規則) 區段中，使用這些欄位建立一個或多個自動核准規則。

> **Note**
>
> 您可以將已核准和已拒絕修補程式清單指定為修補程式例外狀況，而不是指定核准規則。請參閱步驟 10 和 11。

- Product family (產品系列)：您想指定規則的一般 Microsoft 產品系列，例如 `Office` 或 `Exchange Server`。
- 產品：核准規則適用的應用程式版本，例如 `Office 2016` 或 `Active Directory Rights Management Services Client 2.0 2016`。預設的選取為 `All`。

- Classification (分類)：此核准規則適用的修補程式類型，例如 `CriticalUpdates`。預設的選取為 `All`。
- 核准規則 (嚴重性)：此規則適用的修補程式嚴重性值，例如 `Critical`。預設的選取為 `All`。
- Auto-approval (自動核准)：選取修補程式以進行自動核准的方法。
  - Approve patches after a specified number of days (指定天數之後核准修補程式)：修補程式發行或更新之後，Patch Manager 自動核准修補程式之前的等待天數。您可以輸入零 (0) 到 360 的任何整數。在大部分情形下，建議等候不要超過 100 天。
  - Approve patches released up to a specific date (核准特定日期前發布的修補程式)：Patch Manager 會自動套用在此發行或更新日期當天或之前發行的所有修補程式。例如，如果您指定 2023 年 7 月 7 日，則不會自動安裝在 2023 年 7 月 8 日或之後發行或最後更新的修補程式。
- (選用) 合規報告：您要指派給基準所核准之修補程式的嚴重性等級，例如 `Critical` 或 `High`。

> **Note**
>
> 如果您指定合規報告等級，且任何核准之修補程式的修補程式狀態回報為 `Missing`，則修補基準的整體報告合規嚴重性是您指定的嚴重性等級。

11. (選用) 如果您想要明確核准任何修補程式，而非依據核准規則選取修補程式，請在 Patch exceptions (修補程式例外狀況) 區段中進行以下操作：
    - 在 Approved patches (核准的修補程式)，輸入您要核准之修補程式的逗號分隔清單。

      如需已核准修補程式和已拒絕修補程式清單之可接受格式的相關資訊，請參閱 已核准與遭拒的修補程式清單的套件名稱格式。
    - (選用) 在 Approved patches compliance level (核准的修補程式合規層級)中，將合規層級指派至清單中的修補程式。
12. 如果您要明確拒絕任何符合核准規則之修補程式，請在 Patch exceptions (修補程式例外狀況) 部分執行下列動作：
    - 在 Rejected patches (拒絕的修補程式) 中，輸入您要拒絕之修補程式的逗號分隔清單。

      如需已核准修補程式和已拒絕修補程式清單之可接受格式的相關資訊，請參閱 已核准與遭拒的修補程式清單的套件名稱格式。
    - 在 Rejected patches action (已拒絕修補程式動作) 中，選擇要讓 Patch Manager在 Rejected patches (已拒絕修補程式) 清單所包含之修補程式上執行的動作。

- 當做相依性允許：Windows Server 不支援套件相依性的概念。如果遭拒的修補程式清單中的套件已安裝在節點上，則會將其狀態回報為 INSTALLED_OTHER。任何尚未安裝在節點上的套件都會被略過。
- 封鎖：在任何情況下，Patch Manager 都不會安裝遭拒的修補程式清單中的套件。如果在遭拒的修補程式清單中新增套件之前即安裝該套件，或在之後於 Patch Manager 外部安裝該套件，則會被視為不符合修補基準，並且會將其狀態回報為 INSTALLED_REJECTED。

如需有關遭拒的套件動作的詳細資訊，請參閱自訂修補基準中遭拒的修補程式清單選項。

13. (選用) 對於 Manage tags (管理標籤)，請套用一或多個索引鍵名稱/值對至修補基準。

    標籤是您指派給資源的選用性中繼資料。標籤允許您以不同的方式 (例如用途、擁有者或環境) 將資源分類。例如，您可能希望標記修補基準，以識別其指定的修補程式的嚴重性等級、其適用的作業系統系列，以及環境類型。在這種情況下，您可以指定類似以下索引鍵名稱/值對的標籤：

    - Key=PatchSeverity,Value=Critical
    - Key=OS,Value=RHEL
    - Key=Environment,Value=Production

14. 選擇 Create patch baseline (建立修補基準)。

更新或刪除自訂修補基準

您可以更新或刪除您在 中建立的自訂修補程式基準Patch Manager，這是 中的工具 AWS Systems Manager。在更新修補基準時，您可以變更其名稱或說明、核准規則以及已核准和已拒絕的修補程式例外狀況。您也可以更新套用到修補基準的標籤。您無法變更已建立修補程式基準的作業系統類型，也無法變更 提供的預先定義修補程式基準 AWS。

更新或刪除修補基準

請依照以下步驟更新或刪除修補基準。

> **Important**
>
> 刪除 Quick Setup 中修補程式政策組態可能會使用的自訂修補基準時務必小心。
> 如果您在 Quick Setup 中使用修補程式政策組態，則您對自訂修補基準所做的更新會每小時與 Quick Setup 同步一次。
> 如果刪除修補程式政策中參照的自訂修補基準，則修補程式政策的 Quick Setup Configuration details (組態詳細資訊) 頁面上會顯示橫幅。此橫幅會通知您修補程式政策參照修補基準不再存

在，而後續的修補操作將會失敗。在此情況下，請返回到 Quick Setup Configurations (組態) 頁面，選取 Patch Manager 組態，然後選擇 Actions (動作)、Edit configuration (編輯組態)。刪除的修補基準名稱會反白顯示，您必須為受影響的作業系統選取新的修補基準。

更新或刪除修補基準

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Patch Manager。
3. 選擇您要更新或刪除的修補基準，然後執行以下其中一項：
   - 若要從 中移除修補程式基準 AWS 帳戶，請選擇刪除。系統會提示您確認您的動作。
   - 如要變更修補基準名稱或說明、核准規則或修補程式例外狀況，請選擇 Edit (編輯)。在 Edit patch baseline (編輯修補基準) 頁面上，變更您需要的值和選項，然後選擇 Save changes (儲存變更)。
   - 若要新增、變更或刪除套用到修補基準的標籤，請選擇 Tags (標籤) 標籤，然後選擇 Edit tags (編輯標籤)。在 Edit patch baseline tags (編輯修補基準標籤) 頁面上，更新修補基準標籤，然後選擇 Save changes (儲存變更)。

   如需您可以選擇的組態的相關資訊，請參閱 使用自訂修補基準。

將現有的修補基準設為預設值 (主控台)

Important

您在此處選取的任何預設修補基準不會套用至以修補程式政策為基礎的修補操作。修補程式政策使用自己的修補基準規範。如需有關修補程式政策的詳細資訊，請參閱 Quick Setup中的修補程式政策組態。

當您在 中建立自訂修補程式基準Patch Manager時， AWS Systems Manager您可以在建立相關聯的作業系統類型時，將基準設定為預設值。如需相關資訊，請參閱 使用自訂修補基準。

您也可以將現有的修補基準設為作業系統類型的預設值。

> **Note**
>
> 您需要遵循的步驟取決於您是在 2022 年 12 月 22 日修補程式政策發佈之前還是之後首次存取 Patch Manager。如果您在該日期之前使用 Patch Manager，則您可以使用主控台程序。否則，請使用 AWS CLI 程序。在這些修補程式政策發佈之前未使用 Patch Manager 的區域中，主控台程序中所參考的動作選單不會顯示。

將預設修補基準設為預設

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Patch Manager。
3. 選擇 Patch baselines (修補基準) 索引標籤。
4. 在修補基準清單中，請選擇目前未設為作業系統類型預設值的修補基準按鈕。

   Default baseline (預設基準) 欄位會指示目前哪些基線設為預設值。
5. 在 Actions 功能表中，選擇 設定預設修補基準。

   > **Important**
   >
   > 如果您在 2022 年 12 月 22 日之前未在目前 AWS 帳戶 和 Patch Manager 區域中使用 ，則動作功能表無法使用。如需詳細資訊，請參閱本主題稍早的注意部分。
6. 在確認對話方塊中，選擇 Set default (設為預設)。

將預設修補基準設為預設 (AWS CLI)

1. 執行 describe-patch-baselines 命令以檢視可用修補基準及其 ID 和 Amazon 資源名稱 (ARN) 的清單。

   ```
   aws ssm describe-patch-baselines
   ```
2. 執行 register-default-patch-baseline 命令，將一個基準設定為與其相關聯之作業系統的預設值。將 *`baseline-id-or-ARN`* 替換為要使用的自訂修補基準或預先定義基準的 ID。

Linux & macOS

```
aws ssm register-default-patch-baseline \
    --baseline-id baseline-id-or-ARN
```

以下是將自訂基準設定為預設值的範例。

```
aws ssm register-default-patch-baseline \
    --baseline-id pb-abc123cf9bEXAMPLE
```

以下是由 管理的預先定義基準 AWS 做為預設值的設定範例。

```
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
```

以下是將自訂基準設定為預設值的範例。

```
aws ssm register-default-patch-baseline ^
    --baseline-id pb-abc123cf9bEXAMPLE
```

以下是由 管理的預先定義基準 AWS 做為預設值的設定範例。

```
aws ssm register-default-patch-baseline ^
    --baseline-id arn:aws:ssm:us-east-2:733109147000:patchbaseline/
pb-071da192df1226b63
```

檢視可用修補程式

使用 中的Patch Manager工具 AWS Systems Manager，您可以檢視指定作業系統的所有可用修補程式，也可以檢視特定作業系統版本。

> **Tip**
>
> 若要產生可用修補程式的清單並將其儲存到檔案中，您可以使用 describe-available-patches 命令並指定您偏好的輸出。

檢視可用的修補程式

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Patch Manager。
3. 選擇 Patches (修補程式) 索引標籤。

   -或-

   如果您是第一次在目前 AWS 區域存取 Patch Manager，請選擇從概觀開始，然後選擇修補索引標籤。

   > **Note**
   >
   > 對於 Windows Server，修補程式索引標籤會顯示可從 Windows Server Update Service 取得的更新。

4. 對於 Operating system (作業系統)，選擇您要檢視可用修補程式的作業系統，例如 `Windows` 或 `Amazon Linux`。
5. (選用) 對於 Product (產品)，請選擇作業系統版本，例如 `WindowsServer2019` 或 `AmazonLinux2018.03`。
6. (選用) 若要新增或移除結果的資訊欄，請選擇位於 Patches (修補程式) 清單右上角的設定按鈕 (⚙)。
   (根據預設，Patches (修補程式) 索引標籤只會顯示部分可用修補程式中繼資料的欄。)

   如需可新增至檢視的中繼資料類型詳細資訊，請參閱《AWS Systems Manager API 參考》中的修補程式。

建立和管理修補程式群組

如果您未在作業中使用修補程式政策，建議您使用標籤將受管節點新增至修補程式群組來整理您的修補工作。

> **Note**
>
> 修補程式群組不會用於基於修補程式政策的修補操作。如需有關使用修補程式政策的資訊，請參閱 Quick Setup中的修補程式政策組態。
> 在 2022 年 12 月 22 日發行修補程式政策支援之前，尚未使用修補程式群組的帳戶區域對不支援修補程式群組功能。在此日期之前開始使用修補程式群組的帳戶區域對中，修補程式群組功能仍然可用。

若要在修補作業中使用標籤，您必須將標籤鍵鍵 `Patch Group` 或 `PatchGroup` 套用至受管節點。您還必須將要為修補程式群組指定的名稱指定為標籤值。您可以指定任何標籤值，但標籤索引鍵必須是 `Patch Group` 或 `PatchGroup`。

如果您已在 EC2 執行個體中繼資料中允許標籤，則必須使用 `PatchGroup` (不留空格)。

使用標籤將受管節點分組後，請將修補程式群組值新增至修補基準。透過使用修補基準註冊修補程式群組，您可以確保在修補操作期間安裝正確的修補程式。如需有關修補程式群組的詳細資訊，請參閱修補程式群組。

完成本主題中的任務，讓受管節點可以透過節點標籤和修補基準實現修補。只有在修補 Amazon EC2 執行個體時，才需要完成任務 1。只有當您在混合多雲端環境中修補非 EC2 執行個體時，才需要完成任務 2。所有受管節點都需要完成任務 3。

> **Tip**
>
> 您也可以使用 AWS CLI 命令`add-tags-to-resource`或 Systems Manager API 操作 ssm-agent-minimum-s3-permissions-required`AddTagsToResource`，將標籤新增至受管節點。

任務

- 任務 1：使用標籤將 EC2 執行個體新增至修補程式群組
- 任務 2：使用標籤將受管節點新增至修補程式群組
- 任務 3：將修補程式群組新增至修補基準

任務 1：使用標籤將 EC2 執行個體新增至修補程式群組

您可以使用 Systems Manager 主控台或 Amazon EC2 主控台將標籤新增至 EC2 執行個體。只有在修補 Amazon EC2 執行個體時，才需要完成此任務。

> **Important**
>
> 如果在執行個體上啟用 Allow tags in instance metadata (允許執行個體中繼資料中的標籤) 選項，則您不得將 `Patch Group` 標籤 (有空格) 套用至 Amazon EC2 執行個體。允許執行個體中繼資料中的標籤可防止標籤鍵名稱含空格。如果您已在 EC2 執行個體中繼資料中允許標籤，則必須使用標籤索引鍵 `PatchGroup` (不留空格)。

選項 1：將 EC2 執行個體新增至修補程式群組 (Systems Manager 主控台)

1. 開啟 AWS Systems Manager 主控台，網址為 https：//https://console.aws.amazon.com/systems-manager/。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 在受管執行個體清單中選擇您要設定以進行修補之受管 EC2 執行個體的 ID。EC2 執行個體的節點 ID 開頭為 `i-`。

   > **Note**
   >
   > 使用 Amazon EC2 主控台和 時 AWS CLI，可以將 `Key = Patch Group`或 `Key = PatchGroup`標籤套用至尚未設定為與 Systems Manager 搭配使用的執行個體。
   > 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

4. 選擇標籤索引標籤，然後選擇編輯。
5. 在左側欄中，輸入 **`Patch Group`** 或 **`PatchGroup`**。如果您已在 EC2 執行個體中繼資料中允許標籤，則必須使用 `PatchGroup` (不留空格)。
6. 在右側欄中，輸入一個標籤值，用作此修補程式群組的名稱。
7. 選擇儲存。
8. 重複此程序，將其他 EC2 執行個體新增至相同的修補程式群組。

選項 2：將 EC2 執行個體新增至修補程式群組 (Amazon EC2 主控台)

1. 開啟 Amazon EC2 主控台，然後在導覽窗格中選擇 Instances (執行個體)。
2. 在執行個體清單中，選擇要設定修補的執行個體。
3. 在動作功能表中，依次選擇執行個體設定 > 管理標籤。
4. 選擇 Add new tag (新增標籤)。
5. 對於 Key (索引鍵)，輸入 **`Patch Group`** 或 **`PatchGroup`**。如果您已在 EC2 執行個體中繼資料中允許標籤，則必須使用 `PatchGroup` (不留空格)。
6. 針對值，輸入一個值，用作此修補程式群組的名稱。
7. 選擇儲存。
8. 重複此程序，將其他執行個體新增至相同的修補程式群組。

任務 2：使用標籤將受管節點新增至修補程式群組

遵循本主題中的步驟，將標籤新增至 AWS IoT Greengrass 核心裝置non-EC2 混合啟用受管節點 (mi-*)。只有當您在混合多雲端環境中修補非 EC2 執行個體時，才需要完成此任務。

> **Note**
>
> 您不能使用 Amazon EC2 主控台為非 EC2 受管節點新增標籤。

將非 EC2 受管節點新增至修補程式群組 (Systems Manager 主控台)

1. 開啟 AWS Systems Manager 主控台，網址為 https：//https://console.aws.amazon.com/systems-manager/。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 在受管節點清單中，選擇您要設定進行修補的受管節點名稱。

   > **Note**
   >
   > 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

4. 選擇標籤索引標籤，然後選擇編輯。

5. 在左側欄中，輸入 **Patch Group** 或 **PatchGroup**。如果您已在 EC2 執行個體中繼資料中允許標籤，則必須使用 PatchGroup (不留空格)。
6. 在右側欄中，輸入一個標籤值，用作此修補程式群組的名稱。
7. 選擇儲存。
8. 重複此程序以將其他受管節點新增至相同的修補程式群組。

任務 3：將修補程式群組新增至修補基準

要將特定修補基準與受管節點關聯，您必須將修補程式群組值新增至修補基準。透過使用修補基準註冊修補程式群組，您可以確保在修補執行期間安裝正確的修補程式。無論您是修補 EC2 執行個體 或非 EC2 受管節點，還是同時修補兩者，您都需要完成此任務。

如需有關修補程式群組的詳細資訊，請參閱修補程式群組。

> **Note**
>
> 您需要遵循的步驟取決於您是在 2022 年 12 月 22 日修補程式政策發佈之前還是之後首次存取 Patch Manager。

將修補程式群組新增至修補基準 (Systems Manager 主控台)

1. 開啟 AWS Systems Manager 主控台，網址為 https：//https://console.aws.amazon.com/systems-manager/。
2. 在導覽窗格中，選擇 Patch Manager。
3. 如果在目前 AWS 區域 您是第一次存取 Patch Manager，且 Patch Manager 開始頁面開啟，請選擇從概觀開始。
4. 選擇修補基準索引標籤，然後在修補基準清單中，選擇您要為修補程式群組設定的修補基準的名稱。

   如果在修補程式政策發佈之後才首次存取 Patch Manager，則您必須選擇已建立的自訂基準。
5. 如果基準 ID 詳細資訊頁面包含動作選單，請執行下列動作：

   - 選擇 Actions (動作)，然後選擇 Modify patch groups (修改修補程式群組)。
   - 輸入您在 任務 2：使用標籤將受管節點新增至修補程式群組 中新增到受管節點的標籤值，然後選擇新增。

如果基準 ID 詳細資訊頁面未包含動作選單，則無法在主控台中設定修補程式群組。這時您可以執行下列任一操作：

- （建議）在 中設定修補程式政策Quick Setup，這是 中的工具 AWS Systems Manager，可將修補程式基準映射至一或多個 EC2 執行個體。

  如需詳細資訊，請參閱使用修 Quick Setup 補程式政策和使用 Quick Setup 修補程式政策自動化整個組織的修補工作。
- 使用 AWS Command Line Interface (AWS CLI) 中的 register-patch-baseline-for-patch-group命令來設定修補程式群組。

將 AWS Security Hub 與 Patch Manager 整合

AWS Security Hub 可讓您全面檢視 的安全狀態 AWS。Security Hub 會從跨 AWS 帳戶 AWS 服務和支援的第三方合作夥伴產品收集安全資料。使用 Security Hub，您可以檢查環境是否符合安全業界標準和最佳實務。Security Hub 可協助您分析安全趨勢，並識別最高優先級的安全問題。

透過使用 Patch Manager、 中的工具 AWS Systems Manager與 Security Hub 之間的整合，您可以將有關不合規節點的調查結果從 Patch Manager 傳送至 Security Hub。問題清單是安全檢查或安全性相關偵測的可觀察記錄。Security Hub 接著可將這些相關修補問題清單納入其安全狀態的分析中。

無論您使用哪種方法或組態類型進行修補操作，下列主題中的資訊都適用：

- 在 Quick Setup 中設定的修補程式政策
- 在 Quick Setup 中設定的主機管理選項
- 用來執行修補程式 `Scan` 或 `Install` 任務的維護時段
- 隨需 Patch now (立即修補) 操作

內容

- Patch Manager 如何將問題清單傳送到 Security Hub
  - Patch Manager 傳送的問題清單類型
  - 傳送問題清單延遲
  - 無法使用 Security Hub 時重試
  - 檢視 Security Hub 中的調查結果
- 來自 Patch Manager 的一般問題清單

- 開啟與設定整合
- 如何停止傳送問題清單

Patch Manager 如何將問題清單傳送到 Security Hub

在 Security Hub 中，將安全問題作為問題清單進行追蹤。有些問題清單來自其他 AWS 服務 或第三方合作夥伴偵測到的問題。Security Hub 也有一組規則，用來偵測安全問題並產生問題清單。

Patch Manager 是將問題清單傳送到 Security Hub 的 Systems Manager 工具之一。在您執行 SSM 文件 (`AWS-RunPatchBaseline`、 或 `AWS-RunPatchBaselineWithHooks`) 執行修補操作之後`AWS-RunPatchBaselineAssociation`，修補資訊會傳送至庫存或合規部門 AWS Systems Manager、中的工具，或兩者。在「庫存」、「合規」或兩者都收到資料之後，Patch Manager 會收到通知。然後，Patch Manager 在準確性、格式和合規方面對資料進行評估。如果符合所有條件，Patch Manager 會將資料轉寄至 Security Hub。

Security Hub 提供用來跨所有這些來源管理問題清單的工具。您可以檢視並篩選問題清單列表，並檢視問題清單的詳細資訊。如需詳細資訊，請參閱《AWS Security Hub 使用者指南》中的檢視問題清單。您也可以追蹤問題清單的調查狀態。如需詳細資訊，請參閱《AWS Security Hub 使用者指南》中的針對問題清單採取動作。

Security Hub 中的所有問題清單都使用稱為 AWS 安全問題清單格式 (ASFF) 的標準 JSON 格式。ASFF 包含問題來源、受影響的資源以及問題清單目前狀態的詳細資訊。請參閱《AWS Security Hub 使用者指南》中的 AWS 安全問題清單格式 (ASFF)。

Patch Manager 傳送的問題清單類型

Patch Manager 會使用 AWS 安全問題清單格式 (ASFF) 將問題清單傳送到 Security Hub。在 ASFF 中，`Types` 欄位提供問題清單類型。來自 Patch Manager 的問題清單可以具有以下 `Types` 值：

- 軟體和組態檢查/修補程式管理

Patch Manager 會針對每個不合規受管節點傳送一份問題清單。問題清單會以資源類型 `AwsEc2Instance` 報告，讓問題清單可以與報告 `AwsEc2Instance` 資源類型的其他 Security Hub 整合相關聯。Patch Manager 只會在操作發現受管節點不合規時，才將問題清單轉寄至 Security Hub。問題清單包括「修補程式摘要」結果。

> **Note**
>
> 向 Security Hub 報告不合規節點之後。Patch Manager 不會在節點合規之後向 Security Hub 傳送更新。您可以在所需的修補程式套用至受管節點之後，手動解決 Security Hub 中的調查結果。

如需合規定義的詳細資訊，請參閱 修補程式合規狀態值。如需有關 `PatchSummary` 的詳細資訊，請參閱《AWS Security Hub API 參考》中的 PatchSummary。

傳送問題清單延遲

Patch Manager 建立新的問題清單時，通常會在幾秒到 2 小時內傳送至 Security Hub。速度取決於該時間點 AWS 區域 中處理的流量。

無法使用 Security Hub 時重試

如果有服務中斷，則會執行 AWS Lambda 函數，在服務再次執行後將訊息放回主佇列。訊息位於主要佇列之後，會自動重試。

如果 Security Hub 無法使用，Patch Manager 會重試傳送問題清單，直到收到問題清單。

檢視 Security Hub 中的調查結果

此程序說明如何在 Security Hub 中檢視機群中修補程式不合規之受管節點的調查結果。

檢閱 Security Hub 調查結果以檢查修補程式合規性

1. 登入 AWS Management Console ，並在 https：//https://console.aws.amazon.com/securityhub/ 開啟 AWS Security Hub 主控台。
2. 在導覽窗格中，選擇調查結果。
3. 選擇新增篩選條件 ( ) 方塊。
4. 在選單中的篩選條件下，選擇產品名稱。
5. 在開啟的對話方塊中，在第一個欄位中選擇是，然後在第二個欄位中輸入 **`Systems Manager Patch Manager`**。
6. 選擇套用。

7. 新增任何您想要用於縮小搜尋結果範圍的其他篩選條件。
8. 在結果清單中，選擇您想要獲取詳細資訊的調查結果的標題。

   畫面右側會開啟一個窗格，其中會顯示有關資源、發現的問題以及建議的修復方法的詳細資訊。

> **Important**
>
> 此時，Security Hub 會將所有受管節點的資源類型報告為 `EC2 Instance`。這包含您已登記為與 Systems Manager 搭配使用的內部部署伺服器和虛擬機器 (VM)。

嚴重性分類

`Systems Manager Patch Manager` 的調查結果清單包含調查結果嚴重性的報告。嚴重性分下列等級 (程度從最低到最高)：

- 資訊性 – 未發現任何問題。
- 低 – 此問題不需要修復。
- 中 – 此問題必須解決，但不緊急。
- 高 – 此問題必須優先處理。
- 嚴重 – 此問題必須立即修正以免加重。

嚴重性是由執行個體上不合規程度最嚴重的套件所決定。由於您可以擁有多個具有不同嚴重性等級的修補基準，因此會報告所有不合規的套件中最高的嚴重性。例如，假設您有兩個不合規的套件，其中套件 A 的嚴重性為「嚴重」，而套件 B 的嚴重性為「低」。則報告的嚴重性將為「嚴重」。

請注意，嚴重性欄位與 Patch Manager `Compliance` 欄位直接相關。這是您設定並指派給符合規則的個別修補程式的欄位。由於此 `Compliance` 欄位是指派給個別修補程式，因此它不會反映在「修補程式摘要」層級。

相關內容

- 《AWS Security Hub 使用者指南》中的調查結果一節
- AWS 管理與治理部落格中的使用 Patch Manager 和 Security Hub 實現多帳戶修補程式合規

來自 Patch Manager 的一般問題清單

Patch Manager 會使用 AWS 安全問題清單格式 (ASFF) 將問題清單傳送到 Security Hub。

這是來自 Patch Manager 的一般問題清單範例。


```
{
  "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"
  }
}
```

開啟與設定整合

若要使用 Patch Manager 與 Security Hub 的整合，您必須開啟 Security Hub。如需有關如何開啟 Security Hub 的資訊，請參閱《AWS Security Hub 使用者指南》中的設定 Security Hub。

下列處理程序描述了如何在 Security Hub 已經處於作用中狀態但 Patch Manager 整合關閉時將 Patch Manager 與 Security Hub 整合。只有在手動關閉整合時，您才需要完成此處理程序。

新增 Patch Manager 至 Security Hub 整合

1. 在導覽窗格中，選擇 Patch Manager。
2. 選擇 Settings (設定) 標籤。

   -或-

   如果您是第一次在目前 AWS 區域存取 Patch Manager，請選擇從概觀開始，然後選擇設定索引標籤。

3. 在 Export to Security Hub (匯出至 Security Hub) 區段下，Patch compliance findings aren't being exported to Security Hub (修補程式合規問題清單未匯出至 Security Hub) 的右側，選擇 Enable (啟用)。

如何停止傳送問題清單

若要停止將問題清單傳送至 Security Hub，您可以使用 Security Hub 主控台或 API。

如需詳細資訊，請參閱《AWS Security Hub 使用者指南》中的以下主題：

- 停用和啟用從整合接收問題清單的流程 (主控台)
- 從整合停用問題清單流程 (Security Hub API AWS CLI)

透過AWS CLI使用 Patch Manager 資源

本節包含 AWS Command Line Interface (AWS CLI) 命令的範例，您可以用來執行 Patch Manager中的工具 的組態任務 AWS Systems Manager。

如需使用 AWS CLI 來透過自訂修補基準修補伺服器環境的圖例，請參閱 教學課程：使用 修補伺服器環境 AWS CLI。

如需使用 AWS CLI AWS Systems Manager 任務的詳細資訊，請參閱 AWS Systems ManagerAWS CLI 命令參考的 一節。

主題

- AWS CLI 修補程式基準的 命令
- AWS CLI 修補程式群組的 命令
- AWS CLI 檢視修補程式摘要和詳細資訊的 命令
- AWS CLI 用於掃描和修補受管節點的 命令

AWS CLI 修補程式基準的 命令

修補基準的範例命令

- 建立修補基準
- 建立包含不同作業系統版本之自訂儲存庫的修補基準
- 更新修補基準
- 重新命名修補基準

- 刪除修補基準
- 列出所有修補基準
- 列出所有 AWS提供的修補程式基準
- 列出我的修補基準
- 顯示修補基準
- 取得預設的修補基準
- 將自訂修補基準設定為預設
- 重設 AWS 修補基準為預設值
- 標記修補基準
- 列出修補基準的標記
- 從修補基準移除標記

建立修補基準

以下命令建立修補基準，在 Windows Server 2012 R2 5 的所有重大和重要安全性更新發行 5 日之後，核准這些更新。也已針對「已核准」和「已拒絕」修補程式清單指定修補程式。此外，修補基準已加上標籤，以表示其用於生產環境。

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}]"
```

系統會傳回相關資訊，如下所示。

```
{
   "BaselineId":"pb-0c10e65780EXAMPLE"
}
```

建立包含不同作業系統版本之自訂儲存庫的修補基準

僅適用於 Linux 受管節點。以下命令說明如何指定修補程式儲存庫，以用於特定版本的 Amazon Linux 作業系統。此範例使用 Amazon Linux 2017.09 預設啟用的來源儲存庫，但可適應您已為受管節點設定的不同來源儲存庫。

> **Note**
>
> 為了更好的展示這個更為複雜的命令，我們使用 `--cli-input-json` 選項以及存放外部 JSON 檔案的其他選項。

1. 以類似 `my-patch-repository.json` 的名稱建立 JSON 檔案，並將以下內容新增至該檔案：

```
{
    "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. 在您儲存該檔案的目錄中執行下列命令。

```
aws ssm create-patch-baseline --cli-input-json file://my-patch-repository.json
```

系統會傳回相關資訊，如下所示。

```
{
    "BaselineId": "pb-0c10e65780EXAMPLE"
}
```

更新修補基準

以下命令新增兩個修補程式以拒絕現有的修補基準，另一個修補程式核准現有的修補基準。

如需已核准修補程式和已拒絕修補程式清單之可接受格式的相關資訊，請參閱 已核准與遭拒的修補程式清單的套件名稱格式。

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"
```

系統會傳回相關資訊，如下所示。

```
{
   "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"
}
```

重新命名修補基準

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"
```

系統會傳回相關資訊，如下所示。

```
{
   "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"
}
```

刪除修補基準

```
aws ssm delete-patch-baseline --baseline-id "pb-0c10e65780EXAMPLE"
```

系統會傳回相關資訊，如下所示。

```
{
   "BaselineId":"pb-0c10e65780EXAMPLE"
}
```

列出所有修補基準

```
aws ssm describe-patch-baselines
```

系統會傳回相關資訊，如下所示。

```
{
   "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"
      }
   ]
}
```

以下是另一個命令，列出 AWS 區域中的所有修補基準。

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]"
```

系統會傳回相關資訊，如下所示。

```
{
   "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提供的修補程式基準

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]"
```

系統會傳回相關資訊，如下所示。

```
{
   "BaselineIdentities":[
      {
         "BaselineName":"AWS-DefaultPatchBaseline",
         "DefaultBaseline":true,
         "BaselineDescription":"Default Patch Baseline Provided by AWS.",
         "BaselineId":"arn:aws:ssm:us-east-2:111122223333:patchbaseline/
pb-0c10e65780EXAMPLE"
      }
   ]
}
```

列出我的修補基準

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]"
```

系統會傳回相關資訊，如下所示。

```
{
   "BaselineIdentities":[
      {
         "BaselineName":"Windows-Server-2012R2",
         "DefaultBaseline":false,
         "BaselineDescription":"Windows Server 2012 R2, Important and Critical security
 updates",
         "BaselineId":"pb-0c10e65780EXAMPLE"
      }
   ]
}
```

顯示修補基準

```
aws ssm get-patch-baseline --baseline-id pb-0c10e65780EXAMPLE
```

> **Note**
>
> 若為自訂修補基準，您可以指定修補基準 ID 或完整的 Amazon Resource Name (ARN)。對於 AWS提供的修補程式基準，您必須指定完整的 ARN。例如：`arn:aws:ssm:us-east-2:075727635805:patchbaseline/pb-0c10e65780EXAMPLE`。

系統會傳回相關資訊，如下所示。

```
{
   "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"
}
```

取得預設的修補基準

```
aws ssm get-default-patch-baseline --region us-east-2
```

系統會傳回相關資訊，如下所示。

```
{
   "BaselineId":"arn:aws:ssm:us-east-2:111122223333:patchbaseline/pb-0c10e65780EXAMPLE"
}
```

將自訂修補基準設定為預設

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"
```

系統會傳回相關資訊，如下所示。

```
{
   "BaselineId":"pb-0c10e65780EXAMPLE"
}
```

重設 AWS 修補基準為預設值

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"
```

系統會傳回相關資訊，如下所示。

```
{
   "BaselineId":"pb-0c10e65780EXAMPLE"
}
```

標記修補基準

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"
```

列出修補基準的標記

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"
```

從修補基準移除標記

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"
```

AWS CLI 修補程式群組的 命令

修補程式群組的範例命令

- 建立修補程式群組
- 向修補程式群組「Web Servers」註冊修補基準
- 使用 AWS提供的修補程式基準註冊修補程式群組 "Backend"
- 顯示修補程式群組登錄
- 從修補基準重新登錄修補程式群組

建立修補程式群組

> **Note**
>
> 修補程式群組不會用於基於修補程式政策的修補操作。如需有關使用修補程式政策的資訊，請參閱 Quick Setup中的修補程式政策組態。

為了協助您組織修補工作，建議您使用標籤將受管節點新增至修補程式群組。修補程式群組需要使用標籤索引鍵 `Patch Group` 或 `PatchGroup`。如果您已在 EC2 執行個體中繼資料中允許標籤，則必須使用 `PatchGroup` (不留空格)。您可以指定任何標籤值，但標籤索引鍵必須是 `Patch Group` 或 `PatchGroup`。如需有關修補程式群組的詳細資訊，請參閱修補程式群組。

使用標籤將受管節點分組後，請將修補程式群組值新增至修補基準。透過使用修補基準註冊修補程式群組，您可以確保在修補操作期間安裝正確的修補程式。

任務 1：使用標籤將 EC2 執行個體新增至修補程式群組

> **Note**
>
> 使用 Amazon Elastic Compute Cloud (Amazon EC2) 主控台和 時 AWS CLI，可以將 `Key = Patch Group`或 `Key = PatchGroup` 標籤套用至尚未設定為與 Systems Manager 搭配使用的執行個體。如果套用 `Patch Group` 或 `Key = PatchGroup` 標籤後您預期在 Patch Manager 中看到的 EC2 執行個體未列出，請參閱 疑難排解受管節點的可用性 以取得故障診斷秘訣。

執行以下命令來將 `PatchGroup` 標籤新增到 EC2 執行個體。

```
aws ec2 create-tags --resources "i-1234567890abcdef0" --tags
 "Key=PatchGroup,Value=GroupValue"
```

任務 2：使用標籤將受管節點新增至修補程式群組

執行以下命令，來將 `PatchGroup` 標籤新增到受管節點。

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"
```

任務 3：將修補程式群組新增至修補基準

執行以下命令，來將 PatchGroup 標籤值與指定的修補程式基線建立關聯。

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"
```

系統會傳回相關資訊，如下所示。

```
{
  "PatchGroup": "Development",
  "BaselineId": "pb-0c10e65780EXAMPLE"
}
```

向修補程式群組「Web Servers」註冊修補基準

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"
```

系統會傳回相關資訊，如下所示。

```
{
   "PatchGroup":"Web Servers",
   "BaselineId":"pb-0c10e65780EXAMPLE"
}
```

使用 AWS提供的修補程式基準註冊修補程式群組 "Backend"

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"
```

系統會傳回相關資訊，如下所示。

```
{
   "PatchGroup":"Backend",
   "BaselineId":"arn:aws:ssm:us-east-2:111122223333:patchbaseline/pb-0c10e65780EXAMPLE"
```

```
}
```

顯示修補程式群組登錄

```
aws ssm describe-patch-groups --region us-east-2
```

系統會傳回相關資訊，如下所示。

```
{
   "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"
         }
      }
   ]
}
```

從修補基準重新登錄修補程式群組

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"
```

系統會傳回相關資訊，如下所示。

```
{
   "PatchGroup":"Production",
   "BaselineId":"arn:aws:ssm:us-east-2:111122223333:patchbaseline/pb-0c10e65780EXAMPLE"
}
```

AWS CLI 檢視修補程式摘要和詳細資訊的 命令

用於檢視修補程式摘要和詳細資訊的範例命令

- 取得修補基準定義的所有修補程式
- 為擁有 SECURITY 分類和 Critical 嚴重性的 AmazonLinux2018.03 取得全部修補程式。
- 為 Windows Server 2012 取得 CriticalMSRC 嚴重性的所有修補程式
- 取得所有可用的修補程式
- 取得每個受管節點的修補程式摘要狀態
- 取得受管節點的修補程式合規詳細資訊
- 檢視修補程式合規結果 (AWS CLI)

取得修補基準定義的所有修補程式

> **Note**
>
> 此命令僅支援 Windows Server 修補基準。

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"
```

系統會傳回相關資訊，如下所示。


```
{
   "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---
```

為擁有 **SECURITY** 分類和 **Critical** 嚴重性的 AmazonLinux2018.03 取得全部修補程式。

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
```

系統會傳回相關資訊，如下所示。

```
{
    "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---
```

為 Windows Server 2012 取得 **Critical**MSRC 嚴重性的所有修補程式

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
```

系統會傳回相關資訊，如下所示。

```
{
   "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---
```

取得所有可用的修補程式

```
aws ssm describe-available-patches --region us-east-2
```

系統會傳回相關資訊，如下所示。

```
{
   "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---
```

取得每個受管節點的修補程式摘要狀態

各受管節點摘要提供各節點處於以下狀態的修補程式數量："NotApplicable"、"Missing"、"Failed"、"InstalledOther" 和 "Installed"。

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
```

系統會傳回相關資訊，如下所示。

```
{
   "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---
```

取得受管節點的修補程式合規詳細資訊

```
aws ssm describe-instance-patches --instance-id i-08ee91c0b17045407
```

系統會傳回相關資訊，如下所示。

```
{
   "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---
```

檢視修補程式合規結果 (AWS CLI)

為單一受管節點檢視修補程式合規結果

在 AWS Command Line Interface (AWS CLI) 中執行下列命令，以檢視單一受管節點的修補程式合規結果。

```
aws ssm describe-instance-patch-states --instance-id instance-id
```

使用您想要檢視結果之受管節點的 ID 取代 *instance-id*，格式為 i-02573cafcfEXAMPLE 或 mi-0282f7c436EXAMPLE。

系統傳回的資訊如下。

```
{
    "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"
        }
    ]
}
```

檢視區域中所有 EC2 執行個體的修補程式計數摘要

`describe-instance-patch-states` 支援一次只擷取一個受管執行個體的結果。不過，使用具有 `describe-instance-patch-states` 命令的自訂指令碼，您可以產生更精密的報告。

例如，如果在本地計算機上安裝了 jq 篩選工具，則您可以執行以下命令來識別特定 AWS 區域 中狀態為 `InstalledPendingReboot` 的 EC2 執行個體。

```
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 代表 支援的 AWS 區域 識別符 AWS Systems Manager，例如`us-east-2`美國東部（俄亥俄）區域。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

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}'
```

系統會傳回相關資訊，如下所示。

```
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，您可以搜尋的計數類型清單包括下列項目：

- CriticalNonCompliantCount
- SecurityNonCompliantCount
- OtherNonCompliantCount
- UnreportedNotApplicableCount
- InstalledPendingRebootCount
- FailedCount
- NotApplicableCount
- InstalledRejectedCount
- InstalledOtherCount
- MissingCount
- InstalledCount

AWS CLI 用於掃描和修補受管節點的 命令

執行下列命令以掃描檢查修補程式是否合規或安裝修補程式之後，您可以使用 AWS CLI 檢視修補程式摘要和詳細資訊的 命令 區段中的命令，以檢視修補程式狀態和合規的相關資訊。

範例命令

- 掃描受管節點，檢查修補程式是否合規 (AWS CLI)
- 在受管節點上安裝修補程式 (AWS CLI)

掃描受管節點，檢查修補程式是否合規 (AWS CLI)

掃描特定受管節點，檢查修補程式是否合規

執行下列命令。

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
```

系統會傳回相關資訊，如下所示。

```
{
    "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---

    }
}
```

依修補程式群組標籤掃描受管節點，檢查修補程式是否合規

執行下列命令。

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
```

系統會傳回相關資訊，如下所示。

```
{
    "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---

    }
}
```

在受管節點上安裝修補程式 (AWS CLI)

在特定受管節點上安裝修補程式

執行下列命令。

> **Note**
>
> 視需要重新啟動目標受管節點，以完成修補程式安裝。如需詳細資訊，請參閱用於修補的 SSM 命令文件：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
```

系統會傳回相關資訊，如下所示。

```
{
    "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---

    }
}
```

在特定修補程式群組的受管節點上安裝修補程式

執行下列命令。

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
```

系統會傳回相關資訊，如下所示。

```
{
    "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 ManagerPatch Manager 教學課程

本節中的教學課程示範如何針對數個修補案例使用 中的Patch Manager工具 AWS Systems Manager。

主題

- 教學課程：使用主控台建立安裝 Windows Service Pack 的修補基準
- 教學課程：使用主控台更新應用程式相依性、修補受管節點，以及執行應用程式特定的運作狀態檢查
- 教學課程：使用 修補伺服器環境 AWS CLI

教學課程：使用主控台建立安裝 Windows Service Pack 的修補基準

當您建立自訂修補基準時，可以指定安裝所有、部分或僅安裝一種支援的修補程式類型。

在 Windows 的修補基準中，您可以選取 `ServicePacks` 做為唯一的 Classification (分類) 選項，以限制僅修補 Service Pack 的更新。Service Pack 可由 中的工具 Patch Manager自動安裝 AWS Systems Manager，前提是更新可在 Windows Update 或 Windows Server Update Services (WSUS) 中使用。

您可以設定修補基準，以控制是否安裝所有 Windows 版本的 Service Pack，或只安裝特定版本 (例如 Windows 7 或 Windows Server 2016) 的 Service Pack。

使用下列程序建立自訂修補基準，專門用在 Windows 受管節點上安裝所有 Service Pack。

建立修補基準用於安裝 Windows Service Pack (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Patch Manager。
3. 選擇修補基準索引標籤，然後選擇建立修補基準。
4. 在 Name (名稱) 中，為新的修補基準輸入一個名稱，例如 `MyWindowsServicePackPatchBaseline`。
5. (選用) 在 Description (描述) 中，輸入此修補基準的描述。

6. 在 Operating system (作業系統) 中，選擇 `Windows`。
7. 如果要在建立 Windows 時立即開始將此修補基準做為 Windows 的預設設定，請選擇 Set this patch baseline as the default patch baseline for Windows Server instances (將此修補基準設為 Windows Server 執行個體的預設修補基準)。

> **Note**
>
> 唯有當您在 2022 年 12 月 22 日修補程式政策發行前第一次存取 Patch Manager，才能使用此選項。
> 如需有關設定現有修補基準為預設的更多資訊，請參閱 將現有的修補基準設為預設值 (主控台)。

8. 在 Approval rules for operating system (作業系統核准規則) 部分中，使用欄位來建立一或多個自動核准規則。

   - 產品：核准規則適用的作業系統版本，例如 `WindowsServer2012`。您可以選擇一個、多個或所有支援的 Windows 版本。預設的選取為 `All`。
   - Classification (分類)：選擇 `ServicePacks`。
   - Serverity (嚴重性)：此規則適用的修補程式嚴重性值。若要確保所有 Service Pack 都包含在規則中，請選擇 `All`。
   - Auto-approval (自動核准)：選取修補程式以進行自動核准的方法。
     - Approve patches after a specified number of days (指定天數之後核准修補程式)：修補程式發行或更新之後，Patch Manager 自動核准修補程式之前的等待天數。您可以輸入零 (0) 到 360 的任何整數。在大部分情形下，建議等候不要超過 100 天。
     - Approve patches released up to a specific date (核准特定日期前發布的修補程式)：Patch Manager 會自動套用在此發行或更新日期當天或之前發行的所有修補程式。例如，如果您指定 2023 年 7 月 7 日，則不會自動安裝在 2023 年 7 月 8 日或之後發行或最後更新的修補程式。
   - (選用) Compliance reporting (合規報告)：您要指派給基準所核准之 Service Pack 的嚴重性等級，例如 `High`。

> **Note**
>
> 如果您指定合規報告等級，且任何核准之 Service Pack 的修補程式狀態回報為 `Missing`，則修補基準的整體報告合規嚴重性是您指定的嚴重性等級。

9. (選用) 對於 Manage tags (管理標籤)，請套用一或多個索引鍵名稱/值對至修補基準。

   標籤是您指派給資源的選用性中繼資料。標籤允許您以不同的方式 (例如用途、擁有者或環境) 將資源分類。針對此專用於更新 Service Pack 的修補基準，您可以指定如下所示的鍵值組：

   - `Key=OS,Value=Windows`
   - `Key=Classification,Value=ServicePacks`
10. 選擇 Create patch baseline (建立修補基準)。

教學課程：使用主控台更新應用程式相依性、修補受管節點，以及執行應用程式特定的運作狀態檢查

在許多情況下，受管節點必須在使用最新的軟體更新修補後重新啟動。不過，在沒有保護措施的生產環境中重新啟動節點，可能會導致數個問題，例如叫用警示、記錄不正確的指標資料，以及中斷資料同步。

此教學課程會示範如何透過使用 AWS Systems Manager 文件 (SSM 文件) `AWS-RunPatchBaselineWithHooks` 達成完成了以下項目的複雜、多步驟修補操作，以避免類似問題：

1. 防止新連線至應用程式
2. 安裝作業系統的更新。
3. 更新應用程式的套件相依性
4. 重新啟動系統
5. 執行應用程式特定的運作狀態檢查

在此範例中，以這種方式設定了基礎設施：

- 目標虛擬機器會透過 Systems Manager 註冊為受管節點。
- `Iptables` 會用作本機防火牆。
- 受管節點上託管的應用程式正在連接埠 443 上執行。
- 受管節點上託管的應用程式是 `nodeJS` 應用程式。
- 受管節點上託管的應用程式由 pm2 程序管理工具管理。
- 應用程式已經有指定的運作狀態檢查端點。
- 應用程式的運作狀態檢查端點不需要最終使用者身分驗證。端點允許執行符合組織建立可用性需求的運作狀態檢查。(在您的環境中，完全可以簡單地確定 `nodeJS` 應用程式正在執行，並能夠監聽請求。在其他情形下，您可能還想要驗證快取層或資料庫層的連線是否已經建立)。

本教學課程中的範例僅供示範之用，並不代表在生產環境中以原樣實作。此外，請記住，Systems Manager 中的Patch Manager工具 的生命週期掛鉤功能AWS-RunPatchBaselineWithHooks，可支援許多其他案例。以下是數個範例。

- 受管節點重新啟動之後，在修補和重新啟動之前，停止指標報告代理程式。
- 修補之前將受管節點從 CRM 或 PCS 叢集分開，並在節點重新啟動之後重新連接。
- 作業系統 (OS) 更新套用之後，但在受管節點重新開機之前，在 Windows Server 機器上更新第三方軟體 (例如，Java、Tomcat、Adobe 應用程式等)。

更新應用程式相依性、修補受管節點，以及執行應用程式特定的運作狀態檢查

1. 使用下列內容建立預先安裝指令碼的 SSM 文件，並命名為 NodeJSAppPrePatch。使用應用程式的名稱取代 *your_application*。

   此指令碼會立即封鎖新的傳入請求，並在開始修補操作之前提供五秒鐘，讓已處於作用中狀態的請求完成。對於 sleep 選項，指定的秒數要大於完成傳入請求通常所需的秒數。

   ```
   # 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
   ```

   如需有關建立 SSM 文件的資訊，請參閱 建立 SSM 文件內容。

2. 為您的安裝後指令碼建立具有以下內容的另一個 SSM 文件，以更新應用程式相依性並將其命名為 NodeJSAppPostPatch。使用您應用程式的路徑取代 */your/application/path*。

   ```
   cd /your/application/path
   npm update
   # you can use npm-check-updates if you want to upgrade major versions
   ```

3. 為您的 onExit 指令碼建立另一個 SSM 文件，其中包含下列內容，讓您的應用程式備份並執行運作狀態檢查。命名此 SSM 文件 NodeJSAppOnExitPatch。使用應用程式的名稱取代 *your_application*。

```
# 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 中的State Manager工具 中建立關聯 AWS Systems Manager，以執行下列步驟來發出操作：

   1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
   2. 在導覽窗格中，選擇 State Manager，然後選擇 Create association (建立關聯)。
   3. 對於 Name (名稱)，請提供名稱以協助識別關聯的目的。
   4. 在 Document (文件) 清單中，請選擇 `AWS-RunPatchBaselineWithHooks`。
   5. 針對 Operation (操作)，選擇 Install (安裝)。
   6. (選用) 對於 Snapshot Id (快照 ID)，提供您產生的 GUID，以協助加速操作並確保一致性。GUID 值可以像 `00000000-0000-0000-0000-111122223333` 一樣簡單。
   7. 對於 Pre Install Hook Doc Name (安裝前掛鉤文件名稱)，請輸入 `NodeJSAppPrePatch`。
   8. 對於 Post Install Hook Doc Name (安裝後掛鉤文件名稱)，請輸入 `NodeJSAppPostPatch`。
   9. 對於 On ExitHook Doc Name (結束時掛鉤文件名稱)，請輸入 `NodeJSAppOnExitPatch`。
5. 對於 Targets (目標)，透過指定標籤、手動選擇節點、選擇資源群組或選擇所有受管節點來識別您的受管節點。
6. 對於 Specify schedule (指定排程)，請指定執行關聯的頻率。對於受管節點修補，每週一次是常見頻率。
7. 在 Rate control (速率控制) 區段中，選擇選項來控制關聯在多個受管節點上的執行方式。確保一次僅更新部分受管節點。否則，您的所有或大部分機群都可能立即離線。如需使用速率控制的詳細資訊，請參閱了解 State Manager 關聯中的目標和速率控制。
8. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取啟用將輸出寫入 S3 方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

> **Note**
>
> 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給受管節點之執行個體設定檔的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確認與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

9. 選擇 Create Association (建立關聯)。

教學課程：使用 修補伺服器環境 AWS CLI

以下程序說明如何使用自訂修補程式基線、修補程式群組及維護時段來修補伺服器環境。

開始之前

- 在受管節點上安裝或更新 SSM Agent。若要修補 Linux 受管節點，您的節點必須執行 SSM Agent 2.0.834.0 版或更新的版本。如需詳細資訊，請參閱使用 Run Command 更新 SSM Agent。
- 設定 中Maintenance Windows工具 的角色和許可 AWS Systems Manager。如需詳細資訊，請參閱設定 Maintenance Windows。
- 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

  如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

設定 Patch Manager 並修補受管節點 (命令列)

1. 執行下列命令以建立名為 `Production-Baseline` 的 Windows 修補基準。此修補基準會在修補程式發行或最後更新 7 天後核准生產環境的修補程式。也就是說，我們已標記修補基準，以表示其用於生產環境。

> **Note**
>
> `OperatingSystem` 參數和 `PatchFilters` 取決於修補基準所套用之目標受管節點的作業系統。如需詳細資訊，請參閱 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"
```

系統會傳回相關資訊，如下所示。

```
{
   "BaselineId":"pb-0c10e65780EXAMPLE"
}
```

2. 執行下列命令，以註冊兩個修補程式群組的「Production-Baseline」修補基準。群組命名為「資料庫伺服器」和「前端伺服器」。

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"
```

系統會傳回相關資訊，如下所示。

```
{
   "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"
```

系統會傳回相關資訊，如下所示。

```
{
   "PatchGroup":"Front-End Servers",
   "BaselineId":"pb-0c10e65780EXAMPLE"
}
```

3. 執行以下命令來為生產伺服器建立兩個維護時段。第一個時段為每個星期二的晚上 10 點。第二個時段為每個星期六的晚上 10 點。此外，維護時段已加上標籤，表示其用於生產環境。

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
```

系統會傳回相關資訊，如下所示。

```
{
   "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
```

系統會傳回相關資訊，如下所示。

```
{
   "WindowId":"mw-9a8b7c6d5eEXAMPLE"
}
```

4. 執行下列命令，將 Database 和 Front-End 伺服器修補程式群組註冊到各自的維護時段。

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"
```

系統會傳回相關資訊，如下所示。

```
{
   "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"
```

系統會傳回相關資訊，如下所示。

```
{
   "WindowTargetId":"faa01c41-1d57-496c-ba77-ff9caEXAMPLE"
}
```

5. 執行下列命令以註冊修補程式任務，該任務會在 Database 和 Front-End 伺服器各自的維護時段上安裝伺服器缺少的更新。

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}}"
```

系統會傳回相關資訊，如下所示。

```
{
   "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}}"
```

系統會傳回相關資訊，如下所示。

```
{
   "WindowTaskId":"8a5c4629-31b0-4edd-8aea-33698EXAMPLE"
}
```

6. 執行以下命令以取得修補程式群組的高階修補程式合規摘要。高階修補程式合規摘要包括處於各別修補程式狀態之修補程式的受管節點數目。

> **Note**
>
> 在第一個維護時段期間執行修補程式任務之前，摘要中的受管節點數目預計為零。

Linux & macOS

```
aws ssm describe-patch-group-state \
    --patch-group "Database Servers"
```

Windows Server

```
aws ssm describe-patch-group-state ^
    --patch-group "Database Servers"
```

系統會傳回相關資訊，如下所示。

```
{
   "Instances": number,
   "InstancesWithFailedPatches": number,
   "InstancesWithInstalledOtherPatches": number,
   "InstancesWithInstalledPatches": number,
```

```
    "InstancesWithInstalledPendingRebootPatches": number,
    "InstancesWithInstalledRejectedPatches": number,
    "InstancesWithMissingPatches": number,
    "InstancesWithNotApplicablePatches": number,
    "InstancesWithUnreportedNotApplicablePatches": number
}
```

7. 執行以下命令以取得修補程式群組中各受管節點的修補程式摘要狀態。每個受管節點摘要包括修補群組之每個受管節點中，處於個別修補程式狀態的多個修補程式。

   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"
   ```

   系統會傳回相關資訊，如下所示。

   ```
   {
      "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
        }
    ]
}
```

如需可用於Patch Manager組態任務的其他 AWS CLI 命令範例，請參閱 透過AWS CLI使用 Patch Manager 資源。

Patch Manager 疑難排解

使用以下資訊來協助您疑難排解 中的Patch Manager工具 的問題 AWS Systems Manager。

主題

- 問題：baseline_overrides.json 的 "Invoke-PatchBaselineOperation : Access Denied" 錯誤或 "Unable to download file from S3" 錯誤
- 問題：修補失敗且沒有明顯的原因或錯誤訊息
- 問題：非預期的修補程式合規結果
- 在 Linux 上執行 AWS-RunPatchBaseline 時發生錯誤
- 在 Windows Server 上執行 AWS-RunPatchBaseline 時發生錯誤
- 使用 AWS 支援 自動化 Runbook
- 聯絡 AWS 支援

問題：**baseline_overrides.json** 的 "Invoke-PatchBaselineOperation : Access Denied" 錯誤或 "Unable to download file from S3" 錯誤

問題：執行修補程式政策指定的修補作業時，您收到類似以下範例的錯誤。

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
```

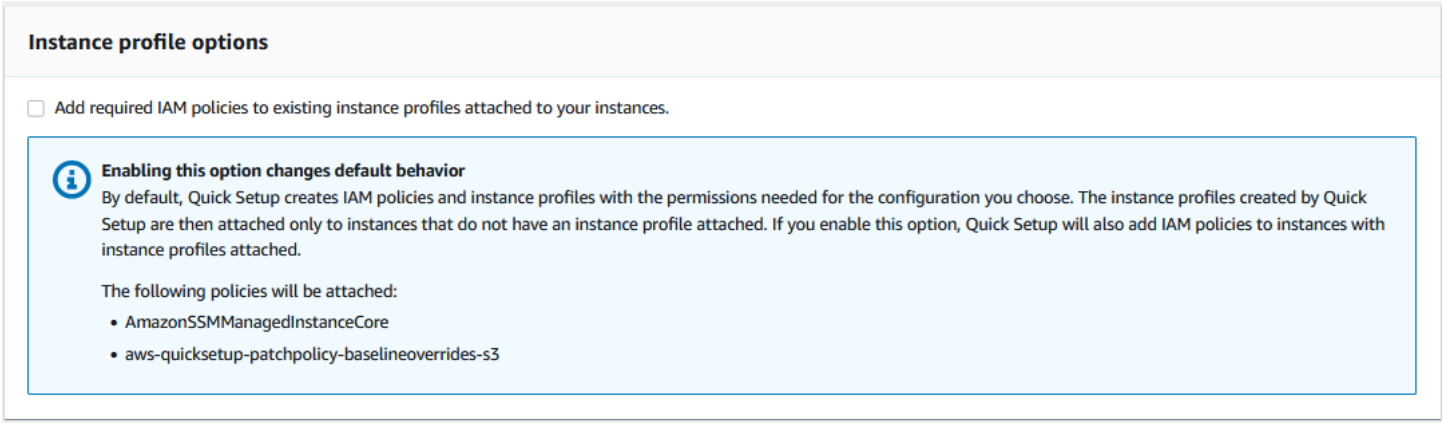
```
tion:InstallWindowsUpdateOperation) [Invoke-PatchBaselineOperation], Amazo
nS3Exception
+ FullyQualifiedErrorId : PatchBaselineOperations,Amazon.Patch.Baseline.Op
erations.PowerShellCmdlets.InvokePatchBaselineOperation
failed to run commands: exit status 0xffffffff
```

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：您已在 Quick Setup 中建立修補程式政策，並且部分受管節點已連接執行個體設定檔 (適用於 EC2 執行個體) 或已連接服務角色 (適用於非 EC2 機器)。

然而，如下圖所示，您並未選取在連接至執行個體的現有執行個體設定檔中新增必要的 IAM 政策核取方塊。



建立修補程式政策時，也會建立 Amazon S3 儲存貯體來存放政策的組態 `baseline_overrides.json` 檔案。如果您在建立政策時未選取將必要的 IAM 政策新增至連接至執行個體的現有執行個體設定檔核取方塊，則存取 S3 儲存貯體中的 `baseline_overrides.json` 所需的 IAM 政策和資源標籤不會自動新增至現有的 IAM 執行個體設定檔和服務角色。

解決方案 1：刪除現有的修補程式政策組態，然後建立替代項目，並在建立過程中確保選取將必要的 IAM 政策新增至連接至執行個體的現有執行個體設定檔核取方塊。此選項會將此 Quick Setup 組態建立的 IAM 政策套用至已連接執行個體設定檔或服務角色的節點。(根據預設，Quick Setup 會將必要的政策新增至仍沒有執行個體設定檔或服務角色的執行個體和節點。) 如需詳細資訊，請參閱使用 Quick Setup 修補程式政策自動化整個組織的修補工作。

解決方案 2：手動將必要的許可和標籤新增至每個您搭配 Quick Setup 使用的 IAM 執行個體設定檔和 IAM 服務角色。如需說明，請參閱適用於修補程式政策 S3 儲存貯體的許可。

問題：修補失敗且沒有明顯的原因或錯誤訊息

問題：修補作業失敗且未傳回錯誤訊息。

可能的原因：如果一次發生多個 AWS-RunPatchBaseline 調用，這些調用之間可能會發生衝突，從而導致修補任務失敗。這可能不會在修補日誌中得到反映。

若要檢查並行修補操作是否可能彼此中斷，請檢閱 中的命令歷史記錄Run Command，這是 中的工具 AWS Systems Manager。對於發生修補失敗的受管節點，請檢查嘗試修補機器的多個作業之間的時間間隔是否不到 2 分鐘。有時候，這種情況可能會導致失敗。

您也可以使用以下命令，使用 AWS Command Line Interface (AWS CLI) 檢查並行修補嘗試。將 *node-id* 替換為受管節點的 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
```

解決方案：如果您判斷修補因為相同受管節點上的修補作業競爭而失敗，請調整修補組態以避免再次發生這種情況。例如，如果兩個維護時段指定了重疊的修補時間，請移除或修訂其中一個維護時段。如果某個維護時段指定了一項修補作業，但修補程式政策在同一時間指定了不同的修補作業，請考慮從維護時段移除相關作業。

如果您判斷發生衝突的修補作業不是造成此情境中的失敗的原因，建議您聯絡 AWS 支援。

問題：非預期的修補程式合規結果

問題：檢閱 Scan 操作之後產生的修補程式合規詳細資訊時，結果包含未反映修補基準中設定的規則之資訊。例如，您在修補基準中新增至 Rejected patches (已拒絕的修補程式) 清單的例外狀況會列為 Missing。或者，即使修補基準僅指定 Critical 修補程式，分類為 Important 的修補程式仍會列為遺失。

原因：Patch Manager 目前支援多種執行 Scan 操作的方法：

- 在 Quick Setup 中設定的修補程式政策

- 在 Quick Setup 中設定的主機管理選項
- 用來執行修補程式 Scan 或 Install 任務的維護時段
- 隨需 Patch now (立即修補) 操作

當 Scan 操作執行時，其會覆寫最近掃描的合規詳細資訊。如果您已設定多個方法來執行 Scan 操作，而且其使用具有不同規則的不同修補基準，則會產生不同的修補程式合規結果。

解決方案：為避免非預期的修補程式合規結果，建議您一次僅使用一種方法來執行 Patch Manager Scan 操作。如需詳細資訊，請參閱避免意外覆寫修補程式合規資料。

在 Linux 上執行 **AWS-RunPatchBaseline** 時發生錯誤

主題

- 問題：'No such file or directory' (沒有該檔案或目錄) 錯誤
- 問題：'another process has acquired yum lock' (另一個程序已取得 yum 鎖定) 錯誤
- 問題：'Permission denied / failed to run commands' (許可被拒/無法執行命令) 錯誤
- 問題：'Unable to download payload' (無法下載酬載) 錯誤
- 問題：'unsupported package manager and python version combination' (不支援的套件管理工具和 python 版本組合) 錯誤
- 問題：Patch Manager 不會套用指定的規則來排除某些套件
- 問題：修補失敗，且 Patch Manager 報告 TLS 的伺服器名稱指示延伸項目無法使用
- 問題：Patch Manager 報告 'No more mirrors to try' (沒有更多鏡像可以嘗試)
- 問題：修補失敗並顯示訊息 "Error code returned from curl is 23"
- 問題：修補失敗並顯示訊息 "Error unpacking rpm package..."
- 問題：在上傳庫存時，修補失敗並出現「遇到服務端錯誤」
- 問題：修補失敗並顯示訊息 "Errors were encountered while downloading packages"
- 問題：修補失敗並顯示訊息 "The following signatures couldn't be verified because the public key is not available"
- 問題：修補失敗並顯示訊息 "NoMoreMirrorsRepoError"
- 問題：修補失敗並顯示訊息 "Unable to download payload"
- 問題：修補失敗並顯示訊息 "install errors: dpkg: error: dpkg frontend is locked by another process"
- 問題：在 Ubuntu Server 上的修補失敗，並顯示 "dpkg was interrupted" 錯誤
- 問題：套件管理工具公用程式無法解決套件相依性問題

問題：'No such file or directory' (沒有該檔案或目錄) 錯誤

問題：當您執行 AWS-RunPatchBaseline 時，修補會失敗，並顯示下列其中一個錯誤。

```
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：要執行 AWS-RunPatchBaseline 的兩個命令在同一受管節點上同時執行。這將建立一個競爭條件，導致無法建立或正常存取臨時 `file patch-baseline-operations*`。

原因 2：/var 目錄下保留的儲存空間不足。

解決方案 1：確定沒有維護時段具有兩個或更多 Run Command 任務 (以相同優先順序執行 AWS-RunPatchBaseline 的任務和在相同目標 ID 上執行的任務)。如果是這種情況，請重新排序優先順序。 Run Command 是 中的工具 AWS Systems Manager。

解決方案 2：確保一次只有一個維護時段正在執行 Run Command 任務，這些任務在相同目標依相同排程使用 AWS-RunPatchBaseline。若發生此情形，請變更排程。

解決方案 3：確保只有一個State Manager關聯AWS-RunPatchBaseline以相同的排程執行，並以相同的受管節點為目標。 State Manager 是 中的工具 AWS Systems Manager。

解決方案 4：在 /var 目錄下為更新套件釋放足夠的儲存空間。

問題：'another process has acquired yum lock' (另一個程序已取得 yum 鎖定) 錯誤

問題：當您執行 AWS-RunPatchBaseline 時，修補會失敗，並顯示下列錯誤。

```
12/20/2019 21:41:48 root [INFO]: another process has acquired yum lock, waiting 2 s and
 retry.
```

原因：AWS-RunPatchBaseline 文件已經開始在另一個操作中執行的受管節點上執行，並且已經取得套件管理工具 yum 程序。

解決方案：確保沒有 State Manager 關聯、維護時段任務或依排程執行 AWS-RunPatchBaseline 的其他組態大約在同一時間針對同一個受管節點。

問題：'Permission denied / failed to run commands' (許可被拒/無法執行命令) 錯誤

問題：當您執行 AWS-RunPatchBaseline 時，修補會失敗，並顯示下列錯誤。

```
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/ 可能掛載了 noexec 許可。這是一個問題，因為 SSM Agent 將酬載指令碼下載至 /var/lib/amazon/ssm 並從該位置執行指令碼。

解決方案：請確定您已將專屬分割區設定為 /var/log/amazon 和 /var/lib/amazon，並且這些分割區掛載了 exec 許可。

問題：'Unable to download payload' (無法下載酬載) 錯誤

問題：當您執行 AWS-RunPatchBaseline 時，修補會失敗，並顯示下列錯誤。

```
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：受管節點沒有存取指定 Amazon Simple Storage Service (Amazon S3) 儲存貯體的必要許可。

解決方案：更新您的網路組態，讓 S3 端點可以連線。如需詳細資訊，請參閱 SSM Agent 與 AWS 受管 S3 儲存貯體通訊 中的對 Patch Manager S3 儲存貯體進行必要存取的相關資訊。

問題：'unsupported package manager and python version combination' (不支援的套件管理工具和 python 版本組合) 錯誤

問題：當您執行 AWS-RunPatchBaseline 時，修補會失敗，並顯示下列錯誤。

```
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：Python3 的支援版本未安裝在 Debian Server、Raspberry Pi OS 或 Ubuntu Server 執行個體上。

解決方案：在伺服器上安裝 python3 (3.0 - 3.10) 的支援版本，這是 Debian Server、Raspberry Pi OS 和 Ubuntu Server 受管節點必需的。

問題：Patch Manager 不會套用指定的規則來排除某些套件

問題：您已嘗試透過在 `/etc/yum.conf` 檔案中指定套件以排除某些套件，格式為 `exclude=`*`package-name`*，但在 Patch Manager `Install` 操作期間，不會排除這些套件。

原因：Patch Manager 未包含 `/etc/yum.conf` 檔案中指定的排除項目。

解決方案：若要排除特定套件，請建立自訂修補基準，並建立規則以排除您不想安裝的套件。

問題：修補失敗，且 Patch Manager 報告 TLS 的伺服器名稱指示延伸項目無法使用

問題：修補操作會發出下列訊息。

```
/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：此訊息並不表示錯誤。相反，這是一個警告，即與作業系統分發的舊版 Python 不支援 TLS 伺服器名稱指示。Systems Manager 修補程式承載指令碼會在連線至支援 SNI AWS APIs 時發出此警告。

解決方案：若要在報告此訊息時對任何修補故障進行故障診斷，請檢閱 `stdout` 和 `stderr` 檔案的內容。如果尚未將修補基準設定為將這些檔案儲存在 S3 儲存貯體或 Amazon CloudWatch Logs 中，則您可以在 Linux 受管節點的下列位置找到這些檔案。

`/var/lib/amazon/ssm/`*`instance-id`*`/document/orchestration/`*`Run-Command-execution-id`*`/awsrunShellScript/PatchLinux`

問題：Patch Manager 報告 'No more mirrors to try' (沒有更多鏡像可以嘗試)

問題：修補操作會發出下列訊息。

```
[Errno 256] No more mirrors to try.
```

原因：受管節點上設定的儲存庫無法正常運作。可能的原因包括：

- yum 快取已毀損。
- 由於網路相關問題，無法連線儲存庫 URL。

解決方案：Patch Manager 會使用受管節點的預設套件管理工具來執行修補操作。再次檢查儲存庫是否已設定並正常運作。

問題：修補失敗並顯示訊息 "Error code returned from curl is 23"

問題：使用 AWS-RunPatchBaseline 的修補作業失敗，並出現類似以下內容的錯誤：

```
05/01/2025 17:04:30 root [ERROR]: Error code returned from curl is 23
```

原因：您的系統上使用的 curl 工具缺少寫入文件系統所需的許可。如果套件管理工具的預設 curl 工具被其他版本取代，例如使用 snap 安裝的版本，就會發生這種情況。

解決方案：如果在安裝不同版本時解除安裝了套件管理工具提供的 curl 版本，請重新安裝。

如果您需要保留安裝的多個 curl 版本，請確保與套件管理工具相關聯的版本位於 PATH 變數中列出的第一個目錄中。您可以透過執行 echo $PATH 命令來檢查，以查看檢查系統上的目錄是否有可執行檔的目錄順序。

問題：修補失敗並顯示訊息 "Error unpacking rpm package..."

問題：修補作業失敗，並出現類似下列內容的錯誤：

```
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：當特定套件存在於多個套件安裝程式 (例如 pip 和 yum 或 dnf) 時，使用預設套件管理工具時可能會發生衝突。

urllib3 套件 (可在 pip、yum 和 dnf 中找到) 就是一個常見的範例。

原因 2：python-urllib3 套件已損毀。如果 rpm 套件在先前由 yum 或 dnf 安裝，並且之後由 pip 安裝或更新了套件檔案，就可能會發生這種情況。

解決方案：透過執行 sudo pip uninstall urllib3 命令從 pip 中移除 python-urllib3 套件，僅將套件保留在預設套件管理工具 (yum 或 dnf) 中。

問題：在上傳庫存時，修補失敗並出現「遇到服務端錯誤」

問題：執行AWS-RunPatchBaseline文件時，您會收到下列錯誤訊息：

```
Encounter service side error when uploading the inventory
```

原因：兩個要執行的命令同時AWS-RunPatchBaseline在相同的受管節點上執行。這會在修補操作期間初始化 boto3 用戶端時建立競爭條件。

解決方案：確保沒有 State Manager 關聯、維護時段任務或依排程執行 AWS-RunPatchBaseline 的其他組態大約在同一時間針對同一個受管節點。

問題：修補失敗並顯示訊息 "Errors were encountered while downloading packages"

問題：在修補期間，收到類似以下內容的錯誤：

```
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：在受管節點上的記憶體不足時，可能會發生此錯誤。

解決方案：設定 swap 記憶體，或將執行個體升級為其他類型以增加記憶體。然後啟動新的修補作業。

問題：修補失敗並顯示訊息 "The following signatures couldn't be verified because the public key is not available"

問題：在 Ubuntu Server 上的修補失敗，並出現類似以下內容的錯誤：

```
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 Privacy Guard (GPG) 密鑰已過期或遺失。

解決方案：重新整理 GPG 金鑰，或者，再次新增金鑰。

例如，根據前面顯示的錯誤，我們看到 467B942D3A79BD29 金鑰遺失且必須新增。若要這麼做，請執行任一下列命令：

```
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，若要重新整理所有金鑰，請執行以下命令：

```
sudo apt-key adv --keyserver hkps://keyserver.ubuntu.com --refresh-keys
```

如果在此之後發生錯誤，建議您將問題報告給維護儲存庫的組織。在修補可用之前，您可以編輯 `/etc/apt/sources.list` 檔案，以便在修補期間省略此儲存庫。

若要這麼做，請開啟要編輯的 `sources.list` 檔案，找到儲存庫所在的行，然後在該行的開頭插入一個 # 字元以將其註解掉。然後儲存並關閉檔案。

問題：修補失敗並顯示訊息 "NoMoreMirrorsRepoError"

問題：收到類似以下內容的錯誤：

```
NoMoreMirrorsRepoError: failure: repodata/repomd.xml from pgdg94: [Errno 256] No more
 mirrors to try.
```

原因：來源儲存庫中存在錯誤。

解決方案：建議您將問題報告給維護該儲存庫的組織。在錯誤得到修正之前，您可以在作業系統層級停用此儲存庫。若要這麼做，請執行以下命令，並以您的儲存庫名稱取代 *repo-name* 的值：

```
yum-config-manager --disable repo-name
```

以下是範例。

```
yum-config-manager --disable pgdg94
```

執行此命令之後，請另執行修補操作。

問題：修補失敗並顯示訊息 "Unable to download payload"

問題：收到類似以下內容的錯誤：

```
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：受管節點組態包含錯誤或不完整。

解決方案：請確保受管節點的設定如下：

- 安全群組中的有傳出 TCP 443 規則。
- NACL 中有傳出 TCP 443 規則。
- NACL 中有傳入 TCP 1024-65535 規則。
- 路由表中有 NAT/IGW 以提供與 S3 端點的連線。如果執行個體無法存取網際網路，請提供與 S3 端點的連線。若要這麼做，請在相關 VPC 中新增 S3 閘道端點，並將其與受管節點的路由表整合。

問題：修補失敗並顯示訊息 "install errors: dpkg: error: dpkg frontend is locked by another process"

問題：修補失敗並出現類似以下內容的錯誤：

```
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：套件管理工具已經在作業系統層級的受管節點上執行另一個程序。如果此另一個程序需要很長時間才能完成，則 Patch Manager 修補作業可能會逾時並失敗。

解決方案：在使用套件管理工具的另一個程序完成之後，執行新的修補作業。

問題：在 Ubuntu Server 上的修補失敗，並顯示 "dpkg was interrupted" 錯誤

問題：在 Ubuntu Server 上，修補失敗並出現類似以下內容的錯誤：

```
E: dpkg was interrupted, you must manually run
'dpkg --configure -a' to correct the problem.
```

原因：一個或多個套件設定錯誤。

解決方案：執行以下步驟：

1. 一次執行一條下列命令，以查看哪些套件受到影響以及每個套件的問題：

   ```
   sudo apt-get check
   ```

```
sudo dpkg -C
```

```
dpkg-query -W -f='${db:Status-Abbrev} ${binary:Package}\n' | grep -E ^.[^nci]
```

2. 執行以下命令以修正有問題的套件：

```
sudo dpkg --configure -a
```

3. 如果上一個命令未能完全解決問題，請執行以下命令：

```
sudo apt --fix-broken install
```

問題：套件管理工具公用程式無法解決套件相依性問題

問題：受管節點上的原生套件管理工具無法解決套件相依性問題，且修補失敗。以下錯誤訊息範例指出在使用 yum 作為套件管理工具的作業系統上發生的這種失敗類型。

```
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：Linux 作業系統上，Patch Manager 使用機器上的原生套件管理工具執行修補作業。例如 yum、dnf、apt 和 zypper。應用程式會根據需要自動偵測、安裝、更新或移除相依套件。但是，某些情況可能會導致套件管理工具無法完成相依性作業，例如：

- 作業系統上設定了多個衝突的儲存庫。
- 由於網路相關的問題，無法存取遠端儲存庫 URL。
- 在儲存庫中找到適用於錯誤架構的套件。

解決方案：修補程式可能會因為各種原因造成的相依性問題而失敗。因此，我們建議您聯絡 AWS 支援以協助故障診斷。

在 Windows Server 上執行 **AWS-RunPatchBaseline** 時發生錯誤

主題

- 問題：產品系列/產品配對不相符
- 問題：AWS-RunPatchBaseline 輸出會傳回 HRESULT (Windows Server)

- 問題：受管節點無法存取 Windows 更新目錄或 WSUS
- 問題：PatchBaselineOperations PowerShell 模組無法下載
- 問題：缺少修補程式

問題：產品系列/產品配對不相符

問題：當您在 Systems Manager 主控台中建立修補基準時，您會指定產品系列和產品。例如，您可能選擇：

- 產品系列：`Office`

  產品：`Office 2016`

原因：如果您嘗試以不符合的產品系列/產品配對來建立修補基準，則會出現錯誤訊息。以下為此狀況發生的原因：

- 您已選擇有效的產品系列和產品配對，但將該產品系列選擇移除了。
- 您選擇了 Obsolete or mismatched options (已淘汰或不符合的選項) 子清單中的產品，而非 Available and matching options (可用和符合的選項) 子清單中的產品。

  產品淘汰或不相符選項子清單中的項目可能已透過 SDK 或 AWS Command Line Interface (AWS CLI) `create-patch-baseline`命令錯誤輸入。這可能代表著導入了拼寫錯誤或產品被指派給錯誤的產品系列。如果已為之前的修補基準指定，但沒有 Microsoft 提供的修補程式，則產品也會包含在 Obsolete or mismatched options (已淘汰或不符合的選項) 子清單中。

解決方案：為了避免在主控台中出現此問題，請一律從 Currently available options (目前可用的選項) 子清單中進行選擇。

您也可以使用 AWS CLI 或 `DescribePatchProperties` API 命令中的 `describe-patch-properties` 命令以檢視擁有可用修補程式的產品。

問題：**`AWS-RunPatchBaseline`** 輸出會傳回 **`HRESULT`** (Windows Server)

問題：您收到以下錯誤：

```
----------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：這個輸出表示原生的 Windows 更新 API 無法執行修補操作。

解決方案：檢查以下 microsoft.com 主題中的 HResult 程式碼以識別解決錯誤的故障診斷步驟：

- 按元件列出的 Windows Update 錯誤代碼
- Windows Update 常見錯誤和緩解措施

問題：受管節點無法存取 Windows 更新目錄或 WSUS

問題：您收到以下錯誤：

```
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：這個錯誤可能與 Windows 更新元件相關，或缺乏至 Windows 更新目錄或 Windows 伺服器更新服務 (WSUS) 的連線。

解決方案：確認您的受管節點已透過網際網路閘道、NAT 閘道或 NAT 執行個體連線至 Microsoft 更新目錄。如果使用 WSUS，請確認受管節點已連線至您環境中的 WSUS 伺服器。如果連線能夠提供給預定的目的地，請查看 Microsoft 文件，了解 `HResult 0x80072EE2` 的其他潛在原因。這可能表示作業系統層級有問題。

問題：PatchBaselineOperations PowerShell 模組無法下載

問題：您收到以下錯誤：

```
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
```

解決方案：檢查受管節點的連線和 Amazon Simple Storage Service (Amazon S3) 的許可。受管節點的 AWS Identity and Access Management (IAM) 角色必須使用 中引用的最低許可SSM Agent 與 AWS 受管 S3 儲存貯體通訊。節點必須透過 Amazon Simple Storage Service (Amazon S3) 閘道端點、NAT 閘道或網際網路閘道與 Amazon Simple Storage Service (Amazon S3) 端點進行通訊。如需 (SSM Agent) 的 AWS Systems Manager SSM Agent VPC 端點需求的詳細資訊，請參閱 使用適用於 Systems Manager 的 VPC 端點來改善 EC2 執行個體的安全性。

問題：缺少修補程式

問題：AWS-RunPatchbaseline 成功完成，但有一些缺少的修補程式。

以下是一些常見原因及其解決方案。

原因 1：基準無效。

解決方案 1：如需檢查這是否為原因，請使用下列處理程序。

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Run Command。
3. 選取 Command history (命令歷史記錄) 標籤，然後選取您要檢查其基準的命令。
4. 選取缺少修補程式的受管節點。
5. 選擇 Step 1 - Output (步驟 1 – 輸出) 並尋找 BaselineId 值。
6. 檢查指派的修補基準組態，即修補基準的作業系統、產品名稱、分類和嚴重性。

7. 前往 Microsoft 更新目錄。
8. 搜尋 Microsoft 知識庫 (KB) 文章 ID (例如 KB3216916)。
9. 確認 Product (產品) 下的該值是否與受管節點的值相符，並選取相應的 Title (標題)。新的 Update Details (更新詳細資訊) 時段將會開啟。
10. 在 Overview (概觀) 標籤上，classification (分類) 和 MSRC severity (MSRC 嚴重性) 必須符合您先前找到的修補基準組態。

原因 2：修補程式已被取代。

解決方案 2：如需檢查這是否為真實情形，請使用下列處理程序。

1. 前往 Microsoft 更新目錄。
2. 搜尋 Microsoft 知識庫 (KB) 文章 ID (例如 KB3216916)。
3. 確認 Product (產品) 下的該值是否與受管節點的值相符，並選取相應的 Title (標題)。新的 Update Details (更新詳細資訊) 時段將會開啟。
4. 前往 Package Details (套件詳細資訊) 標籤。尋找 This update has been replaced by the following updates: (此更新已由以下更新取代) 標頭下的項目。

原因 3：相同的修補程式可能有不同的 KB 數，因為 WSUS 和 Windows 線上更新會作為獨立的 Microsoft 發行通道處理。

解決方案 3：檢查修補程式的資格。如果 WSUS 下無法使用套件，請安裝 OS Build 14393.3115。如果套件適用於所有作業系統組建，請安裝作業系統組建 18362.1256 和 18363.1256。

使用 AWS 支援 自動化 Runbook

AWS 支援 提供兩個 Automation Runbook，可用來疑難排解與修補相關的特定問題。

- `AWSSupport-TroubleshootWindowsUpdate` – `AWSSupport-TroubleshootWindowsUpdate` Runbook 用於識別可能使 Amazon Elastic Compute Cloud (Amazon EC2) Windows Server執行個體Windows Server更新失敗的問題。
- `AWSSupport-TroubleshootPatchManagerLinux` – `AWSSupport-TroubleshootPatchManagerLinux` Runbook 會使用 對 Linux 受管節點上可能導致修補程式失敗的常見問題進行故障診斷Patch Manager。此 Runbook 的主要目標是識別修補程式命令失敗根本原因，並建議修補計畫。

> **Note**
>
> 執行 Automation Runbook 需要付費。如需詳細資訊，請參閱AWS Systems Manager 自動化定價。

聯絡 AWS 支援

如果您在本節或 AWS re:Post 的 Systems Manager 問題中找不到疑難排解解決方案，且您擁有開發人員、商業或企業 支援 計畫，則您可以在 AWS 支援 建立技術支援案例。

聯絡之前 支援，請收集下列項目：

- SSM 代理程式日誌
- Run Command 命令 ID、維護時段 ID 或 Automation 執行 ID
- 對於 Windows Server 受管節點，也會收集下列項目：
  - 如 如何安裝修補程式 的 Windows (Windows) 標籤中所述的 `%PROGRAMDATA%\Amazon\PatchBaselineOperations\Logs`
  - Windows 更新日誌：對於 Windows Server 2012 R2 和更舊版本，請使用 `%windir%/WindowsUpdate.log`。對於 Windows Server 2016 和更新版本，請在使用 `%windir%/WindowsUpdate.log` 之前首先執行 PowerShell 命令 `Get-WindowsUpdateLog`
- 對於 Linux 受管節點，也會收集下列項目：
  - 目錄 `/var/lib/amazon/ssm/instance-id/document/orchestration/Run-Command-execution-id/awsrunShellScript/PatchLinux` 的內容

AWS Systems Manager Run Command

使用 中的Run Command工具 AWS Systems Manager，您可以遠端安全地管理受管節點的組態。受管節點是混合多雲端環境中針對 Systems Manager 設定的任何 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體或非 EC2 機器。Run Command 可讓您自動執行常見的管理任務，並進行大規模的單次組態變更。您可以從 Run Command AWS Management Console、 AWS Command Line Interface (AWS CLI) AWS Tools for Windows PowerShell或 AWS SDKs使用 。 Run Command免費提供。若要開始使用 Run Command，請開啟 Systems Manager 主控台。在導覽窗格中，選擇 Run Command。

管理員使用 Run Command 來安裝或引導應用程式、建置部署管道、在執行個體從 Auto Scaling 群組移除時擷取日誌檔案，將執行個體加入 Windows 網域等。

由於支援 API 的系統的分散式性質，Run Command API 遵循最終一致性模式。這表示您執行的會影響資源的 API 命令的結果，可能不會立即對您執行的後續所有命令可見。在執行緊接先前 API 命令的 API 命令時，應謹記這一點。

開始使用

下表提供的資訊，可協助您開始使用 Run Command。

主題	詳細資訊
設定 AWS Systems Manager的受管節點	確認您已完成混合多雲端環境中 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體和非 EC2 機器的設定要求。
使用 Systems Manager 在混合多雲端環境中管理節點	（選用） 向 註冊現場部署伺服器和 VMs AWS ，以便您可以使用 管理它們Run Command。
the section called “使用 Systems Manager 管理邊緣裝置”	(選用) 設定邊緣裝置，以便您可以使用 Run Command 管理這些裝置。
在受管節點上執行命令	了解如何執行透過使用 AWS Management Console以一或多個受管節點為目標的命令。
Run Command演練	了解如何使用 Tools for Windows PowerShell 或執行命令 AWS CLI。

支援 EventBridge

在 Amazon EventBridge 規則中，此 Systems Manager 工具支援為事件類型和目標類型。如需詳細資訊，請參閱 使用 Amazon EventBridge 監控 Systems Manager 及 參考：Systems Manager 的 Amazon EventBridge 事件模式和類型。

詳細資訊

- 在 EC2 執行個體上遠端 Run Command (10 分鐘教學課程)
- 《Amazon Web Services 一般參考》中的 Systems Manager 服務配額一節
- AWS Systems Manager API 參考

主題

- 設定 Run Command
- 在受管節點上執行命令
- 使用命令中的結束程式碼
- 了解命令狀態
- Run Command演練
- 故障診斷 Systems Manager 執行命令

設定 Run Command

使用 中的工具Run Command來管理節點之前 AWS Systems Manager，請為將執行命令的任何使用者設定 AWS Identity and Access Management (IAM) 政策。如果針對 IAM 政策中的 SendCommand 動作使用任何全域條件索引鍵，則必須包含 aws:ViaAWSService 條件索引鍵並將布林值設定為 true。以下是範例。

```
{
"Version": "2012-10-17",
"Statement": [
    {
    "Effect": "Allow",
    "Action": ["ssm:SendCommand"],
    "Resource": ["arn:aws:ssm:region:account:document/YourDocument"],
    "Condition": {
        "StringEquals": {
        "aws:SourceVpce": ["vpce-example1234"]
        }
    }
    },
    {
    "Effect": "Allow",
    "Action": ["ssm:SendCommand"],
    "Resource": ["arn:aws:ssm:region:account:document/YourDocument"],
    "Condition": {
        "Bool": {"aws:ViaAWSService": "true"}
        }
    }
  ]
}
```

您也必須為 Systems Manager 設定節點。如需詳細資訊，請參閱設定 AWS Systems Manager的受管節點。

我們建議您完成下列選擇性設定任務，以將受管節點安全狀態和日常的管理降到最低。

使用 Amazon EventBridge 監控命令執行

您可以使用 EventBridge 來記錄命令執行狀態變更。您可以建立規則，在有狀態轉換或有轉移到一或多個與您相關的狀態時執行。您也可以指定在 EventBridge 事件發生時，將 Run Command 做為目標動作。如需詳細資訊，請參閱為 Systems Manager 事件設定 EventBridge。

使用 Amazon CloudWatch Logs 來監控命令執行

您可以設定 Run Command 來定期將所有命令輸出和錯誤日誌傳送至 Amazon CloudWatch 日誌群組。您可以以幾乎即時的方式監控這些輸出日誌、搜尋特定字詞、數值或模式，並根據搜尋建立警示。如需詳細資訊，請參閱設定 Run Command 的 Amazon CloudWatch Logs。

限制 Run Command 對特定受管節點的存取

您可以使用 AWS Identity and Access Management (IAM) 限制使用者在受管節點上執行命令的能力。具體來說，您可以建立 IAM 政策，其條件是使用者只能在標記了特定標籤的受管節點上執行命令。如需詳細資訊，請參閱根據標籤限制 Run Command 存取。

根據標籤限制 Run Command 存取

本節描述了如何透過在 IAM 政策中指定標籤條件，限制使用者在受管節點上執行命令的能力。受管節點包含針對 Systems Manager 設定之混合多雲端環境中的 Amazon EC2 執行個體和非 EC2 節點。雖然資訊未明確顯示，但您也可以限制對受管 AWS IoT Greengrass 核心裝置的存取。若要開始使用，您必須標記您的 AWS IoT Greengrass 裝置。如需詳細資訊，請參閱《AWS IoT Greengrass Version 2 開發人員指南》中的標記您的 AWS IoT Greengrass Version 2 資源。

您可以透過建立 IAM 政策 (其中包含條件，限制使用者只能在具有特定標籤的節點上執行命令)，來限制只針對特定受管節點執行命令。在下列範例中，使用者可以在任何節點 (`Resource: arn:aws:ec2:*:*:instance/*`) 使用任何 SSM 文件 (`Resource: arn:aws:ssm:*:*:document/*`) 來使用 Run Command (`Effect: Allow, Action: ssm:SendCommand`)，條件為節點是 Finance WebServer (`ssm:resourceTag/Finance: WebServer`)。如果使用者將命令傳送至未加上標記或有 `Finance: WebServer` 以外任何標記的節點：執行結果會顯示 `AccessDenied`。

```
{
   "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"
               ]
            }
         }
      }
   ]
}
```

您可以建立 IAM 政策，可讓使用者在以多個標籤所標記的受管節點上執行命令。以下政策可讓使用者在擁有兩個標籤的受管理節點上執行命令。如果使用者將命令傳送至未以那兩個標記所標記的節點，執行結果會顯示 AccessDenied。

```
{
   "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":"*"
      }
   ]
}
```

您也可以建立 IAM 政策，其可讓使用者在多組加上標籤的受管節點上執行命令。以下範例政策可讓使用者在一組或兩組含標記的節點上執行命令。

```
{
   "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":"*"
      }
```

```
    ]
}
```

如需有關建立 IAM 政策的詳細資訊，請參閱《IAM 使用者指南》中的受管政策和內嵌政策。如需標記受管節點的詳細資訊，請參閱《AWS Resource Groups 使用者指南》中的標籤編輯器。

在受管節點上執行命令

本節包含如何從 AWS Systems Manager 主控台傳送命令至受管節點的相關資訊。這區段還包含如何取消命令的相關資訊。

請注意，如果您的節點使用 var 目錄的noexec掛載選項設定， Run Command 將無法成功執行命令。

> **Important**
>
> 當您使用 Run Command 發出命令時，請勿包含格式為純文字的敏感資訊，例如密碼、組態資料或其他密碼。您帳戶中的所有 Systems Manager API 活動都會記錄於 S3 儲存貯體中的 AWS CloudTrail 日誌。這意味著任何具有權存取該 S3 儲存貯體的使用者都可以查看這些密碼的純文字值。因此，建議您建立並使用 `SecureString` 參數來加密您在 Systems Manager 操作中使用的敏感資料。
>
> 如需詳細資訊，請參閱使用 IAM 政策限制對 Parameter Store 參數的存取。

執行歷史記錄保留

每個命令的歷史記錄最多可供使用 30 天。此外，您可以在 Amazon Simple Storage Service 中存放所有日誌檔案的副本，或在 中擁有所有 API 呼叫的稽核線索 AWS CloudTrail。

相關資訊

如需使用其他工具傳送命令的資訊，請參閱下列主題：

- 逐步解說： AWS Tools for Windows PowerShell 搭配 使用 Run Command 或 AWS Systems ManagerAWS Tools for PowerShell Cmdlet 參考 區段中的範例。
- 逐步解說： AWS CLI 搭配 使用 Run Command 或 SSM CLI 參考中的範例

目錄

- 從主控台執行命令

- 使用特定文件版本執行命令
- 大規模執行命令
- 取消命令

從主控台執行命令

您可以從 使用 中的Run Command工具 AWS Systems Manager AWS Management Console 來設定受管節點，而無需登入。本主題包含範例，說明如何使用 Run Command 在受管節點上更新 SSM Agent。

開始之前

使用 Run Command 傳送命令之前，確認您的受管節點滿足所有 Systems Manager 設定要求。

使用 Run Command 傳送命令

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Run Command。
3. 選擇執行命令。
4. 在 Command document (命令文件) 清單中，選擇 Systems Manager 文件。
5. 在 Command parameters (命令參數) 區段，指定所需的參數值。
6. 在 Targets (目標) 區段中，透過手動指定標籤、選取執行個體或邊緣裝置，或指定資源群組，選擇您要執行這項操作的受管節點。

   > **Tip**
   >
   > 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

7. 對於其他參數：
   - 在 Comment (註解) 中，輸入此命令的相關資訊。
   - 在逾時 (秒) 中，指定在命令執行全面失敗之前，系統要等候的秒數。
8. 對於 Rate control (速率控制)：
   - 在並行中，指定可同時執行命令的受管節點數目或百分比。

> **Note**
>
> 如果透過指定套用至受管節點的標籤或指定 AWS 資源群組選取了目標，且您不確定會以多少個受管節點為目標，則透過指定百分比限制可以同時執行文件之目標的數量。

- 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比之節點上的命令失敗之後，停止在其他受管節點上執行命令。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止傳送命令。仍在處理命令的受管節點也可能會傳送錯誤。

9. (選用) 選擇要套用至您的監控命令的 CloudWatch 警示。若要將 CloudWatch 警示附加至您的命令，執行命令的 IAM 主體必須具備 `iam:createServiceLinkedRole` 動作的許可。如需有關 CloudWatch 警示的詳細資訊，請參閱使用 Amazon CloudWatch 警示。請注意，如果您的警示啟用，則不會執行任何待處理命令叫用。
10. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取將命令輸出寫入至 S3 儲存貯體方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

> **Note**
>
> 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給執行個體之執行個體設定檔 (適用於 EC2 執行個體) 或 IAM 服務角色 (啟用混合模式的機器) 的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確定與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

11. 在SNS 通知區段中，如果您要傳送有關命令執行狀態的通知，請選取啟用 SNS 通知核取方塊。

    如需為 Run Command 設定 Amazon SNS 通知的詳細資訊，請參閱 使用 Amazon SNS 通知監控 Systems Manager 狀態變更。
12. 選擇執行。

如需取消命令的詳細資訊，請參閱 the section called “取消命令”。

重新執行命令

Systems Manager 包含兩個選項，可協助您從 Systems Manager 主控台的 Run Command (執行命令) 頁面重新執行命令。

- Rerun (重新執行)：此按鈕可讓您執行相同的命令，而不對其進行變更。
- Copy to new (複製到新命令)：此按鈕會將一個命令的設定複製到新命令，並讓您選擇在執行之前編輯這些設定。

重新執行命令

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Run Command。
3. 選擇要重新執行的命令。從命令詳細資訊頁面執行命令之後，您可以立即重新執行命令。或者，您也可以從 Command history (命令歷程記錄) 索引標籤選擇先前執行的命令。
4. 選擇 Rerun (重新執行) 以在不進行變更的情況下執行相同的命令，或選擇 Copy to new (複製到新的) 以在執行命令之前先行編輯命令設定。

使用特定文件版本執行命令

您可以使用文件版本參數來在命令執行時，指定要使用的 AWS Systems Manager 文件版本。您可為此參數指定下列其中一個選項：

- $DEFAULT
- $LATEST
- 版本編號

請執行以下處理程序來使用文件版本參數執行命令。

Linux

在本機 Linux 機器 AWS CLI 上使用 執行命令

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。
2. 列出所有可用的文件

   此命令會根據 AWS Identity and Access Management (IAM) 許可列出您帳戶可用的所有文件。

```
aws ssm list-documents
```

3. 執行下列命令來檢視不同版本的文件。將####取代為自己的資訊。

```
aws ssm list-document-versions \
    --name "document name"
```

4. 執行下列命令來執行使用 SSM 文件版本的命令。將每個########取代為您自己的資訊。

```
aws ssm send-command \
    --document-name "AWS-RunShellScript" \
    --parameters commands="echo Hello" \
    --instance-ids instance-ID \
    --document-version '$LATEST'
```

Windows

在本機 Windows 機器 AWS CLI 上使用 執行命令

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

2. 列出所有可用的文件

   此命令會根據 AWS Identity and Access Management (IAM) 許可列出您帳戶可用的所有文件。

```
aws ssm list-documents
```

3. 執行下列命令來檢視不同版本的文件。將####取代為自己的資訊。

```
aws ssm list-document-versions ^
    --name "document name"
```

4. 執行下列命令來執行使用 SSM 文件版本的命令。將每個########取代為您自己的資訊。

```
aws ssm send-command ^
    --document-name "AWS-RunShellScript" ^
    --parameters commands="echo Hello" ^
    --instance-ids instance-ID ^
```

```
    --document-version "$LATEST"
```

PowerShell

使用 Tools for PowerShell 執行命令

1. 如果您尚未安裝並設定 AWS Tools for PowerShell (Tools for Windows PowerShell)。

   如需相關資訊，請參閱安裝 AWS Tools for PowerShell。
2. 列出所有可用的文件

   此命令會根據 AWS Identity and Access Management (IAM) 許可列出您帳戶可用的所有文件。

   ```
   Get-SSMDocumentList
   ```
3. 執行下列命令來檢視不同版本的文件。將####取代為自己的資訊。

   ```
   Get-SSMDocumentVersionList `
       -Name "document name"
   ```
4. 執行下列命令來執行使用 SSM 文件版本的命令。將每個########取代為您自己的資訊。

   ```
   Send-SSMCommand `
       -DocumentName "AWS-RunShellScript" `
       -Parameter @{commands = "echo helloWorld"} `
       -InstanceIds "instance-ID" `
       -DocumentVersion $LATEST
   ```

大規模執行命令

您可以使用 中的工具 Run Command AWS Systems Manager，在受管節點的機群上執行命令targets。targets 參數接受以您為受管節點指定的標記為基礎的 Key,Value 組。當您執行命令時，系統會尋找並嘗試在符合指定標記的所有受管節點上執行命令。如需標記受管執行個體的詳細資訊，請參閱《標記資源使用者指南》中的標記您的 AWS資源。 AWS 如需有關標記受管 IoT 裝置的資訊，請參閱《 AWS IoT Greengrass Version 2 開發人員指南》中的標記您的 AWS IoT Greengrass Version 2 資源。

您也可以使用 targets 參數設為目標的特定受管節點 ID，清單中所述的下一個部分。

若要在數百或者數千個受管節點間控制命令執行，Run Command 還包含可限制可以同時處理請求之節點數的參數，以及取消命令之前命令可以擲回的錯誤數。

目錄

- 以多個受管節點為目標
- 使用速率控制

以多個受管節點為目標

您可以透過指定標籤、 AWS 資源群組名稱或受管節點 ID，執行命令和目標受管節點。

下列範例顯示Run Command從 AWS Command Line Interface () 使用 時的命令格式AWS CLI 。將每個########取代為您自己的資訊。使用 [...] 將此區段中的範本命令截斷。

範例 1：以標籤為目標

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 ^
    [...]
```

範例 2：依名稱鎖定 AWS 資源群組

您可以為每個命令指定最多一個資源群組名稱。建立資源群組時，我們建議包括在分組條件中包含 AWS::SSM:ManagedInstance 和 AWS::EC2::Instance 作為資源類型。

> **Note**
>
> 若要傳送以資源群組為目標的命令，您必須已授予 AWS Identity and Access Management (IAM) 許可，才能列出或檢視屬於該群組的資源。如需詳細資訊，請參閱《AWS Resource Groups 使用者指南》中的設定許可。

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 ^
    [...]
```

範例 3：依 AWS 資源類型鎖定資源群組

您可以為每個命令指定最多五個資源群組類型。建立資源群組時，我們建議包括在分組條件中包含 `AWS::SSM:ManagedInstance` 和 `AWS::EC2::Instance` 作為資源類型。

> **Note**
>
> 為了傳送以資源群組為目標的命令，您必須獲得 IAM 許可來列出或檢視屬於該群組的資源。如需詳細資訊，請參閱《AWS Resource Groups 使用者指南》中的設定許可。

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 ^
    [...]
```

範例 4：以執行個體 ID 為目標

下列範例示範如何將 instanceids 金鑰與 targets 參數搭配使用才能鎖定受管節點。您可以使用此金鑰鎖定受管 AWS IoT Greengrass 核心裝置，因為每個裝置都會指派 mi-*ID_number*。您可以在 中檢視裝置 IDsFleet Manager，這是 中的工具 AWS Systems Manager。

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 ^
    [...]
```

如果您使用名為 Environment 的 Key 以及 Development、Test 和 Pre-production 的 Values 以及 Production，為不同環境的受管節點加上標記，則您可以使用 targets 參數搭配以下語法，將命令傳送至這些環境中的所有節點之一。

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 ^
    [...]
```

您可以透過新增到 Values 清單，在其他環境中以其他受管節點為目標。使用逗號分隔的獨立項目。

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 ^
    [...]
```

變異：使用多個 Key 條件精簡您的目標

您可以透過包含多個 Key 條件來精簡您命令的目標數。如果您包含多個 Key 條件，系統會將符合所有條件的受管節點視為目標。以下命令為將標上金融部門和資料庫伺服器角色的所有受管節點視為目標。

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 ^
    [...]
```

變異：使用多個 Key 和 Value 條件

在之前的範例中展開，您可以透過在 Values 條件中包含額外的項目，來將多個部門和多個伺服器角色視為目標。

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 ^
    [...]
```

變化：使用多個 Values 條件鎖定標記的受管節點

如果您使用名為 Department 的 Key 以及 Sales 和 Finance 的 Values，為不同環境的受管節點加上標記，則您可以使用 targets 參數搭配以下語法，將命令傳送至這些環境中的所有節點。

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 ^
    [...]
```

您最多可以指定五個索引鍵，每個索引鍵可以指定五個數值。

如果標籤索引鍵 (標籤名稱) 或標籤值包含空格，則將標籤索引鍵或值括在引號中，如下範例所示。

範例：Value 標籤中的空格

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" ^
    [...]
```

範例：tag 索引鍵中的空格和 Value

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" ^
    [...]
```

範例 ：在 Values 清單的單一項目中的空格。

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

您可以控制命令傳送到群組中受管節點的速率，方法是使用並行控制項和錯誤控制項。

主題

- 使用並行控制
- 使用錯誤控制

使用並行控制

您可以使用 max-concurrency 參數 (Run a command (執行命令) 頁面中的 Concurrecy (並行) 選項) 控制同時執行之受管節點的數量。您可以指定絕對數量的受管節點 (例如 **10**) 或目標集的百分比 (例如 **10%**)。佇列系統會將命令傳遞至單一節點並等到系統確認初始叫用，再將該命令傳送至另外兩個節點。系統會以指數方式將命令傳送至更多受管節點，直到系統達到 max-concurrency 值為止。max-concurrency 的預設值為 50。以下範例說明您為 max-concurrency 參數指定數值的方式：

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 ^
    [...]
```

使用錯誤控制

透過使用 `max-errors` 參數(Run a command (執行命令) 頁面中的 Error threshold (錯誤閾值) 欄位)設定錯誤限制，您也可以控制數百個或數千個受管節點的命令執行。參數會指定在系統停止將命令傳送至其他受管節點前允許的錯誤數。您可以指定錯誤的絕對數目 (例如 **10**)，或目標集的百分比 (例如 **10%**)。舉例來說，如果您指定 **3**，系統會在收到第 4 個錯誤時停止傳送命令。如果您指定 **0**，系統會在第一個錯誤結果傳回後停止將命令傳送至其他受管節點。如果您將命令傳送至 50 個受管節點，並將 `max-errors` 設為 **10%**，系統會在收到第六個錯誤時停止將命令傳送至其他節點。

達到 `max-errors` 時，系統會允許完成已在執行命令的呼叫，但一些呼叫也可能會失敗。如果您需要確保不會有超過 `max-errors` 個失敗的呼叫，請將 `max-concurrency` 設定為 **1**，以每次執行一個呼叫。錯誤上限預設值為 0。以下範例說明您為 `max-errors` 參數指定數值的方式：

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

只要該服務會顯示為待定或執行中狀態，您就可以嘗試取消命令。不過，即使命令仍處於其中一個狀態，我們無法保證命令是否會取消且基本程序會停止。

使用主控台來取消命令

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Run Command。
3. 選取您要取消的命令呼叫。
4. 選擇 Cancel command (取消命令)。

使用 取消命令 AWS CLI

執行下列命令。將每個########取代為您自己的資訊。

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"
```

如需取消命令之狀態的詳細資訊，請參閱 了解命令狀態。

使用命令中的結束程式碼

在某些情況下，您可能需要管理如何透過使用結束程式碼來處理命令。

在命令中指定結束程式碼

使用 中的Run Command工具 AWS Systems Manager，您可以指定結束代碼，以判斷如何處理命令。根據預設，在指令碼中執行的最後一個命令的結束程式碼會報告為整個指令碼的結束程式碼。比方說您

有包含三個命令的指令碼。第一個失敗，但後續兩個成功。因為最後一個命令成功，所以執行狀態會報告為 succeeded。

Shell 指令碼

若要在第一次命令故障時讓整個指令碼失效，您可以包含一個 shell 條件陳述式，在有任何命令在最後一個命令之前發生失敗時結束指令碼。請使用以下方法。

```
<command 1>
    if [ $? != 0 ]
    then
        exit <N>
    fi
    <command 2>
    <command 3>
```

在下列範例中，如果第一個命令失敗，整個指令碼就會失敗。

```
cd /test
    if [ $? != 0 ]
    then
        echo "Failed"
        exit 1
    fi
    date
```

PowerShell 指令碼

PowerShell 要求您在指令碼中明確呼叫 exit，Run Command 才能成功擷取結束程式碼。

```
<command 1>
    if ($?) {<do something>}
    else {exit <N>}
    <command 2>
    <command 3>
    exit <N>
```

請見此處範例：

```
cd C:\
```

```
    if ($?) {echo "Success"}
    else {exit 1}
    date
```

執行命令時處理重新啟動

如果您使用 中的Run Command工具 AWS Systems Manager來執行重新啟動受管節點的指令碼，建議您在指令碼中指定結束程式碼。如果您嘗試使用其他機制重新啟動節點，指令碼執行狀態可能不會正確更新 (即使指令碼中最後一個步驟是重新啟動)。對於 Windows 受管節點，您需要在指令碼中指定 `exit 3010`。對於 Linux 和 macOS 受管節點，您需要指定 `exit 194`。結束程式碼會指示 AWS Systems Manager 客服人員 (SSM Agent) 重新啟動受管節點，然後在重新啟動完成後重新啟動指令碼。開始重新啟動前，SSM Agent 會在伺服器重新啟動期間，於通訊遭到中斷的雲端中通知 Systems Manager 服務。

> **Note**
>
> 重新啟動指令碼不能是 `aws:runDocument` 外掛程式的一部分。如果文件包含重新啟動指令碼，而另一份文件嘗試透過 `aws:runDocument` 外掛程式執行該文件，則 SSM Agent 會傳回錯誤。

建立等冪的指令碼

當開發重新啟動受管節點的指令碼時，將指令碼設為等冪，如此一來，指令碼執行會在重新啟動後中斷處繼續進行。等冪指令碼會管理狀態並驗證動作是否已執行。這可在僅預期執行一次步驟時，防止執行多次步驟。

以下是等冪指令碼的概述範例，該指令碼會多次重新啟動受管節點。

```
$name = Get current computer name
If ($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
    }
```

範例

以下指令碼範例使用結束程式碼來重新啟動受管節點。Linux 範例會在 Amazon Linux 上安裝套件更新，然後重新啟動該節點。Windows Server 範例會在節點上安裝 Telnet-Client，然後重新啟動該節點。

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
    }
```

了解命令狀態

Run Command中的工具 會針對處理命令期間和每個處理命令的受管節點 AWS Systems Manager，報告不同狀態的詳細狀態資訊。您可以使用下列方法來監控命令狀態：

- 在 Run Command 主控台介面的 Commands (命令) 標籤中選擇 Refresh (重新整理) 圖示。
- 使用 AWS Command Line Interface () 呼叫 list-commands 或 list-command-invocationsAWS CLI。或使用 呼叫 Get-SSMCommand 或 Get-SSMCommandInvocation AWS Tools for Windows PowerShell。
- 設定 Amazon EventBridge 以回應狀態或狀態變更。
- 將 Amazon Simple Notification Service (Amazon SNS) 設定為針對所有狀態變更或 `Failed` 或 `TimedOut` 之類的特定狀態傳送通知。

Run Command 狀態

Run Command 為以下三個領域報告狀態詳細資訊：外掛程式、呼叫和整體命令狀態。外掛程式是一種程式碼執行區塊，其在命令的 SSM 文件中有定義。如需外掛程式的詳細資訊，請參閱 命令文件外掛程式參考。

當您同時將命令傳送到多個受管節點，將每個節點視為命令的每個命令副本是一種命令呼叫。例如，如果您使用的是 `AWS-RunShellScript` 文件和將 `ifconfig` 命令傳送到 20 個執行個體，該命令會有 20 個叫用。每個命令呼叫會個別報告狀態。指定命令叫用的外掛程式也會個別報告狀態。

最後，Run Command 包含所有外掛程式和呼叫的彙總命令狀態。彙總命令狀態可能會與外掛程式或呼叫所回報的狀態不同，如以下資料表所示。

> **Note**
>
> 如果您使用 `max-concurrency` 或 `max-errors` 參數對大量受管節點執行命令，命令狀態會反映這些參數所實施的限制，如下列資料表所述。如需這些參數的相關資訊，請參閱 大規模執行命令。

命令外掛程式和呼叫的詳細狀態

Status	詳細資訊
待定	該命令尚未傳送至受管節點或尚未被 SSM Agent 接收。如果代理程式未收到命令之前的時間長度等於 Timeout (seconds) (逾時 (秒)) 參數與 Execution timeout (執行逾時) 參數的總和，則狀態會變更為 `Delivery Timed Out`。

Status	詳細資訊
InProgress	Systems Manager 嘗試將命令傳送至執行個體，或由 SSM Agent 接收命令，並開始在受管節點上執行。根據所有命令外掛程式的結果，狀態變更為 Success、Failed、Delivery Timed Out 或 Execution Timed Out。異常情形：如果代理程式未執行或在節點上不可用，則命令狀態會保留在 In Progress，直到代理程式再次可用，或直到達到執行逾時限制為止。此狀態會接著變更為終止狀態。
延遲	系統嘗試將命令傳送至受管節點，但未成功。系統會再次重試。

Status	詳細資訊
成功	此狀態會在各種條件下傳回。此狀態不代表命令在節點成功處理。例如，SSM Agent 可以在受管節點上接收命令，並且由於 PowerShell `ExecutionPolicy` 阻止命令執行，命令會傳回零的結束程式碼。這是一個終端狀態。導致命令傳回 `Success` 狀態的條件為： • 當以單一執行個體為目標時，命令在受管節點上透過 SSM Agent 接收且傳回零的結束程式碼。 • 以多個執行個體為目標時，失敗的調用數量並未超過命令中指定的錯誤閾值。 • 以多個執行個體為目標時，至少有 1 個調用已成功，而其他則逾時。指定的錯誤閾值仍然適用。 • 以標籤為目標時，找不到與該標籤關聯的執行個體。 • 以標籤為目標時，失敗的調用數量並未超過命令中指定的錯誤閾值。 • 以標籤為目標時，至少有 1 個調用已成功，而其他則逾時。指定的錯誤閾值仍然適用。 • 您在作業系統層級強制執行的應用程式或政策可防止或覆寫命令的執行，導致傳回零的結束程式碼。 **Note** 以資源群組為目標時，同樣的條件也適用。若要排除錯誤或取得更多命令執行的相關資訊，透過傳回適當的結束代碼 (非零結束代碼適用於命令失敗)，傳送處理錯誤或例外狀況的命令。

Status	詳細資訊
DeliveryTimedOut	在總逾時過期前，不會將此命令傳送到受管理節點。總逾時不會計入父命令的 max-errors 限制，但它們不會影響父命令狀態是否為 Success、Incomplete 或 DeliveryTimed Out。這是一個終端狀態。
ExecutionTimedOut	命令自動化會在受管節點上開始，但是命令在執行逾時過期前未完成。執行超時計算為失敗，這將會傳送非零回覆，Systems Manager 將會停止嘗試執行命令自動化，並回報失敗狀態。
失敗	命令在受管節點上未成功執行。若是外掛程式，這表示結果碼不為零。若是命令叫用，這表示一或多個外掛程式的結果碼不為零。呼叫失敗不會計入父命令的 max-errors 限制。這是一個終端狀態。
已取消	命令在完成前已取消。這是一個終端狀態。
交付成果	無法將命令傳送至受管節點。節點可能不存在或可能不回應。交付成果呼叫不會計入父命令的 max-errors 限制，但它們不會影響父命令狀態是否為 Success 或 Incomplete 。例如，若命令中所有的呼叫狀態都是 Undeliverable ，則命令狀態會傳回 Failed。但是，若命令進行 5 次呼叫，其中 4 次傳回了狀態 Undeliverable ，1 次則傳回了狀態 Success，則父命令的狀態便是 Success。這是一個終端狀態。
已終止	父命令超過其 max-errors 限制和系統已取消後續命令呼叫。這是一個終端狀態。

Status	詳細資訊
InvalidPlatform	命令已傳送到與所選文件指定的所需平台不相符的受管節點。Invalid Platform 不會計入父命令的最高錯誤限制，但它確實會影響父命令狀態為成功或失敗。例如，若命令中所有的呼叫狀態都是 Invalid Platform，則命令狀態會傳回 Failed。但是，若命令進行 5 次呼叫，其中 4 次傳回了狀態 Invalid Platform，1 次則傳回了狀態 Success，則父命令的狀態便是 Success。這是一個終端狀態。
AccessDenied	啟動命令的 AWS Identity and Access Management (IAM) 使用者或角色無法存取目標受管節點。 Access Denied 不會計入父命令max-errors 的限制，但確實會導致父命令狀態為 Success或 Failed。例如，若命令中所有的呼叫狀態都是 Access Denied，則命令狀態會傳回 Failed。但是，若命令進行 5 次呼叫，其中 4 次傳回了狀態 Access Denied，1 次則傳回了狀態 Success，則父命令的狀態便是 Success。這是一個終端狀態。

命令的詳細狀態

Status	詳細資訊
待定	任何受管節點上的代理程式都尚未收到命令。
InProgress	已將此命令傳送到至少一個受管節點，但在所有節點上都尚未達到最終狀態。
延遲	系統嘗試將命令傳送至節點，但未成功。系統會再次重試。
成功	在指定或目標受管節點上透過 SSM Agent 接收的命令且傳回零的結束程式碼。所有命令呼叫已達到終止狀態，而且未達到 max-errors 的

Status	詳細資訊
	值。此狀態不代表命令在所有指定或目標受管節點上順利處理。這是一個終端狀態。 **Note** 若要排除錯誤或取得更多命令執行的相關資訊，透過傳回適當的結束代碼 (非零結束代碼適用於命令失敗)，傳送處理錯誤或例外狀況的命令。
DeliveryTimedOut	在總逾時過期前，不會將此命令傳送到受管理節點。`max-errors` 或多個命令呼叫的值顯示為狀態 `Delivery Timed Out`。這是一個終端狀態。
失敗	命令在受管節點上未成功執行。`max-errors` 或多個命令呼叫的值顯示為狀態 `Failed`。這是一個終端狀態。
未完成	已在所有受管節點上嘗試命令且一或多個呼叫沒有值 `Success`。不過，沒有足夠的呼叫失敗，狀態會變為 `Failed`。這是一個終端狀態。
已取消	命令在完成前已取消。這是一個終端狀態。
RateExceeded	超過待處理呼叫帳戶配額的命令視為目標之受管節點數量。系統已在任何節點上執行命令前將其取消。這是一個終端狀態。

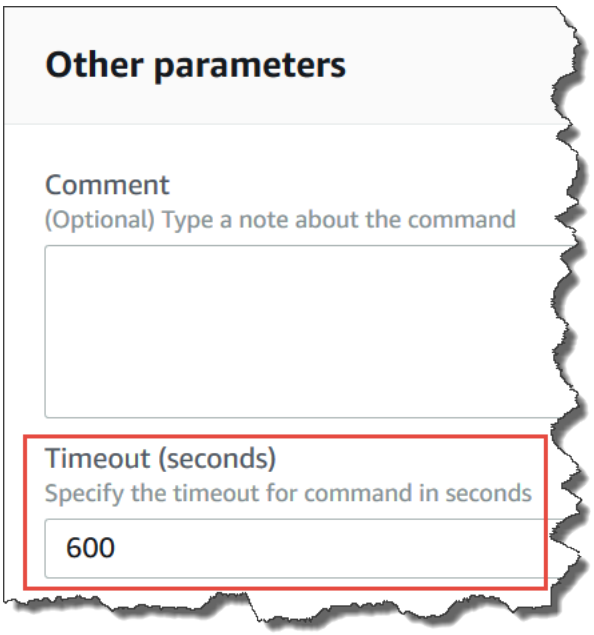
Status	詳細資訊
AccessDenied	啟動命令的使用者或角色沒有存取目標資源群組的權限。AccessDenied 不會計入父命令的 max-errors 限制中，但可能會決定父命令的狀態是 Success 或 Failed。(例如，若命令中所有的叫用狀態為 AccessDenied ，則傳回的命令狀態為 Failed。但是，若命令進行 5 次叫用，其中 4 次則傳回了狀態 AccessDenied ，其中 1 次傳回狀態 Success，則父命令的狀態為 Success。) 這是一個終端狀態。
標籤中沒有執行個體	命令瞄準的標籤金鑰對值或資源群組並未符合任何受管節點。這是一個終端狀態。

了解命令逾時值

執行命令時，Systems Manager 會強制執行下列逾時值。

總逾時

在 Systems Manager 主控台中，您可以在 Timeout (seconds) (逾時 (秒)) 欄位中指定逾時值。傳送命令之後，Run Command 會檢查命令是否已過期。如果命令達到命令過期期限 (總逾時)，則它會將所有狀態為 InProgress、Pending 或 Delayed 之叫用的狀態變更為 DeliveryTimedOut。



在更技術的層面上，總逾時 (Timeout (seconds) (逾時 (秒)) 包含兩個逾時值，如下所示：

```
Total timeout = "Timeout(seconds)" from the console + "timeoutSeconds": "{{ executionTimeout }}" from your SSM document
```

例如，Systems Manager 主控台中 Timeout (seconds) (逾時 (秒)) 的預設值是 600 秒。如果您使用 `AWS-RunShellScript` SSM 文件執行命令，"timeoutSeconds": "{{ executionTimeout }}" 的預設值是 3600 秒，如下列文件範例所示：

```
"executionTimeout": {
    "type": "String",
    "default": "3600",

"runtimeConfig": {
  "aws:runShellScript": {
    "properties": [
      {
        "timeoutSeconds": "{{ executionTimeout }}"
```

這表示在系統將命令狀態設定為 `DeliveryTimedOut` 之前，執行時間為 4,200 秒 (70 分鐘)。

執行逾時

在 Systems Manager 主控台中，您可以在 Execution Timeout (執行逾時) 欄位中指定執行逾時值 (如果有的話)。並非所有 SSM 文件都需要您指定執行逾時。僅在 SSM 文件中已定義對應的輸入參數時，才會顯示執行逾時欄位。如果指定了逾時，命令即必須在此期間內完成。

> **Note**
>
> Run Command 依賴於 SSM Agent 文件終端機回應，以判斷命令是否已傳遞給代理程式。SSM Agent 必須傳送用於叫用或命令、被標記為 `ExecutionTimedOut` 的 `ExecutionTimedOut` 信號。

預設執行逾時

如果 SSM 文件不要求您明確指定執行逾時值，則 Systems Manager 會強制執行硬式編碼預設執行逾時。

Systems Manager 如何報告逾時

如果 Systems Manager 收到目標上 SSM Agent 的 `execution timeout` 回覆，則 Systems Manager 會將命令叫用標記為 `executionTimeout`。

如果 Run Command 不會收到來自 SSM Agent 的文件終端機回應，則命令叫用被標記為 `deliveryTimeout`。

為了判斷目標上的逾時狀態，SSM Agent 會結合所有參數和 SSM 文件的內容，以計算出 `executionTimeout`。當 SSM Agent 判斷命令已逾時，會將 `executionTimeout` 傳送至服務。

Timeout (seconds) (逾時 (秒)) 的預設為 3600 秒。Execution Timeout (執行逾時) 的預設也是 3600 秒。因此，命令的總預設逾時為 7200 秒。

> **Note**
>
> SSM Agent 會根據 SSM 文件類型和文件版本的不同，來處理 `executionTimeout`。

Run Command演練

本節中的逐步解說說明如何使用 中的工具 AWS Command Line Interface (Run Command)AWS CLI或 AWS Systems Manager來執行命令 AWS Tools for Windows PowerShell。

目錄

- 使用 Run Command 更新軟體
- 逐步解說： AWS CLI 搭配 使用 Run Command
- 逐步解說： AWS Tools for Windows PowerShell 搭配 使用 Run Command

您也可以檢視以下參考中的範例命令。

- Systems Manager AWS CLI 參考
- AWS Tools for Windows PowerShell - AWS Systems Manager

使用 Run Command 更新軟體

下列程序描述了如何更新受管節點上的軟體。

使用 Run Command 更新 SSM Agent

下列處理程序描述了如何更新受管節點上執行的 SSM Agent。您可以更新到 SSM Agent 的最新版本或降級到較舊版本。當您執行 命令時，系統會從 下載 版本 AWS、安裝該版本，然後解除安裝命令執行之前已存在的版本。如果在此程序期間發生錯誤，系統將轉返到命令執行前伺服器上的版本，且命令狀態會顯示命令失敗。

> **Note**
>
> 如果執行個體執行的是 macOS 11.0 (Big Sur) 版或更新版本，則執行個體必須具備 SSM Agent 3.1.941.0 版或更新版本才能執行 AWS-UpdateSSMAgent 文件。如果執行個體執行的是 3.1.941.0 之前發行的 SSM Agent 版本，則可以透過執行 `brew update` 和 `brew upgrade amazon-ssm-agent` 命令來更新 SSM Agent 以執行 AWS-UpdateSSMAgent 文件。

若要接收有關 SSM Agent 更新的通知，請訂閱 GitHub 上的 SSM Agent 版本備註頁面。

使用 Run Command 更新 SSM Agent

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Run Command。
3. 選擇執行命令。
4. 在 Command document (命令文件) 清單，請選擇 **AWS-UpdateSSMAgent**。
5. 在 Command parameters (命令參數) 區段，依您所需指定下列參數值：

   a. (選用) 對於 Version (版本)，輸入要安裝的 SSM Agent 版本。您可以安裝較舊版本的代理程式。如果您未指定版本，該服務會安裝最新版本。

   b. (選用) 針對 Allow Downgrade (允許降級)，請選擇 true (true) 來安裝舊版 SSM Agent。如果選擇此選項，則指定舊版本編號。選擇 false (false) 來僅安裝最新版的服務。

6. 在 Targets (目標) 區段中，透過手動指定標籤、選取執行個體或邊緣裝置，或指定資源群組，選擇您要執行這項操作的受管節點。

   > **Tip**
   >
   > 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

7. 對於其他參數：

   - 在 Comment (註解) 中，輸入此命令的相關資訊。
   - 在逾時 (秒) 中，指定在命令執行全面失敗之前，系統要等候的秒數。

8. 對於 Rate control (速率控制)：

   - 在並行中，指定可同時執行命令的受管節點數目或百分比。

     > **Note**
     >
     > 如果透過指定套用至受管節點的標籤或指定 AWS 資源群組選取了目標，且您不確定會以多少個受管節點為目標，則透過指定百分比限制可以同時執行文件之目標的數量。

   - 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比之節點上的命令失敗之後，停止在其他受管節點上執行命令。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止傳送命令。仍在處理命令的受管節點也可能會傳送錯誤。

9. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取將命令輸出寫入至 S3 儲存貯體方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

   > **Note**
   >
   > 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給執行個體之執行個體設定檔 (適用於 EC2 執行個體) 或 IAM 服務角色 (啟用混合模式的機器) 的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確定與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

10. 在SNS 通知區段中，如果您要傳送有關命令執行狀態的通知，請選取啟用 SNS 通知核取方塊。

    如需為 Run Command 設定 Amazon SNS 通知的詳細資訊，請參閱 使用 Amazon SNS 通知監控 Systems Manager 狀態變更。

11. 選擇執行。

使用 Run Command 更新 PowerShell

下列處理程序說明如何將 PowerShell 更新至 Windows Server 2012 和 2012 R2 受管節點 5.1 版。此處理程序中提供的指令碼會下載 Windows 管理架構 (WMF) 5.1 版更新，並開始安裝更新。節點會在此程序期間重新開機，因為在安裝 WMF 5.1 時需要這樣進行。更新的下載和安裝大約需要五分鐘才能完成。

使用 Run Command 更新 PowerShell

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Run Command。
3. 選擇執行命令。
4. 在 Command document (命令文件) 清單，請選擇 **AWS-RunPowerShellScript**。
5. 在 Commands (命令) 區段中，貼上適用於您作業系統的下列命令。

   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 (目標) 區段中，透過手動指定標籤、選取執行個體或邊緣裝置，或指定資源群組，選擇您要執行這項操作的受管節點。

> **Tip**
>
> 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

7. 對於其他參數：

   - 在 Comment (註解) 中，輸入此命令的相關資訊。
   - 在逾時 (秒) 中，指定在命令執行全面失敗之前，系統要等候的秒數。

8. 對於 Rate control (速率控制)：

   - 在並行中，指定可同時執行命令的受管節點數目或百分比。

> **Note**
>
> 如果透過指定套用至受管節點的標籤或指定 AWS 資源群組選取了目標，且您不確定會以多少個受管節點為目標，則透過指定百分比限制可以同時執行文件之目標的數量。

   - 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比之節點上的命令失敗之後，停止在其他受管節點上執行命令。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止傳送命令。仍在處理命令的受管節點也可能會傳送錯誤。

9. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取將命令輸出寫入至 S3 儲存貯體方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

> **Note**
>
> 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給執行個體之執行個體設定檔 (適用於 EC2 執行個體) 或 IAM 服務角色 (啟用混合模式的機器) 的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確定與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

10. 在SNS 通知區段中，如果您要傳送有關命令執行狀態的通知，請選取啟用 SNS 通知核取方塊。

    如需為 Run Command 設定 Amazon SNS 通知的詳細資訊，請參閱 使用 Amazon SNS 通知監控 Systems Manager 狀態變更。

11. 選擇執行。

受管節點重新開機且更新安裝完成後，請連線到節點，以確認 PowerShell 已成功升級至 5.1 版。若要檢查節點上的 PowerShell 版本，請開啟 PowerShell 並輸入 `$PSVersionTable`。如果升級成功，則輸出資料表的 `PSVersion` 值會顯示 5.1。

如果 `PSVersion` 值不同於 5.1，例如 3.0 或 4.0，請檢閱 Windows Logs (Windows 日誌) 下事件檢視器中的 Setup (設定) 日誌。這些日誌會指出更新安裝失敗的原因。

逐步解說： AWS CLI 搭配 使用 Run Command

下列範例逐步解說說明如何使用 AWS Command Line Interface (AWS CLI) 檢視命令和命令參數的相關資訊、如何執行命令，以及如何檢視這些命令的狀態。

> **Important**
>
> 只有信任的管理員才能使用本主題中顯示的 AWS Systems Manager 預先設定文件。Systems Manager 文件中指定的命令或指令碼會使用在您的受管節點上的管理許可執行。如果使用者具備執行任何預先定義的 Systems Manager 文件 (任何以 `AWS-` 為開頭的文件) 的許可，則該使用者還具有該節點的管理員存取權。對於所有其他使用者，您應該建立嚴格的文件和並將它們分享給特定使用者。

主題

- 步驟 1：入門
- 步驟 2：執行 Shell 指令碼以檢視資源詳細資訊
- 步驟 3：使用 AWS-RunShellScript 文件發送簡單的命令
- 步驟 4：使用 Run Command 執行一個簡單的 Python 指令碼
- 步驟 5：使用 Run Command 執行 Bash 指令碼

步驟 1：入門

您必須在您想要設定的受管節點上具有管理員許可，或已在 AWS Identity and Access Management (IAM) 中被授予適當的許可。另請注意，此範例使用美國東部（俄亥俄）區域 (us-east-2)。 Run Command 可在 中 Systems Manager 服務端點中 AWS 區域 列出的 中使用Amazon Web Services 一般參考。如需詳細資訊，請參閱設定 AWS Systems Manager的受管節點。

使用 執行命令 AWS CLI

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。
2. 列出所有可用的文件。

   此命令會列出根據 IAM 許可適用於您帳戶的所有文件。

   ```
   aws ssm list-documents
   ```
3. 確定受管節點已準備好接收命令。

   以下命令的輸出會顯示受管節點是否上線。

   Linux & macOS

   ```
   aws ssm describe-instance-information \
       --output text --query "InstanceInformationList[*]"
   ```

   Windows

   ```
   aws ssm describe-instance-information ^
       --output text --query "InstanceInformationList[*]"
   ```

4. 使用下列命令來檢視特定受管節點的詳細資訊。

> **Note**
>
> 若要在此演練中執行命令，請替換執行個體和命令 ID。對於受管 AWS IoT Greengrass 核心裝置，請使用執行個體 ID 的 mi-*ID_number*。會將命令 ID 傳回做為 send-command 的回應。執行個體 IDs 可從 中取得Fleet Manager，這是 中的工具 AWS Systems Manager。

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
```

步驟 2：執行 Shell 指令碼以檢視資源詳細資訊

您可以使用 Run Command 和 AWS-RunShellScript 文件，在受管節點上執行任何命令或指令碼，如同您在本機登入一樣。

View the description and available parameters (查看描述和可用參數)

請執行下列命令以檢視 Systems Manager JSON 文件的說明。

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]"
```

執行下列命令來檢視可用的參數和這些參數的詳細資訊。

Linux & macOS

```
aws ssm describe-document \
    --name "AWS-RunShellScript" \
    --query "Document.Parameters[*]"
```

Windows

```
aws ssm describe-document ^
    --name "AWS-RunShellScript" ^
    --query "Document.Parameters[*]"
```

步驟 3：使用 **AWS-RunShellScript** 文件發送簡單的命令

若要取得 Linux 受管節點的 IP 資訊，請執行以下命令。

如果您正在鎖定 Windows Server 受管節點，請變更 `document-name` 至 `AWS-RunPowerShellScript`，然後將 `command` 從 `ifconfig` 變更至 `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
```

取得命令資訊與回應資料

以下命令使用從之前命令傳回的命令 ID，以取得命令執行的詳細資訊和回應資料。如果命令完成，系統會傳回回應資料。如果命令執行顯示 "Pending" 或 "InProgress"，您可再次執行此命令，以查看回應資料。

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
```

識別使用者

以下命令會顯示執行命令的預設使用者。

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")
```

取得命令狀態

以下命令使用命令 ID，以取得在受管節點上命令執行的狀態。此範例使用在之前命令傳回的命令 ID。

Linux & macOS

```
aws ssm list-commands \
    --command-id "command-ID"
```

Windows

```
aws ssm list-commands ^
    --command-id "command-ID"
```

取得命令詳細資訊

以下命令使用來自先前命令的命令 ID，以取得每個受管節點命令執行的狀態。

Linux & macOS

```
aws ssm list-command-invocations \
    --command-id "command-ID" \
    --details
```

Windows

```
aws ssm list-command-invocations ^
    --command-id "command-ID" ^
    --details
```

使用特定受管節點的回應資料來取得命令資訊

下列命令會傳回特定受管節點的原始 `aws ssm send-command` 請求輸出。

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
```

顯示 Python 版本

以下命令會傳回在節點上執行的 Python 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}"'
```

步驟 4：使用 Run Command 執行一個簡單的 Python 指令碼

下面的命令會使用 Run Command 執行一個簡單的 Python "Hello World" 指令碼。

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}"'
```

步驟 5：使用 Run Command 執行 Bash 指令碼

本節中的範例演示了如何使用 Run Command 執行以下 Bash 指令碼。

如需使用 Run Command 執行存放在遠端位置之指令碼的範例，請參閱 從 Amazon Simple Storage Service (Amazon S3) 執行指令碼 和 從 GitHub 執行指令碼。

```
#!/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
```

此指令碼會在 Amazon Linux 和 Red Hat Enterprise Linux(RHEL) 執行個體上安裝 AWS CodeDeploy 代理程式，如 AWS CodeDeploy 使用者指南中的為 CodeDeploy 建立 Amazon EC2 執行個體中所述。

指令碼會從 美國東部 （俄亥俄） 區域 (us-east-2) 的 AWS 受管 S3 儲存貯體安裝 CodeDeploy 代理程式`aws-codedeploy-us-east-2`。

在 AWS CLI 命令中執行 bash 指令碼

下列範例示範如何使用 `--parameters` 選項在 CLI 命令中包含 Bash 指令碼。

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 檔案中執行 Bash 指令碼

在下列範例中，Bash 指令碼的內容會存放在 JSON 檔案中，而且檔案會使用 `--cli-input-json` 選項包含在命令中。

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
```

參考 `installCodeDeployAgent.json` 檔案的內容如以下範例所示。

```
{
    "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"
        ]
    }
}
```

逐步解說： AWS Tools for Windows PowerShell 搭配 使用 Run Command

下列範例示範如何使用 AWS Tools for Windows PowerShell 檢視命令和命令參數的相關資訊、如何執行命令，以及如何檢視這些命令的狀態。此逐步解說包含每個預先定義 AWS Systems Manager 文件的範例。

> **Important**
>
> 只有信任管理員應受允許使用在這個主題中顯示的 Systems Manager 預先設定的文件。Systems Manager 文件中指定的命令或指令碼會使用在您的受管節點上的管理許可執行。如果使用者具有執行任何預先定義 Systems Manager 文件 （ 以 開頭的任何文件 AWS) 的許可，則該使用者也具有節點的管理員存取權。對於所有其他使用者，您應該建立嚴格的文件和並將它們分享給特定使用者。

主題

- 設定 AWS Tools for Windows PowerShell 工作階段設定
- 列出所有可用的文件
- 執行 PowerShell 命令或指令碼
- 使用 AWS-InstallApplication 文件來安裝應用程式
- 使用 AWS-InstallPowerShellModule JSON 文件安裝 PowerShell 模組
- 使用 AWS-JoinDirectoryServiceDomain JSON 文件將受管節點加入網域
- 使用 AWS-ConfigureCloudWatch 文件將 Windows 指標傳送到 Amazon CloudWatch Logs
- 使用 AWS-UpdateEC2Config 文件更新 EC2Config
- 使用 AWS-ConfigureWindowsUpdate 文件，開啟或關閉 Windows 自動更新
- 使用Run Command管理 Windows 更新

設定 AWS Tools for Windows PowerShell 工作階段設定

指定您的登入資料。

在您的本機電腦上開啟 Tools for Windows PowerShell，然後執行下列命令，以指定您的憑證。您必須在要設定的受管節點上擁有管理員許可，或者您必須已在 AWS Identity and Access Management (IAM) 中獲授予適當的許可。如需詳細資訊，請參閱設定 AWS Systems Manager的受管節點。

```
Set-AWSCredentials -AccessKey key-name -SecretKey key-name
```

設定預設 AWS 區域

執行下列命令，以設定 PowerShell 工作階段的區域。此範例使用美國東部（俄亥俄）區域 (us-east-2)。 Run Command 可在 中 Systems Manager 服務端點中 AWS 區域 列出的 中使用Amazon Web Services 一般參考。

```
Set-DefaultAWSRegion `
    -Region us-east-2
```

列出所有可用的文件

此命令會列出可供您帳戶使用的所有文件。

```
Get-SSMDocumentList
```

執行 PowerShell 命令或指令碼

您可以使用 Run Command 和 AWS-RunPowerShell 文件，在受管節點上執行任何命令或指令碼，如同您在本機登入一樣。您可以發出命令或在本機指令碼輸入路徑來執行命令。

> **Note**
>
> 如需使用 Run Command 呼叫指令碼時重新啟動受管節點的資訊，請參閱 執行命令時處理重新啟動。

View the description and available parameters (查看描述和可用參數)

```
Get-SSMDocumentDescription `
    -Name "AWS-RunPowerShellScript"
```

View more information about parameters (檢視參數的詳細資訊)

```
Get-SSMDocumentDescription `
    -Name "AWS-RunPowerShellScript" | Select -ExpandProperty Parameters
```

使用 **AWS-RunPowerShellScript** 文件來傳送命令

以下命令顯示 "C:\Users" 目錄的內容，以及在兩個受管節點上 "C:\" 的內容。

```
$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:\')}
```

取得命令請求詳細資訊

以下命令使用 CommandId，以取得在兩個受管節點上命令執行的狀態。此範例使用在之前命令傳回的 CommandId。

```
Get-SSMCommand `
    -CommandId $runPSCommand.CommandId
```

此範例中的狀態命令可以是 Success、Pending 或 InProgress。

取得每個受管節點的命令資訊

以下命令使用來自先前命令的 CommandId，以取得每個受管節點命令執行的狀態。

```
Get-SSMCommandInvocation `
    -CommandId $runPSCommand.CommandId
```

使用特定受管節點的回應資料來取得命令資訊

下列命令會傳回特定受管節點的原始 Send-SSMCommand 輸出。

```
Get-SSMCommandInvocation `
    -CommandId $runPSCommand.CommandId `
    -Details $true `
    -InstanceId instance-ID | Select -ExpandProperty CommandPlugins
```

取消命令

下列命令會取消 AWS-RunPowerShellScript 文件的 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
```

檢查命令狀態

下列命令會檢查 Cancel 命令的狀態。

```
Get-SSMCommand `
    -CommandId $cancelCommand.CommandId
```

使用 **AWS-InstallApplication** 文件來安裝應用程式

使用 Run Command 和 AWS-InstallApplication 文件，您可以在受管節點上安裝、修復或解除安裝應用程式。此命令需要 MSI 的路徑或地址。

> **Note**
>
> 如需使用 Run Command 呼叫指令碼時重新啟動受管節點的資訊，請參閱 執行命令時處理重新啟動。

View the description and available parameters (查看描述和可用參數)

```
Get-SSMDocumentDescription `
    -Name "AWS-InstallApplication"
```

View more information about parameters (檢視參數的詳細資訊)

```
Get-SSMDocumentDescription `
    -Name "AWS-InstallApplication" | Select -ExpandProperty Parameters
```

使用 **AWS-InstallApplication** 文件來傳送命令

以下命令會在受管節點上以自動模式安裝 Python 版本，並將輸出記錄至 C: 磁碟機上的本機文字檔。

```
$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'}
```

取得每個受管節點的命令資訊

以下命令使用 CommandId，以取得命令執行的狀態。

```
Get-SSMCommandInvocation `
    -CommandId $installAppCommand.CommandId `
    -Details $true
```

使用特定受管節點的回應資料來取得命令資訊

以下命令會傳回 Python 安裝的結果。

```
Get-SSMCommandInvocation `
    -CommandId $installAppCommand.CommandId `
```

```
    -Details $true `
    -InstanceId instance-ID | Select -ExpandProperty CommandPlugins
```

使用 **AWS-InstallPowerShellModule** JSON 文件安裝 PowerShell 模組

您可以使用 Run Command，在受管節點上安裝 PowerShell 模組。如需 PowerShell 模組的詳細資訊，請參閱 Windows PowerShell 模組。

View the description and available parameters (查看描述和可用參數)

```
Get-SSMDocumentDescription `
    -Name "AWS-InstallPowerShellModule"
```

View more information about parameters (檢視參數的詳細資訊)

```
Get-SSMDocumentDescription `
    -Name "AWS-InstallPowerShellModule" | Select -ExpandProperty Parameters
```

安裝 PowerShell 模組

以下命令會下載 EZOut.zip 檔案、進行安裝，然後執行額外的命令來安裝 XPS 檢視器。最後，此命令的輸出會上傳到名為 amzn-s3-demo-bucket 的 S3 儲存貯體。

```
$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
```

取得每個受管節點的命令資訊

以下命令使用 CommandId，以取得命令執行的狀態。

```
Get-SSMCommandInvocation `
    -CommandId $installPSCommand.CommandId `
    -Details $true
```

使用受管節點的回應資料來取得命令資訊

下列命令會傳回特定 CommandId 之原始 Send-SSMCommand 的輸出。

```
Get-SSMCommandInvocation `
    -CommandId $installPSCommand.CommandId `
    -Details $true | Select -ExpandProperty CommandPlugins
```

使用 **AWS-JoinDirectoryServiceDomain** JSON 文件將受管節點加入網域

使用 Run Command，您可以快速將受管節點加入 AWS Directory Service 網域。執行此命令之前，建立目錄。也建議您進一步了解 AWS Directory Service的更多資訊。如需詳細資訊，請參閱 AWS Directory Service 管理員指南。

您只能將受管節點加入網域。您無法從網域移除節點。

> **Note**
>
> 如需使用 Run Command 呼叫指令碼時受管節點的資訊，請參閱 執行命令時處理重新啟動。

View the description and available parameters (查看描述和可用參數)

```
Get-SSMDocumentDescription `
    -Name "AWS-JoinDirectoryServiceDomain"
```

View more information about parameters (檢視參數的詳細資訊)

```
Get-SSMDocumentDescription `
    -Name "AWS-JoinDirectoryServiceDomain" | Select -ExpandProperty Parameters
```

將受管節點加入網域

下列命令會將受管節點加入至指定的 AWS Directory Service 網域，並將任何產生的輸出上傳至範例 Amazon Simple Storage Service (Amazon S3) 儲存貯體。

```
$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
```

取得每個受管節點的命令資訊

以下命令使用 CommandId，以取得命令執行的狀態。

```
Get-SSMCommandInvocation `
    -CommandId $domainJoinCommand.CommandId `
    -Details $true
```

使用受管節點的回應資料來取得命令資訊

此命令會傳回特定 CommandId 之原始 Send-SSMCommand 的輸出。

```
Get-SSMCommandInvocation `
    -CommandId $domainJoinCommand.CommandId `
    -Details $true | Select -ExpandProperty CommandPlugins
```

使用 **AWS-ConfigureCloudWatch** 文件將 Windows 指標傳送到 Amazon CloudWatch Logs

您可以在應用程式、系統、安全性和 Windows 事件追蹤 (ETW) 日誌中將 Windows Server 訊息傳送至 Amazon CloudWatch Logs。當您第一次啟用記錄時，Systems Manager 會傳送在您開始上傳應用程式、系統、安全性和 ETW 日誌的日誌之一 (1) 分鐘內產生的所有日誌。在這個時間範圍內產生的日誌，則未包含在內。如果關閉記錄，之後再重新開啟記錄，則 Systems Manager 會從上次上次關閉的時間開始傳送記錄。如果是任何自訂的日誌檔案和 Internet Information Services (IIS) 日誌，Systems Manager 會從頭開始讀取日誌檔案。此外，Systems Manager 也可將效能計數器資料傳送至 CloudWatch Logs。

如果您之前在 EC2Config 已開啟 CloudWatch 整合，則 Systems Manager 設定會覆寫 C:\Program Files\Amazon\EC2ConfigService\Settings\AWS.EC2.Windows.CloudWatch.json 檔案中受管節點上本機存放的任何設定。如需使用 EC2Config 在單一受管節點上管理效能計數器和日誌的詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的使用 CloudWatch 代理程式從 Amazon EC2 執行個體和內部部署伺服器收集指標和日誌。

View the description and available parameters (查看描述和可用參數)

```
Get-SSMDocumentDescription `
    -Name "AWS-ConfigureCloudWatch"
```

View more information about parameters (檢視參數的詳細資訊)

```
Get-SSMDocumentDescription `
    -Name "AWS-ConfigureCloudWatch" | Select -ExpandProperty Parameters
```

將應用程式日誌傳送到 CloudWatch

以下命令會設定受管節點並將 Windows 應用程式日誌移到 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"]}}}'}
```

取得每個受管節點的命令資訊

以下命令使用 CommandId，以取得命令執行的狀態。

```
Get-SSMCommandInvocation `
    -CommandId $cloudWatchCommand.CommandId `
    -Details $true
```

使用特定受管節點的回應資料來取得命令資訊

以下命令會傳回 Amazon CloudWatch 組態的結果。

```
Get-SSMCommandInvocation `
    -CommandId $cloudWatchCommand.CommandId `
    -Details $true `
    -InstanceId instance-ID | Select -ExpandProperty CommandPlugins
```

使用 **AWS-ConfigureCloudWatch** 文件將效能計數器傳送到 CloudWatch

以下示範命令會將效能計數器上傳到 CloudWatch。如需更多資訊，請參閱 Amazon CloudWatch 使用者指南。

```
$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.CloudWatch.CloudWatchOutputComponent,AWS.EC2.Windows.Cl
 "Parameters":{"AccessKey":"", "SecretKey":"","Region":"region", "NameSpace":"Windows-
Default"}}], "Flows":{"Flows":["PerformanceCounter,CloudWatch"]}}}'}
```

使用 **AWS-UpdateEC2Config** 文件更新 EC2Config

您可以使用Run Command和 AWS-EC2ConfigUpdate 文件，更新在 Windows Server 受管節點上執行的 EC2Config 服務。此命令可將 EC2Config 服務更新到最新版本或您指定的版本。

View the description and available parameters (查看描述和可用參數)

```
Get-SSMDocumentDescription `
    -Name "AWS-UpdateEC2Config"
```

View more information about parameters (檢視參數的詳細資訊)

```
Get-SSMDocumentDescription `
    -Name "AWS-UpdateEC2Config" | Select -ExpandProperty Parameters
```

將 EC2Config 更新至最新版本

```
$ec2ConfigCommand = Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-UpdateEC2Config"
```

使用受管節點的回應資料來取得命令資訊

此命令會傳回來自先前 Send-SSMCommand 之指定命令的輸出。

```
Get-SSMCommandInvocation `
    -CommandId $ec2ConfigCommand.CommandId `
    -Details $true `
```

```
    -InstanceId instance-ID | Select -ExpandProperty CommandPlugins
```

將 EC2Config 更新至特定版本

以下命令會將 EC2Config 降級到較舊版本。

```
Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-UpdateEC2Config" `
    -Parameter @{'version'='4.9.3519'; 'allowDowngrade'='true'}
```

使用 **AWS-ConfigureWindowsUpdate** 文件，開啟或關閉 Windows 自動更新

使用 Run Command和 AWS-ConfigureWindowsUpdate 文件，開啟或關閉在 Windows Server 受管節點上的自動 Windows 更新。此命令會將 Windows 更新代理程式設定為在您指定的天和小時來下載和安裝 Windows 更新。如果更新需要重新啟動，受管節點會在已安裝更新後 15 分鐘自動重新啟動。您也可以透過此命令，將 Windows 更新設定為檢查更新，但不加以安裝。AWS-ConfigureWindowsUpdate 文件與 Windows Server 2008、2008 R2、2012、2012 R2 和 2016 相容。

View the description and available parameters (查看描述和可用參數)

```
Get-SSMDocumentDescription `
    -Name "AWS-ConfigureWindowsUpdate"
```

View more information about parameters (檢視參數的詳細資訊)

```
Get-SSMDocumentDescription `
    -Name "AWS-ConfigureWindowsUpdate" | Select -ExpandProperty Parameters
```

開啟 Windows 自動更新

以下命令會將 Windows Update 設定為每日下午 10:00 自動下載並安裝更新。

```
$configureWindowsUpdateCommand = Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-ConfigureWindowsUpdate" `
    -Parameters @{'updateLevel'='InstallUpdatesAutomatically';
 'scheduledInstallDay'='Daily'; 'scheduledInstallTime'='22:00'}
```

查看命令狀態以允許 Windows 自動更新

以下命令會使用 CommandId，以取得命令執行的狀態來允許 Windows 自動更新。

```
Get-SSMCommandInvocation `
    -Details $true `
    -CommandId $configureWindowsUpdateCommand.CommandId | Select -ExpandProperty
 CommandPlugins
```

關閉 Windows 自動更新

以下命令會降低 Windows 更新通知層級，讓系統檢查更新，但不會自動更新受管節點。

```
$configureWindowsUpdateCommand = Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-ConfigureWindowsUpdate" `
    -Parameters @{'updateLevel'='NeverCheckForUpdates'}
```

檢視命令狀態以關閉 Windows 自動更新

以下命令會使用 CommandId，以取得命令執行的狀態來關閉 Windows 自動更新。

```
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 文件，您可以管理 Windows Server 受管節點的更新。此命令會在受管節點上掃描或安裝遺漏的更新，並在安裝後選擇性重新啟動。您也可以為要在您環境中安裝的更新指定適當的分類和嚴重性層級。

> **Note**
>
> 如需使用 Run Command 呼叫指令碼時重新啟動受管節點的資訊，請參閱 執行命令時處理重新啟動。

以下範例示範如何執行指定的 Windows Update 管理任務。

搜尋所有遺漏 Windows 更新

```
Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-InstallWindowsUpdates" `
    -Parameters @{'Action'='Scan'}
```

安裝特定的 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'}
```

安裝重要的遺漏 Windows 更新

```
Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-InstallWindowsUpdates" `
    -Parameters @{'Action'='Install';'SeverityLevels'='Important';'AllowReboot'='True'}
```

安裝含特定排除的遺漏 Windows 更新

```
Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-InstallWindowsUpdates" `
    -Parameters @{'Action'='Install';'ExcludeKbs'='kb-ID-1,kb-
ID-2';'AllowReboot'='True'}
```

故障診斷 Systems Manager 執行命令

Run Command是 中的工具 AWS Systems Manager，提供每個命令執行的狀態詳細資訊。如需命令列狀態的詳細資訊，請參閱 了解命令狀態。您也可以使用此主題中的資訊，以協助排除與 Run Command 有關的問題。

主題

- 遺漏了我的一些受管節點
- 我指令碼中的一個步驟失敗，但整體狀態是「成功」

- SSM Agent 未正常執行

遺漏了我的一些受管節點

在 Run Command (執行命令) 頁面中，選擇要執行的 SSM 文件並在 Targets (目標)區段選取 Manually selecting instances (手動選取執行個體) 後，會顯示您可以選擇在其中執行命令的受管節點清單。

如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

在您建立、啟用、重新啟動或重新啟動受管節點、在節點Run Command上安裝 ，或將 AWS Identity and Access Management (IAM) 執行個體描述檔連接至節點後，可能需要幾分鐘的時間，受管節點才會新增至清單中。

我指令碼中的一個步驟失敗，但整體狀態是「成功」

使用 Run Command，您可以定義指令碼處理結束程式碼的方式。根據預設，在指令碼中執行的最後一個命令的結束程式碼會報告為整個指令碼的結束程式碼。但是，如果有任何命令在最後一個命令之前失敗，您可以包含條件陳述式來結束指令碼。如需詳細資訊和範例，請參閱 在命令中指定結束程式碼。

SSM Agent 未正常執行

如果使用 Run Command 遇到問題執行命令，則 SSM Agent 可能有問題。如需有關 SSM Agent 調查問題的資訊，請參閱 SSM Agent 疑難排解。

AWS Systems Manager Session Manager

Session Manager 是完全受管 AWS Systems Manager 的工具。透過 Session Manager，您可以管理 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、邊緣裝置、內部部署伺服器和虛擬機器。您可以使用互動式一鍵式瀏覽器型 shell 或 AWS Command Line Interface (AWS CLI) Session Manager來提供安全節點管理，而不需要開啟傳入連接埠、維護堡壘主機或管理 SSH 金鑰。 Session Manager 也可讓您遵循需要受管節點受管存取、嚴格安全實務和具有節點存取詳細資訊的日誌的企業政策，同時提供最終使用者對受管節點的簡單一鍵式跨平台存取。若要開始使用 Session Manager，請開啟 Systems Manager 主控台。在導覽窗格中，選擇 Session Manager。

Session Manager 如何為我的組織帶來益處？

Session Manager 提供這些好處：

- 使用 IAM 政策集中存取對受管節點的控制權。

管理員有一個單一位置授權和撤銷對受管節點的存取權。使用 only AWS Identity and Access Management (IAM) 政策，您可以控制組織中哪些個別使用者或群組可以使用，Session Manager以及他們可以存取哪些受管節點。

- 不需要開啟傳入連接埠和不需要管理堡壘主機或 SSH 金鑰

  開啟傳入 SSH 連接埠和遠端 PowerShell 連接埠讓您的受管節點上大幅增加未經授權實體或惡意命令在受管節點上執行的風險。Session Manager 可藉由關閉這些傳入連接埠協助您改善您的安全狀態，讓您免於管理 SSH 金鑰和憑證，堡壘主機以及跳接方塊。

- 從主控台和 CLI 使用一鍵式受管節點存取

  使用 AWS Systems Manager 主控台或 Amazon EC2 主控台，只要按一下即可啟動工作階段。您也可以使用 AWS CLI啟動執行單一命令或一系列命令的工作階段。由於受管節點的許可是透過 IAM 政策提供而不是 SSH 金鑰或其他機制，因此連線時間可大幅降低。

- 連線到混合多雲端環境中的 Amazon EC2 執行個體和非 EC2 節點

  您可以連線到混合多雲端環境中的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體和非 EC2 節點。

  若要使用 Session Manager 連線到非 EC2 節點，您必須先啟用進階執行個體層。使用進階執行個體層會產生費用。但是，使用 Session Manager 連線到 EC2 執行個體無需額外收費。如需相關資訊，請參閱 設定執行個體方案。

- 網路埠轉遞

  將受管節點內的任何連接埠重新引導至用戶端上的本機連接埠。之後，連線到本機連接埠並存取正在節點內執行的伺服器應用程式。

- 對 Windows、Linux 和 macOS 的跨平台支援

  Session Manager 透過單一工具提供對 Windows、Linux 和 macOS 的支援。例如，您不需要在 Linux 和 macOS 受管節點上使用 SSH 用戶端或在 Windows Server 受管節點上使用 RDP 連線。

- 記錄工作階段活動

  為了滿足組織中的操作或安全需求，您可能需要提供與您的受管節點做連結的記錄和在受管節點上執行的指令記錄。當組織中的使用者開始或結束工作階段時，您也會接收到通知。

  記錄功能透過與以下 AWS 服務的整合所提供：

  - AWS CloudTrail – AWS CloudTrail 擷取在 中進行 Session Manager API 呼叫的相關資訊， AWS 帳戶 並將其寫入您指定的 Amazon Simple Storage Service (Amazon S3) 儲存貯體中存放的日誌

檔案。一個儲存貯體可用於您帳戶中所有 CloudTrail 日誌。如需詳細資訊，請參閱使用 記錄 AWS Systems Manager API 呼叫 AWS CloudTrail。

- Amazon Simple Storage Service - 您可以選擇將工作階段日誌資料存放在所選的 Amazon Simple Storage Service (Amazon S3) 儲存貯體中，用於偵錯和疑難排解。透過使用 AWS KMS key，日誌資料可包含或不含加密金鑰傳送到您的 Amazon Simple Storage Service (Amazon S3) 儲存貯體。如需詳細資訊，請參閱使用 Amazon Simple Storage Service (Amazon S3) (主控台) 記錄工作階段資料。
- Amazon CloudWatch Logs - CloudWatch Logs 可讓您從各種 AWS 服務中監控、存放及存取日誌檔案。您可以將工作階段日誌資料傳送到 CloudWatch Logs 日誌群組，用於偵錯和疑難排解。可以使用 KMS 金鑰，在有或沒有 AWS KMS 加密的情況下，將日誌資料傳送到您的日誌群組。如需詳細資訊，請參閱使用 Amazon CloudWatch Logs (主控台) 記錄工作階段資料。。
- Amazon EventBridge 和 Amazon Simple Notification Service – EventBridge 可讓您設定規則，以偵測您指定的 AWS 資源何時發生變更。您可以建立一個規則來偵測當您的組織中的使用者開始或停止工作階段，然後透過 Amazon SNS (例如，文字或電子郵件訊息) 接收到有關事件的通知。您也可以設定 CloudWatch 事件來啟動其他回應。如需詳細資訊，請參閱使用 Amazon EventBridge (主控台) 監控工作階段活動。

> **Note**
>
> 透過連接埠轉送或 SSH 連線的 Session Manager 工作階段無法使用日誌記錄功能。這是因為 SSH 會加密所有工作階段資料，Session Manager 僅用作 SSH 連線的通道。

誰應該使用Session Manager？

- 任何想要改善其安全狀態、透過集中受管節點的存取控制來降低營運開銷，以及減少傳入節點存取 AWS 的客戶。
- 資訊安全專家想要監控並追蹤受管節點存取及活動、關閉受管節點的傳入連接埠、或啟動非公有 IP 地址直接連接受管節點。
- 管理員想要從單一位置取得或撤銷授權或有意替 Linux、macOS 和 Windows Server 受管節點的使用者提供解決方案。
- 使用者只要從瀏覽器按一下滑鼠，或 AWS CLI 不需要提供 SSH 金鑰，即可連線到受管節點。

Session Manager有哪些主要功能？

- 對 Windows Server、Linux 和 macOS 受管節點的支援

Session Manager 可讓您建立安全連線，以連線到 Amazon Elastic Compute Cloud (EC2) 執行個體、邊緣裝置、內部部署伺服器和虛擬機器。如需支援的作業系統類型清單，請參閱 設定 Session Manager。

> **Note**
>
> 針對現場部署機器所提供的 Session Manager 支援僅適用於進階執行個體層。如需相關資訊，請參閱 開啟 advanced-instances 方案。

- 主控台、CLI 和 SDK 存取 Session Manager 功能

  您可以利用下列方式來使用 Session Manager：

  AWS Systems Manager 主控台包含存取所有 Session Manager 功能，管理員和終端使用者皆適用。您可以使用 Systems Manager 主控台來執行任何與您的工作階段相關的任務。

  Amazon EC2 主控台能讓終端使用者連線至已獲得工作階段許可的 EC2 執行個體。

  AWS CLI 包含存取 Session Manager 功能，適用於最終使用者。您可以使用 來啟動工作階段、檢視工作階段清單，以及永久結束工作階段 AWS CLI。

  > **Note**
  >
  > 若要使用 AWS CLI 執行工作階段命令，您必須使用 CLI 的 1.16.12 版（或更新版本），而且您必須在本機電腦上安裝Session Manager外掛程式。如需相關資訊，請參閱 安裝 的 Session Manager外掛程式 AWS CLI。若要在 GitHub 上檢視該外掛程式，請參閱 session-manager-plugin。

- IAM 存取控制

  透過使用 IAM 政策，您可以控制組織中哪些成員可以初始化受管節點到工作階段裡以及那些節點他們可以存取。您也可以提供對受管節點的臨時存取權。例如，您可能想要提供的現場值班工程師 (或一組值班工程師) 只在值班時存取其產品伺服器。

- 記錄支援

  Session Manager 透過整合多個其他 AWS 服務，為您提供在 AWS 帳戶 中記錄工作階段歷史記錄的選項。如需詳細資訊，請參閱 記錄工作階段活動 和 啟用和停用工作階段記錄。

- 可設定的 shell 描述檔

Session Manager 提供在工作階段中設定偏好設定的選項。這些可自訂的描述檔可讓您定義偏好設定，例如 shell 偏好設定、環境變數、工作目錄，以及在工作階段啟動時執行的多個命令。

- 客戶金鑰資料加密支援

  您可以設定 Session Manager 來加密您傳送到 Amazon Simple Storage Service (Amazon S3) 儲存貯體或串流到 CloudWatch Logs 日誌群組的工作階段資料日誌。您也可以將 Session Manager 設定為以進一步加密在工作階段期間用戶端機器與受管節點之間傳輸的資料。如需相關資訊，請參閱啟用和停用工作階段記錄和進行工作階段偏好設定。

- AWS PrivateLink 支援沒有公有 IP 地址的受管節點

  您也可以使用 為 Systems Manager 設定 VPC 端點 AWS PrivateLink ，以進一步保護工作階段。會 AWS PrivateLink 限制受管節點、Systems Manager 和 Amazon EC2 之間的所有網路流量到 Amazon 網路。如需詳細資訊，請參閱使用適用於 Systems Manager 的 VPC 端點來改善 EC2 執行個體的安全性。

- 通道

  在工作階段中，使用 Session-type AWS Systems Manager (SSM) 文件，在用戶端機器的本機連接埠與受管節點的遠端連接埠之間，通道流量，例如 http 或自訂通訊協定。

- 互動式命令

  建立 Session-type SSM 文件，使用工作階段來以互動方式執行單一命令，讓您具備管理使用者能在受管節點上進行何種作業的方式。

什麼是工作階段？

工作階段是使用 Session Manager 對受管節點進行的連線。工作階段是以用戶端 (您) 與遠端受管節點 (串流命令的輸入和輸出) 之間的安全雙向通訊通道為基礎。用戶端和受管節點之間的流量會使用 TLS 1.2 加密，並使用 SigV4 來簽署建立連線的請求。這種雙向通訊允許交互式 bash 並允許 PowerShell 存取受管節點。您也可以使用 AWS Key Management Service (AWS KMS) 金鑰進一步加密超出預設 TLS 加密的資料。

例如，假設 John 是一位在您的 IT 部門值班的工程師。他接收一個事件通知，需要他遠端連接到受管節點，例如需要故障排除的執行失敗或直接在節點上變更簡單的組態選項。使用 AWS Systems Manager 主控台、Amazon EC2 主控台或 AWS CLI，John 會啟動將他連接到受管節點的工作階段，在完成任務所需的節點上執行命令，然後結束工作階段。

當 John 傳送第一個命令來開始工作階段時，Session Manager 服務會驗證他的 ID、驗證 IAM 政策給予他的許可全縣，檢查組態設定 (例如，驗證工作階段的許可限制) 和傳送訊息到 SSM Agent 以開啟雙向連線。在連線建立和 John 樹入下一個命令後、從 SSM Agent 上輸出並命令且傳到此通訊通道然後傳回他的本機電腦。

主題

- 設定 Session Manager
- 使用 Session Manager
- 記錄工作階段活動
- 啟用和停用工作階段記錄
- 工作階段文件結構描述
- Session Manager 疑難排解

設定 Session Manager

使用 AWS Systems Manager Session Manager 連線到帳戶中的受管節點之前，請完成下列主題中的步驟。

主題

- 步驟 1：完成Session Manager事前準備
- 步驟 2：為 Session Manager 確認或新增執行個體許可
- 步驟 3：控制工作階段對受管節點的存取權
- 步驟 4：進行工作階段偏好設定
- 步驟 5：(選用) 限制對工作階段中命令的存取
- 步驟 6：(選用) 使用 AWS PrivateLink 設定 Session Manager 的 VPC 端點
- 步驟 7：(選用) 啟用或停用 ssm-user 帳戶管理許可
- 步驟 8：(選用) 透過 Session Manager 允許和控制 SSH 連線的許可

步驟 1：完成Session Manager事前準備

在您使用 Session Manager 之前，請確定您的環境符合下列要求。

Session Manager 先決條件

需求	描述
支援的作業系統	在使用 advanced-instance 方案的混合多雲端環境中，Session Manager 支援連線至 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體和伺服器或虛擬機器 (VM)。 Session Manager 支援以下作業系統版本： **Note** 在使用 advanced-instance 方案的混合多雲端環境中，Session Manager 支援 EC2 執行個體、邊緣設備及內部部署伺服器和虛擬機器 (VM)。如需進階執行個體的詳細資訊，請參閱 設定執行個體方案。 Linux 和 macOS Session Manager 支援 支援的所有 Linux和 macOS 版本 AWS Systems Manager。如需相關資訊，請參閱 支援的作業系統和機器類型。 Windows Session Manager 支援 Windows Server 2012 至 Windows Server 2022。 **Note** 不支援 Microsoft Windows Server 2016 Nano。

需求	描述
SSM Agent	至少， AWS Systems Manager SSM Agent.2.3.680 版或更新版本必須安裝在您想要透過工作階段連線的受管節點上。 若要使用 選項，使用在 AWS Key Management Service (AWS KMS) 中建立的金鑰來加密工作階段資料， SSM Agent 2.3.539.0 版或更新版本必須安裝在受管節點上。 若要在工作階段中使用 shell 描述檔，必須在受管節點上安裝 SSM Agent 3.0.161.0 版或更新版本。 若要啟動 Session Manager 網路埠轉送或 SSH 工作階段，必須在受管節點上安裝 SSM Agent 3.0.222.0 版或更新版本。 若要使用 Amazon CloudWatch Logs 來串流工作階段資料，必須在受管節點上安裝 SSM Agent 3.0.284.0 版或更新版本。 如需如何判斷在執行個體上執行的版本號的詳細資訊，請參閱 檢查 SSM Agent 版本編號。如需手動安裝或自動更新 SSM Agent 的詳細資訊，請參閱 使用 SSM Agent。 關於 ssm 使用者帳戶 從 SSM Agent 2.3.50.0 版開始，代理程式使用名為 `ssm-user` 的根或管理員權限，在受管節點上建立使用者帳戶。(在 2.3.612.0 之前的版本中，當 SSM Agent 啟動或重新啟動時會建立帳戶。在版本 2.3.612.0 和更新版本中，在受管節點上第一次啟動工作階段時會建立 `ssm-user`。) 透過使用者帳戶的管理登入資料來啟動工作階段。如果有關限制此帳戶的管理控制的

需求	描述
	資訊，請參閱停用或啟用 ssm-user 帳戶管理許可。 Windows Server 網域控制站上的 ssm 使用者 從 SSM Agent 2.3.612.0 版開始，系統不會在用作為 Windows Server 網域控制站的受管節點上自動建立 `ssm-user` 帳戶。若要在被用作網域控制器的 Windows Server 機器上使用 Session Manager，您必須手動建立 `ssm-user` 帳戶 (如果尚不存在)，並將網域管理員許可指派給使用者。每次工作階段啟動時，SSM Agent 會在 Windows Server 上為 `ssm-user` 帳戶設定新的密碼，因此您在建立帳戶時不需要指定密碼。
連線至端點	您連線的受管節點也必須允許 HTTPS (連接埠 443) 傳出流量至下列端點： • ec2messages.*region*.amazonaws.com • ssm.*region*.amazonaws.com • ssmmessages.*region*.amazonaws.com 如需詳細資訊，請參閱下列主題： • 參考：ec2messages、ssmmessages 和其他 API 操作 • 如何在 知識中心建立 VPC 端點，以便使用 Systems Manager 管理私有 EC2 執行個體而不存取網際網路？ AWS re:Post 或者，您可以使用介面端點連線至所需的端點。如需詳細資訊，請參閱步驟 6：(選用) 使用 AWS PrivateLink 設定 Session Manager 的 VPC 端點。

需求	描述
AWS CLI	（選用）如果您使用 AWS Command Line Interface (AWS CLI) 來啟動工作階段（而不是使用 AWS Systems Manager 主控台或 Amazon EC2 主控台），則必須在本機機器上安裝 CLI 的 1.16.12 版或更新版本。 您可以呼叫 `aws --version` 來檢查版本。 如果您需要安裝或升級 CLI，請參閱 AWS Command Line Interface 《使用者指南》中的安裝 AWS Command Line Interface 。 **Important** 每當將新工具新增至 Systems Manager 或對現有工具進行更新時，SSM Agent 就會發行更新的 版本。未使用最新版本的代理程式可能會使您的受管節點無法使用各種 Systems Manager 工具和功能。因此，我們建議您讓機器的 SSM Agent 自動保持最新狀態。如需相關資訊，請參閱 自動化 SSM Agent 更新。請訂閱 GitHub 上的 SSM Agent 版本備註頁面，以便接收有關 SSM Agent 更新的通知。 除此之外，透過 Session Manager 使用 CLI 來管理您的節點，您必須先安裝 Session Manager 外掛程式本機電腦。如需相關資訊，請參閱 安裝 的Session Manager外掛程式 AWS CLI。

需求	描述
開啟 advanced-instance 方案 (混合多雲端環境)	若要使用 連線到non-EC2 機器Session Manager，您必須開啟 中的進階執行個體層 ， AWS 帳戶 並在 AWS 區域 其中建立混合啟用，將non-EC2 機器註冊為受管節點。使用 advanced-instance 方案會產生費用。如需 advanced-instance 方案的詳細資訊，請參閱 設定執行個體方案。
確認 IAM 服務角色許可 (混合多雲端環境)	混合啟用節點使用混合啟用中指定的 AWS Identity and Access Management (IAM) 服務角色，與 Systems Manager API 操作通訊。此服務角色必須包含使用 Session Manager 連線至混合多雲端機器所需的許可。如果您的服務角色包含 AWS 受管政策 `AmazonSSMManagedInstanceCore` ，Session Manager則已提供 所需的許可。 如果發現服務角色不包含必要的許可，則您必須取消註冊受管執行個體，並使用具有所需許可之 IAM 服務角色的新混合式啟用來註冊該執行個體。如需取消註冊受管執行個體的詳細資訊，請參閱 取消註冊混合多雲端環境中的受管節點。如需建立具有 Session Manager 許可之 IAM 政策的詳細資訊，請參閱步驟 2：為 Session Manager 確認或新增執行個體許可。

步驟 2：為 Session Manager 確認或新增執行個體許可

在預設情況下， AWS Systems Manager 沒有在您的執行個體上執行動作的許可。您可以使用 AWS Identity and Access Management (IAM) 角色在帳戶層級提供執行個體許可，或使用執行個體設定檔在執行個體層級提供許可。如果您的使用案例允許，建議您使用預設主機管理組態在帳戶層級授予存取權。如果您已經使用 `AmazonSSMManagedEC2InstanceDefaultPolicy` 政策設定帳戶的預設主機管理組態，則可以執行下一個步驟。如需有關預設主機管理組態的詳細資訊，請參閱 使用預設主機管理組態來自動管理 EC2 執行個體。

您也可以使用執行個體設定檔為執行個體提供必要的許可。使用執行個體設定檔將 IAM 角色傳遞給 Amazon EC2 執行個體。您可以將 IAM 執行個體設定檔連接至啟動的 Amazon EC2 執行個體或先前啟動的執行個體。如需詳細資訊，請參閱使用執行個體設定檔。

對於內部部署伺服器或虛擬機器 (VM)，許可由與混合式啟用相關聯的 IAM 服務角色提供，該角色可用來向 Systems Manager 註冊您的內部部署伺服器和 VM。內部部署伺服器和 VM 不使用執行個體設定檔。

如果您已使用其他 Systems Manager 工具，例如 Run Command或 Parameter Store，則具有 必要基本許可的執行個體描述檔可能Session Manager已連接至您的 Amazon EC2 執行個體。如果包含 AWS 受管政策的執行個體設定檔`AmazonSSMManagedInstanceCore`已連接至您的執行個體，Session Manager則已提供 所需的許可。如果混合式啟用中使用的 IAM 服務角色包含 `AmazonSSMManagedInstanceCore` 管理的政策，則這也會是 True。

但是，在某些案例中，您可能需要修改連接到您執行個體設定檔的許可。例如，您想要提供一組更窄的執行個體許可、為執行個體描述檔建立自訂政策，或是想要使用 Amazon Simple Storage Service (Amazon S3) 加密或 AWS Key Management Service (AWS KMS) 加密選項來保護工作階段資料。針對這些案例，請執行以下其中一項作業，允許在您的執行個體上執行 Session Manager 動作：

- 在自訂 IAM 角色中為 Session Manager 動作內嵌許可

  若要將Session Manager動作的許可新增至現有 IAM 角色，而該角色不依賴 AWS提供的預設政策 `AmazonSSMManagedInstanceCore`，請遵循 中的步驟將 Session Manager 許可新增至現有 IAM 角色。
- 建立只有 Session Manager 許可的自訂 IAM 角色

  若要建立只有包含 Session Manager 動作許可在 IAM 角色中，請依照 建立 Session Manager 的自訂 IAM 角色 中的步驟操作。
- 建立和使用新的 IAM 角色，該角色具有對所有 Systems Manager 動作的許可

  若要為使用 提供的預設政策 AWS 授予所有 Systems Manager 許可的 Systems Manager 受管執行個體建立 IAM 角色，請遵循設定 Systems Manager 所需的執行個體許可中的步驟。

主題

- 將 Session Manager 許可新增至現有 IAM 角色
- 建立 Session Manager 的自訂 IAM 角色

將 Session Manager 許可新增至現有 IAM 角色

使用以下程序將 Session Manager 許可新增至現有 AWS Identity and Access Management (IAM) 角色。透過將許可新增至現有角色，您可以增強運算環境的安全性，而不必將 AWS `AmazonSSMManagedInstanceCore`政策用於執行個體許可。

> **Note**
>
> 記下以下資訊：
>
> - 這個程序假設您現有的角色已經包含其他您希望允許存取動作的 Systems Manager `ssm` 許可。這個政策無法獨立使用 Session Manager。
> - 下列政策範例包含一個 `s3:GetEncryptionConfiguration` 動作。如果您在 Session Manager 記錄偏好設定中選擇了強制 S3 日誌加密選項，則需要執行此動作。

將 Session Manager 許可新增至現有角色 (主控台)

1. 登入 AWS Management Console ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇角色。
3. 選取您要為其新增許可的角色名稱。
4. 選擇許可索引標籤標籤。
5. 選擇新增許可，然後選取建立內嵌政策。
6. 選擇 JSON 標籤。
7. 將預設政策內容取代為以下內容。將 *`key-name`* 取代為您要使用的 AWS Key Management Service 金鑰 () 的 Amazon Resource Name (ARN AWS KMS key)。

```
{
    "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": "key-name"
        }
    ]
}
```

如需使用 KMS 金鑰來加密工作階段資料的詳細資訊，請參閱 開啟工作階段資料的 KMS 金鑰加密 (主控台)。

如果您不對工作階段資料使用 AWS KMS 加密，您可以從政策中移除下列內容。

```
,
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "key-name"
        }
```

8. 選擇下一步：標籤。
9. (選用) 透過選擇 Add tag (新增標籤)，然後輸入政策的首選標籤來新增標籤。
10. 選擇下一步：檢閱。
11. 在檢閱政策頁面上名稱中，輸入該內嵌政策的名稱，例如 **SessionManagerPermissions**。
12. (選用) 在說明中輸入政策的說明。

    選擇 建立政策。

如需有關 ssmmessages 動作的資訊，請參閱 參考：ec2messages、ssmmessages 和其他 API 操作。

建立 Session Manager 的自訂 IAM 角色

您可以建立 AWS Identity and Access Management (IAM) 角色，授予Session Manager許可，以對 Amazon EC2 受管執行個體執行動作。您也可以包含一項政策，以授予將工作階段日誌傳送到 Amazon Simple Storage Service (Amazon S3) 和 Amazon CloudWatch Logs 所需的許可。

建立 IAM 角色後，如需如何將角色連接至執行個體的資訊，請參閱 AWS re:Post 網站上的連接或取代執行個體設定檔。如需詳有關 IAM 執行個體設定檔和角色的細資訊，請參閱《IAM 使用者指南》中的使用執行個體描述檔一節，以及《適用於 Linux 的 Amazon Elastic Compute Cloud 使用者指南》中的 Amazon EC2 IAM 角色一節。如需有關為內部部署機器建立 IAM 服務角色的詳細資訊，請參閱在混合多雲端環境中建立 Systems Manager 所需的 IAM 服務角色。

主題

- 建立具有最小 Session Manager 許可的 IAM 角色 (主控台)
- 建立具有 Session Manager、Amazon Simple Storage Service (Amazon S3) 和 CloudWatch Logs (主控台) 許可的 IAM 角色

建立具有最小 Session Manager 許可的 IAM 角色 (主控台)

請使用下列處理程序來建立具有政策的自訂 IAM 角色，該政策在您的執行個體上只提供唯一的 Session Manager 動作許可。

建立含最小 Session Manager 許可的執行個體設定檔 (主控台)

1. 登入 AWS Management Console ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇政策，然後選擇建立政策。(顯示 Get Started (開始使用) 按鈕時先選擇它，然後選擇 Create Policy (建立政策)。)
3. 選擇 JSON 標籤。
4. 將預設內容取代為以下政策。若要使用 AWS Key Management Service (AWS KMS) 加密工作階段資料，請以您要使用的 Amazon Resource Name AWS KMS key (ARN) 取代 *key-name*。

```
{
    "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": "key-name"
        }
    ]
}
```

如需使用 KMS 金鑰來加密工作階段資料的詳細資訊，請參閱 開啟工作階段資料的 KMS 金鑰加密 (主控台)。

如果您不對工作階段資料使用 AWS KMS 加密，您可以從政策中移除下列內容。

```
,
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "key-name"
        }
```

5. 選擇下一步：標籤。
6. (選用) 透過選擇 Add tag (新增標籤)，然後輸入政策的首選標籤來新增標籤。
7. 選擇下一步：檢閱。
8. 在檢閱政策頁面上名稱中，輸入該內嵌政策的名稱，例如 **SessionManagerPermissions**。
9. (選用) 在說明中輸入政策的說明。
10. 選擇 建立政策。

11. 在導覽窗格中，選擇角色，然後選擇建立角色。
12. 在 Create role (建立角色) 頁面上，選擇 AWS service (服務)，針對 Use case (使用案例)，選擇 EC2。
13. 選擇下一步。
14. 在 Add permissions (新增許可) 頁面，選取您剛建立政策左側的核取方塊，例如 **SessionManagerPermissions**。
15. 選擇下一步。
16. 在 Name, review, and create (名稱、檢閱和建立) 頁面的 Role name (角色名稱) 中，輸入 IAM 角色的名稱，例如 **MySessionManagerRole**。
17. (選用) Role description (角色說明)，輸入執行個體設定檔的說明。
18. (選用) 透過選擇 Add tag (新增標籤)，然後輸入角色的首選標籤來新增標籤。

    選擇建立角色。

如需有關 `ssmmessages` 動作的資訊，請參閱 參考：ec2messages、ssmmessages 和其他 API 操作。

建立具有 Session Manager、Amazon Simple Storage Service (Amazon S3) 和 CloudWatch Logs (主控台) 許可的 IAM 角色

請使用下列處理程序來建立具有政策自訂 IAM 角色，該政策在您的執行個體上提供 Session Manager 動作許可。此政策還提供在 Amazon Simple Storage Service (Amazon S3) 儲存貯體和 Amazon CloudWatch Logs 日誌群組中存放工作階段日誌的所需許可。

> **Important**
>
> 若要將工作階段日誌輸出到不同 AWS 帳戶所擁有的 Amazon Simple Storage Service (Amazon S3) 儲存貯體，您必須將 `s3:PutObjectAcl` 許可新增至該 IAM 角色政策。此外，您必須確保該儲存貯體政策將跨帳戶存取權授予擁有該儲存貯體的帳戶，用於授予受管執行個體 Systems Manager 許可所使用的 IAM 角色。如果該儲存貯體使用 Key Management Service (KMS) 加密，則該儲存貯體的 KMS 政策也必須授予此跨帳戶存取權。如需有關在 Amazon S3 中設定跨帳戶儲存貯體許可的更多資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的授予跨帳戶儲存貯體許可一節。如果未新增此許可，則擁有該 Amazon Simple Storage Service (Amazon S3) 儲存貯體的帳戶無法存取工作階段輸出日誌。

如需有關指定儲存工作階段日誌偏好的更多資訊，請參閱啟用和停用工作階段記錄。

建立具有 Session Manager、Amazon Simple Storage Service (Amazon S3) 和 CloudWatch Logs (主控台) 許可的 IAM 角色

1. 登入 AWS Management Console ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇政策，然後選擇建立政策。(顯示 Get Started (開始使用) 按鈕時先選擇它，然後選擇 Create Policy (建立政策)。)
3. 選擇 JSON 標籤。
4. 將預設內容取代為以下政策。將每個########取代為您自己的資訊。

```
{
    "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": "key-name"
        },
        {
            "Effect": "Allow",
            "Action": "kms:GenerateDataKey",
            "Resource": "*"
        }
    ]
}
```

5. 選擇下一步：標籤。
6. (選用) 透過選擇 Add tag (新增標籤)，然後輸入政策的首選標籤來新增標籤。
7. 選擇下一步：檢閱。
8. 在檢閱政策頁面上名稱中，輸入該內嵌政策的名稱，例如 **SessionManagerPermissions**。
9. (選用) 在說明中輸入政策的說明。
10. 選擇 建立政策。
11. 在導覽窗格中，選擇角色，然後選擇建立角色。
12. 在 Create role (建立角色) 頁面上，選擇 AWS service (服務)，針對 Use case (使用案例)，選擇 EC2。
13. 選擇下一步。
14. 在 Add permissions (新增許可) 頁面，選取您剛建立政策左側的核取方塊，例如 **SessionManagerPermissions**。
15. 選擇下一步。
16. 在 Name, review, and create (名稱、檢閱和建立) 頁面的 Role name (角色名稱) 中，輸入 IAM 角色的名稱，例如 **MySessionManagerRole**。
17. (選用) 在 Role description (角色說明) 中，輸入角色的說明。
18. (選用) 透過選擇 Add tag (新增標籤)，然後輸入角色的首選標籤來新增標籤。

19. 選擇建立角色。

步驟 3：控制工作階段對受管節點的存取權

您可以使用 AWS Identity and Access Management (IAM) 政策授予或撤銷受管節點的Session Manager存取權。您可以建立政策並將其連接到某個 IAM 使用者或群組，以指定該使用者或群組可連線到哪些受管節點。您也可以指定該使用者或群組可在這些受管節點上執行的 Session Manager API 操作。

為了協助您開始使用 Session Manager 的 IAM 許可政策，我們建立了適用於最終使用者和管理員使用者的範例政策。您只需對這些政策稍做變更便可加以利用。您也可以將它們用作建立自訂 IAM 政策時的指南。如需詳細資訊，請參閱適用於 Session Manager 的範例 IAM 政策。如需有關如何建立 IAM 政策並將其連接至使用者或群組的相關資訊，請參閱《IAM 使用者指南》中的建立 IAM 政策和新增和移除 IAM 政策。

關於工作階段 ID ARN 格式

為 Session Manager 存取權建立 IAM 政策時，您需要將工作階段 ID 指定為 Amazon Resource Name (ARN) 的一部分。該工作階段 ID 包含使用者名稱做為變數。為了更好地說明，以下是 Session Manager ARN 的格式和一個範例：

```
arn:aws:ssm:region-id:account-id:session/session-id
```

例如：

```
arn:aws:ssm:us-east-2:123456789012:session/JohnDoe-1a2b3c4d5eEXAMPLE
```

如需有關在 IAM 政策中使用變數的詳細資訊，請參閱 IAM 政策元素：變數。

主題

- 透過在 IAM 政策中指定預設的工作階段文件來啟動預設 Shell 工作階段
- 透過在 IAM 政策中指定工作階段文件使用文件啟動預工作階段
- 適用於 Session Manager 的範例 IAM 政策
- Session Manager 的其他 IAM 政策範例

透過在 IAM 政策中指定預設的工作階段文件來啟動預設 Shell 工作階段

當您Session Manager為 設定 AWS 帳戶 或在 Systems Manager 主控台中變更工作階段偏好設定時，系統會建立名為 的 SSM 工作階段文件SSM-SessionManagerRunShell。這是預設的工作階段文件。Session Manager 使用此文件來儲存您的工作階段偏好設定，其中包含下列資訊：

- 您想要儲存工作階段資料的位置，例如 Amazon Simple Storage Service (Amazon S3) 儲存貯體或 Amazon CloudWatch Logs 日誌群組。
- 用於加密工作階段資料的 AWS Key Management Service (AWS KMS) 金鑰 ID。
- 您的工作階段是否允許「執行身分」支援。

以下是 SSM-SessionManagerRunShell 工作階段偏好設定文件中包含的資訊範例。

```
{
  "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"
  }
}
```

依預設，當使用者從 AWS Management Console啟動工作階段時，Session Manager 會使用預設的工作階段文件。這適用於 Systems Manager 主控台Session Manager中的 Fleet Manager或 ，或 Amazon EC2 主控台中的 EC2 Connect。 Session Manager 也會在使用者使用如下範例的 AWS CLI 命令啟動工作階段時使用預設工作階段文件： Amazon EC2

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

若要啟動預設的 Shell 工作階段，必須在 IAM 政策中指定預設工作階段文件，如下列範例所示。

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "EnableSSMSession",
      "Effect": "Allow",
      "Action": [
        "ssm:StartSession"
      ],
      "Resource": [
        "arn:aws:ec2:us-west-2:123456789012:instance/i-02573cafcfEXAMPLE",
        "arn:aws:ssm:us-west-2:123456789012:document/SSM-SessionManagerRunShell"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "ssmmessages:OpenDataChannel"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

透過在 IAM 政策中指定工作階段文件使用文件啟動預工作階段

如果您使用 start-session AWS CLI 命令及預設工作階段文件，則可以省略文件名稱。系統會自動呼叫 SSM-SessionManagerRunShell 工作階段文件。

在其他所有情況下，您必須指定 document-name 參數值。當使用者在命令中指定工作階段文件的名稱時，系統會檢查其 IAM 政策，以確認其有權存取該文件。如果使用者沒有許可，連線請求就會失敗。下列範例包含 AWS-StartPortForwardingSession 工作階段文件的 document-name 參數。

```
aws ssm start-session \
    --target i-02573cafcfEXAMPLE \
    --document-name AWS-StartPortForwardingSession \
    --parameters '{"portNumber":["80"], "localPortNumber":["56789"]}'
```

如需如何在 IAM 政策中指定 Session Manager 工作階段文件的範例，請參閱 Session Manager的最終使用者政策快速入門。

> **Note**
>
> 若要使用 SSH 來啟動工作階段，必須在目標受管節點和使用者的本機電腦上都完成設定步驟。如需相關資訊，請參閱 (選用) 透過 Session Manager 允許和控制 SSH 連線的許可。

適用於 Session Manager 的範例 IAM 政策

使用本節中的範例來協助您建立 AWS Identity and Access Management (IAM) 政策，以提供最常用的 Session Manager存取許可。

> **Note**
>
> 您也可以使用 AWS KMS key 政策來控制哪些 IAM 實體 （使用者或角色） 和 AWS 帳戶 有權存取您的 KMS 金鑰。如需詳細資訊，請參閱《 AWS Key Management Service 開發人員指南》中的管理 AWS KMS 資源存取和使用金鑰政策的概觀。 AWS KMS

主題

- Session Manager的最終使用者政策快速入門
- Session Manager的管理者政策快速入門

Session Manager的最終使用者政策快速入門

使用以下範例來替 Session Manager 建立 IAM 最終使用者政策。

您可以建立政策，允許使用者只從Session Manager主控台和 AWS Command Line Interface (AWS CLI)、只從 Amazon Elastic Compute Cloud (Amazon EC2) 主控台或從這三個主控台啟動工作階段。

這些政策提供最終使用者對特定受管節點啟動工作階段的能力，也能夠結束自己的工作階段。請參閱Session Manager 的其他 IAM 政策範例取得有關您想在政策進行自訂的範例

在以下範例政策中，將每個########取代為您自己的資訊。

參閱以下各節，來針對您要提供的工作階段存取範圍檢視範例政策。

工作階段管理員 and Fleet Manager

使用此範例政策，讓使用者只能從 Session Manager 與 Fleet Manager 主控台啟動和繼續工作階段。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ec2:region:account-id:instance/instance-id",
                "arn:aws:ssm:region:account-id:document/SSM-
SessionManagerRunShell" (1)
            ]
        },
        {
         "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" 2
            ],
            "Resource": "key-name"
        }
    ]
}
```

Amazon EC2

使用此範例政策，來讓使用者只能從 Amazon EC2 主控台啟動和繼續工作階段。這個政策不提供從 Session Manager 主控台和 AWS CLI啟動工作階段所需要的所有許可。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession",

 "ssm:SendCommand" 3
            ],
            "Resource": [
                "arn:aws:ec2:region:account-id:instance/instance-id",
                "arn:aws:ssm:region:account-id:document/SSM-
SessionManagerRunShell" 1
            ]
        },
        {
         "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

使用此範例政策，讓使用者可以從 AWS CLI啟動和繼續工作階段。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession",

 "ssm:SendCommand" 3
            ],
            "Resource": [
                "arn:aws:ec2:region:account-id:instance/instance-id",
                "arn:aws:ssm:region:account-id:document/SSM-
SessionManagerRunShell" 1
            ]
        },
        {
         "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" 2
            ],
            "Resource": "key-name"
        }
    ]
}
```

[1] SSM-SessionManagerRunShell 是 SSM 文件的預設名稱，Session Manager 會建立該 SSM 文件來存放您的工作階段組態偏好設定。您可以建立自訂工作階段文件，並改在這個政策中進行指定。您也可以AWS-StartSSHSession為使用 SSH 啟動工作階段的使用者指定 AWS提供的文件。如需有關使用 SSH 支援工作階段所需的設定步驟的資訊，請參閱 (選用) 透過 Session Manager 允許和控制 SSH 連線的許可。

[2] kms:GenerateDataKey 許可讓您能夠建立加密工作階段資料所用的資料加密金鑰。如果您要對工作階段資料使用 AWS Key Management Service (AWS KMS) 加密，請將 *key-name* 取代為您要使用之 KMS 金鑰的 Amazon Resource Name (ARN)，格式為 arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-12345EXAMPLE。如果您不使用 KMS 金鑰來加密工作階段資料，請將以下內容從政策中移除：

```
{
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey"
            ],
            "Resource": "key-name"
        }
```

如需使用 AWS KMS 加密工作階段資料的相關資訊，請參閱 開啟工作階段資料的 KMS 金鑰加密 (主控台)。

[3] 當使用者嘗試從 Amazon EC2 主控台啟動工作階段時，需要 SendCommand 許可，但必須先將 SSM Agent 更新至 Session Manager 所需的最低版本。Run Command 用於將命令傳送至執行個體以更新代理程式。

Session Manager的管理者政策快速入門

使用以下範例來替 Session Manager 建立 IAM 管理員政策。

這些政策提供管理員能夠啟動工作階段到被 `Key=Finance,Value=WebServers` 所標記的受管節點的能力，建立、更新和刪除偏好設定的許可，以及僅結束自己工作階段的許可。請參閱Session Manager 的其他 IAM 政策範例取得有關您想在政策進行自訂的範例

您可以建立政策，允許管理員只從Session Manager主控台執行這些任務 AWS CLI，以及只從 Amazon EC2 主控台執行，或從這三個主控台執行這些任務。

在以下範例政策中，將每個*########*取代為您自己的資訊。

請參閱以下章節來檢視三種許可案例的範例政策。

工作階段管理員 and CLI

使用此範例政策，來讓管理員只能從 Session Manager 主控台與 AWS CLI執行與工作階段相關的任務。這個政策不提供從 Amazon EC2 主控台執行與工作階段相關的任務所需要的所有許可。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ec2:region:account-id:instance/*"
            ],
            "Condition": {
                "StringLike": {
                    "ssm:resourceTag/Finance": [
                        "WebServers"
                    ]
                }
            }
        },
```

```
        {
         "Effect": "Allow",
         "Action": ["ssmmessages:OpenDataChannel"],
         "Resource": ["arn:aws:ssm:*:*:session/${aws:userid}-*"]
       },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeSessions",
                "ssm:GetConnectionStatus",
                "ssm:DescribeInstanceProperties",
                "ec2:DescribeInstances"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:CreateDocument",
                "ssm:UpdateDocument",
                "ssm:GetDocument",
                "ssm:StartSession"
            ],
            "Resource": "arn:aws:ssm:region:account-id:document/SSM-
SessionManagerRunShell"
        },
        {
         "Effect": "Allow",
         "Action": ["ssmmessages:OpenDataChannel"],
         "Resource": ["arn:aws:ssm:*:*:session/${aws:userid}-*"]
       },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:TerminateSession",
                "ssm:ResumeSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:userid}-*"
            ]
        }
    ]
}
```

Amazon EC2

使用此範例政策，來讓管理員只能從 Amazon EC2 主控台執行與工作階段相關的任務。這個政策不提供從 Session Manager 主控台與 AWS CLI執行與工作階段相關的任務所需要的所有許可。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession",

 "ssm:SendCommand" 1
            ],
            "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}-*"
            ]
        }
    ]
}
```

工作階段管理員, CLI, and Amazon EC2

使用此範例政策，來讓管理員能從 Session Manager 主控台、 AWS CLI與 Amazon EC2 主控台執行與工作階段相關的任務。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession",

 "ssm:SendCommand" (1)
            ],
            "Resource": [
                "arn:aws:ec2:region:account-id:instance/*"
            ],
            "Condition": {
                "StringLike": {
                    "ssm:resourceTag/tag-key": [
                        "tag-value"
                    ]
                }
            }
```

```
        },
        {
         "Effect": "Allow",
         "Action": ["ssmmessages:OpenDataChannel"],
         "Resource": ["arn:aws:ssm:*:*:session/${aws:userid}-*"]
       },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeSessions",
                "ssm:GetConnectionStatus",
                "ssm:DescribeInstanceInformation",
                "ssm:DescribeInstanceProperties",
                "ec2:DescribeInstances"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:CreateDocument",
                "ssm:UpdateDocument",
                "ssm:GetDocument",
                "ssm:StartSession"
            ],
            "Resource": "arn:aws:ssm:region:account-id:document/SSM-
SessionManagerRunShell"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:TerminateSession",
                "ssm:ResumeSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:userid}-*"
            ]
        }
    ]
}
```

[1] 如果有使用者嘗試從 Amazon EC2 主控台啟動工作階段，但必須先傳送命令來更新 SSM Agent 的話，就需要 SendCommand 的許可。

Session Manager 的其他 IAM 政策範例

請參閱下列範例政策，以協助您建立自訂 AWS Identity and Access Management (IAM) 政策以任何 Session Manager 使用者存取的情境。

主題

- 範例 1：在主控台中授予對文件的存取權
- 範例 2：限制對特定受管節點的存取權
- 範例 3：根據標籤限制受管節點存取
- 範例 4：只允許使用者結束他們啟動的工作階段
- 範例 5：允許完整 (管理) 存取所有工作階段

範例 1：在主控台中授予對文件的存取權

您可以允許使用者在使用 Session Manager 主控台啟動工作階段時指定自訂文件。下列 IAM 政策範例授予對指定 AWS 區域 和 AWS 帳戶中名稱以 **SessionDocument-** 開頭的存取文件的許可。

若要使用此政策，請使用您自己的資訊取代*########*。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetDocument",
                "ssm:ListDocuments"
            ],
            "Resource": [
                "arn:aws:ssm:region:account-id:document/SessionDocument-*"

            ]
        }
    ]
}
```

> **Note**
>
> Session Manager 主控台僅支援 `sessionType` 為 `Standard_Stream` 的工作階段文件 (用於定義工作階段偏好設定)。如需詳細資訊，請參閱工作階段文件結構描述。

範例 2：限制對特定受管節點的存取權

您可以建立 IAM 政策，以定義允許使用者使用 Session Manager 連線到哪些受管節點。例如，下列政策授予使用者在三個特定節點上啟動、結束和繼續其工作階段的許可。此政策會限制使用者連線至指定節點以外的節點。

> **Note**
>
> 如需聯合身分使用者，請參閱 範例 4：只允許使用者結束他們啟動的工作階段。

```
{
    "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}-*"]
       }
    ]
}
```

範例 3：根據標籤限制受管節點存取

您可以根據特定標籤限制對受管節點的存取權。在下列範例中，使用者可以在任何受管節點 (Resource: arn:aws:ec2:*region*:*987654321098*:instance/*) 開始及恢復工作階段 (Effect: Allow, Action: ssm:StartSession, ssm:ResumeSession) 節點的條件是 Finance WebServer (ssm:resourceTag/Finance: WebServer)。如果使用者將命令傳送至未加上標籤或有 Finance: WebServer 以外任何標籤的受管節點，執行結果會顯示 AccessDenied。

```
{
    "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"
            ]
        }
    ]
}
```

您可以建立 IAM 政策，可讓使用者在已多個標籤標記的受管節點上執行命令。以下政策可讓使用者在有這兩種特定標籤的受管節點啟工作階段。如果使用者將命令傳送至未以那兩個標籤標記的受管節點，命令執行結果會包含 AccessDenied。

```
{
   "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"
            ]
      }
   ]
}
```

如需有關建立 IAM 政策的詳細資訊，請參閱《IAM 使用者指南》中的受管政策和內嵌政策。如需有關標記受管節點的詳細資訊，請參閱《Amazon EC2 使用者指南》中的標記您的 Amazon EC2 資源 (內容適用於 Windows 和 Linux 受管節點)。如需提升安全狀態，防範受管節點上未獲授權的根層級命令的詳細資訊，請參閱 限制透過 SSM Agent 存取根層級命令

範例 4：只允許使用者結束他們啟動的工作階段

Session Manager 提供兩種方法來控制 中聯合身分使用者 AWS 帳戶 允許結束的工作階段。

- 在 AWS Identity and Access Management (IAM) 許可政策`{aws:userid}`中使用 變數。聯合身分使用者只能結束他們啟動的工作階段。對於非聯合身分使用者，請使用方法 1。對於聯合身分使用者，請使用方法 2。
- 使用 IAM 許可政策中標籤提供的 AWS 標籤。在政策中，您會包含一個條件，允許使用者只結束使用 AWS提供之特定標籤標記的工作階段。此方法適用於所有帳戶，包括使用聯合身分 ID 授與 AWS 存取權的帳戶。

方法 1：使用變數 **{aws:username}** 授與 TerminateSession 權限

下列 IAM 政策可讓使用者檢視您帳戶中所有工作階段的 ID。不過，使用者只能透過他們啟動的工作階段與受管節點互動。下列政策所指派到的使用者無法連結或結束其他使用者的工作階段。使用變數 `{aws:username}` 來達成目標的政策。

> **Note**
>
> 此方法不適用於使用聯合身分 ID 授與 AWS 存取權的帳戶。

```
{
    "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提供的標籤授與 TerminateSession 權限

您可以在 IAM 政策中包含條件標籤索引鍵變數，來控制使用者可以結束哪些工作階段。條件會指定使用者只能結束以一個或兩個特定標籤鍵變數和指定值加以標記的工作階段。

當您 中的使用者 AWS 帳戶 啟動工作階段時， 會將兩個資源標籤Session Manager套用至工作階段。第一個資源標籤是 `aws:ssmmessages:target-id`，您可以用來指定允許使用者結束之目標的

ID。另一個資源標籤是 aws:ssmmessages:session-id，具有此格式的值：*role-id*:*caller-specified-role-name*。

> **Note**
>
> Session Manager 不支援此 IAM 存取控制政策的自訂標籤。您必須使用 提供的資源標籤 AWS，如下所述。

aws:ssmmessages:target-id

使用此標籤鍵，您可以將受管節點 ID 作為值納入政策。在下列政策區塊中，條件陳述式可讓使用者只結束節點 i-02573cafcfEXAMPLE。

```
{
     "Version": "2012-10-17",
     "Statement": [
         {
             "Effect": "Allow",
             "Action": [
                "ssm:TerminateSession"
             ],
             "Resource": "*",
             "Condition": {
                 "StringLike": {
                     "ssm:resourceTag/aws:ssmmessages:target-id": [
                        "i-02573cafcfEXAMPLE"
                     ]
                 }
             }
         }
     ]
}
```

如果使用者嘗試結束未被授與此 TerminateSession 許可的工作階段，他們會收到 AccessDeniedException 錯誤。

aws:ssmmessages:session-id

此標籤鍵包含工作階段 ID 的變數，做為啟動工作階段之要求中的值。

下列範例會示範發起人類型為 User 的案例政策。您提供給 aws:ssmmessages:session-id 的值是使用者的 ID。在此範例中，AIDIODR4TAW7CSEXAMPLE 代表 AWS 帳戶中使用者的 ID。若要擷取 中使用者的 ID AWS 帳戶，請使用 IAM 命令 get-user。如需詳細資訊，請參閱《IAM 使用者指南 AWS Identity and Access Management 》中的 get-user。

```
{
     "Version": "2012-10-17",
     "Statement": [
         {
             "Effect": "Allow",
             "Action": [
                "ssm:TerminateSession"
             ],
             "Resource": "*",
             "Condition": {
                 "StringLike": {
                     "ssm:resourceTag/aws:ssmmessages:session-id": [
                        "AIDIODR4TAW7CSEXAMPLE"
                     ]
                 }
             }
         }
     ]
}
```

下列範例會示範發起人類型為 AssumedRole 的案例政策。您可以將 {aws:userid} 變數用於您提供給 aws:ssmmessages:session-id 的值。或者，您也可以為您提供給 aws:ssmmessages:session-id 的值硬式編碼角色 ID。如果您硬式編碼角色 ID，則必須以格式 *role-id*:*caller-specified-role-name* 提供值。例如 AIDIODR4TAW7CSEXAMPLE:MyRole。

> **Important**
>
> 為了套用系統標籤，您提供的角色 ID 只能包含下列字元：Unicode 字母、0-9、空格、_、.、:、/、=、+、-、@ 和 \。

若要擷取 中角色的角色 ID AWS 帳戶，請使用 get-caller-identity命令。如需詳細資訊，請參閱《 AWS CLI 命令參考》中的 get-caller-identity。

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "ssm:TerminateSession"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "ssm:resourceTag/aws:ssmmessages:session-id": [
                       "${aws:userid}*"
                    ]
                }
            }
        }
    ]
}
```

如果使用者嘗試結束未被授與此 TerminateSession 許可的工作階段，他們會收到 AccessDeniedException 錯誤。

aws:ssmmessages:target-id 和 aws:ssmmessages:session-id

您也可以建立 IAM 政策，讓使用者結束以兩個系統標籤標記的工作階段，如本範例所示。

```
{
   "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}*"
               ]
            }
```

```
            }
        }
    ]
}
```

範例 5：允許完整 (管理) 存取所有工作階段

以下 IAM 政策可讓使用者完全與所有受管節點和所有使用者所建立在所有節點的所有工作階段互動。它應該只需要授予管理者，管理員需要完全控制組織的 Session Manager 活動。

```
{
    "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}-*"]
       }
    ]
}
```

步驟 4：進行工作階段偏好設定

已在 AWS Identity and Access Management (IAM) 政策中授予管理許可的使用者可以設定工作階段偏好設定，包括下列項目：

- 為 Linux 受管節點開啟執行身分支援。這可讓您使用指定作業系統使用者的登入資料來啟動工作階段，而不是可在受管節點上建立的系統產生`ssm-user`帳戶的 AWS Systems Manager Session Manager登入資料。

- 設定 Session Manager 以使用 AWS KMS key 加密，為用戶端機器與受管節點之間傳輸的資料提供額外的保護。
- 設定 Session Manager 來建立工作階段歷史記錄日誌，並傳送到 Amazon Simple Storage Service (Amazon S3) 儲存貯體或 Amazon CloudWatch Logs 日誌群組。之後，就可以使用儲存的資料來報告與受管節點建立的工作階段連線，以及在工作階段期間對這些節點執行的命令。
- 設定工作階段逾時。您可以使用此設定來指定在閒置一段時間後結束工作階段的時間。
- 設定 Session Manager 以使用可設定的 shell 描述檔。這些可自訂的描述檔可讓您在工作階段內定義偏好設定，例如 shell 偏好設定、環境變數、工作目錄，以及在工作階段啟動時執行的多個命令。

如需設定 Session Manager 偏好設定所需的許可的詳細資訊，請參閱 the section called “授與或拒絕使用者許可來更新Session Manager偏好設定”。

主題

- 授與或拒絕使用者許可來更新Session Manager偏好設定
- 指定閒置工作階段逾時值
- 指定工作階段持續時間上限
- 允許可設定的 Shell 設定檔
- 為 Linux 和 macOS 受管節點開啟執行身分支援
- 開啟工作階段資料的 KMS 金鑰加密 (主控台)
- 建立 Session Manager 偏好設定文件 (命令列)
- 更新 Session Manager 偏好設定 (命令列)

如需如何使用 Systems Manager 主控台設定工作階段資料記錄選項的詳細資訊，請參閱下列主題。

- 使用 Amazon Simple Storage Service (Amazon S3) (主控台) 記錄工作階段資料
- 使用 Amazon CloudWatch Logs (主控台) 串流工作階段資料。
- 使用 Amazon CloudWatch Logs (主控台) 記錄工作階段資料。

授與或拒絕使用者許可來更新Session Manager偏好設定

帳戶偏好設定會儲存為每個 的 AWS Systems Manager (SSM) 文件 AWS 區域。在使用者可以更新帳戶裡工作階段偏好之前，他們必須授與必要的權限來存取這些偏好設定所存放的 SSM 文件。這些許可是透過 AWS Identity and Access Management (IAM) 政策授予。

管理員政策允許建立和更新偏好設定

管理員可以有以下政策在任何時候建立和更新偏好設定。以下政策允許許可存取和更新 SSM-SessionManagerRunShell 文件在 us-east-2 123456789012 帳戶。

```
{
    "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"
            ]
        }
    ]
}
```

使用者政策防止更新偏好設定

使用以下政策防止最終使用者更新或覆寫在您的帳戶中的任何 Session Manager 偏好設定。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ssm:CreateDocument",
                "ssm:GetDocument",
                "ssm:UpdateDocument",
                "ssm:DeleteDocument"
            ],
            "Effect": "Deny",
            "Resource": [
                "arn:aws:ssm:us-east-2:123456789012:document/SSM-
SessionManagerRunShell"
            ]
```

```
        }
    ]
}
```

指定閒置工作階段逾時值

Session Manager中的工具， AWS Systems Manager可讓您指定在系統結束工作階段之前，允許使用者處於非作用中狀態的時間量。根據預設，工作階段在閒置 20 分鐘後逾時。您可以修改此設定，以指定工作階段在閒置 1 到 60 分鐘之間逾時。一些專業的運算安全機構建議將閒置工作階段逾時設定為最長 15 分鐘。

若要允許閒置工作階段逾時 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Session Manager。
3. 選擇 Preferences (偏好) 標籤，然後選擇 Edit (編輯)。
4. 在 Idle session timeout (閒置工作階段逾時) 下的 minutes (分鐘) 欄位中，指定在工作階段結束之前允許使用者處於非作用中狀態的時間長度。
5. 選擇 Save (儲存)。

指定工作階段持續時間上限

Session Manager中的工具， AWS Systems Manager可讓您在工作階段結束前指定工作階段的最長持續時間。依預設，工作階段沒有最長持續時間。您指定的工作階段持續時間上限值必須介於 1 到 1,440 分鐘之間。

指定工作階段持續時間上限 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Session Manager。
3. 選擇 Preferences (偏好) 標籤，然後選擇 Edit (編輯)。
4. 選取 Enable maximum session duration (啟用工作階段持續時間上限) 旁邊的核取方塊。
5. 在 Maximum session duration (工作階段持續時間上限) 下的 minutes (分鐘) 欄位中，指定工作階段結束前的最長持續時間。
6. 選擇 Save (儲存)。

允許可設定的 Shell 設定檔

根據預設，Linux 的 EC2 執行個體上的工作階段開始使用 Bourne shell (sh)。但是，您可能偏好使用像 bash 這樣的其他 shell。透過允許可設定的 shell 描述檔，您可自訂工作階段內的偏好設定，例如 shell 偏好設定、環境變數、工作目錄，以及在工作階段啟動時執行的多個命令。

> **Important**
>
> Systems Manager 不會檢查 shell 描述檔中的命令或指令碼，以查看在執行它們之前會對執行個體進行哪些變更。若要限制使用者修改在其 shell 描述檔中輸入的命令或指令碼的能力，建議執行下列動作：
>
> - 為您的 AWS Identity and Access Management (IAM) 使用者和角色建立自訂工作階段類型文件。然後修改這些使用者和角色的 IAM 政策，以便 `StartSession` API 操作只能使用您為它們建立的工作階段類型文件。如需相關資訊，請參閱 建立 Session Manager 偏好設定文件 (命令列) 及 Session Manager的最終使用者政策快速入門。
> - 修改 IAM 使用者和角色的 IAM 政策，以拒絕對您建立的工作階段類型文件資源的 `UpdateDocument` API 操作。這允許您的使用者和角色使用您為其工作階段偏好設定建立的文件，但不允許他們修改任何設定。

若要開啟可設定的 shell 描述檔

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Session Manager。
3. 選擇 Preferences (偏好) 標籤，然後選擇 Edit (編輯)。
4. 在適用作業系統的欄位中，指定環境變數、shell 偏好設定、或者當您的工作階段啟動時要執行的命令。
5. 選擇 Save (儲存)。

以下是可以新增到 Shell 描述檔的一些命令範例。

變更為 bash shell 並變更為 Linux 執行個體上的 /usr 目錄。

```
exec /bin/bash
cd /usr
```

在工作階段開始時輸出時間戳記和歡迎訊息。

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."
```

在工作階段開始時檢視動態系統活動。

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 "-------  ------    -----      ----- -----   ------     -- -----------"}
```

為 Linux 和 macOS 受管節點開啟執行身分支援

依預設，Session Manager 會使用系統產生之 `ssm-user` 帳戶 (帳戶在受管節點上建立) 的憑證來驗證連線。在 Linux 和 macOS 機器上，帳戶新增到 `/etc/sudoers/`。如果可以選擇，您可以改為使用作業系統 (OS) 使用者帳戶的憑證，或是加入了 Active Directory 的執行個體的網域使用者來驗證工作階

段。在此情況下，Session Manager 會在啟動工作階段之前，驗證您指定的作業系統帳戶是否存在於節點上或網域中。如果您嘗試使用節點上或網域中不存在的作業系統帳戶來啟動工作階段，則連線將會失敗。

> **Note**
>
> Session Manager 不支援使用作業系統的 `root` 使用者帳戶來驗證連線。對於使用作業系統使用者帳戶驗證的工作階段，節點的作業系統層級和目錄政策 (如登入限制或系統資源使用限制) 可能不適用。

運作方式

如果您為工作階段開啟執行身分支援，系統會如下檢查存取許可：

1. 啟動工作階段的使用者的 IAM 實體 (使用者或角色) 是否都已使用 `SSMSessionRunAs = os user account name` 標記？

   如果是，該受管節點上是否存在作業系統使用者名稱？ 如果有，就開始工作階段。如果沒有，則不允許工作階段開始。

   如果 IAM 實體尚未使用 `SSMSessionRunAs = os user account name` 標記，請繼續執行步驟 2。
2. 如果 IAM 實體尚未標記 `SSMSessionRunAs = os user account name`，是否已在 AWS 帳戶的Session Manager偏好設定中指定作業系統使用者名稱？

   如果是，該受管節點上是否存在作業系統使用者名稱？ 如果有，就開始工作階段。如果沒有，則不允許工作階段開始。

> **Note**
>
> 如果啟動執行身分支援，它會防止 Session Manager 使用受管理節點上的 `ssm-user` 帳戶啟動工作階段。這表示如果 Session Manager 無法使用指定的作業系統使用者帳戶進行連線，則不會返回使用預設方法進行連線。
> 如果您在未指定作業系統帳戶或標記 IAM 實體的情況下啟用執行身分，且尚未在 Session Manager 偏好設定中指定作業系統帳戶，則工作階段連線嘗試將會失敗。

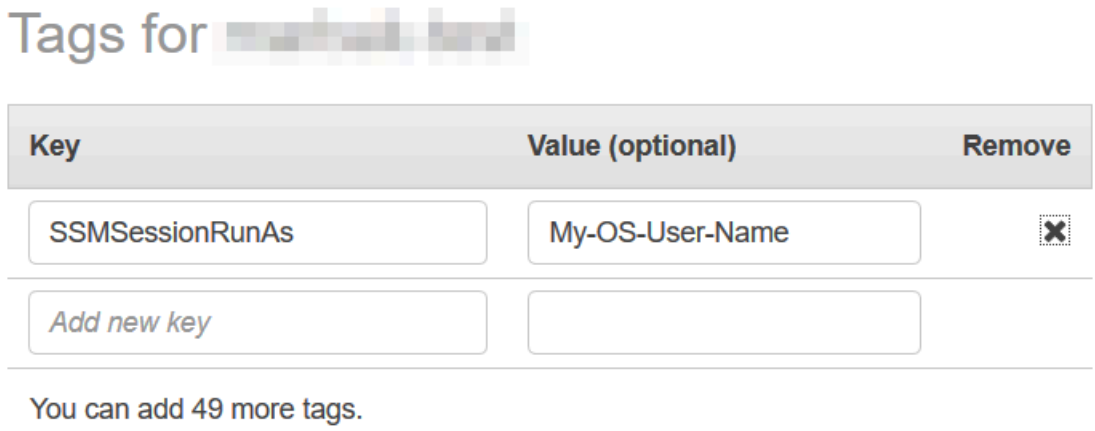
為 Linux 和 macOS 受管節點開啟執行身分支援

1. 開啟 AWS Systems Manager 主控台，網址為 https：//https://console.aws.amazon.com/systems-manager/。
2. 在導覽窗格中，選擇 Session Manager。
3. 選擇 Preferences (偏好) 標籤，然後選擇 Edit (編輯)。
4. 選取 為 Linux 執行個體啟用執行身分支援旁的核取方塊。
5. 執行以下任意一項：

   - 選項 1：在作業系統使用者名稱欄位，輸入您想要用來啟動工作階段的作業系統使用者帳戶名稱。使用此選項，所有工作階段都會由相同的作業系統使用者，針對使用 AWS 帳戶 連線的 中的所有使用者執行Session Manager。
   - 選項 2 （建議）：選擇開啟 IAM 主控台連結。在導覽窗格中，選擇 Users (使用者) 或者 Roles (角色)。選擇要新增標籤的實體 (使用者或角色)，然後選擇 Tags (標籤) 標籤。在金鑰名稱輸入 `SSMSessionRunAs`。在鍵值中輸入作業系統使用者帳戶的名稱。選擇儲存變更。

     使用此選項，您可以視需要為不同的 IAM 實體指定唯一的作業系統使用者。如需標記 IAM 實體（使用者或角色）的詳細資訊，請參閱《IAM 使用者指南》中的標記 IAM 資源。

     以下是範例。



6. 選擇儲存。

開啟工作階段資料的 KMS 金鑰加密 (主控台)

使用 AWS Key Management Service (AWS KMS) 建立和管理加密金鑰。使用 AWS KMS，您可以控制在應用程式的各種 AWS 服務 和 中使用加密。您可以指定在受管節點與 中使用者的本機機器之間

傳輸的工作階段資料 AWS 帳戶 ，會使用 KMS 金鑰加密進行加密。（這是預設 AWS 已提供的 TLS 1.2/1.3 加密之外的 。）若要加密Session Manager工作階段資料，請使用 建立對稱 KMS 金鑰 AWS KMS。

AWS KMS 加密適用於 `Standard_Stream`、 `InteractiveCommands`和 `NonInteractiveCommands`工作階段類型。若要使用 選項，使用在 中建立的金鑰來加密工作階段資料 AWS KMS， 2.3.539.0 版 AWS Systems Manager SSM Agent或更新版本必須安裝在受管節點上。

> **Note**
>
> 您必須允許 AWS KMS 加密，才能從 AWS Systems Manager 主控台重設受管節點上的密碼。如需詳細資訊，請參閱在受管節點上重設密碼。

您可以使用您在 中建立的金鑰 AWS 帳戶。您也可以使用在不同 AWS 帳戶中建立的金鑰。不同 中金鑰的建立者 AWS 帳戶 必須為您提供使用金鑰所需的許可。

在您啟用工作階段資料的 KMS 金鑰加密後，啟動工作階段的使用者和這些工作階段連接的受管節點必須具有該金鑰的使用許可。您提供使用 KMS 金鑰搭配 Session Manager through AWS Identity and Access Management (IAM) 政策的許可。如需詳細資訊，請參閱以下主題：

- 新增您帳戶中使用者的 AWS KMS 許可：適用於 Session Manager 的範例 IAM 政策。
- 新增您帳戶中受管節點的 AWS KMS 許可：步驟 2：為 Session Manager 確認或新增執行個體許可。

如需建立和管理 KMS 金輪的詳細資訊，請參閱 AWS Key Management Service 開發人員指南。

如需有關使用 AWS CLI 開啟您帳戶中工作階段資料的 KMS 金鑰加密的資訊，請參閱 建立 Session Manager 偏好設定文件 (命令列)或 更新 Session Manager 偏好設定 (命令列)。

> **Note**
>
> 使用 KMS 金輪需要付費嗎？ 如需相關資訊，請參閱 AWS Key Management Service 定價。

若要開啟工作階段資料的 KMS 金鑰加密 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。

2. 在導覽窗格中，選擇 Session Manager。
3. 選擇 Preferences (偏好) 標籤，然後選擇 Edit (編輯)。
4. 選取 Enable KMS encryption (啟用 KMS 加密) 旁邊的核取方塊。
5. 執行以下任意一項：

   - 選擇 Select a KMS key in my current account (在我目前的帳戶中選取 KMS 金鑰) 旁邊的按鈕，然後從清單中選取金鑰。

     -或-

     選擇 Enter a KMS key alias or KMS key ARN (輸入 KMS 金鑰別名或 KMS 金鑰 ARN) 旁的按鈕。為您在目前帳戶中建立的金鑰手動輸入 KMS 金鑰別名，或為另一個帳戶中的金鑰輸入其 Amazon Resource Name (ARN)。範例如下：

     - 金鑰別名：`alias/my-kms-key-alias`
     - 金鑰 ARN：`arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-12345EXAMPLE`

     -或-

     選擇 Create new key (建立新的金鑰) 以在您的帳戶中建立新的 KMS 金輪。在建立新的金鑰後，返回 Preferences (偏好設定) 標籤，然後選取要用來在您帳戶中加密工作階段資料的金鑰。

   如需共用金鑰的詳細資訊，請參閱《 AWS Key Management Service 開發人員指南》中的允許外部 AWS 帳戶 存取金鑰。
6. 選擇 Save (儲存)。

建立 Session Manager 偏好設定文件 (命令列)

使用下列程序來建立 SSM 文件，以定義工作階段的 AWS Systems Manager Session Manager偏好設定。您可以使用該文件來設定工作階段選項，包括資料加密、工作階段持續時間和日誌記錄。例如，您可以指定在 Amazon Simple Storage Service (Amazon S3) 儲存貯體或 Amazon CloudWatch Logs 日誌群組中存放工作階段日誌資料的選項。您可以建立定義 AWS 帳戶 和 所有工作階段一般偏好設定的文件 AWS 區域，或定義個別工作階段偏好設定的文件。

> **Note**
>
> 您也可以使用 Session Manager 主控台設定一般工作階段偏好設定。

用來設定 Session Manager 偏好設定的文件的 `sessionType` 必須為 `Standard_Stream`。如需有關工作階段文件的詳細資訊，請參閱 the section called “工作階段文件結構描述”。

如需使用命令列來更新現有 Session Manager 偏好設定的資訊，請參閱 更新 Session Manager 偏好設定 (命令列)。

如需如何使用 建立工作階段偏好設定的範例 AWS CloudFormation，請參閱AWS CloudFormation《 使用者指南》中的為Session Manager偏好設定建立 Systems Manager 文件。

> **Note**
>
> 此程序說明如何建立文件，以在 AWS 帳戶 關卡設定Session Manager偏好設定。若要建立將用於設定工作階段層級偏好設定的文件，請為檔案名稱相關指令輸入指定初 `SSM-SessionManagerRunShell` 之外的值。
>
> 若要使用文件來設定從 AWS Command Line Interface (AWS CLI) 啟動之工作階段的偏好設定，請提供文件名稱作為 `--document-name` 參數值。若要為從 Session Manager 主控台啟動的工作階段設定偏好設定，您可以輸入文件名稱或從清單中選取文件的名稱。

建立 Session Manager 偏好設定 (命令列)

1. 在您的本機電腦建立 JSON 檔案的名稱，例如 `SessionManagerRunShell.json`，然後貼上以下內容。

```
{
    "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"
        }
    }
}
```

您也可以使用參數將值傳遞到工作階段偏好設定，而不是對這些值進行硬編碼，如以下範例所示。

```
{
   "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. 指定您要傳送工作階段資料的位置。您可以指定 S3 儲存貯體名稱 (包含選用字首) 或 CloudWatch Logs 日誌群組名稱。如果您想要進一步加密本機用戶端和受管節點的資料，請提供用於加密的 KMS 金鑰。以下是範例。

```
{
  "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": "MyLogGroupName",
    "cloudWatchEncryptionEnabled": true,
    "cloudWatchStreamingEnabled": false,
    "kmsKeyId": "MyKMSKeyID",
    "runAsEnabled": true,
    "runAsDefaultUser": "MyDefaultRunAsUser",
    "idleSessionTimeout": "20",
    "maxSessionDuration": "60",
    "shellProfile": {
        "windows": "MyCommands",
        "linux": "MyCommands"
    }
  }
}
```

> **Note**
>
> 如果您不想加密工作階段日誌資料，請將 `s3EncryptionEnabled` 的 `true` 設為 `false`。
>
> 如果您不是將日誌傳送到 Amazon Simple Storage Service (Amazon S3) 儲存貯體或 CloudWatch Logs 日誌群組、不想加密作用中的工作階段資料，或不想為帳戶中的工作階段啟用執行身分支援，您可以刪除這幾行選項。請確定在 `inputs` 部分最後一行不是用逗號結尾。
>
> 如果您新增可加密工作階段資料的 KMS 金鑰 ID，啟動工作階段的使用者和這些工作階段連接的受管節點必須具有該金鑰的使用許可。您可透過 IAM 政策提供將 KMS 金輪 與 Session Manager 搭配使用的許可。如需詳細資訊，請參閱以下主題：
>
> - 新增您帳戶中使用者的 AWS KMS 許可： 適用於 Session Manager 的範例 IAM 政策

- 新增您帳戶中受管節點的 AWS KMS 許可： 步驟 2：為 Session Manager 確認或新增執行個體許可

3. 儲存檔案。
4. 在您建立 JSON 檔案的目錄裡執行下列命令。

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，此命令傳回的輸出會類似如下。

```
{
    "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 偏好設定 (命令列)

下列程序說明如何使用您偏好的命令列工具，對所選 AWS Systems Manager Session Manager AWS 帳戶 中的 偏好設定進行變更 AWS 區域。使用 Session Manager 偏好設定，以指定在 Amazon Simple Storage Service (Amazon S3) 儲存貯體或 Amazon CloudWatch Logs 日誌群組中記錄工作階段資料的選項。您也可以使用 Session Manager 偏好設定來加密工作階段資料。

更新 Session Manager 偏好設定 (命令列)

1. 在您的本機電腦建立 JSON 檔案的名稱，例如 `SessionManagerRunShell.json`，然後貼上以下內容。

```
{
    "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. 指定您要傳送工作階段資料的位置。您可以指定 S3 儲存貯體名稱 (包含選用字首) 或 CloudWatch Logs 日誌群組名稱。如果您想要進一步加密本機用戶端與受管節點之間的資料，請提供 AWS KMS key 以用於加密。以下是範例。

```
{
  "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": "MyLogGroupName",
    "cloudWatchEncryptionEnabled": true,
    "cloudWatchStreamingEnabled": false,
    "kmsKeyId": "MyKMSKeyID",
    "runAsEnabled": true,
    "runAsDefaultUser": "MyDefaultRunAsUser",
    "idleSessionTimeout": "20",
    "maxSessionDuration": "60",
    "shellProfile": {
        "windows": "MyCommands",
        "linux": "MyCommands"
    }
  }
}
```

> **Note**
>
> 如果您不想加密工作階段日誌資料，請將 `s3EncryptionEnabled` 的 `true` 設為 `false`。
>
> 如果您不是將日誌傳送到 Amazon Simple Storage Service (Amazon S3) 儲存貯體或 CloudWatch Logs 日誌群組、不想加密作用中的工作階段資料，或不想為帳戶中的工作階

段啟用執行身分支援，您可以刪除這幾行選項。請確定在 inputs 部分最後一行不是用逗號結尾。
如果您新增可加密工作階段資料的 KMS 金鑰 ID，啟動工作階段的使用者和這些工作階段連接的受管節點必須具有該金鑰的使用許可。您提供使用 KMS 金鑰搭配 Session Manager through AWS Identity and Access Management (IAM) 政策的許可。如需詳細資訊，請參閱以下主題：

- 新增您帳戶中使用者的 AWS KMS 許可：適用於 Session Manager 的範例 IAM 政策。
- 新增您帳戶中受管節點的 AWS KMS 許可：步驟 2：為 Session Manager 確認或新增執行個體許可。

3. 儲存檔案。
4. 在您建立 JSON 檔案的目錄裡執行下列命令。

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，此命令傳回的輸出會類似如下。

```
{
```

```
    "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"
    }
}
```

步驟 5：(選用) 限制對工作階段中命令的存取

您可以使用自訂`Session`類型 AWS Systems Manager (SSM) 文件，限制使用者可以在 AWS Systems Manager Session Manager工作階段中執行的命令。在文件中，您可以定義當使用者啟動工作階段時所執行的命令，以及可以提供給命令的參數。`Session` 文件 `schemaVersion` 必須為 1.0，且文件的 `sessionType` 必須為 `InteractiveCommands`。接著，您可以建立 AWS Identity and Access Management (IAM) 政策，只允許使用者存取您定義的 `Session` 文件。如需使用 IAM 政策來限制對工作階段中命令之存取權的詳細資訊，請參閱 互動式命令的 IAM 政策範例。

具有 `sessionType`的 文件`InteractiveCommands`僅支援從 AWS Command Line Interface () 啟動的工作階段AWS CLI。使用者提供自訂文件名稱做為 `--document-name` 參數值，並使用 `--parameters` 選項提供任何命令參數值。若要取得有關執行互動式命令的詳細資訊，請參閱啟動工作階段 (互動和非互動式命令)。

使用以下程序建立自訂 `Session` 類型 SSM 文件，以定義允許使用者執行的命令。

限制對工作階段中命令的存取 (主控台)

限制使用者可以在 Session Manager 工作階段中執行的命令 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Documents (文件)。
3. 選擇 Create command or session (建立命令或工作階段)。
4. 對於 Name (名稱)，輸入文件的描述性名稱。
5. 對於 Document type (文件類型)，請選擇 Session document (工作階段文件)。
6. 使用 JSON 或 YAML 輸入定義使用者可在 Session Manager 工作階段中執行之命令的文件內容，如下列範例所示。

   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. 選擇 Create document (建立文件)。

限制對工作階段中命令的存取 (命令列)

開始之前

如果您尚未安裝，請安裝並設定 AWS Command Line Interface (AWS CLI) 或 AWS Tools for PowerShell。如需相關資訊，請參閱安裝或更新 AWS CLI的最新版本和安裝 AWS Tools for PowerShell。

限制使用者可以在 Session Manager 工作階段中執行的命令 (命令列)

1. 為定義使用者可在 Session Manager 工作階段中執行之命令的文件內容建立 JSON 或 YAML 檔案，如下列範例所示。

   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. 執行下列命令，以使用定義使用者可在 Session Manager 工作階段中執行之命令的內容來建立 SSM 文件。

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"
```

互動式命令參數和 AWS CLI

使用 AWS CLI時，您可以透過多種方式提供互動式命令參數。根據您用來使用 連線至受管節點的用戶端機器作業系統 (OS) AWS CLI，您為包含特殊或逸出字元的命令提供的語法可能會有所不同。下列範例顯示使用 時，您可以提供命令參數的一些不同方式 AWS CLI，以及如何處理特殊或逸出字元。

Parameter Store 您可以在 中參考存放在 中的參數 AWS CLI ，以取得命令參數，如下列範例所示。

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}}"]}'
```

以下範例示範如何搭配使用速記語法與 AWS CLI 來傳遞參數。

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"
```

您也可以在 JSON 中提供參數，如以下範例所示。

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"]}'
```

參數也可以存放在 JSON 檔案中，並提供給 AWS CLI ，如下列範例所示。如需有關在檔案中使用 AWS CLI 參數的詳細資訊，請參閱《AWS Command Line Interface 使用者指南》中的從檔案中載入 AWS CLI 參數。

```
{
    "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
```

您也可以從 JSON AWS CLI 輸入檔案產生骨架，如下列範例所示。如需從 JSON AWS CLI 輸入檔案產生骨架的詳細資訊，請參閱AWS Command Line Interface 《 使用者指南》中的從 JSON 或 YAML 輸入檔案產生 AWS CLI 骨架和輸入參數。

```
{
    "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
```

若要逸出引號內的字元，您必須將其他反斜線新增至逸出字元，如下列範例所示。

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\""]}'
```

如需有關在 AWS CLI中搭配使用引號與命令參數的詳細資訊，請參閱《AWS Command Line Interface 使用者指南》中的在 AWS CLI中搭配使用引號與字串。

互動式命令的 IAM 政策範例

您可以建立 IAM 政策，只允許使用者存取您定義的 Session 文件。這會將使用者可在 Session Manager 工作階段中執行的命令，限制為僅限您的自訂 Session 類型 SSM 文件中定義的命令。

允許使用者在單一受管節點上執行互動式命令

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":"ssm:StartSession",
         "Resource":[
            "arn:aws:ec2:region:987654321098:instance/i-02573cafcfEXAMPLE",
            "arn:aws:ssm:region:987654321098:document/exampleAllowedSessionDocument"
         ]
      }
   ]
},
{
         "Effect": "Allow",
         "Action": ["ssmmessages:OpenDataChannel"],
         "Resource": ["arn:aws:ssm:*:*:session/${aws:userid}-*"]
       }
```

允許使用者在所有受管節點上執行互動式命令

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

```
      {
         "Effect":"Allow",
         "Action":"ssm:StartSession",
         "Resource":[
            "arn:aws:ec2:us-west-2:987654321098:instance/*",
            "arn:aws:ssm:us-
west-2:987654321098:document/exampleAllowedSessionDocument"
         ]
      }
   ]
},
{
         "Effect": "Allow",
         "Action": ["ssmmessages:OpenDataChannel"],
         "Resource": ["arn:aws:ssm:*:*:session/${aws:userid}-*"]
       }
```

允許使用者在所有受管節點上執行多個互動式命令

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":"ssm:StartSession",
         "Resource":[
            "arn:aws:ec2:us-west-2:987654321098:instance/*",
            "arn:aws:ssm:us-
west-2:987654321098:document/exampleAllowedSessionDocument",
            "arn:aws:ssm:us-
west-2:987654321098:document/exampleAllowedSessionDocument2"
         ]
      }
   ]
},
{
         "Effect": "Allow",
         "Action": ["ssmmessages:OpenDataChannel"],
         "Resource": ["arn:aws:ssm:*:*:session/${aws:userid}-*"]
       }
```

步驟 6：(選用) 使用 AWS PrivateLink 設定 Session Manager 的 VPC 端點

您可以將 AWS Systems Manager 設定成使用界面 virtual private cloud (VPC) 端點，以提升您受管節點的安全狀態。界面端點採用一種技術 AWS PrivateLink，可讓您使用私有 IP 地址私下存取 Amazon Elastic Compute Cloud (Amazon EC2) 和 Systems Manager APIs。

AWS PrivateLink 會將受管節點、Systems Manager 和 Amazon EC2 之間的所有網路流量限制在 Amazon 網路。(受管階段無法存取網際網路。) 此外，您不需要網際網路閘道、NAT 裝置或虛擬私有閘道。

如需有關建立 VPC 端點的資訊，請參閱使用適用於 Systems Manager 的 VPC 端點來改善 EC2 執行個體的安全性。

使用 VPC 端點的替代方案是在您的受管節點上啟用對外網際網路存取。在此情況下，受管節點也必須允許 HTTPS (連接埠 443) 傳出流量至下列端點：

- `ec2messages.region.amazonaws.com`
- `ssm.region.amazonaws.com`
- `ssmmessages.region.amazonaws.com`

Systems Manager 使用這些端點的最後一個 (`ssmmessages.region.amazonaws.com`)，從 SSM Agent 中呼叫雲端中的 Session Manager 服務。

若要使用選用的功能，例如 AWS Key Management Service (AWS KMS) 加密、將日誌串流至 Amazon CloudWatch Logs (CloudWatch Logs)，以及將日誌傳送至 Amazon Simple Storage Service (Amazon S3)，您必須允許 HTTPS （連接埠 443) 傳出流量至下列端點：

- `kms.region.amazonaws.com`
- `logs.region.amazonaws.com`
- `s3.region.amazonaws.com`

如需 Systems Manager 所需端點的詳細資訊，請參閱參考：ec2messages、ssmmessages 和其他 API 操作。

步驟 7：(選用) 啟用或停用 ssm-user 帳戶管理許可

從 2.3.50.0 版開始 AWS Systems Manager SSM Agent，代理程式會建立名為 的本機使用者帳戶，`ssm-user`並將其新增至 `/etc/sudoers`(Linux 和 macOS) 或 管理員群組 ()Windows。在代

理程式 2.3.612.0 之前的版本中，當 SSM Agent 第一次啟動，或在安裝後重新啟動時，會建立該帳戶。在版本 2.3.612.0 和更高版本中，當受管節點上初次啟動工作階段時，會建立 `ssm-user` 帳戶。`ssm-user` 這是工作階段開始時的預設作業系統 (OS) 使用者 AWS Systems Manager Session Manager。2019 年 5 月 8 日發行 2.3.612.0 SSM Agent版。

如果您想要防止 Session Manager 使用者在節點上執行管理命令，您可以更新其 `ssm-user` 帳戶許可。您也可以移除這些權限之後恢復權限。

主題

- 在 Linux 和 macOS 上管理 ssm-user sudo 帳戶許可
- 在 Windows Server 上管理 ssm-user 管理員帳戶許可

在 Linux 和 macOS 上管理 ssm-user sudo 帳戶許可

使用以下其中一個程序來停用或啟用在 Linux 和 macOS 受管節點上 ssm-user 帳戶的 sudo 許可：

使用 Run Command 修改 ssm-user sudo 許可 (主控台)

- 搭配以下值在 從主控台執行命令 中使用程序：
  - 如需 Command document (命令文件)，請選擇 `AWS-RunShellScript`。
  - 移除 sudo 存取，在 Command parameters (命令參數) 區裡的 Commands (命令) 方塊貼上以下內容。

```
cd /etc/sudoers.d
echo "#User rules for ssm-user" > ssm-agent-users
```

-或-

要恢復 sudo 存取，請在 Command parameters (命令參數) 區裡的 Commands (命令) 方塊貼上以下內容。

```
cd /etc/sudoers.d
echo "ssm-user ALL=(ALL) NOPASSWD:ALL" > ssm-agent-users
```

使用命令列修改 ssm-user sudo 許可 (AWS CLI)

1. 連線至受管節點並執行下列命令。

```
sudo -s
```

2. 使用下列命令變更工作目錄。

```
cd /etc/sudoers.d
```

3. 開啟檔案 `ssm-agent-users` 進行編輯。
4. 移除 sudo 存取、刪除下行。

```
ssm-user ALL=(ALL) NOPASSWD:ALL
```

-或-

要還原 sudo 存取，請新增下行：

```
ssm-user ALL=(ALL) NOPASSWD:ALL
```

5. 儲存檔案。

在 Windows Server 上管理 ssm-user 管理員帳戶許可

使用以下其中一個程序來停用或啟用在 Windows Server 受管節點上 ssm-user 帳戶的管理者許可：

使用 Run Command 來修改管理者許可 (主控台)

- 搭配以下值在 從主控台執行命令 中使用程序：

  如需 Command document (命令文件)，請選擇 `AWS-RunPowerShellScript`。

  移除管理的存取，在 Command parameters (命令參數) 區域裡，在 Commands (命令) 方塊貼上以下內容。

```
net localgroup "Administrators" "ssm-user" /delete
```

  -或-

  移除恢復的存取，在 Command parameters (命令參數) 區域裡，在 Commands (命令) 方塊貼上以下內容。

```
net localgroup "Administrators" "ssm-user" /add
```

使用 PowerShell 或命令提示字元視窗修改管理員許可

1. 連接到受管節點，並開啟 PowerShell 或命令提示字元視窗。
2. 移除管理的存取、執行下列命令。

   ```
   net localgroup "Administrators" "ssm-user" /delete
   ```

   -或-

   要復原管理的存取，請執行下列命令。

   ```
   net localgroup "Administrators" "ssm-user" /add
   ```

使用 Windows 主控台修改管理員許可

1. 連接到受管節點，並開啟 PowerShell 或命令提示字元視窗。
2. 從命令列執行 `lusrmgr.msc` 以開啟 Local Users and Groups (本機使用者和群組) 主控台。
3. 開啟 Users (使用者) 目錄，然後開放 ssm-user。
4. 在 Member Of (成員) 標籤，執行以下其中一項：

   - 移除管理的存取，選取 Administrators (管理員)，然後選擇 Remove (移除)。

     -或-

     若要恢復的管理存取，在文字方塊裡輸入 **`Administrators`**，然後選擇 Add (新增)。
5. 選擇確定。

步驟 8：(選用) 透過 Session Manager 允許和控制 SSH 連線的許可

您可以允許 中的使用者 AWS 帳戶 使用 AWS Command Line Interface (AWS CLI) 來建立與 受管節點的 Secure Shell (SSH) 連線 AWS Systems Manager Session Manager。使用 SSH 連線的使用者也可以在其本機電腦和受管節點，使用 Secure Copy Protocol (SCP) 來複製檔案。您可以使用此功能，不需開啟傳入連接埠或維持堡壘主機，即可連接至受管節點。

允許 SSH 連線之後，您可以使用 AWS Identity and Access Management (IAM) 政策明確允許或拒絕使用者、群組或角色，以使用 進行 SSH 連線Session Manager。

> **Note**
>
> 透過連接埠轉送或 SSH 連線的 Session Manager 工作階段無法使用日誌記錄功能。這是因為 SSH 會加密所有工作階段資料，Session Manager 僅用作 SSH 連線的通道。

主題

- 允許 Session Manager 的 SSH 連線
- 透過 Session Manager 控制 SSH 連線的使用者許可

允許 Session Manager 的 SSH 連線

使用下列步驟透過 Session Manager 在受管節點上允許 SSH 連線。

允許 Session Manager 的 SSH 連線

1. 在您要啟用 SSH 連線的受管節點，執行下列作業：

   - 請確定 SSH 是在受管節點上執行。(您可以關閉節點上的傳入連接埠。)
   - 確保受管節點上已安裝 SSM Agent 2.3.672.0 版或更新版本。

     如需在受管節點安裝或更新 SSM Agent 的詳細資訊，請參閱下列主題：

     - 在 Windows Server EC2 執行個體上手動安裝和解除安裝 SSM Agent.
     - 在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent
     - 在 macOS EC2 執行個體上手動安裝和解除安裝 SSM Agent
     - 如何在混合 Windows 節點上安裝 SSM Agent
     - 如何在混合 Linux 節點上安裝 SSM Agent

     > **Note**
     >
     > 若要將 Session Manager 與內部部署伺服器、邊緣裝置和您啟用作為受管節點的虛擬機器 (VM) 搭配使用，您必須使用進階執行個體層。如需進階執行個體的詳細資訊，請參閱 設定執行個體方案。

2. 在您要使用 SSH 連接到受管節點的本機機器上，執行下列動作：

   - 確保已安裝 Session Manager 外掛程式的 1.1.23.0 版或更新版本。

     如需安裝 Session Manager 外掛程式的詳細資訊，請參閱 安裝 的Session Manager外掛程式 AWS CLI。

   - 更新 SSH 設定檔可讓開始 Session Manager 工作階段的代理命令得以執行，並透過該連接傳輸所有資料。

     Linux 和 macOS

     > **Tip**
     >
     > SSH 組態檔案通常位於 `~/.ssh/config`。

     將以下內容新增至本機機器上的組態檔。

     ```
     # 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 組態檔案通常位於 `C:\Users\<username>\.ssh\config`。

     將以下內容新增至本機機器上的組態檔。

     ```
     # 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"
     ```

- 建立或驗證您擁有隱私增強郵件 (PEM 檔案) 憑證，或至少擁有公有金鑰，可在建立與受管節點的連線時使用。這必須是已與受管節點相關聯的索引鍵。必須設定私有金鑰檔案的許可，確保只有您可以讀取此檔案。您可以使用下列命令來設定私有金鑰檔案的許可，確保只有您可以讀取此檔案。

  ```
  chmod 400 <my-key-pair>.pem
  ```

  例如，對於 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體，則是您在建立執行個體時建立或選取的金鑰對檔案。(您可以將憑證或金鑰的路徑指定為啟動工作階段命令的一部分。如需有關使用 SSH 啟動工作階段的資訊，請參閱 啟動工作階段 (SSH)。)

透過 Session Manager 控制 SSH 連線的使用者許可

透過 Session Manager 在受管節點上啟用 SSH 連線後，您可以使用 IAM 政策來允許或拒絕使用者、群組或角色透過 Session Manager 進行 SSH 連線。

若要使用 IAM 政策允許透過 Session Manager 進行 SSH 連線

- 使用下列其中一個選項：
  - 選項 1：在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。

    在導覽窗格中，選擇 Policies (政策)，然後更新您要允許透過 Session Manager 啟動 SSH 連線之使用者或角色的許可政策。

    例如，將下列元素新增至您在 Session Manager的最終使用者政策快速入門 中建立的 Quickstart 政策。將每個########取代為您自己的資訊。

    ```
    {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": "ssm:StartSession",
                "Resource": [
                    "arn:aws:ec2:region:account-id:instance/instance-id",
                    "arn:aws:ssm:*:*:document/AWS-StartSSHSession"
                ]
            }
        ]
    ```

```
},
{
         "Effect": "Allow",
         "Action": ["ssmmessages:OpenDataChannel"],
         "Resource": ["arn:aws:ssm:*:*:session/${aws:userid}-*"]
       }
```

- 選項 2：使用 AWS Management Console、 AWS CLI或 AWS API 將內嵌政策連接至使用者政策。

  使用您選擇的方法，將選項 1 中的政策陳述式連接到 AWS 使用者、群組或角色的政策。

  如需相關資訊，請參閱《IAM 使用者指南》中的新增和移除 IAM 身分許可。

若要使用 IAM 政策拒絕透過 Session Manager 進行 SSH 連線

- 使用下列其中一個選項：

  - 選項 1：在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。在導覽窗格中，選擇 Policies (政策)，然後更新使用者或角色的許可政策，以防止他們啟動 Session Manager 工作階段。

    例如，將下列元素新增至您在 Session Manager的最終使用者政策快速入門 中建立的 Quickstart 政策。

```
{
    "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}-*"]
       }
```

- 選項 2：使用 AWS CLI AWS Management Console、 或 AWS API 將內嵌政策連接至使用者政策。

  使用您選擇的方法，將選項 1 中的政策陳述式連接到 AWS 使用者、群組或角色的政策。

  如需相關資訊，請參閱《IAM 使用者指南》中的新增和移除 IAM 身分許可。

使用 Session Manager

您可以使用 AWS Systems Manager 主控台、Amazon Elastic Compute Cloud (Amazon EC2) 主控台或 AWS Command Line Interface (AWS CLI) 啟動工作階段，以將您連接到系統管理員已授予您使用 AWS Identity and Access Management (IAM) 政策存取的受管節點。根據您的許可，您也可以檢視工作階段的相關資訊、恢復尚未逾時的非作用中工作階段，以及結束工作階段。工作階段在建立之後不受 IAM 角色工作階段持續時間的影響。如需資訊了解如何利用 Session Manager 限制工作階段持續時間，請參閱 指定閒置工作階段逾時值 與 指定工作階段持續時間上限。

如需工作階段的詳細資訊，請參閱什麼是工作階段？

主題

- 安裝 的Session Manager外掛程式 AWS CLI
- 啟動工作階段
- 結束工作階段
- 檢視工作階段歷史記錄

安裝 的Session Manager外掛程式 AWS CLI

若要使用 AWS Command Line Interface (AWS CLI) 啟動受管節點的 Session Manager 工作階段，您必須在本機機器上安裝 Session Manager 外掛程式。您可以在受支援的 Microsoft Windows Server、macOS、Linux 和 Ubuntu Server 版本上安裝此外掛程式。

> **Note**
>
> 若要使用 Session Manager 外掛程式，您必須在本機電腦上安裝 AWS CLI 版本 1.16.12 或更新版本。如需詳細資訊，請參閱安裝或更新最新版本的 AWS Command Line Interface。

主題

- Session Manager 外掛程式最新版本和版本歷史記錄
- 在 Windows 上安裝 Session Manager 外掛程式
- 在 macOS 上安裝 Session Manager 外掛程式
- 在 Linux 上安裝 Session Manager 外掛程式
- 驗證 Session Manager 外掛程式安裝
- GitHub 上的外掛程式 Session Manager
- (選用) 開啟 Session Manager 外掛程式日誌

Session Manager 外掛程式最新版本和版本歷史記錄

您的本機電腦必須在支援 Session Manager 外掛程式的版本上執行。目前最低的支援版本為 1.1.17.0。如果您執行之前的版本，您的 Session Manager 操作可能不會成功。

若要查看如果您是否有最新的版本，請在 AWS CLI執行下列命令。

> **Note**
>
> 只在外掛程式位於您作業系統類型適用的預設安裝目錄位置中，此命令才會傳回結果。您也可以透過在您已安裝外掛程式的目錄中 `VERSION` 檔案的內容檢查版本。

```
session-manager-plugin --version
```

下表列出所有 Session Manager 外掛程式的版本及每個版本包含的功能及加強功能。

> **Important**
>
> 我們建議您一律使用最新版本。最新版本包含可改善外掛程式使用體驗的增強功能。

版本	版本日期	詳細資訊
1.2.707.0	2025 年 2 月 6 日	增強功能：將 Dockerfile 中的 Go 版本升級至 1.23。已更新 README 中的版本組態步驟。

版本	版本日期	詳細資訊
1.2.694.0	2024 年 11 月 20 日	錯誤修正：已復原將憑證新增至 OpenDataChannel 請求的變更。
1.2.688.0	2024 年 11 月 6 日	此版本已於 2024 年 11 月 20 日遭棄用。 增強功能： • 已將憑證新增至 OpenDataChannel 請求。 • 升級了 `testify` 和 `objx` 相依套件。
1.2.677.0	2024 年 10 月 10 日	增強功能：新增了支援使用 OpenDataChannel 請求傳遞外掛程式版本。
1.2.650.0	2024 年 7 月 2 日	增強功能：將 Go SDK 升級到了 1.54.10 版。 錯誤修正：重新格式化了 gofmt 檢查的說明。
1.2.633.0	2024 年 5 月 30 日	增強功能：更新了 Dockerfile 以使用 Amazon Elastic Container Registry (Amazon ECR) 映像。
1.2.553.0	2024 年 1 月 10 日	增強功能：升級了 aws-sdk-go 和相依的 Golang 套件。
1.2.536.0	2023 年 12 月 4 日	增強功能：新增將 StartSession API 回應作為環境變數傳遞給 session-manager-plugin 的支援。
1.2.497.0	2023 年 8 月 1 日	增強功能：將 Go SDK 升級到 1.44.302 版。
1.2.463.0	2023 年 3 月 15 日	增強功能：在 macOS 捆綁安裝程式和已簽署安裝程式中新增了對 Apple Mac (M1) 的 Mac with Apple silicon 支援。
1.2.398.0	2022 年 10 月 14 日	增強：支援 golang 1.17 版。更新 macOS 的預設 session-manager-plugin 執行器以使用 python3。將匯入路徑從 SSMCLI 更新為 session-manager-plugin。
1.2.339.0	2022 年 6 月 16 日	錯誤修正：針對連接埠工作階段修復閒置工作階段逾時問題。
1.2.331.0	2022 年 5 月 27 日	錯誤修正：當本機伺服務器在讀取逾時前未完成連線時，修復連接埠工作階段提早關閉的問題。

版本	版本日期	詳細資訊
1.2.323.0	2022 年 5 月 19 日	錯誤修正：停用 smux 持續作用功能來使用閒置工作階段逾時功能。
1.2.312.0	2022 年 3 月 31 日	改進：支援更多輸出訊息承載類型。
1.2.295.0	2022 年 1 月 12 日	錯誤修正：當代理程式處於非作用狀態時，用戶端重新傳送串流資料導致的工作階段停止回應，以及 `start_publication` 和 `pause_publication` 訊息的不正確日誌。
1.2.279.0	2021 年 10 月 27 日	增強功能：適用於 Windows 平台的 zip 封裝。
1.2.245.0	2021 年 8 月 19 日	改進：將 `aws-sdk-go` 升級至最新版 (1.40.17 版) 以支援 AWS IAM Identity Center。
1.2.234.0	2021 年 7 月 26 日	錯誤修正：處理交互式工作階段類型中的工作階段突然終止的情況。
1.2.205.0	2021 年 6 月 10 日	改進：已新增對簽署的 macOS 安裝程式的支援。
1.2.54.0	2021 年 1 月 29 日	改進：已新增在 NonInteractiveCommands 執行模式下執行工作階段的支援。
1.2.30.0	2020 年 11 月 24 日	改進：(僅限連接埠轉送工作階段) 提高了整體效能。
1.2.7.0	2020 年 10 月 15 日	改進：(僅限連接埠轉送工作階段) 降低了延遲並提高了整體效能。
1.1.61.0	2020 年 4 月 17 日	增強功能：新增支援適用於 Linux 和 Ubuntu 的 ARM。
1.1.54.0	2020 年 1 月 6 日	錯誤修正：處理封包在 Session Manager 外掛程式尚未就緒時受到捨棄的競爭條件情況。
1.1.50.0	2019 年 11 月 19 日	增強：新增將埠轉送至本機 unix 通訊端的支援。

版本	版本日期	詳細資訊
1.1.35.0	2019 年 11 月 7 日	增強：(僅限埠轉送工作階段) 當本機使用者按下 `Ctrl+C` 時，傳送 TerminateSession 命令至 SSM Agent。
1.1.33.0	2019 年 9 月 26 日	增強功能：(僅限連接埠轉送工作階段) 當用戶端中斷 TCP 連線時，傳送中斷連線訊號至伺服器。
1.1.31.0	2019 年 9 月 6 日	增強功能：更新為保持連接埠轉送工作階段開啟，直到遠端伺服器關閉連線為止。
1.1.26.0	2019 年 7 月 30 日	增強功能：更新以限制工作階段期間資料傳輸的速率。
1.1.23.0	2019 年 7 月 9 日	增強功能：新增使用 Session Manager 執行 SSH 工作階段的支援。
1.1.17.0	2019 年 4 月 4 日	增強功能：新增使用 AWS Key Management Service (AWS KMS) 為工作階段資料進一步加密的支援。
1.0.37.0	2018 年 9 月 20 日	增強功能：Windows 版本的錯誤修正。
1.0.0.0	2018 年 9 月 11 日	Session Manager 外掛程式初始版本。

在 Windows 上安裝 Session Manager 外掛程式

您可以在 Windows Vista 或更新版本使用獨立安裝程式安裝 Session Manager 外掛程式。

當發佈更新時，您必須重複安裝過程以獲取最新版本的 Session Manager 外掛程式。

> **Note**
>
> 記下以下資訊。
>
> - Session Manager 外掛程式安裝程式需要管理員權限才能安裝外掛程式。
> - 為了獲得最佳結果，我們建議您在開始工作階段時在 Windows 用戶端使用 Windows PowerShell 版本 5 或更新版本。或者，您可以使用 Windows 10 中的命令 Shell。Session Manager 外掛程式只支援 PowerShell 和命令 Shell。第三方命令列工具可能與外掛程式不相容。

使用 EXE 安裝程式安裝 Session Manager 外掛程式

1. 使用以下 URL 下載安裝程式。

   ```
   https://s3.amazonaws.com/session-manager-downloads/plugin/latest/windows/
   SessionManagerPluginSetup.exe
   ```

   或者，您可以使用下列 URL 下載壓縮版的安裝程式。

   ```
   https://s3.amazonaws.com/session-manager-downloads/plugin/latest/windows/
   SessionManagerPlugin.zip
   ```

2. 執行下載的安裝程式，並遵循畫面上的指示操作。如果您已下載壓縮版的安裝程式，則必須先解壓縮安裝程式。

   保留安裝位置方塊空白以便安裝外掛程式預設的目錄。

   - `%PROGRAMFILES%\Amazon\SessionManagerPlugin\bin\`

3. 確認已安裝成功。如需相關資訊，請參閱 驗證 Session Manager 外掛程式安裝。

   > **Note**
   >
   > 如果 Windows 無法找到可執行檔，則可能需要手動重新開啟命令提示，或手動將安裝目錄新增到您的 PATH 環境變數。如需詳細資訊，請參閱疑難排解主題 Session Manager 外掛程式未自動新增到命令列路徑 (Windows)。

在 macOS 上安裝 Session Manager 外掛程式

選擇下列其中一個主題，以在 macOS 上安裝 Session Manager 外掛程式。

> **Note**
>
> 簽章的安裝程式是簽章.pkg的檔案。綁定安裝程式使用 .zip 檔案。檔案解壓縮後，您可以使用二進位檔安裝外掛程式。

在 macOS 上安裝 Session Manager 外掛程式與已簽署的安裝程式

本節說明如何使用已簽署的安裝程式在 macOS 上安裝 Session Manager 外掛程式。

若要使用已簽署的安裝程式安裝 Session Manager 外掛程式 (macOS)

1. 下載已簽署的安裝程式。

   x86_64

   ```
   curl "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/mac/
   session-manager-plugin.pkg" -o "session-manager-plugin.pkg"
   ```

   搭載 Apple 晶片的 Mac

   ```
   curl "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/
   mac_arm64/session-manager-plugin.pkg" -o "session-manager-plugin.pkg"
   ```

2. 執行安裝命令。如果命令失敗，請確認/usr/local/bin資料夾存在。如果沒有，請建立它，然後再次執行命令。

   ```
   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. 確認已安裝成功。如需相關資訊，請參閱驗證 Session Manager 外掛程式安裝。

在 macOS 上安裝 Session Manager 外掛程式

本節說明如何使用隨附的安裝程式在 macOS 上安裝 Session Manager 外掛程式。

> **Important**
>
> 記下以下重要資訊。
>
> - 依預設，安裝程式需要 sudo 存取權才能執行，因為指令碼會將外掛程式安裝至 /usr/local/sessionmanagerplugin 系統目錄。如果您不想使用 sudo 安裝外掛程式，請手動更新安裝程式指令碼，將外掛程式安裝至不需要 sudo 存取權的目錄。
> - Bundled Installer 無法安裝到包含空格的路徑。

使用 Bundled Installer (macOS) 安裝 Session Manager 外掛程式

1. 下載 Bundled Installer。

x86_64

```
curl "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/mac/
sessionmanager-bundle.zip" -o "sessionmanager-bundle.zip"
```

搭載 Apple 晶片的 Mac

```
curl "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/
mac_arm64/sessionmanager-bundle.zip" -o "sessionmanager-bundle.zip"
```

2. 解壓縮套件。

```
unzip sessionmanager-bundle.zip
```

3. 執行安裝命令。

```
sudo ./sessionmanager-bundle/install -i /usr/local/sessionmanagerplugin -b /usr/
local/bin/session-manager-plugin
```

> **Note**
>
> 外掛程式需要 Python 2.6.5 或更新版本，或者 Python 3.3 或更新版本。根據預設，安裝指令碼會在系統預設版本的 Python 下執行。如果您已安裝其他版本的 Python，並想使用該版本軟體安裝 Session Manager 外掛程式，請使用該版本按 Python 可執行檔的絕對路徑來執行安裝指令碼。以下是範例。
>
> ```
> sudo /usr/local/bin/python3.8 sessionmanager-bundle/install -i /usr/local/
> sessionmanagerplugin -b /usr/local/bin/session-manager-plugin
> ```

安裝程式會將 Session Manager 外掛程式安裝到 `/usr/local/sessionmanagerplugin`，並在 `/usr/local/bin` 目錄中建立符號連結 `session-manager-plugin`。這讓您無須在使用者的 `$PATH` 變動指定安裝的目錄。

若要參閱 `-i` 和 `-b` 選項的說明，請使用 `-h` 選項。

```
./sessionmanager-bundle/install -h
```

4. 確認已安裝成功。如需相關資訊，請參閱驗證 Session Manager 外掛程式安裝。

> **Note**
>
> 若要解除安裝外掛程式，請按顯示的順序執行以下兩個命令。
>
> ```
> sudo rm -rf /usr/local/sessionmanagerplugin
> ```
>
> ```
> sudo rm /usr/local/bin/session-manager-plugin
> ```

在 Linux 上安裝 Session Manager 外掛程式

本節包含驗證Session Manager外掛程式安裝程式套件的簽章，以及在下列 Linux 發行版本上安裝外掛程式的相關資訊：

- Amazon Linux 2
- AL2023
- RHEL
- Debian
- Ubuntu

主題

- 驗證Session Manager外掛程式的簽章
- 在 Amazon Linux 2、Amazon Linux 2023 和 Red Hat Enterprise Linux 發行版本上安裝 Session Manager 外掛程式
- 在 Debian Server 和 Ubuntu Server 上安裝 Session Manager 外掛程式

驗證Session Manager外掛程式的簽章

Linux 執行個體的Session Manager外掛程式 RPM 和 Debian 安裝程式套件會以密碼編譯方式簽署。您可以使用公有金鑰來驗證外掛程式二進位檔和套件為原始且未修改。如果檔案遭到修改或損壞，驗證會失敗。您可以使用 GNU Privacy Guard (GPG) 工具來驗證安裝程式套件的簽章。下列資訊適用於Session Manager外掛程式 1.2.707.0 版或更新版本。

完成下列步驟，以驗證Session Manager外掛程式安裝程式套件的簽章。

主題

- 步驟 1：下載Session Manager外掛程式安裝程式套件
- 步驟 2：下載相關聯的簽章檔案
- 步驟 3：安裝 GPG 工具
- 步驟 4：在 Linux 伺服器上驗證Session Manager外掛程式安裝程式套件

步驟 1：下載Session Manager外掛程式安裝程式套件

下載您要驗證的Session Manager外掛程式安裝程式套件。

Amazon Linux 2、AL2023 和 RHEL RPM 套件

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 套件

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"
```

步驟 2：下載相關聯的簽章檔案

下載安裝程式套件之後，請下載相關聯的簽章檔案以進行套件驗證。為了提供額外的保護，防止未經授權複製或使用套件內的 session-manager-plugin 二進位檔案，我們也提供二進位簽章，您可以使用這些簽章來驗證個別的二進位檔案。您可以選擇根據您的安全需求使用這些二進位簽章。

Amazon Linux 2、AL2023 和RHEL簽章套件

x86_64

套件：

```
curl -o "session-manager-plugin.rpm.sig" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/linux_64bit/session-manager-plugin.rpm.sig"
```

二進位：

```
curl -o "session-manager-plugin.sig" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/linux_64bit/session-manager-plugin.sig"
```

x86

套件：

```
curl -o "session-manager-plugin.rpm.sig" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/linux_32bit/session-manager-plugin.rpm.sig"
```

二進位：

```
curl -o "session-manager-plugin.sig" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/linux_32bit/session-manager-plugin.sig"
```

ARM64

套件：

```
curl -o "session-manager-plugin.rpm.sig" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/linux_arm64/session-manager-plugin.rpm.sig"
```

二進位：

```
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 簽章套件

x86_64

套件：

```
curl -o "session-manager-plugin.deb.sig" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/ubuntu_64bit/session-manager-plugin.deb.sig"
```

二進位：

```
curl -o "session-manager-plugin.sig" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/ubuntu_64bit/session-manager-plugin.sig"
```

x86

套件：

```
curl -o "session-manager-plugin.deb.sig" "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/ubuntu_32bit/session-manager-plugin.deb.sig"
```

二進位：

```
curl -o "session-manager-plugin.sig" "https://s3.amazonaws.com/session-manager-
downloads/plugin/latest/ubuntu_32bit/session-manager-plugin.sig"
```

ARM64

套件：

```
curl -o "session-manager-plugin.deb.sig" "https://s3.amazonaws.com/session-manager-
downloads/plugin/latest/ubuntu_arm64/session-manager-plugin.deb.sig"
```

二進位：

```
curl -o "session-manager-plugin.sig" "https://s3.amazonaws.com/session-manager-
downloads/plugin/latest/ubuntu_arm64/session-manager-plugin.sig"
```

步驟 3：安裝 GPG 工具

若要驗證Session Manager外掛程式的簽章，您必須在系統上安裝 GNU Privacy Guard (GPG) 工具。驗證程序需要 GPG 2.1 版或更新版本。您可以執行下列命令來檢查 GPG 版本：

```
gpg --version
```

如果您的 GPG 版本早於 2.1，請在繼續驗證程序之前更新它。對於大多數系統，您可以使用套件管理員更新 GPG 工具。例如，在 Amazon Linux 和RHEL系統上，您可以使用下列命令：

```
sudo yum update
sudo yum install gnupg2
```

在 Ubuntu 或 Debian 系統上，您可以使用下列命令：

```
sudo apt-get update
sudo apt-get install gnupg2
```

在繼續驗證程序之前，請確定您擁有必要的 GPG 版本。

步驟 4：在 Linux 伺服器上驗證Session Manager外掛程式安裝程式套件

使用下列程序來驗證 Linux 伺服器上的Session Manager外掛程式安裝程式套件。

> **Note**
>
> Amazon Linux 2 不支援 gpg 工具 2.1 版或更新版本。如果下列程序不適用於您的 Amazon Linux 2 執行個體，請在其他平台上驗證簽章，再將其安裝在 Amazon Linux 2 執行個體上。

1. 複製下列公有金鑰，並將其儲存至名為 session-manager-plugin.gpg 的檔案。

```
-----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. 將公開金鑰匯入至您的 keyring。傳回的金鑰值應為 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. 執行下列命令來驗證指紋。

```
gpg --fingerprint 2C4D4AFF6F6757EE
```

命令輸出的指紋應符合下列項目。

```
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)
```

如果指紋不相符，請勿安裝外掛程式。聯絡 AWS 支援。

4. 確認安裝程式套件簽章。將 *signature-filename* 和 *downloaded-plugin-filename* 取代為您下載簽章檔案和 session-manager-plugin 時指定的值，如本主題稍早的表格中所列。

```
gpg --verify signature-filename downloaded-plugin-filename
```

例如，對於 Amazon Linux 2 上的 x86_64 架構， 命令如下所示：

```
gpg --verify session-manager-plugin.rpm.sig session-manager-plugin.rpm
```

此命令會傳回類似以下的輸出。

```
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
```

如果輸出包含 BAD signature 片語，請檢查您是否已正確執行程序。如果您繼續取得此回應，請聯絡 AWS 支援 且不要安裝套件。關於信任的警告訊息並不表示該簽章無效，只是您尚未驗證該公有金鑰。只有您或您信任者所簽章的金鑰才能信任。如果輸出包含片語 Can't check signature: No public key，請確認您已下載 1.2.707.0 版或更新版本的Session Manager外掛程式。

在 Amazon Linux 2、Amazon Linux 2023 和 Red Hat Enterprise Linux 發行版本上安裝 Session Manager 外掛程式

按照下列程序在 Amazon Linux 2、Amazon Linux 2023 (AL2023) 和 RHEL 發行版本上安裝 Session Manager 外掛程式。

> **Note**
>
> Amazon Linux 1 不支援 Session Manager 外掛程式。Amazon Linux 2 和更新發行版本予以支援。

1. 下載並安裝 Session Manager 外掛程式 RPM 套件。

   x86_64

   在 Amazon Linux 2 和 RHEL 7 上執行下列命令：

   ```
   sudo yum install -y https://s3.amazonaws.com/session-manager-downloads/plugin/latest/linux_64bit/session-manager-plugin.rpm
   ```

   在 AL2023 與 RHEL 8 和 9 上執行下列命令：

   ```
   sudo dnf install -y https://s3.amazonaws.com/session-manager-downloads/plugin/latest/linux_64bit/session-manager-plugin.rpm
   ```

   x86

   在 RHEL 7 上執行下列命令：

   ```
   sudo yum install -y https://s3.amazonaws.com/session-manager-downloads/plugin/latest/linux_32bit/session-manager-plugin.rpm
   ```

   在 RHEL 8 和 9 上執行下列命令：

   ```
   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 上執行下列命令：

   ```
   sudo yum install -y https://s3.amazonaws.com/session-manager-downloads/plugin/latest/linux_arm64/session-manager-plugin.rpm
   ```

在 AL2023 與 RHEL 8 和 9 上執行下列命令：

```
sudo dnf install -y https://s3.amazonaws.com/session-manager-downloads/plugin/
latest/linux_arm64/session-manager-plugin.rpm
```

2. 確認已安裝成功。如需相關資訊，請參閱 驗證 Session Manager 外掛程式安裝。

> **Note**
>
> 如果您想要解除安裝外掛程式，請執行 `sudo yum erase session-manager-plugin -y`

在 Debian Server 和 Ubuntu Server 上安裝 Session Manager 外掛程式

1. 下載 Session Manager 外掛程式 deb 套件。

   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. 執行安裝命令。

   ```
   sudo dpkg -i session-manager-plugin.deb
   ```

3. 確認已安裝成功。如需相關資訊，請參閱 驗證 Session Manager 外掛程式安裝。

> **Note**
>
> 如果您想要解除安裝外掛程式，請執行 `sudo dpkg -r session-manager-plugin`

驗證 Session Manager 外掛程式安裝

執行下列命令來驗證 Session Manager 外掛程式是否安裝成功：

```
session-manager-plugin
```

如果操作成功，您會看到以下訊息。

```
The Session Manager plugin is installed successfully. Use the AWS CLI to start a
 session.
```

您也可以在 AWS Command Line Interface (AWS CLI) 中透過執行 start-session 命令來測試安裝。在下列命令中：將 *instance-id* 取代為您自己的資訊。

```
aws ssm start-session --target instance-id
```

只有在您已安裝並設定 AWS CLI，且您的Session Manager管理員已授予您使用 存取目標受管節點的必要 IAM 許可時，此命令才會運作Session Manager。

GitHub 上的外掛程式 Session Manager

GitHub 上提供 Session Manager 外掛程式的原始碼，便於您根據需求調整外掛程式。我們建議您為想要進行的變更提交提取請求。但是，Amazon Web Services 不支援執行修改過的軟體複本。

(選用) 開啟 Session Manager 外掛程式日誌

Session Manager 外掛程式包含一個選項讓您能夠記錄執行的工作階段。根據預設，會關閉日誌。

如果您啟用記錄，Session Manager 外掛程式在本機電腦上應用程式活動建立兩個日誌檔 (`session-manager-plugin.log`) 和錯誤 (`errors.log`)。

主題

- 開啟 Session Manager 外掛程式的日誌記錄 (Windows)
- 啟用 Session Manager 外掛程式的日誌記錄 (Linux 和 macOS)

開啟 Session Manager 外掛程式的日誌記錄 (Windows)

1. 找出 seelog.xml.template 檔案的外掛程式。

   預設位置為 C:\Program Files\Amazon\SessionManagerPlugin\seelog.xml.template。

2. 變更檔案的名稱成 seelog.xml。
3. 開啟檔案，然後將 minlevel="off" 變更為 minlevel="info" 或 minlevel="debug"。

   > **Note**
   >
   > 在預設情況下，日誌項目會以 INFO 等級記錄有關開啟資料管道和重新連線工作階段。資料流程 (封包和確認) 項目會記錄在 DEBUG 層級。

4. 變更您希望修改的其他組態選項。您可以變更選項包括：

   - 偵錯層級：您可以從 formatid="fmtinfo" 變更偵錯層級至 formatid="fmtdebug"。
   - 日誌檔案選項：您可以變更日誌檔的選項，包括日誌存放位置和日誌例外狀況名稱。

   > **Important**
   >
   > 請勿變更檔案名稱否則日誌記錄將無法正常運作。

   ```
   <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. 儲存檔案。

啟用 Session Manager 外掛程式的日誌記錄 (Linux 和 macOS)

1. 找出 seelog.xml.template 檔案的外掛程式。

   預設位置為 /usr/local/sessionmanagerplugin/seelog.xml.template。

2. 變更檔案的名稱成 seelog.xml。

3. 開啟檔案，然後將 `minlevel="off"` 變更為 `minlevel="info"` 或 `minlevel="debug"`。

> **Note**
>
> 在預設情況下，日誌項目會以 INFO 等級記錄有關開啟資料管道和重新連線工作階段。資料流程 (封包和確認) 項目會記錄在 DEBUG 層級。

4. 變更您希望修改的其他組態選項。您可以變更選項包括：

   - 偵錯層級：您可以從 `formatid="fmtinfo"` 變更偵錯層級至 `outputs formatid="fmtdebug"`。
   - 日誌檔案選項：您可以變更日誌檔的選項，包括日誌存放位置和日誌例外狀況名稱。

> **Important**
>
> 請勿變更檔案名稱否則日誌記錄將無法正常運作。

```
<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**
>
> 如果您使用指定的預設目錄執行工作階段命令來存放日誌，您必須使用 sudo 或提供外掛程式的安裝目錄來完整的執行命令讀取和寫入權限。若要略過這些限制，請變更日誌存放的位置。

5. 儲存檔案。

啟動工作階段

您可以使用 AWS Systems Manager 主控台、Amazon Elastic Compute Cloud (Amazon EC2) 主控台、 AWS Command Line Interface (AWS CLI) 或 SSH 來啟動工作階段。

主題

- 啟動工作階段 (Systems Manager 主控台)
- 啟動工作階段 (Amazon EC2 主控台)
- 啟動工作階段 (AWS CLI)
- 啟動工作階段 (SSH)
- 啟動工作階段 (網路埠轉遞)
- 啟動工作階段 (連接埠轉送至遠端主機)
- 啟動工作階段 (互動和非互動式命令)

啟動工作階段 (Systems Manager 主控台)

您可以使用 AWS Systems Manager 主控台，在帳戶中啟動具有受管節點的工作階段。

> **Note**
>
> 開始工作階段之前，請確定您已完成 Session Manager 的設定步驟。如需相關資訊，請參閱設定 Session Manager。

若要啟動工作階段 (Systems Manager 主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Session Manager。
3. 選擇 Start session (啟動工作階段)。
4. (選用) 在工作階段的原因欄位中輸入工作階段的描述。
5. 在目標執行個體中，選擇您想要連接到的受管節點旁的選項按鈕。

   如果您想要的節點不在清單中，或者您在選取節點後遇到組態錯誤，請參閱 受管節點無法使用或未設定用於 Session Manager 以獲取疑難排解步驟。
6. 選擇啟動工作階段，以立即啟動工作階段。

   -或-

   選擇下一步以查看工作階段選項。
7. (選用) 對於工作階段文件，選取您要在工作階段啟動時執行的文件。如果您的文件支援執行期參數，您可以在每個參數欄位中輸入一或多個逗號分隔值。

8. 選擇 Next (下一步)。
9. 選擇 Start session (啟動工作階段)。

建立連線之後，您可以執行 bash 命令 (Linux 和 macOS) 或 PowerShell 命令 (Windows)，就像您使用任何其他連線類型時一樣。

> **Important**
>
> 如果您想要允許使用者在 Session Manager 主控台中啟動工作階段時指定文件，請注意下列事項：
>
> - 您必須在使用者的 IAM 政策中授予使用者 `ssm:GetDocument` 和 `ssm:ListDocuments` 許可。如需詳細資訊，請參閱在主控台中授予對自訂工作階段文件的存取權。
> - 主控台僅支援 `sessionType` 定義為 `Standard_Stream` 的工作階段文件。如需詳細資訊，請參閱工作階段文件結構描述。

啟動工作階段 (Amazon EC2 主控台)

您可以使用 Amazon Elastic Compute Cloud (Amazon EC2) 主控台在您的帳戶中使用執行個體啟動工作階段。

> **Note**
>
> 若您收到錯誤，告知您並未獲得執行一或多個 Systems Manager 動作 (`ssm:command-name`) 的授權，您必須聯絡您的管理員以取得協助。您的管理員是為您提供簽署憑證的人員。請求該人員更新您的政策，允許您從 Amazon EC2 主控台啟動工作階段。如果您是管理員，請參閱適用於 Session Manager 的範例 IAM 政策 以取得更多資訊。

啟動工作階段 (Amazon EC2 主控台)

1. 在 https://console.aws.amazon.com/ec2/ 開啟 Amazon EC2 主控台。
2. 在導覽窗格中，選擇 Instances (執行個體)。
3. 選取執行個體，然後選取 Connect (連線)。
4. 對於連線方法，請選擇 Session Manager。
5. 選擇連線。

建立連線之後，您可以執行 bash 命令 (Linux 和 macOS) 或 PowerShell 命令 (Windows)，就像您使用任何其他連線類型時一樣。

啟動工作階段 (AWS CLI)

如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

開始工作階段之前，請確定您已完成 Session Manager 的設定步驟。如需相關資訊，請參閱 設定 Session Manager。

若要使用 AWS CLI 執行工作階段命令，您的本機機器也必須安裝 Session Manager外掛程式。如需相關資訊，請參閱 安裝 的Session Manager外掛程式 AWS CLI。

若要使用 啟動工作階段 AWS CLI，請執行下列命令，以您自己的資訊取代 *instance-id*。

```
aws ssm start-session \
    --target instance-id
```

如需有關可與 start-session 命令搭配使用的其他選項的資訊，請參閱 AWS CLI 命令參考的 start-session AWS Systems Manager 一節。

啟動工作階段 (SSH)

若要啟動 Session Manager SSH 工作階段，必須在受管節點上安裝 SSM Agent 2.3.672.0 版或更新版本。

SSH 連線需求

使用 SSH 建立工作階段連線時，請注意下列需求和限制：

- 您的目標受管節點必須設為支援 SSH 連線。如需詳細資訊，請參閱 (選用) 透過 Session Manager 允許和控制 SSH 連線的許可。
- 您必須使用與隱私權增強式郵件 (PEM) 憑證建立關聯的受管節點帳戶進行連線，而非用於其他工作階段連線類型的 `ssm-user` 帳戶。例如，在 Linux 和 macOS 的 EC2 執行個體上，預設使用者是 `ec2-user`。如需有關識別每種執行個體類型預設使用者的資訊，請參閱《Amazon EC2 使用者指南》中的 Get Information About Your Instance。
- 透過連接埠轉送或 SSH 連線的 Session Manager 工作階段無法使用日誌記錄功能。這是因為 SSH 會加密所有工作階段資料，Session Manager 僅用作 SSH 連線的通道。

> **Note**
>
> 開始工作階段之前，請確定您已完成 Session Manager 的設定步驟。如需相關資訊，請參閱設定 Session Manager。

若要使用 SSH 開始工作階段，請執行以下命令。將每個########取代為您自己的資訊。

```
ssh -i /path/my-key-pair.pem username@instance-id
```

> **Tip**
>
> 使用 SSH 開始工作階段時，您可以使用以下命令格式將本機檔案複製到目標受管節點。
>
> ```
> scp -i /path/my-key-pair.pem /path/ExampleFile.txt username@instance-id:~
> ```

如需有關可與 start-session 命令搭配使用的其他選項的資訊，請參閱 AWS CLI 命令參考的 start-session AWS Systems Manager 一節。

啟動工作階段 (網路埠轉遞)

若要啟動 Session Manager 連接埠轉送工作階段，必須在受管節點上安裝 SSM Agent 2.3.672.0 版或更新版本。

> **Note**
>
> 開始工作階段之前，請確定您已完成 Session Manager 的設定步驟。如需相關資訊，請參閱設定 Session Manager。
> 若要使用 AWS CLI 執行工作階段命令，您必須在本機機器上安裝Session Manager外掛程式。如需相關資訊，請參閱 安裝 的Session Manager外掛程式 AWS CLI。
> 視您的作業系統和命令列工具而定，引號的位置可能會有所不同，且可能需要逸出字元。

若要啟動網路埠轉遞工作階段，請從 CLI 執行以下命令。將每個########取代為您自己的資訊。

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` 代表受管節點上您要將工作階段流量重新導向至其中的遠端連接埠。例如，您可以指定連接埠 3389 用於透過遠端桌面通訊協定 (RDP) 連線至 Windows 節點。如果您未指定 `portNumber` 參數，Session Manager 會使用 80 作為預設值。

`localPortNumber` 是本機電腦上的連接埠 (例如 56789)，流量會從此開始。此值是您在使用用戶端連線到受管節點時所輸入的值。例如：**`localhost:56789`**。

如需有關可與 start-session 命令搭配使用的其他選項的資訊，請參閱 AWS CLI 命令參考的 start-session AWS Systems Manager 一節。

如需有關連接埠轉送工作階段的資訊，請參閱 AWS 新聞部落格中的使用 AWS Systems ManagerSession Manager 的連接埠轉送。

啟動工作階段 (連接埠轉送至遠端主機)

若要啟動 Session Manager 連接埠轉送工作階段至遠端主機，必須在受管理節點上安裝 SSM Agent 的 3.1.1374.0 版或較新版本。遠端主機不需由 Systems Manager 管理。

> **Note**
>
> 開始工作階段之前，請確定您已完成 Session Manager 的設定步驟。如需相關資訊，請參閱設定 Session Manager。
> 若要使用 AWS CLI 執行工作階段命令，您必須在本機機器上安裝Session Manager外掛程式。如需相關資訊，請參閱 安裝 的Session Manager外掛程式 AWS CLI。
> 視您的作業系統和命令列工具而定，引號的位置可能會有所不同，且可能需要逸出字元。

若要啟動連接埠轉遞工作階段，請從 AWS CLI執行以下命令。將每個########取代為您自己的資訊。

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 值代表您想要連線遠端主機的主機名稱或 IP 地址。受管理節點與遠端主機之間的一般連線與名稱解析要求仍然適用。

portNumber 代表受管節點上您要將工作階段流量重新導向至其中的遠端連接埠。例如，您可以指定連接埠 3389 用於透過遠端桌面通訊協定 (RDP) 連線至 Windows 節點。如果您未指定 portNumber 參數，Session Manager 會使用 80 作為預設值。

localPortNumber 是本機電腦上的連接埠 (例如 56789)，流量會從此開始。此值是您在使用用戶端連線到受管節點時所輸入的值。例如：**localhost:56789**。

如需有關可與 start-session 命令搭配使用的其他選項的資訊，請參閱 AWS CLI 命令參考的 start-session AWS Systems Manager 一節。

使用 Amazon ECS 任務啟動工作階段

Session Manager 支援在 Amazon Elastic Container Service (Amazon ECS) 叢集內使用任務啟動連接埠轉遞工作階段。若要這樣做，則必須在 IAM 中更新任務角色來包含下列許可：

```
{
   "Version": "2012-10-17",
   "Statement": [
       {
       "Effect": "Allow",
       "Action": [
            "ssmmessages:CreateControlChannel",
```

```
            "ssmmessages:CreateDataChannel",
            "ssmmessages:OpenControlChannel",
            "ssmmessages:OpenDataChannel"
        ],
       "Resource": "*"
       }
    ]
}
```

若要使用 Amazon ECS 任務啟動連接埠轉遞工作階段，請從 AWS CLI執行以下命令。將每個########取代為您自己的資訊。

> **Note**
>
> 將 < 和 > 符號從 target 參數中移除。這些符號只是為了方便讀者理解。

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"
```

啟動工作階段 (互動和非互動式命令)

開始工作階段之前，請確定您已完成 Session Manager 的設定步驟。如需相關資訊，請參閱 設定 Session Manager。

若要使用 AWS CLI 執行工作階段命令，您的本機機器也必須安裝 Session Manager外掛程式。如需相關資訊，請參閱 安裝 的Session Manager外掛程式 AWS CLI。

若要啟動互動式命令工作階段，請執行以下命令。將每個########取代為您自己的資訊。

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"
```

如需有關可與 start-session 命令搭配使用的其他選項的資訊，請參閱 AWS CLI 命令參考的 start-session AWS Systems Manager 一節。

詳細資訊

- 使用 AWS Systems Manager Session Manager 中的連接埠轉遞來連線至遠端主機
- 使用 轉送 Amazon EC2 執行個體連接埠 AWS Systems Manager
- 使用Session Manager連接埠轉送管理 AWS Managed Microsoft AD 資源
- AWS 新聞部落格中的使用 AWS Systems ManagerSession Manager 的連接埠轉送。

結束工作階段

您可以使用 AWS Systems Manager 主控台或 AWS Command Line Interface () 結束您在帳戶中啟動的工作階段AWS CLI。當您在主控台中選擇工作階段的終止按鈕或使用 呼叫 TerminateSession API 動作時 AWS CLI， 會Session Manager永久結束工作階段，並關閉Session Manager用戶端與受管節點 SSM Agent上的資料連線。您無法繼續已終止的工作階段。

如果開啟的工作階段中 20 分鐘內沒有使用者活動，閒置狀態會觸發逾時。 Session Manager不會呼叫 `TerminateSession`，但會關閉基礎頻道。您無法繼續因閒置逾時而關閉的工作階段。

我們建議一律使用 `terminate-session`命令、使用 時，或使用 主控台時 AWS CLI，使用 終止按鈕明確終止工作階段。(終止按鈕位於工作階段視窗和主Session Manager主控台頁面。) 如果您只關閉瀏覽器或命令視窗，工作階段會在主控台中保持為作用中狀態 30 天。當您未明確終止工作階段，或工作階段逾時時，在受管節點上執行的任何程序都會繼續執行。

主題

- 結束工作階段 (主控台)
- 結束工作階段 (AWS CLI)

結束工作階段 (主控台)

您可以使用 AWS Systems Manager 主控台來結束帳戶中的工作階段。

結束工作階段 (主控台)

1. 開啟 AWS Systems Manager 主控台，網址為 https://console.aws.amazon.com/systems-manager/：//。
2. 在導覽窗格中，選擇 Session Manager。
3. 針對 Sessions (工作階段)，請選擇您要結束的工作階段左側的選項按鈕。
4. 選擇終止。

結束工作階段 (AWS CLI)

若要使用 結束工作階段 AWS CLI，請執行下列命令。將 *session-id* 取代為您自己的資訊。

```
aws ssm terminate-session \
    --session-id session-id
```

如需 terminate-session命令的詳細資訊，請參閱terminate-session《 AWS CLI 命令參考》的 AWS Systems Manager 一節。

檢視工作階段歷史記錄

您可以使用 AWS Systems Manager 主控台或 AWS Command Line Interface (AWS CLI) 來檢視帳戶中工作階段的相關資訊。在主控台，您可以查看工作階段的詳細資訊，如下所示：

- 工作階段的 ID
- 哪個使用者透過工作階段連線至受管節點
- 受管節點的 ID
- 工作階段何時開始和結束
- 工作階段的狀態。
- 指定儲存工作階段日誌的位置 (如果已開啟)

使用 AWS CLI，您可以檢視帳戶中的工作階段清單，但無法檢視主控台中可用的其他詳細資訊。

如需有關記錄工作階段歷史記錄資訊的詳細資訊，請參閱 啟用和停用工作階段記錄。

主題

- 檢視工作階段歷史記錄 (主控台)
- 檢視工作階段歷史記錄 (AWS CLI)

檢視工作階段歷史記錄 (主控台)

您可以使用 AWS Systems Manager 主控台來檢視您帳戶中工作階段的詳細資訊。

檢視工作階段歷史記錄 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Session Manager。
3. 選擇 Session history (工作階段歷史記錄) 標籤。

   -或-

   如果 Session Manager 首頁先開啟，請選擇設定偏好設定，然後選擇工作階段歷史記錄索引標籤。

檢視工作階段歷史記錄 (AWS CLI)

若要使用 檢視您帳戶中的工作階段清單 AWS CLI，請執行下列命令。

```
aws ssm describe-sessions \
    --state History
```

> **Note**
>
> 此命令只會傳回使用 Session Manager 啟動之目標的連線結果。它不會列出透過其他方式建立的連線，例如遠端桌面通訊協定 (RDP) 或 Secure Shell (SSH) 協定。

如需有關可與 describe-sessions 命令搭配使用的其他選項的資訊，請參閱 AWS CLI 命令參考的 describe-sessions AWS Systems Manager 一節。

記錄工作階段活動

除了提供有關在 Systems Manager 主控台目前和完成工作階段的相關資訊，Session Manager 還可使用 AWS CloudTrail為您提供在 AWS 帳戶 中記錄工作階段活動的能力。

CloudTrail 透過 Systems Manager 主控台、 AWS Command Line Interface (AWS CLI) 和 Systems Manager SDK 擷取工作階段 API 呼叫。您可以在 CloudTrail 主控台中檢視資訊，或將它存放在指定的 Amazon Simple Storage Service (Amazon S3) 儲存貯體中。一個 Amazon Simple Storage Service (Amazon S3) 儲存貯體可用於您帳戶中的所有 CloudTrail 日誌。如需詳細資訊，請參閱使用 記錄 AWS Systems Manager API 呼叫 AWS CloudTrail。

> **Note**
>
> 對於日誌檔案的重複性與歷史性分析，請考慮使用 CloudTrail Lake 或自己維護的資料表查詢 CloudTrail 日誌。如需詳細資訊，請參閱AWS CloudTrail 《 使用者指南》中的查詢 AWS CloudTrail 日誌。

使用 Amazon EventBridge (主控台) 監控工作階段活動

使用 EventBridge，您可以設定規則來偵測 AWS 資源何時發生變更。您可以建立一個規則來偵測當您的組織中的使用者開始或結束工作階段，然後例如透過 Amazon SNS 接收到有關事件的通知。

Session Manager 的 EventBridge 支援依賴於 CloudTrail 所記錄的 API 操作的記錄。(您可以使用 CloudTrail 整合 EventBridge，以回應大多數 AWS Systems Manager 事件。) 在工作階段中執行的動作，例如 `exit` 命令，不進行 API 呼叫，不會被 EventBridge 偵測到。

以下步驟概述當 Session Manager API 事件發生時如何透過 Amazon Simple Notification Service (Amazon SNS) 啟動通知，例如 StartSession。

使用 Amazon EventBridge (主控台) 監控工作階段活動

1. 建立一個 Amazon SNS 的主題，此主題用來傳送您所追蹤的 Session Manager 事件發生的提醒。

   如需詳細資訊，請參閱 Amazon Simple Notification Service 開發人員指南中的建立主題。
2. 建立 EventBridge 規則來叫使用 Amazon SNS 目標你想追蹤的 Session Manager 事件類型。

   如需有關建立規則的資訊，請參閱 Amazon EventBridge 使用者指南中的建立對事件做出反應的 Amazon EventBridge 規則。

在依照步驟建立角色時，請選擇下列選項：

- 針對 AWS service (服務)，請選擇 Systems Manager。
- 針對 Event type (事件類型)，選擇 AWS API Call via CloudTrail (透過 CloudTrail 的 API 呼叫)。
- 選擇 Specific operation(s) (特定操作)，然後輸入 Session Manager 命令或您希望接收到通知的命令 (一次一個)。您也可以選擇 StartSession、ResumeSession 和 TerminateSession。(EventBridge 不支援 `Get*`、 `List*` 和 `Describe*` 命令。)
- 對於 Select a target (選取目標)，選擇 SNS topic (SNS 主題)。在 Topic (主題) 中，選擇您在步驟 1 建立的 Amazon SNS 主題的名稱

如需詳細資訊，請參閱《Amazon EventBridge 使用者指南》和《Amazon Simple Notification Service 入門指南》。

啟用和停用工作階段記錄

工作階段記錄會在 Systems Manager 主控台中記錄目前和已完成工作階段的相關資訊。您也可以在 AWS 帳戶中記錄工作階段期間執行之命令的詳細資訊。工作階段記錄可讓您執行以下動作：

- 建立和存放工作階段日誌封存之用。
- 使用 Session Manager 產生報告顯示在過去 30 天每個連接到您的受管節點的詳細資訊。
- 在 中產生工作階段記錄的通知 AWS 帳戶，例如 Amazon Simple Notification Service (Amazon SNS) 通知。
- 由於在工作階段期間執行的動作，例如執行 AWS Lambda 函數、啟動 AWS CodePipeline 管道或執行 AWS Systems Manager Run Command 文件，自動對 AWS 資源啟動另一個動作。

Important

請注意下列 Session Manager 的需求和限制：

- Session Manager 會根據您的工作階段偏好，記錄您在工作階段期間輸入的命令及其輸出。若要防止在工作階段日誌中檢視敏感資料 (例如密碼)，建議您在工作階段期間輸入敏感資料時，使用下列命令。

  Linux & macOS

  ```
  stty -echo; read passwd; stty echo;
  ```

Windows

```
$Passwd = Read-Host -AsSecureString
```

- 如果您使用 Windows Server 2012 或更低版本，可能不會在您的日誌資料進行以最佳方式格式化。為了有最佳的日誌格式，我們建議您使用 Windows Server 2012 R2 和更高版本。
- 如果您使用的是 Linux 或 macOS 受管節點，則請確保有安裝熒幕公用程式。如果沒有，您的日誌資料可能被截斷過。在 Amazon Linux 1、Amazon Linux 2、AL2023 和 Ubuntu Server 上，預設會安裝螢幕公用程式。若要手動安裝熒幕公用程式，根據於您的 Linux 版本而定，請執行 `sudo yum install screen` 或 `sudo apt-get install screen`。
- 透過連接埠轉送或 SSH 連線的 Session Manager 工作階段無法使用日誌記錄功能。這是因為 SSH 會加密所有工作階段資料，Session Manager 僅用作 SSH 連線的通道。

如需有關使用 Amazon Simple Storage Service (Amazon S3) 或 Amazon CloudWatch Logs 來記錄工作階段資料所需許可的詳細資訊，請參閱建立具有 Session Manager、Amazon Simple Storage Service (Amazon S3) 和 CloudWatch Logs (主控台) 許可的 IAM 角色。

如需有關 Session Manager 的記錄選項，請參閱下列主題。

主題

- 使用 Amazon CloudWatch Logs (主控台) 串流工作階段資料。
- 使用 Amazon Simple Storage Service (Amazon S3) (主控台) 記錄工作階段資料
- 使用 Amazon CloudWatch Logs (主控台) 記錄工作階段資料。
- 設定磁碟的工作階段記錄
- 調整 Session Manager 暫時日誌檔案在磁碟中的存放時間
- 停用 CloudWatch Logs 和 Amazon S3 中的 Session Manager 記錄

使用 Amazon CloudWatch Logs (主控台) 串流工作階段資料。

您可以將持續的工作階段資料日誌串流傳送到 Amazon CloudWatch Logs。串流工作階段資料時會包含基本詳細資料，例如使用者在工作階段中執行的命令、執行命令的使用者 ID，以及工作階段資料串流至 CloudWatch Logs 時的時間戳記。串流工作階段資料時，日誌會採用 JSON 格式，以協助您與現有的日誌解決方案進行整合。互動式命令不支援串流工作階段資料。

> **Note**
>
> 若要從 Windows Server 受管節點中串流工作階段資料，必須安裝 PowerShell 5.1 或更新版本。根據預設，Windows Server 2016 及更新版本已安裝必要的 PowerShell 版本。但是，根據預設，Windows Server 2012 和 2012 R2 沒有安裝必要的 PowerShell 版本。如果您尚未在 Windows Server 2012 或 2012 R2 受管節點上更新 PowerShell，您可以使用 Run Command 完成。如需有關使用 Run Command 更新 PowerShell 的資訊，請參閱 使用 Run Command 更新 PowerShell。

> **Important**
>
> 如果您已在 Windows Server 受管節點上配置 PowerShell 轉錄政策設定，您將無法串流工作階段資料。

若要使用 Amazon CloudWatch Logs (主控台) 串流工作階段資料

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Session Manager。
3. 選擇 Preferences (偏好) 標籤，然後選擇 Edit (編輯)。
4. 在 CloudWatch logging (CloudWatch 記錄) 下，選取 Enable (啟用) 旁的核取方塊。
5. 選擇 Stream session logs (串流工作階段日誌) 選項。
6. (建議) 選取 Allow only encrypted CloudWatch log groups (只允許加密的 CloudWatch 日誌群組) 旁的核取方塊。開啟此選項後，系統會使用為該日誌群組指定的伺服器端加密金鑰來加密日誌資料。如果您不想加密要傳送到 CloudWatch Logs 中的日誌資料，清除核取方塊。如果日誌群組不允許加密，您必須同時清除核取方塊。
7. 對於 CloudWatch 日誌，若要在 中指定 AWS 帳戶 要上傳工作階段日誌的現有 CloudWatch Logs 日誌群組，請選取下列其中一項：
   - 在文字方塊中輸入日誌群組，此用戶群組已在帳戶中建立用於存放工作階段日誌資料。
   - Browse log groups (瀏覽日誌群組)：從清單中選擇一個日誌群組名稱 ：選取已在您的帳戶中建立的日誌群組存放工作階段的日誌資料。
8. 選擇 Save (儲存)。

使用 Amazon Simple Storage Service (Amazon S3) (主控台) 記錄工作階段資料

您可以選擇將工作階段日誌資料存放在指定的 Amazon Simple Storage Service (Amazon S3) 儲存貯體中，用於偵錯和疑難排解。預設選項適用於要傳送至加密 Amazon Simple Storage Service (Amazon S3) 儲存貯體的日誌。使用為儲存貯體指定的金鑰來執行加密，可以是 AWS KMS key 或 Amazon S3 伺服器端加密 (SSE) 金鑰 (AES-256)。

> **Important**
>
> 當您使用虛擬託管型儲存貯體與 Secure Sockets Layer (SSL) 時，SSL 萬用字元憑證只符合不包含句點的儲存貯體。若要解決這個問題，請使用 HTTP 或撰寫您自己的憑證驗證邏輯。我們建議您在使用虛擬託管型的儲存貯體時，不要在儲存貯體名稱中使用句號 (".")。

Amazon Simple Storage Service (Amazon S3) 儲存貯體加密

為了傳送加密日誌到您的 Amazon Simple Storage Service (Amazon S3) 儲存貯體、加密與加密必須啟用儲存貯體。如需 Amazon Simple Storage Service (Amazon S3) 儲存貯體加密的詳細資訊，請參閱 S3 儲存貯體的 Amazon Simple Storage Service (Amazon S3) 預設加密。

客戶受管金鑰

如果您使用 KMS 金鑰，您管理自己來加密您的儲存貯體，然後 IAM 執行個體設定檔連接到您的執行個體必須擁有明確權限以讀取金輪。如果您使用 AWS 受管金鑰，執行個體不需要此明確許可。如需有關提供執行個體設定檔存取使用金鑰的資訊，請參閱《AWS Key Management Service 開發人員指南》中的允許金輪使用者使用金輪。

依照以下步驟來設定 Session Manager 以便在 Amazon Simple Storage Service (Amazon S3) 儲存貯體存放工作階段的日誌。

> **Note**
>
> 您也可以使用 AWS CLI 來指定或變更工作階段資料傳送到的 Amazon S3 儲存貯體。如需相關資訊，請參閱更新 Session Manager 偏好設定 (命令列)。

使用 Amazon Simple Storage Service (Amazon S3) (主控台) 記錄工作階段資料

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。

2. 在導覽窗格中，選擇 Session Manager。
3. 選擇 Preferences (偏好) 標籤，然後選擇 Edit (編輯)。
4. 在 S3 logging (S3 日誌) 下，選取 Enable (啟用) 旁的核取方塊。
5. (建議) 選取 Allow only encrypted S3 buckets (只允許加密的 S3 儲存貯體) 旁的核取方塊。開啟此選項後，系統會使用為該儲存貯體指定的伺服器端加密金鑰來加密日誌資料。如果您不想加密要傳送到 Amazon Simple Storage Service (Amazon S3) 中的日誌資料，清除核取方塊。如果 S3 儲存貯體不允許加密，您必須同時清除核取方塊。
6. S3 bucket name (S3 儲存貯體名稱)，選擇下列其中一項作業：

   > **Note**
   >
   > 我們建議您在使用虛擬託管型的儲存貯體時，不要在儲存貯體名稱中使用句號 (".")。如需有關 Amazon Simple Storage Service (Amazon S3) 儲存貯體命名慣例的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的儲存貯體法規與限制。

   - Choose a bucket name from the list (清單中選擇一個儲存貯體名稱)：選取已在您帳戶中建立的 Amazon Simple Storage Service (Amazon S3) 儲存貯體來存放工作階段的日誌資料。
   - Enter a bucket name in the text box (在文字方塊中輸入儲存貯體名稱)：輸入已在帳戶中建立的 Amazon Simple Storage Service (Amazon S3) 儲存貯體名稱來存放工作階段日誌資料。
7. (選用) S3 key prefix (金鑰字首)，輸入現有的名稱或新的資料夾將日誌存放在所選的儲存貯體。
8. 選擇儲存。

如需有關使用 Amazon Simple Storage Service (Amazon S3) 和 Amazon Simple Storage Service (Amazon S3) 儲存貯體的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》和《Amazon Simple Storage Service 使用者指南》。

使用 Amazon CloudWatch Logs (主控台) 記錄工作階段資料。

您可使用 Amazon CloudWatch Logs 從各種 AWS 服務中監控、存放及存取日誌檔案。您可以將工作階段日誌資料傳送到 CloudWatch Logs 日誌群組，用於偵錯和疑難排解。預設選項是要使用 KMS 金鑰來加密要傳送的日誌資料，但您可以將資料加密或不加密下傳送到日誌群組。

請依照下列步驟設定 AWS Systems Manager Session Manager ，在工作階段結束時將工作階段日誌資料傳送至 CloudWatch Logs 日誌群組。

> **Note**
>
> 您也可以使用 AWS CLI 來指定或變更工作階段資料傳送到的 CloudWatch Logs 日誌群組。如需相關資訊，請參閱 更新 Session Manager 偏好設定 (命令列)。

使用 Amazon CloudWatch Logs (主控台) 記錄工作階段的日誌資料。

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Session Manager。
3. 選擇 Preferences (偏好) 標籤，然後選擇 Edit (編輯)。
4. 在 CloudWatch logging (CloudWatch 記錄) 下，選取 Enable (啟用) 旁的核取方塊。
5. 選擇 Upload session logs (上傳工作階段日誌) 選項。
6. (建議) 選取 Allow only encrypted CloudWatch log groups (只允許加密的 CloudWatch 日誌群組) 旁的核取方塊。開啟此選項後，系統會使用為該日誌群組指定的伺服器端加密金鑰來加密日誌資料。如果您不想加密要傳送到 CloudWatch Logs 中的日誌資料，清除核取方塊。如果日誌群組不允許加密，您必須同時清除核取方塊。
7. 對於 CloudWatch 日誌，若要在 中指定 AWS 帳戶 要上傳工作階段日誌的現有 CloudWatch Logs 日誌群組，請選取下列其中一項：
   - 從清單中選擇一個日誌群組名稱 ：選取已在您的帳戶中建立的日誌群組存放工作階段的日誌資料。
   - 在文字方塊中輸入日誌群組名稱：輸入已在帳戶中建立的日誌群組名稱來存放工作階段日誌資料。
8. 選擇 Save (儲存)。

如需有關 CloudWatch Logs 的詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》。

設定磁碟的工作階段記錄

在啟用 CloudWatch 或 Amazon S3 的 Session Manager 記錄之後，工作階段期間執行的所有命令 (以及那些命令產生的輸出) 都會記錄到目標執行個體磁碟上的暫時檔案。暫時檔案名為 `ipcTempFile.log`。

由 SSM Agent 組態檔案中的 `SessionLogsDestination` 參數控制 `ipcTempFile.log`。此參數接受以下值：

- 磁碟：如果指定此參數且 CloudWatch 或 Amazon S3 的工作階段記錄已啟用，SSM Agent 會建立 `ipcTempFile.log` 暫時日誌檔案，並將工作階段命令和輸出記錄到磁碟。Session Manager 會根據記錄組態，在工作階段期間或之後將此日誌上傳至 CloudWatch 或 S3。然後，系統會根據為 SSM Agent `SessionLogsRetentionDurationHours` 組態參數指定的持續時間刪除日誌。

  如果指定此參數且 CloudWatch 和 Amazon S3 的工作階段記錄已停用，SSM Agent 仍會記錄 `ipcTempFile.log` 檔案中的命令歷史記錄和輸出。系統會根據為 SSM Agent `SessionLogsRetentionDurationHours` 組態參數指定的持續時間刪除此檔案。
- 無：如果指定此參數且 CloudWatch 或 Amazon S3 的工作階段記錄已啟用，則記錄到磁碟的運作方式與指定 `disk` 參數的方式完全相同。SSM Agent 會在啟用 CloudWatch 或 Amazon S3 的工作階段記錄時需要暫時檔案。

  如果指定此參數且 CloudWatch 或 Amazon S3 的工作階段記錄已停用，SSM Agent 不會建立 `ipcTempFile.log` 檔案。

按照下列程序，在工作階段開始時啟用或停用在磁碟中建立 `ipcTempFile.log` 暫時日誌檔案。

啟用或停用在磁碟中建立 Session Manager 暫時日誌檔案

1. 在執行個體上安裝 SSM Agent 或升級至 3.2.2086 版或更新版本。如需如何檢查代理程式版本編號的資訊，請參閱檢查 SSM Agent 版本編號。如需如何手動安裝代理程式的資訊，請在下列各節尋找適合作業系統的程序：

   - 在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent
   - 在 macOS EC2 執行個體上手動安裝和解除安裝 SSM Agent
   - 在 Windows Server EC2 執行個體上手動安裝和解除安裝 SSM Agent
2. 連線至執行個體，並在下列位置找到 `amazon-ssm-agent.json` 檔案。

   - Linux：/etc/amazon/ssm/
   - macOS：/opt/aws/ssm/
   - Windows Server：C:\Program Files\Amazon\SSM

   如果 `amazon-ssm-agent.json` 檔案不存在，請將 `amazon-ssm-agent.json.template` 的內容複製到同一目錄的新檔案中。將此新檔案命名為 `amazon-ssm-agent.json`。
3. 為 `SessionLogsDestination` 參數指定 `none` 或 `disk`。儲存您的變更。
4. 重新啟動 SSM Agent。

如果為 SessionLogsDestination 參數指定 disk，則可以啟動新的工作階段，然後將 ipcTempFile.log 放置在下列位置，即可驗證 SSM Agent 是否建立暫時日誌檔案：

- Linux：/var/lib/amazon/ssm/*target ID*/session/orchestration/*session ID*/Standard_Stream/ipcTempFile.log
- macOS：/opt/aws/ssm/data/*target ID*/session/orchestration/*session ID*/Standard_Stream/ipcTempFile.log
- Windows Server：C:\ProgramData\Amazon\SSM\InstanceData*target ID*\session\orchestration*session ID*\Standard_Stream\ipcTempFile.log

> **Note**
>
> 暫時日誌檔案預設會在執行個體上儲存 14 天的時間。

若要跨多個執行個體更新 SessionLogsDestination 參數，建議建立可指定新組態的 SSM 文件。然後，您可以使用 Systems Manager Run Command 在執行個體實作變更。如需詳細資訊，請參閱撰寫您自己的 AWS Systems Manager 文件（部落格）和 在受管節點上執行命令。

調整 Session Manager 暫時日誌檔案在磁碟中的存放時間

在啟用 CloudWatch 或 Amazon S3 的 Session Manager 記錄之後，工作階段期間執行的所有命令 (以及那些命令產生的輸出) 都會記錄到目標執行個體磁碟上的暫時檔案。暫時檔案名為 ipcTempFile.log。在工作階段期間或完成後，Session Manager 會將此暫時日誌上傳至 CloudWatch 或 S3。之後，系統會根據為 SSM Agent SessionLogsRetentionDurationHours 組態參數指定的持續時間刪除暫時日誌。暫時日誌檔案預設會在執行個體上儲存 14 天的時間，位置如下：

- Linux：/var/lib/amazon/ssm/*target ID*/session/orchestration/*session ID*/Standard_Stream/ipcTempFile.log
- macOS：/opt/aws/ssm/data/*target ID*/session/orchestration/*session ID*/Standard_Stream/ipcTempFile.log
- Windows Server：C:\ProgramData\Amazon\SSM\InstanceData*target ID*\session\orchestration*session ID*\Standard_Stream\ipcTempFile.log

按照下列程序調整 Session Manager 暫時日誌檔案在磁碟中的存放時間。

調整 `ipcTempFile.log` 檔案在磁碟中的存放時間

1. 連線至執行個體，並在下列位置找到 `amazon-ssm-agent.json` 檔案。

   - Linux：/etc/amazon/ssm/
   - macOS：/opt/aws/ssm/
   - Windows Server：C:\Program Files\Amazon\SSM

   如果 `amazon-ssm-agent.json` 檔案不存在，請將 `amazon-ssm-agent.json.template` 的內容複製到同一目錄的新檔案中。將此新檔案命名為 `amazon-ssm-agent.json`。
2. 將 `SessionLogsRetentionDurationHours` 的值變更為所需的時數。如果將 `SessionLogsRetentionDurationHours` 設為 0，系統會在工作階段期間建立暫時日誌檔案，並在工作階段完成時將其刪除。此設定應確保日誌檔案在工作階段結束後不會持續存在。
3. 儲存您的變更。
4. 重新啟動 SSM Agent。

停用 CloudWatch Logs 和 Amazon S3 中的 Session Manager 記錄

您可以使用 Systems Manager 主控台或 AWS CLI 來停用帳戶中的工作階段記錄。

停用工作階段記錄 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Session Manager。
3. 選擇 Preferences (偏好) 標籤，然後選擇 Edit (編輯)。
4. 若要停用 CloudWatch 記錄功能，請在 CloudWatch 記錄區段中取消選取啟用核取方塊。
5. 若要停用 S3 記錄功能，請在 S3 記錄區段中取消選取啟用核取方塊。
6. 選擇 Save (儲存)。

停用工作階段記錄 (AWS CLI)

若要使用 停用工作階段記錄 AWS CLI，請遵循 中的指示更新 Session Manager 偏好設定 (命令列)。

在 JSON 檔案中，請確保 `s3BucketName` 和 `cloudWatchLogGroupName` 輸入不包含任何值。例如：

```
"inputs": {
        "s3BucketName": "",
        ...
        "cloudWatchLogGroupName": "",
        ...
    }
```

您也可以透過移除 JSON 檔案中的所有 `S3*` 和 `cloudWatch*` 輸入來停用記錄功能。

> **Note**
>
> 停用 CloudWatch 或 S3 之後，SSM Agent 仍可能會在磁碟中產生暫時日誌檔案，具體視組態而定。如需有關如何停用磁碟記錄的資訊，請參閱設定磁碟的工作階段記錄。

工作階段文件結構描述

以下資訊說明工作階段文件的結構描述元素。 AWS Systems Manager Session Manager使用工作階段文件來決定要啟動的工作階段類型，例如標準工作階段、連接埠轉送工作階段或工作階段，以執行互動式命令。

schemaVersion

工作階段文件的結構描述版本。工作階段文件僅支援 1.0 版。

類型：字串

必要：是

description

您為工作階段文件指定的描述。例如，"Document to start port forwarding session with Session Manager" (使用 Session Manager 啟動連接埠轉送工作階段的文件)。

類型：字串

必要：否

sessionType

工作階段文件用來建立的工作階段類型。

類型：字串

必要：是

有效值：InteractiveCommands | NonInteractiveCommands | Port | Standard_Stream

inputs

用於使用此工作階段文件建立的工作階段的工作階段偏好設定。用來建立 Standard_Stream 工作階段的工作階段文件需要此元素。

類型：StringMap

必要：否

s3BucketName

在工作階段結束時，接收工作階段日誌的 Amazon Simple Storage Service (Amazon S3) 儲存貯體。

類型：字串

必要：否

s3KeyPrefix

將日誌傳送到在 s3BucketName 輸入中指定的 Amazon Simple Storage Service (Amazon S3) 儲存貯體時使用的字首。如需有關搭配使用公用字首與 Amazon Simple Storage Service (Amazon S3) 中存放的物件的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的如何使用 S3 儲存貯體中的資料夾？。

類型：字串

必要：否

s3EncryptionEnabled

如果設定為 true，則必須加密在 s3BucketName 輸入中指定的 Amazon Simple Storage Service (Amazon S3) 儲存貯體。

類型：布林值

必要：是

cloudWatchLogGroupName

在工作階段結束時接收工作階段日誌的 Amazon CloudWatch Logs (CloudWatch Logs) 群組的名稱。

類型：字串

必要：否

cloudWatchEncryptionEnabled

如果設定為 `true`，則必須加密在 `cloudWatchLogGroupName` 輸入中指定的 日誌群組。

類型：布林值

必要：是

cloudWatchStreamingEnabled

如果設定為 `true`，工作階段資料日誌的持續串流會傳送到您在 `cloudWatchLogGroupName` 輸入中指定的日誌群組。如果設定為 `false`，在工作階段結束時，工作階段日誌會傳送到您在 `cloudWatchLogGroupName` 輸入中指定的日誌群組。

類型：布林值

必要：是

kmsKeyId

AWS KMS key 您要用來進一步加密本機用戶端機器與連線的 Amazon Elastic Compute Cloud (Amazon EC2) 受管節點之間的資料 ID。

類型：字串

必要：否

runAsEnabled

如果設定為 `true`，則必須在 `runAsDefaultUser` 輸入中指定您將連線的受管節點上存在的使用者帳戶。否則，工作階段將無法啟動。根據預設，使用 AWS Systems Manager SSM Agent 建立的 `ssm-user` 帳戶來啟動工作階段。執行身分功能僅支援連線至 Linux和 macOS受管節點。

類型：布林值

必要：是

runAsDefaultUser

當`runAsEnabled`輸入設定為 時，在 Linux和macOS受管節點上啟動工作階段的使用者帳戶名稱`true`。您為此輸入指定的使用者帳戶必須存在於您要連線的受管節點上；否則，工作階段將無法啟動。

類型：字串

必要：否

idleSessionTimeout

在工作階段結束前，您想要允許的閒置時間長度。此輸入的測量單位為分鐘。

類型：字串

有效值：1-60

必要：否

maxSessionDuration

在工作階段結束前，您想要允許的最大時間長度。此輸入的測量單位為分鐘。

類型：字串

有效值：1-1440

必要：否

shellProfile

啟動工作階段時，您根據作業系統指定的套用在工作階段中的偏好設定，例如 shell 偏好設定、環境變數、工作目錄以及執行的多個命令。

類型：StringMap

必要：否

windows

您為 Windows Server 受管節點上的工作階段指定的 shell 偏好設定、環境變數、工作目錄和命令。

類型：字串

必要：否

linux

您為 和 macOS受管節點上的工作階段指定的 shell 偏好設定、環境變數、工作目錄Linux和命令。

類型：字串

必要：否

parameters

一種物件，定義文件接受的參數。如需指定文件參數的詳細資訊，請參閱 頂層資料元素 中的參數。對於常用的參數，我們建議您將這些參數存放在 Systems Manager Parameter Store，然後參考使用。您在文件的這部分可以參考 `String` 和 `StringList` Parameter Store 參數。您在文件的這部分不能參考 `SecureString` Parameter Store 參數。您可以使用下列格式參考 Parameter Store 參數。

```
{{ssm:parameter-name}}
```

如需有關 Parameter Store 的詳細資訊，請參閱「AWS Systems Manager Parameter Store」。

類型：StringMap

必要：否

properties

一個物件，在 `StartSession` API 操作中使用您指定的其值。

對於用於 `InteractiveCommands` 工作階段的工作階段文件，屬性物件會包含要在您指定的作業系統上執行的命令。您也可以使用 `runAsElevated` 布林值屬性決定命令是否以 `root` 執行。如需詳細資訊，請參閱限制對工作階段中命令之存取權。

對於用於 `Port` 工作階段的工作階段文件，屬性物件會包含應將流量重新引導的目標連接埠號碼。如需範例，請參閱本主題後面部分中的 `Port` 類型工作階段文件範例。

類型：StringMap

必要：否

`Standard_Stream` 類型工作階段文件範例

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 類型工作階段文件範例

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 類型工作階段文件範例

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"
    }
}
```

含特殊字元的工作階段文件範例

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
      }
   }
}
```

Session Manager 疑難排解

使用以下資訊以協助您對 AWS Systems Manager Session Manager 的問題進行故障診斷。

主題

- 呼叫 TerminateSession 操作時的 AccessDeniedException
- 文件程序意外失敗：文件工作者逾時

- Session Manager 無法從 Amazon EC2 主控台連線
- 無權啟動工作階段
- SSM Agent不在線
- 無權變更工作階段偏好設定
- 受管節點無法使用或未設定用於 Session Manager
- 找不到 Session Manager 外掛程式
- Session Manager 外掛程式未自動新增到命令列路徑 (Windows)
- Session Manager 外掛程式無回應
- TargetNotConnected
- 啟動工作階段後顯示空白畫面
- 受管節點在長時間執行工作階段期間變得沒有回應
- 呼叫 StartSession 操作時發生錯誤 (InvalidDocument)

呼叫 TerminateSession 操作時的 AccessDeniedException

問題：嘗試終止工作階段時，Systems Manager 會傳回下列錯誤：

```
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.
```

解決方案 A：確認節點上安裝最新版本的Session Manager外掛程式

在終端機中輸入下列命令，然後按 Enter。

```
session-manager-plugin --version
```

解決方案 B：安裝或重新安裝最新版本的外掛程式

如需詳細資訊，請參閱安裝 的Session Manager外掛程式 AWS CLI。

解決方案 C：嘗試重新建立節點的連線

確認節點正在回應請求。請嘗試重新建立工作階段。或者，如有必要，請開啟 Amazon EC2 主控台並確認執行個體正在執行狀態。

文件程序意外失敗：文件工作者逾時

問題：在 Linux 主機啟動工作階段時，Systems Manager 會傳回下列錯誤：

```
document process failed unexpectedly: document worker timed out,
check [ssm-document-worker]/[ssm-session-worker] log for crash reason
```

如果按 檢視 SSM Agent 日誌 中所述設定 SSM Agent 記錄，即可在偵錯日誌中檢視更多詳細資訊。針對此問題，Session Manager 會顯示下列日誌項目：

```
failed to create channel: too many open files
```

此錯誤通常表示執行的 Session Manager 工作者程序太多，且基礎作業系統已達到限制。有兩個選項可用來解決此問題。

解決方案 A：提高作業系統檔案通知限制

您可以從個別 Linux 主機執行下列命令來提高限制。此命令使用 Systems Manager Run Command。指定的值會將 `max_user_instances` 提高至 8192。此值遠高於預設值 128，但不會對主機資源造成壓力：

```
aws ssm send-command --document-name AWS-RunShellScript \
--instance-id i-02573cafcfEXAMPLE  --parameters \
"commands=sudo sysctl fs.inotify.max_user_instances=8192"
```

解決方案 B：減少目標主機中 Session Manager 使用的檔案通知

從個別 Linux 主機執行下列命令，列出在目標主機上執行的工作階段：

```
aws ssm describe-sessions --state Active --filters key=Target,value=i-02573cafcfEXAMPLE
```

檢閱命令輸出，識別不再需要的工作階段。您可以從個別 Linux 主機執行下列命令來終止這些工作階段：

```
aws ssm terminate-session –session-id session ID
```

或者，只要遠端伺服器上不再有執行中的工作階段，您就可以從個別 Linux 主機執行下列命令來釋出其他資源。此命令會終止在遠端主機上執行的所有 Session Manager 程序，從而終止遠端主機的所有工作階段。在執行此命令之前，請確認是否有要保留的持續工作階段：

```
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 無法從 Amazon EC2 主控台連線

問題：建立新執行個體之後，Amazon Elastic Compute Cloud (Amazon EC2) 主控台中的 Connect 按鈕 > Session Manager 索引標籤不會提供您連線的選項。

解決方案 A：建立執行個體描述檔：如果您尚未這樣做（如 EC2 主控台中 Session Manager 索引標籤上的資訊指示），請使用 建立 AWS Identity and Access Management (IAM) 執行個體描述檔Quick Setup。 Quick Setup 是 中的工具 AWS Systems Manager。

Session Manager 需要 IAM 執行個體設定檔才能連線到執行個體。您可以使用 Quick Setup 建立主機管理組態，然後再建立執行個體設定檔並將其指派給執行個體。主機管理組態會建立具有必要許可的執行個體設定檔，並將其指派給執行個體。主機管理組態也會啟用其他 Systems Manager 工具，並建立執行這些工具的 IAM 角色。使用 Quick Setup或主機管理組態啟用的工具無需付費。開啟 Quick Setup 並建立主機管理組態。

> **Important**
>
> 建立主機管理組態後，Amazon EC2 可能需要幾分鐘的時間來登記變更並重新整理 Session Manager 索引標籤。如果在兩分鐘後索引標籤中仍未顯示連線按鈕，請重新啟動執行個體。重新啟動後，如果仍然看不到連線選項，請開啟 Quick Setup，確認僅存在一個主機管理組態。如果有兩個，請刪除較舊的組態並等待幾分鐘。

如果在建立主機管理組態後仍無法連線，或是收到錯誤訊息 (包含 SSM Agent 相關錯誤)，請參閱下列其中一個解決方案：

- 解決方案 B：沒有錯誤，但仍無法連線
- 解決方案 C：關於遺失 SSM Agent 的錯誤

解決方案 B：沒有錯誤，但仍無法連線

如果您已建立主機管理組態，而在等待數分鐘後再嘗試連線，但仍無法連線，則您可能需要手動將主機管理組態套用至執行個體。請遵循以下程序以更新 Quick Setup 主機管理組態，並將變更套用至執行個體。

使用 Quick Setup 更新主機管理組態

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Quick Setup。
3. 在組態清單中，選擇您建立的主機管理組態。
4. 選擇動作，然後選擇編輯組態。
5. 在目標區段底部附近，從選擇將執行個體設為目標的方式下選擇手動。
6. 在執行個體區段中，選擇您建立的執行個體。
7. 選擇更新。

等待幾分鐘，讓 EC2 重新整理 Session Manager 索引標籤。如果仍然無法連線或收到錯誤訊息，則請檢閱此問題的其餘解決方案。

解決方案 C：關於遺失 SSM Agent 的錯誤

如果您無法使用 Quick Setup 建立主機管理組態，或者收到關於 SSM Agent 未安裝的錯誤訊息，您可能需要在執行個體上手動安裝 SSM Agent。 SSM Agent 是一種 Amazon 軟體，它讓 Systems Manager 可以使用 Session Manager 連線到您執行個體。大多數 Amazon Elastic Compute (AMI) 預設會安裝 SSM Agent。如果您的執行個體是從非標準 AMI 或較舊的 AMI 建立的，您可能需要手動安裝該代理程式。如需安裝 SSM Agent 的程序，請參閱以下與您執行個體作業系統相對應的主題。

- 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

對於 SSM Agent 相關問題，請參閱 SSM Agent 疑難排解。

無權啟動工作階段

問題：您嘗試啟動工作階段，但系統會提示您沒有必須的許可。

- 解決方案：系統管理員尚未授予您 AWS Identity and Access Management (IAM) 啟動Session Manager工作階段的政策許可。如需相關資訊，請參閱控制使用者工作階段存取執行個體。

SSM Agent不在線

問題：您在 Amazon EC2 執行個體Session Manager索引標籤上看到一則訊息，指出：「SSM Agent 不在線上。SSM Agent 無法連線至 Systems Manager 端點以向 服務註冊。」

解決方案：SSM Agent 是在 Amazon EC2 執行個體上執行的 Amazon 軟體，因此 Session Manager 可以與其連線。如果看到此錯誤，表明 SSM Agent 無法與 Systems Manager 端點建立連線。問題的可能來源可能是防火牆限制、路由問題或缺乏網際網路連線。若要解決此問題，請調查網路連線問題。如需詳細資訊，請參閱 SSM Agent 疑難排解 和 疑難排解受管節點的可用性。如需有關 Systems Manager 端點的資訊，請參閱 AWS Systems Manager 一般參考中的端點和配額。 AWS

無權變更工作階段偏好設定

問題：您嘗試替您的組織更新公有的工作階段偏好設定，但系統會提示您沒有必須的許可。

- 解決方案：系統管理者尚未授予您 IAM 政策權限來設定 Session Manager 工作階段。如需相關資訊，請參閱 授與或拒絕使用者許可來更新Session Manager偏好設定。

受管節點無法使用或未設定用於 Session Manager

問題 1：您想要啟動工作階段在 Start a session (開啟工作階段) 主控台頁面，但受管節點不在清單中。

- 解決方案 A：您可能尚未設定要連線的受管節點 AWS Systems Manager。如需詳細資訊，請參閱為組織設定 Systems Manager 統一主控台。

  > **Note**
  >
  > 如果您 AWS Systems Manager SSM Agent連接 IAM 執行個體描述檔時已在受管節點上執行，您可能需要重新啟動代理程式，執行個體才會列在啟動工作階段主控台頁面上。

- 解決方案 B：套用至受管節點上的 SSM Agent 的代理組態可能不正確。如果代理組態不正確，受管節點將無法連線到所需的服務端點，或節點可能會以不同的作業系統向 Systems Manager 報告。如需詳細資訊，請參閱 將 SSM Agent 設定為使用 Linux 節點上的代理 和 將 SSM Agent 設定為使用 Windows Server 執行個體的代理。

問題 2：您想要連接的受管節點位於 Start a session (開啟一個工作階段) 主控台頁面的清單裡，但頁面報告「您所選擇的執行個體沒有設定使用 Session Manager。」

- 解決方案 A：受管節點已設定為與 Systems Manager 服務搭配使用，但連接至節點的 IAM 執行個體描述檔可能不會包含Session Manager工具的許可。如需相關資訊，請參閱使用 Session Manager 許可以驗證或建立 IAM 執行個體設定檔。
- 解決方案 B：受管節點無法在支援 Session Manager 的 SSM Agent 版本上執行。在節點上更新 SSM Agent 至 2.3.68.0 版本或更新版本。

  依照 在 Windows Server EC2 執行個體上手動安裝和解除安裝 SSM Agent、在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent 或 在 macOS EC2 執行個體上手動安裝和解除安裝 SSM Agent 步驟在受管節點上手動更新 SSM Agent，這取決於作業系統。

  或者您可以在一或多個受管節點上使用 Run Command 文件 `AWS-UpdateSSMAgent` 更新代理程式版本。如需相關資訊，請參閱 使用 Run Command 更新 SSM Agent。

  > **Tip**
  >
  > 若要一直持續更新您的代理程式到最新版本，我們建議您定義自動化排程來更新 SSM Agent 到最新版，請使用下列其中一種方法。
  >
  > - 在 State Manager 關聯過程中執行 `AWS-UpdateSSMAgent`。如需相關資訊，請參閱 逐步解說：SSM Agent使用 自動更新 AWS CLI。
  > - 在維護時段內執行 `AWS-UpdateSSMAgent`。如需使用維護時段的資訊，請參閱 使用主控台建立和管理維護時段 及教學課程：使用 建立和設定維護時段 AWS CLI。

- 解決方案 C：受管節點無法連線到必需的服務端點。您可以使用 支援的介面端點來連線至 Systems Manager 端點 AWS PrivateLink ，以改善受管節點的安全狀態。使用介面端點的替代方案是在您的受管節點上啟用對外網際網路存取。如需詳細資訊，請參閱使用 PrivateLink 設定 Session Manager 的 VPC 端點。
- 解決方案 D：受管節點的可用 CPU 或記憶體資源有限。雖然您的受管節點可能是正常的，但如果節點沒有足夠的可用資源，則無法建立工作階段。如需詳細資訊，請參閱對無法連線的執行個體進行故障診斷。

找不到 Session Manager 外掛程式

若要使用 AWS CLI 執行工作階段命令，外掛程式Session Manager也必須安裝在本機機器上。如需相關資訊，請參閱 安裝 的Session Manager外掛程式 AWS CLI。

Session Manager 外掛程式未自動新增到命令列路徑 (Windows)

當您在 Windows 上安裝 Session Manager 外掛程式，`session-manager-plugin` 可執行檔應該會自動新增到您的作業系統的 PATH 環境變數。如果在您執行之後命令失敗，請檢查 Session Manager 外掛程式是否正確地安裝 (`aws ssm start-session --target` *`instance-id`*)，您可能需要使用以下程序手動設定。

修改 PATH 變數 (Windows)

1. 按下 Windows 鍵並輸入 **`environment variables`**。
2. 選擇 Edit environment variables for your account (編輯您帳戶的環境變數)。
3. 選擇 PATH，然後選擇編輯。
4. 新增路徑至 Variable value (變數值) 欄位，以分號分隔，如下列範例所示：*`C:\existing\path`*;***`C:\new\path`***

   `C:\existing\path` 代表已經在欄位中的值。***`C:\new\path`*** 代表您想要新增的路徑，如下列範例所示。

   - 64 位元機器：`C:\Program Files\Amazon\SessionManagerPlugin\bin\`
   - 32 位元機器：`C:\Program Files (x86)\Amazon\SessionManagerPlugin\bin\`
5. 選擇 OK (確定) 兩次以套用新的設定。
6. 關閉所有正在執行的命令提示並重新開啟。

Session Manager 外掛程式無回應

在連接埠轉送工作階段期間，如果您的本機電腦上安裝了防毒軟體，流量可能會停止轉送。在某些情況下，防毒軟體會干擾 Session Manager 外掛程式，從而導致程序死鎖。若要解決此問題，請在防毒軟體中允許或排除 Session Manager 外掛程式。如需有關 Session Manager 外掛程式的預設安裝路徑的相關資訊，請參閱 安裝 的Session Manager外掛程式 AWS CLI。

TargetNotConnected

問題：您嘗試啟動工作階段，但系統傳回錯誤訊息：「呼叫 StartSession 操作時發生錯誤 (TargetNotConnected)：未連接 *InstanceID*。」

- 解決方案：當工作階段的指定目標受管節點未完全設定為與工作階段管理員搭配使用時，會傳回此錯誤。如需相關資訊，請參閱 設定 Session Manager。
- 解決方案 B：如果您嘗試在位於不同 AWS 帳戶 或 的受管節點上啟動工作階段，也會傳回此錯誤 AWS 區域。

啟動工作階段後顯示空白畫面

問題：您啟動了工作階段，但 Session Manager 顯示空白畫面。

- 解決方案 A：當受管節點上的根磁碟區滿載時，就可能發生此問題。因為缺少磁碟空間，所以節點上的 SSM Agent 停止運作。如果要解決此問題，請使用 Amazon CloudWatch 從作業系統收集指標和日誌。如需詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的 Collect metrics, logs, and traces with the CloudWatch agent。
- 解決方案 B：如果您使用包含不相符端點和區域配對的連結來存取主控台，則可能會顯示空白畫面。例如，在下列主控台 URL 中，`us-west-2` 是指定的端點，但 `us-west-1` 是指定的 AWS 區域：

  ```
  https://us-west-2.console.aws.amazon.com/systems-manager/session-manager/sessions?
  region=us-west-1
  ```

- 解決方案 C：受管節點正在使用 VPC 端點連線至 Systems Manager，而您的 Session Manager 偏好設定會將工作階段輸出寫入 Amazon Simple Storage Service (Amazon S3) 儲存貯體或 Amazon CloudWatch Logs 日誌組，但 VPC 中不存在 `s3` 閘道端點或 `logs` 介面端點。如果您的受管節點使用 VPC 端點連線到 Systems Manager，則需要格式為 **`com.amazonaws.`*`region`*`.s3`** 的 `s3` 端點，並且 Session Manager 偏好設定會將工作階段輸出寫入至 Amazon Simple Storage Service (Amazon S3) 儲存貯體。或者，如果您的受管節點使用 VPC 端點連線到 Systems Manager，則需要格式為 **`com.amazonaws.`*`region`*`.logs`** 的 `logs` 端點，並且 Session Manager 偏好設定會將工

作階段輸出寫入至 CloudWatch Logs 日誌群組。如需詳細資訊，請參閱若要建立 Systems Manager 的 VPC 端點。

- 解決方案 D：您在工作階段偏好設定中指定的日誌群組或 Amazon Simple Storage Service (Amazon S3) 儲存貯體已被刪除。若要解決此問題，請使用有效的日誌群組或 S3 儲存貯體來更新工作階段偏好設定。
- 解決方案 E：您在工作階段偏好設定中指定的日誌群組或 Amazon Simple Storage Service (Amazon S3) 儲存貯體未加密，但您已將 `cloudWatchEncryptionEnabled` 或 `s3EncryptionEnabled` 輸入設定為 `true`。若要解決此問題，請使用已加密的日誌群組或 Amazon Simple Storage Service (Amazon S3) 儲存貯體更新工作階段偏好設定，或將 `cloudWatchEncryptionEnabled` 或 `s3EncryptionEnabled` 輸入設定為 `false`。此案例僅適用於使用命令列工具建立工作階段偏好設定的客戶。

受管節點在長時間執行工作階段期間變得沒有回應

問題：您的受管節點沒有回應或在長時間執行工作階段期間當機。

解決方案：減少 Session Manager 的 SSM Agent 日誌保留期限。

若要減少工作階段的 SSM Agent 日誌保留期限

1. 在 Linux 的 `/etc/amazon/ssm/` 目錄或 Windows 的 `C:\Program Files\Amazon\SSM` 目錄中找到 `amazon-ssm-agent.json.template`。
2. 將 `amazon-ssm-agent.json.template` 的內容複製到同一目錄中的新檔案，名為 `amazon-ssm-agent.json`。
3. 降低 SSM 屬性中 `SessionLogsRetentionDurationHours` 值的預設值，並儲存檔案。
4. 重新啟動SSM Agent。

呼叫 StartSession 操作時發生錯誤 (InvalidDocument)

問題：使用 AWS CLI開始連線階段時，您收到下列錯誤。

```
An error occurred (InvalidDocument) when calling the StartSession operation: Document
 type: 'Command' is not supported. Only type: 'Session' is supported for Session
 Manager.
```

解決方案：您為 `--document-name` 參數指定的 SSM 文件不是連線階段文件。使用下列程序在 AWS Management Console中檢視連線階段文件清單。

若要檢視連線階段文件清單

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Documents (文件)。
3. 在類別清單中，選擇連線階段文件。

AWS Systems Manager State Manager

State Manager是 中的工具 AWS Systems Manager，是一種安全且可擴展的組態管理服務，可自動化將受管節點和其他 AWS 資源保持在您定義狀態的程序。若要開始使用 State Manager，請開啟 Systems Manager 主控台。在導覽窗格中，選擇 State Manager。

> **Note**
>
> State Manager 和 Maintenance Windows 可以在受管節點上執行某些類似的更新。您選擇哪一項，取決於您是否需要在指定的期間內自動化系統合規，或執行高優先順序、時間敏感的任務。
> 如需詳細資訊，請參閱在 State Manager 與 Maintenance Windows 之間進行選擇。

State Manager 如何為我的組織帶來益處？

使用預先設定的 Systems Manager 文件 (SSM 文件)，State Manager 提供下列節點管理優勢：

- 在啟動時透過特定軟體啟動節點。
- 根據定義的排程下載及更新代理程式，包括 SSM Agent。
- 設定網路設定。
- 將節點加入 Microsoft Active Directory 網域。
- 在整個生命週期內，於 Linux、macOS 和 Windows 受管節點上執行指令碼。

若要管理跨其他 AWS 資源的組態偏離，您可以使用 Systems Manager 中的 Automation 工具搭配 State Manager來執行下列類型的任務：

- 將 Systems Manager 角色連接到 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體以作為受管節點。

- 為安全群組強制執行所需的輸入和輸出規則。
- 建立或刪除 Amazon DynamoDB 備份。
- 建立 Amazon Elastic Block Store (Amazon EBS) 快照。
- 關閉 Amazon Simple Storage Service (Amazon S3) 儲存貯體上的讀取和寫入許可。
- 啟動、重新啟動或停止受管節點和 Amazon Relational Database Service (Amazon RDS) 執行個體。
- 將修補程式套用至 Linux、macOS、和 WiWindows AMIs。

如需使用 State Manager 搭配 Automation Runbook 的資訊，請參閱 使用 State Manager 關聯排程自動化。

誰應該使用State Manager？

State Manager 適合任何想要改善 AWS 資源管理和管控並減少組態偏離 AWS 的客戶。

State Manager 有哪些功能？

State Manager 的重要功能如下所示：

- State Manager 關聯

  State Manager 關聯是您指派給 AWS 資源的組態。該組態會定義您想在資源上維持的狀態。例如，關聯可以指定必須在受管節點上安裝和執行防毒軟體，或者必須關閉特定連接埠。

  關聯會指定套用組態並以關聯為目標的排程。例如，防毒軟體的關聯可能在 AWS 帳戶的所有受管節點上一天執行一次。如果節點上未安裝軟體，則關聯會指示 State Manager 來安裝。如果已安裝軟體，但並未執行服務，則關聯可能會指示 State Manager 來啟動服務。
- 彈性排程選項

  State Manager 提供以下選項，用於在關聯執行時排程：

  - 立即處理或延遲處理

    建立關聯時，系統預設會立即在指定的資源上執行該關聯。在初次執行後，關聯會根據您定義的排程依間隔執行。

    您可以在主控台使用 Apply association only at the next specified Cron interval (僅在下一個指定 Cron 間隔套用關聯) 選項，或透過命令列中的 `ApplyOnlyAtCronInterval` 參數，指示 State Manager 不立即執行關聯。

- Cron 與 Rate 運算式

  當您建立關聯時，需要指定 State Manager 套用組態時的排程。State Manager 支援關聯執行時針對排程的大多數標準 Cron 和 Rate 運算式。State Manager 還支援 Cron 運算式，其中包含一週中的某一天和數字符號 (#)，以指定一個月的第 n 天執行關聯，(L) 符號指示一個月的最後 X 天。

  > **Note**
  >
  > State Manager 目前不支援在 Cron 運算式中為關聯指定月份。

  若要進一步控制關聯的執行時間，例如，如果您希望在週二修補程式日後的兩天執行關聯，則可以指定偏移量。同時偏移定義在排程的日期之後等待多少天才能執行關聯。

  如需有關建立 Cron 和 Rate 運算式的詳細資訊，請參閱參考：Systems Manager 的 Cron 和 Rate 運算式。

- 多個目標鎖定選項

  關聯也會指定關聯的目標。 State Manager支援透過使用目前 AWS 區域 和 中的標籤 AWS Resource Groups、個別節點 IDs 或所有受管節點來鎖定 AWS 資源 AWS 帳戶。

- 支援 Amazon S3

  將關聯執行的命令輸出存放在您選擇的 Amazon S3 儲存貯體中。如需詳細資訊，請參閱在 Systems Manager 中使用關聯。

- 支援 EventBridge

  在 Amazon EventBridge 規則中，此 Systems Manager 工具支援為事件類型和目標類型。如需詳細資訊，請參閱 使用 Amazon EventBridge 監控 Systems Manager 及 參考：Systems Manager 的 Amazon EventBridge 事件模式和類型。

使用 State Manager 需要付費嗎？

State Manager 是免費提供的。

主題

- 了解 State Manager 的運作方式
- 在 Systems Manager 中使用關聯

- 建立執行 MOF 檔案的關聯
- 建立執行 Ansible 手冊的關聯
- 建立執行 Chef 配方的關聯
- 逐步解說：SSM Agent使用 自動更新 AWS CLI
- 演練：在 Windows Server 的 EC2 執行個體自動更新 PV 驅動程式

詳細資訊

- 使用 Amazon EC2 Systems Manager 和 Windows PowerShell DSC 來防止組態不一致的情形
- 使用 State Manager 在 Auto Scaling 群組中設定 Amazon EC2 執行個體

了解 State Manager 的運作方式

State Manager是 中的工具 AWS Systems Manager，是一種安全且可擴展的服務，可將受管節點保存在混合多雲端基礎設施中的程序，自動化為您定義的狀態。

這是 State Manager 運作方式：

1. 決定您要套用至 AWS 資源的狀態。

   您是否希望保證使用特定應用程式 (例如防毒或反惡意軟體) 來設定受管節點？ 您是否希望能夠將更新 SSM Agent 或其他 AWS 套件 (例如 `AWSPVDriver`) 的程序自動化？ 您是否需要保證關閉或開啟特定的連接埠？ 若要開始使用 State Manager，請判斷您要套用至 AWS 資源的狀態。您想要套用的狀態會決定您使用哪個 SSM 文件來建立 State Manager 關聯。

   State Manager 關聯是您指派給 AWS 資源的組態。該組態會定義您想在資源上維持的狀態。例如，關聯可以指定必須在受管節點上安裝和執行防毒軟體，或者必須關閉特定連接埠。

   關聯會指定套用組態並以關聯為目標的排程。例如，防毒軟體的關聯可能在 AWS 帳戶的所有受管節點上一天執行一次。如果節點上未安裝軟體，則關聯會指示 State Manager 來安裝。如果已安裝軟體，但並未執行服務，則關聯可能會指示 State Manager 來啟動服務。
2. 判斷預先設定的 SSM 文件是否可協助您在 AWS 資源上建立所需的狀態。

   Systems Manager 包含數十個預先設定的 SSM 文件，可讓您用來建立關聯。預先設定的文件已準備好可執行常見任務，例如安裝應用程式、設定 Amazon CloudWatch、執行 AWS Systems Manager 自動化、執行 PowerShell 和 Shell 指令碼，以及將受管節點加入 Active Directory 的目錄服務網域。

在 Systems Manager 主控台中可檢視所有 SSM 文件。選擇文件名稱來進一步了解各項文件。以下是兩個範例：AWS-ConfigureAWSPackage 和 AWS-InstallApplication。

3. 建立關聯。

您可以使用 Systems Manager 主控台、 AWS Command Line Interface (AWS CLI)、 AWS Tools for Windows PowerShell (Tools for Windows PowerShell) 或 Systems Manager API 來建立關聯。在建立關聯時，指定下列資訊：

- 關聯名稱。
- SSM 文件的參數 (例如，應用程式安裝路徑或在節點上執行的指令碼)。
- 關聯的目標。您可以透過指定標籤，選擇個別節點 ID，或是在 AWS Resource Groups中選擇群組等方式，將受管執行個體設為目標。您也可以將目前 AWS 區域 和 中的所有受管節點設為目標 AWS 帳戶。如果您的目標包含超過 1，000 個節點，系統會使用每小時限流機制。這表示您可能會在狀態彙總計數中看到不正確，因為彙總程序每小時執行一次，而且只有在節點的執行狀態變更時才會執行。
- 狀態套用時間或頻率的排程。您可以指定 Cron 或 Rate 運算式。如需使用 Cron 和 Rate 運算式建立排程的詳細資訊，請參閱關聯的 Cron 與 Rate 運算式。

> **Note**
>
> State Manager 目前不支援在 Cron 運算式中為關聯指定月份。

當您執行命令來建立關聯時，Systems Manager 會將您指定的資訊 (排程、目標、SSM 文件和參數) 繫結至目標資源。關聯狀態一開始會顯示為「Pending」(待定)，同時系統正在嘗試連接所有目標並立即套用關聯中指定的狀態。

> **Note**
>
> 如果您新建的關聯已排定要執行，但同時仍在執行先前的關聯，則先前的關聯會逾時，並執行新的關聯。

Systems Manager 會報告在資源上建立關聯的請求狀態。您可以透過主控台或 (對於受管節點) 使用 DescribeInstanceAssociationsStatus API 操作來檢視狀態詳細資訊。如果選擇在建立關聯時將命令的輸出寫入 Amazon Simple Storage Service (Amazon S3)，您也可以在指定的 Amazon S3 儲存貯體中檢視輸出。

如需詳細資訊，請參閱在 Systems Manager 中使用關聯。

> **Note**
>
> 在關聯執行期間由 SSM 文件啟動的 API 操作不會記錄在 AWS CloudTrail中。

4. 監控與更新。

   在您建立關聯後，State Manager 會根據您在關聯中定義的排程重新套用組態。在主控台的 State Manager 頁面中，或在建立關聯時直接呼叫 Systems Manager 產生的關聯 ID，可檢視關聯狀態。如需詳細資訊，請參閱檢視關聯歷史記錄。您可以視需要更新關聯文件並重新套用。您也可以建立多個版本的關聯。如需詳細資訊，請參閱編輯和建立關聯的新版本。

了解何時將關聯套用至資源

建立關聯時，您需要指定 SSM 文件，此文件會定義組態、目標資源清單，以及套用組態的排程。根據預設，State Manager 會在您建立關聯時執行關聯，然後依照排程執行。State Manager 也會嘗試在以下情況下執行關聯：

- 關聯編輯 – State Manager 在使用者編輯後執行關聯，並儲存對以下任何關聯欄位的變更：`DOCUMENT_VERSION`、`PARAMETERS`、`SCHEDULE_EXPRESSION`、`OUTPUT_S3_LOCATION`。
- 文件編輯 – State Manager 在使用者編輯後執行關聯，並儲存對定義關聯組態狀態的 SSM 文件的變更。具體而言，在對文件進行以下編輯後，關聯便會執行：
  - 使用者指定新的 `$DEFAULT` 文件版本，而關聯是使用 `$DEFAULT` 版本所建立。
  - 使用者更新文件，而關聯是使用 `$LATEST` 版本所建立。
  - 使用者會刪除建立關聯時所指定的文件。
- 手動啟動 – 使用者從 Systems Manager 主控台或以程式設計方式啟動時，State Manager 會執行關聯。
- 目標變更：State Manager會在目標節點上發生下列任何活動之後執行關聯：
  - 受管節點首次上線。
  - 缺少排程的關聯執行後，受管節點便會上線。
  - 受管節點會在停止超過 30 天後上線。

> **Note**
>
> State Manager 不會監控關聯中使用的文件或套件 AWS 帳戶。如果您在一個帳戶中更新文件或套件，則更新不會導致關聯在第二個帳戶中執行。您必須在第二個帳戶中手動執行關聯。

防止在目標變更時執行關聯

在某些情況下，您可能不希望在由受管節點組成的目標變更時執行關聯，但只會根據其指定的排程。

> **Note**
>
> 執行 Automation Runbook 會產生成本。如果與 Automation Runbook 的關聯以您帳戶中的所有執行個體為目標，而且您定期啟動大量執行個體，則會在啟動時在每個執行個體上執行 Runbook。這可能會導致自動化費用提高。

若要防止關聯在關聯目標變更時執行，請選取只在下一個指定的 Cron 間隔套用關聯核取方塊。此核取方塊位於建立關聯和編輯關聯頁面的指定排程區域中。

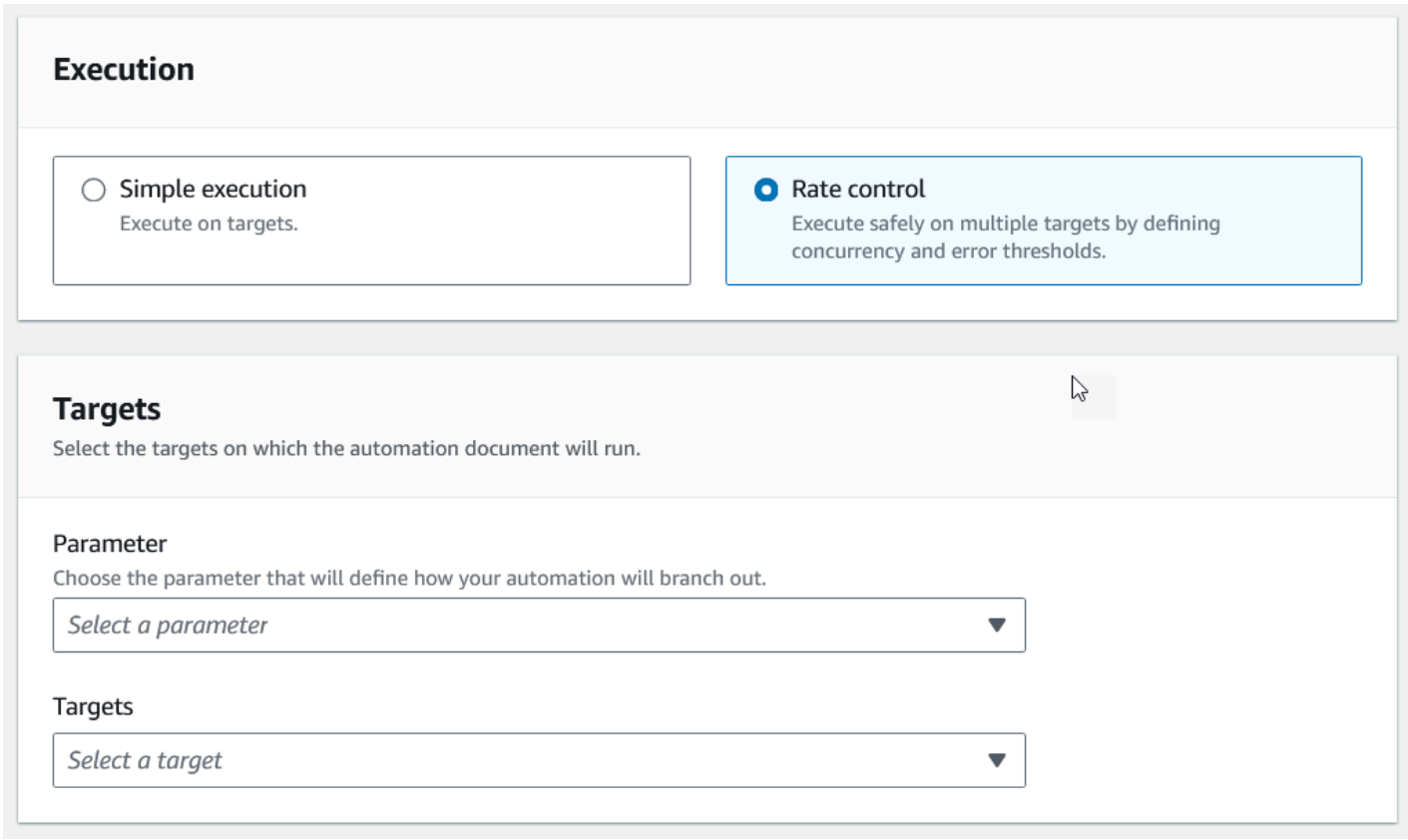
此選項適用於包含 Automation Runbook 或 SSM 文件的關聯。

關於透過 Automation 執行手冊的目標更新

為了在偵測到新目標節點時套用透過 Automation 執行手冊建立的關聯，必須滿足以下條件：

- 關聯必須由Quick Setup組態建立。 Quick Setup 是 中的工具 AWS Systems Manager。目前不支援其他程序建立的關聯。
- Automation 執行手冊必須明確以資源類型 `AWS::EC2::Instance` 或 `AWS::SSM::ManagedInstance` 為目標。
- 此關聯必須同時指定參數和目標。

  在主控台中，當您選擇速率控制執行時，便會顯示參數和目標欄位。



當您使用 CreateAssociation、 CreateAssociationBatch或 UpdateAssociation API 動作時，您可以使用 `AutomationTargetParameterName`和 `Targets`輸入來指定這些值。在每個 API 動作中，您也可以將 `ApplyOnlyAtCronInterval` 參數設定為 `true`，防止每次目標變更時執行關聯。

如需使用主控台控制關聯執行時間的相關資訊，包括避免自動化執行意外高成本的詳細資訊，請參閱了解何時將關聯套用至資源。

在 Systems Manager 中使用關聯

本節說明如何使用 AWS Systems Manager 主控台、 AWS Command Line Interface (AWS CLI) 和 建立和管理State Manager關聯 AWS Tools for PowerShell。

主題

- 了解 State Manager 關聯中的目標和速率控制
- 建立關聯
- 編輯和建立關聯的新版本
- 刪除關聯

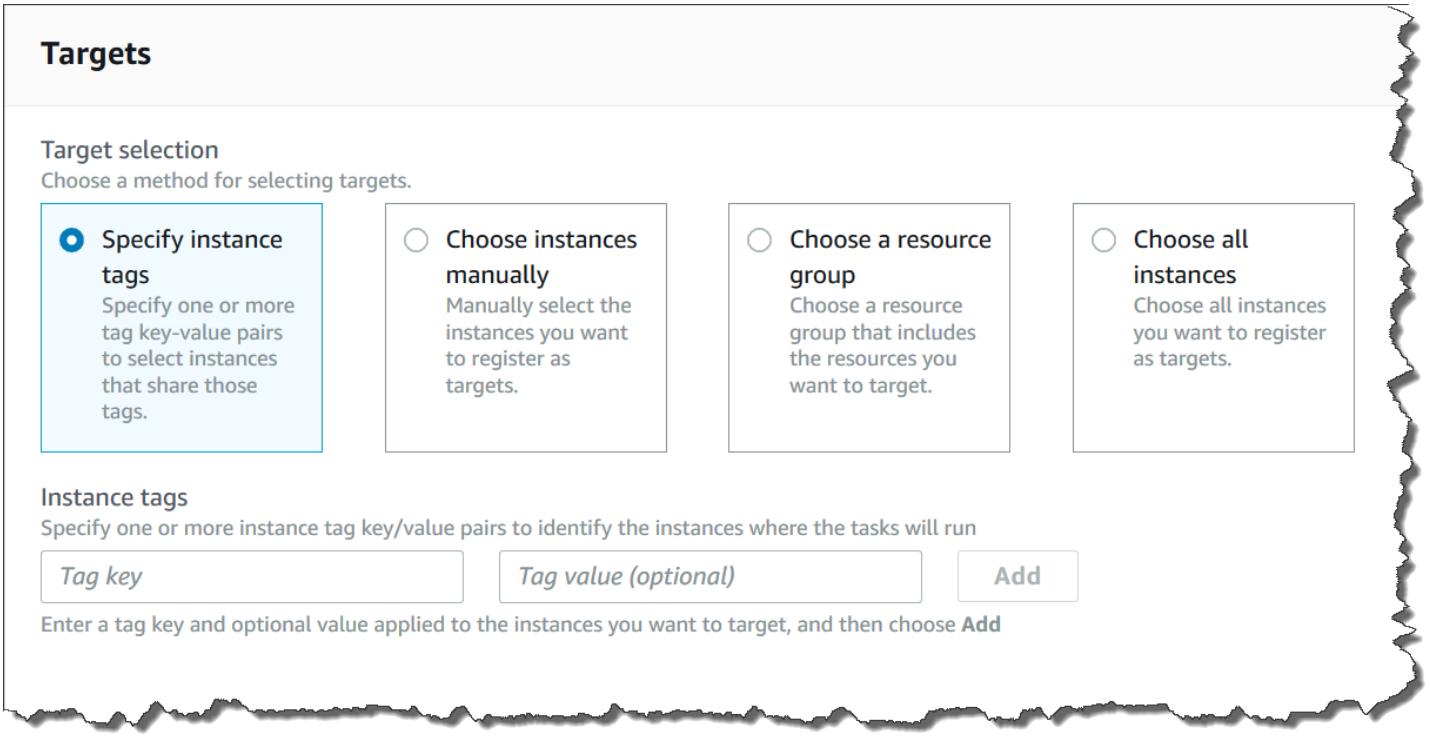
- 執行具有關聯的 Auto Scaling 群組
- 檢視關聯歷史記錄
- 透過 IAM 使用關聯

了解 State Manager 關聯中的目標和速率控制

本主題說明State Manager的功能可協助您將關聯部署到數十或數百個節點，同時控制在排程時間執行關聯的節點數目。 State Manager 是 中的工具 AWS Systems Manager。

使用目標

建立 State Manager 關聯時，您可以在 Systems Manager 主控台的 Targets (目標) 區段中選擇要使用關聯設定哪些節點，如此處所示。



如果您使用命令列工具 (例如 AWS Command Line Interface (AWS CLI)) 建立關聯，請指定 `targets` 參數。將節點設為目標允許您利用關聯來設定數十個、數百個或數千個執行個體，而不用指定或選擇個別節點 ID。

每個受管節點可以作為最多 20 個關聯的目標。

State Manager 在建立關聯時包含下列目標選項。

指定標籤

使用此選項可指定指派給節點的標籤金鑰和 (選用) 標籤值。當您執行請求時，系統會在所有符合指定標籤索引鍵和數值的節點上找出並嘗試建立關聯。如果您指定了多個標籤值，則關聯會將擁有至少其中一個標籤值的任何節點設為目標。系統會在最初建立關聯時執行關聯。在此次初始執行之後，系統會根據您指定的排程執行關聯。

如果您建立新的節點，並將指定的標籤索引鍵和數值指派給該些節點，系統會自動套用關聯，立即執行關聯，然後根據排程執行關聯。這適用於關聯使用命令或政策文件時，如果關聯使用 Automation Runbook，則不適用。如果從節點中刪除指定的標籤，系統將不再在這些節點上執行關聯。

> **Note**
>
> 如果您使用 Automation Runbook 搭配 State Manager，而且標記限制阻止您達成特定目標，請考慮使用 Automation Runbook 搭配 Amazon EventBridge。如需詳細資訊，請參閱根據 EventBridge 事件執行自動化。如需使用 Runbook 搭配 State Manager 的資訊，請參閱使用 State Manager 關聯排程自動化。

最佳實務是在建立使用命令或政策文件的關聯時使用標籤。我們也建議您在建立關聯以執行 Auto Scaling 群組時使用標籤。如需詳細資訊，請參閱執行具有關聯的 Auto Scaling 群組。

> **Note**
>
> 記下以下資訊。
>
> - 使用標籤在 中建立以節點為目標 AWS Management Console 的關聯時，您只能為自動化關聯指定一個標籤索引鍵，並為命令關聯指定五個標籤索引鍵。關聯中指定的所有標籤索引鍵目前必須指派給節點。否則，State Manager 無法將相應節點作為關聯的目標。
> - 如果您想要使用 主控台，而且想要使用多個標籤索引鍵做為自動化關聯的目標節點，以及五個標籤索引鍵做為命令關聯的目標，請將標籤索引鍵指派給 AWS Resource Groups 群組，並將節點新增至其中。然後，您可以在建立 State Manager 關聯時，在目標清單中選擇資源群組選項。
> - 您可以使用 AWS CLI指定最多五個標籤鍵。如果您使用 AWS CLI，`create-association`命令中指定的所有標籤索引鍵目前都必須指派給節點。否則，State Manager 無法將相應節點作為關聯的目標。

手動選擇節點

使用此選項可手動選取您要在其中建立關聯的節點。執行個體窗格會顯示目前 AWS 帳戶 和 中的所有 Systems Manager 受管節點 AWS 區域。您可以根據需要手動選取任意數量的節點。系統會在最初建立關聯時執行關聯。在此次初始執行之後，系統會根據您指定的排程執行關聯。

> **Note**
>
> 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

選擇資源群組

使用此選項可在 AWS Resource Groups 標籤型或 AWS CloudFormation 堆疊型查詢傳回的所有節點上建立關聯。

下面是有關將資源群組設為關聯的目標的詳細資訊。

- 如果您將新節點新增至群組，系統會自動將節點對應至以資源群組為目標的關聯。系統會在發現變更時將關聯套用至節點。在此次初始執行之後，系統會根據您指定的排程執行關聯。
- 如果您建立以資源群組為目標的關聯，且已指定相應群組的資源類型為 `AWS::SSM::ManagedInstance`，則依設計，該關聯會在混合多雲端環境中的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體和非 EC2 節點上執行。

  對數也是 true。如果您建立以資源群組為目標的關聯，且該群組已指定`AWS::EC2::Instance`資源類型，則依設計，該關聯會在混合多雲端環境和 (Amazon EC2non-EC22 節點上執行。
- 如果您建立一個以資源群組為目標的關聯，則指派至該資源群組的標籤索引鍵不得超過五個，或指定至任一標籤索引鍵的值不得超過五個。如果這些條件中的任一條件適用於指派至給您資源群組的標籤和索引鍵，則關聯將無法執行，並傳回 `InvalidTarget` 錯誤。
- 如果您使用標籤建立以資源群組為目標的關聯，則不能選擇標籤值的 (空白值) 選項。
- 如果刪除資源群組，該群組中的所有執行個體都不會再執行關聯。做為最佳實務，應刪除以群組為目標的關聯。
- 您最多可將單一資源群組設為關聯的目標。不支援多個或巢狀群組。
- 建立關聯之後，State Manager 會定期以資源群組中資源的相關資訊來更新關聯。如果您在資源群組加入新資源，則系統何時將關聯套用至新資源的排程將取決於數個因素。您可以在 Systems Manager 主控台的 State Manager 頁面中判斷關聯的狀態。

> **Warning**
>
> 具有建立以 Amazon EC2 執行個體資源群組為目標之關聯許可的 AWS Identity and Access Management (IAM) 使用者、群組或角色，會自動擁有群組中所有執行個體的根層級控制。只有受信任的管理員才能建立關聯。

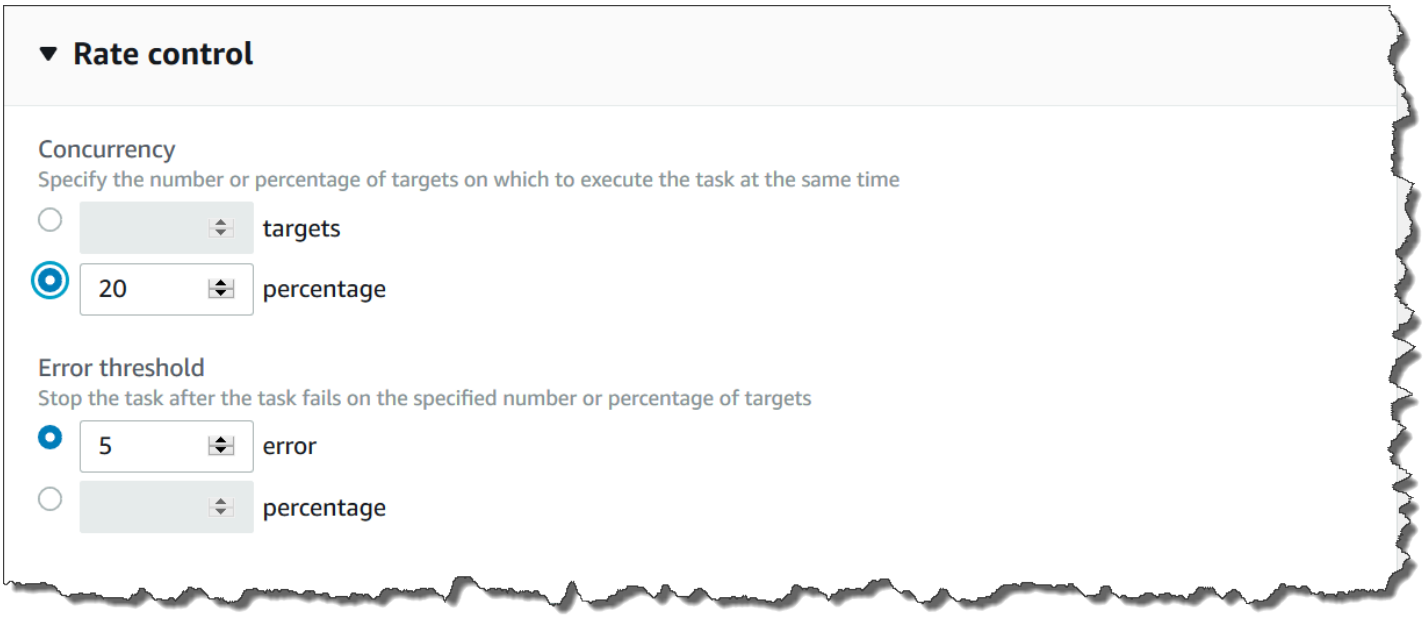
如需 Resource Groups 的詳細資訊，請參閱 AWS Resource Groups 使用者指南中的什麼是 AWS Resource Groups？。

選擇所有節點

使用此選項以目前 AWS 帳戶 和 中的所有節點為目標 AWS 區域。當您執行請求時，系統會尋找並嘗試在目前 AWS 帳戶 和 中的所有節點上建立關聯 AWS 區域。系統會在最初建立關聯時執行關聯。在此次初始執行之後，系統會根據您指定的排程執行關聯。如果您建立新節點，系統會自動套用關聯，立即執行關聯，然後根據排程執行關聯。

使用速率控制

您可以透過指定並行值和錯誤閾值來控制節點關聯的執行。並行值會指定多少節點可同時執行關聯。錯誤閥值會指定在 Systems Manager 傳送命令至每個已設定該關聯的節點停止執行關聯前，允許多少次關聯執行失敗。此命令會讓關聯在下一次排定的執行前都停止執行。並行和錯誤閾值功能統稱為「速率控制」。



並行數量

並行可讓您指定一次只讓特定數量的節點處理關聯，有助於減少對節點造成的影響。您可以指定絕對數量的節點 (例如 20) 或目標節點集的百分比 (例如 10%)。

State Manager並行具有以下約束與限制：

- 如果您選擇使用目標來建立關聯，但不指定並行值，則State Manager會自動強制執行 50 個節點的並行上限。
- 正在執行使用並行的關聯時，如有符合目標條件的新節點上線，則若未超過並行值，新的節點會執行關聯。若超過並行值，則節點會在目前的關聯執行間隔期間遭略過。這些節點會在下一次排定的間隔期間執行關聯，同時遵循並行要求。
- 如果您更新使用並行的關聯，且在更新時有一個或多個節點正在處理該關聯，則允許完成正在執行關聯的所有節點。尚未開始的關聯則會停止。執行中的關聯完成後，所有目標節點會立即再次執行關聯，因為它已更新。當關聯再次執行時，會強制執行並行值。

錯誤閾值

錯誤閥值會指定在 Systems Manager 傳送命令至每個已設定該關聯的節點前，允許多少次關聯執行失敗。此命令會讓關聯在下一次排定的執行前都停止執行。您可以指定絕對數量的錯誤 (例如 10 個) 或目標集的百分比 (例如 10%)。

例如，假設您指定三個錯誤的絕對數量，State Manager會在傳回第四個錯誤時傳送停止命令。如果您指定 0，則State Manager會在第一個錯誤結果傳回後傳送停止命令。

如果您針對 50 個關聯指定錯誤閾值為 10%，則State Manager會在傳回第六個錯誤時傳送停止命令。達到錯誤閾值時已經在執行的關聯允許完成，但其中某些關聯也可能會失敗。為了確保錯誤不會超過針對錯誤閾值指定的數量，請將 Concurrency (並行) 值設為 1，讓關聯一次處理一個。

State Manager錯誤閾值具有以下約束與限制：

- 錯誤閾值針對目前的間隔強制執行。
- 每個錯誤的相關資訊 (包括步驟層級詳細資訊) 會記錄在關聯歷程記錄中。
- 如果您選擇使用目標來建立關聯，但不指定錯誤閾值，則State Manager會自動強制執行 100% 的失敗閾值。

建立關聯

State Manager是 中的工具 AWS Systems Manager，可協助您將 AWS 資源保持在您定義和減少組態偏離的狀態。若要執行此操作，State Manager 會使用關聯。關聯是您指派給 AWS 資源的組態。該組

態會定義您想在資源上維持的狀態。例如，關聯可以指定必須在受管節點上安裝和執行防毒軟體，或者必須關閉特定連接埠。

關聯會指定套用組態並以關聯為目標的排程。例如，防毒軟體的關聯可能在 AWS 帳戶的所有受管節點上一天執行一次。如果節點上未安裝軟體，則關聯會指示 State Manager 來安裝。如果已安裝軟體，但並未執行服務，則關聯可能會指示 State Manager 來啟動服務。

> **Warning**
>
> 建立關聯時，您可以選擇受管節點 AWS 的資源群組做為關聯的目標。如果 AWS Identity and Access Management (IAM) 使用者、群組或角色具有建立以受管節點資源群組為目標之關聯的許可，則該使用者、群組或角色會自動擁有群組中所有節點的根層級控制。只有受信任的管理員才能建立關聯。

關聯目標和速率控制

關聯會指定哪些受管節點 (或目標) 應接收關聯。State Manager 包含多項功能，可協助您將受管節點設為目標，並控制如何將關聯部署至這些目標。如需目標和速率控制的詳細資訊，請參閱了解 State Manager 關聯中的目標和速率控制。

標記關聯

您可以使用 AWS CLI 或 等命令列工具，在建立關聯時將標籤指派給關聯 AWS Tools for PowerShell。不支援使用 Systems Manager 主控台將標籤新增至關聯。

執行關聯

依預設，在您建立關聯後，State Manager 會立即執行關聯，之後再根據您定義的排程執行。

系統也會根據下列規則執行關聯：

- State Manager 嘗試於間隔期間在所有已指定或設為目標的節點上執行關聯。
- 如果未在間隔期間執行關聯 (例如，因為並行值限制了一次所能處理關聯的節點數目)，則 State Manager 會嘗試在下一個間隔期間執行關聯。
- 關聯組態、目標節點、文件或參數發生變更後，State Manager 會執行關聯。如需詳細資訊，請參閱了解何時將關聯套用至資源
- State Manager會記錄所有略過的間隔的歷程記錄。您可以在 Execution History (執行歷程記錄) 標記檢視歷程記錄。

排程關聯

您可以排程關聯，以基本間隔 (例如每 10 小時) 執行，也可以使用自訂 Cron 和 Rate 運算式建立更進階的排程。您也可以在第一次建立關聯時阻止執行關聯。

使用 Cron 和 Rate 運算式來排程關聯執行

State Manager 不僅支援標準的 Cron 和 Rate 運算式，還支援這類 Cron 運算式：包含一週中的某一天和數字符號 (#)，來指定一個月的第 n 天執行關聯。以下是在每月第三個週二 23:30 UTC 執行 cron 排程的範例：

```
cron(30 23 ? * TUE#3 *)
```

以下是在每月第二個週四午夜 UTC 執行的範例：

```
cron(0 0 ? * THU#2 *)
```

State Manager 還支援 (L) 符號來指示一個月的最後 X 天。以下是在每月最後一個週二午夜 UTC 執行 cron 排程的範例：

```
cron(0 0 ? * 3L *)
```

若要進一步控制關聯的執行時間，例如，如果您希望在週二修補程式日後的兩天執行關聯，則可以指定偏移量。同時偏移定義在排程的日期之後等待多少天才能執行關聯。例如，如果您指定了 `cron(0 0 ? * THU#2 *)` 的 cron 排程，則可以在排程偏移欄位指定數字 3，以在該月第二個週四之後的每個週日執行關聯。

> **Note**
>
> 若要使用偏移，必須在主控台選取僅在下一個指定的 Cron 間隔時間套用關聯，或者在命令列中指定 `ApplyOnlyAtCronInterval` 參數。啟用其中任一選項後，State Manager 不會在建立關聯後立即執行。

如需 Cron 和 Rate 運算式的詳細資訊，請參閱參考：Systems Manager 的 Cron 和 Rate 運算式。

建立關聯 (主控台)

下列程序說明如何使用 Systems Manager 主控台來建立 State Manager 關聯。

> **Note**
>
> 記下以下資訊。

- 此程序說明如何建立使用 Command 或 Policy 文件將受管節點設為目標的關聯。如需建立使用 Automation Runbook 將節點或其他類型的 AWS 資源設為目標的關聯，請參閱 使用 State Manager 關聯排程自動化。
- 建立關聯時，您可以使用 指定最多五個標籤索引鍵 AWS Management Console。為關聯指定的所有標籤索引鍵目前必須指派給節點。如果不是，則 State Manager 無法將節點設為關聯的目標。

建立 State Manager 關聯

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 State Manager。
3. 選擇 Create association (建立關聯)。
4. 在 Name (名稱) 欄位中指定名稱。
5. 在 Document (文件) 清單中，選擇文件名稱旁的選項。請注意文件類型。此程序適用於 Command 和 Policy 文件。如需建立使用 Automation Runbook 的關聯的相關資訊，請參閱 使用 State Manager 關聯排程自動化。

   Important

   如果該文件是從另一個帳戶共用的，則 State Manager 不支援執行使用新版文件的關聯。如果是從另一個帳戶共用的，則 State Manager 一律執行文件的 default 版本，即使 Systems Manager 主控台顯示新版本已處理。如果您想要使用從另一個帳戶共用而來的新版本文件來執行關聯，則必須將文件版本設定為 default。

6. 對於 Parameters (參數)，指定所需的輸入參數。
7. (選用) 選擇要套用至您的監控關聯的 CloudWatch 警示。

   Note

   記下有關此步驟的以下資訊。

   - 警示清單最多顯示 100 個警示。如果您在清單中沒有看到您的警示，請使用 AWS Command Line Interface 建立關聯。如需詳細資訊，請參閱建立關聯 (命令列)。

- 若要將 CloudWatch 警示附加至您的命令，建立關聯的 IAM 主體必須具備 `iam:createServiceLinkedRole` 動作的許可。如需有關 CloudWatch 警示的詳細資訊，請參閱使用 Amazon CloudWatch 警示。
- 如果您的警示啟用，則不會執行任何待處理命令叫用或自動化。

8. 對於 Targets (目標)，請選擇選項。如需使用目標的詳細資訊，請參閱了解 State Manager 關聯中的目標和速率控制。

> **Note**
>
> 為了在偵測到新目標節點時套用透過 Automation 執行手冊建立的關聯，必須滿足特定條件。如需相關資訊，請參閱關於透過 Automation 執行手冊的目標更新。

9. 在 Specify schedule (指定排程) 區段中，選擇 On Schedule (按照排程) 或 No schedule (無排程)。如果您選擇 On Schedule (按照排程)，請使用提供的按鈕來為關聯建立 Cron 或 Rate 排程。

   如果您不希望在建立關聯之後立即執行關聯，請選擇 Apply association only at the next specified Cron interval (僅在下一個指定的 Cron 間隔套用關聯)。

10. (選用) 在 Schedule offset (排程偏移) 欄位中，指定介於 1 和 6 之間的數字。

11. 在 Advanced options (進階選項) 區段中，使用 Compliance severity (合規嚴重) 選擇關聯的嚴重性等級，並使用 Change Calendars (變更行事曆) 選擇關聯的變更行事曆。

    合規報告會指出關聯狀態合規與否，以及您在這裡指示的嚴重性等級。如需詳細資訊，請參閱關於 State Manager關聯合規。

    變更行事曆會決定何時執行關聯。如果行事曆已關閉，則不會套用關聯。如果行事曆處於開啟狀態，則會相應地執行關聯。如需詳細資訊，請參閱AWS Systems Manager Change Calendar。

12. 在 Rate control (速率控制) 區段中，選擇選項來控制關聯在多個節點上的執行方式。如需使用速率控制的詳細資訊，請參閱了解 State Manager 關聯中的目標和速率控制。

    在 Concurrency (並行) 部分，選擇一個選項：

    - 選擇 targets (目標)，輸入可以同時執行關聯的目標絕對數量。
    - 選擇 percentage (百分比)，輸入可以同時執行關聯的目標集百分比。

    在 Error threshold (錯誤閾值) 部分，選擇一個選項：

- 選擇 errors (錯誤)，輸入 State Manager 停止在額外目標執行關聯之前允許的錯誤絕對數量。
- 選擇 percentage (百分比)，輸入 State Manager 停止在額外目標執行關聯之前允許的錯誤百分比。

13. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取啟用將輸出寫入 S3 方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

> **Note**
>
> 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給受管節點之執行個體設定檔的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確認與受管節點相關聯的執行個體描述檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

以下是開啟關聯的 Amazon Simple Storage Service (Amazon S3) 輸出所需的最低許可。您可以透過將 IAM 政策連接到帳戶內使用者或角色，以進一步限制存取。Amazon EC2 執行個體設定檔至少應擁有具備 `AmazonSSMManagedInstanceCore` 受管政策和下列內嵌政策的 IAM 角色。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:PutObjectAcl"
            ],
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
        }
    ]
}
```

如需最低許可，接收匯出的 Amazon Simple Storage Service (Amazon S3) 儲存貯體必須具備 Amazon Simple Storage Service (Amazon S3) 主控台定義的預設設定。如需有關建立 Amazon Simple Storage Service (Amazon S3) 儲存貯體的詳細資訊，請參閱 Amazon Simple Storage Service (Amazon S3) 使用者指南中的建立儲存貯體。

> **Note**
>
> 在關聯執行期間由 SSM 文件啟動的 API 操作不會記錄在 AWS CloudTrail中。

14. 選擇 Create Association (建立關聯)。

> **Note**
>
> 如果刪除您建立的關聯，關聯將不再在該關聯的任何目標上執行。

建立關聯 (命令列)

下列程序說明如何使用 AWS CLI （在 Linux 或 Windows 上） 或 Tools for PowerShell 來建立State Manager關聯。本節包含數個示範如何使用目標和速率控制的範例。目標和速率控制允許您將關聯指派給數十個或數百個節點，同時控制這些關聯的執行。如需目標和速率控制的詳細資訊，請參閱[了解 State Manager 關聯中的目標和速率控制](#)。

> **Important**
>
> 此程序說明如何建立使用 `Command` 或 `Policy` 文件將受管節點設為目標的關聯。如需建立使用 Automation Runbook 以鎖定節點或其他 AWS 資源類型之關聯的相關資訊，請參閱 [使用 State Manager 關聯排程自動化](#)。

開始之前

`targets` 參數是一系列的搜尋條件，使用您指定的 `Key`、`Value` 組合將節點設為目標。如果您打算使用 `targets` 參數在數十個或數百個節點上建立關聯，請在程序開始之前檢閱下列目標選項。

透過指定 ID 將特定節點設為目標

```
--targets Key=InstanceIds,Values=instance-id-1,instance-id-2,instance-id-3
```

```
--targets
 Key=InstanceIds,Values=i-02573cafcfEXAMPLE,i-0471e04240EXAMPLE,i-07782c72faEXAMPLE
```

使用 標籤將執行個體設為目標

```
--targets Key=tag:tag-key,Values=tag-value-1,tag-value-2,tag-value-3
```

```
--targets Key=tag:Environment,Values=Development,Test,Pre-production
```

使用 將節點設為目標 AWS Resource Groups

```
--targets Key=resource-groups:Name,Values=resource-group-name
```

```
--targets Key=resource-groups:Name,Values=WindowsInstancesGroup
```

將目前 AWS 帳戶 和 中的所有執行個體設為目標 AWS 區域

```
--targets Key=InstanceIds,Values=*
```

> **Note**
>
> 記下以下資訊。
>
> - 如果該文件是從另一個帳戶共用的，則 State Manager 不支援執行使用新版文件的關聯。如果是從另一個帳戶共用的，則 State Manager 一律執行文件的 `default` 版本，即使 Systems Manager 主控台顯示新版本已處理。如果您想要使用從另一個帳戶共用而來的新版本文件來執行關聯，則必須將文件版本設定為 `default`。
> - 您可以使用 AWS CLI指定最多五個標籤鍵。如果您使用 AWS CLI，`create-association`命令中指定的所有標籤索引鍵目前都必須指派給節點。否則，State Manager 無法將相應節點作為關聯的目標。
> - 在建立關聯時，指定排程的執行時間。使用 Cron 或 Rate 運算式來指定排程。如需 Cron 和 Rate 運算式的詳細資訊，請參閱關聯的 Cron 與 Rate 運算式。
> - 為了在偵測到新目標節點時套用透過 Automation 執行手冊建立的關聯，必須滿足特定條件。如需相關資訊，請參閱關於透過 Automation 執行手冊的目標更新。

建立關聯

1. AWS Tools for PowerShell如果您尚未安裝和設定 AWS CLI 或 。

如需相關資訊，請參閱安裝或更新 AWS CLI的最新版本和安裝 AWS Tools for PowerShell。

2. 使用以下格式來建立會建立 State Manager 關聯的命令。將每個########取代為您自己的資訊。

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"
```

以下範例會在加上 "Environment,Linux" 標籤的節點上建立關聯。關聯會使用 AWS-UpdateSSMAgent 文件在每週日的上午 2:00 UTC 更新目標節點上的 SSM Agent。此關聯在任何指定的時間可在最多 10 個節點上同時執行。此外，如果錯誤計數超過 5 個，此關聯會停止在更多節點上特定執行間隔的執行。針對合規報告，指派給此關聯的嚴重性等級為中等。

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
```

以下範例會透過指定萬用字元值 (*) 來設定目標節點 ID。這可讓 Systems Manager 在目前 AWS 帳戶 和 中的所有節點上建立關聯 AWS 區域。此關聯在任何指定的時間可在最多 10 個節點上同時執行。此外，如果錯誤計數超過 5 個，此關聯會停止在更多節點上特定執行間隔的執行。針對合規報告，指派給此關聯的嚴重性等級為中等。此關聯使用排程偏移，這意味著它在指定的 Cron 排程後會執行兩天。此外還包括 `ApplyOnlyAtCronInterval` 參數，這是使用排程偏移所必需的，意味著在建立後關聯不會立即執行。

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
```

以下範例會在 Resource Groups 中的節點上建立關聯。該群組名為「HR-Department」。該關聯使用 AWS-UpdateSSMAgent 文件，在每週日上午 2:00 UTC 更新目標執行個體上的 SSM Agent。此關聯在任何指定的時間可在最多 10 個節點上同時執行。此外，如果錯誤計數超過 5

個，此關聯會停止在更多節點上特定執行間隔的執行。針對合規報告，指派給此關聯的嚴重性等級為中等。此關聯會在指定的 Cron 排程執行。它不會在建立關聯之後立即執行。

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
```

以下範例會建立一個關聯，它在標記有特定節點 ID 的節點上執行。關聯會使用 SSM Agent 文件，當變更行事曆開啟時在目標節點上更新一次 SSM Agent。關聯會在執行時檢查行事曆狀態。如果行事曆在啟動時關閉且關聯只執行一次，則它不會再執行一次，因為關聯執行時段已結束。如果行事曆處於開啟狀態，則會相應地執行關聯。

> **Note**
>
> 如果您在變更行事曆關閉時，將新節點新增至關聯作用的標籤或資源群組，則在變更行事曆開啟後，關聯就會套用至這些節點。

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)"
```

以下範例會建立一個關聯，它在標記有特定節點 ID 的節點上執行。關聯會使用 SSM Agent 文件，在每週日的上午 2:00 更新目標節點上的 SSM Agent。當變更行事曆開啟時，此關聯只會在指定的 Cron 排程執行。建立關聯時，它會檢查行事曆狀態。如果行事曆已關閉，則不會套用關聯。當套用關聯的間隔在星期日凌晨 2:00 開始時，關聯會檢查行事曆是否已開啟。如果行事曆處於開啟狀態，則會相應地執行關聯。

> **Note**
>
> 如果您在變更行事曆關閉時，將新節點新增至關聯作用的標籤或資源群組，則在變更行事曆開啟後，關聯就會套用至這些節點。

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**
>
> 如果刪除您建立的關聯，關聯將不再在該關聯的任何目標上執行。此外，如果您已指定 `apply-only-at-cron-interval` 參數，則可以重設此選項。若要執行這項操作，請在透過命令列更新關聯時指定 `no-apply-only-at-cron-interval` 參數。此參數會在更新關聯後立即強制執行關聯，以及根據指定的間隔強制執行關聯。

編輯和建立關聯的新版本

您可以編輯 State Manager 關聯來指定新名稱、排程、嚴重性等級、目標或其他值。對於以 SSM Command 類型文件為基礎的關聯，您也可以選擇將命令的輸出寫入 Amazon Simple Storage Service (Amazon S3) 儲存貯體。在您編輯關聯後，State Manager 會建立新的版本。您可以依照下列程序，在編輯後檢視不同版本。

> **Note**
>
> 為了在偵測到新目標節點時套用透過 Automation 執行手冊建立的關聯，必須滿足特定條件。如需相關資訊，請參閱關於透過 Automation 執行手冊的目標更新。

下列程序說明如何使用 Systems Manager 主控台 AWS Command Line Interface (AWS CLI) 和 AWS Tools for PowerShell (Tools for PowerShell) 編輯和建立新的 關聯版本。

> **Important**
>
> 如果該文件是從另一個帳戶共用的，則 State Manager 不支援執行使用新版文件的關聯。若是從另一個帳戶共用的，則 State Manager 一律會執行文件的 `default` 版本，即使 Systems Manager 主控台顯示已處理新版本。如果您想要使用從另一個帳戶共用而來的新版本文件來執行關聯，則必須將文件版本設定為 `default`。

建立關聯 (主控台)

以下程序說明如何使用 Systems Manager 主控台來編輯和建立關聯的新版本。

> **Note**
>
> 對於使用 SSM Command 文件 (而非 Automation 執行手冊) 的關聯，此程序要求您擁有現有 Amazon S3 儲存貯體的寫入存取權。請注意，如果您未曾使用過 Amazon Simple Storage Service (Amazon S3)，您將需要支付使用 Amazon Simple Storage Service (Amazon S3) 的費用。如需建立儲存貯體的詳細資訊，請參閱建立儲存貯體。

編輯 State Manager 關聯

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 State Manager。
3. 選擇現有關聯，然後選擇編輯。
4. 重新設定關聯來符合目前的需求。

   如需使用 `Command` 和 `Policy` 文件關聯選項的相關資訊，請參閱建立關聯。如需使用 Automation 執行手冊關聯選項的相關資訊，請參閱使用 State Manager 關聯排程自動化。
5. 選擇 Save Changes (儲存變更)。
6. (選用) 若要在關聯頁面中檢視關聯資訊，請選擇編輯過的關聯的名稱，然後選擇版本索引標籤。系統會列出您所建立和編輯的關聯的每個版本。
7. (選用) 若要檢視以 SSM `Command` 文件為基礎的關聯的輸出，請執行下列動作：

   a. 開啟位於 https://console.aws.amazon.com/s3/ 的 Amazon S3 主控台。

   b. 選擇您指定來存放命令輸出的 Amazon Simple Storage Service (Amazon S3) 儲存貯體名稱，然後選擇以執行該關聯的節點 ID 命名的資料夾。(如果您選擇將輸出存放在儲存貯體中的資料夾，請先打開它。)

   c. 向下切入多個層級，從 `awsrunPowerShell` 資料夾到 `stdout` 檔案。

   d. 選擇 Open (開啟) 或 Download (下載) 以檢視主機名稱。

建立關聯 (命令列)

下列程序說明如何使用 AWS CLI （在 Linux 或 Windows 上）AWS Tools for PowerShell 或 編輯和建立新的 關聯版本。

編輯 State Manager 關聯

1. AWS Tools for PowerShell如果您尚未安裝和設定 AWS CLI 或 。

   如需相關資訊，請參閱安裝或更新 AWS CLI的最新版本和安裝 AWS Tools for PowerShell。

2. 使用以下格式來建立命令，以編輯和建立現有 State Manager 關聯的新版本。將每個########取代為您自己的資訊。

   > **Important**
   >
   > 當您呼叫 update-association 時，系統會從請求中卸除所有選用參數，並覆寫與這些參數之空值的關聯。這是設計本身所致。您必須在呼叫中指定所有選用參數，即使您未變更參數。這包括 --name 參數。呼叫此動作之前，建議您呼叫 describe-association 操作，並記下 update-association 呼叫所需的所有選用參數。

   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
```

以下範例會更新現有關聯，將名稱變更為 TestHostnameAssociation2。新的關聯版本會每個小時執行，並將命令的輸出寫入指定 Amazon Simple Storage Service (Amazon S3) 儲存貯體。

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 * * ? *)"
```

以下範例會更新現有關聯，將名稱變更為 CalendarAssociation。新的關聯會在行事曆開啟時執行，並將命令輸出寫入指定的 Amazon Simple Storage Service (Amazon S3) 儲存貯體。

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"
```

以下範例會更新現有關聯，將名稱變更為 MultiCalendarAssociation。新的關聯會在行事曆開啟時執行，並將命令輸出寫入指定的 Amazon Simple Storage Service (Amazon S3) 儲存貯體。

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. 若要檢視關聯的新版本，請執行以下命令。

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 *
```

系統會傳回如下資訊。

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}
```

刪除關聯

利用以下程序，使用 AWS Systems Manager 主控台來刪除關聯。

刪除關聯

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 State Manager。
3. 選擇關聯，然後選擇刪除。

您可以從 AWS Systems Manager 主控台執行自動化，在單一操作中刪除多個關聯。在選取要刪除的多個關聯時，State Manager 會啟動 Automation 執行手冊開始頁面，其中的關聯 ID 會以輸入參數值的形式輸入。

在單一操作中刪除多個關聯

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 State Manager。
3. 選擇您要刪除的每個關聯，然後選擇刪除。
4. (選用) 在其他輸入參數區域中，選取要自動化在執行時使用的擔任角色的 Amazon Resource Name (ARN)。若要建立新的擔任角色，請選擇建立。
5. 選擇提交。

執行具有關聯的 Auto Scaling 群組

使用關聯來執行 Auto Scaling 群組的最佳實務是使用標籤目標。不使用標籤可能會導致您達到關聯限制。

如果所有節點都標記有相同的金鑰和值，則只需要一個關聯來執行 Auto Scaling 群組。以下程序說明如何建立此類關聯。

建立執行 Auto Scaling 群組的關聯

1. 確保 Auto Scaling 群組中的所有節點都標記有相同的金鑰和值。如需標記節點的詳細指示，請參閱 AWS Auto Scaling 使用者指南中的標記 Auto Scaling 群組和執行個體。
2. 使用 在 Systems Manager 中使用關聯 中的程序來建立關聯。

   如果您在主控台中工作，請選擇 Targets (目標) 欄位中的 Specify instance tags (指定執行個體標籤)。對於 Instance tags (執行個體標籤)，輸入 Auto Scaling 群組的 Tag (標籤) 鍵和值。

   如果您使用的是 AWS Command Line Interface (AWS CLI)，請指定`--targets Key=tag:`*`tag-key`*`,Values=`*`tag-value`*索引鍵和值與標記節點的位置。

檢視關聯歷史記錄

您可以使用 DescribeAssociationExecutions API 操作來檢視特定關聯 ID 的所有執行。使用此操作來查看State Manager關聯的狀態、詳細狀態、結果、上次執行時間，以及更多資訊。 State Manager 是 中

的工具 AWS Systems Manager。此 API 操作還包括篩選條件，可協助您依據指定的條件找出關聯。例如，您可以指定確切的日期和時間，並使用 GREATER_THAN (大於) 篩選條件來檢視指定日期和時間之後處理的執行。

例如，如果關聯執行失敗，您可以使用 DescribeAssociationExecutionTargets API 操作，深入探索特定執行的詳細資訊。此操作會顯示資源，例如節點 ID、關聯執行的位置和各種關聯狀態。接著，您可以查看哪些資源或節點無法執行關聯。透過資源 ID，您可以檢視命令執行詳細資訊，以查看命令中的哪一個步驟失敗。

本節中的範例也包括有關如何使用 StartAssociationsOnce API 操作在建立時執行一次關聯的相關資訊。您可以使用此 API 操作來調查失敗的關聯執行。若您看到關聯失敗，您可以對資源進行變更，然後立即執行關聯來查看資源上的變更是否允許關聯成功執行。

> **Note**
>
> 在關聯執行期間由 SSM 文件啟動的 API 操作不會記錄在 AWS CloudTrail中。

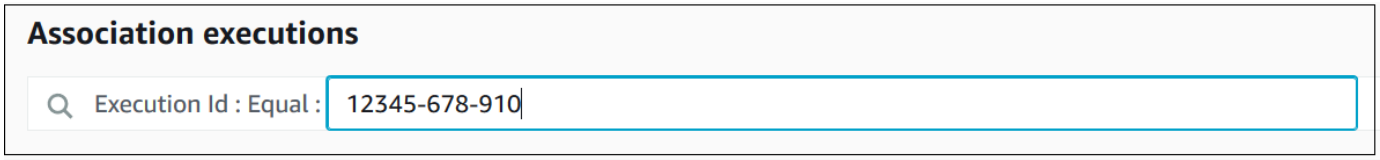
檢視關聯歷史記錄 (主控台)

使用以下程序來檢視特定關聯 ID 的執行歷史記錄，然後檢視一或多個資源的執行詳細資訊。

檢視特定關聯 ID 的執行歷史記錄

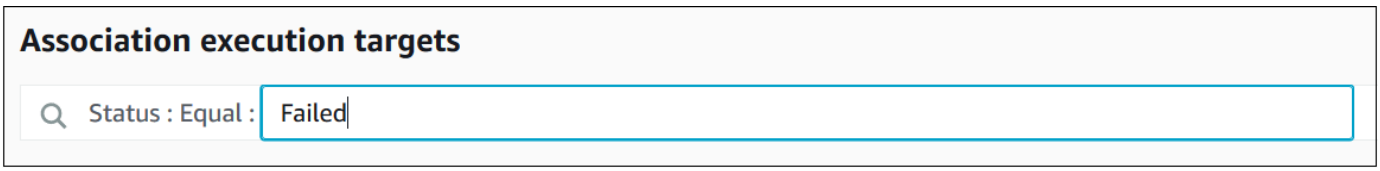
1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 選擇 State Manager。
3. 在 Association id (關聯 ID) 欄位中，選擇您想檢視歷程記錄的關聯。
4. 選擇 View details (檢視詳細資訊) 按鈕。
5. 選擇 Execution history (執行歷程記錄) 標記。
6. 選擇您想檢視資源層級執行詳細資訊的關聯。例如，選擇狀態顯示為 Failed (失敗) 的關聯。接著，您可以檢視執行關聯失敗的節點的執行詳細資訊。

   使用搜尋方框篩選條件，找出您想檢視詳細資訊的執行。



7. 選擇執行 ID。Association execution targets (關聯執行目標) 頁面隨即開啟。此頁面會顯示執行該關聯的所有資源。
8. 選擇資源 ID 來檢視該資源的特定資訊。

   使用搜尋方框篩選條件，找出您想檢視詳細資訊的資源。



9. 如果您正在調查執行失敗的關聯，您可以使用 Apply association now (立即套用關聯) 按鈕，以便在建立時執行一次關聯。在您對執行關聯失敗的資源進行變更後，選擇導覽導航列中的 Association ID (關聯 ID) 連結。
10. 選擇 Apply association now (立即套用關聯) 按鈕。在執行完成後，驗證關聯執行已成功。

檢視關聯歷史記錄 (命令列)

下列程序說明如何使用 AWS Command Line Interface (AWS CLI) （在 Linux 或 Windows 上） AWS Tools for PowerShell 或 檢視特定關聯 ID 的執行歷史記錄。在這之後，程序會說明如何檢視一或多個資源的執行詳細資訊。

檢視特定關聯 ID 的執行歷史記錄

1. AWS Tools for PowerShell如果您尚未安裝和設定 AWS CLI 或 。

   如需相關資訊，請參閱安裝或更新 AWS CLI的最新版本和安裝 AWS Tools for PowerShell。
2. 執行以下命令來檢視特定關聯 ID 的執行清單。

   Linux & macOS

```
aws ssm describe-association-executions \
  --association-id ID \
  --filters Key=CreatedTime,Value="2018-04-10T19:15:38.372Z",Type=GREATER_THAN
```

> **Note**
>
> 此命令包含篩選條件，可將篩選結果為僅限在特定日期和時間之後發生的執行。若要檢視特定關聯 ID 的所有執行，請移除 `--filters` 參數和

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**
>
> 此命令包含篩選條件，可將篩選結果為僅限在特定日期和時間之後發生的執行。若要檢視特定關聯 ID 的所有執行，請移除 --filters 參數和 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**
>
> 此命令包含篩選條件，可將篩選結果為僅限在特定日期和時間之後發生的執行。若要檢視特定關聯 ID 的所有執行，請移除 -Filter 參數和 @{"Key"="CreatedTime";"Value"="2019-06-01T19:15:38.372Z";"Type"="GREAT 值。

系統會傳回相關資訊，如下所示。

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
```

您可以使用一或多個篩選條件來限制結果。以下範例傳回所有於特定日期和時間之前執行的關聯。

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"}
```

以下範例傳回所有於特定日期和時間之後成功執行的關聯。

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. 執行以下命令來檢視執行特定執行的所有目標。

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
```

您可以使用一或多個篩選條件來限制結果。以下範例傳回所有執行特定關聯失敗的目標相關資訊。

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"
    }
```

以下範例傳回執行關聯失敗的特定受管節點相關資訊。

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. 若您正在調查執行失敗的關聯，您可以使用 StartAssociationsOnce API 操作來立即執行關聯 (僅限一次)。在您變更關聯執行失敗的資源後，請執行以下命令來立即執行關聯，並僅限一次。

   Linux & macOS

   ```
   aws ssm start-associations-once \
     --association-id ID
   ```

   Windows

   ```
   aws ssm start-associations-once ^
     --association-id ID
   ```

   PowerShell

   ```
   Start-SSMAssociationsOnce `
     -AssociationId ID
   ```

透過 IAM 使用關聯

State Manager中的工具 AWS Systems Manager使用目標來選擇您設定關聯的執行個體。最初，關聯是透過指定文件名稱 (Name) 和執行個體 ID (InstanceId) 來建立的。這會在文件和執行個體或

受管節點之間建立關聯。關聯通常由這些參數識別。這些參數現在已被取代，但仍支援它們。資源 instance 和 managed-instance 作為資源新增到具有 Name 和 InstanceId 的動作。

AWS Identity and Access Management (IAM) 政策強制執行行為取決於指定的資源類型。僅根據傳入的請求強制執行 State Manager 的資源操作。State Manager 不會對帳戶中資源的屬性執行深入檢查。只有在請求參數包含指定的政策資源時，才會針對政策資源驗證請求。例如，如果您在資源區塊中指定執行個體，若請求使用 InstanceId 參數，則會強制執行政策。不會針對該 InstanceId 來檢查帳戶中每個資源的 Targets 參數。

以下是具有迷惑行為的一些案例：

- DescribeAssociation、DeleteAssociation 和 UpdateAssociation 使用 instance、managed-instance 以及 document 資源來指定參考關聯的已取代方式。這包括使用已取代的 InstanceId 參數建立的所有關聯。
- CreateAssociation、CreateAssociationBatch 以及 UpdateAssociation 使用 instance 和 managed-instance 資源來指定參考關聯的已取代方式。這包括使用已取代的 InstanceId 參數建立的所有關聯。document 資源類型是參考關聯的已取代方式的一部分，並且是關聯的實際屬性。這表示您可以根據文件名稱，使用 Create 和 Update 動作的 Allow 或 Deny 許可來構建 IAM 政策。

如需有關搭配使用 IAM 政策與 Systems Manager 的詳細資訊，請參閱《服務授權參考》中的 適用於 AWS Systems Manager 的 Identity and Access Management 或適用於 AWS Systems Manager的動作、資源及條件金鑰。

建立執行 MOF 檔案的關聯

您可以使用 AWS-ApplyDSCMofs SSM 文件執行受管物件格式 (MOF) 檔案 AWS Systems Manager，在 Windows Server 受管節點上使用 中的State Manager工具 強制執行所需的狀態。AWS-ApplyDSCMofs 文件有兩種執行模式。運用第一種模式，您可以設定關聯，以便掃描並報告受管節點是否處於指定 MOF 檔案中定義的所需狀態。在第二種模式中，您可以執行 MOF 檔案，並根據資源和 MOF 檔案中所定義的資源值，來變更您節點的組態。AWS-ApplyDSCMofs 文件允許您從 Amazon Simple Storage Service (Amazon S3)、本機共用、或具有 HTTPS 網域的安全網站中下載和執行 MOF 組態檔案。

State Manager記錄和報告每個 MOF 檔案執行在每個關聯執行期間的狀態。State Manager也將每個 MOF 檔案執行的輸出以合規事件報告，您可在 AWS Systems Manager 合規頁面中檢視。

MOF 檔案執行建構於 Windows PowerShell Desired State Configuration (PowerShell DSC)。PowerShell DSC 是一個宣告式平台，用於設定、部署和管理 Windows 系統。PowerShell

DSC 可讓管理員以稱為 DSC 設定的簡單文字文件，描述想要如何設定伺服器。PowerShell DSC 設定是一種特殊的 PowerShell 指令碼，用於說明要做什麼，而非做法。執行設定會產生一份 MOF 檔案。MOF 檔案可套用到一或多個伺服器，以達到伺服器所需的設定。PowerShell DSC 資源負責強制執行設定。如需詳細資訊，請參閱 Windows PowerShell Desired State Configuration 概觀。

主題

- 使用 Amazon Simple Storage Service (Amazon S3) 存放成品
- 在 MOF 檔案解析登入資料
- 在 MOF 檔案使用字符
- 建立執行 MOF 檔案之關聯的先決條件
- 建立執行 MOF 檔案的關聯
- 對建立執行 MOF 檔案的關聯時出現的問題進行疑難排解
- 檢視 DSC 資源合規詳細資訊

使用 Amazon Simple Storage Service (Amazon S3) 存放成品

如果您使用 Amazon S3 存放 PowerShell 模組、MOF 檔案、合規報告或狀態報告，則 AWS Systems Manager SSM Agent所使用的 AWS Identity and Access Management (IAM) 角色必須具有 儲存貯體的 `GetObject`和 `ListBucket`許可。如果您不提供這些許可，系統會傳回 Access Denied (存取遭拒) 錯誤。以下是有關在 Amazon Simple Storage Service (Amazon S3) 中存放成品的重要資訊。

- 如果儲存貯體位於不同的 中 AWS 帳戶，請建立授予帳戶 （ 或 IAM 角色 ） `GetObject`和`ListBucket`許可的儲存貯體資源政策。
- 如果您想要使用自訂的 DSC 資源，您可以從 Amazon Simple Storage Service (Amazon S3) 儲存貯體下載這些資源。您也可以從 PowerShell gallery 自動進行安裝。
- 如果您使用 Amazon Simple Storage Service (Amazon S3) 作為模組來源，請以下列區分大小寫的格式上傳模組的 Zip 檔：*`ModuleName_ModuleVersion`*.zip。例如：MyModule_1.0.0.zip。
- 所有檔案必須位於儲存貯體的根資料夾。不支援資料夾結構。

在 MOF 檔案解析登入資料

登入資料是透過使用 AWS Secrets Manager 或 AWS Systems Manager Parameter Store解析。這可讓您設定自動登入資料輪換。這也允許 DSC 將憑證自動傳播到您的伺服器，無需重新部署 MOF。

若要在組態中使用 AWS Secrets Manager 秘密，請建立 PSCredential 物件，其中使用者名稱是包含憑證之秘密的 SecretId 或 SecretARN。您可以為密碼指定任何值。值會被忽略。以下是範例。

```
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
       }
    }
}
```

使用組態資料中的 PsAllowPlaintextPassword 設定編譯您的 MOF。這是可行的，因為登入資料只會包含標記。

在 Secrets Manager 中，確保節點在 IAM 受管政策中，以及選擇性在機密資源政策 (如有的話) 中擁有 GetSecretValue 存取權限。如要使用 DSC，秘密的格式必須如下。

```
{ 'Username': 'a_name', 'Password': 'a_password' }
```

機密可以具有其他屬性 (例如，用於輪換的屬性)，但必須至少具有使用者名稱和密碼屬性。

我們建議您使用多使用者輪換方法，其中有兩個不同的使用者名稱和密碼，輪換 AWS Lambda 函數會在它們之間翻轉。這個方法可讓您擁有多個作用中的帳戶，同時排除在輪換時鎖定使用者的風險。

在 MOF 檔案使用字符

字符讓您能夠在 MOF 編譯完成後，修改資源屬性值。這允許您在需要類似組態的多個伺服器上重複使用常見的 MOF 檔案。

字符替換僅適用於 `String` (字串) 類型的資源屬性。然而，如果您的資源擁有巢狀 CIM 節點屬性，其也會從該 CIM 節點中的 `String` 屬性解析字符。您無法在數字或陣列使用字符替換。

例如，假設您使用的是 xComputerManagement 資源，而您想要使用 DSC 重新命名電腦。一般而言，您需要擁有該機器專用的 MOF 檔案。然而，有了字符支援，您可以建立一個 MOF 檔案並套用到所有節點。在 `ComputerName` 屬性中，您不需要將電腦名稱硬編碼至 MOF，只需要使用執行個體標籤類型字符。值會在 MOF 剖析時進行解析。請參閱以下範例。

```
Configuration MyConfig
{
    xComputer Computer
    {
        ComputerName = '{tag:ComputerName}'
    }
}
```

接著，在 Systems Manager 主控台中的受管節點上設定標籤，或是在 Amazon EC2 主控台中設定 Amazon Elastic Compute Cloud (Amazon EC2) 標籤。當您執行文件時，指令碼會替換執行個體標籤值的 {tag:ComputerName} 字符。

您也可以在單一屬性結合多個標籤，如下例中所示。

```
Configuration MyConfig
{
    File MyFile
    {
        DestinationPath = '{env:TMP}\{tag:ComputerName}'
        Type = 'Directory'
    }
}
```

您可以使用五種不同類型的字符：

- tag：Amazon EC2 或受管節點標籤。
- tagb64：與 tag 相同，但系統使用 base64 來將值解碼。這可讓您在標籤值使用特殊字元。
- env：解析環境變數。
- ssm：Parameter Store 值。僅支援 String 和 Secure String 類型。
- tagssm：與標籤相同，但如果節點上未設定該標籤，系統會嘗試從具有相同名稱的 Systems Manager 參數中解析該值。當您想要擁有「預設全域值」，但希望能在單一節點 (例如 one-box 部署) 上進行覆寫時，這是非常實用的。

以下是使用 `ssm` 字符類型的 Parameter Store 範例。

```
File MyFile
{
    DestinationPath = "C:\ProgramData\ConnectionData.txt"
    Content = "{ssm:%servicePath%/ConnectionData}"
```

```
}
```

字符可讓 MOF 檔案變成泛型和可重複使用，在減少冗餘的程式碼上扮演重要角色。如果您可以避開伺服器特定的 MOF 檔案，則不需要 MOF 建置服務。MOF 建置服務會增加成本，減緩佈建速度，並提高節點群組之間組態不一致的風險，因為在編譯 MOF 時，會在組建伺服器上安裝不同的模組版本。

建立執行 MOF 檔案之關聯的先決條件

建立執行 MOF 檔案的關聯之前，確認您的受管節點已安裝下列先決條件：

- Windows PowerShell 5.0 或更新版本。如需詳細資訊，請參閱 Microsoft.com 的 Windows PowerShell 系統要求。
- AWS Tools for Windows PowerShell 3.3.261.0 版或更新版本。
- SSM Agent 2.2 或更新版本。

建立執行 MOF 檔案的關聯

建立會執行 MOF 檔案的關聯

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 State Manager。
3. 選擇 State Manager，然後選擇 Create association (建立關聯)。
4. 在 Name (名稱) 欄位中指定名稱。此為選用操作，但建議您採用。名稱可協助您了解所建立關聯的目的。名稱中不得使用空格。
5. 在 Document (文件) 清單中，請選擇 **AWS-ApplyDSCMofs**。
6. 在 Parameters (參數) 區段中，指定您對所需和選用輸入參數的選擇。

   a. Mofs To Apply (要套用的 MOF)：指定一或多個執行此關聯時要執行的 MOF 檔案。使用逗號來分隔 MOF 檔案清單。您可以指定以下用來找出 MOF 檔案的選項：

      - Amazon Simple Storage Service (Amazon S3) 儲存貯體名稱。儲存貯體名稱必須使用小寫字母。使用以下格式指定此資訊。

        ```
        s3:amzn-s3-demo-bucket:MOF_file_name.mof
        ```

        如果您想要指定 AWS 區域，請使用下列格式。

```
s3:bucket_Region:amzn-s3-demo-bucket:MOF_file_name.mof
```

- 一個安全網站。使用以下格式指定此資訊。

```
https://domain_name/MOF_file_name.mof
```

請見此處範例。

```
https://www.example.com/TestMOF.mof
```

- 在本機共享的檔案系統。使用以下格式指定此資訊。

```
\server_name\shared_folder_name\MOF_file_name.mof
```

請見此處範例。

```
\StateManagerAssociationsBox\MOFs_folder\MyMof.mof
```

b. Service Path (服務路徑)：(選用) 服務路徑可以是您想要寫入報告和狀態資訊的 Amazon Simple Storage Service (Amazon S3) 儲存貯體字首。或是Parameter Store參數基礎標籤的路徑。解析參數基礎標籤時，系統會使用 {ssm:%servicePath%/*parameter_name*}，將 servicePath 值注入參數名稱。例如，如果您的服務路徑為「WebServers/Production」，則系統會將參數解析為：WebServers/Production/*parameter_name*。當您在相同帳戶中執行多個環境時，這是非常實用的。

c. Report Bucket Name (報告儲存貯體名稱)：(選用) 輸入您想要寫入合規資料的 Amazon Simple Storage Service (Amazon S3) 儲存貯體名稱。報告會以 JSON 格式儲存在這個儲存貯體中。

> **Note**
>
> 您可以根據儲存貯體的所在區域，為儲存貯體名稱加上以區域為名的字首。例如：us-west-2:MyMOFBucket。如果您在 us-east-1 以外的特定區域中使用 Amazon Simple Storage Service (Amazon S3) 端點的代理，則應為儲存貯體名稱加上以區域為名的字首。如果儲存貯體名稱未包含字首，則可透過使用 us-east-1 端點自動探索儲存貯體區域。

d. Mof Operation Mode (Mof 操作模式)：執行 **AWS-ApplyDSCMofs** 關聯時選擇 State Manager 行為：

- Apply (套用)：修正不合規的節點組態。
- ReportOnly (僅報告)：不修正節點組態，但記錄所有合規資料並回報不合規的節點。

e. Status Bucket Name (狀態儲存貯體名稱)：(選用) 輸入您想要寫入 MOF 執行狀態資訊的 Amazon Simple Storage Service (Amazon S3) 儲存貯體名稱。這些狀態報告是節點最近一次合規執行的單一摘要。這表示，報告會在關聯下一次執行 MOF 檔案時遭到覆寫。

> **Note**
>
> 您可以根據儲存貯體的所在區域，為儲存貯體名稱加上以區域為名的字首。範例如下：`us-west-2:amzn-s3-demo-bucket`。如果您在 us-east-1 以外的特定區域中使用 Amazon Simple Storage Service (Amazon S3) 端點的代理，則應為儲存貯體名稱加上以區域為名的字首。如果儲存貯體名稱未包含字首，則會使用 us-east-1 端點自動探索儲存貯體區域。

f. Module Source Bucket Name (模組來源儲存貯體名稱)：(選用) 輸入包含 PowerShell 模組檔案的 Amazon Simple Storage Service (Amazon S3) 儲存貯體名稱。如果您指定 None (無)，請為下一個選項 Allow PS Gallery Module Source (允許 PS Gallery 模組來源) 選擇 True (True)。

> **Note**
>
> 您可以根據儲存貯體的所在區域，為儲存貯體名稱加上以區域為名的字首。範例如下：`us-west-2:amzn-s3-demo-bucket`。如果您在 us-east-1 以外的特定區域中使用 Amazon Simple Storage Service (Amazon S3) 端點的代理，則應為儲存貯體名稱加上以區域為名的字首。如果儲存貯體名稱未包含字首，則會使用 us-east-1 端點自動探索儲存貯體區域。

g. Allow PS Gallery Module Source (允許 PS Gallery 模組來源)：(選擇性) 選擇 True 以從 https://www.powershellgallery.com/ 下載 PowerShell 模組。如果您選擇 False，請為前一個選項 ModuleSourceBucketName 中指定來源。

h. Proxy Uri (代理 Uri)：(選擇性) 使用此選項，從代理伺服器下載 MOF 檔案。

i. Reboot Behavior (重新啟動行為)：(選擇性) 如果您的 MOF 檔案執行需要重新啟動，指定以下其中一個重新啟動行為：

- AfterMof (MOF 之後)：在完成所有 MOF 執行後重新啟動節點。即使有多個 MOF 執行要求重新啟動，系統會等到所有 MOF 執行完成之後，才重新啟動。
- Immediately (立即)：每當 MOF 執行提出要求，就立即重新啟動節點。如果執行的多個 MOF 檔案請求重新啟動，則會重新啟動節點多次。
- Never (從不)：不會重新啟動節點，即使 MOF 執行明確要求重新啟動。

j. Use Computer Name For Reporting (使用電腦名稱進行報告)：(選用) 開啟此選項，以在報告合規資訊時使用電腦名稱。預設值為 false，這表示報告合規資訊時，系統會使用節點 ID。

k. Turn on Verbose Logging (開啟詳細記錄)：(選用) 建議您在首次部署 MOF 檔案時開啟詳細記錄。

> **Important**
>
> 允許後，與標準關聯執行記錄相比，詳細記錄能將更多的資料寫入 Amazon Simple Storage Service (Amazon S3) 儲存貯體。這可能導致效能變慢，及更高的 Amazon Simple Storage Service (Amazon S3) 儲存費用。為了減緩儲存空間大小問題，建議您在 Amazon Simple Storage Service (Amazon S3) 儲存貯體上開啟生命週期政策。如需詳細資訊，請參閱《Amazon Simple Storage Service 主控台使用者指南》中的如何建立 S3 儲存貯體的生命週期政策？。

l. Turn on Debug Logging (開啟偵錯記錄)：(選用) 建議開啟偵錯記錄，以便排除 MOF 錯誤。我們也建議您針對一般使用停用此選項。

> **Important**
>
> 允許後，與標準關聯執行記錄相比，偵錯記錄能將更多的資料寫入 Amazon Simple Storage Service (Amazon S3) 儲存貯體。這可能導致效能變慢，及更高的 Amazon Simple Storage Service (Amazon S3) 儲存費用。為了減緩儲存空間大小問題，建議您在 Amazon Simple Storage Service (Amazon S3) 儲存貯體上開啟生命週期政策。如需詳細資訊，請參閱《Amazon Simple Storage Service 主控台使用者指南》中的如何建立 S3 儲存貯體的生命週期政策？。

m. Compliance Type (合規類型)：(選擇性) 指定報告合規資訊時使用的合規類型。預設合規類型為 Custom:DSC。如果您建立多個執行 MOF 檔案的關聯，請務必為每個關聯指定不同的合規類型。如果您沒有執行此作業，每個使用 Custom:DSC 的額外關聯都會覆寫現有的合規資料。

n. Pre Reboot Script (指令碼在重新啟動前)：(選擇性) 指定若組態指示重新啟動為必要時，所要執行的指令碼。指令碼會在重新啟動之前執行。指令碼必須為單行。使用分號分隔其他行。

7. 在 Targets (目標) 區段，選擇 Specifying tags (指定標籤) 或 Manually Selecting Instance (手動選取執行個體)。如果您選擇使用標籤將資源設為目標，請在提供的欄位中輸入標籤索引鍵和標籤值。如需使用目標的詳細資訊，請參閱了解 State Manager 關聯中的目標和速率控制。
8. 在 Specify schedule (指定排程) 區段中，選擇 On Schedule (按照排程) 或 No schedule (無排程)。如果您選擇 On Schedule (按照排程)，則使用提供的按鈕來為關聯建立 Cron 或 Rate 排程。
9. 在 Advanced options (進階選項) 區段中：

   - 在 Compliance severity (合規嚴重性) 中，選擇關聯的嚴重性等級。合規報告會指出關聯狀態合規與否，以及您在這裡指示的嚴重性等級。如需詳細資訊，請參閱關於State Manager關聯合規。
10. 在 Rate control (速率控制) 區段中，針對在受管節點機群之間執行State Manager關聯設定選項。如需關於這些選項的詳細資訊，請參閱 了解 State Manager 關聯中的目標和速率控制。

    在 Concurrency (並行) 部分，選擇一個選項：

    - 選擇 targets (目標)，輸入可以同時執行關聯的目標絕對數量。
    - 選擇 percentage (百分比)，輸入可以同時執行關聯的目標集百分比。

    在 Error threshold (錯誤閾值) 部分，選擇一個選項：

    - 選擇 errors (錯誤)，輸入State Manager停止在額外目標執行關聯之前允許的錯誤絕對數量。
    - 選擇 percentage (百分比)，輸入State Manager停止在額外目標執行關聯之前允許的錯誤百分比。
11. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取啟用將輸出寫入 S3 方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

> **Note**
>
> 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給受管節點之執行個體設定檔的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確認與受管節點相關聯的執行個體描述檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

12. 選擇 Create Association (建立關聯)。

State Manager會在指定的節點或目標上建立並立即執行關聯。在初次執行後，關聯會根據您定義的排程和以下規則，依間隔執行：

- 間隔開始時，State Manager 在線上節點上執行關聯，並略過離線節點。
- State Manager嘗試於間隔期間在所有已設定的節點上執行關聯。
- 如果未在間隔期間執行關聯 (例如，因為並行值限制了一次所能處理關聯的節點數目)，則 State Manager 會嘗試在下一個間隔期間執行關聯。
- State Manager會記錄所有略過的間隔的歷程記錄。您可以在 Execution History (執行歷程記錄) 標記檢視歷程記錄。

> **Note**
>
> `AWS-ApplyDSCMofs` 是 Systems Manager 命令文件。這表示您也可以使用 中的Run Command工具 來執行此文件 AWS Systems Manager。如需詳細資訊，請參閱AWS Systems Manager Run Command。

對建立執行 MOF 檔案的關聯時出現的問題進行疑難排解

本節包括一些資訊，可協助您排除建立執行 MOF 檔案的關聯時所發生的問題。

開啟增強型記錄

故障診斷的第一個步驟是開啟增強型記錄。具體來說，請執行下列操作：

1. 確認關聯已設定為將命令輸出寫入 Amazon Simple Storage Service (Amazon S3) 或 Amazon CloudWatch Logs (CloudWatch)。
2. 將 Enable Verbose Logging (啟用詳細記錄) 參數設為 True。
3. 將 Enable Debug Logging (啟用偵錯記錄) 參數設為 True。

開啟詳細和偵錯記錄後，Stdout 輸出檔案會包括指令碼執行的詳細資訊。此輸出檔案可協助您找出指令碼失敗的位置。Stderr 輸出檔案包括指令碼執行期間發生的錯誤。

建立執行 MOF 檔案的關聯時的常見問題

本節包括一些資訊，介紹建立執行 MOF 檔案的關聯時可能發生的常見問題，以及排除這些問題的步驟。

我的 MOF 未套用

如果State Manager無法將關聯套用到您的節點，則先檢閱 Stderr 輸出檔案。這個檔案可協助您了解問題的根源。亦請確認下列內容：

- 節點具有所有 MOF 相關 Amazon Simple Storage Service (Amazon S3) 儲存貯體所需的存取許可。具體而言：
  - s3:GetObject 許可：對於私有 Amazon Simple Storage Service (Amazon S3) 儲存貯體中的 MOF 檔案及 Amazon Simple Storage Service (Amazon S3) 儲存貯體中的自訂模組，這是必要的。
  - s3:PutObject 許可：對於將合規報告和合規狀態寫入 Amazon Simple Storage Service (Amazon S3) 儲存貯體，這是必要的。
- 如果您使用的是標籤，則確保節點具有所需的 IAM 政策。若使用標籤，執行個體 IAM 角色必須具有允許 `ec2:DescribeInstances` 和 `ssm:ListTagsForResource` 動作的政策。
- 確保節點已指派預期的標籤或 SSM 參數。
- 確認標籤或 SSM 參數無拼寫錯誤。
- 嘗試在本機於節點上套用 MOF，以確保 MOF 檔案本身沒有問題。

我的 MOF 似乎無法執行，但 Systems Manager 執行成功

如果 `AWS-ApplyDSCMofs` 文件成功執行，則 Systems Manager 執行狀態會顯示為 Success (成功)。此狀態不會依照 MOF 檔案中的組態要求，反映您節點的合規狀態。若要檢視您節點的合規狀態，請查看合規報告。您可以在 Amazon Simple Storage Service (Amazon S3) 報告儲存貯體中檢視 JSON 報告。這適用於 Run Command 和State Manager執行。此外，對於 State Manager，您可以在 Systems Manager 合規頁面查看合規詳細資訊。

Stderr 狀態：嘗試連接服務時名稱解析失敗

此錯誤表示指令碼無法連接遠端服務。最有可能是指令碼無法連接 Amazon Simple Storage Service (Amazon S3)。此問題通常發生在指令碼嘗試將合規報告或合規狀態寫入文件參數中提供的 Amazon Simple Storage Service (Amazon S3) 儲存貯體時。一般而言，此錯誤發生在運算環境使用包括允許清單的防火牆或透明代理時。若要解決此問題：

- 對於所有 Amazon Simple Storage Service (Amazon S3) 儲存貯體參數，請使用區域特定的儲存貯體語法。例如，Mofs to Apply (要套用的 MOF) 參數應採用以下格式：

s3：*bucket-region*：*amzn-s3-demo-bucket*：*mof-file-name*.mof。

請見此處範例： `s3:us-west-2:amzn-s3-demo-bucket:my-mof.mof`

報告、狀態和模組來源儲存貯體名稱應採用以下格式。

bucket-region：*amzn-s3-demo-bucket*。請見此處範例：`us-west-1:amzn-s3-demo-bucket;`

- 如果區域特定的語法無法修正問題，則確保目標節點可以存取想要的區域中的 Amazon Simple Storage Service (Amazon S3)。為了確認：
  1. 在適當的 Amazon Simple Storage Service (Amazon S3) 區域中尋找 Amazon Simple Storage Service (Amazon S3) 的端點名稱。如需詳細資訊，請參閱《Amazon Web Services 一般參考》中的 Amazon S3 服務端點一節。
  2. 登入到目標節點並執行以下 ping 命令。

     ```
     ping s3.s3-region.amazonaws.com
     ```

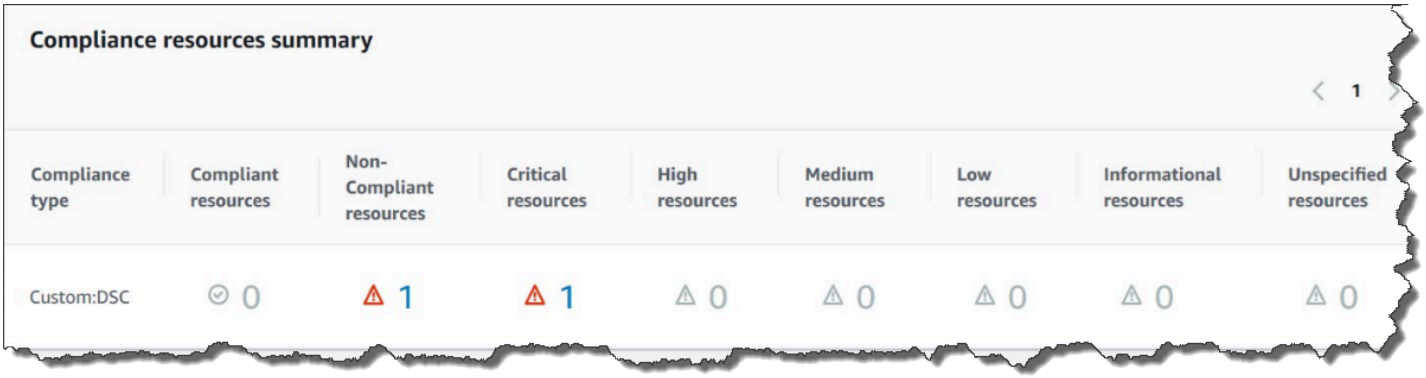
     如果 ping 失敗，則表示 Amazon Simple Storage Service (Amazon S3) 出現故障或防火牆/透明代理封鎖對 Amazon Simple Storage Service (Amazon S3) 區域的存取，或是節點無法存取網際網路。

檢視 DSC 資源合規詳細資訊

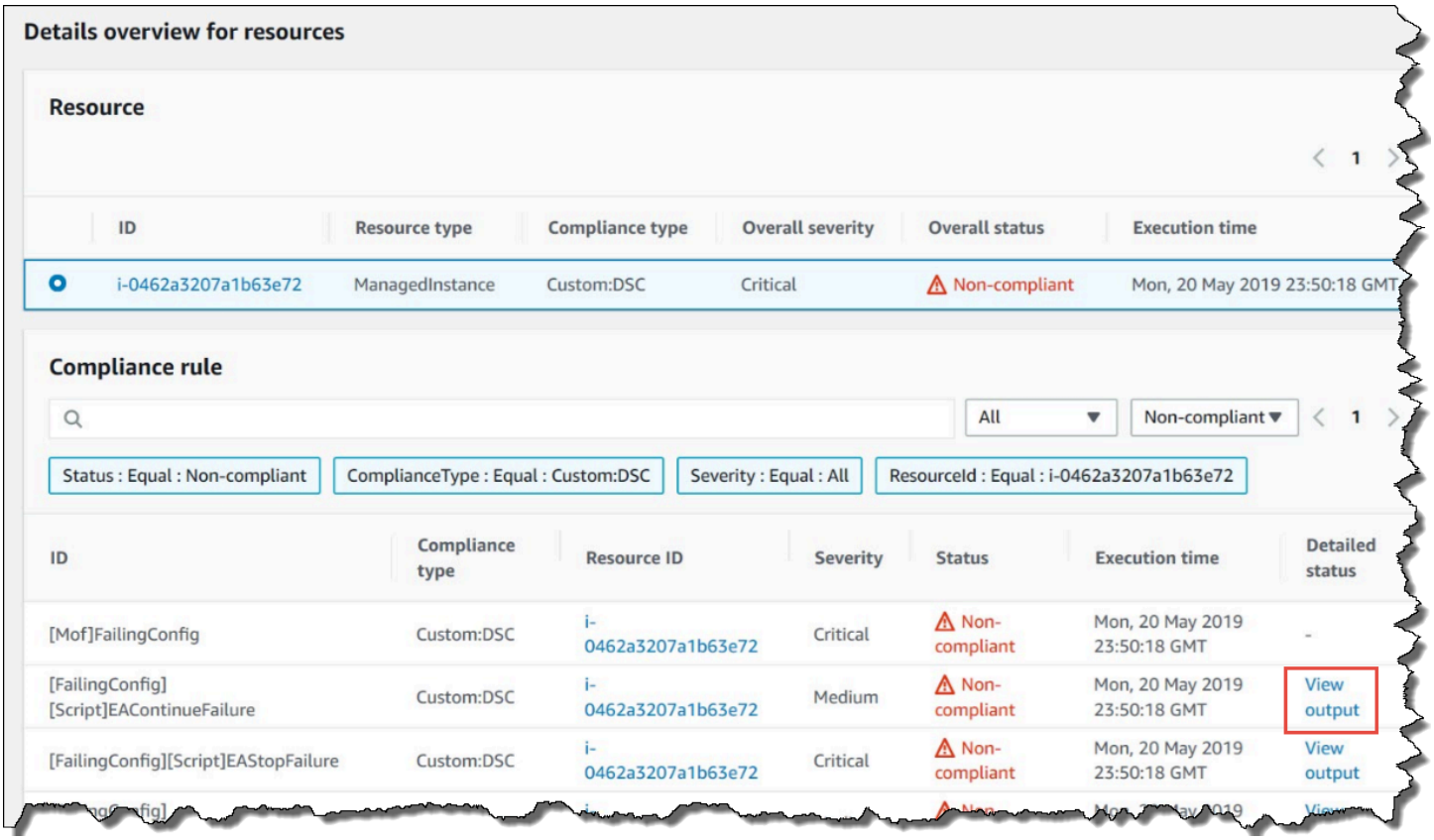
當您執行 `AWS-ApplyDSCMofs` 文件時，Systems Manager 會擷取您指定的 Amazon Simple Storage Service (Amazon S3) 狀態儲存貯體中的 DSC 資源故障的合規資訊。在 Amazon Simple Storage Service (Amazon S3) 儲存貯體中搜尋 DSC 資源故障相關資訊可能非常耗時。您可以改為在 Systems Manager Compliance (合規) 頁面中檢視此資訊。

合規資源摘要區段顯示失敗的資源計數。在以下範例中，ComplianceType 是 Custom:DSC 和不合規的一個資源。

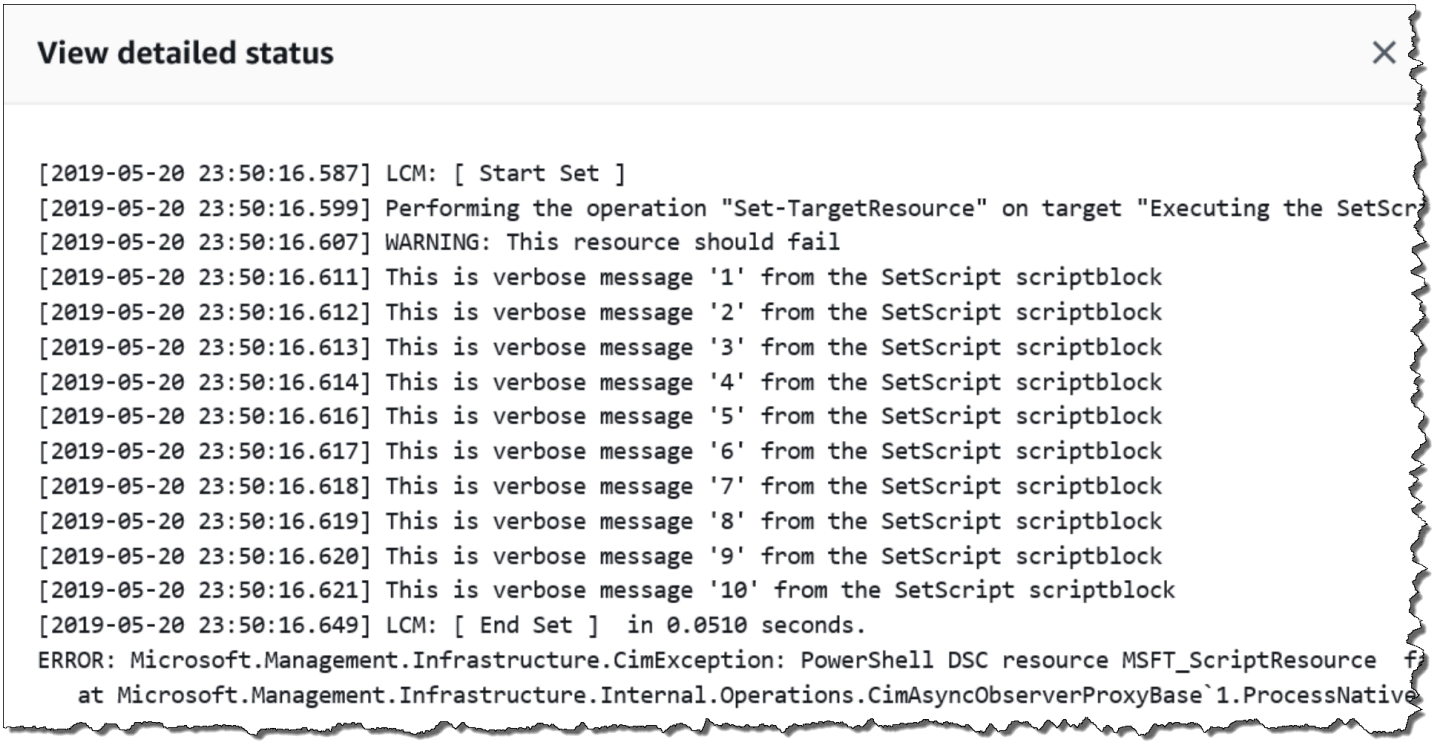
> **Note**
>
> Custom:DSC 是 `AWS-ApplyDSCMofs` 文件中的預設 ComplianceType 值。這個值是可自訂的。



資源的詳細資訊概觀區段會顯示 AWS 資源與不合規 DSC 資源的相關資訊。本章節也包含 MOF 名稱、指令碼執行步驟，以及 (如適用) 檢視輸出連結，以檢視詳細的狀態資訊。



View output (檢視輸出) 連結會顯示詳細狀態的最後 4,000 個字元。Systems Manager 會使用例外狀況做為第一個元素開始，然後掃描詳細訊息，並在到達 4,000 個字元配額之前加上盡可能多的字首。這個程序會顯示拋出例外狀況之前的輸出日誌訊息，也就是與故障診斷最相關的訊息。



如需如何檢視合規資訊的相關資訊，請參閱 AWS Systems Manager合規。

影響合規報告的狀況

若 State Manager 關聯失敗，則系統不會報告合規資料。更具體地說，若 MOF 無法處理，則 Systems Manager 不會報告任何合規項目，因為關聯會失敗。例如，若 Systems Manager 嘗試從節點沒有存取許可的 Amazon Simple Storage Service (Amazon S3) 儲存貯體下載 MOF，則關聯會失敗，也不會報告合規資料。

若第二個 MOF 中的資源失敗，則 Systems Manager 會報告合規資料。例如，若 MOF 嘗試在不存在的硬碟上建立檔案，則 Systems Manager 會報告合規，因為 AWS-ApplyDSCMofs 文件能夠完全處理，這表示關聯成功執行。

建立執行 Ansible 手冊的關聯

您可以使用 AWS-ApplyAnsiblePlaybooks SSM 文件建立執行Ansible手冊的State Manager關聯。State Manager 是 中的工具 AWS Systems Manager。本文件提供執行手冊的下列優點：

- 支援執行複雜的手冊
- 支援從 GitHub 和 Amazon Simple Storage Service (Amazon S3) 下載手冊
- 支援壓縮的手冊結構
- 增強型日誌

- 能夠指定綁定多個手冊時，要執行哪個手冊

> **Note**
>
> Systems Manager 包含兩個 SSM 文件，允許您建立執行 Ansible 手冊的 State Manager 關聯：`AWS-RunAnsiblePlaybook` 和 `AWS-ApplyAnsiblePlaybooks`。`AWS-RunAnsiblePlaybook` 文件已棄用。它在 Systems Manager 中仍維持可用，以用於舊版用途。由於此處所述的增強功能，我們建議您使用 `AWS-ApplyAnsiblePlaybooks` 文件。macOS 不支援執行 Ansible 手冊的關聯。

支援執行複雜的手冊

`AWS-ApplyAnsiblePlaybooks` 文件支援綁定的複雜手冊，因為它可在執行指定的主要手冊之前，將整個檔案結構複製到本機目錄。您可以提供 Zip 檔案或目錄結構的來源手冊。Zip 檔案或目錄可存放在 GitHub 或 Amazon S3 中。

支援從 GitHub 下載手冊

`AWS-ApplyAnsiblePlaybooks` 文件會使用 `aws:downloadContent` 外掛程式來下載手冊檔案。檔案可以儲存於 GitHub 的單一檔案中，也可以儲存成一組手冊檔案。若要從 GitHub 中下載內容，請指定 JSON 格式的 GitHub 儲存庫相關資訊。請見此處範例。

```
{
   "owner":"TestUser",
   "repository":"GitHubTest",
   "path":"scripts/python/test-script",
   "getOptions":"branch:master",
   "tokenInfo":"{{ssm-secure:secure-string-token}}"
}
```

支援從 Amazon Simple Storage Service (Amazon S3) 中下載手冊

您也能以單一 .zip 檔案或目錄結構的形式，在 Amazon Simple Storage Service (Amazon S3) 中儲存和下載 Ansible 手冊。若要從 Amazon Simple Storage Service (Amazon S3) 中下載內容，請指定檔案的路徑。以下是兩個範例。

範例 1：下載特定的手冊檔案

```
{
```

```
   "path":"https://s3.amazonaws.com/amzn-s3-demo-bucket/playbook.yml"
}
```

範例 2：下載目錄的內容

```
{
   "path":"https://s3.amazonaws.com/amzn-s3-demo-bucket/ansible/webservers/"
}
```

> **Important**
>
> 如果您指定 Amazon S3，則受管節點上的 AWS Identity and Access Management (IAM) 執行個體設定檔必須包含 S3 儲存貯體的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可。

支援壓縮的手冊結構

AWS-ApplyAnsiblePlaybooks 文件允許您執行下載套件中的壓縮 .zip 檔。此文件會檢查下載的檔案是否包含 .zip 格式的壓縮檔案。如果找到 .zip，文件會自動解壓縮檔案，然後執行指定的 Ansible 自動化。

增強型日誌

AWS-ApplyAnsiblePlaybooks 文件包含選擇性參數，用於指定不同層級的日誌。指定 -v 表示低詳細資訊等級，-vv 或 -vvv 表示中詳細資訊等級，-vvvv 表示偵錯等級日誌。這些選項會直接映射到 Ansible 詳細資訊選項。

能夠指定綁定多個手冊時，要執行哪個手冊

AWS-ApplyAnsiblePlaybooks 文件包含必要參數，用於指定綁定多個手冊時要執行哪個手冊。此選項提供執行手冊的彈性，以支援不同的使用案例。

了解已安裝的相依性

如果您為 InstallDependencies 參數指定 True，則 Systems Manager 會驗證您的節點是否已安裝下列相依性：

- Ubuntu Server/Debian Server: Apt-get (套件管理)、Python 3、Ansible、Unzip
- Amazon Linux：Ansible

- RHEL：Python 3、Ansible、Unzip

如果找不到這些相依性中的一個或多個項目，則 Systems Manager 會自動安裝它們。

建立執行 Ansible 手冊的關聯 (主控台)

下列程序說明如何使用 Systems Manager 主控台來建立使用 `AWS-ApplyAnsiblePlaybooks` 文件執行 Ansible 手冊的 State Manager 關聯。

建立執行 Ansible 手冊的關聯 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 State Manager。
3. 選擇 State Manager，然後選擇 Create association (建立關聯)。
4. 針對 Name (名稱)，指定可協助您記住關聯用途的名稱。
5. 在 Document (文件) 清單中，請選擇 **AWS-ApplyAnsiblePlaybooks**。
6. 在 Parameters (參數) 區段中，針對 Source Type (來源類型)，選擇 GitHub 或 S3。

   GitHub

   如果您選擇 GitHub，請輸入下列格式的儲存庫資訊。

   ```
   {
      "owner":"user_name",
      "repository":"name",
      "path":"path_to_directory_or_playbook_to_download",
      "getOptions":"branch:branch_name",
      "tokenInfo":"{{(Optional)_token_information}}"
   }
   ```

   S3

   如果您選擇 S3，請輸入下列格式的路徑資訊。

   ```
   {
      "path":"https://s3.amazonaws.com/path_to_directory_or_playbook_to_download"
   }
   ```
7. 針對 Install Dependencies (安裝相依性)，選擇一個選項。

8. (選用) 針對 Playbook File (手冊檔案)，輸入檔案名稱。如果 Zip 檔包含手冊，則必須指定 Zip 檔的相對路徑。
9. (選用) 在額外變數欄位中，輸入您要 State Manager 在執行時期傳送到 Ansible 的變數。
10. (選用) 針對 Check (檢查)，選擇一個選項。
11. (選用) 針對 Verbose (詳細資訊)，選擇一個選項。
12. 對於 Targets (目標)，請選擇選項。如需使用目標的詳細資訊，請參閱了解 State Manager 關聯中的目標和速率控制。
13. 在 Specify schedule (指定排程) 區段中，選擇 On schedule (按照排程) 或 No schedule (無排程)。如果您選擇 On schedule (按照排程)，則使用提供的按鈕來為關聯建立 Cron 或 Rate 排程。
14. 在 Advanced options (進階選項) 區段中，針對 Compliance severity (合規嚴重性)，選擇關聯的嚴重性等級。合規報告會指出關聯狀態合規與否，以及您在這裡指示的嚴重性等級。如需詳細資訊，請參閱關於State Manager關聯合規。
15. 在 Rate control (速率控制) 區段中，設定在受管節點機群之間執行State Manager關聯的選項。如需使用速率控制的詳細資訊，請參閱 了解 State Manager 關聯中的目標和速率控制。

    在 Concurrency (並行) 部分，選擇一個選項：

    - 選擇 targets (目標)，輸入可以同時執行關聯的目標絕對數量。
    - 選擇 percentage (百分比)，輸入可以同時執行關聯的目標集百分比。

    在 Error threshold (錯誤閾值) 部分，選擇一個選項：

    - 選擇 errors (錯誤)，輸入 State Manager 停止在額外目標執行關聯之前允許的錯誤絕對數量。
    - 選擇 percentage (百分比)，輸入 State Manager 停止在額外目標執行關聯之前允許的錯誤百分比。
16. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取啟用將輸出寫入 S3 方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

> **Note**
>
> 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給受管節點之執行個體設定檔的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確認與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

17. 選擇 Create Association (建立關聯)。

> **Note**
>
> 如果您使用標籤在一或多個目標節點上建立關聯，然後從節點移除標籤，則該節點將不再執行該關聯。系統會從 State Manager 文件中取消該節點的關聯。

建立執行 Ansible 手冊的關聯 (CLI)

下列程序說明如何使用 AWS Command Line Interface (AWS CLI) 建立使用 `AWS-ApplyAnsiblePlaybooks` 文件執行Ansible手冊的State Manager關聯。

建立執行 Ansible 手冊的關聯 (CLI)

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

2. 執行以下其中一個命令來建立關聯，該關聯透過使用標籤將節點設為目標來執行執行 Ansible 手冊。將每個*########*取代為您自己的資訊。命令 (A) 指定 GitHub 作為來源類型。命令 (B) 指定 Amazon Simple Storage Service (Amazon S3) 作為來源類型。

   (A) GitHub 來源

   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"
```

請見此處範例。

```
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 來源

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"
```

請見此處範例。

```
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 關聯不支援所有 Cron 和 Rate 表達式。如需針對關聯建立 Cron 和 Rate 運算式的詳細資訊，請參閱參考：Systems Manager 的 Cron 和 Rate 運算式。

系統會嘗試在節點上建立關聯，並立即套用狀態。

3. 執行以下命令來檢視您剛剛所建立的關聯的更新狀態。

```
aws ssm describe-association --association-id "ID"
```

建立執行 Chef 配方的關聯

您可以使用 `AWS-ApplyChefRecipes` SSM 文件建立執行Chef配方的State Manager關聯。 State Manager 是 中的工具 AWS Systems Manager。您可以使用 `AWS-ApplyChefRecipes` SSM 文件，將 Linux 型 Systems Manager 受管節點設定為目標。本文件提供執行 Chef 配方的下列優點：

- 支援多個 Chef 版本 (Chef 11 到 Chef 18)。

- 自動在目標節點上安裝 Chef 用戶端軟體。
- 在目標節點上選用地執行 Systems Manager 合規檢查，並將合規檢查的結果存放在 Amazon Simple Storage Service (Amazon S3) 儲存貯體中。
- 在單一文件執行中執行多個食譜和配方。
- 選用地在 `why-run` 模式下執行配方，以顯示哪些配方會對目標節點進行變更，而無須實際進行變更。
- 選用地將自訂 JSON 屬性套用到 `chef-client` 執行。
- 選擇性地從存放在指定位置的來源檔案套用自訂 JSON 屬性。

您可以使用 Git、GitHub、HTTP 或 Amazon S3 儲存貯體作為您在 `AWS-ApplyChefRecipes` 文件中所指定 Chef 碼元書和配方的下載來源。

> **Note**
>
> 執行 Chef 配方的關聯在 macOS 上不受支援。

開始使用

在您建立 `AWS-ApplyChefRecipes` 文件前，請先準備您的 Chef 碼元書和技術指南儲存庫。如果您還沒有要使用的Chef技術指南，您可以使用 AWS 為您準備的測試`HelloWorld`技術指南開始。`AWS-ApplyChefRecipes` 文件根據預設已指向此食譜。您的食譜應根據以下目錄結構進行相似的設定。在以下範例中，`jenkins` 和 `nginx` 是 Chef 碼元書範例，可在 Chef 網站上的 Chef Supermarket 中取得。

雖然 AWS 無法正式支援Chef Supermarket網站上的技術指南，但其中許多都使用 `AWS-ApplyChefRecipes` 文件。以下是在您測試社群技術指南時應確定的條件範例：

- 技術指南應支援您設為目標的 Systems Manager 受管節點的 Linux 型作業系統。
- 此碼元書針對您使用的 Chef 用戶端版本 (Chef 11 到 Chef 18) 應為有效。
- 碼元書與 Chef Infra Client 相容，並且不需要 Chef 伺服器。

驗證您可以連接到 `Chef.io` 網站，使得您在執行清單中指定的任何碼元書都能在 Systems Manager 文件 (SSM 文件) 執行時進行安裝。支援使用巢狀 `cookbooks` 資料夾，但並非必要；您可以將食譜直接存放在根層級下。

```
<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**
>
> 請注意，在建立執行 Chef 配方的 State Manager 關聯前，除非您將 Chef 用戶端版本的值設為 None，否則執行會在 Systems Manager 受管節點上安裝 Chef 用戶端軟體。此操作會使用 Chef 的安裝指令碼來代表您安裝 Chef 元件。在您執行 AWS-ApplyChefRecipes 文件前，請確認您的企業符合任何適用的法規需求，包括適用於使用 Chef 軟體的授權條款。如需詳細資訊，請參閱 Chef 網站。

Systems Manager 可以將合規報告交付至 S3 儲存貯體、Systems Manager 主控台，或提供合規結果來回應 Systems Manager API 命令。若要執行 Systems Manager 合規報告，連接至 Systems Manager 受管節點的執行個體設定檔必須具備寫入 S3 儲存貯體的許可。執行個體設定檔必須具備使用 Systems Manager PutComplianceItem API 的許可。如需有關 Systems Manager 合規的詳細資訊，請參閱 AWS Systems Manager合規。

將文件執行記錄於日誌

當您透過使用 State Manager 關聯來執行 Systems Manager 文件 (SSM 文件) 時，您可以設定關聯來選擇執行文件的輸出，並且可以將輸出傳送到 Amazon Simple Storage Service (Amazon S3) 或 Amazon CloudWatch Logs (CloudWatch Logs)。為了協助讓關聯完成執行時的故障診斷更為容易，請驗證關聯已設為將命令輸出寫入 Amazon Simple Storage Service (Amazon S3) 儲存貯體或 CloudWatch Logs。如需詳細資訊，請參閱在 Systems Manager 中使用關聯。

執行配方時將 JSON 屬性套用至目標

您可以為 Chef 用戶端指定 JSON 屬性，以便在關聯執行期間將其套用至目標節點。設定關聯時，您可以提供原始 JSON 或提供存放在 Amazon S3 中之 JSON 檔案的路徑。

當您想要自訂配方的執行方式而不修改配方本身時，您可以使用 JSON 屬性，例如：

- 覆寫少量的屬性

使用自訂 JSON 以避免為適應細微差異而必須維護多個版本的配方。

- 提供變數值

  使用自訂 JSON 指定可能會在不同次執行中變更的值。例如，如果您的 Chef 碼元書設定接受付款的第三方應用程式，您可以使用自訂 JSON 來指定付款端點 URL。

在原始 JSON 中指定屬性

以下是您可以用來為 Chef 配方指定自訂 JSON 屬性的格式範例。

```
{"filepath":"/tmp/example.txt", "content":"Hello, World!"}
```

指定 JSON 檔案的路徑

以下是您可以用來為 Chef 配方指定自訂 JSON 屬性之路徑的格式範例。

```
{"sourceType":"s3", "sourceInfo":"someS3URL1"}, {"sourceType":"s3",
 "sourceInfo":"someS3URL2"}
```

使用 Git 做為食譜來源

AWS-ApplyChefRecipes 文件會使用 aws:downloadContent 外掛程式來下載 Chef 碼元書。若要從 Git 中下載內容，請如以下範例所示指定 JSON 格式的 Git 儲存庫相關資訊。將每個 *example-resource-placeholder* 替換為您自己的資訊。

```
{
   "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 做為逐步指南來源

AWS-ApplyChefRecipes 文件會使用 aws:downloadContent 外掛程式來下載碼元書。若要從 GitHub 下載內容，請如以下範例所示指定 JSON 格式的 GitHub 儲存庫相關資訊。將每個 *example-resource-placeholder* 替換為您自己的資訊。

```
{
   "owner":"TestUser",
   "repository":"GitHubCookbookRepository",
   "path":"cookbooks/HelloWorld",
   "getOptions":"branch:refs/head/main",
   "tokenInfo":"{{ssm-secure:token-secure-string-parameter}}"
}
```

使用 HTTP 做為食譜來源

您可以將 Chef 碼元書以單一 .zip 或 tar.gz 檔案或目錄結構的形式，儲存在一個自訂 HTTP 位置。若要從 HTTP 下載內容，請如以下範例所示以 JSON 格式指定相關檔案或目錄的路徑。將每個 *example-resource-placeholder* 替換為您自己的資訊。

```
{
   "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 Simple Storage Service (Amazon S3) 作為技術指南來源

您也能以單一 .zip 或 tar.gz 檔案或目錄結構的形式，在 Amazon S3 中儲存和下載 Chef 碼元書。若要從 Amazon S3 下載內容，請如以下範例所示以 JSON 格式指定相關檔案的路徑。將每個 *example-resource-placeholder* 替換為您自己的資訊。

範例 1：下載特定食譜

```
{
   "path":"https://s3.amazonaws.com/chef-cookbooks/HelloWorld.zip"
}
```

範例 2：下載目錄的內容

```
{
   "path":"https://s3.amazonaws.com/chef-cookbooks-test/HelloWorld"
}
```

> **Important**
>
> 如果您指定 Amazon S3，則受管節點上的 AWS Identity and Access Management (IAM) 執行個體設定檔必須使用 `AmazonS3ReadOnlyAccess`政策設定。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可。

建立執行 Chef 配方的關聯 (主控台)

以下程序說明如何使用 Systems Manager 主控台來建立使用 `AWS-ApplyChefRecipes` 文件執行 Chef 碼元書的 State Manager 關聯。

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 State Manager。
3. 選擇 State Manager，然後選擇 Create association (建立關聯)。
4. 針對 Name (名稱)，輸入可協助您記住關聯用途的名稱。
5. 在 Document (文件) 清單中，請選擇 **AWS-ApplyChefRecipes**。
6. 在參數中，為來源類型選取 Git、GitHub、HTTP 或 S3。
7. 對於來源資訊，請使用您在步驟 6 中選取的來源類型對應的適當格式，輸入食譜來源資訊。如需詳細資訊，請參閱下列主題：
   - the section called “使用 Git 做為食譜來源”
   - the section called “使用 GitHub 做為逐步指南來源”
   - the section called “使用 HTTP 做為食譜來源”
   - the section called “使用 Amazon Simple Storage Service (Amazon S3) 作為技術指南來源”
8. 在 Run list (執行清單) 中，以以下格式列出您希望執行的配方，並以逗號分隔每個配方，如下所示。請不要在逗號後方包含空格。將每個 *`example-resource-placeholder`* 替換為您自己的資訊。

   ```
   recipe[cookbook-name1::recipe-name],recipe[cookbook-name2::recipe-name]
   ```

9. (選用) 指定您希望 Chef 用戶端傳遞至目標節點的自訂 JSON 屬性。
   a. 在 JSON 屬性內容中，新增任何您希望 Chef 用戶端傳遞至目標節點的屬性。
   b. 在 JSON 屬性來源中，新增任何您希望 Chef 用戶端傳遞至目標節點之屬性的路徑。

如需詳細資訊，請參閱the section called “執行配方時將 JSON 屬性套用至目標”。

10. 針對 Chef 用戶端版本，指定 Chef 版本。有效值為 11 到 18 中的一個或 None。若指定 11 至 18 的一個數字 (含首尾兩個值)，Systems Manager 會在目標節點上安裝正確的 Chef 用戶端版本。如果指定 None，在目標節點上安裝 Chef 用戶端前，Systems Manager 不會在執行文件的配方。
11. (選用) 對於 Chef 用戶端引數，指定您正在使用 Chef 版本支援的額外引數。如要進一步了解支援的引數，請在執行 Chef 用戶端的節點上執行 `chef-client -h`。
12. (選用) 開啟 Why-run 來顯示若執行配方將對目標節點進行的變更，而無須實際變更目標節點。
13. 對於 Compliance severity (合規嚴重性)，選擇您希望報告的 Systems Manager 合規結果嚴重性。合規報告會指出關聯狀態合規與否，以及您指定的嚴重性層級。合規報告會存放在您指定為 Compliance report bucket (合規報告儲存貯體) 參數 (步驟 14) 的 S3 儲存貯體中。如需合規的詳細資訊，請參閱本指南中的 了解有關合規性的詳細資訊。

    合規會掃描您 Chef 配方中所指定組態與節點資源組態間的量值漂移。有效值為 `Critical`、`High`、`Medium`、`Low`、`Informational`、`Unspecified` 或 None。如要跳過合規報告，請選擇 None。
14. 針對 Compliance type (合規類型)，請指定您希望結果報告的合規類型。有效值為 `Association` (對於 State Manager 關聯)，或是 `Custom:`*`custom_type`*。預設值為 `Custom:Chef`。
15. 對於合規報告儲存貯體，輸入儲存此文件執行的每個 Chef 執行相關資訊的 S3 儲存貯體，包括資源組態與合規結果。
16. 在 Rate control (速率控制) 區段中，設定在受管節點機群之間執行State Manager關聯的選項。如需使用速率控制的詳細資訊，請參閱 了解 State Manager 關聯中的目標和速率控制。

    在 Concurrency (並行) 中，選擇一個選項：

    - 選擇 targets (目標)，輸入可以同時執行關聯的目標絕對數量。
    - 選擇 percentage (百分比)，輸入可以同時執行關聯的目標集百分比。

    在 Error Threshold (錯誤閾值) 中，選擇一個選項：

    - 選擇 errors (錯誤)，輸入 State Manager 停止在額外目標執行關聯之前允許的錯誤絕對數量。
    - 選擇 percentage (百分比)，輸入 State Manager 停止在額外目標執行關聯之前允許的錯誤百分比。
17. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取啟用將輸出寫入 S3 方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

> **Note**
>
> 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給受管節點之執行個體設定檔的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確認與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

18. 選擇 Create Association (建立關聯)。

建立執行 Chef 配方的關聯 (CLI)

下列程序說明如何使用 AWS Command Line Interface (AWS CLI) 建立State Manager關聯，以使用 `AWS-ApplyChefRecipes` 文件執行 Chef 技術指南。

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

2. 執行以下其中一個 命令，建立在具有特定標籤的目標節點上執行 Chef 碼元書的關聯。使用適用於您的食譜來源類型和作業系統的命令。將每個 *example-resource-placeholder* 替換為您自己的資訊。

   a. Git 來源

      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 來源

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"
```

請見此處範例。

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 來源

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 來源

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"
```

請見此處範例。

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 *)"
```

系統會建立關聯，除非您指定的 cron 或 rate 表達式阻止關聯，否則系統會在目標節點上執行關聯。

> **Note**
>
> State Manager 關聯不支援所有 Cron 和 Rate 表達式。如需針對關聯建立 Cron 和 Rate 運算式的詳細資訊，請參閱參考：Systems Manager 的 Cron 和 Rate 運算式。

3. 執行以下命令來檢視您剛剛建立的關聯的狀態。

```
aws ssm describe-association --association-id "ID"
```

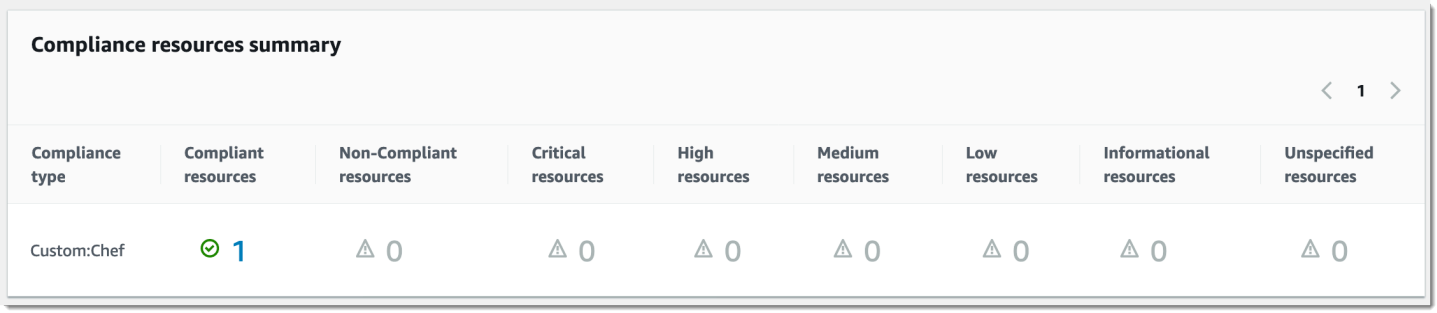
檢視 Chef 資源合規詳細資訊

Systems Manager 在您執行 AWS-ApplyChefRecipes 文件時指定的 Amazon S3 合規報告儲存貯體值中擷取有關 Chef 受管資源的合規資訊。在 S3 儲存貯體中搜尋 Chef 資源故障相關資訊可能非常耗時。您可以改為在 Systems Manager Compliance (合規) 頁面中檢視此資訊。

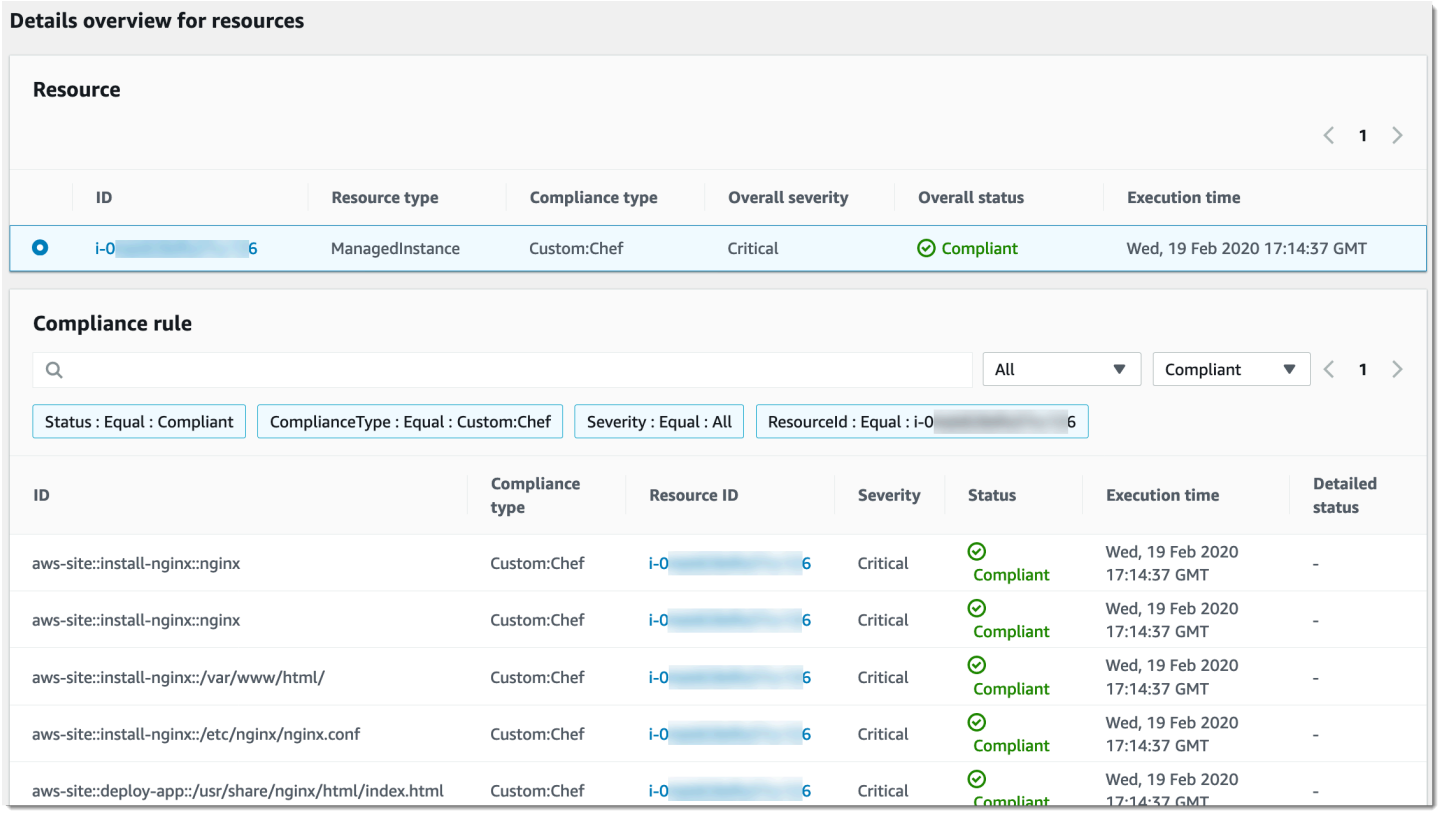
Systems Manager 合規掃描會收集您受管節點上於最近一次 Chef 執行中建立或檢查的資源相關資訊。這些資源可包含檔案、目錄、systemd 服務、yum 套件、範本化檔案、gem 套件，以及相依食譜等。

合規資源摘要區段顯示失敗的資源計數。在以下範例中，ComplianceType 是 Custom:Chef 和不合規的一個資源。

> **Note**
>
> Custom:Chef 是 AWS-ApplyChefRecipes 文件中的預設 ComplianceType 值。這個值是可自訂的。



資源的詳細資訊概觀區段會顯示 AWS 不符合 的資源相關資訊。此區段也會包含執行合規的目標 Chef 資源類型、問題嚴重性、合規狀態，以及詳細資訊的連結 (若適用的話)。



View output (檢視輸出) 會顯示詳細狀態的最後 4,000 個字元。Systems Manager 會使用例外狀況做為第一個元素開始，尋找詳細訊息，並在到達 4,000 個字元配額之前顯示它們。這個程序會顯示拋出例外狀況之前的輸出日誌訊息，也就是與故障診斷最相關的訊息。

如需如何檢視合規資訊的相關資訊，請參閱 AWS Systems Manager合規。

> **Important**
>
> 若State Manager關聯失敗，則不會報告任何合規資料。例如，若 Systems Manager 嘗試從節點沒有存取許可的 S3 儲存貯體下載 Chef 碼元書，則關聯會失敗，Systems Manager 也不會報告任何合規資料。

逐步解說：SSM Agent使用 自動更新 AWS CLI

下列程序會逐步解說如何使用 AWS Command Line Interface建立 State Manager 關聯。關聯會根據您指定的排程自動更新 SSM Agent。如需有關 SSM Agent 的詳細資訊，請參閱「使用 SSM Agent」。若要使用主控台自訂 SSM Agent 的更新排程，請參閱 自動更新 SSM Agent。

若要接收有關 SSM Agent 更新的通知，請訂閱 GitHub 上的 SSM Agent 版本備註頁面。

開始之前

在您完成以下程序前，請先驗證您至少有一個已為 Systems Manager 設定的 Linux、macOS 或 Windows Server 的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體在執行中。如需詳細資訊，請參閱設定 AWS Systems Manager的受管節點。

如果您使用 AWS CLI 或 建立關聯 AWS Tools for Windows PowerShell，請使用 `--Targets` 參數來鎖定執行個體，如下列範例所示。請勿使用 `--InstanceID` 參數。`--InstanceID` 參數是舊參數。

建立自動更新 SSM Agent的關聯

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。
2. 執行以下命令，透過使用 Amazon Elastic Compute Cloud (Amazon EC2) 標籤將執行個體設為目標，從而建立關聯。將每個*########*取代為您自己的資訊。`Schedule` (排程) 參數排定在每週日上午 2:00 執行關聯。(UTC)。

   State Manager 關聯不支援所有 Cron 和 Rate 表達式。如需針對關聯建立 Cron 和 Rate 運算式的詳細資訊，請參閱參考：Systems Manager 的 Cron 和 Rate 運算式。

   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 *)"
   ```

   您可以使用逗號分隔清單來指定執行個體 ID，從而將多個執行個體設為目標。

   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 *)"
```

您可以指定您想要更新到的 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
```

系統會傳回相關資訊，如下所示。

```
{
    "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"
            }
        ]
    }
}
```

系統會嘗試在執行個體上建立關聯，並在建立後套用狀態。關聯狀態會顯示為 Pending (待定)。

3. 執行以下命令來檢視您建立的關聯的更新狀態。

```
aws ssm list-associations
```

如果執行個體執行的「不是」最新版 SSM Agent，則狀態會顯示為 Failed。新版 SSM Agent發佈時，關聯會自動安裝新的代理程式，此時狀態會顯示為 Success (成功)。

演練：在 Windows Server 的 EC2 執行個體自動更新 PV 驅動程式

Amazon Windows Amazon Machine Images (AMIs) 包含一組驅動程式，可用來許可存取虛擬化硬體。Amazon Elastic Compute Cloud (Amazon EC2) 會使用這些驅動程式將執行個體存放區和 Amazon Elastic Block Store (Amazon EBS) 磁碟區映射到其裝置。我們建議您安裝最新的驅動程式，以改善 Windows Server EC2 執行個體的穩定性和效能。如需有關 PV 驅動程式的詳細資訊，請參閱 AWS PV 驅動程式。

下列逐步解說說明如何設定State Manager關聯，以便在驅動程式可用時自動下載和安裝新的 AWS PV 驅動程式。 State Manager 是 中的工具 AWS Systems Manager。

開始之前

在您完成以下程序前，請先驗證您至少有一個已為 Systems Manager 設定的 Windows Server 的 Amazon EC2 執行個體在執行中。如需詳細資訊，請參閱設定 AWS Systems Manager的受管節點。

建立自動更新 PV 驅動程式的State Manager關聯

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 State Manager。
3. 選擇 Create association (建立關聯)。
4. 在名稱欄位中，輸入關聯的描述性名稱。
5. 在 Document (文件) 清單中，請選擇 `AWS-ConfigureAWSPackage`。
6. 在參數區域中，執行以下操作：

   - 針對 Action (動作)，選擇 Install (安裝)。
   - 對於 Installation type (安裝類型)，選擇 Uninstall and reinstall (解除安裝並重新安裝)。

     > **Note**
     >
     > 此套件不支援就地升級。必須解除安裝再重新安裝此套件。

   - 對於名稱，輸入 **`AWSPVDriver`**。

   您不需要為版本和其他引數輸入任何內容。
7. 在目標區段中，透過手動指定標籤、選取執行個體或邊緣裝置，或指定資源群組，選擇您要執行這項操作的受管節點。

   > **Tip**
   >
   > 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

   > **Note**
   >
   > 如果您選擇使用標籤將執行個體設為目標，且指定映射到 Linux 執行個體的標籤，則關聯會在 Windows 執行個體上執行成功，但在 Linux 執行個體上執行失敗。關聯的整體狀態會顯示為 Failed (失敗)。

8. 在指定排程區域中，選擇是否依設定的排程執行關聯，還是只執行一次。更新的 PV 驅動程式一年會發行幾次，所以您可以選擇排程每月執行一次關聯。

9. 在進階選項區域中，在合規嚴重性欄位中選擇關聯的嚴重性等級。合規報告會指出關聯狀態合規與否，以及您在這裡指示的嚴重性等級。如需詳細資訊，請參閱關於State Manager關聯合規。
10. 對於速率控制：
    - 在並行中，指定可同時執行命令的受管節點數目或百分比。

      > **Note**
      >
      > 如果透過指定套用至受管節點的標籤或指定 AWS 資源群組選取了目標，且您不確定會以多少個受管節點為目標，則透過指定百分比限制可以同時執行文件之目標的數量。

    - 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比之節點上的命令失敗之後，停止在其他受管節點上執行命令。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止傳送命令。仍在處理命令的受管節點也可能會傳送錯誤。
11. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取啟用將輸出寫入 S3 方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

    > **Note**
    >
    > 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給受管節點之執行個體設定檔的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確認與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

12. (選用) 在 CloudWatch 警示區段中，在警示名稱欄位中選擇要套用至關聯來進行監控的 CloudWatch 警示。

    > **Note**
    >
    > 記下有關此步驟的以下資訊。
    >
    > - 警示清單最多顯示 100 個警示。如果您在清單中看不到您的警示，請使用 AWS Command Line Interface 建立關聯。如需詳細資訊，請參閱建立關聯 (命令列)。
    > - 若要將 CloudWatch 警示附加至您的命令，建立關聯的 IAM 主體必須具備 `iam:createServiceLinkedRole` 動作的許可。如需有關 CloudWatch 警示的詳細資訊，請參閱使用 Amazon CloudWatch 警示。

- 如果您的警示啟用，則不會執行任何待處理命令叫用或自動化。

13. 選擇 Create association (建立關聯)，接著選擇 Close (關閉)。系統會嘗試在執行個體上建立關聯，並立即套用狀態。

    如果您在一個或多個 Windows Server 的 Amazon EC2 執行個體上建立關聯，則狀態會變更為 Success (成功)。如果未對 Systems Manager 設定執行個體，或者您不小心將 Linux 執行個體設為目標，則狀態會顯示為 Failed (失敗)。

    如果狀態為 Failed (失敗)，請選擇關聯 ID，選擇 Resources (資源) 索引標籤，然後確認已在 Windows Server 的 EC2 執行個體上成功建立關聯。如果 的 EC2 執行個體Windows Server 顯示失敗狀態，請確認 SSM Agent 是否在執行個體上執行，並確認執行個體已設定 Systems Manager 的 AWS Identity and Access Management (IAM) 角色。如需詳細資訊，請參閱為組織設定 Systems Manager 統一主控台。

AWS Systems Manager 變更管理工具

AWS Systems Manager 提供下列工具來變更您的 AWS 資源。

主題

- AWS Systems Manager 自動化
- AWS Systems Manager Change Calendar
- AWS Systems Manager Change Manager
- AWS Systems Manager Documents
- AWS Systems Manager Maintenance Windows
- AWS Systems Manager Quick Setup

AWS Systems Manager 自動化

自動化是 中的工具 AWS Systems Manager，可簡化常見的維護、部署和修復任務， AWS 服務 例如 Amazon Elastic Compute Cloud (Amazon EC2)、Amazon Relational Database Service (Amazon RDS)、Amazon Redshift、Amazon Simple Storage Service (Amazon S3) 等。若要開始使用自動化，請開啟 Systems Manager 主控台。在導覽窗格中，選擇 Automation (自動化)。

Automation 可幫助您建置用於大規模部署、設定和管理 AWS 資源的自動化解決方案。您可以藉助 Automation 精密控制自動化的並行。這意味著您可以指定要同時視為目標的資源數量，以及在停止自動化之前容許發生的錯誤數量。

為了協助您開始使用自動化， AWS 開發和維護數個預先定義的 Runbook。根據自己的使用案例，您可以使用這些預先定義的 Runbook 來執行各種任務，也可以建立自訂 Runbook 來滿足您的需求。若要監控自動化的進度和狀態，可以使用 Systems Manager Automation 主控台或偏好的命令列工具。Automation 還與 Amazon EventBridge 整合，可幫助您大規模建置事件驅動型架構。

Automation 對我組織有何好處？

Automation 提供這些好處：

- Runbook 內容中的指令碼支援

  您可以使用 `aws:executeScript` 動作，直接從 Runbook 執行自訂 Python 和 PowerShell 函數。這為您提供了更大的靈活性來建立自訂 Runbook，因為您可以完成其他 Automation 動作不支援的各種任務。您還可以更好地控制 Runbook 的邏輯。如需如何使用此動作以及如何幫助改進現有自動化解決方案的範例，請參閱 撰寫 Automation Runbook。

- AWS 區域 從集中位置跨多個 AWS 帳戶 和 執行自動化

  管理員可以從 Systems Manager 主控台對多個帳戶和區域的資源執行自動化。

- 強化操作安全性

  管理員有一個集中位置來授予和撤銷 Runbook 的存取權。使用 only AWS Identity and Access Management (IAM) 政策，您可以控制組織中哪些個別使用者或群組可以使用自動化，以及他們可以存取哪些 Runbook。

- 自動化 IT 常見任務

  自動化常見任務有助於提高運營效率、強制執行組織標準以及減少操作員錯誤。例如，您可以使用 `AWS-UpdateCloudFormationStackWithApproval` Runbook 來更新使用 AWS CloudFormation 範本部署的資源。更新會套用新的範本。您可以設定自動化以請求一個或多個 使用者在更新開始之前核准。

- 安全執行大量破壞性工作

  Automation 包括速率控制等功能，這些功能允許您藉由指定並行值和錯誤閾值來控制自動化在機群中的部署。如需有關使用速率控制功能的詳細資訊，請參閱 大規模執行自動化操作。

- 簡化複雜任務

Automation 提供了預先定義的 Runbook，可簡化複雜而耗時的任務，例如建立黃金 Amazon Machine Images (AMIs)。例如，您可以使用 `AWS-UpdateLinuxAmi` 和 `AWS-UpdateWindowsAmi` Runbook 從來源 AMI 建立黃金 AMIs。使用這些 Runbook，您可以在更新套用前後執行自訂指令碼。您也可以包含或排除安裝特定軟體套件。如需執行這些 Runbook 的範例，請參閱 教學課程。

- 定義輸入限制條件

  您可以在自訂 Runbook 中定義限制條件，從而限制 Automation 要接受的特定輸入參數的值。例如：`allowedPattern` 將僅接受與您定義的規則運算式相符之輸入參數的值。如果在輸入參數中指定 `allowedValues`，則系統只接受您在 Runbook 中指定的值。

- 在 Amazon CloudWatch Logs 中記錄自動化動作的輸出

  為了滿足組織中的操作或安全需求，您可能需要提供在 Runbook 執行期間的指令碼記錄。您可使用 CloudWatch Logs 從各種 AWS 服務中監控、存放及存取日誌檔案。您可以將 `aws:executeScript` 動作中的輸出傳送到 CloudWatch Logs 日誌群組，以便進行偵錯和故障診斷。日誌資料可以傳送到您的日誌群組，無論是否使用 KMS 金鑰進行 AWS KMS 加密。如需詳細資訊，請參閱使用 CloudWatch Logs 記錄自動化動作輸出。

- Amazon EventBridge 整合

  Automation 作為 Amazon EventBridge 規則中的目標類型受到支援。這意味著您可以使用事件觸發 Runbook。如需詳細資訊，請參閱 使用 Amazon EventBridge 監控 Systems Manager 及 參考：Systems Manager 的 Amazon EventBridge 事件模式和類型。

- 共用組織最佳實務

  您可以在跨帳戶和區域共用的 Runbook 中定義資源管理、操作任務等項目的最佳實務。

誰應該使用 Automation？

- 任何想要大規模改善營運效率、減少與手動介入相關的錯誤，以及縮短解決常見問題的時間 AWS 的客戶。
- 希望自動化部署和組態任務的基礎設施專家。
- 希望可靠地解決常見問題、提高疑難排解效率和減少重複性操作的管理員。
- 希望自動化通常手動執行之任務的使用者。

什麼是自動化？

自動化包含在 Runbook 中定義並由 Automation 服務執行的所有任務。Automation 使用下列元件來執行自動化。

概念	詳細資訊
Automation Runbook	Systems Manager Automation Runbook 會定義自動化 (Systems Manager 在受管節點 AWS 和資源上執行的動作）。自動化包含數個預先定義的 Runbook，供您用來執行常見任務，像是重新啟動一個或多個 Amazon EC2 執行個體，或建立 Amazon Machine Image (AMI)。您也可以建立自己的 Runbook。Runbook 使用 YAML 或 JSON，並包含您指定的步驟與參數。步驟會循序執行。如需詳細資訊，請參閱建立您自己的執行手冊。 Runbook 是類型 `Automation` 的 Systems Manager 文件，而不是 `Command`、`Policy`、`Session` 文件)。Runbook 支援結構描述版本 0.3。命令文件使用結構描述版本 1.2、2.0 或 2.2。政策文件使用結構描述版本 2.0 或更新版本。
自動化動作	Runbook 中定義的自動化包含一個或多個步驟。每個步驟皆與一個特定動作關聯。動作會決定輸入、行為和步驟的輸出。Runbook 的 `mainSteps` 章節中會定義步驟。自動化支援 20 個不同的動作類型。如需更多資訊，請參閱Systems Manager Automation 動作參考。
自動化配額	每個 AWS 帳戶 可以同時執行 100 個自動化。這包括子系自動化 (由另一個自動化啟動的自動化)，以及速率控制自動化。如果您嘗試執行超過此數量的自動化，Systems Manager 會將額外自動化新增至佇列並顯示待定狀態。此配額可以使用適應性並行來調整。如需詳細資訊，請參

概念	詳細資訊
	閱 允許 Automation 適應並行需求。如需有關執行自動化的詳細資訊，請參閱 執行由 Systems Manager Automation 提供支援的自動化操作。
自動化佇列配額	如果您嘗試執行的自動化超過並行自動化限制，則後續的自動化會新增至佇列。每個 AWS 帳戶可將 5,000 個自動化排入佇列。自動化完成時 (或達到結束狀態)，佇列中的第一個自動化就會啟動。
速率控制自動化配額	每個 AWS 帳戶 可以同時執行 25 個速率控制自動化。如果您嘗試執行的速率控制自動化超過並行速率控制自動化限制，則 Systems Manager 會將後續速率控制自動化新增至佇列並顯示「待處理」狀態。如需執行速率控制自動化的詳細資訊，請參閱 大規模執行自動化操作。
速率控制自動化佇列配額	如果您嘗試執行的自動化超過並行速率控制自動化限制，則後續的自動化會新增至佇列。每個 AWS 帳戶 都可以將 1，000 個速率控制自動化排入佇列。自動化完成時 (或達到結束狀態)，佇列中的第一個自動化就會啟動。

主題

- 設定自動化
- 執行由 Systems Manager Automation 提供支援的自動化操作
- 執行需要核准的自動化
- 大規模執行自動化操作
- 在多個 AWS 區域 和 帳戶中執行自動化
- 根據 EventBridge 事件執行自動化
- 逐步執行自動化
- 使用 State Manager 關聯排程自動化
- 使用維護時段排定自動化

- Systems Manager Automation 動作參考
- 建立您自己的執行手冊
- Systems Manager Automation Runbook 參考
- 教學課程
- 了解 Systems Manager Automation 傳回的狀態
- 故障診斷 Systems Manager Automation

設定自動化

若要在 中設定 Automation 工具 AWS Systems Manager，您必須驗證使用者對 Automation 服務的存取權，並以情境方式設定角色，讓服務可以對您的資源執行動作。我們也建議您在 Automation 偏好設定中選擇使用自適應並行模式。自適應並行會自動擴展自動化配額來滿足您的需求。如需詳細資訊，請參閱允許 Automation 適應並行需求。

為確保正確存取 AWS Systems Manager 自動化，請檢閱下列使用者和服務角色需求。

驗證 Runbook 的使用者存取權

驗證您是否有使用 Runbook 的許可。如果使用者、群組或角色獲指派管理員許可，則您可以存取 Systems Manager Automation。如果您沒有管理員許可，則管理員必須指派 `AmazonSSMFullAccess` 受管政策或提供相當許可的政策給使用者、群組或角色，藉此給予您許可。

> **Important**
>
> IAM 政策 `AmazonSSMFullAccess` 會授予 Systems Manager 動作的許可。不過，有些 Runbook 需要其他服務的許可，例如文件 `AWS-ReleaseElasticIP`，而這需要 `ec2:ReleaseAddress` 的 IAM 許可。因此，您必須檢閱執行手冊中採取的動作，以確保使用者、群組或角色獲指派必要許可，可執行執行手冊中包含的動作。

設定自動化的服務角色 (擔任角色) 存取權

自動化可在服務角色 (或擔任角色) 的內容下啟動。這可讓服務代表您執行動作。如果您未指定擔任角色，自動化會使用呼叫自動化的使用者內容。

然而，以下情況仍需要您為自動化指定服務角色：

- 當您想要限制使用者的資源許可，但您想要使用者執行需要更高許可的自動化時。在此案例中，您可以建立具更高許可的服務角色並允許使用者執行自動化。

- 當您建立執行 Runbook 的 Systems Manager State Manager 關聯。
- 當您有預期會執行超過 12 小時的操作時。
- 當您執行非 Amazon 擁有的 Runbook 時，該 Runbook 會使用 `aws:executeScript`動作來呼叫 AWS API 操作或對 AWS 資源採取行動。如需相關資訊，請參閱 使用 Runbook 的許可。

如果需要為自動化建立服務角色，您可以使用以下其中一個方法。

主題

- 使用 建立自動化的服務角色 AWS CloudFormation
- 使用主控台建立用於自動化的服務角色
- 允許 Automation 適應並行需求
- 實作 Automation 的變更控制

使用 建立自動化的服務角色 AWS CloudFormation

您可以從 AWS CloudFormation 範本建立 Automation 的服務角色 AWS Systems Manager，這是 中的工具。在您建立服務角色之後，您可以使用參數 `AutomationAssumeRole` 在 Runbook 中指定服務角色。

使用 AWS CloudFormation建立服務角色

使用下列程序來建立 Systems Manager Automation 所需的 AWS Identity and Access Management (IAM) 角色 AWS CloudFormation。

建立必要的 IAM 角色

1. 下載並解壓縮 `AWS-SystemsManager-AutomationServiceRole.zip` 檔案。此檔案包含 `AWS-SystemsManager-AutomationServiceRole.yaml` AWS CloudFormation 範本檔案。
2. 在 https：//https://console.aws.amazon.com/cloudformation 開啟 AWS CloudFormation 主控台。
3. 選擇 Create Stack (建立堆疊)。
4. 在 Specify template (指定範本) 區段中，選擇 Upload a template file (上傳範本檔案)。
5. 選擇瀏覽，然後選擇`AWS-SystemsManager-AutomationServiceRole.yaml` AWS CloudFormation 範本檔案。
6. 選擇 Next (下一步)。
7. 在 Specify stack details (指定堆疊詳細資訊) 頁面，於 Stack name (堆疊名稱) 欄位輸入名稱。

8. 在 Configure stack options (設定堆疊選項) 頁面上，您不需要進行任何選取。選擇 Next (下一步)。
9. 在檢閱頁面上，向下捲動並選擇我確認 AWS CloudFormation 可能會建立 IAM 資源選項。
10. 選擇 Create (建立)。

CloudFormation 會顯示 CREATE_IN_PROGRESS 狀態大約三分鐘。在堆疊建立且您的角色可使用之後，狀態會變更為 CREATE_COMPLETE (CREATE_COMPLETE)。

> **Important**
>
> 如果您執行可使用 AWS Identity and Access Management (IAM) 服務角色叫用其他服務的自動化工作流程，請注意您必須為該服務角色設定可叫用這些服務的許可。此要求適用於所有 AWS Automation Runbook (`AWS-*` Runbook)，例如 `AWS-ConfigureS3BucketLogging`、`AWS-CreateDynamoDBBackup` 和 `AWS-RestartEC2Instance` Runbook 等。此要求也適用於您建立的任何自訂自動化執行手冊， AWS 服務 這些執行手冊使用呼叫其他服務的動作來叫用其他 。例如，如果您使用 `aws:executeAwsApi`、`aws:createStack` 或 `aws:copyImage` 動作，為服務角色設定可叫用這些服務的許可。您可以將 IAM 內嵌政策新增至角色， AWS 服務 以將許可授予其他 。如需詳細資訊，請參閱（選用）新增自動化內嵌政策或客戶受管政策以叫用其他政策 AWS 服務。

複製自動化的角色資訊

使用下列程序，從 AWS CloudFormation 主控台複製 Automation 服務角色的相關資訊。使用 Runbook 時，必須指定這些角色。

> **Note**
>
> 如果您執行 `AWS-UpdateLinuxAmi` 或 `AWS-UpdateWindowsAmi` Runbook，則不需要使用此程序複製角色資訊。這些 Runbook 已將所需的角色指定為預設值。這些 Runbook 中指定的角色使用 IAM 受管政策。

複製角色名稱

1. 在 https：//https://console.aws.amazon.com/cloudformation 開啟 AWS CloudFormation 主控台。

2. 選取您在上一個程序中建立的自動化 Stack name (堆疊名稱)。
3. 選擇 Resources (資源) 標籤。
4. 為 AutomationServiceRole 選擇 Physical ID (實體 ID) 連結。IAM 主控台會開啟自動化服務角色的摘要。
5. 複製 Role ARN (角色 ARN) 旁邊的 Amazon Resource Name (ARN)。ARN 的格式類似如下：`arn:aws:iam::12345678:role/AutomationServiceRole`
6. 將 ARN 貼入文字檔案以供日後使用。

您已完成自動化服務角色的設定。您現在可以在 Runbook 中使用 Automation 服務角色 ARN。

使用主控台建立用於自動化的服務角色

如果您需要在 中建立 Automation 的服務角色， AWS Systems Manager請完成下列任務。如需 Automation 需要服務角色之時機的詳細資訊，請參閱 設定自動化。

任務

- 任務 1：建立自動化的服務角色
- 任務 2：將 iam:PassRole 政策連接至自動化角色

任務 1：建立自動化的服務角色

使用下列處理程序以建立 Systems Manager Automation 的服務角色 (或擔任角色)。

> **Note**
>
> 您也可以在 Runbook 中使用此角色，例如 `AWS-CreateManagedLinuxInstance` Runbook。在 Runbook 中使用此角色或 (IAM) 角色的 Amazon Resource Name AWS Identity and Access Management (ARN)，可讓 Automation 在您的環境中執行動作，例如啟動新的執行個體並代表您執行動作。

建立 IAM 角色並允許 Automation 擔任角色

1. 在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。
2. 在導覽窗格中，選擇 Roles (角色)，然後選擇 Create role (建立角色)。
3. 在 Select type of trusted entity (選擇可信任執行個體類型) 下，選擇 AWS service (服務)。

4. 在 Choose a use case (選擇使用案例) 區段中，選擇 Systems Manager (系統管理員)，然後選擇 Next: Permissions (下一步：許可)。
5. 在 Attached permissions policy (已連接許可政策) 頁面，搜尋 AmazonSSMAutomationRole 政策，選取該政策，接著選擇 Next: Review (下一步：檢閱)。
6. 在 Review (檢閱) 頁面，於 Role name (角色名稱) 方塊輸入名稱，接著輸入描述。
7. 選擇 Create role (建立角色)。系統會讓您回到 Roles (角色) 頁面。
8. 在 Roles (角色) 頁面，選擇您剛建立的角色，以開啟 Summary (摘要) 頁面。請記下 Role Name (角色名稱) 和 Role ARN (角色 ARN)。將 iam:PassRole 政策連接至下個程序中的 IAM 帳戶時，您要指定角色 ARN。您也可以在 Runbook 中指定角色名稱和 ARN。

> **Note**
>
> `AmazonSSMAutomationRole` 政策會將自動化角色許可指派給 AWS Lambda 您帳戶中的函數子集。這些函數會以 "Automation" 開頭。如果您打算使用 Automation 搭配 Lambda 函數，Lambda ARN 必須使用以下格式：
> `"arn:aws:lambda:*:*:function:Automation*"`
> 如果您現有的 Lambda 函數之 ARN 不使用此格式，則您還必須將額外的 Lambda 政策連接至您的自動化角色，例如 AWSLambdaRole 政策。額外的政策或角色必須針對 AWS 帳戶帳戶內的 Lambda 函數提供更廣泛的存取。

建立服務角色後，我們建議您編輯信任政策，幫助預防跨服務混淆代理人問題。混淆代理人問題屬於安全性議題，其中沒有執行動作許可的實體可以強制具有更多許可的實體執行該動作。在 中 AWS，跨服務模擬可能會導致混淆代理人問題。在某個服務 (呼叫服務) 呼叫另一個服務 (被呼叫服務) 時，可能會發生跨服務模擬。可以操縱呼叫服務來使用其許可，以其不應有存取許可的方式對其他客戶的資源採取動作。為了防止這種情況， AWS 提供工具，協助您保護所有 服務的資料，讓 服務主體有權存取您帳戶中的資源。

若要限制 Automation 為資源提供另一項服務的許可，我們建議在資源政策中使用 `aws:SourceArn` 和 `aws:SourceAccount` 全域條件內容索引鍵。如果 `aws:SourceArn` 值不包含帳戶 ID (例如 Amazon Simple Storage Service (Amazon S3) 儲存貯體 ARN)，則必須使用這兩個全域條件內容索引鍵來限制許可。如果同時使用這兩個全域條件內容索引鍵，且 `aws:SourceArn` 值包含帳戶 ID，則在相同政策陳述式中使用 `aws:SourceAccount` 值和 `aws:SourceArn` 值中的帳戶時，必須使用相同的帳戶 ID。如果您想要僅允許一個資源與跨服務存取相關聯，則請使用 `aws:SourceArn`。如果您想要允許該帳戶中的任何資源與跨服務使用相關聯，請使用 `aws:SourceAccount`。`aws:SourceArn` 的值必須是自動化執行的 ARN。如果不知道資源的完整 ARN，或者如果您指定了多個資源，請

使用 aws:SourceArn 全域條件內容索引鍵，同時使用萬用字元 (*) 表示 ARN 的未知部分。例如：arn:aws:ssm:*:123456789012:automation-execution/*。

下列範例示範如何使用 Automation 的 aws:SourceArn 和 aws:SourceAccount 全域條件內容索引鍵，來預防混淆代理人問題。

```
{
  "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/*"
        }
      }
    }
  ]
}
```

修改角色信任政策

1. 在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。
2. 在導覽窗格中，選擇 Roles (角色)。
3. 在帳戶的角色清單中選擇 Automation 服務角色的名稱。
4. 選擇 Trust Relationships (信任關係) 標籤，然後選擇 Edit Trust Relationship (編輯信任關係)。
5. 使用 Automation 的 aws:SourceArn 和 aws:SourceAccount 全域條件內容索引鍵來編輯信任政策，幫助預防混淆代理人問題。
6. 若要儲存變更，請選擇 Update Trust Policy (更新信任政策)。

（選用）新增自動化內嵌政策或客戶受管政策以叫用其他政策 AWS 服務

如果您執行 AWS 服務 使用 IAM 服務角色叫用其他 的自動化，則必須設定服務角色以允許叫用這些服務。此要求適用於所有 AWS Automation Runbook (AWS-* Runbook)，例如 AWS-ConfigureS3BucketLogging、 AWS-CreateDynamoDBBackup和 AWS-RestartEC2Instance Runbook，以做為範例。此要求也適用於您所建立會透過呼叫其他服務的動作來叫用其他 AWS 服務 的任何自訂 Runbooks。例如，如果您使用 aws:executeAwsApi、aws:CreateStack 或 aws:copyImage 等動作，則您必須為服務角色設定可叫用這些服務的許可。您可以將 IAM 內嵌政策或客戶受管政策新增至角色， AWS 服務 以將許可授予其他 。

嵌入服務角色的內嵌政策 (IAM 主控台)

1. 登入 AWS Management Console ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇角色。
3. 在清單中，選擇您要編輯的角色名稱。
4. 選擇許可索引標籤標籤。
5. 在新增許可下拉式選單中，選擇連接政策或建立內嵌政策。
6. 如果選擇連接政策，請選取您要新增之政策旁邊的核取方塊，然後選擇新增許可。
7. 如果選擇建立政策，請選擇 JSON 索引標籤。
8. 輸入 AWS 服務 您要叫用 的 JSON 政策文件。以下是兩個 JSON 政策文件範例。

   Amazon Simple Storage Service (Amazon S3) PutObject 和 GetObject 範例

```
{
    "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 範例

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":"ec2:CreateSnapshot",
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":"ec2:DescribeSnapshots",
         "Resource":"*"
      }
   ]
}
```

如需 IAM 政策語言的詳細資訊，請參閱《IAM 使用者指南》中的 IAM JSON 政策參考。

9. 完成時，請選擇 Review policy (檢閱政策)。Policy Validator (政策檢查工具) 會回報任何語法錯誤。
10. 在 Review policy (檢閱政策) 頁面，輸入您所建立政策的 Name (名稱)。檢閱政策 Summary (摘要) 來查看您的政策所授予的許可。然後選擇 Create policy (建立政策) 來儲存您的工作。
11. 在您建立內嵌政策後，它會自動嵌入您的角色中。

任務 2：將 iam:PassRole 政策連接至自動化角色

使用以下程序將 `iam:PassRole` 政策連接至您的自動化服務角色。這可讓自動化服務在執行自動化時將角色傳遞給其他服務或 Systems Manager 工具。

將 iam: PassRole 政策連接至您的自動化角色

1. 在您剛建立的角色之 Summary (摘要) 頁面，選擇 Permissions (許可) 標籤。
2. 選擇 Add inline policy (新增內嵌政策)。
3. 在 Create policy (建立政策) 頁面，選擇 Visual editor (視覺化編輯器) 標籤。
4. 選擇 Service (服務)，接著選擇 IAM (IAM)。
5. 選擇 Select actions (選取動作)。

6. 在 Filter actions (篩選動作) 文字方塊中輸入 **`PassRole`**，接著選擇 PassRole 選項。
7. 選擇資源。確認 Specific (特定) 已選取，接著選擇 Add ARN (新增 ARN)。
8. 在 Specify ARN for role (指定角色的 ARN) 欄位中，貼上您在任務 1 結尾複製的自動化角色 ARN。系統會填入 Account (帳戶) 和 Role name with path (角色名稱與路徑) 欄位。

> **Note**
>
> 如果您想要自動化服務角色將 IAM 執行個體設定檔角色連接到 EC2 執行個體，則必須新增 IAM 執行個體設定檔角色的 ARN。這可讓自動化服務角色傳遞 IAM 執行個體設定檔角色到目標 EC2 執行個體。

9. 選擇 Add (新增)。
10. 選擇 Review policy (檢閱政策)。
11. 在 Review Policy (檢閱政策) 頁面輸入名稱，接著選擇 Create Policy (建立政策)。

允許 Automation 適應並行需求

預設情況下，Automation 允許您一次執行多達 100 個並行自動化。Automation 還提供了一個選用設定，可讓您使用該設定自動調整並行自動化配額。藉由此設定，並行自動化配額最多可容納 500 個並行自動化 (具體數量取決於可用資源)。

> **Note**
>
> 如果自動化呼叫 API 操作，則以適應方式擴展到目標可能會導致調節例外狀況。在啟用了自適應並行的情況下，如果執行自動化時重複出現調節例外狀況，則可能需要請求增加 API 操作的配額 (如果可用)。

使用 開啟適應性並行 AWS Management Console

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Automation (自動化)。
3. 選擇 Preferences (偏好) 標籤，然後選擇 Edit (編輯)。
4. 選取 Enable adaptive concurrency (啟用自適應並行) 旁邊的核取方塊。
5. 選擇 Save (儲存)。

使用命令列開啟適應性並行

- 開啟 AWS CLI 或 Tools for Windows PowerShell，並執行下列命令，以開啟請求區域中您帳戶的適應性並行。

  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"
  ```

實作 Automation 的變更控制

依預設，Automation 會允許您使用沒有日期和時間限制的執行手冊。透過整合自動化與 Change Calendar，您可以將變更控制實作到 中的所有自動化 AWS 帳戶。透過此設定，您帳戶中的 AWS Identity and Access Management (IAM) 主體只能在變更行事曆允許的期間內執行自動化。若要進一步了解 Change Calendar 的使用，請參閱 使用 Change Calendar。

開啟變更控制 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Automation (自動化)。
3. 選擇 Preferences (偏好) 標籤，然後選擇 Edit (編輯)。
4. 選取開啟 Change Calendar 整合旁的核取方塊。
5. 在選擇變更行事曆下拉式清單中，選擇您希望 Automation 追蹤的變更行事曆。

6. 選擇 Save (儲存)。

執行由 Systems Manager Automation 提供支援的自動化操作

執行自動化時，依預設，自動化會在啟動自動化的使用者內容中執行。這表示，如果您的使用者具有管理員許可，則自動化會以管理員許可執行，並能完全存取由自動化設定的資源。做為安全最佳實務，我們建議您在執行自動化時，使用以 AmazonSSMAutomationRole 受管政策設定的 IAM 服務角色，在此案例中稱為擔任角色。您可能需要將其他 IAM 政策新增至您的擔任角色，才能使用各種 Runbook。使用 IAM 服務角色執行自動化稱為委託管理。

在您使用服務角色時，自動化允許對 AWS 資源執行，但執行自動化的使用者限制了對這些資源的存取 (或無存取)。例如，您可以設定服務角色並搭配 Automation 使用，以重新啟動一個或多個 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。自動化是 中的工具 AWS Systems Manager。自動化會重新啟動執行個體，但服務角色不會提供存取這些執行個體的使用者許可。

您可以在執行自動化時指定執行時間的服務角色，或者您可以建立自訂 Runbook 並直接在 Runbook 中指定服務角色。如果您指定服務角色 (在執行時間或在 Runbook 中)，則服務會在指定服務角色的內容中執行。如果您不指定服務角色，則系統會在使用者的內容中建立暫時工作階段並執行自動化。

> **Note**
>
> 針對預期要執行超過 12 小時的自動化，您必須指定服務角色。如果您在使用者的內容中啟動長期執行自動化，使用者的暫時工作階段會在 12 小時後過期。

委派管理可確保您的 AWS 資源有更高層級的安全和控制。它也能夠強化稽核體驗，因為動作是由中央服務角色，而非多個 IAM 帳戶針對您的資源執行的。

開始之前

在完成下列程序之前，您必須建立 IAM 服務角色，並為 中的工具 Automation 設定信任關係 AWS Systems Manager。如需詳細資訊，請參閱任務 1：建立自動化的服務角色。

以下程序說明如何使用 Systems Manager 主控台或您偏好的命令行來執行簡易的自動化。

執行簡易自動化 (主控台)

以下程序說明如何使用 Systems Manager 主控台來執行簡易的自動化。

執行簡易自動化

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Automation (自動化)，接著選擇 Execute automation (執行自動化)。
3. 在 Automation document (自動化文件)清單中，選擇 Runbook。在 Document categories (文件類別) 窗格中選擇一個或多個選項，根據 SSM 文件的用途來進行篩選。若要檢視您擁有的 Runbook，請選擇 Owned by me (我所擁有的) 索引標籤。若要檢視與您帳戶共用的 Runbook，請選擇 Shared with me (與我共用的) 索引標籤。若要檢視所有 Runbook，請選擇 All documents (所有文件) 索引標籤。

   > **Note**
   >
   > 您可以選擇 Runbook 名稱檢視 Runbook 資訊。

4. 在 Document details (文件詳細資訊) 部分，確認 Document version (文件版本) 設定為您想要執行的版本。系統包括以下版本選項：

   - 執行期的預設版本：如果 Automation 執行手冊會定期更新且已指派新的預設版本，則請選擇此選項。
   - 執行期的最新版本：如果 Automation 執行手冊會定期更新，而您想要執行最近更新的版本，請選擇此選項。
   - 1 (預設)：選擇此選項以執行文件的第一個版本，也是預設版本。

5. 選擇 Next (下一步)。
6. 在 Execution Mode (執行模式) 部分，選擇 Simple execution (簡易執行)。
7. 在 Input parameters (輸入參數) 部分，指定所需的輸入。或者，您也可以從 AutomationAssumeRole 清單中選擇 IAM 服務角色。
8. (選用) 選擇要套用至您的自動化以便加以監控的 CloudWatch 警示。若要將 CloudWatch 警示連接至您的自動化，啟動自動化的 IAM 主體必須具備 `iam:createServiceLinkedRole` 動作的許可。如需有關 CloudWatch 警示的詳細資訊，請參閱使用 Amazon CloudWatch 警示。請注意，如果您的警示啟用，則會停止自動化。如果您使用 AWS CloudTrail，您會在追蹤中看到 API 呼叫。
9. 選擇 Execute (執行)。

主控台會顯示自動化的狀態。若自動化無法執行，請參閱 故障診斷 Systems Manager Automation。

執行簡易自動化 (命令列)

下列程序說明如何使用 AWS CLI （在 Linux 或 Windows 上）或 AWS Tools for PowerShell 來執行簡單的自動化。

執行簡易自動化

1. AWS Tools for PowerShell如果您尚未安裝和設定 AWS CLI 或 。

   如需相關資訊，請參閱安裝或更新 AWS CLI的最新版本和安裝 AWS Tools for PowerShell。

2. 執行以下命令來啟動簡易自動化。將每個########取代為您自己的資訊。

   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 Runbook 重新啟動指定 EC2 執行個體的範例。

   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"}
```

系統會傳回相關資訊，如下所示。

Linux & macOS

```
{
    "AutomationExecutionId": "4105a4fc-f944-11e6-9d32-0123456789ab"
}
```

Windows

```
{
    "AutomationExecutionId": "4105a4fc-f944-11e6-9d32-0123456789ab"
}
```

PowerShell

```
4105a4fc-f944-11e6-9d32-0123456789ab
```

3. 執行以下命令來擷取自動化的狀態。

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"}
```

系統會傳回相關資訊，如下所示。

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                     : {}
```

執行需要核准的自動化

下列程序說明如何使用 AWS Systems Manager 主控台和 AWS Command Line Interface (AWS CLI)，使用簡易執行搭配核准來執行自動化。自動化使用自動化動作 `aws:approve`，這會臨時暫停自動化，直到指定的委託人核准或拒絕動作為止。自動化會在目前的使用者內容中執行。這表示只要有使用 Runbook 和 Runbook 所呼叫任何動作的許可，您就不必設定其他的 IAM 許可。如果您在 IAM 中有管理員許可，您便已經有執行此 Runbook 的許可。

開始之前

除了 Runbook 需要的標準輸入，`aws:approve` 動作需要以下兩個參數：

- 核准者清單。核准者清單必須至少包含一個核准者，形式為使用者名稱或使用者 ARN。如果提供多個核准者，必須在 Runbook 指定對應的最低核准計數。
- Amazon Simple Notification Service (Amazon SNS) 主題 ARN。Amazon SNS 主題名稱必須以 `Automation` 開頭。

此程序假設您已經建立一個 Amazon SNS 主題，其為交付核准請求所需。如需資訊，請參閱 Amazon Simple Notification Service 開發人員指南中的建立主題。

以核准者身分執行自動化 (主控台)

以核准者身分執行自動化

以下程序說明如何使用 Systems Manager 主控台以核准者身分執行自動化。

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。

2. 在導覽窗格中，選擇 Automation (自動化)，接著選擇 Execute automation (執行自動化)。
3. 在 Automation document (自動化文件)清單中，選擇 Runbook。在 Document categories (文件類別) 窗格中選擇一個或多個選項，根據 SSM 文件的用途來進行篩選。若要檢視您擁有的 Runbook，請選擇 Owned by me (我所擁有的) 索引標籤。若要檢視與您帳戶共用的 Runbook，請選擇 Shared with me (與我共用的) 索引標籤。若要檢視所有 Runbook，請選擇 All documents (所有文件) 索引標籤。

   > **Note**
   >
   > 您可以選擇 Runbook 名稱檢視 Runbook 資訊。

4. 在 Document details (文件詳細資訊) 部分，確認 Document version (文件版本) 設定為您想要執行的版本。系統包括以下版本選項：

   - 執行期的預設版本：如果 Automation 執行手冊會定期更新且已指派新的預設版本，則請選擇此選項。
   - 執行期的最新版本：如果 Automation 執行手冊會定期更新，而您想要執行最近更新的版本，請選擇此選項。
   - 1 (預設)：選擇此選項以執行文件的第一個版本，也是預設版本。
5. 選擇 Next (下一步)。
6. 在 Execute automation document (執行自動化文件) 頁面上，選擇 Simple execution (簡易執行)。
7. 在 Input Parameters (輸入參數) 部分，指定所需的輸入參數。

   例如，如果您選擇 **`AWS-StartEC2InstanceWithApproval`** Runbook，則必須指定或選擇 InstanceId 參數的執行個體 ID。
8. 在核准者區段，指定自動化動作核准者的使用者名稱或使用者 ARN。
9. 在 SNSTopicARN 部分，指定要用來傳送核准通知的 SNS 主題 ARN。SNS 主題名稱必須以 Automation (自動化) 開頭。
10. 或者，您也可以從 AutomationAssumeRole 清單中選擇 IAM 服務角色。如果以 100 個以上的帳戶和區域為目標，則必須指定 `AWS-SystemsManager-AutomationAdministrationRole`。
11. 選擇 Execute automation (執行自動化)。

指定的核准者將收到 Amazon SNS 通知，通知中含有核准或拒絕自動化的詳細資訊。此核准動作自發出之日起 7 天內有效，並且可以使用 Systems Manager 主控台或 AWS Command Line Interface () 發出AWS CLI。

若您選擇核准自動化，自動化會繼續執行指定 Runbook 中所包含的步驟。主控台會顯示自動化的狀態。若自動化無法執行，請參閱 故障診斷 Systems Manager Automation。

核准或拒絕自動化

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中選擇 Automation (自動化)，然後選取在先前程序中執行的自動化。
3. 選擇 Actions (動作)，然後選擇 Approve/Deny (核准/拒絕)。
4. 選擇 Approve (核准) 或 Deny (拒絕)，並選擇性地提供註解。
5. 選擇提交。

以核准者身分執行自動化 (命令列)

下列程序說明如何使用 AWS CLI （ 在 Linux 或 Windows 上 ） 或 AWS Tools for PowerShell 與核准者執行自動化。

以核准者身分執行自動化

1. AWS Tools for PowerShell如果您尚未安裝和設定 AWS CLI 或 。

   如需相關資訊，請參閱安裝或更新 AWS CLI的最新版本和安裝 AWS Tools for PowerShell。
2. 執行以下命令，以核准者身分執行自動化。將每個########取代為您自己的資訊。在文件名稱區段中，指定包含自動化動作 `aws:approve` 的 Runbook。

   在 `Approvers` 中，指定動作核准者的使用者名稱或使用者 ARN。在 `SNSTopic` 中，指定要用來傳送核准通知的 SNS 主題 ARN。Amazon SNS 主題名稱必須以 `Automation` 開頭。

   > **Note**
   >
   > 核准者參數值的特定名稱和 SNS 主題取決於您所選擇 Runbook 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"
    }
```

系統會傳回相關資訊，如下所示。

Linux & macOS

```
{
    "AutomationExecutionId": "df325c6d-b1b1-4aa0-8003-6cb7338213c6"
}
```

Windows

```
{
    "AutomationExecutionId": "df325c6d-b1b1-4aa0-8003-6cb7338213c6"
}
```

PowerShell

```
df325c6d-b1b1-4aa0-8003-6cb7338213c6
```

核准自動化

- 執行以下命令，核准自動化。將每個########取代為您自己的資訊。

  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"}
  ```

  如果命令成功，則無輸出訊息。

拒絕自動化

- 執行以下命令，拒絕自動化。將每個########取代為您自己的資訊。

  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"}
```

如果命令成功，則無輸出訊息。

大規模執行自動化操作

透過 AWS Systems Manager 自動化，您可以使用目標在 AWS 資源機群上執行自動化。此外，您可以藉由指定並行值和錯誤閾值來控制自動化在機群中的部署。並行和錯誤閾值功能統稱為「速率控制」。並行值會決定允許以多少資源同時執行自動化。Automation 還提供了可以選用的自適應並行模式。自適應並行會自動將自動化配額從 100 個並行執行自動化擴展到 500 個。錯誤閾值會決定在 Systems Manager 停止傳送自動化至其他資源之前允許多少次自動化失敗。

如需並行和錯誤閾值的詳細資訊，請參閱大規模控制自動化。如需目標的詳細資訊，請參閱對應自動化的目標。

下列程序說明如何開啟自適應並行，以及如何借助 Systems Manager 主控台和 AWS Command Line Interface (AWS CLI) 來使用目標和速率控制執行自動化。

以目標和速率控制執行自動化 (主控台)

以下程序說明如何使用 Systems Manager 主控台，以目標和速率控制來執行自動化。

以目標和速率控制執行自動化

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Automation (自動化)，接著選擇 Execute automation (執行自動化)。

3. 在 Automation document (自動化文件)清單中，選擇 Runbook。在 Document categories (文件類別) 窗格中選擇一個或多個選項，根據 SSM 文件的用途來進行篩選。若要檢視您擁有的 Runbook，請選擇 Owned by me (我所擁有的) 索引標籤。若要檢視與您帳戶共用的 Runbook，請選擇 Shared with me (與我共用的) 索引標籤。若要檢視所有 Runbook，請選擇 All documents (所有文件) 索引標籤。

   > **Note**
   >
   > 您可以選擇 Runbook 名稱檢視 Runbook 資訊。

4. 在 Document details (文件詳細資訊) 部分，確認 Document version (文件版本) 設定為您想要執行的版本。系統包括以下版本選項：

   - 執行期的預設版本：如果 Automation 執行手冊會定期更新且已指派新的預設版本，則請選擇此選項。
   - 執行期的最新版本：如果 Automation 執行手冊會定期更新，而您想要執行最近更新的版本，請選擇此選項。
   - 1 (預設)：選擇此選項以執行文件的第一個版本，也是預設版本。

5. 選擇 Next (下一步)。
6. 在 Execution Mode (執行模式) 部分，選擇 Rate Control (速率控制)。如果想要使用目標和速率控制，就必須使用此模式或 Multi-account and Region (多帳戶和區域)。
7. 在目標區段中，選擇您希望如何將資源鎖定在您想要執行自動化 AWS 的位置。這些選項是必要的。

   a. 使用 Parameter (參數) 清單選擇一個參數。Parameter (參數) 清單中的項目，是由您在此程序一開始所選取自動化 Runbook 中的參數決定。藉由選擇參數，您就會定義自動化工作流程執行的資源類型。

   b. 使用 Targets (目標) 清單選擇您想要如何將資源設為目標。

      i. 如果您選擇使用參數值將資源設為目標，則請在 Input parameters (輸入參數) 區段為您所選的參數輸入參數值。

      ii. 如果您選擇使用 來鎖定資源 AWS Resource Groups，請從資源群組清單中選擇群組的名稱。

      iii. 如果您選擇使用標籤將資源設為目標，請在提供的欄位中輸入標籤索引鍵 (選用) 和標籤值。選擇新增。

iv. 如果您想要在目前 的所有執行個體上執行 Automation Runbook AWS 帳戶 AWS 區域，然後選擇所有執行個體。

8. 在 Input parameters (輸入參數) 部分，指定所需的輸入。或者，您也可以從 AutomationAssumeRole 清單中選擇 IAM 服務角色。

> **Note**
>
> 您可能不必選擇 Input parameters (輸入參數) 部分的其中一些選項。這是因為您使用標籤或資源群組將資源設為目標。例如，假設您選擇了 `AWS-RestartEC2Instance` Runbook，就不必在 Input parameters (輸入參數) 部分指定或選擇執行個體 ID。自動化執行會使用您指定的標籤或資源組找出要重新啟動的執行個體。

9. 使用速率控制區段中的選項來限制可在每個帳戶區域對內執行自動化 AWS 的資源數量。

   在 Concurrency (並行) 部分，選擇一個選項：

   - 選擇 targets (目標)，輸入可以同時執行自動化工作流程的目標絕對數量。
   - 選擇 percentage (百分比)，輸入可以同時執行自動化工作流程的目標集百分比。

10. 在 Error threshold (錯誤閾值) 部分，選擇一個選項：

    - 選擇 errors (錯誤)，輸入在自動化停止傳送工作流程至其他資源之前允許的錯誤絕對數量。
    - 選擇 percentage (百分比)，輸入在自動化停止傳送工作流程至其他資源之前允許的錯誤百分比。

11. (選用) 選擇要套用至您的自動化以便加以監控的 CloudWatch 警示。若要將 CloudWatch 警示連接至您的自動化，啟動自動化的 IAM 主體必須具備 `iam:createServiceLinkedRole` 動作的許可。如需有關 CloudWatch 警示的詳細資訊，請參閱使用 Amazon CloudWatch 警示。請注意，如果您的警示啟用，則會停止自動化。如果您使用 AWS CloudTrail，您會在追蹤中看到 API 呼叫。

12. 選擇 Execute (執行)。

若要檢視由您的速率控制自動化啟動的自動化，請在導覽窗格中選擇 Automation，接著選取 Show child automations (顯示子系自動化)。

以目標和速率控制執行自動化 (命令列)

下列程序說明如何使用 AWS CLI （在 Linux 或 Windows 上） 或 AWS Tools for PowerShell 執行具有目標和速率控制的自動化。

以目標和速率控制執行自動化

1. AWS Tools for PowerShell如果您尚未安裝和設定 AWS CLI 或 。

   如需相關資訊，請參閱安裝或更新 AWS CLI的最新版本和安裝 AWS Tools for PowerShell。

2. 執行以下命令來檢視文件清單。

   Linux & macOS

   ```
   aws ssm list-documents
   ```

   Windows

   ```
   aws ssm list-documents
   ```

   PowerShell

   ```
   Get-SSMDocumentList
   ```

   請注意您要使用的 Runbook 名稱。

3. 執行以下命令來檢視您建立的 Runbook 詳細資訊。把 *Runbook ##*取代為您要檢視其詳細資料的 Runbook 名稱。此外，記下您希望用於`--target-parameter-name`選項的參數名稱 (例如 `InstanceId`)。此參數會決定自動化執行的資源類型。

   Linux & macOS

   ```
   aws ssm describe-document \
       --name runbook name
   ```

   Windows

   ```
   aws ssm describe-document ^
       --name runbook name
   ```

   PowerShell

   ```
   Get-SSMDocumentDescription `
   ```

```
    -Name runbook name
```

4. 建立命令以使用您想要執行的目標和速率控制選項。將每個########取代為您自己的資訊。

   「使用標籤設定目標」

   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%"
```

「使用參數值設定目標」

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 鎖定目標 AWS Resource Groups

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%"
```

針對目前 AWS 帳戶 和 中的所有 Amazon EC2 執行個體 AWS 區域

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%"
```

命令會傳回執行 ID。複製此 ID 到剪貼簿。您可以使用此 ID 檢視自動化的狀態。

Linux & macOS

```
{
    "AutomationExecutionId": "a4a3c0e9-7efd-462a-8594-01234EXAMPLE"
}
```

Windows

```
{
    "AutomationExecutionId": "a4a3c0e9-7efd-462a-8594-01234EXAMPLE"
}
```

PowerShell

```
a4a3c0e9-7efd-462a-8594-01234EXAMPLE
```

5. 執行以下命令檢視自動化。把每個*##### ID* 取代為您自己的資訊。

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. 執行以下命令檢視自動化進度的詳細資訊。把每個*##### ID* 取代為您自己的資訊。

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
```

系統會傳回相關資訊，如下所示。

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**
>
> 您也可以在主控台中監控自動化的狀態。在 Automation executions (自動化執行清單) 中，選擇您剛執行的自動化，接著選擇 Execution steps (執行步驟) 標籤。此索引標籤會顯示自動化動作的狀態。

對應自動化的目標

使用 `Targets` 參數迅速定義自動化以哪些資源為目標。例如，假設您想要執行自動化以重新啟動受管執行個體，您可以使用 `Targets` 參數指定 Amazon Elastic Compute Cloud (Amazon EC2) 標籤，以選定執行個體作為目標，而不必手動在主控台選擇或在命令中輸入數十個執行個體 ID。

當您執行使用目標的自動化時， 會為每個目標 AWS Systems Manager 建立子自動化。例如，如果您透過指定標籤來設定目標 Amazon Elastic Block Store (Amazon EBS) 磁碟區，以及這些標籤解析為 100 個 Amazon EBS 磁碟區，則 Systems Manager 會建立 100 個子自動化。所有子自動化達到最終狀態時，父自動化就會完成。

> **Note**
>
> 您在執行時間指定的 `input parameters` (無論是於主控台的 Input parameters (輸入參數) 部分或使用命令列的 `parameters` 選項)，都會自動由所有的子自動化處理。

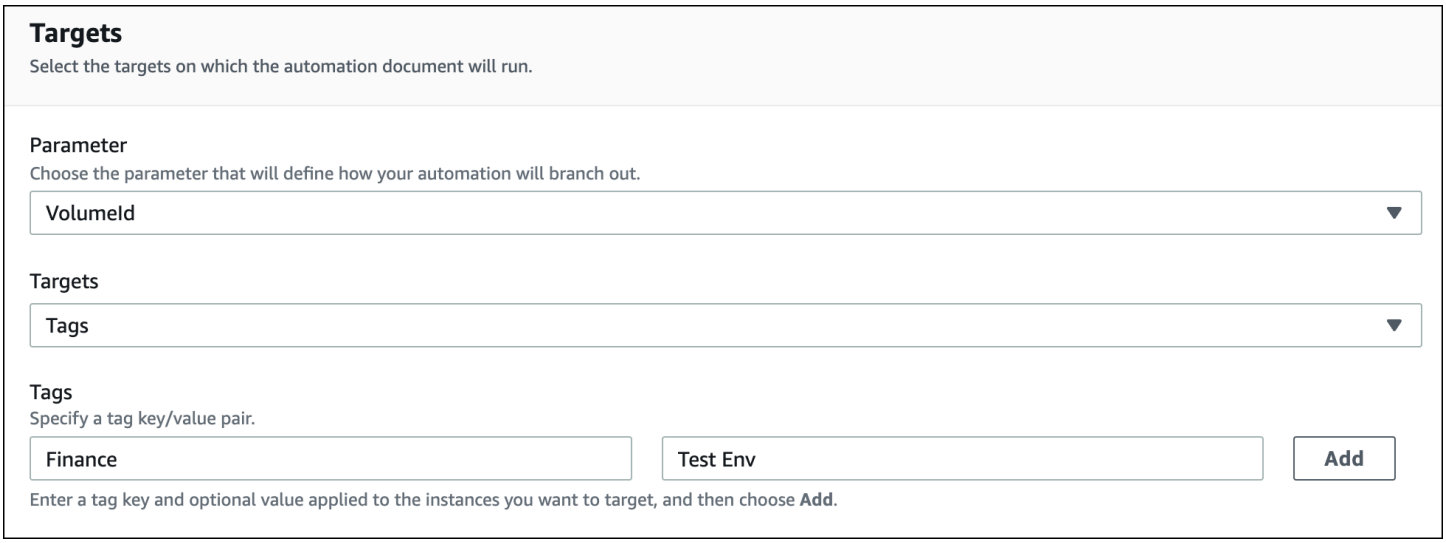
您可，使用標籤、Resource Groups 和參數值將自動化的資源設為目標。此外，您可以使用 `TargetMaps` 選項將命令列或檔案的多個參數值設為目標。以下部分會分別詳細說明這些目標設定選項。

將標籤設為目標

您可以指定單一標籤做為自動化目標。許多 AWS 資源支援標籤，包括 Amazon Elastic Compute Cloud (Amazon EC2) 和 Amazon Relational Database Service (Amazon RDS) 執行個體、Amazon Elastic Block Store (Amazon EBS) 磁碟區和快照、Resource Groups 和 Amazon Simple Storage Service (Amazon S3) 儲存貯體等。您可以鎖定標籤，快速在 AWS 資源上執行自動化。標籤是一種索引鍵/值組，例如 Operating_System:Linux 或 Department:Finance。如果您將特定名稱指派到資源，則您也可以使用「Name」這個詞做為索引鍵，以資源的名稱做為值。

當您指定標籤做為自動化的目標，您也要指定目標參數。目標參數會使用 `TargetParameterName` 選項。藉由選擇目標參數，您就會定義自動化執行的資源類型。您以標籤指定的目標參數必須是在 Runbook 中定義的有效參數。例如，假設您想要使用標籤將數十個 EC2 執行個體設為目標，請選擇 `InstanceId` 目標參數。選擇此參數後，您就會將執行個體定義為自動化的資源類型。建立自訂執行手冊時，必須將目標類型指定為 `/AWS::EC2::Instance` 以確保僅使用執行個體。否則，具有相同標籤的所有資源都將會成為目標。把帶有標籤的執行個體設為目標時，可能會包括已終止的執行個體。

下列螢幕擷取畫面會使用 `AWS-DetachEBSVolume` Runbook。邏輯目標參數為 `VolumeId`。



`AWS-DetachEBSVolume` Runbook 還包含名為 Target type (目標類型) 的特殊屬性，設定為 `/AWS::EC2::Volume`。這表示如果標籤鍵對 `Finance:TestEnv` 傳回不同類型的資源 (例如 EC2 執行個體、Amazon EBS 磁碟區、Amazon EBS 快照)，則只會使用 Amazon EBS 磁碟區。

> **Important**
>
> 目標參數名稱都區分大小寫。如果您使用 AWS Command Line Interface (AWS CLI) 或 執行自動化 AWS Tools for Windows PowerShell，則必須完全依照 Runbook 中所定義輸入目標參數名稱。如果不這麼做，系統會傳回

InvalidAutomationExecutionParametersException 錯誤。您可以使用 DescribeDocument API 操作查看特定 Runbook 中可用目標參數的資訊。以下是範例 AWS CLI 命令，該命令提供有關 AWS-DeleteSnapshot 文件的資訊。

```
aws ssm describe-document \
    --name AWS-DeleteSnapshot
```

以下是一些使用標籤以資源為目標的範例 AWS CLI 命令。

範例 1：使用索引鍵/值組將標籤設為目標以重新啟動 Amazon EC2 執行個體

此範例會重新啟動所有以 Department 為索引鍵和 HumanResources 為值標記的 Amazon EC2 執行個體。目標參數使用 Runbook 的 InstanceId 參數。此範例使用其他參數，藉由自動化服務角色 (也稱為擔任角色) 執行自動化。

```
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"
```

範例 2：使用索引鍵/值組將標籤設為目標以刪除 Amazon EBS 快照

以下範例使用 AWS-DeleteSnapshot Runbook 刪除所有索引鍵為 Name 且值為 January2018Backups 的快照。目標參數使用 VolumeId 參數。

```
aws ssm start-automation-execution \
    --document-name AWS-DeleteSnapshot \
    --targets Key=tag:Name,Values=January2018Backups \
    --target-parameter-name VolumeId
```

鎖定目標 AWS Resource Groups

您可以指定單一 AWS 資源群組做為自動化的目標。Systems Manager 會針對目標 Resource Group 中的每個物件建立子系自動化。

例如，假設您的一個 Resource Groups 名稱為 PatchedAMIs。此 Resource Group 包括一份清單，有 25 個定期修補的 Windows Amazon Machine Images (AMIs)。如果您執行使用 AWS-

CreateManagedWindowsInstance Runbook 的自動化並以此 Resource Group 為目標，則 Systems Manager 會為 25 個 AMIs 各建立一個子自動化。這表示以 PatchedAMIs 資源群組為目標後，自動化就會從已修補 AMIs 的清單建立 25 個執行個體。所有子自動化完成處理或達到最終狀態時，父自動化就會完成。

下列 AWS CLI 命令適用於 PatchAMIs 資源群組範例。命令會針對 --target-parameter-name 選項採用 AmiId 參數。命令不會包括其他參數以定義要從各個 AMI 建立哪種類型的執行個體。AWS-CreateManagedWindowsInstance Runbook 文件預設為 t2.medium 執行個體類型，所以此命令會為 Windows Server 建立 25 個 t2.medium 的 Amazon EC2 執行個體。

```
aws ssm start-automation-execution \
    --document-name AWS-CreateManagedWindowsInstance \
    --targets Key=ResourceGroup,Values=PatchedAMIs  \
    --target-parameter-name AmiId
```

以下主控台範例使用稱為 t2-micro-instances 的資源群組。

將參數值設為目標

您也可以將參數值設為目標。輸入 ParameterValues 做為索引鍵，接著在您想要執行自動化的地方輸入特定資源值。如果您指定多個值，Systems Manager 會於每個指定的值執行子自動化。

例如，假設您的 Runbook 包括 InstanceID 參數。如果您在執行自動化時以 InstanceID 參數的值為目標，則 Systems Manager 會針對每個指定的執行個體 ID 值執行子自動化。當自動化完成執行每個指定的執行個體，或是自動化失敗，父自動化就會完成。您最多可將 50 個參數值設為目標。

下列範例使用 AWS-CreateImage Runbook。指定的目標參數名稱為 InstanceId。索引鍵使用 ParameterValues。值為兩個 Amazon EC2 執行個體 ID。此命令會為每個執行個體建立一個自動化，並從每個執行個體產生一個 AMI。

```
aws ssm start-automation-execution
    --document-name AWS-CreateImage \
    --target-parameter-name InstanceId \
    --targets Key=ParameterValues,Values=i-02573cafcfEXAMPLE,i-0471e04240EXAMPLE
```

> **Note**
>
> AutomationAssumeRole 不是有效的參數。當執行自動化，且這些工作流程會設定目標參數值時，請勿選擇此項目。

將參數值對應設為目標

TargetMaps 選項可讓您更輕易將 ParameterValues 設為目標。您可以使用命令列的 TargetMaps 輸入一系列參數值。您可以在命令列指定最多 50 個參數值。如果想要執行命令以指定超過 50 個參數值，您可以在 JSON 檔案中輸入值。這樣您就可以從命令列呼叫檔案。

> **Note**
>
> 主控台不支援 TargetMaps 選項。

使用以下格式，藉由命令中的 TargetMaps 選項指定多個參數值：將每個########取代為您自己的資訊。

```
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”
```

如果您想要在 TargetMaps 選項輸入超過 50 個參數值，請使用以下 JSON 格式在檔案中指定值。提供多個參數值時，使用 JSON 檔案也能提升可讀性。

```
[

    {“parameter”: "value", “parameter 2”: "value", “parameter 3”: "value"},

    {“parameter 4”: "value", “parameter 5”: "value", "parameter 6": "value"}
```

```
]
```

使用 .json 副檔名儲存檔案。您可以使用下列命令來呼叫檔案：將每個########取代為您自己的資訊。

```
aws ssm start-automation-execution \
    --document-name runbook name \
    --parameters input parameters \
    --target-maps path to file/file name.json
```

只要您有從儲存貯體讀取資料的許可，您也可以從 Amazon Simple Storage Service (Amazon S3) 儲存貯體下載檔案。使用以下命令格式。將每個########取代為您自己的資訊。

```
aws ssm start-automation-execution \
    --document-name runbook name \
    --target-maps http://amzn-s3-demo-bucket.s3.amazonaws.com/file_name.json
```

以下範例案例可協助您了解 TargetMaps 選項。在此案例中，使用者想要從不同的 AMIs 建立不同類型的 Amazon EC2 執行個體。為執行此任務，使用者建立了名為 AMI_Testing 的 Runbook。此 Runbook 定義了兩個輸入參數：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"
  ]
}
```

使用者接著在名稱為 AMI_instance_types.json 的檔案中指定以下目標參數值。

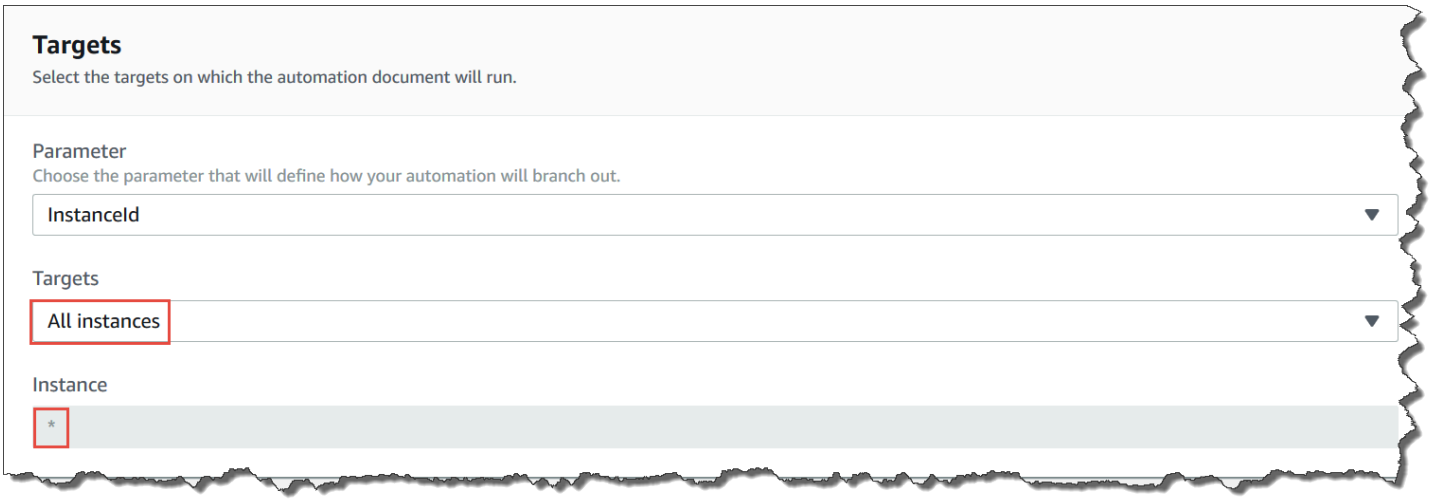
```
[
  {
    "instanceType" : ["t2.micro"],
    "imageId" : ["ami-b70554c8"]
  },
  {
    "instanceType" : ["t2.small"],
    "imageId" : ["ami-b70554c8"]
  },
  {
    "instanceType" : ["t2.medium"],
    "imageId" : ["ami-cfe4b2b0"]
  },
  {
    "instanceType" : ["t2.medium"],
    "imageId" : ["ami-cfe4b2b0"]
  },
  {
    "instanceType" : ["t2.medium"],
    "imageId" : ["ami-cfe4b2b0"]
  }
]
```

使用者可以執行以下命令，以執行自動化並建立 AMI_instance_types.json 中定義的五個 EC2 執行個體：

```
aws ssm start-automation-execution \
    --document-name AMI_Testing \
    --target-parameter-name imageId \
    --target-maps file:///home/TestUser/workspace/runinstances/AMI_instance_types.json
```

以所有 Amazon EC2 執行個體為目標

您可以在目前 AWS 帳戶 和 中的所有 Amazon EC2 執行個體上執行自動化， AWS 區域 方法是選擇目標清單中的所有執行個體。例如，如果您想要重新啟動 AWS 帳戶 和目前 的所有 Amazon EC2 執行個體 AWS 區域，您可以選擇 **AWS-RestartEC2Instance** Runbook，然後從目標清單中選擇所有執行個體。



在您選擇 All instances (所有執行個體) 後，Systems Manager 會填入帶有星號 (*) 的 Instance (執行個體) 欄位，且無法對欄位進行變更 (欄位會變灰)。Systems Manager 也會讓 Input parameters (輸入參數) 欄位中的 InstanceId 不可變更。如果您選擇鎖定所有執行個體，則將這些欄位變得無法變更會是預期的行為。

大規模控制自動化

您可以透過指定並行值和錯誤閾值，控制跨 AWS 資源機群部署自動化。並行和錯誤閾值統稱為「速率控制」。

並行數量

使用並行可讓您指定允許同時以多少資源執行自動化。在處理自動化時，並行可協助限制資源的影響或停機時間。您可以指定絕對數量的資源 (例如 20 個) 或目標集的百分比 (例如 10%)。

佇列系統會將自動化傳遞至單一資源並等到初始叫用完成，再將自動化傳送至另外兩個資源。系統會以指數方式將自動化傳送至更多資源，直到達到並行值為止。

錯誤閾值

使用錯誤閾值指定在 AWS Systems Manager 停止傳送自動化至其他資源之前，允許多少自動化失敗。您可以指定絕對數量的錯誤 (例如 10 個) 或目標集的百分比 (例如 10%)。

例如，假設您指定 3 個錯誤的絕對數量，系統會在收到第四個錯誤時停止執行自動化。如果您指定 0，系統會在第一個錯誤結果傳回時停止其他目標上執行的自動化。

例如，假設您傳送自動化到至 50 個執行個體並將錯誤閾值設為 10%，系統會在收到第五個錯誤時停止傳送命令至其他執行個體。達到錯誤閾值時已經在執行自動化的叫用允許完成，但其中某些自動化也可能會失敗。如果您要確保錯誤不會超過針對錯誤閾值指定的數量，請將 Concurrency (並行) 值設為 1，讓自動化一次執行一個。

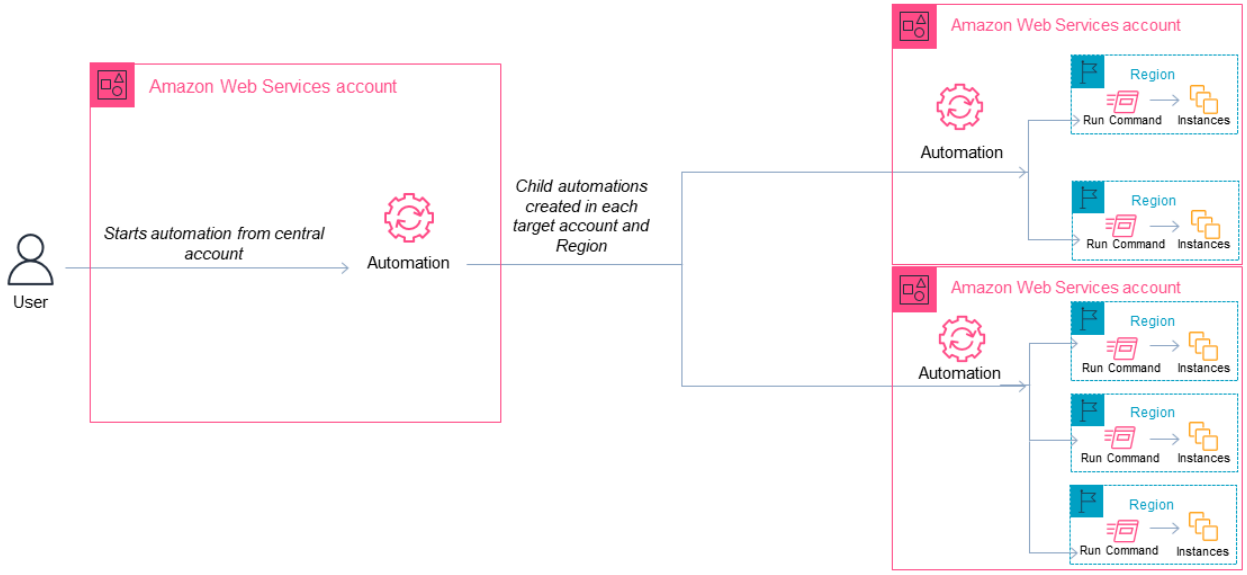
在多個 AWS 區域 和 帳戶中執行自動化

您可以從中央帳戶跨多個 AWS 區域 和 AWS 帳戶 /或 AWS Organizations 組織單位 (OUs) 執行 AWS Systems Manager 自動化。自動化是 中的工具 AWS Systems Manager。在多個區域和帳戶或 OUs 中執行自動化，可縮短管理 AWS 資源所需的時間，同時增強運算環境的安全性。

例如，您可以使用自動化 Runbook 來執行下列動作：

- 集中實作修補和安全性更新。
- 糾正 VPC 組態或 Amazon S3 儲存貯體政策的合規偏離。
- 大規模管理資源 (例如 Amazon Elastic Compute Cloud (Amazon EC2) EC2 執行個體)。

下圖顯示範例為使用者從中央帳戶的多個區域和帳戶中執行 `AWS-RestartEC2Instances` Runbook。自動化會使用目標區域和帳戶中的指定標籤找出執行個體。



選擇自動化的中央帳戶

如果您想要跨 OU 執行自動化，則中央帳戶必須擁有列出 OU 中所有帳戶的許可。只有委派管理員帳戶或組織的管理帳戶才能執行此操作。建議您遵循 AWS Organizations 最佳實務，並使用委派的管理員帳戶。如需 AWS Organizations 最佳實務的詳細資訊，請參閱AWS Organizations 《 使用者指南》中的管理帳戶的最佳實務。若要為 Systems Manager 建立委派管理員帳戶，您可以搭配 使用 `register-delegated-administrator`命令 AWS CLI ，如下列範例所示。

```
aws organizations register-delegated-administrator \
    --account-id delegated admin account ID \
    --service-principal ssm.amazonaws.com
```

如果您想要在不受 AWS Organizations管理的多個帳戶執行自動化，則我們建議您為自動化管理建立專屬帳戶。從專用帳戶執行所有跨帳戶自動化功能，可簡化 IAM 許可管理、排解疑難，並在作業與管理之間建立分隔層。如果您使用 AWS Organizations，但只想要以個別帳戶為目標，而非 OUs則建議使用此方法。

執行自動化的運作方式

跨多個區域和帳戶或 OU 執行自動化的運作方式如下：

1. 登入您想要設定為自動化主帳戶的帳戶。
2. 使用本主題中的 設定多區域和多帳戶自動化的管理帳戶許可 處理程序，建立下列 IAM 角色。
   - **AWS-SystemsManager-AutomationAdministrationRole** - 此角色可給予使用者在多個帳戶 和 OU 中執行自動化的許可。
   - **AWS-SystemsManager-AutomationExecutionRole** - 此角色可給予使用者在目標帳戶中執行自動化的許可。
3. 選擇您想要執行自動化的 Runbook、區域、帳戶或 OU。

   > **Note**
   >
   > 請確定目標 OU 包含所需的帳戶。如果您選擇自訂 Runbook，則 Runbook 必須與所有目標帳戶共用。如需共用 Runbook 的資訊，請參閱 共用 SSM 文件。如需共用 Runbook 的詳細資訊，請參閱 使用共用的 SSM 文件。

4. 執行自動化。

   > **Note**
   >
   > 跨多個區域、帳戶或 OU 執行自動化時，從主要帳戶執行的自動化會在每個目標帳戶中啟動子系自動化。主要帳戶中的自動化對每個目標帳戶包含 `aws:executeAutomation` 個步驟。

5. 使用來自主控台或 的 AWS Systems Manager GetAutomationExecution、DescribeAutomationStepExecutions 和 DescribeAutomationExecutions API 操作 AWS CLI 來監控自動化進度。在您的主要帳戶中自動化步驟的輸出將是子系自動化的 `AutomationExecutionId`。若要檢視在目標帳戶中建立之子系自動化的輸出，請務必在您的請求中指定適當的帳戶、區域和 `AutomationExecutionId`。

設定多區域和多帳戶自動化的管理帳戶許可

依照以下步驟，使用 AWS CloudFormation建立 Systems Manager Automation 多區域和多帳戶自動化所需的 IAM 角色。此處理程序描述了如何建立 **AWS-SystemsManager-AutomationAdministrationRole** 角色。您只需要在自動化中央帳戶中建立此角色。此處理程序也描述了如何建立 **AWS-SystemsManager-AutomationExecutionRole** 角色。您必須在想要設為目標以執行多區域和多帳戶自動化的每個帳戶中建立此角色。建議您使用 AWS CloudFormation StackSets 在帳戶中建立 **AWS-SystemsManager-AutomationExecutionRole** 角色，帳戶是您想要執行多區域和多帳戶自動化的目標帳戶。

使用 建立多區域和多帳戶自動化所需的 IAM 管理角色 AWS CloudFormation

1. 下載並解壓縮 AWS-SystemsManager-AutomationAdministrationRole.zip。或者，如果您的帳戶是由 管理 AWS Organizations AWS-SystemsManager-AutomationAdministrationRole (org).zip。此檔案包含 AWS-SystemsManager-AutomationAdministrationRole.yaml AWS CloudFormation 範本檔案。
2. 在 https：//https://console.aws.amazon.com/cloudformation 開啟 AWS CloudFormation 主控台。
3. 選擇 Create Stack (建立堆疊)。
4. 在 Specify template (指定範本) 區段中，選擇 Upload a template file (上傳範本檔案)。
5. 選擇選擇檔案，然後選擇AWS-SystemsManager-AutomationAdministrationRole.yaml AWS CloudFormation 範本檔案。
6. 選擇 Next (下一步)。
7. 在 Specify stack details (指定堆疊詳細資訊) 頁面，於 Stack name (堆疊名稱) 欄位輸入名稱。
8. 選擇 Next (下一步)。
9. 在 Configure stack options (設定堆疊選項) 頁面，針對您想要使用的任何選項輸入值。選擇 Next (下一步)。
10. 在檢閱頁面上，向下捲動並選擇我確認 AWS CloudFormation 可能會使用自訂名稱選項建立 IAM 資源。
11. 選擇建立堆疊。

AWS CloudFormation 顯示大約三分鐘的 CREATE_IN_PROGRESS 狀態。狀態會變更為 CREATE_COMPLETE (CREATE_COMPLETE)。

您必須在您想要設為目標以執行多區域和多帳戶自動化的每個帳戶中重複下列程序。

使用 建立多區域和多帳戶自動化所需的 IAM 自動化角色 AWS CloudFormation

1. 下載 AWS-SystemsManager-AutomationExecutionRole.zip。或者，如果您的帳戶是由 管理 AWS Organizations AWS-SystemsManager-AutomationExecutionRole (org).zip。此檔案包含 AWS-SystemsManager-AutomationExecutionRole.yaml AWS CloudFormation 範本檔案。
2. 在 https：//https://console.aws.amazon.com/cloudformation 開啟 AWS CloudFormation 主控台。
3. 選擇 Create Stack (建立堆疊)。
4. 在 Specify template (指定範本) 區段中，選擇 Upload a template file (上傳範本檔案)。

5. 選擇選擇檔案，然後選擇AWS-SystemsManager-AutomationExecutionRole.yaml AWS CloudFormation 範本檔案。
6. 選擇 Next (下一步)。
7. 在 Specify stack details (指定堆疊詳細資訊) 頁面，於 Stack name (堆疊名稱) 欄位輸入名稱。
8. 在 Parameters (參數) 區段的 AdminAccountId 欄位中，輸入自動化中央帳戶的 ID。
9. 如果您要為 AWS Organizations 環境設定此角色，則 區段中還有另一個欄位稱為 OrganizationID。輸入 AWS 組織的 ID。
10. 選擇 Next (下一步)。
11. 在 Configure stack options (設定堆疊選項) 頁面，針對您想要使用的任何選項輸入值。選擇 Next (下一步)。
12. 在檢閱頁面上，向下捲動並選擇我確認 AWS CloudFormation 可能會使用自訂名稱選項建立 IAM 資源。
13. 選擇建立堆疊。

AWS CloudFormation 顯示大約三分鐘的 CREATE_IN_PROGRESS 狀態。狀態會變更為 CREATE_COMPLETE (CREATE_COMPLETE)。

在多個區域和帳戶中執行自動化 (主控台)

以下程序說明如何使用 Systems Manager 主控台，從 Automation 管理帳戶中於多個區域和帳戶內執行自動化。

開始之前

完成以下程序之前，請記下以下資訊：

- 您用來執行多區域或多帳戶自動化的使用者或角色必須具有該 AWS-SystemsManager-AutomationAdministrationRole 角色的 iam:PassRole 許可。
- 您要執行自動化的AWS 帳戶 IDs 或 OUs。
- 您希望執行自動化的 Systems Manager 支援區域。
- 您希望執行自動化的資源群組標籤鍵或標籤值，或是其名稱。

在多個區域和帳戶中執行自動化

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。

2. 在導覽窗格中，選擇 Automation (自動化)，接著選擇 Execute automation (執行自動化)。
3. 在 Automation document (自動化文件)清單中，選擇 Runbook。在 Document categories (文件類別) 窗格中選擇一個或多個選項，根據 SSM 文件的用途來進行篩選。若要檢視您擁有的 Runbook，請選擇 Owned by me (我所擁有的) 索引標籤。若要檢視與您帳戶共用的 Runbook，請選擇 Shared with me (與我共用的) 索引標籤。若要檢視所有 Runbook，請選擇 All documents (所有文件) 索引標籤。

   > **Note**
   >
   > 您可以選擇 Runbook 名稱檢視 Runbook 資訊。

4. 在 Document details (文件詳細資訊) 部分，確認 Document version (文件版本) 設定為您想要執行的版本。系統包括以下版本選項：

   - 執行期的預設版本：如果 Automation 執行手冊會定期更新且已指派新的預設版本，則請選擇此選項。
   - 執行期的最新版本：如果 Automation 執行手冊會定期更新，而您想要執行最近更新的版本，請選擇此選項。
   - 1 (預設)：選擇此選項以執行文件的第一個版本，也是預設版本。
5. 選擇 Next (下一步)。
6. 在 Execute automation document (執行自動化文件) 頁面，選擇 Multi-account and Region (多帳戶和區域)。
7. 在目標帳戶和區域區段中，使用帳戶和組織 OUs) 欄位來指定您要執行自動化的不同 AWS 帳戶 或 AWS 組織單位 (OUs)。使用逗號分隔多個帳戶或 OU。
8. 使用 AWS 區域 清單選擇您想要執行自動化的一個或多個區域。
9. 使用 Multi-Region and account rate control (多區域和帳戶速率控制) 選項將自動化限制為數量有限的帳戶在數量有限的區域中執行。這些選項不會限制能夠執行自動化的 AWS 資源數量。

   a. 在 Location (account-Region pair) concurrency (位置 (帳戶區域對) 並行) 部分，選擇一個選項以限制能夠同時在多個帳戶和區域中執行的自動化數量。例如，如果您選擇在位於四 AWS 帳戶(4) 個 的五 (5) 個 中執行自動化 AWS 區域，則 Systems Manager 會在總共 20 個帳戶區域對中執行自動化。您可以使用此選項指定一個絕對數字，例如 **2**，這樣自動化就只會同時在兩個帳戶區域對中執行。或者您也能指定可同時執行的帳戶區域對百分比。例如有 20 個帳戶區域對，假設您指定了 20%，則自動化會同時在最多五 (5) 個帳戶區域對中執行。

      - 選擇 targets (目標)，輸入可以同時執行自動化的帳戶區域對絕對數量。

- 選擇 percent (百分比)，輸入可以同時執行自動化的帳戶區域對總數之百分比。

b. 在 Error threshold (錯誤閾值) 部分，選擇一個選項：

- 選擇 errors (錯誤)，輸入在 Automation 停止傳送自動化至其他資源之前允許的錯誤絕對數量。
- 選擇 percent (百分比)，輸入在 Automation 停止傳送自動化至其他資源之前允許的錯誤百分比。

10. 在目標區段中，選擇您希望如何將資源鎖定在您想要執行自動化 AWS 的位置。這些選項是必要的。

a. 使用 Parameter (參數) 清單選擇一個參數。Parameter (參數) 清單中的項目，是由您在此程序一開始所選取自動化 Runbook 中的參數決定。藉由選擇參數，您就會定義自動化工作流程執行的資源類型。

b. 使用 Targets (目標) 清單選擇您想要如何將資源設為目標。

i. 如果您選擇使用參數值將資源設為目標，則請在 Input parameters (輸入參數) 區段為您所選的參數輸入參數值。

ii. 如果您選擇使用 來鎖定資源 AWS Resource Groups，請從資源群組清單中選擇群組的名稱。

iii. 如果您選擇使用標籤將資源設為目標，請在提供的欄位中輸入標籤索引鍵 (選用) 和標籤值。選擇新增。

iv. 如果您想要在目前 的所有執行個體上執行 Automation Runbook AWS 帳戶 AWS 區域，然後選擇所有執行個體。

11. 在 Input parameters (輸入參數) 部分，指定所需的輸入。從 AutomationAssumeRole 清單中選擇 `AWS-SystemsManager-AutomationAdministrationRole` IAM 服務角色。

> **Note**
>
> 您可能不必選擇 Input parameters (輸入參數) 部分的其中一些選項。這是因為您使用標籤或資源群組在多個區域和帳戶中將資源設為目標。例如，假設您選擇了 `AWS-RestartEC2Instance` Runbook，就不必在 Input parameters (輸入參數) 部分指定或選擇執行個體 ID。自動化會使用您指定的標籤找出要重新啟動的執行個體。

12. (選用) 選擇要套用至您的自動化以便加以監控的 CloudWatch 警示。若要將 CloudWatch 警示連接至您的自動化，啟動自動化的 IAM 主體必須具備 `iam:createServiceLinkedRole` 動作的許可。如需有關 CloudWatch 警示的詳細資訊，請參閱使用 Amazon CloudWatch 警示。請注

意，如果警示啟用，則會取消自動化，並會執行您定義的任何 `OnCancel` 步驟。如果您使用 AWS CloudTrail，您會在追蹤中看到 API 呼叫。

13. 使用速率控制區段中的選項來限制可在每個帳戶區域對內執行自動化 AWS 的資源數量。

    在 Concurrency (並行) 部分，選擇一個選項：

    - 選擇 targets (目標)，輸入可以同時執行自動化工作流程的目標絕對數量。
    - 選擇 percentage (百分比)，輸入可以同時執行自動化工作流程的目標集百分比。

14. 在 Error threshold (錯誤閾值) 部分，選擇一個選項：

    - 選擇 errors (錯誤)，輸入在自動化停止傳送工作流程至其他資源之前允許的錯誤絕對數量。
    - 選擇 percentage (百分比)，輸入在自動化停止傳送工作流程至其他資源之前允許的錯誤百分比。

15. 選擇 Execute (執行)。

在多個區域和帳戶中執行自動化 (命令列)

下列程序說明如何使用 AWS CLI （在 Linux 或 Windows 上） 或 AWS Tools for PowerShell 從自動化管理帳戶在多個區域和帳戶中執行自動化。

開始之前

完成以下程序之前，請記下以下資訊：

- 您要執行自動化的AWS 帳戶 IDs 或 OUs。
- 您希望執行自動化的 Systems Manager 支援區域。
- 您希望執行自動化的資源群組標籤鍵或標籤值，或是其名稱。

在多個區域和帳戶中執行自動化

1. AWS Tools for PowerShell如果您尚未安裝和設定 AWS CLI 或 。

   如需相關資訊，請參閱安裝或更新 AWS CLI的最新版本和安裝 AWS Tools for PowerShell。

2. 使用以下格式建立命令，在多個區域和帳戶中執行自動化。將每個########取代為您自己的資訊。

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" }
```

以下是數個範例。

範例 1：此範例會重新啟動整個 AWS Organizations 組織中三個區域中的 EC2 執行個體。這可透過鎖定組織的根 ID 並包含子 OU 來實現。

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}
```

範例 2：此範例會重新啟動不同帳戶和區域中的特定 EC2 執行個體。

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"]
        })
```

範例 3：此範例會重新啟動位於 us-east-2 和 us-west-1 區域的 123456789012 和 987654321098 帳戶中的 EC2 執行個體。執行個體必須使用標籤金鑰對值 Env-PROD 加上標籤。

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" }
```

範例 4：此範例會重新啟動位於 eu-central-1 區域的 123456789012 和 987654321098 帳戶中 EC2 執行個體。執行個體必須是prod-instances AWS 資源群組的成員。

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" }
```

範例 5：此範例會重新啟動ou-1a2b3c-4d5e6c AWS 組織單位 (OU) 中的 EC2 執行個體。這些執行個體位於 us-west-1 和 us-west-2 區域。執行個體必須是WebServices AWS 資源群組的成員。

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" }
```

系統會傳回與以下相似的資訊。

Linux & macOS

```
{
        "AutomationExecutionId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE"
    }
```

Windows

```
{
        "AutomationExecutionId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE"
    }
```

PowerShell

```
4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE
```

3. 執行以下命令檢視自動化的詳細資訊。把*##### ID* 取代為您自己的資訊。

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. 執行以下命令檢視自動化進度的詳細資訊。

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**
>
> 您也可以在主控台中監控自動化的狀態。在 Automation executions (自動化執行清單) 中，選擇您剛執行的自動化，接著選擇 Execution steps (執行步驟) 標籤。此索引標籤會顯示自動化動作的狀態。

詳細資訊

[以 AWS Systems Manager 自動化集中多帳戶和多區域修補]

根據 EventBridge 事件執行自動化

您可以將 Runbook 指定為 Amazon EventBridge 事件的目標，藉此開始自動化。您可以根據排程或在特定的 AWS 系統事件發生時開始自動化。例如，假設您建立名稱為 BootStrapInstances 的 Runbook，而該文件在執行個體啟動時於執行個體上安裝軟體。若要指定 BootStrapInstances Runbook (和對應的工作流程) 做為 EventBridge 事件的目標，您要先建立新的 EventBridge 規則。(以下為範例規則：Service name (服務名稱)：EC2，Event Type (事件類型)：EC2 執行個體狀態 - 變更通知，Specific state(s) (特定狀態)：執行 Any instance (任何執行個體)。) 然後，您可以使用下列程序，使用 EventBridge 主控台和 AWS Command Line Interface () 將 BootStrapInstances Runbook 指定為事件的目標AWS CLI。新的執行個體啟動時，系統會執行自動化和安裝軟體。

如需建立 Runbook 的資訊，請參閱 建立您自己的執行手冊。

建立使用 Runbook (主控台) 的 EventBridge 事件

使用以下程序設定來將 Runbook 做為 EventBridge 事件的目標。

將 Runbook 設定為 EventBridge 事件的目標

1. 前往 https://console.aws.amazon.com/events/ 開啟 Amazon EventBridge 主控台。
2. 在導覽窗格中，選擇規則。
3. 選擇建立規則。
4. 輸入規則的名稱和描述。

   在同一個區域和同一個事件匯流排上，規則不能與另一個規則同名。
5. 針對事件匯流排，選擇要與此規則建立關聯的事件匯流排。如果您希望此規則回應來自您自己的相符事件 AWS 帳戶，請選取預設值。當您 AWS 服務 帳戶中的 發出事件時，一律會傳送至您帳戶的預設事件匯流排。
6. 選擇該規則的觸發方式。

根據...建立規則	執行此作業...
事件	a. 針對規則類型，選擇具有事件模式的規則。 b. 選擇下一步。 c. 在事件來源欄位中，選擇 AWS 事件或 EventBridge 合作夥伴事件。 d. 在 Event pattern (事件模式) 區段中，執行下列其中一個動作： • 若要使用範本建立您的事件模式，請選擇 Event pattern form (事件模式表單)，然後選擇 Event source (事件來源)、AWS service (服

根據…建立規則	執行此作業...	
	務)，以及 Event type (事件類型)。如果您選擇所有事件做為事件類型，則 發出的所有事件 AWS 服務 都會符合規則。 若要自定範本，請選擇自訂模式 (JSON 編輯器)並進行變更。 • 若要使用自訂事件模式，請選擇 Custom pattern (JSON editor) (自訂模式 (JSON 編輯器)) 並建立事件模式。	

根據…建立規則	執行此作業...	
排程	a. 針對 Rule type (規則類型)，選擇 Schedule (排程)。 b. 選擇 Next (下一步)。 c. 針對 Schedule pattern (排程模式)，執行下列其中一項動作： • 若要使用 Cron 運算式定義排程，請選擇 A fine-grained schedule that runs at a specific time, such as 8:00 a.m. (在特定時間 (如上午 8:00) 執行的精細時間表)。PST on the first Monday of every month (每個月的第一個星期一的 PST) 並輸入 Cron 運算式。 • 若要使用 Rate 運算式定義排程，請選擇 A schedule that runs at a regular rate, such as every 10 minutes (按一般速率執行的排程，例如每 10 分鐘一次)，然後輸入 Rate 運算式。	

7. 選擇 Next (下一步)。
8. 在目標類型欄位中，選擇 AWS 服務。
9. 針對 Select a target (選取目標)，請選擇 Systems Manager Automation。
10. 對於 Document (文件)，選擇叫用目標時要使用的 Runbook。
11. 在 Configure automation parameter(s) (設定自動化參數) 區段中，保留預設參數值 (若有) 或輸入您自己的值。

> **Note**
>
> 若要建立目標，您必須為每個必要參數指定值。如果不這麼做，系統會建立規則，但規則不會執行。

12. 對於許多目標類型而言，EventBridge 需要許可才能將事件傳送到目標。在這些情況下，EventBridge 可建立執行您的規則所需的 IAM 角色。執行以下任意一項：

    - 若要自動建立 IAM 角色，請選擇為此特定資源建立新角色。
    - 若要使用您早前建立的 IAM 角色，請選擇 Use existing role (使用現有角色) 並從下拉式清單中選取現有角色。請注意，您可能需要更新 IAM 角色的信任政策，使其包含 EventBridge。以下是範例：

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

13. 選擇 Next (下一步)。
14. (選用) 為規則輸入一或多個標籤。如需詳細資訊，請參閱《Amazon EventBridge 使用者指南》中的標記您的 Amazon EventBridge 資源。
15. 選擇 Next (下一步)。
16. 檢閱規則的詳細資訊，然後選擇建立規則。

建立使用 Runbook (命令列) 的 EventBridge 事件

下列程序說明如何使用 AWS CLI （在 Linux 或 Windows 上） 或 AWS Tools for PowerShell 建立 EventBridge 事件規則，並將 Runbook 設定為目標。

將 Runbook 設定為 EventBridge 事件的目標

1. AWS Tools for PowerShell如果您尚未安裝和設定 AWS CLI 或 。

   如需相關資訊，請參閱安裝或更新 AWS CLI的最新版本和安裝 AWS Tools for PowerShell。

2. 建立命令來指定新的 EventBridge 事件規則。將每個########取代為您自己的資訊。

   「依據排程觸發」

   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"
   ```

   以下範例會建立 EventBridge 事件規則，在每天早上 9:00 (UTC) 開始。

   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 * * ? *)"
```

「依據事件觸發」

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"]}'
```

以下範例建立 EventBridge 事件規則，當區域中的任何 EC2 執行個體變更狀態時會觸開啟規則。

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"]}'
```

命令會傳回與以下相似的新 EventBridge 規則詳細資訊。

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. 建立命令，指定 Runbook 做為您在步驟 2 中所建立 EventBridge 事件規則的目標。將每個*#####*取代為您自己的資訊。

   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
   ```

   以下範例會建立 EventBridge 事件目標，使用 Runbook `AWS-StartEC2Instance` 文件啟動指定的執行個體 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
```

系統會傳回相關資訊，如下所示。

Linux & macOS

```
{
"FailedEntries": [],
```

```
"FailedEntryCount": 0
}
```

Windows

```
{
"FailedEntries": [],
"FailedEntryCount": 0
}
```

PowerShell

如果 PowerShell 的命令成功，則不會有輸出訊息。

逐步執行自動化

下列程序說明如何使用 AWS Systems Manager 主控台和 AWS Command Line Interface (AWS CLI) 來使用手動執行模式執行自動化。透過使用手動執行模式，自動化開始在等待狀態中開始並在每個步驟之間的等待狀態中暫停。這讓您能夠控制自動化的進行，在您需要檢閱每個步驟的結果再繼續時很有用。

自動化會在目前的使用者內容中執行。這表示只要有使用 Runbook 和 Runbook 所呼叫任何動作的許可，您就不必設定其他的 IAM 許可。如果您在 IAM 中有管理員許可，您便已經有執行此自動化的許可。

逐步執行自動化 (主控台)

以下程序會示範如何使用 Systems Manager 主控台逐步手動執行自動化。

逐步執行自動化

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Automation (自動化)，接著選擇 Execute automation (執行自動化)。
3. 在 Automation document (自動化文件)清單中，選擇 Runbook。在 Document categories (文件類別) 窗格中選擇一個或多個選項，根據 SSM 文件的用途來進行篩選。若要檢視您擁有的 Runbook，請選擇 Owned by me (我所擁有的) 索引標籤。若要檢視與您帳戶共用的 Runbook，請選擇 Shared with me (與我共用的) 索引標籤。若要檢視所有 Runbook，請選擇 All documents (所有文件) 索引標籤。

> **Note**
>
> 您可以選擇 Runbook 名稱檢視 Runbook 資訊。

4. 在 Document details (文件詳細資訊) 部分，確認 Document version (文件版本) 設定為您想要執行的版本。系統包括以下版本選項：
   - 執行期的預設版本：如果 Automation 執行手冊會定期更新且已指派新的預設版本，則請選擇此選項。
   - 執行期的最新版本：如果 Automation 執行手冊會定期更新，而您想要執行最近更新的版本，請選擇此選項。
   - 1 (預設)：選擇此選項以執行文件的第一個版本，也是預設版本。
5. 選擇 Next (下一步)。
6. 在 Execution Mode (執行模式) 部分，選擇 Manual execution (手動執行)。
7. 在 Input parameters (輸入參數) 部分，指定所需的輸入。或者，您也可以從 AutomationAssumeRole 清單中選擇 IAM 服務角色。
8. 選擇 Execute (執行)。
9. 準備好開始自動化的第一步時，選擇 Execute this step (執行此步驟)。自動化會繼續進行步驟一，並在執行您於本程序步驟 3 中所選的 Runbook 指定的任何後續步驟之前暫停。如果 Runbook 有多個步驟，您必須為每個步驟選擇 Execute this step (執行此步驟)，自動化才會繼續。每次您選擇 Execute this step (執行此步驟) 時，動作便會執行。

   > **Note**
   >
   > 主控台會顯示自動化的狀態。若自動化無法執行步驟，請參閱 故障診斷 Systems Manager Automation。

10. 在您完成 Runbook 指定的所有步驟後，請選擇 Complete and view results (完成並檢視結果)，以完成自動化並檢視結果。

逐步執行自動化 (命令列)

下列程序說明如何使用 AWS CLI （ 在 Linux、 macOS或 Windows 上 ） 或 ，逐步 AWS Tools for PowerShell 手動執行自動化。

逐步執行自動化

1. AWS Tools for PowerShell如果您尚未安裝和設定 AWS CLI 或 。

   如需相關資訊，請參閱安裝或更新 AWS CLI的最新版本和安裝 AWS Tools for PowerShell。

2. 執行以下命令來啟動手動自動化。將每個########取代為您自己的資訊。

   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 Runbook 重新啟動指定 EC2 執行個體的範例。

   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"}
```

系統會傳回相關資訊，如下所示。

Linux & macOS

```
{
    "AutomationExecutionId": "ba9cd881-1b36-4d31-a698-0123456789ab"
}
```

Windows

```
{
    "AutomationExecutionId": "ba9cd881-1b36-4d31-a698-0123456789ab"
}
```

PowerShell

```
ba9cd881-1b36-4d31-a698-0123456789ab
```

3. 當您準備好開始自動化的第一步時，請執行以下命令。將每個########取代為您自己的資訊。自動化會繼續進行步驟一，並在執行您於本程序步驟 1 中所選的 Runbook 指定的任何後續步驟之前暫停。若 Runbook 有多個步驟，您必須為每個步驟執行以下命令，自動化才會繼續。

   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"}
```

如果命令成功，則無輸出訊息。

4. 執行以下命令來擷取自動化中每個步驟的執行狀態。

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
```

系統會傳回相關資訊，如下所示。

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. 在所選擇 Runbook 中指定的所有步驟皆完成後，執行以下命令來完成自動化。將每個#######取代為您自己的資訊。

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
```

如果命令成功，則無輸出訊息。

使用 State Manager 關聯排程自動化

您可以透過建立與 Runbook 的State Manager關聯來啟動自動化。 State Manager 是 中的工具 AWS Systems Manager。透過建立與 Runbook 的State Manager關聯，您可以鎖定不同類型的 AWS 資源。例如，您可以建立在 AWS 資源上強制執行所需狀態的關聯，包括下列項目：

- 將 Systems Manager 角色連接到 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體以作為受管執行個體。
- 為安全群組強制執行所需的輸入和輸出規則。
- 建立或刪除 Amazon DynamoDB 備份。
- 建立 Amazon Elastic Block Store (Amazon EBS) 快照。
- 關閉 Amazon Simple Storage Service (Amazon S3) 儲存貯體上的讀取和寫入許可。
- 啟動、重新啟動或停止受管執行個體和 Amazon Relational Database Service (Amazon RDS) 執行個體。
- 將修補程式套用至 Linux、macOS、和 WiWindows AMIs。

使用下列程序建立使用 AWS Systems Manager 主控台和 AWS Command Line Interface () 執行自動化的State Manager關聯AWS CLI。如需有關關聯的一般資訊，以及有關建立使用 SSM Command 文件或 Policy 文件之關聯的資訊，請參閱建立關聯。

開始之前

使用 State Manager 執行自動化前，請注意以下重要的詳細資訊。

- 在建立使用 Runbook 的關聯之前，請確認您已設定 Automation 的許可，這是其中的工具 AWS Systems Manager。如需詳細資訊，請參閱設定自動化。

- 使用 Runbook 的 State Manager 關聯會影響您 AWS 帳戶中同時執行的自動化作業的最大數量。一次最多可執行 100 個並行自動化作業。如需相關資訊，請參閱《Amazon Web Services 一般參考》中的 Systems Manager 服務配額一節。
- 執行自動化時，State Manager 不會在 AWS CloudTrail中記錄由自動化啟動的 API 操作。
- Systems Manager 會自動建立服務連結角色，以便 State Manager 擁有呼叫 Systems Manager Automation API 操作的許可。如果需要，您可以從 或 執行下列命令， AWS CLI 自行建立服務連結角色 AWS Tools for PowerShell。

  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
  ```

  如需服務連結角色的詳細資訊，請參閱使用 Systems Manager 的服務連結角色。

建立執行自動化的關聯 (主控台)

下列程序說明如何使用 Systems Manager 主控台來建立執行自動化的 State Manager 關聯。

建立執行自動化的 State Manager 關聯

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 State Manager，然後選擇 Create association (建立關聯)。
3. 在 Name (名稱) 欄位中指定名稱。此為選用操作，但建議您採用。
4. 在 Document (文件) 清單中，選擇 Runbook。使用搜尋列來篩選 Document type : Equal : Automation Runbook。若要查看更多 Runbook，請使用搜尋列右側的號碼。

> **Note**
>
> 您可以選擇 Runbook 名稱檢視 Runbook 資訊。

5. 選擇 Simple execution (簡易執行)，透過為這些目標指定資源 ID，在一或多個目標上執行自動化。選擇速率控制，透過指定標籤或 等目標選項，在 AWS 資源機群中執行自動化 AWS Resource Groups。您也可以透過指定並行和錯誤閾值，控制在資源上的自動化操作。

   如果選擇 Rate control (速率控制)，便會顯示 Targets (目標) 部分。

6. 在 Targets (目標) 部分，選擇將資源設為目標的方法。

   a. (必要) 在 Parameter (參數) 清單中，選擇一個參數。Parameter (參數) 清單中的項目，是由您在此程序一開始所選取 Runbook 中的參數決定。藉由選擇參數，您就會定義自動化執行的資源類型。

   b. (必要) 在 Targets (目標) 清單中，選擇將資源設為目標的方法。

      - Resource Group (資源群組)：從 Resource Group (資源群組) 清單中選擇群組名稱。如需在 Runbook AWS Resource Groups 中鎖定目標的詳細資訊，請參閱 鎖定目標 AWS Resource Groups。
      - Tags (標籤)：在提供的欄位中輸入標籤鍵和 (選擇性) 標籤值。選擇新增。如需有關在 Runbook 中鎖定標籤的詳細資訊，請參閱 將標籤設為目標。
      - Parameter Values (參數值)：在 Input parameters (輸入參數) 部分輸入值。如果您指定多個值，Systems Manager 會於每個指定的值執行子自動化。

        例如，假設您的 Runbook 包括 InstanceID 參數。如果您在執行自動化時以 InstanceID 參數的值為目標，則 Systems Manager 會針對每個指定的執行個體 ID 值執行子自動化。當自動化完成執行每個指定的執行個體，或是自動化失敗，父自動化就會完成。您最多可將 50 個參數值設為目標。如需有關在 Runbook 中設定參數值目標的詳細資訊，請參閱 將參數值設為目標。

7. 在 Input Parameters (輸入參數) 部分，指定所需的輸入參數。

   如果選擇使用標籤或資源群組將資源設為目標，您可能不必選擇 Input parameters (輸入參數) 部分中的某些選項。例如，假設您選擇 `AWS-RestartEC2Instance` Runbook，而您選擇使用標籤將執行個體設為目標，則您不必在 Input parameters (輸入參數) 部分中指定或選擇執行個體 ID。自動化會使用您指定的標籤找出要重新啟動的執行個體。

> **Important**
>
> 您必須在 AutomationAssumeRole 欄位中指定角色 ARN。 State Manager會使用 擔任角色來呼叫 Runbook 中 AWS 服務 指定的 ，並代表您執行自動化關聯。

8. 在 Specify schedule (指定排程) 部分，如果想以固定間隔執行關聯，請選擇 On Schedule (依排程)。如果選擇此選項，然後使用 Cron 或 Rate 運算式，利用提供的選項建立排程。如需 State Manager 適用的 Cron 和 Rate 運算式的詳細資訊，請參閱關聯的 Cron 與 Rate 運算式。

> **Note**
>
> Rate 運算式是執行 Runbook 的 State Manager 關聯所偏好的排程機制。Rate 運算式可在您達到自動化同時執行最大數量下提供更多執行關聯的靈活性。使用速率排程時，Systems Manager 便能在收到同時自動化已達最大值且遭調節的通知後立即重試自動化。

如果您希望執行一次關聯，請選擇 No schedule (無排程)。

9. （選用）在速率控制區段中，選擇並行和錯誤閾值選項，以控制跨 AWS 資源的自動化部署。

   a. 在 Concurrency (並行) 部分，選擇一個選項：

      - 選擇 targets (目標)，輸入可以同時執行自動化的目標絕對數量。
      - 選擇 percentage (百分比)，輸入可以同時執行自動化的目標集百分比。

   b. 在 Error threshold (錯誤閾值) 部分，選擇一個選項：

      - 選擇 errors (錯誤)，輸入在 Automation 停止傳送自動化至其他資源之前允許的錯誤絕對數量。
      - 選擇 percentage (百分比)，輸入在 Automation 停止傳送自動化至其他資源之前允許的錯誤百分比。

   如需有關使用目標和速率控制與自動化的詳細資訊，請參閱大規模執行自動化操作。

10. 選擇 Create Association (建立關聯)。

> **Important**
>
> 建立關聯時，關聯便會立即在指定的目標上執行。接著關聯會依照您選擇的 Cron 或 Rate 運算式執行。如果您選擇 No schedule (無排程)，關聯不會再次執行。

建立執行自動化的關聯 (命令列)

下列程序說明如何使用 AWS CLI （在 Linux 或 Windows 上） 或 AWS Tools for PowerShell 建立執行自動化的State Manager關聯。

開始之前

在完成下列程序之前，請確定您已建立 IAM 服務角色，其中包含執行 Runbook 所需的許可，並設定 Automation 的信任關係，這是其中的工具 AWS Systems Manager。如需詳細資訊，請參閱任務 1：建立自動化的服務角色。

建立執行自動化的關聯

1. AWS Tools for PowerShell如果您尚未安裝和設定 AWS CLI 或 。

   如需相關資訊，請參閱安裝或更新 AWS CLI的最新版本和安裝 AWS Tools for PowerShell。

2. 執行以下命令來檢視文件清單。

   Linux & macOS

   ```
   aws ssm list-documents
   ```

   Windows

   ```
   aws ssm list-documents
   ```

   PowerShell

   ```
   Get-SSMDocumentList
   ```

   記下您要用於關聯的 Runbook 名稱。

3. 執行以下命令來檢視您建立的 Runbook 詳細資訊。在下列命令中，用您自己的資訊取代 *Runbook name* (Runbook 名稱)。

   Linux & macOS

   ```
   aws ssm describe-document \
   --name runbook name
   ```

   記下您希望用於 `--automation-target-parameter-name` 選項的參數名稱 (例如 `InstanceId`)。此參數會決定自動化執行的資源類型。

   Windows

   ```
   aws ssm describe-document ^
   --name runbook name
   ```

   記下您希望用於 `--automation-target-parameter-name` 選項的參數名稱 (例如 `InstanceId`)。此參數會決定自動化執行的資源類型。

   PowerShell

   ```
   Get-SSMDocumentDescription `
   -Name runbook name
   ```

   記下您希望用於 `AutomationTargetParameterName` 選項的參數名稱 (例如 `InstanceId`)。此參數會決定自動化執行的資源類型。

4. 使用 State Manager 關聯建立執行自動化的命令。將每個`########`取代為您自己的資訊。

   「使用標籤設定目標」

   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 建立關聯 AWS CLI，請使用 `--targets` 參數來鎖定關聯執行個體。請勿使用 `--instance-id` 參數。`--instance-id` 參數是舊參數。

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 建立關聯 AWS CLI，請使用 `--targets` 參數來鎖定關聯執行個體。請勿使用 `--instance-id` 參數。`--instance-id` 參數是舊參數。

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 建立關聯 AWS Tools for PowerShell，請使用 `Target` 參數來鎖定關聯執行個體。請勿使用 `InstanceId` 參數。`InstanceId` 參數是舊參數。

「使用參數值設定目標」

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 鎖定目標 AWS Resource Groups

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"
```

鎖定多個帳戶和區域

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"]
```

命令會傳回與以下相似的新關聯詳細資訊：

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**
>
> 如果您使用標籤在一或多個目標執行個體上建立關聯，然後從執行個體移除標籤，則該執行個體將不再執行該關聯。系統會從State Manager文件中取消該執行個體的關聯。

疑難排解由 State Manager 關聯執行的自動化

Systems Manager Automation 強制執行 100 個並行自動化，以及每個區域每個帳戶 1,000 個排入佇列的自動化限制。如果使用 Runbook 的 State Manager 關聯顯示 Failed (失敗) 狀態和 AutomationExecutionLimitExceeded 詳細狀態，則表示可能已達到自動化的限制。因此，Systems Manager 會調節自動化。要解決此問題，請依照下列步驟：

- 為您的關聯使用不同的 Rate 或 Cron 運算式。例如，如果關聯排程每隔 30 分鐘執行，則變更運算式可每一或二小時執行一次。
- 刪除狀態為 Pending (待定) 的現有自動化。透過刪除這些自動化，即可清除目前的佇列。

使用維護時段排定自動化

您可以將 Runbook 設為維護時段的已註冊任務，以啟動自動化。透過將 Runbook 註冊為已註冊任務，維護時段便能在排程的維護時段期間執行自動化。

例如，假設您建立了名為 `CreateAMI` 的 Runbook，該 Runbook 會建立註冊為維護時段目標的執行個體 Amazon Machine Image (AMI)。若要指定 `CreateAMI` Runbook (和對應的自動化) 做為維護時段的已註冊任務，您必須先建立維護時段和註冊目標。然後您可以使用以下程序來指定 `CreateAMI` 文件做為維護時段內的已註冊任務。當維護時段在排程的期間啟動時，系統將執行自動化，並建立已註冊目標的 AMI。

如需建立 Automation Runbook 的資訊，請參閱 建立您自己的執行手冊。自動化是 中的工具 AWS Systems Manager。

使用下列程序，使用 AWS Systems Manager 主控台、 AWS Command Line Interface (AWS CLI) 或將自動化設定為維護時段的註冊任務 AWS Tools for Windows PowerShell。

向維護時段註冊自動化任務 (主控台)

以下程序會說明如何使用 Systems Manager 主控台將自動化設為維護時段的已註冊任務。

開始之前

您必須先建立維護時段並註冊至少一個目標，才能完成以下程序。如需詳細資訊，請參閱下列程序：

- 使用主控台建立維護時段.
- 使用主控台將目標指派至維護時段

將自動化設為維護時段的已註冊任務

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在左側導覽窗格中，選擇 Maintenance Windows，然後選擇您希望註冊自動化任務的維護時段。
3. 選擇動作。然後選擇 Register Automation task (註冊自動化任務)，使用 Runbook 在目標上執行您所選的自動化。
4. 在 Name (名稱) 中，輸入任務的名稱。
5. 在描述中，輸入描述。
6. 在 Document (文件) 中，選擇定義要執行任務的 Runbook。
7. 在 Document Version (文件版本) 中，選擇要使用的 Runbook 版本。
8. 在 Task priority (任務優先順序) 中，為此任務選擇優先順序。1 是最高優先順序。維護時段內的任務都是以優先順序來排程；相同優先順序的任務會平行排程。
9. 在 Targets (目標) 區段中，如果您選擇的 Runbook 是在資源中執行任務中的一個，手動指定標籤或選取執行個體，以識別您要執行這項自動化的目標。

   > **Note**
   >
   > 如果您想要透過輸入參數而非目標傳遞資源，則不需要指定維護時段目標。
   > 在許多情況下，您不需要明確指定自動化任務的目標。例如，假設您正在建立 Automation 類型任務來使用 `AWS-UpdateLinuxAmi` Runbook 更新 Linux 的 Amazon Machine Image (AMI)。當任務執行時，AMI 已更新為可用的最新版本 Linux 發行版本套件和 Amazon 軟體。從 AMI 建立的新執行個體已經安裝這些更新。因為在 Runbook 的輸入參數中指定了要更新的 AMI ID，所以不需要在維護時段任務中再次指定目標。

   如需不需要目標之維護時段任務的相關資訊，請參閱 the section called “註冊不含目標的維護時段任務”。
10. (選用) 在 Rate control (速率控制) 中：

> **Note**
>
> 如果您正在執行的任務未指定目標，則不需要指定速率控制。

- 在 Concurrency (並行) 中，指定可同時執行自動化的目標數目或百分比。

  如果您已透過選擇標籤鍵值對來選取目標，而且不確定有多少目標會使用所選的標籤，請指定百分比來限制可同時執行的自動化數目。

  執行維護時段時，便會針對每個目標啟動新的自動化。每個 AWS 帳戶有 100 的並行自動化上限。如果您指定大於 100 的並行速率，超過 100 的並行自動化會自動加入到自動化佇列。如需相關資訊，請參閱《Amazon Web Services 一般參考》中的 Systems Manager 服務配額一節。

- 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比目標上的自動化失敗後，停止在其他目標上執行。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止執行自動化。仍在處理自動化的目標也可能傳送錯誤。

11. 在 Input Parameters (輸入參數) 區段中，指定 Runbook 的參數。對於 Runbook，系統會自動填入一些值。您可以保留或取代這些值。

> **Important**
>
> 針對 Runbook，您可以選擇性指定自動化取得角色。若您沒有為此參數指定角色，自動化將取得您在步驟 11 中選擇的維護時段服務角色。因此，您必須確保您選擇的維護時段服務角色具有適當的 AWS Identity and Access Management (IAM) 許可，以執行 Runbook 中定義的動作。
> 例如，Systems Manager 的服務連結角色不具備 IAM 許可 `ec2:CreateSnapshot`，該許可是執行 Runbook `AWS-CopySnapshot` 所需要的許可。在此案例中，您必須使用自訂的維護時段服務角色，或指定具備 `ec2:CreateSnapshot` 許可的自動化取得角色。如需相關資訊，請參閱 設定自動化。

12. 在 IAM service role (IAM 服務角色) 區域，選擇角色以提供授權給 Systems Manager 並開始自動化。

    若要建立服務角色給維護視窗工作，請參閱 設定 Maintenance Windows。

13. 選擇 Register Automation task (註冊自動化任務)。

向維護時段註冊自動化任務 (命令列)

下列程序說明如何使用 AWS CLI （ 在 Linux 或 Windows 上 ） 或 AWS Tools for PowerShell ，將自動化設定為維護時段的已註冊任務。

開始之前

您必須先建立維護時段並註冊至少一個目標，才能完成以下程序。如需詳細資訊，請參閱下列程序：

- 步驟 1：使用 建立維護時段 AWS CLI.
- 步驟 2：使用 向維護時段註冊目標節點 AWS CLI

將自動化設為維護時段的已註冊任務

1. AWS Tools for PowerShell如果您尚未安裝和設定 AWS CLI 或 。

   如需相關資訊，請參閱安裝或更新 AWS CLI的最新版本和安裝 AWS Tools for PowerShell。
2. 建立命令，將自動化設為維護時段的已註冊任務。將每個########取代為您自己的資訊。

   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**
>
> 如果您使用 將自動化設定為已註冊的任務 AWS CLI，請使用 --Task-Invocation-Parameters 參數來指定參數，以在任務執行時傳遞給該任務。請勿使用 --Task-Parameters 參數。--Task-Parameters 參數是舊參數。
> 對於未指定目標的維護時段任務，您無法提供 --max-errors 和 --max-concurrency 的值。系統會插入預留位置值 1，這可能會在回應指令 (例如 describe-

maintenance-window-tasks 和 get-maintenance-window-task) 中回報。這些值不會影響任務的執行，可以忽略。
如需不需要目標之維護時段任務的相關資訊，請參閱 註冊不含目標的維護時段任務。

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

如果您使用 將自動化設定為已註冊的任務 AWS CLI，請使用 `--task-invocation-parameters` 參數來指定參數，以在任務執行時傳遞給該任務。請勿使用 `--task-parameters` 參數。`--task-parameters` 參數是舊參數。
對於未指定目標的維護時段任務，您無法提供 `--max-errors` 和 `--max-concurrency` 的值。系統會插入預留位置值 1，這可能會在回應指令 (例如 describe-maintenance-window-tasks 和 get-maintenance-window-task) 中回報。這些值不會影響任務的執行，可以忽略。
如需不需要目標之維護時段任務的相關資訊，請參閱 註冊不含目標的維護時段任務。

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**
>
> 如果您使用 將自動化設定為已註冊的任務 AWS Tools for PowerShell，請使用 -Automation_Parameter 參數來指定參數，以在任務執行時傳遞至任務。請勿使用 -TaskParameters 參數。-TaskParameters 參數是舊參數。
> 對於未指定目標的維護時段任務，您無法提供 -MaxError 和 -MaxConcurrency 的值。相反地，系統會插入預留位置值 1，這可能會在回應指令 (例如 Get-SSMMaintenanceWindowTaskList 和 Get-SSMMaintenanceWindowTask) 中回報。這些值不會影響任務的執行，可以忽略。
> 如需不需要目標之維護時段任務的相關資訊，請參閱 註冊不含目標的維護時段任務。

以下範例會將自動化設為維護時段的已註冊任務，其優先順序為 1。它還演示了為無目標維護時段任務省略的 --targets、--max-errors 和 --max-concurrency 選項。自動化會使用 AWS-StartEC2Instance Runbook 和指定的自動化取得角色，來啟動已向維護時段註冊為目標的 EC2 執行個體。維護時段在任何指定時間最多可以同時在 5 個執行個體上執行自動化。此外，如果錯誤計數超過 1 個，已註冊任務會在特定的間隔內於更多執行個體上停止執行。

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
```

命令會傳回新已註冊任務的詳細資訊，該資訊與以下相似：

Linux & macOS

```
{
"WindowTaskId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE"
}
```

Windows

```
{
"WindowTaskId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE"
}
```

PowerShell

```
4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE
```

3. 若要檢視已註冊的任務，請執行以下命令。把*#### ID* 取代為您自己的資訊。

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
```

系統會傳回相關資訊，如下所示。

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 Automation 動作參考

此參考描述您可在 Automation Runbook 中指定的自動化動作。自動化是 中的工具 AWS Systems Manager。這些動作無法用於其他類型的 Systems Manager (SSM) 文件中。如需其他 SSM 文件類型的外掛程式詳細資訊，請參閱 命令文件外掛程式參考。

Systems Manager Automation 會執行 Automation Runbook 中定義的步驟。每個步驟皆與一個特定動作關聯。動作會決定輸入、行為和步驟的輸出。Runbook 的 `mainSteps` 章節中會定義步驟。

您不需要指定動作或步驟的輸出。輸出是由與步驟關聯的動作預先定義。在 Runbook 中指定步驟輸入時，您可以參考先前步驟的一個或多個輸出。例如，您可以讓 `aws:runInstances` 輸出用於後續的 `aws:runCommand` 動作。您也可以參考 Runbook `Output` 區段中先前步驟的輸出。

> **Important**
>
> 如果您執行可使用 AWS Identity and Access Management (IAM) 服務角色叫用其他服務的自動化工作流程，請注意您必須為該服務角色設定可叫用這些服務的許可。此要求適用於所有 AWS Automation Runbook (`AWS-*` Runbook)，例如 `AWS-ConfigureS3BucketLogging`、`AWS-CreateDynamoDBBackup` 和 `AWS-RestartEC2Instance` Runbook 等。此要求也適用於您建立的任何自訂自動化執行手冊， AWS 服務 這些執行手冊使用呼叫其他服務的動作來叫用其他 。例如，如果您使用 `aws:executeAwsApi`、`aws:createStack` 或 `aws:copyImage` 動作，為服務角色設定可叫用這些服務的許可。您可以將 IAM 內嵌政策新增至角色， AWS 服務 以將許可授予其他 。如需詳細資訊，請參閱（選用）新增自動化內嵌政策或客戶受管政策以叫用其他政策 AWS 服務。

主題

- 依所有動作共用的屬性
- aws:approve – 暫停自動化以進行手動核准
- aws:assertAwsResourceProperty – 宣告 AWS 資源狀態或事件狀態
- aws:branch – 執行條件式自動化步驟
- aws:changeInstanceState – 變更或宣告執行個體狀態
- aws:copyImage – 複製或加密 Amazon Machine Image
- aws:createImage – 建立 Amazon Machine Image
- aws:createStack – 建立 AWS CloudFormation 堆疊

- aws:createTags – 建立 AWS 資源的標籤
- aws:deleteImage – 刪除 Amazon Machine Image
- aws:deleteStack – 刪除 AWS CloudFormation 堆疊
- aws:executeAutomation – 執行另一項自動化
- aws:executeAwsApi – 呼叫並執行 AWS API 操作
- aws:executeScript – 執行指令碼
- aws:executeStateMachine – 執行 AWS Step Functions 狀態機器
- aws:invokeWebhook：叫用 Automation Webhook 整合
- aws:invokeLambdaFunction – 叫用 AWS Lambda 函數
- aws:loop - 迭代自動化中的步驟
- aws:pause – 暫停自動化
- aws:runCommand – 在受管執行個體上執行命令
- aws:runInstances – 啟動 Amazon EC2 執行個體。
- aws:sleep – 延遲自動化
- aws:updateVariable - 更新執行手冊變數的值
- aws:waitForAwsResourceProperty – 等待 AWS 資源屬性
- 自動化系統變數

依所有動作共用的屬性

一般屬性是可在所有動作中找到的參數或選項。某些選項會定義步驟的行為，例如，等待步驟完成的時間，以及如果步驟失敗時該怎麼做。以下為所有動作中常見的屬性。

description

您提供用於描述執行手冊或步驟之目的的資訊。

類型：字串

必要：否

name

一種識別符，在 Runbook 的所有步驟名稱中必須獨一無二。

類型：字串

允許的模式：[a-zA-Z0-9_]+$

必要：是

action

步驟要執行的動作名稱。aws:runCommand – 在受管執行個體上執行命令 是您可以在此指定的動作範例。此文件提供所有可用動作的詳細資訊。

類型：字串

必要：是

maxAttempts

步驟在故障時應重試的次數。如果值大於 1，則在所有重試嘗試失敗之前，步驟不會視為失敗。預設值為 1。

類型：整數

必要：否

timeoutSeconds

步驟的逾時值。如果達到逾時且 maxAttempts 的值大於 1，則在嘗試完所有重試之前，步驟不會視為逾時。

類型：整數

必要：否

onFailure

指示自動化在失敗時應中止、繼續或前往不同的步驟。此選項的預設值為 abort。

類型：字串

有效值：Abort | Continue | step:*step_name*

必要：否

onCancel

指出當使用者取消自動化時，自動化應移至哪個步驟。自動化會執行取消工作流程最多兩分鐘。

類型：字串

有效值：Abort | step:*step_name*

必要：否

onCancel 屬性不支援移至下列動作：

- aws:approve
- aws:copyImage
- aws:createImage
- aws:createStack
- aws:createTags
- aws:loop
- aws:pause
- aws:runInstances
- aws:sleep

isEnd

此選項會在特定步驟結束時停止自動化。如果步驟執行失敗或成功，自動化就會停止。預設值為 false。

類型：布林值

有效值：true | false

必要：否

nextStep

指定在成功完成步驟後要接著處理自動化中的哪個步驟。

類型：字串

必要：否

isCritical

指定某個步驟是成功完成自動化的關鍵。如果此指定步驟失敗，則自動化會將自動化的最終狀態回報為 Failed (失敗)。只有當您在步驟中明確定義此屬性時，才會評估此屬性。如果 onFailure 屬性在步驟中已設為 Continue，則該值預設為 false。否則，此選項的預設值為 true。

類型：布林值

有效值：true | false

必要：否

inputs

專屬於動作的屬性。

類型：映射

必要：是

範例

```
---
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` – 暫停自動化以進行手動核准

暫時暫停自動化，直到指定的委託人核准或拒絕動作。達到所需的核准數量後，自動化會繼續。您可以將核准步驟插入 Runbook `mainSteps` 章節的任何地方。

> **Note**
>
> 此動作不支援多帳戶和區域自動化。此動作的預設逾時時間為 7 天 (604800 秒)，最大值為 30 天 (2592000 秒)。您可以透過指定 aws:approve 步驟的 timeoutSeconds 參數來限制或延長逾時。

在以下範例中，aws:approve 動作暫時暫停了自動化，直到核准者接受或拒絕自動化為止。核准後，自動化會執行簡單的 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

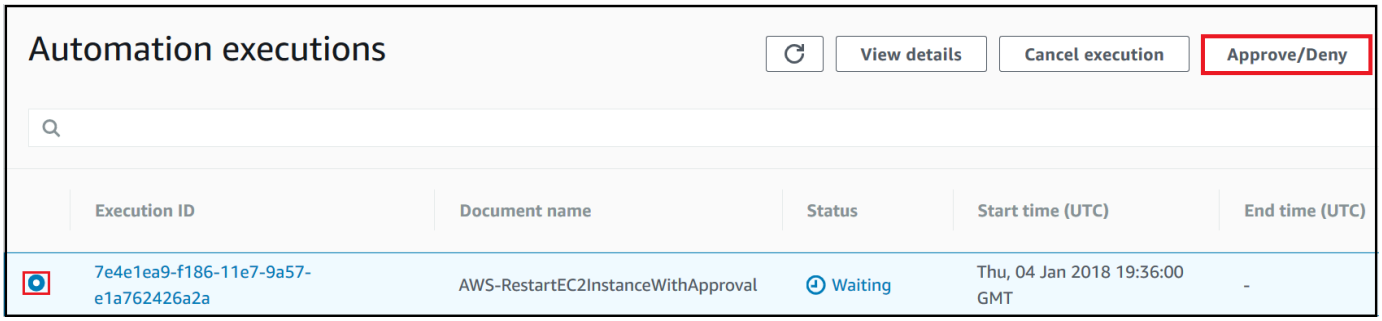
```
{
   "description":"RunInstancesDemo1",
   "schemaVersion":"0.3",
   "assumeRole":"{{ assumeRole }}",
   "parameters":{
      "assumeRole":{
         "type":"String"
      },
      "message":{
         "type":"String"
      }
   },
   "mainSteps":[
      {
         "name":"approve",
         "action":"aws:approve",
         "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"
               ]
            }
         }
```

```
        }
    ]
}
```

您可以核准或拒絕在主控台等待核准的自動化。

核准或拒絕等待自動化

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Automation (自動化)。
3. 選擇狀態為 Waiting (正在等待) 自動化旁的選項。



4. 選擇 Approve/Deny (核准/拒絕)。
5. 檢閱自動化的詳細資訊。
6. 選擇 Approve (核准) 或 Deny (拒絕)，輸入選擇性的註解，接著選擇 Submit (提交)。

輸入範例

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

適用於 Automation 核准的 Amazon Simple Notification Service (Amazon SNS) 主題 Amazon Resource Name (ARN)。當您在 Runbook 中指定 `aws:approve` 步驟，自動化會傳送訊息至此主題，讓委託人知道必須核准或拒絕自動化步驟。Amazon SNS 主題的標題必須以「Automation」為字首。

類型：字串

必要：否

訊息

您想要在核准請求傳送時包含於 Amazon SNS 主題的資訊。訊息長度上限為 4096 個字元。

類型：字串

必要：否

MinRequiredApprovals

繼續自動化所需的核准數量下限。如果您不指定值，系統會預設一個。此參數的值必須為正數。此參數的值不得超過由 `Approvers` 參數定義的核准者數量。

類型：整數

必要：否

Approvers

能夠核准或拒絕動作的已 AWS 驗證主體清單。核准者的數量上限為 10。您可以使用以下任一格式指定委託人：

- 使用者名稱
- 使用者 ARN
- IAM 角色 ARN
- IAM 擔任角色 ARN

類型：StringList

必要：是

EnhancedApprovals

此輸入僅用於 Change Manager 範本。能夠核准或拒絕動作、IAM 主體類型以及核准者人數下限的已驗證 AWS 主體清單。以下是範例：

```
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
```

類型：StringList

必要：是

輸出

ApprovalStatus

步驟的核准狀態。狀態可以是以下其中一項：Approved (核准)、Rejected (拒絕) 或 Waiting (等待)。等待表示自動化要等待核准者輸入。

類型：字串

ApproverDecisions

一種 JSON 對應，包括每個核准者的核准決定。

類型：MapList

aws:assertAwsResourceProperty – 宣告 AWS 資源狀態或事件狀態

aws:assertAwsResourceProperty 動作可讓您針對特定自動化步驟宣告特定的資源狀態或事件狀態。

如需有關如何使用此動作的更多範例，請參閱 其他執行手冊範例。

輸入

輸入是由您選擇的 API 操作定義。

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"
    ]
  }
}
```

服務

包含您要執行之 API 操作的 AWS 服務 命名空間。例如，Systems Manager 的命名空間為 `ssm`。Amazon EC2 的命名空間為 `ec2`。您可以檢視AWS CLI 命令參考可用服務章節中的受支援 AWS 服務 命名空間。

類型：字串

必要：是

Api

您想要執行的 API 操作之名稱。您可以檢視 API 操作 (也稱為方法)，方式是在以下服務參考頁面的左側導覽中選擇一項服務。在您想要呼叫之服務的 Client (用戶端) 部分選擇一個方法。例如，Amazon Relational Database Service (Amazon RDS) 的所有 API 操作 (方法) 均列於以下頁面：Amazon RDS 方法。

類型：字串

必要：是

API 操作輸入

一個或多個 API 操作輸入。您可以檢視可用的輸入 (也稱為參數)，方式是在以下服務參考頁面的左側導覽中選擇一項服務。在您想要呼叫之服務的 Client (用戶端) 部分選擇一個方法。例

如，Amazon RDS 的所有方法均列於以下頁面：Amazon RDS 方法。選擇 describe_db_instances 方法並向下捲動以查看可用的參數，例如 DBInstanceIdentifier、Name、Values。使用以下格式指定一個以上的輸入。

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"
}
```

類型：由所選的 API 操作決定

必要：是

PropertySelector

回應物件中特定屬性的 JSONPath。您可以檢視回應物件，方式是在以下服務參考頁面的左側導覽中選擇一項服務。在您想要呼叫之服務的 Client (用戶端) 部分選擇一個方法。例如，Amazon RDS 的所有方法均列於以下頁面：Amazon RDS 方法。選擇 describe_db_instances 方法並向下捲動至 Response Structure (回應結構) 區段。DBInstances 列為回應物件。

類型：字串

必要：是

DesiredValues

讓自動化繼續的預期狀態。如果指定布林值，您必須使用大寫字母，例如 True 或 False。

類型：StringList

必要：是

aws:branch – 執行條件式自動化步驟

aws:branch 動作可讓您建立動態自動化，以評估單一步驟中的不同選擇，接著根據該評估的結果跳至 Runbook 中的不同步驟。

指定步驟的 aws:branch 動作時，您要指定自動化必須評估的 Choices。Choices 可根據您在 Runbook 之 Parameters 區段所指定的值，或是產生做為先前步驟之輸出的動態值。自動化會使用布林值表達式評估每個選擇。如果第一個選擇為 true，則自動化會跳至針對該選擇指定的步驟。如果第一個選擇為 false，則自動化會評估下一個選擇。自動化會繼續評估每個選擇，直到處理的選擇是 true 為止。接著自動化會跳至選擇為 true 的指定步驟。

如果選擇均不為 true，則自動化會檢查步驟是否包含 default 值。如果沒有選擇為 true，則預設值會定義自動化應跳至的步驟。如果未針對步驟指定 default 值，則自動化會處理 Runbook 中的下一個步驟。

aws:branch 動作可結合使用 And、Not、Or 運算子來支援複雜的選擇評估。如需使用 aws:branch 的詳細資訊，包括使用不同運算子的範例 Runbook 和範例，請參閱 在執行手冊中使用條件陳述式。

輸入

在步驟中指定一個或多個 Choices。Choices 可根據您在 Runbook 之 Parameters 區段所指定的值，或是產生做為先前步驟之輸出的動態值。以下為可評估參數的 YAML 範例。

```
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
```

以下為可評估先前步驟之輸出的 YAML 範例。

```
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
```

選擇

在決定要處理的下一個步驟時，自動化應評估的一個或多個運算式。選擇是使用布林值運算式評估。每個選擇都必須定義以下選項：

- NextStep：如果指定選擇為 true，在 Runbook 中要處理的下一個步驟。
- Variable (變數)：指定 Runbook 之 `Parameters` 區段中定義的參數名稱。或指定 Runbook 中先前步驟的輸出物件。如需為 `aws:branch` 建立變數的詳細資訊，請參閱 關於建立輸出變數。
- Operation (運算)：用於評估選擇的條件。`aws:branch` 動作支援以下運算：

  字串運算

  - StringEquals
  - EqualsIgnoreCase
  - StartsWith
  - EndsWith
  - 包含

  數值運算

  - NumericEquals
  - NumericGreater
  - NumericLesser
  - NumericGreaterOrEquals

- NumericLesser
- NumericLesserOrEquals

布林運算

- BooleanEquals

> **Important**
>
> 當您建立 Runbook 時，系統會驗證 Runbook 中的每個操作。如果不支援操作，系統會在您嘗試建立 Runbook 時傳回錯誤。

預設

自動化在 Choices 均不為 true 時應跳至某步驟的名稱。

類型：字串

必要：否

> **Note**
>
> aws:branch 動作支援 And、Or、Not 運算子。如需使用運算子的 aws:branch 範例，請參閱 在執行手冊中使用條件陳述式。

aws:changeInstanceState – 變更或宣告執行個體狀態

變更或宣告執行個體的狀態。

此動作可用於宣告模式 (不執行 API 以變更狀態，但會驗證執行個體處於所需的狀態)。若要使用宣告模式，請將 CheckStateOnly 參數設定為 true。此模式可用於在 Windows 上執行 Sysprep 命令，這是一種非同步命令，可在背景長時間執行。您可以確保執行個體在您建立 Amazon Machine Image (AMI) 之前停止。

> **Note**
>
> 此動作的預設逾時值為 3600 秒 (1 小時)。您可以透過指定 aws:changeInstanceState 步驟的 timeoutSeconds 參數來限制或延長逾時。

輸入

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

執行個體的 ID。

類型：StringList

必要：是

CheckStateOnly

如果為 false，則會將執行個體狀態設為所需的狀態。如果為 true，則會使用輪詢宣告所需的狀態。

預設：`false`

類型：布林值

必要：否

DesiredState

所需的狀態。設為 `running` 時，則在完成之前，此動作會等待 Amazon EC2 狀態成為 `Running`，執行個體狀態成為 `OK`，系統狀態成為 `OK`。

類型：字串

有效值：`running` | `stopped` | `terminated`

必要：是

Force

如果設定，則會強制執行個體停止。執行個體沒有機會排清檔案系統快取或檔案系統中繼資料。如果使用此選項，您必須執行檔案系統檢查及修復程序。此選項不建議用於 Windows Server 的 EC2 執行個體。

類型：布林值

必要：否

AdditionalInfo

預訂.

類型：字串

必要：否

輸出

無

`aws:copyImage` – 複製或加密 Amazon Machine Image

AWS 區域 將 Amazon Machine Image(AMI) 從任何 複製到目前的 區域。此動作也可以加密新的 AMI。

輸入

此動作支援大部分的 `CopyImage` 參數。如需詳細資訊，請參閱 CopyImage。

以下範例是在首爾區域建立 AMI 的複本 (SourceImageID: ami-0fe10819. SourceRegion: ap-northeast-2)。新 AMI 已複製到您啟動自動化動作的區域。由於選用的 Encrypted 旗標設為 true，因此複製的 AMI 都會加密。

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

來源 AMI 存在的區域。

類型：字串

必要：是

SourceImageId

要從來源區域複製的 AMI ID。

類型：字串

必要：是

ImageName

新映像的名稱。

類型：字串

必要：是

ImageDescription

目標映像的描述。

類型：字串

必要：否

Encrypted

加密目標 AMI。

類型：布林值

必要：否

KmsKeyId

在複製操作期間加密映像快照時， AWS KMS key 要使用的完整 Amazon Resource Name (ARN)。如需詳細資訊，請參閱 CopyImage。

類型：字串

必要：否

ClientToken

唯一且區分大小寫的識別符，由您提供以確保請求的冪等。如需詳細資訊，請參閱 CopyImage。

類型：字串

必要：否

輸出

ImageId

所複製映像的 ID。

ImageState

所複製映像的狀態。

有效值：`available` | `pending` | `failed`

`aws:createImage` – 建立 Amazon Machine Image

從正在執行、正在停止或已停止以及 `ImageState` 輪詢狀態為 `available` 的執行個體建立 Amazon Machine Image (AMI)。

輸入

此動作支援下列 `CreateImage` 參數。如需詳細資訊，請參閱 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

執行個體的 ID。

類型：字串

必要：是

ImageName

映像的名稱。

類型：字串

必要：是

ImageDescription

映像的描述。

類型：字串

必要：否

NoReboot

布林值常值。

根據預設，Amazon Elastic Compute Cloud (Amazon EC2) 會在建立映像之前嘗試關閉並重新開機執行個體。若 No Reboot (不重新啟動) 選項設為 `true`，則 Amazon EC2 在建立映像之前不會關閉執行個體。使用此選項時，無法保證所建立映像的檔案系統完整性。

如果您不希望執行個體在您從其建立 AMI 映像後執行，首先請使用 [aws:changeInstanceState – 變更或宣告執行個體狀態] 動作停止執行個體，接著使用 `aws:createImage` 動作並將 NoReboot 選項設為 `true`。

類型：布林值

必要：否

BlockDeviceMappings

執行個體的區塊型儲存設備。

類型：映射

必要：否

輸出

ImageId

新建立映像的 ID。

類型：字串

ImageState

映像目前的狀態。如果狀態可用，則表示已成功註冊映像，且該映像可以用來啟動執行個體。

類型：字串

`aws:createStack` – 建立 AWS CloudFormation 堆疊

從範本建立 AWS CloudFormation 堆疊。

如需建立 CloudFormation 堆疊的補充資訊，請參閱《AWS CloudFormation API 參考》中的 CreateStack。

輸入

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

在 CloudFormation 可以建立特定堆疊之前，您所指定之值的清單。有些堆疊範本包含的資源可能會影響 中的許可 AWS 帳戶。對於這些堆疊，您必須藉由指定此參數明確地確認其功能。

有效值包括 CAPABILITY_IAM、CAPABILITY_NAMED_IAM 與 CAPABILITY_AUTO_EXPAND。

CAPABILITY_IAM 和 CAPABILITY_NAMED_IAM

如果您有 IAM 資源，您可以指定其中一個功能。如果您有自訂名稱的 IAM 資源，則您必須指定 CAPABILITY_NAMED_IAM。如果您不指定此參數，此動作會傳回 InsufficientCapabilities 錯誤。下列資源需要您指定 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

如果您的堆疊範本包含這些資源，建議您檢閱與其關聯的所有許可並視需要編輯其許可。

如需詳細資訊，請參閱 AWS CloudFormation 範本中的確認 IAM 資源。

CAPABILITY_AUTO_EXPAND

某些範本包含巨集。巨集會在範本上執行自訂處理，範圍涵蓋尋找和取代操作這種簡單動作，乃至於全部範本的大規模轉換。因此，使用者通常會從已處理的範本建立變更集，以在實際建立堆疊前先檢閱巨集產生的變更。如果堆疊範本包含一或多個巨集，且您選擇直接從已處理的範本建立堆疊，而不先檢閱變更集中所產生的變更，則您必須認可此功能。

如需詳細資訊，請參閱AWS CloudFormation 《 使用者指南》中的使用 AWS CloudFormation 巨集在範本上執行自訂處理。

類型：字串的陣列

有效值:CAPABILITY_IAM | CAPABILITY_NAMED_IAM | CAPABILITY_AUTO_EXPAND

必要：否

ClientRequestToken

此 CreateStack 請求的唯一識別符。如果您將此步驟中的 maxAttempts 設為大於 1 之值，請指定此字符。藉由指定此字符，CloudFormation 會知道您並非嘗試以相同名稱建立新的堆疊。

類型：字串

必要：否

長度限制：長度下限為 1。長度上限為 128。

模式：[a-zA-Z0-9][-a-zA-Z0-9]*

DisableRollback

設為 `true` 以在堆疊建立失敗時關閉堆疊回復。

條件：您可以指定 `DisableRollback` 參數或 `OnFailure` 參數，但不能同時指定兩者。

預設：`false`

類型：布林值

必要：否

NotificationARNs

用來發佈堆疊相關事件的 Amazon Simple Notification Service (Amazon SNS) 主題 ARN。您可以使用 Amazon SNS 主控台尋找 SNS 主題 ARN，https://console.aws.amazon.com/sns/v3/home。

類型：字串的陣列

陣列成員：最多 5 個項目。

必要：否

OnFailure

決定堆疊建立失敗時要採取的動作。您必須指定 `DO_NOTHING`、`ROLLBACK` 或 `DELETE`。

條件：您可以指定 `OnFailure` 參數或 `DisableRollback` 參數，但不能同時指定兩者。

預設：`ROLLBACK`

類型：字串

有效值： `DO_NOTHING | ROLLBACK | DELETE`

必要：否

參數

針對堆疊指定輸入參數的 Parameter 結構清單。如需詳細資訊，請參閱 Parameter (參數) 資料類型。

類型：Parameter (參數) 物件的陣列

必要：否

ResourceTypes

針對此建立堆疊的動作，您有許可使用的範本資源類型。例如，AWS::EC2::Instance、AWS::EC2::* 或 Custom::*MyCustomInstance*。使用以下語法來描述範本資源類型。

- 對於所有 AWS 資源：

```
AWS::*
```

- 用於所有自訂資源：

```
Custom::*
```

- 用於特定自訂資源：

```
Custom::logical_ID
```

- 用於特定 AWS 服務的所有資源：

```
AWS::service_name::*
```

- 對於特定 AWS 資源：

```
AWS::service_name::resource_logical_ID
```

如果資源類型的清單不包括您正在建立的資源，則堆疊建立會失敗。根據預設，CloudFormation 會授予所有資源類型的許可。IAM 針對 IAM 政策中的 CloudFormation 特定條件索引鍵使用此參數。如需詳細資訊，請參閱使用 控制存取 AWS Identity and Access Management。

類型：字串的陣列

長度限制：長度下限為 1。長度上限為 256。

必要：否

RoleARN

CloudFormation 假設建立堆疊之 IAM 角色的 Amazon Resource Name (ARN)。CloudFormation 會使用該角色的憑證代表您進行呼叫。CloudFormation 一律會將此角色用於堆疊上的所有未來操作。只要使用者擁有在堆疊上操作的許可，即使使用者沒有傳遞此角色的許可，CloudFormation 也會使用此角色。確保角色授予最少量的權限。

如果您不指定值，CloudFormation 會使用先前與堆疊關聯的角色。如果沒有可用的角色，CloudFormation 會使用從您的使用者登入資料產生的暫時工作階段。

類型：字串

長度限制：長度下限為 20。長度上限為 2048。

必要：否

StackName

與堆疊關聯的名稱。在您建立堆疊的區域中，名稱必須是唯一的。

> **Note**
>
> 堆疊名稱僅能使用英數字元 (區分大小寫) 和連字號。必須以字母字元開頭，且長度不可超過 128 個字元。

類型：字串

必要：是

StackPolicyBody

包含堆疊政策內文的結構。如需詳細資訊，請參閱避免更新堆疊資源。

條件：您可以指定 `StackPolicyBody` 參數或 `StackPolicyURL` 參數，但不能同時指定兩者。

類型：字串

長度限制：長度下限為 1。長度上限為 16384。

必要：否

StackPolicyURL

包含堆疊政策之檔案的位置。URL 指向的政策必須位於與堆疊在相同區域中的 S3 儲存貯體。堆疊政策允許的檔案大小上限為 16 KB。

條件：您可以指定 StackPolicyBody 參數或 StackPolicyURL 參數，但不能同時指定兩者。

類型：字串

長度限制：長度下限為 1。長度上限為 1350。

必要：否

標籤

要與此堆疊相關聯的索引鍵/值組。CloudFormation 也會將這些標籤散佈到堆疊中建立的資源。您最多可指定 10 個標籤。

類型：Tag (標籤) 物件的陣列

必要：否

TemplateBody

包含範本內文的結構，長度下限為 1 位元組，上限為 51,200 位元組。如需詳細資訊，請參閱範本剖析。

條件：您可以指定 TemplateBody 參數或 TemplateURL 參數，但不能同時指定兩者。

類型：字串

長度限制：長度下限為 1。

必要：否

TemplateURL

包含範本內文之檔案的位置。URL 必須指向位於 S3 儲存貯體的範本。範本允許的大小上限為 460,800 位元組。如需詳細資訊，請參閱範本剖析。

條件：您可以指定 TemplateBody 參數或 TemplateURL 參數，但不能同時指定兩者。

類型：字串

長度限制：長度下限為 1。長度上限為 1024。

必要：否

TimeoutInMinutes

在堆疊狀態成為 CREATE_FAILED 之前可經過的時間。如果 DisableRollback 未設定或設為 false，堆疊將會轉返。

類型：整數

有效範圍：最小值為 1。

必要：否

輸出

StackId

堆疊的唯一識別符。

類型：字串

StackStatus

堆疊的目前狀態。

類型：字串

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

必要：是

StackStatusReason

與堆疊狀態關聯的成功或失敗訊息。

類型：字串

必要：否

如需詳細資訊，請參閱 CreateStack。

安全考量

在您可以使用 aws:createStack 動作之前，您必須將以下政策指派至 IAM 自動化擔任角色。如需擔任角色的詳細資訊，請參閱 任務 1：建立自動化的服務角色。

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "sqs:*",
            "cloudformation:CreateStack",
            "cloudformation:DescribeStacks"
         ],
         "Resource":"*"
      }
   ]
}
```

aws:createTags – 建立 AWS 資源的標籤

為 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體或 AWS Systems Manager 受管執行個體建立新的標籤。

輸入

此動作支援大部分 Amazon EC2 CreateTags 和 Systems Manager AddTagsToResource 參數。如需詳細資訊，請參閱 CreateTags 和 AddTagsToResource。

以下範例示範如何將 Amazon Machine Image (AMI) 和執行個體標記為特定部門的生產資源。

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

要標籤的資源之 ID。如果資源類型不是「EC2」，則此欄位只能包含單一項目。

類型：字串清單

必要：是

標籤

要與資源建立關聯的標籤。

類型：對應清單

必要：是

ResourceType

要標籤的資源之類型。如果未提供，會使用「EC2」為預設值。

類型：字串

必要：否

有效值：EC2 | ManagedInstance | MaintenanceWindow | Parameter

輸出

無

aws:deleteImage – 刪除 Amazon Machine Image

刪除指定 Amazon Machine Image (AMI) 和所有相關的快照。

輸入

此動作僅支援一個參數。如需更多資訊，請參閱 DeregisterImage 和 DeleteSnapshot 文件。

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

欲刪除的映像之 ID。

類型：字串

必要：是

輸出

無

`aws:deleteStack` – 刪除 AWS CloudFormation 堆疊

刪除 AWS CloudFormation 堆疊。

輸入

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

`DeleteStack` 請求的唯一識別符。如果您打算重試請求，請指定此字符，讓 CloudFormation 知道您並非嘗試刪除名稱相同的堆疊。您可以重試 `DeleteStack` 請求以確認 CloudFormation 收到。

類型：字串

長度限制：長度下限為 1。長度上限為 128。

模式：[a-zA-Z][-a-zA-Z0-9]*

必要：否

RetainResources.member.N

此輸入僅適用處於 `DELETE_FAILED` 狀態的堆疊。您想要保留之資源的邏輯資源 ID 清單。在刪除期間，CloudFormation 會刪除堆疊，但不會刪除保留資源。

當您無法刪除資源 (例如非空的 S3 儲存貯體)，但想要刪除堆疊，保留資源會很有幫助。

類型：字串的陣列

必要：否

RoleARN

CloudFormation 建立堆疊時所擔任之 AWS Identity and Access Management (IAM) 角色的 Amazon Resource Name (ARN)。CloudFormation 會使用該角色的憑證代表您進行呼叫。CloudFormation 一律會將此角色用於堆疊上的所有未來操作。只要使用者擁有在堆疊上操作的許可，即使使用者沒有傳遞此角色的許可，CloudFormation 也會使用此角色。確保角色授予最少量的權限。

如果您不指定值，CloudFormation 會使用先前與堆疊關聯的角色。如果沒有可用的角色，CloudFormation 會使用從您的使用者登入資料產生的暫時工作階段。

類型：字串

長度限制：長度下限為 20。長度上限為 2048。

必要：否

StackName

與堆疊關聯的名稱或唯一的堆疊 ID。

類型：字串

必要：是

安全考量

在您可以使用 aws:deleteStack 動作之前，您必須將以下政策指派至 IAM 自動化擔任角色。如需擔任角色的詳細資訊，請參閱 任務 1：建立自動化的服務角色。

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "sqs:*",
            "cloudformation:DeleteStack",
            "cloudformation:DescribeStacks"
         ],
         "Resource":"*"
      }
   ]
}
```

aws:executeAutomation – 執行另一項自動化

透過呼叫次要 Runbook 執行次要自動化。透過此動作，您可以為您最常用的操作建立 Runbook，並在自動化執行期間參考那些 Runbook。此動作可簡化您的 Runbook，讓您不需要在類似的 Runbook 之間重複步驟。

次要自動化會在啟動主要自動化的使用者之內容中執行。這表示次要自動化使用與啟動第一個自動化的使用者相同的 AWS Identity and Access Management (IAM) 角色或使用者。

> **Important**
>
> 如果由您指定參數的次要自動化使用擔任角色 (使用 iam:passRole 政策的角色)，則啟動主要自動化的使用者或角色必須擁有將擔任角色傳遞至次要自動化的許可。如需設定自動化之擔任角色的詳細資訊，請參閱 使用主控台建立用於自動化的服務角色。

輸入

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

在步驟期間要執行的次要 Runbook 名稱。對於相同 中的 Runbook AWS 帳戶，指定 Runbook 名稱。對於從不同 共用的 Runbook AWS 帳戶，請指定 Runbook 的 Amazon Resource Name (ARN)。如需共用 Runbook 的詳細資訊，請參閱 使用共用的 SSM 文件。

類型：字串

必要：是

DocumentVersion

要執行的次要 Runbook 版本。如果未指定，自動化會執行預設 Runbook 版本。

類型：字串

必要：否

MaxConcurrency

您可以平行執行此任務的目標數目上限。您可以指定數量如 10 或百分比如 10%。

類型：字串

必要：否

MaxErrors

系統停止在其他目標上執行自動化前所允許的錯誤數量。您可以指定絕對數量的錯誤 (例如 10 個) 或目標集的百分比 (例如 10%)。例如，假設您指定 3，系統會在收到第四個錯誤時停止執行自動化。如果您指定 0，系統會在第一個錯誤結果傳回時停止其他目標上執行的自動化。如果您在 50 個資源上執行自動化，並將 MaxErrors 設為 10%，則系統會在收到第六個錯誤時停止對其他目標執行自動化。

達到 MaxErrors 閾值時已經在執行的自動化允許完成，但其中某些自動化也可能會失敗。如果您需要確保不會有比指定 MaxErrors 多的失敗自動化操作，則將 MaxConcurrency 設定為 1，因此會一次進行一個自動化。

類型：字串

必要：否

RuntimeParameters

次要 Runbook 的必要參數。映射使用以下格式：{"parameter1" : "value1", "parameter2" : "value2" }

類型：映射

必要：否

標籤

您指派給資源的選用中繼資料。您可以為自動化指定最多五個標籤。

類型：MapList

必要：否

TargetLocations

位置是您想要執行自動化的 AWS 區域 和/或 AWS 帳戶 的組合。必須指定最少 1 個項目，且最多可指定 100 個項目。指定此參數的值時，輸出不會傳回至父系自動化。如有需要，您必須對 API 操作進行後續呼叫，才能從子自動化擷取輸出。

類型：MapList

必要：否

TargetMaps

文件參數到目標資源的鍵/值映射清單。不能同時指定 `Targets` 和 `TargetMaps`。

類型：MapList

必要：否

TargetParameterName

作為速率控制自動化之目標資源使用的參數名稱。只在您指定 `Targets` 時需要

類型：字串

必要：否

目標

至目標資源的鍵/值對映清單。只在您指定 TargetParameterName 時需要

類型：MapList

必要：否

輸出

輸出

由次要自動化產生的輸出。您可以使用以下格式參考輸出：*Secondary_Automation_Step_Name*.Output

類型：StringList

請見此處範例：

```
- 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

次要自動化的 ID。

類型：字串

Status

次要自動化的 狀態。

類型：字串

`aws:executeAwsApi` – 呼叫並執行 AWS API 操作

呼叫並執行 AWS API 操作。大部分 API 操作均有支援，但並非所有 API 操作都經過測試。不支援串流 API 操作，例如 GetObject 操作。如果不確定想要使用的 API 操作是否為串流操作，請檢閱 Boto3 服務文件，以確定 API 是否需要串流輸入或輸出。我們會定期更新此動作所使用的 Boto3 版本。但在新的 Boto3 版本發布之後，可能需要長達幾週的時間才能在此動作中反映出相關更改。每個 `aws:executeAwsApi` 動作最多可執行 25 秒。如需有關如何使用此動作的更多範例，請參閱 其他執行手冊範例。

輸入

輸入是由您選擇的 API 操作定義。

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"
      }
   ]
}
```

服務

包含您要執行之 API 操作的 AWS 服務 命名空間。您可以在 的可用服務中檢視支援的 AWS 服務 命名空間清單 適用於 Python (Boto3) 的 AWS SDK。您可以在 Client (用戶端) 區段中找到此命名空間。例如，Systems Manager 的命名空間為 `ssm`。Amazon Elastic Compute Cloud (Amazon EC2) 的命名空間為 `ec2`。

類型：字串

必要：是

Api

您想要執行的 API 操作之名稱。您可以檢視 API 操作 (也稱為方法)，方式是在以下服務參考頁面的左側導覽中選擇一項服務。在您想要呼叫之服務的 Client (用戶端) 部分選擇一個方法。例如，Amazon Relational Database Service (Amazon RDS) 的所有 API 操作 (方法) 均列於以下頁面：Amazon RDS 方法。

類型：字串

必要：是

API 操作輸入

一個或多個 API 操作輸入。您可以檢視可用的輸入 (也稱為參數)，方式是在以下服務參考頁面的左側導覽中選擇一項服務。在您想要呼叫之服務的 Client (用戶端) 部分選擇一個方法。例如，Amazon RDS 的所有方法均列於以下頁面：Amazon RDS 方法。選擇 describe_db_instances 方法並向下捲動以查看可用的參數，例如 DBInstanceIdentifier、Name、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"
}
```

類型：由所選的 API 操作決定

必要：是

輸出

輸出由使用者根據所選 API 操作的回應來指定。

名稱

輸出的名稱。

類型：字串

必要：是

選擇器

回應物件中特定屬性的 JSONPath。您可以檢視回應物件，方式是在以下服務參考頁面的左側導覽中選擇一項服務。在您想要呼叫之服務的 Client (用戶端) 部分選擇一個方法。例如，Amazon RDS 的所有方法均列於以下頁面：Amazon RDS 方法。選擇 describe_db_instances 方法並向下捲動至 Response Structure (回應結構) 區段。DBInstances 列為回應物件。

類型：整數、布林值、字串、StringList、StringMap 或 MapList

必要：是

Type

回應元素的資料類型。

類型：Varies

必要：是

`aws:executeScript` – 執行指令碼

執行使用指定的執行時間和處理程序提供的 Python 或 PowerShell 指令碼。每個 `aws:executeScript` 動作最久可執行 600 秒 (10 分鐘)。您可以透過指定 `aws:executeScript` 步驟的 `timeoutSeconds` 參數來限制逾時。

在函數中使用傳回陳述式將輸出加入輸出承載。有關 `aws:executeScript` 動作定義輸出的範例，請參閱 範例 2：指令碼式 Runbook。您也可以把 Runbook 的 `aws:executeScript` 動作輸出傳送到您指定的 Amazon CloudWatch Logs 日誌群組。如需詳細資訊，請參閱使用 CloudWatch Logs 記錄自動化動作輸出。

如果您想要將`aws:executeScript`動作的輸出傳送至 CloudWatch Logs，或者您為`aws:executeScript`動作指定的指令碼呼叫 AWS API 操作，則一律需要 AWS Identity and Access Management (IAM) 服務角色 （ 或擔任角色 ） 才能執行 Runbook。

`aws:executeScript` 動作包含下列預先安裝的 PowerShell Core 模組。

- Microsoft.PowerShell.Host

- Microsoft.PowerShell.Management
- Microsoft.PowerShell.Security
- Microsoft.PowerShell.Utility
- PackageManagement
- PowerShellGet

若要使用未預先安裝的 PowerShell Core 模組，您的指令碼必須使用 `-Force` 旗標安裝模組，如下列命令所示。不支援 `AWSPowerShell.NetCore` 模組。把 *`ModuleName`* (模組名稱) 取代為您想要安裝的模組。

```
Install-Module ModuleName -Force
```

若要在指令碼中使用 PowerShell Core Cmdlet，建議您使用 `AWS.Tools` 模組，如下列命令所示。將每個*`########`*取代為您自己的資訊。

- Amazon Simple Storage Service (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
```

- 常見或服務獨立 AWS Tools for Windows PowerShell cmdlet。

```
Install-Module AWS.Tools.Common -Force
Get-AWSRegion
```

如果指令碼除了使用 PowerShell Core Cmdlet 之外，還會初始化新物件，您也必須匯入模組，如下列命令所示。

```
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
```

如需在 Runbook 內容中安裝和匯入 `AWS.Tools` 模組，以及使用 PowerShell Core Cmdlet 的範例，請參閱 Automation 執行手冊的視覺化設計體驗。

輸入

提供執行指令碼所需訊息。將每個########取代為您自己的資訊。

> **Note**
>
> Python 指令碼的附件可以是包含指令碼的 .py 檔案或 .zip 檔案。PowerShell 指令碼必須儲存在 .zip 檔案中。

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"
```

```
    }
}
```

執行期

用於執行所提供指令碼的執行時間語言。 `aws:executeScript`支援 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`) 指令碼和 PowerShell 7.4 (`dotnet8`) 指令碼。

支援的值： **`python3.7`** | **`python3.8`** | **`python3.9`** | **`python3.10`** | **`python3.11`** | **`PowerShell Core 6.0`** | **`PowerShell 7.0`** | **`PowerShell 7.4`**

類型：字串

必要：是

> **Note**
>
> 對於 python 執行時間，環境提供 512MB 的記憶體和 512MB 的磁碟空間。對於 PowerShell 執行時間，環境提供 1024MB 的記憶體和 512MB 的磁碟空間。

處理常式

函數名稱。您必須確保處理常式中定義的函數有兩個參數，`events` 和 `context`。PowerShell 執行階段不支援此參數。

類型：字串

必要：是 (Python) | 不支援 (PowerShell)

InputPayload

將傳遞給處理程序的第一個參數的 JSON 或 YAML 物件。這可以用來將輸入資料傳入至指令碼。

類型：字串

必要：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
```

指令碼

您想要在自動化期間執行的內嵌指令碼。

類型：字串

必要：No (Python) | Yes (PowerShell)

連接

動作可呼叫的獨立指令碼檔案或 .zip 檔案的名稱。指定與您在 `Attachments` 請求參數中指定的文件附件檔案的 `Name` 相同的數值。如須詳細資訊，請參閱《AWS Systems Manager API 參考》的附件。如果您使用附件提供指令碼，還必須定義 Runbook 頂層元素的 `files` 區段。如需詳細資訊，請參閱結構描述版本 0.3。

若要為 Python 呼叫檔案，請在 `Handler` 中使用 `filename.method_name` 格式。

> **Note**
>
> Python 指令碼的附件可以是包含指令碼的 .py 檔案或 .zip 檔案。PowerShell 指令碼必須儲存在 .zip 檔案中。

當在附件中包含 Python 庫時，建議在每個模組目錄中新增空的 `__init__.py` 檔案。這允許您從指令碼內容中的附件庫匯入模組。例如：`from library import module`

類型：字串

必要：否

輸出

承載

由函數傳回之物件的 JSON 表示法。最多會傳回 100 KB。如果輸出清單，則最多只能傳回 100 個項目。

搭配 aws：executeScript 使用附件

附件提供了一種強大的方法，可將複雜的指令碼、多個模組和外部相依性與您的`aws:executeScript`動作封裝並重複使用。當您需要下列條件時，請使用附件：

- 將多個 Python 模組或 PowerShell 指令碼封裝在一起。
- 在多個 Runbook 中重複使用相同的指令碼邏輯。
- 在您的指令碼中包含外部程式庫或相依性。
- 分隔複雜的指令碼邏輯，讓您的 Runbook 定義保持乾淨。
- 在團隊或自動化工作流程之間共用指令碼套件。

連接結構和封裝

您可以連接單一檔案或包含多個檔案的 zip 套件。結構取決於您的使用案例：

單一檔案附件

對於簡單的指令碼，您可以連接單一`.py`檔案 (Python) 或包含單一 PowerShell 指令碼`.zip`的檔案。

多模組套件

對於需要多個模組的複雜自動化，請使用下列建議的結構建立 zip 套件：

```
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**
>
> 對於 Python 套件，您必須在包含 Python 模組的每個目錄中包含一個空`__init__.py`檔案。這可讓您使用標準的 Python 匯入語法匯入模組，例如 `from utils import helper_functions`。

PowerShell 套件結構

PowerShell 連接必須使用下列結構封裝在 zip 檔案中：

```
my-powershell-package.zip
### Main.ps1                  # Entry point script
### Modules/
#   ### HelperFunctions.ps1   # Utility functions
#   ### AWSOperations.ps1     # AWS-specific operations
### Config/
    ### Settings.ps1          # Configuration settings
```

使用附件建立 Runbook

請依照下列步驟建立使用附件的 Runbook：

1. 將您的附件上傳至 Amazon S3

   將指令碼檔案或 zip 套件上傳至自動化角色可存取的 S3 儲存貯體。請注意 S3 URI 以用於下一個步驟。

   ```
   aws s3 cp my-automation-package.zip s3://my-automation-bucket/scripts/
   ```

2. 計算附件檢查總和

   計算附件檔案的 SHA-256 檢查總和，以進行安全性驗證：

   ```
   # Linux/macOS
   shasum -a 256 my-automation-package.zip

   # Windows PowerShell
   Get-FileHash -Algorithm SHA256 my-automation-package.zip
   ```

3. 定義 Runbook 中的檔案區段

   在 Runbook 的最上層新增`files`區段，以參考您的附件：

   ```
   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. 參考executeScript步驟中的附件

   使用 Attachment 參數來參考您上傳的檔案：

```
- name: runMyScript
  action: aws:executeScript
  inputs:
    Runtime: python3.11
    Handler: main.process_data
    Attachment: my-automation-package.zip
    InputPayload:
      inputData: "{{InputParameter}}"
```

aws：executeScript 連接範例

下列範例示範搭配 aws:executeScript動作使用附件的不同方式。

範例 1：單一檔案連接

此範例示範如何使用單一 Python 檔案做為附件來處理 EC2 執行個體資料。

附件檔案：process_instance.py

使用下列內容建立 Python 檔案：

```
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
```

完成 Runbook

以下是使用單一檔案附件的完整 Runbook：

```
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
```

範例 2：多模組套件

此範例示範如何使用包含多個 Python 模組的 zip 套件進行複雜的 S3 儲存貯體操作。

套件結構

使用下列結構建立 zip 套件：

```
s3-operations.zip
### main.py
### utils/
#   ### __init__.py
#   ### s3_helper.py
#   ### validation.py
### config/
    ### __init__.py
    ### settings.py
```

main.py（進入點）

協調操作的主要指令碼：

```
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
```

對 aws：executeScript 連接進行故障診斷

使用以下指引來解決aws:executeScript附件的常見問題：

模組匯入錯誤

如果您在使用多模組套件時收到匯入錯誤：

- 請確定您已在每個包含 Python 模組的目錄中包含空__init__.py檔案。
- 確認您的匯入陳述式符合 zip 套件中的實際檔案和目錄結構。
- 一致地使用相對匯入（例如 `from .utils import helper`）或絕對匯入（例如 `from utils import helper`）。

附件找不到錯誤

如果您的自動化找不到附件：

- 確認Attachment參數值與您files區段中的索引鍵完全相符。
- 檢查 files區段中的 S3 儲存貯體路徑和檔案名稱是否正確。
- 確保您的自動化角色具有附件 S3 位置的s3:GetObject許可。
- 確認 Runbook 中的檢查總和符合實際的檔案檢查總和。

處理常式函數錯誤

如果您收到處理常式相關錯誤：

- 對於 Python：在 Handler 參數filename.function_name中使用 格式 （ 例如 main.process_data)。
- 確保您的處理常式函數只接受兩個參數： events和 context。
- 對於 PowerShell：請勿指定Handler參數；指令碼會直接執行。

指令碼執行失敗

如果您的指令碼在執行期間失敗：

- 檢查自動化執行歷史記錄，以取得詳細的錯誤訊息和堆疊追蹤。
- 使用print( )陳述式 (Python) 或 Write-Information(PowerShell) 新增偵錯輸出。
- 確認所有必要的 AWS 許可都授予您的自動化角色。
- 在將指令碼邏輯封裝為附件之前，先在本機測試指令碼邏輯。

結束代碼和錯誤處理

若要正確處理錯誤並傳回結束代碼：

- 在 Python 中：使用 raise Exception("error message") 表示指令碼失敗。
- 在 PowerShell 中：使用 throw "error message"或 Write-Error 表示失敗。
- 從函數傳回結構化資料，以提供詳細的成功/失敗資訊。
- 使用 try-catch 區塊以正常方式處理例外狀況，並提供有意義的錯誤訊息。

aws:executeStateMachine – 執行 AWS Step Functions 狀態機器

執行 AWS Step Functions 狀態機器。

輸入

此動作支援 Step Functions StartExecution API 操作的大部分參數。

Required AWS Identity and Access Management (IAM) 許可

- `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 狀態機的 Amazon Resource Name (ARN)。

類型：字串

必要：是

name

執行的名稱。

類型：字串

必要：否

input

包含執行之 JSON 輸入資料的字串。

類型：字串

必要：否

輸出

以下是針對此動作預先定義的輸出。

executionArn

執行的 ARN。

類型：字串

input

包含執行之 JSON 輸入資料的字串。長度限制適用於承載大小，並以 UTF-8 編碼表示為位元組。

類型：字串

name

執行的名稱。

類型：字串

output

執行的 JSON 輸出資料。長度限制適用於承載大小，在 UTF-8 編碼中表示為位元組。

類型：字串

startDate

開始執行的日期。

類型：字串

stateMachineArn

已執行的指定機器的 ARN。

類型：字串

status

當前執行狀態。

類型：字串

stopDate

如果執行已經結束，則為執行停止的日期。

類型：字串

`aws:invokeWebhook`：叫用 Automation Webhook 整合

叫用指定的 Automation Webhook 整合。如需有關建立 Automation 整合的詳細資訊，請參閱 為 Automation 建立 Webhook 整合。

> **Note**
>
> 若要使用 `aws:invokeWebhook` 動作，使用者或服務角色必須允許執行以下動作：
>
> - ssm:GetParameter
> - kms:解密
>
> 只有在您使用客戶受管金鑰來加密整合的 參數時，才需要 AWS Key Management Service (AWS KMS) `Decrypt`操作的許可。

輸入

提供要叫用的 Automation 整合的資訊。

YAML

```
action: "aws:invokeWebhook"
inputs:
 IntegrationName: "exampleIntegration"
 Body: "Request body"
```

JSON

```
{
    "action": "aws:invokeWebhook",
    "inputs": {
        "IntegrationName": "exampleIntegration",
        "Body": "Request body"
    }
}
```

IntegrationName

Automation 整合的名稱。例如：exampleIntegration。您指定的整合必須已經存在。

類型：字串

必要：是

Body

叫用 Webhook 整合時要傳送的承載。

類型：字串

必要：否

輸出

回應

從 Webhook 提供者回應中收到的文字。

ResponseCode

從 Webhook 提供者回應中收到的 HTTP 狀態碼。

aws:invokeLambdaFunction – 叫用 AWS Lambda 函數

叫用指定的 AWS Lambda 函數。

> **Note**
>
> 每個 aws:invokeLambdaFunction 動作最久可執行 300 秒 (5 分鐘)。您可以透過指定 aws:invokeLambdaFunction 步驟的 timeoutSeconds 參數來限制逾時。

輸入

此動作支援 Lambda 服務大部分的呼叫參數。如需詳細資訊，請參閱呼叫。

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 函數的名稱。此函數必須存在。

類型：字串

必要：是

限定詞

函數版本或別名名稱。

類型：字串

必要：否

InvocationType

呼叫類型。預設值為 RequestResponse。

類型：字串

有效值：Event | RequestResponse | DryRun

必要：否

LogType

如果預設值為 Tail，則叫用類型必須是 RequestResponse。Lambda 會傳回由 Lambda 函數產生的最新 4 KB 日誌資料，且為 base64 編碼。

類型：字串

有效值：None | Tail

必要：否

ClientContext

用戶端特定的資訊。

必要：否

InputPayload

將傳遞給處理常式的第一個參數的 YAML 或 JSON 物件。您可以使用此輸入將資料傳遞到函數。此輸入相比傳統 Payload 輸入提供了更多靈活性和支援。如果您為該動作同時定義 InputPayload 和 Payload，則 InputPayload 優先，並且不使用 Payload 值。

類型：StringMap

必要：否

承載

將傳遞給處理常式的第一個參數的 JSON 字串。這可以用來將輸入資料傳遞到函數。建議您使用 InputPayload 輸入，因為已新增其功能。

類型：字串

必要：否

輸出

StatusCode

HTTP 狀態碼

FunctionError

如果存在，則表示函數執行期間發生錯誤。錯誤詳細資訊包含在回應承載中。

LogResult

Lambda 函數呼叫的 base64 編碼日誌。日誌僅在呼叫類型為 RequestResponse 時才會出現，且日誌經過請求。

承載

由 Lambda 函數傳回之物件的 JSON 表示法。承載僅在呼叫類型為 RequestResponse 時才會出現。

以下節錄自 AWS-PatchInstanceWithRollback Runbook，示範如何從 aws:invokeLambdaFunction 動作參照輸出。

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 - 迭代自動化中的步驟

此動作會迭代自動化執行手冊中的步驟子集。您可以選擇 do while 或 for each 樣式迴圈。若要建構 do while 迴圈，請使用 LoopCondition 輸入參數。若要建構 for each 迴圈，請使用 Iterators 和 IteratorDataType 輸入參數。使用 aws:loop 動作時，僅指定 Iterators 或 LoopCondition 輸入參數。迭代的數量的上限為 100。

onCancel 屬性只能針對迴圈內定義的步驟。aws:loop 動作不支援 onCancel 屬性。onFailure 屬性可用於 aws:loop 動作，但只有在發生意外錯誤導致步驟失敗時，才會使用該此屬性。如果您定義迴圈中步驟的 onFailure 屬性，則 aws:loop 動作會繼承這些屬性，並在發生故障時做出相應反應。

範例

下列是如何建構不同類型迴圈動作的範例。

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
```

輸入

輸入如下。

迭代器

要重複執行的步驟的項目清單。迭代器的數量的上限為 100。

類型：StringList

必要：否

迭代器資料類型

用來指定 Iterators 的資料類型的選用參數。此參數的值可與 Iterators 輸入參數一起提供。如果您不指定此參數和 Iterators 的值，則您必須指定 LoopCondition 參數的值。

類型：字串

有效值：布林值 | 整數 | 字串 | StringMap

預設：字串

必要：否

LoopCondition

包含 Variable 和要評估的運算子條件。如果您不指定此參數的值，則您必須指定 Iterators 和 IteratorDataType 參數的值。您可以透過 And、Not 和 Or 運算子的組合來使用複雜的運算子評估。在迴圈中的步驟完成後，就會評估條件。如果條件為 true 且尚未達到 MaxIterations 值，則迴圈中的步驟會再次執行。運算子條件如下：

字串運算

- StringEquals
- EqualsIgnoreCase
- StartsWith
- EndsWith
- 包含

數值運算

- NumericEquals
- NumericGreater
- NumericLesser
- NumericGreaterOrEquals
- NumericLesser

- NumericLesserOrEquals

布林運算

- BooleanEquals

類型：StringMap

必要：否

MaxIterations

迴圈中步驟執行的最大次數。達到此輸入指定的值後，即使 LoopCondition 仍然 true，或 Iterators 參數中仍有剩餘物件，迴圈也會停止執行。

類型：整數

有效值：1 - 100

必要：否

步驟

在迴圈中執行的步驟清單。這些函數就像一個巢狀的執行手冊。在這些步驟中，您可以使用語法 for each，存取 {{loopStepName.CurrentIteratorValue}} 迴圈的目前迭代器值。您還可以使用語法 {{loopStepName.CurrentIteration}}，存取這兩個迴圈類型的目前迭代的整數值。

類型：步驟清單

必要：是

輸出

CurrentIteration

將目前迴圈迭代作為整數。迭代值從 1 開始。

類型：整數

CurrentIteratorValue

做為字串的目前迭代器的值。此輸出僅存在於 for each 迴圈中。

類型：字串

`aws:pause` – 暫停自動化

此動作會暫停自動化。一旦暫停，自動化狀態就會是 Waiting (等待)。若要繼續自動化，請使用具 Resume 訊號類型的 SendAutomationSignal API 操作。我們建議使用 `aws:sleep` 或者 `aws:approve` 操作，以更精細地控制您的工作流程。

> **Note**
>
> 此動作的預設逾時時間為 7 天 (604800 秒)，最大值為 30 天 (2592000 秒)。您可以透過指定 `aws:pause` 步驟的 `timeoutSeconds` 參數來限制或延長逾時。

輸入

輸入如下。

YAML

```
name: pauseThis
action: aws:pause
timeoutSeconds: 1209600
inputs: {}
```

JSON

```
{
    "name": "pauseThis",
    "action": "aws:pause",
    "timeoutSeconds": "1209600",
    "inputs": {}
}
```

輸出

無

`aws:runCommand` – 在受管執行個體上執行命令

執行指定的命令。

> **Note**
>
> 自動化僅支援一個 AWS Systems Manager Run Command動作的輸出。Runbook 可以包含多個 Run Command 動作，但一次僅一個動作支援輸出。

輸入

此動作支援大部分的傳送命令參數。如需詳細資訊，請參閱 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

如果您擁有 或 命令類型文件 AWS，請指定文件的名稱。如果您使用的文件是由不同 AWS 帳戶共用，則請指定文件的 Amazon Resource Name (ARN)。如需使用共用文件的詳細資訊，請參閱 使用共用的 SSM 文件。

類型：字串

必要：是

InstanceIds

您想要命令執行的執行個體 ID。您最多可以指定 50 個 ID。

您也可以使用虛擬參數 {{RESOURCE_ID}} 取代執行個體 ID，在目標群組中的所有執行個體上執行命令。如需這些虛擬參數的詳細資訊，請參閱註冊維護時段任務時使用虛擬參數。

另一種方法是使用 Targets 參數將命令傳送到執行個體機群。Targets 參數接受 Amazon Elastic Compute Cloud (Amazon EC2) 標籤。如需使用 Targets 參數的詳細資訊，請參閱 大規模執行命令。

類型：StringList

必要：否 (如果您未指定 InstanceIds 或使用 {{RESOURCE_ID}} 虛擬參數，則必須指定 Targets 參數。)

目標

一系列的搜尋條件，使用您指定的鍵值組合將執行個體設為目標。若您沒有在呼叫中提供一或多個執行個體 ID，則 Targets 為必要項目。如需使用 Targets 參數的詳細資訊，請參閱 大規模執行命令。

類型：MapList (清單中的對應結構描述必須符合物件。) 如需詳細資訊，請參閱《AWS Systems Manager API 參考》中的 Target。

必要：否 (如果您未指定 Targets，則必須指定 InstanceIds 或使用 {{RESOURCE_ID}} 虛擬參數)。

以下是範例。

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"
            ]
        }
    }
}
```

參數

文件中指定的必要和選用參數。

類型：映射

必要：否

CloudWatchOutputConfig

用於傳送命令輸出至 Amazon CloudWatch Logs 的組態選項。如需傳送命令輸出至 CloudWatch Logs 的詳細資訊，請參閱 設定 Run Command 的 Amazon CloudWatch Logs。

類型：StringMap (映射的結構描述必須與物件相符。如需詳細資訊，請參閱《AWS Systems Manager API 參考》中的 CloudWatchOutputConfig。

必要：否

以下是範例。

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
        }
    }
}
```

註解

關於命令，由使用者定義的資訊。

類型：字串

必要：否

DocumentHash

文件的雜湊。

類型：字串

必要：否

DocumentHashType

雜湊的類型。

類型：字串

有效值：Sha256 | Sha1

必要：否

NotificationConfig

傳送通知的組態。

必要：否

OutputS3BucketName

用於命令輸出回應的 S3 儲存貯體名稱。您的受管節點必須具有 S3 儲存貯體的許可，才能成功記錄輸出。

類型：字串

必要：否

OutputS3KeyPrefix

字首。

類型：字串

必要：否

ServiceRoleArn

AWS Identity and Access Management (IAM) 角色的 ARN。

類型：字串

必要：否

TimeoutSeconds

在執行個體 AWS Systems Manager SSM Agent上等待命令交付至 的時間，以秒為單位。如果在指定的值到達之前執行個體上的 SSM Agent 沒有收到命令，則命令的狀態會變更為 `Delivery Timed Out`。

類型：整數

必要：否

有效值：30-2592000

輸出

CommandId

命令的 ID。

Status

命令的狀態。

ResponseCode

命令的回應代碼。如果執行的文件有 1 個以上步驟，則不會傳回此輸出的值。

輸出

命令的輸出。如果您使用命令將標籤或多個執行個體設為目標，則不會傳回輸出值。您可以使用 `GetCommandInvocation` 和 `ListCommandInvocations` API 操作來擷取個別執行個體的輸出。

`aws:runInstances` – 啟動 Amazon EC2 執行個體。

啟動新的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。

輸入

動作支援大部分的 API 參數。如需詳細資訊，請參閱 RunInstances API 文件。

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

預訂.

類型：字串

必要：否

BlockDeviceMappings

執行個體的區塊型儲存設備。

類型：MapList

必要：否

ClientToken

用於確保請求之冪等的識別符。

類型：字串

必要：否

DisableApiTermination

開啟或關閉執行個體 API 終止。

類型：布林值

必要：否

EbsOptimized

開啟或關閉 Amazon Elastic Block Store (Amazon EBS) 最佳化。

類型：布林值

必要：否

IamInstanceProfileArn

執行個體 (IAM) 執行個體設定檔的 Amazon Resource Name AWS Identity and Access Management (ARN)。

類型：字串

必要：否

IamInstanceProfileName

執行個體之 IAM 執行個體設定檔的名稱。

類型：字串

必要：否

ImageId

Amazon Machine Image (AMI) 的 ID。

類型：字串

必要：是

InstanceInitiatedShutdownBehavior

指示執行個體在系統關機時停止或終止。

類型：字串

必要：否

InstanceType

執行個體類型。

> **Note**
> 如果不提供執行個體類型價值，則會使用 m1.small 執行個體類型。

類型：字串

必要：否

KernelId

核心的 ID。

類型：字串

必要：否

KeyName

金鑰對的名稱。

類型：字串

必要：否

MaxInstanceCount

要啟動的執行個體數量上限。

類型：字串

必要：否

MetadataOptions

執行個體的中繼資料選項。如需詳細資訊，請參閱 InstanceMetadataOptionsRequest。

類型：StringMap

必要：否

MinInstanceCount

要啟動的執行個體數量下限。

類型：字串

必要：否

監控

打開或關閉詳細監控。

類型：布林值

必要：否

NetworkInterfaces

網路界面。

類型：MapList

必要：否

配置

執行個體的配置。

類型：StringMap

必要：否

PrivateIpAddress

主要 IPv4 地址。

類型：字串

必要：否

RamdiskId

RAM 磁碟的 ID。

類型：字串

必要：否

SecurityGroupIds

執行個體安全群組的 ID。

類型：StringList

必要：否

SecurityGroups

執行個體安全群組的名稱。

類型：StringList

必要：否

SubnetId

子網路 ID。

類型：字串

必要：否

TagSpecifications

要在啟動期間套用到資源的標籤。您只能在啟動時標記執行個體和磁碟區。指定的標籤會套用至所有於啟動期間建立的執行個體或磁碟區。若要在執行個體啟動後將其標記，請使用 aws:createTags – 建立 AWS 資源的標籤 動作。

類型：MapList (如需詳細資訊，請參閱 TagSpecification。)

必要：否

UserData

做為字串常值提供的指令碼。如果輸入常值，則其必須為 Base64 編碼。

類型：字串

必要：否

輸出

InstanceIds

執行個體的 ID。

InstanceStates

執行個體目前的狀態。

aws:sleep – 延遲自動化

將自動化延遲一段指定的時間。此動作採用國際標準組織 (ISO) 8601 日期和時間格式。如需此日期和時間格式的詳細資訊，請參閱 ISO 8601。

輸入

您可以將自動化延遲一段指定的持續時間。

YAML

```
name: sleep
action: aws:sleep
inputs:
  Duration: PT10M
```

JSON

```
{
   "name":"sleep",
   "action":"aws:sleep",
   "inputs":{
      "Duration":"PT10M"
   }
}
```

您也可以延遲自動化直到指定的日期和時間。如果超過指定日期和時間，動作會立即執行。

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**
>
> 自動化支援的延遲上限為 604799 秒 (7 天)。

持續時間

ISO 8601 持續時間。您無法指定負數的持續時間。

類型：字串

必要：否

時間戳記

ISO 8601 時間戳記。如果您不指定此參數的值，則您必須指定 `Duration` 參數的值。

類型：字串

必要：否

輸出

無

`aws:updateVariable` - 更新執行手冊變數的值

此動作會更新執行手冊變數的值。值的資料類型必須符合您希望更新的變數的資料類型。不支援資料類型轉換。aws:updateVariable 動作不支援 onCancel 屬性。

輸入

輸入如下。

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"]
    }
}
```

名稱

您希望更新其值的變數名稱。您必須使用格式 variable:*variable name*

類型：字串

必要：是

Value

要指派給變數的新值。該值必須與變數的資料類型相符。不支援資料類型轉換。

類型：布林值 | 整數 | MapList | 字串 | StringList | StringMap

必要：是

約束:

- MapList 最多可包含 200 個項目。
- 鍵的長度最短可以是 1，最長可以是 50。
- StringList 最少可以是 0 個項目，最多可以是 50 個項目。
- 字串的長度最短可以是 1，最長可以是 512。

輸出

無

`aws:waitForAwsResourceProperty` – 等待 AWS 資源屬性

`aws:waitForAwsResourceProperty` 動作可讓您的自動化在繼續自動化之前等待特定的資源狀態或事件狀態。如需有關如何使用此動作的更多範例，請參閱 其他執行手冊範例。

> **Note**
>
> 此動作的預設逾時值為 3600 秒 (1 小時)。您可以透過指定 `aws:waitForAwsResourceProperty` 步驟的 `timeoutSeconds` 參數來限制或延長逾時。如需使用此動作的詳細資訊和範例，請參閱 處理 Runbook 中的逾時。

輸入

輸入是由您選擇的 API 操作定義。

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"
    ]
  }
}
```

服務

包含您要執行之 API 操作的 AWS 服務 命名空間。例如，的命名空間 AWS Systems Manager 為 `ssm`。Amazon Elastic Compute Cloud (Amazon EC2) 的命名空間為 `ec2`。您可以在 AWS CLI 命令參考的可用服務區段中檢視支援的 AWS 服務 命名空間清單。

類型：字串

必要：是

Api

您想要執行的 API 操作之名稱。您可以檢視 API 操作 (也稱為方法)，方式是在以下服務參考頁面的左側導覽中選擇一項服務。在您想要呼叫之服務的 Client (用戶端) 部分選擇一個方法。例如，Amazon Relational Database Service (Amazon RDS) 的所有 API 操作 (方法) 均列於以下頁面：Amazon RDS 方法。

類型：字串

必要：是

API 操作輸入

一個或多個 API 操作輸入。您可以檢視可用的輸入 (也稱為參數)，方式是在以下服務參考頁面的左側導覽中選擇一項服務。在您想要呼叫之服務的 Client (用戶端) 部分選擇一個方法。例

如，Amazon RDS 的所有方法均列於以下頁面：Amazon RDS 方法。選擇 describe_db_instances 方法並向下捲動以查看可用的參數，例如 DBInstanceIdentifier、Name、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"
}
```

類型：由所選的 API 操作決定

必要：是

PropertySelector

回應物件中特定屬性的 JSONPath。您可以檢視回應物件，方式是在以下服務參考頁面的左側導覽中選擇一項服務。在您想要呼叫之服務的 Client (用戶端) 部分選擇一個方法。例如，Amazon RDS 的所有方法均列於以下頁面：Amazon RDS 方法。選擇 describe_db_instances 方法並向下捲動至 Response Structure (回應結構) 區段。DBInstances 列為回應物件。

類型：字串

必要：是

DesiredValues

讓自動化繼續的預期狀態。

類型：MapList、StringList

必要：是

自動化系統變數

AWS Systems Manager 自動化 Runbook 使用以下變數。如需這些變數的使用範例，請檢視 `AWS-UpdateWindowsAmi` Runbook 的 JSON 來源。

檢視 **`AWS-UpdateWindowsAmi`** Runbook 的 JSON 來源

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Documents (文件)。
3. 在文件清單中，使用搜尋列或搜尋列右側的號碼選擇 Runbook **`AWS-UpdateWindowsAmi`**。
4. 選擇 Content (內容) 索引標籤。

系統變數

Automation Runbook 目前支援以下系統變數。

變數	詳細資訊
`global:ACCOUNT_ID`	自動化執行所在的使用者或角色 AWS 帳戶 ID。
`global:DATE`	(在執行時間的) 日期格式為 yyyy-MM-dd。
`global:DATE_TIME`	(在執行時間的) 日期和時間格式為 yyyy-MM-dd_HH.mm.ss。
`global:AWS_PARTITION`	資源所在的分割區。對於標準 AWS 區域，分割區為 `aws`。如果資源處於其他分割區，則會傳回 `aws-`*`partitionname`* 分割區。例如，AWS GovCloud (US-West) 區域的資源分割區為 `aws-us-gov` 。
`global:REGION`	Runbook 執行的區域。例如 us-east-2。

自動化變數

Runbook 支援以下自動化變數。

變數	詳細資訊
`automation:EXECUTION_ID`	指派給目前自動化的唯一識別符。例如：`1a2b3c-1a2b3c-1a2b3c-1a2b3c-1a2b3c1a2b3c` 。

主題

- 術語
- 支援的案例
- 不支援的案例

術語

以下術語說明如何解決變數和參數。

術語	定義	範例
Constant ARN (常數 ARN)	不含變數的有效 Amazon Resource Name (ARN)。	`arn:aws:iam::123456789012:role/roleName`
Runbook 參數	在 Runbook 層級定義的參數 (例如，`instanceId`)。此參數用於基本的字串替換。此值會在 Start Execution (開始執行) 時間提供。	`{ "description": "Create Image Demo", "version": "0.3", "assumeRole": "Your_Automation_Assume_Role_ARN ", "parameters":{ "instanceId": { "type": "String", "description": "Instance to create image from" }`

術語	定義	範例
		<pre>}</pre>

System variable (系統變數)	在 Runbook 任何部分評估時替換到 Runbook 的一般變數。	<pre>"activities": [
		   {
		      "id": "copyImag
		e",
		      "activityType":
		 "AWS-CopyImage",
		      "maxAttempts": 1,
		      "onFailure":
		 "Continue",
		      "inputs": {
		         "ImageName":
		 "{{imageName}}",
		         "SourceIm
		ageId": "{{source
		ImageId}}",
		         "SourceRe
		gion": "{{source
		Region}}",
		         "Encrypted":
		 true,
		         "ImageDes
		cription": "Test
		 CopyImage Description
		 created on **{{global:
		DATE}}** "
		      }
		   }
		]</pre>

術語	定義	範例

Automation variable (自動化變數)	在文件任何部分評估時替換到 Runbook 且與自動化相關的變數。	{
		"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}}** ”
		]
		}
		}
		}

術語	定義	範例
Systems Manager 參數	在 中定義的變數 AWS Systems Manager Parameter Store。無法在步驟輸入中直接參考它。存取參數可能需要許可。	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

術語	定義	範例
		timeoutSeconds: 1200 onFailure: Abort inputs: ImageId: '{{LatestAmi}}' ...

支援的案例

案例	說明	範例
建立時的常數 ARN assumeRole 。	將會執行授權檢查，以確認呼叫的使用者許可傳遞指定的assumeRole 。	{ "description": "Test all Automation resolvable parameters", "schemaVersion": "0.3", "assumeRole": ***"arn:aws:iam::123456789012:role/roleName"*** , "parameters": { ...
自動化啟動時，為 AssumeRole 提供的 Runbook 參數。	必須在 Runbook 的參數清單中定義。	{ "description": "Test all Automation resolvable parameters", "schemaVersion": "0.3", "assumeRole": **"{{dynamicARN}}"** , "parameters": { ...

案例	說明	範例

在開始時提供給 Runbook 參數的值。	客戶提供用於參數的值。在開始時間提供的任何輸入都必須在 Runbook 的參數清單中定義。	<pre>...
		"parameters": {
		    "amiId": {
		      "type": "String",
		      "default":
		"ami-12345678 ",
		      "description":
		"list of commands to
		run as part of first
		step"
		    },
		...</pre>
		啟動自動執行的輸入包含：{"amiId" : ["ami-12345678 "] }

案例	說明	範例

Runbook 內容中參考的 Systems Manager 參數。	變數存在於客戶帳戶內，或是可公開存取的參數，而且 Runbook 的 AssumeRole 可以存取變數。檢查會於建立時間執行，以確認 AssumeRole 可存取。無法在步驟輸入中直接參考參數。	```
		...
		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}}'
		...
		``

案例	說明	範例

在步驟定義中參考的系統變數	當自動化啟動時，系統變數會替換至 Runbook。插入 Runbook 的值與替換發生的時間相關。例如，由於執行步驟之間耗費的時間，因此在步驟 1 插入的時間變數值會不同於在步驟 3 插入的值。系統變數不必在 Runbook 的參數清單中設定。	```
		...
		  "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
		}}}"
		            ]
		        }
		    }
		}, ...
		``

案例	說明	範例

在步驟定義中參考的自動化變數。	自動化變數不必在 Runbook 的參數清單中設定。唯一支援的自動化變數為 automation:EXECUTION_ID。	```
		...
		"mainSteps": [
		    {
		      "name": "invokeLambdaFunction",
		      "action": "aws:invokeLambdaFunction",
		      "maxAttempts": 1,
		      "onFailure": "Continue",
		      "inputs": {
		        "FunctionName": "Hello-World-LambdaFunction",
		
		"Payload" : "{ "executionId" : "{{automation:EXECUTION_ID}}" }"
		      }
		    }
		...
		``

案例	說明	範例

請在下一個步驟定義中參閱前一個步驟的輸出。	此為參數重新導向。參考先前步驟的輸出時會使用語法 {{stepName.OutputName}} 。客戶無法在 Runbook 參數使用此語法。在參照步驟執行時，會解決此問題。此參數不列於 Runbook 參數中。	```
		...
		"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"
		      }
		    }
		
		...
		``

不支援的案例

案例	註解	範例

建立時，為 assumeRole 提供的 Systems Manager 參數	不支援。	...
		
		{
		"description":
		"Test all Automation
		resolvable parameter
		s",
		"schemaVersion":
		"0.3",
		"assumeRole":
		"*{{ssm:administrato
		rRoleARN}}* ",
		"parameters": {
		
		...

直接在步驟輸入中參考的 Systems Manager 參數。	建立時傳回 InvalidDo cumentContent 例外狀況。	... mainSteps: - name: launchIns tance action: 'aws:runI nstances' maxAttempts: 3 timeoutSeconds: 1200 onFailure: Abort inputs: ImageId: '{{ssm:/ aws/service/ami-win dows-latest/Window s_Server-2016-Engl ish-Full-Base}}' ...
變數步驟定義	Runbook 步驟的定義是由變數建構。	...

案例	註解	範例

		"mainSteps": [
		{
		"name": "LaunchInstance",
		"action": "aws:runInstances",
		"**{{attemptModel}}** ": 1,
		"onFailure": "Continue",
		"inputs": {
		"ImageId": "*ami-12345678* ",
		"MinInstanceCount": 1,
		"MaxInstanceCount": 2
		}
		
		...
		
		User supplies input :
		{ "attemptModel" :
		"**minAttempts** " }

案例	註解	範例

交互參照 Runbook 參數	使用者會在開始時間提供輸入參數，而這是 Runbook 中另一個參數的參考。	```
		...
		"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}
		}"
		    },
		
		...
		``

案例	註解	範例
多層級擴展	Runbook 會定義一個評估變數名稱的變數。這位於變數分隔符號內 (即 {{ }})，且會擴展至該變數/參數的值。	... "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}}*** }}"] }

案例	註解	範例
		``` ... Note: The customer intention here would be to run a command of "echo {Hello world}" ```

案例	註解	範例
參考 Runbook 步驟的輸出，其為不同的變數類型	使用者參考後續步驟內先前 Runbook 步驟的輸出。輸出為不符合後續步驟中動作需求的變數類型。	... 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

案例	註解	範例
		outputs a "StringList" type variable.

建立您自己的執行手冊

自動化執行手冊定義 Systems Manager 在自動化執行時，對受管執行個體和其他 AWS 資源執行的動作。自動化是 中的工具 AWS Systems Manager。Runbook 包含循序執行的一或多個步驟。每個步驟都是圍繞單一動作而建立的。來自一個步驟的輸出可以做為後續步驟中的輸入。

執行這些動作及其步驟的程序稱為自動化。

Runbook 支援的動作類型可讓您自動化 AWS 環境中的各種操作。例如，使用 `executeScript` 動作類型，您可以直接在 Runbook 中嵌入 Python 或 PowerShell 指令碼。(建立自訂 Runbook 時，您可以內嵌新增指令碼，或從 S3 儲存貯體或本機電腦連接指令碼。) 您可以使用 `createStack`和 `deleteStack`動作類型來自動化 AWS CloudFormation 資源的管理。此外，使用 `executeAwsApi`動作類型，步驟可以在任何 中執行任何 API 操作 AWS 服務，包括建立或刪除 AWS 資源、啟動其他程序、啟動通知等。

如需自動化所支援的全部 20 個動作類型清單，請參閱 Systems Manager Automation 動作參考。

AWS Systems Manager 自動化提供數個執行手冊，其中包含預先定義的步驟，可用來執行常見的任務，例如重新啟動一或多個 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體或建立 Amazon Machine Image()AMI。您也可以建立自己的 Runbook 並與其他 Runbook 共用 AWS 帳戶，或公開給所有 Automation 使用者。

Runbook 使用 YAML 或 JSON 編寫而成。使用 Systems Manager Automation 主控台中的 Document Builder (文件建置器)，不過，您可以建立 Runbook，而無需以原生 JSON 或 YAML 撰寫。

> **Important**
>
> 如果您執行可使用 AWS Identity and Access Management (IAM) 服務角色叫用其他服務的自動化工作流程，請注意您必須為該服務角色設定可叫用這些服務的許可。此要求適用於所有 AWS Automation Runbook (`AWS-*` Runbook)，例如 `AWS-ConfigureS3BucketLogging`、`AWS-CreateDynamoDBBackup` 和 `AWS-RestartEC2Instance` Runbook 等。此要求也適用於您建立的任何自訂自動化執行手

冊， AWS 服務 這些執行手冊使用呼叫其他服務的動作來叫用其他 。例如，如果您使用 `aws:executeAwsApi`、`aws:createStack` 或 `aws:copyImage` 動作，為服務角色設定可叫用這些服務的許可。您可以將 IAM 內嵌政策新增至角色， AWS 服務 以將許可授予其他 。如需詳細資訊，請參閱（選用）新增自動化內嵌政策或客戶受管政策以叫用其他政策 AWS 服務。

如需您可以在 Runbook 中指定的動作的相關資訊，請參閱 Systems Manager Automation 動作參考。

如需使用 AWS Toolkit for Visual Studio Code 建立 Runbook 的相關資訊，請參閱AWS Toolkit for Visual Studio Code 《 使用者指南》中的使用 Systems Manager Automation 文件。

如需有關使用視覺化設計工具來建立自訂執行手冊的資訊，請參閱 Automation 執行手冊的視覺化設計體驗。

內容

- Automation 執行手冊的視覺化設計體驗
  - 開始之前
  - 視覺化設計體驗介面的概觀
    - 動作瀏覽器
    - Canvas
    - 表格
    - 鍵盤快速鍵
  - 運用視覺化設計體驗
    - 建立執行手冊工作流程
    - 設計執行手冊
    - 更新執行手冊
    - 匯出執行手冊
  - 設定動作的輸入和輸出
    - 為動作提供輸入資料
    - 定義動作的輸出資料
  - 藉助視覺化設計體驗錯誤處理
    - 出現錯誤時重試動作
    - 逾時

        - 失敗的動作
        - 取消的動作
        - 關鍵動作
        - 結束動作
    - 教學課程：使用視覺化設計體驗建立執行手冊
        - 步驟 1：導覽至視覺化設計體驗
        - 步驟 2：建立工作流程
        - 步驟 3：檢閱自動產生的程式碼
        - 步驟 4：執行新的執行手冊
        - 步驟 5：清除
- 撰寫 Automation Runbook
    - 識別您的使用案例
    - 設定開發環境
    - 開發 Runbook 內容
    - 範例 1：建立父子 Runbook
        - 建立子系 Runbook
        - 建立父系 Runbook
    - 範例 2：指令碼式 Runbook
    - 其他執行手冊範例
        - 部署 VPC 架構和 Microsoft Active Directory 網域控制站
        - 從最新的快照還原根磁碟區
        - 建立 AMI 和跨區域複本
- 建立填入 AWS 資源的輸入參數
- 使用文件建置器建立執行手冊
    - 使用文件建置器建立自訂執行手冊
    - 建立執行指令碼的執行手冊
- 在執行手冊中使用指令碼
    - 使用 Runbook 的許可
    - 將指令碼新增至 Runbook
    - Runbook 的指令碼限制

- 在執行手冊中使用條件陳述式
  - 使用 aws:branch 動作
    - 在 Runbook 中建立 aws:branch 步驟
      - 關於建立輸出變數
    - 範例 aws:branch Runbook
    - 使用運算子建立複雜的分支自動化
  - 如何使用條件選項的範例
- 使用動作輸出作為輸入
  - 在執行手冊中使用 JSONPath
- 為 Automation 建立 Webhook 整合
  - 建立整合 (主控台)
  - 建立整合 (命令列)
  - 為整合建立 Webhook
- 處理 Runbook 中的逾時

Automation 執行手冊的視覺化設計體驗

AWS Systems Manager 自動化提供低程式碼視覺化設計體驗，可協助您建立自動化 Runbook。視覺化設計體驗提供拖放式介面，可選擇新增自己的程式碼，讓您可以更輕鬆地建立和編輯執行手冊。透過視覺化設計體驗，您可以執行下列操作：

- 控制條件陳述式。
- 控制每個動作篩選或轉換輸入和輸出的方式。
- 設定錯誤處理。
- 製作新執行手冊的原型。
- 使用您的原型執行手冊做為 AWS Toolkit for Visual Studio Code本機開發的起點。

當您建立或編輯執行手冊時，可以從 Automation 主控台存取視覺化設計體驗。當您建立執行手冊時，視覺化設計體驗會驗證您的工作並自動產生程式碼。您可以檢閱產生的程式碼，或將其匯出以進行本地開發。完成後，您可以儲存和執行您的執行手冊，並在 Systems Manager Automation 主控台中檢查結果。

開始之前

若要使用視覺化設計體驗，您需要 AWS 帳戶和 登入資料，以為您想要使用的任何資源提供正確的許可。

在視覺化設計體驗中，Automation 與 Amazon CodeGuru 安全工具整合，可協助您偵測 Python 指令碼中的安全政策違規和漏洞。若要將此功能用於 `aws:executeScript` 動作，您的 AWS Identity and Access Management (IAM) 政策必須包含下列許可：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codeguru-security:CreateUploadUrl",
                "codeguru-security:CreateScan",
                "codeguru-security:GetScan",
                "codeguru-security:GetFindings"
            ]
        }
    ]
}
```

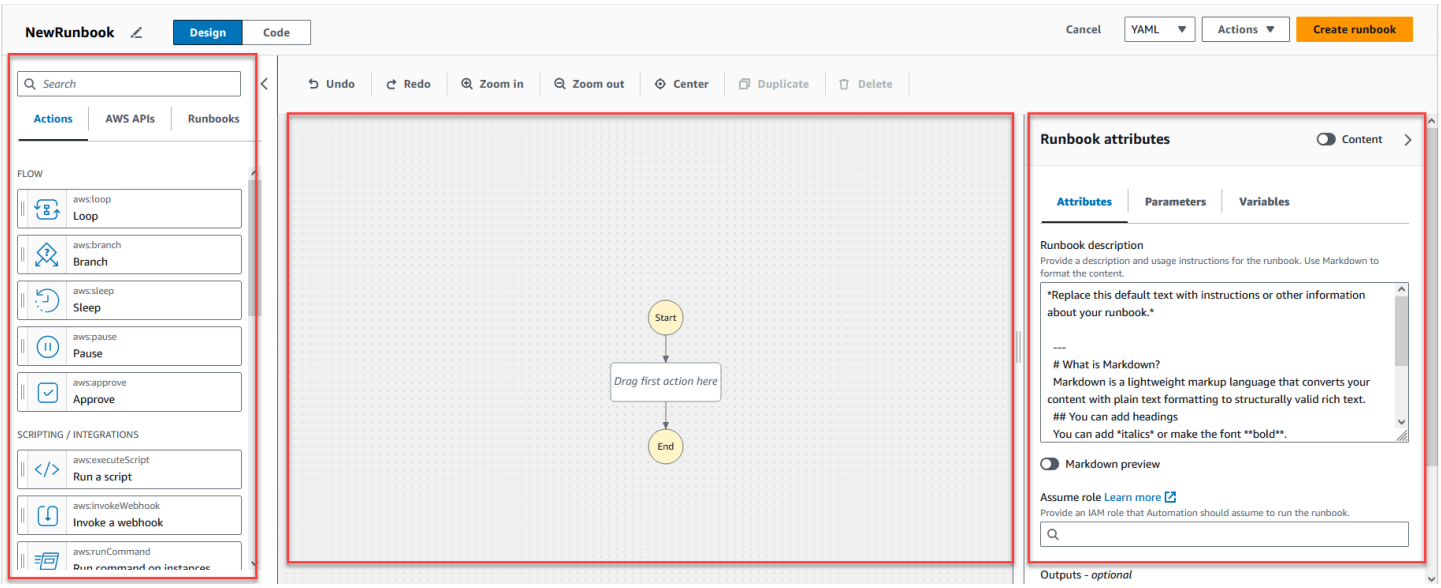
主題

- 視覺化設計體驗介面的概觀
- 運用視覺化設計體驗
- 設定動作的輸入和輸出
- 藉助視覺化設計體驗錯誤處理
- 教學課程：使用視覺化設計體驗建立執行手冊

視覺化設計體驗介面的概觀

Systems Manager Automation 的視覺化設計體驗是低程式碼的視覺化工作流程設計工具，可協助您建立 Automation 執行手冊。

透過介面組件的概觀了解視覺化設計體驗：



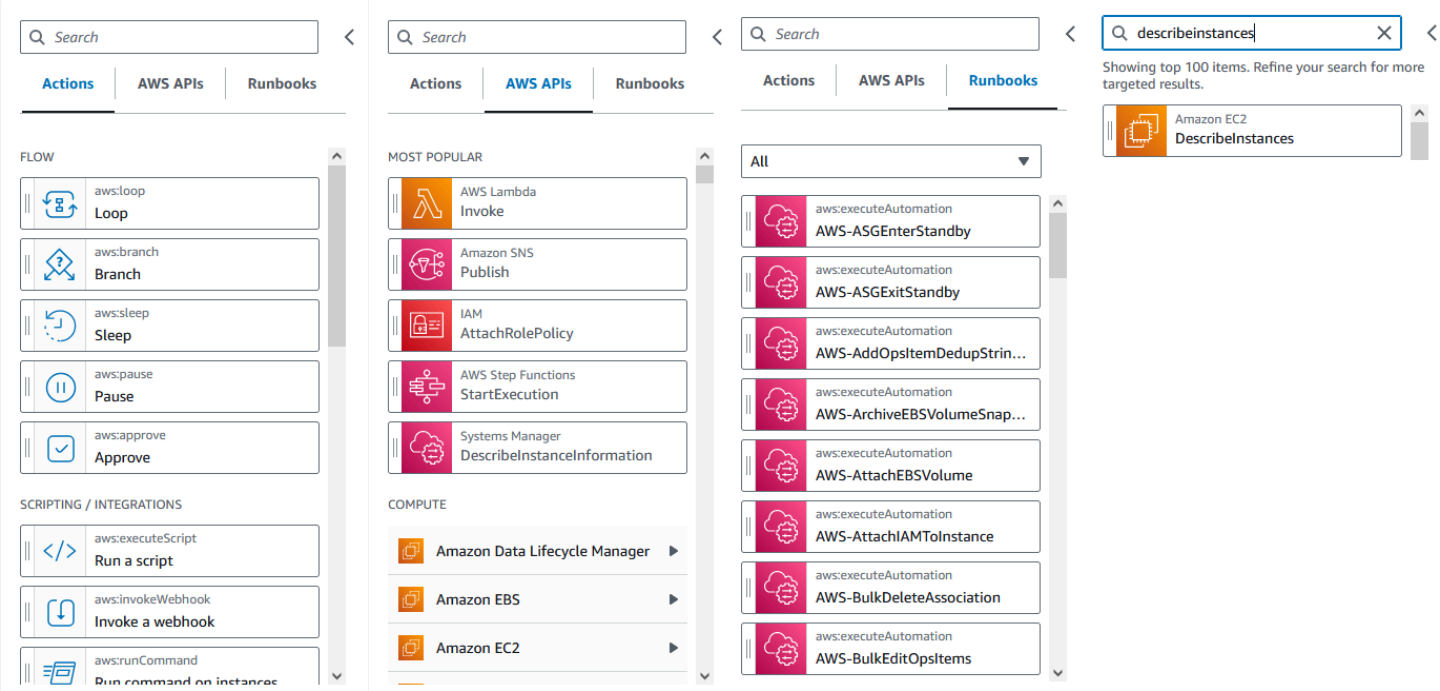
- 動作瀏覽器包含動作、AWS API 和執行手冊索引標籤。
- 在畫布上，您可以將動作拖放到工作流程圖形中、變更動作順序，以及選取要設定或檢視的動作。
- 您可以在表單面板中檢視和編輯您在畫布上選取之任何動作的屬性。選取內容切換按鈕可檢視執行手冊的 YAML 或 JSON，並反白顯示目前選取的動作。

當您需要協助時，資訊連結會開啟包含內容資訊的面板。這些面板也包含 Systems Manager Automation 文件中相關主題的連結。

動作瀏覽器

從動作瀏覽器，您可以選取要拖放到工作流程圖形的動作。您可以使用動作瀏覽器頂端的搜尋欄位來搜尋所有動作。動作瀏覽器包含下列索引標籤：

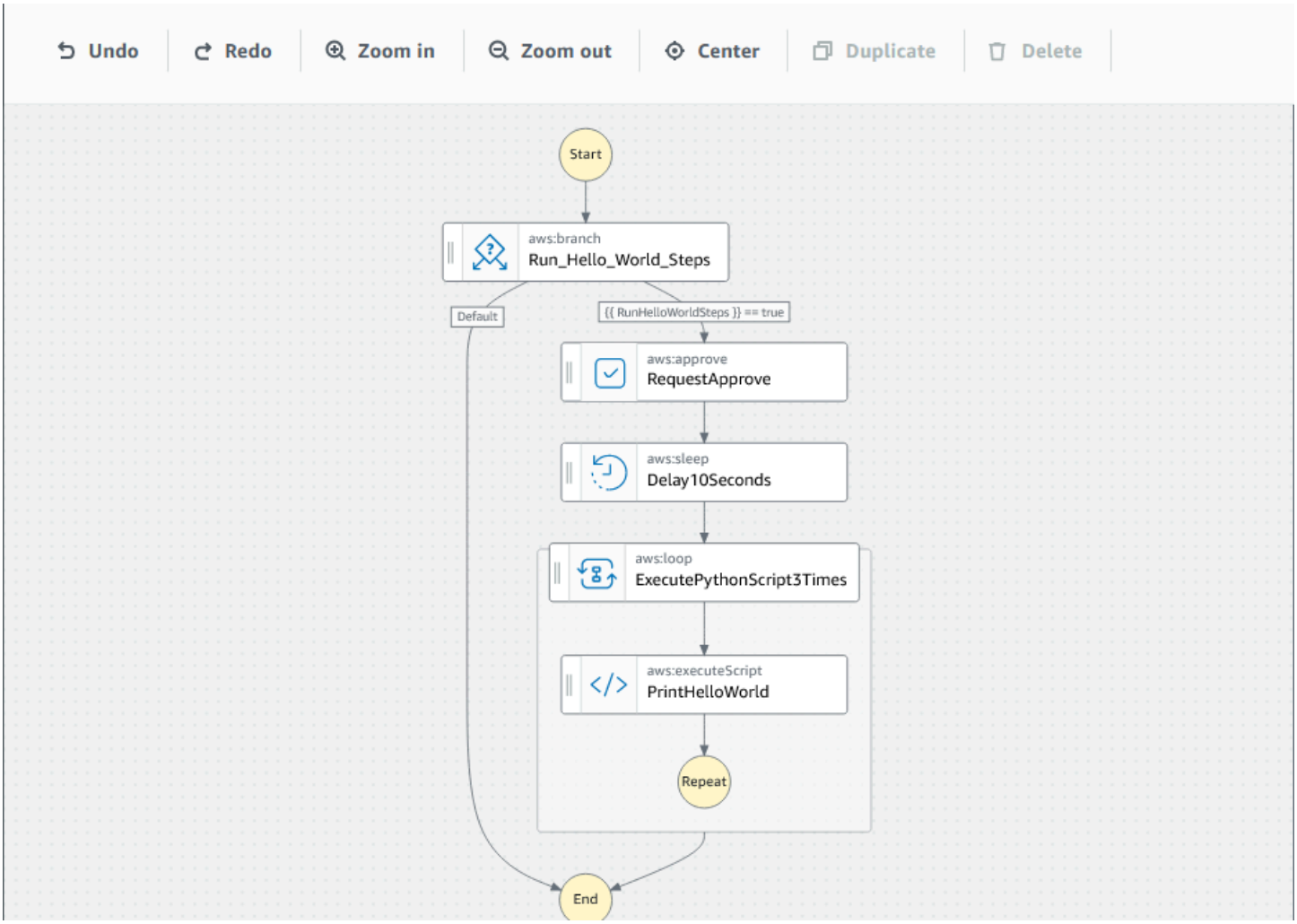
- 動作索引標籤提供自動化動作清單，您可以將這些動作拖放到畫布中執行手冊工作流程圖形中。
- AWS APIs 索引標籤提供 AWS APIs 的清單，您可以在畫布中拖放至 Runbook 的工作流程圖表。
- 執行手冊索引標籤提供了多個即用型、可重複使用的執行手冊做為建置區塊，您可用於各種用例。例如，您可以使用執行手冊在工作流程的 Amazon EC2 執行個體上執行常見的修復任務，而不必重新建立相同的動作。



Canvas

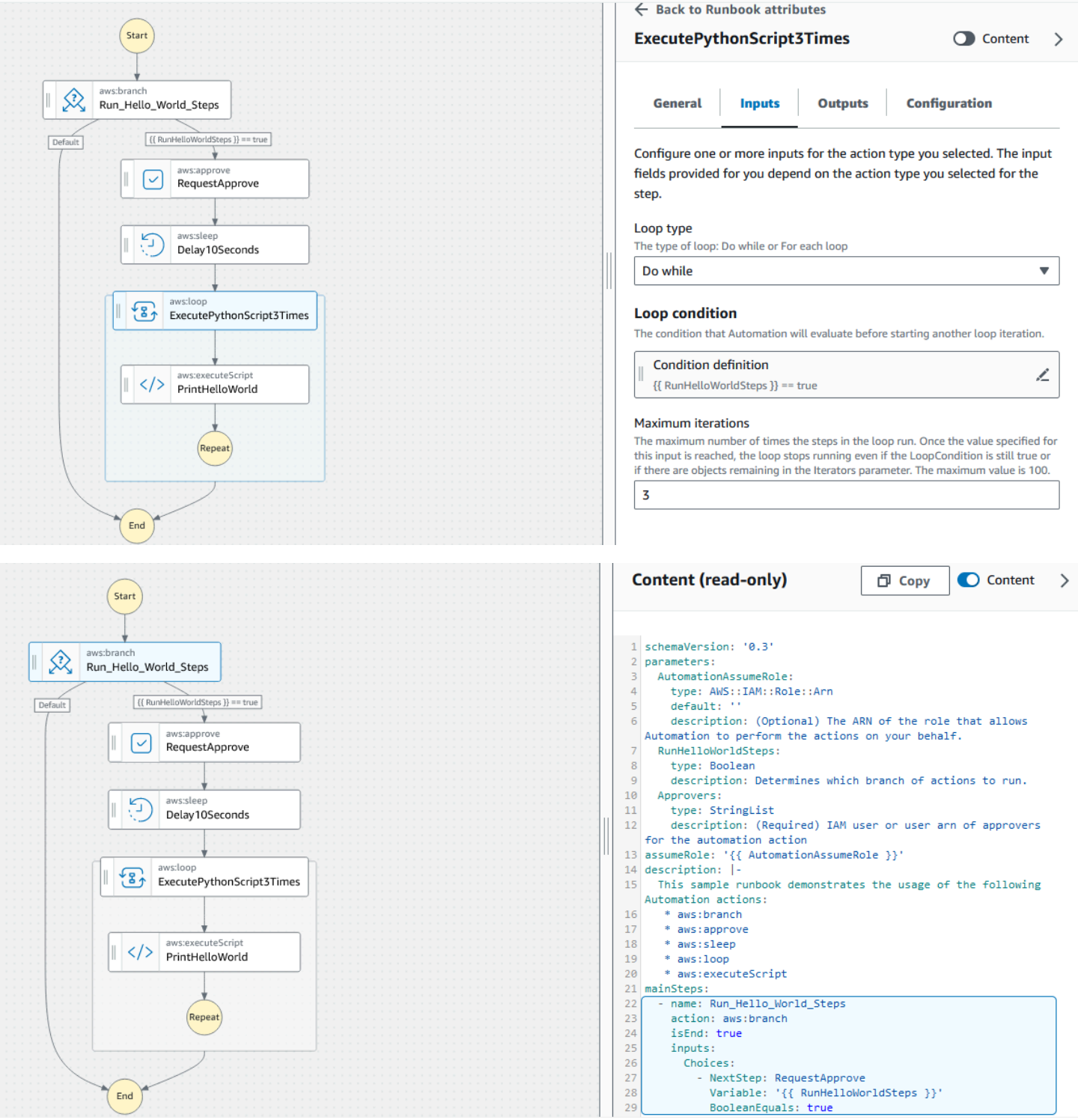
選擇要新增至自動化的動作後，將其拖曳至畫布並放入工作流程圖形中。您也可以拖放動作，將其移動到執行手冊工作流程的不同位置。如果工作流程很複雜，則您可能無法在畫布面板中檢視所有工作流程。使用畫布頂端的控制項來放大或縮小。若要檢視工作流程的不同部分，可以在畫布中拖曳工作流程圖形。

從動作瀏覽器中拖曳動作，將其放入執行手冊的工作流程圖形中。有一條線會顯示它將放置在工作流程中的位置。若要變更動作的順序，可以將其拖曳至工作流程中的其他位置。新動作已新增至您的工作流程，其程式碼會自動產生。



表格

將動作新增至執行手冊工作流程之後，您可以對其進行設定，以符合您的用例。選擇您要設定的動作，您就會在表單面板中看到其參數和選項。您也可以選擇內容按鈕，查看 YAML 或 JSON 程式碼。與您已選取的動作相關聯的程式碼會反白顯示。



鍵盤快速鍵

視覺化設計體驗支援下列資料表所示的鍵盤快速鍵。

鍵盤快速鍵

(復原) Ctrl+Z 上一個操作。

(重做) Ctrl+Shift+Z 上一個操作。

縮放至 100% 作流程置於畫布的中心。

鍵盤快速鍵

遍歷篩選條件

有選取的狀態。

刪除

Delete 除所有選取的狀態。

複製

Ctrl (或 Cmd) 選取的狀態。

運用視覺化設計體驗

了解如何使用視覺化設計體驗來建立、編輯和執行執行手冊工作流程。工作流程準備就緒後，您可以儲存或將其匯出。您也可以使用視覺化設計體驗來快速建立原型。

建立執行手冊工作流程

1. 登入 Systems Manager Automation 主控台。
2. 選擇建立執行手冊。
3. 在名稱方塊中輸入執行手冊的名稱，例如 *MyNewRunbook*。
4. 在設計和程式碼切換按鈕旁，選取鉛筆圖示，然後輸入執行手冊的名稱。

您現在可以為新的執行手冊設計工作流程。

設計執行手冊

若要使用視覺化設計體驗來設計執行手冊工作流程，您將自動化動作從動作瀏覽器拖曳到畫布中，將其放置在想要的工作流程中。您也可以將動作拖曳至其他位置，在工作流程中對動作重新排序。將動作拖曳到畫布上時，您可在工作流程中放置動作的任何位置會出現一條線。將動作拖放到畫布上後，其程式碼將自動產生並新增至您的執行手冊的內容中。

如果您知道要新增的動作名稱，則請使用動作瀏覽器頂端的搜尋方塊尋找動作。

將動作拖放到畫布上後，請使用右側的表單面板進行設定。此面板包含您放置在畫布上的每個自動化動作或 API 動作的一般、輸入、輸出和組態索引標籤。例如，一般索引標籤包含下列區段：

- 步驟名稱可用於識別步驟。為步驟名稱指定唯一值。
- 描述可協助您描述動作在您的執行手冊的工作流程中的作用。

輸入索引標籤包含的欄位會根據動作而有所不同。例如，`aws:executeScript` 自動化動作包含下列區域：

- 執行期是用於執行所提供指令碼的執行期語言。
- 處理常式是您的函數的名稱。您必須確保處理常式中定義的函數有兩個參數：`events` 和 `context`。PowerShell 執行期不支援此參數。
- 指令碼是您想要在工作流程期間執行的嵌入式指令碼。
- (選用) 附件適用於可由動作調用的獨立指令碼或 .zip 檔案。JSON 執行手冊需要此參數。

輸出索引標籤可協助您指定要從動作輸出的值。您可以在工作流程的後續動作中參考輸出值，或從動作產生輸出，以供日誌記錄之用。並非所有動作都支援輸出，因而並非所有動作都會有輸出索引標籤。例如，aws:pause 動作不支援輸出。對於支援輸出的動作，輸出索引標籤包含下列區段：

- 名稱是要用於輸出值的名稱。您可以在工作流程的後續動作中參考輸出。
- 選取器是以 "$." 開頭的 JSONPath 運算式字串，用於在 JSON 元素中選取一個或多個元件。
- 類型是輸出值的資料類型。例如，String 或 Integer 資料類型。

組態索引標籤包含所有自動化動作均可使用的屬性和選項。該動作由下列各部分組成：

- 嘗試次數上限屬性是動作失敗時重試的次數。
- 逾時秒屬性指定動作的逾時值。
- 是關鍵屬性決定動作失敗是否會停止整個自動化作業。
- 下一步屬性決定自動化在執行手冊中接下來執行的動作。
- 失敗時屬性決定如果動作失敗，自動化在執行手冊中接下來執行的動作。
- 取消時屬性會決定如果使用者取消動作，自動化在執行手冊中接下來執行的動作。

若要刪除動作，您可以使用退格鍵 (位於畫布上方的工具列)，或按一下滑鼠右鍵並選擇刪除動作。

隨著工作流程的增長，它可能不適應畫布。若要協助讓工作流程適應畫布，請嘗試下列選項之一：

- 使用側面板上的控制項，調整面板的大小或關閉面板。
- 使用畫布頂端的工具列可放大或縮小工作流程圖形。

更新執行手冊

您可以透過建立新版本的執行手冊，更新現有的執行手冊工作流程。您可以使用視覺化設計體驗或直接編輯程式碼，更新您的執行手冊。請使用下列程序來更新現有執行手冊：

1. 登入 Systems Manager Automation 主控台。
2. 選擇您要更新的執行手冊。
3. 選擇 Create new version (建立新版本)。
4. 視覺化設計體驗包含兩個窗格：程式碼窗格和視覺化工作流程窗格。在視覺化工作流程窗格中選擇設計，以透過視覺化設計體驗編輯工作流程。完成後，請選擇建立新版本，以儲存變更並退出。
5. (選用) 使用程式碼窗格編輯 YAML 或 JSON 中的執行手冊內容。

匯出執行手冊

若要匯出執行手冊工作流程的 YAML 或 JSON 程式碼，以及工作流程的圖表，請使用下列程序：

1. 在文件主控台中選擇您的執行手冊。
2. 選擇 Create new version (建立新版本)。
3. 在動作下拉式清單中，選擇要匯出圖形或執行手冊，以及您偏好的格式。

設定動作的輸入和輸出

每個自動化動作都會根據其收到的輸入作出回應。在大多數情況下，您可將輸出傳遞給後續動作。在視覺化設計體驗中，您可以在表單面板的輸入和輸出索引標籤設定動作的輸入和輸出資料。

如需如何定義和使用自動化動作輸出的詳細資訊，請參閱 使用動作輸出作為輸入。

為動作提供輸入資料

每個自動化動作都有一或多個您必須為其提供值的輸入。您為動作的輸入提供的值取決於動作所接受的資料類型和格式。例如，`aws:sleep` 動作需要 `Duration` 輸入的 ISO 8601 格式字串值。

通常，您可以在執行手冊的工作流程中使用動作，該動作會傳回您要在後續動作中使用的輸出。請務必確保輸入值正確無誤，以避免執行手冊工作流程中發生錯誤。輸入值也很重要，因為會決定動作是否傳回預期的輸出。例如，使用 `aws:executeAwsApi` 動作時，您需要確保為 API 操作提供正確的值。

定義動作的輸出資料

某些自動化動作會在執行其定義的操作後傳回輸出。傳回輸出的動作具有預先定義的輸出，或能讓您自行定義輸出。例如，`aws:createImage` 動作具有傳回 `ImageId` 和 `ImageState` 的預先定義輸出。相比之下，使用 `aws:executeAwsApi` 動作，您可以從指定的 API 操作中定義所需的輸出。因此，您可以從單一 API 操作傳回一或多個值，以便在後續動作中使用。

您必須指定輸出的名稱、資料類型和輸出值，方能定義自己的自動化動作輸出。若要繼續使用 `aws:executeAwsApi` 動作作為範例，假設您正在從 Amazon EC2 呼叫 `DescribeInstances` API 操作。在此範例中，您想要傳回或輸出 Amazon EC2 執行個體的 `State`，並根據輸出對執行手冊的工作流程進行分支。您可以選擇將輸出命名為 **`InstanceState`**，並使用 **`String`** 資料類型。

定義輸出實際值的程序會取決於動作而略有差異。例如，若您正在使用 `aws:executeScript` 動作，則必須在函數中使用 `return` 陳述式，為輸出提供資料。使用其他動作 (如 `aws:executeAwsApi`、`aws:waitForAwsResourceProperty` 和

aws:assertAwsResourceProperty) 會需要 Selector。或正如某些動作所參考的，Selector 或 PropertySelector 是用來處理來自 API 操作的 JSON 回應的 JSONPath 字串。了解來自 API 操作的 JSON 回應物件的結構非常重要，以便您為輸出選擇正確的值。使用前面提到的 DescribeInstances API 操作，請參閱下面的範例 JSON 回應：


```
{
  "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
          }
        }
      }
    }
```

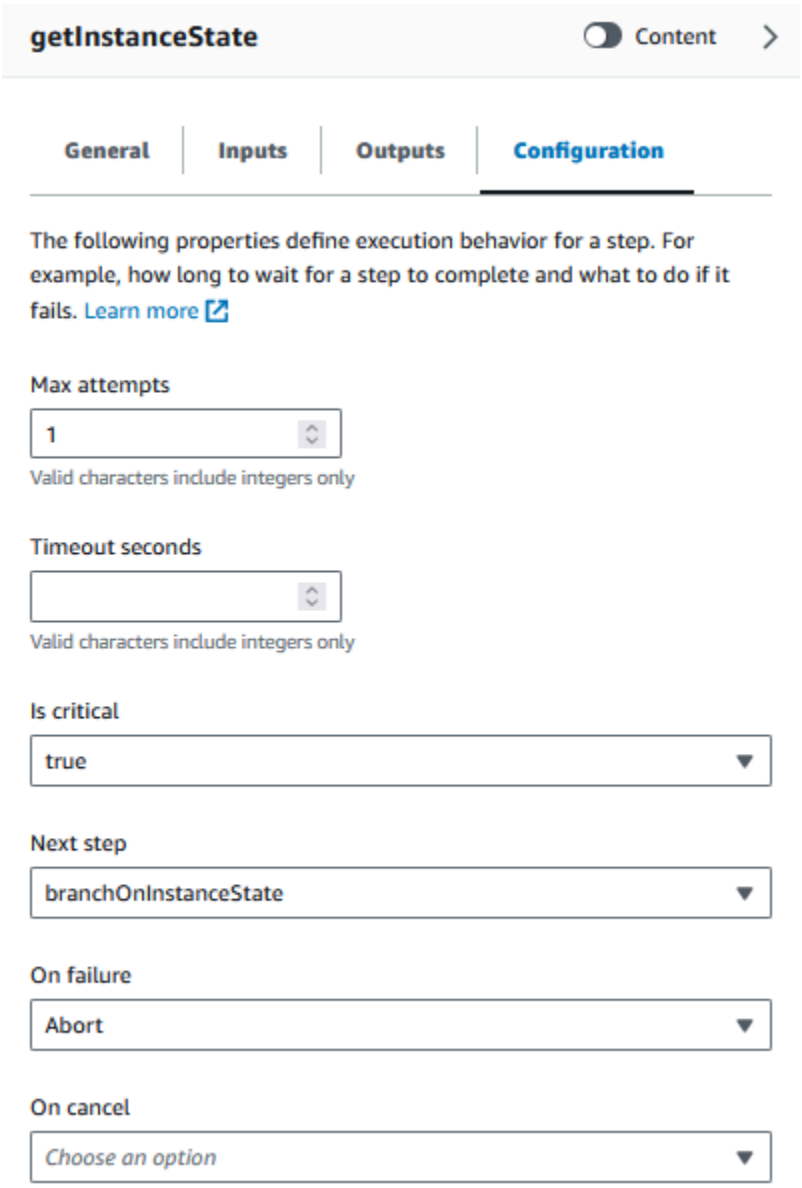
```
  }
}
```

在 JSON 回應物件中，執行個體 State 在 Instances 物件中形成巢狀，該物件又在 Reservations 物件中形成巢狀。若要傳回執行個體 State 的值，請為 Selector 使用下列字串，以便在我們的輸出中使用該值：**$.Reservations[0].Instances[0].State.Name**。

若要在執行手冊工作流程的後續動作中參照輸出值，請使用下列格式：{{ *StepName*.*NameOfOutput* }}。例如：**{{ GetInstanceState.InstanceState }}**。在視覺化設計體驗中，您可以使用輸入的下拉式清單，選擇要在後續動作中使用的輸出值。在後續動作中使用輸出時，輸出的資料類型必須與輸入的資料類型相符。在此範例中，InstanceState 輸出為 String。因此，若要在後續動作的輸入中使用該值，輸入必須接受 String。

藉助視覺化設計體驗錯誤處理

根據預設，當動作報告錯誤時，Automation 會完全停止執行手冊的工作流程。這是因為所有動作的 onFailure 屬性預設值為 Abort。您可以設定 Automation 如何處理您的執行手冊工作流程中的錯誤。即使已設定錯誤處理，某些錯誤仍可能導致自動化操作失敗。如需詳細資訊，請參閱故障診斷 Systems Manager Automation。在視覺化設計體驗中，您可以在組態面板設定錯誤處理。



出現錯誤時重試動作

若要在出現錯誤時重試動作，請指定嘗試次數上限屬性的值。預設值為 1。如果指定的值大於 1，則在所有重試嘗試失敗之前，動作不會視為失敗。

逾時

您可設定動作逾時，以設定動作失敗前可執行的秒數上限。若要設定逾時，請在逾時秒屬性中輸入動作失敗之前，動作應等待的秒數。如果達到逾時且動作的 `Max attempts` 值大於 1，則在完成所有重試之前，步驟不會視為逾時。

失敗的動作

根據預設，當動作失敗時，Automation 會完全停止執行手冊的工作流程。您可以透過為執行手冊中動作的失敗時屬性指定替代值，以修改此行為。如果您希望工作流程繼續執行執行手冊中的下個步驟，則請選擇繼續。如果您希望工作流程跳至執行手冊中的其他後續步驟，則請選擇步驟，然後輸入步驟的名稱。

取消的動作

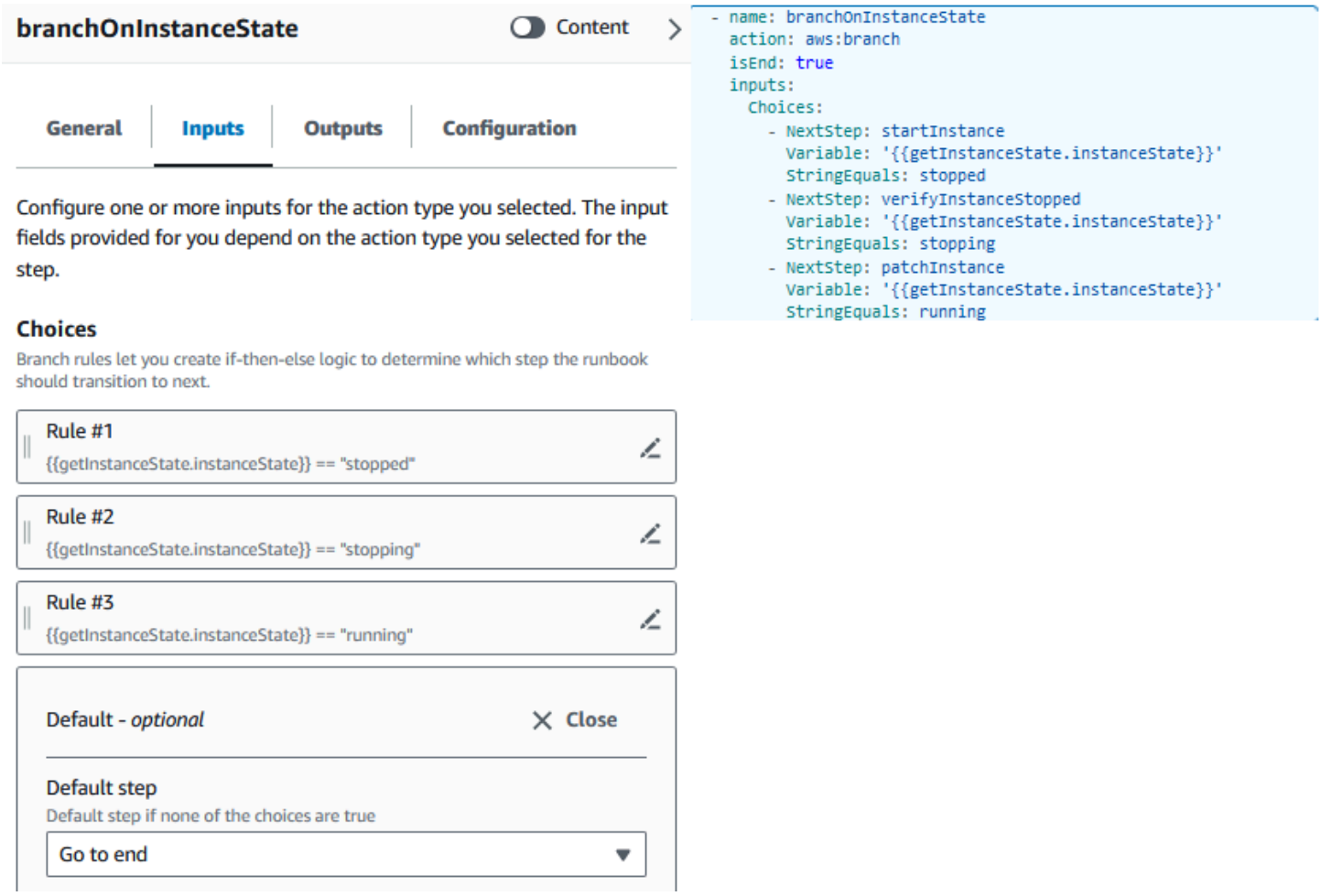
根據預設，當使用者取消動作時，Automation 會完全停止執行手冊的工作流程。您可以透過為執行手冊中動作的取消時屬性指定替代值，以修改此行為。如果您希望工作流程跳至執行手冊中的其他後續步驟，則請選擇步驟，然後輸入步驟的名稱。

關鍵動作

您可以將某個動作指定為關鍵動作，這表示它會決定了自動化操作的整體報告狀態。如果此指定步驟失敗，則 Automation 會將最終狀態報告為 `Failed`，不論其他動作是否成功。若要將動作設定為關鍵，請將為關鍵屬性的預設值保留為 True。

結束動作

為結束屬性會在指定動作結束時停止自動化。此屬性的預設值為 `false`。如果您為動作設定此屬性，則無論動作成功還是失敗，自動化都會停止。此屬性最常與 `aws:branch` 動作搭配使用，以處理非預期或未定義的輸入值。下列範例顯示預期執行個體狀態為 `running`、`stopping` 或 `stopped` 的執行手冊。如果執行個體處於不同的狀態，則自動化將結束。



教學課程：使用視覺化設計體驗建立執行手冊

在本教學課程中，您將了解使用 Systems Manager Automation 提供的視覺化設計體驗的基礎概念。在視覺化設計體驗中，您可以建立使用多個動作的執行手冊。您可使用拖放功能來排列畫布上的動作。您也可搜尋、選取和設定這些動作。然後，您可檢視執行手冊工作流程自動產生的 YAML 程式碼，退出結束視覺化設計體驗，執行該執行手冊，以及檢閱執行詳細資料。

本教學課程還將向您展示如何更新執行手冊並查看新版本。在教學課程結束時，您可執行清理步驟並刪除執行手冊。

完成本教學課程之後，您就會知道如何使用視覺化設計體驗來建立執行手冊。您還將知道如何更新、執行和刪除您的執行手冊。

> **Note**
>
> 在您開始教學課程之前，請務必先完成 設定自動化。

主題

- 步驟 1：導覽至視覺化設計體驗
- 步驟 2：建立工作流程
- 步驟 3：檢閱自動產生的程式碼
- 步驟 4：執行新的執行手冊
- 步驟 5：清除

步驟 1：導覽至視覺化設計體驗

1. 登入 Systems Manager Automation 主控台。
2. 選擇建立自動化以儲存執行手冊。

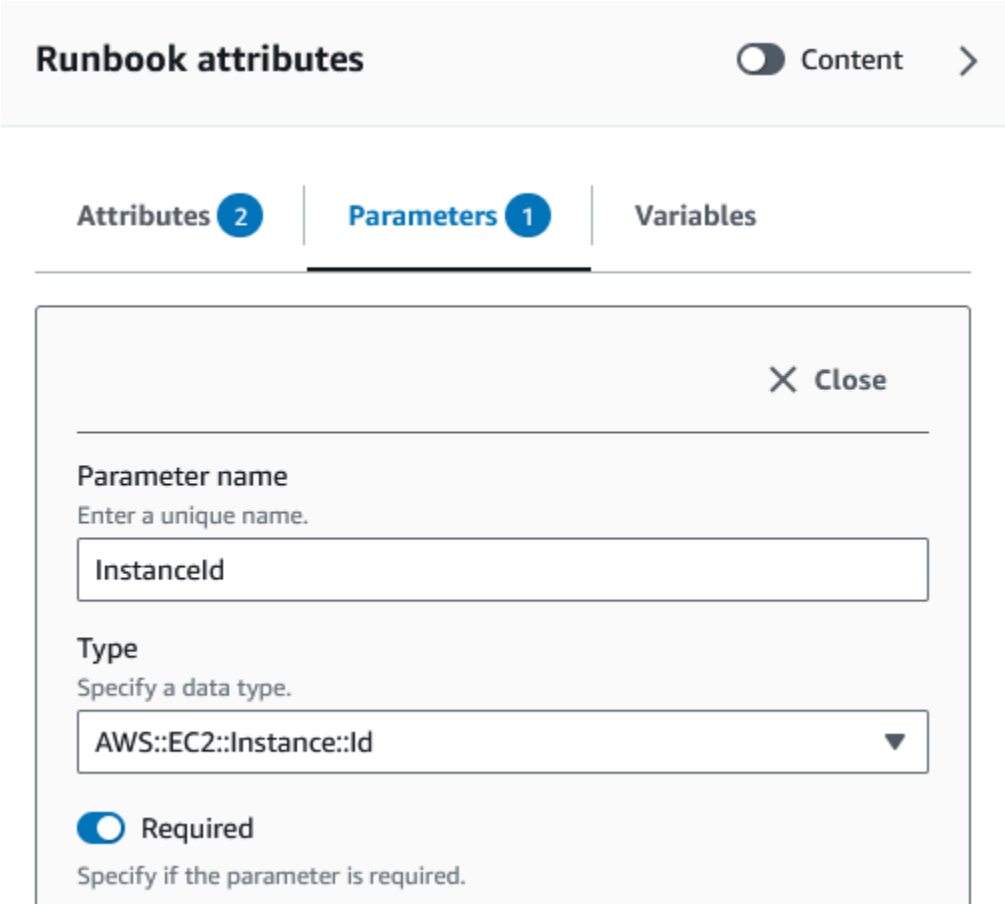
步驟 2：建立工作流程

在視覺化設計體驗中，工作流程是畫布上您的執行手冊的圖形表示。您可以使用視覺化設計體驗來定義、設定和檢查執行手冊的個別動作。

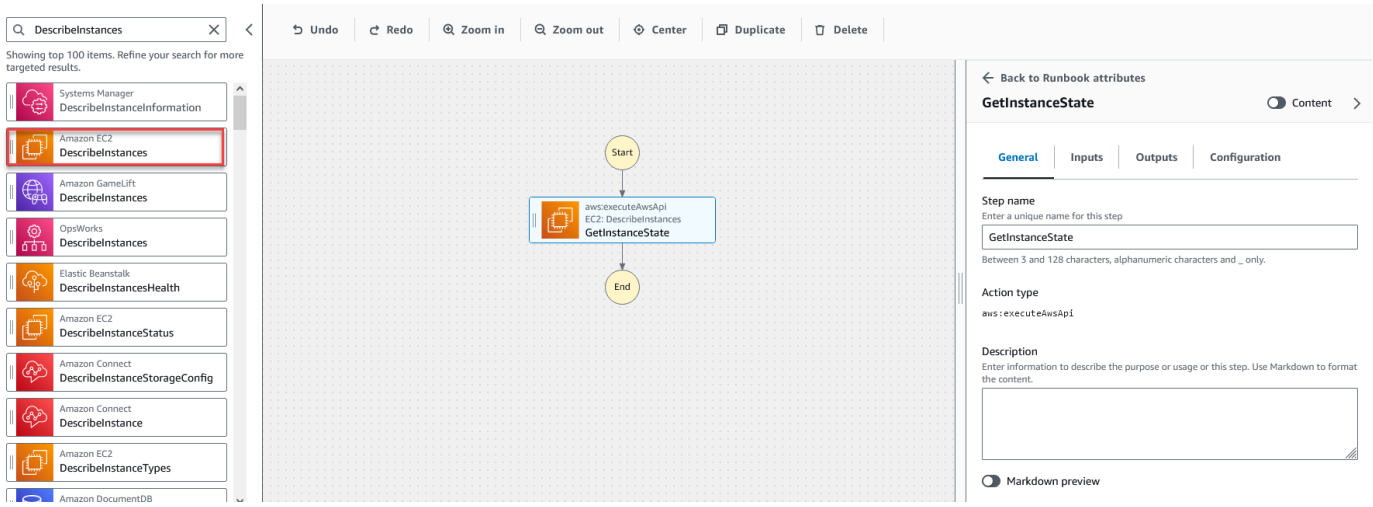
若要建立工作流程

1. 在設計和程式碼切換按鈕旁，選取鉛筆圖示，然後輸入執行手冊的名稱。針對本教學，輸入 **`VisualDesignExperienceTutorial`**。

2. 在表單面板的文件屬性區段中，展開輸入參數下拉式清單，然後選取新增參數。
   a. 在參數名稱欄位中，輸入 **`InstanceId`**。
   b. 在類型下拉式清單中，選擇 AWS::EC2::Instance。
   c. 選取必要切換按鈕。



3. 在 AWS API 瀏覽器中，在搜尋列輸入 **`DescribeInstances`**。
4. 將 Amazon EC2 - DescribeInstances 動作拖曳至空白畫布。
5. 對於步驟名稱，輸入值。在本教學課程中，您可以使用名稱 **`GetInstanceState`**。



   a. 展開其他輸入下拉式清單，然後在輸入名稱欄位中輸入 **`InstanceIds`**。

b. 選擇輸入索引標籤。

c. 在輸入值欄位中，選擇 **InstanceId** 文件輸入。這會參考您在程序開始時建立的輸入參數值。由於 DescribeInstances 動作的 InstanceIds 輸入接受 StringList 值，因此您必須以方括號包裝 InstanceId 輸入。輸入值的 YAML 應符合下方項目：**['{{ InstanceId }}']**。

d. 在輸出索引標籤中，選取新增輸出，然後在名稱欄位中輸入 **InstanceState**。

e. 在選取器欄位中輸入 **$.Reservations[0].Instances[0].State.Name**。

f. 在類型下拉式清單中選擇字串。

6. 從動作瀏覽器拖曳分支動作，並將其放置在 **GetInstanceState** 步驟下方。

7. 對於步驟名稱，輸入值。在本教學課程中，使用名稱 **BranchOnInstanceState**。

   若要定義分支邏輯，請執行下列操作：

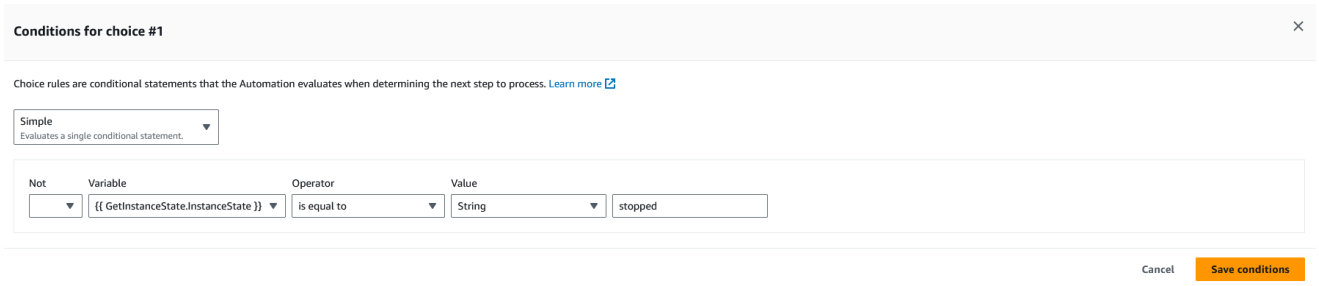
   a. 在畫布上選擇 **Branch** 狀態。然後，在輸入和選擇下，選取鉛筆圖示，以編輯規則 #1。

   b. 選擇新增條件。

   c. 在規則 #1 的條件對話方塊中，從變數下拉式清單中選擇 **GetInstanceState.InstanceState** 步驟輸出。

   d. 對於運算子，選擇等於。

   e. 對於值，從下拉式清單中選擇字串。輸入 **stopped**。



   f. 選取儲存條件。

   g. 選擇新增新的規則。

   h. 選擇規則 #2 的新增條件。

   i. 在規則 #2 的條件對話方塊中，從變數下拉式清單中選擇 **GetInstanceState.InstanceState** 步驟輸出。

   j. 對於運算子，選擇等於。

   k. 對於值，從下拉式清單中選擇字串。輸入 **stopping**。

   l. 選取儲存條件。

m. 選擇新增新的規則。

n. 對於規則 #3，選擇新增條件。

o. 在規則 #3 的條件對話方塊中，從變數下拉式清單中選擇 **GetInstanceState.InstanceState** 步驟輸出。

p. 對於運算子，選擇等於。

q. 對於值，從下拉式清單中選擇字串。輸入 **running**。

r. 選取儲存條件。

s. 在預設規則中，針對預設步驟選擇移至結尾。

8. 將變更執行個體狀態動作拖曳至 {{ GetInstanceState.InstanceState }} == "stopped" 條件下方空白的拖曳至此處方塊。

a. 對於步驟名稱，輸入 **StartInstance**。

b. 在輸入索引標籤的執行個體 ID 下，從下拉式清單中選擇 InstanceId 文件輸入值。

c. 對於所需狀態，請指定 **running**。

9. 將等待 AWS 資源動作拖曳至 {{ GetInstanceState.InstanceState }} == "stopping" 條件下的空白拖曳動作此處方塊。

10. 對於步驟名稱，輸入值。在本教學課程中，使用名稱 **WaitForInstanceStop**。

a. 在服務欄位中，選擇 Amazon EC2。

b. 在 API 欄位中，選擇 DescribeInstances。

c. 在屬性選取器欄位中輸入 **$.Reservations[0].Instances[0].State.Name**。

d. 對於所需值參數，輸入 **["stopped"]**。

e. 在 WaitForInstanceStop 動作的組態索引標籤中，從下一步下拉式清單中選擇 StartInstance。

11. 將在執行個體上執行命令動作拖曳至 {{ GetInstanceState.InstanceState }} == "running" 條件下方空白的拖曳至此處方塊。

12. 對於步驟名稱，輸入 **SayHello**。

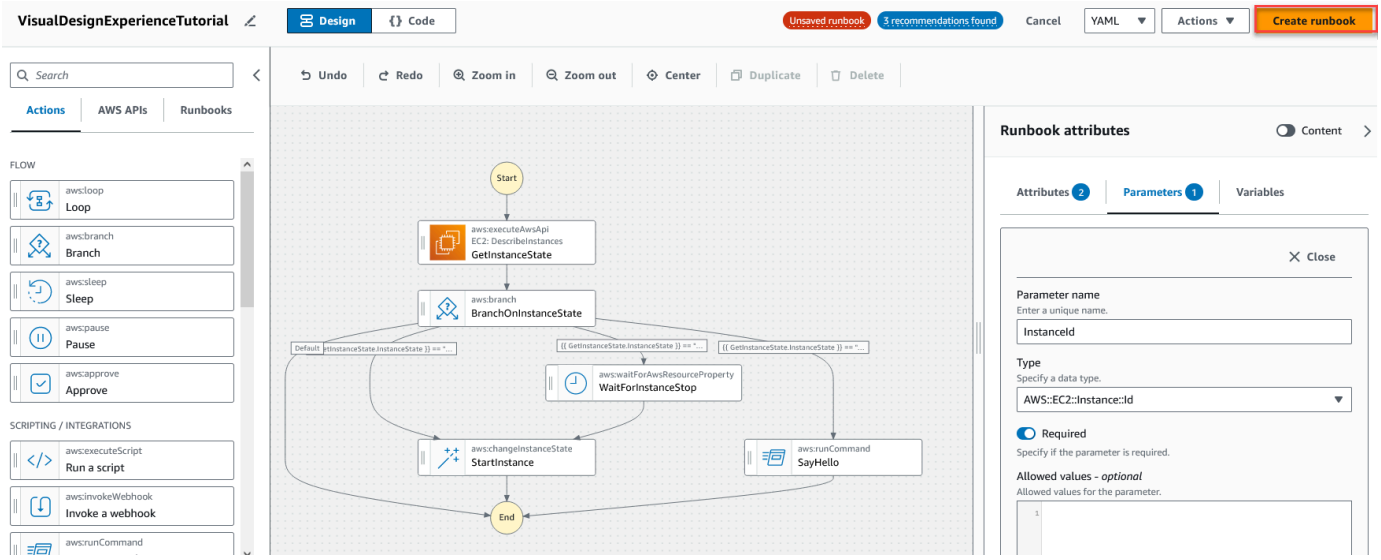
a. 在輸入索引標籤中，為文件名稱參數輸入 **AWS-RunShellScript**。

b. 對於 InstanceIds，從下拉式清單中選擇執行個體 ID 文件輸入值。

c. 展開其他輸入下拉式清單，然後在輸入名稱下拉清單中選擇參數。

d. 在輸入值欄位中輸入 **{"commands": "echo 'Hello World'"}**。

13. 檢閱畫布中已完成的執行手冊，然後選取建立手冊以儲存教學執行手冊。



步驟 3：檢閱自動產生的程式碼

當您將動作從動作瀏覽器拖放到畫布上時，視覺化設計體驗會即時自動撰寫執行手冊的 YAML 或 JSON 內容。您可檢閱和編輯此程式碼。若要檢視自動產生的程式碼，請為設計和程式碼切換按鈕選取程式碼。

步驟 4：執行新的執行手冊

建立您的執行手冊後，您可以執行自動化。

若要執行新的自動化執行手冊

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Automation (自動化)，接著選擇 Execute automation (執行自動化)。
3. 在 Automation document (自動化文件)清單中，選擇 Runbook。在 Document categories (文件類別) 窗格中選擇一個或多個選項，根據 SSM 文件的用途來進行篩選。若要檢視您擁有的 Runbook，請選擇 Owned by me (我所擁有的) 索引標籤。若要檢視與您帳戶共用的 Runbook，請選擇 Shared with me (與我共用的) 索引標籤。若要檢視所有 Runbook，請選擇 All documents (所有文件) 索引標籤。

> **Note**
>
> 您可以選擇 Runbook 名稱檢視 Runbook 資訊。

4. 在 Document details (文件詳細資訊) 部分，確認 Document version (文件版本) 設定為您想要執行的版本。系統包括以下版本選項：

   - 執行期的預設版本：如果 Automation 執行手冊會定期更新且已指派新的預設版本，則請選擇此選項。
   - 執行期的最新版本：如果 Automation 執行手冊會定期更新，而您想要執行最近更新的版本，請選擇此選項。
   - 1 (預設)：選擇此選項以執行文件的第一個版本，也是預設版本。

5. 選擇 Next (下一步)。
6. 在執行自動化執行手冊章節中，選擇簡易執行。
7. 在 Input parameters (輸入參數) 部分，指定所需的輸入。或者，您也可以從 AutomationAssumeRole 清單中選擇 IAM 服務角色。
8. (選用) 選擇要套用至您的自動化以便加以監控的 Amazon CloudWatch 警示。若要將 CloudWatch 警示連接至您的自動化，啟動自動化的 IAM 主體必須具備 `iam:createServiceLinkedRole` 動作的許可。如需有關 CloudWatch 警示的詳細資訊，請參閱使用 Amazon CloudWatch 警示。如果您的警示啟用，則會停止自動化。如果使用 AWS CloudTrail，則您會在追蹤中看到 API 呼叫。
9. 選擇 Execute (執行)。

步驟 5：清除

若要刪除您的執行手冊

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Documents (文件)。
3. 選擇我所擁有索引標籤。
4. 找到 VisualDesignExperienceTutorial 執行手冊。
5. 選取文件卡頁面上的按鈕，然後從動作下拉式清單中選擇刪除文件。

撰寫 Automation Runbook

Automation 中的每個 Runbook 都是其中的工具 AWS Systems Manager，可定義自動化。Automation Runbook 會定義在自動化期間執行的動作。在 Runbook 內容中，您可以定義 Systems Manager 在受管執行個體 AWS 和資源上執行的輸入參數、輸出和動作。

自動化包含數個預先定義的 Runbook，供您用來執行常見任務，像是重新啟動一個或多個 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體，或建立 Amazon Machine Image (AMI)。不過，您的使用案例可能會超出預先定義 Runbook 的功能。如果是這種情況，您可以建立自己的 Runbook，並根據您的需求進行修改。

Runbook 包含自動化動作、這些動作的參數，以及您指定的輸入參數。Runbook 的內容是以 YAML 或 JSON 撰寫。如果您不熟悉 YAML 或 JSON，則建議在嘗試撰寫自己的執行手冊之前使用視覺化設計工具，或了解更多關於任一標記語言的資訊。如需有關視覺化設計工具的詳細資訊，請參閱 Automation 執行手冊的視覺化設計體驗。

下列各節將協助您撰寫首個 Runbook。

識別您的使用案例

撰寫 Runbook 的第一個步驟是識別您的使用案例。例如，您排定了每天在所有生產 Amazon EC2 執行個體上執行的 `AWS-CreateImage` Runbook。在月底，您會決定是否擁有超過復原點所需數量的映像。接下來，在建立新 AMI 時，您想要自動刪除最舊的 Amazon EC2 執行個體 AMI。若要完成這項操作，您可以建立執行以下動作的新 Runbook：

1. 執行 `aws:createImage` 動作，並在映像描述中指定執行個體 ID。
2. 在 `available` 前，執行 `aws:waitForAwsResourceProperty` 動作來輪詢圖像的狀態。
3. 映像狀態為 `available` 後，`aws:executeScript` 動作會執行自訂 Python 指令碼，該指令碼會收集與您的 Amazon EC2 執行個體相關聯的所有映像 ID。指令碼會使用您在建立時指定之映像描述中的執行個體 ID 進行篩選。然後，指令碼會根據映像的 `creationDate` 對映像 ID 進行排序，並輸出最舊 AMI 的 ID。
4. 最後，`aws:deleteImage` 動作會使用上一個步驟輸出的 ID 刪除最舊的 AMI。

在這個案例中，您已經使用 `AWS-CreateImage` Runbook，但發現您的使用案例需要更大的靈活性。這是常見的情況，因為 Runbook 和自動化動作之間可能會有重疊。因此，您可能必須調整您用於處理使用案例的 Runbook 或動作。

例如，`aws:executeScript` 和 `aws:invokeLambdaFunction` 動作都允許您在自動化過程中執行自訂指令碼。若要在兩者之間進行選擇，因為其他支援的執行時間語言，您可能偏好 `aws:invokeLambdaFunction`。但是，您可能偏好 `aws:executeScript`，因為它允許您直接在 YAML Runbook 中撰寫指令碼內容，並提供指令碼內容作為 JSON Runbook 的附件。您也可以考慮 `aws:executeScript`，在 AWS Identity and Access Management (IAM) 設定更簡單。因為它使用中提供的許可`AutomationAssumeRole`，`aws:executeScript`因此不需要額外的 AWS Lambda 函數執行角色。

在任何給定的情況下，相較於另一個動作，一個動作可能會提供更多的靈活性或新增的功能。因此，建議您檢閱要使用之 Runbook 或動作的可用輸入參數，以判斷哪個最適合您的使用案例和偏好設定。

設定開發環境

識別您的使用案例以及您想要在 Runbook 中使用的預先定義 Runbook 或自動化動作之後，便是時候為 Runbook 內容設定開發環境了。若要開發您的 Runbook 內容，我們建議您使用 AWS Toolkit for Visual Studio Code ，而非 Systems Manager 文件主控台。

Toolkit for VS Code 是 Visual Studio 程式碼 (VS 程式碼) 的開放原始碼延伸項目，提供比 Systems Manager 文件主控台更多的功能。實用的功能包括 YAML 和 JSON 的結構描述驗證、自動化動作類型的程式碼片段，以及對 YAML 和 JSON 各種選項的自動完成支援。

如需安裝 Toolkit for VS Code 的詳細資訊，請參閱安裝 AWS Toolkit for Visual Studio Code。如需使用 Toolkit for VS Code 開發 Runbook 的相關資訊，請參閱《AWS Toolkit for Visual Studio Code 使用者指南》中的使用 Systems Manager Automation 文件。

開發 Runbook 內容

識別使用案例並設定環境後，即可準備開發 Runbook 適用的內容。您的使用案例和偏好設定主要會決定您在 Runbook 內容中使用的自動化動作或 Runbook。與允許您完成類似任務的另一個動作相比，某些動作僅支援輸入參數的子集。其他動作具有特定輸出，例如 `aws:createImage`，其中一些動作允許您定義自己的輸出，例如 `aws:executeAwsApi`。

如果您不確定如何在 Runbook 中使用特定動作，建議您檢閱 Systems Manager Automation 動作參考中動作的對應項目。也建議您檢閱預先定義的 Runbook 內容，以查看如何使用這些動作的真實世界範例。如需 Runbook 真實世界應用程式的更多範例，請參閱 其他執行手冊範例。

為了展示 Runbook 內容提供的簡單性和靈活性差異，下列教學課程提供如何分階段修補 Amazon EC2 執行個體群組的範例：

- the section called “範例 1：建立父子 Runbook” – 在此範例中，兩個 Runbook 會用於父子關係中。父系 Runbook 會啟動子系 Runbook 的速率控制自動化。
- the section called “範例 2：指令碼式 Runbook” – 此範例示範如何透過將內容壓縮成單一 Runbook 並在 Runbook 中使用指令碼，來完成範例 1 的相同任務。

範例 1：建立父子 Runbook

以下範例演示如何建立兩個 Runbook，以分階段修補加上標籤的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體群組。這些 Runbook 用於父子關係中，其中父系 Runbook 用來起始子系

Runbook 的速率控制自動化。如需速率控制自動化的詳細資訊，請參閱 大規模執行自動化操作。如需此範例中所使用自動化動作的詳細資訊，請參閱 Systems Manager Automation 動作參考。

建立子系 Runbook

此範例 Runbook 會處理以下案例。Emily 是 AnyCompany Consultants, LLC 的系統工程師。她需要針對託管主要和次要資料庫的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體群組設定修補。應用程式每天 24 小時存取這些資料庫，因此其中一個資料庫執行個體必須永遠可用。

她認為分階段修補執行個體是最好的方法。先修補資料庫執行個體的主要群組，然後再修補資料庫執行個體的次要群組。此外，為了避免讓先前停用的執行個體因執行而產生額外的成本，Emily 希望修補的執行個體在修補發生之前恢復到原始狀態。

Emily 透過與執行個體相關聯的標籤來識別資料庫執行個體的主要和次要群組。她決定建立啟動子系 Runbook 速率控制自動化的父系 Runbook。透過如此操作，她可以鎖定與資料庫執行個體之主要和次要群組相關聯的標籤，並管理子系自動化的並行性。在檢閱可用 Systems Manager (SSM) 文件以進行修補之後，她選擇 `AWS-RunPatchBaseline` 文件。透過使用此 SSM 文件，她的同事可以在修補操作完成後，檢閱相關聯的修補程式合規資訊。

若要開始建立她的 Runbook 內容，Emily 會檢閱可用的自動化動作，並開始撰寫子系 Runbook 的內容，如下所示：

1. 首先，她提供 Runbook 結構描述和描述的值，並定義子系 Runbook 的輸入參數。

   透過使用 `AutomationAssumeRole` 參數，Emily 和她的同事可以使用現有 IAM 角色，允許 Automation 代表他們執行 Runbook 中的動作。Emily 使用 `InstanceId` 參數來決定應該修補的執行個體。(選用) `Operation`、`RebootOption` 和 `SnapshotId` 參數可以用來提供值來記錄 `AWS-RunPatchBaseline` 的文件參數。為了防止提供無效值給這些文件參數，她會視需要定義 `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. 定義頂層元素後，Emily 會繼續撰寫構成 Runbook `mainSteps` 的動作。第一個步驟會輸出目標執行個體的目前狀態，而執行個體是使用 `aws:executeAwsApi` 動作在 `InstanceId` 輸入參數中指定的。此動作的輸出會用於稍後的動作。

   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 不是手動啟動和追蹤需要修補之每個執行個體的原始狀態，而是使用上一個動作的輸出，根據目標執行個體的狀態分支自動化。這樣可讓自動化執行不同的步驟，取決於 aws:branch 動作中定義的條件，並提高自動化的整體效率，而無需人工介入。

   如果執行個體的狀態已經是 running，則自動化會繼續使用 aws:runCommand 動作修補具有 AWS-RunPatchBaseline 文件的執行個體。

   如果執行個體的狀態為 stopping，使用 aws:waitForAwsResourceProperty 動作讓執行個體的自動化輪詢達到 stopped 狀態，使用 executeAwsApi 動作啟動執行個體，對執行個體進行輪詢以達到 running 狀態，然後再修補執行個體。

   如果執行個體的狀態為 stopped，則使用相同的動作，在修補執行個體前，自動化會啟動執行個體並對其進行輪詢，以達到 running 狀態。

   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. 修補操作完成之後，Emily 想要自動化將目標執行個體恢復到自動化開始之前的相同狀態。她透過再次使用第一個動作的輸出，完成此操作。使用 aws:branch 動作，自動化會根據目標執行個體的原始狀態進行分支。如果執行個體先前處於除了 running 以外的任何狀態，則執行個體會停止。否則，如果執行個體狀態為 running，則自動化會結束。

   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 會檢閱已完成的子 Runbook 內容，並在與目標執行個體相同的 AWS 帳戶 和 AWS 區域 中建立 Runbook。現在，她已準備好繼續建立父系 Runbook 的內容。以下是已完成的子系 Runbook 內容。

   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}}"
            ]
         }
      }
   ]
}
```

如需此範例中所使用自動化動作的詳細資訊，請參閱 Systems Manager Automation 動作參考。

建立父系 Runbook

此範例 Runbook 會繼續上一節所述的案例。現在 Emily 已經建立子系 Runbook，她開始撰寫父系 Runbook 的內容，如下所示：

1. 首先，她提供 Runbook 結構描述和描述的值，並定義父系 Runbook 的輸入參數。

   透過使用 AutomationAssumeRole 參數，Emily 和她的同事可以使用現有 IAM 角色，允許 Automation 代表他們執行 Runbook 中的動作。Emily 使用 PatchGroupPrimaryKey 和 PatchGroupPrimaryValue 參數來指定與要修補之資料庫執行個體主要群組相關聯的標籤。她使用 PatchGroupSecondaryKey 和 PatchGroupSecondaryValue 參數來指定與要修補之資料庫執行個體次要群組相關聯的標籤。

   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. 定義頂層元素後，Emily 會繼續撰寫構成 Runbook mainSteps 的動作。

   第一個動作會使用她剛才建立的、以與 PatchGroupPrimaryKey 和 PatchGroupPrimaryValue 輸入參數中指定標籤關聯之執行個體為目標的子系 Runbook，啟動速率控制自動化。她使用提供給輸入參數的值來指定與要修補之資料庫執行個體主要群組相關聯的標籤索引鍵和值。

   第一個自動化完成之後，第二個動作會使用以與 PatchGroupSecondaryKey 和 PatchGroupSecondaryValue 輸入參數中指定標籤關聯之執行個體為目標的子系 Runbook 啟動另一個速率控制自動化。她使用提供給輸入參數的值來指定與要修補之資料庫執行個體次要群組相關聯的標籤索引鍵和值。

   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 會檢閱完成的父系 Runbook 內容，並在 AWS 區域 與目標執行個體相同的 AWS 帳戶 和 中建立 Runbook。現在，她已經準備好測試她的 Runbook，以確保自動化能夠依需要操作，然後再將其實作到她的生產環境中。以下是已完成的父系 Runbook 內容。

   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"
```

```
            }
        }
    ]
}
```

如需此範例中所使用自動化動作的詳細資訊，請參閱 Systems Manager Automation 動作參考。

範例 2：指令碼式 Runbook

此範例 Runbook 會處理以下案例。Emily 是 AnyCompany Consultants, LLC 的系統工程師。她先前已建立兩個 Runbook，用於父子關係，以修補託管主要和次要資料庫的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體群組。應用程式每天 24 小時存取這些資料庫，因此其中一個資料庫執行個體必須永遠可用。

根據此需求，她建置了一個解決方案，使用 `AWS-RunPatchBaseline` Systems Manager (SSM) 文件分階段修補執行個體。透過使用此 SSM 文件，她的同事可以在修補操作完成後，檢閱相關聯的修補程式合規資訊。

先修補資料庫執行個體的主要群組，然後再修補資料庫執行個體的次要群組。此外，為了避免讓先前停用的執行個體因執行而產生額外的成本，Emily 確保了在修補發生之前，自動化將修補的執行個體恢復到其原始狀態。Emily 使用與資料庫執行個體的主要和次要群組相關聯的標籤，來識別應依照想要的順序修補哪些執行個體。

她現有的自動化解決方案可以運作，但她想要盡可能改善解決方案。為了協助維護 Runbook 內容並簡化故障診斷工作，她想要將自動化壓縮成單一 Runbook，並簡化輸入參數的數目。此外，她想避免建立多個子系自動化。

Emily 檢閱可用的自動化動作之後，決定可以使用 `aws:executeScript` 動作來執行她的自訂 Python 指令碼，以改善解決方案。她現在開始撰寫 Runbook 的內容，如下所示：

1. 首先，她提供 Runbook 結構描述和描述的值，並定義父系 Runbook 的輸入參數。

   透過使用 `AutomationAssumeRole` 參數，Emily 和她的同事可以使用現有 IAM 角色，允許 Automation 代表他們執行 Runbook 中的動作。與範例 1 不同，`AutomationAssumeRole` 參數現在是必要的，而不是選用的。由於此 Runbook 包含 `aws:executeScript` 動作，因此一律需要 AWS Identity and Access Management (IAM) 服務角色（或擔任角色）。這個要求是必需的，因為一些為動作指定的 Python 指令碼會呼叫 AWS API 操作。

   Emily 使用 `PrimaryPatchGroupTag` 和 `SecondaryPatchGroupTag` 參數來指定與要修補之資料庫執行個體主要和次要群組相關聯的標籤。為簡化必要的輸入參數，她決

定使用 StringMap 參數，而不是使用多個 String 參數 (如範例 1 Runbook 所用)。(選用) Operation、RebootOption 和 SnapshotId 參數可以用來提供值來記錄 AWS-RunPatchBaseline 的文件參數。為了防止提供無效值給這些文件參數，她會視需要定義 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. 定義頂層元素後，Emily 會繼續撰寫構成 Runbook `mainSteps` 的動作。第一個步驟會收集與 `PrimaryPatchGroupTag` 參數中指定標籤相關聯之所有執行個體的 ID 並輸出 `StringMap` 參數，其中包含執行個體 ID 和執行個體的目前狀態。此動作的輸出會用於稍後的動作。

   請注意，`script` 輸入參數不支援 JSON Runbook。JSON Runbook 必須使用 `attachment` 輸入參數提供指令碼內容。

   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 在另一個 aws:executeScript 動作中使用前一個動作的輸出，以確認所有與 PrimaryPatchGroupTag 參數中指定之標籤相關聯的執行個體處於 running 狀態。

   如果執行個體的狀態已經是 running 或 shutting-down，則指令碼會繼續迴圈剩餘的執行個體。

   如果執行個體的狀態為 stopping，則指令碼會輪詢執行個體以達到 stopped 狀態並啟動執行個體。

   如果執行個體的狀態為 stopped，則指令碼會啟動執行個體。

   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 會驗證所有與 PrimaryPatchGroupTag 參數中指定之標籤相關聯的執行個體是否已啟動或已經處於 running 狀態。然後，她使用另一個指令碼來確認所有執行個體 (包括前一個動作中啟動的執行個體) 是否皆已達到 running 狀態。

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 使用另外兩個指令碼來傳回 PrimaryPatchGroupTag 參數中指定之標籤鍵值的個別 String 值。這些動作傳回的值可讓她直接將值提供給 Targets 參數，用於 AWS-RunPatchBaseline 文件。自動化會繼續使用 aws:runCommand 動作修補具有 AWS-RunPatchBaseline 文件的執行個體。

   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. 修補操作完成之後，Emily 想要自動化將與 PrimaryPatchGroupTag 參數中指定之標籤相關聯的目標執行個體恢復到自動化開始之前的相同狀態。她透過再次使用指令碼中第一個動作的輸出，完成此操作。根據目標執行個體的原始狀態，如果執行個體先前處於除了 running 之外的任何狀態，則執行個體會停止。否則，如果執行個體的狀態為 running，則指令碼會繼續迴圈剩餘的執行個體。

   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. 與 PrimaryPatchGroupTag 參數中指定之標籤相關聯的執行個體已完成修補操作。現在，Emily 會複製其 Runbook 內容中先前的所有動作，以鎖定與 SecondaryPatchGroupTag 參數中指定標籤相關聯的執行個體。

   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 會檢閱已完成的指令碼 Runbook 內容，並在與目標執行個體相同的 AWS 帳戶 和 AWS 區域中建立 Runbook。現在，她已經準備好測試她的 Runbook，以確保自動化能夠依需要操作，然後再將其實作到她的生產環境中。以下是已完成的指令碼式 Runbook 內容。

   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":"..."
         }
      }
   ]
}
```

如需此範例中所使用自動化動作的詳細資訊，請參閱 Systems Manager Automation 動作參考。

其他執行手冊範例

下列範例 Runbook 示範如何使用 AWS Systems Manager 自動化動作來自動化常見的部署、疑難排解和維護任務。

> **Note**
>
> 本節中的範例執行手冊是為了示範如何建立自訂執行手冊，以支援您的特定操作需求。這些 Runbook 並不適用於生產環境中。但是，您可加以自訂以供您自己使用。

範例

- 部署 VPC 架構和 Microsoft Active Directory 網域控制站
- 從最新的快照還原根磁碟區

- 建立 AMI 和跨區域複本

部署 VPC 架構和 Microsoft Active Directory 網域控制站

若要提高效率並將一般任務標準化，您可以選擇自動化部署。如果您定期在多個帳戶和 之間部署相同的架構，這很有用 AWS 區域。自動化架構部署也可以降低手動部署架構時可能發生的人為錯誤的可能性。 AWS Systems Manager 自動化動作可協助您達成此目標。自動化是 中的工具 AWS Systems Manager。

下列範例 AWS Systems Manager Runbook 會執行這些動作：

- 使用 Systems Manager 擷取最新的 Windows Server 2016 Amazon Machine Image(AMI)Parameter Store，以便在啟動將設定為網域控制站的 EC2 執行個體時使用。 Parameter Store 是 中的工具 AWS Systems Manager。
- 使用`aws:executeAwsApi`自動化動作呼叫數個 AWS API 操作來建立 VPC 架構。網域控制站執行個體會在私有子網路中啟動，並使用 NAT 閘道連線到網際網路。如此可讓執行個體上的 SSM Agent 存取必要的 Systems Manager 端點。
- 使用`aws:waitForAwsResourceProperty`自動化動作來確認先前動作啟動的執行個體是`Online`針對的 AWS Systems Manager。
- 使用 `aws:runCommand` 自動化動作，來設定做為 Microsoft Active Directory 網域控制站啟動的執行個體。

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

```
    {
      "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 $_",
                "}"
              ]
            }
          }
        }
      ]
    }
```

從最新的快照還原根磁碟區

根磁碟區上的作業系統可能會因各種原因而損毀。例如，在修補操作之後，執行個體可能會因為核心損毀或登錄而無法成功啟動。自動化常見的故障診斷任務，例如從修補操作之前拍攝的最新快照還原根磁碟區，可以減少停機時間並加快故障診斷工作。 AWS Systems Manager 自動化動作可協助您完成此作業。自動化是 中的工具 AWS Systems Manager。

下列範例 AWS Systems Manager Runbook 會執行這些動作：

- 使用 `aws:executeAwsApi` 自動化動作從執行個體的根磁碟區擷取詳細資訊。
- 使用 `aws:executeScript` 自動化動作來擷取根磁碟區的最新快照。
- 如果找到了根磁碟區的快照，請使用 `aws:branch` 自動化動作來繼續自動化。

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"
                }
            }
        }
    }
```

建立 AMI 和跨區域複本

建立執行個體的 Amazon Machine Image (AMI)，是備份與復原常用的程序。做為災難復原架構的一部分，您也可以選擇將 AMI 複製到其他 AWS 區域 。如果問題需要容錯移轉才能解決，將一般維護任務自動化可以降低停機時間。 AWS Systems Manager 自動化動作可協助您達成此目標。自動化是 中的工具 AWS Systems Manager。

下列範例 AWS Systems Manager Runbook 會執行這些動作：

- 使用 aws:executeAwsApi 自動化動作來建立 AMI。
- 使用 aws:waitForAwsResourceProperty 自動化動作來確認 AMI 的可用性。
- 使用 aws:executeScript 自動化動作將 AMI 複製到目的地區域。

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"
             }
         }
     }
}
```

建立填入 AWS 資源的輸入參數

Automation 是 Systems Manager 中的工具，會在 AWS 中填入資源 AWS Management Console ，以符合您為輸入參數定義的資源類型。符合資源類型之 AWS 帳戶 中的資源會顯示在下拉式清單中供您選擇。您可以定義 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、Amazon Simple Storage Service (Amazon S3) 儲存貯體和 AWS Identity and Access Management (IAM) 角色的輸入參數類型。支援的類型定義和用來尋找相符資源的規則運算式如下：

- `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/.*$

以下為 Runbook 內容中定義之輸入參數類型的範例。

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
     }
  ]
```

```
}
```

使用文件建置器建立執行手冊

如果 AWS Systems Manager 公有 Runbook 不支援您想要在 AWS 資源上執行的所有動作，您可以建立自己的 Runbook。若要建立自訂 Runbook，您可以利用適當的自動化動作來手動建立本機 YAML 或 JSON 格式檔案。或者，您可以使用 Systems Manager Automation 主控台中的文件建置器來建置自訂執行手冊。

使用文件建置器，您可以將自動化動作新增至自訂執行手冊，並提供必要的參數，而不需使用 JSON 或 YAML 語法。新增步驟並建立 Runbook 之後，系統會將您新增的動作轉換成 YAML 格式，以便 Systems Manager 可以用來執行自動化。

Runbook 支援使用 Markdown (一種標示語言)，可讓您新增維基樣式的描述至 Runbook 內，以及在 Runbook 內新增個別步驟。如需使用 Markdown 的相關資訊，請參閱在 AWS中使用 Markdown。

使用文件建置器建立自訂執行手冊

開始之前

建議您了解可在執行手冊中使用的不同動作。如需詳細資訊，請參閱Systems Manager Automation 動作參考。

使用文件建置器建立自訂 Runbook

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Documents (文件)。
3. 選擇 Create automation (建立自動化)。
4. 對於 Name (名稱)，輸入 Runbook 的描述性名稱。
5. 對於 Document description (文件描述)，提供 Runbook 的 Markdown 樣式描述。您可以提供使用 Runbook、編號步驟或任何其他類型的資訊的指示來描述 Runbook。如需格式化內容的相關資訊，請參閱預設文字。

> **Tip**
>
> 在 Hide preview (隱藏預覽) 和 Show preview (顯示預覽) 之間切換，即可在撰寫時查看描述內容的外觀。

6. (選用) 對於 Assume role (擔任角色)，輸入要代表您執行動作的服務角色的名稱或 ARN。如果您未指定角色，自動化會使用執行自動化之使用者的存取許可。

   > **Important**
   >
   > 對於使用 `aws:executeScript` 動作的非 Amazon 擁有的 Runbook，必須指定角色。如需相關資訊，請參閱 使用 Runbook 的許可。

7. (選用) 對於 Outputs (輸出)，輸入用於此 Runbook 自動化以提供其他處理程序使用的任何輸出。

   例如，如果您的 Runbook 建立了新的 AMI，您可以指定 ["CreateImage.ImageId"]，然後使用此輸出在後續的自動化中建立新的執行個體。

8. (選用) 展開 Input parameters (輸入參數) 區段，並執行下列動作。

   1. 對於 Parameter name (參數名稱)，輸入您要建立的 Runbook 參數的描述性名稱。
   2. 對於 Type (類型)，選擇參數的類型，例如 `String` 或 `MapList`。
   3. 對於 Required (必要)，執行下列其中一項作業：
      - 如果必須在執行時間提供此 Runbook 參數的值，請選擇 Yes (是)。
      - 如果不需要參數，請選擇 No (否)，並 (選擇性地) 在 Default value (預設值) 中輸入預設參數值。
   4. 對於 Description (描述)，輸入 Runbook 參數的描述。

   > **Note**
   >
   > 若要新增更多 Runbook 參數，請選擇 Add a parameter (新增參數)。若要移除 Runbook 參數，請選擇 X (移除) 按鈕。

9. (選用) 展開 Target type (目標類型) 區段，並選擇目標類型，以定義自動化可執行所在的資源類型。例如，若要在 EC2 執行個體上使用 Runbook，請選擇 `/AWS::EC2::Instance`。

   > **Note**
   >
   > 如果您指定 '/' 的值，則 Runbook 可以在所有類型的資源上執行。如需有效資源類型的清單，請參閱《AWS CloudFormation 使用者指南》 中的 AWS 資源類型參考。

10. (選用) 展開 Document tags (文件標籤) 區段，並輸入要套用至 Runbook 的一或多個標籤鍵值組。標籤可讓您更容易識別、組織和搜尋資源。

11. 在 Step 1 (步驟 1) 區段中，提供下列資訊。

    - 對於 Step name (步驟名稱)，輸入自動化第一個步驟的描述性名稱。
    - 對於 Action type (動作類型)，選取要用於此步驟的動作類型。

      如需可用動作類型的清單和資訊，請參閱Systems Manager Automation 動作參考。
    - 對於 Description (描述)，輸入自動化步驟的描述。您可以使用 Markdown 來將文字格式化。
    - 根據選取的 Action type (動作類型)，在 Step inputs (步驟輸入) 區段中輸入動作類型的必要輸入。例如，如果您選取動作 `aws:approve`，則必須指定 `Approvers` 屬性的值。

      如需步驟輸入欄位的相關資訊，請參閱 Systems Manager Automation 動作參考 中您所選動作類型的項目。例如：`aws:executeStateMachine` – 執行 AWS Step Functions 狀態機器。
    - (選用) 對於 Additional inputs (其他輸入)，提供 Runbook 所需的任何其他輸入值。可用的輸入類型取決於您為步驟選取的動作類型。(請注意，某些動作類型需要輸入值。)

      > **Note**
      >
      > 若要新增更多輸入，請選擇 Add optional input (新增選用輸入)。若要移除輸入，請選擇 X (移除) 按鈕。

    - (選用) 對於 Outputs (輸出)，輸入用於此步驟以提供其他處理程序使用的任何輸出。

      > **Note**
      >
      > Outputs (輸出) 不適用所有動作類型。

    - (選用) 展開 Common properties (一般屬性) 區段，並指定所有 Automation 動作通用的動作屬性。例如，對於 Timeout seconds (逾時秒)，您可以以秒為單位提供值，以指定步驟在停止之前可以執行的時間長度。

      如需詳細資訊，請參閱依所有動作共用的屬性。

    > **Note**
    >
    > 若要新增更多步驟，請選取 Add step (新增步驟)，然後重複建立步驟的程序。若要移除步驟，請選擇 Remove step (移除步驟)。

12. 選擇 Create automation (建立自動化) 以儲存 Runbook。

建立執行指令碼的執行手冊

下列程序顯示如何在 AWS Systems Manager Automation 主控台中使用文件建置器，以建立可執行指令碼的自訂執行手冊。

您建立 Runbook 的第一個步驟會執行指令碼來啟動 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。第二個步驟會執行另一個指令碼來監控要變更為 ok 的執行個體狀態檢查。然後，會報告自動化的 `Success` 整體狀態。

開始之前

請確認您已完成下列步驟：

- 確認您具有管理員許可，或已獲授與適當的許可，才能存取 AWS Identity and Access Management (IAM) 中的 Systems Manager。

  如需相關資訊，請參閱 驗證 Runbook 的使用者存取權。
- 確認您的 AWS 帳戶帳戶中具有用於自動化的 IAM 服務角色 (也稱為擔任角色)。此角色是必要的，因為此演練使用 `aws:executeScript` 動作。

  如需建立此角色的詳細資訊，請參閱設定自動化的服務角色 (擔任角色) 存取權。

  如需執行 `aws:executeScript` 之 IAM 服務角色需求的相關資訊，請參閱 使用 Runbook 的許可。
- 確認您有啟動 EC2 執行個體的許可。

  如需相關資訊，請參閱《Amazon EC2 使用者指南》中的 IAM 和 Amazon EC2。

使用文件建置器建立執行指令碼的自訂執行手冊

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Documents (文件)。
3. 選擇 Create automation (建立自動化)。
4. 對於 Name (名稱)，輸入 Runbook 的描述性名稱：**`LaunchInstanceAndCheckStatus`**。
5. (選用) 對於 Document description (文件描述)，使用 Markdown，以此 Runbook 的描述取代預設文字。以下是範例。

```
##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. 對於 Assume role (擔任角色)，輸入對於自動化執行，用於自動化 (擔任角色) 的 IAM 服務角色的 ARN，格式為 **arn:aws:iam::111122223333:role/AutomationServiceRole**。將您的 AWS 帳戶 ID 替換為 111122223333。

   您指定的角色是用來提供開始自動化所需的許可。

   > **Important**
   >
   > 對於使用 aws:executeScript 動作的非 Amazon 擁有的 Runbook，必須指定角色。如需相關資訊，請參閱 使用 Runbook 的許可。

7. 展開 Input parameters (輸入參數)，然後執行下列動作。

   1. 對於 Parameter name (參數名稱)，輸入 **imageId**。
   2. 針對 Type (類型)，選擇 **String**。
   3. 對於 Required (必要)，選擇 No。
   4. 對於 Default value (預設值)，輸入以下內容。

      ```
      {{ ssm:/aws/service/ami-amazon-linux-latest/amzn-ami-hvm-x86_64-gp2 }}
      ```

      > **Note**
      >
      > 此值會使用最新的 Amazon Linux 1 Amazon Machine Image (AMI) ID 啟動 Amazon EC2 執行個體。如果您想使用不同的 AMI，請以您的 AMI ID 取代該值。

5. 對於 Description (描述)，輸入以下內容。

```
(Optional) The AMI ID to use for launching the instance. The default value uses
 the latest released Amazon Linux AMI ID.
```

8. 選擇 Add a parameter (新增參數) 來建立第二個參數 **tagValue**，然後輸入下列資訊。

1. 對於 Parameter name (參數名稱)，輸入 **tagValue**。
2. 針對 Type (類型)，選擇 **String**。
3. 對於 Required (必要)，選擇 No。
4. 對於 Default value (預設值)，輸入 **LaunchedBySsmAutomation**。這會將標籤金鑰對值 `Name:LaunchedBySsmAutomation` 新增至該執行個體。
5. 對於 Description (描述)，輸入以下內容。

```
(Optional) The tag value to add to the instance. The default value is
 LaunchedBySsmAutomation.
```

9. 選擇 Add a parameter (新增參數) 來建立第三個參數 **instanceType**，然後輸入下列資訊。

1. 對於 Parameter name (參數名稱)，輸入 **instanceType**。
2. 針對 Type (類型)，選擇 **String**。
3. 對於 Required (必要)，選擇 No。
4. 對於 Default value (預設值)，輸入 **t2.micro**。
5. 對於 Parameter Description (參數描述)，輸入以下內容。

```
(Optional) The instance type to use for the instance. The default value is
 t2.micro.
```

10. 展開 Target type (目標類型)，並選擇 **"/"**。
11. (選用) 展開 Document tags (文件標籤)，將資源標籤套用至您的 Runbook。對於 Tag key (標籤鍵)，輸入 **Purpose**，以及對於 Tag value (標籤值)，輸入 **LaunchInstanceAndCheckState**。
12. 在 Step 1 (步驟 1) 區段中，完成下列步驟。

1. 對於 Step name (步驟名稱)，輸入自動化第一個步驟的此描述性步驟名稱：**LaunchEc2Instance**。
2. 對於 Action type (動作類型)，選擇 Run a script (執行指令碼) (**aws:executeScript**)。
3. 對於 Description (描述)，輸入自動化步驟的描述，如下所示。

```
**About This Step**

   This step first launches an EC2 instance using the ```aws:executeScript```
 action and the provided script.
```

4. 展開 Inputs (輸入)。
5. 對於 Runtime (執行時間)，選擇用於執行所提供指令碼的執行時間語言。
6. 對於 Handler (處理常式)，輸入 **launch_instance**。這是在以下指令碼中宣告的函數名稱。

> **Note**
>
> PowerShell 不需要用到。

7. 對於 Script (指令碼)，請以下列項目取代預設內容。請務必將指令碼與對應的執行時間值相符。

   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. 展開 Additional inputs (其他輸入)。
9. 對於 Input name (輸入名稱)，選擇 InputPayload。對於 Input value (輸入值)，輸入以下 YAML 資料。

```
image_id: "{{ imageId }}"
    tag_value: "{{ tagValue }}"
    instance_type: "{{ instanceType }}"
```

13. 展開 Outputs (輸出)，並執行下列動作：
    - 對於名稱，輸入 **payload**。
    - 對於 Selector (選取器)，輸入 **$.Payload**。
    - 針對 Type (類型)，選擇 StringMap。
14. 選擇 Add step (新增步驟)，將第二個步驟新增至 Runbook。第二個步驟會查詢在步驟 1 中啟動的執行個體狀態，並等候傳回的狀態為 ok 為止。

15. 在 Step 2 (步驟 2) 區段中，執行下列動作。

    1. 對於 Step name (步驟名稱)，輸入自動化第二個步驟的此描述性名稱：**`WaitForInstanceStatusOk`**。
    2. 對於 Action type (動作類型)，選擇 Run a script (執行指令碼) (**`aws:executeScript`**)。
    3. 對於 Description (描述)，輸入自動化步驟的描述，如下所示。

    ```
    **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. 對於 Runtime (執行時間)，選擇用於執行所提供指令碼的執行時間語言。
    5. 對於 Handler (處理常式)，輸入 **`poll_instance`**。這是在以下指令碼中宣告的函數名稱。

    > **Note**
    >
    > PowerShell 不需要用到。

    6. 對於 Script (指令碼)，請以下列項目取代預設內容。請務必將指令碼與對應的執行時間值相符。

    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. 展開 Additional inputs (其他輸入)。
8. 對於 Input name (輸入名稱)，選擇 InputPayload。對於 Input value (輸入值)，輸入以下內容：

```
{{ LaunchEc2Instance.payload }}
```

16. 選擇 Create automation (建立自動化) 以儲存 Runbook。

在執行手冊中使用指令碼

自動化 Runbook 支援在自動化時執行指令碼。自動化是 中的工具 AWS Systems Manager。透過使用 Runbook，您可以直接在 AWS 中執行指令碼，而無需建立個別的運算環境來執行指令碼。因為 Runbooks 可以與其他自動化步驟類型 (例如核准) 一起執行指令碼步驟，所以您可在嚴重或不明確的情況下手動介入。您可以從 Runbook 中的 `aws:executeScript` 動作傳送輸出到 Amazon CloudWatch Logs。如需詳細資訊，請參閱使用 CloudWatch Logs 記錄自動化動作輸出。

使用 Runbook 的許可

若要使用 Runbook，Systems Manager 必須使用 AWS Identity and Access Management (IAM) 角色的許可。自動化用來判斷要使用角色的許可的方法取決於幾個因素，以及步驟是否使用 `aws:executeScript` 動作。

對於不使用 `aws:executeScript` 的 Runbook，Automation 會使用兩個許可來源的其中一個：

- Runbook 中指定或作為參數傳入的 IAM 服務角色或擔任角色的許可。
- 如果未指定 IAM 服務角色，則為啟動自動化執行之使用者的許可。

不過，當 Runbook 中的步驟包含 `aws:executeScript`動作時，如果為動作指定的 Python 或 PowerShell 指令碼正在呼叫任何 AWS API 操作，則一律需要 IAM 服務角色（擔任角色）。自動化會以下列順序檢查此角色：

- Runbook 中指定或作為參數傳入的 IAM 服務角色或擔任角色的許可。
- 如果找不到任何角色，自動化會嘗試執行為 `aws:executeScript` 指定的 Python 或 PowerShell 指令碼，而不使用任何許可。如果指令碼呼叫 AWS API 操作（例如 Amazon EC2 `CreateImage`操作），或嘗試對 AWS 資源（例如 EC2 執行個體）採取行動，則包含指令碼的步驟會失敗，而 Systems Manager 會傳回錯誤訊息，報告失敗。

將指令碼新增至 Runbook

您可以在 Runbook 中將指令碼內嵌做為步驟的一部分，將指令碼新增至 Runbook。您也可以從本機機器上傳指令碼或指定指令碼所在的 Amazon Simple Storage Service (Amazon S3) 儲存貯體，將指令碼連接至 Runbook。執行指令碼的步驟完成後，指令碼的輸出會以 JSON 物件的形式提供，然後您可以將該輸出做為 Runbook 中後續步驟的輸入。如需有關 `aws:executeScript` 動作及如何使用指令碼附件的詳細資訊，請參閱 `aws:executeScript` – 執行指令碼。

Runbook 的指令碼限制

Runbook 會強制執行五個檔案附件的限制。指令碼可以使用 Python 指令碼 (.py) 的形式、PowerShell Core 指令碼 (.ps1)，或附加為 .zip 檔案中的內容。

在執行手冊中使用條件陳述式

根據預設，您在 Runbook 之 `mainSteps` 區段中定義的步驟會循序執行。一個動作完成後，`mainSteps` 區段中指定的下一個動作就會開始。此外，如果動作無法執行，整個自動化就會失敗 (根據預設)。您可以使用本節說明的 `aws:branch` 自動化動作和 Runbook 選項建立執行條件式分支的自動化。這表示您可以建立自動化以在評估不同選擇後跳至不同的步驟，或是在步驟完成時動態回應變更。以下是您可以用來建立動態自動化的選項清單：

- **`aws:branch`**：動作可讓您建立動態自動化，以評估單一步驟中的多個選擇，接著根據該評估的結果跳至 Runbook 中的不同步驟。
- **`nextStep`**：此選項會指定在成功完成步驟後要接著處理自動化中的哪個步驟。
- **`isEnd`**：此選項會在特定步驟結束時停止自動化執行。此選項的預設值為 false。
- **`isCritical`**：此選項會指定某個步驟是成功完成自動化的關鍵。如果此指定步驟失敗，則自動化會將自動化的最終狀態回報為 `Failed`。此選項的預設值為 `true`。
- **`onFailure`**：此選項指示自動化在失敗時應中止、繼續或前往不同的步驟。此選項的預設值為 abort。

以下部分說明 `aws:branch` 自動化動作。如需 `nextStep`、`isEnd`、`isCritical` 和 `onFailure` 選項的詳細資訊，請參閱 範例 `aws:branch` Runbook。

使用 **`aws:branch`** 動作

`aws:branch` 動作為自動化提供了最動態的條件式分支選項。如前所述，此動作可讓您的自動化評估單一步驟中的多個條件，接著根據該評估的結果跳至新的步驟。`aws:branch` 動作的功能類似程式設計中的 `IF-ELIF-ELSE` 陳述式。

以下為 `aws:branch` 步驟的 YAML 範例。

```
- name: ChooseOSforCommands
  action: aws:branch
  inputs:
    Choices:
    - NextStep: runPowerShellCommand
```

```
      Variable: "{{GetInstance.platform}}"
      StringEquals: Windows
    - NextStep: runShellCommand
      Variable: "{{GetInstance.platform}}"
      StringEquals: Linux
    Default:
      PostProcessing
```

指定步驟的 aws:branch 動作時，您要指定自動化必須評估的 Choices。自動化可以根據您在 Runbook Parameters 區段中指定的參數值評估 Choices。自動化也可以根據前一個步驟的輸出評估 Choices。

自動化會使用布林值表達式評估每個選擇。如果評估判斷第一個選擇為 true，則自動化會跳至為該選擇指定的步驟。如果第一個選擇的評估判斷是 false，則自動化會評估下一個選擇。如果您的步驟包括三個或更多 Choices，則工作流程會循序評估每個選擇，直到評估某個選擇是 true 為止。接著自動化會跳至選擇為 true 的指定步驟。

如果 Choices 均不為 true，則自動化會檢查步驟是否包含 Default 值。如果沒有選擇為 true，則 Default 值會定義自動化應跳至的步驟。如果未針對步驟指定 Default 值，則自動化會處理 Runbook 中的下一個步驟。

以下是 YAML 中的 aws:branch 步驟，名稱為 chooseOSfromParameter。步驟包含兩個 Choices：(NextStep: runWindowsCommand) 和 (NextStep: runLinuxCommand)。自動化會評估這些 Choices，決定要針對適當的作業系統執行什麼命令。每個選擇的 Variable 會使用 {{OSName}}，此為 Runbook 撰寫者在 Runbook Parameters 區段中定義的參數。

```
mainSteps:
- name: chooseOSfromParameter
  action: aws:branch
  inputs:
    Choices:
    - NextStep: runWindowsCommand
      Variable: "{{OSName}}"
      StringEquals: Windows
    - NextStep: runLinuxCommand
      Variable: "{{OSName}}"
      StringEquals: Linux
```

以下是 YAML 中的 aws:branch 步驟，名稱為 chooseOSfromOutput。步驟包含兩個 Choices：(NextStep: runPowerShellCommand) 和 (NextStep: runShellCommand)。自動化會

評估這些 Choices，決定要針對適當的作業系統執行什麼命令。每個選擇的 Variable 會使用 {{GetInstance.platform}}，也就是 Runbook 中先前步驟的輸出。此範例也包含一個名稱為 Default 的選項。如果工作流程評估了兩個 Choices，而兩個選擇都不是 true，則自動化工作流程會跳至名稱為 PostProcessing 的步驟。

```
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 Runbook 中建立 **aws:branch** 步驟

當您在 Runbook 中建立 aws:branch 步驟，您要定義應由自動化評估的 Choices，以決定自動化接下來應跳至哪個步驟。如前所述，Choices 是使用布林值運算式評估。每個選擇都必須定義以下選項：

- NextStep：如果指定選擇為 true，在 Runbook 中要處理的下一個步驟。
- 變數：指定執行手冊 Parameters 區段中定義的參數名稱、Variables 區段中定義的變數，或指定上一步的輸出物件。

  使用以下格式指定變數值。

  `Variable: "{{variable name}}"`

  使用以下格式指定參數值。

  `Variable: "{{parameter name}}"`

  使用以下格式指定輸出物件變數。

  `Variable: "{{previousStepName.outputName}}"`

> **Note**
>
> 有關建立輸出變數的詳細資訊，請參閱下一節 關於建立輸出變數。

- Operation：用於評估選擇的條件，例如 `StringEquals: Linux`。`aws:branch` 動作支援以下運算：

  字串運算

  - StringEquals
  - EqualsIgnoreCase
  - StartsWith
  - EndsWith
  - 包含

  數值運算

  - NumericEquals
  - NumericGreater
  - NumericLesser
  - NumericGreaterOrEquals
  - NumericLesser
  - NumericLesserOrEquals

  布林運算

  - BooleanEquals

  > **Important**
  >
  > 當您建立 Runbook 時，系統會驗證 Runbook 中的每個操作。如果不支援操作，系統會在您嘗試建立 Runbook 時傳回錯誤。

- Default：指定 `Choices` 均不為 `true` 時，自動化應跳至的遞補步驟。

> **Note**
>
> 如果不想指定 Default 值，您可以指定 isEnd 選項。如果 Choices 均不為 true，而 Default 值也未指定，則自動化會在步驟結束時停止。

使用以下範本協助您在 Runbook 中建構 aws:branch 步驟：將每個########取代為您自己的資訊。

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"
        }
    }
  ]
}
```

關於建立輸出變數

若要建立參考先前步驟之輸出的 aws:branch 選擇，您必須指定先前步驟的名稱和輸出欄位的名稱。接著，使用以下格式結合步驟和欄位的名稱。

`Variable: "{{previousStepName.outputName}}"`

例如，以下範例中的第一個步驟名稱為 GetInstance。而在 outputs 下，有一個欄位的名稱為 platform。在第二個步驟 (ChooseOSforCommands) 中，撰寫者想要參考 platform 欄位的輸出做為變數。若要建立變數，只需結合步驟名稱 (GetInstance) 和輸出欄位名稱 (platform) 以建立 `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
```

以下範例，說明如何從先前的步驟及輸出建立 *"Variable":"{{ describeInstance.Platform }}"*。

```
- 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
```

範例 **aws:branch** Runbook

以下是一些使用 aws:branch 的範例 Runbook。

範例 1：使用 **aws:branch** 搭配輸出變數以根據作業系統類型執行命令

在此範例 (GetInstance) 的第一個步驟，Runbook 撰寫人使用 aws:executeAwsApi 動作來呼叫 ssm DescribeInstanceInformation API 操作。撰寫者透過此動作來判斷執行個體使用的作業系統類型。aws:executeAwsApi 動作會輸出執行個體 ID 和平台類型。

在第二個步驟中 (ChooseOSforCommands)，撰寫者使用 aws:branch 動作搭配兩個 Choices (NextStep: runPowerShellCommand) 和 (NextStep: runShellCommand)。自動化會使用先前步驟的輸出評估執行個體的作業系統 (Variable: "{{GetInstance.platform}}")。自動化跳至指定作業系統的步驟。

```
---
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
```

範例 2：使用 **aws:branch** 搭配參數變數以根據作業系統類型執行命令

Runbook 撰寫者在 Runbook 開頭的 parameters 區段定義了數個參數選項。其中一個參數名稱為 OperatingSystemName。在第一個步驟中 (ChooseOS)，撰寫者使用 aws:branch 動作搭配兩個 Choices (NextStep: runWindowsCommand) 和 (NextStep: runLinuxCommand)。這些 Choices 的變數會參考在參數區段中指定的參數選項 (Variable: "{{OperatingSystemName}}")。使用者執行此 Runbook 時，針對 OperatingSystemName 指定了在執行時間的值。自動化在 Choices 評估期間使用執行時間參數。自動化會根據針對 OperatingSystemName 指定的執行時間參數跳至指定作業系統的步驟。

```
---
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
```

使用運算子建立複雜的分支自動化

您可以使用 aws:branch 步驟中的 And、Or 和 Not 運算子建立複雜的分支自動化。

「And」運算子

當您希望一個選擇有多個為 true 的變數，請使用 And 運算子。在以下範例中，第一個選擇評估了執行個體是否 running 並使用 Windows 作業系統。如果兩個變數均為 true，則自動化會跳至 runPowerShellCommand 步驟。如果一個或多個變數為 false，則自動化會評估第二個選擇的變數。

```
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」運算子

當您希望一個選擇有多個為 true 的變數，請使用 Or 運算子。在以下範例中，第一個選擇評估了參數字串是否為 Windows 且 AWS Lambda 步驟的輸出是否為 true。如果評估判斷這些變數其中之一為 true，則自動化會跳至 RunPowerShellCommand 步驟。如果兩個變數均為 false，則自動化會評估第二個選擇的變數。

```
- Or:
  - Variable: "{{parameter1}}"
    StringEquals: Windows
```

```
  - Variable: "{{BooleanParam1}}"
    BooleanEquals: true
  NextStep: RunPowershellCommand
- Or:
  - Variable: "{{parameter2}}"
    StringEquals: Linux
  - Variable: "{{BooleanParam2}}"
    BooleanEquals: true
  NextStep: RunShellScript
```

「Not」運算子

當您想要跳至變數為非 true 時定義的步驟時，使用 Not 運算子。在以下範例中，第一個選擇評估了參數字串是否為 Not Linux。如果評估判斷變數不是 Linux，則自動化會跳至 sleep2 步驟。如果第一個選擇的評估判斷是 Linux，則自動化會評估下一個選擇。

```
mainSteps:
- name: switch
  action: aws:branch
  inputs:
    Choices:
    - NextStep: sleep2
      Not:
        Variable: "{{testParam}}"
        StringEquals: Linux
    - NextStep: sleep1
      Variable: "{{testParam}}"
      StringEquals: Windows
    Default:
      sleep3
```

如何使用條件選項的範例

本節包括不同的範例，示範如何在 Runbook 中使用動態選項。本節中每個範例都會延伸以下的 Runbook。此 Runbook 有兩個動作。第一種動作名稱為 InstallMsiPackage。它會使用 aws:runCommand 動作在 Windows Server 執行個體上安裝應用程式。第二個動作名稱為 TestInstall。這會使用 aws:invokeLambdaFunction 動作對安裝的應用程式執行測試，確認應用程式是否安裝成功。步驟一指定 onFailure: Abort。這表示如果應用程式未安裝成功，自動化就在步驟二前停止。

範例 1：具有兩個線性動作的 Runbook

```
---
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
...
```

建立動態自動化以使用 **onFailure** 選項跳至不同步驟

以下範例使用 onFailure: step:*step name*、nextStep、isEnd 選項來建立動態自動化。在此範例中，如果InstallMsiPackage動作失敗，自動化會跳到名為 PostFailure (onFailure: step:PostFailure) 的動作，以執行 AWS Lambda 函數，在安裝失敗時執行一些動作。如果安裝成功，則自動化會跳至 TestInstall 動作 (nextStep: TestInstall)。TestInstall 和 PostFailure 步驟均使用 isEnd 選項 (isEnd: true)，因此自動化會在其中一個步驟完成時結束執行。

> **Note**
>
> 在 mainSteps 區段最後一個步驟使用 isEnd 選項是選用的。如果最後一個步驟未跳至其他步驟，則自動化會在最後一個步驟中執行動作之後停止。

範例 2：跳至不同步驟的動態自動化

```
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**
>
> 在處理 Runbook 之前，系統會驗證 Runbook 是否不會建立無限迴圈。如果偵測到無限迴圈，自動化會傳回錯誤和圓形追蹤，顯示建立迴圈的步驟。

建立定義關鍵步驟的動態自動化

您可以將一個步驟指定為自動化整體成功的關鍵。如果關鍵步驟失敗，即使有一個或多個步驟已執行成功，Automation 仍會將自動化的狀態回報為 Failed。在下列範例中，使用者會在 InstallMsiPackage 步驟失敗 (onFailure: step:VerifyDependencies) 時識別 VerifyDependencies 步驟。使用者指定 InstallMsiPackage 步驟不是關鍵 (isCritical: false)。在此範例中，如果應用程式無法安裝，自動化會處理 VerifyDependencies 步驟以判斷是否有一個或多個相依性遺失，因此導致應用程式安裝失敗。

範例 3：定義自動化的關鍵步驟

```
---
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
...
```

使用動作輸出作為輸入

數個自動化動作會傳回預先定義的輸出。您可以將這些輸出作為輸入傳遞給 {{*stepName*.*outputName*}} 格式執行手冊中的後續步驟。您還可以在執行手冊中定義自動化動作的自訂輸出。這可讓您執行指令碼，或叫用其他 的 API 操作 AWS 服務 一次，以便在稍後的動作中重複使用這些值做為輸入。執行手冊中的參數類型是靜態的。這意味著參數類型在定義後便無法變更。若要定義步驟輸出，請提供下列欄位：

- 名稱：(必填) 用於在後面的步驟中引用輸出值的輸出名稱。
- 選取器：(必填) 用於決定輸出值的 JSONPath 運算式。
- 類型：(選用) 選取器欄位傳回的值的資料類型。有效類型值為 String、Integer、Boolean、StringList、StringMap、MapList。預設值為 String。

如果輸出的值與您指定的資料類型不符，Automation 會嘗試轉換資料類型。例如，若返回的值是 Integer，但指定的 Type 是 String，則最終輸出值是 String 值。支援下列類型的轉換：

- String 值可轉換為 StringList、Integer 和 Boolean。
- Integer 值可轉換為 String 和 StringList。
- Boolean 值可轉換為 String 和 StringList。
- StringList、IntegerList 或 BooleanList 值包含可以轉換為 String、Integer 或 Boolean 的一個元素。

將參數或輸出與自動化動作搭配使用時，無法在動作的輸入中動態變更資料類型。

下面是一個執行手冊範例，示範如何定義動作輸出，並參照該值作為稍後動作的輸入。執行手冊會執行下列操作：

- 使用 aws:executeAwsApi 動作呼叫 Amazon EC2 DescribeImages API 操作，以獲得特定 Windows Server 2016 AMI 的名稱。這會將映像 ID 輸出為 ImageId。
- 使用 aws:executeAwsApi 動作呼叫 Amazon EC2 RunInstances API 操作，以啟動使用先前步驟之 ImageId 的執行個體。這會將執行個體 ID 輸出為 InstanceId。
- 使用 aws:waitForAwsResourceProperty 動作輪詢 Amazon EC2 DescribeInstanceStatus API 操作，以等待執行個體達到 running 狀態。動作在 60 秒逾時。如果執行個體狀態無法在 60 秒的輪詢後達到 running，則步驟會逾時。
- 使用 aws:assertAwsResourceProperty 動作來呼叫 Amazon EC2 DescribeInstanceStatus API 操作，以宣告執行個體位於 running 狀態。如果執行個體狀態不是 running，則步驟會失敗。

```
---
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"
...
```

前述的每個自動化動作都可讓您藉由指定服務命名空間、API 操作名稱、輸入參數、輸出參數來呼叫特定 API 操作。輸入是由您選擇的 API 操作定義。您可以檢視 API 操作 (也稱為方法)，方式是在以下服務參考頁面的左側導覽中選擇一項服務。在您想要呼叫之服務的 Client (用戶端) 部分選擇一個方法。例如，Amazon Relational Database Service (Amazon RDS) 的所有 API 操作 (方法) 均列於以下頁面：Amazon RDS 方法。

您可以在以下位置檢視每個自動化動作的結構描述：

- aws:assertAwsResourceProperty – 宣告 AWS 資源狀態或事件狀態
- aws:executeAwsApi – 呼叫並執行 AWS API 操作
- aws:waitForAwsResourceProperty – 等待 AWS 資源屬性

結構描述包括使用各動作之必要欄位的描述。

使用 Selector/PropertySelector 欄位

每個 Automation 動作都需要您指定輸出 `Selector` (用於 `aws:executeAwsApi`) 或 `PropertySelector` (用於 `aws:assertAwsResourceProperty` 和 `aws:waitForAwsResourceProperty`)。這些欄位用於處理來自 AWS API 操作的 JSON 回應。這些欄位使用 JSONPath 語法。

以下範例可協助說明 `aws:executeAwsAPi` 動作的概念。

```
---
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 步驟 getImageId 中，自動化會叫用 DescribeImages API 操作，並接收來自 ec2 的回應。接著自動化將 Selector - "$.Images[0].ImageId" 套用至 API 回應並將選取的值指派給輸出 ImageId 變數。在相同自動化中的其他步驟可藉由指定 "{{ getImageId.ImageId }}" 使用 ImageId 的值。

以下範例可協助說明 aws:waitForAwsResourceProperty 動作的概念。

```
---
- 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 步驟 waitUntilInstanceStateRunning 中，自動化會叫用 DescribeInstanceStatus API 操作，並接收來自 ec2 的回應。自動化接著將 PropertySelector - "$.InstanceStatuses[0].InstanceState.Name" 套用至回應，並檢查指定傳回的值是否符合 DesiredValues 清單中的值 (在此例中為 running)。步驟會重複程序，直到回應傳回的執行個體狀態為 running。

在執行手冊中使用 JSONPath

JSONPath 運算式是以「$」開頭的字串。用於在 JSON 元素中選取一個或多個元件。以下清單包括由 Systems Manager 自動化支援的 JSONPath 運算子相關資訊：

- 以點標記的子代 (.)：與 JSON 物件搭配使用。此運算子會選取特定索引鍵的值。

- Deep-scan (..)：與 JSON 元素搭配使用。此運算子會在各層級掃描 JSON 元素並選取具有特定索引鍵之值的清單。此運算子的傳回類型一律為 JSON 陣列。在自動化動作輸出類型的內容中，運算子可以是 StringList 或 MapList。
- Array-Index ([ ])：與 JSON 陣列搭配使用。此運算子會取得特定索引的值。
- 篩選 ([?(***expression***)])：與 JSON 數組一起使用。此運算子會篩選與篩選運算式中定義的條件相符的 JSON 陣列值。篩選運算式僅能使用下列運算子：==、!=、>、<、>= 或 <=。不支援將多個篩選運算式與 AND (&&) 或 OR (||) 結合使用。此運算子的傳回類型一律為 JSON 陣列。

為了更全面了解 JSONPath 運算子，請檢閱以下 ec2 `DescribeInstances` API 操作的 JSON 回應。在此回應下有幾個範例，顯示套用不同的 JSONPath 運算式到 `DescribeInstances` API 操作之回應的不同結果。

```
{
    "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 範例 1：從 JSON 回應取得特定字串

```
JSONPath:
$.Reservations[0].Instances[0].ImageId

Returns:
"ami-12345678"

Type: String
```

JSONPath 範例 2：從 JSON 回應取得特定布林值

```
JSONPath:
$.Reservations[0].Instances[0].BlockDeviceMappings[0].Ebs.DeleteOnTermination

Returns:
true
```

```
Type: Boolean
```

JSONPath 範例 3：從 JSON 回應取得特定整數

```
JSONPath:
$.Reservations[0].Instances[0].State.Code

Returns:
16

Type: Integer
```

JSONPath Example 4：深度掃描 JSON 回應，接著取得做為 StringList 的 VolumeId 所有值

```
JSONPath:
$.Reservations..BlockDeviceMappings..VolumeId

Returns:
[
   "vol-000000000000",
   "vol-111111111111"
]

Type: StringList
```

JSONPath 範例 5：取得做為 StringMap 的特定 BlockDeviceMappings 物件

```
JSONPath:
$.Reservations[0].Instances[0].BlockDeviceMappings[0]

Returns:
{
   "Ebs" : {
      "DeleteOnTermination" : true,
      "Status" : "attached",
      "VolumeId" : "vol-000000000000"
   },
   "DeviceName" : "/dev/xvda"
}

Type: StringMap
```

JSONPath Example 6：深度掃描 JSON 回應，接著取得做為 MapList 的所有 State 物件

```
JSONPath:
$.Reservations..Instances..State

Returns:
[
   {
      "Code" : 16,
      "Name" : "running"
   },
   {
      "Code" : 80,
      "Name" : "stopped"
   }
]

Type: MapList
```

JSONPath 範例 7：篩選 **`running`** 狀態中的執行個體

```
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：返回不處於 **`running`** 狀態之執行個體的 **`ImageId`**

```
JSONPath:
$.Reservations..Instances[?(@.State.Name != 'running')].ImageId

Returns:
[
  "ami-12345678"
]

Type: StringList | String
```

為 Automation 建立 Webhook 整合

若要在自動化過程中使用 Webhook 傳送訊息，請建立整合。在自動化過程中，您可以使用 Runbook 中的 `aws:invokeWebhook` 動作來叫用整合。若尚未建立 Webhook，請參閱 為整合建立 Webhook。若要進一步了解 `aws:invokeWebhook` 動作，請參閱 `aws:invokeWebhook`：叫用 Automation Webhook 整合。

如以下程序所示，您可以使用 Systems Manager Automation 主控台或偏好的命令列工具來建立整合。

建立整合 (主控台)

建立 Automation 整合 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Automation (自動化)。
3. 選擇 Integrations (整合) 索引標籤。
4. 選取 Add integration (新增整合)，然後選擇 Webhook。
5. 輸入整合要包含的必要值和選用值。
6. 選擇 Add (新增) 來建立整合。

建立整合 (命令列)

要想使用命令列工具來建立整合，您必須建立必要的 SecureString 參數進行整合。自動化在 Systems Manager 中的Parameter Store工具 中使用預留命名空間，以存放整合的相關資訊。如果您使用 建立整合 AWS Management Console，自動化會為您處理此程序。在命名空間之後，您必須指定要建立的整合類型，然後指定整合的名稱。Automation 目前支援 webhook 類型整合。

webhook 類型整合的支援欄位如下所示：

- 描述
- 標頭
- payload
- URL

開始之前

如果您尚未安裝，請安裝並設定 AWS Command Line Interface (AWS CLI) 或 AWS Tools for PowerShell。如需相關資訊，請參閱安裝或更新 AWS CLI的最新版本和安裝 AWS Tools for PowerShell。

建立 Automation 整合 (命令列)

- 執行下列命令來建立整合所需的 SecureString 參數。將每個########取代為您自己的資訊。/d9d01087-4a3f-49e0-b0b4-d568d7826553/ssm/integrations/webhook/ 命名空間會保留在 Parameter Store 中，以供整合使用。參數名稱必須使用此命名空間，後面接著整合名稱。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"}'
```

為整合建立 Webhook

借助提供商建立 Webhook 時，請注意下列事項：

- 通訊協定必須是 HTTPS。
- 支援自訂請求標頭。
- 可以指定預設請求主體。
- 使用 aws:invokeWebhook 動作叫用整合時，可以覆寫預設請求主體。

處理 Runbook 中的逾時

timeoutSeconds 屬性由所有自動化動作共用。您可以使用此屬性來指定動作的執行逾時值。此外，您還可以變更動作逾時影響自動化和整體執行狀態的方式。您也可以定義動作的 onFailure 和 isCritical 共用屬性來這樣做。

例如，視您的使用案例而定，您可能希望自動化繼續執行不同的動作，而且在動作逾時時不影響自動化的整體狀態。在此範例中，您可以使用 timeoutSeconds 屬性指定動作逾時之前要等待的時間長度。接著指定如果逾時，自動化應該採取的動作或步驟。使用 step:*step name* 格式指定 onFailure

屬性的值，來取代 Abort 的預設值。預設情況下，如果動作逾時，自動化的執行狀態將是 Timed Out。若要避免逾時影響自動化執行狀態，請為 isCritical 屬性指定 false。

下列範例顯示如何定義此案例中所述動作的共用屬性。

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"
}
```

如需所有自動化動作共用屬性的詳細資訊，請參閱 依所有動作共用的屬性。

Systems Manager Automation Runbook 參考

為了協助您快速入門， AWS Systems Manager 提供預先定義的 Runbook。這些 Runbook 由 Amazon Web Services 維護 AWS 支援，以及 AWS Config。Runbook 參考說明 Systems Manager 提供的每個預先定義的 Runbook 支援，以及 AWS Config。如需詳細資訊，請參閱《Systems Manager Automation Runbook 參考》。

教學課程

下列教學課程可協助您使用 AWS Systems Manager 自動化來解決常見的使用案例。這些教學課程示範如何使用您自己的 Runbook、Automation 提供的預先定義 Runbook，以及其他 Systems Manager 工具 AWS 服務。

內容

- 更新 AMIs
  - 更新 Linux AMI
  - 更新 LinuxAMI (AWS CLI)
  - 更新 Windows Server (AMI)
  - AMI 使用自動化更新黃金 AWS Lambda，以及 Parameter Store
    - 任務 1：建立 Systems Manager Parameter Store 參數
    - 任務 2：建立 AWS Lambda的 IAM 角色
    - 任務 3：建立 AWS Lambda 函數
    - 任務 4：建立 Runbook 並修補 AMI
  - 使用 Automation 和 Jenkins 更新 AMIs
  - 更新 Auto Scaling 群組的 AMIs
    - 建立 PatchAMIAndUpdateASG Runbook
- 使用 AWS 支援 自助式 Runbook
  - 在無法觸達的執行個體上執行 EC2Rescue 工具
    - 運作方式
    - 開始之前
      - 授予 AWSSupport-EC2Rescue 在執行個體上執行動作的許可
      - 使用 IAM 政策授予許可

            - 使用 AWS CloudFormation 範本授予許可
        - 執行自動化
    - 在 EC2 執行個體上重設密碼和 SSH 金鑰
        - 運作方式
        - 開始之前
            - 授予 AWSSupport-EC2Rescue 在執行個體上執行動作的許可
                - 使用 IAM 政策授予許可
                - 使用 AWS CloudFormation 範本授予許可
        - 執行自動化
- 使用輸入轉換器將資料傳遞至 Automation

更新 AMIs

下列教學課程說明如何更新 Amazon Machine Image (AMIs) 以包含最新的修補程式。

主題

- 更新 Linux AMI
- 更新 LinuxAMI (AWS CLI)
- 更新 Windows Server (AMI)
- AMI 使用自動化更新黃金 AWS Lambda，以及 Parameter Store
- 使用 Automation 和 Jenkins 更新 AMIs
- 更新 Auto Scaling 群組的 AMIs

更新 Linux AMI

此 Systems Manager Automation 演練會示範如何使用主控台或 AWS CLI 以及 `AWS-UpdateLinuxAmi` 執行手冊，透過您指定的套件的最新修補程式來更新 Linux AMI。自動化是 中的工具 AWS Systems Manager。`AWS-UpdateLinuxAmi` Runbook 也會自動化安裝其他的網站特定套件和組態。您可以使用此演練更新各種 Linux 發行版本，包括 Ubuntu Server、CentOS、RHEL、SLES 或 Amazon Linux AMIs。如需支援的 Linux 版本完整清單，請參閱 Patch Manager 先決條件。

`AWS-UpdateLinuxAmi` 執行手冊可讓您自動化映像維護任務，而無需撰寫 JSON 或 YAML 格式的執行手冊。您可以使用 `AWS-UpdateLinuxAmi` Runbook 執行以下類型的任務。

- 在 Amazon Linux、Red Hat Enterprise Linux、Ubuntu Server、SUSE Linux Enterprise Server 或 CentOS Amazon Machine Image (AMI) 上，升級所有發行版本套件和 Amazon 軟體。這是 Runbook 預設行為。
- 在 AWS Systems Manager SSM Agent現有映像上安裝 以啟用 Systems Manager 工具，例如使用 AWS Systems Manager Run Command 執行遠端命令或使用庫存執行軟體庫存收集。
- 安裝其他軟體套件。

開始之前

在您開始使用 Runbook 之前，請設定 Automation 的角色和 EventBridge (選用)。如需詳細資訊，請參閱設定自動化。此演練也需要您指定 AWS Identity and Access Management (IAM) 執行個體設定檔的名稱。如需有關建立 IAM 執行個體設定檔的詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可。

`AWS-UpdateLinuxAmi` Runbook 接受以下的輸入參數。

參數	類型	描述
SourceAmiId	字串	(必要) 來源 AMI ID。
IamInstanceProfileName	字串	(必要) 您在設定 Systems Manager 所需的執行個體許可中所建立之 IAM 執行個體設定檔角色的名稱。執行個體設定檔角色可讓自動化許可在您的執行個體上執行動作，例如執行命令或啟動和停用服務。Runbook 僅使用執行個體設定檔角色的名稱。如果您指定 Amazon Resource Name (ARN)，自動化會失敗。
AutomationAssumeRole	字串	(必要) 您在 設定自動化 中建立之 IAM 服務角色的名稱。服務角色 (也稱為擔任角色) 會提供自動化許可來擔任您的 IAM 角色並代表您執行動

參數	類型	描述
		作。例如，在 Runbook 中執行 `aws:createImage` 動作時，服務角色會允許自動化建立新的 AMI。針對此參數，必須指定的完整 ARN。
TargetAmiName	字串	(選用) 建立後的新 AMI 之名稱。預設名稱為系統產生的字串，包括來源 AMI ID，以及建立時間和日期。
InstanceType	字串	(選用) 做為工作空間主機啟動的執行個體類型。執行個體類型因區域而異。預設類型為 t2.micro。
PreUpdateScript	字串	(選用) 套用更新前要執行的指令碼之 URL。預設 (\"none\") 為不執行指令碼。
PostUpdateScript	字串	(選用) 套用套件更新後要執行的指令碼之 URL。預設 (\"none\") 為不執行指令碼。
IncludePackages	字串	(選用) 僅更新這些具名的套件。根據預設 (\"all\")，會套用所有可用的更新。
ExcludePackages	字串	(選用) 在各種條件下，要保留不更新的套件之名稱。根據預設 (\"none\")，無排除套件。

自動化步驟

依預設，`AWS-UpdateLinuxAmi` Runbook 包含下列自動化動作。

步驟 1：launchInstance (**aws:runInstances** 動作)

此步驟使用 Amazon Elastic Compute Cloud (Amazon EC2) 使用者資料和 IAM 執行個體設定檔角色啟動執行個體。Userdata 會根據作業系統安裝合適的 SSM Agent 代理程式。安裝 SSM Agent可讓您利用 Systems Manager 工具State Manager，例如 Run Command、 和 庫存。

步驟 2：updateOSSoftware (**aws:runCommand** 動作)

此步驟會在啟動的執行個體上執行以下命令：

- 從 Amazon Simple Storage Service (Amazon S3) 下載更新指令碼。
- 執行選用的更新前指令碼。
- 更新軟體發佈套件和 Amazon 軟體。
- 執行選用的更新後指令碼。

執行日誌存放於 /tmp 資料夾以供使用者日後檢視。

如果您想要升級一組特定的套件，您可以使用 `IncludePackages` 參數提供清單。提供後，系統會嘗試僅更新這些套件及其相依性。其他更新不會執行。根據預設，未指定包含套件時，程式會更新所有可用的套件。

如果您想要排除升級一組特定的套件，您可以使用 `ExcludePackages` 參數提供清單。若提供，這些套件會維持在目前的版本，獨立於指定的其他任何選項。根據預設，未指定排除套件時，就不會排除任何套件。

步驟 3：stopInstance 停止 (**aws:changeInstanceState** 動作)

此步驟會停止更新的執行個體。

步驟 4：createImage (**aws:createImage** 動作)

此步驟會以連結至來源 ID 和建立時間的描述性名稱建立新的 AMI。例如：「AMI Generated by EC2 Automation on {{global:DATE_TIME}} from {{SourceAmiId}}」，其中 DATE_TIME 和 SourceID 代表自動化變數。

步驟 5：terminateInstance (**aws:changeInstanceState** 動作)

此步驟會藉由終止執行中的執行個體來清除自動化。

輸出

自動化會傳回新的 AMI ID 作為輸出。

> **Note**
>
> 根據預設，Automation 執行 `AWS-UpdateLinuxAmi` Runbook 時，系統在預設 VPC (172.30.0.0/16) 中建立暫時執行個體。如果刪除預設 VPC，您會收到以下錯誤：
> `VPC not defined 400`
> 若要解決此問題，您必須複製 `AWS-UpdateLinuxAmi` Runbook 並指定子網路 ID。如需詳細資訊，請參閱VPC 未定義 400。

使用 Automation (AWS Systems Manager) 建立已修補的 AMI

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Automation (自動化)。
3. 選擇 Execute automation (執行自動化)。
4. 在 Automation document (自動化文件) 清單中，選擇 **`AWS-UpdateLinuxAmi`**。
5. 在 Document details (文件詳細資訊) 部分，確認 Document version (文件版本) 設為 Default version at runtime (執行時間的預設版本)。
6. 選擇 Next (下一步)。
7. 在 Execution Mode (執行模式) 部分，選擇 Simple execution (簡易執行)。
8. 在 Input parameters (輸入參數) 部分，輸入您在 Before You Begin (開始之前) 部分收集的資訊。
9. 選擇 Execute (執行)。主控台會顯示自動化執行的狀態。

自動化結束後，從已更新的 AMI 啟動測試執行個體以確認變更。

> **Note**
>
> 如果自動化中有任何步驟失敗，關於失敗的資訊會列於 Automation Executions (自動化執行清單) 頁面。自動化設計為在成功完成所有任務後終止暫時執行個體。如果有步驟失敗，系統可能不會終止執行個體。所以如果有步驟失敗，請手動終止暫時執行個體。

更新 LinuxAMI (AWS CLI)

此 AWS Systems Manager 自動化演練說明如何使用 AWS Command Line Interface (AWS CLI) 和 Systems Manager `AWS-UpdateLinuxAmi` 執行手冊，自動修補 Linux Amazon Machine Image(AMI)

與您指定的最新版本套件。自動化是 中的工具 AWS Systems Manager。AWS-UpdateLinuxAmi Runbook 也會自動化安裝其他的網站特定套件和組態。您可以使用此演練更新各種 Linux 發行版本，包括 Ubuntu Server、CentOS、RHEL、SLES 或 Amazon Linux AMIs。如需支援的 Linux 版本完整清單，請參閱 Patch Manager 先決條件。

AWS-UpdateLinuxAmi Runbook 可讓您自動化映像維護任務，不必使用 JSON 或 YAML 撰寫 Runbook。您可以使用 AWS-UpdateLinuxAmi Runbook 執行以下類型的任務。

- 在 Amazon Linux、Red Hat Enterprise Linux、Ubuntu Server、SLES 或 CentOS Amazon Machine Image (AMI) 上升級所有發行版本套件和 Amazon 軟體。這是 Runbook 預設行為。
- 在 AWS Systems Manager SSM Agent現有映像上安裝 以啟用 Systems Manager 功能，例如使用 AWS Systems Manager Run Command 執行遠端命令，或使用庫存執行軟體庫存收集。
- 安裝其他軟體套件。

開始之前

在您開始使用 Runbook 之前，請設定 Automation 的角色和 EventBridge (選用)。如需詳細資訊，請參閱設定自動化。此演練也需要您指定 AWS Identity and Access Management (IAM) 執行個體設定檔的名稱。如需有關建立 IAM 執行個體設定檔的詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可。

AWS-UpdateLinuxAmi Runbook 接受以下的輸入參數。

參數	類型	描述
SourceAmiId	字串	(必要) 來源 AMI ID。您可以使用公有 AWS Systems Manager Parameter Store 參數自動參考 Amazon EC2 AMI for Linux 的最新 ID。如需詳細資訊，請參閱 Query for the latest Amazon Linux AMI IDs using AWS Systems Manager Parameter Store。
IamInstanceProfileName	字串	(必要) 您在設定 Systems Manager 所需的執行個體許可中所建立之 IAM 執行個體

參數	類型	描述
		設定檔角色的名稱。執行個體設定檔角色可讓自動化許可在您的執行個體上執行動作，例如執行命令或啟動和停用服務。Runbook 僅使用執行個體設定檔角色的名稱。
AutomationAssumeRole	字串	(必要) 您在 設定自動化 中建立之 IAM 服務角色的名稱。服務角色 (也稱為擔任角色) 會提供自動化許可來擔任您的 IAM 角色並代表您執行動作。例如，在 Runbook 中執行 `aws:createImage` 動作時，服務角色會允許自動化建立新的 AMI。針對此參數，必須指定的完整 ARN。
TargetAmiName	字串	(選用) 建立後的新 AMI 之名稱。預設名稱為系統產生的字串，包括來源 AMI ID，以及建立時間和日期。
InstanceType	字串	(選用) 做為工作空間主機啟動的執行個體類型。執行個體類型因區域而異。預設類型為 t2.micro。
PreUpdateScript	字串	(選用) 套用更新前要執行的指令碼之 URL。預設 (\"none\") 為不執行指令碼。
PostUpdateScript	字串	(選用) 套用套件更新後要執行的指令碼之 URL。預設 (\"none\") 為不執行指令碼。

參數	類型	描述
IncludePackages	字串	(選用) 僅更新這些具名的套件。根據預設 (\"all\")，會套用所有可用的更新。
ExcludePackages	字串	(選用) 在各種條件下，要保留不更新的套件之名稱。根據預設 (\"none\")，無排除套件。

自動化步驟

依預設，AWS-UpdateLinuxAmi Runbook 包含下列步驟。

步驟 1：launchInstance (**aws:runInstances** 動作)

此步驟使用 Amazon Elastic Compute Cloud (Amazon EC2) 使用者資料和 IAM 執行個體設定檔角色啟動執行個體。使用者資料會根據作業系統安裝合適的 SSM Agent。安裝 SSM Agent可讓您利用 Systems Manager 工具State Manager，例如 Run Command、 和 庫存。

步驟 2：updateOSSoftware (**aws:runCommand** 動作)

此步驟會在啟動的執行個體上執行以下命令：

- 從 Amazon Simple Storage Service (Amazon S3) 下載更新指令碼。
- 執行選用的更新前指令碼。
- 更新軟體發佈套件和 Amazon 軟體。
- 執行選用的更新後指令碼。

執行日誌存放於 /tmp 資料夾以供使用者日後檢視。

如果您想要升級一組特定的套件，您可以使用 IncludePackages 參數提供清單。提供後，系統會嘗試僅更新這些套件及其相依性。其他更新不會執行。根據預設，未指定包含套件時，程式會更新所有可用的套件。

如果您想要排除升級一組特定的套件，您可以使用 ExcludePackages 參數提供清單。若提供，這些套件會維持在目前的版本，獨立於指定的其他任何選項。根據預設，未指定排除套件時，就不會排除任何套件。

步驟 3：stopInstance 停止 (**aws:changeInstanceState** 動作)

此步驟會停止更新的執行個體。

步驟 4：createImage (**aws:createImage** 動作)

此步驟會以連結至來源 ID 和建立時間的描述性名稱建立新的 AMI。例如：「AMI Generated by EC2 Automation on {{global:DATE_TIME}} from {{SourceAmiId}}」，其中 DATE_TIME 和 SourceID 代表自動化變數。

步驟 5：terminateInstance (**aws:changeInstanceState** 動作)

此步驟會藉由終止執行中的執行個體來清除自動化。

輸出

自動化會傳回新的 AMI ID 作為輸出。

> **Note**
>
> 根據預設，Automation 執行 `AWS-UpdateLinuxAmi` Runbook 時，系統在預設 VPC (172.30.0.0/16) 中建立暫時執行個體。如果刪除預設 VPC，您會收到以下錯誤：
> `VPC not defined 400`
> 若要解決此問題，您必須複製 `AWS-UpdateLinuxAmi` Runbook 並指定子網路 ID。如需詳細資訊，請參閱VPC 未定義 400。

使用 自動化 建立已修補的 AMI

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

2. 執行下列命令，以執行 `AWS-UpdateLinuxAmi` Runbook。將每個########取代為您自己的資訊。

```
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'
```

命令會傳回執行 ID。複製此 ID 到剪貼簿。您可以使用此 ID 檢視自動化的狀態。

```
{
    "AutomationExecutionId": "automation execution ID"
}
```

3. 若要使用 檢視自動化 AWS CLI，請執行下列命令：

```
aws ssm describe-automation-executions
```

4. 執行以下命令檢視自動化進度的詳細資訊。把*##### ID* 取代為您自己的資訊。

```
aws ssm get-automation-execution --automation-execution-id automation execution ID
```

更新程序可能需要 30 分鐘或以上的時間完成。

> **Note**
>
> 您也可以在主控台中監控自動化的狀態。在清單中，選擇您剛執行的自動化，接著選擇 Steps (步驟) 標籤。此索引標籤會顯示自動化動作的狀態。

自動化結束後，從已更新的 AMI 啟動測試執行個體以確認變更。

> **Note**
>
> 如果自動化中有任何步驟失敗，關於失敗的資訊會列於 Automation Executions (自動化執行清單) 頁面。自動化設計為在成功完成所有任務後終止暫時執行個體。如果有步驟失敗，系統可能不會終止執行個體。所以如果有步驟失敗，請手動終止暫時執行個體。

更新 Windows Server (AMI)

`AWS-UpdateWindowsAmi` Runbook 可讓您自動化 Amazon Windows Amazon Machine Image (AMI) 的映像維護任務，不必使用 JSON 或 YAML 撰寫 Runbook。此 Runbook 支援 Windows Server 2008 R2 或更新版本。您可以使用 `AWS-UpdateWindowsAmi` Runbook 執行以下類型的任務。

- 安裝所有 Windows 更新和升級 Amazon 軟體 (預設行為)。
- 安裝特定 Windows 更新和升級 Amazon 軟體。

- 使用您的指令碼自訂 AMI。

開始之前

在您開始使用 Runbook 之前，設定 Automation 的角色以新增 `iam:PassRole` 政策，此政策會參考您想要授予存取之執行個體設定檔的 ARN。或者，設定 Amazon EventBridge for Automation，這是其中的工具 AWS Systems Manager。如需詳細資訊，請參閱設定自動化。此演練也需要您指定 AWS Identity and Access Management (IAM) 執行個體設定檔的名稱。如需有關建立 IAM 執行個體設定檔的詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可。

> **Note**
>
> AWS Systems Manager SSM Agent 的更新通常會在不同時間於不同區域推出。自訂或更新 AMI 時，請只使用針對您工作之區域發佈的來源 AMI。這可確保您使用的是針對該區域發行的最新 SSM Agent，並且避免相容性問題。

`AWS-UpdateWindowsAmi` Runbook 接受以下的輸入參數。

參數	類型	描述
SourceAmiId	字串	(必要) 來源 AMI ID。您可以使用 Systems Manager Parameter Store 公有參數自動參考最新的 Windows Server AMI ID。如需詳細資訊，請參閱使用 AWS Systems Manager查詢最新的 Windows AMI ID Parameter Store。
SubnetId	字串	(選用) 您要啟動暫時執行個體的子網路。如果您已刪除預設 VPC，則必須為此參數指定一個值。
IamInstanceProfileName	字串	(必要) 您在設定 Systems Manager 所需的執行個體許可中所建立之 IAM 執行個體

參數	類型	描述
		設定檔角色的名稱。執行個體設定檔角色可讓自動化許可在您的執行個體上執行動作，例如執行命令或啟動和停用服務。Runbook 僅使用執行個體設定檔角色的名稱。
AutomationAssumeRole	字串	(必要) 您在 設定自動化 中建立之 IAM 服務角色的名稱。服務角色 (也稱為擔任角色) 會提供自動化許可來擔任您的 IAM 角色並代表您執行動作。例如，在 Runbook 中執行 `aws:createImage` 動作時，服務角色會允許自動化建立新的 AMI。針對此參數，必須指定的完整 ARN。
TargetAmiName	字串	(選用) 建立後的新 AMI 之名稱。預設名稱為系統產生的字串，包括來源 AMI ID，以及建立時間和日期。
InstanceType	字串	(選用) 做為工作空間主機啟動的執行個體類型。執行個體類型因區域而異。預設類型為 t2.medium。
PreUpdateScript	字串	(選用) 更新 AMI 之前執行的指令碼。在 Runbook 中或在執行時間輸入指令碼作為參數。
PostUpdateScript	字串	(選用) 更新 AMI 之後執行的指令碼。在 Runbook 中或在執行時間輸入指令碼作為參數。

參數	類型	描述
IncludeKbs	字串	(選用) 指定一個或多個要包含的 Microsoft 知識庫 (KB) 文章 ID。您可以使用逗號分隔值安裝多個 ID。有效格式：KB9876543 或 9876543。
ExcludeKbs	字串	(選用) 指定一個或多個要排除的 Microsoft 知識庫 (KB) 文章 ID。您可以使用逗號分隔值排除多個 ID。有效格式：KB9876543 或 9876543。
類別	字串	(選用) 指定一個或多個更新類別。您可以使用逗號分隔值篩選類別。選項：Critical Update (重大更新)，Security Update (安全性更新)，Definition Update (定義更新)，Update Rollup (更新彙總套件)，Service Pack，Tool (工具)，Update (更新)，或 Driver (驅動程式)。有效格式包含單一項目，例如：Critical Update (重大更新)。或者您可以指定逗號分隔清單：Critical Update,Security Update,Definition Update。

參數	類型	描述
SeverityLevels	字串	(選用) 指定一個或多個與更新關聯的 MSRC 嚴重性等級。您可以使用逗號分隔值篩選嚴重性等級。選項：Critical，Important，Low，Moderate 或 Unspecified。有效格式包括單一項目，例如：Critical。或者，您可以指定逗號分隔清單：Critical，Important，Low。

自動化步驟

依預設，AWS-UpdateWindowsAmi Runbook 包含下列步驟。

步驟 1：launchInstance (**aws:runInstances** 動作)

此步驟藉由指定之 SourceAmiID 的 IAM 執行個體設定檔角色來啟動執行個體。

步驟 2：runPreUpdateScript (**aws:runCommand** 動作)

此步驟可讓您指定指令碼做為字串，在更新安裝之前執行。

步驟 3：updateEC2Config (**aws:runCommand** 動作)

此步驟使用 AWS-InstallPowerShellModule Runbook 下載 AWS 公有 PowerShell 模組。Systems Manager 會使用 SHA-256 雜湊來驗證模組的完整性。接著，Systems Manager 會檢查作業系統，以判斷是否要更新 EC2Config 或 EC2Launch。EC2Config 透過 Windows Server 2012 R2 在 Windows Server 2008 R2 上執行。EC2Launch 在 Windows Server 2016 上執行。

步驟 4：updateSSMAgent (**aws:runCommand** 動作)

此步驟會藉由使用 AWS-UpdateSSMAgent Runbook 更新 SSM Agent。

步驟 5：updateAWSPVDriver (**aws:runCommand** 動作)

此步驟會使用 AWS-ConfigureAWSPackage Runbook 更新 AWS PV 驅動程式。

步驟 6：updateAwsEnaNetworkDriver (**aws:runCommand** 動作)

此步驟會使用 AWS-ConfigureAWSPackage Runbook 更新 AWS ENA Network 驅動程式。

步驟 7：installWindowsUpdates (**aws:runCommand** 動作)

此步驟會藉由使用 AWS-InstallWindowsUpdates Runbook 安裝 Windows 更新。根據預設，Systems Manager 會搜尋和安裝所有缺少的更新。您可以藉由指定以下參數變更預設行為：IncludeKbs、ExcludeKbs、Categories 或 SeverityLevels。

步驟 8：runPostUpdateScript (**aws:runCommand** 動作)

此步驟可讓您指定指令碼做為字串，在更新安裝之後執行。

步驟 9：runSysprepGeneralize (**aws:runCommand** 動作)

此步驟使用 AWS-InstallPowerShellModule Runbook 下載 AWS 公有 PowerShell 模組。Systems Manager 會使用 SHA-256 雜湊來驗證模組的完整性。Systems Manager 接著會使用 AWS支援的 EC2Launch (Windows Server 2016) 或 EC2Config (Windows Server 2008 R2 到 2012 R2) 方法執行 sysprep。

步驟 10：stopInstance 停止 (**aws:changeInstanceState** 動作)

此步驟會停止更新的執行個體。

步驟 11：createImage (**aws:createImage** 動作)

此步驟會以連結至來源 ID 和建立時間的描述性名稱建立新的 AMI。例如：「AMI Generated by EC2 Automation on {{global:DATE_TIME}} from {{SourceAmiId}}」，其中 DATE_TIME 和 SourceID 代表自動化變數。

步驟 12：TerminateInstance (**aws:changeInstanceState** 動作)

此步驟會藉由終止執行中的執行個體來清除自動化。

輸出

本節可讓您將各種步驟的輸出或任何參數的值指定為自動化輸出。根據預設，輸出是由自動化建立的已更新 Windows AMI 之 ID。

> **Note**
>
> 根據預設，自動化執行 AWS-UpdateWindowsAmi Runbook 和建立暫時執行個體時，系統會使用預設 VPC (172.30.0.0/16)。如果刪除預設 VPC，您會收到以下錯誤：
> VPC 未定義 400
> 若要解決此問題，您必須複製 AWS-UpdateWindowsAmi Runbook 並指定子網路 ID。如需詳細資訊，請參閱VPC 未定義 400。

使用自動化建立已修補的 Windows AMI

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

2. 執行下列命令，以執行 AWS-UpdateWindowsAmi Runbook。將每個########取代為您自己的資訊。以下的範例命令使用最新的 Amazon EC2 AMI，將需要套用的修補程式數量降至最低。如果執行此命令超過一次，您必須為 targetAMIname 指定一個唯一的值。AMI 名稱必須為唯一。

   ```
   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'
   ```

   命令會傳回執行 ID。複製此 ID 到剪貼簿。您可以使用此 ID 檢視自動化的狀態。

   ```
   {
       "AutomationExecutionId": "automation execution ID"
   }
   ```

3. 若要使用 檢視自動化 AWS CLI，請執行下列命令：

   ```
   aws ssm describe-automation-executions
   ```

4. 執行以下命令檢視自動化進度的詳細資訊。

   ```
   aws ssm get-automation-execution
       --automation-execution-id automation execution ID
   ```

> **Note**
>
> 在此範例自動化中執行的 Windows 修補程式可能需要 30 分鐘或以上的時間完成，取決於修補程式的數量。

AMI 使用自動化更新黃金 AWS Lambda，以及 Parameter Store

在以下範例使用的模型中，組織會維護並定期修補其自有的專屬 AMIs，而非從 Amazon Elastic Compute Cloud (Amazon EC2) AMIs 建立。

以下程序示範如何自動將作業系統 (OS) 修補程式套用至已視為是最近期或最新 AMI 的 AMI。在此範例中， 參數的預設值`SourceAmiId`是由名為 的 AWS Systems Manager Parameter Store參數定義`latestAmi`。值`latestAmi`會由自動化結束時叫用的 AWS Lambda 函數更新。由於此自動化程序，修補所花費的時間和精力AMIs會最小化，因為修補一律套用至up-to-date AMI。 Parameter Store 和 自動化是 的工具 AWS Systems Manager。

開始之前

設定 Automation 角色和 (選用) Automation 的 Amazon EventBridge。如需詳細資訊，請參閱設定自動化。

目錄

- 任務 1：建立 Systems Manager Parameter Store 參數
- 任務 2：建立 AWS Lambda的 IAM 角色
- 任務 3：建立 AWS Lambda 函數
- 任務 4：建立 Runbook 並修補 AMI

任務 1：建立 Systems Manager Parameter Store 參數

在 Parameter Store 中建立字串參數，使用以下資訊：

- Name (名稱)：`latestAmi`。
- 值：AMI ID。例如： `ami-188d6e0e`。

如需建立 Parameter Store 字串參數的詳細資訊，請參閱 在 Systems Manager 中建立 Parameter Store 參數。

任務 2：建立 AWS Lambda的 IAM 角色

使用下列程序來建立 的 IAM 服務角色 AWS Lambda。這些政策會提供 Lambda 許可，以使用 Lambda 函數和 Systems Manager 來更新 `latestAmi` 參數的值。

建立適用於 Lambda 的 IAM 服務角色

1. 登入 AWS Management Console ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇政策，然後選擇建立政策。
3. 選擇 JSON 標籤。
4. 將預設內容取代為以下政策。將每個########取代為您自己的資訊。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "logs:CreateLogGroup",
            "Resource": "arn:aws:logs:region:123456789012:*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:region:123456789012:log-group:/aws/lambda/function
 name:*"
            ]
        }
    ]
}
```

5. 選擇下一步：標籤。
6. (選用) 新增一個或多個標籤鍵值組來組織、追蹤或控制對此政策的存取。
7. 選擇下一步：檢閱。
8. 在檢閱政策頁面上名稱中，輸入該內嵌政策的名稱，例如 **amiLambda**。
9. 選擇 建立政策。
10. 重複步驟 2 和 3。
11. 貼上下列政策。將每個########取代為您自己的資訊。

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ssm:PutParameter",
            "Resource": "arn:aws:ssm:region:123456789012:parameter/latestAmi"
        },
        {
            "Effect": "Allow",
            "Action": "ssm:DescribeParameters",
            "Resource": "*"
        }
    ]
}
```

12. 選擇下一步：標籤。
13. (選用) 新增一個或多個標籤鍵值組來組織、追蹤或控制對此政策的存取。
14. 選擇下一步：檢閱。
15. 在檢閱政策頁面上名稱中，輸入該內嵌政策的名稱，例如 **amiParameter**。
16. 選擇 建立政策。
17. 在導覽窗格中，選擇角色，然後選擇建立角色。
18. 在使用案例下，隨即選擇 Lambda，然後選擇下一步。
19. 在新增許可頁面上，使用搜尋欄位找出您之前建立的兩個政策。
20. 選取政策旁的核取方塊，然後選擇下一步。
21. 對於 Role name (角色名稱)，輸入新角色的名稱，例如 **lambda-ssm-role** 或另一個您喜好的名稱。

> **Note**
>
> 因為有各種實體可能會參照角色，所以您無法在建立角色之後變更角色名稱。

22. (選用) 新增一或多個標籤鍵值組來整理、追蹤或控制此角色的存取權，然後選擇建立角色。

任務 3：建立 AWS Lambda 函數

使用以下程序建立可自動更新 `latestAmi` 參數值的 Lambda 函數。

建立 Lambda 函式

1. 登入 AWS Management Console ，並在 https：//https://console.aws.amazon.com/lambda/ 開啟 AWS Lambda 主控台。
2. 選擇 Create function (建立函數)。
3. 在 Create function (建立函數) 頁面上，選擇 Author from scratch (從頭開始撰寫)。
4. 針對函數名稱，請輸入 **Automation-UpdateSsmParam**。
5. 針對執行階段，選擇 Python 3.8。
6. 在架構中，選取 Lambda 用來執行函數的電腦處理器類型：x86_64 或 arm64。
7. 在許可區段中，展開變更預設執行角色。
8. 選擇 Use an existing role (使用現有角色)，然後為您在任務 2 中建立的 Lambda 選擇服務角色。
9. 選擇 Create function (建立函數)。
10. 在程式碼來源區域的 lambda_function 索引標籤中，刪除欄位中預先填入的程式碼，接著貼上以下範本程式碼。

```
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. 選擇檔案、儲存。
12. 若要測試 Lambda 函數，請從測試功能表，選擇設定測試事件。
13. 針對 Event name (事件名稱)，輸入測試事件的名稱，例如 **MyTestEvent**。
14. 將現有文字取代為以下的 JSON。把 *AMI ID* 取代為您自己的資訊以設定 `latestAmi` 參數值。

```
{
   "parameterName":"latestAmi",
   "parameterValue":"AMI ID"
}
```

15. 選擇 Save (儲存)。
16. 選擇 Test (測試) 以測試函數。在執行結果索引標籤上，狀態應報告為成功，同時包含與更新有關的其他詳細資訊。

任務 4：建立 Runbook 並修補 AMI

使用以下程序建立和執行 Runbook，以修補您針對 latestAmi 參數指定的 AMI。自動化工作流程完成後，latestAmi 的值會以新修補的 AMI 之 ID 更新。後續的自動化會使用由先前執行建立的 AMI。

建立和執行 Runbook

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Documents (文件)。
3. 在建立文件中，選擇自動化。
4. 對於名稱，輸入 **`UpdateMyLatestWindowsAmi`**。
5. 選擇 Editor (編輯器) 標籤，然後選擇 Edit (編輯)。
6. 出現提示時選擇確定。
7. 在文件編輯器欄位中，使用以下 YAML 範例執行手冊內容取代預設內容。

```
---
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. 選擇 Create automation (建立自動化)。
9. 在導覽窗格中，選擇 Automation (自動化)，接著選擇 Execute automation (執行自動化)。
10. 在 Choose document (選擇文件) 頁面中，選擇 Owned by me (我所擁有) 索引標籤。
11. 搜尋 UpdateMyLatestWindowsAmi Runbook，然後選擇 UpdateMyLatestWindowsAmi 卡中的按鈕。
12. 選擇 Next (下一步)。
13. 選擇 Simple execution (簡單執行)。
14. 請為輸入參數指定值。
15. 選擇 Execute (執行)。
16. 自動化完成後，在導覽窗格中選擇 Parameter Store 並確認 `latestAmi` 的新值符合自動化傳回的值。您也可以在 Amazon EC2 主控台的 AMIs 部分確認新的 AMI ID 符合自動化輸出。

使用 Automation 和 Jenkins 更新 AMIs

如果組織是在 CI/CD 管道中使用 Jenkins 軟體，則可以將 Automation 作為後建置步驟新增，以將應用程式版本預先安裝到 Amazon Machine Images (AMIs) 中。自動化是 中的工具 AWS Systems Manager。也可以使用 Jenkins 排程功能呼叫 Automation 並建立自己的作業系統 (OS) 修補頻率。

以下範例顯示如何從內部部署或在 Amazon Elastic Compute Cloud (Amazon EC2) 中執行的 Jenkins 伺服器調用 Automation。對於身分驗證，Jenkins伺服器會根據您在範例中建立的 IAM 政策使用 AWS 登入資料，並連接至執行個體描述檔。

> **Note**
>
> 設定執行個體時，請務必遵照 Jenkins 安全最佳實務。

開始之前

在透過 Jenkins 設定 Automation 之前，請完成以下任務：

- 完成 AMI 使用自動化更新黃金 AWS Lambda，以及 Parameter Store 範例。以下範例使用在該範例中建立的 UpdateMyLatestWindowsAmi Runbook。
- 為 Automation 設定 IAM 角色。Systems Manager 需要執行個體設定檔角色和服務角色 ARN 以處理自動化。如需詳細資訊，請參閱設定自動化。

為 Jenkins 伺服器建立 IAM 政策

1. 登入 AWS Management Console ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇政策，然後選擇建立政策。
3. 選擇 JSON 標籤。
4. 將每個########取代為您自己的資訊。

```
{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": "ssm:StartAutomationExecution",
                "Resource": [
```

```
                    "arn:aws:ssm:region:account ID:document/
UpdateMyLatestWindowsAmi",
                    "arn:aws:ssm:region:account ID:automation-definition/
UpdateMyLatestWindowsAmi:$DEFAULT"
                ]
            }
        ]
    }
```

5. 選擇檢閱政策。
6. 在檢閱政策頁面上名稱中，輸入該內嵌政策的名稱，例如 **JenkinsPolicy**。
7. 選擇 建立政策。
8. 在導覽窗格中，選擇角色。
9. 選擇連接至 Jenkins 伺服器的執行個體設定檔。
10. 在許可索引標籤上，依序選擇新增許可、連接政策。
11. 在其他許可政策區段中，輸入您在之前的步驟中建立的政策名稱。例如，JenkinsPolicy。
12. 勾選政策旁邊的方塊，然後選擇連接政策。

使用下列程序在您的 AWS CLI Jenkins伺服器上設定 。

為 Automation 設定 Jenkins 伺服器

1. 使用您偏好的瀏覽器存取管理介面，連線至連接埠 8080 的 Jenkins 伺服器。
2. 輸入在 `/var/lib/jenkins/secrets/initialAdminPassword` 中找到的密碼。若要顯示您的密碼，請執行下列命令。

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

3. Jenkins 安裝指令碼會將您引導至自訂 Jenkins 頁面。選取 Install suggested plugins (安裝建議的外掛程式)。
4. 完成安裝之後，請選擇管理員憑證，選取儲存憑證，然後選取開始使用 Jenkins。
5. 在左側導覽窗格中，選擇管理 Jenkins，然後選擇管理外掛程式。
6. 選擇 Available (可用) 索引標籤，然後輸入 **Amazon EC2 plugin**。
7. 選取 **Amazon EC2 plugin** 的核取方塊，然後選取 Install without restart (安裝無需重新啟動)。
8. 當完成安裝時，請選取 Go back to the top page (回到首頁)。
9. 選擇 管理 Jenkins，然後選擇管理節點和雲端。

10. 在設定雲端區段中，選取新增雲端，然後選擇 Amazon EC2。
11. 在剩餘欄位中輸入您的資訊。請務必選取使用 EC2 執行個體設定檔取得憑證選項。

按照下列程序設定 Jenkins 專案以調用 Automation。

設定 Jenkins 伺服器以調用 Automation

1. 在 Web 瀏覽器中開啟 Jenkins 主控台。
2. 選擇您想要以自動化設定的專案，接著選擇 Configure (設定)。
3. 在 Build (建置) 索引標籤，選擇 Add Build Step (新增建置步驟)。
4. 選擇 Execute shell (執行 shell) 或 Execute Windows batch command (執行 Windows 批次命令) (取決於您的作業系統)。
5. 在命令欄位中，執行如下所示的 AWS CLI 命令。將每個########取代為您自己的資訊。

```
aws ssm start-automation-execution \
        --document-name runbook name \
        --region AWS ## of your source AMI \
        --parameters runbook parameters
```

下列範例命令使用 UpdateMyLatestWindowsAmi Runbook 和 AMI 使用自動化更新黃金 AWS Lambda，以及 Parameter Store 中建立的 Systems Manager 參數 `latestAmi`。

```
aws ssm start-automation-execution \
        --document-name UpdateMyLatestWindowsAmi \
        --parameters \
            "sourceAMIid='{{ssm:latestAmi}}'"
        --region region
```

在 Jenkins 中，命令類似於以下螢幕擷取畫面中的範例。

6. 在 Jenkins 專案中，選擇立即建置。Jenkins 會傳回類似於下列範例的輸出。

更新 Auto Scaling 群組的 AMIs

下列範例使用新修補的 AMI 更新 Auto Scaling 群組。此方法可確保新的映像會自動提供給使用 Auto Scaling 群組的不同運算環境。

此範例中自動化的最後步驟使用 Python 函數來建立使用新修補 AMI 的啟動範本。然後，更新 Auto Scaling 群組以使用新的啟動範本。在此類型的 Auto Scaling 情況下，使用者可以在 Auto Scaling 群組中終止現有的執行個體，以強制新的執行個體啟動並使用新映像。或者，使用者可以等待並允許縮減或擴展事件自然啟動較新的執行個體。

開始之前

開始此範例之前，請先完成以下任務。

- 設定 Automation 的 IAM 角色，這是其中的工具 AWS Systems Manager。Systems Manager 需要執行個體設定檔角色和服務角色 ARN 以處理自動化。如需詳細資訊，請參閱設定自動化。

建立 PatchAMIAndUpdateASG Runbook

請使用下列程序建立 PatchAMIAndUpdateASG Runbook，其會修補為 SourceAMI 參數指定的 AMI。此 Runbook 也會更新 Auto Scaling 群組以使用最新且經修補的 AMI。

建立和執行 Runbook

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Documents (文件)。
3. 在 Create document (建立文件) 下拉式清單中，選擇 Automation (自動化)。
4. 在 Name (名稱) 欄位中，輸入 **PatchAMIAndUpdateASG**。
5. 選擇 Editor (編輯器) 索引標籤，然後選擇 Edit (編輯)。
6. 出現提示時選擇 OK (確定)，然後在 Document editor (文件編輯器) 欄位中刪除內容。
7. 在 Document editor (文件編輯器) 欄位中，貼上下列 YAML 範例 Runbook 內容。

```
---
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. 選擇 Create automation (建立自動化)。
9. 在導覽窗格中，選擇 Automation (自動化)，接著選擇 Execute automation (執行自動化)。
10. 在 Choose document (選擇文件) 頁面中，選擇 Owned by me (我所擁有) 索引標籤。
11. 搜尋 PatchAMIAndUpdateASG Runbook，然後選擇 PatchAMIAndUpdateASG 卡中的按鈕。
12. 選擇 Next (下一步)。
13. 選擇 Simple execution (簡單執行)。
14. 為輸入參數指定值。確定您指定的 `SubnetId` 和 `SecurityGroupIds` 允許存取公用 Systems Manager 端點，或是 Systems Manager 的介面端點。
15. 選擇 Execute (執行)。
16. 自動化完成後，在 Amazon EC2 主控台選擇 Auto Scaling，接著選擇 Launch Templates (啟動範本)。確認您已看到新的啟動範本，且其使用新的 AMI。
17. 選擇 Auto Scaling (Auto Scaling)，然後選擇 Auto Scaling Groups (Auto Scaling 群組)。確認 Auto Scaling 群組使用新的啟動範本。
18. 在您的 Auto Scaling 群組中終止一個或多個執行個體。取代執行個體會以新的 AMI 啟動。

使用 AWS 支援 自助式 Runbook

本節說明如何使用 AWS 支援 團隊建立的一些自助式自動化。這些自動化可協助您管理 AWS 資源。

支援 Automation 工作流程

支援自動化工作流程 (SAW) 是由 AWS 支援 團隊撰寫和維護的自動化執行手冊。這些 Runbook 可協助您疑難排解 AWS 資源的常見問題、主動監控和識別網路問題、收集和分析日誌等。

SAW Runbook 使用 **AWSSupport** 字首。例如：AWSSupport-ActivateWindowsWithAmazonLicense。

此外， AWS Enterprise 和 Business Support 客戶也可以存取使用 **AWSPremiumSupport**字首的 Runbook。例如：AWSPremiumSupport-TroubleshootEC2DiskUsage。

若要進一步了解 AWS 支援，請參閱 入門 AWS 支援。

主題

- 在無法觸達的執行個體上執行 EC2Rescue 工具
- 在 EC2 執行個體上重設密碼和 SSH 金鑰

在無法觸達的執行個體上執行 EC2Rescue 工具

EC2Rescue 可協助您對 Linux 和 Windows Server Amazon Elastic Compute Cloud (Amazon EC2) 執行個體的問題進行故障診斷。您可以手動執行工具，請參閱 Using EC2Rescue for Linux Server (使用適用於 Linux 伺服器的 EC2Rescue) 和 Using EC2Rescue for Windows Server (使用適用於 Windows Server 的 EC2Rescue)。或者，您可以使用 Systems Manager Automation 和 **AWSSupport-ExecuteEC2Rescue** Runbook 自動執行工具。自動化是 中的工具 AWS Systems Manager。**AWSSupport-ExecuteEC2Rescue** Runbook 旨在執行一組 Systems Manager 動作、AWS CloudFormation 動作和 Lambda 函數，以將通常必須使用 EC2Rescue 的步驟自動化。

您可以使用 **AWSSupport-ExecuteEC2Rescue** Runbook 針對不同類型的作業系統 (OS) 問題疑難排解並可能修復。具有加密根磁碟區的執行個體不受支援。請參閱下列主題以取得完整的清單：

Windows：請參閱將 EC2Rescue for Windows Server 與命令列搭配使用中的救援動作。

Linux 和 macOS：某些適用於 Linux 模組的 EC2Rescue 會偵測並嘗試修復問題。如需詳細資訊，請參閱 GitHub 上針對各模組的 aws-ec2rescue-linux 文件。

運作方式

使用 Automation 和 **AWSSupport-ExecuteEC2Rescue** Runbook 對執行個體進行故障診斷的運作方式如下：

- 您要指定無法連線的執行個體 ID 並執行 Runbook。
- 系統會建立暫時 VPC，接著執行一系列 Lambda 函數以設定 VPC。
- 系統會在與原始執行個體相同的可用區域中為您的暫時 VPC 識別出子網路。
- 系統會啟動暫時且啟用 SSM 的協助程式執行個體。
- 系統會停止您的原始執行個體，並建立備份。接著系統會將原始根磁碟區連接至協助程式執行個體。
- 系統會使用 Run Command 在協助程式執行個體上執行 EC2Rescue。EC2Rescue 會在已連接的原始根磁碟區上識別並嘗試修正問題。完成後，EC2Rescue 會重新將根磁碟區連接回原始執行個體。
- 系統會重新啟動您的原始執行個體，並終止暫時執行個體。系統也會終止在自動化開始時建立的暫時 VPC 和 Lambda 函數。

開始之前

執行以下自動化之前，請先執行以下項目：

- 複製無法連線之執行個體的執行個體 ID。您會在程序中指定此 ID。
- 或者，收集與無法連線之執行個體位於相同可用區域中的子網路 ID。EC2Rescue 執行個體會在此子網路中建立。如果您未指定子網路，則 Automation 會在您的 中建立新的臨時 VPC AWS 帳戶。確認您的 至少 AWS 帳戶 有一個可用的 VPC。根據預設，您可以在一個區域中建立五個 VPC。如果您已經在區域中建立五個 VPC，則自動化會失敗且不會變更您的執行個體。如需有關 Amazon VPC 配額的詳細資訊，請參閱《Amazon VPC 使用者指南》中的 VPC 和子網路。
- 或者，您可以為自動化建立和指定 AWS Identity and Access Management (IAM) 角色。如果您不指定此角色，則自動化會在執行自動化的使用者內容中執行。

授予 **AWSSupport-EC2Rescue** 在執行個體上執行動作的許可

EC2Rescue 需要許可才能在自動化執行期間於您的執行個體一系列動作。這些動作會叫用 AWS Lambda、IAM 和 Amazon EC2 服務，以安全且安全地嘗試修復執行個體的問題。如果您在 AWS 帳戶和/或 VPC 中具有管理員層級許可，則無需設定許可即可執行自動化，如本節所述。如果您沒有管理員層級的許可，則您或管理員必須使用以下其中一個選項來設定許可。

- 使用 IAM 政策授予許可

- 使用 AWS CloudFormation 範本授予許可

使用 IAM 政策授予許可

您可以將以下 IAM 政策做為內嵌政策連接至使用者、群組或角色；或者，您可以建立新的 IAM 受管政策並將其連接至使用者、群組或角色。如需有關新增內嵌政策至使用者、群組或角色的詳細資訊，請參閱使用內嵌政策。如需建立新受管政策的詳細資訊，請參閱使用受管政策。

> **Note**
>
> 如果您建立新的 IAM 受管政策，您也必須為其連接 AmazonSSMAutomationRole 受管政策，以使您的執行個體能夠與 Systems Manager API 通訊。

AWSSupport-EC2Rescue 的 IAM 政策

把*## ID* 取代為您自己的資訊。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Action": [
            "lambda:InvokeFunction",
            "lambda:DeleteFunction",
            "lambda:GetFunction"
         ],
         "Resource": "arn:aws:lambda:*:account ID: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::account ID:role/AWSSupport-EC2Rescue-*",
            "arn:aws:iam::account ID: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 範本授予許可

AWS CloudFormation 會使用預先設定的範本，自動建立 IAM 角色和政策的程序。藉由以下程序，使用 AWS CloudFormation為 EC2Rescue 自動化建立所需的 IAM 角色和政策。

為 EC2Rescue 建立所需的 IAM 角色和政策

1. 下載 AWSSupport-EC2RescueRole.zip 並將 AWSSupport-EC2RescueRole.json 檔案解壓縮至本機電腦上的目錄。
2. 如果您的 AWS 帳戶 位於特殊分割區中，請編輯範本，將 ARN 值變更為分割區的 ARN 值。

   例如，對於中國區域，將 arn:aws 的所有案例變更為 arn:aws-cn。
3. 登入 AWS Management Console ，並在 https：//https://console.aws.amazon.com/cloudformation 開啟 AWS CloudFormation 主控台。
4. 選擇 Create stack (建立堆疊)、With new resources (standard) (使用新資源 (標準))。
5. 在 Create stack (建立堆疊) 頁面上，對於 Prerequisite - Prepare template (先決條件 - 準備範本)，選擇 Template is ready (範本已準備就緒)。
6. 對於 Specify template (指定範本)，選擇 Upload a template file (上傳範本檔案)。
7. 選擇 Choose file (選擇檔案)，然後瀏覽並從您解壓縮檔案的目錄中選取 AWSSupport-EC2RescueRole.json 檔案。
8. 選擇 Next (下一步)。
9. 在 Specify stack details (指定堆疊詳細資訊) 頁面上，對於 Stack name (堆疊名稱) 欄位，輸入識別此堆疊的名稱，然後選擇 Next (下一步)。
10. (選用) 在 Tags (標籤) 區域中將一個或多個標籤索引鍵名稱/值對套用到堆疊。

    標籤是您指派給資源的選用性中繼資料。標籤可讓您以不同的方式 (例如用途、擁有者或環境) 將資源分類。例如，您可能想要標記堆疊來識別其執行的任務類型、相關的目標類型或其他資源，以及其執行所在的環境。
11. 選擇 Next (下一步)

12. 在檢閱頁面上，檢閱堆疊詳細資訊，然後向下捲動並選擇我確認 AWS CloudFormation 可能會建立 IAM 資源選項。
13. 選擇建立堆疊。

    AWS CloudFormation 會顯示 CREATE_IN_PROGRESS 狀態幾分鐘。堆疊建立之後，狀態會變更為 CREATE_COMPLETE (CREATE_COMPLETE)。您也可以選擇重新整理圖示來檢查建立程序的狀態。
14. 在 Stacks (堆疊) 清單中，選擇您剛建立堆疊的選項按鈕，然後選擇 Outputs (輸出) 索引標籤。
15. 請記下 Value (值)。此為 AssumeRole 的 ARN。當您在下一個程序執行自動化中執行自動化時，請指定此 ARN。

執行自動化

> **Important**
>
> 以下自動化工作流程會停止無法連線的執行個體。停止執行個體可能會導致已連接執行個體存放磁碟區上的資料遺失 (若有)。停止執行個體也可能會導致公有 IP 變更 (若無關聯的彈性 IP)。

執行 **AWSSupport-ExecuteEC2Rescue** 自動化。

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Automation (自動化)。
3. 選擇 Execute automation (執行自動化)。
4. 在 Automation document (自動化文件) 部分，從清單選擇 Owned by Amazon (由 Amazon 所有)。
5. 在 Runbook 清單中，選擇 **AWSSupport-ExecuteEC2Rescue** 卡片中的按鈕，然後選擇 Next (下一步)。
6. 在 Execute automation document (執行自動化文件) 頁面上，選擇 Simple execution (簡易執行)。
7. 在 Document details (文件詳細資訊) 部分，確認 Document version (文件版本) 設為最高的預設版本。例如，$DEFAULT 或 3 (default) (3 (預設))。
8. 在 Input parameters (輸入參數) 區段中，指定以下參數：

   a. 針對 UnreachableInstanceId，指定無法連線之執行個體的 ID。

b. (選用) 針對 EC2RescueInstanceType，指定 EC2Rescue 執行個體的執行個體類型。預設執行個體類型為 `t2.medium`。

c. 對於 AutomationAssumeRole，如果您使用本主題稍早所述的 AWS CloudFormation 程序來建立此自動化的角色，請選擇您在 AWS CloudFormation 主控台中建立之 AssumeRole 的 ARN。

d. (選用) 針對 LogDestination，如果您在對解執行個體進行故障診斷時想要收集作業系統層級的日誌，請指定 S3 儲存貯體。日誌會自動上傳至指定的儲存貯體。

e. 針對 SubnetId，在與無法連線之執行個體位於相同可用區域的現有 VPC 中指定子網路。根據預設，Systems Manager 會建立新的 VPC，但您也可以在現有 VPC 中指定子網路。

> **Note**
>
> 如果您沒見到指定儲存貯體或子網路 ID 的選項，請確認您使用的是 Runbook 最新的 Default (預設) 版本。

9. (選用) 在 Tags (標籤) 區域中，套用一個或多個標籤索引鍵名稱/值對以協助識別自動化，例如 `Key=Purpose,Value=EC2Rescue`。
10. 選擇 Execute (執行)。

作為自動化的一部分，Runbook 會建立備份 AMI。其他所有由自動化建立的資源都會自動刪除，但此 AMI 會保留於您的帳戶。AMI 使用以下慣例命名：

備份 AMI：AWSSupport-EC2Rescue:*UnreachableInstanceId*

您可以搜尋自動化執行 ID 以在 Amazon EC2 主控台找到此 AMI。

在 EC2 執行個體上重設密碼和 SSH 金鑰

您可以使用 `AWSSupport-ResetAccess` 執行手冊來自動恢復在 Windows Server 的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上產生本機管理員密碼，以及在 Linux EC2 執行個體上產生新的 SSH 金鑰。`AWSSupport-ResetAccess` Runbook 旨在執行 AWS Systems Manager 動作、AWS CloudFormation 動作和函數的組合，這些動作和 AWS Lambda 函數會自動執行重設本機管理員密碼通常所需的步驟。

您可以使用 中的 Automation 工具 AWS Systems Manager搭配 `AWSSupport-ResetAccess` Runbook 來解決下列問題：

Windows

遺失 EC2 金鑰對：若要解決此問題，您可以使用 AWSSupport-ResetAccess Runbook 從您目前的執行個體建立啟用密碼的 AMI，從 AMI 啟動新的執行個體，接著選取您自己的金鑰對。

遺失本機管理員密碼：若要解決此問題，您可以使用 `AWSSupport-ResetAccess` Runbook 產生可藉由目前的 EC2 金鑰對解密的新密碼。

Linux

遺失 EC2 金鑰對，或遺失設定執行個體之 SSH 存取的金鑰：若要解決此問題，您可以使用 `AWSSupport-ResetAccess` Runbook 為目前的執行個體建立新的 SSH 金鑰，這可讓您再次連線至執行個體。

> **Note**
>
> 如果您的適用於 Windows Server 的 EC2 執行個體是針對 Systems Manager 設定，您也可以使用 EC2Rescue 和 AWS Systems Manager Run Command 重設本機管理員密碼。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的將 EC2Rescue for Windows Server 與 Systems Manager Run Command 搭配使用。

相關資訊

《Amazon EC2 使用者指南》中的使用 PuTTY 從 Windows 連線至 Linux 執行個體。

運作方式

使用 Automation 和 `AWSSupport-ResetAccess` Runbook 對執行個體進行故障診斷的運作方式如下：

- 您要指定執行個體 ID 並執行 Runbook。
- 系統會建立暫時 VPC，接著執行一系列 Lambda 函數以設定 VPC。
- 系統會在與原始執行個體相同的可用區域中為您的暫時 VPC 識別出子網路。
- 系統會啟動暫時且啟用 SSM 的協助程式執行個體。
- 系統會停止您的原始執行個體，並建立備份。接著系統會將原始根磁碟區連接至協助程式執行個體。
- 系統會使用 Run Command 在協助程式執行個體上執行 EC2Rescue。在 Windows 上，EC2Rescue 會於已連接的原始根磁碟區使用 EC2Config 或 EC2Launch，藉此啟用本機管理員的密碼產生。在 Linux 上，EC2Rescue 會產生和插入新的 SSH 金鑰，並將私密金鑰加密儲存於 Parameter Store。完成後，EC2Rescue 會重新將根磁碟區連接回原始執行個體。

- 密碼產生啟用後，系統會在您的執行個體建立新的 Amazon Machine Image (AMI)。您可以使用此 AMI 建立新的 EC2 執行個體，並視需要與新的金鑰對建立關聯。
- 系統會重新啟動您的原始執行個體，並終止暫時執行個體。系統也會終止在自動化開始時建立的暫時 VPC 和 Lambda 函數。
- Windows：您的執行個體會產生新密碼，而您可以使用目前指派給執行個體的金鑰對從 Amazon EC2 主控台解碼。

  Linux：SSH 金鑰以 /ec2rl/openssh/***instance ID***/key 存放在 Systems Manager 參數存放區，您可用此金鑰 SSH 至執行個體。

開始之前

執行以下自動化之前，請先執行以下項目：

- 複製您想要重設管理員密碼的執行個體之執行個體 ID。您會在程序中指定此 ID。
- 或者，收集與無法連線之執行個體位於相同可用區域中的子網路 ID。EC2Rescue 執行個體會在此子網路中建立。如果您未指定子網路，則 Automation 會在您的 中建立新的臨時 VPC AWS 帳戶。確認您的 至少 AWS 帳戶 有一個可用的 VPC。根據預設，您可以在一個區域中建立五個 VPC。如果您已經在區域中建立五個 VPC，則自動化會失敗且不會變更您的執行個體。如需有關 Amazon VPC 配額的詳細資訊，請參閱《Amazon VPC 使用者指南》中的 VPC 和子網路。
- 或者，您可以為自動化建立和指定 AWS Identity and Access Management (IAM) 角色。如果您不指定此角色，則自動化會在執行自動化的使用者內容中執行。

授予 AWSSupport-EC2Rescue 在執行個體上執行動作的許可

EC2Rescue 需要許可才能在自動化執行期間於您的執行個體一系列動作。這些動作會叫用 AWS Lambda、IAM 和 Amazon EC2 服務，以安全且安全地嘗試修復執行個體的問題。如果您在 AWS 帳戶和/或 VPC 中具有管理員層級許可，則無需設定許可即可執行自動化，如本節所述。如果您沒有管理員層級的許可，則您或管理員必須使用以下其中一個選項來設定許可。

- 使用 IAM 政策授予許可
- 使用 AWS CloudFormation 範本授予許可

使用 IAM 政策授予許可

您可以將以下 IAM 政策做為內嵌政策連接至使用者、群組或角色；或者，您可以建立新的 IAM 受管政策並將其連接至使用者、群組或角色。如需有關新增內嵌政策至使用者、群組或角色的詳細資訊，請參閱使用內嵌政策。如需建立新受管政策的詳細資訊，請參閱使用受管政策。

> **Note**
>
> 如果您建立新的 IAM 受管政策，您也必須為其連接 AmazonSSMAutomationRole 受管政策，以使您的執行個體能夠與 Systems Manager API 通訊。

AWSSupport-ResetAccess 的 IAM 政策

把*## ID* 取代為您自己的資訊。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Action": [
            "lambda:InvokeFunction",
            "lambda:DeleteFunction",
            "lambda:GetFunction"
         ],
         "Resource": "arn:aws:lambda:*:account ID: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::account ID:role/AWSSupport-EC2Rescue-*",
            "arn:aws:iam::account ID: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 範本授予許可

AWS CloudFormation 會使用預先設定的範本，自動建立 IAM 角色和政策的程序。藉由以下程序，使用 AWS CloudFormation為 EC2Rescue 自動化建立所需的 IAM 角色和政策。

為 EC2Rescue 建立所需的 IAM 角色和政策

1. 下載 AWSSupport-EC2RescueRole.zip 並將 AWSSupport-EC2RescueRole.json 檔案解壓縮至本機電腦上的目錄。
2. 如果您的 AWS 帳戶 位於特殊分割區中，請編輯範本，將 ARN 值變更為分割區的 ARN 值。

   例如，對於中國區域，將 arn:aws 的所有案例變更為 arn:aws-cn。
3. 登入 AWS Management Console ，並在 https：//https://console.aws.amazon.com/cloudformation 開啟 AWS CloudFormation 主控台。
4. 選擇 Create stack (建立堆疊)、With new resources (standard) (使用新資源 (標準))。
5. 在 Create stack (建立堆疊) 頁面上，對於 Prerequisite - Prepare template (先決條件 - 準備範本)，選擇 Template is ready (範本已準備就緒)。
6. 對於 Specify template (指定範本)，選擇 Upload a template file (上傳範本檔案)。
7. 選擇 Choose file (選擇檔案)，然後瀏覽並從您解壓縮檔案的目錄中選取 AWSSupport-EC2RescueRole.json 檔案。
8. 選擇 Next (下一步)。
9. 在 Specify stack details (指定堆疊詳細資訊) 頁面上，對於 Stack name (堆疊名稱) 欄位，輸入識別此堆疊的名稱，然後選擇 Next (下一步)。
10. (選用) 在 Tags (標籤) 區域中將一個或多個標籤索引鍵名稱/值對套用到堆疊。

    標籤是您指派給資源的選用性中繼資料。標籤可讓您以不同的方式 (例如用途、擁有者或環境) 將資源分類。例如，您可能想要標記堆疊來識別其執行的任務類型、相關的目標類型或其他資源，以及其執行所在的環境。
11. 選擇 Next (下一步)
12. 在檢閱頁面上，檢閱堆疊詳細資訊，然後向下捲動並選擇我確認 AWS CloudFormation 可能會建立 IAM 資源選項。

13. AWS CloudFormation 會顯示 CREATE_IN_PROGRESS 狀態幾分鐘。堆疊建立之後，狀態會變更為 CREATE_COMPLETE (CREATE_COMPLETE)。您也可以選擇重新整理圖示來檢查建立程序的狀態。
14. 在堆疊清單中，選擇您剛建立堆疊旁的選項，然後選擇 Outputs (輸出) 標籤。
15. 複製 Value (值)。此為 AssumeRole 的 ARN。您會在執行自動化時指定此 ARN。

執行自動化

下列程序說明如何使用 AWS Systems Manager 主控台來執行 `AWSSupport-ResetAccess` Runbook。

> **Important**
>
> 以下自動化會停止執行個體。停止執行個體可能會導致已連接執行個體存放磁碟區上的資料遺失 (若有)。停止執行個體也可能會導致公有 IP 變更 (若無關聯的彈性 IP)。為了避免這些組態變更，請使用 Run Command 重設存取。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的將 EC2Rescue for Windows Server 與 Systems Manager Run Command 搭配使用。

執行 AWSSupport-ResetAccess 自動化

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Automation (自動化)。
3. 選擇 Execute automation (執行自動化)。
4. 在 Automation document (自動化文件) 部分，從清單選擇 Owned by Amazon (由 Amazon 所有)。
5. 在 Runbook 清單中，選擇 AWSSupport-ResetAccess 卡片中的按鈕，然後選擇 Next (下一步)。
6. 在 Execute automation document (執行自動化文件) 頁面上，選擇 Simple execution (簡易執行)。
7. 在 Document details (文件詳細資訊) 部分，確認 Document version (文件版本) 設為最高的預設版本。例如，$DEFAULT 或 3 (default) (3 (預設))。
8. 在 Input parameters (輸入參數) 區段中，指定以下參數：

    a. 針對 InstanceID，指定無法連線之執行個體的 ID。

    b. 針對 SubnetId，在與您指定之執行個體位於相同可用區域的現有 VPC 中指定子網路。根據預設，Systems Manager 會建立新的 VPC，但您也可以在現有 VPC 中指定子網路。

> **Note**
>
> 如果您沒見到指定子網路 ID 的選項，請確認您使用的是 Runbook 最新的 Default (預設) 版本。

c. 針對 EC2RescueInstanceType，指定 EC2Rescue 執行個體的執行個體類型。預設執行個體類型為 `t2.medium`。

d. 針對 AssumeRole，如果您使用本主題先前所述的 AWS CloudFormation 程序來建立此自動化的角色，請指定您在 AWS CloudFormation 主控台中記下的 AssumeRole ARN。

9. (選用) 在 Tags (標籤) 區域中，套用一個或多個標籤索引鍵名稱/值對以協助識別自動化，例如 `Key=Purpose,Value=ResetAccess`。

10. 選擇 Execute (執行)。

11. 若要監控自動化進度，請選擇執行中的自動化，接著選擇 Steps (步驟) 標籤。自動化結束時，選擇 Descriptions (描述) 標籤，接著選擇 View output (檢視輸出) 以檢視結果。若要檢視個別步驟的輸出，請選擇 Steps (步驟) 標籤，然後選擇步驟旁的 View Outputs (檢視輸出)。

作為自動化的一部分，Runbook 會建立備份 AMI 和已啟用密碼的 AMI。其他所有由自動化建立的資源都會自動刪除，但此 AMIs 會保留於您的帳戶。這些 AMIs 使用以下慣例命名：

- 備份 AMI：`AWSSupport-EC2Rescue:`*`InstanceID`*
- 密碼啟動的 AMI：AWSSupport-EC2Rescue：從*`#### ID`* 以密碼啟動的 AMI

您可以搜尋自動化執行 ID 以找到這些 AMIs。

針對 Linux，執行個體的新 SSH 私密金鑰會加密儲存於 Parameter Store。參數名稱為 /ec2rl/openssh/***`instance ID`***/key。

使用輸入轉換器將資料傳遞至 Automation

本 AWS Systems Manager 自動化教學課程說明如何使用 Amazon EventBridge 的輸入轉換器功能，從執行個體狀態變更事件中擷取 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體`instance-id`的 。自動化是 中的工具 AWS Systems Manager。我們使用輸入轉換器將該資料傳遞給 `AWS-CreateImage` Runbook 目標，作為 `InstanceId` 輸入參數。當執行個體變更為 `stopped` 狀態時，將觸發規則。

如需使用輸入轉換器的詳細資訊，請參閱《Amazon EventBridge 使用者指南》中的教學課程：使用輸入轉換器以自訂傳送至事件目標的內容。

開始之前

確認您已將必要的 EventBridge 許可和信任政策新增至您的 Systems Manager Automation 服務角色。如需詳細資訊，請參閱《Amazon EventBridge 使用者指南》中的管理您的 EventBridge 資源之存取許可的概觀。

使用輸入轉換器搭配自動化

1. 前往 https://console.aws.amazon.com/events/ 開啟 Amazon EventBridge 主控台。
2. 在導覽窗格中，選擇規則。
3. 選擇建立規則。
4. 輸入規則的名稱和描述。

   在同一個區域和同一個事件匯流排上，規則不能與另一個規則同名。
5. 針對事件匯流排，選擇要與此規則建立關聯的事件匯流排。如果您希望此規則回應來自您自己的相符事件 AWS 帳戶，請選取預設值。當您帳戶中 AWS 服務 的 發出事件時，一律會前往您帳戶的預設事件匯流排。
6. 針對規則類型，選擇具有事件模式的規則。
7. 選擇下一步。
8. 在事件來源欄位中，選擇 AWS 事件或 EventBridge 合作夥伴事件。
9. 在 Event pattern (事件模式) 區段中，選擇 Event pattern form (事件模式表單)。
10. 在 Event source (事件來源) 欄位中，選擇 AWS services (服務)。
11. 在 AWS 服務中，選擇 EC2。
12. 在 Event Type (事件類型) 中，選擇 EC2 Instance State-change Notification (EC2 執行個體狀態變更通知)。
13. 針對 Specific state(s) (特定狀態)，選擇 stopped (已停止)。
14. 選擇 Next (下一步)。
15. 在目標類型欄位中，選擇 AWS 服務。
16. 針對 Select a target (選取目標)，請選擇 Systems Manager Automation。
17. 在 Document (文件) 中，選擇 AWS-CreateImage。
18. 在 Configure automation parameter(s) (設定自動化參數) 區段中，選擇 Input Transformer (輸入轉換器)。

19. 針對 Input path (輸入路徑)，輸入 **{"instance":"$.detail.instance-id"}**。
20. 針對 Template (範本)，輸入 **{"InstanceId":[<instance>]}**。
21. 針對 Execution role (執行角色)，選擇 Use existing role (使用現有角色)，然後選擇您的自動化服務角色。
22. 選擇 Next (下一步)。
23. (選用) 為規則輸入一或多個標籤。如需詳細資訊，請參閱《Amazon EventBridge 使用者指南》中的標記您的 Amazon EventBridge 資源。
24. 選擇 Next (下一步)。
25. 檢閱規則的詳細資訊，然後選擇建立規則。

了解 Systems Manager Automation 傳回的狀態

AWS Systems Manager 自動化會報告自動化動作或步驟在執行自動化時，以及整體自動化時所經歷之各種狀態的詳細狀態資訊。自動化是 中的工具 AWS Systems Manager。您可以使用下列方法來監控自動化狀態：

- 在 Systems Manager Automation 主控台監控 Execution status (執行狀態)。
- 使用您偏好的命令列工具。對於 AWS Command Line Interface (AWS CLI)，您可以使用 describe-automation-step-executions 或 get-automation-execution。對於 AWS Tools for Windows PowerShell，您可以使用 Get-SSMAutomationStepExecution 或 Get-SSMAutomationExecution。
- 設定 Amazon EventBridge 以回應動作或自動化狀態變更。

如需有關在自動化中處理逾時的詳細資訊，請參閱 處理 Runbook 中的逾時。

關於自動化狀態

除了整體自動化之外，自動化還會報告個別自動化動作的狀態詳細資訊。

整體自動化狀態可能會與個別動作或步驟所回報的狀態不同，如以下資料表所示。

動作的詳細狀態

Status	詳細資訊
待定	步驟尚未開始執行。如果您的自動化使用條件式動作，則在不符合條件的情形下執行步驟時，自動化完成後，步驟仍會維持在此狀態。如果在步

Status	詳細資訊
	驟執行之前取消自動化，則步驟也會保持在此狀態。
InProgress	步驟正在執行中。
等待	步驟正在等待輸入。
成功	步驟已成功完成。這是一個終端狀態。
TimedOut	步驟或核准未在指定的逾時期間之前完成。這是一個終端狀態。
取消	申請者取消後，步驟正在停用過程中。
已取消	在完成之前，申請者停止了步驟。這是一個終端狀態。
失敗	步驟未成功完成。這是一個終端狀態。
Exited	僅由 aws:loop 動作傳回。迴圈未完全完成。迴圈內的步驟使用 nextStep、onCancel 或 onFailure 屬性移至外部步驟。

自動化的詳細狀態

Status	詳細資訊
待定	自動化尚未開始執行。
InProgress	自動化正在執行中。
等待	自動化正在等待輸入。
成功	自動化已成功完成。這是一個終端狀態。
TimedOut	步驟或核准未在指定的逾時期間之前完成。這是一個終端狀態。

Status	詳細資訊
取消	申請者取消後，自動化正在停用過程中。
已取消	在完成之前，申請者停止了自動化。這是一個終端狀態。
失敗	自動化未成功完成。這是一個終端狀態。

故障診斷 Systems Manager Automation

使用下列資訊來協助您疑難排解 Automation 的問題 AWS Systems Manager ，Automation 是其中的工具 AWS Systems Manager。此主題包含根據自動化錯誤訊息解決問題的特定任務。

主題

- 常見的自動化錯誤
- 自動化執行無法啟動
- 執行已開始，但狀態為失敗
- 執行已開始，但發生逾時

常見的自動化錯誤

本節包含常見自動化錯誤的資訊。

VPC 未定義 400

根據預設，自動化執行 `AWS-UpdateLinuxAmi` Runbook 或 `AWS-UpdateWindowsAmi` Runbook 時，系統會在預設 VPC (172.30.0.0/16) 中建立暫時執行個體。如果刪除預設 VPC，您會收到以下錯誤：

```
VPC not defined 400
```

如要解決此問題，您必須指定 `SubnetId` 輸入參數的值。

自動化執行無法啟動

如果您尚未正確設定自動化的 AWS Identity and Access Management (IAM) 角色和政策，自動化可能會失敗，並出現存取遭拒錯誤或無效的擔任角色錯誤。

存取遭拒

以下範例說明自動化發生存取遭拒錯誤而無法啟動的情況。

存取 Systems Manager API 被拒

錯誤訊息：`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)`

- 可能原因 1：嘗試啟動自動化的使用者沒有叫用 `StartAutomationExecution` API 的許可。若要解決此問題，請將所需的 IAM 政策連接至用於啟動自動化的使用者。
- 可能原因 2：嘗試啟動自動化的使用者有叫用 `StartAutomationExecution` API 的許可，但沒有使用特定執行手冊叫用 API 的許可。若要解決此問題，請將所需的 IAM 政策連接至用於啟動自動化的使用者。

由於缺少 PassRole 許可而存取遭拒

錯誤訊息：`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)`

嘗試啟動自動化的使用者沒有所擔任角色的 PassRole 許可。若要解決此問題，請將 iam:PassRole 政策連接至嘗試啟動自動化的使用者角色。如需詳細資訊，請參閱任務 2：將 iam:PassRole 政策連接至自動化角色。

無效的擔任角色

在您執行 Automation 時，擔任角色會由 Runbook 提供或做為 Runbook 的參數值傳遞。如果未指定或正確設定擔任角色，可能會發生不同類型的錯誤。

格式錯誤的擔任角色

錯誤訊息：`The format of the supplied assume role ARN isn't valid.` 擔任角色格式不正確。若要解決此問題，請確認您的 Runbook 中指定了有效的擔任角色，或在執行自動化時將其做為執行時間參數。

擔任角色無法擔任

錯誤訊息：`The defined assume role is unable to be assumed. (Service: AWSSimpleSystemsManagement; Status Code: 400; Error Code: InvalidAutomationExecutionParametersException; Request ID: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx)`

- 可能原因 1：擔任角色不存在。若要解決此問題，請建立角色。如需詳細資訊，請參閱the section called “設定自動化”。以下主題說明建立此角色的特定詳細資訊：任務 1：建立自動化的服務角色。
- 可能原因 2：擔任角色與 Systems Manager 服務沒有信任關係。若要解決此問題，請建立信任關係。如需詳細資訊，請參閱《IAM 使用者指南》中的我無法擔任角色。

執行已開始，但狀態為失敗

動作特定的失敗

Runbook 包含步驟和依序執行的步驟。每個步驟會呼叫一個或多個 AWS 服務 API。API 會決定輸入、行為和步驟的輸出。有許多地方可能會發生錯誤導致步驟失敗。失敗訊息會指出錯誤發生的時間和位置。

若要在 Amazon Elastic Compute Cloud (Amazon EC2) 主控台查看失敗訊息，請選擇失敗步驟的 View Outputs (檢視輸出) 連結。若要從 查看失敗訊息 AWS CLI，請呼叫 `get-automation-execution`，並在失敗的 中尋找 `FailureMessage` 屬性`StepExecution`。

在以下範例中，與 `aws:runInstance` 動作關聯的一個步驟失敗。每個範例都會探討一種不同類型的錯誤。

缺少映像

錯誤訊息：`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` 動作收到不存在的 `ImageId` 輸入。若要解決此問題，請以正確的 AMI ID 更新 Runbook 或參數值。

擔任角色政策缺少足夠的許可

錯誤訊息：`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.`

擔任角色沒有足夠的許可，無法在 EC2 執行個體上呼叫 RunInstances API。若要解決此問題，請將 IAM 政策連接至擁有呼叫 RunInstances API 之許可的擔任角色。如需詳細資訊，請參閱 使用主控台建立用於自動化的服務角色。

未預期的狀態

錯誤訊息：`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.`

- 可能原因 1：執行個體或 Amazon EC2 服務發生問題。若要解決此問題，請登入執行個體或檢閱執行個體系統日誌，以了解執行個體開始關閉的原因。
- 可能原因 2：針對 `aws:runInstances` 動作指定的使用者資料指令碼有問題或語法不正確。確認使用者資料指令碼的語法。此外，確認使用者資料指令碼並未關閉執行個體或呼叫其他指令碼以關閉執行個體。

動作特定的故障參考

步驟失敗時，失敗訊息可能會指出發生失敗時正在呼叫什麼服務。下表列出各動作呼叫的服務。表格也提供了各服務的相關資訊連結。

動作	AWS 服務 此動作調用	此服務的相關資訊	故障診斷內容
`aws:runInstances`	Amazon EC2	Amazon EC2 使用者指南	EC2 執行個體疑難排解
`aws:changeInstanceState`	Amazon EC2	Amazon EC2 使用者指南	EC2 執行個體疑難排解

動作	AWS 服務 此動作調用	此服務的相關資訊	故障診斷內容
aws:runCommand	Systems Manager	AWS Systems Manager Run Command	故障診斷 Systems Manager 執行命令
aws:createImage	Amazon EC2	Amazon Machine Images	
aws:createStack	AWS CloudFormation	AWS CloudFormation 使用者指南	疑難排解 AWS CloudFormation
aws:deleteStack	AWS CloudFormation	AWS CloudFormation 使用者指南	疑難排解 AWS CloudFormation
aws:deleteImage	Amazon EC2	Amazon Machine Image	
aws:copyImage	Amazon EC2	Amazon Machine Images	
aws:createTag	Amazon EC2、Systems Manager	EC2 資源與標籤	
aws:invokeLambdaFunction	AWS Lambda	AWS Lambda 開發人員指南	Lambda 故障診斷

自動化服務內部錯誤

錯誤訊息：`Internal Server Error. Please refer to Automation Service Troubleshooting Guide for more diagnosis details.`

Automation 服務的一項問題使指定的 Runbook 無法正確執行。若要解決此問題，請聯絡 AWS 支援。請提供執行 ID 和客戶 ID (若有)。

執行已開始，但發生逾時

錯誤訊息：`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` 動作中的一項步驟發生逾時。如果步驟動作執行的時間比步驟中 `timeoutSeconds` 的指定值更長，可能就會發生此情況。若要解決此問題，請為 `aws:runInstances` 動作中的 `timeoutSeconds` 參數指定較長的值。如果這樣未解決問題，請調查步驟執行時間比預期更久的原因

AWS Systems Manager Change Calendar

Change Calendar當您指定的動作（例如，在 Systems Manager Automation Runbook 中）可能會在您的 中執行或不執行時， 中的工具 AWS Systems Manager可讓您設定日期和時間範圍 AWS 帳戶。在 Change Calendar 中，這些範圍稱為「事件」。當您建立了 Change Calendar 項目，也會建立類型 `ChangeCalendar` 的 Systems Manager 文件。在 Change Calendar 中，這些文件會以純文字格式儲存 iCalendar 2.0 資料。您新增到 Change Calendar 項目的事件將成為文件的一部份。若要開始使用 Change Calendar，請開啟 Systems Manager 主控台。在導覽窗格中，選擇 Change Calendar。

您可以在 Systems Manager 主控台中建立行事曆及其事件。您還可以匯入從支援的第三方行事曆供應商匯出的 iCalendar (`.ics`) 檔案，以將其事件新增到您的行事曆中。支援的供應商包括 Google 行事曆、Microsoft Outlook 和 iCloud 行事曆。

Change Calendar 項目可能為以下兩種類型之一：

`DEFAULT_OPEN`，或預設開啟

預設情況下 (但行事曆事件期間除外)，所有動作都可以執行。事件進行期間，`DEFAULT_OPEN` 行事曆的狀態為 `CLOSED`，且事件無法執行。

`DEFAULT_CLOSED`，或預設關閉。

預設情況下 (但行事曆事件期間除外)，所有動作都無法執行。事件進行期間，`DEFAULT_CLOSED` 行事曆的狀態為 `OPEN`，且動作能夠執行。

您可以選擇將所有已排程的 Automation 工作流程、維護時段和 State Manager 關聯自動新增至行事曆。您也可以從行事曆顯示中移除這些個別類型的任何項。

誰應該使用Change Calendar？

- AWS 執行下列動作類型的客戶：
  - 建立或執行 Automation 執行手冊。
  - 在 Change Manager 中建立變更請求。
  - 執行維護時段。
  - 在 State Manager 中建立關聯。

  自動化、Maintenance Windows、 和 Change ManagerState Manager都是工具 AWS Systems Manager。透過整合這些工具與 Change Calendar，您可以根據與每個工具相關聯的變更行事曆的目前狀態來允許或封鎖這些動作類型。
- 負責確保 Systems Manager 受管節點組態一致性、穩定性和運作的管理員。

Change Calendar 的優點

以下是 Change Calendar 的部分優點。

- 在套用變更前事先檢閱

  Change Calendar 項目可協助確保事先檢閱可能破壞環境的變更，再加以套用。
- 僅在適當時機套用變更

  Change Calendar 項目可協助確保環境在事件進行期間的穩定性。舉例來說，您可以建立 Change Calendar 項目，以便在您預期有高資源需求 (例如會議或公開行銷期間) 的時候禁止變更。當您預期有管理員支援有限 (例如假期或假日) 時，行事曆項目也可以禁止變更。您可以使用行事曆項目來允許變更，並將管理員支援有限，而不足以針對失敗工作或部署進行故障診斷的特定時段排除。
- 取得目前或即將變更的行事曆狀態

  您可以執行 Systems Manager `GetCalendarState` API 操作來獲得行事曆目前的狀態、指定時間的狀態，或是行事曆已排程變更的下一個狀態。
- EventBridge 支援

  Amazon EventBridge 規則中支援此 Systems Manager 工具做為事件類型。如需詳細資訊，請參閱 使用 Amazon EventBridge 監控 Systems Manager 及 參考：Systems Manager 的 Amazon EventBridge 事件模式和類型。

主題

- 設定 Change Calendar
- 使用 Change Calendar
- 正在新增 Change Calendar 依賴性到自動化 Runbook
- Change Calendar 疑難排解

設定 Change Calendar

在 中使用 工具之前Change Calendar，請先完成下列操作 AWS Systems Manager。

安裝最新的命令列工具

安裝最新的命令列工具，以取得與行事曆相關的狀態資訊。

需求	描述
AWS CLI	（選用）若要使用 AWS Command Line Interface (AWS CLI) 取得行事曆的狀態資訊，請在本機 AWS CLI 電腦上安裝最新版本的 。 如需如何安裝或升級 CLI 的詳細資訊，請參閱《AWS CLI使用者指南》中的安裝、更新和解除安裝AWS Command Line Interface 。
AWS Tools for PowerShell	(選用) 若要使用 Tools for PowerShell 取得與行事曆相關的狀態資訊，請在本機電腦上安裝最新版本的 Tools for PowerShell。 如需如何安裝或升級 Tools for PowerShell 的詳細資訊，請參閱《AWS Tools for PowerShell 使用者指南》中的安裝 AWS Tools for PowerShell。

設定許可

如果使用者、群組或角色受獲指派管理員許可，則您可以完整存取 Change Calendar。如果您沒有管理員許可，則管理員必須指派 AmazonSSMFullAccess 受管政策或指派提供所需許可的政策給使用者、群組或角色，藉此給予您許可。

使用 Change Calendar 需要以下許可。

Change Calendar 項目

若要建立、更新或刪除 Change Calendar 項目 (包含從項目新增或移除事件)，則連接至您的使用者、群組或角色的政策必須允許以下動作。

- ssm:CreateDocument
- ssm:DeleteDocument
- ssm:DescribeDocument
- ssm:DescribeDocumentPermission
- ssm:GetCalendar
- ssm:ListDocuments
- ssm:ModifyDocumentPermission
- ssm:PutCalendar
- ssm:UpdateDocument
- ssm:UpdateDocumentDefaultVersion

行事曆狀態

若要取得目前或即將變更的行事曆狀態，則連接至您的使用者、群組或角色的政策必須允許以下動作。

- ssm:GetCalendarState

操作事件

若要檢視操作事件 (例如維護時段、關聯及計劃的自動化)，連接至您的使用者、群組或角色的政策必須允許以下動作：

- ssm:DescribeMaintenanceWindows
- ssm:DescribeMaintenanceWindowExecution
- ssm:DescribeAutomationExecutions
- ssm:ListAssociations

> **Note**
>
> 他人擁有 (也就是他人建立) 的 Change Calendar 項目僅供唯讀，即便與您的帳戶分享也是如此。維護時段、State Manager 關聯和自動化不會共用。

使用 Change Calendar

您可以使用 AWS Systems Manager 主控台來新增、管理或刪除 中的項目Change Calendar，即 中的工具 AWS Systems Manager。您也可以匯入支援的第三方行事曆供應商的事件，方法是匯入您從來源行事曆匯出的 iCalendar (`.ics`) 檔案。此外，您可以使用 `GetCalendarState` API 操作或 `get-calendar-state` AWS Command Line Interface (AWS CLI) 命令，取得Change Calendar特定時間狀態的相關資訊。

主題

- 建立變更行事曆
- 在 Change Calendar 中建立和管理事件
- 從第三方行事曆中匯入及管理事件
- 更新變更行事曆
- 共用變更行事曆
- 刪除變更行事曆
- 取得變更行事曆的狀態

建立變更行事曆

當您在 Change Calendar的工具 中建立項目時 AWS Systems Manager，您要建立使用 `text` 格式的 Systems Manager 文件 (SSM 文件）。

若要建立變更行事曆

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Change Calendar。
3. 選擇 Create calendar (建立行事曆)。

   -或-

   如果先開啟 Change Calendar 首頁，則選擇 Create change calendar (建立變更行事曆)。
4. 在 Create instance (建立行事曆) 頁面上的 Calendar details (行事曆詳細資料) 中，輸入行事曆項目的名稱。行事曆名稱可包含字母、數字、句點、破折號和底線。請使用足夠特別的名稱，以便迅速辨識行事曆項目的用途。例如，**`support-off-hours`**。建立行事曆項目後就無法更新此名稱。

5. (選用) 在 Description (描述) 中，輸入行事曆項目的描述。
6. (選用) 在 Import calendar (匯入行事曆) 區域中，選擇 Choose file (選擇檔案)，以選取您從第三方行事曆供應商匯出的 iCalendar (`.ics`) 檔案。匯入檔案會將其事件新增至您的行事曆。

   支援的供應商包括 Google 行事曆、Microsoft Outlook 和 iCloud 行事曆。

   如需詳細資訊，請參閱從第三方行事曆供應商匯入事件。
7. 在 Calendar type (行事曆類型) 中，選擇以下其中一項。

   - Open by default (預設開啟) - 行事曆會開啟 (事件開始後可執行自動化動作)，接著在相關活動進行期間關閉。
   - Closed by default (預設關閉) - 行事曆會關閉 (事件開始後可執行自動化動作)，但會在相關活動進行期間開啟。
8. (選用) 在變更管理事件中，選取將變更管理事件新增至行事曆。此選項會在每月行事曆顯示畫面中顯示所有已排程的維護時段、State Manager 關聯、Automation 工作流程及 Change Manager 變更請求。

   > **Tip**
   >
   > 如果您稍後想要從行事曆顯示中永久移除這些事件類型，請編輯行事曆，取消選取此核取方塊，然後選擇儲存。
9. 選擇 Create calendar (建立行事曆)。

   建立行事曆項目後，Systems Manager 會在 Change Calendar 清單中顯示行事曆項目。這些資料欄會顯示行事曆版本和行事曆擁有者的 AWS 帳戶 號碼。除非您至少建立或匯入一個事件，否則行事曆將無法禁止或允許任何動作。如需建立事件的資訊，請參閱 建立 Change Calendar 事件。如需匯入事件的相關資訊，請參閱 從第三方行事曆供應商匯入事件。

在 Change Calendar 中建立和管理事件

在 中建立行事曆後 AWS Systems Manager Change Calendar，您可以建立、更新和刪除已開啟或已關閉行事曆中包含的事件。 Change Calendar 是 中的工具 AWS Systems Manager。

> **Tip**
>
> 作為直接在 Systems Manager 主控台中建立事件的替代方案，您可以從支援的第三方行事曆應用程式匯入 iCalendar (`.ics`) 檔案。如需相關資訊，請參閱 從第三方行事曆中匯入及管理事件。

主題

- 建立 Change Calendar 事件
- 更新 Change Calendar 事件
- 刪除Change Calendar事件

建立 Change Calendar 事件

當您將事件新增至 中的項目Change Calendar時，即 中的工具 AWS Systems Manager，您要指定一段時間，在此期間行事曆項目的預設動作會暫停。舉例來說，如果行事曆項目類型預設為關閉，則行事曆會在事件進行期間開放變更。(您也可以建立建議事件，其僅在行事曆上發揮提供資訊的作用。)

目前您只能使用主控台來建立 Change Calendar 事件。事件會新增到您在建立 Change Calendar 項目時所建立的 Change Calendar 文件。

若要建立 Change Calendar 事件

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Change Calendar。
3. 在行事曆清單中，請為您要新增事件的行事曆項目選擇名稱。
4. 在行事曆項目的詳細資料頁面上選擇 Create event (建立事件)。
5. 在 Create scheduled event (建立排程事件) 頁面的 Event details (事件詳細資料) 中，輸入事件的顯示名稱。事件名稱可包含字母、數字、句點、破折號和底線。請使用足夠特別的名稱，以便辨識事件的用途。例如，**`nighttime-hours`**。
6. 在 Description (描述) 中，輸入事件的描述。例如：**`The support team isn't available during these hours`**。
7. (選用) 如果希望此事件僅用作視覺通知或提醒，可選取 Advisory (建議) 核取方塊。建議事件不會在行事曆上發揮任何作用，僅為檢視行事曆的使用者提供資訊。

8. 在 Event start date (事件開始日期) 中，以 MM/DD/YYYY 格式輸入或選擇事件的開始日期，再以 hh:mm:ss (時分秒) 格式輸入指定日期的時間作為事件的開始時間。
9. 在 Event end date (事件結束日期) 中，以 MM/DD/YYYY 格式輸入或選擇事件的結束日期，再以 hh:mm:ss (時分秒) 格式輸入指定日期的時間作為事件的結束時間。
10. 在 Schedule time zone (排程時區) 中，選擇事件開始和結束時間適用的時區。您可以輸入城市的部分名稱，或是格林威治標準時間 (GMT) 以外的時區，以便更快找到時區。預設值是國際標準時間 (UTC)。
11. (選用) 如果要建立每天、每週或每月循環發生的事件，請開啟 Recurrence (循環)，然後指定循環的頻率和選用的結束日期。
12. 選擇 Create scheduled event (建立排程事件)。新的事件會新增到您的行事曆項目，且會顯示在行事曆項目詳細資料頁面的 Events (事件) 標籤上。

更新 Change Calendar 事件

使用下列程序在 主控台中 AWS Systems Manager 更新Change Calendar事件。 Change Calendar 是中的工具 AWS Systems Manager。

若要更新 Change Calendar 事件

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Change Calendar。
3. 在行事曆清單中，請為您要編輯事件的行事曆項目選擇名稱。
4. 在行事曆項目的詳細資料頁面上選擇 Events (事件)。
5. 在行事曆頁面中選擇您要編輯的事件。

> **Tip**
>
> 使用左上角的按鈕來往後或往前一年，或是往後或往前一個月。如果有需要，可以從右上角的清單選擇正確的時區來變更時區。

6. 在 Event details (事件詳細資訊) 中選擇 Edit (編輯)。

   若要變更事件名稱和描述，請新增或替換目前文字值。
7. 若要變更 Event start date (事件開始日期) 值，請選擇目前開始日期，然後從行事曆中選擇一個新日期。若要變更開始時間，請選擇目前開始時間，然後從清單中選擇一個新時間。

8. 若要變更 Event end date (事件結束日期) 值，請選擇目前日期，然後從行事曆中選擇一個新的結束日期。若要變更結束時間，請選擇目前結束時間，然後從清單中選擇一個新時間。
9. 若要變更 Schedule time zone (排程時區) 值，請選擇事件開始和結束時間適用的時區。您可以輸入城市的部分名稱，或是格林威治標準時間 (GMT) 以外的時區，以便更快找到時區。預設值是國際標準時間 (UTC)。
10. (選用) 如果希望此事件僅用作視覺通知或提醒，可選取 Advisory (建議) 核取方塊。建議事件不會在行事曆上發揮任何作用，僅為檢視行事曆的使用者提供資訊。
11. 選擇 Save (儲存)。您的變更會顯示在行事曆項目詳細資料頁面的 Events (事件) 標籤上。選擇您更新的事件來檢視變更。

刪除Change Calendar事件

您可以使用 AWS Systems Manager一次刪除 中的一個事件Change Calendar，這是 中的工具 AWS Management Console。

> **Tip**
>
> 如果您在建立行事曆時選取了將變更管理事件新增至行事曆，您可以執行下列動作：
>
> - 若要暫時隱藏行事曆顯示中的某個變更管理事件類型，請在每月預覽最上方針對該類型選擇 X。
> - 若要從行事曆顯示中永久移除這些類型，請編輯行事曆，取消選取將變更管理事件新增至行事曆核取方塊，然後選擇儲存。從行事曆顯示中移除類型並不會將其從您的帳戶中刪除。

若要刪除 Change Calendar 事件

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Change Calendar。
3. 在行事曆清單中，請選擇您要從中刪除事件的行事曆項目名稱。
4. 在行事曆項目的詳細資料頁面上選擇 Events (事件)。
5. 在行事曆頁面中選擇您要刪除的事件。

> **Tip**
>
> 使用左上角的按鈕來將行事曆往後或往前一年，或是往後或往前一個月。如果有需要，可以從右上角的清單選擇正確的時區來變更時區。

6. 在 Event details (事件詳細資料) 頁面上，選擇 Delete (刪除)。當系統提示您確認是否要刪除事件時，請選擇 Confirm (確認)。

從第三方行事曆中匯入及管理事件

除了直接在 AWS Systems Manager 主控台中建立事件外，您也可以從支援的第三方行事曆應用程式匯入 iCalendar (`.ics`) 檔案。您的行事曆可以同時包含匯入的事件和您在 中建立的事件Change Calendar，這是 中的工具 AWS Systems Manager。

開始之前

在您嘗試匯入行事曆檔案之前，請先檢閱以下要求與限制：

行事曆檔案格式

僅支援有效的 iCalendar 檔案 (`.ics`)。

支援的行事曆供應商

僅支援從下列第三方行事曆供應商匯出的`.ics`檔案：

- Google 行事曆 (匯出指示)
- Microsoft Outlook (匯出指示)
- iCloud 行事曆 (匯出指示)

檔案大小

您可以匯入任意數量的有效 `.ics` 檔案。不過，每個行事曆的所有匯入檔案大小總計不得超過 64KB。

> **Tip**
>
> 若要將 `.ics` 檔案的大小減至最小，請確定您只匯出行事曆項目的基本詳細資訊。必要時，請縮短要匯出的時間長度。

時區

除了行事曆名稱、行事曆供應商和至少一個事件之外，您匯出的 .ics 檔案也應指出行事曆的時區。如果沒有，或是識別時區時發生問題，系統會在匯入檔案後提示您指定時區。

重複事件限制

匯出的 .ics 檔案可以包含重複事件。不過，如果在來源行事曆中刪除重複事件的一個或多個事件，匯入就會失敗。

主題

- 從第三方行事曆供應商匯入事件
- 從第三方行事曆供應商更新所有事件
- 刪除從第三方行事曆匯入的所有事件

從第三方行事曆供應商匯入事件

使用以下程序可從支援的第三方行事曆應用程式匯入 iCalendar (.ics) 檔案。檔案中包含的事件會納入開啟或關閉行事曆的規則中。您可以將檔案匯入您使用 Change Calendar(中的工具 AWS Systems Manager) 建立的新行事曆，或匯入現有的行事曆。

匯入 .ics 檔案之後，您可以使用 Change Calendar 介面從其中移除個別事件。如需相關資訊，請參閱 刪除Change Calendar事件。您也可以透過刪除 .ics 檔案，來刪除來源行事曆中的所有事件。如需相關資訊，請參閱 刪除從第三方行事曆匯入的所有事件。

若要從第三方行事曆供應商匯入事件

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Change Calendar。
3. 若要從新的行事曆開始，請選擇 Create calendar (建立行事曆)。在 Import calendar (匯入行事曆) 區域中，選擇 Choose file (選擇檔案)。如需有關建立新行事曆的其他步驟的資訊，請參閱 建立變更行事曆。

   -或-

   若要將第三方事件匯入現有的行事曆，請選擇現有行事曆的名稱加以開啟。
4. 選擇 Actions, Edit (動作：編輯)，然後在 Import calendar (匯入行事曆) 區域中選擇 Choose file (選擇檔案)。

5. 導覽並選取本機電腦上匯出的 `.ics` 檔案。
6. 如果出現提示，對於 Select a time zone (選取時區)，選取要套用至行事曆的時區。
7. 選擇 Save (儲存)。

從第三方行事曆供應商更新所有事件

如果在您匯入其 iCalendar `.ics` 檔案之後，將多個事件新增至來源行事曆或從來源行事曆中移除事件，則您可以在 Change Calendar 中反映這些變更。首先，重新匯出來源行事曆，然後將新檔案匯入 Change Calendar，這是 中的工具 AWS Systems Manager。變更行事曆中的事件將會更新，以反映較新檔案的內容。

若要從第三方行事曆供應商更新所有事件

1. 在您的第三方行事曆中，新增或移除您想要反映在 Change Calendar 中的事件，然後再將行事曆重新匯出至新的 `.ics` 檔案。
2. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
3. 在導覽窗格中，選擇 Change Calendar。
4. 從行事曆清單中，從清單中選擇行事曆名稱。
5. 選擇選擇檔案，然後導覽並選取替代 `.ics` 檔案。
6. 若要回應有關覆寫現有檔案的通知，請選擇 Confirm (確認)。

刪除從第三方行事曆匯入的所有事件

如果您不再希望從第三方供應商匯入的任何事件包含在行事曆中，您可以刪除已匯入的 iCalendar `.ics` 檔案。

若要刪除從第三方行事曆匯入的所有事件

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Change Calendar。
3. 從行事曆清單中，從清單中選擇行事曆名稱。
4. 在 Import calendar (匯入行事曆) 區域中的 My imported calendars (我匯入的行事曆)，找出匯入行事曆的名稱，然後選擇卡上的 X。
5. 選擇 Save (儲存)。

更新變更行事曆

您可以更新變更行事曆的描述，但無法更新其名稱。您雖然可以變更行事曆的預設狀態，但是請注意，這會使行事曆相關事件進行期間的變更動作行為發生反轉。舉例來說，如果您將行事曆的狀態從 Open by default (預設開啟) 變更為 Closed by default (預設關閉)，那麼如果有建立相關事件的使用者不知道變更，就可能做出不必要的變更。

當您更新變更行事曆時，您正在編輯建立項目時建立Change Calendar的文件。 Change Calendar 是中的工具 AWS Systems Manager。

若要更新變更行事曆

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Change Calendar。
3. 在行事曆清單中，請選擇您要更新的行事曆名稱。
4. 在行事曆的詳細資訊頁面上，選擇 Actions, Edit (動作：編輯)。
5. 您可以在 Description (描述) 中變更描述文字。您無法編輯變更行事曆的名稱。
6. 如果要變更行事曆狀態，請在 Calendar type (行事曆類型) 中選擇其他值。請注意，這會使行事曆相關事件進行期間的變更動作行為發生反轉。變更行事曆類型之前，您應該和其他 Change Calendar 使用者確認變更行事曆類型並不會在他們所建立的活動進行時出現不必要的變更。
   - Open by default (預設開啟) - 行事曆會開啟 (事件開始後可執行自動化動作)，接著在相關活動進行期間關閉。
   - Closed by default (預設關閉) - 行事曆會關閉 (事件開始後可執行自動化動作)，但會在相關活動進行期間開啟。
7. 選擇 Save (儲存)。

   除非您至少新增一個事件，否則行事曆將無法禁止或允許任何動作。如需有關如何新增事件的資訊，請參閱建立 Change Calendar 事件。

共用變更行事曆

您可以使用 AWS 帳戶 AWS Systems Manager 主控台 AWS Systems Manager，在 Change Calendar 中與其他 共用行事曆。共用行事曆時，行事曆僅供共用帳戶內的使用者唯讀。維護時段、State Manager 關聯和自動化不會共用。

若要共用變更行事曆

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Change Calendar。
3. 在行事曆清單中，請選擇您要共用的行事曆名稱。
4. 在行事曆的詳細資訊頁面上，選擇 Share (共用) 索引標籤。
5. 選擇 Actions, Share (動作：共用)。
6. 在共用行事曆中，針對帳戶 ID，輸入有效 的 ID 號碼 AWS 帳戶，然後選擇共用。

   共用帳戶的使用者可以讀取變更行事曆，但無法進行變更。

刪除變更行事曆

您可以使用 Systems Manager 主控台或 AWS Command Line Interface (AWS CLI) AWS Systems Manager，在 中刪除 中的行事曆Change Calendar。刪除變更行事曆也會刪除所有相關事件。

若要刪除變更行事曆

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Change Calendar。
3. 在行事曆清單中，請選擇您要刪除的行事曆名稱。
4. 在行事曆的詳細資訊頁面上，選擇 Actions, Delete (動作：刪除)。當系統提示您確認是否要刪除行事曆時，請選擇 Delete (刪除)。

取得變更行事曆的狀態

您可以在 中的Change Calendar工具，取得行事曆的整體狀態或行事曆在特定時間的狀態 AWS Systems Manager。您也可以取得下次行事曆狀態從 `OPEN` 變更為 `CLOSED` 的時間，反之亦然。

> **Note**
>
> 如需Change Calendar與 Amazon EventBridge 整合以自動監控行事曆狀態變更的相關資訊，請參閱 the section called “Change Calendar 與 Amazon EventBridge 整合”。EventBridge 整

合會在行事曆狀態轉換時提供事件驅動的通知，補充 GetCalendarState API 動作的輪詢型方法。

您可以使用 GetCalendarState API 操作來執行這項任務。本節中的程序使用 AWS Command Line Interface (AWS CLI)。

若要取得變更行事曆的狀態

- 請執行以下命令，來取得一或多個行事曆在特定時間的狀態。--calendar-names 參數是必要項目，--at-time 則為選用參數。將每個########取代為您自己的資訊。

  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"
  ```

  以下是範例。

  ```
  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"
  ```

  以下是範例。

  ```
  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"
```

該命令會傳回相關資訊，如以下所示。

```
{
    "State": "OPEN",
    "AtTime": "2020-07-30T16:18:18Z",
    "NextTransitionTime": "2020-07-31T00:00:00Z"
}
```

結果會顯示您帳戶擁有 (或與您帳戶共用) 的行事曆狀態 (行事曆類型為 DEFAULT_OPEN 或 DEFAULT_CLOSED，並且以 --at-time 的值指定時間)，以及下一次轉換的時間。如果您不新增 --at-time 參數，就會使用目前的時間。

> **Note**
>
> 如果您在請求中指定多個行事曆，此命令僅會在請求中的所有行事曆均開啟的情況下傳回 OPEN 的狀態。如果請求中的一或多個行事曆已關閉，則傳回的狀態為 CLOSED。

正在新增 Change Calendar 依賴性到自動化 Runbook

若要讓自動化動作遵守 Change Calendar中的工具 AWS Systems Manager，請在使用 aws:assertAwsResourceProperty動作的 Automation Runbook 中新增步驟。設定要執行 GetCalendarState 的動作，來驗證指定的行事曆項目是否處於您希望的狀態 (OPEN 或 CLOSED)。只有在行事曆狀態為 OPEN 時，自動化 Runbook 才可以繼續進行下一個步驟。下列內容是 YAML 的範例摘錄，其中的自動化 Runbook 無法繼續進行下一個步驟 (LaunchInstance)，除非行事曆狀態符合 DesiredValues 中指定的 OPEN 行事曆狀態才可以。

以下是範例。

```
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
...
```

Change Calendar 疑難排解

使用下列資訊來協助您疑難排解 中的Change Calendar工具 的問題 AWS Systems Manager。

主題

- 'Calendar import failed' ('行事曆匯入失敗') 錯誤

'Calendar import failed' ('行事曆匯入失敗') 錯誤

問題：當匯入 iCalendar (`.ics`) 檔案時，系統會報告行事曆匯入失敗。

- 解決方案 1 – 確定您正在匯入從支援的第三方行事曆供應商匯出的檔案，其中包括下列各項：
  - Google 行事曆 (匯出指示)
  - Microsoft Outlook (匯出指示)
  - iCloud 行事曆 (匯出指示)
- 解決方案 2 – 如果您的來源行事曆包含任何重複事件，請確定未取消或刪除任何個別事件。目前，Change Calendar 不支援匯入具有個別取消的重複事件。若要解決這個問題，請從來源行事曆中移除重複事件、重新匯出行事曆並將它重新匯入 Change Calendar，然後使用 Change Calendar 介面新增重複事件。如需相關資訊，請參閱 建立 Change Calendar 事件。
- 解決方案 3 – 確定您的來源行事曆包含至少一個事件。上傳不包含事件的 `.ics` 檔案不會成功。
- 解決方案 4 – 如果系統報告，因為 `.ics` 太大而匯入失敗，則請確定您只匯出行事曆項目的基本詳細資訊。必要時，請縮短您匯出的時間段長度。
- 解決方案 5 – 如果當您嘗試從 Events (事件) 標籤匯入時，Change Calendar 無法判斷匯出的行事曆的時區，您可能會收到此訊息："Calendar import failed. Change Calendar couldn't locate a valid time zone。You can import the calendar from the Edit menu." (行事曆匯入失敗。Change Calendar

無法找到有效時區。您可以從編輯選單中匯入行事曆。) 在此情況下，請依次選擇 Actions, Edit (動作、編輯)，然後嘗試從 Edit calendar (編輯行事曆) 頁面匯入檔案。

- 解決方案 6 – 請勿在匯入之前編輯 `.ics` 檔案。嘗試修改檔案內容可能會損毀行事曆資料。如果您在嘗試匯入之前已修改檔案，請再次從來源行事曆匯出行事曆，然後重新嘗試上傳。

AWS Systems Manager Change Manager

Change Manager是 中的工具 AWS Systems Manager，是一種企業變更管理架構，用於請求、核准、實作和報告應用程式組態和基礎設施的操作變更。從單一委派管理員帳戶，如果您使用 AWS Organizations，您可以管理跨多個 AWS 帳戶 和多個 的變更 AWS 區域。或使用本機帳戶，您可以管理單一 AWS 帳戶的變更。使用 Change Manager 管理 AWS 資源和內部部署資源的變更。若要開始使用 Change Manager，請開啟 Systems Manager 主控台。在導覽窗格中，選擇 Change Manager。

透過 Change Manager，您可以使用預先核准的變更範本，協助自動化資源的變更程序，並協助避免在進行操作變更時的意外結果。每個變更範本可指定下列項目：

- 建立變更請求時，可供使用者選擇的一個或多個 Automation Runbook。對資源所做的變更會在 Automation Runbook 中進行定義。您可以在您建立的變更範本中包含自訂 Runbook 或 AWS 受管 Runbook。當使用者建立變更請求時，他們可以選擇要在請求中包含哪一個可用 Runbook。此外，您可以建立變更範本，讓提出請求的使用者指定變更請求中的任何 Runbook。
- 帳戶中使用者，必須檢閱使用該變更範本提出的變更請求。
- Amazon Simple Notification Service (Amazon SNS) 主題，用於通知指派的核准者變更請求已準備好進行審核。
- Amazon CloudWatch 警示，用於監控 Runbook 工作流程。
- Amazon SNS 主題，用於針對使用變更範本建立的變更請求傳送狀態變更通知。
- 要套用至變更範本的標籤，以便用於分類和篩選變更範本。
- 是否可以在沒有核准步驟的情況下執行從變更範本建立的變更請求 (自動核准的請求)。

透過與 Systems Manager 中的另一個工具 整合Change Calendar， Change Manager也可協助您安全地實作變更，同時避免排程與重要商業事件發生衝突。 Change Manager整合 AWS Organizations ，並 AWS IAM Identity Center 協助您使用現有的身分管理系統，從單一帳戶管理整個組織的變更。您可以從 Change Manager 監控變更進度和稽核整個組織的操作變更，提供更好的可見性和責任。

Change Manager 補充了您的持續整合 (CI) 實務和持續交付 (CD) 方法論的安全控制。Change Manager 不適用於作為自動發行程序 (例如 CI/CD 管道) 的所做的變更，除非有例外狀況或需要核准。

Change Manager 的運作方式

識別出標準或緊急操作變更的需求時，組織中的某個人會根據為您的組織或帳戶中建立的其中一個變更範本來建立變更請求。

如果請求的變更需要手動核准，Change Manager 會透過 Amazon SNS 通知來通知指定的核准者變更請求已準備好進行審核。您可以在變更範本中為變更請求指定核准者，或讓使用者指定變更請求本身的核准者。您可以為不同的範本指派不同的檢閱者。例如，指派一個使用者、使用者群組或 AWS Identity and Access Management (IAM) 角色，其必須核准受管節點的變更請求，以及針對資料庫變更核准其他使用者、群組或 IAM 角色。如果變更範本允許自動核准，且請求者的使用者政策未禁止此操作，則使用者也可以選擇針對其請求執行 Automation 執行手冊，而不需要檢閱步驟 (除變更凍結事件外)。

對於每個變更範本，您最多可以新增五個核准者層級。例如，您可能需要技術檢閱者先核准從變更範本中建立的變更請求，然後再請求一位或多位經理的第二層核准。

Change Manager 已與 AWS Systems Manager Change Calendar 整合。核准請求的變更後，系統會先決定請求是否與其他排定的企業活動衝突。如果偵測到衝突，Change Manager 可以封鎖變更，或在啟動 Runbook 工作流程之前需要其他核准。例如，您可能只允許在上班時間進行變更，以確保團隊可以管理任何未預期的問題。對於請求在這些時間以外執行的任何變更，您可以變更凍結核准者的形式要求更高層級的管理核准。如需緊急變更，Change Manager 可以跳過檢查 Change Calendar 的步驟，以獲得核准變更請求之後的衝突或封鎖事件。

當需要實作核准的變更時，Change Manager 會執行關聯的變更請求中指定的自動化 Runbook。執行 Runbook 工作流程時，只允許在核准的變更請求中定義的操作。這種方法有助於避免實作變更時的意外的結果。

除了限制執行 Runbook 工作流程時可以進行的變更之外，Change Manager 也可以幫助您控制並發和錯誤閾值。您可以選擇 Runbook 工作流程一次可以執行多少資源、一次可以在多少個帳戶中執行變更戶，以及在停止程序之前允許失敗和復原 (如果 Runbook 包含回復指令碼) 多少次。您也可以使用 CloudWatch 警示監控變更的進度。

Runbook 工作流程完成後，您可以檢閱所做變更的詳細資訊。這些詳細資訊包括變更請求的原因、使用的變更範本、請求和核准變更的人員，以及如何實作變更。

詳細資訊

- 在 AWS 新聞部落格上介紹 AWS Systems ManagerChange Manager

Change Manager 對我的營運有何好處？

Change Manager 的優點包括：

- 降低服務中斷和停機的風險

  Change Manager 可以確保在執行 Runbook 工作流程時，只會實作核准的變更，進而更安全地進行操作變更。您可以封鎖未計劃和未檢閱的變更。Change Manager 可協助您避免因人為錯誤造成的意外結果類型，而這些結果需要花費昂貴的時間進行研究和回溯。

- 取得變更記錄的詳細稽核與報告

  Change Manager 提供責任制，以一致的方式報告和稽核整個組織所做的變更、變更的意圖，以及核准和實作變更的人員的詳細資訊。

- 避免排程衝突或違規

  Change Manager 可以根據您組織的作用中變更行事曆，偵測排程衝突，例如假日事件或新產品啟動。您可以允許 Runbook 工作流程僅在上班時間執行，或只允許其他核准。

- 根據不斷變化的業務調整變更需求

  在不同的業務期間，您可以實作不同的變更管理需求。例如，在月底報告、稅季或其他重要業務期間，您可以封鎖變更，或需要主管層級的核准才能進行可能造成不必要的營運風險的變更。

- 集中管理不同帳戶的變更

  透過與 Organizations 整合，Change Manager 可讓您從單一委派系統管理員帳戶管理所有組織單位 (OU) 的變更。您可以開啟 Change Manager 用於整個組織，或僅搭配部分 OU 使用。

誰應該使用Change Manager？

Change Manager 適用於下列 AWS 客戶和組織：

- 任何想要改善對其雲端或內部部署環境進行操作變更的安全性和管控 AWS 的客戶。
- 想要提高團隊之間的協作和可見性的 Organizations，可藉由避免停機來改善應用程式可用性，以及降低手動和重複工作的相關風險。
- Organizations 必須遵守變更管理的最佳實務。
- 需要對其應用程式組態和基礎設施所做變更的完整可稽核歷程記錄的客戶。

Change Manager有哪些主要功能？

Change Manager 的主要功能如下所示：

- 變更管理最佳實務的整合式支援

  利用 Change Manager，您可以將選取變更管理最佳實務套用至您的營運中。您可以選擇開啟以下選項：

  - 檢查 Change Calendar 以查看事件目前是否受到限制，因此只會在開啟的行事曆期間進行變更。
  - 允許在受限事件期間進行變更，並獲得變更凍結核准者的額外的核准。
  - 需要為所有變更範本指定 CloudWatch 警示。
  - 需要在您帳戶中建立的所有變更範本都經過審核和核准，才能用來建立變更請求。
- 已關閉行事曆期間和緊急變更請求的不同核准路徑

  您可以允許選項檢查 Change Calendar 以獲悉受限制的事件，並封鎖核准的變更請求，直到事件完成為止。但是，您還可以指定第二組核准者，即變更凍結核准者，即使行事曆已關閉，他們也可以允許執行變更。您也可以建立緊急變更範本。從緊急變更範本建立的變更請求仍需要定期核准，但不受行事曆限制，也不需要變更凍結核准。
- 控制啟動 Runbook 工作流程的方式和時間

  Runbook 工作流程可以根據排程啟動，或在核准完成後立即啟動 (受行事曆限制規則的約束)。
- 內建通知支援

  指定組織中應檢閱和核准變更範本和變更請求的人員。將 Amazon SNS 主題指派給變更範本，以將使用該變更範本建立的變更請求狀態變更通知傳送給該主題的訂閱者。
- 與 整合 AWS Systems Manager Change Calendar

  Change Manager 允許管理員限制在指定時間段內的排程變更。例如，您可以建立政策，只允許在營業時間進行變更，以確保團隊能夠處理任何問題。您也可以在重要的業務事件期間限制變更。例如，零售業務可能會限制大型銷售活動期間的變更。您也可以要求在限制期間進行額外的核准。
- 與 AWS IAM Identity Center 和 Active Directory 整合支援

  藉助 IAM Identity Center 整合，您組織的成員可以存取 AWS 帳戶 ，並根據一般使用者身分使用 Systems Manager 來管理其資源。使用 IAM Identity Center，您可以將使用者存取指派給 AWS的帳戶。

  與 Active Directory 的整合可讓您將 Active Directory 帳戶中的使用者指派為核准者，以變更為您的 Change Manager 營運建立的範本。

- 與 Amazon CloudWatch 警示的整合

  Change Manager 已與 CloudWatch 警示整合。Change Manager 會在 Runbook 工作流程期間接聽 CloudWatch 警示，並採取針對警示定義的任何動作 (包括傳送通知)。

- 與 AWS CloudTrail Lake 整合

  透過在 AWS CloudTrail Lake 中建立事件資料存放區，您可以檢視在您的帳戶或組織中執行的變更請求所做變更的可稽核資訊。儲存的事件資訊包括下列詳細資訊：

  - 執行的 API 動作
  - 針對這些動作包含的請求參數
  - 執行動作的使用者
  - 在此過程中更新的資源

- 與 整合 AWS Organizations

  使用 Organizations 提供的跨帳戶功能，您可以使用委派管理員帳戶來管理您組織的 OU 中的 Change Manager 營運。在您的 Organizations 管理帳戶中，您可以指定哪個帳戶要作為委派系統管理員帳戶。您也可以控制在您的哪些 OU 中可以使用 Change Manager。

使用 Change Manager 需要付費嗎？

是。Change Manager 均按依使用量付費的方式定價。您僅需按實際用量付費。如需詳細資訊，請參閱AWS Systems Manager 定價。

什麼是 Change Manager 的主要元件？

您用來管理組織或帳戶中變更流程的 Change Manager 元件包括下列各項：

委派管理員帳戶

如果您在整個組織中使用 Change Manager，則會使用委派管理員帳戶。這是指定為管理跨 Systems Manager 操作活動的帳戶的 AWS 帳戶 ，包括 Change Manager。委派管理員帳戶會管理整個組織的變更活動。當您設定組織以配合 Change Manager 使用時，可以指定哪些帳號擔任此角色。委派管理員帳戶必須是指派給組織單位 (OU) 的唯一成員。如果您 AWS 帳戶 只Change Manager搭配單一 使用 ，則不需要委派的管理員帳戶。

> **Important**
>
> 如果您在整個組織中使用 Change Manager，我們建議始終從委派管理員帳戶進行變更。雖然您可以從組織中的其他帳戶進行變更，但這些變更將不會在受委派管理員帳戶中報告，也不可在其中檢視。

變更範本

變更範本是 Change Manager 中的組態設定的集合，可定義必要核准、可用 Runbook 以及變更請求的通知選項等項目。

您可以要求組織或帳戶中的使用者建立的變更範本必須經過核准程序，才能使用。

Change Manager 支援兩種變更範本類型。對於根據緊急變更範本的已核准變更請求，即使在 Change Calendar 中有封鎖事件，也可以進行請求的變更。對於根據標準變更範本的已核准變更請求，如果 Change Calendar 中有封鎖事件，則無法進行請求的變更，除非獲得委派的變更凍結事件核准者的額外核准。

變更請求

變更請求是在 中執行 Automation Runbook Change Manager的請求，該自動化 Runbook 會更新您或 AWS 內部部署環境中的一或多個資源。變更請求是使用變更範本建立的。

當您建立變更請求時，組織或帳戶中的一個或多個核准者必須檢閱並核准該請求。如果沒有所需的核准，則不允許執行會套用您請求的變更的 Runbook 工作流程。

在系統中，變更請求是 OpsItem 中的一種類型 AWS Systems Manager OpsCenter。然而，類型 `/aws/changerequest` 的 OpsItems 不會在 OpsCenter 中顯示。作為 OpsItems，變更要求會受到與其他類型的 OpsItems 相同的強制性配額。

此外，若要以程式設計方式建立變更要求，您不要呼叫 `CreateOpsItem` API 操作。相反，您可以使用 `StartChangeRequestExecution` API 操作。不過，但相較於立即執行，變更請求必須獲得核准並且在 Change Calendar 中必須沒有任何會阻止工作流程執行的封鎖事件。當獲得核准且未封鎖行事曆 (或已授與略過封鎖行事曆事件的許可) 時，即能完成 `StartChangeRequestExecution` 動作。

Runbook 工作流程

Runbook 工作流程是指對雲端或內部部署環境中目標資源進行請求變更的程序。每個變更請求會指定單一的自動化 Runbook 來進行請求的變更。在授予所有必要的核准之後且 Change Calendar 中沒有封

鎖事件，就會發生 Runbook 工作流程。如果變更已排定在特定的日期和時間，則 Runbook 工作流程會在排定之前開始，即使已收到所有核准且未封鎖行事曆。

主題

- 設定 Change Manager
- 使用 Change Manager
- 稽核和記錄 Change Manager 活動
- Change Manager 疑難排解

設定 Change Manager

您可以使用 中的Change Manager工具 AWS Systems Manager來管理整個組織、 中設定的變更 AWS Organizations，或單一組織的變更 AWS 帳戶。

如果您搭配組織使用 Change Manager，則可以主題 設定適用於組織的 Change Manager (管理帳戶) 開始，然後繼續前往 設定 Change Manager 選項和最佳實務。

如果您搭配單一帳戶使用 Change Manager，請直接前往 設定 Change Manager 選項和最佳實務。

> **Note**
>
> 如果您開始時是搭配單一帳戶使用 Change Manager，但該帳戶稍後會新增至 Change Manager 允許的組織單位中，則會忽略您的單一帳戶設定。

主題

- 設定適用於組織的 Change Manager (管理帳戶)
- 設定 Change Manager 選項和最佳實務
- 設定 Change Manager 的角色和許可
- 控制對自動核准 Runbook 工作流程的存取

設定適用於組織的 Change Manager (管理帳戶)

如果您使用 中的工具 Change Manager搭配在 中設定的組織 AWS Systems Manager，則此主題中的任務適用 AWS Organizations。如果您Change Manager只想要將 與單一 搭配使用 AWS 帳戶，請跳至主題 設定 Change Manager 選項和最佳實務。

在做為 Organizations 管理帳戶的 中 AWS 帳戶 ，執行本節中的任務。如需管理帳戶和其他 Organizations 概念的相關資訊，請參閱 AWS Organizations 術語與概念。

如果您需要開啟 Organizations 並將帳戶指定為管理帳戶，然後再繼續進行，請參閱《AWS Organizations 使用者指南》中的建立和管理組織。

> **Note**
>
> 此設定程序無法在下列情況下執行 AWS 區域：
>
> - 歐洲 (米蘭) (eu-south-1)
> - 中東 (巴林) (me-south-1)
> - 非洲 (開普敦) (af-south-1)
> - 亞太區域 (香港) (ap-east-1)
>
> 確保您在管理帳戶中的不同區域工作，以執行此程序。

在設定程序中，您會在 中執行下列主要任務Quick Setup，這是 中的工具 AWS Systems Manager。

- 任務 1：註冊貴組織的委派管理員帳戶

  使用 Change Manager 執行的變更相關任務可以您其中一個成員帳戶中進行管理，而您可將該帳戶指定為委派管理員帳戶。您註冊的 Change Manager 的委派管理員帳戶會成為您所有 Systems Manager 操作的委派管理員帳戶。（您可能已委派其他管理員帳戶） AWS 服務。您的 Change Manager 委派管理員帳戶 (與管理帳戶不同) 可管理整個組織的變更活動，包括變更範本、變更請求和每個核准。在委派管理員帳戶中，您也可以為 Change Manager 營運指定其他組態選項。

  > **Important**
  >
  > 委派管理員帳戶必須是在 Organizations 中指派給組織單位 (OU) 的唯一成員。

- 任務 2：定義並指定變更申請者角色或自訂任務函數的 Runbook 存取政策，且您要將其用於您的 Change Manager 操作

  若要在 中建立變更請求Change Manager，您的成員帳戶中的使用者必須獲得 AWS Identity and Access Management (IAM) 許可，允許他們僅存取您選擇提供給他們的自動化執行手冊和變更範本。

> **Note**
>
> 當使用者建立變更請求時，他們會先選取變更範本。此變更範本可能會提供多個 Runbook，但使用者只能為每個變更要求選取一個 Runbook。變更範本也可以設定為允許使用者在其請求中包含任何可用的 Runbook。

若要授予所需的許可，Change Manager 會使用亦為 IAM 所用的任務職能的概念。然而，與 IAM 中的任務職能的AWS 受管政策不同，您可以指定您的 Change Manager 任務職能的名稱以及這些任務職能的 IAM 許可。

當您設定任務職能時，建議您建立自訂政策，並僅提供執行變更管理任務所需的權限。例如，您可能根據您定義的任務職能指定許可，以將使用者限制至特定的執行手冊組。

例如，您可以建立名為 `DBAdmin` 的任務職能。對於此任務職能，您僅可授予與 Amazon DynamoDB 資料庫相關的 Runbook 所需的許可，例如 `AWS-CreateDynamoDbBackup` 和 `AWSConfigRemediation-DeleteDynamoDbTable`。

作為另一個範例，您可能只想授予某些使用者使用與 Amazon Simple Storage Service (Amazon S3) 儲存貯體相關的 Runbook 所需的許可，例如 `AWS-ConfigureS3BucketLogging` 和 `AWSConfigRemediation-ConfigureS3BucketPublicAccessBlock`。

Change Manager 的 Quick Setup 中的組態程序也會提供一組完整的 Systems Manager 系統管理許可，以供您套用至您建立的管理角色。

您部署的每個 Change Manager Quick Setup 組態會在您的委派管理員帳戶中建立具有許可的任務職能，以在您選取的組織單位中執行 Change Manager 範本和 Automation Runbook。您最多可以為 Change Manager 建立 15 個 Quick Setup 組態。

- 任務 3：選擇組織中要搭配 Change Manager 使用的成員帳戶

  您可以在 Organizations 設定的所有組織單位中以及其營運的所有 AWS 區域 中，搭配所有成員帳戶使用 Change Manager。如果您願意的話，您可以僅搭配部分組織單位使用 Change Manager。

> **Important**
>
> 我們強烈建議您在開始此程序之前，先通讀其步驟，以了解您的組態選擇以及授予的許可。特別是規劃您要建立的自訂任務職能，以及您指派給每個任務職能的許可。這可確保當您稍後將

您建立的任務職能政策連接到個別使用者、使用者群組或 IAM 角色時，只會授予您想要讓他們擁有的許可。
最佳實務是，首先使用管理員的登入來設定委派 AWS 帳戶 管理員帳戶。然後在建立變更範本並識別每個範本使用的 Runbook 之後，設定任務職能及其許可。

若要設定與組織搭配使用的 Change Manager，請在 Systems Manager 主控台的 Quick Setup 區域中執行以下任務。

您可以針對您要為組織建立的每個任務職能重複此任務。您建立的每個任務職能擁有針對不同組織單位的許可。

若要在 Organizations 管理帳戶中設定 Change Manager 的組織

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Quick Setup。
3. 在 Change Manager 卡上，選擇 Create (建立)。
4. 對於 Delegated administrator account (委派管理員帳戶)，輸入您要用來管理變更範本、變更請求和 Change Manager 中的 Runbook 工作流程的 AWS 帳戶 的 ID。

   如果您先前已為 Systems Manager 指定委派管理員帳戶，其 ID 已在此欄位中報告。

   Important

   委派管理員帳戶必須是在 Organizations 中指派給組織單位 (OU) 的唯一成員。
   如果您註冊的委派管理員帳戶稍後會從該角色取消註冊，系統會同時移除其管理 Systems Manager 操作的許可。請注意，您需要返回 Quick Setup、指定不同的委派管理員帳戶，然後再次指定所有任務職能和許可。
   如果您在整個組織中使用 Change Manager，我們建議始終從委派管理員帳戶進行變更。雖然您可以從組織中的其他帳戶進行變更，但這些變更將不會在受委派管理員帳戶中報告，也不可在其中檢視。
5. 在 Permissions to request and make changes (請求和進行變更的許可) 區段中，執行下列動作。

> **Note**
>
> 您建立的每個部署組態只會針對一個任務職能提供許可政策。您可以稍後返回 Quick Setup，以便在您建立要用於操作的變更範本時，建立更多任務職能。

若要建立管理員角色 – 對於對所有 AWS 動作具有 IAM 許可的管理員任務職能 動作，請執行下列動作。

> **Important**
>
> 授予使用者完整的管理許可務必謹慎進行，且只有當他們的角色需要完整的 Systems Manager 存取權限時進行。如需 Systems Manager 存取之安全考量的重要資訊，請參閱 適用於 AWS Systems Manager 的 Identity and Access Management 和 Systems Manager 的安全最佳實務。

1. 對於 Job function (任務職能)，輸入名稱以識別此角色及其許可，例如 **`My AWS Admin`**。
2. 對於 Role and permissions option (角色和許可選項)，選擇 Administrator permissions (管理員許可)。

若要建立其他任務職能 – 若要建立非管理角色，請執行下列動作：

1. 對於 Job function (任務職能)，輸入名稱以識別此角色及建議其許可。您選擇的名稱應代表您將提供許可的 Runbook 範圍，例如 `DBAdmin` 或 `S3Admin`。
2. 對於 Role and permissions option (角色和許可選項)，選擇 Custom permissions (自訂許可)。
3. 在 Permissions policy editor (許可政策編輯器) 中，以 JSON 格式輸入 IAM 許可，進而授與此任務職能。

> **Tip**
>
> 我們建議您使用 IAM 政策編輯器來建構政策，然後將政策 JSON 貼到 Permissions policy (許可政策) 欄位。

範例政策：DynamoDB 資料庫管理

例如，您可以從政策內容開始，提供使用任務職能需要存取的 Systems Manager 文件 (SSM 文件) 的許可。以下是在美國東部（俄亥俄）區域 () AWS 帳戶 123456789012中，授予與 DynamoDB 資料庫相關的所有 AWS 受管 Automation Runbook 的存取權，以及在範本中建立的兩個變更範本的範例政策內容us-east-2。

該政策也包含 StartChangeRequestExecution 操作的許可，這是在 Change Calendar 中建立變更請求所必需的。

> **Note**
>
> 此範例並不全面。使用資料庫和節點等其他 AWS 資源時，可能需要其他許可。

```
{
    "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:123456789012:document/MyFirstDBChangeTemplate",
                "arn:aws:ssm:region:123456789012:document/MySecondDBChangeTemplate"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "ssm:ListDocuments",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ssm:StartChangeRequestExecution",
            "Resource": "arn:aws:ssm:region:123456789012:automation-definition/*:*"
        }
    ]
}
```

如需 IAM 政策的詳細資訊，請參閱《IAM 使用者指南》中的 AWS 資源的存取管理和建立 IAM 政策。

6. 在 Targets (目標) 區段中，選擇要將您建立之任務職能的許可授予整個組織，還是只授予部分組織單位。

   如果您選擇 Entire organization (整個組織)，請繼續步驟 9。

   如果選擇 Custom (自訂)，請繼續步驟 8。

7. 在 Target OUs (目標 OU) 區段中，選取要搭配 Change Manager 使用之組織單位的核取方塊。

8. 選擇 Create (建立)。

系統完成為您組織設定 Change Manager 後，它會顯示部署的摘要。此摘要資訊包含為您設定之任務職能建立的角色名稱。例如：`AWS-QuickSetup-SSMChangeMgr-DBAdminInvocationRole`。

> **Note**
>
> Quick Setup 使用 AWS CloudFormation StackSets 部署您的組態。您還可以在 AWS CloudFormation 主控台中檢視有關已完成的部署組態的資訊。如需有關 StackSets 的資訊，請參閱AWS CloudFormation 《 使用者指南》中的使用 AWS CloudFormation StackSets。

您的下一個步驟是設定其他 Change Manager 選項。您可以使用委派管理員帳戶或組織單位中允許搭配 Change Manager 使用的任何帳戶來完成此任務。您可以設定選項，例如選擇使用者身分識別管理選項、指定哪些使用者可以檢閱和核准或拒絕變更範本和變更請求，以及選擇允許您組織的最佳實務選項。如需相關資訊，請參閱 設定 Change Manager 選項和最佳實務。

設定 Change Manager 選項和最佳實務

無論您是在 、 中Change Manager、跨組織或單一 中使用 工具 AWS Systems Manager，都必須執行本節中的任務 AWS 帳戶。

如果您使用的適用於組織的 Change Manager，您可以在委派管理員帳戶中或組織單位中允許搭配 Change Manager 使用的任何帳戶中執行下列任務。

主題

- 任務 1：設定 Change Manager 使用者身分識別管理和範本檢閱者
- 任務 2：設定 Change Manager 變更凍結事件核准者和最佳實務
- 為 Change Manager 通知設定 Amazon SNS 主題

任務 1：設定 Change Manager 使用者身分識別管理和範本檢閱者

在您第一次存取 Change Manager 時在此程序中執行任務。您可以稍後返回 Change Manager 並選擇 Settings (設定) 標籤上的 Edit (編輯)，進而更新這些組態設定。

若要設定 Change Manager 使用者身分識別管理和範本檢閱者

1. 登入 AWS Management Console。

   如果您使用的是適用於組織的 Change Manager，請使用您的委派管理員帳戶的憑證登入。使用者必須擁有必要的 AWS Identity and Access Management (IAM) 許可，以便更新 Change Manager 設定。

2. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
3. 在導覽窗格中，選擇 Change Manager。
4. 在服務首頁上，根據可用的選項執行下列其中一項操作：

   - 如果您使用 Change Manager搭配 AWS Organizations ，請選擇設定委派帳戶。
   - 如果您使用 Change Manager搭配單一 AWS 帳戶，請選擇設定 Change Manager。

-或-

選擇 Create sample change request (建立範例變更請求)、Skip (略過)，然後選擇 Settings (設定) 標籤。

5. 對於 User identity management (使用者身分識別管理)，選擇下列其中一項。

   - AWS Identity and Access Management (IAM) – Change Manager識別提出和核准請求的使用者，並使用您現有的使用者、群組和角色在 中執行其他動作。
   - AWS IAM Identity Center (IAM Identity Center) – 允許 IAM Identity Center 建立和管理身分，或連線到現有的身分來源，以識別在 中執行動作的使用者Change Manager。

6. 在 Template reviewer notification (範本檢閱者通知) 區段中，指定 Amazon Simple Notification Service (Amazon SNS) 主題，以便通知範本檢閱者新的變更範本或變更範本版本已準備好接受檢閱。確保您選擇的 Amazon SNS 主題已設定為傳送通知給範本檢閱者。

   如需有關針對變更範本檢閱者通知而建立和設定 Amazon SNS 主題的資訊，請參閱 為 Change Manager 通知設定 Amazon SNS 主題。

   1. 若要指定範本檢閱者通知的 Amazon SNS 主題，請選擇下列其中一項：

      - Enter an SNS Amazon Resource Name (ARN) (輸入 SNS Amazon 資源名稱 (ARN)) – 對於 Topic ARN (主題 ARN)，輸入現有 Amazon SNS 主題的 ARN。此主題可以位於您組織的任何帳戶中。
      - 選取現有的 SNS 主題 – 對於 Target notification topic (目標通知主題) 中，選取您目前 AWS 帳戶中的現有 Amazon SNS 主題的 ARN。（ 如果您尚未在目前的 AWS 帳戶 和 中建立任何 Amazon SNS 主題，則此選項無法使用 AWS 區域。）

      > **Note**
      >
      > 您選取的 Amazon SNS 主題必須設定為指定其傳送的通知以及要傳送的訂閱者。其存取政策也必須將許可授予 Systems Manager，以便 Change Manager 可以傳送通知。如需相關資訊，請參閱 為 Change Manager 通知設定 Amazon SNS 主題。

   2. 選擇 Add notification (新增通知)。

7. 在 Change template reviewers (變更範本檢閱者) 區段中，選取組織或帳戶中的使用者，以檢閱新的變更範本或變更範本版本，然後再將其用於您的營運中。

   變更範本檢閱者負責驗證其他使用者已提交供 Change Manager Runbook 工作流程使用之範本的適用性和安全性。

透過執行下列動作來選取變更範本檢閱者：

1. 選擇新增。
2. 選取您要指派為變更範本檢閱者之各個使用者、群組或 IAM 角色名稱旁的核取方塊。
3. 選擇 Add approvers (新增核准者)。

8. 選擇提交。

完成此初始安裝程序之後，請遵循 任務 2：設定 Change Manager 變更凍結事件核准者和最佳實務 中的步驟設定其他 Change Manager 設定和最佳實務。

任務 2：設定 Change Manager 變更凍結事件核准者和最佳實務

在您完成 任務 1：設定 Change Manager 使用者身分識別管理和範本檢閱者 中的步驟後，您可以指定變更凍結事件期間的變更請求的額外檢閱者，並指定您希望允許用於 Change Manager 營運的最佳實務。

變更凍結事件表示限制在目前的變更行事曆中 (中的 AWS Systems Manager Change Calendar行事曆狀態為 CLOSED)。在這些情況下，除了變更請求的一般核准者之外，或者如果使用允許自動核准的範本建立變更請求，則變更凍結核准者必須授予執行此變更請求的許可。如果沒有，則無法處理變更，直至行事曆狀態再次變為 OPEN。

若要設定 Change Manager 變更凍結事件核准者和最佳實務

1. 在導覽窗格中，選擇 Change Manager。
2. 選擇 Settings (設定) 標籤，然後選擇 Edit (編輯)。
3. 在 Approvers for change freeze events (變更凍結事件的核准者) 區段中，選取組織或帳戶中可核准變更的使用者，即使在 Change Calendar 中使用的行事曆目前已關閉。

> **Note**
>
> 若要允許變更凍結檢閱，您必須開啟 Best practices (最佳實務) 中的 Check Change Calendar for restricted change events (檢查變更行事曆是否有限制變更事件) 選項。

執行下列動作，以選取變更凍結事件的核准者：

1. 選擇新增。

2. 選取您要指派為變更凍結事件的核准者之各個使用者、群組或 IAM 角色名稱旁的核取方塊。
3. 選擇 Add approvers (新增核准者)。

4. 在頁面底部附近的 Best practices (最佳實務) 區段中，開啟您要針對下列每個選項強制執行的最佳實務。

- 選項：Check Change Calendar for restricted change events (檢查變更行事曆是否有限制變更事件)

  若要指定 Change Manager 檢查 Change Calendar 中的行事曆，以確保變更不會被排程事件封鎖，請先選取 Enabled (已啟用) 核取方塊，然後從 Change Calendar (變更行事曆) 清單中選取行事曆以檢查是否有限制事件。

  如需有關 Change Calendar 的詳細資訊，請參閱「AWS Systems Manager Change Calendar」。

- 選項：SNS topic for approvers for closed events (已關閉事件的核准者的 SNS 主題)

  1. 選擇下列其中一項，指定帳戶中的 Amazon Simple Notification Service (Amazon SNS) 主題，以便在變更凍結事件期間傳送通知給核准者。(請注意，您也必須在上述 Best practices (最佳實務) 的 Approvers for change freeze events (變更凍結事件的核准者) 中指定核准者。)

     - Enter an SNS Amazon Resource Name (ARN) (輸入 SNS Amazon 資源名稱 (ARN)) – 對於 Topic ARN (主題 ARN)，輸入現有 Amazon SNS 主題的 ARN。此主題可以位於您組織的任何帳戶中。
     - 選取現有的 SNS 主題 – 對於 Target notification topic (目標通知主題) 中，選取您目前 AWS 帳戶中的現有 Amazon SNS 主題的 ARN。（如果您尚未在目前的 AWS 帳戶 和 中建立任何 Amazon SNS 主題，則無法使用此選項 AWS 區域。）

     > **Note**
     >
     > 您選取的 Amazon SNS 主題必須設定為指定其傳送的通知以及要傳送的訂閱者。其存取政策也必須將許可授予 Systems Manager，以便 Change Manager 可以傳送通知。如需相關資訊，請參閱 為 Change Manager 通知設定 Amazon SNS 主題。

  2. 選擇 Add notification (新增通知)。

- 選項：Require monitors for all templates (需要監控所有範本)

  如果您想要確保組織或帳戶的所有範本都指定了 Amazon CloudWatch 警示來監控變更操作，則請選取 Enabled (已啟用) 核取方塊。

- 選項：Require template review and approval before use (使用前需要範本檢閱和核准)

  若要確保不會建立任何變更要求，且不會執行 Runbook 工作流程，而不會以已檢閱和已核准的範本為基礎，請選取 Enabled (已啟用) 核取方塊。

5. 選擇 Save (儲存)。

為 Change Manager 通知設定 Amazon SNS 主題

您可以設定 中的Change Manager工具 AWS Systems Manager，針對與變更請求和變更範本相關的事件傳送通知至 Amazon Simple Notification Service (Amazon SNS) 主題。完成下列任務，以接收您新增主題的 Change Manager 事件的通知。

主題

- 任務 1：建立並訂閱 Amazon SNS 主題
- 任務 2：更新 Amazon SNS 存取政策
- 任務 3：（選用）更新 AWS Key Management Service 存取政策

任務 1：建立並訂閱 Amazon SNS 主題

首先，您必須建立並訂閱 Amazon SNS 主題。如需詳細資訊，請參閱《Amazon Simple Notification Service 開發人員指南》中的建立 Amazon SNS 主題和訂閱 Amazon SNS 主題。

> **Note**
>
> 若要接收通知，您必須指定 Amazon SNS 主題的 Amazon Resource Name (ARN)，該主題與委派的管理員帳戶相同 AWS 區域 且 AWS 帳戶 相同。

任務 2：更新 Amazon SNS 存取政策

使用下列程序更新 Amazon SNS 存取政策，讓 Systems Manager 可以將 Change Manager 通知發佈至您在任務 1 中建立的 Amazon SNS 主題。如果不完成此任務，Change Manager 沒有為您新增主題的事件傳送通知的許可。

1. 登入 AWS Management Console ，並在 https://console.aws.amazon.com/sns/v3/home：// 開啟 Amazon SNS 主控台。
2. 在導覽窗格中，選擇主題。

3. 選擇您在任務 1 中建立的主題，然後選擇 Edit (編輯)。
4. 展開 Access policy (存取政策)。
5. 新增並更新下列 `Sid` 區塊至現有政策，並使用自己的資訊取代每個*#########*。

```
{
    "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"
            ]
        }
    }
}
```

   在現有 `Sid` 區塊後輸入此區塊，並將 *region*、*account-id* 和 *topic_name* 取代為您建立之主題的適當值。
6. 選擇 Save changes (儲存變更)。

現在，當您新增至主題的事件類型發生時，系統會傳送通知給 Amazon SNS 主題。

> **Important**
>
> 如果您使用 AWS Key Management Service (AWS KMS) 伺服器端加密金鑰設定 Amazon SNS 主題，則必須完成任務 3。

任務 3：（選用）更新 AWS Key Management Service 存取政策

如果您開啟 Amazon SNS 主題的 AWS Key Management Service (AWS KMS) 伺服器端加密，則也必須更新 AWS KMS key 您在設定主題時所選 的存取政策。使用下列處理程序更新存取政策，讓 Systems Manager 可以將 Change Manager 核准通知發佈至您在任務 1 中建立的 Amazon SNS 主題。

1. 在 https：//https://console.aws.amazon.com/kms 開啟 AWS KMS 主控台。
2. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。
3. 選擇您在建立主題時所選擇的客戶受管金鑰的 ID。
4. 在 Key policy (金鑰政策) 區段中，選擇 Switch to policy view (切換至政策檢視)。
5. 選擇編輯。
6. 在現有政策中的某個現有 `Sid` 區塊後輸入以下 `Sid` 區塊。將每個*#########*替換為自己的資訊。

```
{
    "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. 現在在資源政策中的某個現有 `Sid` 區塊之後輸入以下 `Sid` 區塊以協助防止跨服務混淆代理人問題。

   此區塊使用 `aws:SourceArn` 和 `aws:SourceAccount` 全域條件內容索引鍵，可限制 Systems Manager 為資源提供其他服務的許可。

   將每個*#########*取代為自己的資訊。

```
{
  "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:region:account-id:topic-name",
      "Condition": {
        "ArnLike": {
          "aws:SourceArn": "arn:aws:ssm:region:account-id:*"
        },
        "StringEquals": {
          "aws:SourceAccount": "account-id"
        }
      }
    }
  ]
}
```

8. 選擇 Save changes (儲存變更)。

設定 Change Manager 的角色和許可

在預設情況下，Change Manager 沒有在您的資源上執行動作的許可。您必須使用 AWS Identity and Access Management (IAM) 服務角色授予存取權，或擔任 角色。此角色可讓 Change Manager 代表您安全地執行在已核准變更請求中指定的 Runbook 工作流程。角色將 AWS Security Token Service (AWS STS) AssumeRole 信任授予 Change Manager。

藉由向角色提供這些權限以代表組織中的使用者執行操作，使用者本身無需被授予該許可陣列。許可允許的動作僅限於已核准的操作。

當您的帳戶或組織中的使用者建立變更請求時，他們可以選取此擔任角色來執行變更操作。

您可以為 Change Manager 建立新的擔任角色，或更新具備所需許可的現有角色。

如果您需要為 Change Manager 建立服務角色，請完成以下任務。

任務

- 任務 1：建立 Change Manager 的擔任角色策略

- 任務 2：建立 Change Manager 的擔任角色
- 任務 3：將 iam:PassRole 策略連接至其他角色
- 任務 4：將內嵌政策新增至 擔任角色以叫用其他 AWS 服務
- 任務 5：設定使用者存取至 Change Manager

任務 1：建立 Change Manager 的擔任角色策略

使用以下程序建立您將連接至 Change Manager 擔任角色的政策。

若要建立 Change Manager 的擔任角色策略

1. 在 https://console.aws.amazon.com/iam/ 中開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Policies (政策)，然後選擇 Create Policy (建立政策)。
3. 在 Create policy (建立政策) 頁面上，選擇 JSON 標籤，並利用以下內容更換預設內容，即您將在以下步驟中修改您自己的 Change Manager 操作。

> **Note**
>
> 如果您要建立政策以搭配單一 使用 AWS 帳戶，而非具有多個帳戶的組織 AWS 區域，您可以省略第一個陳述式區塊。在此使用 Change Manager 的單一帳戶情況中，無須使用 `iam:PassRole` 許可。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::delegated-admin-account-id: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:region:account-id:automation-definition/template-name:
$DEFAULT",
                "arn:aws:ssm:region::document/template-name"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:ListOpsItemEvents",
                "ssm:GetOpsItem",
                "ssm:ListDocuments",
                "ssm:DescribeOpsItems"
            ],
            "Resource": "*"
        }
    ]
}
```

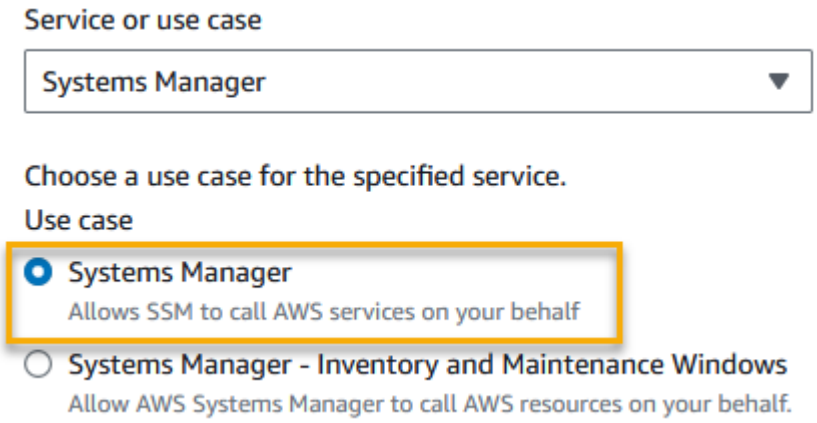
4. 對於 `iam:PassRole` 動作，請更新 `Resource` 值以包括為您組織定義之所有任務職能的 ARN，即您希望授與許可以啟動 Runbook 工作流程。
5. 將 *region*、*account-id*、*template-name*、*delegated-admin-account-id* 及 *job-function* 預留位置替換成您 Change Manager 操作的值。
6. 對於第二個 `Resource` 陳述式，修改列表以包括要授予許可的所有變更範本。或者，指定 `"Resource": "*"` 以向組織中的所有變更範本授與許可。
7. 選擇下一步：標籤。
8. (選用) 新增一個或多個標籤鍵值組來組織、追蹤或控制對此政策的存取。
9. 選擇下一步：檢閱。
10. 在 Review policy (檢閱政策) 頁面，在 Name (名稱) 方塊中輸入名稱 (如 **`MyChangeManagerAssumeRole`**)，接著輸入選用描述。
11. 選擇 Create policy (建立政策)，並繼續 任務 2：建立 Change Manager 的擔任角色。

任務 2：建立 Change Manager 的擔任角色

使用以下程序建立 Change Manager 的 Change Manager 擔任角色 (一種服務角色類型)。

若要建立 Change Manager 的擔任角色

1. 在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。
2. 在導覽窗格中，選擇角色，然後選擇建立角色。
3. 對於 Select trusted entity (選擇信任的實體)，請執行以下選項：
   1. 針對 Trusted entity type (信任的實體類型)，請選擇 AWS service (服務)
   2. 對於其他 的使用案例 AWS 服務，選擇 Systems Manager
   3. 選擇 Systems Manager，如下圖所示。



4. 選擇 Next (下一步)。
5. 在 Attached permissions policy (已連接許可政策) 頁面，搜尋您在 任務 1：建立 Change Manager 的擔任角色策略 中建立的擔任角色政策，如 **MyChangeManagerAssumeRole**。
6. 選取擔任角色政策名稱旁的核取方塊，然後選擇 Next: Tags (下一步：標籤)。
7. 在 Role name (角色名稱) 中，輸入新執行個體設定檔的名稱，如 **MyChangeManagerAssumeRole**。
8. (選用) 對於 Description (說明)，更新此執行個體角色的說明。
9. (選用) 新增一個或多個標籤鍵值組來組織、追蹤或控制對此角色的存取。
10. 選擇下一步：檢閱。
11. (選用) 對於 Tags (標籤)，新增一個或多個標籤鍵值組來整理、追蹤或控制存取此角色的存取權，然後選擇 Create role (建立角色)。系統會讓您回到 Roles (角色) 頁面。
12. 選擇 Create role (建立角色)。系統會讓您回到 Roles (角色) 頁面。

13. 在 Roles (角色) 頁面，選擇您剛建立的角色，以開啟 Summary (摘要) 頁面。

任務 3：將 **iam:PassRole** 策略連接至其他角色

使用以下程序將 iam:PassRole 政策連接至 IAM 執行個體設定檔或 IAM 服務角色。(Systems Manager 服務使用 IAM 執行個體設定檔與 EC2 執行個體進行通訊。對於混合多雲端環境中的非 EC2 受管節點，則會改用 IAM 服務角色。)

透過連接iam:PassRole政策，Change Manager服務可以在執行 Runbook 工作流程時將擔任角色許可傳遞給其他 服務或 Systems Manager 工具。

將 **iam:PassRole** 政策連接至 IAM 執行個體設定檔或服務角色

1. 在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。
2. 在導覽窗格中，選擇角色。
3. 搜尋您建立的 Change Manager 擔任角色 (例如 **MyChangeManagerAssumeRole**)，並選擇其名稱。
4. 在擔任角色的 Summary (摘要) 頁面，選擇 Permissions (許可) 標籤。
5. 選擇 Add permissions, Create inline policy (新增許可，建立內嵌政策)。
6. 在 Create policy (建立政策) 頁面，選擇 Visual editor (視覺化編輯器) 標籤。
7. 選擇 Service (服務)，接著選擇 IAM (IAM)。
8. 在 Filter actions (篩選動作) 文字方塊中輸入 **PassRole**，接著選擇 PassRole 選項。
9. 展開 Resources (資源)。確認 Specific (特定) 已選取，接著選擇 Add ARN (新增 ARN)。
10. 在 Specify ARN for role (指定角色的 ARN) 欄位中，輸入要向其傳送擔任角色許可的 IAM 執行個體設定檔角色或 IAM 服務角色的 ARN。系統會填入 Account (帳戶) 和 Role name with path (角色名稱與路徑) 欄位。
11. 選擇 Add (新增)。
12. 選擇 Review policy (檢閱政策)。
13. 在 Name (名稱) 中，輸入名稱來識別政策，然後選擇 Create policy (建立政策)。

詳細資訊

- 設定 Systems Manager 所需的執行個體許可
- 在混合多雲端環境中建立 Systems Manager 所需的 IAM 服務角色

任務 4：將內嵌政策新增至 擔任角色以叫用其他 AWS 服務

當變更請求 AWS 服務 使用 Change Manager 擔任角色叫用其他 時，必須設定擔任角色以允許叫用這些服務。此要求適用於可用於變更請求的所有 AWS Automation Runbook (AWS-* Runbook)，例如 `AWS-ConfigureS3BucketLogging`、 `AWS-CreateDynamoDBBackup`和 `AWS-RestartEC2Instance` Runbook。此要求也適用於您建立的任何自訂 Runbook， AWS 服務 其透過使用呼叫其他服務的動作來叫用其他 。例如，如果您使用 `aws:executeAwsApi`、`aws:CreateStack` 或 `aws:copyImage` 等動作，則您必須為服務角色設定可叫用這些服務的許可。您可將 IAM 內嵌政策新增至角色，以啟用其他 AWS 服務 的許可。

若要將內嵌政策新增至擔任角色以叫用其他 AWS 服務 (IAM 主控台)

1. 登入 AWS Management Console ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇角色。
3. 在清單中，請選擇您要更新的擔任角色名稱 (如 `MyChangeManagerAssumeRole`)。
4. 選擇許可索引標籤標籤。
5. 選擇 Add permissions, Create inline policy (新增許可，建立內嵌政策)。
6. 選擇 JSON 標籤。
7. 輸入 AWS 服務 您要叫用 的 JSON 政策文件。以下是兩個 JSON 政策文件範例。

   Amazon S3 **`PutObject`** 和 **`GetObject`** 範例

```
{
    "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`** 範例

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":"ec2:CreateSnapshot",
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":"ec2:DescribeSnapshots",
         "Resource":"*"
      }
   ]
}
```

如需 IAM 政策語言的詳細資訊，請參閱《IAM 使用者指南》中的 IAM JSON 政策參考。

8. 完成時，請選擇 Review policy (檢閱政策)。Policy Validator (政策檢查工具) 會回報任何語法錯誤。
9. 在 Name (名稱) 中，輸入名稱來識別您正在建立的政策。檢閱政策 Summary (摘要) 來查看您的政策所授予的許可。然後選擇 Create policy (建立政策) 來儲存您的工作。
10. 在您建立內嵌政策後，它會自動嵌入您的角色中。

任務 5：設定使用者存取至 Change Manager

如果使用者、群組或角色受獲指派管理員許可，則您可以存取 Change Manager。如果您沒有管理員許可，則管理員必須指派 AmazonSSMFullAccess 受管政策或提供相當許可的政策給使用者、群組或角色。

使用以下程序將使用者設定為使用 Change Manager。您選擇的使用者會擁有設定和執行 Change Manager 的許可。

視您在組織中使用的身分應用程式而定，
您可以選取三個可用於設定使用者存取權中的任何一個選項。設定使用者存取權時，指派或新增下列項目：

1. 指派 AmazonSSMFullAccess 政策或相當的政策，以授予存取 Systems Manager 的許可。
2. 指派 iam:PassRole 政策。

3. 新增您在 任務 2：建立 Change Manager 的擔任角色 結尾複製的 Change Manager 擔任角色的 ARN。

若要提供存取權，請新增權限至您的使用者、群組或角色：

- 中的使用者和群組 AWS IAM Identity Center：

  建立權限合集。請按照 AWS IAM Identity Center 使用者指南 中的 建立權限合集 說明進行操作。
- 透過身分提供者在 IAM 中管理的使用者：

  建立聯合身分的角色。遵循「IAM 使用者指南」的為第三方身分提供者 (聯合) 建立角色中的指示。
- IAM 使用者：
  - 建立您的使用者可擔任的角色。請按照「IAM 使用者指南」的為 IAM 使用者建立角色中的指示。
  - (不建議) 將政策直接附加至使用者，或將使用者新增至使用者群組。請遵循 IAM 使用者指南的新增許可到使用者 (主控台) 中的指示。

您已完成設定 Change Manager 所需的角色。您現在可以在 Change Manager 操作中使用 Change Manager 擔任角色 ARN。

控制對自動核准 Runbook 工作流程的存取

在為組織或帳戶建立的每個變更範本中，您可以指定從該範本建立的變更請求是否可以作為自動核准的變更請求來執行，這表示它們會在沒有檢閱步驟的情況下自動執行 (變更凍結事件除外)。

不過，您可能想要防止某些使用者、群組或 AWS Identity and Access Management (IAM) 角色執行自動核准的變更請求，即使變更範本允許亦然。您可以透過在指派給使用者、群組或 IAM 角色的 IAM 政策中使用 `StartChangeRequestExecution` 操作的 `ssm:AutoApprove` 條件金鑰，來執行此操作。

您可以將下列政策新增為內嵌政策，其中條件指定為`false`，可防止使用者執行可自動核准的變更請求。

```
{
    "Version": "2012-10-17",
    "Statement": [
            {
            "Effect": "Allow",
            "Action": "ssm:StartChangeRequestExecution",
            "Resource": "*",
```

```
            "Condition": {
                "BoolIfExists": {
                    "ssm:AutoApprove": "false"
                }
            }
        }
    ]
}
```

如需有關指定內嵌政策的資訊，請參閱《IAM 使用者指南》中的內嵌政策和新增和移除 IAM 身分許可。

如需有關 Systems Manager 政策條件金鑰的詳細資訊，請參閱 Systems Manager 條件金鑰。

使用 Change Manager

使用 Change Manager中的工具 AWS Systems Manager，整個組織或單一 中的使用者 AWS 帳戶 可以執行與變更相關的任務，而他們已獲得必要的許可。 Change Manager任務包括下列項目：

- 建立、檢閱及核准或拒絕變更範本。

  變更範本是 Change Manager 中的組態設定的集合，可定義必要核准、可用 Runbook 以及變更請求的通知選項等項目。
- 建立、檢閱及核准或拒絕變更請求。

  變更請求是在 中執行 Automation Runbook Change Manager的請求，該自動化 Runbook 會更新您或 AWS 內部部署環境中的一或多個資源。變更請求是使用變更範本建立的。
- 指定組織或帳戶中的哪些使用者可以成為變更範本和變更請求的檢閱者。
- 編輯組態設定，例如如何在 Change Manager 中管理使用者身分以及可在 Change Manager 營運中強制使用哪些最佳實務。如需有關設定這些設定的資訊，請參閱 設定 Change Manager 選項和最佳實務。

主題

- 使用變更範本
- 使用變更請求
- 檢閱變更請求詳細資訊、任務和時間表 (主控台)
- 檢視變更請求的彙總計數 (命令列)

使用變更範本

變更範本是 Change Manager 中的組態設定的集合，可定義必要核准、可用 Runbook 以及變更請求的通知選項等項目。

> **Note**
>
> AWS 提供範例 Hello World 變更範本，您可以用來嘗試 Change Manager，這是 中的工具 AWS Systems Manager。但是，您可以建立自己的變更範本，以定義您要允許對您組織或帳戶中的資源的變更。

執行 Runbook 工作流程時所做的變更以自動化 Runbook 內容為基礎。在您建立的每個變更範本中，您可以包含一個或多個自動化 Runbook，而提出變更請求的使用者可以選擇在更新期間執行。您也可以建立變更範本，以允許申請者為變更請求選擇任何可用的自動化 Runbook。

若要建立變更範本，您可以使用 Create template (建立範本) 主控台頁面中的 Builder (建置器) 選項，以建置變更範本。或者，使用 Editor (編輯器) 選項，您可以使用您想要的 Runbook 工作流程所需的組態手動編寫 JSON 或 YAML 內容。您也可以使用命令列工具來建立變更範本，其中包含存放在外部檔案中的變更範本的 JSON 內容。

主題

- 嘗試 AWS 受管Hello World變更範本
- 建立變更範本
- 檢閱及核准或拒絕變更範本
- 刪除變更範本

嘗試 AWS 受管`Hello World`變更範本

您可以使用範例變更範本 `AWS-HelloWorldChangeTemplate`，此範本使用範例自動化執行手冊 `AWS-HelloWorld`，在 中完成 Change Manager工具設定後測試審核和核准程序 AWS Systems Manager。此範本旨在測試或驗證您已設定的許可、核准者指派和核准程序。 AWS以提供在您的組織或帳戶中使用此變更範本的核准。不過，以此變更範本為基礎的任何變更請求仍必須經過組織或帳戶中的檢閱者核准。

與此範本關聯的 Runbook 工作流程的結果，是在自動化步驟的輸出中列印訊息，而不是對資源進行變更。

開始之前

開始之前，請確保您已完成下列任務：

- 如果您使用 AWS Organizations 管理整個組織的變更，請完成中所述的組織設定任務設定適用於組織的 Change Manager (管理帳戶)。
- 如 設定 Change Manager 選項和最佳實務 所述，為您的委派管理員帳戶或單一帳戶設定 Change Manager。

> **Note**
>
> 當您測試 Hello World 變更範本時，如果您在 Change Manager 設定中開啟了最佳實務選項 Require monitors for all templates (需要監控所有範本)，請暫時將其關閉。

嘗試受 AWS 管 Hello World 變更範本

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Change Manager。
3. 選擇 Create request (建立請求)。
4. 選擇名為 `AWS-HelloWorldChangeTemplate` 的變更範本，然後選擇 Next (下一步)。
5. 對於 Name (名稱)，輸入可輕鬆識別用途的變更請求名稱，例如 **`MyChangeRequestTest`**。
6. 如需建立變更請求的剩餘步驟，請參閱 建立變更請求。

後續步驟

如需核准變更請求的資訊，請參閱 檢閱及核准或拒絕變更請求。

若要檢視變更請求的狀態和結果，請在 Change Manager 中的 Requests (請求) 標籤上選擇變更請求的名稱。

建立變更範本

變更範本是 Change Manager 中的組態設定的集合，可定義必要核准、可用 Runbook 以及變更請求的通知選項等項目。

您可以使用 主控台Change Manager，包括建置器和編輯器選項或命令列工具 AWS Systems Manager，在 中的工具 中建立操作的變更範本。

主題

- 關於變更範本中的核准
- 使用建置器建立變更範本
- 使用編輯器建立變更範本
- 使用命令列工具建立變更範本

關於變更範本中的核准

對於根據您所建立的每個變更範本建立的變更請求，您最多可以指定五個核准層級。對於每個層級，您最多可以指定五個潛在核准者。核准者不限於單一使用者。您也可以將 IAM 群組或 IAM 角色指定為個別核准者。對於 IAM 群組和 IAM 角色，屬於該群組或角色的一或多名使用者可以提供核准，以獲得變更請求所需的核准總數。您還可以指定比變更範本要求數量更多的核准者。

Change Manager 支援兩種主要核准方法：逐級核准和逐行核准。在某些情況下，這兩種類型的組合也是可行的。建議您在 Change Manager 操作中僅使用逐級核准。

Per-level approvals

建議使用。自 2023 年 1 月 23 日起，Change Manager 支援逐級核准。在此模型中，對於變更範本中的每個核准層級，您必須先指定該層級需要的核准數量。接著，您至少要為該層級指定該數量的核准者，也可指定更多核准者。但是，只要達到您為該層級指定的核准數量，就可以核准變更請求。例如，您可以指定五名核准者，但只需要三個核准。

如需此核准類型的主控台檢視和 JSON 範例，請參閱the section called “逐級核准組態範例”。

Per-line approvals

支援回溯相容性。Change Manager 原始版本僅支援逐行核准。在此模型中，每個核准層級指定的每名核准者都是一個核准行。變更請求必須獲得每名核准者的核准，才能在該層級獲得核准。在 2023 年 1 月 23 日前這是唯一支援的核准模型。在此日期之前建立的變更範本會繼續支援逐行核准，但我們建議改用逐級核准。

如需此核准類型的主控台檢視和 JSON 範例，請參閱the section called “逐行核准組態範例”。

Combined per-line and per-level approvals

不建議使用。在主控台中，建置器索引標籤不再支援新增逐行核准。但是，在某些情況下，您的變更範本可能會同時有逐行核准與逐級核准。如果您更新在 2023 年 1 月 23 日前建立的變更範本，或者如果您手動編輯變更範本的 YAML 內容，來建立或更新變更範本，就會發生這種情況。

如需此核准類型的主控台檢視和 JSON 範例，請參閱the section called “逐級核准和逐行核准組態組合的範例”。

> **Important**
>
> 雖然您可以建立結合逐行核准和逐級核准的變更範本，但不建議或不需要使用此種組態。核准數量要求更高的核准類型 (逐行核准和逐級核准) 優先。例如：
>
> - 如果變更範本在逐級核准中指定了需要三個核准，但又在逐行核准中指定了需要五個核准，則需要獲得五個核准。
> - 如果變更範本在逐級核准中指定了需要四個核准，但又在逐行核准中指定了需要兩個核准，則需要獲得四個核准。

您可以透過手動編輯 YAML 或 JSON 內容，建立同時包含逐行核准和逐級核准的層級。然後，建置器索引標籤會顯示控制項，以指定層級與個別行所需的核准數量。不過，您使用主控台新增的新層級仍然只支援逐級核准組態。

變更請求通知和拒絕

Amazon SNS 通知

使用變更範本建立變更請求時，系統會向為該層級核准通知指定之 Amazon Simple Notification Service (Amazon SNS) 主題的訂閱用戶傳送通知。您可以在變更範本中指定通知主題，或允許建立變更請求的使用者指定主題。

在某個層級獲得所需核准達到最低數量之後，系統會將通知傳送給為下個層級核准通知指定的 Amazon SNS 主題訂閱用戶 (核准者)，依此類推。

> **Important**
>
> 確保您指定的 IAM 角色、群組和使用者合起來能提供的核准者數量能夠達到您指定的所需核准數量。例如，如果您只指定一個包含三名使用者的 IAM 群組作為核准者，就無法為該層級指定需要五個核准，只能指定三個或更少數量的核准。

變更請求拒絕

無論指定多少核准層級和核准者，只要變更請求遭到一個拒絕，就能阻止該請求的執行手冊工作流程發生。

Change Manager 核准類型範例

下列範例示範 Change Manager 中三種核准類型的主控台檢視和 JSON 內容。

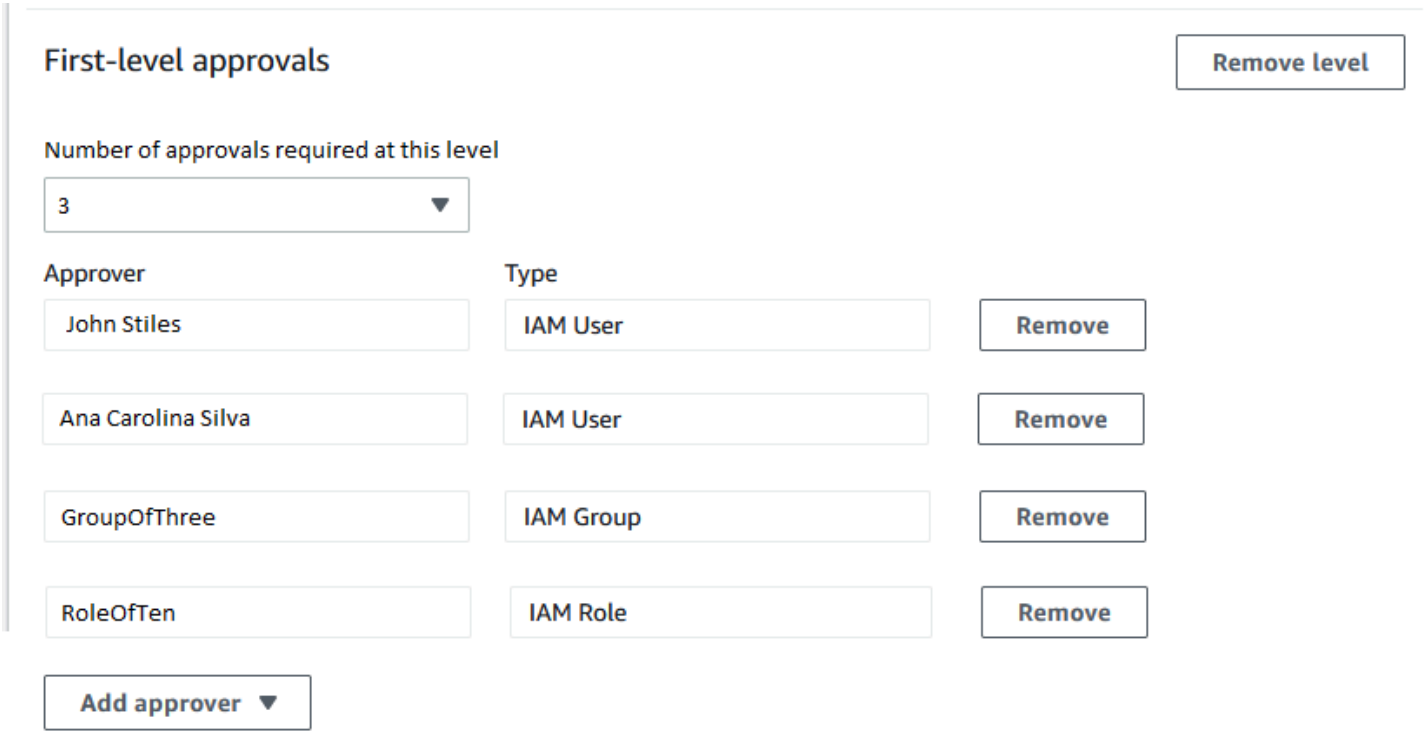
主題

- 逐級核准組態範例
- 逐行核准組態範例
- 逐級核准和逐行核准組態組合的範例

逐級核准組態範例

在下圖中顯示的逐級核准層級設定中，需要三個核准。這些核准可來自指定為核准者的任何 IAM 使用者、群組和角色組合。指定的核准者包括兩名 IAM 使用者 (John Stiles 和 Ana Carolina Silva)、一個包含三名成員的使用者群組 (`GroupOfThree`)，以及代表十名使用者的使用者角色 (`RoleOfTen`)。

如果 `GroupOfThree` 群組中的三名使用者都核准了該變更請求，則意味著變更請求在該層級已獲得核准。不需要獲得每個使用者、群組或角色的核准。最小數量的核准可以來自任何指定核准者的組合。建議您在 Change Manager 操作中使用逐級核准。



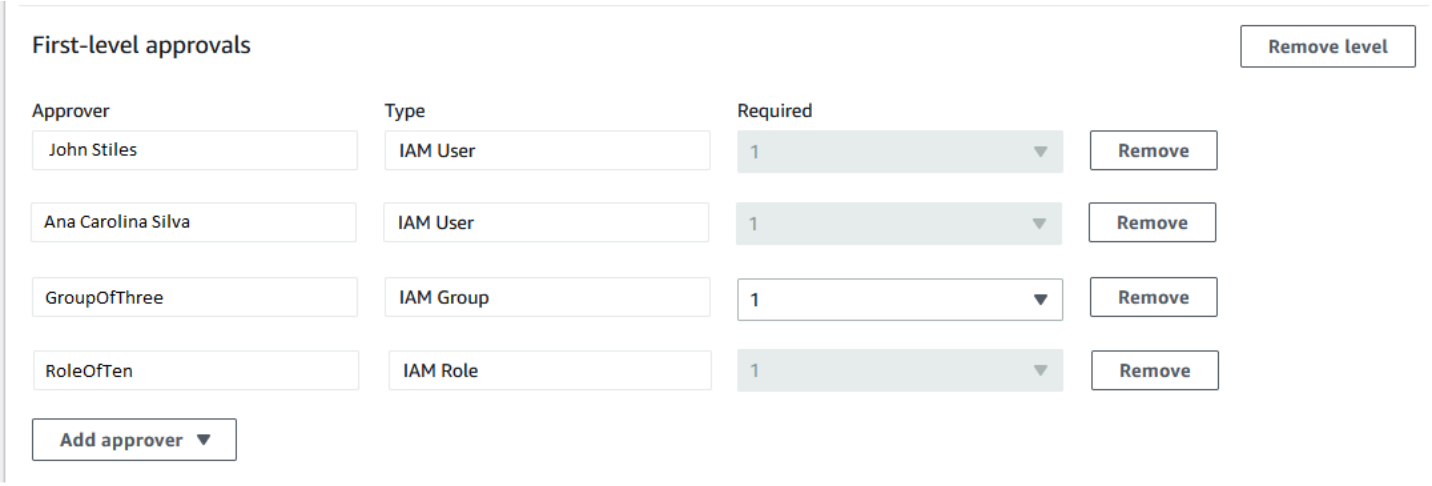
下列範例說明此組態的 YAML 程式碼部分。

> **Note**
>
> 這個版本的 YAML 程式碼包括額外的輸入 `MinRequiredApprovals` (首字母大寫 `M`)。此輸入的值表示需要從所有可用檢閱者獲得的核准數量。另請注意，`Approvers` 清單中每名核准者的 `minRequiredApprovals` (首字母小寫 `m`) 值為 `0` (零)。這表示核准者可以對整體核准做出貢獻，但並不強制要求。

```
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
```

逐行核准組態範例

在下圖中顯示的核准層級設定中，指定了四名核准者。其中包含兩名 IAM 使用者 (John Stiles 和 Ana Carolina Silva)、一個包含三名成員的使用者群組 (`GroupOfThree`)，以及代表十名使用者的使用者角色 (`RoleOfTen`)。處於回溯相容性考慮，仍然支援逐行核准，但不建議使用。



如果使用此逐行核准組態，要核准此變更請求，需要所有核准者行 (John Stilles、Ana Carolina Silva、GroupOfThree 群組的一名成員以及 RoleOfTen 角色的一名成員) 都核准此變更請求。

下列範例說明此組態的 YAML 程式碼部分。

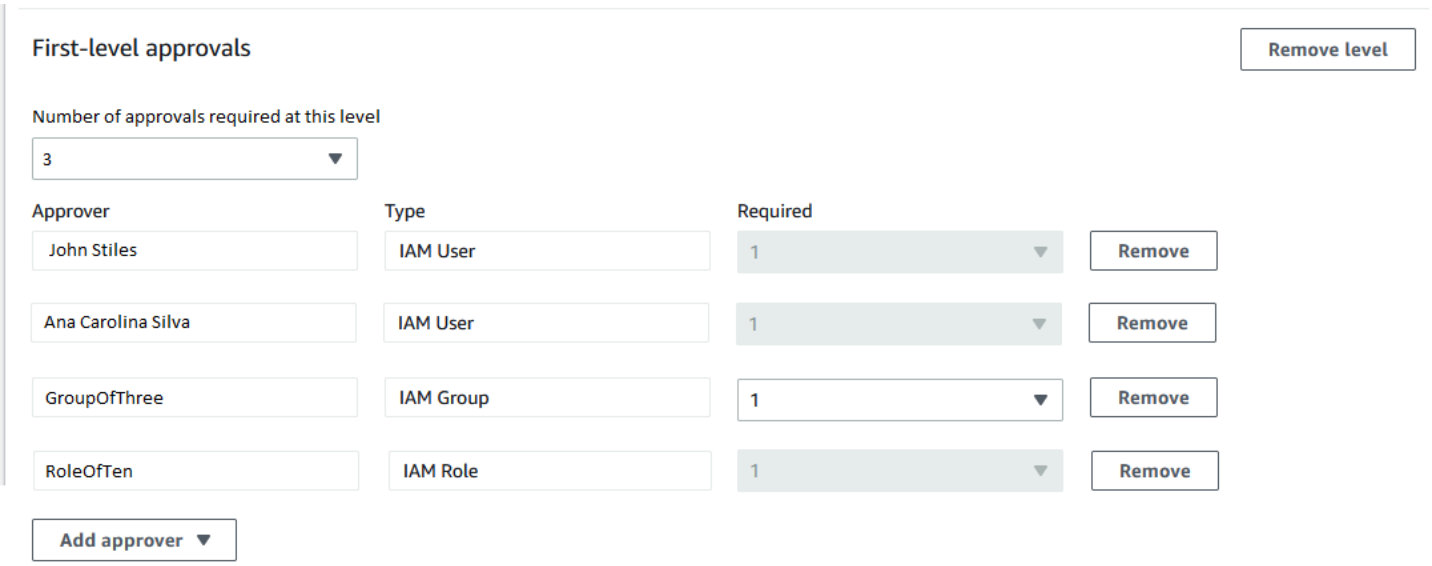
> **Note**
>
> 請注意，每個 minRequiredApprovals 核准者的值都是 1。這表示只需要從每名核准者獲得一個核准。

```
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
```

逐級核准和逐行核准組態組合的範例

在下圖中顯示的逐級核准和逐行核准組合設定中，為該層級指定了三個核准，但行級別指定了四個核准。核准數量要求更高的核准類型會優先於其他核准類型，因此此組態需要四個核准。不建議將逐級核准和逐行核准結合使用。




```
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
```

主題

- 使用建置器建立變更範本
- 使用編輯器建立變更範本
- 使用命令列工具建立變更範本

使用建置器建立變更範本

使用 中用於變更範本的建置器Change Manager，即 中的工具 AWS Systems Manager，您可以設定變更範本中定義的 Runbook 工作流程，而不必使用 JSON 或 YAML 語法。指定您的選項之後，系統會將您的輸入轉換成 YAML 格式，從而讓 Systems Manager 可以用來執行 Runbook 工作流程。

若要使用建置器建立變更範本

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Change Manager。
3. 選擇建立範本。
4. 對於 Name (名稱)，輸入可輕鬆識別用途的範本名稱，例如 **UpdateEC2LinuxAMI**。
5. 在 Change template details (變更範本詳細資訊) 區段中，執行下列動作：
   - 對於 Description (描述)，簡要介紹如何及何時使用您正建立的變更範本。

此貓叔描述可協助建立變更請求的使用者判斷他們是否使用了正確的變更範本。它可協助檢閱變更請求的使用者了解是否應該核准請求。

- 對於 Change template type (變更範本類型)，指定您要建立標準變更範本還是緊急變更範本。

  緊急變更範本用於必須進行變更的情況，即使使用中的行事曆中的事件以其他方式封鎖變更 AWS Systems Manager Change Calendar。從緊急變更範本建立的變更請求仍必須由其指定的核准者核准，但即使行事曆遭到封鎖，仍然可以執行請求的變更。

- 對於 Runbook options (Runbook 選項)，指定使用者在建立變更請求時可以從中選擇的 Runbook。您可以新增單一 Runbook 或多個 Runbook。或者，您可以允許申請者指定要使用的 Runbook。在任何這些情況下，變更請求中只能包含一個 Runbook。
- 對於 Runbook，選取 Runbook 的名稱，以及使用者可以為其變更請求選擇的 Runbook 版本。無論您新增多少個 Runbook 至變更範本，每個變更請求只能選取一個 Runbook。

  如果您之前選擇 Any runbook can be used (任何 Runbook 均可使用)，則不必指定 Runbook。

  > **Tip**
  >
  > 選取 Runbook 和 Runbook 版本，然後選擇 View (檢視) 以在 Systems Manager 文件介面中檢查 Runbook 的內容。

6. 在 Template information (範本資訊) 區段中，使用 Markdown 為從此變更範本建立變更請求的使用者輸入資訊。我們提供了一組問題，您可以為建立變更請求的使用者加入這些問題，或者您可以改為新增其他資訊和問題。

   > **Note**
   >
   > Markdown 是一種標示語言，可讓您新增維基樣式的描述至文件內，以及在文件內新增個別步驟。如需使用 Markdown 的相關資訊，請參閱在 AWS中使用 Markdown。

   我們建議向使用者提供有關變更請求的問題，以供其回答，進而協助核准者決定是否授予每個變更請求，例如列出作為變更的一部分而執行所需的任何手動步驟和回復計劃。

> **Tip**
>
> 在 Hide preview (隱藏預覽) 和 Show preview (顯示預覽) 之間切換，即可在撰寫時查看內容的外觀。

7. 在 Change request approvals (變更請求核准) 區段，執行下列動作：

   - (選用) 如果您要允許自動執行從此變更範本建立的變更請求，而不需由任何核准者進行檢閱 (變更凍結事件除外)，請選取 Enable auto-approval (啟用自動核准)。

     > **Note**
     >
     > 在變更範本中啟用自動核准可為使用者提供繞過檢閱者的選項。在建立變更請求時，他們仍然可以選擇指定檢閱者。因此，您仍必須在變更範本中指定檢閱者選項。

     > **Important**
     >
     > 如果您啟用變更範本的自動核准，則使用者可以使用該範本提交變更請求，而這些範本在執行前不需要檢閱者進行檢閱 (變更凍結事件核准者除外)。如果您想要限制特定使用者、群組或 IAM 角色提交自動核准請求，您可以針對此目的使用 IAM 政策中的條件。如需詳細資訊，請參閱控制對自動核准 Runbook 工作流程的存取。

   - 在此層級所需的核准數量中，選擇從此變更範本建立之變更請求在此層級必須獲得的核准數量。
   - 若要新增強制性第一層級核准者，請選擇 Add approver (新增核准者)，然後從下列選項中選擇：
     - 範本指定的核准者 – 從您的帳戶中選擇一個或多個使用者、群組或 AWS Identity and Access Management (IAM) 角色，以核准從此變更範本建立的變更請求。使用此範本建立的任何變更請求都必須由您指定的每位核准者檢閱和核准。
     - 請求指定的核准者 – 提出變更請求的使用者會在提出請求時指定檢閱者，並且可以從您的帳戶中的使用者清單進行選擇。

       您在 Required (必填) 欄中輸入的數字會決定使用此變更範本的變更請求必須指定的檢閱者數量。

> **Important**
>
> 在 2023 年 1 月 23 日之前，建置器索引標籤僅支援逐行核准。使用建置器索引標籤新增的新變更範本和新增至現有變更範本的新層級僅支援逐級核准。建議您在 Change Manager 操作中僅使用逐級核准。
> 如需詳細資訊，請參閱關於變更範本中的核准。

- 對於 SNS topic to notify approvers (通知核准者的 SNS 主題)，請執行下列動作：

  1. 選擇下列其中一項，以在您的帳戶中指定 Amazon Simple Notification Service (Amazon SNS) 主題，進而用於向核准者傳送變更請求已準備好進行審核的通知：

     - Enter an SNS Amazon Resource Name (ARN) (輸入 SNS Amazon 資源名稱 (ARN)) – 對於 Topic ARN (主題 ARN)，輸入現有 Amazon SNS 主題的 ARN。此主題可以位於您組織的任何帳戶中。
     - 選取現有的 SNS 主題 – 對於 Target notification topic (目標通知主題) 中，選取您目前 AWS 帳戶中的現有 Amazon SNS 主題的 ARN。（如果您尚未在目前的 AWS 帳戶 和 中建立任何 Amazon SNS 主題，則無法使用此選項 AWS 區域。）
     - 建立變更請求時指定 SNS 主題 – 建立變更請求的使用者可以指定要用於通知的 Amazon SNS 主題。

     > **Note**
     >
     > 您選取的 Amazon SNS 主題必須設定為指定其傳送的通知以及要傳送的訂閱者。其存取政策也必須將許可授予 Systems Manager，以便 Change Manager 可以傳送通知。如需相關資訊，請參閱 為 Change Manager 通知設定 Amazon SNS 主題。

  2. 選擇 Add notification (新增通知)。

8. (選用) 若要新增額外層級的核准者，請選擇 Add approval level (新增核准層級)，然後在為此範本指定的核准者和請求指定的核准者之間進行選擇。然後選擇 SNS 主題，以通知此層級的核准者。

   在第一層級核准者收到所有核准之後，系統會通知第二層級核准者，依此類推。

   您可以在每個範本中新增最多 5 個核准者層級。例如，您可能需要第一個層級技術角色的使用者核准，然後需要第二個層級的管理核准。

9. 在 Monitoring (監控) 區段，對於 CloudWatch alarm to monitor (要監控的 CloudWatch 警示)，輸入目前帳戶中的 Amazon CloudWatch 警示的名稱，以監控基於此範本的 Runbook 工作流程的進度。

> **Tip**
>
> 若要建立新的警示，或檢閱您要指定的警示設定，請選擇 Open the Amazon CloudWatch console (開啟 Amazon CloudWatch 主控台)。如需有關使用 CloudWatch 警示的資訊，請參閱《Amazon CloudWatch 使用者指南》中的使用 CloudWatch 警示。

10. 在 Notifications (通知) 區段中，執行以下操作：

    1. 選擇下列其中一項，指定帳戶中的 Amazon SNS 主題，用於傳送使用此變更範本建立之變更請求的通知：
       - Enter an SNS Amazon Resource Name (ARN) (輸入 SNS Amazon 資源名稱 (ARN)) – 對於 Topic ARN (主題 ARN)，輸入現有 Amazon SNS 主題的 ARN。此主題可以位於您組織的任何帳戶中。
       - 選取現有的 SNS 主題 – 對於 Target notification topic (目標通知主題) 中，選取您目前 AWS 帳戶中的現有 Amazon SNS 主題的 ARN。（如果您尚未在目前的 AWS 帳戶 和 中建立任何 Amazon SNS 主題，則此選項無法使用 AWS 區域。）

       > **Note**
       >
       > 您選取的 Amazon SNS 主題必須設定為指定其傳送的通知以及要傳送的訂閱者。其存取政策也必須將許可授予 Systems Manager，以便 Change Manager 可以傳送通知。如需相關資訊，請參閱 為 Change Manager 通知設定 Amazon SNS 主題。

    2. 選擇 Add notification (新增通知)。

11. (選用) 在 Tags (標籤) 區段中，將一個或多個標籤索引鍵名稱/值對套用到變更範本。

    標籤是您指派給資源的選用性中繼資料。使用標籤，您即可以不同的方式 (例如用途、擁有者或環境) 將資源分類。例如，您可能想要標記變更範本，以識別變更範本所進行的類型以及其執行所在的環境。在這種情況下，您可以指定以下索引鍵名稱/值對：

    - `Key=TaskType,Value=InstanceRepair`
    - `Key=Environment,Value=Production`

12. 選擇 Save and preview (儲存與預覽)。
13. 檢閱您正在建立之變更範本的詳細資訊。

    如果您要在提交變更範本以供檢閱之前，對變更範本進行變更，請選擇 Actions, Edit (動作，編輯)。

如果您對變更範本的內容感到滿意，請選擇 Submit for review (提交審核)。您組織或帳戶中已在 Change Manager 中的 Settings (設定) 標籤上指定為範本檢閱者的使用者會收到通知：新的變更範本正待其檢閱。

如果已為變更範本指定 Amazon SNS 主題，則會在變更範本遭到拒絕或獲得核准時傳送通知。如果您沒有收到與此變更範本相關的通知，您可以稍後返回 Change Manager，以檢查其狀態。

使用編輯器建立變更範本

使用本主題中的步驟，透過輸入 JSON 或 YAML Change Manager而非使用主控台控制項 AWS Systems Manager，在 中的工具 中設定變更範本。

若要使用編輯器建立變更範本

1. 在導覽窗格中，選擇 Change Manager。
2. 選擇建立範本。
3. 對於 Name (名稱)，輸入可輕鬆識別用途的範本名稱，例如 **`RestartEC2LinuxInstance`**。
4. 在 Change template details (變更範本詳細資訊) 上方，選擇 Editor (編輯器)。
5. 在 Document editor (文件編輯器) 區段中，選擇 Edit (編輯)，然後輸入變更範本的 JSON 或 YAML 內容。

   以下是範例。

   > **Note**
   >
   > 使用參數 `minRequiredApprovals` 來指定必須要有多少指定層級的審查者核准使用此範本建立的變更請求。
   > 此範例會示範兩個核准層級。您最多可以指定五個核准層級，但只需要一個層級。
   > 在第一級中，特定使用者 "John-Doe" 必須核准每個變更請求。接下來，IAM 角色 `Admin` 的任何三個成員必須核准變更請求。
   > 如需有關變更範本核准的詳細資訊，請參閱關於變更範本中的核准。

   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. 選擇 Save and preview (儲存與預覽)。
7. 檢閱您正在建立之變更範本的詳細資訊。

   如果您要在提交變更範本以供檢閱之前，對變更範本進行變更，請選擇 Actions, Edit (動作，編輯)。

   如果您對變更範本的內容感到滿意，請選擇 Submit for review (提交審核)。您組織或帳戶中已在 Change Manager 中的 Settings (設定) 標籤上指定為範本檢閱者的使用者會收到通知：新的變更範本正待其檢閱。

   如果已為變更範本指定 Amazon Simple Notification Service (Amazon SNS) 主題，則會在變更範本遭到拒絕或獲得核准時傳送通知。如果您沒有收到與此變更範本相關的通知，您可以稍後返回 Change Manager，以檢查其狀態。

使用命令列工具建立變更範本

下列程序說明如何使用 AWS Command Line Interface (AWS CLI)（在 Linux、 或 Windows 上） macOS或在 中 AWS Tools for Windows PowerShell 建立變更請求Change Manager，即 中的工具 AWS Systems Manager。

若要建立變更範本

1. AWS Tools for PowerShell如果您尚未安裝和設定 AWS CLI 或 。

   如需相關資訊，請參閱安裝或更新 AWS CLI的最新版本和安裝 AWS Tools for PowerShell。

2. 在您的本機電腦建立名稱如 `MyChangeTemplate.json` 的 JSON 檔案，然後將您變更範本中的內容貼至其中。

   > **Note**
   >
   > 變更範本會使用 0.3 版本的結構描述，其中並不包含與自動化 Runbook 相同的所有支援。

   以下是範例。

   > **Note**
   >
   > 使用參數 `minRequiredApprovals` 來指定必須要有多少指定層級的審查者核准使用此範本建立的變更請求。
   > 此範例會示範兩個核准層級。您最多可以指定五個核准層級，但只需要一個層級。
   > 在第一級中，特定使用者 "John-Doe" 必須核准每個變更請求。接下來，IAM 角色 `Admin` 的任何三個成員必須核准變更請求。
   > 如需有關變更範本核准的詳細資訊，請參閱關於變更範本中的核准。

```
{
   "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. 執行以下命令來建立變更範本。

   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"
```

如需您可以指定之相關選項的資訊，請參閱 create-document。

系統會傳回相關資訊，如下所示。

```
{
   "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":[

      ]
   }
}
```

您組織或帳戶中已在 Change Manager 中的 Settings (設定) 標籤上指定為範本檢閱者的使用者會收到通知：新的變更範本正待其檢閱。

如果已為變更範本指定 Amazon Simple Notification Service (Amazon SNS) 主題，則會在變更範本遭到拒絕或獲得核准時傳送通知。如果您沒有收到與此變更範本相關的通知，您可以稍後返回 Change Manager，以檢查其狀態。

檢閱及核准或拒絕變更範本

如果您被指定為 中變更範本的檢閱者Change Manager、 中的工具 AWS Systems Manager、當新的變更範本或新版本的變更範本正在等待檢閱時，您會收到通知。Amazon Simple Notification Service (Amazon SNS) 主題會傳送通知。

> **Note**
>
> 此功能取決於您的帳戶是否已設定為使用 Amazon SNS 主題來傳送變更範本檢閱通知。如需有關指定範本檢閱者通知主題的資訊，請參閱 任務 1：設定 Change Manager 使用者身分識別管理和範本檢閱者。

若要檢閱變更範本，請遵循通知中的連結、登入 AWS Management Console，並遵循此程序中的步驟。

若要檢閱及核准或拒絕變更範本

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Change Manager。
3. 在 Overview (概觀) 標籤底部的 Change templates (變更範本) 區段中，選擇 Pending review (有待檢閱) 中的數字。
4. 在 Change templates (變更範本) 清單中，找到並選擇要檢閱的變更範本的名稱。
5. 在摘要頁面中，檢閱變更範本的提議內容，然後執行下列其中一項動作：
   - 若要核准允許在變更請求中使用變更範本，請依次選擇 Approve (核准)。
   - 若要拒絕變更範本，以防止變更請求中使用變更範本，請選擇 Reject (拒絕)。

刪除變更範本

本主題說明如何刪除您在 中建立的範本Change Manager，這是 Systems Manager 中的工具。如果您正在使用組織的 Change Manager，則會在您的委派管理員帳戶中執行此處理程序。

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Change Manager。
3. 選擇 Templates (範本) 標籤。
4. 選擇要刪除的範本名稱。
5. 選擇 Actions, Delete template (動作、刪除範本)。
6. 在確認對話方塊中輸入 **DELETE**，然後選擇 Delete (刪除)。

使用變更請求

變更請求是在 中執行 Automation Runbook Change Manager的請求，該自動化 Runbook 會更新您或 AWS 內部部署環境中的一或多個資源。變更請求是使用變更範本建立的。

當您在 中建立變更請求時Change Manager，在 中的工具 AWS Systems Manager，組織中或帳戶中的一或多個核准者必須檢閱和核准請求。如果沒有所需的核准，則不允許執行會變更您請求的 Runbook 工作流程。

主題

- 建立變更請求
- 檢閱及核准或拒絕變更請求

建立變更請求

當您在 中建立變更請求Change Manager，即 中的工具時 AWS Systems Manager，您選取的變更範本通常會執行下列動作：

- 指定變更請求的核准者，或指定需要多少次核准
- 指定 Amazon Simple Notification Service (Amazon SNS) 主題，以便通知核准者有關變更請求的資訊
- 指定 Amazon CloudWatch 警示，以監控變更請求的 Runbook 工作流程
- 識別您可以選擇哪些自動化 Runbook 來進行請求的變更

在某些情況下，可能會設定變更範本，因此您可以指定要使用的自動化 Runbook，並指定應該檢閱和核准請求的人員。

> **Important**
>
> 如果您在整個組織中使用 Change Manager，我們建議始終從委派管理員帳戶進行變更。雖然您可以從組織中的其他帳戶進行變更，但這些變更將不會在受委派管理員帳戶中報告，也不可在其中檢視。

主題

- 關於變更請求核准

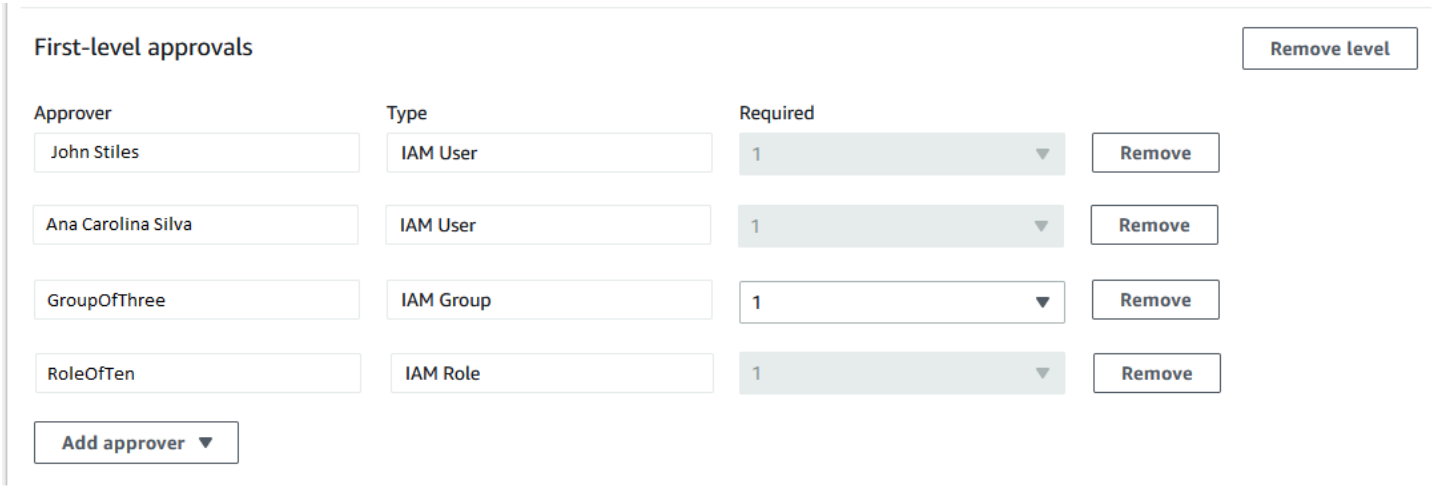
- 建立變更請求 (主控台)
- 建立變更請求 (AWS CLI)

關於變更請求核准

視變更範本中指定的要求而定，您建立的變更請求最多可以設定五個層級的核准，獲得這些層級的核准後，請求的執行手冊工作流程才能進行。對於每個層級，範本建立者最多可以指定五個潛在核准者。核准者不限於單一使用者。從這個意義上講，核准者也可以是 IAM 群組或 IAM 角色。對於 IAM 群組和 IAM 角色，屬於該群組或角色的一或多名使用者可以提供核准，以獲得變更請求所需的核准總數。範本建立者指定的核准者數量可以比變更範本要求的更多。

原始的核准工作流程和/或更新的核准工作流程

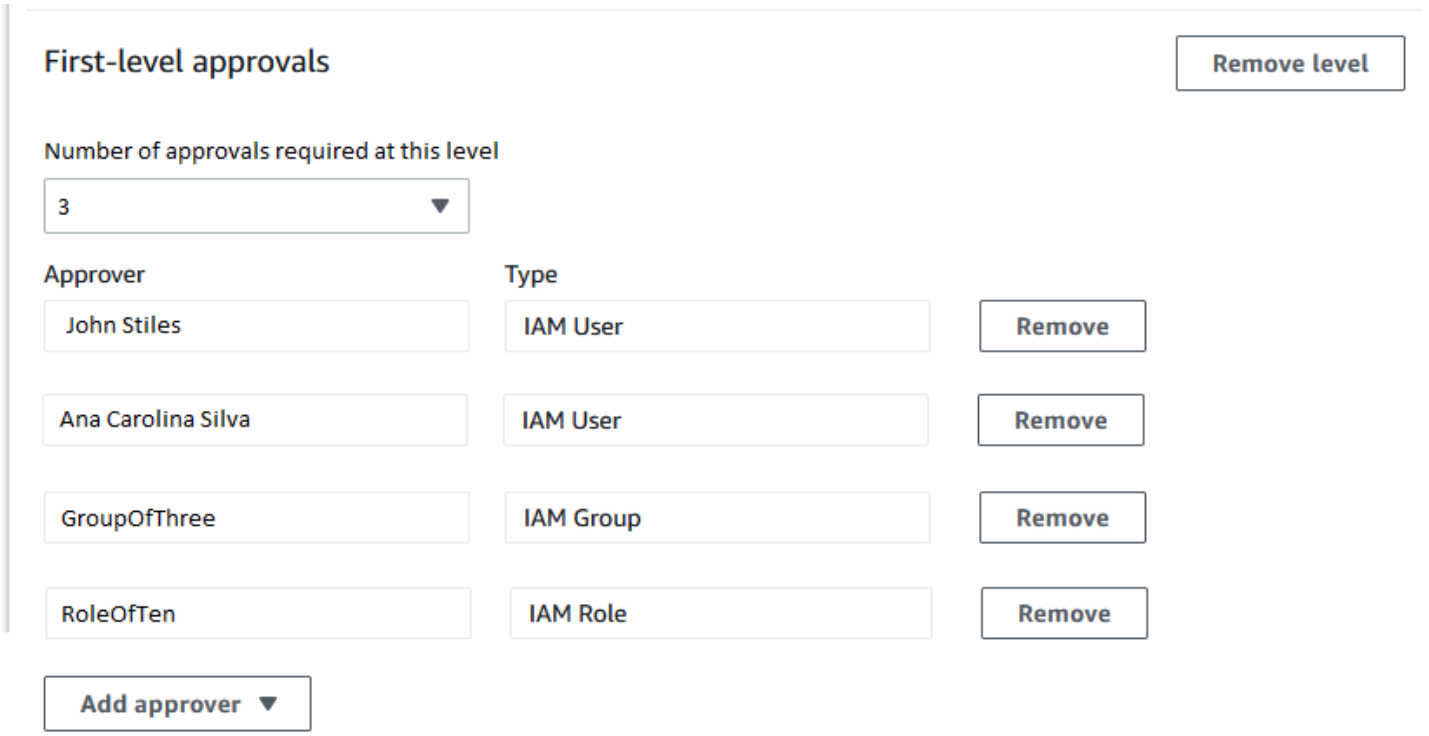
使用 2023 年 1 月 23 日之前建立的變更範本時，必須獲得每個指定核准者的核准，變更請求才能在該層級獲得核准。例如，在以下影像中顯示的核准層級設定中，指定了四個核准者。指定的核准者包括兩名使用者 (John Stiles 和 Ana Carolina Silva)、一個包含三名成員的使用者群組 (GrouPofThree)，以及代表十名使用者的使用者角色 (RoleOfTen)。



變更請求若要在此層級獲得核准，必須獲得 John Stiles、Ana Carolina Silva、`GroupOfThree` 群組成員之一，以及 `RoleOfTen` 角色的一名成員的核准。

如果使用在 2023 年 1 月 23 日當日或之後建立的變更範本，範本建立者可以針對每個核准層級指定所需的核准總數。這些核准可來自指定為核准者的任何使用者、群組和角色組合。變更範本可以設定在一個層級只需要一個核准，但可以指定兩名單獨的使用者、兩個群組和一個角色作為潛在核准者。

例如，在下圖所示的核准層級區域中，需要三個核准。範本指定的核准者包括兩名使用者 (John Stiles 和 Ana Carolina Silva)、一個包含三名成員的使用者群組 (`GroupOfThree`)，以及代表十名使用者的使用者角色 (`RoleOfTen`)。



如果 `GroupOfThree` 群組中的三名使用者都核准了您的變更請求，則意味著變更請求在該層級已獲得核准。不需要獲得每個使用者、群組或角色的核准。最小數量的核准可以來自任何潛在核准者的組合。

建立變更請求後，系統會將通知傳送給指定為接收該層級核准通知的 Amazon SNS 主題的訂閱用戶。變更範本建立者可能已指定必須使用的通知主題或允許您進行指定。

在某個層級獲得的核准數量達到最低數量之後，系統會將通知傳送給下一層級核准通知的 Amazon SNS 主題的訂閱用戶 (核准者)，依此類推。

無論指定多少核准層級和核准者，只要變更請求遭到一個拒絕，就能阻止該請求的執行手冊工作流程發生。

建立變更請求 (主控台)

下列處理程序說明如何使用 Systems Manager 主控台來建立變更請求。

若要建立變更請求 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Change Manager。
3. 選擇 Create request (建立請求)。

4. 搜尋並選取您要用於此變更請求的變更範本。
5. 選擇 Next (下一步)。
6. 對於 Name (名稱)，輸入可輕鬆識別用途的變更請求名稱，例如 **`UpdateEC2LinuxAMI-us-east-2`**。
7. 對於 Runbook (Runbook)，選取您要用來進行請求的變更的 Runbook。

   > **Note**
   >
   > 如果選取 Runbook 的選項不可用，則變更範本作者已指定必須使用的 Runbook。

8. 對於 Change request information (變更請求資訊)，請使用 Markdown 提供有關變更請求的其他資訊，以協助檢閱者決定是否要核准或拒絕變更請求。您使用的範本作者可能已提供指示或供您回答的問題。

   > **Note**
   >
   > Markdown 是一種標示語言，可讓您新增維基樣式的描述至文件內，以及在文件內新增個別步驟。如需使用 Markdown 的相關資訊，請參閱在 AWS中使用 Markdown。

9. 在 Workflow start time (工作流程開始時間) 區段，選擇以下其中一項：

   - 在排定的時間執行作業 – 對於 Requested start time (請求的開始時間)，輸入您提議執行此請求之 Runbook 工作流程的日期和時間。對於 Estimated end time (預測結束時間)，輸入您預期完成 Runbook 工作流程的日期和時間。(該時間只是您提供給檢閱者的預測值)。

     > **Tip**
     >
     > 選擇 View Change Calendar (檢視變更行事曆)，檢查您指定的時間是否有任何封鎖事件。

   - 在核准之後，僅可執行操作 – 如果已核准變更請求，Runbook 工作流程會在可進行變更的非限制時期內立即執行。

10. 在 Change request approvals (變更請求核准) 區段，執行下列動作：

    1. 如果出現 Approval type (核准類型)，請選擇以下其中一個選項：

       - Automatic approval (自動核准) – 您選取的變更範本設定為允許自動執行變更請求，而不需經任何核准者檢閱。繼續步驟 11。

> **Note**
>
> 控管您使用 Systems Manager 的 IAM 政策中指定的許可，不得限制您提交自動核准變更請求，以便自動執行。

- Specify approvers (指定核准者) – 您必須新增一個或多個使用者、群組或 IAM 角色，才能檢閱並核准此變更請求。

> **Note**
>
> 即使控管 Systems Manager 使用的 IAM 政策中指定的許可允許您執行自動核准變更請求，您也可以選擇指定檢閱者。

2. 選擇新增核准者，然後從可用的檢閱者清單中選取一或多個使用者、群組或 AWS Identity and Access Management (IAM) 角色。

> **Note**
>
> 可能已經指定一個或多個核准者。這表示已在您選取的變更範本中指定強制核准者。無法從請求中移除這些核准者。如果無法使用新增核准者按鈕，則表示您選擇的範本不允許將其他檢閱者新增至請求。

如需有關變更請求核准的詳細資訊，請參閱關於變更請求核准。

3. 在 SNS topic to notify approvers (通知核准者的 SNS 主題) 下，選擇以下其中一項，以在您的帳戶中指定 Amazon SNS 主題，進而用於向您新增到此變更請求的核准者傳送通知。

> **Note**
>
> 如果指定 Amazon SNS 主題的選項不可用，則您選取的變更範本已指定要使用的 Amazon SNS 主題。

- Enter an SNS Amazon Resource Name (ARN) (輸入 SNS Amazon 資源名稱 (ARN)) – 對於 Topic ARN (主題 ARN)，輸入現有 Amazon SNS 主題的 ARN。此主題可以位於您組織的任何帳戶中。

- Select an existing SNS topic (選取現有的 SNS 主題) – 對於 Target notification topic (目標通知主題) 中，選取您目前帳戶中的現有 Amazon SNS 主題的 ARN。（如果您尚未在目前的 AWS 帳戶 和 中建立任何 Amazon SNS 主題，則此選項無法使用 AWS 區域。）

> **Note**
>
> 您選取的 Amazon SNS 主題必須設定為指定其傳送的通知以及要傳送的訂閱者。其存取政策也必須將許可授予 Systems Manager，以便 Change Manager 可以傳送通知。如需相關資訊，請參閱 為 Change Manager 通知設定 Amazon SNS 主題。

4. 選擇 Add notification (新增通知)。

11. 選擇 Next (下一步)。
12. 對於 IAM role (IAM 角色)，在您的目前帳戶中，選取 IAM 角色，而該角色具有執行為此變更請求指定的 Runbook 所需的許可。

    此角色也稱為自動化的服務角色，或擔任角色。如需有關此角色的詳細資訊，請參閱 設定自動化。
13. 在 Deployment location (部署位置) 區段中，選擇下列其中一個選項：

> **Note**
>
> 如果您使用 時 AWS 帳戶 只Change Manager搭配單一 ，而不是搭配 中設定的組織 AWS Organizations，則不需要指定部署位置。

- 將變更套用至此帳戶 – Runbook 工作流程僅在目前帳戶中執行。對於組織而言，這表示委派管理員帳戶。
- Apply change to multiple organizational units (OUs) (將變更套用至多個組織單位 (OU)) – 執行下列動作：
  1. 對於 Accounts and organizational units (OUs) (帳戶和組織單位 (OU))，輸入組織中成員帳戶的 ID (格式為 **123456789012**)，或組織單位的 ID (格式為 **o-o96EXAMPLE**)。
  2. (選用) Execution role name (執行角色名稱)，在目標帳戶或具有執行為此變更要求指定的 Runbook 所需許可的 OU 中，輸入 IAM 角色的名稱。您指定之任何 OU 中的所有帳戶都應該對此角色使用相同的名稱。
  3. (選用) 為要指定的每個其他帳戶或 OU 選擇 Add another target location (新增另一個目標位置)，然後重複步驟 a 和 b。

4. 針對目標 AWS 區域，選取要進行變更的區域，例如Ohio (us-east-2)美國東部（俄亥俄）區域。
5. 展開 Rate control (速率控制)。

   對於 Concurrency (並行)，輸入數字，然後從清單中選取這是否代表 Runbook 工作流程可同時執行的帳戶數目或百分比。

   對於 Error threshold (錯誤閾值)，輸入數字，然後從清單中選取這是否代表 Runbook 工作流程在停止操作之前可能會失敗的帳戶數目或百分比。

14. 在 Deployment targets (部署目標) 區段中，執行下列動作：

    1. 選擇下列其中一項：

       - Single resource (單一資源) – 只針對一個資源進行變更。例如，單一節點或單一 Amazon Machine Image (AMI)，視此變更請求的 Runbook 中定義的操作而定。
       - Multiple resources (多個資源) – 對於 Parameter (參數)，從 Runbook 中為此變更請求選取可用參數。此選項會反映正在更新的資源類型。

         例如，如果此變更請求的 Runbook 是 `AWS-RetartEC2Instance`，您可能選擇 `InstanceId`，然後從下列選項中進行選取，定義要更新的執行個體：

         - Specify tags (指定標籤) – 輸入鍵值對，其要更新的所有資源都應使用此鍵值對標記。
         - Choose a resource group (選擇資源群組) – 選擇要更新之所有資源所屬的資源群組名稱。
         - Specify parameter values (指定參數值) – 識別要在 Runbook parameters (Runbook 參數) 區段中更新的資源。
         - Target all instances (將所有執行個體設為目標) – 對目標位置中的所有受管節點進行變更。

    2. 如果您選擇 Multiple resources (多個資源)，展開 Rate control (速率控制)。

       對於 Concurrency (並行)，輸入數字，然後從清單中選取這是否代表 Runbook 工作流程可同時更新的目標數目或百分比。

       對於 Error threshold (錯誤閾值)，輸入數字，然後從清單中選取這是否代表更新在停止操作之前可能會失敗的目標數目或百分比。

15. 如果您選擇 Specify parameter values (指定參數值) 來更新上一個步驟中的多個資源：在 Runbook parameters (Runbook 參數) 區段中，指定所需輸入參數的值。您必須提供的參數值是以與您選擇的變更範本關聯的 Automation Runbook 內容為基礎。

例如，如果變更範本使用 AWS-RetartEC2Instance Runbook，則必須為 InstanceId 參數輸入一個或多個執行個體 ID。或者，選擇 Show interactive instance picker (顯示互動式執行個體選擇器)，然後逐一選取可用的執行個體。

16. 選擇 Next (下一步)。
17. 在 Review and submit (檢閱及提交) 頁面上，再次檢查您為此變更請求指定的資源和選項。

    對於任何您想要變更的區段，選擇 Edit (編輯) 按鈕。

    當您對變更請求的詳細資訊感到滿意時，請選擇 Submit for approval (提交核准)。

如果已為所選請求的變更範本指定 Amazon SNS 主題，則會在請求遭到拒絕或獲得核准時傳送通知。如果您沒有收到請求的通知，您可以返回 Change Manager，檢查請求的狀態。

建立變更請求 (AWS CLI)

您可以使用 AWS Command Line Interface (AWS CLI) 建立變更請求，方法是在 JSON 檔案中指定變更請求的選項和參數，並使用 `--cli-input-json`選項將其包含在命令中。

若要建立變更請求 (AWS CLI)

1. AWS Tools for PowerShell如果您尚未安裝和設定 AWS CLI 或 。

   如需相關資訊，請參閱安裝或更新 AWS CLI的最新版本和安裝 AWS Tools for PowerShell。
2. 在您的本機電腦建立名稱如 `MyChangeRequest.json` 的 JSON 檔案，然後將以下內容貼至其中。

   將####取代為變更請求的值。

   > **Note**
   >
   > 此範例 JSON 會使用 `AWS-HelloWorldChangeTemplate` 變更範本和 `AWS-HelloWorld` Runbook 建立變更請求。若要針對自己的變更請求調整此範例，請參閱《AWS Systems Manager API 參考》中的 StartChangeRequestExecution 一節，以獲取所有可用參數的相關資訊
   > 如需有關變更請求核准的詳細資訊，請參閱關於變更請求核准。

```
{
```

```
    "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. 在您建立 JSON 檔案的目錄裡執行下列命令。

```
aws ssm start-change-request-execution --cli-input-json file://MyChangeRequest.json
```

系統會傳回相關資訊，如下所示。

```
{
    "AutomationExecutionId": "b3c1357a-5756-4839-8617-2d2a4EXAMPLE"
}
```

檢閱及核准或拒絕變更請求

如果您被指定為 中變更請求的檢閱者Change Manager，這是 中的工具 AWS Systems Manager，當新的變更請求正在等待您的檢閱時，會透過 Amazon Simple Notification Service (Amazon SNS) 主題通知您。

> **Note**
>
> 此功能取決於變更範本中是否指定了 Amazon SNS 來傳送檢閱通知。如需相關資訊，請參閱為 Change Manager 通知設定 Amazon SNS 主題。

若要檢閱變更請求，您可以遵循通知中的連結，或 AWS Management Console 直接登入 ，並遵循此程序中的步驟。

> **Note**
>
> 如果將 Amazon SNS 主題指派給變更範本中的檢閱者，則當變更請求狀態變更時，會將通知傳送給該主題的訂閱者。
> 如需有關變更請求核准的詳細資訊，請參閱關於變更請求核准。

檢閱及核准或拒絕變更請求 (主控台)

以下程序說明如何使用 Systems Manager 主控台來檢閱及核准或拒絕變更請求。

若要檢閱及核准或拒絕單一變更請求

1. 開啟您收到的電子郵件通知中的連結，並登入 AWS Management Console，這會引導您前往變更請求以供檢閱。
2. 在摘要頁面中，檢閱變更請求的提議內容。

   若要核准變更請求，請選擇 Approve (核准)。在對話方塊中，提供您要為此核准新增的任何註解，然後選擇 Approve (核准)。此請求所代表的 Runbook 工作流程會在排程時開始執行，或只要變更未被任何限制封鎖即可。

   -或-

   若要拒絕變更請求，請選擇 Reject (拒絕)。在對話方塊中，提供您要為此拒絕新增的任何註解，然後選擇 Reject (拒絕)。

若要檢閱及核准或拒絕大量變更請求

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Change Manager。
3. 選擇 Approvals (核准) 標籤。
4. (選用) 選擇各項請求的名稱即可檢閱待核准之請求的詳細資料，然後返回 Approvals (核准) 標籤。
5. 選取您要核准之每項變更請求的核取方塊。

   -或-

   選取您要拒絕之每項變更請求的核取方塊。
6. 在對話方塊中，提供您要為此核准或拒絕新增的任何註解。
7. 視您要核准或拒絕選取的變更請求而定，選擇 Approve (核准) 或 Reject (拒絕)。

檢閱及核准或拒絕變更請求 (命令列)

下列程序說明如何使用 AWS Command Line Interface (AWS CLI) （ 在 Linux、 macOS或 Windows 上 ） 來檢閱和核准或拒絕變更請求。

若要檢閱及核准或拒絕變更請求

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

2. 在本機機器上建立 JSON 檔案，指定 AWS CLI 呼叫的參數。

```
{
  "OpsItemFilters":
  [
    {
      "Key": "OpsItemType",
      "Values": ["/aws/changerequest"],
      "Operator": "Equal"
    }
  ],
  "MaxResults": number
}
```

您可以在 JSON 檔案中指定核准者的 Amazon Resource Name (ARN)，以篩選特定核准者的結果。請見此處範例。

```
{
  "OpsItemFilters":
  [
    {
      "Key": "OpsItemType",
      "Values": ["/aws/changerequest"],
      "Operator": "Equal"
    },
    {
      "Key": "ChangeRequestByApproverArn",
      "Values": ["arn:aws:iam::account-id:user/user-name"],
      "Operator": "Equal"
    }
  ],
  "MaxResults": number
}
```

3. 執行以下命令，以檢視您在 JSON 檔案中指定的變更請求數目上限。

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. 執行以下命令，以核准或拒絕變更請求。

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"
```

如果已為所選請求的變更範本指定 Amazon SNS 主題，則會在請求遭到拒絕或獲得核准時傳送通知。如果您沒有收到請求的通知，您可以返回 Change Manager，檢查請求的狀態。如需有關使用此命令時可搭配使用的其他選項的資訊，請參閱《AWS CLI 命令參考》中 AWS Systems Manager 一節的 send-automation-signal。

檢閱變更請求詳細資訊、任務和時間表 (主控台)

您可以在 的儀表板中檢視變更請求的相關資訊，包括已處理變更的請求Change Manager，也就是 中的工具 AWS Systems Manager。這些詳細資訊包括執行 Runbook 進行變更之 Automation 操作的連結。建立請求時會產生自動化執行 ID，但是在指定所有核准且沒有限制來封鎖變更之前，程序才會執行。

若要檢閱變更請求詳細資訊、任務和時間表

1. 在導覽窗格中，選擇 Change Manager。
2. 選擇 Requests (請求) 標籤。

3. 在 Change requests (變更請求) 區段中，搜尋您要檢閱的變更請求。

   您可以使用 Create date range (建立日期範圍) 選項，將結果限制在特定期間內。

   您可以依下列屬性篩選請求：

   - `Status`
   - `Request ID`
   - `Approver`
   - `Requester`

   例如，若要檢視過去 24 小時內成功完成之所有變更請求的詳細資訊，請執行下列動作：

   1. 對於 Create date range (建立日期範圍)，選擇 1d。
   2. 在搜尋方塊中，選取 Status, CompletedWithSuccess (狀態 CompletedWithSuccess)。
   3. 在結果中，選擇成功完成的變更請求的名稱，以檢閱其結果。

4. 在下列標籤上檢視變更請求的相關資訊：

   - Request details (請求詳細資訊) – 檢視有關變更請求的基本詳細資訊，包括申請者、變更範本，以及為變更選取的 Automation Runbook。您也可以遵循 Automation 操作詳細資訊的連結，檢視有關請求中指定的任何 Runbook 參數、指派給變更請求的 Amazon CloudWatch 警示，以及針對該請求提供的核准和註解的資訊。
   - Task (任務) – 檢視變更中有關任務的資訊，包括已完成變更請求的任務狀態、目標資源、關聯的自動化 Runbook 中的步驟，以及並行和錯誤閾值詳細資訊。
   - Timeline (時間表) – 檢視與變更請求關聯之所有事件的摘要，按照日期和時間列出。摘要會指出建立變更請求的時間、指派核准者的動作、排定執行核准變更請求的時間備註、Runbook 工作流程詳細資訊，以及整體變更流程和 Runbook 中每個步驟的狀態變更。
   - Associated events (關聯事件) – 檢視有關 AWS CloudTrail Lake 中記錄之變更請求的可稽核詳細資訊。詳細資訊包括執行的 API 動作、針對這些動作所包含的請求參數、執行動作的使用者帳戶、在處理期間更新的資源等等。

     當您啟用 CloudTrail Lake 事件追蹤時，CloudTrail Lake 會針對變更請求相關事件建立事件資料存放區。事件詳細資訊可用於執行變更請求的帳戶或組織。您可以透過帳戶或組織中的任何變更請求來開啟 CloudTrail Lake 事件追蹤。如需有關啟用 CloudTrail Lake 整合以及建立事件資料存放區的資訊，請參閱 監控您的變更請求事件。

> **Note**
>
> 使用 CloudTrail Lake 需要支付費用。如需詳細資訊，請參閱 AWS CloudTrail 定價。

檢視變更請求的彙總計數 (命令列)

您可以使用 GetOpsSummary API 操作 AWS Systems Manager，在 中的Change Manager工具 中檢視變更請求的彙總計數。此 API 操作可以傳回單一 AWS 帳戶 帳戶或 AWS 區域 多個帳戶和多個區域中的單一 計數。

> **Note**
>
> 如果您想要檢視多個 AWS 帳戶 和多個變更請求的彙總計數 AWS 區域，您必須設定和設定資源資料同步。如需詳細資訊，請參閱建立庫存的資源資料同步。

下列程序說明如何使用 AWS Command Line Interface (AWS CLI)（在 Linux、 macOS或 Windows 上）來檢視變更請求的彙總計數。

若有檢視變更請求的彙總計數

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。
2. 請執行下列其中一個命令：

   單一帳戶和區域

   此命令會傳回設定 AWS CLI 工作階段之 AWS 帳戶 和 AWS 區域 的所有變更請求計數。

   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
```

呼叫會傳回相關資訊，如下所示。

```
{
    "Entities": [
        {
            "Data": {
                "AWS:OpsItem": {
                    "Content": [
                        {
                            "Count": "38",
                            "Status": "Open"
                        }
                    ]
                }
            }
        }
    ]
}
```

多個帳戶及/或區域

此命令會傳回 資源資料同步中 AWS 區域 指定的 AWS 帳戶 和 的所有變更請求計數。

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
```

呼叫會傳回相關資訊，如下所示。

```
{
    "Entities": [
        {
            "Data": {
                "AWS:OpsItem": {
                    "Content": [
                        {
                            "Count": "43",
                            "Status": "Open"
                        },
                        {
                            "Count": "2",
                            "Status": "Resolved"
                        }
                    ]
                }
            }
        }
    ]
}
```

多個帳戶及特定區域

此命令會傳回針對資源資料同步中指定的 AWS 帳戶 的所有變更請求的計數。但是，它只會傳回來自命令中指定的區域的資料

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
```

多個帳戶和區域 (具有按區域分組的輸出)

此命令會傳回 資源資料同步中 AWS 區域 指定的 AWS 帳戶 和 的所有變更請求計數。輸出會顯示每個區域的計數資訊。

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"}]}]'
```

呼叫會傳回相關資訊，如下所示。

```
{
        "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"
                            }
                        ]
                    }
                }
            }
        ]
    }
```

多個帳戶和區域 (具有按區域分組的輸出)

此命令會傳回 資源資料同步中 AWS 區域 指定的 AWS 帳戶 和 的所有變更請求計數。輸出按帳戶和區域分組計數資訊。

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"}]}]}]
```

呼叫會傳回相關資訊，如下所示。

```
{
    "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"
                    }
                ]
            }
        }
    ]
}
```

稽核和記錄 Change Manager 活動

您可以使用 Amazon CloudWatch 和 AWS CloudTrail 警示 AWS Systems Manager，稽核 中的 Change Manager工具 中的活動。

如需稽核和記錄 Systems Manager 選項的詳細資訊，請參閱 在 中記錄和監控 AWS Systems Manager。

使用 CloudWatch 警示稽核 Change Manager 活動。

您可以設定 CloudWatch 警示並將其指派給變更範本。如果符合警示中定義的任何條件，則會採取針對警示指定的動作。在警示組態中，您可以指定 Amazon Simple Notification Service (Amazon SNS) 主題，以便在符合警示條件時進行通知。

如需有關建立 Change Manager 範本的資訊，請參閱 使用變更範本。

如需有關建立 CloudWatch 警示的資訊，請參閱《Amazon CloudWatch 使用者指南》中的使用 CloudWatch 警示。

使用 CloudTrail 稽核 Change Manager 活動

CloudTrail 會擷取 Systems Manager 主控台、 AWS Command Line Interface (AWS CLI) 和 Systems Manager SDK 中發出的 API 呼叫。您可以在 CloudTrail 主控台中，或在存放資訊的 Amazon Simple Storage Service (Amazon S3) 儲存貯體中檢視資訊。一個儲存貯體可用於您帳戶中所有 CloudTrail 日誌。

Change Manager 動作的日誌會顯示變更範本文件建立、變更範本和變更請求核准和拒絕、自動化 Runbook 產生的活動等等。如需檢視和使用 Systems Manager 活動的 CloudTrail 日誌的詳細資訊，請參閱 使用 記錄 AWS Systems Manager API 呼叫 AWS CloudTrail。

Change Manager 疑難排解

使用下列資訊來協助您疑難排解 中的Change Manager工具 的問題 AWS Systems Manager。

主題

- 使用 Active Directory (群組) 時變更請求核准期間出現 “Group {GUID} not found” (找不到群組 {GUID}) 錯誤。

使用 Active Directory (群組) 時變更請求核准期間出現 “Group ***{GUID}*** not found” (找不到群組 {GUID}) 錯誤。

問題：當 AWS IAM Identity Center (IAM Identity Center) 用於使用者身分管理時，在 中授予核准許可的 Active Directory 群組成員Change Manager會收到「未授權」或「找不到群組」錯誤。

- 解決方案：當您在 IAM Identity Center 中選取 Active Directory 群組以存取 時 AWS Management Console，系統會排程定期同步，將來自這些 Active Directory 群組的資訊複製到 IAM Identity Center。在透過 Active Directory 群組成員資格授權的使用者可以成功核准請求之前，必須先完成此程序。如需詳細資訊，請參閱《AWS IAM Identity Center 使用者指南》中的連接到您的 Microsoft AD 目錄。

AWS Systems Manager Documents

AWS Systems Manager 文件 (SSM 文件） 會定義 Systems Manager 在受管執行個體上執行的動作。Systems Manager 包含 100 多個預先設定的文件，可讓您用來在執行時間時指定參數。透過選擇 Owned by Amazon (Amazon 擁有) 索引標籤，或者在呼叫 `ListDocuments` API 操作時為 `Owner` 篩選條件指定 Amazon，您可以在 Systems Manager 文件主控台中找到預先設定的文件。文件使用 JavaScript 物件標記法 (JSON) 或 YAML，其中包括您指定的步驟和參數。若要開始使用 SSM 文件，請開啟 Systems Manager 主控台。在導覽窗格中，選擇 Documents (文件)。

> **Important**
>
> 在 Systems Manager 中，Amazon 擁有的 SSM 文件是由 Amazon Web Services 本身建立和管理的文件。Amazon 擁有的文件的文件名中包含如 `AWS-*` 的字首。文件的擁有者被視為 Amazon，而不是其中的特定使用者帳戶 AWS。這些文件公開供所有人使用。

文件工具如何使我的組織受益？

文件是 中的工具 AWS Systems Manager，可提供下列優點：

- 文件類別

  為了幫助您找到所需文件，請根據要搜尋的文件類型選擇類別。若要擴大搜尋範圍，您可以選擇同一文件類型的多個類別。不支援選擇不同文件類型的類別。僅支援 Amazon 擁有的文件類別。

- 文件版本

  您可以建立和儲存不同版本的文件。然後，您可以為每個文件指定一個預設版本。預設版本的文件更新到較新版本或可恢復到舊版的文件。當您變更文件的內容時，Systems Manager 會自動增加文件版本。您可以在主控台、 AWS Command Line Interface (AWS CLI) 命令或 API 呼叫中指定文件版本，以擷取或使用文件的任何版本。

- 根據需求自訂文件

  如果您要在文件中自訂步驟和動作，您可以建立自己的步驟和動作。系統會將 文件與 存放在 AWS 區域 您建立文件的 AWS 帳戶 中。如需有關如何建立 SSM 文件的詳細資訊，請參閱 建立 SSM 文件內容。

- 標記文件

  您可以在文件加上標籤，根據您指派給文件的標籤可以協助您快速找出一或多個文件。例如，您可以為用定的環境、部門、使用者、群組或時段來標記文件。您也可以透過建立指定使用者或群組可存取之標籤的 AWS Identity and Access Management (IAM) 政策來限制文件的存取。

- 共用文件

  您可以將文件設定為公開或者與相同 AWS 區域中特定的 AWS 帳戶 分享。例如，若想讓您提供給客戶或員工的所有 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體都有相同的組態，在帳戶之間共用文件就非常實用。除了將執行個體上的應用程式或修補程式保持在最新狀態，您可能想要限制客戶執行個體避免進行特定的活動。或者，您可能想要確保在組織中員工帳戶所使用的執行個體獲得特定內部資源的存取權。如需詳細資訊，請參閱共用 SSM 文件。

誰應該使用 Documents？

- 任何想要使用 Systems Manager 工具來改善大規模營運效率、減少與手動介入相關的錯誤，以及縮短解決常見問題的時間 AWS 的客戶。
- 希望自動化部署和組態任務的基礎設施專家。
- 希望可靠地解決常見問題、提高疑難排解效率和減少重複性操作的管理員。
- 希望自動化通常手動執行之任務的使用者。

SSM 文件有哪些類型？

下表說明不同類型的 SSM 文件和其使用案例。

Type	搭配使用	詳細資訊
ApplicationConfiguration ApplicationConfigurationSchema	AWS AppConfig	AWS AppConfig是 中的工具 AWS Systems Manager，可讓您建立、管理和快速部署應用程式組態。您可以藉由建立使用 `ApplicationConfiguration` 文件類型的文件，在 SSM 文件中存放組態資料。如需詳細資訊，請參閱《AWS AppConfig 使用者指南》中的 Freeform 組態。 如果您在 SSM 文件中建立組態，則必須指定相應的 JSON 結構描述。該結構描述使用 `ApplicationConfigurationSchema` 文件類型，且與一組規則一樣，會定義每個應用程式組態設定允許的屬性。如需詳細資訊，請參閱《AWS AppConfig 使用者指南》中的關於驗證器。

Type	搭配使用	詳細資訊
Automation Runbook	自動化 State Manager Maintenance Windows	在執行常見的維護和部署任務 (例如建立或更新 Amazon Machine Image (AMI)) 時使用 Automation Runbook。State Manager 會使用 Automation Runbook 來套用組態。您可以在執行個體生命週期期間的任何時間點對一個或多個目標執行這些動作。Maintenance Windows 會使用 Automation Runbook，根據指定的排程執行常見的維護和部署任務。 基於 Linux 的作業系統支援的所有 Automation Runbook 在 macOS 的 EC2 執行個體上也受支援。
變更行事曆文件	Change Calendar	Change Calendar中的工具 AWS Systems Manager使用 `ChangeCalendar` 文件類型。Change Calendar 文件會存放行事曆項目和相關聯事件，這些事件可允許或防止自動化動作變更您的環境。在 Change Calendar 中，文件會以純文字格式存放 iCalendar 2.0 資料。 macOS 的 EC2 執行個體不支援 Change Calendar。

Type	搭配使用	詳細資訊
AWS CloudFormation 範本	AWS CloudFormation	AWS CloudFormation 範本說明您想要在 CloudFormation 堆疊中佈建的資源。透過將 CloudFormation 範本存放為 Systems Manager 文件，可讓您從 Systems Manager 文件功能中受益。其中包括建立和比較範本的多個版本，以及與相同 AWS 區域中的其他帳戶共用範本。 您可以使用 Systems Manager 中的工具 Application Manager 建立和編輯 CloudFormation 範本和堆疊。如需詳細資訊，請參閱在 中使用 AWS CloudFormation 範本和堆疊 Application Manager。

Type	搭配使用	詳細資訊
指令文件	Run Command State Manager Maintenance Windows	Run Command中的工具 AWS Systems Manager使用命令文件來執行命令。State Manager中的工具 AWS Systems Manager使用命令文件來套用組態。這些動作可以在執行個體生命週期的任何時候於一或多個目標上執行。Maintenance Windows中的工具 AWS Systems Manager會使用 命令文件，根據指定的排程來套用組態。 Systems Manager 支援的所有 Linux 和 Windows Server 作業系統支援大部分命令文件。macOS 的 EC2 執行個體支援下列命令文件： • `AWS-ConfigureAWSPackage` • `AWS-RunPatchBaseline` • `AWS-RunPatchBaselineAssociation` • `AWS-RunShellScript`
AWS Config 一致性套件範本	AWS Config	AWS Config 一致性套件範本是 YAML 格式的文件，用於建立一致性套件，其中包含受管或自訂規則和修補動作的 AWS Config 清單。 如需詳細資訊，請參閱一致性套件。

Type	搭配使用	詳細資訊
套件文件	Distributor	在 中的Distributor工具中 AWS Systems Manager，套件由 SSM 文件表示。套件文件包含 ZIP 封存檔案，封存檔包含要安裝在受管執行個體上的軟體或資產。在 Distributor 中建立套件會建立套件文件。 Oracle Linux 和 macOS 受管執行個體不支援 Distributor。
政策文件	State Manager	中的庫存工具 AWS Systems Manager會使用`AWS-GatherSoftwareInventory` 政策文件與State Manager關聯，從受管執行個體收集庫存資料。建立您自己的 SSM 文件時，Automation Runbook 和命令文件是在受管執行個體上強制執行政策的慣用方法。 Systems Manager 支援的所有作業系統都支援 Systems Manager Inventory 和 `AWS-GatherSoftwareInventory` 政策文件。
事件後分析範本	Incident Manager 事件後分析	Incident Manager 使用事件後分析範本，根據 AWS 操作管理最佳實務建立分析。 使用範本建立的分析可供團隊用來找出事件回應的改進。

Type	搭配使用	詳細資訊
工作階段文件	Session Manager	Session Manager中的工具 AWS Systems Manager使用工作階段文件來決定要啟動的工作階段類型，例如連接埠轉送工作階段、執行互動式命令的工作階段，或建立 SSH 通道的工作階段。 Systems Manager 支援的所有 Linux 和 Windows Server 作業系統支援工作階段文件。macOS 的 EC2 執行個體支援下列命令文件： • AWS-PasswordReset • AWS-StartInteractiveCommand • AWS-StartPortForwardingSession • AWS-StartPortForwardingSessionToSocket • AWS-StartSSHSession

SSM 文件配額

如需有關 SSM 文件配額的資訊，請參閱《Amazon Web Services 一般參考》中的 Systems Manager 服務配額一節。

主題

- 文件組成部分
- 建立 SSM 文件內容
- 使用文件

文件組成部分

本節包含有關 SSM 文件組成部分件的資訊。

目錄

- 結構描述、功能以及範例
- 資料元素和參數
- 命令文件外掛程式參考

結構描述、功能以及範例

AWS Systems Manager (SSM) 文件使用以下結構描述版本。

- Command 類型的文件可以使用結構描述 1.2、2.0 和 2.2 版本。如果您使用的文件為結構描述 1.2 版本，我們建議您使用結構描述 2.2 版本建立文件。
- Policy 類型的文件必須使用結構描述 2.0 版本或更新版本。
- Automation 類型的文件必須使用結構描述 0.3 版本。
- Session 類型的文件必須使用結構描述 1.0 版本。
- 您可以以 JSON 或 YAML 建立文件。

如需有關Session文件結構描述的詳細資訊，請參閱工作階段文件結構描述。

Command 和 Policy 文件使用最新的結構描述版本，您可以善用以下功能。

結構描述版本 2.2 文件的功能

功能	詳細資訊
文件編輯	文件現在可以進行更新。在版本 1.2 中，您需要將文件中任何更新儲存為不同名稱的文件。
自動版本控制	文件中任何更新建立新的版本。這並非結構描述的版本，而是文件的版本。
預設版本	如果您有多個版本的文件，您可以指定哪個版本為預設的文件。

功能	詳細資訊
定序	文件中中的外掛程式或步驟將按照您所指定的順序執行。
跨平台支援	跨平台支援允許您在相同的 SSM 文件中為不同的外掛程式指定不同的作業系統。跨平台支援使用 `precondition` 參數，只需要幾個步驟。

> **Note**
>
> 您必須讓執行個體保持 AWS Systems Manager SSM Agent更新至最新版本，才能使用新的 Systems Manager 功能和 SSM 文件功能。如需詳細資訊，請參閱使用 Run Command 更新 SSM Agent。

下表列出了主要的結構描述各版本的差異。

第 1.2 版	版本 2.2 (最新版本)	詳細資訊
runtimeConfig	mainSteps	在版本 2.2 中，`mainSteps` 區塊取代了 `runtimeConfig` 。`mainSteps` 區段可讓 Systems Manager 按順序執行步驟。
屬性	inputs	在版本 2.2 中，`inputs` 區塊取代了 `properties` 區塊。`inputs` 區塊接受了參數的做法步驟。
commands	runCommand	在版本 2.2 中，`inputs` 區塊接受 `runCommand` 的參數，而非 `commands` 的參數。
id	動作	在版本 2.2 中，`Action` 取代了 `ID`。這是名稱變更。

第 1.2 版	版本 2.2 (最新版本)	詳細資訊
不適用	name	在版本 2.2 中，name 是一個任何使用者替步驟定義的名稱。

使用 precondition 參數

使用結構描述 2.2 版或更新版本，您可以使用 precondition 參數為每個外掛程式指定目標作業系統，或驗證您在 SSM 文件中定義的輸入參數。precondition 參數支援引用 SSM 文件的輸入參數，以及使用值 Linux、MacOS 以及 Windows 的 platformType。只支援 StringEquals 運算子。

對於文件使用結構描述版本 2.2 或更新版本，如果未指定 precondition，每個外掛程式是根據外掛程式的相容性來決定在作業系統執行或略過。與作業系統的外掛程式相容性會在 precondition 之前評估。對於文件使用結構描述 2.0 或更舊版本，不相容的外掛程式會產生錯誤。

例如，在結構描述版本 2.2 文件中，如果未指定 precondition 但有列出aws:runShellScript 外掛程式，則在 Linux 執行個體上會執行該步驟，但在 Windows Server 執行個體上會略過該步驟，因為 aws:runShellScript 與 Windows Server 執行個體不相容。但就結構描述版本 2.0 文件來說，如果您指定 aws:runShellScript 外掛程式，然後在 Windows Server 執行個體上執行該文件，則執行會失敗。您可以在本節稍後的 SSM 文件中查看先決條件參數範例。

結構描述版本 2.2

頂層元素

以下範例顯示使用結構描述版本 2.2 的 SSM 文件上層元素。

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 }}"
         }
      }
   ]
}
```

結構描述版本 2.2 範例

下列範例會使用 aws:runPowerShellScript 外掛程式在目標執行個體上執行 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}}"
            ]
         }
      }
   ]
}
```

結構描述版本 2.2 precondition 參數範例

結構描述版本 2.2 提供跨平台支援。這表示您可以在同一個 SSM 文件中為不同的外掛程式指定不同的作業系統。跨平台支援在步驟中使用 `precondition` 參數，如下所示。您也可以使用 `precondition` 參數來驗證您在 SSM 文件中定義的輸入參數。您可以在以下第二個範例中看到它。

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"
            ]
         }
      }
   ]
}
```

結構描述版本 2.2 State Manager 範例

您可以使用下列 SSM 文件搭配 Systems Manager 中的工具 State Manager，下載並安裝 ClamAV 防毒軟體。 會State Manager強制執行特定組態，這表示每次執行State Manager關聯時，系統會檢查 ClamAV 軟體是否已安裝。如果不是，State Manager 會重新執行此文件。

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"
            ]
         }
      }
   ]
}
```

結構描述版本 2.2 庫存範例

您可以搭配使用以下 SSM 文件與 State Manager，收集有關執行個體的庫存中繼資料。

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 }}"
        }
     }
  ]
```

```
}
```

結構描述版本 2.2 **AWS-ConfigureAWSPackage** 範例

以下範例顯示 AWS-ConfigureAWSPackage 文件。mainSteps 區段包含 action 步驟中的 aws:configurePackage 外掛程式。

> **Note**
>
> 在 Linux 作業系統上，只有支援 AmazonCloudWatchAgent 和 AWSSupport-EC2Rescue 的套件。

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 }}"
         }
      }
   ]
}
```

結構描述版本 1.2

以下範例顯示結構描述版本 1.2 文件的上層元素。

```
{
   "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"
            }
         ]
      }
   }
}
```

結構描述版本 1.2 **aws:runShellScript** 範例

以下範例顯示 AWS-RunShellScript SSM 文件。runtimeConfig 區段包含 aws:runShellScript 外掛程式。

```
{
    "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 }}"
                }
            ]
        }
    }
}
```

結構描述版本 0.3

頂層元素

下列範例以 JSON 格式顯示結構描述 0.3 版 Automation Runbook 的最上層元素。

```
{
    "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 Automation Runbook 範例

下列範例以 YAML 格式顯示 Automation Runbook 的內容。文件結構描述的這份 0.3 版運作範例，也示範了如何使用 Markdown 來格式化文件描述。

```
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
```

資料元素和參數

本主題描述 SSM 文件中使用的資料元素。用於建立文件的結構描述版本會定義文件接受的語法和資料元素。建議命令文件使用結構描述版本 2.2 或更新版本。Automation Runbook 使用結構描述版本 0.3。此外，Automation Runbook 支援使用 Markdown (一種標示語言)，可讓您新增維基樣式的描述至文件內，以及在文件內新增個別步驟。如需關於使用 Markdown 的詳細資訊，請參閱《AWS Management Console 入門指南》中的在主控台中使用 Markdown。

下一節說明您可以在 SSM 文件中包含的資料元素。

頂層資料元素

schemaVersion

要使用的結構描述版本。

類型：版本

必要：是

description

您提供來描述文件用途的資訊。您也可以使用此欄位來指定參數是否需要執行文件的值，或者提供參數的值是否為選用項目。您可以在本主題的範例中看到必要參數和選用參數。

類型：字串

必要：否

parameters

一種結構，定義文件接受的參數。

對於您經常使用的參數，我們建議您將這些參數存放在 中Parameter Store，這是 中的工具 AWS Systems Manager。然後，您可以在文件中定義參考 Parameter Store 參數作為預設值的參數。若要參考 Parameter Store 參數，請使用下列語法。

```
{{ssm:parameter-name}}
```

您可以使用參考 Parameter Store 參數的參數，方式與任何其他文件參數相同。在下列範例中，`commands` 參數的預設值是 Parameter Store 參數 `myShellCommands`。透過將 `commands` 參數指定為 `runCommand` 字串，文件會執行在 `myShellCommands` 參數中儲存的命令。

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 }}"]
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 }}"]
      }
    },
    "mainSteps": [
      {
        "action": "aws:runShellScript",
        "name": "runShellScriptDefaultParams",
        "inputs": {
            "runCommand": [
              "{{ commands }}"
          ]
        }
      }
    ]
  }
```

> **Note**
>
> 您在文件的 `parameters` 部分可以參考 `String` 和 `StringList` Parameter Store 參數。您無法參考 `SecureString` Parameter Store 參數。

如需有關 Parameter Store 的詳細資訊，請參閱「AWS Systems Manager Parameter Store」。

類型：結構

`parameters` 結結構接受下列的欄位和數值：

- `type`：(必要) 允許的值包括：`String`、`StringList`、`Integer`、`Boolean`、`MapList` 和 `StringMap`。若要查看每個類型的範例，在下一個部分請參閱 SSM 文件參數 type 範例。

  > **Note**
  >
  > 指令類型文件僅支援 `String` 和 `StringList` 參數類型。

- `description`(選用) 參數說明。
- `default`：(選擇性) 在 Parameter Store 中預設的參數值或參考值。
- `allowedValues`：(選擇性) 參數允許的值陣列。定義參數的允許值會驗證使用者輸入。如果使用者輸入不允許的值，則無法開始執行。

  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`：(選擇性) 規則運算式，可驗證使用者輸入是否符合參數定義的模式。如果使用者輸入不符合允許的模式，則無法開始執行。

> **Note**
>
> Systems Manager 執行兩次 `allowedPattern` 驗證。當您使用文件時，在 API 層級使用 Java regex 程式庫執行第一次驗證。在處理文件之前，藉由使用 GO Regexp 程式庫，在 SSM Agent 上執行第二次驗證。

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`：（選用）用來在 `textarea`中顯示 `textfield`或 AWS Management Console。 `textfield` 是單行文字方塊。 `textarea` 是多行文字區域。
- `minItems`：(選擇性) 允許項目數量的最小值。
- `maxItems`：(選擇性) 允許項目數量的最大值。
- `minChars`：(選擇性) 允許參數字元數量的最小值。

- `maxChars`：(選擇性) 允許參數字元數量的最大值。

必要：否

variables

(僅限結構描述版本 0.3) 您可以在 Automation 執行手冊中的整個步驟中參考或更新的值。變數類似於參數，但有一個非常重要的差異。參數值在執行手冊的內容中是靜態的，但變數的值可以在執行手冊的內容中進行變更。更新變數的值時，資料類型必須與定義的資料類型相符。如需有關在自動化操作中更新變數值的資訊，請參閱 aws:updateVariable - 更新執行手冊變數的值

類型：布林值 | 整數 | MapList | 字串 | StringList | StringMap

必要：否

YAML

```
variables:
    payload:
        type: StringMap
        default: "{}"
```

JSON

```
{
    "variables": [
        "payload": {
            "type": "StringMap",
            "default": "{}"
        }
    ]
}
```

runtimeConfig

(結構描述 1.2 版) 一個或多個 Systems Manager 外掛程式套用的執行個體組態。不能保證外掛程式按順序執行。

類型：Dictionary<string,PluginConfiguration>

必要：否

mainSteps

(僅限結構描述版本 0.3、2.0 和 2.2) 可以包含多個步驟 (外掛程式) 的物件。外掛程式在步驟中定義。步驟會按文件中列出的順序執行。

類型：Dictionary<string,PluginConfiguration>

必要：是

outputs

(僅限結構描述版本 0.3) 執行此文件所產生的資料，可用於其他程序。例如，如果您的文件建立新 AMI，您可以將 "CreateImage.ImageId" 指定為輸出值，然後使用此輸出在後續自動化執行中建立新的執行個體。如需輸出的詳細資訊，請參閱 使用動作輸出作為輸入。

類型：Dictionary<string,OutputConfiguration>

必要：否

files

(僅限結構描述 0.3) 連接至文件並在自動化執行期間執行的指令碼檔案 (及其檢查總和)。僅適用於包含 `aws:executeScript` 動作，以及已在一或多個步驟中指定附件的文件。

若要了解 Automation 執行手冊支援的執行時期，請參閱 `aws:executeScript` – 執行指令碼。如需有關在 Automation Runbook 中包含指令碼的詳細資訊，請參閱 在執行手冊中使用指令碼 和 Automation 執行手冊的視覺化設計體驗。

使用附件建立 Automation Runbook 時，您還必須使用 `--attachments`選項 （適用於 AWS CLI) 或 `Attachments`（適用於 API 和 SDK) 指定附件檔案。您可以為存放在 Amazon Simple Storage Service (Amazon S3) 儲存貯體中的 SSM 文件和檔案指定檔案位置。如需詳細資訊，請參閱 AWS Systems Manager API 參考中的附件。

YAML

```
---
files:
  launch.py:
    checksums:
      sha256: 18871b1311b295c43d0f...
[truncated]...772da97b67e99d84d342ef4aEXAMPLE
```

JSON

```
"files": {
    "launch.py": {
        "checksums": {
            "sha256": "18871b1311b295c43d0f...
[truncated]...772da97b67e99d84d342ef4aEXAMPLE"
        }
    }
}
```

類型：Dictionary<string,FilesConfiguration>

必要：否

SSM 文件參數 **type** 範例

SSM 文件中的參數類型是靜態的。這意味著參數類型在定義後便無法變更。將參數與 SSM 文件外掛程式搭配使用時，無法在外掛程式的輸入中動態變更參數的類型。例如，您無法在 `aws:runShellScript` 外掛程式的 `runCommand` 輸入中參考 `Integer` 參數，因為此輸入接受字串或字串清單。若要對外掛程式輸入使用參數，參數類型必須與接受的類型相符。例如，您必須為 `aws:updateSsmAgent` 外掛程式的 `allowDowngrade` 輸入指定 `Boolean` 類型參數。如果您的參數類型與外掛程式的輸入類型不相符，則 SSM 文件便無法驗證，且系統不會建立文件。在其他外掛程式或 AWS Systems Manager 自動化動作的輸入內使用下游參數時，也是如此。例如，您無法參考 `aws:runDocument` 外掛程式 `documentParameters` 輸入中的 `StringList` 參數。即使下游 SSM 文件參數類型是 `StringList` 參數且與您參考的參數相符，此 `documentParameters` 輸入仍會接受字串映射。

搭配使用參數與 自動化動作時，在大多數情況下建立 SSM 文件並不會驗證參數類型。只有當您使用 `aws:runCommand` 動作時，才會在您建立 SSM 文件時驗證參數類型。在所有其他情況下，則會在執行動作之前驗證動作的輸入之時，在自動化執行期間進行參數驗證。例如，如果您的輸入參數為 `String`，而您將其參考為 `aws:runInstances` 動作的 `MaxInstanceCount` 輸入的數值，則會建立 SSM 文件。不過，執行文件時，自動化會在驗證 `aws:runInstances` 動作時會失敗，因為 `MaxInstanceCount` 輸入需要 `Integer`.

以下是每個參數 `type` 的範例。

字串

一連串零或多個 Unicode 字元以雙引號框住。例如，"i-1234567890abcdef0"。使用反斜線逸出。

YAML

```
---
InstanceId:
  type: String
  description: "(Optional) The target EC2 instance ID."
```

JSON

```
"InstanceId":{
  "type":"String",
  "description":"(Optional) The target EC2 instance ID."
}
```

StringList

由逗號分隔的字串項目清單。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"
}
```

Boolean

僅接受 `true` 或 `false`。不接受「true」或 0。

YAML

```
---
```

```
canRun:
  type: Boolean
  description: ''
  default: true
```

JSON

```
"canRun": {
  "type": "Boolean",
  "description": "",
  "default": true
}
```

Integer

整數號碼。不接受十進位小數，例如 3.14159 或以雙引號括住的號碼，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

金鑰與值的映射。金鑰和值必須是字串。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 物件清單。

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
}
```

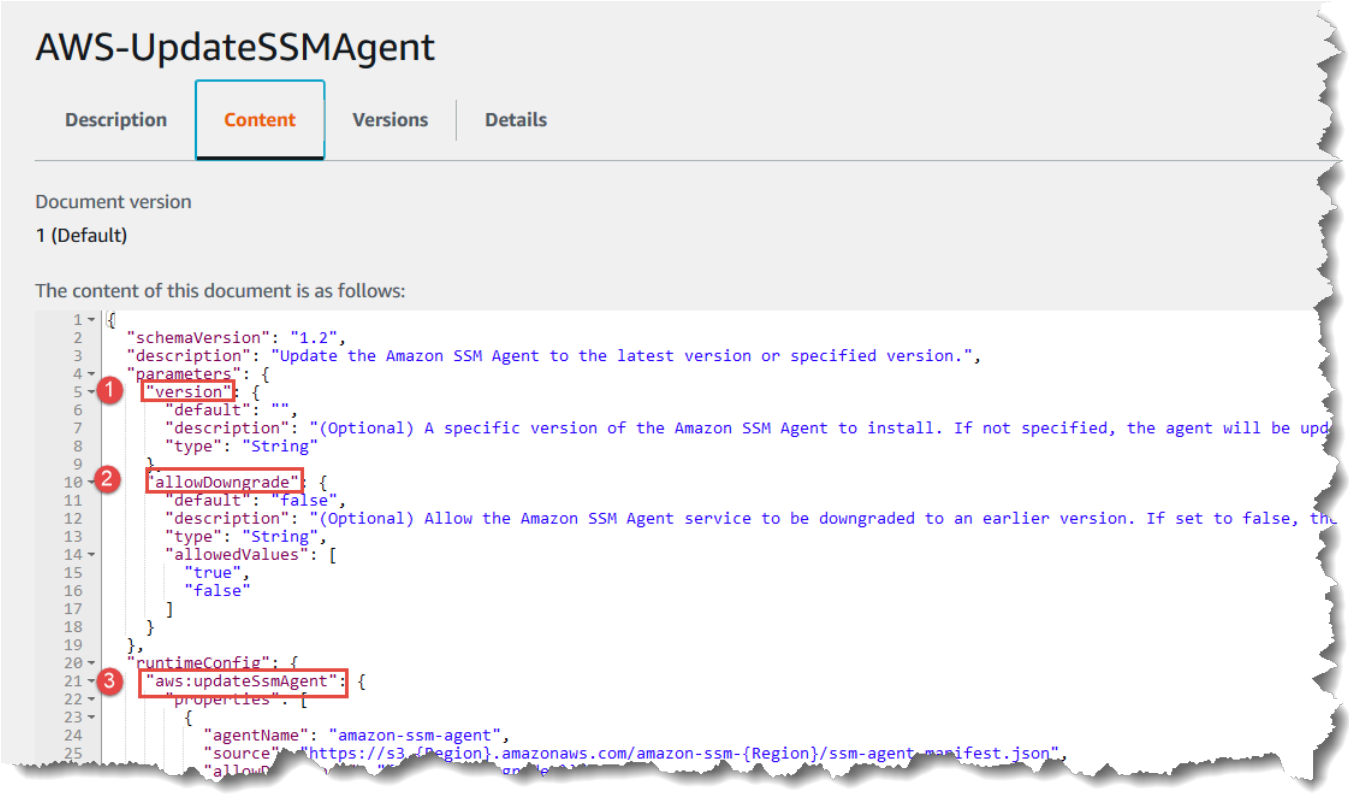
檢視 SSM 命令文件內容

若要預覽 AWS Systems Manager (SSM) 命令文件的必要和選用參數，除了文件執行的動作之外，您還可以在 Systems Manager 主控台中檢視文件的內容。

若要檢視 SSM 命令文件內容

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Documents (文件)。
3. 在搜尋方塊中，選取 Document type (文件類型)，然後選取 Command (命令)。
4. 選擇文件名稱，然後選擇 Content (內容) 標籤。
5. 在內容欄位中，檢閱文件的可用參數和動作步驟。

   例如，下圖顯示 (1) `version` 與 (2) `allowDowngrade` 是 `AWS-UpdateSSMAgent` 文件的可選參數，並且文件執行的第一個操作是 (3) `aws:updateSsmAgent`。



命令文件外掛程式參考

此參考說明您可以在 AWS Systems Manager (SSM) 命令類型文件中指定的外掛程式。這些外掛程式無法在使用 Automation 動作時用於 SSM Automation Runbook。如需 AWS Systems Manager 自動化動作的詳細資訊，請參閱 Systems Manager Automation 動作參考。

Systems Manager 會讀取 SSM 文件的內容，以決定在受管執行個體上執行的動作。每個文件包含程式碼執行部分。根據您文件的結構描述版本，這個程式碼執行部分可以包含一或多個外掛程式或步驟。有關於此的說明主題，外掛程式和步驟稱為 外掛程式。本節包含關於每個 Systems Manager 外掛程式的資訊。如需有關文件，包含建立文件和結構描述版本之間差異的更多資訊，請參閱 AWS Systems Manager Documents。

> **Note**
>
> 此處說明的一些外掛程式僅能夠在 Windows Server 執行個體或 Linux 執行個體上執行。每個外掛程式須注意平台相容性。
> macOS 的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體支援下列文件外掛程式：
>
> - `aws:refreshAssociation`

- aws:runShellScript
- aws:runPowerShellScript
- aws:softwareInventory
- aws:updateSsmAgent

目錄

- 共用的輸入
- 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

共用的輸入

透過 SSM Agent 3.0.502 版及更新版本，所有外掛程式都可使用以下輸入：

finallyStep

您想要文件執行的最後一個步驟。如果對步驟定義了此輸入，則它的優先順序高於在 `onFailure` 或 `onSuccess` 輸入中指定的 `exit` 值。若要讓具有此輸入的步驟預期執行，該步驟必須是在文件的 `mainSteps` 中定義的最後一步。

類型：布林值

有效值：`true` | `false`

必要：否

onFailure

如果您為具有 `exit` 值的外掛程式指定此輸入，且該步驟失敗，則步驟狀態會反映失敗，而且文件不會執行任何剩餘的步驟，除非已定義 `finallyStep`。如果您為具有 `successAndExit` 值的外掛程式指定此輸入，且該步驟失敗，則步驟狀態會顯示成功，而且文件不會執行任何剩餘的步驟，除非已定義 `finallyStep`。

類型：字串

有效值：`exit` | `successAndExit`

必要：否

onSuccess

如果您為外掛程式指定此輸入，且該步驟成功執行，則文件不會執行任何剩餘的步驟，除非已定義 `finallyStep`。

類型：字串

有效值：`exit`

必要：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

安裝、修復或解除安裝在 EC2 執行個體上的應用程式。這個外掛程式只能在 Windows Server 作業系統上執行。

語法

結構描述 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 }}"
      }
    }
  ]
}
```

結構描述 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 }}"
            }
         ]
      }
   }
}
```

屬性

動作

採取動作

類型：列舉

有效值：`Install` | `Repair` | `Uninstall`

必要：是

parameters

安裝的參數。

類型：字串

必要：否

source

應用程式的 `.msi` 檔案 URL。

類型：字串

必要：是

sourceHash

`.msi` 檔案的 SHA256 雜湊。

類型：字串

必要：否

aws:cloudWatch

將資料從 Windows Server 匯出到 Amazon CloudWatch 或 Amazon CloudWatch Logs，並使用 CloudWatch 指標監控資料。這個外掛程式只能在 Windows Server 作業系統上執行。如需有關設定 CloudWatch 與 Amazon Elastic Compute Cloud (Amazon EC2) 整合的詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的 Collecting metrics, logs, and traces with the CloudWatch agent。

> **Important**
>
> 統一的 CloudWatch 代理程式已取代 SSM Agent 作為將日誌資料傳送至 Amazon CloudWatch Logs 的工具。不支援 SSM Agent aws:cloudWatch 插件。建議只使用統一的 CloudWatch 代理程式來執行日誌收集流程。如需詳細資訊，請參閱下列主題：
>
> - 將節點日誌傳送至統一 CloudWatch Logs (CloudWatch 代理程式)
> - 將 Windows Server 節點日誌收集遷移到 CloudWatch 代理程式
> - 《Amazon CloudWatch 使用者指南》中的 Collecting metrics, logs, and traces with the CloudWatch agent。

您可以匯出和監控以下資料類型：

ApplicationEventLog

將應用程式事件日誌資料傳送至 CloudWatch Logs。

CustomLogs

將任何文字類日誌檔案傳送至 Amazon CloudWatch Logs。CloudWatch 外掛程式為日誌檔案建立指紋。系統會將資料位移與每個指紋建立關聯。當檔案改變時，外掛程式上傳檔案並記錄位移然後建立與指紋的關聯。此方法用於避免使用者開啟外掛程式，將服務與包含大量檔案的目錄關聯，並且系統會上傳所有檔案。

> **Warning**
>
> 請注意，如果您的應用程式在輪詢時嘗試截斷或清理日誌的任何日誌，任何 `LogDirectoryPath` 指定的日誌將可能遺失項目。例如，如果您想要限制的日誌檔案大小，達到該限制時建立新的日誌檔，然後持續將資料寫入新的檔案。

ETW

將 Event Tracing for Windows (ETW) 資料傳送到 CloudWatch Logs。

IIS

將 IIS 日誌資料傳送到 CloudWatch Logs。

PerformanceCounter

將 Windows 效能計數器傳送到 CloudWatch。您可以選擇不同類別以作為指標上傳至 CloudWatch。對於每個您要上傳的效能計數器，建立一個 PerformanceCounter 部分具有唯一的 ID (例如，「PerformanceCounter2」、」 PerformanceCounter3」等) 和屬性設定。

> **Note**
>
> 如果 AWS Systems Manager SSM Agent或 CloudWatch 外掛程式停止，效能計數器資料不會記錄在 CloudWatch 中。此行為不同於自訂日誌或 Windows 事件日誌。自訂日誌和 Windows 事件日誌會保留效能計數器資料，並且在 SSM Agent 或 CloudWatch 外掛程式能夠使用之後，將這些資料上傳到 CloudWatch。

SecurityEventLog

將安全性日誌資料傳送至 CloudWatch Logs。

SystemEventLog

將系統事件日誌資料傳送至 CloudWatch Logs。

您可以定義以下資料的目的地：

CloudWatch

傳送您的效能計數器指標資料的目的地。您可以用唯一的 ID 增加多個區段 (例如「CloudWatch2」和「CloudWatch3」)，並替不同的區域指定新的 ID 然後傳遞相同的資料到不同的地點。

CloudWatchLogs

傳送您的效能計數器日誌資料的目的地。您可以用唯一的 ID 增加多個區段 (例如「CloudWatchLogs2」和「CloudWatchLogs3」)，並替不同的區域指定新的 ID 然後傳遞相同的資料到不同的地點。

語法

```
"runtimeConfig":{
        "aws:cloudWatch":{
            "settings":{
                "startType":"{{ status }}"
            },
            "properties":"{{ properties }}"
        }
    }
```

設定和屬性

AccessKey

您的 存取金鑰 ID。此屬性是必要的，除非您使用 IAM 角色啟動執行個體。此屬性無法用於 SSM。

類型：字串

必要：否

CategoryName

效能監控從效能計數器類別而來。

類型：字串

必要：是

CounterName

效能監控的名稱從效能計數器類別而來。

類型：字串

必要：是

CultureName

要記錄時間戳記的所在地區。如果 CultureName 為空白，它會預設為 Windows Server 執行個體使用的相同地區設定。

類型：字串

有效的數值: 支援數值的清單，請參閱 Microsoft 網頁中的 National Language Support (NLS) (國家語言支援 (NLS) 參考)。不支援 div、div-MV、hu 和 hu-HU 值。

必要：否

DimensionName

Amazon CloudWatch 指標的維度。若您指定 DimensionName，您必須指定 DimensionValue。在列出指標時，這些參數會提供另一種視圖。您也可以針對多個指標使用同一種維度，如此就能檢視屬於特定維度的所有指標。

類型：字串

必要：否

DimensionValue

Amazon CloudWatch 指標的維度值。

類型：字串

必要：否

編碼

使用的檔案編碼 (例如，UTF-8)。使用編碼名稱，而非顯示名稱。

類型：字串

有效數值: 如需支援值的清單，請參閱 Microsoft Learn Library 中的編碼類別。

必要：是

篩選條件

日誌名稱的字首。將此參數留白，以監控所有檔案。

類型：字串

有效數值: 如需支援值的清單，請參閱 MSDN 上的 FileSystemWatcherFilter 屬性主題。

必要：否

流程

要上傳的每個資料類型，以及資料目的地 (CloudWatch 或 CloudWatch Logs)。例如，若要將在 "Id": "PerformanceCounter" 下定義的效能計數器傳送至在 "Id": "CloudWatch" 下定義的 CloudWatch 目的地，請輸入 "PerformanceCounter,CloudWatch"。同樣，若要將自

訂日誌、ETW 日誌和系統日誌傳送至在 "Id": "ETW" 下定義的 CloudWatch Logs 目的地，請輸入 "(ETW),CloudWatchLogs"。除此之外，您可以將相同的效能計數器或日誌檔案傳送到一個以上的目的地。例如，如果要將應用程式日誌傳送到您在 "Id": "CloudWatchLogs" 和 "Id": "CloudWatchLogs2" 區段中所定義的兩個不同目的地，輸入 "ApplicationEventLog, (CloudWatchLogs, CloudWatchLogs2)"。

類型：字串

有效數值 (來源): ApplicationEventLog | CustomLogs | ETW | PerformanceCounter | SystemEventLog | SecurityEventLog

有效值 (目的地)：CloudWatch | CloudWatchLogs | CloudWatch*n* | CloudWatchLogs*n*

必要：是

FullName

元件的完全名稱。

類型：字串

必要：是

Id

識別資料來源或目的地。這識別符在組態檔案中必須是唯一的。

類型：字串

必要：是

InstanceName

效能計數器執行個體的名稱。請勿使用星號 (*) 來表示所有的執行個體，因為每個效能計數器元件只支援一個指標。不過您可以使用 _Total。

類型：字串

必要：是

層級

要傳送至 Amazon CloudWatch 的訊息類型。

類型：字串

有效值：

- 1 – 只上傳錯誤訊息。
- 2 – 只上傳警告訊息。
- 4 – 只上傳資訊訊息。

您可以一起新增值，以包含多個類型訊息。例如，3 表示錯誤訊息 (1) 和警告訊息 (2)都包含。數值 7 表示包含錯誤訊息 (1)、警告訊息 (2) 和資訊訊息 (4)。

必要：是

> **Note**
>
> Windows Security Logs 應該設定層級為 7。

LineCount

標頭中的行數，以辨識日誌檔案。例如，IIS 日誌檔幾乎都具有相同的標頭。您可以輸入 3，這會讀取日誌檔標頭的前三行來進行辨識。在 IIS 日誌檔中，第三行是日期和時間戳記，這在不同的日誌檔中是不一樣的。

類型：整數

必要：否

LogDirectoryPath

對於 CustomLogs，此屬性是指 EC2 執行個體上存放日誌的路徑。對於 IIS 日誌，資料夾替個別的網站存放 IIS 日誌 (例如，C:\\inetpub\\logs\\LogFiles\\W3SVC***n***)。IIS 日誌只支援 W3C 日誌格式。不支援 IIS、NCSA 和自訂格式。

類型：字串

必要：是

LogGroup

您的日誌群組名稱。CloudWatch 主控台的 Log Groups (日誌群組) 畫面中會顯示此名稱。

類型：字串

必要：是

LogName

日誌檔案的名稱。

1. 若要在導覽窗格的事件檢視器中查找日誌名稱，請選取 Applications and Services Logs (應用程式和服務日誌)。
2. 在日誌清單中，在您要上傳的日誌上按一下滑鼠右鍵 (例如，Microsoft > Windows > 備份 > 操作)，然後選取 Create Custom View (建立自訂檢視)。
3. 在 Create Custom View (建立自訂檢視) 對話方塊中，選取 XML 標籤。LogName 位於 < 選取途徑 => 標籤 (例如，`Microsoft-Windows-Backup`)。複製此文字到 LogName 參數。

類型：字串

有效值：`Application` | `Security` | `System` | `Microsoft-Windows-WinINet/Analytic`

必要：是

LogStream

日誌串流目的地的名稱 若您使用 {instance_id}，日誌串流名稱是這個執行個體 ID 的預設值。

類型：字串

有效值：`{instance_id}` | `{hostname}` | `{ip_address}` *`<log_stream_name>`*

如果輸入的日誌串流名稱尚未存在，CloudWatch Logs 會自動為您建立。您可以使用文字字串或預先定義的變數 ({instance_id}, {hostname}, {ip_address}，或是組合全部三者來定義日誌串流名稱。

此參數中指定的日誌串流名稱會顯示在 CloudWatch 主控台的Log Groups > Streams for ***`<YourLogStream>`*** (日誌群組 > YourLogStream 串流) 畫面中。

必要：是

MetricName

您希望能包含效能資料的 CloudWatch 指標。

> **Note**
>
> 請不要在名稱中使用特殊字元。如果您這麼做了、指標和相關警示可能不會運作。

類型：字串

必要：是

NameSpace

將會在指標命名空間寫入效能計數器資料。

類型：字串

必要：是

PollInterval

上傳新的效能計數器和日誌資料時需要延遲多少秒。

類型：整數

有效值：將此屬性設為 5 個或多個秒。十五秒 (00:00:15) 是建議。

必要：是

區域

您要傳送日誌資料的 AWS 區域 。雖然您可以將效能計數器，傳送到與您傳送日誌資料所在位置不同的區域，還是建議您將此參數，設定為與執行個體執行位置相同的區域。

類型：字串

有效值：Systems Manager 和 CloudWatch Logs AWS 區域 支援的 區域 IDs，例如 `us-east-2`、`eu-west-1`和 `ap-southeast-1`。如需每個服務 AWS 區域 支援的 清單，請參閱《》中的 Amazon CloudWatch Logs Service Endpoints 和 Systems Manager 服務端點Amazon Web Services 一般參考。

必要：是

SecretKey

您的 私密存取金鑰。此屬性是必要的，除非您使用 IAM 角色啟動執行個體。

類型：字串

必要：否

startType

開啟或關閉執行個體上的 CloudWatch。

類型：字串

有效值：`Enabled` | `Disabled`

必要：是

TimestampFormat

要使用的時間戳記格式。如需支援值的清單，請參閱 MSDN 上的自訂日期和時間格式字串主題。

類型：字串

必要：是

TimeZoneKind

當您的日誌時間戳記中沒有時區資訊時，您需要提供時區資訊。如果此參數保留空白，且如果您的時間戳記不包含時區資訊，則 CloudWatch Logs 預設為本機時區。如果時間戳記已包含時區資訊，則會忽略此參數。

類型：字串

有效值：`Local` | `UTC`

必要：否

單位

適合指標的測量單位。

類型：字串

有效數值: 秒 | 微秒 | 毫秒 | 位元組 | 千位元組 (KB) | 百萬位元組 (MB) | 十億位元組 (GB) | 兆位元組 (TB) | 位元 | 千位元 (kb) | 百萬位元 (Mb) | 十億位元 (Gb) | 兆位元 (Tb) | 百分比 | 計數 | 位元組/秒 | 千位元組 (KB)/秒 | 百萬位元組 (MB)/秒 | 十億位元組 (GB)/秒 | 兆位元組 (TB)/秒 | 位元/秒 | 千位元 (kb)/ 秒 | 百萬位元 (Mb)/秒 | 十億位元 (Gb)/秒 | 兆位元 (Tb)/秒 | 計數/秒 | 無。

必要：是

`aws:configureDocker`

(結構描述 2.0 版本或更新版本) 設定執行個體來處理 Docker 和容器。大部分 Linux 變體和 Windows Server 作業系統都支援此外掛程式。

語法

結構描述 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 }}"
      }
    }
  ]
}
```

輸入

動作

執行動作的類型。

類型：列舉

有效值：Install | Uninstall

必要：是

aws:configurePackage

（結構描述 2.0 版或更新版本）安裝或解除安裝 AWS Systems Manager Distributor套件。您可以安裝最新版本、預設版本或您指定的套件版本。也支援 AWS 提供的套件。這個外掛程式可在 Windows Server 和 Linux 作業系統上執行，但 Linux 作業系統不支援所有的可用套件。

的可用 AWS 套件Windows Server包括下列項目：AWSPVDriver、AWSNVMe、AwsEnaNetworkDriver、AwsVssComponents、AmazonCloudWatchAgent、CodeDeployAgent和 AWSSupport-EC2Rescue.

適用於 Linux 作業系統的 AWS 套件包括下列項目：CodeDeployAgent、AmazonCloudWatchAgent和 AWSSupport-EC2Rescue。

語法

結構描述 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\"}"
            }
         }
      ]
}
```

輸入

name

要安裝或解除安裝的 AWS 套件名稱。可用套件包括：AWSPVDriver、AwsEnaNetworkDriver、AwsVssComponents 和 AmazonCloudWatchAgent。

類型：字串

必要：是

動作

安裝或解除安裝套件

類型：列舉

有效值：Install | Uninstall

必要：是

installationType

要執行的安裝類型。如果您指定 `Uninstall and reinstall`，套件會完全解除安裝，然後重新安裝。在重新安裝完成之前，應用程式無法使用。如果您指定 `In-place update`，根據您在更新指令碼中提供的指示，只會將新的或變更的檔案新增至現有的安裝。應用程式在整個更新程序中仍然可用。已發佈 AWS的套件不支援 `In-place update`選項。 `Uninstall and reinstall`是預設值。

類型：列舉

有效值：`Uninstall and reinstall` | `In-place update`

必要：否

additionalArguments

提供給安裝、解除安裝或更新指令碼的 JSON 字串格式的其他參數。每個參數必須加上字首 `SSM_`。通過使用慣例 `{{ssm:`*`parameter-name`*`}}`，您可以參考其他引數中的 Parameter Store 參數。若要在安裝、解除安裝或更新指令碼中使用其他參數，您必須使用適用於作業系統的語法，將參數作為環境變數參考。例如，在 PowerShell 中，將 `SSM_arg` 引數作為 `$Env:SSM_arg` 參考。您定義的引數數目沒有限制，但額外的引數輸入有 4096 個字元限制。此限制包括您定義的所有金鑰和值。

類型：StringMap

必要：否

version

特定版本的套件安裝或解除安裝。如果安裝，系統預設會安裝最新發佈的版本。如果移除，系統預設移除目前已安裝的版本。如果沒有找到任何安裝的版本，執行下載和解除安裝最新版本的動作。

類型：字串

必要：否

`aws:domainJoin`

將 EC2 執行個體加入網域。這個外掛程式可在 Linux 和 Windows Server 作業系統上執行。此外掛程式會將 Linux 執行個體的主機名稱變更為 EC2AMAZ-*XXXXXXX* 格式。如需加入 EC2 執行個體的詳細資訊，請參閱《 AWS Directory Service 管理指南》中的將 EC2 執行個體加入您的 AWS Managed Microsoft AD Directory。

語法

結構描述 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 }}"
      }
    }
  ]
}
```

結構描述 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 }}"
         }
      }
   }
}
```

屬性

directoryId

目錄的 ID。

類型：字串

必要：是

範例："directoryId": "d-1234567890"

directoryName

網域的名稱。

類型：字串

必要：是

範例："directoryName": "example.com"

directoryOU

組織單位 (OU)。

類型：字串

必要：否

範例："directoryOU": "OU=test,DC=example,DC=com"

dnsIpAddresses

DNS 的 IP 地址。

類型：StringList

必要：是

範例："dnsIpAddresses": ["198.51.100.1","198.51.100.2"]

hostname

要指派給節點的主機名稱。如果未提供，Windows 執行個體的名稱不會變更，而 Linux 執行個體將使用預設命名模式。如果提供，Windows 執行個體將使用確切提供的值，而對於 Linux 執行個體，其做為字首（除非keepHostName設定為「true」)。

類型：字串

必要：否

keepHostName

確定在加入網域時是否變更 Linux 執行個體的主機名稱。這是僅限 Linux 的參數。根據預設（沒有輸入 hostname、 和 keepHostName做為「false」)hostnameNumAppendDigits，Linux 主機將重新命名為模式 EC2AMAZ-XXXXXX。當設定為「true」時，它會保留原始主機名稱，並忽略對hostname和 的輸入hostnameNumAppendDigits。

類型：布林值

必要：否

hostnameNumAppendDigits

定義要在主機名稱值後面附加的隨機數字位數。這是僅限 Linux 的參數，可與 hostname 參數搭配使用。如果未提供 hostname，則會予以忽略。

類型：字串

允許的值：1 到 5

必要：否

範例

如需範例，請參閱AWS Managed Microsoft AD管理指南中的將 Amazon EC2 執行個體加入您的AWS Directory Service 。

aws:downloadContent

(結構描述 2.0 版本或更新版本) 從遠端位置下載 SSM 文件和指令碼。不支援 GitHub Enterprise 儲存庫。Linux 和 Windows Server 作業系統上支援此外掛程式。

語法

結構描述 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 }}"
      }
    }
  ]
}
```

輸入

sourceType

下載來源。Systems Manager 支援下列來源類型，用於下載指令碼和 SSM 文件：GitHub、Git、HTTP、S3 和 SSMDocument。

類型：字串

必要：是

sourceInfo

從所需的來源中擷取所需的資訊內容。

類型：StringMap

必要：是

對於來源類型 **GitHub,**，請指定以下資訊：

- 擁有者：儲存庫擁有者。
- 儲存庫: 儲存庫的名稱。
- 路徑：您想要下載的檔案或目錄路徑。
- getOptions：可從主要分支以外的分支或從儲存庫中的特定遞交中擷取內容的額外選項。如果您使用主要分支中的最新遞交，則可省略 getOptions。如果您的儲存庫是在 2020 年 10 月 1 日之後建立的，則預設分支可能被命名為 main 而不是 master。在此情況下，您需要指定 getOptions 參數的值。

  此參數使用以下的格式：

  - branch:refs/heads/*branch_name*

    預設值為 `master`。

    若要指定非預設分支，請使用以下格式：

    branch:refs/heads/*branch_name*

  - commitID：*commitID*

    預設值為 `head`。

    若要在最新遞交以外之其他遞交中使用您 SSM 文件的版本，請指定完整的遞交 ID。例如：

    ```
    "getOptions": "commitID:bbc1ddb94...b76d3bEXAMPLE",
    ```

- tokenInfo：您存放 GitHub 存取權杖資訊所在的 Systems Manager 參數 (SecureString 參數)，格式為 {{ssm-secure:*secure-string-token-name*}}。

> **Note**
>
> 此 `tokenInfo` 欄位是唯一支援 SecureString 參數的 SSM 文件外掛程式欄位。其他欄位不支援 SecureString 參數，其他 SSM 文件外掛程式也不支援。

```
{
    "owner":"TestUser",
    "repository":"GitHubTest",
    "path":"scripts/python/test-script",
    "getOptions":"branch:master",
    "tokenInfo":"{{ssm-secure:secure-string-token}}"
}
```

對於來源類型 **`Git`**，您必須指定以下資訊：

- repository

  您想要下載的檔案或目錄的 Git 存放庫 URL。

  類型：字串

此外，您還可以指定下列選用參數：

- getOptions

  可從主要分支以外的分支或從儲存庫中的特定遞交中擷取內容的額外選項。如果您使用主要分支中的最新遞交，則可省略 getOptions。

  類型：字串

  此參數使用以下的格式：

  - branch:refs/heads/*branch_name*

    預設值為 `master`。

    只有當 SSM 文件存放於 `master` 以外的分支時，才需要 "`branch`"。例如：

    ```
    "getOptions": "branch:refs/heads/main"
    ```

  - commitID：*commitID*

預設值為 head。

若要在最新遞交以外之其他遞交中使用您 SSM 文件的版本，請指定完整的遞交 ID。例如：

```
"getOptions": "commitID:bbc1ddb94...b76d3bEXAMPLE",
```

- privateSSHKey

  連線至您指定的 repository 時，要使用的 SSH 金鑰。您可以使用下列格式來參考 SSH 金鑰值的 SecureString 參數：{{ssm-secure:*your-secure-string-parameter*}}。

  類型：字串

- skipHostKeyChecking

  連線至您指定的 repository 時，確定 StrictHostKeyChecking 選項的值。預設值為 false。

  類型：布林值

- 使用者名稱

  使用 HTTP 連線至您指定的 repository 時，要使用的使用者名稱。您可以使用下列格式來參考使用者名稱值的 SecureString 參數：{{ssm-secure:*your-secure-string-parameter*}}。

  類型：字串

- 密碼

  使用 HTTP 連線至您指定的 repository 時，要使用的密碼。您可以使用下列格式來參考密碼值的 SecureString 參數：{{ssm-secure:*your-secure-string-parameter*}}。

  類型：字串

對於來源類型 **HTTP**，您必須指定以下資訊：

- url

  您想要下載的檔案或目錄的 URL。

  類型：字串

此外，您還可以指定下列選用參數：

- allowInsecureDownload

判斷是否可透過未使用 Secure Socket Layer (SSL) 或 Transport Layer Security (TLS) 進行加密的連線來執行下載。預設值為 `false`。不建議在未加密的情況下執行下載。如果您選擇這樣做，您應承擔所有相關風險。安全性是 AWS 與您之間的共同責任。這被描述為共同的責任模式。如需進一步了解，請參閱共同的責任模型。

類型：布林值

- authMethod

  當連線至您指定的 `url` 時，判斷是否要使用使用者名稱和密碼進行身分驗證。如果您指定 `Basic` 或 `Digest`，則必須提供 `username` 和 `password` 參數的值。若要使用 `Digest` 方法，必須在您的執行個體上安裝 SSM Agent 3.0.1181.0 版或更新版本。`Digest` 方法支援 MD5 和 SHA256 加密。

  類型：字串

  有效值：`None` | `Basic` | `Digest`

- 使用者名稱

  使用 `Basic` 身分驗證連線至您指定的 `url` 時，要使用的使用者名稱。您可以使用下列格式來參考使用者名稱值的 `SecureString` 參數：`{{ssm-secure:`*`your-secure-string-parameter`*`}}`。

  類型：字串

- 密碼

  使用 `Basic` 身分驗證連線至您指定的 `url` 時，要使用的密碼。您可以使用下列格式來參考密碼值的 `SecureString` 參數：`{{ssm-secure:`*`your-secure-string-parameter`*`}}`。

  類型：字串

對於來源類型 **S3**，請指定以下資訊：

- 路徑：您想要從 Amazon S3 中下載的檔案或目錄的 URL。

```
{
    "path": "https://s3.amazonaws.com/amzn-s3-demo-bucket/powershell/
helloPowershell.ps1"
}
```

對於來源類型 **SSMDocument**，請指定下列其中一項：

- 名稱：名稱和文件的版本，格式如下：name:version。版本是非必須的。

```
{
    "name": "Example-RunPowerShellScript:3"
}
```

- 名稱：文件的 ARN，格式如下：arn:aws:ssm:*region*:*account_id*:document/*document_name*

```
{
   "name":"arn:aws:ssm:us-east-2:3344556677:document/MySharedDoc"
}
```

destinationPath

將您想要下載的檔案選用在的執行個體上的本機路徑。如果您不指定路徑的相對路徑，內容會下載到您的命令 ID 相對應的路徑。

類型：字串

必要：否

aws:psModule

在 Amazon EC2 執行個體上安裝 PowerShell 模組。這個外掛程式只能在 Windows Server 作業系統上執行。

語法

結構描述 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 }}"
      }
    }
  ]
}
```

結構描述 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 }}"
            }
         ]
      }
   }
}
```

屬性

runCommand

在模組安裝完畢後，PowerShell 的命令來執行。

類型：StringList

必要：否

source

在的執行個體上應用程式 `.zip` 檔案的本機路徑或 URL。

類型：字串

必要：是

sourceHash

`.zip` 檔案的 SHA256 雜湊。

類型：字串

必要：否

timeoutSeconds

在命令完成到被認定為失敗之間的秒數。

類型：字串

必要：否

workingDirectory

在您的執行個體上的工作目錄路徑。

類型：字串

必要：否

aws:refreshAssociation

(結構描述 2.0 版本或更新版本) 更新 (強制套用) 有需要的關聯。這個動作會根據選中的關聯或者全部與目標有聯結的關聯裡的定義來改變系統的狀態。這個外掛程式可在 Linux 和 Microsoft Windows Server 作業系統上執行。

語法

結構描述 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 }}"
        ]
      }
    }
  ]
}
```

輸入

associationIds

列出的關聯 ID。如果空，所有綁定到指定的目標的關聯都會被套用。

類型：StringList

必要：否

aws:runDockerAction

(結構描述 2.0 版本或更新版本) 在容器上執行 Docker 動作。這個外掛程式可在 Linux 和 Microsoft Windows Server 作業系統上執行。

語法

結構描述 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 }}"
         }
      }
   ]
}
```

輸入

動作

執行動作的類型。

類型：字串

必要：是

容器

Docker 容器的 ID。

類型：字串

必要：否

image

Docker 影像名稱

類型：字串

必要：否

命令提示字元

容器指令

類型：字串

必要：否

memory

容器記憶體限制。

類型：字串

必要：否

cpuShares

CPU 容器共享 (相對重量)。

類型：字串

必要：否

磁碟區

容器磁碟容量。

類型：StringList

必要：否

env

容器的環境變數。

類型：字串

必要：否

使用者

容器的使用者名稱。

類型：字串

必要：否

發布

容器發佈的連接埠。

類型：字串

必要：否

workingDirectory

受管節點上工作目錄的路徑。

類型：字串

必要：否

timeoutSeconds

在命令完成到被認定為失敗之間的秒數。

類型：字串

必要：否

aws:runDocument

(結構描述 2.0 版本或更新版本) 執行存放在 Systems Manager 或在本機共用的 SSM 文件。您可以搭配使用此外掛程式與 aws:downloadContent外掛程式，將 SSM 文件從遠端位置下載到本機共用，然後執行它。Linux 和 Windows Server 作業系統上支援此外掛程式。此外掛程式不支援執行 AWS-UpdateSSMAgent 文件或任何使用 aws:updateSsmAgent 外掛程式的文件。

語法

結構描述 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 }}"
      }
    }
  ]
}
```

輸入

documentType

執行的文件類型。您可以執行本機文件 (LocalPath) 或存放在 Systems Manager (SSMDocument) 的文件。

類型：字串

必要：是

documentPath

文件的路徑。如果 documentType 是 LocalPath，請指定在本機共享的文件路徑。如果 documentType 是 SSMDocument，請指定文件的名稱。

類型：字串

必要：否

documentParameters

文件的參數。

類型：StringMap

必要：否

aws:runPowerShellScript

執行 PowerShell 指令碼或指定路徑來執行指令碼。這個外掛程式可在 Microsoft Windows Server 和 Linux 作業系統上執行。

語法

結構描述 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 }}"
        ]
      }
    }
  ]
}
```

結構描述 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 }}"
            }
```

```
            ]
        }
    }
}
```

屬性

runCommand

指定命令或指定在執行個體上現有的指令碼的路徑來執行。

類型：StringList

必要：是

timeoutSeconds

在命令完成到被認定為失敗之間的秒數。如果達到逾時，Systems Manager 停止命令執行。

類型：字串

必要：否

workingDirectory

在您的執行個體上的工作目錄路徑。

類型：字串

必要：否

aws:runShellScript

執行 Linux shell 指令碼或指定路徑來執行指令碼。這個外掛程式只可在 Linux 作業系統上執行。

語法

結構描述 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 }}"
        ]
      }
    }
  ]
}
```

結構描述 1.2

YAML

```
---
runtimeConfig:
  aws:runShellScript:
    properties:
    - runCommand: "{{ commands }}"
      workingDirectory: "{{ workingDirectory }}"
      timeoutSeconds: "{{ executionTimeout }}"
```

JSON

```
{
   "runtimeConfig":{
      "aws:runShellScript":{
         "properties":[
            {
               "runCommand":"{{ commands }}",
               "workingDirectory":"{{ workingDirectory }}",
               "timeoutSeconds":"{{ executionTimeout }}"
            }
         ]
      }
   }
}
```

屬性

runCommand

指定命令或指定在執行個體上現有的指令碼的路徑來執行。

類型：StringList

必要：是

timeoutSeconds

在命令完成到被認定為失敗之間的秒數。如果達到逾時，Systems Manager 停止命令執行。

類型：字串

必要：否

workingDirectory

在您的執行個體上的工作目錄路徑。

類型：字串

必要：否

aws:softwareInventory

(結構描述 2.0 版本或更新版本) 在受管執行個體上收集與應用程式、檔案和組態相關的中繼資料。這個外掛程式可在 Linux 和 Microsoft Windows Server 作業系統上執行。當您設定清查收集時，請先建立 AWS Systems Manager State Manager關聯。Systems Manager 會在執行關聯時收集庫存資料。如果沒有先建立關聯，則當您試圖叫用 aws:softwareInventory 外掛程式時，系統會傳回以下錯誤：

```
The aws:softwareInventory plugin can only be invoked via ssm-associate.
```

執行個體一次只能設定一個庫存關聯。若您為執行個體設定兩個以上的關聯，庫存關聯便不會執行，並且也不會收集任何庫存資料。如需收集庫存的相關資訊，請參閱 AWS Systems Manager庫存。

語法

結構描述 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 }}"
         }
      }
   ]
}
```

輸入

應用程式

(選用) 收集已安裝應用程式的中繼資料。

類型：字串

必要：否

awsComponents

（選用）收集 amazon-ssm-agent 等 AWS 元件的中繼資料。

類型：字串

必要：否

files

(選用，需要 SSM Agent 2.2.64.0 版或更新版本) 收集檔案的中繼資料，包括檔案名稱、檔案的建立時間、上次修改和存取檔案的時間以及檔案大小等。如需收集檔案庫存的相關資訊，請參閱 使用檔案與 Windows 登錄檔清查。

類型：字串

必要：否

networkConfig

(選用) 收集網路組態的中繼資料。

類型：字串

必要：否

billingInfo

（選用） 收集與 帳單代碼相關聯的平台詳細資訊中繼資料AMI。

類型：字串

必要：否

windowsUpdates

(選用) 收集所有 Windows 更新的中繼資料。

類型：字串

必要：否

instanceDetailedInformation

(選用) 收集的執行個體資訊比預設庫存外掛程式 (`aws:instanceInformation`) 提供的更多，包括 CPU 模型、速度和核心數量等。

類型：字串

必要：否

services

(選用，僅 Windows 作業系統需要 SSM Agent 2.2.64.0 版或更新版本) 收集服務組態的中繼資料。

類型：字串

必要：否

windowsRegistry

(選用，僅 Windows 作業系統需要 SSM Agent 2.2.64.0 版或更新版本) 收集 Windows 登錄機碼和值。而且，您還能選擇機碼路徑，並以遞迴方式收集所有機碼和值。此外，您也可以收集指定路徑的特定登錄機碼及其值。庫存將收集機碼路徑、名稱、類型與值。如需收集 Windows 登錄檔庫存的詳細資訊，請參閱 使用檔案與 Windows 登錄檔清查。

類型：字串

必要：否

windowsRoles

(選用，僅 Windows 作業系統需要 SSM Agent 2.2.64.0 版或更新版本) 收集適用於 Microsoft Windows 角色組態的中繼資料。

類型：字串

必要：否

customInventory

(選用) 收集自訂庫存資料。如需自訂庫存的詳細資訊，請參閱使用自訂庫存。

類型：字串

必要：否

customInventoryDirectory

（選用）從指定的目錄收集自訂庫存資料。如需自訂庫存的詳細資訊，請參閱使用自訂庫存。

類型：字串

必要：否

aws:updateAgent

更新 EC2Config 服務到最新版本或指定 EC2Config 服務的舊版本。這個外掛程式只能在 Microsoft Windows Server 作業系統上執行。如需有關 EC2Config 服務的詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Configuring a Windows Instance using the EC2Config service (legacy)。

語法

結構描述 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"
      }
    }
  ]
}
```

結構描述 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 }}"
         }
      }
   }
}
```

屬性

agentName

EC2Config。這是執行 EC2Config 服務的代理名稱。

類型：字串

必要：是

allowDowngrade

允許 EC2Config 服務降級到較舊的版本。如果設定為非，該服務只可以升級到較新版本 (預設)。如果設定為是，指定之前的舊版本。

類型：布林值

必要：否

source

Systems Manager 複製 EC2Config 版本進行安裝的位置。您無法變更此位置。

類型：字串

必要：是

targetVersion

安裝指定版本的 EC2Config 服務。如果未指定，服務會更新到最新版本。

類型：字串

必要：否

aws:updateSsmAgent

將 SSM Agent 更新到最新版本或指定舊版本。這個外掛程式可在 Linux 和 Windows Server 作業系統上執行。如需詳細資訊，請參閱使用 SSM Agent。

語法

結構描述 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 }}"
      }
    }
  ]
}
```

結構描述 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 }}"
            }
         ]
      }
   }
}
```

屬性

agentName

amazon-ssm-agent。這是在執行個體上處理請求和執行命令的 Systems Manager 代理程式名稱。

類型：字串

必要：是

allowDowngrade

允許 SSM Agent 降級到較早的版本。如果設定為非，代理程式只可以升級到較新版本 (預設)。如果設定為是，指定之前的舊版本。

類型：布林值

必要：是

source

Systems Manager 複製 SSM Agent 版本進行安裝的位置。您無法變更此位置。

類型：字串

必要：是

targetVersion

指定安裝 SSM Agent 的版本。如果未指定，代理程式會更新到最新版本。

類型：字串

必要：否

建立 SSM 文件內容

如果 AWS Systems Manager 公有文件未執行您想要對 AWS 資源執行的所有動作，您可以建立自己的 SSM 文件。您也可以使用主控台複製 SSM 文件。複製文件會將現有文件的內容複製到您可以修改的新文件中。建立或複製文件時，文件的內容不得超過 64 KB。此配額也包括在執行階段為輸入參數指定的內容。當您建立新的 `Command` 或 `Policy` 文件時，我們建議您使用結構描述 2.2 或更新版本，以便您可以利用最新的功能，例如文件編輯、自動版本控制、排序等。

撰寫 SSM 文件內容

若要建立您自己的 SSM 文件內容，請務必瞭解 SSM 文件提供的不同結構描述、功能、外掛程式和語法。我們建議您熟悉下列資源。

- 撰寫您自己的 AWS Systems Manager 文件
- 資料元素和參數
- 結構描述、功能以及範例
- 命令文件外掛程式參考
- Systems Manager Automation 動作參考
- 自動化系統變數
- 其他執行手冊範例
- 透過 AWS Toolkit for Visual Studio Code 來使用 Systems Manager Automation Runbook
- Automation 執行手冊的視覺化設計體驗
- 在執行手冊中使用指令碼

AWS 預先定義的 SSM 文件可能會執行您需要的一些動作。您可以根據文件類型，在自訂 SSM 文件中使用 `aws:runDocument`、`aws:runCommand` 或 `aws:executeAutomation` 外掛程式來呼叫這些文件。您也可以將這些文件的部分複製到自訂 SSM 文件中，然後編輯內容以符合您的需求。

Tip

建立 SSM 文件內容時，您可能會在測試時數次變更內容並更新 SSM 文件。下列命令會以您的最新內容更新 SSM 文件，並將文件的預設版本更新為文件的最新版本。

Note

Linux 和 Windows 命令使用 `jq` 命令列工具來篩選 JSON 回應資料。

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
```

複製 SSM 文件

您可以使用 Systems Manager 文件主控台複製 AWS Systems Manager 文件，以建立 SSM 文件。複製 SSM 文件會將現有文件的內容複製到您可以修改的新文件中。您無法複製大於 64 KB 的文件。

若要複製 SSM 文件

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Documents (文件)。
3. 在搜尋方塊中，輸入您要複製的文件的名稱。
4. 選擇您要翻製的文件名稱，然後選擇 Actions (動作) 下拉式選單中的 Clone document (複製文件)。
5. 視需要修改文件，然後選擇 Create document (建立文件) 以儲存文件。

撰寫 SSM 文件內容之後，您可以使用下列其中一種方法來建立 SSM 文件。

建立 SSM 文件

- 建立複合文件

建立複合文件

composite AWS Systems Manager (SSM) 文件是一種自訂文件，可透過執行一或多個次要 SSM 文件來執行一系列動作。複合文件提升了 infrastructure as code，讓您能夠為常見任務建立一組標準的 SSM 文件，例如自舉軟體或網域加入執行個體。然後，您可以在相同的 AWS 帳戶 中跨 共用這些文件 AWS 區域 ，以減少 SSM 文件維護並確保一致性。

例如，您可以建立複合的文件來執行下列動作：

1. 安裝允許清單中的所有修補程式。
2. 安裝防毒軟體
3. 從 GitHub 下載指令碼並執行指令碼。

在這個範例中，自訂 SSM 文件中包含下列外掛程式以執行下列動作：

1. 用於執行 `AWS-RunPatchBaseline` 文件的 `aws:runDocument` 外掛程式，可安裝所有允許列出的修補程式。
2. 用於執行 `AWS-InstallApplication` 文件的 `aws:runDocument` 外掛程式，可安裝防毒軟體。
3. 用於從 GitHub 下載並執行指令碼的 `aws:downloadContent` 外掛程式。

複合和次要文件可以存放在 Systems Manager、GitHub (公有和私有儲存庫) 或 Amazon S3。您可以用 JSON 或 YAML 格式建立複合文件和次要文件。

> **Note**
>
> 複合文件的執行深度最多只能為三個文件。這表示複合文件可以呼叫一個子文件，以及該子文件可以再呼叫一個文件。

若要建立複合文件，需要在自訂 SSM 文件中新增 `aws:runDocument` 外掛程式，並指定所需的輸入。下列的範例是個建立複合的文件來執行下列動作：

1. 執行 `aws:downloadContent` 外掛程式，將 SSM 文件從 GitHub 公有儲存庫下載到名為 bootstrap 的本機目錄。此 SSM 文件稱為 StateManagerBootstrap.yml (YAML 文件)。
2. 執行 `aws:runDocument` 外掛程式，以執行 StateManagerBootstrap.yml 文件。無需指定參數。
3. 執行 `aws:runDocument` 外掛程式，以執行 `AWS-ConfigureDocker pre-defined` SSM 文件。指定的參數在執行個體上安裝 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\"}"
      }
    }
  ]
}
```

詳細資訊

- 如需使用 Run Command 呼叫指令碼時重新開機伺服器和執行個體的資訊，請參閱 執行命令時處理重新啟動。
- 如需有關您能夠新增至自訂 SSM 文件的外掛程式的詳細資訊，請參閱 命令文件外掛程式參考。

- 如果您只想簡單從遠端位置 (無須建立複合文件) 執行文件的詳細資訊，請參閱 從遠端位置執行文件。

使用文件

本節包含有關如何使用及處理 SSM 文件的資訊。

主題

- 比較 SSM 文件版本
- 建立 SSM 文件
- 刪除自訂 SSM 文件
- 從遠端位置執行 文件
- 共用 SSM 文件
- 搜尋 SSM 文件

比較 SSM 文件版本

您可以在 Systems Manager 文件主控台中比較 AWS Systems Manager (SSM) 文件版本之間的內容差異。比較 SSM 文件的版本時，會反白顯示版本內容之間的差異。

若要比較 SSM 文件內容 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Documents (文件)。
3. 在文件清單中，選擇您要比較其內容的文件。
4. 在 Content (內容) 索引標籤中，選取 Compare versions (比較版本)，然後選擇您要與之比較內容的文件版本。

建立 SSM 文件

如 撰寫 SSM 文件內容 中所述，建立自訂 SSM 文件的內容後，您可以使用 Systems Manager 主控台，以您的內容建立 SSM 文件。

建立 SSM 文件

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Documents (文件)。
3. 選擇 Create command or session (建立命令或工作階段)。
4. 輸入文件的描述性名稱。
5. (選用) 針對 Target type (目標類型)，指定文件可在其上執行的資源類型。
6. 在 Document type (文件類型)清單中，選擇您要建立的文件類型。
7. 刪除 Content (內容) 欄位中的括號，然後貼上先前建立的文件。
8. (選用) 展開 Document tags (文件標籤) 區段，將一或多個標籤鍵值組套用至文件。

   標籤是您指派給資源的選用性中繼資料。標籤允許您以不同的方式 (例如用途、擁有者或環境) 將資源分類。例如，您可能想要標記文件來識別其執行的任務類型、目標作業系統類型以及其執行所在的環境。在這種情況下，您可以指定以下索引鍵名稱/值對：

   - `Key=TaskType,Value=MyConfigurationUpdate`
   - `Key=OS,Value=AMAZON_LINUX_2`
   - `Key=Environment,Value=Production`
9. 選擇 Create document (建立文件) 以儲存文件。

刪除自訂 SSM 文件

如果您不想再使用自訂 SSM 文件，則可以使用 AWS Systems Manager 主控台刪除它。

刪除 SSM 文件

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Documents (文件)。
3. 選取您要刪除的文件。
4. 選取 Delete (刪除)。當提示您刪除文件時，請選取 Delete (刪除)。

如需使用命令列工具或 SDK 刪除 SSM 文件的範例，請參閱`DeleteDocument` 搭配 AWS SDK 或 CLI 使用。

從遠端位置執行 文件

您可以使用AWS-RunDocument預先定義的 SSM 文件，從遠端位置執行 AWS Systems Manager (SSM) 文件。本文件支援執行在下列位置存放的 SSM 文件：

- 公有和私有 GitHub 儲存庫 (不支援 GitHub Enterprise)
- Amazon S3 儲存貯體
- Systems Manager

雖然您也可以使用 State Manager或 自動化 中的工具來執行遠端文件 AWS Systems Manager，但下列程序只會說明如何在 Systems Manager 主控台中使用 AWS Systems Manager Run Command 來執行遠端 SSM 文件。

> **Note**
>
> AWS-RunDocument 只能用來執行命令類型的 SSM 文件，而不能執行其他類型，例如 Automation Runbook。AWS-RunDocument 使用 aws:downloadContent 外掛程式。如需有關 aws:downloadContent 外掛程式的詳細資訊，請參閱 aws:downloadContent。

開始之前

在您可以開始執行遠端文件之前，請必須完成以下工作。

- 建立一份 SSM 命令文件，並將它儲存在遠端位置。如需詳細資訊，請參閱 建立 SSM 文件內容
- 如果您打算執行存放在 GitHub 私有儲存庫中的遠端文件，則必須為 GitHub 安全存取權杖建立一個 Systems Manager SecureString 參數。您無法透過 SSH 手動傳遞權杖來存取 GitHub 私有儲存庫中的遠端文件。您必須將存取字符做為 Systems Manager SecureString 參數傳遞。如需建立 SecureString 參數的詳細資訊，請參閱 在 Systems Manager 中建立 Parameter Store 參數。

執行遠端文件 (主控台)

執行遠端文件

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Run Command。

3. 選擇執行命令。
4. 在 Document (文件) 清單中，請選擇 **AWS-RunDocument**。
5. 在 Command parameters (命令參數) 中，針對 Source Type (來源類型) 選擇選項。

   - 如果選擇 GitHub，請指定來源資訊，資訊格式如下：

     ```
     {
         "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 是可從主要分支以外的分支或從儲存庫中的特定遞交中擷取內容的額外選項。如果您使用主要分支中的最新遞交，則可省略 getOptions。只有當 SSM 文件存放於 master 以外的分支時，才需要 branch 參數。
     > 若要使用存放庫中特定「遞交」中的 SSM 文件，請使用 commitID 與 getOptions 來代替 branch。例如：
     >
     > ```
     > "getOptions": "commitID:bbc1ddb94...b76d3bEXAMPLE",
     > ```

   - 如果您選擇 S3，指定 Source Info (來源資訊) 的資訊格式如下：

     ```
     {"path":"URL_to_document_in_S3"}
     ```

例如：

```
{"path":"https://s3.amazonaws.com/amzn-s3-demo-bucket/scripts/ruby/
mySSMdoc.json"}
```

- 如果您選擇 SSMDocument，指定 Source Info (來源資訊) 的資訊格式如下：

```
{"name": "document_name"}
```

例如：

```
{"name": "mySSMdoc"}
```

6. 在 Document Parameters (文件參數) 欄位中輸入遠端 SSM 文件的參數。例如，如果您執行 AWS-RunPowerShell 文件，您可以指定：

```
{"commands": ["date", "echo \"Hello World\""]}
```

如果您執行 AWS-ConfigureAWSPack 文件，您可以指定：

```
{
   "action":"Install",
   "name":"AWSPVDriver"
}
```

7. 在 Targets (目標) 區段中，透過手動指定標籤、選取執行個體或邊緣裝置，或指定資源群組，選擇您要執行這項操作的受管節點。

> **Tip**
>
> 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

8. 對於其他參數：

- 在 Comment (註解) 中，輸入此命令的相關資訊。
- 在逾時 (秒) 中，指定在命令執行全面失敗之前，系統要等候的秒數。

9. 對於 Rate control (速率控制)：

- 在並行中，指定可同時執行命令的受管節點數目或百分比。

> **Note**
>
> 如果透過指定套用至受管節點的標籤或指定 AWS 資源群組選取了目標，且您不確定會以多少個受管節點為目標，則透過指定百分比限制可以同時執行文件之目標的數量。

- 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比之節點上的命令失敗之後，停止在其他受管節點上執行命令。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止傳送命令。仍在處理命令的受管節點也可能會傳送錯誤。

10. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取將命令輸出寫入至 S3 儲存貯體方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

> **Note**
>
> 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給執行個體之執行個體設定檔 (適用於 EC2 執行個體) 或 IAM 服務角色 (啟用混合模式的機器) 的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確定與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

11. 在SNS 通知區段中，如果您要傳送有關命令執行狀態的通知，請選取啟用 SNS 通知核取方塊。

    如需為 Run Command 設定 Amazon SNS 通知的詳細資訊，請參閱 使用 Amazon SNS 通知監控 Systems Manager 狀態變更。

12. 選擇執行。

> **Note**
>
> 如需使用 Run Command 呼叫指令碼時重新開機伺服器和執行個體的資訊，請參閱 執行命令時處理重新啟動。

共用 SSM 文件

您可以私下或公開地與相同 中的帳戶共用 AWS Systems Manager (SSM) 文件 AWS 區域。若要私下共用文件，您需要修改文件許可，並根據特定人員的 AWS 帳戶 ID 允許其進行存取。若要公開共用 SSM 文件，請修改文件許可並指定 `All`。文件不能同時公開和私下共用。

> **Warning**
>
> 只能使用受信任來源的共用 SSM 文件。當您使用任何共用文件，請在使用文件之前仔細檢視文件的內容，讓您了解文件將會如何改變您執行個體的組態。如需共用文件的最佳實務詳細資訊，請參閱 共用 SSM 文件的最佳實務。

限制

當您開始使用 SSM 文件時，請注意以下限制。

- 只有文件擁有者可以分享文件。
- 您必須停止共用文件，才能刪除文件。如需詳細資訊，請參閱修改共用 SSM 文件的許可。
- 您可以共用最多 1000 個文件 AWS 帳戶。您可以在 支援 中心請求提高此限制。對於 Limit type (限制類型)，選擇 EC2 Systems Manager 並說明請求的原因。
- 您最多可以公開共用五個 SSM 文件。您可以在 支援 中心請求提高此限制。對於 Limit type (限制類型)，選擇 EC2 Systems Manager 並說明請求的原因。
- 文件只能與相同 中的其他帳戶共用 AWS 區域 。不支援跨區域共用。

> **Important**
>
> 在 Systems Manager 中，Amazon 擁有的 SSM 文件是由 Amazon Web Services 本身建立和管理的文件。Amazon 擁有的文件的文件名中包含如 `AWS-*` 的字首。文件的擁有者被視為 Amazon，而不是其中的特定使用者帳戶 AWS。這些文件公開供所有人使用。

如需有關 Systems Manager 服務配額的詳細資訊，請參閱 AWS Systems Manager Service Quotas。

目錄

- 共用 SSM 文件的最佳實務
- 封鎖 SSM 文件的公有共用

- 共用 SSM 文件
- 修改共用 SSM 文件的許可
- 使用共用的 SSM 文件

共用 SSM 文件的最佳實務

共享或使用共用文件之前，請檢閱下列指導方針。

移除敏感的資訊

仔細檢閱您的 AWS Systems Manager (SSM) 文件，並移除任何敏感資訊。例如，確認文件不包含您的 AWS 登入資料。如果您與特定個人共享文件，這些使用者可以檢視文件中的資訊。如果您將文件設定為公開，任何使用者可以檢視文件中的資訊。

封鎖文件的公有共用

檢閱帳戶中所有公開共用的 SSM 文件，並確認是否要繼續共用它們。若要停止公開共用文件，必須依照本主題的修改共用 SSM 文件的許可章節所述修改文件許可設定。開啟封鎖公開共用設定不會影響目前正在公開共用的任何文件。除非您的使用案例需要公開共用文件，否則建議在 Systems Manager 文件主控台的偏好設定區段中開啟 SSM 文件的封鎖公開共用設定。開啟此設定可避免 SSM 文件遭到不必要的存取。封鎖公有共用設定是帳戶層級設定，每個 AWS 區域各有不同。

使用 IAM 信任政策來限制 Run Command 動作

為可存取 文件的使用者建立限制性 AWS Identity and Access Management (IAM) 政策。IAM 政策決定使用者可在 Amazon Elastic Compute Cloud (Amazon EC2) 主控台中看到哪些 SSM 文件，或使用 `ListDocuments` AWS Command Line Interface (AWS CLI) 或 呼叫 AWS Tools for Windows PowerShell。此政策也限制使用者在 SSM 文件上可執行哪些動作。您可以建立嚴格的政策，讓使用者只能使用特定的文件。如需詳細資訊，請參閱客戶受管政策範例。

請小心使用共用的 SSM 文件

檢閱每個與您共享的文件內容，特別是公有文件。了解那些將在您的執行個體上執行的指令。文件執行後可能會有意或無意地產生負面影響。如果文件參考外部網路、先檢閱外部來源，然後再使用文件。

使用文件雜湊來傳送命令

當您共用文件，系統會建立一個 SHA-256 雜湊並將其指派給文件。系統還會儲存文件內容的快照。當您使用共用文件來傳送命令，您可以在您的命令中指定雜湊以確保以下條件為真：

- 您正在執行正確的 Systems Manager 文件的命令

- 自從與您共用文件後，文件內容尚未變更。

如果雜湊不符合指定的文件或者如果共用文件的內容被更改過，命令會傳回 `InvalidDocument` 的錯誤例外訊息。雜湊無法從外部位置驗證文件的內容。

封鎖 SSM 文件的公有共用

開始之前，請檢閱 AWS 帳戶 中所有公開共用的 SSM 文件，並確認是否要繼續共用它們。若要停止公開共用 SSM 文件，必須依照本主題的修改共用 SSM 文件的許可章節所述修改文件許可設定。開啟封鎖公開共用設定不會影響目前正在公開共用的任何 SSM 文件。啟用封鎖公開共用設定後，您將無法公開共用任何其他 SSM 文件。

除非您的使用案例需要公開共用文件，否則建議開啟 SSM 文件的封鎖公開共用設定。開啟此設定可避免 SSM 文件遭到不必要的存取。封鎖公開共用設定是帳戶層級設定，每個層級都可能不同 AWS 區域。完成下列任務，為目前未共用的 SSM 文件封鎖公開共用。

封鎖公有共用 (主控台)

若要封鎖 SSM 文件的公有共用

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Documents (文件)。
3. 選擇 Preferences (偏好設定)，然後在 Block public sharing (封鎖公有共用) 區段中選擇 Edit (編輯)。
4. 選取 Block public sharing (封鎖公有共用) 核取方塊，再選擇 Save (儲存)。

封鎖公有共用 (命令列)

在本機 AWS Tools for Windows PowerShell 電腦上開啟 AWS Command Line Interface (AWS CLI) 或 ，並執行下列命令來封鎖 SSM 文件的公開共用。

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
```

使用以下命令來確認設定值已更新。

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
```

限制存取以封鎖與 IAM 的公有共用

您可以建立 AWS Identity and Access Management (IAM) 政策，以限制使用者修改區塊公開共用設定。這可防止使用者允許 SSM 文件的不必要存取。

以下是防止使用者更新封鎖公有共用設定的 IAM 政策範例。若要使用此範例，您必須用您自己的帳戶 ID 取代範例 Amazon Web Services 帳戶 ID。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "ssm:UpdateServiceSetting",
            "Resource": "arn:aws:ssm:*:987654321098:servicesetting/ssm/documents/
console/public-sharing-permission"
        }
    ]
}
```

共用 SSM 文件

您可以使用 Systems Manager 主控台來共用 AWS Systems Manager (SSM) 文件。從主控台共用文件時，只能共用文件的預設版本。您也可以透過使用 (AWS CLI) 或 AWS SDK 呼叫 `ModifyDocumentPermission` API AWS Command Line Interface 操作 AWS Tools for Windows PowerShell，以程式設計方式共用 SSM 文件。在共用文件之前，先取得您要與其共用的人員的 AWS 帳戶 ID。當您要共享文件時，您可以指定這些帳戶 ID。

分享文件 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Documents (文件)。
3. 在文件清單中，選擇要共享的文件，然後選擇 View details (查看詳細資訊)。在 Permissions (許可) 索引標籤中，驗證您是文件的擁有者。只有文件擁有者可以分享文件。
4. 選擇編輯。
5. 若要公開共享命令，選擇 Public (公有)，然後選擇 Save (儲存)。若要私下共用命令，選擇 Private (私有)，輸入 AWS 帳戶 ID，選擇 Add permission (新增許可)，然後選擇 Save (儲存)。

分享文件 (命令列)

下列程序需要您 AWS 區域 為命令列工作階段指定 。

1. 在本機 AWS Tools for Windows PowerShell 電腦上開啟 AWS CLI 或 ，並執行下列命令來指定您的登入資料。

   在下列命令中，用您自己的資訊取代*##*。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

   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. 使用下列命令列出所有可供您使用的 SSM 文件。清單中包含您建立和與您共享的文件。

   Linux & macOS

   ```
   aws ssm list-documents
   ```

   Windows

   ```
   aws ssm list-documents
   ```

PowerShell

```
Get-SSMDocumentList
```

3. 使用下列命令來取得特定的文件：

Linux & macOS

```
aws ssm get-document \
    --name document name
```

Windows

```
aws ssm get-document ^
    --name document name
```

PowerShell

```
Get-SSMDocument `
    -Name document name
```

4. 請使用下列命令取得文件的說明。

Linux & macOS

```
aws ssm describe-document \
    --name document name
```

Windows

```
aws ssm describe-document ^
    --name document name
```

PowerShell

```
Get-SSMDocumentDescription `
    -Name document name
```

5. 使用下列的指令來查看文件的許可權限：

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. 使用下列的指令來修改文件的許可權限並共享文件。您必須文件的擁有者才能夠編輯許可權限。對於與特定 AWS 帳戶 ID 共用的文件，也可以使用 `--shared-document-version` 參數指定要共用的文件版本。如果您未指定版本，則會共用文件的 `Default` 版本。如果您與 `all` 公開共用文件，則預設為共用指定文件的所有版本。下列範例命令會根據該人員的 AWS 帳戶 ID，私下與特定個人共用文件。

   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. 使用下列命令來公開共享的文件：

> **Note**
>
> 如果您與 `all` 公開共用文件，則預設為共用指定文件的所有版本。

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')
```

修改共用 SSM 文件的許可

如果您共用命令，使用者可以檢視和使用該命令，直到您移除對 AWS Systems Manager (SSM) 文件的存取權或刪除 SSM 文件為止。不過，您無法刪除正在共享的文件。您必須先停止共享，然後才能將其刪除止。

停止分享文件 (主控台)

停止共享文件

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Documents (文件)。
3. 在文件清單中，選擇要停止共用的文件，然後選擇詳細資訊。在許可區段中，確認您是文件擁有者。只有文件擁有者可以停止共享文件。
4. 選擇編輯。
5. 選擇 X 刪除不應再存取命令的 AWS 帳戶 ID，然後選擇儲存。

停止分享文件 (命令列)

在本機 AWS Tools for Windows PowerShell 電腦上開啟 AWS CLI 或 ，並執行下列命令來停止共用命令。

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
```

使用共用的 SSM 文件

當您共用 AWS Systems Manager (SSM) 文件時，系統會產生 Amazon Resource Name (ARN)，並將其指派給命令。如果您從 Systems Manager 主控台選取和執行共用文件，則不會看到 ARN。但是，如果您想要使用 Systems Manager 主控台以外的其他方法執行共用的 SSM 文件，則必須為 DocumentName 請求參數指定文件的完整 ARN。當您執行命令列出文件時，系統會顯示 SSM 文件的完整 ARN。

> **Note**
>
> 您不需要為 AWS 公有文件 （ 以 開頭的文件AWS-*) 或您擁有的文件指定 ARNs。

使用共用的 SSM 文件 (命令列)

列出所有公有 SSM 文件

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)
```

列出已與您共用的私有 SSM 文件

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 文件

Linux & macOS

```
aws ssm list-documents
```

Windows

```
aws ssm list-documents
```

PowerShell

```
Get-SSMDocumentList
```

取得已與您共用的 SSM 文件的相關資訊

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
```

執行共用的 SSM 文件

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
```

搜尋 SSM 文件

您可以使用任意文字搜尋或篩選條件搜尋，來搜尋 AWS Systems Manager (SSM) 文件存放區中的 SSM 文件。您可以收藏文件，以協助您尋找常用的 SSM 文件。下列各節說明如何使用這些功能。

使用任意文字搜尋

Systems Manager 文件頁面中的搜尋方塊支援任意文字搜尋。任意文字搜尋會根據每個 SSM 文件中的文件名稱來比較搜尋詞語或您輸入的詞語。如果您輸入單個搜尋詞語，例如 **`ansible`**，則 Systems Manager 會傳回含有此詞語的所有 SSM 文件。如果您輸入多個搜尋詞語，則 Systems Manager 會使用 `OR` 陳述式進行搜尋。例如，如果您指定 **`ansible`** 和 **`linux`**，然後搜尋會傳回其名稱中含有任何一個關鍵字的所有文件。

如果您輸入任意文字搜尋詞語，並選擇搜尋選項，例如 平台類型，則搜尋會使用 `AND` 陳述式，並傳回名稱中包含關鍵字和指定平台類型的所有文件。

> **Note**
>
> 請注意下列有關任意文字搜尋的詳細資訊。
>
> - 任意文字搜尋不區分大小寫。
> - 搜尋詞語要求最少三個字元，最多 20 個字元。
> - 任意文字搜尋最多可接受五個搜尋詞語。
> - 如果您在搜尋詞語之間輸入空格，系統會在搜尋時包含空格。
> - 您可以將任意文字搜尋與其他搜尋選項 (例如文件類型或平台類型) 結合在一起。
> - 文件名稱字首篩選條件和任意文字搜尋不能一起使用。它們相互排斥。

若要搜尋 SSM 文件

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Documents (文件)。
3. 在搜尋方塊中輸入您的搜尋詞語，然後按 Enter 鍵。

使用 執行任意文字文件搜尋 AWS CLI

若要使用 CLI 執行任意文字文件搜尋

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

2. 若要使用單一詞語執行任意文字文件搜尋，請執行以下命令。在此命令中，用您自己的資訊取代 *search_term*。

```
aws ssm list-documents --filters Key="SearchKeyword",Values="search_term"
```

範例如下。

```
aws ssm list-documents --filters Key="SearchKeyword",Values="aws-asg" --region us-
east-2
```

若要使用多個詞語進行搜尋 (建立 AND 陳述式)，執行下列命令。在此命令中，用您自己的資訊取代 *search_term_1* 和 *search_term_2*。

```
aws ssm list-documents --filters
 Key="SearchKeyword",Values="search_term_1","search_term_2","search_term_3" --
region us-east-2
```

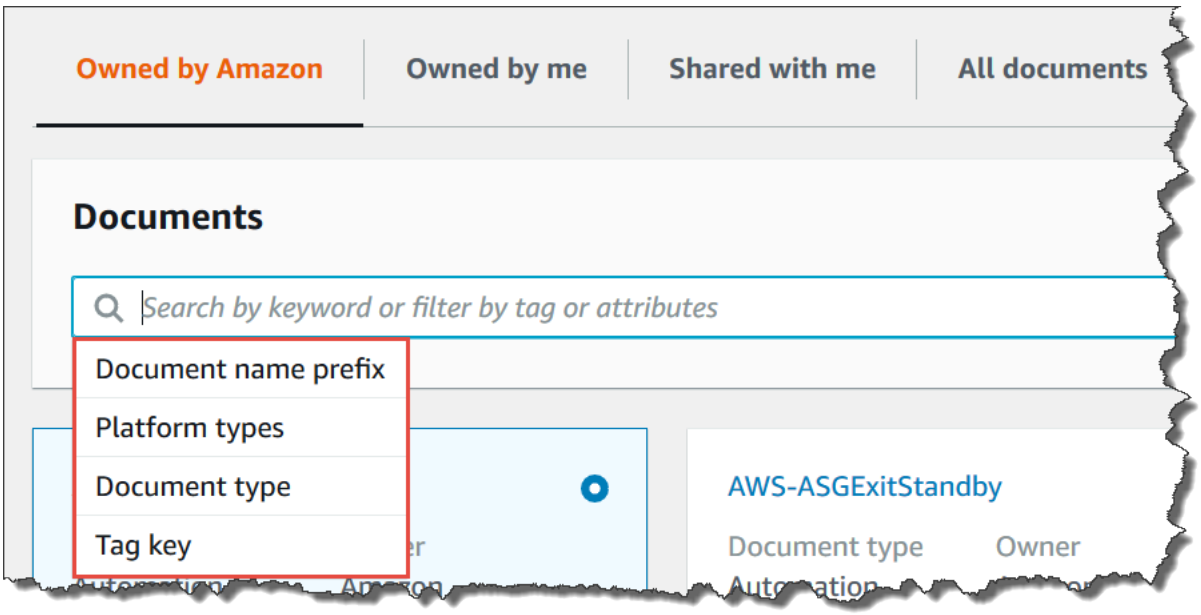
範例如下。

```
aws ssm list-documents --filters Key="SearchKeyword",Values="aws-asg","aws-
ec2","restart" --region us-east-2
```

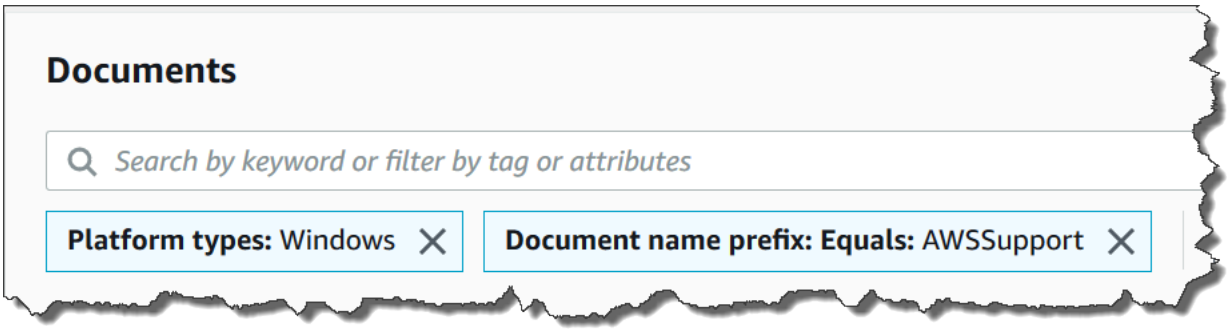
使用篩選條件

Systems Manager 文件頁面會在您選擇搜尋方塊時自動顯示下列篩選條件。

- 文件名稱字首
- 平台類型
- 文件類型
- 標籤鍵



您可以使用單一篩選條件來搜尋 SSM 文件。如果您想要傳回更具體的 SSM 文件集，可以套用多個篩選條件。以下是使用平台類型與文件名稱字首篩選條件的搜尋範例。



如果您套用多個篩選條件，Systems Manager 會根據您選擇的篩選條件建立不同的搜尋陳述式：

- 如果您多次套用相同篩選條件，例如文件名稱字首，Systems Manager 會使用 OR 陳述式進行搜尋。例如，如果您指定的一個篩選條件為文件名稱字首=**AWS**，第二個篩選條件為文件名稱字首=**Lambda**，則搜尋會傳回字首為 "AWS" 的所有文件以及字首為 "Lambda" 的所有文件。
- 如果套用不同的篩選條件，例如 Document name prefix (文件名稱字首) 和 Platform types (平台類型)，則 Systems Manager 會使用 AND 陳述式進行搜尋。例如，如果指定 Document name prefix (文件名稱字首) = **AWS** 篩選條件、Platform types (平台類型) = **Linux** 篩選條件，則搜尋會傳回特定於 Linux 平台且字首為「AWS」的所有文件。

> **Note**
>
> 使用篩選條件的搜尋會區分大小寫。

將文件新增至收藏

為了協助您尋找常用的 SSM 文件，請將文件新增至收藏。每個文件類型、每個 AWS 帳戶 和 最多可以收藏 20 個文件 AWS 區域。您可以從文件 AWS Management Console選擇、修改及檢視收藏。下列程序說明如何選擇、修改和檢視收藏。

收藏 SSM 文件

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Documents (文件)。
3. 選擇要收藏的文件名稱旁的星形圖示。

從收藏移除 SSM 文件

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Documents (文件)。
3. 取消選擇要移出收藏的文件名稱旁的星形圖示。

從文件檢視您的最愛 AWS Management Console

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Documents (文件)。
3. 選取收藏索引標籤。

AWS Systems Manager Maintenance Windows

Maintenance Windows中的工具，可協助您定義何時對節點執行潛在破壞性動作的排程 AWS Systems Manager，例如修補作業系統、更新驅動程式，或安裝軟體或修補程式。

> **Note**
>
> State Manager 和 Maintenance Windows 可以在受管節點上執行某些類似的更新。您選擇哪一項，取決於您是否需要在指定的期間內自動化系統合規，或執行高優先順序、時間敏感的任務。
> 如需詳細資訊，請參閱在 State Manager 與 Maintenance Windows 之間進行選擇。

使用 Maintenance Windows，您可以在許多其他 AWS 資源類型上排程動作，例如 Amazon Simple Storage Service (Amazon S3) 儲存貯體、Amazon Simple Queue Service (Amazon SQS) 佇列、AWS Key Management Service (AWS KMS) 金鑰等。

如需您可以在維護時段目標中包含的支援資源類型完整清單，請參閱AWS Resource Groups 《 使用者指南》中的可與 AWS Resource Groups 和標籤編輯器搭配使用的資源。若要開始使用 Maintenance Windows，請開啟 Systems Manager 主控台。在導覽窗格中，選擇 Maintenance Windows。

每個維護時段都有排程、最長持續時間、一組已註冊的目標 （ 受管節點或其他採取動作 AWS 的資源 ），以及一組已註冊的任務。當您建立或更新維護時段時，可以將標籤新增到您的維護時段。(標籤為索引鍵，可幫助識別和排序組織內的資源。) 您也可以指定不得在某日期之前或之後執行維護時段，亦可指定維護時段排程所依據的國際時區。

如需維護時段的各種排程相關選項彼此之間有何關聯的說明，請參閱 維護時段排程與作用期間選項。

如需使用 `--schedule` 選項的詳細資訊，請參閱參考：Systems Manager 的 Cron 和 Rate 運算式。

受支援任務類型

透過維護時段，您可以執行四種類型的任務：

- 中的命令Run Command，Systems Manager 中的工具

  如需有關 Run Command 的詳細資訊，請參閱 AWS Systems Manager Run Command。
- Automation 中的工作流程，Systems Manager 中的工具

  如需自動化工作流程的相關詳細資訊，請參閱 AWS Systems Manager 自動化。
- 中的函數 AWS Lambda

  如需有關 Lambda 函數的資訊，請參閱《AWS Lambda 開發人員指南》中的 Lambda 入門。
- 中的任務 AWS Step Functions

> **Note**
>
> 維護時段任務僅支援 Step Functions 標準狀態機器工作流程。這些任務不支援快速狀態機器工作流程。如需有關狀態機器工作流程類型的資訊，請參閱《AWS Step Functions 開發人員指南》中的標準與快速工作流程。

如需 Step Functions 的詳細資訊，請參閱《AWS Step Functions 開發人員指南》。

這表示您可以使用維護時段對所選目標執行類似下列任務。

- 安裝或更新應用程式。
- 套用修補程式。
- 安裝或更新 SSM Agent
- 藉由使用 Systems Manager Run Command 任務執行 PowerShell 命令和 Linux shell 指令碼。
- 透過使用 Systems Manager Automation 任務以建置 Amazon Machine Images (AMIs)、啟動軟體和設定節點。
- 執行叫用其他動作的 AWS Lambda 函數，例如掃描節點以取得修補程式更新。
- 執行 AWS Step Functions 狀態機器來執行任務，例如從 Elastic Load Balancing 環境移除節點、修補節點，然後將節點新增回 Elastic Load Balancing 環境。
- 透過指定 AWS 資源群組做為目標，離線的目標節點。

> **Note**
>
> 必須為維護時段 Run Command 類型任務指定一或多個目標。視任務而定，其他維護時段任務類型 （ 自動化 AWS Lambda和 AWS Step Functions) 的目標為選用。如需有關執行未指定目標之任務的詳細資訊，請參閱 註冊不含目標的維護時段任務。

支援 EventBridge

Amazon EventBridge 規則中支援此 Systems Manager 工具做為事件類型。如需詳細資訊，請參閱 使用 Amazon EventBridge 監控 Systems Manager 及 參考：Systems Manager 的 Amazon EventBridge 事件模式和類型。

目錄

- 設定 Maintenance Windows
- 使用主控台建立和管理維護時段
- 教學課程
- 註冊維護時段任務時使用虛擬參數
- 維護時段排程與作用期間選項
- 註冊不含目標的維護時段任務
- 對維護時段進行故障診斷

設定 Maintenance Windows

在 中的使用者 AWS 帳戶 可以使用 中的工具Maintenance Windows來建立和排程維護時段任務之前 AWS Systems Manager，必須先授予他們必要的許可。此外，您必須建立維護時段的 IAM 服務角色，以及要附加至其中的 IAM 政策。

開始之前

除了在本節中設定的許可，IAM 實體 (將與維護時段搭配使用的使用者、角色或群組) 也應具有了一般維護時段許可。您可以透過將 IAM 政策 `AmazonSSMFullAccess` 指派給實體來授予這些許可，或透過指派自訂 IAM 政策來授予這些許可，而自訂政策能為 Systems Manager 提供較小存取許可集 (許可應涵蓋維護時段任務)。

主題

- 使用主控台控制對維護時段的存取權
- 使用 控制維護時段的存取 AWS CLI

使用主控台控制對維護時段的存取權

下列程序說明如何使用 AWS Systems Manager 主控台建立維護時段所需的許可和角色。

主題

- 任務 1：使用主控台為維護時段服務角色建立自訂政策
- 任務 2：使用主控台為維護時段建立自訂服務角色
- 任務 3：使用主控台向指定使用者授予註冊維護時段任務的許可
- 任務 4：使用主控台防止指定的使用者註冊維護時段任務

任務 1：使用主控台為維護時段服務角色建立自訂政策

維護時段任務需要 IAM 角色提供在目標資源上執行所需的許可。許可透過連接到角色的 IAM 政策提供。您執行的任務類型及其他的操作要求決定了此政策的內容。我們提供您可以根據需求調整的基本政策。根據維護時段執行的任務及任務類型，您可能不需要此政策中的所有許可，而您可能需要包含其他的許可。您可以將此政策附加至稍後在 任務 2：使用主控台為維護時段建立自訂服務角色 建立的角色。

使用主控台建立自訂政策

1. 在 https://console.aws.amazon.com/iam/ 中開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Policies (政策)，然後選擇 Create policy (建立政策)。
3. 在政策編輯器區段中，選擇 JSON。
4. 將預設內容取代為以下內容：

```
{
    "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::account-id:role/maintenance-window-role-
name",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ssm.amazonaws.com"
                    ]
                }
            }
```

```
        }
    ]
}
```

5. 根據您在帳戶中執行的維護任務及其需要來修改 JSON 內容。您所做的變更專為您的操作而規劃。

   例如：

   - 您可以為特定功能及狀態機器提供 Amazon Resource Names (ARNs)，而不是使用萬用字圓 (*) 限定詞。
   - 如果您不打算執行 AWS Step Functions 任務，您可以移除`states`許可 和 ARNs)。
   - 如果您不打算執行 AWS Lambda 任務，您可以移除`lambda`許可和 ARNs。
   - 如果您不打算執行自動化任務，您可以移除 `ssm:GetAutomationExecution` 及 `ssm:StartAutomationExecution` 許可。
   - 新增執行任務可能需要的其他許可。例如，有些自動化動作搭配 AWS CloudFormation 堆疊運作。因此，`cloudformation:CreateStack`、`cloudformation:DescribeStacks` 以及 `cloudformation:DeleteStack` 許可是必要的。

     另一個例子：Automation Runbook `AWS-CopySnapshot` 需建立 Amazon Elastic Block Store (Amazon EBS) 快照的權限。因此，服務角色需要 `ec2:CreateSnapshot` 許可。

     如需 Automation Runbook 所需的角色權限資訊，請參閱 AWS Systems Manager Automation Runbook 參考資料中的 Runbook 描述。

6. 完成政策修訂後，選擇下一步。
7. 在政策名稱中輸入名稱，該名稱將政策標識為連接至所建立之服務角色的政策。例如：**`my-maintenance-window-role-policy`**。
8. (選用) 在新增標籤區段中，新增一個或多個標籤鍵值對來組織、追蹤或控制對此政策的存取。
9. 選擇建立政策。

   記下為政策指定的名稱。您可以在接下來的程序 任務 2：使用主控台為維護時段建立自訂服務角色 加以引用。

任務 2：使用主控台為維護時段建立自訂服務角色

在先前任務中建立的政策會連接至您在此任務中建立的維護時段服務角色。使用者註冊維護時段任務時，他們將此 IAM 角色指定為任務組態的一部分。此角色的權限可讓 Systems Manager 代您在維護時段執行任務。

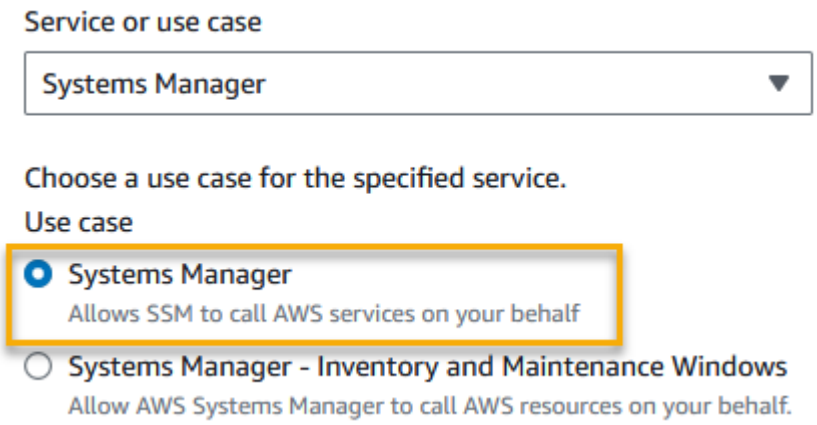
> **Important**
>
> 先前，Systems Manager 主控台可讓您選擇要`AWSServiceRoleForAmazonSSM`用作任務維護角色的 AWS 受管 IAM 服務連結角色。不再建議將此角色及其關聯政策`AmazonSSMServiceRolePolicy`，用於維護時段任務。如果您現在將此角色用於維護時段任務，我們建議您停止使用。相反，請建立您自己的 IAM 角色，以便在執行維護時段任務時，Systems Manager 可跟其他 AWS 服務 溝通。

請使用下列步驟來為 Maintenance Windows 建立自訂服務角色，讓 Systems Manager 能代表您執行 Maintenance Windows 任務。您需要將在前一任務中建立的政策連接至所建立的自訂服務角色。

使用主控台為維護時段建立自訂服務角色

1. 在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。
2. 在導覽窗格中，選擇角色，然後選擇建立角色。
3. 對於 Select trusted entity (選擇信任的實體)，請執行以下選項：
    1. 針對信任的實體類型，請選擇 AWS 服務。
    2. 對於使用案例，選擇 Systems Manager
    3. 選擇 Systems Manager。

        下圖反白顯示了 Systems Manager 選項的位置。



4. 選擇下一步。
5. 在許可政策區域的搜尋方塊中，輸入在 任務 1：使用主控台為維護時段服務角色建立自訂政策 中建立的政策名稱，選取其名稱旁邊的方塊，然後選擇下一步。
6. 在 Role name (角色名稱) 中，輸入識別此角色為 Maintenance Windows 角色的名稱。例如：**my-maintenance-window-role**。
7. (選用) 變更預設的角色描述以反映此角色的用途。例如：**Performs maintenance window tasks on your behalf**。
8. 對於步驟 1：選取信任的實體，確認下列政策顯示在信任的政策方塊中。

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

9. 對於步驟 2：新增許可，確認您在 任務 1：使用主控台為維護時段服務角色建立自訂政策 中建立的政策存在。
10. (選用) 在步驟 3：新增標籤中，新增一個或多個標籤鍵值對來組織、追蹤或控制對此角色的存取。
11. 選擇 Create role (建立角色)。系統會讓您回到 Roles (角色) 頁面。

12. 選擇剛建立之 IAM 角色的名稱。
13. 複製或記下角色名稱及摘要區的 ARN 值。您帳戶的使用者在建立維護時段時指定此訊息。

任務 3：使用主控台向指定使用者授予註冊維護時段任務的許可

為使用者提供存取維護時段自訂服務角色的許可，可讓他們將其用於維護時段任務。這是除了您已授予的許可，以使用 Maintenance Windows工具的 Systems Manager API 命令。此 IAM 角色傳遞了執行維護時段任務所需的許可。因此，如果無法傳遞這些 IAM 許可，使用者就無法使用您的自訂服務角色向維護時段註冊任務。

在向維護時段註冊任務時，需要指定服務角色來執行實際的任務操作。這是服務代您執行任務時所擔任的角色。在此之前，若要註冊任務本身，請將 IAM `PassRole` 政策指派給 IAM 實體 (例如使用者或群組)。這樣便讓 IAM 實體可以指定執行任務時應使用的角色，而指定角色是向維護時段註冊這些任務的其中一步。如需相關資訊，請參閱《IAM 使用者指南》中的 Grant a user permissions to pass a role to an AWS 服務。

設定許可以允許使用者註冊維護時段任務

如果 IAM 實體 (使用者、角色或群組) 設定為具有管理員許可，則該 IAM 使用者或角色可以存取維護時段。對於沒有管理員許可的 IAM 實體，管理員必須將以下許可授予給 IAM 實體。以下是在維護時段中註冊任務所需的最低許可：

- `AmazonSSMFullAccess` 受管政策或提供相當許可的政策。
- 以下是 `iam:PassRole` 和 `iam:ListRoles` 許可。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::account-id:role/my-maintenance-window-role"
        },
        {
            "Effect": "Allow",
            "Action": "iam:ListRoles",
            "Resource": "arn:aws:iam::account-id:role/"
        },
        {
            "Effect": "Allow",
```

```
            "Action": "iam:ListRoles",
            "Resource": "arn:aws:iam::account-id:role/aws-service-role/
ssm.amazonaws.com/"
        }
    ]
}
```

my-maintenance-window-role 代表您先前建立之自訂維護時段服務角色的名稱。

account-id 代表您的 AWS 帳戶 ID。新增此資源 `arn:aws:iam::account-id:role/` 的許可，可讓使用者在建立維護時段任務時，檢視主控台中的客戶角色，並從中選擇客戶角色。新增 `arn:aws:iam::account-id:role/aws-service-role/ssm.amazonaws.com/` 的許可，可讓使用者在建立維護時段任務時，在主控台中選擇 Systems Manager 服務連結的角色。

若要提供存取權，請新增權限至您的使用者、群組或角色：

- 中的使用者和群組 AWS IAM Identity Center：

  建立權限合集。請按照 AWS IAM Identity Center 使用者指南 中的 建立權限合集 說明進行操作。

- 透過身分提供者在 IAM 中管理的使用者：

  建立聯合身分的角色。遵循「IAM 使用者指南」的為第三方身分提供者 (聯合) 建立角色中的指示。

- IAM 使用者：
  - 建立您的使用者可擔任的角色。請按照「IAM 使用者指南」的為 IAM 使用者建立角色中的指示。
  - (不建議) 將政策直接附加至使用者，或將使用者新增至使用者群組。請遵循 IAM 使用者指南的新增許可到使用者 (主控台) 中的指示。

使用主控台為可以註冊維護時段任務的群組設定許可

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 User groups (使用者群組)。
3. 在群組清單中，選取要為其指派 `iam:PassRole` 許可的群組的名稱，或先建立新的群組 (如有必要)
4. 在 Permissions (許可) 索引標籤上，選擇 Add permissions, Create inline policy (新增許可、建立內嵌政策)。
5. 在政策編輯器區域中，選擇 JSON 並使用列內容取代方塊中的預設內容。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::account-id:role/my-maintenance-window-role"
        },
        {
            "Effect": "Allow",
            "Action": "iam:ListRoles",
            "Resource": "arn:aws:iam::account-id:role/"
        },
        {
            "Effect": "Allow",
            "Action": "iam:ListRoles",
            "Resource": "arn:aws:iam::account-id:role/aws-service-role/
ssm.amazonaws.com/"
        }
    ]
}
```

my-maintenance-window-role 代表您先前建立之自訂維護時段角色的名稱。

account-id 代表您的 AWS 帳戶 ID。新增此資源 arn:aws:iam::*account-id*:role/ 的許可，可讓使用者在建立維護時段任務時，檢視主控台中的客戶角色，並從中選擇客戶角色。新增 arn:aws:iam::*account-id*:role/aws-service-role/ssm.amazonaws.com/ 的許可，可讓使用者在建立維護時段任務時，在主控台中選擇 Systems Manager 服務連結的角色。

6. 選擇下一步。
7. 在檢閱並建立頁面上，在政策名稱方塊中輸入名稱以識別此 PassRole 政策 (例如 **my-group-iam-passrole-policy**)，然後選擇建立政策。

任務 4：使用主控台防止指定的使用者註冊維護時段任務

您可以拒絕 AWS 帳戶 您不想向維護時段註冊任務之 中使用者的ssm:RegisterTaskWithMaintenanceWindow許可。這針對不應註冊維護時段任務的使用者提供了額外的保護層。

使用主控台為無法註冊維護時段任務的群組設定許可

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 User groups (使用者群組)。
3. 在群組清單中，選取要為其拒絕 `ssm:RegisterTaskWithMaintenanceWindow` 許可的群組名稱，或先建立新的群組 (如有必要)。
4. 在 Permissions (許可) 索引標籤上，選擇 Add permissions, Create inline policy (新增許可、建立內嵌政策)。
5. 在政策編輯器區域中，選擇 JSON，然後使用下列內容取代方塊中的預設內容。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "ssm:RegisterTaskWithMaintenanceWindow",
            "Resource": "*"
        }
    ]
}
```

6. 選擇下一步。
7. 在檢閱和建立頁面上，在政策名稱中輸入名稱以識別此政策 (例如 **`my-groups-deny-mw-tasks-policy`**)，然後選擇建立政策。

使用 控制維護時段的存取 AWS CLI

下列程序說明如何使用 AWS Command Line Interface (AWS CLI) 為 中的Maintenance Windows工具建立所需的許可和角色 AWS Systems Manager。

主題

- 任務 1：建立 JSON 格式的信任政策和客戶管理政策檔案
- 任務 2：使用 建立和驗證維護時段的自訂服務角色 AWS CLI
- 任務 3：使用 AWS CLI向指定使用者授予註冊維護時段任務的許可
- 任務 4：防止指定的使用者使用 註冊維護時段任務 AWS CLI

任務 1：建立 JSON 格式的信任政策和客戶管理政策檔案

維護時段任務需要 IAM 角色提供在目標資源上執行所需的許可。許可透過連接到角色的 IAM 政策提供。您執行的任務類型及其他的操作要求決定了此政策的內容。我們提供您可以根據需求調整的基本政策。根據維護時段執行的任務及任務類型，您可能不需要此政策中的所有許可，而您可能需要包含其他的許可。

在此任務中，您在一組 JSON 檔案中指定自訂維護時段角色所需的許可。您可以將此政策附加至稍後在 任務 2：使用 建立和驗證維護時段的自訂服務角色 AWS CLI 建立的角色。

建立信任政策和客戶管理政策檔案

1. 將下列信任政策複製並貼入一個文字檔。使用以下名稱和副檔名儲存此檔案：**mw-role-trust-policy.json**。

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

2. 複製下列 JSON 政策並將其貼到另一個文字檔案中。在您建立第一個檔案的相同目錄中，使用下列名稱和副檔名儲存此檔案：**mw-role-custom-policy.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::account-id:role/maintenance-window-role-
name",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ssm.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

3. 根據您在帳戶中執行的維護任務需求來修改 `mw-role-custom-policy.json` 的內容。您所做的變更專為您的操作而規劃。

   例如：

   - 您可以為特定功能及狀態機器提供 Amazon Resource Names (ARNs)，而不是使用萬用字圓 (*) 限定詞。
   - 如果您不打算執行 AWS Step Functions 任務，您可以移除`states`許可 和 ARNs)。
   - 如果您不打算執行 AWS Lambda 任務，您可以移除`lambda`許可和 ARNs。
   - 如果您不打算執行自動化任務，您可以移除 `ssm:GetAutomationExecution` 及 `ssm:StartAutomationExecution` 許可。
   - 新增執行任務可能需要的其他許可。例如，有些自動化動作搭配 AWS CloudFormation 堆疊運作。因此，`cloudformation:CreateStack`、`cloudformation:DescribeStacks` 以及 `cloudformation:DeleteStack` 許可是必要的。

     另一個例子：Automation Runbook `AWS-CopySnapshot` 需建立 Amazon Elastic Block Store (Amazon EBS) 快照的權限。因此，服務角色需要 `ec2:CreateSnapshot` 許可。

     如需 Automation Runbook 所需的角色權限資訊，請參閱 AWS Systems Manager Automation Runbook 參考資料中的 Runbook 描述。

   進行任何所需變更後，再次儲存檔案。

任務 2：使用 建立和驗證維護時段的自訂服務角色 AWS CLI

在先前任務中建立的政策會連接至您在此任務中建立的維護時段服務角色。使用者註冊維護時段任務時，他們將此 IAM 角色指定為任務組態的一部分。此角色的權限可讓 Systems Manager 代您在維護時段執行任務。

> **Important**
>
> 先前，Systems Manager 主控台可讓您選擇要AWSServiceRoleForAmazonSSM用作任務維護角色的 AWS 受管 IAM 服務連結角色。不再建議將此角色及其關聯政策 AmazonSSMServiceRolePolicy，用於維護時段任務。如果您現在將此角色用於維護時段任務，我們建議您停止使用。相反，請建立您自己的 IAM 角色，以便在執行維護時段任務時，Systems Manager 可跟其他 AWS 服務 溝通。

在此任務中，您需執行 CLI 命令來建立維護時段服務角色，並從您建立的 JSON 檔案新增政策內容。

使用 AWS CLI建立用於維護時段的自訂服務角色

1. 開啟 AWS CLI ，並在您放置 mw-role-custom-policy.json 和 的目錄中執行下列命令mw-role-trust-policy.json。該命令會建立名為 my-maintenance-window-role 的維護時段服務角色，並將信任政策連接至該角色。

   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
   ```

   系統會傳回與以下相似的資訊。

   ```
   {
       "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**
>
> 記下 RoleName 與 Arn 值。它們可以包含在下一個命令中。

2. 執行下列命令，將客戶管理政策連接至該角色。使用自己的 AWS 帳戶 ID 取代 *account-id* 預留位置

   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. 執行下列命令，確認角色已建立，並且已連接信任政策。

```
aws iam get-role --role-name my-maintenance-window-role
```

此命令會傳回與以下內容相似的資訊：

```
{
    "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. 執行下列命令，確認客戶管理政策已連接至角色。

```
aws iam list-attached-role-policies --role-name my-maintenance-window-role
```

此命令會傳回與以下內容相似的資訊：

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "mw-role-custom-policy",
```

```
            "PolicyArn": "arn:aws:iam::123456789012:policy/mw-role-custom-policy"
        }
    ]
}
```

任務 3：使用 AWS CLI向指定使用者授予註冊維護時段任務的許可

為使用者提供存取維護時段自訂服務角色的許可，可讓他們將其用於維護時段任務。這是除了您已授予的許可，以使用 Maintenance Windows工具的 Systems Manager API 命令。此 IAM 角色傳遞了執行維護時段任務所需的許可。因此，如果無法傳遞這些 IAM 許可，使用者就無法使用您的自訂服務角色向維護時段註冊任務。

在向維護時段註冊任務時，需要指定服務角色來執行實際的任務操作。這是服務代您執行任務時所擔任的角色。在此之前，若要註冊任務本身，請將 IAM `PassRole` 政策指派給 IAM 實體 (例如使用者或群組)。這樣便讓 IAM 實體可以指定執行任務時應使用的角色，而指定角色是向維護時段註冊這些任務的其中一步。如需相關資訊，請參閱《IAM 使用者指南》中的 Grant a user permissions to pass a role to an AWS 服務。

為允許使用者使用 註冊維護時段任務的使用者設定許可 AWS CLI

1. 將下列 AWS Identity and Access Management (IAM) 政策複製並貼到文字編輯器中，並使用下列名稱和副檔名儲存：`mw-passrole-policy.json`。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::account-id:role/my-maintenance-window-role"
        },
        {
            "Effect": "Allow",
            "Action": "iam:ListRoles",
            "Resource": "arn:aws:iam::account-id:role/"
        },
        {
            "Effect": "Allow",
            "Action": "iam:ListRoles",
            "Resource": "arn:aws:iam::account-id:role/aws-service-role/
ssm.amazonaws.com/"
```

```
        }
    ]
}
```

使用您先前建立之自訂維護時段角色的名稱取代 *my-maintenance-window-role*。

將 *account-id* 取代為 AWS 帳戶的 ID。新增此資源 arn:aws:iam::*account-id*:role/ 的許可，可讓群組中的使用者在建立維護時段任務時，檢視主控台中的客戶角色，並從中選擇客戶角色。新增此 arn:aws:iam::*account-id*:role/aws-service-role/ssm.amazonaws.com/ 許可，可讓群組中的使用者在建立維護時段任務時，在主控台中選擇 Systems Manager 服務連結的角色。

2. 開啟 AWS CLI。
3. 根據您是否將許可指派給 IAM 實體 (使用者或群組)，執行下列其中一個命令。

   - 對於 IAM 實體：

     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
     ```

     對於 *user-name*，指定要將任務指派給維護時段的使用者。對於 *policy-name*，指定您要用於識別該政策的名稱，例如 **my-iam-passrole-policy**。對於 *path-to-document*，指定您儲存在步驟 1 的檔案路徑。例如：file://C:\Temp\mw-passrole-policy.json

     > **Note**
     >
     > 若要授予使用者使用 Systems Manager 主控台為維護時段註冊任務的存取權，您還必須將 AmazonSSMFullAccess 政策指派給使用者 (或 IAM 政策，該政策為涵

蓋維護時段任務的 Systems Manager 提供較小的存取許可集)。執行以下命令，將 AmazonSSMFullAccess 政策指派給使用者。

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"
```

- 對於 IAM 群組：

  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
  ```

  對於 *group-name*，指定其成員會將任務指派給維護時段的群組。對於 *policy-name*，指定您要用於識別該政策的名稱，例如 **my-iam-passrole-policy**。對於 *path-to-document*，指定您儲存在步驟 1 的檔案路徑。例如：file://C:\Temp\mw-passrole-policy.json

> **Note**
>
> 若要授與群組成員使用 Systems Manager 主控台為維護時段註冊任務的存取權限，您還必須將 AmazonSSMFullAccess 政策指派給群組。執行以下命令，將此政策指派到您的群組。
>
> 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. 執行以下命令，確認已經將政策指派到群組。

   Linux & macOS

   ```
   aws iam list-group-policies \
       --group-name "group-name"
   ```

   Windows

   ```
   aws iam list-group-policies ^
       --group-name "group-name"
   ```

任務 4：防止指定的使用者使用 註冊維護時段任務 AWS CLI

您可以拒絕 AWS 帳戶 您不想向維護時段註冊任務之 中使用者的ssm:RegisterTaskWithMaintenanceWindow許可。這針對不應註冊維護時段任務的使用者提供了額外的保護層。

無論您是根據個別使用者還是群組來拒絕 ssm:RegisterTaskWithMaintenanceWindow 許可，請使用下列其中一個處理程序，來阻止使用者向維護時段註冊任務。

為不允許使用 註冊維護時段任務的使用者設定許可 AWS CLI

1. 下列 IAM 政策複製並貼到文字編輯器，並以下名稱和副檔名儲存：**deny-mw-tasks-policy.json**。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "ssm:RegisterTaskWithMaintenanceWindow",
            "Resource": "*"
        }
    ]
}
```

2. 開啟 AWS CLI。
3. 根據您是否將許可指派給 IAM 實體 (使用者或群組)，執行下列其中一個命令。

   - 對於使用者：

     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
     ```

     對於 *user-name*，指定要防止其將任務指派給維護時段的使用者。對於 *policy-name*，指定您要用於識別該政策的名稱，例如 **my-deny-mw-tasks-policy**。對於 *path-to-document*，指定您儲存在步驟 1 的檔案路徑。例如：`file://C:\Temp\deny-mw-tasks-policy.json`

   - 對於群組：

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
```

對於 *group-name*，指定要防止其成員將任務指派給維護時段的群組。對於 *policy-name*，指定您要用於識別該政策的名稱，例如 **my-deny-mw-tasks-policy**。對於 *path-to-document*，指定您儲存在步驟 1 的檔案路徑。例如：`file://C:\Temp\deny-mw-tasks-policy.json`

4. 執行以下命令，確認已經將政策指派到群組。

Linux & macOS

```
aws iam list-group-policies \
    --group-name "group-name"
```

Windows

```
aws iam list-group-policies ^
    --group-name "group-name"
```

使用主控台建立和管理維護時段

本節說明如何使用 AWS Systems Manager 主控台建立、設定、更新和刪除維護時段。本節也提供管理維護時段的目標和任務的資訊。

> **Important**
>
> 我們建議您一開始先在測試環境中建立和設定維護時段。

開始之前

在建立維護時段之前，您必須設定 Maintenance Windows中的工具 的存取權 AWS Systems Manager。如需詳細資訊，請參閱設定 Maintenance Windows。

主題

- 使用主控台建立維護時段
- 使用主控台將目標指派至維護時段
- 使用主控台將任務指派至維護時段
- 使用主控台停用或啟用維護時段
- 使用主控台更新或刪除維護時段資源

使用主控台建立維護時段

在此程序中，您會在 中建立維護時段Maintenance Windows，這是 中的工具 AWS Systems Manager。您可以指定其基本選項，例如名稱、排程和持續時間。在後續步驟中，您可以選擇要更新的目標或資源，以及在維護時段執行時執行的任務。

> **Note**
>
> 如需維護時段的各種排程相關選項彼此之間有何關聯的說明，請參閱 維護時段排程與作用期間選項。
> 如需使用 `--schedule` 選項的詳細資訊，請參閱參考：Systems Manager 的 Cron 和 Rate 運算式。

使用主控台建立維護時段

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Maintenance Windows。

3. 選擇建立維護時段。
4. 對於 Name (名稱)，輸入描述名稱，以協助您識別此維護時段。
5. (選用) 在 Description (描述) 中，輸入描述來確定如何使用此維護時段。
6. (選用) 如果您想允許維護時段任務在受管節點上執行 (即使尚未將這些節點註冊為目標)，請選擇 Allow unregistered targets (允許未註冊的目標)。

   如果您選擇此選項，即可在向維護時段註冊任務時選擇未註冊的節點 (依據節點 ID)。

   如果您未選擇此選項，則必須在向維護時段註冊任務時選擇先前註冊過的目標。
7. 使用三個排程選項的其中一個，來為維護時段指定排程。

   如需有關建立 Cron/Rate 運算式的詳細資訊，請參閱參考：Systems Manager 的 Cron 和 Rate 運算式。
8. 針對 Duration (持續時間)，輸入維護時段將執行的時數。您指定的值會根據維護時段的開始時間，決定維護時段的特定結束時間。在產生的結束時間減去您在下一個步驟中為 Stop initiating tasks (停止啟動任務) 指定的小時數過後，將不允許啟動任何維護時段任務。

   例如，如果維護時段從下午 3 點開始，持續時間為三小時，而 Stop initiating tasks (停止啟動任務) 的值為一小時，則在下午 5 點之後無法啟動任何維護時段任務。
9. 針對 Stop initiating tasks (停止初始任務)，輸入在維護時段執行結束之前，系統應該停止排程新任務的時數。
10. (選用) 對於 Window start date (時段開始日期)，依照 ISO-8601 Extended 格式，指定您希望開始啟用維護時段的日期和時間。這可讓您延遲啟用維護時段，直到指定的未來日期為止。

    > **Note**
    >
    > 您無法指定過去的開始日期和時間。

11. (選用) 對於 Window end date (時段結束日期)，依照 ISO-8601 Extended 格式，指定您希望停用維護時段的日期和時間。這可讓您設定不再執行維護時段的未來日期和時間點。
12. (選用) 對於排程時區，依照網際網路號碼分配局 (IANA) 格式，指定排程的維護時段執行時所依據的時區。例如："America/Los_Angeles"、"etc/UTC" 或 "Asia/Seoul"。

    如需有關有效格式的詳細資訊，請參閱 IANA 網站上的時區資料庫有效格式。
13. (選用) 對於 Schedule offset (排程偏移)，請輸入在執行維護時段之前，在 Cron 或 Rate 表達式所指定的日期和時間之後等待的天數。您可以指定一至六天。

> **Note**
>
> 僅當您透過手動輸入 Cron 或 Rate 表達式指定排程時，此選項才可用。

14. (選用) 在 Manage tags (管理標籤) 區域，將一或多個標籤金鑰名稱/值對套用到維護時段。

    標籤是您指派給資源的選用性中繼資料。標籤允許您以不同的方式 (例如用途、擁有者或環境) 將資源分類。例如，您可能想要標記維護時段來識別其執行的任務類型、目標類型以及其執行所在的環境。在這種情況下，您可以指定以下索引鍵名稱/值對：

    - `Key=TaskType,Value=AgentUpdate`
    - `Key=OS,Value=Windows`
    - `Key=Environment,Value=Production`
15. 選擇建立維護時段。系統會帶您回到維護時段頁面。您剛建立的維護時段為 Enabled (已啟用) 狀態。

使用主控台將目標指派至維護時段

在此程序中，您會向維護時段註冊目標。換言之，您會指定維護時段要對哪些資源執行動作。

> **Note**
>
> 如果單一維護時段任務已向多個目標註冊，則其任務叫用會依序發生，而非平行發生。如果您的任務必須同時在多個目標上執行，請個別註冊每個目標的任務，並為每個任務指派相同的優先順序層級。

使用主控台將目標指派至維護時段

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Maintenance Windows。
3. 在維護時段清單中，選擇您要新增目標的維護時段。
4. 選擇 Actions (動作)，然後選擇 Register targets (註冊目標)。
5. (選用) 在 Target Name (目標名稱) 中，輸入目標的名稱。
6. 在描述，請輸入描述。

7. (選用) 對於 Owner information (擁有者資訊)，指定要在此維護時段為這些目標執行任務時引發的任何 Amazon EventBridge 事件中包含的資訊。

   如需使用 EventBridge 監控 Systems Manager 事件的相關資訊，請參閱 使用 Amazon EventBridge 監控 Systems Manager。

8. 在 Targets (目標) 區域，選擇下表中所述的其中一個選項。

選項	描述
指定執行個體標籤	對於 Specify instance tags (指定執行個體標籤) 方塊，請指定一或多個標籤索引鍵和 (選用) 值，這些鍵值已經或即將新增到您帳戶中的受管節點。維護時段執行時，其會對已新增這些標籤的所有受管節點上嘗試執行任務。 如果您指定多個標籤鍵，則節點必須加上您指定的所有標籤鍵和值，才會包含在目標群組中。
手動選擇執行個體	從清單中選取您要在維護時段目標中包含的每個節點方塊。 此清單會包含在您帳戶中已設定為與 Systems Manager 搭配的所有節點。 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。 對於邊緣裝置和內部部署伺服器和虛擬機器 (VM)，請參閱 使用 Systems Manager 在混合多雲端環境中管理節點
選擇資源群組	對於 Resource group (資源群組)，從清單中選擇您帳戶中現有資源群組的名稱。 如需建立和使用資源群組的詳細資訊，請參閱下列主題：

選項	描述
	• 《AWS Resource Groups 使用者指南》中的什麼是資源群組？ • AWS 新聞部落格中的 AWS Resource Groups 和標記 (選用) 對於 Resource types (資源類型)，選擇最多五個可用資源類型，或選擇 All resource types (所有資源類型)。 如果您指派給維護時段的任務沒有對您新增到目標的其中一個資源類型執行動作，系統可能會報告錯誤。儘管發生這些錯誤，找到支援資源類型的任務會持續執行。 例如，假設您將以下資源類型新增到這個目標： • `AWS::S3::Bucket` • `AWS::DynamoDB::Table` • `AWS::EC2::Instance` 但在稍後當您將任務新增至維護時段時，您只包含在節點上執行動作的任務，例如套用修補基準或重新啟動節點。在維護時段日誌中，可能會回報找不到 Amazon Simple Storage Service (Amazon S3) 儲存貯體或 Amazon DynamoDB 資料表的錯誤。不過，維護時段仍會在資源群組中的節點上執行任務。

9. 選擇 Register target (註冊目標)。

如果您想要將多個目標指派到這個維護時段，選擇 Targets (目標) 標籤，然後選擇 Register target (註冊目標)。使用此選項，您可以選擇不同設定目標的方法。例如，如果您之前透過節點 ID 來將節點作為

目標，您可以透過指定受管節點中套用的標籤，或從資源群組中選擇資源類型來註冊新目標並將節點作為目標。

使用主控台將任務指派至維護時段

在此程序中，您會將任務新增到維護時段。任務是在維護時段執行時所執行的動作。

您可將下列四種類型的任務新增到維護時段：

- AWS Systems Manager Run Command 命令
- Systems Manager Automation 工作流程
- AWS Step Functions 任務
- AWS Lambda 函數

> **Important**
>
> 適用於 Maintenance Windows 的 IAM 政策需要您為 Lambda 函數 (或別名) 名稱新增 `SSM` 字首。在您繼續註冊此類型的任務之前，請在 中更新其名稱 AWS Lambda 以包含 `SSM`。例如，如果 Lambda 函數名稱為 `MyLambdaFunction`，請變更為 `SSMMyLambdaFunction`。

指派任務至維護時段

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Maintenance Windows。
3. 在維護時段清單中，選擇維護時段。
4. 選擇 Actions (動作)，然後選擇您想要向維護時段註冊的任務類型選項：
   - Register Run command task (註冊執行命令任務)
   - Register Automation task (註冊自動化任務)
   - Register Lambda task (註冊 Lambda 任務)
   - Register Step Functions task (註冊步驟函數任務)

> **Note**
>
> 維護時段任務僅支援 Step Functions 標準狀態機器工作流程。這些任務不支援快速狀態機器工作流程。如需有關狀態機器工作流程類型的資訊，請參閱《AWS Step Functions 開發人員指南》中的標準與快速工作流程。

5. (選用) 對於 Name (名稱)，請輸入任務的名稱。
6. 在描述，請輸入描述。
7. 對於 New task invocation cutoff (新任務叫用截止)，如果您不想在到達維護時段截止時間後啟動任何新的任務叫用，則請選擇 Enabled (已啟用)。

   當此選項未啟用時，任務會在到達截止時間後繼續執行，並啟動新的任務叫用，直到完成為止。

   > **Note**
   >
   > 當您啟用此選項時，未完成的任務狀態為 `TIMED_OUT`。

8. 在此步驟中，遵循所選任務類型的子步驟。

   Run Command

   1. 在 Command 文件清單中，選擇定義要執行任務的 Systems Manager Command 文件 (SSM 文件)。
   2. 在 Document Version (文件版本) 中，選擇要使用的文件版本。
   3. 對於 Task priority (任務優先順序)，請指定此任務的優先順序。零 (0) 是最高的優先順序。維護時段內的任務都是以優先順序來排程；相同優先順序的任務會平行排程。

   Automation

   1. 在 Automation 文件清單中，選擇定義要執行任務的 Automation 執行手冊。
   2. 在 Document Version (文件版本) 中，選擇要使用的 Runbook 版本。
   3. 對於 Task priority (任務優先順序)，請指定此任務的優先順序。零 (0) 是最高的優先順序。維護時段內的任務都是以優先順序來排程；相同優先順序的任務會平行排程。

Lambda

1. 在 Lambda 參數區域中，從清單中選擇 Lambda 函數。
2. (選用) 提供您想包含之 Payload (承載)、Client Context (用戶端內容) 或 Qualifier (限定詞) 的任何內容。

> **Note**
>
> 在某些情況下，您可以使用虛擬參數作為 `Payload` 值的一部分。維護時段任務執行時，其會傳遞正確的值 (而不是虛擬參數預留位置)。如需相關資訊，請參閱 註冊維護時段任務時使用虛擬參數。

3. 對於 Task priority (任務優先順序)，請指定此任務的優先順序。零 (0) 是最高的優先順序。維護時段內的任務都是以優先順序來排程；相同優先順序的任務會平行排程。

Step Functions

1. 在 Step Functions 參數區域中，從清單中選擇狀態機器。
2. (選用) 提供狀態機器執行的名稱以及您想包含之 Input (輸入) 的任何內容。

> **Note**
>
> 在某些情況下，您可以使用虛擬參數作為 `Input` 值的一部分。維護時段任務執行時，其會傳遞正確的值 (而不是虛擬參數預留位置)。如需相關資訊，請參閱 註冊維護時段任務時使用虛擬參數。

3. 對於 Task priority (任務優先順序)，請指定此任務的優先順序。零 (0) 是最高的優先順序。維護時段內的任務都是以優先順序來排程；相同優先順序的任務會平行排程。

9. 在 Targets (目標) 區域中，選擇以下其中一項：

- Selecting registered target groups (選取已註冊的目標群組)：選取您已在目前維護時段註冊的一或多個維護時段目標。
- Selecting unregistered targets (選取未註冊的目標)：逐一選擇可用的資源作為任務的目標。

  如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

- Task target not required (不需要任務目標)：任務的目標可能已經針對除 Run Command 類型以外的所有任務在其他函數中指定。

  為維護時段 Run Command 類型任務指定一或多個目標。視任務而定，其他維護時段任務類型（ 自動化 AWS Lambda和 AWS Step Functions) 的目標為選用。如需有關執行未指定目標之任務的詳細資訊，請參閱 註冊不含目標的維護時段任務。

  > **Note**
  >
  > 在許多情況下，您不需要明確指定自動化任務的目標。例如，假設您正在建立 Automation 類型任務來使用 `AWS-UpdateLinuxAmi` Runbook 更新 Linux 的 Amazon Machine Image (AMI)。當任務執行時，AMI 已更新為可用的最新版本 Linux 發行版本套件和 Amazon 軟體。從 AMI 建立的新執行個體已經安裝這些更新。因為在 Runbook 的輸入參數中指定了要更新的 AMI ID，所以不需要在維護時段任務中再次指定目標。

10. 僅限 Automation 任務：

    在 Input parameters (輸入參數) 區域中，為執行任務所需的任何必要或選用參數提供值。

    > **Note**
    >
    > 在某些情況下，您可以針對特定輸入參數值使用虛擬參數。維護時段任務執行時，其會傳遞正確的值 (而不是虛擬參數預留位置)。如需相關資訊，請參閱 註冊維護時段任務時使用虛擬參數。

11. 對於速率控制：

    - 在並行中，指定可同時執行命令的受管節點數目或百分比。

      > **Note**
      >
      > 如果透過指定套用至受管節點的標籤或指定 AWS 資源群組選取了目標，且您不確定會以多少個受管節點為目標，則透過指定百分比限制可以同時執行文件之目標的數量。

    - 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比之節點上的命令失敗之後，停止在其他受管節點上執行命令。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止傳送命令。仍在處理命令的受管節點也可能會傳送錯誤。

12. (選用) 對於 IAM 服務角色，請選擇一個角色，以便為 Systems Manager 提供執行維護時段任務時承擔的許可。

    如果您未指定服務角色 ARN，則 Systems Manager 會使用帳戶中的服務連結角色。此角色未列在下拉式功能表中。如果帳戶中不存在適當的 Systems Manager 服務連結角色，則會在成功註冊任務時建立該角色。

    > **Note**
    >
    > 為了改善安全狀態，強烈建議您建立自訂政策和自訂服務角色，以便執行維護時段任務。您可以制定政策，僅提供特定維護時段任務所需的許可。如需詳細資訊，請參閱設定 Maintenance Windows。

13. 僅限 Run Command 任務：

    (選用) 對於 Output options (輸出選項)，執行下列動作：

    - 選取 Enable writing to S3 (啟用寫入 S3) 核取方塊，將命令輸出儲存成檔案。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。
    - 選取 CloudWatch output (CloudWatch 輸出) 核取方塊，將完整的輸出寫入 Amazon CloudWatch Logs。輸入 CloudWatch Logs 日誌群組的名稱。

    > **Note**
    >
    > 授予能力以將資料寫入至 S3 儲存貯體或 CloudWatch Logs 的許可，是指派給節點的執行個體設定檔的許可 (而不是執行此任務的 IAM 使用者的許可)。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可。此外，如果指定的 S3 儲存貯體或日誌群組位於不同的 中 AWS 帳戶，請確認與節點相關聯的執行個體描述檔具有寫入該儲存貯體的必要許可。

14. 僅限 Run Command 任務：

    在SNS 通知區段中，如果您要傳送有關命令執行狀態的通知，請選取啟用 SNS 通知核取方塊。

    如需為 Run Command 設定 Amazon SNS 通知的詳細資訊，請參閱 使用 Amazon SNS 通知監控 Systems Manager 狀態變更。

15. 僅限 Run Command 任務：

    在 Parameters (參數) 區域中，指定文件的參數。

> **Note**
>
> 在某些情況下，您可以針對特定輸入參數值使用虛擬參數。維護時段任務執行時，其會傳遞正確的值 (而不是虛擬參數預留位置)。如需相關資訊，請參閱 註冊維護時段任務時使用虛擬參數。

16. 僅限 Run Command 和 Automation 任務：

    (選用) 在 CloudWatch 警示區域中，在警示名稱欄位中選擇要套用至任務來進行監控的 CloudWatch 警示。

    如果警示啟用，則會停止任務。

> **Note**
>
> 若要將 CloudWatch 警示附加至您的任務，執行任務的 IAM 主體必須具備 `iam:createServiceLinkedRole` 動作的許可。如需有關 CloudWatch 警示的詳細資訊，請參閱使用 Amazon CloudWatch 警示。

17. 根據任務類型，選擇下列其中一項：

    - Register Run command task (註冊執行命令任務)
    - Register Automation task (註冊自動化任務)
    - Register Lambda task (註冊 Lambda 任務)
    - Register Step Functions task (註冊步驟函數任務)

使用主控台停用或啟用維護時段

您可以在 中停用或啟用維護時段Maintenance Windows，這是 中的工具 AWS Systems Manager。您可以一次選擇一個維護時段，以停用或啟用維護時段。您也可以選取多個或所有維護時段以全部啟用或停用。

本節說明如何使用 Systems Manager 主控台來啟用或停用維護時段。如需如何使用 AWS Command Line Interface (AWS CLI) 執行此操作的範例，請參閱 教學課程：使用 更新維護時段 AWS CLI。

主題

- 使用主控台停用維護時段

- 使用主控台啟用維護時段

使用主控台停用維護時段

您可以透過停用維護時段暫停一項任務一段指定的期間，而且您在之後仍然可以再次啟用此維護時段。

停用維護時段

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Maintenance Windows。
3. 使用您要停用之維護時段旁邊的核取方塊；您可以為一或多個維護時段這麼做。
4. 在動作選單上，選擇停用維護時段。系統會提示您確認您的動作。

使用主控台啟用維護時段

您可以透過啟用維護時段來繼續一項任務。

> **Note**
>
> 如果維護時段使用速率排程，且開始日期目前設定為過去的日期和時間，則會使用目前的日期和時間作為維護時段的開始日期。您可以在啟用維護時段前後變更其開始日期。如需相關資訊，請參閱 使用主控台更新或刪除維護時段資源。

啟用維護時段

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Maintenance Windows。
3. 選取要啟用的維護時段旁的核取方塊。
4. 選擇動作，啟用維護時段。系統會提示您確認您的動作。

使用主控台更新或刪除維護時段資源

您可以在 中更新或刪除維護時段Maintenance Windows，這是 中的工具 AWS Systems Manager。您也可以更新或刪除維護時段的目標或任務。如果您編輯維護時段的詳細資訊，即可變更排程、目標和任

務。您也可以指定時段的名稱和說明、目標和任務，這可協助您更了解他們的用途，並可讓您更輕鬆地管理您佇列的時段。

本節說明如何使用 Systems Manager 主控台來更新或刪除維護時段、目標和任務。如需有關如何使用 AWS Command Line Interface (AWS CLI) 完成此操作的詳細資訊，請參閱 教學課程：使用 更新維護時段 AWS CLI。

主題

- 使用主控台更新或刪除維護時段
- 使用主控台更新或取消註冊維護時段目標
- 使用主控台更新或取消註冊維護時段任務

使用主控台更新或刪除維護時段

您可以更新維護時段來變更其名稱、描述和排程，以及維護時段是否應允許未註冊的目標。

更新或刪除維護時段

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Maintenance Windows。
3. 選取您要更新或刪除之維護時段旁邊的按鈕，然後執行以下其中一項：
   - 選擇 刪除 。系統會提示您確認您的動作。
   - 選擇編輯。在 Edit maintenance window (編輯維護時段) 頁面上，變更您想更改的值和選項，然後選擇 Save changes (儲存變更)。

     如需您可以選擇的組態的相關資訊，請參閱 使用主控台建立維護時段。

使用主控台更新或取消註冊維護時段目標

您可以更新或取消註冊維護時段的目標。如果您選擇更新維護時段目標，您可以指定新的目標名稱、說明和擁有者。您也可以選擇不同的目標。

更新或刪除維護時段的目標

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。

2. 在導覽窗格中，選擇 Maintenance Windows。
3. 選擇要更新之維護時段的名稱、選擇 Targets (目標) 索引標籤，然後執行以下其中一項：
   - 若要更新目標，請選取要更新目標旁邊的按鈕，然後選擇 Edit (編輯)。
   - 若要取消註冊目標，請選取要取消註冊目標旁邊的按鈕，然後選擇 Deregister targets (取消註冊目標)。在 Deregister maintenance windows target (取消註冊維護時段目標) 對話方塊中，選擇 Deregister (取消註冊)。

使用主控台更新或取消註冊維護時段任務

您可以更新或取消註冊維護時段的任務。如果您選擇更新，您可以指定新的任務名稱、說明和擁有者。對於 Run Command 和 Automation 任務，您可以為任務選擇不同的 SSM 文件。不過您無法編輯任務，以變更其類型。例如，如果您建立自動化任務，您不能編輯該任務，並將其變更為 Run Command 任務。

使用主控台更新或刪除維護時段的任務

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Maintenance Windows。
3. 選擇您要更新的維護時段名稱。
4. 選擇 Tasks (任務) 索引標籤，然後選取要更新任務旁邊的按鈕。
5. 執行以下任意一項：
   - 若要取消註冊任務，請選擇 Deregister task (取消註冊任務)。
   - 若要編輯任務，請選擇 Edit (編輯)。變更您想更改的值和選項，然後選擇 Edit task (編輯任務)。

教學課程

本節中的教學課程說明如何在使用維護時段時執行常見任務。

完成事前準備

嘗試這些教學之前，請完成以下事前準備。

- 在本機機器 AWS CLI 上設定 ：您必須先在本機機器上安裝和設定 CLI，才能執行 AWS CLI 命令。如需相關資訊，請參閱安裝或更新 AWS CLI的最新版本和安裝 AWS Tools for PowerShell。

- 驗證維護時段角色和許可 – 您帳戶中的 AWS 管理員必須授予您使用 CLI 管理維護時段所需的 AWS Identity and Access Management (IAM) 許可。如需相關資訊，請參閱 設定 Maintenance Windows。
- 建立或設定與 Systems Manager 相容的執行個體 – 要完成教學課程，您需要至少一個 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體，該執行個體要設定為可與 Systems Manager 搭配使用。這表示 SSM Agent 會安裝在此執行個體，且 Systems Manager 的 IAM 執行個體設定檔會連接到此執行個體。

  我們建議您從一個 AWS 受管 Amazon Machine Image(AMI) 啟動執行個體，其中已預先安裝代理程式。如需詳細資訊，請參閱 尋找預先安裝了 SSM Agent的 AMIs。

  如需在執行個體安裝 SSM Agent 的詳細資訊，請參閱下列主題：

  - 在 Windows Server EC2 執行個體上手動安裝和解除安裝 SSM Agent
  - 在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent

  如需有關為執行個體設定 Systems Manager IAM 許可的詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可。
- 視需要建立其他資源 – Run Command是 Systems Manager 中的工具，包含許多不需要您建立此先決條件主題中所列資源以外的資源的任務。因此，我們提供簡單的 Run Command 任務，供您在第一次演練教學過程中使用。如本主題之前所述，您還需要已設定為可與 Systems Manager 搭配使用的 EC2 執行個體。在設定該執行個體後，您可以註冊簡單的 Run Command 任務。

  Systems Manager Maintenance Windows工具支援執行下列四種類型的任務：

  - Run Command 命令
  - Systems Manager Automation 工作流程
  - AWS Lambda 函數
  - AWS Step Functions 任務

  一般來說，如果您想要執行的維護時段任務需要其他資源，則應先建立這些資源。例如，如果您想要執行 AWS Lambda 函數的維護時段，請在開始之前建立 Lambda 函數；對於Run Command任務，請建立可儲存命令輸出的 S3 儲存貯體 （如果您打算這樣做），以此類推。

教學課程

- 教學課程：使用 建立和管理維護時段 AWS CLI
- 教學課程：使用主控台建立修補維護時段

教學課程：使用 建立和管理維護時段 AWS CLI

這個區段包括的教學，可協助您了解如何使用 AWS Command Line Interface (AWS CLI) 來執行下列動作：

- 建立和設定維護時段
- 檢視維護時段的資訊
- 檢視維護時段任務和任務執行的相關資訊
- 更新維護時段
- 刪除維護時段

追蹤資源 ID

當您完成這些 AWS CLI 教學課程中的任務時，請追蹤您執行的命令所產生的資源 IDs。您可以使用這些 ID 做為後續命令的輸入。例如，在建立維護時段時，系統會以下列格式將維護時段的 ID 提供給您：

```
{
   "WindowId":"mw-0c50858d01EXAMPLE"
}
```

請記下以下系統產生的 ID，因為此區段的教學會用到這些資訊：

- `WindowId`
- `WindowTargetId`
- `WindowTaskId`
- `WindowExecutionId`
- `TaskExecutionId`
- `InvocationId`
- `ExecutionId`

您也需要計劃在此教學課程中使用之 EC2 執行個體的 ID。例如：`i-02573cafcfEXAMPLE`

教學課程

- 教學課程：使用 建立和設定維護時段 AWS CLI

- 教學課程：使用 檢視維護時段的相關資訊 AWS CLI
- 教學課程：使用 檢視任務和任務執行的相關資訊 AWS CLI
- 教學課程：使用 更新維護時段 AWS CLI
- 教學課程：使用 刪除維護時段 AWS CLI

教學課程：使用 建立和設定維護時段 AWS CLI

本教學課程示範如何使用 AWS Command Line Interface (AWS CLI) 來建立和設定維護時段、其目標及其任務。此教學的主要過程包含幾個簡單的步驟。建立一個維護時段、識別單一目標，並為要執行的維護時段設定簡單的任務。我們會在過程中提供您可用來嘗試更複雜案例的資訊。

當您按照此教學課程中的步驟，使用自己的選項和 ID 來取代斜體*##*文字。例如，使用您所建立之資源 ID 取代維護時段 ID *mw-0c50858d01EXAMPLE* 和執行個體 ID *i-02573cafcfEXAMPLE*。

目錄

- 步驟 1：使用 建立維護時段 AWS CLI
- 步驟 2：使用 向維護時段註冊目標節點 AWS CLI
- 步驟 3：使用 向維護時段註冊任務 AWS CLI

步驟 1：使用 建立維護時段 AWS CLI

在此步驟中，您會建立維護時段並指定其基本選項，例如名稱、排程和持續時間。在後續步驟中，您可以選擇其更新的執行個體和其執行的任務。

在我們的範例中，您將建立每 5 分鐘執行一次的維護時段。一般而言，您無法如此頻繁地執行維護時段。不過，這個速率可讓您快速取得教學結果。我們也將示範如何在任務已成功執行之後變更為較低的頻率速率。

> **Note**
>
> 如需維護時段的各種排程相關選項彼此之間有何關聯的說明，請參閱 維護時段排程與作用期間選項。
> 如需使用 `--schedule` 選項的詳細資訊，請參閱參考：Systems Manager 的 Cron 和 Rate 運算式。

使用 建立維護時段 AWS CLI

1. 開啟 AWS Command Line Interface (AWS CLI) 並在本機電腦上執行下列命令，以建立執行下列動作的維護時段：

   - 每 5 分鐘執行一次，持續時間長達兩個小時 (視需要)。
   - 避免讓新任務在維護時段操作結束的 1 小時內啟動。
   - 允許沒有關聯的目標可讓您尚未註冊 (您未向維護時段註冊的執行個體)。
   - 自訂標籤的使用，表示其建立者想要在教學中使用它。

   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"
   ```

   系統會傳回與以下相似的資訊。

   ```
   {
      "WindowId":"mw-0c50858d01EXAMPLE"
   }
   ```

2. 立即執行以下命令來檢視相關詳細資訊，以及您帳戶中已存在的任何其他維護時段。

```
aws ssm describe-maintenance-windows
```

系統會傳回與以下相似的資訊。

```
{
   "WindowIdentities":[
      {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Name": "My-First-Maintenance-Window",
            "Enabled": true,
            "Duration": 2,
            "Cutoff": 1,
            "NextExecutionTime": "2019-05-11T16:46:16.991Z"
      }
   ]
}
```

繼續進行步驟 2：使用 向維護時段註冊目標節點 AWS CLI。

步驟 2：使用 向維護時段註冊目標節點 AWS CLI

在此步驟中，您會使用新的維護時段來註冊目標。在這個情況下，您會指定維護時段執行時要更新哪個節點。

如需使用節點 ID 一次註冊多個節點的範例、使用標籤來識別多個節點的範例，與將資源群組指定為目標的範例，請參閱 範例：向維護時段註冊目標。

> **Note**
>
> 您應已建立此步驟中要使用的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體，如 Maintenance Windows 教學課程事前準備所述。

使用 AWS CLI向維護時段註冊目標節點

1. 在本機機器上執行以下命令。將每個########取代為您自己的資訊。

   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"
```

系統會傳回與以下相似的資訊。

```
{
   "WindowTargetId":"e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
}
```

2. 立即在您的本機機器上執行以下命令，來檢視有關維護時段目標的詳細資訊。

Linux & macOS

```
aws ssm describe-maintenance-window-targets \
    --window-id "mw-0c50858d01EXAMPLE"
```

Windows

```
aws ssm describe-maintenance-window-targets ^
    --window-id "mw-0c50858d01EXAMPLE"
```

系統會傳回與以下相似的資訊。

```
{
    "Targets": [
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "WindowTargetId": "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE",
            "ResourceType": "INSTANCE",
            "Targets": [
                {
```

```
                    "Key": "InstanceIds",
                    "Values": [
                        "i-02573cafcfEXAMPLE"
                    ]
                }
            ]
        }
    ]
}
```

繼續進行步驟 3：使用 向維護時段註冊任務 AWS CLI。

範例：向維護時段註冊目標

您可以使用其節點 ID 將單一節點註冊為目標，如 步驟 2：使用 向維護時段註冊目標節點 AWS CLI 中所示範。您也可以使用此頁面上的命令格式來將一或多個節點註冊為目標。

一般而言，有兩種方法可以識別您想要做為維護時段目標的節點：指定個別的節點，並使用資源標籤。資源標籤方法提供多個選項，如範例 2-3 所示。

您也可以將一或多個資源群組指定為維護時段的目標。資源群組可以包含節點和許多其他類型的支援 AWS 資源。接下來的範例 4 和 5 會示範如何將資源群組新增到維護時段目標。

> **Note**
>
> 如果單一維護時段任務已向多個目標註冊，則其任務叫用會依序發生，而非平行發生。如果您的任務必須同時在多個目標上執行，請個別註冊每個目標的任務，並為每個任務指派相同的優先順序層級。

如需建立和管理資源群組的詳細資訊，請參閱《AWS Resource Groups 使用者指南》中的什麼是資源群組？和 AWS 新聞部落格中的 AWS的資源群組和標記。

如需 中Maintenance Windows工具 的配額資訊 AWS Systems Manager，除了下列範例中指定的配額外，請參閱 中的 Systems Manager 服務配額Amazon Web Services 一般參考。

範例 1：使用節點 ID 註冊多個目標

在本機機器上執行下列命令，以使用其節點 ID 將多個節點註冊為目標。將每個########取代為您自己的資訊。

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
```

建議使用：在第一次使用任何維護時段註冊唯一一組的節點時非常有用，但它們「不會」共用常見的節點標籤。

配額：您可以為每個維護時段目標指定總計最多 50 個節點。

範例 2：使用節點中套用的資源標籤來註冊目標

在本機機器上執行下列命令來註冊節點，這些執行個體皆已包含您已指派之索引鍵值對的標籤。將每個########取代為您自己的資訊。

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"
```

建議使用：在第一次使用任何維護時段註冊唯一一組的節點時非常有用，但它們「會」共用常見的節點標籤。

配額：您可以為每個目標指定總計最多五個鍵值組。如果您指定了多個鍵值對，則節點必須加上您指定的所有標籤鍵和值，才會包含在目標群組中。

> **Note**
>
> 您可以使用標籤金鑰 `Patch Group` 或 `PatchGroup` 為一組節點加上標籤，並將共用的金鑰值 (例如 `my-patch-group`) 指派給節點。（如果您在 EC2 執行個體中繼資料中允許標籤，則必須使用 `PatchGroup`，而不要使用空格。) Systems Manager 中的Patch Manager工具 會評估節點上的 `Patch Group`或 `PatchGroup`金鑰，以協助判斷適用於它們的修補程式基準。如果您的任務會執行 `AWS-RunPatchBaseline` SSM 文件 (或舊版 `AWS-ApplyPatchBaseline` SSM 文件)，則您可以指定向維護時段註冊目標時的相同 `Patch Group` 或 `PatchGroup` 金鑰/值對。例如：`--target "Key=tag:PatchGroup,Values=my-patch-group`。這樣可允許您使用維護時段為一組節點更新修補程式 (已與相同修補基準建立關聯)。如需詳細資訊，請參閱修補程式群組。

範例 3：使用一組標籤索引鍵來註冊目標 (不含標籤值)

在本機機器上執行以下命令來註冊執行個體，這些節點皆已獲指派一或多個標籤索引鍵，無論其索引鍵值為何。將每個*########*取代為您自己的資訊。

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"
```

建議使用：當您想要透過指定多個標籤索引鍵 (不含其值)，而不只是一個標籤索引鍵或標籤索引鍵值對，來鎖定節點時很有用。

配額：您可以為每個目標指定總計最多五個標籤鍵。如果您指定多個標籤鍵，則節點必須加上您指定的所有標籤鍵，才會包含在目標群組中。

範例 4：使用資源群組名稱註冊目標

在本機機器上執行以下命令來註冊指定的資源群組，無論其中包含的資源類型為何。將 *mw-0c50858d01EXAMPLE* 取代為您自己的資訊。如果您指派給維護時段的任務沒有對在此資源群組中包含的資源類型執行動作，系統可能會報告錯誤。儘管發生這些錯誤，找到支援資源類型的任務會持續執行。

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"
```

建議使用：當您想要快速將資源群組指定為目標，而不評估維護時段是否將其所有資源類型視為目標時，或當您知道資源群組僅包含您的任務對其執行動作的資源類型時很有用。

配額：您可以僅將一個資源群組指定為目標。

範例 5：在資源群組中篩選資源類型來註冊目標

在本機機器上執行以下命令來僅註冊特定資源類型，這些資源類型屬於您指定的資源群組。將 *mw-0c50858d01EXAMPLE* 取代為您自己的資訊。在使用此選項的情況下，即使您為屬於資源群組的資源類型新增任務，如果您還沒有將資源類型明確新增到篩選條件，此任務就不會執行。

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"
```

建議用途：當您想要嚴格控制維護時段可以執行動作 AWS 的資源類型，或當資源群組包含大量資源類型，且您想要避免維護時段日誌中不必要的錯誤報告時，請使用此功能。

配額：您可以僅將一個資源群組指定為目標。

步驟 3：使用 向維護時段註冊任務 AWS CLI

在教學課程的此步驟中，您會註冊在 AWS Systems Manager Run Command Linux 的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上執行 df命令的任務。此標準 Linux 命令的結果會顯示有多少可用空間，以及執行個體磁碟檔案系統上使用多少空間。

-或-

如果您以 Windows Server 的 Amazon EC2 執行個體為目標 (而不是 Linux)，請在下列命令中以 ipconfig 取代 df。此命令中的輸出會列出在目標執行個體上適用於轉接器的 IP 地址、子網路遮罩以及預設閘道的詳細資訊。

當您準備好註冊其他任務類型，或使用更多可用 Systems Manager Run Command 選項時，請參閱 範例：向維護時段註冊任務。目前，我們提供所有四個任務類型的詳細資訊，以及其中一些最重要的選項，以協助您規劃更廣泛的真實世界案例。

向維護時段註冊任務

1. 在本機機器上執行以下命令。將每個########取代為您自己的資訊。從本機 Windows 機器執行的版本包含逸出字元 (「/」)，您在透過命令列工具執行命令時會需要這些字元。

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\"]}}}
```

系統會傳回與以下相似的資訊：

```
{
    "WindowTaskId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE"
}
```

2. 立即執行以下命令來檢視與您建立之維護時段任務相關的詳細資訊。

Linux & macOS

```
aws ssm describe-maintenance-window-tasks \
    --window-id mw-0c50858d01EXAMPLE
```

Windows

```
aws ssm describe-maintenance-window-tasks ^
```

```
    --window-id mw-0c50858d01EXAMPLE
```

3. 系統會傳回與以下相似的資訊。

```
{
    "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. 根據您在 步驟 1：使用 建立維護時段 AWS CLI 指定的排程，等到任務的執行時間。例如，如果您已指定 **`--schedule "rate(5 minutes)"`**，請等待五分鐘。然後執行以下命令，來檢視與此任務發生的任何執行所相關的資訊。

   Linux & macOS

```
aws ssm describe-maintenance-window-executions \
    --window-id mw-0c50858d01EXAMPLE
```

   Windows

```
aws ssm describe-maintenance-window-executions ^
    --window-id mw-0c50858d01EXAMPLE
```

系統會傳回與以下相似的資訊。

```
{
    "WindowExecutions": [
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "WindowExecutionId": "14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE",
            "Status": "SUCCESS",
            "StartTime": 1557593493.096,
            "EndTime": 1557593498.611
        }
    ]
}
```

Tip

任務順利執行後，您可以降低維護時段執行的速率。例如，執行以下命令來將頻率降低為一週一次。將 *mw-0c50858d01EXAMPLE* 取代為您自己的資訊。

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)"
```

如需管理維護時段排程的詳細資訊，請參閱參考：Systems Manager 的 Cron 和 Rate 運算式和維護時段排程與作用期間選項。
如需使用 AWS Command Line Interface (AWS CLI) 修改維護時段的詳細資訊，請參閱 教學課程：使用 更新維護時段 AWS CLI。

如需練習執行 AWS CLI 命令以檢視維護時段任務及其執行的詳細資訊，請繼續 教學課程：使用 檢視任務和任務執行的相關資訊 AWS CLI。

存取教學課程命令輸出

超出本教學課程的範圍，您可以使用 AWS CLI 檢視與維護時段任務執行相關聯的Run Command命令輸出。

不過，您可以使用 檢視此資料 AWS CLI。(您也可以在 Systems Manager 主控台中或在 Amazon Simple Storage Service (Amazon S3) 儲存貯體中存放的日誌檔檢視輸出 (如果您已將維護時段設定為在前述日誌檔中存放命令輸出)。) 您會發現在 Linux EC2 執行個體上的 df 命令輸出與以下內容類似。

```
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% /
```

在 Windows Server EC2 執行個體的 ipconfig 命令輸出與以下內容類似：

```
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  . :
```

範例：向維護時段註冊任務

您可以使用 AWS Command Line Interface (AWS CLI) Run Command在 中註冊任務 AWS Systems Manager，在 中註冊具有維護時段的工具，如使用維護時段註冊任務所示。您也可以註冊 Systems Manager Automation 工作流程、 AWS Lambda 函數和 AWS Step Functions 任務的任務，如本主題稍後所示。

> **Note**
>
> 為維護時段 Run Command 類型任務指定一或多個目標。視任務而定，其他維護時段任務類型（ 自動化 AWS Lambda和 AWS Step Functions) 的目標為選用。如需有關執行未指定目標之任務的詳細資訊，請參閱 註冊不含目標的維護時段任務。

在本主題中，我們提供使用 AWS Command Line Interface (AWS CLI) 命令`register-task-with-maintenance-window`向維護時段註冊四種支援任務類型的每一個範例。此範例僅用於示範，但您可以進行修改來建立可運作的任務註冊命令。

使用 --cli-input-json 選項

為了更有效地管理您的任務選項，您可以使用命令選項 `--cli-input-json`，內含 JSON 檔案中參考的選項值。

若要使用我們在以下範例中提供的範本 JSON 檔案內容，請在您的本機機器上執行下列動作：

1. 建立包含 MyRunCommandTask.json、MyAutomationTask.json 這類名稱或您偏好的另一個名稱來建立檔案。
2. 將我們的 JSON 範本內容複製到此檔案中。
3. 將檔案內容修改為任務註冊適用的內容，然後儲存檔案。
4. 在您存放該檔案的相同目錄中執行下列命令。替換 *MyFile.json* 的檔案名稱。

   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
   ```

維護時段任務中的虛擬參數

在某些範例中，我們使用虛擬參數做為將 ID 資訊傳遞到任務的方法。例如，{{TARGET_ID}} 和 {{RESOURCE_ID}} 可以用來將 AWS 資源的 ID 傳遞給 Automation、Lambda 和 Step Functions 任務。如需 --task-invocation-parameters 內容中虛擬參數的詳細資訊，請參閱註冊維護時段任務時使用虛擬參數。

詳細資訊

- register-task-with-maintenance-windows 命令的參數選項.
- 《AWS CLI 命令參考》中的 register-task-with-maintenance-window 一節
- AWS Systems Manager API 參考中的 RegisterTaskWithMaintenanceWindow

任務註冊範例

以下各節提供範例 AWS CLI 命令，用於註冊支援的任務類型和可與 --cli-input-json選項搭配使用的 JSON 範例。

註冊 Systems Manager Run Command 任務

以下範例示範如何使用 AWS CLI向維護時段註冊 Systems Manager Run Command 任務。

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\"]}}}"
```

要與 `--cli-input-json` 檔案選項搭配使用的 JSON 內容：

```
{
    "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
        }
    }
}
```

註冊 Systems Manager Automation 任務

以下範例示範如何使用 AWS CLI向維護時段註冊 Systems Manager Automation 任務：

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"
```

要與 **--cli-input-json** 檔案選項搭配使用的 JSON 內容：

```
{
    "WindowId": "mw-0c50858d01EXAMPLE",
        "TaskArn": "AWS-PatchInstanceWithRollback",
    "TaskType": "AUTOMATION","TaskInvocationParameters": {
        "Automation": {
            "DocumentVersion": "1",
            "Parameters": {
                "instanceId": [
                    "{{RESOURCE_ID}}"
                ]
            }
        }
    }
}
```

註冊 AWS Lambda 任務

以下範例示範如何使用 AWS CLI向維護時段註冊 Lambda 函數任務。

對於這些範例，建立 Lambda 函數的使用者會將其命名為 `SSMrestart-my-instances` 並建立名為 `instanceId` 和 `targetType` 的兩個參數。

> **Important**
>
> 適用於 Maintenance Windows 的 IAM 政策需要您為 Lambda 函數 (或別名) 名稱新增 `SSM` 字首。在您繼續註冊此類型的任務之前，請在 中更新其名稱 AWS Lambda 以包含 `SSM`。例如，如果 Lambda 函數名稱為 `MyLambdaFunction`，請變更為 `SSMMyLambdaFunction`。

AWS CLI 命令：

Linux & macOS

> **Important**
>
> 如果您使用的是 第 2 版 AWS CLI，且 Lambda 承載不是 base64 編碼，則必須在下列命令`--cli-binary-format raw-in-base64-out`中包含 選項。`cli_binary_format` 選項僅在版本 2 中可用。如需有關此 AWS CLI `config`和其他檔案設定的資訊，請參閱AWS Command Line Interface 《 使用者指南》中的支援config的檔案設定。

```
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 第 2 版 AWS CLI，且 Lambda 承載不是 base64 編碼，則必須在下列命令`--cli-binary-format raw-in-base64-out`中包含 選項。`cli_binary_format` 選項僅在版本 2 中可用。如需有關此 AWS CLI `config`和其他檔案設定的資訊，請參閱AWS Command Line Interface 《 使用者指南》中的支援config的檔案設定。

```
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\"}}'
```

要與 **--cli-input-json** 檔案選項搭配使用的 JSON 內容：

```
{
    "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
}
```

註冊 Step Functions 任務

以下範例示範如何使用 AWS CLI向維護時段註冊 Step Functions 狀態機器任務。

> **Note**
>
> 維護時段任務僅支援 Step Functions 標準狀態機器工作流程。這些任務不支援快速狀態機器工作流程。如需有關狀態機器工作流程類型的資訊，請參閱《AWS Step Functions 開發人員指南》中的標準與快速工作流程。

對於這些範例，建立步驟函數狀態機器的使用者會使用名為 `instanceId` 的參數建立名為 `SSMMyStateMachine` 的狀態機器。

> **Important**
>
> 的 AWS Identity and Access Management (IAM) 政策Maintenance Windows需要您在 Step Functions 狀態機器名稱前面加上 SSM。在繼續註冊這類任務前，您必須在 AWS Step Functions 中將其名稱更新為包含 SSM。例如，如果狀態機器名稱為 MyStateMachine，請變更為 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"
```

要與 `--cli-input-json` 檔案選項搭配使用的 JSON 內容：

```
{
    "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 命令的參數選項

register-task-with-maintenance-window 命令會提供多種選項，可供您根據需求來設定任務。有些是必要的，有些是選用的，有些僅適用於單一維護時段任務類型。

本主題提供部分這些選項的相關資訊，以協助您使用此教學區段中的範本。如需其他命令選項的相關資訊，請參閱《AWS CLI 命令參考》中的 register-task-with-maintenance-window。

命令選項：**`--task-arn`**

選項 `--task-arn` 可用來指定執行任務的資源。您指定的值取決於註冊的任務類型，如下表所述。

維護時段任務的 TaskArn 格式

維護時段任務類型	TaskArn 值
RUN_COMMAND 與 **AUTOMATION**	TaskArn 是 SSM 文件名稱或 Amazon Resource Name (ARN)。例如：

維護時段任務類型	TaskArn 值
	AWS-RunBatchShellScript -或- arn:aws:ssm: *region*:11112222 3333:document/My-Document .
LAMBDA	TaskArn 是函數名稱或 ARN。例如： SSMMy-Lambda-Function -或- arn:aws:lambda: *region*:11112222 3333:function:SSMMyLambdaFu nction . **Important** 適用於 Maintenance Windows 的 IAM 政策需要您為 Lambda 函數 (或別名) 名稱新增 SSM 字首。在您繼續註冊此類型的任務之前，請在 中更新其名稱 AWS Lambda 以包含 SSM。例如，如果 Lambda 函數名稱為 MyLambdaF unction ，請變更為 SSMMyLamb daFunction 。

維護時段任務類型	TaskArn 值
STEP_FUNCTIONS	TaskArn 是狀態機器的 ARN。例如： arn:aws:states:us-east-2:111122223333:stateMachine:SSMMyStateMachine . **Important** 維護時段的 IAM 政策需要您為 Step Functions 狀態機器名稱加上 SSM 字首。註冊這類任務前，您必須在 AWS Step Functions 中將其名稱更新為包含 SSM。例如，如果狀態機器名稱為 MyStateMachine ，請變更為 SSMMyStateMachine 。

命令選項：**--service-role-arn**

執行維護時段任務時 AWS Systems Manager ，要擔任的角色。

如需詳細資訊，請參閱 設定 Maintenance Windows

命令選項：**--task-invocation-parameters**

--task-invocation-parameters 選項可用來指定四種任務類型特有的參數。下表描述四種任務類型支援的參數。

Note

如需在 --task-invocation-parameters 內容中使用虛擬參數的資訊，例如 {{TARGET_ID}}，請參閱 註冊維護時段任務時使用虛擬參數。

維護時段任務的任務叫用參數選項

維護時段任務類型	可用參數	範例
RUN_COMMAND	Comment DocumentHash DocumentHashType NotificationConfig OutputS3BucketName OutPutS3KeyPrefix Parameters ServiceRoleArn TimeoutSeconds	"TaskInvocationPar ameters": { "RunCommand": { "Comment" : "My Run Command task comment", "Document Hash": "6554ed3d-- truncated--5EXAMPLE", "Document HashType": "Sha256", "Notifica tionConfig": { "Notifica tionArn": "arn:aws: sns: *region*:12345678 9012:my-sns-topic- name", "NotificationEvents": ["FAILURE"], "NotificationType": "Invocation" }, "OutputS3 BucketName": "amzn-s3- demo-bucket", "OutputS3 KeyPrefix": " *S3-PREFIX* ", "Paramete rs": { "commands": ["Get-ChildItem$env:

維護時段任務類型	可用參數	範例
		temp-Recurse|Remove-Item-Recurse-force"] }, "ServiceRoleArn": "arn:aws:iam::123456789012:role/MyMaintenanceWindowServiceRole", "TimeoutSeconds": 3600 } }
AUTOMATION	DocumentVersion Parameters	"TaskInvocationParameters": { "Automation": { "DocumentVersion": "3", "Parameters": { "instanceid": ["{{TARGET_ID}}"] } } }

維護時段任務類型	可用參數	範例

LAMBDA	ClientContext
	
	Payload
	
	Qualifier
		<pre>"TaskInvocationParameters": {
		        "Lambda": {
		            "ClientContext": "ew0KICAi--truncated--0KIEXAMPLE",
		            "Payload": "{ \"targetId\": \"{{TARGET_ID}}\", \"targetType\": \"{{TARGET_TYPE}}\" }",
		            "Qualifier": "$LATEST"
		        }
		    }</pre>

STEP_FUNCTIONS	Input
	
	Name
		<pre>"TaskInvocationParameters": {
		        "StepFunctions": {
		            "Input": "{ \"targetId\": \"{{TARGET_ID}}\" }",
		            "Name": "{{INVOCATION_ID}}"
		        }
		    }</pre>

教學課程：使用 檢視維護時段的相關資訊 AWS CLI

此教學中包含的命令可協助您更新或取得維護時段、任務、執行和呼叫的相關資訊。此範例是依命令來示範如何使用命令選項來篩選您要查看的類型的詳細資訊。

當您按照此教學課程中的步驟，使用自己的選項和 ID 來取代斜體*##*文字。例如，使用您所建立之資源 ID 取代維護時段 ID *mw-0c50858d01EXAMPLE* 和執行個體 ID *i-02573cafcfEXAMPLE*。

如需有關設定和設定 AWS Command Line Interface (AWS CLI) 的資訊，請參閱安裝、更新和解除安裝 AWS CLI和設定 AWS CLI。

命令範例

- 'describe-maintenance-windows' 的範例
- 'describe-maintenance-window-targets' 的範例
- 'describe-maintenance-window-tasks' 的範例
- 'describe-maintenance-windows-for-target' 的範例
- 'describe-maintenance-window-executions' 的範例
- 'describe-maintenance-window-schedule' 的範例

'describe-maintenance-windows' 的範例

列出 中的所有維護時段 AWS 帳戶

執行下列命令。

```
aws ssm describe-maintenance-windows
```

系統會傳回與以下相似的資訊。

```
{
   "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"
      },
```

```
   ]
}
```

列出所有啟用的維護時段

執行下列命令。

```
aws ssm describe-maintenance-windows --filters "Key=Enabled,Values=true"
```

系統會傳回與以下相似的資訊。

```
{
   "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"
      },
   ]
}
```

列出所有停用的維護時段

執行下列命令。

```
aws ssm describe-maintenance-windows --filters "Key=Enabled,Values=false"
```

系統會傳回與以下相似的資訊。

```
{
    "WindowIdentities": [
```

```
        {
            "WindowId": "mw-6e5c9d4b7cEXAMPLE",
            "Name": "My-Disabled-Maintenance-Window",
            "Enabled": false,
            "Duration": 2,
            "Cutoff": 1
        }
    ]
}
```

列出所有含特定字首開頭名稱的維護時段

執行下列命令。

```
aws ssm describe-maintenance-windows --filters "Key=Name,Values=My"
```

系統會傳回與以下相似的資訊。

```
{
    "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' 的範例

顯示符合特定擁有者資訊值的維護時段目標

執行下列命令。

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**
>
> 支援的篩選條件索引鍵為 Type、WindowTargetId 和 OwnerInformation。

系統會傳回與以下相似的資訊。

```
{
    "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' 的範例

顯示所有叫用 SSM 命令文件 **AWS-RunPowerShellScript** 的註冊任務

執行下列命令。

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"
```

系統會傳回與以下相似的資訊。

```
{
   "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"
            }
         ]
      }
   ]
}
```

顯示所有具有優先順序「3」的註冊任務

執行下列命令。

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"
```

系統會傳回與以下相似的資訊。

```
{
   "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"
            }
         ]
      }
   ]
}
```

顯示所有具有優先順序 "1"，並使用 Run Command 的註冊任務

執行下列命令。

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"
```

系統會傳回與以下相似的資訊。

```
{
    "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' 的範例

列出維護時段目標或任務 (與特定節點相關) 的資訊

執行下列命令。

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
```

系統會傳回與以下相似的資訊。

```
{
```

```
    "WindowIdentities": [
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Name": "My-First-Maintenance-Window"
        },
        {
            "WindowId": "mw-9a8b7c6d5eEXAMPLE",
            "Name": "My-Second-Maintenance-Window"
        }
    ]
}
```

'describe-maintenance-window-executions' 的範例

列出在特定日期之前執行的所有任務

執行下列命令。

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"
```

系統會傳回與以下相似的資訊。

```
{
    "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
        }
    ]
}
```

列出在特定日期之後執行的所有任務

執行下列命令。

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"
```

系統會傳回與以下相似的資訊。

```
{
    "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' 的範例

顯示後續 10 個排程替特定節點執行的維護時段

執行下列命令。

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
```

系統會傳回與以下相似的資訊。

```
{
    "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"
}
```

顯示替標記了特定金鑰/值對之節點排程的維護時段

執行下列命令。

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"
```

系統會傳回與以下相似的資訊。

```
{
    "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"
}
```

顯示後續四個維護時段的開始時間

執行下列命令。

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"
```

系統會傳回與以下相似的資訊。

```
{
    "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"
                }
            ]
        }
    ]
}
```

教學課程：使用 檢視任務和任務執行的相關資訊 AWS CLI

本教學課程示範如何使用 AWS Command Line Interface (AWS CLI) 來檢視已完成維護時段任務的詳細資訊。

如果您直接從 教學課程：使用 建立和設定維護時段 AWS CLI 繼續執行，請確保維護時段有足夠的時間可執行至少一次，以查看其執行的結果。

當您按照此教學課程中的步驟，使用自己的選項和 ID 來取代斜體*##*文字。例如，使用您所建立之資源 ID 取代維護時段 ID *mw-0c50858d01EXAMPLE* 和執行個體 ID *i-02573cafcfEXAMPLE*。

使用 AWS CLI檢視有關任務和任務執行的資訊

1. 執行以下命令，以檢視特定維護時段的任務執行清單。

   Linux & macOS

   ```
   aws ssm describe-maintenance-window-executions \
   ```

```
    --window-id "mw-0c50858d01EXAMPLE"
```

Windows

```
aws ssm describe-maintenance-window-executions ^
    --window-id "mw-0c50858d01EXAMPLE"
```

系統會傳回與以下相似的資訊。

```
{
    "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. 執行以下命令，以取得有關的維護時段任務執行。

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"
```

系統會傳回與以下相似的資訊。

```
{
    "WindowExecutionId": "14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE",
    "TaskIds": [
        "c9b05aba-197f-4d8d-be34-e73fbEXAMPLE"
    ],
    "Status": "SUCCESS",
    "StartTime": 1557593493.096,
    "EndTime": 1557593498.611
}
```

3. 執行以下命令，以列出維護時段執行期間執行的任務清單。

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"
```

系統會傳回與以下相似的資訊。

```
{
    "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. 執行以下命令，以取得有關執行任務的詳細資訊。

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"
```

系統會傳回與以下相似的資訊。

```
{
    "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. 執行以下命令以獲得執行時特定的呼叫式任務。

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"
```

系統會傳回與以下相似的資訊。

```
{
    "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
        }
    ]
}
```

教學課程：使用 更新維護時段 AWS CLI

本教學課程示範如何使用 AWS Command Line Interface (AWS CLI) 來更新維護時段。它也說明如何更新不同的任務類型，包括 AWS Systems Manager Run Command和 自動化的任務類型 AWS Lambda，以及 AWS Step Functions。

本節中的範例使用以下 Systems Manager 動作來更新維護時段。

- UpdateMaintenanceWindow
- UpdateMaintenanceWindowTarget
- UpdateMaintenanceWindowTask
- DeregisterTargetFromMaintenanceWindow

如需如何使用 Systems Manager 主控台更新維護時段的資訊，請參閱 使用主控台更新或刪除維護時段資源。

當您按照此教學課程中的步驟，使用自己的選項和 ID 來取代斜體*##*文字。例如，使用您所建立之資源 ID 取代維護時段 ID *mw-0c50858d01EXAMPLE* 和執行個體 ID *i-02573cafcfEXAMPLE*。

使用 更新維護時段 AWS CLI

1. 開啟 AWS CLI 並執行下列命令來更新目標，以包含名稱和描述。

   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"
```

系統會傳回與以下相似的資訊。

```
{
    "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. 執行以下命令來使用 `replace` 選項移除描述欄位，並新增額外的目標。移除描述欄位，因為更新後不包括此欄位 (空值)。請確定已指定一個額外的執行個體，此節點已設定為與 Systems Manager 搭配使用。

   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
```

系統會傳回與以下相似的資訊。

```
{
    "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` 允許您延遲啟用維護時段直到指定的未來日期。`end-date` 選項可讓您設定不再執行維護時段的未來日期和時間點。以 ISO-8601 Extended 格式指定選項。

   執行以下命令，來指定定期排定維護時段執行的日期和時間範圍。

   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. 執行下列命令以更新 Run Command 任務

> **Tip**
>
> 如果您的目標是適用於 Windows Server 的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體，則將下列命令中的 `df` 變更為 `ipconfig`，以及 `AWS-RunShellScript`變更為 `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"
```

系統會傳回與以下相似的資訊。

```
{
    "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. 調整並執行下列命令以更新 Lambda 任務。

   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"
```

系統會傳回與以下相似的資訊。

```
{
    "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. 如果您更新的是 Step Functions 任務，請調整和執行以下命令來更新其 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"
```

   系統會傳回與以下相似的資訊。

```
{
    "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. 執行下列命令替維護時段取消註冊任務。此範例使用 `safe` 參數來判斷任次參考此目標，因此能夠安全的取消註冊。

   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
   ```

   系統會傳回與以下相似的資訊。

```
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. 執行以下命令替維護時段取消註冊目標，即使任務參考此目標。您可以使用 no-safe 強制取消註冊操作。

   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
   ```

   系統會傳回與以下相似的資訊。

   ```
   {
       "WindowId": "mw-0c50858d01EXAMPLE",
       "WindowTargetId": "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
   }
   ```

9. 執行下列命令以更新 Run Command 任務 這個範例使用名為 UpdateLevel 的 Systems Manager Parameter Store 參數，格式如下：'{{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}}'}}"
```

系統會傳回與以下相似的資訊。

```
{
    "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. 執行以下命令更新 Automation 任務來替 `task-invocation-parameters` 參數指定 `WINDOW_ID` 和 `WINDOW_TASK_ID` 參數：

    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
    ```

    系統會傳回與以下相似的資訊。

    ```
    {
        "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"
}
```

教學課程：使用 刪除維護時段 AWS CLI

若要刪除您在這些教學中所建立的維護時段，請執行下列命令。

```
aws ssm delete-maintenance-window --window-id "mw-0c50858d01EXAMPLE"
```

系統會傳回與以下相似的資訊。

```
{
   "WindowId":"mw-0c50858d01EXAMPLE"
}
```

教學課程：使用主控台建立修補維護時段

> **Important**
>
> 您可以繼續使用這個舊版主題來建立維護時段，以進行修補。不過，建議您改用修補程式政策。如需詳細資訊，請參閱 Quick Setup中的修補程式政策組態 和 使用修補程式政策設定組織中執行個體的Quick Setup修補。

為了盡可能降低對伺服器可用性的影響，建議您設定維護時段在不會插斷您商業運作的期間執行修補。

在開始此程序之前 AWS Systems Manager，您必須為 中的工具 Maintenance Windows設定角色和許可。如需詳細資訊，請參閱設定 Maintenance Windows。

建立維護時段以進行修補

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Maintenance Windows。
3. 選擇建立維護時段。
4. 針對 Name (名稱)，輸入名稱，並將其指定為維護時段以修補關鍵與重要的更新。
5. 在描述，請輸入描述。
6. 如果您想允許維護時段任務在受管節點上執行 (即使您尚未將這些節點註冊為目標)，請選擇 Allow unregistered targets (允許未註冊目標)。

   如果您選擇此選項，即可在向維護時段註冊任務時選擇未註冊的節點 (依據節點 ID)。

   如果您未選擇此選項，則必須在向維護時段註冊任務時選擇先前註冊過的目標。
7. 在 Schedule (排程) 區段頂端，使用三個排程選項之一來指定維護時段的排程。

   如需有關建立 Cron/Rate 運算式的詳細資訊，請參閱參考：Systems Manager 的 Cron 和 Rate 運算式。
8. 針對 Duration (持續時間)，輸入維護時段將執行的時數。您指定的值會根據維護時段的開始時間，決定維護時段的特定結束時間。在產生的結束時間減去您在下一個步驟中為 Stop initiating tasks (停止啟動任務) 指定的小時數過後，將不允許啟動任何維護時段任務。

   例如，如果維護時段從下午 3 點開始，持續時間為三小時，而 Stop initiating tasks (停止啟動任務) 的值為一小時，則在下午 5 點之後無法啟動任何維護時段任務。

9. 針對 Stop initiating tasks (停止初始任務)，輸入在維護時段執行結束之前，系統應該停止排程新任務的時數。
10. (選用) 對於 Window start date (時段開始日期)，依照 ISO-8601 Extended 格式，指定您希望開始啟用維護時段的日期和時間。這可讓您延遲啟用維護時段，直到指定的未來日期為止。
11. (選用) 對於 Window end date (時段結束日期)，依照 ISO-8601 Extended 格式，指定您希望停用維護時段的日期和時間。這可讓您設定不再執行維護時段的未來日期和時間點。
12. (選用) 對於排程時區，以網際網路號碼分配局 (IANA) 格式，指定排程的維護時段執行所依據的時區。例如："America/Los_Angeles"、"etc/UTC" 或 "Asia/Seoul"。

    如需有關有效格式的詳細資訊，請參閱 IANA 網站上的時區資料庫有效格式。
13. (選用) 在 Manage tags (管理標籤) 區域，將一或多個標籤金鑰名稱/值對套用到維護時段。

    標籤是您指派給資源的選用性中繼資料。標籤允許您以不同的方式 (例如用途、擁有者或環境) 將資源分類。例如，您可能想要標記此維護時段，以識別其執行的任務類型。在這種情況下，您可以指定以下索引鍵名稱/值對：

    - `Key=TaskType,Value=Patching`
14. 選擇建立維護時段。
15. 在維護時段清單中，選擇您剛建立的維護時段，然後選擇 Actions (動作)、Register targets (註冊目標)。
16. (選用) 在 Maintenance window target details (維護時段目標詳細資訊) 部分，提供此目標的名稱、描述及擁有者資訊 (您的名稱或別名)。
17. 對於目標選擇，選擇指定執行個體標籤。
18. 對於指定執行個體標籤，輸入標籤鍵和標籤值，以識別要向維護時段註冊的節點，然後選擇新增。
19. 選擇 Register target (註冊目標)。系統會建立一個維護時段目標。
20. 在您建立的維護時段的詳細資訊頁面中，選擇 Actions (動作)、Register run command task (註冊執行命令任務)。
21. (選用) 在 Maintenance window task details (維護時段任務詳細資訊) 中提供此任務的名稱與描述。
22. 如需 Command document (命令文件)，請選擇 `AWS-RunPatchBaseline`。
23. 在 Task priority (任務優先順序) 中選擇優先順序。零 (0) 是最高的優先順序。
24. 針對 Targets (目標)，請在 Target by (目標依據) 下，選擇您之前在此程序建立的維護時段目標。
25. 對於速率控制：

    - 在並行中，指定可同時執行命令的受管節點數目或百分比。

> **Note**
>
> 如果透過指定套用至受管節點的標籤或指定 AWS 資源群組選取了目標，且您不確定會以多少個受管節點為目標，則透過指定百分比限制可以同時執行文件之目標的數量。

- 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比之節點上的命令失敗之後，停止在其他受管節點上執行命令。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止傳送命令。仍在處理命令的受管節點也可能會傳送錯誤。

26. (選用) 對於 IAM 服務角色，請選擇一個角色，以便為 Systems Manager 提供執行維護時段任務時承擔的許可。

    如果您未指定服務角色 ARN，則 Systems Manager 會使用帳戶中的服務連結角色。如果帳戶中不存在適當的 Systems Manager 服務連結角色，則會在成功註冊任務時建立該角色。

    > **Note**
    >
    > 為了改善安全狀態，強烈建議您建立自訂政策和自訂服務角色，以便執行維護時段任務。您可以制定政策，僅提供特定維護時段任務所需的許可。如需詳細資訊，請參閱設定 Maintenance Windows。

27. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取啟用將輸出寫入 S3 方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

    > **Note**
    >
    > 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給受管節點之執行個體設定檔的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確認與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

    若要將輸出串流至 Amazon CloudWatch Logs 日誌群組，請選取 CloudWatch output (CloudWatch 輸出) 方塊。在方塊中輸入日誌群組名稱。

28. 在 SNS notifications (SNS 通知) 區段中，如果您要傳送有關命令執行狀態的通知，請選取 Enable SNS notifications (啟用 SNS 通知) 核取方塊。

如需為 Run Command 設定 Amazon SNS 通知的詳細資訊，請參閱 使用 Amazon SNS 通知監控 Systems Manager 狀態變更。

29. 對於 Parameters (參數)：

    - 在 Operation (操作) 中選擇 Scan (掃描) 以掃描遺漏的修補程式，或選擇 Install (安裝) 以掃描並安裝遺漏的修補程式。
    - 您無需在 Snapshot Id (快照 ID) 欄位中輸入任何資訊。此系統會自動產生並提供此參數。
    - 除非您希望 Patch Manager 使用與修補基準不同的修補程式集，否則不需要在 Install Override List (安裝覆寫清單) 欄位中輸入任何內容。如需相關資訊，請參閱 參數名稱：`InstallOverrideList`。
    - 針對重新啟動選項，指定如果在 `Install` 操作期間安裝了修補程式，或是 Patch Manager 偵測到自上次節點重新啟動後安裝的其他修補程式，是否要重新啟動節點。如需相關資訊，請參閱 參數名稱：`RebootOption`。
    - (選用) 在 Comment (註解) 中輸入有關此命令的追蹤註記或提醒。
    - 在 Timeout (seconds) (逾時 (秒)) 中，輸入系統應等待操作完成的時間，超過此時間將視為失敗。

30. 選擇 Register run command task (註冊執行命令任務)。

維護時段任務完成後，您可以在 Fleet Manager工具的 Systems Manager 主控台中檢視修補程式合規詳細資訊。

您也可以在合規報告索引標籤Patch Manager的工具中檢視合規資訊。

您也可以使用 DescribePatchGroupState 與 DescribeInstancePatchStatesForPatchGroup API，以檢視合規的詳細資訊。如需有關修補程式合規資料的詳細資訊，請參閱關於修補程式合規。

使用維護時段進行修補的排程

在您設定修補基線 (以及選用的修補程式群組) 之後，即可使用維護時段將修補程式套用至您的節點。維護時段可讓您指定執行修補程式的時間，不中斷商業運作，以降低對伺服器可用性的影響。維護時段的運作方式如下：

1. 建立維護時段，其中包含您修補操作的排程。
2. 選擇維護時段目標，方法是指定 `Patch Group` 或 `PatchGroup` 標籤作為標籤名稱，並指定您已定義 Amazon Elastic Compute Cloud (Amazon EC2) 標籤的任何值，例如例如「Web 伺

服器」或「US-EAST-PROD」。(如果您已在 EC2 執行個體中繼資料中允許標籤，則必須使用 PatchGroup，不留空格。)

3. 建立新的維護時段任務，並指定 AWS-RunPatchBaseline 文件。

當您設定任務時，您可以選擇掃描節點，或掃描節點並在其上安裝修補程式。如果您選擇掃描節點 ，Patch Manager即 中的工具 AWS Systems Manager， 會掃描每個節點，並產生遺漏修補程式的清單供您檢閱。

如果您選擇掃描並安裝修補程式，則 Patch Manager 會掃描每個節點，並將已安裝修補程式清單與基準中的已核准修補程式清單進行比較。Patch Manager 會識別遺漏的修補程式，然後下載並安裝所有遺漏且已核准的修補程式。

如果您要執行一次性掃描或進行安裝以修復問題，可使用 Run Command 直接呼叫 AWS-RunPatchBaseline 文件。

> **Important**
>
> 安裝修補程式之後，Systems Manager 將重新啟動每個節點。需要重新啟動是為了確保修補程式已正確安裝，並確保系統未讓節點處於可能不良的狀態。(例外：如果 AWS-RunPatchBaseline 文件中的 RebootOption 參數設為 NoReboot，受管節點不會在 Patch Manager 執行之後重新啟動。如需詳細資訊，請參閱 參數名稱：RebootOption。)

註冊維護時段任務時使用虛擬參數

當您在 中的Maintenance Windows工具 中註冊任務時 AWS Systems Manager，您可以指定四個任務類型各不重複的參數。(在 CLI 命令中，這些都是使用 --task-invocation-parameters 選項提供。)

您也可以使用「虛擬參數」語法來參考特定的值，例如 {{RESOURCE_ID}}、{{TARGET_TYPE}} 和 {{WINDOW_TARGET_ID}}。維護時段任務執行時，它會傳遞正確的值，而不是虛擬參數預留位置。本主題後面的支援的虛擬參數中提供了您可以使用之虛擬參數的完整清單。

> **Important**
>
> 對於目標類型 RESOURCE_GROUP，視任務所需的 ID 格式而定，您可以在任務執行時選擇使用 {{TARGET_ID}} 和 {{RESOURCE_ID}} 參考資源。{{TARGET_ID}} 會傳回資源的完整 ARN。{{RESOURCE_ID}} 只會傳回資源的較短名稱或 ID，如下列範例所示。

- {{TARGET_ID}} 格式：arn:aws:ec2:us-east-1:123456789012:instance/i-02573cafcfEXAMPLE
- {{RESOURCE_ID}} 格式：i-02573cafcfEXAMPLE

對於目標類型 INSTANCE，{{TARGET_ID}} 和 {{RESOURCE_ID}} 參數都只產生執行個體 ID。如需詳細資訊，請參閱支援的虛擬參數。
{{TARGET_ID}} 和 {{RESOURCE_ID}}只能用來將 AWS 資源IDs 傳遞至 Automation、Lambda 和 Step Functions 任務。這兩個虛擬參數不能與 Run Command 任務搭配使用。

虛擬參數範例

假設任務的承載 AWS Lambda 需要依其 ID 參考執行個體。

無論您是使用 INSTANCE 或 RESOURCE_GROUP 維護時段目標，都可以使用 {{RESOURCE_ID}} 虛擬參數加以實現。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"
        }
    }
```

如果除了 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體之外，您的 Lambda 任務還要針對其他受支援的目標類型執行 (例如 Amazon DynamoDB 資料表)，則可以使用相同的語法，{{RESOURCE_ID}} 只會產生資料表的名稱。但是，如果您需要資料表的完整 ARN，請使用 {{TARGET_ID}}，如下列範例所示。

```
"TaskArn": "arn:aws:lambda:us-east-2:111122223333:function:SSMTestFunction",
    "TaskType": "LAMBDA",
    "TaskInvocationParameters": {
        "Lambda": {
            "ClientContext": "ew0KICAi--truncated--0KIEXAMPLE",
            "Payload": "{ \"tableArn\": \"{{TARGET_ID}}\" }",
```

```
            "Qualifier": "$LATEST"
        }
    }
```

相同的語法適用於定位執行個體或其他資源類型。已將多個資源類型新增至資源群組時，任務會針對每個適當的資源執行。

> **Important**
>
> 並非資源群組中包含的所有資源類型都會產生 {{RESOURCE_ID}} 參數的值。如需支援的資源類型清單，請參閱支援的虛擬參數。

另一個例子是，若要執行可停止 EC2 執行個體的 Automation 任務，您可將 AWS-StopEC2Instance Systems Manager 文件 (SSM 文件) 指定為 TaskArn 值，並使用 {{RESOURCE_ID}} 虛擬參數：

```
"TaskArn": "AWS-StopEC2Instance",
    "TaskType": "AUTOMATION"
    "TaskInvocationParameters": {
        "Automation": {
            "DocumentVersion": "1",
            "Parameters": {
                "instanceId": [
                    "{{RESOURCE_ID}}"
                ]
            }
        }
    }
```

若要執行複製 Amazon Elastic Block Store (Amazon EBS)磁碟區快照的 Automation 任務，您可以將 AWS-CopySnapshot SSM 文件指定為 TaskArn 值，並使用 {{RESOURCE_ID}} 虛擬參數：

```
"TaskArn": "AWS-CopySnapshot",
    "TaskType": "AUTOMATION"
    "TaskInvocationParameters": {
        "Automation": {
            "DocumentVersion": "1",
            "Parameters": {
                "SourceRegion": "us-east-2",
                "targetType":"RESOURCE_GROUP",
                "SnapshotId": [
```

```
                "{{RESOURCE_ID}}"
            ]
        }
    }
}
```

支援的虛擬參數

以下清單說明您可以在 `--task-invocation-parameters` 選項中使用 {{*PSEUDO_PARAMETER*}} 語法指定的虛擬參數。

- **WINDOW_ID**：目標維護時段 ID。
- **WINDOW_TASK_ID**：正在執行的時段任務 ID。
- **WINDOW_TARGET_ID**：包含目標的目標時段的 ID (目標 ID)。
- **WINDOW_EXECUTION_ID**：目前執行時段的 ID。
- **TASK_EXECUTION_ID**：目前執行任務的 ID。
- **INVOCATION_ID**：目前呼叫的 ID。
- **TARGET_TYPE**：目標類型。支援的類型包括 RESOURCE_GROUP 和 INSTANCE。
- **TARGET_ID**:

  如果您指定的目標類型為 INSTANCE，則 TARGET_ID 虛擬參數會由執行個體的 ID 取代。例如：i-078a280217EXAMPLE。

  如果您指定的目標類型為 RESOURCE_GROUP，則任務執行所參考的值為資源的完整 ARN。例如：arn:aws:ec2:us-east-1:123456789012:instance/i-078a280217EXAMPLE。下表提供資源群組中特定資源類型的範例 TARGET_ID 值。

> **Note**
>
> TARGET_ID 不支援 Run Command 任務。

資源類型	範例 TARGET_ID	
AWS::CloudWatch::Alarm	arn:aws:cloudwatch:us-east-1:123456789012:alarm:MyClou	

資源類型	範例 TARGET_ID	
	dWatchAlarm 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	
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	

資源類型	範例 TARGET_ID
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**：資源群組中所包含資源類型的簡短 ID。下表提供資源群組中特定資源類型的範例 RESOURCE_ID 值。

> **Note**
>
> RESOURCE_ID 不支援 Run Command 任務。

資源類型	範例 RESOURCE_ID
AWS::CloudWatch::Alarm	MyCloudWatchAlarm
AWS::DynamoDB::Table	MyTable
AWS::EC2::Instance	i-078a280217EXAMPLE
AWS::EC2::Image	ami-02250b3732EXAMPLE

資源類型	範例 RESOURCE_ID	
AWS::EC2::Security Group	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 資源群組包含不會產生RESOURCE_ID值的資源類型，且未列在上表中，則不會填入 RESOURCE_ID 參數。該資源仍會發生執行呼叫。在這些情況下，請改用 TARGET_ID 虛擬參數，這將被取代為資源的完整 ARN。

維護時段排程與作用期間選項

當您建立維護時段時，您必須使用 Cron 或 Rate 表達式，指定維護時段執行的頻繁程度。或者，您可以指定一段日期範圍，讓維護時段能根據其定期排程來執行，以及該定期排程依循的時區。

不過請注意，該時區選項與開始日期和結束日期選項不互相影響。您指定的任何開始日期和結束日期時間 (包含或不含時區位移) 只能決定維護時段能定期執行的「有效期間」 。時區選項決定維護時段排程在有效期間「之中」定期執行的國際時區基準。

> **Note**
>
> 您可以使用 ISO-8601 時間戳記格式指定開始和結束日期。例如：2021-04-07T14:29:00-08:00
> 您可以使用網際網路號碼分配局 (IANA) 格式指定時區。例如，America/Chicago、Europe/Berlin 或 Asia/Tokyo。

範例

- 範例 1：指定維護時段開始日期
- 範例 2：指定維護時段開始日期和結束日期
- 範例 3：建立只執行一次的維護時段
- 範例 4：指定維護時段的排程偏移天數

範例 1：指定維護時段開始日期

假設您使用 AWS Command Line Interface (AWS CLI) 建立具有下列選項的維護時段：

- `--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 *)"
```

這表示維護時段在其指定的開始日期和時間前無法第一次執行，也就是美國太平洋時間 2021 年 1 月 1 日星期五的午夜。(此時區比 UTC 時間慢 8 個小時。) 在此情況下，時段期間的開始日期和時間不代表維護時段第一次執行的時間。結合 `--schedule-timezone` 和 `--schedule` 值，代表維護時段會在美國太平洋時區 (IANA 格式的 "America/Los Angeles" (美洲/洛杉磯)) 每週三的上午 9 點執行。啟用期間的第一次執行，會在美國太平洋時間 2021 年 1 月 4 日週三的上午 9 點。

範例 2：指定維護時段開始日期和結束日期

假設您接下來使用這些選項建立維護時段：

- `--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)"
```

此維護時段的啟用期間將會在日本標準時間 2019 年 1 月 1 日的上午 3:15 開始。此維護時段的有效期間會在日本標準時間 2019 年 6 月 30 日星期日的上午 6:15 結束。(此時區比 UTC 時間快 9 個小時)。結合 `--schedule-timezone` 和 `--schedule` 值，代表維護時段會在日本標準時區 (IANA 格式的 "Asia/Tokyo" (亞洲/東京)) 每週二的上午 3:15 執行。這是因為維護時段每七天執行一次，從 1 月 1 日週二上午 3:15 啟用。最後一次執行會在日本標準時間 2019 年 6 月 25 日星期二的上午 3:15。這是已啟用維護時段期間在五天之後結束之前的最後一個週二。

範例 3：建立只執行一次的維護時段

現在您使用此選項建立維護時段：

- `--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
```

此維護時段只會在 UTC 時間 2020 年 7 月 7 日的下午 3:55 執行一次。維護時段已啟用為視需要執行最多五個小時，但在維護時段期間結束的兩個小時之前都無法啟動新的任務。

範例 4：指定維護時段的排程偏移天數

現在您使用此選項建立維護時段：

```
--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
```

排程偏移是在執行維護時段之前，在 CRON 表達式所指定的日期和時間之後等待的天數。

在上述範例中，CRON 表達式會排定維護時段，在每個月的第三個週二晚上 11:30 執行：

```
--schedule "cron(0 30 23 ? * TUE#3 *)
```

但是，包括 `--schedule-offset 2` 表示維護時段將在每個月的第三個星期二「後」兩天的晚上 11:30 執行。

僅 CRON 表達式支援排程偏移。

詳細資訊

- 參考：Systems Manager 的 Cron 和 Rate 運算式
- 使用主控台建立維護時段
- 教學課程：使用 建立和設定維護時段 AWS CLI
- 《AWS Systems Manager API 參考》中的 CreateMaintenanceWindow
- create-maintenance-window 在 AWS Systems Manager AWS CLI 命令參考 的 區段中
- IANA 網站上的時區資料庫

註冊不含目標的維護時段任務

對於建立的每個維護時段，您可以指定執行維護時段時要執行的一或多個任務。在大多數情況下，您必須指定要在其中執行任務的資源或目標。但是，在某些情況下，您不需要在任務中明確指定目標。

必須為維護時段 Systems Manager Run Command 類型任務指定一或多個目標。根據任務的性質，其他維護時段任務類型 (Systems Manager Automation AWS Lambda和 AWS Step Functions) 的目標為選用。

對於 Lambda 和 Step Functions 任務類型，是否需要目標取決於您建立的函數或狀態機的內容。

> **Note**
>
> 當任務已註冊目標 AWS Lambda時，自動化和 AWS Step Functions 任務會從資源群組和標籤解析目標，並為每個已解析的資源傳送一個調用，這會導致多個任務調用。但例如，假設您只想要對向包含多個執行個體的資源群組註冊的 Lambda 任務進行一次調用。在此情況下，如果您在 中工作 AWS Management Console，請在註冊 Lambda 任務或編輯 Lambda 任務頁面中選擇不需要的任務目標選項。 如果您使用的是 AWS CLI 命令，請勿在執行 register-task-with-maintenance-window命令時，使用 `--targets` 參數指定目標update-maintenance-window-task。

在許多情況下，您不需要明確指定自動化任務的目標。例如，假設您正在建立 Automation 類型任務來使用 `AWS-UpdateLinuxAmi` Runbook 更新 Linux 的 Amazon Machine Image (AMI)。當任務執行時，AMI 已更新為可用的最新版本 Linux 發行版本套件和 Amazon 軟體。從 AMI 建立的新執行個體已經安裝這些更新。因為在 Runbook 的輸入參數中指定了要更新的 AMI ID，所以不需要在維護時段任務中再次指定目標。

同樣地，假設您使用 AWS Command Line Interface (AWS CLI) 來註冊使用 `AWS-RestartEC2Instance` Runbook 的維護時段自動化任務。因為要重新啟動的節點是在 `--task-invocation-parameters` 參數中，所以您不需要指定 `--targets` 選項。

> **Note**
>
> 對於未指定目標的維護時段任務，您無法提供 `--max-errors` 和 `--max-concurrency` 的值。系統會插入預留位置值 1，這可能會在回應指令 (例如 describe-maintenance-window-tasks 和 get-maintenance-window-task) 中回報。這些值不會影響任務的執行，可以忽略。

下列範例示範針對無目標維護時段任務，省略 `--targets`、`--max-errors` 和 `--max-concurrency` 選項。

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**
>
> 對於 2020 年 12 月 23 日之前註冊的維護時段任務：如果您為任務指定目標，且不再需要一個目標，您可以使用 Systems Manager 主控台或 update-maintenance-window-task AWS CLI 命令更新該任務以移除目標。

詳細資訊

- 錯誤訊息：「沒有目標的維護時段任務不支援 MaxConcurrency 值」和「沒有目標的維護時段任務不支援 MaxErrors 值」

對維護時段進行故障診斷

使用以下資訊以協助您對維護時段的問題進行故障診斷。

主題

- 編輯任務錯誤：在可編輯維護時段任務的頁面上，IAM 角色清單傳回錯誤訊息：「我們無法找到為此任務指定的 IAM 維護時段角色。它可能已刪除，也可能尚未建立。」
- 並非所有維護時段目標都會更新
- 任務失敗並顯示任務叫用狀態：「提供的角色未包含正確的 SSM 許可。」
- 任務失敗並顯示錯誤消息：「正在驗證和解決步驟輸入時，步驟失敗」
- 錯誤訊息：「沒有目標的維護時段任務不支援 MaxConcurrency 值」和「沒有目標的維護時段任務不支援 MaxErrors 值」

編輯任務錯誤：在可編輯維護時段任務的頁面上，IAM 角色清單傳回錯誤訊息：「我們無法找到為此任務指定的 IAM 維護時段角色。它可能已刪除，也可能尚未建立。」

問題 1：您最初指定的 AWS Identity and Access Management (IAM) 維護時段角色在您建立任務後已刪除。

Possible fix (可能的修正)：1) 選取不同的 IAM 維護時段角色 (如果您帳戶中有現有的)，或者建立新的角色，並為任務選取此角色。

問題 2：如果任務是使用 AWS Command Line Interface (AWS CLI) AWS Tools for Windows PowerShell或 AWS SDK 建立的，則可能已指定不存在的 IAM 維護時段角色名稱。例如，在建立任務之前，IAM 維護時段角色可能已遭到刪除，或者角色名稱輸入錯誤，例如 **myrole** 而不是 **my-role**。

Possible fix (可能的修正)：選取要使用之 IAM 維護時段角色的正確名稱，或建立新的角色，並為任務指定此角色。

並非所有維護時段目標都會更新

問題：您注意到維護時段任務並未在維護時段目標的所有資源上執行。例如，在維護時段的執行結果中，該資源的任務會標示為失敗或逾時。

解決方案：維護時段任務未在目標資源上執行的最常見原因涉及連線和可用性。例如：

- Systems Manager 在維護時段操作之前或期間失去資源的連線。
- 在維護時段操作期間，資源已離線或停用。

您可以等待下一個排定的維護時段時間，在資源上執行任務。您可以在無法使用或離線的資源上手動執行維護時段任務。

任務失敗並顯示任務叫用狀態：「提供的角色未包含正確的 SSM 許可。」

問題：您已為任務指定維護時段服務角色，但任務無法成功執行，且任務叫用狀態會報告「提供的角色未包含正確的 SSM 許可」。

- 解決方案：在 任務 1：使用主控台為維護時段服務角色建立自訂政策 中，我們提供您可以連接至自訂維護時段服務角色的基本政策。此政策包含許多任務場景所需的許可。不過，由於您可以執行的任務種類繁多，您可能需要在維護時段角色的政策中提供其他許可。

  例如，某些自動化動作適用於 AWS CloudFormation 堆疊。因此，您可能需要為維護時段服務角色的政策新增其他許可：`cloudformation:CreateStack`、`cloudformation:DescribeStacks` 以及 `cloudformation:DeleteStack`。

  另一個例子：Automation Runbook `AWS-CopySnapshot` 需建立 Amazon Elastic Block Store (Amazon EBS) 快照的權限。因此，您可能需要新增許可 `ec2:CreateSnapshot`。

  如需有關 AWS 受管 Automation Runbook 所需角色許可的資訊，請參閱 Automation Runbook AWS Systems Manager 參考中的 Runbook 說明。

如需有關 AWS 受管 SSM 文件所需角色許可的資訊，請在文件區段 Systems Manager 主控台中檢閱文件的內容。

如需有關 Step Functions 任務、Lambda 任務以及自訂 Automation Runbook 和 SSM 文件所需角色許可的資訊，請向這些資源的作者確認許可需求。

任務失敗並顯示錯誤消息：「正在驗證和解決步驟輸入時，步驟失敗」

問題：您在任務中使用的 Automation Runbook 或 Systems Manager 命令文件需要您指定輸入，例如 `InstanceId` 或 `SnapshotId`，但未提供值或未正確提供值。

- Solution 1 (解決方案 1)：如果您的任務是針對單一資源 (例如單一節點或單一快照)，請在任務的輸入參數中輸入其 ID。
- Solution 2 (解決方案 2)：如果您的任務是針對多個資源，例如當您使用 Runbook `AWS-CreateImage` 時，從多個節點建立映像，您可以在輸入參數中使用支援維護時段任務的其中一個虛擬參數來表示命令中的節點 ID。

  以下命令使用 AWS CLI向維護時段註冊 Systems Manager Automation 任務。`--targets` 值表示維護時段目標 ID。此外，即使 `--targets` 參數會指定時段目標 ID，Automation Runbook 的參數需要提供節點 ID。在這種情況下，命令會使用虛擬參數 `{{RESOURCE_ID}}` 作為 `InstanceId` 值。

  AWS CLI 命令：

  Linux & macOS

  下列範例命令會重新啟動屬於維護時段目標群組且ID 為 e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE 的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。

```
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"
```

如需使用維護時段任務之虛擬參數的詳細資訊，請參閱 註冊維護時段任務時使用虛擬參數 和 任務註冊範例。

錯誤訊息：「沒有目標的維護時段任務不支援 MaxConcurrency 值」和「沒有目標的維護時段任務不支援 MaxErrors 值」

問題：當註冊 Run Command 類型任務時，您必須至少指定一個目標，以執行任務。對於其他任務類型 （ 自動化和 AWS Step Functions) AWS Lambda，根據任務的性質，目標是選用的。選項 `MaxConcurrency` (同時執行任務的資源數量) 和 `MaxErrors` (在任務失敗之前在目標資源上執行任務的失敗次數) 不需要或不支援未指定目標的維護時段任務。如果在沒有指定任務目標時為這些選項的其中之一指定了值，則系統會產生這些錯誤訊息。

解決方案：如果您收到其中一個錯誤，請移除並行和錯誤閾值的值，然後繼續註冊或更新維護時段任務。

如需有關執行未指定目標之任務的詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 註冊不含目標的維護時段任務。

AWS Systems Manager Quick Setup

使用 中的Quick Setup工具 AWS Systems Manager，以建議的最佳實務快速設定常用的 Amazon Web Services 服務和功能。 透過自動化常見或建議的任務，Quick Setup簡化了包括 Systems Manager 在內的服務設定。這些任務包括建立必要 AWS Identity and Access Management (IAM) 執行個體描述檔角色和設定操作最佳實務，例如定期修補程式掃描和庫存收集。使用 Quick Setup 無需付費。但是，根據您設定的服務類型以及使用限制可能會產生費用，而對用於設定服務的服務不會收取任何費用。若要開始使用 Quick Setup，請開啟 Systems Manager 主控台。在導覽窗格中，選擇 Quick Setup。

> **Note**
>
> 如果您被引向 Quick Setup，以協助您將執行個體設定為由 Systems Manager 管理，則請完成使用 Quick Setup 設定 Amazon EC2 主機管理 中的程序。

Quick Setup 有哪些優點？

Quick Setup 的優點包括：

- 簡化服務與功能組態

  Quick Setup 會帶您演練設定操作最佳實務，並自動部署這些組態。Quick Setup 儀表板會顯示組態部署狀態的即時檢視。

- 跨多個帳戶自動部署設定

  您可以在個別 Quick Setup中使用 AWS 帳戶 ，或跨多個 AWS 帳戶 和 AWS 區域 進行整合 AWS Organizations。跨多個帳戶使用 Quick Setup，有助於確保您的組織維持一致的組態。

- 消除組態漂移

  每當使用者對透過 Quick Setup 所做的選擇相衝突的服務或功能進行任何變更，即會發生漂移。Quick Setup 會定期檢查組態漂移並嘗試進行修補。

誰應該使用Quick Setup？

對於已經擁有所設定服務和功能的經驗，並且想要簡化設定程序的客戶來說，Quick Setup 非常有用。如果您不熟悉 AWS 服務 使用 設定的 Quick Setup，建議您進一步了解 服務。請先檢閱相關使用者指南中的內容，然後再使用 Quick Setup 建立組態。

AWS 區域中的 Quick Setup 可用性

在下文中 AWS 區域，您可以按照 中的設定，或僅針對您選擇的組織帳戶和區域 AWS Organizations，使用整個組織的所有Quick Setup組態類型。您也可以在這些區域中以單一帳戶來使用 Quick Setup。

- 美國東部 (俄亥俄)
- 美國東部 (維吉尼亞北部)
- 美國西部 (加利佛尼亞北部)
- 美國西部 (奧勒岡)
- 亞太區域 (孟買)
- 亞太區域 (首爾)
- 亞太區域 (新加坡)
- 亞太區域 (雪梨)
- 亞太區域 (東京)
- 加拿大 (中部)
- 歐洲 (法蘭克福)
- 歐洲 (斯德哥爾摩)
- 歐洲 (愛爾蘭)
- 歐洲 (倫敦)
- Europe (Paris)
- 南美洲 (聖保羅)

在下列區域中，個別帳戶僅能使用主機管理組態類型：

- 歐洲 (米蘭)
- 亞太區域 (香港)
- Middle East (Bahrain)
- 中國 (北京)
- 中國 (寧夏)
- AWS GovCloud （美國東部）
- AWS GovCloud （美國西部）

如需取得 Systems Manager 所有支援的區域之清單，請參閱 Amazon Web Services 一般參考 中 Systems Manager 服務端點裡的區域直欄。

Quick Setup 入門

請使用本主題中的資訊來熟悉 Quick Setup 的使用。

主題

- 用於 Quick Setup 登入的 IAM 角色和許可
- 以程式設計方式使用 Quick Setup API 的手動入門

用於 Quick Setup 登入的 IAM 角色和許可

Quick Setup 推出新的主控台體驗和新的 API。現在您可以使用 主控台 AWS CLI AWS CloudFormation、 和 SDKs 與此 API 互動。如果選擇加入新的體驗，則會使用新的 API 重新建立現有的組態。根據帳戶中現有的組態數量，此程序可能需要幾分鐘的時間。

若要使用 Quick Setup 的新主控台，您必須擁有以下動作的許可：

```
{
    "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::account-id: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"
        }
    ]
}
```

若要將使用者限制為唯讀許可，則僅允許 Quick Setup API 的 `ssm-quicksetup:List*` 和 `ssm-quicksetup:Get*` 操作。

在加入期間， 會代表您Quick Setup建立下列 AWS Identity and Access Management (IAM) 角色：

- `AWS-QuickSetup-LocalExecutionRole`：授予 AWS CloudFormation 許可，以便使用任何範本 (不包括修補程式政策範本)，以及建立必要的資源。
- `AWS-QuickSetup-LocalAdministrationRole` – 准許 AWS CloudFormation 擔任 `AWS-QuickSetup-LocalExecutionRole`。
- `AWS-QuickSetup-PatchPolicy-LocalExecutionRole` – 准許 AWS CloudFormation 使用修補程式政策範本，並建立必要的資源。
- `AWS-QuickSetup-PatchPolicy-LocalAdministrationRole` – 准許 AWS CloudFormation 擔任 `AWS-QuickSetup-PatchPolicy-LocalExecutionRole`。

如果您要加入管理帳戶 - 您用來在其中建立組織的帳戶 AWS Organizations-Quick Setup 也會代表您建立下列角色：

- `AWS-QuickSetup-SSM-RoleForEnablingExplorer` – 授予 `AWS-EnableExplorer` 自動化 Runbook 許可。`AWS-EnableExplorer` Runbook 會設定 Systems Manager 中的Explorer工具 ，以顯示多個 AWS 帳戶 和 的資訊 AWS 區域。
- `AWSServiceRoleForAmazonSSM` – 服務連結角色，授予 Systems Manager 管理和使用 AWS 之資源的存取權。
- `AWSServiceRoleForAmazonSSM_AccountDiscovery` – 服務連結角色，授予 Systems Manager 在同步資料時呼叫 AWS 服務 以探索 AWS 帳戶 資訊的許可。如需詳細資訊，請參閱使用角色來收集 OpsCenter和 AWS 帳戶 的資訊 Explorer。

登入管理帳戶時，Quick Setup 會在 AWS Organizations 與 CloudFormation 之間啟用受信任存取，以跨組織部署 Quick Setup 組態。若要啟用受信任存取，您的管理帳戶必須擁有管理員許可。登入後，您不再需要管理員許可。如需詳細資訊，請參閱啟用 Organizations 的受信任存取。

如需 AWS Organizations 帳戶類型的相關資訊，請參閱AWS Organizations 《 使用者指南》中的AWS Organizations 術語和概念。

> **Note**
>
> Quick Setup 使用 AWS CloudFormation StackSets 跨 AWS 帳戶 和 區域部署您的組態。如果目標帳戶數目乘以區域數目得出的結果超過 10,000，則組態無法部署。我們建議您審視您的使用案例，並建立使用較少目標的組態，以因應貴組織的成長。堆疊執行個體不會部署至您組織的管理帳戶。如需詳細資訊，請參閱建立具有服務受管許可的堆疊集時的考量。

以程式設計方式使用 Quick Setup API 的手動入門

如果透過主控台來使用 Quick Setup，此服務會為您處理加入步驟。如果您計劃使用 SDKs或 AWS CLI 來使用 Quick Setup API，您仍然可以使用 主控台來完成入門步驟，因此您不需要手動執行這些步驟。不過，有些客戶需要透過程式設計方式完成 Quick Setup 的加入步驟，無需與主控台互動。如果此方法符合使用案例，則您必須完成下列步驟。所有這些步驟都必須從您的 AWS Organizations 管理帳戶完成。

完成 Quick Setup 的手動加入

1. AWS CloudFormation 使用 Organizations 啟用 的受信任存取。這為管理帳戶提供為組織建立和管理 StackSets 所需的許可。您可以使用 AWS CloudFormation的 `ActivateOrganizationsAccess` API 動作來完成此步驟。如需詳細資訊，請參閱《AWS CloudFormation API 參考》中的 ActivateOrganizationsAccess。
2. 啟用 Systems Manager 與 Organizations 的整合。這可讓 Systems Manager 在組織的所有帳戶中建立服務連結角色。這也允許 Systems Manager 在組織及其帳戶中代表您執行操作。您可以使用 AWS Organizations的 `EnableAWSServiceAccess` API 動作來完成此步驟。Systems Manager 的服務主體為 `ssm.amazonaws.com`。如需詳細資訊，請參閱《AWS Organizations API 參考》中的 EnableAWSServiceAccess。
3. 為 Explorer 建立必要的 IAM 角色。這可讓 Quick Setup 為組態建立儀表板，便於您檢視部署和關聯狀態。建立 IAM 角色，以及連接 `AWSSystemsManagerEnableExplorerExecutionPolicy` 受管政策。修改角色的信任政策以符合下列條件。把每個*## ID* 取代為您的資訊。

```
{
    "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:*:account ID:automation-execution/
*"
                }
            }
        }
    ]
}
```

4. 為 Explorer 更新 Quick Setup 的服務設定。您可以使用 Quick Setup 的 `UpdateServiceSettings` API 動作來完成此步驟。指定您在上一個步驟中為 `ExplorerEnablingRoleArn` 請求參數建立的 IAM 角色 ARN。如需詳細資訊，請參閱《Quick Setup API 參考》中的 UpdateServiceSettings。

5. 建立 AWS CloudFormation StackSets 要使用的必要 IAM 角色。您必須建立執行角色和管理角色。

   a. 建立執行角色。執行角色應至少連接其中一個 `AWSQuickSetupDeploymentRolePolicy` 或 `AWSQuickSetupPatchPolicyDeploymentRolePolicy` 受管政策。如果您只是要建立修補程式政策組態，則可以使用 `AWSQuickSetupPatchPolicyDeploymentRolePolicy` 受管政策。所有其他組態都使用 `AWSQuickSetupDeploymentRolePolicy` 政策。修改角色的信任政策以符合下列條件。將每個*## ID* 和*######*取代為您的資訊。

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
   ```

```
            "Principal": {
                "AWS": "arn:aws:iam::account ID:role/administration role name"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

b. 建立管理角色。此許可政策必須符合下列條件。將每個## ID 和######取代為您的資訊。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "sts:AssumeRole"
            ],
            "Resource": "arn:*:iam::account ID:role/execution role name",
            "Effect": "Allow"
        }
    ]
}
```

修改角色的信任政策以符合下列條件。把每個## ID 取代為您的資訊。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "cloudformation.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "account ID"
                },
                "StringLike": {
                    "aws:SourceArn": "arn:aws:cloudformation:*:account
 ID:stackset/AWS-QuickSetup-*"
                }
            }
```

```
        }
    ]
}
```

使用 Quick Setup 的委派管理員

註冊 的委派管理員帳戶後Quick Setup，在該帳戶中具有適當許可的使用者可以建立、更新、檢視和刪除以 AWS Organizations 結構中組織單位為目標的組態管理員。此委派管理員帳戶也可以管理您組織管理帳戶先前建立的組態管理員。

Organizations 中的管理帳戶可以將組織中的一個帳戶指定為委派管理員。當您將帳戶註冊為 的委派管理員時Quick Setup，此帳戶也會自動成為 StackSets 和 Systems Manager Explorer 的 AWS CloudFormation 委派管理員，因為這些服務需要部署和監控Quick Setup組態。

> **Note**
>
> 目前，Quick Setup 委派管理員不支援修補程式政策組態類型。組織的修補程式政策組態必須在組織的管理帳戶中建立和維護。如需詳細資訊，請參閱the section called “建立修補程式政策”。

下列主題說明如何註冊和取消註冊 Quick Setup 的委派管理員。

主題

- 註冊 Quick Setup 的委派管理員
- 取消註冊 Quick Setup 的委派管理員

註冊 Quick Setup 的委派管理員

按照下列程序註冊 Quick Setup 的委派管理員。

註冊 Quick Setup 委派管理員

1. 登入您的 AWS Organizations 管理帳戶。
2. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
3. 在導覽窗格中，選擇 Quick Setup。

4. 選擇設定。
5. 在 Delegated administrator for Quick Setup (Explorer 的委派管理員) 區段中，確認您已設定必要的服務連結角色和服務存取選項。如有必要，請選擇 Create role (建立角色) 和 Enable access (啟用存取) 按鈕設定這些選項。
6. 針對帳戶 ID，輸入 AWS 帳戶 ID。此帳戶必須是其中的成員帳戶 AWS Organizations。
7. 選擇 Register delegated administrator (註冊委派管理員)。

取消註冊 Quick Setup 的委派管理員

按照下列程序取消註冊 Quick Setup 的委派管理員。

取消註冊 Quick Setup 的委派管理員

1. 登入您的 AWS Organizations 管理帳戶。
2. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
3. 在導覽窗格中，選擇 Quick Setup。
4. 選擇設定。
5. 在 Quick Setup 的委派管理員區段中，從動作下拉式清單中選擇取消註冊。
6. 選取確認。

了解 Quick Setup 的術語和詳細資訊

Quick Setup中的工具 AWS Systems Manager會在Quick Setup首頁的組態管理員資料表 AWS 區域 中顯示您跨所有 建立的所有組態管理員的結果。在此頁面上，您可以就每個組態 View details (檢視詳細資訊)、從 Actions (動作) 下拉式選單中刪除組態，也可以 Create (建立) 組態。組態管理員資料表包含以下資訊：

- 名稱：建立時提供的組態管理員名稱。
- Configuration type (組態類型)：建立組態時選擇的組態類型。
- 版本：目前部署的組態類型版本。
- Organizational units (組織單位)：如果選擇了一組 Custom (自訂) 目標，顯示要將組態部署到其中的組織單位 (OU)。組織單位和自訂目標僅可用於組織的管理帳戶。管理帳戶是您在 AWS Organizations中用於建立組織的帳戶。
- 部署類型 – 指出部署適用於整個組織 (`Organizational`)，還是僅適用於您的帳戶 (`Local`)。

- Regions (區域)：如果選擇了一組 Custom (自訂) 目標或 Current account (目前帳戶) 中的目標，要將組態部署到其中的區域。
- 部署狀態 – 部署狀態會指出 是否 AWS CloudFormation 已成功部署目標或堆疊執行個體。目標和堆疊執行個體包含您在建立組態期間選擇的組態選項。
- Association status (關聯狀態)：關聯狀態是由您建立的組態所建立之所有關聯的狀態。所有目標的關聯必須成功執行，否則狀態為 Failed (失敗)。

  Quick Setup 會為每個組態目標建立並執行State Manager關聯。 State Manager 是 中的工具 AWS Systems Manager。

若要檢視部署到目前瀏覽區域的組態，請選取組態索引標籤。

組態詳細資訊

Configuration details (組態詳細資訊) 頁面會顯示組態及其相關關聯部署的資訊。您可以在此頁面中編輯組態選項、更新目標或刪除組態。您也可以檢視每個組態部署的詳細資訊，從而了解關聯的更多資訊。

根據組態，會顯示下列一或多個狀態圖表：

組態部署狀態

顯示成功、失敗、執行中或待定部署的數量。部署發生在包含受組態影響之節點的指定目標帳戶和區域中。

組態關聯狀態

顯示成功、失敗或待定 State Manager 關聯的數量。Quick Setup 在每個部署中為所選組態選項建立關聯。

設定狀態

顯示組態類型所執行的動作數目及其目前狀態。

支援合規

顯示符合組態指定政策的支援數量。

Configuration details (組態詳細資訊) 資料表會顯示組態部署的相關資訊。您可以檢視有關每個部署的詳細資訊，方法是依序選取部署和 View details (檢視詳細資訊)。每個部署的詳細資訊頁面會顯示部署到該部署中節點的關聯。

編輯和刪除您的組態

您可以從 Configuration details (組態詳細資訊) 頁面編輯組態的組態選項，方法是依序選擇 Actions (動作) 和 Edit configuration options (編輯組態選項)。將新選項新增至組態後，Quick Setup 會執行部署並建立新關聯。從組態中移除選項後，Quick Setup 會執行部署並移除任何相關關聯。

> **Note**
>
> 您可以隨時為帳戶編輯 Quick Setup 組態。若要編輯 Organization (組織) 組態，Configuration status (組態狀態) 必須為 Success (成功) 或 Failed (失敗)。

您也可以透過選擇 Actions (動作) 和 Add OUs (新增 OU)、Add Regions (新增區域)、Remove OUs (移除 OU) 或 Remove Regions (移除區域)，來更新組態中包含的目標。如果您的帳戶未設定為管理帳戶，或您僅建立目前帳戶的組態，則無法更新目標組織單位 OUs)。移除區域或 OU 會從這些區域或 OU 中移除關聯。

Quick Setup 會定期發行新版本的組態。您可以選取升級組態選項，將組態升級到最新版本。

透過依序選擇組態、Actions (動作) 和 Delete configuration (刪除組態)，您可以從 Quick Setup 中刪除組態。您也可以從 Configuration details (組態詳細資訊) 頁面的 Actions (動作) 下拉式清單中，選擇 Delete configuration (刪除組態) 來刪除組態。Quick Setup 隨即會提示您 Remove all OUs and Regions (移除所有 OU 和區域)，而這可能需要一些時間才能完成。刪除組態還會刪除所有相關關聯。此兩步驟刪除過程會將所有已部署資源從所有帳戶和區域中移除，然後刪除組態。

組態合規

您可以檢視您的執行個體是否符合 或 Explorer 合規中組態建立的關聯，兩者都是 中的工具 AWS Systems Manager。若要進一步了解合規，請參閱 了解有關合規性的詳細資訊。若要進一步了解在 Explorer 中檢視合規，請參閱 AWS Systems Manager Explorer。

使用Quick Setup API 來管理組態和部署

您可以使用 提供的 APIQuick Setup，使用 AWS CLI 或您偏好的 SDK 來建立和管理組態和部署。您也可以使用 AWS CloudFormation 來建立部署組態的組態管理員資源。您可以使用 API 來建立用於部署組態定義的組態管理員。組態定義包含部署特定組態類型所需的全部必要資訊。如需有關 Quick Setup API 的詳細資訊，請參閱 Quick Setup API Reference。

下列範例示範如何使用 AWS CLI 和 建立組態管理員 AWS CloudFormation。

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
```

以下是包含修補程式政策組態定義的 JSON 字串範例。

```
'{"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"
```

支援的 Quick Setup 組態類型

支援的組態類型

Quick Setup 會逐步引導您設定許多 Systems Manager 和其他 的操作最佳實務 AWS 服務，並自動部署這些組態。Quick Setup 儀表板會顯示組態部署狀態的即時檢視。

您可以透過與 整合，Quick Setup在個別 AWS 帳戶 或跨多個 AWS 帳戶 和 區域使用 AWS Organizations。跨多個帳戶使用 Quick Setup，有助於確保您的組織維持一致的組態。

Quick Setup 支援下列組態類型。

- 使用 Quick Setup 設定 Amazon EC2 主機管理
- 使用 Quick Setup 為組織設定預設主機管理組態
- 使用 Quick Setup 建立 AWS Config組態記錄器
- 使用 部署 AWS Config 一致性套件 Quick Setup
- 使用修補程式政策設定組織中執行個體的Quick Setup修補
- Change Manager 組織設定
- 使用 Quick Setup 設定 DevOps Guru
- 使用 Quick Setup 部署 Distributor 套件
- 使用Quick Setup依照排程自動停止和啟動 EC2 執行個體
- OpsCenter 組織設定
- AWS 資源總管 使用 設定 Quick Setup

使用 Quick Setup 設定 Amazon EC2 主機管理

使用 中的Quick Setup工具 AWS Systems Manager，快速設定 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上所需的安全角色和常用的 Systems Manager 工具。您可以在Quick Setup個

別帳戶或跨多個帳戶使用 ， AWS 區域 並與 整合 AWS Organizations。這些工具可協助您管理和監控執行個體的運作狀態，同時提供開始使用所需的最低許可。

如果您不熟悉 Systems Manager 服務和功能，我們建議您檢閱《AWS Systems Manager 使用者指南》，然後使用 Quick Setup 建立組態。如需有關 Systems Manager 的詳細資訊，請參閱 什麼是 AWS Systems Manager？。

Important

如果您符合下列其中一種情況，則 Quick Setup 可能不是 EC2 管理的正確工具：

- 您第一次嘗試建立 EC2 執行個體來試用 AWS 功能。
- 您仍然是 EC2 執行個體管理的新手。

反之，建議您探索下列內容：

- Amazon EC2 入門
- 《Amazon EC2 使用者指南》中的 Launch an instance using the new launch instance wizard
- 《Amazon EC2 使用者指南》中的 Tutorial: Get started with Amazon EC2 Linux instances

如果您已熟悉 EC2 執行個體管理，並且想要簡化多個 EC2 執行個體的組態和管理，則請使用 Quick Setup。無論您的組織擁有數十個、數千個還是數百萬個 EC2 執行個體，皆請使用下列 Quick Setup 程序，一次性為其設定多個選項。

Note

此組態類型可讓您為定義於 的整個組織 AWS Organizations、僅部分組織帳戶和區域或單一帳戶設定多個選項。其中一個選項是每兩週檢查並套用更新至 SSM Agent。如果您是組織管理員，也可以選擇使用預設主機管理組態類型，每兩週以代理程式更新組織中的所有 EC2 執行個體。如需相關資訊，請參閱使用 Quick Setup 為組織設定預設主機管理組態。

為 EC2 執行個體設定主機管理選項

若要設定主機管理，請在 AWS Systems Manager Quick Setup主控台中執行下列任務。

開啟主機管理組態頁面

1. 開啟 AWS Systems Manager 主控台，網址為 https：//https://console.aws.amazon.com/systems-manager/。
2. 在導覽窗格中，選擇 Quick Setup。
3. 在主機管理卡中，選擇建立。

> **Tip**
>
> 如果您的帳戶中已有一或多個組態，請先選擇程式庫索引標籤或組態區段中的建立按鈕，以檢視卡。

設定 Systems Manager 主機管理選項

- 若要設定 Systems Manager 功能，請在組態選項區段中，選擇要為組態啟用的 Systems Manager 群組中的選項：

  每兩週更新一次 Systems Manager (SSM) Agent

  讓 Systems Manager 能夠每兩週檢查一次是否有新版本的代理程式。如果有新版本，則 Systems Manager 會自動將受管節點上的代理程式更新為最新發行版本。Quick Setup 不會在沒有代理程式的執行個體上安裝代理程式。如需有關哪些 AMIs 已預先安裝 SSM Agent 的資訊，請參閱 尋找預先安裝了 SSM Agent的 AMIs。

  建議您選擇此選項，以確保節點始終執行最新版本的 SSM Agent。如需 SSM Agent 的詳細資訊，包括如何手動安裝代理程式的資訊。請參閱 使用 SSM Agent。

  每 30 分鐘從執行個體收集一次庫存資訊

  讓 Quick Setup 能夠設定下列類型中繼資料的收集：

  - AWS 元件 – EC2 驅動程式、代理程式、版本等。
  - 應用程式 – 應用程式名稱、發佈者、版本等。
  - 節點詳細資訊 – 系統名稱、作業系統 (OS) 名稱、作業系統版本、上次啟動時間、DNS、網域、工作團隊、作業系統架構等。
  - 網路組態 – IP 地址、MAC 地址、DNS、閘道、子網路遮罩等。

- 服務 – 名稱、顯示名稱、狀態、相依服務、服務類型、啟動類型等 (僅限 Windows Server 節點)。
- Windows 角色 – 名稱、顯示名稱、路徑、功能類型、安裝狀態等 (僅限 Windows Server 受管節點)。
- Windows 更新 – Hotfix ID、安裝人員、安裝日期等 (僅限 Windows Server 節點)。

如需 中的工具庫存的詳細資訊 AWS Systems Manager，請參閱 AWS Systems Manager庫存。

> **Note**
>
> 即使您只選取幾個節點，Inventory collection (庫存收集) 選項可能需要最多 10 分鐘才能完成。

每天掃描執行個體是否遺漏修補程式

讓 Systems Manager 中的Patch Manager工具 每天掃描您的節點，並在合規頁面產生報告。此報告會根據預設修補基準來顯示修補程式相容的受管節點數量。此報告包含每個節點的清單及其合規狀態。

如需修補操作和修補程式基準的資訊，請參閱 AWS Systems Manager Patch Manager。

如需有關修補程式合規資訊，請參閱 Systems Manager Compliance (合規) 頁面。

如需有關在一個組態中修補多個帳戶和區域中受管節點的資訊，請參閱 Quick Setup中的修補程式政策組態 和 使用修補程式政策設定組織中執行個體的Quick Setup修補。

> **Important**
>
> Systems Manager 支援多種掃描受管節點的方法，以檢查修補程式合規性。如果一次實作多個方法，則您看到的修補程式合規資訊永遠是最近一次掃描的結果。先前掃描的結果會覆寫。如果掃描方法使用不同的修補基準，且具有不同的核准規則，則修補程式合規資訊可能會意外變更。如需詳細資訊，請參閱避免意外覆寫修補程式合規資料。

設定 Amazon CloudWatch 主機管理選項

- 若要設定 CloudWatch 功能，請在組態選項區段中，選擇要為組態啟用的 Amazon CloudWatch 群組中的選項：

  安裝並設定 CloudWatch 代理程式

  在 Amazon EC2 執行個體上安裝 CloudWatch 統一代理程式的基本組態。代理程式會從 Amazon CloudWatch 的執行個體收集指標和日誌檔案。其還會整合此資訊，以便您快速判斷執行個體的運作狀態。如需 CloudWatch 代理程式基本組態的詳細資訊，請參閱《CloudWatch 代理程式預先指定的指標集》。可能需要額外付費。如需詳細資訊，請參閱 Amazon CloudWatch 定價。

  每 30 天更新一次 CloudWatch 代理程式

  讓 Systems Manager 能夠每 30 天檢查一次是否有新版本的 CloudWatch 代理程式。如果有新版本，則 Systems Manager 會更新執行個體上的代理程式。建議您選擇此選項，以確保執行個體始終執行最新版本的 CloudWatch 代理程式。

設定 Amazon EC2 啟動代理程式主機管理選項

- 若要設定 Amazon EC2 啟動代理程式功能，請在組態選項區段中，選擇要為組態啟用的 Amazon EC2 啟動代理程式群組中的選項：

  每 30 天更新一次 EC2 啟動代理程式

  讓 Systems Manager 能夠每 30 天檢查一次執行個體上是否安裝了新版本的啟動代理程式。如果有新版本，則 Systems Manager 會更新執行個體上的代理程式。建議您選擇此選項，以確保執行個體始終執行最新版本的適用啟動代理程式。對於 Amazon EC2 Windows 執行個體，此選項支援 EC2Launch、EC2Launch v2 和 EC2Config。對於 Amazon EC2 Linux 執行個體，此選項支援 `cloud-init`。對於 Amazon EC2 Mac 執行個體，此選項支援 `ec2-macos-init`。Quick Setup 不支援更新安裝在啟動代理程式不支援的作業系統上或 AL2023 上的啟動代理程式。

  如需這些初始化代理程式的詳細資訊，請參閱下列主題：

  - 使用 EC2Launch v2 設定 Windows 執行個體

- 使用 EC2Launch 設定 Windows 執行個體
- 使用 EC2Config 服務設定 Windows 執行個體
- cloud-init 文件
- ec2-macos-init

選取要由主機管理組態更新的 EC2 執行個體

- 在目標區段中，選擇對應方法來判斷要部署組態的帳戶和區域：

> **Note**
>
> 您無法建立多個以相同 AWS 區域為目標的 Quick Setup 主機管理組態。

Entire organization

您的組態會部署到組織中的所有組織單位 OUs) 和 AWS 區域 。

> **Note**
>
> Entire organization (整個組織) 選項只有在您從組織的管理帳戶設定主機管理時才可用。

Custom

1. 在目標 OU 區段中，選取要部署此主機管理組態的 OU。
2. 在目標區域區段中，選取要部署此主機管理組態的區域。

Current account

選擇其中一個區域選項，然後依照該選項的步驟操作。

目前區域

選擇如何僅以目前區域中的執行個體為目標：

- 所有執行個體：主機管理組態會自動將目前區域中的每個 EC2 設定為目標。
- 標籤：選擇新增，然後輸入要新增至目標執行個體的索引鍵和選用值。
- 資源群組：在資源群組欄位中，選取包含目標 EC2 執行個體的現有資源群組。
- 手動：在執行個體區段中，選取每個目標 EC2 執行個體的核取方塊。

選擇區域

透過選擇下列其中一項，選擇如何在指定的區域中鎖定目標執行個體：

- 所有執行個體：指定區域中的所有執行個體都是目標執行個體。
- 標籤：選擇新增，然後輸入已新增至目標執行個體的索引鍵和選用值。

在目標區域區段中，選取要部署此主機管理組態的區域。

指定執行個體設定檔選項

- 僅限整個組織和自訂目標。

  在執行個體設定檔選項區段中，選擇是要將所需的 IAM 政策新增至連接到執行個體的現有執行個體設定檔，還是要允許Quick Setup使用您所選組態所需的許可建立 IAM 政策和執行個體設定檔。

指定所有組態選項後，選擇建立。

使用 Quick Setup 為組織設定預設主機管理組態

使用 中的Quick Setup工具 AWS Systems Manager，您可以針對已新增至組織的所有帳戶和區域啟用預設主機管理組態 AWS Organizations。如此可確保組織中所有 Amazon Elastic Compute Cloud (EC2) 執行個體的 SSM Agent 會處於最新狀態，且可連線到 Systems Manager。

開始之前

啟用此設定之前，請先確認符合下列要求。

- 最新版 SSM Agent 已安裝在組織中要管理的所有 EC2 執行個體上。
- 要管理的 EC2 執行個體正在使用 Instance Metadata Service Version 2 (IMDSv2)。
- 您會使用具有管理員許可的 AWS Identity and Access Management (IAM) 身分（使用者、角色或群組） AWS Organizations，登入組織的管理帳戶，如 中所指定。

使用預設 EC2 執行個體管理角色

預設主機管理組態會使用 Systems Manager 的 `default-ec2-instance-management-role` 服務設定。這是具有許可的角色，您希望組織中所有帳戶皆可使用該角色，以允許執行個體上的 SSM Agent 與雲端中的 Systems Manager 服務之間進行通訊。

如果您已使用 update-service-setting CLI 命令設定此角色，則預設主機管理組態會使用該角色。如果您尚未設定此角色，Quick Setup 會為您建立並套用角色。

若要檢查是否已為您的組織指定此角色，請使用 get-service-setting 命令。

啟用每兩週自動更新 SSM Agent

使用下列程序，為您的整個 AWS Organizations 組織啟用預設主機管理組態選項。

若要啟用每兩週自動更新 SSM Agent

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Quick Setup。
3. 在預設主機管理組態卡中，選擇建立。

> **Tip**
>
> 如果您的帳戶中已有一或多個組態，請先選擇程式庫索引標籤或組態區段中的建立按鈕，以檢視卡。

4. 在組態選項區段中，選取啟用 SSM Agent 每兩週的自動更新。
5. 選擇 Create (建立)

使用 Quick Setup 建立 AWS Config組態記錄器

使用 中的Quick Setup工具 AWS Systems Manager，您可以快速建立採用 的組態記錄器 AWS Config。使用組態記錄器來偵測您資源組態中的變更，並將這些變更作為組態項目擷取。如果您不熟悉 AWS Config，建議您先檢閱 AWS Config 開發人員指南中的內容，再使用 建立組態，以進一步了解 服務Quick Setup。如需詳細資訊 AWS Config，請參閱《 AWS Config 開發人員指南》中的什麼是 AWS Config？。

根據預設，組態記錄器會在 AWS Config 執行 AWS 區域 的 中記錄所有支援的資源。您可以自訂組態，以便僅記錄您指定的資源類型。如需詳細資訊，請參閱《 AWS Config 開發人員指南》中的選取哪些資源 AWS Config 記錄。

AWS Config 開始記錄組態時，需支付服務費用。如需定價資訊，請參閱 AWS Config 定價。

> **Note**
>
> 如果您已建立組態記錄器，則 Quick Setup 不會停止記錄或對已在記錄的資源類型進行任何變更。如果您選擇使用 Quick Setup 記錄其他資源類型，服務會將它們附加到現有的記錄器群組。刪除 Quick Setup Config 紀錄組態類型不會停止組態記錄器。變更會繼續進行記錄，並且會收取服務使用費，直到您停止組態記錄器為止。若要進一步了解管理組態記錄器，請參閱《AWS Config 開發人員指南》中的管理組態記錄器。

若要設定 AWS Config 錄音，請在 AWS Systems Manager 主控台中執行下列任務。

使用 設定 AWS Config 錄音 Quick Setup

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Quick Setup。
3. 在設定記錄卡中，選擇建立。

   > **Tip**
   >
   > 如果您的帳戶中已有一或多個組態，請先選擇程式庫索引標籤或組態區段中的建立按鈕，以檢視卡。

4. 在組態選項區段中，執行下列動作：

   a. 對於選擇要記錄 AWS 的資源類型，請指定要記錄所有支援的資源還是只記錄您選擇的資源類型。

   b. 在交付設定欄位中，指定是建立新的 Amazon Simple Storage Service (Amazon S3) 儲存貯體，還是選擇要向其傳送組態快照的現有儲存貯體。

   c. 針對通知選項，選擇您偏好的通知選項。 AWS Config 使用 Amazon Simple Notification Service (Amazon SNS) 來通知您與資源相關的重要 AWS Config 事件。如果您選擇使用現有的 SNS 主題選項，則必須提供您要使用該帳戶中現有 Amazon SNS 主題的 AWS 帳戶 ID 和名稱。如果您以多個 AWS 區域為目標，則每個區域中的主題名稱必須相同。

5. 在 Schedule (排程) 區段中，選擇您希望 Quick Setup 修補對與您的組態不同的資源所做的變更的頻率。Default (預設) 選項會執行一次。如果您不要 Quick Setup 修復對與組態不同的資源所做的變更，請選擇 Custom (自訂) 下的 Disable remediation (停用修補)。

6. 在目標區段中，選擇下列其中一個選項，以識別用於記錄的帳戶和區域。

> **Note**
>
> 如果您使用單一帳戶，則無法使用組織和組織單位 (OU) 的選項。您可以選擇將此組態套用至您帳戶中的所有 AWS 區域 ，還是只套用您選取的區域。

- Entire organization (整個組織) – 組織中的所有帳戶和區域。
- Custom (自訂) – 僅您指定的 OU 與區域。
  - 在目標 OU 區段中，選取您希望允許記錄的 OU。
  - 在目標區域區段中，選取您希望允許記錄的區域。
- Current account (目前帳戶) – 只有您在目前登入的帳戶中指定的區域才會成為目標。選擇下列其中一項：
  - Current Region (目前區域) – 只有在主控台中選取的區域中的受管節點才能成為目標。
  - 選擇區域：選擇要套用記錄組態的個別區域。

7. 選擇 Create (建立)。

使用 部署 AWS Config 一致性套件 Quick Setup

一致性套件是 AWS Config 規則和修補動作的集合。利用 Quick Setup，您可以用單一實體的形式在帳戶和 AWS 區域 中，或是跨 AWS Organizations中的組織部署一致性套件。這可協助您使用通用架構和封裝模型，大規模管理 AWS 資源的組態合規，從政策定義到稽核和彙總報告。

若要部署一致性套件，請在 AWS Systems Manager Quick Setup 主控台中執行下列任務。

> **Note**
>
> 部署此組態之前，您必須先啟用 AWS Config 錄製。如需詳細資訊，請參閱《AWS Config 開發人員指南》中的一致性套件。

若要使用 Quick Setup 部署一致性套件

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Quick Setup。

3. 在一致性套件卡中，選擇建立。

> **Tip**
>
> 如果您的帳戶中已有一或多個組態，請先選擇程式庫索引標籤或組態區段中的建立按鈕，以檢視卡。

4. 在一致性套件區段中，選擇您要部署的一致性套件。
5. 在 Schedule (排程) 區段中，選擇您希望 Quick Setup 修補對與您的組態不同的資源所做的變更的頻率。Default (預設) 選項會執行一次。如果您不要 Quick Setup 修復對與組態不同的資源所做的變更，請選擇 Custom (自訂) 下的 Disabled (已停用)。
6. 在目標區段中，選擇是否要將一致性套件部署到整個組織、部分組織 AWS 區域或您目前登入的帳戶。

   如果您選擇 Entire organization (整個組織)，請繼續步驟 8。

   如果選擇 Custom (自訂)，請繼續步驟 7。
7. 在 Target Regions (目標區域) 區段中，選取您要部署一致性套件的區域的核取方塊。
8. 選擇 Create (建立)。

使用修補程式政策設定組織中執行個體的Quick Setup修補

使用 中的Quick Setup工具 AWS Systems Manager，您可以建立由 提供支援的修補程式政策Patch Manager。修補程式政策定義了在自動化修補 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體和其他受管節點時使用的排程和基準。使用單一修補程式政策組態，您可以定義為組織中多個 AWS 區域 的所有帳戶、僅您選擇的帳戶和區域或者單一帳戶-區域對進行修補。如需有關修補程式政策的詳細資訊，請參閱 Quick Setup中的修補程式政策組態。

先決條件

若要使用 Quick Setup 定義節點的修補程式政策，則節點必須是受管節點。如需有關管理節點的詳細資訊，請參閱 為組織設定 Systems Manager 統一主控台。

> **Important**
>
> 修補程式合規掃描方法：Systems Manager 支援多種掃描受管節點的方法，可檢查修補程式合規情況。如果一次實作多個方法，則您看到的修補程式合規資訊永遠是最近一次掃描的結果。

先前掃描的結果會覆寫。如果掃描方法使用不同的修補基準，且具有不同的核准規則，則修補程式合規資訊可能會意外變更。如需詳細資訊，請參閱避免意外覆寫修補程式合規資料。

關聯合規狀態和修補程式政策：Quick Setup 修補程式政策下受管節點的修補狀態符合該節點的 State Manager 關聯執行狀態。如果關聯執行狀態為 `Compliant`，受管節點的修補程式狀態也會標示為 `Compliant`。如果關聯執行狀態為 `Non-Compliant`，受管節點的修補程式狀態也會標示為 `Non-Compliant`。

修補程式政策組態支援的區域

下列區域目前支援 Quick Setup 中的修補程式政策組態：

- 美國東部 (俄亥俄) (us-east-2)
- 美國東部 (維吉尼亞北部) (us-east-1)
- 美國西部 (加利佛尼亞北部) (us-west-1)
- 美國西部 (奧勒岡) (us-west-2)
- 亞太區域 (孟買) (ap-south-1)
- 亞太區域 (首爾) (ap-northeast-2)
- 亞太區域 (新加坡) (ap-southeast-1)
- 亞太區域 (雪梨) (ap-southeast-2)
- 亞太區域 (東京) (ap-northeast-1)
- 加拿大 (中部) (ca-central-1)
- 歐洲 (法蘭克福) (eu-central-1)
- 歐洲 (愛爾蘭) (eu-west-1)
- 歐洲 (倫敦) (eu-west-2)
- 歐洲 (巴黎) (eu-west-3)
- 歐洲 (斯德哥爾摩) (eu-north-1)
- 南美洲 (聖保羅) (sa-east-1)

適用於修補程式政策 S3 儲存貯體的許可

建立修補程式政策時，Quick Setup 會建立一個包含名為 `baseline_overrides.json` 的檔案的 Amazon S3 儲存貯體。此檔案儲存您為修補程式政策指定之修補基準的相關資訊。

此儲存貯體的名稱格式為 aws-quicksetup-patchpolicy-*account-id*-*quick-setup-configuration-id*。

例如：aws-quicksetup-patchpolicy-123456789012-abcde

如果您為某個組織建立修補程式政策，則此儲存貯體會建立在該組織的管理帳戶中。

您必須向其他 AWS 資源提供使用 AWS Identity and Access Management (IAM) 政策存取此 S3 儲存貯體的許可，有兩種使用案例：

- 情況 1：將您自己的而不是由 Quick Setup 提供的執行個體設定檔或服務角色連接至受管節點
- 情況 2：使用 VPC 端點連線至 Systems Manager

在這兩種情況下您所需要的許可政策都已呈現在下面的 Quick Setup S3 儲存貯體的政策許可 一節中。

情況 1：將您自己的而不是由 Quick Setup 提供的執行個體設定檔或服務角色連接至受管節點

修補程式政策組態包含將必要的 IAM 政策新增至連接至執行個體的現有執行個體設定檔的選項。

如果您未選擇此選項，但想要 Quick Setup 使用此修補程式政策修補受管的節點，則必須實作下列項目：

- IAM 受管政策 AmazonSSMManagedInstanceCore 必須連接至用於為受管節點提供 Systems Manager 許可的 IAM 執行個體設定檔或 IAM 服務角色。
- 您必須將存取修補程式政策儲存貯體的許可作為內嵌政策新增至 IAM 執行個體設定檔或 IAM 服務角色。您可以向所有 aws-quicksetup-patchpolicy 儲存貯體或僅向針對您的組織或帳戶建立的特定儲存貯體提供萬用字元存取權，如先前的程式碼範例所示。
- 您必須使用以下鍵值對標記 IAM 執行個體設定檔或 IAM 服務角色。

  Key: QSConfigId-*quick-setup-configuration-id*, Value: *quick-setup-configuration-id*

  quick-setup-configuration-id 代表套用至 AWS CloudFormation 堆疊的參數值，用於建立修補程式政策組態。若要擷取此 ID，請執行以下操作：

  1. 在 https：//https://console.aws.amazon.com/cloudformation 開啟 AWS CloudFormation 主控台。
  2. 選取用來建立修補程式政策的堆疊名稱。名稱的格式為 StackSet-AWS-QuickSetup-PatchPolicy-LA-q4bkg-52cd2f06-d0f9-499e-9818-d887cEXAMPLE。

3. 選擇參數索引標籤。
4. 在參數清單的鍵欄中，找到鍵 QSConfigurationId。在同一列的值欄中，找到組態 ID，例如 `abcde`。

   在此範例中，對於要套用至執行個體設定檔或服務角色的標籤，其鍵為 `QSConfigId-abcde`，且值為 `abcde`。

如需有關將標籤新增至 IAM 角色的資訊，請參閱《IAM 使用者指南》中的標記 IAM 角色和管理執行個體描述檔 (AWS CLI 或 AWS API) 上的標籤。

情況 2：使用 VPC 端點連線至 Systems Manager

如果您使用 VPC 端點連線至 Systems Manager，則您的 S3 VPC 端點政策必須允許存取 Quick Setup 修補程式政策 S3 儲存貯體。

如需將許可新增至 S3 VPC 端點政策的相關資訊，請參閱《Amazon S3 使用者指南》中的使用儲存貯體政策控制來自 VPC 端點的存取一節。

Quick Setup S3 儲存貯體的政策許可

您可以向所有 `aws-quicksetup-patchpolicy` 儲存貯體或僅向針對您的組織或帳戶建立的特定儲存貯體提供萬用字元存取權。若要為下述兩種情況提供必要的許可，請使用其中一種格式。

All patch policy buckets

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccessToAllPatchPolicyRelatedBuckets",
      "Effect": "Allow",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::aws-quicksetup-patchpolicy-*"
    }
  ]
}
```

Specific patch policy bucket

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccessToMyPatchPolicyRelatedBucket",
      "Effect": "Allow",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::aws-quicksetup-patchpolicy-account-id-quick-setup-configuration-
id"[1]
    }
  ]
}
```

[1] 建立修補程式政策組態後，您可以在 S3 主控台中找到儲存貯體的完整名稱。例如：aws-quicksetup-patchpolicy-123456789012-abcde

修補程式政策作業中的隨機修補基準 ID

修補程式政策的修補作業會使用 AWS-RunPatchBaseline SSM 命令文件中的 BaselineOverride 參數。

當您使用 AWS-RunPatchBaseline 進行修補程式政策以外的修補時，您可以使用 BaselineOverride 來指定作業期間要使用的與指定的預設值不同的修補基準清單。您可以在名為 baseline_overrides.json 的檔案中建立此清單，然後手動將其新增至您擁有的一個 Amazon S3 儲存貯體，如 使用 BaselineOverride 參數 中所述。

然而，對於以修補程式政策為基礎的修補作業，Systems Manager 會自動建立 S3 儲存貯體並在其中新增 baseline_overrides.json 檔案。然後，每次Quick Setup執行修補操作時 （ 使用 Run Command工具時，系統會為每個修補基準產生隨機 ID。每個修補程式政策修補作業的這個 ID 都不同，而且您無法在您的帳戶中儲存或存取它所代表的修補基準。

因此，您不會在修補日誌中看到組態中選取的修補基準的 ID。這適用於您可能選取的 AWS 受管修補程式基準和自訂修補程式基準。日誌中報告的基準 ID 是針對該特定修補作業產生的基準 ID。

此外，如果您嘗試在 Patch Manager 中檢視有關使用隨機 ID 產生之修補基準的詳細資訊，則系統會報告該修補基準不存在。這是預期會出現的行為，可以忽略。

建立修補程式政策

若要建立修補程式政策，請在 Systems Manager 主控台執行下列任務。

使用 Quick Setup 建立修補程式政策

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。

   如果您要設定為組織進行修補，請確保您已登入到組織的管理帳戶。您無法使用委派管理員帳戶或成員帳戶來設定政策。

2. 在導覽窗格中，選擇 Quick Setup。

3. 在 Patch Manager (修補程式管理員) 卡上，選擇 Create (建立)。

   > **Tip**
   >
   > 如果您的帳戶中已有一或多個組態，請先選擇程式庫索引標籤或組態區段中的建立按鈕，以檢視卡。

4. 對於 Configuration name (組態名稱)，請輸入名稱以協助識別修補程式政策。

5. 在 Scanning and installation (掃描和安裝) 區段的 Patch operation (修補程式操作) 下，選擇修補程式政策是要 Scan (掃描) 指定的目標，還是在指定目標上 Scan and install (掃描並安裝) 修補程式。

6. 在 Scanning schedule (掃描排程) 下，選擇 Use recommended defaults (使用建議的預設值) 或 Custom scan schedule (自訂掃描排程)。預設的掃描排程每天會在凌晨 1 點 (UTC) 掃描您的目標。

   - 如果您選擇 Custom scan schedule (自訂掃描排程)，則請選取 Scanning frequency (掃描頻率)。
   - 如果您選擇 Daily (每日)，則請輸入您要掃描目標的時間 (UTC)。
   - 如果您選擇 Custom CRON Expression (自訂 CRON 運算式)，則請將排程輸入為 CRON expression (CRON 運算式)。如需有關格式化 Systems Manager 的 CRON 運算式的詳細資訊，請參閱 參考：Systems Manager 的 Cron 和 Rate 運算式。

     此外，請選取 Wait to scan targets until first CRON interval (等待掃描目標，直到第一個 CRON 間隔為止)。依預設，Patch Manager 會在節點變為目標時立即進行掃描。

7. 如果您選擇 Scan and install (掃描並安裝)，則請選擇將修補程式安裝到指定目標時要使用的 Installation schedule (安裝排程)。如果您選擇 Use recommended defaults (使用建議的預設值)，則 Patch Manager 每週會在星期日凌晨 2 點 (UTC) 安裝修補程式。

- 如果您選擇 Custom install schedule (自訂安裝排程)，請選取 Installation Frequency (安裝頻率)。
- 如果您選擇 Daily (每日)，請輸入您要在目標上安裝更新的時間 (UTC)。
- 如果您選擇 Custom CRON expression (自訂 CRON 運算式)，請將排程輸入為 CRON expression (CRON 運算式)。如需有關格式化 Systems Manager 的 CRON 運算式的詳細資訊，請參閱 參考：Systems Manager 的 Cron 和 Rate 運算式。

  此外，請清除 Wait to install updates until first CRON interval (等待安裝更新，直到第一個 CRON 間隔)，以便在節點變為目標時立即在節點上安裝更新。依預設，Patch Manager 會等到第一個 CRON 間隔才能安裝更新。
- 請選擇 Reboot if needed (必要時重新啟動)，在安裝修補程式之後重新啟動節點 建議在安裝後重新啟動，但可能會導致可用性問題。

8. 在 Patch baseline (修補基準) 中，選擇掃描和更新目標時要使用的修補基準。

   依預設，Patch Manager 會使用預先定義的修補基準。如需詳細資訊，請參閱預先定義的基準。

   如果您選擇自訂修補程式基準，請變更您不想使用預先定義修補程式基準之作業系統的所選 AWS 修補程式基準。

> **Note**
>
> 如果您使用 VPC 端點連線至 Systems Manager，請確保您的 S3 VPC 端點政策允許存取該 S3 儲存貯體。如需詳細資訊，請參閱適用於修補程式政策 S3 儲存貯體的許可。

> **Important**
>
> 如果您在 Quick Setup 中使用修補程式政策組態，則您對自訂修補基準所做的更新會每小時與 Quick Setup 同步一次。
> 如果刪除修補程式政策中參照的自訂修補基準，則修補程式政策的 Quick Setup Configuration details (組態詳細資訊) 頁面上會顯示橫幅。此橫幅會通知您修補程式政策參照修補基準不再存在，而後續的修補操作將會失敗。在此情況下，請返回到 Quick Setup Configurations (組態) 頁面，選取 Patch Manager 組態，然後選擇 Actions (動作)、Edit configuration (編輯組態)。刪除的修補基準名稱會反白顯示，您必須為受影響的作業系統選取新的修補基準。

9. (選用) 在 Patching log storage (修補日誌存放區) 區段中，選取 Write output to S3 bucket (將輸出寫入 S3 儲存貯體) 以將修補程式操作日誌存放在 Amazon S3 儲存貯體中。

> **Note**
>
> 如果您要為組織設定修補程式政策，則組織的管理帳戶必須至少擁有此儲存貯體的唯讀許可。政策中包含的所有組織單位都必須擁有該儲存貯體的寫入存取權。如需有關授予儲存貯體存取權給不同帳戶的資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的範例 2：儲存貯體擁有者授予跨帳戶儲存貯體許可。

10. 選擇瀏覽 S3 以選取要儲存修補程式日誌輸出的儲存貯體。管理帳戶必須擁有此儲存貯體的讀取存取權。在 Targets (目標) 區段中設定的所有非管理帳戶和目標都必須擁有對所提供的 S3 儲存貯體的寫入存取權以進行記錄。
11. 在 Targets (目標) 區段中，選擇下列其中一個選項，以識別此修補程式政策操作的帳戶和區域。

> **Note**
>
> 如果您使用單一帳戶，則無法使用組織和組織單位 (OU) 的選項。您可以選擇是要將此組態套用至帳戶中的所有 AWS 區域 ，還是僅套用至您選擇的區域。
> 如果先前為帳戶指定了主區域，但尚未加入 Quick Setup 的新主控台體驗，則無法將該區域排除在目標組態之外。

- Entire organization (整個組織) – 組織中的所有帳戶和區域。
- Custom (自訂) – 僅您指定的 OU 與區域。
  - 在 Target OUs (目標 OU) 區段中，選取您要設定修補程式政策的 OU。
  - 在 Target Regions (目標區域) 區段中，選取您要套用修補程式政策的區域。
- Current account (目前帳戶) – 只有您在目前登入的帳戶中指定的區域才會成為目標。選擇下列其中一項：
  - Current Region (目前區域) – 只有在主控台中選取的區域中的受管節點才能成為目標。
  - Choose Regions (選擇區域) – 選擇要套用修補程式政策的個別區域。

12. 對於 Choose how you want to target instances (選擇將執行個體設為目標的方式)，選擇下列其中一個選項：

- All managed nodes (所有受管節點) – 所選 OU 與區域中的所有受管節點。

- Specify the resource group (指定資源群組) – 從清單中選擇資源群組的名稱，將其關聯資源設為目標。

> **Note**
>
> 目前，只有單一帳號組態支援選取資源群組。若要修補多個帳戶中的資源，請選擇不同的鎖定目標選項。

- Specify a node tag (指定節點標籤) – 僅在您將其設為目標的所有帳戶和區域中修補標記有您指定之金鑰值對的節點。
- Manual (手動) – 從清單中手動選擇所有指定帳號和區域中的受管節點。

> **Note**
>
> 此選項目前僅支援 Amazon EC2 執行個體。您可以在修補程式政策組態中手動新增最多 25 個執行個體。

13. 在 Rate control (速率控制) 區段中，執行下列操作：

- 針對 Concurrency (並行)，輸入要同時執行修補程式政策的節點數目或百分比。
- 針對 Error threshold (錯誤閾值)，輸入修補程式政策失敗之前可能會發生錯誤的節點數目或百分比。

14. (選用) 選取將必要的 IAM 政策新增至連接至執行個體的現有執行個體設定檔核取方塊。

此選項會將此 Quick Setup 組態建立的 IAM 政策套用至已連接執行個體設定檔 (EC2 執行個體) 或服務角色的節點 (啟用混合模式節點)。當您的受管節點已連接執行個體設定檔或服務角色時，建議選取此選項，但其不包含使用 Systems Manager 所需的所有許可。

您在此選取的項目會套用至稍後在帳戶和區域 (其會套用此修補程式政策組態) 中建立的受管節點。

> **Important**
>
> 如果您未選取此核取方塊，但想要 Quick Setup 使用此修補程式政策修補受管節點，則必須執行下列動作：
> 將存取為修補程式政策建立之 S3 儲存貯體的許可新增至 IAM 執行個體設定檔或 IAM 服務角色
> 使用特定的鍵值對標記您的 IAM 執行個體設定檔或 IAM 服務角色。

如需相關資訊，請參閱情況 1：將您自己的而不是由 Quick Setup 提供的執行個體設定檔或服務角色連接至受管節點。

15. 選擇建立。

若要在建立修補程式政策之後檢閱修補狀態，您可以從 Quick Setup 頁面存取組態。

使用 Quick Setup 設定 DevOps Guru

您可以使用 Quick Setup 快速設定 DevOps Guru 選項。Amazon DevOps Guru 是一項受服務支援的機器學習 (ML)，可讓您輕鬆地改善應用程式的操作效能和可用性。DevOps Guru 會偵測到與正常操作模式不同的行為，因此您可以在這些問題影響客戶之前找出操作問題。DevOps Guru 會自動從您的 AWS 應用程式擷取操作資料，並提供單一儀表板，以視覺化方式呈現操作資料中的問題。您可以開始使用 DevOps Guru 來改善應用程式的可用性和可靠性，而無需手動設定或機器學習專業知識。

使用 Quick Setup 設定 DevOps Guru 在以下 AWS 區域中可用：

- 美國東部 (維吉尼亞北部)
- 美國東部 (俄亥俄)
- 美國西部 (奧勒岡)
- 歐洲 (法蘭克福)
- 歐洲 (愛爾蘭)
- 歐洲 (斯德哥爾摩)
- 亞太區域 (新加坡)
- 亞太區域 (雪梨)
- 亞太區域 (東京)

如需定價詳細資訊，請參閱 Amazon DevOps Guru 定價。

若要設定 DevOpsGuru，請在 AWS Systems Manager Quick Setup 主控台中執行下列任務。

若要使用 Quick Setup 設定 DevOps Guru

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Quick Setup。

3. 在 DevOps Guru 卡中，選擇建立。

   > **Tip**
   >
   > 如果您的帳戶中已有一或多個組態，請先選擇程式庫索引標籤或組態區段中的建立按鈕，以檢視卡。

4. 在 Configuration options (組態選項) 區段中，選擇您要分析的 AWS 資源類型和您的通知偏好設定。

   如果您未選取分析組織中的所有帳戶中的所有 AWS 資源選項，您可以在 DevOpsGuru 主控台中選擇要 AWS 稍後分析的資源。DevOps Guru 會分析不同的 AWS 資源類型 (例如 Amazon Simple Storage Service (Amazon S3) 儲存貯體和 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體)，可分為兩個定價群組。您需針對每個作用中的資源支付分析的 AWS 資源小時數的費用。只有在一小時內產生指標、事件或日誌項目時，資源才會處於作用中狀態。針對特定 AWS 資源類型收取的費率取決於價格群組。

   如果您選取啟用 SNS 通知選項，則會在您設定的目標組織單位 (OUs) 中的每個 AWS 帳戶 中建立 Amazon Simple Notification Service (Amazon SNS) 主題。DevOps Guru 會使用此主題通知您重要的 DevOps Guru 事件，例如建立新的洞察。如果您未啟用此選項，您可以稍後在 DevOps Guru 主控台中新增主題。

   如果您選取 Enable AWS Systems Manager OpsItems 選項，則會為相關的 Amazon EventBridge 事件和 Amazon CloudWatch 警示建立操作工作項目 (OpsItems)。

5. 在 Schedule (排程) 區段中，選擇您希望 Quick Setup 修補對與您的組態不同的資源所做的變更的頻率。Default (預設) 選項會執行一次。如果您不要 Quick Setup 修復對與組態不同的資源所做的變更，請選擇 Custom (自訂) 下的 Disabled (已停用)。

6. 在 Targets (目標) 區段中，選擇是否允許 DevOps Guru 分析部分組織單位 (OU) 中的資源，或是您目前登入的帳戶。

   如果選擇 Custom (自訂)，請繼續步驟 8。

   如果選擇 Current account (目前帳戶)，請繼續步驟 9。

7. 在 Target OUs (目標 OU) 和 Target Regions (目標區域) 區段中，選取您要使用 DevOps Guru 的 OU 和區域的核取方塊。

8. 選擇您要在目前帳戶中使用 DevOps Guru 的區域。

9. 選擇 Create (建立)。

使用 Quick Setup 部署 Distributor 套件

Distributor 是 中的工具 AWS Systems Manager。Distributor 套件是可安裝型軟體或可部署為單一實體的資產的集合。使用 Quick Setup，您可以在 和 中部署Distributor套件 AWS 帳戶 ， AWS 區域 或在中跨組織部署套件 AWS Organizations。目前，只有 EC2Launch v2 代理程式、Amazon Elastic File System (Amazon EFS) 公用程式套件和 Amazon CloudWatch 代理程式可以使用 Quick Setup 部署。如需有關 Distributor 的詳細資訊，請參閱「AWS Systems Manager Distributor」。

若要部署Distributor套件，請在 AWS Systems Manager Quick Setup 主控台中執行下列任務。

若要使用 Quick Setup 部署 Distributor 套件

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Quick Setup。
3. 在分發者卡上，選擇建立。

   > **Tip**
   >
   > 如果您的帳戶中已有一或多個組態，請先選擇程式庫索引標籤或組態區段中的建立按鈕，以檢視卡。

4. 在 Configuration options (組態選項) 區段中，選擇您要部署的套件。
5. 在 Targets (目標) 區段中，選擇是否將套件部署至您的整個組織、某些組織單位 (OU) 或您目前登入的帳戶。

   如果您選擇 Entire organization (整個組織)，請繼續步驟 8。

   如果選擇 Custom (自訂)，請繼續步驟 7。
6. 在 Target OUs (目標 OU) 區段中，選取您要部署套件的 OU 和區域的核取方塊。
7. 選擇 Create (建立)。

使用Quick Setup依照排程自動停止和啟動 EC2 執行個體

使用 中的Quick Setup工具 AWS Systems Manager，您可以設定 Resource Scheduler 來自動啟動和停止 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。

此 Quick Setup 組態可根據您指定的排程啟動和停用執行個體，從而協助您降低營運成本。此工具可協助您避免在不需要執行個體時，對執行執行個體產生不必要的成本。

例如，您目前可能會讓執行個體持續執行，即使每天僅使用 10 小時，每週僅 5 天。相反，您可以對執行個體進行排程，每天在工作時間後停用。因此，這些執行個體可以節省 70% 的費用，因為執行時間從 168 小時縮短為 50 小時。使用 Quick Setup 無需付費。但是，根據您設定的資源以及使用限制可能會產生費用，而對用於設定服務的服務不會收取任何費用。

使用資源排程器，您可以選擇 AWS 帳戶 根據您定義的排程，自動停止和啟動多個 AWS 區域 之間的執行個體。Quick Setup 組態會使用您指定的標籤索引鍵和值來鎖定 Amazon EC2 執行個體。資源排程器只會停用或啟動其標籤符合您在組態中指定值的執行個體。請注意，如果連接至執行個體的 Amazon EBS 磁碟區已加密，您必須將 AWS KMS 金鑰所需的許可新增至 資源排程器的 IAM 角色，才能啟動執行個體。

每個組態的執行個體上限

個別組態支援每個區域排程最多 5,000 個執行個體。如果您的案例需要在指定區域中排程 5,000 個以上的執行個體，則您必須建立多個組態。相應地標記您的執行個體，讓每個組態最多可管理 5,000 個執行個體。建立多個資源排程器 Quick Setup 組態時，您必須指定不同的標籤金鑰值。例如，一個組態可以使用值`Environment`為 的標籤金鑰`Production`，而另一個組態則使用 `Environment`和 `Development`。

排程行為

下列幾點說明排程組態的特定行為：

- 只有當標記的執行個體處於 `Stopped` 狀態時，資源排程器才會啟動執行個體。同樣，只有在執行個體處於 `running` 狀態時，才會停用執行個體。資源排程器會在事件驅動模型上運作，並且只會在您指定的時間啟動或停用執行個體。例如，您可以建立在上午 9 點啟動執行個體的排程。資源排程器會啟動與您指定之標籤相關聯的所有執行個體，其在上午 9 點處於 `Stopped` 狀態 如果稍後手動停用執行個體，則資源排程器將不會再次啟動它們以維持 `Running` 狀態。同樣，如果根據排程在停用執行個體後手動啟動它，則資源排程器將不會再次停用執行個體。
- 如果您建立的排程開始時間晚於停止時間的 24 小時，Resource Scheduler 會假設您的執行個體在夜間執行。例如，您可以建立在晚上 9 點啟動執行個體，在早晨 7 點停用執行個體的排程。資源排程器會啟動與您指定之標籤相關聯的所有執行個體 (其在晚上 9 點處於 `Stopped` 狀態) 並在次日早晨 7 點停用。對於隔夜排程，開始時間適用於您為排程選取的天數。但是，停止時間適用於排程中的次日。
- 當您建立排程組態時，執行個體的目前狀態可能會變更為符合排程的需求。

  例如，假設今天是星期三，而您指定受管執行個體的排程從上午 9 點開始，並在僅限星期二和星期四的下午 5 點停止。由於您目前的時間在執行個體執行的指定時間之外，因此它們會在建立組態後停止。執行個體在下一個指定的小時前不會再次執行，即星期四上午 9 點。

如果您的執行個體目前處於 Stopped 狀態，且您指定了要在目前時間執行的排程，則資源排程器會在建立組態後啟動它們。

如果您刪除組態，則執行個體將不會再根據先前定義的排程停用和啟動。在極少數情況下，執行個體可能會因為 API 操作失敗而無法成功停用或啟動。

若要設定 Amazon EC2 執行個體的排程，請在 AWS Systems Manager Quick Setup主控台中執行下列任務。

使用 Quick Setup 設定執行個體排程

1. 開啟 AWS Systems Manager 主控台，網址為 https：//https://console.aws.amazon.com/systems-manager/。
2. 在導覽窗格中，選擇 Quick Setup。
3. 在資源排程器卡中，選擇建立。

   > **Tip**
   >
   > 如果您的帳戶中已有一或多個組態，請先選擇程式庫索引標籤或組態區段中的建立按鈕，以檢視卡。

4. 在 Instance tag (執行個體標籤) 區段中，指定套用至要與執行個體關聯之執行個體的標籤金鑰和值。
5. 在 Schedule options (排程選項) 區段中，指定要啟動和停用執行個體的時區、天數和時間。
6. 在 Targets (目標) 區段中，選擇是否為您的組織單位 (OU) 的 Custom (自訂) 群組設定排程，還是為您登入的 Current account (當前帳戶) 設定排程：

   - Custom (自訂) – 在 Target OUs (目標 OU) 區段中，選取您要設定排程的 OU。接著，在 Target Regions (目標) 區段中，選取您要設定排程的區域。
   - 目前帳戶 – 選取 Current Region (目前區域) 或 Choose Regions (選擇區域)。如果已選取 Choose Regions (選擇區域)，請選擇您要設定排程的 Target Regions (目標區域)。
7. 驗證 Summary (摘要) 區段中的排程資訊。
8. 選擇建立。

AWS 資源總管 使用 設定 Quick Setup

使用 中的Quick Setup工具 AWS Systems Manager，您可以快速設定 AWS 資源總管 來搜尋和探索 中 AWS 帳戶 或整個 AWS 組織中的資源。您可以使用名稱、標籤和 ID 等中繼資料來搜尋資源。 AWS 資源總管 會使用索引來快速回應搜尋查詢。Resource Explorer 可使用各種資料來源建立並維護索引，以便收集 AWS 帳戶中資源的相關資訊。

Resource Explorer 的 Quick Setup 可自動執行索引組態程序。如需詳細資訊 AWS 資源總管，請參閱 AWS 資源總管 《 使用者指南》中的什麼是 AWS 資源總管？。

在使用 Quick Setup 期間，Resource Explorer 會執行下列動作：

- 在 的每個 AWS 區域 中建立索引 AWS 帳戶。
- 更新區域中指定為帳戶彙總工具索引的索引。
- 在彙總工具索引區域中建立預設檢視。此檢視沒有篩選條件，因此會傳回在索引中發現的所有資源。

最低許可

若要執行下列程序中的步驟，必須擁有下列許可：

- 動作：`resource-explorer-2:*`，資源：沒有具體資源 (*)
- 動作：`iam:CreateServiceLinkedRole`，資源：沒有具體資源 (*)

設定 Resource Explorer

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Quick Setup。
3. 在 Resource Explorer 卡中，選擇建立。
4. 在彙總工具索引區域區段中，選擇要包含彙總工具索引的區域。應該選取適合使用者地理位置的區域。
5. (選用) 選取取代上方所選區域以外的區域中現有的彙總工具索引核取方塊。
6. 在目標區段中，選擇包含了要探索資源的目標組織或特定組織單位 (OU)。
7. 在區域區段中，選擇要在組態中包含的區域。
8. 檢閱組態摘要，然後選擇建立。

在 Resource Explorer 頁面上，您可以監控組態狀態。

針對 Quick Setup 結果進行疑難排解

使用下列資訊來協助您疑難排解 中的Quick Setup工具 的問題 AWS Systems Manager。本主題包含根據 Quick Setup 問題類型解決問題的具體任務。

問題：部署失敗

如果 CloudFormation 堆疊集在建立過程中失敗，則部署將失敗。使用以下步驟調查部署失敗的原因。

1. 導覽至 AWS CloudFormation 主控台。
2. 選擇 Quick Setup 組態建立的堆疊。Stack name (堆疊名稱) 包括 `QuickSetup`，其後接著您選擇的組態類型，例如 `SSMHostMgmt`。

   > **Note**
   >
   > CloudFormation 有時會刪除失敗的堆疊部署。如果堆疊在 Stacks (堆疊) 資料表中不可用，請從篩選條件清單中選擇 Deleted (已刪除)。

3. 檢視 Status (狀態) 和 Status reason (狀態原因)。如需堆疊狀態的詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的堆疊狀態碼。
4. 要想了解失敗的確切步驟，請在 Events (事件) 標籤中檢視每個事件的 Status (狀態)。
5. 檢閱《AWS CloudFormation 使用者指南》中的疑難排解。
6. 如果無法使用 CloudFormation 疑難排解步驟解決部署失敗問題，請刪除組態後重新設定。

問題：關聯失敗

如果任何關聯在設定過程中失敗，Configuration details (組態詳細資訊) 頁面上的 Configuration details (組態詳細資訊) 資料表會顯示 Configuration status (組態狀態) 為 Failed (失敗)。使用以下步驟對失敗關聯進行疑難排解。

1. 在 Configuration details (組態詳細資訊) 資料表中選擇失敗組態，再選擇 View Details (檢視詳細資訊)。
2. 複製 Association name (關聯名稱)。
3. 導覽至 State Manager，然後將關聯名稱貼到搜尋欄位中。

4. 依序選擇關聯與 Execution history (執行歷史記錄) 標籤。
5. 在 Execution ID (執行 ID) 下方，選擇失敗的關聯執行。
6. Association execution targets (關聯執行目標) 頁面會列出執行關聯的所有節點。選擇執行失敗的 Output (輸出) 按鈕。
7. 在 Output (輸出) 頁面中，選擇 Step - Output (步驟 - 輸出) 以檢視命令執行中該步驟的錯誤訊息。每個步驟都可以顯示不同的錯誤訊息。請檢閱所有步驟的錯誤訊息，以協助對問題進行故障診段。

如果檢視步驟輸出無法解決問題，則可嘗試重新建立關聯。若要重新建立關聯，請先刪除 State Manager 中的失敗關聯。刪除關聯後，編輯組態並選擇刪除的選項，然後選擇 Update (更新)。

> **Note**
>
> 若要調查 Organization (組織) 組態的 Failed (失敗) 關聯，您必須登入含失敗關聯的帳戶，然後如先前所述使用下列失敗關聯程序。從管理帳戶檢視結果時，關聯 ID 不是目標帳戶的超連結。

問題：漂移狀態

在檢視組態的詳細資訊頁面時，您可以檢視每個部署的偏離狀態。只要使用者對服務或功能所做的變更會與透過 Quick Setup 所做的選擇產生衝突，就會發生組態偏離。如果關聯在初始組態之後發生了變更，資料表會顯示一個警告圖示，指示偏離項目的數量。將游標停留在圖示上，即可確認偏離的原因。

在 State Manager 中刪除關聯時，相關部署會顯示偏離警告。要想解決此問題，請編輯組態，然後選擇刪除關聯時移除的選項。選擇 Update (更新)，然後等待部署完成。

AWS Systems Manager 應用程式工具

應用程式工具是一套功能，可協助您管理在 AWS中執行的應用程式。

主題

- AWS AppConfig
- AWS Systems Manager Application Manager
- AWS Systems Manager Parameter Store

AWS AppConfig

有關 的資訊 AWS AppConfig 已移至單獨的指南。如需詳細資訊，請參閱下列內容：

- AWS AppConfig 使用者指南
- AWS AppConfig API 參考

AWS Systems Manager Application Manager

Application Manager是 中的工具 AWS Systems Manager，可協助 DevOps 工程師在其應用程式和叢集的內容中調查和修復其 AWS 資源的問題。 會將多個 AWS 服務 和 Systems Manager 工具的操作資訊Application Manager彙總到單一 AWS Management Console。

在 中Application Manager，應用程式是您想要做為單位運作的邏輯 AWS 資源群組。此邏輯群組可以代表應用程式的不同版本、運算子的擁有權邊界或開發人員環境等等。Application Manager 可支援容器叢集，包括 Amazon Elastic Kubernetes Service (Amazon EKS) 和 Amazon Elastic Container Service (Amazon ECS) 叢集。

第一次開啟 時Application Manager，會顯示Application Manager可為您執行的操作頁面。當您選擇開始使用時， Application Manager會自動匯入在其他 AWS 服務 或 Systems Manager 工具中建立之資源的中繼資料。 Application Manager 接著 會在依預先定義類別分組的清單中顯示這些資源。

對於 Applications (應用程式)，清單包含下列項目：

- AWS CloudFormation 堆疊
- 自訂應用程式
- AWS Launch Wizard 應用程式
- AppRegistry 應用程式
- AWS SAP Enterprise 工作負載應用程式
- Amazon ECS 叢集
- Amazon EKS 叢集

在您設定和設定 AWS 服務 和 Systems Manager 工具之後， Application Manager會顯示下列有關 資源的資訊類型：

- 應用程式中 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體的目前狀態、狀態和 Amazon EC2 Auto Scaling 運作狀態相關資訊

- Amazon CloudWatch 提供的警示
- AWS Config 和 State Manager(Systems Manager 的元件） 提供的合規資訊
- Amazon EKS 提供的 Kubernetes 叢集資訊
- AWS CloudTrail 和 Amazon CloudWatch Logs 提供的日誌資料
- Systems Manager OpsCenter 提供的 OpsItems
- 託管資源 AWS 服務 的 所提供的資源詳細資訊。
- Amazon ECS 提供的容器叢集資訊。

若要協助您修正元件或資源的問題，Application Manager 還會提供可與應用程式建立關聯的 Runbook。若要開始使用 Application Manager，請開啟 Systems Manager 主控台。在導覽窗格中，選擇 Application Manager。

使用 Application Manager 有哪些優點？

Application Manager 減少 DevOps 工程師偵測和調查 AWS 資源問題所需的時間。若要這樣做， 會在一個主控台中Application Manager顯示應用程式內容中的許多操作資訊類型。 Application Manager也提供執行 AWS 常見資源修復任務的 Runbook，以減少修復問題所需的時間。

Application Manager 有哪些功能？

Application Manager 包含下列功能：

- 自動匯入您的 AWS 資源

  在初始設定期間，您可以選擇讓 AWS 帳戶 自動Application Manager匯入和顯示以 CloudFormation 堆疊、 AWS Resource Groups、啟動精靈部署、AppRegistry 應用程式，以及 Amazon ECS 和 Amazon EKS 叢集為基礎的 資源。系統會在預先定義的應用程式或叢集類別中顯示這些資源。之後，每當這些類型的新資源新增至您的 時 AWS 帳戶 ， Application Manager會自動在預先定義的應用程式和叢集類別中顯示新資源。

- 建立或編輯 CloudFormation 堆疊和範本

  Application Manager 可與 CloudFormation 整合，進而協助您佈建和管理應用程式的資源。您可以在 中建立、編輯和刪除 AWS CloudFormation 範本和堆疊Application Manager。 Application Manager也包含範本程式庫，您可以在其中複製、建立和存放範本。 Application Manager和 CloudFormation 會顯示堆疊目前狀態的相同資訊。範本和範本更新會存放在 Systems Manager 中，直到您佈建堆疊為止，此時變更也會顯示在 CloudFormation 中。

- 在應用程式環境中檢視執行個體相關資訊

  Application Manager 與 Amazon Elastic Compute Cloud (Amazon EC2) 整合，以便在應用程式環境中顯示執行個體的相關資訊。Application Manager 以圖形格式顯示所選應用程式的執行個體狀態、狀態和 Amazon EC2 Auto Scaling 運作狀態。Instances (執行個體) 標籤也包含一個資料表，其中包含應用程式中每個執行個體的下列資訊。

  - 執行個體狀態 (待定、停止中、執行中、已停止)
  - SSM Agent 的 Ping 狀態
  - 在執行個體上處理的最新 Systems Manager Automation 執行手冊的狀態和名稱
  - 每個狀態的 Amazon CloudWatch Logs 警示計數。
    - `ALARM` – 指標或表達式在定義的閾值外。
    - `OK` – 指標或表達式在定義的閾值內。
    - `INSUFFICIENT_DATA` – 警示剛開始無法使用指標，或資料不足無法讓指標判斷警示狀態。
  - 父群組和個別自動擴展群組的 Auto Scaling 群組運作狀態
- 檢視應用程式或叢集的操作指標和警示

  Application Manager 可與 Amazon CloudWatch 整合，以提供應用程式或叢集的即時操作指標和警示。您可以深入了解應用程式樹狀結構，以檢視每個元件層級的警示，或檢視個別叢集的警示。
- 檢視應用程式的日誌資料

  Application Manager 可與 Amazon CloudWatch Logs 整合，以在您的應用程式環境中提供日誌資料，而不必離開 Systems Manager。
- 檢視及管理應用程式或叢集的 OpsItems

  Application Manager 可與 AWS Systems Manager OpsCenter 整合，以提供應用程式和叢集的營運工作項目 (OpsItems) 清單。該清單反映了自動產生和手動建立的 OpsItems。您可以檢視有關建立 OpsItem 和 OpsItem 狀態、來源和嚴重性的資源的詳細資訊。
- 檢視應用程式或叢集的資源合規資料

  Application Manager 可與 AWS Config 整合，以根據您指定的規則提供有關 AWS 資源的合規歷史記錄和詳細資訊。Application Manager 還可與 AWS Systems Manager State Manager 整合，提供有關您要為 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體維護的狀態的合規資訊。
- 檢視 Amazon ECS 和 Amazon EKS 叢集基礎設施資訊

  Application Manager 可與 Amazon ECS 和 Amazon EKS 整合，以提供有關叢集基礎設施運作狀態的資訊，以及從集中運算、聯網和儲存資源的元件執行時間檢視。

但是，您無法在 Application Manager 中管理或檢視有關 Amazon EKS Pod 或容器的操作資訊。您只能管理和檢視託管 Amazon EKS 資源之基礎設施的操作資訊。

- 檢視應用程式的資源成本詳細資訊

  Application Manager 透過 AWS 帳單與成本管理成本小工具與 AWS Cost Explorer的功能整合。在帳單和成本管理主控台中啟用 Cost Explorer 後，Application Manager 中的 Cost (成本) 小工具會顯示特定非容器應用程式或應用程式元件的成本資料。您可以使用小工具中的篩選條件，根據長條圖或折線圖中的不同時間週期、粒度和成本類型來檢視成本資料。

- 在單一主控台中檢視詳細的資源資訊

  選擇 Application Manager 中列出的資源名稱，並檢視該資源的情境相關資訊和操作資訊，而不必離開 Systems Manager。

- 接收應用程式的自動資源更新

  如果您對服務主控台中的資源進行變更，且該資源屬於 Application Manager 中的應用程式，則 Systems Manager 會自動顯示這些變更。例如，如果您在 AWS CloudFormation 主控台中更新堆疊，而且該堆疊是Application Manager應用程式的一部分，則堆疊更新會自動反映在 中Application Manager。

- 自動探索 Launch Wizard 應用程式

  Application Manager 已與 AWS Launch Wizard 整合。如果您使用 Launch Wizard 來部署應用程式的資源，Application Manager 可以自動將其匯入並顯示在 Launch Wizard 區段中。

- 使用 CloudWatch Application Insights 在 Application Manager 中監控應用程式資源

  Application Manager 與 Amazon CloudWatch Application Insights 整合。Application Insights 會識別和設定金鑰指標、日誌，並在您所有的應用程式資源和技術堆疊中發出警示。Application Insights 持續監控指標和日誌，以偵測和建立異常及錯誤的關聯。系統偵測到錯誤和異常時，Application Insights 會產生 CloudWatch Events，可用於設定通知或採取動作。您可以在 Application Manager 中的 Overview (概觀) 和 Monitoring (監控) 標籤上啟用和檢視 Application Insights。如需有關 Application Insight 的詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的什麼是 Amazon CloudWatch Application Insights。

- 使用 Runbook 修正問題

  Application Manager 包含預先定義的 Systems Manager Runbook，用於修復 AWS 資源的常見問題。您可以針對應用程式中所有適用資源執行執行手冊，而不必離開 Application Manager。

使用 Application Manager 需要付費嗎？

Application Manager 是免費提供的。

Application Manager 的資源配額是什麼？

您可以在 的 Systems Manager 服務配額中檢視所有 Systems Manager 工具的配額Amazon Web Services 一般參考。除非另有說明，否則每個配額都是區域特定規定。

主題

- 設定相關服務
- 設定的 Systems Manager Application Manager 的許可
- 將應用程式和容器叢集新增至 Application Manager
- 使用 應用程式

設定相關服務

Application Manager是 中的工具 AWS Systems Manager，會顯示來自其他 和 Systems Manager 工具的資源 AWS 服務 和資訊。若要最大化 中顯示的操作資訊量Application Manager，建議您在使用 之前設定和設定這些其他服務或工具Application Manager。

主題

- 設定任務以匯入資源
- 設定任務，以檢視資源的操作資訊

設定任務以匯入資源

下列設定任務可協助您檢視 中的 AWS 資源Application Manager。完成這些任務之後，Systems Manager 可以自動將資源匯入 Application Manager。匯入資源之後，您可以在 Application Manager 中建立應用程式並將您匯入的資源移至其中。這可協助您在應用程式環境中檢視操作資訊。

（選用）使用標籤組織您的 AWS 資源

您可以將中繼資料以標籤的形式指派給 AWS 資源。每個標籤都是由使用者定義的金鑰和值組成的標籤。標籤可協助您管理、識別、組織、搜尋及篩選資源。您可建立標籤，依照用途、擁有者、環境或其他條件分類資源。

（選用）使用 組織您的 AWS 資源 AWS Resource Groups

您可以使用資源群組來組織 AWS 資源。資源群組可讓您更輕鬆地一次性管理、監控及自動化許多資源的任務。

Application Manager 會自動匯入所有資源群組，並將它們列在自訂應用程式類別中。

（選用）使用 設定和部署您的 AWS 資源 AWS CloudFormation

AWS CloudFormation 可讓您以可預測且重複的方式建立和佈建 AWS 基礎設施部署。它可以幫助您使用 AWS 服務 ，例如 Amazon EC2、Amazon Elastic Block Store (Amazon EBS)、Amazon Simple Notification Service (Amazon SNS)、Elastic Load Balancing 和 AWS Auto Scaling。使用 CloudFormation，您可以在雲端中建置可靠、可擴展且符合成本效益的應用程式，而不必擔心建立和設定基礎 AWS 基礎設施。

Application Manager 會自動匯入您的所有 AWS CloudFormation 資源，並在AWS CloudFormation 堆疊類別中列出這些資源。您可以在 Application Manager 中建立 CloudFormation 堆疊。堆疊和範本變更會在 Application Manager 和 CloudFormation 之間自動同步。您也可以在 Application Manager 中建立應用程式並將堆疊移到其中。這樣可協助您在應用程式環境中檢視您堆疊中的資源的操作資訊。如需定價資訊，請參閱 AWS CloudFormation 定價。

（選用）使用 設定和部署您的應用程式 AWS Launch Wizard

啟動精靈會引導您完成為第三方應用程式調整大小、設定和部署 AWS 資源的程序，而不需要手動識別和佈建個別 AWS 資源。

Application Manager 會自動匯入所有的 Launch Wizard 資源，並將它們列在 Launch Wizard 堆疊類別中。如需 的詳細資訊 AWS Launch Wizard，請參閱 SQL Server AWS Launch Wizard 入門。Launch Wizard 是免費提供的。您只需支付為執行解決方案而佈建 AWS 的資源。

(選用) 使用 Amazon ECS 和 Amazon EKS 設定及部署您的容器化應用程式

Amazon Elastic Container Service (Amazon ECS) 是具高可擴展性且快速的容器管理服務，可讓您輕鬆執行、停止和管理叢集上的容器。您可用來在服務中執行個別任務或任務的任務定義中會對您的容器進行定義。

Amazon EKS 是一項受管服務，可協助您在 AWS 上執行 Kubernetes，而無需安裝、操作和維護您自己的 Kubernetes 控制平面或節點。Kubernetes 是一套開放原始碼系統，用於容器化應用程式的自動化部署、擴展與管理。

Application Manager 會自動匯入所有 Amazon ECS 和 Amazon EKS 基礎設施資源，並將它們列在容器叢集標籤上。但是，您無法在 Application Manager 中管理或檢視有關 Amazon EKS Pod 或

容器的操作資訊。您只能管理和檢視託管 Amazon EKS 資源之基礎設施的操作資訊。如需定價資訊，請參閱 Amazon ECS 定價和 Amazon EKS 定價。

設定任務，以檢視資源的操作資訊

下列設定任務可協助您在 Application Manager 中檢視 AWS 資源的操作資訊。

(建議) 驗證 Runbook 許可

您可以使用 Application Manager Systems Manager Automation Runbook 修復 AWS 資源的問題。若要使用此修復工具，您必須設定或驗證許可。如需定價資訊，請參閱 AWS Systems Manager 定價。

(選用) 啟用 Cost Explorer

AWS Cost Explorer 是 的一項功能 AWS Cost Management ，可用來視覺化成本資料以供進一步分析。啟用 Cost Explorer 時，您可以在 Application Manager 主控台中檢視應用程式資源的成本資訊、成本歷史記錄和有關成本最佳化的資訊。

(選用) 安裝和設定 Amazon CloudWatch 日誌與警示

CloudWatch 是一種監控和管理服務，可為 AWS混合和多雲端應用程式和基礎設施資源提供資料和可行的洞察。使用 CloudWatch，您可以從單一平台以日誌和指標形式收集和存取所有效能和營運資料。若要在 Application Manager 中檢視您資源的 CloudWatch 日誌和警示，您必須安裝和設定 CloudWatch。如需定價資訊，請參閱 CloudWatch 定價。

> **Note**
>
> CloudWatch Logs 支援僅適用於應用程式，而不適用於叢集。

(選用) 安裝和設定 AWS Config

AWS Config 提供與您 相關聯的資源的詳細檢視 AWS 帳戶，包括其設定方式、彼此的關係，以及組態及其關係如何隨時間變化。您可以使用 AWS Config 來評估 AWS 資源的組態設定。您可以透過建立規則來執行此操作，該 AWS Config 規則代表您的理想組態設定。雖然 AWS Config 會持續追蹤 資源中發生的組態變更，但它會檢查這些變更是否違反規則中的任何條件。如果資源違反規則， 會將資源和規則 AWS Config 標記為不合規。 Application Manager會顯示 AWS Config 規則的合規資訊。若要在 中檢視此資料Application Manager，您必須設定 和 AWS Config。如需定價資訊，請參閱 AWS Config 定價。

(選用) 建立 State Manager 關聯

您可以使用 Systems Manager State Manager，以建立指派到受管節點的組態。稱為關聯的組態會定義您想在節點上維持的狀態。若要在 Application Manager 中檢視關聯合規資料，您必須設定一個或多個 State Manager 關聯。State Manager 無須額外付費。

(選用) 安裝和設定 OpsCenter

您可以使用 OpsCenter 在 Application Manager 中檢視有關您資源的營運工作項目 (OpsItems)。您可以根據警示和事件設定 Amazon CloudWatch 和 Amazon EventBridge，從而自動將 OpsItems 傳送至 OpsCenter。您也可以手動輸入 OpsItems。如需定價資訊，請參閱 AWS Systems Manager 定價。

設定的 Systems Manager Application Manager 的許可

如果您的 AWS Identity and Access Management (IAM) 實體 （ 例如使用者、群組或角色 ） 可存取本主題中列出的 API 操作 AWS Systems Manager，您可以使用 中Application Manager工具 的所有功能。API 操作分成兩個表格，可協助您了解它們執行的不同函數。

下表列出了您在 Application Manager 中選擇資源時 Systems Manager 會呼叫的 API 操作，因為您想要檢視資源詳細資訊。例如，如果 Application Manager 列出了 Amazon EC2 Auto Scaling 群組且您選擇該群組來檢視其詳細資訊，則 Systems Manager 會呼叫 `autoscaling:DescribeAutoScalingGroups` API 操作。如果您的帳戶中沒有任何 Auto Scaling 群組，則不會從 Application Manager 呼叫此 API 操作。

僅限資源詳細資訊

```
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
```

僅限資源詳細資訊

```
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
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 API 操作，其中 Systems Manager 會使用這些操作變更 Application Manager 中列出的應用程式和資源或檢視所選應用程式或資源的操作資訊。

應用程式動作與詳細資訊

```
applicationinsights:CreateApplication
applicationinsights:DescribeApplication
applicationinsights:ListProblems
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: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
```

應用程式動作與詳細資訊

```
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
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
```

應用程式動作與詳細資訊

```
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:UpdateOpsItem
ssm:UpdateOpsMetadata
ssm:UpdateServiceSetting
tag:GetTagKeys
tag:GetTagValues
tag:TagResources
tag:UntagResources
```

所有 Application Manager 許可的範例政策

若要設定 IAM 實體 (例如使用者、群組或角色) 的 Application Manager 許可，請使用下列範例建立 IAM 政策。此政策範例包含 Application Manager 使用的所有 API 操作。

```
                {
    "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

您可以從連接到使用者、群組或角色的 IAM 許可政策中刪除以下 API 操作，進而限制使用者在 Application Manager 變更應用程序和資源的能力。移除這些動作會在 Application Manager 中建立唯讀體驗。以下是允許使用者變更應用程式或任何其他相關資源的所有 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
```

如需有關建立和編輯 IAM 政策的資訊，請參閱《IAM 使用者指南》中的建立 IAM 政策。如需有關如何將此政策指派給 IAM 實體 (例如使用者、群組或角色) 的資訊，請參閱新增和移除 IAM 身分許可。

將應用程式和容器叢集新增至 Application Manager

Application Manager 是 的元件 AWS Systems Manager。在 中Application Manager，應用程式是您想要做為單位運作的邏輯 AWS 資源群組。此邏輯群組可以代表應用程式的不同版本、運算子的擁有權邊界或開發人員環境等等。

第一次開啟 時Application Manager，會顯示Application Manager可為您執行的操作頁面。當您選擇開始使用時， Application Manager會自動匯入在其他 AWS 服務 或 Systems Manager 工具中建立之資源的中繼資料。 Application Manager 接著 會在依預先定義類別分組的清單中顯示這些資源。

對於 Applications (應用程式)，清單包含下列項目：

- AWS CloudFormation 堆疊
- 自訂應用程式
- AWS Launch Wizard 應用程式
- AppRegistry 應用程式
- AWS SAP Enterprise 工作負載應用程式
- Amazon ECS 叢集
- Amazon EKS 叢集

匯入完成後，您可以在這些預先定義的類別中檢視應用程式或特定資源的操作資訊。或者，如果您想要提供有關資源集合的更多內容，您可以在 Application Manager 中手動建立應用程式。然後，您可以將資源或資源群組新增至該應用程式中。在 Application Manager 中建立應用程式之後，您可以在應用程式環境中檢視資源的操作資訊。

在 Application Manager 中建立應用程式

使用下列程序在 Application Manager 建立應用程式並將資源新增至該應用程式。

在 Application Manager 中建立應用程式

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Application Manager。
3. 選擇建立應用程式。
4. 從下拉式清單中選擇選項，並完成出現頁面中的欄位。

使用 應用程式

Application Manager 是 的元件 AWS Systems Manager。該區段包含可協助您使用 Application Manager 應用程式並檢視您的 AWS 資源的操作資訊的主題。

目錄

- Application Manager 中的應用程式概觀
- 管理應用程式 EC2 執行個體
- 檢視與應用程式關聯的資源
- 管理應用程式合規性
- 使用 CloudWatch Application Insights 監控應用程式
- 檢視應用程式的 OpsItems
- 管理應用程式日誌
- 使用自動化執行手冊修復應用程式問題
- 標記 Application Manager 中的資源
- 在 中使用 AWS CloudFormation 範本和堆疊 Application Manager
- 使用 Application Manager 中的叢集

Application Manager 中的應用程式概觀

在 的Application Manager元件中 AWS Systems Manager，概觀索引標籤會顯示 Amazon CloudWatch 警示、操作工作項目 (OpsItems)、CloudWatch Application Insights 和 Runbook 歷史記錄的摘要。選擇任意卡的 View all (檢視所有) 以開啟相應的標籤，您可以在其中檢視所有應用程式洞察、警示、OpsItems 或 Runbook 歷史記錄。

關於 Application Insights

CloudWatch Application Insights 會識別和設定金鑰指標、日誌，並在您所有的應用程式資源和技術堆疊中發出警示。Application Insights 持續監控指標和日誌，以偵測和建立異常及錯誤的關聯。系統偵測到錯誤和異常時，Application Insights 會產生 CloudWatch Events，可用於設定通知或採取動作。如果選擇 Monitoring (監控) 標籤上的 Edit configuration (編輯組態) 按鈕，系統會開啟 CloudWatch Application Insights 主控台。如需有關 Application Insight 的詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的什麼是 Amazon CloudWatch Application Insights。

關於 Cost Explorer

Application Manager 透過 Cost widget 和 Cost 索引標籤 AWS Cost Explorer，與 Cost AWS Management 的功能整合。在成本管理主控台中啟用 Cost Explorer 後，Application Manager 中的成本和成本索引標籤會顯示特定非容器應用程式或應用程式元件的成本資料。您可以使用小工具或索引標籤中的篩選條件，根據長條圖或折線圖中的不同時間週期、粒度和成本類型來檢視成本資料。

您可以選擇前往 AWS 成本管理主控台按鈕來啟用此功能。根據預設，系統會將資料篩選為過去三個月。對於非容器應用程式，如果您選擇 View all (檢視所有) 按鈕，則 Application Manager 會開啟 Resources (資源) 標籤。對於容器應用程式，View all (檢視所有) 按鈕會開啟 AWS Cost Explorer 主控台。

您可以在此標籤上執行的動作

您可以在此頁面的 Overview (概觀) 標籤上開啟並存取下列小工具的相關資訊。啟用小工具時，請選擇 View all (檢視全部) 以查看該區域的相關應用程式詳細資訊。

- 在 Insights and Alarms (深入解析和警示) 區段中，選擇嚴重性數字以開啟 Monitoring (監控) 標籤，您可以在其中檢視所選嚴重性警示的更多詳細資訊。
- 在 Cost (成本) 區段中，選擇 View all (檢視全部) 以開啟 Resources (資源) 標籤，您可以在其中檢視特定應用程式或應用程式元件的成本資料。
- 在合規區段中，選擇檢視全部以開啟合規索引標籤，您可以在其中檢視來自 AWS Config 和 State Manager關聯的合規資訊。

> **Note**
>
> 若要檢視修補程式合規詳細資訊，則請直接選擇 Compliance (合規) 標籤。然後，您可以檢視所選應用程式使用之受管節點的修補程式合規詳細資訊。

- 在 Runbooks 區段中，選擇 Systems Manager Documents (文件) 頁面中的 Runbook 以將其開啟，您可以在其中檢視有關文件的更多詳細資訊。
- 在 OpsItems 區段中，選擇一種嚴重性以開啟 OpsItems 標籤，您可以在其中檢視所選嚴重性的所有 OpsItems。
- 選擇 View all (檢視所有) 按鈕來開啟相應的標籤。您可以檢視應用程式的所有警示、OpsItems 或 Runbook 歷程記錄項目。

開啟 Overview (概觀) 標籤

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Application Manager。
3. 在 Applications (應用程式) 區段中，選擇類別。如果您想要在 Application Manager 中開啟您手動建立的應用程式，選擇 Custom applications (自訂應用程式)。
4. 在清單中選擇應用程式。Application Manager 會開啟 Overview (概觀) 標籤。

管理應用程式 EC2 執行個體

Application Manager 與 Amazon Elastic Compute Cloud (Amazon EC2) 整合，以便在應用程式環境中顯示執行個體的相關資訊。Application Manager 以圖形格式顯示所選應用程式的執行個體狀態、狀態和 Amazon EC2 Auto Scaling 運作狀態。Instances (執行個體) 標籤也包含一個資料表，其中包含應用程式中每個執行個體的下列資訊：

- 執行個體狀態 (待定、停止中、執行中、已停止)
- SSM Agent 的 Ping 狀態
- 在執行個體上處理的最新 Systems Manager Automation 執行手冊的狀態和名稱
- 每個狀態的 Amazon CloudWatch Logs 警示計數。
  - `ALARM` – 指標或表達式在定義的閾值外。
  - `OK` – 指標或表達式在定義的閾值內。
  - `INSUFFICIENT_DATA` – 警示剛開始無法使用指標，或資料不足無法讓指標判斷警示狀態。
- 父群組和個別自動擴展群組的 Auto Scaling 群組運作狀態

如果您在 All instances (所有執行個體) 資料表中選擇執行個體，則 Application Manager 會在四個標籤中顯示該執行個體的相關資訊：

- Details (詳細資訊) – 來自 Amazon EC2 的所有執行個體詳細資訊，包括 Amazon Machine Image (AMI)、DNS 資訊、IP 地址資訊等。
- Health (運作狀態) – EC2 系統和執行個體狀態檢查提供的目前狀態。
- Execution history (執行歷史記錄) – 執行個體所處理的 Systems Manager Automation 執行手冊和 API 呼叫的執行日誌。
- CloudWatch alarms (CloudWatch 警示) – 執行個體引發的任何 CloudWatch 警示的名稱、狀態及其他資訊。

您可以在此標籤上執行的動作

您也可以在此頁面上執行下列動作：

- 啟動、停止和終止執行個體。
- 套用 Chef 配方。
- 將執行個體連接至 Auto Scaling 群組或從中分離。
- 啟用 SSM Agent 的自動化更新。

開啟 Instances (執行個體) 標籤

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Application Manager。
3. 在 Applications (應用程式) 區段中，選擇類別。如果您想要在 Application Manager 中開啟您手動建立的應用程式，則請選擇 Custom applications (自訂應用程式)。
4. 在清單中選擇應用程式。Application Manager 會開啟 Overview (概觀) 標籤。
5. 選擇執行個體標籤。

檢視您的應用程式執行個體的詳細資訊

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Application Manager。
3. 在 Applications (應用程式) 區段中，選擇類別。如果您想要在 Application Manager 中開啟您手動建立的應用程式，則請選擇 Custom applications (自訂應用程式)。
4. 在清單中選擇應用程式。Application Manager 會開啟 Overview (概觀) 標籤。
5. 選擇執行個體標籤。
6. 選取您要檢視其詳細資訊之執行個體旁的按鈕。
7. 檢視頁面底部的執行個體詳細資訊。

自動更新 SSM Agent

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Application Manager。
3. 在 Applications (應用程式) 區段中，選擇類別。如果您想要在 Application Manager 中開啟您手動建立的應用程式，則請選擇 Custom applications (自訂應用程式)。
4. 在清單中選擇應用程式。Application Manager 會開啟 Overview (概觀) 標籤。
5. 選擇執行個體標籤。
6. 在代理程式動作下拉式清單中，選擇設定 SSM Agent 更新。
7. 選擇所有執行個體以設定所有受管執行個體的自動 SSM Agent 更新。或者，選擇執行個體，為應用程式中的單一執行個體設定自動 SSM Agent 更新。

8. 選取啟用自動更新切換開關。
9. 在指定排程下拉式清單中，選擇您要用於 SSM Agent 更新的排程。
10. 選取設定。

檢視與應用程式關聯的資源

在 中Application Manager， AWS Systems Manager資源索引標籤會顯示應用程式中 AWS 的資源。如果您選擇頂層元件，此頁面會顯示該元件及任何子元件的所有資源。如果您選擇子元件，此頁面只會顯示指派給該子元件的資源。

您可以在此標籤上執行的動作

您也可以在此頁面上執行下列動作：

- 選擇資源名稱以檢視其相關資訊，包括建立該資源的主控台提供的詳細資訊、標籤、Amazon CloudWatch 警示、 AWS Config 詳細資訊和 AWS CloudTrail 日誌資訊。
- 選擇資源名稱旁邊的選項按鈕。然後，選擇資源時間軸按鈕來開啟 AWS Config 主控台，您可以在其中檢視所選資源的合規資訊。
- 如果您已啟用 AWS Cost Explorer，Cost Explorer 區段會顯示特定非容器應用程式或應用程式元件的成本資料。您可以選擇前往 AWS 成本管理主控台按鈕來啟用此功能。您可以使用本區段中的篩選條件來檢視應用程式的成本資訊。

開啟 Resources (資源) 索引標籤

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Application Manager。
3. 在 Applications (應用程式) 區段中，選擇類別。如果您想要在 Application Manager 中開啟您手動建立的應用程式，選擇 Custom applications (自訂應用程式)。
4. 在清單中選擇應用程式。Application Manager 會開啟 Overview (概觀) 標籤。
5. 選擇 Resources (資源) 索引標籤。

管理應用程式合規性

在 的Application Manager元件中 AWS Systems Manager，組態頁面會顯示AWS Config資源和組態規則合規資訊。此頁面也會顯示 AWS Systems Manager State Manager關聯合規資訊。您可以選擇資源、規則或關聯，以開啟對應的主控台，進而取得詳細資訊。此頁面會顯示過去 90 天的合規資訊。

您可以在此標籤上執行的動作

您也可以在此頁面上執行下列動作：

- 選擇資源名稱以開啟 AWS Config 主控台，您可以在其中檢視所選資源的合規資訊。
- 選擇資源名稱旁邊的選項按鈕。然後，選擇資源時間軸按鈕來開啟 AWS Config 主控台，您可以在其中檢視所選資源的合規資訊。
- 在 Config rules compliance (組態規則合規) 區段中，您可以執行下列操作：
  - 選擇規則名稱以開啟 AWS Config 主控台，您可以在其中檢視該規則的相關資訊。
  - 選擇新增規則以開啟 AWS Config 主控台，您可以在其中建立規則。
  - 選擇規則名稱旁的選項按鈕，選擇 Actions (動作)，然後選擇 Manage remediation (管理修正) 以變更規則的修正動作。
  - 選擇規則名稱旁的選項按鈕，選擇動作，然後選擇重新評估，以對選取的規則 AWS Config 執行合規檢查。
- 在 Association compliance (關聯合規) 區段中，您可以執行以下操作：
  - 選擇關聯名稱以開啟 Associations (關聯) 頁面，您可在其中檢視該關聯的相關資訊。
  - 選擇 Create association (建立關聯) 以開啟 Systems ManagerState Manager，您可以在其中建立關聯。
  - 選擇關聯名稱旁的選項按鈕，然後選擇 Apply association (套用關聯) 以立即啟動關聯中指定的所有動作。

開啟 Compliance (合規) 標籤

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Application Manager。
3. 在 Applications (應用程式) 區段中，選擇類別。如果您想要在 Application Manager 中開啟您手動建立的應用程式，選擇 Custom applications (自訂應用程式)。
4. 在清單中選擇應用程式。Application Manager 會開啟 Overview (概觀) 標籤。

5. 選擇 Compliance (合規) 標籤。

使用 CloudWatch Application Insights 監控應用程式

在 的Application Manager元件中 AWS Systems Manager，監控索引標籤會顯示應用程式中資源的 Amazon CloudWatch Application Insights 和警示詳細資訊。

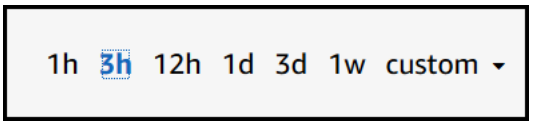
關於 Application Insights

CloudWatch Application Insights 會識別和設定金鑰指標、日誌，並在您所有的應用程式資源和技術堆疊中發出警示。Application Insights 持續監控指標和日誌，以偵測和建立異常及錯誤的關聯。系統偵測到錯誤和異常時，Application Insights 會產生 CloudWatch Events，可用於設定通知或採取動作。如果選擇 Monitoring (監控) 標籤上的 Edit configuration (編輯組態) 按鈕，系統會開啟 CloudWatch Application Insights 主控台。如需有關 Application Insight 的詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的什麼是 Amazon CloudWatch Application Insights。

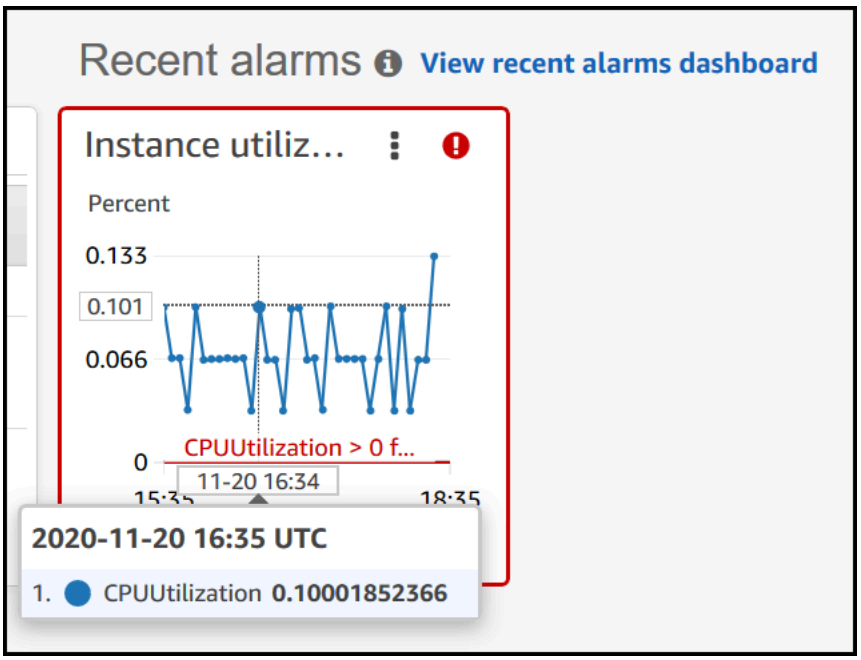
您可以在此標籤上執行的動作

您也可以在此頁面上執行下列動作：

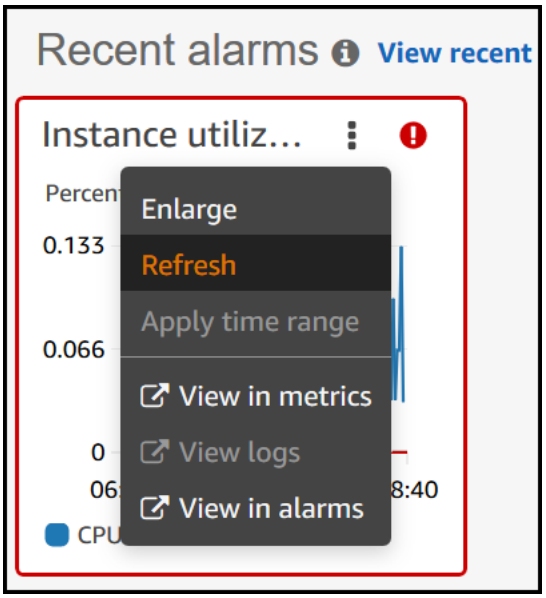
- 在依服務排序的警示區段中選擇 AWS 服務名稱，以開啟 CloudWatch 至選取的服務和警示。
- 選取其中一個預先定義的時段值，在 Recent alarms (最近警示) 區段中，調整小工具中顯示的資料的時段。您可以選擇 custom (自訂) 來定義您自己的時段。



- 將游標暫留在 Recent alarms (最近警示) 區段中的小工具上，以檢視特定時間的資料快顯。



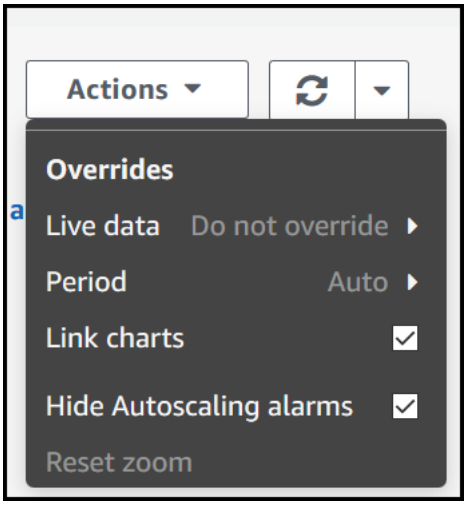
- 選擇小工具中的選項選單，以檢視顯示選項。選擇 Enlarge (放大) 以展開小工具。選擇 Refresh (重新整理) 以更新小工具中的資料。按一下您的游標，並將其拖曳至小工具資料上，以選取特定範圍。然後，您可以選擇 Apply time range (套用時間範圍)。



- 選擇 Actions (動作) 選單，以檢視警示資料 Override (覆寫) 選項，其中包含下列項目：
  - 選擇是否要讓小工具顯示即時資料。即時資料是在尚未完全彙總的最後一分鐘內所發佈的資料。如果即時資料已關閉，則只會顯示至少過去一分鐘彙總期間內的資料點。例如，若使用 5 分鐘的期間，則 12:35 的資料點會從 12:35 彙總到 12:40，並在 12:41 顯示。

如果即時資料已開啟，則會在對應彙總時間間隔內發佈任何資料時，立即顯示最新資料點。每次重新整理顯示畫面時，最新資料點可能會在該彙總期間內發佈新資料的同時隨之變更。

- 指定即時資料的時段。
- 連結 Recent alarms (最近警示) 區段中的圖表，如此一來，當您放大或縮小某個圖表時，另一個圖表會同時放大或縮小。您可以取消連結圖表，以限制對一個圖表進行縮放。
- 隱藏 Auto Scaling 警示。



開啟 Monitoring (監控) 索引標籤

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Application Manager。
3. 在 Applications (應用程式) 區段中，選擇類別。如果您想要在 Application Manager 中開啟您手動建立的應用程式，選擇 Custom applications (自訂應用程式)。
4. 在清單中選擇應用程式。Application Manager 會開啟 Overview (概觀) 標籤。
5. 選擇 Monitoring (監控) 索引標籤。

檢視應用程式的 OpsItems

在 中Application Manager， 的元件 AWS Systems Manager，OpsItems標籤會顯示所選應用程式中資源的操作工作項目 (OpsItems)。您可以設定 Systems Manager OpsCenter 以從 Amazon CloudWatch 警示和 Amazon EventBridge 事件自動建立 OpsItems。您也可以手動建立 OpsItems。

您可以在此標籤上執行的動作

您也可以在此頁面上執行下列動作：

- 使用搜尋欄位篩選 OpsItems 的清單。您可以依OpsItem 名稱、ID、來源 ID 或嚴重性進行篩選。您也可以根據狀態篩選清單。OpsItems 支援下列狀態：Open (開放)、In progress (正在進行)、Open and In progress (開放並正在進行)、Resolved (已解決) 或 All (所有)。
- 選擇其旁邊的選項按鈕然後選擇 Set status (設定狀態) 選單中的選項，變更 OpsItem 的狀態。
- 開啟 Systems Manager OpsCenter，以透過選擇 Create OpsItem (建立 OpsItem) 建立 OpsItem。

若要開啟 OpsItems 標籤

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Application Manager。
3. 在 Applications (應用程式) 區段中，選擇類別。如果您想要在 Application Manager 中開啟您手動建立的應用程式，選擇 Custom applications (自訂應用程式)。
4. 在清單中選擇應用程式。Application Manager 會開啟 Overview (概觀) 標籤。
5. 選擇 (OpsItems) 索引標籤。

管理應用程式日誌

在 中Application Manager，Logs AWS Systems Manager索引標籤會顯示來自 Amazon CloudWatch Logs 的日誌群組清單。

您可以在此標籤上執行的動作

您也可以在此頁面上執行下列動作：

- 選擇日誌群組名稱，以便在 CloudWatch Logs 中將其開啟。然後，您可以選擇日誌串流，以在應用程式環境中檢視資源的日誌。
- 選擇 Create log groups (建立日誌群組) 以在 CloudWatch Logs 中建立日誌群組。

開啟 Logs (日誌) 索引標籤

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Application Manager。

3. 在 Applications (應用程式) 區段中，選擇類別。如果您想要在 Application Manager 中開啟您手動建立的應用程式，選擇 Custom applications (自訂應用程式)。
4. 在清單中選擇應用程式。Application Manager 會開啟 Overview (概觀) 標籤。
5. 選擇 Logs (日誌) 索引標籤。

使用自動化執行手冊修復應用程式問題

您可以使用 AWS Systems Manager Automation Runbook 來修復 中Application Manager工具 AWS 資源的問題。自動化執行手冊定義 Systems Manager 在自動化執行時，對受管執行個體和其他 AWS 資源執行的動作。自動化是 中的工具 AWS Systems Manager。Runbook 包含循序執行的一或多個步驟。每個步驟都是圍繞單一動作而建立的。來自一個步驟的輸出可以做為後續步驟中的輸入。

當您從 Application Manager 應用程式或叢集中選擇 Start Runbook (啟動 Runbook) 時，系統會根據應用程式或叢集中的資源類型，顯示篩選的可用 Runbook 清單。當您選擇您想要啟動的 Runbook 時，Systems Manager 會開啟 Execute automation document (執行自動化文件) 頁面。

Application Manager 包含下列增強功能，可搭配使用 Runbook。

- 如果您在 Application Manager 中選擇資源的名稱，然後選擇 Execute Runbook (執行 Runbook) 時，系統會顯示該資源類型的 Runbook 篩選清單。
- 您可以在清單中選擇 Runbook，然後選擇 Run for resources of same type (執行相同類型的資源)，從而針對所有相同類型的資源執啟動自動化。

開始之前

在您開始從 Application Manager 啟動 Runbook 之前，請執行下列動作：

- 請確認您有啟動 Runbook 的正確許可。如需詳細資訊，請參閱設定自動化。
- 檢閱有關啟動 Runbook 的自動化程序文件。如需詳細資訊，請參閱執行由 Systems Manager Automation 提供支援的自動化操作。

若要從 Application Manager 啟動 Runbook

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Application Manager。

3. 在 Applications (應用程式) 區段中，選擇類別。如果您想要在 Application Manager 中開啟您手動建立的應用程式，選擇 Custom applications (自訂應用程式)。
4. 在清單中選擇應用程式。Application Manager 會開啟 Overview (概觀) 標籤。
5. 選擇開始執行手冊。Application Manager 會開啟 Automation 小工具彈出視窗。如需有關 Automation 小工具中的選項的資訊，請參閱 執行由 Systems Manager Automation 提供支援的自動化操作。

標記 Application Manager 中的資源

您可以在 中快速新增或刪除應用程式 AWS 和資源上的標籤Application Manager。使用下列程序將標籤新增至應用程式和該應用程式中的所有 AWS 資源，或從中刪除標籤。

若要從應用程式或該應用程式的所有 資源新增標籤，或從中刪除標籤。

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Application Manager。
3. 在 Applications (應用程式) 區段中，選擇類別。如果您想要在 Application Manager 中開啟您手動建立的應用程式，選擇 Custom applications (自訂應用程式)。
4. 在清單中選擇應用程式。Application Manager 會開啟 Overview (概觀) 標籤。
5. 在 Application information (應用程式資訊) 區段中，選擇 Application tags (應用程式標籤) 下的編號。如果未向應用程式指派任何標籤，則數字為零。
6. 若要新增標籤，請選擇 Add new tag (新增新標籤)。指定金鑰和選用的值。若要刪除標籤，請選擇 Remove (移除)。
7. 選擇 Save (儲存)。

使用下列程序將標籤新增至 Application Manager 中的指定資源，或從中刪除標籤。

若要將標籤新增至資源或從中刪除標籤

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Application Manager。
3. 在 Applications (應用程式) 區段中，選擇類別。如果您想要在 Application Manager 中開啟您手動建立的應用程式，選擇 Custom applications (自訂應用程式)。

4. 在清單中選擇應用程式。Application Manager 會開啟 Overview (概觀) 標籤。
5. 選擇 Resources (資源) 索引標籤。
6. 選擇資源名稱。
7. 在 Tags (標籤) 區段中，選擇 Edit (編輯)。
8. 若要新增標籤，請選擇 Add new tag (新增新標籤)。指定金鑰和選用的值。若要刪除標籤，請選擇 Remove (移除)。
9. 選擇 Save (儲存)。

在 中使用 AWS CloudFormation 範本和堆疊 Application Manager

Application Manager中的工具 AWS Systems Manager可協助您透過整合 ，為您的應用程式佈建和管理資源 AWS CloudFormation。您可以在 Application Manager 中建立、編輯和刪除 AWS CloudFormation 範本和堆疊。堆疊是一組 AWS 資源，您可將它視為單一單位進行管理。這表示您可以使用 CloudFormation 堆疊來建立、更新或刪除 AWS 資源集合。範本是 JSON 或 YAML 格式的文字檔案，可指定您要在堆疊中佈建的資源。

Application Manager也包含範本程式庫，您可以在其中複製、建立和存放範本。Application Manager 和 CloudFormation 會顯示有關堆疊目前狀態的相同資訊。範本和範本更新會存放在 Systems Manager 中，直到您佈建堆疊為止，此時變更也會顯示在 CloudFormation 中。

當您在 Application Manager 建立堆疊後，CloudFormation stacks (CloudFormation 堆疊) 頁面會顯示其相關的實用資訊。這包括用於建立它的範本、您堆疊中的資源的 OpsItems 資源計數、堆疊狀態，以及偏離狀態。

關於 Cost Explorer

Application Manager 透過成本小工具 AWS Cost Explorer與成本AWS 管理的功能整合。在成本管理主控台中啟用 Cost Explorer 後，Application Manager 中的 Cost (成本) 小工具會顯示特定非容器應用程式或應用程式元件的成本資料。您可以使用小工具中的篩選條件，根據長條圖或折線圖中的不同時間週期、粒度和成本類型來檢視成本資料。

您可以選擇前往 AWS 成本管理主控台按鈕來啟用此功能。根據預設，系統會將資料篩選為過去三個月。對於非容器應用程式，如果您選擇 View all (檢視所有) 按鈕，則 Application Manager 會開啟 Resources (資源) 標籤。對於容器應用程式，View all (檢視所有) 按鈕會開啟 AWS Cost Explorer 主控台。

> **Note**
>
> Cost Explorer 使用標籤來追蹤應用程式成本。如果您的 AWS CloudFormation 堆疊型應用程式未設定`AppManagerCFNStackKey`標籤索引鍵，Cost Explorer 無法在 中呈現準確的成本資料Application Manager。如果未偵測到 `AppManagerCFNStackKey` 標籤鍵，系統會在主控台中提示您將標籤新增至 CloudFormation 堆疊，以啟用成本追蹤。新增標籤索引鍵會將其映射至堆疊的 Amazon Resource Name (ARN)，並讓 Cost (成本) 小工具能顯示準確的成本資料。

> **Important**
>
> 新增 `AppManagerCFNStackKey` 標籤將觸發堆疊更新。在新增使用者標籤之後，任何在最初部署堆疊之後執行的手動組態都不會得到反映。如需有關資源更新行為的詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的更新堆疊資源的行為一節

開始之前

在您使用 Application Manager 建立、編輯或刪除 CloudFormation 範本和堆疊之前，請使用下列連結了解 CloudFormation 概念。

- 什麼是 AWS CloudFormation？
- AWS CloudFormation 最佳實務
- 了解範本的基本知識
- 使用 AWS CloudFormation 堆疊
- 使用 AWS CloudFormation 範本
- 範例範本

主題

- 使用 Application Manager 管理 AWS CloudFormation 範本
- 使用 Application Manager 管理 AWS CloudFormation 堆疊

使用 Application Manager 管理 AWS CloudFormation 範本

Application Manager是 中的工具 AWS Systems Manager，包含範本程式庫和其他工具，可協助您管理 AWS CloudFormation 範本。此區段包含下列資訊：

主題

- 使用範本程式庫
- 建立範本
- 編輯範本

使用範本程式庫

Application Manager 範本程式庫提供的工具可協助您檢視、建立、編輯、刪除和複製範本。您也可以直接從範本程式庫佈建堆疊。範本會以類型 `CloudFormation` 的 Systems Manager (SSM) 文件形式存放。透過將範本儲存為 SSM 文件，您可以使用版本控制來處理範本的不同版本。您也可以設定許可及共享範本。成功佈建堆疊之後，Application Manager 和 CloudFormation 中會提供堆疊和範本。

開始之前

我們建議您先閱讀下列主題，以進一步了解 SSM 文件，然後再開始使用 Application Manager 中的 CloudFormation 範本。

- AWS Systems Manager Documents
- 共用 SSM 文件
- 共用 SSM 文件的最佳實務

若要在 Application Manager 中檢視範本程式庫

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Application Manager。
3. 選擇 CloudFormation 範本程式庫。

建立範本

下列程序說明如何在 Application Manager 中建立 CloudFormation 範本。當您建立範本時，您可以在 JSON 或 YAML 中輸入範本的堆疊詳細資訊。如果您不熟悉 JSON 或 YAML，您可以使用 AWS Infrastructure Composer，這是一種視覺化建立和修改範本的工具。如需詳細資訊，請參閱AWS CloudFormation 《使用者指南》中的使用 Infrastructure Composer 以視覺化方式建立範本。如需範本結構和語法的相關資訊，請參閱AWS CloudFormation 《使用者指南》中的AWS CloudFormation 範本區段。

您也可以從多個範本程式碼片段建構範本。範本程式碼片段是可向您示範如何撰寫特定資源範本的範例。例如，您可以檢視 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、Amazon Simple Storage Service (Amazon S3) 網域、 AWS CloudFormation 映射等程式碼片段。程式碼片段會依資源分組。您可以在《AWS CloudFormation 使用者指南》中的一般範本程式碼片段區段中尋找一般用途 AWS CloudFormation 程式碼片段。

在 Application Manager (主控台) 中建立 CloudFormation 範本

利用以下程序，使用 AWS Management Console在 Application Manager 中建立 CloudFormation 範本。

若要在 Application Manager 中建立 CloudFormation 範本

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Application Manager。
3. 選擇 CloudFormation 範本程式庫，然後選擇建立範本或選擇現有範本，然後選擇動作、複製。
4. 對於 Name (名稱)，輸入範本的名稱，以協助您識別其建立的資源或堆疊的用途。
5. (選用) 對於 Version name (版本名稱)，輸入名稱或編號以識別範本版本。
6. 在 Code editor (程式碼編輯器) 區段中，選擇 YAML 或 JSON，然後輸入或複製並貼上您的範本程式碼。
7. (選用) 在 Tags (標籤) 區段中，將一個或多個標籤索引鍵名稱/值對套用到範本。
8. （選用）在許可區段中，輸入 AWS 帳戶 ID 並選擇新增帳戶。此動作會提供範本的讀取許可。帳戶擁有者可以佈建和複製範本，但無法編輯或刪除範本。
9. 選擇建立。範本會存放在 Systems Manager (SSM) 文件服務中。

在 Application Manager (命令列) 中建立 CloudFormation 範本

在 JSON 或 YAML 中建立 CloudFormation 範本的內容之後，您可以使用 AWS Command Line Interface (AWS CLI) 或 AWS Tools for PowerShell 將範本儲存為 SSM 文件。將每個########取代為您自己的資訊。

開始之前

AWS Tools for PowerShell如果您尚未安裝和設定 AWS CLI 或 。如需相關資訊，請參閱安裝或更新 AWS CLI的最新版本和安裝 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，此命令會傳回類似如下的回應。

```
{
    "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"
            }
        ]
    }
```

編輯範本

利用以下程序，在 Application Manager 中編輯 CloudFormation 範本。佈建使用已更新範本的堆疊之後，即可在 CloudFormation 中使用範本變更。

若要在 Application Manager 中編輯 CloudFormation 範本

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Application Manager。
3. 選擇 CloudFormation 範本程式庫
4. 選擇一個範本，然後選擇 Actions (動作)、Edit (編輯)。您無法變更範本的名稱，但您可以變更所有其他詳細資訊。
5. 選擇儲存。範本會存放在 Systems Manager 文件服務中。

使用 Application Manager 管理 AWS CloudFormation 堆疊

Application Manager是 中的工具 AWS Systems Manager，可透過與 整合，協助您佈建和管理應用程式的 資源 AWS CloudFormation。您可以在 Application Manager 中建立、編輯和刪除 CloudFormation 範本和堆疊。堆疊是您可以單一單位管理的一組 AWS 資源。這表示您可以使用 CloudFormation 堆疊建立、更新或刪除 AWS 資源集合。範本是 JSON 或 YAML 格式的文字檔案，可指定您要在堆疊中佈建的資源。此區段包含下列資訊：

主題

- 建立堆疊
- 更新堆疊

建立堆疊

下列程序說明如何使用 Application Manager 建立 CloudFormation 堆疊。堆疊以範本為基礎。建立堆疊時，您可以選擇現有的範本，或建立新的範本。建立堆疊之後，系統會立即嘗試建立堆疊中識別的資源。在系統成功佈建資源之後，範本和堆疊就可以在 Application Manager 和 CloudFormation 中檢視和編輯。

> **Note**
>
> 使用 Application Manager建立堆疊無需付費，但您需要為堆疊中建立 AWS 的資源付費。

使用 Application Manager (主控台) 建立 CloudFormation 堆疊

利用以下程序使用 AWS Management Console中的 Application Manager 建立堆疊。

若要刪除 CloudFormation 堆疊

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Application Manager。
3. 選擇建立應用程式、CloudFormation Stack。
4. 在 Prepare a template (準備範本) 區段中，選擇一個選項。如果您選擇 Use an existing template (使用現有的範本) 中的標籤，您可以使用 Choose a template (選擇範本) 區段以找出您想要的範本。（如果您選擇其他選項之一，請完成精靈以準備範本。）
5. 選取範本名稱旁的按鈕，然後選擇下一步。
6. 在 Specify template details (指定範本詳細資訊) 頁面上，確認範本的詳細資訊，以確保程序會建立您想要的資源。

   - (選用) 在 Tags (標籤) 區段中，將一個或多個標籤索引鍵名稱/值對套用到範本。
   - 標籤是您指派給資源的選用性中繼資料。使用標籤，您即可以不同的方式 (例如用途、擁有者或環境) 將資源分類。
   - 選擇下一步。
7. 在 Edit stack details (編輯堆疊詳細資訊) 頁面上，針對 Stack name (堆疊名稱)，輸入可協助您識別堆疊建立的資源或用途的名稱。

   - Parameters (參數) 區段包含範本中指定的所有選用和必要參數。在每個欄位中，輸入一個或多個參數。

- (選用) 在 Tags (標籤) 區段中將一個或多個標籤索引鍵名稱/值對套用到堆疊。
- （選用）在許可區段中，指定 AWS Identity and Access Management (IAM) 角色名稱或 IAM Amazon Resource Name (ARN)。系統會使用指定的服務角色來建立堆疊中指定的所有資源。如果您未指定 IAM 角色，則 AWS CloudFormation 會使用從您的使用者憑證產生的暫時工作階段。如需此 IAM 角色的詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的 AWS CloudFormation 服務角色。
- 選擇下一步。

8. 請詳閱 Review and provision (檢閱和佈建) 頁面上的堆疊詳細資訊。在此頁面上選擇 Edit (編輯) 按鈕，以進行變更。
9. 選擇 Provision stack (佈建堆疊)。

Application Manager 會顯示 CloudFormation stacks (CloudFormation 堆疊) 頁面以及堆疊建立和部署的狀態。如果 CloudFormation 無法建立和佈建堆疊，請參閱《AWS CloudFormation 使用者指南》。

- 堆疊狀態碼
- 疑難排解 AWS CloudFormation

佈建和執行堆疊資源之後，使用者可以使用建立資源的底層服務，直接編輯資源。例如，使用者可以使用 Amazon Elastic Compute Cloud (Amazon EC2) 主控台來更新已建立成為 CloudFormation 堆疊一部分的伺服器執行個體。有些變更可能是意外，有些則是為了回應時間急迫性運作事件而刻意為之。無論如何，在 CloudFormation 外部所做的變更會使堆疊更新或刪除操作變得複雜。您可以使用偏離偵測或偏離狀態，來識別已在 CloudFormation 管理之外發生組態變更的堆疊資源。如需偏離狀態的詳細資訊，請參閱偵測堆疊和資源未受管的組態變更。

使用 Application Manager (命令列) 建立 CloudFormation 堆疊

使用下列 AWS Command Line Interface (AWS CLI) 程序，使用存放在 Systems Manager 中做為 SSM 文件的 CloudFormation 範本來佈建堆疊。將每個########取代為您自己的資訊。如需有關建立堆疊的其他 AWS CLI 程序的資訊，請參閱AWS CloudFormation 《 使用者指南》中的建立堆疊。

開始之前

AWS Tools for PowerShell如果您尚未安裝和設定 AWS CLI 或 。如需相關資訊，請參閱安裝或更新 AWS CLI的最新版本和安裝 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" `
```

更新堆疊

您可以直接在 Application Manager 中編輯堆疊，以將更新部署到 CloudFormation 堆疊中。使用直接更新，您可以指定範本或輸入參數的更新。在您儲存和部署變更之後，CloudFormation 會根據您指定的變更更新 AWS 資源。

您可以在進行更新堆疊之前，預覽 CloudFormation 將對堆疊進行的變更，請使用變更集。如需詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的使用變更集更新堆疊。

若要在 Application Manager 中更新 CloudFormation 堆疊

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Application Manager。
3. 選取應用程式名稱旁的選項按鈕，然後選擇動作、更新堆疊。
4. 在 Specify template source (指定範本來源) 頁面上，選擇下列其中一個選項，然後選擇 Next (下一步)。

- 選擇 Use the template code currently provisioned in the stack (使用目前在堆疊中佈建的範本程式碼)，以檢視範本。選擇 Versions (版本) 清單的範本版本，然後選擇 Next (下一步)。
- 選擇 Switch to a different template (切換至不同的範本)，以選擇或建立堆疊的新範本。

5. 完成範本的變更之後，請選擇 Next (下一步)。
6. 在 Edit stack details (編輯堆疊詳細資訊) 頁面上，您可以編輯參數、標籤和許可。您不能變更堆疊名稱。進行變更，然後選擇 Next (下一步)。
7. 請詳閱 Review and provision (檢閱和佈建) 頁面上的堆疊詳細資訊，然後選擇 Provision stack (佈建堆疊)。

使用 Application Manager 中的叢集

本節包含的主題可協助您在 Application Manager的元件中使用 Amazon Elastic Container Service (Amazon ECS) 和 Amazon Elastic Kubernetes Service (Amazon EKS) 容器叢集 AWS Systems Manager。

目錄

- 在 Application Manager 中使用 Amazon ECS
- 在 Application Manager 中使用 Amazon EKS
- 使用叢集的 Runbook

在 Application Manager 中使用 Amazon ECS

透過 中的Application Manager工具 AWS Systems Manager，您可以檢視和管理 Amazon Elastic Container Service (Amazon ECS) 叢集基礎設施。 會使用叢集的 Amazon Resource Name (ARN) 做為標籤值，將標籤Application Manager套用至 Amazon ECS 叢集。 Application Manager 提供叢集中運算、聯網和儲存資源的元件執行期檢視。

> **Note**
>
> 您無法在 Application Manager 中管理或檢視有關容器的操作資訊。您只能管理和檢視託管 Amazon ECS 資源之基礎設施的操作資訊。

您可以在此標籤上執行的動作

您也可以在此頁面上執行下列動作：

- 選擇 Manage cluster (管理叢集) 以開啟 Amazon ECS 中的叢集。
- 選擇 View all (檢視所有) 以檢視叢集中的資源清單。
- 選擇 View in CloudWatch (在 CloudWatch 中檢視) 以在 Amazon CloudWatch 中檢視資源警示。
- 選擇 Manage nodes (管理節點) 或 Manage Fargate profiles (管理 Fargate 設定檔)，以在 Amazon ECS 中檢視這些資源。
- 選擇資源 ID，即可在建立資源的主控台中檢視資源 ID 的詳細資訊。
- 檢視與您的叢集相關的 OpsItems 清單。
- 檢視已在叢集上執行的 Runbook 歷程記錄。

若要開啟 ECS cluster (ECS 叢集)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Application Manager。
3. 在 Container clusters (容器叢集) 區段中，選擇 ECS clusters (ECS 叢集)。
4. 在清單中選擇叢集。Application Manager 會開啟 Overview (概觀) 標籤。

在 Application Manager 中使用 Amazon EKS

Application Manager是 中的工具 AWS Systems Manager，與 Amazon Elastic Kubernetes Service (Amazon EKS) 整合，以提供 Amazon EKS 叢集基礎設施運作狀態的相關資訊。 會使用叢集的 Amazon Resource Name (ARN) 做為標籤值，將標籤Application Manager套用至 Amazon EKS 叢集。 Application Manager 提供叢集中運算、聯網和儲存資源的元件執行期檢視。

> **Note**
>
> 您無法在 Application Manager 中管理或檢視有關 Amazon EKS Pod 或容器的操作資訊。您只能管理和檢視託管 Amazon EKS 資源之基礎設施的操作資訊。

您可以在此標籤上執行的動作

您也可以在此頁面上執行下列動作：

- 選擇 Manage cluster (管理叢集) 以開啟 Amazon EKS 中的叢集。
- 選擇 View all (檢視所有) 以檢視叢集中的資源清單。

- 選擇 View in CloudWatch (在 CloudWatch 中檢視) 以在 Amazon CloudWatch 中檢視資源警示。
- 選擇 Manage nodes (管理節點) 或 Manage Fargate profiles (管理 Fargate 設定檔)，以在 Amazon EKS 中檢視這些資源。
- 選擇資源 ID，即可在建立資源的主控台中檢視資源 ID 的詳細資訊。
- 檢視與您的叢集相關的 OpsItems 清單。
- 檢視已在叢集上執行的 Runbook 歷程記錄。

若要開啟 EKS clusters (EKS 叢集) 應用程式

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Application Manager。
3. 在 Container clusters (容器叢集) 區段中，選擇 EKS clusters (EKS 叢集)。
4. 在清單中選擇叢集。Application Manager 會開啟 Overview (概觀) 標籤。

使用叢集的 Runbook

您可以使用 AWS Systems Manager Systems Manager Automation Runbook 來修復 AWS 中 Application Manager工具 資源的問題。當您從 Application Manager 叢集中選擇 Start Runbook (啟動 Runbook) 時，系統會根據叢集中的資源類型，顯示篩選的 Runbook 清單。當您選擇您想要啟動的 Runbook 時，Systems Manager 會開啟 Execute automation document (執行自動化文件) 頁面。

開始之前

在您開始從 Application Manager 啟動 Runbook 之前，請執行下列動作：

- 請確認您有啟動 Runbook 的正確許可。如需詳細資訊，請參閱設定自動化。
- 檢閱有關啟動 Runbook 的自動化程序文件。如需詳細資訊，請參閱執行由 Systems Manager Automation 提供支援的自動化操作。
- 如果您打算一次性在多個資源上啟動 Runbook，請檢閱有關使用目標和速率控制的文件。如需詳細資訊，請參閱大規模執行自動化操作。

若要從 Application Manager 啟動叢集的 Runbook

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。

2. 在導覽窗格中，選擇 Application Manager。
3. 在 Container clusters (容器叢集) 區段中，選擇容器類型。
4. 在清單中選擇叢集。Application Manager 會開啟 Overview (概觀) 標籤。
5. 在 Runbooks 標籤上，選擇 Start Runbook (啟動 Runbook)。Application Manager 會在新的標籤中開啟 Execute automation document (執行自動化文件) 頁面。如需有關 Execute automation document (執行自動化文件) 頁面中的選項的資訊，請參閱 執行由 Systems Manager Automation 提供支援的自動化操作。

AWS Systems Manager Parameter Store

Parameter Store是 中的工具 AWS Systems Manager，可為組態資料管理和秘密管理提供安全的階層式儲存。您可以存放密碼、資料庫字串、Amazon Machine Image (AMI) ID 和授權碼之類的資料做為參數值。您存放的值可以是純文字或加密資料。您可以使用您在建立參數時指定的唯一名稱，在您的指令碼、命令、SSM 文件，以及組態和自動化工作流程中參考 Systems Manager 參數。若要開始使用 Parameter Store，請開啟 Systems Manager 主控台。在導覽窗格中，選擇 Parameter Store。

Parameter Store 也與 Secrets Manager 整合。您可以在使用其他已支援參考 Parameter Store 參數的 AWS 服務 時，擷取 Secrets Manager 秘密。如需詳細資訊，請參閱從Parameter Store參數參考 AWS Secrets Manager 秘密。

> **Note**
>
> 若要實作密碼輪換生命週期，請使用 AWS Secrets Manager。您可以使用 Secrets Manager 在資料庫登入資料、API 金鑰和其他機密的整個生命週期輕鬆進行輪換、管理和擷取。如需詳細資訊，請參閱AWS Secrets Manager 《 使用者指南》中的什麼是 AWS Secrets Manager？。

Parameter Store 如何為我的組織帶來益處？

Parameter Store 提供這些好處：

- 使用安全、可擴展的託管秘密管理服務 (無伺服器需要管理)。
- 隔離您的資料與程式碼以改善您的安全態勢。
- 將組態資料和加密字串存放在階層和追蹤版本中。
- 以精密分級控制及稽核存取。

- 可靠地存放參數，因為 Parameter Store 託管在 AWS 區域的多個可用區域中。

誰應該使用Parameter Store？

- 任何想要以集中方式管理組態資料 AWS 的客戶。
- 想要存放不同登入資料和參考串流的軟體開發人員。
- 想要在其秘密和密碼發生變更或未變更時接收通知的系統管理員。

Parameter Store 有哪些功能？

- 變更通知

  您可以同時為參數和參數政策設定變更通知和呼叫自動化動作。如需詳細資訊，請參閱根據 Parameter Store 事件設定通知或觸發動作。
- 組織參數

  您可以單獨標記參數，以根據指派給參數的標籤協助您找出一或多個文件。例如，您可以為特定的環境或部門標記參數。
- 標籤版本

  您可以透過建立標籤來關聯參數版本的別名。如果參數有多個版本，標籤可協助您記住參數版本的目的。
- 資料驗證

  您可以建立指向 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體的參數，Parameter Store 會驗證這些參數，以確保它參考預期的資源類型，確保該資源存在並且客戶具有使用資源的許可。例如，您可以建立具有 Amazon Machine Image (AMI) ID 的參數作為具有 `aws:ec2:image` 資料類型的值，Parameter Store 會執行非同步驗證操作，以確保參數值符合 AMI ID 的格式要求，並且指定的 AMI 在您的 AWS 帳戶中可用。
- 參考秘密

  Parameter Store 已與 整合， AWS Secrets Manager 因此在使用已支援參考Parameter Store參數 AWS 服務 的其他 時，您可以擷取 Secrets Manager 秘密。
- 與其他帳戶共用參數

  您可以選擇將組態資料集中在單一 中， AWS 帳戶 並與需要存取它們的其他帳戶共用參數。
- 可從其他 存取 AWS 服務

您可以使用Parameter Store參數搭配其他 Systems Manager 工具 AWS 服務 ，以及從中央存放區擷取秘密和組態資料。參數可與 Systems Manager 工具搭配使用Run Command，例如 、自動化和State Manager中的工具 AWS Systems Manager。您也可以在許多其他 中參考參數 AWS 服務，包括下列項目：

- 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

- 與其他 整合 AWS 服務

  設定與下列項目的整合， AWS 服務 以進行加密、通知、監控和稽核：

  - AWS Key Management Service (AWS KMS)
  - Amazon Simple Notification Service (Amazon SNS)
  - Amazon CloudWatch：如需詳細資訊，請參閱 為參數及參數政策設定 EventBridge。
  - Amazon EventBridge：如需詳細資訊，請參閱 使用 Amazon SNS 通知監控 Systems Manager 狀態變更 和 參考：Systems Manager 的 Amazon EventBridge 事件模式和類型。
  - AWS CloudTrail：如需詳細資訊，請參閱 使用 記錄 AWS Systems Manager API 呼叫 AWS CloudTrail。

什麼是參數？

Parameter Store 參數是儲存在 Parameter Store 中的任何資料片段，例如文字區塊、名稱清單、密碼、AMI ID、授權金鑰等。您可以集中、安全地在您的指令碼、命令和 SSM 文件中參考這項資料。

當您參考參數時，使用以下慣例來指定參數名稱：

```
{{ssm:parameter-name}}
```

> **Note**
>
> 不能在其他參數的值中參考或巢套參數。參數值中不能包含 {{}} 或 {{ssm:*parameter-name*}}。

Parameter Store支援三種參數：String、StringList 和 SecureString。

除了一個例外狀況，當您建立或更新參數時，將參數值輸入為純文字，並且 Parameter Store 不對輸入的文字執行驗證。但是，對於 String 參數，您可以將資料類型指定為 aws:ec2:image，而 Parameter Store 便會驗證您輸入的數值是否為 Amazon EC2 AMI 的正確格式；例如：ami-12345abcdeEXAMPLE。

參數類型：String

根據預設，String 參數是由您輸入的任何文字區塊組成。例如：

- abc123
- Example Corp
-

參數類型：StringList

StringList 參數包含以逗號分隔的數值清單，如下列範例所示。

```
Monday,Wednesday,Friday
```

```
CSV,TSV,CLF,ELF,JSON
```

參數類型：SecureString

SecureString 參數是需要以安全方式存放和參考的所有敏感資料。如果您有資料不希望使用者更改或以純文字參考，例如密碼或授權金鑰，請使用 SecureString 資料類型建立這些參數。

> **Important**
>
> 請勿在 String 或 StringList 參數中存放敏感資料。對於所有必須保持加密的敏感資料，請僅使用 SecureString 參數類型。

如需詳細資訊，請參閱使用 建立 SecureString 參數 AWS CLI。

在以下情況中，我們建議使用 SecureString 參數。

- 您想要跨 使用資料/參數， AWS 服務 而不在命令、函數、代理程式日誌或 CloudTrail 日誌中以純文字形式公開這些值。
- 您希望控制哪些使用者可以存取敏感資料。
- 您希望能夠在敏感資料被存取時進行稽核 (CloudTrail)。
- 您希望將敏感資料加密，而且您希望使用自己的加密金鑰來管理存取。

Important

僅加密 SecureString 參數的值。參數名稱、說明和其他屬性不會加密。

您可以針對您想要加密的文字資料使用 SecureString 參數類型，例如密碼、應用程式秘密、機密組態資料，或您想要保護的任何其他資料類型。 SecureString 資料會使用 AWS KMS 金鑰加密和解密。您可以使用 提供的預設 KMS 金鑰， AWS 或建立和使用您自己的 KMS 金鑰 AWS KMS key。(如果想限制使用者對 SecureString 參數的存取，請使用您自己的 AWS KMS key 。如需詳細資訊，請參閱 使用 AWS 預設金鑰和客戶受管金鑰的 IAM 許可。)

您也可以將SecureString參數與其他 參數搭配使用 AWS 服務。在下列範例中，Lambda 函數使用 GetParameters API 擷取 SecureString 參數。

```
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 加密和定價

如果您在建立SecureString參數時選擇 參數類型，Systems Manager 會使用 AWS KMS 來加密參數值。

> **Important**
>
> Parameter Store 只支援對稱加密 KMS 金鑰。您無法使用非對稱加密 KMS 金鑰來加密您的參數。如需判斷 KMS 金鑰為對稱或非對稱的說明，請參閱《AWS Key Management Service 開發人員指南》中的識別對稱鍵和非對稱金鑰。

Parameter Store 建立SecureString參數無需支付 的費用，但使用 AWS KMS 加密的費用確實適用。如需相關資訊，請參閱 AWS Key Management Service 定價。

如需 AWS 受管金鑰 和客戶受管金鑰的詳細資訊，請參閱《 AWS Key Management Service 開發人員指南》中的AWS Key Management Service 概念。如需 Parameter Store和 AWS KMS 加密的詳細資訊，請參閱 AWS Systems ManagerParameter Store如何使用 AWS KMS。

> **Note**
>
> 若要檢視 AWS 受管金鑰，請使用 AWS KMS DescribeKey操作。此 AWS Command Line Interface (AWS CLI) 範例使用 DescribeKey 來檢視 AWS 受管金鑰。
>
> ```
> aws kms describe-key --key-id alias/aws/ssm
> ```

詳細資訊

- 在 Parameter Store 中建立 SecureString 參數並將節點加入網域 (PowerShell)
- 使用 Parameter Store 安全地存取 CodeDeploy 中的秘密與組態資料
- 關於 Amazon EC2 Systems Manager Parameter Store 的有趣文章

參數大小限制

Parameter Store 根據您使用的參數層，參數值有不同的大小限制：

- 標準參數：值大小上限為 4 KB
- 進階參數：值大小上限為 8 KB

如果您需要存放大於 4 KB 的參數值，則必須使用進階參數層。進階參數提供額外的功能，但 AWS 您的帳戶會產生費用。如需參數層及其功能的詳細資訊，請參閱 管理參數層。

如需Parameter Store配額和限制的完整清單，請參閱《 AWS 一般參考》中的AWS Systems Manager 端點和配額。

設定 Parameter Store

在 中設定參數之前Parameter Store， 中的工具 AWS Systems Manager會先設定 AWS Identity and Access Management (IAM) 政策，為帳戶中的使用者提供執行您指定動作的許可。

本節包含如何使用 IAM 主控台手動設定這些政策的資訊，以及如何將他們指派給使用者及使用者群組。您也可以建立和指派政策，來控制可在受管節點上執行何種參數動作。

本節也包含如何建立 Amazon EventBridge 規則，讓您接收 Systems Manager 參數變更通知的資訊。您也可以 AWS 根據 中的變更，使用 EventBridge 規則在 中叫用其他動作Parameter Store。

目錄

- 使用 IAM 政策限制對 Parameter Store 參數的存取
- 管理參數層
- 增加或重設 Parameter Store 輸送量
- 根據 Parameter Store 事件設定通知或觸發動作

使用 IAM 政策限制對 Parameter Store 參數的存取

您可以使用 AWS Identity and Access Management (IAM) 限制對 AWS Systems Manager 參數的存取。具體而言，您可以建立 IAM 政策以限制存取以下 API 操作：

- DeleteParameter
- DeleteParameters
- DescribeParameters
- GetParameter
- GetParameters
- GetParameterHistory

- GetParametersByPath
- PutParameter

使用 IAM 政策限制對 Systems Manager 參數的存取時，我們建議您建立和使用限制性 IAM 政策。例如，以下政策可讓使用者為有限的一組資源呼叫 `DescribeParameters` 和 `GetParameters` API 操作。這表示使用者可以取得相關資訊，以及使用以 `prod-*` 開頭的所有參數。

```
{
    "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

如果使用者擁有路徑的存取權限，則該使用者可存取該路徑的所有層級。例如，如果使用者擁有存取路徑 `/a` 的許可，則該使用者也可以存取 `/a/b`。雖然使用者在 IAM 中被明確拒絕存取參數 `/a/b`，但他們仍能夠以遞迴方式呼叫 `/a` 的 `GetParametersByPath` API 操作，並檢視 `/a/b`。

針對信任的管理員，您可以使用類似以下範例的政策，提供所有 Systems Manager 參數 API 操作的完整存取。此政策可讓使用者完整存取所有以 `dbserver-prod-*` 開頭的生產參數。

```
{
    "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-2:123456789012:parameter/dbserver-prod-*"
        },
        {
            "Effect": "Allow",
            "Action": "ssm:DescribeParameters",
            "Resource": "*"
        }
    ]
}
```

拒絕許可

每個 API 都是唯一的，且具有不同的操作和許可，您可以單獨允許或拒絕它們。任何政策中的明確拒絕會覆寫任何允許。

> **Note**
>
> default AWS Key Management Service (AWS KMS) 金鑰具有 內所有 IAM 主體的`Decrypt`許可 AWS 帳戶。如果您希望對帳戶中的 `SecureString` 參數擁有不同的存取級別，不建議您使用預設金鑰。

如果您希望擷取參數值的所有 API 操作具有相同的行為，那麼您可以在政策中使用 `GetParameter*` 等模式。下列範例會顯示如何拒絕以 `prod-*` 開頭的所有參數的 `GetParameter`、`GetParameters`、`GetParameterHistory` 以及 `GetParametersByPath`。

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

```
        {
            "Effect": "Deny",
            "Action": [
                "ssm:GetParameter*"
            ],
            "Resource": "arn:aws:ssm:us-east-2:123456789012:parameter/prod-*"
        }
    ]
}
```

下列範例顯示如何拒絕某些命令，同時允許使用者在以 prod-* 開頭的所有參數中執行其他命令。

```
{
    "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-2:123456789012:parameter/prod-*"
        }
    ]
}
```

> **Note**
>
> 參數歷程包含所有參數版本，包括目前的參數版本。因此，如果 GetParameter、GetParameters 和 GetParameterByPath 的使用者許可遭到拒絕、但

允許 GetParameterHistory 的許可，則透過使用 GetParameterHistory，他們可以看到當前參數，包括 SecureString 參數。

僅允許特定參數在節點上執行

您可以控制存取，讓受管節點僅可執行您指定的參數。

如果您在建立SecureString參數時選擇 參數類型，Systems Manager 會使用 AWS KMS 來加密參數值。 會使用 AWS 受管金鑰 或客戶受管金鑰來 AWS KMS 加密該值。如需 AWS KMS 和 的詳細資訊 AWS KMS key，請參閱 AWS Key Management Service 開發人員指南。

您可以從 AWS 受管金鑰 執行下列命令來檢視 AWS CLI。

```
aws kms describe-key --key-id alias/aws/ssm
```

下列範例允許節點僅取得以 prod- 開頭的參數的參數值。如果參數為 SecureString 參數，則節點會使用 AWS KMS來解密字串。

> **Note**
>
> 執行個體政策 (例如以下範例) 會指派給 IAM 中的執行個體角色。如需設定存取 Systems Manager 功能的詳細資訊，包括如何將政策指派給使用者和執行個體，請參閱 使用 Systems Manager 管理 EC2 執行個體。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameters"
            ],
            "Resource": [
                "arn:aws:ssm:us-east-2:123456789012:parameter/prod-*"
            ]
        },
        {
            "Effect": "Allow",
```

```
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-2:123456789012:key/4914ec06-e888-4ea5-
a371-5b88eEXAMPLE"
            ]
        }
    ]
}
```

使用 AWS 預設金鑰和客戶受管金鑰的 IAM 許可

Parameter Store SecureString 參數會使用 AWS KMS 金鑰加密和解密。您可以選擇使用 AWS KMS key 或 提供的預設 KMS 金鑰來加密SecureString參數 AWS。

使用客戶受管金鑰時，授予使用者存取參數或參數路徑的 IAM 政策必須提供金鑰的明確 kms:Encrypt 許可。例如，以下政策允許使用者建立、更新和檢視以 開頭的SecureString參數，其在指定的 AWS 區域 和 prod-中 AWS 帳戶。

```
{
    "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"
  1
            ],
            "Resource": [
```

```
                "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-12345EXAMPLE"
            ]
        }
    ]
}
```

[1] 使用指定的客戶受管金鑰建立加密的進階參數時，需要 kms:GenerateDataKey 許可。

相反地，客戶帳戶中的所有使用者都可以存取預設的 AWS 受管金鑰。如果您使用此預設金鑰加密 SecureString 參數，但不希望使用者使用 SecureString 參數，則其 IAM 政策必須明確拒絕對預設金鑰的存取，如下列政策範例所示。

> **Note**
>
> 您可以在 AWS 受管金鑰頁面的 AWS KMS 主控台中找到預設金鑰的 Amazon Resource Name (ARN)。預設金鑰是別名資料欄中用 aws/ssm 標識的金鑰。

```
{
    "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"
            ]
        }
    ]
}
```

如果您需要對帳戶中的 SecureString 參數進行細微的存取控制，應該使用客戶受管金鑰來保護和限制對這些參數的存取。我們也建議使用 AWS CloudTrail 來監控SecureString參數活動。

如需詳細資訊，請參閱下列主題：

- 《IAM 使用者指南》中的政策評估邏輯
- 使用《AWS Key Management Service 開發人員指南》中的金鑰政策 AWS KMS
- 如需詳細資訊，請參閱《AWS CloudTrail 使用者指南》中的使用 CloudTrail 事件歷史記錄檢視事件。

管理參數層

Parameter Store中的工具 AWS Systems Manager包含標準參數和進階參數。您可以將參數個別設定為使用標準參數層級 (預設層級) 或進階參數層級。

您可以將標準參數變更為進階參數，但您無法將進階參數恢復為標準參數。將進階參數還原為標準參數會造成系統將參數的大小從 8 KB 截斷為 4 KB，從而造成資料遺失。還原也會移除連接到參數的任何政策。以外，進階參數使用的加密形式與標準參數不同。如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的 AWS Systems Manager Parameter Store 如何使用 AWS KMS。

如果您不再需要進階參數，或如果您不希望再支付費用，將它刪除，並將它重新建立為新的標準參數。

下表說明層級之間的差異。

	標準	Advanced (進階)
允許的參數總數 （每個 AWS 帳戶 和 AWS 區域)	10,000	100,000
參數值的大小上限。	4 KB	8 KB
是否可使用參數政策	否	是 如需詳細資訊，請參閱在 Parameter Store 中指派參數政策。
成本	不收取其他費用	需支付費用

	標準	Advanced (進階)
		如需詳細資訊，請參閱 AWS Systems ManagerParameter Store 的定價。

主題

- 指定預設參數層
- 將標準參數變更為進階參數

指定預設參數層

在建立或更新參數的請求中 (即 `PutParameter` 操作)，您可以指定要在請求中使用的參數層。以下是使用 AWS Command Line Interface (AWS CLI) 的範例。

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"
```

當您在請求中指定層時，Parameter Store 會根據您的請求建立或更新參數。但是，如果您沒有在請求中明確指定層，Parameter Store 預設層設定會決定要在哪個層建立參數。

標準參數層是在您開始使用 Parameter Store 時的預設層。若您使用進階參數層，您可以指定以下其中一項做為預設：

- 進階：使用此選項，參數存放區會將所有請求評估為進階參數。

- Intelligent-Tiering：使用此選項，Parameter Store 會評估每個請求，以判斷參數是標準或進階。

  若請求並未包含任何需要進階參數的選項，則參數會在標準參數層中建立。若請求中包含一或多個需要進階參數的選項，Parameter Store 便會在進階參數層中建立參數。

Intelligent-Tiering 的優點

以下是您可能會選擇將 Intelligent-Tiering 做為預設層的理由。

成本控制 – Intelligent-Tiering 可透過一律建立標準參數 (除非絕對需要進階參數) 來協助控制您的參數相關成本。

自動升級到進階參數層 – 當您變更程式碼而需要將標準參數升級到進階參數時，Intelligent-Tiering 會為您處理轉換。您不需要變更程式碼即可處理升級。

以下是一些自動升級的範例：

- 您的 AWS CloudFormation 範本會在執行時佈建許多參數。當此程序導致您達到標準參數層中的 10，000 個參數配額時，Intelligent-Tiering 會自動將您升級至進階參數層，而且您的 AWS CloudFormation 程序不會中斷。
- 您將憑證值存放在參數中，定期輪換憑證值，且內容小於標準參數層的 4 KB 配額。若替換用的憑證值超過 4 KB，Intelligent-Tiering 會自動將參數升級至進階參數層。
- 您希望將多個現有的標準參數與參數政策建立關聯，但參數政策需要進階參數層。Intelligent-Tiering 會自動將參數升級至進階參數層，而不需在所有呼叫中包含 `--tier Advanced` 選項來更新參數。Intelligent-Tiering 選項會在每次滿足進階參數層的條件時，將參數從標準升級至進階。

需要進階參數的選項包含下列項目：

- 參數的內容大小超過 4 KB。
- 參數使用參數政策。
- AWS 帳戶 目前的 中已存在超過 10，000 個參數 AWS 區域。

預設層選項

您可以指定做為預設的層選項包括下列項目。

- 標準 - 標準參數層是在您開始使用 Parameter Store 時的預設層。使用標準參數層，您可以為 AWS 區域 中的每個參數建立 10，000 個參數 AWS 帳戶。每個參數的內容大小最大可以等於 4 KB。標

準參數不支援參數政策。使用標準參數層無須另外付費。選擇 Standard (標準) 做為預設層，表示 Parameter Store 一律會嘗試為沒有指定層的請求建立標準參數。

- 進階 – 使用進階參數層，為 AWS 區域 中的每個參數建立最多 100，000 個參數 AWS 帳戶。每個參數的內容大小最大可以等於 8 KB。進階參數支援參數政策。若要共用參數，該參數必須位於進階參數層。使用進階參數層需要付費。如需詳細資訊，請參閱 AWS Systems ManagerParameter Store 的定價。選擇 Advanced (進階) 做為預設層，表示 Parameter Store 一律會嘗試為沒有指定層的請求建立進階參數。

> **Note**
>
> 當您選擇進階參數層時，明確授權 AWS 向您的帳戶收取任何您所建立進階參數的費用。

- Intelligent-Tiering - Intelligent-Tiering 選項可讓 Parameter Store 根據請求的內容，決定是要使用標準參數層還是進階參數層。例如，如果您執行 命令來建立內容低於 4 KB 的參數，且 AWS 區域 中的目前參數少於 10，000 個 AWS 帳戶，而且您未指定參數政策，則會建立標準參數。如果您執行命令來建立內容超過 4 KB 的參數，則 中目前的 參數已超過 AWS 區域 10，000 AWS 帳戶個，或者您指定參數政策，則會建立進階參數。

> **Note**
>
> 當您選擇 Intelligent-Tiering 時，請明確授權 AWS 向您的 帳戶收取您建立的任何進階參數的費用。

您可以隨時變更 Parameter Store 預設層設定。

設定許可來指定Parameter Store預設層

Parameter Store 執行下列其中一項作業，以確認您在 AWS Identity and Access Management (IAM) 中具有變更 中預設參數層的許可：

- 確定將 `AdministratorAccess` 政策連接到您的 IAM 實體 (如使用者、群組或角色)。
- 確認您具備使用下列 API 操作變更預設層設定的許可：
  - GetServiceSetting
  - UpdateServiceSetting
  - ResetServiceSetting

將下列許可授予 IAM 實體，以允許使用者檢視和變更 AWS 帳戶中特定 AWS 區域 內參數的預設層設定。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetServiceSetting"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:UpdateServiceSetting"
            ],
            "Resource": "arn:aws:ssm:region:account-id:servicesetting/ssm/parameter-
store/default-parameter-tier"
        }
    ]
}
```

管理員可以指派下列許可，以指定唯讀許可。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetServiceSetting"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Deny",
            "Action": [
                "ssm:ResetServiceSetting",
                "ssm:UpdateServiceSetting"
            ],
            "Resource": "*"
```

```
        }
    ]
}
```

若要提供存取權，請新增權限至您的使用者、群組或角色：

- 中的使用者和群組 AWS IAM Identity Center：

  建立權限合集。請按照 AWS IAM Identity Center 使用者指南 中的 建立權限合集 說明進行操作。
- 透過身分提供者在 IAM 中管理的使用者：

  建立聯合身分的角色。遵循「IAM 使用者指南」的為第三方身分提供者 (聯合) 建立角色中的指示。
- IAM 使用者：
  - 建立您的使用者可擔任的角色。請按照「IAM 使用者指南」的為 IAM 使用者建立角色中的指示。
  - (不建議) 將政策直接附加至使用者，或將使用者新增至使用者群組。請遵循 IAM 使用者指南的新增許可到使用者 (主控台) 中的指示。

使用主控台指定或變更 Parameter Store 預設方案

下列程序說明如何使用 Systems Manager 主控台來指定或變更目前 AWS 帳戶 和 的預設參數層 AWS 區域。

> **Tip**
>
> 如果您尚未建立參數，您可以使用 AWS Command Line Interface (AWS CLI) 或 AWS Tools for Windows PowerShell 來變更預設參數層。如需詳細資訊，請參閱 使用 指定或變更 Parameter Store預設層 AWS CLI 及 指定或變更Parameter Store預設層 (PowerShell)。

指定或變更 Parameter Store 預設層

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Parameter Store。
3. 選擇 Settings (設定) 標籤。
4. 選擇 Change default tier (變更預設層)。
5. 選擇下列其中一個選項。

- 標準
- Advanced (進階)
- Intelligent-Tiering

如需這些選項的資訊，請參閱 指定預設參數層。

6. 檢閱訊息，然後選擇 Confirm (確認)。

若您稍後希望變更預設層設定，請重複此程序，並指定不同的預設層選項。

使用 指定或變更Parameter Store預設層 AWS CLI

下列程序說明如何使用 AWS CLI 來變更目前 AWS 帳戶 和 的預設參數層設定 AWS 區域。

使用 AWS CLI指定或變更 Parameter Store 預設層

1. 開啟 AWS CLI 並執行下列命令，以變更 AWS 區域 中特定 的預設參數層設定 AWS 帳戶。

```
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 代表 支援的 AWS 區域 識別符 AWS Systems Manager，例如us-east-2美國東部（俄亥俄）區域。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

tier-option 值包含 Standard、Advanced 和 Intelligent-Tiering。如需這些選項的資訊，請參閱 指定預設參數層。

如果命令成功，則無輸出訊息。

2. 執行下列命令，以檢視目前 AWS 帳戶 和 Parameter Store中 的目前預設參數層服務設定 AWS 區域。

```
aws ssm get-service-setting --setting-id arn:aws:ssm:region:account-
id:servicesetting/ssm/parameter-store/default-parameter-tier
```

系統會傳回與以下相似的資訊。

```
{
    "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"
    }
}
```

若您希望再次變更預設層設定，請重複此程序，並指定不同的 `SettingValue` 選項。

指定或變更Parameter Store預設層 (PowerShell)

下列程序說明如何使用 Tools for Windows PowerShell 來變更 Amazon Web Services AWS 區域 帳戶中特定 的預設參數層設定。

使用 PowerShell 指定或變更 Parameter Store 預設層

1. AWS 區域 使用 AWS Tools for PowerShell (Tools for PowerShell) 變更目前 AWS 帳戶 和 中的 Parameter Store預設層。

   ```
   Update-SSMServiceSetting -SettingId "arn:aws:ssm:region:account-id:servicesetting/
   ssm/parameter-store/default-parameter-tier" -SettingValue "tier-option" -
   Region region
   ```

   region 代表 支援的 AWS 區域 識別符 AWS Systems Manager，例如`us-east-2`美國東部（俄亥俄）區域。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

   tier-option 值包含 `Standard`、`Advanced` 和 `Intelligent-Tiering`。如需這些選項的資訊，請參閱 指定預設參數層。

   如果命令成功，則無輸出訊息。
2. 執行下列命令，以檢視目前 AWS 帳戶 和 Parameter Store中 的目前預設參數層服務設定 AWS 區域。

```
Get-SSMServiceSetting -SettingId "arn:aws:ssm:region:account-id:servicesetting/ssm/
parameter-store/default-parameter-tier" -Region region
```

region 代表 支援的 AWS 區域 識別符 AWS Systems Manager，例如us-east-2美國東部（俄亥俄）區域。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

系統會傳回與以下相似的資訊。

```
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
```

若您希望再次變更預設層設定，請重複此程序，並指定不同的 SettingValue 選項。

將標準參數變更為進階參數

請使用下列步驟，將現有的標準參數變更為進階參數。如需有關如何建立新的進階參數的資訊，請參閱 在 Systems Manager 中建立 Parameter Store 參數。

若要將標準參數變更為進階參數

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Parameter Store。
3. 選擇參數，然後選擇 Edit (編輯)。
4. 對於 Description (說明)，請輸入有關此參數的資訊。
5. 選擇 Advanced (進階)。
6. 對於 Value (值)，請輸入此參數的值。進階參數的最大值限制為 8 KB。
7. 選擇 Save changes (儲存變更)。

增加或重設 Parameter Store 輸送量

提高Parameter Store輸送量會增加 中Parameter Store工具 AWS Systems Manager每秒可處理的最大交易數 (TPS)。增加的輸送量可讓您以更大的流量操作 Parameter Store，以支援需要並行存取多種參數的應用程式和工作負載。您可以將配額提高到 Settings (設定) 索引標籤中的最大輸送量。

Parameter Store 輸送量設定適用於目前 AWS 帳戶 和 中所有 IAM 使用者建立的所有交易 AWS 區域。輸送量設定適用於標準和進階參數。

> **Note**
>
> 一般而言，更新會立即顯示在 Service Quotas 中。在極少數情況下，最多可能需要 24 小時才能反映更新。

如需有關輸送量預設值上限與限制上限的詳細資訊，請參閱 AWS Systems Manager endpoints and quotas。

增加輸送量配額會對您的 產生費用 AWS 帳戶。如需詳細資訊，請參閱AWS Systems Manager 定價。

主題

- 設定許可以變更 Parameter Store 輸送量
- 使用主控台增加或重設輸送量
- 使用 增加或重設輸送量 AWS CLI
- 提高或重設輸送量 (PowerShell)

設定許可以變更 Parameter Store 輸送量

請執行以下其中一項，驗證您擁有在 IAM 中變更 Parameter Store 輸送量的許可：

- 確定將 `AdministratorAccess` 政策連接到您的 IAM 實體 (使用者、群組或角色)。
- 確認您擁有使用以下 API 操作變更輸送量服務設定的許可：
  - GetServiceSetting
  - UpdateServiceSetting
  - ResetServiceSetting

將下列許可授予 IAM 實體，以允許使用者檢視和變更 AWS 帳戶中特定 AWS 區域 內參數的參數輸送量設定。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetServiceSetting"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:UpdateServiceSetting"
            ],
            "Resource": "arn:aws:ssm:region:account-id:servicesetting/ssm/parameter-
store/high-throughput-enabled"
        }
    ]
}
```

管理員可以指派下列許可，以指定唯讀許可。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetServiceSetting"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Deny",
            "Action": [
                "ssm:ResetServiceSetting",
                "ssm:UpdateServiceSetting"
            ],
            "Resource": "*"
```

```
        }
    ]
}
```

若要提供存取權，請新增權限至您的使用者、群組或角色：

- 中的使用者和群組 AWS IAM Identity Center：

  建立權限合集。請按照 AWS IAM Identity Center 使用者指南 中的 建立權限合集 說明進行操作。
- 透過身分提供者在 IAM 中管理的使用者：

  建立聯合身分的角色。遵循「IAM 使用者指南」的為第三方身分提供者 (聯合) 建立角色中的指示。
- IAM 使用者：
  - 建立您的使用者可擔任的角色。請按照「IAM 使用者指南」的為 IAM 使用者建立角色中的指示。
  - (不建議) 將政策直接附加至使用者，或將使用者新增至使用者群組。請遵循 IAM 使用者指南的新增許可到使用者 (主控台) 中的指示。

使用主控台增加或重設輸送量

以下程序示範如何使用 Systems Manager 主控台來增加 Parameter Store 每秒鐘可以為目前 AWS 帳戶 和 AWS 區域處理的交易數。此外，這會說明您不再需要更高的輸送量或不想再產生費用時如何還原為標準設定。

使用主控台增加或重設Parameter Store輸送量

> **Tip**
>
> 如果您尚未建立參數，您可以使用 AWS Command Line Interface (AWS CLI) 或 AWS Tools for Windows PowerShell 來增加輸送量。如需詳細資訊，請參閱 使用 增加或重設輸送量 AWS CLI 及 提高或重設輸送量 (PowerShell)。

1. 開啟 AWS Systems Manager 主控台，網址為 https：//https://console.aws.amazon.com/systems-manager/。
2. 在導覽窗格中，選擇 Parameter Store。
3. 選擇 Settings (設定) 標籤。
4. 若要提高輸送量，請選擇設定限制。

-或-

若要還原為預設限制，請選擇重設限制。

5. 如果您要提高限制，請執行下列動作：

   - 選取我接受變更此設定會在我的 AWS 帳戶上產生費用核取方塊。
   - 選擇 Set limit (設定限制)。

   -或-

   如果要將限制重設為預設值，請執行下列動作：

   - 選取我接受重設為預設輸送量限制會導致 Parameter Store 每秒處理較少的交易核取方塊。
   - 選擇重設限制。

使用 增加或重設輸送量 AWS CLI

下列程序說明如何使用 AWS CLI 來增加Parameter Store每秒可處理目前 AWS 帳戶 和 的交易數量 AWS 區域。您也可以還原為預設限制。

使用 增加Parameter Store輸送量 AWS CLI

1. 開啟 AWS CLI 並執行下列命令，以增加Parameter Store每秒可在目前 AWS 帳戶 和 中處理的交易 AWS 區域。

   ```
   aws ssm update-service-setting --setting-id arn:aws:ssm:region:account-
   id:servicesetting/ssm/parameter-store/high-throughput-enabled --setting-value true
   ```

   如果命令成功，則無輸出訊息。

2. 執行下列命令，以檢視目前 AWS 帳戶 和 Parameter Store中 的目前輸送量服務設定 AWS 區域。

   ```
   aws ssm get-service-setting --setting-id arn:aws:ssm:region:account-
   id:servicesetting/ssm/parameter-store/high-throughput-enabled
   ```

   系統會傳回與以下相似的資訊：

   ```
   {
       "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"
    }
}
```

如果您不再需要更高的輸送量，或如果您不希望再支付費用，您可以恢復為標準的設定。若要恢復您的設定，請執行以下命令。

```
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"
    }
}
```

提高或重設輸送量 (PowerShell)

以下程序示範如何使用 Tools for Windows PowerShell 來增加 Parameter Store 每秒鐘可以為目前 AWS 帳戶 和 AWS 區域處理的交易數。您也可以還原為預設限制。

使用 PowerShell 增加 Parameter Store 輸送量

1. AWS 區域 使用 AWS Tools for PowerShell (Tools for PowerShell) 增加目前 AWS 帳戶 和 中的 Parameter Store輸送量。

```
Update-SSMServiceSetting -SettingId "arn:aws:ssm:region:account-id:servicesetting/
ssm/parameter-store/high-throughput-enabled" -SettingValue "true" -Region region
```

如果命令成功，則無輸出訊息。

2. 執行下列命令，以檢視目前 AWS 帳戶 和 Parameter Store中 的目前輸送量服務設定 AWS 區域。

```
Get-SSMServiceSetting -SettingId "arn:aws:ssm:region:account-id:servicesetting/ssm/
parameter-store/high-throughput-enabled" -Region region
```

系統會傳回與類似以下的資訊：

```
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
```

如果您不再需要更高的輸送量，或如果您不希望再支付費用，您可以恢復為標準的設定。若要恢復您的設定，請執行以下命令。

```
Reset-SSMServiceSetting -SettingId "arn:aws:ssm:region:account-id:servicesetting/ssm/
parameter-store/high-throughput-enabled" -Region region
```

系統會傳回與以下相似的資訊：

```
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
```

根據 Parameter Store 事件設定通知或觸發動作

本節中的主題說明如何使用 Amazon EventBridge 和 Amazon Simple Notification Service (Amazon SNS) 來通知您 AWS Systems Manager 參數的變更。您可以建立 EventBridge 規則，在參數或參數標籤版本建立、更新或遭到刪除時通知您。盡可能發出事件。您可以接收與參數政策相關變更或狀態的通知，例如參數過期、參數即將過期，或是在指定的時間期間內並未發生變更。

> **Note**
>
> 參數政策適用於使用進階參數方案的參數。需支付費用。如需詳細資訊，請參閱 在 Parameter Store 中指派參數政策 及 管理參數層。

此部分的主題也說明了如何為特定參數事件，在目標上開啟其他動作。例如，您可以執行 AWS Lambda 函數，在參數過期或刪除時自動重新建立參數。您可以設定一個通知，在您的資料庫密碼更新時呼叫 Lambda 函數。Lambda 函數可以強制您的資料庫連線重設或以新的密碼重新連線。EventBridge 也支援執行Run Command命令和自動化執行，以及許多其他動作 AWS 服務。 Run Command和 自動化都是其中的工具 AWS Systems Manager。如需詳細資訊，請參閱《Amazon EventBridge 使用者指南》。

開始之前

建立任何您需要的資源，為您建立的規則指定目標動作。例如，若您建立的規格是用來傳送通知，請先建立 Amazon SNS 主題。如需詳細資訊，請參閱《Amazon Simple Notification Service 開發人員指南》中的 Amazon SNS 入門。

為參數及參數政策設定 EventBridge

本主題說明下列內容：

- 如何建立 EventBridge 規則，根據您 AWS 帳戶帳戶中一或多個參數上發生的事件目標。
- 如何建立 EventBridge 規則，根據您 AWS 帳戶帳戶中一或多個參數政策上發生的事件目標。當您建立進階參數時，您可以指定參數過期的時間、參數過期前何時要接收到通知，以及在參數沒有變更的情況下要等待多久的時間才傳送通知。您可以使用下列程序來設定這些事件的通知。如需詳細資訊，請參閱 在 Parameter Store 中指派參數政策 和 管理參數層。

為 Systems Manager 參數或參數政策設定 EventBridge

1. 前往 https://console.aws.amazon.com/events/ 開啟 Amazon EventBridge 主控台。

2. 在導覽窗格中，選擇 Rules (規則)，然後選擇 Create rule (建立規則)。

   -或-

   如果 EventBridge 首頁先開啟，請選擇 Create rule (建立規則)。

3. 輸入規則的名稱和描述。

   在同一個區域和同一個事件匯流排上，規則不能與另一個規則同名。

4. 針對事件匯流排，選擇要與此規則建立關聯的事件匯流排。如果您希望此規則在來自您自己的相符事件上啟動 AWS 帳戶，請選取預設 。當您帳戶中 AWS 服務 的 發出事件時，一律會前往您帳戶的預設事件匯流排。

5. 針對 Rule type (規則類型)，請保持選取預設的 Rule with an event pattern (具有事件模式的規則)。

6. 選擇 Next (下一步)。

7. 針對事件來源，請保持選取預設的 AWS 事件或 EventBridge 合作夥伴事件。您可以略過 Sample event (示範活動) 部分。

8. 針對 Event pattern (事件模式) 請執行下列動作：

   - 選擇 Custom patterns (JSON editor) (自訂模式 (JSON 編輯器))。
   - 針對 Event pattern (活動模式)，將以下內容之一貼到方塊中，取決於是為參數還是為參數政策建立規則：

     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"
        ]
    }
}
```

- 修改參數的內容，以及您希望採取的動作，參閱以下範例。

  Parameter

  在範例中，會在名為/Oncall 及 /Project/Teamlead 的參數更新時採取動作：

```
{
    "source": [
        "aws.ssm"
    ],
    "detail-type": [
        "Parameter Store Change"
    ],
    "detail": {
        "name": [
```

```
            "/Oncall",
            "/Project/Teamlead"
        ],
        "operation": [
            "Update"
        ]
    }
}
```

Parameter policy

在範例中，會在名為/OncallDuties 的參數過期或遭刪除時採取動作：

```
{
    "source": [
        "aws.ssm"
    ],
    "detail-type": [
        "Parameter Store Policy Action"
    ],
    "detail": {
        "parameter-name": [
            "/OncallDuties"
        ],
        "policy-type": [
            "Expiration"
        ]
    }
}
```

9. 選擇 Next (下一步)。
10. 針對 Target 1 (目標 1) 選取目標類型及資源。例如，如果您選擇 SNS topic (SNS 主題)，請選取 Topic (主題)。如果選取 CodePipeline，請在 Pipeline ARN 輸入管線 ARN。根據需要提供其他設定值。

> **Tip**
>
> 如果您的規則需要其他目標，請選擇 Add another target (新增另一個目標)。

11. 選擇 Next (下一步)。

12. (選用) 為規則輸入一或多個標籤。如需詳細資訊，請參閱《Amazon EventBridge 使用者指南》中的 Amazon EventBridge 標籤。
13. 選擇 Next (下一步)。
14. 選擇建立規則。

詳細資訊

- 使用參數標籤以在環境中輕鬆地更新組態
- 教學課程：在 Amazon EventBridge 使用者指南中使用 EventBridge 將事件轉送至 AWS Systems ManagerRun Command EventBridge
- 教學課程：Amazon EventBridge 使用者指南中的將 AWS Systems Manager 自動化設定為 EventBridge 目標 EventBridge

使用 Parameter Store

本節說明如何組織、建立及標記參數，以及如何建立不同版本的參數。

您可以使用 AWS Systems Manager 主控台、 AWS Command Line Interface (AWS CLI) AWS Tools for PowerShell、 和 AWS SDKs來建立和使用參數。如需參數的詳細資訊，請參閱什麼是參數？。

主題

- 在 Systems Manager 中建立 Parameter Store 參數
- 在 Systems Manager 中搜尋 Parameter Store 參數
- 在 Parameter Store 中指派參數政策
- 在 Parameter Store 中使用參數階層
- 防止存取 Parameter Store API 操作
- 在 Parameter Store 中使用參數標籤
- 在Parameter Store中使用參數版本
- 在 Parameter Store 中使用共用參數
- 在 Parameter Store 中透過 Run Command 命令來使用參數
- 使用 Parameter Store 中 Amazon Machine Image ID 的原生參數支援
- 從 Parameter Store 中刪除參數

在 Systems Manager 中建立 Parameter Store 參數

使用以下主題中的資訊，協助您使用 AWS Systems Manager 主控台、 AWS Command Line Interface (AWS CLI) 或 AWS Tools for Windows PowerShell (Tools for Windows PowerShell) 建立 Systems Manager 參數。

此區段展示如何在測試環境中，使用 Parameter Store 建立、存放和執行參數。它還示範如何與其他 Systems Manager 工具Parameter Store搭配使用 AWS 服務。如需詳細資訊，請參閱什麼是參數？

了解參數名稱的要求與限制

使用此主題中的資訊，在建立參數時協助您為參數名稱指定有效的值。

此資訊可補充 AWS Systems Manager API 參考中 PutParameter 主題的詳細資訊，它還提供有關 AllowedPattern (允許的模式)、Description (說明)、KeyId (金鑰 ID)、Overwrite (覆寫)、Type (類型) 以及 Value (值) 等值的資訊。

參數名稱的要求和條件約束包含下列項目：

- 區分大小寫：參數名稱一律區分大小寫。
- 空格：參數名稱不可包含空格。
- 有效字元：參數名稱只能包含下列符號和字母：`a-zA-Z0-9_.-`

  此外，斜線字元 (/) 用於在參數名稱中描述階層。例如：`/Dev/Production/East/Project-ABC/MyParameter`
- 有效的 AMI 格式：當您選擇 `aws:ec2:image` 做為 `String` 參數的資料類型時，您輸入的 ID 必須是有效的 AMI ID 格式 `ami-12345abcdeEXAMPLE`。
- 完整：當您在階層中建立或參考參數時，包含前置正斜線字元 (/)。如果您參考的參數屬於階層一部分，則指定整個階層路徑，包括初始斜線 (/)。
  - 完整參數名稱：`MyParameter1`、`/MyParameter2`、`/Dev/Production/East/Project-ABC/MyParameter`
  - 非完整參數名稱：`MyParameter3/L1`
- 長度：您指定的參數名稱長度上限為 1011 個字元。此 1011 個字元的計數包含 ARN 中在您指定名稱前面的字元，例如 中的 45 個字元`arn:aws:ssm:us-east-2:111122223333:parameter/`。
- 字首：參數名稱的字首不得為「`aws`」或「`ssm`」(無論大小寫)。例如，嘗試使用以下名稱建立參數將會失敗及例外狀況：
  - `awsTestParameter`
  - `SSM-testparameter`

- /aws/testparam1

> **Note**
>
> 當您在 SSM 文件、命令或指令碼中指定參數時，您在語法中包含 ssm。例如：{{ssm:*parameter_name*}} 及 {{ ssm:*parameter-name* }}，例如 {{ssm:MyParameter}} 及 {{ ssm:MyParameter }}.

- 唯一性：參數名稱在一個 AWS 區域中必須是唯一的。例如，Systems Manager 會將以下視為不同的參數，如果它們存在於相同的區域：
  - /Test/TestParam1
  - /TestParam1

  以下範例也都是唯一的：
  - /Test/TestParam1/Logpath1
  - /Test/TestParam1

  不過，如果位在同一區域，則以下範例不是唯一的：
  - /TestParam1
  - TestParam1
- 階層深度：如果您指定參數階層，階層深度最多為 15 個層級。您可以定義任何層級的參數。以下兩個範例在結構上都是有效的：
  - /Level-1/L2/L3/L4/L5/L6/L7/L8/L9/L10/L11/L12/L13/L14/parameter-name
  - parameter-name

  嘗試建立以下參數將會失敗並出現 HierarchyLevelLimitExceededException 例外狀況：
  - /Level-1/L2/L3/L4/L5/L6/L7/L8/L9/L10/L11/L12/L13/L14/L15/L16/parameter-name

> **Important**
>
> 如果使用者擁有路徑的存取權限，則該使用者可存取該路徑的所有層級。例如，如果使用者擁有存取路徑 /a 的許可，則該使用者也可以存取 /a/b。即使使用者在 參數 AWS Identity and Access Management (IAM) 中明確遭到拒絕存取/a/b，他們仍然可以遞迴呼叫 的 GetParametersByPath API 操作/a，並檢視 /a/b。

主題

- 使用主控台建立 Parameter Store 參數
- 使用 AWS CLI建立 Parameter Store 參數
- 使用 Tools for Windows PowerShell 建立 Parameter Store 參數

使用主控台建立 Parameter Store 參數

您可以使用 AWS Systems Manager 主控台來建立和執行 `String`、 `StringList`和 `SecureString` 參數類型。刪除參數後，請等待至少 30 秒以建立具有相同名稱的參數。

> **Note**
>
> 參數僅適用於建立參數 AWS 區域 的 。

以下程序將帶您演練使用 Parameter Store 主控台建立參數的過程。您可以在主控台中建立 `String`、`StringList` 和 `SecureString` 參數類型。

若要建立參數

1. 開啟 AWS Systems Manager 主控台，網址為 https：//https://console.aws.amazon.com/systems-manager/。
2. 在導覽窗格中，選擇 Parameter Store。
3. 選擇 Create parameter (建立參數)。
4. 在 Name (名稱) 方塊中，輸入階層和名稱。例如，輸入 **`/Test/helloWorld`**。

   如需參數階層的詳細資訊，請參閱在 Parameter Store 中使用參數階層。
5. 在 Description (描述) 方塊中輸入描述，以識別此參數為測試參數。
6. 對於 Parameter tier (參數層級)，請選擇 Standard (標準) 或 Advanced (進階)。如需關於進階參數的詳細資訊，請參閱 管理參數層。
7. 在 Type (類型) 中選擇 String、StringList 或 SecureString。

   - 如果您選擇 String (字串)，則會顯示 Data type (資料類型) 欄位。如果您要建立參數來保留 Amazon Machine Image (AMI) 的資源 ID，請選取 `aws:ec2:image`。否則，請將預設的 `text` 維持在選取狀態。
   - 如果您選擇 SecureString，將會顯示 KMS Key ID (KMS 金鑰 ID) 欄位。如果您未提供 AWS Key Management Service AWS KMS key ID、 AWS KMS key Amazon Resource Name

(ARN)、別名名稱或別名 ARN，則系統會使用 `alias/aws/ssm`，這是 Systems Manager AWS 受管金鑰 的 。如果您不想使用此金鑰，您可以使用客戶管理的金鑰。如需 AWS 受管金鑰和客戶受管金鑰的詳細資訊，請參閱 AWS Key Management Service 開發人員指南中的 AWS Key Management Service 概念。如需 Parameter Store和 AWS KMS 加密的詳細資訊，請參閱 AWS Systems ManagerParameter Store如何使用 AWS KMS。

> **Important**
>
> Parameter Store 只支援對稱加密 KMS 金鑰。您無法使用非對稱加密 KMS 金鑰來加密您的參數。如需判斷 KMS 金鑰為對稱或非對稱的說明，請參閱《AWS Key Management Service 開發人員指南》中的識別對稱鍵和非對稱金鑰。

- 在主控台使用 `key-id` 參數及客戶受管金鑰別名名稱或別名 ARN 來建立 `SecureString` 參數時，需要在別名前面指定字首 `alias/`。以下是 ARN 範例：

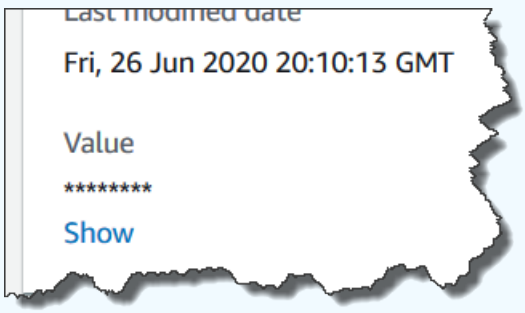
```
arn:aws:kms:us-east-2:123456789012:alias/abcd1234-ab12-cd34-ef56-abcdeEXAMPLE
```

以下是別名名稱範例：

```
alias/MyAliasName
```

8. 在 Value (數值) 方塊中輸入值。例如，輸入 **`This is my first parameter`** 或 **`ami-0dbf5ea29aEXAMPLE`**。

> **Note**
>
> 不能在其他參數的值中參考或巢套參數。參數值中不能包含 {{}} 或 {{ssm:*parameter-name*}}。
> 如果您選擇了 SecureString，當您稍後在參數概觀索引標籤中檢視參數時，預設會遮罩參數的值 ("******")，如下圖所示。選擇 Show (顯示)，以顯示參數值。



9. (選用) 在 Tags (標記) 區域中將一個或多個標記索引鍵/值對套用到該參數。

   標籤是您指派給資源的選用性中繼資料。標籤允許您以不同的方式 (例如用途、擁有者或環境) 將資源分類。例如，您可能想要標記 Systems Manager 參數，以識別其適用的資源類型、環境或該參數所參考的組態資料的類型。在這種情況下，您可以指定以下索引鍵/值組：

   - `Key=Resource,Value=S3bucket`
   - `Key=OS,Value=Windows`
   - `Key=ParameterType,Value=LicenseKey`
10. 選擇 Create parameter (建立參數)。
11. 在參數清單中，選擇您剛才建立的參數名稱。確認 Overview (概觀) 索引標籤上的詳細資訊。如果您建立 `SecureString` 參數，請選擇 Show (顯示) 以檢視未加密的值。

> **Note**
>
> 您無法將進階參數變更為標準參數。如果您不再需要進階參數，或如果您不希望再支付費用，將它刪除，並將它重新建立為新的標準參數。

使用 AWS CLI建立 Parameter Store 參數

您可以使用 AWS Command Line Interface (AWS CLI) 來建立 `String`、 `StringList`和 `SecureString` 參數類型。刪除參數後，請等待至少 30 秒以建立具有相同名稱的參數。

不能在其他參數的值中參考或巢套參數。參數值中不能包含 `{{}}` 或 `{{ssm:parameter-name}}`。

> **Note**
>
> 參數僅適用於建立參數 AWS 區域 的 。

主題

- 使用 建立String參數 AWS CLI
- 使用 建立 StringList 參數 AWS CLI
- 使用 建立 SecureString 參數 AWS CLI
- 使用 建立多行參數 AWS CLI

使用 建立**String**參數 AWS CLI

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

2. 請執行以下命令以建立 String-類參數。將每個########取代為您自己的資訊。

   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-

   執行下列命令，以建立包含 Amazon Machine Image (AMI) ID 做為參數值的參數。

   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 選項支援階層。如需有關階層的詳細資訊，請參閱在 Parameter Store 中使用參數階層。

只有在建立包含 AMI ID 的參數時，才需要指定 --data-type 選項。它會驗證您輸入的參數值是否為格式正確的 Amazon Elastic Compute Cloud (Amazon EC2) AMI ID。對於所有其他參數，預設資料類型為 text，可選擇指定一個值。如需詳細資訊，請參閱使用 Parameter Store 中 Amazon Machine Image ID 的原生參數支援。

> **Important**
>
> 如果成功，該命令將會傳回參數的版本號碼。例外狀況：如果您已將 aws:ec2:image 指定為資料類型，則回應中的新版本號碼並不表示參數值已經過驗證。如需詳細資訊，請參閱使用 Parameter Store 中 Amazon Machine Image ID 的原生參數支援。

此範例將兩個索引鍵/值對標籤新增至參數。

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\"}]
```

以下範例在名稱中使用參數階層來建立純文字 String 參數。傳回參數的版本號碼。如需參數階層的詳細資訊，請參閱[在 Parameter Store 中使用參數階層](#)。

Linux & macOS

不在階層中的參數

```
aws ssm put-parameter \
    --name "golden-ami" \
    --type "String" \
    --value "ami-12345abcdeEXAMPLE"
```

在階層中的參數

```
aws ssm put-parameter \
    --name "/amis/linux/golden-ami" \
    --type "String" \
    --value "ami-12345abcdeEXAMPLE"
```

Windows

不在階層中的參數

```
aws ssm put-parameter ^
    --name "golden-ami" ^
    --type "String" ^
    --value "ami-12345abcdeEXAMPLE"
```

在階層中的參數

```
aws ssm put-parameter ^
    --name "/amis/windows/golden-ami" ^
    --type "String" ^
    --value "ami-12345abcdeEXAMPLE"
```

3. 執行下列命令以檢視最新的參數值，並確認新參數的詳細資訊。

```
aws ssm get-parameters --names "/Test/IAD/helloWorld"
```

系統會傳回如下資訊。

```
{
    "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"
        }
    ]
}
```

執行以下命令，以變更參數值。傳回參數的版本號碼。

```
aws ssm put-parameter --name "/Test/IAD/helloWorld" --value "My updated 1st parameter"
 --type String --overwrite
```

執行以下命令，以檢視參數值的歷程記錄。

```
aws ssm get-parameter-history --name "/Test/IAD/helloWorld"
```

執行以下命令，以便在命令中使用此參數。

```
aws ssm send-command --document-name "AWS-RunShellScript" --parameters '{"commands":
["echo {{ssm:/Test/IAD/helloWorld}}"]}' --targets "Key=instanceids,Values=instance-ids"
```

如果您只想擷取參數值，請執行下列命令。

```
aws ssm get-parameter --name testDataTypeParameter --query "Parameter.Value"
```

如果您只想使用 `get-parameters` 來擷取參數值，請執行下列命令。

```
aws ssm get-parameters --names "testDataTypeParameter" --query "Parameters[*].Value"
```

執行以下命令，以檢視參數中繼資料。

```
aws ssm describe-parameters --filters "Key=Name,Values=/Test/IAD/helloWorld"
```

> **Note**
>
> 名稱 必須是大寫。

系統會傳回如下資訊。

```
{
    "Parameters": [
        {
            "Name": "helloworld",
            "Type": "String",
            "LastModifiedUser": "arn:aws:iam::123456789012:user/JohnDoe",
            "LastModifiedDate": 1494529763.156,
            "Version": 1,
            "Tier": "Standard",
            "Policies": []
        }
    ]
}
```

使用 建立 StringList 參數 AWS CLI

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

2. 請執行以下命令以建立參數。將每個########取代為您自己的資訊。

   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，該命令將會傳回參數的版本號碼。

此範例將兩個索引鍵/值組新增至參數。根據您的本機電腦的作業系統類型，執行以下其中一個命令。從區域 Windows 機器運行的版本包含從命令列工具執行命令所需的逸出字元 ("\"))。

以下是使用參數階層的 StringList 範例。

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 中的項目必須以逗號 (,) 分隔。您無法使用其他標點符號或特殊字元來逸出此清單中的項目。如果您有個參數值需要逗號，那麼請使用 String 資料類型。

3. 執行 get-parameters 命令，以確認參數的詳細資訊。例如：

```
aws ssm get-parameters --name "/IAD/ERP/Oracle/addUsers"
```

使用 建立 SecureString 參數 AWS CLI

使用下列程序建立 SecureString 參數。將每個########取代為您自己的資訊。

> **Important**
>
> 僅加密 SecureString 參數的值。參數名稱、說明和其他屬性不會加密。

> **Important**
>
> Parameter Store 只支援對稱加密 KMS 金鑰。您無法使用非對稱加密 KMS 金鑰來加密您的參數。如需判斷 KMS 金鑰為對稱或非對稱的說明，請參閱《AWS Key Management Service 開發人員指南》中的識別對稱鍵和非對稱金鑰。

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。
2. 執行以下其中一個命令，以建立使用 SecureString 資料類型的參數。

   Linux & macOS

   使用預設值建立**SecureString**參數 AWS 受管金鑰

```
aws ssm put-parameter \
    --name "parameter-name" \
    --value "parameter-value" \
    --type "SecureString"
```

   建立使用自訂受管金鑰的 **SecureString** 參數

```
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"
```

建立使用自訂 AWS KMS 金鑰的**SecureString**參數

```
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

使用預設值建立**SecureString**參數 AWS 受管金鑰

```
aws ssm put-parameter ^
    --name "parameter-name" ^
    --value "parameter-value" ^
    --type "SecureString"
```

建立使用自訂受管金鑰的 **SecureString** 參數

```
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"
```

建立使用自訂 AWS KMS 金鑰的**SecureString**參數

```
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"
```

如果您在帳戶和區域中使用 AWS 受管金鑰 金鑰 建立 SecureString 參數，則不必提供 --key-id 參數的值。

> **Note**
>
> 若要使用 AWS KMS key 指派給 AWS 帳戶 和 的 AWS 區域，請從 命令中移除 key-id 參數。如需 AWS KMS keys的詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的 AWS Key Management Service 概念。

若要使用客戶受管金鑰，而不是 AWS 受管金鑰 指派給您帳戶的 ，請使用 --key-id 參數指定金鑰。此參數支援下列 KMS 參數格式。

- 關鍵 Amazon Resource Name (ARN) 範例：

  ```
  arn:aws:kms:us-east-2:123456789012:key/key-id
  ```

- 別名 ARN 範例：

  ```
  arn:aws:kms:us-east-2:123456789012:alias/alias-name
  ```

- Key ID 範例：

  ```
  12345678-1234-1234-1234-123456789012
  ```

- 別名範例：

  ```
  alias/MyAliasName
  ```

您可以使用 AWS Management Console 或 API 來建立客戶受管 AWS KMS 金鑰。下列 AWS CLI 命令會在 AWS 區域 的目前 中建立客戶受管金鑰 AWS 帳戶。

```
aws kms create-key
```

使用以下格式的命令，以您剛建立的金鑰建立 SecureString 參數。

以下範例使用混淆代碼名稱 (3l3vat3131) 作為密碼參數，並使用 AWS KMS key。

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. 執行以下命令，以驗證參數的詳細資訊。

   如果您不指定 `with-decryption` 參數，或如果您指定了 `no-with-decryption` 參數，命令將會傳回加密的 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. 執行以下命令，以檢視參數中繼資料。

   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. 如果您不使用客戶受管 AWS KMS key，請執行下列命令以變更參數值。

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-

如果您使用客戶受管 AWS KMS key，請執行下列命令之一以變更參數值。

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. 執行以下命令，以檢視最新的參數值。

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. 執行以下命令，以檢視參數值的歷程記錄。

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**
>
> 您可以手動建立具有加密值的參數。在此案例中，因為值已加密，您無需選擇 SecureString 參數類型。如果您選擇 SecureString，您的參數將被加密兩次。

依預設，所有 SecureString 值都顯示為加密文字。若要解密SecureString值，使用者必須具有呼叫 AWS KMS 解密 API 操作的許可。如需有關設定 AWS KMS 存取控制的資訊，請參閱《AWS Key Management Service 開發人員指南》中的 AWS KMS的身分驗證與存取控制。

> **Important**
>
> 如果您變更用於加密參數之 KMS 金鑰的 KMS 金鑰別名，則也必須更新參數用來參考 AWS KMS的金鑰別名。這只適用於 KMS 金鑰別名；別名連接的金鑰 ID 會保持不變，除非您刪除整個金鑰。

使用 建立多行參數 AWS CLI

您可以使用 AWS CLI 建立具有換行符號的參數。使用分行符號來分割較長參數值中的文字以提高可讀性，或者更新網頁的多段落參數內容。使用分行字元 (例如 \n)，您可以將內容包含在 JSON 檔案中並使用 --cli-input-json 選項，如下列範例所示。

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。
2. 執行下列命令以建立多列參數。

   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
```

下列範例顯示 MultiLineParameter.json 檔案的內容。

```
{
    "Value": "<para>Paragraph One</para>\n<para>Paragraph Two</para>
\n<para>Paragraph Three</para>"
}
```

儲存的參數值存放如下。

```
<para>Paragraph One</para>
<para>Paragraph Two</para>
<para>Paragraph Three</para>
```

使用 Tools for Windows PowerShell 建立 Parameter Store 參數

您可以使用 AWS Tools for Windows PowerShell 來建立 String、 StringList和 SecureString 參數類型。刪除參數後，請等待至少 30 秒以建立具有相同名稱的參數。

不能在其他參數的值中參考或巢套參數。參數值中不能包含 {{}} 或 {{ssm:*parameter-name*}}。

> **Note**
>
> 參數僅適用於建立參數 AWS 區域 的 。

主題

- 建立字串參數 (Tools for Windows PowerShell)
- 建立 StringList 參數 (Tools for Windows PowerShell)

- 建立 SecureString 參數 (Tools for Windows PowerShell)

建立字串參數 (Tools for Windows PowerShell)

1. 如果您尚未安裝並設定 AWS Tools for PowerShell (Tools for Windows PowerShell)，請執行此作業。

   如需相關資訊，請參閱安裝 AWS Tools for PowerShell。

2. 執行下列命令，以建立包含純文字值的參數。將每個########取代為您自己的資訊。

```
Write-SSMParameter `
    -Name "parameter-name" `
    -Value "parameter-value" `
    -Type "String"
```

   -或-

   執行下列命令，以建立包含 Amazon Machine Image (AMI) ID 做為參數值的參數。

> **Note**
>
> 若要建立具有標籤的參數，請先建立 Service.model.tag 作為變數。請見此處範例。
>
> ```
> $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
```

   只有在建立包含 AMI ID 的參數時，才需要指定 `-DataType` 選項。對於所有其他參數，預設資料類型為 `text`。如需詳細資訊，請參閱使用 Parameter Store 中 Amazon Machine Image ID 的原生參數支援。

以下是使用參數階層的範例。

```
Write-SSMParameter `
    -Name "/IAD/Web/SQL/IPaddress" `
    -Value "99.99.99.999" `
    -Type "String" `
    -Tags $tag
```

3. 執行以下命令，以驗證參數的詳細資訊。

```
(Get-SSMParameterValue -Name "the-parameter-name-you-specified").Parameters
```

建立 StringList 參數 (Tools for Windows PowerShell)

1. 如果您尚未安裝並設定 AWS Tools for PowerShell (Tools for Windows PowerShell)，請執行此作業。

   如需相關資訊，請參閱安裝 AWS Tools for PowerShell。

2. 執行下列命令以建立 StringList 參數。將每個########取代為您自己的資訊。

> **Note**
>
> 若要建立具有標籤的參數，請先建立 Service.model.tag 作為變數。請見此處範例。
>
> ```
> $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，該命令將會傳回參數的版本號碼。

請見此處範例。

```
Write-SSMParameter `
    -Name "stringlist-parameter" `
    -Value "Milana,Mariana,Mark,Miguel" `
    -Type "StringList" `
    -Tags $tag
```

> **Note**
>
> StringList 中的項目必須以逗號 (,) 分隔。您無法使用其他標點符號或特殊字元來逸出此清單中的項目。如果您有個參數值需要逗號，那麼請使用 String 資料類型。

3. 執行以下命令，以驗證參數的詳細資訊。

```
(Get-SSMParameterValue -Name "the-parameter-name-you-specified").Parameters
```

建立 SecureString 參數 (Tools for Windows PowerShell)

建立 SecureString 參數之前，請先閱讀這種參數的需求。如需詳細資訊，請參閱使用 建立 SecureString 參數 AWS CLI。

> **Important**
>
> 僅加密 SecureString 參數的值。參數名稱、說明和其他屬性不會加密。

> **Important**
>
> Parameter Store 只支援對稱加密 KMS 金鑰。您無法使用非對稱加密 KMS 金鑰來加密您的參數。如需判斷 KMS 金鑰為對稱或非對稱的說明，請參閱《AWS Key Management Service 開發人員指南》中的識別對稱鍵和非對稱金鑰。

1. 如果您尚未安裝並設定 AWS Tools for PowerShell (Tools for Windows PowerShell)，請執行此作業。

   如需相關資訊，請參閱安裝 AWS Tools for PowerShell。
2. 請執行以下命令以建立參數。將每個########取代為您自己的資訊。

> **Note**
>
> 若要建立具有標籤的參數，請先建立 Service.model.tag 做為變數。請見此處範例。
>
> ```
> $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，該命令將會傳回參數的版本號碼。

> **Note**
>
> 若要使用 AWS 受管金鑰 指派給您帳戶的 ，請從 命令中移除 -KeyId 參數。

以下範例將混淆代碼名稱 (3l3vat3131) 使用於密碼參數和 AWS 受管金鑰。

```
Write-SSMParameter `
    -Name "/Finance/Payroll/3l3vat3131" `
    -Value "P@sSwW)rd" `
    -Type "SecureString"`
    -Tags $tag
```

3. 執行以下命令，以驗證參數的詳細資訊。

```
(Get-SSMParameterValue -Name "the-parameter-name-you-specified" -WithDecryption
 $true).Parameters
```

依預設，所有 SecureString 值都顯示為加密文字。若要解密SecureString值，使用者必須具有呼叫 AWS KMS 解密 API 操作的許可。如需有關設定 AWS KMS 存取控制的資訊，請參閱《AWS Key Management Service 開發人員指南》中的 AWS KMS的身分驗證與存取控制。

> **Important**
>
> 如果您變更用於加密參數之 KMS 金鑰的 KMS 金鑰別名，則也必須更新參數用來參考 AWS KMS的金鑰別名。這只適用於 KMS 金鑰別名；別名連接的金鑰 ID 會保持不變，除非您刪除整個金鑰。

在 Systems Manager 中搜尋 Parameter Store 參數

當您的帳戶中有大量參數時，可能很難一次找到僅一個或幾個參數的相關資訊。在此情況下，您可以使用篩選工具，根據所指定的搜尋條件，來搜尋需要相關資訊的項目。您可以使用 AWS Systems Manager 主控台、 AWS Command Line Interface (AWS CLI) AWS Tools for PowerShell、 或 DescribeParameters API 來搜尋參數。

主題

- 使用主控台搜尋參數
- 使用 AWS CLI搜尋參數

使用主控台搜尋參數

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Parameter Store。
3. 在搜尋方塊中選擇，然後選擇您要搜尋的方式。例如 Type 或 Name。
4. 為您選取的搜尋類型提供資訊。例如：
   - 如果要依據 Type 搜尋，請選擇 String、StringList 或 SecureString。
   - 如果要依據 Name 搜尋，請選擇 contains、equals 或 begins-with，然後輸入全部或部分參數名稱。

> **Note**
>
> 在主控台中，Name 的預設搜尋類型為 contains。

5. 按 Enter。

參數的清單會隨您的搜尋結果更新。

使用 AWS CLI搜尋參數

使用 `describe-parameters` 命令來檢視 AWS CLI中一或多個參數的資訊。

下列範例示範各種選項，可用來檢視 中參數的相關資訊 AWS 帳戶。如需這些選項的詳細資訊，請參閱《AWS Command Line Interface 使用者指南》中的 describe-parameters 一節。

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。
2. 以可反映在您的帳戶中所建立參數的值取代下列命令中的範例值。

   Linux & macOS

   ```
   aws ssm describe-parameters \
       --parameter-filters "Key=Name,Values=MyParameterName"
   ```

   Windows

   ```
   aws ssm describe-parameters ^
       --parameter-filters "Key=Name,Values=MyParameterName"
   ```

   > **Note**
   >
   > 對於 `describe-parameters`，Name 的預設搜尋類型為 Equals。在參數篩選器中，指定 "Key=Name,Values=*MyParameterName*" 與指定 "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**
>
> 在最後一個範例中，*key-id* 代表用來加密您帳戶中建立之SecureString參數的 AWS Key Management Service (AWS KMS) 金鑰 ID。或者，您可以輸入 **alias/aws/ssm** 來使用您帳戶的預設 AWS KMS 金鑰。如需詳細資訊，請參閱使用 建立 SecureString 參數 AWS CLI。

如果成功，此命令傳回的輸出會類似如下。

```
{
    "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 中指派參數政策

參數政策讓您可以將特定條件 (例如過期日期或存留時間) 指派給一個參數，以協助您管理一群不斷增長的參數。參數政策在強制您更新或刪除存放在 中的密碼和組態資料方面特別有用 Parameter Store。 AWS Systems ManagerParameter Store提供下列類型的政策：`Expiration`、`ExpirationNotification`和 `NoChangeNotification`。

> **Note**
>
> 若要實作密碼輪換生命週期，請使用 AWS Secrets Manager。您可以使用 Secrets Manager 在資料庫登入資料、API 金鑰和其他機密的整個生命週期輕鬆進行輪換、管理和擷取。如需詳細資訊，請參閱AWS Secrets Manager 《 使用者指南》中的什麼是 AWS Secrets Manager？。

Parameter Store 使用非同步、定期掃描，強制執行參數政策。建立政策之後，您不需要執行額外的動作，即可強制執行政策。Parameter Store 會依據您指定的條件，獨立執行由政策定義的動作。

> **Note**
>
> 參數政策適用於使用進階參數方案的參數。如需詳細資訊，請參閱管理參數層。

參數政策為一 JSON 陣列，如下表所示。當您建立新的進階參數時，可以指派政策，或藉由更新參數來指派政策。Parameter Store 支援以下類型的參數政策。

政策	詳細資訊	範例
Expiration	此政策會刪除參數。您可以透過使用 ISO_INSTANT 格式或 ISO_OFFSET_DATE_TIME 格式來指定特定的日期和時間。若要變更您希望參數被刪除的時間，請更新政策。更新參數不會影響過期日期或連接到它的政策。到達過期日期和時間時，Parameter Store 會刪除參數。 **Note** 此範例使用 ISO_INSTANT 格式。您也可以透過使用 ISO_OFFSET_DATE_TIME 格式來指定日期和時間。請見此處範例：2019-11-01T22:13:48.87+10:30:00 。	`{ "Type": "Expiration", "Version": "1.0", "Attributes": { "Timestamp": "2018-12-02T21:34:33.000Z" } }`
ExpirationNotification	此政策在 Amazon EventBridge (EventBridge) 中啟動事件，藉此通知您有關過期的資訊。使用此政策，您可以在過期時間到達之前接收到通知，以天或小時為單位。	`{ "Type": "ExpirationNotification", "Version": "1.0", "Attributes": { "Before": "15", "Unit": "Days" }`

政策	詳細資訊	範例
		} }

NoChangeNotification	如果在指定的時間段沒有修改參數，此政策會啟動 EventBridge 中的事件。例如，當密碼必須在一時間段內變更時，此一政策類型就很有用。
	
	此政策會讀取參數的 LastModifiedTime 屬性，以決定何時傳送通知。如果您變更或編輯參數，系統會依據 LastModifiedTime 的新值重設通知的時間間隔。
		{
		"Type": "NoChangeNotification",
		"Version": "1.0",
		"Attributes": {
		"After": "20",
		"Unit": "Days"
		}
		}

您可以將多個政策指派給一個參數。例如，您可以指派 Expiration 和 ExpirationNotification 政策，讓系統觸發 EventBridge 事件來通知您參數即將刪除。您最多可以將十 (10) 個政策指派給一個參數。

以下範例顯示 PutParameter API 請求語法，該請求會將四個政策指派給新的 SecureString 參數，名為 ProdDB3。

```
{
    "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"
            }
        }
    ]
}
```

將政策新增至現有參數

本節包含如何使用 AWS Systems Manager 主控台、 AWS Command Line Interface (AWS CLI) 和 將政策新增至現有參數的相關資訊 AWS Tools for Windows PowerShell 。如需關於如何建立包含政策的新參數的資訊，請參閱 在 Systems Manager 中建立 Parameter Store 參數。

主題

- 使用主控台將政策新增至現有參數
- 使用 將政策新增至現有參數 AWS CLI
- 將政策新增至現有參數 (Tools for Windows PowerShell)

使用主控台將政策新增至現有參數

請使用下列程序，以使用 Systems Manager 主控台，將政策新增至現有參數。

將政策新增至現有參數

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Parameter Store。
3. 選擇您要更新之參數旁的選項，以納入政策，然後選擇 Edit (編輯)。
4. 選擇 Advanced (進階)。
5. (選用) 在 Parameter policies (參數政策) 部分，請選擇 Enabled (啟用)。您可以為該參數指定過期日期和一個或多個通知政策。
6. 選擇 Save changes (儲存變更)。

> **Important**
>
> - Parameter Store 會保留參數的政策，直到您用新的政策覆寫該政策或將其移除。
> - 若要從現有參數中移除所有政策，請編輯參數，並使用括弧和大括號套用空的政策，如以下所示：`[{}]`
> - 如果您將新政策新增至已經有政策的參數，則 Systems Manager 會覆寫連接到該參數的政策。現有的政策會被刪除。如果您想要將政策新增至已經有一個或多個政策的參數，則複製並貼上原有的政策，輸入新的政策，然後儲存您的變更。

使用 將政策新增至現有參數 AWS CLI

請使用下列步驟，利用 AWS CLI將政策新增至現有參數。

將政策新增至現有參數

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。
2. 執行以下命令，將政策新增至現有參數。將每個########取代為您自己的資訊。

   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}]"
```

以下範例包含過期政策，該政策在 15 天之後會將參數刪除。此範例也包含通知政策，該政策會在參數被刪除的五 (5) 天以前產生 EventBridge 事件。最後，它包含 NoChangeNotification 政策，如果 60 天之後該參數沒有變更。此範例使用混淆代碼名稱 (3l3vat3131) 做為密碼參數，並使用 AWS Key Management Service AWS KMS key。如需詳細資訊 AWS KMS keys，請參閱《AWS Key Management Service 開發人員指南》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. 執行以下命令，以驗證參數的詳細資訊。將####取代為您自己的資訊。

   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 會保留參數的政策，直到您用新的政策覆寫該政策或將其移除。
- 若要從現有參數中移除所有的政策，請編輯該參數，並套用由括弧和大括號組成的空白政策。將每個########取代為您自己的資訊。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 "[{}]"
  ```

- 如果您將新政策新增至已經有政策的參數，則 Systems Manager 會覆寫連接到該參數的政策。現有的政策會被刪除。如果您想要將政策新增至已經有一個或多個政策的參數，則複製並貼上原有的政策，輸入新的政策，然後儲存您的變更。

將政策新增至現有參數 (Tools for Windows PowerShell)

請使用下列程序，以使用 Tools for Windows PowerShell，將政策新增至現有參數。將每個*#######*取代為您自己的資訊。

將政策新增至現有參數

1. 開啟 Tools for Windows PowerShell 並執行以下命令，以指定您的登入資料。您必須在 Amazon Elastic Compute Cloud (Amazon EC2) 中擁有管理員許可，或者您必須在 AWS Identity and Access Management (IAM) 中獲得適當的許可。

```
Set-AWSCredentials `
    -AccessKey access-key-name `
    -SecretKey secret-key-name
```

2. 執行以下命令，以設定 PowerShell 工作階段的區域。此範例會使用美國東部 (俄亥俄) 區域 (US-east-2)。

```
Set-DefaultAWSRegion `
    -Region us-east-2
```

3. 執行以下命令，將政策新增至現有參數。將每個*########*取代為您自己的資訊。

```
Write-SSMParameter `
    -Name "parameter name" `
    -Value "parameter value" `
    -Type "parameter type" `
    -Policies "[{policies-enclosed-in-brackets-and-curly-braces}]" `
    -Overwrite
```

以下的範例包含過期政策，該政策將在 2020 年 5 月 13 日的午夜 (GMT) 將參數刪除。此範例也包含通知政策，該政策會在參數被刪除的五 (5) 天以前產生 EventBridge 事件。最後，它包含 NoChangeNotification 政策，如果 60 天之後該參數沒有變更。此範例使用混淆代碼名稱 (3l3vat3131) 作為密碼參數，並使用 AWS 受管金鑰。

```
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. 執行以下命令，以驗證參數的詳細資訊。將####取代為您自己的資訊。

```
(Get-SSMParameterValue -Name "parameter name").Parameters
```

Important

- Parameter Store 會保留參數的政策，直到您用新的政策覆寫該政策或將其移除。
- 若要從現有參數中移除所有的政策，請編輯該參數，並套用由括弧和大括號組成的空白政策。例如：

  ```
  Write-SSMParameter `
      -Name "parameter name" `
      -Value "parameter value" `
      -Type "parameter type" `
      -Policies "[{}]"
  ```

- 如果您將新政策新增至已經有政策的參數，則 Systems Manager 會覆寫連接到該參數的政策。現有的政策會被刪除。如果您想要將政策新增至已經有一個或多個政策的參數，則複製並貼上原有的政策，輸入新的政策，然後儲存您的變更。

在 Parameter Store 中使用參數階層

以一般清單來管理數十或數百個參數不僅耗時且容易出錯。同時也很難識別任務的正確參數。這表示您可能會不小心使用錯誤的參數，或者可能會建立多個使用相同組態資料的參數。

您可以使用參數階層結構，來協助您整理和管理 參數。階層結構是一種參數名稱，其中包含您使用正斜線 (/) 定義的路徑。

主題

- 透過範例了解參數階層
- 查詢階層中的參數
- 使用 管理使用階層的參數 AWS CLI

透過範例了解參數階層

以下範例在名稱中使用三個階層來識別下列各項：

```
/Environment/Type of computer/Application/Data
```

```
/Dev/DBServer/MySQL/db-string13
```

您可以建立最多 15 個層級的階層。建議您建立階層以反映環境中現有的階層架構，如下範例所示：

- 您的持續整合與持續交付環境 (CI/CD 工作流程)

  ```
  /Dev/DBServer/MySQL/db-string
  ```

  ```
  /Staging/DBServer/MySQL/db-string
  ```

  ```
  /Prod/DBServer/MySQL/db-string
  ```
- 使用容器的應用程式

  ```
  /MyApp/.NET/Libraries/my-password
  ```
- 您的商業組織

  ```
  /Finance/Accountants/UserList
  ```

  ```
  /Finance/Analysts/UserList
  ```

  ```
  /HR/Employees/EU/UserList
  ```

參數階層將您建立參數的方式標準化，並讓您更輕鬆地隨時管理參數。參數階層也可以協助您識別設定任務的正確參數。這可協助您建立多個使用相同組態資料的參數。

您可以建立階層，讓您在不同的環境中共用參數，如以下在開發和臨時環境中使用密碼的範例所示。

/DevTest/MyApp/database/*my-password*

然後，您可以為生產環境建立唯一的密碼，如以下範例所示：

/prod/MyApp/database/*my-password*

您無需指定參數階層。您可以在第一層建立參數。這些稱為根參數。有關回溯相容性，在發佈階層之前於 Parameter Store 建立的所有參數皆為根參數。系統會將以下兩個參數視為根參數。

/parameter-name

parameter-name

查詢階層中的參數

使用階層的另一個好處是能夠使用 GetParametersByPath API 操作，來查詢在階層中某個層級下的所有參數。例如，如果您從 AWS Command Line Interface (AWS CLI) 執行下列命令，則系統會傳回Oncall關卡下的所有參數：

```
aws ssm get-parameters-by-path --path /Dev/Web/Oncall
```

輸出範例：

```
{
    "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"
        }
    ]
}
```

若要檢視階層中已解密的 SecureString 參數，請指定路徑和 --with-decryption 參數，如以下範例所示。

```
aws ssm get-parameters-by-path --path /Prod/ERP/SAP --with-decryption
```

使用 管理使用階層的參數 AWS CLI

此程序示範如何利用 AWS CLI，來使用參數和參數階層。

使用階層管理參數

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

2. 執行以下命令，以建立使用 allowedPattern 參數和 String 參數類型的參數。此範例中允許的模式表示參數值必須介於 1 位數到 4 位數。

   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
   ```

   該命令會傳回參數的版本號碼。

3. 執行以下命令，以 嘗試 以新的值覆寫您剛建立的參數。

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
```

系統返回以下錯誤，因為新的值不符合您在上個步驟指定的允許模式的要求。

```
An error occurred (ParameterPatternMismatchException) when calling the PutParameter
 operation: Parameter value, cannot be validated against allowedPattern: \d{1,4}
```

4. 執行以下其中SecureString一個命令，以建立使用 AWS 受管金鑰的 參數。此範例中允許的模式表示使用者可以指定任何字元，其值必須介於 8 到 20 個字元。

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. 執行以下命令，以建立更多使用先前步驟的階層結構的參數。

   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. 執行以下命令，以取得兩個參數的值。

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. 執行以下命令，查詢單一層級下的所有參數。

Linux & macOS

```
aws ssm get-parameters-by-path \
    --path "/MyService/Test"
```

Windows

```
aws ssm get-parameters-by-path ^
    --path "/MyService/Test"
```

8. 執行下列命令，以刪除兩個參數

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"
```

防止存取 Parameter Store API 操作

使用 Systems Manager for AWS Identity and Access Management (IAM) 政策支援的服務特定條件，您可以明確允許或拒絕存取 Parameter Store API 操作和內容。若使用這些條件，您只能允許組織中的

特定 IAM 實體 (使用者和角色) 呼叫特定 API 動作，或防止特定 IAM 實體執行這些動作。這包括透過 Parameter Store 主控台、 AWS Command Line Interface (AWS CLI) 和 SDKs執行的動作。

Systems Manager 目前支援特定於 Parameter Store 的三個條件。

主題

- 使用 ssm:Overwrite 防止變更現有參數
- 防止使用 ssm:Policies 建立或更新使用參數政策的參數
- 使用 ssm:Recursive 防止存取階層參數中的層級

使用 **ssm:Overwrite** 防止變更現有參數

使用 `ssm:Overwrite` 條件來控制 IAM 實體是否可以更新現有參數。

在下列範例政策中， `"Allow"`陳述式會授予在美國東部 （俄亥俄） 區域 (us-east-2) 的 AWS 帳戶 123456789012 中執行 `PutParameter` API 操作來建立參數的許可。

但是，`"Deny"` 陳述式會防止實體變更現有參數的值，因為 `PutParameter` 操作會明確拒絕 `Overwrite` 選項。因此，經指派此政策的實體可以建立參數，但無法變更現有參數。

```
{
    "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/*"
        }
    ]
}
```

防止使用 **ssm:Policies** 建立或更新使用參數政策的參數

使用 ssm:Policies 條件，控制實體是否可以建立包含參數政策的參數，並更新包含參數政策的現有參數。

在下列政策範例中， "Allow"陳述式會授予建立參數的一般許可，但 "Deny"陳述式會防止實體在美國東部（俄亥俄）區域 (us-east-2) 的 AWS 帳戶 123456789012 中建立或更新包含參數政策的參數。實體仍可以建立或更新未指派有參數政策的參數。

```
{
    "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:Policies": [
                        "true"
                    ]
                }
            },
            "Resource": "arn:aws:ssm:us-east-2:123456789012:parameter/*"
        }
    ]
}
```

使用 **ssm:Recursive** 防止存取階層參數中的層級

使用 ssm:Recursive 條件來控制 IAM 實體是否可以在階層參數中檢視或參考層級。您可以提供或限制對階層特定層級以外之所有參數的存取權。

在下列範例政策中，"Allow" 陳述式提供對美國東部 (俄亥俄) 區域 (us-east-2) 中 AWS 帳戶 123456789012 之 /Code/Departments/Finance/* 路徑裡所有參數之 Parameter Store 操作的存取權。

之後，"Deny" 陳述式會防止 IAM 實體檢視或擷取處於或低於 /Code/Departments/* 層級的參數資料。不過，實體仍可在該路徑中建立或更新參數。建構此範例以說明，在參數階層中以遞迴方式拒絕低於特定層級的存取，優先於相同政策中更寬鬆的存取。

```
{
    "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**
>
> 如果使用者擁有路徑的存取權限，則該使用者可存取該路徑的所有層級。例如，如果使用者擁有存取路徑 /a 的許可，則該使用者也可以存取 /a/b。除非使用者在 /b 參數的 IAM 中明確遭到拒絕存取，否則都是如此 (如上所示)。

在 Parameter Store 中使用參數標籤

參數標籤是使用者定義的別名，可協助您管理不同版本的參數。當您修改參數時， AWS Systems Manager 會自動儲存新版本，並將版本編號增加一個。如果參數有多個版本，標籤可協助您記住參數版本的目的。

例如，假設您有一個名為 `/MyApp/DB/ConnectionString` 的參數。此參數的值是測試環境中本機資料庫的 MySQL 伺服器連線字串。在您完成更新應用程式之後，您需要參數以使用生產資料庫的連線字串。您可變更 `/MyApp/DB/ConnectionString` 的值。Systems Manager 會使用新的連線字串來自動建立版本 2。為了協助您記住每個版本的目的，您為每個參數加上標籤。您為版本 1 加上標籤 Test，為版本 2 加上標籤 Production。

您可以將標籤從參數的一個版本移動至另一個版本。例如，如果您以新生產資料庫的連線字串建立 `/MyApp/DB/ConnectionString` 參數的版本 3，則可以將 Production 標籤從參數的版本 2 移動至參數的版本 3。

參數標籤是參數標記的輕量型替代方案。您的組織對於必須套用至不同 AWS 資源的標記可能有嚴格的指導方針。反之，標籤只是特定參數版本的文字關聯。

與標記類似，您可以使用標記查詢參數。如本節先前所述，如果您使用 GetParametersByPath API 操作來查詢您的參數集，您將可檢視所有使用相同標籤的特定參數版本的清單。

> **Note**
>
> 如果您執行的命令中指定的參數版本不存在，則命令會失敗。系統不會返回使用參數的最新或預設值。

標籤要求和限制

參數標籤有下列要求與限制：

- 參數的版本最多可以有 10 個標籤。

- 您無法將相同的標籤附加至相同參數的不同版本。例如，如果參數的版本 1 有標籤 Production，那麼您無法將 Production 連接至版本 2。
- 您可以將標籤從參數的一個版本移動至另一個版本。
- 您無法在建立參數時建立標籤。您必須將標籤附加至指定的參數版本。
- 如果您不再使用某個參數標籤，則可以將其移至不同的參數版本或將其刪除。
- 標籤最多可使用 100 個字元。
- 標籤可包含字母 (區分大小寫)、數字、句點 (.)、連字號 (-) 及底線 (_)。
- 標籤的開頭不可以是數字、「aws」或「ssm」(不區分大小寫)。如果標籤未符合上述要求，該標籤將不會附加至參數版本，系統會將它顯示在 `InvalidLabels` 清單中。

主題

- 透過主控台使用參數標籤
- 使用 參數標籤 AWS CLI

透過主控台使用參數標籤

本節說明如何使用 Systems Manager 主控台執行以下工作：

- 使用主控台建立參數標籤
- 使用主控台檢視已附加至參數的標籤
- 使用主控台移動參數標籤
- 使用主控台刪除參數標籤

使用主控台建立參數標籤

下列程序說明如何使用 Systems Manager 主控台，將標籤連接至現有參數的一個特定版本。您無法在建立參數時附加標籤。

將標籤連接至參數版本

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Parameter Store。
3. 選擇參數的名稱以開啟該參數的詳細資訊頁面。
4. 選擇 History (歷程記錄) 索引標籤。

5. 選擇您要附加標籤的參數版本。
6. 選擇 Manage labels (管理標籤)。
7. 選擇 Add new label (新增新標籤)。
8. 在文字方塊中輸入標籤名稱。若要新增更多標籤，請選擇 Add new label (新增新標籤)。您最多可以附加十個標籤。
9. 完成時，請選擇儲存變更。

使用主控台檢視已附加至參數的標籤

一個參數版本最多可以有 10 個標籤。下列程序說明如何使用 Systems Manager 主控台，檢視已附加至參數版本的所有標籤。

檢視已附加至參數版本的標籤

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Parameter Store。
3. 選擇參數的名稱以開啟該參數的詳細資訊頁面。
4. 選擇 History (歷程記錄) 索引標籤。
5. 找到您要檢視所有已附加標籤的參數版本。Labels (標籤) 欄位會顯示所有已附加至該參數版本的標籤。

使用主控台移動參數標籤

下列程序說明如何使用 Systems Manager 主控台，將參數標籤移動至相同參數的不同版本。

將標籤移動至不同的參數版本

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Parameter Store。
3. 選擇參數的名稱以開啟該參數的詳細資訊頁面。
4. 選擇 History (歷程記錄) 索引標籤。
5. 選擇您要移動標籤的參數版本。
6. 選擇 Manage labels (管理標籤)。

7. 選擇 Add new label (新增新標籤)。
8. 在文字方塊中輸入標籤名稱。
9. 完成時，請選擇儲存變更。

使用主控台刪除參數標籤

下列程序說明如何使用 Systems Manager 主控台來刪除一個或多個參數標籤。

若要從參數中刪除標籤

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Parameter Store。
3. 選擇參數的名稱以開啟該參數的詳細資訊頁面。
4. 選擇 History (歷程記錄) 索引標籤。
5. 選擇您要刪除標籤的參數版本。
6. 選擇 Manage labels (管理標籤)。
7. 在您要刪除的每個標籤旁選擇 Remove (移除)。
8. 完成時，請選擇儲存變更。
9. 確認您的變更正確，在文字方塊中輸入 `Confirm`，然後選擇 Confirm (確認)。

使用 參數標籤 AWS CLI

本節說明如何使用 AWS Command Line Interface (AWS CLI) 執行以下工作。

- 使用 建立新的參數標籤 AWS CLI
- 使用 檢視參數的標籤 AWS CLI
- 檢視使用 指派標籤的參數清單 AWS CLI
- 使用 移動參數標籤 AWS CLI
- 使用 刪除參數標籤 AWS CLI

使用 建立新的參數標籤 AWS CLI

下列程序說明如何使用 AWS CLI主控台，將標籤連接至現有參數的一個特定版本。您無法在建立參數時附加標籤。

建立參數標籤

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

2. 執行以下命令，以檢視您具有連接標籤許可的參數清單。

   > **Note**
   >
   > 參數僅適用於建立參數 AWS 區域 的 。如果沒有看到您要附加標籤的參數，請確認您的區域。

   ```
   aws ssm describe-parameters
   ```

   記下您要附加標籤的參數名稱。

3. 執行以下命令，以檢視參數的所有版本。

   ```
   aws ssm get-parameter-history --name "parameter-name"
   ```

   記下您要附加標籤的參數版本。

4. 執行以下命令，依照版本號碼擷取有關參數的資訊。

   ```
   aws ssm get-parameters --names "parameter-name:version-number"
   ```

   請見此處範例。

   ```
   aws ssm get-parameters --names "/Production/SQLConnectionString:3"
   ```

5. 執行以下命令其中之一，將標籤附加至參數版本。如果您要連接多個標籤，以空格分隔標籤名稱。

   將標籤附加至參數的最新版本

   ```
   aws ssm label-parameter-version --name parameter-name  --labels label-name
   ```

   將標籤附加至指定的參數版本。

```
aws ssm label-parameter-version --name parameter-name --parameter-version version-number --labels label-name
```

請見下方範例。

```
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**
>
> 如果輸出顯示您在 InvalidLabels 清單中建立的標籤，表示該標籤不符合本主題中前述的要求。請檢閱需求，然後再試一次。如果 InvalidLabels 清單是空的，表示您的標籤已成功附加至參數的版本。

6. 您可以使用版本號碼或標籤名稱檢視參數的詳細資訊。執行以下命令，並指定您在上一步驟建立的標籤。

```
aws ssm get-parameter --name parameter-name:label-name --with-decryption
```

該命令會傳回相關資訊，如以下所示。

```
{
    "Parameter": {
        "Version": version-number,
        "Type": "parameter-type",
        "Name": "parameter-name",
        "Value": "parameter-value",
        "Selector": ":label-name"
    }
}
```

> **Note**
>
> 輸出中選擇器是您在 Name 輸入欄位中指定的版本號碼或標籤。

使用 檢視參數的標籤 AWS CLI

您可以使用 GetParameterHistory API 操作來檢視完整的歷程記錄，以及已連接至指定參數的所有標籤。或者，您可以使用 GetParametersByPath API 操作來檢視已被指派特定標籤的所有參數清單。

使用 GetParameterHistory API 操作檢視參數的標籤

1. 執行以下命令，以檢視您可以檢視標籤的參數清單。

   > **Note**
   >
   > 參數只能在建立該參數的區域中使用。如果沒有看到您要移動標籤的參數，請確認您的區域。

   ```
   aws ssm describe-parameters
   ```

   記下您要檢視其標籤的參數名稱。

2. 執行以下命令，以檢視參數的所有版本。

   ```
   aws ssm get-parameter-history --name parameter-name --with-decryption
   ```

   系統會傳回相關資訊，如下所示。

   ```
   {
       "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"
        }
    ]
}
```

檢視使用 指派標籤的參數清單 AWS CLI

您可以使用 GetParametersByPath API 操作來檢視某路徑中已被指派特定標籤的所有參數的清單。

執行以下命令，以檢視在某路徑中已被指派特定標籤的參數清單。將每個########取代為您自己的資訊。

```
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
```

系統會傳回相關資訊，如下所示。在此範例中，使用者在 /Config 路徑之下進行搜尋。

```
{
    "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 移動參數標籤 AWS CLI

下列程序說明如何將參數標籤移動至相同參數的不同版本。

若要移動參數標籤

1. 執行以下命令，以檢視參數的所有版本。將####取代為您自己的資訊。

   ```
   aws ssm get-parameter-history \
       --name "parameter name"
   ```

   記下您要在其中移動標籤的參數版本。

2. 執行以下命令，將現有標籤指派到參數的不同版本。將每個########取代為您自己的資訊。

   ```
   aws ssm label-parameter-version \
       --name parameter name \
       --parameter-version version number \
       --labels name-of-existing-label
   ```

   > **Note**
   >
   > 如果您要將現有標籤移動至參數的最新版本，請從命令中將 `--parameter-version` 移除。

使用 刪除參數標籤 AWS CLI

以下程序說明如何使用 AWS CLI來刪除參數標籤。

刪除參數標籤

1. 執行以下命令，以檢視參數的所有版本。將####取代為您自己的資訊。

```
aws ssm get-parameter-history \
    --name "parameter name"
```

系統會傳回相關資訊，如下所示。

```
{
    "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"
        }
    ]
}
```

記下您要刪除標籤的參數版本。

2. 執行以下命令，以刪除您從該參數中選擇的標籤。將每個########取代為您自己的資訊。

```
aws ssm unlabel-parameter-version \
    --name parameter name \
    --parameter-version version \
    --labels label 1,label 2,label 3
```

系統會傳回相關資訊，如下所示。

```
{
    "InvalidLabels": ["invalid"],
    "DeletedLabels" : ["Prod"]
 }
```

在Parameter Store中使用參數版本

每次編輯參數的值時， 中的工具Parameter Store都會 AWS Systems Manager 建立新的參數版本，並保留先前的版本。當您最初建立參數時，Parameter Store 會將該參數指派為版本 1。當您變更參數值時，Parameter Store 會自動將版本號增加 1。您可以在參數歷程記錄中檢視所有版本的詳細資訊，包括值。

您也可以指定要在 API 命令和 SSM 文件中使用的參數版本，例如：`ssm:MyParameter:3`。您可以在 API 呼叫與 SSM 文件中指定參數名稱和特定版本號碼。如果您不指定版本編號，系統會自動使用最新的版本。如果您指定不存在的版本編號，系統會傳回錯誤，而不是返回使用參數的最新或預設版本。

您可以使用參數版本來查看參數在一段時間內變更的次數。參數版本也提供一層保護，以防不小心變更參數值。

一個參數最多可以建立最多 100 個版本。建立 100 個參數版本後，每次建立新版本時，會從歷史記錄中移除最舊版本的參數，以騰出空間供新版本使用。

例外狀況是歷史記錄中已有 100 個參數版本，並將參數標籤指派給最舊版本的參數。在這種情況下，不會從歷史記錄中移除該版本，而建立新參數版本的請求會失敗。此防護措施是為了避免刪除具有指定任務關鍵型標籤的參數版本。若要繼續建立新參數，請先將標籤從最舊版本的參數移至較新的參數，以便在操作中使用。如需有關移動參數標籤的相關資訊，請參閱 使用主控台移動參數標籤 和 使用 移動參數標籤 AWS CLI。

以下程序說明如何編輯參數，然後確認新版本已經建立。您可以使用 `get-parameter` 和 `get-parameters` 命令來檢視參數版本。如需有關使用這些命令的範例，請參閱《AWS Systems Manager API 參考》中的 GetParameter 和 GetParameters

主題

- 使用主控台建立新的參數版本
- 參考參數版本

使用主控台建立新的參數版本

您可以使用 Systems Manager 主控台建立參數的新版本，並檢視參數的版本歷程記錄。

建立參數的新版本

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Parameter Store。
3. 選擇您先前建立之參數的名稱。如需有關建立新參數的資訊，請參閱在 Systems Manager 中建立 Parameter Store 參數。
4. 選擇編輯。
5. 在 Value (值) 方塊中輸入新的值，然後選擇 Save changes (儲存變更)。
6. 選擇剛才更新的參數名稱。請在 Overview (概觀) 索引標籤上確認版本編號增加 1，然後確認新的值。
7. 若要檢視參數所有版本的歷程記錄，請選擇 History (歷程記錄) 索引標籤。

參考參數版本

您可以在命令、API 呼叫和 SSM 文件中，使用下列格式來參考特定參數版本：ssm: *parameter-name*:*version-number*。

在下列範例中，Amazon Elastic Compute Cloud (Amazon EC2) `run-instances command` 使用參數 `golden-ami` 的第 3 版。

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` 和參數值僅適用於 `--image-id` 選項和包含 Amazon Machine Image (AMI) 做為其數值的參數。如需詳細資訊，請參閱使用 Parameter Store 中 Amazon Machine Image ID 的原生參數支援。

以下是在 SSM 文件中指定 `MyRunCommandParameter` 參數第 2 版的範例。

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}}"
                ]
            }
        }
    ]
}
```

在 Parameter Store 中使用共用參數

共用進階參數可簡化在多帳戶環境中的組態資料管理。您可以集中存放和管理參數，並與需要參考這些參數 AWS 帳戶 的其他 共用這些參數。

Parameter Store 與 AWS Resource Access Manager (AWS RAM) 整合以啟用進階參數共用。 AWS RAM 是一種服務，可讓您與其他 AWS 帳戶 或透過 共用資源 AWS Organizations。

您可以透過建立資源共享 AWS RAM，與 共用您擁有的資源。資源共用指定要共用的資源、要授予的許可以及共用的消費者。消費者可包括：

- 中的特定組織 AWS 帳戶 內部或外部 AWS Organizations
- 中的組織單位 AWS Organizations
- 其在 中的整個組織 AWS Organizations

如需 的詳細資訊 AWS RAM，請參閱AWS RAM 《 使用者指南》。

本主題說明如何共用您擁有的參數，以及如何使用與您共用的參數。

目錄

- 共用參數的先決條件
- 共用參數
- 停止共享共用參數
- 識別共用參數
- 存取共用參數
- 共用參數的許可集
- 共用參數的輸送量上限
- 共用參數定價
- 已關閉之 AWS 帳戶的跨帳戶存取權

共用參數的先決條件

必須符合下列先決條件，才能從帳戶共用參數：

- 若要共用參數，您必須在 中擁有該參數 AWS 帳戶。您不能共用已與您共用的參數。
- 若要共用參數，該參數必須位於進階參數層。如需有關參數方案的詳細資訊，請參閱管理參數層。如需有關將現有標準參數變更為進階參數的詳細資訊，請參閱將標準參數變更為進階參數。
- 若要共用`SecureString`參數，必須使用客戶受管金鑰加密，而且您必須透過 分別共用金鑰 AWS Key Management Service。 AWS 受管金鑰 無法共用。使用預設值加密的參數 AWS 受管金鑰 可以更新為改用客戶受管金鑰。如需 AWS KMS 關鍵定義，請參閱《 AWS Key Management Service 開發人員指南》中的AWS KMS 概念。
- 若要與 中的組織或組織單位共用參數 AWS Organizations，您必須啟用與 共用 AWS Organizations。如需詳細資訊，請參閱《AWS RAM 使用者指南》中的透過 AWS Organizations啟用共用。

共用參數

若要共用參數，必須將它新增至資源共用。資源共用是可讓您跨 共用資源 AWS RAM 的資源 AWS 帳戶。資源共享指定要共用的資源，以及共用它們的消費者。

當您與其他 共用您擁有的參數時 AWS 帳戶，您可以從兩個 AWS 受管許可中選擇授予取用者。如需詳細資訊，請參閱共用參數的許可集。

如果您是 中組織的一部分， AWS Organizations 且已啟用組織內的共用，則您可以將從 AWS RAM 主控台對共用參數的存取權授予組織中的取用者。否則，取用者會收到加入資源共用的邀請，並且在接受邀請後便能存取共用的參數。

您可以使用 AWS RAM 主控台或 AWS CLI共用您擁有的參數。

> **Note**
>
> 雖然您可以使用 Systems Manager PutResourcePolicy API 操作共用參數，但我們建議改用 AWS Resource Access Manager (AWS RAM)。這是因為使用 PutResourcePolicy需要使用 AWS RAM PromoteResourceShareCreatedFromPolicy API 操作，將 參數提升為標準資源共享的額外步驟。否則，Systems Manager DescribeParameters API 操作不會使用 --shared 選項傳回參數。

使用 AWS RAM 主控台共用您擁有的參數

請參閱《AWS RAM 使用者指南》中的 Creating a resource share in AWS RAM。

完成該程序時，請進行下列選擇：

- 在「步驟 1」頁面中，對於資源，選取 Parameter Store Advanced Parameter，然後選取要共用之進階參數方案中每個參數的方塊。
- 在「步驟 2」頁面中，對於受管許可，選擇要授予取用者的許可，如稍後在本主題中的共用參數的許可集所述。

根據您的參數共用目標選擇其他選項。

使用 共享您擁有的參數 AWS CLI

使用 create-resource-share 命令將參數新增至新資源共用。

使用 associate-resource-share 命令將參數新增至現有資源共用。

下列範例會建立新的資源共用，以與組織中及個別帳戶中的取用者共用參數。

```
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"
```

停止共享共用參數

當您停止共享共用參數時，取用者帳戶將無法再存取參數。

若要停止共用您擁有的參數，您必須從資源共用中移除該參數。您可以使用 Systems Manager 主控台、 AWS RAM 主控台或 AWS CLI來這樣做。

使用 AWS RAM 主控台停止共用您擁有的參數

請參閱《AWS RAM 使用者指南》中的 Update a resource share in AWS RAM。

使用 停止共用您擁有的參數 AWS CLI

使用 disassociate-resource-share 命令。

識別共用參數

擁有者和取用者可以使用 AWS CLI識別共用參數。

使用 識別共用參數 AWS CLI

若要使用 識別共用參數 AWS CLI，您可以從 Systems Manager `describe-parameters`命令和 AWS RAM `list-resources`命令中選擇。

將 `--shared` 選項與 `describe-parameters` 搭配使用時，命令會傳回與您共用的參數。

以下是範例：

```
aws ssm describe-parameters --shared
```

存取共用參數

消費者可以使用 AWS 命令列工具和 AWS SDKs 存取共用參數。若為取用者帳戶，與該帳戶共用的參數不會包含在我的參數頁面中。

CLI 範例：使用 存取共用參數詳細資訊 AWS CLI

若要使用 存取共用參數詳細資訊 AWS CLI，您可以使用 get-parameter或 get-parameters命令。您必須將完整參數 ARN 指定為 `--name`，才能從其他帳戶擷取參數。

以下是範例。

```
aws ssm get-parameter \
    --name arn:aws:ssm:us-east-2:123456789012:parameter/MySharedParameter
```

共用參數的支援和不支援整合

目前，您可以在下列整合案例中使用共用參數：

- AWS CloudFormation 範本參數
- AWS 參數和秘密 Lambda 延伸
- Amazon Elastic Compute Cloud (EC2) 啟動範本
- 使用 EC2 RunInstances 命令從 Amazon Machine Image (AMI) 建立執行個體的 `ImageID` 值
- 擷取 Runbooks for Automation 中的參數值，這是 Systems Manager 中的工具

下列案例與整合服務目前不支援使用共用參數：

- 中的命令參數Run Command，這是 Systems Manager 中的工具
- AWS CloudFormation 動態參考
- 中的環境變數值 AWS CodeBuild
- 中的環境變數值 AWS App Runner
- Amazon Elastic Container Service 中的秘密值

共用參數的許可集

取用者帳戶會取得您與其共用之參數的唯讀存取權。取用者無法更新或刪除參數。取用者無法與第三個帳戶共用參數。

當您在 中建立資源共用 AWS Resource Access Manager 以共用參數時，您可以從兩個 AWS 受管許可集中選擇，以授予此唯讀存取權：

AWSRAMDefaultPermissionSSMParameterReadOnly

允許的動作：`DescribeParameters`、`GetParameter`、`GetParameters`

AWSRAMPermissionSSMParameterReadOnlyWithHistory

允許的動作：`DescribeParameters`、`GetParameter`、`GetParameters`、`GetParameterHistory`

按照《AWS RAM 使用者指南》 中 Creating a resource share in AWS RAM 的步驟操作時，請根據您是否希望使用者檢視參數歷史記錄，選擇 `Parameter Store Advanced Parameters` 作為資源類型和其中一個受管許可。

> **Note**
>
> 如果您要以程式設計方式擷取共用參數（例如，使用 AWS Lambda)，您可能需要將 `ssm:GetResourcePolicies`和 `ssm:PutResourcePolicy`許可新增至呼叫 AWS Resource Access Manager API 動作的任何 IAM 角色。

共用參數的輸送量上限

Systems Manager 會限制 GetParameter 和 GetParameters 操作的輸送量上限 (每秒交易數)。輸送量會在個別帳戶層級強制執行。因此，每個使用共用參數的帳戶都可以使用其允許的輸送量上限，而不會受到其他帳戶的影響。如需有關參數輸送量上限的詳細資訊，請參閱下列主題：

- 增加Parameter Store輸送量
- Amazon Web Services 一般參考 中的 Systems Manager Service Quotas。

共用參數定價

跨帳戶共用僅可在進階參數方案中使用。對於進階參數，會依每個進階參數儲存和 API 用量的目前價格產生費用。擁有帳戶需要支付進階參數的儲存費用。對共用進階參數進行 API 呼叫的任何耗用帳戶需要支付參數用量的費用。

例如，如果帳戶 A 建立進階參數 `MyAdvancedParameter`，該帳戶每月需要支付 0.05 美元以儲存參數。

接著帳戶 A 會與帳戶 B 和帳戶 C 共用 `MyAdvancedParameter`。在一個月內，這三個帳戶會呼叫 `MyAdvancedParameter`。下表說明每個帳戶進行的呼叫次數所產生的費用。

> **Note**
>
> 下表中的費用僅供說明使用。若要確認目前的定價，請參閱 AWS Systems ManagerParameter Store 的定價。

帳戶	呼叫次數	費用
帳戶 A (擁有帳戶)	10,000 次呼叫	• 一個月進階參數儲存：0.05 美元

帳戶	呼叫次數	費用
		• 10,000 次對 MyAdvanceParameter 的呼叫：0.05 美元 • 總計：0.10 美元
帳戶 B (耗用帳戶)	20,000 次呼叫	• 20,000 次對 MyAdvanceParameter 的呼叫：0.10 美元 • 總計：0.10 美元
帳戶 C (耗用帳戶)	30,000 次呼叫	• 30,000 次對 MyAdvanceParameter 的呼叫：0.15 美元 • 總計：0.15 美元

已關閉之 AWS 帳戶的跨帳户存取權

如果關閉 AWS 帳戶 擁有共用參數的 ，則所有取用帳戶都會失去共用參數的存取權。如果擁有帳戶在帳戶關閉後 90 天內重新開啟，則耗用帳戶會重新取得對先前共用參數的存取權。如需在關閉後期間重新開啟帳戶的詳細資訊，請參閱《 AWS 帳戶管理 參考指南》中的在關閉 AWS 帳戶 之後存取您的 。

在 Parameter Store 中透過 Run Command 命令來使用參數

您可以在 中使用 參數Run Command，這是 中的工具 AWS Systems Manager。如需詳細資訊，請參閱AWS Systems Manager Run Command。

使用主控台執行字串參數

以下程序會逐步解說如何執行使用 String 參數的命令。

使用 Parameter Store 執行字串參數

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Run Command。
3. 選擇執行命令。

4. 在 Command document (命令文件) 清單中選擇 `AWS-RunPowerShellScript` (Windows) 或 `AWS-RunShellScript` (Linux)。
5. 對於 Command parameters (命令參數)，輸入 **`echo {{ssm:parameter-name}}`**。例如：**`echo {{ssm:/Test/helloWorld}}`**。
6. 在 Targets (目標) 區段中，透過手動指定標籤、選取執行個體或邊緣裝置，或指定資源群組，選擇您要執行這項操作的受管節點。

   > **Tip**
   >
   > 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

7. 對於其他參數：
   - 在 Comment (註解) 中，輸入此命令的相關資訊。
   - 在逾時 (秒) 中，指定在命令執行全面失敗之前，系統要等候的秒數。
8. 對於 Rate control (速率控制)：
   - 在並行中，指定可同時執行命令的受管節點數目或百分比。

     > **Note**
     >
     > 如果透過指定套用至受管節點的標籤或指定 AWS 資源群組選取了目標，且您不確定會以多少個受管節點為目標，則透過指定百分比限制可以同時執行文件之目標的數量。

   - 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比之節點上的命令失敗之後，停止在其他受管節點上執行命令。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止傳送命令。仍在處理命令的受管節點也可能會傳送錯誤。
9. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取將命令輸出寫入至 S3 儲存貯體方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

   > **Note**
   >
   > 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給執行個體之執行個體設定檔 (適用於 EC2 執行個體) 或 IAM 服務角色 (啟用混合模式的機器) 的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中

AWS 帳戶，請確定與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

10. 在SNS 通知區段中，如果您要傳送有關命令執行狀態的通知，請選取啟用 SNS 通知核取方塊。

    如需為 Run Command 設定 Amazon SNS 通知的詳細資訊，請參閱 使用 Amazon SNS 通知監控 Systems Manager 狀態變更。

11. 選擇執行。
12. 在 Command ID (命令 ID) 頁面的 Targets and outputs (目標和輸出) 區域中，選取執行命令的節點 ID 旁邊的按鈕，然後選擇 View output (檢視輸出)。確認命令的輸出是您為參數提供的數值，例如 **`This is my first parameter`**。

使用 執行參數 AWS CLI

範例 1：簡單命令

下列範例命令包含名為 `DNS-IP` 的 Systems Manager 參數。此參數的值就是節點的 IP 地址。此範例使用 a AWS Command Line Interface (AWS CLI) 命令來回應參數值。

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
```

該命令會傳回相關資訊，如以下所示。

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

命令執行完成後，您可以使用下列命令來檢視其詳細資訊：

- get-command-invocation：檢視有關命令執行的詳細資訊。
- list-command-invocations：檢視特定受管節點上的命令執行狀態。
- list-commands：檢視受管節點之間的命令執行狀態。

範例 2：解密 **SecureString** 參數值

下一個範例命令會使用名為 SecurePassword 的 SecureString 參數。parameters 命令會擷取並解密 SecureString 參數的值，然後重設本機管理員密碼，而不需要以明文傳送密碼。

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
```

範例 3：參照 SSM 文件中的參數

您也可以在 SSM 文件的參數部分中參考 Systems Manager 參數，如以下範例所示。

```
{
   "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}}"
          }
       }
    ]
}
```

不要混淆 SSM 文件的 `runtimeConfig` 區段中使用的本機參數與 Parameter Store 參數的相似語法。本機參數與 Systems Manager 參數不同。您可以從缺少 `ssm:` 字首這一點來區分本機參數與 Systems Manager 參數。

```
"runtimeConfig":{
        "aws:runShellScript":{
            "properties":[
                {
                    "id":"0.aws:runShellScript",
                    "runCommand":"{{ commands }}",
                    "workingDirectory":"{{ workingDirectory }}",
                    "timeoutSeconds":"{{ executionTimeout }}"
```

Note

SSM 文件不支援參考 SecureString 參數。這表示若要在 Run Command 中 (舉例來說) 使用 SecureString 參數，您必須先擷取參數值，再傳遞給 Run Command，如以下範例所示：

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
```

使用 Parameter Store 中 Amazon Machine Image ID 的原生參數支援

建立 `String` 參數時，您現在可以將資料類型指定為 `aws:ec2:image`，以確保輸入的參數值為有效的 Amazon Machine Image (AMI) ID 格式。

支援 AMI ID 格式可讓您避免每次想要在程序中使用的 AMI 發生變更時，都使用新 ID 來更新所有指令碼和範本。您可以使用資料類型 `aws:ec2:image` 建立參數，並為其數值輸入 AMI 的 ID。這是您目前想要建立新執行個體的 AMI。然後，您可以在範本、命令和指令碼中參考此參數。

例如，您在執行 Amazon Elastic Compute Cloud (Amazon EC2) `run-instances` 命令時，可以指定包含您偏好的 AMI ID 的參數。

> **Note**
>
> 執行此命令的使用者必須具有包含 `ssm:GetParameters` API 操作的 AWS Identity and Access Management (IAM) 許可，才能驗證參數值。否則，參數建立程序會失敗。

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
```

當您使用 Amazon EC2 主控台建立執行個體時，也可以選擇偏好的 AMI。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Using a Systems Manager parameter to find an AMI。

當您需要在執行個體建立工作流程中使用不同的 AMI 時，只需要使用新的 AMI 值更新參數，Parameter Store 會再次驗證您已使用正確的格式輸入 ID。

授予許可以建立 **aws:ec2:image** 資料類型的參數

使用 AWS Identity and Access Management (IAM) 政策，您可以提供或限制使用者存取 Parameter Store API 操作和內容。

若要建立 aws:ec2:image 資料類型參數，使用者必須同時擁有 ssm:PutParameter 和 ec2:DescribeImages 許可。

以下範例政策向使用者授予許可，為 aws:ec2:image 呼叫 PutParameter API 操作。這意味著使用者可以將資料類型 aws:ec2:image 的參數新增至系統。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ssm:PutParameter",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeImages",
            "Resource": "*"
        }
    ]
}
```

AMI 格式驗證的運作方式

當您將 aws:ec2:image 指定為參數的資料類型時，Systems Manager 不會立即建立參數。它會執行非同步驗證操作，以確保參數值符合 AMI ID 的格式化要求，並且指定的 AMI 在您的 AWS 帳戶中可用。

在驗證操作完成之前，可能會產生參數版本號碼。操作可能無法完成，即使會產生參數版本編號。

若要監控參數是否建立成功，建議您使用 Amazon EventBridge 傳送 create 與 update 參數操作的相關通知給您。這些通知會報告參數操作是否成功。如果操作失敗，通知會包含錯誤訊息，指出失敗原因。

```
{
    "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"
    }
}
```

如需在 EventBridge 中訂閱 Parameter Store 事件的詳細資訊，請參閱 根據 Parameter Store 事件設定通知或觸發動作。

從 Parameter Store 中刪除參數

本主題說明如何刪除您在 中建立的參數Parameter Store，這是 中的工具 AWS Systems Manager。

> **Warning**
>
> 刪除參數會移除其所有版本。刪除後，參數及其版本均無法還原。

使用主控台刪除參數

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Parameter Store。
3. 在 My parameters (我的參數) 索引標籤上，選取要刪除的每個參數旁邊的核取方塊。
4. 選擇 刪除 。
5. 在確認對話方塊上，選擇 Delete parameters (刪除參數)。

使用 刪除參數 AWS CLI

- 執行以下命令：

  ```
  aws ssm delete-parameter --name "my-parameter"
  ```

  使用要刪除的參數之名稱取代 *my-parameter*。

  如需有關可與 delete-parameter 命令搭配使用之所有選項的資訊，請參閱 命令參考 的 delete-parameter 一節。 AWS Systems Manager AWS CLI

在 Parameter Store 中使用公有參數

有些 AWS 服務 會將常見成品的相關資訊發佈為 AWS Systems Manager 公有參數。例如，Amazon Elastic Compute Cloud (Amazon EC2) 服務會發佈有關 Amazon Machine Images (AMIs) 的資訊作為公有參數。

本指南的主題

- 在 Parameter Store 中探索公有參數
- 在 Parameter Store 中呼叫 AMI 公有參數
- 在 Parameter Store 中呼叫 ECS 最佳化的 AMI 公有參數
- 在 Parameter Store 中呼叫 EKS 最佳化的 AMI 公有參數
- 在 中呼叫 的公有參數 AWS 服務、區域、端點、可用區域、本機區域和 Wavelength 區域 Parameter Store

相關 AWS 部落格文章

- 使用 查詢 AWS 區域、端點等 AWS Systems ManagerParameter Store
- 使用 AWS Systems ManagerParameter Store 查詢最新的 Amazon Linux AMI ID
- 使用 AWS Systems ManagerParameter Store 查詢最新的 Windows AMI

在 Parameter Store 中探索公有參數

您可以使用 Parameter Store 主控台或 AWS Command Line Interface來搜尋公有參數。

公有參數名稱以 aws/service/list 開頭。名稱的下一個部分對應於擁有該參數的服務。

以下是提供公有參數的 AWS 服務 和其他 資源的部分清單：

- 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

並非所有公有參數都會發佈到每個 AWS 區域。

使用 Parameter Store 主控台查找公有參數

您在 AWS 帳戶 和 AWS 區域 中必須至少有一個參數，才能使用 主控台搜尋公有參數。

若要使用主控台查找公有參數

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。

2. 在導覽窗格中，選擇 Parameter Store。
3. 選擇 Public parameters (公有參數) 索引標籤。
4. 選擇 Select a service (選取服務) 下拉式選單。選擇您要使用其參數的服務。
5. (選用) 在搜尋列中輸入更多資訊，以篩選所選服務擁有的參數。
6. 選擇您要使用的公有參數。

使用 尋找公有參數 AWS CLI

使用 `describe-parameters` 以發現公有參數。

使用 `get-parameters-by-path` 以獲得 `/aws/service/list` 下列出的服務的實際路徑。若要獲得服務的路徑，請從路徑中刪除 `/list`。例如，`/aws/service/list/ecs` 會變成 `/aws/service/ecs`。

若要在 Parameter Store 中擷取不同服務擁有的公有參數清單，請執行以下命令。

```
aws ssm get-parameters-by-path --path /aws/service/list
```

該命令會傳回相關資訊，如以下所示。此範例輸出因空間不足已被截斷。

```
{
    "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"
        }
    ]
}
```

如果您想要檢視特定服務所擁有的參數，請從執行先前命令後產生的清單中選擇服務。然後，使用所需服務的名稱進行 `get-parameters-by-path` 呼叫。

例如：`/aws/service/global-infrastructure`。路徑可能是單層級 (只呼叫與給定的確切值匹配的參數) 或遞迴 (包含您給定的路徑以外的路徑中的元素)。

> **Note**
>
> 所有區域中的查詢都不支援 `/aws/service/global-infrastructure` 路徑。如需相關資訊，請參閱 在 中呼叫 的公有參數 AWS 服務、區域、端點、可用區域、本機區域和 Wavelength 區域 Parameter Store。

對於您指定的服務，如果沒有傳回結果，則應新增 `--recursive` 標誌並重新執行命令。

```
aws ssm get-parameters-by-path --path /aws/service/global-infrastructure
```

這將傳回 `global-infrastructure` 擁有的所有參數。以下是範例。

```
{
    "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"
        }
    ]
}
```

透過使用 Option:BeginsWith 篩選條件，您也可以檢視特定服務所擁有的參數。

```
aws ssm describe-parameters --parameter-filters "Key=Name, Option=BeginsWith, Values=/
aws/service/ami-amazon-linux-latest"
```

該命令會傳回相關資訊，如以下所示。此範例輸出因空間不足已被截斷。

```
{
    "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` 時，傳回的參數可能會不同，因為它使用了不同的搜尋模式。

在 Parameter Store 中呼叫 AMI 公有參數

在以下路徑中，Amazon Elastic Compute Cloud (Amazon EC2) Amazon Machine Image (AMI) 公有參數可用於 Amazon Linux 1、Amazon Linux 2、Amazon Linux 2023 (AL2023)、macOS 以及 Windows Server：

- 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`

呼叫 Amazon Linux 1、Amazon Linux 2 和 Amazon Linux 2023 的 AMI 公有參數。

您可以使用 () AMIs中的 AWS 區域 下列命令，檢視目前 中所有 Amazon Linux 1、Amazon Linux 2 和 Amazon Linux 2023 (AL2023) 的 AWS Command Line Interface 清單AWS CLI。

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
```

該命令會傳回相關資訊，如以下所示。

```
[
    "/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檢視這些 AMIs 的詳細資訊，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 代表 支援的 AWS 區域 識別符 AWS Systems Manager，例如us-east-2美國東部（俄亥俄）區域。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域欄位。

該命令會傳回相關資訊，如以下所示。此範例輸出因空間不足已被截斷。

```
{
    "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 GetParameters API 操作及完整 AMI 名稱 (包括路徑) 檢視特定 AMI 的詳細資訊。以下為範例命令。

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
```

此命令會傳回以下資訊。

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

呼叫 macOS 的 AMI 公有參數

您可以使用 macOSAMIs中的下列命令 AWS 區域 來檢視目前 中所有 的清單 AWS CLI。

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
```

該命令會傳回相關資訊，如以下所示。

```
[
    "/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檢視這些 AMIs 的詳細資訊，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 代表 支援的 AWS 區域 識別符 AWS Systems Manager，例如us-east-2美國東部（俄亥俄）區域。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域欄位。

該命令會傳回相關資訊，如以下所示。此範例輸出因空間不足已被截斷。

```
{
    "Parameters": [
        ...sample results pending...
    ]
}
```

您可以使用 GetParameters API 操作及完整 AMI 名稱 (包括路徑) 檢視特定 AMI 的詳細資訊。以下為範例命令。

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
```

此命令會傳回以下資訊。

```
{
    "Parameters": [
         ...sample results pending...
    ],
```

```
    "InvalidParameters": []
}
```

呼叫 Windows Server 的 AMI 公有參數

您可以使用 Windows ServerAMIs中的下列命令 AWS 區域 來檢視目前 中所有 的清單 AWS CLI。

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
```

該命令會傳回相關資訊，如以下所示。此範例輸出因空間不足已被截斷。

```
[
    "/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檢視這些 AMIs 的詳細資訊，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 代表 支援的 AWS 區域 識別符 AWS Systems Manager，例如us-east-2美國東部（俄亥俄）區域。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域欄位。

該命令會傳回相關資訊，如以下所示。此範例輸出因空間不足已被截斷。

```
{
    "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 GetParameters API 操作及完整 AMI 名稱 (包括路徑) 檢視特定 AMI 的詳細資訊。以下為範例命令。

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
```

此命令會傳回以下資訊。

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

在 Parameter Store 中呼叫 ECS 最佳化的 AMI 公有參數

Amazon Elastic Container Service (Amazon ECS) 服務會發佈最新的 Amazon ECS 最佳化 Amazon Machine Images (AMIs) 的名稱作為公有參數。我們鼓勵使用者為 Amazon ECS 建立新的 Amazon Elastic Compute Cloud (Amazon EC2) 叢集時使用此 AMI，因為最佳化的 AMIs 包含錯誤修復和功能更新。

使用以下命令檢視 Amazon Linux 2 的最新的 Amazon ECS 最佳化 AMI 的名稱。若要查看其他作業系統的命令，請參閱《Amazon Elastic Container Service 開發人員指南》中的擷取 Amazon ECS 最佳化 AMI 中繼資料。

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
```

該命令會傳回相關資訊，如以下所示。

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

在 Parameter Store 中呼叫 EKS 最佳化的 AMI 公有參數

Amazon Elastic Kubernetes Service (Amazon EKS) 服務會發佈最新的 Amazon EKS 最佳化 Amazon Machine Image (AMI) 的名稱作為公有參數。我們鼓勵使用者在將節點新增至 Amazon EKS 叢集時使用此 AMI，因為新版本包括 Kubernetes 修補程式和安全更新。過去，為了確保您是使用最新的 AMI，您得不時檢查 Amazon EKS 文件以及利用新的 AMI ID 手動更新任何部署範本或資源。

使用以下命令檢視 Amazon Linux 2 的最新的 Amazon EKS 最佳化 AMI 的名稱。

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
```

該命令會傳回相關資訊，如以下所示。

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

在 中呼叫 的公有參數 AWS 服務、區域、端點、可用區域、本機區域和 Wavelength 區域 Parameter Store

您可以使用下列路徑呼叫公有參數的 AWS 區域、服務、端點、可用性和 Wavelength 區域。

```
/aws/service/global-infrastructure
```

Note

目前，AWS 區域 只有下列 中的查詢/aws/service/global-infrastructure支援 路徑：

- 美國東部 (維吉尼亞北部) (us-east-1)
- 美國東部 (俄亥俄) (us-east-2)
- 美國西部 (加利佛尼亞北部) (us-west-1)
- 美國西部 (奧勒岡) (us-west-2)
- 亞太區域 (香港) (ap-east-1)
- 亞太區域 (孟買) (ap-south-1)
- 亞太區域 (首爾) (ap-northeast-2)
- 亞太區域 (新加坡) (ap-southeast-1)
- 亞太區域 (雪梨) (ap-southeast-2)
- 亞太區域 (東京) (ap-northeast-1)
- 加拿大 (中部) (ca-central-1)
- 歐洲 (法蘭克福) (eu-central-1)
- 歐洲 (愛爾蘭) (eu-west-1)
- 歐洲 (倫敦) (eu-west-2)
- 歐洲 (巴黎) (eu-west-3)
- 歐洲 (斯德哥爾摩) (eu-north-1)
- 南美洲 (聖保羅) (sa-east-1)

如果您在不同的商業區域中工作，可以在查詢中指定支援的區域以檢視結果。例如，如果您在加拿大西部 (卡加利) (ca-west-1) 區域中工作，可以在查詢中指定加拿大 (中部) (ca-central-1)：

```
aws ssm get-parameters-by-path \
    --path /aws/service/global-infrastructure/regions \
    --region ca-central-1
```

檢視作用中 AWS 區域

您可以使用 AWS Command Line Interface () 中的下列命令 AWS 區域 來檢視所有作用中的清單AWS CLI。

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
```

該命令會傳回相關資訊，如以下所示。

```
[
    "/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"
]
```

檢視可用 AWS 服務

您可以使用下列命令檢視所有可用項目的完整清單 AWS 服務 ，並依字母順序排序。此範例輸出因空間不足已被截斷。

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(@)"
```

該命令會傳回相關資訊，如以下所示。此範例輸出因空間不足已被截斷。

```
[
     "/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",
```

檢視 的支援區域 AWS 服務

您可以檢視提供服務 AWS 區域 的清單。此範例使用 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
```

該命令會傳回相關資訊，如以下所示。

```
[
    "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"
]
```

檢視服務的區域端點

您可以使用以下命令檢視服務的區域性端點。此命令會查詢美國東部 (俄亥俄) (us-east-2) 區域。

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
```

該命令會傳回相關資訊，如以下所示。

```
"ssm.us-east-2.amazonaws.com"
```

檢視完整的可用區域詳細資訊

您可以使用下列命令來檢視可用區域。

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/
```

該命令會傳回相關資訊，如以下所示。此範例輸出因空間不足已被截斷。

```
{
    "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"
        }
    ]
}
```

僅檢視可用區域名稱

您只能使用下列命令來檢視可用區域的名稱。

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(@)"
```

該命令會傳回相關資訊，如以下所示。此範例輸出因空間不足已被截斷。

```
[
    "/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"
```

檢視單一區域中可用區域的名稱

您可以使用下列命令檢視一個區域 (在此範例中，為 us-east-2) 中的可用區域名稱。

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(@)"
```

該命令會傳回相關資訊，如以下所示。

```
[
    "/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"
```

僅檢視可用區域 ARN

您只能使用下列命令來檢視可用區域的 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(@)"
```

該命令會傳回相關資訊，如以下所示。此範例輸出因空間不足已被截斷。

```
[
    "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",
```

檢視本地區域詳細資訊

您可以使用下列命令來檢視本地區域。

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
```

該命令會傳回相關資訊，如以下所示。此範例輸出因空間不足已被截斷。

```
{
    "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"
        }
    ]
}
```

檢視 Wavelength Zone 詳細資訊

您可以使用下列命令來檢視 Wavelength 區域。

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
```

該命令會傳回相關資訊，如以下所示。此範例輸出因空間不足已被截斷。

```
{
    "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"
        }
    ]
}
```

檢視本地區域下的所有參數和數值

您可以使用下列命令來檢視本地區域的所有參數資訊。

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"
```

該命令會傳回相關資訊，如以下所示。此範例輸出因空間不足已被截斷。

```
{
    "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"
        }
    ]
}
```

僅檢視本地區域參數名稱

您可以使用下列命令，只檢視本地區域參數的名稱。

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(@)"
```

該命令會傳回相關資訊，如以下所示。

```
[
    "/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演練

本節中的逐步解說說明如何在測試環境中使用 中的Parameter Store工具 建立 AWS Systems Manager、存放和執行參數。本逐步解說說明如何Parameter Store搭配其他 Systems Manager 工具使用 。您也可以使用 Parameter Store 與其他 AWS 服務。如需詳細資訊，請參閱什麼是參數？。

目錄

- 在 Parameter Store 中建立 SecureString 參數並將節點加入網域 (PowerShell)

在 Parameter Store 中建立 SecureString 參數並將節點加入網域 (PowerShell)

此逐步解說說明如何使用 AWS Systems Manager `SecureString`參數 和 將Windows Server節點加入網域Run Command。此逐步解說使用典型的網域參數，例如網域名稱和網域使用者名稱。這些值會以未加密的字串值的形式進行傳遞。網域密碼是以 AWS 受管金鑰 加密，並以加密字串的形式傳遞。

先決條件

此演練假設您已經在與您的 Amazon VPC 關聯的 DHCP 選項集中指定了網域名稱和 DNS 伺服器 IP 地址。如需資訊，請參閱《Amazon VPC 使用者指南》中的搭配使用 DHCP 選項集。

建立 `SecureString` 參數並將節點加入網域

1. 使用 在系統中輸入參數 AWS Tools for Windows PowerShell。

   在下列命令中，將每個*#########*替換為您自己的資訊。

   ```
   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**
>
> 僅加密 SecureString 參數的值。參數名稱、說明和其他屬性不會加密。

2. 將下列 AWS Identity and Access Management (IAM) 政策連接至節點的 IAM 角色許可：

   - AmazonSSMManagedInstanceCore – 必要。此 AWS 受管政策允許受管節點使用 Systems Manager 服務核心功能。
   - AmazonSSMDirectoryServiceAccess – 必要。此 AWS 受管政策允許 AWS Directory Service 代表您SSM Agent存取 ，以請求由受管節點加入網域。
   - S3 儲存貯體存取的自訂政策 - 必要。SSM Agent (位於您的節點並執行 Systems Manager 任務) 需要存取 Amazon 擁有的特定 Amazon Simple Storage Service (Amazon S3) 儲存貯體。在您建立的自訂 S3 儲存貯體政策中，您也可以提供 Systems Manager 操作所需的自有 S3 儲存貯體存取權限。

     範例：您可以將 Run Command 命令或 Session Manager 工作階段的輸出寫入 S3 儲存貯體，然後稍後使用此輸出進行稽核或故障診斷。您將存取指令碼或自訂修補基準清單儲存在 S3 儲存貯體中，然後在執行命令時或套用修補基準時參考指令碼或清單。

     如需有關為 Amazon Simple Storage Service (Amazon S3) 儲存貯體存取建立自訂政策的相關資訊，請參閱為執行個體設定檔建立自訂 S3 儲存貯體政策

     > **Note**
     >
     > 您可以選擇是否在 S3 儲存貯體中儲存輸出日誌資料，但如果您已決定使用，則建議在 Systems Manager 組態程序開始時進行設定。如需詳細資訊，請參閱 Amazon Simple Storage Service 主控台使用者指南中的建立儲存貯體。

   - CloudWatchAgentServerPolicy – 選用。此 AWS 受管政策可讓您在受管節點上執行 CloudWatch 代理程式。此政策可讓您讀取節點的資訊，並將資訊寫入 Amazon CloudWatch。只有在使用諸如 Amazon EventBridge 或 CloudWatch Logs 等服務時，您的執行個體設定檔才需要此政策。

> **Note**
>
> 您可以選擇是否使用 CloudWatch 和 EventBridge 功能，但如果您決定使用，則建議在 Systems Manager 組態程序開始時進行設定。如需詳細資訊，請參閱《Amazon EventBridge 使用者指南》和《Amazon CloudWatch Logs 使用者指南》。

3. 編輯連接至節點的 IAM 角色，並新增以下政策。此政策可讓節點許可呼叫 `kms:Decrypt` 和 `ssm:CreateDocument` API。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "ssm:CreateDocument"
            ],
            "Resource": [
                "arn:aws:kms:region:account-id:key/kms-key-id"
            ]
        }
    ]
}
```

4. 將以下的 json 文字複製並黏貼到純文字編輯器，並在以下位置將檔案儲存為 `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 中執行下列命令，以建立新的 SSM 文件。

```
$json = Get-Content C:\temp\JoinInstanceToDomain | Out-String
New-SSMDocument -Name JoinInstanceToDomain -Content $json -DocumentType Command
```

6. 在 Tools for Windows PowerShell 中執行下列命令，將節點加入網域。

```
Send-SSMCommand -InstanceId instance-id -DocumentName JoinInstanceToDomain
```

如果命令成功，系統會傳回類似如下的資訊。

```
WARNING: The changes will take effect after you restart the computer EC2ABCD-
EXAMPLE.
Domain join succeeded, restarting
Computer is part of example.local, exiting
```

如果命令失敗，系統會傳回類似如下的資訊：

```
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.
```

稽核和記錄 Parameter Store 活動

AWS CloudTrail 會擷取在 AWS Systems Manager 主控台、 AWS Command Line Interface (AWS CLI) 和 Systems Manager SDK 中發出的 API 呼叫。您可以在 CloudTrail 主控台中，或在 Amazon Simple Storage Service (Amazon S3) 儲存貯體中檢視資訊。帳戶的所有 CloudTrail 日誌使用一個儲存貯體。如需有關檢視和使用 Systems Manager 活動的 CloudTrail 日誌的詳細資訊，請參閱 使用 記錄 AWS Systems Manager API 呼叫 AWS CloudTrail。如需稽核和記錄 Systems Manager 選項的詳細資訊，請參閱 在 中記錄和監控 AWS Systems Manager。

Parameter Store 疑難排解

使用下列資訊來協助您疑難排解 中的Parameter Store工具 的問題 AWS Systems Manager。

故障診斷 **aws:ec2:image** 參數建立

使用下列資訊協助故障診斷建立資 `aws:ec2:image` 料類型參數的問題。

沒有建立執行個體的許可

問題：您嘗試使用 `aws:ec2:image` 參數建立執行個體，但收到「未授權您執行此操作」等錯誤訊息。

- 解決方案：您沒有使用參數值建立 EC2 執行個體所需的所有許可，例如 `ec2:RunInstances`、`ec2:DescribeImages` 和 `ssm:GetParameter` 等的許可。請聯絡組織中擁有管理員許可的使用者，以申請必要的許可。

EventBridge 報告失敗訊息「無法描述資源」

問題：您執行命令來建立 `aws:ec2:image` 參數，但參數建立失敗。您收到來自 Amazon EventBridge 的通知，報告例外狀況「無法描述資源」。

解決方案：此訊息可表示下列情況：

- 您尚未獲得 `ec2:DescribeImages` API 操作的所有許可，或者您缺乏存取參數中參考的特定映像的許可。請聯絡在組織中擁有管理員許可的使用者，以申請必要的許可。
- 您輸入做為參數值的 Amazon Machine Image (AMI) ID 無效。請確定您輸入的 ID AMI 是目前 AWS 區域 和您正在使用的帳戶中可用的 。

無法使用新的 **`aws:ec2:image`** 參數

問題：您剛執行命令來建立 `aws:ec2:image` 參數，系統有報告版本號碼，但參數無法使用。

- 解決方案：當您執行命令來建立使用 `aws:ec2:image` 資料類型的參數時，會立即為該參數產生版本號碼，但必須先驗證參數格式，才能使用參數。此程序需要幾分鐘的時間。若要監控參數建立與驗證程序，您可以執行下列動作：
  - 使用 EventBridge 傳送有關 `create` 和 `update` 參數操作的通知給您。這些通知會報告參數操作是否成功。如需在 EventBridge 中訂閱 Parameter Store 事件的詳細資訊，請參閱 根據 Parameter Store 事件設定通知或觸發動作。
  - 在 Systems Manager 主控台 Parameter Store 區段中，定期重新整理參數清單，以搜尋是否有全新或更新的參數詳細資訊。
  - 使用 GetParameter 命令檢查全新或更新的參數。例如，使用 AWS Command Line Interface (AWS CLI)：

    ```
    aws ssm get-parameter name MyParameter
    ```

    對於新參數，會傳回 `ParameterNotFound` 訊息，直到驗證參數為止。對於您正在更新的現有參數，除非驗證參數，否則不會包含新版本的相關資訊。

  如果您在驗證程序完成之前嘗試再次建立或更新參數，系統會報告驗證仍在進行中。如果參數未建立或更新，您可以在初次嘗試過 5 分鐘後再試一次。

AWS Systems Manager 操作工具

操作工具是一套功能，可協助您管理 AWS 資源。

- Systems Manager 託管的 Amazon CloudWatch 儀表板是 CloudWatch 主控台中的可自訂首頁，您可以在單一檢視 AWS 區域 中用來監控 中的 資源。
- Explorer 是可自訂的操作儀表板，可報告 AWS 資源的相關資訊。 Explorer會顯示您 AWS 帳戶 和之間的操作資料 (OpsData) 彙總檢視 AWS 區域。

- Incident Manager 可協助您減輕並復原影響託管應用程式的事件 AWS。
- OpsCenter 提供中央位置，讓營運工程師和 IT 專業人員可以檢視、調查和解決與 AWS 資源相關的操作工作項目 (OpsItems)。

主題

- AWS Systems Manager Incident Manager
- AWS Systems Manager Explorer
- AWS Systems Manager OpsCenter
- 使用 Systems Manager 託管的 Amazon CloudWatch 儀表板

AWS Systems Manager Incident Manager

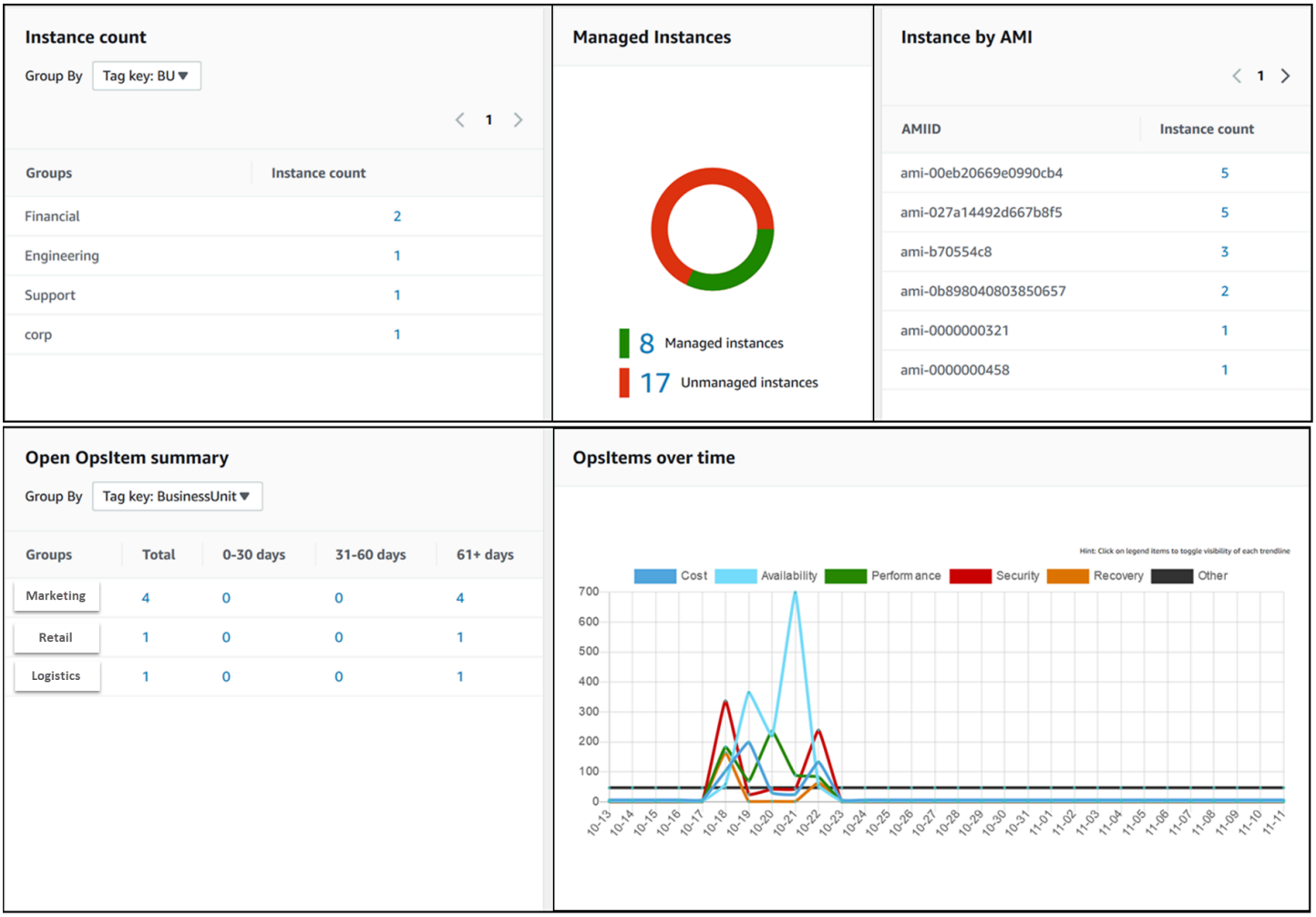
使用 中的 工具 Incident Manager AWS Systems Manager來管理 AWS 託管應用程式中發生的事件。Incident Manager 結合使用者參與、提升、Runbook、回應計劃、聊天頻道和事件後分析，可協助您的團隊更快地分類事件，並將應用程式回復為正常。若要進一步了解 Incident Manager，請參閱《Incident Manager 使用者指南》。

AWS Systems Manager Explorer

AWS Systems Manager Explorer 是可自訂的操作儀表板，可報告 AWS 資源的相關資訊。 Explorer會顯示您 AWS 帳戶 和 之間的操作資料 (OpsData) 彙總檢視 AWS 區域。在 Explorer 中，OpsData 包含有關混合多雲端環境中受管節點的中繼資料。OpsData 也包含其他 Systems Manager 工具提供的資訊，包括Patch Manager修補程式合規和State Manager關聯合規詳細資訊。為了進一步簡化您存取 OpsData 的方式， Explorer會顯示支援服務的資訊 AWS Config， AWS 例如 AWS Trusted Advisor AWS Compute Optimizer和 AWS 支援 （ 支援案例 ）。

為提高營運意識，Explorer 也會顯示營運工作項目 (OpsItems)。Explorer 會提供 OpsItems 如何在您的業務單位或應用程式中分散、隨著時間的趨勢，以及如何因類別而異的相關內容。您可以在 Explorer 中群組和篩選資訊，以專注於與您相關且需要採取動作的項目。當您發現高優先順序的問題時，您可以使用 Systems Manager OpsCenter 來執行Automation Runbook，並快速解決這些問題。若要開始使用 Explorer，請開啟 Systems Manager 主控台。在導覽窗格中，選擇 Explorer。

下列影像顯示一些個別的報告方塊，稱為 Widget，可在 Explorer 中使用。



Explorer 有哪些功能？

Explorer 包含下列功能：

- 可自訂的可操作資訊顯示：Explorer包含drag-and-drop小工具，可自動顯示資源的可操作資訊 AWS 。 會在兩種小工具類型中Explorer顯示資訊。
  - 資訊小工具：這些小工具摘要來自 Amazon EC2、State Manager、 Patch Manager和 AWS 服務等支援 AWS Trusted Advisor、 AWS Compute Optimizer 和 的資料 支援。這些小工具提供重要的內容，協助您了解 AWS 資源的狀態和操作風險。資訊小工具的範例包括 Instance count (執行個體計數)、Instance by AMI (依 AMI 排序的執行個體)、Total noncompliant nodes (不合規節點總數) (修補程式)、Noncompliant associations (不合規關聯) 和 Support Center cases (支援中心案例)。
  - OpsItem 小工具：Systems Manager OpsItem是與一或多個 AWS 資源相關的操作工作項目。OpsItems 是 Systems Manager 的一項功能OpsCenter。 OpsItems可能需要 DevOps 工程師調查並可能修復問題。可能的 OpsItems 範例包括 EC2 執行個體高 CPU 使用率、分離的 Amazon

Elastic Block Store (Amazon EBS) 磁碟區、 AWS CodeDeploy 部署失敗或 Systems Manager 自動化執行失敗。OpsItem 小工具的範例包括 開啟 OpsItem 摘要、依狀態的 OpsItem，以及隨著時間的 OpsItems。

- 篩選條件：每個小工具都能夠根據 AWS 帳戶 AWS 區域和 標籤篩選資訊。篩選器可協助您快速精細化 Explorer 中顯示的資訊。
- 直接連結至服務畫面：為了協助您調查 AWS 資源的問題，Explorer小工具包含相關服務畫面的直接連結。如果您瀏覽至相關的服務畫面，對 Widget 套用的篩選器仍會有效。
- 群組：為協助您了解整個組織的操作問題類型，某些小工具允許您根據帳戶、區域和標籤來群組資料。
- 報告標籤鍵：設定 Explorer 時，可以指定最多五個標籤鍵。這些鍵可協助您在 Explorer 中群組和篩選資料。如果指定的鍵與產生 OpsItem 的資源上的鍵相符，則 OpsItems 中會包含該鍵和值。
- AWS 帳戶 和 AWS 區域 顯示的三種模式： Explorer包含 OpsData 和 和 OpsItems中的下列顯示模式 AWS 帳戶 AWS 區域：
  - 單一帳戶/單一區域：這是預設檢視。此模式允許使用者檢視來自自己的帳戶和目前區域的資料和 OpsItems。
  - 單一帳戶/多重區域：此模式需要您使用 Explorer Settings (設定) 頁面建立一或多個資源資料同步。資源資料同步會從一或多個區域彙總 OpsData。建立資源資料同步之後，您可以切換要在 Explorer 儀表板上使用的同步。然後，您可以根據區域篩選和群組資料。
  - 多重帳戶/多重區域：此模式需要您的組織或公司使用 AWS Organizations 並開啟 All features (所有功能)。在 AWS Organizations 運算環境中設定 後，您可以彙總管理帳戶中的所有帳戶資料。然後，您可以建立資源資料同步，以便根據區域篩選和群組資料。如需所有功能模式的詳細資訊，請參閱啟用組織中的所有功能。
- 報告：您可以將 Explorer 報告以逗號分隔值 (.csv) 檔案格式匯出至 Amazon Simple Storage Service (Amazon S3) 儲存貯體。匯出完成時，您會收到來自 Amazon Simple Notification Service (Amazon SNS) 的提醒。

Explorer 如何與 OpsCenter 相關？

Systems Manager OpsCenter 提供中央位置，讓營運工程師和 IT 專業人員檢視、調查和解決與 AWS 資源OpsItems相關的問題。 Explorer 是報告中樞，DevOps 管理員可檢視其營運資料的彙總摘要 OpsItems，包括跨 AWS 區域 和 帳戶。 Explorer可讓使用者探索趨勢和模式，並視需要使用 Systems Manager Automation Runbook 快速解決問題。

OpsCenter 安裝程式現在已與 Explorer 安裝程式整合。如果您已設定 OpsCenter，則 Explorer 會自動顯示操作資料，包括有關 OpsItems 的彙總資訊。如果您尚未設定 OpsCenter，則可以使用

Explorer設定來開始使用這兩個工具。如需詳細資訊，請參閱開始使用 Systems Manager Explorer 和 OpsCenter。

什麼是 OpsData？

OpsData 是 Systems Manager Explorer 儀表板中顯示的任何操作資料。Explorer 會從下列來源擷取 OpsData：

- Amazon Elastic Compute Cloud (Amazon EC2)

  Explorer 中顯示的資料包括：節點總數、受管和非受管節點總數，以及使用特定 Amazon Machine Image (AMI) 的節點計數。

- Systems Manager OpsCenter

  Explorer 中顯示的資料包括：依狀態的 OpsItems 計數、依嚴重性的 OpsItems 計數、跨群組與 30 天期間開啟的 OpsItems 計數，以及隨時間的 OpsItems 歷史資料。

- Systems Manager Patch Manager

  Explorer 中顯示的資料包括不合規節點和嚴重不合規節點的計數。

- AWS Trusted Advisor

  Explorer 中顯示的資料包括：EC2 預留執行個體在成本最佳化、安全性、容錯能力、效能和服務限制等領域的最佳實務檢查狀態。

- AWS Compute Optimizer

  Explorer 中顯示的資料包括 Under provisioned (佈建不足) 和 Over provisioned (過度佈建) 的 EC2 執行個體計數、最佳化問題清單、隨需定價詳情，以及執行個體類型和價格的建議。

- 支援 中心案例

  Explorer 中顯示的資料包括：案例 ID、嚴重性、狀態、建立時間、主旨、服務和類別。

- AWS Config

  中顯示的資料Explorer包括：合規和不合規 AWS Config 規則的整體摘要、合規和不合規資源的數量，以及每個資源的特定詳細資訊（當您深入研究不合規規則或資源時）。

- AWS Security Hub

  Explorer 中顯示的資料包括：Security Hub 問題清單的整體摘要、依嚴重性分組的每個問題清單的數目，以及有關問題清單的特定詳細資訊。

> **Note**
>
> 若要在 中檢視 AWS Trusted Advisor 和 支援 置中案例Explorer，您必須設定企業或企業帳戶 AWS 支援。

您可以從 Explorer 的 Settings (設定) 頁面檢視和管理 OpsData 來源。如需設定和配置使用 OpsData 填入 Explorer Widget 之服務的詳細資訊，請參閱設定 Explorer 的相關服務。

使用 Explorer 需要付費嗎？

是。當您開啟在整合式設定期間建立 OpsItems 的預設規則時，您會啟動自動建立 OpsItems 的程序。我們會根據每月建立的 OpsItems 數量向您的帳戶收費。也會根據每月進行的 `GetOpsItem`、`DescribeOpsItem`、`UpdateOpsItem` 和 `GetOpsSummary` API 呼叫次數向您收費。此外，您可能需要就向可公開相關診斷資訊的其他服務的公有 API 呼叫支付費用。如需詳細資訊，請參閱 AWS Systems Manager 定價。

主題

- 開始使用 Systems Manager Explorer 和 OpsCenter
- 使用 Explorer
- 從 Systems Manager Explorer 匯出 OpsData
- Systems Manager Explorer 故障診斷

開始使用 Systems Manager Explorer 和 OpsCenter

AWS Systems Manager 使用整合的設定體驗，協助您開始使用 Systems Manager Explorer和 Systems Manager OpsCenter。在本文件中，Explorer 和 OpsCenter 設定稱為整合式設定。如果已設定 OpsCenter，您仍然需要完成整合式設定，以確認設定和選項。如果您尚未設定 OpsCenter，則可以使用整合式設定來開始使用這兩種工具。

> **Note**
>
> 整合式設定僅適用於 Systems Manager 主控台。您無法以程式設計方式設定 Explorer 或 OpsCenter。

整合式設定會執行下列任務：

- 設定角色和許可：整合式設定會建立 AWS Identity and Access Management (IAM) 角色，允許 Amazon EventBridge OpsItems 根據預設規則自動建立。設定之後，您必須如本節所述，為 OpsCenter 設定使用者、群組或角色許可。
- 允許 OpsItem 建立的預設規則：整合式設定會在 EventBridge 中建立預設規則。這些規則會自動建立 OpsItems 以回應事件。這些事件的範例包括： AWS 資源的狀態變更、安全設定變更，或服務變得無法使用。
- 啟用 OpsData 來源：整合式設定會啟用以下可填入 Explorer 小工具的資料來源。
  - 支援 中心 （ 您必須擁有商業或企業支援計劃，才能啟用此來源。）
  - AWS Compute Optimizer （ 您必須有商業或企業支援計劃才能啟用此來源。）
  - Systems Manager State Manager 關聯合規
  - AWS Config 合規
  - Systems Manager OpsCenter
  - Systems Manager Patch Manager 修補程式合規
  - Amazon Elastic Compute Cloud (Amazon EC2)
  - Systems Manager 庫存
  - AWS Trusted Advisor （ 您必須有商業或企業支援計劃才能啟用此來源。）
  - AWS Security Hub

> **Note**
>
> 您可以隨時在 Settings (設定) 頁面上變更設定組態。

完成整合設定後，建議您設定 Explorer 以顯示來自多個區域和帳戶的資料。 Explorer 和 OpsCenter會自動同步 OpsData 和 OpsItems ， AWS 帳戶 以及完成整合設定 AWS 區域 時使用的 。您可以透過建立資源資料同步來彙總來自其他帳戶和區域的 OpsData 及 OpsItems。

設定 Explorer 的相關服務

AWS Systems Manager 探索和 AWS Systems Manager OpsCenter收集來自其他 AWS 服務 和 Systems Manager 工具的資訊，或與之互動。建議您在使用整合式設定之前，設定這些其他服務或工具。

下表包含允許 Explorer和 從其他 AWS 服務 和 Systems Manager 工具OpsCenter收集資訊或與之互動的任務。

任務	資訊
確認 Systems Manager 自動化中的許可	Explorer 和 OpsCenter 允許您使用 Systems Manager Automation Runbook 修正 AWS 資源的問題。若要使用此修復工具，您必須具有執行 Systems Manager Automation Runbook 的許可。如需詳細資訊，請參閱設定自動化。
安裝和設定 Systems Manager Patch Manager	Explorer 包含可提供修補程式符合性相關資訊的 Widget。若要在 Explorer 中檢視此資料，您必須設定修補。如需詳細資訊，請參閱AWS Systems Manager Patch Manager。
安裝和設定 Systems Manager State Manager	Explorer 包含可提供有關 Systems Manager State Manager 關聯合規資訊的小工具。若要在 Explorer 中檢視此資料，您必須設定 State Manager。如需詳細資訊，請參閱AWS Systems Manager State Manager。
開啟 AWS Config 組態記錄器	Explorer 使用 AWS Config 組態記錄器提供的資料，將 EC2 執行個體的相關資訊填入小工具。若要在 中檢視此資料Explorer，請開啟 AWS Config 組態記錄器。如需詳細資訊，請參閱管理組態記錄器。 **Note** 您允許組態記錄器之後，最多可能需要六個小時的時間，Systems Manager 才能在 Explorer 小工具 (顯示有關 EC2 執行個體的資訊) 中顯示資料。
開啟 AWS Trusted Advisor	Explorer 使用 提供的資料 Trusted Advisor ，在成本最佳化、安全性、容錯能力、效能和服務限制等領域顯示 Amazon EC2 預留執行個體最佳實務檢查的狀態。若要在 Explorer 檢視此資

任務	資訊
	料，您必須擁有商業或企業支援方案。如需詳細資訊，請參閱支援。
開啟 AWS Compute Optimizer	Explorer 使用 Compute Optimizer 提供的資料來顯示佈建不足和過度佈建的 EC2 執行個體計數、最佳化問題清單、隨需定價詳細資訊，以及執行個體類型和價格的建議。若要在 Explorer 中檢視此資料，開啟 Compute Optimizer。如需詳細資訊，請參閱 入門 AWS Compute Optimizer。
開啟 AWS Security Hub	Explorer 使用 Security Hub 提供的資料，在小工具中填入有關安全問題清單的資訊。若要在 Explorer 中檢視此資料，開啟 Security Hub 整合。如需詳細資訊，請參閱什麼是 AWS Security Hub。

設定 Systems Manager Explorer 的角色和許可

整合式設定會自動為 AWS Systems Manager Explorer 和 建立和設定 AWS Identity and Access Management (IAM) 角色 AWS Systems Manager OpsCenter。如果您已完成整合式設定，則不需要執行任何其他工作來設定 Explorer 的角色和許可。但是，您必須設定 OpsCenter 的許可，如本主題稍後所述。

整合式設定會建立和設定下列角色，以使用 Explorer 和 OpsCenter。

- `AWSServiceRoleForAmazonSSM`：提供由 Systems Manager 管理或使用的 AWS 資源存取權限。
- `OpsItem-CWE-Role`：允許 CloudWatch Events 和 EventBridge 建立 OpsItems 以回應常見事件。
- `AWSServiceRoleForAmazonSSM_AccountDiscovery`：允許 Systems Manager 在同步資料時呼叫其他 AWS 服務 以探索 AWS 帳戶 資訊。如需有關此角色的詳細資訊，請參閱 使用 角色來收集 OpsCenter和 AWS 帳戶 的資訊 Explorer。
- `AmazonSSMExplorerExport`：允許 Explorer 將 OpsData 匯出至以逗號分隔值的 (CSV) 檔案。

如果您設定 使用 和 資源資料同步Explorer來顯示來自多個帳戶 AWS Organizations 和區域的資料，則 Systems Manager 會建立AWSServiceRoleForAmazonSSM_AccountDiscovery服務連結角色。Systems Manager 使用此角色來取得 AWS 帳戶 中 的相關資訊 AWS Organizations。該角色會使用下列許可政策。

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "organizations:DescribeAccount",
            "organizations:DescribeOrganization",
            "organizations:ListAccounts",
            "organizations:ListAWSServiceAccessForOrganization",
            "organizations:ListChildren",
            "organizations:ListParents"
         ],
         "Resource":"*"
      }
   ]
}
```

如需 AWSServiceRoleForAmazonSSM_AccountDiscovery 角色的詳細資訊，請參閱 使用 角色來收集 OpsCenter和 AWS 帳戶 的資訊 Explorer。

設定的 Systems Manager OpsCenter 的許可

完成整合式設定之後，您必須設定使用者、群組或角色許可，以便使用者在 OpsCenter 中執行動作。

開始之前

可以將 OpsCenter 設定為建立和管理單一帳戶或跨多個帳戶的 OpsItems。如果您OpsCenter將設定為OpsItems跨多個帳戶建立和管理，您可以使用 Systems Manager 委派管理員帳戶或 AWS Organizations 管理帳戶，在其他OpsItems帳戶中手動建立、檢視或編輯。如需有關 Systems Manager 委派管理員帳戶的詳細資訊，請參閱設定 Explorer 的委派管理員。

如果您針對單一帳戶設定 OpsCenter，則只能在建立 OpsItems 的帳戶中檢視或編輯 OpsItems。您無法OpsItems跨 共用或傳輸 AWS 帳戶。因此，建議您在用來執行 AWS 工作負載 AWS 帳戶 的OpsCenter中設定 的許可。然後，您就可以在該帳戶中建立 使用者或群組。利用這種方式，多位營運工程師或 IT 專業人員即可在相同的 AWS 帳戶中建立、檢視和編輯 OpsItems。

Explorer 和 OpsCenter 會使用下列 API 操作。如果使用者、群組或角色可以存取這些動作，您就可以使用 Explorer 和 OpsCenter 的所有功能。您也可以建立更嚴格的存取權，如本節前文所述。

- CreateOpsItem
- CreateResourceDataSync
- DescribeOpsItems
- DeleteResourceDataSync
- GetOpsItem
- GetOpsSummary
- ListResourceDataSync
- UpdateOpsItem
- UpdateResourceDataSync

您可以視需要將下列內嵌政策新增至帳戶、群組或角色，以指定唯讀許可。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:GetOpsItem",
        "ssm:GetOpsSummary",
        "ssm:DescribeOpsItems",
        "ssm:GetServiceSetting",
        "ssm:ListResourceDataSync"
      ],
      "Resource": "*"
    }
  ]
}
```

如需有關建立和編輯 IAM 政策的詳細資訊，請參閱《IAM 使用者指南》中的建立 IAM 政策。如需如何將此政策指派給 IAM 群組的資訊，請參閱將政策連接到 IAM 群組。

使用以下項目建立許可並將許可新增至使用者、群組或角色：

```
{
```

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

視您在組織中使用的身分應用程式而定，您可以選取下列任何選項來設定使用者存取權。

若要提供存取權，請新增權限至您的使用者、群組或角色：

- 中的使用者和群組 AWS IAM Identity Center：

  建立權限合集。請按照 AWS IAM Identity Center 使用者指南 中的 建立權限合集 說明進行操作。
- 透過身分提供者在 IAM 中管理的使用者：

  建立聯合身分的角色。遵循「IAM 使用者指南」的為第三方身分提供者 (聯合) 建立角色中的指示。
- IAM 使用者：
  - 建立您的使用者可擔任的角色。請按照「IAM 使用者指南」的為 IAM 使用者建立角色中的指示。
  - (不建議) 將政策直接附加至使用者，或將使用者新增至使用者群組。請遵循 IAM 使用者指南的新增許可到使用者 (主控台) 中的指示。

使用標籤限制存取 OpsItems

您也可以使用指定標籤的內嵌 IAM 政策，限制存取 OpsItems。以下範例會指定 Department 的標籤鍵和 Finance 的標籤值。使用此政策，使用者只能呼叫 GetOpsItem API 操作，檢視之前以 Key=Department 和 Value=Finance 標記的 OpsItems。使用者無法檢視任何其他 OpsItems。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:GetOpsItem"
             ],
      "Resource": "*"
      ,
      "Condition": { "StringEquals": { "ssm:resourceTag/Department": "Finance" } }
    }
  ]
}
```

以下範例會指定檢視和更新 OpsItems 的 API 操作。此政策還會指定兩組標籤金鑰/值對：Department-Finance 和 Project-Unity。

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "ssm:GetOpsItem",
            "ssm:UpdateOpsItem"
         ],
         "Resource":"*",
         "Condition":{
            "StringEquals":{
               "ssm:resourceTag/Department":"Finance",
               "ssm:resourceTag/Project":"Unity"
            }
         }
      }
   ]
}
```

如需將標籤新增到 OpsItem 的資訊，請參閱 手動建立 OpsItems。

了解整合式設定建立的預設 EventBridge 規則

在 Explorer 和 OpsCenter 的整合式設定程序期間，可以選擇根據 Amazon EventBridge 偵測到的事件啟用多個預設規則。偵測到這些事件時，系統會自動在 AWS Systems Manager OpsCenter 中建立 OpsItems。

例如，EC2 自動擴展執行個體終止失敗時，規則 `SSMOpsItems-Autoscaling-instance-termination-failure` 會導致建立 OpsItem。

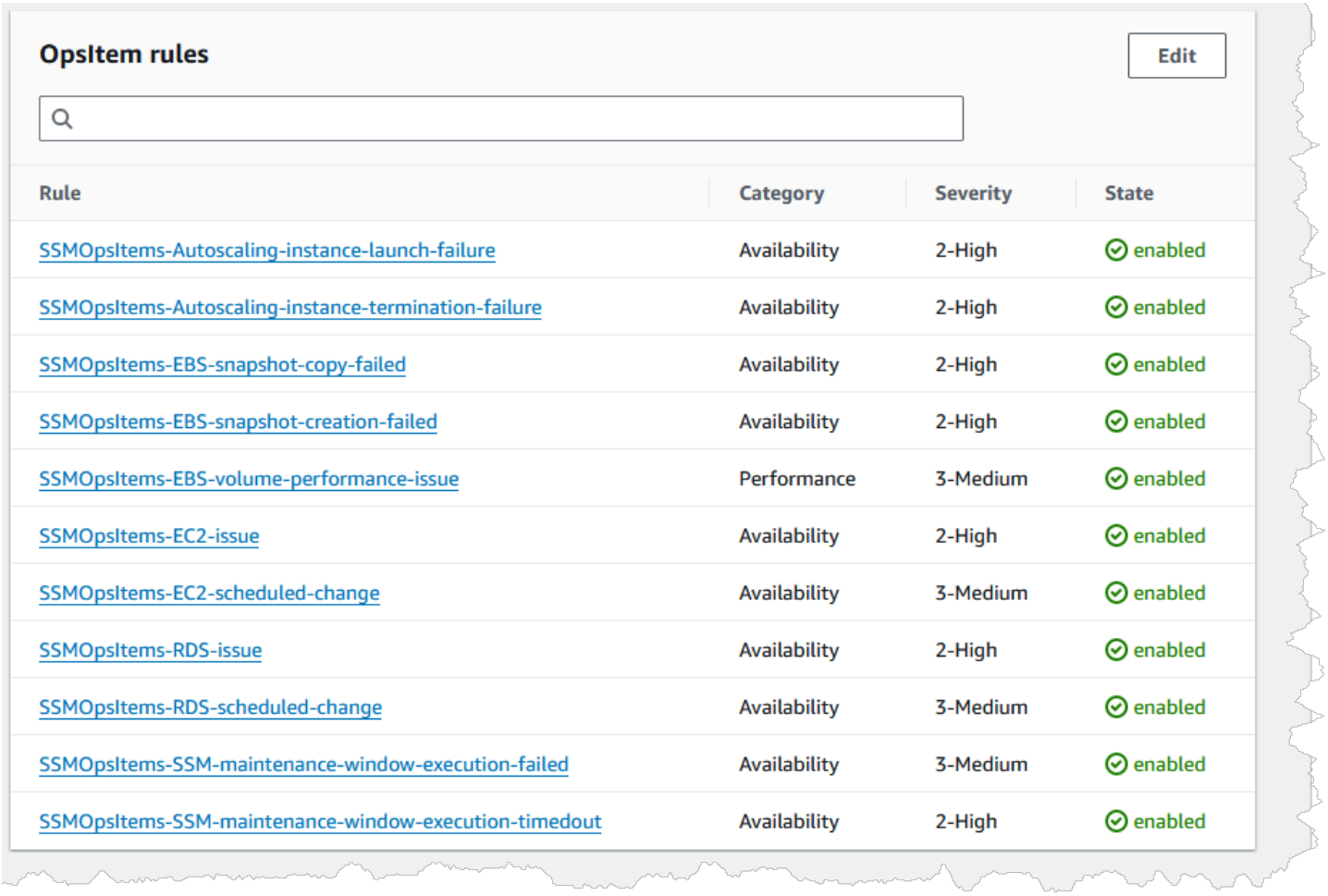
Systems Manager 維護時段無法成功完成時，規則 `SSMOpsItems-SSM-maintenance-window-execution-failed` 會導致建立 OpsItem。

如需可以在設定程序期間啟用的所有 EventBridge 規則的設定指示和說明，請參閱 設定 OpsCenter。

如果您不希望 EventBridge 為這些事件建立 OpsItems，可以選擇不在整合式設定中啟用此選項。如果您願意，您可以將 OpsCenter 指定為特定 EventBridge 事件的目標。如需詳細資訊，請參閱設定 EventBridge 規則以建立 OpsItems。

您可以在 OpsCenter 設定頁面中選擇 OpsCenter、設定，然後在OpsItem規則區域中選擇編輯，以停用預設規則或變更其類別和嚴重性等級。

您也可以在 Systems Manager 主控台中編輯指派給根據這些規則建立的個別 OpsItem 的類別或嚴重性。如需相關資訊，請參閱 編輯 OpsItem。



設定 Explorer 的委派管理員

如果您使用資源資料同步來彙總來自多個 AWS 區域 和 帳戶的 AWS Systems Manager Explorer 資料 AWS Organizations，建議您為 設定委派管理員Explorer。委派系統管理員可透過下列方式改善 Explorer 安全性。

- 限制 Explorer 管理員數目，這些管理員可建立或刪除多帳戶以及同步到個別 AWS 帳戶的區域資源資料。
- 您不再需要登入 AWS Organizations 管理帳戶，即可在 中管理資源資料同步Explorer。

委派管理員可以使用 主控台、 SDK、 AWS Command Line Interface (AWS CLI) 或 使用以下Explorer 資源資料同步 APIs AWS Tools for Windows PowerShell：

- CreateResourceDataSync
- DeleteResourceDataSync

- ListResourceDataSync
- UpdateResourceDataSync

委派管理員可以從主控台或使用 SDK、 AWS CLI或 等程式設計工具來搜尋、篩選和彙總Explorer資料 AWS Tools for Windows PowerShell。搜尋、篩選和資料彙總會使用 GetOpsSummary API 操作。

委派管理員可以為整個組織或組織單位的子集建立最多五個資源資料同步。委派管理員所建立的資源資料同步只能在委派管理員帳戶中使用。您無法在 AWS Organizations 管理帳戶中檢視同步或彙總的資料。

> **Note**
>
> 您無法使用委派的管理員帳戶在選擇加入 AWS 區域中建立資源資料同步。您必須使用 AWS Organizations 管理帳戶。

如需資源資料同步的相關資訊，請參閱 設定 Systems Manager Explorer 以顯示來自多個帳戶和區域的資料。如需 的詳細資訊 AWS Organizations，請參閱AWS Organizations 《 使用者指南》中的什麼是 AWS Organizations？。

主題

- 開始之前
- 設定 Explorer 委派管理員
- 取消註冊 Explorer 委派管理員

開始之前

下列清單包含有關 Explorer 委派管理的重要資訊。

- 您只能委派一個帳戶進行 Explorer 管理。
- 您指定為 Explorer 委派管理員的帳戶 ID 必須列為 AWS Organizations中的成員帳戶。如需詳細資訊，請參閱AWS Organizations 《 使用者指南》中的AWS 帳戶 在組織中建立 。
- 委派管理員可以使用主控台中的所有Explorer資源資料同步 API 操作，或使用程式設計工具，例如 SDK、 AWS Command Line Interface (AWS CLI) 或 AWS Tools for Windows PowerShell。資源資料同步 API 操作包括下列項目：CreateResourceDataSync、DeleteResourceDataSync、ListResourceDataSync 和 UpdateResourceDataSync。

- 委派管理員可以在主控台中搜尋、篩選和彙總 Explorer 資料，或使用程式設計工具 (例如 SDK、AWS CLI或 AWS Tools for Windows PowerShell)。搜尋、篩選和資料彙總會使用 GetOpsSummary API 操作。
- 委派管理員所建立的資源資料同步只能在委派管理員帳戶中使用。您無法檢視 AWS Organizations 管理帳戶中的同步或彙總資料。
- 委派管理員最多可建立五個資源資料同步。
- 委派管理員可以為 中的整個組織 AWS Organizations 或組織單位的子集建立資源資料同步。

設定 Explorer 委派管理員

請使用下列程序來註冊 Explorer 委派管理員。

註冊 Explorer 委派管理員

1. 登入您的 AWS Organizations 管理帳戶。
2. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
3. 在導覽窗格中，選擇 Explorer。
4. 選擇設定。
5. 在 Delegated administrator for Explorer (Explorer 的委派管理員) 區段中，確認您已設定必要的服務連結角色和服務存取選項。如有必要，請選擇 Create role (建立角色) 和 Enable access (啟用存取) 按鈕設定這些選項。
6. 針對帳戶 ID，輸入 AWS 帳戶 ID。此帳戶必須是 中的成員帳戶 AWS Organizations。
7. 選擇 Register delegated administrator (註冊委派管理員)。

委派管理員現在可存取 建立資源資料同步頁面上的選項中包含來自我的 AWS Organizations 組態的所有帳戶和選取組織單位 AWS Organizations。

取消註冊 Explorer 委派管理員

請使用下列程序來取消註冊 Explorer 委派管理員。委派管理員帳戶只能由 AWS Organizations 管理帳戶取消註冊。取消註冊委派管理員帳戶時，系統會刪除委派管理員建立的所有 AWS Organizations 資源資料同步。

取消註冊 Explorer 委派管理員

1. 登入您的 AWS Organizations 管理帳戶。

2. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
3. 在導覽窗格中，選擇 Explorer。
4. 選擇設定。
5. 在 Delegated administrator for Explorer (Explorer 的委派管理員) 區段，選擇 Deregister (取消註冊)。系統會顯示警告。
6. 輸入帳戶 ID，然後選擇 Remove (移除)。

帳戶不再具有 AWS Organizations 資源資料同步 API 操作的存取權。系統會刪除帳戶建立的所有 AWS Organizations 資源資料同步。

設定 Systems Manager Explorer 以顯示來自多個帳戶和區域的資料

AWS Systems Manager 使用整合的設定體驗來協助您開始使用 AWS Systems Manager Explorer 和 AWS Systems Manager OpsCenter 。完成整合設定後，Explorer 和 OpsCenter 會自動同步資料。更具體地說，這些工具OpsItems會同步 OpsData 和 ， AWS 區域 適用於 AWS 帳戶 ，以及您在完成整合設定時使用的 。如果您希望彙整來自其他帳戶和區域的 OpsData 和 OpsItems，您必須建立資源資料同步，如本主題所說明。

> **Note**
>
> 如需整合式設定的詳細資訊，請參閱開始使用 Systems Manager Explorer 和 OpsCenter。

主題

- 了解 Explorer 的資源資料同步
- 了解多個帳戶和區域資源資料同步
- 刪除資源資料同步

了解 Explorer 的資源資料同步

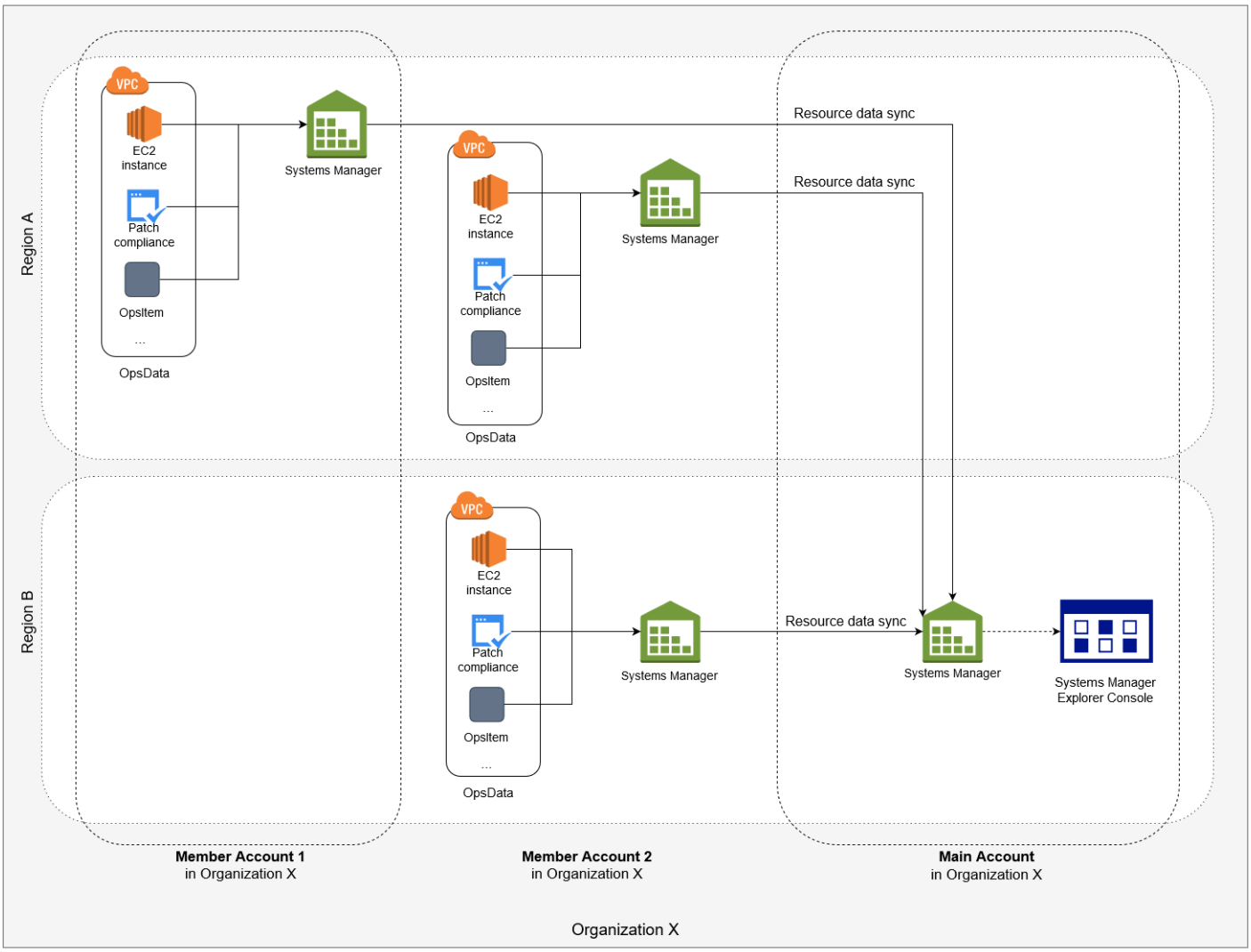
Explorer 的資源資料同步提供兩種彙整選項：

- 單一帳戶/多個區域：您可以設定 Explorer 彙整多個 AWS 區域的 OpsItems 和 OpsData 資料，但資料集會受限於目前的 AWS 帳戶。

- 多帳戶/多區域：您可以設定 Explorer 彙整多個 AWS 區域 和帳戶的資料。此選項需要您設定 AWS Organizations。在您設定和設定之後 AWS Organizations，您可以Explorer依組織單位 (OU) 或整個組織在 中彙總資料。Systems Manager 會將資料彙整至 AWS Organizations 管理帳戶，再於 Explorer 中進行顯示。如需詳細資訊，請參閱AWS Organizations 《 使用者指南》中的什麼是 AWS Organizations？。

> **Warning**
>
> 如果您設定 Explorer 以彙總 AWS Organizations中的組織資料，系統會啟用組織中所有成員帳戶的 OpsData。在所有成員帳户中啟用 OpsData 來源會增加呼叫 CreateOpsItem 和 GetOpsSummary 等 OpsCenter API 的次數。呼叫這些 API 動作需支付費用。

下圖顯示設為使用 AWS Organizations的資源資料同步。在此案例中，使用者具備兩個在 AWS Organizations中定義的帳戶。資源資料同步會將來自兩個帳戶和多個 AWS 區域 的資料彙總到 AWS Organizations 管理帳戶中，然後顯示在 中Explorer。



了解多個帳戶和區域資源資料同步

本節說明有關使用 AWS Organizations的多個帳戶和多個區域資源資料同步的重要詳細資訊。具體而言，如果您選擇了 Create resource data sync (建立資源資料同步) 頁面中的以下其中一個選項，則本節的資訊適用：

- 包含我的 AWS Organizations 組態中的所有帳戶
- 在 中選取組織單位 AWS Organizations

如果您不打算使用以下其中一個選項，您可以略過本節。

當您在 SSM 主控台中建立資源資料同步時，如果您選擇其中一個 AWS Organizations 選項，則 Systems Manager 會自動為組織 （或所選組織單位 AWS 帳戶）中的所有 ，允許所選區域中的所有 OpsData 來源。例如，即使您尚未在區域中Explorer開啟 ，如果您為資源資料同步選取 AWS

Organizations 選項，則 Systems Manager 會自動從該區域收集 OpsData。若要在不允許 OpsData 來源的情況下建立資源資料同步，請在建立資料同步時將 EnableAllOpsDataSources 指定為 false。如需詳細資訊，請參閱《Amazon EC2 Systems Manager API 參考》中 ResourceDataSyncSource 資料類型的 `EnableAllOpsDataSources` 參數詳細資訊。

如果您未選擇資源資料同步的其中一個 AWS Organizations 選項，則必須在Explorer您要存取資料的每個帳戶和區域中完成整合式設定。如果沒有這樣做，Explorer 將不會顯示未完成整合式設定的帳戶和區域的 OpsData 和 OpsItems。

如果您將子帳戶新增至您的組織，Explorer 會自動允許該帳戶的所有 OpsData 來源。如果您稍後從組織中移除子女帳戶，Explorer 會繼續從帳戶收集 OpsData。

如果您更新使用其中一個 AWS Organizations 選項的現有資源資料同步，系統會提示您核准收集受變更影響之所有帳戶和區域的所有 OpsData 來源。

如果您將新服務新增至 AWS 帳戶，且 為該服務Explorer收集 OpsData，則 Systems Manager 會自動設定 Explorer以收集該 OpsData。例如，如果您的組織在您先前建立資源資料同步 AWS Trusted Advisor 時未使用 ，但您的組織註冊此服務， Explorer會自動更新您的資源資料同步以收集此 OpsData。

Important

請注意下列有關多個帳戶和區域資源資料同步的重要資訊：

- 刪除資源資料同步不會關閉 Explorer 中的 OpsData 來源。
- 若要檢視OpsItems來自多個帳戶的 OpsData 和 ，您必須開啟 AWS Organizations 所有功能模式，而且必須登入 AWS Organizations 管理帳戶。
- 大多數 預設為 AWS 區域 作用中 AWS 帳戶，但只有在您手動選取特定區域時才會啟用。這些區域稱為選擇加入區域。根據預設，Explorer跨帳戶/跨區域資源資料同步不支援選擇加入區域中的資料彙總。已於 2025 年 6 月 30 日新增下列選擇加入區域的支援。
  - 歐洲 (米蘭)
  - 非洲 (開普敦)
  - Middle East (Bahrain)
  - 亞太區域 (香港)

  請注意，您無法使用委派的管理員帳戶在選擇加入區域中建立資源資料同步。您必須使用 AWS Organizations 管理帳戶。

刪除資源資料同步

在設定 Explorer 的資源資料同步之前，請留意以下詳細資訊。

- Explorer 支援最多五個資源資料同步。
- 為區域建立資源資料同步之後，您無法變更該同步的帳戶選項。例如，如果您在 us-east-2（俄亥俄）區域中建立同步，並選擇僅包含目前帳戶選項，則您無法稍後編輯該同步，並從我的 AWS Organizations 組態選項中選擇包含所有帳戶。相反地，您必須刪除第一個資源資料同步，並建立新的資源資料同步。
- Explorer 中檢視的 OpsData 是唯讀的。

使用下列程序可為 Explorer 建立資源資料同步。

建立資源資料同步

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Explorer。
3. 選擇設定。
4. 在 Configure resource data sync (設定資源資料同步) 區段中，選擇 Create resource data sync (建立資源資料同步)。
5. 對於 Resource data sync name (資源資料同步名稱)，輸入名稱。
6. 在 Add accounts (新增帳戶) 區段中，選擇一個選項。

   > **Note**
   >
   > 若要使用任一 AWS Organizations 選項，您必須登入 AWS Organizations 管理帳戶，或必須登入Explorer委派管理員帳戶。如需委派管理員帳戶的詳細資訊，請參閱 設定 Explorer 的委派管理員。

7. 在 Regions to include (要包含的區域) 區段中，選擇下列其中一個選項。
   - 選擇所有目前和未來的區域，以自動同步來自所有目前 AWS 區域 和未來上線之新區域的資料。
   - 選擇所有區域以自動同步所有目前資料 AWS 區域。
   - 個別選擇您要包含的區域。

8. 選擇建立資源資料同步。

建立資源資料同步之後，可能需要幾分鐘的時間，才能填入 Explorer 的資料。您可從 Explorer 中的 Select a resource data sync (選取資源資料同步) 清單中進行選擇，從而檢視同步。

使用 Explorer

本節包含如何透過變更小工具配置以及變更儀表板中顯示的資料來自訂 AWS Systems Manager Explorer 的相關資訊。

目錄

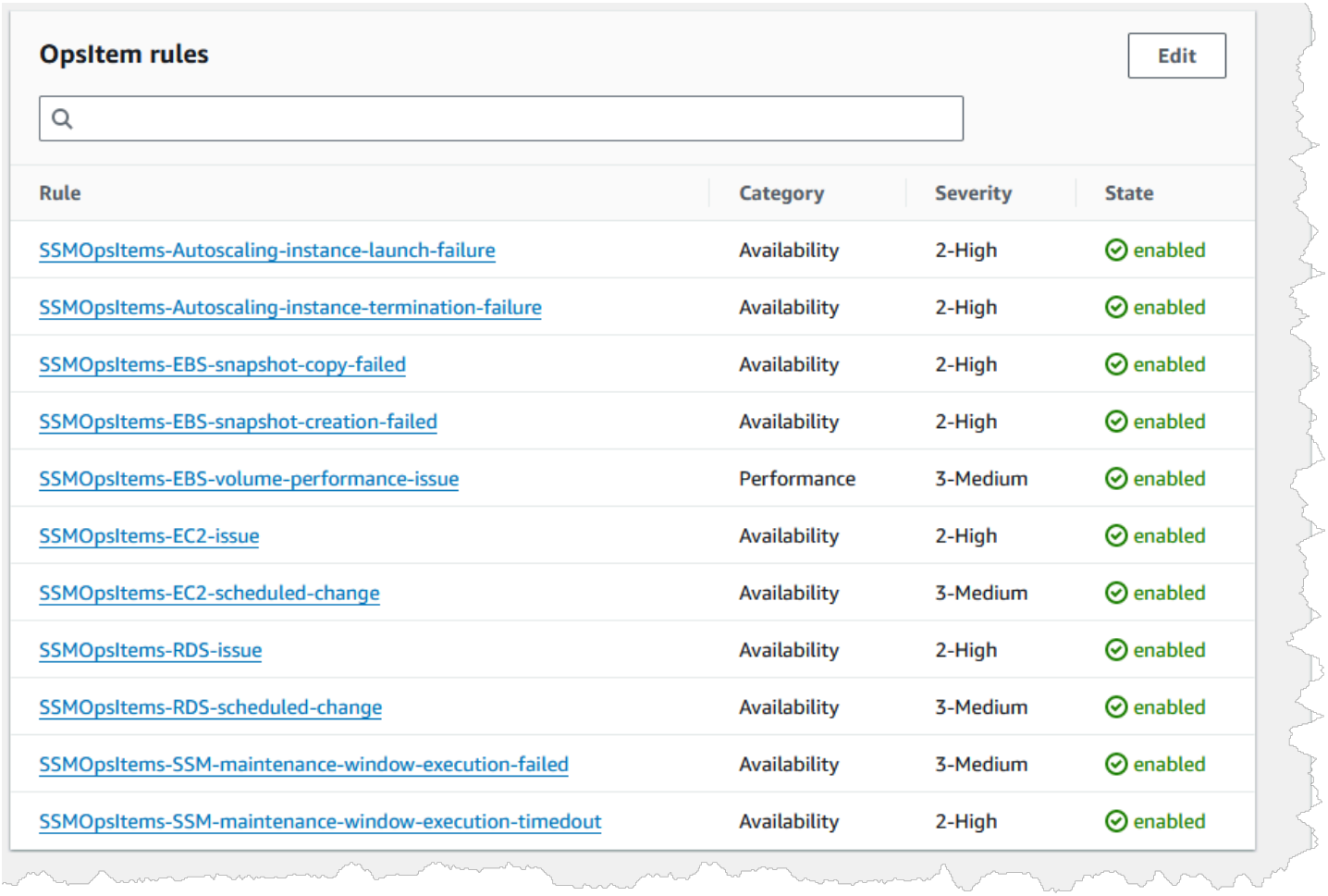
- 編輯為 Explorer 建立的 EventBridge 規則
- 編輯 Systems Manager Explorer 資料來源
- 自訂 Explorer 顯示
- 在 Explorer 中從 AWS Security Hub 接收調查結果
- 刪除 Systems Manager Explorer 資源資料同步

編輯為 Explorer 建立的 EventBridge 規則

完成整合式設定時，系統允許在 Amazon EventBridge 中啟用十幾個以上的規則。這些規則會自動在OpsItems中建立 AWS Systems Manager OpsCenter。 AWS Systems Manager Explorer 然後顯示有關 的彙總資訊OpsItems。

每個規則都包含預設的 Category (類別) 和 Severity (嚴重性) 值。當系統從事件建立 OpsItems 時，它會自動指派預設的 Category (類別) 和 Severity (嚴重性)。

> **Important**
>
> 目前，您無法編輯預設規則的 Category (類別) 和 Severity (嚴重性) 值，但您可以在透過預設規則建立的 OpsItems 上編輯這些值。



編輯用於建立 OpsItems 的預設規則

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Explorer。
3. 選擇設定。
4. 在 OpsItems 規則規則區段中，選擇 Edit (編輯)。
5. 展開 CWE rules (CWE 規則)。
6. 清除您不想使用的規則旁邊的核取方塊。
7. 使用 Category (類別) 和 Severity (嚴重性) 清單來變更規則的此資訊。
8. 選擇 Save (儲存)。

您的變更會在下次系統建立 OpsItem 時生效。

編輯 Systems Manager Explorer 資料來源

對於預設可用的 AWS 區域 ，AWS Systems Manager Explorer 會顯示來自下列來源的資料。您可以編輯 Explorer 設定來新增或移除資料來源：

- Amazon Elastic Compute Cloud (Amazon EC2)
- AWS Systems Manager 庫存
- AWS Systems Manager OpsCenter OpsItems
- AWS Systems Manager Patch Manager 修補程式合規
- AWS Systems Manager State Manager 關聯合規
- AWS Trusted Advisor
- AWS Compute Optimizer
- AWS 支援 中心案例
- AWS Config 規則和資源合規
- AWS Security Hub 問題清單

> **Note**
>
> 對於亞太區域（大阪）區域，Explorer不會顯示來自 AWS Compute Optimizer 和 AWS Security Hub 調查結果的資料。

對於AWS 選擇加入區域，Explorer會顯示來自下列來源的資料：

- Amazon Elastic Compute Cloud (Amazon EC2)
- AWS Systems Manager 庫存
- AWS Systems Manager OpsCenter OpsItems
- AWS Systems Manager Patch Manager 修補程式合規
- AWS Systems Manager State Manager 關聯合規
- AWS Trusted Advisor
- AWS 支援 中心案例
- AWS Config 規則和資源合規

> **Note**
>
> - 若要在 中檢視 支援 中心案例Explorer，您必須設定企業或企業帳戶 支援。
> - 您無法設定 Explorer 來停止顯示 OpsCenter OpsItem 資料。

開始之前

確認您已安裝並設定將資料填入 Explorer 小工具的服務。如需詳細資訊，請參閱設定 Explorer 的相關服務。

編輯資料來源

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Explorer。
3. 選擇設定，然後選擇設定儀表板索引標籤。
4. 在 OpsData 來源區段的狀態欄位中，根據要檢視的資料開啟或關閉來源。

自訂 Explorer 顯示

您可以使用drag-and-drop功能在 AWS Systems Manager Explorer 中自訂小工具配置。您也可以使用篩選器自訂在 Explorer 中顯示的 OpsData 和 OpsItems，如本主題所述。

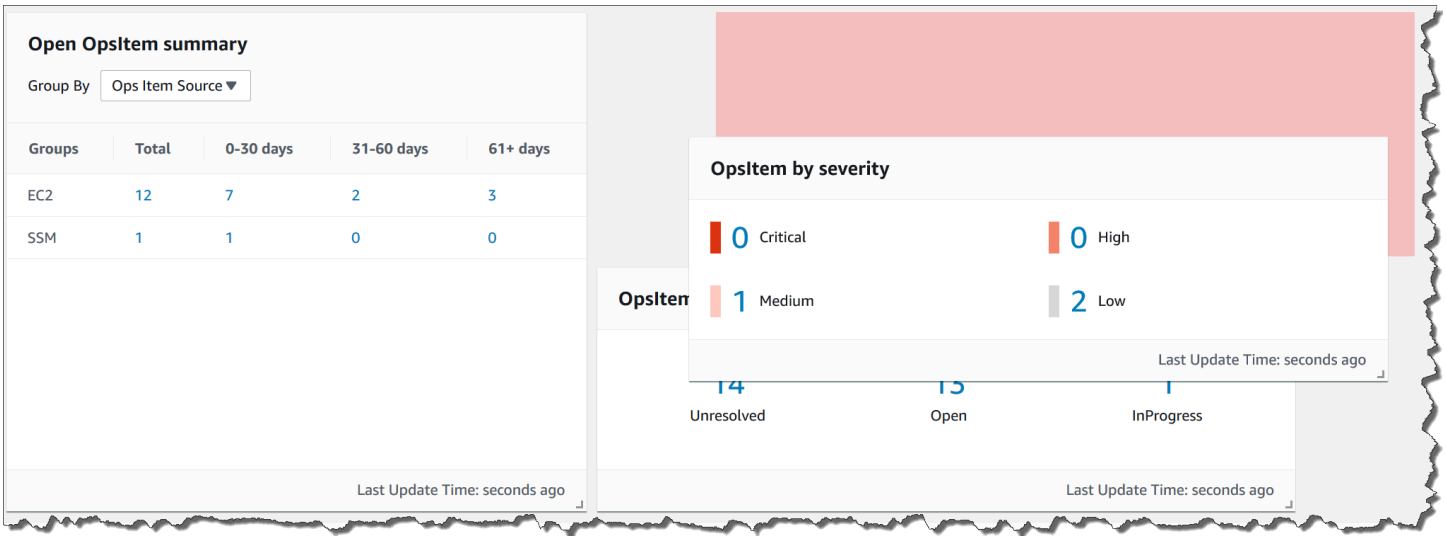
在您自訂小工具配置之前，請驗證您要檢視的小工具目前已顯示在 Explorer 中。若要檢視 Explorer 中的某些小工具 (例如 AWS Config 合規小工具)，必須在 Configure dashboard (設定儀表板) 頁面上將其啟用。

啟用小工具以顯示在 Explorer 中

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Explorer。
3. 選擇 Dashboard actions (儀表板動作)、Configure dashboard (設定儀表板)。
4. 選擇 Configure Dashboard (設定儀表板) 標籤。
5. 選擇 Enable all (全部啟用) 或開啟個別小工具或資料來源。

6. 選擇 Explorer 檢視您的變更。

若要在 Explorer 中自訂小工具配置，請選擇要移動的小工具。按住 Widget 的名稱，然後將它拖曳到新位置。



對您要重新定位的每個 Widget 重複此程序。

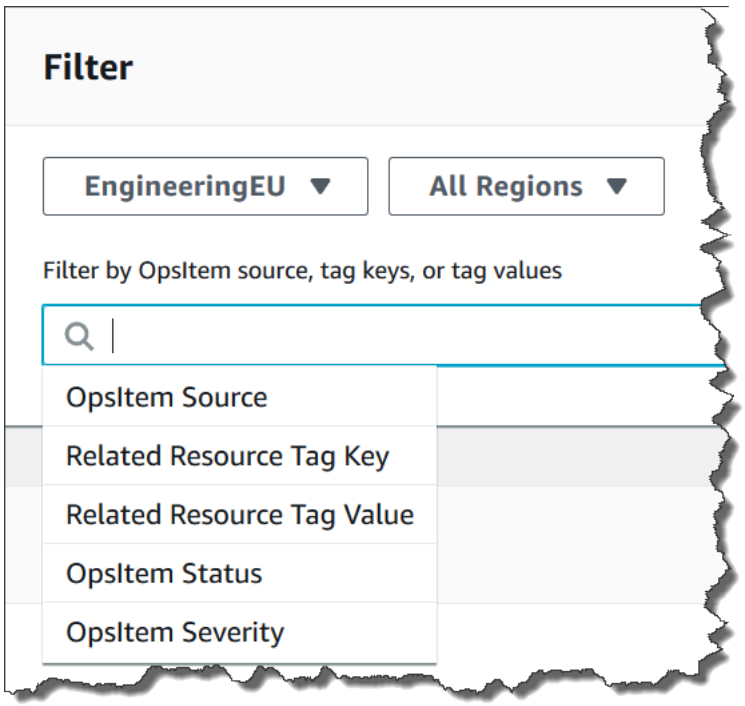
如果您決定不要新的配置，請選擇重設配置，將所有小工移回原始位置。

使用篩選器變更在 Explorer 中顯示的資料

根據預設， Explorer會顯示目前 AWS 帳戶 和目前區域的資料。如果您建立一或多個資源資料同步，您可以使用篩選器來變更哪個同步處於作用中。然後，您可以選擇顯示特定區域或所有區域的資料。您也可以使用搜尋列來依據不同的 OpsItem 和 key-tag 條件篩選。

使用篩選器變更在 Explorer 中顯示的資料

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Explorer。
3. 在 Filter (篩選) 區段中，使用 Select a resource data sync (選取資源資料同步清單) 來選擇同步。
4. 使用 Regions (區域) 清單來選擇特定 AWS 區域 或選擇 All Regions (所有區域)。
5. 選擇搜尋列，然後選擇要篩選資料的依據條件。



6. 按 Enter。

如果您關閉並重新開啟頁面，Explorer 會保留您選取的篩選選項。

在 Explorer 中從 AWS Security Hub 接收調查結果

AWS Security Hub 提供安全狀態的完整檢視 AWS。此服務會從各 AWS 帳戶、服務和支援的第三方產品收集安全資料 (稱為調查結果)。Security Hub 調查結果可協助您檢查環境是否符合安全業界標準和最佳實務、分析安全趨勢，並確定高優先順序的安全問題。

Security Hub 會將調查結果傳送至 Amazon EventBridge，EventBridge 會使用事件規則將調查結果傳送至 Explorer。啟用整合之後，如此處所述，您可以在 Explorer 小工具中檢視 Security Hub 調查結果，並在 OpsCenter OpsItems 中檢視調查結果詳細資訊。小工具可根據嚴重性提供所有 Security Hub 調查結果的摘要。Security Hub 中的新調查結果通常會在建立幾秒鐘後顯示在 Explorer 中。

> **Warning**
>
> 記下以下重要資訊：
>
> - Explorer 已與 Systems Manager 中的OpsCenter工具 整合。啟用 Explorer 與 Security Hub 的整合之後，OpsCenter 會自動為 Security Hub 調查結果建立 OpsItems。根據您的 AWS 環境，啟用整合可能會產生大量 OpsItems，但需付費。

繼續之前，請閱讀有關 OpsCenter 與 Security Hub 整合的相關內容。本主題包含有關對調查結果和 OpsItems 的變更和更新將如何收費的詳細資訊。如需詳細資訊，請參閱了解與的OpsCenter整合 AWS Security Hub。如需 OpsCenter 定價資訊，請參閱 AWS Systems Manager 定價。

- 如果您以系統管理員帳戶登入 Explorer 並建立資源資料同步，則會自動為系統管理員和同步中的所有成員帳戶啟用 Security Hub 整合。啟用之後，OpsCenter 會自動為建立 Security Hub 調查結果建立需要付費的 OpsItems。如需建立資源資料同步的相關資訊，請參閱 設定 Systems Manager Explorer 以顯示來自多個帳戶和區域的資料。

Explorer 接收之調查結果的類型

Explorer 可接收來自 Security Hub 的所有調查結果 當您開啟 Security Hub 預設設定時，您可以在 Explorer 小工具中查看所有根據嚴重性的調查結果。根據預設，Explorer 可為「關鍵」和「高」安全性調查結果建立 OpsItems。您可以手動設定 Explorer，為「中等」和「低」安全性調查結果建立 OpsItems。

雖然 Explorer 不會針對資訊性調查結果建立 OpsItems，但您可以在 Security Hub 調查結果摘要小工具中檢視資訊性操作資料 (OpsData)。Explorer 會為所有調查結果建立 OpsData，不論調查結果的嚴重性如何。如需有關 Security Hub 嚴重性等級的詳細資訊，請參閱 AWS Security Hub API 參考中的安全性一節。

啟用整合

此節說明如何啟用和設定 Explorer 以開始接收 Security Hub 調查結果。

開始之前

在您設定 Explorer 之前，請完成以下任務，以開始接收 Security Hub 調查結果。

- 啟用和設定 Security Hub。如需詳細資訊，請參閱《AWS Security Hub 使用者指南》中的建立 Security Hub。
- 登入 AWS Organizations 管理帳戶。Systems Manager 需要存取 AWS Organizations ，以從 Security Hub 調查結果建立 OpsItems。登入管理帳戶後，系統會提示您選取 Explorer Configure dashboard (設定儀表板) 標籤上的 Enable access (啟用存取) 按鈕 設定儀表板索引標籤，如下列程序所述。如果您未登入 AWS Organizations 管理帳戶，則無法允許存取，Explorer也無法OpsItems 從 Security Hub 調查結果建立 。

若要開始接收 Security Hub 調查結果

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Explorer。
3. 選擇設定。
4. 選取 Configure dashboard (設定儀表板) 標籤。
5. 選取 AWS Security Hub。
6. 選取 Disabled (已停用) 滑桿，以開啟 AWS Security Hub。

   預設會顯示「關鍵」和「高」安全性調查結果。若要顯示「中」和「低」安全性調查結果，請選取中、低旁邊的已停用滑桿。
7. 在 OpsItems Security Hub 調查結果建立的 OpsItems 區段中，選擇啟用存取。如果您沒有看到此按鈕，請登入 AWS Organizations 管理帳戶並返回此頁面以選取按鈕。

如何檢視 Security Hub 的調查結果

下列處理程序描述如何檢視 Security Hub 調查結果。

若要檢視 Security Hub 調查結果

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Explorer。
3. 尋找 AWS Security Hub 調查結果摘要小工具。這會顯示您的 Security Hub 調查結果。您可以選取嚴重性等級，以檢視對應的 OpsItem 的詳細說明。

如何停止接收調查結果

下列處理程序描述如何停止接收 Security Hub 調查結果。

若要停止接收 Security Hub 調查結果

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Explorer。
3. 選擇設定。

4. 選取 Configure dashboard (設定儀表板) 標籤。
5. 選取 Enabled (已啟用) 滑桿，以關閉 AWS Security Hub。

> **Important**
>
> 如果停用 Security Hub 調查結果的選項在主控台中呈現灰色，可以在 AWS CLI中執行下列命令來停用此設定。登入 AWS Organizations 管理帳戶或 Systems Manager 委派管理員帳戶時，您必須執行 命令。針對 `region` 參數，指定您要在 中停止接收 Security Hub 問題清單 AWS 區域 的 Explorer。
>
> ```
> aws ssm update-service-setting --setting-id /ssm/opsdata/SecurityHub --setting-value Disabled --region AWS ##
> ```
>
> 範例如下。
>
> ```
> aws ssm update-service-setting --setting-id /ssm/opsdata/SecurityHub --setting-value Disabled --region us-east-1
> ```

刪除 Systems Manager Explorer 資源資料同步

在 AWS Systems Manager Explorer 中，您可以透過建立資源資料同步，OpsItems從其他帳戶和區域彙總 OpsData 和 。

您無法變更資源資料同步的帳戶選項。例如，如果您在 us-east-2 （俄亥俄） 區域中建立同步，且您選擇僅包含目前帳戶選項，則您無法稍後編輯該同步，並從我的 AWS Organizations 組態選項中選擇包含所有帳戶。相反地，您必須刪除資源資料同步，並建立新的資源資料同步，如下列程序所述。

刪除資源資料同步

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Explorer。
3. 選擇設定。
4. 在 Configure resource data sync (設定資源資料同步) 區段中，選擇您要刪除的資源資料同步。

5. 選擇 Delete (刪除)。

從 Systems Manager Explorer 匯出 OpsData

您可以從 AWS Systems Manager Explorer 將 5，000 個 OpsData 項目匯出為逗號分隔值 (.csv) 檔案到 Amazon Simple Storage Service (Amazon S3) 儲存貯體。Explorer 使用 AWS-ExportOpsDataToS3 自動化執行手冊匯出 OpsData。匯出 OpsData 時，系統會顯示自動化執行手冊頁面，您可以在其中指定詳細資訊，例如要匯出的 assumeRole、Amazon S3 儲存貯體名稱、SNS 主題 ARN 以及欄位。

若要匯出 OpsData：

- 步驟 1：指定 SNS 主題
- 步驟 2：(選用) 設定資料匯出
- 步驟 3：匯出 OpsData

步驟 1：指定 SNS 主題

設定資料匯出時，您必須指定 Amazon Simple Notification Service (Amazon SNS) 主題，該主題存在於您要匯出資料的相同 AWS 區域 位置。匯出完成時，Systems Manager 會將通知傳送到 Amazon SNS 主題。如需建立 Amazon SNS 主題的相關資訊，請參閱建立 Amazon SNS 主題。

步驟 2：(選用) 設定資料匯出

您可以從設定或將操作資料匯出至 S3 儲存貯體頁面進行資料匯出設定。

從 Explorer 設定資料匯出

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Explorer。
3. 選擇設定。
4. 在 Configure data export (設定資料匯出) 區段中，選擇 Edit (編輯)。
5. 若要將資料匯出檔案上傳至現有的 Amazon S3 儲存貯體，請選擇選取現有的 S3 儲存貯體，然後從清單中選擇儲存貯體。

   若要將資料匯出檔案上傳至新的 Amazon S3 儲存貯體，請選擇建立新的 S3 儲存貯體，然後輸入您要用於新儲存貯體的名稱。

> **Note**
>
> 在 Explorer 中，您只能從您第一次設定這些設定的頁面編輯 Amazon S3 儲存貯體名稱和 Amazon SNS 主題 ARN。如果您從設定頁面設定 Amazon S3 儲存貯體和 Amazon SNS 主題 ARN，則只能從設定頁面修改這些設定。

6. 針對選取 Amazon SNS 主題 ARN，選擇匯出完成時要通知的主題。
7. 選擇建立。

步驟 3：匯出 OpsData

當您匯出Explorer資料時，Systems Manager 會建立名為 的 AWS Identity and Access Management (IAM) 角色`AmazonSSMExplorerExportRole`。此角色使用下列 IAM 政策。

```
{
      "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": [
                "{{SnsTopicArn}}"
            ]
        },
        {
            "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": [
                "*"
            ]
        }
    ]
}
```

角色包括下列信任實體。

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

從 Explorer 匯出 OpsData

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Explorer。
3. 選擇Explorer小工具中的資料連結。例如，在 中OpsItems依狀態小工具選擇未解決或開啟問題。或OpsItem依嚴重性小工具在 中選擇關鍵或高資料連結。 Explorer會開啟所選資料集的 OpsData 頁面。
4. 選擇開始匯出。

   > **Note**
   >
   > 第一次匯出 OpsData 時，系統會為匯出建立擔任角色。您無法修改預設擔任角色。

5. 針對 S3 儲存貯體名稱，選擇一個現有的儲存貯體。如果需要，可選擇建立來建立一個 Amazon S3 儲存貯體。

   如果您無法變更某個 S3 儲存貯體的名稱，則表示您此前是從設定頁面設定儲存貯體名稱。您只能從設定頁面變更該儲存貯體的名稱。

> **Note**
>
> 在 Explorer 中，您只能從您第一次設定這些設定的頁面編輯 Amazon S3 儲存貯體名稱和 Amazon SNS 主題 ARN。

6. 針對 SNS 主題 ARN，請選擇一個現有的 Amazon SNS 主題 ARN，用於在下載完成時進行通知。

   如果您無法變更某個 Amazon SNS 主題 ARN，則表示您此前是從設定頁面設定 Amazon SNS 主題 ARN。您只能從設定頁面變更該主題 ARN。
7. (選用) 針對 SNS 成功訊息，請指定匯出成功完成時要顯示的成功訊息。
8. 選擇提交。系統會導覽至上一頁，並顯示按一下以檢視匯出程序狀態訊息。檢視詳細資訊。

   您可以選擇檢視詳細資訊以在 Systems Manager Automation 中檢視執行手冊的狀態和進度。

您現在可以將 OpsData 從 Explorer 匯出至指定的 Amazon S3 儲存貯體。

如果您無法使用此程序匯出資料，請確認使用者、群組或角色是否包含 `iam:CreatePolicyVersion` 和 `iam:DeletePolicyVersion` 動作的許可。如需有關將這些動作新增至使用者、群組或角色的相關資訊，請參閱《IAM 使用者指南》中的編輯 IAM 政策。

Systems Manager Explorer 故障診斷

本主題涵蓋的資訊能協助您了解如何針對 AWS Systems Manager Explorer 的常見問題進行故障診斷。

資源資料同步已停止運作

如果您的資源資料同步已停止為選擇加入 AWS 區域運作，請確認已為 AWS Organizations 管理帳戶和設定同步的成員帳戶選擇加入 區域。

大多數 預設為 AWS 區域 作用中 AWS 帳戶，但只有在您手動選取特定區域時才會啟用。這些區域稱為選擇加入區域。根據預設，Explorer跨帳戶/跨區域資源資料同步不支援選擇加入區域中的資料彙總。已於 2025 年 6 月 30 日新增下列選擇加入區域的支援。

- 歐洲 (米蘭)
- 非洲 (開普敦)
- Middle East (Bahrain)
- 亞太區域 (香港)

請注意，您無法使用委派的管理員帳戶在選擇加入區域中建立資源資料同步。您必須使用 AWS Organizations 管理帳戶。

在 設定 頁面上更新標籤Explorer後，無法在 中篩選 AWS 資源

如果您在 Explorer 中更新標籤鍵或其他資料設定，系統可能需要長達六個小時才能根據您的變更同步資料。

建立資源資料同步頁面上 AWS Organizations 的選項會反灰

只有在您設定和設定 時，才能使用建立資源資料同步頁面上的選項中包含來自我的 AWS Organizations 組態的所有帳戶和選取組織單位 AWS Organizations。 AWS Organizations如果您設定和 AWS Organizations，則 AWS Organizations 管理帳戶或Explorer委派管理員可以建立使用這些選項的資源資料同步。

如需詳細資訊，請參閱設定 Systems Manager Explorer 以顯示來自多個帳戶和區域的資料及設定 Explorer 的委派管理員。

Explorer 完全不顯示任何資料

- 確認您已在您要 Explorer 能存取和顯示資料的每個帳戶和區域中完成整合式設定。如果沒有這樣做，Explorer 將不會顯示未完成整合式設定的帳戶和區域的 OpsData 和 OpsItems。如需詳細資訊，請參閱開始使用 Systems Manager Explorer 和 OpsCenter。
- 使用 Explorer 來檢視來自多個帳戶和區域的資料時，請確認您已登入 AWS Organizations 管理帳戶或Explorer委派管理員帳戶。若要檢視來自多個帳戶和區域的 OpsData 和 OpsItems，您必須登入此帳戶。

Amazon EC2 執行個體的相關小工具未顯示資料

如果 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體的相關小工具 (例如 Instance count (執行個體計數)、Managed instances (受管執行個體) 以及 Instance by AMI (依 AMI 的執行個體) 小工具) 未顯示資料，請確認下列項目：

- 確認您已等待數分鐘。完成整合式設定之後，OpsData 可能需要數分鐘的時間才會在 Explorer 中顯示。
- 確認您已設定 AWS Config 組態記錄器。 Explorer會使用 AWS Config 組態記錄器提供的資料，將 EC2 執行個體的相關資訊填入小工具。如需詳細資訊，請參閱管理組態記錄器。
- 在 Settings (設定) 頁面上，確認 Amazon EC2 OpsData 來源處於作用中狀態。此外，請確認自啟用組態記錄器或自您對執行個體進行變更後，已超過 6 小時。在您最初啟動組態記錄器或變更執行個

體之後，Systems Manager 最多可能需要六個小時才能在 Explorer EC2 小工具 AWS Config 中顯示來自 的資料。

- 請注意，如果執行個體已停止或終止，則 Explorer 會在 24 小時後停止顯示這些執行個體。
- 確認您位於設定 Amazon EC2 執行個體的正確 AWS 區域 位置。 Explorer 不會顯示內部部署執行個體的資料。
- 如果您設定了多個帳戶和區域的資源資料同步，請確認您已登入 Organizations 管理帳戶或 Explorer 委派管理員帳戶。

修補 Widget 不會顯示資料

Non-compliant instances for patching (用於修補的不合規執行個體) 小工具只會顯示不合規之修補程式執行個體的相關資料。如果您的執行個體合規，則此 Widget 不會顯示任何資料。如果您懷疑有不合規的執行個體，請確認您已設定並設定 Systems Manager 修補，並使用 AWS Systems Manager Patch Manager 檢查修補程式合規性。如需詳細資訊，請參閱AWS Systems Manager Patch Manager。

雜項問題

Explorer 不允許您編輯或補救 OpsItems：OpsItems 檢視的跨帳戶或區域是唯讀的。只能從其家用帳戶或區域更新和補救。

AWS Systems Manager OpsCenter

OpsCenter中的工具提供了一個集中位置 AWS Systems Manager，讓營運工程師和 IT 專業人員可以檢視、調查和解決與 AWS 資源相關的操作工作項目 (OpsItems)。 OpsCenter 旨在縮短對影響 AWS 資源的問題進行解決的平均時間。 OpsItems會跨服務OpsCenter彙總和標準化OpsItems，同時提供每個 OpsItem、相關 和相關資源的關聯式調查資料。 OpsCenter也提供 Systems Manager Automation 執行手冊，供您用來快速解決問題。您可以為每個 OpsItem 指定可搜尋的自訂資料。您也可以依狀態及來源，檢視自動產生的 OpsItems 摘要報告。若要開始使用 OpsCenter，請開啟 Systems Manager 主控台。在導覽窗格中，選擇 OpsCenter。

OpsCenter 與 Amazon EventBridge 及 Amazon CloudWatch 整合。這表示您可以將這些服務設定為在 CloudWatch 警示進入 ALARM 狀態OpsCenter時，或在 EventBridge 處理來自發佈事件之任何 AWS 服務的事件時，自動在 OpsItem中建立 。將 CloudWatch 警示和 EventBridge 事件設定為自動建立 OpsItems可讓您快速診斷和修復來自單一主控台 AWS 的資源問題。

為了協助您診斷問題，每個 都OpsItem包含內容相關資訊，例如 AWS 產生 的資源名稱和 IDOpsItem、警示或事件詳細資訊、警示歷史記錄，以及警示時間軸圖表。

對於 AWS 資源， 會OpsCenter彙總來自 AWS Config、 AWS CloudTrail 日誌和 Amazon CloudWatch Events 的資訊，因此您在調查期間不需要跨多個主控台頁面進行導覽。

下列清單包含客戶設定 CloudWatch 警示來建立 AWS 的資源和指標類型OpsItems。

- Amazon DynamoDB：資料庫讀取和寫入動作達到閾值
- Amazon EC2：CPU 使用率達到閾值
- AWS 帳單：預估費用達到閾值
- Amazon EC2：執行個體未通過狀態檢查
- Amazon Elastic Block Store (EBS)：磁碟空間使用率達到閾值

下列清單包含客戶設定的 EventBridge 規則類型，以建立 OpsItems。

- AWS Security Hub：已發出安全提醒
- DynamoDB：調節事件
- Amazon EC2 Auto Scaling：無法啟動執行個體
- Systems Manager：無法執行自動化
- AWS Health：排定維護的提醒
- EC2：執行個體狀態從 `Running` 變更為 `Stopped`

OpsCenter 也已經與 .NET 和 SQL Server 的 Amazon CloudWatch Application Insights 整合。這表示您可以為您應用程式中偵測到的問題自動建立 OpsItems。您也可以OpsCenter與 整合 AWS Security Hub ，以彙整 Systems Manager 中的安全性、效能和操作問題，並對其採取行動。

操作工程師和 IT 專業人員可以使用 AWS Systems Manager 主控台中的 OpsCenter頁面、公有 API 操作、 AWS Command Line Interface (AWS CLI) AWS Tools for Windows PowerShell、 或 AWS SDKs OpsItems來建立、檢視和編輯。OpsCenter公有 API 操作也可讓您OpsCenter整合案例管理系統和運作狀態儀表板。

OpsCenter 如何為我的組織帶來益處？

OpsCenter 提供標準和統一的體驗，用於檢視、處理和修復與 AWS 資源相關的問題。標準及一致的體驗可改善修正問題、調查相關問題及培訓新營運工程師和 IT 專業人員所需的時間。標準和一致的體驗也可以降低在系統中管理和修正問題的手動錯誤數量。

尤有甚者，OpsCenter 為營運工程師和組織提供下列優勢：

- 您不再需要跨多個主控台頁面進行導覽，即可檢視、調查和解決與 AWS 資源OpsItems相關的問題。 OpsItems 會跨服務集中彙整。
- 您可以檢視 OpsItems 的服務特定或與內容相關的資料；這些資料是由 CloudWatch 警示、EventBridge 事件和 .NET 及 SQL Server 的 CloudWatch Application Insights 自動產生。
- 您可以指定與 OpsItem 相關的資源 Amazon Resource Name (ARN)。透過指定相關的資源，OpsCenter 使用內建的邏輯協助您避免建立重複的 OpsItems。
- 您可以檢視類似 OpsItems 的詳細資訊和解決資訊。
- 您可以快速檢視資訊以及執行 Systems Manager Automation Runbook 來解決問題。

OpsCenter 有哪些功能？

- 自動和手動建立 OpsItem

  OpsCenter 與 Amazon CloudWatch 整合。這表示您可以將 CloudWatch 設定為在警示進入 ALARM 狀態時或 Amazon EventBridge 處理來自任何發佈事件之 AWS 服務的事件時，自動在 OpsCenter 中建立 OpsItem。您也可以手動建立 OpsItems。

  OpsCenter 也已經與 .NET 和 SQL Server 的 Amazon CloudWatch Application Insights 整合。這表示您可以為您應用程式中偵測到的問題自動建立 OpsItems。

- 詳細和可搜尋的 OpsItems

  每個 OpsItem 包含多個資訊欄位，包括標題、ID、優先順序、描述、OpsItem 的來源，以及上次更新的日期/時間。每個 OpsItem 還包含以下可設定的功能：

  - Status (狀態)：Open (開放)、In progress (正在進行)、Resolved (已解決)，或 Open and In progress (開放並正在進行)。
  - 相關資源：相關資源是受影響的資源，或起始建立 OpsItem 之 EventBridge 事件的資源。每個 OpsItem 都包含 Related resources (相關資源) 區段，OpsCenter 會在此自動列出相關資源的 Amazon Resource Name (ARN)。您也可以手動指定相關資源的 ARN。針對某些 ARN 類型，OpsCenter 會自動建立深度連結，顯示資源的詳細資訊，不必瀏覽其他主控台頁面即可檢視該資訊。例如，如果您指定 EC2 執行個體的 ARN，您可以在 OpsCenter 中檢視所有 EC2 提供之該執行個體的詳細資訊。您可以手動新增其他相關資源的 ARN。每個 OpsItem 最多可以列出 100 個相關的資源 ARN。如需詳細資訊，請參閱將相關的資源新增至 OpsItem。
  - 相關和類似的 OpsItems：Related OpsItems (相關的 OpsItem) 功能可讓您指定 OpsItems 的 ID，它們與目前的 OpsItem 有某種關聯。Similar OpsItem (類似的 OpsItem) 功能會自動檢閱 OpsItem 標題和描述，然後列出其他相關或您感興趣的 OpsItems。

- Searchable and private operational data (可搜尋的私有營運資料)：營運資料是自訂資料，提供有用的 OpsItem 參考詳細資訊。例如，您可以指定日誌檔案、錯誤字串、授權金鑰、故障診斷秘訣，或其他相關資料。您輸入營運資料做為金鑰/值對。金鑰的最大長度為 128 個字元。值的大小上限為 20 KB。

  這個自訂資料可供搜尋，但有所限制。針對 Searchable operational data (可搜尋操作資料) 功能，所有能存取 OpsItem 概觀頁面的使用者 (如描述 OpsItem API 動作所提供) 都可以檢視和搜尋指定的資料。針對 Private operational data (私有操作資料) 功能，只有能夠存取 OpsItem 的使用者可檢視此資料 (如 GetOpsItem API 操作所提供)。

- 重複資料刪除：透過指定相關資源，OpsCenter 會使用內建的邏輯協助您避免建立重複的 OpsItems。OpsCenter 也包含名為 Operational insights (操作洞察) 的功能，其中顯示有關重複 OpsItems 的資訊。為了進一步限制帳戶中重複 OpsItems 的數量，您可以手動指定 EventBridge 事件規則的重複資料刪除字串。如需詳細資訊，請參閱管理重複的 OpsItems。

- 大量編輯 OpsItems：您可以在 OpsCenter 中選取多個 OpsItems，然後編輯下列其中一個欄位：Status (狀態)、Priority (優先順序)、Severity (嚴重性)、Category (類別)。
- 使用 Runbook 輕鬆修復

  每個 都OpsItem包含 Runbooks 區段，其中包含 Systems Manager Automation Runbook 的清單，可用來自動修復 AWS 資源的常見問題。如果您開啟 OpsItem，請選擇該 AWS 的資源OpsItem，然後選擇主控台中的執行自動化按鈕，然後OpsCenter提供可在產生 AWS 的資源上執行的自動化執行手冊清單OpsItem。從 OpsItem 執行 Automation Runbook 之後，Runbook 會自動與 OpsItem 的相關資源建立關聯，以供日後參考。此外，如果使用 OpsCenter 在 EventBridge 中自動設定 OpsItem 規則，則 EventBridge 會自動關聯常見事件的 Runbook。OpsCenter 會為特定 OpsItem 保留 30 天的 Automation Runbook 執行記錄。如需詳細資訊，請參閱修正 OpsItem 問題。

- 變更通知：您可以指定 Amazon Simple Notification Service (SNS) 主題的 ARN 並在 OpsItem 變更或編輯時隨時發佈通知。SNS 主題必須存在於與 AWS 區域 相同的 中OpsItem。
- Comprehensive search capabilities (全方位的 OpsItem 搜尋功能)：OpsCenter 提供多個搜尋選項，協助您快速找到 OpsItems。以下是幾個搜尋方式範例：OpsItem ID、標題、上次修改時間、營運資料值、來源和 Runbook 執行的 Automation ID，還有更多。您可以使用狀態篩選條件進一步限制搜尋結果。
- OpsItem 摘要報告

  OpsCenter 包含自動顯示下列區段的摘要報告頁面：

  - Status summary (狀態摘要)：依狀態的 OpsItems 摘要 (Open (開放)、In progress (正在進行)、Resolved (已解決)、Open and In progress (開放和正在進行))。
  - 最多開啟的來源OpsItems：最多開啟 AWS 服務的明細OpsItems。

  - by source and age (依來源和天數分組的 OpsItems)：依來源和建立天數分組的 OpsItems 計數。

  如需檢視 OpsCenter 摘要報告的詳細資訊，請參閱 檢視 OpsCenter 摘要報告。

- 記錄和稽核功能支援

  您可以透過與其他 AWS 服務的整合， AWS 帳戶 在 中稽核和記錄OpsCenter使用者動作。如需詳細資訊，請參閱檢視 OpsCenter 日誌和報告。

- 主控台、CLI、PowerShell 和 SDK 對OpsCenter工具的存取

  您可以使用 OpsCenter AWS Systems Manager 主控台、 AWS Command Line Interface (AWS CLI) AWS Tools for PowerShell、 或您選擇的 AWS SDK 來使用 。

OpsCenter 與我現有的案例管理系統整合嗎？

OpsCenter 旨在補充您現有的案例管理系統。您可以使用公有 API 操作，將 OpsItems 整合到您現有的案例管理系統。您也可以在目前的系統中維持手動的生命週期工作流程，使用 OpsCenter 做為調查和修復中樞。

如需 OpsCenter 公有 API 操作的詳細資訊，請參閱《AWS Systems Manager API 參考》中的以下 API 操作。

- CreateOpsItem
- DescribeOpsItems
- GetOpsItem
- GetOpsSummary
- UpdateOpsItem

使用 OpsCenter 需要付費嗎？

是。如需詳細資訊，請參閱 AWS Systems Manager 定價。

OpsCenter 適用於內部部署和混合式受管節點嗎？

是。您可以使用 OpsCenter 調查和修復針對 Systems Manager 設定之內部部署受管節點的問題。如需安裝和設定 Systems Manager 內部部署伺服器和虛擬機器的詳細資訊，請參閱 使用 Systems Manager 在混合多雲端環境中管理節點。

OpsCenter 的配額為何？

您可以在 的 Systems Manager 服務配額中檢視所有 Systems Manager 工具的配額Amazon Web Services 一般參考。除非另有說明，否則每個配額都是區域特定規定。

設定 OpsCenter

AWS Systems Manager 使用整合的設定體驗來協助您開始使用 OpsCenter和 Explorer，這些是 Systems Manager 中的工具。 Explorer 是可自訂的操作儀表板，可報告 AWS 資源的相關資訊。在本文件中，Explorer 和 OpsCenter 設定稱為整合式設定。

您必須使用整合式設定來透過 Explorer 設定 OpsCenter。整合設定僅適用於 AWS Systems Manager 主控台。您無法以程式設計方式設定 Explorer 和 OpsCenter。如需詳細資訊，請參閱開始使用 Systems Manager Explorer 和 OpsCenter。

開始之前

當您設定 OpsCenter 時，您可以在自動建立 OpsItems 的 Amazon EventBridge 中啟用預設規則。下表說明自動建立 OpsItems 的預設 EventBridge 規則。您可以在 OpsItem 規則下的 OpsCenter 設定頁面中停用 EventBridge 規則。

> **Important**
>
> 系統針對預設規則建立的 OpsItems 向您的帳戶收取費用。如需詳細資訊，請參閱AWS Systems Manager 定價。

規則名稱	描述
SSMOpsItems-Autoscaling-instance-launch-failure	此規則會在 EC2 自動擴展執行個體啟動失敗時建立 OpsItems。
SSMOpsItems-Autoscaling-instance-termination-failure	此規則會在 EC2自動擴展執行個體終止失敗時建立 OpsItems。
SSMOpsItems-EBS-snapshot-copy-failed	此規則會在系統複製 Amazon Elastic Block Store (Amazon EBS) 快照失敗時建立 OpsItems。

規則名稱	描述
SSMOpsItems-EBS-snapshot-creation-failed	此規則會在系統建立 Amazon EBS 快照失敗時建立 OpsItems。
SSMOpsItems-EBS-volume-performance-issue	此規則對應至 AWS Health 追蹤規則。該規則會在 Amazon EBS 磁碟區 (運作狀態事件 = `AWS_EBS_DEGRADED_EBS_VOLUME_PERFORMANCE`) 發生效能問題時建立 OpsItems。
SSMOpsItems-EC2-issue	此規則對應到影響 AWS 服務或資源的非預期事件的 AWS Health 追蹤規則。該規則會建立 OpsItems，例如，服務傳送有關造成服務降級之操作問題的通訊，或提高本地化資源層級問題的意識。例如，此規則會為下列事件建立 OpsItem：`AWS_EC2_OPERATIONAL_ISSUE` 。
SSMOpsItems-EC2-scheduled-change	此規則對應至 AWS Health 追蹤規則。 AWS 可以為您的執行個體排程事件，例如重新啟動、停止或啟動執行個體。該規則會為 EC2 已排程事件建立 OpsItems。如需有關已排程事件的詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Scheduled events for your instances。
SSMOpsItems-RDS-issue	此規則對應到影響 AWS 服務或資源的非預期事件的 AWS Health 追蹤規則。該規則會建立 OpsItems，例如，服務傳送有關造成服務降級之操作問題的通訊，或提高本地化資源層級問題的意識。例如，此規則會為下列事件建立 OpsItem：`AWS_RDS_MYSQL_DATABASE_CRASHING_REPEATEDLY` 、`AWS_RDS_EXPORT_TASK_FAILED` 和 `AWS_RDS_CONNECTIVITY_ISSUE` 。

規則名稱	描述
SSMOpsItems-RDS-scheduled-change	此規則對應至 AWS Health 追蹤規則。該規則會為 Amazon RDS 已排程事件建立 OpsItems。已排程事件會提供 Amazon RDS 資源即將發生之變更的相關資訊。部分事件可能會建議您採取行動以避免服務中斷。其他事件會自動發生，您無需採取任何動作。在已排定變更活動期間，您的資源可能暫時無法使用。例如，此規則會為下列事件建立 OpsItem：AWS_RDS_SYSTEM_UPGRADE_SCHEDULED 和 AWS_RDS_MAINTENANCE_SCHEDULED 。如需有關已排程事件的詳細資訊，請參閱《AWS Health 使用者指南》中的事件類型類別。
SSMOpsItems-SSM-maintenance-window-execution-failed	此規則會在處理 Systems Manager 維護視窗失敗時建立 OpsItems。
SSMOpsItems-SSM-maintenance-window-execution-timedout	此規則會在啟動「系統維護」視窗逾時時建立 OpsItems。

使用下列程序來設定 OpsCenter。

設定 OpsCenter

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 OpsCenter。
3. 在 OpsCenter 首頁上，選擇入門。
4. 在OpsCenter設定頁面上，選擇啟用此選項，讓 Explorer設定 AWS Config 和 Amazon CloudWatch 事件OpsItems根據常用的規則和事件自動建立。如果不選擇此選項，OpsCenter 會保持停用。

> **Note**
>
> Amazon EventBridge (前稱為 Amazon CloudWatch Events) 不僅提供 CloudWatch Events 的所有功能，還提供一些新功能，例如自訂事件匯流排、第三方事件來源和結構描述登錄檔。

5. 選擇 啟用OpsCenter 。

啟用 OpsCenter 後，您可以從設定執行以下操作：

- 使用打開 CloudWatch 主控台按鈕建立 CloudWatch 警示。如需詳細資訊，請參閱設定 CloudWatch 警示以建立 OpsItems。
- 啟用操作洞察。如需詳細資訊，請參閱分析操作洞察以減少 OpsItems。
- 啟用 AWS Security Hub 問題清單警示。如需詳細資訊，請參閱了解與 的OpsCenter整合 AWS Security Hub。

目錄

- (選用) 設定 OpsCenter 以跨帳戶集中管理 OpsItems
- (選用) 設定 Amazon SNS 以接收有關 OpsItems 的通知

(選用) 設定 OpsCenter 以跨帳戶集中管理 OpsItems

您可以使用 Systems Manager OpsCenter 集中管理所選 AWS 帳戶 中的OpsItems多個 AWS 區域。此功能可在您在 中設定組織後使用 AWS Organizations。 AWS Organizations 是一種帳戶管理服務，可讓您將多個 AWS 帳戶合併到您建立並集中管理的組織。 AWS Organizations 包含帳戶管理和合併帳單功能，可讓您更充分地滿足業務的預算、安全和合規需求。如需詳細資訊，請參閱AWS Organizations 《 使用者指南》中的什麼是 AWS Organizations？

屬於 AWS Organizations 管理帳戶的使用者可設定 Systems Manager 的委派管理員帳戶。對於 OpsCenter 而言，受委派管理員可以建立、編輯和檢視成員帳戶中的 OpsItems。委派管理員也可以使用 Systems Manager Automation Runbook 大量解決OpsItems或修復產生 AWS 的資源問題 OpsItems。

> **Note**
>
> 您只能將一個帳戶指定為 Systems Manager 的受委派管理員。如需詳細資訊，請參閱為 Systems Manager 建立 AWS Organizations 委派管理員。

Systems Manager 提供下列方法來設定 OpsCenter 以跨多個 AWS 帳戶集中管理 OpsItems。

- Quick Setup：Quick Setup 是 Systems Manager 中的工具，可簡化 Systems Manager 工具的設定和組態任務。如需詳細資訊，請參閱AWS Systems Manager Quick Setup。

  OpsCenter 的快速設定可協助您完成實現跨帳戶管理 OpsItems 所需進行的下列任務：

  - 將一個帳戶註冊為受委派管理員 (如果尚未指定受委派管理員)
  - 建立 required AWS Identity and Access Management (IAM) 政策和角色
  - 指定委派管理員可在帳戶OpsItems間管理 AWS Organizations 的組織或組織單位 (OUs)

  如需詳細資訊，請參閱(選用) 使用 Quick Setup 設定 OpsCenter 以跨帳戶管理 OpsItems。

  > **Note**
  >
  > AWS 區域 在 Systems Manager 目前可用的所有 中，都無法使用快速設定。如果您使用的區域無法使用快速設定對 OpsCenter 進行設定以實現跨多個帳戶集中管理 OpsItems，則您必須使用手動方法。若要檢視可使用 Quick Setup AWS 區域 的清單，請參閱 AWS 區域中的 Quick Setup 可用性。

- 手動設定：如果您使用的區域無法使用快速設定對 OpsCenter 進行設定以實現跨多個帳戶集中管理 OpsItems，則您可以使用手動方法實現。如需詳細資訊，請參閱(選用) 手動設定 OpsCenter 以跨帳戶集中管理 OpsItems。

(選用) 使用 Quick Setup 設定 OpsCenter 以跨帳戶管理 OpsItems

Quick Setup中的工具 AWS Systems Manager可簡化 Systems Manager 工具的設定和組態任務。Quick Setup for OpsCenter可協助您完成下列OpsItems跨帳戶管理任務：

- 指定受委派管理員帳戶
- 建立 required AWS Identity and Access Management (IAM) 政策和角色
- 指定 AWS Organizations 組織或成員帳戶的子集，委派管理員可以OpsItems跨帳戶管理

當您設定 OpsCenter 以使用快速設定跨帳號管理 OpsItems 時，Quick Setup 會在指定的帳戶中建立下列資源。這些資源提供指定帳戶使用 的許可OpsItems，並使用 Automation Runbook 修正產生 AWS 之資源的問題OpsItems。

資源	帳戶
`AWSServiceRoleForAmazonSSM_AccountDiscovery` AWS Identity and Access Management (IAM) 服務連結的角色 如需有關此角色的詳細資訊，請參閱 使用角色來收集 OpsCenter和 AWS 帳戶 的資訊 Explorer。	AWS Organizations 管理帳戶和委派管理員帳戶
`OpsItem-CrossAccountManagementRole` IAM 角色 `AWS-SystemsManager-AutomationAdministrationRole` IAM 角色	委派管理員帳戶
`OpsItem-CrossAccountExecutionRole` IAM 角色 `AWS-SystemsManager-AutomationExecutionRole` IAM 角色 適用於預設 OpsItem 群組 (`OpsItemGroup`) 的 `AWS::SSM::ResourcePolicy` Systems Manager 資源政策	所有 AWS Organizations 成員帳戶

> **Note**
>
> 如果您先前OpsCenter已設定 使用手動方法OpsItems跨帳戶管理 ，則必須刪除在該程序的步驟 4 和 5 期間建立的 AWS CloudFormation 堆疊或堆疊集。當您完成以下程序時，如果這些資源存在於您的帳戶中，則 Quick Setup 無法正確設定跨帳戶 OpsItem 管理。

使用快速設定設定 OpsCenter 以跨帳戶管理 OpsItems

1. AWS Management Console 使用 AWS Organizations 管理帳戶登入 。
2. 開啟 AWS Systems Manager 主控台，網址為 https：/https://console.aws.amazon.com/systems-manager/.。
3. 在導覽窗格中，選擇 Quick Setup。
4. 選擇程式庫索引標籤。
5. 捲動至底部並找到 OpsCenter 組態圖磚。選擇建立。
6. 在 Quick Setup OpsCenter 頁面的受委派管理員區段中，輸入帳戶 ID。如果您無法編輯此欄位，則表示已為 Systems Manager 指定受委派管理員帳戶。
7. 在 Targets (目標) 區段中，選擇一個選項。如果您選擇自訂，請選取您要跨帳戶管理 OpsItems 的組織單位 (OU)。
8. 選擇建立。

Quick Setup 會建立 OpsCenter 組態並將必要的 AWS 資源部署到指定的 OU。

> **Note**
>
> 如果您不想跨多個帳戶管理 OpsItems，可以從 Quick Setup 中刪除相關組態。刪除組態時，Quick Setup 會刪除最初部署組態時建立的下列 IAM 政策和角色：
>
> - 受委派管理員帳戶的 `OpsItem-CrossAccountManagementRole`
> - 來自所有 Organizations 成員帳戶的 `OpsItem-CrossAccountExecutionRole` 和 `SSM::ResourcePolicy`
>
> Quick Setup 會移除所有組織單位以及最初部署組態的 AWS 區域 中的組態。

使用 OpsCenter 的 Quick Setup 組態診斷並解決問題

本節包含的資訊可協助您在使用 Quick Setup 設定跨帳戶 OpsItem 管理時診斷並解決問題。

主題

- 部署到這些堆疊集失敗：delegatedAdmin
- Quick Setup 組態狀態顯示失敗

部署到這些堆疊集失敗：delegatedAdmin

建立 OpsCenter 組態時，Quick Setup 會在 Organizations 管理帳戶中部署兩個 AWS CloudFormation 堆疊集。這些堆疊集使用以下字首：`AWS-QuickSetup-SSMOpsCenter`。如果 Quick Setup 顯示以下錯誤：`Deployment to these StackSets failed: delegatedAdmin` 使用以下程序來修正此問題。

診斷並解決 "StackSets failed:delegatedAdmin" 錯誤

1. 如果您在Quick Setup主控台中收到紅色橫幅中的`Deployment to these StackSets failed: delegatedAdmin`錯誤，請登入委派管理員帳戶，並將 AWS 區域 指定為Quick Setup主要區域。
2. 開啟 AWS CloudFormation 主控台，網址為 https://console.aws.amazon.com/cloudformation：//。
3. 選擇 Quick Setup 組態建立的堆疊。堆疊名稱包含下列項目：AWS-QuickSetup-SSMOpsCenter。

   > **Note**
   >
   > CloudFormation 有時會刪除失敗的堆疊部署。如果堆疊在堆疊資料表中不可用，請從篩選條件清單中選擇已刪除。

4. 檢視 Status (狀態) 和 Status reason (狀態原因)。如需堆疊狀態的詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的堆疊狀態碼。
5. 要想了解失敗的確切步驟，請在事件標籤中檢視每個事件的狀態。如需詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的故障診斷一節。

> **Note**
>
> 如果無法使用 CloudFormation 故障診斷步驟解決部署失敗問題，請刪除組態後再試一次。

Quick Setup 組態狀態顯示失敗

如果組態詳細資訊頁面上的組態詳細資訊資料表顯示 的組態狀態`Failed`，請登入失敗的 AWS 帳戶 和區域。

診斷並解決因 Quick Setup 失敗而無法建立 OpsCenter 組態的問題

1. 登入失敗發生的 AWS 帳戶 和 AWS 區域 。
2. 開啟 AWS CloudFormation 主控台，網址為 https://console.aws.amazon.com/cloudformation：//。
3. 選擇 Quick Setup 組態建立的堆疊。堆疊名稱包含下列項目：AWS-QuickSetup-SSMOpsCenter。

   > **Note**
   >
   > CloudFormation 有時會刪除失敗的堆疊部署。如果堆疊在堆疊資料表中不可用，請從篩選條件清單中選擇已刪除。

4. 檢視 Status (狀態) 和 Status reason (狀態原因)。如需堆疊狀態的詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的堆疊狀態碼。
5. 要想了解失敗的確切步驟，請在事件標籤中檢視每個事件的狀態。如需詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的故障診斷一節。

成員帳戶組態顯示 ResourcePolicyLimitExceededException

如果某個堆疊的狀態顯示為 "`ResourcePolicyLimitExceededException`"，則表示該帳戶先前已使用手動方法加入 OpsCenter 跨帳戶管理。若要解決此問題，您必須刪除在手動加入程序的步驟 4 和 5 期間建立的 AWS CloudFormation 堆疊或堆疊集。如需詳細資訊，請參閱AWS CloudFormation 《 使用者指南》中的刪除堆疊集和刪除 AWS CloudFormation 主控台上的堆疊。

(選用) 手動設定 OpsCenter 以跨帳戶集中管理 OpsItems

本節說明如何手動設定 OpsCenter 以跨帳戶管理 OpsItem。儘管此程序仍然受到支援，但它已被使用 Systems Manager Quick Setup 的新程序所取代。如需詳細資訊，請參閱(選用) 使用 Quick Setup 設定 OpsCenter 以跨帳戶管理 OpsItems。

您可以設定一个中央帳戶，以便為成員帳戶手動建立 OpsItems，管理並修正這些 OpsItems。中央帳戶可以是管理帳戶，也可以是 AWS Organizations AWS Organizations 管理帳戶和 Systems Manager 委派管理員帳戶。我們建議您使用 Systems Manager 受委派的管理員帳戶作為中央帳戶。在設定 AWS Organizations之後才能使用此功能。

使用 AWS Organizations，您可以將多個 合併 AWS 帳戶 到您集中建立和管理的組織。中央帳戶使用者可以為所有選取的成員帳戶同時建立 OpsItems，並管理這些 OpsItems。

使用本節中的程序，在 Organizations 中啟用 Systems Manager 服務主體，並設定 AWS Identity and Access Management (IAM) 許可以OpsItems跨帳戶使用 。

主題

- 開始之前
- 步驟 1：建立資源資料同步
- 步驟 2：在 中啟用 Systems Manager 服務主體 AWS Organizations
- 步驟 3：建立 AWSServiceRoleForAmazonSSM_AccountDiscovery 服務連結角色
- 步驟 4：設定跨帳戶使用 OpsItems 的許可
- 步驟 5：設定跨帳戶使用相關資源的許可

> **Note**
>
> 跨帳戶使用 OpsCenter 時，僅支援 `/aws/issue` 類型的 OpsItems。

開始之前

在設定 OpsItems 來跨帳戶使用 OpsCenter 之前，請確定您已設定下列項目：

- Systems Manager 委派管理員帳戶。如需詳細資訊，請參閱設定 Explorer 的委派管理員。
- 在 Organizations 中設定了一個組織。如需詳細資訊，請參閱 AWS Organizations 使用者指南中的建立和管理組織。
- 您已設定 Systems Manager Automation 跨多個 AWS 區域 和 AWS 帳戶執行自動化 Runbook。如需詳細資訊，請參閱在多個 AWS 區域 和 帳戶中執行自動化。

步驟 1：建立資源資料同步

在您設定和設定之後 AWS Organizations，您可以透過建立資源資料同步OpsCenter，將整個組織彙總OpsItems到 中。如需詳細資訊，請參閱刪除資源資料同步。建立同步時，在新增帳戶區段中，請務必選擇包含我 AWS Organizations 組態中的所有帳戶選項。

步驟 2：在 中啟用 Systems Manager 服務主體 AWS Organizations

若要讓使用者跨OpsItems帳戶使用 ，則必須在 中啟用 Systems Manager 服務主體 AWS Organizations。如果您先前使用其他工具為多帳戶案例設定 Systems Manager，則 Systems Manager 服務主體可能已在 Organizations 中設定。在 AWS Command Line Interface (AWS CLI) 中執行以下命

令以進行驗證。如果您尚未針對其他多帳戶案例設定 Systems Manager，則請跳至下一個程序：啟用 AWS Organizations中的 Systems Manager 服務主體。

若要確認已在 中啟用 Systems Manager 服務主體 AWS Organizations

1. 將最新版本的 下載 AWS CLI 到您的本機機器。
2. 開啟 AWS CLI，並執行下列命令來指定您的登入資料和 AWS 區域。

```
aws configure
```

系統會提示您指定下列項目。在下列範例中，將每個*#########*取代為您自己的資訊。

```
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. 執行以下命令，驗證已為 AWS Organizations啟用 Systems Manager 服務主體。

```
aws organizations list-aws-service-access-for-organization
```

此命令會傳回與以下範例中的內容相似的資訊。

```
{
    "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 中啟用 Systems Manager 服務主體 AWS Organizations

如果您先前沒有為 Organizations 設定 Systems Manager 服務主體，則請使用以下程序進行設定。如需有關此命令的詳細資訊，請參閱《AWS CLI 命令參考》中的 enable-aws-service-access。

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。如需詳細資訊，請參閱安裝 CLI 以及設定 CLI。
2. 將最新版本的 下載 AWS CLI 到您的本機機器。
3. 開啟 AWS CLI，並執行下列命令來指定您的登入資料和 AWS 區域。

   ```
   aws configure
   ```

   系統會提示您指定下列項目。在下列範例中，將每個*########*取代為您自己的資訊。

   ```
   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. 執行以下命令，啟用 AWS Organizations的 Systems Manager 服務主體。

   ```
   aws organizations enable-aws-service-access --service-principal "ssm.amazonaws.com"
   ```

步驟 3：建立 **AWSServiceRoleForAmazonSSM_AccountDiscovery** 服務連結角色

例如 AWSServiceRoleForAmazonSSM_AccountDiscovery角色的服務連結角色是直接連結至 的唯一 IAM 角色類型 AWS 服務，例如 Systems Manager。服務連結角色是由 服務預先定義，並包含服務 AWS 服務 代表您呼叫其他 所需的所有許可。如需 AWSServiceRoleForAmazonSSM_AccountDiscovery 服務連結角色的詳細資訊，請參閱適用於 Systems Manager 帳戶探索的服務連結角色許可。

透過使用 AWS CLI，使用以下程序來建立 AWSServiceRoleForAmazonSSM_AccountDiscovery 服務連結角色。如需有關此程序中所用命令的詳細資訊，請參閱《AWS CLI 命令參考》中的 create-service-linked-role。

建立 **AWSServiceRoleForAmazonSSM_AccountDiscovery** 服務連結角色

1. 登入 AWS Organizations 管理帳戶。
2. 登入 Organizations 管理帳戶時，請執行以下命令。

```
aws iam create-service-linked-role \
    --aws-service-name accountdiscovery.ssm.amazonaws.com \
    --description "Systems Manager account discovery for AWS Organizations service-
linked role"
```

步驟 4：設定跨帳戶使用 OpsItems 的許可

使用 AWS CloudFormation 堆疊集來建立`OpsItemGroup`資源政策和 IAM 執行角色，以授予使用者OpsItems跨帳戶使用 的許可。若要開始使用，請下載並解壓縮 `OpsCenterCrossAccountMembers.zip` 檔案。此檔案包含 `OpsCenterCrossAccountMembers.yaml` AWS CloudFormation 範本檔案。當您使用此範本建立堆疊集時，CloudFormation 會自動在帳戶中建立 `OpsItemCrossAccountResourcePolicy` 資源政策和 `OpsItemCrossAccountExecutionRole` 執行角色。如需有關建立堆疊集的詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的建立堆疊集。

> **Important**
>
> 記下有關此任務的以下重要資訊。
>
> - 您必須在登入 AWS Organizations 管理帳戶時部署堆疊集。
> - 您必須在登入到您希望跨帳戶使用 OpsItems 的每個指定帳戶時重複此過程，包括委派管理員帳戶。
> - 如果您想要在不同的 中啟用跨帳戶OpsItems管理 AWS 區域，請選擇在範本的指定區域區段中新增所有區域。 選擇加入區域不支援跨帳戶 OpsItem 管理。

步驟 5：設定跨帳戶使用相關資源的許可

OpsItem 可包括受影響資源 (例如 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體或 Amazon Simple Storage Service (Amazon S3) 儲存貯體) 的詳細資訊。您在之前步驟 4 中建立的 `OpsItemCrossAccountExecutionRole` 執行角色會為成員帳戶提供 OpsCenter 唯讀許可，以便檢視相關資源。您還必須建立 IAM 角色，以便為管理帳戶提供檢視相關資源並與之互動的許可，您將在此任務中完成。

若要開始使用，請下載並解壓縮 `OpsCenterCrossAccountManagementRole.zip` 檔案。此檔案包含 `OpsCenterCrossAccountManagementRole.yaml` AWS CloudFormation 範本檔案。當您使用此範本建立堆疊時，CloudFormation 會自動在帳戶中建立

`OpsCenterCrossAccountManagementRole` IAM 角色。如需建立堆疊的詳細資訊，請參閱AWS CloudFormation 《使用者指南》中的在 AWS CloudFormation 主控台上建立堆疊。

> **Important**
>
> 記下有關此任務的以下重要資訊。
>
> - 如果您打算將帳戶指定為 的委派管理員OpsCenter，請務必在建立堆疊 AWS 帳戶 時指定 。
> - 您必須在登入 AWS Organizations 管理帳戶時執行此程序，並在登入委派管理員帳戶時再次執行此程序。

(選用) 設定 Amazon SNS 以接收有關 OpsItems 的通知

您可以設定 OpsCenter，以在系統建立 OpsItem 或更新現有 OpsItem 時，將通知傳送至 Amazon Simple Notification Service (Amazon SNS) 主題。

完成下列步驟以接收 OpsItems 的通知。

- 步驟 1：建立並訂閱 Amazon SNS 主題
- 步驟 2：更新 Amazon SNS 存取政策
- 步驟 3：更新 AWS KMS 存取政策

> **Note**
>
> 如果您在步驟 2 中開啟 AWS Key Management Service (AWS KMS) 伺服器端加密，則必須完成步驟 3。否則，可以略過步驟 3。

- 步驟 4：開啟預設 OpsItems 規則以傳送有關新 OpsItems 的通知

步驟 1：建立並訂閱 Amazon SNS 主題

若要接收通知，您必須建立並訂閱 Amazon SNS 主題。如需詳細資訊，請參閱《Amazon Simple Notification Service 開發人員指南》中的建立 Amazon SNS 主題和訂閱 Amazon SNS 主題。

> **Note**
>
> 如果您在OpsCenter多個 AWS 區域 或 帳戶中使用 ，則必須在您想要接收OpsItem通知的每個區域或帳戶中建立並訂閱 Amazon SNS 主題。

步驟 2：更新 Amazon SNS 存取政策

您必須將 Amazon SNS 主題與 OpsItems 相關聯。使用以下程序設定 Amazon SNS 存取政策，讓 Systems Manager 可以將 OpsItems 通知發佈至您在步驟 1 中建立的 Amazon SNS 主題。

1. 登入 AWS Management Console ，並在 https://console.aws.amazon.com/sns/v3/home：// 開啟 Amazon SNS 主控台。
2. 在導覽窗格中，選擇主題。
3. 選擇在步驟 1 中建立的主題，然後選擇編輯。
4. 展開 Access policy (存取政策)。
5. 將以下 `Sid` 區塊新增至現有的政策。將每個*########*取代為您自己的資訊。

```
{
      "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**
>
> `aws:SourceAccount` 全域條件金鑰可防止混淆代理人情境。若要使用此條件金鑰，請將值設定為相應 OpsItem 擁有者的帳戶 ID。如需詳細資訊，請參閱《IAM 使用者指南》中的混淆代理一節。

6. 選擇儲存變更。

現在，建立或更新 OpsItems 時，系統會傳送通知給 Amazon SNS 主題。

> **Important**
>
> 如果您在步驟 2 中使用 AWS Key Management Service (AWS KMS) 伺服器端加密金鑰設定 Amazon SNS 主題，請完成步驟 3。否則，可以略過步驟 3。

步驟 3：更新 AWS KMS 存取政策

如果您開啟 Amazon SNS 主題的 AWS KMS 伺服器端加密，您還必須更新 AWS KMS key 您在設定主題時選擇之 的存取政策。使用下列程序更新存取政策，讓 Systems Manager 可以將 OpsItem 通知發佈至您在步驟 1 中建立的 Amazon SNS 主題。

> **Note**
>
> OpsCenter 不支援將 OpsItems 發佈至以 AWS 受管金鑰設定的 Amazon SNS 主題。

1. 在 https：//https://console.aws.amazon.com/kms 開啟 AWS KMS 主控台。
2. 若要變更 AWS 區域，請使用頁面右上角的區域選擇器。
3. 在導覽窗格中，選擇 Customer managed keys (客戶受管金鑰)。
4. 選擇您在建立主題時所選擇 KMS 金鑰的 ID。
5. 在 Key policy (金鑰政策) 區段中，選擇 Switch to policy view (切換至政策檢視)。
6. 選擇編輯。
7. 將以下 `Sid` 區塊新增至現有的政策。將每個`########`取代為您自己的資訊。

```
{
```

```
    "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"
  }
```

在以下範例中，第 14 行輸入新區塊。

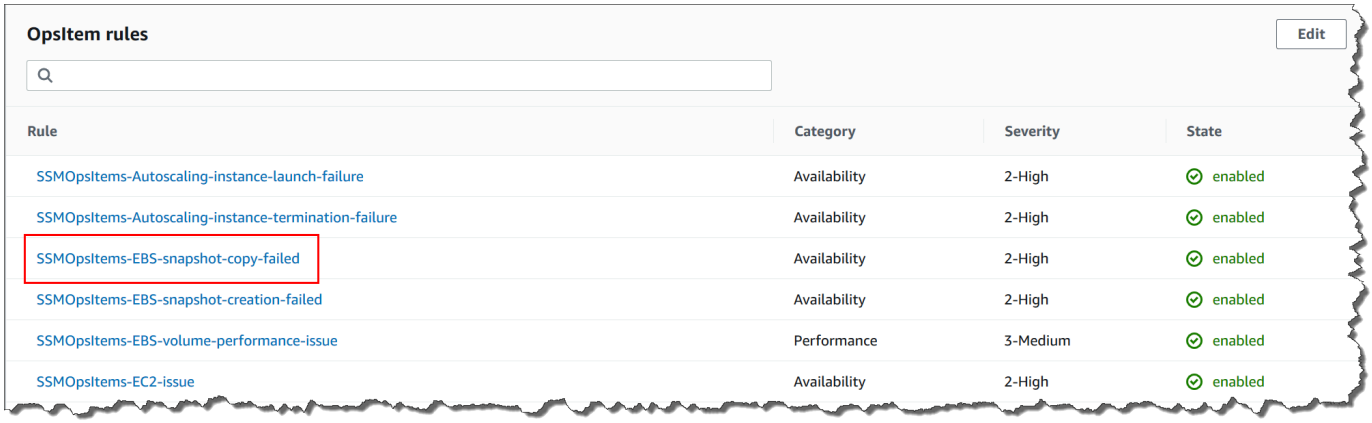
8. 選擇 Save changes (儲存變更)。

步驟 4：開啟預設 OpsItems 規則以傳送有關新 OpsItems 的通知

Amazon EventBridge 中的預設 OpsItems 規則未設定 Amazon Resource Name (ARN) 來傳送 Amazon SNS 通知。使用下列程序在 EventBridge 中編輯規則並輸入 `notifications` 區塊。

將通知區塊新增至預設 OpsItem 規則

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 OpsCenter。
3. 選擇 OpsItems 索引標籤，然後選擇 Configure sources (設定來源)。
4. 選擇您要以 `notifications` 區塊設定的來源規則名稱，如下列範例所示。



規則在 Amazon EventBridge 中開啟。

5. 在規則詳細資訊頁面的 Targets (目標) 索引標籤上，選擇 Edit (編輯)。
6. 在 Additional settings (其他設定) 區段中，選擇 Configure input transformer (設定輸入轉換器)。
7. 在範本方塊中，以下列格式新增 `notifications` 區塊。

```
"notifications":[{"arn":"arn:aws:sns:region:account ID:topic name"}],
```

範例如下。

```
"notifications":[{"arn":"arn:aws:sns:us-west-2:1234567890:MySNSTopic"}],
```

輸入 `resources` 區塊前的通知區塊，如下列美國西部 (奧勒岡) (us-west-2) 區域範例所示。

```
{
    "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. 選擇確認。
9. 選擇下一步。
10. 選擇下一步。
11. 選擇更新規則。

系統下次為預設規則建立 OpsItem 時，就會將通知發佈至 Amazon SNS 主題。

將 OpsCenter 與其他 AWS 服務整合

OpsCenter是 中的工具 AWS Systems Manager，可與多個 整合， AWS 服務 以診斷和修復 AWS 資源的問題。您必須先設定 ， AWS 服務 才能將其與 整合OpsCenter。

根據預設，下列項目 AWS 服務 會與 整合OpsCenter，並且可以OpsItems自動建立：

- Amazon CloudWatch
- Amazon CloudWatch Application Insights
- Amazon EventBridge
- AWS Config
- AWS Systems Manager Incident Manager

您必須將下列服務與 OpsCenter 整合，才能自動建立 OpsItems：

- Amazon DevOps Guru
- AWS Security Hub

這些服務中的任何一項建立 OpsItem 後，您可以在 OpsCenter 中管理並修復 OpsItem。如需詳細資訊，請參閱管理 OpsItems及修正 OpsItem 問題。

如需每個 AWS 服務 及其如何與 整合的詳細資訊OpsCenter，請參閱下列主題。

主題

- 了解 OpsCenter 與 Amazon CloudWatch 的整合
- 了解 OpsCenter 與 Amazon CloudWatch Application Insights 的整合
- 了解 OpsCenter 與 Amazon DevOps Guru 的整合
- 了解 OpsCenter 與 Amazon EventBridge 的整合
- 了解與 的OpsCenter整合 AWS Config
- 了解與 的OpsCenter整合 AWS Security Hub
- 了解 OpsCenter 與 Incident Manager 的整合

了解 OpsCenter 與 Amazon CloudWatch 的整合

Amazon CloudWatch 會監控您的 AWS 資源和服務，並在 AWS 服務 您使用的每個 上顯示指標。警示進入警示狀態時，CloudWatch 會建立 OpsItem。例如，如果 Application Load Balancer 產生的 HTTP 錯誤突然增加，則您可以設定警示來自動建立 OpsItem。

以下清單顯示了您可在 CloudWatch 中設定以建立 OpsItems 的一些警示：

- Amazon DynamoDB：資料庫讀取和寫入動作達到閾值
- Amazon EC2：CPU 使用率達到閾值
- AWS 帳單：預估費用達到閾值
- Amazon EC2：執行個體未通過狀態檢查
- Amazon Elastic Block Store (EBS)：磁碟空間使用率達到閾值

您可以建立警示，也可以編輯現有警示來建立 OpsItem。如需詳細資訊，請參閱設定 CloudWatch 警示以建立 OpsItems。

使用整合式設定啟用 OpsCenter 後，它會將 CloudWatch 與 OpsCenter 整合。

了解 OpsCenter 與 Amazon CloudWatch Application Insights 的整合

使用 Amazon CloudWatch Application Insights，您可以為您應用程式資源設定最佳監控器，以持續分析應用程式問題跡象資料。在 CloudWatch Application Insights 中設定應用程式資源時，可以選擇讓系統在 OpsCenter 中建立 OpsItems。針對應用程式偵測到的每個問題，會在 OpsCenter 主控台上建立一個 OpsItem。如需相關資訊，請參閱《Amazon CloudWatch 使用者指南》中的設定、配置和管理應用程式以進行監控。

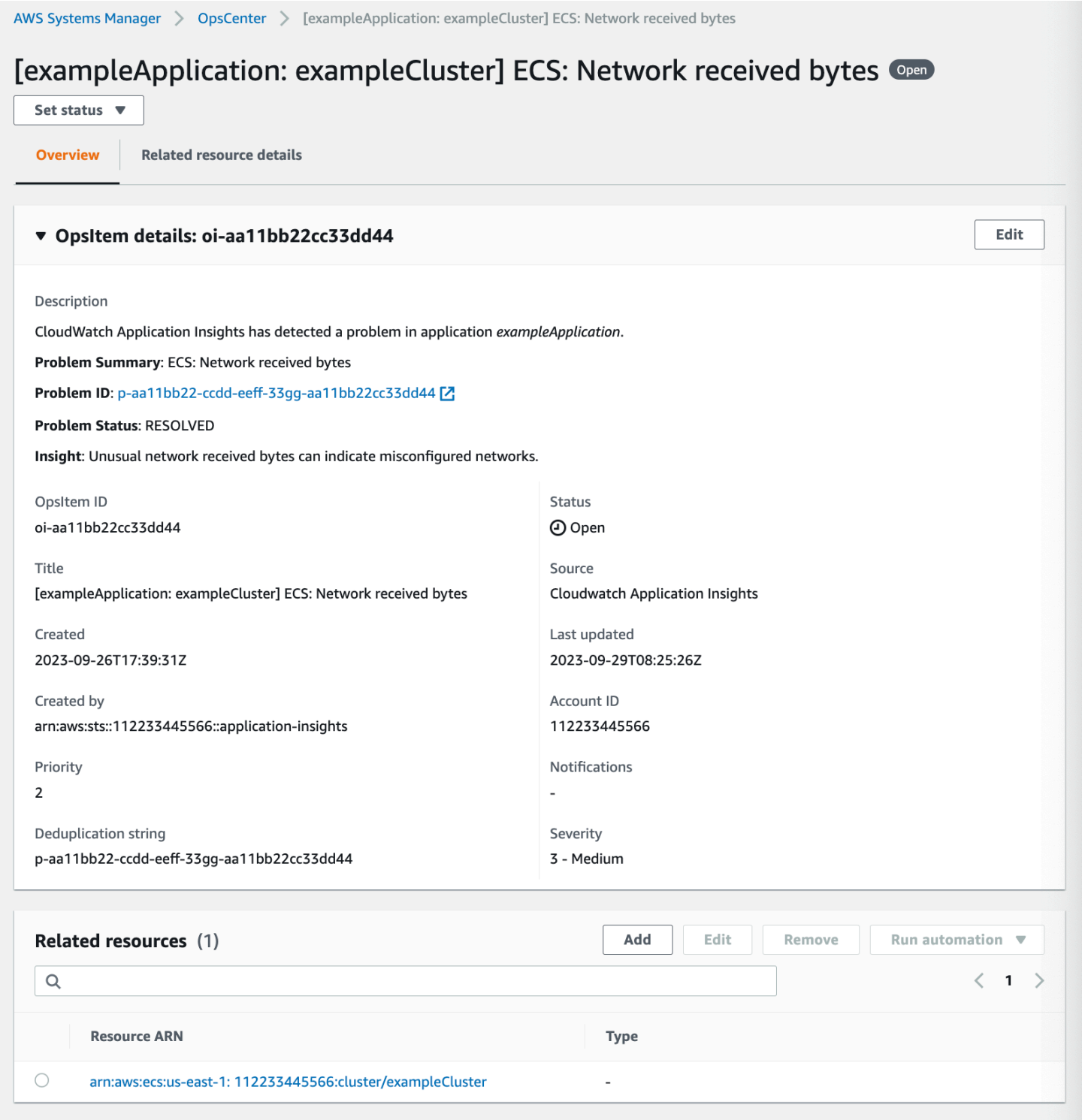
> **Note**
>
> 從 2023 年 10 月 16 日起，CloudWatch Application Insights 所建立之 OpsItems 的標題和描述現在會使用下述改良格式：

```
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>
```

請見此處範例：



了解 OpsCenter 與 Amazon DevOps Guru 的整合

Amazon DevOps Guru 採用機器學習來分析操作資料、應用程式指標和應用程式事件，以識別偏離正常操作模式的行為。如果啟用 DevOps Guru 以在 OpsCenter 中產生 OpsItem，則每個洞見都會產生一個新的 OpsItem。您可以使用 OpsCenter 來管理您的 OpsItems。

DevOps Guru 會自動建立 OpsItems。您可以使用 來啟用 Amazon DevOpsGuru 以建立 OpsItems Quick Setup，這是 Systems Manager 中的工具。系統會OpsItems使用 AWSServiceRoleForDevOpsGuru AWS Identity and Access Management (IAM) 服務連結角色建立 。

將 OpsCenter 與 DevOps Guru 整合

1. 開啟 AWS Systems Manager 主控台，網址為 https://console.aws.amazon.com/systems-manager/：//。
2. 在導覽窗格中，選擇 Quick Setup。
3. 在自訂 DevOps Guru 組態選項頁面中，選擇程式庫索引標籤。
4. 在 DevOps Guru 窗格中，選擇建立。
5. 針對組態選項，選取啟用 AWS Systems Manager OpsItems。
6. 完成設定後，選取建立。

了解 OpsCenter 與 Amazon EventBridge 的整合

Amazon EventBridge 提供描述 AWS 資源變更的事件串流。使用整合式設定啟用 OpsCenter 後，它會將 EventBridge 與 OpsCenter 整合，並啟用預設的 EventBridge 規則。EventBridge 會根據這些規則建立 OpsItems。您可以使用規則篩選事件並將事件路由至 OpsCenter，以進行調查和修復。

> **Note**
>
> Amazon EventBridge (前稱為 Amazon CloudWatch Events) 不僅提供 CloudWatch Events 的所有功能，還提供一些新功能，例如自訂事件匯流排、第三方事件來源和結構描述登錄檔。

以下是一些您可以在 EventBridge 中設定以建立 OpsItem 的規則：

- Security Hub：已發出安全提醒
- Amazon DynamoDB 限流事件
- Amazon Elastic Compute Cloud Auto Scaling：無法啟動執行個體
- Systems Manager：無法執行自動化
- AWS Health：排定維護的提醒
- Amazon EC2：執行個體狀態從「執行中」變更為「停止」

您可以根據需求建立規則或編輯現有規則來建立 OpsItems。如需有關如何編輯規則以建立 OpsItem 的詳細資訊，請參閱設定 EventBridge 規則以建立 OpsItems。

了解與 的OpsCenter整合 AWS Config

AWS Config 提供 中 AWS 資源組態的詳細檢視 AWS 帳戶。

AWS Config 不會直接與 整合OpsCenter。反之，您可以建立將事件傳送至 Amazon EventBridge 的 AWS Config 規則，例如當 AWS Config 偵測到不合規的執行個體時。然後，EventBridge 會根據您建立的 EventBridge 規則評估該事件。如果事件與規則相符，則 EventBridge 會將事件轉換為 OpsItem，並將其傳送至目的地目標 OpsCenter。

使用此 OpsItem，您可以追蹤不合規資源的詳細資訊、記錄調查動作，並提供對一致修復動作的存取權。

相關資訊

設定 EventBridge 規則以建立 OpsItems

使用 AWS Systems ManagerOpsCenter和 AWS Config 進行合規監控

了解與 的OpsCenter整合 AWS Security Hub

AWS Security Hub 從 AWS 帳戶 和 服務收集稱為調查結果的安全資料。Security Hub 使用一組規則來偵測並產生調查結果，可協助您識別您所管理資源的安全性問題、排定其優先順序並進行修補。設定整合之後，如本主題所述，Systems Manager 會在 OpsCenter 中針對 Security Hub 的調查結果建立 OpsItems。

> **Note**
>
> OpsCenter 具有與 Security Hub 的雙向整合。這表示如果您更新與安全性調查結果相關的 OpsItem 的狀態或嚴重性欄位，系統會將變更同步到 Security Hub。同樣的，對調查結果所做的任何變更都會自動在 OpsCenter 中的對應 OpsItems 中得到更新。
> 從 Security Hub 調查結果建立 OpsItem 時，Security Hub 中繼資料會自動新增至 OpsItem 的操作資料欄位。如果刪除這些中繼資料，雙向更新將無法再運作。

根據預設，Systems Manager 可為「關鍵」和「高」安全性調查結果建立 OpsItems。您可以手動設定 OpsCenter 為中低嚴重性調查結果建立 OpsItems。OpsCenter 不會為資訊性調查結果建立 OpsItems，因為此類調查結果不要求採取修補措施。如需有關 Security Hub 嚴重性等級的詳細資訊，請參閱 AWS Security Hub API 參考中的安全性一節。

開始之前

在您設定 OpsCenter 以根據 Security Hub 調查結果建立 OpsItems 之前，請確認您已完成 Security Hub 設定步驟。如需詳細資訊，請參閱《AWS Security Hub 使用者指南》中的建立 Security Hub。

將 Security Hub 與 OpsCenter 整合後，系統會使用 `AWSServiceRoleForSystemsManagerOpsDataSync` IAM 服務連結角色建立 OpsItems。如需有關此角色的詳細資訊，請參閱 使用角色為 OpsItems 建立 OpsData 和 Explorer： 。

> **Warning**
>
> 請注意以下有關 OpsCenter 與 Security Hub 整合的定價的重要資訊：
>
> - 如果您在設定 OpsCenter 與 Security Hub 整合時使用 Security Hub 管理員帳戶登入，則系統會針對管理員和所有成員帳戶中的調查結果建立 OpsItems。所有 OpsItems 都會在管理員帳戶中建立。根據各種因素，這可能會導致來自 的意外大額帳單 AWS。
>
> 如果您在設定整合時使用成員帳戶登入，則系統只會針對該個別帳戶中的調查結果建立 OpsItems。如需有關 Security Hub 管理員帳戶、成員帳戶及其與調查結果 EventBridge 事件摘要之關係的詳細資訊，請參閱《AWS Security Hub 使用者指南》中的 Security Hub 與 EventBridge 整合的類型一節。
> - 針對每個建立 OpsItem 的調查結果，系統會按照標準價格向您收取建立 OpsItem 的費用。如果您編輯 OpsItem 或對應的調查結果已在 Security Hub 中更新 (會觸發 OpsItem 更新)，系統也會向您收取費用。
> - OpsItems 與 整合所建立的 目前 AWS Security Hub 不受區域中OpsItems每個帳戶 500，000 個配額上限的限制。因此，Security Hub 提醒可以在帳戶中OpsItems的每個區域中建立超過 500，000 個收費項目。
>
> 因此，對於高生產環境，我們建議將 Security Hub 問題清單的範圍限制為僅高嚴重性的問題。

設定 OpsCenter 以針對 Security Hub 調查結果建立 OpsItems

1. 開啟 AWS Systems Manager 主控台，網址為 https://console.aws.amazon.com/systems-manager/：//。
2. 在導覽窗格中，選擇 OpsCenter。
3. 選擇設定。

4. 在 Security Hub 調查結果區段中，選擇編輯。
5. 選擇滑桿以將已停用變更為已啟用。
6. 如果您希望系統針對中度或低嚴重性調查結果建立 OpsItems，請切換這些選項。
7. 選擇 Save (儲存) 以儲存您的組態。

如果您不再希望系統針對 Security Hub 調查結果建立 OpsItems，請使用以下程序。

停止接收針對 Security Hub 調查結果的 OpsItems

1. 開啟 AWS Systems Manager 主控台，網址為 https://console.aws.amazon.com/systems-manager/：//。
2. 在導覽窗格中，選擇 OpsCenter。
3. 選擇設定。
4. 在 Security Hub 調查結果區段中，選擇編輯。
5. 選擇滑桿以將已啟用變更為已停用。如果您無法切換滑桿，表示 Security Hub 尚未為您的 AWS 帳戶啟用。
6. 選擇儲存以儲存您的組態。OpsCenter 不再根據 Security Hub 調查結果建立 OpsItems。

> **Important**
>
> Systems Manager 委派管理員或 AWS Organizations 管理帳戶可以在 中OpsCenter為多個帳戶啟用 Security Hub 調查結果， AWS 區域 並在 中建立資源資料同步Explorer。如果在 Explorer 中啟用了 Security Hub 來源，且存在以您停用 Security Hub 整合的成員帳戶為目標的資源資料同步，則管理員選取的設定優先。OpsCenter 會繼續為 Security Hub 調查結果建立 OpsItems。若要停止在資源資料同步鎖定的成員帳戶中為 Security Hub 調查結果建立 OpsItems，請聯絡您的管理員，並要求其從資源資料同步中移除您的帳戶，或在 Explorer 中關閉 Security Hub 來源。如需有關變更 Explorer 中設定的詳細資訊，請參閱編輯 Systems Manager Explorer 資料來源。

了解 OpsCenter 與 Incident Manager 的整合

Incident Manager 是 中的工具 AWS Systems Manager，提供事件管理主控台，協助您減輕影響 AWS 託管應用程式的事件，並從中復原。事件是指任何意外中斷或服務質量下降。設定和配置 Incident Manager 後，系統會自動在 OpsCenter 中建立 OpsItems。

系統在 Incident Manager 中建立事件時，它也會在 OpsCenter 中建立 OpsItem，並將事件顯示為相關項目。如果 OpsItem 已存在，則 Incident Manager 不會建立 OpsItem。第一個 OpsItem 稱為父系 OpsItem。如果事件的規模和範圍成長，則您可以新增事件至現有 OpsItem。如有必要，您可以為 OpsItem 手動建立事件。事件關閉後，您可以在 Incident Manager 中建立分析，以檢閱並改進類似問題的修復程序。

依預設，OpsCenter 會與 Incident Manager 整合。如果未設定 Incident Manager，OpsCenter 頁面會顯示設定 Incident Manager 的訊息。Incident Manager 建立 OpsItem 後，您可以在 OpsCenter 中管理並修復 OpsItem。如需有關為 OpsItem 建立事件的指示，請參閱為 OpsItem 建立事件。

建立 OpsItems

在您設定 OpsCenter、 中的工具 AWS Systems Manager，並將其與您的 整合之後 AWS 服務，您的 OpsItems會根據預設規則、事件或警示 AWS 服務 自動建立 。

您可以檢視預設 Amazon EventBridge 規則的狀態和嚴重性等級。如果需要，您可以在 Amazon EventBridge 中建立或編輯這些規則。您也可以在 Amazon CloudWatch 中檢視警示，並建立或編輯警示。您可以使用規則和警示來設定事件，以便為事件自動產生 OpsItems。

系統建立 OpsItem 後，它會處於待處理狀態。開始調查 OpsItem 後，您可以將狀態變更為進行中，在修復 OpsItem 之後可以將狀態變更為已解決。如需如何在 中設定警示和規則 AWS 服務 以建立 OpsItems以及如何手動建立 OpsItems 的詳細資訊，請參閱下列主題。

主題

- 設定 EventBridge 規則以建立 OpsItems
- 設定 CloudWatch 警示以建立 OpsItems
- 手動建立 OpsItems

設定 EventBridge 規則以建立 OpsItems

Amazon EventBridge 收到事件後，它會根據預設規則建立新的 OpsItem。您可以建立規則或編輯現有的規則，以將 OpsCenter 設定為 EventBridge 事件的目標。如需有關如何建立事件規則的詳細資訊，請參閱《Amazon EventBridge 使用者指南》中的為 AWS 服務建立規則。

設定 EventBridge 規則以在 OpsCenter 中建立 OpsItems

1. 前往 https://console.aws.amazon.com/events/ 開啟 Amazon EventBridge 主控台。
2. 在導覽窗格中，選擇規則。

3. 在 Rules (規則) 頁面，針對 Event bus (事件匯流排)，選擇 default (預設值)。
4. 在規則中，選取名稱旁的核取方塊，以選擇規則。
5. 選取規則的名稱，開啟其詳細資訊頁面。在規則詳細資訊中，確認狀態已設定為已啟用。

> **Note**
>
> 如有必要，您可以使用頁面右上角的編輯來更新狀態。

6. 選擇 Targets (目標) 標籤。
7. 在 Targets (目標) 索引標籤上，選擇 Edit (編輯)。
8. 對於目標類型，請選取 AWS 服務。
9. 針對 Select a target (選取目標)，請選擇 Systems Manager OpsItem。
10. 對於許多目標類型，EventBridge 需要許可才能將事件傳送到目標。在這些情況下，EventBridge 可以建立規則執行所需的 AWS Identity and Access Management (IAM) 角色：
    - 若要自動建立 IAM 角色，請選擇為此特定資源建立新角色。
    - 若要使用您建立的 IAM 角色授予 EventBridge 許可，以在 OpsCenter 中建立 OpsItems，請選擇 Use existing role (使用現有的角色)。
11. 在其他設定的設定目標輸入中，選擇輸入轉換器。

    您可以使用輸入轉換器選項，指定重複資料刪除字串和 OpsItems 的其他重要資訊，例如標題和嚴重性。
12. 選擇設定輸入轉換器。
13. 在目標輸入轉換器的輸入路徑中，指定針對觸發事件要剖析的值。例如，若要透過觸發規則的事件來剖析開始時間、結束時間和其他詳細資訊，請使用下列 JSON。

```
{
    "end-time": "$.detail.EndTime",
    "failure-cause": "$.detail.cause",
    "resources": "$.resources[0]",
    "source": "$.detail.source",
    "start-time": "$.detail.StartTime"
}
```

14. 針對 Template (範本)，指定要傳送至目標的資訊。例如，使用下列 JSON 將資訊傳遞至 OpsCenter。該資訊用於建立 OpsItem。

> **Note**
>
> 如果輸入範本為 JSON 格式，則範本中的物件值不能包含引號。例如，資源、失敗原因、來源、開始時間和結束時間的值不能加引號。

```
{
    "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>
        }
    }
}
```

如需這些欄位的詳細資訊，請參閱《Amazon EventBridge 使用者指南》中的轉換目標輸入。

15. 選擇確認。
16. 選擇下一步。
17. 選擇下一步。
18. 選擇更新規則。

從事件建立 OpsItem 之後，您就可以開啟 OpsItem 並向下捲動至 Private operational data (私有營運資料) 區段，檢視事件詳細資訊。如需如何在 OpsItem 中設定選項的資訊，請參閱 管理 OpsItems。

設定 CloudWatch 警示以建立 OpsItems

在 中OpsCenter工具 的整合設定期間 AWS Systems Manager，您可以讓 Amazon CloudWatch OpsItems 根據常見警示自動建立 。您可以建立警示，也可以編輯現有警示來在 OpsCenter 中建立 OpsItems。

當您設定警示以建立 時，CloudWatch 會在 AWS Identity and Access Management (IAM) 中建立新的服務連結角色OpsItems。新角色已命名為 `AWSServiceRoleForCloudWatchAlarms_ActionSSM`。如需有關 CloudWatch 服務連結角色的詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的使用 CloudWatch 的服務連結角色。

CloudWatch 警示產生 OpsItem 後，OpsItem 會顯示 CloudWatch 警示 - '***alarm_name***' 處於 ALARM 狀態。

若要檢視特定 OpsItem 的詳細資訊，選擇 OpsItem，然後選擇相關資源詳細資訊索引標籤。您可以手動編輯 OpsItems 來變更詳細資訊，例如嚴重性或類別。不過，當您編輯警示的嚴重性或類別時，Systems Manager 無法更新已從警示建立之 OpsItems 的嚴重性或類別。如果警報建立了 OpsItem 而且如果您指定了重複資料刪除字串，則警示不會建立額外的 OpsItems，即使您在 CloudWatch 中編輯警示也是如此。如果 OpsItem 已在 OpsCenter 中解決，CloudWatch 將建立新的 OpsItem。

如需有關設定 CloudWatch 警示的詳細資訊，請參閱下列主題。

主題

- 設定 CloudWatch 警示以建立 OpsItems (主控台)
- 設定現有的 CloudWatch 警示以建立 OpsItems (以程式設計方式)

設定 CloudWatch 警示以建立 OpsItems (主控台)

您可以手動建立警示或更新現有警示，以便在 Amazon CloudWatch 中建立 OpsItems。

建立 CloudWatch 警示並將 Systems Manager 設定為該警示的目標

1. 完成《Amazon CloudWatch 使用者指南》內根據靜態閾值建立 CloudWatch 警示中指定的步驟 1–9。
2. 在 Systems Manager 動作區段中，選擇新增 Systems Manager OpsCenter 動作。
3. 選擇 OpsItems。
4. 在嚴重性中，選擇 1 到 4。
5. (選用) 在類別中，選擇 OpsItem 的類別。
6. 完成《Amazon CloudWatch 使用者指南》內根據靜態閾值建立 CloudWatch 警示中指定的步驟 11–13。
7. 選擇 Next (下一步) 並完成精靈。

編輯現有的警示，並將 Systems Manager 設定為該警示的目標

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇 Alarms (警示)。
3. 選取警示，然後選擇 Actions (動作)、Edit (編輯)。
4. (選用) 變更 Metrics (指標) 和 Conditions (條件) 選項中的設定，然後選擇 Next (下一步)。
5. 在 Systems Manager (Systems Manager) 區段中，選擇 Add Systems Manager OpsCenter action (新增 Systems Manager OpsCenter 動作)。
6. 對於 Severity (嚴重性)，選擇數字。

> **Note**
>
> 嚴重性是使用者定義的值。您或您的組織會決定每個嚴重性值的含義，以及與每個嚴重性相關的任何服務水準協議。

7. (選用) 對於 Category (類別)，選擇一個選項。
8. 選擇 Next (下一步) 並完成精靈。

設定現有的 CloudWatch 警示以建立 OpsItems (以程式設計方式)

您可以使用 AWS Command Line Interface (AWS CLI)、 AWS CloudFormation 範本或Java程式碼片段，將 Amazon CloudWatch 警示設定為OpsItems以程式設計方式建立。

主題

- 開始之前
- 設定 CloudWatch 警示以建立 OpsItems (AWS CLI)
- 設定 CloudWatch 警示以建立或更新 OpsItems (CloudFormation)
- 設定 CloudWatch 警示以建立或更新 OpsItems (Java)

開始之前

如果以程式設計方式編輯現有的警示，或建立警示 (該警示會建立 OpsItems)，則您必須指定 Amazon Resource Name (ARN)。此 ARN 可識別 Systems Manager OpsCenter 作為從警示建立之 OpsItems 的目標。您可以自訂 ARN，以便從警示建立之 OpsItems 包含特定資訊，例如嚴重性或類別。每個 ARN 包含下表中所述的資訊。

參數	詳細資訊
`Region` (必要)	AWS 區域 警示所在的 。例如：`us-west-2` 。如需您可在其中使用 OpsCenter 之 AWS 區域相關資訊，請參閱 AWS Systems Manager 端點和配額。
`account_ID` (必要)	用來建立警示的相同 AWS 帳戶 ID。例如：`123456789012` 。帳戶 ID 後面必須加上冒號 (`:`) 和參數 `opsitem`，如下列範例所示。
`severity` (必要)	從警示建立之 OpsItems 的使用者定義嚴重性層級。有效值：`1`、`2`、`3`、`4`
`Category` (選用)	從警示建立之 OpsItems 的類別 有效值：`Availability` 、`Cost`、`Performance` 、`Recovery` 和 `Security`。

使用下列語法建立 ARN。此 ARN 不包含選用的 `Category` 參數。

```
arn:aws:ssm:Region:account_ID:opsitem:severity
```

以下是範例。

```
arn:aws:ssm:us-west-2:123456789012:opsitem:3
```

若要建立使用選用 Category 參數的 ARN，請使用下列語法。

```
arn:aws:ssm:Region:account_ID:opsitem:severity#CATEGORY=category_name
```

以下是範例。

```
arn:aws:ssm:us-west-2:123456789012:opsitem:3#CATEGORY=Security
```

設定 CloudWatch 警示以建立 OpsItems (AWS CLI)

此命令要求您為 alarm-actions 參數指定 ARN。如需如何建立 ARN 的相關資訊，請參閱 開始之前。

設定 CloudWatch 警示以建立 OpsItems (AWS CLI)

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。
2. 執行下列命令以收集您要設定之警示的相關資訊。

   ```
   aws cloudwatch describe-alarms --alarm-names "alarm name"
   ```
3. 執行下列命令以更新警示。將每個########取代為您自己的資訊。

   ```
   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
   ```

範例如下。

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
```

設定 CloudWatch 警示以建立或更新 OpsItems (CloudFormation)

本節包含可用來設定 CloudWatch 警示以自動建立或更新 的 AWS CloudFormation 範本OpsItems。每個範本都要求您為 `AlarmActions` 參數指定 ARN。如需如何建立 ARN 的相關資訊，請參閱 開始之前。

指標警示 – 使用下列 CloudFormation 範本來建立或更新 CloudWatch 指標警示。此範本中指定的警示會監控 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體狀態檢查。如果警示進入 ALARM 狀態，它會在 OpsCenter 中建立 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"}}]
          }
        }
      }
    }
```

複合警示 – 使用下列 CloudFormation 範本來建立或更新複合警示。複合警示由多個指標警示組成。如果警示進入 ALARM 狀態，它會在 OpsCenter 中建立 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"
         }
      }
```

```
    }
```

設定 CloudWatch 警示以建立或更新 OpsItems (Java)

本節包含 Java 程式碼片段，您可以用這些程式碼片段來設定 CloudWatch 警示以自動建立或更新 OpsItems。每個程式碼片段都要求您為 `validSsmActionStr` 參數指定 ARN。如需如何建立 ARN 的相關資訊，請參閱 開始之前。

特定警示 – 使用以下 Java 程式碼片段來建立或更新 CloudWatch 警示。此範本中指定的警示會監控 Amazon EC2 執行個體狀態檢查。如果警示進入 ALARM 狀態，它會在 OpsCenter 中建立 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);
    }
```

更新所有警示 – 使用以下 Java 程式碼片段來更新 AWS 帳戶 中的所有 CloudWatch 警示，以在警示進入 ALARM 狀態時建立 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;
            }
        }
    }
```

手動建立 OpsItems

當您發現操作問題時，您可以從 OpsItem 中的OpsCenter工具手動建立 AWS Systems Manager，以管理和解決問題。

如果您設定 OpsCenter進行跨帳戶管理，Systems Manager 委派管理員或 AWS Organizations 管理帳戶可以OpsItems為成員帳戶建立 。如需詳細資訊，請參閱(選用) 手動設定 OpsCenter 以跨帳戶集中管理 OpsItems。

您可以使用 AWS Systems Manager 主控台、 AWS Command Line Interface (AWS CLI) 或 OpsItems 來建立 AWS Tools for Windows PowerShell。

主題

- 手動建立 OpsItems (主控台)
- 手動建立 OpsItems (AWS CLI)
- 手動建立 OpsItems (PowerShell)

手動建立 OpsItems (主控台)

您可以使用 OpsItems AWS Systems Manager 主控台手動建立 。建立 OpsItem 後，它會顯示在您的 OpsCenter 帳戶中。如果為 OpsCenter 設定了跨帳戶管理，則 OpsCenter 會向委派管理員或管理帳戶提供為選定成員帳戶建立 OpsItems 的選項。如需詳細資訊，請參閱(選用) 手動設定 OpsCenter 以跨帳戶集中管理 OpsItems。

OpsItem 使用 AWS Systems Manager 主控台建立

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 OpsCenter。
3. 選擇 Create OpsItem (建立 OpsItem)。如果您沒有看到此按鈕，請選擇 OpsItems 索引標籤，然後選擇 Create OpsItem (建立 OpsItem)。
4. (選用) 選擇其他帳戶，然後選擇您要建立 OpsItem 的帳戶。

   > **Note**
   >
   > 如果您要為會員帳戶建立 OpsItems，則需要執行此步驟。

5. 針對 Title (標題)，輸入描述名稱，協助您了解 OpsItem 的用途。

6. 對於 Source (來源)，輸入受影響的 AWS 資源類型或其他來源資訊，協助使用者了解 OpsItem 的根源。

   > **Note**
   >
   > 建立 OpsItem 後，您不能編輯 Source (來源) 欄位。

7. (選用) 對於 Priority (優先順序)，選擇優先順序層級。
8. (選用) 對於 Severity (嚴重性)，選擇嚴重性層級。
9. (選用) 對於 Category (類別)，選擇類別。
10. 針對 Description (描述)，輸入此 OpsItem 的資訊，包括 (如合適) 重現問題的步驟。

    > **Note**
    >
    > 主控台支援 OpsItem 描述欄位中的大多數 Markdown 格式。如需詳細資訊，請參閱《AWS Management Console 入門指南》中的在主控台中使用 Markdown。

11. 在重複資料刪除字串中，輸入系統可用來檢查重複 OpsItems 的單字。如需重複資料刪除字串的詳細資訊，請參閱管理重複的 OpsItems。
12. (選用) 在通知中，指定您希望 OpsItem 更新時發送通知的 Amazon SNS 主題的 Amazon Resource Name (ARN)。您必須指定與 相同的 Amazon SNS ARN AWS 區域 OpsItem。
13. (選用) 在相關資源中，選擇新增，指定受影響資源和任何相關資源的 ID 或 ARN。
14. 選擇 Create OpsItem (建立 OpsItem)。

如果成功，頁面會顯示 OpsItem。委派管理員或管理帳戶為選取的成員帳戶建立 OpsItem 後，新 OpsItems 會顯示在管理員和成員帳戶的 OpsCenter 中。如需如何在 OpsItem 中設定選項的資訊，請參閱 管理 OpsItems。

手動建立 OpsItems (AWS CLI)

下列處理程序說明如何使用 AWS Command Line Interface (AWS CLI) 建立 OpsItem。

OpsItem 使用 建立 AWS CLI

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。
2. 開啟 AWS CLI 並執行下列命令來建立 OpsItem。將每個########取代為您自己的資訊。

```
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"
```

從檔案指定營運資料

當您建立 OpsItem 時，您可以從檔案指定營運資料。這個檔案必須是 JSON 檔案，且檔案內容必須使用以下格式。

```
{
  "key_name": {
    "Type": "SearchableString",
    "Value": "Up to 20 KB of data"
  }
}
```

請見此處範例。

```
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**
>
> 如需如何在不同的本機作業系統的命令列輸入 JSON 格式參數的資訊，請參閱《AWS Command Line Interface 使用者指南》中的在 AWS CLI中搭配字串使用引號。

系統會傳回相關資訊，如下所示。

```
{
    "OpsItemId": "oi-1a2b3c4d5e6f"
}
```

3. 執行以下命令來檢視您建立的 OpsItem 的詳細資訊。

```
aws ssm get-ops-item --ops-item-id ID
```

系統會傳回相關資訊，如下所示。

```
{
    "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. 執行下列命令以更新 OpsItem。此命令會將狀態從 Open (預設值) 變更成 InProgress。

```
aws ssm update-ops-item --ops-item-id ID --status InProgress
```

此命令無輸出。

5. 再次執行以下命令，確認狀態是否已變更為 InProgress。

```
aws ssm get-ops-item --ops-item-id ID
```

建立 OpsItem 的範例

以下範例會說明如何使用 Linux 管理入口網站、macOS 或 Windows 建立 OpsItem。

Linux 管理入口網站或 macOS

以下命令會在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體磁碟已滿時建立 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"
```

以下命令會在 OperationalData 中使用 /aws/resources 金鑰，來建立含 Amazon DynamoDB 相關資源的 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"
```

以下命令會在 OperationalData 中使用 /aws/automations 金鑰，來建立可將 AWS-ASGEnterStandby 文件指定為相關聯的 Automation 執行手冊的 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

以下命令會在 Amazon Relational Database Service (Amazon RDS) 執行個體未回應時建立 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"
```

以下命令會在 `OperationalData` 中使用 `/aws/resources` 金鑰，來建立含 Amazon EC2 執行個體相關資源的 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\"}}
```

以下命令會在 `OperationalData` 中使用 `/aws/automations` 金鑰，來建立可將 `AWS-RestartEC2Instance` 執行手冊指定為相關聯的 Automation 執行手冊的 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\"}}
```

手動建立 OpsItems (PowerShell)

下列程序說明如何使用 AWS Tools for Windows PowerShell建立 OpsItem。

OpsItem 使用 建立 AWS Tools for Windows PowerShell

1. 開啟 AWS Tools for Windows PowerShell 並執行下列命令來指定您的登入資料。

   ```
   Set-AWSCredentials -AccessKey key-name -SecretKey key-name
   ```

2. 執行下列命令， AWS 區域 為您的PowerShell工作階段設定 。

   ```
   Set-DefaultAWSRegion -Region Region
   ```

3. 執行下列命令以建立新的 OpsItem。將每個########取代為您自己的資訊。這個命令會指定 Systems Manager Automation Runbook 來修復此 OpsItem。

   ```
   $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，命令會輸出新 OpsItem 的 ID。

以下範例指定受損 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體的 Amazon Resource Name (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 中的工具 AWS Systems Manager，OpsItems會從建立追蹤到解析。如果為 OpsCenter 設定了跨帳戶管理，則委派管理員或管理帳戶可以從其帳戶管理 OpsItems。如需詳細資訊，請參閱(選用) 手動設定 OpsCenter 以跨帳戶集中管理 OpsItems。

您可以使用 Systems Manager 主控台中的下列頁面來檢視和管理 OpsItems：

- 摘要 – 顯示「待處理」和「進行中」的 OpsItems 計數、依來源和存在時間分類的 OpsItems 計數，以及操作洞察的計數。您可以依來源和 OpsItems 狀態來篩選 OpsItems。

- OpsItems – 顯示包含多個資訊欄位的 OpsItems 清單，其例如標題、ID、優先順序、描述、OpsItem 的來源以及上次更新日期和時間。您可以使用此頁面手動建立 OpsItems、設定來源、變更 OpsItem 的狀態，以及依新事件篩選 OpsItems。您可以選擇 OpsItem 以顯示其 OpsItems 詳細資訊頁面。

- OpsItem 詳細資訊 – 提供詳細的洞察和工具，供您用來管理 OpsItem。OpsItems 詳細資訊頁面包含下列索引標籤：
  - 概觀 – 顯示相關資源、過去 30 天內執行的執行手冊，以及您可以執行的可用執行手冊清單。您還可以檢視類似的 OpsItems，新增操作資料和新增相關的 OpsItems。
  - 相關資源詳細資訊 – 顯示來自數個 AWS 服務的資源相關資訊。展開資源詳細資訊區段，以檢視託管資源 AWS 的服務所提供的資源相關資訊。您也可以使用 Related resources (相關資源) 清單切換與此 OpsItem 關聯的其他相關資源。

如需有關如何管理 OpsItems 的詳細資訊，請參閱下列主題。

主題

- 檢視 OpsItem 的詳細資訊

- 編輯 OpsItem
- 將相關的資源新增至 OpsItem
- 將相關的 OpsItems 新增至 OpsItem
- 將操作資料新增至 OpsItem
- 為 OpsItem 建立事件
- 管理重複的 OpsItems
- 分析操作洞察以減少 OpsItems
- 檢視 OpsCenter 日誌和報告

檢視 OpsItem 的詳細資訊

若要全面檢視 OpsItem，請使用 OpsCenter 主控台中的 OpsItem 詳細資訊頁面。概觀頁面會顯示以下資訊：

- OpsItems 詳細資訊 – 顯示所選 OpsItem 的一般資訊。
- 相關資源 – 相關資源是受影響的資源，或啟動建立 OpsItem 之事件的資源。
- 過去 30 天的自動化執行 – 過去 30 天內執行的執行手冊清單。
- 執行手冊 – 您可以從可用的執行手冊清單中選擇一個執行手冊。
- 類似的 OpsItem – 這是系統產生的 OpsItems 清單，其可能是與您有關或是您感興趣的項目。若要產生清單，系統會掃描所有 OpsItems 的標題和描述，並傳回使用類似字詞的 OpsItems。
- 操作資料 – 操作資料是自訂的資料，可提供有關 OpsItem 的實用參考詳細資訊。例如，您可以指定日誌檔案、錯誤字串、授權金鑰、故障診斷秘訣，或其他相關資料。
- 相關的 OpsItem – 您可指定以某種方式與目前 OpsItem 相關之 OpsItems 的 ID。
- 相關資源詳細資訊 – 顯示資料提供者，包括 Amazon CloudWatch 指標和警示、 AWS CloudTrail 日誌和來自其中的詳細資訊 AWS Config。

檢視 OpsItem 的詳細資訊

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 OpsCenter。
3. 選擇 OpsItem 來檢視其詳細資訊。

編輯 OpsItem

OpsItem 詳細資訊區段包含有關 OpsItem 的資訊，包括描述、標題、來源、OpsItem ID 和狀態。您可以編輯單一 OpsItem，或可以選取多個 OpsItems，然後編輯下列欄位：狀態、優先順序、嚴重性、類別。

Amazon EventBridge 建立 OpsItem 時，會填入標題、來源和描述欄位。您可以編輯標題和描述欄位，但無法編輯來源欄位。

> **Note**
>
> 主控台支援 OpsItem 描述欄位中的大多數 Markdown 格式。如需詳細資訊，請參閱《 入門指南》中的在主控台中使用 Markdown。 AWS Management Console

一般而言，您可以編輯 OpsItem 的下列可設定資料：

- 標題 – OpsItem 的名稱。來源會建立 OpsItem 的標題。
- 描述 – 關於此 OpsItem 的資訊，包含 (如合適) 重現問題的步驟。
- 狀態：OpsItem 的狀態。如需有效狀態值的清單，請參閱《AWS Systems Manager API 參考》中的 OpsItem Status。
- 優先順序 – OpsItem 的優先順序可以介於 1 至 5 之間。建議您的組織決定每個優先順序層級的意義，及各層級對應的服務水準協議。
- 嚴重性 – OpsItem 的嚴重性可以介於 1 至 4 之間，其中 1 為嚴重，2 為高，3 為中等，4 為低。
- 類別 – OpsItem 的類別可以是可用性、成本、效能、復原或安全性。
- 通知 – 編輯 OpsItem 時，可以在通知欄位中指定 Amazon Simple Notification Service 主題的 Amazon Resource Name (ARN)。透過指定 ARN，您確保所有利害關係人都會在編輯 OpsItem 時收到通知，包括狀態變更。如需詳細資訊，請參閱《Amazon Simple Notification Service 開發人員指南》。

> **Important**
>
> Amazon SNS 主題必須存在於與 AWS 區域 相同的 中OpsItem。如果主題和 OpsItem 分屬不同區域，系統會傳回錯誤。

OpsCenter 具有與 的雙向整合 AWS Security Hub。更新與安全問題清單相關的 OpsItem 狀態和嚴重性時，這些變更會自動傳送至 Security Hub，以確保您永遠看到最新且正確的資訊。

從 Security Hub 調查結果建立 OpsItem 時，Security Hub 中繼資料會自動新增至 OpsItem 的操作資料欄位。如果刪除這些中繼資料，雙向更新將無法再運作。

編輯 OpsItem 詳細資訊

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 OpsCenter。
3. 選擇 OpsItem ID 以開啟詳細資訊頁面或選擇多個 OpsItems。如果選擇多個 OpsItems，則您只能編輯狀態、優先順序、嚴重性或類別。如果編輯多個 OpsItems，則 OpsCenter 會在您選擇新的狀態、優先順序、嚴重性或類別時立即更新並儲存您的變更。
4. 在 OpsItem details (OpsItem 詳細資訊) 區段，選擇 Edit (編輯)。
5. 根據您組織指定的需求和準則編輯 OpsItem 的詳細資訊。
6. 完成後，請選擇儲存。

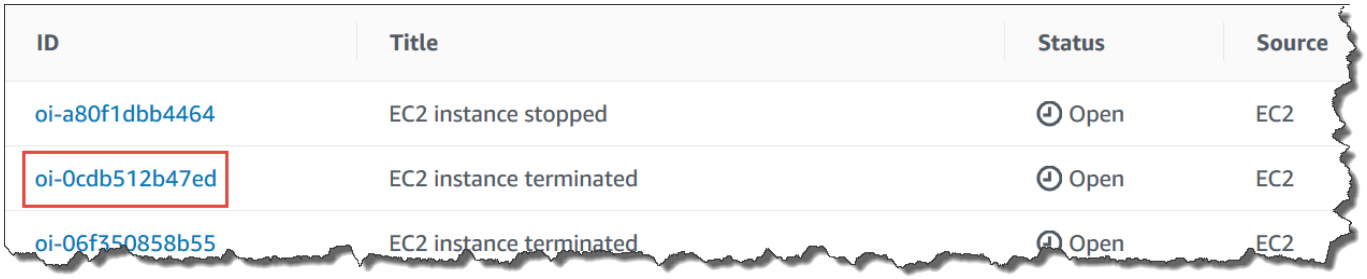
將相關的資源新增至 OpsItem

每個 OpsItem 都包含相關資源區段，此區域會列出相關資源的 Amazon Resource Name (ARN)。相關資源是需要調查的受影響 AWS 資源。

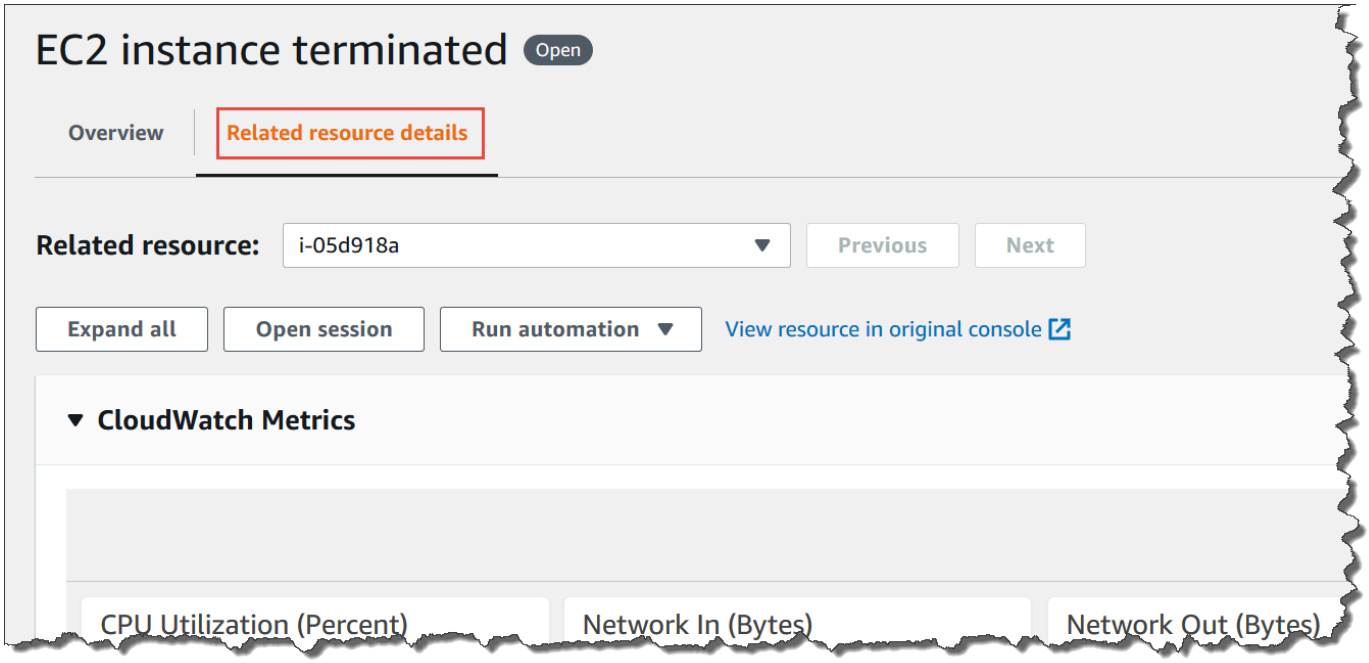
如果 Amazon EventBridge 建立 OpsItem，系統會自動使用資源的 ARN 填入 OpsItem。您可以手動指定相關資源的 ARN。針對某些 ARN 類型，OpsCenter 會自動建立深層連結，在 OpsCenter 主控台中直接顯示資源的詳細資訊。例如，如果您將一個 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體的 ARN 指定為相關資源，則 OpsCenter 會提取該 EC2 執行個體的詳細資訊。這可讓您檢視受影響 AWS 資源的詳細資訊，而不必離開 OpsCenter。

檢視相關資源並新增至 OpsItem

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 OpsCenter。
3. 選擇 (OpsItems) 索引標籤。
4. 選擇 OpsItem ID。



5. 若要檢視受影響資源的資訊，請選擇 Related resources details (相關資源詳細資訊) 標籤。



這個索引標籤會顯示來自多種 AWS 服務之資源的相關資訊。展開資源詳細資訊區段，以檢視由託管資源 AWS 服務 的 所提供此資源的相關資訊。您也可以使用 Related resources (相關資源) 清單切換與此 OpsItem 關聯的其他相關資源。

6. 若要新增其他相關資源，請選擇 Overview (概觀) 標籤。
7. 在 Related resources (相關資源) 區段中，選擇 Add (新增)。
8. 對於 Resource Type (資源類型)，選擇清單中的資源。
9. 對於 Resource ID (資源 ID)，輸入 ID 或 Amazon Resource Name (ARN)。您所選擇的資訊類型取決於您在上個步驟所選擇的資源。

> **Note**
>
> 您可以手動新增其他相關資源的 ARN。每個 OpsItem 最多可以列出 100 個相關的資源 ARN。

下表列出了會自動建立相關資源深層連結的資源類型。

支援的資源類型

資源名稱	ARN 格式
AWS Certificate Manager 憑證	`arn:aws:acm: region:account-id :certificate/ certificate-id`
Amazon EC2 Auto Scaling 群組	`arn:aws:autoscaling: region:account-id :autoScalingGroup: groupid:autoScalingGroupName/ groupfriendlyname`
Amazon CloudFront 分佈	`arn:aws:cloudfront:: account-id :*`
AWS CloudFormation 堆疊	`arn:aws:cloudformation: region:account-id :stack/stackname /additionalidentifier`
Amazon CloudWatch 警示	`arn:aws:cloudwatch: region:account-id :alarm:alarm-name`
AWS CloudTrail 追蹤	`arn:aws:cloudtrail: region:account-id :trail/trailname`
AWS CodeBuild 專案	`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`

資源名稱	ARN 格式
Amazon DynamoDB 資料表	`arn:aws:dynamodb: region:account-id :table/tablename`
Amazon Elastic Compute Cloud (Amazon EC2) 客戶閘道	`arn:aws:ec2: region:account-id :customer-gateway/ cgw-id`
Amazon EC2 彈性 IP	`arn:aws:ec2: region:account-id :eip/eipalloc-id`
Amazon EC2 專用主機	`arn:aws:ec2: region:account-id :dedicated-host/ host-id`
Amazon EC2 執行個體	`arn:aws:ec2: region:account-id :instance/ instance-id`
Amazon EC2 網際網路閘道	`arn:aws:ec2: region:account-id :internet-gateway/ igw-id`
Amazon EC2 網路存取控制清單 (network ACL)	`arn:aws:ec2: region:account-id :network-acl/ nacl-id`
Amazon EC2 網路界面	`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 安全群組	`arn:aws:ec2: region:account-id :security-group/ security-group-id`

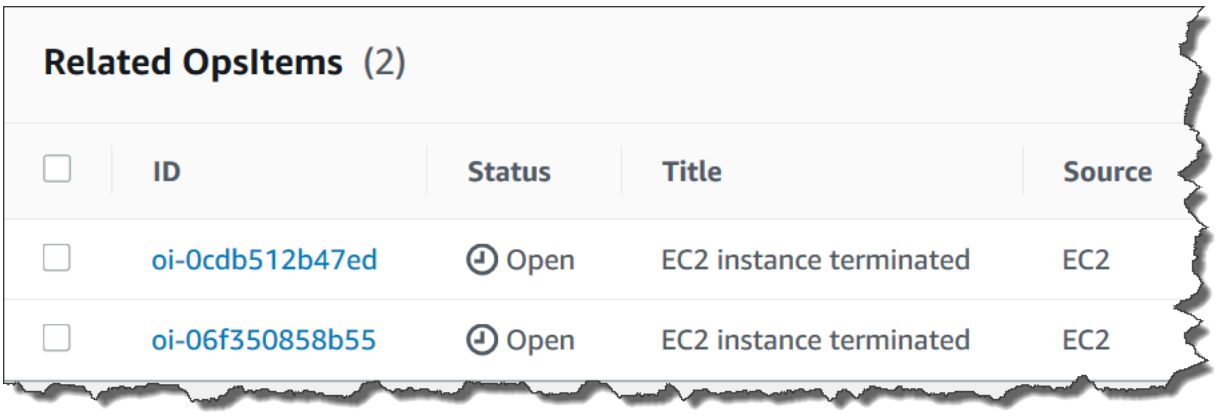
資源名稱	ARN 格式
Amazon EC2 子網路	`arn:aws:ec2: region:account-id :subnet/subnet-id`
Amazon EC2 磁碟區	`arn:aws:ec2: region:account-id :volume/volume-id`
Amazon EC2 VPC	`arn:aws:ec2: region:account-id :vpc/vpc-id`
Amazon EC2 VPN 連接	`arn:aws:ec2: region:account-id :vpn-connection/ vpn-id`
Amazon EC2 VPN 閘道	`arn:aws:ec2: region:account-id :vpn-gateway/ vgw-id`
AWS Elastic Beanstalk 應用程式	`arn:aws:elasticbeanstalk: region:account-id :application/ applicationname`
Elastic Load Balancing (Classic Load Balancer)	`arn:aws:elasticloadbalancing: region:account-id :loadbalancer/ name`
Elastic Load Balancing (Application Load Balancer)	`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`

資源名稱	ARN 格式
AWS Identity and Access Management (IAM) 群組	`arn:aws:iam:: account-id :group/group-name`
IAM 政策	`arn:aws:iam:: account-id :policy/policy-name`
IAM 角色	`arn:aws:iam:: account-id :role/role-name`
IAM 使用者	`arn:aws:iam:: account-id :user/user-name`
AWS Lambda 函數	`arn:aws:lambda: region:account-id :function: function-name`
Amazon Relational Database Service (Amazon RDS) 叢集	`arn:aws:rds: region:account-id :cluster: db-cluster-name`
Amazon RDS 資料庫執行個體	`arn:aws:rds: region:account-id :db:db-instance-name`
Amazon RDS 訂閱	`arn:aws:rds: region:account-id :es:subscription-name`
Amazon RDS 安全群組	`arn:aws:rds: region:account-id :secgrp:security-group-name`
Amazon RDS 叢集快照	`arn:aws:rds: region:account-id :cluster-snapshot:  cluster-snapshot-name`

資源名稱	ARN 格式
Amazon RDS 子網路群組	`arn:aws:rds:region:account-id:subgrp:subnet-group-name`
Amazon Redshift 叢集	`arn:aws:redshift:region:account-id:cluster:cluster-name`
Amazon Redshift 參數群組	`arn:aws:redshift:region:account-id:parametergroup:parameter-group-name`
Amazon Redshift 安全群組	`arn:aws:redshift:region:account-id:securitygroup:security-group-name`
Amazon Redshift 叢集快照	`arn:aws:redshift:region:account-id:snapshot:cluster-name/snapshot-name`
Amazon Redshift 子網路群組	`arn:aws:redshift:region:account-id:subnetgroup:subnet-group-name`
Amazon Simple Storage Service (Amazon S3) 儲存貯體	`arn:aws:s3:::bucket_name`
AWS Config 受 AWS Systems Manager 管節點庫存的記錄	`arn:aws:ssm:region:account-id:managed-instance-inventory/node_id`
Systems Manager State Manager 關聯	`arn:aws:ssm:region:account-id:association/association_ID`

將相關的 OpsItems 新增至 OpsItem

透過使用 OpsItems 詳細資訊頁面相關的 OpsItems，您可以調查操作問題並提供問題的情境。OpsItems 能以不同的方式建立關聯，包括 OpsItems 之間的上/下層關係、根本原因或重複性。您可以將一個 OpsItem 與另一個建立關聯，以便在相關的 OpsItem區段中顯示。您可以為與目前 OpsItem 相關的其他 OpsItems 指定最多 10 個 ID。

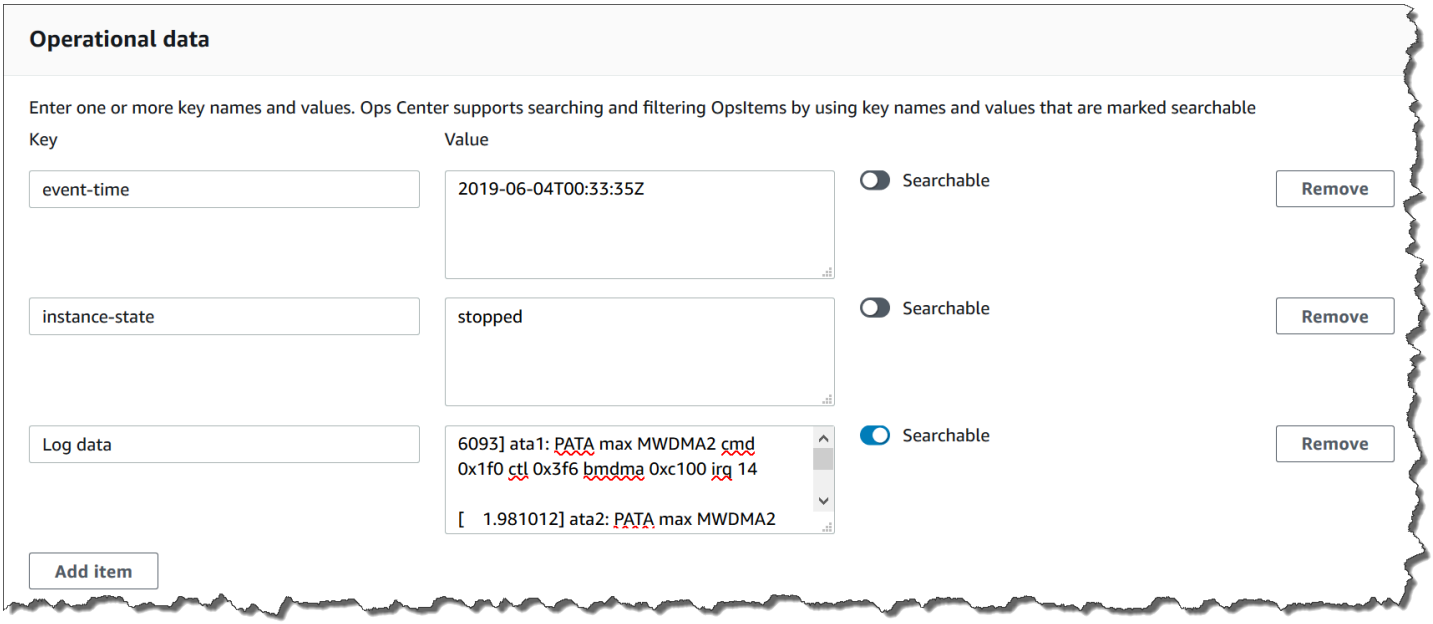


新增相關 OpsItem

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 OpsCenter。
3. 選擇 OpsItem ID 以開啟詳細資訊頁面。
4. 在 Related (相關的 OpsItem) 區段中，選擇 Add (新增)。
5. 針對 OpsItem ID，指定 ID。
6. 選擇新增。

將操作資料新增至 OpsItem

操作資料是自訂的資料，可提供有關 OpsItem 的實用參考詳細資訊。您可以輸入多個營運資料的金鑰/值對。例如，您可以指定日誌檔案、錯誤字串、授權金鑰、故障診斷秘訣，或其他相關資料。索引鍵的最大長度可以是 128 個字元，值的大小上限可以是 20 KB。



您可以讓帳戶中的其他使用者可搜尋此資料，也可以限制搜尋的存取權。可搜尋資料表示，所有能存取 OpsItem 概觀頁面的使用者 (如 DescribeOpsItems API 操作所提供) 都可以檢視和搜尋指定的資料。只有能夠存取 OpsItem 的使用者可檢視非可搜尋操作資料 (如 GetOpsItem API 動作所提供)。

將營運資料新增到 OpsItem

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 OpsCenter。
3. 選擇 OpsItem ID 以開啟其詳細資訊頁面。
4. 展開操作資料。
5. 如果沒有 OpsItem 的操作資料，則選擇新增。如果已有 OpsItem 的營運資料，請選擇 Manage (管理)。

   在您建立營運資料之後，您可以選擇 Manage (管理)，編輯金鑰和值、移除營運資料，或新增其他金鑰/值對。
6. 針對 Key (金鑰)，指定一或多個字詞，協助使用者了解資料的用途。

> **Important**
>
> 操作資料金鑰無法以下列項目開頭：amazon、aws、amzn、ssm、/amazon、/aws、/amzn、/ssm。

7. 針對 Value (值)，指定資料。
8. 選擇 Save (儲存)。

> **Note**
>
> 您可以透過使用 OpsItems 頁面上的 Operational data (營運資料) 運算子來篩選 OpsItems。在搜尋方塊中，選擇操作資料，然後以 JSON 輸入索引鍵/值組。您必須使用以下格式來輸入索引鍵/值組：`{"key":"key_name","value":"a_value"}`

為 OpsItem 建立事件

使用下列程序，手動建立 OpsItem 的事件以追蹤和管理它 AWS Systems Manager Incident Manager，這是 中的工具 AWS Systems Manager。事件是指任何意外中斷或服務質量下降。如需有關 Incident Manager 的詳細資訊，請參閱 the section called “將 OpsCenter 與其他 AWS 服務整合”。

手動建立 OpsItem 事件

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 OpsCenter。
3. 如果 Incident Manager 為您建立 OpsItem，請選擇它並前往步驟 5。如果沒有，請選擇 Create OpsItem (建立 OpsItem) 並填寫表格。如果您沒有看到此按鈕，請選擇 OpsItems 索引標籤，然後選擇 Create OpsItem (建立 OpsItem)。
4. 如果建立了 OpsItem，則請將其開啟。
5. 選擇 Start Incident (開始事件)。
6. 在回應計劃中，選擇您要指派給此事件的 Incident Manager 回應計劃。
7. (選用) 對於 Title (標題) 中，輸入描述名稱，以協助其他團隊成員了解事件的本質。如果不輸入新的標題，OpsCenter 會使用回應計劃中的標題建立 OpsItem 和 Incident Manager 中的對應事件。
8. (選用)對於 Incident impact (事件影響)，選擇此事件的影響層級。如果您沒有選擇影響層級，OpsCenter 會使用回應計劃中的標題建立 OpsItem 和 Incident Manager 中的對應事件。
9. 選擇 開始使用。

管理重複的 OpsItems

OpsCenter 可以從多個 AWS 服務接收單一來源的多個重複 OpsItems。OpsCenter 會使用內建邏輯和可設定重複資料刪除字串的組合來避免建立重複的 OpsItems。在呼叫 CreateOpsItem API 操作時，AWS Systems Manager 會套用重複資料刪除內建邏輯。

AWS Systems Manager 使用以下重複資料刪除邏輯：

1. 建立 OpsItem 時，Systems Manager 會根據重複資料刪除字串和起始 OpsItem 的資源來建立會和存放雜湊。
2. 如果收到了另一個建立 OpsItem 的請求，系統會檢查新請求的重複資料刪除字串。
3. 如果此重複資料刪除字串存在相符的雜湊，Systems Manager 會檢查現有 OpsItem 的狀態。如果現有 OpsItem 的狀態為「待處理」或「進行中」，則不會建立 OpsItem。如果已解決現有的 OpsItem，Systems Manager 會建立新的 OpsItem。

建立 OpsItem 後，您就不能在該 OpsItem 中編輯或變更重複資料刪除字串。

若要管理重複的 OpsItems，可以執行以下操作：

- 編輯針對以 OpsCenter 為目標的 Amazon EventBridge 規則的重複資料刪除字串。如需詳細資訊，請參閱編輯預設 EventBridge 規則中的重複資料刪除字串。
- 請在您手動建立 OpsItem 時，指定重複資料刪除字串。如需詳細資訊，請參閱使用 指定重複資料刪除字串 AWS CLI。
- 使用操作洞察來檢閱和解決重複的 OpsItems。您可以使用執行手冊來解決重複的 OpsItems。

  為協助您解決重複的 OpsItems 並減少來源建立之 OpsItems 的數量，Systems Manager 提供了自動化執行手冊。如需相關資訊，請參閱 根據洞察解決重複 OpsItems。

編輯預設 EventBridge 規則中的重複資料刪除字串

使用下列處理程序指定以 OpsCenter 為目標之 EventBridge 規則的重複資料刪除字串。

編輯 EventBridge 規則的重複資料刪除字串

1. 登入 AWS Management Console ，並在 https://console.aws.amazon.com/events/：// 開啟 Amazon EventBridge 主控台。
2. 在導覽窗格中，選擇規則。
3. 選擇規則，然後選擇 Edit (編輯)。

4. 前往 Select target(s) (選擇目標) 頁面。
5. 在 Additional settings (其他設定) 區段中，選擇 Configure input transformer (設定輸入轉換器)。
6. 在 Template (範本) 方塊中，尋找 `"operationalData": { "/aws/dedup"` JSON 項目和您要編輯的重複資料刪除字串。

   EventBridge 規則中的重複資料刪除字串項目使用下列 JSON 格式。

   ```
   "operationalData": { "/aws/dedup": {"type": "SearchableString","value":
    "{\"dedupString\":\"Words the system should use to check for duplicate
    OpsItems\"}"}}
   ```

   請見此處範例。

   ```
   "operationalData": { "/aws/dedup": {"type": "SearchableString","value":
    "{\"dedupString\":\"SSMOpsCenter-EBS-volume-performance-issue\"}"}}
   ```
7. 編輯重複資料刪除字串，然後選擇確認。
8. 選擇 Next (下一步)。
9. 選擇下一步。
10. 選擇更新規則。

使用 指定重複資料刪除字串 AWS CLI

當您OpsItem使用 AWS Systems Manager 主控台或 手動建立新的 時，可以指定重複資料刪除字串 AWS CLI。如需在主控台手動建立 OpsItem 時輸入重複資料刪除字串的詳細資訊，請參閱 手動建立 OpsItems。如果使用的是 AWS CLI，則可以為 `OperationalData` 參數輸入重複資料刪除字串。參數語法要使用 JSON，如下範例所示。

```
--operational-data '{"/aws/dedup":{"Value":"{\"dedupString\": \"Words the system should
 use to check for duplicate OpsItems\"}","Type":"SearchableString"}}'
```

這裡的範例命令會指定 `disk full` 的重複資料刪除字串。

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"
```

分析操作洞察以減少 OpsItems

OpsCenter 操作洞察會顯示有關重複的 OpsItems 的資訊。OpsCenter 會自動分析您帳戶中的 OpsItems 並產生三種類型的洞察結果。您可以在 OpsCenter 摘要索引標籤的操作洞察區段中檢視此資訊。

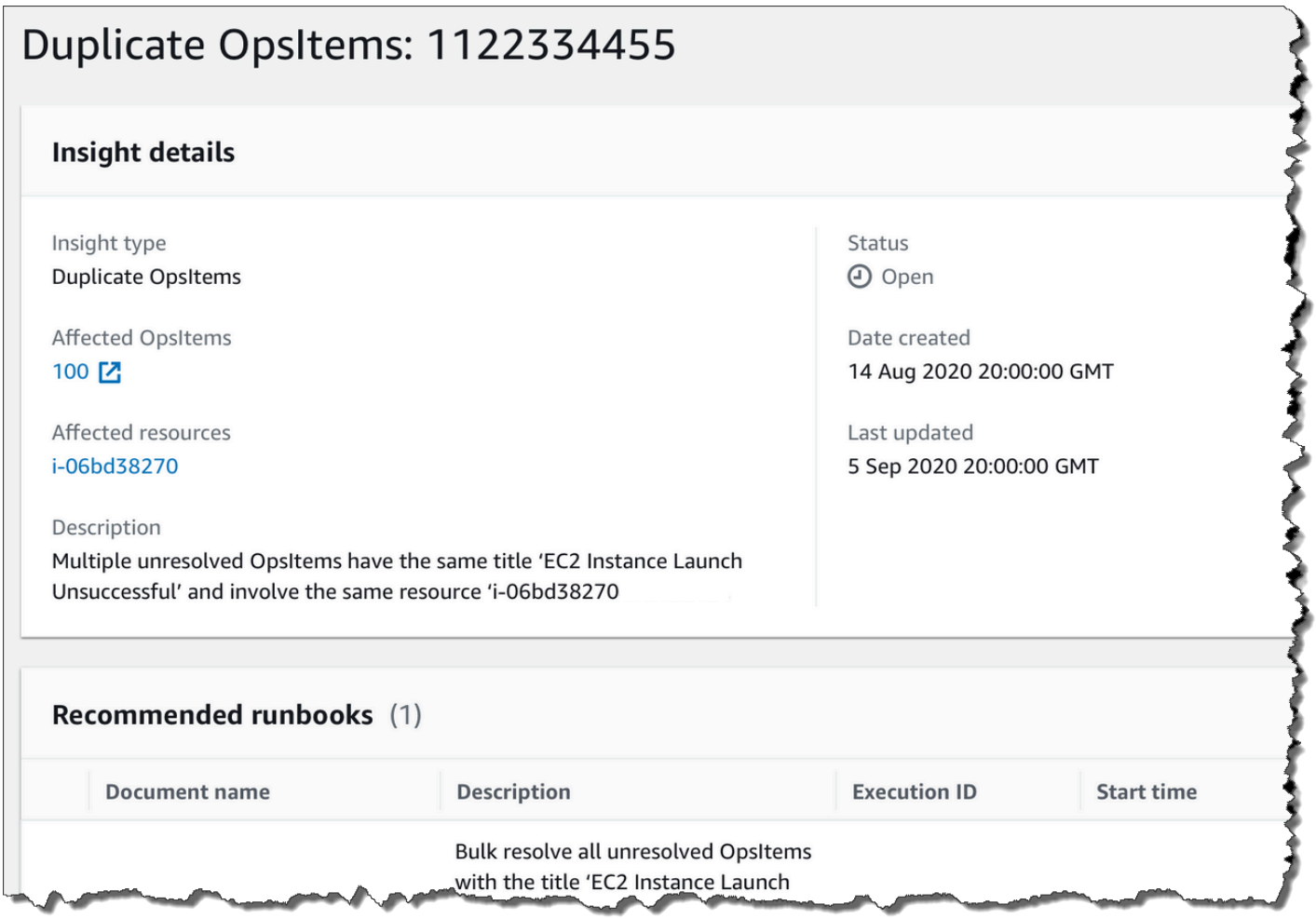
- 重複的 OpsItems：當八個或更多 OpsItems 具有相同資源的相同標題時，便會產生一個洞察結果。
- 最常見的標題：當超過 50 個 OpsItems 具有相同的標題時，便會產生一個洞察結果。
- 產生最多的資源 OpsItems – 當 AWS 資源有超過 10 個開啟的 時，會產生洞見OpsItems。這些洞察結果及其對應的資源會顯示在 OpsCenter 摘要索引標籤中的產生最多 OpsItems 資源表格中。資源會以 OpsItem 計數遞減的順序列出。

> **Note**
>
> OpsCenter 會為下列資源類型建立產生最多 OpsItems 的資源洞察結果：
>
> - Amazon Elastic Compute Cloud (Amazon EC2) 執行個體
> - Amazon EC2 安全群組
> - Amazon EC2 Auto Scaling 群組

- Amazon Relational Database Service (Amazon RDS) 資料庫
- Amazon RDS 叢集
- AWS Lambda 函數
- Amazon DynamoDB 資料表
- Elastic Load Balancing 負載平衡器
- Amazon Redshift 叢集
- AWS Certificate Manager 憑證
- Amazon Elastic Block Store 磁碟區

OpsCenter 強制執行每種類型最多 15 個洞察結果的限制。如果某個類型達到此限制，則 OpsCenter 不再顯示該類型的更多洞察結果。若要檢視更多洞察結果，您必須解決與該類型的操作洞察結果相關聯的所有 OpsItems。如果因為 15 個洞察結果限制而無法在主控台中顯示某個待處理的洞察結果，則在關閉另一個洞察結果之後，該洞察結果就會顯示出來。

選擇洞察後，OpsCenter 會顯示受影響 OpsItems 和資源的相關資訊。以下螢幕擷取畫面顯示範例，其中包含重複 OpsItem 洞察的詳細資訊。



操作洞察預設為關閉。如需有關使用操作洞察的詳細資訊，請參閱下列主題。

主題

- 啟用操作洞察
- 根據洞察解決重複 OpsItems
- 停用操作洞察

啟用操作洞察

您可以在 Systems Manager 主控台的 OpsCenter 頁面上啟用操作洞察。啟用操作洞察後，Systems Manager 會建立稱為 `AWSServiceRoleForAmazonSSM_OpsInsights` 的 AWS Identity and Access Management (IAM) 服務連結角色。服務連結角色是直接連結至 Systems Manager 的特殊 IAM 角色類型。服務連結角色是預先定義的，並包含服務 AWS 服務 代表您呼叫其他 所需的所有許可。如需 `AWSServiceRoleForAmazonSSM_OpsInsights` 服務連結角色的詳細資訊，請參閱使用角色在 Systems Manager OpsCenter 中建立操作洞察 OpsItems。

> **Note**
>
> 記下以下重要資訊：
>
> - 您的 AWS 帳戶 需支付操作洞察費用。如需詳細資訊，請參閱 AWS Systems Manager 定價。
> - OpsCenter 會定期使用批次處理重新整理洞察。這表示顯示在 OpsCenter 中的洞察清單可能不同步。

使用以下程序以在 OpsCenter 中啟用和檢視操作洞察。

啟用和檢視操作洞察

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 OpsCenter。
3. 在操作洞察可用訊息方塊中，選擇啟用。如果您沒有看到此訊息，請向下捲動至操作洞察區段，然後選擇啟用。
4. 啟用此功能後，在摘要索引標籤上，向下捲動至操作洞察區段。
5. 若要檢視已篩選的洞察結果清單，請選擇重複的 OpsItems、最常見的標題或產生最多 OpsItems 的資源旁的連結。若要檢視所有洞察，請選擇 View all operational insights (檢視所有操作洞察)。
6. 選擇洞察 ID 以檢視詳細資訊。

根據洞察解決重複 OpsItems

若要解決洞察，您必須先解決與洞察相關聯的所有 OpsItems。您可以使用 `AWS-BulkResolveOpsItemsForInsight` Runbook 來解決與洞察相關聯的 OpsItems。

為協助您解決重複的 OpsItems 並減少來源建立之 OpsItems 的數量，Systems Manager 提供了下列自動化執行手冊：

- `AWS-BulkResolveOpsItems` Runbook 會解決符合指定篩選條件的 OpsItems。
- `AWS-AddOpsItemDedupStringToEventBridgeRule` 執行手冊新增了與指定 Amazon EventBridge 規則相關聯之所有 OpsItem 目標的重複資料刪除字串。如果規則已經有重複資料刪除字串，則執行手冊不會新增重複資料刪除字串。

- 如果規則產生數十或數百個 OpsItems，則 `AWS-DisableEventBridgeRule` 會關閉 EventBridge 中的規則。

解決操作洞察結果

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 OpsCenter。
3. 在 Overview (概觀) 索引標籤上，向下捲動至 Operational insights (操作洞察)。
4. 選擇檢視所有操作洞察結果。
5. 選擇洞察 ID 以檢視詳細資訊。
6. 選擇一個執行手冊，然後選擇執行。

停用操作洞察

關閉操作洞察後，系統會停止建立新的洞察，並停止在主控台中顯示洞察。任何作用中的洞察都會在系統中保持不變，儘管您不會在主控台中看到這些資訊。如果再次啟用此功能，則系統會顯示先前未解決的洞察，並開始建立新的洞察。使用下列程序來關閉操作洞察。

關閉操作洞察

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 OpsCenter。
3. 選擇設定。
4. 在 Operational insights (操作洞察) 區段中，選擇 Edit (編輯)，然後切換 Disable (停用) 選項。
5. 選擇 Save (儲存)。

檢視 OpsCenter 日誌和報告

AWS CloudTrail 會記錄 AWS Systems Manager OpsCenter對 主控台、 AWS Command Line Interface (AWS CLI) 和 SDK 的 API 呼叫。您可以在 CloudTrail 主控台中，或在 Amazon Simple Storage Service (Amazon S3) 儲存貯體中檢視資訊。Amazon S3 會使用儲存貯體存放您帳戶的所有 CloudTrail 日誌。

OpsCenter 動作的日誌顯示建立、更新、取得和描述 OpsItem 活動。如需檢視和使用 Systems Manager 活動的 CloudTrail 日誌的詳細資訊，請參閱 使用 記錄 AWS Systems Manager API 呼叫 AWS CloudTrail。

AWS Systems Manager OpsCenter 為您提供下列有關 的資訊OpsItems：

- OpsItem 狀態摘要 – 依狀態 (「待處理和正在進行」、「待處理」或「正在進行」) 提供 OpsItems 摘要。
- 最多開啟的來源 OpsItems – 提供開啟 的頂端明細 AWS 服務 OpsItems。
- OpsItems (依來源和存在時間) – 提供依來源和建立天數分組的 OpsItems 計數。

檢視 OpsCenter 報告摘要

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 OpsCenter。
3. 在 OpsItems 概觀頁面上，選擇摘要。
4. 在 OpsItems by source and age (依來源和年齡的 OpsItem) 下，選擇搜尋列以根據 Source (來源) 篩選 OpsItems。使用清單來根據 Status (狀態) 進行篩選。

刪除 OpsItems

您可以使用 OpsItem AWS Command Line Interface 或 AWS SDK 呼叫刪除OpsItem API 操作來刪除個人。您無法刪除 AWS Management Console中的 OpsItem。若要刪除 OpsItem，您的 AWS Identity and Access Management (IAM) 使用者、群組或角色必須具有管理員許可，或者您必須獲得呼叫 `DeleteOpsItem` API 操作的許可。

> **Important**
>
> 請留意有關此操作的下列重要資訊。
>
> - 刪除 OpsItem 是無法復原的動作。您無法復原已刪除的 OpsItem。
> - 此操作會使用最終一致性模式，這表示系統可能需要幾分鐘的時間才能完成此操作。如果您刪除 OpsItem 並立即呼叫 (例如 GetOpsItem)，則已刪除的 OpsItem 可能仍會出現在回應中。
> - 此為等冪操作。如果您為相同的 OpsItem 重複呼叫此操作，則系統不會拋出異常。如果第一次呼叫成功，所有其他呼叫均會傳回與第一次呼叫相同的成功回應。

- 此操作不支援跨帳戶呼叫。委派管理員或管理帳戶無法刪除其他帳戶中的 OpsItems，即使 OpsCenter 已設定了跨帳戶管理亦然。如需有關跨帳戶管理的詳細資訊，請參閱 (選用) 設定 OpsCenter 以跨帳戶集中管理 OpsItems。
- 如果您收到 `OpsItemLimitExceededException`，您就可以刪除一或多個 OpsItems，以減少低於配額限制的 OpsItems 總數。如需此例外狀況的詳細資訊，請參閱 使用 OpsCenter 來疑難排解問題。

刪除 OpsItem

使用下列程序來刪除 OpsItem。

刪除 OpsItem

1. 如果您尚未安裝和設定 AWS CLI。如需詳細資訊，請參閱安裝或更新最新版本的 AWS CLI。
2. 執行下列命令。以您要刪除的 OpsItem 之 ID 來取代 *ID*。

```
aws ssm delete-ops-item --ops-item-id ID
```

如果成功，此命令不會傳回任何資料。

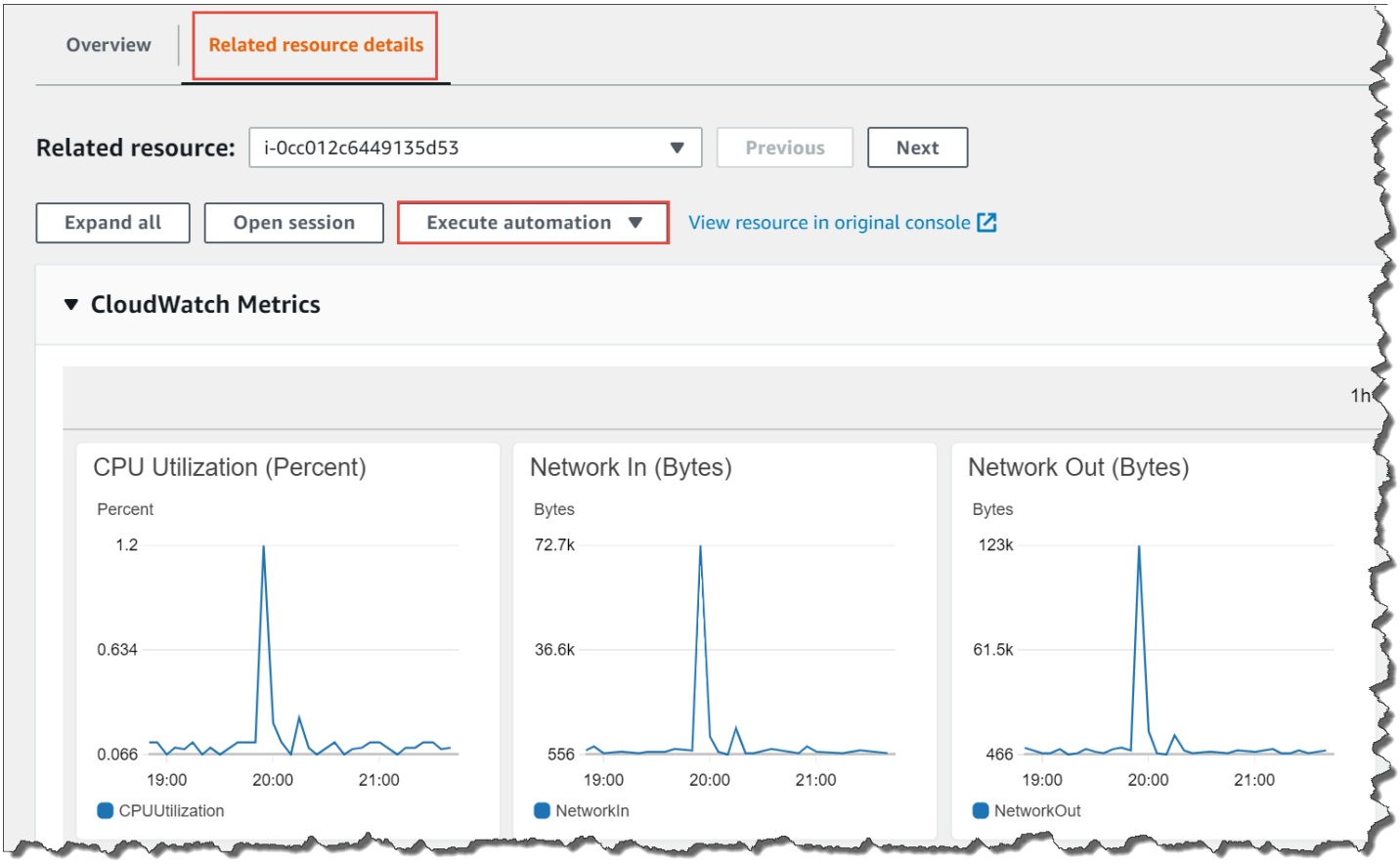
修正 OpsItem 問題

使用 AWS Systems Manager Automation Runbook，您可以修復 中識別 AWS 的資源問題OpsItem。自動化使用預先定義的 Runbook 來修復 AWS 資源的常見問題。

每個 OpsItem 都包括執行手冊區段，其中提供了可用於修復的執行手冊清單。當您從清單中選擇 Automation 執行手冊時，OpsCenter 會自動顯示一些執行文件所需的欄位。執行 Automation 執行手冊時，系統會將執行手冊與 OpsItem 的相關資源建立關聯。Amazon EventBridge 建立 OpsItem 後，它會將執行手冊與該 OpsItem 建立關聯。OpsCenter 會為 OpsItem 保留 30 天的 Automation 執行手冊記錄。

您可以選擇狀態來檢視有關該執行手冊的重要詳細資訊，例如自動化失敗的原因，以及發生失敗時 Automation 執行手冊執行的步驟，如下列範例所示。

所選 OpsItem 的 Related resource details (相關資源詳細資訊) 頁面包括 Run automation (執行自動化) 清單。您可以選擇最近或資源特定的 Automation 執行手冊進行執行以修復問題。此頁面也包含資料提供者，包括 Amazon CloudWatch 指標和警示、 AWS CloudTrail 日誌和來自 的詳細資訊 AWS Config。



您可以在主控台中選擇 Runbook 名稱或使用 Systems Manager Automation Runbook 參考，檢視 Automation Runbook 資訊。

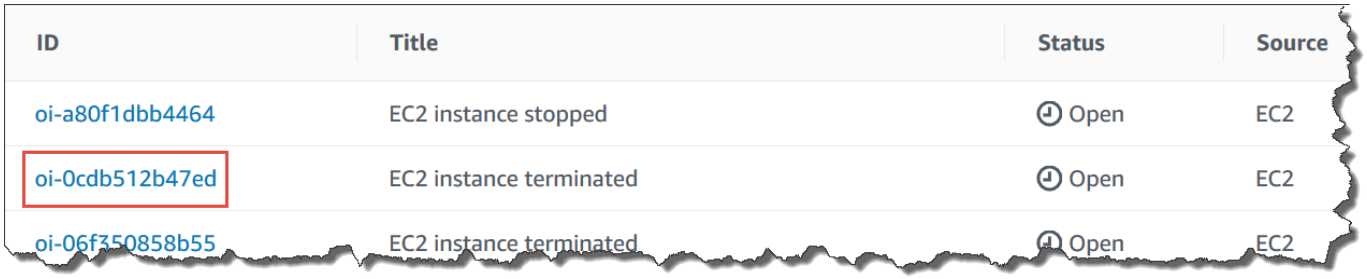
使用執行手冊修復 OpsItem

在您使用 Automation 執行手冊來修復 OpsItem 問題之前，請執行下列動作：

- 驗證您是否具有執行 Systems Manager Automation Runbook 的許可。如需詳細資訊，請參閱設定自動化。
- 收集要執行之自動化的資源特定 ID 資訊。例如，如果想要執行可重新啟動 EC2 執行個體的自動化，則必須指定要重新啟動的 EC2 執行個體 ID。

執行 Automation Runbook 修復 OpsItem 問題

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 OpsCenter。
3. 選擇 OpsItem ID 以開啟詳細資訊頁面。



4. 捲動至 Runbooks 區段。
5. 使用搜尋列或右上角的號碼尋找您要執行的 Automation 執行手冊。
6. 選擇 Runbook，然後選擇 Execute (執行)。
7. 輸入執行手冊所需的資訊，然後選擇提交。

   啟動執行手冊，系統會返回上一個畫面並顯示狀態。
8. 在過去 30 天的 Automation 執行區段中，選擇執行 ID 連結以檢視步驟和執行狀態。

使用關聯的執行手冊修復 OpsItem

在您執行 OpsItem 的 Automation 執行手冊之後，OpsCenter 會將執行手冊與該 OpsItem 建立關聯。相關聯的執行手冊排名會高於執行手冊清單中的其他執行手冊。

使用下列處理程序執行已與 OpsItem 中相關資源建立關聯的 Automation Runbook。如需新增相關資源的資訊，請參閱 管理 OpsItems。

執行與資源相關聯之 Runbook 以修復 OpsItem 問題

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 OpsCenter。
3. 開啟 OpsItem。
4. 在 Related resources (相關資源) 區段中，選擇您要執行 Automation Runbook 的資源。
5. 選擇 Run automation (執行自動化)，然後選擇您要執行的相關聯 Automation Runbook。
6. 輸入 Runbook 所需的資訊，然後選擇 Execute (執行)。

   啟動執行手冊，系統會返回上一個畫面並顯示狀態。
7. 在過去 30 天的 Automation 執行區段中，選擇執行 ID 連結以檢視步驟和執行狀態。

檢視 OpsCenter 摘要報告

AWS Systems Manager OpsCenter 包含會自動顯示下列資訊的摘要頁面：

- OpsItem 狀態摘要：按狀態列出的 OpsItems 摘要，例如 `Open` 和 `In progress`。
- 最多開啟的來源 OpsItems – AWS 服務 已開啟 的頂端明細OpsItems。
- OpsItems (依來源和存在時間) ：依來源和建立天數分組的 OpsItems 計數。

檢視 OpsCenter 摘要報告

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 OpsCenter，然後選擇摘要索引標籤。
3. 在 OpsItems (依來源和存在時間) 區段中，執行下列操作：
    1. (選用) 在篩選欄位中，選擇來源，選取 `Equal`、`Begin With`、或 `Not Equal`，然後輸入搜尋參數。
    2. 在相鄰的清單中選取下列狀態值之一：
        - `Open`
        - `In progress`
        - `Resolved`
        - `Open and in progress`
        - `All`

使用 OpsCenter 來疑難排解問題

本主題包含的資訊可協助您針對 OpsCenter 的常見錯誤和問題進行疑難排解。

您收到 OpsItemLimitExceededException

如果您的 AWS 帳戶 已達到呼叫 CreateOpsItem API 操作時OpsItems允許的數目上限，您會收到 `OpsItemLimitExceededException`。如果您OpsItems的呼叫超過下列任一配額的 數目上限，則會OpsCenter傳回例外狀況：

- OpsItems AWS 帳戶 每個區域 （ 包括 `Open`和 `Resolved` OpsItems) 的總數：500，000
- OpsItems AWS 帳戶 每月數量上限：10，000

這些配額適用於從任何來源建立的 OpsItems，但下列項目除外：

- OpsItems 由 AWS Security Hub 問題清單建立
- 當事件管理員事件開啟時自動生成的 OpsItems

從這些來源建立的 OpsItems 不會計入您的 OpsItem 配額中，但會針對每個 OpsItem 向您收取費用。

如果您收到 `OpsItemLimitExceededException`，您就可以手動刪除 OpsItems，直到您低於使您無法建立新 OpsItem 的配額為止。同樣地，為 Security Hub 調查結果或事件管理員事件建立的 OpsItems 不會減少配額強制執行的 OpsItems 之總數。您必須從其他來源刪除 OpsItems。如需有關如何刪除 OpsItem 的資訊，請參閱 刪除 OpsItems。

您收到來自 的大量 AWS 自動產生的大額帳單 OpsItems

如果您已設定與 的整合 AWS Security Hub， 會OpsItems為 Security Hub 問題清單OpsCenter建立 。根據 Security Hub 產生的調查結果數量與您在設定整合時登入的帳戶而定，OpsCenter 可能會產生大量的 OpsItems，且成本昂貴。以下是與 Security Hub 調查結果產生的 OpsItems 相關之更具體的詳細資料：

- 如果您在設定 OpsCenter 與 Security Hub 整合時使用 Security Hub 管理員帳戶登入，則系統會針對管理員和所有成員帳戶中的調查結果建立 OpsItems。所有 OpsItems 都會在管理員帳戶中建立。取決於各種因素，這可能導致您需要向 AWS支付超乎預期的大帳單。

  如果您在設定整合時使用成員帳戶登入，則系統只會針對該個別帳戶中的調查結果建立 OpsItems。如需有關 Security Hub 管理員帳戶、成員帳戶及其與調查結果 EventBridge 事件摘要之關係的詳細資訊，請參閱《AWS Security Hub 使用者指南》中的 Security Hub 與 EventBridge 整合的類型一節。
- 針對每個建立 OpsItem 的調查結果，系統會按照標準價格向您收取建立 OpsItem 的費用。如果您編輯 OpsItem 或對應的調查結果已在 Security Hub 中更新 (會觸發 OpsItem 更新)，系統也會向您收取費用。
- OpsItems 與 整合所建立的 目前 AWS Security Hub 不受區域中OpsItems每個帳戶 500，000 個配額上限的限制。因此，Security Hub 提醒可以在帳戶中OpsItems的每個區域中建立超過 500，000 個收費項目。

  因此，對於高生產環境，我們建議將 Security Hub 問題清單的範圍限制為僅高嚴重性的問題。

> **Important**
>
> 如果您認為大量 OpsItems 是錯誤建立的，且您的 AWS 帳單未經保證，請聯絡 支援。

如果您不再希望系統針對 Security Hub 調查結果建立 OpsItems，請使用以下程序。

停止接收針對 Security Hub 調查結果的 OpsItems

1. 開啟 AWS Systems Manager 主控台，網址為 https：//https://console.aws.amazon.com/systems-manager/。
2. 在導覽窗格中，選擇 OpsCenter。
3. 選擇設定。
4. 在 Security Hub 調查結果區段中，選擇編輯。
5. 選擇滑桿以將已啟用變更為已停用。如果您無法切換滑桿，表示 Security Hub 尚未為您的 AWS 帳戶啟用。
6. 選擇儲存以儲存您的組態。OpsCenter 不再根據 Security Hub 調查結果建立 OpsItems。

> **Important**
>
> 如果 將設定OpsCenter切換回已啟用並繼續OpsItems為問題清單建立，請登入 Systems Manager 委派管理員帳戶或 AWS Organizations 管理帳戶，然後重複此程序。如果您沒有登入這些帳戶的許可，請聯絡您的系統管理員，並要求他們重複此程序以停用帳戶的整合功能。

使用 Systems Manager 託管的 Amazon CloudWatch 儀表板

Amazon CloudWatch 儀表板是 CloudWatch 主控台中可自訂的首頁，可讓您在單一檢視中監控資源，甚至是分散在不同 AWS 區域的那些資源。您可以使用 CloudWatch 儀表板來建立 AWS 資源指標和警示的自訂檢視。您可以利用儀表板來建立以下項目：

- 選取指標和警示的單一檢視，可協助您評估在一或多個 AWS 區域之資源和應用程式的運作狀態。您可以選取在每個圖形上用於每個指標的顏色，如此您就可以追蹤跨多個圖形的相同指標。
- 運作手冊，其會提供運作事件期間適用於團隊成員與如何回應特定事件的指導。
- 關鍵資源和應用程式測量的常用檢視，可以由團隊成員共用以在運作事件期間形成較快的通訊流程。

您可以使用 主控台、 AWS Command Line Interface (AWS CLI) 或使用 CloudWatch `PutDashboard` API 來建立儀表板。如需詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的使用 Amazon CloudWatch 儀表板。

使用 SSM Agent

AWS Systems Manager Agent (SSM Agent) 是在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、邊緣裝置、內部部署伺服器和虛擬機器 (VMs) 上執行的 Amazon 軟體。 SSM Agent 可讓 Systems Manager 更新、管理和設定這些資源。代理程式會在 中處理來自 Systems Manager 服務的請求 AWS 雲端，然後依照請求中指定的執行它們。 SSM Agent然後使用 Amazon Message Gateway Service(ssmmessages) 將狀態和執行資訊傳回給 Systems Manager 服務。（在 2024 年之前 AWS 區域 啟動時，狀態和執行資訊也可能由 傳回 Amazon Message Delivery Service（服務字首：)ec2messages。）

如果您監控流量，則會看到受管節點會與 ssmmessages.* 端點通訊，也可能會與 ec2messages.* 端點通訊。如需詳細資訊，請參閱參考：ec2messages、ssmmessages 和其他 API 操作。如需有關將 SSM Agent 日誌傳送至 Amazon CloudWatch Logs 的詳細資訊，請參閱在 中記錄和監控 AWS Systems Manager。

目錄

- 了解 SSM Agent 的技術詳細資訊
- 尋找預先安裝了 SSM Agent的 AMIs
- 在 Linux 的 EC2 執行個體使用 SSM Agent
- 在 macOS 專用 EC2 執行個體使用 SSM Agent
- 在 Windows Server 專用 EC2 執行個體使用 SSM Agent
- 檢查 SSM Agent 狀態並啟動代理程式
- 檢查 SSM Agent 版本編號
- 檢視 SSM Agent 日誌
- 限制透過 SSM Agent 存取根層級命令
- 自動化 SSM Agent 更新
- 訂閱 SSM Agent 通知
- SSM Agent 疑難排解

了解 SSM Agent 的技術詳細資訊

使用本主題中的資訊來協助您實作 AWS Systems Manager 代理程式 (SSM Agent)，並了解代理程式的運作方式。

主題

- SSM Agent 3.2.x.x 版憑證行為
- SSM Agent 憑證優先順序
- 設定 SSM Agent 以與聯邦資訊處理標準 (FIPS) 搭配使用
- 關於本機 ssm 使用者帳戶
- SSM Agent 與 Instance Metadata Service (IMDS)
- 讓 SSM Agent 保持最新狀態
- 確定未修改、移動或刪除 SSM Agent 安裝目錄
- SSM Agent 滾動更新 AWS 區域
- SSM Agent 與 AWS 受管 S3 儲存貯體通訊

SSM Agent 3.2.x.x 版憑證行為

當使用 Quick Setup 中的預設主機管理組態登入執行個體時，SSM Agent 會將一組臨時憑證資料儲存在 `/var/lib/amazon/ssm/credentials` (適用於 Linux 和 macOS) 或 `%PROGRAMFILES%\Amazon\SSM\credentials` (適用於 Windows Server)。這些臨時憑證具有您為預設主機管理組態選擇的 IAM 角色指定的許可。在 Linux 中，只有 root 帳戶可以存取這些憑證。在 Windows Server 中，只有 SYSTEM 帳戶和本機管理員可以存取這些憑證。

SSM Agent 憑證優先順序

本主題描述了 SSM Agent 如何授予在您資源上執行動作之許可的重要資訊。

> **Note**
>
> 對邊緣裝置的支援略有不同。您必須將邊緣裝置設定為使用 AWS IoT Greengrass 核心軟體、設定 AWS Identity and Access Management (IAM) 服務角色，以及使用 部署SSM Agent至您的裝置 AWS IoT Greengrass。如需詳細資訊，請參閱使用 Systems Manager 管理邊緣裝置。

SSM Agent 安裝在機器上時，其需要許可才能與 Systems Manager 服務進行通訊。在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上，會在連接到執行個體的執行個體設定檔中提供這些許可。在非 EC2 機器上，SSM Agent 通常會從共用憑證檔案中取得所需的許可，該檔案位於 `/root/.aws/credentials` (Linux 和 macOS) 或 `%USERPROFILE%\.aws\credentials` (Windows Server)。在混合啟用程序中，會將所需的許可新增至此檔案。

然而，在極少數情況下，機器最終可能會將許可新增至多個位置，SSM Agent 在這些位置會檢查許可以執行其任務。

例如，假如您已將某個 EC2 執行個體設定為由 Systems Manager 管理，則該組態包含連接執行個體設定檔。然後您決定也將該執行個體用於開發人員或最終使用者任務，並安裝 AWS Command Line Interface (AWS CLI)。此安裝會導致其他許可新增至執行個體上的憑證檔案。

當您在執行個體上執行 Systems Manager 命令時，SSM Agent 可能會嘗試使用與您預期使用的憑證不同的憑證，例如憑證檔案而非執行個體設定檔。這是因為 SSM Agent 會依照指定的順序尋找預設憑證供應者鏈結的憑證。

> **Note**
>
> 在 Linux 和 macOS 中，SSM Agent 以根使用者身分執行。因此，SSM Agent 在此程序中查找的環境變數和憑證檔案只是根使用者 (`/root/.aws/credentials`) 的環境變數和憑證檔案。在搜尋憑證期間，SSM Agent 不會查看執行個體上的任何其他使用者的環境變數或憑證檔案。

預設供應者鏈結會依以下順序查找憑證：

1. 環境變數 (如果已設定) (`AWS_ACCESS_KEY_ID` 和 `AWS_SECRET_ACCESS_KEY`)。
2. 共用的憑證檔案 (Linux 的 `$HOME/.aws/credentials` 以及 Windows Server 的 macOS 或 `%USERPROFILE%\.aws\credentials`)，例如具有混合啟用或 AWS CLI 安裝提供的許可。
3. 如果應用程式存在，且使用 Amazon Elastic Container Service (Amazon ECS) 任務定義或 RunTask API 操作，則為任務的 AWS Identity and Access Management (IAM) 角色。
4. 連接至 Amazon EC2 執行個體的執行個體設定檔。
5. 為預設主機管理組態選擇的 IAM 角色。

如需相關資訊，請參閱下列主題：

- EC2 執行個體設定檔：設定 Systems Manager 所需的執行個體許可
- 混合啟用：建立混合啟用以向 Systems Manager 註冊節點
- AWS CLI 登入資料 – AWS Command Line Interface 《使用者指南》中的組態和登入資料檔案設定
- 預設憑證供應者鏈結 - 適用於 Go 的 AWS SDK 開發人員指南中的指定憑證

> **Note**
>
> 適用於 Go 的 AWS SDK 開發人員指南中的本主題描述了 SDK for Go 的預設供應者鏈結；但是，相同的原則適用於評估 SSM Agent 憑證。

設定 SSM Agent 以與聯邦資訊處理標準 (FIPS) 搭配使用

如果您需要搭配使用 Systems Manager 與聯邦資訊處理標準 (FIPS) 140-3 驗證的密碼編譯模組，您可以將 AWS Systems Manager Agent (SSM Agent) 設定為在支援的區域中使用 FIPS 端點。

設定 SSM Agent 以連線至 FIPS 140-3 端點

1. 連線至受管節點。
2. 前往內含 `amazon-ssm-agent.json` 檔案的目錄：
   - Linux：`/etc/amazon/ssm/`
   - macOS: `/opt/aws/ssm/`
   - Windows Server: `C:\Program Files\Amazon\SSM\`
3. 開啟檔案 `amazon-ssm-agent.json` 進行編輯。

   > **Tip**
   >
   > 如果尚無 `amazon-ssm-agent.json` 檔案，請將 `amazon-ssm-agent.json.template` 的內容複製到名為 `amazon-ssm-agent.json` 的新檔案。將 `amazon-ssm-agent.json` 儲存在 `amazon-ssm-agent.json.template` 所在的相同目錄中。

4. 將下列內容新增至檔案。將*##*預留位置值取代為分割區的適當區域碼：

```
{
    ---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"
    }
}
```

支援的區域包含以下區域：

- us-east-1，表示美國東部 (維吉尼亞北部) 區域
- us-east-2，表示美國東部 (俄亥俄) 區域
- us-west-1，表示美國西部 (加利佛尼亞北部) 區域
- us-west-2，表示美國西部 (奧勒岡) 區域
- ca-west-1，表示加拿大西部 (卡加利) 區域

5. 儲存檔案，並重新啟動 SSM Agent。

每次變更組態時，請重新啟動 SSM Agent。

您可以使用相同的程序來自訂 SSM Agent 的其他功能。如需可用組態屬性及其預設值的最新清單，請參閱 GitHub 上 amazon-ssm-agent 儲存庫中的組態屬性定義。

如需 FIPS AWS 支援的詳細資訊，請參閱聯邦資訊處理標準 (FIPS) 140-3。

關於本機 ssm 使用者帳戶

從 SSM Agent 的 2.3.50.0 版開始，代理程式會建立稱為 ssm-user 的本機使用者帳戶，並將其新增至 /etc/sudoers.d 目錄 (Linux 和 macOS) 或系統管理員群組 (Windows Server)。在代理程式 2.3.612.0 之前的版本中，當 SSM Agent 第一次啟動，或在安裝後重新啟動時，會建立該帳戶。在版本 2.3.612.0 和更高版本中，當執行個體上初次啟動工作階段時，會建立 ssm-user 帳戶。ssm-user 這是工作階段在 中啟動時的預設作業系統使用者Session Manager，這是 中的工具 AWS Systems Manager。您可以將 ssm-user 移動到權限較少的群組或變更 sudoers 檔案，以改變許可。解除安裝 SSM Agent 時，ssm-user 帳戶並不會從系統中移除。

在 Windows Server 上，SSM Agent 會處理當每個工作階段開始時，為 ssm-user 帳戶設定新的密碼。Linux 受管執行個體上沒有設定 ssm-user 的密碼。

從 SSM Agent 的 2.3.612.0 版本開始，不會在用作網域控制器的 Windows Server 機器上自動建立 ssm-user 帳戶。若要在 Windows Server 網域控制器上使用 Session Manager，請手動建立 ssm-user 帳戶 (如果尚不存在)，並將網域管理員許可指派給使用者。

> **Important**
>
> 為了建立 ssm-user 帳戶，連接到執行個體的執行個體設定檔必須提供必要的許可。如需相關資訊，請參閱步驟 2：為 Session Manager 確認或新增執行個體許可。

SSM Agent 與 Instance Metadata Service (IMDS)

Systems Manager 依賴 EC2 執行個體中繼資料才能正確運作。Systems Manager 可以使用 Instance Metadata Service 的第 1 版或第 2 版 (IMDSv1 和 IMDSv2) 來存取執行個體中繼資料。您的執行個體必須可以存取執行個體中繼資料服務的 IPv4 地址：169.254.169.254。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的執行個體中繼資料與使用者資料。

讓 SSM Agent 保持最新狀態

每當將新工具新增至 Systems Manager 或對現有工具進行更新時，就會SSM Agent發佈 的更新版本。未使用最新版本的代理程式，可能會使您的受管節點無法使用各種 Systems Manager 工具和功能。因此，我們建議您讓機器的 SSM Agent 自動保持最新狀態。如需相關資訊，請參閱自動化 SSM Agent 更新。請訂閱 GitHub 上的 SSM Agent 版本備註頁面，以便接收有關 SSM Agent 更新的通知。

> **Note**
>
> 每當將新工具新增至 Systems Manager 或對現有工具進行更新時，就會SSM Agent發佈 的更新版本。未使用最新版本的代理程式，可能會使您的受管節點無法使用各種 Systems Manager 工具和功能。因此，我們建議您讓機器的 SSM Agent 自動保持最新狀態。如需相關資訊，請參閱自動化 SSM Agent 更新。請訂閱 GitHub 上的 SSM Agent 版本備註頁面，以便接收有關 SSM Agent 更新的通知。
>
> 根據預設，包含 SSM Agent 的 Amazon Machine Images (AMIs) 可能最長需要兩週，才能使用 SSM Agent 的最新版本進行更新。我們建議您設定更頻繁地自動更新 SSM Agent。

確定未修改、移動或刪除 SSM Agent 安裝目錄

SSM Agent 安裝於 `/var/lib/amazon/ssm/` (Linux 和 macOS) 以及 `%PROGRAMFILES%\Amazon\SSM\` (Windows Server)。這些安裝目錄包含 SSM Agent 使用的重要檔案和資料夾，例如認證檔案、處理程序間通訊 (IPC) 資源，以及協同運作資料夾。不應修改、移動或刪除安裝目錄內的任何內容。否則，SSM Agent 可能會停止正常運作。

SSM Agent 滾動更新 AWS 區域

在其GitHub儲存庫中提供SSM Agent更新後，最多可能需要兩週的時間，直到更新版本 AWS 區域 在不同時間推出至所有 為止。因此，您可能會在嘗試於區域中部署新版本 SSM Agent 時收到「不支援目前平台」或「正在將 amazon-ssm-agent 更新到較舊版本，請開啟允許繼續降級」等錯誤。

若要判斷您可使用的 SSM Agent 版本，可執行 `curl` 命令。

若要查看全域下載儲存貯體中可用的代理程式版本，請執行以下命令。

```
curl https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/VERSION
```

若要查看特定區域中可用的代理程式版本，請執行以下命令，使用您工作所在區域替代 *region*，例如使用 `us-east-2` 替代美國東部 (俄亥俄) 區域。

```
curl https://s3.region.amazonaws.com/amazon-ssm-region/latest/VERSION
```

您也可以直接在瀏覽器中開啟 `VERSION` 文件，而不需要 `curl` 命令。

SSM Agent 與 AWS 受管 S3 儲存貯體通訊

在執行各種 Systems Manager 操作的過程中， AWS Systems Manager 代理程式 (SSM Agent) 會存取多個 Amazon Simple Storage Service (Amazon S3) 儲存貯體。可公開存取這些 S3 儲存貯體，SSM Agent 預設使用 HTTP 呼叫連線到它們。

但是，如果您在 Systems Manager 操作中使用虛擬私有雲端 (VPC) 端點，則必須在適用於 Systems Manager 的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體設定檔中提供明確許可，或在混合多雲端環境中非 EC2 機器的服務角色中提供明確許可。否則，您的資源無法存取這些公有儲存貯體。

若要在使用 VPC 端點時對這些節點授予受管執行個體存取權，您可以建立自訂 Amazon S3 許可政策，然後將它連接到您的執行個體描述檔 (適用於 EC2 執行個體) 或服務角色 (適用於非 EC2 受管節點)。

如需有關在 Systems Manager 操作中使用虛擬私有雲端 (VPC) 端點的資訊，請參閱使用適用於 Systems Manager 的 VPC 端點來改善 EC2 執行個體的安全性。

> **Note**
>
> 這些許可僅提供 所需的 AWS 受管儲存貯體存取權SSM Agent。這些許可不提供其他 Amazon Simple Storage Service (Amazon S3) 操作所需的許可。這些許可也不提供自有 S3 儲存貯體的許可。

如需詳細資訊，請參閱下列主題：

- 設定 Systems Manager 所需的執行個體許可
- 在混合多雲端環境中建立 Systems Manager 所需的 IAM 服務角色
- 參考：用於修補操作的 Amazon S3 儲存貯體

目錄

- 所需的儲存貯體許可
- 範例
- 使用硬體指紋驗證啟用混合模式機器
- GitHub 的 SSM Agent

所需的儲存貯體許可

下表說明 SSM Agent 可能需要存取的每個 S3 儲存貯體，以便進行 Systems Manager 操作。

> **Note**
>
> *region* 代表 支援的 AWS 區域 識別符 AWS Systems Manager，例如us-east-2美國東部（俄亥俄）區域。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域欄位。

SSM Agent 需要的 Amazon Simple Storage Service (Amazon S3) 許可

S3 儲存貯體 ARN	描述
arn:aws:s3:::aws-windows-downloads- *region*/*	某些僅支援 Windows Server 作業系統以及某些提供跨平台支援的 SSM 文件 (例如 AWSEC2-ConfigureSTIG) 所必需。
arn:aws:s3:::amazon-ssm- *region*/*	更新 SSM Agent 安裝的必要項目。這些儲存貯體包含 SSM Agent 安裝套件及 AWS-UpdateSSMAgent 文件和外掛程式所參考的安裝資訊清單。如果沒有提供這些許可，則 SSM Agent 會進行 HTTP 呼叫以下載更新。
arn:aws:s3:::aws-ssm- *region*/*	支援存取 S3 儲存貯體，其中包含與非修補 Systems Manager 文件 (SSM Command 文件) 搭配使用所需的模組。例如：arn:aws:s3:::aws-ssm-us-east-2/* 。 以下是存放在這些儲存貯體中的一些常用 SSM 文件。 • AWS-ConfigureWindowsUpdate • AWS-FindWindowsUpdates • AWS-UpdateSSMAgent • AWS-UpdateEC2Config
arn:aws:s3:::patch-baseline-snapshot- *region*/* -或- arn:aws:s3:::patch-baseline-snapshot- *region-unique-suffix* /*	允許存取包含修補基準快照的 S3 儲存貯體。如果您使用下列任一 SSM Command 文件，則這是必需的： • AWS-RunPatchBaseline • AWS-RunPatchBaselineAssociation • AWS-RunPatchBaselineWithHooks • AWS-ApplyPatchBaseline (舊有 SSM 文件)

S3 儲存貯體 ARN	描述
	大多數支援的儲存貯體 AWS 區域 使用以下格式： `arn:aws:s3:::patch-baseline-snapshot-` *region* 對於某些區域，儲存貯體名稱中會包含額外的唯一尾碼。例如，中東 (巴林) 區域 (me-south-1) 的儲存貯體名稱如下所示： • `patch-baseline-snapshot-me-south-1-uduvl7q8` 如需修補基準快照儲存貯體名稱的完整清單，請參閱包含 AWS 受管修補基準快照的儲存貯體。 **Note** 如果您使用內部部署防火牆並計劃使用 Patch Manager，則該防火牆也必須允許存取適當的修補基準端點。

S3 儲存貯體 ARN	描述
對於 Linux 和 Windows Server 受管節點：`arn:aws:s3:::aws-patch-manager-` *`region`*`-`*`unique-suffix`*` /*` 對於 macOS 的 Amazon EC2 執行個體：`arn:aws:s3:::aws-patchmanager-macos-` *`region`*`-`*`unique-suffix`*` /*	支援存取包含 SSM Command 文件的 S3 儲存貯體，以便在 Patch Manager 中進行修補操作。每個儲存貯體名稱都包含唯一的尾碼，例如 552881074 表示美國東部 (俄亥俄) (us-east-2) 區域中的儲存貯體： • `arn:aws:s3:::aws-patch-manager-us-east-2-552881074/*` • `arn:aws:s3::: aws-patchmanager-macos-us-east-2-552881074/*` SSM 文件 以下是存放在這些儲存貯體中的一些常用 SSM 文件。 • `AWS-RunPatchBaseline` • `AWS-RunPatchBaselineAssociation` • `AWS-RunPatchBaselineWithHooks` • `AWS-InstanceRebootWithHooks` • `AWS-PatchAsgInstance` • `AWS-PatchInstanceWithRollback` 如需修補操作的 AWS 受管 S3 儲存貯體完整清單，請參閱下列主題： • 包含用於修補操作之 SSM Command 文件的儲存貯體 (Linux 和 Windows Server) • 包含用於修補操作之 SSM Command 文件的儲存貯體 (macOS)

範例

以下範例說明如何在美國東部 (俄亥俄) 區域 (us-east-2) 中提供 Systems Manager 操作所需的 S3 儲存貯體的存取權。在大多數情況下，只有在使用 VPC 端點時，才需要在執行個體設定檔或服務角色中明確提供這些許可。

> **Important**
>
> 我們建議您避免在這個政策中的特定區域使用萬用字元 (*)。例如，使用 `arn:aws:s3:::aws-ssm-us-east-2/*` 而不使用 `arn:aws:s3:::aws-ssm-*/*`。使用萬用字元可能允許存取您不想授與存取權的 S3 儲存貯體。如果您要將執行個體設定檔用於多個區域，建議您為每個區域重複第一個 `Statement` 區塊。

```
{
        "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/*"
                ]
            }
        ]
    }
```

使用硬體指紋驗證啟用混合模式機器

對於混合多雲端環境中的非 EC2 機器，SSM Agent 會收集許多系統屬性 (稱為硬體雜湊），並使用這些屬性來計算指紋。指紋是不透明字串，代理程式會將其傳遞給某些 Systems Manager API。此唯一的指紋會將呼叫者與特定的啟用混合模式受管節點建立關聯。代理程式會將指紋和硬體雜湊存放在某個位置的本機磁碟上 (稱為保存庫)。

當註冊機器以搭配 Systems Manager 使用時，代理程式會計算硬體雜湊和指紋。然後，當代理程式傳送 RegisterManagedInstance 命令時，指紋會傳回到 Systems Manager 服務。

稍後，當傳送 RequestManagedInstanceRoleToken 命令時，代理程式會檢查保存庫中的指紋和硬體雜湊，以確保目前的機器屬性與儲存的硬體雜湊相符。如果目前的機器屬性與儲存在「保存庫」中的硬體雜湊相符，則代理程式會將指紋從「保存庫」傳遞至 RegisterManagedInstance，從而產生成功呼叫。

如果目前的機器屬性與儲存的硬體雜湊不相符，SSM Agent 會計算新指紋，將新硬體雜湊和指紋儲存在「保存庫」中，並將新指紋傳遞至 RequestManagedInstanceRoleToken。這會導致 *RequestManagedInstanceRoleToken* 失敗，而代理程式將無法取得角色權杖以連線到 Systems Manager 服務。

此失敗是故意而為之，用來作為驗證步驟，以防止多個受管節點與作為相同受管節點的 Systems Manager 服務進行通訊。

將目前機器屬性與儲存在保存庫中的硬體雜湊比較時，代理程式會使用下列邏輯來判斷舊雜湊與新雜湊是否相符：

- 如果 SID (系統/機器 ID) 不同，則不符。
- 否則，如果 IP 地址相同，則相符。
- 否則，系統會計算相符的機器屬性百分比，並與使用者設定的相似度臨界值進行比較，以判斷是否有相符項。

相似度臨界值會儲存在保存庫中，作為硬體雜湊的一部分。

使用下列命令註冊執行個體後，可以設定相似度臨界值。

在 Linux 機器上：

```
sudo amazon-ssm-agent -fingerprint -similarityThreshold 1
```

在使用 Windows Server 的機器上：

```
cd "C:\Program Files\Amazon\SSM\" `
    .\amazon-ssm-agent.exe -fingerprint -similarityThreshold 1
```

> **Important**
>
> 如果用來計算指紋的其中一個元件發生變化，這可能會導致代理程式休眠。若要避免這種休眠狀態，請將相似度閾值設定為較低的值，例如 **1**。

GitHub 的 SSM Agent

可在 GitHub 上使用 SSM Agent 的原始碼，以便調整代理程式來符合需求。我們建議您為想要進行的變更提交提取請求。但是，Amazon Web Services 不支援執行修改過的軟體複本。

尋找預先安裝了 SSM Agent的 AMIs

AWS Systems Manager 代理程式 (SSM Agent) 預先安裝在 AWS 和信任的第三方提供的某些 Amazon Machine Images(AMIs) 上。

例如，當您使用下列其中一個作業系統，啟動透過 AMI 建立的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體時，您可能會發現已經安裝 SSM Agent：

- AlmaLinux
- 2017 年 9 月及更新版本的 Amazon Linux 1 基本 AMI
- Amazon Linux 2
- Amazon Linux 2 ECS 最佳化基礎 AMIs
- Amazon Linux 2023 (AL2023)
- Amazon EKS 最佳化的 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 位元 (Snap)、18.04、20.04、22.04 LTS、23.10、24.04 LTS、24.0 和 25.04
- Windows Server 2008-2012 R2 AMIs 發佈於 2016 月 11 月或之後
- Windows Server 2016、2019、2022 (Nano 版本除外) 和 2025

> **Note**
>
> AMI 上預先安裝的 SSM Agent版本可能不是可用的最新版本。作為最佳實務，建議在受管節點上一律使用 SSM Agent 的最新可用版本。如需有關 SSM Agent自動更新的詳細資訊，請參閱自動化 SSM Agent 更新。
> SSM Agent 可能預先安裝在此清單AMIs以外的 AWS 受管 上。這通常表示並非所有 Systems Manager 工具都完全支援作業系統 (OS)。
> SSM Agent 也可能預先安裝在 AMIs 或 社群AMIs儲存庫 AWS Marketplace 中的 ，但 AWS 不支援這些 AMIs。

驗證 SSM Agent 的狀態

根據初始化的時間而定，從上述清單中的 AMI 建立的執行個體可能沒有預先安裝 SSM Agent。執行個體也有可能已預先安裝代理程式，但代理程式未執行。因此，建議您首次嘗試在執行個體上使用 Systems Manager 之前，先檢查 SSM Agent 的狀態。

使用下列程序驗證 SSM Agent 已安裝並正在執行個體上執行。如果發現未安裝代理程式，您可以在 Linux、macOS 以及 Windows Server 執行個體上手動安裝。

驗證 SSM Agent 已安裝在執行個體

1. 啟動新的執行個體之後，請等待幾分鐘使其完成初始化。
2. 使用您偏好的方式連線至執行個體。例如，您可以使用 SSH 連線至 Linux 執行個體，或使用遠端桌面連線至 Windows Server 執行個體。
3. 針對執行個體的作業系統類型執行命令，以檢查 SSM Agent 的狀態。

作業系統	Command
Amazon Linux 1	`sudo status amazon-ssm-agent`
Amazon Linux 2 和 Amazon Linux 2023	`sudo systemctl status amazon-ssm-agent`
macOS	沒有用於在 macOS 上檢查 SSM Agent 狀態的命令。透過尋找和評估代理程式日誌檔案 /

作業系統	Command
	`var/log/amazon/ssm/amazon-ssm-agent.log` ，您可以檢查狀態。
SUSE Linux Enterprise Server	`sudo systemctl status amazon-ssm-agent`
Ubuntu Server (32 位元)	`sudo status amazon-ssm-agent`
Ubuntu Server (64 位元 - Deb)	`sudo systemctl status amazon-ssm-agent`
Ubuntu Server (64 位元 - Snap)	`sudo systemctl status snap.amazon-ssm-agent.amazon-ssm-agent.service`
Windows Server	`Get-Service AmazonSSMAgent`

> **Tip**
>
> 若要檢視命令以檢查 Systems Manager 所支援之所有作業系統類型的 SSM Agent 狀態，請參閱 檢查 SSM Agent 狀態並啟動代理程式。

4. 評估命令輸出，以了解 SSM Agent 的狀態。

   狀態：已安裝且正在執行

   在大多數情況下，命令輸出表明代理程式已安裝且正在執行。

   下列範例顯示在 Amazon Linux 2 執行個體上已安裝且正在執行 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--
```

   下列範例顯示在 Windows Server 執行個體上已安裝且正在執行 SSM Agent。

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

狀態：已安裝但未執行

在某些情況下，命令輸出表明代理程式已安裝但未執行。

下列範例顯示在 Amazon Linux 2 執行個體上已安裝但未執行 SSM Agent。

```
amazon-ssm-agent.service - amazon-ssm-agent
Loaded: loaded (/usr/lib/systemd/system/amazon-ssm-agent.service; enabled; vendor
 preset: enabled)
Active: inactive (dead) since Wed 2021-10-20 22:16:41 UTC; 18s ago
--truncated--
```

下列範例顯示在 Windows Server 執行個體上已安裝但未執行 SSM Agent。

```
Status   Name               DisplayName
------   ----               -----------
Stopped  AmazonSSMAgent     Amazon SSM Agent
```

如果代理程式已安裝但未執行，您可以使用適用於執行個體作業系統類型的命令手動啟用。

作業系統	Command
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`

作業系統	Command

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 位元)	`sudo start amazon-ssm-agent`

Ubuntu Server (64 位元 - Deb)	`sudo systemctl enable amazon-ssm-agent`
	
	`sudo systemctl start amazon-ssm-agent`

Ubuntu Server (64 位元 - Snap)	`sudo snap start amazon-ssm-agent`
Windows Server	在 PowerShell 中執行下列命令。 `Start-Service AmazonSSMAgent`

狀態：未安裝

在某些情況下，命令輸出表明代理程式未安裝。

下列範例顯示在 Amazon Linux 2 執行個體上未安裝 SSM Agent。

```
Unit amazon-ssm-agent.service could not be found.
```

下列範例顯示在 Windows Server 執行個體上未安裝 SSM Agent。

```
Get-Service : Cannot find any service with service name 'AmazonSSMAgent'.
--truncated--
```

如果未安裝代理程式，您可以使用適用於您作業系統類型的程序手動安裝：

- 在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent
- 在 macOS EC2 執行個體上手動安裝和解除安裝 SSM Agent
- 在 Windows Server EC2 執行個體上手動安裝和解除安裝 SSM Agent

在 Linux 的 EC2 執行個體使用 SSM Agent

AWS Systems Manager 代理程式 (SSM Agent) 會處理 Systems Manager 請求，並設定您的機器，如請求中所指定。在 Linux 作業系統使用下列主題的程序以安裝、設定或解除安裝 SSM Agent。

主題

- 驗證 SSM Agent 的簽章
- 在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent
- 將 SSM Agent 設定為使用 Linux 節點上的代理

驗證 SSM Agent 的簽章

Linux 執行個體的 AWS Systems Manager Agent (SSM Agent) deb 和 rpm 安裝程式套件會以密碼編譯方式簽署。您可以使用公有金鑰來驗證代理程式套件為原版且未經修改。如果檔案受損或遭到修改，驗證會失敗。您可以使用 RPM 或 GPG 來驗證安裝程式套件的簽章。以下資訊適用於 SSM Agent 版本 3.1.1141.0 或更新版本。

若要尋找執行個體架構和作業系統的正確簽章檔案，請參閱下表。

region 代表 支援的 AWS 區域 識別符 AWS Systems Manager，例如us-east-2美國東部（俄亥俄）區域。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

架構	作業系統	簽章檔案 URL	代理程式下載檔案名稱

x86_64	Almalinux、Amazon Linux 1、Amazon Linux 2、Amazon Linux 2023、CentOS、CentOS Stream、RHEL、Oracle Linux、Rocky Linux 和 SLES	`https://s3.region.amazonaws.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`
			`amazon-ssm-agent.rpm`

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/`
			`amazon-ssm-agent.deb`

架構	作業系統	簽章檔案 URL	代理程式下載檔案名稱
		debian_amd64/amazon-ssm-agent.deb.sig	
x86	Amazon Linux 1、Amazon Linux 2、Amazon Linux 2023、CentOS 和 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

架構	作業系統	簽章檔案 URL	代理程式下載檔案名稱
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

架構	作業系統	簽章檔案 URL	代理程式下載檔案名稱

ARM64	Amazon Linux 1、Amazon Linux 2、Amazon Linux 2023、CentOS 和 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 伺服器上驗證SSM Agent套件 (v3.3.1802.0 及更新版本)

開始之前

本節中的 GPG 和 RPM 程序適用於 3.3.1802.0 版和更新SSM Agent版本。SSM Agent 使用下列程序驗證 的簽章之前，請確定您已為您的作業系統下載最新的代理程式套件。例如 `https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/linux_arm64/amazon-ssm-agent.rpm`。如需下載 SSM Agent 套件的詳細資訊，請參閱在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent。

如果您有理由繼續使用代理程式 3.3.1611.0 版或更早版本，在 Linux 伺服器上驗證SSM Agent套件 (v3.3.1611.0 及更早版本) 請改為遵循 中的指示。

GPG

在 Linux 伺服器上驗證SSM Agent套件 (v3.3.1802.0 及更新版本）

1. 複製下列公有金鑰，並將其儲存至名為 的檔案amazon-ssm-agent.gpg。

> **Important**
>
> 下列公有金鑰將於 2026-07-152026 年 7 月 15 日）過期。Systems Manager 會在舊的公有金鑰過期之前，在本主題發佈新的公有金鑰。我們鼓勵您訂閱本主題的 RSS 摘要，以便在新的金鑰可用時收到通知。

```
-----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. 將公有金鑰匯入至您的 keyring，並記下傳回的鍵值。

```
gpg --import amazon-ssm-agent.gpg
```

3. 驗證指紋。請務必將 *key-value* 取代為上述步驟中的值。即使您使用 RPM 來驗證安裝程式套件，我們仍建議您使用 GPG 來驗證指紋。

```
gpg --fingerprint key-value
```

此命令會傳回類似以下的輸出：

```
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>
```

指紋應該符合下列項目。

```
4855 A9E6 8332 16D6 A77D 8FE4 51A8 E050 D005 2E5D
```

若指紋不相符，請勿安裝代理程式。聯絡 AWS 支援。

4. 如果您尚未下載簽章檔案，請根據執行個體的架構和作業系統進行下載。
5. 確認安裝程式套件簽章。請務必將 *signature-filename* 和 *agent-download-filename* 替換為您在下載簽章檔案和代理程式時指定的值 (已在本主題稍前部分的表格中列出)。

```
gpg --verify signature-filename agent-download-filename
```

例如，對於 Amazon Linux 2 上的 x86_64 架構：

```
gpg --verify amazon-ssm-agent.rpm.sig amazon-ssm-agent.rpm
```

此命令會傳回類似以下的輸出：

```
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
```

如果輸出包含 `BAD signature` 片語，請檢查您是否已正確執行程序。如果您繼續取得此回應，請聯絡 支援 ，不要安裝 代理程式。關於信任的警告訊息並不表示該簽章無效，只是您尚未驗證該公有金鑰。只有您或您信任者所簽章的金鑰才能信任。如果輸出包含 `Can't check signature: No public key` 片語，請確認是否已下載 SSM Agent 版本 3.1.1141.0 或更新版本。

RPM

在 Linux 伺服器上驗證SSM Agent套件 (v3.3.1802.0 及更新版本）

1. 複製下列公有金鑰，並將其儲存至名為 的檔案`amazon-ssm-agent.gpg`。

> **Important**
>
> 下列公有金鑰將於 2026-07-152026 年 7 月 15 日）過期。Systems Manager 會在舊的公有金鑰過期之前，在本主題發佈新的公有金鑰。我們鼓勵您訂閱本主題的 RSS 摘要，以便在新的金鑰可用時收到通知。

```
-----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. 將公有金鑰匯入至您的 keyring，並記下傳回的鍵值。

```
rpm --import amazon-ssm-agent.gpg
```

3. 驗證指紋。即使您使用 RPM 來驗證安裝程式套件，我們仍建議您使用 GPG 來驗證指紋。

```
rpm -qa gpg-pubkey --qf '%{Description}' | gpg --with-fingerprint | grep -A 1
 "ssm-agent-signer@amazon.com"
```

此命令會傳回類似以下的輸出：

```
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
```

指紋應該符合下列項目。

```
4855 A9E6 8332 16D6 A77D 8FE4 51A8 E050 D005 2E5D
```

若指紋不相符，請勿安裝代理程式。聯絡 AWS 支援。

4. 確認安裝程式套件簽章。請務必將 *agent-download-filename* 取代為您下載代理程式時指定的值，如本主題稍早的表格中所列。

```
rpm --checksig agent-download-filename
```

例如，對於 Amazon Linux 2 上的 x86_64 架構：

```
rpm --checksig amazon-ssm-agent.rpm
```

此命令會傳回類似以下的輸出。

```
amazon-ssm-agent.rpm: rsa sha1 md5 OK
```

如果輸出中缺失 pgp，且您已匯入公有金鑰，則不會簽署代理程式。如果輸出包含 NOT OK (MISSING KEYS: (MD5) *key-id*) 片語，請檢查您是否已正確執行程序，並確認是否已下載 SSM Agent 版本 3.1.1141.0 或更新版本。如果您繼續取得此回應，請聯絡 支援 ，不要安裝代理程式。

在 Linux 伺服器上驗證SSM Agent套件 (v3.3.1611.0 及更早版本)

開始之前

本節中的 GPG 和 RPM 程序適用於 3.3.1611.0 SSM Agent版和更早版本。我們建議一律使用最新版本的代理程式。如需相關資訊，請參閱在 Linux 伺服器上驗證SSM Agent套件 (v3.3.1802.0 及更新版本)。不過，如果您有特定原因可以繼續使用代理程式 3.3.1611.0 版或更早版本，請遵循下列其中一個程序的指示。

GPG

在 Linux 伺服器上驗證SSM Agent套件 (v3.3.1611.0 及更早版本）

1. 複製下列公有金鑰，並將其儲存至名為 的檔案amazon-ssm-agent.gpg。

> **Important**
>
> 以下顯示的公有金鑰已於 2025-02-172025 年 2 月 17 日）過期，並適用於 3.3.1611.0 版和 3.2.1542.0 之前的版本，且僅適用於先前用於驗證代理程式的簽章。Systems Manager 會在舊的公有金鑰過期之前，在本主題發佈新的公有金鑰。我們鼓勵您訂閱本主題的 RSS 摘要，以便在新的金鑰可用時收到通知。

```
-----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. 將公有金鑰匯入至您的 keyring，並記下傳回的鍵值。

```
gpg --import amazon-ssm-agent.gpg
```

3. 驗證指紋。請務必將 *key-value* 取代為上述步驟中的值。即使您使用 RPM 來驗證安裝程式套件，我們仍建議您使用 GPG 來驗證指紋。

```
gpg --fingerprint key-value
```

此命令會傳回類似以下的輸出。

```
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>
```

指紋應該符合下列項目。

```
DE92 C7DA 3E56 E923 31D6 2A36 BC1F 495C 97DD 04ED
```

若指紋不相符，請勿安裝代理程式。聯絡 AWS 支援。

4. 如果您尚未下載簽章檔案，請根據執行個體的架構和作業系統進行下載。
5. 確認安裝程式套件簽章。請務必將 *signature-filename* 和 *agent-download-filename* 替換為您在下載簽章檔案和代理程式時指定的值 (已在本主題稍前部分的表格中列出)。

```
gpg --verify signature-filename agent-download-filename
```

例如，對於 Amazon Linux 2 上的 x86_64 架構：

```
gpg --verify amazon-ssm-agent.rpm.sig amazon-ssm-agent.rpm
```

此命令會傳回類似以下的輸出：

```
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
```

如果輸出包含 BAD signature 片語，請檢查您是否已正確執行程序。如果您繼續取得此回應，請聯絡 支援 ，不要安裝 代理程式。關於信任的警告訊息並不表示該簽章無效，只是您尚未驗證該公有金鑰。只有您或您信任者所簽章的金鑰才能信任。如果輸出包含 Can't check signature: No public key 片語，請確認是否已下載 SSM Agent 版本 3.1.1141.0 或更新版本。

RPM

在 Linux 伺服器上驗證SSM Agent套件 (v3.3.1611.0 及更早版本)

1. 複製下列公有金鑰，並將其儲存至名為 的檔案amazon-ssm-agent.gpg。

> **Important**
>
> 以下顯示的公有金鑰已於 2025-02-172025 年 2 月 17 日 ） 過期，並適用於 3.3.1611.0 版和 3.2.1542.0 之前的版本，且僅適用於先前用於驗證代理程式的簽章。Systems Manager 會在舊的公有金鑰過期之前，在本主題發佈新的公有金鑰。我們鼓勵您訂閱本主題的 RSS 摘要，以便在新的金鑰可用時收到通知。

```
-----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. 將公有金鑰匯入至您的 keyring，並記下傳回的鍵值。

```
rpm --import amazon-ssm-agent.gpg
```

3. 驗證指紋。即使您使用 RPM 來驗證安裝程式套件，我們仍建議您使用 GPG 來驗證指紋。

```
rpm -qa gpg-pubkey --qf '%{Description}' | gpg --with-fingerprint | grep -A 1
 "ssm-agent-signer@amazon.com"
```

此命令會傳回類似以下的輸出：

```
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
```

指紋應該符合下列項目。

```
DE92 C7DA 3E56 E923 31D6 2A36 BC1F 495C 97DD 04ED
```

若指紋不相符，請勿安裝代理程式。聯絡 AWS 支援。

4. 確認安裝程式套件簽章。請務必將 *agent-download-filename* 取代為您下載代理程式時指定的值，如本主題稍早的表格中所列。

```
rpm --checksig agent-download-filename
```

例如，對於 Amazon Linux 2 上的 x86_64 架構：

```
rpm --checksig amazon-ssm-agent.rpm
```

此命令會傳回類似以下的輸出。

```
amazon-ssm-agent.rpm: rsa sha1 md5 OK
```

如果輸出中缺失 pgp，且您已匯入公有金鑰，則不會簽署代理程式。如果輸出包含 NOT OK (MISSING KEYS: (MD5) *key-id*) 片語，請檢查您是否已正確執行程序，並確認是否已下載 SSM Agent 版本 3.1.1141.0 或更新版本。如果您繼續取得此回應，請聯絡 支援 ，不要安裝代理程式。

在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent

在 Amazon Elastic Compute Cloud (Amazon EC2SSM Agent) Linux 作業系統上手動安裝 AWS Systems Manager Agent () 之前，請檢閱下列資訊。

SSM Agent 安裝檔 URL

您可以存取存放在任何商業中的 SSM Agent的安裝檔案 AWS 區域。我們也在 Amazon Simple Storage Service (Amazon S3) 儲存貯體中提供全域可用的安裝檔案，可用作替代方案或檔案的備份來源。

如果您在一個或兩個執行個體手動安裝代理程式，您可以使用我們所提供快速安裝程序的命令來節省時間。這些程序中提供的命令也可以透過使用者資料以指令碼的形式傳遞至 Amazon EC2 執行個體。

如果您為了在多個執行個體安裝代理程式而建立指令碼或範本，建議使用您所在地理位置的 AWS 區域之內或附近的安裝檔案。針對大批安裝，這可以提高下載速度並減少延遲。在這些情況下，建議使用安裝主題的建立自訂安裝命令程序。

預先安裝代理程式的 Amazon Machine Images

SSM Agent 預先安裝在 提供的一些 Amazon Machine Images(AMIs) 上 AWS。如需相關資訊，請參閱尋找預先安裝了 SSM Agent的 AMIs。

其他機器類型的安裝

如果您需要在內部部署伺服器或虛擬機器 (VM) 上安裝代理程式，以便與 Systems Manager 搭配使用，請參閱如何在混合 Linux 節點上安裝 SSM Agent。如需詳細資訊了解在邊緣裝置安裝代理程式，請參閱 使用 Systems Manager 管理邊緣裝置。

保持代理程式的最新狀態

每當將新工具新增至 Systems Manager 或對現有工具進行更新時，SSM Agent就會發行更新的 版本。未使用最新版本的代理程式可能會使您的受管節點無法使用各種 Systems Manager 工具和功能。因此，我們建議您讓機器的 SSM Agent 自動保持最新狀態。如需相關資訊，請參閱 自動化 SSM Agent 更新。請訂閱 GitHub 上的 SSM Agent 版本備註頁面，以便接收有關 SSM Agent 更新的通知。

選擇您的作業系統

若要檢視在指定作業系統手動安裝 SSM Agent 的程序，請選擇以下列表的任一連結：

> **Note**
>
> 如需下列每個作業系統的支援版本清單，請參閱Systems Manager 支援的作業系統。

- 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

從 Linux 執行個體解除安裝 SSM Agent

使用作業系統的套件管理員從 Linux 執行個體解除安裝 SSM Agent。視作業系統而定，解除安裝命令類似於下列範例命令：

```
sudo dpkg -r amazon-ssm-agent
```

在 AlmaLinux 執行個體上手動安裝 SSM Agent

此章節的資訊可協助您在 AlmaLinux 執行個體手動安裝或重新安裝 SSM Agent。

開始之前

在 AlmaLinux 執行個體上安裝 SSM Agent 之前，請注意下列事項：

- 請確保您的 AlmaLinux 執行個體已安裝 Python 3。這是必要的，如此 SSM Agent 才能正常運作。
- 關於在所有 Linux 作業系統安裝 SSM Agent 的重要資訊，請參閱在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent。

主題

- 適用於 AlmaLinux 的 SSM Agent 之快速安裝命令
- 在您所在區域建立適用於 AlmaLinux 的自訂代理程式安裝命令

適用於 AlmaLinux 的 SSM Agent 之快速安裝命令

使用以下步驟手動安裝 SSM Agent 在單一執行個體。此程序使用全域可用的安裝檔案。

開始之前

在 AlmaLinux 執行個體上安裝 SSM Agent 之前，請注意下列事項：

- 請確保您的 AlmaLinux 執行個體已安裝 Python 3。這是必要的，如此 SSM Agent 才能正常運作。

在 AlmaLinux 上安裝 SSM Agent

1. 使用您偏好的方式 (如 SSH) 連線至您的 AlmaLinux 執行個體。
2. 複製適用於執行個體架構的命令，並在執行個體上執行該命令。

> **Note**
>
> 雖然以下命令的 URL 包含 `ec2-downloads-windows` 目錄，但這些是適用於 AlmaLinux 的正確全域安裝檔案。

x86_64 執行個體

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

ARM64 執行個體

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. (建議) 若要驗證該代理程式是否在執行，請使用以下命令。

```
sudo systemctl status amazon-ssm-agent
```

在大部分情況下，命令會報告代理程式正在執行，如下列範例所示。

```
# 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--
```

在極少數情況下，命令會報告代理程式已安裝但未執行，如下列範例所示。

```
# 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--
```

若要在這些情況下啟用代理程式，請執行下列命令。

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

在您所在區域建立適用於 AlmaLinux 的自訂代理程式安裝命令

當利用指令碼或範本在多個執行個體上安裝 SSM Agent 時，建議使用存放在所用 AWS 區域 的安裝檔案。

針對以下命令，我們所提供的範例使用美國東部 (俄亥俄) 區域 (us-east-2) 可公開存取的 S3 儲存貯體。

> **Tip**
>
> 此主題前文所提及 適用於 AlmaLinux 的 SSM Agent 之快速安裝命令 程序的全域 URL 也可取代為您建構的自定區域 URL。

在下列命令中，用您自己的資訊取代*##*。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

x86_64

```
sudo dnf install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

請參閱以下範例。

```
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
```

請參閱以下範例。

```
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 執行個體手動安裝 SSM Agent

> **Important**
>
> 本主題提供在 Amazon Linux 2 和 Amazon Linux 2023 執行個體上使用 SSM Agent 的命令。Amazon Linux 1 執行個體不支援部分這些命令。在繼續之前，請確定您正在檢視執行個體類型的正確主題。如需在 Amazon Linux 1 執行個體上執行的命令，請參閱在 Amazon Linux 1 執行個體上手動安裝 SSM Agent。

在大多數情況下， 提供的 Amazon Linux 2 和 Amazon Linux 2023 的 Amazon Machine Images(AMIs) AWS 會隨附預設預先安裝的 AWS Systems Manager 代理程式 (SSM Agent)。如需詳細資訊，請參閱尋找預先安裝了 SSM Agent的 AMIs。

如果 SSM Agent 未預先安裝在新的 Amazon Linux 2 或 Amazon Linux 2023 執行個體，或者如果您需要手動重新安裝代理程式，請使用此頁面的資訊為您提供協助。

開始之前

在 Amazon Linux 2 或 Amazon Linux 2023 執行個體上安裝 SSM Agent 之前，請注意下列事項：

- 關於在所有 Linux 作業系統安裝 SSM Agent 的重要資訊，請參閱在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent。
- 在使用 SSM 文件 `AWS-UpdateSSMAgent` 安裝或更新代理程式之後，如果您使用 `yum` 命令更新受管節點上的 SSM Agent，您可能會看到如下訊息：「Warning: RPMDB altered outside of yum. (警告：RPMDB 已變更超出 yum)。」 預期會出現此訊息，且可以安全忽略。

主題

- 適用於 Amazon Linux 2 或 Amazon Linux 2023 的 SSM Agent 之快速安裝命令
- 在您所在區域建立適用於 Amazon Linux 2 或 Amazon Linux 2023 的自訂代理程式安裝命令

適用於 Amazon Linux 2 或 Amazon Linux 2023 的 SSM Agent 之快速安裝命令

使用以下步驟手動安裝 SSM Agent 在單一執行個體。此程序使用全域可用的安裝檔案。

使用快速複製及貼上命令在 Amazon Linux 2 或 Amazon Linux 2023 安裝 SSM Agent

1. 使用您偏好的方式 (如 SSH) 連線至您的 Amazon Linux 2 或 Amazon Linux 2023 執行個體。

2. 複製適用於執行個體架構的命令，並在執行個體上執行該命令。

> **Note**
>
> 雖然以下命令的 URL 包含 `ec2-downloads-windows` 目錄，但這些是適用於 Amazon Linux 2 和 Amazon Linux 2023 的正確全域安裝檔案。

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. (建議) 若要驗證該代理程式是否在執行，請使用以下命令。

```
sudo systemctl status amazon-ssm-agent
```

在大部分情況下，命令會報告代理程式正在執行，如下列範例所示。

```
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--
```

在極少數情況下，命令會報告代理程式已安裝但未執行，如下列範例所示。

```
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--
```

若要在這些情況下啟用代理程式，請執行下列命令。

```
sudo systemctl start amazon-ssm-agent
```

在您所在區域建立適用於 Amazon Linux 2 或 Amazon Linux 2023 的自訂代理程式安裝命令

當利用指令碼或範本在多個執行個體上安裝 SSM Agent 時，建議使用存放在所用 AWS 區域 的安裝檔案。

針對以下命令，我們所提供的範例使用美國東部 (俄亥俄) 區域 (us-east-2) 可公開存取的 S3 儲存貯體。

> **Tip**
>
> 此主題前文所提及 適用於 Amazon Linux 1 的 SSM Agent 之快速安裝命令 程序的全域 URL 也可取代為您建構的自定區域 URL。

在下列命令中，用您自己的資訊取代*##*。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

x86_64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

請參閱以下範例。

```
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
```

請參閱以下範例。

```
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 執行個體上手動安裝 SSM Agent

> **Important**
>
> Amazon Linux 1 已於 2020 年 12 月 31 日達到標準支援結束，並於 2023 年 12 月 31 日達到生命週期結束，如AWS 新聞部落格上的 Amazon Linux 生命週期AMI結束更新 end-of-life中所宣布。 AWS 不再為此作業系統提供 Amazon Machine Images(AMIs)。不過， AWS Systems Manager 繼續為現有的 Amazon Linux 1 執行個體提供支援。
> 本主題提供在 Amazon Linux 1 執行個體上使用 SSM Agent 的命令。Amazon Linux 2 和 Amazon Linux 2023 執行個體不支援部分命令。繼續之前，請確認您正在檢視的主題符合您的執行個體類型。如需在 Amazon Linux 2 或 Amazon Linux 2023 執行個體上執行的命令，請參閱 在 Amazon Linux 2 和 Amazon Linux 2023 執行個體手動安裝 SSM Agent。

在大多數情況下，由 提供的 Amazon Linux 1 Amazon Machine Images(AMIs) AWS 隨附預設預先安裝的 AWS Systems Manager 代理程式 (SSM Agent)。如需詳細資訊，請參閱 尋找預先安裝了 SSM Agent的 AMIs。

如果您需要在 Amazon Linux 1 上手動重新安裝代理程式，請使用此頁面的資訊為您提供協助。

開始之前

在 Amazon Linux 1 執行個體上安裝 SSM Agent 之前，請注意下列事項：

- 關於在所有 Linux 作業系統安裝 SSM Agent 的重要資訊，請參閱在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent。
- 在使用 SSM 文件 `AWS-UpdateSSMAgent` 安裝或更新代理程式之後，如果您使用 `yum` 命令更新受管節點上的 SSM Agent，您可能會看到如下訊息：「Warning: RPMDB altered outside of yum. (警告：RPMDB 已變更超出 yum)。」 預期會出現此訊息，且可以安全忽略。

主題

- 適用於 Amazon Linux 1 的 SSM Agent 之快速安裝命令
- 在您所在區域建立適用於 Amazon Linux 1 的自訂代理程式安裝命令

適用於 Amazon Linux 1 的 SSM Agent 之快速安裝命令

使用以下步驟手動安裝 SSM Agent 在單一執行個體。此程序使用全域可用的安裝檔案。

使用快速複製及貼上命令在 Amazon Linux 1 上安裝 SSM Agent

1. 使用您偏好的方式 (如 SSH) 連線至您的 Amazon Linux 1 執行個體。
2. 複製適用於執行個體架構的命令，並在執行個體上執行該命令。

> **Note**
>
> 雖然以下命令的 URL 包含 `ec2-downloads-windows` 目錄，但這些是適用於 Amazon Linux 1 的正確全域安裝檔案。

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. (建議) 針對您的執行個體架構執行命令，驗證代理程式正在執行。

x86_64 和 x86

```
sudo status amazon-ssm-agent
```

ARM64

```
sudo systemctl status amazon-ssm-agent
```

在大部分情況下，命令會報告代理程式正在執行，如下列範例所示。

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--
```

在極少數情況下，命令會報告代理程式已安裝但未執行，如以下範例所示。

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--
```

若要在這些情況下啟用代理程式，請執行適用於您執行個體架構的命令。

x86_64 和 x86

```
sudo start amazon-ssm-agent
```

ARM64

```
sudo systemctl start amazon-ssm-agent
```

在您所在區域建立適用於 Amazon Linux 1 的自訂代理程式安裝命令

當利用指令碼或範本在多個執行個體上安裝 SSM Agent 時，建議使用存放在所用 AWS 區域 的安裝檔案。

針對以下命令，我們所提供的範例使用美國東部 (俄亥俄) 區域 (us-east-2) 可公開存取的 S3 儲存貯體。

> **Tip**
>
> 此主題前文所提及 適用於 Amazon Linux 1 的 SSM Agent 之快速安裝命令 程序的全域 URL 也可取代為您建構的自定區域 URL。

在下列命令中，用您自己的資訊取代*##*。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

x86_64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

請參閱以下範例。

```
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
```

請參閱以下範例。

```
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
```

請參閱以下範例。

```
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 執行個體手動安裝 SSM Agent

提供的 CentOS 的 Amazon Machine Images(AMIs) AWS 不會隨附預設預先安裝的 AWS Systems Manager 代理程式 (SSM Agent)。關於可能已預先安裝代理程式且由 AWS 管理的 AMIs，如需清單，請參閱 尋找預先安裝了 SSM Agent的 AMIs。

此章節的資訊可協助您在 CentOS 執行個體手動安裝或重新安裝 SSM Agent。

開始之前

在 CentOS 執行個體上安裝 SSM Agent 之前，請注意下列事項：

- 關於在所有 Linux 作業系統安裝 SSM Agent 的重要資訊，請參閱在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent。
- 在使用 SSM 文件 `AWS-UpdateSSMAgent` 安裝或更新代理程式之後，如果您使用 `yum` 命令更新受管節點上的 SSM Agent，您可能會看到如下訊息：「Warning: RPMDB altered outside of yum. (警告：RPMDB 已變更超出 yum)。」 預期會出現此訊息，且可以安全忽略。

主題

- 在 CentOS 8.x 上安裝 SSM Agent
- 在 CentOS 7.x 安裝 SSM Agent
- 在 CentOS 6.x 安裝 SSM Agent

在 CentOS 8.x 上安裝 SSM Agent

提供的 CentOS 8 Amazon Machine Images(AMIs) AWS 不會隨附預設預先安裝的 AWS Systems Manager 代理程式 (SSM Agent)。請使用此頁面的資訊協助您在 CentOS 8 執行個體上安裝或重新安裝代理程式。

開始之前

在 CentOS 8 執行個體上安裝 SSM Agent 之前，請注意下列事項：

- 確保您的 CentOS 8 執行個體上已安裝 Python 2 或 Python 3。這是必要的，如此 SSM Agent 才能正常運作。

主題

- 適用於 CentOS 8 的 SSM Agent 之快速安裝命令
- 在您所在區域建立適用於 CentOS 8 的自訂代理程式安裝命令

適用於 CentOS 8 的 SSM Agent 之快速安裝命令

使用以下步驟手動安裝 SSM Agent 在單一執行個體。此程序使用全域可用的安裝檔案。

在 CentOS 8.x 上安裝 SSM Agent

1. 使用您偏好的方式 (如 SSH) 連線至您的 CentOS 8 執行個體。
2. 複製適用於執行個體架構的命令，並在執行個體上執行該命令。

> **Note**
>
> 雖然以下命令的 URL 包含 `ec2-downloads-windows` 目錄，但這些是適用於 CentOS 8 的正確全域安裝檔案。

x86_64 執行個體

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

ARM64 執行個體

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. (建議) 若要驗證該代理程式是否在執行，請使用以下命令。

```
sudo systemctl status amazon-ssm-agent
```

在大部分情況下，命令會報告代理程式正在執行，如下列範例所示。

```
# 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--
```

在極少數情況下，命令會報告代理程式已安裝但未執行，如下列範例所示。

```
# amazon-ssm-agent.service - amazon-ssm-agent
   Loaded: loaded (/etc/systemd/system/amazon-ssm-agent.service; disabled; vend>
   Active: inactive (dead)
            --truncated--
```

若要在這些情況下啟用代理程式，請執行下列命令。

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

在您所在區域建立適用於 CentOS 8 的自訂代理程式安裝命令

當利用指令碼或範本在多個執行個體上安裝 SSM Agent 時，建議使用存放在所用 AWS 區域 的安裝檔案。

針對以下命令，我們所提供的範例使用美國東部 (俄亥俄) 區域 (us-east-2) 可公開存取的 S3 儲存貯體。

> **Tip**
>
> 此主題前文所提及 適用於 CentOS 8 的 SSM Agent 之快速安裝命令 程序的全域 URL 也可取代為您建構的自定區域 URL。

在下列命令中，用您自己的資訊取代*##*。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

x86_64

```
sudo dnf install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

請參閱以下範例。

```
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
```

請參閱以下範例。

```
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 安裝 SSM Agent

提供的 CentOS 7 Amazon Machine Images(AMIs) AWS 不會隨附預設預先安裝的 AWS Systems Manager 代理程式 (SSM Agent)。請使用此頁面的資訊協助您在 CentOS 7 執行個體上安裝或重新安裝代理程式。

主題

- 適用於 CentOS 7 的 SSM Agent 之快速安裝命令
- 在您所在區域建立適用於 CentOS 7 的自訂代理程式安裝命令

適用於 CentOS 7 的 SSM Agent 之快速安裝命令

使用以下步驟手動安裝 SSM Agent 在單一執行個體。此程序使用全域可用的安裝檔案。

在 CentOS 7.x 上安裝 SSM Agent

1. 使用您偏好的方式 (如 SSH) 連線至您的 CentOS 7 執行個體。
2. 複製適用於執行個體架構的命令，並在執行個體上執行該命令。

> **Note**
>
> 雖然以下命令的 URL 包含 `ec2-downloads-windows` 目錄，但這些是適用於 CentOS 7 的正確全域安裝檔案。

x86_64 執行個體

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

ARM64 執行個體

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. (建議) 若要驗證該代理程式是否在執行，請使用以下命令。

```
sudo systemctl status amazon-ssm-agent
```

在大部分情況下，命令會報告代理程式正在執行，如下列範例所示。

```
# 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--
```

在極少數情況下，命令會報告代理程式已安裝但未執行，如下列範例所示。

```
# 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--
```

若要在這些情況下啟用代理程式，請執行下列命令。

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

在您所在區域建立適用於 CentOS 7 的自訂代理程式安裝命令

當利用指令碼或範本在多個執行個體上安裝 SSM Agent 時，建議使用存放在所用 AWS 區域 的安裝檔案。

針對以下命令，我們所提供的範例使用美國東部 (俄亥俄) 區域 (us-east-2) 可公開存取的 S3 儲存貯體。

> **Tip**
>
> 此主題前文所提及 適用於 CentOS 7 的 SSM Agent 之快速安裝命令 程序的全域 URL 也可取代為您建構的自定區域 URL。

在下列命令中，用您自己的資訊取代*##*。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

x86_64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

請參閱以下範例。

```
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
```

請參閱以下範例。

```
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 安裝 SSM Agent

提供的 CentOS 6 Amazon Machine Images(AMIs) AWS 不會隨附預設預先安裝的 AWS Systems Manager 代理程式 (SSM Agent)。請使用此頁面的資訊協助您在 CentOS 6 執行個體上安裝或重新安裝代理程式。

主題

- 適用於 CentOS 6 的 SSM Agent 之快速安裝命令
- 在您所在區域建立適用於 CentOS 6 的自訂代理程式安裝命令

適用於 CentOS 6 的 SSM Agent 之快速安裝命令

使用以下步驟手動安裝 SSM Agent 在單一執行個體。此程序使用全域可用的安裝檔案。

在 CentOS 6.x 上安裝 SSM Agent

1. 使用您偏好的方式 (如 SSH) 連線至您的 CentOS 6 執行個體。
2. 複製適用於執行個體架構的命令，並在執行個體上執行該命令。

> **Note**
>
> 雖然以下命令的 URL 包含 `ec2-downloads-windows` 目錄，但這些是適用於 CentOS 6 的正確全域安裝檔案。
> 以下命令指定目錄 `3.0.1479.0` 版本而不是目錄 `latest`。這是因為 CentOS 6 不支援 SSM Agent 3.1 版及更新版本。

x86_64 執行個體

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/
SSMAgent/3.0.1479.0/linux_amd64/amazon-ssm-agent.rpm
```

x86 執行個體

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/
SSMAgent/3.0.1479.0/linux_386/amazon-ssm-agent.rpm
```

3. (建議) 若要驗證該代理程式是否在執行，請使用以下命令。

```
sudo status amazon-ssm-agent
```

在大部分情況下，命令會報告代理程式正在執行，如下列範例所示。

```
amazon-ssm-agent start/running, process 1744
```

在極少數情況下，命令會報告代理程式已安裝但未執行，如下列範例所示。

```
amazon-ssm-agent stop/waiting
```

若要在這些情況下啟用代理程式，請執行下列命令。

```
sudo start amazon-ssm-agent
```

在您所在區域建立適用於 CentOS 6 的自訂代理程式安裝命令

當利用指令碼或範本在多個執行個體上安裝 SSM Agent 時，建議使用存放在所用 AWS 區域 的安裝檔案。

針對以下命令，我們所提供的範例使用美國東部 (俄亥俄) 區域 (us-east-2) 可公開存取的 S3 儲存貯體。

> **Tip**
>
> 此主題前文所提及 適用於 CentOS 6 的 SSM Agent 之快速安裝命令 程序的全域 URL 也可取代為您建構的自定區域 URL。

在下列命令中，用您自己的資訊取代*##*。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

> **Note**
>
> 以下命令指定目錄 `3.0.1479.0` 版本而不是目錄 `latest`。這是因為 CentOS 6 不支援 SSM Agent 3.1 版及更新版本。

x86_64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/3.0.1479.0/
linux_amd64/amazon-ssm-agent.rpm
```

請參閱以下範例。

```
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
```

請參閱以下範例。

```
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
```

手動在 CentOS Stream 執行個體上安裝 SSM Agent

CentOS Stream 提供的 Amazon Machine Images(AMIs) AWS 不會隨附預設預先安裝的 AWS Systems Manager 代理程式 (SSM Agent)。關於可能已預先安裝代理程式且由 AWS 管理的 AMIs，如需清單，請參閱 尋找預先安裝了 SSM Agent的 AMIs。

此章節的資訊可協助您在 CentOS Stream 執行個體手動安裝或重新安裝 SSM Agent。

開始之前

在 CentOS Stream 執行個體上安裝 SSM Agent 之前，請注意下列事項：

- 關於在所有 Linux 作業系統安裝 SSM Agent 的重要資訊，請參閱在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent。

主題

- 適用於 CentOS Stream 的 SSM Agent 之快速安裝命令
- 在您所在區域建立適用於 CentOS Stream 的自訂代理程式安裝命令

適用於 CentOS Stream 的 SSM Agent 之快速安裝命令

使用以下步驟手動安裝 SSM Agent 在單一執行個體。此程序使用全域可用的安裝檔案。

開始之前

在 CentOS Stream 執行個體上安裝 SSM Agent 之前，請注意下列事項：

- 請確保您的 CentOS Stream 8 執行個體已安裝 Python 2 或 Python 3。這是必要的，如此 SSM Agent 才能正常運作。

在 CentOS Stream 上安裝 SSM Agent

1. 使用您偏好的方式 (如 SSH) 連線至您的 CentOS Stream 執行個體。
2. 複製適用於執行個體架構的命令，並在執行個體上執行該命令。

> **Note**
>
> 雖然以下命令的 URL 包含 `ec2-downloads-windows` 目錄，但這些是適用於 CentOS Stream 的正確全域安裝檔案。

x86_64 執行個體

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

ARM64 執行個體

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. (建議) 若要驗證該代理程式是否在執行，請使用以下命令。

```
sudo systemctl status amazon-ssm-agent
```

在大部分情況下，命令會報告代理程式正在執行，如下列範例所示。

```
# 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--
```

在極少數情況下，命令會報告代理程式已安裝但未執行，如下列範例所示。

```
# 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--
```

若要在這些情況下啟用代理程式，請執行下列命令。

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

在您所在區域建立適用於 CentOS Stream 的自訂代理程式安裝命令

當利用指令碼或範本在多個執行個體上安裝 SSM Agent 時，建議使用存放在所用 AWS 區域 的安裝檔案。

針對以下命令，我們所提供的範例使用美國東部 (俄亥俄) 區域 (us-east-2) 可公開存取的 S3 儲存貯體。

> **Tip**
>
> 此主題前文所提及 適用於 CentOS Stream 的 SSM Agent 之快速安裝命令 程序的全域 URL 也可取代為您建構的自定區域 URL。

在下列命令中，用您自己的資訊取代*##*。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

x86_64

```
sudo dnf install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

請參閱以下範例。

```
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
```

請參閱以下範例。

```
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 執行個體手動安裝 SSM Agent

Debian Server 提供的 Amazon Machine Images(AMIs) AWS 不會隨附預設預先安裝的 AWS Systems Manager 代理程式 (SSM Agent)。關於可能已預先安裝代理程式且由 AWS 管理的 AMIs，如需清單，請參閱 尋找預先安裝了 SSM Agent的 AMIs。

此章節的資訊可協助您在 Debian Server 執行個體手動安裝或重新安裝 SSM Agent。

開始之前

在 Debian Server 執行個體上安裝 SSM Agent 之前，請注意下列事項：

- 關於在所有 Linux 作業系統安裝 SSM Agent 的重要資訊，請參閱在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent。

主題

- 適用於 Debian Server 的 SSM Agent 之快速安裝命令
- 在您所在區域建立適用於 Debian Server 的自訂代理程式安裝命令

適用於 Debian Server 的 SSM Agent 之快速安裝命令

使用以下步驟手動安裝 SSM Agent 在單一執行個體。此程序使用全域可用的安裝檔案。

在 Debian Server 上安裝 SSM Agent

1. 使用您偏好的方式 (如 SSH) 連線至您的 Debian Server 執行個體。
2. 執行以下命令，可在執行個體建立暫時目錄。

```
mkdir /tmp/ssm
```

3. 執行以下命令，可切換至暫時目錄。

```
cd /tmp/ssm
```

4. 複製適用於執行個體架構的命令，並在執行個體上執行該命令。

> **Note**
>
> 雖然以下命令的 URL 包含 `ec2-downloads-windows` 目錄，但這些是適用於 Debian Server 的正確全域安裝檔案。

Debian Server 8 僅支援 x86_64 架構。

x86_64 執行個體

```
wget https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/
debian_amd64/amazon-ssm-agent.deb
```

ARM64 執行個體

```
wget https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/
debian_arm64/amazon-ssm-agent.deb
```

5. 執行下列命令。

```
sudo dpkg -i amazon-ssm-agent.deb
```

6. (建議) 若要驗證該代理程式是否在執行，請使用以下命令。

```
sudo systemctl status amazon-ssm-agent
```

在大部分情況下，命令會報告代理程式正在執行，如下列範例所示。

```
# 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--
```

在極少數情況下，命令會報告代理程式已安裝但未執行，如下列範例所示。

```
# 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--
```

若要在這些情況下啟用代理程式，請執行下列命令。

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

在您所在區域建立適用於 Debian Server 的自訂代理程式安裝命令

當利用指令碼或範本在多個執行個體上安裝 SSM Agent 時，建議使用存放在所用 AWS 區域 的安裝檔案。

針對以下命令，我們所提供的範例使用美國東部 (俄亥俄) 區域 (us-east-2) 可公開存取的 S3 儲存貯體。

> **Tip**
>
> 此主題前文所提及 適用於 Debian Server 的 SSM Agent 之快速安裝命令 程序的全域 URL 也可取代為您建構的自定區域 URL。

在下列命令中，用您自己的資訊取代*##*。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域欄位。

> **Note**
>
> Debian Server 8 僅支援 x86_64 架構。

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
```

請參閱以下範例。

```
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
```

請參閱以下範例。

```
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 執行個體手動安裝 SSM Agent

Oracle Linux 提供的 Amazon Machine Images(AMIs) AWS 不會隨附預設預先安裝的 AWS Systems Manager 代理程式 (SSM Agent)。關於可能已預先安裝代理程式且由 AWS 管理的 AMIs，如需清單，請參閱 尋找預先安裝了 SSM Agent的 AMIs。

此章節的資訊可協助您在 Oracle Linux 執行個體手動安裝或重新安裝 SSM Agent。

開始之前

在 Oracle Linux 執行個體上安裝 SSM Agent 之前，請注意下列事項：

- 關於在所有 Linux 作業系統安裝 SSM Agent 的重要資訊，請參閱在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent。
- 在使用 SSM 文件 `AWS-UpdateSSMAgent` 安裝或更新代理程式之後，如果您使用 `yum` 命令更新受管節點上的 SSM Agent，您可能會看到如下訊息：「Warning: RPMDB altered outside of yum. (警告：RPMDB 已變更超出 yum)。」 預期會出現此訊息，且可以安全忽略。

主題

- 適用於 Oracle Linux 的 SSM Agent 之快速安裝命令
- 在您所在區域建立適用於 Oracle Linux 的自訂代理程式安裝命令

適用於 Oracle Linux 的 SSM Agent 之快速安裝命令

使用以下步驟手動安裝 SSM Agent 在單一執行個體。此程序使用全域可用的安裝檔案。

使用快速複製及貼上命令在 Oracle Linux 安裝 SSM Agent

1. 使用您偏好的方式 (如 SSH) 連線至您的 Oracle Linux 執行個體。
2. 複製下列命令並在執行個體執行。

   > **Note**
   >
   > 雖然以下命令的 URL 包含 `ec2-downloads-windows` 目錄，但這些是適用於 Oracle Linux 的正確全域安裝檔案。

   x86_64

   ```
   sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
   latest/linux_amd64/amazon-ssm-agent.rpm
   ```

3. (建議) 若要驗證該代理程式是否在執行，請使用以下命令。

   ```
   sudo systemctl status amazon-ssm-agent
   ```

   在大部分情況下，命令會報告代理程式正在執行，如下列範例所示。

   ```
   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--
   ```

   在極少數情況下，命令會報告代理程式已安裝但未執行，如下列範例所示。

   ```
   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--
```

若要在這些情況下啟用代理程式，請執行下列命令。

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

在您所在區域建立適用於 Oracle Linux 的自訂代理程式安裝命令

當利用指令碼或範本在多個執行個體上安裝 SSM Agent 時，建議使用存放在所用 AWS 區域 的安裝檔案。

針對以下命令，我們所提供的範例使用美國東部 (俄亥俄) 區域 (us-east-2) 可公開存取的 S3 儲存貯體。

> **Tip**
>
> 此主題前文所提及 適用於 Oracle Linux 的 SSM Agent 之快速安裝命令 程序的全域 URL 也可取代為您建構的自定區域 URL。

在下列命令中，用您自己的資訊取代*##*。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

x86_64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

請參閱以下範例。

```
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 執行個體手動安裝 SSM Agent

提供的 AWS Red Hat Enterprise Linux(AMIs) 的 Amazon Machine Images(RHEL) 不會隨附預設預先安裝的 AWS Systems Manager 代理程式 (SSM Agent)。如需可能預先安裝代理程式AMIs的 AWS 受管清單，請參閱 尋找預先安裝了 SSM Agent的 AMIs。

此章節的資訊可協助您在 RHEL 執行個體手動安裝或重新安裝 SSM Agent。

開始之前

在 RHEL 執行個體上安裝 SSM Agent 之前，請注意下列事項：

- 關於在所有 Linux 作業系統安裝 SSM Agent 的重要資訊，請參閱在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent。
- 在使用 SSM 文件 `AWS-UpdateSSMAgent` 安裝或更新代理程式之後，如果您使用 `yum` 命令更新受管節點上的 SSM Agent，您可能會看到如下訊息：「Warning: RPMDB altered outside of yum. (警告：RPMDB 已變更超出 yum)。」 預期會出現此訊息，且可以安全忽略。

主題

- 在 RHEL 8.x、9.x 和 10.x SSM Agent上安裝
- 在 RHEL 7.x 安裝 SSM Agent
- 在 RHEL 6.x 安裝 SSM Agent

在 RHEL 8.x、9.x 和 10.x SSM Agent上安裝

提供的 RHEL AWS 8 和 9 的 Amazon Machine Images(AMIs) 不會隨附預設預先安裝的 AWS Systems Manager 代理程式 (SSM Agent)。請使用此頁面的資訊協助您在 RHEL 8 和 9 執行個體上安裝或重新安裝代理程式。

開始之前

在 RHEL 8、9 或 10 執行個體SSM Agent上安裝 之前，請注意下列事項：

- 請確定您的 8、9 或 10 執行個體已安裝 Python RHEL 2 或 Python 3。這是必要的，如此 SSM Agent 才能正常運作。

主題

- 8、9 和 10 RHELSSM Agent上的 快速安裝命令

- 在您區域中為 RHEL 8、9 和 10 建立自訂代理程式安裝命令

8、9 和 10 RHELSSM Agent上的 快速安裝命令

使用以下步驟手動安裝 SSM Agent 在單一執行個體。此程序使用全域可用的安裝檔案。

在 RHEL 8.x、9.x 或 10.x SSM Agent上安裝

1. 使用您偏好的方法連線至 RHEL 8、9 或 10 執行個體，例如 SSH。
2. 複製適用於執行個體架構的命令，並在執行個體上執行該命令。

> **Note**
>
> 即使下列命令中的 URLs包含 `ec2-downloads-windows`目錄，這些是RHEL適用於 8、9 和 10 的正確全域安裝檔案。

x86_64 執行個體

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

ARM64 執行個體

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. (建議) 若要驗證該代理程式是否在執行，請使用以下命令。

```
sudo systemctl status amazon-ssm-agent
```

在大部分情況下，命令會報告代理程式正在執行，如下列範例所示。

```
# 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--
```

在極少數情況下，命令會報告代理程式已安裝但未執行，如下列範例所示。

```
# 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--
```

若要在這些情況下啟用代理程式，請執行下列命令。

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

在您區域中為 RHEL 8、9 和 10 建立自訂代理程式安裝命令

當利用指令碼或範本在多個執行個體上安裝 SSM Agent 時，建議使用存放在所用 AWS 區域 的安裝檔案。

針對以下命令，我們所提供的範例使用美國東部 (俄亥俄) 區域 (us-east-2) 可公開存取的 S3 儲存貯體。

> **Tip**
>
> 此主題前文所提及 8、9 和 10 RHELSSM Agent上的 快速安裝命令 程序的全域 URL 也可取代為您建構的自定區域 URL。

在下列命令中，用您自己的資訊取代*##*。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

x86_64

```
sudo dnf install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

請參閱以下範例。

```
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
```

請參閱以下範例。

```
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 安裝 SSM Agent

提供的 7 RHEL 的 Amazon Machine Images(AMIs) AWS 不會隨附預設預先安裝的 AWS Systems Manager 代理程式 (SSM Agent)。請使用此頁面的資訊協助您在 RHEL 7 執行個體上安裝或重新安裝代理程式。

主題

- 適用於 RHEL 7 的 SSM Agent 之快速安裝命令
- 在您所在區域建立適用於 RHEL 7 的自訂代理程式安裝命令

適用於 RHEL 7 的 SSM Agent 之快速安裝命令

使用以下步驟手動安裝 SSM Agent 在單一執行個體。此程序使用全域可用的安裝檔案。

在 SSM Agent 7.x 上安裝 RHEL

1. 使用您偏好的方式 (如 SSH) 連線至您的 RHEL 7 執行個體。
2. 複製適用於執行個體架構的命令，並在執行個體上執行該命令。

> **Note**
>
> 雖然以下命令的 URL 包含 `ec2-downloads-windows` 目錄，但這些是適用於 RHEL 7 的正確全域安裝檔案。

x86_64 執行個體

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

ARM64 執行個體

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. (建議) 若要驗證該代理程式是否在執行，請使用以下命令。

```
sudo systemctl status amazon-ssm-agent
```

在大部分情況下，命令會報告代理程式正在執行，如下列範例所示。

```
# 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--
```

在極少數情況下，命令會報告代理程式已安裝但未執行，如下列範例所示。

```
# 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--
```

若要在這些情況下啟用代理程式，請執行下列命令。

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

在您所在區域建立適用於 RHEL 7 的自訂代理程式安裝命令

當利用指令碼或範本在多個執行個體上安裝 SSM Agent 時，建議使用存放在所用 AWS 區域 的安裝檔案。

針對以下命令，我們所提供的範例使用美國東部 (俄亥俄) 區域 (us-east-2) 可公開存取的 S3 儲存貯體。

> **Tip**
>
> 此主題前文所提及 適用於 RHEL 7 的 SSM Agent 之快速安裝命令 程序的全域 URL 也可取代為您建構的自定區域 URL。

在下列命令中，用您自己的資訊取代*##*。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

x86_64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

請參閱以下範例。

```
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
```

請參閱以下範例。

```
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 安裝 SSM Agent

提供的 6 RHEL 的 Amazon Machine Images(AMIs) AWS 不會隨附預設預先安裝的 AWS Systems Manager 代理程式 (SSM Agent)。請使用此頁面的資訊協助您在 RHEL 6 執行個體上安裝或重新安裝代理程式。

主題

- 適用於 RHEL 6 的 SSM Agent 之快速安裝命令
- 在您所在區域建立適用於 RHEL 6 的自訂代理程式安裝命令

適用於 RHEL 6 的 SSM Agent 之快速安裝命令

使用以下步驟手動安裝 SSM Agent 在單一執行個體。此程序使用全域可用的安裝檔案。

在 SSM Agent 6.x 上安裝 RHEL

1. 使用您偏好的方式 (如 SSH) 連線至您的 RHEL 6 執行個體。
2. 複製適用於執行個體架構的命令，並在執行個體上執行該命令。

> **Note**
>
> 雖然以下命令的 URL 包含 `ec2-downloads-windows` 目錄，但這些是適用於 RHEL 6 的正確全域安裝檔案。
> 以下命令指定目錄 `3.0.1479.0` 版本而不是目錄 `latest`。這是因為 RHEL 6 不支援 SSM Agent 3.1 版及更新版本。

x86_64 執行個體

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/
SSMAgent/3.0.1479.0/linux_amd64/amazon-ssm-agent.rpm
```

x86 執行個體

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/
SSMAgent/3.0.1479.0/linux_386/amazon-ssm-agent.rpm
```

3. (建議) 若要驗證該代理程式是否在執行，請使用以下命令。

```
sudo status amazon-ssm-agent
```

在大部分情況下，命令會報告代理程式正在執行，如下列範例所示。

```
amazon-ssm-agent start/running, process 1788
```

在極少數情況下，命令會報告代理程式已安裝但未執行，如下列範例所示。

```
amazon-ssm-agent stop/waiting
```

若要在這些情況下啟用代理程式，請執行下列命令。

```
sudo start amazon-ssm-agent
```

在您所在區域建立適用於 RHEL 6 的自訂代理程式安裝命令

當利用指令碼或範本在多個執行個體上安裝 SSM Agent 時，建議使用存放在所用 AWS 區域 的安裝檔案。

針對以下命令，我們所提供的範例使用美國東部 (俄亥俄) 區域 (us-east-2) 可公開存取的 S3 儲存貯體。

> **Tip**
>
> 此主題前文所提及 適用於 RHEL 6 的 SSM Agent 之快速安裝命令 程序的全域 URL 也可取代為您建構的自定區域 URL。

在下列命令中，用您自己的資訊取代*##*。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

> **Note**
>
> 以下命令指定目錄 `3.0.1479.0` 版本而不是目錄 `latest`。這是因為 RHEL 6 不支援 SSM Agent 3.1 版及更新版本。

x86_64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/3.0.1479.0/linux_amd64/amazon-ssm-agent.rpm
```

請參閱以下範例。

```
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
```

請參閱以下範例。

```
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
```

手動在 Rocky Linux 執行個體上安裝 SSM Agent

Rocky Linux 提供的 Amazon Machine Images(AMIs) AWS 不會隨附預設預先安裝的 AWS Systems Manager 代理程式 (SSM Agent)。關於可能已預先安裝代理程式且由 AWS 管理的 AMIs，如需清單，請參閱 尋找預先安裝了 SSM Agent的 AMIs。

此章節的資訊可協助您在 Rocky Linux 執行個體手動安裝或重新安裝 SSM Agent。

開始之前

在 Rocky Linux 執行個體上安裝 SSM Agent 之前，請注意下列事項：

- 關於在所有 Linux 作業系統安裝 SSM Agent 的重要資訊，請參閱在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent。

主題

- 適用於 Rocky Linux 的 SSM Agent 之快速安裝命令
- 在您所在區域建立適用於 Rocky Linux 的自訂代理程式安裝命令

適用於 Rocky Linux 的 SSM Agent 之快速安裝命令

使用以下步驟手動安裝 SSM Agent 在單一執行個體。此程序使用全域可用的安裝檔案。

開始之前

在 Rocky Linux 執行個體上安裝 SSM Agent 之前，請注意下列事項：

- 請確保您的 Rocky Linux 執行個體已安裝 Python 2 或 Python 3。這是必要的，如此 SSM Agent 才能正常運作。

在 Rocky Linux 上安裝 SSM Agent

1. 使用您偏好的方式 (如 SSH) 連線至您的 Rocky Linux 執行個體。
2. 複製適用於執行個體架構的命令，並在執行個體上執行該命令。

   > **Note**
   >
   > 雖然以下命令的 URL 包含 `ec2-downloads-windows` 目錄，但這些是適用於 Rocky Linux 的正確全域安裝檔案。

   x86_64 執行個體

   ```
   sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
   latest/linux_amd64/amazon-ssm-agent.rpm
   ```

   ARM64 執行個體

   ```
   sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
   latest/linux_arm64/amazon-ssm-agent.rpm
   ```

3. (建議) 若要驗證該代理程式是否在執行，請使用以下命令。

```
sudo systemctl status amazon-ssm-agent
```

在大部分情況下，命令會報告代理程式正在執行，如下列範例所示。

```
# 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--
```

在極少數情況下，命令會報告代理程式已安裝但未執行，如下列範例所示。

```
# 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--
```

若要在這些情況下啟用代理程式，請執行下列命令。

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

在您所在區域建立適用於 Rocky Linux 的自訂代理程式安裝命令

當利用指令碼或範本在多個執行個體上安裝 SSM Agent 時，建議使用存放在所用 AWS 區域 的安裝檔案。

針對以下命令，我們所提供的範例使用美國東部 (俄亥俄) 區域 (us-east-2) 可公開存取的 S3 儲存貯體。

> **Tip**
>
> 此主題前文所提及 適用於 Rocky Linux 的 SSM Agent 之快速安裝命令 程序的全域 URL 也可取代為您建構的自定區域 URL。

在下列命令中，用您自己的資訊取代*##*。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

x86_64

```
sudo dnf install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

請參閱以下範例。

```
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
```

請參閱以下範例。

```
sudo dnf install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_arm64/amazon-ssm-agent.rpm
```

手動在 SUSE Linux Enterprise Server 執行個體上安裝 SSM Agent

在大多數情況下， 提供的 SUSE Linux Enterprise Server(AMIs) 的 Amazon Machine Images(SLES) AWS 會隨附預設預先安裝的 AWS Systems Manager 代理程式 (SSM Agent)。如需詳細資訊，請參閱尋找預先安裝了 SSM Agent的 AMIs。

如果新的 SLES 執行個體未預先安裝 SSM Agent，或者如果您需要手動重新安裝代理程式，此頁面的訊息會為您提供協助。

開始之前

在 SLES 執行個體上安裝 SSM Agent 之前，請注意下列事項：

- 關於在所有 Linux 作業系統安裝 SSM Agent 的重要資訊，請參閱在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent。

主題

- 適用於 SLES 的 SSM Agent 之快速安裝命令
- 在您所在區域建立適用於 SLES 的自訂代理程式安裝命令

適用於 SLES 的 SSM Agent 之快速安裝命令

使用以下步驟手動安裝 SSM Agent 在單一執行個體。此程序使用全域可用的安裝檔案。

使用快速複製及貼上命令在 SLES 安裝 SSM Agent

1. 使用您偏好的方式 (如 SSH) 連線至您的 SLES 執行個體。
2. 選項 1：使用 `zypper` 命令：

   - 執行以下命令：

     ```
     sudo zypper install amazon-ssm-agent
     ```

   - 輸入 `y` 回應任何提示。

   選項 2：使用 `rpm` 命令。

   - 在執行個體上建立暫時的目錄：

     ```
     mkdir /tmp/ssm
     ```

   - 變更為暫時的目錄。

     ```
     cd /tmp/ssm
     ```

   - 一次執行下列命令來下載和執行 SSM Agent 安裝程式。

> **Note**
>
> 雖然以下命令的 URL 包含 `ec2-downloads-windows` 目錄，但這些是適用於 SLES 的正確全域安裝檔案。

x86_64 執行個體：

```
wget https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/linux_amd64/
amazon-ssm-agent.rpm
```

ARM64 執行個體：

```
wget https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/linux_arm64/
amazon-ssm-agent.rpm
```

- 執行下列命令。

```
sudo rpm --install amazon-ssm-agent.rpm
```

- (建議) 若要驗證該代理程式是否在執行，請使用以下命令。

```
sudo systemctl status amazon-ssm-agent
```

在大部分情況下，命令會報告代理程式正在執行，如下列範例所示。

```
# 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--
```

在極少數情況下，命令會報告代理程式已安裝但未執行，如下列範例所示。

```
# 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--
```

若要在這些情況下啟用代理程式，請執行下列命令。

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

在您所在區域建立適用於 SLES 的自訂代理程式安裝命令

當利用指令碼或範本在多個執行個體上安裝 SSM Agent 時，建議使用存放在所用 AWS 區域 的安裝檔案。

針對以下命令，我們所提供的範例使用美國東部 (俄亥俄) 區域 (us-east-2) 可公開存取的 S3 儲存貯體。

> **Tip**
>
> 此主題前文所提及 適用於 Amazon Linux 1 的 SSM Agent 之快速安裝命令 程序的全域 URL 也可取代為您建構的自定區域 URL。

在下列命令中，用您自己的資訊取代*##*。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

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
```

請參閱以下範例。

```
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
```

請參閱以下範例。

```
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 執行個體手動安裝 SSM Agent

> **Important**
>
> 在 Ubuntu Server 的 64 位元版本上安裝 SSM Agent 之前，請確定您使用的是正確安裝工具。從使用 20180627 識別的 Amazon Machine Image (AMI) 開始，SSM Agent 已使用 Snap 套件預先安裝於 16.04 版。在舊版 AMI 建立的執行個體，SSM Agent 必須使用 deb 安裝程式套件進行安裝。如需詳細資訊，請參閱確定正確的 SSM Agent 版本以安裝在 64 位元 Ubuntu Server 16.04 執行個體。

在大多數情況下， 提供的 Amazon Machine Images(AMIs) Ubuntu Server 會 AWS 隨附預設預先安裝的 AWS Systems Manager 代理程式 (SSM Agent)。如需詳細資訊，請參閱 尋找預先安裝了 SSM Agent的 AMIs。

如果新的 Ubuntu Server 執行個體未預先安裝 SSM Agent，或者如果您需要手動重新安裝代理程式，此章節的訊息會為您提供協助。

開始之前

在 Ubuntu Server 執行個體上安裝 SSM Agent 之前，請注意下列事項：

- 關於在所有 Linux 作業系統安裝 SSM Agent 的重要資訊，請參閱在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent。

主題

- SSM Agent 在 Ubuntu Server 16.04 LTS 64 位元 (Snap)、18.04、20.04、20.10 STR、22.04 LTS、23.10、24.04 LTS、24.0 和 25.04 上安裝
- 在 Ubuntu Server 16.04 及 14.04 64 位元 (deb) 上安裝 SSM Agent
- 在 Ubuntu Server 16.04 及 14.04 32 位元上安裝 SSM Agent
- 確定正確的 SSM Agent 版本以安裝在 64 位元 Ubuntu Server 16.04 執行個體

SSM Agent 在 Ubuntu Server 16.04 LTS 64 位元 (Snap)、18.04、20.04、20.10 STR、22.04 LTS、23.10、24.04 LTS、24.0 和 25.04 上安裝

開始之前

在 Ubuntu Server 16.04 LTS 64 位元 (Snap)、18.04、20.04、20.10 STR、22.04 LTS、23.10、24.04 LTS、24.0 和 25.04 SSM Agent上安裝 之前，請注意下列事項：

透過 Snap 或 deb 安裝程式安裝 16.04 版

依 Ubuntu Server 16.04 AMI 的版本而定，在版本 16.04 AMI 上，使用 Snaps 或 deb 安裝套件安裝 SSM Agent。

SSM Agent 安裝程式檔案位置

在 Ubuntu Server 16.04 LTS 64 位元 (Snap)、18.04、20.04、20.10 STR、22.04 LTS、23.10、24.04 LTS、24.0 和 25.04 上，SSM Agent安裝程式檔案，包括代理程式二進位檔和組態檔案，會存放在下列目錄中：`/snap/amazon-ssm-agent/current/`。如果您變更此目錄中的任何組態檔案，則必須將這些檔案從 `/snap` 目錄複製到 `/etc/amazon/ssm/` 目錄。日誌和程式庫檔案未變更 (`/var/lib/amazon/ssm`、`/var/log/amazon/ssm`)。

使用 Snap `candidate` 管道

Snap 商店中的候選頻道包含最新版本的 SSM Agent (包括所有最新錯誤修正)；而不是穩定的頻道。若要進一步了解候選頻道和穩定頻道之間的差異，請參閱風險層級，網址為 https://snapcraft.io/docs/channels。

如果您想追蹤候選頻道上的 SSM Agent 版本資訊，請在 Ubuntu Server 20.10 STR、20.04、18.04 和 16.04 LTS 64 位元執行個體上執行下列命令。

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

18.04 版及較新版本推薦使用的 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，我們建議您僅使用 Snaps。另外驗證代理程式只有一個執行個體在您的執行個體上安裝和執行。如果您想要在沒有 Snaps 的情況下使用 SSM Agent，請解除安裝 SSM Agent。接著，使用在 Ubuntu Server 16.04 及 14.04 64 位元 (deb) 上安裝 SSM Agent 的說明，安裝 SSM Agent 作為 debian 套件。在安裝之前，請確定您安裝的任何 Snaps 不會與您想要作為 debian 套件進行管理的套件清單重疊。

`Maximum timeout exceeded` 錯誤訊息

由於 Snap 的已知問題，您可能會看到 `snap` 命令的 `Maximum timeout exceeded` 錯誤。如果您收到此錯誤，請一次執行下列一個命令來啟動、停止代理程式和檢查其狀態：

```
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 位元 (Snap)、18.04、20.04、20.10 STR、22.04 LTS、23.10、24.04 LTS、24.0 和 25.04 （使用 Snap 套件） SSM Agent上安裝

1. SSM Agent 根據預設， 會安裝在識別符AMIs為 20180627或更新版本Ubuntu Server的 16.04 LTS 64 位元 (Snap)、18.04、20.04、20.10 STR、22.04 LTS、23.10、24.04 LTS、24.0 和 25.04 上。

   如果您需要在現場部署伺服器上安裝 SSM Agent 或者如果您需要重新安裝代理程式，您可以使用以下指令碼。您不需要指定 URL 以供下載，因為 `snap` 命令會自動從 Snap 應用程式商店下載代理程式，網址為 https://snapcraft.io。

   ```
   sudo snap install amazon-ssm-agent --classic
   ```

2. 執行下列命令來判斷 SSM Agent 是否在執行。

```
sudo snap list amazon-ssm-agent
```

3. 若之前的命令傳回 `amazon-ssm-agent is stopped`、`inactive` 或 `disabled`，請執行以下命令，以啟動服務。

```
sudo snap start amazon-ssm-agent
```

4. 檢查代理程式的狀態。

```
sudo snap services amazon-ssm-agent
```

在 Ubuntu Server 16.04 及 14.04 64 位元 (deb) 上安裝 SSM Agent

> **Important**
>
> 在 Ubuntu Server 的 64 位元版本上安裝 SSM Agent 之前，請確定您使用的是修正安裝工具。從使用 20180627 識別的 Amazon Machine Image (AMI) 開始，SSM Agent 已使用 Snap 套件預先安裝於 16.04 版。在舊版 AMI 建立的執行個體，SSM Agent 必須使用 deb 安裝程式套件進行安裝。如需更多資訊，請參閱 確定正確的 SSM Agent 版本以安裝在 64 位元 Ubuntu Server 16.04 執行個體。如果 SSM Agent 安裝在您的執行個體並搭配 Snap 使用，而且您使用 deb 安裝程式套件安裝或更新 SSM Agent，該安裝或 SSM Agent 操作可能會失敗。

在大多數情況下，由 提供的 Amazon Machine Images(AMIs) Ubuntu Server 16.04 AWS 會隨附預設預先安裝的 AWS Systems Manager 代理程式 (SSM Agent)。如需詳細資訊，請參閱 尋找預先安裝了 SSM Agent的 AMIs。

如果新的 Ubuntu Server 16.04 執行個體 (20180627版以前) 未預先安裝 SSM Agent、您要在 Ubuntu Server 14.04 進行安裝，或者您需要手動重新安裝代理程式，此頁面的訊息會為您提供協助。

適用於 Ubuntu Server 16.04 及 14.04 64 位元 (deb) 的 SSM Agent 之快速安裝命令

使用以下步驟手動安裝 SSM Agent 在單一執行個體。此程序使用全域可用的安裝檔案。

使用快速複製及貼上命令在 Ubuntu Server 16.04 及 14.04 64 位元 (deb) 上安裝 SSM Agent

1. 使用您偏好的方式 (如 SSH) 連線至您的 Ubuntu Server 執行個體。

2. 執行以下命令，可在執行個體建立暫時目錄。

```
mkdir /tmp/ssm
```

3. 變更為暫時的目錄。

```
cd /tmp/ssm
```

4. 執行下列命令。

> **Note**
>
> 雖然以下命令的 URL 包含 `ec2-downloads-windows` 目錄，但這些是適用於 Ubuntu Server 16.04 及 14.04 64 位元的正確全域安裝檔案。

```
wget https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/debian_amd64/amazon-ssm-agent.deb
```

```
sudo dpkg -i amazon-ssm-agent.deb
```

5. (建議) 執行下列其中一個命令來判斷 SSM Agent 是否正在執行。

   Ubuntu Server16.04

   ```
   sudo systemctl status amazon-ssm-agent
   ```

   Ubuntu Server 14.04

   ```
   sudo status amazon-ssm-agent
   ```

   在大部分情況下，命令會報告代理程式正在執行。

   在極少數情況下，命令會報告代理程式已安裝但未執行，如下列範例所示。

6. 若之前的命令傳回 `amazon-ssm-agent is stopped`、`inactive` 或 `disabled`，請執行以下其中一個命令來啟動服務。

   Ubuntu Server 16.04：

```
sudo systemctl enable amazon-ssm-agent
```

Ubuntu Server 14.04：

```
sudo start amazon-ssm-agent
```

在您所在區域，在 Ubuntu Server 16.04 及 14.04 64 位元 (deb)上，建立適用於 SSM Agent 的自定安裝命令

當利用指令碼或範本在多個執行個體上安裝 SSM Agent 時，建議使用存放在所用 AWS 區域 的安裝檔案。

針對以下命令，我們所提供的範例使用美國東部 (俄亥俄) 區域 (us-east-2) 可公開存取的 S3 儲存貯體。

> **Tip**
>
> 此主題前文所提及 適用於 Ubuntu Server 16.04 及 14.04 64 位元 (deb) 的 SSM Agent 之快速安裝命令 程序的全域 URL 也可取代為您建構的自定區域 URL。

在下列命令中，用您自己的資訊取代*##*。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

```
wget https://s3.region.amazonaws.com/amazon-ssm-region/latest/debian_amd64/amazon-ssm-agent.deb
```

```
sudo dpkg -i amazon-ssm-agent.deb
```

請參閱以下範例。

```
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 及 14.04 32 位元上安裝 SSM Agent

在大多數情況下，由 提供的 Amazon Machine Images(AMIs) Ubuntu Server 16.04 AWS 會隨附預設預先安裝的 AWS Systems Manager 代理程式 (SSM Agent)。如需詳細資訊，請參閱 尋找預先安裝了 SSM Agent的 AMIs。

如果新的 SSM Agent 16.04 執行個體未預先安裝 Ubuntu Server 、您要在 Ubuntu Server 14.04 進行安裝，或者如果您需要手動重新安裝代理程式，此頁面的訊息會為您提供協助。

適用於 Ubuntu Server 16.04 及 14.04 32 位元 (deb) 的 SSM Agent 之快速安裝命令

使用以下步驟手動安裝 SSM Agent 在單一執行個體。此程序使用全域可用的安裝檔案。

使用快速複製及貼上命令在 Ubuntu Server 16.04 及 14.04 32 位元 (deb) 上安裝 SSM Agent

1. 使用您偏好的方式 (如 SSH) 連線至您的 Ubuntu Server 執行個體。
2. 執行以下命令，可在執行個體建立暫時目錄。

```
mkdir /tmp/ssm
```

3. 變更為暫時的目錄。

```
cd /tmp/ssm
```

4. 執行下列命令。

> **Note**
>
> 雖然以下命令的 URL 包含 `ec2-downloads-windows` 目錄，但這些是適用於 Ubuntu Server 16.04 及 14.04 32 位元的正確全域安裝檔案。

```
wget https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/debian_386/
amazon-ssm-agent.deb
```

```
sudo dpkg -i amazon-ssm-agent.deb
```

5. (建議) 執行下列其中一個命令來判斷 SSM Agent 是否正在執行。

Ubuntu Server16.04

```
sudo systemctl status amazon-ssm-agent
```

Ubuntu Server 14.04

```
sudo status amazon-ssm-agent
```

在大部分情況下，命令會報告代理程式正在執行。

在極少數情況下，命令會報告代理程式已安裝但未執行，如下列範例所示。

6. 若之前的命令傳回 `amazon-ssm-agent is stopped`、`inactive` 或 `disabled`，請執行以下其中一個命令來啟動服務。

Ubuntu Server 16.04：

```
sudo systemctl enable amazon-ssm-agent
```

Ubuntu Server 14.04：

```
sudo start amazon-ssm-agent
```

在您所在區域，在 Ubuntu Server 16.04 及 14.04 32 位元 (deb)上，建立適用於 SSM Agent 的自訂安裝命令

當利用指令碼或範本在多個執行個體上安裝 SSM Agent 時，建議使用存放在所用 AWS 區域 的安裝檔案。

針對以下命令，我們所提供的範例使用美國東部 (俄亥俄) 區域 (`us-east-2`) 可公開存取的 S3 儲存貯體。

> **Tip**
>
> 此主題前文所提及 適用於 Ubuntu Server 16.04 及 14.04 32 位元 (deb) 的 SSM Agent 之快速安裝命令 程序的全域 URL 也可取代為您建構的自定區域 URL。

在下列命令中，用您自己的資訊取代*##*。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

```
wget https://s3.region.amazonaws.com/amazon-ssm-region/latest/debian_386/amazon-ssm-agent.deb
```

```
sudo dpkg -i amazon-ssm-agent.deb
```

請參閱以下範例。

```
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
```

確定正確的 SSM Agent 版本以安裝在 64 位元 Ubuntu Server 16.04 執行個體

> **Important**
>
> 在 Ubuntu Server 的 64 位元版本上安裝 SSM Agent 之前，請確定您使用的是修正安裝工具。從使用 20180627 識別的 Amazon Machine Image (AMI) 開始，SSM Agent 已使用 Snap 套件預先安裝於 16.04 版。在舊版 AMI 建立的執行個體，SSM Agent 必須使用 deb 安裝程式套件進行安裝。如需詳細資訊，請參閱 確定正確的 SSM Agent 版本以安裝在 64 位元 Ubuntu Server 16.04 執行個體。
>
> 請注意，如果執行個體有多個 SSM Agent 安裝 (例如，一個安裝使用 Snap，一個安裝使用 deb 安裝程式)，您的代理程式將無法正常運作。

您可以使用下列任一種方法來驗證執行個體的來源 AMI ID。這些程序僅適用於 AWS 受管 AMIs。

驗證來源 AMI ID 建立日期 (主控台)

1. 在 https://console.aws.amazon.com/ec2/ 開啟 Amazon EC2 主控台。
2. 在左側導覽窗格中選擇 (執行個體)。
3. 選取執行個體。
4. 在 Details (詳細資料) 索引標籤，檢查 AMI 名稱欄位下的值是否有 YYYYMMDD 限定詞。例如：`ubuntu/images/hvm-ssd/ubuntu-xenial-16.04-amd64-server-20180627`。

驗證來源 AMI ID 建立日期 (AWS CLI)

- 執行下列命令。

```
aws ec2 describe-images --image-ids ami-id
```

ami-id 代表 AMI提供的 ID AWS，例如 ami-07c8bc5c1ce9598c3。

如果成功，命令會傳回如下資訊，您可以在其中檢查 CreationDate 和 Name 欄位以取得資訊。

```
{
    "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"
        }
    ]
}
```

將 SSM Agent 設定為使用 Linux 節點上的代理

您可以透過建立覆寫組態檔案並將 http_proxy、 和 no_proxy設定新增至檔案https_proxy，來設定 AWS Systems Manager 客服人員 (SSM Agent) 透過 HTTP 代理進行通訊。如果您安裝較新或較舊版本的 SSM Agent，覆寫檔案也會保留代理設定。本節包括在 upstart 和 systemd 環境中建立覆寫檔案的程序。如果您打算使用 Session Manager，請注意不支援 HTTPS 代理伺服器。

主題

- 將 SSM Agent設定為使用代理 (upstart)
- 將 SSM Agent設定為使用代理 (systemd)

將 SSM Agent設定為使用代理 (upstart)

使用下列程序可建立 `upstart` 環境的覆寫組態檔案。

設定 SSM Agent 以使用代理 (upstart)

1. 連線至安裝 SSM Agent 所在的受管執行個體。
2. 開啟諸如 VIM 的簡單編輯器，並根據使用的是 HTTP 代理伺服器或 HTTPS 代理伺服器，來新增下列其中一個組態。

   對於 HTTP 代理伺服器：

   ```
   env http_proxy=http://hostname:port
   env https_proxy=http://hostname:port
   env no_proxy=IP address for instance metadata services (IMDS)
   ```

   對於 HTTPS 代理伺服器：

   ```
   env http_proxy=http://hostname:port
   env https_proxy=https://hostname:port
   env no_proxy=IP address for instance metadata services (IMDS)
   ```

   Important

   將 `no_proxy` 設定新增到檔案，並指定 IP 位址。`no_proxy` 的 IP 位址是 Systems Manager 的執行個體中繼資料服務 (IMDS) 端點。如果未指定 `no_proxy`，則對 Systems Manager 的呼叫會從代理服務取得身分 (如果啟用了 IMDSv1 備用)，或對 Systems Manager 的呼叫失敗 (如果強制執行 IMDSv2)。

   - 針對 IPv4，請指定 `no_proxy=169.254.169.254`。
   - 針對 IPv6，請指定 `no_proxy=[fd00:ec2::254]`。執行個體中繼資料服務的 IPv6 地址與 IMDSv2 命令相容。IPv6 位址只能在建置於 AWS Nitro 系統的執行個體上進行存取。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的 How Instance Metadata Service Version 2 works。
3. 在下列位置使用名稱 `amazon-ssm-agent.override` 儲存檔案：`/etc/init/`
4. 使用下列命令停止和重新啟動 SSM Agent：

```
sudo service stop amazon-ssm-agent
sudo service start amazon-ssm-agent
```

> **Note**
>
> 如需有關在 Upstart 環境中使用 `.override` 檔案的詳細資訊，請參閱 init：Upstart init 協助程式任務組態。

將 SSM Agent設定為使用代理 (systemd)

使用以下程序來設定 SSM Agent，以使用 `systemd` 環境中的代理。

> **Note**
>
> 此程序中的某些步驟包含 Ubuntu Server 執行個體的明確指示，以使用 Snap 安裝 SSM Agent。

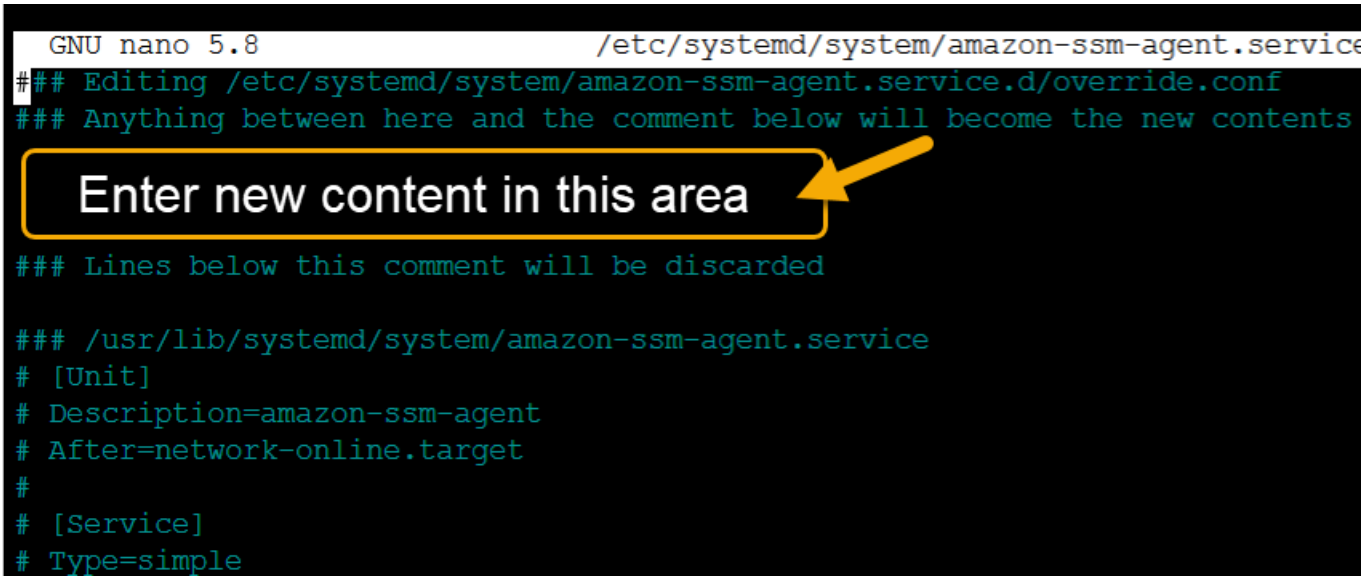
1. 連接到其中安裝 SSM Agent 的執行個體。
2. 根據作業系統類型，執行以下其中一個命令。
   - 在 Ubuntu Server 執行個體上，使用 Snap 安裝 SSM Agent：

     ```
     sudo systemctl edit snap.amazon-ssm-agent.amazon-ssm-agent
     ```

     在其他作業系統中：

     ```
     sudo systemctl edit amazon-ssm-agent
     ```
3. 開啟諸如 VIM 的簡單編輯器，並根據使用的是 HTTP 代理伺服器或 HTTPS 代理伺服器，來新增下列其中一個組態。

   確保在「`### Lines below this comment will be discarded`」的註解上方輸入資訊，如下圖所示。



對於 HTTP 代理伺服器：

```
[Service]
Environment="http_proxy=http://hostname:port"
Environment="https_proxy=http://hostname:port"
Environment="no_proxy=IP address for instance metadata services (IMDS)"
```

對於 HTTPS 代理伺服器：

```
[Service]
Environment="http_proxy=http://hostname:port"
Environment="https_proxy=https://hostname:port"
Environment="no_proxy=IP address for instance metadata services (IMDS)"
```

Important

將 no_proxy 設定新增到檔案，並指定 IP 位址。no_proxy 的 IP 位址是 Systems Manager 的執行個體中繼資料服務 (IMDS) 端點。如果未指定 no_proxy，則對 Systems Manager 的呼叫會從代理服務取得身分 (如果啟用了 IMDSv1 備用)，或對 Systems Manager 的呼叫失敗 (如果強制執行 IMDSv2)。

- 針對 IPv4，請指定 no_proxy=169.254.169.254。
- 針對 IPv6，請指定 no_proxy=[fd00:ec2::254]。執行個體中繼資料服務的 IPv6 地址與 IMDSv2 命令相容。IPv6 位址只能在建置於 AWS Nitro 系統的執行個體上

進行存取。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的 How Instance Metadata Service Version 2 works。

4. 儲存您的變更。根據作業系統類型，系統會自動建立下列其中一個檔案。

   - 在 Ubuntu Server 執行個體上，使用 Snap 安裝 SSM Agent：

     `/etc/systemd/system/snap.amazon-ssm-agent.amazon-ssm-agent.service.d/override.conf`

   - 在 Amazon Linux 2 和 Amazon Linux 2023 執行個體上：

     `/etc/systemd/system/amazon-ssm-agent.service.d/override.conf`

   - 在其他作業系統中：

     `/etc/systemd/system/amazon-ssm-agent.service.d/amazon-ssm-agent.override`

5. 根據作業系統類型，使用下列其中一個命令，重新啟動 SSM Agent。

   - 在使用 Snap 安裝的 Ubuntu Server 執行個體上：

     ```
     sudo systemctl daemon-reload && sudo systemctl restart snap.amazon-ssm-agent.amazon-ssm-agent
     ```

   - 在其他作業系統中：

     ```
     sudo systemctl daemon-reload && sudo systemctl restart amazon-ssm-agent
     ```

> **Note**
>
> 如需有關在 systemd 環境中使用 `.override` 檔案的詳細資訊，請參閱 Red Hat Enterprise Linux 7 系統管理員指南中的修改現有單元檔案。

在 macOS 專用 EC2 執行個體使用 SSM Agent

AWS Systems Manager (SSM Agent) 會處理 Systems Manager 請求，並依請求中指定的方式設定您的機器。請使用下列程序為 macOS 安裝、設定或解除安裝 SSM Agent。

> **Note**
>
> SSM Agent 預設會預先安裝在 macOS 的 Amazon Machine Images (AMIs) 上。您不需要在 macOS 的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上安裝 SSM Agent，除非您已將其解除安裝。

可在 GitHub 上使用 SSM Agent 的原始碼，以便調整代理程式來符合需求。我們建議您為想要進行的變更提交提取請求。但是， AWS 不支援執行修改過的軟體複本。

> **Note**
>
> 若要檢視不同版本的 SSM Agent 的詳細資訊，請參閱版本備註。

在 macOS 作業系統上手動安裝 SSM Agent 之前，請先檢閱下列資訊。

- SSM Agent 預設會安裝在 EC2 執行個體上，並由 Systems Manager Amazon Machine Images支援。如需詳細資訊，請參閱 支援的作業系統清單macOS。

  不需要在 macOS EC2 執行個體上手動安裝 SSM Agent，除非已解除安裝。
- 並非所有 macOS 都支援 的 EC2 執行個體 AWS 區域。如需支援 macOS 的 x86 型和 M1 EC2 執行個體的區域清單，請參閱《Amazon EC2 常見問答集》中的 macOS 工作負載。
- 每當將新工具新增至 Systems Manager 或對現有工具進行更新時，就會SSM Agent發佈 的更新版本。未使用最新版本的代理程式，可能會使您的受管節點無法使用各種 Systems Manager 工具和功能。因此，我們建議您讓機器的 SSM Agent 自動保持最新狀態。如需相關資訊，請參閱自動化 SSM Agent 更新。請訂閱 GitHub 上的 SSM Agent 版本備註頁面，以便接收有關 SSM Agent 更新的通知。

主題

- 在 macOS EC2 執行個體上手動安裝和解除安裝 SSM Agent

在 macOS EC2 執行個體上手動安裝和解除安裝 SSM Agent

連線至您的 macOS 執行個體並執行下列步驟，以安裝 AWS Systems Manager Agent (SSM Agent)。在將使用 Systems Manager 執行命令的每個執行個體上執行這些步驟。此程序中所提供的命令也可以透過使用者資料，以指令碼的形式傳遞至 Amazon EC2 執行個體。

開始之前

使用 Homebrew 安裝 `wget`。

在 macOS 上安裝 SSM Agent

1. 使用以下命令來下載 x86_64 執行個體的代理程式安裝程式檔案。

   在下列命令中，用您自己的資訊取代*##*。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域欄位。

   ```
   sudo wget https://s3.region.amazonaws.com/amazon-ssm-region/latest/darwin_amd64/
   amazon-ssm-agent.pkg
   ```

   針對 Apple silicon 執行個體使用下列命令。

   ```
   sudo wget https://s3.region.amazonaws.com/amazon-ssm-region/latest/darwin_arm64/
   amazon-ssm-agent.pkg
   ```

   請見此處範例。

   ```
   sudo wget https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
   darwin_amd64/amazon-ssm-agent.pkg
   ```

2. 使用以下命令來執行 SSM Agent 安裝程式。

   x86_64:

   ```
   sudo installer -pkg amazon-ssm-agent.pkg -target /
   ```

3. 檢查代理程式的狀態。

   若要判斷 SSM Agent 是否正在執行，請檢查位於 `/var/log/amazon/ssm/amazon-ssm-agent.log` 的代理程式日誌。

4. 如果代理程式日誌指出 "amazon-ssm-agent is stopped." (amazon-ssm-agent 已停止)，請執行下列命令來啟動服務。

   ```
   sudo launchctl load -w /Library/LaunchDaemons/com.amazon.aws.ssm.plist && sudo
    launchctl start com.amazon.aws.ssm
   ```

> **Important**
>
> 每當將新工具新增至 Systems Manager 或對現有工具進行更新時，SSM Agent就會發行更新的 版本。未使用最新版本的代理程式可能會使您的受管節點無法使用各種 Systems Manager 工具和功能。因此，我們建議您讓機器的 SSM Agent 自動保持最新狀態。如需相關資訊，請參閱 自動化 SSM Agent 更新。請訂閱 GitHub 上的 SSM Agent 版本備註頁面，以便接收有關 SSM Agent 更新的通知。

從 macOS 執行個體解除安裝 SSM Agent

macOS 本就不支援解除安裝 PKG 檔案。若要從 的 Amazon Elastic Compute Cloud (Amazon EC2SSM Agent) 執行個體解除安裝 AWS Systems Manager 代理程式 ()macOS，您可以從下列位置使用 AWS 受管指令碼。

https://github.com/aws/amazon-ssm-agent/blob/mainline/Tools/src/update/darwin/uninstall.sh

在 Windows Server 專用 EC2 執行個體使用 SSM Agent

AWS Systems Manager 根據預設，代理程式 (SSM Agent) 預先安裝在 提供的 Windows Server Amazon Machine Images(AMIs) 上 AWS。支援下列作業系統 (OS) 版本。

- Windows Server 2008-2012 R2 AMIs 發佈於 2016 月 11 月或之後
- Windows Server 2016、2019、2022 (Nano 版本除外） 和 2025

有關對先前版本所提供之支援的說明

2016 年 11 月之前發佈的 Windows Server AMIs 使用 EC2Config 服務來處理請求和設定執行個體。

除非您有特定理由需使用 EC2Config 服務或舊版 SSM Agent 來處理 Systems Manager 請求，否則建議對已為 Systems Manager 設定的在混合多雲端環境中的每個 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體或非 EC2 機器，下載並安裝最新版的 SSM Agent。

從 2020 年 1 月 14 日起，Microsoft 不再支援 Windows Server 2008 的功能或安全性更新。Windows Server 2008 和 2008 R2 的舊版 Amazon Machine Images (AMIs) 仍包含預先安裝的 SSM Agent 的版本 2，但 Systems Manager 不再正式支援 2008 版本，並且不再更新這些 Windows Server 版本的代理程式。除此之外，SSM Agent 第 3 版可能無法與 Windows Server 2008 和 2008 R2 上的所有操作相容。Windows Server 2008 版本的 SSM Agent 的最終的正式支援版本是 2.3.1644.0。

讓 SSM Agent 保持最新

每當將新工具新增至 Systems Manager 或對現有工具進行更新時，SSM Agent就會發佈 的更新版本。若未使用最新版本的代理程式，您的受管節點可能會無法使用各種 Systems Manager 工具和功能。因此，我們建議您讓機器的 SSM Agent 自動保持最新狀態。如需相關資訊，請參閱自動化 SSM Agent 更新。請訂閱 GitHub 上的 SSM Agent 版本備註頁面，以便接收有關 SSM Agent 更新的通知。

若要檢視不同版本的 SSM Agent 的詳細資訊，請參閱版本備註。

主題

- 在 Windows Server EC2 執行個體上手動安裝和解除安裝 SSM Agent
- 將 SSM Agent 設定為使用 Windows Server 執行個體的代理

在 Windows Server EC2 執行個體上手動安裝和解除安裝 SSM Agent

AWS Systems Manager 在 Amazon Windows Server提供的下列 (SSM Agent) 上，預設會預先安裝代理程式 Amazon Machine Images(AMIs)：

- Windows Server 2008-2012 R2 AMIs 發佈於 2016 月 11 月或之後
- Windows Server 2016、2019、2022 (Nano 版本除外） 和 2025

在 Windows Server EC2 執行個體上安裝 SSM Agent

如有需要，您可以使用以下程序，在 Windows Server 的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上手動下載並安裝最新版本的 SSM Agent。此程序中所提供的命令也可以透過使用者資料，以指令碼的形式傳遞至 Amazon EC2 執行個體。

SSM Agent 需要 Windows PowerShell 3.0 或更新版本，才能在 Windows Server 執行個體上執行特定 AWS Systems Manager 文件 (SSM 文件) (例如，舊版 `AWS-ApplyPatchBaseline` 文件)。確認您的 Windows Server 執行個體執行 Windows Management Framework 3.0 或更新版本。這個架構包含 Windows PowerShell。如需詳細資訊，請參閱 Windows Management Framework 3.0。

> **Note**
>
> 此程序適用於在 Windows Server 的 EC2 執行個體上安裝或重新安裝 SSM Agent。如果您需要在內部部署伺服器或虛擬機器 (VM) 安裝代理程式，以便與 Systems Manager 搭配使用，請參閱如何在混合 Windows 節點上安裝 SSM Agent。

在 Windows Server 的 EC2 執行個體上手動安裝 SSM Agent 的最新版本

1. 使用遠端桌面或 Windows PowerShell 連線至您的執行個體。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Connect to your instance。
2. 下載最新版本的 SSM Agent 到您的執行個體。您可以使用 PowerShell 命令或直接下載連結進行下載。

> **Note**
>
> 此步驟中的 URLs 可讓您SSM Agent從任何 AWS 區域下載。如果您想要從特定區域下載代理程式，請改為使用區域特定的 URL：
> `https://amazon-ssm-region.s3.region.amazonaws.com/latest/windows_amd64/AmazonSSMAgentSetup.exe`
> *region* 代表 支援的 AWS 區域 識別符 AWS Systems Manager，例如us-east-2美國東部（俄亥俄）區域。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域資料欄。

PowerShell

依序執行下列三個 PowerShell 命令。這些命令允許您直接下載 SSM Agent，而不用調整 Internet Explorer (IE) 增強安全性設定，並可接著安裝代理程式及移除安裝檔案。

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
```

直接下載

使用下列連結，下載 SSM Agent 的最新版本至執行個體。如果需要，請使用 AWS 區域特定 URL 更新此 URL。

https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/windows_amd64/AmazonSSMAgentSetup.exe

執行下載的 `AmazonSSMAgentSetup.exe` 檔案來安裝 SSM Agent。

3. 在 PowerShell 中傳送以下命令來啟動或重新啟動 SSM Agent：

```
Restart-Service AmazonSSMAgent
```

> **Note**
>
> 若要從 Windows Server 執行個體解除安裝 SSM Agent，請開啟控制面板、程式。選擇 Uninstall a program (解除安裝程式) 選項。開啟 Amazon SSM Agent 的內容 (按一下滑鼠右鍵) 選單，然後選擇解除安裝。

將 SSM Agent 設定為使用 Windows Server 執行個體的代理

本主題中的資訊適用於在 2016 年 11 月或之後建立的未使用 Nano 安裝選項的 Windows Server 執行個體。如果您打算使用 Session Manager，請注意不支援 HTTPS 代理伺服器。

> **Note**
>
> 從 2020 年 1 月 14 日起，Microsoft 不再支援 Windows Server 2008 的功能或安全性更新。Windows Server 2008 和 2008 R2 的舊版 Amazon Machine Images (AMIs) 仍包含預先安裝的 SSM Agent 的版本 2，但 Systems Manager 不再正式支援 2008 版本，並且不再更新

這些 Windows Server 版本的代理程式。除此之外，SSM Agent 第 3 版可能無法與 Windows Server 2008 和 2008 R2 上的所有操作相容。Windows Server 2008 版本的 SSM Agent 的最終的正式支援版本是 2.3.1644.0。

開始之前

將 SSM Agent 設定為使用代理之前，請注意下列重要資訊。

在下列程序中，您需執行命令來將 SSM Agent 設定為使用代理。該命令包含具有 IP 位址的 `no_proxy` 設定。其中的 IP 位址是 Systems Manager 的執行個體中繼資料服務 (IMDS) 端點。如果未指定 `no_proxy`，則對 Systems Manager 的呼叫會從代理服務取得身分 (如果啟用了 IMDSv1 備用)，或對 Systems Manager 的呼叫失敗 (如果強制執行 IMDSv2)。

- 針對 IPv4，請指定 `no_proxy=169.254.169.254`。
- 針對 IPv6，請指定 `no_proxy=[fd00:ec2::254]`。執行個體中繼資料服務的 IPv6 地址與 IMDSv2 命令相容。IPv6 位址只能在建置於 AWS Nitro 系統的執行個體上進行存取。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的 How Instance Metadata Service Version 2 works。

若要設定 SSM Agent 以使用代理

1. 使用遠端桌面或 Windows PowerShell 連接到您要設定為使用代理的執行個體。
2. 在 PowerShell 中執行下列命令區塊。使用與您的代理有關的資訊取代*####*和*###*。

```
$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
```

執行上述命令之後，您可以檢閱 SSM Agent 記錄檔以確認已套用 Proxy 設定。記錄檔中的項目看起來如下所示。如需 SSM Agent 日誌的詳細資訊，請參閱檢視 SSM Agent 日誌。

```
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
```

若要重設 SSM Agent 代理組態

1. 使用遠端桌面或 Windows PowerShell 連接到您要設定的執行個體。
2. 如果您使用遠端桌面連接，請將 PowerShell 啟動為系統管理員。
3. 在 PowerShell 中執行下列命令區塊。

```
Remove-ItemProperty -Path HKLM:\SYSTEM\CurrentControlSet\Services\AmazonSSMAgent -
Name Environment
Restart-Service AmazonSSMAgent
```

SSM Agent 代理設定優先順序

在 Windows Server 執行個體上設定 SSM Agent的代理設定時，請務必瞭解在 SSM Agent 啟動時，系統會評估這些設定並套用至代理程式組態。您為 Windows Server 執行個體設定的代理設定會決定其他設定是否可能取代您所要的設定。代理程式會使用其找到的第一個代理設定。

> **Important**
>
> SSM Agent 使用 HTTPS 協定進行通訊。因此，您必須使用以下其中一個設定選項配置 HTTPS `proxy` 參數。

依以下順序評估 SSM Agent 代理設定。

1. AmazonSSMAgent 登錄檔設定 (HKLM:\SYSTEM\CurrentControlSet\Services\AmazonSSMAgent)
2. System 環境變數 (http_proxy、https_proxy、no_proxy)
3. LocalSystem 使用者帳戶環境變數 (http_proxy、https_proxy、no_proxy)
4. 瀏覽器設定 (HTTP、secure、exceptions)
5. WinHTTP 代理設定 (http=、https=、bypass-list=)

SSM Agent 代理設定和 Systems Manager 服務

如果您SSM Agent將 設定為使用代理，並使用在Windows Server執行個體上執行期間使用 PowerShell 或 Windows Update Run Command 用戶端 AWS Systems Manager 的工具Patch Manager，請設定其他代理設定。否則，操作可能會失敗，因為 PowerShell 和 Windows Update 用戶端使用的代理設定不會繼承自 SSM Agent 代理組態。

對於 Run Command，在您的 Windows Server 執行個體上設定 WinINet 代理設定。根據每個工作階段提供 [System.Net.WebRequest] 命令。若要將這些組態套用至在 Run Command 中執行的後續網路命令，這些命令在相同的 aws:runPowershellScript 外掛程式輸入中必須優先於其他 PowerShell 命令。

下列 PowerShell 命令會傳回目前的 WinINet 代理設定，並將您的代理設定套用到 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
```

對於 Patch Manager，設定全系統範圍的代理設定，讓 Windows Update 用戶端可以掃描和下載更新。我們建議您使用 Run Command 來執行下列命令，因為這些命令會在系統管理員帳戶上執行，且設定會套用到全系統。下列 netsh 命令會傳回目前的代理設定，並將您的代理設定套用到本機系統。

```
netsh winhttp show proxy

netsh winhttp set proxy proxy-server="hostname:port" bypass-list="169.254.169.254"
```

如需有關使用 Run Command 的詳細資訊，請參閱 AWS Systems Manager Run Command。

檢查 SSM Agent 狀態並啟動代理程式

本主題列出 命令，以檢查 AWS Systems Manager 代理程式 (SSM Agent) 是否在每個支援的作業系統上執行。它也會提供用於啟動代理程式 (如果代理程式未執行) 的命令。

作業系統	用於檢查 SSM Agent 狀態的命令	用於啟動 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`
		
		`sudo systemctl start amazon-ssm-agent`

macOS	檢查位於 `/var/log/amazon/ssm/amazon-ssm-agent.log` 的代理程式日誌檔案	`sudo launchctl load -w /Library/LaunchDaemons/com.amazon.aws.ssm.plist`
		
		`sudo launchctl start com.amazon.aws.ssm`

作業系統	用於檢查 SSM Agent 狀態的命令	用於啟動 SSM Agent 的命令
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 位元)	`sudo status amazon-ssm-agent`	`sudo start amazon-ssm-agent`
Ubuntu Server 16.04 64 位元執行個體 (deb 套件安裝)	`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`

作業系統	用於檢查 SSM Agent 狀態的命令	用於啟動 SSM Agent 的命令
Windows Server	在 PowerShell 中執行： `Get-Service AmazonSSM Agent`	在 PowerShell 管理員模式下執行： `Start-Service AmazonSSMAgent`

詳細資訊

- 在 Linux 的 EC2 執行個體使用 SSM Agent
- 在 Windows Server 專用 EC2 執行個體使用 SSM Agent
- 檢查 SSM Agent 版本編號

檢查 SSM Agent 版本編號

某些 AWS Systems Manager 功能具有先決條件，其中包括安裝在受管節點上的 Systems Manager Agent (SSM Agent) 版本下限。您可以使用 Systems Manager 主控台或登入受管節點，在受管節點上取得目前安裝的 SSM Agent 版本。

> **Note**
>
> 若要檢視舊版的詳細資訊，請參閱 GitHub 上的SSM Agent版本備註。

下列程序描述如何在受管節點上取得目前安裝的 SSM Agent 版本。

檢查受管節點上安裝的 SSM Agent 版本編號

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 在 SSM Agent 版本欄位，註明代理程式版本數字。

從作業系統內取得目前安裝的 SSM Agent 版本

從下列索引標籤中選擇，以便從作業系統內取得目前安裝的 SSM Agent 版本。

Amazon Linux 1, Amazon Linux 2, and Amazon Linux 2023

> **Note**
>
> 這個命令會根據作業系統的套件管理員而有所不同。

1. 登入您的受管節點。
2. 執行下列命令。

```
yum info amazon-ssm-agent
```

此命令會傳回類似以下的輸出。

```
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Installed Packages
Name        : amazon-ssm-agent
Arch        : x86_64
Version     : 3.0.655.0
```

CentOS

1. 登入您的受管節點。
2. 對於 CentOS 6 和 7，執行下列命令。

```
yum info amazon-ssm-agent
```

此命令會傳回類似以下的輸出。

```
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Installed Packages
Name        : amazon-ssm-agent
Arch        : x86_64
Version     : 3.0.655.0
```

Debian Server

1. 登入您的受管節點。
2. 執行下列命令。

```
apt list amazon-ssm-agent
```

此命令會傳回類似以下的輸出。

```
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. 登入您的受管節點。
2. 執行下列命令。

```
pkgutil --pkg-info com.amazon.aws.ssm
```

RHEL

1. 登入您的受管節點。
2. 對於 RHEL 6、7、8 和 9，執行以下命令。

```
yum info amazon-ssm-agent
```

此命令會傳回類似以下的輸出。

```
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Installed Packages
Name        : amazon-ssm-agent
Arch        : x86_64
Version     : 3.0.655.0
```

對於 DNF 套件公用程式，執行以下命令。

```
dnf info amazon-ssm-agent
```

SLES

1. 登入您的受管節點。
2. 對於 SLES 12 和 15，執行下列命令。

```
zypper info amazon-ssm-agent
```

此命令會傳回類似以下的輸出。

```
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**
>
> 若要檢查您的 Ubuntu Server 16.04 執行個體是否使用 deb 或 Snap 套件，請參閱 在 Ubuntu Server 執行個體手動安裝 SSM Agent。

1. 登入您的受管節點。
2. 對 Ubuntu Server 16.04 和 14.04 64 位元 (包含 deb 安裝程式套件) 執行以下命令。

```
apt list amazon-ssm-agent
```

此命令會傳回類似以下的輸出。

```
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
```

針對 Ubuntu Server 16.04 LTS 64 位元 (Snap)、18.04、20.04、20.10 STR、22.04 LTS、23.10、24.04 LTS、24.0 和 25.04 執行下列命令。

```
sudo snap list amazon-ssm-agent
```

此命令會傳回類似以下的輸出。

```
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. 登入您的受管節點。
2. 執行下列 PowerShell 命令。

```
& "C:\Program Files\Amazon\SSM\amazon-ssm-agent.exe" -version
```

此命令會傳回類似以下的輸出。

```
SSM Agent version: 3.1.804.0
```

建議您使用最新版本的 SSM Agent，讓您可以從全新或更新的功能中受益。若要確保受管執行個體一律執行最新版本的 SSM Agent，您可以將更新 SSM Agent 的程序自動化。如需詳細資訊，請參閱自動化 SSM Agent 更新。

檢視 SSM Agent 日誌

AWS Systems Manager 代理程式 (SSM Agent) 會寫入有關執行、命令、排程動作、錯誤和運作狀態的資訊，以記錄每個受管節點上的檔案。透過手動連線至受管節點，可檢視日誌檔案，或者您可以自動傳送日誌至 Amazon CloudWatch Logs。如需有關將日誌傳送至 CloudWatch Logs 的詳細資訊，請參閱在 中記錄和監控 AWS Systems Manager。

您可以在下列位置中檢視受管節點上的 SSM Agent 日誌。

Linux and macOS

```
/var/log/amazon/ssm/
```

Windows

```
%PROGRAMDATA%\Amazon\SSM\Logs\
```

對於 Linux 受管節點，SSM Agent `stderr` 和 `stdout` 檔案會寫入下列目錄：`/var/lib/amazon/ssm/`。

對於 Windows 受管節點，SSM Agent `stderr` 和 `stdout` 檔案會寫入下列目錄：`%PROGRAMDATA%\Amazon\SSM\InstanceData\`。

如需有關允許 SSM Agent 偵錯記錄的資訊，請參閱 允許 SSM Agent 偵錯記錄。

如需有關 `cihub/seelog` 組態的詳細資訊，請參閱 GitHub 上的 Seelog Wiki。如需 `cihub/seelog` 組態的範例，請參閱 GitHub 上的 cihub/seelog 範例儲存庫。

允許 SSM Agent 偵錯記錄

使用下列程序，以便允許受管節點上的 SSM Agent 偵錯記錄。

Linux and macOS

允許 SSM Agent 對 Linux 和 macOS 受管節點上的日誌記錄進行偵錯

1. 使用 中的Session Manager工具 AWS Systems Manager，連線到您要允許偵錯記錄的受管節點，或登入受管節點。如需詳細資訊，請參閱使用 Session Manager。
2. 找到 seelog.xml.template 檔案。

   Linux：

在大多數 Linux 受管節點類型上，檔案位於目錄 `/etc/amazon/ssm/seelog.xml.template` 中。

在 Ubuntu Server 20.10 STR、20.04、18.04 和 16.04 LTS 上，檔案位於目錄 `/snap/amazon-ssm-agent/current/seelog.xml.template` 中。將此檔案從 `/snap/amazon-ssm-agent/current/` 目錄複製至 `/etc/amazon/ssm/` 目錄，然後再進行任何變更。

macOS:

在 macOS 執行個體類型上，檔案位於目錄 `/opt/aws/ssm/seelog.xml.template` 中。

3. 將檔名從 `seelog.xml.template` 變更為 `seelog.xml`。

> **Note**
>
> 在 Ubuntu Server 20.10 STR、20.04、18.04 和 16.04 LTS 上，必須在目錄 `/etc/amazon/ssm/` 中建立檔案 `seelog.xml`。可透過執行下列命令來建立此目錄和檔案。
>
> ```
> sudo mkdir -p /etc/amazon/ssm
> ```
>
> ```
> sudo cp -p /snap/amazon-ssm-agent/current/seelog.xml.template /etc/amazon/ssm/seelog.xml
> ```

4. 編輯 `seelog.xml` 檔案以變更預設的記錄行為。將 minlevel (minlevel) 的值從 info (資訊) 變更為 debug (除錯)，如以下範例所示。

```
<seelog type="adaptive" mininterval="2000000" maxinterval="100000000" critmsgcount="500" minlevel="debug">
```

5. (選用) 使用以下命令重新啟動 SSM Agent。

Linux：

```
sudo service amazon-ssm-agent restart
```

macOS:

```
sudo /opt/aws/ssm/bin/amazon-ssm-agent restart
```

Windows

允許 SSM Agent 對 Windows Server 受管節點上的日誌記錄進行偵錯

1. 您可以使用 Session Manager 來連線至想要允許偵錯日誌記錄的受管節點，或登入受管節點。如需詳細資訊，請參閱使用 Session Manager。
2. 請複製 seelog.xml.template 檔案。將該複製的檔案名稱改為 seelog.xml。檔案位於以下目錄：

   ```
   %PROGRAMFILES%\Amazon\SSM\seelog.xml.template
   ```
3. 編輯 `seelog.xml` 檔案以變更預設的記錄行為。將 minlevel (minlevel) 的值從 info (資訊) 變更為 debug (除錯)，如以下範例所示。

   ```
   <seelog type="adaptive" mininterval="2000000" maxinterval="100000000"
   critmsgcount="500" minlevel="debug">
   ```
4. 找到下列項目。

   ```
   filename="{{LOCALAPPDATA}}\Amazon\SSM\Logs\{{EXECUTABLENAME}}.log"
   ```

   變更此項目以使用以下路徑。

   ```
   filename="C:\ProgramData\Amazon\SSM\Logs\{{EXECUTABLENAME}}.log"
   ```
5. 找到下列項目。

   ```
   filename="{{LOCALAPPDATA}}\Amazon\SSM\Logs\errors.log"
   ```

   變更此項目以使用以下路徑。

   ```
   filename="C:\ProgramData\Amazon\SSM\Logs\errors.log"
   ```
6. 在管理員模式中使用下列 PowerShell 命令，重新啟動 SSM Agent。

   ```
   Restart-Service AmazonSSMAgent
   ```

限制透過 SSM Agent 存取根層級命令

AWS Systems Manager 代理程式 (SSM Agent) 使用根許可 (Linux) 或 SYSTEM 許可 ()，在混合和多雲端環境中的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體和其他機器類型上執行 Windows Server。由於這些是最高層級的系統存取許可，任何已獲得許可傳送命令至 SSM Agent 的受信任實體，具有根或系統管理員許可。（在 中 AWS，可在 中執行動作和存取資源的受信任實體 AWS 稱為主體。 委託人可以是 AWS 帳戶根使用者、 使用者或 角色。）

委託人需要這個層級的存取權，才能傳送授權的 Systems Manager 命令至 SSM Agent，委託人也可在 SSM Agent 中利用任何潛在的漏洞來執行惡意程式碼。

尤其是執行命令 SendCommand 和 StartSession 的許可，應謹慎加以限制。一個良好的第一步是只將每個命令的許可授與組織中的精選委託人。不過，我們建議您限制哪些受管節點委託人可以在其上執行這些命令，以進一步加強您的安全狀態。這可在指派給主體的 IAM 政策中實現。在 IAM 政策中，您可以包含條件，限制使用者僅在標記有特定標籤或是標籤組合的受管節點上執行命令。

例如，假設您有兩個機群的伺服器，一個用於測試，一個用於生產。在套用到資淺工程師的 IAM 政策中，您指定他們僅可在標記有 `ssm:resourceTag/testServer` 標籤的執行個體上執行命令。但是，對於應該能夠存取所有執行個體的一小群首席工程師，您對標記有 `ssm:resourceTag/testServer` 和 `ssm:resourceTag/productionServer` 標籤的執行個體授予存取權。

使用此方法，如果資淺工程師嘗試在生產執行個體上執行命令，系統將拒絕其存取，因為其指派的 IAM 政策未對標記有 `ssm:resourceTag/productionServer` 標籤的執行個體提供明確存取權。

如需詳細資訊及範例，請參閱下列主題：

- 根據標籤限制 Run Command 存取
- 根據執行個體標籤限制工作階段存取

自動化 SSM Agent 更新

AWS 當我們新增或更新 Systems Manager 工具時， 會發行新版本的 AWS Systems Manager 代理程式 (SSM Agent)。如果您的受管節點使用舊版的代理程式，則無法使用新工具或受益於更新的工具。基於這些原因，建議您使用下列方法其中之一，讓您將受管節點上的 SSM Agent 更新程序自動化。

Bottlerocket 作業系統上的代理程式更新

Bottlerocket 作業操作上的 SSM Agent 無法使用 Systems Manager 命令文件 `AWS-UpdateSSMAgent` 更新。更新是在 Bottlerocket 控制容器中進行管理。如需詳細資訊，請參閱 GitHub 上的 Bottlerocket 控制容器和 Bottlerocket 更新運算子。

macOS 版本要求

如果執行個體執行的是 macOS 11.0 (Big Sur) 版或更新版本，則執行個體必須具備 SSM Agent 3.1.941.0 版或更新版本才能執行 AWS-UpdateSSMAgent 文件。如果執行個體執行的是 3.1.941.0 之前發行的 SSM Agent 版本，則透過執行 SSM Agent 和 AWS-UpdateSSMAgent 命令來更新 `brew update` 以執行 `brew upgrade amazon-ssm-agent`。

方法	詳細資訊
在所有受管節點上按一下自動更新 (建議)	您可以設定 中的所有受管節點 AWS 帳戶 ，以自動檢查和下載新版本的 SSM Agent。若要這麼做，請在 Fleet Manager 的設定索引標籤中選擇自動更新 SSM Agent，如本主題稍後所述。
全域或選擇性更新	您可以使用 中的工具 State Manager來建立關聯 AWS Systems Manager，以自動下載並安裝在受管節點SSM Agent上。如果您想要限制對工作負載的中斷，可以建立 Systems Manager 維護時段，以在指定的時段內執行安裝。這兩個方法都允許您為所有受管節點或選擇性地選擇要更新的執行個體，來建立全域更新組態。如需有關建立 State Manager 關聯的詳細資訊，請參閱逐步解說：SSM Agent使用 自動更新 AWS CLI。如需建立維護時段的相關資訊，請參閱教學課程：使用主控台建立修補維護時段。
新環境的全域或選擇性更新	如果您開始使用 Systems Manager，建議您每兩週使用 中的更新 Systems Manager (SSM) 代理程式選項Quick Setup。 中的工具 AWS Systems ManagerQuick Setup可讓您為所有受管節點建立全域更新組態，或選擇性地選擇哪些受管節點要更新。如需詳細資訊，請參閱使用 Quick Setup 設定 Amazon EC2 主機管理。

如果您想要手動SSM Agent更新受管節點，您可以訂閱在發行新版本的代理程式時 AWS 發佈的通知。如需相關資訊，請參閱 訂閱 SSM Agent 通知。訂閱通知後，您可以使用 Run Command 將一個或多個受管節點手動更新為最新版本。如需詳細資訊，請參閱使用 Run Command 更新 SSM Agent。

自動更新 SSM Agent

您可以設定 Systems Manager，在您的 SSM Agent 中的所有 Linux 型和 Windows 型受管節點上自動更新 AWS 帳戶。如果您開啟此選項，則 Systems Manager 每兩週會自動檢查是否有新版本的代理程式。如果有新版本，則 Systems Manager 會使用 SSM 文件 `AWS-UpdateSSMAgent` 自動將代理程式更新為最新發行版本。建議您選擇此選項，以確保受管節點始終執行最新版本的 SSM Agent。

> **Note**
>
> 在使用 SSM 文件 `AWS-UpdateSSMAgent` 安裝或更新代理程式之後，如果您使用 `yum` 命令更新受管節點上的 SSM Agent，您可能會看到如下訊息：「Warning: RPMDB altered outside of yum. (警告：RPMDB 已變更超出 yum)。」 預期會出現此訊息，且可以安全忽略。

自動更新 SSM Agent

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Fleet Manager。
3. 選擇 Settings (設定) 標籤。
4. 在代理程式自動更新區域中，選擇自動更新 SSM Agent。

若要變更您的機群更新到 SSM Agent 的版本，請在 Settings (設定) 標籤的 Agent auto update (代理程式自動更新) 下選擇 Edit (編輯)。然後在 Parameters (參數) 下的 Version (版本) 中輸入您想要更新到的 SSM Agent 的版本編號。如果未指定，代理程式會更新到最新版本。

若要停止將 SSM Agent 的更新版本自動部署到您帳戶中的所有受管節點，請選擇 Settings (設定) 標籤中 Agent auto update (代理程式自動更新) 下的 Delete (刪除)。此動作會刪除會在您的受管節點上自動更新 SSM Agent 的 State Manager 關聯。

訂閱 SSM Agent 通知

Amazon Simple Notification Service (Amazon SNS) 可以在發行新版本的 AWS Systems Manager 代理程式 (SSM Agent) 時通知您。使用下列程序訂閱這些通知。

> **Tip**
>
> 您也可以觀看 GitHub 上的 SSM Agent 版本備註頁面來訂閱通知。

訂閱 SSM Agent 通知

1. 在 https://console.aws.amazon.com/sns/v3/home 開啟 Amazon SNS 主控台。
2. 從導覽列的區域選擇器中，選擇美國東部 (維吉尼亞北部) (如果尚未選取)。您必須選取此選項，AWS 區域 因為您訂閱SSM Agent的 Amazon SNS 通知只會從此區域產生。
3. 在導覽窗格中，選擇訂閱。
4. 選擇 Create subscription (建立訂閱)。
5. 針對 Create subscription (建立訂閱)，執行下列動作：
   a. 對於 Topic ARN (主題 ARN)，請使用下列 Amazon Resource Name (ARN)：

      ```
      arn:aws:sns:us-east-1:720620558202:SSM-Agent-Update
      ```

   b. 對於 Protocol (通訊協定)，請選擇 `Email` 或 `SMS`。
   c. 針對 Endpoint (端點)，依據您在前一步驟選擇 `Email` 或 `SMS`，輸入電子郵件地址或國家/地區代碼與號碼以接收通知。
   d. 選擇 Create subscription (建立訂閱)。
6. 如果您選擇 `Email`，您會收到請求確認訂閱的電子郵件訊息。開啟該訊息並遵循指示完成訂閱。

每當 SSM Agent 有新版本發佈時，我們將傳送通知給訂閱者。如果您不想再接收這些通知，請使用下列程序來取消訂閱。

取消訂閱 SSM Agent 通知

1. 開啟 Amazon SNS 主控台。
2. 在導覽窗格中，選擇 Subscriptions (訂閱)。
3. 選取訂閱，然後選擇 Delete (刪除)。出現確認提示時，請選擇 Delete (刪除)。

SSM Agent 疑難排解

如果您在受管節點上執行操作時遇到問題，則 AWS Systems Manager 代理程式 () 可能會發生問題 SSM Agent。使用以下資訊以協助檢視 SSM Agent 日誌檔並對代理程式的問題進行疑難排解。

主題

- SSM Agent 過期
- 使用 SSM Agent 日誌檔案診斷並解決問題
- 代理程式日誌檔案不會輪換 (Windows)
- 無法連線至 SSM 端點
- 驗證 VPC 組態
- 驗證 VPC DNS 相關屬性
- 驗證端點安全群組的相關輸入規則
- 使用 ssm-cli 診斷並解決受管節點的可用性問題

SSM Agent 過期

每當將新工具新增至 Systems Manager 或對現有工具進行更新時，SSM Agent就會發行更新的 版本。未使用最新版本的代理程式可能會使您的受管節點無法使用各種 Systems Manager 工具和功能。因此，我們建議您讓機器的 SSM Agent 自動保持最新狀態。如需相關資訊，請參閱 自動化 SSM Agent 更新。請訂閱 GitHub 上的 SSM Agent 版本備註頁面，以便接收有關 SSM Agent 更新的通知。

使用 SSM Agent 日誌檔案診斷並解決問題

SSM Agent 將資訊記錄在下列檔案中。這些檔案中的資訊也可協助您排除問題。如需有關 SSM Agent 日誌檔案的詳細資訊，包括如何開啟偵錯記錄，請參閱檢視 SSM Agent 日誌。

> **Note**
>
> 如果您選擇使用 Windows 檔案總管查看這些日誌，請務必在資料夾選項中檢視隱藏檔案和系統檔案。

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

對於 Linux 受管節點，您可能會在寫入到以下目錄的 messages 檔案中找到更多資訊：/var/log。

如需使用代理程式日誌診斷並解決問題的詳細資訊，請參閱 AWS re:Post 知識中心中的「如何使用 SSM Agent 日誌診斷並解決受管執行個體中的 SSM Agent 問題？」一文。

代理程式日誌檔案不會輪換 (Windows)

如果您在 seelog.xml 檔案中指定以日期為基礎的日誌檔案輪換 (在 Windows Server 受管節點上)，且日誌不會輪換，請指定 fullname=true 參數。以下是已指定 fullname=true 參數的 seelog.xml 組態檔案的範例。

```
<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>
```

無法連線至 SSM 端點

SSM Agent 必須允許 HTTPS (連接埠 443) 傳出流量至下列端點：

- ssm.*region*.amazonaws.com

- ssmmessages.*region*.amazonaws.com

region 代表 支援的 AWS 區域 識別符 AWS Systems Manager，例如us-east-2美國東部（俄亥俄）區域。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域欄位。

> **Note**
>
> 在 2024 年之前，也需要 ec2messages.*region*.amazonaws.com。對於 2024 年之前 AWS 區域 啟動的 ，仍然需要允許流量至 ，但 ssmmessages.*region*.amazonaws.com 是選用的ec2messages.*region*.amazonaws.com。
> 對於在 2024 年及之後啟動的區域，仍然需要允許 ssmmessages.*region*.amazonaws.com 的流量，但這些區域不支援 ec2messages.*region*.amazonaws.com 端點。

SSM Agent 如果無法與上述端點通訊，即使使用 Amazon Linux 2 或 Amazon Linux 2023 等 AWS 提供的 Amazon Machine Images(AMIs)，也無法運作。您的網路組態必須具有開放的網際網路存取權，或者您必須設定 Virtual Private Cloud (VPC) 端點。如果您不打算建立自訂 VPC 端點，請檢查您的網際網路閘道或 NAT 閘道。如需有關如何管理 VPC 端點的詳細資訊，請參閱使用適用於 Systems Manager 的 VPC 端點來改善 EC2 執行個體的安全性。

驗證 VPC 組態

為了使用 Systems Manager 管理 EC2 執行個體，必須針對 ssm.*region*.amazonaws.com、ssmmessages.*region*.amazonaws.com 正確設定 VPC 端點，有時還須針對 ec2messages.*region*.amazonaws.com (如本主題前面的無法連線至 SSM 端點部分所解釋) 正確設定 VPC 端點。您的網路組態必須具有開放的網際網路存取權，或者您必須設定這些虛擬私有雲端 (VPC) 端點。

若要對 VPC 端點的問題進行疑難排解，請執行下列動作：

- 確保在 VPC 層級包含 VPC 端點。如果在 VPC 上找不到具有特定服務名稱的 VPC 端點，請先驗證是否已在 VPC 層級啟用 DNS 支援。接下來，建立一個新的 VPC 端點，並將其與每個可用區域中的一個子網路建立關聯。
- 確保已在 VPC 端點層級啟用私有 DNS 名稱。私有 DNS 名稱預設已啟用，但可能已在某些時間點手動停用。
- 確保現有 VPC 端點與適當子網路相關聯。此外，確保 VPCE 已與該可用區域中的子網路相關聯。

如需詳細資訊，請參閱下列主題：

- 《AWS PrivateLink 指南》中的 Access an AWS 服務 using an interface VPC endpoint
- 《AWS PrivateLink 指南》中的 Associate a private DNS name
- 使用適用於 Systems Manager 的 VPC 端點來改善 EC2 執行個體的安全性

驗證 VPC DNS 相關屬性

在驗證 VPC 組態時，請確保已啟用 `enableDnsSupport` 與 `enableDnsHostnames` 屬性。

您可以使用 Amazon EC2 ModifyVPCAttribute API 動作或 AWS CLI 命令 modify-vpc-attribute 來啟用這些屬性。

如需有關在 Amazon VPC 主控台中啟用這些屬性的資訊，請參閱《Amazon VPC 使用者指南》中的 View and update DNS attributes for your VPC。

驗證端點安全群組的相關輸入規則

請確保您已設定的任何 VPC 端點 (`ssm`、`ssmmessages` 和 `ec2messages`) 在其安全群組上包含輸入規則，以便允許連接埠 443 的流量。如有必要，您可以使用輸入規則在 VPC 中建立新的安全群組，以便允許 VPC 的無類別域間路由 (CIDR) 區塊在連接埠 443 的流量。建立安全群組之後，請將其附加至每個 VPC 端點。

如需詳細資訊，請參閱下列主題：

- 如何建立 VPC 端點，以便使用 Systems Manager 來管理私有 EC2 執行個體，無需網際網路存取？re AWS ：Post 上的
- 《Amazon VPC 使用者指南》中的 VPC CIDR blocks

使用 `ssm-cli` 診斷並解決受管節點的可用性問題

從 SSM Agent 3.1.501.0 版開始，您可以使用 `ssm-cli` 判斷受管理節點是否滿足由 Systems Manager 管理的主要要求，以及是否顯示在 Fleet Manager 中的受管節點清單中。`ssm-cli` 是獨立的命令列工具，包含在 SSM Agent 安裝中。包含了預先設定的命令，收集所需資訊，協助您診斷為何您確認正在執行的 Amazon EC2 執行個體或非 EC2 機器未包含在 Systems Manager 的受管節點清單中。這些命令會在您指定 `get-diagnostics` 選項時執行。

如需詳細資訊，請參閱使用 `ssm-cli` 診斷並解決受管節點的可用性問題。

AWS Systems Manager 中的安全性

雲端安全是 Amazon Web Services 最重視的一環。身為 AWS 客戶，您可以從資料中心和網路架構中受益，這些架構是為了滿足最安全敏感組織的需求而建置。

安全性是 AWS 和 之間的共同責任。共同責任模型將其描述為雲端的安全性和雲端中的安全性：

- 雲端的安全性 – AWS 負責保護在 AWS 服務 中執行的基礎設施 AWS 雲端。 AWS 也為您提供可安全使用的服務。第三方稽核人員會定期測試和驗證我們的安全有效性，做為AWS 合規計畫的一部分。若要了解適用於 的合規計劃AWS Systems Manager，請參閱AWS 服務 合規計劃範圍內。
- 雲端安全性 – 您的責任取決於您使用 AWS 服務 的 。您也必須對其他因素負責，包括資料的機密性、您公司的要求和適用法律和法規。

本文件可協助您了解如何在使用 時套用共同的責任模型 AWS Systems Manager。下列主題說明如何將 Systems Manager 設定為達到您的安全及法規遵循目標。您也會了解如何使用其他 AWS 服務 來協助您監控和保護 Systems Manager 資源。

主題

- AWS Systems Manager 中的資料保護
- 適用於 AWS Systems Manager 的 Identity and Access Management
- 使用 Systems Manager 的服務連結角色
- AWS Systems Manager 中的日誌記錄和監控
- AWS Systems Manager 的合規驗證
- AWS Systems Manager 中的恢復能力
- AWS Systems Manager 中的基礎設施安全
- AWS Systems Manager 中的組態與漏洞分析
- Systems Manager 的安全最佳實務

AWS Systems Manager 中的資料保護

資料保護是指在傳輸中 （ 往返 時Systems Manager) 和靜態 （ 存放於 AWS 資料中心時 ） 時保護資料。

AWS 共同的責任模型適用於 中的資料保護AWS Systems Manager。如此模型所述， AWS 負責保護執行所有 的 全球基礎設施 AWS 雲端。您負責維護在此基礎設施上託管內容的控制權。您也同時負

責所使用 AWS 服務 的安全組態和管理任務。如需資料隱私權的詳細資訊，請參閱資料隱私權常見問答集。如需有關歐洲資料保護的相關資訊，請參閱 AWS 安全性部落格上的 AWS 共同的責任模型和 GDPR 部落格文章。

基於資料保護目的，建議您保護 AWS 帳戶 登入資料，並使用 AWS IAM Identity Center 或 AWS Identity and Access Management (IAM) 設定個別使用者。如此一來，每個使用者都只會獲得授與完成其任務所必須的許可。我們也建議您採用下列方式保護資料：

- 每個帳戶均要使用多重要素驗證 (MFA)。
- 使用 SSL/TLS 與 AWS 資源通訊。我們需要 TLS 1.2 並建議使用 TLS 1.3。
- 使用 設定 API 和使用者活動記錄 AWS CloudTrail。如需有關使用 CloudTrail 追蹤擷取 AWS 活動的資訊，請參閱AWS CloudTrail 《 使用者指南》中的使用 CloudTrail 追蹤。
- 使用 AWS 加密解決方案，以及其中的所有預設安全控制 AWS 服務。
- 使用進階的受管安全服務 (例如 Amazon Macie)，協助探索和保護儲存在 Amazon S3 的敏感資料。
- 如果您在 AWS 透過命令列界面或 API 存取 時需要 FIPS 140-3 驗證的密碼編譯模組，請使用 FIPS 端點。如需有關 FIPS 和 FIPS 端點的更多相關資訊，請參閱聯邦資訊處理標準 (FIPS) 140-3。

我們強烈建議您絕對不要將客戶的電子郵件地址等機密或敏感資訊，放在標籤或自由格式的文字欄位中，例如名稱欄位。這包括當您使用 Systems Manager 或使用主控台、API AWS CLI或其他 AWS 服務 AWS SDKs 時。您在標籤或自由格式文字欄位中輸入的任何資料都可能用於計費或診斷日誌。如果您提供外部伺服器的 URL，我們強烈建議請勿在驗證您對該伺服器請求的 URL 中包含憑證資訊。

資料加密

靜態加密

Parameter Store 參數

您可以在 中建立的參數類型Parameter Store、 中的工具 AWS Systems Manager，包括 `String`、`StringList`和 `SecureString`。

所有參數，無論其類型為何，都會在傳輸中和靜態時加密。在傳輸中，參數會使用傳輸層安全性 (TLS) 加密，為 API 請求建立安全的 HTTPS 連線。靜態時，它們會使用 AWS 擁有的金鑰 in AWS Key Management Service () 加密AWS KMS。如需 AWS 擁有的金鑰 加密的詳細資訊，請參閱《 AWS Key Management Service 開發人員指南AWS 擁有的金鑰》中的 。

`SecureString` 類型提供額外的加密選項，建議用於所有敏感資料。您可以選擇下列類型的 AWS KMS 金鑰來加密和解密`SecureString`參數的值：

- AWS 受管金鑰 您帳戶的
- 您在帳戶中建立的客戶受管金鑰 (CMK)
- 另一個 中與您共用 AWS 帳戶 的 CMK

如需 AWS KMS 加密的詳細資訊，請參閱 AWS Key Management Service 開發人員指南。

S3 儲存貯體中的內容

做為 Systems Manager 操作的一部分，您可以選擇將資料上傳或存放在一或多個 Amazon Simple Storage Service (Amazon S3) 儲存貯體中。

如需 S3 儲存貯體加密的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的使用加密保護資料和 Amazon Simple Storage Service (Amazon S3) 的資料保護。

以下是您可以在 Systems Manager 活動中上傳或存放在 S3 儲存貯體中的資料類型。

- 中的命令輸出Run Command，這是 中的工具 AWS Systems Manager
- 中的套件Distributor， 中的工具 AWS Systems Manager
- 在 中修補操作日誌Patch Manager，這是 中的工具 AWS Systems Manager
- Patch Manager 修補程式會覆寫清單
- 要在 Automation 中的 Runbook 工作流程中執行的指令碼或Ansible手冊，這是 中的工具 AWS Systems Manager
- Chef InSpec 描述檔，可在 的 Compliance 中使用掃描，這是 中的工具 AWS Systems Manager
- AWS CloudTrail 日誌
- 中的工作階段歷史記錄日誌Session Manager， 中的工具 AWS Systems Manager
- 來自 的報告Explorer，這是 中的工具 AWS Systems Manager
- OpsData 來自 OpsCenter，這是 中的工具 AWS Systems Manager
- AWS CloudFormation 用於自動化工作流程的 範本
- 來自資源資料同步掃描的合規資料
- 在 中建立或編輯關聯的請求輸出State Manager，這是 中受管節點 AWS Systems Manager的工具
- 自訂 Systems Manager 文件 (SSM 文件)，您可以使用 AWS 受管 SSM 文件 `AWS-RunDocument` 執行

CloudWatch Logs 日誌群組

做為 Systems Manager 操作的一部分，您可以選擇將資料串流至一或多個 Amazon CloudWatch Logs 日誌群組。

如需 CloudWatch Logs 日誌群組加密的相關資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的在 CloudWatch Logs 中使用 AWS Key Management Service加密日誌資料。

下列是您可能已在 Systems Manager 活動中串流至 CloudWatch Logs 日誌群組的資料類型。

- Run Command 命令的輸出
- 使用 Automation Runbook 中的 `aws:executeScript` 動作執行指令碼輸出
- Session Manager 工作階段歷史記錄日誌
- 受管節點上來自 SSM Agent 的日誌

傳輸中加密

我們建議您使用 Transport Layer Security (TLS) 此類加密通訊協定，在用戶端和您的節點之間加密傳輸中的敏感資料。

Systems Manager 提供下列對傳輸中資料進行加密的支援。

Systems Manager API 端點的連線

Systems Manager API 端點僅支援透過 HTTPS 的安全連線。當您使用 AWS Management Console、 AWS SDK 或 Systems Manager API 管理 Systems Manager 資源時，所有通訊都會使用 Transport Layer Security (TLS) 加密。如需完整的 API 端點清單，請參閱《Amazon Web Services 一般參考》中的 AWS 服務 端點。

受管執行個體

AWS 在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體之間提供安全且私有的連線。此外，我們使用 AEAD 演算法搭配 256 位元加密，在相同 Virtual Private Cloud (VPC) 或對等 VPC 中的受支援執行個體之間自動加密傳輸中流量。此加密功能使用基礎硬體的卸載能力，不影響網路效能。受支援執行個體包括：C5n、G4、I3en、M5dn、M5n、P3dn、R5dn 和 R5n。

Session Manager 工作階段

依預設，Session Manager 使用 TLS 1.3 對帳戶中使用者的本機電腦與 EC2 執行個體之間傳輸的工作階段資料進行加密。您也可以選擇使用已在 中建立 AWS KMS key 的 進一步加密傳輸中的資料 AWS KMS。 AWS KMS 加密可用於 `Standard_Stream`、 `InteractiveCommands`和 `NonInteractiveCommands`工作階段類型。

Run Command 存取

依預設，使用 Run Command 來遠端存取您的節點是使用 TLS 1.3 進行加密，然後使用 SigV4 來簽署建立連線的請求。

網際網路流量隱私權

您可以使用 Amazon Virtual Private Cloud (Amazon VPC) 在受管節點的資源之間建立邊界，並控制這些資源、內部部署網路和網際網路之間的流量。如需詳細資訊，請參閱使用適用於 Systems Manager 的 VPC 端點來改善 EC2 執行個體的安全性。

如需 Amazon Virtual Private Cloud 安全的詳細資訊，請參閱《Amazon VPC 使用者指南》中的 Amazon VPC 網際網路流量隱私權。

適用於 AWS Systems Manager 的 Identity and Access Management

AWS Identity and Access Management (IAM) 是一種 AWS 服務 ，可協助管理員安全地控制對 AWS 資源的存取。IAM 管理員可以控制身分身分驗證 (已登入) 和授權 (具有許可) 以使用 Systems Manager 資源。IAM 是您可以免費使用 AWS 服務 的 。

主題

- 目標對象
- 使用身分驗證
- 使用政策管理存取權
- AWS Systems Manager 搭配 IAM 的運作方式
- AWS Systems Manager 身分型政策範例
- AWS 的 受管政策 AWS Systems Manager
- 對 AWS Systems Manager 身分與存取進行疑難排解

目標對象

AWS Identity and Access Management (IAM) 的使用方式會有所不同，取決於您在 中執行的工作 Systems Manager。

服務使用者：如果使用 Systems Manager 執行任務，管理員會為您提供所需的憑證和許可。隨著您為了執行作業而使用的 Systems Manager 功能數量變多，您可能會需要額外的許可。了解存取的管理方式可協助您向管理員請求正確的許可。若您無法存取 Systems Manager 中的某項功能，請參閱 對 AWS Systems Manager 身分與存取進行疑難排解。

服務管理員：如果您負責公司內的 Systems Manager 資源，您可能具備 Systems Manager 的完整存取權限。您的任務是判斷服務使用者應存取的 Systems Manager 功能及資源。接著，您必須將請求提交給您的 IAM 管理員，來變更您服務使用者的許可。檢閱此頁面上的資訊，了解 IAM 的基本概念。若要進一步了解貴公司可搭配 Systems Manager 使用 IAM 的方式，請參閱 AWS Systems Manager 搭配 IAM 的運作方式。

IAM 管理員：如果您是 IAM 管理員，建議您掌握如何撰寫政策以管理 Systems Manager 存取權的詳細資訊。若要檢視您可以在 IAM 中使用的範例 Systems Manager 身分型政策，請參閱 AWS Systems Manager 身分型政策範例。

使用身分驗證

身分驗證是您 AWS 使用身分憑證登入 的方式。您必須以 AWS 帳戶根使用者身分、IAM 使用者身分或擔任 IAM 角色來驗證 （ 登入 AWS)。

您可以使用透過身分來源提供的憑證，以聯合身分 AWS 身分身分身分身分登入 。 AWS IAM Identity Center (IAM Identity Center) 使用者、您公司的單一登入身分驗證，以及您的 Google 或 Facebook 登入資料，都是聯合身分的範例。您以聯合身分登入時，您的管理員先前已設定使用 IAM 角色的聯合身分。當您使用聯合 AWS 身分存取 時，您會間接擔任角色。

根據您身分的使用者類型，您可以登入 AWS Management Console 或 AWS 存取入口網站。如需登入的詳細資訊 AWS，請參閱AWS 登入 《 使用者指南》中的如何登入您的 AWS 帳戶 。

如果您以 AWS 程式設計方式存取 ， AWS 會提供軟體開發套件 (SDK) 和命令列界面 (CLI)，以使用您的 憑證以密碼編譯方式簽署您的請求。如果您不使用 AWS 工具，則必須自行簽署請求。如需使用建議的方法自行簽署請求的詳細資訊，請參閱《IAM 使用者指南》中的適用於 API 請求的AWS Signature 第 4 版。

無論您使用何種身分驗證方法，您可能都需要提供額外的安全性資訊。例如， AWS 建議您使用多重驗證 (MFA) 來提高帳戶的安全性。如需更多資訊，請參閱《AWS IAM Identity Center 使用者指南》中的多重要素驗證和《IAM 使用者指南》中的 IAM 中的AWS 多重要素驗證。

AWS 帳戶 根使用者

當您建立 時 AWS 帳戶，您會從一個登入身分開始，該身分可以完整存取帳戶中的所有 AWS 服務 和資源。此身分稱為 AWS 帳戶 Theroot 使用者，可透過使用您用來建立帳戶的電子郵件地址和密碼登入

來存取。強烈建議您不要以根使用者處理日常任務。保護您的根使用者憑證，並將其用來執行只能由根使用者執行的任務。如需這些任務的完整清單，了解需以根使用者登入的任務，請參閱《IAM 使用者指南》中的需要根使用者憑證的任務。

IAM 使用者和群組

IAM 使用者是 中的身分 AWS 帳戶 ，具有單一人員或應用程式的特定許可。建議您盡可能依賴臨時憑證，而不是擁有建立長期憑證 (例如密碼和存取金鑰) 的 IAM 使用者。但是如果特定使用案例需要擁有長期憑證的 IAM 使用者，建議您輪換存取金鑰。如需更多資訊，請參閱 IAM 使用者指南中的為需要長期憑證的使用案例定期輪換存取金鑰。

IAM 群組是一種指定 IAM 使用者集合的身分。您無法以群組身分簽署。您可以使用群組來一次為多名使用者指定許可。群組可讓管理大量使用者許可的程序變得更為容易。例如，您可以擁有一個名為 IAMAdmins 的群組，並給予該群組管理 IAM 資源的許可。

使用者與角色不同。使用者只會與單一人員或應用程式建立關聯，但角色的目的是在由任何需要它的人員取得。使用者擁有永久的長期憑證，但角色僅提供臨時憑證。如需更多資訊，請參閱《IAM 使用者指南》中的 IAM 使用者的使用案例。

IAM 角色

IAM 角色是 中具有特定許可 AWS 帳戶 的身分。它類似 IAM 使用者，但不與特定的人員相關聯。若要暫時在 中擔任 IAM 角色 AWS Management Console，您可以從使用者切換至 IAM 角色 （主控台）。您可以透過呼叫 AWS CLI 或 AWS API 操作或使用自訂 URL 來擔任角色。如需使用角色的方法詳細資訊，請參閱《IAM 使用者指南》中的擔任角色的方法。

使用臨時憑證的 IAM 角色在下列情況中非常有用：

- 聯合身分使用者存取 — 如需向聯合身分指派許可，請建立角色，並為角色定義許可。當聯合身分進行身分驗證時，該身分會與角色建立關聯，並獲授予由角色定義的許可。如需有關聯合角色的相關資訊，請參閱《IAM 使用者指南》中的為第三方身分提供者 (聯合) 建立角色。如果您使用 IAM Identity Center，則需要設定許可集。為控制身分驗證後可以存取的內容，IAM Identity Center 將許可集與 IAM 中的角色相關聯。如需有關許可集的資訊，請參閱 AWS IAM Identity Center 使用者指南中的許可集。
- 暫時 IAM 使用者許可 – IAM 使用者或角色可以擔任 IAM 角色來暫時針對特定任務採用不同的許可。
- 跨帳戶存取權：您可以使用 IAM 角色，允許不同帳戶中的某人 (信任的主體) 存取您帳戶的資源。角色是授予跨帳戶存取權的主要方式。不過，對於某些 AWS 服務，您可以直接將政策連接到資源（而不是使用角色做為代理）。如需了解使用角色和資源型政策進行跨帳戶存取之間的差異，請參閱《IAM 使用者指南》中的 IAM 中的跨帳戶資源存取。

- 跨服務存取 – 有些 AWS 服務 使用其他 中的功能 AWS 服務。例如，當您在服務中進行呼叫時，該服務通常會在 Amazon EC2 中執行應用程式或將物件儲存在 Amazon Simple Storage Service (Amazon S3) 中。服務可能會使用呼叫主體的許可、使用服務角色或使用服務連結角色來執行此作業。
  - 轉送存取工作階段 (FAS) – 當您使用 IAM 使用者或角色在 中執行動作時 AWS，您會被視為委託人。使用某些服務時，您可能會執行某個動作，進而在不同服務中啟動另一個動作。FAS 使用呼叫 的委託人許可 AWS 服務，結合 AWS 服務 請求向下游服務提出請求。只有在服務收到需要與其他 AWS 服務 或 資源互動才能完成的請求時，才會提出 FAS 請求。在此情況下，您必須具有執行這兩個動作的許可。如需提出 FAS 請求時的政策詳細資訊，請參閱《轉發存取工作階段》。
  - 服務角色 – 服務角色是服務擔任的 IAM 角色，可代表您執行動作。IAM 管理員可以從 IAM 內建立、修改和刪除服務角色。如需詳細資訊，請參閱《IAM 使用者指南》中的建立角色以委派許可權給 AWS 服務。
  - 服務連結角色 – 服務連結角色是一種連結至 的服務角色類型 AWS 服務。服務可以擔任代表您執行動作的角色。服務連結角色會出現在您的 中 AWS 帳戶 ，並由服務擁有。IAM 管理員可以檢視，但不能編輯服務連結角色的許可。
- 在 Amazon EC2 上執行的應用程式 – 您可以使用 IAM 角色來管理在 EC2 執行個體上執行之應用程式的臨時登入資料，以及提出 AWS CLI 或 AWS API 請求。這是在 EC2 執行個體內儲存存取金鑰的較好方式。若要將 AWS 角色指派給 EC2 執行個體，並將其提供給其所有應用程式，您可以建立連接至執行個體的執行個體描述檔。執行個體設定檔包含該角色，並且可讓 EC2 執行個體上執行的程式取得臨時憑證。如需詳細資訊，請參閱《IAM 使用者指南》中的使用 IAM 角色來授予許可權給 Amazon EC2 執行個體上執行的應用程式。

使用政策管理存取權

您可以透過建立政策並將其連接到身分或資源 AWS 來控制 AWS 中的存取。政策是 中的物件，當與身分或資源相關聯時， AWS 會定義其許可。當委託人（使用者、根使用者或角色工作階段）發出請求時， 會 AWS 評估這些政策。政策中的許可決定是否允許或拒絕請求。大多數政策會以 JSON 文件 AWS 的形式存放在 中。如需 JSON 政策文件結構和內容的詳細資訊，請參閱 IAM 使用者指南中的 JSON 政策概觀。

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

預設情況下，使用者和角色沒有許可。若要授予使用者對其所需資源執行動作的許可，IAM 管理員可以建立 IAM 政策。然後，管理員可以將 IAM 政策新增至角色，使用者便能擔任這些角色。

IAM 政策定義該動作的許可，無論您使用何種方法來執行操作。例如，假設您有一個允許 `iam:GetRole` 動作的政策。具有該政策的使用者可以從 AWS Management Console AWS CLI、 或 API AWS 取得角色資訊。

身分型政策

身分型政策是可以附加到身分 (例如 IAM 使用者、使用者群組或角色) 的 JSON 許可政策文件。這些政策可控制身分在何種條件下能對哪些資源執行哪些動作。如需了解如何建立身分型政策，請參閱《IAM 使用者指南》中的透過客戶管理政策定義自訂 IAM 許可。

身分型政策可進一步分類成內嵌政策或受管政策。內嵌政策會直接內嵌到單一使用者、群組或角色。受管政策是獨立的政策，您可以連接到 中的多個使用者、群組和角色 AWS 帳戶。受管政策包括 AWS 受管政策和客戶受管政策。如需了解如何在受管政策及內嵌政策之間選擇，請參閱《IAM 使用者指南》中的在受管政策和內嵌政策間選擇。

如需 AWS 受管政策的相關資訊Systems Manager，請參閱 AWS Systems Manager 受管政策。

資源型政策

資源型政策是連接到資源的 JSON 政策文件。資源型政策的最常見範例是 IAM 角色信任政策和 Amazon S3 儲存貯體政策。在支援資源型政策的服務中，服務管理員可以使用它們來控制對特定資源的存取權限。對於附加政策的資源，政策會定義指定的主體可以對該資源執行的動作以及在何種條件下執行的動作。您必須在資源型政策中指定主體。委託人可以包含帳戶、使用者、角色、聯合身分使用者或 AWS 服務。

資源型政策是位於該服務中的內嵌政策。您無法在資源型政策中使用來自 IAM 的 AWS 受管政策。

政策條件索引鍵

使用者和角色可執行的動作，以及他們可以採取這些動作的資源，都可以透過特定條件進一步限制。

在 JSON 政策文件中，`Condition` 元素 (或 `Condition` 區塊) 可讓您指定使陳述式生效的條件。`Condition` 元素是選用項目。您可以建立使用條件運算子的條件運算式 (例如 `StringEquals` 或 `StringNotLike`)，來比對政策中的條件和請求中的值。

若您在陳述式中指定多個 `Condition` 元素，或是在單一 `Condition` 元素中指定多個索引鍵， AWS 會使用邏輯 AND 操作評估他們。如果您為單一條件索引鍵指定多個值， 會使用邏輯OR操作 AWS 評估條件。必須符合所有條件，才會授與陳述式的許可。

您也可以在指定條件時使用預留位置變數。例如，您可以只在使用者使用其 IAM 使用者名稱標記時，將存取資源的許可授予該 IAM 使用者。如需更多資訊，請參閱 IAM 使用者指南中的 IAM 政策元素：變數和標籤。

AWS 支援全域條件金鑰和服務特定的條件金鑰。如需詳細資訊，請參閱《IAM 使用者指南》中的AWS 全域條件內容金鑰。

> **Important**
>
> 如果您使用 Systems Manager Automation，建議您不要在政策中使用 aws：SourceIp 條件金鑰。此條件索引鍵的行為取決於多個因素，包括是否提供 Automation Runbook 執行的 IAM 角色，以及 Runbook 中使用的自動化動作。因此， 條件索引鍵可能會產生非預期的行為。因此，我們建議您不要使用它。

Systems Manager 支援許多自身的條件索引鍵。如需詳細資訊，請參閱《服務授權參考》中的 Condition Keys for AWS Systems Manager。可以與 Systems Manager 特定條件索引鍵搭配使用的動作和資源，都列在《服務授權參考》中的 Resource types defined by AWS Systems Manager 內。

如果政策必須依賴 Systems Manager 服務所擁有的服務主體名稱，建議您使用 `aws:PrincipalServiceNamesList` 多重值條件索引鍵 (而不是 `aws:PrincipalServiceName` 條件索引鍵) 來檢查其是否存在。`aws:PrincipalServiceName` 條件索引鍵只包含服務主體名稱清單中的一個項目，不一定是您預期的服務主體名稱。以下 `Condition` 區塊示範如何檢查 `ssm.amazonaws.com` 是否存在。

```
{
    "Condition": {
        "ForAnyValue:StringEquals": {
            "aws:PrincipalServiceNamesList": "ssm.amazonaws.com"
        }
    }
}
```

如需檢視 Systems Manager 身分型政策範例，請參閱 AWS Systems Manager 身分型政策範例。

存取控制清單 (ACL)

存取控制清單 (ACL) 可控制哪些主體 (帳戶成員、使用者或角色) 擁有存取某資源的許可。ACL 類似於資源型政策，但它們不使用 JSON 政策文件格式。

Amazon S3 AWS WAF和 Amazon VPC 是支援 ACLs的服務範例。如需進一步了解 ACL，請參閱 Amazon Simple Storage Service 開發人員指南中的存取控制清單 (ACL) 概觀。

其他政策類型

AWS 支援其他較不常見的政策類型。這些政策類型可設定較常見政策類型授予您的最大許可。

- 許可界限 – 許可範圍是一種進階功能，可供您設定身分型政策能授予 IAM 實體 (IAM 使用者或角色) 的最大許可。您可以為實體設定許可界限。所產生的許可會是實體的身分型政策和其許可界限的交集。會在 `Principal` 欄位中指定使用者或角色的資源型政策則不會受到許可界限限制。所有這類政策中的明確拒絕都會覆寫該允許。如需許可界限的詳細資訊，請參閱 IAM 使用者指南中的 IAM 實體許可界限。
- 服務控制政策 SCPs) – SCPs是 JSON 政策，可指定 中組織或組織單位 (OU) 的最大許可 AWS Organizations。 AWS Organizations 是一種服務，用於分組和集中管理您企業擁有 AWS 帳戶 的多個 。若您啟用組織中的所有功能，您可以將服務控制政策 (SCP) 套用到任何或所有帳戶。SCP 會限制成員帳戶中實體的許可，包括每個實體 AWS 帳戶根使用者。如需 Organizations 和 SCP 的詳細資訊，請參閱《AWS Organizations 使用者指南》中的服務控制政策。
- 資源控制政策 (RCP) - RCP 是 JSON 政策，可用來設定您帳戶中資源的可用許可上限，採取這種方式就不需要更新附加至您所擁有的每個資源的 IAM 政策。RCP 會限制成員帳戶中資源的許可，並可能影響身分的有效許可，包括 AWS 帳戶根使用者，無論它們是否屬於您的組織。如需 Organizations 和 RCPs的詳細資訊，包括 AWS 服務 支援 RCPs 清單，請參閱AWS Organizations 《 使用者指南》中的資源控制政策 (RCPs)。
- 工作階段政策 – 工作階段政策是一種進階政策，您可以在透過撰寫程式的方式建立角色或聯合使用者的暫時工作階段時，做為參數傳遞。所產生工作階段的許可會是使用者或角色的身分型政策和工作階段政策的交集。許可也可以來自資源型政策。所有這類政策中的明確拒絕都會覆寫該允許。如需詳細資訊，請參閱 IAM 使用者指南中的工作階段政策。

多種政策類型

將多種政策類型套用到請求時，其結果形成的許可會更為複雜、更加難以理解。若要了解如何 AWS 在涉及多種政策類型時決定是否允許請求，請參閱《IAM 使用者指南》中的政策評估邏輯。

AWS Systems Manager 搭配 IAM 的運作方式

在您使用 AWS Identity and Access Management (IAM) 來管理對 的存取之前 AWS Systems Manager，您應該了解哪些 IAM 功能可與 搭配使用Systems Manager。若要深入了解 Systems Manager和其他 如何與 IAM AWS 服務 搭配使用，請參閱《AWS 服務 IAM 使用者指南》中的 與 IAM 搭配使用。

主題

- Systems Manager 身分型政策
- Systems Manager 資源型政策
- 以 Systems Manager 標籤為基礎的授權
- Systems Manager IAM 角色

Systems Manager 身分型政策

使用 IAM 身分型政策，您可以指定允許或拒絕的動作和資源，以及在何種條件下允許或拒絕動作。Systems Manager 支援特定動作、資源及條件金鑰。若要了解您在 JSON 政策中使用的所有元素，請參閱《IAM 使用者指南》中的 IAM JSON 政策元素參考。

動作

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。也就是說，哪個主體在什麼条件下可以對什麼資源執行哪些動作。

JSON 政策的 `Action` 元素描述您可以用來允許或拒絕政策中存取的動作。政策動作通常具有與相關聯 AWS API 操作相同的名稱。有一些例外狀況，例如沒有相符的 API 操作的僅限許可動作。也有一些作業需要政策中的多個動作。這些額外的動作稱為相依動作。

政策會使用動作來授予執行相關聯動作的許可。

Systems Manager 中的政策動作會在動作之前使用以下字首：`ssm:`。例如，若要授予某人使用 Systems Manager `PutParameter` API 操作建立 Systems Manager 參數 (SSM 參數) 的許可，請在其政策中加入 `ssm:PutParameter` 動作。政策陳述式必須包含 `Action` 或 `NotAction` 元素。Systems Manager 會定義一組自己的動作，來描述您可以使用此服務執行的任務。

若要在單一陳述式中指定多個動作，請用逗號分隔，如下所示：

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

> **Note**
>
> 中的下列工具在動作之前 AWS Systems Manager 使用不同的字首。
>
> - AWS AppConfig 在動作`appconfig:`之前使用 字首。

- Incident Manager 會在動作之前使用 `ssm-incidents:` 或 `ssm-contacts:` 字首。
- Systems Manager GUI Connect 會在動作之前使用 `ssm-guiconnect:` 字首。
- Quick Setup 會在動作之前使用 `ssm-quicksetup:` 字首。

您也可以使用萬用字元 (*) 來指定多個動作。例如，若要指定開頭是 `Describe` 文字的所有動作，請包含以下動作：

```
"Action": "ssm:Describe*"
```

如要查看 Systems Manager 動作的清單，請參閱《服務授權參考》中的 AWS Systems Manager 定義的動作。

資源

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

`Resource` JSON 政策元素可指定要套用動作的物件。陳述式必須包含 `Resource` 或 `NotResource` 元素。最佳實務是使用其 Amazon Resource Name (ARN) 來指定資源。您可以針對支援特定資源類型的動作 (稱為資源層級許可) 來這麼做。

對於不支援資源層級許可的動作 (例如列出操作)，請使用萬用字元 (*) 來表示陳述式適用於所有資源。

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

例如，Systems Manager 維護時段資源具有下列 ARN 格式。

```
arn:aws:ssm:region:account-id:maintenancewindow/window-id
```

若要在您美國東部 (俄亥俄) 區域的陳述式中指定 mw-0c50858d01EXAMPLE 維護時段，請使用類似於下列的 ARN。

```
"Resource": "arn:aws:ssm:us-east-2:123456789012:maintenancewindow/mw-0c50858d01EXAMPLE"
```

若要指定所有屬於特定帳戶的維護時段，請使用萬用字元 (*)。

```
"Resource": "arn:aws:ssm:region:123456789012:maintenancewindow/*"
```

對於 Parameter Store API 操作，您可以使用階層式名稱和 AWS Identity and Access Management (IAM) 政策，提供或限制存取階層的一個層級中的所有參數，如下所示。

```
"Resource": "arn:aws:ssm:region:123456789012:parameter/Dev/ERP/Oracle/*"
```

有些 Systems Manager 動作 (例如用來建立資源的動作) 無法在特定資源上執行。在這些情況下，您必須使用萬用字元 (*)。

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

一些 Systems Manager API 作業都接受多個資源。若要在單一陳述式中指定多項資源，請用逗號分隔其 ARN，如下所示。

```
"Resource": [
      "resource1",
      "resource2"
```

> **Note**
>
> 在 ARNs 中，大多數 AWS 服務 會將冒號 (:) 或正斜線 (/) 視為相同的字元。不過，Systems Manager 在資源模式和規則中請求完全相符。在建立事件模式時，請務必使用正確的 ARN 字元，使這些字元符合資源的 ARN。

下表說明 Systems Manager 支援的資源類型 ARN 格式。

> **Note**
>
> 請注意 ARN 格式的下列例外狀況。
>
> - 中的下列工具在動作之前 AWS Systems Manager 使用不同的字首。
>   - AWS AppConfig 在動作`appconfig:`之前使用 字首。
>   - Incident Manager 會在動作之前使用 `ssm-incidents:` 或 `ssm-contacts:` 字首。
>   - Systems Manager GUI Connect 會在動作之前使用 `ssm-guiconnect` 字首。
> - Amazon 擁有的文件和自動化定義資源，以及 Amazon 和第三方來源提供的公有參數，都不會在 ARN 格式中包含帳戶 ID。例如：
>   - SSM 文件 `AWS-RunPatchBaseline`：

```
arn:aws:ssm:us-east-2::document/AWS-RunPatchBaseline
```

- Automation 執行手冊 `AWS-ConfigureMaintenanceWindows`：

  ```
  arn:aws:ssm:us-east-2::automation-definition/AWS-ConfigureMaintenanceWindows
  ```

- 公有參數 `/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
  ```

如需有關這三個資源類型的詳細資訊，請參閱下列主題：

- 使用文件
- 執行由 Systems Manager Automation 提供支援的自動化操作
- 在 Parameter Store 中使用公有參數

- Quick Setup 會在動作之前使用 `ssm-quicksetup:` 字首。

資源類型	ARN 格式
應用程式 (AWS AppConfig)	arn:aws:appconfig:*region*:*account-id* :application/*application-id*
關聯	arn:aws:ssm:*region*:*account-id* :association/*association-id*
自動化執行	arn:aws:ssm:*region*:*account-id* :automation-execution/*automation-execution-id*
自動化定義 (包含版本子資源)	arn:aws:ssm:*region*:*account-id* :automation-definition/*automation-definition-id* :*version-id* [1]
組態描述檔 (AWS AppConfig)	arn:aws:appconfig:*region*:*account-id* :application/*application-id* /configurationprofile/*configurationprofile-id*

資源類型	ARN 格式
聯絡 (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*
文件	arn:aws:ssm:*region*:*account-id* :document/*document-name*
環境 (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*
Maintenance window (維護時段)	arn:aws:ssm:*region*:*account-id* :maintenancewindow/*window-id*
受管節點	arn:aws:ssm:*region*:*account-id* :managed-instance/*managed-node-id*
受管節點庫存	arn:aws:ssm:*region*:*account-id* :managed-instance-inventory/*managed-node-id*
OpsItem	arn:aws:ssm:*region*:*account-id* :opsitem/*OpsItem-id*
參數	單層級參數： • arn:aws:ssm:*region*:*account-id* :parameter/*parameter-name*/ 使用階層結構命名的參數： • arn:aws:ssm:*region*:*account-id* :parameter/*parameter-name-root* /*level-2*/*level-3*/*level-4*/*level-5* [2]

資源類型	ARN 格式
修補基準	arn:aws:ssm:*region*:*account-id* :patchbaseline/*patch-baseline-id*
反應計劃	arn:aws:ssm-incidents:*region*:*account-id* :response-plan/*response-plan-name*
Session (工作階段)	arn:aws:ssm:*region*:*account-id* :session/*session-id* [3]
所有 Systems Manager 資源	arn:aws:ssm:*
指定的 擁有的所有Systems Manager資源 AWS 帳戶 AWS 區域	arn:aws:ssm:*region*:*account-id* :*

[1]

針對自動化定義，Systems Manager 支援第二層級的資源，版本 ID。在 中 AWS，這些第二級資源稱為子資源。指定自動化定義資源的版本子資源，允許您提供特定版本之自動化定義的存取。例如，您可能想要確保您的節點管理僅使用最新版本的自動化定義。

[2]

若要整理和管理參數，您可以建立具有階層結構的參數名稱。擁有階層結構的參數名稱可以包含您使用斜線定義的路徑。您最多可以命名包含十五個層級的參數資源。建議您建立階層以反映環境中現有的階層架構。如需詳細資訊，請參閱在 Systems Manager 中建立 Parameter Store 參數。

[3]

在大多數情況下，工作階段 ID 是使用開始工作階段的帳戶使用者 ID，加上英數尾碼。例如：

```
arn:aws:us-east-2:111122223333:session/JohnDoe-1a2b3c4sEXAMPLE
```

不過，如果使用者 ID 無法使用，則會以下列方式建構 ARN：

```
arn:aws:us-east-2:111122223333:session/session-1a2b3c4sEXAMPLE
```

如需有關 ARN 格式的詳細資訊，請參閱《Amazon Web Services 一般參考》中的 Amazon Resource Name (ARN)。

如需 Systems Manager 資源類型及其 ARN 的清單，請參閱《服務授權參考》中的 AWS Systems Manager 定義的資源。若要了解您可以使用哪些動作指定每個資源的 ARN，請參閱 AWS Systems Manager 定義的動作。

Systems Manager 的條件索引鍵

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

Condition 元素 (或 Condition 區塊) 可讓您指定使陳述式生效的條件。Condition 元素是選用項目。您可以建立使用條件運算子的條件運算式 (例如等於或小於)，來比對政策中的條件和請求中的值。

若您在陳述式中指定多個 Condition 元素，或是在單一 Condition 元素中指定多個索引鍵， AWS 會使用邏輯 AND 操作評估他們。如果您為單一條件索引鍵指定多個值， 會使用邏輯OR操作 AWS 評估條件。必須符合所有條件，才會授與陳述式的許可。

您也可以在指定條件時使用預留位置變數。例如，您可以只在使用者使用其 IAM 使用者名稱標記時，將存取資源的許可授予該 IAM 使用者。如需更多資訊，請參閱 IAM 使用者指南中的 IAM 政策元素：變數和標籤。

AWS 支援全域條件金鑰和服務特定的條件金鑰。若要查看所有 AWS 全域條件索引鍵，請參閱《IAM 使用者指南》中的AWS 全域條件內容索引鍵。

如要查看 Systems Manager 條件索引鍵的清單，請參閱《服務授權參考》中的 AWS Systems Manager 的條件索引鍵。若要了解您可以針對何種動作及資源使用條件索引鍵，請參閱 AWS Systems Manager 定義的動作。

如需使用 ssm:resourceTag/* 條件金鑰的相關資訊，請參閱以下主題：

- 限制透過 SSM Agent 存取根層級命令
- 根據標籤限制 Run Command 存取
- 根據執行個體標籤限制工作階段存取

如需使用 ssm:Recursive、ssm:Policies 和 ssm:Overwrite 條件索引鍵的相關資訊，請參閱防止存取 Parameter Store API 操作。

範例

若要檢視 Systems Manager 身分型政策範例，請參閱 AWS Systems Manager 身分型政策範例。

Systems Manager 資源型政策

其他 AWS 服務，例如 Amazon Simple Storage Service (Amazon S3)，支援以資源為基礎的許可政策。例如，您可以將許可政策連接至 S3 儲存貯體，以管理該儲存貯體的存取許可。

Systems Manager 不支援資源型政策。

以 Systems Manager 標籤為基礎的授權

您可以將標籤連接到 Systems Manager 資源，或是在請求中將標籤傳遞給 Systems Manager。若要根據標籤控制存取，請使用 `ssm:resourceTag/key-name`、`aws:ResourceTag/key-name`、`aws:RequestTag/key-name` 或 `aws:TagKeys` 條件索引鍵，在政策的條件元素中，提供標籤資訊。您可以在建立或更新下列資源類型時，將標籤新增至下列資源類型：

- 文件
- 受管節點
- Maintenance window (維護時段)
- 參數
- 修補基準
- OpsItem

若要檢視身分型政策範例，以根據該資源上的標籤來限制存取資源，請參閱 根據標籤來檢視 Systems Manager 文件。

Systems Manager IAM 角色

IAM 角色是 中具有特定許可 AWS 帳戶 的實體。

將臨時憑證與 Systems Manager 搭配使用

您可以搭配聯合使用暫時憑證、擔任 IAM 角色，或是擔任跨帳戶角色。您可以透過呼叫 AssumeRole 或 GetFederationToken 等 AWS Security Token Service AWS STS API 操作來取得臨時安全登入資料。

Systems Manager 支援使用臨時憑證。

服務連結角色

服務連結角色 AWS 服務 允許 存取其他服務中的資源，以代表您完成 動作。服務連結角色會列於您的 IAM 帳戶中，並由該服務所擁有。 管理員可以檢視，但不能編輯服務連結角色的許可。

Systems Manager 支援服務連結角色。如需建立或管理 Systems Manager 服務連結角色的詳細資訊，請參閱使用 Systems Manager 的服務連結角色。

服務角色

此功能可讓服務代表您擔任服務角色。此角色可讓服務存取其他服務中的資源，以代表您完成動作。服務角色會顯示在您的 IAM 帳戶中，且由該帳戶所擁有。這表示 管理員可以變更此角色的許可。不過，這樣可能會破壞此服務的功能。

Systems Manager 支援服務角色。

在 Systems Manager 中選擇 IAM 角色

若要讓 Systems Manager 與您的受管節點互動，您必須選擇角色以允許 Systems Manager 代表您存取節點。如果您之前已建立服務角色或服務連結角色，Systems Manager 會提供您角色清單讓您選擇。請務必選擇允許存取啟動和停止受管節點的角色。

若要存取 EC2 執行個體，您必須設定執行個體許可。如需相關資訊，請參閱設定 Systems Manager 所需的執行個體許可。

若要存取混合多雲端環境中的非 EC2 節點，您的 AWS 帳戶 需要的角色是 IAM 服務角色。如需相關資訊，請參閱在混合多雲端環境中建立 Systems Manager 所需的 IAM 服務角色。

自動化工作流程可在服務角色 (或擔任角色) 的內容下啟動。這可讓服務代表您執行動作。如果您未指定擔任角色，Automation 會使用呼叫執行的使用者內容。然而，某些情況仍需要您為自動化指定服務角色。如需詳細資訊，請參閱設定自動化的服務角色 (擔任角色) 存取權。

AWS Systems Manager 受管政策

AWS 提供由 建立和管理的獨立 IAM 政策，以解決許多常見的使用案例 AWS。這些 AWS 管理的政策會授予常用案例所需的許可，讓您不需調查需要哪些許可。(您也可以建立自己的自訂 IAM 政策，以允許 Systems Manager 動作與資源的許可。)

如需有關 Systems Manager 受管政策的詳細資訊，請參閱 AWS 的 受管政策 AWS Systems Manager

如需受管政策的一般資訊，請參閱《IAM 使用者指南》中的 AWS managed policies。

AWS Systems Manager 身分型政策範例

根據預設， AWS Identity and Access Management (IAM) 實體 （ 使用者和角色 ） 沒有建立或修改 AWS Systems Manager 資源的許可。他們也無法使用 Systems Manager 主控台、 AWS Command Line Interface (AWS CLI) 或 AWS API 執行任務。管理員必須建立 IAM 政策，授與使用者和角色在指定資源上執行特定 API 操作所需的許可。管理員接著必須將這些政策連接至需要這些許可的使用者或群組。

以下是許可政策的範例，允許使用者刪除名稱開頭為 的文件**MyDocument-**，在美國東部 （ 俄亥俄 ） (us-east-2) AWS 區域。

```
{
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "ssm:DeleteDocument"
      ],
      "Resource" : [
        "arn:aws:ssm:us-east-2:111122223333:document/MyDocument-*"
      ]
    }
  ]
}
```

若要了解如何使用這些範例 JSON 政策文件建立 IAM 身分型政策，請參閱《IAM 使用者指南》中的建立 IAM 政策。

主題

- 政策最佳實務
- 使用 Systems Manager 主控台
- 允許使用者檢視他們自己的許可
- 預防跨服務混淆代理人
- 客戶受管政策範例
- 根據標籤來檢視 Systems Manager 文件

政策最佳實務

身分型政策會判斷您帳戶中的某個人員是否可以建立、存取或刪除 Systems Manager 資源。這些動作可能會讓您的 AWS 帳戶產生費用。當您建立或編輯身分型政策時，請遵循下列準則及建議事項：

- 開始使用 AWS 受管政策並邁向最低權限許可 – 若要開始將許可授予您的使用者和工作負載，請使用將許可授予許多常見使用案例的 AWS 受管政策。它們可在您的 中使用 AWS 帳戶。建議您定義特定於使用案例 AWS 的客戶受管政策，以進一步減少許可。如需更多資訊，請參閱 IAM 使用者指南中的 AWS 受管政策或任務職能的AWS 受管政策。
- 套用最低權限許可 – 設定 IAM 政策的許可時，請僅授予執行任務所需的許可。為實現此目的，您可以定義在特定條件下可以對特定資源採取的動作，這也稱為最低權限許可。如需使用 IAM 套用許可的更多相關資訊，請參閱 IAM 使用者指南中的 IAM 中的政策和許可。
- 使用 IAM 政策中的條件進一步限制存取權 – 您可以將條件新增至政策，以限制動作和資源的存取。例如，您可以撰寫政策條件，指定必須使用 SSL 傳送所有請求。如果透過特定 等使用服務動作 AWS 服務，您也可以使用條件來授予存取 AWS CloudFormation。如需詳細資訊，請參閱 IAM 使用者指南中的 IAM JSON 政策元素：條件。
- 使用 IAM Access Analyzer 驗證 IAM 政策，確保許可安全且可正常運作 – IAM Access Analyzer 驗證新政策和現有政策，確保這些政策遵從 IAM 政策語言 (JSON) 和 IAM 最佳實務。IAM Access Analyzer 提供 100 多項政策檢查及切實可行的建議，可協助您撰寫安全且實用的政策。如需詳細資訊，請參閱《IAM 使用者指南》中的使用 IAM Access Analyzer 驗證政策。
- 需要多重要素驗證 (MFA) – 如果您的案例需要 IAM 使用者或 中的根使用者 AWS 帳戶，請開啟 MFA 以提高安全性。如需在呼叫 API 操作時請求 MFA，請將 MFA 條件新增至您的政策。如需詳細資訊，請參閱《IAM 使用者指南》https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_mfa_configure-api-require.html中的透過 MFA 的安全 API 存取。

如需 IAM 中最佳實務的相關資訊，請參閱 IAM 使用者指南中的 IAM 安全最佳實務。

使用 Systems Manager 主控台

若要存取 Systems Manager 主控台，您必須擁有最低的一組許可。這些許可必須允許您列出和檢視您 AWS 帳戶中 Systems Manager 資源和其他資源的詳細資訊。

如果您建立比最低必要許可更嚴格的身分型政策，則對於具有該政策的 IAM 實體 (使用者或角色) 而言，主控台就無法如預期運作。

對於僅呼叫 AWS CLI 或 AWS API 的使用者，您不需要允許最低主控台許可。反之，只需允許存取符合您嘗試執行之 API 作業的動作就可以了。

為了確保使用者和角色仍可使用 Systems Manager 主控台，請將 AmazonSSMFullAccess 或 AmazonSSMReadOnlyAccess AWS 受管政策連接至實體。如需詳細資訊，請參閱《IAM 使用者指南》中的新增許可到使用者。

允許使用者檢視他們自己的許可

此範例會示範如何建立政策，允許 IAM 使用者檢視附加到他們使用者身分的內嵌及受管政策。此政策包含在 主控台上完成此動作的許可，或使用 AWS CLI 或 AWS API 以程式設計方式完成此動作的許可。

```
{
    "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": "*"
        }
    ]
}
```

預防跨服務混淆代理人

混淆代理人問題屬於安全性議題，其中沒有執行動作許可的實體可以強制具有更多許可的實體執行該動作。在 中 AWS，跨服務模擬可能會導致混淆代理人問題。在某個服務 (呼叫服務) 呼叫另一個服務 (被呼叫服務) 時，可能會發生跨服務模擬。可以操縱呼叫服務來使用其許可，以其不應有存取許可的方式對其他客戶的資源採取動作。為了防止這種情況， AWS 提供工具，協助您保護所有 服務的資料，讓服務主體能夠存取您帳戶中的資源。

若要限制 AWS Systems Manager 為資源提供另一項服務的許可，我們建議在資源政策中使用 aws:SourceArn 和 aws:SourceAccount 全域條件內容索引鍵。如果 aws:SourceArn 值不包含帳戶 ID (例如 S3 儲存貯體的 Amazon Resource Name (ARN))，則您必須使用這兩個全域條件內容索引鍵來限制許可。如果同時使用這兩個全域條件內容索引鍵，且 aws:SourceArn 值包含帳戶 ID，則在相同政策陳述式中使用 aws:SourceAccount 值和 aws:SourceArn 值中的帳戶時，必須使用相同的帳戶 ID。如果您想要僅允許一個資源與跨服務存取相關聯，則請使用 aws:SourceArn。如果您想要允許該帳戶中的任何資源與跨服務使用相關聯，請使用 aws:SourceAccount。

下列各節提供AWS Systems Manager工具的範例政策。

混合式啟用政策範例

對於混合式啟用中使用的服務角色，aws:SourceArn 的值必須是 AWS 帳戶的 ARN。請務必在您建立混合啟用的 ARN AWS 區域 中指定 。如果不知道資源的完整 ARN，或者如果您指定了多個資源，請使用 aws:SourceArn 全域條件內容索引鍵，同時使用萬用字元 (*) 表示 ARN 的未知部分。例如：arn:aws:ssm:*:*region*:*123456789012*:*。

下列範例示範對於 Automation 使用 aws:SourceArn 和 aws:SourceAccount 全局條件內容索引鍵，以防止美國東部 (俄亥俄) 區域 (us-east-2) 的混淆代理問題。

```
{
   "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:*"
            }
         }
      }
   ]
}
```

資源資料同步政策範例

Systems Manager 庫存、Explorer 和合規可讓您建立資源資料同步，以集中在中央 Amazon Simple Storage Service 儲存貯體中儲存操作資料 (OpsData)。如果您想要使用 AWS Key Management Service (AWS KMS) 加密資源資料同步，則必須建立新的金鑰，其中包含下列政策，或者您必須更新現有的金鑰並將此政策新增至其中。此政策中的 `aws:SourceArn` 和 `aws:SourceAccount` 條件索引鍵會預防混淆代理人問題。以下是一個範例政策。

```
{
    "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**
>
> 政策範例中的 ARN 可讓系統加密來自除 AWS Security Hub以外所有來源的 OpsData。如果您需要加密 Security Hub 資料，例如，如果您使用 Explorer 來收集 Security Hub 資料，則必須連接指定下列 ARN 的其他政策：
> `"aws:SourceArn": "arn:aws:ssm:*:account-id:role/aws-service-role/opsdatasync.ssm.amazonaws.com/AWSServiceRoleForSystemsManagerOpsDataSync"`

客戶受管政策範例

您可以建立獨立的政策，在您自己的 AWS 帳戶進行管理。我們將這些稱為客戶受管政策。您可以將這些政策連接至 中的多個委託人實體 AWS 帳戶。將政策連接到主體實體時，便向實體授予了政策中定義的許可。如需詳細資訊，請參閱《IAM 使用者指南》中的客戶受管政策範例。

下列使用者政策範例授予執行各種 Systems Manager 動作的許可。使用這些政策來限制 IAM 實體 (使用者或角色) 的 Systems Manager 存取。這些政策會在 Systems Manager API、 AWS SDKs或 中執行動作時運作 AWS CLI。使用主控台的使用者，需要授予專屬於主控台的額外許可。如需詳細資訊，請參閱使用 Systems Manager 主控台。

> **Note**
>
> 所有範例皆使用美國西部 (奧勒岡) 區域 (us-west-2) 並包含虛構帳戶 ID。不應在 AWS 公有文件 （ 以 開頭的文件 ） 的 Amazon Resource Name (ARN) 中指定帳戶 IDAWS-*。

範例

- 範例 1：允許使用者在單一區域執行 Systems Manager 操作
- 範例 2：允許使用者列出單一區域的文件

範例 1：允許使用者在單一區域執行 Systems Manager 操作

下列範例會授予許可，僅在美國東部 (俄亥俄) 區域 (us-east-2) 執行 Systems Manager 操作：

```
{
```

```
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "ssm:*"
      ],
      "Resource" : [
        "arn:aws:ssm:us-east-2:aws-account-ID:*"
      ]
    }
  ]
}
```

範例 2：允許使用者列出單一區域的文件

以下範例會授予許可，列出在美國東部 (俄亥俄) 區域 (us-east-2) 中所有以 **Update** 開頭的文件名稱：

```
{
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "ssm:ListDocuments"
      ],
      "Resource" : [
        "arn:aws:ssm:us-east-2:aws-account-ID:document/Update*"
      ]
    }
  ]
}
```

範例 3：允許使用者使用特定 SSM 文件，以在特定受管節點上執行命令

以下範例為 IAM 政策允許使用者美國東部 (俄亥俄) 區域 (us-east-2) 中執行以下操作：

- 列出 Systems Manager 文件 (SSM 文件) 和文件版本。
- 查看文件的詳細資訊。
- 使用此政策中指定的文件傳送命令。文件的名稱取決於以下項目。

```
arn:aws:ssm:us-east-2:aws-account-ID:document/Systems-Manager-document-name
```

- 將命令傳送到三個節點。節點取決於第二個 Resource 區段中的下列項目。

```
"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"
```

- 命令傳送後查看其詳細資訊。
- 在 Automation 中啟動和停止工作流程，這是 中的工具 AWS Systems Manager。
- 取得有關 Automation 工作流程的資訊。

如果想要授予使用者許可，讓他們可使用此文件來在使用者有權存取的任何節點上傳送命令，您可以在 Resource 區段中指定與以下類似的項目，並移除其他節點項目。以下範例會使用美國東部 (俄亥俄) 區域 (US-east-2)。

```
"arn:aws:ec2:us-east-2:*:instance/*"
```

```
{
    "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: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",
```

```
                "arn:aws:ssm:us-east-2:aws-account-ID: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:aws-account-ID:automation-definition/*"
            ]
        },
        {
            "Action": "ssm:DescribeAutomationExecutions",
            "Effect": "Allow",
            "Resource": [
                "*"
            ]
        },
        {
            "Action": [
                "ssm:StopAutomationExecution",
                "ssm:GetAutomationExecution"
            ],
            "Effect": "Allow",
            "Resource": [
                "*"
            ]
        }
    ]
```

```
}
```

根據標籤來檢視 Systems Manager 文件

您可以在身分型政策中使用條件，以根據標籤控制存取 Systems Manager 資源。此範例會示範如何建立政策，允許檢視 SSM 文件。但是，只有在文件標籤 `Owner` 值是該使用者的使用者名稱時，才會授予許可。此政策也會授予在主控台完成此動作的必要許可。

```
{
    "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}"}
            }
        }
    ]
}
```

您可以將此政策連接到您帳戶中的 使用者。如果名為 `richard-roe` 的使用者嘗試檢視 Systems Manager 文件，則必須將文件標記為 `Owner=richard-roe` 或 `owner=richard-roe`。否則，其會被拒絕存取。條件標籤鍵 `Owner` 符合 `Owner` 和 `owner`，因為條件索引鍵名稱不區分大小寫。如需詳細資訊，請參閱 IAM 使用者指南中的 IAM JSON 政策元素：條件。

AWS 的 受管政策 AWS Systems Manager

AWS 受管政策是由 AWS AWS 受管政策建立和管理的獨立政策旨在為許多常用案例提供許可，以便您可以開始將許可指派給使用者、群組和角色。

請記住，AWS 受管政策可能不會授予特定使用案例的最低權限許可，因為這些許可可供所有 AWS 客戶使用。我們建議您定義使用案例專屬的客戶管理政策，以便進一步減少許可。

您無法變更 AWS 受管政策中定義的許可。如果 AWS 更新受 AWS 管政策中定義的許可，則更新會影響政策連接的所有委託人身分（使用者、群組和角色）。當新的 AWS 服務 啟動或新的 API 操作可供現有服務使用時，AWS 最有可能更新 AWS 受管政策。

如需詳細資訊，請參閱《IAM 使用者指南》中的 AWS 受管政策。

主題

- AWS 受管政策：AmazonSSMServiceRolePolicy
- AWS 受管政策：AmazonSSMReadOnlyAccess
- AWS 受管政策：AWSSystemsManagerOpsDataSyncServiceRolePolicy
- AWS 管理的政策：AmazonSSMManagedEC2InstanceDefaultPolicy
- AWS 受管政策：SSMQuickSetupRolePolicy
- AWS 受管政策：AWSQuickSetupDeploymentRolePolicy
- AWS 受管政策：AWSQuickSetupPatchPolicyDeploymentRolePolicy
- AWS 受管政策：AWSQuickSetupPatchPolicyBaselineAccess
- AWS 受管政策： AWSSystemsManagerEnableExplorerExecutionPolicy
- AWS 受管政策： AWSSystemsManagerEnableConfigRecordingExecutionPolicy
- AWS 受管政策：AWSQuickSetupDevOpsGuruPermissionsBoundary
- AWS 受管政策：AWSQuickSetupDistributorPermissionsBoundary
- AWS 受管政策：AWSQuickSetupSSMHostMgmtPermissionsBoundary
- AWS 受管政策：AWSQuickSetupPatchPolicyPermissionsBoundary
- AWS 受管政策：AWSQuickSetupSchedulerPermissionsBoundary
- AWS 受管政策：AWSQuickSetupCFGCPacksPermissionsBoundary
- AWS 受管政策：AWS-SSM-DiagnosisAutomation-AdministrationRolePolicy
- AWS 受管政策：AWS-SSM-DiagnosisAutomation-ExecutionRolePolicy
- AWS 受管政策：AWS-SSM-RemediationAutomation-AdministrationRolePolicy
- AWS 受管政策：AWS-SSM-RemediationAutomation-ExecutionRolePolicy
- AWS 受管政策：AWSQuickSetupSSMManageResourcesExecutionPolicy

- AWS 受管政策：AWSQuickSetupSSMLifecycleManagementExecutionPolicy
- AWS 受管政策：AWSQuickSetupSSMDeploymentRolePolicy
- AWS 受管政策：AWSQuickSetupSSMDeploymentS3BucketRolePolicy
- AWS 受管政策：AWSQuickSetupEnableDHMCExecutionPolicy
- AWS 受管政策：AWSQuickSetupEnableAREXExecutionPolicy
- AWS 受管政策：AWSQuickSetupManagedInstanceProfileExecutionPolicy
- AWS 受管政策：AWSQuickSetupFullAccess
- AWS 受管政策：AWSQuickSetupReadOnlyAccess
- AWS 受管政策：AWSQuickSetupManageJITNAResourcesExecutionPolicy
- AWS 受管政策：AWSQuickSetupJITNADeploymentRolePolicy
- AWS 受管政策：AWSSystemsManagerJustInTimeAccessServicePolicy
- AWS 受管政策：AWSSystemsManagerJustInTimeAccessTokenPolicy
- AWS 受管政策：AWSSystemsManagerJustInTimeAccessTokenSessionPolicy
- AWS 受管政策：AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy
- AWS 受管政策：AWSSystemsManagerNotificationsServicePolicy
- AWS 受管政策：AWS-SSM-Automation-DiagnosisBucketPolicy
- AWS 受管政策：AWS-SSM-RemediationAutomation-OperationalAccountAdministrationRolePolicy
- AWS 受管政策：AWS-SSM-DiagnosisAutomation-OperationalAccountAdministrationRolePolicy
- Systems ManagerAWS 受管政策的更新
- Systems Manager 的其他受管政策

AWS 受管政策：AmazonSSMServiceRolePolicy

此政策可讓您存取由 Systems Manager 操作管理 AWS Systems Manager 或用於 Systems Manager 操作的許多 AWS 資源。

您無法`AmazonSSMServiceRolePolicy`連接至您的 AWS Identity and Access Management (IAM) 實體。此政策會連接到服務連結角色， AWS Systems Manager 允許 代表您執行動作。如需詳細資訊，請參閱使用角色來收集庫存和查看 OpsData。

許可詳細資訊

此政策包含以下許可。

- ssm：允許主體啟動並逐步執行 Run Command 和 Automation；擷取 Run Command 和 Automation 操作的相關資訊；擷取 Parameter Store 參數 Change Calendar 行事曆的相關資訊；更新和擷取 OpsCenter 資源的 Systems Manager 服務設定的相關資訊；以及讀取已套用至資源的標籤相關資訊。
- cloudformation：允許主體擷取堆疊集操作和堆疊集執行個體的相關資訊，以及刪除資源 arn:aws:cloudformation:*:*:stackset/AWS-QuickSetup-SSM*:* 上的堆疊集。允許主體刪除與下列資源相關聯的堆疊執行個體：

```
arn:aws:cloudformation:*:*:stackset/AWS-QuickSetup-SSM*:*
arn:aws:cloudformation:*:*:stackset-target/AWS-QuickSetup-SSM*:*
arn:aws:cloudformation:*:*:type/resource/*
```

- cloudwatch：允許主體擷取 Amazon CloudWatch 警示的相關資訊。
- compute-optimizer – 允許主體擷取帳戶對 AWS Compute Optimizer 服務的註冊 （選擇加入） 狀態，以及擷取符合特定陳述要求之 Amazon EC2 執行個體的建議。
- config – 允許主體在 中擷取資訊修復組態和組態記錄器 AWS Config，並判斷指定的 AWS Config 規則 AWS 和資源是否合規。
- events：允許主體擷取 EventBridge 規則的相關資訊；只為 Systems Manager 服務 (ssm.amazonaws.com) 建立 EventBridge 規則和目標；以及刪除資源 arn:aws:events:*:*:rule/SSMExplorerManagedRule 的規則和目標。
- ec2：允許主體擷取 Amazon EC2 執行個體的相關資訊。
- iam：允許主體傳遞 Systems Manager 服務 (ssm.amazonaws.com) 的角色許可。
- lambda：允許主體調用 Lambda 函數，這些函數是專為供 Systems Manager 使用而設定。
- resource-explorer-2：允許主體擷取 EC2 執行個體的相關資料，以便判斷每個執行個體目前是否由 Systems Manager 管理。

  允許對 arn:aws:resource-explorer-2:*:*:managed-view/AWSManagedViewForSSM* 資源執行 resource-explorer-2:CreateManagedView 動作。
- resource-groups – 允許主體從屬於資源群組 AWS Resource Groups 的資源擷取清單資源群組及其成員。
- securityhub – 允許主體擷取目前帳戶中中 AWS Security Hub 樞資源的相關資訊。
- states – 允許主體啟動和擷取專門為 Systems Manager AWS Step Functions 使用而設定的資訊。
- support：允許主體擷取 AWS Trusted Advisor中檢查和案例的相關資訊。
- tag：允許主體擷取帳戶指定 AWS 區域 中所有已標記或先前標記之資源的相關資訊。

若要檢視此政策的詳細資訊 (包括 JSON 政策文件的最新版本)，請參閱《AWS 管理的政策參考指南》中的 AmazonSSMServiceRolePolicy 一節。

AWS 受管政策：AmazonSSMReadOnlyAccess

您可將 AmazonSSMReadOnlyAccess 政策連接到 IAM 身分。此政策授予 AWS Systems Manager API 操作的唯讀存取權Describe*，包括 Get*、 和 List*。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 管理的政策參考指南》中的 AmazonSSMReadOnlyAccess 一節。

AWS 受管政策：AWSSystemsManagerOpsDataSyncServiceRolePolicy

您不得將 AWSSystemsManagerOpsDataSyncServiceRolePolicy 連接到 IAM 實體。此政策會連接到服務連結角色，而此角色可讓 Systems Manager 代表您執行動作。如需詳細資訊，請參閱使用角色為 OpsItems 建立 OpsData 和 Explorer：。

AWSSystemsManagerOpsDataSyncServiceRolePolicy 允許AWSServiceRoleForSystemsManagerOpsDataSync服務連結角色從 AWS Security Hub 調查結果建立和更新 OpsItems和 OpsData。

此政策允許 Systems Manager 對所有相關資源 ("Resource": "*") 完成下列動作，除非另有說明：

- 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] 透過以下條件，僅允許對 Systems Manager 服務進行 ssm:GetOpsItem 和 ssm:UpdateOpsItem 動作的許可。

```
"Condition": {
    "StringEquals": {
```

```
        "aws:ResourceTag/ExplorerSecurityHubOpsItem": "true"
    }
}
```

[2] 僅允許對以下資源進行 ssm:AddTagsToResource 動作的許可。

```
arn:aws:ssm:*:*:opsitem/*
```

[3] 僅允許對以下資源進行 ssm:UpdateServiceSetting 和 ssm:GetServiceSetting 動作的許可。

```
arn:aws:ssm:*:*:servicesetting/ssm/opsitem/*
arn:aws:ssm:*:*:servicesetting/ssm/opsdata/*
```

[4] 透過以下條件，僅拒絕對 Systems Manager 服務進行 securityhub:BatchUpdateFindings 的許可。

```
{
   "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
    }
   }
```

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 管理的政策參考指南》中的 AWSSystemsManagerOpsDataSyncServiceRolePolicy 一節。

AWS 管理的政策：AmazonSSMManagedEC2InstanceDefaultPolicy

只應將 `AmazonSSMManagedEC2InstanceDefaultPolicy` 連接到您希望有權使用 Systems Manager 功能的 Amazon EC2 執行個體 IAM 角色。不應將此角色連接到其他 IAM 實體 (例如 IAM 使用者和 IAM 群組)，或連接到其他用途的 IAM 角色。如需詳細資訊，請參閱使用預設主機管理組態來自動管理 EC2 執行個體。

此政策會授予許可，這些許可允許 Amazon EC2 執行個體上的 SSM Agent 與雲端中的 Systems Manager 服務通訊，以便執行各種任務。此政策還會為提供授權權杖的兩項服務授予許可，確保都是在正確的執行個體執行操作。

許可詳細資訊

此政策包含以下許可。

- `ssm`：允許主體擷取文件；使用 Run Command 執行命令；使用 Session Manager 建立工作階段；收集執行個體的庫存資訊；以及使用 Patch Manager 掃描修補程式和修補程式合規情況。
- `ssmmessages`：允許主體存取由 Amazon Message Gateway Service 為每個執行個體建立的個人化授權權杖。Systems Manager 根據 API 操作中提供的執行個體 Amazon Resource Name (ARN) 驗證個人化授權權杖。為確保 SSM Agent 是在正確的執行個體執行 API 操作，此存取權是必要的。
- `ec2messages`：允許主體存取由 Amazon Message Delivery Service 為每個執行個體建立的個人化授權權杖。Systems Manager 根據 API 操作中提供的執行個體 Amazon Resource Name (ARN) 驗證個人化授權權杖。為確保 SSM Agent 是在正確的執行個體執行 API 操作，此存取權是必要的。

如需 `ssmmessages` 和 `ec2messages` 端點的相關資訊 (包括兩者之間的差異)，請參閱代理程式相關的 API 操作 (ssmmessages 和 ec2messages 端點)。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 管理的政策參考指南》中的 AmazonSSMManagedEC2InstanceDefaultPolicy 一節。

AWS 受管政策：SSMQuickSetupRolePolicy

您無法將 SSMQuickSetupRolePolicy 連接至 IAM 實體。此政策會連接到服務連結角色，而此角色可讓 Systems Manager 代表您執行動作。如需詳細資訊，請參閱使用角色來維護 Quick Setup 佈建的資源運作狀態和一致性。

此政策會授予唯讀許可，這些許可允許 Systems Manager 檢查組態運作狀態、確認參數和佈建資源的使用情形是否一致，以及在偵測到漂移時修復資源。此政策還授予建立服務連結角色的管理許可。

許可詳細資訊

此政策包含以下許可。

- `ssm`：允許主體在 Systems Manager 中 (包括在委派管理員帳戶中) 讀取資訊資源資料同步和 SSM 文件。這是必要條件，因此 Quick Setup 可以判斷已設定資源的預期狀態。
- `organizations`：允許主體讀取屬於組織 (如 AWS Organizations中所設定) 的成員帳戶的相關資訊。這是必要條件，因此 Quick Setup 可以識別組織中要執行資源運作狀態檢查的所有帳戶。

- `cloudformation` – 允許主體從中讀取資訊 AWS CloudFormation。這是必要的，因此 Quick Setup可以收集用於管理資源狀態和 CloudFormation AWS CloudFormation 堆疊集操作之堆疊的資料。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 SSMQuickSetupRolePolicy。

AWS 受管政策：AWSQuickSetupDeploymentRolePolicy

受管政策 `AWSQuickSetupDeploymentRolePolicy` 支援多種 Quick Setup 組態類型。這些組態類型會建立 IAM 角色和自動化，透過建議的最佳實務來設定常用的 Amazon Web Services 服務和功能。

您可以將 `AWSQuickSetupDeploymentRolePolicy` 連接到 IAM 實體。

此政策會授予管理許可，在建立與下列 Quick Setup 組態相關聯的資源時需要這些許可：

- 使用 Quick Setup 設定 Amazon EC2 主機管理
- 使用 Quick Setup 建立 AWS Config組態記錄器
- 使用 部署 AWS Config 一致性套件 Quick Setup
- 使用 Quick Setup 設定 DevOps Guru
- 使用 Quick Setup 部署 Distributor 套件
- 使用Quick Setup依照排程自動停止和啟動 EC2 執行個體

許可詳細資訊

此政策包含以下許可。

- `iam`：允許主體管理和刪除自動化組態任務所需的 IAM 角色，以及管理自動化角色政策。
- `cloudformation`：允許主體建立和管理堆疊集。
- `config`：允許主體建立、管理和刪除一致性套件。
- `events`：允許主體建立、更新和刪除排程動作的事件規則。
- `resource-groups`：允許主體擷取資源查詢，這些查詢與被 Quick Setup 組態視為目標的資源群組相關聯。
- `ssm`：允許主體建立會套用 Quick Setup 組態的 Automation 執行手冊和關聯。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWSQuickSetupDeploymentRolePolicy。

AWS 受管政策：AWSQuickSetupPatchPolicyDeploymentRolePolicy

受管政策 AWSQuickSetupPatchPolicyDeploymentRolePolicy 支援使用修補程式政策設定組織中執行個體的Quick Setup修補 Quick Setup 類型。此組態類型有助於將修補單一帳戶或組織中應用程式和節點的程序自動化。

您可以將 AWSQuickSetupPatchPolicyDeploymentRolePolicy 連接至 IAM 實體。Systems Manager 也會將此政策連接至允許 Systems Manager 代表您執行動作的服務角色。

此政策會授予管理許可，這些許可允許 Quick Setup 建立與修補程式政策組態相關聯的資源。

許可詳細資訊

此政策包含以下許可。

- iam：允許主體管理和刪除自動化組態任務所需的 IAM 角色，以及管理自動化角色政策。
- cloudformation – 允許主體讀取 AWS CloudFormation 堆疊資訊，以及控制Quick Setup使用 AWS CloudFormation 堆疊集建立的 AWS CloudFormation 堆疊。
- ssm：允許主體建立、更新、讀取和刪除組態任務所需的 Automation 執行手冊，以及建立、更新和刪除 State Manager 關聯。
- resource-groups：允許主體擷取資源查詢，這些查詢與被 Quick Setup 組態視為目標的資源群組相關聯。
- s3：允許主體列出 Amazon S3 儲存貯體，以及管理可用來存放修補程式政策存取日誌的儲存貯體。
- lambda – 允許主體管理將組態維持在正確狀態的 AWS Lambda 修復函數。
- logs：允許主體描述和管理 Lambda 組態資源的日誌群組。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWSQuickSetupPatchPolicyDeploymentRolePolicy。

AWS 受管政策：AWSQuickSetupPatchPolicyBaselineAccess

受管政策 AWSQuickSetupPatchPolicyBaselineAccess 支援使用修補程式政策設定組織中執行個體的Quick Setup修補 Quick Setup 類型。此組態類型有助於將修補單一帳戶或組織中應用程式和節點的程序自動化。

您可以將 AWSQuickSetupPatchPolicyBaselineAccess 連接至 IAM 實體。Systems Manager 也會將此政策連接至允許 Systems Manager 代表您執行動作的服務角色。

此政策提供唯讀許可，以使用 存取目前 AWS 帳戶 或組織中管理員已設定的修補程式基準Quick Setup。修補基準存放在 Amazon S3 儲存貯體中，可用於修補單一帳戶或整個組織中的執行個體。

許可詳細資訊

此政策包含以下許可。

- s3：允許主體讀取存放在 Amazon S3 儲存貯體中的修補基準覆寫。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWSQuickSetupPatchPolicyBaselineAccess。

AWS 受管政策： **AWSSystemsManagerEnableExplorerExecutionPolicy**

受管政策AWSSystemsManagerEnableExplorerExecutionPolicy支援啟用 Explorer，這是 中的工具 AWS Systems Manager。

您可以將 AWSSystemsManagerEnableExplorerExecutionPolicy 連接至 IAM 實體。Systems Manager 也會將此政策連接至允許 Systems Manager 代表您執行動作的服務角色。

此政策會授予啟用 Explorer 的管理許可。其中包括更新相關 Systems Manager 服務設定的許可，以及為 Systems Manager 建立服務連結角色的許可。

許可詳細資訊

此政策包含以下許可。

- config：允許主體透過提供組態記錄器詳細資訊的唯讀存取權來協助啟用 Explorer。
- iam：允許主體協助啟用 Explorer。
- ssm：允許主體啟動會啟用 Explorer 的自動化工作流程。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWSSystemsManagerEnableExplorerExecutionPolicy。

AWS 受管政策：**AWSSystemsManagerEnableConfigRecordingExecutionPolicy**

受管政策 AWSSystemsManagerEnableConfigRecordingExecutionPolicy 支援使用 Quick Setup 建立 AWS Config組態記錄器 Quick Setup 組態類型。此組態類型可讓 Quick Setup 追蹤和記錄您選擇的 AWS 資源類型的變更 AWS Config。其也可讓 Quick Setup 設定所記錄資料的交付和通知選項。

您可以將 AWSSystemsManagerEnableConfigRecordingExecutionPolicy 連接至 IAM 實體。Systems Manager 也會將此政策連接至允許 Systems Manager 代表您執行動作的服務角色。

此政策會授予允許 Quick Setup 啟用和設定 AWS Config 組態記錄的管理許可。

許可詳細資訊

此政策包含以下許可。

- s3：允許主體建立和設定 Amazon S3 儲存貯體來交付組態記錄。
- sns：允許主體列出和建立 Amazon SNS 主題。
- config：允許主體設定和啟動組態記錄器，以及協助啟用 Explorer。
- iam – 允許主體建立、取得和傳遞 的服務連結角色 AWS Config；以及建立 Systems Manager 的服務連結角色；以及協助啟用 Explorer。
- ssm：允許主體啟動會啟用 Explorer 的自動化工作流程。
- compute-optimizer – 允許主體Explorer透過提供唯讀存取權來協助啟用 ，以判斷資源是否已註冊 AWS Compute Optimizer。
- support：允許主體透過提供唯讀存取權，判斷是否已向 AWS Compute Optimizer註冊資源來協助啟用 Explorer。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWSSystemsManagerEnableConfigRecordingExecutionPolicy。

AWS 受管政策：AWSQuickSetupDevOpsGuruPermissionsBoundary

> **Note**
>
> 此政策是許可界限。許可界限負責設定身分型政策可為 IAM 實體授予的最大許可。不應自行使用和連接 Quick Setup 許可界限政策。Quick Setup 許可界限政策只應連接到 Quick Setup 受管角色。如需許可界限的詳細資訊，請參閱 IAM 使用者指南中的 IAM 實體許可界限。

受管政策 `AWSQuickSetupDevOpsGuruPermissionsBoundary` 支援使用 Quick Setup 設定 DevOps Guru類型。此組態類型會啟用採用了機器學習技術的 Amazon DevOps Guru。DevOps Guru 服務可協助改善應用程式的運作效能和可用性。

在使用 Quick Setup 建立 `AWSQuickSetupDevOpsGuruPermissionsBoundary` 組態時，系統會將此許可界限套用至部署組態時建立的 IAM 角色。此許可界限會限制 Quick Setup 建立的角色範圍。

此政策會授予管理許可，這些許可允許 Quick Setup 啟用和設定 Amazon DevOps Guru。

許可詳細資訊

此政策包含以下許可。

- `iam`：允許主體為 DevOps Guru 和 Systems Manager 建立服務連結角色，以及列出有助啟用 Explorer 的角色。
- `cloudformation`：允許主體列出和描述 AWS CloudFormation 堆疊。
- `sns`：允許主體列出和建立 Amazon SNS 主題。
- `devops-guru`：允許主體設定 DevOps Guru，以及新增通知管道。
- `config`：允許主體透過提供組態記錄器詳細資訊的唯讀存取權來協助啟用 Explorer。
- `ssm`：允許主體啟動會啟用 Explorer 的自動化工作流程，以及讀取和更新 Explorer 服務設定。
- `compute-optimizer` – 允許主體Explorer透過提供唯讀存取權來協助啟用 ，以判斷資源是否已註冊 AWS Compute Optimizer。
- `support`：允許主體透過提供唯讀存取權，判斷是否已向 AWS Compute Optimizer註冊資源來協助啟用 Explorer。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWSQuickSetupDevOpsGuruPermissionsBoundary。

AWS 受管政策：AWSQuickSetupDistributorPermissionsBoundary

> **Note**
>
> 此政策是許可界限。許可界限負責設定身分型政策可為 IAM 實體授予的最大許可。不應自行使用和連接 Quick Setup 許可界限政策。Quick Setup 許可界限政策只應連接到 Quick Setup 受管角色。如需許可界限的詳細資訊，請參閱 IAM 使用者指南中的 IAM 實體許可界限。

受管政策 AWSQuickSetupDistributorPermissionsBoundary 支援使用 Quick Setup 部署 Distributor 套件 Quick Setup 組態類型。組態類型有助於使用 Distributor 這個工具，將 代理程式等軟體套件分發到您的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體 AWS Systems Manager。

在使用 Quick Setup 建立 AWSQuickSetupDistributorPermissionsBoundary 組態時，系統會將此許可界限套用至部署組態時建立的 IAM 角色。此許可界限會限制 Quick Setup 建立的角色範圍。

此政策會授予管理許可，這些許可允許 Quick Setup 使用 Distributor，將代理程式等軟體套件發布至 Amazon EC2 執行個體。

許可詳細資訊

此政策包含以下許可。

- iam：允許主體取得與傳遞 Distributor 自動化角色；建立、讀取、更新和刪除預設執行個體角色；將預設執行個體角色傳遞至 Amazon EC2 和 Systems Manager；將執行個體管理政策連接至執行個體角色；為 Systems Manager 建立服務連結角色；將預設執行個體角色新增至執行個體設定檔；讀取 IAM 角色和執行個體設定檔的相關資訊；以及建立預設執行個體設定檔。
- ec2：允許主體將預設執行個體設定檔與 EC2 執行個體建立關聯，以及協助啟用 Explorer。
- ssm：允許主體啟動會設定執行個體和安裝套件的自動化工作流程；協助啟動會啟用 Explorer 的自動化工作流程；以及讀取和更新 Explorer 服務設定。
- config：允許主體透過提供組態記錄器詳細資訊的唯讀存取權來協助啟用 Explorer。
- compute-optimizer – 允許主體Explorer透過提供唯讀存取權來協助啟用 ，以判斷資源是否已註冊 AWS Compute Optimizer。
- support：允許主體透過提供唯讀存取權，判斷是否已向 AWS Compute Optimizer註冊資源來協助啟用 Explorer。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWSQuickSetupDistributorPermissionsBoundary。

AWS 受管政策：AWSQuickSetupSSMHostMgmtPermissionsBoundary

> **Note**
>
> 此政策是許可界限。許可界限負責設定身分型政策可為 IAM 實體授予的最大許可。不應自行使用和連接 Quick Setup 許可界限政策。Quick Setup 許可界限政策只應連接到 Quick Setup 受管角色。如需許可界限的詳細資訊，請參閱 IAM 使用者指南中的 IAM 實體許可界限。

受管政策 `AWSQuickSetupSSMHostMgmtPermissionsBoundary` 支援使用 Quick Setup 設定 Amazon EC2 主機管理 Quick Setup 組態類型。此組態類型會設定 IAM 角色，並啟用常用的 Systems Manager 工具，以安全地管理您的 Amazon EC2 執行個體。

在使用 Quick Setup 建立 `AWSQuickSetupSSMHostMgmtPermissionsBoundary` 組態時，系統會將此許可界限套用至部署組態時建立的 IAM 角色。此許可界限會限制 Quick Setup 建立的角色範圍。

此政策授予管理許可，Quick Setup允許 啟用和設定安全管理 EC2 執行個體所需的 Systems Manager 工具。

許可詳細資訊

此政策包含以下許可。

- `iam`：允許主體取得此服務角色，以及將其傳遞至 Automation。允許主體建立、讀取、更新和刪除預設執行個體角色；將預設執行個體角色傳遞至 Amazon EC2 和 Systems Manager；將執行個體管理政策連接至執行個體角色；為 Systems Manager 建立服務連結角色；將預設執行個體角色新增至執行個體設定檔；讀取 IAM 角色和執行個體設定檔的相關資訊；以及建立預設執行個體設定檔。
- `ec2`：允許主體將預設執行個體設定檔與 EC2 執行個體建立關聯和取消關聯。
- `ssm` – 允許主體啟動自動化工作流程，以啟用 Explorer；讀取和更新Explorer服務設定；設定執行個體；並在執行個體上啟用 Systems Manager 工具。
- `compute-optimizer` – 允許主體Explorer透過提供唯讀存取權來協助啟用 ，以判斷資源是否已註冊 AWS Compute Optimizer。
- `support`：允許主體透過提供唯讀存取權，判斷是否已向 AWS Compute Optimizer註冊資源來協助啟用 Explorer。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWSQuickSetupSSMHostMgmtPermissionsBoundary。

AWS 受管政策：AWSQuickSetupPatchPolicyPermissionsBoundary

> **Note**
>
> 此政策是許可界限。許可界限負責設定身分型政策可為 IAM 實體授予的最大許可。不應自行使用和連接 Quick Setup 許可界限政策。Quick Setup 許可界限政策只應連接到 Quick Setup 受管角色。如需許可界限的詳細資訊，請參閱 IAM 使用者指南中的 IAM 實體許可界限。

受管政策 AWSQuickSetupPatchPolicyPermissionsBoundary 支援使用修補程式政策設定組織中執行個體的Quick Setup修補 Quick Setup 類型。此組態類型有助於將修補單一帳戶或組織中應用程式和節點的程序自動化。

在使用 Quick Setup 建立 AWSQuickSetupPatchPolicyPermissionsBoundary 組態時，系統會將此許可界限套用至部署組態時建立的 IAM 角色。此許可界限會限制 Quick Setup 建立的角色範圍。

此政策授予管理許可，Quick Setup允許 在 中的Patch Manager工具 中啟用和設定修補程式政策 AWS Systems Manager。

許可詳細資訊

此政策包含以下許可。

- iam – 允許主體取得Patch Manager自動化角色； 將自動化角色傳遞至Patch Manager修補操作； 建立預設執行個體角色 AmazonSSMRoleForInstancesQuickSetup； 將預設執行個體角色傳遞給 Amazon EC2 和 Systems Manager； 將選取的 AWS 受管政策連接至執行個體角色； 為 Systems Manager 建立服務連結角色； 將預設執行個體角色新增至執行個體描述檔； 讀取執行個體描述檔和角色的相關資訊； 建立預設執行個體描述檔； 和 標記具有讀取修補程式基準覆寫許可的角色。
- ssm：允許主體更新由 Systems Manager 管理的執行個體角色；管理由 Patch Manager 修補程式政策在 Quick Setup 中建立的關聯；標記被修補程式政策組態視為目標的執行個體；讀取執行個體和修補狀態的相關資訊；啟動會設定、啟用和修復執行個體修補的自動化工作流程；啟動會啟用 Explorer 的自動化工作流程；協助啟用 Explorer；以及讀取和更新 Explorer 服務設定。
- ec2：允許主體將預設執行個體設定檔與 EC2 執行個體建立關聯和取消關聯；標記被修補程式政策組態視為目標的執行個體；標記被修補程式政策組態視為目標的執行個體；以及協助啟用 Explorer。
- s3：允許主體建立和設定 S3 儲存貯體，以存放修補基準覆寫。

- `lambda` – 允許主體叫用設定修補的 AWS Lambda 函數，並在刪除Quick Setup修補政策組態後執行清除操作。
- `logs` – 允許主體設定Patch ManagerQuick Setup AWS Lambda 函數的記錄。
- `config`：允許主體透過提供組態記錄器詳細資訊的唯讀存取權來協助啟用 Explorer。
- `compute-optimizer` – 允許主體Explorer透過提供唯讀存取權來協助啟用 ，以判斷資源是否已註冊 AWS Compute Optimizer。
- `support`：允許主體透過提供唯讀存取權，判斷是否已向 AWS Compute Optimizer註冊資源來協助啟用 Explorer。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWSQuickSetupPatchPolicyPermissionsBoundary。

AWS 受管政策：AWSQuickSetupSchedulerPermissionsBoundary

> **Note**
>
> 此政策是許可界限。許可界限負責設定身分型政策可為 IAM 實體授予的最大許可。不應自行使用和連接 Quick Setup 許可界限政策。Quick Setup 許可界限政策只應連接到 Quick Setup 受管角色。如需許可界限的詳細資訊，請參閱 IAM 使用者指南中的 IAM 實體許可界限。

受管政策 `AWSQuickSetupSchedulerPermissionsBoundary` 支援使用Quick Setup依照排程自動停止和啟動 EC2 執行個體 Quick Setup 組態類型。此組態類型可讓您在指定的時間停止和啟動 EC2 執行個體及其他資源。

在使用 Quick Setup 建立 `AWSQuickSetupSchedulerPermissionsBoundary` 組態時，系統會將此許可界限套用至部署組態時建立的 IAM 角色。此許可界限會限制 Quick Setup 建立的角色範圍。

此政策會授予管理許可，這些許可允許 Quick Setup 啟用和設定在 EC2 執行個體及其他資源上的排程操作。

許可詳細資訊

此政策包含以下許可。

- `iam` – 允許主體擷取和傳遞執行個體管理自動化動作的角色； 管理 通過、 和 連接 EC2 執行個體管理的預設執行個體角色； 建立預設執行個體描述檔； 將預設執行個體角色新增至執行個體描述檔； 為 建立服務連結角色Systems Manager； 讀取 IAM 角色和執行個體描述檔的相關資訊； 將預設執

行個體描述檔與 EC2 執行個體建立關聯； 和 啟動自動化工作流程，以設定執行個體並啟用其上的 Systems Manager工具。

- ssm：允許主體啟動會啟用 Explorer 的自動化工作流程，以及讀取和更新 Explorer 服務設定。
- ec2：允許主體尋找目標執行個體，以及依排程啟動和停止這些執行個體。
- config：允許主體透過提供組態記錄器詳細資訊的唯讀存取權來協助啟用 Explorer。
- compute-optimizer – 允許主體Explorer透過提供唯讀存取權來協助啟用 ，以判斷資源是否已註冊 AWS Compute Optimizer。
- support – 允許主體Explorer透過提供帳戶的 AWS Trusted Advisor 檢查唯讀存取權來協助啟用 。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWSQuickSetupSchedulerPermissionsBoundary。

AWS 受管政策：AWSQuickSetupCFGCPacksPermissionsBoundary

> **Note**
>
> 此政策是許可界限。許可界限負責設定身分型政策可為 IAM 實體授予的最大許可。不應自行使用和連接 Quick Setup 許可界限政策。Quick Setup 許可界限政策只應連接到 Quick Setup 受管角色。如需許可界限的詳細資訊，請參閱 IAM 使用者指南中的 IAM 實體許可界限。

受管政策 AWSQuickSetupCFGCPacksPermissionsBoundary 支援使用 部署 AWS Config 一致性套件 Quick Setup Quick Setup 組態類型。此組態類型會部署 AWS Config 一致性套件。一致性套件是 AWS Config 規則和修補動作的集合，可部署為單一實體。

在使用 Quick Setup 建立 AWSQuickSetupCFGCPacksPermissionsBoundary 組態時，系統會將此許可界限套用至部署組態時建立的 IAM 角色。此許可界限會限制 Quick Setup 建立的角色範圍。

此政策會授予管理許可，這些許可允許 Quick Setup 部署 AWS Config 一致性套件。

許可詳細資訊

此政策包含以下許可。

- iam – 允許主體建立、取得和傳遞服務連結角色 AWS Config。
- sns：允許主體列出 Amazon SNS 中的平台應用程式。
- config – 允許主體部署 AWS Config 一致性套件；取得一致性套件的狀態；以及取得組態記錄器的相關資訊。

- ssm：允許主體取得 SSM 文件和 Automation 工作流程的相關資訊；取得資源標籤的相關資訊；以及取得更新服務設定的相關資訊。
- compute-optimizer：允許主體取得帳戶的選擇加入狀態。
- support：允許主體取得 AWS Trusted Advisor 檢查的相關資訊。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWSQuickSetupCFGCPacksPermissionsBoundary。

AWS 受管政策：AWS-SSM-DiagnosisAutomation-AdministrationRolePolicy

透過在節點受管的帳戶和區域中啟動自動化工作流程，政策 AWS-SSM-DiagnosisAutomation-AdministrationRolePolicy 為診斷與 Systems Manager 服務互動的節點的問題提供許可。

您可以將 AWS-SSM-DiagnosisAutomation-AdministrationRolePolicy 連接至 IAM 實體。Systems Manager 也會將此政策連接至允許 Systems Manager 代表您執行診斷動作的服務角色。

許可詳細資訊

此政策包含以下許可。

- ssm：允許主體執行 Automation 執行手冊，這些手冊可診斷節點問題，也可存取工作流程的執行狀態。
- kms：允許主體使用加密 S3 儲存貯體中物件所用的客戶指定 AWS Key Management Service 金鑰，以便解密和存取儲存貯體中的物件內容。
- sts：允許主體擔任診斷執行角色，以便在同一帳戶中執行 Automation 執行手冊。
- iam：允許主體將診斷管理角色 (例如，自我) 傳遞至 Systems Manager，以便執行 Automation 執行手冊。
- s3：允許主體存取物件以及將物件寫入 S3 儲存貯體。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWS-SSM-DiagnosisAutomation-AdministrationRolePolicy。

AWS 受管政策：AWS-SSM-DiagnosisAutomation-ExecutionRolePolicy

受管政策 AWS-SSM-DiagnosisAutomation-ExecutionRolePolicy 提供在目標 AWS 帳戶 和區域中執行 Automation 執行手冊的管理許可，以便診斷與 Systems Manager 服務互動的受管節點的問題。

您可以將 AWS-SSM-DiagnosisAutomation-ExecutionRolePolicy 連接至 IAM 實體。Systems Manager 也會將此政策連接至允許 Systems Manager 代表您執行動作的服務角色。

許可詳細資訊

此政策包含以下許可。

- ssm：允許主體執行診斷特定的 Automation 執行手冊，以及存取自動化工作流程狀態和執行中繼資料。
- ec2：允許主體描述 Amazon EC2 和 Amazon VPC 資源及其組態，以便診斷 Systems Manager 服務的問題。
- kms：允許主體使用加密 S3 儲存貯體中物件所用的客戶指定 AWS Key Management Service 金鑰，以便解密和存取儲存貯體中的物件內容。
- iam：允許主體將診斷執行角色 (例如，自我) 傳遞至 Systems Manager，以便執行 Automation 文件。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWS-SSM-DiagnosisAutomation-ExecutionRolePolicy。

AWS 受管政策：AWS-SSM-RemediationAutomation-AdministrationRolePolicy

透過在受管節點的帳戶和區域中啟動自動化工作流程，受管政策 AWS-SSM-RemediationAutomation-AdministrationRolePolicy 為修復與 Systems Manager 服務互動的受管節點中的問題提供許可。

您可以將此政策連接至 IAM 實體。Systems Manager 也會將此政策連接至允許 Systems Manager 代表您執行修復動作的服務角色。

許可詳細資訊

此政策包含以下許可。

- ssm：允許主體執行特定的 Automation 執行手冊，以及存取自動化工作流程狀態和執行狀態。
- kms：允許主體使用加密 S3 儲存貯體中物件所用的客戶指定 AWS Key Management Service 金鑰，以便解密和存取儲存貯體中的物件內容。
- sts：允許主體擔任修復執行角色，以便在同一帳戶中執行 SSM Automation 文件。
- iam：允許主體將修復管理員角色 (例如，自我) 傳遞至 Systems Manager，以便執行 Automation 文件。

- s3：允許主體存取物件以及將物件寫入 S3 儲存貯體。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWS-SSM-RemediationAutomation-AdministrationRolePolicy。

AWS 受管政策：AWS-SSM-RemediationAutomation-ExecutionRolePolicy

受管政策 AWS-SSM-RemediationAutomation-ExecutionRolePolicy 提供在特定目標帳戶和區域中執行 Automation 執行手冊的許可，以便修復與 Systems Manager 服務互動的受管節點的問題。

您可以將此政策連接至 IAM 實體。Systems Manager 也會將此政策連接至允許 Systems Manager 代表您執行修復動作的服務角色。

許可詳細資訊

此政策包含以下許可。

- ssm：允許主體執行特定的 Automation 執行手冊，以及存取執行中繼資料和狀態。
- ec2：允許主體建立、存取和修改 Amazon EC2 資源與 Amazon VPC 資源及其組態，以便修復 Systems Manager 服務和相關資源 (例如，安全群組) 的問題；以及將標籤連接到資源。
- kms：允許主體使用加密 S3 儲存貯體中物件所用的客戶指定 AWS Key Management Service 金鑰，以便解密和存取儲存貯體中的物件內容。
- iam：允許主體將修復執行角色 (例如，自我) 傳遞至 SSM 服務，以便執行 Automation 文件。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWS-SSM-RemediationAutomation-ExecutionRolePolicy。

AWS 受管政策：AWSQuickSetupSSMManageResourcesExecutionPolicy

此政策會授予許可，這些許可允許 Quick Setup 執行 AWSQuickSetupType-SSM-SetupResources Automation 執行手冊。此執行手冊會為 Quick Setup 關聯建立 IAM 角色，而關聯是由 AWSQuickSetupType-SSM 部署所建立。此政策還會授予許可，以便在 Quick Setup 刪除操作期間清理關聯的 Amazon S3 儲存貯體。

您可以將此政策連接至 IAM 實體。Systems Manager 也會將此政策連接至允許 Systems Manager 代表您執行動作的服務角色。

許可詳細資訊

此政策包含以下許可。

- `iam`：允許主體列出和管理 IAM 角色，以便與 Quick Setup Systems Manager Explorer 操作搭配使用；檢視、連接和分離 IAM 政策，以便與Quick Setup和 Systems Manager Explorer 搭配使用。這些許可是必要的，因此Quick Setup可以建立其某些組態操作所需的角色。
- `s3`：允許主體從主體帳戶的 Amazon S3 儲存貯體中擷取物件的相關資訊，以及從中刪除專門用於 Quick Setup 組態操作的物件。這是必要許可，如此就能夠移除完成組態後不再需要的 S3 物件。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWSQuickSetupSSMManageResourcesExecutionPolicy。

AWS 受管政策：AWSQuickSetupSSMLifecycleManagementExecutionPolicy

`AWSQuickSetupSSMLifecycleManagementExecutionPolicy` 政策會授予管理許可，Quick Setup允許 在 中的Quick Setup部署期間，在生命週期事件上執行 AWS CloudFormation 自訂資源 Systems Manager。

您可以將此政策連接至 IAM 實體。Systems Manager 也會將此政策連接至允許 Systems Manager 代表您執行動作的服務角色。

許可詳細資訊

此政策包含以下許可。

- `ssm`：允許主體取得自動化執行的相關資訊，以及啟動自動化執行來設定特定的 Quick Setup 操作。
- `iam`：允許主體從 IAM 傳遞角色來設定特定的 Quick Setup 資源。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWSQuickSetupSSMLifecycleManagementExecutionPolicy。

AWS 受管政策：AWSQuickSetupSSMDeploymentRolePolicy

受管政策 `AWSQuickSetupSSMDeploymentRolePolicy` 會授予管理許可，這些許可允許 Quick Setup 建立在 Systems Manager 加入程序期間使用的資源。

雖然可以手動將此政策連接至 IAM 實體，但不建議這麼做。因為 Quick Setup 建立的實體，會將此政策連接至允許 Systems Manager 代表您執行動作的服務角色。

此政策與 SSMQuickSetupRolePolicy 政策無關，後者會用於為 `AWSServiceRoleForSSMQuickSetup` 服務連結角色提供許可。

許可詳細資訊

此政策包含以下許可。

- `ssm` – 允許主體管理使用 AWS CloudFormation 範本和一組特定 SSM 文件建立之特定資源的關聯；使用 管理角色和角色政策，以透過 AWS CloudFormation 範本診斷和修復受管節點；以及連接和刪除Quick Setup生命週期事件的政策
- `iam` – 允許主體標記角色並傳遞 Systems Manager 服務和 Lambda 服務的角色許可，以及傳遞診斷操作的角色許可。
- `lambda` – 允許主體使用 AWS CloudFormation 範本標記和管理主體帳戶中Quick Setup生命週期的函數。
- `cloudformation` – 允許主體從中讀取資訊 AWS CloudFormation。這是必要的，因此 Quick Setup可以收集用於管理資源狀態和 CloudFormation AWS CloudFormation 堆疊集操作之堆疊的資料。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWSQuickSetupSSMDeploymentRolePolicy。

AWS 受管政策：AWSQuickSetupSSMDeploymentS3BucketRolePolicy

`AWSQuickSetupSSMDeploymentS3BucketRolePolicy` 政策授予的許可，可列出帳戶中的所有 S3 儲存貯體，也可管理和擷取透過 AWS CloudFormation 範本管理的主體帳戶中特定儲存貯體的相關資訊。

您可以將 `AWSQuickSetupSSMDeploymentS3BucketRolePolicy` 連接至 IAM 實體。Systems Manager 也會將此政策連接至允許 Systems Manager 代表您執行動作的服務角色。

許可詳細資訊

此政策包含以下許可。

- `s3` ：允許主體列出帳戶中的所有 S3 儲存貯體；以及管理和擷取透過 AWS CloudFormation 範本管理之主體帳戶中特定儲存貯體的相關資訊。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWSQuickSetupSSMDeploymentS3BucketRolePolicy。

AWS 受管政策：AWSQuickSetupEnableDHMCExecutionPolicy

此政策會授予管理許可，這些許可允許主體執行 `AWSQuickSetupType-EnableDHMC` Automation 執行手冊，而這又會啟用預設主機管理組態。預設主機管理組態設定允許 Systems Manager 自動

將 Amazon EC2 執行個體作為受管執行個體來管理。受管執行個體是指一種設定為搭配 Systems Manager 使用的 EC2 執行個體。此政策也會授予許可，可用來建立 Systems Manager 服務設定中指定作為 SSM Agent 預設角色的 IAM 角色。

您可以將 `AWSQuickSetupEnableDHMCExecutionPolicy` 連接至 IAM 實體。Systems Manager 也會將此政策連接至允許 Systems Manager 代表您執行動作的服務角色。

許可詳細資訊

此政策包含以下許可。

- `ssm`：允許主體更新和取得 Systems Manager 服務設定的相關資訊。
- `iam`：允許主體建立和擷取與 Quick Setup 操作的 IAM 角色相關的資訊。

若要檢視政策的詳細資訊，包括最新版本的 JSON 政策文件，請參閱《AWS 受管政策參考指南AWSQuickSetupEnableDHMCExecutionPolicy》中的 。

AWS 受管政策：AWSQuickSetupEnableAREXExecutionPolicy

此政策會授予管理許可，這些許可允許 Systems Manager 執行 `AWSQuickSetupType-EnableAREX` Automation 執行手冊，而這又讓 AWS 資源總管 能夠與 Systems Manager 搭配使用。Resource Explorer 可讓您檢視帳戶中的資源，其搜尋體驗與網際網路搜尋引擎類似。此政策也會授予 Resource Explorer 索引和檢視的管理許可。

您可以將 `AWSQuickSetupEnableAREXExecutionPolicy` 連接至 IAM 實體。Systems Manager 也會將此政策連接至允許 Systems Manager 代表您執行動作的服務角色。

許可詳細資訊

此政策包含以下許可。

- `iam` – 允許主體在 AWS Identity and Access Management (IAM) 服務中建立服務連結角色。
- `resource-explorer-2`：允許主體擷取與 Resource Explorer 檢視和索引相關的資訊；建立 Resource Explorer 檢視和索引；變更 Quick Setup 中所顯示索引的索引類型。

若要檢視政策的詳細資訊，包括最新版本的 JSON 政策文件，請參閱《AWS 受管政策參考指南AWSQuickSetupEnableAREXExecutionPolicy》中的 。

AWS 受管政策：AWSQuickSetupManagedInstanceProfileExecutionPolicy

THhis 政策授予管理許可，Systems Manager允許 為Quick Setup工具建立預設 IAM 執行個體描述檔，並將其連接到尚未連接執行個體描述檔的 Amazon EC2 執行個體。此政策也會授予 Systems Manager 將許可連接到現有執行個體設定檔的能力。這是為了確保 Systems Manager 與 EC2 執行個體上的 SSM Agent 通訊所需的許可已就緒。

您可以將 `AWSQuickSetupManagedInstanceProfileExecutionPolicy` 連接至 IAM 實體。Systems Manager 也會將此政策連接至允許 Systems Manager 代表您執行動作的服務角色。

許可詳細資訊

此政策包含以下許可。

- `ssm`：允許主體啟動與 Quick Setup 程序相關聯的自動化工作流程。
- `ec2`：允許主體將 IAM 執行個體設定檔連接至由 Quick Setup 管理的 EC2 執行個體。
- `iam`：允許主體從 IAM 建立、更新和擷取在 Quick Setup 程序中使用的角色相關資訊；建立 IAM 執行個體設定檔；將 `AmazonSSMManagedInstanceCore` 受管政策連接至 IAM 執行個體設定檔。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWSQuickSetupManagedInstanceProfileExecutionPolicy。

AWS 受管政策：AWSQuickSetupFullAccess

此政策授予管理許可，允許完整存取 AWS Management Console 和 AWS SDKs 中的 AWS Systems ManagerQuick Setup API 動作和資料，以及對Quick Setup操作所需的其他 AWS 服務 資源的有限存取。

您可將 `AWSQuickSetupFullAccess` 政策連接到 IAM 身分。

許可詳細資訊

此政策包含以下許可。

- `ssm`：允許主體啟用 Explorer；在 State Manager 中執行資源資料同步操作；以及使用 SSM Command 文件和 Automation 執行手冊執行操作。

  Explorer、、State Manager文件和自動化都是 Systems Manager 中的工具。
- `cloudformation` – 允許主體執行跨 AWS 區域 和 佈建資源所需的 AWS CloudFormation 操作 AWS 帳戶。

- ec2：允許主體為指定的組態選取必要的參數，以及在 AWS Management Console中提供驗證。
- iam：允許主體為 Quick Setup 操作建立所需的服務角色和服務連結角色。
- organizations – 允許主體讀取 AWS Organizations 組織中帳戶的狀態；擷取組織的結構；啟用受信任存取；以及從管理帳戶註冊委派管理員帳戶。
- resource-groups：允許主體為指定的組態選取必要的參數，以及在 AWS Management Console 中提供驗證。
- s3：允許主體為指定的組態選取必要的參數，以及在 AWS Management Console中提供驗證。
- ssm-quicksetup：允許主體在 Quick Setup 中執行唯讀動作。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWSQuickSetupFullAccess。

AWS 受管政策：AWSQuickSetupReadOnlyAccess

此政策授予唯讀許可，允許主體檢視 AWS Systems Manager Quick Setup資料和報告，包括來自 Quick Setup操作所需的其他 AWS 服務 資源的資訊。

您可將 AWSQuickSetupReadOnlyAccess 政策連接到 IAM 身分。

許可詳細資訊

此政策包含以下許可。

- ssm：允許主體讀取 SSM Command 文件和 Automation 執行手冊，以及擷取 State Manager 關聯執行的狀態。
- cloudformation – 允許主體啟動擷取 AWS CloudFormation 部署狀態所需的操作。
- organizations – 允許主體讀取 AWS Organizations 組織中帳戶的狀態。
- ssm-quicksetup：允許主體在 Quick Setup 中執行唯讀動作。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWSQuickSetupReadOnlyAccess。

AWS 受管政策：AWSQuickSetupManageJITNAResourcesExecutionPolicy

受管政策AWSQuickSetupManageJITNAResourcesExecutionPolicy可讓 Systems Manager 中的Quick Setup工具 設定just-in-time節點存取。

您可以將 AWSQuickSetupManageJITNAResourcesExecutionPolicy 連接至 IAM 實體。Systems Manager 也會將此政策連接至允許 Systems Manager 代表您執行動作的服務角色。

此政策會授予管理許可，Systems Manager允許 建立與just-in-time節點存取相關聯的資源。

許可詳細資訊

此政策包含以下許可。

- ssm – 允許主體取得和更新指定身分提供者的服務設定，以進行just-in-time節點存取。
- iam – 允許主體建立、標記和取得角色、為just-in-time節點存取受管政策連接角色政策，以及為just-in-time節點存取和通知建立服務連結角色。

若要檢視政策的詳細資訊，包括最新版本的 JSON 政策文件，請參閱《 AWS 受管政策參考指南》中的 AWSQuickSetupManageJITNAResourcesExecutionPolicy。

AWS 受管政策：AWSQuickSetupJITNADeploymentRolePolicy

受管政策AWSQuickSetupJITNADeploymentRolePolicy允許 Quick Setup 部署設定just-in-time節點存取所需的組態類型。

您可以將 AWSQuickSetupJITNADeploymentRolePolicy 連接至 IAM 實體。Systems Manager 也會將此政策連接至允許 Systems Manager 代表您執行動作的服務角色。

此政策會授予管理許可，Systems Manager允許 建立與just-in-time節點存取相關聯的資源。

許可詳細資訊

此政策包含以下許可。

- cloudformation – 允許主體建立、更新、刪除和讀取 AWS CloudFormation 堆疊。
- ssm – 允許主體建立、刪除、更新和讀取由 呼叫的State Manager關聯 AWS CloudFormation。
- iam – 允許主體建立、刪除、讀取和標記由 呼叫的 IAM 角色 AWS CloudFormation。

若要檢視政策的詳細資訊，包括最新版本的 JSON 政策文件，請參閱《 AWS 受管政策參考指南》中的 AWSQuickSetupJITNADeploymentRolePolicy。

AWS 受管政策：AWSSystemsManagerJustInTimeAccessServicePolicy

受管政策AWSSystemsManagerJustInTimeAccessServicePolicy提供許可給 Systems Manager just-in-time節點存取功能所管理或使用 AWS 的資源。

您不得將 AWSSystemsManagerJustInTimeAccessServicePolicy 連接到 IAM 實體。此政策會連接到服務連結角色，而此角色可讓 Systems Manager 代表您執行動作。如需詳細資訊，請參閱使用角色來啟用just-in-time節點存取。

此政策會授予管理許可，允許存取與just-in-time節點存取相關聯的資源。

許可詳細資訊

此政策包含以下許可。

- ssm – 允許主體建立、描述和更新 OpsItems、描述工作階段和列出受管節點的標籤。
- ssm-guiconnect – 允許主體列出連線。
- identitystore – 允許 取得使用者和群組 IDs、描述使用者和列出群組成員資格。
- sso-directory – 允許主體描述使用者，並判斷使用者是否為群組的成員。
- sso – 允許主體描述已註冊的區域，並列出執行個體和目錄關聯。
- cloudwatch – 允許主體放置AWS/SSM/JustInTimeAccess命名空間的指標資料。
- ec2 – 允許主體描述標籤。

若要檢視政策的詳細資訊，包括最新版本的 JSON 政策文件，請參閱《 AWS 受管政策參考指南》中的 AWSSystemsManagerJustInTimeAccessServicePolicy。

AWS 受管政策：AWSSystemsManagerJustInTimeAccessTokenPolicy

受管政策AWSSystemsManagerJustInTimeAccessTokenPolicy允許 Systems Manager產生用於just-in-time節點存取的存取字符。

您可以將 AWSSystemsManagerJustInTimeAccessTokenPolicy 連接到 IAM 實體。

此政策會授予管理許可，Systems Manager允許 啟動工作階段並產生存取權杖以just-in-time存取節點。

許可詳細資訊

此政策包含以下許可。

- ssm – 允許主體傳送命令、列出命令叫用，以及開始、繼續和結束Session Manager工作階段。
- ssm-guiconnect – 允許主體啟動、描述和結束 Systems Manager GUI Connect RDP 連線。

- kms – 允許主體建立授予並產生金鑰資料。
- sso – 允許主體列出目錄關聯。
- identitystore – 允許主體描述使用者。

若要檢視政策的詳細資訊，包括最新版本的 JSON 政策文件，請參閱《 AWS 受管政策參考指南》中的 AWSSystemsManagerJustInTimeAccessTokenPolicy。

AWS 受管政策：AWSSystemsManagerJustInTimeAccessTokenSessionPolicy

受管政策AWSSystemsManagerJustInTimeAccessTokenSessionPolicy允許 Systems Manager將範圍縮小許可套用至just-in-time節點存取字符。

您可以將 AWSSystemsManagerJustInTimeAccessTokenSessionPolicy 連接到 IAM 實體。

此政策會授予管理許可，Systems Manager允許 縮小just-in-time節點存取字符的許可範圍。

許可詳細資訊

此政策包含以下許可。

- ssm – 允許主體使用 SSM-SessionManagerRunShell 文件啟動Session Manager工作階段。此外，當第一次透過 呼叫 時ssm-guiconnect，請使用 AWS-StartPortForwardingSession 文件啟動工作階段、列出命令叫用，以及使用 AWSSSO-CreateSSOUser 文件傳送命令。
- ssm-guiconnect – 允許主體取消、取得和啟動所有資源的連線。
- kms - 允許主體建立授予，並在ssm-guiconnect透過 AWS 服務呼叫 SystemsManagerJustInTimeNodeAccessManaged時為標記 的金鑰產生資料金鑰。
- sso – 允許主體在透過 呼叫時列出目錄關聯ssm-guiconnect。
- identitystore – 允許主體在透過 呼叫 時描述使用者ssm-guiconnect。

若要檢視政策的詳細資訊，包括最新版本的 JSON 政策文件，請參閱《 AWS 受管政策參考指南》中的 AWSSystemsManagerJustInTimeAccessTokenSessionPolicy。

AWS 受管政策：AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy

受管政策AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy允許 Systems Manager將拒絕存取政策從委派管理員帳戶共用給成員帳戶，並跨多個帳戶複寫政策 AWS 區域。

您可以將 AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy 連接到 IAM 實體。

此政策提供 Systems Manager共用和建立拒絕存取政策所需的管理許可。這可確保拒絕存取政策會套用至針對just-in-time節點存取所設定之 AWS Organizations 組織和區域中的所有帳戶。

許可詳細資訊

此政策包含以下許可。

- ssm – 允許主體管理 SSM 文件和資源政策。
- ssm-quicksetup – 允許主體讀取Quick Setup組態管理員。
- organizations – 允許主體列出有關 AWS Organizations 組織和委派管理員的詳細資訊。
- ram – 允許主體建立、標記和描述資源共用。
- iam – 允許主體描述服務角色。

若要檢視政策的詳細資訊，包括最新版本的 JSON 政策文件，請參閱《 AWS 受管政策參考指南》中的 AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy。

AWS 受管政策：AWSSystemsManagerNotificationsServicePolicy

受管政策AWSSystemsManagerNotificationsServicePolicy允許 Systems Manager 傳送just-in-time節點存取請求的電子郵件通知，以存取請求核准者。

您不得將 AWSSystemsManagerJustInTimeAccessServicePolicy 連接到 IAM 實體。此政策會連接到服務連結角色，而此角色可讓 Systems Manager 代表您執行動作。如需詳細資訊，請參閱使用角色傳送just-in-time節點存取請求通知。

此政策會授予管理許可，Systems Manager允許 傳送電子郵件通知給just-in-time節點存取請求，以存取請求核准者。

許可詳細資訊

此政策包含以下許可。

- identitystore – 允許主體列出和描述使用者和群組成員資格。
- sso – 允許主體列出執行個體、目錄和描述已註冊的區域。
- sso-directory – 允許主體描述 群組中的使用者和清單成員。
- iam – 允許主體取得角色的相關資訊。

若要檢視政策的詳細資訊，包括最新版本的 JSON 政策文件，請參閱《 AWS 受管政策參考指南》中的 AWSSystemsManagerNotificationsServicePolicy。

AWS 受管政策：AWS-SSM-Automation-DiagnosisBucketPolicy

受管政策透過允許存取用於診斷和修復問題的 S3 儲存貯體 AWS Systems Manager ，AWS-SSM-Automation-DiagnosisBucketPolicy提供診斷與服務互動之節點問題的許可。

您可將 AWS-SSM-Automation-DiagnosisBucketPolicy 政策連接到 IAM 身分。Systems Manager 也會將此政策連接至 IAM 角色，此角色允許 Systems Manager 代表您執行診斷動作。

許可詳細資訊

此政策包含以下許可。

- s3：允許主體存取物件以及將物件寫入 Amazon S3 儲存貯體。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWS-SSM-Automation-DiagnosisBucketPolicy。

AWS 受管政策：AWS-SSM-RemediationAutomation-OperationalAccountAdministrationRolePolicy

受管政策 AWS-SSM-RemediationAutomation-OperationalAccountAdministrationRolePolicy 透過提供組織特定的許可，為操作帳戶診斷節點問題提供許可。

您可將 AWS-SSM-RemediationAutomation-OperationalAccountAdministrationRolePolicy 連接到 IAM 身分。Systems Manager 也會將此政策連接至 IAM 角色，此角色允許 Systems Manager 代表您執行診斷動作。

許可詳細資訊

此政策包含以下許可。

- organizations：允許主體列出組織的根目錄，以及取得成員帳戶來判斷目標帳戶。
- sts：允許主體擔任修復執行角色，以便在同一組織中跨帳戶和區域執行 SSM Automation 文件。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWS-SSM-RemediationAutomation-OperationalAccountAdministrationRolePolicy。

AWS 受管政策：AWS-SSM-DiagnosisAutomation-OperationalAccountAdministrationRolePolicy

受管政策 `AWS-SSM-DiagnosisAutomation-OperationalAccountAdministrationRolePolicy` 透過提供組織特定的許可，為操作帳戶診斷節點問題提供許可。

您可將 `AWS-SSM-DiagnosisAutomation-OperationalAccountAdministrationRolePolicy` 政策連接到 IAM 身分。Systems Manager 也會將此政策連接至 IAM 角色，此角色允許 Systems Manager 代表您執行診斷動作。

許可詳細資訊

此政策包含以下許可。

- `organizations`：允許主體列出組織的根目錄，以及取得成員帳戶來判斷目標帳戶。
- `sts`：允許主體擔任診斷執行角色，以便在同一組織中跨帳戶和區域執行 SSM Automation 文件。

若要檢視此政策的詳細資訊 (包含 JSON 政策文件的最新版本)，請參閱《AWS 受管政策參考指南》中的 AWS-SSM-DiagnosisAutomation-OperationalAccountAdministrationRolePolicy。

Systems ManagerAWS 受管政策的更新

在下表中，檢視自此服務於 2021 年 3 月 12 日開始追蹤 AWS 受管政策更新Systems Manager以來的詳細資訊。如需 Systems Manager 服務其他受管政策的相關資訊，請參閱本主題後文的 Systems Manager 的其他受管政策小節。如需有關此頁面變更的自動提醒，請訂閱 Systems Manager 文件歷史記錄 頁面的 RSS 摘要。

變更	描述	日期
AWSQuickSetupDeploymentRolePolicy – 更新至政策	Systems Manager 新增存取 Amazon 擁有的 Runbook AWSQuickSetupType-ManageInstanceProfile `AWSQuickSetupDeploymentRolePolicy` 的受管	2025 年 6 月 10 日

變更	描述	日期
	政策許可。此許可可讓 Quick Setup 使用 受管政策而非內嵌政策建立關聯。	
AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy – 政策更新	Systems Manager 新增許可，Systems Manager以允許 標記 共用的資源 AWS Resource Access Manager ，以進行just-in-time節點存取。	2025 年 4 月 30 日
AWSQuickSetupManageJITNAResourcesExecutionPolicy – 政策更新	Systems Manager 新增許可，Systems Manager允許標記為即時節點存取而建立的just-in-time角色。	2025 年 4 月 30 日
AWSSystemsManagerJustInTimeAccessTokenSessionPolicy – 新政策	Systems Manager 新增了新的政策，以允許 Systems Manager 將範圍縮小許可套用至just-in-time節點存取字符。	2025 年 4 月 30 日
AWSSystemsManagerNotificationsServicePolicy – 新政策	Systems Manager 新增了新的政策，Systems Manager以允許 just-in-time節點存取請求的電子郵件通知，以存取請求核准者。	2025 年 4 月 30 日
AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy – 新政策	Systems Manager 新增了新的政策，以允許 Systems Manager 將核准政策複寫到不同的區域。	2025 年 4 月 30 日
AWSSystemsManagerJustInTimeAccessTokenPolicy – 新政策	Systems Manager 新增了新的政策，Systems Manager以允許 產生用於just-in-time節點存取的存取字符。	2025 年 4 月 30 日

變更	描述	日期
AWSSystemsManagerJustInTimeAccessServicePolicy – 新政策	Systems Manager 新增了新的政策，以提供許可給 AWS Systems Manager just-in-time節點存取功能所管理或使用的資源。	2025 年 4 月 30 日
AWSQuickSetupManageJITNAResourcesExecutionPolicy – 新政策	Systems Manager 新增了新的政策，以允許 Systems Manager 中的Quick Setup工具建立just-in-time節點存取所需的 IAM 角色。	2025 年 4 月 30 日
AWSQuickSetupJITNADeploymentRolePolicy – 新政策	Systems Manager 新增了新的政策，提供許可，Quick Setup 允許 部署設定just-in-time節點存取所需的組態類型。	2025 年 4 月 30 日
AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy – 政策更新	Systems Manager 新增許可，Systems Manager以允許 標記 共用的資源 AWS Resource Access Manager ，以進行just-in-time節點存取。	2025 年 4 月 30 日
AWSQuickSetupManageJITNAResourcesExecutionPolicy – 政策更新	Systems Manager 新增許可，Systems Manager允許標記為即時節點存取而建立的just-in-time角色。	2025 年 4 月 30 日
AWSSystemsManagerJustInTimeAccessTokenSessionPolicy – 新政策	Systems Manager 新增了新的政策，以允許 Systems Manager 將範圍縮小許可套用至just-in-time節點存取字符。	2025 年 4 月 30 日

變更	描述	日期
AWSSystemsManagerNotificationsServicePolicy – 新政策	Systems Manager 新增了新的政策，Systems Manager允許 傳送電子郵件通知給just-in-time節點存取請求，以存取請求核准者。	2025 年 4 月 30 日
AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy – 新政策	Systems Manager 新增了新的政策，以允許 Systems Manager 將核准政策複寫到不同的區域。	2025 年 4 月 30 日
AWSSystemsManagerJustInTimeAccessTokenPolicy – 新政策	Systems Manager 新增了新的政策，Systems Manager以允許 產生用於just-in-time節點存取的存取字符。	2025 年 4 月 30 日
AWSSystemsManagerJustInTimeAccessServicePolicy – 新政策	Systems Manager 新增了新的政策，以提供許可給 Systems Manager just-in-time節點存取功能所管理或使用 AWS 的資源。	2025 年 4 月 30 日
AWSQuickSetupManageJITNAResourcesExecutionPolicy – 新政策	Systems Manager 新增了新的政策Quick Setup，以允許 Systems Manager 中的工具 建立just-in-time節點存取所需的 IAM 角色。	2025 年 4 月 30 日
AWSQuickSetupJITNADeploymentRolePolicy – 新政策	Systems Manager 新增了新的政策，提供許可，Quick Setup 允許 部署設定just-in-time節點存取所需的組態類型。	2025 年 4 月 30 日

變更	描述	日期
AWS-SSM-DiagnosisAutomation-OperationalAccountAdministrationRolePolicy – 新政策	Systems Manager 新增了政策，此政策會透過提供組織特定的許可，為操作帳戶提供診斷節點問題的許可。	2024 年 11 月 21 日
AWS-SSM-RemediationAutomation-OperationalAccountAdministrationRolePolicy – 新政策	Systems Manager 新增了政策，此政策會透過提供組織特定的許可，為操作帳戶提供診斷節點問題的許可。	2024 年 11 月 21 日
AWS-SSM-Automation-DiagnosisBucketPolicy – 新政策	Systems Manager 新增了政策，可支援啟動自動化工作流程，這些工作流程會診斷目標帳戶和區域中受管節點的問題。	2024 年 11 月 21 日
AmazonSSMServiceRolePolicy – 更新現有政策	Systems Manager 新增了許可， AWS 資源總管 允許 收集 Amazon EC2 執行個體的詳細資訊，並在新Systems Manager儀表板的小工具中顯示結果。	2024 年 11 月 21 日
SSMQuickSetupRolePolicy – 更新現有政策	Systems Manager 更新了受管政策 SSMQuickSetupRolePolicy 。此更新可讓相關聯的服務連結角色 AWSServiceRoleForSSMQuickSetup 管理資源資料同步。	2024 年 11 月 21 日

變更	描述	日期
AWS-SSM-DiagnosisAutomation-AdministrationRolePolicy – 新政策	Systems Manager 新增了政策，可支援啟動自動化工作流程，這些工作流程會診斷目標帳戶和區域中受管節點的問題。	2024 年 11 月 21 日
AWS-SSM-DiagnosisAutomation-ExecutionRolePolicy – 新政策	Systems Manager 新增了政策，可支援啟動自動化工作流程，這些工作流程會診斷目標帳戶和區域中受管節點的問題。	2024 年 11 月 21 日
AWS-SSM-RemediationAutomation-AdministrationRolePolicy – 新政策	Systems Manager 新增了政策，可支援啟動自動化工作流程，這些工作流程會修復目標帳戶和區域中受管節點的問題。	2024 年 11 月 21 日
AWS-SSM-RemediationAutomation-ExecutionRolePolicy – 新政策	Systems Manager 新增了政策，可支援啟動自動化工作流程，這些工作流程會修復目標帳戶和區域中受管節點的問題。	2024 年 11 月 21 日
AWSQuickSetupSSMDeploymentRolePolicy – 政策更新	Systems Manager 新增許可，Systems Manager允許 標記為統一主控台建立的 IAM 角色和 Lambda。	2025 年 5 月 7 日
AWSQuickSetupSSMManageResourcesExecutionPolicy – 新政策	Systems Manager 新增了政策，可支援在 Quick Setup 中執行為 Quick Setup 關聯建立 IAM 角色的操作，而關聯則由 AWSQuickSetupType-SSM 部署所建立。	2024 年 11 月 21 日

變更	描述	日期
AWSQuickSetupSSMLifecycleManagementExecutionPolicy – 新政策	Systems Manager 新增了新的政策，以支援在Quick Setup部署期間Quick Setup於生命週期事件上執行 AWS CloudFormation 自訂資源。	2024 年 11 月 21 日
AWSQuickSetupSSMDeploymentRolePolicy – 新政策	Systems Manager 新增了政策，可支援授予管理許可，這些許可允許 Quick Setup 建立 Systems Manager 加入程序期間使用的資源。	2024 年 11 月 21 日
AWSQuickSetupSSMDeploymentS3BucketRolePolicy – 新政策	Systems Manager 新增了新的政策，以支援管理和擷取透過 AWS CloudFormation 範本管理之主體帳戶中特定儲存貯體的相關資訊	2024 年 11 月 21 日
AWSQuickSetupEnableDHMCExecutionPolicy – 新政策	Systems Manager 將導入新的政策，允許 Quick Setup 建立本身使用現有 AmazonSSMManagedEC2InstanceDefaultPolicy 的 IAM 角色。此政策包含 SSM Agent 與 Systems Manager 服務通訊所需的所有許可。新政策也允許對 Systems Manager 服務設定進行修改。	2024 年 11 月 21 日
AWSQuickSetupEnableAREXExecutionPolicy – 新政策	Systems Manager 新增了新的政策，Quick Setup以允許為 建立服務連結角色 AWS 資源總管，以存取 Resource Explorer 檢視和彙總工具索引。	2024 年 11 月 21 日

變更	描述	日期
AWSQuickSetupManagedInstanceProfileExecutionPolicy – 新政策	Systems Manager 新增了政策，允許 Quick Setup 建立預設 Quick Setup 執行個體設定檔，以及將其連接至任何缺少相關聯執行個體設定檔的 Amazon EC2 執行個體。這個新政策也允許 Quick Setup 將許可連接到現有的設定檔，確保已授予所有必要的 Systems Manager 許可。	2024 年 11 月 21 日
AWSQuickSetupFullAccess – 新政策	Systems Manager 新增了新的政策，以允許實體完整存取和 SDKs 中的 AWS Systems ManagerQuick Setup API 動作 AWS Management Console 和 AWS 資料，以及有限存取 Quick Setup操作所需的其他 AWS 服務 資源。	2024 年 11 月 21 日
AWSQuickSetupReadOnlyAccess – 新政策	Systems Manager 新增了新的政策，以授予唯讀許可，允許主體檢視 AWS Systems Manager Quick Setup資料和報告，包括Quick Setup操作所需的其他 AWS 服務 資源資訊。	2024 年 11 月 21 日
SSMQuickSetupRolePolicy – 更新現有政策	Systems Manager 新增了新的許可，Quick Setup允許檢查其已建立之其他 AWS CloudFormation 堆疊集的運作狀態。	2024 年 8 月 13 日

變更	描述	日期
AmazonSSMManagedEC2InstanceDefaultPolicy – 更新現有政策	Systems Manager 已將陳述式 ID (Sid) 新增至 AmazonSSMManagedEC2InstanceDefaultPolicy 的 JSON 政策。這些 Sid 提供每個政策陳述式目的之內嵌描述。	2024 年 7 月 18 日
SSMQuickSetupRolePolicy – 新政策	Systems Manager 新增了政策，允許 Quick Setup 檢查已部署資源的運作狀態，以及修偏離了移原始組態的執行個體。	2024 年 7 月 3 日
AWSQuickSetupDeploymentRolePolicy – 新政策	Systems Manager 新增了政策，可支援建立 IAM 角色和自動化的多種 Quick Setup 組態類型，進而使用建議的最佳實務來設定常用的 Amazon Web Services 服務和功能。	2024 年 7 月 3 日
AWSQuickSetupPatchPolicyDeploymentRolePolicy：新政策	Systems Manager 新增了政策，允許 Quick Setup 建立與 Patch Manager 修補程式政策 Quick Setup 組態相關聯的資源。	2024 年 7 月 3 日
AWSQuickSetupPatchPolicyBaselineAccess：新政策	Systems Manager 新增了政策，允許 Quick Setup 使用唯讀許可存取 Patch Manager 中的修補基準。	2024 年 7 月 3 日
AWSSystemsManagerEnableExplorerExecutionPolicy：新政策	Systems Manager 新增了政策，允許 Explorer 授予會啟用 Quick Setup 的管理許可。	2024 年 7 月 3 日

變更	描述	日期
AWSSystemsManagerEnableConfigRecordingExecutionPolicy：新政策	Systems Manager 新增了新的政策，Quick Setup以允許 啟用和設定 AWS Config 組態記錄。	2024 年 7 月 3 日
AWSQuickSetupDevOpsGuruPermissionsBoundary：新政策	Systems Manager 新增了政策，允許 Quick Setup 啟用和設定 Amazon DevOps Guru。	2024 年 7 月 3 日
AWSQuickSetupDistributorPermissionsBoundary：新政策	Systems Manager 新增了新的政策，Quick Setup以允許啟用和設定 Distributor，這是其中的工具 AWS Systems Manager。	2024 年 7 月 3 日
AWSQuickSetupSSMHostMgmtPermissionsBoundary：新政策	Systems Manager 新增了新的政策，Quick Setup以允許 啟用和設定 Systems Manager 工具，以安全地管理 Amazon EC2 執行個體。	2024 年 7 月 3 日
AWSQuickSetupPatchPolicyPermissionsBoundary：新政策	Systems Manager 新增了新的政策，Quick Setup以允許在 中啟用和設定修補程式政策 Patch Manager，這是 中的工具 AWS Systems Manager。	2024 年 7 月 3 日
AWSQuickSetupSchedulerPermissionsBoundary：新政策	Systems Manager 新增了政策，允許 Quick Setup 在 Amazon EC2 執行個體和其他資源上啟用和設定排程操作。	2024 年 7 月 3 日
AWSQuickSetupCFGCPacksPermissionsBoundary：新政策	Systems Manager 新增了政策，允許 Quick Setup 部署 AWS Config 一致性套件。	2024 年 7 月 3 日

變更	描述	日期
AWSSystemsManagerOpsDataSyncServiceRolePolicy – 更新現有政策	OpsCenter 更新了政策，以改善 Explorer 用於管理 OpsData 相關操作的服務連結角色內的服務程式碼安全性。	2023 年 7 月 3 日
AmazonSSMManagedEC2InstanceDefaultPolicy – 新政策	Systems Manager 新增新政策，以允許 Amazon EC2 執行個體上的 Systems Manager 功能，無需使用 IAM 執行個體設定檔。	2022 年 8 月 18 日
AmazonSSMServiceRolePolicy：更新現有政策	Systems Manager 新增了新的許可，以允許 Explorer 在從 Explorer 或 OpsCenter 開啟 Security Hub 時建立受管規則。新增新的許可來檢查該組態，並在允許 OpsData 前，檢查 compute-optimizer 是否滿足必要要求。	2021 年 4 月 27 日
AWSSystemsManagerOpsDataSyncServiceRolePolicy – 新政策	Systems Manager 新增了新政策，以建立並更新 OpsItems 和來自 Explorer 和 OpsCenter Security Hub 問題清單的 OpsData。	2021 年 4 月 27 日
AmazonSSMServiceRolePolicy – 更新現有政策	Systems Manager 新增了新的許可，以允許檢視彙總 OpsData 和 Explorer 中多個帳戶和 AWS 區域 的 OpsItems 詳細資訊。	2021 年 3 月 24 日
Systems Manager 已開始追蹤變更	Systems Manager 已開始追蹤其 AWS 受管政策的變更。	2021 年 3 月 12 日

Systems Manager 的其他受管政策

除了本主題前文所述的受管政策，Systems Manager 也支援下列政策。

- AmazonSSMAutomationApproverAccess： AWS 受管政策，允許檢視自動化執行，以及將核准決策傳送到等待核准的自動化。
- AmazonSSMAutomationRole： AWS 受管政策，提供 Systems Manager Automation 服務的許可，以便執行 Automation 執行手冊中所定義的活動。將此政策指派給管理員和信任的高權限使用者。
- AmazonSSMDirectoryServiceAccess – AWS 受管政策，允許 AWS Directory Service 代表使用者SSM Agent存取 ，以請求由受管節點加入網域。
- AmazonSSMFullAccess – AWS 受管政策，授予 Systems Manager API 和文件的完整存取權。
- AmazonSSMMaintenanceWindowRole： AWS 受管政策，提供具有 Systems Manager API 許可的維護時段。
- AmazonSSMManagedInstanceCore： AWS 受管政策，讓節點可以使用 Systems Manager 服務的核心功能。
- AmazonSSMPatchAssociation： AWS 受管政策，提供對子執行個體的存取權，以便進行修補關聯操作。
- AmazonSSMReadOnlyAccess： AWS 受管政策，授予 `Get*` 和 `List*` 等 Systems Manager 唯讀 API 操作的存取權。
- AWSSSMOpsInsightsServiceRolePolicy – AWS 受管政策，提供在 中建立和更新營運洞見 OpsItems 的許可Systems Manager。用於透過服務連結角色 AWSServiceRoleForAmazonSSM_OpsInsights 提供許可。
- AWSSystemsManagerAccountDiscoveryServicePolicy – AWS 受管政策，授予 Systems Manager 探索 AWS 帳戶 資訊的許可。
- AWSSystemsManagerChangeManagementServicePolicy – AWS 受管政策，提供存取由Systems Manager變更管理架構所管理或使用，並由服務連結角色 使用 AWS 的資源`AWSServiceRoleForSystemsManagerChangeManagement`。
- AmazonEC2RoleforSSM：不再支援也不應使用此政策。請改用 `AmazonSSMManagedInstanceCore` 政策，以便在 EC2 執行個體上允許 Systems Manager 服務核心功能。如需相關資訊，請參閱設定 Systems Manager 所需的執行個體許可。

對 AWS Systems Manager 身分與存取進行疑難排解

使用下列資訊來協助您診斷和修正使用 AWS Systems Manager 和 AWS Identity and Access Management (IAM) 時可能遇到的常見問題。

主題

- 我未獲授權，不得在 Systems Manager 中執行動作
- 我未獲得執行 iam:PassRole 的授權
- 我想要允許 以外的人員 AWS 帳戶 存取我的Systems Manager資源

我未獲授權，不得在 Systems Manager 中執行動作

如果 AWS Management Console 告訴您未獲授權執行 動作，則必須聯絡管理員尋求協助。您的管理員是為您提供簽署憑證的人員。

以下範例錯誤會在 `mateojackson` 使用者嘗試使用主控台檢視文件的詳細資訊，但卻沒有 `ssm:GetDocument` 許可時發生。

```
User: arn:aws:ssm::123456789012:user/mateojackson isn't authorized to perform:
 ssm:GetDocument on resource: MyExampleDocument
```

在此情況下，Mateo 會請求管理員更新他的政策，允許他使用 `MyExampleDocument` 動作存取 `ssm:GetDocument` 資源。

我未獲得執行 iam:PassRole 的授權

如果您收到錯誤，告知您未獲授權執行 `iam:PassRole` 動作，您的政策必須更新，允許您將角色傳遞給 Systems Manager。

有些 AWS 服務 可讓您將現有角色傳遞給該服務，而不是建立新的服務角色或服務連結角色。如需執行此作業，您必須擁有將角色傳遞至該服務的許可。

名為 `marymajor` 的 IAM 使用者嘗試使用主控台在 Systems Manager 中執行動作時，發生下列範例錯誤。但是，動作要求服務具備服務角色授予的許可。Mary 沒有將角色傳遞至該服務的許可。

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

在這種情況下，Mary 的政策必須更新，允許她執行 `iam:PassRole` 動作。

如果您需要協助，請聯絡您的 AWS 管理員。您的管理員提供您的簽署憑證。

我想要允許 以外的人員 AWS 帳戶 存取我的Systems Manager資源

您可以建立一個角色，讓其他帳戶中的使用者或您組織外部的人員存取您的資源。您可以指定要允許哪些信任物件取得該角色。針對支援基於資源的政策或存取控制清單 (ACL) 的服務，您可以使用那些政策來授予人員存取您的資源的許可。

如需進一步了解，請參閱以下內容：

- 若要了解 Systems Manager 是否支援這些功能，請參閱 AWS Systems Manager 搭配 IAM 的運作方式。
- 若要了解如何 AWS 帳戶 在您擁有的 資源間提供存取權，請參閱《IAM 使用者指南》中的在您擁有 AWS 帳戶 的另一個資源中提供存取權給 IAM 使用者。
- 若要了解如何將資源的存取權提供給第三方 AWS 帳戶，請參閱《IAM 使用者指南》中的提供存取權給第三方 AWS 帳戶 擁有。
- 如需了解如何透過聯合身分提供存取權，請參閱 IAM 使用者指南中的將存取權提供給在外部進行身分驗證的使用者 (聯合身分)。
- 如需了解使用角色和資源型政策進行跨帳戶存取之間的差異，請參閱《IAM 使用者指南》中的 IAM 中的跨帳戶資源存取。

使用 Systems Manager 的服務連結角色

AWS Systems Manager use AWS Identity and Access Management (IAM) 服務連結角色。服務連結角色是直接連結至 Systems Manager 的一種特殊 IAM 角色類型。服務連結角色由 Systems Manager 預先定義，且內含該服務代您呼叫其他 AWS 服務 所需的所有許可。

> **Note**
>
> 服務角色與服務連結角色不同。服務角色是一種 AWS Identity and Access Management (IAM) 角色，可授予 許可， AWS 服務 讓服務可以存取 AWS 資源。只有幾個 Systems Manager 案例需要服務角色。當您建立 Systems Manager 的服務角色時，您可以選擇要授予的許可，以便它能夠存取或與其他 AWS 資源互動。

服務連結的角色可讓設定 Systems Manager 更為簡單，因為您不必手動新增必要的許可。Systems Manager 定義其服務連結角色的許可，除非另有定義，否則僅有 Systems Manager 可以擔任其角色。定義的許可包括信任政策和許可政策，並且該許可政策不能連接到任何其他 IAM 實體。

您必須先刪除服務連結角色的相關資源，才能將其刪除。如此可保護您 Systems Manager 的資源，避免您不小心移除資源的存取許可。

> **Note**
>
> 在混合多雲端環境中，您的非 EC2 節點需要其他 IAM 角色，讓這些機器能夠與 Systems Manager 服務通訊。這是 Systems Manager 的 IAM 服務角色。此角色會將 AWS Security Token Service (AWS STS) AssumeRole 信任授予Systems Manager服務。`AssumeRole` 動作會傳回一組臨時的安全登入資料 (包括存取金鑰 ID、私密存取金鑰和安全字符)。您可以使用這些臨時登入資料來存取您通常無法存取 AWS 的資源。如需詳細資訊，請參閱《AWS Security Token Service API 參考》中的在混合多雲端環境中建立 Systems Manager 所需的 IAM 服務角色和 AssumeRole。

如需關於支援服務連結角色的其他服務資訊，請參閱可搭配 IAM 運作的AWS 服務，尋找 Service-linked roles (服務連結角色) 欄中顯示為 Yes (是) 的服務。選擇具有連結的是，以檢視該服務的服務連結角色文件。

主題

- 使用角色來收集庫存和查看 OpsData
- 使用 角色來收集 OpsCenter和 AWS 帳戶 的資訊 Explorer
- 使用角色為 OpsItems 建立 OpsData 和 Explorer：
- 使用角色在 Systems Manager OpsCenter 中建立操作洞察 OpsItems
- 使用角色來維護 Quick Setup 佈建的資源運作狀態和一致性
- 使用角色匯出 Explorer OpsData
- 使用 角色來啟用just-in-time節點存取
- 使用 角色傳送just-in-time節點存取請求通知

使用角色來收集庫存和查看 OpsData

Systems Manager 使用名為 的服務連結角色**`AWSServiceRoleForAmazonSSM`**。 AWS Systems Manager 使用此 IAM 服務角色代表您管理 AWS 資源。

用於庫存、OpsData 和 OpsItems 的服務連結角色許可

AWSServiceRoleForAmazonSSM 服務連結角色僅信任 ssm.amazonaws.com 服務擔任此角色。

您可以將 Systems Manager 服務連結角色 AWSServiceRoleForAmazonSSM 用於下列功能：

- Systems Manager 庫存工具使用服務連結角色AWSServiceRoleForAmazonSSM，從標籤和資源群組收集庫存中繼資料。
- 此Explorer工具使用服務連結角色AWSServiceRoleForAmazonSSM來啟用檢視 OpsData 和 OpsItems來自多個帳戶的 。當您從 Explorer 或 OpsCenter 中啟用 Security Hub 作為資料來源時，此服務連結角色也允許 Explorer 建立受管規則。

> **Important**
>
> 先前，Systems Manager 主控台可讓您選擇要AWSServiceRoleForAmazonSSM用作任務維護角色的 AWS 受管 IAM 服務連結角色。不再建議將此角色及其關聯政策 AmazonSSMServiceRolePolicy，用於維護時段任務。如果您現在將此角色用於維護時段任務，我們建議您停止使用。相反，請建立您自己的 IAM 角色，以便在執行維護時段任務時，Systems Manager 可跟其他 AWS 服務 溝通。
> 如需詳細資訊，請參閱設定 Maintenance Windows。

用於為 AWSServiceRoleForAmazonSSM 角色提供許可的受管政策是 AmazonSSMServiceRolePolicy。如需授予許可的詳細資訊，請參閱 AWS 受管政策：AmazonSSMServiceRolePolicy。

建立 Systems Manager 的 **AWSServiceRoleForAmazonSSM** 服務連結角色

您可以在 IAM 主控台透過 EC2 使用案例建立服務連結角色。使用 AWS Command Line Interface (AWS CLI) 中的 IAM 命令或使用 IAM API，建立使用 ssm.amazonaws.com 服務名稱的服務連結角色。如需詳細資訊，請參閱《IAM 使用者指南》中的「建立服務連結角色」。

若您刪除此服務連結角色，之後需要再次建立，您可以在帳戶中使用相同程序重新建立角色。

編輯 Systems Manager 的 **AWSServiceRoleForAmazonSSM** 服務連結角色

Systems Manager 不允許您編輯 AWSServiceRoleForAmazonSSM 服務連結角色。因為可能有各種實體會參考服務連結角色，所以您無法在建立角色之後變更其名稱。然而，您可使用 IAM 來編輯角色描述。如需詳細資訊，請參閱「IAM 使用者指南」的編輯服務連結角色。

刪除 Systems Manager 的 **AWSServiceRoleForAmazonSSM** 服務連結角色

若您不再使用需要服務連結角色的任何功能或服務，我們建議您刪除該角色。如此一來，您就沒有未主動監控或維護的未使用實體。您可以使用 IAM 主控台 AWS CLI、 或 IAM API 來手動刪除服務連結角色。若要執行此操作，您必須先手動清除服務連結角色的資源，然後才能手動刪除它。

由於AWSServiceRoleForAmazonSSM服務連結角色可以由多個工具使用，因此在嘗試刪除角色之前，請確保沒有使用該角色。

- 庫存：如果您刪除庫存工具使用的服務連結角色，則標籤和資源群組的庫存資料將不再同步。手動刪除服務連結角色之前，您必須先清理資源。
- Explorer：如果您刪除Explorer工具使用的服務連結角色，則跨帳戶和跨區域 OpsData OpsItems 將無法再檢視。

> **Note**
>
> 如果在您嘗試刪除標籤或資源組時 Systems Manager 服務正在使用該角色，則刪除可能會失敗。若此情況發生，請等待數分鐘後並再次嘗試操作。

刪除 **AWSServiceRoleForAmazonSSM** 所使用的 Systems Manager 資源

1. 若要刪除標籤，請參閱在個別資源上新增和刪除標籤。
2. 若要刪除資源群組，請參閱從中刪除群組 AWS Resource Groups。

若要使用 IAM 手動刪除 **AWSServiceRoleForAmazonSSM** 服務連結角色

使用 IAM 主控台 AWS CLI、 或 IAM API 來刪除AWSServiceRoleForAmazonSSM服務連結角色。如需詳細資訊，請參閱「IAM 使用者指南」中的刪除服務連結角色。

支援 Systems Manager **AWSServiceRoleForAmazonSSM** 服務連結角色的區域

Systems Manager 支援在所有提供服務 AWS 區域 的 中使用AWSServiceRoleForAmazonSSM服務連結角色。如需詳細資訊，請參閱 AWS Systems Manager 端點和配額。

使用 角色來收集 OpsCenter和 AWS 帳戶 的資訊 Explorer

Systems Manager 使用名為 的服務連結角色**AWSServiceRoleForAmazonSSM_AccountDiscovery**。 AWS Systems Manager 使用此 IAM 服務角色呼叫其他 AWS 服務 來探索 AWS 帳戶 資訊。

適用於 Systems Manager 帳戶探索的服務連結角色許可

AWSServiceRoleForAmazonSSM_AccountDiscovery 服務連結角色信任下列服務以擔任角色：

- accountdiscovery.ssm.amazonaws.com

此角色許可政策允許 Systems Manager 對指定資源完成下列動作：

- organizations:DescribeAccount
- organizations:DescribeOrganizationalUnit
- organizations:DescribeOrganization
- organizations:ListAccounts
- organizations:ListAWSServiceAccessForOrganization
- organizations:ListChildren
- organizations:ListParents
- organizations:ListDelegatedServicesForAccount
- organizations:ListDelegatedAdministrators
- organizations:ListRoots

您必須設定許可，IAM 實體 (如使用者、群組或角色) 才可建立、編輯或刪除服務連結角色。如需詳細資訊，請參閱 IAM 使用者指南中的服務連結角色許可。

建立 Systems Manager 的 **AWSServiceRoleForAmazonSSM_AccountDiscovery** 服務連結角色

如果您想要跨多個 使用 Systems Manager 中的 Explorer和 OpsCenter工具，則必須建立服務連結角色 AWS 帳戶。若為 OpsCenter，您必須手動建立服務連結角色。如需詳細資訊，請參閱(選用) 手動設定 OpsCenter 以跨帳戶集中管理 OpsItems。

若為 Explorer，如果透過在 AWS Management Console中使用 Systems Manager 來建立資源資料同步，則您可以選擇 Create role (建立角色) 按鈕來建立服務連結角色。如果您想要以程式設計方式建立

資源資料同步，則必須先建立角色，再建立資源資料同步。您可以使用 CreateServiceLinkedRole API 操作建立角色。

編輯 Systems Manager 的 **AWSServiceRoleForAmazonSSM_AccountDiscovery** 服務連結角色

Systems Manager 不允許您編輯 AWSServiceRoleForAmazonSSM_AccountDiscovery 服務連結角色。因為可能有各種實體會參考服務連結角色，所以您無法在建立角色之後變更其名稱。然而，您可使用 IAM 來編輯角色描述。如需詳細資訊，請參閱「IAM 使用者指南」的編輯服務連結角色。

刪除 Systems Manager 的 **AWSServiceRoleForAmazonSSM_AccountDiscovery** 服務連結角色

若您不再使用需要服務連結角色的功能或服務，我們建議您刪除該角色。如此一來，您就沒有未主動監控或維護的未使用實體。然而，務必清除您的服務連結角色，之後才能以手動方式將其刪除。

清除 **AWSServiceRoleForAmazonSSM_AccountDiscovery** 服務連結角色

您必須先刪除所有 Explorer 資源資料同步，才能使用 IAM 刪除 AWSServiceRoleForAmazonSSM_AccountDiscovery 服務連結角色。

> **Note**
>
> 若 Systems Manager 服務在您試圖刪除資源時正在使用該角色，刪除可能會失敗。若此情況發生，請等待數分鐘後並再次嘗試操作。

手動刪除 **AWSServiceRoleForAmazonSSM_AccountDiscovery** 服務連結角色

使用 IAM 主控台 AWS CLI、 或 AWS API 來刪除AWSServiceRoleForAmazonSSM_AccountDiscovery服務連結角色。如需詳細資訊，請參閱「IAM 使用者指南」中的刪除服務連結角色。

支援 Systems Manager **AWSServiceRoleForAmazonSSM_AccountDiscovery** 服務連結角色的區域

Systems Manager 支援在所有提供服務的區域中使用服務連結角色。如需詳細資訊，請參閱 AWS Systems Manager 端點和配額。

AWSServiceRoleForAmazonSSM_AccountDiscovery 服務連接角色更新

檢視自從此服務開始追蹤這些變更以來對 AWSServiceRoleForAmazonSSM_AccountDiscovery 服務連結角色的更新的詳細資訊。如需有關此頁面變更的自動提醒，請訂閱 Systems Manager 文件歷史記錄 頁面的 RSS 摘要。

變更	描述	日期
已新增新許可	此服務連結角色現在包含 `organizations:DescribeOrganizationalUnit` 和 `organizations:ListRoots` 許可。這些許可可讓 AWS Organizations 管理帳戶或 Systems Manager 委派管理員帳戶OpsItems跨帳戶使用 。如需詳細資訊，請參閱(選用) 手動設定 OpsCenter 以跨帳戶集中管理 OpsItems。	2022 年 10 月 17 日

使用角色為 OpsItems 建立 OpsData 和 Explorer：

Systems Manager 使用名為 的服務連結角色**`AWSServiceRoleForSystemsManagerOpsDataSync`**。 AWS Systems Manager 使用此 IAM 服務角色Explorer來建立 OpsData 和 OpsItems。

適用於 Systems Manager OpsData 資料同步的服務連結角色許可

`AWSServiceRoleForSystemsManagerOpsDataSync` 服務連結角色信任下列服務以擔任角色：

- `opsdatasync.ssm.amazonaws.com`

此角色許可政策允許 Systems Manager 對指定資源完成下列動作：

- Systems Manager Explorer 需要服務連結角色來授予許可，以便在更新 OpsItem 時更新安全問題清單、建立和更新 OpsItem，以及在客戶刪除 SSM 受管規則時關閉 Security Hub 資料來源。

用於為 AWSServiceRoleForSystemsManagerOpsDataSync 角色提供許可的受管政策是 AWSSystemsManagerOpsDataSyncServiceRolePolicy。如需授予許可的詳細資訊，請參閱 AWS 受管政策：AWSSystemsManagerOpsDataSyncServiceRolePolicy。

您必須設定許可，IAM 實體 (如使用者、群組或角色) 才可建立、編輯或刪除服務連結角色。如需詳細資訊，請參閱 IAM 使用者指南中的服務連結角色許可。

建立 Systems Manager 的 **AWSServiceRoleForSystemsManagerOpsDataSync** 服務連結角色

您不需要手動建立一個服務連結角色。當您在 Explorer中啟用 時 AWS Management Console， 會為您Systems Manager建立服務連結角色。

> **Important**
>
> 此服務連結的角色可以顯示在您的帳戶，如果您於其他服務中完成一項動作時，可以使用支援此角色的功能。另外，若您在 2017 年 1 月 1 日之前使用 Systems Manager 服務 (那時起開始支援服務連結角色)，Systems Manager 會於您的帳戶中建立 AWSServiceRoleForSystemsManagerOpsDataSync 角色。若要進一步了解，請參閱我的 IAM 帳戶中出現的新角色。

若您刪除此服務連結角色，之後需要再次建立，您可以在帳戶中使用相同程序重新建立角色。當您在 Explorer中啟用 時 AWS Management Console， 會再次為您Systems Manager建立服務連結角色。

您也可以使用 IAM 主控台透過允許 Explorer 建立 OpsData 和 OpsItems 使用案例的 AWS 服務角色，建立服務連結角色。在 AWS CLI 或 AWS API 中，使用服務名稱建立opsdatasync.ssm.amazonaws.com服務連結角色。如需詳細資訊，請參閱《IAM 使用者指南》中的「建立服務連結角色」。如果您刪除此服務連結角色，您可以使用此相同的程序以再次建立該角色。

編輯 Systems Manager 的 **AWSServiceRoleForSystemsManagerOpsDataSync** 服務連結角色

Systems Manager 不允許您編輯 AWSServiceRoleForSystemsManagerOpsDataSync 服務連結角色。因為可能有各種實體會參考服務連結角色，所以您無法在建立角色之後變更其名稱。然而，您可使用 IAM 來編輯角色描述。如需詳細資訊，請參閱「IAM 使用者指南」的編輯服務連結角色。

刪除 Systems Manager 的 **AWSServiceRoleForSystemsManagerOpsDataSync** 服務連結角色

若您不再使用需要服務連結角色的功能或服務，我們建議您刪除該角色。如此一來，您就沒有未主動監控或維護的未使用實體。然而，在手動刪除服務連結角色之前，您必須先清除資源。

> **Note**
>
> 若 Systems Manager 服務在您試圖刪除資源時正在使用該角色，刪除可能會失敗。若此情況發生，請等待數分鐘後並再次嘗試操作。

該刪除 `AWSServiceRoleForSystemsManagerOpsDataSync` 角色所使用之 Systems Manager 資源的處理程序取決於您是否已設定 Explorer 或 OpsCenter 與 Security Hub 整合。

刪除 **AWSServiceRoleForSystemsManagerOpsDataSync** 角色所使用的 Systems Manager 資源

- 若要停止 Explorer 為 Security Hub 問題清單建立新的 OpsItems，請參閱 如何停止接收調查結果。
- 若要停止 OpsCenter 為 Security Hub 問題清單建立新的 OpsItems，請參閱

若要使用 IAM 手動刪除 **AWSServiceRoleForSystemsManagerOpsDataSync** 服務連結角色

使用 IAM 主控台 AWS CLI、 或 AWS API 來刪除`AWSServiceRoleForSystemsManagerOpsDataSync`服務連結角色。如需詳細資訊，請參閱「IAM 使用者指南」中的刪除服務連結角色。

支援 Systems Manager **AWSServiceRoleForSystemsManagerOpsDataSync** 服務連結角色的區域

Systems Manager 支援在所有提供服務的區域中使用服務連結角色。如需詳細資訊，請參閱 AWS Systems Manager 端點和配額。

Systems Manager 不支援在每一個提供服務的區域中使用服務連結角色。您可以在下列區域中使用 `AWSServiceRoleForSystemsManagerOpsDataSync` 角色。

AWS 區域 名稱	區域身分	在 Systems Manager 中支援
美國東部 (維吉尼亞北部)	us-east-1	是

AWS 區域 名稱	區域身分	在 Systems Manager 中支援
美國東部 (俄亥俄)	us-east-2	是
美國西部 (加利佛尼亞北部)	us-west-1	是
美國西部 (奧勒岡)	us-west-2	是
亞太區域 (孟買)	ap-south-1	是
亞太區域 (大阪)	ap-northeast-3	是
亞太區域 (首爾)	ap-northeast-2	是
亞太區域 (新加坡)	ap-southeast-1	是
亞太區域 (雪梨)	ap-southeast-2	是
亞太區域 (東京)	ap-northeast-1	是
加拿大 (中部)	ca-central-1	是
歐洲 (法蘭克福)	eu-central-1	是
歐洲 (愛爾蘭)	eu-west-1	是
歐洲 (倫敦)	eu-west-2	是
歐洲 (巴黎)	eu-west-3	是
歐洲 (斯德哥爾摩)	eu-north-1	是
南美洲 (聖保羅)	sa-east-1	是
AWS GovCloud (US)	us-gov-west-1	否

使用角色在 Systems Manager OpsCenter 中建立操作洞察 OpsItems

Systems Manager 使用名為 的服務連結角色**AWSServiceRoleForAmazonSSM_OpsInsights**。AWS Systems Manager 使用此 IAM 服務角色，在 Systems Manager 中建立和更新營運洞見 OpsItemsOpsCenter。

用於 Systems Manager 操作洞察 **AWSServiceRoleForAmazonSSM_OpsInsights** 的服務連結角色許可

AWSServiceRoleForAmazonSSM_OpsInsights 服務連結角色信任下列服務以擔任角色：

- opsinsights.ssm.amazonaws.com

此角色許可政策允許 Systems Manager 對指定資源完成下列動作：

```
{
 "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"
    }
   }
  }
 ]
```

```
}
```

您必須設定許可，IAM 實體 (如使用者、群組或角色) 才可建立、編輯或刪除服務連結角色。如需詳細資訊，請參閱 IAM 使用者指南中的服務連結角色許可。

建立 Systems Manager 的 **AWSServiceRoleForAmazonSSM_OpsInsights** 服務連結角色

您必須建立服務連結角色。如果您在 Systems Manager 中使用 啟用操作洞察 AWS Management Console，您可以選擇啟用按鈕來建立服務連結角色。

編輯 Systems Manager 的 **AWSServiceRoleForAmazonSSM_OpsInsights** 服務連結角色

Systems Manager 不允許您編輯 AWSServiceRoleForAmazonSSM_OpsInsights 服務連結角色。因為有各種實體可能會參考服務連結角色，所以您無法在建立角色之後變更角色名稱。然而，您可使用 IAM 來編輯角色描述。如需詳細資訊，請參閱「IAM 使用者指南」的編輯服務連結角色。

刪除 Systems Manager 的 **AWSServiceRoleForAmazonSSM_OpsInsights** 服務連結角色

若您不再使用需要服務連結角色的功能或服務，我們建議您刪除該角色。如此一來，您就沒有未主動監控或維護的未使用實體。然而，務必清除您的服務連結角色，之後才能以手動方式將其刪除。

清除 **AWSServiceRoleForAmazonSSM_OpsInsights** 服務連結角色

您必須先在 Systems Manager OpsCenter 中停用操作洞察，才能使用 IAM 刪除 AWSServiceRoleForAmazonSSM_OpsInsights 服務連結角色。如需詳細資訊，請參閱分析操作洞察以減少 OpsItems。

手動刪除 **AWSServiceRoleForAmazonSSM_OpsInsights** 服務連結角色

使用 IAM 主控台 AWS CLI、 或 AWS API 來刪除AWSServiceRoleForAmazonSSM_OpsInsights服務連結角色。如需詳細資訊，請參閱「IAM 使用者指南」中的刪除服務連結角色。

支援 Systems Manager **AWSServiceRoleForAmazonSSM_OpsInsights** 服務連結角色的區域

Systems Manager 不支援在提供服務的每個區域中使用服務連結角色。您可以在下列區域使用 AWSServiceRoleForAmazonSSM_OpsInsights 角色。

區域名稱	區域身分	在 Systems Manager 中支援
美國東部 (維吉尼亞北部)	us-east-1	是
美國東部 (俄亥俄)	us-east-2	是
美國西部 (加利佛尼亞北部)	us-west-1	是
美國西部 (奧勒岡)	us-west-2	是
亞太區域 (孟買)	ap-south-1	是
亞太區域 (東京)	ap-northeast-1	是
亞太區域 (首爾)	ap-northeast-2	是
亞太區域 (新加坡)	ap-southeast-1	是
亞太區域 (雪梨)	ap-southeast-2	是
亞太區域 (香港)	ap-east-1	是
加拿大 (中部)	ca-central-1	是
歐洲 (法蘭克福)	eu-central-1	是
歐洲 (愛爾蘭)	eu-west-1	是
歐洲 (倫敦)	eu-west-2	是
歐洲 (巴黎)	eu-west-3	是
歐洲 (斯德哥爾摩)	eu-north-1	是
歐洲 (米蘭)	eu-south-1	是
南美洲 (聖保羅)	sa-east-1	是
中東 (巴林)	me-south-1	是
非洲 (開普敦)	af-south-1	是

區域名稱	區域身分	在 Systems Manager 中支援
AWS GovCloud (US)	us-gov-west-1	是
AWS GovCloud (US)	us-gov-east-1	是

使用角色來維護 Quick Setup 佈建的資源運作狀態和一致性

Systems Manager 使用名為 **AWSServiceRoleForSSMQuickSetup** 的服務連結角色。

Systems Manager 的 **AWSServiceRoleForSSMQuickSetup** 服務連結角色許可

AWSServiceRoleForSSMQuickSetup 服務連結角色信任下列服務以擔任角色：

- ssm-quicksetup.amazonaws.com

AWS Systems Manager 使用此 IAM 服務角色來檢查組態運作狀態、確保參數和佈建資源的一致使用，以及在偵測到偏離時修復資源。

此角色許可政策允許 Systems Manager 對指定資源完成下列動作：

- ssm (Systems Manager)：讀取已設定資源預期狀態的相關資訊 (包括在委派管理員帳戶中)。
- iam (AWS Identity and Access Management)：這是在 AWS Organizations中跨整個組織存取資源資料同步的必要條件。
- organizations (AWS Organizations)：如 Organizations 中所設定，讀取屬於組織成員帳戶的相關資訊。
- cloudformation (AWS CloudFormation)：讀取用於管理資源狀態和 CloudFormation 堆疊集操作的 CloudFormation 堆疊相關資訊。

用於為 AWSServiceRoleForSSMQuickSetup 角色提供許可的受管政策是 SSMQuickSetupRolePolicy。如需授予許可的詳細資訊，請參閱 AWS 受管政策：SSMQuickSetupRolePolicy。

您必須設定許可，IAM 實體 (如使用者、群組或角色) 才可建立、編輯或刪除服務連結角色。如需詳細資訊，請參閱 IAM 使用者指南中的服務連結角色許可。

建立 Systems Manager 的 **AWSServiceRoleForSSMQuickSetup** 服務連結角色

您不需要手動建立 AWSServiceRoleForSSMQuickSetup 服務連結角色。如果在 AWS Management Console建立 Quick Setup 組態，Systems Manager 會為您建立服務連結角色。

編輯 Systems Manager 的 **AWSServiceRoleForSSMQuickSetup** 服務連結角色

Systems Manager 不允許您編輯 AWSServiceRoleForSSMQuickSetup 服務連結角色。因為有各種實體可能會參考服務連結角色，所以您無法在建立角色之後變更角色名稱。然而，您可使用 IAM 來編輯角色描述。如需詳細資訊，請參閱「IAM 使用者指南」的編輯服務連結角色。

刪除 Systems Manager 的 **AWSServiceRoleForSSMQuickSetup** 服務連結角色

若您不再使用需要服務連結角色的功能或服務，我們建議您刪除該角色。如此一來，您就沒有未主動監控或維護的未使用實體。然而，務必清除您的服務連結角色，之後才能以手動方式將其刪除。

清除 **AWSServiceRoleForSSMQuickSetup** 服務連結角色

您必須先刪除使用該角色的 Quick Setup 組態，才可以使用 IAM 刪除 AWSServiceRoleForSSMQuickSetup 服務連結角色。如需詳細資訊，請參閱編輯和刪除您的組態。

手動刪除 **AWSServiceRoleForSSMQuickSetup** 服務連結角色

使用 IAM 主控台 AWS CLI、 或 AWS API 來刪除AWSServiceRoleForSSMQuickSetup服務連結角色。如需詳細資訊，請參閱下列主題：

- 《IAM 使用者指南》中的刪除服務連結角色
- 《AWS CLI 參考》中 Quick Setup 部分的 delete-configuration-manager
- Quick Setup API 參考中的 DeleteConfigurationManager

支援 Systems Manager **AWSServiceRoleForSSMQuickSetup** 服務連結角色的區域

Systems Manager 不支援在提供服務的每個區域中使用服務連結角色。您可以在下列區域使用 AWSServiceRoleForSSMQuickSetup 角色。

- 美國東部 (俄亥俄)
- 美國東部 (維吉尼亞北部)
- 美國西部 (加利佛尼亞北部)

- 美國西部 (奧勒岡)
- 亞太區域 (孟買)
- 亞太區域 (首爾)
- 亞太區域 (新加坡)
- 亞太區域 (雪梨)
- 亞太區域 (東京)
- 加拿大 (中部)
- 歐洲 (法蘭克福)
- 歐洲 (斯德哥爾摩)
- 歐洲 (愛爾蘭)
- 歐洲 (倫敦)
- Europe (Paris)
- 南美洲 (聖保羅)

使用角色匯出 Explorer OpsData

AWS Systems Manager Explorer 使用 AmazonSSMExplorerExportRole 服務角色，使用AWS-ExportOpsDataToS3自動化 Runbook 匯出操作資料 (OpsData)。

Explorer 的服務連結角色許可

AmazonSSMExplorerExportRole 服務連結角色僅信任 ssm.amazonaws.com 服務擔任此角色。

您可以使用 AmazonSSMExplorerExportRole 服務連結角色，以 AWS-ExportOpsDataToS3 Automation 執行手冊匯出操作資料 (OpsData)。您可以從 Explorer 將 5,000 個 OpsData 項目以逗號分隔值 (.csv) 檔案格式匯出至 Amazon Simple Storage Service (Amazon S3) 儲存貯體。

此角色許可政策允許 Systems Manager 對指定資源完成下列動作：

- s3:PutObject
- s3:GetBucketAcl
- s3:GetBucketLocation
- sns:Publish
- logs:DescribeLogGroups
- logs:DescribeLogStreams

- logs:CreateLogGroup
- logs:PutLogEvents
- logs:CreateLogStream
- ssm:GetOpsSummary

您必須設定許可，IAM 實體 (如使用者、群組或角色) 才可建立、編輯或刪除服務連結角色。如需詳細資訊，請參閱 IAM 使用者指南中的服務連結角色許可。

建立 Systems Manager 的 **AmazonSSMExplorerExportRole** 服務連結角色

當您在 Systems Manager 主控台中使用 Explorer 匯出 OpsData 時，Systems Manager 會建立 AmazonSSMExplorerExportRole 服務連結角色。如需詳細資訊，請參閱從 Systems Manager Explorer 匯出 OpsData。

若您刪除此服務連結角色，之後需要再次建立，您可以在帳戶中使用相同程序重新建立角色。

編輯 Systems Manager 的 **AmazonSSMExplorerExportRole** 服務連結角色

Systems Manager 不允許您編輯 AmazonSSMExplorerExportRole 服務連結角色。因為可能有各種實體會參考服務連結角色，所以您無法在建立角色之後變更其名稱。然而，您可使用 IAM 來編輯角色描述。如需詳細資訊，請參閱「IAM 使用者指南」的編輯服務連結角色。

刪除 Systems Manager 的 **AmazonSSMExplorerExportRole** 服務連結角色

若您不再使用需要服務連結角色的任何功能或服務，我們建議您刪除該角色。如此一來，您就沒有未主動監控或維護的未使用實體。您可以使用 IAM 主控台 AWS CLI、 或 IAM API 來手動刪除服務連結角色。若要執行此操作，您必須先手動清除服務連結角色的資源，然後才能手動刪除它。

> **Note**
>
> 如果在您嘗試刪除標籤或資源組時 Systems Manager 服務正在使用該角色，則刪除可能會失敗。若此情況發生，請等待數分鐘後並再次嘗試操作。

刪除 **AmazonSSMExplorerExportRole** 所使用的 Systems Manager 資源

1. 若要刪除標籤，請參閱在個別資源上新增和刪除標籤。
2. 若要刪除資源群組，請參閱從中刪除群組 AWS Resource Groups。

若要使用 IAM 手動刪除 **AmazonSSMExplorerExportRole** 服務連結角色

使用 IAM 主控台 AWS CLI、 或 IAM API 來刪除AmazonSSMExplorerExportRole服務連結角色。如需詳細資訊，請參閱「IAM 使用者指南」中的刪除服務連結角色。

支援 Systems Manager **AmazonSSMExplorerExportRole** 服務連結角色的區域

Systems Manager 支援在所有提供服務 AWS 區域 的 中使用AmazonSSMExplorerExportRole服務連結角色。如需詳細資訊，請參閱 AWS Systems Manager 端點和配額。

使用 角色來啟用just-in-time節點存取

Systems Manager 使用名為 的服務連結角色**AWSServiceRoleForSystemsManagerJustInTimeAccess**。 AWS Systems Manager 使用此 IAM 服務角色來啟用just-in-time節點存取。

Systems Manager just-in-time節點存取的服務連結角色許可

AWSServiceRoleForSystemsManagerJustInTimeAccess 服務連結角色信任下列服務以擔任角色：

- ssm.amazonaws.com

此角色許可政策允許 Systems Manager 對指定資源完成下列動作：

- 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

用於為 AWSServiceRoleForSystemsManagerJustInTimeAccess 角色提供許可的受管政策是 AWSSystemsManagerEnableJustInTimeAccessPolicy。如需授予許可的詳細資訊，請參閱 AWS 受管政策：AWSSystemsManagerJustInTimeAccessServicePolicy。

您必須設定許可，IAM 實體 (如使用者、群組或角色) 才可建立、編輯或刪除服務連結角色。如需詳細資訊，請參閱 IAM 使用者指南中的服務連結角色許可。

建立 Systems Manager 的 **AWSServiceRoleForSystemsManagerJustInTimeAccess** 服務連結角色

您不需要手動建立一個服務連結角色。當您在 just-in-time節點存取時 AWS Management Console，會為您Systems Manager建立服務連結角色。

> **Important**
>
> 此服務連結的角色可以顯示在您的帳戶，如果您於其他服務中完成一項動作時，可以使用支援此角色的功能。此外，如果您在 2024 年 11 月 19 日之前使用Systems Manager服務，則當服務連結角色開始支援服務時， 會在您的帳戶中Systems Manager建立該AWSServiceRoleForSystemsManagerJustInTimeAccess角色。若要進一步了解，請參閱我的 IAM 帳戶中出現的新角色。

若您刪除此服務連結角色，之後需要再次建立，您可以在帳戶中使用相同程序重新建立角色。當您在 just-in-time節點存取時 AWS Management Console， 會再次為您Systems Manager建立服務連結角色。

您也可以使用 IAM 主控台建立服務連結角色，其中包含AWS 允許 Systems Manager just-in-time節點存取的服務角色。 使用案例。在 AWS CLI 或 AWS API 中，使用服務名稱建立ssm.amazonaws.com服務連結角色。如需詳細資訊，請參閱《IAM 使用者指南》中的「建立服務連結角色」。如果您刪除此服務連結角色，您可以使用此相同的程序以再次建立該角色。

編輯 Systems Manager 的 **AWSServiceRoleForSystemsManagerJustInTimeAccess** 服務連結角色

Systems Manager 不允許您編輯 AWSServiceRoleForSystemsManagerJustInTimeAccess 服務連結角色。因為可能有各種實體會參考服務連結角色，所以您無法在建立角色之後變更其名稱。然而，您可使用 IAM 來編輯角色描述。如需詳細資訊，請參閱「IAM 使用者指南」的編輯服務連結角色。

刪除 Systems Manager 的 **AWSServiceRoleForSystemsManagerJustInTimeAccess** 服務連結角色

若您不再使用需要服務連結角色的功能或服務，我們建議您刪除該角色。如此一來，您就沒有未主動監控或維護的未使用實體。然而，在手動刪除服務連結角色之前，您必須先清除資源。

> **Note**
>
> 若 Systems Manager 服務在您試圖刪除資源時正在使用該角色，刪除可能會失敗。若此情況發生，請等待數分鐘後並再次嘗試操作。

若要使用 IAM 手動刪除 **AWSServiceRoleForSystemsManagerJustInTimeAccess** 服務連結角色

使用 IAM 主控台 AWS CLI、 或 AWS API 來刪除AWSServiceRoleForSystemsManagerJustInTimeAccess服務連結角色。如需詳細資訊，請參閱「IAM 使用者指南」中的刪除服務連結角色。

支援 Systems Manager **AWSServiceRoleForSystemsManagerJustInTimeAccess** 服務連結角色的區域

AWS 區域 名稱	區域身分	在 Systems Manager 中支援
美國東部 (維吉尼亞北部)	us-east-1	是
美國東部 (俄亥俄)	us-east-2	是
美國西部 (加利佛尼亞北部)	us-west-1	是

AWS 區域 名稱	區域身分	在 Systems Manager 中支援
美國西部 (奧勒岡)	us-west-2	是
亞太區域 (孟買)	ap-south-1	是
亞太區域 (大阪)	ap-northeast-3	是
亞太區域 (首爾)	ap-northeast-2	是
亞太區域 (新加坡)	ap-southeast-1	是
亞太區域 (雪梨)	ap-southeast-2	是
亞太區域 (東京)	ap-northeast-1	是
加拿大 (中部)	ca-central-1	是
歐洲 (法蘭克福)	eu-central-1	是
歐洲 (愛爾蘭)	eu-west-1	是
歐洲 (倫敦)	eu-west-2	是
歐洲 (巴黎)	eu-west-3	是
Europe (Stockholm)	eu-north-1	是
南美洲 (聖保羅)	sa-east-1	是
AWS GovCloud (US)	us-gov-west-1	否

使用 角色傳送just-in-time節點存取請求通知

Systems Manager 使用名為 的服務連結角色**AWSServiceRoleForSystemsManagerNotifications**。 AWS Systems Manager 使用此 IAM 服務角色傳送通知以存取請求核准者。

Systems Manager just-in-time節點存取通知的服務連結角色許可

AWSServiceRoleForSystemsManagerNotifications 服務連結角色信任下列服務以擔任角色：

- ssm.amazonaws.com

此角色許可政策允許 Systems Manager 對指定資源完成下列動作：

- identitystore:ListGroupMembershipsForMember
- identitystore:ListGroupMemberships
- identitystore:DescribeUser
- sso:ListInstances
- sso:DescribeRegisteredRegions
- sso:ListDirectoryAssociations
- sso-directory:DescribeUser
- sso-directory:ListMembersInGroup
- iam:GetRole

用於為 AWSServiceRoleForSystemsManagerNotifications 角色提供許可的受管政策是 AWSSystemsManagerNotificationsServicePolicy。如需授予許可的詳細資訊，請參閱 AWS 受管政策：AWSSystemsManagerNotificationsServicePolicy。

您必須設定許可，IAM 實體 (如使用者、群組或角色) 才可建立、編輯或刪除服務連結角色。如需詳細資訊，請參閱 IAM 使用者指南中的服務連結角色許可。

建立 Systems Manager 的 **AWSServiceRoleForSystemsManagerNotifications** 服務連結角色

您不需要手動建立一個服務連結角色。當您在 just-in-time節點存取時 AWS Management Console，會為您Systems Manager建立服務連結角色。

> **Important**
>
> 此服務連結的角色可以顯示在您的帳戶，如果您於其他服務中完成一項動作時，可以使用支援此角色的功能。此外，如果您在 2024 年 11 月 19 日之前使用Systems Manager

服務，則當服務連結角色開始支援服務時， 會在您的帳戶中Systems Manager建立該AWSServiceRoleForSystemsManagerNotifications角色。若要進一步了解，請參閱我的 IAM 帳戶中出現的新角色。

若您刪除此服務連結角色，之後需要再次建立，您可以在帳戶中使用相同程序重新建立角色。當您在just-in-time節點存取時 AWS Management Console， 會再次為您Systems Manager建立服務連結角色。

您也可以使用 IAM 主控台建立服務連結角色，其中包含AWS 允許 Systems Manager 傳送通知給存取請求核准者的服務角色。 使用案例。在 AWS CLI 或 AWS API 中，使用服務名稱建立ssm.amazonaws.com服務連結角色。如需詳細資訊，請參閱《IAM 使用者指南》中的「建立服務連結角色」。如果您刪除此服務連結角色，您可以使用此相同的程序以再次建立該角色。

編輯 Systems Manager 的 **AWSServiceRoleForSystemsManagerNotifications** 服務連結角色

Systems Manager 不允許您編輯 AWSServiceRoleForSystemsManagerNotifications 服務連結角色。因為可能有各種實體會參考服務連結角色，所以您無法在建立角色之後變更其名稱。然而，您可使用 IAM 來編輯角色描述。如需詳細資訊，請參閱「IAM 使用者指南」的編輯服務連結角色。

刪除 Systems Manager 的 **AWSServiceRoleForSystemsManagerNotifications** 服務連結角色

若您不再使用需要服務連結角色的功能或服務，我們建議您刪除該角色。如此一來，您就沒有未主動監控或維護的未使用實體。然而，在手動刪除服務連結角色之前，您必須先清除資源。

> **Note**
>
> 若 Systems Manager 服務在您試圖刪除資源時正在使用該角色，刪除可能會失敗。若此情況發生，請等待數分鐘後並再次嘗試操作。

若要使用 IAM 手動刪除 **AWSServiceRoleForSystemsManagerNotifications** 服務連結角色

使用 IAM 主控台 AWS CLI、 或 AWS API 來刪除AWSServiceRoleForSystemsManagerNotifications服務連結角色。如需詳細資訊，請參閱「IAM 使用者指南」中的刪除服務連結角色。

支援 Systems Manager **AWSServiceRoleForSystemsManagerNotifications** 服務連結角色的區域

AWS 區域 名稱	區域身分	在 Systems Manager 中支援
美國東部 (維吉尼亞北部)	us-east-1	是
美國東部 (俄亥俄)	us-east-2	是
美國西部 (加利佛尼亞北部)	us-west-1	是
美國西部 (奧勒岡)	us-west-2	是
亞太區域 (孟買)	ap-south-1	是
亞太區域 (大阪)	ap-northeast-3	是
亞太區域 (首爾)	ap-northeast-2	是
亞太區域 (新加坡)	ap-southeast-1	是
亞太區域 (雪梨)	ap-southeast-2	是
亞太區域 (東京)	ap-northeast-1	是
加拿大 (中部)	ca-central-1	是
歐洲 (法蘭克福)	eu-central-1	是
歐洲 (愛爾蘭)	eu-west-1	是
歐洲 (倫敦)	eu-west-2	是
歐洲 (巴黎)	eu-west-3	是
Europe (Stockholm)	eu-north-1	是
南美洲 (聖保羅)	sa-east-1	是
AWS GovCloud (US)	us-gov-west-1	否

AWS Systems Manager 中的日誌記錄和監控

監控是維護 和 AWS 解決方案可靠性、可用性 AWS Systems Manager 和效能的重要部分。您應該從 AWS 解決方案的所有部分收集監控資料，以便在發生多點失敗時進行更多偵錯。 AWS 提供數種工具來監控您的 Systems Manager和其他資源並回應潛在事件。

AWS CloudTrail 日誌

CloudTrail 會提供由使用者、角色或 AWS 服務 在 Systems Manager 中採取之動作的記錄。您可以利用 CloudTrail 所收集的資訊來判斷向 Systems Manager 發出的請求，以及發出請求的 IP 地址、人員、時間和其他詳細資訊。如需詳細資訊，請參閱使用 記錄 AWS Systems Manager API 呼叫 AWS CloudTrail。

Amazon CloudWatch 警示

使用 Amazon CloudWatch 警示，在一段時間內觀察您為 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體和其他資源指定的單一指標。如果指標超過指定的閾值，則會將通知傳送至 Amazon Simple Notification Service (Amazon SNS) 主題或 AWS Auto Scaling 政策。CloudWatch 警示不會因為它們處於特定狀態而叫用動作。必須是狀態已變更並維持了所指定的時間長度，才會呼叫動作。如需詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的使用 Amazon CloudWatch 警示。

Amazon CloudWatch 儀表板

CloudWatch 儀表板是 CloudWatch 主控台中可自訂的首頁，可讓您在單一檢視中監控資源，甚至是分散在不同 AWS 區域的那些資源。您可以使用 CloudWatch 儀表板來為 AWS 資源的指標和警示建立自訂檢視。如需詳細資訊，請參閱使用 Systems Manager 託管的 Amazon CloudWatch 儀表板。

Amazon EventBridge

使用 Amazon EventBridge，您可以設定規則，提醒您 Systems Manager 資源中發生變更，並指示 EventBridge 根據事件的內容採取動作。EventBridge 支援各種Systems Manager工具發出的許多事件。如需詳細資訊，請參閱使用 Amazon EventBridge 監控 Systems Manager。

Amazon CloudWatch Logs 和 SSM Agent 日誌

SSM Agent 會在各個節點的日誌檔寫入有關執行、排定動作、錯誤和運作狀態的資訊。您可以手動連線到節點來檢視日誌檔案。建議自動將代理程式日誌資料傳送至 CloudWatch Logs 中的日誌群組以進行分析。如需詳細資訊，請參閱 將節點日誌傳送至統一 CloudWatch Logs (CloudWatch 代理程式) 和 檢視 SSM Agent 日誌。

AWS Systems Manager 合規

您可以使用其中的工具 Compliance AWS Systems Manager來掃描受管節點機群的修補程式合規和組態不一致。您可以從多個 AWS 帳戶 和 收集和彙總資料 AWS 區域，然後深入研究不合規的特定資源。在預設情況下，合規會顯示有關在 中修補Patch Manager、在 中的工具 AWS Systems Manager，以及在 中State Manager建立關聯的目前合規資料 AWS Systems Manager。如需詳細資訊，請參閱AWS Systems Manager合規。

AWS Systems Manager Explorer

Explorer中的工具 是可自訂的操作儀表板 AWS Systems Manager，可報告 AWS 資源的相關資訊。 Explorer會顯示您 AWS 帳戶 和 之間的操作資料 (OpsData) 彙總檢視 AWS 區域。在 Explorer 中，OpsData 包含有關 EC2 執行個體、修補程式合規詳細資料和操作工作項目 (OpsItems) 的中繼資料。Explorer 提供脈絡指出 OpsItems 如何分散於業務單位或應用程式、如何隨著時間而演變趨勢，以及如何隨類別而變化。您可以在 Explorer 中群組和篩選資訊，以專注於與您相關且需要採取動作的項目。如需詳細資訊，請參閱AWS Systems Manager Explorer。

AWS Systems Manager OpsCenter

OpsCenter中的工具 提供中央位置 AWS Systems Manager，讓營運工程師和 IT 專業人員可以檢視、調查和解決與 AWS 資源相關的操作工作項目 (OpsItems)。 OpsItems會跨 服務OpsCenter彙總和標準化OpsItems，同時提供每個 OpsItem、相關 和相關資源的關聯式調查資料。 OpsCenter 也提供 Automation 中的執行手冊，這項工具 AWS Systems Manager可讓您快速解決問題。OpsCenter 已與 Amazon EventBridge 整合。這表示您可以建立 EventBridge 規則，為將事件發佈到 EventBridge OpsItems的任何 AWS 服務 自動建立。如需詳細資訊，請參閱AWS Systems Manager OpsCenter。

Amazon Simple Notification Service

您可以設定 Amazon Simple Notification Service (Amazon SNS) 傳送有關您使用 Run Command或 Maintenance Windows、 工具傳送之命令狀態的通知 AWS Systems Manager。Amazon SNS 會協調和管理傳送和傳遞通知給已訂閱 Amazon SNS 主題的用戶端或端點。每當命令變更為新狀態或特定狀態時 (如「`Failed`」或「`Timed Out`」)，您都會收到通知。當您將命令傳送至多個節點時，您都可以接收到傳送到特定節點之每個命令複本的通知。如需詳細資訊，請參閱使用 Amazon SNS 通知監控 Systems Manager 狀態變更。

AWS Trusted Advisor 而且 AWS Health Dashboard

Trusted Advisor 利用從服務數十萬 AWS 客戶中學到的最佳實務。 Trusted Advisor 會檢查您的 AWS 環境，然後在有機會節省成本、改善系統可用性和效能，或協助解決安全漏洞時提出建議。所有 AWS 客戶都可以存取五個 Trusted Advisor 檢查。具有 AWS 支援 商業或企業計劃的客戶可以

檢視所有 Trusted Advisor 檢查。如需詳細資訊，請參閱 AWS Trusted Advisor 的 AWS 支援 使用者指南 與 AWS Health 使用者指南。

詳細資訊

- 在 中記錄和監控 AWS Systems Manager

AWS Systems Manager 的合規驗證

本主題討論 AWS Systems Manager 合規與第三方保證計劃。如需檢視受管節點的合規資料相關資訊，請參閱 AWS Systems Manager合規。

在多個 AWS 合規計劃中，第三方稽核人員會評估 Systems Manager 的安全與合規。這些計劃包括 SOC、PCI、FedRAMP、HIPAA 等等。

如需特定合規計劃 AWS 服務 範圍內的 清單，請參閱AWS 合規計劃範圍內合規的服務。如需一般資訊，請參閱AWS 合規計劃。

您可以使用 下載第三方稽核報告 AWS Artifact。如需詳細資訊，請參閱在 中下載報告 AWS Artifact。

使用 時的合規責任Systems Manager取決於資料的敏感度、您公司的合規目標，以及適用的法律和法規。 AWS 提供下列資源以協助合規：

- 安全與合規快速入門指南：這些部署指南討論架構考量，並提供在 AWS上部署以安全及合規為重心之基準環境的步驟。
- HIPAA 安全與合規架構白皮書 – 此白皮書說明公司如何使用 AWS 來建立符合 HIPAA 規範的應用程式。
- AWS 合規資源 – 此工作手冊和指南的集合可能適用於您的產業和位置。
- AWS Config 開發人員指南中的使用規則評估資源 – AWS Config 服務會評估資源組態符合內部實務、產業準則和法規的程度。
- AWS Security Hub – 這 AWS 服務 可讓您全面檢視 的安全狀態 AWS ，協助您檢查是否符合安全產業標準和最佳實務。

AWS Systems Manager 中的恢復能力

AWS 全球基礎設施是以 AWS 區域 和 可用區域為基礎建置。 AWS 區域 提供多個實體隔離和隔離的可用區域，這些區域與低延遲、高輸送量和高度備援聯網連接。透過可用區域，您可以設計與操作的應用程式和資料庫，在可用區域之間自動容錯移轉而不會發生中斷。可用區域的可用性、容錯能力和擴展能力，均較單一或多個資料中心的傳統基礎設施還高。

如需 AWS 區域 和 可用區域的詳細資訊，請參閱AWS 全球基礎設施。

AWS Systems Manager 中的基礎設施安全

作為受管服務， AWS Systems Manager 受到 AWS 全球網路安全的保護。如需 AWS 安全服務以及如何 AWS 保護基礎設施的相關資訊，請參閱AWS 雲端安全。若要使用基礎設施安全的最佳實務設計您的 AWS 環境，請參閱 Security Pillar AWS Well-Architected Framework 中的基礎設施保護。

您可以使用 AWS 發佈的 API 呼叫，Systems Manager透過網路存取 。使用者端必須支援下列專案：

- Transport Layer Security (TLS)。我們需要 TLS 1.2 並建議使用 TLS 1.3。
- 具備完美轉送私密(PFS)的密碼套件，例如 DHE (Ephemeral Diffie-Hellman)或 ECDHE (Elliptic Curve Ephemeral Diffie-Hellman)。現代系統(如 Java 7 和更新版本)大多會支援這些模式。

此外，請求必須使用存取金鑰 ID 和與 IAM 主體相關聯的私密存取金鑰來簽署。或者，您可以使用 AWS Security Token Service (AWS STS) 以產生暫時安全憑證以簽署請求。

AWS Systems Manager 中的組態與漏洞分析

AWS 處理基本安全任務，例如防火牆組態和災難復原。這些程序已由適當的第三方進行檢閱並認證。如需詳細資訊，請參閱以下 資源：

- AWS Systems Manager 的合規驗證
- 共同的責任模型
- 安全性、身分與合規的最佳實務

Systems Manager 的安全最佳實務

AWS Systems Manager 提供許多安全功能，供您在開發和實作自己的安全政策時考慮。以下最佳實務為一般準則，並不代表完整的安全解決方案。這些最佳實務可能不適用或無法滿足您的環境需求，因此請將其視為實用建議就好，而不要當作是指示。

主題

- Systems Manager 預防性安全最佳實務
- Systems Manager 監控和稽核最佳實務

Systems Manager 預防性安全最佳實務

以下 Systems Manager 最佳實務有助於預防安全事件的發生。

實作最低權限存取

當您授予許可時，需要決定哪些使用者會取得哪些 Systems Manager 資源的許可。您允許針對這些資源啟用允許執行的動作。因此，您只應授與執行任務所需的許可。對降低錯誤或惡意意圖所引起的安全風險和影響而言，實作最低權限存取是相當重要的一環。

下列工具可用來實作最低權限存取：

- IAM 政策和 IAM 實體的許可界限
- 服務控制政策

將 SSM Agent 設定為使用代理時，使用建議的設定

如果將 SSM Agent 設定為使用代理，請將 no_proxy 變數與 Systems Manager 執行個體中繼資料服務的 IP 位址搭配使用，確保對 Systems Manager 的呼叫不會採用代理服務的身分。

如需詳細資訊，請參閱 將 SSM Agent 設定為使用 Linux 節點上的代理 和 將 SSM Agent 設定為使用 Windows Server 執行個體的代理。

使用 SecureString 參數來加密和保護機密資料

在 中Parameter Store， 中的工具 AWS Systems Manager參數SecureString是任何需要以安全方式存放和參考的敏感資料。如果您有不希望使用者以純文字變更或參考的資料，例如密碼或授權金鑰，請使用SecureString資料類型建立這些參數。 Parameter Store 會使用 AWS KMS key in AWS Key Management Service (AWS KMS) 來加密參數值。 AWS KMS 會在加密參數值 AWS 受管金鑰 時使用客戶受管金鑰或 。為了獲得最大的安全，我們建議您使用自己的 KMS 金鑰。如果您使用 AWS 受管金鑰，擁有在帳戶中執行 GetParameter和 GetParameters動作許可的任何使用者都可以檢視或擷取所有SecureString參數的內容。如果您使用客戶受管金鑰來加密您的安全 SecureString 值，您可以使用 IAM 政策和金鑰政策來管理加密和解密參數的許可。

若使用了 AWS 受管金鑰，則很難為這些操作建立存取控制政策。例如，如果您使用 AWS 受管金鑰 來加密SecureString參數，且不希望使用者使用SecureString參數，使用者的 IAM 政策必須明確拒絕存取預設金鑰。

如需詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的 使用 IAM 政策限制對 Parameter Store 參數的存取 和 AWS Systems ManagerParameter Store 如何使用 AWS KMS。

定義文件參數的 allowedValues 和 allowedPattern

您可以定義 `allowedValues` 和 `allowedPattern` 來驗證 Systems Manager 文件 (SSM 文件) 參數的使用者輸入。對於 `allowedValues`，您可以定義參數允許的值陣列。如果使用者輸入不允許的值，則無法開始執行。對於 `allowedPattern`，您可以定義規則運算式，以驗證使用者輸入是否符合參數定義的模式。如果使用者輸入不符合允許的模式，則無法開始執行。

如需 `allowedValues` 和 `allowedPattern` 的更多相關資訊，請參閱資料元素和參數。

封鎖文件的公有共用

除非您的使用案例需要允許公有共用，否則建議您在 Systems Manager 文件主控台的 Preferences (偏好設定) 區段中開啟 SSM 文件的封鎖公有共用設定。

使用 Amazon Virtual Private Cloud (Amazon VPC) 和 VPC 端點

您可以使用 Amazon VPC 將 AWS 資源啟動至您定義的虛擬網路。這個虛擬網路與您在資料中心中操作的傳統網路非常相似，且具備使用 AWS可擴展基礎設施的優勢。

透過實作 VPC 端點，您可以私下將 VPC 連線至支援的 AWS 服務 和 VPC 端點服務， AWS PrivateLink 而不需要網際網路閘道、NAT 裝置、VPN 連線或 AWS Direct Connect 連線。VPC 中的執行個體不需要公有 IP 地址，即可與服務中的資源通訊。VPC 與另一個服務之間的流量都會保持在 Amazon 網路的範圍內。

如需有關 Amazon VPC 安全的詳細資訊，請參閱《Amazon VPC 使用者指南》中的使用適用於 Systems Manager 的 VPC 端點來改善 EC2 執行個體的安全性和 Amazon VPC 的網際網路流量隱私權。

限制 Session Manager 使用者只能操作使用互動式命令和特定 SSM 工作階段文件的工作階段

Session Manager是 中的工具AWS Systems Manager，提供數種方法，可啟動受管節點的工作階段。為了獲得最安全的連線，您可以要求使用者使用互動式命令方法來連線，以限制使用者與特定命令或命令序列的互動。這可以協助您管理使用者可採取的互動動作。如需詳細資訊，請參閱啟動工作階段 (互動和非互動式命令)。

為了增加安全性，您可以限制 Session Manager 只能存取特定的 Amazon EC2 執行個體和 Session Manager 工作階段文件。您可以使用 AWS Identity and Access Management (IAM) 政策以這種方式授予或撤銷Session Manager存取權。如需詳細資訊，請參閱步驟 3：控制工作階段對受管節點的存取權。

提供 Automation 工作流程的臨時節點許可

在 Automation 中的工作流程期間，這是 中的工具AWS Systems Manager，您的節點可能需要僅該執行所需的許可，但不需要其他Systems Manager操作所需的許可。例如，自動化工作流程可能需

要節點來呼叫特定 API 操作，或在工作流程期間特別存取 AWS 資源。如果這些呼叫或資源是您想要限制存取的呼叫或資源，您可以在 Automation Runbook 本身內為節點提供暫時的補充許可，而不用將許可新增至 IAM 執行個體設定檔。在 Automation 工作流程結束時，暫時許可會移除。如需詳細資訊，請參閱 AWS 管理和管控部落格中的提供 AWS Systems Manager Automation 的臨時執行個體許可。

讓 AWS 和 Systems Manager 工具保持最新狀態

AWS 會定期發行工具和外掛程式的更新版本，供您在 AWS 和 Systems Manager操作中使用。將這些資源保持在最新狀態，可確保帳戶中的使用者和節點可以存取這些工具的最新功能和安全功能。

- SSM Agent – AWS Systems Manager 代理程式 (SSM Agent) 是 Amazon 軟體，可在 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、內部部署伺服器或虛擬機器 (VM) 上安裝和設定。SSM Agent 可讓 Systems Manager 更新、管理和設定這些資源。我們建議您至少每兩週檢查一次新版本或自動更新代理程式。如需相關資訊，請參閱 自動化 SSM Agent 更新。我們也建議您在更新程序期間驗證 SSM Agent 的簽章。如需相關資訊，請參閱 驗證 SSM Agent 的簽章。
- AWS CLI – The AWS Command Line Interface (AWS CLI) 是一種開放原始碼工具，可讓您 AWS 服務 使用命令列 shell 中的命令與 互動。若要更新 AWS CLI，請執行用來安裝 的相同命令 AWS CLI。建議您在本機電腦上建立排程任務，至少每兩週執行一次適合您作業系統的命令。如需安裝命令的相關資訊，請參閱AWS Command Line Interface 《 使用者指南》中的安裝 AWS CLI 版本 2。
- AWS Tools for Windows PowerShell – Tools for Windows PowerShell 是一組 PowerShell 模組，建置在 AWS SDK for .NET 公開的功能上。 AWS Tools for Windows PowerShell 可讓您從 PowerShell 命令列在您的 AWS 資源上編寫操作指令碼。當 Tools for Windows PowerShell 的版本更新發佈時，您應定期更新於本機執行的版本。如需詳細資訊，請參閱《IAM 政策模擬器使用者指南》中的AWS Tools for Windows PowerShell 在 Windows 上更新 或在 Linux AWS Tools for Windows PowerShell 上更新 macOS 。
- Session Manager 外掛程式 – 如果您組織中具有使用 Session Manager 許可的使用者想要使用 AWS CLI來連線至節點，他們必須先在其本機電腦上安裝 Session Manager 外掛程式。若要更新外掛程式，請執行與用於安裝外掛程式相同的命令。建議您在本機電腦上建立排程任務，至少每兩週執行一次適合您作業系統的命令。如需相關資訊，請參閱 安裝 的Session Manager外掛程式 AWS CLI。
- CloudWatch 代理程式 – 您可以設定並使用 CloudWatch 代理程式，以從 EC2 執行個體、內部部署執行個體和虛擬機器 (VM) 收集指標和日誌。這些日誌可以傳送到 Amazon CloudWatch Logs 供監控和分析。我們建議您至少每兩週檢查一次新版本或自動更新代理程式。針對最簡單的更新，請使用 AWS Systems Manager 快速設定。如需相關資訊，請參閱 AWS Systems Manager Quick Setup。

Systems Manager 監控和稽核最佳實務

以下 Systems Manager 最佳實務有助於偵測潛在安全弱點與事件。

識別並稽核所有 Systems Manager 資源

識別 IT 資產是控管和保障安全的重要環節。您需要識別所有 Systems Manager 資源，才能評估其安全狀態並對潛在弱點採取行動。

您可使用標籤編輯器來識別重視安全或重視稽核的資源，接著在需要搜尋上述資源時運用這些標籤。如需詳細資訊，請參閱《AWS Resource Groups 使用者指南》中的尋找要加上標籤的資源。

請為 Systems Manager 資源建立資源群組。如需詳細資訊，請參閱什麼是 Resource Groups？

使用 Amazon CloudWatch 監控工具來實作監控

監控是維護 Systems Manager 及您 AWS 解決方案安全性、可靠性、可用性和效能的重要部分。Amazon CloudWatch 提供數種工具和服務，協助您監控 Systems Manager 和其他 AWS 服務。如需詳細資訊，請參閱 將節點日誌傳送至統一 CloudWatch Logs (CloudWatch 代理程式) 和使用 Amazon EventBridge 監控 Systems Manager。

使用 CloudTrail

AWS CloudTrail 提供 AWS 服務 中使用者、角色或 所採取動作的記錄Systems Manager。您可以利用 CloudTrail 所收集的資訊來判斷向 Systems Manager 發出的請求，以及發出請求的 IP 地址、人員、時間和其他詳細資訊。如需詳細資訊，請參閱使用 記錄 AWS Systems Manager API 呼叫 AWS CloudTrail。

開啟 AWS Config

AWS Config 可讓您評估、稽核和評估 AWS resources. AWS Config monitors 資源組態的組態，讓您根據所需的安全組態評估記錄的組態。使用 AWS Config，您可以檢閱 AWS 資源之間組態和關係的變更、調查詳細的資源組態歷史記錄，並根據內部準則中指定的組態來判斷整體合規性。如此一來，您就能輕鬆進行合規稽核、安全分析、變更管理和操作故障診斷的程序。如需詳細資訊，請參閱《AWS Config 開發人員指南》中的使用主控台設定 AWS Config。當您指定要記錄的資源類型時，請確定其中包含 Systems Manager 資源。

監控 AWS 安全建議

您應該定期檢查張貼在 Trusted Advisor 中的安全建議 AWS 帳戶。您也可以使用 describe-trusted-advisor-checks，以程式設計方式來執行此操作。

此外，請主動監控註冊到每個 的主要電子郵件地址 AWS 帳戶。 AWS 將使用此電子郵件地址與您聯絡，以解決可能會影響您的安全問題。

AWS 具有廣泛影響的操作問題會張貼在 AWS Service Health Dashboard 上。系統也會透過 Personal Health Dashboard，將操作問題張貼至個別帳戶。如需詳細資訊，請參閱 AWS Health 文件。

詳細資訊

- 安全性、身分與合規的最佳實務
- 入門：在設定 AWS 資源時遵循安全最佳實務 (AWS 安全部落格)
- IAM 中的安全最佳實務
- 中的安全最佳實務 AWS CloudTrail
- Amazon Simple Storage Service (Amazon S3) 的安全最佳實務
- 的安全最佳實務 AWS Key Management Service

Systems Manager AWS SDKs的程式碼範例

下列程式碼範例示範如何使用 Systems Manager 搭配 AWS 軟體開發套件 (SDK)。

基本概念是程式碼範例，這些範例說明如何在服務內執行基本操作。

Actions 是大型程式的程式碼摘錄，必須在內容中執行。雖然動作會告訴您如何呼叫個別服務函數，但您可以在其相關情境中查看內容中的動作。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含入門相關資訊和舊版 SDK 的詳細資訊。

開始使用

嗨，Systems Manager

下列程式碼範例示範如何開始使用 Systems Manager。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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);
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 ListDocuments。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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();
}
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 ListDocuments。

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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"])
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 ListDocuments。

程式碼範例

- Systems Manager AWS SDKs的基本範例
  - 嗨，Systems Manager
  - 使用 AWS SDK 了解 Systems Manager 的基本概念
  - Systems Manager 使用 AWS SDKs的動作
    - 搭配使用 AddTagsToResource 與 CLI
    - 搭配使用 CancelCommand 與 CLI
    - 搭配使用 CreateActivation 與 CLI
    - 搭配使用 CreateAssociation 與 CLI
    - 搭配使用 CreateAssociationBatch 與 CLI
    - CreateDocument 搭配 AWS SDK 或 CLI 使用
    - CreateMaintenanceWindow 搭配 AWS SDK 或 CLI 使用
    - CreateOpsItem 搭配 AWS SDK 或 CLI 使用
    - 搭配使用 CreatePatchBaseline 與 CLI
    - 搭配使用 DeleteActivation 與 CLI
    - 搭配使用 DeleteAssociation 與 CLI
    - DeleteDocument 搭配 AWS SDK 或 CLI 使用
    - DeleteMaintenanceWindow 搭配 AWS SDK 或 CLI 使用
    - DeleteOpsItem 搭配 AWS SDK 使用

- 搭配使用 DeleteParameter 與 CLI
- 搭配使用 DeletePatchBaseline 與 CLI
- 搭配使用 DeregisterManagedInstance 與 CLI
- 搭配使用 DeregisterPatchBaselineForPatchGroup 與 CLI
- 搭配使用 DeregisterTargetFromMaintenanceWindow 與 CLI
- 搭配使用 DeregisterTaskFromMaintenanceWindow 與 CLI
- 搭配使用 DescribeActivations 與 CLI
- 搭配使用 DescribeAssociation 與 CLI
- 搭配使用 DescribeAssociationExecutionTargets 與 CLI
- 搭配使用 DescribeAssociationExecutions 與 CLI
- 搭配使用 DescribeAutomationExecutions 與 CLI
- 搭配使用 DescribeAutomationStepExecutions 與 CLI
- 搭配使用 DescribeAvailablePatches 與 CLI
- 搭配使用 DescribeDocument 與 CLI
- 搭配使用 DescribeDocumentPermission 與 CLI
- 搭配使用 DescribeEffectiveInstanceAssociations 與 CLI
- 搭配使用 DescribeEffectivePatchesForPatchBaseline 與 CLI
- 搭配使用 DescribeInstanceAssociationsStatus 與 CLI
- 搭配使用 DescribeInstanceInformation 與 CLI
- 搭配使用 DescribeInstancePatchStates 與 CLI
- 搭配使用 DescribeInstancePatchStatesForPatchGroup 與 CLI
- 搭配使用 DescribeInstancePatches 與 CLI
- 搭配使用 DescribeMaintenanceWindowExecutionTaskInvocations 與 CLI
- 搭配使用 DescribeMaintenanceWindowExecutionTasks 與 CLI
- 搭配使用 DescribeMaintenanceWindowExecutions 與 CLI
- 搭配使用 DescribeMaintenanceWindowTargets 與 CLI
- 搭配使用 DescribeMaintenanceWindowTasks 與 CLI
- 搭配使用 DescribeMaintenanceWindows 與 CLI
- DescribeOpsItems 搭配 AWS SDK 或 CLI 使用
- DescribeParameters 搭配 AWS SDK 或 CLI 使用

- 搭配使用 DescribePatchBaselines 與 CLI
- 搭配使用 DescribePatchGroupState 與 CLI
- 搭配使用 DescribePatchGroups 與 CLI
- 搭配使用 GetAutomationExecution 與 CLI
- 搭配使用 GetCommandInvocation 與 CLI
- 搭配使用 GetConnectionStatus 與 CLI
- 搭配使用 GetDefaultPatchBaseline 與 CLI
- 搭配使用 GetDeployablePatchSnapshotForInstance 與 CLI
- 搭配使用 GetDocument 與 CLI
- 搭配使用 GetInventory 與 CLI
- 搭配使用 GetInventorySchema 與 CLI
- 搭配使用 GetMaintenanceWindow 與 CLI
- 搭配使用 GetMaintenanceWindowExecution 與 CLI
- 搭配使用 GetMaintenanceWindowExecutionTask 與 CLI
- GetParameter 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetParameterHistory 與 CLI
- 搭配使用 GetParameters 與 CLI
- 搭配使用 GetPatchBaseline 與 CLI
- 搭配使用 GetPatchBaselineForPatchGroup 與 CLI
- 搭配使用 ListAssociationVersions 與 CLI
- 搭配使用 ListAssociations 與 CLI
- ListCommandInvocations 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListCommands 與 CLI
- 搭配使用 ListComplianceItems 與 CLI
- 搭配使用 ListComplianceSummaries 與 CLI
- 搭配使用 ListDocumentVersions 與 CLI
- 搭配使用 ListDocuments 與 CLI
- 搭配使用 ListInventoryEntries 與 CLI
- 搭配使用 ListResourceComplianceSummaries 與 CLI
- 搭配使用 ListTagsForResource 與 CLI

- 搭配使用 ModifyDocumentPermission 與 CLI
- 搭配使用 PutComplianceItems 與 CLI
- 搭配使用 PutInventory 與 CLI
- PutParameter 搭配 AWS SDK 或 CLI 使用
- 搭配使用 RegisterDefaultPatchBaseline 與 CLI
- 搭配使用 RegisterPatchBaselineForPatchGroup 與 CLI
- 搭配使用 RegisterTargetWithMaintenanceWindow 與 CLI
- 搭配使用 RegisterTaskWithMaintenanceWindow 與 CLI
- 搭配使用 RemoveTagsFromResource 與 CLI
- SendCommand 搭配 AWS SDK 或 CLI 使用
- 搭配使用 StartAutomationExecution 與 CLI
- 搭配使用 StartSession 與 CLI
- 搭配使用 StopAutomationExecution 與 CLI
- 搭配使用 UpdateAssociation 與 CLI
- 搭配使用 UpdateAssociationStatus 與 CLI
- 搭配使用 UpdateDocument 與 CLI
- 搭配使用 UpdateDocumentDefaultVersion 與 CLI
- UpdateMaintenanceWindow 搭配 AWS SDK 或 CLI 使用
- 搭配使用 UpdateManagedInstanceRole 與 CLI
- UpdateOpsItem 搭配 AWS SDK 或 CLI 使用
- 搭配使用 UpdatePatchBaseline 與 CLI

Systems Manager AWS SDKs的基本範例

下列程式碼範例示範如何 AWS Systems Manager 搭配 AWS SDKs 使用 的基本概念。

範例

- 嗨，Systems Manager
- 使用 AWS SDK 了解 Systems Manager 的基本概念
- Systems Manager 使用 AWS SDKs的動作
  - 搭配使用 AddTagsToResource 與 CLI

- 搭配使用 CancelCommand 與 CLI
- 搭配使用 CreateActivation 與 CLI
- 搭配使用 CreateAssociation 與 CLI
- 搭配使用 CreateAssociationBatch 與 CLI
- CreateDocument 搭配 AWS SDK 或 CLI 使用
- CreateMaintenanceWindow 搭配 AWS SDK 或 CLI 使用
- CreateOpsItem 搭配 AWS SDK 或 CLI 使用
- 搭配使用 CreatePatchBaseline 與 CLI
- 搭配使用 DeleteActivation 與 CLI
- 搭配使用 DeleteAssociation 與 CLI
- DeleteDocument 搭配 AWS SDK 或 CLI 使用
- DeleteMaintenanceWindow 搭配 AWS SDK 或 CLI 使用
- DeleteOpsItem 搭配 AWS SDK 使用
- 搭配使用 DeleteParameter 與 CLI
- 搭配使用 DeletePatchBaseline 與 CLI
- 搭配使用 DeregisterManagedInstance 與 CLI
- 搭配使用 DeregisterPatchBaselineForPatchGroup 與 CLI
- 搭配使用 DeregisterTargetFromMaintenanceWindow 與 CLI
- 搭配使用 DeregisterTaskFromMaintenanceWindow 與 CLI
- 搭配使用 DescribeActivations 與 CLI
- 搭配使用 DescribeAssociation 與 CLI
- 搭配使用 DescribeAssociationExecutionTargets 與 CLI
- 搭配使用 DescribeAssociationExecutions 與 CLI
- 搭配使用 DescribeAutomationExecutions 與 CLI
- 搭配使用 DescribeAutomationStepExecutions 與 CLI
- 搭配使用 DescribeAvailablePatches 與 CLI
- 搭配使用 DescribeDocument 與 CLI
- 搭配使用 DescribeDocumentPermission 與 CLI
- 搭配使用 DescribeEffectiveInstanceAssociations 與 CLI
- 搭配使用 DescribeEffectivePatchesForPatchBaseline 與 CLI

- 搭配使用 DescribeInstanceAssociationsStatus 與 CLI
- 搭配使用 DescribeInstanceInformation 與 CLI
- 搭配使用 DescribeInstancePatchStates 與 CLI
- 搭配使用 DescribeInstancePatchStatesForPatchGroup 與 CLI
- 搭配使用 DescribeInstancePatches 與 CLI
- 搭配使用 DescribeMaintenanceWindowExecutionTaskInvocations 與 CLI
- 搭配使用 DescribeMaintenanceWindowExecutionTasks 與 CLI
- 搭配使用 DescribeMaintenanceWindowExecutions 與 CLI
- 搭配使用 DescribeMaintenanceWindowTargets 與 CLI
- 搭配使用 DescribeMaintenanceWindowTasks 與 CLI
- 搭配使用 DescribeMaintenanceWindows 與 CLI
- DescribeOpsItems 搭配 AWS SDK 或 CLI 使用
- DescribeParameters 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DescribePatchBaselines 與 CLI
- 搭配使用 DescribePatchGroupState 與 CLI
- 搭配使用 DescribePatchGroups 與 CLI
- 搭配使用 GetAutomationExecution 與 CLI
- 搭配使用 GetCommandInvocation 與 CLI
- 搭配使用 GetConnectionStatus 與 CLI
- 搭配使用 GetDefaultPatchBaseline 與 CLI
- 搭配使用 GetDeployablePatchSnapshotForInstance 與 CLI
- 搭配使用 GetDocument 與 CLI
- 搭配使用 GetInventory 與 CLI
- 搭配使用 GetInventorySchema 與 CLI
- 搭配使用 GetMaintenanceWindow 與 CLI
- 搭配使用 GetMaintenanceWindowExecution 與 CLI
- 搭配使用 GetMaintenanceWindowExecutionTask 與 CLI
- GetParameter 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetParameterHistory 與 CLI
- 搭配使用 GetParameters 與 CLI

- 搭配使用 GetPatchBaseline 與 CLI
- 搭配使用 GetPatchBaselineForPatchGroup 與 CLI
- 搭配使用 ListAssociationVersions 與 CLI
- 搭配使用 ListAssociations 與 CLI
- ListCommandInvocations 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListCommands 與 CLI
- 搭配使用 ListComplianceItems 與 CLI
- 搭配使用 ListComplianceSummaries 與 CLI
- 搭配使用 ListDocumentVersions 與 CLI
- 搭配使用 ListDocuments 與 CLI
- 搭配使用 ListInventoryEntries 與 CLI
- 搭配使用 ListResourceComplianceSummaries 與 CLI
- 搭配使用 ListTagsForResource 與 CLI
- 搭配使用 ModifyDocumentPermission 與 CLI
- 搭配使用 PutComplianceItems 與 CLI
- 搭配使用 PutInventory 與 CLI
- PutParameter 搭配 AWS SDK 或 CLI 使用
- 搭配使用 RegisterDefaultPatchBaseline 與 CLI
- 搭配使用 RegisterPatchBaselineForPatchGroup 與 CLI
- 搭配使用 RegisterTargetWithMaintenanceWindow 與 CLI
- 搭配使用 RegisterTaskWithMaintenanceWindow 與 CLI
- 搭配使用 RemoveTagsFromResource 與 CLI
- SendCommand 搭配 AWS SDK 或 CLI 使用
- 搭配使用 StartAutomationExecution 與 CLI
- 搭配使用 StartSession 與 CLI
- 搭配使用 StopAutomationExecution 與 CLI
- 搭配使用 UpdateAssociation 與 CLI
- 搭配使用 UpdateAssociationStatus 與 CLI
- 搭配使用 UpdateDocument 與 CLI
- 搭配使用 UpdateDocumentDefaultVersion 與 CLI

- UpdateMaintenanceWindow 搭配 AWS SDK 或 CLI 使用
- 搭配使用 UpdateManagedInstanceRole 與 CLI
- UpdateOpsItem 搭配 AWS SDK 或 CLI 使用
- 搭配使用 UpdatePatchBaseline 與 CLI

嗨，Systems Manager

下列程式碼範例示範如何開始使用 Systems Manager。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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);
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 ListDocuments。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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();
}
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 ListDocuments。

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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"])
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 ListDocuments。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS SDK 了解 Systems Manager 的基本概念

下列程式碼範例示範如何：

- 建立維護時段。
- 修改維護時段排程。
- 建立文件。
- 將命令傳送至指定的 EC2 執行個體。
- 建立 OpsItem。
- 更新並解析 OpsItem。
- 刪除維護時段、OpsItem 和文件。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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 方法的包裝函式類別。

```
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];
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的下列主題。
  - CreateDocument
  - CreateMaintenanceWindow
  - CreateOpsItem
  - DeleteMaintenanceWindow

- ListCommandInvocations
- SendCommand
- UpdateOpsItem

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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 });
```

```
}
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的下列主題。
  - CreateDocument
  - CreateMaintenanceWindow
  - CreateOpsItem
  - DeleteMaintenanceWindow
  - ListCommandInvocations
  - SendCommand
  - UpdateOpsItem

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

在命令提示中執行互動式案例。

```
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.")
```

定義包裝文件和命令動作的類別。

```
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
```

定義包裝 opsitem 動作的類別。

```
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
```

定義包裝維護時段動作的類別。

```
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
```

- 如需 API 的詳細資訊，請參閱 AWS SDK for Python (Boto3) API Reference 中的下列主題。
  - CreateDocument
  - CreateMaintenanceWindow
  - CreateOpsItem
  - DeleteMaintenanceWindow
  - ListCommandInvocations
  - SendCommand
  - UpdateOpsItem

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

Systems Manager 使用 AWS SDKs的動作

下列程式碼範例示範如何使用 AWS SDKs 執行個別 Systems Manager 動作。每個範例均包含 GitHub 的連結，您可以在連結中找到設定和執行程式碼的相關說明。

下列範例僅包含最常使用的動作。如需完整清單，請參閱《AWS Systems Manager API 參考》。

範例

- 搭配使用 AddTagsToResource 與 CLI

- 搭配使用 CancelCommand 與 CLI
- 搭配使用 CreateActivation 與 CLI
- 搭配使用 CreateAssociation 與 CLI
- 搭配使用 CreateAssociationBatch 與 CLI
- CreateDocument 搭配 AWS SDK 或 CLI 使用
- CreateMaintenanceWindow 搭配 AWS SDK 或 CLI 使用
- CreateOpsItem 搭配 AWS SDK 或 CLI 使用
- 搭配使用 CreatePatchBaseline 與 CLI
- 搭配使用 DeleteActivation 與 CLI
- 搭配使用 DeleteAssociation 與 CLI
- DeleteDocument 搭配 AWS SDK 或 CLI 使用
- DeleteMaintenanceWindow 搭配 AWS SDK 或 CLI 使用
- DeleteOpsItem 搭配 AWS SDK 使用
- 搭配使用 DeleteParameter 與 CLI
- 搭配使用 DeletePatchBaseline 與 CLI
- 搭配使用 DeregisterManagedInstance 與 CLI
- 搭配使用 DeregisterPatchBaselineForPatchGroup 與 CLI
- 搭配使用 DeregisterTargetFromMaintenanceWindow 與 CLI
- 搭配使用 DeregisterTaskFromMaintenanceWindow 與 CLI
- 搭配使用 DescribeActivations 與 CLI
- 搭配使用 DescribeAssociation 與 CLI
- 搭配使用 DescribeAssociationExecutionTargets 與 CLI
- 搭配使用 DescribeAssociationExecutions 與 CLI
- 搭配使用 DescribeAutomationExecutions 與 CLI
- 搭配使用 DescribeAutomationStepExecutions 與 CLI
- 搭配使用 DescribeAvailablePatches 與 CLI
- 搭配使用 DescribeDocument 與 CLI
- 搭配使用 DescribeDocumentPermission 與 CLI
- 搭配使用 DescribeEffectiveInstanceAssociations 與 CLI
- 搭配使用 DescribeEffectivePatchesForPatchBaseline 與 CLI

- 搭配使用 DescribeInstanceAssociationsStatus 與 CLI
- 搭配使用 DescribeInstanceInformation 與 CLI
- 搭配使用 DescribeInstancePatchStates 與 CLI
- 搭配使用 DescribeInstancePatchStatesForPatchGroup 與 CLI
- 搭配使用 DescribeInstancePatches 與 CLI
- 搭配使用 DescribeMaintenanceWindowExecutionTaskInvocations 與 CLI
- 搭配使用 DescribeMaintenanceWindowExecutionTasks 與 CLI
- 搭配使用 DescribeMaintenanceWindowExecutions 與 CLI
- 搭配使用 DescribeMaintenanceWindowTargets 與 CLI
- 搭配使用 DescribeMaintenanceWindowTasks 與 CLI
- 搭配使用 DescribeMaintenanceWindows 與 CLI
- DescribeOpsItems 搭配 AWS SDK 或 CLI 使用
- DescribeParameters 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DescribePatchBaselines 與 CLI
- 搭配使用 DescribePatchGroupState 與 CLI
- 搭配使用 DescribePatchGroups 與 CLI
- 搭配使用 GetAutomationExecution 與 CLI
- 搭配使用 GetCommandInvocation 與 CLI
- 搭配使用 GetConnectionStatus 與 CLI
- 搭配使用 GetDefaultPatchBaseline 與 CLI
- 搭配使用 GetDeployablePatchSnapshotForInstance 與 CLI
- 搭配使用 GetDocument 與 CLI
- 搭配使用 GetInventory 與 CLI
- 搭配使用 GetInventorySchema 與 CLI
- 搭配使用 GetMaintenanceWindow 與 CLI
- 搭配使用 GetMaintenanceWindowExecution 與 CLI
- 搭配使用 GetMaintenanceWindowExecutionTask 與 CLI
- GetParameter 搭配 AWS SDK 或 CLI 使用
- 搭配使用 GetParameterHistory 與 CLI
- 搭配使用 GetParameters 與 CLI

- 搭配使用 GetPatchBaseline 與 CLI
- 搭配使用 GetPatchBaselineForPatchGroup 與 CLI
- 搭配使用 ListAssociationVersions 與 CLI
- 搭配使用 ListAssociations 與 CLI
- ListCommandInvocations 搭配 AWS SDK 或 CLI 使用
- 搭配使用 ListCommands 與 CLI
- 搭配使用 ListComplianceItems 與 CLI
- 搭配使用 ListComplianceSummaries 與 CLI
- 搭配使用 ListDocumentVersions 與 CLI
- 搭配使用 ListDocuments 與 CLI
- 搭配使用 ListInventoryEntries 與 CLI
- 搭配使用 ListResourceComplianceSummaries 與 CLI
- 搭配使用 ListTagsForResource 與 CLI
- 搭配使用 ModifyDocumentPermission 與 CLI
- 搭配使用 PutComplianceItems 與 CLI
- 搭配使用 PutInventory 與 CLI
- PutParameter 搭配 AWS SDK 或 CLI 使用
- 搭配使用 RegisterDefaultPatchBaseline 與 CLI
- 搭配使用 RegisterPatchBaselineForPatchGroup 與 CLI
- 搭配使用 RegisterTargetWithMaintenanceWindow 與 CLI
- 搭配使用 RegisterTaskWithMaintenanceWindow 與 CLI
- 搭配使用 RemoveTagsFromResource 與 CLI
- SendCommand 搭配 AWS SDK 或 CLI 使用
- 搭配使用 StartAutomationExecution 與 CLI
- 搭配使用 StartSession 與 CLI
- 搭配使用 StopAutomationExecution 與 CLI
- 搭配使用 UpdateAssociation 與 CLI
- 搭配使用 UpdateAssociationStatus 與 CLI
- 搭配使用 UpdateDocument 與 CLI
- 搭配使用 UpdateDocumentDefaultVersion 與 CLI

- UpdateMaintenanceWindow 搭配 AWS SDK 或 CLI 使用
- 搭配使用 UpdateManagedInstanceRole 與 CLI
- UpdateOpsItem 搭配 AWS SDK 或 CLI 使用
- 搭配使用 UpdatePatchBaseline 與 CLI

搭配使用 **AddTagsToResource** 與 CLI

下列程式碼範例示範如何使用 AddTagsToResource。

CLI

AWS CLI

範例 1：將標籤新增至維護時段

下列 add-tags-to-resource 範例示範將標籤新增至指定的維護時段。

```
aws ssm add-tags-to-resource \
    --resource-type "MaintenanceWindow" \
    --resource-id "mw-03eb9db428EXAMPLE" \
    --tags "Key=Stack,Value=Production"
```

此命令不會產生輸出。

範例 2：將標籤新增至參數

下列 add-tags-to-resource 範例示範將兩個標籤新增至指定的參數。

```
aws ssm add-tags-to-resource \
    --resource-type "Parameter" \
    --resource-id "My-Parameter" \
    --tags '[{"Key":"Region","Value":"East"},{"Key":"Environment",
 "Value":"Production"}]'
```

此命令不會產生輸出。

範例 3：將標籤新增至 SSM 文件

下列 add-tags-to-resource 範例示範將標籤新增至指定的文件。

```
aws ssm add-tags-to-resource \
```

```
    --resource-type "Document" \
    --resource-id "My-Document" \
    --tags "Key=Quarter,Value=Q322"
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Tagging Systems Manager resources。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 AddTagsToResource。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範使用新標籤更新維護時段。如果命令成功，則無輸出訊息。此範例使用的語法需要 PowerShell 版本 3 或更新版本。

```
$option1 = @{Key="Stack";Value=@("Production")}
Add-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
 "MaintenanceWindow" -Tag $option1
```

範例 2：使用 PowerShell 版本 2 時，必須使用 New-Object 來建立每個標籤。如果命令成功，則無輸出訊息。

```
$tag1 = New-Object Amazon.SimpleSystemsManagement.Model.Tag
$tag1.Key = "Stack"
$tag1.Value = "Production"

Add-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
 "MaintenanceWindow" -Tag $tag1
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 AddTagsToResource。

PowerShell V5 的工具

範例 1：此範例示範使用新標籤更新維護時段。如果命令成功，則無輸出訊息。此範例使用的語法需要 PowerShell 版本 3 或更新版本。

```
$option1 = @{Key="Stack";Value=@("Production")}
```

```
Add-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
 "MaintenanceWindow" -Tag $option1
```

範例 2：使用 PowerShell 版本 2 時，必須使用 New-Object 來建立每個標籤。如果命令成功，則無輸出訊息。

```
$tag1 = New-Object Amazon.SimpleSystemsManagement.Model.Tag
$tag1.Key = "Stack"
$tag1.Value = "Production"

Add-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
 "MaintenanceWindow" -Tag $tag1
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 AddTagsToResource。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **CancelCommand** 與 CLI

下列程式碼範例示範如何使用 CancelCommand。

CLI

AWS CLI

範例 1：取消所有執行個體的命令

下列 cancel-command 範例嘗試取消已針對所有執行個體執行的指定命令。

```
aws ssm cancel-command \
    --command-id "662add3d-5831-4a10-b64a-f2ff3EXAMPLE"
```

此命令不會產生輸出。

範例 2：取消特定執行個體的命令

下列 cancel-command 範例嘗試僅取消指定執行個體的命令。

```
aws ssm cancel-command \
```

```
    --command-id "662add3d-5831-4a10-b64a-f2ff3EXAMPLE"
    --instance-ids "i-02573cafcfEXAMPLE"
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Tagging Systems Manager Parameters。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CancelCommand。

PowerShell

PowerShell V4 的工具

範例 1：此範例嘗試取消命令。如果操作成功，則無輸出訊息。

```
Stop-SSMCommand -CommandId "9ded293e-e792-4440-8e3e-7b8ec5feaa38"
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 CancelCommand。

PowerShell V5 的工具

範例 1：此範例嘗試取消命令。如果操作成功，則無輸出訊息。

```
Stop-SSMCommand -CommandId "9ded293e-e792-4440-8e3e-7b8ec5feaa38"
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 CancelCommand。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **CreateActivation** 與 CLI

下列程式碼範例示範如何使用 `CreateActivation`。

CLI

AWS CLI

建立受管執行個體啟用

下列 create-activation 範例示範建立受管執行個體啟用。

```
aws ssm create-activation \
    --default-instance-name "HybridWebServers" \
    --iam-role "HybridWebServersRole" \
    --registration-limit 5
```

輸出：

```
{
    "ActivationId": "5743558d-563b-4457-8682-d16c3EXAMPLE",
    "ActivationCode": "dRmgnYaFv567vEXAMPLE"
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Step 4: Create a Managed-Instance Activation for a Hybrid Environment。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateActivation。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範建立受管執行個體。

```
New-SSMActivation -DefaultInstanceName "MyWebServers" -IamRole
 "SSMAutomationRole" -RegistrationLimit 10
```

輸出：

```
ActivationCode       ActivationId
--------------       ------------
KWChhOxBTiwDcKE9BlKC 08e51e79-1e36-446c-8e63-9458569c1363
```

- 如需 API 詳細資訊，請參閱《 Cmdlet 參考》中的 CreateActivation (V4)。 AWS Tools for PowerShell

PowerShell V5 的工具

範例 1：此範例示範建立受管執行個體。

```
New-SSMActivation -DefaultInstanceName "MyWebServers" -IamRole
 "SSMAutomationRole" -RegistrationLimit 10
```

輸出：

```
ActivationCode       ActivationId
--------------       ------------
KWChhOxBTiwDcKE9BlKC 08e51e79-1e36-446c-8e63-9458569c1363
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 CreateActivation。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **CreateAssociation** 與 CLI

下列程式碼範例示範如何使用 `CreateAssociation`。

CLI

AWS CLI

範例 1：使用執行個體 ID 關聯文件

此範例使用執行個體 ID 來關聯組態文件與執行個體。

```
aws ssm create-association \
    --instance-id "i-0cb2b964d3e14fd9f" \
    --name "AWS-UpdateSSMAgent"
```

輸出：

```
{
    "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"
            }
        ]
    }
}
```

如需詳細資訊，請參閱《AWS Systems Manager API 參考》中的 CreateAssociation。

範例 2：使用目標關聯文件

此範例使用目標來關聯組態文件與執行個體。

```
aws ssm create-association \
    --name "AWS-UpdateSSMAgent" \
    --targets "Key=instanceids,Values=i-0cb2b964d3e14fd9f"
```

輸出：

```
{
    "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"
            }
        ]
    }
}
```

如需詳細資訊，請參閱《AWS Systems Manager API 參考》中的 CreateAssociation。

範例 3：建立僅執行一次的關聯

此範例建立在指定日期和時間只執行一次的新關聯。以過去或現在的日期建立的關聯 (在處理時，該日期已是過去的日期) 會立即執行。

```
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
```

輸出：

```
{
    "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"
            }
        ]
    }
}
```

如需詳細資訊，請參閱《AWS Systems Manager API 參考》中的 CreateAssociation，或《AWS Systems Manager 使用者指南》中的 Reference: Cron and rate expressions for Systems Manager。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateAssociation。

PowerShell

PowerShell V4 的工具

範例 1：此範例使用執行個體 ID 來關聯組態文件與執行個體。

```
New-SSMAssociation -InstanceId "i-0cb2b964d3e14fd9f" -Name "AWS-UpdateSSMAgent"
```

輸出：

```
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 :
```

範例 2：此範例使用目標來關聯組態文件與執行個體。

```
$target = @{Key="instanceids";Values=@("i-0cb2b964d3e14fd9f")}
New-SSMAssociation -Name "AWS-UpdateSSMAgent" -Target $target
```

輸出：

```
Name                  : AWS-UpdateSSMAgent
InstanceId            :
Date                  : 3/1/2017 6:22:21 PM
Status.Name           :
Status.Date           :
Status.Message        :
Status.AdditionalInfo :
```

範例 3：此範例使用目標和參數來關聯組態文件與執行個體。

```
$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
```

輸出：

```
Name                  : Configure-CloudWatch
InstanceId            :
Date                  : 5/17/2018 3:17:44 PM
Status.Name           :
Status.Date           :
Status.Message        :
Status.AdditionalInfo :
```

範例 4：此範例使用 AWS-GatherSoftwareInventory 建立與區域中所有執行個體的關聯。它也提供參數中要收集的自訂檔案和登錄檔位置

```
$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)"
```

輸出：

```
Name                 : AWS-GatherSoftwareInventory
InstanceId           :
Date                 : 6/9/2019 8:57:56 AM
Status.Name          :
Status.Date          :
Status.Message       :
Status.AdditionalInfo :
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 CreateAssociation。

PowerShell V5 的工具

範例 1：此範例使用執行個體 ID 來關聯組態文件與執行個體。

```
New-SSMAssociation -InstanceId "i-0cb2b964d3e14fd9f" -Name "AWS-UpdateSSMAgent"
```

輸出：

```
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 :
```

範例 2：此範例使用目標來關聯組態文件與執行個體。

```
$target = @{Key="instanceids";Values=@("i-0cb2b964d3e14fd9f")}
New-SSMAssociation -Name "AWS-UpdateSSMAgent" -Target $target
```

輸出：

```
Name                 : AWS-UpdateSSMAgent
InstanceId           :
Date                 : 3/1/2017 6:22:21 PM
Status.Name          :
Status.Date          :
Status.Message       :
Status.AdditionalInfo :
```

範例 3：此範例使用目標和參數來關聯組態文件與執行個體。

```
$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
```

輸出：

```
Name                 : Configure-CloudWatch
InstanceId           :
Date                 : 5/17/2018 3:17:44 PM
Status.Name          :
Status.Date          :
Status.Message       :
Status.AdditionalInfo :
```

範例 4：此範例使用 AWS-GatherSoftwareInventory 建立與區域中所有執行個體的關聯。它也提供參數中要收集的自訂檔案和登錄檔位置

```
$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)"
```

輸出：

```
Name                  : AWS-GatherSoftwareInventory
InstanceId            :
Date                  : 6/9/2019 8:57:56 AM
Status.Name           :
Status.Date           :
Status.Message        :
Status.AdditionalInfo :
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 CreateAssociation。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **CreateAssociationBatch** 與 CLI

下列程式碼範例示範如何使用 CreateAssociationBatch。

CLI

AWS CLI

建立多個關聯

此範例會將組態文件與多個執行個體建立關聯。輸出會傳回成功操作和失敗操作的清單 (如適用)。

命令：

```
aws ssm create-association-batch --entries "Name=AWS-
UpdateSSMAgent,InstanceId=i-1234567890abcdef0" "Name=AWS-
UpdateSSMAgent,InstanceId=i-9876543210abcdef0"
```

輸出：

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

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateAssociationBatch。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範將組態文件與多個執行個體關聯。輸出會傳回成功操作和失敗操作的清單 (如適用)。

```
$option1 = @{InstanceId="i-0cb2b964d3e14fd9f";Name=@("AWS-UpdateSSMAgent")}
$option2 = @{InstanceId="i-0000293ffd8c57862";Name=@("AWS-UpdateSSMAgent")}
New-SSMAssociationFromBatch -Entry $option1,$option2
```

輸出：

```
Failed  Successful
------  ----------
{}      {Amazon.SimpleSystemsManagement.Model.FailedCreateAssociation,
 Amazon.SimpleSystemsManagement.Model.FailedCreateAsso...
```

範例 2：此範例會顯示成功操作的完整詳細資訊。

```
$option1 = @{InstanceId="i-0cb2b964d3e14fd9f";Name=@("AWS-UpdateSSMAgent")}
$option2 = @{InstanceId="i-0000293ffd8c57862";Name=@("AWS-UpdateSSMAgent")}
(New-SSMAssociationFromBatch -Entry $option1,$option2).Successful
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 CreateAssociationBatch。

PowerShell V5 的工具

範例 1：此範例示範將組態文件與多個執行個體關聯。輸出會傳回成功操作和失敗操作的清單 (如適用)。

```
$option1 = @{InstanceId="i-0cb2b964d3e14fd9f";Name=@("AWS-UpdateSSMAgent")}
$option2 = @{InstanceId="i-0000293ffd8c57862";Name=@("AWS-UpdateSSMAgent")}
New-SSMAssociationFromBatch -Entry $option1,$option2
```

輸出：

```
Failed  Successful
------  ----------
{}      {Amazon.SimpleSystemsManagement.Model.FailedCreateAssociation,
 Amazon.SimpleSystemsManagement.Model.FailedCreateAsso...
```

範例 2：此範例會顯示成功操作的完整詳細資訊。

```
$option1 = @{InstanceId="i-0cb2b964d3e14fd9f";Name=@("AWS-UpdateSSMAgent")}
$option2 = @{InstanceId="i-0000293ffd8c57862";Name=@("AWS-UpdateSSMAgent")}
(New-SSMAssociationFromBatch -Entry $option1,$option2).Successful
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 CreateAssociationBatch。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`CreateDocument` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `CreateDocument`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 了解基本概念

CLI

AWS CLI

建立文件

以下 `create-document` 範例示範建立 Systems Manager 文件。

```
aws ssm create-document \
    --content file://exampleDocument.yml \
    --name "Example" \
    --document-type "Automation" \
    --document-format YAML
```

輸出：

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

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Creating Systems Manager Documents。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateDocument。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /**
     * 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();
    }
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 CreateDocument。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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;
    }
  }
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 CreateDocument。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範在您的帳戶中建立文件。文件必須為 JSON 格式。如需有關撰寫組態文件的詳細資訊，請參閱《SSM API 參考》中的組態文件。

```
New-SSMDocument -Content (Get-Content -Raw "c:\temp\RunShellScript.json") -Name
 "RunShellScript" -DocumentType "Command"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 CreateDocument。

PowerShell V5 的工具

範例 1：此範例示範在您的帳戶中建立文件。文件必須為 JSON 格式。如需有關撰寫組態文件的詳細資訊，請參閱《SSM API 參考》中的組態文件。

```
New-SSMDocument -Content (Get-Content -Raw "c:\temp\RunShellScript.json") -Name
 "RunShellScript" -DocumentType "Command"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 CreateDocument。

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 CreateDocument。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`CreateMaintenanceWindow` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `CreateMaintenanceWindow`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 了解基本概念

CLI

AWS CLI

範例 1：建立維護時段

下列 `create-maintenance-window` 範例會建立一個新的維護時段，該維護時段每五分鐘 (最長持續兩個小時，視需要) 防止新任務在維護時段執行結束後一小時內啟動、允許未關聯的目標 (尚未向維護時段註冊的執行個體)，以及透過使用其建立者打算在教學課程中使用的自訂標籤來進行表示。

```
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"
```

輸出：

```
{
    "WindowId": "mw-0c50858d01EXAMPLE"
}
```

範例 2：建立只執行一次的維護時段

下列 `create-maintenance-window` 範例會建立只在指定的日期和時間執行一次的新維護時段。

```
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"
```

輸出：

```
{
    "WindowId": "mw-01234567890abcdef"
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Maintenance Windows。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateMaintenanceWindow。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
   /**
    * 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];
    }
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 CreateMaintenanceWindow。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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;
    }
  }
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 CreateMaintenanceWindow。

PowerShell

PowerShell V4 的工具

範例 1：此範例會建立一個具有指定名稱的新維護時段，該維護時段在每個星期二下午 4 點執行 4 小時並有一個 1 小時的截止，而且允許未關聯的目標。

```
New-SSMMaintenanceWindow -Name "MyMaintenanceWindow" -Duration 4 -Cutoff 1 -
AllowUnassociatedTarget $true -Schedule "cron(0 16 ? * TUE *)"
```

輸出：

```
mw-03eb53e1ea7383998
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 CreateMaintenanceWindow。

PowerShell V5 的工具

範例 1：此範例會建立一個具有指定名稱的新維護時段，該維護時段在每個星期二下午 4 點執行 4 小時並有一個 1 小時的截止，而且允許未關聯的目標。

```
New-SSMMaintenanceWindow -Name "MyMaintenanceWindow" -Duration 4 -Cutoff 1 -
AllowUnassociatedTarget $true -Schedule "cron(0 16 ? * TUE *)"
```

輸出：

```
mw-03eb53e1ea7383998
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 CreateMaintenanceWindow。

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 CreateMaintenanceWindow。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`CreateOpsItem` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `CreateOpsItem`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 了解基本概念

CLI

AWS CLI

建立 OpsItems

下列 `create-ops-item` 範例在 OperationalData 中使用 /aws/resources 索引鍵來建立含 Amazon DynamoDB 相關資源的 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"
```

輸出：

```
{
    "OpsItemId": "oi-1a2b3c4d5e6f"
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Creating OpsItems。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreateOpsItem。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /**
     * 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);
            }
        }
    }
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 CreateOpsItem。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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;
```

```
    }
  }
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 CreateOpsItem。

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 CreateOpsItem。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **CreatePatchBaseline** 與 CLI

下列程式碼範例示範如何使用 CreatePatchBaseline。

CLI

AWS CLI

範例 1：建立具有自動核准的修補基準

下列 create-patch-baseline 範例示範建立 Windows Server 修補基準，該基準會在 Microsoft 發行適用於生產環境的修補程式七天之後對它們進行核准。

```
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"
```

輸出：

```
{
    "BaselineId": "pb-045f10b4f3EXAMPLE"
}
```

範例 2：建立具有核准截止日期的修補基準

下列 create-patch-baseline 範例示範建立 Windows Server 修補基準，該基準會對 2020 年 7 月 7 日 (含) 之前發布的適用於生產環境的所有修補程式進行核准。

```
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"
```

輸出：

```
{
    "BaselineId": "pb-045f10b4f3EXAMPLE"
}
```

範例 3：建立核准規則存放在 JSON 檔案中的修補基準

下列 create-patch-baseline 範例示範建立 Amazon Linux 2017.09 修補基準，該基準會在適用於生產環境的修補程式發布 7 天後對它們進行核准、指定修補基準的核准規則，以及指定修補程式的自訂儲存庫。

```
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 的內容：

```
{
    "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"
        }
    ]
}
```

範例 4：建立將指定已核准和遭拒的修補程式的修補基準

下列 create-patch-baseline 範例明確指定作為預設核准規則例外狀況要核准和拒絕的修補程式。

```
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"
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Create a Custom Patch Baseline。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 CreatePatchBaseline。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範針對在生產環境中執行 Windows Server 2019 的受管執行個體建立修補基準，該修補基準在 Microsoft 發行修補程式七天後對修補程式進行核准。

```
$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
```

輸出：

```
pb-0z4z6221c4296b23z
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 CreatePatchBaseline。

PowerShell V5 的工具

範例 1：此範例示範針對在生產環境中執行 Windows Server 2019 的受管執行個體建立修補基準，該修補基準在 Microsoft 發行修補程式七天後對修補程式進行核准。

```
$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
```

輸出：

```
pb-0z4z6221c4296b23z
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 CreatePatchBaseline。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeleteActivation** 與 CLI

下列程式碼範例示範如何使用 DeleteActivation。

CLI

AWS CLI

刪除受管執行個體啟用

下列 delete-activation 範例示範刪除受管執行個體啟用。

```
aws ssm delete-activation \
    --activation-id "aa673477-d926-42c1-8757-1358cEXAMPLE"
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的設定混合環境的 AWS Systems Manager。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteActivation。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範刪除啟用。如果命令成功，則無輸出訊息。

```
Remove-SSMActivation -ActivationId "08e51e79-1e36-446c-8e63-9458569c1363"
```

- 如需 API 詳細資訊，請參閱《 Cmdlet 參考》中的 DeleteActivation (V4)。 AWS Tools for PowerShell

PowerShell V5 的工具

範例 1：此範例示範刪除啟用。如果命令成功，則無輸出訊息。

```
Remove-SSMActivation -ActivationId "08e51e79-1e36-446c-8e63-9458569c1363"
```

- 如需 API 詳細資訊，請參閱《 Cmdlet 參考》中的 DeleteActivation (V5)。 AWS Tools for PowerShell

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeleteAssociation** 與 CLI

下列程式碼範例示範如何使用 `DeleteAssociation`。

CLI

AWS CLI

範例 1：使用關聯 ID 來刪除關聯

下列 `delete-association` 範例示範刪除指定關聯 ID 的關聯。如果命令成功，則無輸出訊息。

```
aws ssm delete-association \
```

```
    --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Editing and creating a new version of an association。

範例 2：刪除關聯

下列 delete-association 範例示範刪除執行個體與文件之間的關聯。如果命令成功，則無輸出訊息。

```
aws ssm delete-association \
    --instance-id "i-1234567890abcdef0" \
    --name "AWS-UpdateSSMAgent"
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Working with associations in Systems Manager。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteAssociation。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範刪除執行個體與文件之間的關聯。如果命令成功，則無輸出訊息。

```
Remove-SSMAssociation -InstanceId "i-0cb2b964d3e14fd9f" -Name "AWS-
UpdateSSMAgent"
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 DeleteAssociation。

PowerShell V5 的工具

範例 1：此範例示範刪除執行個體與文件之間的關聯。如果命令成功，則無輸出訊息。

```
Remove-SSMAssociation -InstanceId "i-0cb2b964d3e14fd9f" -Name "AWS-
UpdateSSMAgent"
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 DeleteAssociation。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`DeleteDocument` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `DeleteDocument`。

CLI

AWS CLI

刪除文件

下列 `delete-document` 範例示範刪除 Systems Manager 文件。

```
aws ssm delete-document \
    --name "Example"
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Creating Systems Manager Documents。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteDocument。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
/**
```

```
     * 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;
        }
    }
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 DeleteDocument。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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;
    }
  }
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DeleteDocument。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範刪除文件。如果命令成功，則無輸出訊息。

```
Remove-SSMDocument -Name "RunShellScript"
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V4) 中的 DeleteDocument。

PowerShell V5 的工具

範例 1：此範例示範刪除文件。如果命令成功，則無輸出訊息。

```
Remove-SSMDocument -Name "RunShellScript"
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 DeleteDocument。

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 DeleteDocument。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`DeleteMaintenanceWindow` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `DeleteMaintenanceWindow`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 了解基本概念

CLI

AWS CLI

刪除維護時段

此 delete-maintenance-window 範例示範移除指定的維護時段。

```
aws ssm delete-maintenance-window \
    --window-id "mw-1a2b3c4d5e6f7g8h9"
```

輸出：

```
{
    "WindowId":"mw-1a2b3c4d5e6f7g8h9"
}
```

如需詳細資訊，請參閱 AWS Systems Manager 使用者指南中的刪除維護時段 (AWS CLI)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteMaintenanceWindow。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /**
     * 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;
        }
    }
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 DeleteMaintenanceWindow。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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;
    }
  }
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DeleteMaintenanceWindow。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範移除維護時段。

```
Remove-SSMMaintenanceWindow -WindowId "mw-06d59c1a07c022145"
```

輸出：

```
mw-06d59c1a07c022145
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 DeleteMaintenanceWindow。

PowerShell V5 的工具

範例 1：此範例示範移除維護時段。

```
Remove-SSMMaintenanceWindow -WindowId "mw-06d59c1a07c022145"
```

輸出：

```
mw-06d59c1a07c022145
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 DeleteMaintenanceWindow。

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 DeleteMaintenanceWindow。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`DeleteOpsItem` 搭配 AWS SDK 使用

以下程式碼範例顯示如何使用 `DeleteOpsItem`。

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 DeleteOpsItem。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeleteParameter** 與 CLI

下列程式碼範例示範如何使用 DeleteParameter。

CLI

AWS CLI

刪除參數

下列 delete-parameter 範例示範刪除指定的單個參數。

```
aws ssm delete-parameter \
    --name "MyParameter"
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Working with Parameter Store。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeleteParameter。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範刪除參數。如果命令成功，則無輸出訊息。

```
Remove-SSMParameter -Name "helloWorld"
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 DeleteParameter。

PowerShell V5 的工具

範例 1：此範例示範刪除參數。如果命令成功，則無輸出訊息。

```
Remove-SSMParameter -Name "helloWorld"
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 DeleteParameter。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeletePatchBaseline** 與 CLI

下列程式碼範例示範如何使用 DeletePatchBaseline。

CLI

AWS CLI

刪除修補基準

下列 delete-patch-baseline 範例示範刪除指定的修補基準。

```
aws ssm delete-patch-baseline \
    --baseline-id "pb-045f10b4f382baeda"
```

輸出：

```
{
    "BaselineId": "pb-045f10b4f382baeda"
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Update or Delete a Patch Baseline (Console)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeletePatchBaseline。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範刪除修補基準。

```
Remove-SSMPatchBaseline -BaselineId "pb-045f10b4f382baeda"
```

輸出：

```
pb-045f10b4f382baeda
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 DeletePatchBaseline。

PowerShell V5 的工具

範例 1：此範例示範刪除修補基準。

```
Remove-SSMPatchBaseline -BaselineId "pb-045f10b4f382baeda"
```

輸出：

```
pb-045f10b4f382baeda
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 DeletePatchBaseline。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeregisterManagedInstance** 與 CLI

下列程式碼範例示範如何使用 `DeregisterManagedInstance`。

CLI

AWS CLI

取消註冊受管執行個體

下列 `deregister-managed-instance` 範例示範取消註冊指定的受管執行個體。

```
aws ssm deregister-managed-instance \
    --instance-id 'mi-08ab247cdfEXAMPLE'
```

此命令不會產生輸出。

如需詳細資訊，請參閱 AWS Systems Manager 使用者指南中的在混合多雲端環境中取消註冊受管節點。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeregisterManagedInstance。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範取消註冊受管執行個體。如果命令成功，則無輸出訊息。

```
Unregister-SSMManagedInstance -InstanceId "mi-08ab247cdf1046573"
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 DeregisterManagedInstance。

PowerShell V5 的工具

範例 1：此範例示範取消註冊受管執行個體。如果命令成功，則無輸出訊息。

```
Unregister-SSMManagedInstance -InstanceId "mi-08ab247cdf1046573"
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 DeregisterManagedInstance。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeregisterPatchBaselineForPatchGroup** 與 CLI

下列程式碼範例示範如何使用 `DeregisterPatchBaselineForPatchGroup`。

CLI

AWS CLI

從修補基準中取消註冊修補程式群組

下列 deregister-patch-baseline-for-patch-group 範例示範從指定的修補基準中取消註冊指定的修補程式群組。

```
aws ssm deregister-patch-baseline-for-patch-group \
    --patch-group "Production" \
    --baseline-id "pb-0ca44a362fEXAMPLE"
```

輸出：

```
{
  "PatchGroup":"Production",
  "BaselineId":"pb-0ca44a362fEXAMPLE"
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Add a Patch Group to a Patch Baseline。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeregisterPatchBaselineForPatchGroup。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範從修補基準中取消註冊修補程式群組。

```
Unregister-SSMPatchBaselineForPatchGroup -BaselineId "pb-045f10b4f382baeda" -
PatchGroup "Production"
```

輸出：

```
BaselineId           PatchGroup
----------           ----------
pb-045f10b4f382baeda Production
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V4) 中的 DeregisterPatchBaselineForPatchGroup。

PowerShell V5 的工具

範例 1：此範例示範從修補基準中取消註冊修補程式群組。

```
Unregister-SSMPatchBaselineForPatchGroup -BaselineId "pb-045f10b4f382baeda" -
PatchGroup "Production"
```

輸出：

```
BaselineId           PatchGroup
----------           ----------
pb-045f10b4f382baeda Production
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 DeregisterPatchBaselineForPatchGroup。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeregisterTargetFromMaintenanceWindow** 與 CLI

下列程式碼範例示範如何使用 DeregisterTargetFromMaintenanceWindow。

CLI

AWS CLI

從維護時段中移除目標

下列 deregister-target-from-maintenance-window 範例示範從指定的維護時段中移除指定目標。

```
aws ssm deregister-target-from-maintenance-window \
    --window-id "mw-ab12cd34ef56gh78" \
    --window-target-id "1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
```

輸出：

```
{
    "WindowId":"mw-ab12cd34ef56gh78",
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

如需詳細資訊，請參閱 AWS Systems Manager 使用者指南中的更新維護時段 (AWS CLI)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeregisterTargetFromMaintenanceWindow。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範從維護時段中移除目標。

```
Unregister-SSMTargetFromMaintenanceWindow -WindowTargetId
 "6ab5c208-9fc4-4697-84b7-b02a6cc25f7d" -WindowId "mw-06cf17cbefcb4bf4f"
```

輸出：

```
WindowId             WindowTargetId
--------             --------------
mw-06cf17cbefcb4bf4f 6ab5c208-9fc4-4697-84b7-b02a6cc25f7d
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V4) 中的 DeregisterTargetFromMaintenanceWindow。

PowerShell V5 的工具

範例 1：此範例示範從維護時段中移除目標。

```
Unregister-SSMTargetFromMaintenanceWindow -WindowTargetId
 "6ab5c208-9fc4-4697-84b7-b02a6cc25f7d" -WindowId "mw-06cf17cbefcb4bf4f"
```

輸出：

```
WindowId             WindowTargetId
--------             --------------
mw-06cf17cbefcb4bf4f 6ab5c208-9fc4-4697-84b7-b02a6cc25f7d
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 DeregisterTargetFromMaintenanceWindow。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeregisterTaskFromMaintenanceWindow** 與 CLI

下列程式碼範例示範如何使用 DeregisterTaskFromMaintenanceWindow。

CLI

AWS CLI

從維護時段中移除任務

下列 deregister-task-from-maintenance-window 範例示範從指定的維護時段中移除指定任務。

```
aws ssm deregister-task-from-maintenance-window \
    --window-id "mw-ab12cd34ef56gh78" \
    --window-task-id "1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d5e6c"
```

輸出：

```
{
    "WindowTaskId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d5e6c",
    "WindowId":"mw-ab12cd34ef56gh78"
}
```

如需詳細資訊，請參閱 Systems Manager 使用者指南中的 Systems Manager 維護 Windows 教學課程 (AWS CLI)。 AWS

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DeregisterTaskFromMaintenanceWindow。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範從維護時段中移除任務。

```
Unregister-SSMTaskFromMaintenanceWindow -WindowTaskId "f34a2c47-ddfd-4c85-
a88d-72366b69af1b" -WindowId "mw-03a342e62c96d31b0"
```

輸出：

```
WindowId             WindowTaskId
```

```
--------              ------------
mw-03a342e62c96d31b0 f34a2c47-ddfd-4c85-a88d-72366b69af1b
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 DeregisterTaskFromMaintenanceWindow。

PowerShell V5 的工具

範例 1：此範例示範從維護時段中移除任務。

```
Unregister-SSMTaskFromMaintenanceWindow -WindowTaskId "f34a2c47-ddfd-4c85-
a88d-72366b69af1b" -WindowId "mw-03a342e62c96d31b0"
```

輸出：

```
WindowId             WindowTaskId
--------             ------------
mw-03a342e62c96d31b0 f34a2c47-ddfd-4c85-a88d-72366b69af1b
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 DeregisterTaskFromMaintenanceWindow。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeActivations** 與 CLI

下列程式碼範例示範如何使用 DescribeActivations。

CLI

AWS CLI

描述啟用

下列describe-activations範例列出您 AWS 帳戶中啟用的詳細資訊。

```
aws ssm describe-activations
```

輸出：

```
{
```

```
    "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
        },
    ]
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Step 4: Create a Managed-Instance Activation for a Hybrid Environment。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeActivations。

PowerShell

PowerShell V4 的工具

範例 1：此範例提供有關您帳戶中啟用的詳細資訊。

```
Get-SSMActivation
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V4) 中的 DescribeActivations。

PowerShell V5 的工具

範例 1：此範例提供有關您帳戶中啟用的詳細資訊。

```
Get-SSMActivation
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 DescribeActivations。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeAssociation** 與 CLI

下列程式碼範例示範如何使用 `DescribeAssociation`。

CLI

AWS CLI

範例 1：取得關聯的詳細資訊

下列 describe-association 範例說明指定關聯 ID 的關聯。

```
aws ssm describe-association \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

輸出：

```
{
    "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"
    }
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Editing and creating a new version of an association。

範例 2：取得特定執行個體和文件之關聯的詳細資訊

下列 `describe-association` 範例說明執行個體與文件之間的關聯。

```
aws ssm describe-association \
    --instance-id "i-1234567890abcdef0" \
    --name "AWS-UpdateSSMAgent"
```

輸出：

```
{
    "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"
            }
        ]
    }
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Editing and creating a new version of an association。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeAssociation。

PowerShell

PowerShell V4 的工具

範例 1：此範例說明執行個體與文件之間的關聯。

```
Get-SSMAssociation -InstanceId "i-0000293ffd8c57862" -Name "AWS-UpdateSSMAgent"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 DescribeAssociation。

PowerShell V5 的工具

範例 1：此範例說明執行個體與文件之間的關聯。

```
Get-SSMAssociation -InstanceId "i-0000293ffd8c57862" -Name "AWS-UpdateSSMAgent"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 DescribeAssociation。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeAssociationExecutionTargets** 與 CLI

下列程式碼範例示範如何使用 DescribeAssociationExecutionTargets。

CLI

AWS CLI

取得關聯執行的詳細資訊

下列 describe-association-execution-targets 範例說明指定的關聯執行。

```
aws ssm describe-association-execution-targets \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab" \
    --execution-id "7abb6378-a4a5-4f10-8312-0123456789ab"
```

輸出：

```
{
    "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"
            }
        }
    ]
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Viewing association histories。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeAssociationExecutionTargets。

PowerShell

PowerShell V4 的工具

範例 1：此範例顯示資源 ID 及其執行狀態，它們是關聯執行目標的一部分

```
Get-SSMAssociationExecutionTarget -AssociationId 123a45a0-
c678-9012-3456-78901234db5e -ExecutionId 123a45a0-c678-9012-3456-78901234db5e |
 Select-Object ResourceId, Status
```

輸出：

```
ResourceId           Status
----------           ------
i-0b1b2a3456f7a890b  Success
i-01c12a45d6fc7a89f  Success
i-0a1caf234f56d7dc8  Success
i-012a3fd45af6dbcfe  Failed
i-0ddc1df23c4a5fb67  Success
```

範例 2：此命令會檢查自昨天以來特定自動化的特定執行，其中關聯有命令文件。它會進一步檢查關聯執行是否失敗，如果是，則會顯示該執行的命令調用詳細資訊以及執行個體 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
    }
}
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V4) 中的 DescribeAssociationExecutionTargets。

PowerShell V5 的工具

範例 1：此範例顯示資源 ID 及其執行狀態，它們是關聯執行目標的一部分

```
Get-SSMAssociationExecutionTarget -AssociationId 123a45a0-
c678-9012-3456-78901234db5e -ExecutionId 123a45a0-c678-9012-3456-78901234db5e |
 Select-Object ResourceId, Status
```

輸出：

```
ResourceId           Status
----------           ------
i-0b1b2a3456f7a890b  Success
i-01c12a45d6fc7a89f  Success
i-0a1caf234f56d7dc8  Success
i-012a3fd45af6dbcfe  Failed
i-0ddc1df23c4a5fb67  Success
```

範例 2：此命令會檢查自昨天以來特定自動化的特定執行，其中關聯有命令文件。它會進一步檢查關聯執行是否失敗，如果是，則會顯示該執行的命令調用詳細資訊以及執行個體 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
    }
}
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 DescribeAssociationExecutionTargets。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeAssociationExecutions** 與 CLI

下列程式碼範例示範如何使用 DescribeAssociationExecutions。

CLI

AWS CLI

範例 1：取得關聯之所有執行的詳細資訊

下列 describe-association-executions 範例說明指定關聯的所有執行。

```
aws ssm describe-association-executions \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

輸出：

```
{
    "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}"
        },
        ...
    ]
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Viewing association histories。

範例 2：取得關聯在特定日期和時間之後的所有執行的詳細資訊

下列 describe-association-executions 範例說明關聯在指定日期和時間之後的所有執行。

```
aws ssm describe-association-executions \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab" \
```

```
    --filters "Key=CreatedTime,Value=2019-02-18T16:00:00Z,Type=GREATER_THAN"
```

輸出：

```
{
    "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}"
        },
        ...
    ]
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Viewing association histories。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeAssociationExecutions。

PowerShell

PowerShell V4 的工具

範例 1：此範例傳回所提供關聯 ID 的執行

```
Get-SSMAssociationExecution -AssociationId 123a45a0-c678-9012-3456-78901234db5e
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V4) 中的 DescribeAssociationExecutions。

PowerShell V5 的工具

範例 1：此範例傳回所提供關聯 ID 的執行

```
Get-SSMAssociationExecution -AssociationId 123a45a0-c678-9012-3456-78901234db5e
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 DescribeAssociationExecutions。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeAutomationExecutions** 與 CLI

下列程式碼範例示範如何使用 DescribeAutomationExecutions。

CLI

AWS CLI

描述自動化執行

下列 describe-automation-executions 範例顯示有關自動化執行的詳細資訊。

```
aws ssm describe-automation-executions \
    --filters Key=ExecutionId,Values=73c8eef8-f4ee-4a05-820c-e354fEXAMPLE
```

輸出：

```
{
    "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"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Running a Simple Automation Workflow。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeAutomationExecutions。

PowerShell

PowerShell V4 的工具

範例 1：此範例說明與帳戶關聯的所有作用中和已終止的自動化執行。

```
Get-SSMAutomationExecutionList
```

輸出：

```
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]]}
```

範例 2：此範例顯示具有 AutomationExecutionStatus 而非 "Success" 之執行的 ExecutionID、文件、執行開始/結束時間戳記

```
Get-SSMAutomationExecutionList | Where-Object AutomationExecutionStatus
 -ne "Success" | Select-Object AutomationExecutionId, DocumentName,
 AutomationExecutionStatus, ExecutionStartTime, ExecutionEndTime | Format-Table -
AutoSize
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 DescribeAutomationExecutions。

PowerShell V5 的工具

範例 1：此範例說明與帳戶關聯的所有作用中和已終止的自動化執行。

```
Get-SSMAutomationExecutionList
```

輸出：

```
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]]}
```

範例 2：此範例顯示具有 AutomationExecutionStatus 而非 "Success" 之執行的 ExecutionID、文件、執行開始/結束時間戳記

```
Get-SSMAutomationExecutionList | Where-Object AutomationExecutionStatus
 -ne "Success" | Select-Object AutomationExecutionId, DocumentName,
 AutomationExecutionStatus, ExecutionStartTime, ExecutionEndTime | Format-Table -
AutoSize
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 DescribeAutomationExecutions。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeAutomationStepExecutions** 與 CLI

下列程式碼範例示範如何使用 DescribeAutomationStepExecutions。

CLI

AWS CLI

範例 1：描述自動化執行的所有步驟

下列 describe-automation-step-executions 範例顯示有關自動化執行步驟的詳細資訊。

```
aws ssm describe-automation-step-executions \
    --automation-execution-id 73c8eef8-f4ee-4a05-820c-e354fEXAMPLE
```

輸出：

```
{
    "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": {}
        }
    ]
}
```

範例 2：描述自動化執行的特定步驟

下列 `describe-automation-step-executions` 範例顯示有關自動化執行的特定步驟的詳細資訊。

```
aws ssm describe-automation-step-executions \
    --automation-execution-id 73c8eef8-f4ee-4a05-820c-e354fEXAMPLE \
    --filters Key=StepExecutionId,Values=95e70479-cf20-4d80-8018-7e4e2EXAMPLE
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Running an Automation Workflow Step by Step (Command Line)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeAutomationStepExecutions。

PowerShell

PowerShell V4 的工具

範例 1：此範例顯示有關自動化工作流程中所有作用中和已終止之步驟執行的資訊。

```
Get-SSMAutomationStepExecution -AutomationExecutionId e1d2bad3-4567-8901-
ae23-456c7c8901be | Select-Object StepName, Action, StepStatus
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V4) 中的 DescribeAutomationStepExecutions。

PowerShell V5 的工具

範例 1：此範例顯示有關自動化工作流程中所有作用中和已終止之步驟執行的資訊。

```
Get-SSMAutomationStepExecution -AutomationExecutionId e1d2bad3-4567-8901-
ae23-456c7c8901be | Select-Object StepName, Action, StepStatus
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 DescribeAutomationStepExecutions。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeAvailablePatches** 與 CLI

下列程式碼範例示範如何使用 DescribeAvailablePatches。

CLI

AWS CLI

取得可用的修補程式

下列 describe-available-patches 範例擷取有關適用於 Windows Server 2019 的所有可用修補程式的詳細資訊，這些修補程式的 MSRC 嚴重性為「關鍵」。

```
aws ssm describe-available-patches \
    --
filters "Key=PRODUCT,Values=WindowsServer2019" "Key=MSRC_SEVERITY,Values=Critical"
```

輸出：

```
{
    "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"
        },
        ...
    ]
}
```

取得特定修補程式的詳細資訊

下列 describe-available-patches 範例擷取有關指定修補程式的詳細資訊。

```
aws ssm describe-available-patches \
    --filters "Key=PATCH_ID,Values=KB4480979"
```

輸出：

```
{
    "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"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 How Patch Manager Operations Work。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeAvailablePatches。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範取得適用於 Windows Server 2012 的所有可用修補程式，這些修補程式的 MSRC 嚴重性為「關鍵」。此範例使用的語法需要 PowerShell 版本 3 或更新版本。

```
$filter1 = @{Key="PRODUCT";Values=@("WindowsServer2012")}
$filter2 = @{Key="MSRC_SEVERITY";Values=@("Critical")}

Get-SSMAvailablePatch -Filter $filter1,$filter2
```

輸出：

```
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
...
```

範例 2：使用 PowerShell 版本 2 時，必須使用 New-Object 來建立每個篩選條件。

```
$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
```

範例 3：此範例示範擷取過去 20 天內發行並適用於符合 WindowsServer2019 的產品的所有更新

```
Get-SSMAvailablePatch | Where-Object ReleaseDate -ge (Get-Date).AddDays(-20)
 | Where-Object Product -eq "WindowsServer2019" | Select-Object ReleaseDate,
 Product, Title
```

輸出：

```
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)
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V4) 中的 DescribeAvailablePatches。

PowerShell V5 的工具

範例 1：此範例示範取得適用於 Windows Server 2012 的所有可用修補程式，這些修補程式的 MSRC 嚴重性為「關鍵」。此範例使用的語法需要 PowerShell 版本 3 或更新版本。

```
$filter1 = @{Key="PRODUCT";Values=@("WindowsServer2012")}
$filter2 = @{Key="MSRC_SEVERITY";Values=@("Critical")}

Get-SSMAvailablePatch -Filter $filter1,$filter2
```

輸出：

```
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
...
```

範例 2：使用 PowerShell 版本 2 時，必須使用 New-Object 來建立每個篩選條件。

```
$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
```

範例 3：此範例示範擷取過去 20 天內發行並適用於符合 WindowsServer2019 的產品的所有更新

```
Get-SSMAvailablePatch | Where-Object ReleaseDate -ge (Get-Date).AddDays(-20)
 | Where-Object Product -eq "WindowsServer2019" | Select-Object ReleaseDate,
 Product, Title
```

輸出：

```
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)
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 DescribeAvailablePatches。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeDocument** 與 CLI

下列程式碼範例示範如何使用 DescribeDocument。

CLI

AWS CLI

顯示文件的詳細資訊

下列describe-document範例顯示您 AWS 帳戶中 Systems Manager 文件的詳細資訊。

```
aws ssm describe-document \
    --name "Example"
```

輸出：

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

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Creating Systems Manager Documents。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeDocument。

PowerShell

PowerShell V4 的工具

範例 1：此範例傳回文件的相關資訊。

```
Get-SSMDocumentDescription -Name "RunShellScript"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 DescribeDocument。

PowerShell V5 的工具

範例 1：此範例傳回文件的相關資訊。

```
Get-SSMDocumentDescription -Name "RunShellScript"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 DescribeDocument。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeDocumentPermission** 與 CLI

下列程式碼範例示範如何使用 DescribeDocumentPermission。

CLI

AWS CLI

描述文件許可

下列 describe-document-permission 範例顯示有關公開共用之 Systems Manager 文件的許可詳細資訊。

```
aws ssm describe-document-permission \
    --name "Example" \
    --permission-type "Share"
```

輸出：

```
{
    "AccountIds": [
        "all"
    ],
    "AccountSharingInfoList": [
        {
            "AccountId": "all",
            "SharedDocumentVersion": "$DEFAULT"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Share a Systems Manager Document。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeDocumentPermission。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了文件的所有版本。

```
Get-SSMDocumentVersionList -Name "RunShellScript"
```

輸出：

```
CreatedDate          DocumentVersion IsDefaultVersion Name
-----------          --------------- ---------------- ----
2/24/2017 5:25:13 AM 1               True             RunShellScript
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 DescribeDocumentPermission。

PowerShell V5 的工具

範例 1：此範例列出了文件的所有版本。

```
Get-SSMDocumentVersionList -Name "RunShellScript"
```

輸出：

```
CreatedDate          DocumentVersion IsDefaultVersion Name
-----------          --------------- ---------------- ----
2/24/2017 5:25:13 AM 1               True             RunShellScript
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 DescribeDocumentPermission。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeEffectiveInstanceAssociations** 與 CLI

下列程式碼範例示範如何使用 DescribeEffectiveInstanceAssociations。

CLI

AWS CLI

取得執行個體有效關聯的詳細資訊

下列 describe-effective-instance-associations 範例示範擷取有關執行個體有效關聯的詳細資訊。

命令：

```
aws ssm describe-effective-instance-associations --instance-
id "i-1234567890abcdef0"
```

輸出：

```
{
    "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"
        }
    ]
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeEffectiveInstanceAssociations。

PowerShell

PowerShell V4 的工具

範例 1：此範例說明執行個體的有效關聯。

```
Get-SSMEffectiveInstanceAssociationList -InstanceId "i-0000293ffd8c57862" -
MaxResult 5
```

輸出：

```
AssociationId                        Content
-------------                        -------
d8617c07-2079-4c18-9847-1655fc2698b0 {...
```

範例 2：此範例顯示執行個體有效關聯的內容。

```
(Get-SSMEffectiveInstanceAssociationList -InstanceId "i-0000293ffd8c57862" -
MaxResult 5).Content
```

輸出：

```
{
    "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 }}"
                }
            ]
        }
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V4) 中的 DescribeEffectiveInstanceAssociations。

PowerShell V5 的工具

範例 1：此範例說明執行個體的有效關聯。

```
Get-SSMEffectiveInstanceAssociationList -InstanceId "i-0000293ffd8c57862" -
MaxResult 5
```

輸出：

```
AssociationId                        Content
-------------                        -------
d8617c07-2079-4c18-9847-1655fc2698b0 {...
```

範例 2：此範例顯示執行個體有效關聯的內容。

```
(Get-SSMEffectiveInstanceAssociationList -InstanceId "i-0000293ffd8c57862" -
MaxResult 5).Content
```

輸出：

```
{
    "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 }}"
                }
            ]
        }
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 DescribeEffectiveInstanceAssociations。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeEffectivePatchesForPatchBaseline** 與 CLI

下列程式碼範例示範如何使用 DescribeEffectivePatchesForPatchBaseline。

CLI

AWS CLI

範例 1：取得由自訂修補基準定義的所有修補程式

下列describe-effective-patches-for-patch-baseline範例會傳回目前 AWS 帳戶中自訂修補程式基準定義的修補程式。請注意，對於自訂基準，--baseline-id 只需要 ID。

```
aws ssm describe-effective-patches-for-patch-baseline \
    --baseline-id "pb-08b654cf9b9681f04"
```

輸出：

```
{
    "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--"
}
```

範例 2：取得受 AWS 管修補程式基準定義的所有修補程式

下列`describe-effective-patches-for-patch-baseline`範例會傳回 AWS 受管修補程式基準定義的修補程式。請注意，對於 AWS 受管基準， 需要完整的基準 ARN `--baseline-id`

```
aws ssm describe-effective-patches-for-patch-baseline \
    --baseline-id "arn:aws:ssm:us-east-2:733109147000:patchbaseline/
pb-020d361a05defe4ed"
```

如需範例輸出，請參閱範例 1。

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 How Security Patches Are Selected。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeEffectivePatchesForPatchBaseline。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了所有修補基準，結果清單上限為 1。

```
Get-SSMEffectivePatchesForPatchBaseline -BaselineId "pb-0a2f1059b670ebd31" -
MaxResult 1
```

輸出：

```
Patch                                       PatchStatus
-----                                       -----------
Amazon.SimpleSystemsManagement.Model.Patch
 Amazon.SimpleSystemsManagement.Model.PatchStatus
```

範例 2：此範例顯示所有修補基準的修補程式狀態，結果清單上限為 1。

```
(Get-SSMEffectivePatchesForPatchBaseline -BaselineId "pb-0a2f1059b670ebd31" -
MaxResult 1).PatchStatus
```

輸出：

```
ApprovalDate          DeploymentStatus
------------          ----------------
12/21/2010 6:00:00 PM APPROVED
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V4) 中的 DescribeEffectivePatchesForPatchBaseline。

PowerShell V5 的工具

範例 1：此範例列出了所有修補基準，結果清單上限為 1。

```
Get-SSMEffectivePatchesForPatchBaseline -BaselineId "pb-0a2f1059b670ebd31" -
MaxResult 1
```

輸出：

```
Patch                                       PatchStatus
-----                                       -----------
Amazon.SimpleSystemsManagement.Model.Patch
 Amazon.SimpleSystemsManagement.Model.PatchStatus
```

範例 2：此範例顯示所有修補基準的修補程式狀態，結果清單上限為 1。

```
(Get-SSMEffectivePatchesForPatchBaseline -BaselineId "pb-0a2f1059b670ebd31" -
MaxResult 1).PatchStatus
```

輸出：

```
ApprovalDate          DeploymentStatus
------------          ----------------
12/21/2010 6:00:00 PM APPROVED
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 DescribeEffectivePatchesForPatchBaseline。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeInstanceAssociationsStatus** 與 CLI

下列程式碼範例示範如何使用 DescribeInstanceAssociationsStatus。

CLI

AWS CLI

描述執行個體關聯的狀態

此範例顯示執行個體關聯的詳細資訊。

命令：

```
aws ssm describe-instance-associations-status --instance-id "i-1234567890abcdef0"
```

輸出：

```
{
  "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"
        }
    ]
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeInstanceAssociationsStatus。

PowerShell

PowerShell V4 的工具

範例 1：此範例顯示執行個體關聯的詳細資訊。

```
Get-SSMInstanceAssociationsStatus -InstanceId "i-0000293ffd8c57862"
```

輸出：

```
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
```

範例 2：此範例檢查指定執行個體 ID 的執行個體關聯狀態，並進一步顯示這些關聯的執行狀態

```
Get-SSMInstanceAssociationsStatus -InstanceId i-012e3cb4df567e8aa | ForEach-
Object {Get-SSMAssociationExecution -AssociationId .AssociationId}
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 DescribeInstanceAssociationsStatus。

PowerShell V5 的工具

範例 1：此範例顯示執行個體關聯的詳細資訊。

```
Get-SSMInstanceAssociationsStatus -InstanceId "i-0000293ffd8c57862"
```

輸出：

```
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
```

範例 2：此範例檢查指定執行個體 ID 的執行個體關聯狀態，並進一步顯示這些關聯的執行狀態

```
Get-SSMInstanceAssociationsStatus -InstanceId i-012e3cb4df567e8aa | ForEach-
Object {Get-SSMAssociationExecution -AssociationId .AssociationId}
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 DescribeInstanceAssociationsStatus。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeInstanceInformation** 與 CLI

下列程式碼範例示範如何使用 DescribeInstanceInformation。

CLI

AWS CLI

範例 1：描述受管執行個體資訊

下列 describe-instance-information 範例擷取每個受管執行個體的詳細資訊。

```
aws ssm describe-instance-information
```

範例 2：描述有關特定受管執行個體的資訊

下列 describe-instance-information 範例顯示受管執行個體的詳細資訊i-028ea792daEXAMPLE。

```
aws ssm describe-instance-information \
```

```
    --filters "Key=InstanceIds,Values=i-028ea792daEXAMPLE"
```

範例 3：描述有關具有特定標籤索引鍵之受管執行個體的資訊

下列 describe-instance-information 範例顯示具有標籤索引鍵 DEV 之受管執行個體的詳細資訊。

```
aws ssm describe-instance-information \
    --filters "Key=tag-key,Values=DEV"
```

輸出：

```
{
    "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
                }
            }
        }
    ]
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Managed Instances。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeInstanceInformation。

PowerShell

PowerShell V4 的工具

範例 1：此範例顯示每個執行個體的詳細資訊。

```
Get-SSMInstanceInformation
```

輸出：

```
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
```

範例 2：此範例示範如何使用 -Filter 參數，僅篩選 區域中`us-east-1`具有 `AgentVersion`之的 AWS Systems Manager 執行個體的結果`2.2.800.0`。InstanceInformation API 參考主題中提供了有效的 -Filter 鍵值清單 (https://docs.aws.amazon.com/systems-manager/latest/APIReference/API_InstanceInformation.html#systemsmanager-Type-InstanceInformation-ActivationId)。

```
$Filters = @{
    Key="AgentVersion"
    Values="2.2.800.0"
}
```

```
Get-SSMInstanceInformation -Region us-east-1 -Filter $Filters
```

輸出：

```
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
```

範例 3：此範例示範如何使用 -InstanceInformationFilterList 參數，將結果篩選為 **Windows**或 區域中**us-east-1PlatformTypes**的 AWS Systems Manager 執行個體**Linux**。InstanceInformation API 參考主題中提供了有效的 -InstanceInformationFilterList 鍵值清單 (https://docs.aws.amazon.com/systems-manager/latest/APIReference/API_InstanceInformationFilter.html)。

```
$Filters = @{
   Key="PlatformTypes"
   ValueSet=("Windows","Linux")
}
Get-SSMInstanceInformation -Region us-east-1 -InstanceInformationFilterList
 $Filters
```

輸出：

```
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
```

範例 4：此範例列出了 ssm 受管執行個體，並將 InstanceId、PingStatus、LastPingDateTime 和 PlatformName 匯出至 csv 檔案。

```
Get-SSMInstanceInformation | Select-Object InstanceId, PingStatus,
 LastPingDateTime, PlatformName | Export-Csv Instance-details.csv -
NoTypeInformation
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 DescribeInstanceInformation。

PowerShell V5 的工具

範例 1：此範例顯示每個執行個體的詳細資訊。

```
Get-SSMInstanceInformation
```

輸出：

```
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
```

範例 2：此範例示範如何使用 -Filter 參數，僅篩選 區域中**us-east-1**具有 **AgentVersion**之的 AWS Systems Manager 執行個體的結果**2.2.800.0**。InstanceInformation API 參考主題中提供了有效的 -Filter 鍵值清單 (https://docs.aws.amazon.com/systems-manager/latest/APIReference/API_InstanceInformation.html#systemsmanager-Type-InstanceInformation-ActivationId)。

```
$Filters = @{
    Key="AgentVersion"
    Values="2.2.800.0"
}
Get-SSMInstanceInformation -Region us-east-1 -Filter $Filters
```

輸出：

```
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
```

範例 3：此範例示範如何使用 -InstanceInformationFilterList 參數，將結果篩選為**Windows**或 區域中**us-east-1PlatformTypes**的 AWS Systems Manager 執行個體**Linux**。InstanceInformation API 參考主題中提供了有效的 -InstanceInformationFilterList 鍵值清單 (https://docs.aws.amazon.com/systems-manager/latest/APIReference/API_InstanceInformationFilter.html)。

```
$Filters = @{
   Key="PlatformTypes"
   ValueSet=("Windows","Linux")
}
```

```
Get-SSMInstanceInformation -Region us-east-1 -InstanceInformationFilterList
 $Filters
```

輸出：

```
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
```

範例 4：此範例列出了 ssm 受管執行個體，並將 InstanceId、PingStatus、LastPingDateTime 和 PlatformName 匯出至 csv 檔案。

```
Get-SSMInstanceInformation | Select-Object InstanceId, PingStatus,
 LastPingDateTime, PlatformName | Export-Csv Instance-details.csv -
NoTypeInformation
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 DescribeInstanceInformation。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeInstancePatchStates** 與 CLI

下列程式碼範例示範如何使用 DescribeInstancePatchStates。

CLI

AWS CLI

取得執行個體的修補程式摘要狀態

此 describe-instance-patch-states 範例示範取得執行個體的修補程式摘要狀態。

```
aws ssm describe-instance-patch-states \
    --instance-ids "i-1234567890abcdef0"
```

輸出：

```
{
    "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"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS 使用者指南》中的 About Patch Compliance。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeInstancePatchStates。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範取得一個執行個體的修補程式摘要狀態。

```
Get-SSMInstancePatchState -InstanceId "i-08ee91c0b17045407"
```

範例 2：此範例示範取得兩個執行個體的修補程式摘要狀態。

```
Get-SSMInstancePatchState -InstanceId "i-08ee91c0b17045407","i-09a618aec652973a9"
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 DescribeInstancePatchStates。

PowerShell V5 的工具

範例 1：此範例示範取得一個執行個體的修補程式摘要狀態。

```
Get-SSMInstancePatchState -InstanceId "i-08ee91c0b17045407"
```

範例 2：此範例示範取得兩個執行個體的修補程式摘要狀態。

```
Get-SSMInstancePatchState -InstanceId "i-08ee91c0b17045407","i-09a618aec652973a9"
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 DescribeInstancePatchStates。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeInstancePatchStatesForPatchGroup** 與 CLI

下列程式碼範例示範如何使用 DescribeInstancePatchStatesForPatchGroup。

CLI

AWS CLI

範例 1：取得修補程式群組的執行個體狀態

下列 describe-instance-patch-states-for-patch-group 範例示範擷取有關指定修補程式群組的每個執行個體之修補程式摘要狀態的詳細資訊。

```
aws ssm describe-instance-patch-states-for-patch-group \
    --patch-group "Production"
```

輸出：

```
{
    "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
        }
    ]
}
```

範例 2：取得缺少超過五個修補程式之修補程式群組的執行個體狀態

下列 describe-instance-patch-states-for-patch-group 範例針對缺少超過五個修補程式的執行個體，擷取有關指定修補程式群組之修補程式摘要狀態的詳細資訊。

```
aws ssm describe-instance-patch-states-for-patch-group \
    --filters Key=MissingCount,Type=GreaterThan,Values=5 \
    --patch-group "Production"
```

輸出：

```
{
    "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
        }
    ]
}
```

範例 3：取得修補程式群組的執行個體狀態，其中需要重新啟動的執行個體少於十個

下列 describe-instance-patch-states-for-patch-group 範例示範擷取有關執行個體的指定修補程式群組之修補程式摘要狀態的詳細資訊，其中需要重新啟動的執行個體少於十個。

```
aws ssm describe-instance-patch-states-for-patch-group \
    --filters Key=InstalledPendingRebootCount,Type=LessThan,Values=10 \
    --patch-group "Production"
```

輸出：

```
{
    "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
        }
    ]
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Understanding patch compliance state values。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeInstancePatchStatesForPatchGroup。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範取得修補程式群組中每個執行個體的修補程式摘要狀態。

```
Get-SSMInstancePatchStatesForPatchGroup -PatchGroup "Production"
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V4) 中的 DescribeInstancePatchStatesForPatchGroup。

PowerShell V5 的工具

範例 1：此範例示範取得修補程式群組中每個執行個體的修補程式摘要狀態。

```
Get-SSMInstancePatchStatesForPatchGroup -PatchGroup "Production"
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 DescribeInstancePatchStatesForPatchGroup。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeInstancePatches** 與 CLI

下列程式碼範例示範如何使用 DescribeInstancePatches。

CLI

AWS CLI

範例 1：取得執行個體的修補程式狀態詳細資訊

下列 describe-instance-patches 範例示範擷取有關指定執行個體修補程式的詳細資訊。

```
aws ssm describe-instance-patches \
    --instance-id "i-1234567890abcdef0"
```

輸出：

```
{
    "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--"
}
```

範例 2：取得狀態為「缺少」的執行個體修補程式清單

下列 describe-instance-patches 範例示範擷取有關指定執行個體處於「缺少」狀態的修補程式資訊。

```
aws ssm describe-instance-patches \
    --instance-id "i-1234567890abcdef0" \
    --filters Key=State,Values=Missing
```

輸出：

```
{
    "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--"
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 About Patch Compliance States。

範例 3：取得執行個體自指定 InstalledTime 以來安裝的修補程式清單

下列 `describe-instance-patches` 範例結合使用 `--filters` 和 `--query`，擷取有關指定執行個體自指定時間以來所安裝修補程式的資訊。

```
aws ssm describe-instance-patches \
    --instance-id "i-1234567890abcdef0" \
    --filters Key=State,Values=Installed \
    --query "Patches[?InstalledTime >= `2023-01-01T16:00:00`]"
```

輸出：

```
{
    "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--"
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeInstancePatches。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範取得執行個體的修補程式合規性詳細資訊。

```
Get-SSMInstancePatch -InstanceId "i-08ee91c0b17045407"
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 DescribeInstancePatches。

PowerShell V5 的工具

範例 1：此範例示範取得執行個體的修補程式合規性詳細資訊。

```
Get-SSMInstancePatch -InstanceId "i-08ee91c0b17045407"
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 DescribeInstancePatches。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeMaintenanceWindowExecutionTaskInvocations** 與 CLI

下列程式碼範例示範如何使用 DescribeMaintenanceWindowExecutionTaskInvocations。

CLI

AWS CLI

取得針對維護時段任務執行所進行的特定任務調用

下列 describe-maintenance-window-execution-task-invocations 範例列出了作為指定維護時段執行的一部分執行之指定任務的調用。

```
aws ssm describe-maintenance-window-execution-task-invocations \
    --window-execution-id "518d5565-5969-4cca-8f0e-da3b2a638355" \
    --task-id "ac0c6ae1-daa3-4a89-832e-d384503b6586"
```

輸出：

```
{
    "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"
        }
    ]
}
```

如需詳細資訊，請參閱 AWS Systems Manager 使用者指南中的檢視任務和任務執行 (AWS CLI) 的相關資訊。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeMaintenanceWindowExecutionTaskInvocations。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了作為維護時段執行的一部分執行之任務的調用。

```
Get-SSMMaintenanceWindowExecutionTaskInvocationList -TaskId "ac0c6ae1-
daa3-4a89-832e-d384503b6586" -WindowExecutionId "518d5565-5969-4cca-8f0e-
da3b2a638355"
```

輸出：

```
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    :
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V4) 中的 DescribeMaintenanceWindowExecutionTaskInvocations。

PowerShell V5 的工具

範例 1：此範例列出了作為維護時段執行的一部分執行之任務的調用。

```
Get-SSMMaintenanceWindowExecutionTaskInvocationList -TaskId "ac0c6ae1-
daa3-4a89-832e-d384503b6586" -WindowExecutionId "518d5565-5969-4cca-8f0e-
da3b2a638355"
```

輸出：

```
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    :
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 DescribeMaintenanceWindowExecutionTaskInvocations。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeMaintenanceWindowExecutionTasks** 與 CLI

下列程式碼範例示範如何使用 DescribeMaintenanceWindowExecutionTasks。

CLI

AWS CLI

列出與維護時段執行關聯的所有任務

下列 ssm describe-maintenance-window-execution-tasks 範例列出了與指定維護時段執行關聯的任務。

```
aws ssm describe-maintenance-window-execution-tasks \
    --window-execution-id "518d5565-5969-4cca-8f0e-da3b2EXAMPLE"
```

輸出：

```
{
    "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"
        }
    ]
}
```

如需詳細資訊，請參閱 AWS Systems Manager 使用者指南中的檢視任務和任務執行 (AWS CLI) 的相關資訊。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeMaintenanceWindowExecutionTasks。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了與維護時段執行關聯的任務。

```
Get-SSMMaintenanceWindowExecutionTaskList -WindowExecutionId
 "518d5565-5969-4cca-8f0e-da3b2a638355"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V4) 中的 DescribeMaintenanceWindowExecutionTasks。

PowerShell V5 的工具

範例 1：此範例列出了與維護時段執行關聯的任務。

```
Get-SSMMaintenanceWindowExecutionTaskList -WindowExecutionId
 "518d5565-5969-4cca-8f0e-da3b2a638355"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 DescribeMaintenanceWindowExecutionTasks。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeMaintenanceWindowExecutions** 與 CLI

下列程式碼範例示範如何使用 DescribeMaintenanceWindowExecutions。

CLI

AWS CLI

範例 1：列出維護時段的所有執行

下列 describe-maintenance-window-executions 範例列出了指定維護時段的所有執行。

```
aws ssm describe-maintenance-window-executions \
    --window-id "mw-ab12cd34eEXAMPLE"
```

輸出：

```
{
    "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"
        }
    ]
}
```

範例 2：列出指定日期之前維護時段的所有執行

下列 describe-maintenance-window-executions 範例列出了指定日期之前指定維護時段的所有執行。

```
aws ssm describe-maintenance-window-executions \
    --window-id "mw-ab12cd34eEXAMPLE" \
    --filters "Key=ExecutedBefore,Values=2021-08-03T00:00:00Z"
```

輸出：

```
{
    "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"
    }
    ]
}
```

範例 3：列出指定日期後維護時段的所有執行

下列 describe-maintenance-window-executions 範例列出了指定日期之後指定維護時段的所有執行。

```
aws ssm describe-maintenance-window-executions \
    --window-id "mw-ab12cd34eEXAMPLE" \
    --filters "Key=ExecutedAfter,Values=2021-08-04T00:00:00Z"
```

輸出：

```
{
    "WindowExecutions": [
        {
        "WindowId": "mw-ab12cd34eEXAMPLE",
        "WindowExecutionId": "6027b513-64fe-4cf0-be7d-1191aEXAMPLE",
        "Status": "IN_PROGRESS",
        "StartTime": "2021-08-04T11:00:00.000000-07:00"
        }
    ]
}
```

如需詳細資訊，請參閱 AWS Systems Manager 使用者指南中的檢視任務和任務執行 (AWS CLI) 的相關資訊。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeMaintenanceWindowExecutions。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了維護時段的所有執行。

```
Get-SSMMaintenanceWindowExecutionList -WindowId "mw-03eb9db42890fb82d"
```

輸出：

```
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
```

範例 2：此範例列出了指定日期之前維護時段的所有執行。

```
$option1 = @{Key="ExecutedBefore";Values=@("2016-11-04T05:00:00Z")}
Get-SSMMaintenanceWindowExecutionList -WindowId "mw-03eb9db42890fb82d" -Filter
 $option1
```

範例 3：此範例列出了指定日期之後維護時段的所有執行。

```
$option1 = @{Key="ExecutedAfter";Values=@("2016-11-04T05:00:00Z")}
Get-SSMMaintenanceWindowExecutionList -WindowId "mw-03eb9db42890fb82d" -Filter
 $option1
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 DescribeMaintenanceWindowExecutions。

PowerShell V5 的工具

範例 1：此範例列出了維護時段的所有執行。

```
Get-SSMMaintenanceWindowExecutionList -WindowId "mw-03eb9db42890fb82d"
```

輸出：

```
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
```

範例 2：此範例列出了指定日期之前維護時段的所有執行。

```
$option1 = @{Key="ExecutedBefore";Values=@("2016-11-04T05:00:00Z")}
Get-SSMMaintenanceWindowExecutionList -WindowId "mw-03eb9db42890fb82d" -Filter
 $option1
```

範例 3：此範例列出了指定日期之後維護時段的所有執行。

```
$option1 = @{Key="ExecutedAfter";Values=@("2016-11-04T05:00:00Z")}
Get-SSMMaintenanceWindowExecutionList -WindowId "mw-03eb9db42890fb82d" -Filter
 $option1
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 DescribeMaintenanceWindowExecutions。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeMaintenanceWindowTargets** 與 CLI

下列程式碼範例示範如何使用 DescribeMaintenanceWindowTargets。

CLI

AWS CLI

範例 1：列出維護時段的所有目標

下列 describe-maintenance-window-targets 範例列出了維護時段的所有目標。

```
aws ssm describe-maintenance-window-targets \
    --window-id "mw-06cf17cbefEXAMPLE"
```

輸出：

```
{
    "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"
        }
    ]
}
```

範例 2：列出符合特定擁有者資訊值的維護時段目標

此 `describe-maintenance-window-targets` 範例列出了具有特定值的維護時段的所有目標。

```
aws ssm describe-maintenance-window-targets \
    --window-id "mw-0ecb1226ddEXAMPLE" \
    --filters "Key=OwnerInformation,Values=CostCenter1"
```

輸出：

```
{
    "Targets": [
        {
            "WindowId": "mw-0ecb1226ddEXAMPLE",
            "WindowTargetId": "da89dcc3-7f9c-481d-ba2b-edcb7d0057f9",
            "ResourceType": "INSTANCE",
            "Targets": [
                {
                    "Key": "tag:Environment",
                    "Values": [
                        "Prod"
                    ]
                }
            ],
            "OwnerInformation": "CostCenter1",
            "Name": "ProdTarget1"
        }
    ]
}
```

如需詳細資訊，請參閱 AWS Systems Manager 使用者指南中的檢視維護 Windows (AWS CLI) 的相關資訊。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeMaintenanceWindowTargets。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了維護時段的所有目標。

```
Get-SSMMaintenanceWindowTarget -WindowId "mw-06cf17cbefcb4bf4f"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V4) 中的 DescribeMaintenanceWindowTargets。

PowerShell V5 的工具

範例 1：此範例列出了維護時段的所有目標。

```
Get-SSMMaintenanceWindowTarget -WindowId "mw-06cf17cbefcb4bf4f"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 DescribeMaintenanceWindowTargets。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeMaintenanceWindowTasks** 與 CLI

下列程式碼範例示範如何使用 DescribeMaintenanceWindowTasks。

CLI

AWS CLI

範例 1：列出維護時段的所有任務

下列 `describe-maintenance-window-tasks` 範例列出了指定維護時段的所有任務。

```
aws ssm describe-maintenance-window-tasks \
    --window-id "mw-06cf17cbefEXAMPLE"
```

輸出：

```
{
    "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"
                    ]
                }
            ]
        }
    ]
}
```

範例 2：列出叫用 AWS-RunPowerShellScript 命令文件的維護時段的所有任務

下列 `describe-maintenance-window-tasks` 範例列出了調用 AWS-RunPowerShellScript 命令文件之指定維護時段的所有任務。

```
aws ssm describe-maintenance-window-tasks \
    --window-id "mw-ab12cd34eEXAMPLE" \
    --filters "Key=TaskArn,Values=AWS-RunPowerShellScript"
```

輸出：

```
{
    "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"
        }
    ]
}
```

範例 3：列出優先順序為 3 的維護時段的所有任務

下列 describe-maintenance-window-tasks 範例列出了 Priority 為 3 的指定維護時段的所有任務。

```
aws ssm describe-maintenance-window-tasks \
    --window-id "mw-ab12cd34eEXAMPLE" \
    --filters "Key=Priority,Values=3"
```

輸出：

```
{
    "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"
        }
    ]
}
```

範例 4：列出優先順序為 1 且使用 Run Command 的維護時段的所有任務

此 `describe-maintenance-window-tasks` 範例列出了 `Priority` 為 1 且使用 Run Command 的指定維護時段的所有任務。

```
aws ssm describe-maintenance-window-tasks \
    --window-id "mw-ab12cd34eEXAMPLE" \
    --filters "Key=Priority,Values=1" "Key=TaskType,Values=RUN_COMMAND"
```

輸出：

```
{
```

```
    "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"
        }
    ]
}
```

如需詳細資訊，請參閱 AWS Systems Manager 使用者指南中的檢視維護時段 (AWS CLI) 的相關資訊。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeMaintenanceWindowTasks。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了維護時段的所有任務。

```
Get-SSMMaintenanceWindowTaskList -WindowId "mw-06cf17cbefcb4bf4f"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 DescribeMaintenanceWindowTasks。

PowerShell V5 的工具

範例 1：此範例列出了維護時段的所有任務。

```
Get-SSMMaintenanceWindowTaskList -WindowId "mw-06cf17cbefcb4bf4f"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 DescribeMaintenanceWindowTasks。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeMaintenanceWindows** 與 CLI

下列程式碼範例示範如何使用 DescribeMaintenanceWindows。

CLI

AWS CLI

範例 1：列出所有維護時段

下列describe-maintenance-windows範例列出目前區域中 AWS 您帳戶中的所有維護時段。

```
aws ssm describe-maintenance-windows
```

輸出：

```
{
    "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"
        },
    ]
}
```

列出所有已啟用的維護時段

下列 describe-maintenance-windows 範例列出了所有已啟用的維護時段。

```
aws ssm describe-maintenance-windows \
    --filters "Key=Enabled,Values=true"
```

範例 3：列出符合特定名稱的維護時段

此 describe-maintenance-windows 範例列出了具有指定名稱的所有維護時段。

```
aws ssm describe-maintenance-windows \
    --filters "Key=Name,Values=MyMaintenanceWindow"
```

如需詳細資訊，請參閱 AWS Systems Manager 使用者指南中的檢視維護 Windows (AWS CLI) 的相關資訊。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeMaintenanceWindows。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了帳戶的所有維護時段。

```
Get-SSMMaintenanceWindowList
```

輸出：

```
Cutoff   : 1
Duration : 4
Enabled  : True
Name     : My-First-Maintenance-Window
WindowId : mw-06d59c1a07c022145
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 DescribeMaintenanceWindows。

PowerShell V5 的工具

範例 1：此範例列出了帳戶的所有維護時段。

```
Get-SSMMaintenanceWindowList
```

輸出：

```
Cutoff   : 1
Duration : 4
Enabled  : True
Name     : My-First-Maintenance-Window
WindowId : mw-06d59c1a07c022145
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 DescribeMaintenanceWindows。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`DescribeOpsItems` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `DescribeOpsItems`。

CLI

AWS CLI

列出一組 OpsItems

下列`describe-ops-items`範例顯示您 AWS 帳戶中所有開啟 OpsItems 的清單。

```
aws ssm describe-ops-items \
    --ops-item-filters "Key=Status,Values=Open,Operator=Equal"
```

輸出：

```
{
    "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"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Working with OpsItems。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeOpsItems。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /**
     * 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;
        }
    }
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 DescribeOpsItems。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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;
  }
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 DescribeOpsItems。

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 DescribeOpsItems。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DescribeParameters 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DescribeParameters。

CLI

AWS CLI

範例 1：列出所有參數

下列describe-parameters範例列出目前 AWS 帳戶和區域中的所有參數。

```
aws ssm describe-parameters
```

輸出：

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

範例 2：列出符合特定中繼資料的所有參數

此 `describe-parameters` 範例列出了符合篩選條件的所有參數。

aws ssm describe-parameters --filters "Key=Type,Values=StringList"

輸出：

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

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Searching for Systems Manager Parameters。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribeParameter。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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);
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 DescribeParameters。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了所有參數。

```
Get-SSMParameterList
```

輸出：

```
Description      :
KeyId            :
LastModifiedDate : 3/3/2017 6:58:23 PM
LastModifiedUser : arn:aws:iam::123456789012:user/admin
Name             : Welcome
Type             : String
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 DescribeParameters。

PowerShell V5 的工具

範例 1：此範例列出了所有參數。

```
Get-SSMParameterList
```

輸出：

```
Description      :
KeyId            :
LastModifiedDate : 3/3/2017 6:58:23 PM
LastModifiedUser : arn:aws:iam::123456789012:user/admin
Name             : Welcome
Type             : String
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 DescribeParameters。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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(())
}
```

- 如需 API 詳細資訊，請參閱《適用於 Rust 的AWS SDK API 參考》中的 DescribeParameters。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribePatchBaselines** 與 CLI

下列程式碼範例示範如何使用 `DescribePatchBaselines`。

CLI

AWS CLI

範例 1：列出所有修補基準

下列 `describe-patch-baselines` 範例示範擷取目前區域的帳戶中所有修補基準的詳細資訊。

```
aws ssm describe-patch-baselines
```

輸出：

```
{
    "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"
        }
    ]
}
```

範例 2：列出 提供的所有修補程式基準 AWS

下列`describe-patch-baselines`範例列出 提供的所有修補程式基準 AWS。

```
aws ssm describe-patch-baselines \
    --filters "Key=OWNER,Values=[AWS]"
```

範例 3：列出您擁有的所有修補基準

下列 describe-patch-baselines 範例列出了在目前區域的帳戶中建立的所有自訂修補基準。

```
aws ssm describe-patch-baselines \
    --filters "Key=OWNER,Values=[Self]"
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 About Predefined and Custom Patch Baselines。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribePatchBaselines。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了所有修補基準。

```
Get-SSMPatchBaseline
```

輸出：

```
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...
```

範例 2：此範例列出 提供的所有修補程式基準 AWS。此範例使用的語法需要 PowerShell 版本 3 或更新版本。

```
$filter1 = @{Key="OWNER";Values=@("AWS")}
```

輸出：

```
Get-SSMPatchBaseline -Filter $filter1
```

範例 3：此範例列出了您作為擁有者的所有修補基準。此範例使用的語法需要 PowerShell 版本 3 或更新版本。

```
$filter1 = @{Key="OWNER";Values=@("Self")}
```

輸出：

```
Get-SSMPatchBaseline -Filter $filter1
```

範例 4：使用 PowerShell 版本 2 時，必須使用 New-Object 來建立每個標籤。

```
$filter1 = New-Object
 Amazon.SimpleSystemsManagement.Model.PatchOrchestratorFilter
$filter1.Key = "OWNER"
$filter1.Values = "AWS"

Get-SSMPatchBaseline -Filter $filter1
```

輸出：

```
BaselineDescription                     BaselineId
                            BaselineName             DefaultBaselin

                                                     e
------------------                      ----------
                            ------------             --------------
Default Patch Baseline Provided by AWS. arn:aws:ssm:us-
west-2:123456789012:patchbaseline/pb-04fb4ae6142167966 AWS-DefaultPatchBaseline
 True
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 DescribePatchBaselines。

PowerShell V5 的工具

範例 1：此範例列出了所有修補基準。

```
Get-SSMPatchBaseline
```

輸出：

```
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...
```

範例 2：此範例列出 提供的所有修補程式基準 AWS。此範例使用的語法需要 PowerShell 版本 3 或更新版本。

```
$filter1 = @{Key="OWNER";Values=@("AWS")}
```

輸出：

```
Get-SSMPatchBaseline -Filter $filter1
```

範例 3：此範例列出了您作為擁有者的所有修補基準。此範例使用的語法需要 PowerShell 版本 3 或更新版本。

```
$filter1 = @{Key="OWNER";Values=@("Self")}
```

輸出：

```
Get-SSMPatchBaseline -Filter $filter1
```

範例 4：使用 PowerShell 版本 2 時，必須使用 New-Object 來建立每個標籤。

```
$filter1 = New-Object
 Amazon.SimpleSystemsManagement.Model.PatchOrchestratorFilter
$filter1.Key = "OWNER"
$filter1.Values = "AWS"
```

```
Get-SSMPatchBaseline -Filter $filter1
```

輸出：

```
BaselineDescription                     BaselineId
                            BaselineName            DefaultBaselin

                                                    e
------------------                      ----------
                            ------------            --------------
Default Patch Baseline Provided by AWS. arn:aws:ssm:us-
west-2:123456789012:patchbaseline/pb-04fb4ae6142167966 AWS-DefaultPatchBaseline
 True
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 DescribePatchBaselines。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribePatchGroupState** 與 CLI

下列程式碼範例示範如何使用 DescribePatchGroupState。

CLI

AWS CLI

取得修補程式群組的狀態

下列 describe-patch-group-state 範例示範擷取修補程式群組的高階修補程式合規摘要。

```
aws ssm describe-patch-group-state \
    --patch-group "Production"
```

輸出：

```
{
    "Instances": 21,
    "InstancesWithCriticalNonCompliantPatches": 1,
    "InstancesWithFailedPatches": 2,
```

```
    "InstancesWithInstalledOtherPatches": 3,
    "InstancesWithInstalledPatches": 21,
    "InstancesWithInstalledPendingRebootPatches": 2,
    "InstancesWithInstalledRejectedPatches": 1,
    "InstancesWithMissingPatches": 3,
    "InstancesWithNotApplicablePatches": 4,
    "InstancesWithOtherNonCompliantPatches": 1,
    "InstancesWithSecurityNonCompliantPatches": 1,
    "InstancesWithUnreportedNotApplicablePatches": 2
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 About patch groups <https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-patchgroups.html>__ 和 Understanding patch compliance state values。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribePatchGroupState。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範取得修補程式群組的高階修補程式合規摘要。

```
Get-SSMPatchGroupState -PatchGroup "Production"
```

輸出：

```
Instances                          : 4
InstancesWithFailedPatches         : 1
InstancesWithInstalledOtherPatches : 4
InstancesWithInstalledPatches      : 3
InstancesWithMissingPatches        : 0
InstancesWithNotApplicablePatches  : 0
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 DescribePatchGroupState。

PowerShell V5 的工具

範例 1：此範例示範取得修補程式群組的高階修補程式合規摘要。

```
Get-SSMPatchGroupState -PatchGroup "Production"
```

輸出：

```
Instances                          : 4
InstancesWithFailedPatches         : 1
InstancesWithInstalledOtherPatches : 4
InstancesWithInstalledPatches      : 3
InstancesWithMissingPatches        : 0
InstancesWithNotApplicablePatches  : 0
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 DescribePatchGroupState。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribePatchGroups** 與 CLI

下列程式碼範例示範如何使用 DescribePatchGroups。

CLI

AWS CLI

顯示修補程式群組註冊

下列 describe-patch-groups 範例列出了修補程式群組註冊。

```
aws ssm describe-patch-groups
```

輸出：

```
{
    "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
            }
        },
        ...
    ]
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Create a Patch Group <https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-group-tagging.html>__ 和 Add a Patch Group to a Patch Baseline。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 DescribePatchGroups。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了修補程式群組註冊。

```
Get-SSMPatchGroup
```

輸出：

```
BaselineIdentity                                                PatchGroup
----------------                                                ----------
Amazon.SimpleSystemsManagement.Model.PatchBaselineIdentity Production
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 DescribePatchGroups。

PowerShell V5 的工具

範例 1：此範例列出了修補程式群組註冊。

```
Get-SSMPatchGroup
```

輸出：

```
BaselineIdentity                                                PatchGroup
----------------                                                ----------
Amazon.SimpleSystemsManagement.Model.PatchBaselineIdentity Production
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 DescribePatchGroups。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetAutomationExecution** 與 CLI

下列程式碼範例示範如何使用 GetAutomationExecution。

CLI

AWS CLI

顯示有關自動化執行的詳細資訊

下列 get-automation-execution 範例顯示有關自動化執行的詳細資訊。

```
aws ssm get-automation-execution \
    --automation-execution-id 73c8eef8-f4ee-4a05-820c-e354fEXAMPLE
```

輸出：

```
{
    "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
        }
    }
}
```

如需詳細資訊，請參閱 AWS Systems Manager 使用者指南中的逐步解說：修補 Linux AMI (AWS CLI)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetAutomationExecution。

PowerShell

PowerShell V4 的工具

範例 1：此範例顯示自動化執行的詳細資訊。

```
Get-SSMAutomationExecution -AutomationExecutionId "4105a4fc-
f944-11e6-9d32-8fb2db27a909"
```

輸出：

```
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...}
```

範例 2：此範例列出了指定自動化執行 ID 的步驟詳細資訊

```
Get-SSMAutomationExecution -AutomationExecutionId e1d2bad3-4567-8901-
ae23-456c7c8901be | Select-Object -ExpandProperty StepExecutions | Select-Object
 StepName, Action, StepStatus, ValidNextSteps
```

輸出：

```
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    {}
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 GetAutomationExecution。

PowerShell V5 的工具

範例 1：此範例顯示自動化執行的詳細資訊。

```
Get-SSMAutomationExecution -AutomationExecutionId "4105a4fc-
f944-11e6-9d32-8fb2db27a909"
```

輸出：

```
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...}
```

範例 2：此範例列出了指定自動化執行 ID 的步驟詳細資訊

```
Get-SSMAutomationExecution -AutomationExecutionId e1d2bad3-4567-8901-
ae23-456c7c8901be | Select-Object -ExpandProperty StepExecutions | Select-Object
 StepName, Action, StepStatus, ValidNextSteps
```

輸出：

```
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    {}
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 GetAutomationExecution。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetCommandInvocation** 與 CLI

下列程式碼範例示範如何使用 GetCommandInvocation。

CLI

AWS CLI

顯示命令調用的詳細資訊

下列 get-command-invocation 範例列出了指定執行個體上指定命令的所有調用。

```
aws ssm get-command-invocation \
    --command-id "ef7fdfd8-9b57-4151-a15c-db9a12345678" \
    --instance-id "i-1234567890abcdef0"
```

輸出：

```
{
    "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
    }
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Understanding Command Statuses。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetCommandInvocation。

PowerShell

PowerShell V4 的工具

範例 1：此範例顯示執行個體上所執行之命令的詳細資訊。

```
Get-SSMCommandInvocationDetail -InstanceId "i-0cb2b964d3e14fd9f" -CommandId
 "b8eac879-0541-439d-94ec-47a80d554f44"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 GetCommandInvocation。

PowerShell V5 的工具

範例 1：此範例顯示執行個體上所執行之命令的詳細資訊。

```
Get-SSMCommandInvocationDetail -InstanceId "i-0cb2b964d3e14fd9f" -CommandId
 "b8eac879-0541-439d-94ec-47a80d554f44"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 GetCommandInvocation。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetConnectionStatus** 與 CLI

下列程式碼範例示範如何使用 GetConnectionStatus。

CLI

AWS CLI

顯示受管執行個體的連線狀態

此 get-connection-status 範例傳回指定受管執行個體的連線狀態。

```
aws ssm get-connection-status \
    --target i-1234567890abcdef0
```

輸出：

```
{
    "Target": "i-1234567890abcdef0",
    "Status": "connected"
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetConnectionStatus。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範擷取執行個體的 Session Manager 連線狀態，以判斷執行個體是否已連線並準備好接收 Session Manager 連線。

```
Get-SSMConnectionStatus -Target i-0a1caf234f12d3dc4
```

輸出：

```
Status    Target
------    ------
Connected i-0a1caf234f12d3dc4
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 GetConnectionStatus。

PowerShell V5 的工具

範例 1：此範例示範擷取執行個體的 Session Manager 連線狀態，以判斷執行個體是否已連線並準備好接收 Session Manager 連線。

```
Get-SSMConnectionStatus -Target i-0a1caf234f12d3dc4
```

輸出：

```
Status    Target
------    ------
Connected i-0a1caf234f12d3dc4
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 GetConnectionStatus。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetDefaultPatchBaseline** 與 CLI

下列程式碼範例示範如何使用 GetDefaultPatchBaseline。

CLI

AWS CLI

範例 1：顯示 Windows 預設修補基準

下列 get-default-patch-baseline 範例示範擷取 Windows Server 預設修補基準的詳細資訊。

```
aws ssm get-default-patch-baseline
```

輸出：

```
{
  "BaselineId": "pb-0713accee01612345",
  "OperatingSystem": "WINDOWS"
}
```

範例 2：顯示 Amazon Linux 預設修補基準

下列 `get-default-patch-baseline` 範例示範擷取 Amazon Linux 預設修補基準的詳細資訊。

```
aws ssm get-default-patch-baseline \
    --operating-system AMAZON_LINUX
```

輸出：

```
{
    "BaselineId": "pb-047c6eb9c8fc12345",
    "OperatingSystem": "AMAZON_LINUX"
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 About Predefined and Custom Patch Baselines <https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-baselines.html>__ 和 Set an Existing Patch Baseline as the Default。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetDefaultPatchBaseline。

PowerShell

PowerShell V4 的工具

範例 1：此範例顯示預設修補基準。

```
Get-SSMDefaultPatchBaseline
```

輸出：

```
arn:aws:ssm:us-west-2:123456789012:patchbaseline/pb-04fb4ae6142167966
```

- 如需 API 詳細資訊，請參閱《 AWS Tools for PowerShell Cmdlet Reference (V4)》中的 GetDefaultPatchBaseline。

PowerShell V5 的工具

範例 1：此範例顯示預設修補基準。

```
Get-SSMDefaultPatchBaseline
```

輸出：

```
arn:aws:ssm:us-west-2:123456789012:patchbaseline/pb-04fb4ae6142167966
```

- 如需 API 詳細資訊，請參閱《 AWS Tools for PowerShell Cmdlet Reference (V5)》中的 GetDefaultPatchBaseline。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetDeployablePatchSnapshotForInstance** 與 CLI

下列程式碼範例示範如何使用 GetDeployablePatchSnapshotForInstance。

CLI

AWS CLI

擷取執行個體使用的修補基準的目前快照

下列 get-deployable-patch-snapshot-for-instance 範例示範擷取執行個體所用之指定修補基準的目前快照詳細資訊。此命令必須使用執行個體憑證從執行個體執行。若要確保它使用執行個體憑證，請執行 aws configure，並僅指定執行個體的區域。將 Access Key 和 Secret Key 欄位留空。

秘訣：使用 uuidgen 來產生 snapshot-id。

```
aws ssm get-deployable-patch-snapshot-for-instance \
    --instance-id "i-1234567890abcdef0" \
    --snapshot-id "521c3536-930c-4aa9-950e-01234567abcd"
```

輸出：

```
{
    "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"
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Parameter name: Snapshot ID。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetDeployablePatchSnapshotForInstance。

PowerShell

PowerShell V4 的工具

範例 1：此範例顯示執行個體所用之修補基準的目前快照。此命令必須使用執行個體憑證從執行個體執行。為確保它使用執行個體憑證，範例會將 **Amazon.Runtime.InstanceProfileAWSCredentials** 物件傳遞至憑證參數。

```
$credentials = [Amazon.Runtime.InstanceProfileAWSCredentials]::new()
Get-SSMDeployablePatchSnapshotForInstance -SnapshotId "4681775b-098f-4435-
a956-0ef33373ac11" -InstanceId "i-0cb2b964d3e14fd9f" -Credentials $credentials
```

輸出：

```
InstanceId          SnapshotDownloadUrl
----------          -------------------
i-0cb2b964d3e14fd9f https://patch-baseline-snapshot-us-west-2.s3-us-
west-2.amazonaws.com/853d0d3db0f0cafe...1692/4681775b-098f-4435...
```

範例 2：此範例顯示如何取得完整的 SnapshotDownloadUrl。此命令必須使用執行個體憑證從執行個體執行。為確保它使用執行個體憑證，此範例會將 PowerShell 工作階段設定為使用 **Amazon.Runtime.InstanceProfileAWSCredentials** 物件。

```
Set-AWSCredential -Credential
 ([Amazon.Runtime.InstanceProfileAWSCredentials]::new())
(Get-SSMDeployablePatchSnapshotForInstance -SnapshotId "4681775b-098f-4435-
a956-0ef33373ac11" -InstanceId "i-0cb2b964d3e14fd9f").SnapshotDownloadUrl
```

輸出：

```
https://patch-baseline-snapshot-us-west-2.s3-us-
west-2.amazonaws.com/853d0d3db0f0cafe...
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 GetDeployablePatchSnapshotForInstance。

PowerShell V5 的工具

範例 1：此範例顯示執行個體所用之修補基準的目前快照。此命令必須使用執行個體憑證從執行個體執行。為確保它使用執行個體憑證，範例會將 **Amazon.Runtime.InstanceProfileAWSCredentials** 物件傳遞至憑證參數。

```
$credentials = [Amazon.Runtime.InstanceProfileAWSCredentials]::new()
Get-SSMDeployablePatchSnapshotForInstance -SnapshotId "4681775b-098f-4435-
a956-0ef33373ac11" -InstanceId "i-0cb2b964d3e14fd9f" -Credentials $credentials
```

輸出：

```
InstanceId          SnapshotDownloadUrl
----------          -------------------
i-0cb2b964d3e14fd9f https://patch-baseline-snapshot-us-west-2.s3-us-
west-2.amazonaws.com/853d0d3db0f0cafe...1692/4681775b-098f-4435...
```

範例 2：此範例顯示如何取得完整的 SnapshotDownloadUrl。此命令必須使用執行個體憑證從執行個體執行。為確保它使用執行個體憑證，此範例會將 PowerShell 工作階段設定為使用 **Amazon.Runtime.InstanceProfileAWSCredentials** 物件。

```
Set-AWSCredential -Credential
 ([Amazon.Runtime.InstanceProfileAWSCredentials]::new())
```

```
(Get-SSMDeployablePatchSnapshotForInstance -SnapshotId "4681775b-098f-4435-
a956-0ef33373ac11" -InstanceId "i-0cb2b964d3e14fd9f").SnapshotDownloadUrl
```

輸出：

```
https://patch-baseline-snapshot-us-west-2.s3-us-
west-2.amazonaws.com/853d0d3db0f0cafe...
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 GetDeployablePatchSnapshotForInstance。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetDocument** 與 CLI

下列程式碼範例示範如何使用 GetDocument。

CLI

AWS CLI

取得文件內容

下列 get-document 範例顯示 Systems Manager 文件的內容。

```
aws ssm get-document \
    --name "AWS-RunShellScript"
```

輸出：

```
{
    "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"
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 AWS Systems Manager Documents。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetDocument。

PowerShell

PowerShell V4 的工具

範例 1：此範例傳回文件的內容。

```
Get-SSMDocument -Name "RunShellScript"
```

輸出：

```
Content
-------
{...
```

範例 2：此範例顯示文件的完整內容。

```
(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 }}"
         }
      }
   ]
}
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V4) 中的 GetDocument。

PowerShell V5 的工具

範例 1：此範例傳回文件的內容。

```
Get-SSMDocument -Name "RunShellScript"
```

輸出：

```
Content
-------
{...
```

範例 2：此範例顯示文件的完整內容。

```
(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 }}"
         }
      }
   ]
}
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 GetDocument。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetInventory** 與 CLI

下列程式碼範例示範如何使用 `GetInventory`。

CLI

AWS CLI

檢視庫存

此範例示範取得庫存的自訂中繼資料。

命令：

```
aws ssm get-inventory
```

輸出：

```
{
  "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"
      }
  ]
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetInventory。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範取得庫存的自訂中繼資料。

```
Get-SSMInventory
```

輸出：

```
Data
     Id
----
     --
{[AWS:InstanceInformation,
 Amazon.SimpleSystemsManagement.Model.InventoryResultItem]} i-0cb2b964d3e14fd9f
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 GetInventory。

PowerShell V5 的工具

範例 1：此範例示範取得庫存的自訂中繼資料。

```
Get-SSMInventory
```

輸出：

```
Data
     Id
----
     --
{[AWS:InstanceInformation,
 Amazon.SimpleSystemsManagement.Model.InventoryResultItem]} i-0cb2b964d3e14fd9f
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 GetInventory。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetInventorySchema** 與 CLI

下列程式碼範例示範如何使用 GetInventorySchema。

CLI

AWS CLI

檢視庫存結構描述

此範例傳回帳戶的庫存類型名稱清單。

命令：

```
aws ssm get-inventory-schema
```

輸出：

```
{
  "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--"
}
```

檢視特定庫存類型的庫存結構描述

此範例會傳回 AWS：AWS Component 清查類型的清查結構描述。

命令：

```
aws ssm get-inventory-schema --type-name "AWS:AWSComponent"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetInventorySchema。

PowerShell

PowerShell V4 的工具

範例 1：此範例傳回帳戶的庫存類型名稱清單。

```
Get-SSMInventorySchema
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 GetInventorySchema。

PowerShell V5 的工具

範例 1：此範例傳回帳戶的庫存類型名稱清單。

```
Get-SSMInventorySchema
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 GetInventorySchema。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetMaintenanceWindow** 與 CLI

下列程式碼範例示範如何使用 GetMaintenanceWindow。

CLI

AWS CLI

取得有關維護時段的資訊

下列 get-maintenance-window 範例示範擷取有關指定維護時段的詳細資訊。

```
aws ssm get-maintenance-window \
    --window-id "mw-03eb9db428EXAMPLE"
```

輸出：

```
{
    "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"
}
```

如需詳細資訊，請參閱 AWS Systems Manager 使用者指南中的檢視維護時段 (AWS CLI) 的相關資訊。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetMaintenanceWindow。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範取得有關維護時段的詳細資訊。

```
Get-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 GetMaintenanceWindow。

PowerShell V5 的工具

範例 1：此範例示範取得有關維護時段的詳細資訊。

```
Get-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 GetMaintenanceWindow。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetMaintenanceWindowExecution** 與 CLI

下列程式碼範例示範如何使用 GetMaintenanceWindowExecution。

CLI

AWS CLI

取得維護時段任務執行的相關資訊

下列 get-maintenance-window-execution 範例列出了有關作為指定維護時段執行的一部分執行之任務的資訊。

```
aws ssm get-maintenance-window-execution \
    --window-execution-id "518d5565-5969-4cca-8f0e-da3b2EXAMPLE"
```

輸出：

```
{
    "Status": "SUCCESS",
    "TaskIds": [
        "ac0c6ae1-daa3-4a89-832e-d3845EXAMPLE"
    ],
    "StartTime": 1487692834.595,
    "EndTime": 1487692835.051,
    "WindowExecutionId": "518d5565-5969-4cca-8f0e-da3b2EXAMPLE",
}
```

如需詳細資訊，請參閱 AWS Systems Manager 使用者指南中的檢視任務和任務執行 (AWS CLI) 的相關資訊。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetMaintenanceWindowExecution。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了作為維護時段執行的一部分執行之任務的資訊。

```
Get-SSMMaintenanceWindowExecution -WindowExecutionId "518d5565-5969-4cca-8f0e-
da3b2a638355"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 GetMaintenanceWindowExecution。

PowerShell V5 的工具

範例 1：此範例列出了作為維護時段執行的一部分執行之任務的資訊。

```
Get-SSMMaintenanceWindowExecution -WindowExecutionId "518d5565-5969-4cca-8f0e-
da3b2a638355"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 GetMaintenanceWindowExecution。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetMaintenanceWindowExecutionTask** 與 CLI

下列程式碼範例示範如何使用 GetMaintenanceWindowExecutionTask。

CLI

AWS CLI

取得維護時段任務執行的相關資訊

下列 get-maintenance-window-execution-task 範例列出了有關屬於指定維護時段執行之任務的資訊。

```
aws ssm get-maintenance-window-execution-task \
    --window-execution-id "518d5565-5969-4cca-8f0e-da3b2EXAMPLE" \
    --task-id "ac0c6ae1-daa3-4a89-832e-d3845EXAMPLE"
```

輸出：

```
{
    "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"
}
```

如需詳細資訊，請參閱 AWS Systems Manager 使用者指南中的檢視任務和任務執行 (AWS CLI) 的相關資訊。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetMaintenanceWindowExecutionTask。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了有關屬於維護時段執行之任務的資訊。

```
Get-SSMMaintenanceWindowExecutionTask -TaskId "ac0c6ae1-daa3-4a89-832e-
d384503b6586" -WindowExecutionId "518d5565-5969-4cca-8f0e-da3b2a638355"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V4) 中的 GetMaintenanceWindowExecutionTask。

PowerShell V5 的工具

範例 1：此範例列出了有關屬於維護時段執行之任務的資訊。

```
Get-SSMMaintenanceWindowExecutionTask -TaskId "ac0c6ae1-daa3-4a89-832e-
d384503b6586" -WindowExecutionId "518d5565-5969-4cca-8f0e-da3b2a638355"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱《 AWS Tools for PowerShell Cmdlet Reference (V5)》中的 GetMaintenanceWindowExecutionTask。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`GetParameter` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `GetParameter`。

CLI

AWS CLI

範例 1：顯示參數的值

下列 `get-parameter` 範例列出了指定單一參數的值。

```
aws ssm get-parameter \
    --name "MyStringParameter"
```

輸出：

```
{
    "Parameter": {
        "Name": "MyStringParameter",
        "Type": "String",
        "Value": "Veni",
        "Version": 1,
        "LastModifiedDate": 1530018761.888,
        "ARN": "arn:aws:ssm:us-east-2:111222333444:parameter/MyStringParameter"
        "DataType": "text"
    }
```

```
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Working with Parameter Store。

範例 2：解密 SecureString 參數的值

下列 get-parameter 範例示範解密指定 SecureString 參數的值。

```
aws ssm get-parameter \
    --name "MySecureStringParameter" \
    --with-decryption
```

輸出：

```
{
    "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"
    }
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Working with Parameter Store。

範例 3：使用標籤顯示參數的值

下列 get-parameter 範例列出具有指定標籤之指定單一參數的值。

```
aws ssm get-parameter \
    --name "MyParameter:label"
```

輸出：

```
{
```

```
    "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"
    }
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Working with parameter labels。

範例 4：以版本顯示參數的值

下列 get-parameter 範例列出了指定單一參數版本的值。

```
aws ssm get-parameter \
    --name "MyParameter:2"
```

輸出：

```
{
    "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"
    }
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Working with parameter labels。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetParameter。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    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)
    }
```

- 如需 API 詳細資訊，請參閱《適用於 Rust 的AWS SDK API 參考》中的 GetParameter。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetParameterHistory** 與 CLI

下列程式碼範例示範如何使用 GetParameterHistory。

CLI

AWS CLI

取得參數的值歷史記錄

下列 get-parameter-history 範例列出了指定參數的變更歷史記錄，包括其值。

```
aws ssm get-parameter-history \
    --name "MyStringParameter"
```

輸出：

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

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Working with parameter versions。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetParameterHistory。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了參數的值歷史記錄。

```
Get-SSMParameterHistory -Name "Welcome"
```

輸出：

```
Description      :
KeyId            :
LastModifiedDate : 3/3/2017 6:55:25 PM
LastModifiedUser : arn:aws:iam::123456789012:user/admin
Name             : Welcome
Type             : String
Value            : helloWorld
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 GetParameterHistory。

PowerShell V5 的工具

範例 1：此範例列出了參數的值歷史記錄。

```
Get-SSMParameterHistory -Name "Welcome"
```

輸出：

```
Description      :
KeyId            :
LastModifiedDate : 3/3/2017 6:55:25 PM
LastModifiedUser : arn:aws:iam::123456789012:user/admin
Name             : Welcome
Type             : String
Value            : helloWorld
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 GetParameterHistory。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetParameters** 與 CLI

下列程式碼範例示範如何使用 GetParameters。

CLI

AWS CLI

範例 1：列出參數的值

下列 get-parameters 範例列出了三個指定參數的值。

```
aws ssm get-parameters \
    --names "MyStringParameter" "MyStringListParameter" "MyInvalidParameterName"
```

輸出：

```
{
    "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"
    ]
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Working with Parameter Store。

範例 2：使用 ``--query`` 選項列出多個參數的名稱和值

下列 `get-parameters` 範例列出了指定參數的名稱和值。

```
aws ssm get-parameters \
    --names MyStringParameter MyStringListParameter \
    --query "Parameters[*].{Name:Name,Value:Value}"
```

輸出：

```
[
    {
        "Name": "MyStringListParameter",
        "Value": "alpha,beta,gamma"
    },
    {
        "Name": "MyStringParameter",
        "Value": "Vidi"
    }
```

```
]
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Working with Parameter Store。

範例 3：使用標籤顯示參數的值

下列 get-parameter 範例列出具有指定標籤之指定單一參數的值。

```
aws ssm get-parameter \
    --name "MyParameter:label"
```

輸出：

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

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Working with parameter labels。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetParameters。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了參數的多個值。

```
Get-SSMParameterValue -Name "Welcome"
```

輸出：

```
InvalidParameters Parameters
----------------- ----------
{}                {Welcome}
```

範例 2：此範例列出了值的詳細資訊。

```
(Get-SSMParameterValue -Name "Welcome").Parameters
```

輸出：

```
Name    Type   Value
----    ----   -----
Welcome String Good day, Sunshine!
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 GetParameters。

PowerShell V5 的工具

範例 1：此範例列出了參數的多個值。

```
Get-SSMParameterValue -Name "Welcome"
```

輸出：

```
InvalidParameters Parameters
----------------- ----------
{}                {Welcome}
```

範例 2：此範例列出了值的詳細資訊。

```
(Get-SSMParameterValue -Name "Welcome").Parameters
```

輸出：

```
Name    Type   Value
----    ----   -----
Welcome String Good day, Sunshine!
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 GetParameters。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetPatchBaseline** 與 CLI

下列程式碼範例示範如何使用 GetPatchBaseline。

CLI

AWS CLI

顯示修補基準

下列 get-patch-baseline 範例示範擷取指定修補基準的詳細資訊。

```
aws ssm get-patch-baseline \
    --baseline-id "pb-0123456789abcdef0"
```

輸出：

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

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 About Patch Baselines。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetPatchBaseline。

PowerShell

PowerShell V4 的工具

範例 1：此範例顯示修補基準的詳細資訊。

```
Get-SSMPatchBaselineDetail -BaselineId "pb-03da896ca3b68b639"
```

輸出：

```
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 : {}
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 GetPatchBaseline。

PowerShell V5 的工具

範例 1：此範例顯示修補基準的詳細資訊。

```
Get-SSMPatchBaselineDetail -BaselineId "pb-03da896ca3b68b639"
```

輸出：

```
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 : {}
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 GetPatchBaseline。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **GetPatchBaselineForPatchGroup** 與 CLI

下列程式碼範例示範如何使用 GetPatchBaselineForPatchGroup。

CLI

AWS CLI

顯示修補程式群組的修補基準

下列 get-patch-baseline-for-patch-group 範例示範擷取有關指定修補程式群組之修補基準的詳細資訊。

```
aws ssm get-patch-baseline-for-patch-group \
    --patch-group "DEV"
```

輸出：

```
{
    "PatchGroup": "DEV",
    "BaselineId": "pb-0123456789abcdef0",
    "OperatingSystem": "WINDOWS"
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Create a Patch Group <https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-group-tagging.html>__ 和 Add a Patch Group to a Patch Baseline。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 GetPatchBaselineForPatchGroup。

PowerShell

PowerShell V4 的工具

範例 1：此範例顯示修補程式群組的修補基準。

```
Get-SSMPatchBaselineForPatchGroup -PatchGroup "Production"
```

輸出：

```
BaselineId           PatchGroup
----------           ----------
pb-045f10b4f382baeda Production
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 GetPatchBaselineForPatchGroup。

PowerShell V5 的工具

範例 1：此範例顯示修補程式群組的修補基準。

```
Get-SSMPatchBaselineForPatchGroup -PatchGroup "Production"
```

輸出：

```
BaselineId           PatchGroup
----------           ----------
pb-045f10b4f382baeda Production
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 GetPatchBaselineForPatchGroup。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListAssociationVersions** 與 CLI

下列程式碼範例示範如何使用 `ListAssociationVersions`。

CLI

AWS CLI

列出特定關聯 ID 之關聯的所有版本

下列 `list-association-versions` 範例列出了指定關聯的所有版本。

```
aws ssm list-association-versions \
```

```
    --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

輸出：

```
{
"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"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Working with associations in Systems Manager。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListAssociationVersions。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範擷取所提供之關聯的所有版本。

```
Get-SSMAssociationVersionList -AssociationId 123a45a0-c678-9012-3456-78901234db5e
```

輸出：

```
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}
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 ListAssociationVersions。

PowerShell V5 的工具

範例 1：此範例示範擷取所提供之關聯的所有版本。

```
Get-SSMAssociationVersionList -AssociationId 123a45a0-c678-9012-3456-78901234db5e
```

輸出：

```
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}
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 ListAssociationVersions。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListAssociations** 與 CLI

下列程式碼範例示範如何使用 `ListAssociations`。

CLI

AWS CLI

範例 1：列出特定執行個體的關聯

下列 list-associations 範例列出了具有 AssociationName、UpdateSSMAgent 的所有關聯。

```
aws ssm list-associations /
    --association-filter-list "key=AssociationName,value=UpdateSSMAgent"
```

輸出：

```
{
    "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"
        }
    ]
}
```

如需詳細資訊，請參閱《Systems Manager 使用者指南》中的 Working with associations in Systems Manager。

範例 2：列出特定文件的關聯

下列 list-associations 範例列出了指定文件的所有關聯。

```
aws ssm list-associations /
    --association-filter-list "key=Name,value=AWS-UpdateSSMAgent"
```

輸出：

```
{
    "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
                }
            }
        }
    ]
}
```

如需詳細資訊，請參閱《Systems Manager 使用者指南》中的 Working with associations in Systems Manager。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListAssociations。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了執行個體的所有關聯。此範例使用的語法需要 PowerShell 版本 3 或更新版本。

```
$filter1 = @{Key="InstanceId";Value=@("i-0000293ffd8c57862")}
Get-SSMAssociationList -AssociationFilterList $filter1
```

輸出：

```
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}
```

範例 2：此範例列出了組態文件的所有關聯。此範例使用的語法需要 PowerShell 版本 3 或更新版本。

```
$filter2 = @{Key="Name";Value=@("AWS-UpdateSSMAgent")}
Get-SSMAssociationList -AssociationFilterList $filter2
```

輸出：

```
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}
```

範例 3：使用 PowerShell 版本 2 時，必須使用 New-Object 來建立每個篩選條件。

```
$filter1 = New-Object Amazon.SimpleSystemsManagement.Model.AssociationFilter
$filter1.Key = "InstanceId"
$filter1.Value = "i-0000293ffd8c57862"

Get-SSMAssociationList -AssociationFilterList $filter1
```

輸出：

```
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}
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 ListAssociations。

PowerShell V5 的工具

範例 1：此範例列出了執行個體的所有關聯。此範例使用的語法需要 PowerShell 版本 3 或更新版本。

```
$filter1 = @{Key="InstanceId";Value=@("i-0000293ffd8c57862")}
Get-SSMAssociationList -AssociationFilterList $filter1
```

輸出：

```
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}
```

範例 2：此範例列出了組態文件的所有關聯。此範例使用的語法需要 PowerShell 版本 3 或更新版本。

```
$filter2 = @{Key="Name";Value=@("AWS-UpdateSSMAgent")}
Get-SSMAssociationList -AssociationFilterList $filter2
```

輸出：

```
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}
```

範例 3：使用 PowerShell 版本 2 時，必須使用 New-Object 來建立每個篩選條件。

```
$filter1 = New-Object Amazon.SimpleSystemsManagement.Model.AssociationFilter
$filter1.Key = "InstanceId"
$filter1.Value = "i-0000293ffd8c57862"
```

```
Get-SSMAssociationList -AssociationFilterList $filter1
```

輸出：

```
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}
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 ListAssociations。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`ListCommandInvocations` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `ListCommandInvocations`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 了解基本概念

CLI

AWS CLI

列出特定命令的調用

下列 `list-command-invocations` 範例列出了命令的所有調用。

```
aws ssm list-command-invocations \
    --command-id "ef7fdfd8-9b57-4151-a15c-db9a12345678" \
    --details
```

輸出：

```
{
    "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
            }
        }
    ]
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Understanding Command Statuses。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListCommandInvocations。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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;
  }
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 ListCommandInvocations。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了命令的所有調用。

```
Get-SSMCommandInvocation -CommandId "b8eac879-0541-439d-94ec-47a80d554f44" -
Detail $true
```

輸出：

```
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        :
```

範例 2：此範例列出了用於調用命令 ID 為 e1eb2e3c-ed4c-5123-45c1-234f5612345f 的 CommandPlugins

```
Get-SSMCommandInvocation -CommandId e1eb2e3c-ed4c-5123-45c1-234f5612345f -Detail:
$true | Select-Object -ExpandProperty CommandPlugins
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 ListCommandInvocations。

PowerShell V5 的工具

範例 1：此範例列出了命令的所有調用。

```
Get-SSMCommandInvocation -CommandId "b8eac879-0541-439d-94ec-47a80d554f44" -
Detail $true
```

輸出：

```
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        :
```

範例 2：此範例列出了用於調用命令 ID 為 e1eb2e3c-ed4c-5123-45c1-234f5612345f 的 CommandPlugins

```
Get-SSMCommandInvocation -CommandId e1eb2e3c-ed4c-5123-45c1-234f5612345f -Detail:
$true | Select-Object -ExpandProperty CommandPlugins
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 ListCommandInvocations。

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) 的 API 參考》中的 ListCommandInvocations。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListCommands** 與 CLI

下列程式碼範例示範如何使用 ListCommands。

CLI

AWS CLI

範例 1：取得特定命令的狀態

下列 list-commands 範例示範擷取並顯示指定命令的狀態。

```
aws ssm list-commands \
    --command-id "0831e1a8-a1ac-4257-a1fd-c831bEXAMPLE"
```

範例 2：取得在特定日期後請求之命令的狀態

下列 list-commands 範例示範擷取在指定日期後請求的命令詳細資訊。

```
aws ssm list-commands \
    --filter "key=InvokedAfter,value=2020-02-01T00:00:00Z"
```

範例 3：列出 AWS 帳戶中請求的所有命令

下列list-commands範例列出目前 AWS 帳戶和區域中使用者請求的所有命令。

```
aws ssm list-commands
```

輸出：

```
{
    "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
            }
        }
    ]
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Running Commands Using Systems Manager Run Command。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListCommands。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了所請求的所有命令。

```
Get-SSMCommand
```

輸出：

```
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            : {}
```

範例 2：此範例示範取得特定命令的狀態。

```
Get-SSMCommand -CommandId "4b75a163-d39a-4d97-87c9-98ae52c6be35"
```

範例 3：此範例示範擷取在 2019-04-01T00:00:00Z 之後調用的所有 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
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 ListCommands。

PowerShell V5 的工具

範例 1：此範例列出了所請求的所有命令。

```
Get-SSMCommand
```

輸出：

```
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            : {}
```

範例 2：此範例示範取得特定命令的狀態。

```
Get-SSMCommand -CommandId "4b75a163-d39a-4d97-87c9-98ae52c6be35"
```

範例 3：此範例示範擷取在 2019-04-01T00:00:00Z 之後調用的所有 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
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 ListCommands。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListComplianceItems** 與 CLI

下列程式碼範例示範如何使用 ListComplianceItems。

CLI

AWS CLI

列出特定執行個體的合規項目

此範例列出了指定執行個體的所有合規項目。

命令：

```
aws ssm list-compliance-items --resource-ids "i-1234567890abcdef0" --resource-types "ManagedInstance"
```

輸出：

```
{
  "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--"
}
```

列出特定執行個體和關聯 ID 的合規項目

此範例列出了指定執行個體和關聯 ID 的所有合規項目。

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"
```

列出執行個體在特定日期和時間之後的合規項目

此範例列出了執行個體在指定日期和時間之後的所有合規項目。

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"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListComplianceItems。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了指定資源 ID 和類型的合規項目清單，篩選合規類型為「關聯」

```
Get-SSMComplianceItemList -ResourceId i-1a2caf345f67d0dc2 -ResourceType
 ManagedInstance -Filter @{Key="ComplianceType";Values="Association"}
```

輸出：

```
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            :
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 ListComplianceItems。

PowerShell V5 的工具

範例 1：此範例列出了指定資源 ID 和類型的合規項目清單，篩選合規類型為「關聯」

```
Get-SSMComplianceItemList -ResourceId i-1a2caf345f67d0dc2 -ResourceType
 ManagedInstance -Filter @{Key="ComplianceType";Values="Association"}
```

輸出：

```
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            :
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 ListComplianceItems。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListComplianceSummaries** 與 CLI

下列程式碼範例示範如何使用 `ListComplianceSummaries`。

CLI

AWS CLI

列出所有合規類型的合規摘要

此範例列出了帳戶中所有合規類型的合規摘要。

命令：

```
aws ssm list-compliance-summaries
```

輸出：

```
{
  "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=="
}
```

列出特定合規類型的合規摘要

此範例列出了「修補程式」合規類型的合規摘要。

命令：

```
aws ssm list-compliance-summaries --
filters "Key=ComplianceType,Values=Patch,Type=EQUAL"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListComplianceSummaries。

PowerShell

PowerShell V4 的工具

範例 1：此範例傳回所有合規類型的合規與不合規資源計數摘要。

```
Get-SSMComplianceSummaryList
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 ListComplianceSummaries。

PowerShell V5 的工具

範例 1：此範例傳回所有合規類型的合規與不合規資源計數摘要。

```
Get-SSMComplianceSummaryList
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 ListComplianceSummaries。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListDocumentVersions** 與 CLI

下列程式碼範例示範如何使用 `ListDocumentVersions`。

CLI

AWS CLI

列出文件版本

下列 `list-document-versions` 範例列出了 Systems Manager 文件的所有版本。

```
aws ssm list-document-versions \
    --name "Example"
```

輸出：

```
{
    "DocumentVersions": [
        {
```

```
            "Name": "Example",
            "DocumentVersion": "1",
            "CreatedDate": 1583257938.266,
            "IsDefaultVersion": true,
            "DocumentFormat": "YAML",
            "Status": "Active"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Sending Commands that Use the Document Version Parameter。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListDocumentVersion。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了文件的所有版本。

```
Get-SSMDocumentVersionList -Name "AWS-UpdateSSMAgent"
```

輸出：

```
CreatedDate       : 6/1/2021 5:19:10 PM
DocumentFormat    : JSON
DocumentVersion   : 1
IsDefaultVersion  : True
Name              : AWS-UpdateSSMAgent
Status            : Active
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 ListDocumentVersions。

PowerShell V5 的工具

範例 1：此範例列出了文件的所有版本。

```
Get-SSMDocumentVersionList -Name "AWS-UpdateSSMAgent"
```

輸出：

```
CreatedDate       : 6/1/2021 5:19:10 PM
DocumentFormat    : JSON
DocumentVersion   : 1
IsDefaultVersion  : True
Name              : AWS-UpdateSSMAgent
Status            : Active
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 ListDocumentVersions。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListDocuments** 與 CLI

下列程式碼範例示範如何使用 ListDocuments。

CLI

AWS CLI

範例 1：列出文件

下列 list-documents 範例列出了由請求帳戶擁有、以自訂標籤標記的文件。

```
aws ssm list-documents \
    --filters Key=Owner,Values=Self Key=tag:DocUse,Values=Testing
```

輸出：

```
{
    "DocumentIdentifiers": [
        {
            "Name": "Example",
            "Owner": "29884EXAMPLE",
            "PlatformTypes": [
                "Windows",
                "Linux"
            ],
            "DocumentVersion": "1",
```

```
            "DocumentType": "Automation",
            "SchemaVersion": "0.3",
            "DocumentFormat": "YAML",
            "Tags": [
                {
                    "Key": "DocUse",
                    "Value": "Testing"
                }
            ]
        }
    ]
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 AWS Systems Manager Documents。

範例 2：列出共用文件

下列list-documents範例列出共用文件，包括非 擁有的私有共用文件 AWS。

```
aws ssm list-documents \
    --filters Key=Name,Values=sharedDocNamePrefix  Key=Owner,Values=Private
```

輸出：

```
{
    "DocumentIdentifiers": [
        {
            "Name": "Example",
            "Owner": "12345EXAMPLE",
            "PlatformTypes": [
                "Windows",
                "Linux"
            ],
            "DocumentVersion": "1",
            "DocumentType": "Command",
            "SchemaVersion": "0.3",
            "DocumentFormat": "YAML",
            "Tags": []
        }
    ]
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 AWS Systems Manager Documents。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListDocuments。

PowerShell

PowerShell V4 的工具

範例 1：列出帳戶中的所有組態文件。

```
Get-SSMDocumentList
```

輸出：

```
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
...
```

範例 2：此範例示範擷取名稱符合 "Platform" 的所有自動化文件

```
Get-SSMDocumentList -DocumentFilterList @{Key="DocumentType";Value="Automation"}
 | Where-Object Name -Match "Platform"
```

輸出：

```
DocumentFormat  : JSON
DocumentType    : Automation
DocumentVersion : 7
Name            : KT-Get-Platform
Owner           : 987654123456
PlatformTypes   : {Windows, Linux}
SchemaVersion   : 0.3
Tags            : {}
TargetType      :
VersionName     :
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 ListDocuments。

PowerShell V5 的工具

範例 1：列出帳戶中的所有組態文件。

```
Get-SSMDocumentList
```

輸出：

```
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
...
```

範例 2：此範例示範擷取名稱符合 "Platform" 的所有自動化文件

```
Get-SSMDocumentList -DocumentFilterList @{Key="DocumentType";Value="Automation"}
 | Where-Object Name -Match "Platform"
```

輸出：

```
DocumentFormat  : JSON
DocumentType    : Automation
DocumentVersion : 7
Name            : KT-Get-Platform
Owner           : 987654123456
PlatformTypes   : {Windows, Linux}
SchemaVersion   : 0.3
Tags            : {}
TargetType      :
VersionName     :
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 ListDocuments。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListInventoryEntries** 與 CLI

下列程式碼範例示範如何使用 `ListInventoryEntries`。

CLI

AWS CLI

範例 1：檢視執行個體的特定庫存類型項目

下列`list-inventory-entries`範例列出特定執行個體上 AWS：應用程式庫存類型的庫存項目。

```
aws ssm list-inventory-entries \
```

```
    --instance-id "i-1234567890abcdef0" \
    --type-name "AWS:Application"
```

輸出：

```
{
  "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"
    }
  ]
}
```

範例 2：檢視指派給執行個體的自訂庫存項目

下列 `list-inventory-entries` 範例列出了指派給執行個體的自訂庫存項目。

```
aws ssm list-inventory-entries \
    --instance-id "i-1234567890abcdef0" \
    --type-name "Custom:RackInfo"
```

輸出：

```
{
  "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"
    }
  ]
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListInventoryEntries。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了執行個體的所有自訂庫存項目。

```
Get-SSMInventoryEntriesList -InstanceId "i-0cb2b964d3e14fd9f" -TypeName
 "Custom:RackInfo"
```

輸出：

```
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
```

範例 2：此範例列出了詳細資訊。

```
(Get-SSMInventoryEntriesList -InstanceId "i-0cb2b964d3e14fd9f" -TypeName
 "Custom:RackInfo").Entries
```

輸出：

```
Key          Value
---          -----
RackLocation Bay B/Row C/Rack D/Shelf E
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 ListInventoryEntries。

PowerShell V5 的工具

範例 1：此範例列出了執行個體的所有自訂庫存項目。

```
Get-SSMInventoryEntriesList -InstanceId "i-0cb2b964d3e14fd9f" -TypeName
 "Custom:RackInfo"
```

輸出：

```
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
```

範例 2：此範例列出了詳細資訊。

```
(Get-SSMInventoryEntriesList -InstanceId "i-0cb2b964d3e14fd9f" -TypeName
 "Custom:RackInfo").Entries
```

輸出：

```
Key          Value
---          -----
RackLocation Bay B/Row C/Rack D/Shelf E
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 ListInventoryEntries。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListResourceComplianceSummaries** 與 CLI

下列程式碼範例示範如何使用 ListResourceComplianceSummaries。

CLI

AWS CLI

列出資源層級合規摘要計數

此範例列出了資源層級合規摘要計數。

命令：

```
aws ssm list-resource-compliance-summaries
```

輸出：

```
{
  "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--"
}
```

列出特定合規類型的資源層級合規摘要

此範例列出了「修補程式」合規類型的資源層級合規摘要。

命令：

```
aws ssm list-resource-compliance-summaries --
filters "Key=ComplianceType,Values=Patch,Type=EQUAL"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListResourceComplianceSummaries。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範取得資源層級摘要計數。摘要包含有關合規和不合規狀態的資訊，以及符合 "Windows10" 之產品的詳細合規項目嚴重性計數。由於未不指定該參數時，MaxResult 預設為 100，且該值無效，因此新增 MaxResult 參數，並將該值設定為 50。

```
$FilterValues = @{
  "Key"="Product"
        "Type"="EQUAL"
        "Values"="Windows10"
}
        Get-SSMResourceComplianceSummaryList -Filter $FilterValues -MaxResult 50
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 ListResourceComplianceSummaries。

PowerShell V5 的工具

範例 1：此範例示範取得資源層級摘要計數。摘要包含有關合規和不合規狀態的資訊，以及符合 "Windows10" 之產品的詳細合規項目嚴重性計數。由於未不指定該參數時，MaxResult 預設為 100，且該值無效，因此新增 MaxResult 參數，並將該值設定為 50。

```
$FilterValues = @{
  "Key"="Product"
        "Type"="EQUAL"
        "Values"="Windows10"
}
        Get-SSMResourceComplianceSummaryList -Filter $FilterValues -MaxResult 50
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 ListResourceComplianceSummaries。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ListTagsForResource** 與 CLI

下列程式碼範例示範如何使用 `ListTagsForResource`。

CLI

AWS CLI

列出套用至修補基準的標籤

下列 `list-tags-for-resource` 範例列出了修補基準的標籤。

```
aws ssm list-tags-for-resource \
    --resource-type "PatchBaseline" \
    --resource-id "pb-0123456789abcdef0"
```

輸出：

```
{
    "TagList": [
        {
            "Key": "Environment",
            "Value": "Production"
        },
        {
            "Key": "Region",
            "Value": "EMEA"
        }
    ]
}
```

如需詳細資訊，請參閱《 AWS 一般參考》中的標記 AWS 資源。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ListTagsForResource。

PowerShell

PowerShell V4 的工具

範例 1：此範例列出了維護時段的標籤。

```
Get-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
 "MaintenanceWindow"
```

輸出：

```
Key   Value
---   -----
Stack Production
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 ListTagsForResource。

PowerShell V5 的工具

範例 1：此範例列出了維護時段的標籤。

```
Get-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
 "MaintenanceWindow"
```

輸出：

```
Key   Value
---   -----
Stack Production
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 ListTagsForResource。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **ModifyDocumentPermission** 與 CLI

下列程式碼範例示範如何使用 ModifyDocumentPermission。

CLI

AWS CLI

修改文件許可

下列 modify-document-permission 範例公開共用 Systems Manager 文件。

```
aws ssm modify-document-permission \
    --name "Example" \
    --permission-type "Share" \
    --account-ids-to-add "All"
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Share a Systems Manager Document。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 ModifyDocumentPermission。

PowerShell

PowerShell V4 的工具

範例 1：此範例將文件的「共用」許可新增至所有帳戶。如果命令成功，則無輸出訊息。

```
Edit-SSMDocumentPermission -Name "RunShellScript" -PermissionType "Share" -
AccountIdsToAdd all
```

範例 2：此範例將文件的「共用」許可新增至特定帳戶。如果命令成功，則無輸出訊息。

```
Edit-SSMDocumentPermission -Name "RunShellScriptNew" -PermissionType "Share" -
AccountIdsToAdd "123456789012"
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 ModifyDocumentPermission。

PowerShell V5 的工具

範例 1：此範例將文件的「共用」許可新增至所有帳戶。如果命令成功，則無輸出訊息。

```
Edit-SSMDocumentPermission -Name "RunShellScript" -PermissionType "Share" -
AccountIdsToAdd all
```

範例 2：此範例將文件的「共用」許可新增至特定帳戶。如果命令成功，則無輸出訊息。

```
Edit-SSMDocumentPermission -Name "RunShellScriptNew" -PermissionType "Share" -
AccountIdsToAdd "123456789012"
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 ModifyDocumentPermission。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **PutComplianceItems** 與 CLI

下列程式碼範例示範如何使用 PutComplianceItems。

CLI

AWS CLI

將合規類型和合規詳細資訊註冊至指定的執行個體

此範例示範將合規類型 Custom:AVCheck 註冊到指定的受管執行個體。如果命令成功，則無輸出訊息。

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"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 PutComplianceItems。

PowerShell

PowerShell V4 的工具

範例 1：此範例為指定的受管執行個體撰寫自訂合規項目

```
$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"
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 PutComplianceItems。

PowerShell V5 的工具

範例 1：此範例為指定的受管執行個體撰寫自訂合規項目

```
$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"
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 PutComplianceItems。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **PutInventory** 與 CLI

下列程式碼範例示範如何使用 PutInventory。

CLI

AWS CLI

向執行個體指派客戶中繼資料

此範例會將機架位置資訊指派給執行個體。如果命令成功，則無輸出訊息。

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'}]"
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 PutInventory。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範將機架位置資訊指派給執行個體。如果命令成功，則無輸出訊息。

```
$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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V4) 中的 PutInventory。

PowerShell V5 的工具

範例 1：此範例示範將機架位置資訊指派給執行個體。如果命令成功，則無輸出訊息。

```
$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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 PutInventory。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`PutParameter` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `PutParameter`。

CLI

AWS CLI

範例 1：變更參數值

下列 put-parameter 範例示範變更指定參數的值。

```
aws ssm put-parameter \
    --name "MyStringParameter" \
    --type "String" \
    --value "Vici" \
    --overwrite
```

輸出：

```
{
    "Version": 2,
    "Tier": "Standard"
}
```

如需詳細資訊，請參閱 Systems Manager 使用者指南中的建立 Systems Manager 參數 (AWS CLI)、管理參數層和使用參數政策。 AWS

範例 2：建立進階參數

下列 put-parameter 範例示範建立進階參數。

```
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
```

輸出：

```
{
    "Version": 1,
    "Tier": "Advanced"
}
```

如需詳細資訊，請參閱 Systems Manager 使用者指南中的建立 Systems Manager 參數 (AWS CLI)、管理參數層和使用參數政策。 AWS

範例 3：將標準參數轉換為進階參數

下列put-parameter範例會將現有的標準參數轉換為進階參數。

```
aws ssm put-parameter \
    --name "MyConvertedParameter" \
    --value "abc123" \
    --type "String" \
    --tier Advanced \
    --overwrite
```

輸出：

```
{
    "Version": 2,
    "Tier": "Advanced"
}
```

如需詳細資訊，請參閱 Systems Manager 使用者指南中的建立 Systems Manager 參數 (AWS CLI)、管理參數層和使用參數政策。 AWS

範例 4：建立已附加政策的參數

下列 put-parameter 範例示範建立已附加參數政策的進階參數。

```
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\"}}]"
```

輸出：

```
{
    "Version": 1,
    "Tier": "Advanced"
}
```

如需詳細資訊，請參閱 Systems Manager 使用者指南中的建立 Systems Manager 參數 (AWS CLI)、管理參數層和使用參數政策。 AWS

範例 5：將政策新增至現有參數

下列 put-parameter 範例示範將政策連接至現有的進階參數。

```
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
```

輸出：

```
{
    "Version": 2,
    "Tier": "Advanced"
}
```

如需詳細資訊，請參閱 Systems Manager 使用者指南中的建立 Systems Manager 參數 (AWS CLI)、管理參數層和使用參數政策。 AWS

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 PutParameter。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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);
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 PutParameter。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範建立參數。如果命令成功，則無輸出訊息。

```
Write-SSMParameter -Name "Welcome" -Type "String" -Value "helloWorld"
```

範例 2：此範例示範變更參數。如果命令成功，則無輸出訊息。

```
Write-SSMParameter -Name "Welcome" -Type "String" -Value "Good day, Sunshine!" -
Overwrite $true
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 PutParameter。

PowerShell V5 的工具

範例 1：此範例示範建立參數。如果命令成功，則無輸出訊息。

```
Write-SSMParameter -Name "Welcome" -Type "String" -Value "helloWorld"
```

範例 2：此範例示範變更參數。如果命令成功，則無輸出訊息。

```
Write-SSMParameter -Name "Welcome" -Type "String" -Value "Good day, Sunshine!" -
Overwrite $true
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 PutParameter。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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(())
}
```

- 如需 API 詳細資訊，請參閱《適用於 Rust 的AWS SDK API 參考》中的 PutParameter。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **RegisterDefaultPatchBaseline** 與 CLI

下列程式碼範例示範如何使用 RegisterDefaultPatchBaseline。

CLI

AWS CLI

設定預設的修補基準

下列 register-default-patch-baseline 範例示範將指定的自訂修補基準註冊為其支援的作業系統類型的預設修補基準。

```
aws ssm register-default-patch-baseline \
    --baseline-id "pb-abc123cf9bEXAMPLE"
```

輸出：

```
{
    "BaselineId":"pb-abc123cf9bEXAMPLE"
}
```

下列register-default-patch-baseline範例會將 AWS for CentOS 提供的預設修補程式基準註冊為預設修補程式基準。

```
aws ssm register-default-patch-baseline \
    --baseline-id "arn:aws:ssm:us-east-2:733109147000:patchbaseline/
pb-0574b43a65ea646ed"
```

輸出：

```
{
    "BaselineId":"pb-abc123cf9bEXAMPLE"
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 About Predefined and Custom Patch Baselines。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 RegisterDefaultPatchBaseline。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範將修補基準註冊為預設修補基準。

```
Register-SSMDefaultPatchBaseline -BaselineId "pb-03da896ca3b68b639"
```

輸出：

```
pb-03da896ca3b68b639
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 RegisterDefaultPatchBaseline。

PowerShell V5 的工具

範例 1：此範例示範將修補基準註冊為預設修補基準。

```
Register-SSMDefaultPatchBaseline -BaselineId "pb-03da896ca3b68b639"
```

輸出：

```
pb-03da896ca3b68b639
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 RegisterDefaultPatchBaseline。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **RegisterPatchBaselineForPatchGroup** 與 CLI

下列程式碼範例示範如何使用 `RegisterPatchBaselineForPatchGroup`。

CLI

AWS CLI

註冊修補程式群組的修補基準

下列 register-patch-baseline-for-patch-group 範例示範註冊修補程式群組的修補基準。

```
aws ssm register-patch-baseline-for-patch-group \
    --baseline-id "pb-045f10b4f382baeda" \
    --patch-group "Production"
```

輸出：

```
{
    "BaselineId": "pb-045f10b4f382baeda",
    "PatchGroup": "Production"
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Create a Patch Group <https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-group-tagging.html>__ 和 Add a Patch Group to a Patch Baseline。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 RegisterPatchBaselineForPatchGroup。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範註冊修補程式群組的修補基準。

```
Register-SSMPatchBaselineForPatchGroup -BaselineId "pb-03da896ca3b68b639" -
PatchGroup "Production"
```

輸出：

```
BaselineId           PatchGroup
----------           ----------
pb-03da896ca3b68b639 Production
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 RegisterPatchBaselineForPatchGroup。

PowerShell V5 的工具

範例 1：此範例示範註冊修補程式群組的修補基準。

```
Register-SSMPatchBaselineForPatchGroup -BaselineId "pb-03da896ca3b68b639" -
PatchGroup "Production"
```

輸出：

```
BaselineId           PatchGroup
----------           ----------
pb-03da896ca3b68b639 Production
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 RegisterPatchBaselineForPatchGroup。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **RegisterTargetWithMaintenanceWindow** 與 CLI

下列程式碼範例示範如何使用 RegisterTargetWithMaintenanceWindow。

CLI

AWS CLI

範例 1：向維護時段註冊單一目標

下列 register-target-with-maintenance-window 範例示範向維護時段註冊執行個體。

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-ab12cd34ef56gh78" \
    --target "Key=InstanceIds,Values=i-0000293ffd8c57862" \
    --owner-information "Single instance" \
    --resource-type "INSTANCE"
```

輸出：

```
{
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

範例 2：使用執行個體 ID 向維護時段註冊多個目標

下列 register-target-with-maintenance-window 範例示範透過指定執行個體 ID，向維護時段註冊兩個執行個體。

```
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"
```

輸出：

```
{
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

範例 3：使用資源標籤向維護時段註冊目標

下列 register-target-with-maintenance-window 範例示範透過指定已套用至執行個體的資源標籤，向維護時段註冊執行個體。

```
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"
```

輸出：

```
{
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

範例 4：使用一組標籤索引鍵註冊目標

下列 register-target-with-maintenance-window 範例示範註冊執行個體，這些執行個體擁有一個或多個標指派的標籤索引鍵，無論其索引鍵值為何。

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-0c50858d01EXAMPLE" \
    --resource-type "INSTANCE" \
```

```
    --target "Key=tag-key,Values=Name,Instance-Type,CostCenter"
```

輸出：

```
{
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

範例 5：使用資源群組名稱註冊目標

下列 `register-target-with-maintenance-window` 範例示範註冊指定的資源群組，無論其中包含的資源類型為何。

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-0c50858d01EXAMPLE" \
    --resource-type "RESOURCE_GROUP" \
    --target "Key=resource-groups:Name,Values=MyResourceGroup"
```

輸出：

```
{
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

如需詳細資訊，請參閱 AWS Systems Manager 使用者指南中的向維護時段 (AWS CLI) 註冊目標執行個體。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 RegisterTargetWithMaintenanceWindow。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範向維護時段註冊執行個體。

```
$option1 = @{Key="InstanceIds";Values=@("i-0000293ffd8c57862")}
Register-SSMTargetWithMaintenanceWindow -WindowId "mw-06cf17cbefcb4bf4f" -Target
 $option1 -OwnerInformation "Single instance" -ResourceType "INSTANCE"
```

輸出：

```
d8e47760-23ed-46a5-9f28-927337725398
```

範例 2：此範例示範向維護時段註冊多個執行個體。

```
$option1 =
 @{Key="InstanceIds";Values=@("i-0000293ffd8c57862","i-0cb2b964d3e14fd9f")}
Register-SSMTargetWithMaintenanceWindow -WindowId "mw-06cf17cbefcb4bf4f" -Target
 $option1 -OwnerInformation "Single instance" -ResourceType "INSTANCE"
```

輸出：

```
6ab5c208-9fc4-4697-84b7-b02a6cc25f7d
```

範例 3：此範例示範使用 EC2 標籤向維護時段註冊執行個體。

```
$option1 = @{Key="tag:Environment";Values=@("Production")}
Register-SSMTargetWithMaintenanceWindow -WindowId "mw-06cf17cbefcb4bf4f" -Target
 $option1 -OwnerInformation "Production Web Servers" -ResourceType "INSTANCE"
```

輸出：

```
2994977e-aefb-4a71-beac-df620352f184
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 RegisterTargetWithMaintenanceWindow。

PowerShell V5 的工具

範例 1：此範例示範向維護時段註冊執行個體。

```
$option1 = @{Key="InstanceIds";Values=@("i-0000293ffd8c57862")}
Register-SSMTargetWithMaintenanceWindow -WindowId "mw-06cf17cbefcb4bf4f" -Target
 $option1 -OwnerInformation "Single instance" -ResourceType "INSTANCE"
```

輸出：

```
d8e47760-23ed-46a5-9f28-927337725398
```

範例 2：此範例示範向維護時段註冊多個執行個體。

```
$option1 =
 @{Key="InstanceIds";Values=@("i-0000293ffd8c57862","i-0cb2b964d3e14fd9f")}
Register-SSMTargetWithMaintenanceWindow -WindowId "mw-06cf17cbefcb4bf4f" -Target
 $option1 -OwnerInformation "Single instance" -ResourceType "INSTANCE"
```

輸出：

```
6ab5c208-9fc4-4697-84b7-b02a6cc25f7d
```

範例 3：此範例示範使用 EC2 標籤向維護時段註冊執行個體。

```
$option1 = @{Key="tag:Environment";Values=@("Production")}
Register-SSMTargetWithMaintenanceWindow -WindowId "mw-06cf17cbefcb4bf4f" -Target
 $option1 -OwnerInformation "Production Web Servers" -ResourceType "INSTANCE"
```

輸出：

```
2994977e-aefb-4a71-beac-df620352f184
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 RegisterTargetWithMaintenanceWindow。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **RegisterTaskWithMaintenanceWindow** 與 CLI

下列程式碼範例示範如何使用 `RegisterTaskWithMaintenanceWindow`。

CLI

AWS CLI

範例 1：向維護時段註冊 Automation 任務

下列 `register-task-with-maintenance-window` 範例示範向以執行個體為目標的維護時段註冊 Automation 任務。

```
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"
```

輸出：

```
{
    "WindowTaskId":"11144444-5555-6666-7777-88888888"
}
```

如需詳細資訊，請參閱 AWS Systems Manager 使用者指南中的向維護時段 (AWS CLI) 註冊任務。

範例 2：向維護時段註冊 Lambda 任務

下列 `register-task-with-maintenance-window` 範例示範向以執行個體為目標的維護時段註冊 Lambda 任務。

```
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"
```

輸出：

```
{
    "WindowTaskId":"22244444-5555-6666-7777-88888888"
}
```

如需詳細資訊，請參閱 AWS Systems Manager 使用者指南中的向維護時段 (AWS CLI) 註冊任務。

範例 3：向維護時段註冊 Run Command 任務

下列 `register-task-with-maintenance-window` 範例示範向以執行個體為目標的維護時段註冊 Run Command 任務。

```
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
```

輸出：

```
{
    "WindowTaskId":"33344444-5555-6666-7777-88888888"
}
```

如需詳細資訊，請參閱 AWS Systems Manager 使用者指南中的向維護時段 (AWS CLI) 註冊任務。

範例 4：向維護時段註冊 Step Functions 任務

下列 register-task-with-maintenance-window 範例示範向以執行個體為目標的維護時段註冊 Step Functions 任務。

```
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"
```

輸出：

```
{
    "WindowTaskId":"44444444-5555-6666-7777-88888888"
}
```

如需詳細資訊，請參閱 AWS Systems Manager 使用者指南中的向維護時段 (AWS CLI) 註冊任務。

範例 5：使用維護時段目標 ID 註冊任務

下列 register-task-with-maintenance-window 範例示範使用維護時段目標 ID 註冊任務。維護時段目標 ID 位於 aws ssm register-target-with-maintenance-window 命令的輸出中。您也可以從 aws ssm describe-maintenance-window-targets 命令的輸出中擷取它。

```
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
```

輸出：

```
{
    "WindowTaskId":"33344444-5555-6666-7777-88888888"
}
```

如需詳細資訊，請參閱 AWS Systems Manager 使用者指南中的向維護時段 (AWS CLI) 註冊任務。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 RegisterTaskWithMaintenanceWindow。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範使用執行個體 ID 向維護時段註冊任務。輸出為任務 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
```

輸出：

```
f34a2c47-ddfd-4c85-a88d-72366b69af1b
```

範例 2：此範例示範使用目標 ID 向維護時段註冊任務。輸出為任務 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
```

輸出：

```
f34a2c47-ddfd-4c85-a88d-72366b69af1b
```

範例 3：此範例示範建立 Run Command 文件 **AWS-RunPowerShellScript** 的參數物件，並使用目標 ID 向指定維護時段建立任務。傳回輸出為任務 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
```

輸出：

```
f1e2ef34-5678-12e3-456a-12334c5c6cbe
```

範例 4：此範例會使用名為 的文件來註冊 AWS Systems Manager Automation 任務**Create-Snapshots**。

```
$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"
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 RegisterTaskWithMaintenanceWindow。

PowerShell V5 的工具

範例 1：此範例示範使用執行個體 ID 向維護時段註冊任務。輸出為任務 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
```

輸出：

```
f34a2c47-ddfd-4c85-a88d-72366b69af1b
```

範例 2：此範例示範使用目標 ID 向維護時段註冊任務。輸出為任務 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
```

輸出：

```
f34a2c47-ddfd-4c85-a88d-72366b69af1b
```

範例 3：此範例示範建立 Run Command 文件 **AWS-RunPowerShellScript** 的參數物件，並使用目標 ID 向指定維護時段建立任務。傳回輸出為任務 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
```

輸出：

```
f1e2ef34-5678-12e3-456a-12334c5c6cbe
```

範例 4：此範例會使用名為 的文件來註冊 AWS Systems Manager Automation 任務**Create-Snapshots**。

```
$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"
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 RegisterTaskWithMaintenanceWindow。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **RemoveTagsFromResource** 與 CLI

下列程式碼範例示範如何使用 RemoveTagsFromResource。

CLI

AWS CLI

從修補基準中移除標籤

下列 remove-tags-from-resource 範例示範從修補基準中移除標籤。

```
aws ssm remove-tags-from-resource \
    --resource-type "PatchBaseline" \
    --resource-id "pb-0123456789abcdef0" \
```

```
    --tag-keys "Region"
```

此命令不會產生輸出。

如需詳細資訊，請參閱《 AWS 一般參考》中的標記 AWS 資源。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 RemoveTagsFromResource。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範從維護時段中移除標籤。如果命令成功，則無輸出訊息。

```
Remove-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
 "MaintenanceWindow" -TagKey "Production"
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 RemoveTagsFromResource。

PowerShell V5 的工具

範例 1：此範例示範從維護時段中移除標籤。如果命令成功，則無輸出訊息。

```
Remove-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
 "MaintenanceWindow" -TagKey "Production"
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 RemoveTagsFromResource。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

SendCommand 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 SendCommand。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 了解基本概念

CLI

AWS CLI

範例 1：在一個或多個遠端執行個體上執行命令

下列 send-command 範例示範在目標執行個體上執行 echo 命令。

```
aws ssm send-command \
    --document-name "AWS-RunShellScript" \
    --parameters 'commands=["echo HelloWorld"]' \
    --targets "Key=instanceids,Values=i-1234567890abcdef0" \
    --comment "echo HelloWorld"
```

輸出：

```
{
    "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
        }
    }
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Running Commands Using Systems Manager Run Command。

範例 2：取得有關執行個體的 IP 資訊

以下 send-command 範例示範擷取有關執行個體的 IP 資訊。

```
aws ssm send-command \
    --instance-ids "i-1234567890abcdef0" \
    --document-name "AWS-RunShellScript" \
    --comment "IP config" \
    --parameters "commands=ifconfig"
```

如需範例輸出，請參閱範例 1。

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Running Commands Using Systems Manager Run Command。

範例 3：在具有特定標籤的執行個體上執行命令

下列 send-command 範例示範在具有標籤索引鍵 "ENV" 和值 "Dev" 的執行個體上執行命令。

```
aws ssm send-command \
    --targets "Key=tag:ENV,Values=Dev" \
    --document-name "AWS-RunShellScript" \
    --parameters "commands=ifconfig"
```

如需範例輸出，請參閱範例 1。

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Running Commands Using Systems Manager Run Command。

範例 4：執行傳送 SNS 通知的命令

下列 send-command 範例示範執行傳送所有通知事件和 Command 通知類型之 SNS 通知的命令。

```
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"
```

如需範例輸出，請參閱範例 1。

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Running Commands Using Systems Manager Run Command。

範例 5：執行輸出至 S3 和 CloudWatch 的命令

下列 send-command 範例示範執行命令，該命令將命令詳細資訊輸出至 S3 儲存貯體和 CloudWatch Logs 日誌群組。

```
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"
```

如需範例輸出，請參閱範例 1。

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Running Commands Using Systems Manager Run Command。

範例 6：在具有不同標籤的多個執行個體上執行命令

下列 send-command 範例示範在具有兩個不同標籤索引鍵和值的執行個體上執行命令。

```
aws ssm send-command \
    --document-name "AWS-RunPowerShellScript" \
    --parameters commands=["echo helloWorld"] \
    --targets Key=tag:Env,Values=Dev Key=tag:Role,Values=WebServers
```

如需範例輸出，請參閱範例 1。

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Running Commands Using Systems Manager Run Command。

範例 7：將具有相同標籤索引鍵的多個執行個體設定為目標

下列 send-command 範例示範在具有相同標籤索引鍵但值不同的執行個體上執行命令。

```
aws ssm send-command \
    --document-name "AWS-RunPowerShellScript" \
    --parameters commands=["echo helloWorld"] \
    --targets Key=tag:Env,Values=Dev,Test
```

如需範例輸出，請參閱範例 1。

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Running Commands Using Systems Manager Run Command。

範例 8：執行使用共用文件的命令

下列 send-command 範例示範在目標執行個體上執行共用文件。

```
aws ssm send-command \
    --document-name "arn:aws:ssm:us-east-1:123456789012:document/ExampleDocument"
 \
    --targets "Key=instanceids,Values=i-1234567890abcdef0"
```

如需範例輸出，請參閱範例 1。

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Using shared SSM documents。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 SendCommand。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
   /**
    * 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];
    }
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 SendCommand。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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;
    }
  }
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 SendCommand。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範在目標執行個體上執行 echo 命令。

```
Send-SSMCommand -DocumentName "AWS-RunPowerShellScript" -Parameter @{commands =
 "echo helloWorld"} -Target @{Key="instanceids";Values=@("i-0cb2b964d3e14fd9f")}
```

輸出：

```
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}
```

範例 2：此範例顯示如何執行接受巢狀參數的命令。

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 SendCommand。

PowerShell V5 的工具

範例 1：此範例示範在目標執行個體上執行 echo 命令。

```
Send-SSMCommand -DocumentName "AWS-RunPowerShellScript" -Parameter @{commands =
 "echo helloWorld"} -Target @{Key="instanceids";Values=@("i-0cb2b964d3e14fd9f")}
```

輸出：

```
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}
```

範例 2：此範例顯示如何執行接受巢狀參數的命令。

```
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
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 SendCommand。

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 SendCommand。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **StartAutomationExecution** 與 CLI

下列程式碼範例示範如何使用 StartAutomationExecution。

CLI

AWS CLI

範例 1：執行 Automation 文件

下列 start-automation-execution 範例示範執行 Automation 文件。

```
aws ssm start-automation-execution \
    --document-name "AWS-UpdateLinuxAmi" \
    --parameters "AutomationAssumeRole=arn:aws:iam::123456789012:role/
SSMAutomationRole,SourceAmiId=ami-EXAMPLE,IamInstanceProfileName=EC2InstanceRole"
```

輸出：

```
{
  "AutomationExecutionId": "4105a4fc-f944-11e6-9d32-0a1b2EXAMPLE"
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Running an Automation Workflow Manually。

範例 2：執行共用 Automation 文件

下列 start-automation-execution 範例示範執行共用的 Automation 文件。

```
aws ssm start-automation-execution \
    --document-name "arn:aws:ssm:us-east-1:123456789012:document/ExampleDocument"
```

輸出：

```
{
  "AutomationExecutionId": "4105a4fc-f944-11e6-9d32-0a1b2EXAMPLE"
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Using shared SSM documents。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 StartAutomationExecution。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範執行指定自動化角色、AMI 來源 ID 和 Amazon EC2 執行個體角色的文件。

```
Start-SSMAutomationExecution -DocumentName AWS-UpdateLinuxAmi -
Parameter @{'AutomationAssumeRole'='arn:aws:iam::123456789012:role/
SSMAutomationRole';'SourceAmiId'='ami-
f173cc91';'InstanceIamRole'='EC2InstanceRole'}
```

輸出：

```
3a532a4f-0382-11e7-9df7-6f11185f6dd1
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 StartAutomationExecution。

PowerShell V5 的工具

範例 1：此範例示範執行指定自動化角色、AMI 來源 ID 和 Amazon EC2 執行個體角色的文件。

```
Start-SSMAutomationExecution -DocumentName AWS-UpdateLinuxAmi -
Parameter @{'AutomationAssumeRole'='arn:aws:iam::123456789012:role/
```

```
SSMAutomationRole';'SourceAmiId'='ami-
f173cc91';'InstanceIamRole'='EC2InstanceRole'}
```

輸出：

```
3a532a4f-0382-11e7-9df7-6f11185f6dd1
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 StartAutomationExecution。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **StartSession** 與 CLI

下列程式碼範例示範如何使用 `StartSession`。

CLI

AWS CLI

範例 1：啟動 Session Manager 工作階段

此 `start-session` 範例示範建立與 Session Manager 工作階段執行個體之間的連線。請注意，若要使用此互動式命令，則需要在進行呼叫的用戶端機器上安裝 Session Manager 外掛程式。

```
aws ssm start-session \
    --target "i-1234567890abcdef0"
```

輸出：

```
Starting session with SessionId: Jane-Roe-07a16060613c408b5
```

範例 2：使用 SSH 啟動 Session Manager 工作階段

此 `start-session` 範例示範使用 SSH 建立與 Session Manager 工作階段執行個體之間的連線。請注意，若要使用此互動式命令，則需要在進行呼叫的用戶端機器上安裝 Session Manager 外掛程式；而且該命令會使用執行個體上的預設使用者，例如 Linux EC2 執行個體的 `ec2-user`。

```
ssh -i /path/my-key-pair.pem ec2-user@i-02573cafcfEXAMPLE
```

輸出：

```
Starting session with SessionId: ec2-user-07a16060613c408b5
```

如需詳細資訊，請參閱 Systems Manager 使用者指南中的啟動工作階段和安裝 CLI 的 Session Manager 外掛程式。 AWS AWS

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 StartSession。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範啟動與 Session Manager 工作階段目標的連線，以啟用連接埠轉送。

```
Start-SSMSession -Target 'i-064578e5e7454488f' -DocumentName 'AWS-
StartPortForwardingSession' -Parameter @{ localPortNumber = '8080'; portNumber =
 '80' }
```

輸出：

```
SessionId    StreamUrl
----------    ----------
random-id0    wss://ssmmessages.amazonaws.com/v1/data-channel/random-id
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V4) 中的 StartSession。

PowerShell V5 的工具

範例 1：此範例示範啟動與 Session Manager 工作階段目標的連線，以啟用連接埠轉送。注意：如果尚未使用 AWSCredentials 設定 ，則需要新增區域參數。

```
Start-SSMSession -Target 'i-064578e5e7454488f' -DocumentName 'AWS-
StartPortForwardingSession' -Parameter @{ localPortNumber = '8080'; portNumber =
 '80' } -Region 'us-west-1'
```

輸出：

```
Starting session with SessionId: testUser-xi4glew849asyeryde34u4dfsdfy
Port 8080 opened for sessionId testUser-xi4glew849asyeryde34u4dfsdfy.
Waiting for connections...
```

範例 2：此範例會建立具有 Session Manager 工作階段指定執行個體的互動式工作階段。

```
Start-SSMSession -Target 'i-1234567890abcdef0' -Region 'us-west-1'
```

輸出：

```
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>
```

範例 3：此範例會在未連線到工作階段的情況下建立工作階段，並傳回連線至工作階段所需的 SessionId、StreamUrl 和 TokenValue 屬性。

```
Start-SSMSession -Target 'i-1234567890abcdef0' -Region 'us-west-1' -
DisablePluginInvocation
```

輸出：

```
SessionId        : testUser-xi4glew849asyeryde34u4dfsdfy
StreamUrl        : {StreamUrl value redacted}
TokenValue       : {Token value redacted}
ContentLength    : 1207
HttpStatusCode   : OK
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 StartSession。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **StopAutomationExecution** 與 CLI

下列程式碼範例示範如何使用 StopAutomationExecution。

CLI

AWS CLI

停止自動化執行

下列 stop-automation-execution 範例示範停止執行 Automation 文件。

```
aws ssm stop-automation-execution
    --automation-execution-id "4105a4fc-f944-11e6-9d32-0a1b2EXAMPLE"
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Running an Automation Workflow Manually。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 StopAutomationExecution。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範停止自動化執行。如果命令成功，則無輸出訊息。

```
Stop-SSMAutomationExecution -AutomationExecutionId "4105a4fc-
f944-11e6-9d32-8fb2db27a909"
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 StopAutomationExecution。

PowerShell V5 的工具

範例 1：此範例示範停止自動化執行。如果命令成功，則無輸出訊息。

```
Stop-SSMAutomationExecution -AutomationExecutionId "4105a4fc-
f944-11e6-9d32-8fb2db27a909"
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 StopAutomationExecution。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **UpdateAssociation** 與 CLI

下列程式碼範例示範如何使用 UpdateAssociation。

CLI

AWS CLI

範例 1：更新文件關聯

下列 update-association 範例示範以新文件版本更新關聯。

```
aws ssm update-association \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab" \
    --document-version "\$LATEST"
```

輸出：

```
{
    "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
    }
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Editing and creating a new version of an association。

範例 2：更新關聯的排程表達式

下列 update-association 範例示範更新指定關聯的排程表達式。

```
aws ssm update-association \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab" \
    --schedule-expression "cron(0 0 0/4 1/1 * ? *)"
```

輸出：

```
{
    "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
```

```
    }
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Editing and creating a new version of an association。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UpdateAssociation。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範以新文件版本更新關聯。

```
Update-SSMAssociation -AssociationId "93285663-92df-44cb-9f26-2292d4ecc439" -
DocumentVersion "1"
```

輸出：

```
Name                  : AWS-UpdateSSMAgent
InstanceId            :
Date                  : 3/1/2017 6:22:21 PM
Status.Name           :
Status.Date           :
Status.Message        :
Status.AdditionalInfo :
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 UpdateAssociation。

PowerShell V5 的工具

範例 1：此範例示範以新文件版本更新關聯。

```
Update-SSMAssociation -AssociationId "93285663-92df-44cb-9f26-2292d4ecc439" -
DocumentVersion "1"
```

輸出：

```
Name                  : AWS-UpdateSSMAgent
InstanceId            :
```

```
Date                  : 3/1/2017 6:22:21 PM
Status.Name           :
Status.Date           :
Status.Message        :
Status.AdditionalInfo :
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 UpdateAssociation。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **UpdateAssociationStatus** 與 CLI

下列程式碼範例示範如何使用 UpdateAssociationStatus。

CLI

AWS CLI

更新關聯狀態

下列 update-association-status 範例示範更新執行個體與文件之間的關聯狀態。

```
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"
```

輸出：

```
{
    "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
    }
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Working with associations in Systems Manager。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UpdateAssociationStatus。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範更新執行個體與組態文件之間的關聯狀態。

```
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"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 UpdateAssociationStatus。

PowerShell V5 的工具

範例 1：此範例示範更新執行個體與組態文件之間的關聯狀態。

```
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"
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 UpdateAssociationStatus。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **UpdateDocument** 與 CLI

下列程式碼範例示範如何使用 UpdateDocument。

CLI

AWS CLI

建立新版本的文件

下列 update-document 範例示範在 Windows 電腦上執行時建立新版本的文件。--document 指定的文件必須為 JSON 格式。請注意，必須參考 file://，後面接著內容檔案的路徑。由於 --document-version 參數開頭的 $，在 Windows 上，必須使用雙引號括住值。在 Linux、MacOS 或 PowerShell 提示中，必須使用單引號括住值。

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'
```

輸出：

```
{
  "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"
    }
}
```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UpdateDocument。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範建立新版本的文件，其中包含您指定的 json 檔案的更新內容。文件必須為 JSON 格式。您可以使用 "Get-SSMDocumentVersionList" cmdlet 取得文件版本。

```
Update-SSMDocument -Name RunShellScript -DocumentVersion "1" -Content (Get-
Content -Raw "c:\temp\RunShellScript.json")
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 UpdateDocument。

PowerShell V5 的工具

範例 1：此範例示範建立新版本的文件，其中包含您指定的 json 檔案的更新內容。文件必須為 JSON 格式。您可以使用 "Get-SSMDocumentVersionList" cmdlet 取得文件版本。

```
Update-SSMDocument -Name RunShellScript -DocumentVersion "1" -Content (Get-
Content -Raw "c:\temp\RunShellScript.json")
```

輸出：

```
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
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 UpdateDocument。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **UpdateDocumentDefaultVersion** 與 CLI

下列程式碼範例示範如何使用 UpdateDocumentDefaultVersion。

CLI

AWS CLI

更新文件的預設版本

下列 update-document-default-version 範例示範更新 Systems Manager 文件的預設版本。

```
aws ssm update-document-default-version \
    --name "Example" \
    --document-version "2"
```

輸出：

```
{
    "Description": {
        "Name": "Example",
        "DefaultVersion": "2"
    }
}
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Writing SSM Document Content。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UpdateDocumentDefaultVersion。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範更新文件的預設版本。您可以使用 "Get-SSMDocumentVersionList" cmdlet 取得可用的文件版本。

```
Update-SSMDocumentDefaultVersion -Name "RunShellScript" -DocumentVersion "2"
```

輸出：

```
DefaultVersion Name
-------------- ----
2              RunShellScript
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 UpdateDocumentDefaultVersion。

PowerShell V5 的工具

範例 1：此範例示範更新文件的預設版本。您可以使用 "Get-SSMDocumentVersionList" cmdlet 取得可用的文件版本。

```
Update-SSMDocumentDefaultVersion -Name "RunShellScript" -DocumentVersion "2"
```

輸出：

```
DefaultVersion Name
-------------- ----
2              RunShellScript
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 UpdateDocumentDefaultVersion。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`UpdateMaintenanceWindow` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `UpdateMaintenanceWindow`。

CLI

AWS CLI

範例 1：更新維護時段

下列 `update-maintenance-window` 範例示範更新維護時段的名稱。

```
aws ssm update-maintenance-window \
    --window-id "mw-1a2b3c4d5e6f7g8h9" \
    --name "My-Renamed-MW"
```

輸出：

```
{
```

```
    "Cutoff": 1,
    "Name": "My-Renamed-MW",
    "Schedule": "cron(0 16 ? * TUE *)",
    "Enabled": true,
    "AllowUnassociatedTargets": true,
    "WindowId": "mw-1a2b3c4d5e6f7g8h9",
    "Duration": 4
}
```

範例 2：停用維護時段

下列 update-maintenance-window 範例示範停用維護時段。

```
aws ssm update-maintenance-window \
    --window-id "mw-1a2b3c4d5e6f7g8h9" \
    --no-enabled
```

範例 3：啟用維護時段

下列 update-maintenance-window 範例示範啟用維護時段。

```
aws ssm update-maintenance-window \
    --window-id "mw-1a2b3c4d5e6f7g8h9" \
    --enabled
```

如需詳細資訊，請參閱 AWS Systems Manager 使用者指南中的更新維護時段 (AWS CLI)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UpdateMaintenanceWindow。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /**
```

```
    * 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();
   }
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 UpdateMaintenanceWindow。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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;
    }
  }
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 UpdateMaintenanceWindow。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範更新維護時段的名稱。

```
Update-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d" -Name "My-Renamed-
MW"
```

輸出：

```
AllowUnassociatedTargets : False
Cutoff                   : 1
Duration                 : 2
Enabled                  : True
Name                     : My-Renamed-MW
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d
```

範例 2：此範例示範啟用維護時段。

```
Update-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d" -Enabled $true
```

輸出：

```
AllowUnassociatedTargets : False
Cutoff                   : 1
Duration                 : 2
```

```
Enabled                   : True
Name                      : My-Renamed-MW
Schedule                  : cron(0 */30 * * * ? *)
WindowId                  : mw-03eb9db42890fb82d
```

範例 3：此範例示範停用維護時段。

```
Update-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d" -Enabled $false
```

輸出：

```
AllowUnassociatedTargets : False
Cutoff                   : 1
Duration                 : 2
Enabled                  : False
Name                     : My-Renamed-MW
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 UpdateMaintenanceWindow。

PowerShell V5 的工具

範例 1：此範例示範更新維護時段的名稱。

```
Update-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d" -Name "My-Renamed-
MW"
```

輸出：

```
AllowUnassociatedTargets : False
Cutoff                   : 1
Duration                 : 2
Enabled                  : True
Name                     : My-Renamed-MW
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d
```

範例 2：此範例示範啟用維護時段。

```
Update-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d" -Enabled $true
```

輸出：

```
AllowUnassociatedTargets : False
Cutoff                   : 1
Duration                 : 2
Enabled                  : True
Name                     : My-Renamed-MW
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d
```

範例 3：此範例示範停用維護時段。

```
Update-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d" -Enabled $false
```

輸出：

```
AllowUnassociatedTargets : False
Cutoff                   : 1
Duration                 : 2
Enabled                  : False
Name                     : My-Renamed-MW
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 UpdateMaintenanceWindow。

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 UpdateMaintenanceWindow。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **UpdateManagedInstanceRole** 與 CLI

下列程式碼範例示範如何使用 UpdateManagedInstanceRole。

CLI

AWS CLI

更新受管執行個體的 IAM 角色

下列 update-managed-instance-role 範例示範更新受管執行個體的 IAM 執行個體設定檔。

```
aws ssm update-managed-instance-role \
    --instance-id "mi-08ab247cdfEXAMPLE" \
    --iam-role "ExampleRole"
```

此命令不會產生輸出。

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Step 4: Create an IAM Instance Profile for Systems Manager。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UpdateManagedInstanceRole。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範更新受管執行個體的角色。如果命令成功，則無輸出訊息。

```
Update-SSMManagedInstanceRole -InstanceId "mi-08ab247cdf1046573" -IamRole
 "AutomationRole"
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 UpdateManagedInstanceRole。

PowerShell V5 的工具

範例 1：此範例示範更新受管執行個體的角色。如果命令成功，則無輸出訊息。

```
Update-SSMManagedInstanceRole -InstanceId "mi-08ab247cdf1046573" -IamRole
 "AutomationRole"
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 UpdateManagedInstanceRole。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`UpdateOpsItem` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `UpdateOpsItem`。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 了解基本概念

CLI

AWS CLI

更新 OpsItem

下列 `update-ops-item` 範例示範更新 OpsItem 的描述、優先順序和類別。此外，該命令還指定 SNS 主題，在編輯或變更此 OpsItem 時會傳送通知。

```
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"
```

輸出：

```
This command produces no output.
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Working with OpsItems。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UpdateOpsItem。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
    /**
     * 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;
        }
    }
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 UpdateOpsItem。

JavaScript

SDK for JavaScript (v3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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;
    }
  }
};
```

- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 UpdateOpsItem。

Python

SDK for Python (Boto3)

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
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
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 UpdateOpsItem。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **UpdatePatchBaseline** 與 CLI

下列程式碼範例示範如何使用 UpdatePatchBaseline。

CLI

AWS CLI

範例 1：更新修補基準

下列 update-patch-baseline 範例示範將指定修補程式新增到指定修補基準，兩個修補程式新增為已拒絕，一個修補程式新增為已核准。

```
aws ssm update-patch-baseline \
        --baseline-id "pb-0123456789abcdef0" \
        --rejected-patches "KB2032276" "MS10-048" \
        --approved-patches "KB2124261"
```

輸出：

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

範例 2：重新命名修補基準

下列 update-patch-baseline 範例示範重新命名指定的修補基準。

```
aws ssm update-patch-baseline \
    --baseline-id "pb-0713accee01234567" \
    --name "Windows-Server-2012-R2-Important-and-Critical-Security-Updates"
```

如需詳細資訊，請參閱《AWS Systems Manager 使用者指南》中的 Update or Delete a Patch Baseline <https://docs.aws.amazon.com/systems-manager/latest/userguide/patch-baseline-update-or-delete.html>`__。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UpdatePatchBaseline。

PowerShell

PowerShell V4 的工具

範例 1：此範例示範將修補程式新增到現有修補基準，兩個修補程式新增為已拒絕，一個修補程式新增為已核准。

```
Update-SSMPatchBaseline -BaselineId "pb-03da896ca3b68b639" -RejectedPatch
 "KB2032276","MS10-048" -ApprovedPatch "KB2124261"
```

輸出：

```
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}
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 UpdatePatchBaseline。

PowerShell V5 的工具

範例 1：此範例示範將修補程式新增到現有修補基準，兩個修補程式新增為已拒絕，一個修補程式新增為已核准。

```
Update-SSMPatchBaseline -BaselineId "pb-03da896ca3b68b639" -RejectedPatch
 "KB2032276","MS10-048" -ApprovedPatch "KB2124261"
```

輸出：

```
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}
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 UpdatePatchBaseline。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

在 中記錄和監控 AWS Systems Manager

監控是維護 和 AWS 解決方案的可靠性、可用性 AWS Systems Manager 和效能的重要部分。您應該從 AWS 解決方案的所有部分收集監控資料，以便在發生多點失敗時偵錯。但在開始監控 Systems Manager 之前，您應該建立監控計劃來回答下列問題：

- 監控目標是什麼？
- 要監控哪些資源？
- 監控這些資源的頻率為何？
- 要使用哪些監控工具？
- 誰會執行監控任務？
- 發生問題時應該通知誰？

在您定義監控目標並建立監控計畫之後，下一步是建立您環境中的正常 Systems Manager 效能基準。您應該在不同的時間及負載條件下測量 Systems Manager 效能。當您監控 Systems Manager 時，應該存放所收集的監控資料的歷史記錄。您可以比較目前的 Systems Manager 效能資料與歷史資料，協助您辨識正常效能模式和效能異常狀況，並建立其處理方式。

例如，您可以監控作業的成功或失敗，例如「自動化」工作流程、修補基準的應用程式、維護時段事件以及組態相容性。自動化是 中的工具 AWS Systems Manager。

您可以監控受管節點的 CPU 使用率、磁碟輸入/輸出和網路使用率。若效能不符合您所建立的基準，您可能需要重新設定或將節點最佳化，以降低 CPU 使用率、改善磁碟輸入/輸出、降低網路流量。如需有關監控 EC2 執行個體的詳細資訊，請參閱《Amazon EC2 使用者指南》中的監控 Amazon EC2。

主題

- 監控工具
- 將節點日誌傳送至統一 CloudWatch Logs (CloudWatch 代理程式)
- 將 SSM Agent 日誌傳送至 CloudWatch Logs
- 監控您的變更請求事件
- 監控自動化
- 使用 Amazon CloudWatch 監控 Run Command 指標
- 使用 記錄 AWS Systems Manager API 呼叫 AWS CloudTrail
- 使用 CloudWatch Logs 記錄自動化動作輸出

- 設定 Run Command 的 Amazon CloudWatch Logs
- 使用 Amazon EventBridge 監控 Systems Manager
- 使用 Amazon SNS 通知監控 Systems Manager 狀態變更

監控工具

本章的內容提供使用可用於監控 Systems Manager 和其他 AWS 資源的工具的資訊。如需完整的工具清單，請參閱 AWS Systems Manager 中的日誌記錄和監控。

將節點日誌傳送至統一 CloudWatch Logs (CloudWatch 代理程式)

您可以設定和使用 Amazon CloudWatch 代理程式從您的節點收集指標和日誌，而不是使用這些 AWS Systems Manager 代理程式 (SSM Agent) 執行這些任務。CloudWatch 代理程式允許您在 EC2 執行個體上收集的指標，比利用 SSM Agent 收集的指標更多。此外，您可以使用 CloudWatch 代理程式從內部部署伺服器收集指標。

您也可以將代理程式組態設定存放在 Systems Manager 中Parameter Store，以與 CloudWatch 代理程式搭配使用。 Parameter Store 是 中的工具 AWS Systems Manager。

> **Note**
>
> AWS Systems Manager 支援從 遷移SSM Agent至統一 CloudWatch 代理程式，以僅收集 64 位元版本的 Windows 日誌和指標。如需在其他作業系統上設定統一 CloudWatch 代理程式的相關資訊，以及使用 CloudWatch 代理程式的完整資訊，請參閱《Amazon CloudWatch 使用者指南》中的使用 CloudWatch 代理程式從 Amazon EC2 執行個體和內部部署伺服器中收集指標。
> 您可以在其他支援的作業系統上使用 CloudWatch 代理程式，但是您將無法使用 Systems Manager 來執行工具遷移。

SSM Agent 會在各個節點的日誌檔寫入有關執行、排定動作、錯誤和運作狀態的資訊。手動連線至節點來檢視日誌檔，並對 SSM Agent 的問題進行疑難排解，需要耗費許多時間。若要讓節點監控更有效率，您可以設定 SSM Agent 本身或 CloudWatch 代理程式傳送此日誌資料至 Amazon CloudWatch Logs。

> **Important**
>
> 統一的 CloudWatch 代理程式已取代 SSM Agent 作為將日誌資料傳送至 Amazon CloudWatch Logs 的工具。不支援 SSM Agent aws:cloudWatch 插件。建議只使用統一的 CloudWatch 代理程式來執行日誌收集流程。如需詳細資訊，請參閱下列主題：
>
> - 將節點日誌傳送至統一 CloudWatch Logs (CloudWatch 代理程式)
> - 將 Windows Server 節點日誌收集遷移到 CloudWatch 代理程式
> - 《Amazon CloudWatch 使用者指南》中的 Collecting metrics, logs, and traces with the CloudWatch agent。

您可以使用 CloudWatch Logs 即時監控日誌資料；藉由建立一個以上的指標篩選條件，來搜尋和篩選日誌資料；並將歷史資料封存，需要時再進行擷取。如需 CloudWatch Logs 的詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》。

設定代理程式將日誌資料傳送至 Amazon CloudWatch Logs，有下列優點：

- 將所有的 SSM Agent 日誌檔案以集中日誌檔儲存空間存放。
- 更快存取檔案以調查錯誤。
- 日誌檔無限期保留 (可設定)。
- 日誌可以隨時維護及存取，無論節點的狀態為何。
- 存取其他 CloudWatch 功能，例如指標和警示。

如需監控 Session Manager 活動的相關資訊，請參閱 記錄工作階段活動 和 啟用和停用工作階段記錄。

將 Windows Server 節點日誌收集遷移到 CloudWatch 代理程式

如果您在支援的 Windows Server 節點使用 SSM Agent 將 SSM Agent 日誌檔案傳送到 Amazon CloudWatch Logs，則您可以使用 Systems Manager 從 SSM Agent 遷移到 CloudWatch 代理程式作為日誌收集工具，以及遷移您的組態設定。

CloudWatch 代理程式不支援 32 位元版本的 Windows Server。

對於 Windows Server 的 64 位元 EC2 執行個體，您可以自動或手動遷移到 CloudWatch 代理程式。若是現場部署伺服器和虛擬機器，則必須手動執行程序。

> **Note**
>
> 遷移過程中，傳送到 CloudWatch 的資料可能中斷或重複。遷移完成後，您的指標和日誌資料將會再次準確記錄於 CloudWatch。

我們建議在遷移整個機群到 CloudWatch 代理程式前，先使用幾個節點測試遷移。遷移之後，如果您比較習慣使用 SSM Agent 收集日誌，可以再換回去。

> **Important**
>
> 在下列情況下，您將無法使用這個主題中所述的步驟，遷移到 CloudWatch 代理程式：
>
> - 現有的 SSM Agent 組態指定了多個區域。
> - 現有的 SSM Agent 組態指定了多組存取/秘密金鑰登入資料。
>
> 在這些情況下，必須關閉 SSM Agent 的日誌收集，並在無遷移程序的情況下安裝 CloudWatch 代理程式。如需詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的下列主題。
>
> - 安裝 CloudWatch 代理程式
> - 在內部部署伺服器上安裝 CloudWatch 代理程式

開始之前

開始遷移到 CloudWatch 代理程式進行日誌收集前，請先確認您要執行遷移的節點符合這些需求：

- 作業系統為 64 位元版本的 Windows Server。
- 節點上已安裝 SSM Agent 2.2.93.0 或更新版本。
- SSM Agent 在節點上設定進行監控。

主題

- 自動遷移到 CloudWatch 代理程式
- 手動遷移到 CloudWatch 代理程式

自動遷移到 CloudWatch 代理程式

對於Windows Server僅限 的 EC2 執行個體，您可以使用 AWS Systems Manager 主控台或 AWS Command Line Interface (AWS CLI) 自動遷移至 CloudWatch 代理程式做為您的日誌收集工具。

> **Note**
>
> AWS Systems Manager 支援從 遷移SSM Agent至統一 CloudWatch 代理程式，以僅收集 64 位元版本的 Windows 日誌和指標。如需在其他作業系統上設定統一 CloudWatch 代理程式的相關資訊，以及使用 CloudWatch 代理程式的完整資訊，請參閱《Amazon CloudWatch 使用者指南》中的使用 CloudWatch 代理程式從 Amazon EC2 執行個體和內部部署伺服器中收集指標。
> 您可以在其他支援的作業系統上使用 CloudWatch 代理程式，但是您將無法使用 Systems Manager 來執行工具遷移。

遷移成功後，請檢查您的 CloudWatch 結果，以確保您接收到的指標、日誌或 Windows 事件日誌正確。如果對結果滿意，您還可以選擇 將 CloudWatch 代理程式組態設定存放在 Parameter Store。如果遷移不成功或結果不如預期，您可以嘗試 回復為使用 SSM Agent收集日誌。

> **Note**
>
> 如果您要遷移包含 {hostname} 項目的來源組態檔案，請注意，遷移完成後 {hostname} 項目可能改變欄位值。例如，假設以下 "LogStream": "{hostname}" 項目對應至名為 MyLogServer001 的伺服器。
>
> ```
> {
> "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}"
>      }
> }
> ```

遷移之後，這個項目會對應到網域，例如 ip-11-1-1-11.production。ExampleCompany。com。若要保留本機主機名稱值，請指定 `{local_hostname}` 而不是 `{hostname}`。

若要自動遷移到 CloudWatch 代理程式 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Run Command，然後選擇執行命令。
3. 在命令文件清單中，請選擇 `AmazonCloudWatch-MigrateCloudWatchAgent`。
4. 針對 Status (狀態)，請選擇 Enabled (啟用)。
5. 在目標區段中，透過手動指定標籤、選取執行個體或邊緣裝置，或指定資源群組，選擇您要執行這項操作的受管節點。

   > **Tip**
   >
   > 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

6. 對於速率控制：
   - 在並行中，指定可同時執行命令的受管節點數目或百分比。

     > **Note**
     >
     > 如果透過指定套用至受管節點的標籤或指定 AWS 資源群組選取了目標，且您不確定會以多少個受管節點為目標，則透過指定百分比限制可以同時執行文件之目標的數量。

   - 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比之節點上的命令失敗之後，停止在其他受管節點上執行命令。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止傳送命令。仍在處理命令的受管節點也可能會傳送錯誤。
7. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取將命令輸出寫入至 S3 儲存貯體方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

> **Note**
>
> 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給執行個體之執行個體設定檔 (適用於 EC2 執行個體) 或 IAM 服務角色 (啟用混合模式的機器) 的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確定與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

8. 在SNS 通知區段中，如果您要傳送有關命令執行狀態的通知，請選取啟用 SNS 通知核取方塊。

   如需為 Run Command 設定 Amazon SNS 通知的詳細資訊，請參閱 使用 Amazon SNS 通知監控 Systems Manager 狀態變更。

9. 選擇執行。

若要自動遷移到 CloudWatch 代理程式 (AWS CLI)

- 執行下列命令。

```
aws ssm send-command --document-name AmazonCloudWatch-MigrateCloudWatchAgent --
targets Key=instanceids,Values=ID1,ID2,ID3
```

ID1、*ID2* 和 *ID3* 代表您要更新的節點 ID，例如 i-02573cafcfEXAMPLE。

手動遷移到 CloudWatch 代理程式

對於內部部署 Windows Server 節點或 Windows Server 的 EC2 執行個體，請依照以下步驟手動將日誌收集遷移到 Amazon CloudWatch 代理程式。

> **Note**
>
> 如果您要遷移包含 {hostname} 項目的來源組態檔案，請注意，遷移完成後 {hostname} 項目可能改變欄位值。例如，假設以下 "LogStream": "{hostname}" 項目對應至名為 MyLogServer001 的伺服器。

```
{
"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}"
     }
}
```

遷移之後，這個項目會對應到網域，例如 ip-11-1-1-11.production.ExampleCompany.com。若要保留本機主機名稱值，請指定 `{local_hostname}` 而不是 `{hostname}`。

第一步：安裝 CloudWatch 代理程式 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Run Command，然後選擇執行命令。
3. 在命令文件清單中，請選擇 `AWS-ConfigureAWSPackage`。
4. 對於 Action (動作)，選擇 `Install`。
5. 對於名稱，輸入 **`AmazonCloudWatchAgent`**。
6. 在 Version (版本) 中輸入 **`latest`** (如果預設未提供)。
7. 在目標區段中，透過手動指定標籤、選取執行個體或邊緣裝置，或指定資源群組，選擇您要執行這項操作的受管節點。

   > **Tip**
   >
   > 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

8. 對於速率控制：
   - 在並行中，指定可同時執行命令的受管節點數目或百分比。

> **Note**
>
> 如果透過指定套用至受管節點的標籤或指定 AWS 資源群組選取了目標，且您不確定會以多少個受管節點為目標，則透過指定百分比限制可以同時執行文件之目標的數量。

- 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比之節點上的命令失敗之後，停止在其他受管節點上執行命令。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止傳送命令。仍在處理命令的受管節點也可能會傳送錯誤。

9. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取將命令輸出寫入至 S3 儲存貯體方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

> **Note**
>
> 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給執行個體之執行個體設定檔 (適用於 EC2 執行個體) 或 IAM 服務角色 (啟用混合模式的機器) 的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確定與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

10. 在SNS 通知區段中，如果您要傳送有關命令執行狀態的通知，請選取啟用 SNS 通知核取方塊。

    如需為 Run Command 設定 Amazon SNS 通知的詳細資訊，請參閱 使用 Amazon SNS 通知監控 Systems Manager 狀態變更。

11. 選擇執行。

第二步：更新配置資料 JSON 格式

- 若要更新 CloudWatch 代理程式現有組態設定的 JSON 格式，請使用 Run Command、 中的工具 AWS Systems Manager，或直接透過 RDP 連線登入節點，在節點上執行下列 Windows PowerShell 命令，一次一個。

```
cd ${Env:ProgramFiles}\\Amazon\\AmazonCloudWatchAgent
```

```
.\\amazon-cloudwatch-agent-config-wizard.exe --isNonInteractiveWindowsMigration
```

{Env:ProgramFiles} 代表包含 CloudWatch 代理程式的 Amazon 目錄位置，通常是 C:\Program Files。

第三步：設定和啟動 CloudWatch 代理程式 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Run Command，然後選擇執行命令。
3. 在命令文件清單中，請選擇 AWS-RunPowerShellScript。
4. 在 Commands (命令) 中輸入以下兩個命令。

```
cd ${Env:ProgramFiles}\Amazon\AmazonCloudWatchAgent
```

```
.\amazon-cloudwatch-agent-ctl.ps1 -a fetch-config -m ec2 -c file:config.json -s
```

{Env:ProgramFiles} 代表包含 CloudWatch 代理程式的 Amazon 目錄位置，通常是 C:\Program Files。

5. 在目標區段中，透過手動指定標籤、選取執行個體或邊緣裝置，或指定資源群組，選擇您要執行這項操作的受管節點。

> **Tip**
>
> 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

6. 對於速率控制：
   - 在並行中，指定可同時執行命令的受管節點數目或百分比。

> **Note**
>
> 如果透過指定套用至受管節點的標籤或指定 AWS 資源群組選取了目標，且您不確定會以多少個受管節點為目標，則透過指定百分比限制可以同時執行文件之目標的數量。

- 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比之節點上的命令失敗之後，停止在其他受管節點上執行命令。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止傳送命令。仍在處理命令的受管節點也可能會傳送錯誤。

7. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取將命令輸出寫入至 S3 儲存貯體方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

> **Note**
>
> 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給執行個體之執行個體設定檔 (適用於 EC2 執行個體) 或 IAM 服務角色 (啟用混合模式的機器) 的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確定與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

8. 在SNS 通知區段中，如果您要傳送有關命令執行狀態的通知，請選取啟用 SNS 通知核取方塊。

   如需為 Run Command 設定 Amazon SNS 通知的詳細資訊，請參閱 使用 Amazon SNS 通知監控 Systems Manager 狀態變更。

9. 選擇執行。

第四步：在 SSM Agent (主控台) 中關閉日誌收集

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Run Command，然後選擇執行命令。
3. 在命令文件清單中，請選擇 `AWS-ConfigureCloudWatch`。
4. 在 Status (狀態) 中選擇 Disabled (停用)。
5. 在目標區段中，透過手動指定標籤、選取執行個體或邊緣裝置，或指定資源群組，選擇您要執行這項操作的受管節點。

> **Tip**
>
> 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

6. 在 Status (狀態) 中選擇 `Disabled`。
7. 對於速率控制：
   - 在並行中，指定可同時執行命令的受管節點數目或百分比。

     > **Note**
     >
     > 如果透過指定套用至受管節點的標籤或指定 AWS 資源群組選取了目標，且您不確定會以多少個受管節點為目標，則透過指定百分比限制可以同時執行文件之目標的數量。

   - 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比之節點上的命令失敗之後，停止在其他受管節點上執行命令。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止傳送命令。仍在處理命令的受管節點也可能會傳送錯誤。
8. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取將命令輸出寫入至 S3 儲存貯體方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

   > **Note**
   >
   > 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給執行個體之執行個體設定檔 (適用於 EC2 執行個體) 或 IAM 服務角色 (啟用混合模式的機器) 的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確定與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

9. 在SNS 通知區段中，如果您要傳送有關命令執行狀態的通知，請選取啟用 SNS 通知核取方塊。

   如需為 Run Command 設定 Amazon SNS 通知的詳細資訊，請參閱 使用 Amazon SNS 通知監控 Systems Manager 狀態變更。
10. 選擇執行。

    完成這些步驟後，請檢查您於 CloudWatch 中的日誌，以確認您接收到的指標、日誌或 Windows 事件日誌正確。如果對結果滿意，您可以選擇 將 CloudWatch 代理程式組態設定存放在 Parameter Store。如果遷移不成功或結果不如預期，您可以 回復為使用 SSM Agent收集日誌。

將 CloudWatch 代理程式組態設定存放在 Parameter Store

您可以將 CloudWatch 代理程式組態檔案的內容存放在 Parameter Store。將此組態資料維護在參數中，多個節點就能由其中衍生組態設定，而不用您在節點建立或手動更新組態檔案。例如，您可以使用 Run Command將 參數的內容寫入多個節點上的組態檔案State Manager，或使用 中的工具 AWS Systems Manager，協助避免 CloudWatch 代理程式組態設定中跨節點機群的組態偏離。

執行 CloudWatch 代理程式組態精靈時，您可以選擇讓精靈將您的組態設定儲存為 Parameter Store 的新參數。如需有關執行 CloudWatch 代理程式組態精靈的資訊，請參閱《Amazon CloudWatch 使用者指南》中的使用精靈建立 CloudWatch 代理程式組態檔案。

如果您執行精靈，但並未選擇將設定儲存為參數的選項，或者您手動建立 CloudWatch 代理程式組態檔案，則可以從下列檔案擷取資料，儲存為您節點上的參數。

```
${Env:ProgramFiles}\Amazon\AmazonCloudWatchAgent\config.json
```

{Env:ProgramFiles} 代表包含 CloudWatch 代理程式的 Amazon 目錄位置，通常是 `C:\Program Files`。

我們建議將此檔案的 JSON 備份到這個節點本身以外的位置。

如需有關建立參數的資訊，請參閱 在 Systems Manager 中建立 Parameter Store 參數。

如需 CloudWatch 代理程式的詳細資訊，請參閱 Amazon CloudWatch 使用者指南中的使用 CloudWatch 代理程式從 Amazon EC2 執行個體和內部部署伺服器收集指標和日誌。

回復為使用 SSM Agent收集日誌

如果您想要換回使用 SSM Agent 收集日誌，請依照以下步驟。

第一步：從 SSM Agent 擷取組態資料

1. 在您想要換回使用 SSM Agent 收集日誌的節點上，找出 SSM Agent 組態檔案的內容。此 JSON 檔案通常位於下列位置：

   ```
   ${Env:ProgramFiles}\\Amazon\\SSM\\Plugins\\awsCloudWatch\
   \AWS.EC2.Windows.CloudWatch.json
   ```

   {Env:ProgramFiles} 代表可找到的 `Amazon` 目錄的位置，通常是 `C:\Program Files`。
2. 複製此資料到文字檔案，以便用於後續步驟。

我們建議將此 JSON 備份到這個節點本身以外的位置。

第二步：解除安裝 CloudWatch 代理程式 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Run Command，然後選擇執行命令。
3. 在命令文件清單中，請選擇 `AWS-ConfigureAWSPackage`。
4. 對於 Action (動作)，選擇 Uninstall (解除安裝)。
5. 對於名稱，輸入 **`AmazonCloudWatchAgent`**。
6. 在目標區段中，透過手動指定標籤、選取執行個體或邊緣裝置，或指定資源群組，選擇您要執行這項操作的受管節點。

   > **Tip**
   >
   > 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

7. 對於速率控制：

   - 在並行中，指定可同時執行命令的受管節點數目或百分比。

     > **Note**
     >
     > 如果透過指定套用至受管節點的標籤或指定 AWS 資源群組選取了目標，且您不確定會以多少個受管節點為目標，則透過指定百分比限制可以同時執行文件之目標的數量。

   - 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比之節點上的命令失敗之後，停止在其他受管節點上執行命令。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止傳送命令。仍在處理命令的受管節點也可能會傳送錯誤。

8. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取將命令輸出寫入至 S3 儲存貯體方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

> **Note**
>
> 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給執行個體之執行個體設定檔 (適用於 EC2 執行個體) 或 IAM 服務角色 (啟用混合模式的機器) 的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確定與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

9. 在SNS 通知區段中，如果您要傳送有關命令執行狀態的通知，請選取啟用 SNS 通知核取方塊。

   如需為 Run Command 設定 Amazon SNS 通知的詳細資訊，請參閱 使用 Amazon SNS 通知監控 Systems Manager 狀態變更。

10. 選擇執行。

第三步：在 SSM Agent (主控台) 中開啟日誌收集

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Run Command，然後選擇執行命令。
3. 在命令文件清單中，請選擇 `AWS-ConfigureCloudWatch`。
4. 在 Status (狀態) 中選擇 `Enabled`。
5. 在 Properties (屬性) 中，貼入您儲存在文字檔案的舊組態資料內容。
6. 在目標區段中，透過手動指定標籤、選取執行個體或邊緣裝置，或指定資源群組，選擇您要執行這項操作的受管節點。

> **Tip**
>
> 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

7. 對於速率控制：

   - 在並行中，指定可同時執行命令的受管節點數目或百分比。

> **Note**
>
> 如果透過指定套用至受管節點的標籤或指定 AWS 資源群組選取了目標，且您不確定會以多少個受管節點為目標，則透過指定百分比限制可以同時執行文件之目標的數量。

- 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比之節點上的命令失敗之後，停止在其他受管節點上執行命令。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止傳送命令。仍在處理命令的受管節點也可能會傳送錯誤。

8. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取將命令輸出寫入至 S3 儲存貯體方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

> **Note**
>
> 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給執行個體之執行個體設定檔 (適用於 EC2 執行個體) 或 IAM 服務角色 (啟用混合模式的機器) 的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確定與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

9. 在SNS 通知區段中，如果您要傳送有關命令執行狀態的通知，請選取啟用 SNS 通知核取方塊。

   如需為 Run Command 設定 Amazon SNS 通知的詳細資訊，請參閱 使用 Amazon SNS 通知監控 Systems Manager 狀態變更。

10. 選擇執行。

將 SSM Agent 日誌傳送至 CloudWatch Logs

AWS Systems Manager 代理程式 (SSM Agent) 是在 EC2 執行個體、邊緣裝置、內部部署伺服器和虛擬機器 (VMs) 上執行的 Amazon 軟體，這些虛擬機器是為 Systems Manager 設定的。 SSM Agent 會處理雲端 Systems Manager 服務的請求，並依照請求中指定的方式設定您的機器。如需有關 SSM Agent 的詳細資訊，請參閱「使用 SSM Agent」。

此外，您可以使用下列步驟，設定 SSM Agent 以傳送日誌資料至 Amazon CloudWatch Logs。

開始之前

在 CloudWatch Logs 中建立日誌群組。如需詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的 CloudWatch Logs 入門。

若要設定 SSM Agent 以將日誌傳送至 CloudWatch

1. 登入節點並找出下列檔案：

   Linux

   在大多數 Linux 節點類型上：`/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. 將檔案名稱從 `seelog.xml.template` 變更為 `seelog.xml`

   > **Note**
   >
   > 在 Ubuntu Server 20.10 STR、20.04、18.04 和 16.04 LTS 上，必須在目錄 `seelog.xml` 中建立檔案 `/etc/amazon/ssm/`。可透過執行下列命令來建立此目錄和檔案。
   >
   > ```
   > 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. 使用文字編輯器開啟 `seelog.xml` 檔案，並找出下列區段。

   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. 編輯檔案，在 </filter> 結束標籤後面新增自訂名稱元素。在以下範例中，自訂名稱已指定為 `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. 儲存變更，然後重新啟動 SSM Agent 或節點。
6. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
7. 在導覽窗格中，選擇 Log groups (日誌群組)，然後選擇日誌群組的名稱。

> **Tip**
>
> SSM Agent日誌檔資料的日誌串流依節點 ID 組織。

監控您的變更請求事件

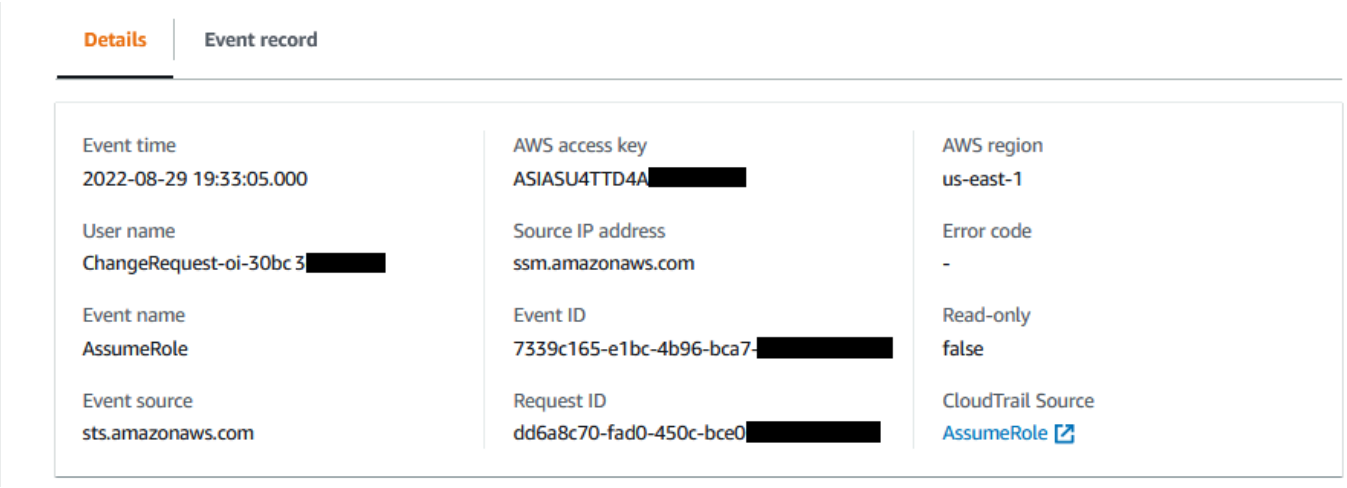
開啟與 AWS CloudTrail Lake 的整合並建立事件資料存放區後，您可以檢視在您的帳戶或組織中執行的變更請求的可稽核詳細資訊。這包括以下詳細資訊：

- 啟動變更請求的使用者身分
- 進行變更的 AWS 區域 位置
- 請求的來源 IP 地址
- 用於請求的 AWS 存取金鑰
- 針對變更請求執行的 API 動作
- 針對這些動作包含的請求參數
- 在此過程中更新的資源

以下是在 AWS CloudTrail Lake 中建立事件資料存放區後，您可以檢視變更請求的事件詳細資訊範例。

Details

下圖顯示在 Details (詳細資訊) 標籤中可用的變更請求的高階資訊。這些詳細資訊包括如下資訊：變更請求操作開始的時間、啟動變更請求的使用者 ID、受影響的 AWS 區域以及與請求相關聯的事件 ID 與請求 ID。



Event record

下圖顯示 CloudTrail Lake 針對變更請求事件提供的 JSON 內容結構。會在變更請求的 Event record (事件記錄) 標籤中提供此資料。

> **Important**
>
> 如果您對組織使用 Change Manager，則可以在登入 Change Manager 的管理帳戶或委派系統管理員帳戶時完成下列程序。
> 不過，若要使用委派系統管理員帳戶來完成這些步驟，必須同時為 CloudTrail 和 Change Manager 指定相同的委派管理員帳戶。
> 當您登入 Change Manager 的管理帳戶時，您可以在 CloudTrail 的 Settings (設定) 頁面中新增或變更 CloudTrail 的委派管理員帳戶。必須先完成此操作，委派的管理員帳戶才能建立事件資料存放區供整個組織使用。

若要從 Change Manager 中開啟 CloudTrail Lake 事件追蹤功能

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Change Manager。
3. 選擇 Requests (請求) 標籤。
4. 選擇任何現有的變更請求，然後選擇 Associated events (關聯事件) 標籤。
5. 選擇 Enable CloudTrail Lake (啟用 CloudTrail Lake)。
6. 請按照《AWS CloudTrail 使用者指南》為 CloudTrail 事件建立事件資料存放區一節中的步驟進行操作。

   若要確保已儲存變更請求的事件資料，在操作過程中請選取下列選項：

   - 在事件類型中，請保持選取預設的 AWS 事件和 CloudTrail 事件。
   - 如果您搭配使用 Change Manager 與組織，請選取針對組織中的所有帳戶啟用。
   - 對於管理事件，請不要清除寫入核取方塊。

   您在建立事件資料存放區時選擇的其他選項不會影響變更請求的事件資料存放。

監控自動化

指標為 Amazon CloudWatch 中的基本概念。指標代表按時間順序發佈到 CloudWatch 的一組資料點。您可以將指標視為要監控的變數，將資料點視作代表該變數隨著時間的值。

自動化是 中的工具 AWS Systems Manager。Systems Manager 會將與 Automation 使用情況相關的指標發佈到 CloudWatch，讓您能夠根據那些指標設定警示。

在 CloudWatch 主控台中檢視 Automation 指標

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇指標。
3. 選擇 SSM。
4. 在指標索引標籤上，選擇用量，然後選擇依 AWS 資源。
5. 在指標清單旁的搜尋方塊中輸入 SSM。

使用 檢視自動化指標 AWS CLI

開啟命令提示並使用下列命令。

```
aws cloudwatch list-metrics \
    --namespace "AWS/Usage"
```

Automation 指標

Systems Manager 會將下列 Automation 指標傳送至 CloudWatch。

指標	描述
ConcurrentAutomationUsage	目前 AWS 帳戶 和 中同時執行的自動化數量 AWS 區域。
QueuedAutomationUsage	目前排入佇列但尚未啟動且狀態為 Pending 的自動化數量。

如需使用 CloudWatch 指標的詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的下列主題：

- 指標
- 使用 Amazon CloudWatch 指標
- 使用 Amazon CloudWatch 警示

使用 Amazon CloudWatch 監控 Run Command 指標

指標為 Amazon CloudWatch 中的基本概念。指標代表按時間順序發佈到 CloudWatch 的一組資料點。您可以將指標視為要監控的變數，且資料點代表該變數隨著時間的值。

AWS Systems Manager 會將有關Run Command命令狀態的指標發佈至 CloudWatch，讓您可以根據這些指標設定警示。 Run Command 是 中的工具 AWS Systems Manager。這些統計資料會保存一段時間，以便您存取歷史資訊，更好了解命令在您 AWS 帳戶中執行的成功率。

您可追蹤指標之命令的終端狀態值包括 `Success`、`Failed` 和 `Delivery Timed Out`。舉例來說，如果有 SSM 命令文件設定為每小時執行一次，就可以設定警示，以便在某個小時沒有回報 `Success` 狀態時向您發出通知。如需不同命令狀態值的詳細資訊，請參閱了解命令狀態。

若要在 CloudWatch 主控台中檢視指標

1. 透過 https://console.aws.amazon.com/cloudwatch/ 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇指標。
3. 在依 AWS 服務區分的警示區域中，針對服務，選擇 SSM-Run Command。

使用 檢視指標 AWS CLI

開啟命令提示並使用下列命令。

```
aws cloudwatch list-metrics --namespace "AWS/SSM-RunCommand"
```

使用以下命令列出所有可用的指標。

```
aws cloudwatch list-metrics
```

Systems Manager Run Command 指標與維度

Systems Manager 每分鐘會向 CloudWatch 傳送 Run Command 命令指標一次。

Systems Manager 會傳送下列命令指標至 CloudWatch。

> **Note**
>
> 這些指標皆使用 `Count` 做為單位，因此 `Sum` 與 `SampleCount` 是最有用的統計資訊。

指標	描述
CommandsDeliveryTimedOut	具有 Delivery Timed Out 終端狀態的命令數。
CommandsFailed	具有 Failed 終端狀態的命令數。
CommandsSucceeded	具有 Success 終端狀態的命令數。

如需使用 CloudWatch 指標的詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的下列主題：

- 指標
- 使用 Amazon CloudWatch 指標
- 使用 Amazon CloudWatch 警示

使用 記錄 AWS Systems Manager API 呼叫 AWS CloudTrail

AWS Systems Manager 已與 整合AWS CloudTrail，此服務提供使用者、角色或 所採取動作的記錄 AWS 服務。CloudTrail 會將 Systems Manager 的所有 API 呼叫擷取為事件。擷取的呼叫包括從 Systems Manager 主控台進行的呼叫，以及針對 Systems Manager API 操作的程式碼呼叫。您可以利用 CloudTrail 收集的資訊來判斷向 Systems Manager 發出的請求，以及發出請求的 IP 位址、時間和其他詳細資訊。

每一筆事件或日誌專案都會包含產生請求者的資訊。身分資訊可協助您判斷下列事項：

- 該請求是使用根使用者還是使用者憑證提出。
- 請求是否代表 IAM Identity Center 使用者提出。
- 提出該請求時，是否使用了特定角色或聯合身分使用者的暫時安全憑證。
- 該請求是否由另一項 AWS 服務服務提出。

當您建立帳戶 AWS 帳戶 時CloudTrail 會在 中處於作用中狀態，而且您會自動存取 CloudTrail 事件歷史記錄。CloudTrail 事件歷史記錄為 AWS 區域中過去 90 天記錄的管理事件，提供可檢視、可搜尋、可下載且不可變的記錄。如需詳細資訊，請參閱「AWS CloudTrail 使用者指南」中的使用 CloudTrail 事件歷史記錄。檢視事件歷史記錄不會產生 CloudTrail 費用。

如需 AWS 帳戶 過去 90 天內持續記錄的事件，請建立追蹤或 CloudTrail Lake 事件資料存放區。

CloudTrail 追蹤

線索能讓 CloudTrail 將日誌檔案交付至 Amazon S3 儲存貯體。使用 建立的所有線索 AWS Management Console 都是多區域。您可以使用 AWS CLI建立單一或多區域追蹤。建議您建立多區域追蹤，因為您擷取 AWS 區域 帳戶中所有 的活動。如果您建立單一區域追蹤，您只能檢視追蹤 AWS 區域中記錄的事件。如需追蹤的詳細資訊，請參閱《AWS CloudTrail 使用者指南》中的為您的 AWS 帳戶建立追蹤和為組織建立追蹤。

您可以透過建立追蹤，免費將持續管理事件的一個複本從 CloudTrail 傳遞至您的 Amazon S3 儲存貯體，但這樣做會產生 Amazon S3 儲存費用。如需 CloudTrail 定價的詳細資訊，請參閱 AWS CloudTrail 定價。如需 Amazon S3 定價的相關資訊，請參閱 Amazon S3 定價。

CloudTrail Lake 事件資料存放區

CloudTrail Lake 讓您能夠對事件執行 SQL 型查詢。CloudTrail Lake 會將分列式 JSON 格式的現有事件轉換為 Apache ORC 格式。ORC 是一種單欄式儲存格式，針對快速擷取資料進行了最佳化。系統會將事件彙總到事件資料存放區中，事件資料存放區是事件的不可變集合，其依據為您透過套用進階事件選取器選取的條件。套用於事件資料存放區的選取器控制哪些事件持續存在並可供您查詢。如需 CloudTrail Lake 的詳細資訊，請參閱AWS CloudTrail 《 使用者指南》中的使用 AWS CloudTrail Lake。

CloudTrail Lake 事件資料存放區和查詢會產生費用。建立事件資料存放區時，您可以選擇要用於事件資料存放區的定價選項。此定價選項將決定擷取和儲存事件的成本，以及事件資料存放區的預設和最長保留期。如需 CloudTrail 定價的詳細資訊，請參閱 AWS CloudTrail 定價。

CloudTrail 中的 Systems Manager 資料事件

資料事件提供有關在資源上或在資源中執行的資源操作的資訊 (例如，建立或開啟控制通道)。這些也稱為資料平面操作。資料事件通常是大量資料的活動。根據預設，CloudTrail 不會記錄資料事件。CloudTrail 事件歷史記錄不會記錄資料事件。

資料事件需支付額外的費用。如需 CloudTrail 定價的詳細資訊，請參閱 AWS CloudTrail 定價。

您可以使用 CloudTrail 主控台或 CloudTrail API 操作 AWS CLI來記錄 Systems Manager 資源類型的資料事件。如需有關如何記錄資料事件的詳細資訊，請參閱《AWS CloudTrail 使用者指南》中的 Logging data events with the AWS Management Console 和 Logging data events with the AWS Command Line Interface。

下表列出了您可以記錄資料事件的 Systems Manager 資源類型。資料事件類型 (主控台) 欄位顯示從 CloudTrail 主控台上的資料事件類型清單中選擇的值。resources.type 值欄會顯示值，您會在使用

AWS CLI 或 CloudTrail APIs 設定進階事件選取器時指定該`resources.type`值。記錄到 CloudTrail 的資料 API 資料行會針對資源類型顯示記錄到 CloudTrail 的 API 呼叫。

資料事件類型 (主控台)	resources.type 值	記錄到 CloudTrail 的資料 API
Systems Manager	`AWS::SSMMessages::ControlChannel`	• `CreateControlChannel` • `OpenControlChannel` 如需有關這些操作的詳細資訊，請參閱《服務授權參考》中的 Actions defined by Amazon Message Gateway Service。
Systems Manager 受管節點	`AWS::SSM::ManagedNode`	• `RequestManagedInstanceRoleToken` – 當在 Systems Manager 管理的節點上執行的 AWS Systems Manager 代理程式 (SSM Agent) 從 Systems Manager 登入資料服務請求登入資料時，就會產生此事件。 如需有關 `RequestManagedInstanceRoleToken` 操作的詳細資訊，請參閱 使用硬體指紋驗證啟用混合模式機器

您可以設定進階事件選取器來篩選 `eventName`、`readOnly` 和 `resources.ARN` 欄位，以僅記錄對您重要的事件。如需有關這些欄位的詳細資訊，請參閱《AWS CloudTrail API 參考》中的 AdvancedFieldSelector。

CloudTrail 中的 Systems Manager 管理事件

管理事件提供在 資源上執行的管理操作的相關資訊 AWS 帳戶。這些也稱為控制平面操作。依預設，CloudTrail 會記錄管理事件。

Systems Manager 會將所有控制平面操作以管理事件的形式記錄到 CloudTrail。Systems Manager API 操作記載於《AWS Systems Manager API 參考》https://docs.aws.amazon.com/systems-manager/latest/APIReference/Welcome.html中。例如，對 `CreateMaintenanceWindows`、`PutInventory`、`SendCommand` 以及 `StartSession` 動作發出的呼叫會在 CloudTrail 日誌檔案中產生項目。如需設定 CloudTrail 以監控 Systems Manager API 呼叫的範例，請參閱 使用 Amazon EventBridge (主控台) 監控工作階段活動。

Systems Manager 事件範例

一個事件代表任何來源提出的單一請求，並包含請求 API 操作的相關資訊、操作的日期和時間、請求參數等等。CloudTrail 日誌檔案並非依公有 API 呼叫的堆疊追蹤排序，因此事件不會以任何特定順序出現。

範例：

- 管理事件範例
- 資料事件範例

管理事件範例

範例 1：**`DeleteDocument`**

以下範例顯示的是對美國東部 (俄亥俄) 區域 (us-east-2) 中名為 `example-Document` 的文件進行 `DeleteDocument` 操作的 CloudTrail 事件。

```
{
    "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"
}
```

範例 2：**StartConnection**

以下範例顯示適用於在美國東部 (俄亥俄) 區域 (us-east-2) 使用 Fleet Manager 啟動 RDP 連線之使用者的 CloudTrail 事件。基礎 API 動作是 `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"
}
```

資料事件範例

範例 1：**CreateControlChannel**

以下範例顯示的 CloudTrail 事件會示範 `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"
}
```

範例 2：**RequestManagedInstanceRoleToken**

以下範例顯示的 CloudTrail 事件會示範 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"
}
```


如需有關 CloudTrail 記錄內容的資訊，請參閱《AWS CloudTrail 使用者指南》中的 CloudTrail record contents。

使用 CloudWatch Logs 記錄自動化動作輸出

Automation 是 中的工具 AWS Systems Manager，可與 Amazon CloudWatch Logs 整合。您可以將您 Runbook 中的 `aws:executeScript` 動作輸出傳送到您指定的日誌群組。Systems Manager 不會建立日誌群組，或者任何文件的日誌串流不會使用 `aws:executeScript` 動作。如果文件確實使用 `aws:executeScript`，則傳送至 CloudWatch Logs 的輸出僅與這些動作有關。您可以使用存放在您 CloudWatch Logs 日誌群組中的 `aws:executeScript` 動作輸出，以進行偵錯和故障診斷。如果您選

擇已加密的記錄群組，aws:executeScript 動作輸出也會加密。aws:executeScript 動作的日誌記錄輸出是帳戶層級的設定。

若要將動作輸出傳送到 CloudWatch Logs，用於 Amazon 擁有的執行手冊，執行自動化的使用者或角色必須具有以下操作許可：

- logs:CreateLogGroup
- logs:CreateLogStream
- logs:DescribeLogGroups
- logs:DescribeLogStreams
- logs:PutLogEvents

對於您擁有的執行手冊，必須將相同的許可新增至您用來執行執行手冊的 IAM 服務角色 (或 AssumeRole)。

若要將動作輸出傳送至 CloudWatch Logs (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Automation (自動化)。
3. 選擇 Preferences (偏好) 標籤，然後選擇 Edit (編輯)。
4. 選取 Send output to CloudWatch Logs (將輸出傳送至 CloudWatch Logs) 旁的核取方塊。
5. (建議) 選取 Encrypt log data (加密日誌資料) 旁的核取方塊。開啟此選項後，系統會使用為該日誌群組指定的伺服器端加密金鑰來加密日誌資料。如果您不想加密要傳送到 CloudWatch Logs 中的日誌資料，清除核取方塊。如果日誌群組不允許加密，清除核取方塊。
6. 對於 CloudWatch Logs 日誌群組，若要指定您要將動作輸出傳送到 AWS 帳戶 的 中現有的 CloudWatch Logs 日誌群組，請選取下列其中一項：
   - Send output to the default log group (將輸出傳送至預設記錄群組) – 如果預設日誌群組不存在 (/aws/ssm/automation/executeScript)，自動化會為您建立一個。
   - Choose from a list of log groups (從日誌群組清單中選擇) – 選取已在您的帳戶中建立的日誌群組，以存放動作輸出。
   - Enter a log group name (輸入日誌群組名稱) – 在文字方塊中，輸入已在帳戶中建立的日誌群組名稱，以存放動作輸出。
7. 選擇 Save (儲存)。

若要將動作輸出傳送至 CloudWatch Logs (命令列)

1. 開啟您偏好的命令列工具，然後執行以下命令來更新動作輸出目的地。

   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"
   ```

   如果命令成功，則無輸出訊息。

2. 執行以下命令，指定您要傳送動作輸出至的日誌群組。

   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"
```

如果命令成功，則無輸出訊息。

3. 執行下列命令，以檢視目前 AWS 帳戶 和 中自動化動作記錄偏好設定的目前服務設定 AWS 區域。

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"
```

該命令會傳回相關資訊，如以下所示。

```
{
    "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"
    }
}
```

設定 Run Command 的 Amazon CloudWatch Logs

當您使用 中的Run Command工具 傳送命令時 AWS Systems Manager，您可以指定要傳送命令輸出的位置。在預設情況下，Systems Manager 只會傳回命令輸出的前 24,000 個字元。如果您想要檢視命令輸出的完整詳細資訊，您可以指定 Amazon Simple Storage Service (Amazon S3) 儲存貯體。或者您可以指定 Amazon CloudWatch Logs。如果您指定 CloudWatch Logs，Run Command 會定期將所有命令輸出和錯誤日誌傳送至 CloudWatch Logs。您可以以幾乎即時的方式監控輸出日誌、搜尋特定字詞、數值或模式，並根據搜尋建立警示。

如果您將受管節點設定為使用 AWS Identity and Access Management (IAM) 受管政策和 `AmazonSSMManagedInstanceCore` `CloudWatchAgentServerPolicy`，則節點不需要額外的組態，即可將輸出傳送至 CloudWatch Logs。如果從主控台傳送命令，請選擇此選項；如果使用 AWS Command Line Interface (AWS CLI) AWS Tools for Windows PowerShell、 或 API 操作，請選擇 `cloud-watch-output-config`區段和 `CloudWatchOutputEnabled` 參數。本主題稍後會更詳細地說明 `cloud-watch-output-config` 區段和 `CloudWatchOutputEnabled` 參數。

如需有關將政策新增至 EC2 執行個體執行個體設定檔的資訊，請參閱設定 Systems Manager 所需的執行個體許可。如需有關將政策新增到您計劃作為受管節點使用的內部部署伺服器和虛擬機器的服務角色的資訊，請參閱在混合多雲端環境中建立 Systems Manager 所需的 IAM 服務角色。

若您正在節點上使用自訂政策，則須在每個節點上更新政策，允許 Systems Manager 將輸出和日誌傳送到 CloudWatch Logs。將以下政策物件新增至您的自訂政策。如需更新 IAM 政策的詳細資訊，請參閱《IAM 使用者指南》中的編輯 IAM 政策。

```
{
    "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/*"
},
```

傳送命令時指定 CloudWatch Logs

若要在從 傳送命令時指定 CloudWatch Logs 做為輸出 AWS Management Console，請在輸出選項區段中選擇 CloudWatch Output。或者，您可以指定 CloudWatch Logs 群組的名稱，您想要將命令輸出傳送到該群組中。如果您不指定群組名稱，Systems Manager 會自動為您建立一個日誌群組。日誌群組使用以下命名格式：`/aws/ssm/`*`SystemsManagerDocumentName`*

如果您使用 執行命令 AWS CLI，請在命令中指定 `cloud-watch-output-config`區段。此區段允許您指定 `CloudWatchOutputEnabled` 參數與 (選用) `CloudWatchLogGroupName` 參數。請見此處範例。

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 中檢視命令輸出

一旦命令開始執行，Systems Manager 會以近乎即時的方式將輸出傳送至 CloudWatch Logs。CloudWatch Logs 中的輸出使用下列格式：

CommandID/*InstanceID*/*PluginID*/stdout

CommandID/*InstanceID*/*PluginID*/stderr

每 30 秒或當緩衝區超過 200 KB 時 (視何者先發生)，即會從執行上傳輸出。

> **Note**
>
> 只有當輸出資料可供使用時，才會建立日誌串流。例如，如果執行沒有錯誤的資料，則不會建立 stderr 串流。

此為當命令輸出出現在 CloudWatch Logs 中時的範例。

```
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 監控 Systems Manager

Amazon EventBridge 為無伺服器匯流排服務，可讓您輕鬆將應用程式與來自各種來源的資料互相連線。EventBridge 可從您自己的應用程式、software-as-a-service (SaaS) 應用程式提供即時資料串流，AWS 服務 並將該資料路由至 等目標 AWS Lambda。您可設定路由規則來決定要將資料送往何處，以便建立即時對您所有資料來源做出反應的應用程式架構。EventBridge 可讓您建置鬆耦合和分散式的事件驅動架構。

EventBridge 之前被稱為 Amazon CloudWatch Events。EventBridge 包含的新功能可允許您從 SaaS 合作夥伴和自己的應用程式接收事件。現有 CloudWatch Events 使用者可以在新的 EventBridge 主控台和 CloudWatch Events 主控台中存取其現有的預設匯流排、規則和事件。EventBridge 使用相同的 CloudWatch Events API，因此您的所有現有 CloudWatch Events API 用量保持不變。

EventBridge 可以將數十個 的事件新增至 AWS 服務 您的規則，以及超過 20 個 的目標 AWS 服務。

EventBridge 支援 AWS Systems Manager 事件和 Systems Manager 目標。

支援的 Systems Manager 事件類型

EventBridge 可以偵測到的許多類型的 Systems Manager 事件包括：

- 手動核准的just-in-time節點存取請求狀態更新。
- 失敗just-in-time節點存取請求。
- 正在關閉維護時段。
- 成功完成自動化工作流程。自動化是 中的工具 AWS Systems Manager。
- 受管節點不符合修補程式合規性。
- 正在更新參數值。

EventBridge 支援下列 AWS Systems Manager 工具的事件：

- Just-in-time節點存取 （ 事件會盡最大努力發出。）
- 自動化 (盡可能發出事件。)
- Change Calendar (盡可能發出事件。)
- 合規
- 庫存 (盡可能發出事件。)
- Maintenance Windows (盡可能發出事件。)
- Parameter Store (盡可能發出事件。)
- Run Command (盡可能發出事件。)
- State Manager (盡可能發出事件。)

如需支援的 Systems Manager 事件類型的完整詳細資訊，請參閱 參考：Systems Manager 的 Amazon EventBridge 事件模式和類型 和 Systems Manager 的 Amazon EventBridge 事件範例。

支援的 Systems Manager 目標類型

EventBridge 支援以下三個 Systems Manager 工具作為事件規則的目標：

- 手動執行自動化工作流程
- 執行 Run Command 命令文件 (盡可能發出事件。)

- 建立 OpsCenter OpsItem

如需建議使用這些目標的方式，請參閱 範例方案：Amazon EventBridge 規則的 Systems Manager 目標。

如需有關如何開始使用 EventBridge 及設定規則的詳細資訊，請參閱《Amazon EventBridge 使用者指南》中的 Amazon EventBridge 入門。如需使用 EventBridge 的完整資訊，請參閱《Amazon EventBridge 使用者指南》。

主題

- 為 Systems Manager 事件設定 EventBridge
- Systems Manager 的 Amazon EventBridge 事件範例
- 範例方案：Amazon EventBridge 規則的 Systems Manager 目標

為 Systems Manager 事件設定 EventBridge

當支援 AWS Systems Manager 的狀態變更、狀態變更或其他條件發生時，您可以使用 Amazon EventBridge 來執行目標事件。您可以建立規則，在有狀態轉換或有轉移到一個或多個與您相關的狀態時執行。

下列程序提供建立 EventBridge 規則的一般步驟，以便在 Systems Manager 發出指定的事件時進行。如需本使用使用者指南中解決特定案例的程序清單，請參閱本主題末尾的詳細資訊。

> **Note**
>
> 當您 AWS 帳戶 中的服務發出事件時，一律會進入您帳戶的預設事件匯流排。若要從您帳戶中的 AWS 服務 編寫回應事件的規則，則須將它與預設事件匯流排建立關聯。您可以在尋找來自 事件的自訂事件匯流排上建立規則 AWS 服務，但此規則只會在您透過跨帳戶事件交付從另一個帳戶接收此類事件時才會啟用。如需詳細資訊，請參閱《Amazon EventBridge 使用者指南》中的在 AWS 帳戶之間傳送和接收 Amazon EventBridge 事件。

若要為 Systems Manager 事件設定 EventBridge

1. 前往 https://console.aws.amazon.com/events/ 開啟 Amazon EventBridge 主控台。
2. 在導覽窗格中，選擇規則。
3. 選擇建立規則。

4. 輸入規則的名稱和描述。

   規則不能與相同 AWS 區域 和相同事件匯流排上的另一個規則具有相同的名稱。

5. 針對事件匯流排，選擇要與此規則建立關聯的事件匯流排。如果您希望此規則回應來自您自己的相符事件 AWS 帳戶，請選取預設值。當您 AWS 服務 帳戶中的 發出事件時，一律會傳送至您帳戶的預設事件匯流排。
6. 針對規則類型，選擇具有事件模式的規則。
7. 選擇下一步。
8. 在事件來源欄位中，選擇 AWS 事件或 EventBridge 合作夥伴事件。
9. 在 Event pattern (事件模式) 區段中，選擇 Event pattern form (事件模式表單)。
10. 在事件來源欄位中，選擇 AWS 服務。
11. 針對 AWS service (服務)，請選擇 Systems Manager。
12. 針對 Event type (事件類型)，執行下列其中一項操作：

    - 選擇 All Events (所有事件)。

      如果您選擇 All Events (所有事件)，此 Systems Manager 服務發出的所有事件都將符合規則。請注意，此選項可能會導致許多事件目標動作。

    - 選擇要用於此規則的 Systems Manager 事件類型。EventBridge 支援來自下列 AWS Systems Manager 工具的事件：
      - 自動化
      - Change Calendar
      - 合規
      - 庫存
      - Maintenance Windows
      - Parameter Store
      - Run Command
      - State Manager

      > **Note**
      >
      > 對於 EventBridge 不支援的 Systems Manager 動作，您可以透過 CloudTrail 選擇 AWS API 呼叫，進而建立以 API 呼叫為基礎的事件規則 (該規則由 CloudTrail 記錄)。如需範例，請參閱「使用 Amazon EventBridge (主控台) 監控工作階段活動」。

13. (選用) 若要使規則更具體，請新增篩選條件值。例如，如果您選擇 State Manager，並希望將規則限制為關聯所針對的單一受管執行個體狀態，則針對 Specific type(s) (特定類型)，選擇 EC2 State Manager Instance Association State Change (EC2 狀態管理員執行個體關聯狀態變更)。

    如需受支援詳細資訊類型的完整詳細資訊，請參閱 參考：Systems Manager 的 Amazon EventBridge 事件模式和類型。

    某些詳細資訊類型具有其他支援的選項，如狀態。可用的選項取決於您選擇的工具。

14. 選擇 Next (下一步)。
15. 在目標類型欄位中，選擇 AWS 服務。
16. 針對選取目標，選擇目標，例如 Amazon SNS 主題或 AWS Lambda 函數。當接收到符合規則中定義之事件模式的事件時，就會觸發目標。
17. 對於許多目標類型而言，EventBridge 需要許可才能將事件傳送到目標。在這些情況下，EventBridge 可以建立規則執行所需的 AWS Identity and Access Management (IAM) 角色：
    - 如需自動建立 IAM 角色，請選擇 為此特定資源建立新角色。
    - 如需使用您早前建立的 IAM 角色，請選擇 使用現有角色 。
18. (選用) 選擇新增其他目標，為此規則新增另一個目標。
19. 選擇下一步。
20. (選用) 為規則輸入一或多個標籤。如需詳細資訊，請參閱《Amazon EventBridge 使用者指南》中的 Amazon EventBridge 標籤。
21. 選擇下一步。
22. 檢閱規則的詳細資訊，然後選擇建立規則。

詳細資訊

- 建立使用 Runbook (主控台) 的 EventBridge 事件
- 使用輸入轉換器將資料傳遞至 Automation
- 使用 EventBridge 修正合規問題
- 在 EventBridge 中檢視清查刪除操作
- 設定 EventBridge 規則以建立 OpsItems
- 為參數及參數政策設定 EventBridge

Systems Manager 的 Amazon EventBridge 事件範例

以下是支援的 EventBridge 事件的 JSON 格式範例 AWS Systems Manager。

Systems Manager 事件類型

- AWS Systems Manager 自動化事件
- AWS Systems Manager事件Change Calendar
- AWS Systems Manager事件Change Manager
- AWS Systems Manager 合規事件
- 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 自動化事件

自動步驟狀態 - 變更通知

```
{
  "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"
  }
}
```

自動執行狀態 - 變更通知

```
{
  "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

使用本主題中的資訊來規劃和了解 EventBridge 事件的行為 AWS Systems ManagerChange Calendar。

> **Note**
>
> AWS 帳戶 目前不支援從其他 共用行事曆的狀態變更。

Change Calendar 與 Amazon EventBridge 整合

AWS Systems Manager Change Calendar 與 Amazon EventBridge 整合，以通知您行事曆狀態變更。請注意與基礎排程架構相關的下列行為：

事件計時和可靠性

- EventBridge 會盡最大努力提供通知，排程公差長達 15 分鐘。
- 狀態變更事件反映整體行事曆狀態轉換，而非個別行事曆事件。
- 當同時發生多個行事曆事件時，EventBridge 只會為每個實際行事曆狀態變更產生一個事件。
- EventBridge 只會在行事曆的整體狀態轉換（例如，從 CLOSED 轉換至 OPEN) 時觸發事件，而不會針對不會導致狀態變更的個別行事曆事件觸發事件。
- 不修改行事曆狀態的諮詢事件不會觸發 EventBridge 通知。

事件修改和時間考量

- 如果您在排程的開始或結束時間的 15 分鐘內修改行事曆事件，EventBridge 可能會產生重複的通知或遺漏的通知。
- 此行為是因為排程系統可能沒有足夠的時間正確更新或取消先前排定的通知。
- 對於週期性事件，此行為通常只會影響修改後第一次出現的情況。

相鄰和重疊的事件

- 當行事曆事件排程在彼此的 5 分鐘內時，狀態轉換事件可能會也可能不會發生，視實際狀態變更而定。
- 即使沒有發生實際狀態變更，在特定順序中建立重疊事件仍可能會產生額外的 EventBridge 事件。
- 為了確保可預測的行為，請避免建立或修改接近其執行時間的行事曆事件。

最佳實務

- 設計您的 EventBridge 規則和下游自動化，以處理潛在的重複事件。
- 在自動化工作流程中實作冪等性，以防止重複的通知發生問題。
- 建立或修改行事曆事件時，請預留足夠的前置時間（至少 15 分鐘）。
- 使用特定行事曆事件模式完整測試 EventBridge 整合。

行事曆開啟

```
{
    "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"
    }
}
```

行事曆關閉

```
{
    "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

變更請求狀態更新通知 - 範例 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"
    ]
  }
}
```

變更請求狀態更新通知 - 範例 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 合規事件

以下是 AWS Systems Manager 合規事件的範例。

關聯相容

```
{
  "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"
  }
}
```

關聯不相容

```
{
  "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"
  }
}
```

修補程式相容

```
{
  "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"
  }
```

```
}
```

修補程式不相容

```
{
  "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

以下為 Systems Manager Maintenance Windows 的事件範例。

登錄一個目標

有效狀態值包括 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"
   }
}
```

視窗執行類型

有效狀態值包括 PENDING、IN_PROGRESS、SUCCESS、TIMED_OUT、 FAILED和 SKIPPED_OVERLAPPING。

```
{
   "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"
   }
}
```

任務執行類型

有效狀態值包括 IN_PROGRESS、FAILED、 SUCCESS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"
   }
}
```

任務目標已處理

有效狀態值包括 IN_PROGRESS、FAILED、 SUCCESS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-1234567890123456789"
   ],
   "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"
   }
}
```

視窗狀態變更

有效狀態值包括 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

以下為 Systems Manager Parameter Store 的事件範例。

建立參數

```
{
  "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"
  }
}
```

更新參數

```
{
  "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"
  }
}
```

刪除參數

```
{
  "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 建立通知

```
{
  "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 狀態 - 變更通知

```
{
    "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 叫用狀態 - 變更通知

```
{
```

```
    "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 關聯狀態變更

```
{
   "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 執行個體關聯狀態變更

```
{
   "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"
   }
}
```

範例方案：Amazon EventBridge 規則的 Systems Manager 目標

當您指定要在 Amazon EventBridge 規則中叫用的目標時，您可以從超過 20 種目標類型中進行選擇，並在每個規則中新增最多五個目標。

在各種目標中，您可以從 Automation、 OpsCenter和 中選擇Run Command，這些是 中的工具 AWS Systems Manager，作為 EventBridge 事件發生時的目標動作。

以下是幾個範例，說明您可以使用這些工具做為 EventBridge 規則的目標。

Automation 範例

您可以設定 EventBridge 規則，以在發生下列事件時啟動 Automation 工作流程：

- 當 Amazon CloudWatch 警示報告受管節點未通過狀態檢查 (`StatusCheckFailed_Instance=1`) 時，執行節點上的 `AWSSupport-ExecuteEC2Rescue` Automation Runbook。
- 當 `EC2 Instance State-change Notification` 事件發生時，由於新的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體正在執行，請執行執行個體上的 `AWS-AttachEBSVolume` Automation Runbook。
- 當建立 Amazon Elastic Block Store (Amazon EBS) 磁碟區且其可用時，執行磁碟區上的 `AWS-CreateSnapshot` Automation Runbook。

OpsCenter 範例

當發生諸如下列事件時，您可以設定 EventBridge 規則來建立新的 OpsItem：

- 發生 Amazon DynamoDB 的調節事件，或 Amazon EBS 磁碟區效能下降。
- Amazon EC2 Auto Scaling 群組無法啟動節點，或者 Systems Manager Automation 工作流程失敗。
- EC2 執行個體將狀態從 `Running` 變更為 `Stopped`。

Run Command 範例

當發生諸如下列事件時，您可以設定 EventBridge 規則來執行 Run Command中的 Systems Manager 命令文件：

- 當 Auto Scaling 群組即將結束時，Run Command 指令碼可以在其結束之前從節點擷取日誌檔案。
- 在 Auto Scaling 群組中建立新節點時，Run Command 目標動作可開啟 Web 伺服器角色或在節點上安裝軟體。
- 當發現受管節點不符合規範時，透過執行 `AWS-RunPatchBaseline` 文件，Run Command 目標動作可更新節點上的修補程式。

使用 Amazon SNS 通知監控 Systems Manager 狀態變更

您可以設定 Amazon Simple Notification Service (Amazon SNS) 傳送您使用 Run Command或 傳送之命令狀態的通知Maintenance Windows，這些是 中的工具 AWS Systems Manager。Amazon SNS 會協調和管理傳送和傳遞通知給已訂閱 Amazon SNS 主題的用戶端或端點。每當命令變更為新狀態或特定狀態時 (如「失敗」或「逾時」)，您都可以收到通知。當您將命令傳送至多個節點時，您都可以接收到傳送到特定節點之每個命令複本的通知。每個複本稱為呼叫。

Amazon SNS 可以使用 HTTP 或 HTTPS POST、電子郵件 (SMTP，純文字或 JSON 格式) 或發佈到 Amazon Simple Queue Service (Amazon SQS) 佇列訊息的方式，傳遞通知。如需詳細資訊，請參閱《Amazon Simple Notification Service 開發人員指南》中的什麼是 Amazon SNS。如需 Run Command 和 Maintenance Windows 提供之 Amazon SNS 通知中所含 JSON 資料結構的範例，請參閱 AWS Systems Manager的 Amazon SNS 通知範例。

> **Important**
>
> 記下以下重要資訊。
>
> - 不支援 Amazon Simple Notification Service FIFO 主題。
> - 聊天應用程式中的 Amazon Q Developer 不支援使用 Amazon SNS 監控 Systems Manager。如果您想要在聊天應用程式中使用 Amazon Q Developer 來監控 Systems Manager，您必須將其與 Amazon EventBridge 搭配使用。如需使用 EventBridge 監控 Systems Manager 的資訊，請參閱 使用 Amazon EventBridge 監控 Systems Manager。如需聊天應用程式中 Amazon EventBridge 和 Amazon Q Developer 的相關資訊，請參閱聊天應用程式管理員指南中的教學課程：建立 EventBridge 規則，以在聊天應用程式中將通知傳送給 Amazon Q Developer。

設定 AWS Systems Manager的 Amazon SNS 通知

向維護時段註冊的 Run Command 和 Maintenance Windows 任務可以針對已進入下列狀態的命令任務傳送 Amazon SNS 通知。

- 進行中
- 成功
- 失敗
- 逾時

- 已取消

如需導致命令進入這些其中一種狀態之條件的相關資訊，請參閱 了解命令狀態。

> **Note**
>
> 使用 Run Command 傳送的命令也會報告 Canceling 和 Pending 狀態。Amazon SNS 通知不會擷取這些狀態。

命令摘要 Amazon SNS 通知

如果您在 Amazon SNS 通知的維護時段中設定 Run Command 或 Run Command 任務，Amazon SNS 會傳送包含以下資訊的摘要訊息。

欄位	Type	描述
eventTime	字串	事件啟動的時間。此時間戳記非常重要，因為 Amazon SNS 不保證訊息交付的順序。範例：2016-04-26T13:15:30Z
documentName	字串	用來執行此命令之 SSM 文件名稱。
commandId	字串	在傳送命令後 Run Command 產生的 ID。
expiresAfter	日期	如果已達到此時間但系統尚未開始執行命令，則系統不會執行該命令。
outputS3BucketName	字串	Amazon Simple Storage Service (Amazon S3) 儲存貯體，命令執行的回應應存放的位置。
outputS3KeyPrefix	字串	儲存貯體中的 Amazon Simple Storage Service (Amazon S3)

欄位	Type	描述
		目錄路徑，命令執行的回應應存放的位置。
requestedDateTime	字串	將請求傳送到此特定節點的時間和日期。
instanceIds	StringList	被命令視為目標的節點。 **Note** 只有在Run Command任務直接以執行個體 ID 為目標時，摘要訊息才會包含執行個體 ID。如果是使用標籤型目標鎖定來發出 Run Command 任務，則摘要訊息不會包含執行個體 ID。
status	字串	命令的命令狀態。

以叫用為基礎的 Amazon SNS 通知

如果您將命令傳送到多個節點，Amazon SNS 可以傳送與每個命令副本或叫用相關的訊息。此訊息包含下列資訊。

欄位	Type	描述
eventTime	字串	事件啟動的時間。此時間戳記非常重要，因為 Amazon SNS 不保證訊息交付的順序。範例：2016-04-26T13:15:30Z

欄位	Type	描述
documentName	字串	用來執行此命令之 Systems Manager 文件 (SSM 文件) 名稱。
requestedDateTime	字串	將請求傳送到此特定節點的時間和日期。
commandId	字串	在傳送命令後 Run Command 產生的 ID。
instanceId	字串	被命令視為目標的執行個體。
status	字串	此呼叫的命令狀態。

若要設定當命令變更狀態時的 Amazon SNS 通知，則須先完成下列任務。

> **Note**
>
> 如未針對維護時段設定 Amazon SNS 通知，則可略過本主題稍後的任務 5。

主題

- 任務 1：建立並訂閱 Amazon SNS 主題
- 任務 2：為 Amazon SNS 通知建立 IAM 政策
- 任務 3：為 Amazon SNS 通知建立 IAM 角色
- 任務 4：設定使用者存取
- 任務 5：將 iam:PassRole 政策連接至維護時段角色

任務 1：建立並訂閱 Amazon SNS 主題

Amazon SNS 主題 是一個通訊通道，向維護時段註冊的 Run Command 和 Run Command 任務使用此通道來傳送命令狀態的相關通知。Amazon SNS 支援不同的通訊協定，包括 HTTP/S、電子郵件和其他 AWS 服務 Amazon Simple Queue Service (Amazon SQS)。為了入門，我們建議您先從電子郵件通訊協定開始。如需建立主題的詳細資訊，請參閱《Amazon Simple Notification Service 開發人員指南》中的建立 Amazon SNS 主題。

> **Note**
>
> 建立主題後，請複製或記下 Topic ARN (主題 ARN)。當您傳送設定為傳回狀態通知的命令時，您會指定此 ARN。

建立主題後，透過指定 Endpoint (端點) 來進行訂閱。如果您選擇電子郵件通訊協定，該端點是您想要從中接收到通知的電子郵件地址。如需如何訂閱主題的詳細資訊，請參閱《Amazon Simple Notification Service 開發人員指南》中的訂閱 Amazon SNS 主題。

Amazon SNS 會從 AWS 通知傳送確認電子郵件到您所指定的電子郵件地址。開啟電子郵件，然後選擇 Confirm subscription (確認訂閱) 連結。

您會收到來自 的確認訊息 AWS。Amazon SNS 現在已設定為採用您指定的電子郵件地址，以電子郵件方式接收和傳送通知。

任務 2：為 Amazon SNS 通知建立 IAM 政策

使用下列程序建立自訂 AWS Identity and Access Management (IAM) 政策，提供啟動 Amazon SNS 通知的許可。

若要建立 Amazon SNS 通知的自訂 IAM 政策

1. 在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。
2. 在導覽窗格中，選擇 Policies (政策)，然後選擇 Create Policy (建立政策)。(顯示 Get Started (開始使用) 按鈕時先選擇它，然後選擇 Create Policy (建立政策)。)
3. 選擇 JSON 標籤。
4. 根據 Amazon SNS 主題是否使用 AWS KMS 加密，將預設內容取代為下列其中一項：

   SNS topic not encrypted

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sns:Publish"
            ],
            "Resource": "arn:aws:sns:region:account-id:sns-topic-name"
```

```
        }
    ]
}
```

region 代表 支援的 AWS 區域 識別符 AWS Systems Manager，例如us-east-2美國東部（俄亥俄）區域。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域欄位。

account-id 代表 的 12 位數識別符 AWS 帳戶，格式為 123456789012。

sns-topic-name 代表您想要用於發佈通知的 Amazon SNS 主題名稱。

SNS topic encrypted

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sns:Publish"
            ],
            "Resource": "arn:aws:sns:region:account-id:sns-topic-name"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "arn:aws:kms:region:account-id:key/kms-key-id"
        }
    ]
}
```

region 代表 支援的 AWS 區域 識別符 AWS Systems Manager，例如us-east-2美國東部（俄亥俄）區域。如需支援的 *region* 值的清單，請參閱《Amazon Web Services 一般參考》中 Systems Manager 服務端點一節的區域欄位。

account-id 代表 的 12 位數識別符 AWS 帳戶，格式為 123456789012。

sns-topic-name 代表您想要用於發佈通知的 Amazon SNS 主題名稱。

kms-key-id 代表 中對稱加密 KMS 金鑰的 ID， AWS KMS 用於加密和解密主題，格式為 1234abcd-12ab-34cd-56ef-12345EXAMPLE。

> **Note**
>
> 使用 AWS KMS 加密需要付費。如需詳細資訊，請參閱《 AWS Key Management Service 開發人員指南》中的管理 Amazon SNS 加密金鑰和成本。

5. 選擇下一步：標籤。
6. (選用) 新增一個或多個標籤鍵值組來組織、追蹤或控制對此政策的存取。
7. 選擇下一步：檢閱。
8. 在檢閱政策頁面上名稱中，輸入該內嵌政策的名稱。例如：**my-sns-publish-permissions**。
9. (選用) 在說明中輸入政策的說明。
10. 選擇建立政策。

任務 3：為 Amazon SNS 通知建立 IAM 角色

請遵循以下程序，為 Amazon SNS 通知建立 IAM 角色。Systems Manager 會使用此服務角色來啟動 Amazon SNS 通知。在後續的所有程序中，此角色就是所謂的 Amazon SNS IAM 角色。

若要為 Amazon SNS 通知建立 IAM 服務角色

1. 登入 AWS Management Console ，並在 https：//https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在 IAM 主控台的導覽窗格中，選擇角色，然後選擇建立角色。
3. 選擇 AWS 服務 角色類型，然後選擇 Systems Manager。
4. 選擇 Systems Manager 使用案例。然後選擇下一步。
5. 在 Attached permissions policy (連接的許可政策) 頁面上，選取您在任務 2 中建立之自訂政策名稱左側的核取方塊。例如：**my-sns-publish-permissions**。
6. (選用) 設定許可界限。這是進階功能，可用於服務角色，而不是服務連結的角色。

   展開 Permissions boundary (許可界限) 區段，並選擇 Use a permissions boundary to control the maximum role permissions (使用許可界限來控制角色許可上限)。IAM 包含您帳戶中 AWS 受管和客戶受管政策的清單。選取用於許可界限的政策，或者選擇 Create policy (建立政策) 以開啟新的

瀏覽器標籤，並從頭建立新的政策。如需詳細資訊，請參閱「IAM 使用者指南」中的建立 IAM 政策。在您建立政策後，關閉該標籤並返回您的原始標籤，以選取用於許可界限的政策。

7. 選擇下一步。
8. 如果可能，請輸入角色名稱或角色名稱後綴，以協助您識別此角色的用途。角色名稱在您的 AWS 帳戶內必須是獨一無二的。它們無法透過大小寫進行區分。例如，您無法建立名為 **PRODROLE** 和 **prodrole** 的角色。因為有各種實體可能會參照角色，所以您無法在建立角色之後編輯角色名稱。
9. (選用) 在 Description (說明) 中，輸入新角色的說明。
10. 在 Step 1: Select trusted entities (步驟 1：選取受信任的實體) 或者 Step 2: Select permissions (步驟 2：選取許可) 區段中選擇 Edit (編輯)，可編輯角色的使用案例和許可。
11. (選用) 藉由連接標籤作為鍵值對，將中繼資料新增至使用者。如需有關在 IAM 中使用標籤的詳細資訊，請參閱《IAM 使用者指南》中的標記 IAM 資源。
12. 檢閱角色，然後選擇建立角色。
13. 選擇角色的名稱，然後複製或記下 Role ARN (角色 ARN) 值。當您傳送的命令設定為傳回 Amazon SNS 通知時，就會使用針對此角色的 Amazon Resource Name (ARN)。
14. Summary (摘要) 頁面隨即開啟。

任務 4：設定使用者存取

如果 IAM 實體 （使用者、角色或群組） 獲指派管理員許可，則使用者或角色可以存取 中的 Run Command和 Maintenance Windows工具 AWS Systems Manager。

對於沒有管理員許可的實體，管理員必須將以下許可授予給 IAM 實體：

- `AmazonSSMFullAccess` 受管政策或提供相當許可的政策。
- 在 任務 3：為 Amazon SNS 通知建立 IAM 角色 中建立之角色的 `iam:PassRole` 許可。例如：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::account-id:role/sns-role-name",
            "Condition": {
```

```
                "StringEquals": {
                    "iam:PassedToService": "ssm.amazonaws.com"
                }
            }
        }
    ]
}
```

若要提供存取權，請新增權限至您的使用者、群組或角色：

- 中的使用者和群組 AWS IAM Identity Center：

  建立權限合集。請按照 AWS IAM Identity Center 使用者指南 中的 建立權限合集 說明進行操作。
- 透過身分提供者在 IAM 中管理的使用者：

  建立聯合身分的角色。遵循「IAM 使用者指南」的為第三方身分提供者 (聯合) 建立角色中的指示。
- IAM 使用者：
  - 建立您的使用者可擔任的角色。請按照「IAM 使用者指南」的為 IAM 使用者建立角色中的指示。
  - (不建議) 將政策直接附加至使用者，或將使用者新增至使用者群組。請遵循 IAM 使用者指南的新增許可到使用者 (主控台) 中的指示。

若要設定使用者存取權並將 **iam:PassRole** 政策連接至使用者帳戶

1. 在 IAM 導覽窗格中，選擇 Users (使用者)，接著選擇要設定的使用者帳戶。
2. 在 Permissions (許可) 標籤，於政策清單中，確認 **AmazonSSMFullAccess** 政策已列出，或是有同等的政策能夠給予帳戶存取 Systems Manager 的許可。
3. 選擇 Add inline policy (新增內嵌政策)。
4. 在 Create policy (建立政策) 頁面，選擇 Visual editor (視覺化編輯器) 標籤。
5. 選擇 Choose a service (選擇一個服務)，然後選擇 IAM。
6. 對於 Actions (動作)，在 Filter actions (篩選動作) 文字方塊中輸入 **PassRole**，接著選取 PassRole 旁的核取方塊。
7. 對於 Resources (資源)，確認已選取 Specific (特定)，接著選擇 Add ARN (新增 ARN)。
8. 在 Specify ARN for role (指定角色的 ARN) 欄位中，貼上您在任務 3 結尾複製的 Amazon SNS IAM 角色 ARN。系統會自動填入 Account (帳戶) 和 Role name with path (角色名稱與路徑) 欄位。
9. 選擇 Add (新增)。

10. 選擇 Review policy (檢閱政策)。
11. 在 Review Policy (檢閱政策) 頁面輸入名稱，接著選擇 Create Policy (建立政策)。

任務 5：將 iam:PassRole 政策連接至維護時段角色

當您使用維護時段註冊 Run Command 任務時，您需要指定服務角色 Amazon Resource Name (ARN)。此服務角色是 Systems Manager 用來執行向維護時段註冊的任務。若要為已註冊的 Run Command 任務設定 Amazon SNS 通知，則須將 `iam:PassRole` 政策連接到指定的維護時段服務角色。如果您沒有打算為 Amazon SNS 通知設定已註冊的任務，則您可以略過此任務。

此 `iam:PassRole` 政策允許 Maintenance Windows 服務角色將在任務 3 中建立的 Amazon SNS IAM 角色中傳遞到 Amazon SNS 服務。以下程序說明如何將 `iam:PassRole` 政策連接至 Maintenance Windows 服務角色。

> **Note**
>
> 使用維護時段的自訂服務角色，傳送與已註冊 Run Command 任務相關的通知。如需相關資訊，請參閱設定 Maintenance Windows。
> 如果您需要制定用於維護時段任務的自訂服務角色，請參閱 設定 Maintenance Windows。

若要將 **`iam:PassRole`** 政策連接至 Maintenance Windows 角色。

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 Roles (角色) 並選取任務 3 中建立的 Amazon SNS IAM 角色。
3. 複製或記下 Role ARN (角色 ARN)，並返回 IAM 主控台的 Roles (角色) 區段。
4. 從 Role name (角色名稱) 清單中，選取您建立的自訂 Maintenance Windows 服務角色。
5. 在 Permissions (許可) 索引標籤中，驗證是否列出 `AmazonSSMMaintenanceWindowRole` 政策，或是否有同等的政策能夠將維護時段許可提供給 Systems Manager API。若否，則應選擇新增許可、連接政策，以進行連接。
6. 選擇 Add permissions, Create inline policy (新增許可，建立內嵌政策)。
7. 選擇 Visual Editor (視覺化編輯器) 標籤。
8. 針對 Service (服務)，選擇 IAM (IAM)。
9. 對於 Actions (動作)，在 Filter actions (篩選動作) 文字方塊中輸入 **`PassRole`**，接著選取 PassRole 旁的核取方塊。

10. 針對 Resources (資源)，選擇 Specific (特定)，然後選擇 Add ARN (新增 ARN)。
11. 在 Specify ARN for role (指定角色的 ARN) 方塊中，貼上在任務 3 中建立之 Amazon SNS IAM 角色的 ARN，然後選擇 Add (新增)。
12. 選擇檢閱政策。
13. 在檢閱政策頁面上，提供 `PassRole` 政策的名稱，然後選擇建立政策。

AWS Systems Manager的 Amazon SNS 通知範例

您可以設定 Amazon Simple Notification Service (Amazon SNS) 傳送有關您使用 Run Command或 傳送之命令狀態的通知Maintenance Windows，這些是 中的工具 AWS Systems Manager。

> **Note**
>
> 本指南不會談論如何為 Run Command 或 Maintenance Windows 設定通知。如需將 Run Command 或 Maintenance Windows 設定為傳送與命令狀態相關之 Amazon SNS 通知的詳細資訊，請參閱 設定 AWS Systems Manager的 Amazon SNS 通知。

以下範例示範在為 Run Command 或 Maintenance Windows 進行設定時，Amazon SNS 通知傳回的 JSON 輸出結構。

使用執行個體 ID 的目標鎖定時命令摘要訊息的範例 JSON 輸出

```
{
    "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"
}
```

使用以標籤為基礎之目標鎖定時的命令摘要訊息範例 JSON 輸出

```
{
    "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"
}
```

叫用訊息的範例 JSON 輸出

```
{
    "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傳送命令以傳回狀態通知

下列程序說明如何使用 AWS Command Line Interface (AWS CLI) 或 AWS Systems Manager 主控台，透過 中的工具 傳送命令 AWS Systems Manager，該Run Command工具設定為傳回狀態通知。

傳送Run Command以傳回通知 (主控台)

請使用下列程序，透過已設定為使用 Systems Manager 主控台傳回狀態通知的 Run Command 來傳送命令。

傳送命令以傳回通知 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Run Command。
3. 選擇執行命令。
4. 在 Command document (命令文件) 清單中，選擇 Systems Manager 文件。

5. 在 Command parameters (命令參數) 區段，指定所需的參數值。
6. 在 Targets (目標) 區段中，透過手動指定標籤、選取執行個體或邊緣裝置，或指定資源群組，選擇您要執行這項操作的受管節點。

> **Tip**
>
> 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

7. 對於其他參數：
   - 在 Comment (註解) 中，輸入此命令的相關資訊。
   - 在逾時 (秒) 中，指定在命令執行全面失敗之前，系統要等候的秒數。
8. 對於 Rate control (速率控制)：
   - 在並行中，指定可同時執行命令的受管節點數目或百分比。

> **Note**
>
> 如果透過指定套用至受管節點的標籤或指定 AWS 資源群組選取了目標，且您不確定會以多少個受管節點為目標，則透過指定百分比限制可以同時執行文件之目標的數量。

   - 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比之節點上的命令失敗之後，停止在其他受管節點上執行命令。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止傳送命令。仍在處理命令的受管節點也可能會傳送錯誤。
9. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取將命令輸出寫入至 S3 儲存貯體方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

> **Note**
>
> 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給執行個體之執行個體設定檔 (適用於 EC2 執行個體) 或 IAM 服務角色 (啟用混合模式的機器) 的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確定與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

10. 在 SNS Notifications (SNS 通知) 區段，選擇 Enable SNS notifications (啟用 SNS 通知)。

11. 對於 IAM role (IAM 角色，選擇您在 使用 Amazon SNS 通知監控 Systems Manager 狀態變更 的任務 3 中建立的 Amazon SNS IAM 角色 ARN。
12. 對於 SNS topic (SNS 主題)，輸入要使用的 Amazon SNS 主題 ARN。
13. 對於 Event notifications (事件通知)，選擇您要接收通知的事件。
14. 對於 Change notifications (變更通知)，選擇僅接收命令摘要的通知 (Command status changes (命令狀態變更))，或是傳送到多個節點的每個命令副本 (Command status on each instance changes (每個執行個體的命令狀態變更))。
15. 選擇執行。
16. 檢查從 Amazon SNS 所寄來的電子郵信通知訊息並開啟此電子郵件訊息。Amazon SNS 可能需要幾分鐘的時間來傳送電子郵件訊息。

傳送Run Command以傳回通知 (CLI)

請使用下列程序，透過已設定為使用 AWS CLI傳回狀態通知的 Run Command 來傳送命令。

傳送會傳回通知的命令 (CLI)

1. 開啟 AWS CLI。
2. 在以下命令中指定要根據受管節點 ID 來作為目標的參數。

```
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"}'
```

以下是範例。

```
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

在以下命令中指定要使用標籤來將受管執行個體作為目標的參數。

```
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"}'
```

以下是範例。

```
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。
4. 檢查從 Amazon SNS 所寄來的電子郵信通知訊息並開啟此電子郵件訊息。Amazon SNS 可能需要幾分鐘的時間來傳送電子郵件訊息。

如需詳細資訊，請參閱 AWS CLI 命令參考中的 send-command。

使用維護時段傳送命令以傳回狀態通知

下列程序說明如何使用 AWS Systems Manager 主控台或 AWS Command Line Interface (AWS CLI) 向維護時段註冊Run Command任務。 Run Command 是 中的工具 AWS Systems Manager。此程序也說明如何將 Run Command 任務設定為傳回狀態通知。

開始之前

如果您尚未建立維護時段或註冊目標，請參閱使用主控台建立和管理維護時段，以了解如何建立維護時段和註冊目標的步驟。

若要接收來自 Amazon Simple Notification Service (Amazon SNS) 服務的通知，您必須將 `iam:PassRole` 政策連接到已註冊任務中指定的 Maintenance Windows 服務角色。如果您尚未將 `iam:PassRole` 許可新增至 Maintenance Windows 服務角色，請參閱 任務 5：將 iam:PassRole 政策連接至維護時段角色。

向維護時段註冊Run Command任務以傳回通知 (主控台)

使用下列程序註冊 Run Command 任務，它已設定為使用 Systems Manager 主控台向您的維護時段傳回狀態通知。

向維護時段註冊Run Command任務以傳回通知 (主控台)

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Maintenance Windows。
3. 選取您要為已設定為傳送 Amazon Simple Notification Service (Amazon SNS) 通知的 Run Command 任務進行註冊的維護時段。
4. 選擇 Actions (動作)，然後選擇 Register Run command task (註冊執行命令任務)。
5. (選用) 在 Name (名稱) 欄位中，輸入任務的名稱。
6. (選用) 在 Description (描述) 欄位中，輸入描述。
7. 對於 Command document (命令文件)，選擇一個命令文件。
8. 對於 Task priority (任務優先順序)，請指定此任務的優先順序。零 (0) 是最高的優先順序。維護時段中的任務會依優先順序排程。具有相同優先順序的任務會排程平行處理。
9. 在 Targets (目標) 區段中，選取已註冊的目標群組或選取未註冊的目標。
10. 對於速率控制：
    - 在並行中，指定可同時執行命令的受管節點數目或百分比。

      > **Note**
      >
      > 如果透過指定套用至受管節點的標籤或指定 AWS 資源群組選取了目標，且您不確定會以多少個受管節點為目標，則透過指定百分比限制可以同時執行文件之目標的數量。

    - 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比之節點上的命令失敗之後，停止在其他受管節點上執行命令。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止傳送命令。仍在處理命令的受管節點也可能會傳送錯誤。
11. 在 IAM service role (IAM 服務角色) 區域中，選擇具有 SNS 角色 `iam:PassRole` 許可的 Maintenance Windows 服務角色。

    > **Note**
    >
    > 將 `iam:PassRole` 許可新增至 Maintenance Windows 角色，以允許 Systems Manager 將 SNS 角色 傳遞至 Amazon SNS。如果您尚未新增 `iam:PassRole` 許可，請參閱使用 Amazon SNS 通知監控 Systems Manager 狀態變更主題中的任務 5。

12. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取啟用將輸出寫入 S3 方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

> **Note**
>
> 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給受管節點之執行個體設定檔的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確認與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

13. 在 SNS notifications (SNS 通知) 區段中，執行以下操作：

    - 選擇 Enable SNS Notifications (啟用 SNS 通知)。
    - 對於 IAM role (IAM 角色)，選擇您在 使用 Amazon SNS 通知監控 Systems Manager 狀態變更 中的任務 3 中建立的 Amazon SNS IAM 角色 Amazon Resource Name (ARN)，以啟動 Amazon SNS。
    - 對於 SNS topic (SNS 主題)，輸入要使用的 Amazon SNS 主題 ARN。
    - 對於 Event type (事件類型)，選擇您要接收通知的事件。
    - 對於 Notification type (通知類型)，選擇是要接收傳送到多個節點 (叫用) 的每個命令複本的通知，還是接收命令摘要的通知。

14. 在 Parameters (參數) 區段，根據您選擇的命令文件輸入所需的參數。
15. 選擇 Register run command task (註冊執行命令任務)。
16. 在下一次執行維護時段後，請檢查您的電子郵件是否有來自 Amazon SNS 的訊息並開啟該電子郵件訊息。Amazon SNS 可能需要幾分鐘的時間來傳送該電子郵件訊息。

向維護時段註冊Run Command任務以傳回通知 (CLI)

使用下列程序，來註冊已設定為使用 AWS CLI向維護時段傳回狀態通知的 Run Command 任務。

向會傳回通知的維護時段註冊 Run Command 任務 (CLI)

> **Note**
>
> 為了更有效地管理任務選項，這個程序會將命令選項 `--cli-input-json` 與在 JSON 檔案中存放的選項值搭配使用。

1. 在本機電腦上，建立名為 RunCommandTask.json 的檔案。
2. 將以下內容貼到 檔案。

```
{
    "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. 將範例值取代為您自有資源的相關資訊。

   如果您想要使用在這個範例中省略的選項，也可以將其恢復。例如，您可以將命令輸出儲存到 S3 儲存貯體。

   如需詳細資訊，請參閱 AWS CLI 命令參考中的 register-task-with-maintenance-window。

4. 儲存檔案。
5. 在檔案儲存所在之本機電腦上的目錄中，執行下列命令。

```
aws ssm register-task-with-maintenance-window --cli-input-json file://
RunCommandTask.json
```

> **Important**
>
> 請確認在檔案名稱之前包含 `file://`。這是此命令必要項目。

如果成功，此命令會傳回類似如下的資訊。

```
{
    "WindowTaskId": "j2l8d5b5c-mw66-tk4d-r3g9-1d4d1EXAMPLE"
}
```

6. 在下一次執行維護時段後，請檢查您的電子郵件是否有來自 Amazon SNS 的訊息並開啟該電子郵件訊息。Amazon SNS 可能需要幾分鐘的時間來傳送該電子郵件訊息。

如需透過命令列向維護時段註冊任務的詳細資訊，請參閱向維護時段註冊任務。

與 Systems Manager 的產品和服務整合

根據預設， 會與 AWS 服務 和其他 產品和服務 AWS Systems Manager 整合。下列資訊可協助您設定 Systems Manager 以整合您要使用的產品和服務。

- 與 整合 AWS 服務
- 與其他產品及服務整合

與 整合 AWS 服務

透過使用 Systems Manager Command 文件 (SSM 文件） 和 Automation Runbook，您可以使用 AWS Systems Manager 與 整合 AWS 服務。如需這些資源的詳細資訊，請參閱 AWS Systems Manager Documents。

Systems Manager 已與下列項目整合 AWS 服務。

運算

Amazon Elastic Compute Cloud (Amazon EC2)	Amazon EC2 在 AWS 雲端中提供可擴展的運算容量。使用 Amazon EC2 可減少前期所需的硬體投資，讓您更快速開發並部署應用程式。您可使用 Amazon EC2 按需要啟動任意數量的虛擬伺服器，設定安全性和聯網功能以及管理儲存。 Systems Manager 允許您在 EC2 執行個體上執行數項任務。例如，您可以啟動、設定、管理、維護、故障診斷，以及安全地連線到 EC2 執行個體。您也可以使用 Systems Manager 來部署軟體、判斷合規狀態，以及從 EC2 執行個體收集庫存。 進一步了解 • 使用受管節點 • 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合規 • AWS Systems Manager庫存
Amazon EC2 Auto Scaling	Auto Scaling 能確保您有正確的 EC2 執行個體數量可處理應用程式的負載。您可以建立 EC2 執行個體的集合，此集合稱為「Auto Scaling 群組」。 Systems Manager 允許您自動化常用程序，例如修補在適用於的 Auto Scaling 群組的 Auto Scaling 範本中使用的 Amazon Machine Image(AMI)。 進一步了解 更新 Auto Scaling 群組的 AMIs
Amazon Elastic Container Service (Amazon ECS)	Amazon ECS 是具高可擴展性且快速的容器管理服務，允許您在叢集上執行、停用及管理 Docker 容器。 Systems Manager 可讓您遠端管理容器執行個體，並將敏感資料插入您的容器Parameter Store，方法是將敏感資料存放在 中的參數、Systems Manager 中的工具，然後在容器定義中參考它們。 進一步了解 • 使用 AWS Systems Manager遠端管理容器執行個體 • 使用 Systems Manager Parameter Store 指定敏感資料

AWS Lambda	Lambda 是一項運算服務，允許您執行程式碼，而無需佈建或管理伺服器。Lambda 只有在需要時才會執行程式碼，可自動從每天數項請求擴展成每秒數千項請求。 Systems Manager 允許您透過使用 `aws:invokeLambdaFunction` 動作在自動化 Runbook 內容中使用 Lambda 函數。 若要在 AWS Lambda 函數Parameter Store中使用來自 的參數，您可以使用 AWS Parameters and Secrets Lambda Extension 來擷取參數值並快取它們以供日後使用。 進一步了解 AMI 使用自動化更新黃金 AWS Lambda，以及 Parameter Store 在 AWS Lambda 函數中使用 Parameter Store 參數

物聯網 (IoT)

AWS IoT Greengrass 核心裝置	AWS IoT Greengrass 是開放原始碼 IoT 邊緣執行時間和雲端服務，可協助您在裝置上建置、部署和管理 IoT 應用程式。Systems Manager 為 AWS IoT Greengrass 核心裝置提供原生支援。 進一步了解 使用 Systems Manager 管理邊緣裝置
AWS IoT 核心裝置	AWS IoT 提供將您的 IoT 裝置連接到其他裝置和雲端服務的 AWS 雲端服務。 AWS IoT 提供裝置軟體，可協助您將 IoT 裝置整合到 AWS IoT解決方案。如果您的裝置可以連接到 AWS IoT， AWS IoT 可以將其連接到 AWS 提供的雲

	端服務。Systems Manager 支援 AWS IoT 核心裝置，只要這些裝置在混合多雲端環境中設定為受管節點即可。 進一步了解 使用 Systems Manager 在混合多雲端環境中管理節點

儲存

Amazon Simple Storage Service (Amazon S3)	Amazon Simple Storage Service (Amazon S3) 是針對網際網路的儲存服務。其旨在降低開發人員進行 web 規模運算的難度。Amazon Simple Storage Service (Amazon S3) 具有一個簡單的 Web 服務介面，可讓您隨時從 Web 上的任何位置存放和擷取任意數量的資料。 Systems Manager 可讓您執行存放在 Amazon S3 中的遠端指令碼和 SSM 文件。 Distributor 是 中的工具 AWS Systems Manager，使用 Amazon S3 存放套件。您也可以將輸出傳送至 Amazon S3，適用於 Run Command和 Session Manager中的工具 AWS Systems Manager。 進一步了解 • 從 Amazon Simple Storage Service (Amazon S3) 執行指令碼 • 從遠端位置執行 文件 • AWS Systems Manager Distributor • 使用 Amazon Simple Storage Service (Amazon S3) (主控台) 記錄工作階段資料

開發人員工具

AWS CodeBuild	CodeBuild 是在雲端的全受管建置服務。CodeBuild 可編譯原始碼、執行單元測試，並產生可立即部署的成品。使用 CodeBuild 即可不必佈建、管理、擴展自己的組建伺服器。 Parameter Store 允許您為建置建規格和專案存放敏感資訊。 進一步了解 • CodeBuild 的建置規格參考 • 在 中建立建置專案 AWS CodeBuild
AWS CDK	AWS Cloud Development Kit (AWS CDK) 是將雲端基礎設施定義為程式碼、使用程式設計語言並透過 部署雲端基礎設施的架構 AWS CloudFormation。 Application Manager 允許您檢視分組為應用程式的 CDK 建構模組、檢視應用程式結構 (包含基礎資源)、檢視提醒、調查和修補操作問題，以及在 Application Manager 主控台中追蹤成本。 進一步了解 • 檢視應用程式的概觀資訊 • 檢視應用程式資源

安全性、身分與合規

AWS Identity and Access Management (IAM)	IAM 是一種 Web 服務，可協助您安全地控制對 AWS 資源的存取。您可以使用 IAM 來控制能通過身分驗證 (登入) 和授權使用資源的 (具有許可) 的人員。

	Systems Manager 允許您使用 IAM 控制對服務的存取。 進一步了解 • AWS Systems Manager 搭配 IAM 的運作方式 • 適用於 AWS Systems Manager的動作、資源及條件金鑰 • 設定 Systems Manager 所需的執行個體許可
AWS Secrets Manager	Secrets Manager 可簡化秘密的管理。秘密可能是資料庫憑證、密碼、第三方 API 金鑰，甚至是任意文字。 Parameter Store 允許您在使用其他已支援參考 Parameter Store 參數的 AWS 服務 時，擷取 Secrets Manager 秘密。 進一步了解 從Parameter Store參數參考 AWS Secrets Manager 秘密

AWS Security Hub	Security Hub 可為您提供跨 AWS 帳戶的高優先級安全性警示和合規性狀態的全方位檢視。Security Hub 會從多個 彙總、組織和排定安全提醒或調查結果的優先順序 AWS 服務。 當您開啟 Security Hub 與 中的Patch Manager 工具 之間的整合時 AWS Systems Manager，Security Hub 會從安全角度監控機群的修補狀態。修補程式合規詳細資訊會自動匯出至 Security Hub。這可讓您使用單一檢視來集中監控修補程式合規狀態，並追蹤其他安全調查結果。您可以在機群中的節點違反修補程式合規性時接收警示，並在 Security Hub 主控台中檢閱修補程式合規調查結果。 您也可以將 Security Hub 與 Explorer和 中的 OpsCenter工具整合 AWS Systems Manager。與 Security Hub 整合可讓您在 Explorer 和 OpsCenter 中接收來自 Security Hub 的調查結果。Security Hub 調查結果提供您可在 Explorer 和 OpsCenter 中使用的安全資訊，以對 AWS Systems Manager中的安全、效能和操作問題進行彙總並採取動作。 使用 Security Hub 會產生費用。如需詳細資訊，請參閱 Security Hub 定價。 進一步了解 • 在 Explorer 中從 AWS Security Hub 接收調查結果 • 了解與 的OpsCenter整合 AWS Security Hub • 將 AWS Security Hub 與 Patch Manager 整合

密碼編譯和 PKI

AWS Key Management Service (AWS KMS)	AWS KMS 是一種受管服務，可讓您輕鬆地建立和控制客戶受管金鑰，這是用來加密資料的加密金鑰。 Systems Manager 可讓您使用 AWS KMS 來建立`SecureString` 參數和加密Session Manager工作階段資料。 進一步了解 • AWS KMSAWS Systems ManagerParameter Store SecureString 參數的加密 • 開啟工作階段資料的 KMS 金鑰加密 (主控台)

管理與管控

AWS CloudFormation	AWS CloudFormation 是一項服務，能幫助您模型化與設定 Amazon Web Services 資源，讓您花較少的時間管理這些資源，並且有更多時間專注在執行於 AWS的應用程式上。 Parameter Store 是動態參考的來源。動態參考提供簡潔、強大的方式，讓您指定在 AWS CloudFormation 堆疊範本中其他 服務中存放和管理的外部值。 進一步了解 使用動態參考來指定範本值
AWS CloudTrail	CloudTrail 是 AWS 服務 ，可協助您授權 的控管、合規，以及營運和風險稽核 AWS 帳戶。使用者、角色或 採取的動作 AWS 服務 會在 CloudTrail 中記錄為事件。事件包括在 AWS Management Console、 AWS Command Line

	Interface (AWS CLI) 和 AWS SDKs和 APIs中採取的動作。 Systems Manager 與 CloudTrail 整合，可將大多數 Systems Manager API 呼叫作為事件進行擷取。這包括從 Systems Manager 主控台啟動的 API 呼叫以及對 Systems Manager API 發出的呼叫。 進一步了解 使用 記錄 AWS Systems Manager API 呼叫 AWS CloudTrail
Amazon CloudWatch Logs	Amazon CloudWatch Logs 可讓您集中所有系統、應用程式和您使用 AWS 服務 的日誌。然後，您可以檢視日誌、在日誌中搜尋特定的錯誤碼或模式、根據特定欄位篩選日誌，或安全封存日誌以供日後分析。 Systems Manager 支援將 SSM Agent、Run Command 和 Session Manager 的日誌傳送至 CloudWatch Logs。 進一步了解 • 將節點日誌傳送至統一 CloudWatch Logs (CloudWatch 代理程式) • 設定 Run Command 的 Amazon CloudWatch Logs • 使用 Amazon CloudWatch Logs (主控台) 記錄工作階段資料。

Amazon EventBridge	EventBridge 會傳送近乎即時的系統事件串流，說明 Amazon Web Services 資源的變動。使用您可以快速設定的簡單規則，您可以比對事件並將它們路由到一或多個目標函數或串流。EventBridge 在操作變更時會查覺到。EventBridge 會回應這些操作變更並視需要進行修正動作。這些動作包括傳送訊息來回應環境、啟用函數和擷取狀態資訊。 Systems Manager 有多個 EventBridge 支援的事件，允許您根據那些事件的內容採取相關動作。 進一步了解 使用 Amazon EventBridge 監控 Systems Manager **Note** Amazon EventBridge 是管理事件的首選方式。CloudWatch Events 和 EventBridge 是相同的基礎服務和 API，但 EventBridge 提供了更多功能。您在 CloudWatch 或 EventBridge 中所做的變更將會顯現在每個主控台中。如需詳細資訊，請參閱《Amazon EventBridge 使用者指南》。

AWS Config	AWS Config 提供 中 AWS 資源組態的詳細檢視 AWS 帳戶。這包含資源彼此之間的關係和之前的組態方式。這樣一來，您可看到一段時間中組態和關係的變化。 Systems Manager 已與 整合 AWS Config，提供多個規則，協助您了解 EC2 執行個體。這些規則可協助您識別由 Systems Manager、作業系統組態、系統層級更新、已安裝的應用程式、網路組態等管理的 EC2 執行個體。 進一步了解 • AWS Config 支援的資源類型 • 記錄受管執行個體的軟體組態 • 檢視清查歷程記錄和變更追蹤
AWS Trusted Advisor	Trusted Advisor 這個線上工具可提供您的即時指導，來協助您佈建遵循 AWS 最佳實務的資源。 Systems Manager 主機 Trusted Advisor ，您可以在 中檢視 Trusted Advisor 資料Explorer。 進一步了解 • AWS Systems Manager Explorer • 入門 AWS Trusted Advisor

AWS Organizations	Organizations 是一種帳戶管理服務，可讓您將多個 合併 AWS 帳戶 到您建立並集中管理的組織。Organizations 包括帳戶管理和合併帳單功能，可讓您更符合您商業的預算、安全及合規需求。 中的Change Manager工具 AWS Systems Manager與 Organizations 整合，可讓您透過此單一帳戶，使用委派的管理員帳戶來管理整個組織的變更請求、變更範本和核准。 Organizations 與 中的清查工具整合 AWS Systems Manager，Explorer可讓您從多個 AWS 區域 和 彙總清查和操作資料 (OpsData) AWS 帳戶。 Quick Setup 在 AWS Systems Manager和 Organizations 中的工具整合，可自動化常見的服務設定任務，並根據整個組織單位 (OUs) 的最佳實務部署服務組態。

聯網與內容交付

AWS PrivateLink	AWS PrivateLink 可讓您將虛擬私有雲端 (VPC) 私下連線至支援的 AWS 服務 和 VPC 端點服務，而不需要網際網路閘道、NAT 裝置、VPN 連線或 AWS Direct Connect 連線。 Systems Manager 支援使用 AWS PrivateLink 將受管節點連接到 Systems Manager API。這可改善受管節點的安全狀態，因為 會將受管節點、Systems Manager 和 Amazon EC2 之間的所有網路流量 AWS PrivateLink 限制在 Amazon 網路。這意味著受管節點無需存取網際網路。 進一步了解

	使用適用於 Systems Manager 的 VPC 端點來改善 EC2 執行個體的安全性

分析

Amazon Athena	Athena 是一種互動式查詢服務，允許您在 Amazon Simple Storage Service (Amazon S3) 中使用標準 SQL 直接分析資料。透過 中的幾個動作 AWS Management Console，您可以將 Athena 指向存放在 Amazon S3 中的資料，並開始使用標準 SQL 來執行一次性查詢，並在幾秒鐘內取得結果。 Systems Manager 庫存與 Athena 整合，協助您查詢來自多個 AWS 區域 和 的庫存資料 AWS 帳戶。Athena 整合會使用資源資料同步，讓您能夠在 Systems Manager 庫存主控台的 Detailed View (詳細檢視) 頁面上檢視來自所有受管節點的庫存資料。 進一步了解 • 查詢來自多個區域和帳戶的清查資料 • 演練：使用資源資料同步來彙總庫存資料
AWS Glue	AWS Glue 是全受管的擷取、轉換和載入 (ETL) 服務，可讓您以輕鬆且經濟實惠的方式，將您的資料進行分類、清理、富集，以及可靠地在各種資料存放區和資料串流之間移動。 Systems Manager 會使用 AWS Glue 來編目 S3 儲存貯體中的庫存資料。 進一步了解 查詢來自多個區域和帳戶的清查資料

Amazon QuickSight	QuickSight 是一種商業分析服務，可用來建置視覺化效果、執行一次性分析，並從資料中取得商業洞見。這能夠自動探索 AWS 資料來源，而且能夠搭配您的資料來源使用。 Systems Manager 資源資料同步可將從所有受管節點收集到的庫存資料傳送至單一 S3 儲存貯體。您可以使用 QuickSight 來查詢和分析彙總的資料。 進一步了解 • 建立庫存的資源資料同步 • 演練：使用資源資料同步來彙總庫存資料

應用程式整合

Amazon Simple Notification Service (Amazon SNS)	Amazon SNS 是一種 Web 服務，可協調與管理將訊息交付或傳送到訂閱端點或用戶端。 Systems Manager 會為多個服務產生狀態，而這些服務可透過 Amazon SNS 通知擷取。 進一步了解 • 使用 Amazon SNS 通知監控 Systems Manager 狀態變更 • 根據 Parameter Store 事件設定通知或觸發動作

AWS Management Console

AWS Resource Groups	資源群組會組織您的 AWS 資源。資源群組可讓您更輕鬆地管理、監控及一次自動化大量資源的任務。

	可以將 Systems Manager 資源類型 (例如受管節點、SSM 文件、維護時段、Parameter Store 參數和修補基準) 新增至資源群組。 進一步了解 什麼是 AWS Resource Groups?

主題

- 從 Amazon Simple Storage Service (Amazon S3) 執行指令碼
- 從Parameter Store參數參考 AWS Secrets Manager 秘密
- AWS KMSAWS Systems ManagerParameter Store SecureString 參數的加密
- 在 Amazon Elastic Kubernetes Service 中使用 Parameter Store 參數
- 在 AWS Lambda 函數中使用 Parameter Store 參數

從 Amazon Simple Storage Service (Amazon S3) 執行指令碼

本節說明如何從 Amazon Simple Storage Service (Amazon S3) 下載並執行指令碼。下列主題包含與 Amazon S3 相關的資訊和術語。若要進一步了解 Amazon S3，請參閱 What is Amazon S3? 您能夠執行不同類型的指令碼，包括 Ansible Playbook、Python、Ruby 和 PowerShell。

另外，您也能下載包含多個指令碼的目錄。當您在 目錄中執行主要指令碼時， AWS Systems Manager 也會執行 目錄中包含的任何參考指令碼。

從 Amazon Simple Storage Service (Amazon S3) 執行指令碼時，請注意以下重要詳細資訊：

- Systems Manager 不會驗證指令碼是否能夠在受管節點上執行。請確認節點上已安裝必要軟體，然後再下載和執行指令碼。或者，您可以建立複合文件，使用 Run Command或 State Manager、 中的工具安裝軟體 AWS Systems Manager，然後下載並執行指令碼。
- 確認使用者、角色或群組具備讀取 S3 儲存貯體所需的 AWS Identity and Access Management (IAM) 許可。
- 確保 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體上的執行個體設定檔具有 `s3:ListBucket` 和 `s3:GetObject` 許可。如果執行個體設定檔沒有這些許可，則系統無法從 S3 儲存貯體下載指令碼。如需詳細資訊，請參閱《IAM 使用者指南》中的使用執行個體設定檔。

從 Amazon Simple Storage Service (Amazon S3) 執行 Shell 指令碼

下列資訊包含使用 AWS Systems Manager 主控台或 AWS Command Line Interface ()，協助您從 Amazon Simple Storage Service (Amazon S3) 執行指令碼的程序AWS CLI。雖然範例中以 Shell 指令碼為例，但可替換為其他類型的指令碼。

從 Amazon Simple Storage Service (Amazon S3) (主控台) 執行 Shell 指令碼

從 Amazon Simple Storage Service (Amazon S3) 執行 Shell 指令碼

1. 開啟 AWS Systems Manager 主控台，網址為 https：//https://console.aws.amazon.com/systems-manager/。
2. 在導覽窗格中，選擇 Run Command。
3. 選擇執行命令。
4. 在 Command document (命令文件) 清單，請選擇 **AWS-RunRemoteScript**。
5. 在命令參數中，執行以下操作：

   - 在 Source Type (來源類型) 中，選取 S3。
   - 在 Source Info (來源資訊) 文字方塊中，按照以下格式輸入所需資訊，藉此存取來源。將每個*########*取代為您自己的資訊。

     > **Note**
     >
     > 用儲存貯體的 URL 取代 https://s3.*aws-api-domain*。您可以在 Objects (物件) 索引標籤上複製 Amazon S3 中的儲存貯體 URL。

     ```
     {"path":"https://s3.aws-api-domain/path to script"}
     ```

     以下是範例。

     ```
     {"path":"https://amzn-s3-demo-bucket.s3.us-east-2.amazonaws.com/scripts/shell/
     helloWorld.sh"}
     ```

   - 在 Command Line (命令列) 欄位中，輸入指令碼執行參數。請見此處範例。

     ```
     helloWorld.sh argument-1 argument-2
     ```

- (選用) 在 Working Directory (工作目錄) 欄位中，輸入節點上的目錄名稱；您將下載指令碼至該目錄並予以執行。
- (選用) 在執行逾時中，指定指令碼命令執行失敗前，系統的等待時間 (秒)。

6. 在 Targets (目標) 區段中，透過手動指定標籤、選取執行個體或邊緣裝置，或指定資源群組，選擇您要執行這項操作的受管節點。

> **Tip**
>
> 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

7. 對於其他參數：

- 在 Comment (註解) 中，輸入此命令的相關資訊。
- 在逾時 (秒) 中，指定在命令執行全面失敗之前，系統要等候的秒數。

8. 對於 Rate control (速率控制)：

- 在並行中，指定可同時執行命令的受管節點數目或百分比。

> **Note**
>
> 如果透過指定套用至受管節點的標籤或指定 AWS 資源群組選取了目標，且您不確定會以多少個受管節點為目標，則透過指定百分比限制可以同時執行文件之目標的數量。

- 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比之節點上的命令失敗之後，停止在其他受管節點上執行命令。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止傳送命令。仍在處理命令的受管節點也可能會傳送錯誤。

9. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取將命令輸出寫入至 S3 儲存貯體方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

> **Note**
>
> 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給執行個體之執行個體設定檔 (適用於 EC2 執行個體) 或 IAM 服務角色 (啟用混合模式的機器) 的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中

AWS 帳戶，請確定與受管節點相關聯的執行個體描述檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

10. 在SNS 通知區段中，如果您要傳送有關命令執行狀態的通知，請選取啟用 SNS 通知核取方塊。

    如需為 Run Command 設定 Amazon SNS 通知的詳細資訊，請參閱 使用 Amazon SNS 通知監控 Systems Manager 狀態變更。

11. 選擇執行。

從 Amazon Simple Storage Service (Amazon S3) (命令行) 執行 Shell 指令碼

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請安裝並設定 。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

2. 執行下列命令。將每個*########*取代為您自己的資訊。

> **Note**
>
> 使用儲存貯體的 URL 取代 https://s3.*aws-api-domain*/*#####*。您可以在 Objects (物件) 索引標籤上複製 Amazon S3 中的儲存貯體 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"
    }
```

從Parameter Store參數參考 AWS Secrets Manager 秘密

AWS Secrets Manager 可協助您組織和管理重要的組態資料，例如登入資料、密碼和授權金鑰。Parameter Store是 中的工具 AWS Systems Manager，與 Secrets Manager 整合，因此您可以使用已支援Parameter Store參數參考 AWS 服務 的其他 來擷取 Secrets Manager 秘密。這些服務包括 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。透過使用 Parameter Store 來參考 Secrets Manager 秘密，您就能夠建立安全、一致的流程，來呼叫和使用程式碼與組態指令碼中的秘密和參考資料。

如需 Secrets Manager 的詳細資訊，請參閱AWS Secrets Manager 《 使用者指南》中的什麼是 AWS Secrets Manager？。

限制

使用 Parameter Store 參考 Secrets Manager 秘密時，請注意以下限制：

- 您只能透過使用 GetParameter 和 GetParameters API 操作來擷取 Secrets Manager 秘密。修改作業和進階查詢 API 操作，例如 DescribeParameters 和 GetParametersByPath，不支援 Secrets Manager。
- 您可以使用 AWS Command Line Interface (AWS CLI) AWS Tools for Windows PowerShell和 SDKs，使用 擷取秘密Parameter Store。

- Parameter Store 中的 Secrets Manager 秘密必須具有 `/aws/reference/secretsmanager` 的字首。範例如下：

  ```
  /aws/reference/secretsmanager/CFCreds1
  ```

  ```
  /aws/reference/secretsmanager/myapp/db/password
  ```

- Parameter Store 連接到 Secrets Manager 秘密的 honors AWS Identity and Access Management (IAM) 政策。例如，如果使用者 1 沒有秘密 A 的存取權限，則表示使用者 1 無法透過 Parameter Store 擷取秘密 A。
- 參考 Secrets Manager 秘密的參數將無法使用 Parameter Store 的版本控制或歷程記錄功能。
- Parameter Store 會以 Secrets Manager 版本階段為準。您所參考的版本階段使用字母、數字、句號 (.)、連字號 (-) 或底線 (_)。若您在版本階段中指定其他符號，都會導致參考失敗。

如何使用 Parameter Store 參考 Secrets Manager 秘密

下列程序會說明如何使用 Parameter Store API 參考 Secrets Manager 秘密。另外，此程序還會參考《AWS Secrets Manager 使用者指南》中的其他程序。

> **Note**
>
> 開始操作之前，請確認您具備參考 Parameter Store 參數中 Secrets Manager 秘密的許可。如果您在 Secrets Manager 和 Systems Manager 中擁有管理員許可，則可使用 Parameter Store API 來參考或擷取秘密。如果您參考了 Parameter Store 參數中的 Secrets Manager 秘密，卻不具備存取該秘密的許可，參考作業將會失敗。如需詳細資訊，請參閱《AWS Secrets Manager 使用者指南》中的 AWS Secrets Manager的身分驗證與存取控制。

> **Important**
>
> Parameter Store 為可傳遞 Secrets Manager 秘密參考的服務。Parameter Store 並不會保留秘密的相關資料或中繼資料。此外，參考資料皆為無狀態。

若要使用 Parameter Store 參考 Secrets Manager 秘密

1. 在 Secrets Manager 中建立秘密。如需詳細資訊，請參閱使用 建立和管理秘密 AWS Secrets Manager。

2. 使用 AWS CLI AWS Tools for Windows PowerShell或 SDK 參考秘密。當您參考 Secrets Manager 秘密時，名稱的開頭必須採用以下預訂路徑：`/aws/reference/secretsmanager/`。一旦指定此路徑，Systems Manager 便會知道要從 Secrets Manager 擷取秘密，而非 Parameter Store。以下是一些使用 Parameter Store 正確參考 Secrets Manager、`CFCreds1` 和 `DBPass` 的參數範例：

   - `/aws/reference/secretsmanager/CFCreds1`
   - `/aws/reference/secretsmanager/DBPass`

   下方的 Java 程式碼範例會參考 Secrets Manager 中存放的存取金鑰和秘密金鑰。此程式碼範例將設定 Amazon DynamoDB 用戶端。除此之外，該程式碼會從 Parameter Store 擷取組態資料和登入資料。組態資料是以字串參數的形式存放在 Parameter Store 中，而憑證則是存放在 Secrets Manager 中。儘管組態資料和憑證存放於不同的服務，您仍可使用 `GetParameter` API 從 Parameter Store 存取這兩組資料。

```
/**
* 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();
}
```

以下是一些 AWS CLI 範例。使用 `aws secretsmanager list-secrets` 命令來尋找您秘密的名稱。

AWS CLI 範例 1：使用秘密名稱進行參考

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
```

該命令會傳回相關資訊，如以下所示。

```
{
    "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 範例 2：包含版本 ID 的參考

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
```

該命令會傳回相關資訊，如以下所示。

```
{
    "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 範例 3：包含版本階段的參考

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
```

該命令會傳回相關資訊，如以下所示。

```
{
    "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 KMSAWS Systems ManagerParameter Store SecureString 參數的加密

您可以使用 AWS Systems Manager Parameter Store建立 SecureString 參數，這些參數具有純文字參數名稱和加密參數值。 Parameter Store使用 AWS KMS 來加密和解密SecureString參數的參數值。

使用 Parameter Store 可以將資料當做具有值的參數來建立、存放和管理。您可以在 Parameter Store 中建立參數，並將它用在遵守您設計之政策和許可的多個應用程式和服務。需要變更參數值時，您只需變更一個執行個體，而不用管理容易出錯的多個來源變更。Parameter Store 支援層級結構的參數名稱，所以您可以限定參數用於特定用途。

若要管理敏感資料，您可以建立SecureString參數。 會在您建立或變更參數時 AWS KMS keys ，Parameter Store使用 來加密SecureString參數的參數值。當您存取這些參數時，它也使用 KMS 金鑰來解密參數值。使用 Parameter Store 為帳戶建立的 AWS 受管金鑰，或指定自己的客戶自管金鑰。

> **Important**
>
> Parameter Store 僅支援對稱 KMS 金鑰。您無法使用非對稱 KMS 金鑰來加密您的參數。如需判斷 KMS 金鑰是對稱金鑰還是非對稱金鑰的說明，請參閱《AWS Key Management Service 開發人員指南》中的 Identify different key types。

Parameter Store 支援兩種層級的SecureString參數：標準和進階。標準參數，不能超過 4096 個位元組，直接在您指定的 KMS 金鑰下加密和解密。為了加密和解密進階SecureString參數，Parameter Store 會使用信封加密搭配 AWS Encryption SDK。您可以將標準SecureString參數

轉換成進階參數，但無法將進階參數轉換成標準參數。如需有關標準和進階 SecureString 參數之間差異的詳細資訊，請參閱管理參數層。

主題

- 保護標準 SecureString 參數
- 保護進階 SecureString 參數
- 設定許可來加密和解密參數值
- Parameter Store 加密內容
- 對 Parameter Store 中的 KMS 金鑰問題進行疑難排解

保護標準 SecureString 參數

Parameter Store 不會執行任何密碼編譯操作。而是倚賴 AWS KMS 來加密和解密SecureString參數值。當您建立或變更標準SecureString參數值時， 會Parameter Store呼叫 AWS KMS 加密操作。這個操作會直接使用對稱加密 KMS 金鑰來加密參數值，而不使用 KMS 金鑰產生資料金鑰。

您可以選擇 Parameter Store 用來加密參數值的 KMS 金鑰。如果未指定 KMS 金鑰，Parameter Store 會使用 Systems Manager 在帳戶中自動建立的 AWS 受管金鑰 。此 KMS 金鑰具有 aws/ssm 別名。

若要檢視您帳戶的預設 aws/ssm KMS 金鑰，請使用 AWS KMS API 中的 DescribeKey 操作。下列範例使用 AWS Command Line Interface (AWS CLI) 中的 aws/ssm describe-key命令搭配別名名稱。

```
aws kms describe-key \
    --key-id alias/aws/ssm
```

若要建立標準 SecureString 參數，請使用 Systems Manager API 中的 PutParameter 操作。省略 Tier 參數，或指定值為 Standard (預設值)。加入值為 SecureString 的 Type 參數。若要指定 KMS 金鑰，請使用 KeyId 參數。預設為 AWS 受管金鑰 您帳戶的 aws/ssm。

Parameter Store 然後使用 KMS 金鑰和純文字參數值呼叫 AWS KMS Encrypt操作。 AWS KMS 會傳回以參數名稱Parameter Store存放的加密參數值。

下列範例使用 Systems Manager put-parameter 命令及其在 中的--type參數 AWS CLI 來建立SecureString參數。由於命令會省略選用 --tier 和 --key-id 參數，Parameter Store 會建立標準 SecureString 參數，並在 AWS 受管金鑰下加密此參數。

```
aws ssm put-parameter \
    --name MyParameter \
    --value "secret_value" \
    --type SecureString
```

以下類似範例使用 `--key-id` 參數來指定客戶受管金鑰。此範例使用 KMS 金鑰 ID 來識別 KMS 金鑰，但您可以使用任何有效的 KMS 金鑰識別符。由於命令省略 `Tier` 參數 (`--tier`)，Parameter Store 會建立標準`SecureString`參數，而不是進階參數。

```
aws ssm put-parameter \
    --name param1 \
    --value "secret" \
    --type SecureString \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab
```

當您從 Parameter Store 取得`SecureString`參數時，它的值會加密。若要取得參數，請使用 Systems Manager API 中的 GetParameter 操作。

下列範例使用 中的 Systems Manager get-parameter 命令 AWS CLI ，從 取得 `MyParameter` 參數，Parameter Store而不解密其值。

```
aws ssm get-parameter --name MyParameter
```

```
{
    "Parameter": {
        "Type": "SecureString",
        "Name": "MyParameter",
        "Value": "AQECAHgnOkMROh5LaLXkA4j0+vYi6tmM17Lg"
    }
}
```

若要在傳回參數值之前先解密，請將 `GetParameter` 的 `WithDecryption` 參數設定為 `true`。當您使用 時`WithDecryption`， 會代表您Parameter Store呼叫 AWS KMS 解密操作來解密參數值。因此，`GetParameter` 請求會傳回具有純文字參數值的參數，如以下範例所示。

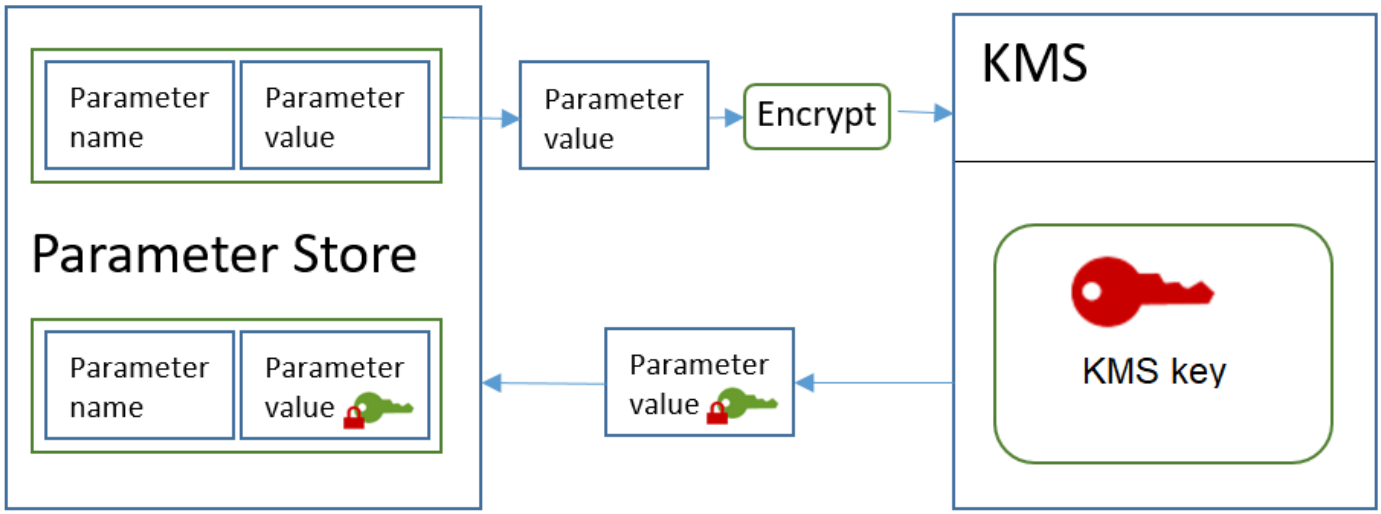
```
aws ssm get-parameter \
    --name MyParameter \
    --with-decryption
```

```
{
    "Parameter": {
        "Type": "SecureString",
        "Name": "MyParameter",
        "Value": "secret_value"
    }
}
```

以下工作流程說明 Parameter Store 如何使用 KMS 金鑰來加密和解密標準 `SecureString` 參數。

加密標準參數

1. 當您使用 `PutParameter` 建立`SecureString` 參數時，Parameter Store 會傳送 `Encrypt` 請求給 AWS KMS。該請求包含純文字參數值、選定的 KMS 金鑰，以及 Parameter Store 加密內容。在傳輸到 期間 AWS KMS， `SecureString` 參數中的純文字值受到 Transport Layer Security (TLS) 的保護。
2. AWS KMS 會使用指定的 KMS 金鑰和加密內容來加密參數值。它將加密文字傳回給 Parameter Store 來儲存參數名稱及其加密的值。



解密標準參數

1. 當您在`GetParameter`請求中包含 `WithDecryption` 參數時， 會將`Decrypt`請求Parameter Store傳送至 AWS KMS ，其中包含加密`SecureString`參數值和Parameter Store加密內容。
2. AWS KMS 使用相同的 KMS 金鑰和提供的加密內容來解密加密值。它將純文字 (解密的) 參數值傳回給 Parameter Store。在傳輸過程中，純文字資料受到 TLS 保護。
3. Parameter Store 在 `GetParameter` 回應中傳回純文字參數值給您。

保護進階 SecureString 參數

當您使用 PutParameter建立進階SecureString參數時， Parameter Store會使用信封加密搭配 AWS Encryption SDK 和對稱加密 AWS KMS key 來保護參數值。每個進階參數值都在唯一的資料金鑰下加密，而資料金鑰是在 KMS 金鑰下加密。您可以使用帳戶 (aws/ssm) 的 AWS 受管金鑰，或任何客戶受管金鑰。

AWS Encryption SDK 是開放原始碼的用戶端程式庫，可協助您使用產業標準和最佳實務來加密和解密資料。它支援多種平台和多種程式設計語言，包括命令列界面。您可以檢視原始程式碼，並在 GitHub 中對其開發做出貢獻。

對於每個SecureString參數值，Parameter Store 會呼叫 AWS Encryption SDK ，並使用 AWS KMS 產生的唯一資料金鑰 (GenerateDataKey) 來加密參數值。 AWS Encryption SDK 會傳回Parameter Store加密的訊息，其中包含加密的參數值和唯一資料金鑰的加密複本。 會將整個加密訊息Parameter Store存放在 SecureString 參數值中。然後，當您取得進階SecureString參數值時， Parameter Store會使用 AWS Encryption SDK 解密參數值。這需要呼叫 AWS KMS 來解密加密的資料金鑰。

若要建立進階 SecureString 參數，請使用 Systems Manager API 中的 PutParameter 操作。將 Tier 參數的值設為 Advanced。加入值為 SecureString 的 Type 參數。若要指定 KMS 金鑰，請使用 KeyId 參數。預設為 AWS 受管金鑰 您帳戶的 aws/ssm。

```
aws ssm put-parameter \
    --name MyParameter \
    --value "secret_value" \
    --type SecureString \
    --tier Advanced
```

以下類似範例使用 --key-id 參數來指定客戶受管金鑰。此範例使用 KMS 金鑰的 Amazon Resource Name (ARN)，但您可以使用任何有效的 KMS 金鑰識別符。

```
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
```

當您從 取得SecureString參數時Parameter Store，其值是 AWS Encryption SDK 傳回的加密訊息。若要取得參數，請使用 Systems Manager API 中的 GetParameter 操作。

以下範例使用 Systems Manager GetParameter 操作從 Parameter Store 取得 MyParameter 參數，無需解密其值。

```
aws ssm get-parameter --name MyParameter
```

```
{
    "Parameter": {
        "Type": "SecureString",
        "Name": "MyParameter",
        "Value": "AQECAHgnOkMROh5LaLXkA4j0+vYi6tmM17Lg"
    }
}
```

若要在傳回參數值之前先解密，請將 GetParameter 的 WithDecryption 參數設定為 true。當您使用 時WithDecryption， 會代表您Parameter Store呼叫 AWS KMS 解密操作，以解密 參數值。因此，GetParameter 請求會傳回具有純文字參數值的參數，如以下範例所示。

```
aws ssm get-parameter \
    --name MyParameter \
    --with-decryption
```

```
{
    "Parameter": {
        "Type": "SecureString",
        "Name": "MyParameter",
        "Value": "secret_value"
    }
}
```

您無法將進階SecureString參數轉換成標準參數，但可以將標準SecureString轉換成進階參數。若要將標準SecureString參數轉換成進階SecureString，請使用 PutParameter 操作搭配 Overwrite 參數。Type 必須是 SecureString，Tier 值必須是 Advanced。識別客戶受管金鑰的 KeyId 參數是選用的。如果您省略它， Parameter Store會使用 AWS 受管金鑰 帳戶的 。即使您使用不同的 KMS 金鑰來加密標準參數，您也可以指定委託人有權使用的任何 KMS 金鑰。

當您使用 Overwrite 參數時，Parameter Store 會使用 AWS Encryption SDK 來加密參數值。然後，它會將新加密的訊息存放在 Parameter Store 中。

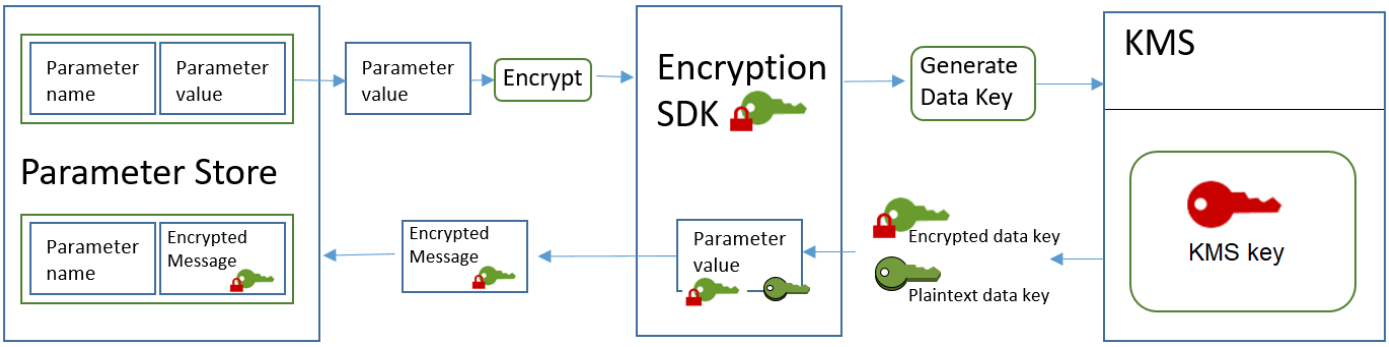
```
aws ssm put-parameter \
    --name myStdParameter \
```

```
    --value "secret_value"  \
    --type SecureString \
    --tier Advanced \
    --key-id 1234abcd-12ab-34cd-56ef-1234567890ab \
    --overwrite
```

以下工作流程說明 Parameter Store 如何使用 KMS 金鑰來加密和解密進階 SecureString 參數。

加密進階參數

1. 當您使用 PutParameter建立進階SecureString參數時， Parameter Store會使用 AWS Encryption SDK 和 AWS KMS 來加密參數值。 AWS Encryption SDK 會使用 參數值、您指定的 KMS 金鑰和Parameter Store加密內容來Parameter Store呼叫 。
2. AWS Encryption SDK 會將 GenerateDataKey 請求傳送至 ， AWS KMS 其中包含您指定的 KMS 金鑰識別符和Parameter Store加密內容。 AWS KMS 會傳回唯一資料金鑰的兩份副本：一份為純文字，一份為 KMS 金鑰下加密。(加密資料金鑰時會使用加密內容。)
3. AWS Encryption SDK 使用純文字資料金鑰來加密參數值。它會傳回 加密的訊息，其中包含加密的參數值、加密的資料金鑰和其他資料 (包括 Parameter Store 加密內容)。
4. Parameter Store 將加密的訊息儲存為參數值。



解密進階參數

1. 您可以在 GetParameter 請求中包含 WithDecryption 參數，以取得進階SecureString參數。當您這麼做時，Parameter Store 會從參數值中將加密的訊息傳遞到 AWS Encryption SDK的解密方法。
2. 會 AWS Encryption SDK 呼叫 AWS KMS 解密操作。它從加密的訊息傳入加密的資料金鑰和 Parameter Store 加密內容。
3. AWS KMS 使用 KMS 金鑰和Parameter Store加密內容來解密加密的資料金鑰。然後，它將純文字 (解密的) 資料金鑰傳回給 AWS Encryption SDK。

4. AWS Encryption SDK 使用純文字資料金鑰來解密參數值。它將純文字參數值傳回給 Parameter Store。
5. Parameter Store 驗證加密內容，並在 GetParameter 回應中將純文字參數值傳回給您。

設定許可來加密和解密參數值

若要加密標準SecureString參數值，使用者需要 kms:Encrypt 許可。若要加密進階SecureString參數值，使用者需要 kms:GenerateDataKey 許可。若要解密任一類型的SecureString參數值，使用者需要 kms:Decrypt 許可。

您可以使用 AWS Identity and Access Management (IAM) 政策來允許或拒絕使用者呼叫 Systems Manager PutParameter和 GetParameter 操作的許可。

如果使用客戶自管金鑰來加密 SecureString 參數值，則可以使用 IAM 政策和金鑰政策來管理加密和解密許可。不過，您無法為預設 aws/ssm KMS 金鑰建立存取控制政策。如需有關控制客戶自管金鑰存取的詳細資訊，請參閱《AWS Key Management Service 開發人員指南》中的 KMS key access and permissions。

以下範例顯示專為標準 SecureString 參數設計的 IAM 政策。它可讓使用者在 FinancialParameters 路徑中的所有參數上呼叫 Systems Manager PutParameter 操作。此政策也允許使用者在 AWS KMS Encrypt範例客戶受管金鑰上呼叫 操作。

```
{
    "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"
```

```
        }
    ]
}
```

以下範例顯示專為進階 SecureString 參數設計的 IAM 政策。它可讓使用者在 ReservedParameters 路徑中的所有參數上呼叫 Systems Manager PutParameter 操作。此政策也允許使用者在 AWS KMS GenerateDataKey範例客戶受管金鑰上呼叫 操作。

```
{
    "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-2:987654321098:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ]
}
```

最後一個範例還顯示可用於標準或進階SecureString參數的 IAM 政策。它可讓使用者在 ITParameters 路徑中的所有參數上呼叫 Systems Manager GetParameter 操作 (以及相關操作)。此政策也允許使用者在 AWS KMS Decrypt範例客戶受管金鑰上呼叫 操作。

```
{
    "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 加密內容

加密內容是一組金鑰/值對，其中包含任意非私密資料。當您在加密資料的請求中包含加密內容時，AWS KMS 加密會以加密內容繫結至加密的資料。若要解密資料，您必須傳遞相同的加密內容。

您也可以使用加密內容來識別稽核記錄和日誌中的密碼編譯操作。加密內容會以純文字形式出現在日誌中，例如 AWS CloudTrail 日誌。

AWS Encryption SDK 也會採用加密內容，雖然處理方式不同。 會將加密內容Parameter Store提供給加密方法。 AWS Encryption SDK 密碼編譯會將加密內容繫結至加密的資料。它在傳回的已加密訊息的標頭中，還會包含純文字形式的加密內容。不過，與解密方法不同 AWS KMS， AWS Encryption SDK 解密方法不會將加密內容視為輸入。相反地，當 解密資料時， 會從加密的訊息 AWS Encryption SDK 取得加密內容。 Parameter Store 會驗證加密內容包含預期的值，然後再將純文字參數值傳回給您。

Parameter Store 在其密碼編譯操作中使用下列加密內容：

- 索引鍵：PARAMETER_ARN
- 值：要加密之參數的 Amazon Resource Name (ARN)。

加密內容的格式如下：

```
"PARAMETER_ARN":"arn:aws:ssm:region-id:account-id:parameter/parameter-name"
```

例如， 會在呼叫中Parameter Store包含此加密內容，以加密和解密範例 AWS 帳戶 和區域中的 MyParameter 參數。

```
"PARAMETER_ARN":"arn:aws:ssm:us-east-2:111122223333:parameter/MyParameter"
```

如果參數位於 Parameter Store 階層路徑，則路徑和名稱會包含在加密內容中。例如，在範例和區域中的/ReadableParameters路徑中加密 AWS 帳戶 和解密 MyParameter 參數時，會使用此加密內容。

```
"PARAMETER_ARN":"arn:aws:ssm:us-east-2:111122223333:parameter/ReadableParameters/
MyParameter"
```

您可以透過使用正確的加密內容和 Systems Manager 操作傳回的加密SecureString參數值呼叫 AWS KMS DecryptGetParameter操作，來解密加密的參數值。不過，我們建議您使用 GetParameter 操作搭配 WithDecryption 參數來解密 Parameter Store 參數值。

您也可以在 IAM 政策包含加密內容。例如，您可以允許使用者只解密一個特定的參數值或一組參數值。

以下範例 IAM 政策陳述式允許使用者取得 MyParameter 參數的值和使用指定的 KMS 金鑰來解密其值。不過，只有在加密內容符合指定的字串時，才會套用許可。這些許可不適用於任何其他參數或 KMS 金鑰，如果加密內容不符合字串，則呼叫 GetParameter 會失敗。

使用這類政策陳述式之前，請使用有效值取代*## ARN*。

```
{
    "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-2:987654321098:key/1234abcd-12ab-34cd-56ef-1234567890ab",
            "Condition": {
```

```
                "StringEquals": {
                    "kms:EncryptionContext:PARAMETER_ARN":"arn:aws:ssm:us-
east-2:111122223333:parameter/MyParameter"
                }
            }
        }
    ]
}
```

對 Parameter Store 中的 KMS 金鑰問題進行疑難排解

若要對SecureString參數執行任何操作， Parameter Store 必須能夠使用您為預期操作指定的 AWS KMS KMS 金鑰。KMS 金鑰相關的大部分 Parameter Store 失敗是由下列問題引起：

- 應用程式使用的登入資料無權在 KMS 金鑰上執行指定的動作。

  若要修正此錯誤，請使用不同的登入資料來執行應用程式，或修改防止操作的 IAM 或金鑰政策。如需 IAM AWS KMS 和金鑰政策的說明，請參閱《 AWS Key Management Service 開發人員指南》中的 KMS 金鑰存取和許可。
- 找不到 KMS 金鑰。

  這種情況通常發生在您為 KMS 金鑰使用不正確的識別符。為 KMS 金鑰尋找正確的識別符，再試一次命令。
- 未啟用 KMS 金鑰。發生這種情況時， 會Parameter Store傳回 InvalidKeyId 例外狀況，其中包含來自 的詳細錯誤訊息 AWS KMS。如果 KMS 金鑰狀態為 Disabled，請啟用。如果是 Pending Import，請完成匯入程序。如果金鑰狀態為 Pending Deletion，請取消金鑰刪除或使用不同的 KMS 金鑰。

  若要尋找 KMS 金鑰的金鑰狀態，請使用 DescribeKey 操作。

在 Amazon Elastic Kubernetes Service 中使用 Parameter Store 參數

若要將 Parameter Store的工具 中的參數顯示 AWS Systems Manager為掛載在 Amazon EKS Pod 中的檔案，您可以使用 Kubernetes AWS Secrets Store CSI Driver 的 Secrets and Configuration Provider。ASCP 可與執行 Amazon EC2 節點群組的 Amazon Elastic Kubernetes Service 1.17+ 搭配使用。不支援 Amazon EC2 節點群組。 AWS Fargate

使用 ASCP，您可以在 中存放和管理參數，Parameter Store然後透過在 Amazon EKS 上執行的工作負載擷取參數。如果您的參數包含 JSON 格式的多個鍵值對，您可以選擇要在 Amazon EKS 中掛載的

鍵值對。ASCP 使用JMESPath語法來查詢秘密中的鍵/值對。ASCP 也可使用 AWS Secrets Manager 秘密。

ASCP 提供兩種使用 Amazon EKS 進行身分驗證的方法。第一種方法使用服務帳戶的 IAM 角色 (IRSA)。第二個方法使用 Pod 身分。每種方法都有其優點和使用案例。

具有服務帳戶 IAM 角色的 ASCP (IRSA)

具有服務帳戶 IAM 角色 (IRSA) 的 ASCP 可讓您從 將參數掛載Parameter Store為 Amazon EKS Pod 中的檔案。此方法適用於下列情況：

- 您需要將參數掛載為 Pod 中的檔案。
- 您使用 Amazon EKS 1.17 版或更新版本搭配 Amazon EC2 節點群組。
- 您想要從 JSON 格式參數擷取特定的鍵值對。

如需詳細資訊，請參閱the section called “將 ASCP 與適用於 Amazon EKS 的 IRSA 整合”。

具有 Pod 身分的 ASCP

[具有 EKS Pod 身分的 ASCP]

具有 Pod Identity 的 ASCP 方法可增強安全性，並簡化存取 中參數的組態Parameter Store。此方法在下列情況下非常有用：

- 您需要在 Pod 層級進行更精細的許可管理。
- 您使用的是 Amazon EKS 1.24 版或更新版本。
- 您想要改善效能和可擴展性。

如需詳細資訊，請參閱the section called “將 ASCP 與 Amazon EKS 的 Pod Identity 整合”。

選擇正確的方法

在 ASCP 搭配 IRSA 和 ASCP 搭配 Pod 身分之間進行決策時，請考慮下列因素：

- Amazon EKSversion：Pod Identity 需要 Amazon EKS 1.24+，而 CSI 驅動程式可與 Amazon EKS 1.17+ 搭配使用。
- 安全需求：Pod Identity 在 Pod 層級提供更精細的控制。
- 效能：Pod Identity 通常在大規模環境中表現更佳。
- 複雜性：Pod Identity 透過消除對個別服務帳戶的需求，簡化設定。

選擇最符合您特定需求和 Amazon EKS 環境的方法。

安裝適用於 Amazon EKS 的 ASCP

本節說明如何安裝 Amazon EKS 的 AWS Secrets and Configuration Provider。使用 ASCP，您可以將來自 的參數Parameter Store和來自 的秘密掛載 AWS Secrets Manager 為 Amazon EKS Pod 中的檔案。

先決條件

- Amazon EKS 叢集
  - 適用於 Pod Identity 的 1.24 版或更新版本
  - 適用於 IRSA 的 1.17 版或更新版本
- AWS CLI 已安裝和設定的
- 為 Amazon EKS 叢集安裝和設定 kubectl
- Helm (3.0 版或更新版本)

安裝和設定 ASCP

您可在 secrets-store-csi-provider-aws 儲存庫中的 GitHub 取得 ASCP。儲存庫也包含範例 YAML 檔案，可透過將`objectType`值從 變更為 `secretsmanager` 來建立和掛載秘密`ssmparameter`。

在安裝期間，您可以將 ASCP 設定為使用 FIPS 端點。如需 Systems Manager 端點的清單，請參閱《》中的 Systems Manager 服務端點Amazon Web Services 一般參考。

使用 Helm 安裝 ASCP

1. 若要確保儲存庫指向最新的圖表，請使用 `helm repo update.`
2. 新增 Secrets Store CSI 驅動程式圖表。

```
helm repo add secrets-store-csi-driver https://kubernetes-sigs.github.io/secrets-store-csi-driver/charts
```

3. 安裝圖表。若要設定限流，請新增下列旗標： `--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. 新增 ASCP 圖表。

```
helm repo add aws-secrets-manager https://aws.github.io/secrets-store-csi-driver-provider-aws
```

5. 安裝圖表。若要使用 FIPS 端點，請新增下列旗標： `--set useFipsEndpoint=true`

```
helm install -n kube-system secrets-provider-aws aws-secrets-manager/secrets-store-csi-driver-provider-aws
```

使用儲存庫中的 YAML 安裝

- 使用下列命令。

```
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
```

驗證安裝

若要驗證 EKS 叢集、Secrets Store CSI 驅動程式和 ASCP 外掛程式的安裝，請遵循下列步驟：

1. 驗證 EKS 叢集：

```
eksctl get cluster --name clusterName
```

此命令應會傳回叢集的相關資訊。

2. 驗證 Secrets Store CSI 驅動程式安裝：

```
kubectl get pods -n kube-system -l app=secrets-store-csi-driver
```

您應該會看到執行的名稱為 的 Pod`csi-secrets-store-secrets-store-csi-driver-xxx`。

3. 驗證 ASCP 外掛程式安裝：

YAML installation

```
$ kubectl get pods -n kube-system -l app=csi-secrets-store-provider-aws
```

輸出範例：

```
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
```

輸出範例：

```
NAME                                                  READY   STATUS    RESTARTS
 AGE
secrets-provider-aws-secrets-store-csi-driver-provider-67890       1/1
 Running   0          2m
```

您應該會看到處於 Running 狀態的 Pod。

執行這些命令後，如果一切設定正確，您應該會看到所有元件在執行中沒有任何錯誤。如果您遇到任何問題，您可能需要檢查有問題的特定 Pod 日誌來進行故障診斷。

故障診斷

1. 若要檢查 ASCP 提供者的日誌，請執行：

   ```
   kubectl logs -n kube-system -l app=csi-secrets-store-provider-aws
   ```

2. 檢查kube-system命名空間中所有 Pod 的狀態。

   將########取代為您自己的 Pod ID：

   ```
   kubectl -n kube-system get pods
   ```

```
kubectl -n kube-system logs pod/pod-id
```

與 CSI 驅動程式和 ASCP 相關的所有 Pod 都應處於「執行中」狀態。

3. 檢查 CSI 驅動程式版本：

```
kubectl get csidriver secrets-store.csi.k8s.io -o yaml
```

此命令應會傳回已安裝 CSI 驅動程式的相關資訊。

其他資源

如需搭配 Amazon EKS 使用 ASCP 的詳細資訊，請參閱下列資源：

- 搭配 Amazon EKS 使用 Pod 身分
- AWS Secrets 在 GitHub 上存放 CSI 驅動程式

使用 AWS 秘密和組態提供者 CSI 搭配 Amazon EKS 的 Pod 身分

AWS Secrets and Configuration Provider 與 Pod Identity Agent for Amazon Elastic Kubernetes Service 整合，可為在 Amazon EKS 上執行的應用程式提供增強的安全性、簡化組態並改善效能。從 AWS Systems Manager Parameter Store Secrets Manager 擷取參數或秘密時，Pod Identity 可簡化 Amazon EKS 的 AWS Identity and Access Management (IAM) 身分驗證。

Amazon EKS Pod Identity 透過允許直接透過 Amazon EKS 介面設定許可、減少步驟數量，並消除在 Amazon EKS 和 IAM 服務之間切換的需求，簡化設定 Kubernetes 應用程式 IAM 許可的程序。Pod Identity 可讓您跨多個叢集使用單一 IAM 角色，而無需更新信任政策，並支援角色工作階段標籤，以實現更精細的存取控制。這種方法不僅允許跨角色重複使用許可政策來簡化政策管理，還可以根據相符標籤啟用對 AWS 資源的存取，以增強安全性。

運作方式

1. Pod Identity 會將 IAM 角色指派給 Pod。
2. ASCP 使用此角色進行身分驗證 AWS 服務。
3. 如果獲得授權，ASCP 會擷取請求的參數，並將其提供給 Pod。

如需詳細資訊，請參閱《Amazon EKS 使用者指南》中的了解 Amazon EKS Pod Identity 的運作方式。

先決條件

> **Important**
>
> Pod Identity 僅支援雲端中的 Amazon EKS。Amazon EC2 執行個體上的 Amazon EKS Anywhere、Red Hat OpenShift Service on AWS、 或自我管理 Kubernetes 叢集不支援此功能。

- Amazon EKS 叢集 (1.24 版或更新版本）
- 透過 存取 AWS CLI 和 Amazon EKS 叢集 `kubectl`
- （選用） 存取兩個 AWS 帳戶 以進行跨帳戶存取

安裝 Amazon EKS Pod Identity Agent

若要在叢集中使用 Pod Identity，您必須安裝 Amazon EKS Pod Identity Agent 附加元件。

安裝 Pod Identity Agent

- 在叢集上安裝 Pod Identity Agent 附加元件。

  將*########*取代為您自己的值：

  ```
  eksctl create addon \
    --name eks-pod-identity-agent \
    --cluster clusterName \
    --region region
  ```

使用 Pod 身分設定 ASCP

1. 建立許可政策，將 Pod 需要存取的參數授予 `ssm:GetParameters`和 `ssm:DescribeParameters`許可。
2. 建立可由 Amazon EKS 服務主體擔任的 Pod Identity IAM 角色：

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

```
      {
        "Effect": "Allow",
        "Principal": {
          "Service": "pods.eks.amazonaws.com"
        },
        "Action": [
          "sts:AssumeRole",
          "sts:TagSession"
        ]
      }
    ]
  }
```

將 IAM 政策連接至角色。

將########取代為您自己的值：

```
aws iam attach-role-policy \
  --role-name MY_ROLE \
  --policy-arn POLICY_ARN
```

3. 建立 Pod Identity 關聯。如需範例，請參閱《Amazon EKS 使用者指南》中的建立 Pod 身分關聯
4. 建立 SecretProviderClass，指定要在 Pod 中掛載的參數或秘密：

```
kubectl apply -f kubectl apply -f https://raw.githubusercontent.com/aws/
secrets-store-csi-driver-provider-aws/main/examples/ExampleSecretProviderClass-
PodIdentity.yaml
```

IRSA 和 Pod Identity SecretProviderClass之間的主要差異是選用參數 usePodIdentity。這是選用欄位，可決定身分驗證方法。未指定時，它會預設為使用服務帳戶的 IAM 角色 (IRSA)。

- 若要使用 EKS Pod Identity，請使用下列任何值："true", "True", "TRUE", "t", "T"。
- 若要明確使用 IRSA，請將 設定為下列任何值："false", "False", "FALSE", "f", or "F"。

5. 在 下部署掛載參數或秘密的 Pod/mnt/secrets-store：

```
kubectl apply -f kubectl apply -f https://raw.githubusercontent.com/aws/secrets-
store-csi-driver-provider-aws/main/examples/ExampleDeployment-PodIdentity.yaml
```

6. 如果您使用私有 Amazon EKS 叢集，請確定叢集所在的 VPC 具有 AWS STS 端點。如需有關建立端點的資訊，請參閱AWS Identity and Access Management 《 使用者指南》中的界面 VPC 端點。

驗證秘密掛載

若要驗證 參數或秘密是否已正確掛載，請執行下列命令。

將########取代為您自己的值：

```
kubectl exec -it $(kubectl get pods | awk '/pod-identity-deployment/{print $1}' | head
 -1) -- cat /mnt/secrets-store/MyParameter
```

設定 Amazon EKS Pod Identity 以存取 中的參數 Parameter Store

1. 建立許可政策，將 Pod 需要存取的參數授予 `ssm:GetParameters`和 `ssm:DescribeParameters`許可。
2. 如果您還沒有參數Parameter Store，請在 中建立參數。如需相關資訊，請參閱在 Systems Manager 中建立 Parameter Store 參數。

疑難排解

您可以透過描述 Pod 部署來檢視大多數錯誤。

若要查看容器的錯誤訊息

1. 使用下列命令取得 Pod 名稱清單。如果不使用預設命名空間，請使用 `-n` *namespace*。

   ```
   kubectl get pods
   ```

2. 若要描述 Pod，請在下列命令中，針對 *Pod-id* 使用您在上一個步驟中找到的 Pod 中的 Pod ID。如果不使用預設命名空間，請使用 `-n` *NAMESPACE*。

   ```
   kubectl describe pod/pod-id
   ```

若要查看 ASCP 的錯誤

- 若要在提供者日誌中尋找更多資訊，請在下列命令中，針對 *PODID* 使用 csi-secrets-store-provider-aws Pod 的 ID。

```
kubectl -n kube-system get pods
kubectl -n kube-system logs pod/pod-id
```

使用 AWS Secrets and Configuration Provider CSI 搭配服務帳戶 (IRSA) 的 IAM 角色

主題

- 先決條件
- 設定存取控制
- 識別要掛載的參數
- 疑難排解

先決條件

- Amazon EKS 叢集 (1.17 版或更新版本)
- 透過 存取 AWS CLI 和 Amazon EKS 叢集 `kubectl`

設定存取控制

ASCP 會擷取 Amazon EKS Pod 身分，並將其交換為 IAM 角色。您可以在 IAM 政策中為該 IAM 角色設定許可。當 ASCP 擔任 IAM 角色時，它會存取您授權的參數。除非您也將其與 IAM 角色建立關聯，否則其他容器無法存取參數。

若要授予 中參數的 Amazon EKS Pod 存取權 Parameter Store

1. 建立許可政策，將 Pod 需要存取的參數授予 `ssm:GetParameters`和 `ssm:DescribeParameters`許可。
2. 如果尚未建立，請為叢集建立 IAM OpenID Connect (OIDC) 提供者。如需詳細資訊，請參閱《Amazon EKS 使用者指南》中的為您的叢集建立 IAM OIDC 提供者。
3. 為服務帳戶建立 IAM 角色，並將政策連接至該角色。如需詳細資訊，請參閱《Amazon EKS 使用者指南》中的為服務帳戶建立 IAM 角色。

4. 如果您使用私有 Amazon EKS 叢集，請確定叢集所在的 VPC 具有 AWS STS 端點。如需有關建立端點的資訊，請參閱AWS Identity and Access Management 《 使用者指南》中的界面 VPC 端點。

識別要掛載的參數

若要判斷 ASCP 在 Amazon EKS 中掛載哪些參數做為檔案系統上的檔案，您可以建立 the section called “SecretProviderClass” YAML 檔案。`SecretProviderClass` 列出要掛載的參數和要掛載的檔案名稱。`SecretProviderClass` 必須與參考的 Amazon EKS Pod 位於相同的命名空間中。

將參數掛載為檔案

下列指示說明如何使用範例 YAML 檔案 ExampleSecretProviderClass.yaml 和 ExampleDeployment.yaml 將參數掛載為檔案。

在 Amazon EKS 中掛載參數

1. 將 `SecretProviderClass`套用至 Pod：

```
kubectl apply -f ExampleSecretProviderClass.yaml
```

2. 部署您的 Pod：

```
kubectl apply -f ExampleDeployment.yaml
```

3. ASCP 掛載檔案。

疑難排解

您可以透過描述 Pod 部署來檢視大多數錯誤。

若要查看容器的錯誤訊息

1. 使用下列命令取得 Pod 名稱清單。如果不使用預設命名空間，請使用 `-n` *`name-space`*。

```
kubectl get pods
```

2. 若要描述 Pod，請在下列命令中，針對 *`Pod-id`* 使用您在上一個步驟中找到的 Pod 中的 Pod ID。如果不使用預設命名空間，請使用 `-n` *`nameSpace`*。

```
kubectl describe pod/pod-id
```

若要查看 ASCP 的錯誤

- 若要在提供者日誌中尋找詳細資訊，請在下列命令中，針對 *Pod-id*，使用 csi-secrets-store-provider-aws Pod 的 ID。

```
kubectl -n kube-system get pods
kubectl -n kube-system logs Pod/pod-id
```

- 確認 **SecretProviderClass** CRD 已安裝：

```
kubectl get crd secretproviderclasses.secrets-store.csi.x-k8s.io
```

此命令應會傳回SecretProviderClass自訂資源定義的相關資訊。

- 確認已建立 SecretProviderClass 物件。

```
kubectl get secretproviderclass SecretProviderClassName -o yaml
```

AWS 秘密和組態提供者程式碼範例

ASCP 身分驗證和存取控制範例

範例：允許 Amazon EKS Pod Identity Service (pods.eks.amazonaws.com：//) 擔任角色並標記工作階段的 IAM 政策：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "pods.eks.amazonaws.com"
      },
      "Action": [
```

```
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
```

SecretProviderClass

您可以使用 YAML 來描述要使用 ASCP 在 Amazon EKS 中掛載哪些參數。如需範例，請參閱 the section called “SecretProviderClass 用量”。

SecretProviderClass YAML 結構

```
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
   name: name
spec:
  provider: aws
  parameters:
    region:
    failoverRegion:
    pathTranslation:
    usePodIdentity:
    preferredAddressType:
    objects:
```

參數欄位包含掛載請求的詳細資訊：

region

（選用）參數 AWS 區域 的 。如果不使用此欄位，ASCP 會從節點上的註釋尋找區域。此查詢會增加掛載請求的額外負荷，因此建議您為使用大量 Pod 的叢集提供 區域。

如果您也指定 failoverRegion，ASCP 會嘗試從兩個區域擷取 參數。如果任一區域傳回4xx錯誤，例如身分驗證問題，則 ASCP 不會掛載任一參數。如果成功從 擷取參數region，則 ASCP 會掛載該參數值。如果未成功從 擷取參數region，但成功從 擷取failoverRegion，則 ASCP 會掛載該參數值。

failoverRegion

（選用）如果您包含此欄位，ASCP 會嘗試從 region和此欄位中定義的區域擷取參數。如果任一區域傳回4xx錯誤，例如身分驗證問題，則 ASCP 不會掛載任一參數。如果成功從 擷取參數region，則 ASCP 會掛載該參數值。如果未成功從 擷取參數region，但成功從 擷取failoverRegion，則 ASCP 會掛載該參數值。如需如何使用此欄位的範例，請參閱 多區域參數容錯移轉。

pathTranslation

(選用) 如果 Amazon EKS 中的檔案名稱將包含路徑分隔符號字元，例如 Linux 上的斜線 (/)，則要使用的單一替代字元。ASCP 無法建立包含路徑分隔符號字元的掛載檔案。相反地，ASCP 會將路徑分隔符號字元取代為其他字元。如果不使用此欄位，則取代字元為底線 (_)，例如 My/Path/Parameter 掛載為 My_Path_Parameter。

若要避免發生字元取代的情況，請輸入字串 False。

usePodIdentity

（選用）決定身分驗證方法。未指定時，它會預設為服務帳戶 (IRSA) (IRSA) 的 IAM 角色。

- 若要使用 EKS Pod 身分，請使用下列任何值："true""、"True"、"TRUE"、 "t"或 "T"。
- 若要明確使用 IRSA，請將 設定為下列任何值："false"、"False"、"f"、 "FALSE"或 "F""=。

preferredAddressType

（選用）指定 Pod Identity Agent 端點通訊的慣用 IP 地址類型。欄位僅適用於使用 EKS Pod 身分功能時，並在使用服務帳戶的 IAM 角色時忽略。值不區分大小寫。有效的 值如下：

- "ipv4"、"IPv4"" 或 "IPV4" – 強制使用 Pod Identity Agent IPv4 端點
- "ipv6"、 "IPv6"或 "IPV6" – 強制使用 Pod Identity Agent IPv6 端點
- 未指定 – 使用自動端點選擇、先嘗試 IPv4 端點，並在 IPv4 失敗時IPv4IPv6 端點

objects

包含待掛載秘密的 YAML 宣告的字串。建議使用 YAML 多行字串或分隔號 (|) 字元。

objectName

必要。指定要擷取的參數或秘密的名稱。對於 Parameter Store，這是 參數Name的 ，可以是 參數的名稱或完整 ARN。對於 Secrets Manager，這是 SecretId 參數，可以是秘密的易記名稱或完整 ARN。

objectType

如果未針對 objectName 使用 Secrets Manager ARN，其則為必要欄位。對於 Parameter Store，請使用 ssmparameter。對於 Secrets Manager，請使用 secretsmanager。

objectAlias

（選用）Amazon EKS Pod 中秘密的檔案名稱。如果您未指定此欄位，objectName 會顯示為檔案名稱。

objectVersion

（選用）參數的版本 ID。不建議，因為每次更新 參數時都必須更新版本 ID。依預設，會使用最新版本。如果您包含 failoverRegion，則此欄位代表主要 objectVersion。

objectVersionLabel

(選用) 版本的別名。預設值為最新版本 AWSCURRENT。如果您包含 failoverRegion，則此欄位代表主要 objectVersionLabel。

jmesPath

（選用）參數中索引鍵映射至要在 Amazon EKS 中掛載的檔案。若要使用此欄位，您的參數值必須是 JSON 格式。

下列範例會顯示 JSON 編碼參數的樣子。

```
{
    "username" : "myusername",
    "password" : "mypassword"
}
```

金鑰為 username 和 password。與 username 關聯的值是 myusername，與 password 關聯的值是 mypassword。

如果使用此欄位，您必須包含 path 和 objectAlias。

路徑

來自參數值 JSON 中鍵值對的索引鍵。如果欄位包含連字號，請使用單引號將其逸出，例如：path: '"hyphenated-path"'。

objectAlias

要掛載在 Amazon EKS Pod 中的檔案名稱。如果欄位包含連字號，請使用單引號將其逸出，例如：objectAlias: '"hyphenated-alias"'。

failoverObject

（選用）如果您指定此欄位，ASCP 會嘗試同時擷取主要欄位中指定的參數`objectName`和`failoverObject`的`objectName`子欄位中指定的參數。如果任一 傳回`4xx`錯誤，例如身分驗證問題，則 ASCP 不會掛載任一參數。如果從主要 成功擷取 參數`objectName`，則 ASCP 會掛載該參數值。如果未成功從主要 擷取參數`objectName`，但成功從容錯移轉 擷取`objectName`，則 ASCP 會掛載該參數值。如果包含此欄位，則您必須包含欄位 `objectAlias`。如需如何使用此欄位的範例，請參閱 容錯移轉至不同的參數。

當容錯移轉參數不是複本時，您通常會使用此欄位。如需如何指定複本的範例，請參閱 多區域參數容錯移轉。

objectName

容錯移轉參數的名稱或完整 ARN。如果您使用 ARN，則 ARN 中的區域必須與欄位 `failoverRegion` 相符。

objectVersion

（選用）參數的版本 ID。必須與主要 `objectVersion` 相符。不建議，因為每次更新 參數時都必須更新版本 ID。依預設，會使用最新版本。

objectVersionLabel

(選用) 版本的別名。預設值為最新版本 `AWSCURRENT`。

建立基本 SecretProviderClass 組態，以在 Amazon EKS Pod 中掛載參數。

Pod Identity

SecretProviderClass 可在相同的 Amazon EKS 叢集中使用參數：

```
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 用量

使用這些範例來建立不同案例的SecretProviderClass組態。

範例：依名稱或 ARN 掛載參數

此範例說明如何掛載三種不同類型的參數：

- 完整 ARN 指定的參數
- 名稱指定的參數
- 秘密的參數版本

```
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"
```

範例：從參數掛載鍵/值對

此範例示範如何從 JSON 格式參數掛載特定的鍵值對：

```
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
```

範例：容錯移轉組態範例

這些範例示範如何設定參數的容錯移轉。

多區域參數容錯移轉

此範例說明如何為跨多個區域複寫的參數設定自動容錯移轉：

```
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"
```

容錯移轉至不同的參數

此範例說明如何設定容錯移轉至不同的參數（非複本）：

```
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"
```

其他資源

如需搭配 Amazon EKS 使用 ASCP 的詳細資訊，請參閱下列資源：

- 搭配 Amazon EKS 使用 Pod Identity
- AWS Secrets 在 GitHub 上存放 CSI 驅動程式

在 AWS Lambda 函數中使用 Parameter Store 參數

Parameter Store是 中的工具 AWS Systems Manager，可為組態資料管理和秘密管理提供安全的階層式儲存。您可以存放密碼、資料庫字串、Amazon Machine Image (AMI) ID 和授權碼之類的資料做為參數值。

若要在 AWS Lambda 函數Parameter Store中使用來自 的參數而不使用 SDK，您可以使用 AWS 參數和秘密 Lambda 延伸。此延伸會擷取參數值並對其進行快取以供今後使用。使用 Lambda 延伸可藉由減少對 Parameter Store 的 API 呼叫次數來降低成本。使用延伸還可改善延遲，因為擷取快取的參數比從 Parameter Store 中擷取該參數快。

Lambda 延伸是增強 Lambda 函數功能的隨附程序。延伸就像是與 Lambda 叫用平行執行的用戶端。此平行用戶端可在其生命週期中的任何時間點與您的功能連接。如需有關 Lambda 延伸的詳細資訊，

請參閱《AWS Lambda 開發人員指南》https://docs.aws.amazon.com/lambda/latest/dg/runtimes-extensions-api.html中的 Lambda 延伸 API。

AWS 參數和秘密 Lambda 延伸模組適用於 Parameter Store和 AWS Secrets Manager。若要了解如何使用 Lambda 擴充功能搭配 Secrets Manager 的秘密，請參閱AWS Secrets Manager 《 使用者指南》中的在 AWS Lambda 函數中使用 AWS Secrets Manager 秘密。

相關資訊

使用 AWS 參數和秘密 Lambda 延伸來快取參數和秘密 (AWS 運算部落格)

延伸運作方式

若要在沒有 Lambda 延伸的情況下使用 Lambda 函數中的參數，必須將 Lambda 函數設定為透過整合 Parameter Store 的 `GetParameter` API 動作來接收組態更新。

當您使用 AWS Parameters and Secrets Lambda Extension 時，該延伸模組會從 擷取參數值，Parameter Store並將其存放在本機快取中。然後，快取值會用於進一步叫用，直至其過期。快取值在超過其存留時間 (TTL) 後過期。您可以使用 `SSM_PARAMETER_STORE_TTL` 環境變數來設定 TTL 值，如本主題稍後所述。

如果設定的快取 TTL 尚未過期，則會使用快取的參數值。如果時間已過期，則快取值會失效，並從 Parameter Store 中擷取參數值。

此外，系統會偵測經常使用的參數值，並將其保留在快取中，同時清除已過期或未使用的參數值。

> **Important**
>
> 延伸模組只能在 Lambda 操作的 `INVOKE`階段調用，不能在 `INIT`階段調用。

實作詳細資訊

使用下列詳細資訊來協助您設定 AWS Parameters and Secrets Lambda Extension。

身分驗證

為了授權和驗證 Parameter Store 請求，延伸會使用與執行 Lambda 函數本身所用相同的憑證。因此，用來執行函數的 AWS Identity and Access Management (IAM) 角色必須具有下列許可，才能與 互動Parameter Store：

- `ssm:GetParameter` – 從 Parameter Store 擷取參數時需要

- `kms:Decrypt` – 要從 Parameter Store 中擷取 `SecureString` 參數時需要

如需詳細資訊，請參閱《AWS Lambda 開發人員指南》中的 AWS Lambda 執行角色。

執行個體化

Lambda 會將與函數所要求並行層級相符的另外執行個體具現化。每個執行個體都彼此隔離，並維護自己組態資料的本機快取。如需有關 Lambda 執行個體的詳細資訊，請參閱《AWS Lambda 開發人員指南》中的設定預留並行一節。

沒有 SDK 相依性

AWS 參數和秘密 Lambda 延伸模組的運作獨立於任何 AWS SDK 語言程式庫。向 提出 GET 請求不需要 AWS SDKParameter Store。

Localhost 連接埠

在 GET 請求中使用 `localhost`。延伸會向 localhost 連接埠 2773 提出請求。您無需指定外部或內部端點即可使用延伸。您可以設定環境變數 `PARAMETERS_SECRETS_EXTENSION_HTTP_PORT` 來設定連接埠。

例如，在 Python 中，您的 GET URL 可能看起來像下面的範例。

```
parameter_url = ('http://localhost:' + port + '/systemsmanager/parameters/get/?
name=' + ssm_parameter_path)
```

TTL 到期前的參數值變更

延伸不會偵測到參數值變更，也不會在 TTL 到期之前執行自動重新整理。如果您變更參數值，則使用快取參數值的操作可能會失敗，直至下次重新整理快取為止。如果您期望頻繁變更參數值，則建議您設定較短的 TTL 值。

標頭要求

若要從延伸快取擷取參數，GET 請求的標頭必須包含 `X-Aws-Parameters-Secrets-Token` 參考。將字符設定為 `AWS_SESSION_TOKEN`，這會由 Lambda 針對所有運行中的函數提供。使用此標頭表示呼叫者位於 Lambda 環境中。

範例

下列 Python 範例示範了擷取快取參數值的基本請求。

```
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 支援

此延伸模組在支援 x86_64 和 x86 架構的大多數 AWS 區域 中支援 ARM 架構。如果使用的是 ARM 架構，建議確認架構是否受支援。如需延伸 ARN 的完整清單，請參閱 AWS 參數和秘密 Lambda 延伸ARNs。

日誌

Lambda 會使用 Amazon CloudWatch Logs，記錄有關延伸模組及函數的執行資訊。延伸模組預設會將最少量的資訊記錄到 CloudWatch。若要記錄更多詳細資訊，請將環境變數 PARAMETERS_SECRETS_EXTENSION_LOG_LEVEL 設定為 DEBUG。

將延伸新增至 Lambda 函數

若要使用 AWS Parameters and Secrets Lambda Extension，請將延伸新增至 Lambda 函數做為 layer。

使用以下方法之一將延伸新增至您的函數。

AWS Management Console （新增 layer 選項）

1. 在 https：//https://console.aws.amazon.com/lambda/ 開啟 AWS Lambda 主控台。
2. 選擇函數。在 Layers (圖層) 區域中，選擇 Add a layer (新增圖層)。
3. 在選擇圖層區域中，選擇 AWS 圖層選項。
4. 針對 AWS 圖層，選擇 AWS-Parameters-and-Secrets-Lambda-Extension，並選擇版本，然後選擇新增。

AWS Management Console （指定 ARN 選項）

1. 在 https：//https://console.aws.amazon.com/lambda/ 開啟 AWS Lambda 主控台。

2. 選擇函數。在 Layers (圖層) 區域中，選擇 Add a layer (新增圖層)。
3. 在 Choose a layer (選擇圖層) 區域中，選擇 Specify an ARN (指定 ARN) 選項。
4. 針對指定 ARN，輸入 AWS 區域 和 架構的延伸 ARN，然後選擇新增。

AWS Command Line Interface

在 AWS CLI中執行以下命令。將每個########取代為您自己的資訊。

```
aws lambda update-function-configuration \
    --function-name function-name \
    --layers layer-ARN
```

相關資訊

搭配使用圖層和 Lambda 函數

設定延伸 (.zip 檔案封存)

AWS 參數和秘密 Lambda 延伸環境變數

您可以變更下列環境變數來設定延伸。若要查看目前的設定，請將 PARAMETERS_SECRETS_EXTENSION_LOG_LEVEL 設定為 DEBUG。如需詳細資訊，請參閱《 AWS Lambda 開發人員指南》中的使用 AWS Lambda 環境變數。

> **Note**
>
> AWS Lambda 在 Amazon CloudWatch Logs 中記錄 Lambda 延伸模組和 Lambda 函數的操作詳細資訊。

環境變數	詳細資訊	必要	有效值	預設值
SSM_PARAMETER_STORE_TIMEOUT_MILLIS	對 Parameter Store 的請求逾時 (以毫秒為單位)。 0 (零) 值表示沒有逾時。	否	所有整數	0 (零)

環境變數	詳細資訊	必要	有效值	預設值
SECRETS_MANAGER_TIMEOUT_MILLIS	對 Secrets Manager 的請求逾時 (以毫秒為單位)。 0 (零) 值表示沒有逾時。	否	所有整數	0 (零)
SSM_PARAMETER_STORE_TTL	快取中參數失效之前的最大有效期限 (以秒為單位)。0 (零) 值表示應略過快取。如果 PARAMETERS_SECRETS_EXTENSION_CACHE_SIZE 的值為 0 (零)，則會忽略此變數。	否	0 (零) 至 300 秒 (五分鐘)	300 秒 (五分鐘)
SECRETS_MANAGER_TTL	快取中機密失效之前的最大有效期限 (以秒為單位)。0 (零) 值表示略過快取。如果 PARAMETERS_SECRETS_EXTENSION_CACHE_SIZE 的值為 0 (零)，則會忽略此變數。	否	0 (零) 至 300 秒 (五分鐘)	300 秒 (5 分鐘)

環境變數	詳細資訊	必要	有效值	預設值
PARAMETERS_SECRETS_EXTENSION_CACHE_ENABLED	確定是否啟用此延伸的快取。有效值：TRUE | FALSE	否	TRUE | FALSE	TRUE
PARAMETERS_SECRETS_EXTENSION_CACHE_SIZE	在項目數方面的快取大小上限。0 (零) 值表示略過快取。如果兩個快取 TTL 值都是 0 (零)，則會忽略此變數。	否	0 (零) 至 1000	1000
PARAMETERS_SECRETS_EXTENSION_HTTP_PORT	本機 HTTP 伺服器的連接埠。	否	1 - 65535	2773
PARAMETERS_SECRETS_EXTENSION_MAX_CONNECTIONS	延伸用於向 Parameter Store 或 Secrets Manager 提出請求的 HTTP 用戶端連線數量上限。這是針對 Secrets Manager 用戶端和 Parameter Store 用戶端對後端服務建立連線數目的每個用戶端組態。	否	最少 1 個；沒有上限。	3

環境變數	詳細資訊	必要	有效值	預設值
PARAMETERS_SECRETS_EXTENSION_LOG_LEVEL	延伸日誌中報告的詳細資訊層級。 我們建議您在設定和測試延伸時，使用 DEBUG 了解快取組態的大多數詳細資訊。 Lambda 操作的日誌會自動推送至關聯的 CloudWatch Logs 日誌群組。	否	DEBUG | WARN | ERROR | NONE | INFO	INFO

使用 AWS Systems ManagerParameter Store和 AWS Secrets Manager 延伸模組的範例命令

本節中的範例示範與 Parameter Store和 AWS Secrets Manager 延伸模組搭配使用的 AWS Systems Manager API 動作。

Parameter Store 的範例命令

Lambda 延伸會使用 GetParameter API 動作的唯讀存取權。

若要呼叫此動作，請進行類似下列內容的 HTTP GET 呼叫。此命令格式可讓您存取標準參數層中的參數。

```
GET http://localhost:port/systemsmanager/parameters/get?name=parameter-name&version=version&label=label&withDecryption={true|false}
```

在此範例中，*parameter-name* 代表完整參數名稱：例如 MyParameter，針對不在階層中的參數；又例如 %2FDev%2FProduction%2FEast%2FProject-ABC%2FMyParameter，針對名為 /Dev/Production/East/Project-ABC/MyParameter 且屬於階層的一部分的參數。

> **Note**
>
> 使用 GET 呼叫時，必須針對 HTTP 編碼參數值才能保留特殊字元。例如，不要格式化 /a/b/c 之類的階層式路徑，而是將可解譯為 URL 一部分的字元進行編碼，例如 %2Fa%2Fb%2Fc。

version 和 *label* 是可用於 GetParameter 動作的選取器。

```
GET http://localhost:port/systemsmanager/parameters/get/?name=MyParameter&version=5
```

若要呼叫階層中的參數，請進行類似下列所示的 HTTP GET 呼叫。

```
GET http://localhost:port/systemsmanager/parameters/get?name=%2Fa%2Fb%2F&label=release
```

若要呼叫公有 (全域) 參數，請進行類似下列所示的 HTTP GET 呼叫。

```
GET http://localhost:port/systemsmanager/parameters/get/?name=%2Faws%2Fservice%20list
%2F…
```

若要使用 Parameter Store 參考來對 Secrets Manager 機密進行 HTTP GET 呼叫，請進行類似下列所示的 HTTP GET 呼叫。

```
GET http://localhost:port/systemsmanager/parameters/get?name=%2Faws%2Freference
%2Fsecretsmanager%2F…
```

若要使用 Amazon Resource Name (ARN) 做為參數進行呼叫，請進行類似於下列所示的 HTTP GET 呼叫。

```
GET http://localhost:port/systemsmanager/parameters/get?name=arn:aws:ssm:us-
east-1:123456789012:parameter/MyParameter
```

若要透過解密方式存取 SecureString 參數進行呼叫，請進行類似下列所示的 HTTP GET 呼叫。

```
GET http://localhost:port/systemsmanager/parameters/get?
name=MyParameter&withDecryption=true
```

您可以省略 withDecryption 或明確地將其設定為 false，藉此來指定不解密的參數。您還可以指定版本或標籤，但不能同時指定兩者。如果這樣做，則只會使用 URL 中問號 (?) 後面的第一項。

AWS 參數和秘密 Lambda 延伸ARNs

下表針對支援的架構和區域提供了延伸 ARN。

主題

- 適用於 x86_64 和 x86 架構的延伸 ARN
- 適用於 ARM64 和 Mac with Apple silicon 架構的延伸模組 ARN

適用於 x86_64 和 x86 架構的延伸 ARN

上次更新日期：2025 年 6 月 26 日

區域	ARN
美國東部 (俄亥俄)	`arn:aws:lambda:us-east-2:590474943231:layer:AWS-Parameters-and-Secrets-Lambda-Extension:22`
美國東部 (維吉尼亞北部)	`arn:aws:lambda:us-east-1:177933569100:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18`
美國西部 (加利佛尼亞北部)	`arn:aws:lambda:us-west-1:997803712105:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18`
美國西部 (奧勒岡)	`arn:aws:lambda:us-west-2:345057560386:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18`
非洲 (開普敦)	`arn:aws:lambda:af-south-1:317013901791:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18`

區域	ARN
Asia Pacific (Hong Kong)	arn:aws:lambda:ap-east-1:768336418462:layer:AWS-Parameters-and-Secrets-Lambda-Extension:20
亞太區域（台北）	arn:aws:lambda:ap-east-2:890742577149:layer:AWS-Parameters-and-Secrets-Lambda-Extension:6
亞太區域 (海德拉巴)	arn:aws:lambda:ap-south-2:070087711984:layer:AWS-Parameters-and-Secrets-Lambda-Extension:15
亞太區域 (雅加達)	arn:aws:lambda:ap-southeast-3:490737872127:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
亞太區域 (墨爾本)	arn:aws:lambda:ap-southeast-4:090732460067:layer:AWS-Parameters-and-Secrets-Lambda-Extension:8
亞太區域 (馬來西亞)	arn:aws:lambda:ap-southeast-5:381492012281:layer:AWS-Parameters-and-Secrets-Lambda-Extension:7
亞太區域 (孟買)	arn:aws:lambda:ap-south-1:176022468876:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18

區域	ARN
亞太區域 (大阪)	arn:aws:lambda:ap-northeast-3:576959938190:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
亞太區域 (首爾)	arn:aws:lambda:ap-northeast-2:738900069198:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
亞太區域 (新加坡)	arn:aws:lambda:ap-southeast-1:044395824272:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
亞太區域 (悉尼)	arn:aws:lambda:ap-southeast-2:665172237481:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
亞太區域 （泰國）	arn:aws:lambda:ap-southeast-7:941377119484:layer:AWS-Parameters-and-Secrets-Lambda-Extension:6
亞太區域 (東京)	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

區域	ARN
加拿大西部 (卡加利)	arn:aws:lambda:ca-west-1:243964427225:layer:AWS-Parameters-and-Secrets-Lambda-Extension:8
中國 (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
中國 (寧夏)	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
歐洲 (法蘭克福)	arn:aws:lambda:eu-central-1:187925254637:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
歐洲 (愛爾蘭)	arn:aws:lambda:eu-west-1:015030872274:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
歐洲 (倫敦)	arn:aws:lambda:eu-west-2:133256977650:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18

區域	ARN
歐洲 (米蘭)	arn:aws:lambda:eu-south-1:325218067255:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
Europe (Paris)	arn:aws:lambda:eu-west-3:780235371811:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
歐洲 (西班牙) 區域	arn:aws:lambda:eu-south-2:524103009944:layer:AWS-Parameters-and-Secrets-Lambda-Extension:17
歐洲 (斯德哥爾摩)	arn:aws:lambda:eu-north-1:427196147048:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
以色列 (特拉維夫)	arn:aws:lambda:il-central-1:148806536434:layer:AWS-Parameters-and-Secrets-Lambda-Extension:8
歐洲 (蘇黎世) 區域	arn:aws:lambda:eu-central-2:772501565639:layer:AWS-Parameters-and-Secrets-Lambda-Extension:15
墨西哥 （ 中部 ） 區域	arn:aws:lambda:mx-central-1:241533131596:layer:AWS-Parameters-and-Secrets-Lambda-Extension:5

區域	ARN
Middle East (Bahrain)	arn:aws:lambda:me-south-1:832021897121:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
中東 (阿拉伯聯合大公國)	arn:aws:lambda:me-central-1:858974508948:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
南美洲 (聖保羅)	arn:aws:lambda:sa-east-1:933737806257:layer:AWS-Parameters-and-Secrets-Lambda-Extension:18
AWS GovCloud （美國東部）	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 （美國西部）	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

適用於 ARM64 和 Mac with Apple silicon 架構的延伸模組 ARN

上次更新日期：2025 年 6 月 26 日

區域	ARN
美國東部 (俄亥俄)	arn:aws:lambda:us-east-2:590474943231:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:22
美國東部 (維吉尼亞北部)	arn:aws:lambda:us-east-1:177933569100:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:18
美國西部 (加利佛尼亞北部) 區域	arn:aws:lambda:us-west-1:997803712105:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:15
美國西部 (奧勒岡)	arn:aws:lambda:us-west-2:345057560386:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:18
非洲 (開普敦) 區域	arn:aws:lambda:af-south-1:317013901791:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:15
亞太區域 (香港) 區域	arn:aws:lambda:ap-east-1:768336418462:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:17
亞太區域 （ 台北 ）	arn:aws:lambda:ap-east-2:890742577149:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:2

區域	ARN
亞太區域 (海德拉巴)	arn:aws:lambda:ap-south-2:070087711984:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7
亞太區域 (雅加達)	arn:aws:lambda:ap-southeast-3:490737872127:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:15
亞太區域 (墨爾本)	arn:aws:lambda:ap-southeast-4:090732460067:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7
亞太區域 (馬來西亞)	arn:aws:lambda:ap-southeast-5:381492012281:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7
亞太區域 (孟買)	arn:aws:lambda:ap-south-1:176022468876:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:18
亞太區域 (大阪)	arn:aws:lambda:ap-northeast-3:576959938190:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:15
亞太 (首爾) 區域	arn:aws:lambda:ap-northeast-2:738900069198:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:15

區域	ARN
亞太區域 (新加坡)	arn:aws:lambda:ap-southeast-1:044395824272:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:18
亞太區域 (悉尼)	arn:aws:lambda:ap-southeast-2:665172237481:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:18
亞太區域 （ 泰國 ）	arn:aws:lambda:ap-southeast-7:941377119484:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:5
亞太區域 (東京)	arn:aws:lambda:ap-northeast-1:133490724326:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:18
加拿大 (中部) 區域	arn:aws:lambda:ca-central-1:200266452380:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:17
加拿大西部 (卡加利)	arn:aws:lambda:ca-west-1:243964427225:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7

區域	ARN
中國 (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
中國 (寧夏)	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
歐洲 (法蘭克福)	arn:aws:lambda:eu-central-1:187925254637:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:18
歐洲 (愛爾蘭)	arn:aws:lambda:eu-west-1:015030872274:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:18
歐洲 (倫敦)	arn:aws:lambda:eu-west-2:133256977650:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:18

區域	ARN
Europe (Milan) Region	arn:aws:lambda:eu-south-1:325218067255:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:15
歐洲 (巴黎) 區域	arn:aws:lambda:eu-west-3:780235371811:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:15
歐洲 (西班牙) 區域	arn:aws:lambda:eu-south-2:524103009944:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:9
歐洲 (斯德哥爾摩) 區域	arn:aws:lambda:eu-north-1:427196147048:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:15
以色列 (特拉維夫)	arn:aws:lambda:il-central-1:148806536434:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7
歐洲 (蘇黎世) 區域	arn:aws:lambda:eu-central-2:772501565639:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7
墨西哥 （ 中部 ） 區域	arn:aws:lambda:mx-central-1:241533131596:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:4

區域	ARN
Middle East (Bahrain) Region	arn:aws:lambda:me-south-1:832021897121:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:15
中東 (阿拉伯聯合大公國)	arn:aws:lambda:me-central-1:858974508948:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:7
南美洲 (聖保羅) 區域	arn:aws:lambda:sa-east-1:933737806257:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:15
AWS GovCloud （美國東部）	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 （美國西部）	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

與其他產品及服務整合

AWS Systems Manager 具有下表所示產品和服務的內建整合。

Ansible	Ansible 是 IT 自動化平台，可讓您更輕鬆地部署應用程式和系統。 Systems Manager 提供 Systems Manager 文件 (SSM 文件) `AWS-ApplyAnsiblePlaybooks` ，其允許您建立可執行 Ansible Playbook 的 State Manager 關聯。 進一步了解 建立執行 Ansible 手冊的關聯
Chef	Chef 是 IT 自動化工具，可讓您更輕鬆地部署應用程式和系統。 Systems Manager 提供 `AWS-ApplyChefRecipes` SSM 文件，可讓您在執行 Chef配方 AWS Systems Manager的 State Manager工具 中建立關聯。 進一步了解 建立執行 Chef 配方的關聯 Systems Manager 也與 Chef InSpec 設定檔整合，可讓您執行合規掃描，以及檢視合規和不合規的節點。 進一步了解 將 Chef InSpec 設定檔與 Systems Manager 合規搭配使用
GitHub	GitHub 為軟體開發版本控制和協作提供託管服務。

	Systems Manager 提供 SSM 文件 `AWS-RunDocument` (允許您執行存放在 GitHub 中的其他 SSM 文件)，以及 SSM 文件 `AWS-RunRemoteScript` (允許您執行存放在 GitHub 中的指令碼)。 進一步了解 • 從遠端位置執行 文件 • 從 GitHub 執行指令碼
Jenkins	Jenkins 是開放原始碼的自動化伺服器，讓開發人員能夠可靠地建置、測試及部署其軟體。 Automation 是 Systems Manager 中的工具，可用來做為建置後步驟，將應用程式版本預先安裝至 Amazon Machine Images(AMIs)。 進一步了解 使用 Automation 和 Jenkins 更新 AMIs
ServiceNow	ServiceNow 是企業服務管理系統，方便您管理 IT 服務和操作。 Automation、Change Manager、Invents Manager 和 Systems Manager 中的所有 OpsCenter工具，ServiceNow都使用 Service Management Connector AWS 與 整合。透過此整合，您可以從 檢視、建立、更新、新增通訊和解決 AWS 支援 案例ServiceNow。 進一步了解 與 ServiceNow 整合

Terraform	HashiCorp Terraform 是一種開放原始碼基礎設施即程式碼 (IaC) 軟體工具，可提供命令列介面 (CLI) 工作流程來管理各種雲端服務。對於 Systems Manager，可以使用 Terraform 來管理或佈建下列項目： 資源 • 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 資料來源 • aws_ssm_document • aws_ssm_instances • ssm_maintenance_windows • aws_ssm_parameter • aws_ssm_parameters_by_path • aws_ssm_patch_baseline

主題

- 從 GitHub 執行指令碼
- 將 Chef InSpec 設定檔與 Systems Manager 合規搭配使用

- 與 ServiceNow 整合

從 GitHub 執行指令碼

本主題說明如何利用預先定義的 Systems Manager 文件 (SSM 文件) `AWS-RunRemoteScript` 從 GitHub 下載指令碼，包括 Ansible Playbook、Python、Ruby 與 PowerShell 指令碼。透過使用此 SSM 文件，您不再需要手動將指令碼移植到 Amazon Elastic Compute Cloud (Amazon EC2) 或將其包裝在 SSM 文件中。 與 AWS Systems Manager 整合會將基礎設施GitHub提升為程式碼，這可縮短管理節點所需的時間，同時將整個機群的組態標準化。

您也可以建立自訂 SSM 文件，以允許從遠端位置下載和執行指令碼，或是其他 SSM 文件。如需詳細資訊，請參閱建立複合文件。

另外，您也能下載包含多個指令碼的目錄。當您在目錄中執行主要指令碼時，Systems Manager 會一併執行涵蓋在該目錄內的任何參考的指令碼。

從 GitHub 執行指令碼時，請注意以下重要詳細資訊。

- Systems Manager 不會驗證指令碼是否能夠在節點上執行。請確認節點上已安裝必要軟體，然後再下載和執行指令碼。或者，您可以建立複合文件，使用 Run Command或 State Manager中的工具來安裝軟體 AWS Systems Manager，然後下載並執行指令碼。
- 您要負責確保符合所有 GitHub 需求。包括視需求重新整理存取字符。確保沒有超出已驗證或未驗證請求的數量。如需詳細資訊，請參閱GitHub文件。
- 不支援 GitHub Enterprise 儲存庫。

主題

- 從 GitHub 執行 Ansible Playbook
- 從 GitHub 執行 Python 指令碼

從 GitHub 執行 Ansible Playbook

本節所涵蓋的程序能協助您使用主控台或 AWS Command Line Interface (AWS CLI) 從 GitHub 執行 Ansible Playbook。

開始之前

如果您計劃執行存放在私有GitHub儲存庫中的指令碼，請為您的GitHub安全存取字符建立 AWS Systems Manager `SecureString` 參數。若透過 SSH 手動傳遞權杖，則無法存取私有 GitHub 儲

存庫中的指令碼。您必須將存取字符做為 Systems Manager SecureString 參數傳遞。如需建立 SecureString 參數的詳細資訊，請參閱 在 Systems Manager 中建立 Parameter Store 參數。

從 GitHub 執行 Ansible Playbook (主控台)

從 GitHub 執行 Ansible Playbook

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Run Command。
3. 選擇執行命令。
4. 在 Command document (命令文件) 清單，請選擇 **AWS-RunRemoteScript**。
5. 在命令參數中，執行以下操作：

   - 在來源類型中，選取 GitHub。
   - 在 Source Info (來源資訊) 方塊中，按照以下格式輸入所需資訊，藉此存取來源。

   ```
   {
     "owner": "owner_name",
     "repository": "repository_name",
     "getOptions": "branch:branch_name",
     "path": "path_to_scripts_or_directory",
     "tokenInfo": "{{ssm-secure:SecureString_parameter_name}}"
   }
   ```

   此範例會下載名為 webserver.yml 的檔案。

   ```
   {
       "owner": "TestUser1",
       "repository": "GitHubPrivateTest",
       "getOptions": "branch:myBranch",
       "path": "scripts/webserver.yml",
       "tokenInfo": "{{ssm-secure:mySecureStringParameter}}"
   }
   ```

   > **Note**
   >
   > 只有當 SSM 文件存放於 master 以外的分支時，才需要 "branch"。

若要使用存放庫中特定「遞交」中的指令碼版本，請使用 commitID 與 getOptions 來代替 branch。例如：
"getOptions": "commitID:bbc1ddb94...b76d3bEXAMPLE",

- 在 Command Line (命令列) 欄位中，輸入指令碼執行參數。請見此處範例。

  ansible-playbook -i “localhost,” --check -c local webserver.yml

- (選用) 在 Working Directory (工作目錄) 欄位中，輸入節點上的目錄名稱；您將下載指令碼至該目錄並予以執行。
- (選用) 在執行逾時中，指定指令碼命令執行失敗前，系統的等待時間 (秒)。

6. 在 Targets (目標) 區段中，透過手動指定標籤、選取執行個體或邊緣裝置，或指定資源群組，選擇您要執行這項操作的受管節點。

> **Tip**
>
> 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

7. 對於其他參數：

- 在 Comment (註解) 中，輸入此命令的相關資訊。
- 在逾時 (秒) 中，指定在命令執行全面失敗之前，系統要等候的秒數。

8. 對於 Rate control (速率控制)：

- 在並行中，指定可同時執行命令的受管節點數目或百分比。

> **Note**
>
> 如果透過指定套用至受管節點的標籤或指定 AWS 資源群組選取了目標，且您不確定會以多少個受管節點為目標，則透過指定百分比限制可以同時執行文件之目標的數量。

- 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比之節點上的命令失敗之後，停止在其他受管節點上執行命令。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止傳送命令。仍在處理命令的受管節點也可能會傳送錯誤。

9. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取將命令輸出寫入至 S3 儲存貯體方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

> **Note**
>
> 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給執行個體之執行個體設定檔 (適用於 EC2 執行個體) 或 IAM 服務角色 (啟用混合模式的機器) 的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確定與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

10. 在SNS 通知區段中，如果您要傳送有關命令執行狀態的通知，請選取啟用 SNS 通知核取方塊。

    如需為 Run Command 設定 Amazon SNS 通知的詳細資訊，請參閱 使用 Amazon SNS 通知監控 Systems Manager 狀態變更。

11. 選擇執行。

GitHub 使用 從 執行 Ansible Playbook AWS CLI

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

2. 執行以下命令，即可從 GitHub 下載指令碼並予以執行。

```
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"]}'
```

   以下是在本機 Linux 機器上執行的範例命令。

```
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"]}'
```

從 GitHub 執行 Python 指令碼

本節包含使用 AWS Systems Manager 主控台或 () GitHub來協助您從 執行 Python AWS Command Line Interface 指令碼的程序AWS CLI。

從 GitHub 執行 Python 指令碼 (主控台)

從 GitHub 執行 Python 指令碼

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 Run Command。
3. 選擇執行命令。
4. 在 Command document (命令文件) 清單，請選擇 **AWS-RunRemoteScript**。
5. 在 Command parameters (命令參數) 中，執行以下操作：
   - 在來源類型中，選取 GitHub。
   - 在 Source Info (來源資訊) 方塊中，按照以下格式輸入所需資訊，藉此存取來源：

```
{
    "owner": "owner_name",
    "repository": "repository_name",
    "getOptions": "branch:branch_name",
    "path": "path_to_document",
    "tokenInfo": "{{ssm-secure:SecureString_parameter_name}}"
}
```

下列範例會下載名為 complex-script 的指令碼目錄。

```
{
    "owner": "TestUser1",
    "repository": "SSMTestDocsRepo",
    "getOptions": "branch:myBranch",
    "path": "scripts/python/complex-script",
    "tokenInfo": "{{ssm-secure:myAccessTokenParam}}"
```

```
}
```

> **Note**
>
> 只有當指令碼存放在 master 以外的分支中時，才需要 "branch"。
> 若要使用存放庫中特定「遞交」中的指令碼版本，請使用 commitID 與 getOptions 來代替 branch。例如：
> "getOptions": "commitID:bbc1ddb94...b76d3bEXAMPLE",

- 針對 Command Line (命令列)，輸入指令碼執行參數。請見此處範例。

```
mainFile.py argument-1 argument-2
```

此範例會執行 mainFile.py，其稍後可執行 complex-script 目錄中的其他指令碼。

- (選用) 針對 Working Directory (工作目錄)，輸入節點上的目錄名稱；您將下載指令碼至該目錄並予以執行。
- (選用) 針對 Execution Timeout (執行逾時)，指定指令碼命令執行失敗前，系統的等待時間 (秒)。

6. 在 Targets (目標) 區段中，透過手動指定標籤、選取執行個體或邊緣裝置，或指定資源群組，選擇您要執行這項操作的受管節點。

> **Tip**
>
> 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。

7. 對於其他參數：
   - 在 Comment (註解) 中，輸入此命令的相關資訊。
   - 在逾時 (秒) 中，指定在命令執行全面失敗之前，系統要等候的秒數。
8. 對於 Rate control (速率控制)：
   - 在並行中，指定可同時執行命令的受管節點數目或百分比。

> **Note**
>
> 如果透過指定套用至受管節點的標籤或指定 AWS 資源群組選取了目標，且您不確定會以多少個受管節點為目標，則透過指定百分比限制可以同時執行文件之目標的數量。

- 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比之節點上的命令失敗之後，停止在其他受管節點上執行命令。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止傳送命令。仍在處理命令的受管節點也可能會傳送錯誤。

9. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取將命令輸出寫入至 S3 儲存貯體方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

> **Note**
>
> 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給執行個體之執行個體設定檔 (適用於 EC2 執行個體) 或 IAM 服務角色 (啟用混合模式的機器) 的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確定與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

10. 在SNS 通知區段中，如果您要傳送有關命令執行狀態的通知，請選取啟用 SNS 通知核取方塊。

    如需為 Run Command 設定 Amazon SNS 通知的詳細資訊，請參閱 使用 Amazon SNS 通知監控 Systems Manager 狀態變更。

11. 選擇執行。

GitHub 使用 從 執行 Python 指令碼 AWS CLI

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

2. 執行以下命令，即可從 GitHub 下載指令碼並予以執行。

```
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"]}'
```

請見此處範例。

```
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 "]}'
```

此範例會下載名為 `complex-script` 的指令碼目錄。`commandLine` 項目會執行 `mainFile.py`，其稍後可執行 `complex-script` 目錄中的其他指令碼。

將 Chef InSpec 設定檔與 Systems Manager 合規搭配使用

AWS Systems Manager 與 整合Chef InSpec。 Chef InSpec 是開放原始碼測試架構，可讓您建立人類可讀設定檔，以存放在 GitHub或 Amazon Simple Storage Service (Amazon S3) 中。然後，您可以使用 Systems Manager 執行合規掃描，檢視合規與不合規的節點。描述檔能滿足適合運算環境的安全、合規或政策需求。例如，您可以使用 中的工具 Compliance 掃描節點時，建立執行下列檢查的設定檔 AWS Systems Manager：

- 檢查特定連接埠是開啟或關閉狀態。
- 檢查特定應用程式是否正在執行。
- 檢查是否已安裝特定套件。
- 檢查特定屬性的 Windows 登錄機碼。

對於您以 Systems Manager 管理的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、內部部署伺服器或虛擬機器 (VM)，您可以建立 InSpec 描述檔。以下 Chef InSpec 範例設定檔會檢查連接埠 22 是否為開啟狀態。

```
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 內含資源集合，能協助您快速編寫檢查作業並稽核控制項。為了以 Ruby 撰寫這些控制項，InSpec 會採用 InSpec 特定領域語言 (DSL)。此外，您也能使用由廣大 InSpec 使用者社群所建立的描述檔。例如，GitHub 上的 DevSec chef-os-hardening 專案包含數十種設定檔，皆可保障節點的安全。您能夠撰寫設定檔，並將其存放在 GitHub 或 Amazon S3 中。

運作方式

以下程序將說明如何結合使用 InSpec 描述檔與合規：

1. 請找出要使用的預先定義 InSpec 描述檔，或建立自己的描述檔。您可以使用 GitHub 上預先定義的設定檔，立刻開始操作。有關如何建立自有 InSpec 設定檔的資訊，請參閱 ChefChef InSpec Profiles。
2. 在公有或私有 GitHub 儲存庫或 S3 儲存貯體中存放設定檔。
3. 請使用 Systems Manager 文件 (SSM 文件) `AWS-RunInspecChecks`，以透過 InSpec 描述檔執行合規操作。您可以使用 Run Command、 中的工具 AWS Systems Manager、 進行隨需掃描來開始合規掃描，也可以使用 中的工具State Manager，來排程定期合規掃描 AWS Systems Manager。
4. 利用合規 API 或合規主控台來辨識未合規的節點。

> **Note**
>
> 記下以下資訊。
>
> - Chef 會採用節點上的用戶端處理設定檔。因此，您不需要另外安裝用戶端。Systems Manager 執行 SSM 文件 `AWS-RunInspecChecks` 時，系統即會檢查是否已安裝用戶端。若尚未安裝，Systems Manager 就會在掃描期間安裝 Chef 用戶端，然後在掃描完成後解除安裝該用戶端。
> - 執行 SSM 文件 `AWS-RunInspecChecks` (如本主題所述) 會指派類型 `Custom:Inspec` 的合規項目到每個目標節點。若要指派此合規類型，文件會呼叫 PutComplianceItems API 操作。

執行 InSpec 合規掃描

本節包含如何使用 Systems Manager 主控台和 AWS Command Line Interface () 執行 InSpec 合規掃描的相關資訊AWS CLI。主控台程序會說明如何設定 State Manager，以執行掃描作業。程序 AWS CLI 說明如何設定 Run Command 以執行掃描。

使用 State Manager (主控台) 執行 InSpec 合規掃描

使用 AWS Systems Manager 主控台透過State Manager執行 InSpec 合規掃描

1. 在 https：//https://console.aws.amazon.com/systems-manager/ 開啟 AWS Systems Manager 主控台。
2. 在導覽窗格中，選擇 State Manager。
3. 選擇 Create association (建立關聯)。
4. 在 Provide association details (提供關聯詳細資訊) 區段中，輸入一個名稱。
5. 在 Document (文件) 清單中，請選擇 **AWS-RunInspecChecks**。
6. 在 Document version (文件版本) 清單中，選擇 Latest at runtime (執行時間的最新版本)。
7. 在參數區段的來源類型清單中，選擇 GitHub 或 S3。

   若您選擇 GitHub，請在來源資訊欄位中輸入公有或私有 GitHub 儲存庫中的 InSpec 設定檔路徑。此處是由 Systems Manager 團隊提供的公有描述檔範例路徑，其來自以下位置：https://github.com/awslabs/amazon-ssm/tree/master/Compliance/InSpec/PortCheck。

   ```
   {"owner":"awslabs","repository":"amazon-ssm","path":"Compliance/InSpec/
   PortCheck","getOptions":"branch:master"}
   ```

   如果您選擇 S3，請在 Source Info (來源資訊) 欄位中，輸入 S3 儲存貯體中的 InSpec 描述檔的有效 URL。

   如需有關 Systems Manager 如何與 GitHub 及 Amazon Simple Storage Service (Amazon S3) 整合的詳細資訊，請參閱 從 GitHub 執行指令碼。
8. 在目標區段中，透過手動指定標籤、選取執行個體或邊緣裝置，或指定資源群組，選擇您要執行這項操作的受管節點。

   > **Tip**
   >
   > 如果您預期看到的受管節點未列出，請參閱 疑難排解受管節點的可用性 以取得疑難排解秘訣。
9. 在 Specify schedule (指定排程) 區段中，使用排程建置器選項來建立排程，以指定合規掃描的執行時間。
10. 對於速率控制：

- 在並行中，指定可同時執行命令的受管節點數目或百分比。

> **Note**
>
> 如果透過指定套用至受管節點的標籤或指定 AWS 資源群組選取了目標，且您不確定會以多少個受管節點為目標，則透過指定百分比限制可以同時執行文件之目標的數量。

- 在 Error threshold (錯誤閾值) 中，指定在特定數目或百分比之節點上的命令失敗之後，停止在其他受管節點上執行命令。例如，如果您指定三個錯誤，則 Systems Manager 會在收到第四個錯誤時停止傳送命令。仍在處理命令的受管節點也可能會傳送錯誤。

11. (選用) 針對輸出選項，若要將命令輸出儲存至檔案，請選取將命令輸出寫入至 S3 儲存貯體方塊。在方塊中輸入儲存貯體和字首 (資料夾) 名稱。

> **Note**
>
> 授予能力以將資料寫入至 S3 儲存貯體的 S3 許可，會是指派給執行個體之執行個體設定檔 (適用於 EC2 執行個體) 或 IAM 服務角色 (啟用混合模式的機器) 的許可，而不是執行此任務之 IAM 使用者的許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可或建立混合環境的 IAM 服務角色。此外，如果指定的 S3 儲存貯體位於不同的 中 AWS 帳戶，請確定與受管節點相關聯的執行個體設定檔或 IAM 服務角色具有寫入該儲存貯體的必要許可。

12. 選擇 Create Association (建立關聯)。系統會隨即建立關聯，並自動執行合規掃描。
13. 請稍候幾分鐘，等待掃描作業完成。接著，在導覽窗格中選擇 Compliance (合規)。
14. 在 Corresponding managed instances (對應的受管執行個體) 中，尋找 Compliance Type (合規類型) 欄位是 Custom:Inspec (Custom:Inspec) 的節點。
15. 選擇節點 ID，藉此檢視未合規狀態的詳細資訊。

使用 Run Command (AWS CLI) 執行 InSpec 合規掃描

1. 如果您尚未安裝並設定 AWS Command Line Interface (AWS CLI)，請執行此作業。

   如需相關資訊，請參閱安裝或更新最新版本的 AWS CLI。

2. 執行下列其中一個命令，即可從 GitHub 或 Amazon S3 執行 InSpec 設定檔。

   命令接受下列參數：

- sourceType：GitHub 或 Amazon S3
- sourceInfo：GitHub 或 S3 儲存貯體中的 InSpec 設定檔資料夾的 URL。該資料夾必須包含基本 InSpec 檔案 (*.yml)，以及所有相關的控制項 (*.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"}"]}'
```

請見此處範例。

```
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 Simple Storage Service (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\"}"]}'
```

請見此處範例。

```
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. 執行下列命令，以檢視合規掃描摘要。

```
aws ssm list-resource-compliance-summaries --filters
 Key=ComplianceType,Values=Custom:Inspec
```

4. 執行下列命令，以檢視不相容的節點的詳細資訊。

```
aws ssm list-compliance-items --resource-ids node_ID --resource-type
 ManagedInstance --filters Key=DocumentName,Values=AWS-RunInspecChecks
```

與 ServiceNow 整合

ServiceNow 提供雲端服務管理系統，可建立和管理組織層級的工作流程，例如 IT 服務、票證系統和支援。 AWS Service Management Connector ServiceNow與 Systems Manager 整合，以佈建、管理和操作 AWS 來自 的資源ServiceNow。您可以使用 AWS Service Management Connector ServiceNow與 Automation、、Change ManagerInvent Manager 和 OpsCenter整合，所有 工具皆在其中 AWS Systems Manager。

可以使用 ServiceNow 執行下列任務：

- 從 Systems Manager 執行自動化手冊。
- 從 Systems Manager OpsItems 檢視、更新及解決事件。
- 透過 Systems Manager OpsCenter 檢視和管理營運項目，例如事件。
- 從預先核准的變更範本的精選清單中，檢視並執行 Systems Manager 變更要求。
- 與 Incident Manager 整合，以管理和解決涉及 AWS 託管應用程式的事件。

> **Note**
>
> 如需有關如何與 整合的資訊ServiceNow，請參閱 Service AWS Management Connector 管理員指南中的設定 AWS 服務整合。

AWS Systems Manager 參考

以下資訊和主題可協助您更好地實作 AWS Systems Manager 解決方案。

Principal

在 AWS Identity and Access Management (IAM) 中，您可以使用主體政策元素授予或拒絕對資源的服務存取權。Systems Manager 的委託人政策元素值為 `ssm.amazonaws.com`。

支援的 AWS 區域 和 端點

請參閱《Amazon Web Services 一般參考》中的 Systems Manager 服務端點一節。

Service Quotas

請參閱《Amazon Web Services 一般參考》中的 Systems Manager 服務配額一節。

API 參考

請參閱 AWS Systems Manager API 參考。

AWS CLI 命令參考

請參閱 AWS Systems ManagerAWS CLI 命令參考的 章節。

AWS Tools for PowerShell Cmdlet 參考

請參閱 AWS Systems ManagerAWS Tools for PowerShell Cmdlet 參考一節。

GitHub 上的 SSM Agent 儲存庫

請參閱 aws/amazon-ssm-agent。

詢問問題

AWS re:Post 中的 Systems Manager 問題

AWS 新聞部落格

管理工具

更多參考主題

- 搭配 AWS SDK 使用此服務
- 參考：用於修補操作的 Amazon S3 儲存貯體
- 參考：Systems Manager 的 Amazon EventBridge 事件模式和類型

- 參考：Systems Manager 的 Cron 和 Rate 運算式
- 參考：ec2messages、ssmmessages 和其他 API 操作
- 參考：Systems Manager 的日期和時間字串格式

搭配 AWS SDK 使用此服務

AWS 軟體開發套件 (SDKs) 適用於許多熱門的程式設計語言。每個 SDK 都提供 API、程式碼範例和說明文件，讓開發人員能夠更輕鬆地以偏好的語言建置應用程式。

SDK 文件	代碼範例
適用於 C++ 的 AWS SDK	適用於 C++ 的 AWS SDK 程式碼範例
AWS CLI	AWS CLI 程式碼範例
適用於 Go 的 AWS SDK	適用於 Go 的 AWS SDK 程式碼範例
適用於 Java 的 AWS SDK	適用於 Java 的 AWS SDK 程式碼範例
適用於 JavaScript 的 AWS SDK	適用於 JavaScript 的 AWS SDK 程式碼範例
適用於 Kotlin 的 AWS SDK	適用於 Kotlin 的 AWS SDK 程式碼範例
適用於 .NET 的 AWS SDK	適用於 .NET 的 AWS SDK 程式碼範例
適用於 PHP 的 AWS SDK	適用於 PHP 的 AWS SDK 程式碼範例
AWS Tools for PowerShell	AWS Tools for PowerShell 程式碼範例
適用於 Python (Boto3) 的 AWS SDK	適用於 Python (Boto3) 的 AWS SDK 程式碼範例
適用於 Ruby 的 AWS SDK	適用於 Ruby 的 AWS SDK 程式碼範例
適用於 Rust 的 AWS SDK	適用於 Rust 的 AWS SDK 程式碼範例
適用於 SAP ABAP 的 AWS SDK	適用於 SAP ABAP 的 AWS SDK 程式碼範例
適用於 Swift 的 AWS SDK	適用於 Swift 的 AWS SDK 程式碼範例

可用性範例

找不到所需的內容嗎？ 請使用本頁面底部的提供意見回饋連結申請程式碼範例。

參考：用於修補操作的 Amazon S3 儲存貯體

在執行各種Patch Manager修補操作的過程中， AWS Systems Manager 代理 (SSM Agent) 會存取 Amazon Web Services () 擁有和維護的特定 Amazon Simple Storage Service (Amazon S3) 儲存貯體 AWS。可公開存取這些 S3 儲存貯體，SSM Agent 預設使用 HTTP 呼叫連線到它們。

但是，如果您在 Systems Manager 操作中使用虛擬私有雲端 (VPC) 端點，則必須在適用於 Systems Manager 的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體設定檔中提供明確許可，或在混合多雲端環境中非 EC2 機器的服務角色中提供明確許可。否則，您的資源無法存取這些公有儲存貯體。

此參考主題列出了每個受支援 AWS 區域的修補相關儲存貯體。

如需有關在 EC2 執行個體設定檔中指定這些 S3 儲存貯體的詳細資訊，請參閱SSM Agent 與 AWS 受管 S3 儲存貯體通訊。

如需有關將 VPC 端點與 Systems Manager 搭配使用的詳細資訊，請參閱使用適用於 Systems Manager 的 VPC 端點來改善 EC2 執行個體的安全性。

主題

- 包含用於修補操作之 SSM Command 文件的儲存貯體 (Linux 和 Windows Server)
- 包含用於修補操作之 SSM Command 文件的儲存貯體 (macOS)
- 包含 AWS 受管修補基準快照的儲存貯體

包含用於修補操作之 SSM Command 文件的儲存貯體 (Linux 和 Windows Server)

格式為 `aws-patch-manager-region-unique-suffix` 的儲存貯體包含下列在 Linux 和 Windows Server 作業系統上由 Patch Manager 修補操作所用的文件：

- `AWS-RunPatchBaseline`
- `AWS-RunPatchBaselineAssociation`

- AWS-RunPatchBaselineWithHooks
- AWS-InstanceRebootWithHooks
- AWS-PatchAsgInstance
- AWS-PatchInstanceWithRollback

區域名稱	區域代碼	**aws-patch-manager-*region*-*suffix*** 儲存貯體
美國東部 (俄亥俄)	us-east-2	aws-patch-manager-us-east-2-552881074
美國東部 (維吉尼亞北部)	us-east-1	aws-patch-manager-us-east-1-1970c647d
美國西部 (加利佛尼亞北部)	us-west-1	aws-patch-manager-us-west-1-8badb4304
美國西部 (奧勒岡)	us-west-2	aws-patch-manager-us-west-2-34d7f99f8
非洲 (開普敦)	af-south-1	aws-patch-manager-af-south-1-bdd5f65a9
亞太區域 (香港)	ap-east-1	aws-patch-manager-ap-east-1-632356271
亞太區域 (海德拉巴)	ap-south-2	aws-patch-manager-ap-south-2-32f4b4128
亞太區域 (雅加達)	ap-southeast-3	aws-patch-manager-ap-southeast-3-aa48fc462
亞太區域 (墨爾本)	ap-southeast-4	aws-patch-manager-ap-southeast-4-01e2c40d3
亞太區域 (孟買)	ap-south-1	aws-patch-manager-ap-south-1-cb7c62ff9

區域名稱	區域代碼	**aws-patch-manager-*region*-*suffix*** 儲存貯體
亞太區域 (大阪)	ap-northeast-3	aws-patch-manager-ap-northeast-3-67373598a
亞太區域 (首爾)	ap-northeast-2	aws-patch-manager-ap-northeast-2-10467995c
亞太區域 (新加坡)	ap-southeast-1	aws-patch-manager-ap-southeast-1-7fdfd9ef7
亞太區域 (悉尼)	ap-southeast-2	aws-patch-manager-ap-southeast-2-17283a275
亞太區域 (東京)	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 (卡加利)	ca-west-1	aws-patch-manager-ca-west-1-9e3a4b2f9
歐洲 (法蘭克福)	eu-central-1	aws-patch-manager-eu-central-1-9163fdaaf
歐洲 (愛爾蘭)	eu-west-1	aws-patch-manager-eu-west-1-5522fb710
歐洲 (倫敦)	eu-west-2	aws-patch-manager-eu-west-2-902a2bc74
歐洲 (米蘭)	eu-south-1	aws-patch-manager-eu-south-1-c52f3f594
歐洲 (巴黎)	eu-west-3	aws-patch-manager-eu-west-3-29bf85721

區域名稱	區域代碼	**aws-patch-manager-*region*-*suffix*** 儲存貯體
歐洲 (西班牙)	eu-south-2	aws-patch-manager-eu-south-2-a4cf248b1
歐洲 (斯德哥爾摩)	eu-north-1	aws-patch-manager-eu-north-1-795879e9b
歐洲 (蘇黎世)	eu-central-2	aws-patch-manager-eu-central-2-184ce43c8
以色列 (特拉維夫)	il-central-1	aws-patch-manager-il-central-1-e221cb57b
中東 (巴林)	me-south-1	aws-patch-manager-me-south-1-a53fc9dce
中東 (阿拉伯聯合大公國)	me-central-1	aws-patch-manager-me-central-1-2932f2f80
南美洲 (聖保羅)	sa-east-1	aws-patch-manager-sa-east-1-ddf4b6a09

包含用於修補操作之 SSM Command 文件的儲存貯體 (macOS)

格式為 aws-patchmanager-macos-*region*-*unique-suffix* 的儲存貯體包含下列在 macOS 作業系統上由 Patch Manager 修補操作所用的文件：

- AWS-RunPatchBaseline
- AWS-RunPatchBaselineAssociation
- AWS-RunPatchBaselineWithHooks
- AWS-InstanceRebootWithHooks
- AWS-PatchAsgInstance
- AWS-PatchInstanceWithRollback

區域名稱	區域代碼	**aws-patchmanager-macos-*region-suffix*** 儲存貯體
美國東部 (俄亥俄)	us-east-2	aws-patchmanager-macos-us-east-2-552881074
美國東部 (維吉尼亞北部)	us-east-1	aws-patchmanager-macos-us-east-1-1970c647d
美國西部 (加利佛尼亞北部)	us-west-1	aws-patchmanager-macos-us-west-1-8badb4304
美國西部 (奧勒岡)	us-west-2	aws-patchmanager-macos-us-west-2-34d7f99f8
非洲 (開普敦)	af-south-1	aws-patchmanager-macos-af-south-1-bdd5f65a9
亞太區域 (香港)	ap-east-1	aws-patchmanager-macos-ap-east-1-632356271
亞太區域 (海德拉巴)	ap-south-2	aws-patchmanager-macos-ap-south-2-32f4b4128
亞太區域 (雅加達)	ap-southeast-3	aws-patchmanager-macos-ap-southeast-3-aa48fc462
亞太區域 (墨爾本)	ap-southeast-4	aws-patchmanager-macos-ap-southeast-4-01e2c40d3
亞太區域 (孟買)	ap-south-1	aws-patchmanager-macos-ap-south-1-cb7c62ff9
亞太區域 (大阪)	ap-northeast-3	aws-patchmanager-macos-ap-northeast-3-67373598a
亞太區域 (首爾)	ap-northeast-2	aws-patchmanager-macos-ap-northeast-2-10467995c

區域名稱	區域代碼	**aws-patchmanager-macos-*region*-*suffix*** 儲存貯體
亞太區域 (新加坡)	ap-southeast-1	aws-patchmanager-macos-ap-southeast-1-7fdfd9ef7
亞太區域 (悉尼)	ap-southeast-2	aws-patchmanager-macos-ap-southeast-2-17283a275
亞太區域 (東京)	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 (卡加利)	ca-west-1	aws-patchmanager-macos-ca-west-1-9e3a4b2f9
歐洲 (法蘭克福)	eu-central-1	aws-patchmanager-macos-eu-central-1-9163fdaaf
歐洲 (愛爾蘭)	eu-west-1	aws-patchmanager-macos-eu-west-1-5522fb710
歐洲 (倫敦)	eu-west-2	aws-patchmanager-macos-eu-west-2-902a2bc74
歐洲 (米蘭)	eu-south-1	aws-patchmanager-macos-eu-south-1-c52f3f594
歐洲 (巴黎)	eu-west-3	aws-patchmanager-macos-eu-west-3-29bf85721
歐洲 (西班牙)	eu-south-2	aws-patchmanager-macos-eu-south-2-a4cf248b1
歐洲 (斯德哥爾摩)	eu-north-1	aws-patchmanager-macos-eu-north-1-795879e9b

區域名稱	區域代碼	**aws-patchmanager-macos-*region-suffix*** 儲存貯體
歐洲 (蘇黎世)	eu-central-2	aws-patchmanager-macos-eu-central-2-184ce43c8
以色列 (特拉維夫)	il-central-1	aws-patchmanager-macos-il-central-1-e221cb57b
中東 (巴林)	me-south-1	aws-patchmanager-macos-me-south-1-a53fc9dce
中東 (阿拉伯聯合大公國)	me-central-1	aws-patchmanager-macos-me-central-1-2932f2f80
南美洲 (聖保羅)	sa-east-1	aws-patchmanager-macos-sa-east-1-ddf4b6a09

包含 AWS 受管修補基準快照的儲存貯體

格式為 `patch-baseline-snapshot-`*`region`*或`patch-baseline-snapshot-`*`region-unique-suffix*包含 AWS 受管修補程式基準快照的儲存貯體。如果您使用下列任何 SSM 文件，則需要存取此 S3 儲存貯體：

- `AWS-RunPatchBaseline`
- `AWS-RunPatchBaselineAssociation`
- `AWS-RunPatchBaselineWithHooks`
- `AWS-ApplyPatchBaseline` (舊版 SSM 文件)

區域名稱	區域代碼	**patch-baseline-snapshot-*** 儲存貯體
美國東部 (俄亥俄)	us-east-2	patch-baseline-snapshot-us-east-2

區域名稱	區域代碼	**patch-baseline-snapshot-*** 儲存貯體
美國東部 (維吉尼亞北部)	us-east-1	patch-baseline-snapshot-us-east-1
美國西部 (加利佛尼亞北部)	us-west-1	patch-baseline-snapshot-us-west-1
美國西部 (奧勒岡)	us-west-2	patch-baseline-snapshot-us-west-2
非洲 (開普敦)	af-south-1	patch-baseline-snapshot-af-south-1-tbxdb5b9
亞太區域 (香港)	ap-east-1	patch-baseline-snapshot-ap-east-1
亞太區域 (海德拉巴)	ap-south-2	patch-baseline-snapshot-ap-south-2-50209442
亞太區域 (雅加達)	ap-southeast-3	patch-baseline-snapshot-ap-southeast-3-be0a3174
亞太區域 (墨爾本)	ap-southeast-4	patch-baseline-snapshot-ap-southeast-4-dc6f76ce
亞太區域 (孟買)	ap-south-1	patch-baseline-snapshot-ap-south-1
亞太區域 (大阪)	ap-northeast-3	patch-baseline-snapshot-ap-northeast-3
亞太區域 (首爾)	ap-northeast-2	patch-baseline-snapshot-ap-northeast-2
亞太區域 (新加坡)	ap-southeast-1	patch-baseline-snapshot-ap-southeast-1

區域名稱	區域代碼	**patch-baseline-snapshot-*** 儲存貯體
亞太區域 (悉尼)	ap-southeast-2	patch-baseline-snapshot-ap-southeast-2
亞太區域 (東京)	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 (卡加利)	ca-west-1	patch-baseline-snapshot-ca-west-1
歐洲 (法蘭克福)	eu-central-1	patch-baseline-snapshot-eu-central-1
歐洲 (愛爾蘭)	eu-west-1	patch-baseline-snapshot-eu-west-1
歐洲 (倫敦)	eu-west-2	patch-baseline-snapshot-eu-west-2
歐洲 (米蘭)	eu-south-1	patch-baseline-snapshot-eu-south-1
歐洲 (巴黎)	eu-west-3	patch-baseline-snapshot-eu-west-3
歐洲 (西班牙)	eu-south-2	patch-baseline-snapshot-eu-south-2-df2c9d70
歐洲 (斯德哥爾摩)	eu-north-1	patch-baseline-snapshot-eu-north-1
歐洲 (蘇黎世)	eu-central-2	patch-baseline-snapshot-eu-central-2

區域名稱	區域代碼	**patch-baseline-snapshot-*** 儲存貯體
以色列 (特拉維夫)	il-central-1	patch-baseline-snapshot-il-central-1
中東 (巴林)	me-south-1	patch-baseline-snapshot-me-south-1-uduvl7q8
中東 (阿拉伯聯合大公國)	me-central-1	patch-baseline-snapshot-me-central-1
南美洲 (聖保羅)	sa-east-1	patch-baseline-snapshot-sa-east-1

參考：Systems Manager 的 Amazon EventBridge 事件模式和類型

> **Note**
>
> Amazon EventBridge 是管理事件的首選方式。CloudWatch Events 和 EventBridge 是相同的基礎服務和 API，但 EventBridge 提供了更多功能。您在 CloudWatch 或 EventBridge 中所做的變更將會顯現在每個主控台中。如需詳細資訊，請參閱《Amazon EventBridge 使用者指南》。

使用 Amazon EventBridge，您可以建立符合傳入事件的規則，並將其路由到目標以進行處理。

事件表示您自有應用程式、軟體即服務 (SaaS) 應用程式或 AWS 服務中的環境中有變更的事件。盡可能產生事件。偵測到規則中指定的事件類型之後，EventBridge 會將其路由至指定的目標以進行處理。目標可以包括 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體、 AWS Lambda 函數、Amazon Kinesis 串流、Amazon Elastic Container Service (Amazon ECS) 任務、 AWS Step Functions 狀態機器、Amazon Simple Notification Service (Amazon SNS) 主題、Amazon Simple Queue Service (Amazon SQS) 佇列、內建目標等等。

如需有關建立 Eventbridge 的資訊，請參閱下列主題：

- 使用 Amazon EventBridge 監控 Systems Manager

- Systems Manager 的 Amazon EventBridge 事件範例
- 《Amazon EventBridge 使用者指南》中的 Amazon EventBridge 入門

本主題的其餘部分說明您可以包含在 EventBridge 規則中的 Systems Manager 事件類型。

事件類型：自動化

事件類型名稱	您可以新增至規則的事件描述
EC2 自動化執行狀態 - 變更通知	自動化工作流程的整體狀態發生變更。您可以將以下一個或多個狀態變更新增至事件規則： • Approved • 已取消 • 失敗 • PendingApproval • PendingChangeCalendarOverride • 已拒絕 • Scheduled (已排程) • 成功 • TimedOut
EC2 自動化步驟狀態 - 變更通知	自動化工作流程中特定步驟的狀態發生變更。您可以將以下一個或多個狀態變更新增至事件規則： • 已取消 • 失敗 • 成功 • TimedOut

事件類型：Change Calendar

事件類型名稱	您可以新增至規則的事件描述
行事曆狀態變更	Change Calendar 的狀態發生變更。您可以將以下一個或兩個狀態變更新增至事件規則： • OPEN • CLOSED 不支援從其他 AWS 帳戶 共享之行事曆的狀態變更。

事件類型：Change Manager

事件類型名稱	您可以新增至規則的事件描述
變更請求狀態更新	Change Manager 的狀態變更請求。您可在事件規則中使用下列狀態： • Approved • 已拒絕 • InProgress

事件類型：組態合規

事件類型名稱	您可以新增至規則的事件描述
組態合規狀態變更	受管節點的狀態會隨著關聯合規或修補程式合規而變更。您可以將以下一個或多個狀態變更新增至事件規則： • compliant • non_compliant

事件類型：庫存

事件類型名稱	您可以新增至規則的事件描述
庫存資源狀態變更	刪除自訂庫存和使用舊版結構描述的 PutInventory 呼叫。您可以將以下一個或多個狀態變更新增至事件規則： • 自訂庫存類型已刪除特定節點上的事件。EventBridge 會針對每個自訂 InventoryType 的每個節點傳送一個事件。 • 自訂庫存類型已刪除所有節點的事件。 • 使用舊版結構描述事件進行 PutInventory 呼叫。當結構描述版本小於目前的結構描述時，EventBridge 會傳送此事件。此事件適用於所有庫存類型。 如需詳細資訊，請參閱使用 EventBridge 監控庫存事件。

事件類型：維護時段

事件類型名稱	您可以新增至規則的事件描述
維護時段狀態-變更通知	一個或多個維護時段的整體狀態發生變更。您可以將以下一個或多個狀態變更新增至事件規則： • DISABLED • ENABLED
維護時段目標註冊通知	一個或多個維護時段目標的狀態發生變更。您可以將以下一個或多個狀態變更新增至事件規則： • DEREGISTERED • REGISTERED

事件類型名稱	您可以新增至規則的事件描述
	• UPDATED (已更新)
維護時段執行狀態 - 變更通知	維護時段執行時的整體狀態發生變更。您可以將以下一個或多個狀態變更新增至事件規則： • CANCELLED (已取消) • CANCEL • 失敗 • IN_PROGRESS • 待定 • SKIPPED_OVERLAPPING • 成功 • TIMED_OUT
維護時段任務執行狀態 - 變更通知	維護時段中的任務執行時的狀態發生變更。您可以將以下一個或多個狀態變更新增至事件規則： • CANCELLED (已取消) • CANCEL • 失敗 • IN_PROGRESS • 成功 • TIMED_OUT

事件類型名稱	您可以新增至規則的事件描述
維護時段任務目標叫用狀態-變更通知	特定目標上的維護時段任務的狀態發生變更。 此通知僅完全支援 Run Command 任務。對於這種類型的任務，您可以將以下一個或多個狀態變更新增至事件規則： • CANCELLED (已取消) • CANCEL • 失敗 • IN_PROGRESS • 成功 • TIMED_OUT 對於自動化 AWS Lambda和 AWS Step Functions 任務，EventBridge 只會報告狀態 `IN_PROGRESS` 和 `COMPLETE`。無論任務是否成功，`COMPLETE`都會報告 。
維護時段任務註冊通知	一個或多個維護時段任務的狀態發生變更。您可以將以下一個或多個狀態變更新增至事件規則： • DEREGISTERED • REGISTERED • UPDATED (已更新)

事件類型：OpsCenter

事件類型名稱	您可以新增至規則的事件描述
OpsItem 建立	建立 OpsItem 時發生。您可以為以下 OpsItem 類型之一新增規則： • /aws/issue

事件類型名稱	您可以新增至規則的事件描述
	• /aws/task • /aws/insight • /aws/actionitem
OpsItem 更新	更新 OpsItem 時發生。您可以為以下 OpsItem 類型之一新增規則： • /aws/issue • /aws/task • /aws/insight • /aws/actionitem

事件類型：Parameter Store

事件類型名稱	您可以新增至規則的事件描述
參數存放區變更	參數的狀態發生變更。您可以將以下一個或多個狀態變更新增至事件規則： • 建立 • 更新 • Delete • LabelParameterVersion 如需詳細資訊，請參閱為參數及參數政策設定 EventBridge。
參數存放區政策動作	符合進階參數政策變更的條件。您可以將以下一個或多個狀態變更新增至事件規則： • 過期 • ExpirationNotification • NoChangeNotification

事件類型名稱	您可以新增至規則的事件描述
	如需詳細資訊，請參閱為參數及參數政策設定 EventBridge。

事件類型：Run Command

事件類型名稱	您可以新增至規則的事件描述
EC2 命令叫用狀態-變更通知	傳送至個別受管執行個體的命令狀態發生變更。您可以將以下一個或多個狀態變更新增至事件規則： • 成功 • InProgress • TimedOut • 已取消 • 失敗
EC2 命令狀態-變更通知	命令的整體狀態發生變更。您可以將以下一個或多個狀態變更新增至事件規則： • 成功 • InProgress • TimedOut • 已取消 • 失敗

事件類型：State Manager

事件類型名稱	您可以新增至規則的事件描述
EC2 State Manager 關聯狀態變更	關聯的整體狀態會隨著其套用而發生變更。您可以將以下一個或多個狀態變更新增至事件規則：

事件類型名稱	您可以新增至規則的事件描述
	• 失敗 • 待定 • 成功
EC2 State Manager 執行個體關聯狀態變更	關聯所針對之狀態的單一受管執行個體發生變更。您可以將以下一個或多個狀態變更新增至事件規則： • 失敗 • 待定 • 成功

參考：Systems Manager 的 Cron 和 Rate 運算式

當您在 中建立State Manager關聯或維護時段時 AWS Systems Manager，您可以指定時段或關聯應執行的排程。您可以將排程指定為時間類型的項目 (稱為 cron 運算式)，或頻率類型的項目 (稱為 rate 運算式)。

有關 Cron 和 Rate 運算式的一般資訊

下列資訊適用於維護時段與關聯的 Cron 和 Rate 運算式。

單一執行排程

當您建立關聯或維護時段時，可以使用國際標準時間 (UTC) 格式指定時間戳記，以便在指定的時間執行一次。例如："at(2020-07-07T15:55:00)"

排程位移

關聯與維護時段僅支援 Cron 運算式的排程偏移。排程偏移是在執行關聯和維護時段之前，在 cron 運算式所指定的日期和時間之後等待的天數。

Maintenance window example

在以下命令中，Cron 表達式會排定維護時段，在每個月的第三個星期二晚上 11:30 執行：但是，由於排程偏移為 2，因而維護時段會等到兩天後在晚上 11:30 執行。

```
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

在下列命令中，Cron 表達式會排定關聯，以在每個月的第二個星期四執行。但是，由於排程偏移是 3，因此在三天後的下個星期日之前，才會執行關聯。

```
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**
>
> 若要使用具有關聯的偏移，您必須指定 `--apply-only-at-cron-interval` 選項。此選項會告訴系統不要在建立關聯後立即執行。

如果您使用 cron 運算式來建立維護時段，且其目標日期是在目前期間中已經過去的日期，但新增的排程偏移落在未來日期，則關聯或維護時段將不會在該期間內執行。它將在其次期間生效。例如，如果您指定的 Cron 運算式是要在昨天執行維護時段，並新增兩天的排程偏移，則維護時段將不會在明天執行。

必要欄位

維護時段的 Cron 運算式有六個必要欄位。關聯的 Cron 運算式有五個。(State Manager 目前不支援在 Cron 運算式中為關聯指定月份。) 其他欄位、Seconds 欄位 (cron 運算式中的第一個) 為選用。欄位以空格隔開。

Cron 運算式範例

分鐘	小時	月中的日	月	週中的日	年	意義
0	10	*	*	?	*	在每天上午 10:00

分鐘	小時	月中的日	月	週中的日	年	意義
						(UTC) 執行
15	12	*	*	?	*	在每天下午 12:15 (UTC) 執行
0	18	?	*	MON-FRI	*	在每週一至週五下午 6:00 (UTC) 執行
0	8	1	*	?	*	在每個月第 1 天上午 8:00 (UTC) 執行

支援的值

下表顯示支援的必要 cron 項目的值。

Cron 運算式支援的值

欄位	值	萬用字元
分鐘	0-59	, - * /
小時	0-23	, - * /
月中的日	1-31	, - * ? / L W
月份 (僅限維護時段)	1-12 或 JAN-DEC	, - * /
週中的日	1-7 或 SUN-SAT	, - * ? / L #

欄位	值	萬用字元
年	1970-2199	, - * /

> **Note**
>
> 您無法在同一個 Cron 運算式中指定 day-of-month (月中的日) 和 day-of-week (週中的日) 欄位中的值。如果您在其中一個欄位指定了數值，就請在另一個欄位中使用 ? (問號)。

Cron 運算式的萬用字元

下表顯示 cron 運算式支援的萬用字元值。

> **Note**
>
> 不支援頻率快於 5 分鐘的 Cron 運算式。尚未完成支援同時指定週間日與月中日的值。在其中一個欄位使用問號 (?) 字元。

Cron 運算式支援的萬用字元

萬用字元	描述
,	, (逗號) 萬用字元包含額外的值。在 Month (月) 欄位，JAN、FEB、MAR 包括 January (一月)、February (二月) 與 March (三月)。
-	- (破折號) 萬用字元用於指定範圍。在 Day (日) 欄位，1-15 包含指定月份的 1 至 15 號。
*	* (星號) 包含欄位中所有的值。在 Hours (小時) 欄位中，* 包含每個小時。
/	/ (斜線) 萬用字元用於指定增量。在分鐘欄位，您可以輸入 1/10 指定每 10 分鐘的間隔，從小時的第一分鐘開始。因此，1/10 指定第 1、11、21 和 31 分鐘，以此類推。

萬用字元	描述
?	? (問號) 萬用字元用於表示不限定任何一個。在 Day-of-month (月中的日) 欄位，您可以輸入 7，如果您不在意這個月的 7 號是星期幾，就可以在 Day-of-month (月中的日) 欄位中輸入 ?。
L	L 萬用字元在 Day-of-month (月中的日) 或 Day-of-week (週中的日) 欄位可指定月份或週的最後一天。
W	W 萬用字元在 Day-of-month (月中的日) 欄位可指定工作日。在 Day-of-month (月中的日) 欄位，3W 指定的是月份中最接近第三個工作日的日子。
#	後接數字一至五之星期幾欄位中的 # 萬用字元，會指定月份的特定日期。5#3 指定當前月份的第三個星期四。

Rate 運算式

Rate 運算式有以下兩個必要欄位。欄位以空格隔開。

Rate 運算式的必要欄位

欄位	值
Value	正數，例如 1 或 15

單位	`minute`
	
	`minutes`
	
	`hour`
	
	`hours`
	
	`day`

欄位	值
	days

如果值等於 1，則單位必須為單數。同樣地，對於大於 1 的數值，單位必須為複數。例如，`rate(1 hours)` 與 `rate(5 hour)` 不是有效的，但 `rate(1 hour)` 與 `rate(5 hours)` 是有效的。

主題

- 關聯的 Cron 與 Rate 運算式
- 維護時段的 Cron 與 Rate 運算式

關聯的 Cron 與 Rate 運算式

此部分包括 State Manager 關聯的 cron 與 rate 運算式範例。在建立其中一個這些運算式之前，請注意以下資訊：

- 關聯支援以下 Cron 表達式：每 1/2、1、2、4、8 或 12 小時；每天、每週或每週的每個指定日期和時間；該月特定週的特定一天，或該月最後 x 天的特定時間。
- 關聯支援以下 Rate 運算式：間隔為 30 分鐘或者大於 30 分鐘並小於 31 天。
- 如果您指定選用的 `Seconds` 欄位，它的值只能為 0 (零)。例如：`cron(0 */30 * * * ? *)`。
- 對於收集庫存中繼資料的關聯，在 中的工具 AWS Systems Manager，我們建議您使用速率表達式。
- State Manager 目前不支援在 Cron 運算式中為關聯指定月份。

關聯支援 cron 運算式，其中包括一週中的某一天和數字符號 (#)，以指定一個月的第 n 天執行關聯。以下是在每月第三個週二 23:30 UTC 執行 cron 排程的範例：

```
cron(30 23 ? * TUE#3 *)
```

以下是在每月第二個週四午夜 UTC 執行的範例：

```
cron(0 0 ? * THU#2 *)
```

關聯還支援 (L) 符號來指示一個月的最後 X 天。以下是在每月最後一個週二午夜 UTC 執行 cron 排程的範例：

`cron(0 0 ? * 3L *)`

若要進一步控制關聯的執行時間，例如，如果您希望在週二修補程式日後的兩天執行關聯，則可以指定偏移量。同時偏移定義在排程的日期之後等待多少天才能執行關聯。例如，如果您指定了 `cron(0 0 ? * THU#2 *)` 的 cron 排程，則可以在排程偏移欄位指定數字 3，以在該月第二個週四之後的每個週日執行關聯。

若要使用偏移，必須在主控台選擇 Apply association only at the next specified Cron interval (僅在下一個指定 Cron 間隔時間套用關聯) 選項，或者必須在命令列中指定使用 `--apply-only-at-cron-interval` 參數。此選項會告訴 State Manager 不要在建立關聯後立即執行。

下表顯示關聯的 Cron 範例。

關聯的 Cron 範例

範例	詳細資訊
cron(0/30 * * * ? *)	每 30 分鐘
cron(0 0/1 * * ? *)	每小時
cron(0 0/2 * * ? *)	每 2 小時
cron(0 0/4 * * ? *)	每 4 小時
cron(0 0/8 * * ? *)	每 8 小時
cron(0 0/12 * * ? *)	每 12 小時
cron(15 13 ? * * *)	每天下午 1:15
cron(15 13 ? * MON *)	每個週一下午 1:15
cron(30 23 ? * TUE#3 *)	每月第三個週二下午 11:30

以下是一些關聯的 rate 範例。

關聯的 Rate 範例

範例	詳細資訊
rate(30 minutes)	每 30 分鐘

範例	詳細資訊
rate(1 hour)	每小時
rate(5 hours)	每 5 小時
rate(15 days)	每 15 天

AWS CLI 關聯的範例

若要使用 建立State Manager關聯 AWS CLI，請使用 Cron 或 Rate 表達式來包含 `--schedule-expression` 參數。下列範例會在本機 Linux AWS CLI 電腦上使用 。

> **Note**
>
> 根據預設，當您建立新的關聯時，系統會在建立該關聯後立即執行，然後根據您指定的排程。指定 `--apply-only-at-cron-interval`，以便不會在您建立關聯之後立即執行。此參數不支援 Rate 運算式。

```
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
```

維護時段的 Cron 與 Rate 運算式

本節包含適用於維護時段的 Cron 和 Rate 運算式。

與 State Manager 關聯不同，維護時段支援所有 cron 和 rate 運算式。這包括在秒欄位中支援數值。

例如，以下 6 個欄位的 Cron 運算式會在每天上午 9:30 執行維護時段。

```
cron(30 09 ? * * *)
```

透過將值新增至 Seconds 欄位，以下 7 個欄位的 Cron 運算式會在每天上午 9:30:24 執行維護時段。

```
cron(24 30 09 ? * * *)
```

下表提供適用於維護時段的額外 6 個欄位 Cron 範例。

維護時段的 Cron 範例

範例	詳細資訊
cron(0 2 ? * THU#3 *)	每個月第三個週四的上午 02:00
cron(15 10 ? * * *)	每天的上午 10:15
cron(15 10 ? * MON-FRI *)	每個週一、週二、週三、週四和週五的上午 10:15
cron(0 2 L * ? *)	每個月最後一天的上午 02:00
cron(15 10 ? * 6L *)	每個月最後一個星期五的上午 10:15

下表提供適用於維護時段的 Rate 範例。

維護時段的 Rate 範例

範例	詳細資訊
rate(30 minutes)	每 30 分鐘
rate(1 hour)	每小時

範例	詳細資訊
rate(5 hours)	每 5 小時
rate(25 days)	每 25 天

AWS CLI 維護時段的範例

若要使用 建立維護時段 AWS CLI，請使用 Cron 或速率表達式或時間戳記來包含 `--schedule` 參數。下列範例會在本機 Linux AWS CLI 電腦上使用 。

```
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
```

詳細資訊

在 Wikipedia 網站的 Cron 運算式

參考：ec2messages、ssmmessages 和其他 API 操作

如果您監控 API 操作，則可能會看到對下列操作的呼叫：

- 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

這些是 使用的特殊操作 AWS Systems Manager，如本主題的其餘部分所述。

代理程式相關的 API 操作 (**ssmmessages** 和 **ec2messages** 端點)

ssmmessages API 操作

Systems Manager 會將 ssmmessages 端點用於下列 API 操作類型：

- 從 Systems Manager Agent (SSM Agent) 至雲端中的 Systems Manager 服務的操作。
- 在雲端中從 SSM Agent 到 的操作 AWS Systems Manager，Session Manager這是 中的工具。必須使用這個端點建立和刪除連往雲端 Session Manager 服務的工作階段管道。此外，如果允許連線，則 SSM Agent 會透過此 Amazon Message Gateway Service 接收 Command 文件。如果不允許連線，則 SSM Agent 會透過 Amazon Message Delivery Service 接收 Command 文件。如需詳細資訊，請參閱適用於 Amazon Message Gateway Service 的動作、資源和條件鍵。
- 來自 Run Command 的操作。

ec2messages API 操作

ec2messages:* API 操作會對 Amazon Message Delivery Service 端點執行。Systems Manager 使用此端點用於從 Systems Manager Agent (SSM Agent) 至雲端中的 Systems Manager 服務的 API 操作。

> **Important**
>
> 只有在 2024 年之前推出的 AWS 區域 中才支援 ec2messages:* API 操作。在 2024 年及以後推出的區域中僅支援 ssmmessages:* API 操作。

端點連線優先順序

從 SSM Agent 3.3.40.0 版開始，Systems Manager 會在可用時開始使用 ssmmessages:* 端點 (Amazon Message Gateway Service)，而不是 ec2messages:* 端點 (Amazon Message Delivery Service)。

如果您在 AWS Identity and Access Management (IAM) 許可政策ssmmessages:*中提供對 的存取權， SSM Agent會連線至ssmmessages:*端點，即使您的 IAM 執行個體設定檔設定為允許兩個端點。這包括您自行建立的 IAM 執行個體設定檔和 IAM 服務角色，以及Quick Setup主機管理組態和預設主機管理組態建立的 IAM 執行個體設定檔的政策。

如果您已提供兩個端點的許可，且使用 CloudWatch Metrics 監控 API 操作，則不會看到對 ec2messages:* 的呼叫。

對於 2024 年之前 AWS 區域 啟動的 ：此時，您可以安全地從政策中移除ec2messages:*許可。

端點連線容錯移轉

僅適用於 2024 年之前 AWS 區域 啟動的 ：如果您的 IAM 執行個體描述檔ssmmessages:*在代理程式啟動時未提供 的許可，但只有 ec2messages:*， SSM Agent會連接至ec2messages:*端點。如果您在 SSM Agent 啟動時擁有 ssmmessages:* 和 ec2messages:* 兩者，但在代理程式啟動後移除 ssmmessages:*，則 SSM Agent 很快就會將連線切換到 ec2messages:* 端點。對於 2024 年及以後推出的區域，僅支援 ssmmessages:* 端點。

如需有關 ssmmessages 和 ec2messages:* 端點的詳細資訊，請參閱《AWS 服務授權參考》中的下列主題：

- Actions, resources, and condition keys for Amazon Message Gateway Service (ssmmessages)
- Actions, resources, and condition keys for Amazon Message Delivery Service (ec2messages:*)

ssm:* 命名空間執行個體相關的 API 操作

DescribeDocumentParameters

Systems Manager 會執行此 API 操作，在 Amazon EC2 主控台中轉譯特定的節點。DescribeDocumentParameters 操作的結果會顯示在文件節點中。

DescribeInstanceProperties

Systems Manager 會執行這些 API 操作，在 Amazon EC2 主控台中轉譯特定的節點。DescribeInstanceProperties 操作的結果會顯示在 Fleet Manager 節點中。

GetCalendar

Systems Manager 會執行此 API 操作，在 Change Calendar 主控台中轉譯 Change Calendar 類型文件。

GetManifest

SSM Agent 會執行此 API 操作，判斷安裝或更新 AWS Systems Manager Distributor 套件指定版本的系統要求。這是舊版 API 操作，在 2017 年之後 AWS 區域 啟動時無法使用。

ListInstanceAssociations

SSM Agent 會執行此 API 操作，確認是否有新的 State Manager 關聯可用。State Manager 需要有這個 API 操作才能運作。

PutCalendar

Systems Manager 會執行此 API 操作，在 Change Calendar 主控台中更新 Change Calendar 類型文件。

`PutConfigurePackageResult`

SSM Agent 會執行此 API 操作，將 Distributor 公開套件的安裝錯誤和延遲指標發布到套件擁有者的帳戶。

`RegisterManagedInstance`

SSM Agent 針對下列案例執行此 API 操作：

- 使用啟用代碼和 ID，向 Systems Manager 註冊內部部署伺服器或虛擬機器 (VM) 作為受管執行個體。
- 註冊 AWS IoT Greengrass Version 2 登入資料。

執行 SSM Agent 3.1.x 版或更新版本的 Amazon EC2 執行個體也會呼叫此操作。

`RequestManagedInstanceRoleToken`

SSM Agent 會執行此 API 操作來擷取可存取受管節點的臨時憑證。

`UpdateInstanceAssociationStatus`

SSM Agent 會執行此 API 操作來更新關聯。中的State Manager工具 需要此 API 操作 AWS Systems Manager才能運作。

`UpdateInstanceInformation`

SSM Agent 會每 5 分鐘呼叫雲端中的 Systems Manager 服務，提供活動訊號資訊。必須要有此呼叫維持與代理程式的活動訊號，讓服務知道代理程式運作正常。

`UpdateManagedInstancePublicKey`

在受管節點上輪換金鑰對後，SSM Agent 會執行此 API 操作來提供公有金鑰。公有金鑰會用於驗證使用私有金鑰簽署的請求，以便從 Systems Manager 取得臨時憑證。

ssm:* 命名空間其他 API 操作

`ExecuteApi`

在 OpsCenter 中管理 OpsItems 的 Systems Manager 委派管理員需要存取此 API 動作，以便他們可以檢視多個 AWS 帳戶之間 OpsItems 的相關資源詳細資訊。具體而言，此 API 會提供委派管理員許可，以在 中檢視下列OpsItem詳細資訊 AWS Management Console：OpsItem描述、標籤、AWS CloudFormation 範本、 AWS Config 變更、CloudWatch Logs 警示和 AWS CloudTrail 事件。如需有關跨帳戶使用 OpsItems 的詳細資訊，請參閱 (選用) 手動設定 OpsCenter 以跨帳戶集中管理 OpsItems。如需有關 OpsItems 相關資源的詳細資訊，請參閱將相關的資源新增至 OpsItem。

參考：Systems Manager 的日期和時間字串格式

AWS Systems Manager API 操作接受篩選條件，以限制請求傳回的結果數量。其中一些 API 操作接受的篩選條件需要格式化字串以代表特定日期和時間。例如，DescribeSessions API 操作接受 InvokedAfter 和 InvokedBefore 金鑰做為 SessionFilter 物件的部分有效值。另一個範例是 DescribeAutomationExecutions API 操作，它接受 StartTimeBefore 和 StartTimeAfter 金鑰做為 AutomationExecutionFilter 物件的部分有效值。篩選請求時，您為這些索引鍵提供的數值必須符合 ISO 8601 標準。如需 ISO 8601 的詳細資訊，請參閱 ISO 8601。

這些格式化的日期和時間字串不限於篩選條件。在提供請求參數值時，還有一些 API 操作需要 ISO 8601 格式的字串來表示特定的日期和時間。例如，AtTime 操作的 GetCalendarState 請求參數。這些字串很難建立。使用本主題中的範例，來建立格式化的日期和時間字串，以搭配 Systems Manager API 操作使用。

為 Systems Manager 格式化日期和時間字串

以下是 ISO 8601 格式化的日期和時間字串的範例。

```
2024-05-08T15:16:43Z
```

這代表 2024 年 5 月 8 日國際標準時間 (UTC) 15:16。字串的行事曆日期部分以四位數的年份、二位數的月份和二位數的日期來表示，並以連字號分隔。這可以用以下格式表示。

```
YYYY-MM-DD
```

字串的時間部分以字母「T」開頭做為分隔符號，然後以冒號分隔的二位數小時、二位數分鐘，以及二位數秒表示。這可以用以下格式表示。

```
hh:mm:ss
```

字串的時間部分以字母「Z」結尾，表示 UTC 標準。

為 Systems Manager 建立自訂的日期和時間字串

您可以使用偏好的命令列工具從本機機器建立自訂日期和時間字串。您建立 ISO 8601 格式化的日期和時間字串所用的語法會因您本機電腦的作業系統而有所不同。以下是如何在 Linux 上使用 date GNU 的核心使用程序或在 Windows 上的 PowerShell 來建立 ISO 8601 格式的日期和時間字串的範例。

coreutils

```
date '+%Y-%m-%dT%H:%M:%SZ'
```

PowerShell

```
(Get-Date).ToString("yyyy-MM-ddTH:mm:ssZ")
```

使用 Systems Manager API 操作時，您可能需要建立歷史日期和時間字串以供報告或故障排除之用。以下是如何為 AWS Tools for PowerShell 和 AWS Command Line Interface () 建立和使用自訂歷史 ISO 8601 格式日期和時間字串的範例AWS CLI。

AWS CLI

- 擷取 SSM 文件最後一週的命令歷史記錄。

```
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
```

- 擷取最後一週的自動化執行歷史記錄。

```
lastWeekStamp=$(date '+%Y-%m-%dT%H:%M:%SZ' -d '7 days ago')

aws ssm describe-automation-executions \
    --filters Key=StartTimeAfter,Values=$lastWeekStamp
```

- 擷取最後一個月的工作階段歷史記錄。

```
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

- 擷取 SSM 文件最後一週的命令歷史記錄。

```
$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
```

- 擷取最後一週的自動化執行歷史記錄。

```
$lastWeekStamp = (Get-Date).AddDays(-7).ToString("yyyy-MM-ddTH:mm:ssZ")

Get-SSMAutomationExecutionList `
    -Filters @{Key="StartTimeAfter";Values=$lastWeekStamp}
```

- 擷取最後一個月的工作階段歷史記錄。

```
$lastWeekStamp = (Get-Date).AddDays(-30).ToString("yyyy-MM-ddTH:mm:ssZ")

Get-SSMSession `
    -State History `
    -Filters @{Key="InvokedAfter";Value=$lastWeekStamp}
```

使用案例與最佳實務

本主題列出 AWS Systems Manager 工具的常見使用案例和最佳實務。如果可用，這個主題也包含連結，可連結到相關的部落格文章和技術文件。

> **Note**
>
> 此處每個區段的標題，都是有效的連結，可連結到技術文件中的對應區段。

自動化

- 建立基礎設施用的自助服務自動化 Runbook。
- 使用 中的自動化工具 AWS Systems Manager，透過使用公有 Systems Manager 文件 Amazon Machine Images(SSM 文件AMIs) 或編寫您自己的工作流程AMIs，簡化從 AWS Marketplace 或自訂建立 ()。
- 使用 `AWS-UpdateLinuxAmi` 和 `AWS-UpdateWindowsAmi` 自動化 Runbook，或使用您建立的自訂自動化 Runbook，建置和維護 AMIs。

合規

- 作為安全最佳實務，建議您更新受管節點使用的 AWS Identity and Access Management (IAM) 角色，以限制節點使用 PutComplianceItems API 動作的能力。此 API 動作會在指定的資源上註冊合規類型和其他合規詳細資訊，例如 Amazon EC2 執行個體或受管節點。如需詳細資訊，請參閱設定合規許可。

庫存

- 搭配 使用 中的工具 Inventory AWS Systems Manager，以稽核您的 AWS Config 應用程式組態。

Maintenance Windows

- 定義排程，在您的節點上執行可能中斷的動作，例如作業系統 (OS) 修補、驅動程式更新，或軟體安裝。
- 如需 State Manager和 工具之間差異的資訊Maintenance Windows， AWS Systems Manager請參閱 在 State Manager 與 Maintenance Windows 之間進行選擇。

Parameter Store

- 使用 中的Parameter Store工具 AWS Systems Manager，集中管理全域組態設定。
- AWS Systems ManagerParameter Store 如何使用 AWS KMS 。
- 參考Parameter Store參數中的 AWS Secrets Manager 秘密。

Patch Manager

- 使用 Patch Manager中的工具 AWS Systems Manager，大規模推出修補程式，並提高節點的機群合規可見性。
- 整合 Patch Manager 與 AWS Security Hub，以便在機群中的節點不合規時接收提醒，並從安全角度監控您的機群的修補狀態。使用 Security Hub 會產生費用。如需詳細資訊，請參閱 定價。
- 一次只能使用一種方法來掃描受管節點，檢查修補程式的合規性，以避免意外覆寫合規資料。

Run Command

- 使用 EC2 Run 命令來進行大規模的執行個體管理，而不需存取 SSH。
- AWS Systems Manager使用 稽核 中Run Command工具 所發出或代表發出的所有 API 呼叫 AWS CloudTrail。
- 當您使用 Run Command 發出命令時，請勿包含格式為純文字的敏感資訊，例如密碼、組態資料或其他密碼。您帳戶中的所有 Systems Manager API 活動都會記錄於 S3 儲存貯體中的 AWS CloudTrail 日誌。這意味著任何具有權存取該 S3 儲存貯體的使用者都可以查看這些密碼的純文字值。因此，建議您建立並使用 `SecureString` 參數來加密您在 Systems Manager 操作中使用的敏感資料。

  如需詳細資訊，請參閱使用 IAM 政策限制對 Parameter Store 參數的存取。

> **Note**
>
> 根據預設，CloudTrail 交付至您儲存貯體的日誌檔案是透過 Amazon 伺服器端加密與 Amazon Simple Storage Service (Amazon S3) 受管加密金鑰 (SSE-S3) 進行加密。若要提供可直接管理的安全層，您可以改為將伺服器端加密與 AWS KMS受管金鑰 (SSE-KMS) 用於 CloudTrail 日誌檔案。
> 如需詳細資訊，請參閱AWS CloudTrail 《 使用者指南》中的使用 AWS KMS受管金鑰 (SSE-KMS) 加密 CloudTrail 日誌檔案。

- 使用 Run Command 中的目標和速率控制功能，來進行階段式的指令操作。
- 透過使用 Run Command(IAM) 政策，使用 AWS Identity and Access Management （ 和所有 Systems Manager 工具 ） 的精細存取許可。

Session Manager

- 使用 AWS CloudTrail在 AWS 帳戶 中記錄工作階段活動。
- 使用 Amazon CloudWatch Logs 或 Amazon Simple Storage Service (Amazon S3) 在您的 AWS 帳戶 中記錄工作階段資料。
- 控制使用者工作階段存取執行個體。
- 限制對工作階段中命令的存取。
- 停用或啟用 ssm-user 帳戶管理許可。

State Manager

- 使用預先設定的 `AWS-UpdateSSMAgent` 文件，來至少一個月更新一次 SSM Agent。
- (Windows) 將 PowerShell 或 DSC 模組上傳到 Amazon Simple Storage Service (Amazon S3)，並使用 `AWS-InstallPowerShellModule`。
- 使用標籤來建立您節點的應用程式群組。然後使用 `Targets` 參數，而不是指定個別的節點 ID，來鎖定節點。
- 使用 Systems Manager，來自動修正 Amazon Inspector 所產生的結果。
- 針對您的 SSM 文件，使用集中化的組態儲存庫，然後在整個組織之間共用文件。
- 如需有關 State Manager 和 Maintenance Windows 之間的差異的資訊，請參閱 在 State Manager 與 Maintenance Windows 之間進行選擇。

受管節點

- Systems Manager 需要準確的時間參考才能執行其操作。如果您節點的日期和時間未正確設定，可能會與您 API 請求的簽章日期不符。這可能會導致錯誤或不完整的功能。例如，在您的受管節點清單中，將不會包含具有錯誤時間設定的節點。

  如需有關在節點設定時間的資訊，請參閱設定 Amazon EC2 執行個體的時間。
- 在 Linux 受管節點上，驗證 SSM Agent 的簽章。

詳細資訊

- Systems Manager 的安全最佳實務

刪除 Systems Manager 資源和成品

最佳實務是，如果您不再需要檢視這些資源的相關資料或以任何方式使用成品，建議您刪除 Systems Manager 資源和成品。下表列出每個 Systems Manager 工具或成品，以及有關刪除 Systems Manager 所建立資源或成品的詳細資訊連結。

功能或成品	詳細資訊
Application Manager	您無法刪除 Application Manager 中的應用城市程式，但您可以刪除基礎的 tags (標籤)、Resource Groups 或 AWS CloudFormation stacks (堆疊)，進而從服務中移除應用程式。
自動化	如果您使用 Systems Manager Automation 建立 AWS 資源，則必須使用對應的 手動刪除這些資源 AWS Management Console。如果您已建立自訂 Runbook，您可以刪除底層 SSM 文件。如需詳細資訊，請參閱刪除自訂 SSM 文件。
Change Calendar	您可以刪除變更行事曆和變更行事曆事件。如需詳細資訊，請參閱 刪除變更行事曆 及 刪除 Change Calendar事件。
Change Manager	您可以刪除變更範本。如需詳細資訊，請參閱刪除變更範本。
合規	Systems Manager 合規會自動顯示有關 Patch Manager 修補程式與 State Manager 關聯的合規資料。您無法刪除此資料。如果您已設定資源資料同步來集中 S3 儲存貯體中的合規資料，您可以刪除同步。如需詳細資訊，請參閱刪除合規的資源資料同步。

功能或成品	詳細資訊
Distributor	您可以刪除 Distributor 中的套件。如需詳細資訊，請參閱刪除 Distributor 套件。
Explorer	您可以斷開連結 Explorer 從中收集 OpsData 的資料來源。如需詳細資訊，請參閱編輯 Systems Manager Explorer 資料來源。 您也可以刪除 用來Explorer將 OpsData 和 OpsItems 從多個 AWS 區域 和 帳戶彙總到單一 Amazon Simple Storage Service (Amazon S3) 儲存貯體的資源資料同步。如需詳細資訊，請參閱刪除 Systems Manager Explorer 資源資料同步。如需有關刪除 S3 儲存貯體的資訊，請參閱《Amazon Simple Email Service 開發人員指南》中的刪除儲存貯體。
Fleet Manager	您無法使用 Fleet Manager 刪除受管節點。您必須使用 Amazon Elastic Compute Cloud (Amazon EC2)。如需詳細資訊，請參閱終止您的執行個體 (Linux) 和終止您的執行個體 (Windows)。
庫存	您可以藉由刪除定義排程的 State Manager 關聯和要從中收集中繼資料的資源，停止庫存資料收集。如需詳細資訊，請參閱停用資料收集和刪除庫存資料。 如果您不想再使用 AWS Systems Manager 庫存來檢視 AWS 資源的中繼資料，我們也建議您刪除用於庫存資料收集的資源資料同步。如需詳細資訊，請參閱刪除庫存資源資料同步。
Maintenance Windows	您可以刪除維護時段、維護時段目標和維護時段任務。如需詳細資訊，請參閱使用主控台更新或刪除維護時段資源。

功能或成品	詳細資訊
OpsCenter	您可以使用 AWS Command Line Interface 或 AWS SDK OpsItem 呼叫刪除OpsItem API 操作來刪除個人。您無法刪除 AWS Management Console中的 OpsItem。如需詳細資訊，請參閱刪除 OpsItems。
Parameter Store	您可以刪除您已建立的參數。如需詳細資訊，請參閱從 Parameter Store 中刪除參數。
Patch Manager	您可以刪除自訂修補基準。如需詳細資訊，請參閱更新或刪除自訂修補基準。
快速設定	您可以刪除由「快速設定」建立的關聯。這些關聯會由 State Manager 存放和處理。如需詳細資訊，請參閱刪除關聯。
Run Command	完成命令處理之後，其相關資訊會存放在 Command history (命令歷史記錄) 標籤上。您無法從 Command history (命令歷史記錄) 標籤刪除資訊。
自動化	自動化完成處理後，相關資訊會存放在執行索引標籤中。您無法從執行索引標籤刪除資訊。
服務連結角色	Systems Manager 會自動為某些工具建立服務連結角色。您可以刪除這些角色。如需詳細資訊，請參閱刪除 Systems Manager 的 AWSServiceRoleForAmazonSSM 服務連結角色。
Session Manager	Session Manager 在終止工作階段後不會保留資源的相關資料。若要終止工作階段，請參閱 結束工作階段。

功能或成品	詳細資訊
SSM Agent	您可以手動從節點解除安裝 SSM Agent。如需詳細資訊，請參閱下列主題。 • Linux：在 Linux EC2 執行個體上手動安裝和解除安裝 SSM Agent • macOS: 在 macOS EC2 執行個體上手動安裝和解除安裝 SSM Agent • Windows Server：開啟 Control panel (控制面板) 然後選擇 Add/remove programs (新增/移除計劃)。
State Manager	您可以刪除關聯。如需詳細資訊，請參閱刪除關聯。
Systems Manager 文件服務	您無法刪除 AWS 或 提供的 Runbook AWS 支援，但可以刪除自訂 Runbook。如需詳細資訊，請參閱刪除自訂 SSM 文件。

在 State Manager 與 Maintenance Windows 之間進行選擇

State Manager 中的 和 Maintenance Windows，這兩種工具 AWS Systems Manager都可以在受管節點上執行一些類似的更新類型。您選擇哪一項，取決於您是否需要在指定的期間內自動化系統合規，或執行高優先順序、時間敏感的任務。

State Manager 和 Maintenance Windows：關鍵使用案例

State Manager是 中的工具 AWS Systems Manager，可設定和維護 中受管節點 AWS 和資源的目標狀態組態 AWS 帳戶。您可以將組態和目標的組合定義為關聯物件。如果您想要將帳戶中的所有受管節點維持在一致狀態、使用 Amazon EC2 Auto Scaling 產生新節點，或對帳戶中的受管節點具有嚴格的合規報告要求，State Manager則建議使用 工具。

主要使用案例 State Manager 如下所示：

- Auto Scaling 案例：State Manager 可以手動或透過 Auto Scaling 群組監控帳戶內啟動的所有新節點。如果帳戶中有任何針對該新節點的關聯 (透過標籤或所有節點)，則該特定關聯會自動套用至新節點。

- 合規報告：State Manager 可以為帳戶中的資源產生所需狀態的合規報告。
- 支援所有節點：State Manager 可以鎖定指定帳戶內的所有節點。

維護時段會在指定的時段內執行一或多個 AWS 資源。您可以定義包含開始和結束時間的單一維護時段。您可以指定要在此維護時段內執行的多個任務。如果您的高優先順序操作包括修補受管節點 AWS Systems Manager、在更新期間在節點上執行多種類型的任務，或控制何時可以在節點上執行更新操作，請使用 中的Maintenance Windows工具。

主要使用案例 Maintenance Windows 如下所示：

- 執行多個文件：維護時段可以執行多個工作。每個任務都可以使用不同的文件類型。因此，您可以在單一維護時段內使用不同的任務來建置複雜的工作流程。
- 修補：維護時段可以針對單一區域中標記特定標籤或在特定資源群組中的所有受管節點提供修補支援。由於修補通常涉及停止節點 (例如，從負載平衡器移除節點)、修補和後續處理 (將節點放回生產環境)，因此其可以在指定的修補程式時段內做為一系列任務來完成修補。

> **Note**
>
> 使用維護時段時，修補作業將侷限於單一帳戶的單一區域。使用 Systems Manager 中工具中建立Quick Setup的修補程式政策，您可以改為為在 中建立的組織中的部分或所有帳戶和區域設定修補 AWS Organizations。如需詳細資訊，請參閱Quick Setup中的修補程式政策組態。

- 時段動作：維護時段可讓一或多組動作在特定時段內啟動。維護時段不會在該時段之外啟動。已啟動的動作會繼續進行直到完成為止，即使它們在該時段外完成也是如此。

下表會比較 State Manager 和 Maintenance Windows 的主要功能。

功能	State Manager	Maintenance Windows
AWS CloudFormation 整合	AWS CloudFormation 範本支援State Manager關聯。	AWS CloudFormation 範本支援維護時段、時段目標和時段任務。
合規	每個 State Manager 關聯都會根據目標資源所需狀態報告合	不適用。

功能	State Manager	Maintenance Windows
	規性。您可以使用合規儀表板來彙總和檢視報告的合規。	
組態管理整合	State Manager支援外部目標狀態解決方案，例如 Microsoft PowerShell Desired State Configuration (DSC)、Ansible 手冊和 Chef 配方。您可以使用 State Manager 關聯來測試組態管理解決方案是否正常運作，並在您準備就緒時將其組態變更套用至您的節點。	不適用。
文件	State Manager 組態可以定義為政策文件 (用於收集庫存資訊)、適用於 AWS 的 Automation Runbook，例如 Amazon Simple Storage Service (Amazon S3) 儲存貯體，或受管節點的 Systems Manager 命令文件 (SSM 文件)。	Maintenance Windows 組態可以定義為自動化文件 (具有選用核准工作流程的多步驟動作) 或 SSM 文件 (受管節點的所需狀態)。
監控	State Manager 會監控節點組態、關聯或狀態的變更 (例如，即將上線的新節點)。當 State Manager 偵測到這些變更時，指定的關聯會重新套用至最初以該關聯為目標的節點。	不適用。

功能	State Manager	Maintenance Windows
任務中的優先順序	不適用。	維護時段內的任務可以獲指派優先順序。具有相同優先順序的所有任務都會平行執行。優先順序較低的工作會在具有較高優先順序的工作達到最終狀態之後執行。無法有條件地執行任務。優先順序較高的任務達到其最終狀態後，不論先前工作的狀態為何，都會執行下一個優先順序工作。
安全控制	在跨大型機群部署組態時，State Manager 支援兩種安全控制。您可以使用最大並行性來定義應套用組態的並行節點或資源數目。如果整個機群發生特定數量或百分比的錯誤，您可以定義一個可用於暫停 State Manager 關聯的最大錯誤率。	在跨大型機群部署組態時，維護時段支援兩種安全控制。您可以使用最大並行性來定義應套用組態的並行節點或資源數目。如果整個機群發生特定數量或百分比的錯誤，您可以定義一個可用於暫停維護時段內的動作的最大錯誤率。

功能	State Manager	Maintenance Windows
排程	您可以隨需執行 State Manager 關聯，以特定的 Cron 間隔，以給定的速率，或者在建立後。如果您想要以一致且及時的方式維持所需的資源狀態，該方法非常有用。 **Important** State Manager 關聯的 Cron 運算式不支援月份欄位，如三月份的 03 或 MAR。如果您需要每月或每季度更新組態，則維護時段最符合您的需求。如需詳細資訊，請參閱參考：Systems Manager 的 Cron 和 Rate 運算式。	維護時段支援多種排程選項，包括 at 運算式 (例如 "at(2021-07-07T13:15:30)")、cron 和 rate 運算式、帶偏移量的 cron、應執行維護時段的開始和結束時間，以及指定在給定時段內何時停止排程的截止時間。
目標鎖定	State Manager 關聯可以使用節點 ID、標籤或資源群組鎖定一個或多個節點。State Manager 可以鎖定指定帳戶內的所有受管節點。	維護時段可以使用節點 ID、標籤或資源群組來鎖定一個或多個節點。

功能	State Manager	Maintenance Windows
維護時段內的任務	不適用。	維護時段可以支援一個或多個任務，其中每個任務以特定自動化 Runbook 或命令文件動作為目標。除非為不同的任務設定了不同的優先順序，否則維護時段內的所有任務都會平行執行。 整體而言，維護時段支援執行四種任務類型： • AWS Systems Manager Run Command 命令 • AWS Systems Manager 自動化工作流程 • AWS Lambda 函數 • AWS Step Functions 任務

相關資訊

以下相關資源可協助您使用此服務。

定價

有些 Systems Manager 工具會收取費用。如需詳細資訊，請參閱 AWS Systems Manager 定價。

AWS Systems Manager 文件程式庫

AWS Systems Manager 文件 – 存取 Systems Manager 的所有使用者文件，包括 AWS AppConfig Incident Manager 和 AWS Systems Manager SAP。

AWS re:Post

AWS re:Post – AWS 受管問題和答案 (Q & A) 服務，為您的技術問題提供群眾來源、專家審查的答案。

AWS 部落格和播客

閱讀有關 AWS 管理工具類別中 Systems Manager 的部落格文章，以及含有 #Systems Manager 標籤的其他文章。

Service Quotas

檢閱《Amazon Web Services 一般參考》中的 Systems Manager 服務配額一節。除非另有說明，否則每個配額套用於一個 AWS 帳戶中的一個區域。

Systems Manager 的服務授權參考

在AWS 服務授權參考中，檢視您可以在 Systems Manager 的 AWS Identity and Access Management (IAM) 政策中使用的動作、資源和條件內容金鑰的相關資訊。

AWS Systems Manager 服務層級協議

AWS Systems Manager 服務層級協議 (SLA) 是管理 Systems Manager 使用的政策，並使用 AWS 帳戶 Systems Manager 分別套用至每個 。

一般 AWS 資源

下列一般資源可協助您使用 AWS。

- 課程和研討會 – 連結至以角色為基礎的特殊課程，以及自行安排進度的實驗室，以協助強化您的 AWS 技能並取得實際經驗。
- AWS 開發人員中心 – 探索教學課程、下載工具，並了解 AWS 開發人員事件。
- AWS 開發人員工具 – 開發人員工具、SDKs、IDE 工具組和命令列工具的連結，用於開發和管理 AWS 應用程式。
- 入門資源中心 – 了解如何設定您的 AWS 帳戶、加入 AWS 社群，以及啟動您的第一個應用程式。
- 實用的教學課程 - 按照逐步教學課程在 AWS上啟動第一個應用程式。
- AWS 白皮書 – 技術 AWS 白皮書完整清單的連結，涵蓋架構、安全和經濟等主題，並由 AWS Solutions Architects 或其他技術專家撰寫。
- AWS 支援 中心 – 建立和管理 AWS 支援 案例的中樞。也包含其他實用資源的連結，例如論壇、技術 FAQs、服務運作狀態和 AWS Trusted Advisor。
- 支援 – 有關one-on-one、快速回應支援管道的資訊的主要網頁 支援，可協助您在雲端中建置和執行應用程式。
- 聯絡我們 – 查詢有關 AWS 帳單、帳戶、事件、濫用與其他問題的聯絡中心。
- AWS 網站條款 – 有關我們的著作權和商標、您的帳戶、授權和網站存取，以及其他主題的詳細資訊。

文件歷史記錄

下表說明文件自上次發行以來的重要變更 AWS Systems Manager。如需獲得此文件更新的通知，您可以訂閱 RSS 摘要。

變更	描述	日期
Explorer 資源資料同步可在四個選擇加入中使用 AWS 區域	根據預設，Explorer跨帳戶/跨區域資源資料同步不支援選擇加入區域中的資料彙總。已於 2025 年 6 月 30 日新增下列選擇加入區域的支援。 • 歐洲 (米蘭) • 非洲 (開普敦) • Middle East (Bahrain) • 亞太區域 (香港) 請注意，您無法使用委派的管理員帳戶在選擇加入區域中建立資源資料同步。您必須使用 AWS Organizations 管理帳戶。如需詳細資訊，請參閱了解多個帳戶和區域資源資料同步。	2025 年 6 月 30 日
AWS 參數和秘密 Lambda 延伸模組的新版本	AWS 參數和秘密 Lambda 延伸模組的新版本現在適用於所有支援的區域和架構。此外，已新增亞太區域 （ 台北 ） 區域 (ap-east-2) 的支援。	2025 年 6 月 26 日
新主題：與 Amazon EventBridge Change Calendar整合	Change Calendar 與 Amazon EventBridge 的 主題整合提供下列類別中此整合的相關資訊：	2025 年 6 月 25 日

	• 事件計時和可靠性 • 事件修改和時間考量 • 相鄰和重疊的事件 • 最佳實務	
自動化動作 'aws：executeScript' 的執行期支援更新	Automation 動作 aws：executeScript 中的 `Runtime` 參數現在支援 PowerShell 7.4 (`dotnet8`) 指令碼。	2025 年 6 月 24 日
SSM Agent 和 Patch Manager 現在支援 RHEL 10 和 Ubuntu Server 25.04	您現在可以使用 Systems Manager 管理 Red Hat Enterprise Linux 10 Ubuntu Server 和 25Patch Manager.04 機器，包括使用 修補受管節點。如需支援的作業系統和版本的完整清單，請參閱以下主題： • Systems Manager 支援的作業系統 • Patch Manager 支援的作業系統	2025 年 6 月 23 日

just-in-time節點存取主題的更新	我們已使用 Systems Manager 章節對Just-in-time節點存取中的內容進行下列更新： • 釐清just-in-time節點存取的身分驗證支援。如需詳細資訊，請參閱使用 Systems Manager just-in-time存取。 • 釐清使用標籤 AWS KMS 金鑰 和標籤值 ，在just-in-time節點存取中標記用於Session Manager加密 `SystemsManagerJustInTimeNodeAccessManaged` 和 RDP 記錄的金鑰的需求`true`。如需詳細資訊，請參閱更新just-in-time節點存取工作階段偏好設定和設定 RDP 錄製的 S3 儲存貯體加密。 • 已更新設定just-in-time存取請求的通知，其中包含其他組態詳細資訊。	2025 年 6 月 17 日

增強型 RDP just-in-time 記錄，可即時存取節點	AWS Systems Manager just-in-time節點存取現在提供 RDP 記錄組態的其他資訊。在存放記錄的 S3 儲存貯體上使用 AWS KMS 加密時，您必須提供存取用於儲存貯體加密的 KMS 金鑰給`ssm-guiconnect` 服務主體。此版本也會引入新的錄製狀態 `ProcessingError` ，這表示影片處理期間的中繼失敗，這可能是由於服務相依性失敗或缺少許可。如需詳細資訊，請參閱錄製 RDP 連線。	2025 年 6 月 13 日
更新 中修補程式選取和安裝的相關資訊 Patch Manager	新增資訊，說明 如何在修補程式選取和安裝期間Patch Manager處理過時的套件。如需詳細資訊，請參閱如何選取安全性修補程式和如何安裝修補程式。	2025 年 6 月 12 日
已更新 的受管政策Quick Setup：AWSQuickSetupDeploymentRolePolicy	Systems Manager 已更新受管政策，`AWSQuickSetupDeploymentRolePolicy` 讓 Quick Setup 可以使用此受管政策建立關聯，而不是內嵌政策。如需詳細資訊，請參閱 Systems Manager 對 AWS 受管政策的更新。	2025 年 6 月 10 日

擴展 的簡介 Patch Manager	為了更好地Patch Manager介紹潛在客戶和使用者，我們在Patch Manager章節中新增了以下三個章節： • 如何使我的組織Patch Manager受益？ • 誰應該使用 Patch Manager？ • 的主要功能是什麼Patch Manager？	2025 年 5 月 21 日
已更新 的受管政策Quick Setup：AWSQuickSetupSSMDeploymentRolePolicy	Systems Manager 已更新 受管政策`AWSQuickSetupSSMDeploymentRolePolicy`，以提供許可來標記為統一主控台建立的 IAM 角色和 Lambda 資源。如需詳細資訊，請參閱 Systems Manager 對 AWS 受管政策的更新。	2025 年 5 月 7 日
統一主控台的功能組態選項	Systems Manager 現在會在設定統一主控台時提供功能組態選項。這些功能組態可讓您選擇是否設定預設主機管理組態 (DHMC)、清查中繼資料收集和自動SSM Agent更新。如需詳細資訊，請參閱為組織設定 Systems Manager 統一主控台和為單一帳戶和區域設定 Systems Manager 統一主控台。	2025 年 5 月 7 日

已更新 的受管政策Quick Setup：AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy	Systems Manager 已更新 受管政策AWSSystemsManagerJustInTimeNodeAccessRolePropagationPolicy ，以提供許可來標記共用的資源 AWS Resource Access Manager ，以便just-in-time存取節點。如需詳細資訊，請參閱 Systems Manager 對 AWS 受管政策的更新。	2025 年 4 月 29 日
已更新 的受管政策Quick Setup：AWSQuickSetupManageJITNAResourcesExecutionPolicy	Systems Manager 已更新 受管政策AWSQuickSetupManageJITNAResourcesExecutionPolicy ，以提供許可來標記just-in-time角色。如需詳細資訊，請參閱 Systems Manager 對 AWS 受管政策的更新。	2025 年 4 月 29 日
Systems Manager just-in-time節點存取	AWS Systems Manager 現在支援連線至受管節點的just-in-time存取方法。這種連接到受管節點的新方法有助於提高節點的安全性，只需在需要時提供節點的暫時、有時間限制的存取，無需長期的許可。此外，Systems Manager 會將 RDP 工作階段的工作階段記錄提供給Windows Server節點，協助您符合合規要求、執行根本原因分析等。如需詳細資訊，請參閱使用 Systems Manager Just-in-time節點存取。	2025 年 4 月 29 日

just-in-time節點存取的新受管政策	Systems Manager 已發佈數個新的受管政策，以支援just-in-time節點存取。如需詳細資訊，請參閱 Systems Manager 對 AWS 受管政策的更新。	2025 年 4 月 29 日
AWS 參數和秘密 Lambda 延伸模組的新版本	AWS 參數和秘密 Lambda 延伸模組的新版本現在適用於所有支援的區域和架構。	2025 年 4 月 23 日
新主題：「什麼是 中的合規 Patch Manager？」	您機群中受管節點的修補程式合規不是由 AWS 或其他第三方定義。新主題 中什麼是合規Patch Manager？ 釐清 Systems Manager 管理之節點的修補程式合規，表示所有修補程式都已安裝在節點上，且該節點符合相關聯修補程式基準中指定的核准條件。合規是由您在修補基準中設定的規則或您為修補操作 AWS 選擇的預先定義修補基準中定義的。	2025 年 4 月 11 日
AWS 參數和秘密 Lambda 延伸模組的新版本	現已推出 AWS 參數和秘密 Lambda 延伸的新版本。此外，已針對下列區域新增延伸支援： • 亞太區域 （ 泰國 ） (ap-southeast-7) • 墨西哥 （ 中部 ） (mx-central-1)	2025 年 4 月 3 日

新主題：設定合規許可	作為安全最佳實務，我們建議您更新受管節點使用的 AWS Identity and Access Management (IAM) 角色，以限制節點使用 PutComplianceItems API 動作的能力。此 API 動作會在指定的資源上註冊合規類型和其他合規詳細資訊，例如 Amazon EC2 執行個體或受管節點。如需詳細資訊，請參閱設定合規許可。	2025 年 3 月 11 日
更新主題：檢閱節點洞見	在 Systems Manager 的統一主控台中，檢閱節點洞察頁面會報告組織或帳戶中受管和未受管 EC2 執行個體的數量。我們已釐清 Systems Manager 在處於 `Stopped` 狀態時，會在「未受管」計數中包含 EC2 執行個體。如需詳細資訊，請參閱什麼是未受管執行個體？。	2025 年 2 月 25 日
更新主題：驗證 的簽章 SSM Agent	Linux 執行個體的 AWS Systems Manager Agent (SSM Agent) RPM 和 Debian 安裝程式套件會以密碼編譯方式簽署。您可以使用公有金鑰來驗證代理程式套件為原版且未經修改。如果檔案損壞或遭到修改，驗證會失敗。您可以使用 RPM 或 GPG 來驗證安裝程式套件的簽章。主題包含 的新公有金鑰SSM Agent。如需詳細資訊，請參閱驗證 的簽章SSM Agent。	2025 年 2 月 14 日

新主題：驗證Session Manager外掛程式的簽章	Linux 執行個體的Session Manager外掛程式 RPM 和 Debian 安裝程式套件會以密碼編譯方式簽署。您可以使用公有金鑰來驗證外掛程式二進位檔和套件是否為原始且未修改。如果檔案遭到修改或損壞，驗證會失敗。您可以使用 GNU Privacy Guard (GPG) 工具來驗證安裝程式套件的簽章。如需詳細資訊，請參閱驗證Session Manager外掛程式的簽章。	2025 年 2 月 13 日
Session Manager 外掛程式已更新	Session Manager 外掛程式最近更新了下列增強功能：在 Dockerfile 中將 Go 版本升級至 1.23。已更新 README 中的版本組態步驟。如需Session Manager外掛程式的詳細資訊，請參閱安裝 的Session Manager外掛程式 AWS CLI。	2025 年 2 月 6 日
針對自動化 API 動作發佈的 TPS 配額	我們已針對數個 Systems Manager Automation 相關 API 動作發佈每秒交易 (TPS) 配額，先前稱為有限。如需相關資訊，請參閱《Amazon Web Services 一般參考》中的 Systems Manager 服務配額一節。	2025 年 1 月 21 日

術語變更：Systems Manager 功能重新命名為「工具」	到目前為止，構成 Systems Manager 的功能單位稱為功能，例如自動化功能、Parameter Store功能等。這些功能單位現在稱為工具，例如自動化工具和 Parameter Store 工具。此使用者指南已更新以反映變更。	2025 年 1 月 10 日
Patch Manager 支援其他 macOS版本 (Amazon EC2 AMIs)	Patch Manager 現在支援 12.7、13.6 和 13.7 版，以及 15.x macOS版。如需 支援的 OSs和版本的完整清單Patch Manager，請參閱 支援的作業系統Patch Manager。	2025 年 1 月 8 日
SSM Agent 和 Patch Manager 支援其他版本：CentOS Stream、Ubuntu Server 和 Windows Server	Patch Manager 現支援 CentOS Stream 9、Ubuntu Server 24.10 和 Windows Server 2025。SSM Agent 現支援 Ubuntu Server 24.10 和 Windows Server 2025。(先前已發布對 CentOS Stream 9 的代理程式支援。) 如需支援的作業系統和版本的完整清單，請參閱以下主題： • Systems Manager 支援的作業系統 • Patch Manager 支援的作業系統	2024 年 11 月 22 日

新主題：Parameter StoreSecureString 參數的 AWS KMS 加密	在下列主題Parameter Store中，了解 AWS Systems ManagerParameter Store 如何使用 AWS Key Management Service 來加密中的`SecureString` 參數值： • AWS KMS 中的 SecureString 參數加密 Parameter Store	2024 年 11 月 22 日
Systems Manager 全新和更新的受管政策	為了支援 Systems Manager 的新功能，我們即將發布多項新的受管政策，用以支援 Systems Manager 的新組態和操作，也將更新其他受管政策。如需詳細資訊，請參閱 Systems Manager 對 AWS 受管政策的更新。	2024 年 11 月 21 日

Systems Manager 全新、簡化的節點管理體驗	AWS Systems Manager 已發佈新的統一主控台體驗，用於跨帳戶和區域大規模管理節點。您現在可以從單一位置查看組織和區域的所有受管 AWS 帳戶 和未受管節點。您也可以識別、診斷和修復非受管節點。Systems Manager 現在也與 Amazon Q Developer (Amazon Q) 整合，透過 AWS Management Console 輸入自然語言提示，擴展您從 中的任何位置查看和控制節點的能力。在此版本中，您也可以使用 AWS Organizations 來允許委派管理員帳戶從中央觀點管理整個組織的節點。如需詳細資訊，請參閱下列主題： • 什麼是 AWS Systems Manager？ • 設定 AWS Systems Manager • 使用 完成節點任務 AWS Systems Manager	2024 年 11 月 21 日
Session Manager 外掛程式錯誤修正	Session Manager 外掛程式最近已更新，修正了下列錯誤：已復原將憑證新增至 OpenDataChannel 請求的變更。	2024 年 11 月 20 日

Session Manager 外掛程式增強功能	此版本已於 2024 年 11 月 20 日遭棄用。 Session Manager 外掛程式最近已更新，新增了下列增強功能。 • 已將憑證新增至 OpenDataChannel 請求。 • 升級了 `testify` 和 `objx` 相依套件。	2024 年 11 月 6 日
macOS 支援的其他作業系統版本	Systems Manager 現支援 macOS 作業系統的 15.x 版 (Sequoia) (僅限 EC2 執行個體)。如需所有支援的作業系統和版本的清單，請參閱 Systems Manager 支援的作業系統。	2024 年 11 月 6 日
Patch Manager：更新已核准和已遭拒清單的支援套件名稱格式	我們已針對數個作業系統更新並擴充套件名稱的格式清單，您可以在修補基準的核准的修補程式和遭拒的修補程式清單中指定這些格式。如需詳細資訊，請參閱 Amazon Linux 1、Amazon Linux 2、Amazon Linux 2022、Amazon Linux 2023、CentOS、Oracle Linux 和 Red Hat Enterprise Linux (RHEL) 的套件名稱格式。	2024 年 11 月 1 日

Patch Manager 支援的其他作業系統版本	Patch Manager 現支援 Oracle Linux (8.10 和 9.4) 和 Ubuntu Server (23.10 和 24.04) 的其他版本。如需所有支援的作業系統和版本的清單，請參閱以下主題： • Systems Manager 支援的作業系統 • Patch Manager 支援的作業系統	2024 年 11 月 1 日
SSM Agent 支援其他版本：CentOS Stream、Oracle Linux 和 Ubuntu Server	除了支援的舊版，SSM Agent 現在也支援 CentOS Stream 9、Oracle Linux 8.10 和 9.4 和 Ubuntu Server 24.04 LTS。如需 Systems Manager 支援的作業系統和版本的完整清單，請參閱 Systems Manager 支援的作業系統。	2024 年 10 月 25 日
Session Manager 外掛程式增強	新增了支援使用 OpenDataChannel 請求傳遞外掛程式版本。	2024 年 10 月 10 日
新增：檢視使用 Fleet Manager 進行 RDP 連線的詳細資訊	您現在可以檢視 中使用者所建立遠端桌面通訊協定連線的相關資訊 AWS 帳戶。如需詳細資訊，請參閱檢視目前和已完成連線的相關資訊。	2024 年 10 月 10 日

Patch Manager 目前支援 SLES 版本 15.6	已發布對 SUSE Linux Enterprise Server (SLES) 15.6 的修補支援。您現在可以使用 Patch Manager 修補 SLES 15.6 機器。如需 Patch Manager 支援的作業系統和版本的完整清單，請參閱 Patch Manager 支援的作業系統。	2024 年 9 月 29 日

AWS 參數和秘密 Lambda 延伸模組的新版本	現已推出 AWS 參數和秘密 Lambda 延伸的新版本。已針對亞太區域 (馬來西亞) (ap-southeast-5) 推出所有架構的支援。此外，已為下列區域新增 ARM64 和 Mac with Apple silicon 架構延伸支援： • 亞太區域 (海德拉巴) (ap-south-2) • 亞太區域 (墨爾本) (ap-southeast-4) • 加拿大西部 (卡加利) (ca-west-1) • 歐洲 (蘇黎世) (eu-central-2) • 歐洲 (西班牙) (eu-south-2) • 中東 (阿拉伯聯合大公國) (me-central-) • 中國 (北京) (cn-north-1) • 中國 (寧夏) (cn-northwest-1) • 以色列 (特拉維夫) (il-central-1) • AWS GovCloud （美國東部） (us-gov-east-1) • AWS GovCloud （美國西部） (us-gov-west-1)	2024 年 9 月 19 日
新主題：使用 AWS CLI設定維護時段的許可	使用 AWS CLI設定維護時段的許可主題會提供指示，說明如何建立自訂服務角色 (及其政策) 來代表使用者執行維護時段任務。	2024 年 8 月 19 日

SSM Agent 和 Patch Manager 支援其他版本：AlmaLinux、Oracle Linux 和 Rocky Linux	除了較舊的支援版本，SSM Agent 和 Patch Manager 現在也支援 8.10、9.3 和 9.4 版的 AlmaLinux 和 Rocky Linux，以及 Oracle Linux 9.3 版。如需支援的作業系統和版本的完整清單，請參閱以下主題： • Systems Manager 支援的作業系統 • Patch Manager 支援的作業系統	2024 年 8 月 14 日
Parameter Store 支援的新 IAM 政策條件：ssm:Policies	您可以使用 `ssm:Policies` (IAM 政策的新支援條件)，防止實體建立或更新內含參數政策的參數。如需詳細資訊，請參閱下列主題： • ssm:Policies：防止建立或更新使用參數政策的參數 • 《服務授權參考》中的 Condition keys for AWS Systems Manager。	2024 年 8 月 14 日
Quick Setup 的已更新受管政策：SSMQuickSetupRolePolicy	Systems Manager 已更新 受管政策`SSMQuickSetupRolePolicy` ，以提供其他 AWS CloudFormation 堆疊集的存取權。如需詳細資訊，請參閱 Systems Manager 對 AWS 受管政策的更新。	2024 年 8 月 13 日

支援使用 Terraform 佈建和管理 Systems Manager 資源	我們已將 HashiCorp Terraform 新增至 Systems Manager 支援的第三方整合清單中。Terraform 是一種開放原始碼基礎設施即程式碼 (IaC) 軟體工具，可提供命令列介面 (CLI) 工作流程來管理各種雲端服務。您可以使用 Terraform 來佈建和管理許多常用的 Systems Manager 資源和資料來源。如需有關此整合以及 Systems Manager 支援的其他第三方整合的資訊，請參閱與其他產品和服務的整合。	2024 年 8 月 1 日
Quick Setup 的新主控台體驗和 API	Systems Manager Quick Setup 發布了新的主控台體驗和 API。現在您可以使用 主控台 AWS CLI AWS CloudFormation、 和 SDKs 與此 API 互動。您可以使用 Quick Setup 主控台選擇加入新的主控台。如需有關 Quick Setup 新體驗入門的詳細資訊，請參閱 Quick Setup 入門。如需有關可透過 Quick Setup API 使用的 API 操作的詳細資訊，請參閱《Quick Setup API 參考》。	2024 年 8 月 1 日

新主題：自訂修補基準中遭拒的修補程式清單選項	對於在 Patch Manager 中使用自訂修補基準的修補操作，我們已釐清在新增至遭拒的修補程式清單的修補程式獲指派動作 `Allow as dependency` 時的行為。由於 Windows Server 不支援修補程式相依性的概念，因此未安裝在受管節點的修補程式會遭到略過。已安裝在節點的修補程式會獲指派狀態 `INSTALLED_REJECTED` 。如需詳細資訊，請參閱自訂修補基準中遭拒的修補程式清單選項以及其他作業系統的修補程式合規值。	2024 年 7 月 23 日
新主題：設定 SSM Agent 以與聯邦資訊處理標準 (FIPS) 搭配使用	我們已提供指示，說明如何設定 SSM Agent 以與聯邦資訊處理標準 (FIPS) 搭配使用。如需詳細資訊，請參閱設定 SSM Agent 以與聯邦資訊處理標準 (FIPS) 使用搭配。	2024 年 7 月 22 日
更新：已釐清對 Fleet Manager 使用者名稱中 @ 符號的支援	如果 IAM Identity Center 使用者名稱包含一個或多個 @ 符號，Fleet Manager RDP 會忽略第一個 @ 符號及其後面的所有字元，無論 @ 是否引入電子郵件地址的網域部分。如需有關 Fleet Manager RDP 連線中使用者名稱支援字元的詳細資訊，請參閱驗證遠端桌面連線。	2024 年 7 月 21 日

更新的受管政策：AmazonSSMManagedEC2InstanceDefaultPolicy	Systems Manager 已透過提供內嵌陳述式 ID (Sid) 來更新受管政策 `AmazonSSMManagedEC2InstanceDefaultPolicy` ，釐清了每個政策陳述式的目的。如需詳細資訊，請參閱 Systems Manager 對 AWS 受管政策的更新。	2024 年 7 月 18 日

用於Patch Manager修補操作的 AWS 受管儲存貯體名稱變更	AWS 擁有和維護許多在執行各種Patch Manager修補操作的過程中SSM Agent存取的 Amazon S3 儲存貯體。可公開存取這些 S3 儲存貯體，SSM Agent 預設使用 HTTP 呼叫連線到它們。但是，如果您在 Systems Manager 操作中使用虛擬私有雲端 (VPC) 端點，則必須在適用於 Systems Manager 的 Amazon EC2 執行個體設定檔中提供明確許可，或在混合多雲端環境中非 EC2 機器的服務角色中提供明確許可。否則，您的資源無法存取這些公有儲存貯體。在大多數情況下，我們會變更這些儲存貯體的名稱。例如，針對修補操作，`aws-patch manager-macos-us-e ast-2-552881074` 會取代 `aws-patchmanager-macos-us-east-2` 儲存貯體，而 `aws-patch-manager-us-east-2 -552881074` 會取代 `aws-ssm-us-east-2` 儲存貯體。如需詳細資訊，請參閱下列主題： • SSM Agent 與 AWS 受管 S3 儲存貯體的通訊 • 參考：用於修補操作的 Amazon S3 儲存貯體	2024 年 7 月 18 日

Quick Setup 的全新服務連結角色	Systems Manager 發布了新的服務連結角色 `AWSServiceRoleForSSMQuickSetup` 。Systems Manager 使用此角色來檢查使用 Quick Setup 設定的資源組態運作狀態、確保使用一致的參數和佈建資源，以及在偵測到漂移時修復資源。與此角色相關聯的受管政策為 `SSMQuickSetupRolePolicy` 。如需詳細資訊，請參閱 Systems Manager 的 AWSServiceRoleForSSMQuickSetup 服務連結角色許可。	2024 年 7 月 3 日
適用於 Quick Setup 組態類型的全新受管政策	Systems Manager 發布了另外 12 項新的受管政策，用以支援各種 Quick Setup 組態類型和程序。如需詳細資訊，請參閱 Systems Manager 對 AWS 受管政策的更新。	2024 年 7 月 3 日
支援 RHEL 8.10 和 9.4	Systems Manager 和 Patch Manager 現支援 Red Hat Enterprise Linux 8.10 和 9.4 版。如需詳細資訊，請參閱支援的作業系統和機器類型以及 Patch Manager 支援的作業系統。	2024 年 6 月 26 日

Patch Manager 支援 8.8 和 8.9 版本：AlmaLinux、Oracle Linux 和 Rocky Linux	除了較舊的 8.x 版本，Patch Manager 現在也支援 AlmaLinux、Oracle Linux 和 Rocky Linux 的 8.8 和 8.9 版。如需 Patch Manager 支援的作業系統和版本的完整清單，請參閱 Patch Manager 支援的作業系統。	2024 年 6 月 17 日
macOS Amazon EC2 AMIs 的全新公有參數	已發布公有參數，用以支援 macOS Amazon Elastic Compute Cloud 執行個體的 Amazon Machine Images。如需詳細資訊，請參閱下列主題。 • 尋找公有參數 • 呼叫 macOS 的 AMI 公有參數 • 《Amazon EC2 使用者指南》中的 Launch a Mac instance • 《Amazon EC2 使用者指南》中的 Amazon EC2 Mac instances • 《Amazon EC2 Auto Scaling 使用者指南》中的在啟動範本中使用 AWS Systems Manager 參數而非 AMI IDs	2024 年 6 月 17 日

更新：/aws/service/global-infrastructure 參數路徑的區域可用性	我們已釐清可透過哪些商業區域查詢 /aws/service/global-infrastructure 公有參數路徑，以及如果在不同的商業 AWS 區域中作業時，如何執行路徑查詢。如需詳細資訊，請參閱呼叫 AWS 服務、區域、端點、可用區域、本機區域和 Wavelength 區域的公有參數。	2024 年 6 月 12 日
新增：程式碼範例章節	新的一章是使用 AWS SDKs 的 Systems Manager 程式碼範例，提供不同 SDK 語言的範例，說明如何使用 Systems Manager 服務。	2024 年 5 月 8 日
ec2messages:* 端點支援的變更	對於在 2024 年或之後 AWS 區域 啟動，ec2messages:* 端點不支援將狀態和執行資訊 SSM Agent傳回 Systems Manager 服務。這些區域中的帳戶必須使用 ssmmessages:* 。在 2024 年之前推出的區域中，系統仍然支援 ssmmessages:* 和 ec2messages:* ，但建議僅使用 ssmmessages:* 端點 (Amazon Message Gateway Service)。此時您可以安全地將 ec2messages:* 許可從政策中移除。如需詳細資訊，請參閱使用 SSM Agent 和代理程式相關的 API 操作 (ssmmessages 和 ec2messages 端點)。	2024 年 5 月 3 日

在 Automation 執行手冊中執行指令碼時可用的其他執行時期	`aws:executeScript` 動作現支援 Python 3.9、3.10 和 3.11 執行時期。如需有關如何使用此動作的詳細資訊，請參閱 `aws:executeScript` 。	2024 年 4 月 23 日
支援 8.8 和 8.9 版本：AlmaLinux、Oracle Linux 和 Rocky Linux	除了較舊的 8.x 版本，Systems Manager 現在也支援 AlmaLinux、Oracle Linux 和 Rocky Linux 的 8.8 和 8.9 版。如需支援的作業系統和版本的完整清單，請參閱 Systems Manager 支援的作業系統。	2024 年 4 月 22 日
Patch Manager：變更為修補狀態 INSTALLED_PENDING_REBOOT	先前，只能夠將透過 Patch Manager 安裝的修補程式標記為 `INSTALLED_PENDING_REBOOT` 。安裝於 Patch Manager 之外的修補程式，一律不會具有此狀態。如今，`INSTALLED_PENDING_REBOOT` 可以套用至自上次重新啟動後套用至受管節點的任何修補程式。這包括透過選取 `NoReboot` 選項安裝 Patch Manager 的修補程式，以及在節點最近一次重新啟動後安裝在 Patch Manager 之外的修補程式。如需所有 Patch Manager 修補狀態值的說明，請參閱了解修補程式合規狀態值。	2024 年 4 月 16 日

支援 RHEL 8.9 和 9.3	除了較舊的 8.x 和 9.x 版本，Systems Manager (包含Patch Manager) 現在也支援 Red Hat Enterprise Linux (RHEL) 8.9 和 9.3 版。	2024 年 3 月 26 日
主題更新： 的 AWS 受管政策 AWS Systems Manager	AWS Systems Manager的AWS 受管政策主題提供相關資訊，說明自 2021 年 3 月 12 日起，針對 Systems Manager 推出或更新的四項受管政策。我們已在此主題新增一節，其中包含 12 項其他受管政策的相關資訊，這些政策可與在該日期之前建立或最後一次更新的 Systems Manager 搭配使用。如需詳細資訊，請參閱 Systems Manager 的其他受管政策。	2024 年 3 月 1 日
Parameter Store 現支援跨帳戶共用	您現在可以透過設定資源共用，在 AWS 組織之間 AWS 帳戶 或內部安全且有效率地共用進階參數。資源共用可讓您集中控制應用程式組態管理，降低與您擁有的每個單一帳戶共用參數的營運開銷。您可以使用 Parameter Store主控台、AWS RAM 主控台或 跨帳戶共用參數 AWS CLI。如需詳細資訊，請參閱使用共用參數。	2024 年 2 月 21 日

自動化動作增強功能	您現在可以將 `onFailure` 和 `isCritical` 屬性與 `aws:approve` 動作搭配使用。如需有關 `aws:approve` 動作的詳細資訊，請參閱 aws：approve：暫停自動化以進行手動核准。	2024 年 2 月 12 日
Patch Manager 支援的其他作業系統版本	我們已新增 Patch Manager 支援的作業系統版本清單。新增對以下系統版本的支援： • 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 主控台設定自動化的 SSM Agent 更新	您現在可以使用 Application Manager 主控台自動化應用程式執行個體的 SSM Agent 更新。如需詳細資訊，請參閱使用您的應用程式執行個體。	2023 年 12 月 21 日
在混合多雲端環境中註冊非 Amazon EC2 機器的更新程序	Systems Manager 現在提供 `ssm-setup-cli` 以協助您在混合多雲端環境中註冊非 Amazon Elastic Compute Cloud (Amazon EC2) 機器。如需詳細資訊，請參閱如何在混合 Linux 節點上安裝 SSM Agent，以及如何在混合 Windows 節點上安裝 SSM Agent。	2023 年 12 月 20 日

使用 Fleet Manager 管理 Amazon EBS 磁碟區	您現在可以使用 中的Fleet Manager工具 AWS Systems Manager，在受管執行個體上管理 Amazon Elastic Block Store 磁碟區。例如，您可以初始化 EBS 磁碟區、格式化分割區，然後掛載磁碟區以供使用。如需詳細資訊，請參閱 EBS 磁碟區管理。	2023 年 12 月 14 日
Session Manager 外掛程式增強	新增將 StartSession API 回應作為環境變數傳遞給 session-manager-plugin 的支援。	2023 年 12 月 4 日
Automation 執行手冊的全新視覺化設計體驗	您現在可以使用由 Systems Manager Automation 開發的全新視覺化設計體驗來建立和編輯執行手冊。視覺化設計體驗提供低程式碼、拖放式介面，讓您可以更輕鬆地建立和編輯執行手冊。如需詳細資訊，請參閱 Automation 執行手冊的視覺化設計體驗。	2023 年 11 月 26 日

適用於執行手冊的新 Systems Manager Automation 動作、資料元素和功能增強	您現在可以使用 aws:loop 動作在執行手冊中循環執行多個動作。這個新的動作支援 do while 和 for each 樣式迴圈。此外，使用新的變數資料元素，您可以在執行手冊的內容中動態定義、參考和更新值。若要更新執行手冊中變數的值，請使用新的 aws:updateVariable 動作。Automation 還增加了對輸出動態資料類型轉換的支援。這意味著，如果輸出的值與您指定的資料類型不符，則 Automation 會嘗試轉換資料類型。例如，若返回的值是 Integer，但指定的 Type 是 String，則最終輸出值是 String 值。最後，Automation 現在支援選取器的 JSONPath 篩選條件運算式。如需詳細資訊，請參閱下列主題： • aws:loop - 迭代自動化中的步驟 • aws:updateVariable – 更新一個執行手冊變數的值 • 資料元素和參數 - 頂層資料元素 • 使用動作輸出作為輸入。 • 在執行手冊中使用 JSONPath。	2023 年 11 月 17 日

更新了 Remote Desktop Protocol (RDP) 連線的區域支援	由 Amazon DCV 提供支援的 Fleet Manager 遠端桌面，可讓您直接從 Systems Manager 主控台安全連線至 Windows Server 執行個體。已針對 Fleet Manager 遠端桌面連線啟用下列三個額外的區域： • 非洲 (開普敦) (af-south-1) • 亞太區域 (雅加達) (ap-southeast-3) • 以色列 (特拉維夫) (il-central-1)	2023 年 11 月 15 日
Patch Manager：擴充的 OS 版本支援 RHEL 和 macOS	Patch Manager 現在支援下列其他作業系統版本： • 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
全新 OpsCenter API - DeleteOpsItem	OpsCenter 現在提供用於刪除個別 OpsItems 的 DeleteOpsItem API。如需詳細資訊，請參閱 AWS Systems Manager API 參考中的 DeleteOpsItem。	2023 年 10 月 20 日

Quick Setup 全新組態類型：更新整個組織的 SSM Agent	新的組態類型 預設主機管理組態可讓 中定義的組織管理員 AWS Organizations提示自動檢查和更新組織帳戶和區域中所有 EC2 執行個體SSM Agent上的 。如需詳細資訊，請參閱組織的預設主機管理。	2023 年 10 月 16 日
由 CloudWatch Application Insights 所建立的 OpsItems 的新標題和描述格式	CloudWatch Application Insights 所建立的 OpsItems 的標題和描述於 2023 年 10 月 16 日變更為改良的格式。若要檢視此新格式，請參閱 Amazon CloudWatch Application Insights。	2023 年 9 月 29 日
在 Fleet Manager RDP 連線中支援多重顯示解析度	現在，當您使用 Fleet Manager 中的遠端桌面通訊協定 (RDP) 選項連線到 Windows Server 受管節點時，您可以選擇顯示解析度。以前，所有連線都使用了固定的 720P (1366 x 768) 解析度。您現在可以為每個連線從下列選項中進行選擇： • 自動調整 (根據偵測到的螢幕大小自動設定最佳解析度) • 1920 x 1080 • 1400 x 900 • 1366 x 768 • 800 x 600 如需相關資訊，請參閱使用遠端桌面連線至受管節點。	2023 年 9 月 22 日

新主題：修補程式政策操作中的隨機修補基準 ID	新增了內容說明 Quick Setup 修補程式政策如何使用 AWS-RunPatchBaseline SSM Command 文件中的 BaselineOverride 參數，在每次執行修補程式政策作業時為修補基準產生隨機 ID。如需相關資訊，請參閱修補程式政策作業中的隨機修補基準 ID。	2023 年 9 月 22 日
用於管理 OpsItems 的全新操作洞察結果	OpsCenter 現在包含一項名為產生最多 OpsItems 的資源的操作洞察結果。當 AWS 資源有超過 10 個開啟的 時，就會產生此類型的洞見OpsItems。使用此洞察結果來找出有問題的資源。使用洞察結果中的 AWS-BulkResolveOpsItems 執行手冊快速解決與某個資源相關聯的 OpsItems。如需詳細資訊，請參閱分析操作洞察結果以減少 OpsItems。	2023 年 9 月 22 日
更新了 GPG 公有金鑰	已建立新的公有金鑰來驗證 SSM Agent 的簽章。如需詳細資訊，請參閱驗證 SSM Agent 的簽章。	2023 年 9 月 5 日

增加了對更多版本的 Almalinux、Oracle Linux、RHEL 和 Rocky Linux 的支援	更新了 AWS Systems Manager 和 Patch Manager 支援的作業系統清單，以反映對下列作業系統版本的支援： • 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 以及 9.0–9.2	2023 年 8 月 30 日
OpsCenter 的 OpsItem 描述欄位增加對 Markdown 格式的支援。	OpsCenter 的 OpsItem 描述欄位現支援 Markdown 格式。支援下列類型的 Markdown 格式： • 段落 • 行距 • 水平線 • 標題 • 文字格式 • 連結 • 清單 如需詳細資訊，請參閱《 入門指南》中的在主控台中使用 Markdown。 AWS Management Console	2023 年 8 月 18 日

AWS 參數和秘密 Lambda 延伸模組的新版本	AWS 參數和秘密 Lambda 延伸的新版本現已推出。此外，亞太區域 (墨爾本) (ap-Southest-4) 和以色列 (特拉維夫) (il-Central-1) 區域 (僅限 x86_64 和 x86 架構) 新增對延伸的支援。如需詳細資訊，請參閱在 AWS Lambda 函數中使用Parameter Store參數。	2023 年 8 月 16 日
更新：增加了有關 Quick Setup 修補程式政策儲存貯體所需許可的資訊	建立修補程式政策時，Quick Setup 會建立一個包含名為 `baseline_overrides.json` 的檔案的 Amazon S3 儲存貯體。此檔案儲存您為修補程式政策指定之修補基準的相關資訊。設定修補程式政策時，您可以選取將必要的 IAM 政策新增至連結至執行個體的現有執行個體設定檔核取方塊。如果您選擇不選取此選項，則必須手動為特定資源提供存取此儲存貯體的許可，否則您的政策作業可能會失敗。如需詳細資訊，請參閱下列主題： • 適用於修補程式政策 S3 儲存貯體的許可 • 問題：`baseline_overrides.json` 的 "Invoke-PatchBaselineOperation : Access Denied" 錯誤或 "Unable to download file from S3" 錯誤	2023 年 7 月 6 日

使用 Quick Setup 設定 OpsCenter 以實現多帳戶 OpsItem 管理	OpsCenter 的 Quick Setup 可協助您完成實現跨帳戶管理 OpsItems 所需進行的下列任務： • 指定受委派管理員帳戶 • 建立 required AWS Identity and Access Management (IAM) 政策和角色 • 指定 AWS Organizations 組織或成員帳戶的子集，委派管理員可以OpsItems跨帳戶管理 如需詳細資訊，請參閱 (選用) 使用 Quick Setup 設定 OpsCenter 以跨帳戶管理 OpsItems。	2023 年 6 月 19 日
使用 Quick Setup 更新 Amazon EC2 啟動代理程式	您現在可以讓 Systems Manager 每隔 30 天檢查執行個體上安裝的啟動代理程式的新版本。如果有新版本，則 Systems Manager 會更新執行個體上的代理程式。如需詳細資訊，請參閱 Quick Setup 主機管理。	2023 年 6 月 19 日
Patch Manager 現支援 Ubuntu Server 22.04 LTS	您現在可以使用 Patch Manager 來修補 Ubuntu Server 22.04 LTS 節點。如同其他支援的 22.04 LTS Ubuntu Server版本， 使用 AWS 受管`AWS-UbuntuDefaultPatchBaseline` 修補程式基準。	2023 年 5 月 15 日

Systems Manager 現支援 Almalinux，包含 Patch Manager	您現在可以使用 Systems Manager 來管理 AlmaLinux 8.3-8.7 和 9.0-9.1 節點。許多套用至 RHEL 8 的修補規則也適用於 AlmaLinux。AlmaLinux 使用新的 `AWS-DefaultAlmaLinuxPatchBaseline` 。如需詳細資訊，請參閱下列主題： • 在 AlmaLinux 執行個體上手動安裝 SSM Agent • 如何選取安全性修補程式 • 如何安裝修補程式 • 修補基準規則在 AlmaLinux、RHEL 和 Rocky Linux 上的運作方式。	2023 年 5 月 8 日
使用 Quick Setup 部署 EC2Launch v2 代理程式	您現在可以使用 Quick Setup 部署 EC2Launch v2 代理程式。如需詳細資訊，請參閱使用 Quick Setup 部署 Distributor 套件。	2023 年 4 月 13 日

Systems Manager 現支援 Amazon Linux 2023	Systems Manager 現支援新的 Amazon Linux 2023 (AL2023) EC2 執行個體類型，包含對 Patch Manager 操作的支援。適用於 Amazon Linux 2 的許多修補規則也適用於 Amazon Linux 2023。(Patch Manager 還會繼續支援預覽版 Amazon Linux 2022。) 如需詳細資訊，請參閱下列主題： • 如何選取安全性修補程式 • 如何安裝修補程式 • 修補基準規則在 Amazon Linux 1、Amazon Linux 2、Amazon Linux 2022 和 Amazon Linux 2023 上的運作方式	2023 年 3 月 23 日
修訂 Amazon EC2 執行個體的設定內容	我們已修訂 Amazon EC2 執行個體的設定內容。現在建議您使用新發佈的預設主機管理組態來取得執行個體許可。如需詳細資訊，請參閱設定 Systems Manager 所需的執行個體許可。	2023 年 2 月 15 日
使用預設主機管理組態自動管理執行個體	您現在可以使用 Systems Manager 在整個 AWS 區域 中自動管理 Amazon EC2 執行個體。如需詳細資訊，請參閱預設主機管理組態。	2023 年 2 月 15 日

將 SSM 文件新增至收藏	為了協助您尋找常用的 SSM 文件，您現在可以將文件新增至收藏。每個 AWS 帳戶 和 每個文件類型最多可以收藏 20 個文件 AWS 區域。您可以從 Systems Manager 文件主控台選擇、修改及檢視收藏。如需詳細資訊，請參閱將文件新增至收藏。	2023 年 2 月 7 日
使用 Change Calendar 實作 Automation 的變更控制	透過整合自動化與 Change Calendar，您現在可以對 中的所有自動化實作變更控制 AWS 帳戶。如需詳細資訊，請參閱實作 Automation 的變更控制。	2023 年 1 月 24 日
新的 Change Manager 核准工作流程	Change Manager 核准工作流程現在支援逐級核准，而不是逐行核准。之前，您新增至核准層級的每個核准者都必須核准變更請求。否則，該層級不會獲得核准。現在，您可以指定層級需要的核准數量，並且可以新增多個或更多核准者。例如，您可以要求某個層級需要三個核准，但可指定多達五名核准者。這些核准者中的任意三個提供核准皆足以核准該層級。如需詳細資訊，請參閱關於變更範本中的核准。	2023 年 1 月 23 日

新增：使用 Quick Setup 中的修補程式政策為整個組織設定修補程式	使用 Systems Manager 中的工具 Quick Setup，您現在可以建立採用 技術的修補程式政策Patch Manager。修補程式政策會定義自動修補受管節點時要使用的排程和修補基準。使用單一修補程式政策組態，您可以定義為組織中所有區域的所有帳戶、僅您選擇的帳戶和區域或者單一帳戶-區域對進行修補。如需詳細資訊，請參閱下列主題。 • 使用 Quick Setup 修補政策 • 使用 Quick Setup 修補程式政策來自動化整個組織的修補	2022 年 12 月 22 日

Application Manager 與 Amazon EC2 整合，以便在應用程式環境中顯示執行個體的相關資訊。	Application Manager 會以圖形格式顯示所選應用程式的執行個體狀態、狀態和 Amazon EC2 Auto Scaling 運作狀態。Instances (執行個體) 標籤也包含一個資料表，其中包含應用程式中每個執行個體的下列資訊。
	• 執行個體狀態 (待定、停止中、執行中、已停止)
	• SSM Agent 的 Ping 狀態
	• 在執行個體上處理的最新 Systems Manager Automation 執行手冊的狀態和名稱
	• 每個狀態的 Amazon CloudWatch Logs 警示計數。
	  • ALARM – 指標或表達式在定義的閾值外。
	  • OK – 指標或表達式在定義的閾值內。
	  • INSUFFICIENT_DATA – 警示剛開始無法使用指標，或資料不足無法讓指標判斷警示狀態。
	• 父群組和個別自動擴展群組的 Auto Scaling 群組運作狀態
		2022 年 12 月 22 日

使用 Quick Setup，排程 Amazon EC2 執行個體的啟動和停用。	您現在可以部署資源排程器解決方案，使用 Quick Setup 來自動啟動和停用 Amazon EC2 執行個體。如需詳細資訊，請參閱資源排程器。	2022 年 12 月 19 日
OpsCenter 現在支援跨帳戶使用 OpsItems	OpsCenter 支援在工作階段期間從管理帳戶 (AWS Organizations 管理帳戶或 Systems Manager 委派的管理員帳戶) 和成員帳戶中使用 OpsItems。設定完成後，使用者可以執行以下類型的動作： • 在成員帳戶中建立、檢視和更新 OpsItems • 檢視成員帳戶中在 OpsItems 中指定的 AWS 資源詳細資訊 • 啟動 Systems Manager Automation 執行手冊，以解決成員帳戶中 AWS 資源的問題 如需詳細資訊，請參閱設定 OpsCenter 以跨帳戶使用 OpsItems。	2022 年 11 月 16 日

使用 AWS CloudTrail Lake 追蹤Change Manager變更請求的詳細資訊	您現在可以使用 AWS CloudTrail Lake 中的事件資料存放區來擷取和檢閱Change Manager針對您的組織或帳戶在 中執行的變更請求詳細資訊。此資訊包含有關建立變更請求的使用者身分、提出請求的 IP 地址、進行變更 AWS 區域 的 、目標資源等可稽核的詳細資訊。如需相關資訊，請參閱監控您的變更請求事件和檢閱變更請求詳細資訊、任務和時間表。	2022 年 11 月 11 日
使用 CloudWatch 警示的其他 Systems Manager 任務控制措施	透過使用 CloudWatch 警示，現在可在跨多個帳戶和區域執行自動化時實作額外的控制。透過將指標或複合 CloudWatch 警示套用至自動化，您可以根據您定義的指標來控制自動化停止的時間。如需有關將 CloudWatch 警示套用至跨多個帳戶和區域執行的自動化的詳細資訊，請參閱在多個帳戶和區域中執行自動化 (主控台)	2022 年 11 月 9 日

已更新：「在 AWS Lambda 函數中使用Parameter Store參數」	我們已提供其他資訊，協助您使用 AWS Parameters and Secrets Lambda Extension 來擷取參數值並快取這些值，以供未來在 Lambda 函數中使用。使用 Lambda 延伸可藉由減少對 Parameter Store 的 API 呼叫次數來降低成本。如需詳細資訊，請參閱在 AWS Lambda 函數中使用Parameter Store參數。	2022 年 10 月 25 日
使用 CloudWatch 警示的其他 Systems Manager 任務控制措施	在執行自動化和命令時，您現在可以使用 CloudWatch 警示實作額外的控制措施。使用 State Manager 關聯或維護時段任務註冊時，也可以將 CloudWatch 警示新增至自動化或命令。透過將複合 CloudWatch 警示套用至自動化或命令，您可以根據您定義的指標控制自動化或命令停止的時間。如需將 CloudWatch 警示套用至自動化或命令的詳細資訊，請參閱下列程序： • 如何選取安全性修補程式 • 如何安裝修補程式 • 修補基準規則在 Amazon Linux 1、Amazon Linux 2 和 Amazon Linux 2022 上的運作方式。	2022 年 9 月 26 日

使用 CloudWatch 警示的其他 Systems Manager 任務控制措施	在執行自動化和命令時，您現在可以使用 CloudWatch 警示實作額外的控制措施。使用 State Manager 關聯或維護時段任務註冊時，也可以將 CloudWatch 警示新增至自動化或命令。透過將複合 CloudWatch 警示套用至自動化或命令，您可以根據您定義的指標控制自動化或命令停止的時間。如需將 CloudWatch 警示套用至自動化或命令的詳細資訊，請參閱下列程序： • 執行簡易自動化 • 從主控台執行命令 • 建立關聯 • 指派任務至維護時段	2022 年 9 月 26 日
釐清進階執行個體層要求	根據客戶意見回饋，我們已經釐清了需要您在設定執行個體層中啟用進階執行個體層的案例。	2022 年 9 月 21 日
使用 Quick Setup 部署 Amazon CloudWatch 代理程式	您現在可以使用 Quick Setup 部署 Amazon CloudWatch 代理程式。如需詳細資訊，請參閱使用 Quick Setup 部署 Distributor 套件。	2022 年 9 月 20 日

當允許 EC2 執行個體中繼資料時，現在支援針對修補程式群組的「PatchGroup」索引鍵	如果在 EC2 執行個體中繼資料中允許標籤，則您建立的標籤索引鍵不得包含任何空格。先前，這樣可防止客戶將部分 EC2 執行個體新增至 Patch Manager 中的群組，因為標籤索引鍵 `Patch Group` 必須套用至執行個體。Patch Manager 現在支援 `Patch Group` (有空格) 和 `PatchGroup` (沒有空格) 作為識別修補程式群組執行個體的標籤索引鍵。執行個體中繼資料中允許標籤的 EC2 執行個體現在可以新增至 Patch Manager 中的修補程式群組。如需資訊，請參閱關於修補程式群組。	2022 年 8 月 31 日
新主題：「套件發行日期和更新日期的計算方式」	在 管理的修補程式基準中 AWS，新修補程式會在發行或更新後 7 天自動核准。在您建立的自訂修補基準中，您可以選擇性地指定在發行或更新後要等待多少天，然後才會自動核准安裝。對於 Amazon Linux 1 和 Amazon Linux 2，各種因素會影響最新發行日期和更新日期的計算方式。為了協助您避免在選擇自動核准延遲時產生未預期的結果，這些因素會在主題套件發行日期和更新日期的計算方式中說明。	2022 年 8 月 24 日

更新內容：修補 AMI 和更新 Auto Scaling 群組	我們已經更新更新 Auto Scaling 群組的 AMIs 演練，以使用啟動範本而非啟動組態。此外，我們已在 Runbook 內容中實作最新的自動化動作和執行階段。	2022 年 6 月 22 日
Change Manager：阻止使用者建立可自動批准的請求	您可在 Change Manager 中設定變更範本以支援自動核准，這表示具備必要 IAM 許可的使用者可以選擇啟動變更請求，而不需要其他核准。現在您也可以限制個別使用者、群組或 IAM 角色提交自動核准請求，即使有變更範本支援它們。這是透過使用新的 IAM 條件金鑰 `ssm:AutoApprove` 來實現的。如需詳細資訊，請參閱控制對自動核准 Runbook 工作流程的存取	2022 年 6 月 15 日
維護時段任務角色的更新指南	先前，Systems Manager 主控台可讓您選擇 AWS 管理的 IAM 服務連結角色 `AWSServiceRoleForAmazonSSM` 用作任務的維護角色。不再建議將此角色及其關聯政策 `AmazonSSMServiceRolePolicy` ，用於維護時段任務。針對維護時段任務，您應改為建立自訂政策及角色。如需詳細資訊，請參閱設定 Maintenance Windows。	2022 年 6 月 9 日

針對 Session Manager 的網路埠轉送至遠端主機支援	Session Manager 現在支援網路埠轉送工作階段至遠端主機。遠端主機不需由 Systems Manager 管理。如需詳細資訊，請參閱啟動工作階段 (網路埠轉遞送遠端主機)。	2022 年 5 月 25 日
更新內容：針對在 Amazon EC2 Linux 執行個體 手動安裝 SSM Agent 的說明	為了回應客戶意見，針對 Amazon EC2 執行個體手動安裝 SSM Agent 的說明，我們對相關主題進行了全面修改。這些主題現在透過全域可用的檔案提供命令，您可在任何 AWS 區域的 EC2 執行個體上複製並貼上這些命令，進行快速安裝。這些主題也提供資訊，幫助您利用工作區的可用檔案來建立安裝命令。使用指令碼或範本在多個執行個體安裝代理程式時，建議使用後一種方法。如需詳細資訊，請參閱 在 Linux EC2 執行個體上手動安裝 SSM Agent章節的 Linux 作業系統說明。	2022 年 5 月 9 日

新增主題：預先安裝 SSM Agent 的 Amazon Machine Images (AMIs)	為了回應客戶意見，我們把由 AWS 管理的 AMIs 包括預先安裝 SSM Agent 的相關資訊集中在一處。本主題也提供說明，解釋如何驗證從這些 AMIs 所建立的 Amazon EC2 執行個體已成功安裝並正在執行。在極少數情況下，代理程式可能無法成功安裝或已安裝但未啟動，我們也提供資訊說明如何在執行個體啟動或手動安裝代理程式。如需詳細資訊，請參閱預先安裝 Amazon Machine Images 的 AMIs (SSM Agent)。	2022 年 5 月 8 日
新增 State Manager 章節	新增全新章節說明 State Manager 執行關聯的詳細資訊。如需詳細資訊，請參閱關於關聯排程。	2022 年 4 月 27 日
Patch Manager 現在支援 Rocky Linux	您現在可以使用 Patch Manager 來修補 Rocky Linux 節點。許多套用至 RHEL 8 的修補規則也適用於 Rocky Linux。Rocky Linux 8 會使用新的 `AWS-DefaultRockyLinuxPatchBaseline` 。如需詳細資訊，請參閱下列主題： • 如何選取安全性修補程式 • 如何安裝修補程式 • 修補基準規則在 RHEL、CentOS Stream 和 Rocky Linux 上的運作方式。	2022 年 4 月 14 日

Patch Manager 現在支援 CentOS Stream 8	您現在可以使用 Patch Manager 來修補 CentOS Stream 8 執行個體和 Red Hat Enterprise Linux (RHEL) 4.4-4.5 執行個體。許多套用至 RHEL 8 的修補規則也適用於 CentOS Stream 8。CentOS Stream 8 會使用 `AWS-DefaultCentOSPatchBaseline` 。如需詳細資訊，請參閱下列主題： • 如何選取安全性修補程式 • 如何安裝修補程式 • 修補基準規則在 RHEL 和 CentOS Stream 上的運作方式	2022 年 4 月 4 日
建立 Change Manager 的擔任角色	新的一節闡明建立和實作 Change Manager 的 擔任角色 的要求。擔任角色為 AWS Identity and Access Management (IAM) 服務角色，使 Change Manager 能代表您安全地執行在已核准的變更請求中指定的 Runbook 工作流程。角色授予 AWS Systems Manager (AWS STS) `AssumeRole` 信任。 Change Manager如需詳細資訊，請參閱設定 Change Manager 的角色和許可。	2022 年 3 月 18 日

核準或拒絕 Change Manager 大量變更請求	在 Systems Manager 主控台中，您現在可以在單一操作中選擇多個待核准或拒絕的變更請求。如需詳細資訊，請參閱檢閱及核准或拒絕變更請求 (主控台)。	2022 年 3 月 8 日
支援 Rocky Linux 和 Windows Server 2022 的受管節點	Systems Manager 支援 Rocky Linux 和 Windows Server 2022 的受管節點，包括位於內部部署或其他雲端供應商的邊緣裝置和混合機器。若要將 Systems Manager 用於這些作業系統，您必須完成所有必要的 Systems Manager 設定程序，包括混合環境或邊緣裝置的程序 (如適用)。如需詳細資訊，請參閱設定 Systems Manager。關於 Rocky Linux 機器，您還必須手動安裝 SSM Agent。如需詳細資訊，請參閱在 Rocky Linux 執行個體上手動安裝 SSM Agent。關於 Windows Server 2022 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體，預設為安裝 SSM Agent。	2022 年 3 月 1 日

允許 Automation 適應並行需求及檢視 Automation 使用情況指標	現在，您可以允許 Automation 自動調整並行自動化配額，以及檢視發佈到 CloudWatch 的 Automation 使用情況指標。如需有關自適應並行的詳細資訊，請參閱允許 Automation 適應並行需求。如需有關如何檢視 Automation 使用情況指標的詳細資訊，請參閱使用 Amazon CloudWatch 監控 Automation 指標。	2022 年 1 月 27 日
允許 Automation 適應並行需求及檢視 Automation 使用情況指標	現在，您可以允許 Automation 自動調整並行自動化配額，以及檢視發佈到 CloudWatch 的 Automation 使用情況指標。如需有關自適應並行的詳細資訊，請參閱允許 Automation 適應並行需求。如需有關如何檢視 Automation 使用情況指標的詳細資訊，請參閱使用 Amazon CloudWatch 監控 Automation 指標。	2022 年 1 月 27 日
按類別組織的 Systems Manager 文件	Amazon 擁有的 Systems Manager 文件現在按類型和類別組織，以便幫助您找到所需文件。	2022 年 1 月 13 日

建立與叫用 Automation 整合	透過建立整合，您現在可以在自動化過程中使用 Webhook 傳送訊息。在自動化過程中，您可以使用 Runbook 中的新 `aws:invokeWebhook` 動作來叫用整合。如需有關建立整合的詳細資訊，請參閱建立 Automation 的 Webhook 整合。若要進一步了解 `aws:invokeWebhook` 動作，請參閱 aws:invokeWebhook：調用 Automation Webhook 整合。	2022 年 1 月 13 日
新 中無法使用的功能 AWS 區域	下列 Systems Manager 工具目前不適用於新的亞太區域（雅加達）區域。 • Application Manager • Change Calendar • Change Manager • Explorer • Fleet Manager • Incident Manager • Quick Setup	2021 年 12 月 13 日

檢視應用程式的資源成本詳細資訊	Application Manager 透過 Cost Explorer 小工具與 AWS 帳單與成本管理 整合。在帳單和成本管理主控台中啟用 Cost Explorer 後，Application Manager 中的 Cost Explorer 小工具會顯示特定非容器應用程式或應用程式元件的成本資料。您可以使用小工具中的篩選條件，根據長條圖或折線圖中的不同時間週期、粒度和成本類型來檢視成本資料。如需詳細資訊，請參閱檢視應用程式的概觀資訊。	2021 年 12 月 7 日
使用 Fleet Manager 管理程序	您現在可以使用 Fleet Manager 來管理節點上的程序。如需詳細資訊，請參閱使用程序。	2021 年 12 月 6 日
術語變更：受管執行個體現在是受管節點	透過支援 AWS IoT Greengrass 核心裝置，在大多數 Systems Manager 文件中，受管執行個體一詞已變更為受管節點。Systems Manager 主控台、API 呼叫、錯誤訊息和 SSM 文件仍然使用字詞「執行個體」。	2021 年 11 月 29 日

支援邊緣裝置	Systems Manager 支援以下邊緣裝置組態。 • AWS IoT Greengrass：Systems Manager 現在支援為 設定 AWS IoT Greengrass 並執行 AWS IoT Greengrass Core 軟體的任何裝置。若要加入 AWS IoT Greengrass 核心裝置，您必須建立 AWS Identity and Access Management (IAM) 服務角色。您也必須使用 AWS IoT Greengrass 主控台來部署SSM Agent為裝置上的 AWS IoT Greengrass 元件。如需詳細資訊，請參閱AWS Systems Manager 設定邊緣裝置。 • 混合環境中的邊緣裝置：當您將 AWS IoT 核心裝置和非AWS IoT 裝置設定為內部部署機器後，Systems Manager 也支援這些裝置。若要上線裝置，您必須建立 IAM 服務角色、為混合環境建立受管節點啟用，以及在您的裝置上手動安裝 SSM Agent。如需詳細資訊，請參閱AWS Systems Manager 設定混合環境	2021 年 11 月 29 日

使用遠端桌面連線至受管執行個體	您現在可以透過遠端桌面通訊協定 (RDP) 使用 Fleet Manager 連線至受管 Windows 執行個體。這些由 Amazon DCV 提供技術支援的遠端桌面工作階段，直接從瀏覽器提供了執行個體的安全連線。如需詳細資訊，請參閱使用遠端桌面進行連線。	2021 年 11 月 23 日
指定工作階段持續時間上限並提供工作階段的原因	您現在可以為 AWS 帳戶中 AWS 區域 的所有 Session Manager 工作階段指定工作階段持續時間上限。當工作階段達到您指定的持續時間時，就會終止。您現在也可以在啟動工作階段時選擇性地新增原因。如需詳細資訊，請參閱指定工作階段持續時間上限。	2021 年 11 月 16 日

Patch Manager 現在支援 Raspberry Pi OS 作業系統	您現在可以使用 Patch Manager 來修補 Raspberry Pi OS 執行個體。Patch Manager 支援修補 Raspberry Pi OS 9 (Stretch) 和 10 (Buster)。因為 Raspberry Pi OS 是基於 Debian 的作業系統，許多相同的修補規則也對其適用，同樣於對 Debian Server 也適用。如需詳細資訊，請參閱下列主題： • 如何選取安全性修補程式 • 如何安裝修補程式 • 修補基準規則在 Debian Server 和 Raspberry Pi OS 上的運作方式	2021 年 11 月 16 日
存取 Red Hat 知識庫入口網站	使用 Fleet Manager 以存取 RHEL 知識庫入口網站，尋找有關使用 Red Hat 產品的解決方案、文章、文件和影片。如需詳細資訊，請參閱存取 Red Hat 知識庫入口網站。	2021 年 11 月 3 日
大量編輯 OpsItems	OpsCenter 現在支援大量編輯 OpsItems。您可以選取多個 OpsItems，然後編輯下列其中一個欄位：Status (狀態)、Priority (優先順序)、Severity (嚴重性)、Category (類別)。如需詳細資訊，請參閱編輯 OpsItems。	2021 年 10 月 15 日

建立填入 AWS 資源的輸入參數	您現在可以在 AWS Management Console中建立填入 AWS 資源的 Automation Runbook 輸入參數。如需詳細資訊，請參閱建立填入 AWS 資源的輸入參數。	2021 年 10 月 14 日
維護時段的新任務叫用截止選項	您現在可以選擇在到達維護時段指定的截止時間後，阻止任何新的任務叫用啟動。如需詳細資訊，請參閱將任務指派給維護時段 (主控台)。	2021 年 10 月 13 日
Patch Manager 支援 macOS 11.3.1 和 11.4 (Big Sur)	適用於 macOS 11.3.1 和 11.4 (Big Sur) 的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體現在可以使用 Patch Manager 進行修補。這是除了 macOS 10.14.x (Mojave) 和 10.15.x (Catalina) 之現有支援的新增支援。如需有關處理 Patch Manager 的詳細資訊，請參閱 AWS Systems Manager和 Patch Manager。	2021 年 10 月 1 日

Application Manager 中的 Application Insights	Application Manager 與 Amazon CloudWatch Application Insights 整合。Application Insights 會識別和設定金鑰指標、日誌，並在您所有的應用程式資源和技術堆疊中發出警示。Application Insights 會持續監控指標和日誌，以偵測和建立異常及錯誤的關聯。系統偵測到錯誤和異常時，Application Insights 會產生 CloudWatch Events，可用於設定通知或採取動作。您可以在 Application Manager 中的 Overview (概觀) 和 Monitoring (監控) 標籤上啟用和檢視 Application Insights。如需有關 Application Insight 的詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的什麼是 Amazon CloudWatch Application Insights。	2021 年 9 月 21 日

將事件從其他行事曆匯入到 Change Calendar	現在可以將事件從第三方行事曆中匯入到 Change Calendar 中的行事曆。之前，每個事件都必須手動輸入到行事曆中。將支援的第三方行事曆供應商的行事曆匯出至 iCalendar (`.ics`) 檔案，將其匯入 Change Calendar，其事件會包含在 Systems Manager 中已開啟或關閉的行事曆規則中。支援的供應商包括 iCloud 日曆、Google 日曆和 Microsoft Outlook。如需詳細資訊，請參閱從第三方行事曆中匯入及管理事件。	2021 年 9 月 8 日
Application Manager 中的新標籤和 Runbook 功能	標籤增強功能包括在特定資源或 Application Manager 應用程式的所有資源中新增或刪除標籤。Runbook 增強功能包括檢視特定資源類型的已篩選的 Runbook 清單，或在相同類型的所有資源上啟動 Runbook。如需詳細資訊，請參閱在 Application Manager 中使用標籤和在 Application Manager 中使用 Runbook。	2021 年 8 月 31 日
新範例：使用 建立變更請求 AWS CLI	使用 建立變更請求的範例 AWS CLI 已新增至Change Manager章節。範例使用示例 `AWS-HelloWorldChangeTemplate` 變更範本和 `AWS-HelloWorld` runbook： • 建立變更請求 (AWS CLI)	2021 年 8 月 20 日

新章節：在 Amazon EKS 中使用參數	新章節已新增至 Parameter Store 小節。本主題是有關如何在 Amazon EKS 叢集中使用參數的演練。如需詳細資訊，請參閱在 Amazon Elastic Kubernetes Service 中使用 Parameter Store 參數。	2021 年 8 月 19 日
已更新 Patch Manager 生命週期掛鉤	在立即修補修補操作過程中，Patch Manager 現在為其他點提供生lifecycle hook (可執行 Systems Manager 命令文件)。如果排定在執行立即修補後重新啟動執行個體，可以指定在重新啟動完成後執行的lifecycle hook。如需詳細資訊，請參閱使用「立即修補」生命週期掛鉤和關於 AWS-RunPatchBaselineWithHooks SSM 文件。	2021 年 8 月 9 日

自動核准現在支援 Change Manager 請求	現在可在 Change Manager 中設定變更範本以支援自動核准，這表示具有必要 IAM 許可的使用者可以選擇啟動變更請求，而不需要其他核准。具有自動核准範本存取權的使用者仍然可以選擇指定核准者 (如果他們選擇)。為了協助您控制 Change Manager 程序，在變更凍結期間，所有請求仍需要核准。如需詳細資訊，請參閱下列主題： • 建立變更範本 • 建立變更請求 • 試用 AWS 受管Hello World變更範本	2021 年 7 月 30 日
OpsCenter 操作洞察	OpsCenter 會在您的帳戶中自動分析 OpsItems，並產生洞察。洞察包含的資訊可協助您瞭解您的帳戶中有多少重複的 OpsItems，以及哪些來源正在建立它們。Insights 也提供建議的最佳實務和 Automation Runbook，以協助您解決重複的 OpsItems。如需詳細資訊，請參閱使用操作洞察。	2021 年 7 月 13 日
在 Fleet Manager 中檢視已停止的執行個體	您現在可在 Fleet Manager 主控台中檢視哪些執行個體`running`以及哪些執行個體`stopped`。如需更多詳細資訊，請參閱 AWS Systems ManagerFleet Manager。	2021 年 7 月 12 日

新主題：撰寫 Automation Runbook	新主題撰寫 Automation Runbook 提供如何撰寫自訂 Automation Runbook 內容的指導方針和敘述範例。	2021 年 7 月 8 日
AWS CloudFormation 在 中建立堆疊和範本 Application Manager	Application Manager 可與 CloudFormation 整合，進而協助您佈建和管理應用程式的資源。您可以在 中建立、編輯和刪除 AWS CloudFormation 範本和堆疊Application Manager。 Application Manager也包含範本程式庫，您可以在其中複製、建立和存放範本。 Application Manager和 CloudFormation 會顯示堆疊目前狀態的相同資訊。範本和範本更新會存放在 Systems Manager 中，直到您佈建堆疊為止，此時變更也會顯示在 CloudFormation 中。如需詳細資訊，請參閱在 AWS CloudFormation 中使用 StacksApplication Manager。	2021 年 7 月 8 日
新主題：自動輪換混合執行個體上的 SSM Agent 的私有金鑰	新主題設定私有金鑰自動輪換提供有關如何透過設定 SSM Agent 來自動輪換混合環境私有金鑰從而加強您的安全狀態的說明。	2021 年 6 月 15 日
Session Manager 1.2.205.0 AWS CLI 版的外掛程式	AWS CLI 已發行 的 Session Manager 外掛程式新版本。如需詳細資訊，請參閱 Session Manager 外掛程式最新版本和發行歷史記錄。	2021 年 6 月 10 日

新的 IAM 服務連結的角色	啟用 OpsCenter 操作洞察時，Systems Manager 會建立稱為 `AWSSSMOpsInsightsServiceRolePolicy` 的新的 AWS Identity and Access Management (IAM) 服務連結角色。如需有關此角色的詳細資訊，請參閱使用角色在 Systems Manager OpsCenter 中建立操作洞察 OpsItems：`AWSSSMOpsInsightsServiceRolePolicy`。	2021 年 6 月 9 日
針對 Linux 的新的 Patch Manager 疑難排解內容	新主題在 Linux 中執行 `AWS-RunPatchBaseline` 時出現的錯誤提供在使用 Linux 作業系統修補受管執行個體時可能遇到的數個問題的說明和解決方案。	2021 年 6 月 8 日
改善了不需要指定目標的維護時段任務的支援 (主控台)	您現在可以在主控台中建立維護時段任務，而不必在任務中指定目標 (如果不需要的話)。先前，只有在使用 AWS CLI 或 API 時才能使用此選項。此選項適用於自動化 AWS Lambda、 和 AWS Step Functions 任務類型。例如，如果您建立 Automation 任務，並且在 Automation 文件參數中指定要更新的資源，則不再需要在任務本身中指定目標。如需詳細資訊，請參閱註冊不含目標的維護時段任務、將任務指派給維護時段 (主控台)以及使用維護時段排程自動化。	2021 年 5 月 28 日

重新放置 Automation Runbook 參考	Automation Runbook 參考已移到新的位置。如需詳細資訊，請參閱《Systems Manager Automation Runbook 參考》。	2021 年 5 月 10 日
AWS Systems Manager Incident Manager 啟動	Incident Manager 是一種事件管理主控台，旨在協助使用者減輕影響其 AWS 託管應用程式的事件並從中復原。如需詳細資訊，請參閱「AWS Systems Manager Incident Manager 使用者指南」。	2021 年 5 月 10 日
State Manager 支援 Change Calendar	建立或更新 State Manager 關聯時，您現在可以指定 Change Calendar 名稱或 Amazon Resource Name (ARN)。只有在變更行事曆開啟時，State Manager 才會套用關聯，而不會在關閉時套用關聯。如需詳細資訊，請參閱建立關聯和編輯和建立關聯的新版本。	2021 年 5 月 6 日
複製 Systems Manager 文件	您現在可以使用 Systems Manager 文件主控台，將現有文件的內容複製到您可以修改的新文件中。如需進一步了解，請參閱複製 SSM 文件。	2021 年 5 月 4 日

將 Security Hub 與 Explorer 和 OpsCenter 整合	您現在可以與 整合 Explorer 和 OpsCenter AWS Security Hub。Security Hub 提供 中安全狀態的全方位檢視，AWS 可協助您根據安全產業標準和最佳實務檢查環境。與 Explorer 整合時，可以在 Explorer 儀表板的 Security Hub 小工具中檢視安全問題清單。與 OpsCenter 整合時，可以為 Security Hub 問題清單建立 OpsItems。如需詳細資訊，請參閱從 AWS Security Hub 接收問題Explorer清單，以及AWS Security Hub 從 接收問題清單OpsCenter。	2021 年 4 月 27 日
新主題：文件慣例	我們新增了一個新主題，協助使用者了解 AWS Systems Manager 使用者指南的常見印刷慣例。如需詳細資訊，請參閱文件慣例相關文章。	2021 年 4 月 21 日
更新的主題：關於在 Windows Server 上修補 Microsoft 發行的應用程式	關於在 Windows Server 上由 Microsoft 發行的修補應用程式主題現在澄清，為了讓 Patch Manager 能夠在您的 Windows Server 受管執行個體上修補 Microsoft 發行的應用程式，必須在執行個體上允許 Give me updates for other Microsoft products when I update Windows (當我更新Windows 時，為我提供其他 Microsoft 產品的更新) 的 Windows 更新選項。	2021 年 4 月 12 日

Automation Runbook 參考重新組織	為了協助您查找所需的 Runbook 並更有效率地導覽參考，我們透過相關 AWS 服務重新組織了 Automation Runbook 參考中的內容。若要檢視這些變更，請參閱《Systems Manager Automation Runbook 參考》。	2021 年 4 月 12 日
Patch Manager：產生 .csv 修補程式合規報告	Patch Manager 現在支援為執行個體產生修補程式合規報告，並將報告儲存在您選擇的 S3 儲存貯體中 (格式為 .csv)。然後，使用諸如 Amazon QuickSight 等工具，您可以分析修補程式合規報告資料。您可以為單一執行個體或 AWS 帳戶中的所有執行個體產生修補程式合規報告。您可以依需求產生一次性報告，或設定排程以便自動建立報告。您也可以指定 Amazon Simple Notification Service 主題，在產生報告時提供通知。如需詳細資訊，請參閱產生 CSV 修補程式合規報告。	2021 年 4 月 9 日
刪除 Parameter Store 參數標籤	您現在可以使用 Systems Manager 主控台或 AWS CLI來刪除 Parameter Store 參數標籤。如需詳細資訊，請參閱使用參數標籤。	2021 年 4 月 6 日

使用「立即修補」時，排程執行個體重新啟動	Patch Manager 現在支援在使用「立即修補」功能安裝修補程式之後，排定您的執行個體重新啟動的時間。這是對現有選項的補充，只有在需要完成修補程式安裝或在修補操作後跳過所有重新啟動時，才會重新啟動執行個體。如需相關資訊，請參閱隨需修補執行個體。	2021 年 4 月 1 日
新主題：發現公有參數	Parameter Store 現在可以使用 AWS CLI 或 Systems Manager 主控台找到公有參數。如需詳細資訊，請參閱查找公有參數。	2021 年 4 月 1 日
「立即修補」更新：將日誌存放在 S3 中並執行生命週期掛鉤	當您執行 Patch Manager 立即修補操作時，您可以選擇在其中自動存放修補程式日誌的 S3 儲存貯體。此外，您可以選擇在操作期間的三個點執行 Systems Manager 命令文件 (SSM 文件) 作為生命週期掛鉤：安裝前、安裝後以及退出時。如需詳細資訊，請參閱隨需修補執行個體。	2021 年 3 月 31 日
Systems Manager 現在會報告其 AWS 受管政策的變更	從 2021 年 3 月 24 日開始，受管政策的變更會在 Systems ManagerAWS 受管政策的主題更新中報告。列出的第一個變更是新增 Explorer 工具的支援，以報告 OpsData 以及 OpsItems來自多個帳戶和區域的 。	2021 年 3 月 24 日

Explorer 會根據 中的帳戶，自動允許資源資料同步的所有 OpsData 來源 AWS Organizations	當您建立資源資料同步時，如果您選擇其中一個 AWS Organizations 選項，Systems Manager 會自動為組織（或在選取的組織單位中）中的所有 AWS 帳戶 中選取 AWS 區域 的所有 OpsData 來源。例如，這表示即使您尚未在 Explorer中允許 AWS 區域，如果您為資源資料同步選取 AWS Organizations 選項，則 Systems Manager 會自動從該區域收集 OpsData。如需詳細資訊，請參閱關於多個帳戶和區域資源資料同步。	2021 年 3 月 24 日
Systems Manager Automation 為您的 Runbook 提供新的系統變數	使用新的`global:AWS_PARTITION` 系統變數，您可以在撰寫 Runbook 時指定資源所在的 AWS 分割區。如需系統資料表的詳細資訊，請參閱 Automation 系統變數。	2021 年 3 月 18 日
允許 Change Manager 變更請求的多層級核准	建立 Change Manager 變更範本時，您現在可以要求多個層級的核准者授予許可，以便執行變更請求。例如，您可能需要技術檢閱者先核准從變更範本中建立的變更請求，然後再請求一位或多位經理的第二層核准。如需詳細資訊，請參閱建立變更範例。	2021 年 3 月 4 日

Patch Manager 現在支援 Oracle Linux 8.x	您現在可以使用 Patch Manager 透過版本 8.3 來修補 Oracle Linux 8.x 執行個體。如需詳細資訊，請參閱下列主題： • 如何選取安全性修補程式 • 如何安裝修補程式 • 修補基準規則在 Oracle Linux 上的運作方式	2021 年 3 月 1 日
OpsCenter 顯示所選資源的其他 OpsItems	為了協助您調查問題並提供問題的內容，您可以檢視特定 AWS 資源OpsItems的 清單。清單會顯示每個 OpsItem 的狀態、嚴重性和標題。該清單還包括每個 OpsItem 的深層連結。如需詳細資訊，請參閱檢視特定資源的其他 OpsItems。	2021 年 3 月 1 日
在執行時間定義修補程式偏好設定	使用基準覆寫功能，您可以在執行時間定義修補偏好設定。如需詳細資訊，請參閱使用 BaselineOverride 參數。	2021 年 2 月 25 日
新的 Systems Manager 文件類型	AWS CloudFormation 範本現在可以儲存為 Systems Manager 文件。將 CloudFormation 範本存放為 Systems Manager 文件，可讓您從 Systems Manager 文件功能中受益，例如版本控制、比較版本內容以及與帳戶共用。如需詳細資訊，請參閱 AWS Systems Manager 文件。	2021 年 2 月 9 日

使用選用掛鉤來修補執行個體	新的 SSM 文件 AWS-RunPatchBaselineWithHooks 提供掛鉤，用於您在執行個體修補週期期間的三個點執行 SSM 文件。如需有關 AWS-RunPatchBaselineWithHooks 的資訊，請參閱關於 AWS-RunPatchBaselineWithHooks SSM 文件。如需使用全部三個掛鉤之修補操作的範例演練，請參閱演練：更新應用程式相依性、修補執行個體以及執行應用程式特定的運作狀態檢查。	2021 年 2 月 2 日
新主題：使用硬體指紋驗證內部部署伺服器和虛擬機器	透過使用已運算的指紋，SSM Agent 會驗證內部部署伺服器、虛擬機器和您註冊服務的虛擬機器的身分。指紋是不透明字串，存放在保存庫中，代理程式會將其傳遞給某些 Systems Manager API。如需硬體指紋的相關資訊，以及設定相似度臨界值協助進行機器驗證的說明，請參閱使用硬體指紋驗證內部部署伺服器和虛擬機器。	2021 年 1 月 25 日
新主題：SSM Agent 技術參考	主題SSM Agent技術參考匯集資訊，協助您實作 AWS Systems Manager SSM Agent 和了解代理程式的運作方式。本主題包含全新的 區段，SSM Agent由 滾動更新 AWS 區域。	2021 年 1 月 21 日

Windows Server 2008 上的 SSM Agent	從 2020 年 1 月 14 日起，Microsoft 不再支援 Windows Server 2008 的功能或安全性更新。Windows Server 2008 AMIs 包括 SSM Agent，但不再對此作業系統更新代理程式。	2021 年 1 月 5 日
改善對不需要指定目標的維護時段任務的支援（僅限AWS CLI 和 API)	如果不需要維護時段任務（僅限AWS CLI 和 API)，您現在可以在任務中建立維護時段任務，而無需指定目標。這適用於自動化 AWS Lambda 和 AWS Step Functions 任務類型。例如，如果您建立 Automation 任務，並且在 Automation Runbook 參數中指定要更新的資源，則不再需要在任務本身中指定目標。如需詳細資訊，請參閱註冊不含目標的維護時段任務和使用維護時段排程自動化。	2020 年 12 月 23 日
新的 Automation 功能	新的共用屬性已新增至 Systems Manager Automation Runbook。`onCancel` 屬性可讓您指定在使用者取消自動化時，自動化應該移至哪個步驟。如需詳細資訊，請參閱所有動作共用的屬性。	2020 年 12 月 21 日
新主題：透過 IAM 使用關聯	新主題已新增至 Systems Manager State Manager 章節，它說明使用 IAM 建立關聯的最佳實務。如需相關資訊，請參閱以 IAM 使用關聯。	2020 年 12 月 18 日

State Manager 現在支援多區域和多帳戶	現在可以使用多個區域或帳戶建立或更新關聯。如需詳細資訊，請參閱建立關聯。	2020 年 12 月 15 日
新工具： Fleet Manager	Fleet Manager是 中的工具 AWS Systems Manager，是一種統一的使用者介面 (UI) 體驗，可協助您遠端管理在 AWS 或內部部署上執行的伺服器機群。利用 Fleet Manager，您可以從單一主控台檢視整個伺服器機群的運作狀態和效能狀態。您也可以從個別執行個體收集資料，進而從主控台執行常見的故障診斷和管理任務。如需相關資訊，請參閱 AWS Systems ManagerFleet Manager。	2020 年 12 月 15 日
新工具： Change Manager	Amazon Web Services 已發行 Change Manager，它是一個企業變更管理架構，用於請求、核准、實作和報告應用程式組態和基礎設施的操作變更。從單一委派管理員帳戶，如果您使用 AWS Organizations，您可以管理多個 AWS 帳戶 中多個 的變更 AWS 區域。或使用本機帳戶，您可以管理單一 AWS 帳戶的變更。使用 Change Manager 管理 AWS 資源和內部部署資源的變更。如需相關資訊，請參閱 AWS Systems ManagerChange Manager。	2020 年 12 月 15 日

新工具： Application Manager	Application Manager 可協助您調查和修復應用程式內容中 AWS 資源的問題。 會將多個 AWS 服務 和 Systems Manager 工具的操作資訊Application Manager彙總到單一 AWS Management Console。如需相關資訊，請參閱 AWS Systems ManagerApplication Manager。	2020 年 12 月 15 日

AWS Systems Manager 支援 的 Amazon EC2 執行個體 macOS	同時發行 macOS 執行個體的 Amazon Elastic Compute Cloud (Amazon EC2) 支援，Systems Manager 現在支援 macOS 的 EC2 執行個體上的許多操作。支援的版本包括 macOS 10.14.x (Mojave) 和 10.15.x (Catalina)。如需詳細資訊，請參閱下列主題。 • 如需對適用於 macOS 之 EC2 執行個體安裝 SSM Agent 的詳細資訊，請參閱在適用於 macOS 的 EC2 執行個體上安裝和設定 SSM Agent • 如需修補適用於 macOS 之 EC2 執行個體的相關資訊，請參閱如何安裝修補程式，以及建立自訂修補基準 (macOS)。 • 如需有關支援 macOS EC2 執行個體的一般資訊，請參閱《Amazon EC2 使用者指南》中的 Amazon EC2 Mac instances。	2020 年 11 月 30 日
維護時段虛擬參數：支援的新資源類型 {{TARGET_ID}} 和 {{RESOURCE_ID}}	現在有一個額外的資源類型可用於虛擬參數 {{TARGET_ID}} 和 {{RESOURCE_ID}} 。您現在可以使用具有這些虛擬參數的資源類型 AWS::RDS::DBCluster 。如需有關維護時段虛擬參數的詳細資訊，請參閱在註冊維護時段任務時使用虛擬參數。	2020 年 11 月 27 日

Session Manager 1.2.30.0 AWS CLI 版的外掛程式	AWS CLI 已發行 的 Session Manager 外掛程式新版本。如需詳細資訊，請參閱 Session Manager 外掛程式最新版本和發行歷史記錄。	2020 年 11 月 24 日
新主題：比較 SSM 文件版本	您現在可以在 Systems Manager 文件主控台中比較 SSM 文件版本之間的內容差異。如需詳細資訊，請參閱比較 SSM 文件版本。	2020 年 11 月 24 日
Systems Manager 現在支援 VPC 端點政策	您現在可以為 Systems Manager 的 VPC 介面端點建立政策。如需詳細資訊，請參閱建立介面 VPC 端點政策。	2020 年 11 月 18 日
新主題：指定閒置工作階段逾時值	您現在可以指定在 Session Manager 結束工作階段前使用者可處於非作用中狀態的時間長度。如需詳細資訊，請參閱指定閒置工作階段逾時值。	2020 年 11 月 18 日
新的 Session Manager 日誌功能	您現在可以將 JSON 格式的工作階段資料日誌的持續串流傳送到 Amazon CloudWatch Logs。如需詳細資訊，請參閱使用 Amazon CloudWatch Logs 串流工作階段資料。	2020 年 11 月 18 日
新主題：驗證 SSM Agent 的簽章	您現在可以驗證 Linux 執行個體上 SSM Agent 的安裝程式套件的加密簽章。如需詳細資訊，請參閱 SSM 文件結構描述和功能。	2020 年 11 月 17 日

新主題：了解自動化狀態	Systems Manager Automation 章節新增了一個新主題，它說明動作和自動化的狀態。如需詳細資訊，請參閱了解自動化狀態。	2020 年 11 月 17 日
aws:downloadContent 外掛程式的新來源類型	現在支援 Git 和 HTTP 作為 aws:downloadContent 外掛程式的來源類型。如需詳細資訊，請參閱aws:downloadContent 。	2020 年 11 月 17 日
新的 Systems Manager 文件 (SSM 文件) 結構描述功能	在具有結構描述 2.2 版或更新版本的 SSM 文件中，precondition 參數現在支援參考文件的輸入參數。如需詳細資訊，請參閱 SSM 文件結構描述和功能。	2020 年 11 月 17 日
中的新資料來源Explorer：AWS Config	Explorer 現在會顯示 AWS Config 合規的相關資訊，包括合規和不合規 AWS Config 規則的整體摘要、合規和不合規資源的數量，以及每個資源的特定詳細資訊（當您深入了解不合規規則或資源時）。如需詳細資訊，請參閱編輯 Systems Manager Explorer 資料來源。	2020 年 11 月 11 日
新主題：執行具有關聯的 Auto Scaling 群組	新章節已新增至 State Manager，它說明建立關聯以執行 Auto Scaling 群組的最佳實務。如需詳細資訊，請參閱執行具有關聯的 Auto Scaling 群組。	2020 年 11 月 10 日

Quick Setup 現在支援以資源群組為目標	Quick Setup 現在支援選擇資源群組作為本機設定類型的目標。如需詳細資訊，請參閱選擇 Quick Setup 的目標。	2020 年 11 月 5 日
Patch Manager 新增對 Debian Server 10 LTS、Oracle Linux 7.9 LTS 和 Ubuntu Server 20.10 STR 的支援	您現在可以使用 Patch Manager 來修補 Debian Server 10 LTS、Oracle Linux 7.9 LTS 和 Ubuntu Server 20.10 STR 執行個體。如需詳細資訊，請參閱下列主題： • Patch Manager 先決條件 • 如何選取安全性修補程式 • 如何安裝修補程式 • 修補基準規則在 Debian Server 上的運作方式 • 修補基準規則在 Oracle Linux 上的運作方式 • 修補基準規則在 Ubuntu Server 上的運作方式	2020 年 11 月 4 日

的新 EventBridge 支援 AWS Systems ManagerChange Calendar	Amazon EventBridge 現在事件規則中提供 Change Calendar 事件支援。當行事曆的狀態變更時，EventBridge 可以啟動您定義的 EventBridge 規則的目標動作。如需有關使用 EventBridge 和 Systems Manager 事件的資訊，請參閱下列主題。 • 為 Systems Manager 事件設定 EventBridge • 參考：Systems Manager 的 Amazon EventBridge 事件模式和類型	2020 年 11 月 4 日
設定 CloudWatch 以便從警示中建立 OpsItems	當警示進入 ALARM 狀態時，可設定 Amazon CloudWatch 以自動在 Systems Manager OpsCenter 中建立 OpsItem。這樣做可讓您從單一主控台快速診斷和修復 AWS 資源的問題。如需詳細資訊，請參閱設定 CloudWatch 以從警示中建立 OpsItems。	2020 年 11 月 4 日

支援 Ubuntu Server 20.10	AWS Systems Manager 現在支援 Ubuntu Server 20.10 短期版本 (STR)。如需詳細資訊，請參閱下列主題： • 支援的作業系統 • 在混合環境 (Linux) 中安裝 SSM Agent • 手動在 Ubuntu Server 執行個體上安裝 SSM Agent • 檢查 SSM Agent 狀態並啟動代理程式	2020 年 10 月 22 日
新主題：允許可設定的 shell 描述檔	現在可使用 Session Manager 允許可設定的 shell 描述檔。透過允許可設定的 shell 描述檔，您可自訂工作階段內的偏好設定，例如 shell 偏好設定、環境變數、工作目錄，以及在工作階段啟動時執行的多個命令。如需詳細資訊，請參閱允許可設定的 shell 描述檔。	2020 年 10 月 21 日
修補程式合規結果現在會報告哪些修補程式解決了哪些 CVE	對於大多數支援的 Linux 系統，當您檢視受管執行個體的修補程式合規結果時，您可檢視的詳細資訊現在會報告可用的修補程式解決了哪些常見漏洞和暴露 (CVE) 公告問題。此資訊可協助您判斷安裝遺漏或失效修補程式的迫切性。如需詳細資訊有，請參檢視修補程式合規結果。	2020 年 10 月 20 日

Linux 修補程式中繼資料的擴充支援	您現在可以在 Patch Manager 中檢視有關可用 Linux 修補程式的許多詳細資訊。您可以選擇檢視修補程式資料，例如架構、epoch、版本、CVE ID、Advisory ID、Bugzilla ID、儲存庫等。此外，DescribeAvailablePatches API 操作已更新，可支援 Linux 作業系統以及根據這些最新可用的修補程式中繼資料類型進行篩選。如需詳細資訊，請參閱下列主題： • 檢視可用修補程式 • 《AWS Systems Manager API 參考》中的 DescribeAvailablePatches 和修補部分 • describe-available-patches 在 AWS Systems Manager AWS CLI 命令參考的 區段中	2020 年 10 月 16 日
Session Manager 1.2.7.0 AWS CLI 版的外掛程式	AWS CLI 已發行 的 Session Manager 外掛程式新版本。如需詳細資訊，請參閱 Session Manager 外掛程式最新版本和發行歷史記錄。	2020 年 10 月 15 日
新主題：工作階段文件結構描述	新主題工作階段文件結構描述說明工作階段文件的結構描述元素。此資訊可協助您建立自訂工作階段文件，您可以在其中指定與 Session Manager 搭配使用的工作階段類型的偏好設定。	2020 年 10 月 15 日

新主題：SSM 文件的任意文字搜尋	Systems Manager 文件頁面中的搜尋方塊現在支援任意文字搜尋。任意文字搜尋會根據每個 SSM 文件中的文件名稱來比較搜尋詞語或您輸入的詞語。如需詳細資訊，請參閱使用任意文字搜尋。	2020 年 10 月 15 日
新主題：對 Amazon EC2 受管執行個體可用性進行故障診斷	新主題對 Amazon EC2 受管執行個體可用性進行故障診斷可協助您調查為什麼您已確認正在執行的 Amazon EC2 執行個體在 Systems Manager 的可用受管執行個體清單中無法使用。	2020 年 10 月 6 日
Parameter Store 章節重組	為了協助您更有效率地找到所需資訊，我們重新組織了 AWS Systems Manager 使用者指南的 Parameter Store 章節中的內容。現在會在設定 Parameter Store 和使用 Parameter Store 章節中組織大多數內容。此外，主題 AWS Systems ManagerParameter Store 已擴展以包括以下部分： • Parameter Store 如何為我的組織帶來益處？ • 誰應該使用Parameter Store？ • Parameter Store 有哪些功能？ • 什麼是參數？	2020 年 10 月 1 日

新的修補程式合規相關主題	已新增下列主題，以協助您識別不符合修補程式規範的受管執行個體、了解不同類型的修補程式合規掃描、以及採取適當的措施使您的執行個符合規。 • 識別不合規的執行個體 • 修補不合規的執行個體 • 檢視修補程式合規結果	2020 年 9 月 24 日
SSM Agent 3.0 版	Systems Manager 推出了 SSM Agent 的新版本。	2020 年 9 月 21 日
新主題和已更新主題：Amazon EventBridge 取代 CloudWatch Events 進行活動管理	CloudWatch Events 和 EventBridge 是相同的基礎服務和 API，但 EventBridge 提供更多功能，現在是管理事件的首選方式 AWS。(您在 CloudWatch 或 EventBridge 中所做的變更將會顯現在每個主控台中。) AWS Systems Manager 使用者指南的 CloudWatch Events 和現有程序的參考已更新，以反映 EventBridge 支援。此外，已新增以下新主題。 • 監控 Systems Manager 事件 • 為 Systems Manager 事件設定 EventBridge • Systems Manager 目標類型範例 • 參考：Systems Manager 的 Amazon EventBridge 事件模式和類型	2020 年 9 月 18 日

整合 AWS Security Hub 和 Patch Manager	您現在可以Patch Manager 與 整合 AWS Security Hub。Security Hub 提供 中安全狀態的完整檢視， AWS 並協助您根據安全產業標準和最佳實務檢查環境。與 Patch Manager 整合時，Security Hub 會從安全角度監控機群的修補狀態。如需詳細資訊，請參閱Patch Manager與 整合 AWS Security Hub。	2020 年 9 月 17 日
維護時段虛擬參數：支援的新資源類型 {{TARGET_ID}} 和 {{RESOURCE_ID}}	註冊維護時段任務時，您可以使用 `--task-invocation-parameters` 選項來指定四種任務類型特有的參數。您也可以使用虛擬參數語法來參考特定的值，例如 {{TARGET_ID}} 和 {{RESOURCE_ID}} 。維護時段任務執行時，它會傳遞正確的值，而不是虛擬參數預留位置。現在有兩個額外的資源類型可用於虛擬參數 {{TARGET_ID}} 和 {{RESOURCE_ID}} 。您現在可以使用具有這些虛擬參數的資源類型 `AWS::RDS::DBInstance` 和 `AWS::SSM::ManagedInstance` 。如需有關維護時段虛擬參數的詳細資訊，請參閱在註冊維護時段任務時使用虛擬參數。	2020 年 9 月 14 日

使用新的立即修補選項，根據需求修補執行個體	您現在可以隨時使用 Systems Manager 主控台來修補執行個體，或掃描遺失的修補程式。您不需要建立或修改排程，也可以指定完整的修補組態選項，以滿足即時的修補需求。您只需指定是否要掃描或安裝修補程式，並識別操作的目標執行個體。Patch Manager 會自動套用執行個體類型的目前預設修補基準，並針對一次修補多少個執行個體套用最佳實務選項，以及在操作失敗之前允許多少個錯誤。如需詳細資訊，請參閱隨需修補執行個體。	2020 年 9 月 9 日
新主題：檢查 SSM Agent 狀態並啟動代理程式	新主題檢查 SSM Agent 狀態並啟動代理程式提供命令來檢查 SSM Agent 是否在每個支援的作業系統上執行。它也會提供用於啟動代理程式 (如果代理程式未執行) 的命令。	2020 年 9 月 7 日
Patch Manager 現在支援 Ubuntu Server 20.04 LTS	您現在可以使用 Patch Manager 來修補 Ubuntu Server 20.04 LTS 執行個體。如需詳細資訊，請參閱下列主題： • 如何選取安全性修補程式 • 如何安裝修補程式 • 修補基準規則在 Ubuntu Server 上的運作方式	2020 年 8 月 31 日

使用案例與最佳實務新主題	我們新增了一個新主題，協助使用者快速了解 Maintenance Windows 和 State Manager 之間的差異。如需詳細資訊，請參閱在 State Manager 和 Maintenance Windows 之間選擇。	2020 年 8 月 28 日
新的 OpsCenter 功能	OpsCenter 包含新功能，可協助您快速找到並執行 Automation Runbook 來補救問題。如需詳細資訊，請參閱 OpsCenter 中的 Automation Runbook 功能。	2020 年 8 月 19 日
Explorer： AWS 支援 cases 中的新資料來源	Explorer 現在會顯示 支援 案例的相關資訊。您必須設定企業或企業帳戶 支援。如需詳細資訊，請參閱編輯 Systems Manager Explorer 資料來源。	2020 年 8 月 13 日
Distributor 現在提供 Trend Micro 的第三方套件。	Distributor 現在包括 Trend Micro 的第三方套件。您可以使用 Distributor，在受管執行個體上安裝 Trend Micro Cloud One 代理程式。Trend Micro Cloud One 可協助您保護雲端中的工作負載。如需詳細資訊，請參閱AWSDistributor。	2020 年 8 月 12 日

aws:configurePackage 文件外掛程式現在包含 additionalArguments 參數。	Systems Manager 命令文件外掛程式 aws:configurePackage 現在支援為您的指令碼提供其他參數 (安裝、解除安裝和更新)，以及新的 additionalArguments 參數。如需詳細資訊，請參閱 aws:configurePackage 主題。	2020 年 8 月 11 日
AppConfig 內容移至單獨的使用者指南	AWS AppConfig 的相關資訊已移至單獨的使用者指南。如需詳細資訊，請參閱什麼是 AWSAppConfig？ AppConfig 也有單獨的文件登陸頁面，其中包含使用者指南、AppConfigAPI 參考和新AppConfig研討會的連結。	2020 年 8 月 3 日
Quick Setup 現在支援 AWS Organizations	Quick Setup 現在支援 ， AWS Organizations 可讓您跨多個帳戶和區域快速設定所需的安全角色和常用的 Systems Manager 工具。如需更多詳細資訊，請參閱 AWS Systems ManagerQuick Setup。	2020 年 7 月 23 日
Explorer 中的新資料來源：關聯合規	Explorer 現在會顯示 State Manager 的關聯合規資料。如需詳細資訊，請參閱編輯 Systems Manager Explorer 資料來源。	2020 年 7 月 23 日

用於開啟和關閉 Kernel Live Patching 的新的 Systems Manager 命令文件	當您想要在 Amazon Linux 2 上執行個體上開啟或關閉 Kernel Live Patching 時，文件 `AWS-ConfigureKernelLivePatching` 現在可以與 Run Command 搭配使用。本文件取代了針對這些任務建立您自己的自訂命令文件的需求。如需詳細資訊，請參閱在 Amazon Linux 2 執行個體上使用核心即時修補。	2020 年 7 月 22 日
已更新 Automation 配額	已更新 Automation 的服務配額，包括速率控制自動化的單獨佇列。如需詳細資訊，請參閱 AWS Systems Manager Automation。	2020 年 7 月 20 日
使用主控台指定維護時段的排程偏移天數	使用 Systems Manager 主控台，指定在執行維護時段之前，在 CRON 運算式所指定的日期和時間之後等待的天數。（先前，此選項僅適用於使用 AWS SDK 或命令列工具時。）例如，如果您的 CRON 運算式將維護時段排程在每個月的第三個星期二晚上 11:30 執行 - `cron(0 30 23 ? * TUE#3 *)` - 而您指定 2 的排程偏移，則該時段會等到兩天之後在 11:30 PM 執行。如需詳細資訊，請參閱 Systems Manager 的 Cron 和 rate 運算式以及指定維護時段的排程偏移天數。	2020 年 7 月 17 日

使用 Run Command 更新 PowerShell	為了協助您將 PowerShell 更新至 2012 和 2012 R2 執行個體上的 Windows Server 5.1 版，我們已將逐步解說新增至 AWS Systems Manager 使用者指南。如需詳細資訊，請參閱使用 Run Command 更新 PowerShell。	2020 年 6 月 30 日
Patch Manager 現在支援 CentOS 8.0 和 8.1	您現在可以使用 Patch Manager 修補 CentOS 8.0 和 8.1 執行個體。如需詳細資訊，請參閱下列主題： • 如何選取安全性修補程式 • 如何安裝修補程式 • 修補基準規則在 CentOS 上的運作方式 • 手動在 CentOS 執行個體上安裝 SSM Agent • 如何在混合 Linux 節點上安裝 SSM Agent	2020 年 6 月 27 日

AppConfig 與 整合 AWS CodePipeline	AppConfig 是 AWS CodePipeline (CodePipeline) 的整合部署動作。CodePipeline 是一種全受管持續交付服務，可協助您自動化發行管道，以快速且可靠地更新應用程式和基礎設施。根據您定義的發行模型，CodePipeline 可以自動在每次程式碼變更時建置、測試和部署程式碼。AppConfig 與 CodePipeline 的整合具有以下優勢。如需詳細資訊，請參閱 AppConfig 與 CodePipeline 整合。 • 使用 CodePipeline 管理協同運作的客戶現在可以輕鬆地將組態變更部署至其應用程式，而不需要部署整個程式碼庫。 • 想要使用 AppConfig 來管理組態部署但由於 AppConfig 不支援其目前代碼或組態存放區而受到限制的客戶現在可使用其他選項。CodePipeline AWS CodeCommit 支援 GitHub、 和 BitBucket （舉例來說）。	2020 年 6 月 25 日
新增章節：產品和服務整合	為了協助您了解 Systems Manager 如何與 AWS 服務 和其他產品和服務整合， AWS Systems Manager 使用者指南已新增一章。如需詳細資訊，請參閱與 Systems Manager 整合的產品和服務。	2020 年 6 月 23 日

Automation 章節重新組織	為了協助您找到所需內容，我們重新組織了 AWS Systems Manager 使用者指南的 Automation 章節的主題。例如，Automation 動作和 Automation Runbook 參考現在是本章中的最重要部分。如需詳細資訊，請參閱 AWS Systems Manager Automation。	2020 年 6 月 23 日
指定維護時段的排程偏移天數	使用命令列工具或 AWS SDK，您現在可以指定在執行維護時段之前，在 CRON 表達式指定的日期和時間之後等待的天數。例如，如果您的 CRON 運算式將維護時段排程在每個月的第三個星期二晚上 11:30 執行 - `cron(0 30 23 ? * TUE#3 *)` - 而您指定 2 的排程偏移，則該時段會等到兩天之後在 11:30 PM 執行。如需詳細資訊，請參閱 Systems Manager 的 Cron 和 rate 運算式以及指定維護時段的排程偏移天數。	2020 年 6 月 19 日

Patch Manager 支援在 Amazon Linux 2 執行個體上進行核心即時修補	Amazon Linux 2 的 Kernel Live Patching 可讓您將安全性漏洞和重大錯誤修補程式套用至執行中的 Linux 核心，而不需要重新啟動或中斷執行中的應用程式。您現在可以啟用此功能，並使用 Patch Manager 套用核心即時修補程式。如需相關資訊，請參閱在 Amazon Linux 2 執行個體上使用核心即時修補。	2020 年 6 月 16 日
Patch Manager 增加 Oracle Linux 版本支援	過去，Patch Manager 僅支援 Oracle Linux 的 7.6 版。如 Patch Manager 先決條件中所列，支援現在涵蓋 7.5-7.8 版。	2020 年 6 月 16 日
在修補操作中使用 InstallOverrideList 參數的範例案例	新主題：使用 InstallOverrideList 參數的範例案例說明使用 AWS-RunPatchBaseline 文件中的 InstallOverrideList 參數，依不同的維護時段排程將不同類型的修補程式套用至目標群組的策略，同時繼續使用單一修補基準。	2020 年 6 月 11 日
預先定義 AppConfig 的部署策略	AppConfig 現在提供預先定義的部署策略。如需詳細資訊，請參閱建立部署策略。	2020 年 6 月 10 日

Patch Manager 現在支援 Red Hat Enterprise Linux (RHEL) 7.8-8.2	您現在可以使用 Patch Manager 修補 RHEL 7.8-8.2 執行個體。如需詳細資訊，請參閱下列主題： • 如何選取安全性修補程式 • 如何安裝修補程式 • 修補基準規則在 RHEL 上的運作方式 • 手動在 Red Hat Enterprise Linux 執行個體上安裝 SSM Agent • 如何在混合 Linux 節點上安裝 SSM Agent	2020 年 6 月 9 日
Explorer 支援委派的管理員	如果您使用資源Explorer資料同步來彙總來自多個 AWS 區域 和 AWS 帳戶 的資料 AWS Organizations，建議您為 設定委派管理員Explorer。委派管理員透過限制 Explorer 管理員數目，這些管理員可建立或刪除多帳戶和只同步至一個人的區域資源資料，藉此改善 Explorer 安全性。您不再需要登入 AWS Organizations 管理帳戶，即可管理 Explorer 中的資源資料同步。如需詳細資訊，請參閱設定委派的管理員。	2020 年 6 月 3 日

僅在下一個指定的 Cron 間隔套用 State Manager 關聯	如果您不希望在建立 State Manager 關聯之後立即執行關聯，可在 Systems Manager 主控台中選擇 Apply association only at the next specified Cron interval (僅在下一個指定的 Cron 間隔套用關聯) 選項。如需詳細資訊，請參閱建立關聯。	2020 年 6 月 3 日
中的新資料來源Explorer：AWS Compute Optimizer	Explorer 現在會顯示來自 的資料 AWS Compute Optimizer。這包括 Under provisioned (佈建不足) 和 Over provisioned (過度佈建) 的 EC2 執行個體計數、最佳化問題清單、隨需定價詳情，以及執行個體類型和價格的建議。如需詳細資訊，請參閱設定相關服務 AWS Compute Optimizer 中的設定詳細資訊。 https://docs.aws.amazon.com/systems-manager/latest/userguide/Explorer-setup-related-services.html	2020 年 5 月 26 日
使用 Patch Manager 安裝 Windows Service Pack 和 Linux 次要版本升級	新主題教學課程：建立修補基準用於安裝 Windows Service Pack (主控台) 示範如何建立修補基準，專門用於安裝 Windows Service Pack。主題建立自訂修補基準 (Linux) 已更新，提供在修補基準中包含 Linux 作業系統的次要版本升級的相關資訊。	2020 年 5 月 21 日

Parameter Store 章節重組	有關配置或設定 Parameter Store 操作選項的所有主題都已合併到 Parameter Store 設定一節。這包括管理參數層和增加 Parameter Store 輸送量主題，這些主題已從本章其他部分重新調動。	2020 年 5 月 18 日
針對建立與 Systems Manager API 操作互動的日期和時間字串的新主題。	新主題為 Systems Manager 建立格式化的日期和時間字串說明如何建立格式化的日期和時間字串，以便與 Systems Manager API 操作互動。	2020 年 5 月 13 日
關於加密 SecureString 參數的許可	新主題使用 IAM 政策限制對 Systems Manager 參數的存取，說明使用 AWS KMS key 和使用 AWS 受管金鑰 提供的加密`SecureString` 參數之間的差異 AWS。	2020 年 5 月 13 日
Patch Manager 現在支援 Debian Server 和 Oracle Linux 7.6 作業系統	您現在可以使用 Patch Manager 來修補 Debian Server 和 Oracle Linux 執行個體。Patch Manager 支援修補 Debian Server 8.x、9.x 和 Oracle Linux 7.6 版。如需詳細資訊，請參閱下列主題： • 如何選取安全性修補程式 • 如何安裝修補程式 • 修補基準規則在 Debian Server 上的運作方式 • 修補基準規則在 Oracle Linux 上的運作方式	2020 年 5 月 7 日

建立以目標為目標的State Manager關聯 AWS Resource Groups	除了將 AWS 帳戶中的標籤、個別執行個體和所有執行個體設為目標之外，您現在還可以建立以 AWS Resource Groups 中的執行個體為目標的 State Manager 關聯。如需詳細資訊，請參閱關於 State Manager 關聯中的目標和速率控制	2020 年 5 月 7 日
Parameter Store 中的新 `aws:ec2:image` 資料類型以驗證 AMI ID	建立 `String` 參數時，您現在可以將資料類型指定為 `aws:ec2:image` ，以確保輸入的參數值為有效的 Amazon Machine Image (AMI) ID 格式。支援 AMI ID 格式可讓您避免每次想要在程序中使用的 AMI 發生變更時，都使用新 ID 來更新所有指令碼和範本。您可以使用資料類型 `aws:ec2:image` 建立參數，並為其數值輸入 AMI 的 ID。這是您目前想要建立新執行個體的 AMI。然後，您可以在範本、命令中參考此參數。當您準備好使用不同的 AMI 時，請更新參數值。Parameter Store 會驗證新的 AMI ID，而且您不需要更新指令碼和範本。如需詳細資訊，請參閱 Amazon Machine Image ID 的原生參數支援。	2020 年 5 月 5 日

管理 Run Command 命令中的結束程式碼	Run Command 可讓您定義在指令碼中處理結束程式碼的方式。根據預設，在指令碼中執行的最後一個命令的結束程式碼會報告為整個指令碼的結束程式碼。但是，如果有任何命令在最後一個命令之前失敗，您可以使用下列方法包含 shell 條件陳述式來結束程式碼。如需範例，請參閱新主題管理 Run Command 命令中的結束程式碼。	2020 年 5 月 5 日
針對可用區域和本機區域發行的新公有參數	為了以程式設計方式提供 AWS 可用區域和本機區域的相關資訊，已發行公有參數。這些是 AWS 服務 和 的現有全域基礎設施公有參數之外的 AWS 區域。如需詳細資訊，請參閱呼叫 的公有參數 AWS 服務、區域、端點、可用區域、本機區域和 Wavelength 區域。	2020 年 5 月 4 日
中的新資料來源Explorer：AWS Trusted Advisor	Explorer 現在會顯示來自 的資料 AWS Trusted Advisor。這包括最佳實務檢查的狀態和以下方面的建議：成本最佳化、安全性、容錯能力、效能和 Service Quotas。如需詳細資訊，請參閱設定相關服務 Trusted Advisor 中的設定詳細資訊。 https://docs.aws.amazon.com/systems-manager/latest/userguide/Explorer-setup-related-services.html	2020 年 5 月 4 日

建立執行 Chef 配方的 State Manager 關聯	您可以使用 AWS-ApplyChefRecipes 文件來建立執行 Chef 碼元書和配方的 State Manager 關聯。本文件提供執行 Chef 配方的下列優點： • 支援多個 Chef 版本 (Chef 11 到 Chef 14)。 • 自動在目標執行個體上安裝 Chef 用戶端軟體。 • 在目標執行個體上選擇性執行 Systems Manager 合規檢查，並將合規檢查的結果存放在 S3 儲存貯體中。 • 在單一文件執行中執行多個食譜和配方。 • 選用地在 why-run 模式下執行配方，以顯示哪些配方會對目標執行個體進行變更，而無須實際進行變更。 • 選用地將自訂 JSON 屬性套用到 chef-client 執行。 如需詳細資訊，請參閱建立執行 Chef 配方的關聯	2020 年 3 月 19 日

將庫存資料從多個 同步 AWS 帳戶 到中央 Amazon S3 儲存貯體	您可以將 Systems Manager 庫存資料從多個 同步 AWS 帳戶 到中央 S3 儲存貯體。帳戶必須在其中定義 AWS Organizations。如需詳細資訊，請參閱為 AWS Organizations中定義的多個帳戶建立清查資源資料同步。	2020 年 3 月 16 日
在 Amazon Simple Storage Service (Amazon S3) 中存放 AppConfig 組態	先前，AppConfig 只支援存放在 Systems Manager (SSM) 文件或 Parameter Store 參數中的應用程式組態。除了這些選項之外，AppConfig 現在還支援將組態存放在 Amazon Simple Storage Service (Amazon S3) 中。如需詳細資訊，請參閱關於 Amazon Simple Storage Service (Amazon S3) 中存放的組態。	2020 年 3 月 13 日
SSM Agent 預設安裝在 Amazon ECS 最佳化的 AMIs 上	SSM Agent 現在預設會安裝在 Amazon ECS 最佳化的 AMIs 上。如需詳細資訊，請參閱使用 SSM Agent。	2020 年 2 月 25 日
在主控台建立 AppConfig 組態	AppConfig 現在可讓您在建立組態設定檔時，在主控台中建立應用程式組態。如需詳細資訊，請參閱建立組態和組態設定檔。	2020 年 2 月 13 日

僅自動核准在指定日期之前發行的修補程式	除了自動核准修補程式以在發行指定天數後安裝的選項之外，Patch Manager 現在也支援僅自動核准在您指定日期當天或之前發行的修補程式。例如，如果您將 2020 年 7 月 7 日指定為修補基準中的截止日期，則不會自動安裝在 2020 年 7 月 8 日或之後發行的修補程式。如需詳細資訊，請參閱關於自訂基準和使用自訂修補基準 (主控台)。	2020 年 2 月 12 日
在維護時段任務中使用 {{RESOURCE_ID}} 虛擬參數	當您登錄維護時段任務時，您可以指定專屬於任務類型的參數。您可以使用虛擬參數語法來參考特定的值，例如 `{{TARGET_ID}}` 、`{{TARGET_TYPE}}` 和 `{{WINDOW_TARGET_ID}}` 。維護時段任務執行時，它會傳遞正確的值，而不是虛擬參數預留位置。若要支援將屬於資源群組的資源作為目標，您可以使用 `{{RESOURCE_ID}}` 虛擬參數傳遞資源的值，例如 DynamoDB 資料表、S3 儲存貯體和其他支援的類型。如需詳細資訊，請參閱教學課程：建立和設定維護時段 (AWS CLI) 中的下列主題： • 註冊維護時段任務時使用虛擬參數 • 範例：向維護時段註冊任務	2020 年 2 月 6 日

快速重新執行命令	Systems Manager 包含兩個選項，可協助您從 AWS Systems Manager 主控台的 Run Command頁面重新執行命令。Rerun (重新執行)：此按鈕可讓您執行相同的命令，而不對其進行變更。Copy to new (複製到新命令)：此按鈕會將一個命令的設定複製到新命令，並讓您選擇在執行之前編輯這些設定。如需詳細資訊，請參閱重新執行命令。	2020 年 2 月 5 日
從進階執行個體層還原至標準執行個體層	如果您先前已將在混合環境中執行的所有現場部署執行個體設定為使用進階執行個體層，您現在可以快速將這些執行個體設定為使用進階執行個體層。還原至 standard-instances 層適用於 AWS 帳戶 和單一 中的所有混合執行個體 AWS 區域。還原至 standard-instances 層會影響某些 Systems Manager 工具的可用性。如需詳細資訊，請參閱從進階執行個體層還原至標準執行個體層。	2020 年 1 月 16 日

可在安裝修補程式後略過執行個體重新啟動的新選項	先前，受管執行個體一律會在 Patch Manager 安裝修補程式後重新啟動。SSM 文件中的新 `RebootOption` 參數 `AWS-RunPatchBaseline` 可讓您指定是否要在安裝新修補程式後自動重新啟動執行個體。如需詳細資訊，請參閱主題關於 SSM 文件 `AWS-RunPatchBaseline` 中的參數名稱：RebootOption。	2020 年 1 月 15 日
新主題：「在 Linux 執行個體上執行 PowerShell 指令碼」	描述如何使用 Run Command 在 Linux 執行個體上執行 PowerShell 指令碼的新主題。如需詳細資訊，請參閱在 Linux 執行個體上執行 PowerShell 指令碼。	2020 年 1 月 10 日
「將 SSM Agent 設定為使用代理」的更新	在設定 SSM Agent 以使用代理時所指定的值已更新，以反映 HTTP 代理伺服器和 HTTPS 代理伺服器的選項。如需詳細資訊，請參閱將 SSM Agent設定為使用代理。	2020 年 1 月 9 日

新的「安全」章節概述了保護 Systems Manager 資源的做法	《AWS Systems Manager 使用者指南》中的安全性新章節，可幫助您了解如何在使用 Systems Manager 時套用共同的責任模型。該章節中的主題，將說明如何設定 Systems Manager 以達到您的安全和合規目標。您也會了解如何使用其他 AWS 服務 來協助您監控和保護 Systems Manager 資源。 **Note** 在此更新中，使用者指南「驗證和存取控制」章節已換成更簡單的新章節：AWS Systems Manager的身分和存取管理。	2019 年 12 月 24 日
新的自訂 Automation Runbook 範例	我們已在使用者指南中新增一組範例自訂 Automation Runbook。這些範例說明如何使用各種 Automation 動作來簡化部署、故障診斷與維護任務，旨在協助您撰寫自己的自訂 Automation Runbook。如需詳細資訊，請參閱自訂 Automation Runbook 範例。您也可以在 Systems Manager 主控台中檢視 Amazon 受管 Automation Runbook 內容。如需詳細資訊，請參閱《Systems Manager Automation Runbook 參考》。	2019 年 12 月 23 日

支援 Oracle Linux	Systems Manager 現在支援 Oracle Linux 7.5 和 7.7。如需在 Oracle Linux 執行個體的 EC2 執行個體上手動安裝 SSM Agent 的相關資訊，請參閱 Oracle Linux。如需有關在混合環境中的 Oracle Linux 伺服器上安裝 SSM Agent 的詳細資訊，請參閱如何在混合 Linux 節點上安裝 SSM Agent。	2019 年 12 月 19 日
從 Amazon EC2 主控台中啟動 Session Manager 工作階段	您現在可以從 Amazon Elastic Compute Cloud (Amazon EC2) 主控台中啟動 Session Manager 工作階段。從 Amazon EC2 主控台使用與工作階段相關的任務，使用者和系統管理員都需要不同的 IAM 許可。您可以提供使用Session Manager主控台的許可， AWS CLI 以及僅提供使用 Amazon EC2 主控台的許可，或提供使用全部三個工具的許可。如需詳細資訊，請參閱下列主題。 • Session Manager的預設 IAM 政策快速入門 • 啟動工作階段 (Amazon EC2 主控台)	2019 年 12 月 18 日

CloudWatch 支援 Run Command 指標和警示	AWS Systems Manager 現在會將有關Run Command命令狀態的指標發佈至 CloudWatch，讓您可以根據這些指標設定警示。您可追蹤指標之命令的終端狀態值包括 `Success`、`Failed` 和 `Delivery Timed Out`。如需詳細資訊，請參閱使用 Amazon CloudWatch 監控 Run Command 指標。	2019 年 12 月 17 日
新的 Systems Manager 工具：Change Calendar	使用 Systems Manager Change Calendar，在您想要對資源限制或禁止程式碼變更 (例如使用 Systems Manager Automation Runbook 或 AWS Lambda 函數) 的期間指定時段 (事件)。變更行事曆是一個新的 Systems Manager 文件類型，它可存放純文字格式的 iCalendar 2.0 資料。如需詳細資訊，請參閱 AWS Systems Manager 變更行事曆。	2019 年 12 月 11 日

新的 Systems Manager 工具：AWSAppConfig	使用 AppConfig 協助您建立、管理及快速部署應用程式組態。AppConfig 支援對任何大小應用程式的受控制部署。您可以AppConfig搭配 EC2 執行個體 AWS Lambda、容器、行動應用程式或 IoT 裝置上託管的應用程式使用 。為了防止部署應用程式組態時發生錯誤，AppConfig 包括了驗證器。驗證器提供了一個語法或語義檢查，以確保您要部署的組態可如預期運作。AppConfig 會在組態部署期間監視應用程式，以確保部署成功。如果系統遇到錯誤或部署開啟警示，AppConfig 會復原變更，以將對應用程式使用者的影響降到最低。如需詳細資訊，請參閱AWSAppConfig。	2019 年 11 月 25 日

新的 Systems Manager 工具：Systems Manager Explorer	AWS Systems Manager Explorer 是可自訂的操作儀表板，可報告 AWS 資源的相關資訊。 Explorer會顯示您 AWS 帳戶 和 之間的操作資料 (OpsData) 彙總檢視 AWS 區域。在 Explorer 中，OpsData 包含有關 EC2 執行個體、修補程式合規詳細資料和操作工作項目 (OpsItems) 的中繼資料。Explorer 提供脈絡指出 OpsItems 如何分散於業務單位或應用程式、如何隨著時間而演變趨勢，以及如何隨類別而變化。您可以在 Explorer 中群組和篩選資訊，以專注於與您相關且需要採取動作的項目。當您發現高優先順序的問題時，您可以使用 Systems Manager OpsCenter 來執行 Automation Runbook，並快速解決這些問題。如需相關資訊，請參閱 AWS Systems ManagerExplorer。 **Note** 設定 Systems Manager OpsCenter 與設定 Explorer 整合。如果已設定 OpsCenter，您仍然需要完成整合式設定，以確認設定和選項。如果您尚未設定 OpsCenter，則可以使用整合式設定來開始使用這兩	2019 年 11 月 18 日

	種工具。如需詳細資訊，請參閱 Explorer 和 OpsCenter 入門。	
改進的參數搜尋功能	現在，當您的帳戶中有大量參數或當您不記得參數的確切名稱時，用於搜尋參數的工具可讓您更輕鬆地尋找參數。有了搜尋工具，您現在可以依據 `contains` 篩選。之前，搜尋工具僅支援依據 `equals` 和 `begins-with` 搜尋參數名稱。如需詳細資訊，請參閱搜尋 Systems Manager 參數。	2019 年 11 月 15 日

適用於自動化的主控台型全新文件建置器 | 支援在自動化步驟中執行指令碼	您現在可以使用 Systems Manager Automation 來建置和共用標準化的操作手冊 AWS 帳戶，以確保使用者和 之間的一致性 AWS 區域。使用 Markdown 來執行指令碼並將內嵌文件新增至 Automation Runbook，您可以減少錯誤並消除手動步驟，例如在 Wiki 中瀏覽書面程序以及執行終端機命令。 如需詳細資訊，請參閱下列主題。 • 演練：使用文件建置器建立自訂 Automation Runbook • `aws:executeScript` (Automation 動作參考) • 使用文件建置器建立 Automation Runbook • AWS 新聞部落格上的 Systems Manager 中的新自動化功能	2019 年 11 月 14 日

使用 Distributor 執行就地套件更新	之前，當您想要使用 Distributor 將更新安裝至套件時，唯一的選擇是解除安裝整個套件並重新安裝新版本。現在，您可以選擇改為執行就地更新。在就地更新期間，Distributor 會根據您在套件中包含的 update script (更新指令碼)，僅安裝自上次安裝後新增或變更的檔案。使用此選項，您的套件應用程式可以保持可用，且不需在更新期間離線。如需詳細資訊，請參閱下列主題。 • 建立套件 • 安裝或更新套件	2019 年 11 月 11 日
新的 SSM Agent 自動更新功能	只要按一下，您就可以設定 中的所有執行個體 AWS 帳戶 ，以自動檢查和下載新版本的 SSM Agent。若要這樣做，請在 AWS Systems Manager 主控台的受管執行個體頁面上選擇客服人員自動更新。如需詳細資訊，請參閱自動更新 SSM Agent。	2019 年 11 月 5 日

使用 AWS提供的標籤限制 Session Manager存取	現在提供第二種控制使用者存取工作階段動作的方法。這個新方法可讓您使用 AWS提供的工作階段標籤來建立 IAM 存取政策，不必使用 `{aws:username}` 變數。使用這些 AWS 提供的工作階段標籤，可讓使用聯合 IDs 的組織控制使用者存取工作階段。如需相關資訊，請參閱僅允許使用者終止他們啟動的工作階段。	2019 年 10 月 2 日
新的 SSM Command 文件會套用 Ansible 手冊	您可以使用 `AWS-ApplyAnsiblePlaybooks` 文件來建立執行 Ansible 手冊的 State Manager 關聯。本文件提供執行手冊的下列優點： • 支援執行複雜的手冊 • 支援從 GitHub 和 Amazon Simple Storage Service (Amazon S3) 下載手冊 • 支援壓縮的手冊結構 • 增強型日誌 • 能夠指定綁定多個手冊時，要執行哪個手冊 如需詳細資訊，請參閱建立執行 Ansible 手冊的關聯	2019 年 9 月 24 日

Session Manager 的網路埠轉送支援	Session Manager 現在支援網路埠轉遞工作階段。網路埠轉遞可讓您安全地建立您在私有子網路中所部署執行個體間的通道，而無須在伺服器上啟動 SSH 服務來在安全群組中開啟 SSH 連接埠，或是使用堡壘主機。與 SSH 通道相似，網路埠轉遞可讓您轉送您筆記型電腦間的流量，來在您的執行個體上開啟連接埠。設定網路埠轉遞後，您便可以連線到本機連接埠，然後存取在執行個體內執行的伺服器應用程式。如需詳細資訊，請參閱下列主題： • AWS 新聞部落格中的使用 AWS Systems ManagerSession Manager 的連接埠轉送 • 啟動工作階段 (網路埠轉遞)	2019 年 8 月 29 日

指定預設參數層或自動化層選取項目	您現在可以指定請求使用的預設參數層，來建立或更新並未指定層的參數。您可以將預設層設為標準參數、進階參數，或是全新的 Intelligent-Tiering 選項。Intelligent-Tiering 會評估每個 PutParameter 請求，並只在需要的時候建立進階參數。(若參數的大小超過 4 KB、參數具備相關聯的參數政策，或是已建立標準層支援的最大 10,000 個參數，便需要進階參數。) 如需指定預設層和使用 Intelligent-Tiering 的詳細資訊，請參閱指定預設參數層。	2019 年 8 月 27 日
「使用關聯」一節已更新為包含 CLI 和 PowerShell 程序	使用關聯區段已更新，以包含使用 AWS CLI 或 管理關聯的程序文件 AWS Tools for PowerShell。如需資訊，請參閱在 Systems Manager 中使用關聯。	2019 年 8 月 26 日
「使用自動化執行」一節已更新為包含 CLI 和 PowerShell 程序	已更新使用自動化執行一節，以包含使用 AWS CLI 或 執行自動化工作流程的程序文件 AWS Tools for PowerShell。如需相關資訊，請參閱使用自動化執行。	2019 年 8 月 20 日

OpsCenter 與 Application Insights 整合	OpsCenter 與 .NET 和 SQL Server 的 Amazon CloudWatch Application Insights 整合。這表示您可以為您應用程式中偵測到的問題自動建立 OpsItems。如需關於如何設定 Application Insights 以建立 OpsItems 的資訊，請參閱《Amazon CloudWatch 使用者指南》中的設定、配置和管理應用程式以進行監控。	2019 年 8 月 7 日

新的主控台功能： AWS Systems ManagerQuick Setup	Quick Setup 是 Systems Manager 主控台的新功能，可協助您在 EC2 執行個體上快速設定數個 Systems Manager 元件。特別是，快速設定可協助您使用標籤，以在您選擇或設為目標的執行個體上設定下列元件： • Systems Manager 的 AWS Identity and Access Management (IAM) 執行個體描述檔角色。 • 依據排程每兩個月更新 SSM Agent。 • 依據排程每 30 分鐘收集清查中繼資料。 • 每日掃描您的執行個體，識別遺漏的修補程式。 • Amazon CloudWatch 代理程式的一次性安裝和設定。 • 依據排程每月更新 CloudWatch 代理程式。 如需詳細資訊，請參閱 AWS Systems Manager Quick Setup。	2019 年 8 月 7 日

將資源群組註冊為維護時段目標	除了將受管執行個體註冊為維護時段的目標之外，您現在可以將資源群組註冊為維護時段目標。 Maintenance Windows支援 支援 AWS Resource Groups 的所有 AWS 資源類型，包括 `AWS::EC2::Instance` 、`AWS::DynamoDB::Table` 、`AWS::OpsWorks::Instance` 、`AWS::Redshift::Cluster` 等。透過此版本，您也可以將命令傳送至資源群組，例如使用 Run Command主控台或 AWS CLI send-command命令。如需詳細資訊，請參閱下列主題： • 將目標指派給維護時段 (主控台) • 範例：向維護時段註冊目標 • 使用目標和速率控制將命令傳送至機群	2019 年 7 月 23 日
使用 AWS Systems ManagerDistributor 簡化的套件建立和版本控制	Distributor 有全新、簡化的套件建立工作流程，此工作流程可以為您產生套件資訊清單、指令碼和檔案雜湊。您也可以在將版本新增到現有的套件時使用簡化的工作流程。	2019 年 7 月 22 日

Systems Manager Automation 的全新文件類別窗格	當您在主控台中執行 Automation 時，Systems Manager 會包含全新的文件類別窗格。使用此窗格來根據類別用途篩選 Automation Runbook。	2019 年 7 月 18 日
支援使用作業系統使用者登入資料來開始 Session Manager 工作階段	在預設情況下，系統會使用在受管執行個體上建立的系統產生之 `ssm-user` 帳戶登入資料來啟動 Session Manager 工作階段。在 Linux 機器上，您現在可以改用作業系統帳戶的登入資料來啟動工作階段。如需相關資訊，請參閱為 Linux 執行個體開啟執行身分支援。	2019 年 7 月 9 日
支援使用 SSH 來啟動 Session Manager 工作階段	您現在可以使用 在受管執行個體上使用 AWS CLI 啟動 SSH 工作階段Session Manager。如需使用 Session Manager 啟用 SSH 工作階段的詳細資訊，請參閱 (選用) 開啟 SSH Session Manager工作階段。如需使用Session Manager啟動 SSH 工作階段的詳細資訊，請參閱啟動工作階段 (SSH)。	2019 年 7 月 9 日
支援在受管執行個體上變更密碼	您現在可以使用 Systems Manager (受管執行個體) 在您管理的機器上重設密碼。您可以使用 Systems Manager 主控台或 AWS CLI來重設密碼。如需相關資訊，請參閱在受管執行個體上重設密碼。	2019 年 7 月 9 日

「什麼是 AWS Systems Manager？」的修訂	什麼是 AWS Systems Manager？ 中的簡介內容已擴展，以提供更廣泛的服務簡介，並反映最近發佈的 Systems Manager 工具。此外，本章節中的其他內容已移到個別主題，以改善可探索性。	2019 年 6 月 10 日
新的 Systems Manager 工具：OpsCenter	OpsCenter 提供中央位置，讓營運工程師和 IT 專業人員可以檢視、調查和解決與 AWS 資源相關的操作工作項目 (OpsItems)。 OpsCenter 旨在減少對影響 AWS 資源的問題進行解決的平均時間。此 Systems Manager 工具會跨 OpsItems服務彙總和標準化OpsItems，同時提供每個 OpsItem、相關和相關資源的相關內容調查資料。 OpsCenter也提供 Systems Manager Automation Runbook，可讓您用來快速解決問題。您可以為每個 OpsItem 指定可搜尋的自訂資料。您也可以依狀態及來源，檢視自動產生的 OpsItems 摘要報告。如需更多詳細資訊，請參閱 AWS Systems ManagerOpsCenter。	2019 年 6 月 6 日

中的 Systems Manager 左側導覽窗格變更 AWS Management Console	中的 Systems Manager 左側導覽窗格 AWS Management Console 包含新的標題，包括 Ops Center 的新標題，可提供更邏輯的 Systems Manager 工具分組。	2019 年 6 月 6 日
修訂使用 AWS CLI來建立和設定維護時段的教學	教學課程：建立和設定維護時段 (AWS CLI) 已全面修訂，透過練習步驟提供簡單路徑。建立一個維護時段、識別單一目標，並為要執行的維護時段設定簡單的任務。過程中，我們會提供資訊和範例，您可用來建立自己任務註冊命令，包括使用 {{TARGET_ID}} 這類虛擬參數的資訊。如需其他資訊及範例，請參閱下列主題： • 範例：向維護時段註冊目標 • 範例：向維護時段註冊任務 • 關於 register-task-with-maintenance-windows 選項 • 註冊維護時段任務時使用虛擬參數	2019 年 5 月 31 日
SSM Agent 更新的相關通知	若要接收有關 SSM Agent 更新的通知，請訂閱 GitHub 上的 SSM Agent 版本備註頁面。	2019 年 5 月 24 日

根據 Parameter Store 中的變更接收通知或觸發動作	設定以 Parameter Store 事件為基礎的通知或觸發動作主題現在可協助您設定 Amazon EventBridge 規則，以回應 Parameter Store 中的變更。您可以在發生下列任何情況時收到通知或觸發其他動作： • 建立、更新或刪除參數。 • 建立、更新或刪除參數標籤版本。 • 參數過期、即將過期或在指定期間尚未變更。	2019 年 5 月 22 日

設定和入門內容的主要修訂	我們已擴充並重新組織了《AWS Systems Manager 使用者指南》中的設定和入門內容。「設定」內容已分為兩個部分。一部分著重的任務是設定 Systems Manager 以設定及管理 EC2 執行個體。另一部分著重於設定 Systems Manager 以在混合環境中設定及管理現場部署伺服器和虛擬機器 (VM) 的任務。這兩個部分現在都會以主要編號步驟來呈現所有設定主題，並建議完成順序。新的入門章節著重於協助最終使用者在完成帳戶和服務設定任務之後開始使用 Systems Manager。 • 設定 AWS Systems Manager • AWS Systems Manager 設定混合環境 • 入門 AWS Systems Manager	2019 年 5 月 15 日

包含修補基準中適用於 Microsoft 應用程式的修補程式 (Windows)	Patch Manager 現在支援 Windows Server 執行個體上 Microsoft 應用程式的修補程式更新。以前，僅支援 Windows Server 作業系統的修補程式。Patch Manager 為 Windows Server 執行個體提供兩種預先定義的修補基準。修補基準 `AWS-WindowsPredefinedPatchBaseline-OS` 僅適用於作業系統修補程式。`AWS-WindowsPredefinedPatchBaseline-OS-Applications` 適用於 Windows Server 作業系統和 Windows 上由 Microsoft 發行的應用程式。如需建立自訂修補基準 (其中包括 Microsoft 發行之應用程式的修補程式) 的詳細資訊，請參閱建立自訂修補基準的第一個程序。此外，作為此更新的一部分，正在變更 AWS所提供預先定義修補程式基準的名稱。如需詳細資訊，請參閱預先定義的基準。	2019 年 5 月 7 日
使用 註冊維護時段目標的範例 AWS CLI	新主題範例：向維護時段註冊目標提供三個範例命令，示範您在使用 AWS CLI時可用不同方法指定維護時段的目標。該主題還說明了每一個範例命令的最佳使用案例。	2019 年 5 月 3 日

修補程式群組主題的更新	關於修補程式群組主題已更新，加入一節討論受管執行個體如何決定修補操作期間適用的修補基準。此外，已新增使用 AWS CLI 或 Systems Manager 主控台將修補程式群組或PatchGroup標籤新增至受管執行個體的指示，以及如何將修補程式群組或PatchGroup新增至修補程式基準。(如果您已在 EC2 執行個體中繼資料中允許標籤，則必須使用 **PatchGroup** ，不留空格。)如需詳細資訊，請參閱建立修補程式群組和將修補程式群組新增至修補基準。	2019 年 5 月 1 日

新的 Parameter Store 功能	Parameter Store 提供以下的新功能： • 進階參數： Parameter Store 現在可讓您個別設定參數要使用標準的參數方案 (預設方案) 或進階參數方案。進階參數為參數值提供較大的配額、為每個 AWS 帳戶 和建立的參數數量提供更高的配額 AWS 區域，以及使用參數政策的能力。如需進階參數的詳細資訊，請參閱關於 Systems Manager 進階參數。 • 參數政策： 參數政策讓您可以將特定條件 (例如過期日期或存留時間) 指派給一個參數，以協助您管理一群不斷增長的參數。參數政策對於強制更新或刪除密碼和儲存於 Parameter Store 中的組態資料特別有幫助。參數政策僅適用於使用進階參數方案的參數。如需詳細資訊，請參閱使用參數政策。 • 較高的輸送量： 您現在可以將 Parameter Store 輸送量的配額提高到每秒最多 1,000 筆交易。如需詳細資訊，請參閱增加Parameter Store輸送量。	2019 年 4 月 25 日

自動化部分的更新	自動化部分已更新，更容易識別。此外，自動化部分增加了三個新主題： • 逐步執行自動化 • 執行需要核准的自動化 • 使用 State Manager 關聯排程自動化	2019 年 4 月 17 日
使用 AWS KMS 金鑰加密工作階段資料	根據預設，Session Manager 會使用 TLS 1.2，將您帳戶中使用者的本機機器與 EC2 執行個體之間傳輸的工作階段資料加密。現在，您可以選擇使用已在其中建立 AWS KMS key 的 進一步加密該資料 AWS Key Management Service。您可以使用 KMS 金輪在您的 AWS 帳戶 中建立，或從其他帳戶與您共享的金鑰。如需有關指定 KMS 金鑰來加密工作階段資料的資訊，請參閱開啟工作階段資料的 AWS KMS 金鑰加密 （主控台）、建立Session Manager偏好設定 (AWS CLI) 或更新Session Manager偏好設定 (AWS CLI)。	2019 年 4 月 4 日

設定 的 Amazon SNS 通知 AWS Systems Manager	新增使用 AWS CLI 或 Systems Manager 主控台為 Run Command和註冊到維護時段Run Command的任務設定 Amazon SNS 通知的指示。如需詳細資訊，請參閱設定 AWS Systems Manager的 Amazon SNS 通知。	2019 年 3 月 6 日
混合環境中伺服器和虛擬機器的進階執行個體	AWS Systems Manager 為混合環境中的伺服器和VMs 提供 standard-instances 層和 advanced-instances 層。Standard-instances 方案可讓您 AWS 帳戶 為每個 註冊最多 1，000 個伺服器或VMs AWS 區域。如果您需要在單一帳戶和區域中登錄 1,000 部以上的伺服器或虛擬機器，則使用進階執行個體層。在進階執行個體層中，您可以建立任意數量的執行個體，但針對 Systems Manager 設定的所有執行個體均按依使用量付費的方式提供。進階執行個體也可讓您使用 連線至您的混合機器 AWS Systems Manager Session Manager。 Session Manager提供執行個體的互動式 shell 存取。如需有關啟用進階執行個體的詳細資訊，請參閱使用進階執行個體層。	2019 年 3 月 4 日

建立使用共用 SSM 文件的 State Manager 關聯	您可以建立使用從其他 共用的 SSM Command 和 Automation Runbook 的State Manager 關聯 AWS 帳戶。使用共享 SSM 文件建立關聯有助於讓您的 Amazon EC2 和混合基礎設施維持於一致狀態，即使執行個體不在同一帳戶。如需有關共用 SSM 文件的詳細資訊，請參閱 AWS Systems Manager 文件。如需有關建立 State Manager關聯的資訊，請參閱建立關聯。	2019 年 2 月 28 日
檢視 Amazon EventBridge 規則支援的 Systems Manager 事件清單	新主題以 Amazon EventBridge 監控 Systems Manager 事件提供 Systems Manager 發出的各種事件的摘要，您可以在 EventBridge 中設定這些事件的監控規則。	2019 年 2 月 25 日
在建立 Systems Manager 資源時新增標籤	Systems Manager 現在支援在建立標籤時，將其新增至特定資源類型的功能。使用 AWS CLI 或 SDK 建立資源時，您可以標記這些資源，包括維護時段、修補程式基準、Parameter Store參數和 SSM 文件。建立受管執行個體的啟用時，也可以將標籤指派給該執行個體。使用 Systems Manager 主控台時，您可以將標籤新增至維護時段、修補基準和參數。	2019 年 2 月 24 日

Systems Manager Inventory 的自動 IAM 角色建立	先前您必須建立 AWS Identity and Access Management (IAM) 角色，並將個別政策連接至此角色，才能在 主控台的庫存詳細資訊檢視頁面上檢視庫存資料。您不再需要建立此角色或將政策附加至該角色。當您在 Inventory Detail View (清查詳細檢視) 頁面上選擇遠端資料同步時，Systems Manager 會自動建立 `Amazon-GlueServicePolicyForSSM` 角色，然後將 Amazon-GlueServicePolicyForSSM-{S3 儲存貯體名稱} 政策和AWSGlueServiceRole 政策指派給該角色。如需詳細資訊，請參閱查詢來自多個區域和帳戶的清查資料。	2019 年 2 月 14 日
Maintenance Windows 更新 SSM Agent 的演練	在 Maintenance Windows 文件中增加兩個新的逐步解說：逐步解說詳細說明如何使用 Systems Manager 主控台或 AWS CLI 來建立自動保持 SSM Agentup-to-date的維護時段。如需詳細資訊，請參閱Maintenance Windows演練。	2019 年 2 月 11 日
使用 Parameter Store 公有參數	新增說明 Parameter Store 公有參數的簡短部分。如需詳細資訊，請參閱使用 Systems Manager 公有參數。	2019 年 1 月 31 日

使用 AWS CLI 建立Session Manager偏好設定	新增使用 AWS CLI 建立 Session Manager偏好設定的指示，例如 CloudWatch Logs、S3 儲存貯體記錄選項和工作階段加密設定。如需詳細資訊，請參閱使用 AWS CLI 建立Session Manager偏好設定。	2019 年 1 月 22 日
使用 State Manager 執行 Systems Manager 自動化工作流程	AWS Systems Manager State Manager 現在支援建立使用 SSM Automation Runbook 的關聯。 State Manager 先前僅支援 `command`和 `policy` 文件，這表示您只能建立以受管執行個體為目標的關聯。由於支援 SSM Automation Runbook，您現在可以針對不同類型的 AWS 資源建立關聯。如需詳細資訊，請參閱使用 State Manager 執行 Systems Manager Automation 工作流程。	2019 年 1 月 22 日
Cron 和 Rate 運算式及維護時段排程選項的參考更新	Systems Manager 的 Cron 與 Rate 運算式主題已修訂。新版本提供更多的範例和更清楚的說明，解釋如何使用 cron 和 rate 運算式，來將維護時段與 State Manager 的關聯排程。此外，新主題Maintenance Windows排程與作用期間選項說明維護時段的各種排程相關選項 (開始日期、結束日期、時區、排程頻率) 彼此之間的關係。	2018 年 12 月 6 日

開啟 SSM Agent 偵錯記錄	您可以藉由編輯受管執行個體上的 seelog.xml.template 檔案，來開啟 SSM Agent 除錯日誌記錄功能。如需詳細資訊，請參閱開啟 SSM Agent 除錯日誌記錄。	2018 年 11 月 30 日
ARM64 處理器架構支援	AWS Systems Manager 現在支援 Amazon Linux 2、7.6 Red Hat Enterprise Linux 和 Ubuntu Server(18.0ARM64 LTS 和 16.04 LTS) 作業系統的 ARM64 版本。如需詳細資訊，請參閱 Amazon Linux 2、RHEL 以及 Ubuntu Server 18.04 和 16.04 LTS 搭配 Snap 套件的安裝說明。如需有關 A1 執行個體類型的詳細資訊，請參閱《Amazon EC2 使用者指南》中的 General purpose instances。	2018 年 11 月 26 日

使用 建立和部署套件 AWS Systems ManagerDistributor	使用 AWS Systems Manager Distributor，您可以封裝自己的軟體，或尋找 AWS提供的代理程式軟體套件，例如，`AmazonCloudWatchAgent` 在 AWS Systems Manager 受管執行個體上安裝 。 Distributor 會將軟體套件等資源發佈到 AWS Systems Manager 受管執行個體。發佈套件會向受管執行個體 (您根據受管執行個體 ID、 AWS 帳戶 ID、標籤或 AWS 區域所找出的執行個體) 告知套件文件的特定版本，而套件文件是您在 Distributor 中新增套件時所建立的 Systems Manager 文件。如需詳細資訊，請參閱AWS Systems ManagerDistributor。	2018 年 11 月 20 日
同時從中央帳戶跨多個 AWS 區域 和 AWS 帳戶 執行 AWS Systems Manager 自動化工作流程	您可以從 AWS Systems Manager 自動化管理帳戶同時跨多個 AWS 區域 和 AWS 帳戶 或 AWS 組織單位 (OUs) 執行自動化工作流程。在多個區域和帳戶或 OU 中並行執行 Automation 操作，可減少管理 AWS 資源所需的時間，同時提升運算環境的安全性。如需詳細資訊，請參閱在多個 AWS 區域 和 中執行自動化工作流程 AWS 帳戶。	2018 年 11 月 19 日

查詢來自多個 AWS 區域 和 的庫存資料 AWS 帳戶	Systems Manager 庫存與 Amazon Athena 整合，協助您查詢來自多個 AWS 區域 和 的庫存資料 AWS 帳戶。Athena 整合使用資源資料同步，因此您可以在 AWS Systems Manager 主控台的庫存詳細資訊檢視頁面上檢視所有受管執行個體的庫存資料。如需詳細資訊，請參閱查詢來自多個區域和帳戶的清查資料。	2018 年 11 月 15 日
建立 State Manager 關聯來執行 MOF 檔案	您可以使用 AWS-ApplyDSCMofs SSM 文件，來執行受管物件格式 (MOF) 檔案，從而透過 State Manager 在 Windows Server 受管執行個體上強制達成目標狀態。AWS-ApplyDSCMofs 文件有兩種執行模式。運用第一種模式，您可以設定關聯，來進行掃描並報告受管執行個體目前的狀態是否符合在指定 MOF 檔案中定義的目標狀態。在第二種模式中，您可以執行 MOF 檔案，並根據資源和 MOF 檔案中所定義的資源值，來變更您執行個體的組態。AWS-ApplyDSCMofs 文件允許您從 Amazon Simple Storage Service (Amazon S3)、本機共用、或具有 HTTPS 網域的安全網站中下載和執行 MOF 組態檔案。如需詳細資訊，請參閱建立執行 MOF 檔案的關聯。	2018 年 11 月 15 日

在 Session Manager 工作階段中限制管理存取	Session Manager 工作階段會透過使用者帳戶的登入資料啟動，此帳戶是使用預設根或稱為 ssm-user 的管理員權限所建立的。在啟用或停用 ssm-user 帳戶管理許可主題中，現在提供有關對此帳戶限制管理控制權限的資訊。	2018 年 11 月 13 日
自動化動作參考中的 YAML 範例	現在，自動化動作參考在已包含 JSON 範例的每個動作中加入 YAML 範例。	2018 年 10 月 31 日
指派合規嚴重性等級給關聯	您現在可以將合規嚴重性等級，指派給 State Manager 的關聯。這些嚴重性等級會在合規儀表板中呈報，也可以用來篩選您的合規報告。您可以指派的嚴重性等級包括重大、高、中、低和未指定。如需詳細資訊，請參閱建立關聯 (主控台)。	2018 年 10 月 26 日
使用目標和速率控制搭配自動化與 State Manager	利用目標、並行和錯誤閾值，來控管資源機群之間的自動化執行和 State Manager 關聯。如需詳細資訊，請參閱使用目標和速率控制在機群上執行自動化工作流程和對State Manager關聯使用目標和速率控制。	2018 年 10 月 23 日

為維護時段指定有效的時間範圍和國際時區	您也可以指定不應該在之前或之後執行維護時段的日期 (開始日期和結束日期)，同時您可以指定維護時段排程做為基準的國際時區。如需詳細資訊，請參閱建立維護時段 (主控台)和更新維護時段 (AWS CLI)。	2018 年 10 月 9 日
在 S3 儲存貯體中，針對您的修補基準，維持修補程式的自訂清單	使用 SSM 命令文件 `AWS-RunPatchBaseline` 中的新 'InstallOverrideList' 參數，您可以將 https URL 或 Amazon Simple Storage Service (Amazon S3) 路徑樣式 URL 指定至要安裝的修補程式清單。此修補程式安裝清單 (YAML 格式，留存於 S3 儲存貯體中) 會覆寫預設修補基準所指定的修補程式。如需詳細資訊，請參閱參數名稱：InstallOverrideList。	2018 年 10 月 5 日
針對是否安裝修補程式的相依項目來擴展控制	之前，如果在拒絕的修補程式清單中，有修補程式被辨識為另一個修補程式的相依項目，則仍然會安裝該修補程式。現在，您可以選擇是否安裝這些相依項目，或是封鎖這些項目的安裝。如需詳細資訊，請參閱建立修補基準。	2018 年 10 月 5 日

使用條件式分支建立動態自動化工作流程	`aws:branch` Automation 動作可讓您建立動態 Automation 工作流程，此等流程可評估單一步驟中的多項選擇，然後根據評估的結果，跳到 Automation Runbook 中的不同步驟。如需詳細資訊，請參閱在執行手冊中使用條件陳述式。	2018 年 9 月 26 日
使用 AWS CLI 更新Session Manager偏好設定	使用 CLI 來更新 Session Manager 偏好設定 (例如 CloudWatch Logs 和 S3 儲存貯體日誌記錄選項) 的說明已新增至 AWS Systems Manager 使用者指南。如需詳細資訊，請參閱使用 AWS CLI 更新 Session Manager偏好設定。	2018 年 9 月 25 日
更新 Session Manager 的 SSM Agent 需求	Session Manager 現在需要 SSM Agent 2.3.68.0 版或更新版本。如需關於 Session Manager 先決條件的詳細資訊，請參閱完成 Session Manager 先決條件。	2018 年 9 月 17 日

使用 Session Manager，不需開啟傳入連接埠或維持堡壘主機，即可管理執行個體	使用 Session Manager、全受管工具 AWS Systems Manager， 您可以透過互動式一鍵式瀏覽器型 shell 或透過 來管理 EC2 執行個體 AWS CLI。 Session Manager 提供安全且可稽核的執行個體管理，無需開啟傳入連接埠。 維護堡壘主機、 或 管理 SSH 金鑰。 Session Manager也可讓您遵守需要受控制存取執行個體的公司政策， 嚴格的安全實務， 具有執行個體存取詳細資訊的可完整稽核日誌， 同時仍為最終使用者提供 EC2 執行個體的簡單一鍵式跨平台存取。 如需詳細資訊，請參閱進一步了解Session Manager。	2018 年 9 月 11 日
AWS 服務 從 Systems Manager 自動化工作流程叫用其他	您可以在自動化執行手冊中使用三個新的自動化動作（或外掛程式），在自動化工作流程中叫用其他 AWS 服務 和其他 Systems Manager 工具。如需詳細資訊，請參閱使用動作輸出作為輸入。	2018 年 8 月 28 日
在 IAM 政策中使用 Systems Manager 特定的條件金鑰	在政策中指定條件主題已更新，列出 Systems Manager 的 IAM 條件索引鍵，供您納入政策中。您可以利用這些索引鍵，來指定政策應該生效的條件。此主題也包含範例政策和其他相關主題的連結。	2018 年 8 月 18 日

使用群組彙總清查資料，以查看哪些執行個體已設定和未設定成收集某個清查類型	群組可供您快速查看設定成收集一個或多個庫存類型的受管執行個體計數，以及未進行設定的執行個體數。透過群組功能，您可以指定一個或多個清查類型，並使用 `exists` 運算子做為篩選條件。如需詳細資訊，請參閱彙總清查資料。	2018 年 8 月 16 日
針對庫存與組態合規，來檢視歷程記錄和追蹤變更	您現在可以針對從受管執行個體收集的庫存資料，檢視歷程記錄並追蹤變更。您也可以針對 Patch Manager 的修補，和組態合規所呈報的 State Manager 關聯，來檢視歷程記錄與追蹤變更。如需詳細資訊，請參閱檢視清查歷程記錄和變更追蹤。	2018 年 8 月 9 日

Parameter Store 與 Secrets Manager 整合	Parameter Store 現在已與 整合， AWS Secrets Manager 因此在使用已支援參考Parameter Store參數 AWS 服務 的其他 時，您可以擷取 Secrets Manager 秘密。這些服務包括 Amazon EC2、Amazon Elastic Container Service AWS Lambda、 AWS CodeBuild 、 AWS CodeDeploy、 AWS CloudFormation和其他 Systems Manager 工具。透過使用 Parameter Store 來參考 Secrets Manager 秘密，您就能夠建立安全、一致的流程，來呼叫和使用程式碼與組態指令碼中的秘密和參考資料。如需詳細資訊，請參閱從Parameter Store參數參考 AWS Secrets Manager 秘密。	2018 年 7 月 26 日
為 Parameter Store 參數加上標籤	參數標籤是使用者定義的別名，可協助您管理不同版本的參數。當您修改參數時， Systems Manager 會自動儲存新版本，並將版本號碼增加 1。如果參數有多個版本，標籤可協助您記住參數版本的目的。如需詳細資訊，請參閱標記參數。	2018 年 7 月 26 日

建立動態自動化工作流程	根據預設，您在 Automation Runbook 的 mainSteps (主要步驟) 區段中所定義的步驟 (或動作)，會按順序執行。在一項動作完成之後，mainSteps (主要步驟) 區段中所指定的下一個動作就會開始。在此版本中，您現在可以建立自動化工作流程，此流程會進行條件式分支。這表示您可以建立自動化工作流程，來動態回應條件的變動，並跳到指定的步驟。如需相關資訊，請參閱在執行手冊中使用條件陳述式。	2018 年 7 月 18 日
使用 Snap 的 SSM Agent 現在已預先安裝於 Ubuntu Server 16.04 AMIs 上	從 Ubuntu Server 16.04 AMIs (編號 20180627) 所建立的執行個體開始，會預先安裝使用 Snap 套件的 SSM Agent。在從較舊的 AMIs 中建立的執行個體上，您應該繼續使用 deb 安裝程式套件。如需詳細資訊，請參閱關於在 64 位元 Ubuntu Server 16.04 執行個體上安裝 SSM Agent。	2018 年 7 月 7 日

檢閱 SSM Agent 要求的最低限度 S3 許可	新主題SSM Agent 的最小 S3 儲存貯體許可提供 Amazon Simple Storage Service (Amazon S3) 儲存貯體的相關資訊，資源可能需要存取這些儲存貯體才能執行 Systems Manager 操作。如果您想要將執行個體設定檔或 VPC 端點使用 Systems Manager 時所需的 S3 儲存貯體存取權，限制在最低程度，您可以在自訂政策中指定這些儲存貯體。	2018 年 7 月 5 日
針對特定的 State Manager 關聯 ID，檢視完整的執行歷程記錄	新主題檢視關聯歷程記錄說明如何檢視特定關聯 ID 的所有執行，然後檢視一個或多個資源的執行詳細資訊。	2018 年 7 月 2 日
Patch Manager 引入了對 Amazon Linux 2 的支援	您現在可以使用 Patch Manager，來將修補程式套用到 Amazon Linux 2 執行個體。如需 Patch Manager 作業系統支援的一般資訊，請參閱 Patch Manager 先決條件。如需有關在定義修補程式篩選條件時 Amazon Linux 2 支援的鍵值對的相關資訊，請參閱 AWS Systems Manager API 參考中的 PatchFilter。	2018 年 6 月 26 日
將命令輸出傳送到 Amazon CloudWatch Logs	新主題針對 Run Command 設定 Amazon CloudWatch Logs 說明如何將 Run Command 輸出傳送至 CloudWatch Logs。	2018 年 6 月 18 日

利用 AWS CloudFormation來快速建立或刪除庫存的資源資料同步	您可以使用 AWS CloudFormation 來建立或刪除 Systems Manager 庫存的資源資料同步。若要使用 AWS CloudFormation，請將 AWS::SSM::ResourceDataSync 資源新增至您的 AWS CloudFormation 範本。如需詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的使用 AWS CloudFormation 範本。您也可以如建立庫存的資源資料同步中的說明，手動建立庫存的資源資料同步。	2018 年 6 月 11 日
AWS Systems Manager 使用者指南更新通知現在可透過 RSS 取得	《Systems Manager 使用者指南》的 HTML 版本現在支援在 Systems Manager Documentation Update History (Systems Manager 文件更新歷史記錄) 頁面中記錄的 RSS 摘要更新。RSS 饋送包括 2018 年 6 月份 (含) 以後所製作的更新。Systems Manager 文件更新歷程記錄頁面仍保留先前發佈的更新。使用頂部選單面板中的 RSS 按鈕來訂閱摘要。	2018 年 6 月 6 日
在指令碼中指定結束程式碼，來重新啟動受管執行個體	在新主題從指令碼重新啟動受管執行個體中，說明如何在您以 Run Command 執行的指令碼中指定結束程式碼，以指示 Systems Manager 重新啟動受管執行個體。	2018 年 6 月 3 日

每次刪除自訂清查時在 Amazon EventBridge 中建立事件	在新主題在 EventBridge 中檢視清查刪除動作中，說明如何將 Amazon EventBridge 設定為每次使用者刪除自訂清查時，就建立事件。	2018 年 6 月 1 日

2018 年 6 月前的更新

下表描述 2018 年 6 月前，每個 AWS Systems Manager 使用者指南版本的重要變更。

變更	描述	發行日期
清查您 中的所有受管執行個體 AWS 帳戶	您可以透過 AWS 帳戶 建立全域庫存關聯來清查 中的所有受管執行個體。如需詳細資訊，請參閱清查 中的所有受管節點 AWS 帳戶。 **Note** SSM Agent 2.0.790.0 版或更新版本皆能使用全域庫存關聯。如需如何在執行個體上更新 SSM Agent 的相關資訊，請參閱 使用 Run Command 更新 SSM Agent。	2018 年 5 月 3 日
SSM Agent預設安裝在 Ubuntu Server 18 上	SSM Agent 預設安裝在 Ubuntu Server 18.04 LTS 64 位元和 32 位元 AMI 上AMIs。	2018 年 5 月 2 日
新主題	新主題 使用特定文件版本執行命令 說明如何使用文件版參數，在指令執行時指定所要使用的 SSM 文件版本。	2018 年 5 月 1 日
新主題	新主題 刪除自訂清查 說明如何使用 AWS CLI，來從 Amazon Simple Storage Service (Amazon S3) 刪除自訂的庫存資料。此主題也說明如何使用 `SchemaDeleteOption` 來關閉或刪除自訂庫存類型，以管理自訂庫存。這項新功能會使用 DeleteInventory API 操作。	2018 年 4 月 19 日

變更	描述	發行日期
SSM Agent 的 Amazon SNS 通知	您可以訂閱 Amazon SNS 主題，在有可用的 SSM Agent 新版本時收到通知。如需詳細資訊，請參閱訂閱 SSM Agent 通知。	2018 年 4 月 9 日
CentOS 修補支援	Systems Manager 現在支援修補 CentOS 執行個體。關於支援的 CentOS 版本，詳細資訊請參閱 Patch Manager 先決條件。關於修補的運作方式，詳細資訊請參閱 Patch Manager 操作的運作方式。	2018 年 3 月 29 日
新增 章節	為了在 AWS Systems Manager 使用者指南中提供單一的參考資訊來源，推出了新的 AWS Systems Manager 參考 小節。其他內容將會在推出時加入到此小節中。	2018 年 3 月 15 日
新主題	新主題 已核准與遭拒的修補程式清單的套件名稱格式 詳細說明了您可以在自訂修補基準的核准和拒絕修補程式清單中，所輸入的套件名稱格式。針對 Patch Manager 所支援的每種作業系統類型，提供了範例格式。	2018 年 3 月 9 日
新主題	Systems Manager 現已整合 Chef Chef InSpec。InSpec 是一種開放原始碼的執行時期架構，讓您能夠在 GitHub 或 Amazon S3 建立人類可讀的設定檔。然後，您可以使用 Systems Manager 執行合規掃描，檢視合規與不合規的執行個體。如需詳細資訊，請參閱將 Chef InSpec 設定檔與 Systems Manager 合規搭配使用。	2018 年 3 月 7 日
新主題	新主題使用 Systems Manager 的服務連結角色說明如何搭配 Systems Manager 使用 AWS Identity and Access Management (IAM) 服務連結角色。目前，只有在使用 Systems Manager Inventory 來收集關於標記和 Resource Groups 的中繼資料時，才需要使用服務連結角色。	2018 年 2 月 27 日

變更	描述	發行日期
新增與更新主題	您現在可以使用 Patch Manager，來安裝不同來源儲存庫中的修補程式，而非安裝執行個體上設定好的預設修補程式。使用與安全無關的更新、Ubuntu Server 個人軟體套件封存檔 (PPA) 的內容，或是內部企業應用程式的更新等項目，來修補執行個體時，這項功能會非常實用。當您建立自訂修補基準時，可以指定替代的修補程式來源儲存庫。如需詳細資訊，請參閱下列主題： • 如何指定替代修補程式來源儲存庫 (Linux) • 使用自訂修補基準 • 建立包含不同作業系統版本之自訂儲存庫的修補基準 此外，您現在可以使用 Patch Manager 來修補 SUSE Linux Enterprise Server 執行個體。Patch Manager 支援修補 SLES 12.* 版本 (僅限 64 位元)。如需詳細資訊，請參閱下列主題中的 SLES 特定資訊： • 如何選取安全性修補程式 • 如何安裝修補程式 • 修補基準規則在 SUSE Linux Enterprise Server 上的運作方式	2018 年 2 月 6 日
新主題	新主題 用於修補受管節點的 SSM 命令文件 說明可用的 7 個 SSM 文件，協助您以最新的安全相關更新，讓受管執行個體獲得修補。	2018 年 1 月 10 日
關於 Linux 支援的重要更新	使用下列資訊來更新各種主題： • SSM Agent 預設會安裝於 Amazon Linux 1 基本 AMIs 2017.09 和更高的版本上。 • 手動將 SSM Agent 安裝在其他版本的 Linux 上，包括像是 Amazon ECS 最佳化之 AMIs 等非基礎映像。	2018 年 1 月 9 日

變更	描述	發行日期
新主題	新主題 用於修補的 SSM 命令文件：AWS-RunPatchBaseline 提供了詳細資訊，說明這個 SSM 文件如何在 Windows 和 Linux 系統上運作。此主題也針對 AWS-RunPatchBaseline 文件中的兩個可用參數 (Operation 和Snapshot ID) 提供了相關資訊。	2018 年 1 月 5 日
新增主題	新的 Patch Manager 操作的運作方式 小節，提供了技術詳細資訊，來說明 Patch Manager 如何決定安裝哪些安全修補程式，以及如何在各個支援的作業系統上安裝這些程式。此節也提供資訊，說明在 Linux 作業系統的不同版本上，修補程式的基準規則如何運作。	2018 年 1 月 2 日
修改標題，並移除 Systems Manager 自動化動作參考內容	根據客戶的意見反映，「自動化動作參考」現在稱為「Systems Manager 自動化 Runbook 參考」。此外，我們將該參考內容移到了「共用的資源」>「文件」節點，使其更接近 命令文件外掛程式參考。如需詳細資訊，請參閱Systems Manager Automation 動作參考。	2017 年 12 月 20 日
新的監控章節與內容	新章節 在 中記錄和監控 AWS Systems Manager 提供了指示，將指標和日誌資料傳送到 Amazon CloudWatch Logs。新主題 將節點日誌傳送至統一 CloudWatch Logs (CloudWatch 代理程式) 提供指示，說明如何將執行個體內監控任務從 SSM Agent 遷移到 CloudWatch (僅限於 64 位元 Windows Server 執行個體上)。	2017 年 12 月 14 日
新增章節	新章節 適用於 AWS Systems Manager 的 Identity and Access Management提供使用 AWS Identity and Access Management (IAM) 和 的完整資訊 AWS Systems Manager ，以協助透過使用 登入資料安全地存取您的 資源。這些登入資料提供存取 AWS 資源所需的許可，例如存取存放在 S3 儲存貯體中的資料，以及將命令傳送至 EC2 執行個體，以及讀取 EC2 執行個體上的標籤。	2017 年 12 月 11 日
對左側導覽列的變更	我們變更了本使用者指南左側導覽列中的標題，使其符合新 AWS Systems Manager 主控台中的標題。	2017 年 12 月 8 日

變更	描述	發行日期
re:Invent 2017 的多項變更	• 正式推出 AWS Systems Manager： AWS Systems Manager（先前為 Amazon EC2 Systems Manager) 是一個統一的界面，可讓您集中營運資料，並自動化資源 AWS 的任務。您可以在此處存取新的 AWS Systems Manager 主控台。如需詳細資訊，請參閱什麼是 AWS Systems Manager？ • YAML 支援：您可以在 YAML 中建立 SSM 文件。如需詳細資訊，請參閱AWS Systems Manager Documents。	2017 年 11 月 29 日
使用 Run Command 來擷取具備 VSS 功能的 EBS 磁碟區快照	使用 Run Command，可取得連接至 Amazon EC2 Windows 執行個體的所有 Amazon Elastic Block Store (Amazon EBS) 磁碟區的應用程式一致性快照。快照程序會使用 Windows 磁碟區陰影複製服務 (VSS) 來取得 VSS 感知應用程式的映像層級備份，其中包括這些應用程式與磁碟間的待定交易資料。此外，若需要備份所有連接的磁碟區，您不需要關閉執行個體或中斷其連結。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Take Microsoft VSS-Enabled Snapshots Using AWS Systems Manager。	2017 年 11 月 20 日
藉由使用 VPC 端點來提供增強的 Systems Manager 安全性	您可以將 Systems Manager 設定為使用界面 VPC 端點，來改善受管執行個體 (包括混合環境中的受管執行個體) 的安全狀態。界面端點採用 PrivateLink 技術，此技術可讓您使用私有 IP 地址，來以私密的方式存取 Amazon EC2 和 Systems Manager API。PrivateLink 會限制受管執行個體、Systems Manager 和 EC2 到 Amazon 網路之間的所有網路流量 (受管執行個體無法存取網際網路)。此外，您不需要網際網路閘道、NAT 裝置或虛擬私有閘道。如需詳細資訊，請參閱使用適用於 Systems Manager 的 VPC 端點來改善 EC2 執行個體的安全性。	2017 年 11 月 7 日

變更	描述	發行日期
檔案、服務、Windows 角色與 Windows 登錄檔的庫存清點支援	SSM Inventory 現在支援從您的受管執行個體，收集下列的資訊。 • 檔案：名稱、大小、版本、安裝日期、修改時間和上次存取時間等。 • 服務：名稱、顯示名稱、狀態、相依服務、服務類型、開啟類型等。 • Windows 登錄檔：登錄機碼路徑、數值名稱、數值類型和數值。 • Windows 角色：名稱、顯示名稱、路徑、功能類型、安裝狀態等。 在嘗試收集這些庫存類型的資訊之前，請在您想要清點的執行個體上更新 SSM Agent。透過執行最新版本的 SSM Agent，就能確保您可以收集所有受支援庫存類型的中繼資料。如需使用 SSM Agent 更新 State Manager 的相關資訊，請參閱 逐步解說：SSM Agent使用 自動更新 AWS CLI。 如需關於庫存的詳細資訊，請參閱 進一步了解 Systems Manager 庫存。	2017 年 11 月 6 日
自動化文件的更新	針對設置和設定對 Systems Manager 自動化的存取，修正相關資訊中的幾個問題。如需詳細資訊，請參閱設定自動化。	2017 年 10 月 31 日

變更	描述	發行日期
GitHub 和 Amazon S3 整合	執行遠端指令碼：Systems Manager 現支援從私有或公有 GitHub 儲存庫，以及從 Amazon S3 下載和執行指令碼。您可以使用 `AWS-RunRemoteScript` 預先定義的 SSM 文件或自訂 SSM 文件中的 `aws:downloadContent` 外掛程式，執行 Ansible 手冊和 Python、Ruby 或 PowerShell 等多種語言的指令碼。當您使用 Systems Manager 在混合式環境中自動化 EC2 執行個體和內部部署受管執行個體的組態和部署時，這些變更進一步增強了 Infrastructure as Code。如需詳細資訊，請參閱 從 GitHub 執行指令碼 及 從 Amazon Simple Storage Service (Amazon S3) 執行指令碼。 建立複合 SSM 文件：Systems Manager 現在支援從主要 SSM 文件，執行一個或多個次要 SSM 文件。這些執行其他文件的主要文件稱為複合文件。複合文件可讓您在 之間建立和共用一組標準次要 SSM 文件， AWS 帳戶 用於常見的任務，例如引導防毒軟體或加入網域的執行個體。您可以執行在 Systems Manager、GitHub 或 Amazon S3 中存放的複合和次要文件。在建立複合文件之後，您可以使用 SSM 預先定義的 `AWS-RunDocument` 文件，來執行該複合文件。如需詳細資訊，請參閱建立複合文件及從遠端位置執行 文件。 SSM 文件外掛程式參考為了便於使用，我們將 SSM 文件的 SSM 外掛程式參考，從 Systems Manager API 參考移出，並且移到使用者指南中。如需詳細資訊，請參閱命令文件外掛程式參考。	2017 年 10 月 26 日

變更	描述	發行日期
Parameter Store 中的參數版本支援	當您編輯參數時，Parameter Store 現在會自動將版本號碼重複增加 1。您可以在 API 呼叫與 SSM 文件中指定參數名稱和特定版本號碼。如果您不指定版本編號，系統會自動使用最新的版本。 參數版本可在意外變更參數時提供一層保護。您可以檢視所有版本的值，並在必要時參考較舊的版本。您也可以使用參數版本來查看參數變更的次數。如需詳細資訊，請參閱在 Parameter Store中使用參數版本。	2017 年 10 月 24 日
支援標記 Systems Manager 文件	您現在可以使用 AddTagsToResource API、 AWS CLI或 AWS Tools for PowerShell 來標記 Systems Manager 文件與鍵值對。標記功能可協助您根據指派給資源的標記，快速地找出特定資源。在受管執行個體、維護時段、Parameter Store 參數和修補基準的現有標記支援之外，這是額外的功能。	2017 年 10 月 3 日
各種文件更新，根據意見反映來修正錯誤或更新內容	• 已使用 Raspbian Linux 的資訊，更新 使用 Systems Manager 在混合多雲端環境中管理節點。 • 已更新 使用 Systems Manager 管理 EC2 執行個體，提出 Windows Server 執行個體的新需求。SSM Agent 需要 Windows PowerShell 3.0 或更新版本，才能在 Windows Server 執行個體上執行某些 SSM 文件 (例如，舊版 `AWS-ApplyPatchBaseline` SSM 文件)。確認您的 Windows Server 執行個體執行 Windows Management Framework 3.0 或更新版本。這個架構包含 PowerShell。如需詳細資訊，請參閱 Windows Management Framework 3.0。	2017 年 10 月 2 日

變更	描述	發行日期
使用 EC2Rescue 自動化工作流程，針對無法連線到的 Windows 執行個體，來排解問題	EC2Rescue 可協助您對 Amazon EC2 Windows Server 執行個體的問題進行故障診斷。您可以使用 AWSSupport-ExecuteEC2Rescue 文件，將工具作為 Systems Manager Automation 工作流程執行。AWSSupport-Execute EC2Rescue 文件旨在執行一組 Systems Manager 動作、AWS CloudFormation 動作和 Lambda 函數，以將通常必須使用 EC2Rescue 的步驟自動化。如需詳細資訊，請參閱在無法觸達的執行個體上執行 EC2Rescue 工具。	2017 年 9 月 29 日
在 Amazon Linux 上預設安裝 SSM Agent	SSM Agent 預設會安裝於 Amazon Linux AMIs 2017.09 和更高的版本上。在 Linux 的其他版本上手動安裝 SSM Agent，如 在 Linux 的 EC2 執行個體使用 SSM Agent 中所述。	2017 年 9 月 27 日
Run Command 增強功能	Run Command 包含下列增強功能。 • 您可以建立並指派 IAM 政策 (其中包含條件，限制使用者只能在標記了特定 Amazon EC2 標籤的執行個體上執行命令)，來限制只針對特定執行個體執行命令。如需詳細資訊，請參閱根據標籤限制 Run Command 存取。 • 透過使用 Amazon EC2 標記，在設定執行個體目標時，您可以擁有更多的選項。您現在可以在傳送指令時，指定多個標記索引鍵和多個標記值。如需詳細資訊，請參閱大規模執行命令。	2017 年 9 月 12 日
Raspbian 上支援 Systems Manager	Systems Manager 現在可以在 Raspbian Jessie 和 Raspbian Stretch 裝置上執行，包括 Raspberry Pi (32 位元)。	2017 年 9 月 7 日
將 SSM Agent 日誌自動傳送到 Amazon CloudWatch Logs	您現在可以在執行個體上進行簡單的組態變更，來讓 SSM Agent 將日誌檔傳送到 CloudWatch。如需詳細資訊，請參閱將 SSM Agent 日誌傳送至 CloudWatch Logs。	2017 年 9 月 7 日

變更	描述	發行日期
加密資源資料同步	Systems Manager 資源資料同步可讓在中央 S3 儲存貯體，彙總從數十或數百個受管執行個體收集的清查資料。您現在可以使用 AWS Key Management Service 金鑰來加密資源資料同步。如需詳細資訊，請參閱演練：使用資源資料同步來彙總庫存資料。	2017 年 9 月 1 日
新的 State Manager 逐步解說	在 State Manager 中增加兩個新的逐步解說： 逐步解說：SSM Agent使用 自動更新 AWS CLI 演練：在 Windows Server 的 EC2 執行個體自動更新 PV 驅動程式	2017 年 8 月 31 日
Systems Manager 組態合規	使用 組態合規 (Configuration Compliance) 功能，來掃描您的受管執行個體機群，以檢查修補程式合規與組態的不一致。您可以從多個 AWS 帳戶 和 收集和彙總資料 AWS 區域，然後深入探討不合規的特定資源。根據預設，組態合規會顯示有關 Patch Manager 修補和 State Manager 關聯的合規資料。您也可以根據 IT 或業務的需求，來自訂服務和建立自己的合規類型。如需詳細資訊，請參閱AWS Systems Manager合規。	2017 年 8 月 28 日
新 Automation 動作：`aws:executeAutomation`	會藉由呼叫次要 Automation Runbook 執行次要自動化工作流程。透過此動作，您可以為您最常用的工作流程建立 Automation Runbook，並在自動化執行期間參考那些文件。此動作可簡化您的 Automation Runbook，讓您不需要在類似的 Runbook 之間重複步驟。如需詳細資訊，請參閱`aws:executeAutomation` – 執行另一項自動化。	2017 年 8 月 22 日
讓 Automation 成為 CloudWatch 事件的目標	您可以將 Automation Runbook 指定為 Amazon CloudWatch 事件的目標，藉此開始 Automation 工作流程。您可以根據排程或在發生特定 AWS 系統事件時啟動工作流程。如需詳細資訊，請參閱根據 EventBridge 事件執行自動化。	2017 年 8 月 21 日

變更	描述	發行日期
State Manager 關聯版本控制和一般更新	您現在可以建立不同的 State Manager 關聯版本。每個關聯的配額為 1,000 個版本。您也可以指定關聯的名稱。另外，State Manager 文件已經更新，以解決資訊過時和不一致的問題。如需詳細資訊，請參閱AWS Systems Manager State Manager。	2017 年 8 月 21 日
對Maintenance Windows的變更	Maintenance Windows包含下列的變更或增強功能： • 之前，Maintenance Windows只能藉由使用 Run Command 來執行任務。您現在可以使用 Systems Manager Automation AWS Lambda和 來執行任務 AWS Step Functions。 • 您可以編輯維護時段的目標，指定目標的名稱、說明和擁有者。 • 您可以編輯維護時段中的任務，包括指定新的 SSM 文件，此文件可用於 Run Command 和 Automation 任務。 • 現在支援所有的 Run Command 參數，包括 DocumentHash、DocumentHashType、TimeoutSeconds、Comment 和 NotificationConfig。 • 在嘗試取消目標的登錄時，您現在可以使用 `safe` 旗標。如果開啟，系統會在有任何任務參照該目標時傳回錯誤。 如需詳細資訊，請參閱AWS Systems Manager Maintenance Windows。	2017 年 8 月 16 日
新 Automation 動作：`aws:approve`	這項自動化 Runbook 的新動作，會暫停 Automation 的執行，直到指定的委託人核准或拒絕動作。達到所需的核准數量後，自動化執行會繼續。 如需詳細資訊，請參閱Systems Manager Automation 動作參考。	2017 年 8 月 10 日

變更	描述	發行日期
Automation 不再需要承擔角色	自動化先前會需要您指定服務角色 (或擔任角色) 以使服務有代表您執行動作的許可。由於服務現在的運作是使用呼叫執行的使用者內容，因此自動化已不再需要此角色。 然而，以下情況仍然需要您為自動化指定服務角色： • 當您想要限制使用者的資源許可，但您想要使用者執行需要更高許可的自動化工作流程時。在此案例中，您可以建立具更高許可的服務角色並允許使用者執行工作流程。 • 您預期要執行超過 12 小時的操作會需要服務角色。 如需詳細資訊，請參閱設定自動化。	2017 年 8 月 3 日
組態合規	使用 Amazon EC2 Systems Manager 組態合規 (Configuration Compliance) 功能，來掃描您的受管執行個體機群，以檢查修補程式合規與組態的不一致。您可以從多個 AWS 帳戶 和 收集和彙總資料 AWS 區域，然後深入探討不合規的特定資源。如需詳細資訊，請參閱AWS Systems Manager合規。	2017 年 8 月 8 日
SSM 文件增強功能	SSM 指令和政策文件現在提供跨平台的支援。這代表單一 SSM 文件，可以處理適用於 Windows 和 Linux 作業系統的外掛程式。跨平台支援可讓您整合所管理的文件數量。在使用結構描述 2.2 版或更新版本的 SSM 文件中，提供了跨平台的支援。 使用結構描述 2.0 版或更新版本的 SSM 指令文件，現在可以包含多個相同類型的外掛程式。例如，您可以建立指令文件，來呼叫 `aws:runRunShellScript` 外掛程式多次。 如需結構描述 2.2 版變更的詳細資訊，請參閱 AWS Systems Manager 文件。如需有關 SSM 外掛程式的詳細資訊，請參閱《命令文件外掛程式參考》。	2017 年 7 月 12 日

變更	描述	發行日期
Linux 修補	Patch Manager 現在可以修補下列的 Linux 版本： 64 位元和 32 位元系統 • Amazon Linux 2014.03、2014.09 或更新的版本 • Ubuntu Server 16.04 LTS、14.04 LTS 或 12.04 LTS • Red Hat Enterprise Linux (RHEL) 6.5 或更新版本 僅限 64 位元系統 • Amazon Linux 2015.03、2015.09 或更新的版本 • Red Hat Enterprise Linux (RHEL) 7.x 或更新版本 如需詳細資訊，請參閱AWS Systems Manager Patch Manager。 Note • 若要修補 Linux 執行個體，您的執行個體必須執行 SSM Agent 2.0.834.0 版或更新的版本。如需有關更新代理程式的資訊，請參閱 從主控台執行命令 中標題為範例：更新 SSM Agent 的章節。 • `AWS-ApplyPatchBaseline` SSM 文件被 `AWS-RunPatchBaseline` 文件取代。	2017 年 7 月 6 日

變更	描述	發行日期
資源資料同步	您可以使用 Systems Manager 資源資料同步，將從所有受管執行個體收集到的庫存資料傳送至單一 Amazon Simple Storage Service (Amazon S3) 儲存貯體。然後，資源資料同步會在系統收集新的清查資料時自動更新集中的資料。當所有清查資料存放至目標 S3 儲存貯體後，您就能利用 Amazon Athena 和 Amazon QuickSight 等服務來查詢及分析彙總的資料。如需詳細資訊，請參閱 建立庫存的資源資料同步。如需關於如何使用資源資料同步的範例，請參閱 演練：使用資源資料同步來彙總庫存資料。	2017 年 6 月 29 日
Systems Manager 參數階層	以一般清單的方式來管理數十或數百個 Systems Manager 參數，不僅耗時而且容易出錯。您可以使用參數階層結構，來協助您整理和管理 Systems Manager 參數。階層結構是一種參數名稱，其中包含您使用斜線定義的路徑。以下範例在名稱中使用三個階層來識別下列各項： /Environment/Type of computer/Application/Data `/Dev/DBServer/MySQL/db-string13` 如需詳細資訊，請參閱在 Parameter Store 中使用參數階層。	2017 年 6 月 22 日
SUSE Linux Enterprise Server 的 SSM Agent 支援	您可以在 64 位元 SUSE Linux Enterprise Server (SLES) 上安裝 SSM Agent。如需詳細資訊，請參閱在 Linux 的 EC2 執行個體使用 SSM Agent。	2017 年 6 月 14 日

文件慣用形式

以下列出《AWS Systems Manager 使用者指南》的常見印刷慣例。

本機作業系統或命令列語言的差異範例

我們根據使用者的本機作業系統類型，使用索引標籤顯示不同的命令範例。在 Linux 和 macOS 範例中，我們使用反斜線 (\) 字元將長命令分割為多行。在 Windows Server 範例中，我們使用插入號 (^) 字元將長命令分割為多行。

範例：

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
```

使用者界面中的元素

格式：粗體文字

範例：選擇 File (檔案)、Properties (屬性)。

使用者輸入 (使用者鍵入的文字)

格式：單一空格字體的文字

範例：對於名稱，請輸入 **my-new-resource**。

必要值的預留位置文字

格式：*##*文字

範例：

```
aws ec2 register-image --image-location amzn-s3-demo-bucket/image.manifest.xml
```

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。