사용자 가이드

AWS Systems Manager

AWS Systems Manager: 사용자 가이드

Table of Contents

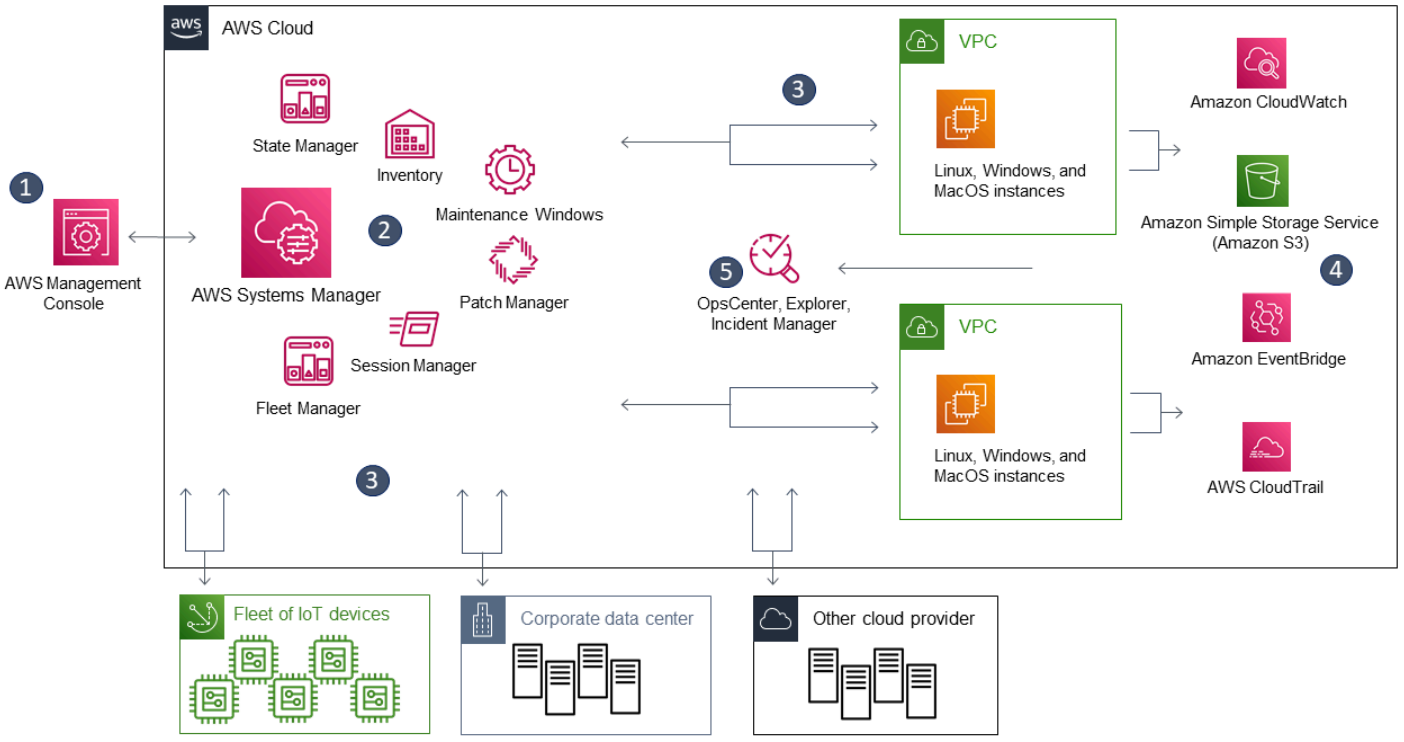
AWS Systems Manager이란?

AWS Systems Manager는 AWS 애플리케이션 및 리소스를 위한 운영 허브이자, 안전한 운영이 대규모로 활성화되는 하이브리드 및 멀티클라우드 환경을 위한 안전한 엔드 투 엔드 관리 솔루션입니다.

Systems Manager의 작동 방식

다음 다이어그램에서는 일부 Systems Manager 기능이 리소스에 대한 작업을 수행하는 방법을 설명합니다. 이 다이어그램에서 모든 기능을 다루지는 않습니다. 각 열거된 상호 작용은 다이어그램 앞에 설명되어 있습니다.

1. Systems Manager 액세스 - Systems Manager 액세스에 대해 사용 가능한 옵션 중 하나를 사용합니다.
2. Systems Manager 기능 선택 - 리소스에 대해 수행할 작업을 수행하는 데 도움이 될 수 있는 기능을 확인합니다. 이 다이어그램은 IT 관리자와 DevOps 인력이 리소스를 구성하고 관리하는 데 사용하는 몇 가지 기능만 보여줍니다.
3. 확인 및 처리 - Systems Manager는 사용자, 그룹 또는 역할에 지정한 작업을 수행하는 데 필요한 AWS Identity and Access Management(IAM) 권한이 있는지 확인합니다. 작업 대상이 관리형 노드인 경우 노드에서 실행 중인 Systems Manager 에이전트(SSM Agent)가 작업을 수행합니다. 다른 유형의 리소스의 경우 Systems Manager는 지정된 작업을 수행하거나 다른 AWS 서비스와(과) 통신하여 Systems Manager를 대신하여 작업을 수행합니다.
4. 보고 - Systems Manager, SSM Agent 및 Systems Manager를 대신하여 작업을 수행하는 기타 AWS 서비스은(는) 상태를 보고합니다. Systems Manager는 다른 AWS 서비스에 상태 세부 정보를 보낼 수 있습니다(구성된 경우).
5. Systems Manager 운영 관리 기능 - 활성화된 경우 Systems Manager 운영 관리 기능(예:Explorer, OpsCenter), Incident Manager는 리소스의 이벤트나 오류에 대응하여 운영 데이터를 집계하거나 아티팩트를 생성합니다. 이러한 아티팩트에는 운영 작업 항목(OpsItems) 및 인시던트가 있습니다. Systems Manager 운영 관리 기능은 애플리케이션 및 리소스에 대한 운영 통찰력과 자동화된 문제 해결 솔루션을 제공하여 문제를 해결하는 데 도움이 됩니다.



Systems Manager 기능

Systems Manager는 기능을 다음 범주로 그룹화합니다. 각 범주 아래에 있는 탭을 선택하여 각 기능에 대해 자세히 알아봅니다.

주제

- 애플리케이션 관리
- 변경 관리
- 노드 관리
- 운영 관리
- Quick Setup
- 공유 리소스

애플리케이션 관리

Application Manager

Application Manager는 DevOps 엔지니어가 애플리케이션 및 클러스터 맥락에서 AWS 리소스와 관련된 문제를 조사하고 해결하는 데 도움이 됩니다. Application Manager에서 애플리케이션은 하나

의 단위로 작동하려는 AWS 리소스의 논리적 그룹입니다. 이 논리 그룹은 몇 가지 예를 들면 다양한 버전의 애플리케이션, 운영자의 소유권 경계 또는 개발자 환경을 나타낼 수 있습니다. Application Manager의 컨테이너 클러스터 지원에는 Amazon Elastic Kubernetes Service(Amazon EKS) 및 Amazon Elastic Container Service(Amazon ECS) 클러스터가 모두 포함됩니다. Application Manager은(는) 여러 AWS 서비스 및 Systems Manager 기능으로부터의 운영 정보를 단일 AWS Management Console(으)로 집계합니다.

AppConfig

AppConfig를 사용하여 애플리케이션 구성 및 기능 플래그를 생성, 관리 및 배포할 수 있습니다. AppConfig는 모든 규모의 애플리케이션에 대한 제어된 배포를 지원합니다. Amazon EC2 인스턴스, AWS Lambda 컨테이너, 모바일 애플리케이션 또는 엣지 디바이스에서 호스팅되는 애플리케이션에서 AppConfig를 사용할 수 있습니다. 애플리케이션 구성을 배포할 때 오류를 방지하기 위해 AppConfig에는 유효성 검사기가 포함됩니다. 검증기는 구문 또는 의미 검사를 제공하여 배포하려는 구성이 의도한 대로 작동하는지 확인합니다. 구성 배포 중에 AppConfig는 애플리케이션을 모니터링하여 배포가 성공적인지 확인합니다. 시스템에 오류가 발생하거나 배포가 경보를 호출하면 애플리케이션 사용자에 대한 영향을 최소화하기 위해 AppConfig가 변경 사항을 롤백합니다.

파라미터 스토어

Parameter Store는 구성 데이터 및 암호 관리를 위한 안전한 계층적 스토리지를 제공합니다. 암호, 데이터베이스 문자열, Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 ID, Amazon Machine Image(AMI) ID 및 라이선스 코드와 같은 데이터를 파라미터 값으로 저장할 수 있습니다. 값을 일반 텍스트 또는 암호화된 데이터로 저장할 수 있습니다. 그런 다음 파라미터를 생성할 때 지정한 고유한 이름을 사용하여 값을 참조할 수 있습니다.

변경 관리

Change Manager

Change Manager는 애플리케이션 구성 및 인프라에 대한 운영 변경을 요청, 승인, 구현 및 보고하기 위한 엔터프라이즈 변경 관리 프레임워크입니다. 하나의 위임된 관리자 계정에서 AWS Organizations를 사용하면 여러 AWS 리전에 있는 여러 AWS 계정의 변경사항을 관리할 수 있습니다. 또는 로컬 계정을 사용하여 하나의 AWS 계정에 대한 변경 사항을 관리할 수 있습니다. AWS 리소스와 온프레미스 리소스 모두에 대한 변경 사항을 관리하려면 Change Manager를 사용합니다.

Automation

Automation으로 일반적인 유지 관리 및 배포 태스크를 자동화합니다. Automation을 사용해 Amazon Machine Images(AMIs)를 생성 및 업데이트하고, 드라이버 및 에이전트 업데이트를 적용

하고, Windows Server 인스턴스에서 암호를 재설정하고, Linux 인스턴스에서 SSH 키를 재설정하고, OS 패치 또는 애플리케이션 업데이트를 적용할 수 있습니다.

Change Calendar

Change Calendar는 지정한 작업(예: Systems Manager Automation 실행서에서)이 AWS 계정에서 수행되거나 수행되지 않을 때 날짜 및 시간 범위를 설정하는 데 도움이 됩니다. Change Calendar에서는 이러한 범위를 이벤트라고 합니다. Change Calendar 항목을 생성하면 `ChangeCalendar` 유형의 Systems Manager 문서가 생성됩니다. Change Calendar에서 문서는 iCalendar 2.0 데이터를 일반 텍스트 형식으로 저장합니다. Change Calendar 항목에 추가하는 이벤트는 문서의 일부가 됩니다. Change Calendar 인터페이스에서 수동으로 이벤트를 추가하거나 `.ics` 파일을 사용하여 지원되는 서드 파티 캘린더에서 이벤트를 가져올 수 있습니다.

유지 관리 기간

업무상 중요한 태스크를 중단하지 않고 패치 및 업데이트 설치와 같은 관리 태스크를 실행하려면 Maintenance Windows를 사용하여 관리형 인스턴스에 대한 반복 일정을 설정합니다.

노드 관리

관리형 노드는 하이브리드 및 멀티클라우드 환경에서 Systems Manager와 함께 사용하도록 구성된 모든 시스템입니다.

Compliance

Compliance를 사용하여 관리형 노드 플릿에 대해 패치 규정 준수 및 구성 일관성을 스캔할 수 있습니다. 여러 AWS 계정 및 AWS 리전의 데이터를 수집하여 집계한 후 규정을 준수하지 않는 특정 리소스로 드릴다운할 수 있습니다. 기본적으로 Compliance는 Patch Manager 패치 및 State Manager 연결에 대한 규정 준수 데이터를 표시합니다. 또한 IT 또는 비즈니스 요구 사항에 따라 서비스를 사용자 지정하고 자체 규정 준수 유형을 만들 수도 있습니다.

Fleet Manager

Fleet Manager는 노드를 원격으로 관리하는 데 도움이 되는 통합 사용자 인터페이스(UI) 환경입니다. Fleet Manager를 사용하면 하나의 콘솔에서 전체 플릿의 상태 및 성능 상태를 볼 수 있습니다. 또한 개별 디바이스 및 인스턴스에서 데이터를 수집하여 콘솔에서 일반적인 문제 해결 및 관리 태스크를 수행할 수 있습니다. 여기에는 디렉터리 및 파일 내용 보기, Windows 레지스트리 관리, 운영 체제 사용자 관리 등이 포함됩니다.

Inventory

Inventory는 관리형 노드에서 소프트웨어 인벤토리를 수집하는 과정을 자동화합니다. Inventory를 사용하여 애플리케이션과 파일, 구성 요소, 패치 등에 대한 메타데이터를 수집할 수 있습니다.

세션 관리자

Session Manager를 사용하여 대화형 원클릭 브라우저 기반 셸 또는 AWS CLI를 통해 엣지 디바이스와 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스를 관리할 수 있습니다. Session Manager는 인바운드 포트를 열고, Bastion Host를 유지하고, SSH 키를 관리할 필요 없이 보안성과 감사 가능성을 갖춘 엣지 디바이스 및 인스턴스 관리 기능을 제공합니다. 또한 Session Manager를 통해 인스턴스에 대한 제어된 액세스를 요구하는 회사 정책, 엄격한 보안 관행을 손쉽게 준수하고, 인스턴스 액세스 세부 정보가 포함된 완전히 감사 가능한 로그를 제공하는 동시에 최종 사용자에게 엣지 디바이스 및 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에 대한 간단한 원 클릭 크로스 플랫폼 액세스를 제공합니다. Session Manager를 사용하려면 고급 인스턴스 티어를 사용해야 합니다. 자세한 내용은 고급 인스턴스 티어 설정 섹션을 참조하세요.

Run Command

Run Command를 사용하여 관리형 노드의 구성을 원격으로 안전하게 대규모로 관리합니다. Run Command를 사용하여 수십 개나 수백 개의 대상 관리형 노드 집합에서 애플리케이션 업데이트나 Linux 셸 스크립트 및 Windows PowerShell 명령 실행과 같은 온디맨드 변경 사항을 수행합니다.

상태 관리자

State Manager를 사용하여 관리형 노드를 정의된 상태로 유지하는 과정을 자동화합니다. State Manager를 사용하여 관리형 노드가 시작 시 특정 소프트웨어로 부트스트랩되거나, Windows 도메인에 연결되거나(Windows Server 노드만 해당), 특정 소프트웨어 업데이트로 패치되도록 할 수 있습니다.

패치 관리자

Patch Manager를 사용하여 보안 관련 및 기타 유형의 업데이트로 관리형 노드를 패치하는 프로세스를 자동화합니다. Patch Manager를 사용하면 운영 체제와 애플리케이션 모두를 패치할 수 있습니다. (Windows Server에서 애플리케이션 지원은 Microsoft에서 릴리스한 애플리케이션의 업데이트로 제한됩니다.)

이 기능을 사용하면 누락된 패치가 있는지 관리형 노드를 스캔하고 태그를 사용하여 누락된 패치를 개별적으로 또는 대규모 관리형 노드 그룹에 적용할 수 있습니다. Patch Manager는 패치 기준을 사용합니다. 여기에는 릴리스 후 며칠 이내에 패치를 자동 승인하는 규칙과 승인 및 거부된 패치 목록이 포함될 수 있습니다. Systems Manager 유지 관리 기간 태스크로 실행되도록 패치를 예약하여 정기적으로 패치를 설치하거나 언제든지 온디맨드로 관리형 노드를 패치할 수 있습니다.

Linux 운영 체제에서는 패치 기준의 일부로서 패칭 작업에 사용해야 하는 리포지토리를 정의할 수 있습니다. 그러면 관리형 노드에 어떤 리포지토리가 구성되었든 신뢰하는 리포지토리에서만 업데이트를 설치할 수 있습니다. Linux의 경우 운영 체제 보안 업데이트로 분류된 것들뿐만 아니라 관리형 노드 상의 모든 패키지를 업데이트할 수 있습니다. 선택한 S3 버킷에 전송되는 패치 보고서를 생성할 수도 있습니다. 단일 관리형 노드의 경우 보고서에 노드에 대한 모든 패치의 세부 정보가 포함됩니다. 모든 관리형 노드에 대한 보고서의 경우 누락된 패치 수에 대한 요약만 제공됩니다.

Distributor

Distributor를 사용하여 패키지를 생성하고 관리형 노드에 배포합니다. Distributor를 사용하면 자체 소프트웨어를 패키징하거나, AmazonCloudWatchAgent와 같이 AWS에서 제공한 에이전트 소프트웨어 패키지를 찾아서 Systems Manager 관리형 노드에 설치할 수 있습니다. 패키지를 처음 설치한 후 Distributor를 사용하여 새 패키지 버전을 제거하고 새 패키지를 다시 설치하거나 새 파일이나 변경된 파일을 추가하는 인플레이스 업데이트를 수행할 수 있습니다. Distributor는 소프트웨어 패키지와 같은 리소스를 Systems Manager 관리형 노드에 게시합니다.

Hybrid Activations

하이브리드 및 멀티클라우드 환경에서 비 EC2 시스템을 관리형 노드로 설정하려면 하이브리드 정품 인증을 생성합니다. 정품 인증을 마치면 정품 인증 코드와 ID를 받게 됩니다. Amazon Elastic Compute Cloud(Amazon EC2) 액세스 ID 및 보안 암호 키와 같은 기능을 수행하는 이 코드와 ID 조합으로 관리형 인스턴스에서 Systems Manager 서비스에 안전하게 액세스할 수 있습니다.

Systems Manager를 사용하여 관리하려는 경우 엣지 디바이스에 대한 활성화를 만들 수도 있습니다.

운영 관리

Incident Manager

Incident Manager는 사용자가 AWS 호스팅 애플리케이션에 영향을 미치는 인시던트를 완화하고 복구하는 데 도움이 되는 인시던트 관리 콘솔입니다.

Incident Manager는 대응자에게 영향을 알리고, 관련 문제 해결 데이터를 강조 표시하고, 서비스를 백업 및 실행하기 위한 협업 도구를 제공하여 인시던트 해결을 개선합니다. Incident Manager는 또한 대응 계획을 자동화하고 대응 팀 에스컬레이션을 허용합니다.

Explorer

Explorer는 AWS 리소스에 대한 정보를 보고하는 사용자 지정 가능한 운영 대시보드입니다. Explorer에는 AWS 계정 및 AWS 리전의 운영 데이터(OpsData)에 대한 집계 보기가 표시됩니다.

Explorer에서 OpsData에는 Amazon EC2 인스턴스에 대한 메타데이터, 패치 규정 준수 세부 정보 및 운영 작업 항목(OpsItems)이 포함됩니다. Explorer은 사업부 또는 애플리케이션에 OpsItems가 배포되는 방식, 시간 경과에 따른 추세 및 범주별 차이점에 대한 컨텍스트를 제공합니다. Explorer에서 정보를 그룹화 및 필터링하여 사용자와 관련이 있고 작업이 필요한 항목에 초점을 맞출 수 있습니다. 우선순위가 높은 문제를 식별하면 Systems Manager의 기능인 OpsCenter를 사용하여 Automation 실행서를 실행하고 이러한 문제를 해결할 수 있습니다.

OpsCenter

OpsCenter는 운영 엔지니어와 IT 전문가가 AWS 리소스와 관련된 운영 작업 항목(OpsItems)을 보고, 조사하고, 해결할 수 있는 중앙 위치를 제공합니다. OpsCenter는 AWS 리소스에 영향을 미치는 문제의 평균 해결 시간을 단축하도록 설계되었습니다. 이 Systems Manager는 각 OpsItem, 관련 OpsItems 및 관련 리소스에 대한 컨텍스트 조사 데이터를 제공하면서 서비스 전반에 걸쳐 OpsItems를 집계하고 표준화합니다. 또한 OpsCenter는 문제를 해결하는 데 사용할 수 있는 Systems Manager Automation 실행서를 제공합니다. 각 OpsItem에 대해 검색 가능한 사용자 지정 데이터를 지정할 수 있습니다. 상태 및 소스별로 OpsItems에 대한 자동 생성 요약 보고서를 볼 수도 있습니다.

CloudWatch Dashboards

Amazon CloudWatch 대시보드는 CloudWatch 콘솔에서 사용자 지정이 가능한 페이지로, 다른 리전에 분산된 리소스를 비롯하여 단일 보기에서 리소스를 모니터링하는 데 사용할 수 있습니다. CloudWatch 대시보드를 사용하면 AWS 리소스에 대한 지표 및 경보의 사용자 지정 보기를 생성할 수 있습니다.

Quick Setup

Quick Setup을(를) 사용하여 권장되는 모범 사례에 따라 자주 사용하는 AWS 서비스 및 기능을 구성합니다. Quick Setup을 개별 AWS 계정에 사용하거나 AWS Organizations와 통합하여 여러 AWS 계정 및 AWS 리전에서 사용할 수 있습니다. Quick Setup은 공통 또는 권장 태스크를 자동화하여 Systems Manager를 포함한 서비스 설정을 간소화합니다. 이러한 태스크에는 필수 AWS Identity and Access Management(IAM) 인스턴스 프로파일 역할 생성과 정기 패치 검사 및 인벤토리 수집과 같은 운영 모범 사례 설정이 포함됩니다.

공유 리소스

Documents

Systems Manager 문서(SSM 문서)는 Systems Manager가 수행하는 작업을 정의합니다. SSM 문서 유형에는 State Manager와 Run Command에서 사용하는 Command 문서와 Systems Manager

Automation에서 사용하는 Automation 실행서가 포함됩니다. Systems Manager에는 런타임에 파라미터를 지정하여 사용할 수 있는 사전 구성 문서가 수십 개 포함되어 있습니다. 문서는 JSON 또는 YAML로 표현 가능하며 사용자가 지정하는 단계와 파라미터를 포함합니다.

Systems Manager 액세스

다음 방법 중 하나를 사용하여 Systems Manager를 사용할 수 있습니다.

Systems Manager 콘솔

Systems Manager 콘솔은 Systems Manager에 액세스하고 사용하기 위한 브라우저 기반 인터페이스입니다.

AWS IoT Greengrass V2 콘솔

Greengrass 콘솔의 AWS IoT Greengrass를 위해 구성된 엣지 디바이스를 확인하고 관리할 수 있습니다.

AWS 명령줄 도구

AWS 명령줄 도구를 통해 시스템 명령줄에서 명령을 실행하여 Systems Manager 및 기타 AWS 태스크를 수행할 수 있습니다. 도구는 Linux, macOS 및 Windows에서 지원됩니다. AWS Command Line Interface(AWS CLI)를 사용하는 것이 콘솔을 사용하는 것보다 더 빠르고 편리할 수 있습니다. AWS 작업을 수행하는 스크립트를 작성할 때도 명령줄 도구가 유용합니다.

AWS에서는 AWS Command Line Interface와 AWS Tools for Windows PowerShell이라는 두 가지 명령줄 도구 집합을 제공합니다. AWS CLI 설치 및 사용에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서를 참조하세요. Tools for Windows PowerShell 도구 설치 및 사용에 대한 자세한 내용은 AWS Tools for Windows PowerShell User Guide를 참조하세요.

> **Note**
>
> Windows Server 인스턴스에서 특정 SSM 문서(예: 레거시 `AWS-ApplyPatchBaseline` 문서)를 실행하려면 Windows PowerShell 3.0 이상이 필요합니다. Windows Server 인스턴스가 Windows Management Framework 3.0 이상을 실행 중인지 확인합니다. 프레임워크에 Windows PowerShell이 포함되어 있습니다.

AWS SDK

AWS에서는 다양한 프로그래밍 언어 및 플랫폼(예: Java, Python, Ruby, .NET, iOS 및 Android 등)을 위한 라이브러리와 샘플 코드로 구성된 소프트웨어 개발 키트(SDK)를 제공합니다. SDK는 Systems Manager에 대한 프로그래밍 방식의 액세스 권한을 부여하는 편리한 방법을 제공합니다. 다운로드 및 설치 방법을 비롯하여 AWS SDK에 대한 자세한 내용은 Amazon Web Services용 도구 페이지를 참조하세요.

Systems Manager 서비스 이름 기록

AWS Systems Manager(Systems Manager)는 이전에는 "Amazon Simple Systems Manager (SSM)" 및 "Amazon EC2 Systems Manager (SSM)"로 알려졌습니다. 서비스 이름의 기존 약어인 “SSM”의 경우 여전히 몇 개의 다른 서비스 콘솔을 포함하여 다양한 AWS 리소스에서 반영됩니다. 다음은 일부 예입니다.

- Systems Manager Agent: SSM Agent
- Systems Manager 파라미터: SSM 파라미터
- Systems Manager 서비스 엔드포인트: `ssm.region.amazonaws.com`
- AWS CloudFormation 리소스 유형: `AWS::SSM::Document`
- AWS Config 규칙 식별자: `EC2_INSTANCE_MANAGED_BY_SSM`
- AWS Command Line Interface(AWS CLI) 명령: `aws ssm describe-patch-baselines`
- AWS Identity and Access Management(IAM) 관리형 정책 이름: `AmazonSSMReadOnlyAccess`
- Systems Manager 리소스 ARN: `arn:aws:ssm:region:account-id:patchbaseline/pb-07d8884178EXAMPLE`

지원되는 AWS 리전

Systems Manager는 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 나열된 AWS 리전에서 사용할 수 있습니다. Systems Manager 구성 프로세스를 시작하기 전에 서비스를 사용하려는 각 AWS 리전에서 해당 서비스 사용이 가능한지 확인하는 것이 좋습니다.

하이브리드 및 멀티클라우드 환경에 있는 비 EC2 시스템의 경우 데이터 센터 또는 컴퓨팅 환경과 가장 가까운 리전을 선택하는 것이 좋습니다.

지원되는 운영 체제 및 시스템 유형

Systems Manager로 작업하기 전에 OS(운영 체제), OS 버전 및 시스템 유형이 관리형 노드로 지원되는지 확인합니다.

주제

Systems Manager가 지원되는 운영 체제

다음 섹션에는 Systems Manager에서 지원되는 OS 및 OS 버전이 나열되어 있습니다.

> **Note**
>
> 관리 및 구성을 계획하는 경우 AWS IoT Greengrass 코어 디바이스를 Systems Manager를 사용하여 구성하세요. 이러한 장치는 AWS IoT Greengrass의 요구 사항을 충족해야 합니다. 자세한 내용은 AWS IoT Greengrass 개발자 안내서의 AWS IoT Greengrass Version 2 코어 디바이스 설정 섹션을 참조하세요.
>
> AWS IoT 및 비-AWS 엣지 디바이스를 관리하고 구성하려는 계획이 있다면, 이러한 디바이스는 여기 목록의 요구 사항을 충족시켜야 하며 Systems Manager용 온프레미스 관리형 노드로 구성되어야 합니다. 자세한 내용은 Systems Manager를 통한 엣지 디바이스 관리 단원을 참조하십시오.

> **Important**
>
> Systems Manager의 기능인 Patch Manager에서는 이 주제에 나열된 일부 OS 버전이 지원되지 않을 수 있습니다. Patch Manager에서 지원하는 OS 버전 목록은 Patch Manager 필수 조건 섹션을 참조하세요.

운영 체제 유형

- Linux
- macOS(Amazon EC2 인스턴스만 해당)
- Raspberry Pi OS(구 Raspbian)

- Windows Server

Linux

AlmaLinux

버전	x86	x86_64	ARM64
8.3~8.9		✓	✓
9.0~9.2		✓	✓

Amazon Linux 1

버전	x86	x86_64	ARM64
2012.03~2018.03	✓	✓	

> **Note**
>
> Amazon Linux 1은 2015.03 버전부터 x86_64 버전으로 출시됩니다.
> Amazon Linux 1은 AWS 뉴스 블로그의 Amazon Linux AMI 수명 종료 업데이트에서 발표한 대로 2020년 12월 31일 표준 지원이 종료되고 2023년 12월 31일 수명 주기가 종료되었습니다. AWS는 더 이상 이 운영 체제에 대한 Amazon Machine Images(AMIs)를 제공하지 않습니다. 하지만 AWS Systems Manager은 기존 Amazon Linux 1 인스턴스에 대한 지원을 계속 제공합니다.

Amazon Linux 2

버전	x86	x86_64	ARM64
2.0 이상 버전		✓	✓

Amazon Linux 2023

버전	x86	x86_64	ARM64
2023.0.20230315.0 이상 모든 버전		✓	✓

Bottlerocket

버전	x86_64	ARM64
1.0.0 이상 버전	✓	✓

CentOS

버전	x86	x86_64	ARM64
6.x[1]	✓	✓	
7.1 이상 7.x 버전		✓	✓
8.0~8.5		✓	✓

[1]이러한 버전을 사용하려면 SSM Agent의 3.0.x 버전을 사용해야 합니다. SSM Agent의 최신 3.0.x 버전을 사용하는 것이 좋습니다. SSM Agent 이상 버전(3.1 이상)은 지원되지 않습니다.

CentOS Stream

버전	x86	x86_64	ARM64
8		✓	✓

Debian 서버

버전	x86	x86_64	ARM64
Jessie(8)		✓	

버전	x86	x86_64	ARM64
Stretch(9)		✓	✓
Buster(10)		✓	✓
Bullseye(11)		✓	✓
Bookworm(12)		✓	✓

Oracle Linux

버전	x86	x86_64	ARM64
7.5~7.8		✓	
8.1~8.9		✓	
9.0~9.2		✓	

Red Hat Enterprise Linux (RHEL)

버전	x86	x86_64	ARM64
6.x[1]	✓	✓	
7.0~7.5		✓	
7.6~8.9		✓	✓
9.0~9.3		✓	✓

[1]이러한 버전을 사용하려면 SSM Agent의 3.0.x 버전을 사용해야 합니다. SSM Agent의 최신 3.0.x 버전을 사용하는 것이 좋습니다. SSM Agent 이상 버전(3.1 이상)은 지원되지 않습니다.

Rocky Linux

버전	x86	x86_64	ARM64
8.4~8.9		✓	✓
9.0~9.2		✓	✓

SUSE Linux Enterprise Server (SLES)

버전	x86	x86_64	ARM64
12 이상 12.x 버전		✓	
15 이상 15.x 버전		✓	✓

Ubuntu 서버

버전	x86	x86_64	ARM64
12.04 LTS 및 14.04 LTS	✓	✓	
16.04 LTS 및 18.04 LTS		✓	✓
20.04 LTS 및 20.10 STR		✓	✓
22.04 LTS		✓	✓
23.04		✓	✓

macOS(Amazon EC2 인스턴스만 해당)

버전	x86	x86_64	Mac with Apple silicon
10.14.x(Mojave)		✓	

버전	x86	x86_64	Mac with Apple silicon
10.15.x(Catalina)		✓	
11.x(Big Sur)		✓	✓
12.x(몬터레이)		✓	✓
13.x(Ventura)		✓	✓
14.x(Sonoma)		✓	✓

> **Note**
>
> 일부 AWS 리전에서는 macOS가 지원되지 않습니다. macOS용 Amazon EC2 지원에 대한 자세한 내용은 Amazon EC2 사용 설명서의 Amazon EC2 Mac 인스턴스를 참조하세요.

Raspberry Pi OS(구 Raspbian)

버전	ARM32
8(Jessie)	✓
9(Stretch)	✓

추가 정보

- AWS Systems Manager를 사용하여 Raspberry Pi 디바이스 관리

Windows Server

SSM Agent는 Windows Server 인스턴스(예: 레거시 `AWS-ApplyPatchBaseline` 문서)에서 특정 AWS Systems Manager 문서(SSM 문서)를 실행하려면 Windows PowerShell 3.0 이상이 필요합니다. Windows Server 인스턴스가 Windows Management Framework 3.0 이상을 실행 중인지 확인합니다. 이 프레임워크에는 Windows PowerShell이 포함되어 있습니다. 자세한 내용은 Windows Management Framework 3.0 단원을 참조하십시오.

버전	x86	x86_64	ARM64
2008[1]	✓	✓	
2008 R2[1]		✓	
2012 및 2012 R2		✓	
2016		✓	
2019		✓	
2022		✓	

[1] 2020년 1월 14일부로 Windows Server 2008은 Microsoft의 기능 또는 보안 업데이트에 대해 더 이상 지원되지 않습니다. Windows Server 2008 및 2008 R2용 레거시 Amazon Machine Images(AMIs)에는 여전히 SSM Agent 버전 2가 사전 설치된 상태로 포함되어 있지만 Systems Manager는 더 이상 2008 버전을 공식적으로 지원하지 않으며 이러한 Windows Server 버전용 에이전트를 더 이상 업데이트하지 않습니다. 또한 SSM Agent 버전 3은 Windows Server 2008 및 2008 R2의 일부 작업과 호환되지 않을 수 있습니다. Windows Server 2008 버전에 대해 공식적으로 지원되는 최종 SSM Agent 버전은 2.3.1644.0입니다.

하이브리드 및 멀티클라우드 환경에서 지원되는 시스템 유형

Systems Manager에서는 수많은 시스템 유형이 관리형 노드로 지원됩니다. 관리형 노드는 Systems Manager로 작업하도록 구성된 시스템입니다.

이 사용 설명서에서는 하이브리드 및 멀티클라우드라는 용어를 사용하여 다음과 같은 시스템 유형의 조합이 포함된 환경을 나타냅니다.

- Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스
- 자체 프레미스의 서버(온프레미스 서버)
- AWS IoT Greengrass 코어 디바이스
- AWS IoT 및 비 AWS 엣지 디바이스
- 다른 클라우드 환경의 VM을 포함한 가상 머신

하이브리드 및 멀티클라우드 환경의 AWS 지원에 대한 자세한 내용은 하이브리드 및 멀티클라우드용 AWS 솔루션을 참조하세요.

AWS SDK를 사용하여 Systems Manager 사용

다양한 프로그래밍 언어에 대해 AWS 소프트웨어 개발 키트(SDK)을 사용할 수 있습니다. 각 SDK는 개발자가 선호하는 언어로 애플리케이션을 쉽게 구축할 수 있도록 하는 API, 코드 예시 및 설명서를 제공합니다.

SDK 설명서	코드 예시
AWS SDK for C++	AWS SDK for C++ 코드 예시
AWS CLI	AWS CLI 코드 예시
AWS SDK for Go	AWS SDK for Go 코드 예시
AWS SDK for Java	AWS SDK for Java 코드 예시
AWS SDK for JavaScript	AWS SDK for JavaScript 코드 예시
AWS SDK for Kotlin	AWS SDK for Kotlin 코드 예시
AWS SDK for .NET	AWS SDK for .NET 코드 예시
AWS SDK for PHP	AWS SDK for PHP 코드 예시
AWS Tools for PowerShell	Tools for PowerShell 코드 예시
AWS SDK for Python (Boto3)	AWS SDK for Python (Boto3) 코드 예시
AWS SDK for Ruby	AWS SDK for Ruby 코드 예시
AWS SDK for Rust	AWS SDK for Rust 코드 예시
AWS SDK for SAP ABAP	AWS SDK for SAP ABAP 코드 예시
AWS SDK for Swift	AWS SDK for Swift 코드 예시

가용성 예제

필요한 예제를 찾을 수 없습니까? 이 페이지 하단의 피드백 제공 링크를 사용하여 코드 예시를 요청하세요.

AWS Systems Manager 설정

이 섹션의 작업을 완료하여 AWS Systems Manager에 대한 역할, 사용자 계정, 권한 및 초기 리소스를 설정하고 구성합니다. 이 섹션에서 설명하는 작업은 일반적으로 AWS 계정 및 시스템 관리자가 수행합니다. 이러한 단계를 완료한 후에는 조직의 사용자가 Systems Manager를 사용하여 관리형 노드를 구성, 관리 및 액세스할 수 있습니다. 관리형 노드는 하이브리드 및 멀티클라우드 환경에서 Systems Manager와 함께 사용하도록 구성된 모든 시스템입니다.

> **Note**
>
> 하이브리드 및 멀티클라우드 환경에서 Amazon EC2 인스턴스 및 자체 컴퓨팅 리소스를 사용할 계획인 경우, EC2 인스턴스에 Systems Manager 사용의 단계를 따릅니다. 이 주제에서는 EC2 인스턴스 및 비 EC2 시스템에 대한 Systems Manager 설정을 완료하는 단계를 최상의 순서로 설명합니다.

다른 AWS 서비스를 이미 사용하는 경우 이러한 단계 중 일부를 완료한 것입니다. 하지만 다른 단계는 Systems Manager에 특정합니다. 따라서 이 섹션 전체를 검토하여 모든 Systems Manager 기능을 사용할 준비가 되었는지 확인하는 것이 좋습니다.

주제

- EC2 인스턴스에 Systems Manager 사용
- 하이브리드 및 멀티클라우드 환경에서 Systems Manager 사용하기
- Systems Manager를 통한 엣지 디바이스 관리
- Systems Manager에 대한 AWS Organizations 위임 관리자 생성
- AWS Systems Manager에 대한 일반 설정

EC2 인스턴스에 Systems Manager 사용

이 섹션의 태스크를 수행하여 AWS Systems Manager에 대한 역할, 권한 및 초기 리소스를 설정하고 구성합니다. 이 섹션에서 설명하는 작업은 일반적으로 AWS 계정 및 시스템 관리자가 수행합니다. 이러한 단계를 완료한 후에는 조직의 사용자가 Systems Manager를 사용하여 자신의 계정에서 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스를 구성, 관리 및 액세스할 수 있습니다.

> **Note**
>
> Systems Manager를 사용하여 온프레미스 시스템을 관리하고 구성하려면 하이브리드 및 멀티 클라우드 환경에서 Systems Manager 사용하기의 설정 단계를 따릅니다. 하이브리드 및 멀티 클라우드 환경에서 Amazon EC2 인스턴스 및 비 EC2 시스템을 모두 사용할 계획인 경우 먼저 여기의 단계를 따릅니다. 이 섹션에서는 Systems Manager 작업에서 사용할 역할, 사용자, 권한 및 초기 리소스를 구성하는 권장되는 순서의 단계를 제공합니다.

다른 AWS 서비스를 이미 사용하는 경우 이러한 단계 중 일부를 완료한 것입니다. 하지만 다른 단계는 Systems Manager에 특정합니다. 따라서 이 섹션 전체를 검토하여 모든 Systems Manager 기능을 사용할 준비가 되었는지 확인하는 것이 좋습니다.

내용

Systems Manager에 필요한 인스턴스 권한 구성

AWS Systems Manager는 기본적으로 인스턴스에서 작업을 수행할 권한이 없습니다. AWS Identity and Access Management(IAM) 역할을 사용하여 계정 수준에서 또는 인스턴스 프로파일을 사용하여 인스턴스 수준에서 인스턴스 권한을 제공할 수 있습니다. 사용 사례에서 허용하는 경우 기본 호스트 관리 구성을 사용하여 계정 수준에서 액세스 권한을 부여하는 것이 좋습니다.

EC2 인스턴스 권한에 대한 권장 구성

기본 호스트 관리 구성을 통해 Systems Manager는 Amazon EC2 인스턴스를 자동으로 관리할 수 있습니다. 이 설정을 켜면 AWS 리전에서 인스턴스 메타데이터 서비스 버전 2(IMDSv2)를 사용하는 모든 인스턴스와 SSM Agent 버전 3.2.582.0 이상이 설치된 AWS 계정이 자동으로 관리형 인스턴스가 됩니다. 기본 호스트 관리 구성은 인스턴스 메타데이터 서비스 버전 1을 지원하지 않습니다. IMDSv2로 전환에 대한 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스 메타데이터 서비스 버전 2 사용으로 전환을 참조하세요. 인스턴스에 설치된 SSM Agent 버전 확인에 대한 자세한 내용은 SSM Agent 버전 번호 확인 섹션을 참조하세요. SSM Agent 업데이트에 대한 자세한 내용은 SSM Agent 자동 업데이트 섹션을 참조하세요. 관리형 인스턴스의 이점은 다음과 같습니다.

- Session Manager를 사용하여 안전하게 인스턴스에 연결합니다.
- Patch Manager를 사용하여 자동 패치 스캔을 수행합니다.

- Systems Manager Inventory를 사용하여 인스턴스에 대한 세부 정보를 봅니다.
- Fleet Manager를 사용하여 인스턴스를 추적하고 관리합니다.
- SSM Agent를 자동으로 최신 상태로 유지합니다.

Fleet Manager, Inventory, Patch Manager 및 Session Manager는 AWS Systems Manager의 기능입니다.

기본 호스트 관리 구성을 사용하면 인스턴스 프로파일을 사용하지 않고 인스턴스를 관리할 수 있으며 Systems Manager에 리전 및 계정의 모든 인스턴스를 관리할 수 있는 권한이 있는지 확인할 수 있습니다. 제공된 권한이 사용 사례에 충분하지 않은 경우 기본 호스트 관리 구성에서 생성된 기본 IAM 역할에 정책을 추가할 수도 있습니다. 또는 기본 IAM 역할에서 제공하는 모든 기능에 대한 권한이 필요하지 않은 경우 사용자 지정 역할과 정책을 직접 생성할 수 있습니다. 기본 호스트 관리 구성에 대해 선택한 IAM 역할의 모든 변경 사항은 리전 및 계정의 모든 관리형 Amazon EC2 인스턴스에 적용됩니다. 기본 호스트 관리 구성에서 사용하는 정책에 대한 자세한 내용은 AWS 관리형 정책: AmazonSSMManagedEC2InstanceDefaultPolicy 섹션을 참조하세요. 기본 호스트 관리 구성에 대한 자세한 내용은 기본 호스트 관리 구성 설정 관리 섹션을 참조하세요.

Important

기본 호스트 관리 구성을 사용하여 등록된 인스턴스는 `/lib/amazon/ssm` 또는 `C:\ProgramData\Amazon` 디렉터리에 로컬로 등록 정보를 저장합니다. 이러한 디렉토리 또는 해당 파일을 제거하면 인스턴스가 기본 호스트 관리 구성을 사용하여 시스템 관리자에 연결하는 데 필요한 보안 인증을 획득하지 못하게 됩니다. 이러한 경우 인스턴스 프로파일을 사용하여 인스턴스에 필요한 권한을 제공하거나 인스턴스를 다시 생성해야 합니다.

Note

이 절차는 관리자만 수행할 수 있습니다. 개인이 기본 호스트 관리 구성을 구성하거나 수정할 수 있도록 허용할 때 최소 권한 액세스를 구현합니다. Amazon EC2 인스턴스를 자동으로 관리하려는 각 AWS 리전에서 기본 호스트 관리 구성을 켜야 합니다.

기본 호스트 관리 구성 설정 켜기

Fleet Manager 콘솔에서 기본 호스트 관리 구성을 켤 수 있습니다. AWS Management Console 또는 기본 명령줄 도구를 사용하여 이 절차를 완료하려면 GetServiceSetting,

ResetServiceSetting 및 UpdateServiceSetting API 작업에 대한 권한이 있어야 합니다. 또한 AWSSystemsManagerDefaultEC2InstanceManagementRole IAM 역할에 대한 iam:PassRole 권한이 있어야 합니다. 다음은 예제 정책입니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

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

시작하기 전에 Amazon EC2 인스턴스에 연결된 인스턴스 프로파일이 있는 경우ssm:UpdateInstanceInformation 작업을 허용하는 모든 권한을 제거합니다. SSM Agent는 기본 호스트 관리 구성 권한을 사용하기 전에 인스턴스 프로파일 권한 사용을 시도합니다. 인스턴스 프로파일에서 ssm:UpdateInstanceInformation 작업을 허용하면 인스턴스가 기본 호스트 관리 구성 권한을 사용하지 않습니다.

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 계정 관리 드롭다운에서 기본 호스트 관리 구성 구성하기를 선택합니다.
4. 기본 호스트 관리 구성 활성화를 켭니다.
5. 인스턴스에 대해 Systems Manager 기능을 활성화하는 데 사용되는 IAM 역할을 선택합니다. 기본 호스트 관리 구성에서 제공하는 기본 역할을 사용하는 것이 좋습니다. 여기에는 Systems Manager를 사용하여 Amazon EC2 인스턴스를 관리하는 데 필요한 최소 권한 세트가 포함되어 있습니다. 사용자 지정 역할을 사용하는 것을 선호하는 경우 역할의 신뢰 정책에서 Systems Manager를 신뢰할 수 있는 엔터티로 허용해야 합니다.
6. 구성을 선택하여 설정을 완료합니다.

기본 호스트 관리 구성을 켠 후 인스턴스가 선택한 역할의 자격 증명을 사용하는 데 30분 정도 걸릴 수 있습니다. Amazon EC2 인스턴스를 자동으로 관리하려는 각 리전에서 기본 호스트 관리 구성을 켜야 합니다.

EC2 인스턴스 권한에 대한 대체 구성

AWS Identity and Access Management(IAM) 인스턴스 프로파일을 사용하여 개별 인스턴스 수준에서 액세스 권한을 부여할 수 있습니다. 인스턴스 프로파일은 시작 시 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에 IAM 역할 정보를 전달하는 컨테이너입니다. 새로운 역할이나 이미 생성된 역할에 필요한 권한을 정의하는 IAM 정책을 하나 이상 연결하여 Systems Manager에 대한 인스턴스 프로파일을 생성할 수 있습니다.

> **Note**
>
> AWS Systems Manager의 기능인 Quick Setup을 사용하여 AWS 계정의 모든 인스턴스에서 인스턴스 프로파일을 빠르게 구성할 수 있습니다. Quick Setup은 또한 Systems Manager가 사용자를 대신하여 인스턴스에서 명령을 안전하게 실행할 수 있도록 하는 IAM 서비스 역할이나 수임 역할을 생성할 수 있습니다. Quick Setup을 사용하여 이 단계(3단계)와 4단계를 건너뛸 수 있습니다. 자세한 내용은 AWS Systems Manager Quick Setup 단원을 참조하십시오.

IAM 인스턴스 프로파일 생성에 대한 다음 세부 정보를 참조하세요.

- Systems Manager용 하이브리드 및 멀티클라우드 환경에서 비 EC2 시스템을 구성하는 경우에는 인스턴스 프로파일을 생성할 필요가 없습니다. 그 대신에 IAM 서비스 역할을 사용할 서버와 VM을 구

성합니다. 자세한 내용은 하이브리드 및 멀티클라우드 환경에서 Systems Manager에 필요한 IAM 서비스 역할 생성을 참조하세요.

- IAM 인스턴스 프로파일을 변경한 경우에는 인스턴스 자격 증명이 갱신될 때까지 시간이 걸릴 수 있습니다. 갱신이 되어야 SSM Agent가 요청을 처리하기 시작합니다. SSM Agent 또는 인스턴스를 다시 시작하여 갱신 프로세스의 속도를 높일 수 있습니다.

인스턴스 프로파일의 역할을 새로 생성하려는 경우 또는 기존 역할에 필요한 권한을 추가하려는 경우에 각각 다음 절차 중 하나를 사용합니다.

Systems Manager 관리형 인스턴스의 인스턴스 프로파일을 생성하려면(콘솔)

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할(Roles)을 선택한 후 역할 생성(Create role)을 선택합니다.
3. 신뢰할 수 있는 엔터티 유형(Trusted entity type)에 AWS 서비스을 선택합니다.
4. 사용 사례(Use case)에서 EC2를 선택한 후 다음(Next)을 선택합니다.
5. 권한 추가(Add permissions) 페이지에서 다음을 수행합니다.

   - 검색(Search) 필드를 사용하여 AmazonSSMManagedInstanceCore 정책을 찾습니다. 이름 옆의 확인란을 선택합니다.

     이 콘솔은 사용자가 다른 정책을 검색하더라도 사용자의 선택을 유지합니다.

   - 이전 절차에서 사용자 지정 S3 버킷 정책 (선택 사항) S3 버킷 액세스에 대한 사용자 지정 정책 생성를 생성한 경우 해당 이름을 검색하고 이름 옆의 확인란을 선택합니다.
   - AWS Directory Service에 의해 관리되는 Active Directory에 인스턴스를 조인하려는 경우 AmazonSSMDirectoryServiceAccess를 검색하고 이름 옆의 확인란을 선택합니다.
   - 인스턴스를 관리하거나 모니터링하기 위해 EventBridge 또는 CloudWatch Logs를 사용하려는 경우 CloudWatchAgentServerPolicy를 검색하고 이름 옆의 확인란을 선택합니다.

6. Next(다음)를 선택합니다.

7. 역할 이름(Role name)에 새 인스턴스 프로파일의 이름(예: **SSMInstanceProfile**)을 입력합니다.

> **Note**
>
> 역할 이름을 기록해 둡니다. 이 역할은 Systems Manager를 사용하여 관리할 새 인스턴스를 생성할 때 선택합니다.

8. (선택) 설명(Description)에 이 인스턴스 프로파일에 대한 설명을 입력합니다.
9. (선택) 태그(Tags)에서 이 역할에 대한 액세스를 구성, 추적 또는 제어할 태그-키 값 페어를 하나 이상 추가한 후 역할 생성(Create role)을 선택합니다. 그러면 역할 페이지로 돌아갑니다.

Systems Manager의 인스턴스 프로파일 권한을 기존 역할에 추가하려면(콘솔)

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할(Roles)을 선택한 후, Systems Manager 작업을 위한 인스턴스 프로파일과 연결하려는 기존 역할을 선택합니다.
3. 권한(Permissions) 탭에서 권한 추가, 정책 연결(Add permissions, Attach policies)을 선택합니다.
4. 정책 연결 페이지에서 다음 작업을 수행합니다.
   - 검색(Search) 필드를 사용하여 AmazonSSMManagedInstanceCore 정책을 찾습니다. 이름 옆의 확인란을 선택합니다.
   - 사용자 지정 S3 버킷 정책을 생성한 경우 검색하고 이름 옆의 확인란을 선택합니다. 인스턴스 프로파일의 사용자 지정 S3 버킷 정책에 대한 내용은 (선택 사항) S3 버킷 액세스에 대한 사용자 지정 정책 생성을 참조하십시오.
   - AWS Directory Service에 의해 관리되는 Active Directory에 인스턴스를 조인하려는 경우 AmazonSSMDirectoryServiceAccess를 검색하고 이름 옆의 확인란을 선택합니다.
   - 인스턴스를 관리하거나 모니터링하기 위해 EventBridge 또는 CloudWatch Logs를 사용하려는 경우 CloudWatchAgentServerPolicy를 검색하고 이름 옆의 확인란을 선택합니다.
5. 정책 연결을 선택합니다.

역할을 업데이트하여 신뢰할 수 있는 엔터티를 포함하거나 액세스를 추가로 제한하는 방법에 대한 자세한 내용은 IAM 사용 설명서의 역할 변경을 참조하세요.

(선택 사항) S3 버킷 액세스에 대한 사용자 지정 정책 생성

Amazon S3 액세스를 위한 사용자 정의 정책 생성은 Systems Manager 작업에서 VPC 종단점을 사용 중이거나 자체의 S3 버킷을 사용 중인 경우에만 필요합니다. 기본 호스트 관리 구성에서 생성한 기본 IAM 역할 또는 이전 절차에서 생성한 인스턴스 프로파일에 이 정책을 연결할 수 있습니다.

다음 정책에서 액세스 권한을 제공하는 AWS 관리형 S3 버킷에 대한 내용은 AWS 관리형 S3 버킷과 SSM Agent 통신 섹션을 참조하세요.

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 정책을 선택한 후 정책 생성을 선택합니다.
3. JSON 탭을 선택하고 기본 텍스트를 다음과 같이 바꿉니다.

```
{
    "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:::DOC-EXAMPLE-BUCKET/*",
                "arn:aws:s3:::DOC-EXAMPLE-
BUCKET" 5
            ]
        }
    ]
}
```

[1] 첫 번째 `Statement` 요소는 VPC 엔드포인트를 사용 중인 경우에만 필요합니다.

[2] 두 번째 `Statement` 요소는 Systems Manager 작업에서 사용하기 위해 생성한 S3 버킷을 사용 중인 경우에만 필요합니다.

[3] `PutObjectAcl` 액세스 제어 목록 권한은 다른 계정에서 S3 버킷에 대한 교차 계정 액세스를 지원하려는 경우에만 필요합니다.

[4] `GetEncryptionConfiguration` 요소는 암호화를 사용하기 위해 S3 버킷을 구성할 경우 필요합니다.

[5] 암호화를 사용하기 위해 S3 버킷을 구성할 경우 S3 버킷 루트(예: `arn:aws:s3:::DOC-EXAMPLE-BUCKET`)가 리소스(Resource) 섹션에 있어야 합니다. 사용자, 그룹 또는 역할은 루트 버킷에 대한 액세스 권한을 사용하여 구성해야 합니다.

4. 작업에 VPC 엔드포인트를 사용 중인 경우 다음을 수행합니다.

   첫 번째 `Statement` 요소에서, 각 *`region`* 자리 표시자를 이 정책이 사용되는 AWS 리전의 식별자로 대체합니다. 예를 들어 미국 동부(오하이오) 리전의 경우 `us-east-2`를 사용합니다. 지원되는 *`##`* 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

   > **Important**
   >
   > 이 정책에서는 특정 리전 대신에 와일드카드 문자(*)를 사용하지 않는 것이 좋습니다. 예를 들어, `arn:aws:s3:::aws-ssm-*/*`는 사용하지 마시고 `arn:aws:s3:::aws-ssm-us-east-2/*`를 사용하세요. 와일드카드를 사용하면 액세스 권한을 부여하도록 의도하지 않은 S3 버킷에 액세스할 수 있습니다. 둘 이상의 리전에 대해 인스턴스 프로파일을 사용하려는 경우, 각 리전에 대해 첫 번째 `Statement` 요소를 반복하는 것이 좋습니다.

-또는-

작업에서 VPC 엔드포인트를 사용하고 있지 않다면, 첫 번째 `Statement` 요소를 삭제할 수 있습니다.

5. Systems Manager 작업에 자체의 S3 버킷을 사용 중인 경우 다음을 수행합니다.

   두 번째 `Statement` 요소에서 *`DOC-EXAMPLE-BUCKET`*을 계정에 있는 S3 버킷의 이름으로 바꿉니다. 이 버킷은 Systems Manager 작업을 위해 사용되며, 리소스로 `"arn:aws:s3:::my-bucket-name/*"`을 사용하여 버킷의 객체에 대한 권한을 제공합니다. 버킷 또는 버킷의 객체에 대한 권한 제공에 대한 자세한 내용은 Amazon Simple Storage Service 개발자 안내서의 Amazon S3 작업 및 AWS 블로그 게시물 IAM 정책, 버킷 정책 및 ACLs!를 참조하세요. ACL (S3 리소스에 대한 액세스 제어).

   > **Note**
   >
   > 둘 이상의 버킷을 사용하는 경우 각각에 대한 ARN을 제공합니다. 버킷에 대한 권한은 다음 예를 참조하세요.
   >
   > ```
   > "Resource": [
   > "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*",
   > "arn:aws:s3:::DOC-EXAMPLE-BUCKET2/*"
   >              ]
   > ```

   -또는-

   Systems Manager 작업에서 자체의 S3 버킷을 사용하고 있지 않다면 두 번째 `Statement` 요소를 삭제할 수 있습니다.

6. 다음: 태그를 선택합니다.
7. (선택 사항) 태그 추가(Add tag)를 선택하고 정책에 대한 기본 설정 태그를 입력하여 태그를 추가합니다.
8. 다음: 검토를 선택합니다.
9. 이름(Name)에 이 정책을 식별할 수 있는 이름(예: **`SSMInstanceProfileS3Policy`**)을 입력합니다.
10. 정책 생성을 선택합니다.

관리형 인스턴스에 대한 추가 정책 고려 사항

이 섹션에서는 기본 호스트 관리 구성에서 생성한 기본 IAM 역할 또는 AWS Systems Manager용 인스턴스 프로파일에 추가할 수 있는 몇 가지 정책에 대해 설명합니다. 인스턴스와 Systems Manager API 간 통신에 권한을 부여하려면 시스템 요구 사항과 보안 요구 사항을 반영하는 사용자 정의 정책을 만드는 것이 좋습니다. 운영 계획에 따라, 다른 정책 중 하나 이상에 제시된 권한이 필요할 수 있습니다.

정책: **`AmazonSSMDirectoryServiceAccess`**

Windows Server의 Amazon EC2 인스턴스를 Microsoft AD 디렉터리에 조인하려는 경우에만 필요합니다.

이 AWS 관리형 정책은 SSM Agent가 관리형 인스턴스의 도메인 조인 요청을 위해 사용자 대신 AWS Directory Service에 액세스하도록 허용합니다. 자세한 내용은 AWS Directory Service 관리 안내서의 Windows EC2 인스턴스를 원활하게 조인을 참조하세요.

정책: **`CloudWatchAgentServerPolicy`**

인스턴스에서 지표 및 로그 데이터를 읽고 Amazon CloudWatch에 쓰기 위해 인스턴스에서 CloudWatch 에이전트를 설치하고 실행하려는 경우에만 필요합니다. 이렇게 하면 AWS 리소스의 문제 및 변경을 모니터링, 분석하고 빠르게 대응할 수 있습니다.

기본 호스트 관리 구성 또는 인스턴스 프로파일에서 생성된 기본 IAM 역할에는 Amazon EventBridge 또는 Amazon CloudWatch Logs와 같은 기능을 사용하는 경우에만 이 정책이 필요합니다. (예를 들어 특정 CloudWatch Logs 로그 스트림에 대한 쓰기 액세스를 제어하는 더 제한적인 정책을 생성할 수도 있습니다.)

> **Note**
>
> EventBridge 및 CloudWatch Logs 기능 사용은 선택 사항입니다. 그러나 사용하기로 한 경우에는 Systems Manager 구성 프로세스 시작 시 옵션을 설정하는 것이 좋습니다. 자세한 내용은 Amazon EventBridge User Guide와 Amazon CloudWatch Logs User Guide를 참조하세요.

추가 Systems Manager 기능에 대한 권한이 있는 IAM 정책을 생성하려면 다음 리소스를 참조하세요.

- IAM 정책을 사용하여 Systems Manager 파라미터에 대한 액세스 제한
- Automation 설정

- 2단계: Session Manager의 인스턴스 권한 확인 또는 추가

인스턴스에 Systems Manager 인스턴스 프로파일 연결(콘솔)

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창의 인스턴스에서 인스턴스를 선택합니다.
3. 목록에서 EC2 인스턴스를 찾아서 선택합니다.
4. 작업(Actions) 메뉴에서 보안(Security), IAM 역할 수정(Modify IAM role)을 선택합니다.
5. IAM 역할(IAM role)에 EC2 인스턴스 권한에 대한 대체 구성의 절차를 사용하여 생성한 인스턴스 프로파일을 선택합니다.
6. IAM 역할 업데이트(Update IAM role)를 선택합니다.

인스턴스에 IAM 역할 연결에 대한 자세한 내용은 선택한 운영 체제 유형에 따라 다음 중 하나를 참조하세요.

- Amazon EC2 사용 설명서의 인스턴스에 IAM 역할 연결
- Amazon EC2 사용 설명서의 인스턴스에 IAM 역할 연결

계속해서 Systems Manager용 VPC 엔드포인트를 사용하여 EC2 인스턴스의 보안 개선로 이동하십시오.

Systems Manager용 VPC 엔드포인트를 사용하여 EC2 인스턴스의 보안 개선

Amazon Virtual Private Cloud(VPC)에서 인터페이스 VPC 엔드포인트를 사용하도록 AWS Systems Manager를 구성하여 관리형 노드(하이브리드 및 멀티클라우드 환경의 비 EC2 시스템 포함)의 보안 태세를 개선할 수 있습니다. 인터페이스 VPC 엔드포인트(인터페이스 엔드포인트)를 사용하면 AWS PrivateLink에 의해 구동되는 서비스에 연결할 수 있습니다. AWS PrivateLink는 프라이빗 IP 주소를 사용하여 Amazon Elastic Compute Cloud(Amazon EC2) 및 Systems Manager API에 비공개로 액세스할 수 있는 기술입니다.

AWS PrivateLink는 관리형 인스턴스, Systems Manager 및 Amazon EC2 간의 모든 네트워크 트래픽을 Amazon 네트워크로 제한합니다. 즉, 관리형 인스턴스는 인터넷에 액세스할 수 없습니다. AWS PrivateLink를 사용하는 경우 인터넷 게이트웨이, NAT 디바이스 또는 가상 프라이빗 게이트웨이가 필요 없습니다.

AWS PrivateLink를 구성하는 것이 필수는 아니지만 구성하는 것이 좋습니다. AWS PrivateLink 및 VPC 엔드포인트에 대한 자세한 내용은 AWS PrivateLink 및 VPC 엔드포인트를 참조하세요.

> **Note**
>
> VPC 엔드포인트 사용의 대체 방법은 관리형 인스턴스에서 아웃바운드 인터넷 액세스를 허용하는 것입니다. 이 경우 관리형 인스턴스는 다음 엔드포인트에 대한 HTTPS(포트 443) 아웃바운드 트래픽도 허용해야 합니다.
>
> - `ssm.region.amazonaws.com`
> - `ssmmessages.region.amazonaws.com`
> - `ec2messages.region.amazonaws.com`
>
> SSM Agent는 클라우드의 Systems Manager 서비스에 대한 모든 연결을 시작합니다. 이러한 이유로 Systems Manager의 인스턴스에 대한 인바운드 트래픽을 허용하도록 방화벽을 구성할 필요가 없습니다.
> 이러한 엔드포인트 호출에 대한 자세한 내용은 참조: ec2messages, ssmmessages 및 기타 API 작업 섹션을 참조하세요.

Amazon VPC 정보

Amazon Virtual Private Cloud(Amazon VPC)를 사용하면 AWS 클라우드 내에서 논리적으로 격리된 자체 영역에 Virtual Private Cloud(VPC)라고 하는 가상 네트워크를 정의할 수 있습니다. 인스턴스와 같은 AWS 리소스를 VPC에서 시작할 수 있습니다. VPC는 고객의 자체 데이터 센터에서 운영하는 기존 네트워크와 매우 유사하지만 AWS의 확장 가능한 인프라를 사용한다는 이점을 제공합니다. 해당 IP 주소 범위를 선택하고, 서브넷을 만든 후 라우팅 테이블, 네트워크 게이트웨이 및 보안 설정을 구성하여 VPC를 구성할 수 있습니다. VPC의 인스턴스를 인터넷에 연결합니다. VPC를 사내 데이터 센터에 연결하여 AWS 클라우드에서 데이터 센터를 확장할 수 있습니다. 각의 서브넷에서 리소스를 보호하기 위해 보안 그룹 및 네트워크 액세스 제어 목록을 포함한 다중 보안 계층을 사용할 수 있습니다. 자세한 내용은 Amazon VPC 사용 설명서를 참조하세요.

주제

- VPC 엔드포인트 제약 및 제한
- Systems Manager용 VPC 엔드포인트 생성
- 인터페이스 VPC 엔드포인트 정책 생성

VPC 엔드포인트 제약 및 제한

Systems Manager에 대해 VPC 엔드포인트를 구성하기 전에 다음 제한 사항에 유의합니다.

교차 리전 요청

VPC 엔드포인트는 교차 리전 요청을 지원하지 않습니다. 버킷과 같은 AWS 리전에 엔드포인트를 생성하는지 확인합니다. Amazon S3 콘솔 또는 get-bucket-location 명령을 사용하여 버킷의 위치를 확인할 수 있습니다. 리전별 Amazon S3 엔드포인트를 사용하여 버킷에 액세스하십시오(예: `DOC-EXAMPLE-BUCKET.s3-us-west-2.amazonaws.com`). Amazon S3의 리전별 엔드포인트에 대한 자세한 내용은Amazon Web Services 일반 참조의 Amazon S3 엔드포인트를 참조하세요. AWS CLI를 사용하여 Amazon S3에 요청할 경우 기본 리전을 버킷과 동일한 리전으로 설정하거나 요청에 `--region` 파라미터를 사용합니다.

VPC 피어링 연결

VPC 인터페이스 엔드포인트는 리전 내 및 리전 간 VPC 피어링 연결을 통해 액세스할 수 있습니다. VPC 인터페이스 엔드포인트에 대한 VPC 피어링 연결 요청에 대한 자세한 내용은 Amazon Virtual Private Cloud 사용 설명서의 VPC 피어링 연결(할당량)을 참조하세요.

VPC 게이트웨이 엔드포인트 연결은 VPC 외부로 확장할 수 없습니다. VPC의 VPC 피어링 연결의 반대편에 있는 리소스는 게이트웨이 엔드포인트를 사용하여 게이트웨이 엔드포인트 서비스의 리소스와 통신할 수 없습니다. VPC 게이트웨이 엔드포인트에 대한 VPC 피어링 연결 요청에 대한 자세한 내용은 Amazon Virtual Private Cloud 사용 설명서의 VPC 엔드포인트(할당량)를 참조하세요.

수신 연결

VPC 종단점에 연결된 보안 그룹은 관리형 인스턴스의 프라이빗 서브넷에서 443 포트로 들어오는 연결을 허용해야 합니다. 수신 연결을 허용하지 않으면 관리형 인스턴스가 SSM 및 EC2 엔드포인트에 연결할 수 없습니다.

DNS 확인

사용자 지정 DNS 서버를 사용하는 경우 `amazonaws.com` 도메인에 대한 모든 쿼리의 조건부 전달자를 VPC의 Amazon DNS 서버에 추가해야 합니다.

S3 버킷

VPC 엔드포인트 정책이 최소한 다음 Amazon S3 버킷에 대해 액세스를 허용해야 합니다.

- AWS 관리형 S3 버킷과 SSM Agent 통신에 열거된 S3 버킷

- AWS 리전의 패치 기준 작업을 위해 Patch Manager가 사용하는 S3 버킷 이 버킷에는 패치 기준선 서비스가 가져와 인스턴스에서 실행하는 코드가 포함되어 있습니다. 각 AWS 리전에는 자체 패치 기준 작업 버킷이 있으므로 패킷 기준 문서를 실행할 때 코드를 검색할 수 있습니다. 이 코드를 다운로드할 수 없으면 패치 기준선 명령이 실패합니다.

  > **Note**
  >
  > 온프레미스 방화벽을 사용하고 Patch Manager를 사용하려는 경우 해당 방화벽에서 적절한 패치 기준 엔드포인트에 대한 액세스도 허용해야 합니다.

  AWS 리전의 버킷에 대한 액세스 권한을 제공하려면 엔드포인트 정책에 다음 권한을 포함합니다.

  ```
  arn:aws:s3:::patch-baseline-snapshot-region/*
  arn:aws:s3:::aws-ssm-region/*
  ```

  *##*은 미국 동부(오하이오) 리전의 `us-east-2` 같이 AWS Systems Manager가 지원하는 AWS 리전의 식별자를 나타냅니다. 지원되는 *##* 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

  다음 예를 참조하세요.

  ```
  arn:aws:s3:::patch-baseline-snapshot-us-east-2/*
  arn:aws:s3:::aws-ssm-us-east-2/*
  ```

  > **Note**
  >
  > 중동(바레인) 리전(me-south-1)에서만 이러한 버킷이 다른 명명 규칙을 사용합니다. 이 AWS 리전의 경우에만 다음 2개의 버킷을 대신 사용합니다.
  >
  > - `patch-baseline-snapshot-me-south-1-uduvl7q8`
  > - `aws-patch-manager-me-south-1-a53fc9dce`

Amazon CloudWatch Logs

인스턴스가 인터넷에 액세스하는 것을 허용하지 않는 경우 CloudWatch Logs에 로그를 보내는 기능을 사용하도록 CloudWatch Logs에 대한 VPC 엔드포인트를 생성합니다. CloudWatch Logs용 엔드포인

트 생성에 대한 자세한 내용은 Amazon CloudWatch Logs User Guide의 Creating a VPC endpoint for CloudWatch Logs를 참조하세요.

하이브리드 및 멀티클라우드 환경의 DNS

하이브리드 및 멀티클라우드 환경에서 AWS PrivateLink 엔드포인트로 작업하도록 DNS를 구성하는 방법에 대한 자세한 내용은 Amazon VPC 사용 설명서의 인터페이스 엔드포인트의 프라이빗 DNS를 참조하세요. 자신의 DNS를 사용하는 경우에는 Route 53 해석기를 사용할 수 있습니다. 자세한 내용은 Amazon Route 53 개발자 안내서의 VPC와 네트워크 간 DNS 쿼리 해석을 참조하세요.

Systems Manager용 VPC 엔드포인트 생성

다음 정보를 사용하여 AWS Systems Manager에 대한 VPC 인터페이스 및 게이트웨이 엔드포인트를 생성합니다. 이 주제는 Amazon VPC 사용 설명서의 절차에 연결됩니다.

Systems Manager용 VPC 엔드포인트를 생성하려면

이 절차의 첫 번째 단계에서는 Systems Manager에 대해 3개의 필수 인터페이스 엔드포인트와 1개의 옵션 인터페이스 엔드포인트를 생성합니다. 처음 3개의 엔드포인트는 Systems Manager가 VPC에서 작동하는 데 필요합니다. 네 번째 엔드포인트인 `com.amazonaws.region.ssmmessages`는 Session Manager 기능을 사용하는 경우에만 필요합니다.

두 번째 단계에서는 Systems Manager가 Amazon S3에 액세스하는 데 필요한 게이트웨이 엔드포인트를 생성합니다.

> **Note**
>
> ##은 미국 동부(오하이오) 리전의 `us-east-2` 같이 AWS Systems Manager가 지원하는 AWS 리전의 식별자를 나타냅니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

1. 인터페이스 엔드포인트 생성의 단계에 따라 다음 인터페이스 엔드포인트를 생성합니다.

   - **com.amazonaws.*region*.ssm**: Systems Manager 서비스의 엔드포인트입니다.
   - **com.amazonaws.*region*.ec2messages**: Systems Manager는 이 엔드포인트를 사용하여 SSM Agent에서 Systems Manager 서비스를 호출합니다.
   - **com.amazonaws.*region*.ec2**: Systems Manager를 사용하여 VSS 지원 스냅샷을 만든 경우 EC2 서비스에 대한 엔드포인트가 있어야 합니다. EC2 엔드포인트가 정의되어 있지 않으면 연

결된 Amazon EBS 볼륨을 표시하는 호출이 실패하고 이에 따라 Systems Manager 명령에 실패합니다.

- **com.amazonaws.*region*.ssmmessages**: 이 엔드포인트는 Session Manager를 사용하여 보안 데이터 채널을 통해 인스턴스에 연결하는 경우에만 필요합니다. 자세한 내용은 AWS Systems Manager Session Manager 및 참조: ec2messages, ssmmessages 및 기타 API 작업을(를) 참조하세요.
- **com.amazonaws.*region*.kms** - 이 엔드포인트는 선택 사항입니다. 그러나 Session Manager 또는 Parameter Store 파라미터에 대해 AWS Key Management Service(AWS KMS)를 사용하려는 경우 이를 만들 수 있습니다.
- **com.amazonaws.*region*.logs** - 이 엔드포인트는 선택 사항입니다. Session Manager, Run Command 또는 SSM Agent 로그에 대해 Amazon CloudWatch Logs(CloudWatch Logs)를 사용하려는 경우 이를 생성할 수 있습니다.

2. 게이트웨이 엔드포인트 생성의 단계에 따라 Amazon S3에 대한 다음 게이트웨이 엔드포인트를 생성합니다.

   - **com.amazonaws.*region*.s3** - Systems Manager가 이 엔드포인트를 사용하여 SSM Agent를 업데이트하고 패치 작업을 수행할 수 있습니다. Systems Manager는 이 엔드포인트를 사용하여 S3 버킷 저장을 선택한 출력 로그 업로드, 버킷에 저장한 스크립트 또는 기타 파일 검색 등의 태스크를 수행합니다. 인스턴스와 연결된 보안 그룹이 아웃바운드 트래픽을 제한하는 경우 Amazon S3의 접두사 목록에 대한 트래픽을 허용하는 규칙을 추가해야 합니다. 자세한 내용은 AWS PrivateLink Guide의 Modify your security group을 참조하세요.

   SSM Agent가 액세스할 수 있어야 하는 AWS 관리형 S3 버킷에 대한 자세한 내용은 AWS 관리형 S3 버킷과 SSM Agent 통신 섹션을 참조하세요. Systems Manager 작업에서 Virtual Private Cloud(VPC) 엔드포인트를 사용하는 경우 Systems Manager를 위한 Amazon EC2 인스턴스 프로파일 또는 하이브리드 및 멀티클라우드 환경의 비 EC2 관리형 노드를 위한 서비스 역할에 명시적 권한을 제공해야 합니다.

인터페이스 VPC 엔드포인트 정책 생성

AWS Systems Manager에 대한 VPC 인터페이스 엔드포인트의 정책을 생성할 수 있습니다. 이 정책에서 다음을 지정할 수 있습니다.

- 작업을 수행할 수 있는 보안 주체.
- 수행할 수 있는 작업
- 수행되는 작업을 가질 수 있는 리소스

자세한 내용은 Amazon VPC 사용 설명서의 VPC 엔드포인트를 통해 서비스에 대한 액세스 제어를 참조하세요.

하이브리드 및 멀티클라우드 환경에서 Systems Manager 사용하기

AWS Systems Manager를 사용하여 Amazon Elastic Compute Cloud(EC2) 인스턴스와 다양한 비 EC2 시스템 유형을 모두 관리할 수 있습니다. 이 섹션에서는 계정 및 시스템 관리자가 하이브리드 및 멀티클라우드 환경에서 Systems Manager를 사용하여 비 EC2 시스템을 관리하려고 수행하는 설정 작업을 설명합니다. 이러한 설정을 완료하면 AWS 계정 관리자가 권한을 부여한 사용자는 Systems Manager를 사용하여 조직의 비 EC2 시스템을 구성하고 관리할 수 있습니다.

Systems Manager와 함께 사용하도록 구성된 모든 시스템을 관리형 노드라고 합니다.

> **Note**
>
> - 다른 비 EC2 시스템과 동일한 하이브리드 정품 인증 단계를 사용하여 엣지 디바이스를 관리형 노드로 등록할 수 있습니다. 이러한 유형의 엣지 디바이스에는 AWS IoT 디바이스 이외의 디바이스와 AWS IoT 디바이스가 모두 포함됩니다. 이 섹션에서 설명하는 프로세스에 따라 이러한 유형의 엣지 디바이스를 설정합니다.
>
> Systems Manager AWS IoT Greengrass 코어 소프트웨어를 사용하는 엣지 디바이스도 지원합니다. AWS IoT Greengrass 코어 디바이스의 설정 프로세스 및 요구 사항은 AWS 엣지 디바이스 이외의 디바이스와 AWS IoT 디바이스의 설정 프로세스 및 요구 사항과 다릅니다. Systems Manager에 사용할 AWS IoT Greengrass 디바이스를 등록하는 방법에 대한 자세한 내용은 Systems Manager를 통한 엣지 디바이스 관리 섹션을 참조하세요.
> - 비 EC2 macOS 시스템에서는 Systems Manager 하이브리드 및 멀티클라우드 환경이 지원되지 않습니다.

Systems Manager를 사용하여 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스를 관리하거나 하이브리드 및 멀티클라우드 환경에서 Amazon EC2 인스턴스와 비 EC2 시스템을 모두 사용하려는 경우 먼저 EC2 인스턴스에 Systems Manager 사용의 단계를 따릅니다.

Systems Manager에 대한 하이브리드 및 멀티클라우드 환경을 구성하면 다음을 수행할 수 있습니다.

- 한곳에서 동일한 도구와 스크립트를 사용하여 하이브리드 및 멀티클라우드 워크로드를 원격으로 관리할 수 있는 일관성 있고 안전한 수단이 됩니다.

- AWS Identity and Access Management(IAM)를 통해 서버와 VM에서 수행할 수 있는 작업에 대한 액세스 제어를 중앙 집중화합니다.
- AWS CloudTrail에 기록된 API 활동을 확인하여 시스템에 대해 수행되는 작업에 대한 감사를 중앙 집중화합니다.

  CloudTrail을 사용한 Systems Manager 작업 모니터링에 대한 자세한 내용은 AWS CloudTrail을 사용하여 AWS Systems ManagerAPI 호출을 로깅 섹션을 참조하세요.
- 서비스 실행 성공에 대한 알림을 보내도록 Amazon EventBridge 및 Amazon Simple Notification Service(Amazon SNS)를 구성하여 모니터링을 중앙 집중화합니다.

  EventBridge를 사용하여 Systems Manager 이벤트 모니터링에 대한 자세한 내용은 Amazon EventBridge로 Systems Manager 이벤트 모니터링 섹션을 참조하세요.

관리형 노드 정보

이 섹션에 설명된 대로 Systems Manager용 비 EC2 시스템 구성을 마치면 하이브리드 정품 인증 시스템이 AWS Management Console에 나열되고 관리형 노드로 설명됩니다. 콘솔에서는 하이브리드 정품 인증 관리형 노드의 ID가 접두사 "mi-"로 Amazon EC2 인스턴스와 구별됩니다. Amazon EC2 인스턴스 ID는 접두사 “i-”를 사용합니다.

관리형 노드는 Systems Manager에 사용하도록 구성된 시스템입니다. 이전에는 관리형 노드를 모두 관리형 인스턴스라고 했습니다. 이제 인스턴스라는 용어는 EC2 인스턴스만을 지칭합니다. deregister-managed-instance 명령은 이러한 용어 변경 이전에 이름이 붙여졌습니다.

자세한 내용은 관리형 노드 작업 단원을 참조하십시오.

인스턴스 사용자 정보

Systems Manager에서는 하이브리드 및 멀티클라우드 환경의 비 EC2 관리형 노드에 대한 표준 인스턴스 티어와 고급 인스턴스 티어를 제공합니다. 표준 인스턴스 티어를 사용하면 한 AWS 리전의 AWS 계정당 최대 1,000개의 하이브리드 정품 인증 시스템을 등록할 수 있습니다. 단일 계정 및 리전에 1,000개가 넘는 비 EC2 시스템을 등록해야 하는 경우에는 고급 인스턴스 티어를 사용합니다. 고급 인스턴스에서는 AWS Systems Manager Session Manager를 사용하여 비 EC2 시스템에도 연결할 수 있습니다. Session Manager에서는 관리형 노드에 대한 대화형 쉘 액세스 권한를 제공합니다.

자세한 내용은 인스턴스 티어 구성 단원을 참조하십시오.

주제

- 하이브리드 및 멀티클라우드 환경에서 Systems Manager에 필요한 IAM 서비스 역할 생성

- 하이브리드 활성화를 생성하여 Systems Manager에 노드 등록
- 하이브리드 Linux 노드에 SSM Agent를 설치하는 방법
- 하이브리드 Windows 노드에 SSM Agent를 설치하는 방법

하이브리드 및 멀티클라우드 환경에서 Systems Manager에 필요한 IAM 서비스 역할 생성

하이브리드 및 멀티클라우드 환경의 비 EC2(Amazon Elastic Compute Cloud) 시스템에서 AWS Systems Manager 서비스와 통신하려면 AWS Identity and Access Management(IAM) 서비스 역할이 필요합니다. 역할이 Systems Manager 서비스에 AWS Security Token Service(AWS STS) `AssumeRole` 신뢰를 부여합니다. AWS 계정마다 한 번만 하이브리드 및 멀티클라우드 환경의 서비스 역할을 생성하면 됩니다. 그러나 하이브리드 및 멀티클라우드 환경의 시스템에 다양한 권한이 필요한 경우 다양한 하이브리드 정품 인증을 위한 여러 서비스 역할 생성을 선택할 수 있습니다.

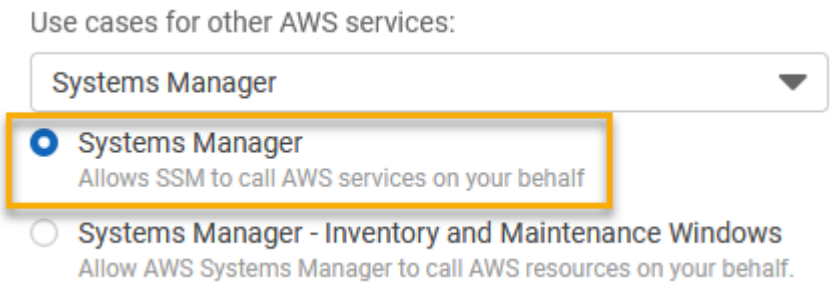
다음 절차에서는 Systems Manager 콘솔 또는 기본 명령줄 도구를 사용하여 필수 서비스 역할을 생성하는 방법을 설명합니다.

AWS Management Console을 사용하여 Systems Manager 하이브리드 활성화를 위한 IAM 서비스 역할 생성

다음 절차를 사용하여 용 IAM 서비스 역할을 생성합니다. 이 절차에서는 Systems Manager 코어 기능을 위한 `AmazonSSMManagedInstanceCore` 정책을 사용합니다. 사용 사례에 따라 온프레미스 컴퓨터가 다른 기능이나 AWS 서비스에 액세스하려면 서비스 역할에 정책을 추가해야 할 수 있습니다. 예를 들어, 필수 AWS 관리형 Amazon Simple Storage Service(Amazon S3) 버킷에 액세스하지 않으면 Patch Manager 패치 적용 작업이 실패합니다.

서비스 역할을 만들려면(콘솔)

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할(Roles)을 선택한 후 역할 생성(Create role)을 선택합니다.
3. 신뢰할 수 있는 엔터티 선택(Select trusted entity)에서 다음을 선택합니다.
    1. 신뢰할 수 있는 엔터티 유형에 AWS 서비스를 선택합니다.
    2. 기타 AWS 서비스 사용 사례에서 Systems Manager를 선택합니다.
    3. 다음 이미지에 표시된 것과 같이 Systems Manager를 선택합니다.



4. Next(다음)를 선택합니다.
5. 권한 추가(Add permissions) 페이지에서 다음을 수행합니다.
   - 검색(Search) 필드를 사용하여 AmazonSSMManagedInstanceCore 정책을 찾습니다. 이름 옆의 확인란을 선택합니다.

   - 이 콘솔은 사용자가 다른 정책을 검색하더라도 사용자의 선택을 유지합니다.
   - (선택 사항) S3 버킷 액세스에 대한 사용자 지정 정책 생성 절차에서 사용자 지정 S3 버킷 정책을 생성한 경우 해당 이름을 검색하고 이름 옆의 확인란을 선택합니다.
   - AWS Directory Service에서 관리하는 Active Directory에 비 EC2 시스템을 조인하려는 경우 AmazonSSMDirectoryServiceAccess를 검색하고 이름 옆의 확인란을 선택합니다.
   - EventBridge 또는 CloudWatch Logs를 사용하여 관리형 노드를 관리하거나 모니터링하려는 경우 CloudWatchAgentServerPolicy를 검색하고 이름 옆의 확인란을 선택합니다.
6. Next(다음)를 선택합니다.
7. 역할 이름의 경우 새 IAM 서버 역할의 이름(예: **SSMServerRole**)을 입력합니다.

> **Note**
>
> 역할 이름을 기록해 둡니다. 이 역할은 Systems Manager를 사용하여 관리하려는 새 시스템을 등록할 때 선택하게 됩니다.

8. (선택 사항) 설명의 경우 이 IAM 서버 역할에 대한 설명을 업데이트합니다.

9. (선택 사항) Tags(태그)에서 이 역할에 대한 액세스를 구성, 추적 또는 제어할 태그-키 값 페어를 하나 이상 추가합니다.
10. 역할 생성(Create role)을 선택합니다. 그러면 역할 페이지로 돌아갑니다.

AWS CLI을 사용하여 Systems Manager 하이브리드 활성화를 위한 IAM 서비스 역할 생성

다음 절차를 사용하여 용 IAM 서비스 역할을 생성합니다. 이 절차에서는 `AmazonSSMManagedInstanceCore` 정책 Systems Manager 코어 기능을 사용합니다. 사용 사례에 따라 하이브리드 및 멀티클라우드 환경의 비 EC2 시스템에서 기타 기능 또는 AWS 서비스에 액세스할 수 있도록 서비스 역할에 추가 정책을 추가하는 것이 좋습니다.

S3 버킷 정책 요구 사항

다음 사례 중 하나에 해당되는 경우 이 절차를 완료하기 전에 Amazon Simple Storage Service(Amazon S3) 버킷에 대한 사용자 정의 IAM 권한 정책을 생성해야 합니다.

- 사례 1 - VPC 엔드포인트를 사용하여 VPC를 지원되는 AWS 서비스 및 AWS PrivateLink에 의해 제공되는 VPC 엔드포인트 서비스에 비공개로 연결하고 있습니다.
- 사례 2 - Amazon S3 버킷에 대한 Run Command 명령 또는 Session Manager 세션의 출력을 저장하기 위한 작업 등 Systems Manager 작업의 일부로 생성되는 S3 버킷을 사용할 계획입니다. 진행하기 전에 인스턴스 프로파일에 대한 사용자 정의 S3 버킷 정책 생성의 단계를 수행합니다. 해당 주제의 S3 버킷 정책에 대한 정보 또한 서비스 역할에 적용됩니다.

AWS CLI

하이브리드 및 멀티클라우드 환경의 IAM 서비스 역할 생성 방법(AWS CLI)

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.
2. 로컬 컴퓨터에서 다음 신뢰 정책과 함께 `SSMService-Trust.json`과 같은 이름으로 텍스트 파일을 생성합니다. 파일을 `.json` 파일 확장명으로 저장해야 합니다. 하이브리드 활성화를 생성한 ARN에서 AWS 계정 및 AWS 리전를 지정해야 합니다.

```
{
   "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. AWS CLI를 열고, JSON 파일을 생성한 디렉터리에서 create-role 명령을 사용하여 서비스 역할을 생성합니다. 이 예제에서는 SSMServiceRole라는 역할을 생성합니다. 원하는 경우 다른 이름을 선택할 수 있습니다.

   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. 다음과 같이 attach-role-policy 명령을 실행하여, 방금 생성한 서비스 역할이 세션 토큰을 생성할 수 있도록 허용합니다. 세션 토큰은 Systems Manager를 사용하여 명령을 실행할 수 있도록 관리형 노드 권한을 부여합니다.

> **Note**
>
> 하이브리드 및 멀티클라우드 환경의 관리형 노드를 위한 서비스 프로파일을 위해 추가하는 정책은 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스를 위한 인스턴스 프로파일을 생성하는 데 사용되는 정책과 동일합니다. 다음 명령에 사용되는 AWS 정책에 대한 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성을 참조하세요.

(필수) 다음 명령을 실행하여 관리형 노드가 AWS Systems Manager 서비스 핵심 기능을 사용할 수 있게 허용합니다.

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
```

서비스 역할을 위한 사용자 정의 S3 버킷 정책을 생성한 경우 다음 명령을 실행하여 AWS Systems Manager Agent(SSM Agent)가 정책에서 지정한 버킷에 액세스할 수 있게 합니다. *account-id* 및 *DOC-EXAMPLE-BUCKET*을 AWS 계정 ID 및 버킷 이름으로 바꿉니다.

Linux & macOS

```
aws iam attach-role-policy \
    --role-name SSMServiceRole \
    --policy-arn arn:aws:iam::account-id:policy/DOC-EXAMPLE-BUCKET
```

Windows

```
aws iam attach-role-policy ^
```

```
    --role-name SSMServiceRole ^
    --policy-arn arn:aws:iam::account-id:policy/DOC-EXAMPLE-BUCKET
```

(선택) 다음 명령을 실행하여 SSM Agent가 관리형 노드의 도메인 조인 요청을 위해 사용자 대신 AWS Directory Service에 액세스하도록 허용합니다. 노드를 Microsoft AD 디렉터리에 조인하는 경우에만 서비스 역할에 이 정책이 필요합니다.

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
```

(선택) 다음 명령을 실행하여 CloudWatch 에이전트가 관리형 노드에서 실행되도록 허용합니다. 이 명령을 사용하면 노드에서 정보를 읽고 CloudWatch에 쓸 수 있습니다. Amazon EventBridge 또는 Amazon CloudWatch Logs 등의 서비스를 사용할 경우에만 서비스 프로파일에 이 정책이 필요합니다.

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

Tools for PowerShell

하이브리드 및 멀티클라우드 환경의 IAM 서비스 역할 생성 방법(AWS Tools for Windows PowerShell)

1. 아직 설치하지 않은 경우 AWS Tools for PowerShell(Tools for Windows PowerShell)을 설치하고 구성합니다.

   자세한 내용은 AWS Tools for PowerShell 설치를 참조하세요.

2. 로컬 컴퓨터에서 다음 신뢰 정책과 함께 SSMService-Trust.json과 같은 이름으로 텍스트 파일을 생성합니다. 파일을 .json 파일 확장명으로 저장해야 합니다. 하이브리드 활성화를 생성한 ARN에서 AWS 계정 및 AWS 리전를 지정해야 합니다.

```
{
   "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. 관리 모드에서 PowerShell을 열고 JSON 파일을 생성한 디렉터리에서 New-IAMRole을 실행해 다음과 같이 서비스 역할을 생성합니다. 이 예제에서는 SSMServiceRole라는 역할을 생성합니다. 원하는 경우 다른 이름을 선택할 수 있습니다.

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

4. 다음과 같이 Register-IAMRolePolicy를 사용하여, 생성한 서비스 역할이 세션 토큰을 생성할 수 있도록 허용합니다. 세션 토큰은 Systems Manager를 사용하여 명령을 실행할 수 있도록 관리형 노드 권한을 부여합니다.

> **Note**
>
> 하이브리드 및 멀티클라우드 환경의 관리형 노드를 위한 서비스 프로파일을 위해 추가하는 정책은 EC2 인스턴스를 위한 인스턴스 프로파일을 생성하는 데 사용되는 정책과 동일합니다. 다음 명령에 사용되는 AWS 정책에 대한 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성을 참조하세요.

(필수) 다음 명령을 실행하여 관리형 노드가 AWS Systems Manager 서비스 핵심 기능을 사용할 수 있게 허용합니다.

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

서비스 역할을 위한 사용자 정의 S3 버킷 정책을 생성한 경우 다음 명령을 실행하여 SSM Agent가 정책에서 지정한 버킷에 액세스할 수 있게 허용합니다. *account-id* 및 *my-bucket-policy-name*을 AWS 계정 ID 및 버킷 이름으로 바꿉니다.

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

(선택) 다음 명령을 실행하여 SSM Agent가 관리형 노드의 도메인 조인 요청을 위해 사용자 대신 AWS Directory Service에 액세스하도록 허용합니다. 노드를 Microsoft AD 디렉터리에 조인하는 경우에만 서버 역할에 이 정책이 필요합니다.

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

(선택) 다음 명령을 실행하여 CloudWatch 에이전트가 관리형 노드에서 실행되도록 허용합니다. 이 명령을 사용하면 노드에서 정보를 읽고 CloudWatch에 쓸 수 있습니다. Amazon EventBridge 또는 Amazon CloudWatch Logs 등의 서비스를 사용할 경우에만 서비스 프로파일에 이 정책이 필요합니다.

```
Register-IAMRolePolicy `
```

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

계속해서 하이브리드 활성화를 생성하여 Systems Manager에 노드 등록로 이동하십시오.

하이브리드 활성화를 생성하여 Systems Manager에 노드 등록

Amazon Elastic Compute Cloud(EC2) 인스턴스 이외의 시스템을 하이브리드 및 멀티클라우드 환경의 관리형 노드로 설정하려면 하이브리드 정품 인증을 생성하여 적용합니다. 정품 인증을 완료하면 정품 인증 코드와 정품 인증 ID가 즉시 콘솔 페이지 상단에 표시됩니다. 하이브리드 및 멀티클라우드 환경의 비 EC2 시스템에 AWS Systems Manager SSM Agent를 설치할 때 이 코드 및 ID 조합을 지정합니다. 코드 및 ID는 관리형 노드에서 Systems Manager 서비스에 대한 보안 액세스를 제공합니다.

> **Important**
>
> 활성화를 생성한 방법에 따라, Systems Manager는 활성화 코드 및 ID를 콘솔이나 명령 창에 즉시 반환합니다. 이 정보를 복사하여 안전한 장소에 보관하십시오. 콘솔에서 벗어나거나 명령 창을 닫으면 이 정보가 손실될 수 있습니다. 이 정보가 손실하면 새로운 정품 인증을 생성해야 합니다.

정품 인증 만료 정보

활성화 만료란 온프레미스 시스템을 Systems Manager에 등록할 수 있는 기간을 말합니다. 활성화가 만료되어도 Systems Manager에 이전에 등록한 서버나 VM에는 영향을 주지 않습니다. 활성화가 만료되면 해당 정품 인증으로는 추가로 서버나 VM을 Systems Manager에 등록할 수 없게 됩니다. 새로 만들어야 합니다.

이전에 등록한 모든 온프레미스 서버와 VM은 명시적으로 등록 취소할 때까지 Systems Manager 관리형 노드로 여전히 등록됩니다. AWS CLI 명령 deregister-managed-instance를 사용하거나 API 호출 DeregisterManagedInstance를 사용하여 Systems Manager 콘솔의 Fleet Manager에 있는 관리형 노드 탭에서 관리형 노드를 등록 취소할 수 있습니다.

관리형 노드 정보

관리형 노드는 AWS Systems Manager의 대상으로 구성된 모든 관리형 노드입니다. AWS Systems Manager는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스, 엣지 디바이스, 온프레미스 서버, 또는 다른 클라우드 환경의 VM을 포함한 가상 머신(VM)을 지원합니다. 이전에는 관리형 노드

를 모두 관리형 인스턴스라고 했습니다. 이제 인스턴스라는 용어는 EC2 인스턴스만을 지칭합니다. deregister-managed-instance 명령은 이러한 용어 변경 이전에 이름이 붙여졌습니다.

정품 인증 태그 정보

AWS Command Line Interface(AWS CLI) 또는 AWS Tools for Windows PowerShell을 사용하여 활성화를 생성하는 경우 태그를 지정할 수 있습니다. 태그는 리소스에 할당하는 선택적 메타데이터입니다. 태그를 사용하면 용도, 소유자 또는 환경을 기준으로 하는 등 리소스를 다양한 방식으로 분류할 수 있습니다. 다음은 옵션 태그가 포함된 로컬 Linux 시스템에서 실행할 AWS CLI 샘플 명령입니다.

```
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"
```

정품 인증을 생성할 때 태그를 지정하면 해당 태그는 온프레미스 서버 및 VM 활성화 시 관리형 노드에 자동으로 할당됩니다.

기존 정품 인증에 태그를 추가하거나 기존 정품 인증에서 태그를 삭제할 수 없습니다. 정품 인증을 사용하여 온프레미스 서버 및 VM에 태그를 자동으로 할당하지 않으려면 나중에 온프레미스 서버 및 VM에 태그를 추가할 수 있습니다. 더 명확히 말하면, 온프레미스 서버 및 VM이 Systems Manager에 처음으로 연결된 후 온프레미스 서버 및 VM에 태그를 지정할 수 있습니다. 온프레미스 서버 및 VM은 연결된 후 관리형 노드 ID를 할당받고 'mi-'라는 접두사가 붙는 ID와 함께 Systems Manager 콘솔에 열거됩니다. 활성화 프로세스를 사용하지 않고 관리형 노드에 태그를 추가하는 방법에 대한 자세한 내용은 관리형 노드 태깅 섹션을 참조하세요.

> **Note**
>
> Systems Manager 콘솔을 사용하여 활성화를 생성하는 경우 태그를 활성화에 할당할 수 없습니다. AWS CLI 또는 Tools for Windows PowerShell를 사용하여 생성해야 합니다.

더 이상 Systems Manager를 사용하여 온프레미스 서버 또는 가상 머신(VM)을 관리하지 않으려면 등록을 취소할 수 있습니다. 자세한 내용은 하이브리드 및 멀티클라우드 환경의 관리형 노드 등록 취소 섹션을 참조하세요.

주제

- AWS Management Console을 사용하여 Systems Manager에 관리형 노드를 등록하기 위한 활성화 생성
- 명령줄을 사용하여 Systems Manager에 관리형 노드를 등록하기 위한 활성화 생성

AWS Management Console을 사용하여 Systems Manager에 관리형 노드를 등록하기 위한 활성화 생성

관리형 노드 활성화를 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 하이브리드 정품 인증을 선택합니다.
3. 정품 인증 생성을 선택합니다.

   -또는-

   현재 AWS 리전의 하이브리드 정품 인증(Hybrid Activations)에 처음으로 액세스하는 경우 정품 인증 생성(Create an Activation)을 선택합니다.
4. (선택 사항) 정품 인증 설명(Activation description) 필드에 이 정품 인증에 대한 설명을 입력합니다. 많은 수의 서버 및 VM을 정품 인증하려는 경우 설명을 입력하는 것이 좋습니다.
5. Instance limit(인스턴스 한도)에서 AWS를 이 활성화의 일부로 사용하여 등록하려는 노드의 총 수를 지정합니다. 기본값은 인스턴스 1개입니다.
6. IAM 역할(IAM role)에서 서버와 VM이 클라우드에서 AWS Systems Manager와 통신할 수 있도록 하는 서비스 역할 옵션을 선택합니다.

   - 옵션 1: 시스템에서 생성한 기본 역할 사용(Use the default role created by the system)을 선택하여 AWS에서 제공하는 역할 및 관리형 정책을 사용합니다.
   - 옵션 2: 앞에서 생성한 선택적 사용자 정의 역할을 사용하려면 필요한 권한을 가진 기존 사용자 정의 IAM 역할 선택(Select an existing custom IAM role that has the required permissions)을 선택합니다. 이 역할에는 `"Service": "ssm.amazonaws.com"`을 지정하는 신뢰 관계 정책이 있어야 합니다. IAM 역할이 신뢰 관계 정책에서 이 원칙을 지정하지 않으면 다음 오류가 발생합니다.

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

이 역할 생성에 관한 자세한 내용은 하이브리드 및 멀티클라우드 환경에서 Systems Manager에 필요한 IAM 서비스 역할 생성 섹션을 참조하십시오.

7. 정품 인증 만료 날짜(Activation expiry date)에서 정품 인증 만료 날짜를 지정합니다. 만료 날짜는 미래 날짜여야 하며 향후 30일 이내여야 합니다. 기본값은 24시간입니다.

> **Note**
>
> 만료 날짜 이후에 추가 관리형 노드를 등록하려는 경우 새로운 활성화를 생성해야 합니다. 만료 날짜는 등록된 노드 및 실행 중인 노드에 영향을 미치지 않습니다.

8. (선택) 기본 인스턴스 이름 필드에 이 활성화와 연관된 모든 관리형 노드에 대해 표시할 이름 값을 지정합니다.
9. 정품 인증 생성을 선택합니다. Systems Manager는 즉시 활성화 코드와 ID를 콘솔에 반환합니다.

명령줄을 사용하여 Systems Manager에 관리형 노드를 등록하기 위한 활성화 생성

다음 절차에서는 AWS Command Line Interface(AWS CLI)(Linux 또는 Windows) 또는 AWS Tools for PowerShell을 사용하여 관리형 노드 활성화를 생성하는 방법을 설명합니다.

정품 인증을 생성하려면

1. 아직 하지 않은 경우 AWS CLI 또는 AWS Tools for PowerShell을 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트 및 AWS Tools for PowerShell 설치를 참조하세요.
2. 다음 명령을 실행하여 정품 인증을 생성합니다.

> **Note**
>
> - 다음 명령에서 *region*을 자신의 정보로 바꿉니다. 지원되는 *##* 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.
> - *iam-role* 파라미터에 대해 지정하는 역할에는 `"Service": "ssm.amazonaws.com"`을 지정하는 신뢰 관계 정책이 있어야 합니다. AWS Identity and Access Management(IAM) 역할이 신뢰 관계 정책에서 이 원칙을 지정하지 않으면 다음 오류가 발생합니다.

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

이 역할 생성에 관한 자세한 내용은 하이브리드 및 멀티클라우드 환경에서 Systems Manager에 필요한 IAM 서비스 역할 생성 섹션을 참조하십시오.

- `--expiration-date`의 경우 활성화 코드가 만료되는 날짜를 `"2021-07-07T00:00:00"`과 같은 타임스탬프 형식으로 제공합니다. 최대 30일 전에 날짜를 지정할 수 있습니다. 만료 날짜를 제공하지 않으면 활성화 코드는 24시간 후에 만료됩니다.

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"}
```

다음 예를 참고하세요

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"}
```

정품 인증이 성공적으로 생성되면 시스템에서 정품 인증 코드 및 ID가 즉시 반환됩니다.

하이브리드 Linux 노드에 SSM Agent를 설치하는 방법

이 주제에서는 하이브리드 및 멀티클라우드 환경의 비 EC2(Amazon Elastic Compute Cloud) Linux 시스템에 AWS Systems Manager SSM Agent를 설치하는 방법을 설명합니다. 하이브리드 및 멀티클라우드 환경에서 Windows Server 시스템을 사용할 계획인 경우 다음 단계인 하이브리드 Windows 노드에 SSM Agent를 설치하는 방법을 참조하세요.

> **Important**
>
> 이 절차는 하이브리드 및 멀티클라우드 환경의 EC2 인스턴스가 아닌 시스템 유형입니다. Linux용 EC2 인스턴스에서 SSM Agent를 다운로드하고 설치하려면 Linux용 EC2 인스턴스에 수동으로 SSM Agent 설치 및 제거 섹션을 참조하세요.

시작하기 전에 앞의 하이브리드 활성화를 생성하여 Systems Manager에 노드 등록에서 하이브리드 정품 인증을 완료한 후 받은 정품 인증 코드 및 정품 인증 ID를 찾습니다. 다음 절차에서 코드 및 ID를 지정합니다.

*##*은 미국 동부(오하이오) 리전의 `us-east-2` 같이 AWS Systems Manager가 지원하는 AWS 리전의 식별자를 나타냅니다. 지원되는 *##* 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

예를 들어 미국 동부(오하이오) 리전(us-east-2)에서 Amazon Linux, RHEL, CentOS 및 SLES 64비트용 SSM Agent를 다운로드하려면 다음 URL을 사용합니다.

```
https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/linux_amd64/amazon-ssm-agent.rpm
```

Amazon Linux 1,Amazon Linux 2, RHEL, Oracle Linux, CentOS, and SLES

- x86_64

  `https://s3.region.amazonaws.com/amazon-ssm-region/latest/linux_amd64/amazon-ssm-agent.rpm`

- x86

  `https://s3.region.amazonaws.com/amazon-ssm-region/latest/linux_386/amazon-ssm-agent.rpm`

- ARM64

```
https://s3.region.amazonaws.com/amazon-ssm-region/latest/linux_arm64/
amazon-ssm-agent.rpm
```

RHEL 6.x, CentOS 6.x

- x86_64

  ```
  https://s3.region.amazonaws.com/amazon-ssm-region/3.0.1479.0/
  linux_amd64/amazon-ssm-agent.rpm
  ```

- x86

  ```
  https://s3.region.amazonaws.com/amazon-ssm-region/3.0.1479.0/
  linux_386/amazon-ssm-agent.rpm
  ```

Ubuntu 서버

- x86_64

  ```
  https://s3.region.amazonaws.com/amazon-ssm-region/latest/debian_amd64/
  amazon-ssm-agent.deb
  ```

- ARM64

  ```
  https://s3.region.amazonaws.com/amazon-ssm-region/latest/debian_arm64/
  amazon-ssm-agent.deb
  ```

- x86

  ```
  https://s3.region.amazonaws.com/amazon-ssm-region/latest/debian_386/
  amazon-ssm-agent.deb
  ```

Debian 서버

- x86_64

  ```
  https://s3.region.amazonaws.com/amazon-ssm-region/latest/debian_amd64/
  amazon-ssm-agent.deb
  ```

- ARM64

```
https://s3.region.amazonaws.com/amazon-ssm-region/latest/debian_arm64/
amazon-ssm-agent.deb
```

Raspberry Pi OS (formerly Raspbian)

- `https://s3.region.amazonaws.com/amazon-ssm-region/latest/debian_arm/amazon-ssm-agent.deb`

하이브리드 및 멀티클라우드 환경의 비 EC2 시스템에 SSM Agent를 설치하는 방법

1. 하이브리드 및 멀티클라우드 환경의 서버 또는 VM에 로그온합니다.
2. HTTP 또는 HTTPS 프록시를 사용하는 경우 현재 셸 세션에서 `http_proxy` 또는 `https_proxy` 환경 변수를 설정해야 합니다. 프록시를 사용하지 않는 경우 이 단계를 건너뛸 수 있습니다.

   HTTP 프록시 서버의 경우 명령줄에 다음 명령을 입력합니다.

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

   HTTPS 프록시 서버의 경우 명령줄에 다음 명령을 입력합니다.

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

3. 다음 명령 블록 중 하나를 복사하여 SSH에 붙여 넣습니다. 관리형 노드 활성화를 생성할 때 생성된 활성화 코드 및 활성화 ID와 SSM Agent를 다운로드하려는 AWS 리전의 식별자로 자리 표시자 값을 바꾸고 `Enter` 키를 누릅니다.

   > **Note**
   >
   > 다음과 같은 중요 세부 정보에 주의합니다.
   >
   > - 루트 사용자인 경우 `sudo`가 필요 없습니다.
   > - 하이브리드 활성화가 생성된 동일한 AWS 리전에서 `ssm-setup-cli`를 다운로드합니다.
   > - `ssm-setup-cli`에서는 에이전트가 다운로드되는 소스를 결정하는 `manifest-url` 옵션을 지원합니다. 조직에서 요구하지 않는 한 이 옵션의 값을 지정하지 않습니다.

- 인스턴스를 등록할 때는 `ssm-setup-cli`에 대해 제공된 다운로드 링크만 사용합니다. `ssm-setup-cli`는 나중에 사용할 수 있도록 별도로 보관해서는 안 됩니다.
- 여기에 제공된 스크립트를 사용하여 `ssm-setup-cli` 서명을 검증할 수 있습니다.

*##*은 미국 동부(오하이오) 리전의 `us-east-2` 같이 AWS Systems Manager가 지원하는 AWS 리전의 식별자를 나타냅니다. 지원되는 *##* 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

또한 `ssm-setup-cli`에는 다음 옵션도 포함됩니다.

- `version` - 유효 값은 `latest` 및 `stable`입니다.
- `downgrade` - SSM Agent를 이전 버전으로 다운그레이드할 수 있습니다. 이전 버전의 에이전트를 설치하려면 `true`를 지정합니다.
- `skip-signature-validation` - 에이전트 다운로드 및 설치 중에 서명 검증을 건너뜁니다.

RHEL 6.x 및 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, 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 및 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(구 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

- .deb 패키지 사용

  ```
  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"
```

- Snap 패키지 사용

  snap 명령은 Snap 앱 스토어(https://snapcraft.io)에서 에이전트를 자동으로 다운로드하기 때문에 다운로드를 위해 URL을 지정할 필요는 없습니다.

  Ubuntu Server 20.10 STR 및 20.04, 18.04 및 16.04 LTS에서는 에이전트 바이너리 및 구성 파일을 포함한 SSM Agent 설치 관리자 파일이 `/snap/amazon-ssm-agent/current/` 디렉터리에 저장됩니다. 이 디렉터리의 구성 파일을 변경하는 경우 `/snap` 디렉터리에서 `/etc/amazon/ssm/` 디렉터리로 이러한 파일을 복사해야 합니다. 로그 및 라이브러리 파일이 변경되지 않았습니다(`/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 Store의 후보(candidate) 채널에는 최신 버전의 SSM Agent가 있습니다. 안정적인 채널은 아닙니다. 후보 채널에서 SSM Agent 버전 정보를 추적하려면 Ubuntu Server 18.04 및 16.04 LTS 64비트 관리형 노드에서 다음 명령을 실행합니다.

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

명령을 통해 하이브리드 및 멀티클라우드 환경의 하이브리드 정품 인증 시스템에 SSM Agent를 다운로드하고 설치합니다. 명령을 통해 SSM Agent를 중지한 다음에 시스템을 Systems Manager 서비스에 등록합니다. 이제 시스템이 관리형 노드입니다. Systems Manager에 대해 구성된 Amazon EC2 인스턴스도 관리형 노드입니다. 그러나 Systems Manager 콘솔에서는 하이브리드 정품 인증 노드는 접두사 "mi-"로 Amazon EC2 인스턴스와 구별됩니다.

계속해서 하이브리드 Windows 노드에 SSM Agent를 설치하는 방법로 이동하십시오.

프라이빗 키 자동 교체 설정

보안 태세를 강화하기 위해 하이브리드 및 멀티클라우드 환경의 프라이빗 키를 자동으로 교체하도록 AWS Systems Manager 에이전트(SSM Agent)를 구성할 수 있습니다. SSM Agent 버전 3.0.1031.0 이상을 사용하여 이 기능에 액세스할 수 있습니다. 다음 절차에 따라 이 기능을 설정합니다.

하이브리드 및 멀티클라우드 환경의 프라이빗 키를 교체하도록 SSM Agent를 구성하는 방법

1. `/etc/amazon/ssm/`(Linux 시스템) 또는 `C:\Program Files\Amazon\SSM`(Windows 시스템)으로 이동합니다.
2. `amazon-ssm-agent.json.template`의 내용을 `amazon-ssm-agent.json`이라는 새 파일에 복사합니다. `amazon-ssm-agent.json.template`이 위치한 동일한 디렉터리에 `amazon-ssm-agent.json`을 저장합니다.
3. `Profile`, `KeyAutoRotateDays`를 찾습니다. 자동 프라이빗 키 교체 간격(일)을 입력합니다.
4. SSM Agent을 다시 시작합니다.

구성을 변경할 때마다 SSM Agent를 다시 시작합니다.

동일한 절차를 사용하여 SSM Agent의 다른 기능을 사용자 정의할 수 있습니다. 사용 가능한 구성 속성 및 기본값의 최신 목록은 Config 속성 정의를 참조하세요.

관리형 노드 등록 취소 및 다시 등록

AWS CLI 또는 Tools for Windows PowerShell에서 DeregisterManagedInstance API 작업을 호출하여 하이브리드 정품 인증 관리형 노드 등록을 취소할 수 있습니다. 다음은 CLI 명령의 예입니다.

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

에이전트의 나머지 등록 정보를 제거하려면 `amazon-ssm-agent.json` 파일에서 `IdentityConsumptionOrder` 키를 제거합니다. 그런 다음, 다음 명령을 실행합니다.

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

시스템 등록 취소 후 다시 등록할 수 있습니다. 다음 절차에 따라 시스템을 다시 등록합니다. 이 절차를 완료하면 관리형 노드가 관리형 노드 목록에 다시 표시됩니다.

비 EC2 Linux 시스템의 관리형 노드를 다시 등록하는 방법

1. 시스템에 연결합니다.

2. 다음 명령을 실행합니다. 자리표시자 값을 관리형 노드 활성화를 생성할 때 생성된 활성화 코드 및 활성화 ID와 SSM Agent를 다운로드하려는 리전의 식별자로 바꿔야 합니다.

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

비 EC2 Linux 시스템의 SSM Agent 설치 문제 해결

다음 정보를 사용하면 하이브리드 및 멀티클라우드 환경에서 하이브리드 정품 인증 Linux 시스템에 SSM Agent를 설치하는 문제를 해결하는 데 도움이 됩니다.

DeliveryTimedOut 오류 발생

문제: 하나의 AWS 계정에 시스템을 별도의 AWS 계정에 대한 관리형 노드로 구성하는 동안 대상 시스템에 SSM Agent를 설치하는 명령을 실행한 후 `DeliveryTimedOut` 오류가 표시됩니다.

해결 방법: `DeliveryTimedOut`은 이 시나리오에 대한 예상 응답 코드입니다. 대상 노드에 SSM Agent를 설치하는 명령은 소스 노드의 노드 ID를 변경합니다. 노드 ID가 변경되었기 때문에 소스 노드는 명령 실행 중 실패, 완료 또는 시간 초과된 대상 노드에 응답할 수 없습니다.

노드 연결을 로드할 수 없음

문제: 설치 명령을 실행한 후 SSM Agent 오류 로그에 다음 오류가 표시됩니다.

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

재부팅 후 시스템 ID가 유지되지 않는 경우 이 오류가 표시됩니다.

해결 방법: 이 문제를 수정하려면 다음 명령을 실행합니다. 이 명령은 재부팅 후에도 시스템 ID가 유지되도록 합니다.

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

하이브리드 Windows 노드에 SSM Agent를 설치하는 방법

이 주제에서는 하이브리드 환경 및 멀티클라우드 환경의 Windows Server 시스템에 SSM Agent를 설치하는 방법을 설명합니다. 하이브리드 및 멀티클라우드 환경에서 비 EC2 Linux 시스템을 사용할 계획인 경우 다음 단계인 하이브리드 Linux 노드에 SSM Agent를 설치하는 방법를 참조하세요.

> **Important**
>
> 이 절차는 하이브리드 및 멀티클라우드 환경의 비 EC2 (Amazon Elastic Compute Cloud) 시스템입니다. SSM Agent를 위한 EC2 인스턴스에 Windows Server을 다운로드하고 설치하려면 SSM Agent용 EC2 인스턴스에 수동으로 Windows Server 설치 및 제거 섹션을 참조하세요.

시작하기 전에 앞의 하이브리드 활성화를 생성하여 Systems Manager에 노드 등록에서 하이브리드 정품 인증을 완료한 후 받은 정품 인증 코드 및 정품 인증 ID를 찾습니다. 다음 절차에서 코드 및 ID를 지정합니다.

하이브리드 및 멀티클라우드 환경의 비 EC2 Windows Server 시스템에 SSM Agent를 설치하는 방법

1. 하이브리드 및 멀티클라우드 환경의 서버 또는 VM에 로그온합니다.
2. HTTP 또는 HTTPS 프록시를 사용하는 경우 현재 셸 세션에서 `http_proxy` 또는 `https_proxy` 환경 변수를 설정해야 합니다. 프록시를 사용하지 않는 경우 이 단계를 건너뛸 수 있습니다.

   HTTP 프록시 서버의 경우 다음 변수를 설정합니다.

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

   HTTPS 프록시 서버의 경우 다음 변수를 설정합니다.

   ```
   http_proxy=http://hostname:port
   https_proxy=https://hostname:port
   ```
3. 승격된(관리) 권한 모드에서 Windows PowerShell을 엽니다.
4. 다음 명령 블록을 복사하여 Windows PowerShell에 붙여 넣습니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다. 예를 들면, 관리형 노드 정품 인증을 생성할 때 생성된 정품 인증 코드 및 정품 인증 ID와 SSM Agent를 다운로드하려는 AWS 리전의 식별자로 바꿔야 합니다.

> **Note**
>
> 다음과 같은 중요 세부 정보에 주의합니다.
>
> - `ssm-setup-cli`에서는 에이전트가 다운로드되는 소스를 결정하는 `manifest-url` 옵션을 지원합니다. 조직에서 요구하지 않는 한 이 옵션의 값을 지정하지 않습니다.
> - 여기에 제공된 스크립트를 사용하여 `ssm-setup-cli` 서명을 검증할 수 있습니다.
> - 인스턴스를 등록할 때는 `ssm-setup-cli`에 대해 제공된 다운로드 링크만 사용합니다. `ssm-setup-cli`는 나중에 사용할 수 있도록 별도로 보관해서는 안 됩니다.

*##*은 미국 동부(오하이오) 리전의 `us-east-2` 같이 AWS Systems Manager가 지원하는 AWS 리전의 식별자를 나타냅니다. 지원되는 *##* 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

또한 `ssm-setup-cli`에는 다음 옵션도 포함됩니다.

- `version`-유효 값은 `latest` 및 `stable`입니다.
- `downgrade` - 에이전트를 이전 버전으로 되돌립니다.
- `skip-signature-validation`-에이전트 다운로드 및 설치 중에 서명 검증을 건너뜁니다.

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. Enter을 누릅니다.

> **Note**
>
> 명령이 실패할 경우 실행하는 AWS Tools for PowerShell이 최신 버전인지 확인합니다.

명령은 다음 작업을 수행합니다.

- SSM Agent를 시스템에 다운로드하고 설치합니다.
- Systems Manager 서비스에 시스템을 등록합니다.
- 다음과 비슷한 응답을 요청에 반환합니다.

```
    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
```

이제 시스템이 관리형 노드입니다. 이러한 관리형 노드는 이제 접두사 'mi-'로 식별됩니다. AWS CLI 명령 describe-instance-information을 사용하거나 API 명령 DescribeInstanceInformation을 사용하여 Fleet Manager의 관리형 노드 페이지에서 관리형 노드를 볼 수 있습니다.

프라이빗 키 자동 교체 설정

보안 태세를 강화하기 위해 하이브리드 및 멀티클라우드 환경의 프라이빗 키를 자동으로 교체하도록 AWS Systems Manager 에이전트(SSM Agent)를 구성할 수 있습니다. SSM Agent 버전 3.0.1031.0 이상을 사용하여 이 기능에 액세스할 수 있습니다. 다음 절차에 따라 이 기능을 설정합니다.

하이브리드 및 멀티클라우드 환경의 프라이빗 키를 교체하도록 SSM Agent를 구성하는 방법

1. `/etc/amazon/ssm/`(Linux 시스템) 또는 `C:\Program Files\Amazon\SSM`(Windows Server 시스템)으로 이동합니다.
2. `amazon-ssm-agent.json.template`의 내용을 `amazon-ssm-agent.json`이라는 새 파일에 복사합니다. `amazon-ssm-agent.json.template`이 위치한 동일한 디렉터리에 `amazon-ssm-agent.json`을 저장합니다.
3. `Profile`, `KeyAutoRotateDays`를 찾습니다. 자동 프라이빗 키 교체 간격(일)을 입력합니다.
4. SSM Agent을 다시 시작합니다.

구성을 변경할 때마다 SSM Agent를 다시 시작합니다.

동일한 절차를 사용하여 SSM Agent의 다른 기능을 사용자 정의할 수 있습니다. 사용 가능한 구성 속성 및 기본값의 최신 목록은 Config 속성 정의를 참조하세요.

관리형 노드 등록 취소 및 다시 등록

AWS CLI 또는 Tools for Windows PowerShell에서 DeregisterManagedInstance API 작업을 호출하여 관리형 노드를 등록 취소할 수 있습니다. 다음은 CLI 명령의 예입니다.

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

에이전트의 나머지 등록 정보를 제거하려면 `amazon-ssm-agent.json` 파일에서 `IdentityConsumptionOrder` 키를 제거합니다. 그런 다음, 다음 명령을 실행합니다.

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

시스템 등록 취소 후 다시 등록할 수 있습니다. 다음 절차에 따라 시스템을 관리형 노드로 다시 등록합니다. 이 절차를 완료하면 관리형 노드가 관리형 노드 목록에 다시 표시됩니다.

Windows 하이브리드 시스템에서 관리형 노드를 다시 등록하려면

1. 시스템에 연결합니다.
2. 다음 명령을 실행합니다. 자리 표시자 값을 관리형 노드 활성화를 생성할 때 생성된 정품 인증 코드 및 정품 인증 ID와 SSM Agent를 다운로드하려는 리전의 식별자로 바꿔야 합니다.

```
'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"
```

Systems Manager를 통한 엣지 디바이스 관리

이 섹션에서는 계정 및 시스템 관리자가 AWS IoT Greengrass 코어 디바이스의 구성 및 관리를 활성화하기 위해 수행하는 설정 작업을 설명합니다. 이러한 태스크를 수행한 후, AWS 계정 관리자에게 권한을 부여받은 관리자는 AWS Systems Manager를 사용해 조직의 AWS IoT Greengrass 코어 디바이스를 구성하고 관리할 수 있습니다.

> **Note**
>
> - AWS IoT Greengrass용 SSM Agent는 macOS 및 Windows 10에서 지원되지 않습니다. Systems Manager 기능을 사용하여 이러한 운영 체제를 사용하는 엣지 디바이스를 관리하고 구성할 수 없습니다.
> - Systems Manager는 AWS IoT Greengrass 코어 디바이스로 구성되지 않은 엣지 디바이스도 지원합니다. Systems Manager를 사용하여 AWS IoT 코어 디바이스 및 비 AWS 엣지 디바이스를 관리하려면 하이브리드 정품 인증을 사용하여 구성해야 합니다. 자세한 내용은 하이브리드 및 멀티클라우드 환경에서 Systems Manager 사용하기 단원을 참조하십시오.
> - 엣지 디바이스에서 Session Manager 및 Microsoft 애플리케이션 패치를 사용하려면 고급 인스턴스 티어를 사용해야 합니다. 자세한 내용은 고급 인스턴스 티어 설정 단원을 참조하십시오.

시작하기 전 준비 사항

엣지 디바이스가 다음 요구 사항을 충족하는지 확인합니다.

- AWS IoT Greengrass 코어 디바이스로 구성되려면, 엣지 디바이스는 요구 사항을 충족해야 합니다. 자세한 내용은 AWS IoT Greengrass Version 2 개발자 안내서의 AWS IoT Greengrass 코어 디바이스 설정 섹션을 참조하세요.
- 엣지 디바이스는 AWS Systems Manager 에이전트(SSM Agent)와 호환되어야 합니다. 자세한 내용은 Systems Manager가 지원되는 운영 체제 단원을 참조하십시오.
- 엣지 디바이스는 클라우드의 Systems Manager 서비스와 통신할 수 있어야 합니다. Systems Manager는 연결이 끊긴 엣지 디바이스를 지원하지 않습니다.

엣지 디바이스 설정 정보

Systems Manager용 AWS IoT Greengrass 디바이스 설정에는 다음 프로세스가 포함됩니다.

> **Note**
>
> 엣지 디바이스에서의 SSM Agent 제거에 대한 자세한 내용은 AWS IoT Greengrass Version 2 개발자 안내서의 AWS Systems Manager 에이전트 제거를 참조하세요.

엣지 디바이스에 대한 IAM 서비스 역할 생성

AWS IoT Greengrass 코어 디바이스는 AWS Identity and Access Management(IAM) 서비스 역할을 사용하여 AWS Systems Manager과 통신합니다. 역할이 Systems Manager 서비스에 AWS Security Token Service(AWS STS) `AssumeRole` 신뢰를 부여합니다. 각 AWS 계정에 대해 한 번만 서비스 역할을 생성하면 됩니다. AWS IoT Greengrass 디바이스에 SSM Agent 구성 요소를 구성하고 배포할 때 `RegistrationRole` 파라미터를 위한 이 역할을 지정하게 됩니다. 하이브리드 및 멀티클라우드 환경의 비 EC2 노드를 설정하면서 이 역할을 이미 생성했다면 이 단계를 건너뛸 수 있습니다.

> **Note**
>
> 엣지 디바이스를 기반으로 Systems Manager를 사용할 회사 또는 조직의 사용자에게는 IAM에서 Systems Manager API를 호출할 수 있는 권한이 주어져야 합니다.

S3 버킷 정책 요구 사항

다음 사례 중 하나에 해당되는 경우 이 절차를 완료하기 전에 Amazon Simple Storage Service(Amazon S3) 버킷에 대한 사용자 정의 IAM 권한 정책을 생성해야 합니다.

- 사례 1: VPC 엔드포인트를 사용하여 VPC를 지원되는 AWS 서비스 및 AWS PrivateLink에 의해 제공되는 VPC 엔드포인트 서비스에 비공개로 연결하고 있습니다.
- 사례 2: S3 버킷에 대한 Run Command 명령 또는 Session Manager 세션의 출력을 저장하기 위한 작업 등 Systems Manager 작업의 일부로 생성되는 S3 버킷을 사용할 계획입니다. 진행하기 전에 인스턴스 프로파일에 대한 사용자 정의 S3 버킷 정책 생성의 단계를 수행합니다. 해당 주제의 S3 버킷 정책에 대한 정보 또한 서비스 역할에 적용됩니다.

> **Note**
>
> 디바이스가 방화벽으로 보호되고 Patch Manager를 사용할 계획인 경우, 방화벽은 패치 기준 엔드포인트 `arn:aws:s3:::patch-baseline-snapshot-region/*`에 액세스를 허용해야 합니다.
> *##*은 미국 동부(오하이오) 리전의 `us-east-2` 같이 AWS Systems Manager가 지원하는 AWS 리전의 식별자를 나타냅니다. 지원되는 *##* 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

AWS CLI

AWS IoT Greengrass 환경(AWS CLI)을 위한 IAM 서비스 역할 생성

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.

2. 로컬 컴퓨터에서 다음 신뢰 정책과 함께 `SSMService-Trust.json`과 같은 이름으로 텍스트 파일을 생성합니다. 파일을 `.json` 파일 확장명으로 저장해야 합니다.

   > **Note**
   >
   > 이름을 기록해 둡니다. AWS IoT Greengrass 코어 디바이스에 SSM Agent를 배포할 때 이를 지정하게 됩니다.

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

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

3. AWS CLI를 열고, JSON 파일을 생성한 디렉터리에서 create-role 명령을 사용하여 서비스 역할을 생성합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   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. 다음과 같이 attach-role-policy 명령을 실행하여, 방금 생성한 서비스 역할이 세션 토큰을 생성할 수 있도록 허용합니다. 세션 토큰은 Systems Manager를 사용하여 명령을 실행할 수 있도록 엣지 디바이스 권한을 부여합니다.

   > **Note**
   >
   > 엣지 디바이스를 위한 서비스 프로파일을 위해 추가하는 정책은 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스를 위한 인스턴스 프로파일을 생성하는 데 사용되는 정책과 동일합니다. 다음 명령에 사용되는 IAM 정책에 대한 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성을 참조하세요.

   (필수) 다음 명령을 실행하여 엣지 디바이스가 AWS Systems Manager 서비스 핵심 기능을 사용할 수 있게 허용합니다.

   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
```

서비스 역할을 위한 사용자 정의 S3 버킷 정책을 생성한 경우 다음 명령을 실행하여 AWS Systems Manager Agent(SSM Agent)가 정책에서 지정한 버킷에 액세스할 수 있게 합니다. *account-id* 및 *my-bucket-policy-name*을 AWS 계정 ID 및 버킷 이름으로 바꿉니다.

Linux & 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
```

(선택) 다음 명령을 실행하여 SSM Agent가 엣지 디바이스로부터 도메인 조인 요청을 위해 사용자 대신 AWS Directory Service에 액세스하도록 허용합니다. 엣지 디바이스를 Microsoft AD 디렉터리에 조인하는 경우에만 서비스 역할에 이 정책이 필요합니다.

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
```

(선택) 다음 명령을 실행하여 CloudWatch 에이전트가 엣지 디바이스에서 실행되도록 허용합니다. 이 명령을 사용하면 디바이스에서 정보를 읽고 CloudWatch에 쓸 수 있습니다. Amazon EventBridge 또는 Amazon CloudWatch Logs 등의 서비스를 사용할 경우에만 서비스 역할에 이 정책이 필요합니다.

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

Tools for PowerShell

AWS IoT Greengrass 환경(AWS Tools for Windows PowerShell)을 위한 IAM 서비스 역할 생성

1. 아직 설치하지 않은 경우 AWS Tools for PowerShell(Tools for Windows PowerShell)을 설치하고 구성합니다.

   자세한 내용은 AWS Tools for PowerShell 설치를 참조하세요.
2. 로컬 컴퓨터에서 다음 신뢰 정책과 함께 `SSMService-Trust.json`과 같은 이름으로 텍스트 파일을 생성합니다. 파일을 `.json` 파일 확장명으로 저장해야 합니다.

   > **Note**
   >
   > 이름을 기록해 둡니다. AWS IoT Greengrass 코어 디바이스에 SSM Agent를 배포할 때 이를 지정하게 됩니다.

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

```
    }
}
```

3. 관리 모드에서 PowerShell을 열고 JSON 파일을 생성한 디렉터리에서 New-IAMRole을 실행해 다음과 같이 서비스 역할을 생성합니다.

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

4. 다음과 같이 Register-IAMRolePolicy를 사용하여, 생성한 서비스 역할이 세션 토큰을 생성할 수 있도록 허용합니다. 세션 토큰은 Systems Manager를 사용하여 명령을 실행할 수 있도록 엣지 디바이스 권한을 부여합니다.

> **Note**
>
> AWS IoT Greengrass 환경의 엣지 디바이스를 위한 서비스 역할을 위해 추가하는 정책은 EC2 인스턴스를 위한 인스턴스 프로파일을 생성하는 데 사용되는 정책과 동일합니다. 다음 명령에 사용되는 AWS 정책에 대한 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성을 참조하세요.

(필수) 다음 명령을 실행하여 엣지 디바이스가 AWS Systems Manager 서비스 핵심 기능을 사용할 수 있게 허용합니다.

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

서비스 역할을 위한 사용자 정의 S3 버킷 정책을 생성한 경우 다음 명령을 실행하여 SSM Agent가 정책에서 지정한 버킷에 액세스할 수 있게 허용합니다. *account-id* 및 *my-bucket-policy-name*을 AWS 계정 ID 및 버킷 이름으로 바꿉니다.

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

(선택) 다음 명령을 실행하여 SSM Agent가 엣지 디바이스로부터 도메인 조인 요청을 위해 사용자 대신 AWS Directory Service에 액세스하도록 허용합니다. 엣지 디바이스를 Microsoft AD 디렉터리에 조인하는 경우에만 서비스 역할에 이 정책이 필요합니다.

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

(선택) 다음 명령을 실행하여 CloudWatch 에이전트가 엣지 디바이스에서 실행되도록 허용합니다. 이 명령을 사용하면 디바이스에서 정보를 읽고 CloudWatch에 쓸 수 있습니다. Amazon EventBridge 또는 Amazon CloudWatch Logs 등의 서비스를 사용할 경우에만 서비스 역할에 이 정책이 필요합니다.

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

AWS IoT Greengrass를 위한 엣지 디바이스를 구성

엣지 디바이스를 AWS IoT Greengrass 코어 디바이스로 설정합니다. 설치 프로세스에는 디바이스에 AWS IoT Greengrass 코어 소프트웨어를 설치하고 구성하는 것뿐만 아니라 지원되는 운영 체제 및 시스템 요구 사항을 확인하는 일이 포함됩니다. 자세한 내용은 AWS IoT Greengrass 개발자 안내서의 AWS IoT Greengrass Version 2 코어 디바이스 설정 섹션을 참조하세요.

AWS IoT Greengrass 토큰 교환 역할 업데이트 및 엣지 디바이스에 SSM Agent 설치

Systems Manager용 AWS IoT Greengrass 코어 디바이스 설정 및 구성을 위한 마지막 단계에서는 토큰 교환 역할이라고도 부르는 AWS IoT Greengrass AWS Identity and Access Management(IAM) 디바이스 서비스 역할을 업데이트해야 하며 AWS IoT Greengrass 디바이스에 AWS Systems Manager 에이전트(SSM Agent)를 배포해야 합니다. 이러한 프로세스에 대한 자세한 내용은 AWS IoT Greengrass Version 2 개발자 안내서의 AWS Systems Manager 에이전트 설치를 참조하세요.

디바이스에 SSM Agent를 배포하고 나면, AWS IoT Greengrass가 자동으로 Systems Manager에 디바이스를 등록합니다. 추가 등록은 필요하지 않습니다. Systems Manager 기능을 사용하여 AWS IoT Greengrass 디바이스의 액세스, 관리 및 구성 작업을 시작할 수 있습니다.

> **Note**
>
> 엣지 디바이스는 클라우드의 Systems Manager 서비스와 통신할 수 있어야 합니다. Systems Manager는 연결이 끊긴 엣지 디바이스를 지원하지 않습니다.

Systems Manager에 대한 AWS Organizations 위임 관리자 생성

AWS Organizations에서 조직을 설정할 때 모든 AWS 서비스의 모든 관리 작업을 수행하는 관리 계정을 할당합니다. 관리 계정 사용자는 Systems Manager에서 Change Manager, Explorer 및 OpsCenter의 관리 작업을 수행할 수 있는 위임된 관리자 계정만 할당할 수 있습니다. AWS Organizations는 조직을 생성하고 이러한 계정을 중앙에서 관리하는 AWS 계정를 할당하는 데 사용할 수 있는 계정 관리 서비스입니다. AWS Organizations에 대한 자세한 내용은 AWS Organizations 사용 설명서의 AWS Organizations 섹션을 참조하십시오.

AWS Systems Manager의 기능인 Change Manager, Explorer 및 OpsCenter에서는 AWS Organizations로 작업하여 조직의 모든 멤버 계정에 대한 작업을 수행합니다. 모든 Systems Manager의 위임된 관리자를 한 사람만 할당할 수 있습니다. 위임된 관리자 계정은 할당된 조직 단위의 멤버여야 합니다.

주제

- Change Manager에 위임된 관리자 사용
- Explorer에 위임된 관리자 사용
- OpsCenter에 위임된 관리자 사용

Change Manager에 위임된 관리자 사용

Change Manager는 애플리케이션 구성 및 인프라에 대한 운영 변경을 요청, 승인, 구현 및 보고하기 위한 엔터프라이즈 변경 관리 프레임워크입니다.

조직 전체에서 Change Manager를 사용하는 경우 위임된 관리자 계정을 할당하여 모든 멤버 계정의 변경 템플릿, 승인 및 보고를 관리합니다. 빠른 설정을 사용하면 조직에서 사용하는 Change Manager를 설정하고 위임된 관리자 계정을 선택할 수 있습니다. AWS 계정 하나로 Change Manager를 사용하는 경우 위임된 관리자 계정이 필요하지 않습니다.

기본적으로 Change Manager에는 위임된 관리자 계정의 모든 변경 관련 작업이 표시됩니다. 조직의 Change Manager를 설정하는 동안 위임된 관리자를 구성하는 방법에 대한 지침은 조직용 Change Manager 설정(관리 계정)을 참조하세요.

> **Important**
>
> 조직 전체에서 Change Manager를 사용하는 경우 항상 위임된 관리자 계정에서 변경하는 것이 좋습니다. 조직의 다른 계정에서 변경할 수 있지만 이러한 변경 사항은 위임된 관리자 계정에서 보고되거나 볼 수 없습니다.

Explorer에 위임된 관리자 사용

Explorer는 전체 AWS 리전에서 AWS 계정에 대한 운영 데이터(OpsData)를 집계한 보기를 보고하는 사용자 지정 가능한 운영 대시보드입니다.

AWS Organizations에서 리소스 데이터 동기화를 사용하여 여러 리전 및 계정의 Explorer 데이터를 집계하도록 Systems Manager의 위임된 관리자를 구성할 수 있습니다. 위임된 관리자는 AWS Management Console, AWS Command Line Interface(AWS CLI) 또는 AWS Tools for Windows PowerShell을 사용하여 Explorer 데이터를 검색, 필터링 및 집계할 수 있습니다.

Explorer의 위임된 관리자 계정을 사용하면 위임된 관리자는 다중 계정 및 리전 리소스 데이터 동기화를 생성하거나 삭제할 수 있는 관리자 수가 개별 AWS 계정으로 제한됩니다.

Explorer를 사용하여 조직의 모든 AWS 계정 계정에서 작업 데이터를 동기화할 수 있습니다. Explorer에서 위임된 관리자를 할당하는 방법에 대한 자세한 내용은 위임된 관리자 구성를 참조하세요.

OpsCenter에 위임된 관리자 사용

OpsCenter에서는 운영 엔지니어 및 IT 전문가가 AWS 리소스와 관련된 운영 작업 항목(OpsItems)을 관리할 수 있는 중앙 위치를 제공합니다. OpsCenter를 사용하여 여러 계정의 OpsItems를 중앙에서 관리하려면 AWS Organizations에서 조직을 설정해야 합니다.

OpsCenter에 Quick Setup을 사용하여 위임된 관리자 계정을 할당하고 중앙에서 OpsItems를 관리하도록 OpsCenter를 구성할 수 있습니다. 자세한 내용은 (선택 사항) Quick Setup을 사용하여 여러 계정의 OpsItems를 관리하도록 OpsCenter 구성 단원을 참조하십시오.

AWS Systems Manager에 대한 일반 설정

아직 가입하지 않았다면 AWS 계정에 가입하고 관리자를 생성합니다.

AWS 계정에 등록

AWS 계정이 없는 경우 다음 절차에 따라 계정을 생성합니다.

AWS 계정에 등록하려면

1. https://portal.aws.amazon.com/billing/signup을 여세요.
2. 온라인 지시 사항을 따르세요.

   등록 절차 중에는 전화를 받고 키패드로 인증 코드를 입력하는 과정이 있습니다.

   AWS 계정에 가입하면 AWS 계정 루트 사용자이 생성됩니다. 루트 사용자에게는 계정의 모든 AWS 서비스 및 리소스 액세스 권한이 있습니다. 보안 모범 사례는 사용자에게 관리 액세스 권한을 할당하고, 루트 사용자만 사용하여 루트 사용자 액세스 권한이 필요한 작업을 수행하는 것입니다.

AWS는 가입 절차 완료된 후 사용자에게 확인 이메일을 전송합니다. 언제든지 https://aws.amazon.com/으로 가서 내 계정(My Account)을 선택하여 현재 계정 활동을 보고 계정을 관리할 수 있습니다.

관리자 액세스 권한이 있는 사용자 생성

AWS 계정에 가입하고 AWS 계정 루트 사용자에 보안 조치를 한 다음, AWS IAM Identity Center를 활성화하고 일상적인 작업에 루트 사용자를 사용하지 않도록 관리 사용자를 생성합니다.

귀하의 AWS 계정 루트 사용자 보호

1. 루트 사용자를 선택하고 AWS 계정이메일 주소를 입력하여 AWS Management Console에 계정 소유자로 로그인합니다. 다음 페이지에서 비밀번호를 입력합니다.

   루트 사용자를 사용하여 로그인하는 데 도움이 필요하면 AWS 로그인 사용 설명서의 루트 사용자로 로그인을 참조하세요.
2. 루트 사용자의 다중 인증(MFA)을 활성화합니다.

   지침은 IAM 사용 설명서의 AWS 계정루트 사용자용 가상 MFA 디바이스 활성화(콘솔)를 참조하세요.

관리자 액세스 권한이 있는 사용자 생성

1. IAM Identity Center를 활성화합니다.

   지침은 AWS IAM Identity Center 사용 설명서의 AWS IAM Identity Center 설정을 참조하세요.
2. IAM Identity Center에서 사용자에게 관리 액세스 권한을 부여합니다.

IAM Identity Center 디렉터리를 ID 소스로 사용하는 방법에 대한 자습서는 AWS IAM Identity Center 사용 설명서의 기본 IAM Identity Center 디렉터리로 사용자 액세스 구성을 참조하세요.

관리 액세스 권한이 있는 사용자로 로그인

- IAM IDentity Center 사용자로 로그인하려면 IAM IDentity Center 사용자를 생성할 때 이메일 주소로 전송된 로그인 URL을 사용합니다.

  IAM Identity Center 사용자로 로그인하는 데 도움이 필요한 경우 AWS 로그인 사용 설명서의 AWS 액세스 포털에 로그인을 참조하세요.

추가 사용자에게 액세스 권한 할당

1. IAM Identity Center에서 최소 권한 적용 모범 사례를 따르는 권한 세트를 생성합니다.

   지침은 AWS IAM Identity Center 사용 설명서의 Create a permission set를 참조하세요.
2. 사용자를 그룹에 할당하고, 그룹에 Single Sign-On 액세스 권한을 할당합니다.

   지침은 AWS IAM Identity Center 사용 설명서의 Add groups를 참조하세요.

Systems Manager를 사용하여 관리 태스크 수행

이 자습서를 사용하여 AWS Systems Manager를 시작하세요. Systems Manager에서 관리하는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스를 시작하는 방법과 관리형 인스턴스에 연결하는 방법을 배웁니다.

Systems Manager는 여러 기능의 모음이기 때문에 단일 시연이나 튜토리얼로 전체 서비스를 소개할 수 없습니다. 이 자습서에서는 몇 가지 기능을 소개합니다.

필수 조건

시작하기 전에 먼저 EC2 인스턴스에 Systems Manager 사용의 단계를 완료해야 합니다.

SSM Agent가 사전 설치된 AMI를 사용하여 인스턴스 시작

다음 절차의 설명에 따라 AWS Management Console을 사용하여 Amazon EC2 인스턴스를 시작할 수 있습니다. 이 자습서는 첫 번째 관리형 인스턴스를 빠르게 시작하도록 돕기 위한 것이므로 가능한 모든 옵션을 다루지는 않습니다.

인스턴스 시작

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. EC2 콘솔 대시보드의 인스턴스 시작(Launch instance) 대화 상자에서 인스턴스 시작(Launch instance)을 선택한 다음, 표시되는 옵션에서 인스턴스 시작(Launch instance)을 선택합니다.
3. 이름 및 태그 아래의 이름에 인스턴스를 설명하는 이름을 입력합니다.
4. 애플리케이션 및 OS 이미지(Amazon Machine Image)에서 다음을 수행합니다.
   a. 빠른 시작 탭을 선택한 다음 Amazon Linux를 선택합니다. 인스턴스의 운영 체제(OS)입니다.
   b. Amazon Machine Image(AMI)에서 Amazon Linux 2의 HVM 버전을 선택합니다.
5. 인스턴스 유형의 인스턴스 유형 목록에서 인스턴스의 하드웨어 구성을 선택합니다. 기본적으로 선택된 `t2.micro` 인스턴스 유형을 선택합니다. `t2.micro` 인스턴스 유형은 AWS 프리 티어로 이용할 수 있습니다. `t2.micro`를 사용할 수 없는 AWS 리전에서는 프리 티어로 `t3.micro` 인스턴스를 사용할 수 있습니다. 자세한 내용은 AWS 프리 티어를 참조하세요.
6. 키 페어(로그인)에서 키 페어 이름에 대해 키 페어를 선택합니다.

7. 네트워크 설정에서 편집을 선택합니다. 보안 그룹 이름에서 마법사가 보안 그룹을 생성하고 선택했음을 확인합니다. 이 보안 그룹을 사용하거나 다음 단계를 사용하여 이전에 생성한 보안 그룹을 선택할 수 있습니다.

   a. 기존 보안 그룹 선택(Select an existing security group)을 선택합니다.

   b. 공통 보안 그룹(Common security groups)의 기존 보안 그룹 목록에서 보안 그룹을 선택합니다.

8. 기본 호스트 관리 구성을 사용하지 않는 경우 고급 세부 정보 섹션을 확장하고 IAM 인스턴스 프로파일에서 Systems Manager에 필요한 인스턴스 권한 구성에서 설정할 때 생성한 인스턴스 프로파일을 선택합니다.

9. 인스턴스의 다른 구성 설정에 대한 기본 선택 사항을 유지합니다.

10. 요약 창에서 인스턴스 구성 요약을 검토합니다. 준비가 완료되면 인스턴스 시작을 선택합니다.

11. 확인 페이지를 통해 인스턴스가 시작 중임을 알 수 있습니다. 모든 인스턴스 보기(View all instances)를 선택하여 확인 페이지를 닫고 콘솔로 돌아갑니다.

12. 인스턴스 화면에서 시작 상태를 볼 수 있습니다. 인스턴스를 출범하는 데 약간 시간이 걸립니다.

13. 인스턴스가 관리형으로 표시되고 인스턴스에 연결할 준비가 될 때까지 몇 분 정도 걸릴 수 있습니다. 인스턴스가 상태 확인을 통과했는지 확인하기 위해 상태 검사 열에서 이 정보를 볼 수 있습니다.

Systems Manager를 사용하여 관리형 인스턴스에 연결

관리형 인스턴스에 연결

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.

2. 탐색 창에서 Fleet Manager를 선택합니다.

3. 연결하려는 인스턴스 옆에 있는 버튼을 선택합니다.

4. 노드 작업 메뉴에서 터미널 세션 시작을 선택합니다.

5. 연결(Connect)을 선택합니다.

인스턴스 정리

이 자습서용으로 생성한 관리형 인스턴스 작업을 마쳤으면 종료합니다. 인스턴스를 종료하면 효과적으로 삭제됩니다.

인스턴스를 종료하려면

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 Instances(인스턴스)를 선택합니다. 인스턴스 목록에서 인스턴스를 선택합니다.
3. 인스턴스 상태, 인스턴스 종료를 차례로 선택합니다.
4. 확인 메시지가 나타나면 종료를 선택합니다.

   Amazon EC2가 인스턴스를 종료합니다. 인스턴스는 종료한 후에도 일시적으로 콘솔에서 표시된 상태로 유지되며, 그 이후 항목이 자동으로 삭제됩니다. 종료된 인스턴스는 콘솔 디스플레이에서 직접 제거할 수 없습니다.

SSM Agent 작업

AWS Systems Manager Agent(SSM Agent)는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스, 엣지 디바이스, 온프레미스 서버 또는 가상 머신에서 실행되는 Amazon 소프트웨어입니다. SSM Agent를 사용하면 Systems Manager에서 이러한 리소스를 업데이트, 관리 및 구성할 수 있습니다. 에이전트는 AWS 클라우드에서 Systems Manager 서비스로부터 요청을 처리한 다음, 이를 요청에서 지정한 것과 같이 실행합니다. 이후 SSM Agent에서 Amazon Message Gateway Service(ssmmessages)를 사용하여 상태 및 실행 정보를 Systems Manager 서비스로 다시 보냅니다. (2024년 이전에 출시된 AWS 리전의 경우 Amazon Message Delivery Service(서비스 접두사: ec2messages)를 통해 상태 및 실행 정보가 다시 전송될 수도 있습니다.)

트래픽을 모니터링하면 관리형 노드가 ssmmessages.* 엔드포인트 및 가능한 경우 ec2messages.* 엔드포인트트와 통신하는 것을 볼 수 있습니다. 자세한 내용은 참조: ec2messages, ssmmessages 및 기타 API 작업 단원을 참조하십시오. Amazon CloudWatch Logs에 SSM Agent 포팅에 대한 자세한 내용은 AWS Systems Manager 모니터링 섹션을 참조하세요.

내용

SSM Agent에 대한 기술 세부 정보를 알아보기

이 주제의 정보를 사용하여 AWS Systems Manager Agent(SSM Agent)를 구현하고 에이전트의 작동 방식을 이해합니다.

주제

- SSM Agent가 사전 설치된 상태로 AMIs 검색
- Linux용 EC2 인스턴스에서 SSM Agent 사용
- macOS용 EC2 인스턴스에서 SSM Agent 사용
- Windows Server용 EC2 인스턴스에서 SSM Agent 사용
- SSM Agent 상태 확인 및 에이전트 시작
- SSM Agent 버전 번호 확인
- SSM Agent 로그 보기
- SSM Agent를 통한 루트 수준 명령에 대한 액세스 제한
- SSM Agent 업데이트 자동화
- SSM Agent 알림 구독

SSM Agent 버전 3.2.x.x 보안 인증 정보 동작

SSM Agent에서는 Quick Setup에서 기본 호스트 관리 구성을 사용하여 인스턴스를 온보딩할 때 `/var/lib/amazon/ssm/credentials`(Linux 및 macOS의 경우) 또는 `%PROGRAMFILES%\Amazon\SSM\credentials`(Windows Server의 경우)에서 임시 보안 인증 정보 세트를 저장합니다. 임시 보안 인증 정보에는 기본 호스트 관리 구성을 위해 선택한 IAM 역할에 대해 지정하는 권한이 있습니다. Linux에서는 루트 계정에서만 이러한 보안 인증 정보에 액세스할 수 있습니다. Windows Server에서는 SYSTEM 계정과 로컬 관리자만 이러한 보안 인증 정보에 액세스할 수 있습니다.

SSM Agent 자격 증명 우선순위

다음 주제에서는 SSM Agent가 리소스에서 작업을 수행하기 위해 권한을 부여하는 방법에 대한 중요한 정보를 설명합니다.

> **Note**
>
> 엣지 디바이스에 대한 지원은 약간 다릅니다. AWS IoT Greengrass 코어 소프트웨어를 사용하고, AWS Identity and Access Management(IAM) 서비스 역할을 구성하고, AWS IoT Greengrass를 사용하여 디바이스에 SSM Agent를 배포하도록 엣지 디바이스를 구성해야 합니다. 자세한 내용은 Systems Manager를 통한 엣지 디바이스 관리 단원을 참조하십시오.

SSM Agent가 시스템에 설치되면 Systems Manager 서비스와 통신할 권한이 필요합니다. Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에서 이러한 권한은 인스턴스에 연결된 인스

턴스 프로파일에 제공됩니다. 비 EC2 시스템의 SSM Agent에서는 일반적으로 `/root/.aws/credentials`(Linux 및 macOS) 또는 `%USERPROFILE%\.aws\credentials`(Windows Server)에 있는 공유 보안 인증 정보 파일에서 필요한 권한을 가져옵니다. 필요한 권한은 하이브리드 정품 인증 프로세스 동안에 이 파일에 추가됩니다.

그러나 드물지만 SSM Agent에서 해당 작업을 실행할 수 있는 권한을 확인하는 2개 이상의 위치에 권한이 시스템에 추가될 수 있습니다.

예를 들면, Systems Manager에서 관리하도록 EC2 인스턴스를 구성할 수 있습니다. 해당 구성에는 인스턴스 프로파일 연결이 포함됩니다. 그러나 개발자 또는 최종 사용자 작업에도 해당 인스턴스를 사용하기로 결정하고 여기에 AWS Command Line Interface(AWS CLI)를 설치합니다. 이렇게 설치하면 인스턴스의 자격 증명 파일에 권한이 더 추가됩니다.

인스턴스에서 Systems Manager 명령을 실행할 때 SSM Agent는 인스턴스 프로파일 대신 자격 증명 파일에서와 같이 예상과 다른 자격 증명을 사용하려고 할 수 있습니다. 이는 SSM Agent가 기본 자격 증명 공급자 체인에 대해 규정된 순서대로 자격 증명을 찾기 때문입니다.

> **Note**
>
> Linux 및 macOS에서 SSM Agent는 루트 사용자로 실행됩니다. 따라서 이 프로세스에서 SSM Agent는 루트 사용자의 환경 변수와 자격 증명 파일(`/root/.aws/credentials`)만 찾습니다. SSM Agent는 자격 증명을 검색하는 동안 인스턴스에 있는 다른 사용자의 환경 변수나 자격 증명 파일을 확인하지 않습니다.

기본 공급자 체인은 다음 순서대로 자격 증명을 찾습니다.

1. 환경 변수(구성된 경우)(`AWS_ACCESS_KEY_ID` 및 `AWS_SECRET_ACCESS_KEY`).
2. 하이브리드 활성화 또는 AWS CLI 설치 등에 의해 제공되는 권한이 있는 공유 자격 증명 파일(Linux의 경우 `$HOME/.aws/credentials` 및 Windows Server의 경우 macOS 또는 `%USERPROFILE%\.aws\credentials`).
3. Amazon Elastic Container Service(Amazon ECS) 태스크 정의 또는 RunTask API 작업을 사용하는 애플리케이션이 있는 경우 태스크에 대한 AWS Identity and Access Management(IAM) 역할.
4. Amazon EC2 인스턴스에 연결된 인스턴스 프로파일.
5. 기본 호스트 관리 구성에 선택된 IAM 역할입니다.

관련 내용은 다음 주제를 참조하세요.

- EC2 인스턴용 인스턴스 프로파일 - Systems Manager에 필요한 인스턴스 권한 구성
- 하이브리드 활성화 - 하이브리드 활성화를 생성하여 Systems Manager에 노드 등록
- AWS CLI 자격 증명 - AWS Command Line Interface 사용 설명서의 구성 및 자격 증명 파일 설정
- 기본 자격 증명 공급자 체인 - AWS SDK for Go Developer Guide의 Specifying Credentials

> **Note**
>
> AWS SDK for Go Developer Guide의 이 주제에서는 SDK for Go 측면에서 기본 공급자 체인을 설명합니다. 그러나 SSM Agent에 대한 자격 증명을 평가할 때도 동일한 원칙이 적용됩니다.

로컬 ssm-user 계정 정보

SSM Agent 버전 2.3.50.0부터 에이전트는 `ssm-user`라는 로컬 사용자 계정을 생성하여 `/etc/sudoers.d` 디렉터리(Linux 및 macOS)나 Administrators group (Windows Server)에 추가합니다. 2.3.612.0 이전 버전의 에이전트에서 SSM Agent 최초 시작 또는 설치 후 재시작 시 계정이 생성됩니다. 2.3.612.0 이후 버전에서 `ssm-user` 계정은 인스턴스에서 세션을 처음 시작할 때 생성됩니다. 이 `ssm-user`는 AWS Systems Manager의 기능인 Session Manager에서 세션이 시작될 때 기본 OS 사용자입니다. `ssm-user`를 권한이 낮은 그룹으로 이동하거나 `sudoers` 파일을 변경하여 권한을 변경할 수 있습니다. SSM Agent를 설치 제거해도 `ssm-user` 계정은 시스템에서 제거되지 않습니다.

Windows Server에서 SSM Agent는 각 세션이 시작될 때 `ssm-user` 계정에 대한 새 암호 설정을 처리합니다. Linux 관리형 인스턴스의 `ssm-user`에 대해서는 암호가 설정되지 않습니다.

SSM Agent 버전 2.3.612.0부터 `ssm-user` 계정은 도메인 컨트롤러로 사용되는 Windows Server 시스템에서 자동으로 생성되지 않습니다. Windows Server 도메인 컨트롤러에서 Session Manager를 사용하려면 `ssm-user` 계정이 없는 경우 수동으로 생성하고 사용자에게 도메인 관리자 권한을 할당합니다.

> **Important**
>
> `ssm-user` 계정을 생성하려면 인스턴스에 연결된 인스턴스 프로파일에서 필요한 권한을 제공해야 합니다. 자세한 내용은 2단계: Session Manager에 대한 인스턴스 권한 확인 또는 추가를 참조하세요.

SSM Agent 및 Instance Metadata Service(IMDS)

Systems Manager는 올바르게 작동하기 위해 EC2 인스턴스 메타데이터를 사용합니다. Systems Manager는 Instance Metadata Service의 버전 1 또는 버전 2(IMDSv1 및 IMDSv2)를 사용하여 인스턴스 메타데이터에 액세스할 수 있습니다. 인스턴스에서 인스턴스 메타데이터 서비스의 IPv4 주소 169.254.169.254에 액세스할 수 있어야 합니다. 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스 메타데이터 및 사용자 데이터를 참조하세요.

SSM Agent 최신 상태 유지

SSM Agent의 업데이트된 버전은 Systems Manager에 새 기능이 추가되거나 기존 기능에 업데이트가 발생할 때마다 릴리스됩니다. 최신 버전의 에이전트를 사용하지 못하면 관리형 노드에서 다양한 Systems Manager 기능을 사용하지 못할 수 있습니다. 이러한 이유로 시스템의 SSM Agent 상태를 최신 상태로 유지하는 프로세스를 자동화하는 것이 좋습니다. 자세한 설명은 SSM Agent 업데이트 자동화을 참조하세요. SSM Agent 업데이트에 대해 알림을 수신하려면 GitHub에서 SSM Agent 릴리스 정보 페이지를 구독합니다.

> **Note**
>
> SSM Agent의 업데이트된 버전은 Systems Manager에 새 기능이 추가되거나 기존 기능에 업데이트가 발생할 때마다 릴리스됩니다. 최신 버전의 에이전트를 사용하지 못하면 관리형 노드에서 다양한 Systems Manager 기능을 사용하지 못할 수 있습니다. 이러한 이유로 시스템의 SSM Agent 상태를 최신 상태로 유지하는 프로세스를 자동화하는 것이 좋습니다. 자세한 설명은 SSM Agent 업데이트 자동화을 참조하세요. SSM Agent 업데이트에 대해 알림을 수신하려면 GitHub에서 SSM Agent 릴리스 정보 페이지를 구독합니다.
>
> SSM Agent가 기본적으로 포함된 최신 버전의 SSM Agent를 사용하는 Amazon Machine Images(AMIs)를 업데이트하는 데 최대 2주가 걸릴 수 있습니다. SSM Agent에 대해 더욱 주기적인 자동 업데이트를 구성하는 것이 좋습니다.

SSM Agent 설치 디렉터리가 수정, 이동 또는 삭제되지 않았는지 확인

SSM Agent는 `/var/lib/amazon/ssm/`(Linux와 macOS) 및 `%PROGRAMFILES%\Amazon\SSM\`(Windows Server)에 설치됩니다. 이러한 설치 디렉터리에는 보안 인증 파일, IPC(프로세스 간 통신)용 리소스, 오케스트레이션 폴더 등 SSM Agent에서 사용하는 중요한 파일 및 폴더가 포함되어 있습니다. 설치 디렉터리 내의 어떤 내용도 수정, 이동 또는 삭제해서는 안 됩니다. 그렇지 않으면 SSM Agent가 제대로 작동하지 않을 수 있습니다.

AWS 리전의 SSM Agent 롤링 업데이트

GitHub 리포지토리에서 SSM Agent 업데이트를 사용할 수 있게 된 후 업데이트된 버전이 다른 시간에 모든 AWS 리전에 배포될 때까지 최대 2주가 소요될 수 있습니다. 이러한 이유로 리전에서 새 버전의 SSM Agent를 배포하려고 하면 "현재 플랫폼에서 지원되지 않음(Unsupported on current platform)" 또는 "amazon-ssm-agent를 이전 버전으로 업데이트 중, 진행하려면 다운그레이드 허용 설정(updating amazon-ssm-agent to an older version, please turn on allow downgrade to proceed)"과 같은 오류가 표시될 수 있습니다.

사용 가능한 SSM Agent 버전을 확인하려면 `curl` 명령을 실행합니다.

글로벌 다운로드 버킷에서 사용 가능한 에이전트 버전을 보려면 다음 명령을 실행합니다.

```
curl https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/VERSION
```

특정 리전에서 사용 가능한 에이전트 버전을 보려면 *##*을 작업 중인 리전(예: 미국 동부(오하이오) 리전의 경우 `us-east-2`)으로 바꿔서 다음 명령을 실행합니다.

```
curl https://s3.region.amazonaws.com/amazon-ssm-region/latest/VERSION
```

`curl` 명령 없이 브라우저에서 직접 `VERSION` 파일을 열 수도 있습니다.

AWS 관리형 S3 버킷과 SSM Agent 통신

다양한 Systems Manager 작업을 수행하는 과정에서 AWS Systems Manager Agent(SSM Agent)는 여러 Amazon Simple Storage Service(Amazon S3) 버킷에 액세스합니다. 이러한 S3 버킷은 공개적으로 액세스할 수 있으며 기본적으로 SSM Agent는 HTTP 호출을 사용하여 연결합니다.

그러나 Systems Manager 작업에서 Virtual Private Cloud(VPC) 엔드포인트를 사용하는 경우 Systems Manager를 위한 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 프로파일 또는 하이브리드 및 멀티클라우드 환경의 비 EC2 시스템을 위한 서비스 역할에 명시적 권한을 제공해야 합니다. 그렇지 않을 경우, 리소스가 이러한 퍼블릭 버킷에 액세스할 수 없습니다.

VPC 엔드포인트를 사용할 때 관리형 노드에 이러한 버킷에 대한 액세스 권한을 부여하려면 사용자 정의 Amazon S3 권한 정책을 생성한 다음에 인스턴스 프로파일(EC2 인스턴스의 경우) 또는 서비스 역할(비 EC2 관리형 노드의 경우)에 이를 연결합니다.

Systems Manager 작업에서 Virtual Private Cloud(VPC) 엔드포인트를 사용하는 방법에 대한 자세한 내용은 Systems Manager용 VPC 엔드포인트를 사용하여 EC2 인스턴스의 보안 개선을 참조하세요.

> **Note**
>
> 이러한 권한은 SSM Agent에 필요한 AWS 관리형 버킷에 대한 액세스만 제공합니다. 다른 Amazon S3 작업에 필요한 권한은 제공하지 않습니다. 또한 자체 S3 버킷에 대해서도 권한을 제공하지 않습니다.

자세한 정보는 다음 주제를 참조하세요.

- Systems Manager에 필요한 인스턴스 권한 구성
- 하이브리드 및 멀티클라우드 환경에서 Systems Manager에 필요한 IAM 서비스 역할 생성

내용

필요한 버킷 권한

다음 표에서는 SSM Agent가 Systems Manager 작업을 위해 액세스해야 할 수 있는 각 S3 버킷에 대해 설명합니다.

> **Note**
>
> ##은 미국 동부(오하이오) 리전의 `us-east-2` 같이 AWS Systems Manager가 지원하는 AWS 리전의 식별자를 나타냅니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

SSM Agent에 필요한 Amazon S3 권한

S3 버킷 ARN	설명
arn:aws:s3:::aws-windows-downloads- *region*/*	Windows Server 운영 체제만 지원되는 일부 SSM 문서와 교차 플랫폼 지원용 문서(예: AWSEC2-ConfigureSTIG)에 필요합니다.
arn:aws:s3:::amazon-ssm- *region*/*	SSM Agent 설치를 업데이트하는 데 필요합니다. 이 버킷에는 SSM Agent 설치 패키지와 AWS-UpdateSSMAgent 문서 및 플러그인에서 참조되는 설치 매니페스트가 포함되어 있습니다. 이러한 권한이 제공되지 않으면 SSM Agent에서는 HTTP 호출을 통해 업데이트를 다운로드합니다.
arn:aws:s3:::amazon-ssm-packages- *region*/*	2.2.45.0 이전의 SSM Agent 버전을 사용하여 SSM 문서 AWS-ConfigureAWSPackage 를 실행하는 데 필요합니다.
arn:aws:s3:: *region*-birdwatcher-prod/*	버전 2.2.45.0 이후의 SSM Agent에 사용되는 배포 서비스에 대한 액세스를 제공합니다. 이 서비스는 AWS-ConfigureAWSPackage 문서를 실행하는 데 사용합니다. 이 권한은 아프리카(케이프타운) 리전(af-south-1)과 유럽(밀라노) 리전(eu-south-1)을 제외한 모든 AWS 리전에 필요합니다.
arn:aws:s3:::aws-ssm-distributor-file- *region*/*	버전 2.2.45.0 이후의 SSM Agent에 사용되는 배포 서비스에 대한 액세스를 제공합니다. 이 서비스는 SSM 문서 AWS-ConfigureAWSPackage 를 실행하는 데 사용됩니다. 이 권한은 아프리카(케이프타운) 리전(af-south-1)과 유럽(밀라노) 리전(eu-south-1)에만 필요합니다.

S3 버킷 ARN	설명
`arn:aws:s3:::aws-ssm-document-attachments-` *region*`/*`	AWS에서 소유하는 AWS Systems Manager의 기능인 Distributor에 대한 패키지가 포함된 S3 버킷에 대한 액세스를 제공합니다.
`arn:aws:s3:::patch-baseline-snapshot-` *region*`/*`	패치 기준 스냅샷이 포함된 S3 버킷에 대한 액세스를 제공합니다. 다음 SSM 문서를 사용하는 경우 필요합니다. • `AWS-RunPatchBaseline` • `AWS-RunPatchBaselineAssociation` • `AWS-RunPatchBaselineWithHooks` • `AWS-ApplyPatchBaseline` (레거시 SSM 문서) **Note** 중동(바레인) 리전(me-south-1)에서만 이 S3 버킷이 다른 명명 규칙을 사용합니다. 이 AWS 리전의 경우에만 다음 버킷을 대신 사용합니다. • `patch-baseline-snapshot-me-south-1-uduvl7q8` 아프리카(케이프타운) 리전(me-south-1)에서만 이 S3 버킷이 다른 명명 규칙을 사용합니다. 이 AWS 리전의 경우에만 다음 버킷을 대신 사용합니다. • `patch-baseline-snapshot-af-south-1-tbxdb5b9`

S3 버킷 ARN	설명
Linux 및 Windows Server 관리형 노드용: `arn:aws:s3:::aws-ssm-` *region*`/*` macOS에 대한 Amazon EC2 인스턴스의 경우: `arn:aws:s3:::aws-patchmanager-macos-` *region*`/*`	특정 Systems Manager 문서(SSM 문서)에 사용하는 데 필요한 모듈을 포함하여 S3 버킷에 대한 액세스를 제공합니다. 예: • `arn:aws:s3:::aws-ssm-us-east-2/*` • `aws-patchmanager-macos-us-east-2/*` 예외 몇 개의 AWS 리전에 있는 S3 버킷 이름은 ARN에 표시된 대로 확장된 명명 규칙을 사용합니다. 이러한 리전의 경우 다음 ARN을 대신 사용합니다. • 중동(바레인) 리전(me-south-1): `aws-patch-manager-me-south-1-a53fc9dce` • 아프리카(케이프타운) 리전(af-south-1): `aws-patch-manager-af-south-1-bdd5f65a9` • 유럽(밀라노) 리전(eu-south-1): `aws-patch-manager-eu-south-1-c52f3f594` • 아시아 태평양(오사카) 리전(ap-northeast-3): `aws-patch-manager-ap-northeast-3-67373598a` SSM 문서 다음은 이러한 버킷에 저장된 일반적으로 사용되는 몇 가지 SSM 문서입니다. `arn:aws:s3:::aws-ssm-` *region*`/`에서: • `AWS-RunPatchBaseline`

S3 버킷 ARN	설명
	• AWS-RunPatchBaselineAssociation • AWS-RunPatchBaselineWithHooks • AWS-InstanceRebootWithHooks • AWS-ConfigureWindowsUpdate • AWS-FindWindowsUpdates • AWS-PatchAsgInstance • AWS-PatchInstanceWithRollback • AWS-UpdateSSMAgent • AWS-UpdateEC2Config arn:aws:s3:::aws-patchmanager-macos- *region*/에서: • AWS-RunPatchBaseline • AWS-RunPatchBaselineAssociation • AWS-RunPatchBaselineWithHooks • AWS-InstanceRebootWithHooks • AWS-PatchAsgInstance • AWS-PatchInstanceWithRollback

예

다음 예에서는 미국 동부(오하이오) 리전(us-east-2)에서 Systems Manager 작업에 필요한 S3 버킷에 대한 액세스를 제공하는 방법을 보여줍니다. 대부분의 경우 VPC 엔드포인트를 사용할 때만 인스턴스 프로파일 또는 서비스 역할에서 이러한 권한을 명시적으로 제공해야 합니다.

> **Important**
>
> 이 정책에서는 특정 리전 대신에 와일드카드 문자(*)를 사용하지 않는 것이 좋습니다. 예를 들어 arn:aws:s3:::aws-ssm-*/*를 사용하지 말고 arn:aws:s3:::aws-ssm-us-

east-2/*를 사용합니다. 와일드카드를 사용하면 액세스 권한을 부여하도록 의도하지 않은 S3 버킷에 액세스할 수 있습니다. 둘 이상의 리전에 대해 인스턴스 프로파일을 사용하려는 경우, 각 리전에 대해 첫 번째 Statement 블록을 반복하는 것이 좋습니다.

```
{
        "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:::amazon-ssm-packages-us-east-2/*",
                    "arn:aws:s3:::us-east-2-birdwatcher-prod/*",
                    "arn:aws:s3:::aws-ssm-document-attachments-us-east-2/*",
                    "arn:aws:s3:::patch-baseline-snapshot-us-east-2/*",
                    "arn:aws:s3:::aws-ssm-us-east-2/*",
                    "arn:aws:s3:::aws-patchmanager-macos-us-east-2/*"
                ]
            }
        ]
    }
```

하드웨어 지문을 사용하여 하이브리드 정품 인증 시스템 검증

비 EC2 시스템이 하이브리드 및 멀티클라우드 환경에 있을 때 SSM Agent에서는 여러 시스템 속성(하드웨어 해시라고 함)을 수집하고 이러한 속성을 사용하여 지문을 계산합니다. 지문은 에이전트가 특정 Systems Manager API에 전달하는 불투명 문자열입니다. 이 고유 지문은 호출자를 특정 하이브리드 정품 인증 관리형 노드와 연결합니다. 에이전트는 로컬 디스크의 저장소라는 위치에 지문 및 하드웨어 해시를 저장합니다.

에이전트에서는 Systems Manager에서 사용하도록 시스템이 등록될 때 하드웨어 해시와 지문을 계산합니다. 그런 다음 에이전트가 RegisterManagedInstance 명령을 전송할 때 Systems Manager 서비스로 지문이 다시 전달됩니다.

나중에 RequestManagedInstanceRoleToken 명령을 전송할 때 에이전트는 현재 시스템 속성이 저장된 하드웨어 해시와 일치하는지 확인하기 위해 저장소의 지문 및 하드웨어 해시를 확인합니

다. 현재 시스템 속성이 저장소에 저장된 하드웨어 해시와 일치하는 경우 에이전트는 저장소에서 `RegisterManagedInstance`로 지문을 전달하여 호출에 성공합니다.

현재 시스템 속성이 저장된 하드웨어 해시와 일치하지 않으면 SSM Agent는 새 지문을 계산하고 저장소에 새 하드웨어 해시와 지문을 저장하고 새 지문을 `RequestManagedInstanceRoleToken`에 전달합니다. 이로 인해 *`RequestManagedInstanceRoleToken`*이 실패하고 에이전트는 Systems Manager 서비스에 연결하기 위한 역할 토큰을 얻을 수 없습니다.

이 실패는 의도적으로 설계된 것이며 여러 관리형 노드가 동일한 관리형 노드로 Systems Manager 서비스와 통신하지 못하도록 하기 위한 확인 단계로 사용됩니다.

현재 시스템 속성을 저장소에 저장된 하드웨어 해시와 비교할 때 에이전트는 다음 논리를 사용하여 이전 해시와 새 해시가 일치하는지 확인합니다.

- 시스템 ID(SID)가 다르면 일치하지 않습니다.
- 그렇지 않고 IP 주소가 같으면 일치합니다.
- 그렇지 않으면 일치하는 시스템 속성의 백분율이 계산되고 사용자가 구성한 유사성 임계값과 비교되어 일치 여부를 결정합니다.

유사성 임계값은 저장소에 하드웨어 해시의 일부로 저장됩니다.

유사성 임계값은 다음과 같은 명령을 사용하여 인스턴스를 등록한 후에 설정할 수 있습니다.

Linux 시스템의 경우:

```
sudo amazon-ssm-agent -fingerprint -similarityThreshold 1
```

PowerShell을 사용하는 Windows Server 시스템의 경우:

```
cd "C:\Program Files\Amazon\SSM\" `
    .\amazon-ssm-agent.exe -fingerprint -similarityThreshold 1
```

> **Important**
>
> 지문을 계산하는 데 사용되는 구성 요소 중 하나가 변경되면 에이전트가 최대 절전 모드로 전환될 수 있습니다. 이 최대 절전 모드를 방지하려면 유사성 임계값을 낮은 값(예: **1**)으로 설정합니다.

GitHub의 SSM Agent

SSM Agent의 소스 코드는 GitHub에 제공되므로 필요에 따라 에이전트를 조정할 수 있습니다. 포함하고 싶은 변경에 대해서는 풀 요청을 제출하는 것이 좋습니다. 단, Amazon Web Services에서 이 소프트웨어의 수정된 사본 실행을 지원하지 않습니다.

SSM Agent가 사전 설치된 상태로 AMIs 검색

AWS Systems Manager 에이전트(SSM Agent)는 AWS 및 신뢰할 수 있는 타사에서 제공하는 일부 Amazon Machine Images (AMIs)에 사전 설치되어 있습니다.

예를 들어 다음 운영 체제 중 하나를 사용하여 AMI에서 생성된 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스를 실행하면 SSM Agent가 이미 설치되어 있는 것을 알 수 있습니다.

- AlmaLinux
- 2017년 9월 이후의 Amazon Linux 1 Base AMI
- Amazon Linux 2
- Amazon Linux 2 ECS 최적화 기본 AMIs
- Amazon Linux 2023(AL2023)
- Amazon EKS 최적화 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)
- SUSE Linux Enterprise Server(SLES) 12 및 15
- Ubuntu Server 16.04, 18.04, 20.04 및 22.04
- Windows Server 2008-2012 R2 AMIs는 2016년 11월 이후에 게시되었습니다.
- Windows Server 2016, 2019, 2022

> **Note**
>
> SSM Agent가 이 목록에 없는 AWS 관리형 AMIs에 사전 설치되어 있을 수 있습니다. 이 경우 일반적으로 운영 체제(OS)에서 모든 Systems Manager 기능이 완벽하게 지원되지 않습니다. 또는 SSM Agent가 AWS Marketplace 또는 커뮤니티 AMIs 리포지토리의 AMIs에 사전 설치되었을 수 있습니다. 하지만 AWS는 이러한 AMIs를 지원하지 않습니다.

SSM Agent의 상태 확인

초기화된 시점에 따라 위 목록의 SSM Agent에서 생성한 인스턴스에 AMI가 사전 설치되지 않았을 수 있습니다. 인스턴스에 이 에이전트가 사전 설치되어 있지만 에이전트가 실행되고 있지 않을 수도 있습니다. 따라서 인스턴스에서 Systems Manager를 처음 사용하기 전에 SSM Agent의 상태를 확인하는 것이 좋습니다.

다음 절차를 사용하여 SSM Agent가 인스턴스에 설치되어 실행되고 있는지 확인합니다. 이 에이전트가 설치되어 있지 않은 경우 Linux, macOS 및 Windows Server 인스턴스에서 에이전트를 수동으로 설치할 수 있습니다.

인스턴스에서 SSM Agent 설치 확인

1. 새 인스턴스를 실행한 후 초기화되는 동안 몇 분 정도 기다리세요.
2. 선호하는 방법을 사용하여 인스턴스에 연결합니다. 예를 들어 SSH를 사용하여 Linux 인스턴스에 연결하거나 원격 데스크톱을 사용하여 Windows Server 인스턴스에 연결할 수 있습니다.
3. 인스턴스의 운영 체제 유형에 대한 명령을 실행하여 SSM Agent 상태를 확인합니다.

운영 체제	Command
Amazon Linux 1	`sudo status amazon-ssm-agent`
Amazon Linux 2 및 Amazon Linux 2023	`sudo systemctl status amazon-ssm-agent`
macOS	macOS에서 SSM Agent 상태를 확인하는 명령이 없습니다. 에이전트 로그 파일 `/var/log/amazon/ssm/amazon-ssm-agent.log` 를 찾아서 평가하여 상태를 확인할 수 있습니다.
SUSE Linux Enterprise Server	`sudo systemctl status amazon-ssm-agent`
Ubuntu Server (32비트)	`sudo status amazon-ssm-agent`
Ubuntu Server(64비트 - Deb)	`sudo systemctl status amazon-ssm-agent`

운영 체제	Command
Ubuntu Server(64비트-Snap)	`sudo systemctl status snap.amazon-ssm-agent.amazon-ssm-agent.service`
Windows Server	`Get-Service AmazonSSMAgent`

> **Tip**
>
> Systems Manager에서 지원하는 모든 운영 체제 유형의 SSM Agent 상태를 확인하기 위한 명령을 보려면 SSM Agent 상태 확인 및 에이전트 시작 단원을 참조하세요.

4. 명령 출력을 평가하여 SSM Agent의 상태를 확인하세요.

   상태: 설치되었고 실행 중임

   대부분의 경우 명령 출력에서 에이전트가 설치되어 있고 실행 중임을 나타냅니다.

   다음 예는 SSM Agent가 Amazon Linux 2 인스턴스에 설치되었고 실행 중임을 보여줍니다.

   ```
   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--
   ```

   다음 예는 SSM Agent가 Windows Server 인스턴스에 설치되었고 실행 중임을 보여줍니다.

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

   상태: 설치되었지만 실행되지 않음

   경우에 따라 명령 출력에서 에이전트가 설치되었지만 실행 중이지 않은 것으로 나타납니다.

다음 예는 SSM Agent가 Amazon Linux 2 인스턴스에 설치되었지만 실행되고 있지 않은 것을 보여줍니다.

```
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--
```

다음 예는 SSM Agent가 Windows Server 인스턴스에 설치되었지만 실행되고 있지 않은 것을 보여줍니다.

```
Status   Name               DisplayName
------   ----               -----------
Stopped  AmazonSSMAgent     Amazon SSM Agent
```

에이전트가 설치되어 있지만 실행 중이지 않은 경우 인스턴스의 운영 체제 유형에 대한 명령을 사용하여 수동으로 활성화할 수 있습니다.

운영 체제	Command
Amazon Linux 1	`sudo start amazon-ssm-agent`
Amazon Linux 2 및 Amazon Linux 2023	`sudo systemctl enable amazon-ssm-agent` `sudo systemctl start amazon-ssm-agent`
macOS	`sudo launchctl load -w /Library/LaunchDaemons/com.amazon.aws.ssm.plist` `sudo launchctl start com.amazon.aws.ssm`

운영 체제	Command
SUSE Linux Enterprise Server	`sudo systemctl enable amazon-ssm-agent` `sudo systemctl start amazon-ssm-agent`
Ubuntu Server (32비트)	`sudo start amazon-ssm-agent`
Ubuntu Server(64비트 - Deb)	`sudo systemctl enable amazon-ssm-agent` `sudo systemctl start amazon-ssm-agent`
Ubuntu Server(64비트 - Snap)	`sudo snap start amazon-ssm-agent`
Windows Server	PowerShell에서 다음 명령을 실행합니다. `Start-Service AmazonSSMAgent`

상태: 설치되지 않음

경우에 따라 명령 출력에서 에이전트가 설치되지 않은 것으로 나타납니다.

다음 예는 SSM Agent가 Amazon Linux 2 인스턴스에 설치되지 않은 것을 보여줍니다.

```
Unit amazon-ssm-agent.service could not be found.
```

다음 예는 SSM Agent가 Windows Server 인스턴스에 설치되지 않은 것을 보여줍니다.

```
Get-Service : Cannot find any service with service name 'AmazonSSMAgent'.
--truncated--
```

에이전트가 설치되지 않은 경우 다음 운영 체제 유형에 대한 절차를 사용하여 수동으로 설치할 수 있습니다.

- Linux용 EC2 인스턴스에 수동으로 SSM Agent 설치 및 제거

- SSM Agent용 EC2 인스턴스에 수동으로 macOS 설치 및 제거
- SSM Agent용 EC2 인스턴스에 수동으로 Windows Server 설치 및 제거

Linux용 EC2 인스턴스에서 SSM Agent 사용

AWS Systems Manager Agent(SSM Agent)는 Systems Manager 요청을 처리하고 요청에 지정된 대로 시스템을 구성합니다. 다음 주제의 절차를 사용하여 Linux 운영 체제에서 SSM Agent의 설치, 구성 또는 제거를 수행합니다.

주제

- SSM Agent의 서명 확인
- Linux용 EC2 인스턴스에 수동으로 SSM Agent 설치 및 제거
- SSM Agent를 구성하여 Linux 노드에 프록시 사용

SSM Agent의 서명 확인

Linux 인스턴스용 AWS Systems Manager Agent(SSM Agent) deb 및 rpm 설치 관리자 패키지는 암호화 서명됩니다. 퍼블릭 키를 사용하여 에이전트 패키지가 원본이며 수정되지 않았는지 확인할 수 있습니다. 파일에 손상이나 변경이 있을 경우 확인에 실패합니다. RPM 또는 GPG를 사용하여 설치 관리자 패키지의 서명을 확인할 수 있습니다. 다음 정보는 SSM Agent 버전 3.1.1141.0 이상용입니다.

> **Important**
>
> 이 항목의 뒷부분에 나와 있는 퍼블릭 키는 2025년 2월 17일에 만료됩니다. Systems Manager는 이전 공개 키가 만료되기 전에 이 주제에 새 공개 키를 게시합니다. 새 키를 사용할 수 있을 때 알림을 받으려면 이 주제에 대한 RSS 피드를 구독하는 것이 좋습니다.

인스턴스의 아키텍처 및 운영 체제에 대한 올바른 서명 파일을 찾으려면 다음 표를 참조하세요.

##은 미국 동부(오하이오) 리전의 `us-east-2` 같이 AWS Systems Manager가 지원하는 AWS 리전의 식별자를 나타냅니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

아키텍처	운영 체제	서명 파일 URL	에이전트 다운로드 파일 이름
x86_64	AlmaLinux, Amazon Linux 1, Amazon Linux 2, Amazon Linux 2023, CentOS, CentOS Stream, RHEL, Oracle Linux, Rocky Linux, 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`

아키텍처	운영 체제	서명 파일 URL	에이전트 다운로드 파일 이름
		debian_amd64/amazon-ssm-agent.deb.sig	
x86	Amazon Linux 1, Amazon Linux 2, Amazon Linux 2023, CentOS, RHEL	https://s3.*region*.amazonaws.com/amazon-ssm-*region*/latest/linux_386/amazon-ssm-agent.rpm.sig https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/linux_386/amazon-ssm-agent.rpm.sig	amazon-ssm-agent.rpm

아키텍처	운영 체제	서명 파일 URL	에이전트 다운로드 파일 이름
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

아키텍처	운영 체제	서명 파일 URL	에이전트 다운로드 파일 이름
ARM64	Amazon Linux 1, Amazon Linux 2, Amazon Linux 2023, CentOS, RHEL	`https://s3.region.amazonaws.com/amazon-ssm-region/latest/linux_arm64/amazon-ssm-agent.rpm.sig` `https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/linux_arm64/amazon-ssm-agent.rpm.sig`	`amazon-ssm-agent.rpm`

시작하기 전 준비 사항

SSM Agent의 서명을 확인하기 전에 운영 체제에 적합한 에이전트 패키지를 다운로드해야 합니다. 예를 들면 `https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/linux_arm64/amazon-ssm-agent.rpm`입니다. SSM Agent 패키지 다운로드에 대한 자세한 내용은 Linux용 EC2 인스턴스에 수동으로 SSM Agent 설치 및 제거 섹션을 참조하세요.

GPG

Linux 서버에서 SSM Agent 패키지를 확인하려면

1. 다음 퍼블릭를 복사해 `amazon-ssm-agent.gpg` 파일에 저장합니다.

```
-----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. 퍼블릭 키를 키링으로 가져오고 반환된 키 값을 기록해 둡니다.

```
gpg --import amazon-ssm-agent.gpg
```

3. 지문을 확인합니다. *# #*을 이전 단계의 값으로 바꿔야 합니다. RPM을 사용하여 설치 관리자 패키지를 확인하더라도 GPG를 사용하여 지문을 확인하는 것이 좋습니다.

```
gpg --fingerprint key-value
```

다음과 비슷한 출력이 반환됩니다.

```
pub   2048R/97DD04ED 2023-08-28 [expires: 2025-02-17]
      Key fingerprint = DE92 C7DA 3E56 E923 31D6  2A36 BC1F 495C 97DD 04ED
uid                  SSM Agent <ssm-agent-signer@amazon.com>
```

지문은 다음과 일치해야 합니다.

```
DE92 C7DA 3E56 E923 31D6 2A36 BC1F 495C 97DD 04ED
```

지문이 일치하지 않으면 에이전트를 설치하지 말고 AWS Support에 문의하십시오.

4. 아직 다운로드하지 않은 경우 인스턴스의 아키텍처 및 운영 체제에 따라 서명 파일을 다운로드합니다.
5. 설치 관리자 패키지 서명을 확인합니다. 이 주제의 앞부분에 나열된 대로 서명 파일과 에이전트를 다운로드할 때 *signature-filename* 및 *agent-download-filename*을 지정한 값으로 바꿔야 합니다.

```
gpg --verify signature-filename agent-download-filename
```

예를 들어, Amazon Linux 2의 x86_64 아키텍처의 경우:

```
gpg --verify amazon-ssm-agent.rpm.sig amazon-ssm-agent.rpm
```

다음과 비슷한 출력이 반환됩니다.

```
gpg: Signature made Thu 31 Aug 2023 07:46:49 PM UTC using RSA key ID 97DD04ED
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: DE92 C7DA 3E56 E923 31D6 2A36 BC1F 495C 97DD 04ED
```

출력에 `BAD signature` 문구가 포함된 경우, 절차를 올바르게 수행했는지 확인합니다. 이 응답이 계속되면 AWS Support에 연락하고 에이전트를 설치하지 않습니다. 신뢰에 대한 경고 메시지는 서명이 유효하지 않다는 의미가 아니라 퍼블릭 키를 확인하지 않았다는 의미입니다. 사용자 또는 사용자가 신뢰하는 사람이 서명한 키만 신뢰됩니다. 출력에 문구 `Can't check signature: No public key`가 포함된 경우 SSM Agent 버전 3.1.1141.0 이상을 다운로드했는지 확인합니다.

RPM

Linux 서버에서 SSM Agent 패키지를 확인하려면

1. 다음 퍼블릭를 복사해 `amazon-ssm-agent.gpg` 파일에 저장합니다.

```
-----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. 퍼블릭 키를 키링으로 가져오고 반환된 키 값을 기록해 둡니다.

```
rpm --import amazon-ssm-agent.gpg
```

3. 지문을 확인합니다. *# #*을 이전 단계의 값으로 바꿔야 합니다. RPM을 사용하여 설치 관리자 패키지를 확인하더라도 GPG를 사용하여 지문을 확인하는 것이 좋습니다.

```
gpg --fingerprint key-value
```

다음과 비슷한 출력이 반환됩니다.

```
pub   2048R/97DD04ED 2023-08-28 [expires: 2025-02-17]
      Key fingerprint = DE92 C7DA 3E56 E923 31D6  2A36 BC1F 495C 97DD 04ED
uid                  SSM Agent <ssm-agent-signer@amazon.com>
```

지문은 다음과 일치해야 합니다.

```
DE92 C7DA 3E56 E923 31D6 2A36 BC1F 495C 97DD 04ED
```

지문이 일치하지 않으면 에이전트를 설치하지 말고 AWS Support에 문의하십시오.

4. 설치 관리자 패키지 서명을 확인합니다. 이 주제의 앞부분에 나열된 대로 서명 파일과 에이전트를 다운로드할 때 *signature-filename* 및 *agent-download-filename*을 지정한 값으로 바꿔야 합니다.

```
rpm --checksig signature-filename agent-download-filename
```

예를 들어, Amazon Linux 2의 x86_64 아키텍처의 경우:

```
rpm --checksig amazon-ssm-agent.rpm.sig amazon-ssm-agent.rpm
```

다음과 비슷한 출력이 반환됩니다.

```
amazon-ssm-agent-3.1.1141.0-1.amzn2.x86_64.rpm: rsa sha1 (md5) pgp md5 OK
```

pgp가 출력에 없고 퍼블릭 키를 가져온 경우 에이전트가 서명되지 않은 것입니다. 출력에 `NOT OK (MISSING KEYS: (MD5) key-id)` 문구가 포함된 경우 절차를 올바르게 수행했는지 확인하고 SSM Agent 버전 3.1.1141.0 이상을 다운로드했는지 확인합니다. 이 응답이 계속되면 AWS Support에 연락하고 에이전트를 설치하지 않습니다.

Linux용 EC2 인스턴스에 수동으로 SSM Agent 설치 및 제거

Amazon Elastic Compute Cloud(Amazon EC2) Linux 운영 체제에 AWS Systems Manager Agent(SSM Agent)를 수동으로 설치하기 전에 다음 정보를 검토하세요.

SSM Agent 설치 파일 URL

모든 상용 AWS 리전에 저장된 SSM Agent 설치 파일에 액세스할 수 있습니다. 또한 파일의 대체 또는 백업 소스로 사용할 수 있는 전역 사용 가능한 Amazon Simple Storage Service(Amazon S3) 버킷에도 설치 파일이 제공됩니다.

1~2개의 인스턴스에서 에이전트를 수동으로 설치하는 경우, 빠른 설치 절차에 제공되는 명령을 사용하여 시간을 절약할 수 있습니다. 이러한 절차에서 제공되는 명령은 사용자 데이터를 통해 Amazon EC2 인스턴스에 스크립트로 전달될 수도 있습니다.

여러 인스턴스에 에이전트를 설치하는 데 사용할 스크립트나 템플릿을 생성하는 경우 지리적으로 위치한 AWS 리전 또는 근처에 있는 설치 파일을 사용하는 것이 좋습니다. 대량 설치의 경우 다운로드 속도가 빨라지고 지연 시간이 줄어들 수 있습니다. 이러한 경우 설치 주제의 사용자 지정 설치 명령 생성 절차를 사용하는 것이 좋습니다.

에이전트가 사전 설치된 Amazon Machine Images

SSM Agent는 AWS에서 제공하는 일부 Amazon Machine Images(AMIs)에 사전 설치됩니다. 자세한 설명은 SSM Agent가 사전 설치된 상태로 AMIs 검색을 참조하세요.

다른 머신 유형에 설치

에이전트를 Systems Manager에서 사용할 수 있도록 온프레미스 서버 또는 가상 머신(VM)에 설치해야 하는 경우 하이브리드 Linux 노드에 SSM Agent를 설치하는 방법을 참조하세요. 엣지 디바이스에서의 에이전트 설치에 대한 자세한 내용은 Systems Manager를 통한 엣지 디바이스 관리 단원을 참조하세요.

에이전트를 최신 상태로 유지

SSM Agent의 업데이트된 버전은 Systems Manager에 새 기능이 추가되거나 기존 기능에 업데이트가 발생할 때마다 릴리스됩니다. 최신 버전의 에이전트를 사용하지 못하면 관리형 노드에서 다양한 Systems Manager 기능을 사용하지 못할 수 있습니다. 이러한 이유로 시스템의 SSM Agent 상태를 최신 상태로 유지하는 프로세스를 자동화하는 것이 좋습니다. 자세한 설명은 SSM Agent 업데이트 자동화을 참조하세요. SSM Agent 업데이트에 대해 알림을 수신하려면 GitHub에서 SSM Agent 릴리스 정보 페이지를 구독합니다.

운영 체제 선택

지정된 운영 체제에 SSM Agent를 수동으로 설치하는 절차를 보려면 다음 목록에서 링크를 선택합니다.

> **Note**
>
> 다음 각 운영 체제의 지원되는 버전 목록은 Systems Manager가 지원되는 운영 체제 섹션을 참조하세요.

- AlmaLinux
- Amazon Linux 2 및 Amazon Linux 2023
- Amazon Linux 1 1
- CentOS
- CentOS Stream
- Debian Server
- Oracle Linux
- Red Hat Enterprise Linux
- Rocky Linux
- SUSE Linux Enterprise Server
- Ubuntu Server

Linux 인스턴스에서 SSM Agent 제거

Linux 인스턴스에서 SSM Agent를 제거하려면 운영 체제의 패키지 관리자를 사용합니다. 운영 체제에 따라 제거 명령은 다음 예제 명령과 비슷합니다.

```
sudo dpkg -r amazon-ssm-agent
```

AlmaLinux 인스턴스에 SSM Agent 수동 설치

이 섹션의 정보를 사용하면 AlmaLinux 인스턴스에 SSM Agent를 수동으로 설치 또는 재설치하는 데 도움이 됩니다.

시작하기 전 준비 사항

AlmaLinux 인스턴스에 SSM Agent를 설치하기 전에 다음을 참고합니다.

- AlmaLinux 인스턴스에 Python 3가 설치되어 있는지 확인합니다. SSM Agent가 제대로 작동하기 위해서는 이 작업을 순서대로 수행해야 합니다.

- 모든 Linux 기반 운영 체제에서의 SSM Agent 설치에 적용되는 중요한 정보는 Linux용 EC2 인스턴스에 수동으로 SSM Agent 설치 및 제거의 내용을 참조하세요.

주제

- AlmaLinux의 SSM Agent에 대한 빠른 설치 명령
- 리전에서 AlmaLinux용 사용자 지정 에이전트 설치 명령 생성

AlmaLinux의 SSM Agent에 대한 빠른 설치 명령

다음 단계를 사용하여 단일 인스턴스에서 SSM Agent를 수동으로 설치합니다. 이 절차에서는 전역에서 사용 가능한 설치 파일을 사용합니다.

시작하기 전 준비 사항

AlmaLinux 인스턴스에 SSM Agent를 설치하기 전에 다음을 참고합니다.

- AlmaLinux 인스턴스에 Python 3가 설치되어 있는지 확인합니다. SSM Agent가 제대로 작동하기 위해서는 이 작업을 순서대로 수행해야 합니다.

AlmaLinux에 SSM Agent를 설치하는 방법

1. 선호하는 방법(예: SSH)을 사용하여 AlmaLinux 인스턴스에 연결합니다.
2. 인스턴스 아키텍처에 대한 명령을 복사하고 인스턴스에서 실행합니다.

> **Note**
>
> 다음 명령의 URL에는 `ec2-downloads-windows` 디렉터리가 포함되어 있지만 AlmaLinux에 대한 올바른 전역 설치 파일입니다.

x86_64 인스턴스

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

ARM64 인스턴스

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. (권장) 다음 명령을 실행하여 에이전트가 실행 중인지 확인합니다.

```
sudo systemctl status amazon-ssm-agent
```

대부분의 경우 다음 예시와 같이 명령에서 에이전트가 실행 중임을 보고합니다.

```
# 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--
```

드물게 다음 예시와 같이 명령에서 에이전트가 설치되었지만 실행 중이지 않음을 보고합니다.

```
# 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--
```

이러한 경우 에이전트를 활성화하려면 다음 명령을 실행합니다.

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

리전에서 AlmaLinux용 사용자 지정 에이전트 설치 명령 생성

스크립트 또는 템플릿을 사용하여 여러 인스턴스에서 SSM Agent를 설치할 때 사용 중인 AWS 리전에 저장된 설치 파일을 사용하는 것이 좋습니다.

다음 명령의 경우 미국 동부(오하이오) 리전(us-east-2)에서 공개적으로 액세스할 수 있는 S3 버킷을 사용하는 예제를 제공합니다.

> **Tip**
>
> 또한 앞선 주제의 AlmaLinux의 SSM Agent에 대한 빠른 설치 명령 절차에 있는 전역 URL을 구성한 사용자 지정 리전 URL로 바꿀 수 있습니다.

다음 명령에서 *region*을 자신의 정보로 바꿉니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

x86_64

```
sudo dnf install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

다음 예를 참조하세요.

```
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
```

다음 예를 참조하세요.

```
sudo dnf install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_arm64/amazon-ssm-agent.rpm
```

Amazon Linux 2 및 Amazon Linux 2023 인스턴스에 SSM Agent 수동 설치

> **Important**
>
> 이 주제에서는 Amazon Linux 2 및 Amazon Linux 2023 인스턴스에서 SSM Agent로 작업하는 명령을 제공합니다. 이러한 명령 중 일부는 Amazon Linux 1 인스턴스에서 지원되지 않습니다. 계속하기 전에 인스턴스 유형에 대한 올바른 주제를 보고 있는지 확인하세요. Amazon Linux 1 인스턴스에서 실행할 명령은 Amazon Linux 1 인스턴스에 SSM Agent 수동 설치 섹션을 참조하세요.

대부분의 경우 AWS에서 제공하는 Amazon Linux 2 및 Amazon Linux 2023용 Amazon Machine Images(AMIs)에는 AWS Systems Manager 에이전트(SSM Agent)가 기본적으로 사전 설치되어 있습니다. 자세한 내용은 SSM Agent가 사전 설치된 상태로 AMIs 검색 단원을 참조하십시오.

SSM Agent가 새 Amazon Linux 2 또는 Amazon Linux 2023 인스턴스에 사전 설치되어 있지 않거나 에이전트를 수동으로 다시 설치해야 하는 경우 이 페이지의 정보를 사용하면 도움이 됩니다.

시작하기 전 준비 사항

Amazon Linux 2 또는 Amazon Linux 2023 인스턴스에 SSM Agent를 설치하기 전에 다음을 참고합니다.

- 모든 Linux 기반 운영 체제에서의 SSM Agent 설치에 적용되는 중요한 정보는 Linux용 EC2 인스턴스에 수동으로 SSM Agent 설치 및 제거의 내용을 참조하세요.
- SSM 문서 `AWS-UpdateSSMAgent`를 사용하여 에이전트를 설치하거나 업데이트한 후 관리형 노드에서 `yum` 명령을 사용하여 SSM Agent를 업데이트하는 경우, 다음과 같은 메시지가 표시될 수 있습니다. "경고: RPMDB가 yum 외부에서 변경되었습니다." 이 메시지는 예상되는 결과이며 무시해도 됩니다.

주제

- Amazon Linux 2 또는 Amazon Linux 2023의 SSM Agent에 대한 빠른 설치 명령
- 리전에서 Amazon Linux 2 또는 Amazon Linux 2023용 사용자 지정 에이전트 설치 명령 생성

Amazon Linux 2 또는 Amazon Linux 2023의 SSM Agent에 대한 빠른 설치 명령

다음 단계를 사용하여 단일 인스턴스에서 SSM Agent를 수동으로 설치합니다. 이 절차에서는 전역에서 사용 가능한 설치 파일을 사용합니다.

빠른 복사 및 붙여넣기 명령을 사용하여 Amazon Linux 2 또는 Amazon Linux 2023에서 SSM Agent를 설치하는 방법

1. 선호하는 방법(예: SSH)을 사용하여 Amazon Linux 2 또는 Amazon Linux 2023 인스턴스에 연결합니다.
2. 인스턴스 아키텍처에 대한 명령을 복사하고 인스턴스에서 실행합니다.

   > **Note**
   >
   > 다음 명령의 URL에는 `ec2-downloads-windows` 디렉터리가 포함되어 있지만 Amazon Linux 2 및 Amazon Linux 2023에 대한 올바른 전역 설치 파일입니다.

   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. (권장) 다음 명령을 실행하여 에이전트가 실행 중인지 확인합니다.

   ```
   sudo systemctl status amazon-ssm-agent
   ```

   대부분의 경우 다음 예시와 같이 명령에서 에이전트가 실행 중임을 보고합니다.

   ```
   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--
   ```

   드물게 다음 예시와 같이 명령에서 에이전트가 설치되었지만 실행 중이지 않음을 보고합니다.

   ```
   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--
```

이러한 경우 에이전트를 활성화하려면 다음 명령을 실행합니다.

```
sudo systemctl start amazon-ssm-agent
```

리전에서 Amazon Linux 2 또는 Amazon Linux 2023용 사용자 지정 에이전트 설치 명령 생성

스크립트 또는 템플릿을 사용하여 여러 인스턴스에서 SSM Agent를 설치할 때 사용 중인 AWS 리전에 저장된 설치 파일을 사용하는 것이 좋습니다.

다음 명령의 경우 미국 동부(오하이오) 리전(us-east-2)에서 공개적으로 액세스할 수 있는 S3 버킷을 사용하는 예제를 제공합니다.

> **Tip**
>
> 또한 앞선 주제의 Amazon Linux 1의 SSM Agent에 대한 빠른 설치 명령 절차에 있는 전역 URL을 구성한 사용자 지정 리전 URL로 바꿀 수 있습니다.

다음 명령에서 *region*을 자신의 정보로 바꿉니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

x86_64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

다음 예를 참조하세요.

```
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
```

다음 예를 참조하세요.

```
sudo yum install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_arm64/amazon-ssm-agent.rpm
```

Amazon Linux 1 인스턴스에 SSM Agent 수동 설치

> **Important**
>
> Amazon Linux 1은 AWS 뉴스 블로그의 Amazon Linux AMI 수명 종료 업데이트에서 발표한 대로 2020년 12월 31일 표준 지원이 종료되고 2023년 12월 31일 수명 주기가 종료되었습니다. AWS는 더 이상 이 운영 체제에 대한 Amazon Machine Images(AMIs)를 제공하지 않습니다. 하지만 AWS Systems Manager은 기존 Amazon Linux 1 인스턴스에 대한 지원을 계속 제공합니다.
> 이 주제에서는 Amazon Linux 1 인스턴스에서 SSM Agent로 작업하는 명령을 제공합니다. 이러한 명령 중 일부는 Amazon Linux 2 및 Amazon Linux 2023 인스턴스에서 지원되지 않습니다. 계속하기 전에 인스턴스 유형에 대한 올바른 주제를 보고 있는지 확인하세요. Amazon Linux 2 또는 Amazon Linux 2023 인스턴스에서 실행되는 명령은 Amazon Linux 2 및 Amazon Linux 2023 인스턴스에 SSM Agent 수동 설치 섹션을 참조하세요.

대부분의 경우 AWS에서 제공하는 Amazon Linux 1용 Amazon Machine Images(AMIs)에는 AWS Systems Manager 에이전트(SSM Agent)가 기본적으로 사전 설치되어 있습니다. 자세한 내용은 SSM Agent가 사전 설치된 상태로 AMIs 검색 단원을 참조하십시오.

Amazon Linux 1에 에이전트를 수동으로 다시 설치해야 하는 경우 이 페이지의 정보를 사용하면 도움이 됩니다.

시작하기 전 준비 사항

Amazon Linux 1 인스턴스에 SSM Agent를 설치하기 전에 다음 사항에 유의하세요.

- 모든 Linux 기반 운영 체제에서의 SSM Agent 설치에 적용되는 중요한 정보는 Linux용 EC2 인스턴스에 수동으로 SSM Agent 설치 및 제거의 내용을 참조하세요.

- SSM 문서 `AWS-UpdateSSMAgent`를 사용하여 에이전트를 설치하거나 업데이트한 후 관리형 노드에서 `yum` 명령을 사용하여 SSM Agent를 업데이트하는 경우, 다음과 같은 메시지가 표시될 수 있습니다. "경고: RPMDB가 yum 외부에서 변경되었습니다." 이 메시지는 예상되는 결과이며 무시해도 됩니다.

주제

Amazon Linux 1의 SSM Agent에 대한 빠른 설치 명령

다음 단계를 사용하여 단일 인스턴스에서 SSM Agent를 수동으로 설치합니다. 이 절차에서는 전역에서 사용 가능한 설치 파일을 사용합니다.

빠른 복사 및 붙여넣기 명령을 사용하여 Amazon Linux 1에서 SSM Agent 설치

1. 선호하는 방법(예: SSH)을 사용하여 Amazon Linux 1 인스턴스에 연결합니다.
2. 인스턴스 아키텍처에 대한 명령을 복사하고 인스턴스에서 실행합니다.

> **Note**
>
> 다음 명령의 URL에는 `ec2-downloads-windows` 디렉터리가 포함되어 있지만 Amazon Linux 1에 대한 올바른 전역 설치 파일입니다.

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. (권장) 인스턴스 아키텍처에 대한 명령을 실행하여 에이전트가 실행 중인지 확인합니다.

   x86_64 및 x86

   ```
   sudo status amazon-ssm-agent
   ```

   ARM64

   ```
   sudo systemctl status amazon-ssm-agent
   ```

   대부분의 경우 다음 예시와 같이 명령에서 에이전트가 실행 중임을 보고합니다.

   x86_64 및 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--
   ```

   드물게 다음 예시와 같이 명령에서 에이전트가 설치되었지만 실행 중이지 않음을 보고합니다.

   x86_64 및 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--
```

이러한 경우 에이전트를 활성화하려면 인스턴스 아키텍처에 대해 다음 명령을 실행합니다.

x86_64 및 x86

```
sudo start amazon-ssm-agent
```

ARM64

```
sudo systemctl start amazon-ssm-agent
```

리전에서 Amazon Linux 1용 사용자 지정 에이전트 설치 명령 생성

스크립트 또는 템플릿을 사용하여 여러 인스턴스에서 SSM Agent를 설치할 때 사용 중인 AWS 리전에 저장된 설치 파일을 사용하는 것이 좋습니다.

다음 명령의 경우 미국 동부(오하이오) 리전(us-east-2)에서 공개적으로 액세스할 수 있는 S3 버킷을 사용하는 예제를 제공합니다.

> **Tip**
>
> 또한 앞선 주제의 Amazon Linux 1의 SSM Agent에 대한 빠른 설치 명령 절차에 있는 전역 URL을 구성한 사용자 지정 리전 URL로 바꿀 수 있습니다.

다음 명령에서 *region*을 자신의 정보로 바꿉니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

x86_64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

다음 예를 참조하세요.

```
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
```

다음 예를 참조하세요.

```
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
```

다음 예를 참조하세요.

```
sudo yum install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_arm64/amazon-ssm-agent.rpm
```

CentOS 인스턴스에 SSM Agent 수동 설치

AWS에서 제공하는 CentOS용 Amazon Machine Images(AMIs)에는 AWS Systems Manager 에이전트(SSM Agent)가 기본적으로 사전 설치되어 있습니다. 에이전트가 사전 설치될 수 있는 AWS에서 관리하는 AMIs 목록은 SSM Agent가 사전 설치된 상태로 AMIs 검색의 내용을 참조하세요.

이 섹션의 정보를 사용하면 CentOS 인스턴스에서 SSM Agent를 수동으로 설치 또는 재설치하는 데 도움이 됩니다.

시작하기 전 준비 사항

CentOS 인스턴스에 SSM Agent를 설치하기 전에 다음 사항에 유의하세요.

- 모든 Linux 기반 운영 체제에서의 SSM Agent 설치에 적용되는 중요한 정보는 Linux용 EC2 인스턴스에 수동으로 SSM Agent 설치 및 제거의 내용을 참조하세요.

- SSM 문서 `AWS-UpdateSSMAgent`를 사용하여 에이전트를 설치하거나 업데이트한 후 관리형 노드에서 `yum` 명령을 사용하여 SSM Agent를 업데이트하는 경우, 다음과 같은 메시지가 표시될 수 있습니다. "경고: RPMDB가 yum 외부에서 변경되었습니다." 이 메시지는 예상되는 결과이며 무시해도 됩니다.

주제

- CentOS 8.x에 SSM Agent 설치
- CentOS 7.x에 SSM Agent 설치
- CentOS 6.x에 SSM Agent 설치

CentOS 8.x에 SSM Agent 설치

Amazon Machine Images에서 제공하는 CentOS 8용 AMIs(AWS)에는 AWS Systems Manager 에이전트(SSM Agent)가 기본적으로 사전 설치되어 있습니다. 이 페이지의 정보를 사용하면 CentOS 8 인스턴스에서 에이전트를 수동으로 설치 또는 재설치하는 데 도움이 됩니다.

시작하기 전 준비 사항

CentOS 8 인스턴스에 SSM Agent를 설치하기 전에 다음 사항에 유의하세요.

- CentOS 8 인스턴스에 Python 2 또는 Python 3가 설치되어 있는지 확인합니다. SSM Agent가 제대로 작동하기 위해서는 이 작업을 순서대로 수행해야 합니다.

주제

- CentOS 8의 SSM Agent에 대한 빠른 설치 명령
- 리전에서 CentOS 8용 사용자 지정 에이전트 설치 명령 생성

CentOS 8의 SSM Agent에 대한 빠른 설치 명령

다음 단계를 사용하여 단일 인스턴스에서 SSM Agent를 수동으로 설치합니다. 이 절차에서는 전역에서 사용 가능한 설치 파일을 사용합니다.

CentOS 8.x에 SSM Agent를 설치하려면

1. 선호하는 방법(예: SSH)을 사용하여 CentOS 8 인스턴스에 연결합니다.
2. 인스턴스 아키텍처에 대한 명령을 복사하고 인스턴스에서 실행합니다.

> **Note**
>
> 다음 명령의 URL에는 `ec2-downloads-windows` 디렉터리가 포함되어 있지만 CentOS 8에 대한 올바른 전역 설치 파일입니다.

x86_64 인스턴스

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

ARM64 인스턴스

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. (권장) 다음 명령을 실행하여 에이전트가 실행 중인지 확인합니다.

```
sudo systemctl status amazon-ssm-agent
```

대부분의 경우 다음 예시와 같이 명령에서 에이전트가 실행 중임을 보고합니다.

```
# 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--
```

드물게 다음 예시와 같이 명령에서 에이전트가 설치되었지만 실행 중이지 않음을 보고합니다.

```
# amazon-ssm-agent.service - amazon-ssm-agent
   Loaded: loaded (/etc/systemd/system/amazon-ssm-agent.service; disabled; vend>
   Active: inactive (dead)
           --truncated--
```

이러한 경우 에이전트를 활성화하려면 다음 명령을 실행합니다.

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

리전에서 CentOS 8용 사용자 지정 에이전트 설치 명령 생성

스크립트 또는 템플릿을 사용하여 여러 인스턴스에서 SSM Agent를 설치할 때 사용 중인 AWS 리전에 저장된 설치 파일을 사용하는 것이 좋습니다.

다음 명령의 경우 미국 동부(오하이오) 리전(us-east-2)에서 공개적으로 액세스할 수 있는 S3 버킷을 사용하는 예제를 제공합니다.

> **Tip**
>
> 또한 앞선 주제의 CentOS 8의 SSM Agent에 대한 빠른 설치 명령 절차에 있는 전역 URL을 구성한 사용자 지정 리전 URL로 바꿀 수 있습니다.

다음 명령에서 *region*을 자신의 정보로 바꿉니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

x86_64

```
sudo dnf install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

다음 예를 참조하세요.

```
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
```

다음 예를 참조하세요.

```
sudo dnf install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_arm64/amazon-ssm-agent.rpm
```

CentOS 7.x에 SSM Agent 설치

AWS에서 제공하는 CentOS용 Amazon Machine Images(AMIs)에는 AWS Systems Manager Agent(SSM Agent)가 기본적으로 사전 설치되어 있지 않습니다. 에이전트가 사전 설치된 AWS 관리형 AMIs 목록은 SSM Agent가 사전 설치된 Amazon Machine Images(AMIs)를 확인하세요.

주제

- CentOS 7의 SSM Agent에 대한 빠른 설치 명령
- 리전에서 CentOS 7용 사용자 지정 에이전트 설치 명령 생성

CentOS 7의 SSM Agent에 대한 빠른 설치 명령

다음 단계를 사용하여 단일 인스턴스에서 SSM Agent를 수동으로 설치합니다. 이 절차에서는 전역에서 사용 가능한 설치 파일을 사용합니다.

CentOS 7.x에 SSM Agent를 설치하려면

1. 선호하는 방법(예: SSH)을 사용하여 CentOS 7 인스턴스에 연결합니다.
2. 인스턴스 아키텍처에 대한 명령을 복사하고 인스턴스에서 실행합니다.

> **Note**
>
> 다음 명령의 URL에는 `ec2-downloads-windows` 디렉터리가 포함되어 있지만 CentOS 7에 대한 올바른 전역 설치 파일입니다.

x86_64 인스턴스

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

ARM64 인스턴스

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. (권장) 다음 명령을 실행하여 에이전트가 실행 중인지 확인합니다.

```
sudo systemctl status amazon-ssm-agent
```

대부분의 경우 다음 예시와 같이 명령에서 에이전트가 실행 중임을 보고합니다.

```
# 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--
```

드물게 다음 예시와 같이 명령에서 에이전트가 설치되었지만 실행 중이지 않음을 보고합니다.

```
# 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--
```

이러한 경우 에이전트를 활성화하려면 다음 명령을 실행합니다.

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

리전에서 CentOS 7용 사용자 지정 에이전트 설치 명령 생성

스크립트 또는 템플릿을 사용하여 여러 인스턴스에서 SSM Agent를 설치할 때 사용 중인 AWS 리전에 저장된 설치 파일을 사용하는 것이 좋습니다.

다음 명령의 경우 미국 동부(오하이오) 리전(us-east-2)에서 공개적으로 액세스할 수 있는 S3 버킷을 사용하는 예제를 제공합니다.

> **Tip**
>
> 또한 앞선 주제의 CentOS 7의 SSM Agent에 대한 빠른 설치 명령 절차에 있는 전역 URL을 구성한 사용자 지정 리전 URL로 바꿀 수 있습니다.

다음 명령에서 *region*을 자신의 정보로 바꿉니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

x86_64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

다음 예를 참조하세요.

```
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
```

다음 예를 참조하세요.

```
sudo yum install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_arm64/amazon-ssm-agent.rpm
```

CentOS 6.x에 SSM Agent 설치

Amazon Machine Images에서 제공하는 CentOS 6용 AMIs(AWS)에는 AWS Systems Manager 에이전트(SSM Agent)가 기본적으로 사전 설치되어 있습니다. 이 페이지의 정보를 사용하면 CentOS 6 인스턴스에서 에이전트를 수동으로 설치 또는 재설치하는 데 도움이 됩니다.

주제

- CentOS 6의 SSM Agent에 대한 빠른 설치 명령
- 리전에서 CentOS 6용 사용자 지정 에이전트 설치 명령 생성

CentOS 6의 SSM Agent에 대한 빠른 설치 명령

다음 단계를 사용하여 단일 인스턴스에서 SSM Agent를 수동으로 설치합니다. 이 절차에서는 전역에서 사용 가능한 설치 파일을 사용합니다.

CentOS 6.x에 SSM Agent를 설치하려면

1. 선호하는 방법(예: SSH)을 사용하여 CentOS 6 인스턴스에 연결합니다.
2. 인스턴스 아키텍처에 대한 명령을 복사하고 인스턴스에서 실행합니다.

> **Note**
>
> 다음 명령의 URL에는 `ec2-downloads-windows` 디렉터리가 포함되어 있지만 CentOS 6에 대한 올바른 전역 설치 파일입니다.
> 다음 명령은 `latest` 디렉터리 대신 버전 디렉터리 `3.0.1479.0`을 지정합니다. 이는 SSM Agent 버전 3.1 이상이 CentOS 6에서 지원되지 않기 때문입니다.

x86_64 인스턴스

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/
SSMAgent/3.0.1479.0/linux_amd64/amazon-ssm-agent.rpm
```

x86 인스턴스

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/
SSMAgent/3.0.1479.0/linux_386/amazon-ssm-agent.rpm
```

3. (권장) 다음 명령을 실행하여 에이전트가 실행 중인지 확인합니다.

```
sudo status amazon-ssm-agent
```

대부분의 경우 다음 예시와 같이 명령에서 에이전트가 실행 중임을 보고합니다.

```
amazon-ssm-agent start/running, process 1744
```

드물게 다음 예시와 같이 명령에서 에이전트가 설치되었지만 실행 중이지 않음을 보고합니다.

```
amazon-ssm-agent stop/waiting
```

이러한 경우 에이전트를 활성화하려면 다음 명령을 실행합니다.

```
sudo start amazon-ssm-agent
```

리전에서 CentOS 6용 사용자 지정 에이전트 설치 명령 생성

스크립트 또는 템플릿을 사용하여 여러 인스턴스에서 SSM Agent를 설치할 때 사용 중인 AWS 리전에 저장된 설치 파일을 사용하는 것이 좋습니다.

다음 명령의 경우 미국 동부(오하이오) 리전(us-east-2)에서 공개적으로 액세스할 수 있는 S3 버킷을 사용하는 예제를 제공합니다.

> **Tip**
>
> 또한 앞선 주제의 CentOS 6의 SSM Agent에 대한 빠른 설치 명령 절차에 있는 전역 URL을 구성한 사용자 지정 리전 URL로 바꿀 수 있습니다.

다음 명령에서 *region*을 자신의 정보로 바꿉니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

> **Note**
>
> 다음 명령은 latest 디렉터리 대신 버전 디렉터리 3.0.1390.0을 지정합니다. 이는 SSM Agent 버전 3.1 이상이 CentOS 6에서 지원되지 않기 때문입니다.

x86_64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/3.0.1479.0/
linux_amd64/amazon-ssm-agent.rpm
```

다음 예를 참조하세요.

```
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
```

다음 예를 참조하세요.

```
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
```

CentOS Stream 인스턴스에 SSM Agent 수동 설치

AWS에서 제공하는 CentOS Stream용 Amazon Machine Images(AMIs)에는 AWS Systems Manager 에이전트(SSM Agent)가 기본적으로 사전 설치되어 있습니다. 에이전트가 사전 설치될 수 있는 AWS에서 관리하는 AMIs 목록은 SSM Agent가 사전 설치된 상태로 AMIs 검색의 내용을 참조하세요.

이 섹션의 정보를 사용하면 CentOS Stream 인스턴스에서 SSM Agent를 수동으로 설치 또는 재설치하는 데 도움이 됩니다.

시작하기 전 준비 사항

CentOS Stream 인스턴스에 SSM Agent를 설치하기 전에 다음 사항에 유의하세요.

- 모든 Linux 기반 운영 체제에서의 SSM Agent 설치에 적용되는 중요한 정보는 Linux용 EC2 인스턴스에 수동으로 SSM Agent 설치 및 제거의 내용을 참조하세요.

주제

- CentOS Stream의 SSM Agent에 대한 빠른 설치 명령
- 리전에서 CentOS Stream용 사용자 지정 에이전트 설치 명령 생성

CentOS Stream의 SSM Agent에 대한 빠른 설치 명령

다음 단계를 사용하여 단일 인스턴스에서 SSM Agent를 수동으로 설치합니다. 이 절차에서는 전역에서 사용 가능한 설치 파일을 사용합니다.

시작하기 전 준비 사항

CentOS Stream 인스턴스에 SSM Agent를 설치하기 전에 다음 사항에 유의하세요.

- CentOS Stream 8 인스턴스에 Python 2 또는 Python 3가 설치되어 있는지 확인합니다. SSM Agent가 제대로 작동하기 위해서는 이 작업을 순서대로 수행해야 합니다.

CentOS Stream에 SSM Agent를 설치하려면

1. 선호하는 방법(예: SSH)을 사용하여 CentOS Stream 인스턴스에 연결합니다.

2. 인스턴스 아키텍처에 대한 명령을 복사하고 인스턴스에서 실행합니다.

> **Note**
>
> 다음 명령의 URL에는 `ec2-downloads-windows` 디렉터리가 포함되어 있지만 CentOS Stream에 대한 올바른 전역 설치 파일입니다.

x86_64 인스턴스

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

ARM64 인스턴스

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. (권장) 다음 명령을 실행하여 에이전트가 실행 중인지 확인합니다.

```
sudo systemctl status amazon-ssm-agent
```

대부분의 경우 다음 예시와 같이 명령에서 에이전트가 실행 중임을 보고합니다.

```
# 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--
```

드물게 다음 예시와 같이 명령에서 에이전트가 설치되었지만 실행 중이지 않음을 보고합니다.

```
# 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--
```

이러한 경우 에이전트를 활성화하려면 다음 명령을 실행합니다.

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

리전에서 CentOS Stream용 사용자 지정 에이전트 설치 명령 생성

스크립트 또는 템플릿을 사용하여 여러 인스턴스에서 SSM Agent를 설치할 때 사용 중인 AWS 리전에 저장된 설치 파일을 사용하는 것이 좋습니다.

다음 명령의 경우 미국 동부(오하이오) 리전(us-east-2)에서 공개적으로 액세스할 수 있는 S3 버킷을 사용하는 예제를 제공합니다.

> **Tip**
>
> 또한 앞선 주제의 CentOS Stream의 SSM Agent에 대한 빠른 설치 명령 절차에 있는 전역 URL을 구성한 사용자 지정 리전 URL로 바꿀 수 있습니다.

다음 명령에서 *region*을 자신의 정보로 바꿉니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

x86_64

```
sudo dnf install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

다음 예를 참조하세요.

```
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
```

다음 예를 참조하세요.

```
sudo dnf install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_arm64/amazon-ssm-agent.rpm
```

Debian Server 인스턴스에 SSM Agent 수동 설치

AWS에서 제공하는 Debian Server용 Amazon Machine Images(AMIs)에는 AWS Systems Manager 에이전트(SSM Agent)가 기본적으로 사전 설치되어 있습니다. 에이전트가 사전 설치될 수 있는 AWS에서 관리하는 AMIs 목록은 SSM Agent가 사전 설치된 상태로 AMIs 검색의 내용을 참조하세요.

이 섹션의 정보를 사용하면 Debian Server 인스턴스에서 SSM Agent를 수동으로 설치 또는 재설치하는 데 도움이 됩니다.

시작하기 전 준비 사항

Debian Server 인스턴스에 SSM Agent를 설치하기 전에 다음 사항에 유의하세요.

- 모든 Linux 기반 운영 체제에서의 SSM Agent 설치에 적용되는 중요한 정보는 Linux용 EC2 인스턴스에 수동으로 SSM Agent 설치 및 제거의 내용을 참조하세요.

주제

- Debian Server의 SSM Agent에 대한 빠른 설치 명령
- 리전에서 Debian Server용 사용자 지정 에이전트 설치 명령 생성

Debian Server의 SSM Agent에 대한 빠른 설치 명령

다음 단계를 사용하여 단일 인스턴스에서 SSM Agent를 수동으로 설치합니다. 이 절차에서는 전역에서 사용 가능한 설치 파일을 사용합니다.

Debian Server에 SSM Agent를 설치하려면

1. 선호하는 방법(예: SSH)을 사용하여 Debian Server 인스턴스에 연결합니다.
2. 다음 명령을 실행하여 인스턴스에서 임시 디렉터리를 생성합니다.

```
mkdir /tmp/ssm
```

3. 다음 명령을 실행하여 임시 디렉터리로 변경합니다.

```
cd /tmp/ssm
```

4. 인스턴스 아키텍처에 대한 명령을 복사하고 인스턴스에서 실행합니다.

> **Note**
>
> 다음 명령의 URL에는 `ec2-downloads-windows` 디렉터리가 포함되어 있지만, 이는 Debian Server에 대한 올바른 전역 설치 파일입니다.
> Debian Server 8의 경우 x86_64 아키텍처만 지원됩니다.

x86_64 인스턴스

```
wget https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/
debian_amd64/amazon-ssm-agent.deb
```

ARM64 인스턴스

```
wget https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/
debian_arm64/amazon-ssm-agent.deb
```

5. 다음 명령을 실행합니다.

```
sudo dpkg -i amazon-ssm-agent.deb
```

6. (권장) 다음 명령을 실행하여 에이전트가 실행 중인지 확인합니다.

```
sudo systemctl status amazon-ssm-agent
```

대부분의 경우 다음 예시와 같이 명령에서 에이전트가 실행 중임을 보고합니다.

```
# 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--
```

드물게 다음 예시와 같이 명령에서 에이전트가 설치되었지만 실행 중이지 않음을 보고합니다.

```
# 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--
```

이러한 경우 에이전트를 활성화하려면 다음 명령을 실행합니다.

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

리전에서 Debian Server용 사용자 지정 에이전트 설치 명령 생성

스크립트 또는 템플릿을 사용하여 여러 인스턴스에서 SSM Agent를 설치할 때 사용 중인 AWS 리전에 저장된 설치 파일을 사용하는 것이 좋습니다.

다음 명령의 경우 미국 동부(오하이오) 리전(us-east-2)에서 공개적으로 액세스할 수 있는 S3 버킷을 사용하는 예제를 제공합니다.

> **Tip**
>
> 또한 앞선 주제의 Debian Server의 SSM Agent에 대한 빠른 설치 명령 절차에 있는 전역 URL을 구성한 사용자 지정 리전 URL로 바꿀 수 있습니다.

다음 명령에서 *region*을 자신의 정보로 바꿉니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

> **Note**
>
> Debian Server 8의 경우 x86_64 아키텍처만 지원됩니다.

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
```

다음 예를 참조하세요.

```
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
```

다음 예를 참조하세요.

```
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
```

Oracle Linux 인스턴스에 SSM Agent 수동 설치

AWS에서 제공하는 Oracle Linux용 Amazon Machine Images(AMIs)에는 AWS Systems Manager 에이전트(SSM Agent)가 기본적으로 사전 설치되어 있습니다. 에이전트가 사전 설치될 수 있는 AWS에서 관리하는 AMIs 목록은 SSM Agent가 사전 설치된 상태로 AMIs 검색의 내용을 참조하세요.

이 섹션의 정보를 사용하면 Oracle Linux 인스턴스에서 SSM Agent를 수동으로 설치 또는 재설치하는 데 도움이 됩니다.

시작하기 전 준비 사항

Oracle Linux 인스턴스에 SSM Agent를 설치하기 전에 다음 사항에 유의하세요.

- 모든 Linux 기반 운영 체제에서의 SSM Agent 설치에 적용되는 중요한 정보는 Linux용 EC2 인스턴스에 수동으로 SSM Agent 설치 및 제거의 내용을 참조하세요.
- SSM 문서 `AWS-UpdateSSMAgent`를 사용하여 에이전트를 설치하거나 업데이트한 후 관리형 노드에서 `yum` 명령을 사용하여 SSM Agent를 업데이트하는 경우, 다음과 같은 메시지가 표시될 수 있습니다. "경고: RPMDB가 yum 외부에서 변경되었습니다." 이 메시지는 예상되는 결과이며 무시해도 됩니다.

주제

- Oracle Linux의 SSM Agent에 대한 빠른 설치 명령
- 리전에서 Oracle Linux용 사용자 지정 에이전트 설치 명령 생성

Oracle Linux의 SSM Agent에 대한 빠른 설치 명령

다음 단계를 사용하여 단일 인스턴스에서 SSM Agent를 수동으로 설치합니다. 이 절차에서는 전역에서 사용 가능한 설치 파일을 사용합니다.

빠른 복사 및 붙여넣기 명령을 사용하여 Oracle Linux에서 SSM Agent 설치

1. 선호하는 방법(예: SSH)을 사용하여 Oracle Linux 인스턴스에 연결합니다.
2. 다음 명령을 복사하고 인스턴스에서 실행합니다.

> **Note**
>
> 다음 명령의 URL에는 `ec2-downloads-windows` 디렉터리가 포함되어 있지만 Oracle Linux에 대한 올바른 전역 설치 파일입니다.

x86_64

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

3. (권장) 다음 명령을 실행하여 에이전트가 실행 중인지 확인합니다.

```
sudo systemctl status amazon-ssm-agent
```

대부분의 경우 다음 예시와 같이 명령에서 에이전트가 실행 중임을 보고합니다.

```
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--
```

드물게 다음 예시와 같이 명령에서 에이전트가 설치되었지만 실행 중이지 않음을 보고합니다.

```
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--
```

이러한 경우 에이전트를 활성화하려면 다음 명령을 실행합니다.

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

리전에서 Oracle Linux용 사용자 지정 에이전트 설치 명령 생성

스크립트 또는 템플릿을 사용하여 여러 인스턴스에서 SSM Agent를 설치할 때 사용 중인 AWS 리전에 저장된 설치 파일을 사용하는 것이 좋습니다.

다음 명령의 경우 미국 동부(오하이오) 리전(us-east-2)에서 공개적으로 액세스할 수 있는 S3 버킷을 사용하는 예제를 제공합니다.

> **Tip**
>
> 또한 앞선 주제의 Oracle Linux의 SSM Agent에 대한 빠른 설치 명령 절차에 있는 전역 URL을 구성한 사용자 지정 리전 URL로 바꿀 수 있습니다.

다음 명령에서 *region*을 자신의 정보로 바꿉니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

x86_64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

다음 예를 참조하세요.

```
sudo yum install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_amd64/amazon-ssm-agent.rpm
```

Red Hat Enterprise Linux 인스턴스에 SSM Agent 수동 설치

AWS에서 제공하는 Red Hat Enterprise Linux(RHEL)용 Amazon Machine Images(AMIs)에는 AWS Systems Manager 에이전트(SSM Agent)가 기본적으로 사전 설치되어 있습니다. 에이전트가 사전 설치될 수 있는 AWS에서 관리하는 AMIs 목록은 SSM Agent가 사전 설치된 상태로 AMIs 검색의 내용을 참조하세요.

이 섹션의 정보를 사용하면 RHEL 인스턴스에서 SSM Agent를 수동으로 설치 또는 재설치하는 데 도움이 됩니다.

시작하기 전 준비 사항

RHEL 인스턴스에 SSM Agent를 설치하기 전에 다음 사항에 유의하세요.

- 모든 Linux 기반 운영 체제에서의 SSM Agent 설치에 적용되는 중요한 정보는 Linux용 EC2 인스턴스에 수동으로 SSM Agent 설치 및 제거의 내용을 참조하세요.

- SSM 문서 `AWS-UpdateSSMAgent`를 사용하여 에이전트를 설치하거나 업데이트한 후 관리형 노드에서 `yum` 명령을 사용하여 SSM Agent를 업데이트하는 경우, 다음과 같은 메시지가 표시될 수 있습니다. "경고: RPMDB가 yum 외부에서 변경되었습니다." 이 메시지는 예상되는 결과이며 무시해도 됩니다.

주제

RHEL 8.x 및 9.x에 SSM Agent 설치

AWS에서 제공하는 RHEL 8 및 9용 Amazon Machine Images(AMIs)에는 AWS Systems Manager 에이전트(SSM Agent)가 기본적으로 사전 설치되어 있습니다. 이 페이지의 정보를 사용하면 RHEL 8 및 9 인스턴스에서 에이전트를 수동으로 설치 또는 재설치하는 데 도움이 됩니다.

시작하기 전 준비 사항

RHEL 8 또는 9 인스턴스에 SSM Agent를 설치하기 전에 다음을 참고합니다.

- RHEL 8 또는 9 인스턴스에 Python 2 또는 Python 3가 설치되어 있는지 확인합니다. SSM Agent가 제대로 작동하기 위해서는 이 작업을 순서대로 수행해야 합니다.

주제

SSM Agent 8 또는 9의 RHEL에 대한 빠른 설치 명령

다음 단계를 사용하여 단일 인스턴스에서 SSM Agent를 수동으로 설치합니다. 이 절차에서는 전역에서 사용 가능한 설치 파일을 사용합니다.

RHEL 8.x 또는 9.x에 SSM Agent를 설치하는 방법

1. 선호하는 방법(예: SSH)을 사용하여 RHEL 8 또는 9 인스턴스에 연결합니다.
2. 인스턴스 아키텍처에 대한 명령을 복사하고 인스턴스에서 실행합니다.

> **Note**
>
> 다음 명령의 URL에는 `ec2-downloads-windows` 디렉터리가 포함되어 있지만 RHEL 8 및 9에 대한 올바른 전역 설치 파일입니다.

x86_64 인스턴스

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

ARM64 인스턴스

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. (권장) 다음 명령을 실행하여 에이전트가 실행 중인지 확인합니다.

```
sudo systemctl status amazon-ssm-agent
```

대부분의 경우 다음 예시와 같이 명령에서 에이전트가 실행 중임을 보고합니다.

```
# 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--
```

드물게 다음 예시와 같이 명령에서 에이전트가 설치되었지만 실행 중이지 않음을 보고합니다.

```
# 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--
```

이러한 경우 에이전트를 활성화하려면 다음 명령을 실행합니다.

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

리전에서 RHEL 8 및 9용 사용자 지정 에이전트 설치 명령 생성

스크립트 또는 템플릿을 사용하여 여러 인스턴스에서 SSM Agent를 설치할 때 사용 중인 AWS 리전에 저장된 설치 파일을 사용하는 것이 좋습니다.

다음 명령의 경우 미국 동부(오하이오) 리전(us-east-2)에서 공개적으로 액세스할 수 있는 S3 버킷을 사용하는 예제를 제공합니다.

> **Tip**
>
> 또한 앞선 주제의 SSM Agent 8 또는 9의 RHEL에 대한 빠른 설치 명령 절차에 있는 전역 URL을 구성한 사용자 지정 리전 URL로 바꿀 수 있습니다.

다음 명령에서 *region*을 자신의 정보로 바꿉니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

x86_64

```
sudo dnf install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

다음 예를 참조하세요.

```
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
```

다음 예를 참조하세요.

```
sudo dnf install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_arm64/amazon-ssm-agent.rpm
```

SSM Agent 7.x에 RHEL 설치

AWS에서 제공하는 RHEL 7용 Amazon Machine Images(AMIs)에는 AWS Systems Manager 에이전트(SSM Agent)가 기본적으로 사전 설치되어 있습니다. 이 페이지의 정보를 사용하면 RHEL 7 인스턴스에서 에이전트를 수동으로 설치 또는 재설치하는 데 도움이 됩니다.

주제

- SSM Agent 7의 RHEL에 대한 빠른 설치 명령
- 리전에서 RHEL 7용 사용자 지정 에이전트 설치 명령 생성

SSM Agent 7의 RHEL에 대한 빠른 설치 명령

다음 단계를 사용하여 단일 인스턴스에서 SSM Agent를 수동으로 설치합니다. 이 절차에서는 전역에서 사용 가능한 설치 파일을 사용합니다.

RHEL 7.x에 SSM Agent를 설치하려면

1. 선호하는 방법(예: SSH)을 사용하여 RHEL 7 인스턴스에 연결합니다.
2. 인스턴스 아키텍처에 대한 명령을 복사하고 인스턴스에서 실행합니다.

> **Note**
>
> 다음 명령의 URL에는 `ec2-downloads-windows` 디렉터리가 포함되어 있지만 RHEL 7에 대한 올바른 전역 설치 파일입니다.

x86_64 인스턴스

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

ARM64 인스턴스

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. (권장) 다음 명령을 실행하여 에이전트가 실행 중인지 확인합니다.

```
sudo systemctl status amazon-ssm-agent
```

대부분의 경우 다음 예시와 같이 명령에서 에이전트가 실행 중임을 보고합니다.

```
# 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--
```

드물게 다음 예시와 같이 명령에서 에이전트가 설치되었지만 실행 중이지 않음을 보고합니다.

```
# 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--
```

이러한 경우 에이전트를 활성화하려면 다음 명령을 실행합니다.

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

리전에서 RHEL 7용 사용자 지정 에이전트 설치 명령 생성

스크립트 또는 템플릿을 사용하여 여러 인스턴스에서 SSM Agent를 설치할 때 사용 중인 AWS 리전에 저장된 설치 파일을 사용하는 것이 좋습니다.

다음 명령의 경우 미국 동부(오하이오) 리전(us-east-2)에서 공개적으로 액세스할 수 있는 S3 버킷을 사용하는 예제를 제공합니다.

> **Tip**
>
> 또한 앞선 주제의 SSM Agent 7의 RHEL에 대한 빠른 설치 명령 절차에 있는 전역 URL을 구성한 사용자 지정 리전 URL로 바꿀 수 있습니다.

다음 명령에서 *region*을 자신의 정보로 바꿉니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

x86_64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

다음 예를 참조하세요.

```
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
```

다음 예를 참조하세요.

```
sudo yum install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_arm64/amazon-ssm-agent.rpm
```

SSM Agent 6.x에 RHEL 설치

AWS에서 제공하는 RHEL 6용 Amazon Machine Images(AMIs)에는 AWS Systems Manager 에이전트(SSM Agent)가 기본적으로 사전 설치되어 있습니다. 이 페이지의 정보를 사용하면 RHEL 6 인스턴스에서 에이전트를 수동으로 설치 또는 재설치하는 데 도움이 됩니다.

주제

- SSM Agent 6의 RHEL에 대한 빠른 설치 명령
- 리전에서 RHEL 6용 사용자 지정 에이전트 설치 명령 생성

SSM Agent 6의 RHEL에 대한 빠른 설치 명령

다음 단계를 사용하여 단일 인스턴스에서 SSM Agent를 수동으로 설치합니다. 이 절차에서는 전역에서 사용 가능한 설치 파일을 사용합니다.

RHEL 6.x에 SSM Agent를 설치하려면

1. 선호하는 방법(예: SSH)을 사용하여 RHEL 6 인스턴스에 연결합니다.
2. 인스턴스 아키텍처에 대한 명령을 복사하고 인스턴스에서 실행합니다.

> **Note**
>
> 다음 명령의 URL에는 `ec2-downloads-windows` 디렉터리가 포함되어 있지만 RHEL 6에 대한 올바른 전역 설치 파일입니다.
> 다음 명령은 `3.0.1479.0` 디렉터리 대신 버전 디렉터리 `latest`을 지정합니다. 이는 SSM Agent 버전 3.1 이상이 RHEL 6에서 지원되지 않기 때문입니다.

x86_64 인스턴스

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/
SSMAgent/3.0.1479.0/linux_amd64/amazon-ssm-agent.rpm
```

x86 인스턴스

```
sudo yum install -y https://s3.amazonaws.com/ec2-downloads-windows/
SSMAgent/3.0.1479.0/linux_386/amazon-ssm-agent.rpm
```

3. (권장) 다음 명령을 실행하여 에이전트가 실행 중인지 확인합니다.

```
sudo status amazon-ssm-agent
```

대부분의 경우 다음 예시와 같이 명령에서 에이전트가 실행 중임을 보고합니다.

```
amazon-ssm-agent start/running, process 1788
```

드물게 다음 예시와 같이 명령에서 에이전트가 설치되었지만 실행 중이지 않음을 보고합니다.

```
amazon-ssm-agent stop/waiting
```

이러한 경우 에이전트를 활성화하려면 다음 명령을 실행합니다.

```
sudo start amazon-ssm-agent
```

리전에서 RHEL 6용 사용자 지정 에이전트 설치 명령 생성

스크립트 또는 템플릿을 사용하여 여러 인스턴스에서 SSM Agent를 설치할 때 사용 중인 AWS 리전에 저장된 설치 파일을 사용하는 것이 좋습니다.

다음 명령의 경우 미국 동부(오하이오) 리전(us-east-2)에서 공개적으로 액세스할 수 있는 S3 버킷을 사용하는 예제를 제공합니다.

> **Tip**
>
> 또한 앞선 주제의 SSM Agent 6의 RHEL에 대한 빠른 설치 명령 절차에 있는 전역 URL을 구성한 사용자 지정 리전 URL로 바꿀 수 있습니다.

다음 명령에서 *region*을 자신의 정보로 바꿉니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

> **Note**
>
> 다음 명령은 3.0.1390.0 디렉터리 대신 버전 디렉터리 latest을 지정합니다. 이는 SSM Agent 버전 3.1 이상이 RHEL 6에서 지원되지 않기 때문입니다.

x86_64

```
sudo yum install -y https://s3.region.amazonaws.com/amazon-ssm-region/3.0.1479.0/
linux_amd64/amazon-ssm-agent.rpm
```

다음 예를 참조하세요.

```
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
```

다음 예를 참조하세요.

```
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
```

Rocky Linux 인스턴스에 SSM Agent 수동 설치

AWS에서 제공하는 Rocky Linux용 Amazon Machine Images(AMIs)에는 AWS Systems Manager 에이전트(SSM Agent)가 기본적으로 사전 설치되어 있습니다. 에이전트가 사전 설치될 수 있는 AWS에서 관리하는 AMIs 목록은 SSM Agent가 사전 설치된 상태로 AMIs 검색의 내용을 참조하세요.

이 섹션의 정보를 사용하면 Rocky Linux 인스턴스에서 SSM Agent를 수동으로 설치 또는 재설치하는 데 도움이 됩니다.

시작하기 전 준비 사항

Rocky Linux 인스턴스에 SSM Agent를 설치하기 전에 다음 사항에 유의하세요.

- 모든 Linux 기반 운영 체제에서의 SSM Agent 설치에 적용되는 중요한 정보는 Linux용 EC2 인스턴스에 수동으로 SSM Agent 설치 및 제거의 내용을 참조하세요.

주제

- Rocky Linux의 SSM Agent에 대한 빠른 설치 명령

- 리전에서 Rocky Linux용 사용자 지정 에이전트 설치 명령 생성

Rocky Linux의 SSM Agent에 대한 빠른 설치 명령

다음 단계를 사용하여 단일 인스턴스에서 SSM Agent를 수동으로 설치합니다. 이 절차에서는 전역에서 사용 가능한 설치 파일을 사용합니다.

시작하기 전 준비 사항

Rocky Linux 인스턴스에 SSM Agent를 설치하기 전에 다음 사항에 유의하세요.

- Rocky Linux 인스턴스에 Python 2 또는 Python 3가 설치되어 있는지 확인합니다. SSM Agent가 제대로 작동하기 위해서는 이 작업을 순서대로 수행해야 합니다.

Rocky Linux에 SSM Agent를 설치하려면

1. 선호하는 방법(예: SSH)을 사용하여 Rocky Linux 인스턴스에 연결합니다.
2. 인스턴스 아키텍처에 대한 명령을 복사하고 인스턴스에서 실행합니다.

> **Note**
>
> 다음 명령의 URL에는 `ec2-downloads-windows` 디렉터리가 포함되어 있지만 Rocky Linux에 대한 올바른 전역 설치 파일입니다.

x86_64 인스턴스

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_amd64/amazon-ssm-agent.rpm
```

ARM64 인스턴스

```
sudo dnf install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/
latest/linux_arm64/amazon-ssm-agent.rpm
```

3. (권장) 다음 명령을 실행하여 에이전트가 실행 중인지 확인합니다.

```
sudo systemctl status amazon-ssm-agent
```

대부분의 경우 다음 예시와 같이 명령에서 에이전트가 실행 중임을 보고합니다.

```
# 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--
```

드물게 다음 예시와 같이 명령에서 에이전트가 설치되었지만 실행 중이지 않음을 보고합니다.

```
# 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--
```

이러한 경우 에이전트를 활성화하려면 다음 명령을 실행합니다.

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

리전에서 Rocky Linux용 사용자 지정 에이전트 설치 명령 생성

스크립트 또는 템플릿을 사용하여 여러 인스턴스에서 SSM Agent를 설치할 때 사용 중인 AWS 리전에 저장된 설치 파일을 사용하는 것이 좋습니다.

다음 명령의 경우 미국 동부(오하이오) 리전(`us-east-2`)에서 공개적으로 액세스할 수 있는 S3 버킷을 사용하는 예제를 제공합니다.

> **Tip**
>
> 또한 앞선 주제의 Rocky Linux의 SSM Agent에 대한 빠른 설치 명령 절차에 있는 전역 URL을 구성한 사용자 지정 리전 URL로 바꿀 수 있습니다.

다음 명령에서 *region*을 자신의 정보로 바꿉니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

x86_64

```
sudo dnf install -y https://s3.region.amazonaws.com/amazon-ssm-region/latest/
linux_amd64/amazon-ssm-agent.rpm
```

다음 예를 참조하세요.

```
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
```

다음 예를 참조하세요.

```
sudo dnf install -y https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
linux_arm64/amazon-ssm-agent.rpm
```

SUSE Linux Enterprise Server 인스턴스에 SSM Agent 수동 설치

대부분의 경우 AWS에서 제공하는 SUSE Linux Enterprise Server(SLES)용 Amazon Machine Images(AMIs)에는 AWS Systems Manager 에이전트(SSM Agent)가 기본적으로 사전 설치되어 있습니다. 자세한 내용은 SSM Agent가 사전 설치된 상태로 AMIs 검색 단원을 참조하십시오.

SSM Agent가 새 SLES 인스턴스에 사전 설치되어 있지 않거나 에이전트를 수동으로 다시 설치해야 하는 경우 이 페이지의 정보를 사용하면 도움이 됩니다.

시작하기 전 준비 사항

SLES 인스턴스에 SSM Agent를 설치하기 전에 다음 사항에 유의하세요.

- 모든 Linux 기반 운영 체제에서의 SSM Agent 설치에 적용되는 중요한 정보는 Linux용 EC2 인스턴스에 수동으로 SSM Agent 설치 및 제거의 내용을 참조하세요.

주제

SLES의 SSM Agent에 대한 빠른 설치 명령

다음 단계를 사용하여 단일 인스턴스에서 SSM Agent를 수동으로 설치합니다. 이 절차에서는 전역에서 사용 가능한 설치 파일을 사용합니다.

빠른 복사 및 붙여넣기 명령을 사용하여 SLES에서 SSM Agent 설치

1. 선호하는 방법(예: SSH)을 사용하여 SLES 인스턴스에 연결합니다.
2. 옵션 1: `zypper` 명령 사용

   - 다음 명령을 실행합니다:

     ```
     sudo zypper install amazon-ssm-agent
     ```

   - 프롬프트 응답으로 y를 입력합니다.

   옵션 2: `rpm` 명령 사용

   - 인스턴스에 임시 디렉터리를 만듭니다.

     ```
     mkdir /tmp/ssm
     ```

   - 임시 디렉터리로 변경합니다.

     ```
     cd /tmp/ssm
     ```

   - 다음 명령을 한 번에 하나씩 실행하여 SSM Agent 설치 관리자를 다운로드하고 실행합니다.

     > **Note**
     >
     > 다음 명령의 URL에는 `ec2-downloads-windows` 디렉터리가 포함되어 있지만 SLES에 대한 올바른 전역 설치 파일입니다.

     x86_64 인스턴스:

```
wget https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/linux_amd64/
amazon-ssm-agent.rpm
```

ARM64 인스턴스:

```
wget https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/linux_arm64/
amazon-ssm-agent.rpm
```

- 다음 명령을 실행합니다.

```
sudo rpm --install amazon-ssm-agent.rpm
```

- (권장) 다음 명령을 실행하여 에이전트가 실행 중인지 확인합니다.

```
sudo systemctl status amazon-ssm-agent
```

대부분의 경우 다음 예시와 같이 명령에서 에이전트가 실행 중임을 보고합니다.

```
# 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--
```

드물게 다음 예시와 같이 명령에서 에이전트가 설치되었지만 실행 중이지 않음을 보고합니다.

```
# 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--
```

이러한 경우 에이전트를 활성화하려면 다음 명령을 실행합니다.

```
sudo systemctl enable amazon-ssm-agent
```

```
sudo systemctl start amazon-ssm-agent
```

리전에서 SLES용 사용자 지정 에이전트 설치 명령 생성

스크립트 또는 템플릿을 사용하여 여러 인스턴스에서 SSM Agent를 설치할 때 사용 중인 AWS 리전에 저장된 설치 파일을 사용하는 것이 좋습니다.

다음 명령의 경우 미국 동부(오하이오) 리전(us-east-2)에서 공개적으로 액세스할 수 있는 S3 버킷을 사용하는 예제를 제공합니다.

> **Tip**
>
> 또한 앞선 주제의 Amazon Linux 1의 SSM Agent에 대한 빠른 설치 명령 절차에 있는 전역 URL을 구성한 사용자 지정 리전 URL로 바꿀 수 있습니다.

다음 명령에서 *region*을 자신의 정보로 바꿉니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

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
```

다음 예를 참조하세요.

```
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
```

다음 예를 참조하세요.

```
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
```

Ubuntu Server 인스턴스에 SSM Agent 수동 설치

> **Important**
>
> Ubuntu Server의 64비트 버전에 SSM Agent을(를) 설치하기 전에 올바른 설치 도구를 사용 중인지 확인합니다. 20180627로 식별된 Amazon Machine Image(AMI)부터 SSM Agent는 Snap 패키지를 사용하여 버전 16.04에 사전 설치됩니다. 이전 AMI에서 생성된 인스턴스에서는 deb 설치 관리자 패키지를 사용하여 SSM Agent를 설치해야 합니다. 자세한 내용은 64비트 Ubuntu Server 16.04 인스턴스에 설치할 올바른 SSM Agent 버전 확인 단원을 참조하십시오.

대부분의 경우 AWS에서 제공하는 Ubuntu Server용 Amazon Machine Images(AMIs)에는 AWS Systems Manager 에이전트(SSM Agent)가 기본적으로 사전 설치되어 있습니다. 자세한 내용은 SSM Agent가 사전 설치된 상태로 AMIs 검색 단원을 참조하십시오.

SSM Agent가 새 Ubuntu Server 인스턴스에 사전 설치되어 있지 않거나 에이전트를 수동으로 다시 설치해야 하는 경우 이 섹션의 정보를 사용하면 도움이 됩니다.

시작하기 전 준비 사항

Ubuntu Server 인스턴스에 SSM Agent를 설치하기 전에 다음 사항에 유의하세요.

- 모든 Linux 기반 운영 체제에서의 SSM Agent 설치에 적용되는 중요한 정보는 Linux용 EC2 인스턴스에 수동으로 SSM Agent 설치 및 제거의 내용을 참조하세요.

주제

Ubuntu Server 22.04 LTS, 20.10 STR & 20.04, 18.04 및 16.04 LTS 64비트(Snap)에 SSM Agent 설치

시작하기 전 준비 사항

Ubuntu Server 22.04 LTS, 20.10 STR & 20.04, 18.04 및 16.04 LTS 64비트(Snap)에 SSM Agent를 설치하기 전에 다음을 참고합니다.

Snap 또는 deb 설치 관리자를 사용한 버전 16.04 설치

Ubuntu Server 16.04에서 SSM Agent는 16.04 AMI의 버전에 따라 Snap 또는 deb 설치 패키지를 사용하여 설치됩니다.

SSM Agent 설치 관리자 파일 위치

Ubuntu Server22.04 LTS, 20.10 STR & 20.04, 18.04 및 16.04 LTS(Snap 포함)에서는 에이전트 바이너리 및 구성 파일을 포함한 SSM Agent 설치 관리자 파일이 `/snap/amazon-ssm-agent/current/` 디렉터리에 저장됩니다. 이 디렉터리의 구성 파일을 변경하는 경우 `/snap` 디렉터리에서 `/etc/amazon/ssm/` 디렉터리로 이러한 파일을 복사해야 합니다. 로그 및 라이브러리 파일이 변경되지 않았습니다(`/var/lib/amazon/ssm`, `/var/log/amazon/ssm`).

Snap `candidate` 채널 사용

Snap Store의 후보(candidate) 채널에는 최신 버전의 SSM Agent(최신 버그 수정 모두 포함)가 있습니다. 안정적인 채널은 아닙니다. 후보 채널과 안정적인 채널의 차이점에 대해 자세히 알아보려면 https://snapcraft.io/docs/channels의 Risk-levels를 참조하세요.

후보 채널에서 SSM Agent 버전 정보를 추적하려면 Ubuntu Server 20.10 STR 및 20.04, 18.04 및 16.04 LTS 64비트 인스턴스에서 다음 명령을 실행합니다.

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

버전 18.04 이상에서 권장되는 Snap

Ubuntu Server 22.04 LTS, 20.10 STR & 20.04 및 18.04 LTS에서는 Snap만 사용하는 것이 좋습니다. 또한 인스턴스에 에이전트가 단 한 개만 설치되어 실행 중인지 확인하십시오. Snap 없이 SSM Agent를 사용하려면 SSM Agent를 제거합니다. 그런 다음 Ubuntu Server 16.04 및 14.04 64비트(deb)에서의 SSM Agent 설치 지침을 사용하여 SSM Agent를 Debian 패키지로 설치합니다. 설치하기 전에 Debian 패키지로 관리하려는 패키지 목록과 겹치는 Snap이 설치되어 있지 않은지 확인합니다.

`Maximum timeout exceeded` 오류 메시지

Snap과 관련된 알려진 문제로 인해 `snap` 명령에서 `Maximum timeout exceeded` 오류가 발생할 수 있습니다. 이 오류가 발생하면 한 번에 하나씩 다음 명령을 실행하여 에이전트를 시작하고 중지한 다음 상태를 확인합니다.

```
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
```

Ubuntu Server 22.04 LTS, 20.10 STR & 20.04, 18.04 및 16.04 LTS 64비트 인스턴스(Snap 패키지 포함)에 SSM Agent 설치

1. SSM Agent는 식별자가 `20180627` 이후인 Ubuntu Server 22.04 LTS, 20.04, 18.04 및 16.04 LTS 64비트 AMIs에 기본적으로 설치됩니다.

   온프레미스 서버에 SSM Agent를 설치해야 하거나 에이전트를 다시 설치해야 하는 경우 다음 스크립트를 사용할 수 있습니다. `snap` 명령은 Snap 앱 스토어(https://snapcraft.io)에서 에이전트를 자동으로 다운로드하기 때문에 다운로드를 위해 URL을 지정할 필요는 없습니다.

   ```
   sudo snap install amazon-ssm-agent --classic
   ```

2. 다음 명령을 실행하여 SSM Agent가 실행 중인지 확인합니다.

   ```
   sudo snap list amazon-ssm-agent
   ```

3. 이전 명령에서 `amazon-ssm-agent is stopped`, `inactive` 또는 `disabled`가 반환되는 경우 다음 명령을 실행하여 서비스를 시작합니다.

```
sudo snap start amazon-ssm-agent
```

4. 에이전트의 상태를 확인합니다.

```
sudo snap services amazon-ssm-agent
```

Ubuntu Server 16.04 및 14.04 64비트(deb)에 SSM Agent 설치

> **Important**
>
> Ubuntu Server의 64비트 버전에 SSM Agent을(를) 설치하기 전에 올바른 설치 도구를 사용 중인지 확인합니다. 20180627로 식별된 Amazon Machine Image(AMI)부터 SSM Agent는 Snap 패키지를 사용하여 버전 16.04에 사전 설치됩니다. 이전 AMI에서 생성된 인스턴스에서는 deb 설치 관리자 패키지를 사용하여 SSM Agent를 설치해야 합니다. 자세한 내용은 64비트 Ubuntu Server 16.04 인스턴스에 설치할 올바른 SSM Agent 버전 확인 단원을 참조하세요. If SSM Agent가 Snap과 함께 인스턴스에 설치될 때 deb 설치 관리자 패키지를 사용하여 SSM Agent를 설치하거나 업데이트하는 경우 설치 또는 SSM Agent 작업이 실패할 수 있습니다.

대부분의 경우 AWS에서 제공하는 Amazon Machine Images(AMIs) Ubuntu Server 16.04에는 AWS Systems Manager 에이전트(SSM Agent)가 기본적으로 사전 설치되어 있습니다. 자세한 내용은 SSM Agent가 사전 설치된 상태로 AMIs 검색 단원을 참조하십시오.

SSM Agent가 20180627 이전 버전의 새 Ubuntu Server 16.04 인스턴스에 사전 설치되어 있지 않거나 Ubuntu Server 14.04를 설치하거나 에이전트를 수동으로 다시 설치해야 하는 경우 이 페이지의 정보를 사용하면 도움이 됩니다.

Ubuntu Server 16.04 및 14.04 64비트(deb)의 SSM Agent에 대한 빠른 설치 명령

다음 단계를 사용하여 단일 인스턴스에서 SSM Agent를 수동으로 설치합니다. 이 절차에서는 전역에서 사용 가능한 설치 파일을 사용합니다.

빠른 복사 및 붙여넣기 명령을 사용하여 Ubuntu Server 16.04 및 14.04 64비트(deb)에서 SSM Agent 설치

1. 선호하는 방법(예: SSH)을 사용하여 Ubuntu Server 인스턴스에 연결합니다.
2. 다음 명령을 실행하여 인스턴스에서 임시 디렉터리를 생성합니다.

```
mkdir /tmp/ssm
```

3. 임시 디렉터리로 변경합니다.

```
cd /tmp/ssm
```

4. 다음 명령을 실행합니다.

> **Note**
>
> 다음 명령의 URL에는 `ec2-downloads-windows` 디렉터리가 포함되어 있지만 Ubuntu Server 16.04 및 14.04 64비트에 대한 올바른 전역 설치 파일입니다.

```
wget https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/debian_amd64/
amazon-ssm-agent.deb
```

```
sudo dpkg -i amazon-ssm-agent.deb
```

5. (권장) 다음 명령 중 하나를 실행하여 SSM Agent가 실행 중인지 확인합니다.

   Ubuntu Server 16.04

   ```
   sudo systemctl status amazon-ssm-agent
   ```

   Ubuntu Server 14.04

   ```
   sudo status amazon-ssm-agent
   ```

   대부분의 경우 명령에서 에이전트가 실행 중임을 보고합니다.

   드물게 다음 예시와 같이 명령에서 에이전트가 설치되었지만 실행 중이지 않음을 보고합니다.

6. 이전 명령에서 `amazon-ssm-agent is stopped`, `inactive` 또는 `disabled`가 반환되는 경우 다음 명령 중 하나를 실행하여 서비스를 시작합니다.

   Ubuntu Server 16.04:

```
sudo systemctl enable amazon-ssm-agent
```

Ubuntu Server 14.04:

```
sudo start amazon-ssm-agent
```

리전에서 Ubuntu Server 16.04 및 14.04 64비트(deb)용 SSM Agent의 사용자 지정 설치 명령 생성

스크립트 또는 템플릿을 사용하여 여러 인스턴스에서 SSM Agent를 설치할 때 사용 중인 AWS 리전에 저장된 설치 파일을 사용하는 것이 좋습니다.

다음 명령의 경우 미국 동부(오하이오) 리전(us-east-2)에서 공개적으로 액세스할 수 있는 S3 버킷을 사용하는 예제를 제공합니다.

> **Tip**
>
> 또한 앞선 주제의 Ubuntu Server 16.04 및 14.04 64비트(deb)의 SSM Agent에 대한 빠른 설치 명령 절차에 있는 전역 URL을 구성한 사용자 지정 리전 URL로 바꿀 수 있습니다.

다음 명령에서 *region*을 자신의 정보로 바꿉니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

```
wget https://s3.region.amazonaws.com/amazon-ssm-region/latest/debian_amd64/amazon-ssm-agent.deb
```

```
sudo dpkg -i amazon-ssm-agent.deb
```

다음 예를 참조하세요.

```
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
```

Ubuntu Server 16.04 및 14.04 32비트에 SSM Agent 설치

대부분의 경우 AWS에서 제공하는 Amazon Machine Images(AMIs) Ubuntu Server 16.04에는 AWS Systems Manager 에이전트(SSM Agent)가 기본적으로 사전 설치되어 있습니다. 자세한 내용은 SSM Agent가 사전 설치된 상태로 AMIs 검색 단원을 참조하십시오.

SSM Agent가 새 Ubuntu Server 16.04 인스턴스에 사전 설치되어 있지 않거나 Ubuntu Server 14.04를 설치하거나 에이전트를 수동으로 다시 설치해야 하는 경우 이 페이지의 정보를 사용하면 도움이 됩니다.

Ubuntu Server 16.04 및 14.04 32비트(deb)에 SSM Agent를 설치하기 위한 빠른 설치 명령

다음 단계를 사용하여 단일 인스턴스에서 SSM Agent를 수동으로 설치합니다. 이 절차에서는 전역에서 사용 가능한 설치 파일을 사용합니다.

빠른 복사 및 붙여넣기 명령을 사용하여 Ubuntu Server 16.04 및 14.04 32비트에서 SSM Agent 설치

1. 선호하는 방법(예: SSH)을 사용하여 Ubuntu Server 인스턴스에 연결합니다.
2. 다음 명령을 실행하여 인스턴스에서 임시 디렉터리를 생성합니다.

```
mkdir /tmp/ssm
```

3. 임시 디렉터리로 변경합니다.

```
cd /tmp/ssm
```

4. 다음 명령을 실행합니다.

> **Note**
>
> 다음 명령의 URL에는 `ec2-downloads-windows` 디렉터리가 포함되어 있지만, 이 파일들이 Ubuntu Server 16.04 및 14.04 32비트용 올바른 글로벌 설치 파일입니다.

```
wget https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/debian_386/
amazon-ssm-agent.deb
```

```
sudo dpkg -i amazon-ssm-agent.deb
```

5. (권장) 다음 명령 중 하나를 실행하여 SSM Agent가 실행 중인지 확인합니다.

Ubuntu Server 16.04

```
sudo systemctl status amazon-ssm-agent
```

Ubuntu Server 14.04

```
sudo status amazon-ssm-agent
```

대부분의 경우 명령에서 에이전트가 실행 중임을 보고합니다.

드물게 다음 예시와 같이 명령에서 에이전트가 설치되었지만 실행 중이지 않음을 보고합니다.

6. 이전 명령에서 `amazon-ssm-agent is stopped`, `inactive` 또는 `disabled`가 반환되는 경우 다음 명령 중 하나를 실행하여 서비스를 시작합니다.

Ubuntu Server 16.04:

```
sudo systemctl enable amazon-ssm-agent
```

Ubuntu Server 14.04:

```
sudo start amazon-ssm-agent
```

리전에서 Ubuntu Server 16.04 및 14.04 32비트용 SSM Agent의 사용자 지정 설치 명령 생성

스크립트 또는 템플릿을 사용하여 여러 인스턴스에서 SSM Agent를 설치할 때 사용 중인 AWS 리전에 저장된 설치 파일을 사용하는 것이 좋습니다.

다음 명령의 경우 미국 동부(오하이오) 리전(`us-east-2`)에서 공개적으로 액세스할 수 있는 S3 버킷을 사용하는 예제를 제공합니다.

> **Tip**
>
> 또한 앞선 주제의 Ubuntu Server 16.04 및 14.04 32비트(deb)에 SSM Agent를 설치하기 위한 빠른 설치 명령 절차에 있는 전역 URL을 구성한 사용자 지정 리전 URL로 바꿀 수 있습니다.

다음 명령에서 *region*을 자신의 정보로 바꿉니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

```
wget https://s3.region.amazonaws.com/amazon-ssm-region/latest/debian_386/amazon-ssm-agent.deb
```

```
sudo dpkg -i amazon-ssm-agent.deb
```

다음 예를 참조하세요.

```
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
```

64비트 Ubuntu Server 16.04 인스턴스에 설치할 올바른 SSM Agent 버전 확인

> **Important**
>
> Ubuntu Server의 64비트 버전에 SSM Agent을(를) 설치하기 전에 올바른 설치 도구를 사용 중인지 확인합니다. 20180627로 식별된 Amazon Machine Image(AMI)부터 SSM Agent는 Snap 패키지를 사용하여 버전 16.04에 사전 설치됩니다. 이전 AMI에서 생성된 인스턴스에서는 deb 설치 관리자 패키지를 사용하여 SSM Agent를 설치해야 합니다. 자세한 내용은 64비트 Ubuntu Server 16.04 인스턴스에 설치할 올바른 SSM Agent 버전 확인 단원을 참조하세요.
> 인스턴스에 SSM Agent 설치가 두 개 이상 있는 경우(예: Snap을 사용하여 설치된 항목, deb 설치 관리자를 사용하여 설치된 항목) 에이전트 작업이 제대로 작동하지 않습니다.

다음 방법 중 하나를 사용하여 인스턴스에 대한 소스 AMI ID 생성 날짜를 확인할 수 있습니다. 이 절차는 AWS 관리형 AMIs에만 적용됩니다.

소스 AMI ID 생성 날짜 확인(콘솔)

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 인스턴스를 선택합니다.
3. 인스턴스를 선택합니다.
4. 세부 정보(Details) 탭의 AMI 이름(name) 필드 값에서 YYYYMMDD 식별자를 확인합니다. 예: `ubuntu/images/hvm-ssd/ubuntu-xenial-16.04-amd64-server-20180627`.

소스 AMI ID 생성 날짜 확인(AWS CLI)

- 다음 명령을 실행합니다.

```
aws ec2 describe-images --image-ids ami-id
```

*ami-id*는 AWS에서 제공하는 AMI의 ID를 나타냅니다(예: ami-07c8bc5c1ce9598c3).

성공하면 명령에서 다음과 같은 정보를 반환합니다. CreationDate 및 Name 필드에서 정보를 확인할 수 있습니다.

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

SSM Agent를 구성하여 Linux 노드에 프록시 사용

재정의 구성 파일을 생성하고 파일에 http_proxy, https_proxy 및 no_proxy 설정을 추가하여 HTTP 프록시를 통해 통신을 하도록 AWS Systems Manager Agent(SSM Agent)을 구성할 수 있습니다. 재정의 파일은 SSM Agent 새 버전이나 이전 버전을 설치할 경우에도 프록시 설정을 유지합니다. 이 섹션에는 upstart 및 systemd 환경에 재정의 파일을 생성하는 절차가 포함되어 있습니다. Session Manager를 사용하려는 경우 HTTPS 프록시 서버는 지원되지 않습니다.

주제

- 프록시를 사용하도록 SSM Agent 구성(Upstart)
- 프록시를 사용하도록 SSM Agent 구성(systemd)

프록시를 사용하도록 SSM Agent 구성(Upstart)

다음 절차를 사용하여 upstart 환경에 대한 재정의 구성 파일을 생성합니다.

프록시를 사용하도록 SSM Agent를 구성하려면(Upstart)

1. SSM Agent가 설치된 관리형 인스턴스에 연결합니다.
2. VIM과 같은 간단한 편집기를 열고 HTTP 프록시 서버 또는 HTTPS 프록시 서버를 사용하는지 여부에 따라 다음 구성 중 하나를 추가합니다.

   HTTP 프록시 서버의 경우:

   ```
   env http_proxy=http://hostname:port
   env https_proxy=http://hostname:port
   env no_proxy=IP address for instance metadata services (IMDS)
   ```

   HTTPS 프록시 서버의 경우:

   ```
   env http_proxy=http://hostname:port
   env https_proxy=https://hostname:port
   env no_proxy=IP address for instance metadata services (IMDS)
   ```

   > **Important**
   >
   > 파일에 no_proxy 설정을 추가하고 IP 주소를 지정합니다. no_proxy의 IP 주소는 Systems Manager의 인스턴스 메타데이터 서비스(IMDS) 엔드포인트입니다. no_proxy를 지정하지 않으면 Systems Manager에 대한 직접 호출은 프록시 서비스에서 자격 증명을 가져옵니다(ImDSv1 폴백이 활성화된 경우). 그렇지 않으면 Systems Manager에 대한 직접 호출이 실패합니다(ImDSv2가 적용된 경우).
   >
   > - IPv4의 경우 no_proxy=169.254.169.254를 지정합니다.
   > - IPv6의 경우 no_proxy=[fd00:ec2::254]를 지정합니다. 인스턴스 메타데이터 서비스의 IPv6 주소는 IMDSv2 명령과 호환됩니다. IPv6 주소는 AWS Nitro 시스템에 구축된 인스턴스에서만 액세스할 수 있습니다. 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스 메타데이터 서비스 버전 2 작동 방식을 참조하세요.

3. 파일을 /etc/init/에 amazon-ssm-agent.override라는 이름으로 저장합니다.
4. 다음 명령을 사용하여 SSM Agent를 중지하고 다시 시작합니다.

```
sudo service stop amazon-ssm-agent
sudo service start amazon-ssm-agent
```

> **Note**
>
> Upstart 환경에서 `.override` 파일을 사용하는 방법은 init: Upstart init 데몬 작업 구성을 참조하십시오.

프록시를 사용하도록 SSM Agent 구성(systemd)

다음 절차에 따라 `systemd` 환경에서 프록시를 사용하도록 SSM Agent를 구성합니다.

> **Note**
>
> 이 절차의 일부 단계에는 Snap을 사용하여 SSM Agent가 설치된 Ubuntu Server 인스턴스의 명시적 지침이 포함되어 있습니다.

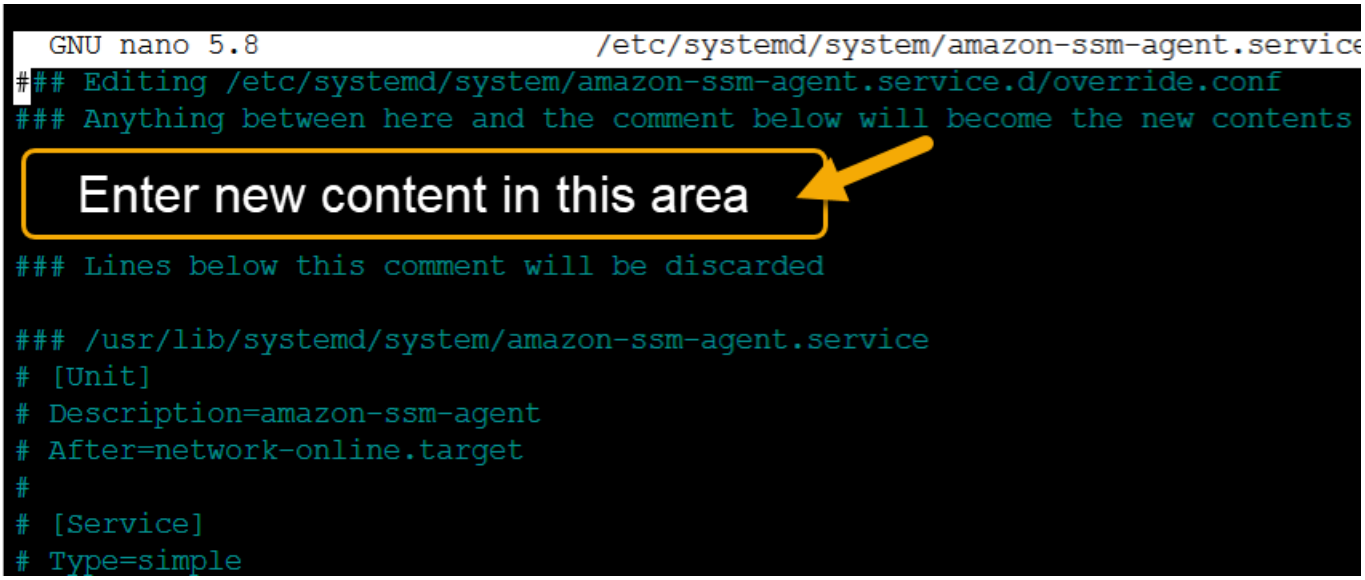
1. SSM Agent가 설치된 인스턴스에 연결합니다.
2. 운영 체제 유형에 따라 다음 명령 중 하나를 실행합니다.
   - Snap을 사용하여 SSM Agent가 설치된 Ubuntu Server 인스턴스의 경우:

     ```
     sudo systemctl edit snap.amazon-ssm-agent.amazon-ssm-agent
     ```

     기타 운영 체제의 경우:

     ```
     sudo systemctl edit amazon-ssm-agent
     ```
3. VIM과 같은 간단한 편집기를 열고 HTTP 프록시 서버 또는 HTTPS 프록시 서버를 사용하는지 여부에 따라 다음 구성 중 하나를 추가합니다.

   다음 이미지에 표시된 대로 "`### Lines below this comment will be discarded`"라고 표시된 설명 위에 정보를 입력해야 합니다.



HTTP 프록시 서버의 경우:

```
[Service]
Environment="http_proxy=http://hostname:port"
Environment="https_proxy=http://hostname:port"
Environment="no_proxy=IP address for instance metadata services (IMDS)"
```

HTTPS 프록시 서버의 경우:

```
[Service]
Environment="http_proxy=http://hostname:port"
Environment="https_proxy=https://hostname:port"
Environment="no_proxy=IP address for instance metadata services (IMDS)"
```

Important

파일에 no_proxy 설정을 추가하고 IP 주소를 지정합니다. no_proxy의 IP 주소는 Systems Manager의 인스턴스 메타데이터 서비스(IMDS) 엔드포인트입니다. no_proxy를 지정하지 않으면 Systems Manager에 대한 직접 호출은 프록시 서버에서 자격 증명을 가져옵니다(ImDSv1 폴백이 활성화된 경우). 그렇지 않으면 Systems Manager에 대한 직접 호출이 실패합니다(ImDSv2가 적용된 경우).

- IPv4의 경우 no_proxy=169.254.169.254를 지정합니다.
- IPv6의 경우 no_proxy=[fd00:ec2::254]를 지정합니다. 인스턴스 메타데이터 서비스의 IPv6 주소는 IMDSv2 명령과 호환됩니다. IPv6 주소는 AWS Nitro 시스템에 구축된

인스턴스에서만 액세스할 수 있습니다. 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스 메타데이터 서비스 버전 2 작동 방식을 참조하세요.

4. 변경 내용을 저장합니다. 운영 체제 유형에 따라 다음 파일 중 하나가 자동으로 생성됩니다.

   - Snap을 사용하여 SSM Agent가 설치된 Ubuntu Server 인스턴스의 경우:

     `/etc/systemd/system/snap.amazon-ssm-agent.amazon-ssm-agent.service.d/override.conf`

   - Amazon Linux 2 및 Amazon Linux 2023 인스턴스의 경우:

     `/etc/systemd/system/amazon-ssm-agent.service.d/override.conf`

   - 기타 운영 체제의 경우:

     `/etc/systemd/system/amazon-ssm-agent.service.d/amazon-ssm-agent.override`

5. 운영 체제 유형에 따라 다음 명령 중 하나를 사용하여 SSM Agent를 다시 시작합니다.

   - Snap을 사용하여 설치된 Ubuntu Server 인스턴스의 경우:

     ```
     sudo systemctl daemon-reload && sudo systemctl restart snap.amazon-ssm-agent.amazon-ssm-agent
     ```

   - 기타 운영 체제의 경우:

     ```
     sudo systemctl daemon-reload && sudo systemctl restart amazon-ssm-agent
     ```

> **Note**
>
> systemd 환경에서 `.override` 파일 작업에 대한 자세한 내용은 Red Hat Enterprise Linux 7 System Administrator's Guide의 Modifying Existing Unit Files를 참조하세요.

macOS용 EC2 인스턴스에서 SSM Agent 사용

AWS Systems Manager(SSM Agent)는 Systems Manager 요청을 처리하고 요청에 지정된 대로 시스템을 구성합니다. 다음 절차를 사용하여 macOS용 SSM Agent를 설치, 구성 또는 제거합니다.

> **Note**
>
> SSM Agent는 기본적으로 macOS용 Amazon Machine Images(AMIs)에 사전 설치됩니다. SSM Agent는 제거하지 않은 한 macOS에 대한 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에 설치할 필요가 없습니다.

SSM Agent의 소스 코드는 GitHub에 제공되므로 필요에 따라 에이전트를 조정할 수 있습니다. 포함하고 싶은 변경에 대해서는 풀 요청을 제출하는 것이 좋습니다. 단, AWS에서 이 소프트웨어의 수정된 사본 실행을 지원하지 않습니다.

> **Note**
>
> SSM Agent의 다양한 버전에 대한 세부 정보를 보려면 출시 정보를 참조하십시오.

macOS 운영 체제에 SSM Agent를 수동으로 설치하기 전에 다음 정보를 검토합니다.

- SSM Agent는 다음 EC2 인스턴스와 Amazon Machine Images에 기본적으로 설치됩니다.
  - macOS 10.14.x(Mojave)
  - macOS 10.15.x(Catalina)
  - macOS 11.x(BigSur)
  - macOS 12.x(Monterey)
  - macOS 13.x(Ventura)
  - macOS 14.x(Sonoma)

  SSM Agent는 제거하지 않은 한 macOS EC2 인스턴스에 수동으로 설치할 필요가 없습니다.
- 일부 AWS 리전에서는 macOS의 EC2 인스턴스가 지원되지 않습니다. macOS의 x86 기반 및 M1 EC2 인스턴스가 지원되는 리전의 목록은 Amazon EC2 FAQ에서 macOS 워크로드를 참조하세요.
- SSM Agent의 업데이트된 버전은 Systems Manager에 새 기능이 추가되거나 기존 기능에 업데이트가 발생할 때마다 릴리스됩니다. 최신 버전의 에이전트를 사용하지 못하면 관리형 노드에서 다양한 Systems Manager 기능을 사용하지 못할 수 있습니다. 이러한 이유로 시스템의 SSM Agent 상태를 최신 상태로 유지하는 프로세스를 자동화하는 것이 좋습니다. 자세한 설명은 SSM Agent 업데이트 자동화을 참조하세요. SSM Agent 업데이트에 대해 알림을 수신하려면 GitHub에서 SSM Agent 릴리스 정보 페이지를 구독합니다.

주제

SSM Agent용 EC2 인스턴스에 수동으로 macOS 설치 및 제거

macOS 인스턴스에 연결하고 다음 단계를 수행하여 AWS Systems Manager Agent(SSM Agent)를 설치합니다. Systems Manager를 사용하여 명령을 실행하는 각 인스턴스에서 이 단계를 수행합니다. 이 절차에서 제공되는 명령은 사용자 데이터를 통해 Amazon EC2 인스턴스에 스크립트로 전달될 수도 있습니다.

macOS에 SSM Agent를 설치하려면

1. 다음 명령을 사용하여 x86_64 인스턴스용 에이전트 설치 관리자 파일을 다운로드합니다.

   다음 명령에서 *region*을 자신의 정보로 바꿉니다. 지원되는 *##* 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

   ```
   sudo wget https://s3.region.amazonaws.com/amazon-ssm-region/latest/darwin_amd64/
   amazon-ssm-agent.pkg
   ```

   Apple silicon 인스턴스는 다음 명령을 사용합니다.

   ```
   sudo wget https://s3.region.amazonaws.com/amazon-ssm-region/latest/darwin_arm64/
   amazon-ssm-agent.pkg
   ```

   다음 예를 참고하세요

   ```
   sudo wget https://s3.us-east-2.amazonaws.com/amazon-ssm-us-east-2/latest/
   darwin_amd64/amazon-ssm-agent.pkg
   ```

2. 다음 명령을 사용하여 SSM Agent 설치 관리자를 실행합니다.

   x86_64:

   ```
   sudo installer -pkg amazon-ssm-agent.pkg -target /
   ```

3. 에이전트의 상태를 확인합니다.

SSM Agent의 실행 여부를 확인하려면 `/var/log/amazon/ssm/amazon-ssm-agent.log`에서 에이전트 로그를 확인합니다.

4. 에이전트 로그에 "amazon-ssm-agent가 중지됨(amazon-ssm-agent is stopped)"이라고 표시되면 다음 명령을 실행하여 서비스를 시작합니다.

```
sudo launchctl load -w /Library/LaunchDaemons/com.amazon.aws.ssm.plist && sudo
 launchctl start com.amazon.aws.ssm
```

> **Important**
>
> SSM Agent의 업데이트된 버전은 Systems Manager에 새 기능이 추가되거나 기존 기능에 업데이트가 발생할 때마다 릴리스됩니다. 최신 버전의 에이전트를 사용하지 못하면 관리형 노드에서 다양한 Systems Manager 기능을 사용하지 못할 수 있습니다. 이러한 이유로 시스템의 SSM Agent 상태를 최신 상태로 유지하는 프로세스를 자동화하는 것이 좋습니다. 자세한 설명은 SSM Agent 업데이트 자동화을 참조하세요. SSM Agent 업데이트에 대해 알림을 수신하려면 GitHub에서 SSM Agent 릴리스 정보 페이지를 구독합니다.

macOS 인스턴스에서 SSM Agent 제거

macOS는 기본적으로 PKG 파일 제거를 지원하지 않습니다. macOS에 대한 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에서 AWS Systems Manager Agent(SSM Agent)를 제거하려면 다음 위치에서 AWS 관리형 스크립트를 사용할 수 있습니다.

https://github.com/aws/amazon-ssm-agent/blob/mainline/Tools/src/update/darwin/uninstall.sh

Windows Server용 EC2 인스턴스에서 SSM Agent 사용

AWS에서 제공하는 Windows Server용 Amazon Machine Images(AMIs)에는 AWS Systems Manager 에이전트(SSM Agent)가 기본적으로 사전 설치되어 있습니다. 다음과 같은 OS(운영 체제) 버전에 대한 지원이 제공됩니다.

- Windows Server 2008-2012 R2 AMIs는 2016년 11월 이후에 게시되었습니다.
- Windows Server 2016, 2019, 2022

이전 버전의 지원 참고 사항

Windows Server 2016년 11월 이전에 발표된 AMIs는 EC2Config 서비스를 사용하여 요청을 처리하고 인스턴스를 구성합니다.

EC2Config 서비스 또는 이전 버전의 SSM Agent를 사용하여 Systems Manager 요청을 처리하는 특별한 이유가 있지 않은 한, 최신 버전의 SSM Agent를 다운로드하여 하이브리드 및 멀티클라우드 환경의 Systems Manager용으로 구성된 각 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 또는 비 EC2 시스템에 설치하는 것이 좋습니다.

2020년 1월 14일부로 Windows Server 2008은 Microsoft의 기능 또는 보안 업데이트가 더 이상 지원되지 않습니다. Windows Server 2008 및 2008 R2용 레거시 Amazon Machine Images(AMIs)에는 여전히 SSM Agent 버전 2가 사전 설치된 상태로 포함되어 있지만 Systems Manager는 더 이상 2008 버전을 공식적으로 지원하지 않으며 이러한 Windows Server 버전용 에이전트를 더 이상 업데이트하지 않습니다. 또한 SSM Agent 버전 3은 Windows Server 2008 및 2008 R2의 일부 작업과 호환되지 않을 수 있습니다. Windows Server 2008 버전에 대해 공식적으로 지원되는 최종 SSM Agent 버전은 2.3.1644.0입니다.

SSM Agent 최신 상태 유지

SSM Agent의 업데이트된 버전은 Systems Manager에 새 기능이 추가되거나 기존 기능에 업데이트가 발생할 때마다 릴리스됩니다. 최신 버전의 에이전트를 사용하지 못하면 관리형 노드에서 다양한 Systems Manager 기능을 사용하지 못할 수 있습니다. 이러한 이유로 시스템의 SSM Agent 상태를 최신 상태로 유지하는 프로세스를 자동화하는 것이 좋습니다. 자세한 설명은 SSM Agent 업데이트 자동화을 참조하세요. SSM Agent 업데이트에 대해 알림을 수신하려면 GitHub에서 SSM Agent 릴리스 정보 페이지를 구독합니다.

SSM Agent의 다양한 버전에 대한 세부 정보를 보려면 출시 정보를 참조하십시오.

주제

- SSM Agent용 EC2 인스턴스에 수동으로 Windows Server 설치 및 제거
- Windows Server 인스턴스에 프록시를 사용하도록 SSM Agent 구성

SSM Agent용 EC2 인스턴스에 수동으로 Windows Server 설치 및 제거

AWS Systems Manager 에이전트(SSM Agent)는 Amazon에서 제공하는 다음 Windows Server용 Amazon Machine Images(AMIs)에 기본적으로 사전 설치되어 있습니다.

- Windows Server 2008-2012 R2 AMIs는 2016년 11월 이후에 게시되었습니다.
- Windows Server 2016, 2019, 2022

Windows Server용 EC2 인스턴스에 SSM Agent 설치

필요할 경우 다음 절차에 따라 최신 버전의 SSM Agent를 Windows Server용 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에 수동으로 다운로드하여 설치할 수 있습니다. 이 절차에서 제공되는 명령은 사용자 데이터를 통해 Amazon EC2 인스턴스에 스크립트로 전달될 수도 있습니다.

Windows Server 인스턴스에서 레거시 `AWS-ApplyPatchBaseline` 문서와 같은 특정 AWS Systems Manager 문서(SSM 문서)를 실행하려면 SSM Agent에 Windows PowerShell 3.0 이상이 필요합니다. Windows Server 인스턴스에서 Windows Management Framework 3.0 이상을 실행하고 있는지 확인합니다. 이 프레임워크에는 Windows PowerShell이 들어 있습니다. 자세한 내용은 Windows Management Framework 3.0을 참조하십시오.

> **Note**
>
> 이 절차는 Windows Server용 EC2 인스턴스에 SSM Agent를 설치 또는 재설치하는 데 적용됩니다. 에이전트를 Systems Manager에서 사용할 수 있도록 온프레미스 서버 또는 가상 머신(VM)에 설치해야 하는 경우 하이브리드 Windows 환경에 SSM Agent를 설치하는 방법을 참조하세요.

Windows Server에 대한 EC2 인스턴스에 최신 버전의 SSM Agent를 수동으로 설치하려면

1. 원격 데스크톱이나 Windows PowerShell을 사용하여 인스턴스에 연결합니다. 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스에 연결을 참조하세요.
2. 인스턴스에 최신 버전의 SSM Agent를 다운로드합니다. PowerShell 명령 또는 직접 다운로드 링크를 사용하여 다운로드할 수 있습니다.

   > **Note**
   >
   > 이 단계의 URL을 사용하면 모든 AWS 리전에서 SSM Agent를 다운로드할 수 있습니다. 특정 리전의 에이전트를 다운로드하려는 경우 대신에 다음과 같은 리전별 URL을 사용합니다.
   > `https://amazon-ssm-region.s3.region.amazonaws.com/latest/windows_amd64/AmazonSSMAgentSetup.exe`
   > `##`은 미국 동부(오하이오) 리전의 `us-east-2` 같이 AWS Systems Manager가 지원하는 AWS 리전의 식별자를 나타냅니다. 지원되는 `##` 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

PowerShell

다음 세 개의 PowerShell 명령을 순서대로 실행합니다. 이 명령을 사용하면 IE(Internet Explorer) 보안 강화 설정을 조정하지 않고 SSM Agent를 다운로드한 다음 에이전트를 설치하고 설치 파일을 제거할 수 있습니다.

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"
```

```
rm -Force $env:USERPROFILE\Desktop\SSMAgent_latest.exe
```

직접 다운로드

다음 링크를 사용하여 SSM Agent의 최신 버전을 인스턴스에 다운로드하십시오. 원하는 경우 AWS 리전별 URL로 이 URL을 업데이트합니다.

https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/windows_amd64/AmazonSSMAgentSetup.exe

다운로드한 `AmazonSSMAgentSetup.exe` 파일을 실행하여 SSM Agent를 설치합니다.

3. PowerShell에서 다음 명령을 전송하여 SSM Agent를 시작하거나 다시 시작합니다.

```
Restart-Service AmazonSSMAgent
```

Windows Server용 EC2 인스턴스에서 SSM Agent 제거

Windows Server 인스턴스에서 SSM Agent를 제거하려면 제어판, 프로그램을 엽니다. Uninstall a program(프로그램 제거) 옵션을 선택합니다. Amazon SSM Agent를 마우스 오른쪽 버튼으로 클릭하여 컨텍스트 메뉴를 열고 Uninstall(제거)을 선택합니다.

Windows Server 인스턴스에 프록시를 사용하도록 SSM Agent 구성

이 주제의 정보는 Nano 설치 옵션을 사용하지 않는 2016년 11월 이후 생성된 Windows Server 인스턴스에 적용됩니다. Session Manager를 사용하려는 경우 HTTPS 프록시 서버는 지원되지 않습니다.

> **Note**
>
> 2020년 1월 14일부로 Windows Server 2008은 Microsoft의 기능 또는 보안 업데이트가 더 이상 지원되지 않습니다. Windows Server 2008 및 2008 R2용 레거시 Amazon Machine Images(AMIs)에는 여전히 SSM Agent 버전 2가 사전 설치된 상태로 포함되어 있지만 Systems Manager는 더 이상 2008 버전을 공식적으로 지원하지 않으며 이러한 Windows Server 버전용 에이전트를 더 이상 업데이트하지 않습니다. 또한 SSM Agent 버전 3은 Windows Server 2008 및 2008 R2의 일부 작업과 호환되지 않을 수 있습니다. Windows Server 2008 버전에 대해 공식적으로 지원되는 최종 SSM Agent 버전은 2.3.1644.0입니다.

시작하기 전 준비 사항

프록시를 사용하도록 SSM Agent를 구성하기 전에 다음 중요 정보를 확인하세요.

다음 절차에서는 명령을 실행하여 프록시를 사용하도록 SSM Agent를 구성합니다. 명령에는 IP 주소를 포함하는 `no_proxy` 설정이 포함되어 있습니다. IP 주소는 Systems Manager의 인스턴스 메타데이터 서비스(IMDS) 엔드포인트입니다. `no_proxy`를 지정하지 않으면 Systems Manager에 대한 직접 호출은 프록시 서비스에서 자격 증명을 가져옵니다(ImDSv1 폴백이 활성화된 경우). 그렇지 않으면 Systems Manager에 대한 직접 호출이 실패합니다(ImDSv2가 적용된 경우).

- IPv4의 경우 `no_proxy=169.254.169.254`를 지정합니다.
- IPv6의 경우 `no_proxy=[fd00:ec2::254]`를 지정합니다. 인스턴스 메타데이터 서비스의 IPv6 주소는 IMDSv2 명령과 호환됩니다. IPv6 주소는 AWS Nitro 시스템에 구축된 인스턴스에서만 액세스

할 수 있습니다. 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스 메타데이터 서비스 버전 2 작동 방식을 참조하세요.

프록시를 사용하도록 SSM Agent를 구성하려면

1. Remote Desktop이나 Windows PowerShell을 사용해 프록시를 사용하도록 구성하려는 인스턴스에 연결합니다.
2. PowerShell에서 다음 명령 블록을 실행합니다. *hostname* 및 *port*를 사용 중인 프록시에 대한 정보로 바꿉니다.

```
$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
```

이전 명령을 실행한 후 SSM Agent 로그를 검토하여 프록시 설정이 적용되었는지 확인할 수 있습니다. 로그의 항목은 다음과 유사합니다. SSM Agent 로그에 대한 자세한 내용은 SSM Agent 로그 보기 섹션을 참조하세요.

```
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
```

SSM Agent 프록시 구성을 재설정하려면

1. 원격 데스크톱이나 Windows PowerShell을 사용해 구성하려는 인스턴스에 연결합니다.
2. 원격 데스크톱을 사용해 연결한 경우에는 관리자 권한으로 PowerShell을 시작합니다.
3. PowerShell에서 다음 명령 블록을 실행합니다.

```
Remove-ItemProperty -Path HKLM:\SYSTEM\CurrentControlSet\Services\AmazonSSMAgent -
Name Environment
Restart-Service AmazonSSMAgent
```

SSM Agent 프록시 설정 우선순위

Windows Server 인스턴스에서 SSM Agent에 대한 프록시 설정을 구성할 때는 SSM Agent가 시작될 때 이러한 설정이 평가되고 에이전트 구성에 적용된다는 것을 이해하는 것이 중요합니다. Windows Server 인스턴스에 대한 프록시 설정을 구성하는 방법을 통해 다른 설정이 의도한 설정을 대체할 수 있는지 여부를 결정할 수 있습니다.

> **Important**
>
> SSM Agent는 HTTPS 프로토콜을 사용하여 통신합니다. 이런 이유로 다음 설정 옵션 중 하나를 사용하여 `HTTPS proxy` 파라미터를 구성해야 합니다.

SSM Agent 프록시 설정은 다음과 같은 순서로 평가됩니다.

1. AmazonSSMAgent 레지스트리 설정(`HKLM:\SYSTEM\CurrentControlSet\Services\AmazonSSMAgent`)
2. 시스템 환경 변수(`http_proxy`, `https_proxy`, `no_proxy`)
3. LocalSystem 사용자 계정 환경 변수(`http_proxy`, `https_proxy`, `no_proxy`)
4. Internet Explorer 설정(HTTP, `secure`, `exceptions`)
5. WinHTTP 프록시 설정(`http=`, `https=`, `bypass-list=`)

SSM Agent 프록시 설정 및 Systems Manager 서비스

프록시를 사용하도록 SSM Agent를 구성하고 Windows Server 인스턴스에서 실행하는 동안 PowerShell 또는 Windows Update 클라이언트를 사용하는 Run Command 및 Patch Manager와 같은

AWS Systems Manager 기능을 사용하는 경우 추가 프록시 설정을 구성해야 합니다. 그렇지 않으면 PowerShell 및 Windows Update 클라이언트에서 사용된 프록시 설정이 SSM Agent 프록시 구성에서 상속되지 않기 때문에 작업이 실패할 수 있습니다.

Run Command의 경우 Windows Server 인스턴스에서 `WinINet` 프록시 설정을 구성합니다. 제공되는 `[System.Net.WebRequest]` 명령은 세션 단위입니다. Run Command에서 실행되는 후속 네트워크 명령에 이러한 구성을 적용하려면 해당 명령이 동일한 `aws:runPowershellScript` 플러그 인 입력에서 다른 Powershell 명령 앞에 와야 합니다.

다음 PowerShell 명령은 현재 `WinINet` 프록시 설정을 반환하고 프록시 설정을 `WinINet`에 적용합니다.

```
[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
```

Patch Manager의 경우 Windows Update 클라이언트가 업데이트를 검색하고 다운로드할 수 있도록 시스템 전체에 대한 프록시 설정을 구성해야 합니다. 다음 명령은 SYSTEM 계정에서 실행되고 설정이 시스템 전체에 적용되므로 Run Command를 사용하여 실행하는 것이 좋습니다. 다음 `netsh` 명령은 현재 프록시 설정을 반환하고 프록시 설정을 로컬 시스템에 적용합니다.

```
netsh winhttp show proxy

netsh winhttp set proxy proxy-server="hostname:port" bypass-list="169.254.169.254"
```

Run Command 사용에 관한 자세한 내용은 AWS Systems Manager Run Command 섹션을 참조하세요.

SSM Agent 상태 확인 및 에이전트 시작

이 주제에서는 지원되는 각 운영 체제에서 AWS Systems Manager Agent(SSM Agent)가 실행 중인지 확인하는 명령을 나열합니다. 에이전트가 실행 중이 아닌 경우 에이전트를 시작하는 명령도 제공합니다.

운영 체제	SSM Agent 상태 확인 명령	SSM Agent 시작 명령
Amazon Linux 1	`sudo status amazon-ssm-agent`	`sudo start amazon-ssm-agent`
Amazon Linux 2 및 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 및 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 및 10	`sudo systemctl status amazon-ssm-agent`	`sudo systemctl enable amazon-ssm-agent` `sudo systemctl start amazon-ssm-agent`
macOS	`/var/log/amazon/ssm/amazon-ssm-agent.log` 에서 에이전트 로그 파일을 확인합니다.	`sudo launchctl load -w /Library/LaunchDaemons/com.amazon.aws.ssm.plist` `sudo launchctl start com.amazon.aws.ssm`
Oracle Linux	`sudo systemctl status amazon-ssm-agent`	`sudo systemctl enable amazon-ssm-agent` `sudo systemctl start amazon-ssm-agent`

운영 체제	SSM Agent 상태 확인 명령	SSM Agent 시작 명령
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	`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(모두) 및 16.04(32비트)	`sudo status amazon-ssm-agent`	`sudo start amazon-ssm-agent`
Ubuntu Server 16.04 64비트 인스턴스(deb 패키지 설치)	`sudo systemctl status amazon-ssm-agent`	`sudo systemctl enable amazon-ssm-agent` `sudo systemctl start amazon-ssm-agent`
Ubuntu Server 16.04, 18.04 및 20.04 LTS, 20.10 STR 64비트 및 22.04 LTS(Snap 패키지 설치)	`sudo systemctl status snap.amazon-ssm-agent.amazon-ssm-agent.service`	`sudo snap start amazon-ssm-agent`
Windows Server	PowerShell에서 실행: `Get-Service AmazonSSMAgent`	PowerShell 관리자 모드에서 실행: `Start-Service AmazonSSMAgent`

추가 정보

- Linux용 EC2 인스턴스에서 SSM Agent 사용

- Windows Server용 EC2 인스턴스에서 SSM Agent 사용
- SSM Agent 버전 번호 확인

SSM Agent 버전 번호 확인

특정 AWS Systems Manager 기능에는 관리형 노드에 설치할 최소 Systems Manager Agent(SSM Agent) 버전을 포함하는 사전 조건이 있습니다. Systems Manager 콘솔을 사용하거나 관리형 노드에 로그인하여 관리형 노드에 현재 설치된 SSM Agent 버전을 확인할 수 있습니다.

다음 절차에서는 관리형 노드에 현재 설치된 SSM Agent 버전을 확인하는 방법을 설명합니다.

관리형 노드에 설치된 SSM Agent의 버전 확인

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. SSM Agent 버전(version) 열에서 에이전트 버전(Agent version) 숫자를 기록합니다.

운영 체제 내에 현재 설치된 SSM Agent 버전을 확인하려면

다음 탭에서 선택하여 운영 체제 내에서 현재 설치된 SSM Agent 버전을 가져옵니다.

Amazon Linux 1, Amazon Linux 2, and Amazon Linux 2023

> **Note**
>
> 이 명령은 운영 체제의 패키지 관리자에 따라 다릅니다.

1. 관리형 노드에 로그인합니다.
2. 다음 명령을 실행합니다.

```
yum info amazon-ssm-agent
```

다음과 비슷한 출력이 반환됩니다.

```
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Installed Packages
Name        : amazon-ssm-agent
```

```
Arch        : x86_64
Version     : 3.0.655.0
```

CentOS

1. 관리형 노드에 로그인합니다.
2. CentOS 6 및 7에 대해 다음 명령을 실행합니다.

```
yum info amazon-ssm-agent
```

다음과 비슷한 출력이 반환됩니다.

```
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Installed Packages
Name        : amazon-ssm-agent
Arch        : x86_64
Version     : 3.0.655.0
```

Debian 서버

1. 관리형 노드에 로그인합니다.
2. 다음 명령을 실행합니다.

```
apt list amazon-ssm-agent
```

다음과 비슷한 출력이 반환됩니다.

```
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. 관리형 노드에 로그인합니다.
2. 다음 명령을 실행합니다.

```
pkgutil --pkg-info com.amazon.aws.ssm
```

RHEL

1. 관리형 노드에 로그인합니다.
2. RHEL 6, 7, 8 및 9에 대해 다음 명령을 실행합니다.

```
yum info amazon-ssm-agent
```

다음과 비슷한 출력이 반환됩니다.

```
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Installed Packages
Name        : amazon-ssm-agent
Arch        : x86_64
Version     : 3.0.655.0
```

DNF 패키지 유틸리티에 대해 다음 명령을 실행합니다.

```
dnf info amazon-ssm-agent
```

SLES

1. 관리형 노드에 로그인합니다.
2. SLES 12 및 15에 대해 다음 명령을 실행합니다.

```
zypper info amazon-ssm-agent
```

다음과 비슷한 출력이 반환됩니다.

```
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 서버

> **Note**
>
> Ubuntu Server 16.04 인스턴스가 deb 또는 Snap 패키지를 사용하는지 확인하기 위해서는 Ubuntu Server 인스턴스에 SSM Agent 수동 설치 섹션을 참조하세요.

1. 관리형 노드에 로그인합니다.
2. Ubuntu Server 16.04 및 14.04 64비트(deb 설치 관리자 패키지 포함)에 대해 다음 명령을 실행합니다.

   ```
   apt list amazon-ssm-agent
   ```

   다음과 비슷한 출력이 반환됩니다.

   ```
   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
   ```

   Ubuntu Server 22.04 LTS, 20.10 STR 및 20.04, 18.04, 16.04 LTS 64비트 인스턴스(Snap 패키지 포함)에 대해 다음 명령을 실행합니다.

   ```
   sudo snap list amazon-ssm-agent
   ```

   다음과 비슷한 출력이 반환됩니다.

   ```
   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. 관리형 노드에 로그인합니다.
2. 다음 PowerShell 명령을 실행합니다.

```
& "C:\Program Files\Amazon\SSM\amazon-ssm-agent.exe" -version
```

다음과 비슷한 출력이 반환됩니다.

```
SSM Agent version: 3.1.804.0
```

새로운 기능 또는 업데이트된 기능을 활용하려면 SSM Agent의 최신 버전을 사용하는 것이 좋습니다. 관리형 인스턴스가 항상 SSM Agent의 최신 버전을 실행하도록 하려면 SSM Agent을 업데이트하는 프로세스를 자동화합니다. 자세한 내용은 SSM Agent 업데이트 자동화 단원을 참조하십시오.

SSM Agent 로그 보기

AWS Systems Manager Agent(SSM Agent)는 실행, 명령, 예약된 작업, 오류 및 상태에 대한 정보를 각 관리형 노드의 로그 파일에 기록합니다. 관리형 노드에 수동으로 연결하여 로그 파일을 보거나 로그를 Amazon CloudWatch Logs에 자동으로 전송할 수 있습니다. CloudWatch Logs로 로그 전송에 대한 자세한 내용은 AWS Systems Manager 모니터링 섹션을 참조하세요.

다음 위치에서 관리형 노드에 대한 SSM Agent 로그를 볼 수 있습니다.

Linux and macOS

```
/var/log/amazon/ssm/
```

Windows

```
%PROGRAMDATA%\Amazon\SSM\Logs\
```

Linux 관리형 노드의 경우 SSM Agent `stderr` 및 `stdout` 파일은 `/var/lib/amazon/ssm/` 디렉터리에 기록됩니다.

Windows 관리형 노드의 경우 SSM Agent `stderr` 및 `stdout` 파일은 `%PROGRAMDATA%\Amazon\SSM\InstanceData\` 디렉터리에 기록됩니다.

SSM Agent 디버그 로깅 허용에 대한 자세한 내용은 SSM Agent 디버그 로깅 허용 섹션을 참조하세요.

cihub/seelog 구성에 대한 자세한 내용은 GitHub의 Seelog Wiki를 참조하세요. cihub/seelog 구성의 예는 GitHub의 cihub/seelog 예제 리포지토리를 참조하세요.

SSM Agent 디버그 로깅 허용

다음 절차를 사용하여 관리형 노드에서 SSM Agent 디버그 로깅을 허용합니다.

Linux and macOS

SSM Agent 디버그가 Linux 및 macOS 관리형 노드에 로그하도록 허용

1. AWS Systems Manager의 기능인 Session Manager를 사용하여 디버그 로깅을 허용할 관리형 노드에 연결하거나 관리형 노드에 로그인합니다. 자세한 내용은 Session Manager 작업 단원을 참조하십시오.
2. seelog.xml.template 파일을 찾습니다.

   Linux

   대부분의 Linux 관리형 노드 유형에서 파일은 /etc/amazon/ssm/seelog.xml.template 디렉터리에 있습니다.

   Ubuntu Server 20.10 STR 및 20.04, 18.04 및 16.04 LTS에서 파일은 /snap/amazon-ssm-agent/current/seelog.xml.template 디렉터리에 있습니다. 변경하기 전에 이 파일을 /snap/amazon-ssm-agent/current/ 디렉터리에서 /etc/amazon/ssm/ 디렉터리로 복사합니다.

   macOS:

   macOS 인스턴스 유형에서 파일은 /opt/aws/ssm/seelog.xml.template 디렉터리에 있습니다.
3. 파일 이름을 seelog.xml.template에서 seelog.xml로 변경합니다.

   > **Note**
   >
   > Ubuntu Server 20.10 STR 및 20.04, 18.04 및 16.04 LTS에서 파일 seelog.xml은 /etc/amazon/ssm/ 디렉터리에 생성되어야 합니다. 다음 명령을 실행하여 이 디렉터리와 파일을 생성할 수 있습니다.
   >
   > ```
   > sudo mkdir -p /etc/amazon/ssm
   > ```

```
sudo cp -p /snap/amazon-ssm-agent/current/seelog.xml.template /etc/
amazon/ssm/seelog.xml
```

4. `seelog.xml` 파일을 편집하여 기본 로깅 동작을 변경합니다. 다음 예제와 같이 minlevel 값을 info에서 debug로 변경합니다.

   ```
   <seelog type="adaptive" mininterval="2000000" maxinterval="100000000"
   critmsgcount="500" minlevel="debug">
   ```

5. (옵션) 다음 명령을 사용하여 SSM Agent를 다시 시작합니다.

   Linux:

   ```
   sudo service amazon-ssm-agent restart
   ```

   macOS:

   ```
   sudo /opt/aws/ssm/bin/amazon-ssm-agent restart
   ```

Windows

SSM Agent 디버그가 Linux 및 Windows Server 관리형 노드에 로그하도록 허용

1. Session Manager를 사용하여 디버그 로깅을 허용할 관리형 노드에 연결하거나 관리형 노드에 로그인합니다. 자세한 내용은 Session Manager 작업 단원을 참조하십시오.
2. seelog.xml.template 파일 복사본을 생성합니다. 복사본 이름을 seelog.xml로 변경합니다. 이 파일은 다음 디렉터리에 있습니다.

   ```
   %PROGRAMFILES%\Amazon\SSM\seelog.xml.template
   ```

3. `seelog.xml` 파일을 편집하여 기본 로깅 동작을 변경합니다. 다음 예제와 같이 minlevel 값을 info에서 debug로 변경합니다.

   ```
   <seelog type="adaptive" mininterval="2000000" maxinterval="100000000"
   critmsgcount="500" minlevel="debug">
   ```

4. 다음 항목을 찾습니다.

   ```
   filename="{{LOCALAPPDATA}}\Amazon\SSM\Logs\{{EXECUTABLENAME}}.log"
   ```

다음 경로를 사용하도록 이 항목을 변경합니다.

```
filename="C:\ProgramData\Amazon\SSM\Logs\amazon-ssm-agent.log"
```

5. 다음 항목을 찾습니다.

```
filename="{{LOCALAPPDATA}}\Amazon\SSM\Logs\errors.log"
```

다음 경로를 사용하도록 이 항목을 변경합니다.

```
filename="C:\ProgramData\Amazon\SSM\Logs\errors.log"
```

6. 관리자 모드에서 다음 PowerShell 명령을 사용하여 SSM Agent를 다시 시작합니다.

```
Restart-Service AmazonSSMAgent
```

SSM Agent를 통한 루트 수준 명령에 대한 액세스 제한

AWS Systems Manager 에이전트(SSM Agent)는 루트 권한(Linux) 또는 SYSTEM 권한(Windows Server)을 사용하여 하이브리드 및 멀티클라우드 환경에서 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스와 기타 시스템 유형에서 실행됩니다. 이 경우 시스템 액세스 권한이 최고 수준이기 때문에 SSM Agent로 명령을 보낼 권한이 부여된 신뢰할 수 있는 엔터티는 모두 루트 권한이나 시스템 권한을 가지고 있습니다. (AWS에서 AWS의 작업을 수행하고 리소스에 액세스할 수 있는 신뢰할 수 있는 엔터티를 보안 주체라고 합니다. 보안 주체는 AWS 계정 루트 사용자, 사용자 또는 역할일 수 있습니다.)

보안 주체가 SSM Agent에 권한이 부여된 Systems Manager 명령을 보내기 위해서는 이 수준의 액세스 권한이 필요하지만 SSM Agent의 잠재적 취약성을 이용함으로써 이 보안 주체가 악성 코드를 실행하게 될 가능성도 있습니다.

특히 SendCommand 및 StartSession 명령을 실행하는 권한은 신중하게 제한해야 합니다. 처음에는 해당 조직에서 엄선된 보안 주체에게만 각 명령에 대한 권한을 부여하는 것이 좋습니다. 단, 보안 주체가 명령을 실행할 수 있는 관리형 노드를 제한함으로써 보안 태세를 더욱 강화할 것을 권장합니다. 그러려면 해당 보안 주체에게 IAM 정책을 할당하면 됩니다. 해당 사용자가 특정한 태그 또는 태그 조합이 지정된 관리형 노드에서만 명령을 실행할 수 있도록 제한하는 조건을 IAM 정책에 포함시킬 수 있습니다.

예를 들면, 테스트용과 프로덕션용으로 하나씩 2개의 서버 플릿이 있을 수 있습니다. 하급 엔지니어에게 적용되는 IAM 정책에서는 `ssm:resourceTag/testServer` 태그가 있는 인스턴스에서만 명령을 실행할 수 있게 지정합니다. 그러나 모든 인스턴스에 액세스할 수 있는 소수의 상급 엔지니어 그룹에는

ssm:resourceTag/testServer 및 ssm:resourceTag/productionServer 태그가 둘 다 있는 인스턴스에 대한 액세스 권한을 부여합니다.

이 방식을 이용하면 하급 엔지니어가 프로덕션 인스턴스에서 명령을 실행하려고 해도 이들에게 할당된 IAM 정책이 ssm:resourceTag/productionServer 태그가 있는 인스턴스에 대한 명시적 액세스 권한을 부여하지 않기 때문에 액세스가 거부될 것입니다.

자세한 내용 및 예제는 다음 항목을 참조하십시오.

- 태그를 기반으로 Run Command 액세스 제한
- 인스턴스 태그를 기준으로 세션 액세스 제한

SSM Agent 업데이트 자동화

AWS는 Systems Manager 기능을 추가하거나 업데이트할 때 AWS Systems Manager Agent(SSM Agent)의 새 버전을 릴리스합니다. 관리형 노드가 이전 버전의 에이전트를 사용하는 경우 새 기능을 사용하거나 업데이트된 기능을 사용할 수 없습니다. 이러한 이유로 다음 방법 중 하나를 사용하여 관리형 노드에서 SSM Agent을 업데이트하는 프로세스를 자동화하는 것이 좋습니다.

Bottlerocket 운영 체제의 에이전트 업데이트

Bottlerocket 운영 체제의 SSM Agent는 Systems Manager 명령 문서 AWS-UpdateSSMAgent를 사용하여 업데이트할 수 없습니다. Bottlerocket 제어 컨테이너 내에서 업데이트가 관리됩니다. 자세한 내용은 GitHub의 Bottlerocket 제어 컨테이너 및 Bottlerocket 업데이트 인프라를 참조하세요.

macOS 버전 요구 사항

인스턴스에서 macOS 버전 11.0(Big Sur) 이상이 실행되는 경우 AWS-UpdateSSMAgent 문서를 실행하려면 인스턴스에 SSM Agent 버전이 3.1.941.0 이상이 있어야 합니다. 인스턴스가 3.1.941.0 이전에 릴리스된 SSM Agent 버전을 실행 중인 경우 brew update 및 brew upgrade amazon-ssm-agent 명령을 실행하여 AWS-UpdateSSMAgent를 실행하도록 SSM Agent를 업데이트합니다.

메서드	Details
모든 관리형 노드에서 원클릭 자동 업데이트(권장)	SSM Agent의 새 버전을 자동으로 확인하고 다운로드하도록 AWS 계정의 모든 관리형 노드를 구성할 수 있습니다. 이렇게 하려면 이 주제의 뒷부분에 설명된 대로 Fleet Manager의 설정 탭에서 SSM Agent 자동 업데이트를 선택합니다.

메서드	Details
글로벌 또는 선택적 업데이트	AWS Systems Manager의 기능인 State Manager를 사용하여 관리형 노드에 SSM Agent를 자동으로 다운로드하고 설치하는 연결을 생성할 수 있습니다. 워크로드의 중단을 제한하려는 경우 지정된 기간 동안 설치를 수행할 수 있도록 Systems Manager 유지 관리 기간을 생성할 수 있습니다. 두 방법 모두 모든 관리형 노드에 대한 글로벌 업데이트 구성을 생성하거나 업데이트할 인스턴스를 선택적으로 선택할 수 있습니다. State Manager 연결 생성에 대한 자세한 내용은 시연: SSM Agent(CLI) 자동 업데이트 섹션을 참조하세요. 유지 관리 기간을 수정하는 방법에 대한 자세한 내용은 연습: SSM Agent를 업데이트할 유지 관리 기간 생성(AWS CLI) 및 연습: SSM Agent를 자동으로 업데이트할 유지 관리 기간 생성(콘솔) 섹션을 참조하세요.
새로운 환경을 위한 글로벌 또는 선택적 업데이트	Systems Manager를 시작하는 경우 AWS Systems Manager의 기능인 Quick Setup에서 2주마다 Systems Manager(SSM) Agent 업데이트(Update Systems Manager SSM) Agent every two weeks)를 사용하는 것이 좋습니다. Quick Setup을 사용하면 모든 관리형 노드에 대한 글로벌 업데이트 구성을 생성하거나 업데이트할 관리형 노드를 선택할 수 있습니다. 자세한 내용은 Amazon EC2 호스트 관리 단원을 참조하십시오.

관리형 노드에서 SSM Agent를 수동으로 업데이트하는 것을 선호하는 경우 에이전트의 새 버전이 릴리스될 때 AWS에서 게시하는 알림에 대해 구독할 수 있습니다. 자세한 설명은 SSM Agent 알림 구독을 참조하세요. 알림을 구독한 후 Run Command를 사용하여 하나 이상의 관리형 노드를 최신 버전으로 수동으로 업데이트할 수 있습니다. 자세한 내용은 Run Command를 사용하여 SSM Agent 업데이트 단원을 참조하십시오.

SSM Agent 자동 업데이트

AWS 계정의 모든 Linux 기반 및 Windows 기반 관리형 노드에서 SSM Agent를 자동으로 업데이트하도록 Systems Manager를 구성할 수 있습니다. 이 옵션을 설정하면 Systems Manager가 에이전트의 새 버전이 있는지 격주로 자동 확인합니다. 새 버전이 있는 경우 Systems Manager는 SSM 문서 `AWS-UpdateSSMAgent`를 사용하여 에이전트를 최신 릴리스 버전으로 자동 업데이트합니다. 관리형 노드가 항상 SSM Agent의 최신 업데이트 버전을 실행할 수 있도록 이 옵션을 선택하는 것이 좋습니다.

> **Note**
>
> SSM 문서 `AWS-UpdateSSMAgent`를 사용하여 에이전트를 설치하거나 업데이트한 후 관리형 노드에서 `yum` 명령을 사용하여 SSM Agent를 업데이트하는 경우, 다음과 같은 메시지가 표시될 수 있습니다. "경고: RPMDB가 yum 외부에서 변경되었습니다." 이 메시지는 예상되는 결과이며 무시해도 됩니다.

SSM Agent을 자동 업데이트하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 설정 탭을 선택합니다.
4. 에이전트 자동 업데이트 영역에서 SSM Agent 자동 업데이트를 선택합니다.

플릿이 업데이트하는 SSM Agent 버전을 변경하려면 설정(Settings) 탭의 에이전트 자동 업데이트(Agent auto update)에서 편집(Edit)을 선택합니다. 그런 다음 파라미터(Parameters)의 버전(Version)에서 업데이트하려는 SSM Agent의 버전 번호를 입력합니다. 버전을 지정하지 않으면 에이전트는 최신 버전으로 업데이트됩니다.

계정의 모든 관리형 노드에 업데이트된 SSM Agent 버전의 자동 배포를 중지하려면 설정(Settings) 탭의 에이전트 자동 업데이트(Agent auto update)에서 삭제(Delete)를 선택합니다. 이 작업은 관리형 노드에서 SSM Agent를 자동 업데이트하는 State Manager 연결을 삭제합니다.

SSM Agent 알림 구독

AWS Systems Manager Agent(SSM Agent)의 새 버전이 릴리스되면 Amazon Simple Notification Service(Amazon SNS)가 알려줄 수 있습니다. 알림을 받으려면 다음 절차를 수행합니다.

> **Tip**
>
> GitHub에서 SSM Agent 릴리스 정보 페이지를 확인하여 알림을 구독할 수도 있습니다.

SSM Agent 알림을 구독하려면

1. https://console.aws.amazon.com/sns/v3/home에서 Amazon SNS 콘솔을 엽니다.
2. 아직 선택하지 않은 경우 탐색 모음의 리전 선택기에서 [미국 동부(버지니아 북부)(US East (N. Virginia))]를 선택합니다. 구독하려는 SSM Agent에 대한 Amazon SNS 알림이 이 리전에서 생성되기 때문에 이 AWS 리전을 선택해야 합니다.
3. 탐색 창에서 구독을 선택합니다.
4. 구독 생성을 선택합니다.
5. 구독 생성에서 다음을 수행합니다.
   a. 주제 ARN에 다음 Amazon 리소스 이름(ARN)을 사용합니다.

      ```
      arn:aws:sns:us-east-1:720620558202:SSM-Agent-Update
      ```

   b. 프로토콜에서 `Email` 또는 `SMS`를 선택합니다.
   c. 엔드포인트(Endpoint)의 경우 이전 단계에서 `Email` 또는 `SMS` 선택 여부에 따라 이메일 주소나 지역 코드와 번호를 입력하고 알림을 받습니다.
   d. 구독 생성을 선택합니다.
6. `Email`을 선택한 경우에는 구독 사실을 확인하는 이메일 메시지가 옵니다. 메시지를 열고 지침에 따라 구독을 완료합니다.

SSM Agent 새 버전이 릴리스될 때마다 구독자에게 알림이 전송됩니다. 이런 알림을 더 이상 받지 않기를 원하는 경우, 다음 절차를 수행해서 구독을 해제하세요.

SSM Agent 알림을 구독 해제하려면

1. Amazon SNS 콘솔을 엽니다.
2. 탐색 창에서 구독을 선택합니다.
3. 구독을 선택한 다음에 삭제(Delete)를 선택합니다. 확인 메시지가 나타나면 Delete(삭제)를 선택합니다.

SSM Agent 문제 해결

관리형 노드에서 작업을 실행하는 데 문제가 발생하면 AWS Systems Manager Agent(SSM Agent)에 문제가 있는 것일 수 있습니다. 다음 정보를 사용하면 SSM Agent 로그 파일을 보고 에이전트 문제를 해결하는 데 도움이 됩니다.

주제

SSM Agent가 만료됨

SSM Agent의 업데이트된 버전은 Systems Manager에 새 기능이 추가되거나 기존 기능에 업데이트가 발생할 때마다 릴리스됩니다. 최신 버전의 에이전트를 사용하지 못하면 관리형 노드에서 다양한 Systems Manager 기능을 사용하지 못할 수 있습니다. 이러한 이유로 시스템의 SSM Agent 상태를 최신 상태로 유지하는 프로세스를 자동화하는 것이 좋습니다. 자세한 설명은 SSM Agent 업데이트 자동화을 참조하세요. SSM Agent 업데이트에 대해 알림을 수신하려면 GitHub에서 SSM Agent 릴리스 정보 페이지를 구독합니다.

SSM Agent 로그 파일을 사용하여 문제 해결

SSM Agent는 다음 파일에 정보를 기록합니다. 이 파일에 있는 정보는 문제를 해결하는 데도 도움이 됩니다. 디버그 로깅을 켜는 방법을 포함하여 SSM Agent 로그 파일에 대한 자세한 내용은 SSM Agent 로그 보기 섹션을 참조하세요.

> **Note**
>
> Windows 파일 탐색기를 사용하여 이러한 로그를 보려면 폴더 옵션에서 숨긴 파일 및 시스템 파일을 볼 수 있도록 허용해야 합니다.

Windows

- `%PROGRAMDATA%\Amazon\SSM\Logs\amazon-ssm-agent.log`

- %PROGRAMDATA%\Amazon\SSM\Logs\errors.log

Linux 및 macOS에서

- /var/log/amazon/ssm/amazon-ssm-agent.log
- /var/log/amazon/ssm/errors.log

Linux 관리형 노드의 경우 /var/log 디렉터리에 기록된 messages 파일에서 자세한 정보를 찾을 수 있습니다.

에이전트 로그를 사용하는 문제 해결에 대한 자세한 내용은 AWSre:Post 지식 센터의 SSM Agent 로그를 사용하여 내 관리형 인스턴스의 SSM Agent 관련 문제를 해결하려면 어떻게 해야 하나요?를 참조하세요.

에이전트 로그 파일이 교체되지 않음(Windows)

Windows Server 관리형 노드의 seelog.xml 파일에서 날짜 기반 로그 파일 교체를 지정해도 로그가 교체되지 않는 경우 fullname=true 파라미터를 지정합니다. 다음은 fullname=true 파라미터가 지정된 seelog.xml 구성 파일의 예입니다.

```
<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>
```

SSM 엔드포인트에 연결할 수 없음

SSM Agent에서는 다음과 같은 엔드포인트에 대한 HTTPS(포트 443) 아웃바운드 트래픽을 허용해야 합니다.

- `ssm.region.amazonaws.com`
- `ssmmessages.region.amazonaws.com`

*##*은 미국 동부(오하이오) 리전의 `us-east-2` 같이 AWS Systems Manager가 지원하는 AWS 리전의 식별자를 나타냅니다. 지원되는 *##* 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

> **Note**
>
> 2024년 이전에도 `ec2messages.region.amazonaws.com`도 필요했습니다. 2024년 이전에 출시된 AWS 리전의 경우 `ssmmessages.region.amazonaws.com`으로의 트래픽을 허용해야 하지만, `ec2messages.region.amazonaws.com`으로의 트래픽 허용은 선택 사항입니다. 2024년 이후에 출시된 리전의 경우 `ssmmessages.region.amazonaws.com`으로의 트래픽을 허용해야 하지만, 이러한 리전에서는 `ec2messages.region.amazonaws.com` 엔드포인트가 지원되지 않습니다.

설명한 대로 Amazon Linux 2 또는 Amazon Linux 2023과 같은 AWS에서 제공되는 Amazon Machine Images(AMIs)를 사용해도 앞의 엔드포인트와 통신할 수 없으면 SSM Agent가 작동하지 않습니다. 네트워크 구성에 개방형 인터넷 액세스가 있거나 사용자 정의 Virtual Private Cloud(VPC) 엔드포인트가 구성되어 있어야 합니다. 사용자 정의 VPC 엔드포인트를 생성할 계획이 없다면 인터넷 게이트웨이 또는 NAT 게이트웨이를 확인합니다. VPC 엔드포인트 관리 방법에 대한 자세한 내용은 Systems Manager용 VPC 엔드포인트를 사용하여 EC2 인스턴스의 보안 개선 섹션을 참조하세요.

관리형 노드 가용성 문제 해결에 **`ssm-cli`** 사용

SSM Agent 버전 3.1.501.0부터 `ssm-cli`를 사용하여 Systems Manager에서 관리하는 기본 요구 사항을 관리형 노드에서 충족하는지를 판단하고 Fleet Manager의 관리형 노드 목록에 표시할 수 있습니다. `ssm-cli`는 SSM Agent 설치에 포함된 독립 실행형 명령줄 도구입니다. 실행 중인 것으로 확인된

Amazon EC2 인스턴스 또는 비 EC2 시스템이 Systems Manager의 관리형 노드 목록에 포함되지 않은 이유를 진단하는 데 도움이 되는 필수 정보를 수집하는 사전 구성된 명령이 포함되어 있습니다. 이러한 명령은 `get-diagnostics` 옵션을 지정할 때 실행됩니다.

자세한 내용은 ssm-cli를 사용하여 관리형 노드 가용성 문제 해결 단원을 참조하십시오.

AWS Systems Manager Quick Setup

Quick Setup의 AWS Systems Manager 기능을 사용하면 자주 사용하는 Amazon Web Services 서비스 및 기능을 권장 모범 사례로 신속하게 구성할 수 있습니다. Quick Setup에서는 일반적인 작업 또는 권장 작업을 자동화하여 Systems Manager를 비롯한 서비스 설정을 간소화합니다. 이러한 태스크에는 필수 AWS Identity and Access Management(IAM) 인스턴스 프로파일 역할 생성과 정기 패치 검사 및 인벤토리 수집과 같은 운영 모범 사례 설정이 포함됩니다. Quick Setup을 사용하는 것은 무료입니다. 그러나 설정한 서비스의 유형과 사용량 한도에 따라 비용이 발생할 수 있으며, 서비스 설정에 사용된 서비스에 대한 수수료는 없습니다. Quick Setup를 시작하려면 Systems Manager 콘솔을 엽니다. 탐색 창에서 Quick Setup를 선택합니다.

> **Note**
>
> Systems Manager에서 관리할 인스턴스를 구성하는 데 도움이 되도록 Quick Setup으로 이동한 경우 Amazon EC2 호스트 관리의 절차를 완료하세요.

Quick Setup의 이점은 무엇인가요?

Quick Setup의 이점은 다음과 같습니다.

- 서비스 및 기능 구성 간소화

  Quick Setup은 운영 모범 사례를 구성하는 과정을 안내하고 이러한 구성을 자동으로 배포합니다. Quick Setup 대시보드에 구성 배포 상태에 대한 실시간 뷰가 표시됩니다.

- 여러 계정에 자동으로 구성 배포

  Quick Setup을 AWS Organizations와 통합하여 개별 AWS 계정 또는 여러 AWS 계정 및 AWS 리전에서 사용할 수 있습니다. Quick Setup을 사용해 여러 계정을 통해 조직에서 일관된 구성을 유지할 수 있습니다.

- 구성 드리프트 제거

  구성 드리프트는 서비스나 기능의 변경 사항이 Quick Setup을 통한 선택 사항과 충돌할 때마다 발생합니다. Quick Setup은 구성 드리프트를 주기적으로 확인하고 수정을 시도합니다.

Quick Setup는 누가 사용해야 하나요?

Quick Setup은 설정하려는 서비스 및 기능에 대해 이미 어느 정도 경험이 있고 설정 프로세스를 단순화하려는 고객에게 가장 유용합니다. Quick Setup(으)로 AWS 서비스 구성에 익숙하지 않은 경우 서비스에 대해 자세히 알아보는 것이 좋습니다. Quick Setup으로 구성을 생성하기 전에 관련 사용 설명서의 내용을 검토하세요.

AWS 리전에서의 Quick Setup 가용성

다음 AWS 리전에서는 AWS Organizations에 구성한 대로 전체 조직의 모든 Quick Setup 구성 유형을 사용하거나 선택한 조직 계정 및 지역에만 사용할 수 있습니다. 또한 이 리전에서는 Quick Setup을(를) 단일 계정으로만 사용할 수 있습니다.

- 미국 동부(오하이오)
- 미국 동부(버지니아 북부)
- 미국 서부(캘리포니아 북부)
- 미국 서부(오레곤)
- 아시아 태평양(뭄바이)
- 아시아 태평양(서울)
- 아시아 태평양(싱가포르)
- 아시아 태평양(시드니)
- 아시아 태평양(도쿄)
- 캐나다(중부)
- 유럽(프랑크푸르트)
- 유럽(스톡홀름)
- 유럽(아일랜드)
- 유럽(런던)
- 유럽(파리)
- 남아메리카(상파울루)

다음 지역에서는 개별 계정에 대해 포스트 관리 구성 유형만 사용할 수 있습니다.

- 유럽(밀라노)

- 아시아 태평양(홍콩)
- 중동(바레인)
- 중국(베이징)
- China (Ningxia)
- AWS GovCloud(미국 동부)
- AWS GovCloud(미국 서부)

지원되는 모든 시스템 관리자용 리전의 목록은 Amazon Web Services 일반 참조의 시스템 관리자 서비스 엔드포인트에 있는 리전 열을 참조하세요.

Quick Setup 시작하기

이 주제의 정보를 사용하면 Quick Setup 사용을 준비하는 데 도움이 됩니다.

주제

홈 AWS 리전 구성

AWS Systems Manager 기능의 Quick Setup을 시작하려면 홈 AWS 리전을 선택하고 Quick Setup을 온보딩해야 합니다. 홈 리전은 Quick Setup이 구성을 배포하는 데 사용되는 AWS 리소스를 생성하는 위치입니다. 홈 리전을 선택한 후에는 변경할 수 없습니다.

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Quick Setup를 선택합니다.
3. Choose a home Region(홈 리전 선택)에서 Quick Setup이 구성을 배포하는 데 사용하는 AWS 리소스를 생성할 AWS 리전을 선택합니다.
4. Get started를 선택합니다.

Quick Setup 사용을 시작하려면 사용 가능한 구성 유형 목록에서 서비스 또는 기능을 선택합니다. Quick Setup의 구성 유형은 AWS 서비스 또는 기능에 따라 다릅니다. 구성 유형을 선택할 때 해당 서비스 또는 기능에 대해 구성할 옵션을 선택합니다. 기본적으로 구성 유형은 권장되는 모범 사례를 사용하도록 서비스 또는 기능을 설정하는 데 도움이 됩니다.

구성을 설정한 후 OU(조직 단위) 및 리전에서 구성 및 배포 상태에 대한 세부 정보를 볼 수 있습니다. 구성에 대한 State Manager 연결 상태도 볼 수 있습니다. State Manager는 AWS Systems Manager의 기능입니다. [구성 세부 정보(Configuration details)]창에서 Quick Setup 구성의 요약을 볼 수 있습니다. 이 요약에는 모든 계정의 세부 정보와 감지된 구성 드리프트가 포함됩니다.

Quick Setup 온보딩을 위한 IAM 역할 및 권한

온보딩하는 동안 Quick Setup은 사용자를 대신할 다음의 AWS Identity and Access Management(IAM) 역할을 생성합니다.

- `AWS-QuickSetup-StackSet-Local-ExecutionRole` - 모든 템플릿을 사용할 수 있는 AWS CloudFormation 권한을 부여합니다.
- `AWS-QuickSetup-StackSet-Local-AdministrationRole` - AWS CloudFormation에 `AWS-QuickSetup-StackSet-Local-ExecutionRole`을 수임하는 권한을 부여합니다.

관리 계정(AWS Organizations에서 조직을 생성하는 데 사용하는 계정)을 온보딩하는 경우에는 다음과 같은 역할도 Quick Setup을 통해 자동으로 생성됩니다.

- `AWS-QuickSetup-SSM-RoleForEnablingExplorer` - `AWS-EnableExplorer` 자동화 실행서에 대한 권한을 부여합니다. `AWS-EnableExplorer` 실행서는 Explorer, Systems Manager 기능을 구성해 여러 AWS 계정 및 AWS 리전에 대한 정보를 표시합니다.
- `AWSServiceRoleForAmazonSSM` - Systems Manager에서 관리 및 사용되는 AWS 리소스에 대한 액세스 권한을 부여하는 서비스 연결 역할입니다.
- `AWSServiceRoleForAmazonSSM_AccountDiscovery` - 데이터를 동기화할 때 Systems Manager가 AWS 계정 정보를 검색하는 AWS 서비스을(를) 호출할 수 있는 권한을 부여하는 서비스 연결 역할입니다. 자세한 내용은 `AWSServiceRoleForAmazonSSM_AccountDiscovery` 역할 정보 단원을 참조하십시오.

관리 계정을 온보딩할 때 Quick Setup은 AWS Organizations와 CloudFormation 사이에 신뢰할 수 있는 액세스를 활성화하여 조직 전체에 Quick Setup 구성을 배포합니다. 신뢰할 수 있는 액세스를 사용하려면 관리 계정에 관리자 권한이 있어야 합니다. 온보딩 후에는 더 이상 관리자 권한이 필요하지 않습니다. 자세한 내용은 조직에서 신뢰할 수 있는 액세스 활성화를 참조하세요.

AWS Organizations 계정 유형에 대한 자세한 내용은 AWS Organizations 사용 설명서의 AWS Organizations용어 및 개념을 참조하세요.

> **Note**
>
> Quick Setup에서는 AWS CloudFormation StackSets를 사용하여 여러 AWS 계정 및 리전에 구성을 배포합니다. 대상 계정 수에 리전 수를 곱한 값이 10,000을 초과하면 구성이 배포되지 않습니다. 사용 사례를 검토하고 조직의 성장이 수용되도록 더 적은 수의 대상을 사용하는 구성을 생성하는 것이 좋습니다. 스택 인스턴스는 조직의 관리 계정에 배포되지 않습니다. 자세한 내용은 서비스 관리형 권한으로 스택 세트 생성 시 고려 사항을 참조하세요.

사용자, 그룹 또는 역할이 다음 테이블에 나열된 API 작업에 액세스할 수 있는 경우 Quick Setup의 모든 기능을 사용할 수 있습니다. API 작업에는 두 개의 탭이 있습니다. 첫번째 탭은 모든 계정에 필요한 권한이고 두번째 탭은 조직의 관리 계정에 필요한 추가 권한에 대한 탭입니다.

Non-management account

```
"iam:CreateRole",
"iam:AttachRolePolicy",
"iam:PutRolePolicy",
"iam:GetRole",
"iam:ListRoles",
"iam:PassRole"
"ssm:ListAssociations",
"ssm:ListDocuments",
"ssm:GetDocument",
"ssm:DescribeAssociation",
"ssm:DescribeAutomationExecutions",
"cloudformation:DescribeStackSet",
"cloudformation:DescribeStackInstance",
"cloudformation:DescribeStacks",
"cloudformation:DescribeStackResources",
"cloudformation:ListStackSetOperations",
"cloudformation:ListStackSets",
"cloudformation:ListStacks",
"cloudformation:ListStackInstances",
"cloudformation:ListStackSetOperationResults",
"cloudformation:TagResource",
"cloudformation:CreateStack",
"cloudformation:DeleteStackSet",
"cloudformation:UpdateStackSet",
"cloudformation:CreateStackSet",
"cloudformation:DeleteStackInstances",
"cloudformation:CreateStackInstances"
```

Management account

```
"ssm:createResourceDataSync",
"ssm:listResourceDataSync",
"ssm:getOpsSummary",
"ssm:createAssociation",
"ssm:createDocument",
"ssm:startAssociationsOnce",
"ssm:startAutomationExecution",
"ssm:updateAssociation",
"ssm:listAssociations",
"ssm:listDocuments",
"ssm:getDocument",
"ssm:describeAssociation",
"ssm:describeAutomationExecutions",
"organizations:ListRoots",
"organizations:DescribeOrganization",
"organizations:ListOrganizationalUnitsForParent"
"organizations:EnableAWSServiceAccess",
"cloudformation:describe*"
```

Quick Setup 사용하기

AWS Systems Manager의 기능인 Quick Setup은 Quick Setup 홈 페이지의 구성(Configurations) 테이블에 각 구성의 결과를 표시합니다. 이 페이지에서 각 구성의 세부 정보 보기(View details)를 수행하거나, 작업(Actions) 드롭다운에서 구성을 삭제하거나, 구성을 생성(Create)할 수 있습니다. 구성(Configurations) 표에는 다음 정보가 포함되어 있습니다.

- 구성 유형(Configuration type) - 구성을 생성할 때 선택한 구성 유형입니다.
- 배포 유형 — 배포가 전체 조직(`Organizational`)에 적용되는지, 아니면 사용자 계정(`Local`)에만 적용되는지를 나타냅니다.
- 조직 단위(Organizational units) - 대상의 사용자 정의(Custom) 집합을 선택한 경우 구성이 배포된 OU(조직 단위)를 표시합니다. 조직 단위와 사용자 정의 대상은 조직의 관리 계정에서만 사용할 수 있습니다. 관리 계정은 AWS Organizations에서 조직을 생성하는 데 사용하는 계정입니다.

- 리전(Regions) - 현재 계정(Current account) 내에서 대상 또는 대상의 사용자 정의(Custom) 집합을 선택한 경우 구성이 배포되는 리전입니다.
- 배포 상태(Deployment status) - 배포 상태는 AWS CloudFormation이 대상 또는 스택 인스턴스를 성공적으로 배포했는지 여부를 나타냅니다. 대상 및 스택 인스턴스에는 구성 생성 중 선택한 구성 옵션이 포함되어 있습니다.
- 연결 상태(Association status) - 연결 상태는 생성한 구성에 의해 생성된 모든 연결의 상태입니다. 모든 대상에 대한 연결이 성공적으로 실행되어야 합니다. 그렇지 않으면 대상의 상태가 실패(Failed)입니다.

  Quick Setup은 각 구성 대상에 대한 State Manager 연결을 생성하고 실행합니다. State Manager는 AWS Systems Manager의 기능입니다.

구성 세부 정보

구성 세부 정보(Configuration details) 페이지에는 구성 배포와 관련 연결에 대한 정보가 표시됩니다. 이 페이지에서 구성 옵션을 편집하거나 대상을 업데이트하거나 구성을 삭제할 수 있습니다. 또한 각 구성 배포의 세부 정보를 보고 연결에 대한 자세한 정보를 얻을 수 있습니다.

구성 유형에 따라 다음 상태 그래프 중 하나 이상이 표시됩니다.

구성 배포 상태

성공, 실패 또는 실행 중이거나 보류 중인 배포 수를 표시합니다. 구성의 영향을 받는 노드를 포함하는 지정된 대상 계정 및 리전에서 배포가 발생합니다.

구성 연결 상태

성공, 실패 또는 보류 중인 State Manager 연결 수를 표시합니다. Quick Setup은 선택한 구성 옵션에 대해 각 배포에서 연결을 생성합니다.

설치 상태

구성 유형별로 수행된 작업 수와 해당 작업의 현재 상태를 표시합니다.

리소스 규정 준수

구성에 지정된 정책을 준수하는 리소스의 수를 표시합니다.

구성 세부 정보(Configuration details) 테이블에는 구성 배포에 대한 정보가 표시됩니다. 배포를 선택한 다음 세부 정보 보기(View details)를 선택하여 각 배포에 대한 세부 정보를 더 볼 수 있습니다. 각 배포의 세부 정보 페이지에는 해당 배포의 노드에 배포된 연결이 표시됩니다.

구성 편집 및 삭제

작업(Actions)을 선택한 다음 구성 옵션 편집(Edit configuration options)을 선택하여 구성 세부 정보(Configuration details) 페이지에서 구성의 구성 옵션을 편집할 수 있습니다. 구성에 새 옵션을 추가하면 Quick Setup이 배포를 실행하고 새 연결을 생성합니다. 구성에서 옵션을 제거하면 Quick Setup이 배포를 실행하고 모든 관련 연결을 제거합니다.

> **Note**
>
> 언제든지 계정에 대한 Quick Setup 구성을 편집할 수 있습니다. 조직(Organization) 구성을 편집하려면 구성 상태(Configuration status)가 성공(Success) 또는 실패(Failed) 여야 합니다.

작업(Actions)과 OU 추가(Add OUs), 리전 추가(Add Regions), OU 제거(Remove OUs) 또는 리전 제거(Remove Regions)를 선택하여 구성에 포함된 대상을 업데이트할 수도 있습니다. 계정이 관리 계정으로 구성되지 않았거나 현재 계정에 대한 구성만 생성한 경우 대상 조직 단위(OU)를 업데이트할 수 없습니다. 리전 또는 OU를 제거하면 해당 리전 또는 OU에서 연결이 제거됩니다.

구성, 작업(Actions), 구성 삭제(Delete configuration)를 차례로 선택하여 Quick Setup에서 구성을 삭제할 수 있습니다. 또는 작업(Actions) 드롭다운에서 구성 삭제(Delete configuration)를 선택하여 구성 세부 정보(Configuration details) 페이지에서 구성을 삭제할 수 있습니다. 그러면 Quick Setup에서 완료하는 데 시간이 걸릴 수 있는 모든 OU 및 리전 제거(Remove all OUs and Regions)를 수행하라는 메시지를 표시합니다. 구성을 삭제하면 모든 관련 연결도 삭제됩니다. 이 2단계 삭제 프로세스는 모든 계정 및 리전에서 배포된 리소스를 모두 제거한 다음 구성을 삭제합니다.

구성 규정 준수

둘 다 AWS Systems Manager의 기능인 Explorer 또는 규정 준수에서 인스턴스가 구성에 의해 생성된 연결을 준수하는지 여부를 확인할 수 있습니다. 규정 준수에 대한 자세한 내용은 Compliance 작업 섹션을 참조하세요. Explorer의 규정 준수 보기에 대한 자세한 내용은 AWS Systems Manager Explorer 섹션을 참조하세요.

지원되는 Quick Setup 구성 유형

지원되는 구성 유형

Quick Setup에서는 다음과 구성 유형에 대한 지원이 제공됩니다.

- Amazon EC2 호스트 관리

- 조직의 기본 호스트 관리
- AWS Config 구성 레코더
- AWS Config 규정 준수 팩 배포
- Patch Manager 조직 패치 적용 구성
- Change Manager 조직 설정
- DevOps Guru 구성
- Distributor 패키지 배포
- Amazon EC2 인스턴스 리소스 예약
- OpsCenter 조직 설정
- AWS 리소스 탐색기 구성

Amazon EC2 호스트 관리

AWS Systems Manager의 기능인 Quick Setup을 사용하여 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에서 필요한 보안 역할과 일반적으로 사용되는 Systems Manager 기능을 빠르게 구성합니다. Quick Setup을 AWS Organizations와 통합하여 개별 계정 또는 여러 계정 및 AWS 리전에서 사용할 수 있습니다. 이러한 기능은 시작하는 데 필요한 최소 권한을 제공하고 인스턴스의 상태를 관리 및 모니터링하는 데 도움이 됩니다.

Systems Manager 서비스 및 기능에 대해 잘 모르는 경우 Quick Setup으로 구성을 생성하기 전에 AWS Systems Manager 사용 설명서의 내용을 검토하여 서비스에 대해 자세히 알아보는 것이 좋습니다. Systems Manager에 대한 자세한 내용은 AWS Systems Manager이란? 섹션을 참조하세요.

Important

다음 중 하나에 해당하는 경우 Quick Setup이 EC2 관리에 사용하기에 적합한 도구가 아닐 수 있습니다.

- AWS 기능을 사용해 보기 위해 처음으로 EC2 인스턴스를 생성하려고 합니다.
- 아직 EC2 인스턴스 관리가 처음입니다.

대신 다음 내용을 살펴보는 것이 좋습니다.

- Amazon EC2 시작하기
- Amazon EC2 사용 설명서의 새 인스턴스 시작 마법사를 사용하여 인스턴스 시작

- Amazon EC2 사용 설명서의 새 인스턴스 시작 마법사를 사용하여 인스턴스 시작
- Amazon EC2 사용 설명서의 자습서: Amazon EC2 Linux 인스턴스 시작하기

EC2 인스턴스 관리에 이미 익숙하고 여러 EC2 인스턴스에 대한 구성 및 관리를 간소화하려면 Quick Setup을 사용하세요. 조직에 몇 개의 EC2 인스턴스가 있는지에 관계없이 다음 Quick Setup 절차를 사용하여 EC2 인스턴스에 대해 한 번에 여러 옵션을 구성할 수 있습니다.

필수 조건

다음 작업을 완료하기 전에 Quick Setup의 홈 리전이 이미 지정되어 있어야 합니다. 자세한 설명은 홈 AWS 리전 구성을 참조하세요.

> **Note**
>
> 이 구성 유형을 사용하면 AWS Organizations에 정의된 전체 조직, 일부 조직 계정 및 리전 또는 단일 계정에 대해 여러 옵션을 설정할 수 있습니다. 이러한 옵션 중 하나는 2주마다 SSM Agent에 대한 업데이트를 확인하고 적용하는 것입니다. 조직 관리자인 경우 기본 호스트 관리 구성 유형을 사용하여 2주마다 에이전트 업데이트로 조직의 모든 EC2 인스턴스를 업데이트하도록 선택할 수도 있습니다. 자세한 설명은 조직의 기본 호스트 관리을 참조하세요.

EC2 인스턴스의 호스트 관리 옵션 구성

호스트 관리를 설정하려면 AWS Systems Manager Quick Setup 콘솔에서 다음 태스크를 수행합니다.

호스트 관리 구성 페이지를 열려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Quick Setup를 선택합니다.
3. 호스트 관리 카드에서 생성을 선택합니다.

> **Tip**
>
> 계정에 이미 하나 이상의 구성이 있으면 먼저 구성 섹션에서 라이브러리 탭 또는 생성 버튼을 선택하여 카드를 봅니다.

Systems Manager 호스트 관리 옵션을 구성하려면

- Systems Manager 기능을 구성하려면 구성 옵션 섹션에서 구성에 사용할 Systems Manager 그룹의 옵션을 선택합니다.

2주마다 Systems Manager(SSM) 에이전트 업데이트

Systems Manager로 2주마다 에이전트의 새 버전이 있는지 확인할 수 있습니다. 새 버전이 있는 경우에는 Systems Manager에서 자동으로 관리형 노드의 에이전트를 최신 릴리스 버전으로 업데이트합니다. Quick Setup에서는 아직 없는 인스턴스에 에이전트를 설치하지 않습니다. 사전 설치된 SSM Agent가 있는 AMIs에 대한 자세한 내용은 SSM Agent가 사전 설치된 상태로 AMIs 검색를 참조하세요.

노드가 항상 SSM Agent의 최신 업데이트 버전을 실행할 수 있도록 이 옵션을 선택하는 것이 좋습니다. 에이전트를 수동으로 설치하는 방법을 포함하여 SSM Agent에 대한 자세한 내용은 SSM Agent 작업 섹션을 참조하세요.

인스턴스에서 30분마다 인벤토리 수집

Quick Setup으로 다음 유형의 메타데이터 수집을 구성할 수 있습니다.

- AWS 구성 요소 - EC2 드라이버, 에이전트, 버전 등
- 애플리케이션 - 애플리케이션 이름, 게시자, 버전 등
- 노드 세부 정보 - 시스템 이름, 운영 체제(OS) 이름, OS 버전, 마지막 부팅, DNS, 도메인, 작업 그룹, OS 아키텍처 등
- 네트워크 구성 - IP 주소, MAC 주소, DNS, 게이트웨이, 서브넷 마스크 등
- 서비스 - 이름, 표시 이름, 상태, 종속 서비스, 서비스 유형, 시작 유형 등(Windows Server 노드만 해당)
- Windows 역할 - 이름, 표시 이름, 경로, 기능 유형, 설치된 상태 등(Windows Server 노드만 해당)
- Windows 업데이트 - Hotfix ID, 설치한 사람, 설치한 날짜 등(Windows Server 노드만 해당)

AWS Systems Manager의 기능인 Inventory에 대한 자세한 내용은 AWS Systems Manager Inventory 섹션을 참조하세요.

> **Note**
>
> 인벤토리 수집(Inventory collection) 옵션은 소수의 노드만 선택하더라도 완료하는 데 최대 10분이 걸릴 수 있습니다.

인스턴스의 패치 누락 여부 매일 스캔

Systems Manager의 기능인 Patch Manager를 사용하여 매일 노드를 스캔하고 Compliance 페이지에서 보고서를 생성할 수 있습니다. 보고서는 기본 패치 기준선에 따라 패치 규정을 준수한 노드의 수를 보여줍니다. 보고서에는 각 노드 및 규정 준수 상태 목록이 포함되어 있습니다.

패치 적용 작업 및 패치 기준선에 대한 자세한 내용은 AWS Systems Manager Patch Manager 섹션을 참조하세요.

패치 규정 준수에 대한 정보는 Systems Manager Compliance(규정 준수) 페이지를 참조하세요.

하나의 구성에서 여러 계정 및 리전의 관리형 노드를 패치하는 방법에 대한 자세한 내용은 Quick Setup 패치 정책 사용 및 Patch Manager 조직 패치 적용 구성 섹션을 참조하세요.

> **Important**
>
> Systems Manager는 몇 가지 관리형 노드 패치 규정 준수 검사 방법을 지원합니다. 이러한 방법을 한 번에 두 가지 이상 구현할 경우 항상 가장 최근 검사의 결과가 패치 규정 준수 정보로 표시됩니다. 이전 검사의 결과는 덮어씁니다. 검사 방법에서 승인 규칙이 서로 다른 여러 패치 기준선을 사용하는 경우, 패치 규정 준수 정보가 예기치 않게 변경될 수 있습니다. 자세한 내용은 의도치 않은 패치 규정 준수 데이터 덮어쓰기 방지 단원을 참조하십시오.

Amazon CloudWatch 호스트 관리 옵션을 구성하려면

- CloudWatch 기능을 구성하려면 구성 옵션 섹션에서 구성을 위해 활성화하려는 Amazon CloudWatch 그룹의 옵션을 선택합니다.

CloudWatch 에이전트 설치 및 구성

Amazon EC2 인스턴스에 통합된 CloudWatch 에이전트의 기본 구성을 설치합니다. 에이전트는 Amazon CloudWatch에 대한 인스턴스에서 지표와 로그 파일을 수집합니다. 이 정보는 통합되어 있어 인스턴스의 상태를 빠르게 확인할 수 있습니다. CloudWatch 에이전트 기본 구성에 대한 자세한 내용은 CloudWatch 에이전트 사전 정의된 지표 세트를 참조하세요. 비용이 추가될 수 있습니다. 자세한 내용은 Amazon CloudWatch 요금을 참조하십시오.

30일마다 CloudWatch 에이전트 업데이트

Systems Manager로 30일마다 CloudWatch 에이전트의 새 버전이 있는지 확인할 수 있습니다. 새 버전이 있는 경우 Systems Manager는 인스턴스의 에이전트를 업데이트합니다. 인스턴스가 항상 CloudWatch 에이전트의 최신 업데이트 버전을 실행할 수 있도록 이 옵션을 선택하는 것이 좋습니다.

Amazon EC2 시작 에이전트 호스트 관리 옵션을 구성하려면

- Amazon EC2 시작 에이전트 기능을 구성하려면 구성 옵션 섹션에서 구성을 위해 활성화하려는 Amazon EC2 시작 에이전트 그룹의 옵션을 선택합니다.

30일마다 EC2 시작 에이전트 업데이트

Systems Manager로 30일마다 인스턴스에 설치된 시작 에이전트의 새 버전을 확인할 수 있습니다. 새 버전을 사용할 수 있는 경우 Systems Manager에서 인스턴스의 에이전트를 업데이트합니다. 해당하는 시작 에이전트의 최신 버전이 항상 인스턴스에서 실행되도록 이 옵션을 선택하는 것이 좋습니다. Amazon EC2 Windows 인스턴스의 경우 이 옵션에서는 EC2Launch, EC2Launch v2 및 EC2Config를 지원합니다. Amazon EC2 Linux 인스턴스의 경우 이 옵션에서는 `cloud-init`을 지원합니다. Amazon EC2 Mac 인스턴스의 경우 이 옵션에서는 `ec2-macos-init`을 지원합니다. Quick Setup에서는 시작 에이전트를 통해 지원되지 않는 운영 체제 또는 AL2023에 설치된 시작 에이전트 업데이트를 지원하지 않습니다.

이러한 초기화 에이전트에 대한 자세한 내용은 다음과 같은 주제를 참조하세요.

- EC2Launch v2를 사용하여 Windows 인스턴스 구성
- EC2Launch를 사용하여 Windows 인스턴스 구성
- EC2Config 서비스를 사용하여 Windows 인스턴스 구성
- cloud-init 설명서

- ec2-macos-init

호스트 관리 구성으로 업데이트할 EC2 인스턴스를 선택하려면

- 대상 섹션에서 구성을 배포할 계정 및 리전을 결정하는 방법을 선택합니다.

> **Note**
>
> 동일한 AWS 리전을 대상으로 하는 Quick Setup 호스트 관리 구성을 여러 개 생성할 수 없습니다.

Entire organization

구성은 조직 내 모든 조직 단위(OU)와 AWS 리전에 배포됩니다.

> **Note**
>
> 조직 전체(Entire organization) 옵션은 조직의 관리 계정에서 호스트 관리를 구성하는 경우에만 사용할 수 있습니다.

Custom

1. 대상 OU 섹션에서 호스트 관리 구성을 배포할 OU를 선택합니다.
2. 대상 리전 섹션에서 호스트 관리 구성을 배포할 리전을 선택합니다.

Current account

리전 옵션 중 하나를 선택하고 해당 옵션의 단계를 따릅니다.

현재 리전

현재 리전의 인스턴스만 대상으로 하는 방법을 선택합니다.

- 모든 인스턴스-호스트 관리 구성은 자동으로 현재 리전의 모든 EC2를 대상으로 합니다.

- 태그-추가를 선택하고 대상으로 지정할 인스턴스에 추가된 키와 선택적 값을 입력합니다.
- 리소스 그룹-리소스 그룹의 경우 대상으로 지정할 EC2 인스턴스가 포함된 기존 리소스 그룹을 선택합니다.
- 수동-인스턴스 섹션에서 대상으로 지정할 각 EC2 인스턴스의 확인란을 선택합니다.

리전 선택

다음 중 하나를 선택하여 지정한 리전의 인스턴스를 대상으로 하는 방법을 선택합니다.

- 모든 인스턴스-지정한 리전의 모든 인스턴스를 대상으로 합니다.
- 태그-추가를 선택하고 대상으로 지정할 인스턴스에 추가된 키와 선택적 값을 입력합니다.

대상 리전 섹션에서 호스트 관리 구성을 배포할 리전을 선택합니다.

인스턴스 프로파일 옵션을 지정하려면

- 전체 조직 및 사용자 지정 대상에만 해당됩니다.

  [인스턴스 프로파일 옵션(Instance profile options)] 섹션에서 인스턴스에 연결된 기존 인스턴스 프로파일에 필요한 IAM 정책을 추가할지 아니면 Quick Setup에서 선택한 구성에 필요한 권한으로 IAM 정책 및 인스턴스 프로필을 생성하도록 허용할지 선택합니다.

구성 선택 항목을 모두 지정한 후 생성을 선택합니다.

조직의 기본 호스트 관리

AWS Systems Manager의 기능인 Quick Setup을(를) 사용하면 AWS Organizations에서 조직에 추가된 모든 계정 및 리전에 대해 기본 호스트 관리 구성을 활성화할 수 있습니다. 이렇게 하면 조직 내 모든 Amazon Elastic Compute Cloud(EC2) 인스턴스에서 SSM Agent이(가) 최신 상태로 유지되고 시스템 관리자에 연결할 수 있습니다.

시작하기 전 준비 사항

이 설정을 활성화하기 전에 다음 요구 사항을 충족하는지 확인하세요.

- 다음 작업을 완료하기 전에 Quick Setup의 홈 리전이 이미 지정되어 있어야 합니다. 자세한 설명은 홈 AWS 리전 구성을 참조하세요.
- 조직에서 관리할 모든 EC2 인스턴스에 최신 버전의 SSM Agent이(가) 이미 설치되어 있습니다.

- 관리할 EC2 인스턴스가 인스턴스 메타데이터 서비스 버전 2(IMDSv2)를 사용하고 있습니다.
- AWS Organizations에 명시된 대로 관리자 권한이 있는 AWS Identity and Access Management(IAM) ID(사용자, 역할 또는 그룹)를 사용하여 조직의 관리 계정에 로그인합니다.

기본 EC2 인스턴스 관리 역할 사용

기본 호스트 관리 구성에서는 시스템 관리자를 위한 `default-ec2-instance-management-role` 서비스 설정을 사용합니다. 이 역할에는 인스턴스의 SSM Agent 및 클라우드의 시스템 관리자 서비스 간의 통신을 허용하기 위해 조직의 모든 계정에 제공하려는 권한이 있습니다.

update-service-setting CLI 명령을 사용하여 이 역할을 이미 설정한 경우 기본 호스트 관리 구성에서는 해당 역할을 사용합니다. 이 역할을 아직 설정하지 않은 경우 Quick Setup은(는) 이 역할을 생성하여 적용합니다.

조직에 이 역할이 이미 지정되었는지 확인하려면 get-service-setting 명령을 사용하세요.

2주마다 SSM Agent의 자동 업데이트를 활성화합니다

다음 절차를 사용하여 전체 AWS Organizations 조직에 대해 기본 호스트 관리 구성 옵션을 활성화하세요.

2주마다 SSM Agent의 자동 업데이트를 활성화하기

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Quick Setup를 선택합니다.
3. 기본 호스트 관리 구성 카드에서 생성을 선택합니다.

> **Tip**
>
> 계정에 이미 하나 이상의 구성이 있으면 먼저 구성 섹션에서 라이브러리 탭 또는 생성 버튼을 선택하여 카드를 봅니다.

4. 구성 옵션 섹션에서 SSM Agent 2주마다 자동 업데이트 활성화를 선택합니다.
5. 생성을 선택합니다.

AWS Config 구성 레코더

AWS Systems Manager의 기능인 Quick Setup을 사용하면 AWS Config로 구동되는 구성 레코더를 빠르게 생성할 수 있습니다. 구성 레코더를 사용하여 리소스 구성의 변경 사항을 발견하고 이러한 변경 사항을 구성 항목으로 캡처합니다. AWS Config에 대해 잘 모르는 경우 Quick Setup으로 구성을 생성하기 전에 AWS Config Developer Guide의 내용을 검토하여 서비스에 대해 자세히 알아보는 것이 좋습니다. AWS Config에 대한 자세한 내용은 AWS Config 개발자 안내서의 AWS Config란 무엇입니까?를 참조하세요.

기본적으로 구성 레코더는 AWS Config가 실행되는 AWS 리전에서 지원되는 모든 리소스를 기록합니다. 지정하는 리소스 유형만 기록되도록 구성을 사용자 지정할 수 있습니다. 자세한 내용은 AWS Config Developer Guide의 Selecting which resources AWS Config records를 참조하세요.

AWS Config가 구성 기록을 시작하면 서비스 이용 요금이 청구됩니다. 요금 정보는 AWS Config 요금을 참조하세요.

> **Note**
>
> 구성 레코더를 이미 생성했다면 Quick Setup에서는 레코딩을 중지하거나 이미 레코딩 중인 리소스 유형을 변경하지 않습니다. Quick Setup을 사용하여 추가 리소스 유형을 레코딩하도록 선택하면 서비스에서는 해당 리소스 유형을 기존 레코더 그룹에 추가합니다. Quick Setup Config 기록(Config recording) 구성 유형을 삭제해도 구성 레코더가 중지되지 않습니다. 변경 사항은 계속 기록되며 구성 레코더를 중지할 때까지 서비스 사용 요금이 적용됩니다. 구성 레코더 관리에 대한 자세한 내용은 AWS Config Developer Guide의 Managing the Configuration Recorder를 참조하세요.

필수 조건

다음 작업을 완료하기 전에 Quick Setup의 홈 리전이 이미 지정되어 있어야 합니다. 자세한 설명은 홈 AWS 리전 구성을 참조하세요.

AWS Config 기록을 설정하려면 AWS Systems Manager 콘솔에서 다음 태스크를 수행합니다.

Quick Setup으로 AWS Config 기록을 설정하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Quick Setup를 선택합니다.
3. Config 레코딩 카드에서 생성을 선택합니다.

> **Tip**
>
> 계정에 이미 하나 이상의 구성이 있으면 먼저 구성 섹션에서 라이브러리 탭 또는 생성 버튼을 선택하여 카드를 봅니다.

4. 구성 옵션 섹션에서 다음을 수행합니다.

   a. 기록할 AWS 리소스 유형 선택에서 모든 지원 리소스를 기록할지 또는 선택한 리소스 유형만 기록할지 지정합니다.

   b. 전송 설정에서 Amazon Simple Storage Service(Amazon S3) 버킷을 새로 생성할지 또는 구성 스냅샷을 전송할 기존 버킷을 선택할지 지정합니다.

   c. 알림 옵션에서 원하는 알림 옵션을 선택합니다. AWS Config는 Amazon Simple Notification Service(Amazon SNS)를 사용하여 리소스와 관련된 중요한 AWS Config 이벤트를 알려줍니다. [기존 SNS 주제 사용(Use existing SNS topics)] 옵션을 선택하는 경우 해당 계정에서 사용하려는 기존 Amazon SNS 주제의 AWS 계정 ID와 이름을 제공해야 합니다. 여러 AWS 리전을 대상으로 하는 경우 각 리전에서 주제 이름이 동일해야 합니다.

5. [일정(Schedule)] 섹션에서 구성과 다른 리소스에 대한 변경 사항을 Quick Setup에서 수정할 빈도를 선택합니다. [기본값(Default)] 옵션은 한 번 실행됩니다. Quick Setup에서 구성과 다른 리소스에 대한 변경 사항을 수정하지 않도록 하려면 [사용자 정의(Custom)]에서 [수정 사용 중지(Disable remediation)]를 선택합니다.

6. 대상 섹션에서 다음 중 하나를 선택하여 기록할 계정 및 리전을 식별합니다.

> **Note**
>
> 단일 계정에서 작업하는 경우 조직 및 조직 단위(OU) 작업을 위한 옵션을 사용할 수 없습니다. 이 구성을 계정의 모든 AWS 리전에 적용할지 아니면 선택한 리전에만 적용할지 선택할 수 있습니다.

- Entire organization(전체 조직) - 조직 내 모든 계정 및 리전
- Custom(사용자 지정) - 사용자가 지정한 OU 및 리전만
  - 대상 OU 섹션에서 기록을 허용할 OU을 선택합니다.
  - 대상 리전 섹션에서 기록을 허용할 리전을 선택합니다.
- Current account(현재 계정) - 현재 로그인한 계정에서 지정한 리전만 대상으로 합니다. 다음 중 하나를 선택합니다.

- Current Region(현재 리전) - 콘솔에서 선택한 리전의 관리형 노드만 대상으로 합니다.
- 리전 선택 - 기록 구성을 적용할 개별 리전을 선택합니다.

7. 생성(Create)을 선택합니다.

AWS Config 규정 준수 팩 배포

적합성 팩은 AWS Config 규칙 및 수정 작업의 모음입니다. Quick Setup을 사용하면 계정과 AWS 리전의 단일 엔터티 또는 AWS Organizations의 조직 전체에 적합성 팩을 배포할 수 있습니다. 이를 통해 공통 프레임워크 및 패키징 모델을 사용하여 정책 정의에서 감사 및 집계 보고에 이르기까지 AWS 리소스의 구성 규정 준수를 대규모로 관리할 수 있습니다.

필수 조건

다음 작업을 완료하기 전에 Quick Setup의 홈 리전이 이미 지정되어 있어야 합니다. 자세한 설명은 홈 AWS 리전 구성을 참조하세요.

적합성 팩을 배포하려면 AWS Systems Manager Quick Setup콘솔에서 다음 태스크를 수행합니다.

> **Note**
>
> 이 구성을 배포하기 전에 AWS Config 기록을 사용해야 합니다. 자세한 내용은 AWS Config Developer Guide의 Conformance packs를 참조하세요.

Quick Setup으로 적합성 팩을 배포하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Quick Setup를 선택합니다.
3. 규정 준수 팩 카드에서 생성을 선택합니다.

> **Tip**
>
> 계정에 이미 하나 이상의 구성이 있으면 먼저 구성 섹션에서 라이브러리 탭 또는 생성 버튼을 선택하여 카드를 봅니다.

4. 규정 준수 팩 선택 섹션에서 배포하려는 규정 준수 팩을 선택합니다.

> **Note**
>
> AWS 관리형 규정 준수 팩 외에도 직접 생성한 사용자 정의 규정 준수 팩을 선택할 수 있습니다. 자세한 내용은 AWS Config 개발자 설명서에서 다음 주제를 참조하세요.
>
> - 사용자 지정 규정 준수 팩
> - AWS Config 콘솔을 사용하여 적합성 팩 배포
> - AWS Command Line Interface를 사용하여 규정 준수 팩 배포

5. [일정(Schedule)] 섹션에서 구성과 다른 리소스에 대한 변경 사항을 Quick Setup에서 수정할 빈도를 선택합니다. [기본값(Default)] 옵션은 한 번 실행됩니다. Quick Setup에서 구성과 다른 리소스에 대한 변경 사항을 수정하지 않도록 하려면 [사용자 정의(Custom)]에서 [사용 중지됨(Disabled)]을 선택합니다.
6. [대상(Targets)] 섹션에서 전체 조직, 일부 AWS 리전 또는 현재 로그인한 계정에 적합성 팩을 배포할지 선택합니다.

   [전체 조직(Entire organization)]을 선택하는 경우 8단계로 진행합니다.

   [사용자 정의(Custom)]를 선택하는 경우 7단계로 진행합니다.
7. [대상 리전(Target Regions)] 섹션에서 적합성 팩을 배포할 리전의 확인란을 선택합니다.
8. 생성(Create)을 선택합니다.

Patch Manager 조직 패치 적용 구성

이제 AWS Systems Manager의 기능인 Quick Setup을 사용하여 Patch Manager 기반 패치 정책을 생성할 수 있습니다. 패치 정책은 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스와 기타 관리형 노드를 자동으로 패치할 때 사용할 일정과 기준선을 정의합니다. 단일 패치 정책 구성을 사용하여 조직 내 여러 AWS 리전의 모든 계정이나, 선택한 계정 및 리전이나, 단일 계정-리전 쌍에 대한 패치 적용을 정의할 수 있습니다. 패치 정책에 대한 자세한 내용은 Quick Setup 패치 정책 사용 섹션을 참조하세요.

전제 조건

Quick Setup을 사용하여 노드에 대한 패치 정책을 정의하려면, 해당 노드가 관리형 노드여야 합니다. 노드 관리에 대한 자세한 내용은 AWS Systems Manager 설정 섹션을 참조하세요.

Important

패치 규정 준수 검사 방법-Systems Manager는 몇 가지 관리형 노드 패치 규정 준수 검사 방법을 지원합니다. 이러한 방법을 한 번에 두 가지 이상 구현할 경우 항상 가장 최근 검사의 결과가 패치 규정 준수 정보로 표시됩니다. 이전 검사의 결과는 덮어씁니다. 검사 방법에서 승인 규칙이 서로 다른 여러 패치 기준선을 사용하는 경우, 패치 규정 준수 정보가 예기치 않게 변경될 수 있습니다. 자세한 내용은 의도치 않은 패치 규정 준수 데이터 덮어쓰기 방지 단원을 참조하십시오.

연결 규정 준수 상태 및 패치 정책-Quick Setup 패치 정책이 적용되는 관리형 노드의 패치 상태는 해당 노드의 State Manager 연결 실행 상태와 일치합니다. 연결 실행 상태가 `Compliant`인 경우 관리형 노드의 패치 상태가 `Compliant`로 표시됩니다. 연결 실행 상태가 `Non-Compliant`인 경우 관리형 노드의 패치 상태가 `Non-Compliant`로 표시됩니다.

패치 정책 구성을 지원하는 리전

Quick Setup의 패치 정책 구성은 현재 다음 리전에서 지원됩니다.

- 미국 동부(오하이오)(us-east-2)
- 미국 동부(버지니아 북부)(us-east-1)
- 미국 서부(캘리포니아 북부) (us-west-1)
- 미국 서부(오레곤)(us-west-2)
- 아시아 태평양(뭄바이)(ap-south-1)
- 아시아 태평양(서울)(ap-northeast-2)
- 아시아 태평양(싱가포르)(ap-southeast-1)
- 아시아 태평양(시드니)(ap-southeast-2)
- 아시아 태평양(도쿄)(ap-northeast-1)
- 캐나다(중부)(ca-central-1)
- 유럽(프랑크푸르트)(eu-central-1)
- 유럽(아일랜드)(eu-west-1)
- 유럽(런던) (eu-west-2)
- 유럽(파리) (eu-west-3)
- 유럽(스톡홀름) (eu-north-1)

- 남아메리카(상파울루)(sa-east-1)

패치 정책 S3 버킷에 대한 권한

패치 정책을 생성할 때 `baseline_overrides.json`이라는 파일이 포함된 Amazon S3 버킷을 Quick Setup에서 생성합니다. 이 파일에는 패치 정책에 지정한 패치 기준선에 대한 정보가 저장됩니다.

S3 버킷 이름은 `aws-quicksetup-patchpolicy-`*`account-id`*`-`*`quick-setup-configuration-id`* 형식으로 되어 있습니다.

예: `aws-quicksetup-patchpolicy-123456789012-abcde`

조직의 패치 정책을 생성하는 경우 해당 조직의 관리 계정에 버킷이 생성됩니다.

AWS Identity and Access Management(IAM) 정책을 사용하여 이 S3 버킷에 액세스하는 권한을 다른 AWS 리소스에 제공해야 하는 두 가지 사용 사례가 있습니다.

- 사례 1: Quick Setup에서 제공되는 것이 아니라 관리형 노드가 있는 자체 인스턴스 프로파일 또는 서비스 역할을 사용합니다.
- 사례 2: VPC 엔드포인트를 사용하여 Systems Manager에 연결

둘 중 하나에 필요한 권한 정책은 아래 섹션에 있습니다(Quick Setup S3 버킷에 대한 정책 권한).

사례 1: Quick Setup에서 제공되는 것이 아니라 관리형 노드가 있는 자체 인스턴스 프로파일 또는 서비스 역할을 사용합니다.

패치 정책 구성에는 인스턴스에 연결된 기존 인스턴스 프로파일에 필수 IAM 정책 추가 옵션이 포함됩니다.

이 옵션을 선택하지 않고 이 패치 정책을 사용하여 Quick Setup을 통해 관리형 노드에 패치를 적용하려면 다음이 구현되었는지 확인해야 합니다.

- 관리형 노드에 Systems Manager 권한을 제공하는 데 사용되는 IAM 인스턴스 프로파일 또는 IAM 서비스 역할에 IAM 관리형 정책 `AmazonSSMManagedInstanceCore`가 연결되어야 합니다.
- 패치 정책 버킷에 액세스하는 권한을 IAM 인스턴스 프로파일 또는 IAM 서비스 역할에 인라인 정책으로 추가해야 합니다. 이전 코드 샘플과 같이 모든 `aws-quicksetup-patchpolicy` 버킷 또는 조직 또는 계정용으로 생성된 특정 버킷에만 와일드카드 액세스 권한을 제공할 수 있습니다.
- 다음 키-값 쌍으로 IAM 인스턴스 프로파일 또는 IAM 서비스 역할에 태그를 지정해야 합니다.

`Key: QSConfigId-quick-setup-configuration-id, Value: quick-setup-configuration-id`

*quick-setup-configuration-id*는 패치 정책 구성을 생성하는 데 사용되는 AWS CloudFormation 스택에 적용되는 파라미터의 값을 나타냅니다. 이 문제를 해결하려면 다음을 수행하세요.

1. AWS CloudFormation 콘솔(https://console.aws.amazon.com/cloudformation)을 엽니다.
2. 패치 정책을 생성하는 데 사용되는 스택의 이름을 선택합니다. 이름은 `StackSet-AWS-QuickSetup-PatchPolicy-LA-q4bkg-52cd2f06-d0f9-499e-9818-d887cEXAMPLE`과 같은 형식으로 되어 있습니다.
3. 파라미터 탭을 선택합니다.
4. 파라미터 목록의 키 열에서 QSConfigurationId라는 키를 찾습니다. 해당 행의 값 열에서 구성 ID(예: `abcde`)를 찾습니다.

   이 예제에서 인스턴스 프로파일 또는 서비스 역할에 적용할 태그의 키는 `QSConfigId-abcde`이고 값은 `abcde`입니다.

IAM 역할에 태그를 추가하는 방법에 대한 자세한 내용은 IAM 사용 설명서의 IAM 역할 태깅 및 인스턴스 프로파일의 태그 관리(AWS CLI 또는 AWS API)를 참조하세요.

사례 2: VPC 엔드포인트를 사용하여 Systems Manager에 연결

VPC 엔드포인트를 사용하여 Systems Manager에 연결하는 경우, S3용 VPC 엔드포인트 정책에서 Quick Setup 패치 정책 S3 버킷에 대한 액세스를 허용해야 합니다.

S3의 VPC 엔드포인트 정책에 권한을 추가하는 방법에 대한 자세한 내용은 Amazon S3 사용 설명서의 버킷 정책을 통해 VPC 엔드포인트의 액세스 제어를 참조하세요.

Quick Setup S3 버킷에 대한 정책 권한

모든 `aws-quicksetup-patchpolicy` 버킷 또는 조직 또는 계정용으로 생성된 특정 버킷에만 와일드카드 액세스 권한을 제공할 수 있습니다. 아래에 설명된 두 가지 사례에 필요한 권한을 제공하려면 두 형식 중 하나를 사용합니다.

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] 패치 정책 구성이 생성되면 S3 콘솔에서 버킷의 전체 이름을 찾을 수 있습니다. 예: aws-quicksetup-patchpolicy-123456789012-abcde

패치 정책 작업의 무작위 패치 기준선 ID

패치 정책의 패치 적용 작업에서는 AWS-RunPatchBaseline SSM 명령 문서의 BaselineOverride 파라미터를 활용합니다.

패치 정책 외부의 패치 적용에 AWS-RunPatchBaseline을 사용하면 BaselineOverride를 사용하여 지정된 기본값과 다른 작업 중에 사용할 패치 기준선 목록을 지정할 수 있습니다. BaselineOverride 파라미터 사용에 설명된 대로 baseline_overrides.json이라는 파일에 이 목록을 생성하고 소유하는 Amazon S3 버킷에 수동으로 추가합니다.

그러나 패치 정책에 따른 패치 적용 작업의 경우 Systems Manager에서 자동으로 S3 버킷을 생성하고 baseline_overrides.json 파일을 추가합니다. 그러면 Quick Setup에서 패치 적용(Run Command 사용) 기능을 실행할 때마다 시스템에서는 각 패치 기준선의 무작위 ID를 생성합니다. 이 ID

는 패치 정책 패치 적용 작업마다 다르며, 이 ID가 나타내는 패치 기준선은 계정에 저장되거나 액세스할 수 없습니다.

따라서 구성에서 선택한 패치 기준선의 ID는 패치 적용 로그에 표시되지 않습니다. 이는 AWS 관리형 패치 기준선과 사용자가 선택했을 수 있는 사용자 지정 패치 기준선에 모두 적용됩니다. 그 대신 로그에서 보고되는 기준선 ID는 해당 특정 패치 적용 작업용으로 생성된 ID입니다.

또한 무작위 ID로 생성된 패치 기준선에 대한 Patch Manager의 세부 정보를 보려고 시도하면 시스템에서는 패치 기준선이 존재하지 않는다고 보고합니다. 이는 예상되는 동작이며 무시해도 됩니다.

패치 정책 생성

필수 조건

다음 작업을 완료하기 전에 Quick Setup의 홈 리전이 이미 지정되어 있어야 합니다. 자세한 설명은 홈 AWS 리전 구성을 참조하세요.

패치 정책을 생성하려면 Systems Manager 콘솔에서 다음 태스크를 수행합니다.

Quick Setup을 사용하여 패치 정책을 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.

   조직에 대한 패치를 설정하는 경우, 조직의 관리 계정에 로그인해야 합니다. 위임된 관리자 계정이나 멤버 계정을 사용할 경우 정책을 설정할 수 없습니다.
2. 탐색 창에서 Quick Setup를 선택합니다.
3. Patch Manager 카드에서 Create(생성)을 선택합니다.

   > **Tip**
   >
   > 계정에 이미 하나 이상의 구성이 있으면 먼저 구성 섹션에서 라이브러리 탭 또는 생성 버튼을 선택하여 카드를 봅니다.

4. Configuration name(구성 이름)에 패치 정책을 식별하는 데 유용한 이름을 입력합니다.
5. Scanning and installation(검사 및 설치) 섹션의 Patch operation(패치 작업)에서 패치 정책이 지정된 대상을 Scan(검사)하도록 할지 아니면 지정된 대상에 대해 패치 Scan and install(검사 및 설치)을 모두 수행할지를 선택합니다.
6. Scanning schedule(검사 일정)에서 Use recommended defaults(권장 기본값 사용) 또는 Custom scan schedule(사용자 지정 검사 일정)을 선택합니다. 기본 검사 일정을 선택할 경우 매일 오전 1시(UTC)에 대상을 검사합니다.

- Custom scan schedule(사용자 지정 검사 일정)을 선택한 경우 Scanning frequency(검사 빈도)를 선택합니다.
- Daily(매일)를 선택한 경우 대상을 검사할 시간을 UTC 시간으로 입력합니다.
- Custom CRON Expression(사용자 지정 CRON 표현식)을 선택한 경우 일정을 CRON expression(CRON 표현식)으로 입력합니다. Systems Manager의 CRON 표현식 형식에 대한 자세한 내용은 참조: Systems Manager용 Cron 및 Rate 표현식 섹션을 참조하세요.

  또한 Wait to scan targets until first CRON interval(첫 번째 CRON 간격까지 대상 검사 대기)을 선택합니다. 기본적으로 Patch Manager는 노드가 대상으로 설정되는 즉시 노드를 검사합니다.

7. Scan and install(검사 및 설치)을 선택한 경우 지정된 대상에 패치를 설치할 때 사용할 Installation schedule(설치 일정)을 선택합니다. Use recommended defaults(권장 기본값 사용)을 선택하면 Patch Manager는 일요일 오전 2:00 UTC에 주간 패치를 설치합니다.

   - Custom install schedule(사용자 지정 설치 일정)을 선택한 경우 Installation frequency(설치 빈도)를 선택합니다.
   - Daily(매일)를 선택한 경우 대상에 업데이트를 설치할 시간을 UTC 시간으로 입력합니다.
   - Custom CRON Expression(사용자 지정 CRON 표현식)을 선택한 경우 일정을 CRON expression(CRON 표현식)으로 입력합니다. Systems Manager의 CRON 표현식 형식에 대한 자세한 내용은 참조: Systems Manager용 Cron 및 Rate 표현식 섹션을 참조하세요.

     또한 노드가 대상으로 설정되는 즉시 업데이트를 설치하려면 Wait to install updates until first CRON interval(첫 번째 CRON 간격까지 업데이트 설치 대기) 확인란을 선택 취소합니다. 기본적으로 Patch Manager는 첫 번째 CRON 간격까지 기다린 후에 업데이트를 설치합니다.
   - 패치 설치 후 노드를 재부팅하려면 Reboot if needed(필요한 경우 재부팅)를 선택합니다. 설치 후 재부팅하는 것이 좋지만 가용성 문제가 발생할 수 있습니다.

8. Patch baseline(패치 기준선) 섹션에서 대상을 검사하고 업데이트할 때 사용할 패치 기준선을 선택합니다.

   기본적으로 Patch Manager는 사전 정의된 패치 기준선을 사용합니다. 자세한 내용은 미리 정의된 패치 기준 정보 단원을 참조하십시오.

   Custom patch baseline(사용자 지정 패치 기준선)을 선택한 경우 사전 정의된 AWS 패치 기준선을 사용하지 않으려는 운영 체제에 대해 선택된 패치 기준선을 변경합니다.

   AWS 사전 정의된 패치 기준선을 사용하든 사용자 지정 패치 기준선을 사용하든 관계없이, Quick Setup에서 사용 가능한 패치 기준선은 선택한 홈 리전의 기준선입니다.

> **Note**
>
> VPC 엔드포인트를 사용하여 Systems Manager에 연결하는 경우, S3에 대한 VPC 엔드포인트 정책에서 이 S3 버킷에 대한 액세스를 허용하는지 확인하세요. 자세한 내용은 패치 정책 S3 버킷에 대한 권한 단원을 참조하십시오.

> **Important**
>
> Quick Setup에서 패치 정책 구성을 사용하는 경우 사용자 지정 패치 기준선에 대한 업데이트는 한 시간에 한 번씩 Quick Setup과 동기화됩니다.
> 패치 정책에서 참조된 사용자 지정 패치 기준선이 삭제되면 해당 패치 정책의 Quick Setup Configuration details(구성 세부 정보) 페이지에 배너가 표시됩니다. 배너는 패치 정책에서 더 이상 존재하지 않는 패치 기준선을 참조하고 있으며 후속 패치 작업이 실패할 것임을 알려줍니다. 이 경우 Quick Setup Configurations(구성) 페이지로 돌아가서 Patch Manager 구성을 선택하고 Actions(작업), Edit configuration(구성 편집)을 선택합니다. 삭제된 패치 기준선 이름이 강조 표시되며, 영향을 받는 운영 체제의 새 패치 기준선을 선택해야 합니다.

9. (선택 사항) Patching log storage(패치 로그 스토리지) 섹션에서 Write output to S3 bucket(S3 버킷에 출력 쓰기)을 선택하여 패치 작업 로그를 Amazon S3 버킷에 저장합니다.

> **Note**
>
> 조직의 패치 정책을 설정하는 경우, 조직의 관리 계정에 이 버킷에 대한 읽기 전용 권한 이상의 권한이 있어야 합니다. 정책에 포함된 모든 조직 단위에는 버킷에 대한 쓰기 권한이 있어야 합니다. 여러 계정에 버킷 액세스 권한을 부여하는 방법에 대한 자세한 내용은 Amazon Simple Storage Service 사용 설명서에서 예제 2: 버킷 소유자가 교차 계정 버킷 권한 부여를 참조하세요.

10. S3 찾아보기를 선택하여 패치 로그 출력을 저장할 버킷을 선택합니다. 관리 계정에 이 버킷에 대한 읽기 권한이 있어야 합니다. 로깅을 위해, Targets(대상) 섹션에 구성된 모든 비관리 계정 및 대상에는 제공된 S3 버킷에 대한 쓰기 권한이 있어야 합니다.
11. Targets(대상) 섹션에서 다음 중 하나를 선택하여 이 패치 정책 작업의 계정 및 리전을 식별합니다.

> **Note**
>
> 단일 계정에서 작업하는 경우 조직 및 조직 단위(OU) 작업을 위한 옵션을 사용할 수 없습니다. 이 구성을 계정의 모든 AWS 리전에 적용할지 아니면 선택한 리전에만 적용할지 선택할 수 있습니다.

- Entire organization(전체 조직) - 조직 내 모든 계정 및 리전
- Custom(사용자 지정) - 사용자가 지정한 OU 및 리전만
  - Target OUs(대상 OU) 섹션에서 패치 정책을 설정할 OU를 선택합니다.
  - Target Regions(대상 리전) 섹션에서 패치 정책을 적용할 리전을 선택합니다.
- Current account(현재 계정) - 현재 로그인한 계정에서 지정한 리전만 대상으로 합니다. 다음 중 하나를 선택합니다.
  - Current Region(현재 리전) - 콘솔에서 선택한 리전의 관리형 노드만 대상으로 합니다.
  - Choose Regions(리전 선택) - 패치 정책을 적용할 개별 리전을 선택합니다.

12. Choose how you want to target instances(대상 인스턴스 지정 방식 선택)에서 다음 중 하나를 선택하여 패치할 노드를 식별합니다.

- All managed nodes(모든 관리형 노드) - 선택한 OU 및 리전의 모든 관리형 노드.
- Specify the resource group(리소스 그룹 지정) - 목록에서 리소스 그룹의 이름을 선택하여 관련 리소스를 대상으로 지정합니다.

> **Note**
>
> 현재, 리소스 그룹 선택은 단일 계정 구성에 대해서만 지원됩니다. 여러 계정의 리소스를 패치하려면 다른 대상 지정 옵션을 선택하세요.

- Specify a node tag(노드 태그 지정) - 대상으로 지정한 모든 계정 및 리전에서, 사용자가 지정한 키-값 페어로 태깅된 노드만 패치됩니다.
- Manual(수동) - 지정된 모든 계정 및 리전의 관리형 노드를 목록에서 수동으로 선택합니다.

> **Note**
>
> 이 옵션은 현재 Amazon EC2 인스턴스만 지원합니다.

13. Rate control(속도 제어) 섹션에서 다음을 수행합니다.

    - Concurrency(동시성)에 패치 정책을 동시에 실행할 노드의 개수 또는 백분율을 입력합니다.
    - Error threshold(오류 임계값)에 패치 정책이 실패하기까지 오류 발생이 허용되는 노드의 개수 또는 백분율을 입력합니다.

14. (선택 사항) 인스턴스에 연결된 기존 인스턴스 프로파일에 필수 IAM 정책 추가 확인란을 선택합니다.

    이렇게 선택하면 연결된 인스턴스 프로파일(EC2 인스턴스스) 또는 연결된 서비스 역할(하이브리드 정품 인증 노드)이 이미 있는 노드에 이 Quick Setup 구성을 통해 생성된 IAM 정책이 적용됩니다. 관리형 노드에 이미 인스턴스 프로파일 또는 서비스 역할이 연결되어 있지만, Systems Manager를 사용하는 데 필요한 모든 권한이 포함되어 있지는 않은 경우 이렇게 선택하는 것이 좋습니다.

    여기서 선택한 항목은 이 패치 정책 구성이 적용되는 계정 및 리전에서 나중에 생성되는 관리형 노드에 적용됩니다.

    > **Important**
    >
    > 이 확인란을 선택하지 않고 이 패치 정책을 사용하여 Quick Setup을 통해 관리형 노드에 패치를 적용하려면 다음을 수행해야 합니다.
    > 패치 정책에 대해 생성된 S3 버킷에 액세스하는 권한을 IAM 인스턴스 프로파일 또는 IAM 서비스 역할에 추가합니다.
    > IAM 인스턴스 프로파일 또는 IAM 서비스 역할에 특정 키-값 쌍으로 태그를 지정합니다.
    > 자세한 설명은 사례 1: Quick Setup에서 제공되는 것이 아니라 관리형 노드가 있는 자체 인스턴스 프로파일 또는 서비스 역할을 사용합니다.을 참조하세요.

15. 생성(Create)을 선택합니다.

    패치 정책이 생성된 후 패치 적용 상태를 검토하려는 경우, Quick Setup 페이지에서 구성에 액세스할 수 있습니다.

DevOps Guru 구성

Quick Setup을 사용하여 DevOps Guru 옵션을 빠르게 구성할 수 있습니다. Amazon DevOps Guru는 기계 학습(ML) 기반 서비스로, 애플리케이션의 운영 성능과 가용성을 쉽게 개선할 수 있습니다. DevOps Guru는 정상적인 운영 패턴과 다른 동작을 감지하므로 고객에게 영향을 미치기 훨씬 전에 운영 문제를 식별할 수 있습니다. DevOps Guru는 AWS 애플리케이션에서 운영 데이터를 자동으로 수집

하고 운영 데이터의 문제를 시각화하는 단일 대시보드를 제공합니다. 수동 설정이나 기계 학습 전문 지식 없이도 DevOps Guru를 시작하여 애플리케이션 가용성과 안정성을 개선할 수 있습니다.

다음 AWS 리전에서 Quick Setup로 DevOps Guru 구성을 사용할 수 있습니다.

- 미국 동부(버지니아 북부)
- 미국 동부(오하이오)
- 미국 서부(오레곤)
- 유럽(프랑크푸르트)
- 유럽(아일랜드)
- 유럽(스톡홀름)
- 아시아 태평양(싱가포르)
- 아시아 태평양(시드니)
- 아시아 태평양(도쿄)

요금 정보는 Amazon DevOps Guru 요금을 참조하세요.

필수 조건

다음 작업을 완료하기 전에 Quick Setup의 홈 리전이 이미 지정되어 있어야 합니다. 자세한 설명은 홈 AWS 리전 구성을 참조하세요.

DevOps Guru를 설정하려면 AWS Systems Manager Quick Setup 콘솔에서 다음 태스크를 수행합니다.

Quick Setup으로 DevOps Guru를 설정하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Quick Setup를 선택합니다.
3. DevOps Guru 카드에서 생성을 선택합니다.

> **Tip**
>
> 계정에 이미 하나 이상의 구성이 있으면 먼저 구성 섹션에서 라이브러리 탭 또는 생성 버튼을 선택하여 카드를 봅니다.

4. [구성 옵션(Configuration options)] 섹션에서 분석하려는 AWS 리소스 유형과 알림 기본 설정을 선택합니다.

   내 조직의 모든 계정에 있는 모든 AWS 리소스 분석(Analyze all AWS resources in all the accounts in my organization) 옵션을 선택하지 않으면 DevOps Guru 콘솔에서 나중에 분석할 AWS 리소스를 선택할 수 있습니다. DevOps Guru는 2개의 요금 그룹으로 분류되는 다양한 AWS 리소스 유형(예: Amazon Simple Storage Service(Amazon S3) 버킷, Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스)을 분석합니다. 각 활성 리소스에 대해 분석된 AWS 리소스 시간에 대해 비용을 지불합니다. 리소스는 1시간 이내에 지표, 이벤트 또는 로그 항목을 생성하는 경우에만 활성화됩니다. 특정 AWS 리소스 유형에 대해 부과되는 요금은 가격 그룹에 따라 다릅니다.

   [SNS 알림 사용(Enable SNS notifications)] 옵션을 선택하면 구성에서 대상으로 하는 조직 단위(OU)의 각 AWS 계정에 Amazon Simple Notification Service(Amazon SNS) 주제가 생성됩니다. DevOps Guru는 이 주제를 사용하여 새로운 인사이트 생성과 같은 중요한 DevOps Guru 이벤트에 대해 알려줍니다. 이 옵션을 사용하지 않으면 나중에 DevOps Guru 콘솔에서 주제를 추가할 수 있습니다.

   AWS Systems Manager OpsItems 사용(Enable AWS Systems Manager OpsItems) 옵션을 선택하면 관련 Amazon EventBridge 이벤트 및 Amazon CloudWatch 경보에 대해 운영 작업 항목(OpsItems)이 생성됩니다.

5. [일정(Schedule)] 섹션에서 구성과 다른 리소스에 대한 변경 사항을 Quick Setup에서 수정할 빈도를 선택합니다. [기본값(Default)] 옵션은 한 번 실행됩니다. Quick Setup에서 구성과 다른 리소스에 대한 변경 사항을 수정하지 않도록 하려면 [사용자 정의(Custom)]에서 [사용 중지됨(Disabled)]을 선택합니다.

6. [대상(Targets)] 섹션에서 DevOps Guru가 일부 조직 단위(OU) 또는 현재 로그인한 계정의 리소스를 분석하도록 허용할지 여부를 선택합니다.

   [사용자 정의(Custom)]를 선택하는 경우 8단계로 진행합니다.

   [사용자 정의 계정(Custom account)]을 선택하는 경우 9단계로 진행합니다.

7. 대상 OU(Target OUs) 및 대상 리전(Target Regions) 섹션에서 DevOps Guru를 사용할 OU 및 리전의 확인란을 선택합니다.

8. 현재 계정에서 DevOps Guru를 사용할 리전을 선택합니다.

9. 생성(Create)을 선택합니다.

Distributor 패키지 배포

Distributor는 AWS Systems Manager의 기능입니다. Distributor 패키지는 단일 엔터티로 배포할 수 있는 설치 가능한 소프트웨어 또는 자산의 모음입니다. Quick Setup을 사용하면 Distributor 패키지를 AWS 계정 및 AWS 리전으로 배포하거나 AWS Organizations로 조직 전체에 배포할 수 있습니다. 현재는 EC2Launch v2 에이전트, Amazon Elastic File System(Amazon EFS) 유틸리티 패키지와 Amazon CloudWatch 에이전트만 Quick Setup을 통해 배포할 수 있습니다. Distributor에 대한 자세한 정보는 AWS Systems Manager Distributor 섹션을 참조하십시오.

필수 조건

다음 작업을 완료하기 전에 Quick Setup의 홈 리전이 이미 지정되어 있어야 합니다. 자세한 설명은 홈 AWS 리전 구성을 참조하세요.

Distributor 패키지를 배포하려면 AWS Systems Manager Quick Setup 콘솔에서 다음 태스크를 수행합니다.

Quick Setup으로 Distributor 패키지를 배포하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Quick Setup를 선택합니다.
3. 배포자 카드에서 생성을 선택합니다.

   > **Tip**
   >
   > 계정에 이미 하나 이상의 구성이 있으면 먼저 구성 섹션에서 라이브러리 탭 또는 생성 버튼을 선택하여 카드를 봅니다.

4. [구성 옵션(Configuration options)] 섹션에서 배포할 패키지를 선택합니다.
5. [대상(Targets)] 섹션에서 패키지를 전체 조직, 일부 조직 단위(OU) 또는 현재 로그인한 계정에 배포할지 여부를 선택합니다.

   [전체 조직(Entire organization)]을 선택하는 경우 8단계로 진행합니다.

   [사용자 정의(Custom)]를 선택하는 경우 7단계로 진행합니다.
6. [대상 리전(Target Regions)] 섹션에서 패키지를 배포할 OU 및 리전의 확인란을 선택합니다.
7. 생성(Create)을 선택합니다.

Amazon EC2 인스턴스 리소스 예약

AWS Systems Manager의 기능인 Quick Setup을 사용하면, Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스스의 시작 및 중지를 자동화하는 Resource Scheduler를 구성할 수 있습니다.

이 Quick Setup 구성을 사용하면 지정한 일정에 따라 인스턴스를 시작하고 중지하여 운영 비용을 절감할 수 있습니다. 이 기능은 필요하지 않을 때 인스턴스를 실행하여 발생하는 불필요한 비용을 방지하는 데 유용합니다. 현재 주 5일, 하루 10시간씩만 사용하는 인스턴스를 계속 실행 상태로 두는 경우를 예를 들어 보겠습니다. 이렇게 하는 대신, 매일 업무 시간 이후에 인스턴스가 중지되도록 예약할 수 있습니다. 그러면 실행 시간이 168시간에서 50시간으로 단축되므로, 해당 인스턴스의 비용을 70% 절감할 수 있습니다. Quick Setup 사용은 무료입니다. 그러나 설정한 리소스와 사용량 한도에 따라 비용이 발생할 수 있으며, 구성 설정에 사용된 서비스에 대한 수수료는 없습니다.

Resource Scheduler를 사용하면 정의한 일정에 따라 여러 AWS 리전 및 AWS 계정의 인스턴스를 자동으로 중지하고 시작하도록 선택할 수 있습니다. Quick Setup 구성은 지정한 태그 키와 값을 사용하여 Amazon EC2 인스턴스를 대상으로 적용됩니다. 구성에서 지정한 값과 일치하는 태그가 있는 인스턴스만 Resource Scheduler에 의해 중지되거나 시작됩니다.

개별 구성은 리전당 최대 5,000개의 인스턴스 일정을 지원합니다. 특정 리전에서 5,000개 이상의 인스턴스를 예약해야 하는 경우 구성을 여러 개 생성해야 합니다. 그에 따라 인스턴스를 태깅하여 각 구성별로 최대 5,000개의 인스턴스를 관리하도록 합니다. Resource Scheduler Quick Setup 구성을 여러 개 생성하는 경우 서로 다른 태그 키 값을 지정해야 합니다. 예를 들어 한 구성에는 값이 ‘Prod’인 태그 키 ‘Env’를 사용하고 다른 구성에서는 ‘Env’와 ‘Dev’를 사용할 수 있습니다.

구성을 삭제하면 더 이상 이전에 정의한 일정에 따라 인스턴스가 중지되거나 시작되지 않습니다. 드물지만 API 작업 실패로 인해 인스턴스가 정상적으로 중지되거나 시작되지 않을 수 있습니다.

Resource Scheduler는 태깅된 인스턴스가 `stopped` 상태인 경우에만 해당 인스턴스를 시작합니다. 마찬가지로, 인스턴스가 `running` 상태일 때만 중지됩니다. Resource Scheduler는 이벤트 기반 모델을 기반으로 작동하며 지정한 시간에만 인스턴스를 시작하거나 중지합니다. 예를 들어 오전 9시에 인스턴스를 시작하는 일정을 생성합니다. Resource Scheduler는 지정한 태그와 연결된 인스턴스 중 오전 9시에 `stopped` 상태인 모든 인스턴스를 시작합니다. 나중에 인스턴스를 수동으로 중지하면 Resource Scheduler가 `running` 상태를 유지하기 위해 인스턴스를 다시 시작하지 않습니다. 마찬가지로, 일정에 따라 중지된 후 인스턴스를 수동으로 시작할 경우 Resource Scheduler는 인스턴스를 다시 중지하지 않습니다.

시작 시간이 중지 시간보다 늦은 일정을 생성하면 Resource Scheduler는 인스턴스가 하룻밤 사이에 실행되는 것으로 간주합니다. 예를 들어 오후 9시에 인스턴스를 시작하고 오전 7시에 인스턴스를 중지하는 일정을 생성합니다. Resource Scheduler는 지정한 태그와 연결된 인스턴스 중 오후 9시에

stopped 상태인 모든 인스턴스를 시작하고 다음 날 오전 7시에 중지합니다. 하룻밤 일정의 경우 시작 시간은 일정에 선택한 날짜에 적용됩니다. 단, 중지 시간은 일정의 다음 날에 적용됩니다.

필수 조건

다음 작업을 완료하기 전에 Quick Setup의 홈 리전이 이미 지정되어 있어야 합니다. 자세한 설명은 홈 AWS 리전 구성을 참조하세요.

Amazon EC2 인스턴스의 일정을 설정하려면 AWS Systems Manager Quick Setup 콘솔에서 다음 태스크를 수행합니다.

Quick Setup을 사용하여 인스턴스 일정을 설정하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Quick Setup를 선택합니다.
3. Resource Scheduler 카드에서 생성을 선택합니다.

> **Tip**
>
> 계정에 이미 하나 이상의 구성이 있으면 먼저 구성 섹션에서 라이브러리 탭 또는 생성 버튼을 선택하여 카드를 봅니다.

4. Instance tag(인스턴스 태그) 섹션에서 일정과 연결하려는 인스턴스에 적용할 태그 키와 값을 지정합니다.
5. Schedule options(일정 옵션) 섹션에서 인스턴스를 시작하고 중지할 시간대, 날짜 및 시간을 지정합니다.
6. Targets(대상) 섹션에서 조직의 Custom(사용자 지정) 그룹 또는 로그인한 Current account(현재 계정) 중 무엇에 대해 호스트 관리를 설정할지 선택합니다.
   - Custom(사용자 지정) - Target OUs(대상 OU) 섹션에서 일정을 설정할 OU를 선택합니다. 다음으로 Target Regions(대상 리전) 섹션에서 일정을 설정할 리전을 선택합니다.
   - 현재 계정(Current account) - 현재 리전(Current Region) 또는 리전 선택(Choose Regions)을 선택합니다. Choose Regions(리전 선택)를 선택한 경우 일정을 설정할 Target Regions(대상 리전)를 선택합니다.
7. Summary(요약) 섹션에서 일정 정보를 확인합니다.
8. 생성(Create)을 선택합니다.

AWS 리소스 탐색기 구성

AWS Systems Manager의 기능인 Quick Setup을 사용하면 AWS 계정 또는 전체 AWS 조직의 리소스를 검색하고 발견할 수 있도록 AWS 리소스 탐색기을 빠르게 구성할 수 있습니다. 이름, 태그, ID와 같은 메타데이터를 사용하여 리소스를 검색할 수 있습니다. AWS 리소스 탐색기은 인덱스를 사용하여 검색 쿼리에 빠르게 응답합니다. 리소스 탐색기는 다양한 데이터 소스를 사용하여 사용자의 AWS 계정 리소스에 대한 정보를 수집함으로써 인덱스를 생성 및 유지합니다.

리소스 탐색기용 Quick Setup이 인덱스 구성 프로세스를 자동화합니다. AWS 리소스 탐색기에 대한 자세한 내용은 AWS 리소스 탐색기 사용 설명서의 AWS 리소스 탐색기이란 무엇인가요?를 참조하세요.

Quick Setup 중에 리소스 탐색기가 다음을 수행합니다.

- AWS 계정의 모든 AWS 리전마다 인덱스를 생성합니다.
- 계정의 애그리게이터 인덱스로 지정한 리전에 있는 인덱스를 업데이트합니다.
- 애그리게이터 인덱스 리전에 기본 뷰를 생성합니다. 이 뷰에는 필터가 없으므로 인덱스에서 찾은 모든 리소스를 반환합니다.

최소 권한

다음 절차의 단계를 수행하려면 다음 권한이 있어야 합니다.

- 작업: `resource-explorer-2:*`-리소스: 특정 리소스 없음(*)
- 작업: `iam:CreateServiceLinkedRole` - 리소스: 특정 리소스 없음(*)

리소스 탐색기 구성하기

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Quick Setup를 선택합니다.
3. 홈 리전을 선택한 다음 시작하기를 선택합니다.
4. 리소스 탐색기 카드에서 생성을 선택합니다.
5. 집계 인덱스 리전 섹션에서 집계 인덱스를 포함할 리전을 선택합니다. 사용자의 지리적 위치에 적합한 리전을 선택해야 합니다.
6. (선택 사항) 위에서 선택한 리전이외의 지역에서 기존 집계 인덱스 바꾸기 확인란을 선택합니다.
7. 대상 섹션에서 대상 조직 또는 검색하려는 리소스가 포함된 특정 조직 단위(OU)를 선택합니다.

8. 리전 섹션에서 구성에 포함할 리전을 선택합니다.
9. 구성 요약을 살펴본 후 생성을 선택합니다.

리소스 탐색기 페이지에서 구성 상태를 모니터링할 수 있습니다.

Quick Setup 결과 문제 해결

실패한 배포

생성 중 CloudFormation 스택 집합이 실패하면 배포가 실패합니다. 다음 단계에 따라 배포 실패를 조사합니다.

1. AWS CloudFormation 콘솔로 이동합니다.
2. Quick Setup 구성에 의해 생성된 스택을 선택합니다. 스택 이름(Stack name)에는 `SSMHostMgmt`와 같이 선택한 구성 유형이 뒤에 오는 `QuickSetup`이 포함됩니다.

   > **Note**
   >
   > CloudFormation이 실패한 스택 배포를 삭제하는 경우가 있습니다. 스택(Stacks) 테이블에서 스택을 사용할 수 없는 경우 필터 목록에서 삭제됨(Deleted)을 선택합니다.

3. 상태(Status)와 상태 이유(Status reason)를 봅니다. 스택 상태에 대한 자세한 내용은 AWS CloudFormation 사용 설명서의 스택 상태 코드를 참조하세요.
4. 실패한 정확한 단계를 이해하려면 이벤트(Events) 탭을 보고 각 이벤트의 상태(Status)를 검토합니다.
5. AWS CloudFormation 사용 설명서의 문제 해결을 검토합니다.
6. CloudFormation 문제 해결 단계를 사용하여 배포 실패를 해결할 수 없는 경우 구성을 삭제하고 다시 구성합니다.

실패한 연결

설정하는 동안 연결이 실패한 경우 구성의 구성 세부 정보(Configuration details) 페이지에 있는 구성 세부 정보(Configuration details) 테이블에 구성 상태(Configuration status)가 실패(Failed)로 표시됩니다. 실패한 연결 문제를 해결하려면 다음 단계를 사용하세요.

1. 구성 세부 정보(Configuration details) 테이블에서 실패한 구성을 선택한 다음 세부 정보 보기(View Details)를 선택합니다.

2. 연결 이름(Association name)을 복사합니다.
3. State Manager로 이동하고 검색 필드에 연결 이름을 붙여 넣습니다.
4. 연결을 선택하고 실행 내역(Execution history) 탭을 선택합니다.
5. 실행 ID에서 실패한 연결 실행을 선택합니다.
6. 연결 실행 대상 페이지는 연결이 실행되는 모든 노드를 나열합니다. 실행되지 않은 실행에 대한 출력 버튼을 선택합니다.
7. 출력 페이지에서 Step - Output(단계 - 출력)을 선택하여 명령 실행의 해당 단계에 대한 오류 메시지를 확인합니다. 각 단계는 다른 오류 메시지를 표시할 수 있습니다. 문제를 해결할 수 있도록 모든 단계에 대한 오류 메시지를 검토합니다.

단계 출력을 봐도 문제가 해결되지 않으면 연결을 재생성해 봅니다. 연결을 다시 생성하려면 먼저 State Manager에서 실패한 연결을 삭제합니다. 연결을 삭제한 후 구성을 편집하고 삭제한 옵션을 선택하고 업데이트(Update)를 선택합니다.

> **Note**
>
> 조직(Organization) 구성의 실패(Failed)한 연결을 조사하려면 실패한 연결이 있는 계정에 로그인하고 앞에서 설명한 다음 실패한 연결 절차를 사용해야 합니다. [연결 ID(Association ID)]는 관리 계정의 결과를 볼 때 대상 계정에 대한 하이퍼링크가 아닙니다.

드리프트 상태

구성의 세부 정보 페이지를 볼 때 각 배포의 드리프트 상태를 볼 수 있습니다. 구성 드리프트는 사용자가 Quick Setup을 통한 선택 사항과 충돌하는 서비스나 기능을 변경할 때마다 발생합니다. 초기 구성 후 연결이 변경된 경우 표에 드리프트된 항목 수를 나타내는 경고 아이콘이 표시됩니다. 아이콘 위로 마우스를 가져가면 드리프트의 원인을 확인할 수 있습니다.

State Manager에서 연결이 삭제되면 관련 배포에 드리프트 경고가 표시됩니다. 이 문제를 해결하려면 구성을 편집하고 연결이 삭제되었을 때 제거된 옵션을 선택합니다. 업데이트(Update)를 선택하고 배포가 완료될 때까지 기다립니다.

운영 관리

운영 관리는 AWS 리소스를 관리하는 데 도움이 되는 기능 모음입니다.

주제

- AWS Systems Manager Incident Manager
- AWS Systems Manager Explorer
- AWS Systems Manager OpsCenter
- Systems Manager에서 호스팅하는 Amazon CloudWatch 대시보드

AWS Systems Manager Incident Manager

AWS Systems Manager의 기능인 Incident Manager를 사용하여 AWS 호스팅 애플리케이션에서 발생하는 인시던트를 관리합니다. Incident Manager는 사용자 참여, 에스컬레이션, 실행서, 대응 계획, 채팅 채널 및 인시던트 후 분석을 결합하여 팀이 인시던트를 더 빠르게 분류하고 애플리케이션을 정상 상태로 되돌릴 수 있도록 지원합니다. Incident Manager에 대해 자세히 알아보려면 Incident Manager User Guide를 참조하세요.

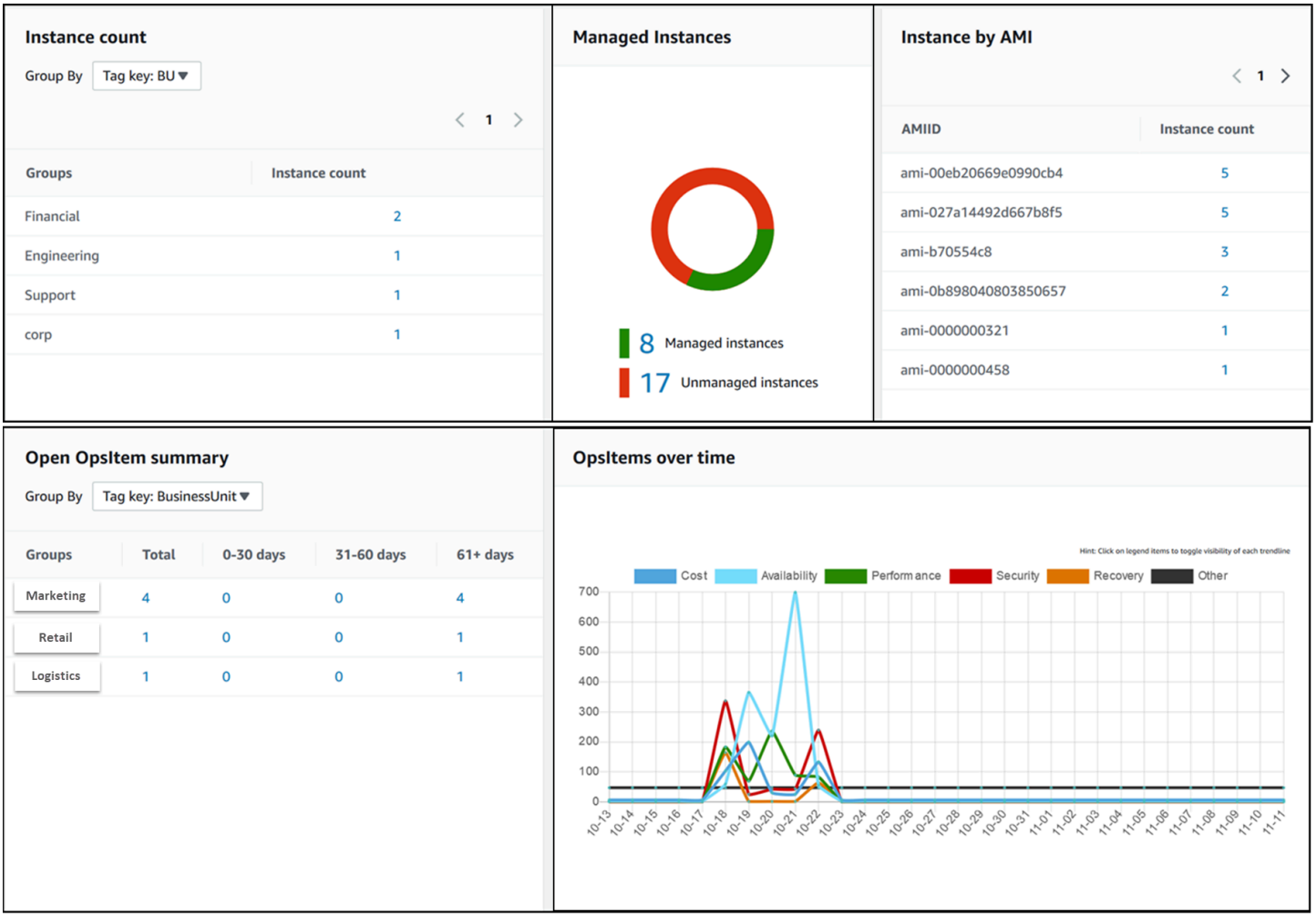
AWS Systems Manager Explorer

AWS Systems Manager Explorer는 AWS 리소스에 대한 정보를 보고하는 사용자 지정 가능한 운영 대시보드입니다. Explorer에는 AWS 계정 및 AWS 리전의 운영 데이터(OpsData)에 대한 집계 보기가 표시됩니다. Explorer에서는 하이브리드 및 멀티클라우드 환경의 관리형 노드에 대한 메타데이터가 OpsData에 포함되어 있습니다. OpsData에는 Patch Manager 패치 규정 준수 및 State Manager 연결 규정 준수 세부 정보를 비롯하여, 다른 Systems Manager 기능에서 제공하는 정보도 포함됩니다. OpsData에 액세스하는 방법을 더욱 간순화하기 위해 Explorer는 AWS Config, AWS Trusted Advisor, AWS Compute Optimizer, AWS Support(지원 사례) 등의 지원 AWS 서비스에서 제공하는 정보를 표시합니다.

운영상의 인지도를 높이기 위해 Explorer는 운영 작업 항목(OpsItems)도 표시합니다. Explorer는 OpsItems가 비즈니스 단위 또는 애플리케이션 전반에 어떻게 배포되었는지, 시간 경과에 따른 추세 및 범주별로 어떻게 다른지에 대한 컨텍스트를 제공합니다. Explorer에서 정보를 그룹화 및 필터링하여 사용자와 관련이 있고 작업이 필요한 항목에 초점을 맞출 수 있습니다. 우선순위가 높은 문제를 식별하면 Systems Manager OpsCenter를 사용하여 Automation 실행서를 실행하고 이러한 문제를 신속하게

해결할 수 있습니다. Explorer를 시작하려면 Systems Manager 콘솔을 엽니다. 탐색 창에서 Explorer를 선택합니다.

다음 이미지는 Explorer에서 사용할 수 있는 위젯이라는 개별 보고서 상자의 일부를 보여줍니다.



Explorer에는 어떤 기능이 있나요?

Explorer에는 다음 기능이 포함됩니다.

- 실행 가능한 정보의 사용자 정의 가능 표시: Explorer에는 AWS 리소스에 대한 실행 가능한 정보를 자동으로 표시하는 끌어서 놓기 위젯이 포함됩니다. Explorer는 2가지 유형의 위젯에 정보를 표시합니다.
  - 정보 위젯: 이 위젯은 Amazon EC2, Patch Manager, State Manager 및 지원하는 AWS 서비스(예: AWS Trusted Advisor, AWS Compute Optimizer 및 AWS Support)의 데이터를 요약합니다. 이러한 위젯은 AWS 리소스의 상태 및 운영 위험을 이해하는 데 도움이 되는 중요한 컨텍스트를 제공합니다. 정보 위젯의 예로 Instance count(인스턴스 수), Instance by AMI(AMI별 인스턴스), Total

noncompliant nodes(총 미준수 노드)(패치), Noncompliant associations(미준수 연결), Support Center cases(지원 센터 사례) 등이 있습니다.

  - OpsItem 위젯: Systems Manager OpsItem은 하나 이상의 AWS 리소스와 관련된 운영 작업 항목입니다. OpsItems는 Systems Manager OpsCenter의 기능입니다. OpsItems에서는 DevOps 엔지니어가 문제를 조사하고 잠재적으로 해결해야 할 수 있습니다. 가능한 OpsItems로는 높은 EC2 인스턴스 CPU 사용률, 분리된 Amazon Elastic Block Store(Amazon EBS) 볼륨, AWS CodeDeploy 배포 실패 또는 Systems Manager Automation 실행 실패 등이 있습니다. OpsItem 위젯으로는 [OpsItem 요약 열기(Open OpsItem summary)], [상태별 OpsItem(OpsItem by status)] 및 [시간 경과에 따른 OpsItems(OpsItems over time)] 등이 있습니다.

- [필터(Filters)]: 각 위젯은 AWS 계정, AWS 리전 및 태그를 토대로 정보를 필터링할 수 있는 기능을 제공합니다. 필터를 사용하면 Explorer에 표시되는 정보를 빠르게 구체화할 수 있습니다.
- [서비스 화면 직접 링크(Direct links to service screens)]: AWS 리소스의 문제를 조사하는 데 도움이 되도록 Explorer 위젯에는 관련 서비스 화면으로 연결되는 직접 링크가 포함되어 있습니다. 위젯에 적용되는 필터는 관련 서비스 화면으로 이동할 경우 계속 적용됩니다.
- [그룹(Groups)]: 조직 전반의 운영 문제의 유형을 이해하는 데 도움이 되도록 일부 위젯을 사용하면 계정, 리전 및 태그를 기준으로 데이터를 그룹화할 수 있습니다.
- 태그 키 보고: Explorer를 설정할 때 태그 키를 최대 5개까지 지정할 수 있습니다. 이 키는 Explorer에서 데이터를 그룹화하고 필터링하는 데 도움이 됩니다. 지정된 키가 OpsItem을 생성하는 리소스의 키와 일치하면 키와 값이 OpsItems에 포함됩니다.
- AWS 계정 및 AWS 리전 표시의 세 가지 모드: Explorer에는 다음과 같이 OpsData와 AWS 계정 및 AWS 리전의 OpsItems에 대한 표시 모드가 포함됩니다.
  - 단일 계정/단일 리전: 이는 기본 보기입니다. 이 모드를 사용하면 자체 계정과 현재 리전에서 데이터 및 OpsItems를 볼 수 있습니다.
  - 단일 계정/다중 리전: 이 모드에서는 Explorer 설정 페이지를 사용하여 하나 이상의 리소스 데이터 동기화를 생성해야 합니다. 리소스 데이터 동기화는 하나 이상의 리전에서 OpsData를 집계합니다. 리소스 데이터 동기화를 생성한 후 Explorer 대시보드에서 사용할 동기화를 토글할 수 있습니다. 그런 다음 리전을 기준으로 데이터를 필터링하고 그룹화할 수 있습니다.
  - [여러 계정/여러 리전(Multiple-account/multiple-Region)]: 이 모드를 사용하려면 조직이나 회사에서 [모든 기능(All features)]이 설정된 상태에서 AWS Organizations를 사용해야 합니다. 컴퓨팅 환경에서 AWS Organizations를 구성한 후에는 관리 계정의 모든 계정 데이터를 집계할 수 있습니다. 그런 다음 리전을 기준으로 데이터를 필터링 및 그룹화할 수 있도록 리소스 데이터 동기화를 생성할 수 있습니다. 조직의 [모든 기능(All features)] 모드에 대한 자세한 내용은 조직에서 모든 기능 사용을 참조하세요.

- [보고(Reporting)]: 쉼표로 구분된 값(.csv) 파일 형태의 Explorer 보고서를 Amazon Simple Storage Service(Amazon S3) 버킷으로 내보낼 수 있습니다. 내보내기가 완료되면 Amazon Simple Notification Service(Amazon SNS)에서 알림을 받습니다.

Explorer이 OpsCenter와 어떻게 관련이 있습니까?

Systems Manager OpsCenter는 운영 엔지니어와 IT 전문가가 AWS 리소스와 관련된 OpsItems를 확인 및 조사하고 문제를 해결할 수 있도록 중앙 위치를 제공합니다. Explorer는 DevOps 관리자가 AWS 리전 및 계정 전반에서 OpsItems를 포함하여 운영 데이터의 집계된 요약을 볼 수 있는 보고서 허브입니다. Explorer는 사용자가 트렌드와 패턴을 발견하고, 필요한 경우 Systems Manager Automation 실행서를 사용하여 문제를 신속하게 해결할 수 있도록 도와줍니다.

이제 OpsCenter 설정이 Explorer 설정에 통합됩니다. 이미 OpsCenter을 설정한 경우, Explorer에는 OpsItems에 대해 집계된 정보를 포함하여 작업 데이터가 자동으로 표시됩니다. OpsCenter를 설정하지 않은 경우에는 Explorer 설치를 사용하여 두 기능을 모두 시작할 수 있습니다. 자세한 내용은 Systems Manager Explorer 및 OpsCenter 시작하기 단원을 참조하십시오.

OpsData란 무엇입니까?

OpsData란 Systems Manager 대시보드에 표시되는 모든 운영 데이터입니다. Explorer는 다음 소스에서 OpsData를 검색합니다.

- Amazon Elastic Compute Cloud(Amazon EC2)

  Explorer에 표시되는 데이터는 총 노드 수, 관리형 및 비관리형 노드의 총 수, 특정 Amazon Machine Image(AMI)를 사용하는 노드 개수입니다.
- Systems Manager OpsCenter

  Explorer에 표시되는 데이터는 상태별 OpsItems 개수, 심각도별 OpsItems 개수, 여러 그룹 및 30일 기간에 걸쳐 열린 OpsItems 개수, 시간 경과에 따른 OpsItems 기록 데이터입니다.
- Systems Manager Patch Manager

  Explorer에 표시되는 데이터에는 미준수 노드와 심각한 미준수 노드의 수가 포함됩니다.
- AWS Trusted Advisor

  Explorer에 표시되는 데이터에는 비용 최적화, 보안, 내결함성, 성능 및 서비스 제한 영역에서 EC2 예약 인스턴스에 대한 모범 사례 확인 상태가 포함됩니다.
- AWS Compute Optimizer

Explorer에 표시되는 데이터에는 미달 프로비저닝되거나 과잉 프로비저닝된 EC2 인스턴스 수, 최적화 결과, 온디맨드 요금 내역, 인스턴스 유형 및 가격에 대한 권장 사항이 포함됩니다.

- AWS Support Center 사례

  Explorer에 표시되는 데이터에는 사례 ID, 심각도, 상태, 생성 시간, 제목, 서비스 및 범주가 있습니다.

- AWS Config

  Explorer에 표시되는 데이터에는 준수 및 비준수 AWS Config 규칙의 전체 요약, 준수 및 비준수 리소스 수, 각각에 대한 특정 세부 정보(비준수 규칙 또는 리소스로 드릴다운할 때)가 있습니다.

- AWS Security Hub

  Explorer에 표시되는 데이터에는 Security Hub 결과의 전체 요약, 심각도별로 그룹화된 각 결과의 수 및 결과에 대한 특정 세부 정보가 있습니다.

> **Note**
>
> Explorer에서 AWS Trusted Advisor 및 AWS Support Center 사례를 보려면 AWS Support에서 설정된 Enterprise 또는 Business 계정이 있어야 합니다.

Explorer [설정(Settings)] 페이지에서 OpsData 소스를 보고 관리할 수 있습니다. OpsData로 Explorer 위젯을 채우는 서비스 설정 및 구성에 대한 자세한 내용은 관련 서비스 설정 섹션을 참조하세요.

Explorer를 사용하는 데 비용이 듭니까?

예. 통합 설정 중에 OpsItems를 생성하기 위한 기본 규칙을 설정하면 자동으로 OpsItems를 생성하는 프로세스가 시작됩니다. 한 달에 생성된 OpsItems의 수를 기준으로 계정에 요금이 청구됩니다. 또한 한 달에 이루어진 `GetOpsItem`, `DescribeOpsItem`, `UpdateOpsItem` 및 `GetOpsSummary` API 호출 수를 기준으로 계정에 요금이 청구됩니다. 또한 관련 진단 정보를 노출하는 다른 서비스에 대한 공개 API 호출에 대해 요금이 부과될 수 있습니다. 자세한 설명은 AWS Systems Manager 요금을 참조하세요.

주제

- Systems Manager Explorer 및 OpsCenter 시작하기
- Systems Manager Explorer 탐색
- Systems Manager Explorer에서 OpsData 내보내기

- Systems Manager Explorer 문제 해결

Systems Manager Explorer 및 OpsCenter 시작하기

AWS Systems Manager는 Systems Manager Explorer 및 Systems Manager OpsCenter를 시작하는 데 도움이 되는 통합 설정 환경을 사용합니다. 이 설명서에서는 Explorer 및 OpsCenter 설정을 통합 설정이라고 부릅니다. 이미 OpsCenter를 설정한 경우, 통합 설정을 완료하여 설정 및 옵션을 확인해야 합니다. OpsCenter를 설정하지 않은 경우에는 통합 설정을 사용하여 두 기능을 모두 시작할 수 있습니다.

> **Note**
>
> 통합 설정은 Systems Manager 콘솔에서만 사용할 수 있습니다. Explorer 또는 OpsCenter를 프로그래밍 방식으로 설정할 수 없습니다.

통합 설정은 다음 태스크를 수행합니다.

- 역할 및 권한 구성: 통합 설정은 기본 규칙에 따라 Amazon EventBridge에서 OpsItems를 자동으로 생성할 수 있는 AWS Identity and Access Management(IAM) 역할을 생성합니다. 설정이 끝나면 이 섹션에 설명된 대로 OpsCenter에 대해 사용자, 그룹 또는 역할 권한을 구성해야 합니다.
- OpsItem 생성에 기본 역할 사용: 통합 설정은 EventBridge에서 기본 규칙을 생성합니다. 이러한 규칙은 이벤트에 대한 응답으로 OpsItems를 자동 생성합니다. 이러한 이벤트의 예로는 AWS 리소스의 상태 변경, 보안 설정 변경 또는 사용 불가 상태가 된 서비스가 있습니다.
- OpsData 소스 허용: 통합 설정을 사용하면 Explorer 위젯을 채우는 데이터 원본을 사용할 수 있습니다.
- 보고 태그 키를 지정하는 기능 허용: 통합 설정을 사용하면 특정 기준을 충족하는 새로운 OpsItems에 자동 할당이 되도록 최대 5개의 보고 태그 키를 지정할 수 있습니다.

통합 설정을 완료한 후에는 여러 리전 및 계정에서 데이터를 표시하도록 Explorer를 설정하는 것이 좋습니다. Explorer 및 OpsCenter는 통합 설정을 완료했을 때 사용한 AWS 계정 및 AWS 리전에 대해 OpsData 및 OpsItems를 자동으로 동기화합니다. 리소스 데이터 동기화를 생성하여 다른 계정 및 리전에서 OpsData 및 OpsItems을 집계할 수 있습니다.

> **Note**
>
> 설정 페이지에서 언제든지 설정 구성을 변경할 수 있습니다.

관련 서비스 설정

AWS Systems Manager Explorer 및 AWS Systems Manager OpsCenter은(는) 다른 AWS 서비스 및 Systems Manager 기능에서 정보를 수집하거나 이와 상호 작용합니다. 통합 설정을 사용하기 전에 이러한 다른 서비스나 기능을 설정하고 구성하는 것이 좋습니다.

다음 표에는 Explorer 및 OpsCenter이(가) 다른 AWS 서비스 및 Systems Manager 기능에서 정보를 수집하거나 이와 상호 작용할 수 있게 하는 작업이 포함되어 있습니다.

작업	정보
Systems Manager Automation에서 권한 확인	Explorer 및 OpsCenter를 사용하면 Systems Manager Automation 실행서를 사용하여 AWS 리소스 문제를 해결할 수 있습니다. 이 해결 기능을 사용하려면 Systems Manager Automation 실행서를 실행할 수 있는 권한이 있어야 합니다. 자세한 내용은 Automation 설정 단원을 참조하십시오.
Systems Manager Patch Manager 설정 및 구성	Explorer에는 패치 규정 준수에 대한 정보를 제공하는 위젯이 포함되어 있습니다. Explorer에서 이 데이터를 보려면 패치 작업을 구성해야 합니다. 자세한 내용은 AWS Systems Manager Patch Manager 단원을 참조하십시오.
Systems Manager State Manager 설정 및 구성	Explorer에는 Systems Manager State Manager 연결 규정 준수에 대한 정보를 제공하는 위젯이 포함되어 있습니다. Explorer에서 이 데이터를 보려면 State Manager를 구성해야 합니다. 자세한 내용은 AWS Systems Manager State Manager 단원을 참조하십시오.
AWS Config 구성 레코더 설정	Explorer은 AWS Config 구성 레코더가 제공하는 데이터를 사용하여 위젯에 EC2 인스턴스에 대한 정보를 채웁니다. Explorer에서 이 데이터를 보려면 AWS Config 구성 레코더를 설정합니다. 자세한 내용은 구성 레코더 관리를 참조하십시오.

작업	정보
	Note 구성 레코더를 활성화한 후에는 Systems Manager가 EC2 인스턴스에 대한 정보를 표시하는 Explorer 위젯에 데이터를 표시하는 데 최대 6시간이 걸릴 수 있습니다.
AWS Trusted Advisor 켜기	Explorer는 Trusted Advisor에서 제공한 데이터를 사용하여 비용 최적화, 보안, 내결함성, 성능 및 서비스 한도의 영역에서 Amazon EC2 예약 인스턴스에 대한 모범 사례 확인 상태를 표시합니다. Explorer에서 이 데이터를 보려면 비즈니스 또는 Enterprise Support 플랜이 있어야 합니다. 자세한 내용은 AWS Support 단원을 참조하십시오.
AWS Compute Optimizer 켜기	Explorer는 Compute Optimizer에서 제공한 데이터를 사용하여 미달 프로비저닝되거나 과다 프로비저닝된 EC2 인스턴스 수, 최적화 결과, 온디맨드 요금 내역, 인스턴스 유형 및 가격에 대한 권장 사항을 표시합니다. Explorer에서 이 데이터를 보려면 Compute Optimizer를 설정합니다. 자세한 내용은 AWS Compute Optimizer 시작하기를 참조하십시오.
AWS Security Hub 켜기	Explorer는 Security Hub에서 제공하는 데이터를 사용하여 보안 결과에 대한 정보로 위젯을 채웁니다. Explorer에서 이 데이터를 보려면 Security Hub 통합을 설정합니다. 자세한 내용은 AWS Security Hub란 무엇인가요?를 참조하세요.

Systems Manager Explorer에 대한 역할 및 권한 구성

통합 설정은 AWS Systems Manager Explorer 및 AWS Systems Manager OpsCenter에 대한 AWS Identity and Access Management(IAM) 역할을 자동으로 생성하고 구성합니다. 통합 설정을 완료한 경우에는 Explorer에 대한 역할 및 사용 권한을 구성하기 위해 추가 작업을 수행할 필요가 없습니다. 그러나 이 항목의 뒷부분에 설명된 대로 OpsCenter에 대한 권한을 구성해야 합니다.

내용

통합 설정으로 생성된 역할에 대한 정보

통합 설정은 Explorer 및 OpsCenter 작업을 위해 다음과 같은 역할을 생성 및 구성합니다.

- `AWSServiceRoleForAmazonSSM`: Systems Manager에서 관리하거나 사용하는 AWS 리소스에 대한 액세스를 제공합니다.
- `OpsItem-CWE-Role`: CloudWatch Events 및 EventBridge가 일반적인 이벤트에 대한 응답으로 OpsItems를 생성하도록 허용합니다.
- `AWSServiceRoleForAmazonSSM_AccountDiscovery`: Systems Manager가 데이터를 동기화할 때 AWS 계정 정보를 검색하기 위해 다른 AWS 서비스을(를) 호출하도록 허용합니다. 이에 대한 자세한 내용은 `AWSServiceRoleForAmazonSSM_AccountDiscovery` 역할 정보 섹션을 참조하세요.
- `AmazonSSMExplorerExport`: Explorer가 OpsData를 쉼표로 구분된 값(CSV) 파일로 내보내도록 허용합니다.

`AWSServiceRoleForAmazonSSM_AccountDiscovery` 역할 정보

AWS Organizations 및 리소스 데이터 동기화를 사용하여 여러 계정 및 리전의 데이터를 표시하도록 Explorer를 구성하면 Systems Manager가 서비스 연결 역할이 생성됩니다. Systems Manager는 이 역할을 사용하여 AWS Organizations에서의 AWS 계정에 대한 정보를 얻습니다. 역할은 다음과 같은 권한 정책을 사용합니다.

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

```
         "Action":[
            "organizations:DescribeAccount",
            "organizations:DescribeOrganization",
            "organizations:ListAccounts",
            "organizations:ListAWSServiceAccessForOrganization",
            "organizations:ListChildren",
            "organizations:ListParents"
         ],
         "Resource":"*"
      }
   ]
}
```

AWSServiceRoleForAmazonSSM_AccountDiscovery 역할에 대한 자세한 내용은 역할을 사용하여 OpsCenter 및 Explorer에 대한 AWS 계정 정보 수집 섹션을 참조하세요.

Systems Manager OpsCenter에 대한 권한 구성

통합 설정을 완료한 후에는 사용자가 OpsCenter에서 작업을 수행할 수 있도록 사용자, 그룹 또는 역할 권한을 구성해야 합니다.

시작하기 전 준비 사항

여러 계정 또는 단일 계정에서 OpsItems를 생성하고 관리하도록 OpsCenter를 구성할 수 있습니다. 여러 계정의 OpsItems를 생성하고 관리하도록 OpsCenter를 구성하는 경우 AWS Organizations 관리 계정은 다른 계정의 OpsItems를 수동으로 생성, 확인 또는 편집할 수 있습니다. 필요한 경우 Systems Manager 위임 관리자 계정을 선택하여 멤버 계정에 OpsItems를 생성하고 관리할 수도 있습니다. 그러나 단일 계정에 대해 OpsCenter를 구성하는 경우 OpsItems가 생성된 계정에서만 OpsItems를 보거나 편집할 수 있습니다. AWS 계정에서 OpsItems를 공유하거나 전송할 수 없습니다. 그러므로 AWS 워크로드를 실행하는 데 사용되는 AWS 계정에서 OpsCenter에 대한 권한을 구성하는 것이 좋습니다. 그런 다음 해당 계정에서 사용자 또는 그룹을 생성할 수 있습니다. 이러한 방식으로 여러 운영 엔지니어 또는 IT 전문가가 동일한 AWS 계정에서 OpsItems를 생성하고, 보고, 편집할 수 있습니다.

Explorer와 OpsCenter는 다음 API 작업을 사용합니다. 사용자, 그룹 또는 역할이 이러한 작업에 대한 액세스 권한을 가지고 있는 경우 Explorer 및 OpsCenter의 모든 기능을 사용할 수 있습니다. 이 단원의 뒷부분에서 설명하는 것처럼 보다 제한적인 액세스를 만들 수도 있습니다.

- CreateOpsItem
- CreateResourceDataSync
- DescribeOpsItems
- DeleteResourceDataSync

- GetOpsItem
- GetOpsSummary
- ListResourceDataSync
- UpdateOpsItem
- UpdateResourceDataSync

원하는 경우 계정, 그룹 또는 역할에 다음 인라인 정책을 추가하여 읽기 전용 권한을 지정할 수 있습니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:GetOpsItem",
        "ssm:GetOpsSummary",
        "ssm:DescribeOpsItems",
        "ssm:GetServiceSetting",
        "ssm:ListResourceDataSync"
      ],
      "Resource": "*"
    }
  ]
}
```

IAM 정책 생성 및 편집에 대한 자세한 내용은 IAM User Guide의 Creating IAM Policies를 참조하세요. 이 정책을 IAM 그룹에 할당하는 방법에 대한 자세한 내용은 IAM 그룹에 정책 연결 섹션을 참조하세요.

다음을 사용하여 권한을 생성하고 사용자, 그룹 또는 역할에 추가합니다.

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

조직에서 사용 중인 ID 애플리케이션에 따라 다음 옵션 중 하나를 선택하여 사용자 액세스를 구성할 수 있습니다.

액세스 권한을 제공하려면 사용자, 그룹 또는 역할에 권한을 추가하세요:

- AWS IAM Identity Center의 사용자 및 그룹:

  권한 세트를 생성합니다. AWS IAM Identity Center 사용 설명서의 권한 세트 생성의 지침을 따르세요.
- ID 제공자를 통해 IAM에서 관리되는 사용자:

  ID 페더레이션을 위한 역할을 생성합니다. IAM 사용 설명서의 서드 파티 자격 증명 공급자의 역할 만들기(연합)의 지침을 따르세요.
- IAM 사용자:
  - 사용자가 맡을 수 있는 역할을 생성합니다. IAM 사용 설명서에서 IAM 사용자의 역할 생성의 지침을 따르세요.
  - (권장되지 않음)정책을 사용자에게 직접 연결하거나 사용자를 사용자 그룹에 추가합니다. IAM 사용 설명서에서 사용자(콘솔)에 권한 추가의 지침을 따르세요.

태그를 사용하여 OpsItems에 대한 액세스 제한

태그를 지정하는 인라인 IAM 정책을 사용하여 OpsItems에 대한 액세스를 제한할 수도 있습니다. 다음은 Department라는 태그 키와 Finance라는 태그 값을 지정하는 예제입니다. 이 정책을 사용하는 경우 사용자는 GetOpsItem API 작업을 호출하여 이전에 키=Department 및 값=Finance로 태그를 지정한 OpsItems만 볼 수 있습니다. 다른 OpsItems는 볼 수 없습니다.

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

```
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:GetOpsItem"
             ],
      "Resource": "*"
      ,
      "Condition": { "StringEquals": { "ssm:resourceTag/Department": "Finance" } }
    }
  ]
}
```

다음은 OpsItems를 보고 업데이트하기 위한 API 작업을 지정하는 예제입니다. 또한 이 정책은 Department-Finance와 Project-Unity의 두 가지 태그 키-값 페어를 지정합니다.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "ssm:GetOpsItem",
            "ssm:UpdateOpsItem"
         ],
         "Resource":"*",
         "Condition":{
            "StringEquals":{
               "ssm:resourceTag/Department":"Finance",
               "ssm:resourceTag/Project":"Unity"
            }
         }
      }
   ]
}
```

OpsItem에 태그를 추가하는 방법에 대한 자세한 내용은 수동으로 OpsItems 만들기 섹션을 참조하세요.

기본 규칙 설정

통합 설정은 Amazon EventBridge에서 다음과 같은 기본 규칙을 자동으로 구성합니다. 이러한 규칙은 AWS Systems ManagerOpsCenter에서 OpsItems를 생성합니다. EventBridge에서 다음 이벤트에 대

해 OpsItems를 생성하고 싶지 않으면 통합 설정에서 이 옵션을 선택 취소합니다. 이를 선호하는 경우 특정 EventBridge 이벤트의 대상으로 OpsCenter를 지정할 수 있습니다. 자세한 내용은 OpsItems를 생성하도록 EventBridge 규칙 구성 단원을 참조하십시오. [설정(Settings)] 페이지에서 언제든지 기본 규칙을 해제할 수도 있습니다.

> **Important**
>
> 기본 규칙의 [범주(Category)] 및 [심각도(Severity)] 값은 편집할 수 없지만 기본 규칙에서 생성된 OpsItems에서는 이러한 값을 편집할 수 있습니다.

Rule	Category	Severity
CWE rules (11)		
SSMOpsItems-Autoscaling-instance-launch-failure	Availability	2-High
SSMOpsItems-Autoscaling-instance-termination-failure	Availability	2-High
SSMOpsItems-EBS-snapshot-copy-failed	Availability	2-High
SSMOpsItems-EBS-snapshot-creation-failed	Availability	2-High
SSMOpsItems-EBS-volume-performance-issue	Performance	3-Medium
SSMOpsItems-EC2-issue	Availability	2-High
SSMOpsItems-EC2-scheduled-change	Availability	3-Medium
SSMOpsItems-RDS-issue	Availability	2-High
SSMOpsItems-RDS-scheduled-change	Availability	3-Medium
SSMOpsItems-SSM-maintenance-window-execution-failed	Availability	3-Medium
SSMOpsItems-SSM-maintenance-window-execution-timedout	Availability	2-High

OpsData 소스 구성

통합 설정은 데이터 원본이 Explorer 위젯을 채우도록 활성화합니다.

- AWS Support 센터(이 원본을 활성화하려면 Business 또는 Enterprise Support 플랜이 있어야 합니다.)
- AWS Compute Optimizer(이 원본을 활성화하려면 Business 또는 Enterprise Support 플랜이 있어야 합니다.)
- Systems Manager State Manager 연결 규정 준수
- AWS Config 규정 준수

- Systems Manager OpsCenter
- Systems Manager Patch Manager 패치 규정 준수
- Amazon Elastic Compute Cloud(Amazon EC2)
- Systems Manager Inventory
- AWS Trusted Advisor(이 원본을 활성화하려면 Business 또는 Enterprise Support 플랜이 있어야 합니다.)
- AWS Security Hub

태그 키 지정

AWS Systems Manager Explorer를 설정할 때 최대 5개의 보고 태그 키를 지정할 수 있습니다. 이러한 태그 키는 AWS 리소스에 이미 있어야 합니다. 이들은 새로운 태그 키가 아닙니다. 시스템에 키를 추가한 후, 이러한 태그 키를 사용하여 Explorer에서 OpsItems를 필터링할 수 있습니다.

> **Note**
>
> 설정 페이지에서 보고 태그 키를 지정할 수도 있습니다.

여러 계정 및 리전에서 데이터를 표시하도록 Systems Manager Explorer 설정

AWS Systems Manager은(는) AWS Systems Manager Explorer 및 AWS Systems Manager OpsCenter을(를) 시작하는 데 도움이 되는 통합 설정 환경을 사용합니다. 통합 설정을 완료한 후 Explorer 및 OpsCenter는 자동으로 데이터를 동기화합니다. 보다 구체적으로 말하자면, 이러한 기능은 통합 설치를 완료할 때 사용한 AWS 계정 및 AWS 리전에 대해 OpsData 및 OpsItems를 동기화합니다. 다른 계정 및 리전에서 OpsData 및 OpsItems를 집계하려면 이 주제에 설명된 대로 리소스 데이터 동기화를 생성해야 합니다.

> **Note**
>
> 통합 설정에 대한 자세한 내용은 Systems Manager Explorer 및 OpsCenter 시작하기 섹션을 참조하세요.

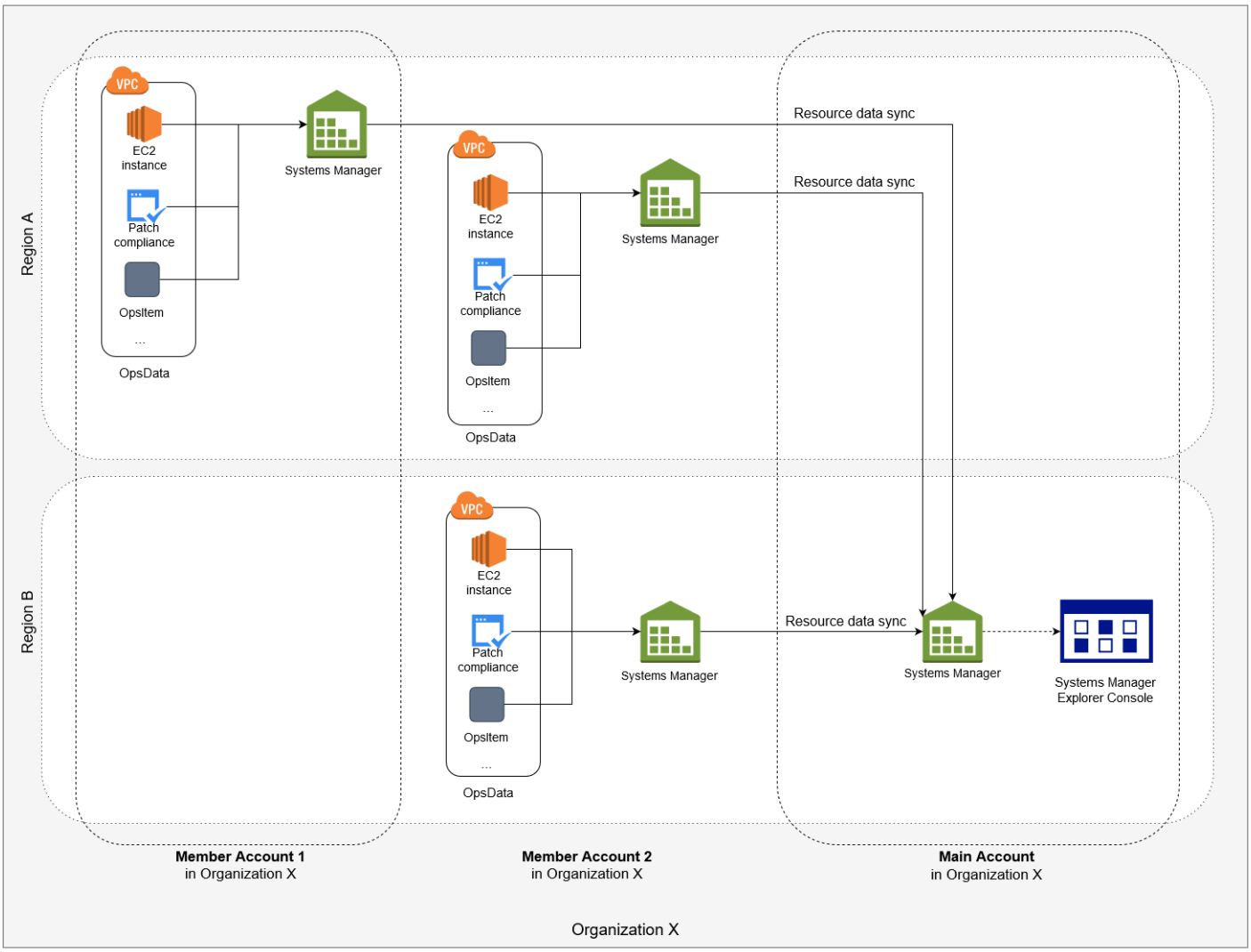
Explorer에 대한 리소스 데이터 동기화 정보

Explorer에 대한 리소스 데이터 동기화는 다음 두 가지 집계 옵션을 제공합니다.

- 단일 계정/다중 리전: 여러 AWS 리전에서 OpsItems 및 OpsData 데이터를 집계하도록 Explorer를 구성할 수 있지만 데이터 집합은 현재 AWS 계정로 제한됩니다.
- 다중 계정/다중 리전: 여러 AWS 리전 및 계정의 데이터를 집계하도록 Explorer를 구성할 수 있습니다. 이 옵션을 사용하려면 AWS Organizations를 설정하고 구성해야 합니다. AWS Organizations를 설정하고 구성한 후 조직 단위(OU) 또는 전체 조직에 대해 Explorer에서 데이터를 집계할 수 있습니다. Systems Manager는 Explorer에 표시하기 전에 데이터를 AWS Organizations 관리 계정으로 집계합니다. 자세한 내용은 AWS Organizations 사용 설명서의 AWS Organizations(이)란 무엇입니까? 섹션을 참조하세요.

> **Warning**
>
> AWS Organizations에 조직의 데이터를 집계하도록 Explorer를 구성하는 경우 시스템에서 조직의 모든 멤버 계정에 OpsData를 활성화할 수 있습니다. 모든 멤버 계정에서 OpsData 소스를 활성화하면 CreateOpsItem, GetOpsSummary와 같은 OpsCenter API 호출 횟수가 증가합니다. 이러한 API 작업 호출에 대한 요금이 부과됩니다.

다음 다이어그램은 AWS Organizations에서 작동하도록 구성된 리소스 데이터 동기화를 보여줍니다. 이 시나리오에서 사용자는 AWS Organizations에 두 개의 계정이 정의되어 있습니다. 리소스 데이터 동기화는 계정과 여러 AWS 리전를 AWS Organizations 관리 계정으로 집계한 다음 Explorer에 표시됩니다.



여러 계정 및 리전 리소스 데이터 동기화 정보

이 섹션에서는 AWS Organizations를 사용하는 여러 계정 및 여러 리전 리소스 데이터 동기화에 대한 중요한 세부 정보를 설명합니다. 특히, 이 섹션의 정보는 [리소스 데이터 동기화 생성(Create resource data sync)] 페이지에서 다음 옵션 중 하나를 선택하는 경우 적용됩니다.

- 내 AWS Organizations 구성의 모든 계정 포함
- AWS Organizations에서 조직 단위 선택

이러한 옵션 중 하나를 사용하지 않으려는 경우 이 섹션을 건너뛰어도 됩니다.

SSM 콘솔에서 리소스 데이터 동기화를 생성할 때 AWS Organizations 옵션 중 하나를 선택하는 경우에는 조직(또는 선택한 조직 단위)의 모든 AWS 계정에 대해 선택한 리전의 모든 OpsData 소스가 자동으로 Systems Manager에서 허용됩니다. 예를 들어 리전에서 Explorer를 설정하지 않은 경우에도 리소

스 데이터 동기화에 대해 AWS Organizations 옵션을 선택하면 Systems Manager가 자동으로 해당 리전에서 OpsData를 수집합니다. OpsData 소스를 허용하지 않고 리소스 데이터 동기화를 생성하려면 데이터 동기화를 생성할 때 EnableAllOpsDataSources를 false로 지정합니다. 자세한 내용은 Amazon EC2 Systems Manager API 참조의 EnableAllOpsDataSources를 참조하세요.

리소스 데이터 동기화를 위해 AWS Organizations 옵션 중 하나를 선택하지 않으면 Explorer가 데이터에 액세스할 각 계정 및 리전에서 통합 설정을 완료해야 합니다. 그렇지 않으면 Explorer는 통합 설정을 완료하지 않은 계정과 리전에 대해 OpsData 및 OpsItems을 표시하지 않습니다.

조직에 하위 계정을 추가하면 Explorer가 자동으로 해당 계정에 대한 모든 OpsData 소스를 허용합니다. 나중에 조직에서 하위 계정을 제거하면 Explorer는 계정에서 OpsData를 계속 수집합니다.

AWS Organizations 옵션 중 하나를 사용하는 기존 리소스 데이터 동기화를 업데이트하는 경우 변경의 영향을 받는 모든 계정 및 리전에 대한 모든 OpsData 소스 수집을 승인하라는 메시지가 나타납니다.

AWS 계정에 새 서비스를 추가하고 Explorer가 해당 서비스에 대한 OpsData를 수집하는 경우 Systems Manager는 해당 OpsData를 수집하도록 Explorer를 자동으로 구성합니다. 예를 들어 이전에 리소스 데이터 동기화를 생성할 때 조직에서 AWS Trusted Advisor을(를) 사용하지 않았지만 이 서비스에 등록한 경우 Explorer은(는) 이 OpsData를 수집하기 위해 리소스 데이터 동기화를 자동으로 업데이트합니다.

> **Important**
>
> 여러 계정 및 리전 리소스 데이터 동기화에 대한 다음과 같은 중요한 정보에 유의하세요.
>
> - 리소스 데이터 동기화를 삭제해도 Explorer의 OpsData 소스는 해제되지 않습니다.
> - 여러 계정에서 OpsData 및 OpsItems를 보려면 AWS Organizations [모든 기능(All features)] 모드가 설정되어 있어야 하며 AWS Organizations 관리 계정에 로그인해야 합니다.

리소스 데이터 동기화 생성

Explorer에 대한 리소스 데이터 동기화를 구성하기 전에 다음 세부 정보에 유의합니다.

- Explorer은 최대 5개의 리소스 데이터 동기화를 지원합니다.
- 리전에 대한 리소스 데이터 동기화를 생성한 후에는 해당 동기화에 대한 계정 옵션을 변경할 수 없습니다. 예를 들어 us-east-2(오하이오) 리전에서 동기화를 생성하고 현재 계정만 포함(Include only the current account) 옵션을 선택한 경우 나중에 해당 동기화를 편집하고 내 AWS Organizations 구성의 모든 계정 포함(Include all accounts from my AOlong configuration) 옵션을 선택할 수 없습니다. 대

신 첫 번째 리소스 데이터 동기화를 삭제하고 새 리소스 데이터 동기화를 생성해야 합니다. 자세한 내용은 Systems Manager Explorer 리소스 데이터 동기화 삭제 섹션을 참조하세요.

- Explorer에서 본 OpsData는 읽기 전용입니다.

다음 절차에 따라 Explorer에 대한 리소스 데이터 동기화를 생성합니다.

리소스 데이터 동기화를 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Explorer를 선택합니다.
3. 설정을 선택합니다.
4. Configure resource data sync(리소스 데이터 동기화 구성) 섹션에서 Create resource data sync(리소스 데이터 동기화 생성)를 선택합니다.
5. Resource data sync name(리소스 데이터 동기화 이름)에 이름을 입력합니다.
6. Add accounts(계정 추가) 섹션에서 옵션을 선택합니다.

> **Note**
>
> 두 AWS Organizations 옵션 중 하나를 사용하려면 AWS Organizations 관리 계정에 로그인하거나 Explorer 위임된 관리자 계정에 로그인해야 합니다. 위임된 관리자 계정에 대한 자세한 내용은 위임된 관리자 구성 섹션을 참조하세요.

7. Regions to include(포함할 리전) 섹션에서 다음 옵션 중 하나를 선택합니다.
   - [모든 현재 및 향후 리전(All current and future regions)]을 선택하여 현재 AWS 리전 모두와 향후 온라인 상태가 되는 모든 새 리전의 데이터를 자동으로 동기화합니다.
   - [모든 리전(All regions)]을 선택하여 현재 AWS 리전 모두의 데이터를 자동으로 동기화합니다.
   - 포함하고 싶은 리전을 개별적으로 선택합니다.
8. Create resource data sync(리소스 데이터 동기화 생성)를 선택합니다.

리소스 데이터 동기화를 생성한 후 시스템에서 Explorer에 데이터를 채우는 데 몇 분 정도 걸릴 수 있습니다. Explorer의 [리소스 데이터 동기화 선택(Select a resource data sync)] 목록에서 동기화를 선택하여 볼 수 있습니다.

위임된 관리자 구성

AWS Organizations에서 리소스 데이터 동기화를 사용하여 여러 AWS 리전 및 계정에서 AWS Systems Manager Explorer 데이터를 집계하는 경우에는 Explorer에 대한 위임된 관리자를 구성하는 것이 좋습니다.

위임된 관리자는 콘솔, SDK, AWS Command Line Interface(AWS CLI) 또는 AWS Tools for Windows PowerShell을 사용하여 다음과 같은 Explorer 리소스 데이터 동기화 API를 사용할 수 있습니다.

- CreateResourceDataSync
- DeleteResourceDataSync
- ListResourceDataSync
- UpdateResourceDataSync

위임된 관리자는 전체 조직 또는 조직 단위의 하위 집합에 대한 리소스 데이터 동기화를 5개까지 생성할 수 있습니다. 위임된 관리자가 만든 리소스 데이터 동기화는 위임된 관리자 계정에서만 사용할 수 있습니다. AWS Organizations 관리 계정에서는 동기화 또는 집계된 데이터를 볼 수 없습니다.

리소스 데이터 동기화에 대한 자세한 내용은 여러 계정 및 리전에서 데이터를 표시하도록 Systems Manager Explorer 설정 섹션을 참조하세요. AWS Organizations에 대한 자세한 내용은 AWS Organizations 사용 설명서의 AWS Organizations이란 무엇인가요?를 참조하세요.

주제

- Explorer 위임된 관리자 구성
- Explorer 위임된 관리자 등록 취소

Explorer 위임된 관리자 구성

Explorer 위임된 관리자를 등록하려면 다음 절차를 따르십시오.

Explorer 위임된 관리자를 등록하려면

1. AWS Organizations 관리 계정에 로그인합니다.
2. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
3. 탐색 창에서 Explorer를 선택합니다.
4. 설정을 선택합니다.

5. Explorer를 위해 위임된 관리자 섹션에서 필요한 서비스 연결 역할 및 서비스 액세스 옵션을 구성했는지 확인합니다. 필요한 경우 이러한 옵션을 구성할 [역할 생성(Create role)] 및 [액세스 사용(Enable access)] 버튼을 선택합니다.
6. [계정 ID(Account ID)]에 AWS 계정 ID를 입력합니다. 이 계정은 AWS Organizations의 멤버 계정이어야 합니다.
7. 위임된 관리자 등록을 선택합니다.

위임된 관리자가 이제 리소스 데이터 동기화 생성(Create resource data sync) 페이지의 내 AWS Organizations 구성의 모든 계정 포함(Include all accounts from my AOlong configuration) 및 AWS Organizations에서 조직 단위 선택(Select organization units in AOlong) 옵션에 액세스할 수 있습니다.

Explorer 위임된 관리자 등록 취소

Explorer 위임된 관리자의 등록을 취소하려면 다음 절차를 따르십시오. 위임된 관리자 계정은 AWS Organizations 관리 계정에서만 등록을 취소할 수 있습니다. 위임된 관리자 계정이 등록 취소되면 위임된 관리자가 만든 모든 AWS Organizations 리소스 데이터 동기화가 삭제됩니다.

Explorer 위임된 관리자의 등록을 취소하려면

1. AWS Organizations 관리 계정에 로그인합니다.
2. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
3. 탐색 창에서 Explorer를 선택합니다.
4. 설정을 선택합니다.
5. Explorer를 위해 위임된 관리자 섹션에서 등록 취소를 선택합니다. 시스템에서 경고를 표시합니다.
6. 계정 ID를 입력하고 [제거(Remove)]를 선택합니다.

계정은 AWS Organizations 리소스 데이터 동기화 API 작업에 더 이상 액세스할 수 없습니다. 계정에 의해 생성된 모든 AWS Organizations 리소스 데이터 동기화가 삭제됩니다.

Systems Manager Explorer 탐색

이 섹션에서는 위젯 레이아웃을 변경하고 대시보드에 표시되는 데이터를 변경하여 AWS Systems Manager Explorer를 사용자 지정하는 방법에 대한 정보가 포함되어 있습니다.

내용

- OpsItems에 대한 기본 규칙 편집
- Systems Manager Explorer 데이터 원본 편집
- 디스플레이 사용자 정의 및 필터 사용
- Systems Manager Explorer 리소스 데이터 동기화 삭제
- Explorer에서 AWS Security Hub의 결과 받기

OpsItems에 대한 기본 규칙 편집

통합 설정을 완료하면 Amazon EventBridge에서 12개 이상의 규칙을 사용할 수 있습니다. 이러한 규칙은 AWS Systems Manager OpsCenter에서 OpsItems를 자동으로 생성됩니다. 그러면 AWS Systems Manager Explorer이 OpsItems에 대한 집계된 정보를 표시합니다.

각 규칙에는 사전 설정된 범주 및 심각도 값이 포함됩니다. 이벤트에서 OpsItems이 생성되면 시스템에서 사전 설정된 범주 및 심각도가 자동으로 할당됩니다.

> **Important**
>
> 기본 규칙의 [범주(Category)] 및 [심각도(Severity)] 값은 편집할 수 없지만 기본 규칙에서 생성된 OpsItems에서는 이러한 값을 편집할 수 있습니다.

Rule	Category	Severity
CWE rules (11)		
SSMOpsItems-Autoscaling-instance-launch-failure	Availability	2-High
SSMOpsItems-Autoscaling-instance-termination-failure	Availability	2-High
SSMOpsItems-EBS-snapshot-copy-failed	Availability	2-High
SSMOpsItems-EBS-snapshot-creation-failed	Availability	2-High
SSMOpsItems-EBS-volume-performance-issue	Performance	3-Medium
SSMOpsItems-EC2-issue	Availability	2-High
SSMOpsItems-EC2-scheduled-change	Availability	3-Medium
SSMOpsItems-RDS-issue	Availability	2-High
SSMOpsItems-RDS-scheduled-change	Availability	3-Medium
SSMOpsItems-SSM-maintenance-window-execution-failed	Availability	3-Medium
SSMOpsItems-SSM-maintenance-window-execution-timedout	Availability	2-High

OpsItems을 생성하기 위한 기본 규칙을 편집하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Explorer를 선택합니다.
3. 설정을 선택합니다.
4. [OpsItems 규칙(OptsItems rules)] 섹션에서 [편집(Edit)]을 선택합니다.
5. CWE rules(CWE 규칙)를 확장합니다.
6. 사용하지 않을 규칙 옆의 확인란의 선택을 취소합니다.
7. 범주 및 심각도 목록을 사용하여 규칙에 대한 이 정보를 변경할 수 있습니다.
8. Save(저장)를 선택합니다.

변경 사항은 다음에 시스템에서 OpsItem을 생성할 때 적용됩니다.

Systems Manager Explorer 데이터 원본 편집

AWS Systems Manager Explorer는 다음과 같은 소스의 데이터를 표시합니다. Explorer 설정을 편집하여 데이터 원본을 추가하거나 제거할 수 있습니다.

- Amazon Elastic Compute Cloud(Amazon EC2)
- AWS Systems Manager OpsCenter
- AWS Systems Manager Patch Manager 패치 규정 준수
- AWS Systems Manager State Manager 연결 규정 준수
- AWS Trusted Advisor
- AWS Compute Optimizer
- AWS Support Center 사례
- AWS Config 규칙 및 리소스 규정 준수
- AWS Security Hub 결과

> **Note**
>
> - Explorer에서 AWS Support Center 사례를 보려면 AWS Support에서 설정된 Enterprise 또는 Business 계정이 있어야 합니다.
> - OpsCenter OpsItem 데이터 표시를 중지하도록 Explorer을 구성할 수 없습니다.

시작하기 전 준비 사항

데이터를 Explorer 위젯에 채우는 서비스를 설정하고 구성했는지 확인합니다. 자세한 내용은 관련 서비스 설정 단원을 참조하십시오.

데이터 소스를 편집하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Explorer를 선택합니다.
3. 설정을 선택합니다.
4. OpsData sources(OpsData 소스) 섹션에서 편집을 선택합니다.
5. OpsData sources(OpsData 소스)를 확장합니다.
6. 하나 이상의 소스를 추가하거나 제거합니다.
7. Save(저장)를 선택합니다.

디스플레이 사용자 정의 및 필터 사용

드래그 앤 드롭 기능을 사용하여 AWS Systems Manager Explorer에서 위젯 레이아웃을 사용자 지정할 수 있습니다. 이 항목에 설명된 대로 필터를 사용하여 Explorer에 표시되는 OpsData 및 OpsItems을 사용자 정의할 수도 있습니다.

시작하기 전 준비 사항

위젯 레이아웃을 사용자 지정하기 전에 표시하려는 위젯이 현재 Explorer에 표시되어 있는지 확인합니다. Explorer에서 일부 위젯(예: AWS Config 규정 준수 위젯)을 보려면 Configure dashboard(대시보드 구성) 페이지에서 해당 위젯을 활성화해야 합니다.

Explorer에 위젯이 표시되도록 하려면

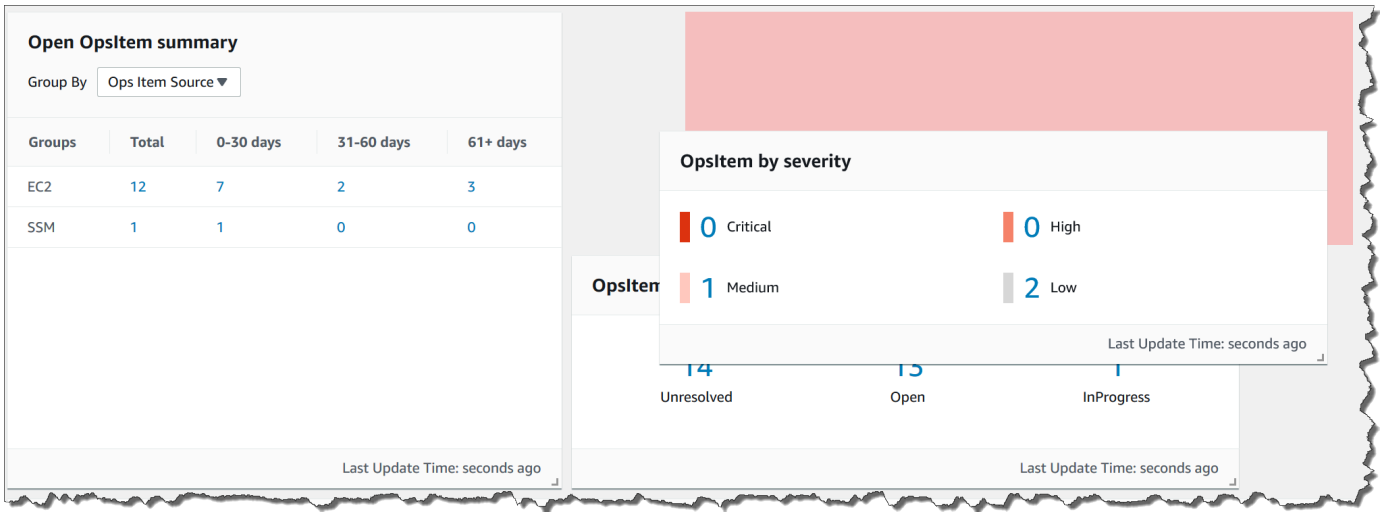
1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Explorer를 선택합니다.
3. Dashboard actions(대시보드 작업), Configure dashboard(대시보드 구성)을 차례로 선택합니다.
4. Configure Dashboard(대시보드 구성) 탭을 선택합니다.
5. Enable all(모두 활성화)를 선택하거나 개별 위젯 또는 데이터 소스를 활성화합니다.
6. Explorer를 선택하여 변경 사항을 봅니다.

위젯 레이아웃 사용자 정의

Explorer에서 위젯 레이아웃을 사용자 정의하려면 다음 절차를 따르십시오.

위젯 레이아웃을 사용자 정의하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Explorer를 선택합니다.
3. 이동할 위젯을 선택합니다.
4. 위젯의 이름을 클릭한 상태에서 새 위치로 끕니다.



5. 위치를 조정할 각 위젯에 대해 이 프로세스를 반복합니다.

새 레이아웃이 마음에 들지 않으면 Reset layout(레이아웃 재설정)을 선택하여 모든 위젯을 원래 위치로 되돌립니다.

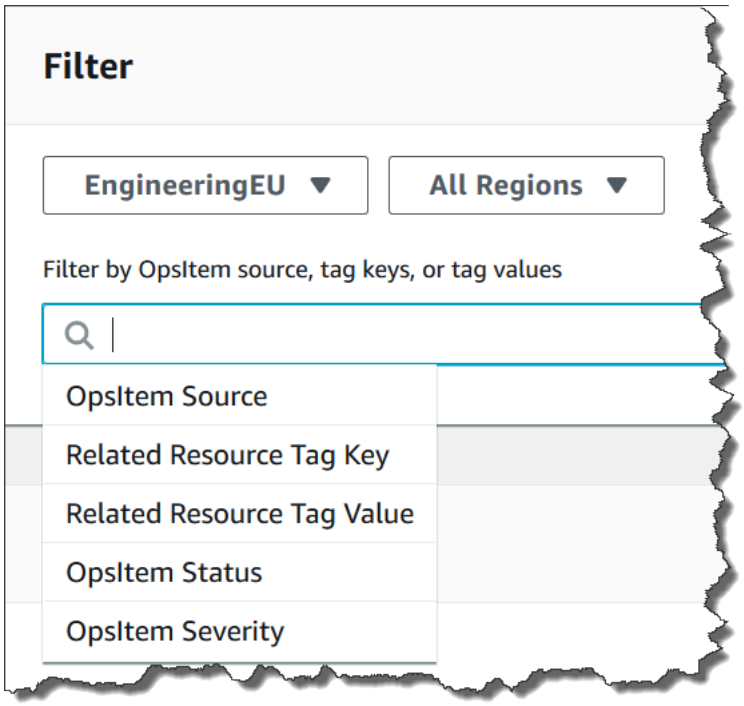
필터를 사용하여 Explorer에 표시되는 데이터 변경

기본적으로 Explorer에는 현재 AWS 계정와 현재 리전의 데이터가 표시됩니다. 하나 이상의 리소스 데이터 동기화를 생성하는 경우, 필터를 사용하여 활성 중인 동기화를 변경할 수 있습니다. 그런 다음 특정 리전 또는 모든 리전에 대한 데이터를 표시하도록 선택할 수 있습니다. 검색 창을 사용하여 다른 OpsItem 및 키-태그 기준을 필터링할 수도 있습니다.

필터를 사용하여 Explorer에 표시되는 데이터를 변경하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Explorer를 선택합니다.

3. 필터 섹션에서 Select a resource data sync(리소스 데이터 동기화 선택) 목록을 사용하여 동기화를 선택합니다.
4. [리전(Regions)] 목록을 사용하여 특정 AWS 리전을 선택하거나 [모든 리전(All Regions)]을 선택합니다.
5. 검색 창을 선택한 다음 데이터를 필터링할 기준을 선택합니다.



6. Enter를 누릅니다.

Explorer은 페이지를 닫았다가 다시 열 경우 선택한 필터 옵션을 유지합니다.

Systems Manager Explorer 리소스 데이터 동기화 삭제

AWS Systems Manager Explorer에서는 리소스 데이터 동기화를 생성하여 다른 계정 및 리전에서 OpsData 및 OpsItems을(를) 집계할 수 있습니다.

리소스 데이터 동기화에 대한 계정 옵션은 변경할 수 없습니다. 예를 들어 us-east-2(오하이오) 리전에서 동기화를 생성하고 현재 계정만 포함(Include only the current account) 옵션을 선택한 경우 나중에 해당 동기화를 편집하고 내 AWS Organizations 구성의 모든 계정 포함(Include all accounts from my AOlong configuration) 옵션을 선택할 수 없습니다. 대신에 다음 절차에 설명된 대로 리소스 데이터 동기화를 삭제하고 새 동기화를 생성해야 합니다.

리소스 데이터 동기화를 삭제하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Explorer를 선택합니다.
3. 설정을 선택합니다.
4. Configure resource data sync(리소스 데이터 동기화 구성) 섹션에서 삭제할 리소스 데이터 동기화를 선택합니다.
5. 삭제를 선택합니다.

Explorer에서 AWS Security Hub의 결과 받기

AWS Security Hub에서는 AWS의 보안 상태에 대한 포괄적인 보기가 제공됩니다. 이 서비스에서는 AWS 계정, 서비스 및 지원되는 서드 파티 제품에서 조사 결과라는 보안 데이터를 수집합니다. Security Hub 조사 결과는 보안 업계 표준 및 모범 사례를 준수하는지 확인하고, 보안 추세를 분석하고, 우선순위가 가장 높은 보안 문제를 식별하는 데 도움이 될 수 있습니다.

Security Hub에서는 이벤트 규칙을 사용하여 조사 결과를 Explorer로 보내는 Amazon EventBridge로 조사 결과를 보냅니다. 여기에 설명된 대로 통합을 활성화하면 Explorer 위젯에서 Security Hub 조사 결과를 보고 OpsCenter OpsItems에서 조사 결과 세부 정보를 볼 수 있습니다. 위젯에서는 모든 Security Hub 조사 결과의 요약이 심각도에 따라 제공됩니다. Security Hub의 새 조사 결과는 일반적으로 생성 후 몇 초 이내에 Explorer에 표시됩니다.

⚠ Warning

다음과 같은 중요한 정보를 참고합니다.

- Explorer는 Systems Manager의 기능인 OpsCenter와 통합되어 있습니다. Security Hub와 Explorer의 통합을 활성화하면 Security Hub 조사 결과에 대한 OpsItems가 OpsCenter에서 자동으로 생성됩니다. AWS 환경에 따라, 통합을 활성화하면 요금이 부과되는 다수의 OpsItems가 생성될 수 있습니다.

  계속하기 전에 Security Hub와 OpsCenter 통합에 대해 읽어보세요. 이 주제에는 조사 결과 변경 및 업데이트 방식과 계정에 OpsItems 요금이 부과되는 방식에 대한 구체적인 세부 정보가 포함되어 있습니다. 자세한 내용은 AWS Security Hub 단원을 참조하십시오. OpsCenter 요금 정보는 AWS Systems Manager 요금을 참조하세요.
- 관리자 계정에 로그인한 상태에서 Explorer에서 자원 데이터 동기화를 생성하면 동기화에 포함된 관리자 및 모든 멤버 계정에 대한 Security Hub 통합이 자동으로 활성화됩니다. 활성화

되면 비용이 부과되는 Security Hub 조사 결과에 대한 OpsItems가 OpsCenter에서 자동으로 생성됩니다. 리소스 데이터 동기화 생성에 대한 자세한 내용은 여러 계정 및 리전에서 데이터를 표시하도록 Systems Manager Explorer 설정 섹션을 참조하세요.

Explorer가 받는 결과의 유형

Explorer는 Security Hub에서 모든 결과를 받습니다. Security Hub 기본 설정을 켜면 Explorer 위젯에서 심각도에 따른 모든 조사 결과를 볼 수 있습니다. 기본적으로 Explorer에서는 심각도 심각 및 높음 조사 결과에 대한 OpsItems가 생성됩니다. 심각도 중간 및 낮음 조사 결과에 대한 OpsItems가 생성되도록 Explorer를 수동으로 구성할 수 있습니다.

Explorer에서는 정보가 제공되는 조사 결과에 대한 OpsItems가 생성되지 않지만, Security Hub 조사 결과 요약 위젯에서 정보가 제공되는 운영 데이터(OpsData)를 볼수 있습니다. 심각도와 관계없이 모든 조사 결과에 대한 OpsData가 Explorer에서 생성됩니다. Security Hub 심각도 수준에 대한 자세한 내용은 AWS Security Hub API 참조의 심각도를 참조하세요.

통합 활성화

이 섹션에서는 Security Hub 조사 결과 수신을 시작하도록 Explorer를 활성화하고 구성하는 방법을 설명합니다.

시작하기 전 준비 사항

Security Hub 결과 수신을 시작하도록 Explorer를 구성하기 전에 다음 태스크를 완료합니다.

- Security Hub를 활성화하고 구성합니다. 자세한 내용은 AWS Security Hub User Guide의 Setting up Security Hub를 참조하세요.
- AWS Organizations 관리 계정에 로그인합니다. Systems Manager는 Security Hub 결과에서 OpsItems를 생성하기 위해 AWS Organizations에 대한 액세스 권한이 필요합니다. 관리 계정에 로그인하면 다음 절차에 설명된 대로 Explorer [대시보드 구성(Configure dashboard)] 탭에서 [액세스 사용(Enable access)] 버튼을 선택하라는 메시지가 표시됩니다. AWS Organizations 관리 계정에 로그인하지 않으면 액세스를 허용할 수 없으며 Explorer는 Security Hub 결과에서 OpsItems를 생성할 수 없습니다.

Security Hub 결과 수신을 시작하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.

2. 탐색 창에서 Explorer를 선택합니다.
3. 설정을 선택합니다.
4. [대시보드 구성(Configure dashboard)] 탭을 선택합니다.
5. AWS Security Hub을 선택합니다.
6. [사용 중지됨(Disabled)] 슬라이더를 선택하여 AWS Security Hub를 켭니다.

   기본적으로 심각도 심각 및 높음 조사 결과가 표시됩니다. 심각도 중간 및 낮음 조사 결과를 표시하려면 중간, 낮음 옆에 있는 비활성화됨 슬라이더를 선택합니다.
7. [Security Hub 검색 결과에 의해 생성된 OpsItems(OpsItems created by Security Hub findings)] 섹션에서 [액세스 사용(Enable access)]을 선택합니다. 이 버튼이 보이지 않으면 AWS Organizations 관리 계정에 로그인한 후 이 페이지로 돌아가서 버튼을 선택합니다.

Security Hub의 결과를 보는 방법

다음 절차에서는 Security Hub 결과를 보는 방법을 설명합니다.

Security Hub 결과를 보는 방법

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Explorer를 선택합니다.
3. AWS Security Hub 결과 요약(ASHlong findings summary) 위젯을 찾습니다. 이 위젯은 Security Hub 결과를 표시합니다. 심각도를 선택하면 해당 OpsItem에 대한 자세한 설명을 볼 수 있습니다.

결과 수신을 중지하는 방법

다음 절차에서는 Security Hub 결과 수신을 중지하는 방법을 설명합니다.

Security Hub 결과 수신을 중지하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Explorer를 선택합니다.
3. 설정을 선택합니다.
4. [대시보드 구성(Configure dashboard)] 탭을 선택합니다.
5. [사용됨(Enabled)] 슬라이더를 선택하여 AWS Security Hub를 끕니다.

Important

콘솔에서 Security Hub 조사 결과를 비활성화하는 옵션이 회색으로 표시된 경우 AWS CLI에서 다음 명령을 실행하여 이 설정을 비활성화할 수 있습니다. AWS Organizations 관리 계정 또는 Systems Manager의 위임 관리자 계정에 로그인한 상태에서 명령을 실행해야 합니다. `region` 파라미터의 경우 AWS 리전에서 Security Hub 조사 결과 수신을 중지할 Explorer을 지정합니다.

```
aws ssm update-service-setting --setting-id /ssm/opsdata/SecurityHub --setting-
value Disabled --region AWS ##
```

다음은 그 예입니다.

```
aws ssm update-service-setting --setting-id /ssm/opsdata/SecurityHub --setting-
value Disabled --region us-east-1
```

Systems Manager Explorer에서 OpsData 내보내기

AWS Systems Manager Explorer에서 쉼표로 구분된 값(.csv) 파일로 OpsData 항목 5,000개를 Amazon Simple Storage Service(S3) 버킷으로 내보낼 수 있습니다. Explorer에서는 AWS-ExportOpsDataToS3 자동화 런북을 사용하여 OpsData를 내보냅니다. OpsData를 내보내면 내보낼 assumeRole, Amazon S3 버킷 이름, SNS 주제 ARN, 필드와 같은 세부 정보를 지정할 수 있는 자동화 런북 페이지가 시스템에 표시됩니다.

OpsData를 내보내는 방법

- 1단계: SNS 주제 지정
- 2단계: (선택 사항) 데이터 내보내기 구성
- 3단계: OpsData 내보내기

1단계: SNS 주제 지정

데이터 내보내기를 구성할 때 데이터를 내보내려는 동일한 AWS 리전에 있는 Amazon Simple Notification Service(Amazon SNS) 주제를 지정해야 합니다. 내보내기가 완료되면 Systems Manager

가 Amazon SNS 주제에 알림을 보냅니다. Amazon SNS 주제 생성에 대한 내용은 Amazon SNS 주제 생성을 참조하세요.

2단계: (선택 사항) 데이터 내보내기 구성

설정 또는 S3 버킷으로 운영 데이터 내보내기 페이지에서 데이터 내보내기 설정을 구성할 수 있습니다.

Explorer에서 데이터 내보내기를 구성하는 방법

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Explorer를 선택합니다.
3. 설정을 선택합니다.
4. Configure data export(데이터 내보내기 구성) 섹션에서 편집을 선택합니다.
5. 데이터 내보내기 파일을 기존 Amazon S3 버킷에 업로드하려면 기존 S3 버킷 선택을 선택하고 목록에서 버킷을 선택합니다.

   데이터 내보내기 파일을 새 Amazon S3 버킷에 업로드하려면 새 S3 버킷 생성을 선택하고 새 버킷에 사용할 이름을 입력합니다.

   > **Note**
   >
   > Amazon S3 버킷 이름과 Amazon SNS 주제 ARN은 Explorer에서 해당 설정을 처음으로 구성한 페이지에서만 편집할 수 있습니다. 설정 페이지에서 Amazon S3 버킷과 Amazon SNS 주제 ARN을 설정하는 경우에는 설정 페이지에서만 해당 설정을 수정할 수 있습니다.

6. Amazon SNS 주제 ARN 선택의 경우 내보내기가 완료되면 알려줄 주제를 선택합니다.
7. 생성(Create)을 선택합니다.

3단계: OpsData 내보내기

Explorer 데이터를 내보낼 때 이름이 `AmazonSSMExplorerExportRole`인 AWS Identity and Access Management(IAM) 역할이 Systems Manager에서 생성됩니다. 이 역할은 다음과 같은 IAM 정책을 사용합니다.

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

이 역할에는 다음과 같은 신뢰할 수 있는 엔터티가 포함됩니다.

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

Explorer에서 OpsData를 내보내려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.

2. 탐색 창에서 Explorer를 선택합니다.
3. 테이블 내보내기를 선택합니다.

> **Note**
>
> OpsData를 처음 내보내면 시스템에서 내보내기에 대한 수임 역할이 생성됩니다. 기본 수임 역할은 수정할 수 없습니다.

4. Amazon S3 버킷 이름의 경우 기존 버킷을 선택합니다. 필요한 경우 생성을 선택하여 Amazon S3 버킷을 생성합니다. S3 버킷 이름을 변경할 수 없다면 설정 페이지에서 버킷 이름을 구성한 것입니다. 설정 페이지에서만 버킷 이름을 변경할 수 있습니다.

> **Note**
>
> Amazon S3 버킷 이름과 Amazon SNS 주제 ARN은 Explorer에서 해당 설정을 처음으로 구성한 페이지에서만 편집할 수 있습니다.

5. SNS 주제 ARN의 경우 다운로드가 완료되면 알려줄 기존 Amazon SNS 주제 ARN을 선택합니다.

   Amazon SNS 주제 ARN을 변경할 수 없다면 설정 페이지에서 Amazon SNS 주제 ARN을 구성한 것입니다. 설정 페이지에서만 주제 ARN을 변경할 수 있습니다.

6. (선택 사항) SNS 성공 메시지의 경우 내보내기가 완료되면 표시할 성공 메시지를 지정합니다.
7. 제출을 선택합니다. 이전 페이지로 이동되고 내보내기 프로세스 상태를 보려면 클릭이라는 메시지가 시스템에 표시됩니다. 세부 정보 보기입니다.

   세부 정보 보기를 선택하여 Systems Manager Automation에서 런북의 상태와 진행률을 볼 수 있습니다.

이제 지정한 Amazon S3 버킷으로 Explorer에서 OpsData를 내보낼 수 있습니다.

이 절차를 사용하여 데이터를 내보낼 수 없는 경우 사용자, 그룹 또는 역할에 `iam:CreatePolicyVersion` 및 `iam:DeletePolicyVersion` 작업이 포함되어 있는지 확인합니다. 사용자, 그룹 또는 역할에 이러한 작업 추가에 대한 자세한 내용은 IAM 사용 설명서의 IAM 정책 편집을 참조하세요.

Systems Manager Explorer 문제 해결

이 항목에는 AWS Systems Manager Explorer에서 일반적인 문제를 해결하는 방법에 대한 정보가 포함되어 있습니다.

설정(Settings) 페이지에 태그를 업데이트한 후 Explorer에서 AWS 리소스를 필터링할 수 없음

Explorer에서 태그 키 또는 기타 데이터 설정을 업데이트하면 변경 사항에 따라 데이터를 동기화하는 데 최대 6시간이 걸릴 수 있습니다.

리소스 데이터 동기화 생성(Create resource data sync) 페이지의 AWS Organizations 옵션이 회색으로 표시됩니다.

내 AWS Organizations 구성의 모든 계정 포함하기 및 리소스 데이터 싱크 생성 페이지의 AWS Organizations에서 조직 단위 선택 옵션은 AWS Organizations을 설정하고 구성했을 경우에만 사용 가능합니다. AWS Organizations를 설정 및 구성한 경우 AWS Organizations 관리 계정 또는 Explorer 위임된 관리자가 이러한 옵션을 사용하는 리소스 데이터 동기화를 생성할 수 있습니다.

자세한 내용은 여러 계정 및 리전에서 데이터를 표시하도록 Systems Manager Explorer 설정 및 위임된 관리자 구성 섹션을 참조하세요.

Explorer는 데이터를 전혀 표시하지 않음

- Explorer에서 데이터를 액세스 및 표시하고 싶은 각 계정 및 리전에서 통합 설정이 완료되었는지 확인합니다. 그렇지 않으면 Explorer는 통합 설정을 완료하지 않은 계정과 리전에 대해 OpsData 및 OpsItems을 표시하지 않습니다. 자세한 내용은 Systems Manager Explorer 및 OpsCenter 시작하기 단원을 참조하십시오.
- Explorer을 사용하여 여러 계정 및 리전의 데이터를 보는 경우 AWS Organizations 관리 계정에 로그인했는지 확인합니다. 여러 계정 및 리전에서 OpsData 및 OpsItems을 보려면 이 계정에 로그인해야 합니다.

Amazon EC2 인스턴스에 대한 위젯은 데이터를 표시하지 않음

[인스턴스 수(Instance count)], [관리형 인스턴스(Managed instances)] 및 [AMI별 인스턴스(Instance by AMI)] 위젯과 같은 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에 대한 위젯에 데이터가 표시되지 않는 경우에는 다음을 확인합니다.

- 몇 분을 기다렸는지 확인합니다. 통합 설정을 완료한 후 OpsData가 Explorer에 표시되는 데 몇 분 정도 걸릴 수 있습니다.

- AWS Config 구성 레코더를 구성했는지 확인합니다. Explorer에서는 AWS Config 구성 레코더가 제공하는 데이터를 사용하여 위젯에 EC2 인스턴스에 대한 정보를 채웁니다. 자세한 내용은 구성 레코더 관리를 참조하십시오.
- [설정(Settings)] 페이지에서 [Amazon EC2] OpsData 소스가 활성화되어 있는지 확인합니다. 또한 구성 레코더를 활성화했거나 인스턴스를 변경한 후 6시간 이상 경과했는지 확인합니다. 처음에 구성 레코더를 활성화하거나 인스턴스를 변경한 후 Systems Manager가 Explorer EC2 위젯에 AWS Config의 데이터를 표시하는 데 최대 6시간이 걸릴 수 있습니다.
- Explorer은 인스턴스가 중지 또는 종료된 경우 24시간 후에 해당 인스턴스 표시를 중지한다는 점에 유의하십시오.
- Amazon EC2 인스턴스를 구성한 올바른 AWS 리전에 있는지 확인합니다. Explorer는 온프레미스 인스턴스에 대한 데이터를 표시하지 않습니다.
- 여러 계정 및 리전에 대해 리소스 데이터 동기화를 구성한 경우 Organizations 관리 계정에 로그인했는지 확인합니다.

패치 위젯이 데이터를 표시하지 않음

패치를 위한 규정 미준수 인스턴스 위젯에는 규정을 준수하지 않는 패치 인스턴스에 대한 데이터만 표시됩니다. 인스턴스가 규정을 준수하는 경우 이 위젯에는 데이터가 표시되지 않습니다. 규정 미준수 인스턴스가 있는 것으로 의심되는 경우 Systems Manager 패치 작업을 설정 및 구성했는지 확인하고 AWS Systems Manager Patch Manager를 사용하여 패치 규정 준수 여부를 확인합니다. 자세한 내용은 AWS Systems Manager Patch Manager 단원을 참조하십시오.

기타 문제

Explorer에서는 OpsItems을 편집하거나 수정이 불가: 계정 또는 리전에서 확인한 OpsItems는 읽기 전용입니다. 홈 계정 또는 리전에서만 업데이트 및 문제 해결이 가능합니다.

AWS Systems Manager OpsCenter

AWS Systems Manager의 기능인 OpsCenter는 운영 엔지니어 및 IT 전문가가 AWS 리소스와 관련된 운영 작업 항목(OpsItems)을 관리할 수 있는 중앙 위치를 제공합니다. OpsItem은 조사와 수정이 필요한 모든 운영 문제 또는 중단입니다. OpsCenter를 사용하면 관련 OpsItems와 관련 리소스를 포함하여 각 OpsItem에 대한 컨텍스트 조사 데이터를 볼 수 있습니다. 또한 Systems Manager Automation 런북을 실행하여 OpsItems를 해결할 수 있습니다.

각 OpsItem에는 OpsItem을 생성한 AWS 리소스의 이름과 ID를 비롯하여 이벤트를 해결하는 데 필요한 관련 정보가 포함되어 있습니다. OpsCenter를 설정하고 다른 AWS 서비스와 통합하면 OpsItems가

자동으로 생성될 수 있습니다. 이러한 서비스와 통합된 경우 OpsCenter는 OpsItem을 조사하는 데 도움이 되도록 AWS Config, AWS CloudTrail 및 Amazon EventBridge의 정보를 표시합니다. 따라서 조사를 위해 여러 콘솔 페이지를 탐색할 필요가 없습니다.

OpsCenter를 사용하여 Systems Manager용으로 구성된 온프레미스 관리형 노드의 문제를 조사하고 해결할 수 있습니다. Systems Manager에 대한 온프레미스 서버 및 가상 머신 설정과 구성에 대한 자세한 내용은 하이브리드 및 멀티클라우드 환경에서 Systems Manager 사용하기 섹션을 참조하세요.

선택한 Systems Manager 콘솔, AWS Command Line Interface(AWS CLI), AWS Tools for PowerShell 또는 AWS SDK를 사용하여 OpsCenter로 작업할 수 있습니다. AWS Identity and Access Management(IAM) 정책을 사용하여 OpsItems를 생성, 표시, 나열 및 업데이트할 수 있는 조직의 멤버를 결정할 수 있습니다. OpsItems에 태그를 할당한 다음 태그를 기반으로 사용자 및 그룹에 대한 액세스 권한을 부여하는 IAM 정책을 만들 수 있습니다.

> **Note**
>
> OpsCenter를 사용하는 데 비용이 듭니다. 자세한 내용은 AWS Systems Manager 요금을 참조하십시오.
> Amazon Web Services 일반 참조의 Systems Manager 서비스 할당량에서 모든 Systems Manager 기능의 할당량을 볼 수 있습니다. 다르게 표시되지 않는 한 리전별로 각 할당량이 적용됩니다.

OpsCenter 워크플로

OpsCenter를 설정하고 작업하여 OpsItems를 수정하려면 다음 단계를 수행하세요.

1. OpsCenter 설정 여러 계정의 OpsItems를 중앙에서 관리하도록 OpsCenter를 설정할 수도 있습니다.
2. OpsCenter와 기타 AWS 서비스를 통합합니다. OpsCenter는 Amazon CloudWatch, Amazon CloudWatch Application Insights, Amazon EventBridge, Amazon DevOps Guru, AWS Config, AWS Security Hub 및 AWS Systems Manager Incident Manager와 통합할 수 있습니다.
3. OpsItems를 생성합니다. OpsItems를 자동 및 수동으로 생성할 수 있습니다.
4. 관련 리소스, 관련 OpsItems 및 운영 데이터에 대한 컨텍스트를 추가하고 중복 OpsItems를 제거하여 OpsItems를 관리합니다.
5. Systems Manager Automation 런북을 사용하여 OpsItems를 수정합니다.

OpsCenter 설정

AWS Systems Manager는 통합 설정 환경을 사용하여 Systems Manager의 기능인 OpsCenter 및 Explorer를 시작할 수 있도록 지원합니다. Explorer는 AWS 리소스에 대한 정보를 보고하는 사용자 지정 가능한 작업 대시보드입니다. 이 설명서에서는 Explorer 및 OpsCenter 설정을 통합 설정이라고 합니다.

Explorer로 OpsCenter를 설정하려면 통합 설정을 사용해야 합니다. 통합 설정은 AWS Systems Manager 콘솔에서만 사용할 수 있습니다. Explorer 또는 OpsCenter를 프로그래밍 방식으로 설정할 수 없습니다. 자세한 내용은 Systems Manager Explorer 및 OpsCenter 시작하기 단원을 참조하십시오.

설치 시 활성화되는 기본 규칙

OpsCenter을(를) 설정하면 OpsItems을(를) 자동으로 생성하는 Amazon EventBridge에서 기본 규칙을 활성화합니다. 다음 테이블에는 자동으로 OpsItems가 생성되는 기본 EventBridge 규칙이 설명되어 있습니다. OpsItem 규칙에 따라 OpsCenter 설정 페이지의 EventBridge 규칙을 비활성화할 수 있습니다.

> **Important**
>
> 기본 규칙에 따라 생성된 OpsItems에 대해 계정에 요금이 부과됩니다. 자세한 내용은 AWS Systems Manager 요금을 참조하세요.

규칙 이름	설명
SSMOpsItems-Autoscaling-instance-launch-failure	EC2 Auto Scaling 인스턴스 시작에 실패하면 이 규칙을 통해 OpsItems이(가) 생성됩니다.
SSMOpsItems-Autoscaling-instance-termination-failure	EC2 Auto Scaling 인스턴스 종료에 실패하면 이 규칙을 통해 OpsItems이(가) 생성됩니다.
SSMOpsItems-EBS-snapshot-copy-failed	시스템에서 Amazon Elastic Block Store(Amazon EBS) 스냅샷을 복사하지 못하면 이 규칙을 통해 OpsItems이(가) 생성됩니다.
SSMOpsItems-EBS-snapshot-creation-failed	시스템에서 Amazon EBS 스냅샷을 생성하지 못하면 이 규칙을 통해 OpsItems이(가) 생성됩니다.

규칙 이름	설명
SSMOpsItems-EBS-volume-performance-issue	이 규칙은 AWS Health 추적 규칙에 해당합니다. 규칙은 Amazon EBS 볼륨(건강 이벤트 = `AWS_EBS_DEGRADED_EBS_VOLUME_PERFORMANCE`)에 성능 문제가 있을 때마다 OpsItems을(를) 생성합니다.
SSMOpsItems-EC2-issue	이 규칙은 AWS 서비스 또는 리소스에 영향을 미치는 예상치 못한 이벤트에 대한 AWS Health 추적 규칙에 해당합니다. 예를 들어 규칙은 서비스가 서비스 성능 저하를 유발하는 운영 문제에 대한 통신을 보내거나 현지화된 리소스 수준 문제에 대한 인식을 높이기 위해 통신을 보낼 때 OpsItems을(를) 생성합니다. 예를 들어 이 규칙은 `AWS_EC2_OPERATIONAL_ISSUE` 이벤트에 대한 OpsItem을(를) 생성합니다.
SSMOpsItems-EC2-scheduled-change	이 규칙은 AWS Health 추적 규칙에 해당합니다. AWS은(는) 재부팅, 중단 또는 인스턴트 시작 등 인스턴스에 대한 이벤트를 예약할 수 있습니다. 이 규칙은 EC2 예약 이벤트에 대한 OpsItems을(를) 생성합니다. 예약 이벤트에 대한 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스를 위한 예약 이벤트를 참조하세요.

규칙 이름	설명
SSMOpsItems-RDS-issue	이 규칙은 AWS 서비스 또는 리소스에 영향을 미치는 예상치 못한 이벤트에 대한 AWS Health 추적 규칙에 해당합니다. 예를 들어 규칙은 서비스가 서비스 성능 저하를 유발하는 운영 문제에 대한 통신을 보내거나 현지화된 리소스 수준 문제에 대한 인식을 높이기 위해 통신을 보낼 때 OpsItems을(를) 생성합니다. 예를 들어 이 규칙은 `AWS_RDS_MYSQL_DATABASE_CRASHING_REPEATEDLY` , `AWS_RDS_EXPORT_TASK_FAILED` 및 `AWS_RDS_CONNECTIVITY_ISSUE` 이벤트에 대한 OpsItem을(를) 생성합니다.
SSMOpsItems-RDS-scheduled-change	이 규칙은 AWS Health 추적 규칙에 해당합니다. 이 규칙은 Amazon RDS 예약 이벤트에 대한 OpsItems을(를) 생성합니다. 예정된 이벤트는 Amazon RDS 리소스의 예정된 변경 사항에 대한 정보를 제공합니다. 일부 이벤트에서는 서비스 중단을 방지하기 위한 조치를 취하라고 권장할 수 있습니다. 사용자 측의 조치는 필요하지 않은 다른 이벤트는 자동으로 발생합니다. 예약된 변경 활동 중에는 리소스를 일시적으로 사용할 수 없을 수도 있습니다. 예를 들어 이 규칙은 `AWS_RDS_SYSTEM_UPGRADE_SCHEDULED` 및 `AWS_RDS_MAINTENANCE_SCHEDULED` 이벤트에 대한 OpsItem을(를) 생성합니다. 예약된 이벤트에 대한 자세한 내용은 AWS Health 사용 설명서의 이벤트 유형 카테고리를 참조하세요.
SSMOpsItems-SSM-maintenance-window-execution-failed	시스템 유지보수 유지 기간 처리에 실패하면 이 규칙을 통해 OpsItems이(가) 생성됩니다.
SSMOpsItems-SSM-maintenance-window-execution-timedout	Systems Manager 유지 기간 처리에 실패하면 이 규칙을 통해 OpsItems가 생성됩니다.

OpsCenter 설정

다음 절차에 따라 OpsCenter을(를) 설정합니다.

OpsCenter 설정

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 OpsCenter를 선택합니다.
3. OpsCenter 홈 페이지에서 시작하기를 선택합니다.
4. OpsCenter 설정 페이지에서 Explorer에서 AWS Config 및 Amazon CloudWatch 이벤트를 구성하여 일반적으로 사용되는 규칙 및 이벤트에 따라 자동으로 OpsItems가 생성되도록 이 옵션 활성화를 선택합니다. 이 옵션을 선택하지 않으면 OpsCenter는 비활성화된 상태로 유지됩니다.

   > **Note**
   >
   > Amazon EventBridge(이전 Amazon CloudWatch Events)는 CloudWatch Events의 모든 기능과 사용자 지정 이벤트 버스, 타사 이벤트 소스, 스키마 레지스트리 등의 몇 가지 새로운 기능을 제공합니다.

5. 활성화OpsCenter를 선택합니다.

OpsCenter를 활성화하면 설정에서 다음을 수행할 수 있습니다.

- CloudWatch 콘솔 열기 버튼을 사용하여 CloudWatch 경보를 생성합니다. 자세한 내용은 OpsItems를 생성하도록 CloudWatch 경보 구성 단원을 참조하십시오.
- 운영 인사이트 활성화. 자세한 내용은 운영 인사이트를 분석하여 OpsItems 줄이기 단원을 참조하십시오.
- AWS Security Hub 결과 경보를 활성화합니다. 자세한 내용은 AWS Security Hub 단원을 참조하십시오.

목차

(선택 사항) 여러 계정에서 OpsItems를 중앙 집중식으로 관리하도록 OpsCenter 설정

Systems Manager OpsCenter를 사용하여 선택된 AWS 리전에 있는 여러 AWS 계정의 OpsItems를 중앙에서 관리할 수 있습니다. 이 특성은 AWS Organizations에서 조직을 설정한 후에 사용할 수 있습니다. AWS Organizations는 본인이 생성하고 중앙에서 관리하는 조직에 여러 AWS 계정을 통합할 수 있는 계정 관리 서비스입니다. AWS Organizations에는 비즈니스의 예산, 보안 및 규정 준수 필요성을 더 잘 충족할 수 있는 계정 관리 및 통합 청구 기능이 포함되어 있습니다. 자세한 내용은 AWS Organizations 사용 설명서의 AWS Organizations란 무엇인가요? 섹션을 참조하세요.

AWS Organizations 관리 계정에 속하는 사용자는 Systems Manager의 위임된 관리자 계정을 설정할 수 있습니다. OpsCenter 컨텍스트에서, 위임된 관리자는 멤버 계정의 OpsItems를 생성하고, 편집하고, 볼 수 있습니다. 또한 위임된 관리자는 Systems Manager Automation 런북을 사용하여 OpsItems를 대량으로 해결하거나 OpsItems가 생성되고 있는 AWS 리소스와 관련된 문제를 해결할 수 있습니다.

> **Note**
>
> Systems Manager의 위임된 관리자로 하나의 계정만 할당할 수 있습니다. 자세한 내용은 Systems Manager에 대한 AWS Organizations 위임 관리자 생성 단원을 참조하십시오.

Systems Manager에서는 여러 AWS 계정의 OpsItems를 중앙에서 관리하도록 OpsCenter를 설정하는 다음과 같은 방법이 제공됩니다.

- 빠른 설정: Systems Manager의 기능인 빠른 설정을 통해 Systems Manager 기능의 설정 및 구성 작업이 간소화됩니다. 자세한 내용은 AWS Systems Manager Quick Setup 단원을 참조하십시오.

  OpsCenter 빠른 설정은 여러 계정의 OpsItems를 관리하는 다음과 같은 작업을 완료하는 데 도움이 됩니다.

  - 위임된 관리자로 계정 등록(위임된 관리자가 아직 지정되지 않은 경우)
  - 필수 AWS Identity and Access Management(IAM) 정책 및 역할 생성
  - 위임된 관리자가 여러 계정의 OpsItems를 관리할 수 있는 AWS Organizations 조직 또는 OU(조직 단위) 지정

  자세한 내용은 (선택 사항) Quick Setup을 사용하여 여러 계정의 OpsItems를 관리하도록 OpsCenter 구성 단원을 참조하십시오.

> **Note**
>
> 현재 Systems Manager를 사용할 수 있는 모든 AWS 리전에서 빠른 설정을 사용할 수 있는 것은 아닙니다. 빠른 설정을 사용하여 여러 계정의 OpsItems를 중앙에서 관리하도록 OpsCenter를 구성하려는 리전에서 빠른 설정을 사용할 수 없는 경우에는 수동 방법을 사용해야 합니다. 빠른 설정을 사용할 수 있는 AWS 리전 목록은 AWS 리전에서의 Quick Setup 가용성에서 확인하세요.

- 수동 설정: 여러 계정의 OpsItems를 중앙에서 관리하도록 OpsCenter를 구성하려는 리전에서 빠른 설정을 사용할 수 없는 경우에는 수동 절차를 사용하여 그렇게 구성할 수 있습니다. 자세한 내용은 (선택 사항) 여러 계정에서 OpsItems를 중앙 집중식으로 관리하도록 OpsCenter 설정 단원을 참조하십시오.

(선택 사항) Quick Setup을 사용하여 여러 계정의 OpsItems를 관리하도록 OpsCenter 구성

AWS Systems Manager의 기능인 Quick Setup을 통해 Systems Manager 기능의 설정 및 구성 작업이 간소화됩니다. OpsCenter용 Quick Setup은 여러 계정의 OpsItems를 관리하는 다음과 같은 작업을 완료하는 데 도움이 됩니다.

- 위임된 관리자 계정 지정
- 필수 AWS Identity and Access Management(IAM) 정책 및 역할 생성
- 위임된 관리자가 여러 계정의 OpsItems를 관리할 수 있는 AWS Organizations 조직 또는 멤버 계정 하위 집합 지정

빠른 설정을 사용하여 여러 계정의 OpsItems를 관리하도록 OpsCenter를 구성하면 지정한 계정에 다음과 같은 리소스가 Quick Setup을 통해 생성됩니다. 이러한 리소스에서는 지정한 계정에 OpsItems로 작업하고 자동화 런북을 사용하여 OpsItems 생성 AWS 리소스 관련 문제를 해결하는 권한을 부여합니다.

리소스	계정
`AWSServiceRoleForAmazonSSM_AccountDiscovery` AWS Identity and Access Management(IAM) 서비스 연결 역할	AWS Organizations 관리 계정 및 위임된 관리자 계정

리소스	계정
이에 대한 자세한 내용은 역할을 사용하여 OpsCenter 및 Explorer에 대한 AWS 계정 정보 수집 섹션을 참조하세요.	
`OpsItem-CrossAccountManagementRole` IAM 역할 `AWS-SystemsManager-AutomationAdministrationRole` IAM 역할	위임된 관리자 계정
`OpsItem-CrossAccountExecutionRole` IAM 역할 `AWS-SystemsManager-AutomationExecutionRole` IAM 역할 기본 OpsItem 그룹(`OpsItemGroup`)에 대한 `AWS::SSM::ResourcePolicy` Systems Manager 리소스 정책	모든 AWS Organizations 멤버 계정

> **Note**
>
> 이전에 수동 방법을 사용하여 여러 계정의 OpsItems를 관리하도록 OpsCenter를 구성했다면 해당 프로세스의 4단계와 5단계에서 생성한 AWS CloudFormation 스택 또는 스택 세트를 삭제해야 합니다. 다음 절차를 완료할 때 계정에 해당 리소스가 있으면 Quick Setup에서 크로스 계정 OpsItem 관리가 올바르게 구성되지 않습니다.

빠른 설정을 사용하여 여러 계정의 OpsItems를 관리하도록 OpsCenter를 구성하는 방법

1. AWS Organizations 관리 계정을 사용하여 AWS Management Console에 로그인합니다.
2. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
3. 탐색 창에서 Quick Setup를 선택합니다.
4. 라이브러리 탭을 선택합니다.
5. 아래쪽으로 스크롤하여 OpsCenter 구성 타일을 찾습니다. 생성(Create)을 선택합니다.

6. Quick Setup OpsCenter 페이지의 위임된 관리자 섹션에서 계정 ID를 입력합니다. 이 필드를 편집할 수 없다면 위임된 관리자 계정이 Systems Manager에 이미 지정된 것입니다.
7. [대상(Targets)] 섹션에서 옵션을 선택합니다. 사용자 지정을 선택하는 경우에는 여러 계정의 OpsItems를 관리할 OU(조직 단위)를 선택합니다.
8. 생성(Create)을 선택합니다.

Quick Setup을 통해 OpsCenter 구성이 생성되고 필수 AWS 리소스가 지정된 OU에 배포됩니다.

> **Note**
>
> 여러 계정의 OpsItems를 관리하지 않으려면 Quick Setup에서 구성을 삭제할 수 있습니다. 구성을 삭제하면 구성이 원래 배포되었을 때 생성된 다음과 같은 IAM 정책과 역할이 Quick Setup을 통해 삭제됩니다.
>
> - 위임된 관리자 계정의 `OpsItem-CrossAccountManagementRole`
> - Organizations 멤버 계정의 `OpsItem-CrossAccountExecutionRole` 및 `SSM::ResourcePolicy`
>
> Quick Setup을 통해 구성이 원래 배포되었던 모든 조직 단위와 AWS 리전에서 구성이 제거됩니다.

OpsCenter의 Quick Setup 구성 문제 해결

이 섹션에는 Quick Setup을 사용하여 크로스 계정 OpsItem 관리를 구성할 때 문제를 해결하는 데 도움이 되는 정보가 포함되어 있습니다.

주제

- 이러한 StackSets에 대한 배포 실패: delegatedAdmin
- Quick Setup 구성 상태가 실패로 표시되는 경우

이러한 StackSets에 대한 배포 실패: delegatedAdmin

OpsCenter 구성을 생성할 때 Quick Setup을 통해 Organizations 관리 계정에 두 개의 AWS CloudFormation 스택 세트가 배포됩니다. 스택 세트에서는 접두사로 `AWS-QuickSetup-SSMOpsCenter`를 사용합니다. Quick Setup에서 `Deployment to these StackSets failed: delegatedAdmin` 오류가 표시되면 다음 절차를 사용하여 이 문제를 해결합니다.

StackSets failed:delegatedAdmin 오류 해결 방법

1. Quick Setup 콘솔의 빨간색 배너에 `Deployment to these StackSets failed: delegatedAdmin` 오류가 표시되면 위임된 관리자 계정 및 Quick Setup 홈 리전으로 지정된 AWS 리전에 로그인합니다.
2. AWS CloudFormation 콘솔(https://console.aws.amazon.com/cloudformation)을 엽니다.
3. Quick Setup 구성에 의해 생성된 스택을 선택합니다. 스택 이름에 AWS-QuickSetup-SSMOpsCenter가 포함되어 있습니다.

   > **Note**
   >
   > 실패한 스택 배포가 CloudFormation에서 삭제되는 경우가 있습니다. 스택(Stacks) 테이블에서 스택을 사용할 수 없는 경우 필터 목록에서 삭제됨(Deleted)을 선택합니다.

4. 상태(Status)와 상태 이유(Status reason)를 봅니다. 스택 상태에 대한 자세한 내용은 AWS CloudFormation 사용 설명서의 스택 상태 코드를 참조하세요.
5. 실패한 정확한 단계를 이해하려면 이벤트(Events) 탭을 보고 각 이벤트의 상태(Status)를 검토합니다. 자세한 내용은 AWS CloudFormation 사용 설명서의 문제 해결을 참조하세요.

> **Note**
>
> CloudFormation 문제 해결 단계를 사용하여 배포 실패를 해결할 수 없으면 구성을 삭제하고 다시 시도하세요.

Quick Setup 구성 상태가 실패로 표시되는 경우

구성 세부 정보 페이지의 구성 세부 정보 테이블에 구성 상태가 `Failed`로 표시되면 AWS 계정 및 실패한 리전에 로그인합니다.

Quick Setup의 OpsCenter 구성 생성 실패를 해결하는 방법

1. AWS 계정 및 실패가 발생한 AWS 리전에 로그인합니다.
2. AWS CloudFormation 콘솔(https://console.aws.amazon.com/cloudformation)을 엽니다.
3. Quick Setup 구성에 의해 생성된 스택을 선택합니다. 스택 이름에 AWS-QuickSetup-SSMOpsCenter가 포함되어 있습니다.

> **Note**
>
> 실패한 스택 배포가 CloudFormation에서 삭제되는 경우가 있습니다. 스택(Stacks) 테이블에서 스택을 사용할 수 없는 경우 필터 목록에서 삭제됨(Deleted)을 선택합니다.

4. 상태(Status)와 상태 이유(Status reason)를 봅니다. 스택 상태에 대한 자세한 내용은 AWS CloudFormation 사용 설명서의 스택 상태 코드를 참조하세요.
5. 실패한 정확한 단계를 이해하려면 이벤트(Events) 탭을 보고 각 이벤트의 상태(Status)를 검토합니다. 자세한 내용은 AWS CloudFormation 사용 설명서의 문제 해결을 참조하세요.

멤버 계정 구성이 ResourcePolicyLimitExceededException으로 표시되는 경우

스택 상태가 `ResourcePolicyLimitExceededException`으로 표시되면 계정이 이전에 수동 방법을 통해 OpsCenter 크로스 계정 관리에 온보딩된 것입니다. 이 문제를 해결하려면 수동 온보딩 프로세스의 4단계와 5단계에서 생성된 AWS CloudFormation 스택 또는 스택 세트를 삭제해야 합니다. 자세한 내용은 AWS CloudFormation 사용 설명서의 스택 세트 삭제와 AWS CloudFormation 콘솔에서 스택 삭제를 참조하세요.

(선택 사항) 여러 계정에서 OpsItems를 중앙 집중식으로 관리하도록 OpsCenter 설정

이 섹션에서는 크로스 계정 OpsItem 관리용 OpsCenter를 수동으로 구성하는 방법을 설명합니다. 이 프로세스는 아직 지원되지만, Systems Manager Quick Setup을 사용하는 새로운 프로세스로 대체되었습니다. 자세한 내용은 (선택 사항) Quick Setup을 사용하여 여러 계정의 OpsItems를 관리하도록 OpsCenter 구성 단원을 참조하십시오.

중앙 계정을 설정하여 멤버 계정에 대한 수동 OpsItems를 생성하고 해당 OpsItems를 관리하고 수정할 수 있습니다. 중앙 계정은 AWS Organizations 관리 계정이거나 AWS Organizations 관리 계정이면서 Systems Manager 위임 관리자 계정일 수 있습니다. Systems Manager 위임 관리자 계정을 중앙 계정으로 사용하는 것이 좋습니다. 이 기능은 AWS Organizations를 구성한 후에만 사용할 수 있습니다.

AWS Organizations를 사용하면 여러 AWS 계정을 중앙에서 생성하고 관리하는 조직으로 통합할 수 있습니다. 중앙 계정 사용자는 선택된 모든 멤버 계정의 OpsItems를 동시에 생성하고 해당 OpsItems를 관리할 수 있습니다.

이 섹션의 프로세스에 따라 Organizations에서 Systems Manager 서비스 보안 주체를 활성화하고 여러 계정에 걸쳐 OpsItems 작업을 수행하기 위한 AWS Identity and Access Management(IAM) 권한을 구성합니다.

주제

- 시작하기 전 준비 사항
- 1단계: 리소스 데이터 동기화 생성
- 2단계: AWS Organizations에서 Systems Manager 서비스 보안 주체 활성화
- 3단계: AWSServiceRoleForAmazonSSM_AccountDiscovery 서비스 연결 역할 생성
- 4단계: 여러 계정에 걸쳐 OpsItems 작업을 수행하도록 권한 구성
- 5단계: 여러 계정에 걸쳐 관련 리소스 작업을 수행하도록 권한 구성

> **Note**
>
> 여러 계정에 걸쳐 OpsCenter에서 작업을 수행할 때는 `/aws/issue` 유형의 OpsItems만 지원됩니다.

시작하기 전 준비 사항

여러 계정에서 OpsItems와 함께 작동하도록 OpsCenter를 설정하기 전에 다음을 설정했는지 확인하세요.

- Systems Manager 위임 관리자 계정. 자세한 내용은 위임된 관리자 구성 단원을 참조하십시오.
- 조직에서 설정 및 구성된 하나의 조직. 자세한 내용은 AWS Organizations 사용 설명서에서 조직 생성 및 관리를 참조하세요.
- 여러 AWS 리전 및 AWS 계정에서 자동화 런북이 실행되도록 Systems Manager Automation을 구성하셨습니다. 자세한 내용은 여러 AWS 리전 및 계정에서 자동화 실행 단원을 참조하십시오.

1단계: 리소스 데이터 동기화 생성

AWS Organizations를 설정하고 구성한 후 리소스 데이터 동기화를 생성하여 전체 조직에 대해 OpsCenter의 OpsItems을 집계할 수 있습니다. 자세한 내용은 리소스 데이터 동기화 생성 단원을 참조하십시오. 동기화를 생성할 때 계정 추가 섹션에서 내 AWS Organizations 구성의 모든 계정 포함 옵션을 선택해야 합니다.

2단계: AWS Organizations에서 Systems Manager 서비스 보안 주체 활성화

사용자가 여러 계정에 걸쳐 OpsItems 작업을 수행할 수 있게 하려면 AWS Organizations에서 Systems Manager 서비스 주체를 활성화해야 합니다. 이전에 다른 기능을 사용하여 다중 계정 시나리오를 지원하도록 Systems Manager를 구성한 경우, Systems Manager 서비스 주체가 이미 Organizations에 구성되어 있을 수 있습니다. AWS Command Line Interface(AWS CLI)에서 다음 명령을 실행하여 확인

합니다. 다른 다중 계정 시나리오를 지원하도록 Systems Manager를 구성하지 않은 경우 다음 절차인 AWS Organizations에서 Systems Manager 서비스 보안 주체 활성화로 건너뜁니다.

AWS Organizations에서 Systems Manager 서비스 주체가 활성화되었는지 확인하려면

1. AWS CLI의 최신 버전을 로컬 시스템에 다운로드합니다.
2. AWS CLI를 열고 다음 명령을 실행하여 보안 인증 정보 및 AWS 리전을 지정합니다.

```
aws configure
```

시스템에서 다음을 지정하라는 메시지를 표시합니다. 다음 예제에서는 자신의 정보로 각각의 *### ## ## ###*를 바꿉니다.

```
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. 다음 명령을 실행하여 AWS Organizations에 대해 Systems Manager 서비스 주체가 활성화되었는지 확인합니다.

```
aws organizations list-aws-service-access-for-organization
```

이 명령은 다음 예와 유사한 정보를 반환합니다.

```
{
    "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"
        }
    ]
```

```
}
```

AWS Organizations에서 Systems Manager 서비스 주체를 활성화하려면

Organizations의 Systems Manager 서비스 주체를 구성하지 않은 경우에는 아래의 단계를 수행하여 Systems Manager 서비스 주체를 구성합니다. 이 명령에 대한 자세한 내용은 AWS CLI 명령 참조에서 enable-aws-service-access를 참조하세요.

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다. 자세한 내용은 CLI 설치와 CLI 구성을 참조하세요.
2. AWS CLI의 최신 버전을 로컬 시스템에 다운로드합니다.
3. AWS CLI를 열고 다음 명령을 실행하여 보안 인증 정보 및 AWS 리전을 지정합니다.

   ```
   aws configure
   ```

   시스템에서 다음을 지정하라는 메시지를 표시합니다. 다음 예제에서는 자신의 정보로 각각의 *### ## ## ###*를 바꿉니다.

   ```
   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. 다음 명령을 실행하여 AWS Organizations에 대해 Systems Manager 서비스 주체를 활성화합니다.

   ```
   aws organizations enable-aws-service-access --service-principal "ssm.amazonaws.com"
   ```

3단계: **AWSServiceRoleForAmazonSSM_AccountDiscovery** 서비스 연결 역할 생성

AWSServiceRoleForAmazonSSM_AccountDiscovery 역할과 같은 서비스 연결 역할은 Systems Manager와 같은 AWS 서비스에 직접 연결된 고유한 유형의 IAM 역할입니다. 서비스 연결 역할은 해당 서비스에서 사전 정의하며 서비스에서 다른 AWS 서비스를 자동으로 호출하기 위해 필요한 모든 권한을 포함합니다. AWSServiceRoleForAmazonSSM_AccountDiscovery 서비스 연결 역할에 대한 자세한 정보는 Systems Manager 계정 검색을 위한 서비스 연결 역할 권한 섹션을 참조하십시오.

다음 절차에 따라 AWS CLI를 사용하여 AWSServiceRoleForAmazonSSM_AccountDiscovery 서비스 연결 역할을 생성합니다. 이 절차에서 사용되는 명령에 대한 자세한 내용은 AWS CLI 명령 참조에서 create-service-linked-role을 참조하세요.

AWSServiceRoleForAmazonSSM_AccountDiscovery 서비스 연결 역할을 생성하려면

1. AWS Organizations 관리 계정에 로그인합니다.
2. Organizations 관리 계정에 로그인한 상태에서 다음 명령을 실행합니다.

```
aws iam create-service-linked-role \
    --aws-service-name accountdiscovery.ssm.amazonaws.com \
    --description "Systems Manager account discovery for AWS Organizations service-
linked role"
```

4단계: 여러 계정에 걸쳐 OpsItems 작업을 수행하도록 권한 구성

AWS CloudFormation 스택 세트를 사용하여 여러 계정에 걸쳐 OpsItems 작업을 수행할 권한을 사용자에게 부여하는 OpsItemGroup 리소스 정책과 IAM 실행 역할을 생성합니다. 시작하려면 OpsCenterCrossAccountMembers.zip 파일을 다운로드하여 압축 해제합니다. 이 파일에는 OpsCenterCrossAccountMembers.yaml AWS CloudFormation 템플릿 파일이 들어 있습니다. 이 템플릿을 사용하여 스택 세트를 생성하면 CloudFormation이 계정에 OpsItemCrossAccountResourcePolicy 리소스 정책과 OpsItemCrossAccountExecutionRole 실행 역할을 자동으로 생성합니다. 스택 세트 생성에 대한 자세한 내용은 AWS CloudFormation 사용 설명서에서 스택 세트 생성을 참조하세요.

Important

이 태스크에 대한 다음 중요 정보를 기록해 둡니다.

- AWS Organizations 관리 계정에 로그인한 상태에서 스택 세트를 배포해야 합니다.
- 위임된 관리자 계정을 포함하여 여러 계정에서 OpsItems 작업을 수행하기 위한 대상으로 지정하려는 모든 계정에 로그인한 상태에서 이 절차를 반복해야 합니다.
- 다른 AWS 리전에서 계정 간 OpsItems 관리를 활성화하려면, 이 템플릿의 Specify regions(리전 지정) 섹션에서 Add all regions(모든 리전 추가)를 선택합니다. 옵트인 리전에는 계정 간 OpsItem 관리가 지원되지 않습니다.

5단계: 여러 계정에 걸쳐 관련 리소스 작업을 수행하도록 권한 구성

OpsItem에 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스, Amazon Simple Storage Service(S3) 버킷 등 영향을 받는 리소스에 대한 자세한 정보가 포함될 수 있습니다. 이전 4단계에서 생성한 `OpsItemCrossAccountExecutionRole` 실행 역할은 멤버 계정으로 관련 리소스를 볼 수 있는 읽기 전용 권한을 OpsCenter에 제공합니다. 또한 IAM 역할을 생성하여 관련 리소스를 보고 사용할 수 있는 권한을 관리 계정에 제공해야 하며, 이 단계은 이 태스크에서 수행합니다.

시작하려면 `OpsCenterCrossAccountManagementRole.zip` 파일을 다운로드하여 압축 해제합니다. 이 파일에는 `OpsCenterCrossAccountManagementRole.yaml` AWS CloudFormation 템플릿 파일이 들어 있습니다. 이 템플릿을 사용하여 스택을 생성하면 CloudFormation이 계정에 `OpsCenterCrossAccountManagementRole` IAM 역할을 자동으로 생성합니다. 스택 생성에 대한 자세한 내용은 AWS CloudFormation 사용 설명서의 AWS CloudFormation 콘솔에서 스택 생성을 참조하세요.

> **Important**
>
> 이 태스크에 대한 다음 중요 정보를 기록해 둡니다.
>
> - 계정을 OpsCenter의 위임된 관리자로 지정하려면 스택을 생헝할 때 해당 AWS 계정을 지정해야 합니다.
> - AWS Organizations 관리 계정에 로그인한 상태에서 이 절차를 수행하고 위임된 관리자 계정에 로그인한 상태에서 다시 수행해야 합니다.

(선택 사항) OpsItems에 대한 알림을 수신하도록 Amazon SNS 설정

시스템에서 OpsItem을 생성하거나 기존 OpsItem을 업데이트할 때 Amazon Simple Notification Service(Amazon SNS) 주제에 알림을 보내도록 OpsCenter를 구성할 수 있습니다.

OpsItems에 대한 알림을 수신하려면 다음 단계를 수행하세요.

- 1단계: Amazon SNS 주제 생성 및 구독
- 2단계: Amazon SNS 액세스 정책 업데이트
- 3단계: AWS KMS 액세스 정책 업데이트

> **Note**
>
> 2단계에서 AWS Key Management Service(AWS KMS) 서버 측 암호화를 켜면 3단계를 수행해야 합니다. 그렇지 않으면 3단계로 건너뛰어도 됩니다.

- 4단계: 새 OpsItems에 대한 알림을 보내도록 기본 OpsItems 규칙 켜기

1단계: Amazon SNS 주제 생성 및 구독

알림을 수신하려면 Amazon SNS 주제를 생성하고 구독해야 합니다. 자세한 내용은 Amazon Simple Notification Service 개발자 가이드의 Amazon SNS 주제 생성 및 Amazon SNS 주제 구독을 참조하세요.

> **Note**
>
> 여러 AWS 리전 또는 계정에서 OpsCenter를 사용하는 경우 OpsItem 알림을 받을 각 리전 또는 계정에서 Amazon SNS 주제를 생성하여 구독해야 합니다.

2단계: Amazon SNS 액세스 정책 업데이트

Amazon SNS 주제를 OpsItems와 연결해야 합니다. 1단계에서 생성한 Amazon SNS 주제에 OpsItems 알림을 Systems Manager에서 게시할 수 있도록 Amazon SNS 액세스 정책을 설정합니다.

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/sns/v3/home에서 Amazon SNS 콘솔을 엽니다.
2. 탐색 창에서 주제를 선택합니다.
3. 1단계에서 생성한 주제를 선택한 다음 편집을 선택합니다.
4. 액세스 정책(Access policy)를 확장합니다.
5. 기존 정책에 다음 `Sid` 블록을 추가합니다. 각 *`example resource placeholder`*를 사용자의 정보로 바꿉니다.

```
{
      "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 전역 조건 키로 혼동된 대리자 시나리오를 방지합니다. 이 조건 키를 사용하려면 값을 OpsItem 소유자의 계정 ID로 설정합니다. 자세한 내용은 IAM 사용 설명서의 혼동된 대리자를 참조하세요.

6. Save changes(변경 사항 저장)를 선택합니다.

이제 시스템에서는 OpsItems가 생성되거나 업데이트될 때 Amazon SNS 주제로 알림을 보냅니다.

> **Important**
>
> 2단계에서 AWS Key Management Service(AWS KMS) 서버 측 암호화 키로 Amazon SNS 주제를 구성하는 경우에는 3단계를 완료합니다. 그렇지 않으면 3단계로 건너뛰어도 됩니다.

3단계: AWS KMS 액세스 정책 업데이트

Amazon SNS 주제에 대해 AWS KMS 서버 측 암호화를 설정한 경우 주제를 구성할 때 선택한 AWS KMS key의 액세스 정책도 업데이트해야 합니다. 다음 절차에 따라 Systems Manager가 1단계에서 생성한 Amazon SNS 주제에 OpsItem 알림을 게시할 수 있도록 액세스 정책을 업데이트합니다.

> **Note**
>
> OpsCenter에서는 AWS 관리형 키로 구성된 Amazon SNS 주제에 대한 OpsItems 게시를 지원하지 않습니다.

1. AWS KMS 콘솔(https://console.aws.amazon.com/kms)을 엽니다.

2. AWS 리전을 변경하려면 페이지의 오른쪽 상단 모서리에 있는 리전 선택기를 사용합니다.
3. 탐색 창에서 고객 관리형 키를 선택합니다.
4. 주제를 생성할 때 선택한 KMS 키의 ID를 선택합니다.
5. 키 정책(Key policy) 섹션에서 정책 보기로 전환(Switch to policy view)을 선택합니다.
6. 편집을 선택합니다.
7. 기존 정책에 다음 `Sid` 블록을 추가합니다. 각 *`example resource placeholder`*를 사용자의 정보로 바꿉니다.

```
{
      "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"
    }
```

다음 예제에서는 새 블록이 14행에 입력됩니다.

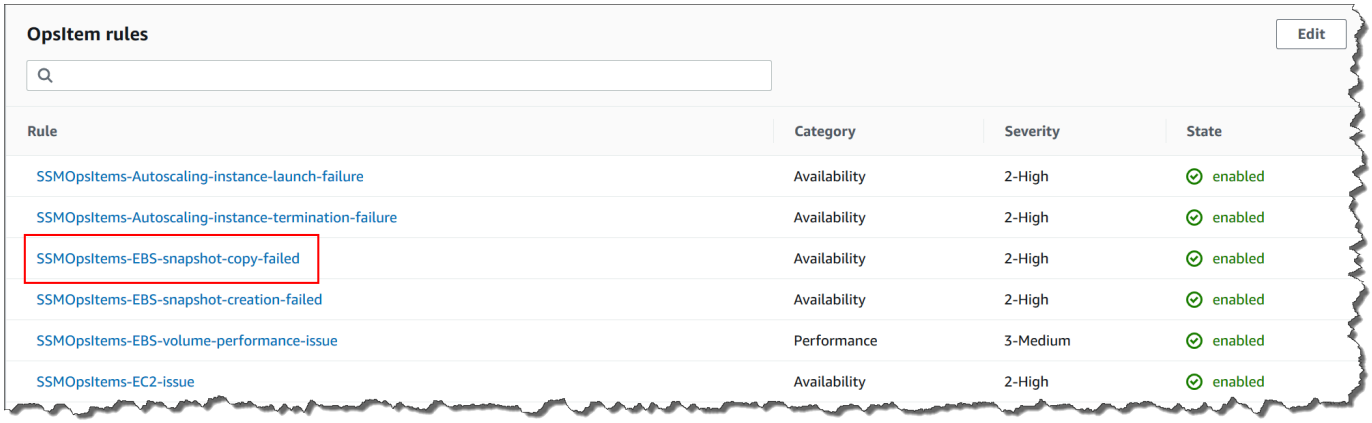
8. Save changes(변경 사항 저장)를 선택합니다.

4단계: 새 OpsItems에 대한 알림을 보내도록 기본 OpsItems 규칙 켜기

Amazon EventBridge의 기본 OpsItems 규칙은 Amazon SNS 알림에 대한 Amazon 리소스 이름(ARN)으로 구성되어 있지 않습니다. 다음 절차에 따라 EventBridge에서 규칙을 편집하고 `notifications` 블록을 입력합니다.

기본 OpsItem 규칙에 알림 블록을 추가하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 OpsCenter를 선택합니다.
3. OpsItems 탭을 선택한 다음 Configure sources(소스 구성)을 선택합니다.
4. 다음 예제와 같이 `notifications` 블록을 사용해 구성할 소스 규칙의 이름을 선택합니다.



   규칙이 Amazon EventBridge에서 열립니다.
5. 규칙 세부 정보 페이지의 Targets(대상) 탭에서 Edit(편집)을 선택합니다.
6. Additional settings(추가 설정) 섹션에서 Configure input transformer(입력 변환기 구성)을 선택합니다.
7. 템플릿 상자에서 다음 형식으로 `notifications` 블록으로 추가합니다.

```
"notifications":[{"arn":"arn:aws:sns:region:account ID:topic name"}],
```

   다음은 그 예입니다.

```
"notifications":[{"arn":"arn:aws:sns:us-west-2:1234567890:MySNSTopic"}],
```

   미국 서부(오레곤)(us-west-2) 리전에 대한 다음 예제에서와 같이 `resources` 블록 앞에 알림 블록을 입력합니다.

```
{
    "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. 확인을 선택합니다.
9. 다음을 선택합니다.
10. 다음을 선택합니다.
11. 규칙 업데이트를 선택합니다.

시스템에서 기본 규칙에 대한 OpsItem이 생성되고 나면 Amazon SNS 주제에 대한 알림이 게시됩니다.

다른 AWS 서비스와 OpsCenter 통합

AWS Systems Manager의 기능인 OpsCenter는 여러 AWS 서비스와 통합되어 AWS 리소스 관련 문제를 진단하고 해결합니다. OpsCenter와 통합하기 전에 AWS 서비스를 설정해야 합니다.

기본적으로 다음 AWS 서비스는 OpsCenter와 통합되어 있으며 OpsItems를 자동으로 생성할 수 있습니다.

- Amazon CloudWatch
- Amazon CloudWatch Application Insights
- Amazon EventBridge
- AWS Config
- AWS Systems Manager Incident Manager

OpsItems를 자동으로 생성하려면 다음 서비스를 OpsCenter와 통합해야 합니다.

- Amazon DevOps Guru
- AWS Security Hub

이러한 서비스 중 하나가 OpsItem을 생성하면 OpsCenter에서 OpsItem을 관리하고 수정할 수 있습니다. 자세한 내용은 OpsItems 관리 및 OpsItem 문제 해결 단원을 참조하세요.

각 AWS 서비스 및 서비스와 OpsCenter 통합 방식에 대한 자세한 내용은 다음 주제를 참조하세요.

주제

- Amazon CloudWatch
- Amazon CloudWatch Application Insights
- Amazon DevOps Guru
- Amazon EventBridge
- AWS Config
- AWS Security Hub
- Incident Manager

Amazon CloudWatch

Amazon CloudWatch는 AWS 리소스와 서비스를 모니터링하고 사용하는 모든 AWS 서비스에 대한 지표를 표시합니다. CloudWatch는 경보가 경보 상태가 되면 OpsItem을 생성합니다. 예를 들어 Application Load Balancer에서 생성된 HTTP 오류가 급증하는 경우 자동으로 OpsItem을 생성하도록 경보를 구성할 수 있습니다.

OpsItems를 생성하기 위해 CloudWatch에서 구성할 수 있는 일부 경보는 다음 목록에 나와 있습니다.

- Amazon DynamoDB: 데이터베이스 읽기 및 쓰기 작업이 임계값에 도달함
- Amazon EC2: CPU 사용률이 임계값에 도달함
- AWS 결제: 예상 요금이 임계값에 도달함
- Amazon EC2: 인스턴스 상태 확인 실패
- Amazon Elastic Block Store(EBS): 디스크 공간 사용률이 임계값에 도달함

경보를 생성하거나 기존 경보를 편집하여 OpsItem을 생성할 수 있습니다. 자세한 내용은 OpsItems를 생성하도록 CloudWatch 경보 구성 단원을 참조하십시오.

통합 설정을 사용하여 OpsCenter를 활성화하면 CloudWatch가 OpsCenter와 통합됩니다.

Amazon CloudWatch Application Insights

Amazon CloudWatch Application Insights를 사용하면 애플리케이션 리소스에 가장 적합한 모니터를 설정하여 애플리케이션의 문제 징후에 대한 데이터를 지속적으로 분석할 수 있습니다. CloudWatch Application Insights에서 애플리케이션 리소스를 구성할 때 시스템이 OpsCenter에 OpsItems를 생성하도록 선택할 수 있습니다. 애플리케이션에서 감지된 모든 문제에 대한 OpsItem이 OpsCenter 콘솔에 생성됩니다. 자세한 내용은 Amazon CloudWatch 사용 설명서의 모니터링을 위해 애플리케이션 설정, 구성, 관리를 참조하세요.

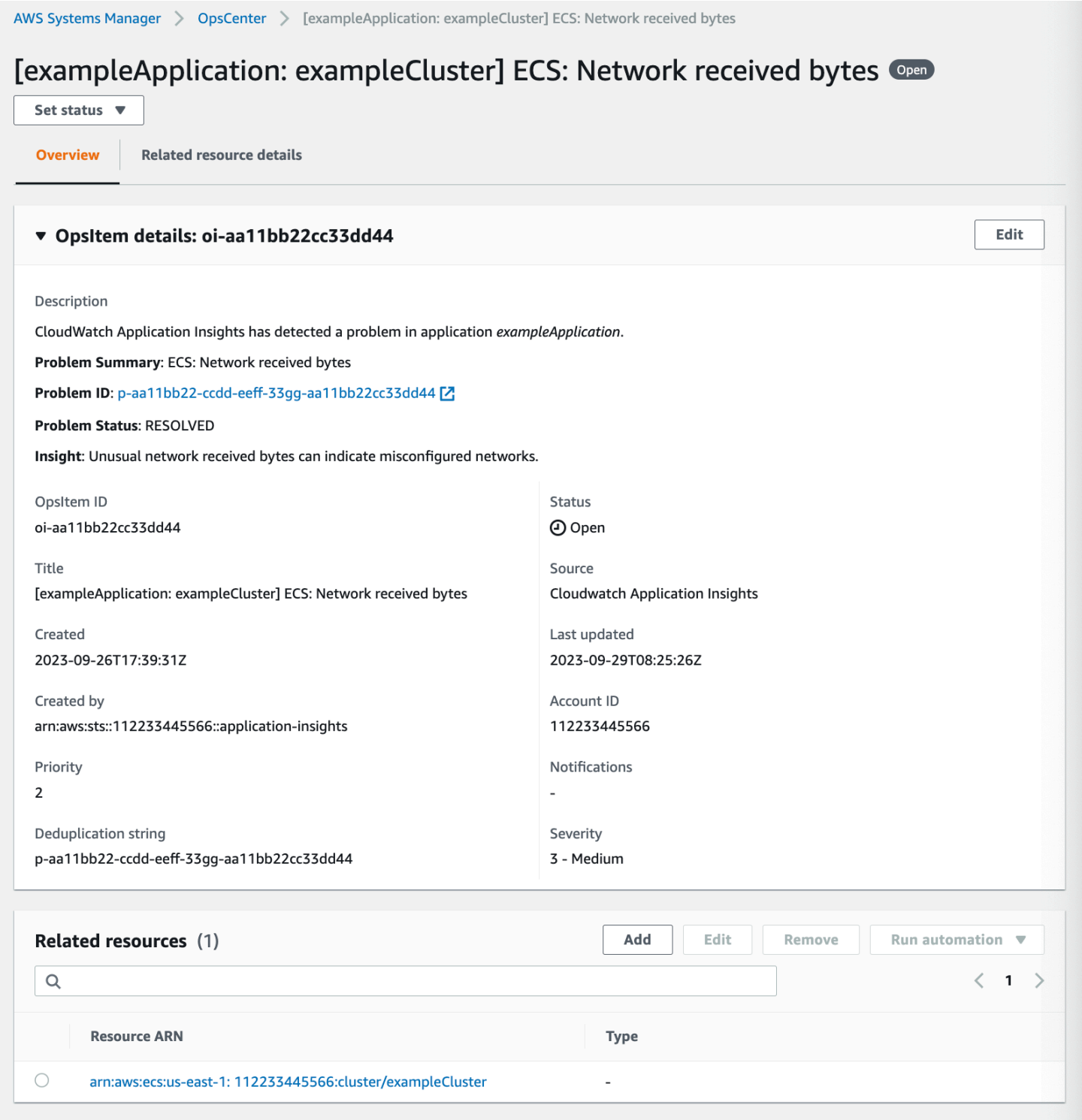
> **Note**
>
> 2023년 10월 16일부터 CloudWatch Application Insights에서 생성된 OpsItems의 제목과 설명에 다음과 같은 개선된 형식이 사용됩니다.

```
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>
```

예:



Amazon DevOps Guru

Amazon DevOps Guru는 운영 데이터, 애플리케이션 지표 및 애플리케이션 이벤트를 분석하여 정상적인 운영 패턴에서 벗어나는 동작을 식별하기 위해 기계 학습을 적용합니다. DevOps Guru가

OpsCenter에서 OpsItem을 생성하도록 설정하면 각 인사이트가 새 OpsItem을 생성합니다. OpsCenter를 사용하여 OpsItems를 관리할 수 있습니다.

DevOps Guru에서 자동으로 OpsItems가 생성됩니다. Amazon DevOps Guru를 활성화하여 Systems Manager의 기능인 Quick Setup을 사용하여 OpsItems를 생성할 수 있습니다. 시스템은 AWSServiceRoleForDevOpsGuru AWS Identity and Access Management(IAM) 서비스 연결 역할을 사용하여 OpsItems를 생성합니다.

DevOps Guru와 OpsCenter 통합

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Quick Setup를 선택합니다.
3. DevOps Guru 구성 옵션 사용자 지정 페이지에서 라이브러리 탭을 선택합니다.
4. DevOps Guru 창에서 생성을 선택합니다.
5. 구성 옵션에서 AWS Systems Manager OpsItems 활성화를 선택합니다.
6. 설정을 완료한 후 생성을 선택합니다.

Amazon EventBridge

Amazon EventBridge는 AWS 리소스의 변경 사항을 설명하는 이벤트의 스트림을 거의 제공합니다. 통합 설정을 사용하여 OpsCenter를 활성화하면 EventBridge가 OpsCenter와 통합되고 기본 EventBridge 규칙이 활성화됩니다. 이러한 규칙에 따라 EventBridge는 OpsItems를 생성합니다. 규칙을 사용하여 조사 및 수정을 위해 이벤트를 필터링하고 OpsCenter 라우팅할 수 있습니다.

> **Note**
>
> Amazon EventBridge(이전 Amazon CloudWatch Events)는 CloudWatch Events의 모든 기능과 사용자 지정 이벤트 버스, 타사 이벤트 소스, 스키마 레지스트리 등의 몇 가지 새로운 기능을 제공합니다.

다음은 OpsItem을 생성하기 위해 EventBridge에서 구성할 수 있는 몇 가지 규칙입니다.

- Security Hub: 보안 경보 발생
- Amazon DynamoDB 제한 이벤트
- Amazon Elastic Compute Cloud Auto Scaling 인스턴스 시작 실패
- Systems Manager: 자동화 실행 실패

- AWS Health: 예약된 유지 관리에 대한 알림
- Amazon EC2: 인스턴스 상태가 실행 중에서 중지로 변경됨

요구 사항에 따라 규칙을 생성하거나 기존 규칙을 편집하여 OpsItems를 생성할 수 있습니다. 규칙을 편집하여 OpsItem을 생성하는 방법에 대한 지침은 OpsItems를 생성하도록 EventBridge 규칙 구성 섹션을 참조하세요.

AWS Config

AWS Config은 귀하의 AWS 계정에 있는 AWS 리소스의 구성을 자세히 보여줍니다.

AWS Config는 직접 OpsCenter와 통합되지 않습니다. 그 대신에 Amazon EventBridge로 이벤트를 보내는 AWS Config 규칙을 사용자가 생성합니다(예: AWS Config에서 규정 미준수 인스턴스를 탐지하는 경우). 그러면 EventBridge에서는 사용자가 생성한 EventBridge 규칙에 따라 해당 이벤트를 평가합니다. 규칙이 일치하면 EventBridge에서는 이벤트를 OpsItem으로 변환하여 OpsCenter에 대상 타겟으로 전송합니다.

이 OpsItem을 사용하여 비준수 리소스의 세부 정보를 추적하고, 조사 작업을 기록하고, 일관된 수정 작업에 대한 액세스를 제공할 수 있습니다.

관련 정보

OpsItems를 생성하도록 EventBridge 규칙 구성

규정 준수 모니터링에 AWS Systems ManagerOpsCenter 및 AWS Config 사용

AWS Security Hub

AWS Security Hub에서는 AWS 계정 및 서비스에서 조사 결과라는 보안 데이터를 수집합니다. 규칙 세트를 사용하여 조사 결과를 탐지하고 생성하는 Security Hub는 사용자가 관리하는 리소스의 보안 문제를 식별하고, 우선순위를 지정하고, 해결하는 데 도움이 됩니다. 이 주제에 설명된 대로 통합을 구성하면 Systems Manager에서는 OpsCenter의 Security Hub 조사 결과에 대한 OpsItems가 생성됩니다.

> **Note**
>
> OpsCenter는 Security Hub와 양방향으로 통합되어 있습니다. 즉, 보안 조사 결과와 관련된 OpsItem에 대한 상태 또는 심각도 필드를 업데이트하면 시스템에서는 변경 사항이 Security Hub와 동기화됩니다. 마찬가지로, 조사 결과의 변경 사항이 OpsCenter의 해당 OpsItems에서 자동으로 업데이트됩니다.

Security Hub 검색 결과에서 OpsItem을 생성하면 Security Hub 메타데이터가 OpsItem의 운영 데이터 필드에 자동으로 추가됩니다. 이 메타데이터를 삭제하면 양방향 업데이트가 더 이상 작동하지 않습니다.

기본적으로 Systems Manager에서는 심각도 심각 및 높음 조사 결과에 대한 OpsItems가 생성됩니다. 심각도 중간 및 낮음 조사 결과에 대한 OpsItems가 생성되도록 OpsCenter를 수동으로 구성할 수 있습니다. 정보를 제공하는 조사 결과는 해결이 필요하지 않아서 OpsCenter에서 OpsItems가 생성되지 않습니다. Security Hub 심각도 수준에 대한 자세한 내용은 AWS Security Hub API 참조의 심각도를 참조하세요.

시작하기 전 준비 사항

Security Hub 조사 결과에 따라 OpsItems가 생성되도록 OpsCenter를 구성하기 전에 Security Hub 설정 작업을 완료했는지 확인하세요. 자세한 내용은 AWS Security Hub User Guide의 Setting up Security Hub를 참조하세요.

Security Hub를 OpsCenter와 통합하면 시스템에서는 `AWSServiceRoleForSystemsManagerOpsDataSync` IAM 서비스 연결 역할을 사용하여 OpsItems를 생성합니다. 이에 대한 자세한 내용은 역할을 사용하여 Explorer용 OpsData 및 OpsItems 생성 섹션을 참조하세요.

Warning

Security Hub와 OpsCenter 통합 요금에 대한 다음과 같은 중요한 정보를 참고합니다.

- OpsCenter 및 Security Hub 통합을 구성할 때 Security Hub 관리자 계정으로 로그인한 경우 시스템에서는 관리자 및 모든 멤버 계정의 조사 결과에 대한 OpsItems가 생성됩니다. OpsItems는 모두 관리자 계정에서 생성됩니다. 다양한 요인에 따라 예상외로 많은 요금이 AWS에서 청구될 수 있습니다.

  통합을 구성할 때 멤버 계정에 로그인한 경우 시스템에서는 해당 개별 계정의 조사 결과에 대한 OpsItems만 생성됩니다. Security Hub 관리자 계정, 멤버 계정, 조사 결과에 대한 EventBridge 이벤트 피드 관련성에 대한 자세한 내용은 AWS Security Hub 사용 설명서의 EventBridge와 Security Hub 통합 유형을 참조하세요.
- OpsItem이 생성되는 조사 결과마다 정상 OpsItem 생성 가격이 부과됩니다. OpsItem을(를) 편집하거나 Security Hub에서 해당 조사 결과가 업데이트된 경우(OpsItem 업데이트가 트리거된 경우)에도 요금이 부과됩니다.

Security Hub 조사 결과의 OpsItems가 생성되도록 OpsCenter를 구성하는 방법

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 OpsCenter를 선택합니다.
3. 설정을 선택합니다.
4. Security Hub 조사 결과 섹션에서 편집을 선택합니다.
5. 슬라이더를 선택하여 비활성화됨을 활성화됨으로 변경합니다.
6. 시스템에서 심각도 중간 또는 낮음 조사 결과의 OpsItems가 생성되도록 하려면 이러한 옵션을 전환합니다.
7. 저장을 선택하여 구성을 저장합니다.

시스템에서 Security Hub 조사 결과의 OpsItems가 더는 생성되지 않도록 하려면 다음 절차를 사용합니다.

Security Hub 조사 결과의 OpsItems 수신을 중지하는 방법

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 OpsCenter를 선택합니다.
3. 설정을 선택합니다.
4. Security Hub 조사 결과 섹션에서 편집을 선택합니다.
5. 슬라이더를 선택하여 활성화됨을 비활성화됨으로 변경합니다. 슬라이더를 전환할 수 없으면 Security Hub가 AWS 계정에 대해 활성화되지 않은 것입니다.
6. 저장을 선택하여 구성을 저장합니다.OpsCenter에서 더는 Security Hub 조사 결과에 따라 OpsItems가 생성되지 않습니다.

> **Important**
>
> Systems Manager가 위임한 관리자 또는 AWS Organizations 관리 계정은 Explorer에서 리소스 데이터 동기화를 생성하여 여러 계정과 AWS 리전에 대해 OpsCenter에서 Security Hub 조사 결과를 활성화할 수 있습니다. Explorer에서 Security Hub 소스가 활성화되고 Security Hub 통합을 비활성화한 구성원 계정을 대상으로 하는 리소스 데이터 동기화가 있는 경우 관리자가 선택한 설정이 우선합니다. OpsCenter는 Security Hub 조사 결과에 대해 OpsItems를 계속 생성합니다. 리소스 데이터 동기화 대상 멤버 계정에서 Security Hub 조사 결과에 대한 OpsItems 생성을 중지하려면 관리자에게 문의하여 리소스 데이터 동기화에서 계정을 제거하

거나 Explorer에서 Security Hub 소스를 비활성화하도록 요청하세요. Explorer의 설정 변경에 대한 자세한 내용은 Systems Manager Explorer 데이터 원본 편집 섹션을 참조하세요.

Incident Manager

AWS Systems Manager의 기능인 Incident Manager는 AWS 호스팅 애플리케이션에 영향을 주는 인시던트를 완화하고 복구하는 데 사용할 수 있는 인시던트 관리 콘솔을 제공합니다. 인시던트는 서비스 품질의 계획되지 않은 중단 또는 감소입니다. Incident Manager를 설정하고 구성한 후에는 OpsCenter에 OpsItems이 자동으로 생성됩니다.

시스템이 Incident Manager에서 인시던트를 생성하면 OpsCenter에도 OpsItem이 생성되고 인시던트가 관련 항목으로 표시됩니다. OpsItem이 이미 있는 경우 Incident Manager는 OpsItem을 생성하지 않습니다. 첫 번째 OpsItem은 상위 OpsItem이라고 합니다. 인시던트의 규모와 범위가 커지면 기존 OpsItem에 인시던트를 추가할 수 있습니다. 필요한 경우 OpsItem에 대한 인시던트를 수동으로 생성할 수 있습니다. 인시던트가 종결된 후 Incident Manager에서 분석을 생성하여 유사한 문제에 대한 수정 프로세스를 검토하고 개선할 수 있습니다.

기본적으로 OpsCenter는 Incident Manager와 통합됩니다. Incident Manager가 설정되지 않은 경우 OpsCenter 페이지에 Incident Manager를 설정하라는 메시지가 표시됩니다. Incident Manager가 OpsItem을 생성하면 OpsCenter에서 OpsItem을 관리하고 수정할 수 있습니다. OpsItem에 대한 인시던트 생성 지침은 OpsItem에 대한 인시던트 생성 섹션을 참조하세요.

Create OpsItems

AWS Systems Manager의 기능인 OpsCenter를 설정하고 AWS 서비스와 통합하면 AWS 서비스가 기본 규칙, 이벤트 또는 경보를 기반으로 OpsItems를 자동으로 생성합니다.

기본 Amazon EventBridge 규칙의 상태와 심각도 수준을 볼 수 있습니다. 필요한 경우 Amazon EventBridge에서 이러한 규칙을 생성하거나 편집할 수 있습니다. Amazon CloudWatch에서 경보를 보고 생성하거나 편집할 수도 있습니다. 규칙과 경보를 사용하여 OpsItems를 자동으로 생성하려는 이벤트를 구성할 수 있습니다.

시스템에서 OpsItem을 생성할 때 미결 상태입니다. 상태를 OpsItem 조사 시작 시 진행 중으로 변경하고 OpsItem 수정 후 해결됨으로 변경할 수 있습니다. AWS 서비스에서 경보와 규칙을 구성하여 OpsItems를 생성하는 방법과 수동으로 OpsItems를 생성하는 방법에 대한 자세한 내용은 다음 주제를 참조하세요.

주제

- OpsItems를 생성하도록 EventBridge 규칙 구성
- OpsItems를 생성하도록 CloudWatch 경보 구성
- 수동으로 OpsItems 만들기

OpsItems를 생성하도록 EventBridge 규칙 구성

Amazon EventBridge는 이벤트를 수신하면 기본 규칙을 기반으로 새 OpsItem을 생성합니다. 규칙을 생성하거나 기존 규칙을 편집하여 OpsCenter를 EventBridge 이벤트의 대상으로 설정할 수 있습니다. 이벤트 규칙을 생성하는 방법에 대한 자세한 내용은 Amazon EventBridge 사용 설명서의 AWS 서비스에 대한 규칙 생성을 참조하세요.

OpsCenter에서 OpsItems를 생성하도록 EventBridge 규칙 구성

1. https://console.aws.amazon.com/events/에서 Amazon EventBridge 콘솔을 엽니다.
2. 탐색 창에서 규칙을 선택합니다.
3. Rules(규칙) 페이지의 Event bus(이벤트 버스)에서 default(기본)를 선택합니다.
4. 규칙에서 이름 옆에 있는 확인란을 선택하여 규칙을 선택합니다.
5. 규칙 이름을 선택하여 세부 정보 페이지를 엽니다. 규칙 세부 정보 섹션에서 상태가 활성화됨으로 설정되었는지 확인합니다.

   > **Note**
   >
   > 필요한 경우 페이지 오른쪽 상단에 있는 이벤트를 사용하여 상태를 업데이트할 수 있습니다.

6. 대상 탭을 선택합니다.
7. [Targets] 탭에서 [Edit]를 선택합니다.
8. 대상 유형의 경우 AWS 서비스를 선택합니다.
9. Select a target(대상 선택)에서 Systems Manager OpsItem을 선택합니다.
10. 여러 대상 유형에 대해 EventBridge에서는 대상에 이벤트를 보낼 권한이 필요합니다. 이 경우 EventBridge는 이벤트 실행에 필요한 AWS Identity and Access Management(IAM) 역할을 생성할 수 있습니다.

    - IAM 역할을 자동으로 생성하려면 이 특정 리소스에 대해 새 역할 생성을 선택합니다.
    - OpsCenter에서 OpsItems을(를) 생성할 수 있는 권한을 EventBridge 부여하기 위해 생성한 IAM 역할을 사용하려면 Use existing role(기존 역할 사용)을 선택합니다.

11. 추가 설정 섹션의 대상 입력 구성에서 입력 변환기를 선택합니다.

    입력 변환기 옵션을 사용하여 중복 제거 문자열과 제목 및 심각도와 같은 OpsItems에 대한 기타 중요한 정보를 지정할 수 있습니다.

12. Configure input transformer(입력 구성 변환기)를 선택합니다.

13. 대상 입력 변환기의 입력 경로에서 트리거하는 이벤트에서 구문 분석할 값을 지정합니다. 예를 들어 규칙을 트리거하는 이벤트에서 시작 시간, 종료 시간 및 기타 세부 정보를 구문 분석하려면 다음 JSON을 사용합니다.

```
{
    "end-time": "$.detail.EndTime",
    "failure-cause": "$.detail.cause",
    "resources": "$.resources",
    "source": "$.detail.source",
    "start-time": "$.detail.StartTime"
}
```

14. Template(템플릿)에서 대상으로 전송할 정보를 지정합니다. 예를 들어 다음 JSON을 사용하여 OpsCenter에 정보를 전달합니다. 해당 정보는 OpsItem을(를) 생성하는 데 사용됩니다.

> **Note**
>
> 입력 템플릿이 JSON 형식인 경우 템플릿의 객체 값에는 따옴표가 포함될 수 없습니다. 예를 들면, 리소스, 오류 원인, 소스, 시작 시간 및 종료 시간의 값은 따옴표로 묶을 수 없습니다.

```
{
    "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",
    "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>
        }
    }
}
```

이러한 필드에 대한 자세한 내용은 Amazon EventBridge User Guide의 Transforming target input을 참조하세요.

15. 확인을 선택합니다.
16. 다음을 선택합니다.
17. 다음을 선택합니다.
18. 규칙 업데이트를 선택합니다.

이벤트에서 OpsItem이 생성되면 OpsItem을 열고 Private operational data(프라이빗 운영 데이터) 섹션으로 스크롤하여 이벤트 세부 정보를 볼 수 있습니다. OpsItem에서 옵션을 구성하는 방법에 대한 자세한 내용은 OpsItems 관리 섹션을 참조하세요.

OpsItems를 생성하도록 CloudWatch 경보 구성

AWS Systems Manager의 기능인 OpsCenter의 통합 설정 중 Amazon CloudWatch에서 일반 경보를 기반으로 OpsItems을 자동으로 생성하도록 설정할 수 있습니다. 경보를 생성하거나 기존 경보를 편집하여 OpsCenter에 OpsItems를 생성할 수 있습니다.

CloudWatch는 OpsItems를 생성하도록 경보를 구성할 때 AWS Identity and Access Management(IAM)에 새로운 서비스 연결 역할을 생성합니다. 새 역할의 이름은 `AWSServiceRoleForCloudWatchAlarms_ActionSSM`입니다. CloudWatch Service 연결 역할에

대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 CloudWatch에 서비스 연결 역할 사용을 참조하세요.

CloudWatch 경보가 OpsItem을 생성하면 OpsItem은 CloudWatch 경보 - '***alarm_name***'이 ALARM 상태에 있음을 표시합니다.

특정 OpsItem에 대한 세부 정보를 보려면 OpsItem을 선택한 다음 관련 리소스 세부 정보 탭을 선택합니다. OpsItems를 수동으로 편집하여 심각도 또는 범주와 같은 세부 정보를 변경할 수 있습니다. 그러나 경보의 심각도 또는 범주를 편집할 때 Systems Manager는 경보에서 이미 생성된 OpsItems의 심각도 또는 범주를 업데이트할 수 없습니다. 경보가 OpsItem을 생성하고 중복 제거 문자열을 지정한 경우 CloudWatch에서 경보를 편집하더라도 경보는 추가 OpsItems를 생성하지 않습니다. OpsCenter에서 OpsItem이 해결되면 경우 CloudWatch가 새로운 OpsItem을 생성합니다.

CloudWatch 경보 구성에 대한 자세한 내용은 다음 주제를 참조하세요.

주제

- OpsItems를 생성하도록 CloudWatch 경보 구성(콘솔)
- OpsItems를 생성하도록 기존 CloudWatch 경보 구성(프로그래밍 방식으로)

OpsItems를 생성하도록 CloudWatch 경보 구성(콘솔)

수동으로 경보를 생성하거나 기존 경보를 업데이트하여 Amazon CloudWatch에서 OpsItems를 생성할 수 있습니다.

CloudWatch 경보 생성 및 해당 경보의 대상으로 Systems Manager 구성

1. Amazon CloudWatch 사용 설명서의 정적 임계값을 기반으로 CloudWatch 경보 생성에 지정된 대로 1~9단계를 완료합니다.
2. Systems Manager 작업 섹션에서 Systems Manager OpsCenter 작업 추가를 선택합니다.
3. OpsItems를 선택합니다.
4. 심각도는에서 1~4를 선택합니다.
5. (옵션) 범주에서 OpsItem에 대한 범주를 선택합니다.
6. Amazon CloudWatch 사용 설명서의 정적 임계값을 기반으로 CloudWatch 경보 생성에 지정된 대로 11~13단계를 완료합니다.
7. [다음(Next)]을 선택하고 마법사를 완료합니다.

기존 경보를 편집하고 Systems Manager를 해당 경보의 대상으로 지정하려면

1. https://console.aws.amazon.com/cloudwatch/에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 경보(Alarms)를 선택합니다.
3. 경보를 선택하고 [작업(Actions)] 및 [편집(Edit)]을 선택합니다.
4. (옵션) [지표(Metrics)] 및 [조건(Conditions)] 섹션에서 설정을 변경하고 [다음(Next)]을 선택합니다.
5. [Systems Manager] 섹션에서 [Systems Manager OpsCenter 작업 추가(Add Systems Manager OpsCenter action)]를 선택합니다.
6. [심각도(Severity)]에서 숫자를 선택합니다.

> **Note**
>
> 심각도는 사용자가 정의하는 값입니다. 사용자 또는 사용자의 조직이 각 심각도 값의 의미와 각 심각도와 관련된 서비스 수준 계약을 결정합니다.

7. (옵션) [범주(Category)]에서 옵션을 선택합니다.
8. [다음(Next)]을 선택하고 마법사를 완료합니다.

OpsItems를 생성하도록 기존 CloudWatch 경보 구성(프로그래밍 방식으로)

AWS Command Line Interface(AWS CLI), AWS CloudFormation 템플릿 또는 Java 코드 조각을 사용하여 프로그래밍 방식으로 OpsItems를 생성하도록 Amazon CloudWatch 경보를 구성할 수 있습니다.

주제

- 시작하기 전 준비 사항
- OpsItems(AWS CLI)를 생성하도록 CloudWatch 경보 구성
- OpsItems를 생성하거나 업데이트하도록 CloudWatch 경보 구성(CloudFormation)
- OpsItems(Java)를 생성하거나 업데이트하도록 CloudWatch 경보 구성

시작하기 전 준비 사항

기존 경보를 프로그래밍 방식으로 편집하거나 OpsItems를 생성하는 경보를 생성하는 경우 Amazon 리소스 이름(ARN)을 지정해야 합니다. 이 ARN은 경보에서 생성된 OpsItems의 대상으로 Systems Manager OpsCenter를 식별합니다. 경보에서 생성된 OpsItems에 심각도 또는 범주와 같은 특정 정보가 포함되도록 ARN을 사용자 지정할 수 있습니다. 각 ARN에는 다음 표에 설명된 정보가 포함됩니다.

파라미터	Details
Region(필수)	경보가 있는 AWS 리전입니다. 예: us-west-2 . OpsCenter를 사용할 수 있는 AWS 리전에 대한 자세한 내용은 AWS Systems Manager 엔드포인트 및 할당량을 참조하세요.
account_ID (필수)	경보 생성에 사용된 동일한 AWS 계정 ID입니다. 예: 123456789012 . 계정 ID 다음에는 다음 예와 같이 콜론(:)과 파라미터 opsitem이 와야 합니다.
severity(필수)	경보에서 생성된 OpsItems에 대한 사용자 정의 심각도 수준입니다. 유효값: 1, 2, 3, 4
Category(선택 사항)	경보에서 생성된 OpsItems에 대한 범주입니다. 유효한 값: Availability , Cost, Performance , Recovery 및 Security.

다음 구문을 사용하여 ARN을 생성합니다. 이 ARN에는 선택적 Category 파라미터가 포함되어 있지 않습니다.

```
arn:aws:ssm:Region:account_ID:opsitem:severity
```

다음은 한 예입니다.

```
arn:aws:ssm:us-west-2:123456789012:opsitem:3
```

선택적 Category 파라미터를 사용하는 ARN을 생성하려면 다음 구문을 사용합니다.

```
arn:aws:ssm:Region:account_ID:opsitem:severity#CATEGORY=category_name
```

다음은 한 예입니다.

```
arn:aws:ssm:us-west-2:123456789012:opsitem:3#CATEGORY=Security
```

OpsItems(AWS CLI)를 생성하도록 CloudWatch 경보 구성

이 명령을 사용하려면 `alarm-actions` 파라미터에 대한 ARN을 지정해야 합니다. ARN을 생성하는 방법에 대한 자세한 내용은 시작하기 전 준비 사항 섹션을 참조하세요.

OpsItems(AWS CLI)를 생성하도록 CloudWatch 경보 구성

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.

2. 다음 명령을 실행하여 구성하려는 경보에 대한 정보를 수집합니다.

```
aws cloudwatch describe-alarms --alarm-names "alarm name"
```

3. 다음 명령을 실행하여 경보를 업데이트합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
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
```

   다음은 그 예입니다.

   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
```

OpsItems를 생성하거나 업데이트하도록 CloudWatch 경보 구성(CloudFormation)

이 섹션에는 AWS CloudFormation을 자동으로 생성하거나 업데이트하도록 CloudWatch 경보를 구성하는 데 사용할 수 있는 OpsItems 템플릿이 포함되어 있습니다. 각 템플릿을 사용하려면 `AlarmActions` 파라미터에 대한 ARN을 지정해야 합니다. ARN을 생성하는 방법에 대한 자세한 내용은 시작하기 전 준비 사항 섹션을 참조하세요.

지표 경보 - 다음 CloudFormation 템플릿을 사용하여 CloudWatch 지표 경보를 생성하거나 업데이트합니다. 이 템플릿에 지정된 경보는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 상태 확인을 모니터링합니다. 경보가 `ALARM` 상태가 되면 OpsCenter에 OpsItem이 생성됩니다.

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

복합 경보 - 다음 CloudFormation 템플릿을 사용하여 복합 경보를 생성하거나 업데이트합니다. 복합 경보는 여러 지표 경보로 구성됩니다. 경보가 ALARM 상태가 되면 OpsCenter에 OpsItem이 생성됩니다.

```
"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"
           }
        }
    }
```

OpsItems(Java)를 생성하거나 업데이트하도록 CloudWatch 경보 구성

이 섹션에는 OpsItems를 자동으로 생성하거나 업데이트하도록 CloudWatch 경보를 구성하는 데 사용할 수 있는 Java 코드 조각이 포함되어 있습니다. 각 코드 조각을 사용하려면 `validSsmActionStr` 파라미터에 대한 ARN을 지정해야 합니다. ARN을 생성하는 방법에 대한 자세한 내용은 시작하기 전 준비 사항 섹션을 참조하세요.

특정 경보 - 다음 Java 코드 조각을 사용하여 CloudWatch 경보를 생성하거나 업데이트합니다. 이 템플릿에 지정된 경보는 Amazon EC2 인스턴스 상태 확인을 모니터링합니다. 경보가 `ALARM` 상태가 되면 OpsCenter에 OpsItem이 생성됩니다.

```
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);
    }
```

모든 경보 업데이트 - 다음 Java 코드 조각을 사용하여 경보가 ALARM 상태가 될 때 OpsItems를 생성하도록 AWS 계정의 모든 CloudWatch 경보를 업데이트합니다.

```
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;
            }
        }
    }
```

수동으로 OpsItems 만들기

운영 문제를 발견하면 AWS Systems Manager의 기능인 OpsCenter에서 OpsItem을 수동으로 생성하여 문제를 관리하고 해결할 수 있습니다.

크로스 계정 관리를 위해 OpsCenter를 설정하는 경우 Systems Manager 위임 관리자 또는 AWS Organizations 관리 계정은 멤버 계정에 대해 OpsItems를 생성할 수 있습니다. 자세한 내용은 (선택 사항) 여러 계정에서 OpsItems를 중앙 집중식으로 관리하도록 OpsCenter 설정 단원을 참조하십시오.

AWS Systems Manager 콘솔, AWS Command Line Interface(AWS CLI) 또는 AWS Tools for Windows PowerShell를 사용하여 OpsItems를 생성할 수 있습니다.

주제

수동으로 OpsItems 생성(콘솔)

AWS Systems Manager 콘솔을 사용하여 OpsItems를 수동으로 생성할 수 있습니다. OpsItem을 생성하면 OpsCenter 계정에 표시됩니다. 크로스 계정 관리를 위해 OpsCenter를 설정하는 경우 OpsCenter는 위임된 관리자 또는 관리 계정에 선택한 멤버 계정에 대한 OpsItems를 생성하는 옵션을 제공합니다. 자세한 내용은 (선택 사항) 여러 계정에서 OpsItems를 중앙 집중식으로 관리하도록 OpsCenter 설정 단원을 참조하십시오.

AWS Systems Manager 콘솔을 사용하여 OpsItem 생성

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 OpsCenter를 선택합니다.
3. OpsItem 생성(Create)을 선택합니다. 이 버튼이 표시되지 않으면 [OpsItems] 탭을 선택한 다음 [OpsItem 생성(Create OpsItem)]을 선택합니다.
4. (선택 사항) 기타 계정을 선택한 다음 OpsItem을 생성하려는 계정을 선택합니다.

   > **Note**
   >
   > 이 단계는 멤버 계정에 대한 OpsItems을 생성하는 경우에 필요합니다.

5. 제목에 OpsItem의 목적을 이해하는 데 도움이 되는 설명이 포함된 이름을 입력합니다.
6. [소스(Source)]에 사용자가 OpsItem의 출처를 이해하는 데 도움을 줄 수 있도록 영향을 받는 AWS 리소스 유형 또는 기타 소스 정보를 입력합니다.

   > **Note**
   >
   > OpsItem을 생성한 후에는 [소스(Source)] 필드를 편집할 수 없습니다.

7. (옵션) [우선순위(Priority)]에서 우선순위 수준을 선택합니다.

8. (옵션) [심각도(Severity)]에서 심각도 수준을 선택합니다.
9. (옵션) [범주(Category)]에서 범주를 선택합니다.
10. 설명에 문제를 재현하기 위한 단계(해당하는 경우)를 포함하여 이 OpsItem에 대한 정보를 입력합니다.

> **Note**
>
> 콘솔에서는 OpsItem 설명 필드에 있는 대부분의 마크다운 서식이 지원됩니다. 자세한 내용은 AWS Management Console 시작하기 시작 안내서의 콘솔에서 마크다운 사용을 참조하세요.

11. 중복 제거 문자열에 시스템에서 중복 OpsItems를 확인하는 데 사용할 수 있는 단어를 입력합니다. 중복 제거 문자열에 대한 자세한 내용은 중복 OpsItems 관리 섹션을 참조하세요.
12. (옵션) 알림에 이 OpsItem이 업데이트될 때 알림을 보낼 Amazon SNS 주제의 Amazon 리소스 이름(ARN)을 지정합니다. OpsItem과 동일한 AWS 리전에 있는 Amazon SNS ARN을 지정해야 합니다.
13. (선택 사항) 관련 리소스에서 추가를 선택하여 영향을 받는 리소스 및 관련 리소스의 ID 또는 ARN을 지정합니다.
14. 생성(Create)OpsItem을 선택합니다.

성공하면 페이지에 OpsItem이 표시됩니다. 위임된 관리자 또는 관리 계정이 선택한 멤버 계정에 대한 OpsItem을 생성하면 새 OpsItems가 관리자 및 멤버 계정의 OpsCenter에 표시됩니다. OpsItem에서 옵션을 구성하는 방법에 대한 자세한 내용은 OpsItems 관리 섹션을 참조하세요.

수동으로 OpsItems 생성(AWS CLI)

다음 절차에서는 AWS Command Line Interface(AWS CLI)를 사용하여 OpsItem을 생성하는 방법에 대해 설명합니다.

AWS CLI를 사용하여 OpsItem를 생성하려면

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.
2. AWS CLI를 열고 다음 명령을 실행하여 OpsItem을 생성합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
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"
```

파일에서 운영 데이터 지정

OpsItem을 생성할 때 파일에서 운영 데이터를 지정할 수 있습니다. 파일은 JSON 파일이어야 하며 파일의 콘텐츠는 다음 형식을 사용해야 합니다.

```
{
  "key_name": {
    "Type": "SearchableString",
    "Value": "Up to 20 KB of data"
  }
}
```

의 예는 다음과 같습니다.

```
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**
>
> 다른 로컬 운영 체제의 명령줄에 JSON 형식의 파라미터를 입력하는 방법에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서의 AWS CLI에서 문자열에 따옴표 사용을 참조하세요.

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "OpsItemId": "oi-1a2b3c4d5e6f"
}
```

3. 다음 명령을 실행하여 생성한 OpsItem에 대한 세부 정보를 봅니다.

```
aws ssm get-ops-item --ops-item-id ID
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "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. 다음 명령을 실행하여 OpsItem을 업데이트합니다. 이 명령은 상태를 Open(기본값)에서 InProgress로 변경합니다.

```
aws ssm update-ops-item --ops-item-id ID --status InProgress
```

이 명령에는 출력이 없습니다.

5. 다음 명령을 다시 실행하여 상태가 InProgress로 변경되었는지 확인합니다.

```
aws ssm get-ops-item --ops-item-id ID
```

OpsItem 생성의 예제

다음 코드 예제에서는 Linux 관리 포털, macOS 또는 Windows를 사용하여 OpsItem을 생성하는 방법을 보여줍니다.

Linux 관리 포털 또는 macOS

다음 명령은 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 디스크가 가득 차면 OpsItem을 생성합니다.

```
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"
```

다음 명령은 OperationalData의 /aws/resources 키를 사용하여 Amazon DynamoDB 관련 리소스가 있는 OpsItem을 생성합니다.

```
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"
```

다음 명령은 OperationalData의 /aws/automations 키를 사용하여 AWS-ASGEnterStandby 문서를 연결된 Automation 런북으로 지정하는 OpsItem을 생성합니다.

```
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

다음 명령은 Amazon Relational Database Service(Amazon RDS) 인스턴스가 응답하지 않을 때 OpsItem을 생성합니다.

```
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"
```

다음 명령은 OperationalData의 /aws/resources 키를 사용하여 Amazon EC2 인스턴스 관련 리소스가 있는 OpsItem을 생성합니다.

```
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\"}}
```

다음 명령은 OperationalData의 /aws/automations 키를 사용하여 AWS-RestartEC2Instance 런북을 연결된 Automation 런북으로 지정하는 OpsItem을 생성합니다.

```
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\"}}
```

수동으로 OpsItems 생성(PowerShell)

다음 절차에서는 AWS Tools for Windows PowerShell을 사용하여 OpsItem을 생성하는 방법을 설명합니다.

AWS Tools for Windows PowerShell을 사용하여 OpsItem 생성

1. AWS Tools for Windows PowerShell을 열고 다음 명령을 실행하여 자격 증명을 지정합니다.

   ```
   Set-AWSCredentials –AccessKey key-name –SecretKey key-name
   ```

2. 다음 명령을 실행하여 PowerShell 세션의 AWS 리전을 설정합니다.

   ```
   Set-DefaultAWSRegion -Region Region
   ```

3. 다음 명령을 실행해 새 OpsItem을 생성합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다. 이 명령은 이 OpsItem을 해결하기 위한 Systems Manager 자동화 실행서를 지정합니다.

   ```
   $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
```

성공할 경우 명령은 새로운 OpsItem의 ID를 출력합니다.

다음 예제에서는 손상된 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스의 Amazon 리소스 이름(ARN)을 지정합니다.

```
$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
```

OpsItems 관리

AWS Systems Manager의 기능인 OpsCenter는 생성에서 해결까지 OpsItems를 추적합니다. 크로스 계정 관리를 위해 OpsCenter를 설정하는 경우 위임된 관리자 또는 관리 계정은 해당 계정에서 OpsItems를 관리할 수 있습니다. 자세한 내용은 (선택 사항) 여러 계정에서 OpsItems를 중앙 집중식으로 관리하도록 OpsCenter 설정 단원을 참조하십시오.

Systems Manager 콘솔에서 다음 페이지를 사용하여 OpsItems를 보고 관리할 수 있습니다.

- 요약 - 미결 및 진행 중 OpsItems 수, 소스 및 기간별 OpsItems 수, 운영 인사이트를 표시합니다. 소스 및 OpsItems 상태별로 OpsItems를 필터링할 수 있습니다.

- OpsItems - 제목, ID, 우선순위, 설명, OpsItem 소스, 마지막 업데이트 날짜 및 시간과 같은 여러 정보 필드가 있는 OpsItems 목록을 표시합니다. 이 페이지를 사용하여 수동으로 OpsItems을 생성하고, 소스를 구성하고, OpsItem의 상태를 변경하고, 새 인시던트별로 OpsItems을 필터링할 수 있습니다. OpsItem을 선택하여 OpsItems 세부 정보 페이지를 표시할 수 있습니다.

- OpsItem 세부 정보 - OpsItem을 관리하는 데 사용할 수 있는 세부 인사이트와 도구를 제공합니다. OpsItems 세부 정보 페이지에는 다음과 같은 탭이 있습니다.

- 개요 - 관련 리소스, 지난 30일 동안 실행된 런북 및 실행할 수 있는 사용 가능한 런북 목록을 표시합니다. 유사한 OpsItems을 보고, 운영 데이터를 추가하고, 관련 OpsItems을 추가할 수도 있습니다.
- 관련 리소스 세부 정보 – 여러 AWS 서비스의 리소스에 대한 정보를 표시합니다. [리소스 세부 정보(Resource details)] 섹션을 확장하여 이 리소스를 호스팅하는 AWS 서비스에서 제공하는 이 리소스 관련 정보를 봅니다. [관련 리소스(Related resources)] 목록을 사용하여 이 OpsItem과 연결된 다른 관련 리소스를 전환할 수도 있습니다.

OpsItems 관리 방법에 대한 자세한 내용은 다음 주제를 참조하세요.

주제

OpsItem의 세부 정보 보기

OpsItem을 포괄적으로 보려면 OpsCenter 콘솔에서 OpsItem 세부 정보 페이지를 사용하세요. 요약 페이지에 다음 정보가 표시됩니다.

- OpsItems 세부 정보 - 선택한 OpsItem에 대한 일반 정보를 표시합니다.
- 관련 리소스 - 관련 리소스는 영향을 받은 리소스 또는 OpsItem을 생성한 이벤트를 시작한 리소스입니다.
- 지난 30일 동안의 Automation 실행 - 지난 30일 동안 실행된 런북 목록입니다.
- 런북 - 사용 가능한 런북 목록에서 런북을 선택할 수 있습니다.
- 유사한 OpsItems – 시스템에서 생성한 OpsItems 목록으로, 사용자와 관련되거나 관심이 있을 만한 항목입니다. 목록을 생성하기 위해 시스템은 모든 OpsItems의 제목 및 설명을 검색하여 유사한 단어를 사용하는 OpsItems를 반환합니다.

- 운영 데이터 – 운영 데이터는 OpsItem에 대한 유용한 참조 세부 정보를 제공하는 사용자 지정 데이터입니다. 예를 들어 로그 파일, 오류 문자열, 라이선스 키, 문제 해결 팁 또는 기타 관련 데이터를 지정할 수 있습니다.
- 관련 OpsItems - 어떤 식으로든 현재 OpsItem과 관련된 OpsItems의 ID를 지정할 수 있습니다.
- 관련 리소스 세부 정보 - Amazon CloudWatch 지표 및 경보, AWS CloudTrail 로그, AWS Config의 세부 정보를 비롯한 데이터 공급자를 표시합니다.

OpsItem 세부 정보를 보려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 OpsCenter를 선택합니다.
3. 세부 정보를 볼 OpsItem을 선택합니다.

OpsItem 편집

OpsItem 세부 정보 섹션에는 설명, 제목, 소스, OpsItem ID, 상태 등 OpsItem에 대한 정보가 포함되어 있습니다.
단일 OpsItem을 편집하거나, 여러 OpsItems를 선택하고 상태, 우선순위, 심각도, 범주 필드를 편집할 수 있습니다.

Amazon EventBridge는 OpsItem을 생성할 때 제목, 소스 및 설명 필드를 채웁니다. 제목 및 설명 필드는 편집할 수 있지만 소스 필드는 편집할 수 없습니다.

> **Note**
>
> 콘솔에서는 OpsItem 설명 필드에 있는 대부분의 마크다운 서식이 지원됩니다. 자세한 내용은 AWS Management Console 시작하기 시작 안내서의 콘솔에서 마크다운 사용을 참조하세요.

일반적으로 OpsItem에 대해 다음과 같은 구성 가능한 데이터를 편집할 수 있습니다.

- 제목 - OpsItem의 이름입니다. 소스에서 OpsItem의 제목을 생성합니다.
- 설명 – 설명에 문제를 재현하기 위한 단계(해당하는 경우)가 포함된 이 OpsItem에 대한 정보입니다.
- 상태 - OpsItem의 상태는 미결, 진행 중 또는 해결됨일 수 있습니다.
- 우선순위 - OpsItem의 우선순위는 1~5일 수 있습니다. 조직에서 각 우선순위 수준의 의미와 각 수준에 대한 해당 서비스 수준 계약을 결정하는 것이 좋습니다.

- 심각도 - OpsItem의 심각도는 1~4일 수 있습니다. 여기서 1은 위험, 2는 높음, 3은 중간, 4는 낮음입니다.
- 범주 - OpsItem의 범주는 가용성, 비용, 성능, 복구 또는 보안일 수 있습니다.
- 알림 – OpsItem을 편집할 때 알림 필드에서 Amazon Simple Notification Service 주제의 Amazon 리소스 이름(ARN)을 지정할 수 있습니다. ARN을 지정하면 상태 변경을 포함하여 OpsItem이 편집될 때 모든 이해관계자에게 알림이 수신됩니다. 자세한 내용은 Amazon Simple Notification Service Developer Guide를 참조하세요.

> **Important**
>
> Amazon SNS 주제는 OpsItem와 동일한 AWS 리전에 있어야 합니다. 주제와 OpsItem이 다른 리전에 있는 경우 오류가 반환됩니다.

OpsCenter는 AWS Security Hub와 양방향으로 통합되어 있습니다. 보안 결과와 관련된 OpsItem 상태 및 심각도 필드를 업데이트하면 해당 변경 사항이 자동으로 Security Hub로 전송되어 항상 정확한 최신 정보를 확인할 수 있습니다.

Security Hub 검색 결과에서 OpsItem을 생성하면 Security Hub 메타데이터가 OpsItem의 운영 데이터 필드에 자동으로 추가됩니다. 이 메타데이터를 삭제하면 양방향 업데이트가 더 이상 작동하지 않습니다.

OpsItem 세부 정보를 편집하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 OpsCenter를 선택합니다.
3. OpsItem ID를 선택해 세부 정보 페이지를 열거나 여러 OpsItems를 선택합니다. 여러 OpsItems를 선택한 경우 상태, 우선순위, 심각도 또는 범주만 편집할 수 있습니다. 여러 OpsItems를 편집한 경우 OpsCenter는 새로운 상태, 우선순위, 심각도 또는 범주를 선택하는 즉시 변경 사항을 업데이트하고 저장합니다.
4. OpsItem details(OpsItem 세부 정보) 섹션에서 편집을 선택합니다.
5. 조직에서 지정한 요구 사항 및 지침에 따라 OpsItem의 세부 정보를 편집합니다.
6. 작업을 마쳤으면 저장을 선택합니다.

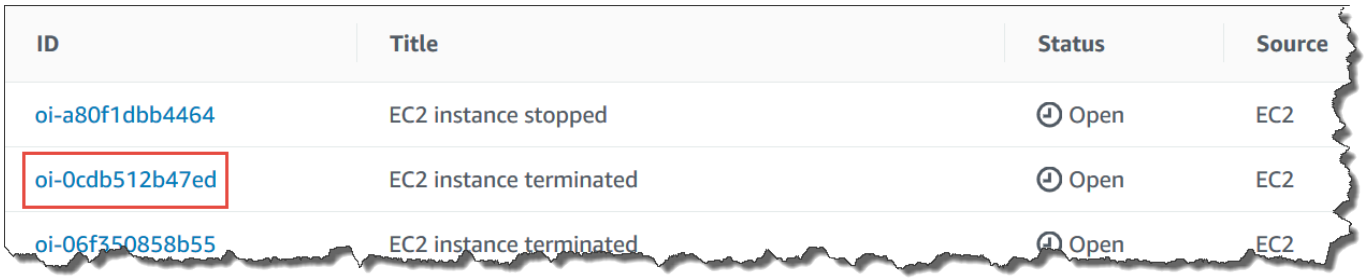
OpsItem에 관련 리소스 추가

각 OpsItem에는 관련 리소스의 Amazon 리소스 이름(ARN)이 나열된 관련 리소스 섹션이 있습니다. 관련 리소스는 조사가 필요한, 영향을 받는 AWS 리소스입니다.

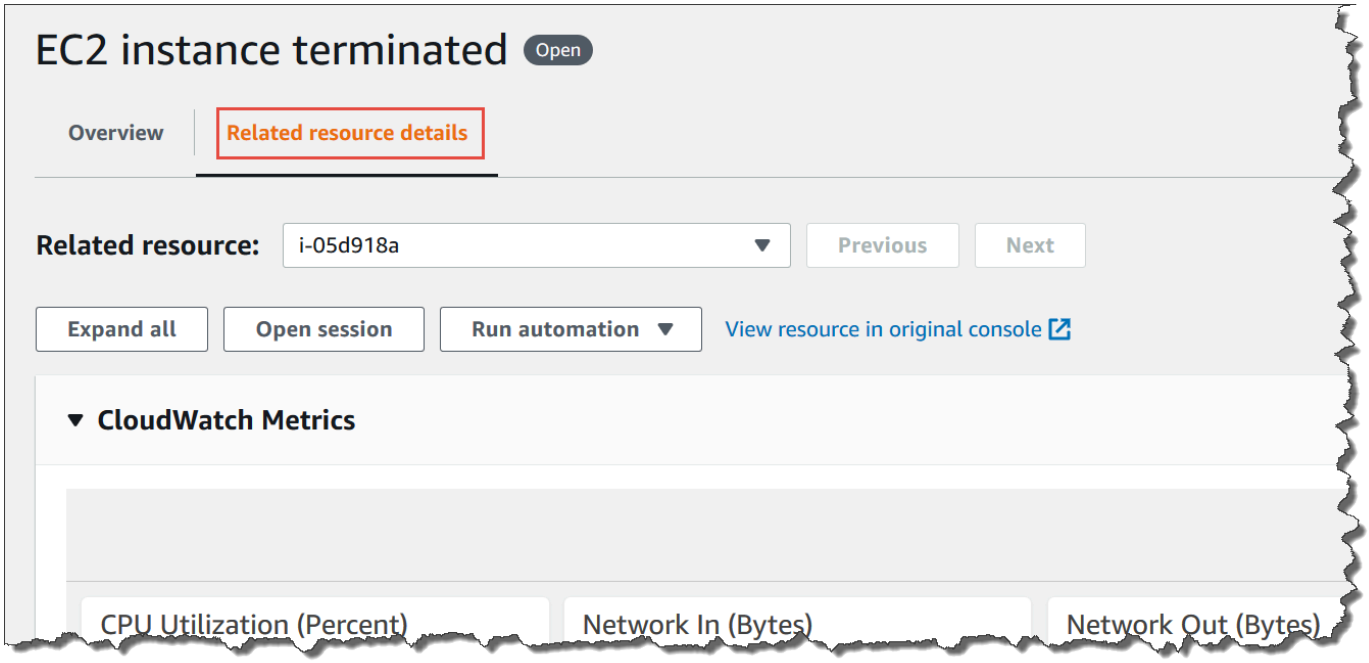
Amazon EventBridge가 OpsItem을 생성하면 시스템에서 자동으로 OpsItem을 리소스의 ARN으로 채웁니다. 관련 리소스의 ARN을 수동으로 지정할 수 있습니다. 일부 ARN 유형은 리소스에 대한 세부 정보를 OpsCenter 콘솔에 직접 표시하는 딥 링크가 OpsCenter에서 자동으로 생성됩니다. 예를 들어 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스의 ARN을 관련 리소스로 지정하면 OpsCenter에서는 해당 EC2 인스턴스에 대한 세부 정보를 가져옵니다. 따라서 OpsCenter를 종료하지 않고도 영향을 받는 AWS 리소스에 대한 자세한 정보를 볼 수 있습니다.

관련 리소스를 보고 OpsItem에 추가

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 OpsCenter를 선택합니다.
3. OpsItems 탭을 선택합니다.
4. OpsItem ID를 선택합니다.



5. 영향을 받는 리소스에 대한 정보를 보려면 관련 리소스 세부 정보(Related resources details) 탭을 선택합니다.



이 탭에는 여러 AWS 서비스의 리소스에 대한 정보가 표시됩니다. Resource details(리소스 세부 정보) 섹션을 확장하여 이 리소스를 호스팅하는 AWS 서비스에서 제공하는 이 리소스 관련 정보를 볼 수 있습니다. [관련 리소스(Related resources)] 목록을 사용하여 이 OpsItem과 연결된 다른 관련 리소스를 전환할 수도 있습니다.

6. 관련 리소스를 추가하려면 개요 탭을 선택합니다.
7. 관련 리소스 섹션에서 추가를 선택합니다.
8. [리소스 유형(Resource type)] 목록에서 리소스를 선택합니다.
9. [리소스 ID(Resource ID)]에 ID 또는 Amazon 리소스 이름(ARN)을 입력합니다. 선택하는 정보 유형은 이전 단계에서 선택한 리소스에 따라 다릅니다.

> **Note**
>
> 관련 추가 리소스의 ARN을 수동으로 추가할 수 있습니다. 각 OpsItem은 최대 100개의 관련 리소스 ARN을 나열할 수 있습니다.

다음 표에는 관련 리소스에 대한 딥 링크를 자동으로 생성하는 리소스 유형이 나열되어 있습니다.

지원되는 리소스 유형

리소스 이름	ARN 형식
AWS Certificate Manager 인증서	`arn:aws:acm: region:account-id :certificate/ certificate-id`
Amazon EC2 Auto Scaling 그룹	`arn:aws:autoscaling: region:account-id :autoScalingGroup: groupid:autoScalingGroupName/ groupfriendlyname`
Amazon CloudFront 배포	`arn:aws:cloudfront:: account-id :*`
AWS CloudFormation 스택	`arn:aws:cloudformation: region:account-id :stack/stackname /additionalidentifier`
Amazon CloudWatch 경보	`arn:aws:cloudwatch: region:account-id :alarm:alarm-name`
AWS CloudTrail 추적	`arn:aws:cloudtrail: region:account-id :trail/trailname`
AWS CodeBuild 프로젝트	`arn:aws:codebuild: region:account-id :resourcetype /resource`
AWS CodePipeline	`arn:aws:codepipeline: region:account-id :resource-specifier`
Amazon DevOps Guru 인사이트	`arn:aws:devops-guru: region:account-id :insight/ proactive or reactive/resource-id`

리소스 이름	ARN 형식
Amazon DynamoDB 테이블	`arn:aws:dynamodb: region:account-id :table/tablename`
Amazon Elastic Compute Cloud(Amazon EC2) 고객 게이트웨이	`arn:aws:ec2: region:account-id :customer-gateway/ cgw-id`
Amazon EC2 탄력적 IP	`arn:aws:ec2: region:account-id :eip/eipalloc-id`
Amazon EC2 전용 호스트	`arn:aws:ec2: region:account-id :dedicated-host/ host-id`
Amazon EC2 인스턴스	`arn:aws:ec2: region:account-id :instance/ instance-id`
Amazon EC2 인터넷 게이트웨이	`arn:aws:ec2: region:account-id :internet-gateway/ igw-id`
Amazon EC2 네트워크 액세스 제어 목록(네트워크 ACL)	`arn:aws:ec2: region:account-id :network-acl/ nacl-id`
Amazon EC2 네트워크 인터페이스	`arn:aws:ec2: region:account-id :network-interface/ eni-id`
Amazon EC2 라우팅 테이블	`arn:aws:ec2: region:account-id :route-table/ route-table-id`
Amazon EC2 보안 그룹	`arn:aws:ec2: region:account-id :security-group/ security-group-id`

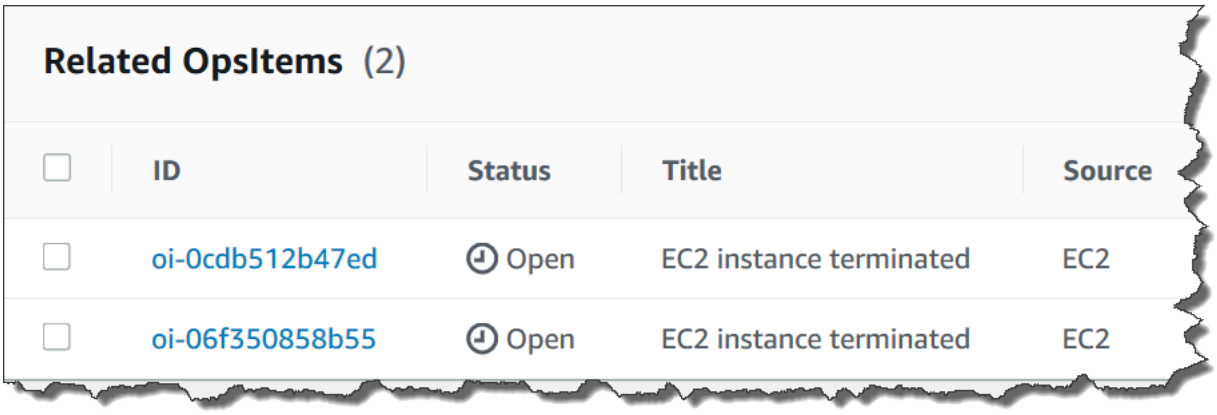
리소스 이름	ARN 형식
Amazon EC2 서브넷	`arn:aws:ec2: region:account-id :subnet/subnet-id`
Amazon EC2 볼륨	`arn:aws:ec2: region:account-id :volume/volume-id`
Amazon EC2 VPC	`arn:aws:ec2: region:account-id :vpc/vpc-id`
Amazon EC2 VPN 연결	`arn:aws:ec2: region:account-id :vpn-connection/ vpn-id`
Amazon EC2 VPN 게이트웨이	`arn:aws:ec2: region:account-id :vpn-gateway/ vgw-id`
AWS Elastic Beanstalk 애플리케이션	`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`

리소스 이름	ARN 형식
AWS Identity and Access Management(IAM) 그룹	`arn:aws:iam:: account-id :group/group-name`
IAM 정책	`arn:aws:iam:: account-id :policy/policy-name`
IAM 역할	`arn:aws:iam:: account-id :role/role-name`
IAM 사용자	`arn:aws:iam:: account-id :user/user-name`
AWS Lambda 함수	`arn:aws:lambda: region:account-id :function: function-name`
Amazon Relational Database Service(Amazon RDS) 클러스터	`arn:aws:rds: region:account-id :cluster: db-cluster-name`
Amazon RDS 데이터베이스 인스턴스	`arn:aws:rds: region:account-id :db:db-instance-name`
Amazon RDS 구독	`arn:aws:rds: region:account-id :es:subscription-name`
Amazon RDS 보안 그룹	`arn:aws:rds: region:account-id :secgrp:security-group-name`
Amazon RDS 클러스터 스냅샷	`arn:aws:rds: region:account-id :cluster-snapshot:  cluster-snapshot-name`

리소스 이름	ARN 형식
Amazon RDS 서브넷 그룹	`arn:aws:rds: region:account-id :subgrp:subnet-group-name`
Amazon Redshift 클러스터	`arn:aws:redshift: region:account-id :cluster: cluster-name`
Amazon Redshift 파라미터 그룹	`arn:aws:redshift: region:account-id :parametergroup: parameter-group-name`
Amazon Redshift 보안 그룹	`arn:aws:redshift: region:account-id :securitygroup: security-group-name`
Amazon Redshift 클러스터 스냅샷	`arn:aws:redshift: region:account-id :snapshot: cluster-name /snapshot-name`
Amazon Redshift 서브넷 그룹	`arn:aws:redshift: region:account-id :subnetgroup: subnet-group-name`
Amazon Simple Storage Service(Amazon S3) 버킷	`arn:aws:s3::: bucket_name`
AWS Systems Manager 관리형 노드 인벤토리의 AWS Config 레코딩	`arn:aws:ssm: region:account-id :managed-instance-inventory / node_id`
Systems Manager State Manager 연결	`arn:aws:ssm: region:account-id :association/ association_ID`

OpsItem에 관련 OpsItems 추가

OpsItems 세부 정보 페이지의 관련 OpsItems를 사용하여 운영 문제를 조사하고 문제에 대한 컨텍스트를 제공할 수 있습니다. OpsItems는 OpsItems 간의 상위-하위 관계, 근본 원인 또는 중복 항목을 포함하여 다양한 방식으로 관련될 수 있습니다. 하나의 OpsItem을 다른 과 연결하여 관련 OpsItem 섹션에 연결할 수 있습니다. 현재 OpsItem과 관련된 다른 OpsItems에 대해 최대 10개의 ID를 지정할 수 있습니다.

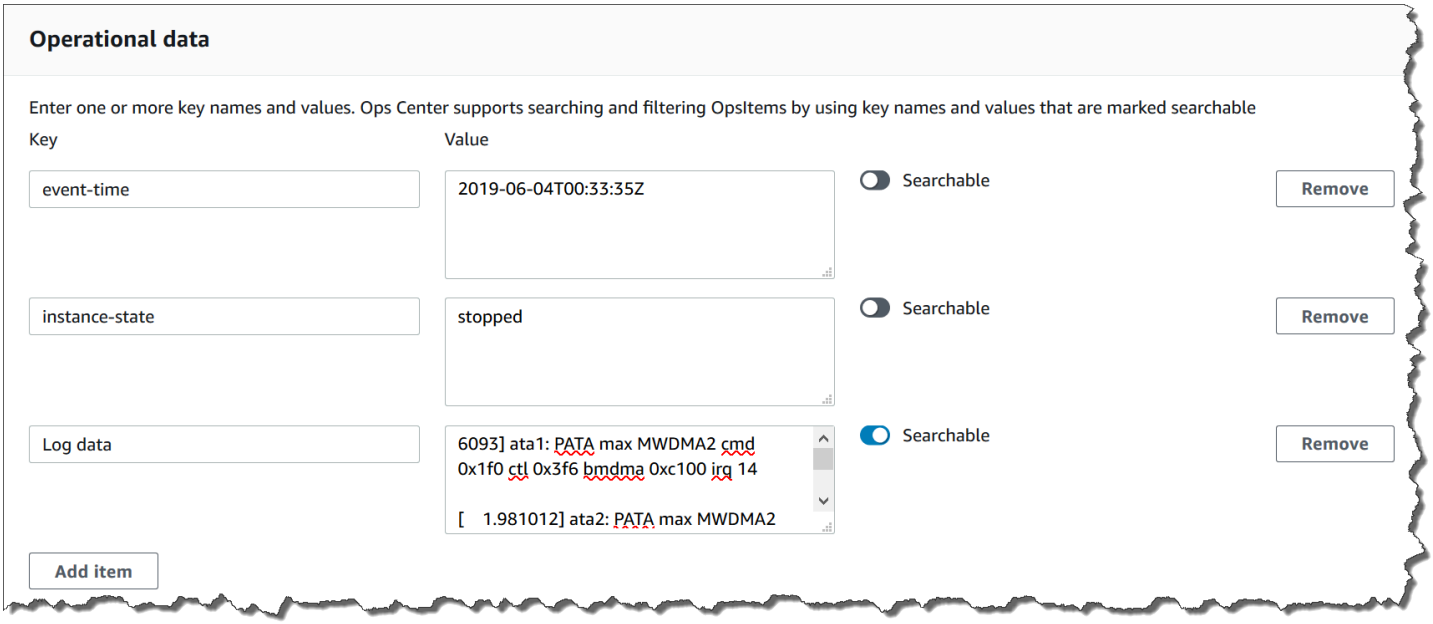


관련 OpsItem을 추가하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 OpsCenter를 선택합니다.
3. 세부 정보 페이지를 열려면 OpsItem ID를 선택합니다.
4. Related OpsItem(관련된 OpsItem) 섹션에서 추가를 선택합니다.
5. OpsItem ID(OpsItem ID)에서 ID를 지정합니다.
6. 추가를 선택합니다.

OpsItem에 운영 데이터 추가

운영 데이터는 OpsItem에 대한 유용한 참조 세부 정보를 제공하는 사용자 지정 데이터입니다. 운영 데이터의 여러 키-값 페어를 입력할 수 있습니다. 예를 들어 로그 파일, 오류 문자열, 라이선스 키, 문제 해결 팁 또는 기타 관련 데이터를 지정할 수 있습니다. 키의 최대 길이는 128자이고, 값의 최대 크기는 20KB일 수 있습니다.



계정의 다른 사용자가 데이터를 검색할 수 있도록 하거나 검색 액세스를 제한할 수 있습니다. 검색 가능한 데이터는 OpsItem 개요 페이지(DescribeOpsItems API 작업에서 제공)에 액세스할 수 있는 모든 사용자가 지정된 데이터를 보고 검색할 수 있음을 의미합니다. 검색할 수 없는 운영 데이터는 OpsItem(GetOpsItem API 작업에서 제공)에 액세스할 수 있는 사용자만 볼 수 있습니다.

OpsItem에 운영 데이터를 추가하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 OpsCenter를 선택합니다.
3. OpsItem ID를 선택하여 해당 세부 정보 페이지를 엽니다.
4. 운영 데이터를 확장합니다.
5. OpsItem에 대한 운영 데이터가 없으면 추가를 선택합니다. OpsItem에 대한 운영 데이터가 있으면 관리를 선택합니다.

   운영 데이터를 생성한 후에는 관리를 선택하여 키와 값을 편집하거나, 운영 데이터를 제거하거나, 키-값 페어를 추가할 수 있습니다.
6. 키에서 사용자가 데이터의 목적을 이해하는 데 도움이 되는 단어를 지정합니다.

> **Important**
>
> 운영 데이터 키는 `amazon`, `aws`, `amzn`, `ssm`, `/amazon`, `/aws`, `/amzn`, `/ssm`으로 시작할 수 없습니다.

7. 값에서 데이터를 지정합니다.
8. Save(저장)를 선택합니다.

> **Note**
>
> OpsItems 페이지의 운영 데이터 연산자를 사용하여 OpsItems를 필터링할 수 있습니다. 검색 상자에서 운영 데이터를 선택한 다음 JSON에 키-값 페어를 입력합니다. 키-값 페어는 `{"key":"key_name","value":"a_value"}` 형식을 사용하여 입력해야 합니다.

OpsItem에 대한 인시던트 생성

다음 절차에 따라 AWS Systems Manager의 기능인 AWS Systems Manager Incident Manager에서 OpsItem이 추적하고 관리할 인시던트를 수동으로 생성합니다. 인시던트는 서비스 품질의 계획되지 않은 중단 또는 감소입니다. Incident Manager에 대한 자세한 내용은 the section called “다른 AWS 서비스와 OpsCenter 통합” 섹션을 참조하세요.

OpsItem에 대한 인시던트를 수동으로 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 OpsCenter를 선택합니다.
3. Incident Manager가 OpsItem을 생성한 경우 이를 선택하고 5단계로 이동합니다. 그렇지 않은 경우 [OpsItem 생성(Create OpsItem)] 양식을 선택하고 작성합니다. 이 버튼이 표시되지 않으면 [OpsItems] 탭을 선택한 다음 [OpsItem 생성(Create OpsItem)]을 선택합니다.
4. OpsItem을 생성한 경우 이를 엽니다.
5. [인시던트 시작(Start Incident)]을 선택합니다.
6. 대응 계획에서 이 인시던트에 할당할 Incident Manager 대응 계획을 선택합니다.
7. (옵션) [제목(Title)]에 다른 팀원이 인시던트의 특성을 이해하는 데 도움이 되는 설명이 포함된 이름을 입력합니다. 새 제목을 입력하지 않으면 OpsCenter가 대응 계획의 제목을 사용하여 Incident Manager에서 OpsItem과 해당 인시던트를 생성합니다.
8. (옵션) [인시던트 영향(Incident impact)]에서 이 인시던트에 대한 영향 수준을 선택합니다. 영향 수준을 선택하지 않으면 OpsCenter가 대응 계획의 영향 수준을 사용하여 Incident Manager에서 OpsItem과 해당 인시던트를 생성합니다.
9. 시작을 선택합니다.

중복 OpsItems 관리

OpsCenter는 여러 AWS 서비스에서 단일 소스에 대한 여러 중복 OpsItems를 수신할 수 있습니다. OpsCenter는 중복 OpsItems이 생성되지 않도록 기본 제공 로직과 구성 가능한 중복 제거 문자열을 함께 사용합니다. AWS Systems Manager는 CreateOpsItem API 작업이 호출될 때 중복 제거 기본 제공 로직을 적용합니다.

AWS Systems Manager는 다음 중복 제거 로직을 사용합니다.

1. OpsItem을 생성할 때 Systems Manager는 중복 제거 문자열과 OpsItem을 시작한 리소스를 기반으로 해시를 생성하고 저장합니다.
2. 또 다른 OpsItem 생성 요청이 있는 경우 시스템은 새 요청의 중복 제거 문자열을 확인합니다.
3. 이 중복 제거 문자열에 대해 일치하는 해시가 있으면 Systems Manager는 기존 OpsItem의 상태를 확인합니다. 기존 OpsItem의 상태가 미결 또는 진행 중이면 OpsItem이 생성되지 않습니다. 기존 OpsItem이 해결되면 Systems Manager에서 새 OpsItem을 생성합니다.

OpsItem을 생성한 후에는 해당 OpsItem의 중복 제거 문자열을 편집하거나 변경할 수 없습니다.

중복 OpsItems를 관리하려면 다음 작업을 수행하세요.

- OpsCenter를 대상으로 하는 Amazon EventBridge 규칙의 중복 제거 문자열을 편집합니다. 자세한 내용은 기본 EventBridge 규칙에서 중복 제거 문자열 편집 단원을 참조하십시오.
- 수동으로 새 OpsItem을 생성할 때 중복 제거 문자열을 지정합니다. 자세한 내용은 AWS CLI를 사용하여 중복 제거 문자열 지정 단원을 참조하십시오.
- 운영 인사이트를 사용하여 중복 OpsItems를 검토하고 해결합니다. 런북을 사용하여 중복 OpsItems를 해결할 수 있습니다.

  중복 OpsItems를 해결하고 소스에서 생성되는 OpsItems 수를 줄이는 데 도움이 되도록 Systems Manager는 자동화 런북을 제공합니다. 자세한 설명은 인사이트를 기반으로 중복 OpsItems 해결을 참조하세요.

기본 EventBridge 규칙에서 중복 제거 문자열 편집

OpsCenter를 대상으로 하는 EventBridge 규칙에 대한 중복 제거 문자열을 지정하려면 다음 절차를 사용합니다.

EventBridge 규칙의 중복 제거 문자열 편집

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/events/에서 Amazon EventBridge 콘솔을 엽니다.
2. 탐색 창에서 규칙을 선택합니다.
3. 규칙을 선택한 다음 [편집(Edit)]을 선택합니다.
4. Select target(s)(대상 선택) 페이지로 이동합니다.
5. Additional settings(추가 설정) 섹션에서 Configure input transformer(입력 변환기 구성)을 선택합니다.
6. Template(템플릿) 박스에서 편집할 `"operationalData": { "/aws/dedup"` JSON 항목 및 중복 제거 문자열을 찾습니다.

   EventBridge 규칙의 중복 제거 문자열 항목은 다음 JSON 형식을 사용합니다.

   ```
   "operationalData": { "/aws/dedup": {"type": "SearchableString","value":
    "{\"dedupString\":\"Words the system should use to check for duplicate
    OpsItems\"}"}}
   ```

   의 예는 다음과 같습니다.

   ```
   "operationalData": { "/aws/dedup": {"type": "SearchableString","value":
    "{\"dedupString\":\"SSMOpsCenter-EBS-volume-performance-issue\"}"}}
   ```
7. 중복 제거 문자열을 편집한 다음 확인을 선택합니다.
8. 다음을 선택합니다.
9. 다음을 선택합니다.
10. 규칙 업데이트를 선택합니다.

AWS CLI를 사용하여 중복 제거 문자열 지정

AWS Systems Manager 콘솔 또는 AWS CLI를 사용하여 수동으로 새 OpsItem을 생성할 때 중복 제거 문자열을 지정할 수 있습니다. 콘솔에서 OpsItem을 수동으로 생성할 때 중복 제거 문자열을 입력하는 방법에 대한 자세한 내용은 수동으로 OpsItems 만들기 섹션을 참조하세요. AWS CLI를 사용하는 경우 `OperationalData` 파라미터에 대한 복 제거 문자열을 입력할 수 있습니다. 파라미터 구문은 다음 예와 같이 JSON을 사용합니다.

```
--operational-data '{"/aws/dedup":{"Value":"{\"dedupString\": \"Words the system should
 use to check for duplicate OpsItems\"}","Type":"SearchableString"}}'
```

다음은 `disk full`의 중복 제거 문자열을 지정하는 예제 명령입니다.

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"
```

운영 인사이트를 분석하여 OpsItems 줄이기

OpsCenter 운영 인사이트에는 중복 OpsItems에 대한 정보가 표시됩니다. OpsCenter에서 계정의 OpsItems를 자동으로 분석하고 세 가지 인사이트 유형을 생성합니다. OpsCenter 요약 탭의 운영 인사이트 섹션에서 이 정보를 볼 수 있습니다.

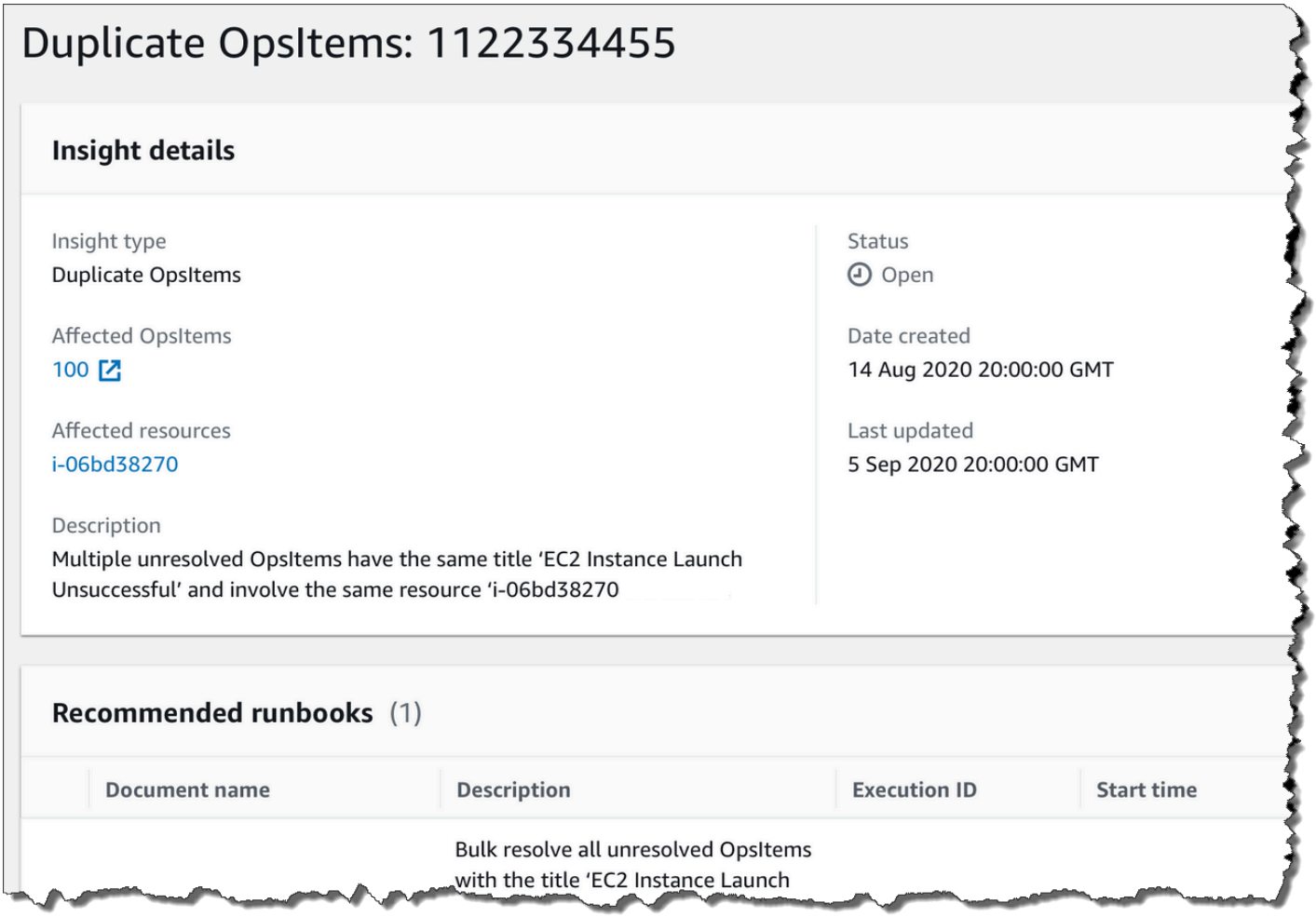
- OpsItems 복제 – 8개 이상의 OpsItems에 동일한 리소스에 대한 동일한 제목이 있으면 인사이트가 생성됩니다.
- 가장 일반적인 제목 – 50개를 초과하는 OpsItems에 동일한 제목이 있으면 인사이트가 생성됩니다.

- 대부분의 OpsItems가 생성되는 리소스 – AWS 리소스에 10개를 초과하는 미결 OpsItems가 있으면 인사이트가 생성됩니다. 이러한 인사이트와 해당 리소스는 OpsCenter 요약 탭의 OpsItems가 가장 많이 생성되는 리소스 테이블에 표시됩니다. OpsItem 수가 감소하는 순서로 리소스가 나열됩니다.

> **Note**
>
> OpsCenter에서는 다음과 같은 리소스 유형에 대한 대부분의 OpsItems가 생성되는 리소스 인사이트가 생성됩니다.
>
> - Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스
> - Amazon EC2 보안 그룹
> - Amazon EC2 Auto Scaling 그룹
> - Amazon Relational Database Service(RDS) 데이터베이스
> - Amazon RDS 클러스터
> - AWS Lambda 함수
> - Amazon DynamoDB 테이블
> - Elastic Load Balancing 로드 밸런서
> - Amazon Redshift 클러스터
> - AWS Certificate Manager 인증서
> - Amazon Elastic Block Store 볼륨

OpsCenter에서는 유형당 15개 인사이트 한도가 적용됩니다. 유형이 이 한도에 도달하면 해당 유형에 대한 추가 인사이트 표시가 OpsCenter에서 중지됩니다. 추가 인사이트를 보려면 해당 유형의 OpsInsight와 연관된 모든 OpsItems를 해결해야 합니다. 보류 중인 인사이트가 15개 인사이트 한도 때문에 콘솔에 표시되지 않는 경우 다른 인사이트를 닫으면 해당 인사이트가 표시됩니다.

인사이트를 선택하면 OpsCenter는 영향을 받는 OpsItems와 리소스에 대한 정보를 표시합니다. 다음 스크린샷은 중복 OpsItem 인사이트의 세부 정보가 포함된 예를 보여줍니다.



운영 인사이트는 기본적으로 꺼져 있습니다. 운영 인사이트 작업에 대한 자세한 내용은 다음 주제를 참조하세요.

주제

- 운영 인사이트 활성화
- 인사이트를 기반으로 중복 OpsItems 해결
- 운영 인사이트 사용 중지

운영 인사이트 활성화

Systems Manager 콘솔의 OpsCenter 페이지에서 운영 인사이트를 활성화할 수 있습니다. 운영 인사이트를 활성화하면 Systems Manager가 `AWSServiceRoleForAmazonSSM_OpsInsights`라는 AWS Identity and Access Management(IAM) 서비스 연결 역할을 생성합니다. 서비스 연결 역할은 Systems Manager에 직접 연결된 고유한 유형의 IAM 역할입니다. 서비스 연결 역할은 미리 정의되어 있으며 서비스가 사용자를 대신하여 다른 AWS 서비스를 호출하는 데 필요한 모든 권한을 포함합니다.

AWSServiceRoleForAmazonSSM_OpsInsights 서비스 연결 역할에 대한 자세한 정보는 역할을 사용하여 Systems Manager OpsCenter에서 운영 인사이트 OpsItems 생성 섹션을 참조하십시오.

> **Note**
>
> 다음과 같은 중요한 정보를 참고합니다.
>
> - AWS 계정에 운영 인사이트 요금이 부과됩니다. 자세한 내용은 AWS Systems Manager 요금을 참조하십시오.
> - OpsCenter는 배치 프로세스를 사용하여 인사이트를 주기적으로 새로 고칩니다. 이는 OpsCenter에 표시되는 인사이트 목록이 동기화되지 않았을 수 있음을 의미합니다.

OpsCenter의 운영 인사이트를 활성화하고 보려면 다음 절차를 사용합니다.

운영 인사이트 활성화 및 보기

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 OpsCenter를 선택합니다.
3. 운영 인사이트 사용 가능 메시지 상자에서 활성화를 선택합니다. 이 메시지가 표시되지 않으면 운영 인사이트 섹션까지 아래로 스크롤하여 활성화를 선택합니다.
4. 이 특성을 활성화한 후 요약 탭에서 운영 인사이트 섹션까지 아래로 스크롤합니다.
5. 필터링된 인사이트 목록을 보려면 중복 OpsItems, 가장 일반적인 제목 또는 대부분의 OpsItems가 생성되는 리소스 옆에 있는 링크를 선택합니다. 모든 인사이트를 보려면 [모든 운영 인사이트 보기(View all operational insights)]를 선택합니다.
6. 자세한 정보를 보려면 인사이트 ID를 선택합니다.

인사이트를 기반으로 중복 OpsItems 해결

인사이트을 해결하려면 먼저 인사이트와 연결된 모든 OpsItems를 해결해야 합니다. AWS-BulkResolveOpsItemsForInsight 실행서를 사용하여 인사이트와 연결된 OpsItems를 해결할 수 있습니다.

중복 OpsItems를 해결하고 소스에서 생성된 OpsItems 수를 줄이는 데 도움이 되도록 Systems Manager는 다음과 같은 자동화 런북을 제공합니다.

- AWS-BulkResolveOpsItems 실행서는 지정된 필터와 일치하는 OpsItems를 해결합니다.

- `AWS-AddOpsItemDedupStringToEventBridgeRule` 런북은 특정 Amazon EventBridge 규칙과 연결된 모든 OpsItem 대상에 대해 중복 제거 문자열을 추가합니다. 규칙에 이미 중복 제거 문자열이 있는 경우 이 런북은 중복 제거 문자열을 추가하지 않습니다.
- `AWS-DisableEventBridgeRule`은 규칙이 수십 또는 수백 개의 OpsItems을 생성하는 경우 EventBridge에서 규칙을 끕니다.

운영 인사이트를 해결하는 방법

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 OpsCenter를 선택합니다.
3. 개요(Overview) 탭에서 운영 인사이트(Operational insights)까지 아래로 스크롤합니다.
4. 모든 운영 인사이트 보기를 선택합니다.
5. 자세한 정보를 보려면 인사이트 ID를 선택합니다.
6. 런북과 실행을 선택을 차례로 선택합니다.

운영 인사이트 사용 중지

운영 인사이트를 끄면 시스템이 새 인사이트 생성을 중지하고 콘솔에 인사이트 표시를 중지합니다. 활성 인사이트는 시스템에 변경되지 않은 상태로 유지되지만 콘솔에 표시되지 않습니다. 이 기능을 다시 사용하면 이전에 해결되지 않은 인사이트가 표시되고 새 인사이트 생성이 시작됩니다. 다음 절차에 따라 운영 인사이트를 끕니다.

운영 인사이트 끄기

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 OpsCenter를 선택합니다.
3. 설정을 선택합니다.
4. [운영 인사이트(Operational insights)] 섹션에서 [편집(Edit)]을 선택한 다음 [사용 중지(Disable)] 옵션을 전환합니다.
5. Save(저장)를 선택합니다.

OpsCenter 로그 및 보고서 보기

AWS CloudTrail은 콘솔, AWS Command Line Interface(AWS CLI) 및 SDK에 대한 AWS Systems Manager OpsCenter API 호출을 기록합니다. 이 정보는 CloudTrail 콘솔 또는 Amazon Simple Storage

Service(Amazon S3) 버킷에서 볼 수 있습니다. Amazon S3는 계정에 대한 모든 CloudTrail 로그를 저장하는 데 하나의 버킷을 사용합니다.

OpsCenter 작업의 로그는 OpsItem 활동의 생성, 업데이트, 가져오기 및 설명을 표시합니다. Systems Manager 활동의 CloudTrail 로그 보기 및 사용에 대한 자세한 내용은 AWS CloudTrail을 사용하여 AWS Systems ManagerAPI 호출을 로깅 섹션을 참조하세요.

AWS Systems Manager OpsCenter는 OpsItems에 대한 다음 정보를 제공합니다.

- OpsItem 상태 요약 - 상태별 OpsItems 요약(미결 및 진행 중, 미결 또는 진행 중)을 제공합니다.
- 미결 OpsItems이 가장 많은 소스: 미결 OpsItems이 있는 상위 AWS 서비스에 대한 분석을 제공합니다.
- 소스 및 결과 시간별 OpsItems: 소스 및 생성 후 기간(일)에 따라 그룹화된 OpsItems 수를 제공합니다.

OpsCenter 요약 보고서 보기

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 OpsCenter를 선택합니다.
3. OpsItems 개요 페이지에서 요약을 선택합니다.
4. [소스 및 수명별 OpsItems(OpsItems by source and age)] 아래에서 검색 창을 선택하여 [소스(Source)]에 따라 OpsItems를 필터링합니다. 상태에 따라 필터링하려면 목록을 사용합니다.

OpsItems 삭제

AWS Command Line Interface 또는 AWS SDK를 사용하여 API 작업 삭제OpsItem를 호출하면 개인 OpsItem을(를) 삭제할 수 있습니다. AWS Management Console에서는 OpsItem을(를) 삭제할 수 없습니다. OpsItem을(를) 삭제하려면 AWS Identity and Access Management (IAM) 사용자, 그룹 또는 역할에 관리자 권한이 있거나 `DeleteOpsItem` API 작업을 호출할 수 있는 권한이 있어야 합니다.

> **Important**
>
> 이 운영에 대한 다음 중요 정보를 기록해 둡니다.
>
> - OpsItem을(를) 삭제하면 되돌릴 수 없습니다. OpsItem을(를) 복구할 수 없습니다.

- 이 작업은 최종 정합성 모델 을 사용하므로 시스템에서 이 작업을 완료하는 데 몇 분 정도 걸릴 수 있습니다. 예를 들어 OpsItem을(를) 삭제하고 즉시 호출하는 경우 삭제된 OpsItem은(는) 응답에 계속 표시되도록 할OpsItem 수 있습니다.
- 이 작업은 멱등성을 갖습니다. 동일한 OpsItem에 대해 이 작업을 반복해서 호출해도 시스템에서 예외가 발생하지 않습니다. 첫 번째 호출이 성공하면 모든 추가 호출은 첫 번째 호출과 동일한 성공적인 응답을 반환합니다.
- 이 작업은 교차 계정 호출을 지원하지 않습니다. 위임된 관리자 또는 관리 계정은 OpsCenter이(가) 교차 계정 관리를 위해 설정된 경우에도 다른 계정에서 OpsItems을(를) 삭제할 수 없습니다. 교차 계정 관리에 관한 자세한 내용은 (선택 사항) 여러 계정에서 OpsItems를 중앙 집중식으로 관리하도록 OpsCenter 설정을(를) 참조하세요.
- `OpsItemLimitExceededException`을(를) 받은 경우 하나 이상의 OpsItems을(를) 삭제하여 OpsItems의 총수를 할당량 한도 미만으로 줄일 수 있습니다. 예외에 대한 자세한 내용은 OpsCenter 문제 해결을(를) 참조하세요.

OpsItem 삭제

OpsItem을(를) 삭제하려면 다음 절차에 따르세요.

OpsItem을 삭제하려면

1. 아직 하지 않은 경우 AWS CLI을 설치하고 구성합니다. 자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.
2. 다음 명령을 실행합니다. *ID*를 삭제하고자 하는 OpsItem의 ID로 교체하세요.

```
aws ssm delete-ops-item --OpsItemId ID
```

성공한 경우 이 명령은 아무 데이터도 반환하지 않습니다.

OpsItem 문제 해결

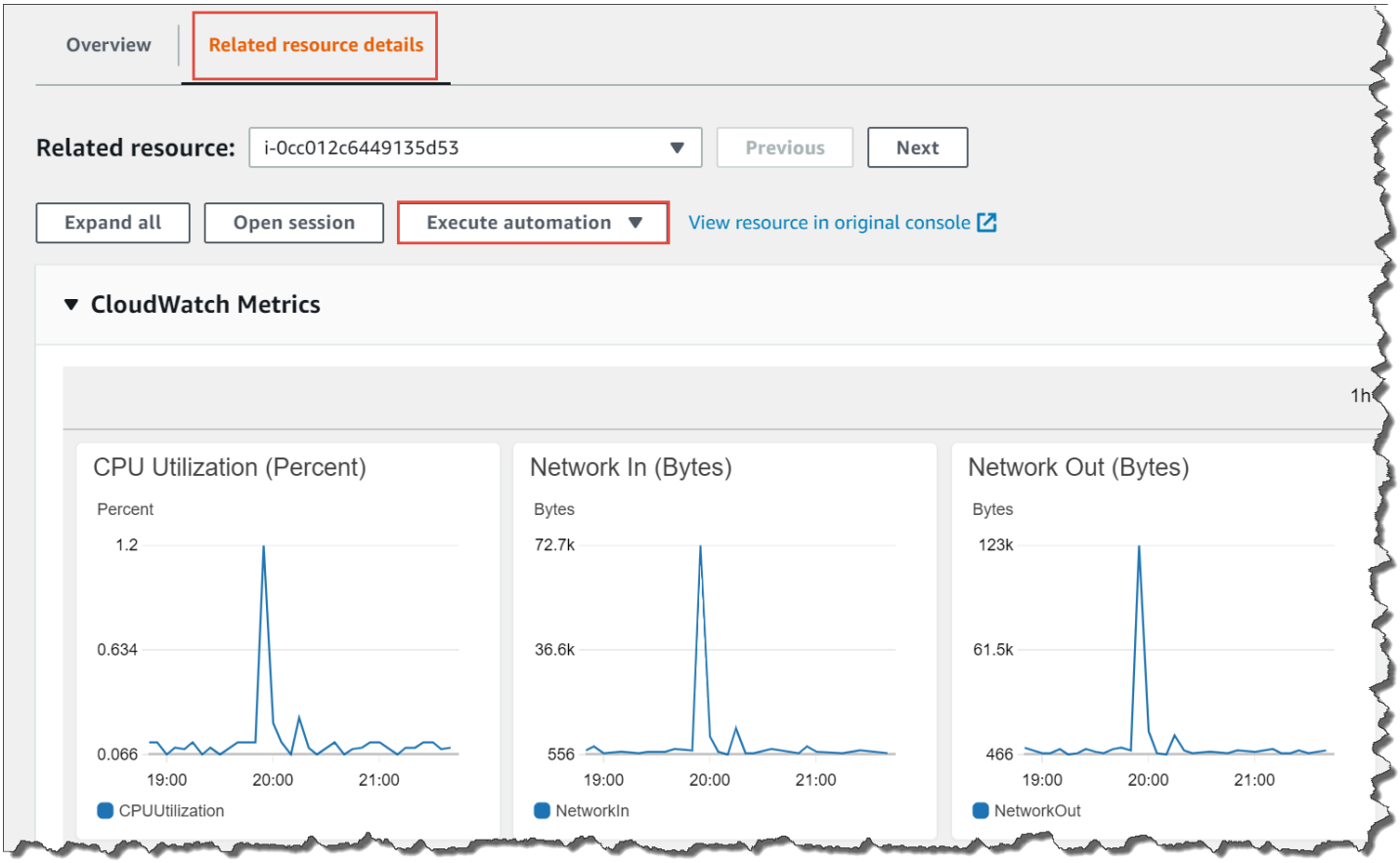
AWS Systems Manager Automation 런북을 사용하여 OpsItem에서 식별된 AWS 리소스 문제를 해결할 수 있습니다. Automation은 미리 정의된 런북을 사용하여 AWS 리소스와 관련된 일반적인 문제를 해결합니다.

각 OpsItem에는 수정에 사용할 수 있는 런북 목록을 제공하는 런북 섹션이 포함되어 있습니다. 목록에서 Automation 런북을 선택하면 OpsCenter가 문서를 실행하는 데 필요한 몇 가지 필드를 자동으

로 표시합니다. Automation 런북을 실행할 때 시스템은 런북을 OpsItem의 관련 리소스와 연결합니다. Amazon EventBridge가 OpsItem을 생성하면 런북을 OpsItem과 연결합니다. OpsCenter는 OpsItem에 대한 Automation 런북의 30일 레코드를 유지합니다.

상태를 선택하여 다음 예제와 같이 자동화가 실패한 이유 및 오류가 발생했을 때 실행 중인 Automation 런북의 단계와 같은 런북에 대한 중요 세부 정보를 볼 수 있습니다.

선택한 OpsItem에 대한 [관련 리소스 세부 정보(Related resource details)] 페이지에는 [자동화 실행(Run automation)] 목록이 있습니다. 최신 또는 리소스별 Automation 런북을 선택하여 실행하여 문제를 해결할 수 있습니다. 이 페이지에는 Amazon CloudWatch 지표 및 경보, AWS CloudTrail 로그, AWS Config의 세부 정보 등의 데이터 공급자도 포함되어 있습니다.



콘솔에서 실행서 이름을 선택하거나 Systems Manager Automation 실행서 참조를 사용하여 Automation 실행서에 대한 정보를 볼 수 있습니다.

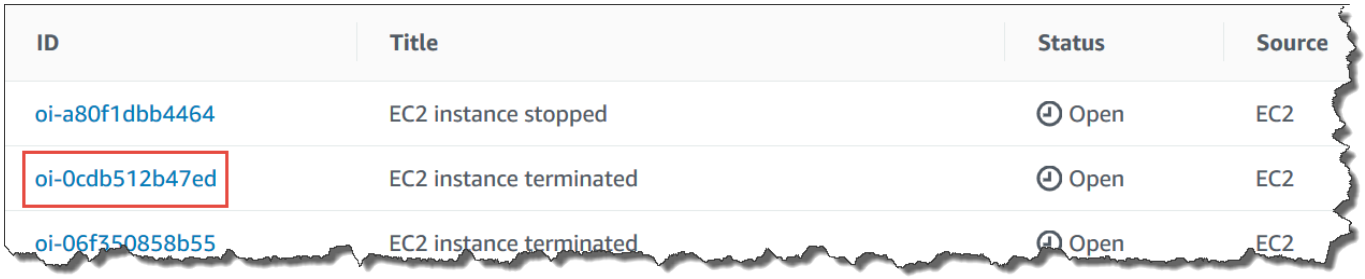
런북을 사용하여 OpsItem 수정

Automation 런북을 사용하여 OpsItem 문제를 해결하기 전에 다음을 수행합니다.

- Systems Manager Automation 실행서를 실행할 권한이 있는지 확인합니다. 자세한 내용은 Automation 설정 단원을 참조하십시오.
- 실행하려는 자동화에 대한 리소스별 ID 정보를 수집합니다. 예를 들어 EC2 인스턴스를 재시작하는 자동화를 실행하려면 다시 시작할 EC2 인스턴스의 ID를 지정해야 합니다.

Automation 실행서를 실행하여 OpsItem 문제를 해결하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 OpsCenter를 선택합니다.
3. OpsItem ID를 선택하여 세부 정보 페이지를 엽니다.



4. Runbooks(실행서) 섹션으로 스크롤합니다.
5. 런북 검색 창이나 오른쪽 상단의 숫자를 사용하여 실행할 Automation 런북을 찾습니다.
6. 실행서를 선택한 다음 실행을 선택합니다.
7. 런북에 필요한 정보를 입력한 다음에 제출을 선택합니다.

   런북을 시작하면 시스템이 이전 화면으로 돌아가서 상태가 표시됩니다.
8. 지난 30일 동안의 자동화 실행 섹션에서 실행 ID 링크를 선택하여 단계와 실행 상태를 조회합니다.

연결된 런북을 사용하여 OpsItem 수정

OpsItem에서 Automation 런북을 실행하면 OpsCenter는 런북을 OpsItem과 연결합니다. 연결된 런북은 런북 목록에서 다른 런북보다 순위가 높습니다.

다음 절차에 따라 OpsItem의 관련 리소스와 이미 연결된 Automation 실행서를 실행합니다. 관련 리소스 추가에 대한 자세한 내용은 OpsItems 관리 섹션을 참조하세요.

OpsItem 문제를 해결하기 위해 리소스와 연결된 실행서를 실행하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 OpsCenter를 선택합니다.
3. OpsItem을 엽니다.
4. [관련 리소스(Related resources)] 섹션에서 Automation 실행서를 실행할 리소스를 선택합니다.
5. [자동화 실행(Run automation)]을 선택한 다음 실행할 연결된 Automation 실행서를 선택합니다.
6. 실행서에 필요한 정보를 입력한 다음 실행을 선택합니다.

런북을 시작하면 시스템이 이전 화면으로 돌아가서 상태가 표시됩니다.

7. 지난 30일 동안의 자동화 실행 섹션에서 실행 ID 링크를 선택하여 단계와 실행 상태를 조회합니다.

OpsCenter 요약 보고서 보기

AWS Systems Manager OpsCenter에는 다음 정보가 자동으로 표시되는 요약 페이지가 있습니다.

- OpsItem 상태 요약 – 상태별 OpsItems 요약(예: `Open` 및 `In progress`)입니다.
- 미결 OpsItems가 가장 많은 소스 – 미결 OpsItems가 있는 상위 AWS 서비스 분석입니다.
- 소스 및 결과 시간별 OpsItems – 소스 및 생성 후 기간(일)에 따라 그룹화된 OpsItems 수입니다.

OpsCenter 요약 보고서를 보는 방법

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 해당하는 OpsCenter를 선택한 다음에 요약 탭을 선택합니다.
3. 소스 및 경과 시간별 OpsItems 섹션에서 다음을 수행합니다.
   1. (선택 사항) 필터 필드에서 소스를 선택하고 `Equal`, `Begin With` 또는 `Not Equal`을 선택한 다음에 검색 파라미터를 입력합니다.
   2. 인접한 목록에서 다음과 같은 상태 값 중 하나를 선택합니다.
      - `Open`
      - `In progress`
      - `Resolved`
      - `Open and in progress`
      - `All`

OpsCenter 문제 해결

이 항목에는 일반적인 오류 및 OpsCenter이(가) 포함된 문제를 해결하는 데 도움이 되는 정보가 포함되어 있습니다.

OpsItemLimitExceededException을(를) 받습니다

CreateOpsItem API 작업을 호출할 때 허용되는 OpsItems의 최대 수에 AWS 계정이(가) 도달하면 `OpsItemLimitExceededException`을(를) 드립니다. OpsCenter은(는) 호출이 다음 할당량 중 OpsItems의 최대 수를 초과할 경우 예외를 반환합니다.

- 리전당 AWS 계정당 OpsItems의 총 개수(`Open` 및 `Resolved` OpsItems 포함): 500,000
- 월별 AWS 계정당 OpsItems의 최대 수: 10,000

이 할당량은 다음 사항을 제외한 소스에서 생성된 OpsItems에 적용됩니다.

- AWS Security Hub 조사 결과에 따라 OpsItems이(가) 생성됨
- OpsItems은(는) 인시던트 관리자 인시던트가 열릴 때 자동으로 생성됩니다

이러한 소스에서 생성한 OpsItems은(는) OpsItem 할당량에 포함되지 않지만, 각 OpsItem에 요금이 부과됩니다.

`OpsItemLimitExceededException`을(를) 받은 경우 할당량 미만으로 줄어들 때까지 OpsItems을(를) 수동으로 삭제하여 새로운 OpsItem을(를) 생성하는 것을 방지합니다. 다시 말하지만, Security Hub 조사 결과 또는 인시던트 관리자 인시던트에 대해 생성된 OpsItems을(를) 삭제해도 할당량이 적용되는 OpsItems의 총 횟수는 줄어들지 않습니다. 다른 소스에서 OpsItems을(를) 삭제해야 합니다. OpsItem을(를) 삭제하는 방법에 대한 자세한 정보는 OpsItems 삭제을(를) 참조하세요.

많은 수의 자동으로 생성된 OpsItems에 대한 AWS(으)로부터 큰 금액의 청구서를 받습니다

AWS Security Hub와(과) 통합을 구성한 경우 OpsCenter은(는) Security Hub 조사 결과를 위한 OpsItems을(를) 생성합니다. 통합을 구성할 때 Security Hub가 생성한 조사 결과의 수와 로그인한 계정에 따라 많은 검색 결과가 OpsCenter은(는) OpsItems을(를) 많이 생성할 수 있으며, 비용이 들 수 있습니다. Security Hub 조사 결과에서 생성된 OpsItems와(과) 관련된 더욱 구체적인 세부 정보는 다음과 같습니다.

- OpsCenter 및 Security Hub 통합을 구성할 때 Security Hub 관리자 계정으로 로그인한 경우 시스템에서는 관리자 및 모든 멤버 계정의 조사 결과에 대한 OpsItems가 생성됩니다. OpsItems는 모두 관리자 계정에서 생성됩니다. 다양한 요인에 따라 예상외로 많은 요금이 AWS에서 청구될 수 있습니다.

통합을 구성할 때 멤버 계정에 로그인한 경우 시스템에서는 해당 개별 계정의 조사 결과에 대한 OpsItems만 생성됩니다. Security Hub 관리자 계정, 멤버 계정, 조사 결과에 대한 EventBridge 이벤트 피드 관련성에 대한 자세한 내용은 AWS Security Hub 사용 설명서의 EventBridge와 Security Hub 통합 유형을 참조하세요.

- OpsItem이 생성되는 조사 결과마다 정상 OpsItem 생성 가격이 부과됩니다. OpsItem을(를) 편집하거나 Security Hub에서 해당 조사 결과가 업데이트된 경우(OpsItem 업데이트가 트리거된 경우)에도 요금이 부과됩니다.

> **Important**
>
> 많은 수의 OpsItems이(가) 실수로 생성되었고 AWS 청구서가 부적절한 것으로 판단되는 경우 AWS Support에 문의해 주세요.

시스템에서 Security Hub 조사 결과의 OpsItems가 더는 생성되지 않도록 하려면 다음 절차를 사용합니다.

Security Hub 조사 결과의 OpsItems 수신을 중지하는 방법

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 OpsCenter를 선택합니다.
3. 설정을 선택합니다.
4. Security Hub 조사 결과 섹션에서 편집을 선택합니다.
5. 슬라이더를 선택하여 활성화됨을 비활성화됨으로 변경합니다. 슬라이더를 전환할 수 없으면 Security Hub가 AWS 계정에 대해 활성화되지 않은 것입니다.
6. 저장을 선택하여 구성을 저장합니다.OpsCenter에서 더는 Security Hub 조사 결과에 따라 OpsItems가 생성되지 않습니다.

> **Important**
>
> OpsCenter이(가) 설정을 다시 활성화로 전환하고 조사 결과에 대한 OpsItems 생성을 계속하는 경우 시스템 관리자가 위임한 관리자 계정 또는 AWS Organizations 관리 계정에 로그인하고 이 절차를 반복합니다. 이러한 계정 중 하나에 로그인할 권한이 없는 경우 관리자에게 문의하고 이 절차를 반복하여 계정에 대한 통합을 비활성화해달라고 요청하세요.

Systems Manager에서 호스팅하는 Amazon CloudWatch 대시보드

Amazon CloudWatch 대시보드는 CloudWatch 콘솔에서 사용자 지정이 가능한 홈 페이지로, 다른 AWS 리전에 분산되어 있는 리소스들을 비롯하여 단일 보기에서 리소스를 모니터링하는 데 사용할 수 있습니다. CloudWatch 대시보드를 사용해 AWS 리소스에 대한 지표 및 경보를 보여주는 사용자 정의 뷰를 생성할 수 있습니다. 대시보드에서 다음을 생성할 수 있습니다.

- 선택한 지표 및 경보에 대한 단일 뷰를 생성하여 하나 이상의 AWS 리전에서 리소스 및 애플리케이션의 상태를 평가할 수 있습니다. 여러 그래프에서 동일한 지표를 추적할 수 있도록 각 그래프에서 지표 각각에 사용되는 색상을 선택할 수 있습니다.
- 작동 지침서를 생성하여 운영 이벤트 동안 팀원들에게 특정 사고에 대한 대응 방법에 관해 지침을 제공할 수 있습니다.
- 중요한 리소스 및 애플리케이션 지표에 대한 공통 뷰를 생성하여 운영 이벤트 동안 신속하고 원활한 의사 소통을 위해 팀원들이 이를 공유하도록 할 수 있습니다.

콘솔, AWS Command Line Interface(AWS CLI) 또는 CloudWatch `PutDashboard` API를 사용하여 대시보드를 생성할 수 있습니다. 자세한 내용은 Amazon CloudWatch 사용 설명서의 Amazon CloudWatch 대시보드 사용을 참조하세요.

AWS Systems Manager 애플리케이션 관리

애플리케이션 관리는 AWS에서 실행되는 애플리케이션을 관리하는 데 도움이 되는 기능 모음입니다.

주제

- AWS Systems Manager Application Manager
- AWS AppConfig
- AWS Systems Manager Parameter Store

AWS Systems Manager Application Manager

AWS Systems Manager의 기능인 Application Manager은(는) DevOps 엔지니어가 애플리케이션과 클러스터의 컨텍스트에서 AWS 리소스의 문제를 조사하고 수정하는 데 도움이 됩니다. Application Manager은(는) 여러 AWS 서비스 및 Systems Manager 기능의 작업 정보를 하나의 AWS Management Console(으)로 집계합니다.

Application Manager에서 애플리케이션은 하나의 단위로 작동하려는 AWS 리소스의 논리적 그룹입니다. 이 논리 그룹은 몇 가지 예를 들면 다양한 버전의 애플리케이션, 운영자의 소유권 경계 또는 개발자 환경을 나타낼 수 있습니다. Application Manager의 컨테이너 클러스터 지원에는 Amazon Elastic Kubernetes Service(Amazon EKS) 및 Amazon Elastic Container Service(Amazon ECS) 클러스터가 모두 포함됩니다.

Application Manager 홈 페이지에서 Get started(시작하기)를 선택하면 Application Manager은(는) 다른 AWS 서비스 또는 Systems Manager 기능에서 생성된 리소스에 대한 메타데이터를 자동으로 가져옵니다. 애플리케이션의 경우 Application Manager는 리소스 그룹으로 구성된 모든 AWS 리소스에 대한 메타데이터를 가져옵니다. 각 리소스 그룹은 사용자 정의 애플리케이션(Custom applications) 범주에 고유한 애플리케이션으로 나열됩니다. Application Manager는 AWS CloudFormation, AWS Launch Wizard, Amazon ECS 및 Amazon EKS에서 생성한 리소스에 대한 메타데이터도 자동으로 가져옵니다. 그런 다음 Application Manager가 미리 정의된 범주에 해당 리소스를 표시합니다.

애플리케이션(Applications)의 경우 목록에 다음이 포함됩니다.

- 사용자 정의 애플리케이션
- Launch Wizard
- CloudFormation 스택

- AppRegistry 애플리케이션

컨테이너 클러스터(Container clusters)의 경우 목록에 다음이 포함됩니다.

- Amazon ECS 클러스터
- Amazon EKS 클러스터

가져오기가 완료되면 이러한 미리 정의된 범주에서 리소스에 대한 작업 정보를 볼 수 있습니다. 또는 리소스 컬렉션에 대한 추가 컨텍스트를 제공하려면 Application Manager에서 수동으로 애플리케이션을 생성하고 리소스 또는 리소스 그룹을 해당 애플리케이션으로 이동할 수 있습니다. 이를 통해 애플리케이션 컨텍스트에서 작업 정보를 볼 수 있습니다.

AWS 서비스 및 Systems Manager 기능을 설정하고 구성한 후 Application Manager은(는) 리소스에 대한 다음 유형의 정보를 표시합니다.

- 애플리케이션에서 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스의 현재 상태, Amazon EC2 Auto Scaling 상태에 대한 정보
- Amazon CloudWatch에서 제공하는 경보
- AWS Config 및 State Manager(Systems Manager의 구성 요소)에서 제공하는 Compliance 정보
- Amazon EKS에서 제공하는 Kubernetes 클러스터 정보
- AWS CloudTrail 및 Amazon CloudWatch Logs에서 제공하는 로그 데이터
- Systems Manager OpsCenter에서 제공하는 OpsItems
- 이를 호스팅하는 AWS 서비스에서 제공하는 리소스 세부 정보.
- Amazon ECS에서 제공하는 컨테이너 클러스터 정보.

구성 요소 또는 리소스와 관련된 문제를 해결하는 데 도움이 되도록 Application Manager는 애플리케이션과 연결할 수 있는 실행서도 제공합니다. Application Manager를 시작하려면 Systems Manager 콘솔을 엽니다. 탐색 창에서 Application Manager를 선택합니다.

Application Manager 사용의 이점은 무엇인가요?

Application Manager는 DevOps 엔지니어가 AWS 리소스와 관련된 문제를 감지하고 조사하는 데 걸리는 시간을 줄입니다. 이를 위해 Application Manager는 하나의 콘솔에서 애플리케이션의 컨텍스트에서 여러 유형의 작업 정보를 표시합니다. Application Manager는 또한 AWS 리소스에 대한 일반적인 수정 태스크를 수행하는 실행서를 제공하여 문제 해결에 걸리는 시간을 줄입니다.

Application Manager에는 어떤 기능이 있나요?

Application Manager에는 다음 기능이 포함됩니다.

- 자동으로 AWS 리소스 가져오기

  초기 설정 중 Application Manager가 CloudFormation 스택, AWS Resource Groups, Launch Wizard 배포, AppRegistry 애플리케이션, Amazon ECS 및 Amazon EKS 클러스터를 기반으로 AWS 계정의 리소스를 자동으로 가져오고 표시하도록 선택할 수 있습니다. 시스템은 이러한 리소스를 미리 정의된 애플리케이션 또는 클러스터 범주로 표시합니다. 그 후 이러한 유형의 새 리소스가 AWS 계정에 추가될 때마다 Application Manager는 미리 정의된 애플리케이션 및 클러스터 범주에 새 리소스를 자동으로 표시합니다.

- CloudFormation 스택 및 템플릿 생성 또는 편집

  Application Manager는 CloudFormation과 통합하여 애플리케이션에 대한 리소스를 프로비저닝하고 관리하는 데 도움이 됩니다. Application Manager에서는 AWS CloudFormation 템플릿과 스택을 생성, 수정, 삭제할 수 있습니다. Application Manager에는 템플릿을 복제, 생성 및 저장할 수 있는 템플릿 라이브러리도 포함되어 있습니다. Application Manager와 CloudFormation은 스택의 현재 상태에 대해 동일한 정보를 표시합니다. 템플릿 및 템플릿 업데이트는 스택을 프로비저닝할 때까지 Systems Manager에 저장되며, 이때 변경 사항은 CloudFormation에도 표시됩니다.

- 애플리케이션 컨텍스트에서 인스턴스에 대한 정보 보기

  Application Manager는 Amazon Elastic Compute Cloud(Amazon EC2)와 통합되어 애플리케이션의 컨텍스트에서 인스턴스에 대한 정보를 표시합니다. Application Manager는 선택한 애플리케이션의 인스턴스 상태, 상태 및 Amazon EC2 Auto Scaling 상태를 그래픽 형식으로 표시합니다. Instances(인스턴스) 탭에는 애플리케이션의 각 인스턴스에 대한 다음 정보가 포함된 테이블도 있습니다.

  - 인스턴스 상태(보류 중, 중지 중, 실행 중, 중지됨)
  - SSM Agent의 Ping 상태
  - 인스턴스에서 대해 마지막으로 처리된 Systems Manager Automation 런북의 상태 및 이름
  - 상태별 Amazon CloudWatch Logs 경보의 수입니다.
    - `ALARM` – 지표 또는 표현식이 정의된 임곗값을 벗어났습니다.
    - `OK` – 지표 또는 표현식이 정의된 임곗값 내에 있습니다.
    - `INSUFFICIENT_DATA` – 경보가 방금 시작되었거나 지표를 사용할 수 없거나 지표에서 경보 상태를 결정하는 데 사용할 수 있는 데이터가 충분하지 않습니다.
  - 상위 및 개별 오토 스케일링 그룹의 Auto Scaling 상태

- 애플리케이션 또는 클러스터에 대한 운영 지표 및 경보 보기

  Application Manager가 Amazon CloudWatch와 통합되어 애플리케이션 또는 클러스터에 대한 실시간 운영 지표 및 경보를 제공합니다. 애플리케이션 트리로 드릴다운하여 각 구성 요소 수준에서 경보를 보거나 개별 클러스터에 대한 경보를 볼 수 있습니다.

- 애플리케이션에 대한 로그 데이터 보기

  Application Manager는 Amazon CloudWatch Logs와 통합되어 Systems Manager를 종료하지 않고도 애플리케이션 컨텍스트에서 로그 데이터를 제공합니다.

- 애플리케이션 또는 클러스터에 대한 OpsItems 보기 및 관리

  Application Manager는 AWS Systems Manager OpsCenter와 통합되어 애플리케이션 및 클러스터에 대한 운영 작업 항목(OpsItems) 목록을 제공합니다. 목록은 자동 생성 및 수동 생성 OpsItems를 반영합니다. OpsItem을 생성한 리소스와 OpsItem 상태, 소스, 심각도에 대한 세부 정보를 볼 수 있습니다.

- 애플리케이션 또는 클러스터에 대한 리소스 규정 준수 데이터 보기

  Application Manager는 AWS Config와 통합되어 사용자가 지정한 규칙에 따라 AWS 리소스에 대한 규정 준수 및 기록 세부 정보를 제공합니다. Application Manager 또한 AWS Systems Manager State Manager와 통합되어 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에 대해 유지하려는 상태에 대한 규정 준수 정보를 제공합니다.

- Amazon ECS 및 Amazon EKS 클러스터 인프라 정보 보기

  Application Manager는 Amazon ECS 및 Amazon EKS와 통합되어 클러스터 인프라의 상태에 대한 정보와 클러스터의 컴퓨팅, 네트워킹 및 스토리지 리소스에 대한 구성 요소 런타임 뷰를 제공합니다.

  그러나 Application Manager에서 Amazon EKS 포드 또는 컨테이너에 대한 작업 정보를 관리하거나 볼 수 없습니다. Amazon EKS 리소스를 호스팅하는 인프라에 대한 작업 정보만 관리하고 볼 수 있습니다.

- 애플리케이션에 대한 리소스 비용 세부 정보 보기

  Application Manager는 Cost(비용) 위젯을 통해 AWS Billing and Cost Management의 기능인 AWS Cost Explorer와 통합됩니다. Billing and Cost Management 콘솔에서 Cost Explorer를 활성화하면, Application Manager의 Cost(비용) 위젯은 특정 비컨테이너 애플리케이션 또는 애플리케이션 구성 요소에 대한 비용 데이터를 보여줍니다. 위젯의 필터를 사용하여 막대 또는 꺾은선형 차트에서 다양한 기간, 세부 기간 및 비용 유형에 따라 비용 데이터를 볼 수 있습니다.

- 하나의 콘솔에서 자세한 리소스 정보 보기

Application Manager에 나열된 리소스 이름을 선택하고 Systems Manager를 종료하지 않고도 해당 리소스에 대한 컨텍스트 정보와 작업 정보를 봅니다.

- 애플리케이션에 대한 자동 리소스 업데이트 수신

  서비스 콘솔에서 리소스를 변경하고 해당 리소스가 Application Manager에 있는 애플리케이션의 일부인 경우 Systems Manager는 이러한 변경 사항을 자동으로 표시합니다. 예를 들어 AWS CloudFormation 콘솔에서 스택을 업데이트하고 해당 스택이 Application Manager 애플리케이션의 일부인 경우 스택 업데이트는 자동으로 Application Manager에 반영됩니다.

- Launch Wizard 애플리케이션 자동 검색

  Application Manager는 AWS Launch Wizard에 통합됩니다. Launch Wizard를 사용하여 애플리케이션에 대한 리소스를 배포한 경우 Application Manager가 자동으로 해당 리소스를 가져오고 Launch Wizard 섹션에 표시할 수 있습니다.

- CloudWatch Application Insights를 사용하여 Application Manager에서 애플리케이션 리소스 모니터링

  Application Manager는 Amazon CloudWatch Application Insights와 통합됩니다. Application Insights는 애플리케이션 리소스와 기술 스택 전반에 걸쳐 주요 지표, 로그, 경보를 식별하고 설정합니다. Application Insights는 지표 및 로그를 지속적으로 모니터링하여 이상 및 오류를 감지하고 연결합니다. 시스템이 오류 및 이상을 감지하면 Application Insights에서 알림을 설정하고 작업을 수행할 수 있는 CloudWatch Events를 생성합니다. Application Manager의 개요(Overview) 및 모니터링(Monitoring) 탭에서 Application Insights를 활성화하고 볼 수 있습니다. Application Insights에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 Amazon CloudWatch Application Insights란 무엇인가요?를 참조하세요.

- 실행서 관련 문제 해결

  Application Manager에는 AWS 리소스와 관련된 일반적인 문제를 해결하기 위해 미리 정의된 Systems Manager 실행서가 포함되어 있습니다. Application Manager를 나가지 않고도 애플리케이션의 모든 해당 리소스에 대해 런북을 실행할 수 있습니다.

Application Manager를 사용하는 데 비용이 듭니까?

Application Manager는 추가 요금 없이 사용할 수 있습니다.

Application Manager에 대한 리소스 할당량은 어떻게 되나요?

Amazon Web Services 일반 참조의 Systems Manager 서비스 할당량에서 모든 Systems Manager 기능의 할당량을 볼 수 있습니다. 다르게 표시되지 않는 한, 리전별로 각 할당량이 적용됩니다.

주제

- Systems Manager Application Manager 시작하기
- Application Manager 작업

Systems Manager Application Manager 시작하기

이 섹션의 정보를 사용하여 다양한 AWS 서비스 및 Systems Manager 기능의 작업 정보를 표시하도록 AWS Systems Manager의 기능인 Application Manager을(를) 설정하고 구성합니다. 이 섹션에는 Application Manager에 애플리케이션 및 클러스터 추가에 대한 정보가 포함되어 있습니다.

주제

- 관련 서비스 설정
- Systems Manager Application Manager에 대한 권한 구성
- Application Manager에 애플리케이션 및 클러스터 추가

관련 서비스 설정

AWS Systems Manager의 기능인 Application Manager은(는) 다른 AWS 서비스 및 Systems Manager 기능의 리소스 및 정보를 표시합니다. Application Manager에 표시되는 작업 정보의 양을 최대화하려면 Application Manager를 사용하기 전에 이러한 다른 서비스나 기능을 설정하고 구성하는 것이 좋습니다.

주제

- 리소스 가져오기 태스크 설정
- 리소스에 대한 작업 정보를 보기 위한 태스크 설정

리소스 가져오기 태스크 설정

다음 설정 태스크를 통해 AWS의 Application Manager 리소스를 볼 수 있습니다. 이러한 각 태스크가 완료된 후 Systems Manager는 자동으로 리소스를 Application Manager로 가져올 수 있습니다. 리소스

를 가져온 후 Application Manager에서 애플리케이션을 생성하고 가져온 리소스를 애플리케이션으로 이동할 수 있습니다. 이렇게 하면 애플리케이션 컨텍스트에서 작업 정보를 볼 수 있습니다.

(옵션) 태그를 사용하여 AWS 리소스 구성

AWS 리소스에 태그 형태로 메타데이터를 지정할 수 있습니다. 각 태그는 사용자 정의 키와 값으로 구성된 레이블입니다. 태그를 사용하면 리소스를 손쉽게 관리, 식별, 정리, 검색 및 필터링할 수 있습니다. 태그를 생성하여 용도, 소유자, 환경 또는 기타 기준으로 리소스를 분류할 수 있습니다.

(옵션) AWS Resource Groups를 사용하여 AWS 리소스 구성

리소스 그룹을 사용하여 AWS 리소스를 정리할 수 있습니다. 리소스 그룹을 사용하여 많은 리소스에 대한 태스크를 한 번에 관리, 모니터링 및 자동화할 수 있습니다.

Application Manager가 모든 리소스 그룹을 자동으로 가져와서 사용자 정의 애플리케이션(Custom applications) 범주에 나열합니다.

(옵션) AWS CloudFormation을 사용하여 AWS 리소스 설정 및 배포

AWS CloudFormation을 사용하면 AWS 인프라 배포를 예상한 대로 반복해서 생성 및 프로비저닝할 수 있습니다. Amazon EC2, Amazon Elastic Block Store(Amazon EBS), Amazon Simple Notification Service(Amazon SNS), Elastic Load Balancing, AWS Auto Scaling 등의 AWS 서비스을(를) 사용하는 데 도움이 됩니다. CloudFormation을 사용하면 기본 AWS 인프라 생성 및 구성에 대해 걱정할 필요 없이 클라우드에서 안정적이고 확장 가능하며 비용 효율적인 애플리케이션을 구축할 수 있습니다.

Application Manager가 모든 AWS CloudFormation 리소스를 자동으로 가져와서 AWS CloudFormation 스택 범주에 나열합니다. Application Manager에서 CloudFormation 스택 및 템플릿을 생성할 수 있습니다. 스택 및 템플릿 변경 사항은 Application Manager와 CloudFormation 간에 자동으로 동기화됩니다. Application Manager에 애플리케이션을 생성하고 해당 애플리케이션으로 스택을 이동할 수도 있습니다. 이렇게 하면 애플리케이션 컨텍스트에서 스택의 리소스에 대한 작업 정보를 볼 수 있습니다. 요금 정보는 AWS CloudFormation 요금을 참조하세요.

(옵션) AWS Launch Wizard를 사용하여 애플리케이션 설정 및 배포

Launch Wizard는 개별 AWS 리소스를 수동으로 식별하고 프로비저닝할 필요 없이 서드 파티 애플리케이션용 AWS 리소스의 크기 조정, 구성 및 배포 프로세스를 안내합니다.

Application Manager가 모든 Launch Wizard 리소스를 자동으로 가져와서 [Launch Wizard] 범주에 나열합니다. AWS Launch Wizard에 대한 자세한 내용은 SQL Server용 AWS Launch Wizard 시작하기를 참조하세요. Launch Wizard는 추가 요금 없이 사용할 수 있습니다. 솔루션을 실행하기 위해 프로비저닝하는 AWS 리소스에 대해서만 비용을 지불합니다.

(옵션) Amazon ECS 및 Amazon EKS를 사용하여 컨테이너화된 애플리케이션 설정 및 배포

Amazon Elastic Container Service(Amazon ECS)는 클러스터에서 컨테이너를 손쉽게 실행, 중지 및 관리할 수 있게 하는 컨테이너 관리 서비스로서 확장성과 속도가 뛰어납니다. 컨테이너는 서비스 내에서 개별 태스크나 여러 태스크를 실행하는 데 사용하는 태스크 정의에 정의됩니다.

Amazon EKS는 Kubernetes를 실행하는 데 사용할 수 있는 관리형 서비스입니다. AWS Kubernetes 제어 플레인 또는 노드를 설치, 작동 및 유지 관리할 필요가 없습니다. Kubernetes는 컨테이너화된 애플리케이션의 배포, 조정 및 관리 자동화를 위한 오픈 소스 시스템입니다.

Application Manager는 모든 Amazon ECS 및 Amazon EKS 인프라 리소스를 자동으로 가져와서 컨테이너 클러스터(Container clusters) 탭에 나열합니다. 그러나 Application Manager에서 Amazon EKS 포드 또는 컨테이너에 대한 작업 정보를 관리하거나 볼 수 없습니다. Amazon EKS 리소스를 호스팅하는 인프라에 대한 작업 정보만 관리하고 볼 수 있습니다. 요금 정보는 Amazon ECS 요금 및 Amazon EKS 요금을 참조하세요.

리소스에 대한 작업 정보를 보기 위한 태스크 설정

다음 설정 태스크를 통해 Application Manager에서 AWS 리소스에 대한 태스크 정보를 볼 수 있습니다.

(권장) 실행서 권한 확인

Systems Manager Automation 실행서를 사용하여 Application Manager에서 AWS 리소스 문제를 해결할 수 있습니다. 이 수정 기능을 사용하려면 권한을 구성하거나 확인해야 합니다. 요금 정보는 AWS Systems Manager 요금을 참조하세요.

(선택 사항) Cost Explorer 활성화

AWS Cost Explorer는 추가 분석을 위해 비용 데이터를 시각화하는 데 사용할 수 있는 AWS Cost Management의 특성입니다. Cost Explorer를 활성화하면 Application Manager 콘솔에서 애플리케이션의 리소스에 대한 비용 정보, 비용 내역 및 비용 최적화를 볼 수 있습니다.

(옵션) Amazon CloudWatch 로그 및 경보 설정 및 구성

CloudWatch는 AWS, 하이브리드 및 멀티클라우드 애플리케이션과 인프라 리소스에 대한 데이터와 유용한 인사이트를 제공하는 모니터링 및 관리 서비스입니다. CloudWatch를 사용하면 단일 플랫폼에서 모든 성능 및 운영 데이터를 로그 및 지표 형태로 수집하고 액세스할 수 있습니다. Application Manager의 리소스에 대한 CloudWatch 로그 및 경보를 보려면 CloudWatch를 설정하고 구성해야 합니다. 요금에 대한 자세한 내용은 CloudWatch 요금을 참조하세요.

> **Note**
>
> CloudWatch Logs 지원은 클러스터가 아닌 애플리케이션에만 적용됩니다.

(옵션) AWS Config 설정 및 구성

AWS Config는 AWS 계정와 연관된 리소스의 상세 뷰를 제공합니다. 여기에는 리소스 구성 방식, 서로 관련된 방식 및 시간 경과에 따른 구성 및 관계 변경 방식이 포함됩니다. AWS Config를 사용하여 AWS 리소스의 구성 설정을 평가할 수 있습니다. 이상적인 구성 설정을 나타내는 AWS Config 규칙을 생성하여 이를 수행합니다. AWS Config는 리소스에 발생하는 구성 변경을 계속 추적하면서 이러한 변경 사항이 규칙의 조건을 위반하는지 여부를 확인합니다. 리소스가 규칙을 위반하는 경우 AWS Config는 리소스와 규칙을 비준수로 표시합니다. Application Manager는 AWS Config 규칙에 대한 규정 준수 정보를 표시합니다. Application Manager에서 이 데이터를 보려면 AWS Config를 설정하고 구성해야 합니다. 요금 정보는 AWS Config 요금을 참조하세요.

(선택) State Manager 연결 생성

Systems Manager State Manager를 사용하여 관리형 노드에 할당하는 구성을 생성할 수 있습니다. 어소시에이션이라는 구성은 노드에서 유지하려는 상태를 정의합니다. Application Manager에서 연결 규정 준수 데이터를 보려면 하나 이상의 State Manager 연결을 구성해야 합니다. State Manager는 추가 요금 없이 제공됩니다.

(옵션) OpsCenter 설정 및 구성

OpsCenter를 사용하여 Application Manager의 리소스에 대한 운영 작업 항목(OpsItems)을 볼 수 있습니다. 경보 및 이벤트에 따라 OpsCenter로 OpsItems를 자동으로 전송하도록 Amazon CloudWatch 및 Amazon EventBridge를 구성할 수 있습니다. OpsItems를 수동으로 입력할 수도 있습니다. 요금 정보는 AWS Systems Manager 요금을 참조하세요.

Systems Manager Application Manager에 대한 권한 구성

AWS Identity and Access Management(IAM) 엔터티(사용자, 그룹 또는 역할 등)가 이 주제에 나열된 API 작업에 액세스할 수 있는 경우 AWS Systems Manager의 기능인 Application Manager의 모든 기능을 사용할 수 있습니다. API 작업은 수행하는 다양한 기능을 이해하는 데 도움이 되도록 2개의 테이블로 구분됩니다.

다음 표에는 리소스 세부 정보를 보기 위해 Application Manager에서 리소스를 선택하는 경우 Systems Manager가 호출하는 API 작업이 나열되어 있습니다. 예를 들어 Application Manager가 Amazon EC2

Auto Scaling 그룹을 나열하고 세부 정보를 보기 위해 해당 그룹을 선택하면 Systems Manager가 `autoscaling:DescribeAutoScalingGroups` API 작업을 호출합니다. 계정에 Auto Scaling 그룹이 없으면 이 API 작업은 Application Manager에서 호출되지 않습니다.

리소스 세부 정보만

```
acm:DescribeCertificate
acm:ListTagsForCertificate
autoscaling:DescribeAutoScalingGroups
cloudfront:GetDistribution
cloudfront:ListTagsForResource
cloudtrail:DescribeTrails
cloudtrail:ListTags
cloudtrail:LookupEvents
codebuild:BatchGetProjects
codepipeline:GetPipeline
codepipeline:ListTagsForResource
dynamodb:DescribeTable
dynamodb:ListTagsOfResource
ec2:DescribeAddresses
ec2:DescribeCustomerGateways
ec2:DescribeHosts
ec2:DescribeInternetGateways
ec2:DescribeNetworkAcls
ec2:DescribeNetworkInterfaces
ec2:DescribeRouteTables
ec2:DescribeSecurityGroups
ec2:DescribeSubnets
ec2:DescribeVolumes
ec2:DescribeVpcs
ec2:DescribeVpnConnections
ec2:DescribeVpnGateways
elasticbeanstalk:DescribeApplications
elasticbeanstalk:ListTagsForResource
elasticloadbalancing:DescribeInstanceHealth
elasticloadbalancing:DescribeListeners
elasticloadbalancing:DescribeLoadBalancers
elasticloadbalancing:DescribeTags
iam:GetGroup
iam:GetPolicy
iam:GetRole
iam:GetUser
```

리소스 세부 정보만

```
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
```

다음 표에는 Systems Manager가 Application Manager에 나열된 애플리케이션 및 리소스를 변경하거나 선택한 애플리케이션 또는 리소스에 대한 작업 정보를 보는 데 사용하는 API 작업이 나열되어 있습니다.

애플리케이션 작업 및 세부 정보

```
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
```

애플리케이션 작업 및 세부 정보

```
cloudwatch:DescribeAlarms
cloudwatch:DescribeInsightRules
cloudwatch:DisableAlarmActions
cloudwatch:EnableAlarmActions
cloudwatch:GetMetricData
cloudwatch:ListTagsForResource
cloudwatch:PutMetricAlarm
config:DescribeComplianceByConfigRule
config:DescribeComplianceByResource
config:DescribeConfigRules
config:DescribeRemediationConfigurations
config:GetComplianceDetailsByConfigRule
config:GetComplianceDetailsByResource
config:GetResourceConfigHistory
config:ListDiscoveredResources
config:PutRemediationConfigurations
config:SelectResourceConfig
config:StartConfigRulesEvaluation
config:StartRemediationExecution
ec2:DescribeInstances
ecs:DescribeCapacityProviders
ecs:DescribeClusters
ecs:DescribeContainerInstances
ecs:ListClusters
ecs:ListContainerInstances
ecs:TagResource
eks:DescribeCluster
eks:DescribeFargateProfile
eks:DescribeNodegroup
eks:ListClusters
eks:ListFargateProfiles
eks:ListNodegroups
eks:TagResource
iam:CreateServiceLinkedRole
iam:ListRoles
logs:DescribeLogGroups
resource-groups:CreateGroup
resource-groups:DeleteGroup
resource-groups:GetGroup
resource-groups:GetGroupQuery
resource-groups:GetTags
resource-groups:ListGroupResources
```

애플리케이션 작업 및 세부 정보

```
resource-groups:ListGroups
resource-groups:Tag
resource-groups:Untag
resource-groups:UpdateGroup
s3:ListAllMyBuckets
s3:ListBucket
s3:ListBucketVersions
servicecatalog:GetApplication
servicecatalog:ListApplications
sns:CreateTopic
sns:ListSubscriptionsByTopic
sns:ListTopics
sns:Subscribe
ssm:AddTagsToResource
ssm:CreateDocument
ssm:CreateOpsMetadata
ssm:DeleteDocument
ssm:DeleteOpsMetadata
ssm:DescribeAssociation
ssm:DescribeAutomationExecutions
ssm:DescribeDocument
ssm:DescribeDocumentPermission
ssm:GetDocument
ssm:GetInventory
ssm:GetOpsMetadata
ssm:GetOpsSummary
ssm:GetServiceSetting
ssm:ListAssociations
ssm:ListComplianceItems
ssm:ListDocuments
ssm:ListDocumentVersions
ssm:ListOpsMetadata
ssm:ListResourceComplianceSummaries
ssm:ListTagsForResource
ssm:ModifyDocumentPermission
ssm:RemoveTagsFromResource
ssm:StartAssociationsOnce
ssm:StartAutomationExecution
ssm:UpdateDocument
ssm:UpdateDocumentDefaultVersion
ssm:UpdateOpsItem
ssm:UpdateOpsMetadata
```

애플리케이션 작업 및 세부 정보

```
ssm:UpdateServiceSetting
tag:GetTagKeys
tag:GetTagValues
tag:TagResources
tag:UntagResources
```

권한 구성

IAM 엔터티(사용자, 그룹 또는 역할 등)에 대한 Application Manager 권한을 구성하려면 다음 예제를 사용하여 IAM 정책을 생성합니다. 이 정책 예에는 Application Manager에서 사용하는 모든 API 작업이 포함됩니다.

```
                {
    "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

사용자, 그룹 또는 역할에 연결된 IAM 권한 정책에서 다음 API 작업을 제거하여 Application Manager에서 애플리케이션과 리소스를 변경할 수 있는 사용자의 기능을 제한할 수 있습니다. 이러한 작업을 제거하면 Application Manager에서 읽기 전용 환경이 조성됩니다. 다음은 사용자가 애플리케이션 또는 기타 관련 리소스를 변경할 수 있도록 지원하는 모든 API입니다.

```
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
```

IAM 정책 생성 및 편집에 대한 자세한 내용은 IAM User Guide의 Creating IAM Policies를 참조하세요. IAM 엔터티(사용자, 그룹 또는 역할 등)에 이 정책을 할당하는 방법에 대한 자세한 내용은 IAM 자격 증명 권한 추가 및 제거를 참조하세요.

Application Manager에 애플리케이션 및 클러스터 추가

Application Manager는 AWS Systems Manager의 구성 요소입니다. Application Manager에서 애플리케이션은 하나의 단위로 작동하려는 AWS 리소스의 논리적 그룹입니다. 이 논리적 그룹은 다양한 버전의 애플리케이션, 운영자의 소유권 경계 또는 개발자 환경 등을 나타낼 수 있습니다.

Application Manager 홈 페이지에서 Get started(시작하기)를 선택하면 Application Manager은(는) 다른 AWS 서비스 또는 Systems Manager 기능에서 생성된 리소스에 대한 메타데이터를 자동으로 가져옵니다. 애플리케이션의 경우 Application Manager는 리소스 그룹으로 구성된 모든 AWS 리소스에 대한 메타데이터를 가져옵니다. 각 리소스 그룹은 사용자 정의 애플리케이션(Custom applications) 범주에 고유한 애플리케이션으로 나열됩니다. Application Manager는 AWS CloudFormation, AWS Launch Wizard, Amazon Elastic Container Service(Amazon ECS) 및 Amazon Elastic Kubernetes Service(Amazon EKS)에서 생성한 리소스에 대한 메타데이터도 자동으로 가져옵니다. 그런 다음 Application Manager가 미리 정의된 범주에 해당 리소스를 표시합니다.

애플리케이션(Applications)의 경우 목록에 다음이 포함됩니다.

- 사용자 정의 애플리케이션
- Launch Wizard
- CloudFormation 스택

- AppRegistry 애플리케이션

컨테이너 클러스터(Container clusters)의 경우 목록에 다음이 포함됩니다.

- Amazon ECS 클러스터
- Amazon EKS 클러스터

가져오기가 완료되면 이러한 미리 정의된 범주의 애플리케이션 또는 특정 리소스에 대한 작업 정보를 볼 수 있습니다. 또는 리소스 컬렉션에 대한 추가 컨텍스트를 제공하려면 Application Manager에서 수동으로 애플리케이션을 생성할 수 있습니다. 그런 다음 해당 애플리케이션에 리소스 또는 리소스 그룹을 추가할 수 있습니다. Application Manager에서 애플리케이션을 생성한 후 애플리케이션 컨텍스트에서 리소스에 대한 작업 정보를 볼 수 있습니다.

Application Manager에서 애플리케이션 생성

다음 절차에 따라 Application Manager에 애플리케이션을 생성하고 해당 애플리케이션에 리소스를 추가합니다.

Application Manager에서 애플리케이션을 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Application Manager를 선택합니다.
3. 애플리케이션(Applications) 탭을 선택한 다음 애플리케이션 생성(Create application)을 선택합니다.
4. 애플리케이션 이름(Application name)에 이 애플리케이션에 추가될 리소스의 용도를 이해하는 데 도움이 되는 이름을 입력합니다.
5. 애플리케이션 설명(Application description)에 이 애플리케이션에 대한 설명을 입력합니다.
6. 애플리케이션 구성 요소 선택(Choose application components) 섹션에서 제공된 옵션을 사용하여 이 애플리케이션에 대한 리소스를 선택합니다. 태그가 지정된 리소스, 리소스 그룹 및 스택의 조합을 애플리케이션에 추가할 수 있습니다. 최소 2개의 구성 요소와 최대 15개의 구성 요소를 선택해야 합니다. 태그를 사용하여 리소스를 선택하면 새 애플리케이션을 추가한 후 해당 태그에 할당된 모든 리소스가 리소스(Resources) 탭에 나열됩니다. 리소스 그룹 또는 스택에 포함된 리소스의 경우에도 마찬가지입니다.

   애플리케이션에 추가하려는 리소스가 표시되지 않으면 리소스에 제대로 태그가 지정되었는지, AWS Resource Groups 그룹에 추가되었거나, AWS CloudFormation 스택에 추가되었는지 확인합니다.

7. 애플리케이션 태그 - 옵션(Application tags - optional)에서 이 애플리케이션의 태그를 지정합니다.
8. 생성(Create)을 선택합니다.

Application Manager가 애플리케이션을 생성하고 엽니다. 구성 요소(Components) 트리에 새 애플리케이션이 최상위 구성 요소로 나열되고 선택한 리소스, 그룹 또는 스택이 하위 구성 요소로 나열됩니다. 다음에 Application Manager를 열면 사용자 정의 애플리케이션(Custom applications) 범주에서 새 애플리케이션을 찾을 수 있습니다.

Application Manager 작업

Application Manager는 AWS Systems Manager의 구성 요소입니다. 이 섹션에는 Application Manager 애플리케이션 및 클러스터로 작업하고 AWS 리소스에 대한 작업 정보를 보는 데 도움이 되는 주제가 포함되어 있습니다.

내용

- 애플리케이션 작업
- Application Manager에서 AWS CloudFormation 템플릿 및 스택 작업
- Application Manager에서 클러스터 작업

애플리케이션 작업

Application Manager는 AWS Systems Manager의 구성 요소입니다. 이 섹션에는 Application Manager 애플리케이션으로 작업하고 AWS 리소스에 대한 작업 정보를 보는 데 도움이 되는 주제가 포함되어 있습니다.

내용

- 애플리케이션에 대한 개요 정보 보기
- 애플리케이션 인스턴스 작업
- 애플리케이션 리소스 보기
- 규정 준수 정보 보기
- 모니터링 정보 보기
- 애플리케이션에 대한 OpsItems 보기
- 로그 그룹 및 로그 데이터 보기
- Application Manager에서 실행서로 작업

- Application Manager에서 태그 작업하기

애플리케이션에 대한 개요 정보 보기

AWS Systems Manager의 구성 요소인 Application Manager의 개요(Overview) 탭에는 Amazon CloudWatch 경보, 운영 작업 항목(OpsItems), CloudWatch Application Insights, 실행서 기록에 대한 요약이 표시됩니다. 카드에서 모두 보기(View all)를 선택하여 모든 Application Insights, 경보, OpsItems, 또는 실행서 기록을 볼 수 있는 해당 탭을 엽니다.

Application Insights 정보

CloudWatch Application Insights는 애플리케이션 리소스와 기술 스택 전반에 걸쳐 주요 지표, 로그, 경보를 식별하고 설정합니다. Application Insights는 지표 및 로그를 지속적으로 모니터링하여 이상 및 오류를 감지하고 연결합니다. 시스템이 오류 및 이상을 감지하면 Application Insights에서 알림을 설정하고 작업을 수행할 수 있는 CloudWatch Events를 생성합니다. 모니터링(Monitoring) 탭에서 구성 편집(Edit configuration) 버튼을 선택하면 CloudWatch Application Insights 콘솔이 열립니다. Application Insights에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 Amazon CloudWatch Application Insights란 무엇인가요?를 참조하세요.

Cost Explorer 정보

Application Manager는 비용 위젯 및 비용 탭을 통해 AWS Cost Management의 특성인 AWS Cost Explorer와 통합됩니다. Cost Management 콘솔에서 Cost Explorer를 활성화하면, Application Manager의 비용 위젯과 비용 탭에 특정 비컨테이너 애플리케이션 또는 애플리케이션 구성 요소에 대한 비용 데이터가 표시됩니다. 위젯 또는 탭의 필터를 사용하여 다양한 기간, 세부 수준 및 비용 유형에 따라 비용 데이터를 막대 또는 선 차트로 볼 수 있습니다.

Go to AWS Cost Management console 버튼을 선택하여 이 기능을 활성화할 수 있습니다. 기본적으로 지난 3개월 치 데이터가 필터링됩니다. 비컨테이너 애플리케이션의 경우, View all(전체 보기) 버튼을 선택하면 Application Manager가 Resources(리소스) 탭을 엽니다. 컨테이너 애플리케이션의 경우전체 보기 버튼이 AWS Cost Explorer 콘솔을 엽니다.

이 페이지에서 수행할 수 있는 작업

이 페이지의 Overview(개요) 탭에서 다음 위젯을 활성화하고 관련 정보에 액세스할 수 있습니다. 위젯이 활성화되면 View all(모두 보기)을 선택하여 해당 영역의 관련 애플리케이션 세부 정보를 봅니다.

- Insights and Alarms(인사이트 및 경보) 섹션에서 심각도를 나타내는 숫자를 선택하여 선택한 심각도의 경보에 대한 세부 정보를 볼 수 있는 Monitoring(모니터링) 탭을 엽니다.

- Cost(비용) 섹션에서 View all(모두 보기)을 선택하여 특정 애플리케이션 또는 애플리케이션 구성 요소에 대한 비용 데이터를 볼 수 있는 Resources(리소스) 탭을 엽니다.
- Compliance(규정 준수) 섹션에서 View all(모두 보기)을 선택하여 AWS Config 및 State Manager 연결의 규정 준수 정보를 볼 수 있는 Compliance(규정 준수) 탭을 엽니다.

> **Note**
>
> 패치 규정 준수 세부 정보를 보려면 Compliance(규정 준수) 탭을 직접 선택합니다. 그러면 선택한 애플리케이션에서 사용하는 관리형 노드에 대한 패치 규정 준수 세부 정보를 볼 수 있습니다.

- 실행서(Runbooks) 섹션에서 실행서를 선택하여 문서에 대한 자세한 내용을 볼 수 있는 Systems Manager Documents 페이지에서 엽니다.
- OpsItems 섹션에서 심각도를 선택하여 선택한 심각도의 OpsItems를 모두 볼 수 있는 OpsItems 탭을 엽니다.
- 모두 보기(View all) 버튼을 선택하여 해당 탭을 엽니다. 애플리케이션에 대한 모든 경보, OpsItem 또는 실행서 기록 항목을 볼 수 있습니다.

개요(Overview) 탭 열기

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Application Manager를 선택합니다.
3. 애플리케이션(Applications) 섹션에서 범주를 선택합니다. Application Manager에서 수동으로 생성한 애플리케이션을 열려면 사용자 정의 애플리케이션(Custom applications)을 선택합니다.
4. 목록에서 애플리케이션을 선택합니다. Application Manager가 개요(Overview) 탭을 엽니다.

애플리케이션 인스턴스 작업

Application Manager는 Amazon Elastic Compute Cloud(Amazon EC2)와 통합되어 애플리케이션의 컨텍스트에서 인스턴스에 대한 정보를 표시합니다. Application Manager는 선택한 애플리케이션의 인스턴스 상태, 상태 및 Amazon EC2 Auto Scaling 상태를 그래픽 형식으로 표시합니다. Instances(인스턴스) 탭에는 애플리케이션의 각 인스턴스에 대한 다음 정보가 포함된 테이블도 있습니다.

- 인스턴스 상태(보류 중, 중지 중, 실행 중, 중지됨)
- SSM Agent의 Ping 상태
- 인스턴스에서 대해 가장 최근에 처리된 Systems Manager Automation 런북의 상태 및 이름

- 상태별 Amazon CloudWatch Logs 경보의 수입니다.
  - `ALARM`-지표 또는 표현식이 정의된 임곗값을 벗어났습니다.
  - `OK`-지표 또는 표현식이 정의된 임곗값 내에 있습니다.
  - `INSUFFICIENT_DATA`-경보가 방금 시작되었거나 지표를 사용할 수 없거나 지표에서 경보 상태를 결정하는 데 사용할 수 있는 데이터가 충분하지 않습니다.
- 상위 및 개별 오토 스케일링 그룹의 오토 스케일링 상태

All instances(모든 인스턴스) 테이블에서 인스턴스를 선택하면 Application Manager가 해당 인스턴스에 대한 정보를 다음 네 개의 탭에 표시합니다.

- Details(세부 정보) - Amazon Machine Image(AMI), DNS 정보, IP 주소 정보 등, Amazon EC2의 모든 인스턴스 세부 정보입니다.
- Health(상태) - EC2 시스템 및 인스턴스 상태 검사에서 제공하는 현재 상태입니다.
- Execution history(실행 기록) - 인스턴스에서 처리한 Systems Manager Automation 런북 및 API 호출에 대한 실행 로그입니다.
- CloudWatch alarms(CloudWatch 경보) - 인스턴스에서 발생한 모든 CloudWatch 경보의 이름, 상태 및 기타 정보입니다.

이 페이지에서 수행할 수 있는 작업

이 페이지에서 다음 작업을 수행할 수 있습니다.

- 인스턴스 시작, 중지 및 종료
- Chef 레시피를 적용합니다.
- 오토 스케일링에 인스턴스를 연결하거나 오토 스케일링에서 인스턴스를 분리합니다.
- SSM Agent에 대한 자동 업데이트를 활성화합니다.

Resources(리소스) 탭을 열려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Application Manager를 선택합니다.
3. 애플리케이션(Applications) 섹션에서 범주를 선택합니다. Application Manager에서 수동으로 생성한 애플리케이션을 열려면 Custom applications(사용자 지정 애플리케이션)를 선택합니다.
4. 목록에서 애플리케이션을 선택합니다. Application Manager가 개요(Overview) 탭을 엽니다.

5. [Instances] 탭을 선택합니다.

애플리케이션 인스턴스에 대한 세부 정보를 보는 방법

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Application Manager를 선택합니다.
3. 애플리케이션(Applications) 섹션에서 범주를 선택합니다. Application Manager에서 수동으로 생성한 애플리케이션을 열려면 Custom applications(사용자 지정 애플리케이션)를 선택합니다.
4. 목록에서 애플리케이션을 선택합니다. Application Manager가 개요(Overview) 탭을 엽니다.
5. [Instances] 탭을 선택합니다.
6. 세부 정보를 보려는 인스턴스 옆에 있는 버튼을 선택합니다.
7. 페이지 하단에 있는 인스턴스 세부 정보를 검토합니다.

SSM Agent을 자동 업데이트하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Application Manager를 선택합니다.
3. 애플리케이션(Applications) 섹션에서 범주를 선택합니다. Application Manager에서 수동으로 생성한 애플리케이션을 열려면 Custom applications(사용자 지정 애플리케이션)를 선택합니다.
4. 목록에서 애플리케이션을 선택합니다. Application Manager가 개요(Overview) 탭을 엽니다.
5. [Instances] 탭을 선택합니다.
6. 에이전트 작업 드롭다운에서 SSM Agent 업데이트 구성을 선택합니다.
7. 모든 관리형 인스턴스에 대한 자동 SSM Agent 업데이트를 구성하려면 모든 인스턴스를 선택합니다. 또는 인스턴스를 선택하여 애플리케이션의 단일 인스턴스에 대한 자동 SSM Agent 업데이트를 구성합니다.
8. 자동 업데이트 활성화 토글을 선택합니다.
9. 일정 지정 드롭다운에서 SSM Agent 업데이트에 사용할 일정을 선택합니다.
10. 구성을 선택합니다.

애플리케이션 리소스 보기

AWS Systems Manager의 구성 요소인 Application Manager의 리소스(Resources) 탭에는 애플리케이션의 AWS 리소스가 표시됩니다. 최상위 구성 요소를 선택하면 이 페이지에 해당 구성 요소와 모든 하

위 구성 요소에 대한 모든 리소스가 표시됩니다. 하위 구성 요소를 선택하면 이 페이지에 해당 하위 구성 요소에 할당된 리소스만 표시됩니다.

이 페이지에서 수행할 수 있는 작업

이 페이지에서 다음 작업을 수행할 수 있습니다.

- 리소스 이름을 선택하여 리소스가 생성된 콘솔에서 제공하는 세부 정보, 태그, Amazon CloudWatch 경보, AWS Config 세부 정보 및 AWS CloudTrail 로그 정보를 포함한 리소스에 대한 정보를 봅니다.
- 리소스 이름 옆에 있는 옵션 버튼을 선택합니다. 그런 다음 리소스 타임라인(Resource timeline) 버튼을 선택하여 선택한 리소스에 대한 규정 준수 정보를 볼 수 있는 AWS Config 콘솔을 엽니다.
- AWS Cost Explorer를 활성화한 경우, Cost Explorer 섹션에는 특정 애플리케이션 또는 애플리케이션 구성 요소에 대한 비용 데이터가 표시됩니다. Go to AWS Cost Management console 버튼을 선택하여 이 기능을 활성화할 수 있습니다. 이 섹션의 필터를 사용하여 애플리케이션에 대한 비용 정보를 볼 수 있습니다.

리소스(Resources) 탭 열기

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Application Manager를 선택합니다.
3. 애플리케이션(Applications) 섹션에서 범주를 선택합니다. Application Manager에서 수동으로 생성한 애플리케이션을 열려면 사용자 정의 애플리케이션(Custom applications)을 선택합니다.
4. 목록에서 애플리케이션을 선택합니다. Application Manager가 개요(Overview) 탭을 엽니다.
5. 리소스 탭을 선택합니다.

규정 준수 정보 보기

Application Manager의 구성 요소인 AWS Systems Manager에서 구성(Configurations) 페이지에는 AWS Config 리소스와 구성 규칙 규정 준수 정보가 표시됩니다. 이 페이지에는 AWS Systems Manager State Manager 연결 규정 준수 정보도 표시됩니다. 리소스, 규칙 또는 연결을 선택하여 해당 콘솔을 열어 자세한 내용을 볼 수 있습니다. 이 페이지에는 지난 90일간의 규정 준수 정보가 표시됩니다.

이 페이지에서 수행할 수 있는 작업

이 페이지에서 다음 작업을 수행할 수 있습니다.

- 리소스 이름을 선택하여 선택한 리소스에 대한 규정 준수 정보를 볼 수 있는 AWS Config 콘솔을 엽니다.
- 리소스 이름 옆에 있는 옵션 버튼을 선택합니다. 그런 다음 리소스 타임라인(Resource timeline) 버튼을 선택하여 선택한 리소스에 대한 규정 준수 정보를 볼 수 있는 AWS Config 콘솔을 엽니다.
- 구성 규칙 규정 준수(Config rules compliance) 섹션에서 다음을 수행할 수 있습니다.
  - 규칙 이름을 선택하여 해당 규칙에 대한 정보를 볼 수 있는 AWS Config 콘솔을 엽니다.
  - 규칙 추가(Add rules)를 선택하여 규칙을 생성할 수 있는 AWS Config 콘솔을 엽니다.
  - 규칙 이름 옆에 있는 옵션 버튼을 선택하고 작업(Actions)을 선택한 다음 문제 해결 관리(Manage remediation)를 선택하여 규칙에 대한 문제 해결 작업을 변경합니다.
  - 규칙 이름 옆에 있는 옵션 버튼을 선택하고 작업(Actions)을 선택한 다음 재평가(Re-evaluate)를 선택하여 AWS Config가 선택한 규칙에 대한 규정 준수 점검을 실행하도록 합니다.
- 연결 규정 준수(Association compliance) 섹션에서 다음을 수행할 수 있습니다.
  - 연결 이름을 선택하여 해당 연결에 대한 정보를 볼 수 있는 연결(Associations) 페이지를 엽니다.
  - 연결 생성(Create association)을 선택하여 연결을 생성할 수 있는 Systems Manager State Manager를 엽니다.
  - 연결 이름 옆의 옵션 버튼을 선택하고 연결 적용(Apply association)을 선택하여 연결에 지정된 모든 작업을 즉시 시작합니다.

규정 준수(Compliance) 탭 열기

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Application Manager를 선택합니다.
3. 애플리케이션(Applications) 섹션에서 범주를 선택합니다. Application Manager에서 수동으로 생성한 애플리케이션을 열려면 사용자 정의 애플리케이션(Custom applications)을 선택합니다.
4. 목록에서 애플리케이션을 선택합니다. Application Manager가 개요(Overview) 탭을 엽니다.
5. 규정 준수(Compliance) 탭을 엽니다.

모니터링 정보 보기

AWS Systems Manager의 구성 요소인 Application Manager의 모니터링(Monitoring) 탭에는 애플리케이션의 리소스에 대한 Amazon CloudWatch Application Insights 및 경보 세부 정보가 표시됩니다.

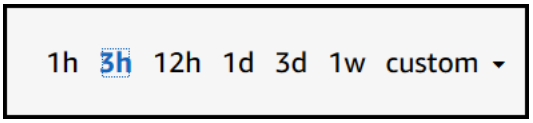
Application Insights 정보

CloudWatch Application Insights는 애플리케이션 리소스와 기술 스택 전반에 걸쳐 주요 지표, 로그, 경보를 식별하고 설정합니다. Application Insights는 지표 및 로그를 지속적으로 모니터링하여 이상 및 오류를 감지하고 연결합니다. 시스템이 오류 및 이상을 감지하면 Application Insights에서 알림을 설정하고 작업을 수행할 수 있는 CloudWatch Events를 생성합니다. 모니터링(Monitoring) 탭에서 구성 편집(Edit configuration) 버튼을 선택하면 CloudWatch Application Insights 콘솔이 열립니다. Application Insights에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 Amazon CloudWatch Application Insights란 무엇인가요?를 참조하세요.

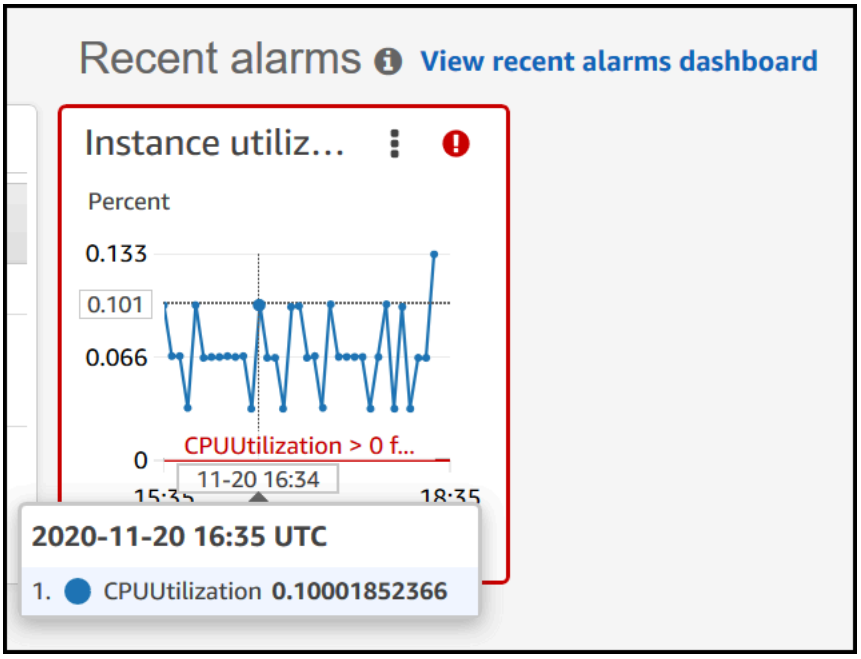
이 페이지에서 수행할 수 있는 작업

이 페이지에서 다음 작업을 수행할 수 있습니다.

- AWS 서비스별 경보 섹션에서 서비스 이름을 선택하여 선택한 서비스 및 경보에 대한 CloudWatch 를 엽니다.
- 미리 정의된 기간 값 중 하나를 선택하여 최근 경보(Recent alarms) 섹션의 위젯에 표시되는 데이터의 기간을 조정합니다. 사용자 정의(custom)를 선택하여 기간을 직접 정의할 수 있습니다.

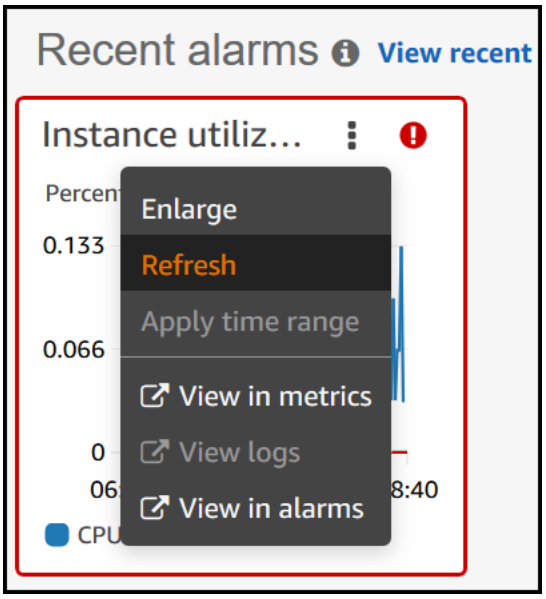


- 최근 경보(Recent alarms) 섹션에서 위젯 위로 커서를 가져가서 특정 시간에 대한 데이터 팝업을 봅니다.



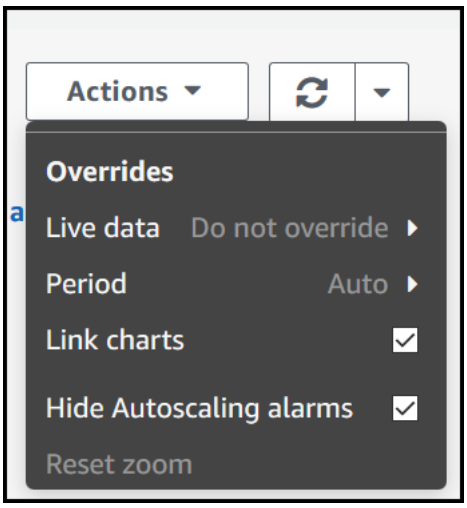
- 위젯에서 옵션 메뉴를 선택하여 표시 옵션을 봅니다. 확대(Enlarge)를 선택하여 위젯을 확장합니다. 새로 고침(Refresh)을 선택하여 위젯의 데이터를 업데이트합니다. 위젯 데이터 표시에서 커서를 클

릭하고 끌어 특정 범위를 선택합니다. 그런 다음 시간 범위 적용(Apply time range)을 선택할 수 있습니다.



- 작업(Actions) 메뉴를 선택하여 다음을 포함한 경보 데이터 재정의(Override) 옵션을 봅니다.
  - 위젯에서 라이브 데이터를 표시할지 여부를 선택합니다. 라이브 데이터는 마지막 1분 이내에 게시된 완전히 집계되지 않은 데이터입니다. 라이브 데이터가 해제된 경우 집계 기간이 지난 1분 이상인 데이터 포인트만 표시됩니다. 예를 들어 5분의 기간을 사용하는 경우 12:35의 데이터 포인트는 12:35~12:40에 집계되고 12:41에 표시됩니다.

    라이브 데이터가 설정되어 있으면 해당 집계 간격으로 데이터가 게시되는 즉시 최신 데이터 포인트가 표시됩니다. 화면을 새로 고칠 때마다 해당 집계 기간 내의 새 데이터가 게시됨에 따라 최신 데이터 포인트가 변경될 수 있습니다.
  - 라이브 데이터의 기간을 지정합니다.
  - 한 차트를 확대하거나 축소할 때 다른 차트가 동시에 확대 또는 축소되도록 최근 경보(Recent alarms) 섹션에서 차트를 연결합니다. 차트 연결을 해제하여 확대/축소를 한 차트로 제한할 수 있습니다.
  - Auto Scaling 경보를 숨깁니다.



모니터링(Monitoring) 탭 열기

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Application Manager를 선택합니다.
3. 애플리케이션(Applications) 섹션에서 범주를 선택합니다. Application Manager에서 수동으로 생성한 애플리케이션을 열려면 사용자 정의 애플리케이션(Custom applications)을 선택합니다.
4. 목록에서 애플리케이션을 선택합니다. Application Manager가 개요(Overview) 탭을 엽니다.
5. 모니터링(Monitoring) 탭을 선택합니다.

애플리케이션에 대한 OpsItems 보기

AWS Systems Manager의 구성 요소인 Application Manager의 OpsItems 탭에는 선택한 애플리케이션의 리소스에 대한 운영 작업 항목(OpsItems)이 표시됩니다. Amazon CloudWatch 경보와 Amazon EventBridge 이벤트에서 OpsItems를 자동으로 생성하도록 Systems Manager OpsCenter를 구성할 수 있습니다. OpsItems를 수동으로 생성할 수도 있습니다.

이 탭에서 수행할 수 있는 작업

이 페이지에서 다음 작업을 수행할 수 있습니다.

- 검색 필드를 사용하여 OpsItems 목록을 필터링합니다. OpsItem 이름, ID, 소스 ID 또는 심각도별로 필터링할 수 있습니다. 상태에 따라 목록을 필터링할 수도 있습니다. OpsItems는 미결(Open), 진행 중(In progress), 미결 및 진행 중(Open and In progress), 해결됨(Resolved) 또는 모두(All) 상태를 지원합니다.

- 옆에 있는 옵션 버튼을 선택한 다음 상태 설정(Set status) 메뉴에서 옵션을 선택하여 OpsItem의 상태를 변경합니다.
- Systems Manager OpsCenter를 열고 OpsItem 생성(Create OpsItem)을 선택하여 OpsItem을 생성합니다.

OpsItems 탭을 열려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Application Manager를 선택합니다.
3. 애플리케이션(Applications) 섹션에서 범주를 선택합니다. Application Manager에서 수동으로 생성한 애플리케이션을 열려면 사용자 정의 애플리케이션(Custom applications)을 선택합니다.
4. 목록에서 애플리케이션을 선택합니다. Application Manager가 [개요(Overview)] 탭을 엽니다.
5. OpsItems 탭을 선택합니다.

로그 그룹 및 로그 데이터 보기

AWS Systems Manager의 구성 요소인 Application Manager의 [로그(Logs)] 탭에는 Amazon CloudWatch Logs의 로그 그룹 목록이 표시됩니다.

이 탭에서 수행할 수 있는 작업

이 페이지에서 다음 작업을 수행할 수 있습니다.

- 로그 그룹 이름을 선택하여 CloudWatch Logs에서 엽니다. 그런 다음 로그 스트림을 선택하여 애플리케이션 컨텍스트에서 리소스에 대한 로그를 볼 수 있습니다.
- [로그 그룹 생성(Create log groups)]을 선택하여 CloudWatch Logs에서 로그 그룹을 생성합니다.

로그(Logs) 탭 열기

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Application Manager를 선택합니다.
3. 애플리케이션(Applications) 섹션에서 범주를 선택합니다. Application Manager에서 수동으로 생성한 애플리케이션을 열려면 사용자 정의 애플리케이션(Custom applications)을 선택합니다.
4. 목록에서 애플리케이션을 선택합니다. Application Manager가 [개요(Overview)] 탭을 엽니다.
5. 로그 탭을 선택합니다.

Application Manager에서 실행서로 작업

Automation 실행서를 사용하여 AWS Systems Manager의 기능인 Application Manager에서 AWS 리소스 문제를 해결할 수 있습니다. Automation 실행서는 자동화가 실행될 때 Systems Manager가 관리형 인스턴스 및 기타 AWS 리소스에 대해 수행하는 작업을 정의합니다. Automation은 AWS Systems Manager의 기능입니다. 실행서에는 순차적으로 실행되는 하나 이상의 단계가 포함되어 있습니다. 각 단계는 단일 작업을 중심으로 구축됩니다. 한 단계의 출력을 이후 단계에서 입력으로 사용할 수 있습니다.

Application Manager 애플리케이션 또는 클러스터에서 Start runbook(실행서 시작)을 선택하면 애플리케이션 또는 클러스터의 리소스 유형에 따라 필터링된 사용 가능한 실행서 목록이 표시됩니다. 시작할 실행서를 선택하면 Systems Manager에서 [자동화 문서 실행(Execute automation document)] 페이지를 엽니다.

Application Manager에는 실행서 작업에 대한 다음과 같은 향상된 기능이 포함되어 있습니다.

- Application Manager에서 리소스 이름을 선택한 다음 실행서 실행(Execute runbook)을 선택하면, 해당 리소스 유형에 대해 필터링된 실행서 목록이 표시됩니다.
- 목록에서 실행서를 선택한 다음 동일한 유형의 리소스에 대해 실행(Run for resources of same type)을 선택하여 동일한 유형의 모든 리소스에 대해 자동화를 시작할 수 있습니다.

시작하기 전 준비 사항

Application Manager에서 실행서를 시작하기 전에 다음을 수행합니다.

- 실행서를 시작하는 데 올바른 권한이 있는지 확인합니다. 자세한 내용은 Automation 설정 단원을 참조하십시오.
- 실행서 시작에 대한 Automation 절차 설명서를 검토합니다. 자세한 내용은 자동화 실행 단원을 참조하십시오.

Application Manager에서 실행서를 시작하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Application Manager를 선택합니다.
3. 애플리케이션(Applications) 섹션에서 범주를 선택합니다. Application Manager에서 수동으로 생성한 애플리케이션을 열려면 사용자 정의 애플리케이션(Custom applications)을 선택합니다.
4. 목록에서 애플리케이션을 선택합니다. Application Manager가 개요(Overview) 탭을 엽니다.

5. 런북 시작을 선택합니다. Application Manager에서 자동화 위젯 팝업이 열립니다. 자동화 위젯의 옵션에 대한 자세한 내용은 자동화 실행 섹션을 참조하세요.

Application Manager에서 태그 작업하기

Application Manager에서 애플리케이션 및 AWS 리소스에서 태그를 빠르게 추가하거나 삭제할 수 있습니다. 태그에 대한 자세한 내용은 Systems Manager 리소스에 태그 지정 섹션을 참조하세요.

애플리케이션 및 애플리케이션의 모든 AWS 리소스에 태그를 추가하거나 삭제하려면 다음 절차를 따릅니다.

애플리케이션 및 애플리케이션의 모든 리소스에 태그를 추가하거나 삭제하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Application Manager를 선택합니다.
3. 애플리케이션(Applications) 섹션에서 범주를 선택합니다. Application Manager에서 수동으로 생성한 애플리케이션을 열려면 사용자 정의 애플리케이션(Custom applications)을 선택합니다.
4. 목록에서 애플리케이션을 선택합니다. Application Manager가 개요(Overview) 탭을 엽니다.
5. 애플리케이션 정보(Application information) 섹션에서 애플리케이션 태그(Application tags) 아래의 숫자를 선택합니다. 애플리케이션에 할당된 태그가 없는 경우 숫자는 0입니다.
6. 태그를 추가하려면 태그 추가(Add new tag)를 선택합니다. 키와 값(선택 사항)을 지정합니다. 태그를 삭제하려면 제거를 선택합니다.
7. Save(저장)를 선택합니다.

Application Manager의 특정 리소스에 태그를 추가하거나 리소스에서 태그를 삭제하려면 다음 절차를 따릅니다.

리소스에 태그를 추가하거나 리소스에서 태그를 삭제하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Application Manager를 선택합니다.
3. 애플리케이션(Applications) 섹션에서 범주를 선택합니다. Application Manager에서 수동으로 생성한 애플리케이션을 열려면 사용자 정의 애플리케이션(Custom applications)을 선택합니다.
4. 목록에서 애플리케이션을 선택합니다. Application Manager가 개요(Overview) 탭을 엽니다.
5. 리소스(Resources) 탭을 선택합니다.
6. 리소스 이름을 선택합니다.

7. 태그(Tags) 섹션에서 편집(Edit)을 선택합니다.
8. 태그를 추가하려면 태그 추가(Add new tag)를 선택합니다. 키와 값(선택 사항)을 지정합니다. 태그를 삭제하려면 제거를 선택합니다.
9. Save(저장)를 선택합니다.

Application Manager에서 AWS CloudFormation 템플릿 및 스택 작업

AWS Systems Manager의 기능인 Application Manager는 AWS CloudFormation과 통합되어 애플리케이션에 대한 리소스를 프로비저닝하고 관리하는 데 도움이 됩니다. Application Manager에서 AWS CloudFormation 템플릿과 스택을 생성, 수정 및 삭제할 수 있습니다. 스택이란 하나의 단위로 관리할 수 있는 AWS 리소스의 모음입니다. 즉, CloudFormation 스택을 사용하여 AWS 리소스 모음을 생성, 업데이트 또는 삭제할 수 있습니다. 템플릿은 스택에서 프로비저닝할 리소스를 지정하는 JSON 또는 YAML 형식의 텍스트 파일입니다.

Application Manager에는 템플릿을 복제, 생성 및 저장할 수 있는 템플릿 라이브러리도 포함되어 있습니다. Application Manager와 CloudFormation은 스택의 현재 상태에 대해 동일한 정보를 표시합니다. 템플릿 및 템플릿 업데이트는 스택을 프로비저닝할 때까지 Systems Manager에 저장되며, 이때 변경 사항은 CloudFormation에도 표시됩니다.

Application Manager에서 스택을 생성하면 [CloudFormation 스택(CloudFormation stacks)] 페이지에 이에 대한 유용한 정보가 표시됩니다. 여기에는 스택을 생성하는 데 사용되는 템플릿, 스택의 리소스에 대한 OpsItems 수, 스택 상태 및 드리프트 상태가 포함됩니다.

Cost Explorer 정보

Application Manager는 Cost(비용) 위젯을 통해 AWS Cost Management의 기능인 AWS Cost Explorer와 통합됩니다. Cost Management 콘솔에서 Cost Explorer를 활성화하면, Application Manager의 Cost(비용) 위젯은 특정 비컨테이너 애플리케이션 또는 애플리케이션 구성 요소에 대한 비용 데이터를 보여줍니다. 위젯의 필터를 사용하여 막대 또는 꺾은선형 차트에서 다양한 기간, 세부 기간 및 비용 유형에 따라 비용 데이터를 볼 수 있습니다.

Go to AWS Cost Management console 버튼을 선택하여 이 기능을 활성화할 수 있습니다. 기본적으로 지난 3개월 치 데이터가 필터링됩니다. 비컨테이너 애플리케이션의 경우, View all(전체 보기) 버튼을 선택하면 Application Manager가 Resources(리소스) 탭을 엽니다. 컨테이너 애플리케이션의 경우전체 보기 버튼이 AWS Cost Explorer 콘솔을 엽니다.

> **Note**
>
> Cost Explorer는 태그를 사용하여 애플리케이션 비용을 추적합니다. AWS CloudFormation 스택 기반 애플리케이션이 `AppManagerCFNStackKey` 태그 키로 구성되지 않은 경우 Cost Explorer는 Application Manager에 정확한 비용 데이터를 표시하지 못합니다. `AppManagerCFNStackKey` 태그 키가 감지되지 않으면 CloudFormation 스택에 태그를 추가하여 비용 추적을 활성화하라는 메시지가 콘솔에 표시됩니다. 그러면 태그 키가 스택의 Amazon 리소스 이름(ARN)에 매핑되고 Cost(비용) 위젯에서 정확한 비용 데이터를 표시할 수 있습니다.

> **Important**
>
> `AppManagerCFNStackKey` 태그를 추가하면 스택 업데이트가 트리거됩니다. 스택이 처음 배포된 이후에 수행된 수동 구성은 사용자 태그가 추가되면 반영되지 않습니다. 리소스 업데이트 동작에 대한 자세한 내용은 AWS CloudFormation 사용 설명서의 스택 리소스의 업데이트 동작을 참조하세요.

시작하기 전 준비 사항

Application Manager를 사용하여 CloudFormation 템플릿 및 스택을 생성, 편집 또는 삭제하기 전에 다음 링크를 사용하여 CloudFormation 개념에 대해 알아봅니다.

- AWS CloudFormation이란 무엇인가요?
- AWS CloudFormation 모범 사례
- 템플릿 기본 사항 알아보기
- AWS CloudFormation 스택 작업
- AWS CloudFormation 템플릿 사용
- 샘플 템플릿

주제

- CloudFormation 템플릿 작업
- CloudFormation 스택 작업

CloudFormation 템플릿 작업

AWS Systems Manager의 기능인 Application Manager에는 AWS CloudFormation 템플릿을 관리하는 데 도움이 되는 템플릿 라이브러리 및 기타 도구가 포함되어 있습니다. 이 섹션에는 다음 정보가 포함됩니다.

주제

- 템플릿 라이브러리 작업
- 템플릿 생성
- 템플릿 편집

템플릿 라이브러리 작업

Application Manager 템플릿 라이브러리는 템플릿을 보고, 생성하고, 편집하고, 삭제하고, 복제하는 데 유용한 도구를 제공합니다. 템플릿 라이브러리에서 직접 스택을 프로비저닝할 수도 있습니다. 템플릿은 `CloudFormation` 유형의 Systems Manager(SSM) 문서로 저장됩니다. 템플릿을 SSM 문서로 저장하면 버전 컨트롤을 사용하여 다양한 버전의 템플릿으로 작업할 수 있습니다. 권한을 설정하고 템플릿을 공유할 수도 있습니다. 스택을 성공적으로 프로비저닝하면 Application Manager 및 CloudFormation에서 스택과 템플릿을 사용할 수 있습니다.

시작하기 전 준비 사항

Application Manager에서 CloudFormation 템플릿 작업을 시작하기 전에 다음 주제를 읽고 SSM 문서에 대해 자세히 알아보는 것이 좋습니다.

- AWS Systems Manager Documents
- SSM 문서 공유
- 공유 SSM 문서에 대한 모범 사례

Application Manager에서 템플릿 라이브러리를 보려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Application Manager를 선택합니다.
3. [애플리케이션(Applications)] 섹션에서 [CloudFormation 스택(CloudFormation stacks)]을 선택합니다.
4. [템플릿 라이브러리(Template library)]를 선택합니다.

템플릿 생성

다음 절차에서는 Application Manager에서 CloudFormation 템플릿을 생성하는 방법을 설명합니다. 템플릿을 생성할 때 템플릿의 스택 세부 정보를 JSON 또는 YAML로 입력합니다. JSON 또는 YAML에 익숙하지 않은 경우 템플릿을 시각적으로 생성하고 수정하기 위한 도구인 AWS CloudFormation Designer를 사용할 수 있습니다. 자세한 내용은 AWS CloudFormation 사용 설명서의 AWS CloudFormation Designer이란 무엇인가요?를 참조하세요. 템플릿의 구조 및 구문에 대한 자세한 내용은 템플릿 구조를 참조하세요.

여러 템플릿 조각에서 템플릿을 구성할 수도 있습니다. 템플릿 조각은 특정 리소스에 대한 템플릿을 작성하는 방법을 보여주는 예입니다. 예를 들어 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스, Amazon Simple Storage Service(Amazon S3) 도메인, AWS CloudFormation 매핑 등에 대한 조각을 볼 수 있습니다. 조각은 리소스별로 그룹화됩니다. AWS CloudFormation 사용 설명서의 일반 템플릿 조각 섹션에서 범용 AWS CloudFormation 조각을 찾을 수 있습니다.

Application Manager에서 CloudFormation 템플릿 생성(콘솔)

다음 절차에 따라 AWS Management Console을 사용하여 Application Manager에서 CloudFormation 템플릿을 생성합니다.

Application Manager에서 CloudFormation 템플릿을 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Application Manager를 선택합니다.
3. [애플리케이션(Applications)] 섹션에서 [CloudFormation 스택(CloudFormation stacks)]을 선택합니다.
4. [템플릿 라이브러리(Template library)]를 선택한 다음 [템플릿 생성(Create template)]을 선택하거나 기존 템플릿을 선택한 다음 [작업(Actions)], [복제(Clone)]를 선택합니다.
5. [이름(Name)]에 템플릿이 생성하는 리소스 또는 스택의 용도를 식별하는 데 도움이 되는 템플릿의 이름을 입력합니다.
6. (옵션) [버전 이름(Version name)]에 템플릿 버전을 식별할 이름 또는 숫자를 입력합니다.
7. (옵션) [설명(Description)]에 이 템플릿에 대한 정보를 입력합니다.
8. [코드 편집기(Code editor)] 섹션에서 [YAML] 또는 [JSON]을 선택한 다음 템플릿 코드를 입력하거나 복사하여 붙여 넣습니다.
9. (옵션) [태그(Tags)] 섹션에서 템플릿에 하나 이상의 태그 키 이름/값 페어를 적용합니다.

태그는 리소스에 할당하는 선택적 메타데이터입니다. 태그를 사용하여 용도, 소유자 또는 환경을 기준으로 하는 등 리소스를 다양한 방식으로 분류할 수 있습니다. Systems Manager 리소스 태그 지정에 대한 자세한 내용은 Systems Manager 리소스에 태그 지정 섹션을 참조하세요.

10. (옵션) [권한(Permissions)] 섹션에서 AWS 계정 ID를 입력하고 [계정 추가(Add account)]를 선택합니다. 이 작업은 템플릿에 대한 읽기 권한을 제공합니다. 계정 소유자는 템플릿을 프로비저닝하고 복제할 수 있지만 편집하거나 삭제할 수는 없습니다.
11. 생성(Create)을 선택합니다. 템플릿은 Systems Manager(SSM) 문서 서비스에 저장됩니다.

Application Manager에서 CloudFormation 템플릿 생성(명령줄)

JSON 또는 YAML로 CloudFormation 템플릿의 콘텐츠를 생성한 후 AWS Command Line Interface(AWS CLI) 또는 AWS Tools for PowerShell을 사용하여 템플릿을 SSM 문서로 저장할 수 있습니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

시작하기 전 준비 사항

아직 하지 않은 경우 AWS CLI 또는 AWS Tools for PowerShell를 설치하고 구성합니다. 자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트 및 AWS Tools for PowerShell 설치를 참조하세요.

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"
```

이 작업이 성공하면 명령에서 다음과 비슷한 응답이 반환됩니다.

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

템플릿 편집

다음 절차에 따라 Application Manager에서 CloudFormation 템플릿을 편집합니다. 업데이트된 템플릿을 사용하는 스택을 프로비저닝한 후 CloudFormation에서 템플릿 변경 사항을 사용할 수 있습니다.

Application Manager에서 CloudFormation 템플릿을 편집하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Application Manager를 선택합니다.
3. [애플리케이션(Applications)] 섹션에서 [CloudFormation 스택(CloudFormation stacks)]을 선택합니다.
4. [템플릿 라이브러리(Template library)]를 선택합니다.
5. 템플릿을 선택한 다음 [작업(Actions)]과 [편집(Edit)]을 차례대로 선택합니다. 템플릿의 이름은 변경할 수 없지만 다른 모든 세부 정보는 변경할 수 있습니다.
6. Save(저장)를 선택합니다. 템플릿은 Systems Manager 문서 서비스에 저장됩니다.

CloudFormation 스택 작업

AWS Systems Manager의 기능인 Application Manager는 AWS CloudFormation과 통합되어 애플리케이션에 대한 리소스를 프로비저닝하고 관리하는 데 도움이 됩니다. Application Manager에서 CloudFormation 템플릿과 스택을 생성, 수정 및 삭제할 수 있습니다. 스택이란 하나의 단위로 관리할 수 있는 AWS 리소스의 모음입니다. 즉, CloudFormation 스택을 사용하여 AWS 리소스 모음을 생성, 업데이트 또는 삭제할 수 있습니다. 템플릿은 스택에서 프로비저닝할 리소스를 지정하는 JSON 또는 YAML 형식의 텍스트 파일입니다. 이 섹션에는 다음 정보가 포함됩니다.

주제

- 스택 생성
- 스택 업데이트

스택 생성

다음 절차에서는 Application Manager를 사용하여 CloudFormation 스택을 생성하는 방법을 설명합니다. 스택은 템플릿을 기반으로 합니다. 스택을 생성할 때 기존 템플릿을 선택하거나 새 템플릿을 생성할 수 있습니다. 스택을 생성한 후 시스템은 즉시 스택에서 식별된 리소스 생성을 시도합니다. 시스템이 리소스를 성공적으로 프로비저닝하면 템플릿과 스택을 Application Manager 및 CloudFormation에서 보고 편집할 수 있습니다.

> **Note**
>
> Application Manager를 사용하여 스택을 생성하는 데는 요금이 부과되지 않지만 스택에 생성된 AWS 리소스에 대해서는 요금이 부과됩니다.

Application Manager를 사용하여 CloudFormation 스택 생성(콘솔)

다음 절차를 따라 AWS Management Console에서 Application Manager를 사용하여 스택을 생성합니다.

CloudFormation 스택을 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Application Manager를 선택합니다.
3. [애플리케이션(Applications)] 섹션에서 [CloudFormation 스택(CloudFormation stacks)]을 선택합니다.
4. [템플릿 준비(Prepare a template)] 섹션에서 옵션을 선택합니다. [기존 템플릿 사용(Use an existing template)]을 선택하면 [템플릿 선택(Choose a template)] 섹션의 탭을 사용하여 원하는 템플릿을 찾을 수 있습니다. 다른 옵션 중 하나를 선택한 경우 마법사를 완료하여 템플릿을 준비합니다.
5. [템플릿 세부 정보 지정(Specify template details)] 페이지에서 템플릿의 세부 정보를 확인하여 프로세스가 원하는 리소스를 생성하는지 확인합니다.

   - (옵션) [태그(Tags)] 섹션에서 템플릿에 하나 이상의 태그 키 이름/값 페어를 적용합니다.
   - 태그는 리소스에 할당하는 선택적 메타데이터입니다. 태그를 사용하여 용도, 소유자 또는 환경을 기준으로 하는 등 리소스를 다양한 방식으로 분류할 수 있습니다. Systems Manager 리소스 태그 지정에 대한 자세한 내용은 Systems Manager 리소스에 태그 지정 섹션을 참조하세요.
   - 다음을 선택합니다.
6. [스택 세부 정보 편집(Edit stack details)] 페이지의 [스택 이름(Stack name)]에 스택에 의해 생성된 리소스나 해당 용도를 식별하는 데 도움이 되는 이름을 입력합니다.

   - [파라미터(Parameters)] 섹션에는 템플릿에 지정된 선택적 및 필수 파라미터가 모두 포함되어 있습니다. 각 필드에 하나 이상의 파라미터를 입력합니다.
   - (옵션) [태그(Tags)] 섹션에서 스택에 하나 이상의 태그 키 이름/값 페어를 적용합니다.
   - (옵션) [권한(Permissions)] 섹션에서 AWS Identity and Access Management(IAM) 역할 이름 또는 IAM Amazon 리소스 이름(ARN)을 지정합니다. 시스템은 지정된 서비스 역할을 사용하여 스택에 지정된 모든 리소스를 생성합니다. IAM 역할을 지정하지 않으면 AWS CloudFormation은 시스템이 사용자 자격 증명에서 생성하는 임시 세션을 사용합니다. 이 IAM 역할에 대한 자세한 내용은 AWS CloudFormation 사용 설명서의 AWS CloudFormation 서비스 역할을 참조하세요.
   - 다음을 선택합니다.

7. [검토 및 프로비저닝(Review and provision)] 페이지에서 스택의 모든 세부 정보를 검토합니다. 변경하려면 이 페이지에서 [편집(Edit)] 버튼을 선택합니다.
8. [스택 프로비저닝(Provision stack)]을 선택합니다.

Application Manager는 [CloudFormation 스택(CloudFormation stacks)] 페이지와 스택 생성 및 배포 상태를 표시합니다. CloudFormation이 스택을 생성하고 프로비저닝하지 못하면 AWS CloudFormation 사용 설명서의 다음 주제를 참조하세요.

- 스택 상태 코드
- AWS CloudFormation 문제 해결

스택 리소스가 프로비저닝되고 실행되면 사용자는 리소스를 생성하는 기본 서비스를 사용하여 리소스를 직접 편집할 수 있습니다. 예를 들어 사용자는 Amazon Elastic Compute Cloud(Amazon EC2) 콘솔에서 CloudFormation 스택의 일부로 생성된 서버 인스턴스를 업데이트할 수 있습니다. 일부 변경 사항은 실수일 수 있으며, 일부는 시간에 민감한 작업 이벤트에 의도적으로 응답하는 것일 수 있습니다. 그와 무관하게 CloudFormation 외부에서의 변경 사항은 스택 업데이트 또는 삭제 작업을 복잡하게 만들 수 있습니다. 드리프트 감지 또는 드리프트 상태를 사용하여 구성 변경이 CloudFormation 관리 외부에서 이루어진 스택 리소스를 식별할 수 있습니다. 드리프트 상태에 대한 자세한 내용은 스택 및 리소스에 대한 비관리형 구성 변경 감지를 참조하세요.

Application Manager를 사용하여 CloudFormation 스택 생성(명령줄)

다음 AWS Command Line Interface(AWS CLI) 절차를 사용하여 Systems Manager에 SSM 문서로 저장된 CloudFormation 템플릿을 사용하여 스택을 프로비저닝합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다. 스택 생성을 위한 다른 AWS CLI 절차에 대한 자세한 내용은 AWS CloudFormation 사용 설명서의 스택 생성을 참조하세요.

시작하기 전 준비 사항

아직 하지 않은 경우 AWS CLI 또는 AWS Tools for PowerShell를 설치하고 구성합니다. 자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트 및 AWS Tools for PowerShell 설치를 참조하세요.

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" `
```

스택 업데이트

Application Manager에서 스택을 직접 편집하여 CloudFormation 스택에 업데이트를 배포할 수 있습니다. 직접 업데이트를 사용하면 템플릿 또는 입력 파라미터에 대한 업데이트를 지정할 수 있습니다. 변경 사항을 저장하고 배포한 후 CloudFormation은 지정한 변경 사항에 따라 AWS 리소스를 업데이트합니다.

변경 집합을 사용하여 업데이트하기 전에 CloudFormation이 스택에 적용할 변경 사항을 미리 볼 수 있습니다. 자세한 내용은 AWS CloudFormation 사용 설명서의 변경 집합을 사용하여 스택 업데이트를 참조하세요.

Application Manager에서 CloudFormation 스택을 업데이트하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Application Manager를 선택합니다.
3. [애플리케이션(Applications)] 섹션에서 [CloudFormation 스택(CloudFormation stacks)]을 선택합니다.
4. 목록에서 스택을 선택한 다음 [작업(Actions)], [스택 업데이트(Update stack)]를 선택합니다.
5. [템플릿 소스 지정(Specify template source)] 페이지에서 다음 옵션 중 하나를 선택한 후 [다음(Next)]을 선택합니다.

   - [스택에 현재 프로비저닝된 템플릿 코드 사용(Use the template code currently provisioned in the stack)]을 선택하여 템플릿을 봅니다. [버전(Versions)] 목록에서 템플릿 버전을 선택하고 [다음(Next)]을 선택합니다.

- [다른 템플릿으로 전환(Switch to a different template)]을 선택하여 스택에 대한 새 템플릿을 선택하거나 생성합니다.

6. 변경 템플릿을 마치면 [다음(Next)]을 선택합니다.
7. [스택 세부 정보 편집(Edit stack details)] 페이지에서 파라미터, 태그 및 권한을 편집할 수 있습니다. 스택 이름은 변경할 수 없습니다. 변경하고 [다음(Next)]을 선택합니다.
8. [검토 및 프로비저닝(Review and provision)] 페이지에서 스택의 모든 세부 정보를 검토한 다음 [스택 프로비저닝(Provision stack)]을 선택합니다.

Application Manager에서 클러스터 작업

이 섹션에는 AWS Systems Manager의 구성 요소인 Application Manager에서 Amazon Elastic Container Service(Amazon ECS) 및 Amazon Elastic Kubernetes Service(Amazon EKS) 컨테이너 클러스터 작업에 도움이 되는 주제가 포함되어 있습니다.

내용

Application Manager에서 Amazon ECS 작업

AWS Systems Manager의 기능인 Application Manager를 사용하면 Amazon Elastic Container Service(Amazon ECS) 클러스터 인프라를 보고 관리할 수 있습니다. Application Manager는 클러스터의 Amazon 리소스 이름(ARN)을 태그 값으로 사용하여 Amazon ECS 클러스터에 태그를 적용합니다. Application Manager는 클러스터의 컴퓨팅, 네트워킹 및 스토리지 리소스에 대한 구성 요소 런타임 보기를 제공합니다.

> **Note**
>
> Application Manager에서 컨테이너에 대한 작업 정보를 관리하거나 볼 수 없습니다. Amazon ECS 리소스를 호스팅하는 인프라에 대한 작업 정보만 관리하고 볼 수 있습니다.

이 페이지에서 수행할 수 있는 작업

이 페이지에서 다음 작업을 수행할 수 있습니다.

- [클러스터 관리(Manage cluster)]를 선택하여 Amazon ECS에서 클러스터를 엽니다.
- [모두 보기(View all)]를 선택하여 클러스터의 리소스 목록을 봅니다.
- [CloudWatch에서 보기(View in CloudWatch)]를 선택하여 Amazon CloudWatch에서 리소스 경보를 봅니다.
- Manage nodes(노드 관리) 또는 Manage Fargate profiles(Fargate 프로파일 관리)를 선택하여 Amazon ECS에서 이러한 리소스를 볼 수 있습니다.
- 리소스 ID를 선택하여 리소스 ID가 생성된 콘솔에서 리소스 ID에 대한 자세한 정보를 봅니다.
- 클러스터와 관련된 OpsItems 목록을 봅니다.
- 클러스터에서 실행된 실행서의 기록을 봅니다.

[ECS 클러스터(ECS cluster)]를 열려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Application Manager를 선택합니다.
3. [컨테이너 클러스터(Container clusters)] 섹션에서 [ECS 클러스터(ECS clusters)]를 선택합니다.
4. 목록에서 클러스터를 선택합니다. Application Manager가 [개요(Overview)] 탭을 엽니다.

Application Manager에서 Amazon EKS 작업

AWS Systems Manager의 기능인 Application Manager는 Amazon Elastic Kubernetes Service(Amazon EKS)와 통합되어 Amazon EKS 클러스터 인프라의 상태에 대한 정보를 제공합니다. Application Manager는 클러스터의 Amazon 리소스 이름(ARN)을 태그 값으로 사용하여 Amazon EKS 클러스터에 태그를 적용합니다. Application Manager는 클러스터의 컴퓨팅, 네트워킹 및 스토리지 리소스에 대한 구성 요소 런타임 보기를 제공합니다.

> **Note**
>
> Application Manager에서 Amazon EKS 포드 또는 컨테이너에 대한 작업 정보를 관리하거나 볼 수 없습니다. Amazon EKS 리소스를 호스팅하는 인프라에 대한 작업 정보만 관리하고 볼 수 있습니다.

이 페이지에서 수행할 수 있는 작업

이 페이지에서 다음 작업을 수행할 수 있습니다.

- [클러스터 관리(Manage cluster)]를 선택하여 Amazon EKS에서 클러스터를 엽니다.
- [모두 보기(View all)]를 선택하여 클러스터의 리소스 목록을 봅니다.
- [CloudWatch에서 보기(View in CloudWatch)]를 선택하여 Amazon CloudWatch에서 리소스 경보를 봅니다.
- Manage nodes(노드 관리) 또는 Manage Fargate profiles(Fargate 프로파일 관리)를 선택하여 Amazon EKS에서 이러한 리소스를 볼 수 있습니다.
- 리소스 ID를 선택하여 리소스 ID가 생성된 콘솔에서 리소스 ID에 대한 자세한 정보를 봅니다.
- 클러스터와 관련된 OpsItems 목록을 봅니다.
- 클러스터에서 실행된 실행서의 기록을 봅니다.

EKS 클러스터 애플리케이션을 열려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Application Manager를 선택합니다.
3. [컨테이너 클러스터(Container clusters)] 섹션에서 [EKS 클러스터(EKS clusters)]를 선택합니다.
4. 목록에서 클러스터를 선택합니다. Application Manager가 [개요(Overview)] 탭을 엽니다.

클러스터에 대한 실행서 작업

Systems Manager Automation 실행서를 사용하여 AWS Systems Manager의 기능인 Application Manager에서 AWS 리소스 문제를 해결할 수 있습니다. Application Manager 클러스터에서 [실행서 시작(Start runbook)]을 선택하면 클러스터의 리소스 유형에 따라 필터링된 실행서 목록이 표시됩니다. 시작할 실행서를 선택하면 Systems Manager에서 [자동화 문서 실행(Execute automation document)] 페이지를 엽니다.

시작하기 전 준비 사항

Application Manager에서 실행서를 시작하기 전에 다음을 수행합니다.

- 실행서를 시작하는 데 올바른 권한이 있는지 확인합니다. 자세한 내용은 Automation 설정 단원을 참조하십시오.
- 실행서 시작에 대한 Automation 절차 설명서를 검토합니다. 자세한 내용은 자동화 실행 단원을 참조하십시오.
- 한 번에 여러 리소스에서 실행서를 시작하려면 대상 및 속도 제어 사용에 대한 설명서를 검토합니다. 자세한 내용은 대규모로 자동화 실행 단원을 참조하십시오.

Application Manager에서 클러스터의 실행서를 시작하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Application Manager를 선택합니다.
3. [컨테이너 클러스터(Container clusters)] 섹션에서 컨테이너 유형을 선택합니다.
4. 목록에서 클러스터를 선택합니다. Application Manager가 [개요(Overview)] 탭을 엽니다.
5. 실행서(Runbooks) 탭에서 실행서 시작(Start runbook)을 선택하면 Application Manager가 새 탭에서 자동화 문서 실행(Execute automation document) 페이지를 엽니다. [자동화 문서 실행(Execute automation document)] 페이지의 옵션에 대한 자세한 내용은 자동화 실행 섹션을 참조하세요.

AWS AppConfig

AWS AppConfig 기능 플래그와 동적 구성을 사용하면 소프트웨어 빌더가 전체 코드를 배포하지 않고도 프로덕션 환경에서 애플리케이션 동작을 빠르고 안전하게 조정할 수 있습니다. AWS AppConfig는 소프트웨어 릴리스 빈도를 높이고, 애플리케이션 복원력을 개선하며, 새로운 문제를 더 빠르게 해결하는 데 도움이 됩니다. 기능 플래그를 사용하면 새로운 기능을 모든 사용자에게 완전히 배포하기 전에 새로운 기능을 점진적으로 사용자에게 릴리스하고 이러한 변경의 영향을 측정할 수 있습니다. 운영 플래그와 동적 구성을 사용하여 차단 목록, 허용 목록, 조절 제한, 로깅 상세도를 업데이트하고 기타 운영 조정을 수행하여 프로덕션 환경의 문제에 신속하게 대응할 수 있습니다.

자세한 내용은 AWS AppConfig 사용 설명서의 AWS AppConfig이란? 섹션을 참조하세요.

AWS Systems Manager Parameter Store

AWS Systems Manager의 기능인 Parameter Store는 구성 데이터 관리 및 암호 관리를 위한 안전한 계층적 스토리지를 제공합니다. 암호, 데이터베이스 문자열, Amazon Machine Image(AMI) ID, 라이선스 코드와 같은 데이터를 파라미터 값으로 저장할 수 있습니다. 값을 일반 텍스트 또는 암호화된 데이터로 저장할 수 있습니다. 파라미터를 생성할 때 지정한 고유 이름을 사용하여 스크립트, 명령, SSM 문서, 구성 및 자동화 워크플로에서 Systems Manager 파라미터를 참조할 수 있습니다. Parameter Store를 시작하려면 Systems Manager 콘솔을 엽니다. 탐색 창에서 Parameter Store를 선택합니다.

Parameter Store는 Secrets Manager와도 통합되어 있습니다. 이미 Parameter Store 파라미터 참조를 지원하는 다른 AWS 서비스을(를) 사용할 때 Secrets Manager 암호를 검색할 수 있습니다. 자세한 내용은 Parameter Store 파라미터에서 AWS Secrets Manager 암호 참조 단원을 참조하십시오.

> **Note**
>
> 암호 교체 수명 주기를 구현하려면 AWS Secrets Manager를 사용합니다. Secrets Manager를 사용하면 수명 주기 동안 데이터베이스 자격 증명, API 키 및 기타 보안 암호를 손쉽게 교체, 관리 및 검색할 수 있습니다. 자세한 내용은 AWS Secrets Manager 사용 설명서의 AWS Secrets Manager(이)란 무엇입니까? 섹션을 참조하세요.

Parameter Store가 조직에 주는 이점은 무엇인가요?

Parameter Store에서 제공하는 이점은 다음과 같습니다.

- 안전하고 확장 가능한 호스팅 방식 암호 관리 서비스를 사용합니다(관리할 서버가 없음).
- 데이터를 코드와 격리하여 보안 태세를 개선합니다.
- 구성 데이터 및 암호화된 문자열을 계층으로 저장하고 버전을 추적합니다.
- 세분화된 수준에서 액세스를 제어하고 감사합니다.
- Parameter Store는 AWS 리전의 여러 가용 영역에서 호스팅되기 때문에 파라미터를 안정적으로 저장합니다.

Parameter Store는 누가 사용해야 하나요?

- 구성 데이터를 중앙 집중식으로 관리하려는 모든 AWS 고객.
- 다양한 로그인 및 참조 스트림을 저장하려는 소프트웨어 개발자.
- 자신의 보안 암호와 암호가 변경되거나 변경되지 않을 때 알림을 받으려는 관리자.

Parameter Store에는 어떤 기능이 있나요?

- 변경 알림

  파라미터 및 파라미터 정책 모두에 대해 변경 알림을 구성하고 자동화된 작업을 호출할 수 있습니다. 자세한 내용은 Parameter Store 이벤트 기반 알림 설정 또는 작업 트리거 단원을 참조하십시오.
- 파라미터 구성

  파라미터에 태그를 지정하면 파라미터에 지정한 태그를 토대로 하나 이상의 파라미터를 개별적으로 식별할 수 있습니다. 예를 들어, 특정 환경이나 부서에 대한 파라미터에 태그를 지정할 수 있습니다. 자세한 내용은 Systems Manager 파라미터에 태그 지정 단원을 참조하십시오.

- 레이블 버전

  레이블을 생성하여 파라미터 버전에 대한 별칭을 연결할 수 있습니다. 레이블이 있으면 파라미터 버전이 여러 개일 때 버전의 용도를 기억하기 쉽습니다.

- 데이터 유효성 검사

  Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스를 가리키는 파라미터를 생성할 수 있으며 Parameter Store는 이러한 파라미터를 검증하여 예상 리소스 유형을 참조하는지, 리소스가 존재하는지, 고객에게 리소스 사용 권한이 있는지 확인합니다. 예를 들어 `aws:ec2:image` 데이터 형식의 값으로 Amazon Machine Image(AMI) ID가 있는 파라미터를 생성할 수 있습니다. Parameter Store은 파라미터 값이 AMI ID에 대한 형식 지정 요구 사항을 충족하고 지정된 AMI가 AWS 계정에서 사용 가능한지 확인하기 위해 비동기 검증 작업을 수행합니다.

- 참조 암호

  이미 Parameter Store 파라미터 참조를 지원하는 다른 AWS 서비스을(를) 사용할 때 Secrets Manager 암호를 검색할 수 있도록 Parameter Store을(를) AWS Secrets Manager와(과) 통합했습니다.

- 다른 계정과 파라미터 공유

  선택적으로 구성 데이터를 단일 AWS 계정으로 중앙 집중화하고 파라미터에 액세스해야 하는 다른 계정과 파라미터를 공유할 수 있습니다.

- 다른 AWS 서비스에서 액세스 가능

  Parameter Store 파라미터와 기타 Systems Manager 기능 및 AWS 서비스을(를) 사용하여 중앙 스토어에서 암호와 구성 데이터를 검색할 수 있습니다. 파라미터는 Run Command, Automation, State Manager, AWS Systems Manager의 기능과 같은 Systems Manager 기능과 함께 사용할 수 있습니다. 또한 다음을 비롯한 다양한 다른 AWS 서비스에서도 파라미터를 참조할 수 있습니다.

  - 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

- 다른 AWS 서비스와(과)의 통합

암호화, 알림, 모니터링 및 감사 기능을 위해 다음 AWS 서비스와(과)의 통합을 구성합니다.

- AWS Key Management Service (AWS KMS)
- Amazon Simple Notification Service(Amazon SNS)
- Amazon CloudWatch: 자세한 내용은 파라미터 및 파라미터 정책에 대해 EventBridge 구성 섹션을 참조하세요.
- Amazon EventBridge: 자세한 내용은 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 및 참조: Systems Manager용 Amazon EventBridge 이벤트 패턴 및 유형를 참조하세요.
- AWS CloudTrail: 자세한 내용은 AWS CloudTrail을 사용하여 AWS Systems ManagerAPI 호출을 로깅 섹션을 참조하세요.

파라미터란 무엇인가요?

Parameter Store 파라미터는 텍스트 블록, 이름 목록, 암호, AMI ID, 라이선스 키 등과 같이 Parameter Store에 저장되는 모든 데이터입니다. 스크립트, 명령 및 SSM 문서에서 이 데이터를 중앙에서 안전하게 참조할 수 있습니다.

파라미터를 참조할 때 다음 규칙을 사용하여 파라미터 이름을 지정합니다.

{{ssm:*parameter-name*}}

> **Note**
>
> 파라미터는 다른 파라미터의 값에 참조되거나 중첩될 수 없습니다. 파라미터 값에 {{}} 또는 {{ssm:*parameter-name*}}을 포함할 수 없습니다.

Parameter Store는 String, StringList, SecureString 등 세 가지 유형의 파라미터를 지원합니다.

단, 파라미터를 생성하거나 업데이트할 때 파라미터 값을 일반 텍스트로 입력해야 하며 Parameter Store는 입력한 텍스트에 대해 검증을 수행하지 않습니다. 그러나 String 파라미터의 경우 데이터 형식을 aws:ec2:image로 지정할 수 있으며 Parameter Store는 입력한 값이 Amazon EC2 AMI에 적합한 형식인지 검증합니다(예: ami-12345abcdeEXAMPLE).

파라미터 유형: String

기본적으로 String 파라미터는 입력한 텍스트 블록으로 구성됩니다. 예:

- `abc123`
- `Example Corp`
- `<img src="images/bannerImage1.png"/>`

파라미터 유형: StringList

`StringList` 파라미터에는 다음 예제와 같이 쉼표로 구분된 값 목록이 포함되어 있습니다.

```
Monday,Wednesday,Friday
```

```
CSV,TSV,CLF,ELF,JSON
```

파라미터 유형: SecureString

`SecureString` 파라미터는 안전한 방식으로 저장되고 참조되어야 하는 모든 민감한 데이터를 뜻합니다. 암호나 라이선스 키처럼 사용자가 일반 텍스트로 수정하거나 참조해서는 안 되는 데이터가 있는 경우 `SecureString` 데이터 형식을 사용하여 이 파라미터를 생성합니다.

> **Important**
>
> `String` 또는 `StringList` 파라미터에 중요한 데이터를 저장하지 않습니다. 암호화된 상태로 유지해야 하는 모든 중요한 데이터의 경우 `SecureString` 파라미터 유형만 사용하십시오. 자세한 내용은 SecureString 파라미터 생성(AWS CLI) 단원을 참조하십시오.

다음 시나리오에 `SecureString` 파라미터를 사용하는 것이 좋습니다.

- 명령, 함수, 에이전트 로그 또는 CloudTrail 로그에 일반 텍스트로 값을 노출하지 않고 AWS 서비스 전반에 걸쳐 데이터/파라미터를 사용하고 싶은 경우.
- 민감한 데이터에 액세스하는 대상을 제어하고 싶은 경우.
- 민감한 데이터에 액세스하는 시점을 감사하고 싶은 경우(CloudTrail).
- 민감한 데이터를 암호화하고 싶은 경우와 자체 암호화 키로 액세스를 관리하고 싶은 경우.

> **Important**
>
> `SecureString` 파라미터의 값만 암호화됩니다. 파라미터 이름, 설명 및 기타 속성은 암호화되지 않습니다.

암호, 애플리케이션 암호, 기밀 구성 데이터 또는 보호가 필요한 기타 유형의 데이터와 같이 암호화하려는 텍스트 데이터에 SecureString 파라미터 유형을 사용할 수 있습니다. SecureString 데이터는 AWS KMS 키를 사용하여 암호화되고 복호화합니다. AWS에서 제공하는 기본 KMS 키를 사용하거나 자체 AWS KMS key를 생성하여 사용할 수 있습니다. (SecureString 파라미터에 대한 사용자 액세스를 제한하려면 고유의 AWS KMS key를 사용합니다. 자세한 내용은 AWS 기본 키 및 고객 관리형 키 사용에 대한 IAM 권한 섹션을 참조하세요.)

다른 AWS 서비스와(과) 함께 SecureString 파라미터를 사용할 수도 있습니다. 다음 예에서는 Lambda 함수가 GetParameters API를 사용하여 SecureString 파라미터를 검색합니다.

```
from __future__ import print_function

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 암호화 및 요금

파라미터를 생성할 때 SecureString 파라미터 형식을 선택하면 Systems Manager가 AWS KMS를 사용하여 파라미터 값을 암호화합니다.

> **Important**
>
> Parameter Store는 대칭 암호화 KMS 키만 지원합니다. 비대칭 암호화 KMS 키를 사용하여 파라미터를 암호화할 수 없습니다. KMS 키가 대칭인지 비대칭인지 확인하는 것과 관련된 도움말은 AWS Key Management Service Developer Guide의 Identifying symmetric and asymmetric KMS keys를 참조하세요.

SecureString 파라미터 생성 시 Parameter Store에서 비용이 부과되지 않지만 AWS KMS 암호화 사용에 대한 요금이 적용됩니다. 자세한 내용은 AWS Key Management Service 요금을 참조하십시오.

AWS 관리형 키 및 고객 관리형 키에 대한 자세한 내용은 AWS Key Management Service 개발자 가이드의 AWS Key Management Service 개념을 참조하세요. Parameter Store 및 AWS KMS 암호화에 대한 자세한 내용은 AWS Systems ManagerParameter Store의 AWS KMS사용 방법을 참조하십시오.

> **Note**
>
> AWS 관리형 키를 보려면 AWS KMS DescribeKey 작업을 사용합니다. 이 AWS Command Line Interface(AWS CLI) 예제에서는 DescribeKey를 사용하여 AWS 관리형 키를 봅니다.
>
> ```
> aws kms describe-key --key-id alias/aws/ssm
> ```

추가 정보

- SecureString 파라미터를 생성하고 도메인에 노드 조인(PowerShell)
- Parameter Store를 사용하여 CodeDeploy의 암호 및 구성 데이터에 안전하게 액세스
- Amazon EC2 Systems Manager Parameter Store에 대한 흥미로운 글

Parameter Store 설정

AWS Systems Manager의 기능인 Parameter Store에서 파라미터를 설정하기 전에 먼저 계정에 있는 사용자에게 지정한 작업을 수행할 수 있는 권한을 제공하는 AWS Identity and Access Management(IAM) 정책을 구성합니다. 이 섹션에서는 IAM 콘솔을 사용하여 수동으로 이러한 정책을 구성하는 방법과 이러한 정책을 사용자 및 사용자 그룹에 할당하는 방법에 대한 정보를 제공합니다. 또한 관리형 노드에서 실행될 수 있는 파라미터 작업을 제어하는 정책을 생성하고 할당할 수 있습니다. 이 섹션에서는 Systems Manager 파라미터 변경 사항에 대한 알림을 받을 수 있는 Amazon EventBridge 규칙을 생성하는 방법에 대한 정보도 제공합니다. 또한 EventBridge 규칙을 사용하면 Parameter Store의 변경 사항을 기준으로 AWS에서 다른 작업을 호출할 수 있습니다.

내용

- IAM 정책을 사용하여 Systems Manager 파라미터에 대한 액세스 제한
- 파라미터 티어 관리
- Parameter Store 처리량 증가 또는 재설정
- Parameter Store 이벤트 기반 알림 설정 또는 작업 트리거

IAM 정책을 사용하여 Systems Manager 파라미터에 대한 액세스 제한

AWS Identity and Access Management(IAM)를 사용하여 AWS Systems Manager 파라미터에 대한 액세스를 제한합니다. 구체적으로 말하면, 다음 API 작업에 대한 액세스를 제한하는 IAM 정책을 생성합니다.

- DeleteParameter
- DeleteParameters
- DescribeParameters
- GetParameter
- GetParameters
- GetParameterHistory
- GetParametersByPath
- PutParameter

IAM 정책을 사용하여 Systems Manager 파라미터에 대한 액세스를 제한할 때는 제한적인 IAM 정책을 생성하고 사용하는 것이 좋습니다. 예를 들어 다음 정책을 통해 사용자는 제한된 리소스에 대한 `DescribeParameters` 및 `GetParameters` API 작업을 호출할 수 있습니다. 즉, 사용자가 `prod-*`로 시작하는 모든 파라미터를 사용하고 관련 정보를 얻을 수 있습니다.

```
{
    "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**
>
> 경로에 대해 액세스 권한이 있는 사용자는 해당 경로의 모든 수준에 액세스할 수 있습니다. 예를 들어 /a 경로에 대한 액세스 권한이 있는 사용자는 /a/b에도 액세스할 수 있습니다. 사용자가 IAM에서 파라미터 /a/b에 대한 액세스를 명시적으로 거부한 경우에도 /a에 대해 재귀적으로 GetParametersByPath API 작업을 호출하고 /a/b를 볼 수 있습니다.

신뢰할 수 있는 관리자에게는 다음 예와 같은 정책을 사용하여 모든 Systems Manager 파라미터 API 작업에 액세스할 수 있는 권한을 부여할 수 있습니다. 이 정책은 사용자에게 dbserver-prod-*로 시작하는 모든 프로덕션 파라미터에 완전히 액세스할 수 있는 권한을 제공합니다.

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

권한 거부

각 API는 고유하며 개별적으로 허용하거나 거부할 수 있는 고유한 작업과 권한을 가집니다. 어떠한 정책의 명시적 거부도 허용을 무시합니다.

> **Note**
>
> 기본 AWS Key Management Service(AWS KMS) 키에는 AWS 계정 내의 모든 IAM 보안 주체에 대해 `Decrypt` 권한이 있습니다. 계정의 `SecureString` 파라미터에 대해 여러 액세스 수준을 사용하려는 경우 기본 키를 사용하지 않는 것이 좋습니다.

파라미터 값을 검색하는 모든 API 작업이 동일하게 동작하게 하려면 정책에서 `GetParameter*`와 같은 패턴을 사용합니다. 다음 예에서는 `prod-*`로 시작하는 모든 파라미터에 대해 `GetParameter`, `GetParameters`, `GetParameterHistory` 및 `GetParametersByPath`를 거부하는 방법을 보여줍니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "ssm:GetParameter*"
            ],
            "Resource": "arn:aws:ssm:us-east-2:123456789012:parameter/prod-*"
        }
    ]
}
```

다음 예에서는 사용자가 `prod-*`로 시작하는 모든 파라미터에 대해 다른 명령을 수행하도록 허용하면서 일부 명령을 거부하는 방법을 보여줍니다.

```
{
    "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**
>
> 파라미터 기록에는 현재 파라미터를 포함하여 모든 파라미터 버전이 포함됩니다. 따라서 `GetParameter`, `GetParameters` 및 `GetParameterByPath`에 대한 사용자 권한은 거부되었지만 `GetParameterHistory`에 대한 권한은 허용된 경우 `GetParameterHistory`를 사용하여 `SecureString` 파라미터를 포함한 현재 파라미터를 볼 수 있습니다.

노드에서 특정 파라미터만 실행하도록 허용

관리형 노드에서 사용자가 지정한 파라미터만 실행할 수 있도록 액세스를 제어할 수 있습니다.

파라미터를 생성할 때 `SecureString` 파라미터 유형을 선택하면 Systems Manager가 AWS KMS를 사용하여 파라미터 값을 암호화합니다. AWS KMS는 AWS 관리형 키 또는 고객 관리형 키를 사용하여 값을 암호화합니다. AWS KMS 및 AWS KMS key에 대한 자세한 내용은 AWS Key Management Service Developer Guide를 참조하세요.

AWS CLI에서 다음 명령을 실행하여 AWS 관리형 키를 볼 수 있습니다.

```
aws kms describe-key --key-id alias/aws/ssm
```

다음 예에서는 노드가 `prod-`로 시작하는 파라미터에 대해서만 파라미터 값을 가져오도록 허용합니다. 파라미터가 `SecureString` 파라미터이면 노드가 AWS KMS를 사용하여 문자열을 복호화합니다.

> **Note**
>
> 앞의 예에서와 같은 인스턴스 정책이 IAM의 인스턴스 역할에 할당됩니다. 사용자 및 인스턴스에 정책을 할당하는 방법을 포함하여 Systems Manager 기능 액세스를 구성하는 방법에 대한 자세한 내용은 EC2 인스턴스에 Systems Manager 사용 섹션을 참조하세요.

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

AWS 기본 키 및 고객 관리형 키 사용에 대한 IAM 권한

Parameter Store `SecureString` 파라미터는 AWS KMS 키를 사용하여 암호화되고 복호화됩니다. AWS KMS key 또는 AWS에서 제공하는 기본 KMS 키를 사용하여 `SecureString` 파라미터를 암호화하도록 선택할 수 있습니다.

고객 관리형 키를 사용하는 경우, 파라미터 또는 파라미터 경로에 대한 사용자 액세스 권한을 부여하는 IAM 정책에서 키에 대한 명시적 `kms:Encrypt` 권한을 제공해야 합니다. 예를 들어 다음 정책은 사용

자가 지정된 AWS 리전 및 AWS 계정에서 prod-로 시작하는 SecureString 파라미터를 생성, 업데이트 및 확인할 수 있도록 허용합니다.

```
{
    "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]지정된 고객 관리형 키를 사용하여 암호화된 고급 파라미터를 만들려면 kms:GenerateDataKey 권한이 필요합니다.

이와 대조적으로 고객 계정 내의 모든 사용자는 기본 AWS 관리형 키에 액세스할 수 있습니다. 이 기본 키를 사용하여 SecureString 파라미터를 암호화하고 사용자가 SecureString 파라미터로 작업하지 않도록 하려면, 다음 정책 예제와 같이 IAM 정책에서 기본 키에 대한 액세스를 명시적으로 거부해야 합니다.

> **Note**
>
> AWS 관리형 키 페이지의 AWS KMS 콘솔에서 기본 키의 Amazon 리소스 이름(ARN)을 찾을 수 있습니다. 기본 키는 [별칭(Alias)] 열에 aws/ssm이 있는 키입니다.

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

계정의 SecureString 파라미터에 대한 세분화된 액세스 제어가 필요한 경우 고객 관리형 키를 사용하여 이러한 파라미터에 대한 액세스를 보호하고 제한해야 합니다. 또한 AWS CloudTrail를 사용하여 SecureString 파라미터 활동을 모니터링하는 것이 좋습니다.

자세한 내용은 다음 주제를 참조하세요.

- IAM 사용 설명서의 정책 평가 로직
- AWS Key Management Service Developer Guide의 Using key policies in AWS KMS
- AWS CloudTrail 사용 설명서의 CloudTrail 이벤트 기록을 사용하여 이벤트 보기

파라미터 티어 관리

AWS Systems Manager의 기능인 Parameter Store에는 표준 파라미터와 고급 파라미터가 포함됩니다. 표준 파라미터 티어(기본 티어) 또는 고급 파라미터 티어에서 사용하도록 파라미터를 개별적으로 구성할 수 있습니다.

언제든 표준 파라미터를 고급 파라미터로 변경할 수 있지만 고급 파라미터를 표준 파라미터로 되돌릴 수 없습니다. 이는 고급 파라미터를 표준 파라미터로 되돌리면 시스템이 파라미터의 크기를 8KB에서 4KB로 자르기 때문에 데이터가 손실을 일으키기 때문입니다. 되돌리면 파라미터에 연결된 모든 정책도 제거됩니다. 또한 고급 파라미터는 표준 파라미터와 다른 암호화 형식을 사용합니다. 자세한 내용은 AWS Key Management Service 개발자 안내서에서 AWS Systems Manager Parameter Store가 AWS KMS를 사용하는 방식을 참조하세요.

고급 파라미터가 더 이상 필요하지 않거나 더 이상 고급 파라미터에 요금을 부과하지 않으려면 파라미터를 삭제하고 새 표준 파라미터로 다시 생성합니다.

다음 표에서는 여러 티어의 차이점을 설명합니다.

	표준	고급
허용되는 총 파라미터 수 (AWS 계정 및 AWS 리전당)	10,000	100,000건
파라미터 값의 최대 크기	4KB	8KB
파라미터 정책 사용 가능	아니요	예 자세한 내용은 파라미터 정책 할당 단원을 참조하십시오.
비용	추가 요금 없음	요금 적용 자세한 내용은 Parameter Store에 대한 AWS Systems Manager 요금을 참조하세요.

주제

- 기본 파라미터 티어 지정
- 표준 파라미터를 고급 파라미터로 변경

기본 파라미터 티어 지정

파라미터 생성 또는 업데이트(즉 `PutParameter` 작업) 요청에 대해 파라미터 티어를 지정하여 요청에 사용할 수 있습니다. 다음은 AWS Command Line Interface(AWS CLI) 사용을 보여주는 예입니다.

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"
```

요청에서 티어를 지정할 때마다 Parameter Store이 요청에 따라 파라미터를 생성하거나 업데이트합니다. 그러나 요청에서 티어를 명시적으로 지정하지 않는 경우 Parameter Store 기본 티어 설정이 파라미터가 생성되는 티어를 결정합니다.

Parameter Store 사용을 시작할 때 표준 파라미터 티어이 기본 티어입니다. 고급 파라미터 티어를 사용하는 경우 다음 중 하나를 기본으로 지정할 수 있습니다.

- 고급: 이 옵션을 사용하면 파라미터 스토어가 모든 요청을 고급 파라미터로 평가합니다.
- [지능형 계층화(Intelligent-Tiering)]: 이 옵션을 사용하면 Parameter Store가 각 요청을 평가하여 파라미터가 표준인지 아니면 고급인지 결정합니다.

  요청에 고급 파라미터를 필요로 하는 옵션이 포함되지 않는 경우 파라미터는 표준 파라미터 티어에서 생성됩니다. 고급 파라미터가 필요한 하나 이상의 옵션이 요청에 포함되는 경우 Parameter Store는 고급 파라미터 티어에서 파라미터를 생성합니다.

지능형 계층화의 이점

다음은 지능형 계층화를 기본 티어로 선택할 수 있는 이유입니다.

비용 관리 - 지능형 계층화를 사용하면 고급 파라미터가 절대적으로 필요하지 않는 한 항상 표준 파라미터를 생성하여 파라미터 관련 비용을 관리할 수 있습니다.

고급 파라미터 티어로 자동 업그레이드 - 표준 파라미터를 고급 파라미터로 업그레이드해야 하는 코드 변경 시 지능형 계층화를 사용하여 변환을 처리할 수 있습니다. 업그레이드를 처리하기 위해 코드를 변경할 필요가 없습니다.

다음은 자동 업그레이드의 몇 가지 예입니다.

- AWS CloudFormation 템플릿은 파라미터가 실행될 때 많은 파라미터를 프로비저닝합니다. 이 프로세스를 통해 표준 파라미터 티어의 10,000개 파라미터 할당량에 도달하면 지능형 계층화가 고급 파라미터 티어로 자동으로 업그레이드하여 AWS CloudFormation 프로세스가 중단되지 않습니다.
- 파라미터에 증명서 값을 저장하고 증명서 값을 정기적으로 교체하면 콘텐츠가 표준 파라미터 티어의 4KB 할당량 미만이 됩니다. 교체 증명서 값이 4KB를 초과하는 경우 지능형 계층화는 파라미터를 고급 파라미터 티어로 자동으로 업그레이드합니다.
- 기존의 많은 표준 파라미터를 고급 파라미터 티어가 필요한 파라미터 정책에 연결하려고 합니다. 파라미터를 업데이트하기 위한 모든 호출에 `--tier Advanced` 옵션을 포함시킬 필요 없이 지능형 계층화는 파라미터를 고급 파라미터 티어로 자동으로 업그레이드합니다. 지능형 계층화 옵션은 고급 파라미터 티어의 기준이 도입될 때마다 파라미터를 표준에서 고급으로 업그레이드합니다.

고급 파라미터가 필요한 옵션에는 다음이 포함됩니다.

- 파라미터의 콘텐츠 크기는 4KB 이상입니다.
- 파라미터는 파라미터 정책을 사용합니다.
- 현재 AWS 리전의 AWS 계정에 10,000개 이상의 파라미터가 이미 존재합니다.

기본 티어 옵션

기본으로 지정할 수 있는 티어 옵션에는 다음이 포함됩니다.

- [표준(Standard)] - Parameter Store를 사용하기 시작할 때 표준 파라미터 티어가 기본 티어입니다. 표준 파라미터 티어를 사용하여 AWS 계정의 각 AWS 리전에 대해 10,000개의 파라미터를 생성할 수 있습니다. 각 파라미터의 콘텐츠 크기는 최대 4KB에 해당될 수 있습니다. 표준 파라미터는 파라미터 정책을 지원하지 않습니다. 표준 파라미터 티어 사용에 따르는 추가 요금은 없습니다. 표준을 기본 티어로 선택하면 Parameter Store이 티어를 지정하지 않는 요청에 대해 항상 표준 파라미터 생성을 시도합니다.
- [고급(Advanced)] - 고급 파라미터 티어를 사용하면 AWS 계정의 각 AWS 리전에 대해 최대 100,000개 파라미터를 생성할 수 있습니다. 각 파라미터의 콘텐츠 크기는 최대 8KB에 해당될 수 있습니다. 고급 파라미터는 파라미터 정책을 지원합니다. 고급 파라미터 티어 사용에는 요금이 따릅니다. 자세한 내용은 Parameter Store에 대한 AWS Systems Manager 요금을 참조하세요. 고급을 기본 티어로

선택하면 Parameter Store이 티어를 지정하지 않는 요청에 대해 항상 고급 파라미터 생성을 시도합니다.

> **Note**
>
> 고급 파라미터 티어를 선택하는 경우 생성하는 고급 파라미터에 대해 AWS가 사용자의 계정에 요금을 부과하도록 명시적으로 승인합니다.

- [지능형 계층화(Intelligent-Tiering)] - [지능형 계층화(Intelligent-Tiering)] 옵션을 사용하면 Parameter Store가 요청 내용에 따라 표준 파라미터 티어 또는 고급 파라미터 티어의 사용 여부를 결정할 수 있습니다. 예를 들어 콘텐츠가 4KB 미만인 파라미터를 생성하도록 명령을 실행하고 AWS 계정의 현재 AWS 리전에 10,000개 미만의 파라미터가 있는 경우 파라미터 정책을 지정하지 않으면 표준 파라미터가 생성됩니다. 예를 들어 콘텐츠가 4KB 이상인 파라미터를 생성하도록 명령을 실행하고 AWS 계정의 현재 AWS 리전에 10,000개 이상의 파라미터가 있거나 파라미터 정책을 지정하는 경우 고급 파라미터가 생성됩니다.

> **Note**
>
> 지능형 계층화를 선택하는 경우 생성되는 고급 파라미터에 대해 AWS가 사용자의 계정에 요금을 부과하도록 명시적으로 승인합니다.

Parameter Store 기본 티어 설정은 언제라도 변경할 수 있습니다.

Parameter Store 기본 티어를 지정할 권한 구성

다음 중 하나를 수행하여 Parameter Store의 기본 파라미터를 변경할 AWS Identity and Access Management(IAM)의 권한이 있는지 확인합니다.

- `AdministratorAccess` 정책을 IAM 엔터티(사용자, 그룹 또는 역할 등)에 연결해야 합니다.
- 다음 API 작업을 사용하여 기본 티어 설정을 변경할 권한이 있는지 확인합니다.
  - GetServiceSetting
  - UpdateServiceSetting
  - ResetServiceSetting

사용자가 AWS 계정의 특정 AWS 리전에 있는 파라미터의 기본 티어 설정을 보고 변경할 수 있도록 IAM 엔터티에 다음 권한을 부여합니다.

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

관리자는 다음 권한을 할당하여 읽기 전용 권한을 지정할 수 있습니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetServiceSetting"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Deny",
            "Action": [
                "ssm:ResetServiceSetting",
                "ssm:UpdateServiceSetting"
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

액세스 권한을 제공하려면 사용자, 그룹 또는 역할에 권한을 추가하세요:

- AWS IAM Identity Center의 사용자 및 그룹:

  권한 세트를 생성합니다. AWS IAM Identity Center 사용 설명서의 권한 세트 생성의 지침을 따르세요.
- ID 제공자를 통해 IAM에서 관리되는 사용자:

  ID 페더레이션을 위한 역할을 생성합니다. IAM 사용 설명서의 서드 파티 자격 증명 공급자의 역할 만들기(연합)의 지침을 따르세요.
- IAM 사용자:
  - 사용자가 맡을 수 있는 역할을 생성합니다. IAM 사용 설명서에서 IAM 사용자의 역할 생성의 지침을 따르세요.
  - (권장되지 않음)정책을 사용자에게 직접 연결하거나 사용자를 사용자 그룹에 추가합니다. IAM 사용 설명서에서 사용자(콘솔)에 권한 추가의 지침을 따르세요.

Parameter Store 기본 티어 지정 또는 변경(콘솔)

다음 절차에서는 Systems Manager 콘솔을 사용하여 현재 AWS 계정 및 AWS 리전에 대한 기본 파라미터를 지정하거나 변경하는 방법을 보여줍니다.

> **Tip**
>
> 파라미터를 아직 만들지 않은 경우 AWS Command Line Interface(AWS CLI) 또는 AWS Tools for Windows PowerShell을 사용하여 기본 파라미터 티어를 변경할 수 있습니다. 자세한 내용은 Parameter Store 기본 티어 지정 또는 변경(AWS CLI) 및 Parameter Store 기본 티어 지정 또는 변경(PowerShell) 섹션을 참조하세요.

Parameter Store 기본 티어를 지정하거나 변경하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Parameter Store를 선택합니다.
3. 설정 탭을 선택합니다.
4. [기본 티어 변경(Change default tier)]을 선택합니다.

5. 다음 옵션 중 하나를 선택합니다.

   - 표준
   - 고급
   - 지능형 계층화

   이러한 옵션에 대한 자세한 내용은 기본 파라미터 티어 지정 섹션을 참조하세요.
6. 메시지를 검토하고 확인을 선택하십시오.

나중에 기본 티어 설정을 변경하려면 이 절차를 반복하고 다른 기본 티어 옵션을 지정합니다.

Parameter Store 기본 티어 지정 또는 변경(AWS CLI)

다음 절차에서는 AWS CLI를 사용하여 현재 AWS 계정 및 AWS 리전에 대한 기본 파라미터 티어 설정을 변경하는 방법을 보여줍니다.

AWS CLI를 사용하여 Parameter Store 기본 티어를 지정하거나 변경하려면

1. AWS CLI를 열고 다음 명령을 실행하여 AWS 계정의 특정 AWS 리전에 대한 기본 파라미터 티어 설정을 변경합니다.

   ```
   aws ssm update-service-setting --setting-id arn:aws:ssm:region:account-id:servicesetting/ssm/parameter-store/default-parameter-tier --setting-value tier-option
   ```

   ##은 미국 동부(오하이오) 리전의 us-east-2 같이 AWS Systems Manager가 지원하는 AWS 리전의 식별자를 나타냅니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

   ## ## 값에는 Standard, Advanced 및 Intelligent-Tiering이 있습니다. 이러한 옵션에 대한 자세한 내용은 기본 파라미터 티어 지정 섹션을 참조하세요.

   명령이 성공해도 결과는 없습니다.
2. 다음 명령을 실행하여 현재 AWS 계정 및 AWS 리전의 Parameter Store에 대한 현재 기본 파라미터 티어 서비스 설정을 확인합니다.

   ```
   aws ssm get-service-setting --setting-id arn:aws:ssm:region:account-id:servicesetting/ssm/parameter-store/default-parameter-tier
   ```

시스템은 다음과 유사한 정보를 반환합니다.

```
{
    "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"
    }
}
```

기본 티어 설정을 다시 변경하려면 이 절차를 반복하고 다른 `SettingValue` 옵션을 지정합니다.

Parameter Store 기본 티어 지정 또는 변경(PowerShell)

다음 절차에서는 Tools for Windows PowerShell을 사용하여 Amazon Web Services 계정의 특정 AWS 리전에 대한 기본 파라미터 티어 설정을 변경하는 방법을 보여줍니다.

PowerShell을 사용하여 Parameter Store 기본 티어를 지정하거나 변경하려면

1. AWS Tools for PowerShell(Tools for PowerShell)을 사용하여 현재 AWS 계정 및 AWS 리전의 Parameter Store 기본 티어를 변경합니다.

   ```
   Update-SSMServiceSetting -SettingId "arn:aws:ssm:region:account-id:servicesetting/
   ssm/parameter-store/default-parameter-tier" -SettingValue "tier-option" -
   Region region
   ```

   *##*은 미국 동부(오하이오) 리전의 `us-east-2` 같이 AWS Systems Manager가 지원하는 AWS 리전의 식별자를 나타냅니다. 지원되는 *##* 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

   ## ## 값에는 `Standard`, `Advanced` 및 `Intelligent-Tiering`이 있습니다. 이러한 옵션에 대한 자세한 내용은 기본 파라미터 티어 지정 섹션을 참조하세요.

   명령이 성공해도 결과는 없습니다.

2. 다음 명령을 실행하여 현재 AWS 계정 및 AWS 리전의 Parameter Store에 대한 현재 기본 파라미터 티어 서비스 설정을 확인합니다.

```
Get-SSMServiceSetting -SettingId "arn:aws:ssm:region:account-id:servicesetting/ssm/
parameter-store/default-parameter-tier" -Region region
```

   *##*은 미국 동부(오하이오) 리전의 `us-east-2` 같이 AWS Systems Manager가 지원하는 AWS 리전의 식별자를 나타냅니다. 지원되는 *##* 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

   시스템은 다음과 유사한 정보를 반환합니다.

```
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
```

기본 티어 설정을 다시 변경하려면 이 절차를 반복하고 다른 `SettingValue` 옵션을 지정합니다.

표준 파라미터를 고급 파라미터로 변경

기존 표준 파라미터를 고급 파라미터로 변경하려면 다음 절차를 사용하십시오. 새 고급 파라미터를 생성하는 방법에 대한 자세한 내용은 Systems Manager 파라미터 생성 섹션을 참조하세요.

표준 파라미터를 고급 파라미터로 변경하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Parameter Store를 선택합니다.
3. 파라미터를 선택한 후 편집을 선택합니다.
4. 설명에 이 파라미터에 대한 설명을 입력합니다.
5. 고급을 선택합니다.
6. 값에 이 파라미터의 값을 입력합니다. 고급 파라미터의 최대 값 제한은 8KB입니다.
7. 변경 사항 저장(Save changes)을 선택합니다.

Parameter Store 처리량 증가 또는 재설정

Parameter Store 처리량을 늘리면 AWS Systems Manager의 기능인 Parameter Store에서 처리할 수 있는 초당 최대 트랜잭션 수(TPS)가 증가합니다. 높아진 처리량 덕분에 더 높은 볼륨으로 Parameter Store를 작동하여 여러 파라미터에 동시 액세스해야 하는 애플리케이션 및 워크로드를 지원할 수 있습니다. 설정(Settings) 탭에서 최대 처리량까지 할당량을 늘릴 수 있습니다.

기본 최대 처리량과 최대 한도에 대한 자세한 내용은 AWS Systems Manager 엔드포인트 및 할당량을 참조하세요.

처리량 할당량을 늘리면 AWS 계정에 요금이 부과됩니다. 자세한 내용은 AWS Systems Manager 요금을 참조하십시오.

> **Note**
>
> Parameter Store 처리량 설정은 현재 AWS 계정 및 AWS 리전의 모든 IAM 사용자가 생성하는 모든 트랜잭션에 적용됩니다. 처리량 설정은 표준 및 고급 파라미터에 적용됩니다.

주제

Parameter Store 처리량 변경을 위한 권한 구성

다음 중 하나를 수행하여 IAM에서 Parameter Store 처리량을 변경할 권한이 있는지 확인합니다.

- `AdministratorAccess` 정책이 IAM 엔터티(사용자, 그룹 또는 역할)에 연결되어 있는지 확인합니다.
- 다음 API 작업을 사용하여 처리량 서비스 설정을 변경할 권한이 있는지 확인합니다.
  - GetServiceSetting
  - UpdateServiceSetting
  - ResetServiceSetting

사용자가 AWS 계정의 특정 AWS 리전에 있는 파라미터의 파라미터 처리량 설정을 보고 변경할 수 있도록 IAM 엔터티에 다음 권한을 부여합니다.

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

관리자는 다음 권한을 할당하여 읽기 전용 권한을 지정할 수 있습니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetServiceSetting"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Deny",
            "Action": [
                "ssm:ResetServiceSetting",
                "ssm:UpdateServiceSetting"
            ],
            "Resource": "*"
```

```
        }
    ]
}
```

액세스 권한을 제공하려면 사용자, 그룹 또는 역할에 권한을 추가하세요:

- AWS IAM Identity Center의 사용자 및 그룹:

  권한 세트를 생성합니다. AWS IAM Identity Center 사용 설명서의 권한 세트 생성의 지침을 따르세요.
- ID 제공자를 통해 IAM에서 관리되는 사용자:

  ID 페더레이션을 위한 역할을 생성합니다. IAM 사용 설명서의 서드 파티 자격 증명 공급자의 역할 만들기(연합)의 지침을 따르세요.
- IAM 사용자:
  - 사용자가 맡을 수 있는 역할을 생성합니다. IAM 사용 설명서에서 IAM 사용자의 역할 생성의 지침을 따르세요.
  - (권장되지 않음)정책을 사용자에게 직접 연결하거나 사용자를 사용자 그룹에 추가합니다. IAM 사용 설명서에서 사용자(콘솔)에 권한 추가의 지침을 따르세요.

처리량 증가 또는 재설정(콘솔)

다음 절차에서는 Systems Manager를 사용하여 Parameter Store가 현재 AWS 계정 및 AWS 리전에 대해 처리할 수 있는 초당 트랜잭션 수를 늘리는 방법을 보여줍니다. 높은 처리량이 더 이상 필요하지 않거나 요금이 더 이상 청구되지 않게 하려는 경우 표준 설정으로 되돌릴 방법을 보여줍니다.

> **Tip**
>
> 파라미터를 아직 생성하지 않은 경우 AWS Command Line Interface(AWS CLI) 또는 AWS Tools for Windows PowerShell를 사용하여 처리량을 높일 수 있습니다. 자세한 내용은 처리량 증가 또는 재설정(AWS CLI) 및 처리량 증가 또는 재설정(PowerShell) 섹션을 참조하세요.

Parameter Store 처리량을 높이거나 재설정하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Parameter Store를 선택합니다.
3. 설정 탭을 선택합니다.

4. 처리량을 늘리려면 한도 설정을 선택합니다.

   -또는-

   기본 한도로 되돌리려면 한도 재설정을 선택합니다.

5. 한도를 늘리려면 다음 작업을 수행합니다.

   - 이 설정을 변경하면 AWS 계정에 요금이 부과된다는 점을 수락함 확인란을 선택합니다.
   - Set limit(한도 설정)을 선택하십시오.

   -또는-

   한도를 기본값으로 재설정하는 경우 다음 작업을 수행합니다.

   - 기본 처리량 한도로 재설정하면 Parameter Store에서 초당 처리되는 트랜잭션 수가 줄어든다는 점에 동의함 확인란을 선택합니다.
   - 한도 재설정을 선택합니다.

처리량 증가 또는 재설정(AWS CLI)

다음 절차에서는 AWS CLI를 사용하여 Parameter Store이 현재 AWS 계정 및 AWS 리전에 대해 처리할 수 있는 초당 트랜잭션 수를 늘리는 방법을 보여줍니다. 기본 한도로 되돌릴 수도 있습니다.

AWS CLI를 사용하여 Parameter Store 처리량을 높이려면

1. AWS CLI를 열고 다음 명령을 실행하여 Parameter Store가 현재 AWS 계정 및 AWS 리전에서 처리할 수 있는 초당 트랜잭션을 늘립니다.

   ```
   aws ssm update-service-setting --setting-id arn:aws:ssm:region:account-
   id:servicesetting/ssm/parameter-store/high-throughput-enabled --setting-value true
   ```

   명령이 성공해도 결과는 없습니다.

2. 다음 명령을 실행하여 현재 AWS 계정 및 AWS 리전의 Parameter Store에 대한 현재 처리량 서비스 설정을 확인합니다.

   ```
   aws ssm get-service-setting --setting-id arn:aws:ssm:region:account-
   id:servicesetting/ssm/parameter-store/high-throughput-enabled
   ```

시스템은 다음과 유사한 정보를 반환합니다.

```
{
    "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"
    }
}
```

높은 처리량이 더 이상 필요하지 않거나 요금이 더 이상 청구되지 않도록 하려면 표준 설정으로 되돌릴 수 있습니다. 설정을 되돌리려면 다음 명령을 실행하십시오.

```
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"
    }
}
```

처리량 증가 또는 재설정(PowerShell)

다음 절차에서는 Tools for Windows PowerShell을 사용하여 Parameter Store가 현재 AWS 계정 및 AWS 리전에 대해 처리할 수 있는 초당 트랜잭션 수를 늘리는 방법을 보여줍니다. 기본 한도로 되돌릴 수도 있습니다.

PowerShell을 사용하여 Parameter Store 처리량을 늘리려면

1. AWS Tools for PowerShell(Tools for PowerShell)을 사용하여 현재 AWS 계정 및 AWS 리전의 Parameter Store 처리량을 늘립니다.

   ```
   Update-SSMServiceSetting -SettingId "arn:aws:ssm:region:account-id:servicesetting/
   ssm/parameter-store/high-throughput-enabled" -SettingValue "true" -Region region
   ```

   명령이 성공해도 결과는 없습니다.
2. 다음 명령을 실행하여 현재 AWS 계정 및 AWS 리전의 Parameter Store에 대한 현재 처리량 서비스 설정을 확인합니다.

   ```
   Get-SSMServiceSetting -SettingId "arn:aws:ssm:region:account-id:servicesetting/ssm/
   parameter-store/high-throughput-enabled" -Region region
   ```

   시스템은 다음과 유사한 정보를 반환합니다.

   ```
   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
   ```

높은 처리량이 더 이상 필요하지 않거나 요금이 더 이상 청구되지 않도록 하려면 표준 설정으로 되돌릴 수 있습니다. 설정을 되돌리려면 다음 명령을 실행하십시오.

```
Reset-SSMServiceSetting -SettingId "arn:aws:ssm:region:account-id:servicesetting/ssm/
parameter-store/high-throughput-enabled" -Region region
```

시스템은 다음과 유사한 정보를 반환합니다.

```
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
```

Parameter Store 이벤트 기반 알림 설정 또는 작업 트리거

이 섹션의 주제에서는 Amazon EventBridge 및 Amazon Simple Notification Service(Amazon SNS)를 사용하여 AWS Systems Manager 파라미터의 변경 사항을 알리는 방법을 설명합니다. 파라미터 또는 파라미터 레이블 버전이 생성, 업데이트 또는 삭제될 때 알리는 EventBridge 규칙을 생성할 수 있습니다. 이벤트는 최선의 작업을 기반으로 발생됩니다. 파라미터가 만료될 때, 파라미터가 만료 예정일 때 또는 지정된 기간 동안 파라미터가 변경되지 않았을 때와 같이 파라미터 정책과 관련된 변경 사항 또는 상태에 대한 알림을 받을 수 있습니다.

> **Note**
>
> 파라미터 정책은 고급 파라미터 티어를 사용하는 파라미터에 대해 사용할 수 있습니다. 요금이 적용됩니다. 자세한 내용은 파라미터 정책 할당 및 파라미터 티어 관리 섹션을 참조하세요.

이 섹션의 주제에서는 특정 파라미터 이벤트에 대해 대상에서 기타 작업을 시작하는 방법도 설명합니다. 예를 들어, 파라미터가 만료되거나 삭제될 때 자동으로 재생성하는 AWS Lambda 함수를 실행할 수 있습니다. 데이터베이스 암호가 업데이트되면 Lambda 함수를 호출하도록 알림을 설정할 수 있습니다. Lambda 함수는 강제로 데이터베이스 연결을 재설정하거나 새 암호로 다시 연결할 수 있습니다. EventBridge는 또한 Run Command 명령, Automation 실행 및 기타 많은 AWS 서비스의 작업 실행을 지원합니다. Run Command 및 Automation 모두 AWS Systems Manager의 기능입니다. 자세한 내용은 https://docs.aws.amazon.com/eventbridge/latest/userguide/Amazon EventBridge 사용 설명서를 참조하세요.

시작하기 전

생성한 규칙에 대한 대상 작업을 지정하는 데 필요한 리소스를 생성합니다. 예를 들어 알림 보내기에 대한 규칙을 생성하는 경우 먼저 Amazon SNS 주제를 생성합니다. 자세한 내용은 Amazon Simple Notification Service 개발자 안내서의 Amazon SNS 시작하기를 참조하세요.

파라미터 및 파라미터 정책에 대해 EventBridge 구성

이 주제에서는 다음을 설명합니다.

- AWS 계정에서 하나 이상의 파라미터에 대해 발생하는 이벤트를 기반으로 대상을 호출하는 EventBridge 규칙을 생성하는 방법.
- AWS 계정에서 하나 이상의 파라미터 정책에 대해 발생하는 이벤트를 기반으로 대상을 호출하는 EventBridge 규칙을 생성하는 방법. 고급 파라미터를 생성할 때 파라미터가 만료될 때, 파라미터가

만료되기 전 알림을 수신할 때 및 파라미터가 변경되지 않았다는 알림을 보내야 하기 전에 대기하는 시간을 지정합니다. 다음 절차를 사용하여 이러한 이벤트에 대한 알림을 설정합니다. 자세한 내용은 파라미터 정책 할당 및 파라미터 티어 관리을(를) 참조하세요.

Systems Manager 파라미터 또는 파라미터 정책에 대해 EventBridge 규칙 구성

1. Amazon EventBridge 콘솔(https://console.aws.amazon.com/events/)을 엽니다.
2. 탐색 창에서 규칙(Rules)을 선택한 후 규칙 생성(Create rule)을 선택합니다.

   -또는-

   EventBridge 홈 페이지가 처음으로 열리면 규칙 생성(Create rule)을 선택합니다.
3. 규칙에 대해 이름과 설명을 입력합니다.

   규칙은 동일한 지역과 동일한 이벤트 버스의 다른 규칙과 동일한 이름을 가질 수 없습니다.
4. 이벤트 버스(Event bus)에서 이 규칙과 연결할 이벤트 버스를 선택합니다. 이 규칙이 자신의 AWS 계정에서 오는 발생하는 해당 이벤트에서 시작되도록 하려면 기본값(Default)을 선택합니다. 계정의 AWS 서비스이(가) 이벤트를 출력하면 항상 계정의 기본 이벤트 버스로 이동합니다.
5. 규칙 유형(Rule type)의 경우 기본값인 이벤트 패턴이 있는 규칙(Rule with an event pattern)을 선택한 채로 둡니다.
6. 다음을 선택합니다.
7. Event source(이벤트 소스)의 경우 기본값인 AWS events or EventBridge partner events(이벤트 또는 EventBridge 파트너 이벤트)를 선택된 상태로 둡니다. 샘플 이벤트(Sample event) 섹션은 건너뛸 수 있습니다.
8. 이벤트 패턴(Event pattern)에서 다음을 수행합니다.
   - 사용자 지정 패턴(JSON 편집기)(Custom patterns (JSON editor))을 선택합니다.
   - 이벤트 패턴(Event pattern)에서 파라미터 또는 파라미터 정책에 대해 규칙을 생성하는지 여부에 따라 상자에 있는 다음 콘텐츠 중 하나를 붙여넣습니다.

     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"
        ]
    }
}
```

- 다음 샘플에 표시된 것과 같이 조치를 취할 작업 및 파라미터에 대한 콘텐츠를 수정합니다.

Parameter

이 예시에서는 이름이 /Oncall 및 /Project/Teamlead인 파라미터 중 하나가 업데이트 될 때 작업이 실시됩니다.

```
{
    "source": [
        "aws.ssm"
    ],
    "detail-type": [
        "Parameter Store Change"
    ],
    "detail": {
        "name": [
            "/Oncall",
            "/Project/Teamlead"
        ],
        "operation": [
            "Update"
        ]
    }
}
```

Parameter policy

이 예시에서는 이름이 /OncallDuties인 파라미터가 만료되고 삭제될 때마다 작업이 실시됩니다.

```
{
    "source": [
        "aws.ssm"
    ],
    "detail-type": [
        "Parameter Store Policy Action"
    ],
    "detail": {
        "parameter-name": [
            "/OncallDuties"
        ],
        "policy-type": [
            "Expiration"
        ]
```

```
    }
}
```

9. 다음을 선택합니다.
10. 대상(Target 1)에서 대상 유형과 지원되는 리소스를 선택합니다. 예를 들어 [SNS 주제(SNS topic)]를 선택하는 경우 [주제(Topic)]를 선택합니다. CodePipeline을 선택한 경우 파이프라인 ARN(Pipeline ARN)에 파이프라인 ARN을 입력합니다. 필요에 따라 추가 구성 값을 입력합니다.

> **Tip**
>
> 규칙에 추가 대상이 필요한 경우 다른 대상 추가(Add another target)를 선택합니다.

11. 다음을 선택합니다.
12. (선택 사항) 규칙에 대해 하나 이상의 태그를 입력하십시오. 자세한 정보는 Amazon EventBridge 사용 설명서의 Amazon EventBridge 태그를 참조하십시오.
13. 다음을 선택합니다.
14. Create rule을 선택합니다.

추가 정보

- 환경 간 간단한 구성 업데이트를 위해 파라미터 레이블 사용
- Amazon EventBridge 사용 설명서에서 Tutorial: Use EventBridge to relay events to AWS Systems Manager Run Command
- Amazon EventBridge 사용 설명서의 자습서: AWS Systems Manager 자동화를 EventBridge 대상으로 설정

Parameter Store 작업

이 단원에서는 태그 파라미터를 구성 및 생성하는 방법과 다양한 버전의 파라미터를 생성하는 방법을 설명합니다. AWS Systems Manager 콘솔, Amazon Elastic Compute Cloud(Amazon EC2) 콘솔 또는 AWS Command Line Interface(AWS CLI)를 사용하여 파라미터를 생성하고 작업할 수 있습니다. 파라미터에 대한 자세한 내용은 파라미터란 무엇인가요? 섹션을 참조하세요.

주제

- Systems Manager 파라미터 생성
- Systems Manager 파라미터 검색

- 파라미터 정책 할당
- 파라미터 계층 구조 작업
- 파라미터 레이블 작업
- 파라미터 버전 작업
- 공유 파라미터 작업
- Run Command 명령을 사용하여 파라미터 작업
- Amazon Machine Image ID에 대한 기본 파라미터 지원
- Systems Manager 파라미터 삭제

Systems Manager 파라미터 생성

다음 주제의 정보를 사용하여 AWS Systems Manager 콘솔, AWS Command Line Interface(AWS CLI) 또는 AWS Tools for Windows PowerShell(Tools for Windows PowerShell)로 Systems Manager 파라미터를 생성할 수 있습니다.

이 섹션에서는 테스트 환경에서 Parameter Store로 파라미터를 생성, 저장 및 실행하는 방법을 보여줍니다. 또한 Parameter Store을(를) 다른 Systems Manager 기능 및 AWS 서비스와(과) 함께 사용하는 방법을 설명합니다. 자세한 내용은 파라미터란 무엇인가요?을(를) 참조하세요.

파라미터 이름의 요구 사항 및 제약 조건 정보

이 주제의 정보를 사용하면 파라미터를 생성할 때 파라미터 이름에 유효한 값을 지정하는 데 도움이 됩니다.

이 정보는 AllowedPattern, Description, KeyId, Overwrite, Type 및 Value 값에 대한 정보도 제공하는 AWS Systems Manager API Reference의 PutParameter 주제의 세부 정보를 보완합니다.

파라미터 이름에 대한 요구 사항 및 제약 조건은 다음과 같습니다.

- 대/소문자 구분: 파라미터 이름은 대/소문자를 구분합니다.
- 공백: 파라미터 이름에는 공백이 포함될 수 없습니다.
- 유효한 값: 파라미터 이름은 `a-zA-Z0-9_.-`과 같은 기호와 문자로 구성될 수 있습니다.

  또한 슬래시 문자(/)는 파라미터 이름의 계층 구조를 나타내는 데 사용됩니다. 예: `/Dev/Production/East/Project-ABC/MyParameter`
- [유효한 AMI 형식(Valid AMI format)]: `String` 파라미터의 데이터 형식으로 `aws:ec2:image`를 선택하는 경우 입력한 ID가 AMI ID 형식 `ami-12345abcdeEXAMPLE`에 유효한지 검증해야 합니다.

- [정규화(Fully qualified)]: 계층에서 파라미터를 생성하거나 참조하는 경우 선행 슬래시 문자(/)를 포함합니다. 계층의 일부인 파라미터를 참조하는 경우 초기 슬래시(/)를 포함한 전체 계층 경로를 지정합니다.
  - 정규화된 파라미터 이름: `MyParameter1`, `/MyParameter2`, `/Dev/Production/East/Project-ABC/MyParameter`
  - 정규화되지 않은 파라미터 이름: `MyParameter3/L1`
- 길이: 생성할 수 있는 파라미터 이름의 최대 길이는 1011자입니다. 여기에는 `arn:aws:ssm:us-east-2:111122223333:parameter/`와 같이 지정한 이름 앞에 오는 ARN의 문자가 포함됩니다.
- [접두사(Prefixes)]: 파라미터 이름은 "aws" 또는 "ssm"(대/소문자 구별 안 함)으로 시작할 수 없습니다. 예를 들어 다음 이름의 파라미터를 생성하려고 하면 예외와 함께 실패합니다.
  - `awsTestParameter`
  - `SSM-testparameter`
  - `/aws/testparam1`

  > **Note**
  >
  > SSM 문서, 명령 또는 스크립트에서 파라미터를 지정하면 다음 예제와 같이 `ssm`이 구문의 일부로 포함됩니다. 예: {{ssm:*parameter-name*}} and {{ ssm:*parameter-name* }}(예: `{{ssm:MyParameter}}` 및 `{{ ssm:MyParameter }}`.)

- [고유성(Uniqueness)]: 파라미터 이름은 AWS 리전 내에서 고유해야 합니다. 예를 들어 Systems Manager는 다음 항목이 동일한 리전에 존재하는 경우 별도의 파라미터로 취급합니다.
  - `/Test/TestParam1`
  - `/TestParam1`

  다음 예도 고유한 항목입니다.
  - `/Test/TestParam1/Logpath1`
  - `/Test/TestParam1`

  하지만 다음 예제는 동일한 리전에 있는 경우 고유하지 않습니다.
  - `/TestParam1`
  - `TestParam1`
- 계층 구조 깊이: 파라미터 계층 구조를 지정하는 경우 계층 구조에는 최대 15개의 레벨이 있을 수 있습니다. 원하는 계층 구조 레벨에 파라미터를 정의할 수 있습니다. 다음의 두 예제 모두 구조적으로 유효합니다.

- `/Level-1/L2/L3/L4/L5/L6/L7/L8/L9/L10/L11/L12/L13/L14/parameter-name`
- `parameter-name`

다음 파라미터를 생성하려고 하면 `HierarchyLevelLimitExceededException` 예외로 인해 실패할 것입니다.

- `/Level-1/L2/L3/L4/L5/L6/L7/L8/L9/L10/L11/L12/L13/L14/L15/L16/parameter-name`

Important

경로에 대해 액세스 권한이 있는 사용자는 해당 경로의 모든 수준에 액세스할 수 있습니다. 예를 들어 `/a` 경로에 대한 액세스 권한이 있는 사용자는 `/a/b`에도 액세스할 수 있습니다. 사용자가 AWS Identity and Access Management(IAM)에서 파라미터`/a/b` 에 대한 액세스를 명시적으로 거부한 경우에도 `/a` 및 뷰 `/a/b`에 대해 GetParametersByPath API 작업을 재귀적으로 호출할 수 있습니다.

주제

- Systems Manager 파라미터 생성(콘솔)
- Systems Manager 파라미터 생성(AWS CLI)
- Systems Manager 파라미터 생성(Tools for Windows PowerShell)

Systems Manager 파라미터 생성(콘솔)

AWS Systems Manager 콘솔을 사용하여 `String`, `StringList` 및 `SecureString` 파라미터 유형을 생성하고 실행할 수 있습니다. 파라미터를 삭제한 후 최소 30초 동안 기다리고 같은 이름의 파라미터를 생성합니다.

Note

파라미터는 생성된 해당 파라미터가 생성된 AWS 리전에서만 사용할 수 있습니다.

다음 절차에서는 Parameter Store 콘솔에서 파라미터를 생성하는 과정을 안내합니다. 콘솔에서 `String`, `StringList` 및 `SecureString` 파라미터 유형을 생성할 수 있습니다.

파라미터를 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Parameter Store를 선택합니다.
3. 파라미터 생성(Create parameter)을 선택합니다.
4. 이름 상자에 계층 구조 및 이름을 입력합니다. 예를 들면 **/Test/helloWorld**를 입력합니다.

   파라미터 계층 구조에 대한 자세한 내용은 파라미터 계층 구조 작업 섹션을 참조하세요.
5. 설명 상자에 이 파라미터가 테스트 파라미터라는 설명을 입력합니다.
6. Parameter tier(파라미터 티어)에서 표준 또는 고급을 선택합니다. 고급 파라미터에 대한 자세한 내용은 파라미터 티어 관리 섹션을 참조하세요.
7. 유형에서 String, StringList 또는 SecureString을 선택합니다.

   - [문자열(String)]을 선택하면 [데이터 형식(Data type)] 필드가 표시됩니다. Amazon Machine Image(AMI)의 리소스 ID를 보관할 파라미터를 생성하는 경우 `aws:ec2:image`를 선택합니다. 그렇지 않으면 기본값인 `text`를 선택한 상태로 유지합니다.
   - [SecureString]을 선택한 경우 [KMS 키 ID(KMS Key ID)] 필드가 표시됩니다. AWS Key Management Service AWS KMS key ID, AWS KMS key Amazon 리소스 이름(ARN), 별칭 이름 또는 별칭 ARN을 제공하지 않으면 시스템은 Systems Manager의 AWS 관리형 키인 `alias/aws/ssm`을 사용합니다. 이 키를 사용하지 않으려면 고객 관리형 키를 사용할 수 있습니다. AWS 관리형 키 및 고객 관리형 키에 대한 자세한 내용은 AWS Key Management Service 개발자 가이드의 AWS Key Management Service 개념을 참조하세요. Parameter Store 및 AWS KMS 암호화에 대한 자세한 내용은 AWS Systems ManagerParameter Store의 AWS KMS사용 방법을 참조하십시오.

     Important

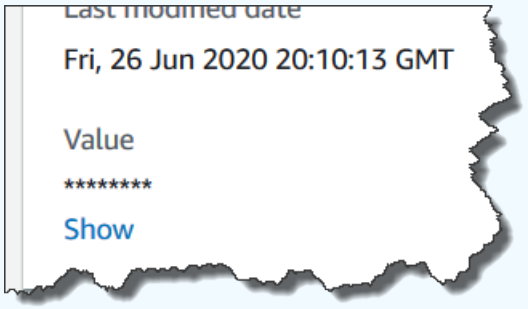
     Parameter Store는 대칭 암호화 KMS 키만 지원합니다. 비대칭 암호화 KMS 키를 사용하여 파라미터를 암호화할 수 없습니다. KMS 키가 대칭인지 비대칭인지 확인하는 것과 관련된 도움말은 AWS Key Management Service Developer Guide의 Identifying symmetric and asymmetric KMS keys를 참조하세요.
   - 고객 관리형 키 별칭 이름 또는 별칭 ARN과 함께 `key-id` 파라미터를 사용하여 콘솔에 `SecureString` 파라미터를 생성할 때 별칭 앞에 접두사 `alias/`를 지정합니다. ARN 예제는 다음과 같습니다.

```
arn:aws:kms:us-east-2:123456789012:alias/abcd1234-ab12-cd34-ef56-abcdeEXAMPLE
```

별칭 이름 예제는 다음과 같습니다.

```
alias/MyAliasName
```

8. 값 상자에 값을 입력합니다. 예를 들어 **`This is my first parameter`** 또는 **`ami-0dbf5ea29aEXAMPLE`**을 입력합니다.

> **Note**
>
> 파라미터는 다른 파라미터의 값에 참조되거나 중첩될 수 없습니다. 파라미터 값에 `{{}}` 또는 `{{ssm:parameter-name}}`을 포함할 수 없습니다.
> [SecureString]을 선택한 경우 파라미터의 값은 나중에 파라미터 [개요(Overview)] 탭에서 볼 때 기본적으로 마스킹됩니다("******"). [표시(Show)]를 선택하여 파라미터 값을 표시합니다.
>
> 
>

9. (선택 사항) 태그 영역에서 하나 이상의 태그 키-값 페어를 파라미터에 적용합니다.

   태그는 리소스에 할당하는 선택적 메타데이터입니다. 태그를 사용하면 용도, 소유자 또는 환경을 기준으로 하는 등 리소스를 다양한 방식으로 분류할 수 있습니다. 예를 들어 Systems Manager 파라미터에 태그를 지정하여 적용할 리소스 유형, 환경 또는 파라미터가 참조하는 구성 데이터의 유형을 식별할 수 있습니다. 이 경우 다음 키-값 페어를 지정할 수 있습니다.

   - `Key=Resource,Value=S3bucket`
   - `Key=OS,Value=Windows`
   - `Key=ParameterType,Value=LicenseKey`

10. 파라미터 생성(Create parameter)을 선택합니다.

11. 파라미터 목록에서 방금 생성한 파라미터 이름을 선택합니다. 개요 탭에서 세부 정보를 확인합니다. `SecureString` 파라미터를 생성한 경우 [표시(Show)]를 선택하여 암호화되지 않은 값을 확인합니다.

> **Note**
>
> 고급 파라미터는 표준 파라미터로 변경할 수 없습니다. 고급 파라미터가 더 이상 필요하지 않거나 더 이상 고급 파라미터에 요금을 부과하지 않으려면 파라미터를 삭제하고 새 표준 파라미터로 다시 생성합니다.

Systems Manager 파라미터 생성(AWS CLI)

AWS Command Line Interface(AWS CLI)를 사용하여 `String`, `StringList` 및 `SecureString` 파라미터 유형을 생성할 수 있습니다. 파라미터를 삭제한 후 최소 30초 동안 기다리고 같은 이름의 파라미터를 생성합니다.

파라미터는 다른 파라미터의 값에 참조되거나 중첩될 수 없습니다. 파라미터 값에 `{{}}` 또는 `{{ssm:parameter-name}}`을 포함할 수 없습니다.

> **Note**
>
> 파라미터는 생성된 해당 파라미터가 생성된 AWS 리전에서만 사용할 수 있습니다.

주제

- String 파라미터 생성(AWS CLI)
- StringList 파라미터 생성(AWS CLI)
- SecureString 파라미터 생성(AWS CLI)
- 여러 줄 파라미터 생성(AWS CLI)

`String` 파라미터 생성(AWS CLI)

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.

2. 다음 명령을 실행하여 String 형식 파라미터를 생성합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   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"
   ```

   -또는-

   다음 명령을 실행하여 파라미터 값으로 Amazon Machine Image(AMI) ID를 포함하는 파라미터를 생성합니다.

   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` 옵션은 계층 구조를 지원합니다. 계층 구조에 대한 자세한 내용은 파라미터 계층 구조 작업 섹션을 참조하세요.

AMI ID를 포함하는 파라미터를 생성하는 경우에만 `--data-type` 옵션을 지정해야 합니다. 입력한 파라미터 값이 올바른 형식의 Amazon Elastic Compute Cloud(Amazon EC2) AMI ID인지 검증합니다. 다른 모든 파라미터의 경우 기본 데이터 형식은 `text`이며 값을 지정하는 것은 옵션입니다. 자세한 내용은 Amazon Machine Image ID에 대한 기본 파라미터 지원 단원을 참조하십시오.

> **Important**
>
> 성공할 경우 명령은 파라미터의 버전 번호를 반환합니다. 예외: `aws:ec2:image`를 데이터 형식으로 지정한 경우 응답에서 새 버전 번호는 파라미터 값이 이미 검증되었음을 의미하지는 않습니다. 자세한 내용은 Amazon Machine Image ID에 대한 기본 파라미터 지원 단원을 참조하십시오.

다음 예제에서는 두 개의 키-값 페어 태그를 파라미터에 추가합니다.

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\"}]
```

다음 예에서는 이름에 파라미터 계층 구조를 사용하여 일반 텍스트 String 파라미터를 생성합니다. 이 명령은 파라미터의 버전 번호를 반환합니다. 파라미터 계층 구조에 대한 자세한 내용은 파라미터 계층 구조 작업 섹션을 참조하세요.

Linux & macOS

계층 구조에 없는 파라미터

```
aws ssm put-parameter \
    --name "golden-ami" \
    --type "String" \
    --value "ami-12345abcdeEXAMPLE"
```

계층 구조에 있는 파라미터

```
aws ssm put-parameter \
    --name "/amis/linux/golden-ami" \
    --type "String" \
    --value "ami-12345abcdeEXAMPLE"
```

Windows

계층 구조에 없는 파라미터

```
aws ssm put-parameter ^
    --name "golden-ami" ^
    --type "String" ^
    --value "ami-12345abcdeEXAMPLE"
```

계층 구조에 있는 파라미터

```
aws ssm put-parameter ^
    --name "/amis/windows/golden-ami" ^
    --type "String" ^
    --value "ami-12345abcdeEXAMPLE"
```

3. 다음 명령을 실행하여 최신 파라미터 값을 보고 새 파라미터의 세부 정보를 확인합니다.

```
aws ssm get-parameters --names "/Test/IAD/helloWorld"
```

시스템은 다음과 같은 정보를 반환합니다.

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

다음 명령을 실행하여 파라미터 값을 변경합니다. 이 명령은 파라미터의 버전 번호를 반환합니다.

```
aws ssm put-parameter --name "/Test/IAD/helloWorld" --value "My updated 1st parameter"
 --type String --overwrite
```

다음 명령을 실행하여 파라미터 값 기록을 확인합니다.

```
aws ssm get-parameter-history --name "/Test/IAD/helloWorld"
```

다음 명령을 실행하여 명령에 이 파라미터를 사용합니다.

```
aws ssm send-command --document-name "AWS-RunShellScript" --parameters '{"commands":
["echo {{ssm:/Test/IAD/helloWorld}}"]}' --targets "Key=instanceids,Values=instance-ids"
```

파라미터 값만 검색하려면 다음 명령을 실행합니다.

```
aws ssm get-parameter --name testDataTypeParameter --query "Parameter.Value"
```

`get-parameters`를 사용하여 파라미터 값만 검색하려면 다음 명령을 실행합니다.

```
aws ssm get-parameters --names "testDataTypeParameter" --query "Parameters[*].Value"
```

다음 명령을 실행하여 파라미터 메타데이터를 확인합니다.

```
aws ssm describe-parameters --filters "Key=Name,Values=/Test/IAD/helloWorld"
```

> **Note**
>
> 이름은 대문자여야 합니다.

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "Parameters": [
        {
            "Name": "helloworld",
            "Type": "String",
            "LastModifiedUser": "arn:aws:iam::123456789012:user/JohnDoe",
            "LastModifiedDate": 1494529763.156,
            "Version": 1,
            "Tier": "Standard",
            "Policies": []
        }
    ]
}
```

`StringList` 파라미터 생성(AWS CLI)

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.
2. 다음 명령을 실행하여 파라미터를 생성합니다. 각 *`example resource placeholder`*를 사용자의 정보로 바꿉니다.

   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**
>
> 성공할 경우 명령은 파라미터의 버전 번호를 반환합니다.

이 예제는 두 개의 키-값 페어 태그를 파라미터에 추가합니다. 로컬 시스템의 운영 체제 유형에 따라 다음 명령 중 하나를 실행합니다. 로컬 Windows 시스템에서 실행할 버전에는 명령줄 도구에서 명령을 실행하는 데 필요한 이스케이프 문자("\")가 포함되어 있습니다.

다음은 파라미터 계층 구조를 사용하는 `StringList` 예입니다.

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**
>
> StringList의 항목들은 쉼표(,)로 구분해야 합니다. 목록에서 항목을 이스케이프하기 위해 다른 문장 부호나 특수 기호를 사용할 수 없습니다. 쉼표가 필요한 파라미터 값이 있는 경우 String 유형을 사용합니다.

3. get-parameters 명령을 실행하여 파라미터의 세부 정보를 확인합니다. 예:

```
aws ssm get-parameters --name "/IAD/ERP/Oracle/addUsers"
```

SecureString 파라미터 생성(AWS CLI)

SecureString 파라미터를 생성하려면 다음 절차에 따르십시오. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

> **Important**
>
> SecureString 파라미터의 값만 암호화됩니다. 파라미터 이름, 설명 및 기타 속성은 암호화되지 않습니다.

> **Important**
>
> Parameter Store는 대칭 암호화 KMS 키만 지원합니다. 비대칭 암호화 KMS 키를 사용하여 파라미터를 암호화할 수 없습니다. KMS 키가 대칭인지 비대칭인지 확인하는 것과 관련된 도움말은 AWS Key Management Service Developer Guide의 Identifying symmetric and asymmetric KMS keys를 참조하세요.

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.
2. 다음 명령 중 하나를 실행하여 SecureString 데이터 형식을 사용하는 파라미터를 생성합니다.

   Linux & macOS

   기본 AWS 관리형 키을(를) 사용하여 **SecureString** 파라미터 생성

```
aws ssm put-parameter \
    --name "parameter-name" \
    --value "parameter-value" \
    --type "SecureString"
```

고객 관리형 키를 사용하는 **SecureString** 파라미터 생성

```
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"
```

사용자 정의 AWS KMS 키를 사용하는 **SecureString** 파라미터 생성

```
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

기본 AWS 관리형 키을(를) 사용하여 **SecureString** 파라미터 생성

```
aws ssm put-parameter ^
    --name "parameter-name" ^
    --value "parameter-value" ^
    --type "SecureString"
```

고객 관리형 키를 사용하는 **SecureString** 파라미터 생성

```
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"
```

사용자 정의 AWS KMS 키를 사용하는 **SecureString** 파라미터 생성

```
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"
```

계정 및 리전에서 AWS 관리형 키 키를 사용하여 SecureString 파라미터를 생성하는 경우 --key-id 파라미터에 값을 제공할 필요가 없습니다.

> **Note**
>
> AWS 계정 및 AWS 리전에 할당된 AWS KMS key를 사용하려면 명령에서 key-id 파라미터를 제거합니다. AWS KMS keys에 대한 자세한 내용은 AWS Key Management Service Developer Guide의 AWS Key Management Service Concepts 섹션을 참조하세요.

계정에 할당된 AWS 관리형 키 대신 고객 관리형 키를 사용하려면 --key-id 파라미터를 사용하여 키를 지정합니다. 파라미터는 다음 KMS 파라미터 형식을 지원합니다.

- 키 Amazon 리소스 이름(ARN) 예:

  arn:aws:kms:us-east-2:123456789012:key/*key-id*

- 별칭 ARN 예:

  arn:aws:kms:us-east-2:123456789012:alias/*alias-name*

- 키 ID 예:

  12345678-1234-1234-1234-123456789012

- 별칭 이름 예:

  alias/MyAliasName

AWS Management Console 또는 AWS KMS API를 사용하여 고객 관리형 키를 생성할 수 있습니다. 다음 AWS CLI 명령은 AWS 계정의 현재 AWS 리전에 고객 관리형 키를 생성합니다.

```
aws kms create-key
```

다음 형식의 명령을 통해 방금 생성한 키를 사용하여 SecureString 파라미터를 생성합니다.

다음은 암호 파라미터 및 AWS KMS key에 난독화된 이름(3l3vat3131)을 사용하는 예입니다.

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. 다음 명령을 실행하여 파라미터의 세부 정보를 확인합니다.

with-decryption 파라미터를 지정하지 않거나 no-with-decryption 파라미터를 지정하면 명령이 암호화된 GUID를 반환합니다.

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. 다음 명령을 실행하여 파라미터 메타데이터를 확인합니다.

   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. 고객 관리형 AWS KMS key를 사용하지 않는 경우 파라미터 값을 변경하려면 다음 명령을 실행합니다.

   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
   ```

   -또는-

   고객 관리형 AWS KMS key를 사용하는 경우 파라미터 값을 변경하려면 다음 명령 중 하나를 실행합니다.

   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. 다음 명령을 실행하여 최신 파라미터 값을 확인합니다.

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. 다음 명령을 실행하여 파라미터 값 기록을 확인합니다.

   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**
>
> 암호화된 값을 사용하여 파라미터를 수동으로 만들 수 있습니다. 이 경우 값이 이미 암호화되어 있기 때문에 SecureString 파라미터 유형을 선택할 필요가 없습니다. SecureString을 선택하면 파라미터가 이중으로 암호화됩니다.

기본적으로 모든 SecureString 값은 암호 텍스트로 표시됩니다. SecureString 값을 복호화하려면 사용자에게 AWS KMS Decrypt API 작업을 호출할 수 있는 권한이 있어야 합니다. AWS KMS 액세스 제어 구성에 대한 자세한 내용은 AWS Key Management Service Developer Guide의 Authentication and Access Control for AWS KMS를 참조하세요.

> **Important**
>
> 파라미터를 암호화하는 데 사용되는 KMS 키에 대해 KMS 키 별칭을 변경하는 경우, 파라미터가 AWS KMS를 참조하는 데 사용하는 키 별칭도 업데이트해야 합니다. 이는 KMS 키 별칭에만 적용되며, 전체 키를 삭제하지 않는 한 별칭이 첨부되는 키 ID는 동일하게 유지됩니다.

여러 줄 파라미터 생성(AWS CLI)

AWS CLI를 사용하여 줄 바꿈이 있는 파라미터를 생성할 수 있습니다. 줄 바꿈을 사용하여 더 나은 가독성을 위해 긴 파라미터 값으로 텍스트를 나누거나 웹 페이지에 대한 여러 단락 파라미터 내용을 업

데이트합니다. 다음 예와 같이 JSON 파일에 내용을 포함하고 \n과 같은 줄 바꿈 문자를 사용하여 `--cli-input-json` 옵션을 사용할 수 있습니다.

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.

2. 다음 명령을 실행하여 여러 줄 파라미터를 생성합니다.

   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
   ```

   다음 예제에서는 `MultiLineParameter.json` 파일의 콘텐츠를 표시합니다.

   ```
   {
       "Value": "<para>Paragraph One</para>\n<para>Paragraph Two</para>
   \n<para>Paragraph Three</para>"
   }
   ```

저장된 파라미터 값은 다음과 같이 저장됩니다.

```
<para>Paragraph One</para>
<para>Paragraph Two</para>
<para>Paragraph Three</para>
```

Systems Manager 파라미터 생성(Tools for Windows PowerShell)

AWS Tools for Windows PowerShell를 사용하여 `String`, `StringList` 및 `SecureString` 파라미터 유형을 생성할 수 있습니다. 파라미터를 삭제한 후 최소 30초 동안 기다리고 같은 이름의 파라미터를 생성합니다.

파라미터는 다른 파라미터의 값에 참조되거나 중첩될 수 없습니다. 파라미터 값에 `{{}}` 또는 `{{ssm:parameter-name}}`을 포함할 수 없습니다.

> **Note**
>
> 파라미터는 생성된 해당 파라미터가 생성된 AWS 리전에서만 사용할 수 있습니다.

주제

- String 파라미터 생성(Tools for Windows PowerShell)
- StringList 파라미터 생성(Tools for Windows PowerShell)
- SecureString 파라미터 생성(Tools for Windows PowerShell)

`String` 파라미터 생성(Tools for Windows PowerShell)

1. 아직 설치하지 않은 경우 AWS Tools for PowerShell(Tools for Windows PowerShell)을 설치하고 구성합니다.

   자세한 내용은 AWS Tools for PowerShell 설치를 참조하세요.

2. 다음 명령을 실행하여 일반 텍스트 값을 포함하는 파라미터를 생성합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   ```
   Write-SSMParameter `
       -Name "parameter-name" `
       -Value "parameter-value" `
       -Type "String"
   ```

   -또는-

   다음 명령을 실행하여 파라미터 값으로 Amazon Machine Image(AMI) ID를 포함하는 파라미터를 생성합니다.

> **Note**
>
> 태그가 있는 파라미터를 생성하려면 변수로 service.model.tag를 생성합니다. 의 예는 다음과 같습니다.
>
> ```
> $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
```

AMI ID를 포함하는 파라미터를 생성하는 경우에만 `-DataType` 옵션을 지정해야 합니다. 다른 모든 파라미터의 경우 기본 데이터 유형은 `text`입니다. 자세한 내용은 Amazon Machine Image ID에 대한 기본 파라미터 지원 단원을 참조하십시오.

다음은 파라미터 계층 구조를 사용하는 예입니다.

```
Write-SSMParameter `
    -Name "/IAD/Web/SQL/IPaddress" `
    -Value "99.99.99.999" `
    -Type "String" `
    -Tags $tag
```

3. 다음 명령을 실행하여 파라미터의 세부 정보를 확인합니다.

```
(Get-SSMParameterValue -Name "the-parameter-name-you-specified").Parameters
```

`StringList` 파라미터 생성(Tools for Windows PowerShell)

1. 아직 설치하지 않은 경우 AWS Tools for PowerShell(Tools for Windows PowerShell)을 설치하고 구성합니다.

자세한 내용은 AWS Tools for PowerShell 설치를 참조하세요.

2. 다음 명령을 실행하여 StringList 파라미터를 생성합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

> **Note**
>
> 태그가 있는 파라미터를 생성하려면 변수로 service.model.tag를 생성합니다. 의 예는 다음과 같습니다.
>
> ```
> $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
```

성공할 경우 명령은 파라미터의 버전 번호를 반환합니다.

의 예는 다음과 같습니다.

```
Write-SSMParameter `
    -Name "stringlist-parameter" `
    -Value "Milana,Mariana,Mark,Miguel" `
    -Type "StringList" `
    -Tags $tag
```

> **Note**
>
> StringList의 항목들은 쉼표(,)로 구분해야 합니다. 목록에서 항목을 이스케이프하기 위해 다른 문장 부호나 특수 기호를 사용할 수 없습니다. 쉼표가 필요한 파라미터 값이 있는 경우 String 유형을 사용합니다.

3. 다음 명령을 실행하여 파라미터의 세부 정보를 확인합니다.

```
(Get-SSMParameterValue -Name "the-parameter-name-you-specified").Parameters
```

SecureString 파라미터 생성(Tools for Windows PowerShell)

`SecureString` 파라미터를 생성하기 전에, 이 파라미터 유형에 대한 요건을 읽어 보십시오. 자세한 내용은 SecureString 파라미터 생성(AWS CLI) 단원을 참조하십시오.

> **Important**
>
> `SecureString` 파라미터의 값만 암호화됩니다. 파라미터 이름, 설명 및 기타 속성은 암호화되지 않습니다.

> **Important**
>
> Parameter Store는 대칭 암호화 KMS 키만 지원합니다. 비대칭 암호화 KMS 키를 사용하여 파라미터를 암호화할 수 없습니다. KMS 키가 대칭인지 비대칭인지 확인하는 것과 관련된 도움말은 AWS Key Management Service Developer Guide의 Identifying symmetric and asymmetric KMS keys를 참조하세요.

1. 아직 설치하지 않은 경우 AWS Tools for PowerShell(Tools for Windows PowerShell)을 설치하고 구성합니다.

   자세한 내용은 AWS Tools for PowerShell 설치를 참조하세요.
2. 다음 명령을 실행하여 파라미터를 생성합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   > **Note**
   >
   > 태그가 있는 파라미터를 생성하려면 우선 변수로 service.model.tag를 생성합니다. 의 예는 다음과 같습니다.
   >
   > ```
   > $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
```

성공할 경우 명령은 파라미터의 버전 번호를 반환합니다.

> **Note**
>
> 계정에 할당된 AWS 관리형 키을(를) 사용하려면 명령에서 `-KeyId` 파라미터를 제거합니다.

다음은 암호 파라미터 및 AWS 관리형 키에 난독화된 이름(3l3vat3131)을 사용하는 예입니다.

```
Write-SSMParameter `
    -Name "/Finance/Payroll/3l3vat3131" `
    -Value "P@sSwW)rd" `
    -Type "SecureString"`
    -Tags $tag
```

3. 다음 명령을 실행하여 파라미터의 세부 정보를 확인합니다.

```
(Get-SSMParameterValue -Name "the-parameter-name-you-specified" -WithDecryption
 $true).Parameters
```

기본적으로 모든 `SecureString` 값은 암호 텍스트로 표시됩니다. `SecureString` 값을 복호화하려면 사용자에게 AWS KMS Decrypt API 작업을 호출할 수 있는 권한이 있어야 합니다. AWS KMS 액세스 제어 구성에 대한 자세한 내용은 AWS Key Management Service Developer Guide의 Authentication and Access Control for AWS KMS를 참조하세요.

> **Important**
>
> 파라미터를 암호화하는 데 사용되는 KMS 키에 대해 KMS 키 별칭을 변경하는 경우, 파라미터가 AWS KMS를 참조하는 데 사용하는 키 별칭도 업데이트해야 합니다. 이는 KMS 키 별칭에만 적용되며, 전체 키를 삭제하지 않는 한 별칭이 첨부되는 키 ID는 동일하게 유지됩니다.

Systems Manager 파라미터 검색

계정에 많은 수의 파라미터가 있는 경우 한 번에 단일 또는 여러 파라미터에 대한 정보를 찾기 어려울 수 있습니다. 이 경우 필터 도구를 사용하여 지정한 검색 조건에 따라 정보가 필요한 필터를 검색할 수 있습니다. AWS Systems Manager 콘솔, AWS Command Line Interface(AWS CLI), AWS Tools for PowerShell 또는 DescribeParameters API를 사용하여 파라미터를 검색할 수 있습니다.

주제

- 파라미터 검색(콘솔)
- 파라미터 검색(AWS CLI)

파라미터 검색(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Parameter Store를 선택합니다.
3. 검색 상자를 선택하고 검색 방법을 선택합니다. 예: `Type`또는 `Name`.
4. 선택한 검색 유형에 대한 정보를 제공합니다. 예:
   - `Type`을 기준으로 검색하는 경우 `String`, `StringList` 또는 `SecureString` 중에서 선택합니다.
   - `Name`을 기준으로 검색하는 경우 `contains`, `equals` 또는 `begins-with`를 선택한 다음, 파라미터 이름의 전체 또는 일부를 입력합니다.

     > **Note**
     >
     > 콘솔에서 `Name`에 대한 기본 검색 유형은 `contains`입니다.
5. Enter을 누릅니다.

파라미터의 목록이 검색 결과로 업데이트됩니다.

파라미터 검색(AWS CLI)

`describe-parameters` 명령을 사용하여 AWS CLI에서 하나 이상의 파라미터에 대한 정보를 봅니다.

다음 예에서는 AWS 계정의 파라미터에 대한 정보를 보는 데 사용할 수 있는 다양한 옵션을 보여줍니다. 이러한 옵션에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서의 describe-parameters를 참조하세요.

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.
2. 다음 명령의 샘플 값을 계정에 생성된 파라미터를 반영하는 값으로 바꿉니다.

   Linux & macOS

   ```
   aws ssm describe-parameters \
       --parameter-filters "Key=Name,Values=MyParameterName"
   ```

   Windows

   ```
   aws ssm describe-parameters ^
       --parameter-filters "Key=Name,Values=MyParameterName"
   ```

   > **Note**
   >
   > `describe-parameters`에서 `Name`의 기본 검색 유형은 `Equals`입니다. 파라미터 필터에서 `"Key=Name,Values=MyParameterName"`를 지정하는 것은 `"Key=Name,Option=Equals,Values=MyParameterName"`를 지정하는 것과 동일합니다.

   ```
   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**
>
> 마지막 예에서 *key-id*는 계정에서 생성된 SecureString 파라미터를 암호화하는 데 사용되는 AWS Key Management Service(AWS KMS) 키의 ID를 나타냅니다. 또는 **alias/aws/ssm**를 입력하여 계정의 기본 AWS KMS 키를 사용할 수 있습니다. 자세한 내용은 SecureString 파라미터 생성(AWS CLI) 단원을 참조하십시오.

이 작업이 성공하면 다음과 비슷한 출력이 반환됩니다.

```
{
    "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": []
        }
    ]
}
```

파라미터 정책 할당

파라미터 정책은 만료 날짜 또는 유지 시간(TTL)과 같이 파라미터에 대한 특정 기준을 지정할 수 있도록 허용으로써 증가하는 파라미터 집합을 관리하는 데 도움이 됩니다. 파라미터 정책은 특히 AWS Systems Manager의 기능인 Parameter Store에 저장된 암호 및 구성 데이터를 강제로 업데이트 또는 삭제하도록 하는 데 유용합니다. Parameter Store는 `Expiration`, `ExpirationNotification` 및 `NoChangeNotification` 유형의 정책을 제공합니다.

> **Note**
>
> 암호 교체 수명 주기를 구현하려면 AWS Secrets Manager를 사용합니다. Secrets Manager를 사용하면 수명 주기 동안 데이터베이스 자격 증명, API 키 및 기타 보안 암호를 손쉽게 교체, 관리 및 검색할 수 있습니다. 자세한 내용은 AWS Secrets Manager 사용 설명서의 AWS Secrets Manager(이)란 무엇입니까? 섹션을 참조하세요.

Parameter Store은 비동기 정기 검사를 사용하여 파라미터 정책을 적용합니다. 정책을 생성한 후에는 정책을 실행하기 위해 추가 작업을 수행할 필요가 없습니다. Parameter Store는 지정한 기준에 따라 정책에 의해 정의된 작업을 독립적으로 수행합니다.

> **Note**
>
> 파라미터 정책은 고급 파라미터 티어를 사용하는 파라미터에 대해 사용할 수 있습니다. 자세한 내용은 파라미터 티어 관리 단원을 참조하십시오.

파라미터 정책은 다음 표에 표시된 대로 JSON 배열입니다. 새 고급 파라미터를 생성할 때 정책을 할당하거나 파라미터를 업데이트하여 정책을 적용할 수 있습니다. Parameter Store은 다음 유형의 파라미터 정책을 지원합니다.

정책	Details	예제
만료	이 정책은 파라미터를 삭제합니다. `ISO_INSTANT` 형식 또는 `ISO_OFFSET_DATE_TIME` 형식을 사용하여 특정 날짜와 시간을 지정할 수 있습니다. 파라미터를 삭제할 시점을 변경하려면 정책을 업데이트합니다. 파라미터를 업데이트해도 연결된 정책의 만료 날짜 또는 시간에는 영향을 주지 않습니다. 만료 날짜와 시간에 도달하면 Parameter Store에서 파라미터를 삭제합니다. **Note** 이 예에서는 `ISO_INSTANT` 형식을 사용합니다. `ISO_OFFSET_DATE_TIME` 형식을 사용하여 날짜와 시간을 지정할 수도 있습니다. 예: `2019-11-0`	`{ "Type": "Expiration", "Version": "1.0", "Attributes": { "Timestamp": "2018-12-02T21:34:33.000Z" } }`

정책	Details	예제
	1T22:13:4 8.87+10:30:00	
ExpirationNotification	이 정책은 만료를 알리는 Amazon EventBridge(EventBridge) 이벤트를 시작합니다. 이 정책을 사용하면 만료 시간에 도달하기 전에 일 또는 시간 단위로 알림을 수신할 수 있습니다.	{ "Type": "ExpirationNotification", "Version": "1.0", "Attributes": { "Before": "15", "Unit": "Days" } }
NoChangeNotification	이 정책은 지정된 기간 동안 파라미터가 수정되지 않은 경우 EventBridge에서 이벤트를 시작합니다. 이 정책 유형은 일정 기간 내에 암호를 변경해야 하는 경우 등에 유용합니다. 이 정책은 파라미터의 `LastModifiedTime` 속성을 읽음으로써 알림을 보낼 시기를 결정합니다. 파라미터를 변경하거나 편집하면 시스템이 `LastModifiedTime` 의 새 값에 따라 알림 기간을 재설정합니다.	{ "Type": "NoChangeNotification", "Version": "1.0", "Attributes": { "After": "20", "Unit": "Days" } }

파라미터에 여러 정책을 할당할 수 있습니다. 예를 들어 시스템에서 파라미터가 곧 삭제됨을 알리는 EventBridge 이벤트를 시작하도록 `Expiration` 및 `ExpirationNotification` 정책을 할당할 수 있습니다. 파라미터에 최대 10개의 정책을 할당할 수 있습니다.

다음 예에서는 `ProdDB3`이라는 새 `SecureString` 파라미터에 4개의 정책을 할당하는 PutParameter API 요청을 위한 요청 신택스를 보여줍니다.

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

기존 파라미터에 정책 추가

이 절에는 AWS Systems Manager 콘솔, AWS Command Line Interface(AWS CLI) 및 AWS Tools for Windows PowerShell을 사용하여 기존 파라미터에 정책을 추가하는 방법에 대한 정보가 포함되어 있습니다. 정책이 포함된 새 파라미터를 만드는 방법에 대한 자세한 내용은 Systems Manager 파라미터 생성 섹션을 참조하세요.

주제

기존 파라미터에 정책 추가(콘솔)

Systems Manager 콘솔을 사용하여 기존 파라미터에 정책을 추가하려면 다음 절차를 사용합니다.

기존 파라미터에 정책을 추가하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Parameter Store를 선택합니다.
3. 정책을 포함하도록 업데이트할 파라미터 옆의 옵션을 선택한 다음 편집을 선택합니다.
4. 고급을 선택합니다.
5. (선택 사항) 파라미터 정책 섹션에서 활성을 선택합니다. 이 파라미터에 대한 만료 날짜와 하나 이상의 알림 정책을 지정할 수 있습니다.
6. Save changes(변경 사항 저장)를 선택합니다.

> **Important**
>
> - Parameter Store은 정책을 새 정책으로 덮어쓰거나 정책을 제거할 때까지 파라미터에 대한 정책을 보존합니다.
> - 기존 파라미터에서 모든 정책을 제거하려면 다음과 같이 파라미터를 편집하고 대괄호 및 중괄호를 사용하여 빈 정책을 적용하십시오. `[{}]`
> - 이미 정책이 있는 파라미터에 새 정책을 추가하면 Systems Manager가 파라미터에 연결된 정책을 덮어씁니다. 기존 정책은 삭제됩니다. 이미 하나 이상의 정책이 있는 파라미터에 새

정책을 추가하려면 원래 정책을 복사하여 붙여넣고 새 정책을 입력한 다음 변경 사항을 저장합니다.

기존 파라미터에 정책 추가(AWS CLI)

AWS CLI를 사용하여 기존 파라미터에 정책을 추가하려면 다음 절차를 사용하십시오.

기존 파라미터에 정책을 추가하려면

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.
2. 다음 명령을 실행하여 기존 파라미터에 정책을 추가합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   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}]"
   ```

   다음은 15일 후에 파라미터를 삭제하는 만료 정책을 포함하는 예제입니다. 이 예제에는 파라미터가 삭제되기 5일 전에 EventBridge 이벤트를 생성하는 알림 정책도 포함되어 있습니다. 마지막으로 60일 후에 이 파라미터를 변경하지 않으면 NoChangeNotification 정책이 포함됩니다. 다음은 암호 및 AWS Key Management Service AWS KMS key에 난독화된 이름(3l3vat3131)

을 사용하는 예입니다. AWS KMS keys에 대한 자세한 내용은 AWS Key Management Service Developer Guide의 AWS Key Management Service Concepts 섹션을 참조하세요.

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. 다음 명령을 실행하여 파라미터의 세부 정보를 확인합니다. *parameter name*을 사용자의 정보로 바꿉니다.

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 Store은 정책을 새 정책으로 덮어쓰거나 정책을 제거할 때까지 파라미터에 대한 정책을 보유합니다.
- 기존 파라미터에서 모든 정책을 제거하려면 파라미터를 편집하고 대괄호 및 중괄호로 빈 정책을 적용하십시오. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다. 예:

  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 "[{}]"
  ```

- 이미 정책이 있는 파라미터에 새 정책을 추가하면 Systems Manager가 파라미터에 연결된 정책을 덮어씁니다. 기존 정책은 삭제됩니다. 이미 하나 이상의 정책이 있는 파라미터에 새 정책을 추가하려면 원래 정책을 복사하여 붙여넣고 새 정책을 입력한 다음 변경 사항을 저장합니다.

기존 파라미터에 정책 추가(Tools for Windows PowerShell)

Tools for Windows PowerShell을 사용하여 기존 파라미터에 정책을 추가하려면 다음 절차를 사용합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

기존 파라미터에 정책을 추가하려면

1. Tools for Windows PowerShell을 열고 다음 명령을 실행하여 자격 증명을 지정합니다. Amazon Elastic Compute Cloud(Amazon EC2)에 관리자 권한이 있거나 AWS Identity and Access Management(IAM)에서 적절한 권한을 부여 받아야 합니다.

```
Set-AWSCredentials `
    -AccessKey access-key-name `
    -SecretKey secret-key-name
```

2. 다음 명령을 실행하여 PowerShell 세션의 리전을 설정합니다. 이 예에서는 미국 동부(오하이오) 리전(us-east-2)을 사용합니다.

```
Set-DefaultAWSRegion `
    -Region us-east-2
```

3. 다음 명령을 실행하여 기존 파라미터에 정책을 추가합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
Write-SSMParameter `
    -Name "parameter name" `
    -Value "parameter value" `
    -Type "parameter type" `
    -Policies "[{policies-enclosed-in-brackets-and-curly-braces}]" `
    -Overwrite
```

다음은 2020년 5월 13일 자정(GMT)에 파라미터를 삭제하는 만료 정책을 포함하는 예제입니다. 이 예제에는 파라미터가 삭제되기 5일 전에 EventBridge 이벤트를 생성하는 알림 정책도 포함되어 있습니다. 마지막으로 60일 후에 이 파라미터를 변경하지 않으면 NoChangeNotification 정책이 포함됩니다. 다음은 암호 및 AWS 관리형 키에 난독화된 이름(3l3vat3131)을 사용하는 예입니다.

```
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. 다음 명령을 실행하여 파라미터의 세부 정보를 확인합니다. *parameter name*을 사용자의 정보로 바꿉니다.

```
(Get-SSMParameterValue -Name "parameter name").Parameters
```

Important

- Parameter Store은 정책을 새 정책으로 덮어쓰거나 정책을 제거할 때까지 파라미터에 대한 정책을 보존합니다.
- 기존 파라미터에서 모든 정책을 제거하려면 파라미터를 편집하고 대괄호 및 중괄호로 빈 정책을 적용하십시오. 예:

  ```
  Write-SSMParameter `
      -Name "parameter name" `
      -Value "parameter value" `
      -Type "parameter type" `
      -Policies "[{}]"
  ```

- 이미 정책이 있는 파라미터에 새 정책을 추가하면 Systems Manager가 파라미터에 연결된 정책을 덮어씁니다. 기존 정책은 삭제됩니다. 이미 하나 이상의 정책이 있는 파라미터에 새 정책을 추가하려면 원래 정책을 복사하여 붙여넣고 새 정책을 입력한 다음 변경 사항을 저장합니다.

파라미터 계층 구조 작업

수십 또는 수백 개 파라미터를 하나의 집합 목록으로 관리하면 많은 시간이 들고 오류에 취약합니다. 작업에 적합한 파라미터를 식별하기가 어려울 수도 있습니다. 즉, 실수로 잘못된 파라미터를 사용하거나 동일한 구성 데이터를 사용하는 여러 파라미터를 생성할 수 있다는 뜻입니다.

파라미터 계층 구조를 사용하면 파라미터를 조직하고 관리하는 데 도움이 됩니다. 계층 구조는 슬래시(/)를 사용하여 정의하는 경로를 포함한 파라미터 이름입니다.

주제

- 파라미터 계층 예제
- 계층 구조에서 파라미터 쿼리
- Parameter Store API 작업에 대한 액세스 제한
- 계층 구조를 이용한 파라미터 관리(AWS CLI)

파라미터 계층 예제

다음 예제에서는 이름에서 세 가지 계층 구조 수준을 사용하여 다음을 식별합니다.

```
/Environment/Type of computer/Application/Data
```

```
/Dev/DBServer/MySQL/db-string13
```

최대 15레벨의 계층 구조를 생성할 수 있습니다. 다음 예와 같이 현재 환경의 기존 계층 구조를 반영하는 계층 구조를 생성할 것을 권장합니다.

- 지속적인 통합 및 지속적인 전송 환경(CI/CD 워크플로)

  ```
  /Dev/DBServer/MySQL/db-string
  ```

  ```
  /Staging/DBServer/MySQL/db-string
  ```

  ```
  /Prod/DBServer/MySQL/db-string
  ```

- 컨테이너를 사용하는 애플리케이션

  ```
  /MyApp/.NET/Libraries/my-password
  ```

- 비즈니스 조직

  ```
  /Finance/Accountants/UserList
  ```

  ```
  /Finance/Analysts/UserList
  ```

  ```
  /HR/Employees/EU/UserList
  ```

파라미터 계층 구조는 파라미터를 생성하는 방식을 표준화하고, 시간에 따른 파라미터 관리를 용이하게 해줍니다. 파라미터 계층 구조는 구성 작업에 적합한 파라미터를 식별하는 데에도 도움이 될 수 있습니다. 이렇게 하면 같은 구성 데이터로 여러 파라미터가 생성되지 않도록 할 수 있습니다.

개발 및 스테이징 환경에 암호를 사용하는 다음 예에서 보듯이, 여러 환경에 걸쳐 파라미터를 공유할 수 있는 계층 구조를 생성할 수 있습니다.

```
/DevTest/MyApp/database/my-password
```

그러면 다음 예와 같이 프로덕션 환경에 고유한 암호를 생성할 수 있습니다.

```
/prod/MyApp/database/my-password
```

파라미터 계층 구조를 지정할 필요가 없습니다. 1레벨에 파라미터를 생성할 수 있습니다. 이를 루트 파라미터라고 합니다. 이전 버전과의 호환성을 위해 계층 구조가 배포되기 전 Parameter Store에서 생성된 모든 파라미터가 루트 파라미터입니다. 시스템은 다음 두 파라미터를 모두 루트 파라미터로 취급합니다.

`/parameter-name`

`parameter-name`

계층 구조에서 파라미터 쿼리

계층 구조 사용의 또 한 가지 장점은 GetParametersByPath API 작업을 이용해 계층 구조 내에서 모든 파라미터에 대한 쿼리가 가능하다는 점입니다. 예를 들어 AWS Command Line Interface(AWS CLI)에서 다음 명령을 실행하면 시스템이 IIS 레벨의 모든 파라미터를 반환합니다.

```
aws ssm get-parameters-by-path --path /Dev/Web/IIS
```

계층 구조에서 복호화된 `SecureString` 파라미터를 보려면 다음 예제에서 보듯이 경로와 `--with-decryption` 파라미터를 지정합니다.

```
aws ssm get-parameters-by-path --path /Prod/ERP/SAP --with-decryption
```

Parameter Store API 작업에 대한 액세스 제한

AWS Identity and Access Management(IAM) 정책을 사용하여 Parameter Store API 작업 및 콘텐츠에 대한 사용자 액세스를 제공하거나 제한할 수 있습니다.

다음 샘플 정책에서는 미국 동부(오하이오) 리전(us-east-2)의 AWS 계정 123456789012에 있는 모든 파라미터에 대해 `PutParameter` API 작업을 실행할 수 있는 액세스 권한을 사용자에게 먼저 부여합니다. 그러나 `PutParameter` 작업에 대해 `Overwrite` 옵션이 명시적으로 거부되기 때문에 사용자는 기존 파라미터의 값을 변경할 수 없습니다. 즉, 이 정책이 할당된 사용자는 파라미터를 만들 수 있지만 기존 파라미터를 변경할 수는 없습니다.

```
{
    "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/*"
        }
    ]
}
```

계층 구조를 이용한 파라미터 관리(AWS CLI)

이 절차는 AWS CLI를 이용하여 파라미터와 파라미터 계층 구조를 다루는 방법을 보여줍니다.

계층 구조를 사용하여 파라미터를 관리하려면

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.
2. 다음 명령을 실행하여 `allowedPattern` 파라미터 및 `String` 데이터 형식을 사용하는 파라미터를 생성합니다. 이 예에서 허용된 패턴은 파라미터 값이 1 - 4자 길이여야 한다는 뜻입니다.

   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
```

이 명령은 파라미터의 버전 번호를 반환합니다.

3. 다음 명령을 실행하여 방금 새로운 값으로 생성한 파라미터의 덮어쓰기를 시도합니다.

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
```

새 값이 이전 단계에 지정한 허용 패턴의 요구 사항에 부합하지 않기 때문에 시스템이 다음 오류를 반환합니다.

```
An error occurred (ParameterPatternMismatchException) when calling the PutParameter
 operation: Parameter value, cannot be validated against allowedPattern: \d{1,4}
```

4. 다음 명령을 실행하여 AWS 관리형 키을 사용하는 SecureString 파라미터를 생성합니다. 이 예에서 허용된 패턴은 사용자가 어떤 문자든 지정할 수 있으나 값이 8 - 20자여야 한다는 뜻입니다.

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. 다음 명령을 실행하여 이전 단계의 계층 구조를 사용하는 더 많은 파라미터를 생성합니다.

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. 다음 명령을 실행하여 두 가지 파라미터의 값을 가져옵니다.

   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. 다음 명령을 실행하여 단일 레벨의 모든 파라미터에 대해 쿼리합니다.

   Linux & macOS

   ```
   aws ssm get-parameters-by-path \
       --path "/MyService/Test"
   ```

   Windows

   ```
   aws ssm get-parameters-by-path ^
       --path "/MyService/Test"
   ```

8. 다음 명령을 실행하여 2개의 파라미터를 삭제합니다.

   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"
   ```

파라미터 레이블 작업

파라미터 레이블이란 다양한 버전의 파라미터를 관리하는 데 도움이 되는 사용자 정의 별칭입니다. 파라미터를 수정하면 AWS Systems Manager에서 버전 번호를 하나씩 늘려서 새 버전을 저장합니다. 레이블이 있으면 파라미터 버전이 여러 개일 때 버전의 용도를 기억하기 쉽습니다.

예를 들어, `/MyApp/DB/ConnectionString`이라는 파라미터가 있다고 가정해 보겠습니다. 이 파라미터의 값은 테스트 환경의 로컬 데이터베이스에 있는 MySQL 서버에 대한 연결 문자열입니다. 애플리케이션 업데이트가 끝나면 이 파라미터가 프로덕션 데이터베이스에 대한 연결 문자열을 사용하도록 해야 합니다. `/MyApp/DB/ConnectionString`의 값을 변경합니다. Systems Manager에서 새 연결 문자열로 버전 2를 자동으로 생성합니다. 각 버전의 용도를 기억하기 쉽게 각각의 파라미터에 레이블을 부착합니다. 버전 1에는 테스트 레이블을, 버전 2에는 프로덕션 레이블을 부착합니다.

파라미터 버전 간에 레이블을 이동할 수 있습니다. 예를 들어 새 프로덕션 데이터베이스의 연결 문자열로 `/MyApp/DB/ConnectionString` 파라미터 버전 3을 생성한 다음, 파라미터 버전 2의 [프로덕션(Production)] 레이블을 파라미터 버전 3으로 옮길 수 있습니다.

파라미터 레이블은 파라미터 태그에 비해 간단한 대안입니다. 태그의 경우, 다양한 AWS 리소스에 반드시 적용해야 하는 해당 조직의 엄격한 태그 기준이 있을 수도 있습니다. 그러나 레이블은 특정 파라미터 버전에 대한 텍스트 연결에 불과합니다.

태그와 마찬가지로, 레이블로도 파라미터를 쿼리할 수 있습니다. 이 섹션 후반부의 설명에 따라 GetParametersByPath API 작업을 사용하여 파라미터 집합을 쿼리하면 모두 동일한 레이블을 사용하는 특정한 파라미터 버전 목록이 표시됩니다.

> **Note**
>
> 존재하지 않는 파라미터의 버전을 지정하는 명령을 실행하면 명령이 실패합니다. 파라미터의 최신 값이나 기본값으로 폴백되지 않습니다.

레이블 요구 사항 및 제한

파라미터 레이블의 요구 및 제한 사항은 다음과 같습니다.

- 파라미터 버전 하나에 레이블을 최대 10개 지정할 수 있습니다.
- 동일 파라미터의 서로 다른 버전에 동일한 레이블을 부착할 수 없습니다. 예를 들어 파라미터의 버전 1에 프로덕션 레이블이 있으면 버전 2에는 프로덕션을 부착할 수 없습니다.
- 파라미터 버전 간에 레이블을 이동할 수 있습니다.

- 파라미터를 생성할 때 레이블을 함께 생성할 수 없습니다. 레이블은 특정 파라미터 버전에 부착해야 합니다.
- 사용하지 않을 파라미터 레이블은 다른 파라미터 버전으로 옮기거나 삭제할 수 있습니다.
- 레이블은 최대 100자까지 가능합니다.
- 레이블은 문자(대소문자 구분), 숫자, 마침표(.), 하이픈(-) 또는 밑줄(_)을 포함할 수 있습니다.
- 레이블은 숫자, "aws", "ssm"으로 시작할 수 없습니다(대/소문자 구별하지 않음). 이러한 요구 사항에 맞지 않는 레이블은 파라미터 버전에 부착되지 않으며 `InvalidLabels` 목록에 표시됩니다.

주제

- 파라미터 레이블 작업(콘솔)
- 파라미터 레이블 작업(AWS CLI)

파라미터 레이블 작업(콘솔)

이 섹션에서는 Systems Manager 콘솔을 사용하여 다음 태스크를 수행하는 방법을 설명합니다.

- 파라미터 레이블 생성(콘솔)
- 파라미터에 부착된 레이블 보기(콘솔)
- 파라미터 레이블 이동(콘솔)
- 파라미터 레이블 삭제(콘솔)

파라미터 레이블 생성(콘솔)

다음 절차에서는 Systems Manager 콘솔을 사용하여 기존 파라미터의 특정 버전에 레이블을 부착하는 방법을 설명합니다. 파라미터를 생성할 때 레이블을 부착할 수는 없습니다.

파라미터 버전에 레이블을 부착하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Parameter Store를 선택합니다.
3. 파라미터 이름을 선택하여 해당 파라미터의 세부 정보 페이지를 엽니다.
4. 기록 탭을 선택합니다.
5. 레이블을 부착할 파라미터 버전을 선택합니다.
6. [레이블 관리(Manage labels)]를 선택합니다.

7. [새 레이블 추가(Add new label)]를 선택합니다.
8. 텍스트 상자에 레이블 이름을 입력합니다. 레이블을 더 추가하려면 [새 레이블 추가(Add new label)]를 선택합니다. 레이블을 최대 열 개까지 부착할 수 있습니다.
9. 작업을 마쳤으면 변경 내용 저장을 선택합니다.

파라미터에 부착된 레이블 보기(콘솔)

파라미터 버전 하나에 레이블을 최대 열 개까지 지정할 수 있습니다. 다음 절차에서는 Systems Manager 콘솔을 사용하여 파라미터 버전에 부착된 모든 레이블을 보는 방법을 설명합니다.

파라미터 버전에 부착된 레이블을 보려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Parameter Store를 선택합니다.
3. 파라미터 이름을 선택하여 해당 파라미터의 세부 정보 페이지를 엽니다.
4. 기록 탭을 선택합니다.
5. 부착된 모든 레이블을 볼 파라미터 버전을 찾습니다. 레이블 열에 해당 파라미터 버전에 부착된 레이블이 모두 표시됩니다.

파라미터 레이블 이동(콘솔)

다음 절차에서는 Systems Manager 콘솔을 사용하여 같은 파라미터의 다른 버전으로 파라미터 레이블을 옮기는 방법을 설명합니다.

레이블을 다른 파라미터 버전으로 옮기려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Parameter Store를 선택합니다.
3. 파라미터 이름을 선택하여 해당 파라미터의 세부 정보 페이지를 엽니다.
4. 기록 탭을 선택합니다.
5. 레이블을 이동할 파라미터 버전을 선택합니다.
6. [레이블 관리(Manage labels)]를 선택합니다.
7. [새 레이블 추가(Add new label)]를 선택합니다.
8. 텍스트 상자에 레이블 이름을 입력합니다.

9. 작업을 마쳤으면 변경 내용 저장을 선택합니다.

파라미터 레이블 삭제(콘솔)

다음 절차에서는 Systems Manager 콘솔을 사용하여 하나 이상의 파라미터 레이블을 삭제하는 방법을 설명합니다.

파라미터에서 레이블을 삭제하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Parameter Store를 선택합니다.
3. 파라미터 이름을 선택하여 해당 파라미터의 세부 정보 페이지를 엽니다.
4. 기록 탭을 선택합니다.
5. 레이블을 삭제할 파라미터 버전을 선택합니다.
6. [레이블 관리(Manage labels)]를 선택합니다.
7. 삭제하려는 각 레이블 옆에 있는 [제거(Remove)]를 선택합니다.
8. 작업을 마쳤으면 변경 내용 저장을 선택합니다.
9. 변경 사항이 올바른지 확인하고 텍스트 상자에 `Confirm`를 입력하고 [확인(Confirm)]을 선택합니다.

파라미터 레이블 작업(AWS CLI)

이 섹션에서는 AWS Command Line Interface(AWS CLI)를 사용하여 다음 태스크를 수행하는 방법을 설명합니다.

- 새 파라미터 레이블 생성(AWS CLI)
- 파라미터 레이블 보기(AWS CLI)
- 레이블이 할당된 파라미터 목록 보기(AWS CLI)
- 파라미터 레이블 이동(AWS CLI)
- 파라미터 레이블 삭제(AWS CLI)

새 파라미터 레이블 생성(AWS CLI)

다음 절차에서는 AWS CLI를 사용하여 기존 파라미터의 특정 버전에 레이블을 부착하는 방법을 설명합니다. 파라미터를 생성할 때 레이블을 부착할 수는 없습니다.

파라미터 레이블을 생성하려면

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.

2. 다음 명령을 실행하여 본인에게 레이블 부착 권한이 있는 파라미터 목록을 봅니다.

   > **Note**
   >
   > 파라미터는 생성된 해당 파라미터가 생성된 AWS 리전에서만 사용할 수 있습니다. 레이블을 부착할 파라미터가 보이지 않으면 리전을 확인하십시오.

   ```
   aws ssm describe-parameters
   ```

   레이블을 부착할 파라미터의 이름을 적어 둡니다.

3. 다음 명령을 실행하여 파라미터의 모든 버전을 봅니다.

   ```
   aws ssm get-parameter-history --name "parameter-name"
   ```

   레이블을 부착할 파라미터 버전을 적어 둡니다.

4. 다음 명령을 사용하여 버전 번호별로 파라미터의 정보를 검색합니다.

   ```
   aws ssm get-parameters --names "parameter-name:version-number"
   ```

   다음 예를 참고하세요

   ```
   aws ssm get-parameters --names "/Production/SQLConnectionString:3"
   ```

5. 다음 명령 중 하나를 실행하여 파라미터 버전에 레이블을 부착합니다. 레이블을 여러 개 부착하는 경우 레이블 이름을 공백으로 구분합니다.

   최신 파라미터 버전에 레이블 부착

   ```
   aws ssm label-parameter-version --name parameter-name  --labels label-name
   ```

   특정 파라미터 버전에 레이블 부착

```
aws ssm label-parameter-version --name parameter-name --parameter-version version-number --labels label-name
```

여기 몇 가지 예가 있습니다.

```
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**
>
> 출력 결과, 생성한 레이블이 InvalidLabels 목록에 표시되면 그 레이블은 이 주제 전반부에서 설명한 요구 사항에 맞지 않는 것입니다. 요구 사항을 검토한 후 다시 시도하십시오. InvalidLabels 목록이 비어 있으면 레이블이 해당 파라미터 버전에 적용된 것입니다.

6. 버전 번호나 레이블 이름을 사용하여 파라미터의 세부 정보를 볼 수 있습니다. 다음 명령을 실행하고, 이전 단계에서 만든 레이블을 지정하십시오.

```
aws ssm get-parameter --name parameter-name:label-name --with-decryption
```

명령은 다음과 같은 정보를 반환합니다.

```
{
    "Parameter": {
        "Version": version-number,
        "Type": "parameter-type",
        "Name": "parameter-name",
        "Value": "parameter-value",
        "Selector": ":label-name"
    }
}
```

> **Note**
>
> 이 출력에서 선택자는 Name 입력 필드에서 지정한 레이블 또는 버전 번호입니다.

파라미터 레이블 보기(AWS CLI)

GetParameterHistory API 작업을 사용하여 지정된 파라미터에 부착되어 있는 모든 레이블과 전체 기록을 볼 수 있습니다. 또는 GetParametersByPath API 작업으로 특정 레이블에 할당된 전체 파라미터 목록을 볼 수도 있습니다.

GetParameterHistory API 작업을 사용하여 파라미터의 레이블을 보려면

1. 다음 명령을 실행하여 본인에게 레이블 보기 권한이 있는 파라미터 목록을 봅니다.

   > **Note**
   >
   > 파라미터는 생성된 리전에서만 사용할 수 있습니다. 레이블을 옮길 파라미터가 보이지 않으면 리전을 확인하십시오.

   ```
   aws ssm describe-parameters
   ```

   레이블을 보려는 파라미터의 이름을 기록해 둡니다.
2. 다음 명령을 실행하여 파라미터의 모든 버전을 봅니다.

   ```
   aws ssm get-parameter-history --name parameter-name --with-decryption
   ```

   시스템은 다음과 같은 정보를 반환합니다.

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

레이블이 할당된 파라미터 목록 보기(AWS CLI)

GetParametersByPath API 작업으로 특정 레이블에 할당된 경로의 전체 파라미터 목록을 볼 수 있습니다.

다음 명령을 실행하여 특정 레이블에 할당된 경로의 파라미터 목록을 봅니다. *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
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
```

시스템은 다음과 같은 정보를 반환합니다. 이 예제의 사용자는 /Config 경로에서 검색했습니다.

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

파라미터 레이블 이동(AWS CLI)

다음 절차에서는 같은 파라미터의 다른 버전으로 파라미터 레이블을 옮기는 방법을 설명합니다.

파라미터 레이블을 옮기려면

1. 다음 명령을 실행하여 파라미터의 모든 버전을 봅니다. *parameter name*을 사용자의 정보로 바꿉니다.

   ```
   aws ssm get-parameter-history \
       --name "parameter name"
   ```

   레이블을 이동할 파라미터 버전을 기록해 둡니다.

2. 다음 명령을 실행하여 기존 레이블을 파라미터의 다른 버전에 할당합니다. *example resource placeholder*를 사용자의 정보로 바꿉니다.

   ```
   aws ssm label-parameter-version \
       --name parameter name \
       --parameter-version version number \
       --labels name-of-existing-label
   ```

> **Note**
>
> 기존 레이블을 최신 버전의 파라미터로 옮기려면 명령에서 `--parameter-version`을 빼십시오.

파라미터 레이블 삭제(AWS CLI)

다음 절차에서는 AWS CLI를 사용하여 파라미터 레이블을 삭제하는 방법을 설명합니다.

파라미터 레이블을 삭제하려면

1. 다음 명령을 실행하여 파라미터의 모든 버전을 봅니다. *parameter name*을 사용자의 정보로 바꿉니다.

```
aws ssm get-parameter-history \
    --name "parameter name"
```

시스템은 다음과 같은 정보를 반환합니다.

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

하나 이상의 레이블을 삭제할 파라미터 버전을 적어 둡니다.

2. 다음 명령을 실행하여 해당 파라미터에서 선택한 레이블을 삭제합니다. *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
aws ssm unlabel-parameter-version \
    --name parameter name \
    --parameter-version version \
    --labels label 1,label 2,label 3
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "InvalidLabels": ["invalid"],
    "DeletedLabels" : ["Prod"]
 }
```

파라미터 버전 작업

파라미터 값을 편집할 때마다 AWS Systems Manager의 기능인 Parameter Store는 파라미터의 새 버전을 생성하고 이전 버전을 유지합니다. 파라미터를 처음 생성하면 Parameter Store가 이 파라미터에 버전 1을 지정합니다. 파라미터 값을 변경할 때 Parameter Store는 자동으로 버전 번호를 1씩 늘립니다. 파라미터 기록에서 값을 포함하여 모든 버전의 세부 정보를 볼 수 있습니다.

API 명령 및 SSM 문서에 사용할 파라미터의 버전을 지정할 수도 있습니다(예: `ssm:MyParameter:3`). API 호출 및 SSM 문서에서 파라미터 이름과 특정 버전 번호를 지정할 수 있

습니다. 버전 번호를 지정하지 않을 경우 시스템이 자동으로 최신 버전을 사용합니다. 존재하지 않는 버전의 번호를 지정하면 파라미터의 최신 버전 또는 기본 버전으로 폴백되지 않고 오류가 반환됩니다.

또한 파라미터 버전을 사용하여 일정 기간 동안 파라미터가 몇 번 변경되었는지 확인할 수 있습니다. 파라미터 버전은 파라미터 값이 실수로 변경된 경우에도 보호 계층을 제공합니다.

최대 100개의 파라미터 버전을 생성하고 유지할 수 있습니다. 100개의 파라미터 버전을 생성한 후 새 버전을 생성할 때마다 새 버전을 위한 공간을 만들기 위해 가장 오래된 버전의 파라미터가 기록에서 제거됩니다.

기록에 이미 100개의 파라미터 버전이 있고 파라미터 레이블이 가장 오래된 버전의 파라미터에 할당된 경우는 예외입니다. 이 경우 해당 버전은 기록에서 제거되지 않고 새 파라미터 버전 생성 요청이 실패합니다. 이 안전 조치는 미션 크리티컬 라벨이 할당된 파라미터 버전이 삭제되는 것을 방지하기 위한 것입니다. 새 파라미터를 계속 생성하려면 먼저 파라미터의 가장 오래된 버전에서 작업에 사용할 새 파라미터로 레이블을 이동합니다. 파라미터 레이블 이동에 대한 자세한 내용은 파라미터 레이블 이동(콘솔) 및 파라미터 레이블 이동(AWS CLI) 섹션을 참조하세요.

다음 절차에서는 파라미터를 편집하고 새 버전이 생성되었는지 확인하는 방법을 보여줍니다. `get-parameter` 및 `get-parameters` 명령을 사용하여 파라미터 버전을 볼 수 있습니다. 이러한 명령 사용 예는 AWS Systems Manager API Reference의 GetParameter 및 GetParameters를 참조하세요.

주제

- 파라미터의 새 버전 생성(콘솔)
- 파라미터 버전 참조

파라미터의 새 버전 생성(콘솔)

Systems Manager 콘솔을 사용하여 새 버전의 파라미터를 생성하고 파라미터의 버전 기록을 볼 수 있습니다.

파라미터의 새 버전을 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Parameter Store를 선택합니다.
3. 앞서 생성한 파라미터 이름을 선택합니다. 새 파라미터 생성에 대한 자세한 내용은 Systems Manager 파라미터 생성 섹션을 참조하세요.
4. 편집을 선택합니다.
5. [값(Value)] 상자에 새 값을 입력한 다음 [변경 사항 저장(Save changes)]을 선택합니다.

6. 방금 업데이트한 파라미터의 이름을 선택합니다. 개요 탭에서 버전 번호가 1만큼 증가했는지 확인하고 새 값을 확인합니다.
7. 파라미터의 모든 버전 기록을 보려면 기록 탭을 선택합니다.

파라미터 버전 참조

ssm:*parameter-name*:*version-number* 형식을 사용하여 명령, API 호출 및 SSM 문서에서 특정 파라미터 버전을 참조할 수 있습니다.

다음 예에서 Amazon Elastic Compute Cloud(Amazon EC2) `run-instances command`는 파라미터 `golden-ami`의 버전 3을 사용합니다.

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**
>
> `resolve` 및 파라미터 값은 Amazon Machine Image(AMI)를 값으로 포함하는 파라미터 및 `--image-id` 옵션에서만 사용할 수 있습니다. 자세한 내용은 Amazon Machine Image ID에 대한 기본 파라미터 지원 단원을 참조하십시오.

다음은 SSM 문서에서 `MyRunCommandParameter`이라는 파라미터의 버전 2를 지정하는 예제입니다.

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}}"
                ]
            }
        }
    ]
}
```

공유 파라미터 작업

고급 파라미터를 공유하면 다중 계정 환경에서 구성 데이터 관리가 간소화됩니다. 파라미터를 중앙에서 저장 및 관리하고 이를 참조해야 하는 다른 AWS 계정와 공유할 수 있습니다.

Parameter Store가 AWS Resource Access Manager(AWS RAM)와 통합되어 고급 파라미터 공유가 가능합니다. AWS RAM은 리소스를 다른 AWS 계정과 공유하거나 AWS Organizations을 통해 공유할 수 있는 서비스입니다.

AWS RAM(을)를 사용하면 리소스 공유를 생성하여 내 소유의 리소스를 공유할 수 있습니다. 리소스 공유는 공유할 리소스, 부여할 권한, 공유 대상 소비자를 지정합니다. 소비자에는 다음이 포함될 수 있습니다.

- AWS 계정의 조직 내부 또는 외부의 특정 AWS Organizations.
- AWS Organizations에서 조직 내부의 조직 단위
- AWS Organizations의 전체 조직

AWS RAM에 대한 자세한 내용은 AWS RAM 사용 설명서를 참조하세요.

이 항목에서는 소유한 파라미터를 공유하는 방법과 사용자와 공유 파라미터를 사용하는 방법을 설명합니다.

내용

파라미터 공유를 위한 사전 조건

계정에서 파라미터를 공유하려면 다음 사전 조건을 충족해야 합니다.

- 파라미터를 공유하려면 AWS 계정에 소유하고 있어야 합니다. 공유 받은 파라미터는 공유할 수 없습니다.
- 파라미터를 공유하려면 해당 파라미터가 고급 파라미터 티어에 속해야 합니다. 파라미터 티어에 대한 자세한 내용은 파라미터 티어 관리 섹션을 참조하세요. 기존 표준 파라미터를 고급 파라미터로 변경하는 방법에 대한 자세한 내용은 표준 파라미터를 고급 파라미터로 변경 섹션을 참조하세요.
- `SecureString` 파라미터를 공유하려면 고객 관리 키로 암호화해야 하며, AWS Key Management Service를 통해 키를 별도로 공유해야 합니다. AWS 관리형 키는 공유할 수 없습니다. 기본 AWS 관리형 키로 암호화된 파라미터를 업데이트하여 고객 관리형 키를 대신 사용할 수 있습니다. AWS KMS 키 정의는 AWS Key Management Service 개발자 안내서의 AWS KMS 개념을 참조하세요.
- 파라미터를 AWS Organizations의 조직 또는 조직 단위와 공유하려면, AWS Organizations와의 공유를 활성화해야 합니다. 자세한 내용은 AWS RAM 사용 설명서의 AWS Organizations과(와) 공유 활성화를 참조하세요.

파라미터 공유

파라미터를 공유하려면 리소스 공유에 추가해야 합니다. 리소스 공유는 AWS 계정 전반에서 리소스를 공유할 수 있게 해주는 AWS RAM 리소스입니다. 리소스 공유는 공유할 리소스와 공유 대상 소비자를 지정합니다.

소유한 파라미터를 다른 AWS 계정과 공유하는 경우 두 가지 AWS 관리형 권한 중에서 선택하고 소비자에게 부여할 수 있습니다. 자세한 내용은 파라미터 공유를 위한 권한 세트 단원을 참조하십시오.

AWS Organizations의 조직에 속해 있고 조직 내의 공유가 활성화되어 있으면, 조직의 소비자에게 AWS RAM 콘솔에서 공유 파라미터에 대한 액세스 권한을 부여할 수 있습니다. 그렇지 않으면 소비자는 리소스 공유에 가입하라는 초대장을 받고 초대를 수락한 후 공유 파라미터에 대한 액세스 권한을 받습니다.

AWS RAM 콘솔이나 AWS CLI를 사용하여 자신이 소유한 파라미터를 공유할 수 있습니다.

> **Note**
>
> Systems Manager PutResourcePolicy API 작업을 사용하여 파라미터를 공유할 수 있지만, 대신 AWS Resource Access Manager(AWS RAM)를 사용하는 것이 좋습니다. `PutResourcePolicy`를 사용하려면 AWS RAM PromoteResourceShareCreatedFromPolicy API 작업을 사용하여 파라미터를 표준 리소스 공유로 승격하는 추가 단계가 필요하기 때문입니다. 그러지 않으면 `--shared` 옵션을 사용하는 Systems Manager DescribeParameters API 작업에서 파라미터를 반환하지 않습니다.

AWS RAM 콘솔을 사용하여 소유한 파라미터를 공유하려면

AWS RAM 사용 설명서의 Creating a resource share in AWS RAM을 참조하세요.

절차를 완료할 때 다음을 선택합니다.

- 1단계 페이지의 리소스에서 `Parameter Store Advanced Parameter`를 선택하고 고급 파라미터 계층에서 공유하려는 각 파라미터 상자를 선택합니다.
- 2단계 페이지의 관리 권한에서 소비자에게 부여할 권한을 선택합니다(이 주제의 뒷부분에 나오는 파라미터 공유를 위한 권한 세트 참조).

파라미터 공유 목표에 따라 다른 옵션을 선택합니다.

AWS CLI를 사용하여 소유한 파라미터를 공유하려면

create-resource-share 명령을 사용하여 새 리소스 공유에 파라미터를 추가합니다.

associate-resource-share 명령을 사용하여 기존 리소스 공유에 파라미터를 추가합니다.

다음 예제에서는 새 리소스 공유를 만들어 조직 및 개별 계정의 소비자와 파라미터를 공유합니다.

```
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"
```

공유 파라미터 공유 중지

공유 파라미터의 공유를 중지하면 소비자 계정이 더 이상 파라미터에 액세스할 수 없습니다.

소유한 파라미터의 공유를 중지하려면 리소스 공유에서 제거해야 합니다. 이를 위해 Systems Manager 콘솔, AWS RAM 콘솔 또는 AWS CLI를 사용할 수 있습니다.

AWS RAM 콘솔을 사용하여 소유한 파라미터 공유를 중지하려면

AWS RAM 사용 설명서의 AWS RAM에서 리소스 공유 업데이트를 참조하세요.

AWS CLI를 사용하여 소유한 파라미터 공유를 중지하려면

disassociate-resource-share 명령을 사용합니다.

공유 파라미터 식별

소유자와 소비자는 AWS CLI를 사용하여 공유된 파라미터를 식별할 수 있습니다.

AWS CLI를 사용하여 공유 파라미터를 식별하려면

AWS CLI를 사용하여 공유 파라미터를 식별하려면 Systems Manager `describe-parameters` 명령과 AWS RAM `list-resources`명령 중에서 선택할 수 있습니다.

`describe-parameters`와 함께 `--shared` 옵션을 사용하면 명령이 사용자와 공유된 파라미터를 반환합니다.

다음은 그 예제입니다.

```
aws ssm describe-parameters --shared
```

공유 파라미터 액세스

소비자는 AWS 명령줄 도구 및 AWS SDK를 사용하여 공유 파라미터에 액세스할 수 있습니다. 소비자 계정의 경우 해당 계정과 공유된 파라미터는 내 파라미터 페이지에 포함되지 않습니다.

CLI 예제: AWS CLI를 사용하여 공유 파라미터 세부 정보에 액세스

AWS CLI를 사용하여 공유 파라미터 세부 정보에 액세스하려면 get-parameter 또는 get-parameters 명령을 사용할 수 있습니다. 다른 계정에서 파라미터를 검색하려면 전체 파라미터 ARN을 `--name`으로 지정해야 합니다.

다음은 예입니다.

```
aws ssm get-parameter \
    --name arn:aws:ssm:us-east-2:123456789012:parameter/MySharedParameter
```

공유 파라미터에 지원되는 통합 및 지원되지 않는 통합

현재 다음과 같은 통합 시나리오에서 공유 파라미터를 사용할 수 있습니다.

- AWS CloudFormation 템플릿 파라미터
- AWS 파라미터 및 보안 암호 Lambda 확장
- Amazon Elastic Compute Cloud(EC2) 시작 템플릿
- Amazon Machine Image(AMI)에서 인스턴스를 생성하기 위한 EC2 RunInstances 명령의 `ImageID` 값

- Systems Manager의 기능인 Automation용 런북에서 파라미터 값 검색

다음 시나리오 및 통합 서비스는 현재 공유 파라미터 사용을 지원하지 않습니다.

- Systems Manager의 기능인 Run Command에서 명령의 파라미터
- AWS CloudFormation 동적 참조
- AWS CodeBuild에서 환경 변수의 값
- AWS App Runner에서 환경 변수의 값
- Amazon Elastic Container Service에서 보안 암호의 값

파라미터 공유를 위한 권한 세트

소비자 계정은 공유하는 파라미터에 대한 읽기 전용 액세스 권한을 받습니다. 소비자는 파라미터를 업데이트하거나 삭제할 수 없습니다. 소비자는 파라미터를 제3의 계정과 공유할 수 없습니다.

파라미터 공유를 위해 AWS Resource Access Manager에서 리소스 공유를 생성할 때 두 개의 AWS 관리형 권한 집합 중에서 선택하여 이 읽기 전용 액세스 권한을 부여할 수 있습니다.

AWSRAMDefaultPermissionSSMParameterReadOnly

허용된 작업: `DescribeParameters`, `GetParameter`, `GetParameters`

AWSRAMPermissionSSMParameterReadOnlyWithHistory

허용된 작업: `DescribeParameters`, `GetParameter`, `GetParameters`, `GetParameterHistory`

AWS RAM 사용 설명서의 Creating a resource share in AWS RAM에 나온 단계를 따를 때 리소스 유형으로 `Parameter Store Advanced Parameters`를 선택하고, 사용자가 파라미터 기록을 볼 수 있는지에 따라 해당 관리형 권한 중 하나를 선택합니다.

공유 파라미터의 최대 처리량

Systems Manager는 GetParameter 및 GetParameters 작업에 대한 최대 처리량(초당 트랜잭션 수)을 제한합니다. 처리량은 개별 계정 수준에서 적용됩니다. 따라서 공유 파라미터를 사용하는 각 계정은 다른 계정의 영향을 받지 않으면서 최대 허용 처리량을 사용할 수 있습니다. 파라미터 최대 처리량에 대한 자세한 내용은 다음 항목을 참조하세요.

- Parameter Store 처리량 증가

- Amazon Web Services 일반 참조의 Systems Manager 서비스 할당량

공유 파라미터 요금

계정 간 공유는 고급 파라미터 티어에서만 사용할 수 있습니다. 고급 파라미터의 경우 각 고급 파라미터의 스토리지 및 API 사용량에 대해 현재 가격으로 요금이 부과됩니다. 고급 파라미터 저장은 소유 계정에 요금이 부과됩니다. 공유된 고급 파라미터에 대해 API를 호출하는 모든 소비 계정에 파라미터 사용 요금이 청구됩니다.

예를 들어 계정 A가 고급 파라미터 `MyAdvancedParameter`를 생성한 경우 해당 계정에 파라미터 저장 요금이 매월 0.05 USD 청구됩니다.

그다음 계정 A가 계정 B 및 계정 C와 `MyAdvancedParameter`를 공유합니다. 한 달 동안 세 계정이 `MyAdvancedParameter`를 호출합니다. 다음 표는 각 호출 횟수에 따라 발생하는 요금을 보여줍니다.

> **Note**
>
> 다음 표의 요금은 설명을 돕기 위한 것입니다. 현재 가격을 확인하려면 Parameter Store의 AWS Systems Manager 가격 책정을 참조하세요.

계정	호출 횟수	요금
계정 A(소유 계정)	호출 10,000회	• 1개월 고급 파라미터 저장: 0.05 USD • `MyAdvancedParameter` 에 대한 호출 10,000회: 0.05 USD • 총액: 0.10 USD
계정 B(소비 계정)	호출 20,000회	• `MyAdvancedParameter` 에 대한 호출 20,000회: 0.10 USD • 총액: 0.10 USD
계정 C(소비 계정)	호출 30,000회	• `MyAdvancedParameter` 에 대한 호출 30,000회: 0.15 USD

계정	호출 횟수	요금
		• 총액: 0.15 USD

해지된 AWS 계정에 대한 크로스 계정 액세스

공유 파라미터를 소유한 AWS 계정이 해지되면 모든 소비 계정은 공유 파라미터에 대한 액세스 권한을 잃게 됩니다. 계정이 해지된 후 90일 이내에 소유 계정을 다시 열면 소비 계정은 이전에 공유한 파라미터에 다시 액세스할 수 있습니다. 해지 후 기간 동안 계정 다시 열기에 대한 자세한 내용은 AWS Account Management 참조 안내서의 AWS 계정 해지 후 액세스를 참조하세요.

Run Command 명령을 사용하여 파라미터 작업

Run Command의 기능인 AWS Systems Manager에서 파라미터로 작업할 수 있습니다. 자세한 내용은 AWS Systems Manager Run Command 단원을 참조하십시오.

String 파라미터 실행(콘솔)

다음 절차는 `String` 파라미터를 사용하는 명령을 실행하는 과정을 안내합니다.

Parameter Store를 사용하여 문자열 파라미터를 실행하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택합니다.
3. Run command(Run 명령)를 선택합니다.
4. 명령 문서 목록에서 `AWS-RunPowerShellScript`(Windows) 또는 `AWS-RunShellScript`(Linux)를 선택합니다.
5. Command parameters(명령 파라미터)에 **`echo {{ssm:parameter-name}}`**를 입력합니다. 예: **`echo {{ssm:/Test/helloWorld}}`**.
6. 대상 섹션에서, 태그를 지정하거나, 수동으로 인스턴스나 엣지 디바이스를 선택하거나, 리소스 그룹을 지정하여 이 작업을 실행할 관리형 노드를 식별합니다.

   > **Tip**
   >
   > 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

7. Other parameters(다른 파라미터):

- Comment(설명)에 명령에 대한 정보를 입력합니다.
- 제한 시간(초)에서 전체 명령 실행이 실패할 때까지 시스템이 기다리는 시간을 초 단위로 지정합니다.

8. Rate control(속도 제어)에서

   - Concurrency(동시성)에서 명령을 동시에 실행할 관리형 노드의 백분율 또는 개수를 지정합니다.

     > **Note**
     >
     > 관리형 노드에 적용할 태그를 지정하거나, AWS 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상 수를 제한합니다.

   - Error threshold(오류 임계값)에서, 명령이 노드의 개수 또는 백분율에서 실패한 후 다른 관리형 노드에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째 오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.

9. (선택 사항) Output options(출력 옵션)에서 명령 출력을 파일에 저장하려면 Write command output to an S3 bucket(S3 버킷에 명령 출력 쓰기) 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

   > **Note**
   >
   > 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아니라 인스턴스에 할당된 인스턴스 프로파일(EC2 인스턴스용) 또는 IAM 서비스 역할(하이브리드 정품 인증 시스템)의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

10. SNS notifications(SNS 알림) 섹션에서, 명령 실행 상태에 대한 알림이 전송되도록 하려면 Enable SNS notifications(SNS 알림 활성화) 확인란을 선택합니다.

    Run Command에 대한 Amazon SNS 알림 구성에 대한 자세한 내용은 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 섹션을 참조하세요.

11. Run(실행)을 선택합니다.

12. 명령 ID 페이지의 대상 및 출력(Targets and outputs) 영역에서 명령을 실행한 노드의 ID 옆에 있는 버튼을 선택한 다음, 출력 보기(View output)를 선택합니다. 명령의 출력이 파라미터에 제공한 값(예: **`This is my first parameter`**)인지 확인합니다.

파라미터 실행(AWS CLI)

예 1: 단순 명령

다음 예제의 명령에는 `DNS-IP`라는 Systems Manager 파라미터가 포함되어 있습니다. 이 파라미터의 값은 단순히 노드의 IP 주소입니다. 이 예에서는 AWS Command Line Interface(AWS CLI) 명령을 사용하여 파라미터 값을 반향합니다.

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
```

명령은 다음과 같은 정보를 반환합니다.

```
{
    "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": []
    }
}
```

명령 실행을 마친 후 다음 명령을 사용하여 해당 명령에 대한 자세한 정보를 볼 수 있습니다.

- get-command-invocation - 명령 실행에 대한 세부 정보를 확인합니다.
- list-command-invocations - 특정 관리형 노드의 명령 실행 상태를 확인합니다.
- list-commands - 여러 관리형 노드의 명령 실행 상태를 확인합니다.

예 2: **SecureString** 파라미터 값 암호화 해제

다음 예제 명령은 SecurePassword라는 SecureString 파라미터를 사용합니다. parameters 필드에서 사용되는 명령은 SecureString 파라미터 값을 검색하고 암호를 복호화한 다음 로컬 관리자 암호를 일반 텍스트로 전달하지 않고도 재설정합니다.

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
```

예 3: SSM 문서에서 파라미터 참조

다음 예에서 보듯이 SSM 문서의 [파라미터(Parameters)] 섹션에서 Systems Manager 파라미터를 참조할 수도 있습니다.

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

SSM 문서의 `runtimeConfig` 섹션에 사용된 로컬 파라미터에 대한 유사한 구문을 Parameter Store 파라미터와 혼동하지 마세요. 로컬 파라미터는 Systems Manager 파라미터와 동일하지 않습니다. 로컬 파라미터는 `ssm:` 접두사가 없다는 점에서 Systems Manager 파라미터와 구별됩니다.

```
"runtimeConfig":{
        "aws:runShellScript":{
            "properties":[
                {
                    "id":"0.aws:runShellScript",
                    "runCommand":"{{ commands }}",
                    "workingDirectory":"{{ workingDirectory }}",
```

```
                "timeoutSeconds":"{{ executionTimeout }}"
```

> **Note**
>
> SSM 문서는 SecureString 파라미터에 대한 참조를 지원하지 않습니다. 다시 말해 SecureString 등과 함께 Run Command 파라미터를 사용하기 위해서는 다음 예제에서와 같이 Run Command로 전달하기 전에 파라미터 값을 검색해야 합니다.
>
> 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
> ```

Amazon Machine Image ID에 대한 기본 파라미터 지원

String 파라미터를 생성할 때 이제 데이터 형식을 aws:ec2:image로 지정하여 입력한 파라미터 값이 유효한 Amazon Machine Image(AMI) ID 형식인지 확인할 수 있습니다.

AMI ID 형식을 지원하므로 프로세스에서 사용하려는 AMI가 변경될 때마다 모든 스크립트 및 템플릿을 새 ID로 업데이트하지 않아도 됩니다. `aws:ec2:image` 데이터 형식을 사용하여 파라미터를 생성하고 해당 값에 AMI의 ID를 입력할 수 있습니다. 이것은 지금 새 인스턴스를 생성하려는 AMI입니다. 그런 다음 템플릿, 명령 및 스크립트에서 이 파라미터를 참조합니다.

예를 들어 Amazon Elastic Compute Cloud(Amazon EC2) `run-instances` 명령을 실행할 때 선호하는 AMI ID가 포함된 파라미터를 지정할 수 있습니다.

> **Note**
>
> 이 명령을 실행하는 사용자에게 `ssm:GetParameters` API 작업이 포함된 AWS Identity and Access Management(IAM) 권한이 있어야 파라미터 값을 검증할 수 있습니다. 그렇지 않으면 파라미터 생성 프로세스가 실패합니다.

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
```

또한 Amazon EC2 콘솔을 사용하여 인스턴스를 생성할 때 원하는 AMI를 선택할 수 있습니다. 자세한 내용은 Amazon EC2 사용 설명서의 AMI를 찾기 위해 Systems Manager 파라미터 사용을 참조하세요.

인스턴스 생성 워크플로에서 다른 AMI를 사용해야 하는 경우 파라미터를 새 AMI 값으로 업데이트만 하면 되며 Parameter Store는 올바른 형식의 ID를 입력했는지 다시 검증합니다.

aws:ec2:image 데이터 형식의 파라미터를 생성할 수 있는 권한 부여

AWS Identity and Access Management(IAM) 정책을 사용하여 Parameter Store API 작업 및 콘텐츠에 대한 사용자 액세스를 제공하거나 제한할 수 있습니다.

aws:ec2:image 데이터 유형 파라미터를 만들려면 사용자에게 ssm:PutParameter 및 ec2:DescribeImages 권한이 모두 있어야 합니다.

다음 예제 정책은 사용자에게 aws:ec2:image에 대한 PutParameter API 작업을 호출할 수 있는 권한을 부여합니다. 이는 사용자가 aws:ec2:image 데이터 형식의 파라미터를 시스템에 추가할 수 있음을 의미합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ssm:PutParameter",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeImages",
            "Resource": "*"
        }
    ]
}
```

AMI 형식 검증이 수행되는 방법

파라미터에 대한 데이터 형식으로 aws:ec2:image를 지정하면 Systems Manager는 파라미터를 즉시 생성하지 않습니다. 대신, 비동기 검증 작업을 수행하여 파라미터 값이 AMI ID의 형식 지정 요구 사항을 충족하는지와 지정된 AMI를 AWS 계정에서 사용할 수 있는지 확인합니다.

검증 작업이 완료되기 전에 파라미터 버전 번호가 생성될 수 있습니다. 파라미터 버전 번호가 생성되더라도 작업이 완료되지 않을 수 있습니다.

파라미터가 생성되었는지 모니터링하려면 Amazon EventBridge를 사용하여 파라미터 작업의 create 및 update에 대한 알림을 받는 것이 좋습니다. 이러한 알림은 파라미터 작업의 성공 여부를 보고합니다. 작업이 실패하는 경우 실패의 원인을 나타내는 오류 메시지가 알림에 포함되어 있습니다.

```
{
```

```
    "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"
    }
}
```

EventBridge에서 Parameter Store 이벤트 구독에 대한 자세한 내용은 Parameter Store 이벤트 기반 알림 설정 또는 작업 트리거 섹션을 참조하세요.

Systems Manager 파라미터 삭제

이 주제에서는 AWS Systems Manager의 기능인 Parameter Store에서 생성한 파라미터를 삭제하는 방법을 설명합니다.

파라미터를 삭제하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Parameter Store를 선택합니다.
3. 내 파라미터(My parameters) 탭에서 삭제할 각 파라미터 옆의 확인란을 선택합니다.
4. 삭제를 선택합니다.
5. 확인 대화 상자에서 파라미터 삭제(Delete parameters)를 선택합니다.

파라미터를 삭제하려면(AWS CLI)

- 다음 명령을 실행합니다:

```
aws ssm delete-parameter --name "my-parameter"
```

*my-parameter*를 삭제할 파라미터의 이름으로 바꿉니다.

`delete-parameter` 명령과 함께 사용할 수 있는 모든 옵션에 대한 자세한 내용은 AWS CLI 명령 참조의 AWS Systems Manager 섹션에 있는 delete-parameter를 참조하세요.

퍼블릭 파라미터 작업

일부 AWS 서비스은(는) 공통 아티팩트에 대한 정보를 AWS Systems Manager 공개 파라미터로 게시합니다. 예를 들어 Amazon Elastic Compute Cloud(Amazon EC2) 서비스는 Amazon Machine Images(AMIs)에 대한 정보를 퍼블릭 파라미터로 게시합니다.

이 안내서의 주제

- 퍼블릭 파라미터 찾기
- AMI 퍼블릭 파라미터 호출
- ECS에 최적화된 AMI 퍼블릭 파라미터 호출
- EKS에 최적화된 AMI 퍼블릭 파라미터 호출
- AWS 서비스, 리전, 엔드포인트, 가용 영역, 로컬 영역 및 Wavelength Zone에 대한 퍼블릭 파라미터 호출

관련 AWS 블로그 게시물

- AWS Systems ManagerParameter Store를 사용하여 AWS 리전, 엔드포인트 등 쿼리
- AWS Systems ManagerParameter Store를 사용하여 최신 Amazon Linux AMI ID 쿼리
- AWS Systems ManagerParameter Store를 사용하여 최신 Windows AMI 쿼리

퍼블릭 파라미터 찾기

Parameter Store 콘솔이나 AWS Command Line Interface를 사용하여 퍼블릭 파라미터를 검색할 수 있습니다.

퍼블릭 파라미터 이름은 `aws/service/list`로 시작합니다. 이름의 다음 부분은 해당 파라미터를 소유하는 서비스에 해당합니다.

다음은 퍼블릭 파라미터를 제공하는 일부 서비스 목록입니다.

- `ami-amazon-linux-latest`

- ami-windows-latest
- appmesh
- aws-for-fluent-bit
- bottlerocket
- canonical
- cloud9
- datasync
- debian
- ecs
- eks
- freebsd
- global-infrastructure
- marketplace
- storagegateway

모든 퍼블릭 파라미터가 모든 AWS 리전에 게시되지는 않습니다.

Parameter Store 콘솔을 사용하여 퍼블릭 파라미터 찾기

콘솔을 사용하여 퍼블릭 파라미터를 검색하려면 먼저 AWS 계정 및 AWS 리전에 하나 이상의 파라미터가 있어야 합니다.

콘솔을 사용하여 퍼블릭 파라미터를 찾으려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Parameter Store를 선택합니다.
3. [퍼블릭 파라미터(Public parameters)] 탭을 선택합니다.
4. [서비스 선택(Select a service)] 드롭다운을 선택합니다. 파라미터를 사용하려는 서비스를 선택합니다.
5. (선택 사항) 검색 창에 추가 정보를 입력하여 선택한 서비스가 소유한 파라미터를 필터링합니다.
6. 사용할 퍼블릭 파라미터를 선택합니다.

AWS CLI를 사용하여 퍼블릭 파라미터 찾기

퍼블릭 파라미터 검색에 `describe-parameters`를 사용합니다.

`get-parameters-by-path`를 사용하여 `/aws/service/list` 아래에 나열된 서비스의 실제 경로를 가져올 수 있습니다. 서비스의 경로를 가져오려면 경로에서 `/list`를 이동합니다. 예를 들어, `/aws/service/list/ecs`는 `/aws/service/ecs`가 됩니다.

Parameter Store의 다른 서비스가 소유한 퍼블릭 파라미터 목록을 검색하려면 다음 명령을 실행합니다.

```
aws ssm get-parameters-by-path --path /aws/service/list
```

명령은 다음과 같은 정보를 반환합니다. 이 예제는 공백을 위해 잘렸습니다.

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

특정 서비스가 소유한 파라미터를 보려면 이전 명령을 실행한 후 생성된 목록에서 서비스를 선택합니다. 그런 다음 원하는 서비스 이름을 사용하여 `get-parameters-by-path`를 호출합니다.

예를 들면 `/aws/service/global-infrastructure`입니다. 경로는 한 수준(지정된 정확한 값과 일치하는 파라미터만 호출) 또는 재귀적(지정한 것 이상의 경로에 요소 포함)일 수 있습니다.

> **Note**
>
> 일부 리전에서는 쿼리에 `/aws/service/global-infrastructure` 경로가 지원되지 않습니다. 자세한 설명은 AWS 서비스, 리전, 엔드포인트, 가용 영역, 로컬 영역 및 Wavelength Zone에 대한 퍼블릭 파라미터 호출을 참조하세요.

지정한 서비스에 대한 결과가 반환되지 않은 경우에 `--recursive` 플래그를 추가하고 명령을 다시 실행합니다.

```
aws ssm get-parameters-by-path --path /aws/service/global-infrastructure
```

`global-infrastructure`가 소유한 모든 파라미터가 반환됩니다. 다음은 예입니다.

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

Option:BeginsWith 필터를 사용하여 특정 서비스가 소유한 파라미터를 볼 수도 있습니다.

```
aws ssm describe-parameters --parameter-filters "Key=Name, Option=BeginsWith, Values=/
aws/service/ami-amazon-linux-latest"
```

명령은 다음과 같은 정보를 반환합니다. 이 예제 출력은 공백을 위해 잘렸습니다.

```
{
    "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**
>
> 다른 검색 패턴을 사용하기 때문에 `Option=BeginsWith`를 사용할 때 반환되는 파라미터가 다를 수 있습니다.

AMI 퍼블릭 파라미터 호출

Amazon Elastic Compute Cloud(Amazon EC2) Amazon Machine Image(AMI) 퍼블릭 파라미터는 다음 경로에서 Amazon Linux 1, Amazon Linux 2, Amazon Linux 2023(AL2023), Windows Server에 대해 사용할 수 있습니다.

- Amazon Linux 1, Amazon Linux 2, Amazon Linux 2023: `/aws/service/ami-amazon-linux-latest`
- Windows Server: `/aws/service/ami-windows-latest`

Amazon Linux 1, Amazon Linux 2, Amazon Linux 2023용 AMI 퍼블릭 파라미터 호출

AWS Command Line Interface(AWS CLI)에서 다음 명령을 사용하여 현재 AWS 리전의 모든 Amazon Linux 1, Amazon Linux 2, Amazon Linux 2023(AL2023) AMIs 목록을 볼 수 있습니다.

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
```

명령은 다음과 같은 정보를 반환합니다.

```
[
    "/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"
]
```

다음 명령을 사용하여 AMI ID 및 Amazon 리소스 이름(ARN)을 포함하여 이러한 AMIs에 대한 세부 정보를 볼 수 있습니다.

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
```

##은 미국 동부(오하이오) 리전의 us-east-2 같이 AWS Systems Manager가 지원하는 AWS 리전의 식별자를 나타냅니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

명령은 다음과 같은 정보를 반환합니다. 이 예제 출력은 공백을 위해 잘렸습니다.

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

경로를 포함하여 전체 AMI 이름과 함께 GetParameters API 작업을 사용하여 특정 AMI의 세부 정보를 볼 수 있습니다. 다음은 명령 예제입니다.

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
```

명령은 다음과 같은 정보를 반환합니다.

```
{
    "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": []
```

```
}
```

Windows Server에 대한 AMI 퍼블릭 파라미터 호출

AWS CLI에서 다음 명령을 사용하여 현재 AWS 리전의 모든 Windows Server AMIs 목록을 볼 수 있습니다.

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
```

명령은 다음과 같은 정보를 반환합니다. 이 예제 출력은 공백을 위해 잘렸습니다.

```
[
    "/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
]
```

다음 명령을 사용하여 AMI ID 및 Amazon 리소스 이름(ARN)을 포함하여 이러한 AMIs에 대한 세부 정보를 볼 수 있습니다.

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
```

*##*은 미국 동부(오하이오) 리전의 `us-east-2` 같이 AWS Systems Manager가 지원하는 AWS 리전의 식별자를 나타냅니다. 지원되는 *##* 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

명령은 다음과 같은 정보를 반환합니다. 이 예제 출력은 공백을 위해 잘렸습니다.


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


경로를 포함하여 전체 AMI 이름과 함께 GetParameters API 작업을 사용하여 특정 AMI의 세부 정보를 볼 수 있습니다. 다음은 명령 예제입니다.

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
```

명령은 다음과 같은 정보를 반환합니다.

```
{
    "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": []
}
```

ECS에 최적화된 AMI 퍼블릭 파라미터 호출

Amazon Elastic Container Service(Amazon ECS) 서비스는 최신 Amazon ECS에 최적화된 Amazon Machine Images(AMIs)의 이름을 퍼블릭 파라미터로 게시합니다. 최적화된 AMIs에는 버그 수정 및 기능 업데이트가 포함되므로 Amazon ECS용 새 Amazon Elastic Compute Cloud(Amazon EC2) 클러스터를 생성할 때 이 AMI를 사용하는 것이 좋습니다.

Amazon Linux 2용 최신 Amazon ECS에 최적화된 AMI의 이름을 보려면 다음 명령을 사용합니다. 다른 운영 체제의 명령을 보려면 Amazon Elastic Container Service Developer Guide의 Retrieving Amazon ECS-Optimized AMI metadata를 참조하세요.

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
```

명령은 다음과 같은 정보를 반환합니다.

```
{
    "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": []
}
```

EKS에 최적화된 AMI 퍼블릭 파라미터 호출

Amazon Elastic Kubernetes Service(Amazon EKS) 서비스는 최신 Amazon EKS에 최적화된 Amazon Machine Image(AMI)의 이름을 퍼블릭 파라미터로 게시합니다. 새 릴리스에는 Kubernetes 패치 및 보안 업데이트가 포함되어 있으므로 Amazon EKS 클러스터에 노드를 추가할 때 이 AMI를 사용하는

것이 좋습니다. 이전에는 최신 AMI를 사용했는지 확인하기 위해 Amazon EKS 설명서를 확인하고 새 AMI ID로 배포 템플릿 또는 리소스를 수동으로 업데이트해야 했습니다.

Amazon Linux 2용 최신 Amazon EKS에 최적화된 AMI의 이름을 보려면 다음 명령을 사용합니다.

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
```

명령은 다음과 같은 정보를 반환합니다.

```
{
    "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": []
}
```

AWS 서비스, 리전, 엔드포인트, 가용 영역, 로컬 영역 및 Wavelength Zone에 대한 퍼블릭 파라미터 호출

다음 경로를 사용하여 AWS 리전, 서비스, 엔드포인트, 가용성 및 퍼블릭 파라미터의 Wavelength Zone을 호출할 수 있습니다.

`/aws/service/global-infrastructure`

> **Note**
>
> 현재 `/aws/service/global-infrastructure` 경로는 다음 AWS 리전의 쿼리에만 지원됩니다.
>
> - 미국 동부(버지니아 북부)(us-east-1)
> - 미국 동부(오하이오)(us-east-2)
> - 미국 서부(캘리포니아 북부) (us-west-1)
> - 미국 서부(오레곤)(us-west-2)
> - 아시아 태평양(홍콩)(ap-east-1)
> - 아시아 태평양(뭄바이)(ap-south-1)
> - 아시아 태평양(서울)(ap-northeast-2)
> - 아시아 태평양(싱가포르)(ap-southeast-1)
> - 아시아 태평양(시드니)(ap-southeast-2)
> - 아시아 태평양(도쿄)(ap-northeast-1)
> - 캐나다(중부)(ca-central-1)
> - 유럽(프랑크푸르트)(eu-central-1)
> - 유럽(아일랜드)(eu-west-1)
> - 유럽(런던) (eu-west-2)
> - 유럽(파리) (eu-west-3)
> - 유럽(스톡홀름) (eu-north-1)
> - 남아메리카(상파울루)(sa-east-1)
>
> 다른 상업 리전에서 작업하는 경우, 쿼리에 지원 리전을 지정하여 결과를 볼 수 있습니다. 예를 들어, 캐나다 서부(캘거리)(ca-west-1) 리전에서 작업하는 경우 쿼리에 캐나다(중부)(ca-central-1)를 지정할 수 있습니다.
>
> ```
> aws ssm get-parameters-by-path \
>     --path /aws/service/global-infrastructure/regions \
>     --region ca-central-1
> ```

활성 AWS 리전 보기

AWS Command Line Interface(AWS CLI)에서 다음 명령을 사용하여 모든 활성 AWS 리전 목록을 볼 수 있습니다.

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
```

명령은 다음과 같은 정보를 반환합니다.

```
[
    "/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"
]
```

사용 가능한 AWS 서비스 보기

사용 가능한 모든 AWS 서비스의 전체 목록을 보고 다음 명령을 사용하여 사전순으로 정렬할 수 있습니다. 이 예제 출력은 공백을 위해 잘렸습니다.

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(@)"
```

명령은 다음과 같은 정보를 반환합니다. 이 예제는 공백을 위해 잘렸습니다.

```
[
     "/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",
```

AWS 서비스에 대해 지원되는 리전 보기

서비스를 사용할 수 있는 AWS 리전 목록을 볼 수 있습니다. 이 예에서는 AWS Systems Manager(ssm)를 사용합니다.

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
```

명령은 다음과 같은 정보를 반환합니다.

```
[
    "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"
]
```

서비스에 대한 리전 엔드포인트 보기

다음 명령을 사용하여 서비스의 리전별 엔드포인트를 볼 수 있습니다. 이 명령은 미국 동부(오하이오) (us-east-2) 리전을 쿼리합니다.

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
```

명령은 다음과 같은 정보를 반환합니다.

```
"ssm.us-east-2.amazonaws.com"
```

전체 가용 영역 세부 정보 보기

다음 명령을 사용하여 가용 영역을 볼 수 있습니다.

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/
```

명령은 다음과 같은 정보를 반환합니다. 이 예제는 공백을 위해 잘렸습니다.

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

가용 영역 이름만 보기

다음 명령을 사용하여 가용 영역의 이름만 볼 수 있습니다.

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(@)"
```

명령은 다음과 같은 정보를 반환합니다. 이 예제는 공백을 위해 잘렸습니다.

```
[
    "/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"
```

단일 리전의 가용 영역 이름 보기

다음 명령을 사용하여 한 리전(이 예제의 경우 us-east-2)에 있는 가용 영역의 이름을 볼 수 있습니다.

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(@)"
```

명령은 다음과 같은 정보를 반환합니다.

```
[
    "/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"
```

가용 영역 ARN만 보기

다음 명령을 사용하여 가용 영역의 Amazon 리소스 이름(ARN)만 볼 수 있습니다.

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(@)"
```

명령은 다음과 같은 정보를 반환합니다. 이 예제는 공백을 위해 잘렸습니다.

```
[
    "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",
```

로컬 영역 세부 정보 보기

다음 명령을 사용하여 로컬 영역을 볼 수 있습니다.

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
```

명령은 다음과 같은 정보를 반환합니다. 이 예제는 공백을 위해 잘렸습니다.

```
{
    "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"
        }
    ]
```

```
}
```

Wavelength Zone 세부 정보 보기

다음 명령을 사용하여 Wavelength Zone을 볼 수 있습니다.

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
```

명령은 다음과 같은 정보를 반환합니다. 이 예제는 공백을 위해 잘렸습니다.

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

로컬 영역에 있는 모든 파라미터 및 값 보기

다음 명령을 사용하여 로컬 영역에 대한 모든 파라미터 데이터를 볼 수 있습니다.

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"
```

명령은 다음과 같은 정보를 반환합니다. 이 예제는 공백을 위해 잘렸습니다.

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

로컬 영역 파라미터 이름만 보기

다음 명령을 사용하여 로컬 영역 파라미터의 이름만 볼 수 있습니다.

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(@)"
```

명령은 다음과 같은 정보를 반환합니다.

```
[
    "/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 연습

이 섹션의 시연에서는 테스트 환경에서 AWS Systems Manager의 기능인 Parameter Store로 파라미터를 생성, 저장 및 실행하는 방법을 보여줍니다. 이 시연에서는 Parameter Store를 다른 Systems Manager 기능과 함께 사용하는 방법을 보여줍니다. 다른 AWS 서비스와(과) 함께 Parameter Store을(를) 사용할 수도 있습니다. 자세한 내용은 파라미터란 무엇인가요? 단원을 참조하십시오.

목차

SecureString 파라미터를 생성하고 도메인에 노드 조인(PowerShell)

이 연습에서는 AWS Systems Manager `SecureString` 파라미터와 Run Command를 사용하여 Windows Server 노드를 도메인에 조인하는 방법을 보여줍니다. 본 연습에서는 도메인 이름 및 도메인 사용자 이름과 같은 일반적인 도메인 파라미터를 사용합니다. 이러한 값은 암호화되지 않은 문자열 값으로 전달됩니다. 도메인 암호는 AWS 관리형 키를 통해 암호화되어 암호화된 문자열로 전달됩니다.

필수 조건

이 시연에서는 Amazon VPC와 연결된 DHCP 옵션 집합에 도메인 이름과 DNS 서버 IP 주소를 이미 지정했다고 가정합니다. 자세한 내용은 Amazon VPC 사용 설명서의 DHCP 옵션 집합을 사용한 작업을 참조하세요.

SecureString 파라미터를 생성하고 도메인에 노드를 조인

1. AWS Tools for Windows PowerShell를 사용하여 시스템에 파라미터를 입력합니다.

   다음 명령에서는 자신의 정보로 각각의 *### ## ## ###*를 바꿉니다.

   ```
   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**
   >
   > SecureString 파라미터의 값만 암호화됩니다. 파라미터 이름, 설명 및 기타 속성은 암호화되지 않습니다.

2. 다음 AWS Identity and Access Management(IAM) 정책을 노드에 대한 IAM 역할 권한에 연결합니다.

   - AmazonSSMManagedInstanceCore - 필수. 이 AWS 관리형 정책을 사용하면 관리형 노드가 Systems Manager 서비스 핵심 기능을 사용할 수 있습니다.
   - AmazonSSMDirectoryServiceAccess - 필수. 이 AWS 관리형 정책은 SSM Agent가 관리형 노드의 도메인 조인 요청을 위해 사용자 대신 AWS Directory Service에 액세스하도록 허용합니다.
   - Amazon Simple Storage Service(Amazon S3) 버킷 액세스를 위한 사용자 정의 정책 – 필수. 노드에 있으면서 Systems Manager 태스크를 수행하는 SSM Agent에는 특정 Amazon 소유의 Amazon Simple Storage Service(Amazon S3) 버킷에 대한 액세스가 필요합니다. 생성한 사용자 정의 S3 버킷 정책에서는 Systems Manager 작업에 필요한 자체의 S3 버킷에 대한 액세스도 제공합니다.

     예제: Run Command 명령 또는 Session Manager 세션에 대한 출력을 S3 버킷에 쓴 다음, 감사 또는 문제 해결을 위해 나중에 이 출력을 사용할 수 있습니다. 액세스 스크립트 또는 사용자 지정 패치 기준 목록을S3 버킷에 저장한 다음, 명령을 실행하거나 패치 기준이 적용될 때 스크립트 또는 목록을 참조할 수 있습니다.

     Amazon S3 버킷 액세스를 위한 사용자 정의 정책 생성에 대한 자세한 내용은 인스턴스 프로파일에 대한 사용자 정의 S3 버킷 정책 생성을 참조하세요.

> **Note**
>
> S3 버킷에 출력 로그 데이터 저장은 옵션이지만 이 옵션을 사용하기로 결정한 경우에는 Systems Manager 구성 프로세스 시작 시 옵션을 설정하는 것이 좋습니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 버킷 생성을 참조하세요.

- CloudWatchAgentServerPolicy - 선택. 이 AWS 관리형 정책은 관리형 노드에서 CloudWatch 에이전트를 실행하도록 허용합니다. 이 정책을 사용하면 노드에서 정보를 읽고 Amazon CloudWatch에 쓸 수 있습니다. Amazon EventBridge 또는 CloudWatch Logs 등의 서비스를 사용할 경우에만 인스턴스 프로파일에 이 정책이 필요합니다.

> **Note**
>
> CloudWatch 및 EventBridge 기능의 사용은 선택적이지만, 이 기능을 사용하기로 결정한 경우에는 Systems Manager 구성 프로세스 시작 시에 설정하는 것이 좋습니다. 자세한 내용은 Amazon EventBridge User Guide와 Amazon CloudWatch Logs User Guide를 참조하세요.

3. 노드에 연결된 IAM 역할을 편집하고 다음 정책을 추가합니다. 이 정책은 `kms:Decrypt`와 `ssm:CreateDocument` API를 호출하는 노드 권한을 부여합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "ssm:CreateDocument"
            ],
            "Resource": [
                "arn:aws:kms:region:account-id:key/kms-key-id"
            ]
        }
    ]
}
```

4. 다음 json 텍스트를 복사하여 텍스트 편집기에 붙여넣은 다음 이 파일을 `c:\temp\JoinInstanceToDomain.json` 위치에 `JoinInstanceToDomain.json`으로 저장합니다.

```
{
    "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. Tools for Windows PowerShell에서 다음 명령을 실행하여 새 SSM 문서를 생성합니다.

```
$json = Get-Content C:\temp\JoinInstanceToDomain | Out-String
New-SSMDocument -Name JoinInstanceToDomain -Content $json -DocumentType Command
```

6. Tools for Windows PowerShell에서 다음 명령을 실행하여 노드를 도메인에 조인합니다.

```
Send-SSMCommand -InstanceId instance-id -DocumentName JoinInstanceToDomain
```

명령이 제대로 실행되면 시스템에서 다음과 비슷한 정보를 반환합니다.

```
WARNING: The changes will take effect after you restart the computer EC2ABCD-
EXAMPLE.
Domain join succeeded, restarting
Computer is part of example.local, exiting
```

명령이 제대로 실행되지 않으면 시스템에서 다음과 비슷한 정보를 반환합니다.

```
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.
```

Amazon Elastic Kubernetes Service에서 Parameter Store 파라미터 사용

Secrets Manager의 보안 암호와 Parameter Store의 파라미터를 Amazon EKS 포드에 탑재된 파일로 표시하려면 Kubernetes 보안 암호 스토어 CSI 드라이버에 대해 AWS 보안 암호 및 구성 공급자(ASCP)를 사용할 수 있습니다.(Parameter Store는 AWS Systems Manager의 기능입니다.) ASCP는 Amazon Elastic Kubernetes Service(Amazon EKS) 1.17+에서 작동합니다. AWS Fargate (Fargate) 노드 그룹은 지원되지 않습니다.

ASCP를 사용하여 Parameter Store에서 저장 및 관리되는 파라미터를 검색할 수 있습니다. 그런 다음 Amazon EKS에서 실행되는 워크로드에서 파라미터를 사용할 수 있습니다. 파라미터에 JSON 형식의 여러 키 값 쌍이 포함되어 있는 경우 Amazon EKS에서 탑재할 키 값 쌍을 선택할 수 있습니다. ASCP는 JMESPath 구문을 사용하여 파라미터의 키 값 쌍을 쿼리할 수 있습니다.

AWS Identity and Access Management(IAM) 역할 및 정책을 사용하여 파라미터에 대한 액세스를 클러스터의 특정 Amazon EKS 포드로 제한할 수 있습니다. ASCP는 포드 자격 증명을 검색하고 IAM 역할에 대한 자격 증명을 교환합니다. ASCP는 포드의 IAM 역할을 맡습니다. 그런 다음 Parameter Store에서 해당 역할에 대해 권한이 부여된 파라미터를 검색할 수 있습니다.

Secrets Manager와 Amazon EKS를 통합하는 방법에 대한 자세한 내용은 Amazon Elastic Kubernetes Service에서 Secrets Manager 사용을 참조하세요.

ASCP 설치

ASCP는 secrets-store-csi-driver-provider-aws 리포지토리의 GitHub에서 제공됩니다. 리포지토리에는 보안 암호를 생성하고 탑재하기 위한 예제 YAML 파일도 포함되어 있습니다. 먼저 Kubernetes 보안 암호 스토어 CSI 드라이버를 설치한 후 ASCP를 설치합니다.

Kubernetes 보안 암호 스토어 CSI 드라이버 및 ASCP를 설치하려면

1. Kubernetes 보안 암호 스토어 CSI 드라이버를 설치하려면 다음 명령을 실행합니다. 전체 설치 지침은 Kubernetes 보안 암호 스토어 CSI 드라이버 북의 설치를 참조하세요. Helm 설치에 대한 자세한 내용은 Amazon EKS에서 Helm 사용을 참조하세요.

   ```
   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
   ```

2. ASCP를 설치하려면 GitHub 리포지토리 배포 디렉터리에서 YAML 파일을 사용합니다. `kubectl` 설치에 대한 자세한 내용은 `kubectl` 설치를 참조하세요.

   ```
   kubectl apply -f https://raw.githubusercontent.com/aws/secrets-store-csi-driver-provider-aws/main/deployment/aws-provider-installer.yaml
   ```

1단계: 액세스 제어 설정

Parameter Store에서 Amazon EKS 포드에 파라미터에 대한 액세스 권한을 부여하려면, 먼저 포드가 액세스해야 하는 파라미터에 대한 액세스를 제한하는 정책을 생성합니다. 그런 다음 서비스 계정용 IAM 역할을 생성하고 정책을 연결합니다. IAM 정책을 사용하여 Systems Manager 파라미터에 대한 액세스를 제한하는 것에 대한 자세한 내용은 IAM 정책을 사용하여 Systems Manager 파라미터에 대한 액세스 제한 섹션을 참조하세요.

> **Note**
>
> Parameter Store 파라미터를 사용할 때 `ssm:GetParameters` 권한이 정책에 필요합니다.

ASCP는 포드 자격 증명을 검색하고 IAM 역할에 대한 자격 증명을 교환합니다. ASCP는 포드의 IAM 역할을 가정하여 사용자가 권한을 부여한 파라미터에 대한 액세스 권한을 부여합니다. 다른 컨테이너는 IAM 역할과 연결하지 않는 한 파라미터에 액세스할 수 없습니다.

2단계: Amazon EKS에 파라미터 탑재

Amazon EKS에서 파일 시스템의 파일인 것처럼 파라미터를 표시하려면 파라미터에 대한 정보 및 Amazon EKS 포드에 탑재하는 방법이 포함된 `SecretProviderClass` YAML 파일을 생성합니다.

이 `SecretProviderClass`는 참조하는 Amazon EKS 포드와 동일한 네임스페이스에 있어야 합니다.

`SecretProviderClass`

`SecretProviderClass` YAML의 형식은 다음과 같습니다.

```
apiVersion: secrets-store.csi.x-k8s.io/v1alpha1
kind: SecretProviderClass
metadata:
   name: <NAME>
spec:
  provider: aws
  parameters:
```

파라미터

탑재 요청의 세부 정보를 포함합니다.

객체

탑재할 파라미터의 YAML 선언을 포함하는 문자열입니다. YAML 다중 행 문자열 또는 파이프(|) 문자를 사용하는 것이 좋습니다.

objectName

파라미터의 표시 이름입니다. 이 이름은 사용자가 `objectAlias`를 지정하지 않는 한 Amazon EKS 포드에서 파라미터의 파일 이름이 됩니다. Parameter Store의 경우 파라미터의 `Name`이어야 하며, 전체 Amazon 리소스 이름(ARN)일 수 없습니다.

jmesPath

(선택 사항) Amazon EKS에 탑재할 파일에 대한 JSON 인코딩 파라미터의 키 맵입니다. 다음 예제에서는 JSON 인코딩 파라미터의 형태를 보여줍니다.

```
{
    "username" : "myusername",
    "password" : "mypassword"
}
```

키는 username 및 password입니다. username에 연결된 값은 myusername이며, password에 연결된 값은 mypassword입니다.

경로

파라미터의 키입니다.

objectAlias

Amazon EKS 포드에 탑재할 파일 이름입니다.

objectType

Parameter Store의 경우 이 필드는 필수 사항입니다. ssmparameter를 사용합니다.

objectAlias

(선택 사항) Amazon EKS 포드에 있는 파라미터의 파일 이름입니다. 이 필드를 지정하지 않은 경우 objectName이 파일 이름으로 표시됩니다.

objectVersion

(선택 사항) 파라미터의 버전 번호입니다. 파라미터를 업데이트할 때마다 업데이트해야 하므로 이 필드를 사용하지 않는 것이 좋습니다. 기본적으로 가장 최신 버전이 사용됩니다. Parameter Store 파라미터의 경우, objectVersion 또는 objectVersionLabel 중 하나를 사용할 수 있지만 둘 다 사용할 수는 없습니다.

objectVersionLabel

(선택 사항) 버전에 대한 파라미터 레이블입니다. 기본값은 가장 최근 버전입니다. Parameter Store 파라미터의 경우, objectVersion 또는 objectVersionLabel 중 하나를 사용할 수 있지만 둘 다 사용할 수는 없습니다.

region

(선택 사항) 파라미터의 AWS 리전입니다. 이 필드를 사용하지 않는 경우 ASCP는 노드의 주석에서 지역을 조회합니다. 이 조회는 탑재 요청에 오버헤드를 추가하므로 많은 수의 포드를 사용하는 클러스터에 리전을 제공하는 것이 좋습니다.

pathTranslation

(선택 사항) 파일 이름(objectName또는objectAlias)에 Linux의 슬래시(/)와 같은 경로 구분 문자가 포함된 경우 사용할 단일 대체 문자입니다. 파라미터 이름에 경로 구분 기호가 포함된 경우 ASCP는 해당 이름으로 탑재된 파일을 생성할 수 없습니다. 대신 이 필드에 경로 구분 문자를 입력하여 다른 문자로 바꿀 수 있습니다. 이 필드를 사용하지 않는 경우 기본값은 밑줄(_)이므로 예를 들어 My/Path/Parameter은 My_Path_Parameter으로 탑재됩니다.

문자 대체를 방지하려면 `False` 문자열을 입력합니다.

예

다음 예제 구성은 Parameter Store 파라미터 리소스가 포함된 `SecretProviderClass`를 보여줍니다.

```
apiVersion: secrets-store.csi.x-k8s.io/v1alpha1
kind: SecretProviderClass
metadata:
    name: aws-secrets
spec:
    provider: aws
    parameters:
        objects: |
            - objectName: "MyParameter"
              objectType: "ssmparameter"
```

3단계: 배포 YAML 업데이트

배포 YAML을 업데이트하여 `secrets-store.csi.k8s.io` 드라이버를 사용하고 이전 단계에서 생성된 `SecretProviderClass` 리소스를 참조합니다. 이렇게 하면 클러스터에서 보안 암호 스토어 CSI 드라이버를 사용합니다.

아래는 이름이 `aws-secrets`인 `SecretProviderClass`를 사용하는 샘플 배포 YAML입니다.

```
volumes:
  - name: secrets-store-inline
    csi:
      driver: secrets-store.csi.k8s.io
      readOnly: true
      volumeAttributes:
        secretProviderClass: "aws-secrets"
```

자습서: Amazon EKS 포드에서 파라미터 생성 및 탑재

이 자습서에서는 Parameter Store에서 예제 파라미터를 만든 후 Amazon EKS 포드에 파라미터를 탑재하고 배포합니다.

시작하기 전에 ASCP를 설치합니다. 자세한 내용은 the section called “ASCP 설치” 단원을 참조하십시오.

보안 암호 생성 및 탑재

1. AWS 리전 및 클러스터 이름을 쉘 변수로 설정하므로 bash 명령에서 사용할 수 있습니다. *region*에서 Amazon EKS 클러스터가 실행되는 AWS 리전을 입력합니다. *clustername*에 클러스터의 이름을 입력합니다.

```
REGION=region
CLUSTERNAME=clustername
```

2. 테스트 파라미터를 생성합니다.

```
aws ssm put-parameter --name "MyParameter" --value "EKS parameter" --type String --
region "$REGION"
```

3. 이전 단계에서 생성한 파라미터에 대한 액세스를 제한하는 포드에 대한 리소스 정책을 생성합니다. *parameter-arn*의 경우 파라미터의 ARN을 사용합니다. 셸 변수에 정책 ARN을 저장합니다. 파라미터 ARN을 검색하려면 `get-parameter`를 사용합니다.

```
POLICY_ARN=$(aws --region "$REGION" --query Policy.Arn --output text iam create-
policy --policy-name nginx-parameter-deployment-policy --policy-document '{
    "Version": "2012-10-17",
    "Statement": [ {
        "Effect": "Allow",
        "Action": ["ssm:GetParameter", "ssm:GetParameters"],
        "Resource": ["parameter-arn"]
    } ]
}')
```

4. 아직 없는 경우 클러스터에 대한 IAM OpenID Connect(OIDC) 공급자를 생성합니다. 자세한 내용은 클러스터에 사용할 IAM OIDC 공급자 생성을 참조하세요.

```
eksctl utils associate-iam-oidc-provider --region="$REGION" --
cluster="$CLUSTERNAME" --approve # Only run this once
```

5. 포드에서 사용하는 서비스 계정을 만들고 3단계에서 생성한 리소스 정책을 해당 서비스 계정에 연결합니다. 이 자습서에서는 서비스 계정 이름에 nginx-deployment-sa를 사용합니다. 자세한 내용은 서비스 계정에 대한 IAM 역할 생성을 참조하세요.

```
eksctl create iamserviceaccount --name nginx-deployment-sa --region="$REGION" --
cluster "$CLUSTERNAME" --attach-policy-arn "$POLICY_ARN" --approve --override-
existing-serviceaccounts
```

6. SecretProviderClass를 생성하여 포드에 탑재할 파라미터를 지정합니다. 다음 명령은 이름이 ExampleSecretProviderClass.yaml인 SecretProviderClass 파일의 파일 위치를 사용합니다. 자체 SecretProviderClass 생성에 대한 자세한 내용은 the section called "SecretProviderClass" 섹션을 참조하세요.

```
kubectl apply -f ./ExampleSecretProviderClass.yaml
```

7. 포드를 배포합니다. 다음 명령은 이름이 ExampleDeployment.yaml인 배포 파일을 사용합니다. 자체 SecretProviderClass 생성에 대한 자세한 내용은 the section called "3단계: 배포 YAML 업데이트" 섹션을 참조하세요.

```
kubectl apply -f ./ExampleDeployment.yaml
```

8. 파라미터가 제대로 탑재되었는지 확인하려면 다음 명령을 사용하여 파라미터 값이 나타나는지 확인합니다.

```
kubectl exec -it $(kubectl get pods | awk '/nginx-deployment/{print $1}' | head -1)
 cat /mnt/secrets-store/MyParameter; echo
```

파라미터 값이 나타납니다.

```
"EKS parameter"
```

문제 해결

포드 배포를 설명하여 대부분의 오류를 볼 수 있습니다.

컨테이너에 대한 오류 메시지 확인

1. 다음 명령을 사용하여 포드 이름 목록을 가져옵니다. 기본 네임스페이스를 사용하지 않는 경우에는 -n <NAMESPACE>를 사용합니다.

```
kubectl get pods
```

2. 포드를 설명하기 위해 다음 명령에서 *pod-id*에 대해 이전 단계에서 찾은 포드의 포드 ID를 사용합니다. 기본 네임스페이스를 사용하지 않는 경우에는 -n <NAMESPACE>를 사용합니다.

```
kubectl describe pod/pod-id
```

ASCP에 대한 오류를 확인하려면

- 공급자 로그에서 자세한 정보를 찾으려면 다음 명령에서 *pod-id*에 대해 csi-secrets-store-provider-aws 포드의 ID를 사용합니다.

```
kubectl -n kube-system get pods
kubectl -n kube-system logs pod/pod-id
```

Parameter Store 활동 감사 및 로깅

AWS CloudTrail에서는 AWS Systems Manager 콘솔, AWS Command Line Interface(AWS CLI) 및 Systems Manager SDK에서 생성된 API 호출을 캡처합니다. 이 정보는 CloudTrail 콘솔 또는 Amazon Simple Storage Service(Amazon S3) 버킷에서 볼 수 있습니다. 계정에 대한 모든 CloudTrail 로그는 하나의 버킷을 사용합니다. Systems Manager 활동의 CloudTrail 로그 보기 및 사용에 대한 자세한 내용은 AWS CloudTrail을 사용하여 AWS Systems ManagerAPI 호출을 로깅 섹션을 참조하세요. Systems Manager에 대한 자세한 감사 및 로깅 옵션은 AWS Systems Manager 모니터링 섹션을 참조하세요.

Parameter Store 문제 해결

다음 정보를 사용하면 AWS Systems Manager의 기능인 Parameter Store 관련 문제를 해결하는 데 도움이 됩니다.

`aws:ec2:image` 파라미터 생성 문제 해결

다음 정보를 사용하면 `aws:ec2:image` 데이터 유형 파라미터 생성과 관련된 문제를 해결하는 데 도움이 됩니다.

인스턴스 생성 권한 없음

문제: `aws:ec2:image` 파라미터를 사용하여 인스턴스를 생성하려고 하지만 "이 작업을 수행할 권한이 없습니다" 라는 오류 메시지가 표시됩니다.

- 해결 방법: 파라미터 값을 사용하여 EC2 인스턴스를 생성하는 데 필요한 모든 권한(예: `ec2:RunInstances`, `ec2:DescribeImages`, `ssm:GetParameter` 등에 대한 권한)이 없습니다. 조직의 관리자 권한을 가진 사용자에게 문의하여 필요한 권한을 요청합니다.

EventBridge에서 “리소스를 설명할 수 없음(Unable to Describe Resource)”라는 실패 메시지를 보고합니다.

문제: aws:ec2:image 파라미터를 생성하는 명령을 실행했지만 파라미터를 생성하지 못했습니다. Amazon EventBridge로부터 “리소스를 설명할 수 없음(Unable to Describe Resource)”라는 예외를 보고하는 알림을 받습니다.

해결 방법: 이러한 메시지를 받는 이유는 다음 경우일 수 있습니다.

- ec2:DescribeImages API 작업에 필요한 모든 권한이 없거나 파라미터에서 참조된 특정 이미지에 액세스할 수 있는 권한이 없습니다. 조직의 관리자 권한을 가진 사용자에게 문의하여 필요한 권한을 요청합니다.
- 파라미터 값으로 입력한 Amazon Machine Image(AMI) ID가 유효하지 않습니다. 현재 작업 중인 AWS 리전 및 계정에서 사용할 수 있는 AMI ID를 입력했는지 확인합니다.

새 **aws:ec2:image** 파라미터를 사용할 수 없습니다.

문제: aws:ec2:image 파라미터를 생성하는 명령을 실행했고 버전 번호가 보고되었지만 파라미터를 사용할 수 없습니다.

- 해결 방법: aws:ec2:image 데이터 유형을 사용하는 파라미터를 생성하기 위해 명령을 실행하면 파라미터에 대한 버전 번호는 즉시 생성되지만 파라미터를 사용하기 전에 파라미터 형식을 검증해야 합니다. 이 프로세스는 몇 분 이상 걸릴 수 있습니다. 파라미터 생성 및 검증 프로세스를 모니터링하려면 다음을 수행합니다.
  - EventBridge를 사용하여 파라미터 작업의 create 및 update에 대한 알림을 받습니다. 이러한 알림은 파라미터 작업의 성공 여부를 보고합니다. EventBridge에서 Parameter Store 이벤트 구독에 대한 자세한 내용은 Parameter Store 이벤트 기반 알림 설정 또는 작업 트리거 섹션을 참조하세요.
  - Systems Manager 콘솔의 Parameter Store 섹션에서 파라미터 목록을 주기적으로 새로 고쳐 새 파라미터 세부 정보 또는 업데이트된 파라미터 세부 정보를 검색합니다.
  - GetParameter 명령을 사용하여 새 파라미터 또는 업데이트된 파라미터를 확인합니다. 예를 들어 AWS Command Line Interface(AWS CLI)를 사용합니다.

    ```
    aws ssm get-parameter name MyParameter
    ```

    새 파라미터의 경우, 파라미터를 검증할 때까지 ParameterNotFound 메시지가 반환됩니다. 업데이트 중인 기존 파라미터의 경우 파라미터를 검증할 때까지 새 버전에 대한 정보가 포함되지 않습니다.

검증 프로세스가 완료되기 전에 파라미터를 다시 생성하거나 업데이트하려고 하면 시스템은 검증이 아직 진행 중에 있음을 보고합니다. 파라미터가 생성되거나 업데이트되지 않으면 첫 시도 시점에서 5분이 지난 후에 다시 시도할 수 있습니다.

AWS Systems Manager 변경 관리

AWS Systems Manager는 AWS 리소스 변경을 위해 다음과 같은 기능을 제공합니다.

주제

AWS Systems Manager Change Manager

AWS Systems Manager의 기능인 Change Manager는 애플리케이션 구성 및 인프라에 대한 운영 변경을 요청, 승인, 구현 및 보고하기 위한 엔터프라이즈 변경 관리 프레임워크입니다. 하나의 위임된 관리자 계정에서 AWS Organizations를 사용하면 AWS 리전에 있는 여러 AWS 계정의 변경사항을 관리할 수 있습니다. 또는 로컬 계정을 사용하여 하나의 AWS 계정에 대한 변경 사항을 관리할 수 있습니다. AWS 리소스와 온프레미스 리소스 모두에 대한 변경 사항을 관리하려면 Change Manager를 사용합니다. Change Manager를 시작하려면 Systems Manager 콘솔을 엽니다. 탐색 창에서 Change Manager를 선택합니다.

Change Manager를 사용하면 사전 승인된 변경 템플릿을 사용하여 리소스에 대한 변경 프로세스를 자동화하고 운영 변경 시 의도하지 않은 결과를 방지할 수 있습니다. 각 변경 템플릿은 다음을 지정합니다.

- 사용자가 변경 요청을 생성할 때 선택할 수 있는 하나 이상의 Automation 실행서. 리소스의 변경 사항은 Automation 실행서에 정의되어 있습니다. 생성하는 변경 템플릿에 사용자 정의 실행서나 AWS 관리형 실행서를 포함할 수 있습니다. 사용자가 변경 요청을 생성할 때 요청에 포함할 사용 가능한 실행서 중 하나를 선택할 수 있습니다. 또한 요청을 수행하는 사용자가 변경 요청에서 실행서를 지정할 수 있도록 하는 변경 템플릿을 생성할 수 있습니다.
- 해당 변경 템플릿을 사용하여 작성된 변경 요청을 검토해야 하는 계정의 사용자입니다.
- 할당된 승인자에게 변경 요청을 검토할 준비가 되었음을 알리는 데 사용되는 Amazon Simple Notification Service(Amazon SNS) 주제.
- 실행서 워크플로를 모니터링하는 데 사용되는 Amazon CloudWatch 경보.
- 변경 템플릿을 사용하여 생성된 변경 요청의 상태 변경에 대한 알림을 보내는 데 사용되는 Amazon SNS 주제.

- 변경 템플릿을 분류하고 필터링하는 데 사용할 변경 템플릿에 적용할 태그.
- 변경 템플릿에서 생성된 변경 요청을 승인 단계 없이 실행할 수 있는지 여부(자동 승인된 요청).

Systems Manager의 또 다른 기능인 Change Calendar과의 통합을 통해 Change Manager는 중요한 비즈니스 이벤트와 일정 충돌을 피하면서 변경 사항을 안전하게 구현할 수 있습니다. 또한 Change Manager가 AWS Organizations 및 AWS IAM Identity Center과 통합되어 기존 자격 증명 관리 시스템을 사용하여 단일 계정에서 조직 전체의 변경 사항을 관리할 수 있습니다. Change Manager에서 변경 진행 상황을 모니터링하고 조직 전체의 운영 변경 사항을 감사하여 가시성과 책임성을 개선할 수 있습니다.

Change Manager는 지속적 통합(CI) 관행과 지속적 전달(CD) 방법론의 안전 제어를 보완합니다. Change Manager는 예외나 승인이 필요한 경우를 제외하고 CI/CD 파이프라인과 같은 자동화된 릴리스 프로세스의 일부로 이루어진 변경을 위한 것이 아닙니다.

Change Manager 작동 방식

표준 또는 긴급 운영 변경의 필요성이 확인되면 조직의 누군가가 조직 또는 계정에서 사용하기 위해 생성된 변경 템플릿 중 하나를 기반으로 변경 요청을 생성합니다.

요청된 변경에 수동 승인이 필요한 경우 Change Manager는 Amazon SNS 알림을 통해 지정된 승인자에게 변경 요청이 검토할 준비가 되었음을 알립니다. 변경 템플릿에서 변경 요청에 대한 승인자를 지정하거나 사용자가 변경 요청 자체에 승인자를 지정하도록 할 수 있습니다. 템플릿마다 다른 검토자를 지정할 수 있습니다. 예를 들어 관리형 노드에 대한 변경 요청을 승인해야 하는 사용자, 사용자 그룹 또는 AWS Identity and Access Management(IAM) 역할 하나와 데이터베이스 변경에 대한 다른 사용자, 그룹 또는 IAM 역할을 할당합니다. 변경 템플릿이 자동 승인을 허용하고 요청자의 사용자 정책에서 이를 금지하지 않는 경우 사용자는 검토 단계 없이 요청에 대해 Automation 런북을 실행하도록 선택할 수도 있습니다(변경 고정 이벤트 제외).

각 변경 템플릿에 대해 최대 5개 수준의 승인자를 추가할 수 있습니다. 예를 들어 기술 검토자가 먼저 변경 템플릿에서 생성된 변경 요청을 승인한 다음 한 명 이상의 관리자에게 두 번째 수준의 승인을 요구할 수 있습니다.

Change Manager는 AWS Systems Manager Change Calendar에 통합됩니다. 요청된 변경이 승인되면 시스템은 먼저 요청이 예약된 다른 비즈니스 활동과 충돌하는지 여부를 결정합니다. 충돌이 감지되면 Change Manager는 실행서 워크플로를 시작하기 전에 변경을 차단하거나 추가 승인을 요구할 수 있습니다. 예를 들어 팀이 예기치 않은 문제를 관리할 수 있도록 업무 시간 중에만 변경을 허용할 수 있습니다. 이 시간 외에 실행하도록 요청한 변경의 경우 변경 고정 승인자의 형태로 상위 관리 승인을 요

구할 수 있습니다. 긴급 변경의 경우 Change Manager는 변경 요청이 승인된 후 Change Calendar에서 충돌 또는 차단 이벤트를 확인하는 단계를 건너뛸 수 있습니다.

승인된 변경을 구현해야 할 때가 되면 Change Manager는 연결된 변경 요청에 지정된 Automation 실행서를 실행합니다. 실행서 워크플로가 실행될 때 승인된 변경 요청에 정의된 작업만 허용됩니다. 이 방법으로 변경 사항이 구현되는 동안 의도하지 않은 결과를 방지할 수 있습니다.

Change Manager로 실행서 워크플로가 실행될 때 수행할 수 있는 변경 사항을 제한할 수 있을 뿐만 아니라 동시성 및 오류 임계값을 제어할 수 있습니다. 사용자는 실행서 워크플로가 한 번에 실행할 수 있는 리소스 수, 변경 사항을 한 번에 실행할 수 있는 계정 수, 프로세스가 중지되고(실행서에 롤백 스크립트가 포함된 경우) 롤백되기 전에 허용할 실패 수를 선택합니다. CloudWatch 경보를 사용하여 진행 상황을 모니터링할 수도 있습니다.

실행서 워크플로가 완료된 후 변경 사항에 대한 세부 정보를 검토할 수 있습니다. 이러한 세부 정보에는 변경 요청의 이유, 사용된 변경 템플릿, 변경을 요청하고 승인한 사람, 변경이 구현된 방법이 포함됩니다.

추가 정보

AWS News Blog의 Introducing AWS Systems ManagerChange Manager

Change Manager가 운영에 주는 이점은 무엇인가요?

Change Manager의 이점은 다음과 같습니다.

- 서비스 중단 및 가동 중지 위험 감소

  Change Manager는 실행서 워크플로가 실행될 때 승인된 변경 사항만 구현되도록 하여 운영 변경을 보다 안전하게 수행할 수 있습니다. 계획되지 않은 변경 사항과 검토되지 않은 변경 사항을 차단할 수 있습니다. Change Manager를 사용하면 비용과 시간이 많이 드는 연구 및 역추적이 필요한 인적 오류로 인해 발생하는 의도치 않은 결과 유형을 방지할 수 있습니다.
- 변경 기록에 대한 자세한 감사 및 보고 받기

  Change Manager는 조직 전체의 변경 사항, 변경 의도, 변경 사항을 승인하고 구현한 사람에 대한 일관되게 보고하고 감사하는 방법을 제공합니다.
- 일정 충돌 또는 위반 방지

Change Manager는 조직의 활성 변경 일정을 기반으로 휴일 이벤트 또는 신제품 출시와 같은 일정 충돌을 감지할 수 있습니다. 실행서 워크플로가 업무 시간에만 실행되도록 허용하거나 추가 승인이 있는 경우에만 허용할 수 있습니다.

- 변화하는 비즈니스에 맞게 변경 요구 사항 조정

  업무 기간마다 다른 변경 관리 요구 사항을 구현할 수 있습니다. 예를 들어 월말 보고, 세금 기간 또는 기타 중요한 비즈니스 기간 동안 불필요한 운영 위험을 초래할 수 있는 변경 사항에 대해 변경을 차단하거나 이사 수준의 승인을 요구할 수 있습니다.

- 계정 전체의 변경 사항을 중앙에서 관리

  Organizations와의 통합을 통해 Change Manager를 사용하면 위임된 단일 관리자 계정에서 모든 조직 단위(OU)의 변경 사항을 관리할 수 있습니다. 전체 조직 또는 일부 OU에만 사용하도록 Change Manager를 설정할 수 있습니다.

Change Manager는 누가 사용해야 하나요?

Change Manager는 다음 AWS 고객 및 조직에 적합합니다.

- 모두 AWS 고객은 클라우드 또는 온프레미스 환경에 대한 운영 변경 사항의 안전성과 거버넌스를 개선하고자 합니다.
- 팀 간 협업 및 가시성을 높이고 가동 중지 시간을 방지하여 애플리케이션 가용성을 개선하며 수동 및 반복 태스크와 관련된 위험을 줄이려는 조직.
- 변경 관리를 위한 모범 사례를 준수해야 하는 조직.
- 애플리케이션 구성 및 인프라에 대한 전체 감사 가능한 변경 기록이 필요한 고객.

Change Manager의 주요 기능은 무엇입니까?

Change Manager의 주요 기능은 다음과 같습니다.

- 변경 관리 모범 사례에 대한 통합 지원

  Change Manager를 사용하면 선택 변경 관리 모범 사례를 작업에 적용할 수 있습니다. 다음 옵션을 설정하도록 선택할 수 있습니다.

  - Change Calendar에서 이벤트가 현재 제한되어 열려 있는 일정 기간 동안에만 변경이 이루어지는지 확인합니다.
  - 변경 고정 승인자의 추가 승인으로 제한된 이벤트 동안 변경을 허용합니다.

  - 모든 변경 템플릿에 대해 CloudWatch 경보를 지정해야 합니다.
  - 사용자 계정에서 생성된 모든 변경 템플릿이 변경 요청을 생성하는 데 사용되기 전에 검토 및 승인을 받아야 합니다.
- 닫힌 일정 기간 및 긴급 변경 요청에 대한 다양한 승인 경로

  Change Calendar에서 제한된 이벤트를 확인하는 옵션을 허용하고 이벤트가 완료될 때까지 승인된 변경 요청을 차단할 수 있습니다. 그러나 일정이 닫힌 경우에도 변경을 허용할 수 있는 두 번째 승인자 그룹인 변경 고정 승인자를 지정할 수도 있습니다. 또한 긴급 변경 템플릿을 생성할 수 있습니다. 긴급 변경 템플릿에서 생성된 변경 요청은 여전히 일반 승인이 필요하지만 일정 제한이 적용되지 않으며 변경 고정 승인이 필요하지 않습니다.
- 실행서 워크플로 시작 방법 및 시기 제어

  실행서 워크플로는 일정에 따라 시작하거나 승인이 완료되는 즉시 시작할 수 있습니다 (일정 제한 규칙에 따름).
- 기본 제공 알림 지원

  조직 내에서 변경 템플릿과 변경 요청을 검토하고 승인해야 하는 사용자를 지정합니다. Amazon SNS 주제를 변경 템플릿에 할당하여 해당 변경 템플릿으로 생성된 변경 요청의 상태 변경에 대한 알림을 주제 구독자에게 보냅니다.
- AWS Systems Manager Change Calendar와 통합

  Change Manager와의 통합을 통해 지정된 기간 동안 예약 변경을 제한할 수 있습니다. 예를 들어 팀이 문제를 처리할 수 있도록 비즈니스 시간에만 변경을 허용하는 정책을 만들 수 있습니다. 중요한 비즈니스 이벤트 중에는 변경을 제한할 수도 있습니다. 예를 들어 소매 비즈니스는 대규모 판매 이벤트 동안 변경을 제한할 수 있습니다. 제한된 기간 동안 추가 승인을 요구할 수도 있습니다.
- AWS IAM Identity Center 및 Active Directory 지원과 통합

  IAM Identity Center 통합을 통해 조직의 멤버는 공통 사용자 자격 증명을 기반으로 Systems Manager를 사용하여 AWS 계정에 액세스하고 리소스를 관리할 수 있습니다. IAM Identity Center를 사용하여 사용자에게 AWS의 계정에 대한 액세스 권한을 할당할 수 있습니다.

  Active Directory와의 통합을 통해 Active Directory 계정의 사용자를 Change Manager 작업을 위해 생성된 변경 템플릿에 대한 승인자로 할당할 수 있습니다.
- Amazon CloudWatch 경보와 통합

Change Manager는 CloudWatch 경보와 통합됩니다. Change Manager는 실행서 워크플로 동안 CloudWatch 경보를 수신 대기하고 알림 전송을 포함하여 경보에 대해 정의된 모든 작업을 수행합니다.

- AWS CloudTrail Lake와의 통합

  AWS CloudTrail Lake에 이벤트 데이터 스토어를 생성하면 계정 또는 조직에서 실행되는 변경 요청으로 인한 변경 사항과 관련하여 감사 가능한 정보를 볼 수 있습니다. 저장된 이벤트 정보에는 다음 세부 정보가 포함됩니다.

  - 실행된 API 작업
  - 해당 작업에 대해 포함된 요청 파라미터
  - 작업을 실행한 사용자
  - 프로세스 실행 중에 업데이트된 리소스

- AWS Organizations과 통합

  Organizations에서 제공하는 크로스 계정 기능으로 조직의 OU에서 Change Manager 작업을 관리하기 위해 위임된 관리자 계정을 사용할 수 있습니다. Organizations 관리 계정에서 위임된 관리자 계정이 될 계정을 지정할 수 있습니다. Change Manager를 사용할 수 있는 OU를 제어할 수도 있습니다.

Change Manager를 사용하는 데 비용이 듭니까?

예. Change Manager는 종량제로 요금이 부과됩니다. 사용한 만큼만 지불합니다. 자세한 내용은 AWS Systems Manager 요금을 참조하십시오.

Change Manager의 주요 구성 요소는 무엇인가요?

Change Manager 조직 또는 계정의 변경 프로세스를 관리하는 데 사용하는 구성 요소는 다음과 같습니다.

위임된 관리자 계정

조직 전체에서 Change Manager를 사용하는 경우 위임된 관리자 계정을 사용합니다. Change Manager를 포함하여 Systems Manager 전반의 운영 활동을 관리하기 위한 계정으로 지정된 AWS 계정입니다. 위임된 관리자 계정은 조직 전체의 변경 활동을 관리합니다. Change Manager에서 사용할 조직을 설정할 때 이 역할을 수행하는 계정을 지정합니다. 위임된 관리자 계정은 할당된 조직 단위

(OU)의 유일한 멤버여야 합니다. AWS 계정 하나만으로 Change Manager를 사용하는 경우 위임된 관리자 계정이 필요하지 않습니다.

> **Important**
>
> 조직 전체에서 Change Manager를 사용하는 경우 항상 위임된 관리자 계정에서 변경하는 것이 좋습니다. 조직의 다른 계정에서 변경할 수 있지만 이러한 변경 사항은 위임된 관리자 계정에서 보고되거나 볼 수 없습니다.

변경 템플릿

변경 템플릿은 필수 승인, 사용 가능한 실행서 및 변경 요청에 대한 알림 옵션과 같은 항목을 정의하는 Change Manager의 구성 설정 모음입니다.

조직 또는 계정의 사용자가 만든 변경 템플릿을 사용하려면 승인 프로세스를 거치도록 할 수 있습니다.

Change Manager는 2가지 유형의 변경 템플릿을 지원합니다. 긴급 변경 템플릿을 기반으로 하는 승인된 변경 요청의 경우 Change Calendar에 차단 이벤트가 있는 경우에도 요청된 변경을 수행할 수 있습니다. 표준 변경 템플릿을 기반으로 승인된 변경 요청의 경우 Change Calendar에 차단 이벤트가 있으면 지정된 변경 고정 이벤트 승인자로부터 추가 승인을 받지 않는 한 요청한 변경을 수행할 수 없습니다.

변경 요청

변경 요청은 AWS 또는 온프레미스 환경에서 하나 이상의 리소스를 업데이트하는 Automation 실행서를 실행하기 위한 Change Manager의 요청입니다. 변경 요청은 변경 템플릿을 사용하여 생성됩니다.

변경 요청을 생성할 때 조직 또는 계정에 있는 한 명 이상의 승인자가 요청을 검토하고 승인해야 합니다. 필요한 승인이 없으면 요청한 변경 내용을 적용하는 실행서 워크플로를 실행할 수 없습니다.

시스템에서 변경 요청은 AWS Systems Manager OpsCenter의 OpsItem 유형입니다. 단, `/aws/changerequest` 유형의 OpsItems는 OpsCenter에 표시되지 않습니다. OpsItems와 마찬가지로 변경 요청에는 다른 유형의 OpsItems와 동일한 할당량이 적용됩니다.

또한 프로그래밍 방식으로 변경 요청을 생성하기 위해 `CreateOpsItem` API 작업을 호출하지 않습니다. 대신 `StartChangeRequestExecution` API 작업을 사용합니다. 단, 바로 실행하기보다는 변경 요청을 승인받아야 하며, Change Calendar에 차단 이벤트가 없어야 워크플로가 실행되지 않습니다. 승인을 받았고 일정이 차단되지 않은 경우 또는 일정 차단 이벤트를 우회할 수 있는 권한이 부여된 경우 `StartChangeRequestExecution` 작업을 완료할 수 있습니다.

실행서 워크플로

실행서 워크플로는 클라우드 또는 온프레미스 환경에서 대상 리소스에 대해 요청된 변경을 수행하는 프로세스입니다. 각 변경 요청은 요청된 변경을 수행하는 데 사용할 단일 Automation 실행서를 지정합니다. 실행서 워크플로는 필요한 모든 승인이 부여되고 Change Calendar에 차단 이벤트가 없는 후에 발생합니다. 특정 날짜 및 시간에 변경이 예약된 경우 모든 승인이 수신되고 일정이 차단되지 않은 경우에도 예약될 때까지 실행서 워크플로가 시작되지 않습니다.

주제

- Change Manager 설정
- Change Manager 작업
- Change Manager 활동 감사 및 로깅
- Change Manager 문제 해결

Change Manager 설정

AWS Systems Manager의 기능인 Change Manager를 사용하여 AWS Organizations에 구성된 전체 조직 또는 단일 AWS 계정에 대한 변경 사항을 관리할 수 있습니다.

조직에서 Change Manager를 사용하는 경우 주제 조직용 Change Manager 설정(관리 계정)으로 시작한 다음 Change Manager 옵션 및 모범 사례 구성으로 진행합니다.

하나의 계정으로 Change Manager를 사용하는 경우 바로 Change Manager 옵션 및 모범 사례 구성으로 진행합니다.

> **Note**
>
> 단일 계정으로 Change Manager를 사용하기 시작했지만 나중에 해당 계정이 Change Manager가 허용되는 조직 단위에 추가되면 단일 계정 설정이 무시됩니다.

주제

- 조직용 Change Manager 설정(관리 계정)
- Change Manager 옵션 및 모범 사례 구성
- Change Manager에 대한 역할 및 권한 구성
- 자동 승인 실행서 워크플로에 대한 액세스 제어

조직용 Change Manager 설정(관리 계정)

이 항목의 태스크는 AWS Organizations에 설정된 조직과 함께 AWS Systems Manager의 기능인 Change Manager를 사용하는 경우에 적용됩니다. 단일 AWS 계정로만 Change Manager를 사용하려면 주제 Change Manager 옵션 및 모범 사례 구성으로 건너뜁니다.

Organizations에서 관리 계정 역할을 하는 AWS 계정로 이 섹션의 태스크를 수행합니다. 관리 계정 및 기타 Organizations 개념에 대한 자세한 내용은 AWS Organizations 용어 및 개념을 참조하세요.

계속하기 전에 Organizations를 설정하고 계정을 관리 계정으로 지정해야 하는 경우 AWS Organizations User Guide의 Creating and managing an organization을 참조하세요.

> **Note**
>
> 이 설정 프로세스는 다음 AWS 리전에서 수행할 수 없습니다.
>
> - 유럽(밀라노)(eu-south-1)
> - 중동(바레인)(me-south-1)
> - 아프리카(케이프타운)(af-south-1)
> - 아시아 태평양(홍콩)(ap-east-1)
>
> 이 절차를 수행하려면 관리 계정의 다른 리전에서 작업하고 있는지 확인합니다.

설치 절차를 진행하는 동안 AWS Systems Manager의 기능인 Quick Setup에서 다음과 같은 주요 태스크를수행합니다.

- 태스크 1 - 조직을 위해 위임된 관리자 계정 등록

  Change Manager를 사용하여 수행되는 변경 관련 태스크는 위임된 관리자 계정으로 지정한 멤버 계정 중 하나에서 관리됩니다. Change Manager에 등록한 위임된 관리자 계정은 모든 Systems Manager 작업에 대한 위임된 관리자 계정이 됩니다. (다른 AWS 서비스에 대한 관리자 계정을 위임했을 수 있습니다.) 관리 계정과 다른 Change Manager에 대한 위임된 관리자 계정은 변경 템플릿, 변경 요청 및 각각에 대한 승인을 포함하여 조직 전체의 변경 활동을 관리합니다. 위임된 관리자 계정에서 Change Manager 작업에 대한 다른 구성 옵션도 지정합니다.

> **Important**
>
> 위임된 관리자 계정은 Organizations에서 할당된 조직 단위(OU)의 유일한 멤버여야 합니다.

- 태스크 2: Change Manager 작업에 사용할 변경 요청자 역할 또는 사용자 정의 직무에 대한 실행서 액세스 정책 정의 및 지정

  Change Manager에서 변경 요청을 생성하려면 멤버 계정의 사용자에게 AWS Identity and Access Management(IAM) 권한이 부여되어 사용자가 사용할 수 있도록 선택한 Automation 실행서와 변경 템플릿에만 액세스할 수 있어야 합니다.

  > **Note**
  >
  > 사용자가 변경 요청을 생성할 때 먼저 변경 템플릿을 선택합니다. 이 변경 템플릿을 사용하면 여러 실행서를 사용할 수 있지만 사용자는 각 변경 요청에 대해 하나의 실행서만 선택할 수 있습니다. 사용자가 요청에 사용 가능한 실행서를 포함하도록 변경 템플릿을 구성할 수도 있습니다.

  Change Manager는 필요한 권한을 부여하기 위해 IAM에서도 사용되는 직무 개념을 사용합니다. 그러나 IAM의 직무에 대한 AWS 관리형 정책과 달리 Change Manager 직무의 이름과 해당 직무에 대한 IAM 권한을 모두 지정합니다.

  직무를 구성할 때 사용자 정의 정책을 생성하고 변경 관리 태스크를 수행하는 데 필요한 권한만 제공하는 것이 좋습니다. 예를 들어, 정의한 직무를 기반으로 특정 런북 집합으로 사용자를 제한하는 권한을 지정할 수 있습니다.

  예를 들어 이름이 `DBAdmin`인 직무를 생성할 수 있습니다. 이 직무의 경우 `AWS-CreateDynamoDbBackup` 및 `AWSConfigRemediation-DeleteDynamoDbTable`과 같은 Amazon DynamoDB 데이터베이스와 관련된 실행서에 필요한 권한만 부여할 수 있습니다.

  또 다른 예로 `AWS-ConfigureS3BucketLogging` 및 `AWSConfigRemediation-ConfigureS3BucketPublicAccessBlock`과 같은 Amazon Simple Storage Service(Amazon S3) 버킷과 관련된 실행서 작업에 필요한 권한만 일부 사용자에게 부여할 수 있습니다.

  Change Manager에 대한 Quick Setup의 구성 프로세스에서는 생성한 관리 역할에 적용할 수 있는 전체 Systems Manager 관리 권한 집합도 만듭니다.

배포하는 각 Change Manager Quick Setup 구성은 선택한 조직 단위에서 Change Manager 템플릿 및 Automation 실행서를 실행할 수 있는 권한이 있는 위임된 관리자 계정에 직무를 생성합니다. Change Manager에 대해 최대 15개의 Quick Setup 구성을 생성할 수 있습니다.

- 작업 3: 조직에서 Change Manager에 사용할 멤버 계정 선택

  Organizations에 설정된 모든 조직 단위의 모든 멤버 계정과 해당 조직이 운영하는 모든 AWS 리전에 Change Manager를 사용할 수 있습니다. 원하는 경우 일부 조직 단위에만 Change Manager를 사용할 수 있습니다.

> **Important**
>
> 이 절차를 시작하기 전에 해당 단계를 읽고 구성 선택 사항과 부여할 권한을 이해하는 것이 좋습니다. 특히 생성할 사용자 정의 직무와 각 직무에 할당할 권한을 계획합니다. 이렇게 하면 나중에 생성한 직무 정책을 개별 사용자, 사용자 그룹 또는 IAM 역할에 연결할 때 원하는 권한만 부여됩니다.
>
> 가장 좋은 방법은 AWS 계정 관리자 로그인을 사용하여 위임된 관리자 계정을 설정하는 것부터 시작하는 것입니다. 그런 다음 변경 템플릿을 생성하고 각 템플릿에서 사용하는 실행서를 식별한 후 직무와 해당 권한을 구성합니다.

조직에 사용하도록 Change Manager를 설정하려면 Systems Manager 콘솔의 Quick Setup 영역에서 다음 태스크를 수행합니다.

조직에 대해 생성하려는 직무마다 이 태스크를 반복합니다. 생성하는 각 직무는 서로 다른 조직 단위 집합에 대한 권한을 가질 수 있습니다.

Organizations 관리 계정에서 Change Manager에 대한 조직을 설정하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Quick Setup를 선택합니다.
3. Change Manager 카드에서 Create(생성)를 선택합니다.
4. 위임된 관리자 계정의 경우 Change Manager에 변경 템플릿, 변경 요청 및 실행서 워크플로 관리에 사용할 AWS 계정의 ID를 입력합니다.

   이전에 Systems Manager에 대해 위임된 관리자 계정을 지정한 경우 이 필드에 해당 ID가 이미 보고됩니다.

> **Important**
>
> 위임된 관리자 계정은 Organizations에서 할당된 조직 단위(OU)의 유일한 멤버여야 합니다.
> 등록한 위임된 관리자 계정이 나중에 해당 역할에서 등록 취소되는 경우 시스템은 동시에 Systems Manager 작업을 관리할 수 있는 권한을 제거합니다. Quick Setup으로 돌아가서 다른 위임된 관리자 계정을 지정하고 모든 직무와 권한을 다시 지정해야 합니다.
> 조직 전체에서 Change Manager를 사용하는 경우 항상 위임된 관리자 계정에서 변경하는 것이 좋습니다. 조직의 다른 계정에서 변경할 수 있지만 이러한 변경 사항은 위임된 관리자 계정에서 보고되거나 볼 수 없습니다.

5. [요청 및 변경 권한(Permissions to request and make changes)] 섹션에서 다음을 수행합니다.

> **Note**
>
> 생성하는 각 배포 구성은 하나의 직무에 대한 권한 정책을 제공합니다. 작업에 사용할 변경 템플릿을 만든 경우 나중에 Quick Setup으로 돌아가서 더 많은 직무를 생성할 수 있습니다.

관리 역할을 생성하려면 - 모든 AWS 작업에 대한 IAM 권한이 있는 관리자 직무의 경우 다음을 수행합니다.

> **Important**
>
> 사용자에게 전체 관리 권한을 부여하는 것은 사용자의 역할에 전체 Systems Manager 액세스가 필요한 경우에만 수행해야 합니다. Systems Manager 액세스의 보안 고려 사항에 대한 중요한 내용은 AWS Systems Manager의 I자격 증명 및 액세스 관리 및 Systems Manager의 보안 모범 사례 섹션을 참조하세요.

1. [직무(Job function)]에 이 역할과 권한을 식별하는 이름을 입력합니다(예: **`MyAWSAdmin`**).
2. [역할 및 권한 옵션(Role and permissions option)]에서 [관리자 권한(Administrator permissions)]을 선택합니다.

다른 직무를 생성하려면 - 비관리 역할을 생성하려면 다음을 수행합니다.

1. [직무(Job function)]에 이 역할을 식별하고 해당 권한을 제안하는 이름을 입력합니다. 선택하는 이름은 권한을 제공할 실행서의 범위를 나타내야 합니다(예: `DBAdmin` 또는 `S3Admin`).
2. [역할 및 권한 옵션(Role and permissions option)]에서 [사용자 정의 권한(Custom permissions)]을 선택합니다.
3. [권한 정책 편집기(Permissions policy editor)]에 이 직무에 부여할 IAM 권한을 JSON 형식으로 입력합니다.

> **Tip**
>
> IAM 정책 편집기를 사용하여 정책을 구성한 다음 정책 JSON을 [권한 정책(Permissions policy)] 필드에 붙여넣는 것이 좋습니다.

샘플 정책: DynamoDB 데이터베이스 관리

예를 들어 직무가 액세스해야 하는 Systems Manager 문서(SSM 문서) 작업에 대한 권한을 제공하는 정책 콘텐츠로 시작할 수 있습니다. 다음은 미국 동부(오하이오) 리전(`us-east-2`)의 샘플 AWS 계정 `123456789012`에서 생성된 2개의 변경 템플릿 및 DynamoDB 데이터베이스와 관련된 모든 AWS 관리형 Automation 실행서에 대한 액세스 권한을 부여하는 샘플 정책 콘텐츠입니다.

이 정책에는 Change Calendar에서 변경 요청을 생성하는 데 필요한 StartChangeRequestExecution 작업에 대한 권한도 포함되어 있습니다.

> **Note**
>
> 이 예는 포괄적이지 않습니다. 데이터베이스, 노드 등의 다른 AWS 리소스를 사용하려면 추가 권한이 필요할 수 있습니다.

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

IAM 정책에 대한 자세한 내용은 IAM User Guide의 Access management for AWS resources 및 Creating IAM policies를 참조하세요.

6. [대상(Targets)] 섹션에서 생성 중인 직무에 대한 권한을 전체 조직에 부여할지 아니면 일부 조직 단위에만 부여할지 선택합니다.

   [전체 조직(Entire organization)]을 선택하는 경우 9단계로 진행합니다.

[사용자 정의(Custom)]를 선택하는 경우 8단계로 진행합니다.

7. [대상 OU(Target OUs)] 섹션에서 Change Manager에서 사용할 조직 단위의 확인란을 선택합니다.
8. 생성(Create)을 선택합니다.

시스템이 조직에 대한 Change Manager 설정을 완료하면 배포 요약이 표시됩니다. 이 요약 정보에는 구성한 직무에 대해 생성된 역할의 이름이 포함됩니다. 예를 들면 `AWS-QuickSetup-SSMChangeMgr-DBAdminInvocationRole`입니다.

> **Note**
>
> Quick Setup은 AWS CloudFormation StackSets를 사용하여 구성을 배포합니다. AWS CloudFormation 콘솔에서 완료된 배포 구성에 대한 정보를 볼 수도 있습니다. StackSets 대한 자세한 내용은 AWS CloudFormation User Guide의 Working with AWS CloudFormation StackSets를 참조하세요.

다음 단계는 추가 Change Manager 옵션을 구성하는 것입니다. 위임된 관리자 계정 또는 Change Manager에 사용하도록 허용한 조직 단위의 모든 계정에서 이 태스크를 완료할 수 있습니다. 사용자 자격 증명 관리 옵션 선택, 변경 템플릿 및 변경 요청을 검토하고 승인 또는 거부할 수 있는 사용자 지정, 조직에 허용할 모범 사례 옵션 선택 등의 옵션을 구성합니다. 자세한 내용은 Change Manager 옵션 및 모범 사례 구성 섹션을 참조하세요.

Change Manager 옵션 및 모범 사례 구성

AWS Systems Manager의 기능인 Change Manager를 조직 전체에서 사용하는지 아니면 단일 AWS 계정에서 사용하는지에 따라 이 섹션의 태스크를 수행해야 합니다.

조직에 Change Manager를 사용하는 경우 위임된 관리자 계정 또는 Change Manager에 사용하도록 허용한 조직 단위의 모든 계정에서 다음 태스크를 수행할 수 있습니다.

주제

- 태스크 1: Change Manager 사용자 자격 증명 관리 및 템플릿 검토자 구성
- 태스크 2: Change Manager 변경 고정 이벤트 승인자 및 모범 사례 구성
- Change Manager 알림에 대한 Amazon SNS 주제 구성

태스크 1: Change Manager 사용자 자격 증명 관리 및 템플릿 검토자 구성

Change Manager에 처음 액세스할 때 이 절차의 태스크를 수행합니다. 나중에 Change Manager로 돌아가서 [설정(Settings)] 탭에서 [편집(Edit)]을 선택하여 이러한 구성 설정을 업데이트할 수 있습니다.

Change Manager 사용자 자격 증명 관리 및 템플릿 검토자를 구성하려면

1. AWS Management Console에 로그인합니다.

   조직에 Change Manager를 사용하는 경우 위임된 관리자 계정의 자격 증명을 사용하여 로그인합니다. 사용자에게 Change Manager 설정을 업데이트하는 데 필요한 AWS Identity and Access Management(IAM) 권한이 있어야 합니다.
2. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
3. 탐색 창에서 Change Manager를 선택합니다.
4. 서비스 홈 페이지에서 사용 가능한 옵션에 따라 다음 중 하나를 수행합니다.

   - AWS Organizations에 Change Manager를 사용하는 경우 위임된 계정 설정(Set up delegated account)을 선택합니다.
   - 단일 AWS 계정에 Change Manager를 사용하는 경우 Change Manager 설정(Set up Change Manager)을 선택합니다.

     -또는-

     샘플 변경 요청 생성(Create sample change request), 건너뛰기(Skip)를 선택한 다음 설정(Settings) 탭을 선택합니다.
5. 사용자 자격 증명 관리(User identity management)에서 다음 중 하나를 선택합니다.

   - AWS Identity and Access Management(IAM) - 기존 사용자, 그룹 및 역할을 사용하여 Change Manager에서 요청, 요청 승인 및 기타 작업을 수행하는 사용자를 식별합니다.
   - AWS IAM Identity Center(IAM Identity Center) - IAM Identity Center가 자격 증명을 생성 및 관리하도록 허용하거나 Change Manager에서 작업을 수행하는 사용자를 식별하기 위해 기존 자격 증명 소스에 연결합니다.
6. Template reviewer notification(템플릿 검토자 알림) 섹션에서 템플릿 검토자에게 새 변경 템플릿 또는 변경 템플릿 버전을 검토할 준비가 되었음을 알리는 데 사용할 Amazon Simple Notification Service(SNS) 주제를 지정합니다. 선택한 Amazon SNS 주제가 템플릿 검토자에게 알림을 보내도록 구성되어 있는지 확인합니다.

변경 템플릿 검토자 알림을 위한 Amazon SNS 주제 생성 및 구성에 대한 자세한 내용은 Change Manager 알림에 대한 Amazon SNS 주제 구성 섹션을 참조하세요.

1. 템플릿 검토자 알림에 대한 Amazon SNS 주제를 지정하려면 다음 중 하나를 선택합니다.
   - SNS Amazon 리소스 이름(ARN) 입력(Enter an SNS Amazon Resource Name (ARN)) - 주제 ARN(Topic ARN)에 기존 Amazon SNS 주제의 ARN을 입력합니다. 이 주제는 조직의 모든 계정에 있을 수 있습니다.
   - 기존 SNS 주제 선택(Select an existing SNS topic) - 대상 알림 주제(Target notification topic)에서 현재 AWS 계정의 기존 Amazon SNS 주제의 ARN을 선택합니다. (현재 AWS 계정 및 AWS 리전에서 Amazon SNS 주제를 아직 생성하지 않은 경우 이 옵션을 사용할 수 없습니다.)

   > **Note**
   >
   > 선택한 Amazon SNS 주제는 전송되는 알림과 알림이 전송되는 구독자를 지정하도록 구성되어야 합니다. Change Manager에서 알림을 보낼 수 있도록 액세스 정책도 Systems Manager에 권한을 부여해야 합니다. 자세한 내용은 Change Manager 알림에 대한 Amazon SNS 주제 구성 섹션을 참조하세요.

2. 알림 추가를 선택합니다.

7. 변경 템플릿 검토자(Change template reviewers) 섹션에서 조직 또는 계정의 사용자를 선택하여 새 변경 템플릿을 검토하거나 템플릿 버전을 변경해야 작업에 템플릿을 사용할 수 있습니다.

   변경 템플릿 검토자는 Change Manager 실행서 워크플로에 사용하기 위해 다른 사용자가 제출한 템플릿의 적합성과 보안을 확인할 책임이 있습니다.

   다음을 수행하여 변경 템플릿 검토자를 선택합니다.

   1. 추가를 선택합니다.
   2. 변경 템플릿 검토자로 할당할 각 사용자, 그룹 또는 IAM 역할의 이름 옆에 있는 확인란을 선택합니다.
   3. 승인자 추가(Add approvers)를 선택합니다.

8. 제출을 선택합니다.

이 초기 설정 프로세스를 완료한 후 태스크 2: Change Manager 변경 고정 이벤트 승인자 및 모범 사례 구성의 단계에 따라 추가 Change Manager 설정과 모범 사례를 구성합니다.

태스크 2: Change Manager 변경 고정 이벤트 승인자 및 모범 사례 구성

태스크 1: Change Manager 사용자 자격 증명 관리 및 템플릿 검토자 구성의 단계를 완료한 후 변경 고정 이벤트 중 변경 요청에 대한 추가 검토자를 지정하고 Change Manager 작업에 허용할 사용 가능한 모범 사례를 지정할 수 있습니다.

변경 고정 이벤트는 현재 변경 일정에 제한이 있음을 의미합니다(AWS Systems Manager Change Calendar의 일정 상태는 CLOSED임). 이러한 경우 변경 요청에 대한 일반 승인자 외에 또는 자동 승인을 허용하는 템플릿을 사용하여 변경 요청을 생성하는 경우 변경 고정 승인자는 이 변경 요청을 실행할 수 있는 권한을 부여해야 합니다. 그렇지 않으면 일정 상태가 다시 OPEN이 될 때까지 변경 사항이 처리되지 않습니다.

Change Manager 변경 고정 이벤트 승인자 및 모범 사례를 구성하려면

1. 탐색 창에서 Change Manager를 선택합니다.
2. 설정(Settings) 탭을 선택한 후 편집(Edit)을 선택합니다.
3. 변경 고정 이벤트 승인자(Approvers for change freeze events) 섹션에서 Change Calendar에서 사용 중인 일정이 현재 CLOSED인 경우에도 실행되도록 변경을 승인할 수 있는 조직 또는 계정의 사용자를 선택합니다.

   > **Note**
   >
   > 변경 고정 검토를 허용하려면 모범 사례(Best practices)에서 제한된 변경 이벤트에 대한 변경 일정 확인(Check Change Calendar for restricted change events) 옵션을 설정해야 합니다.

   다음을 수행하여 변경 고정 이벤트에 대한 승인자를 선택합니다.

   1. 추가를 선택합니다.
   2. 변경 고정 이벤트에 대한 승인자로 할당할 각 사용자, 그룹 또는 IAM 역할의 이름 옆에 있는 확인란을 선택합니다.
   3. 승인자 추가(Add approvers)를 선택합니다.
4. 페이지 하단 근처의 [모범 사례(Best practices)] 섹션에서 다음 각 옵션에 대해 시행하려는 모범 사례를 설정합니다.

   - 옵션: 제한된 변경 이벤트에 대한 변경 일정 확인(Check Change Calendar for restricted change events)

Change Manager가 Change Calendar에서 일정을 확인하여 변경이 예정된 이벤트에 의해 차단되지 않도록 지정하려면 먼저 [사용(Enabled)] 확인란을 선택한 다음 [변경 일정(Change Calendar)] 목록에서 제한된 일정을 확인할 일정을 선택합니다.

Change Calendar에 대한 자세한 내용은 AWS Systems Manager Change Calendar 섹션을 참조하세요.

- 옵션: 마감된 이벤트 승인자를 위한 SNS 주제(SNS topic for approvers for closed events)

  1. 다음 중 하나를 선택하여 계정에서 변경 고정 이벤트 중에 승인자에게 알림을 보내는 데 사용할 Amazon Simple Notification Service(Amazon SNS) 주제를 지정합니다. ([모범 사례(Best practices)] 위의 변경 고정 이벤트 승인자(Approvers for change freeze events) 섹션에서도 승인자를 지정해야 합니다.)

     - SNS Amazon 리소스 이름(ARN) 입력(Enter an SNS Amazon Resource Name (ARN)) - 주제 ARN(Topic ARN)에 기존 Amazon SNS 주제의 ARN을 입력합니다. 이 주제는 조직의 모든 계정에 있을 수 있습니다.
     - 기존 SNS 주제 선택(Select an existing SNS topic) - 대상 알림 주제(Target notification topic)에서 현재 AWS 계정의 기존 Amazon SNS 주제의 ARN을 선택합니다. (현재 AWS 계정 및 AWS 리전에서 Amazon SNS 주제를 아직 생성하지 않은 경우 이 옵션을 사용할 수 없습니다.)

     > **Note**
     >
     > 선택한 Amazon SNS 주제는 전송되는 알림과 알림이 전송되는 구독자를 지정하도록 구성되어야 합니다. Change Manager에서 알림을 보낼 수 있도록 액세스 정책도 Systems Manager에 권한을 부여해야 합니다. 자세한 내용은 Change Manager 알림에 대한 Amazon SNS 주제 구성 섹션을 참조하세요.

  2. 알림 추가를 선택합니다.

- 옵션: 모든 템플릿에 모니터 필요(Require monitors for all templates)

  조직 또는 계정의 모든 템플릿이 Amazon CloudWatch 경보를 지정하여 변경 작업을 모니터링하도록 하려면 Enabled(활성화됨) 확인란을 선택합니다.

- 옵션: 사용하기 전에 템플릿 검토 및 승인 필요(Require template review and approval before use)

  검토 및 승인된 템플릿을 기반으로 하지 않고 변경 요청이 생성되지 않고 실행서 워크플로가 실행되지 않도록 하려면 [사용(Enabled)] 확인란을 선택합니다.

5. Save(저장)를 선택합니다.

Change Manager 알림에 대한 Amazon SNS 주제 구성

변경 요청 및 변경 템플릿과 관련된 이벤트에 대해 Amazon Simple Notification Service(Amazon SNS) 주제에 알림을 보내도록 AWS Systems Manager의 기능인 Change Manager를 구성할 수 있습니다. 주제를 추가한 Change Manager 이벤트에 대한 알림을 받으려면 다음 태스크를 완료합니다.

주제

- 태스크 1: Amazon SNS 주제 생성 및 구독
- 태스크 2: Amazon SNS 액세스 정책 업데이트
- 태스크 3: (옵션) AWS Key Management Service 액세스 정책 업데이트

태스크 1: Amazon SNS 주제 생성 및 구독

먼저, Amazon SNS 주제를 생성하고 구독합니다. 자세한 내용은 Amazon Simple Notification Service 개발자 안내서의 Amazon SNS 주제 생성 및 Amazon SNS 주제 구독을 참조하세요.

> **Note**
>
> 알림을 수신하려면 위임된 관리자 계정과 동일한 AWS 리전 및 AWS 계정에 있는 Amazon SNS 주제의 Amazon 리소스 이름(ARN)을 지정해야 합니다.

태스크 2: Amazon SNS 액세스 정책 업데이트

다음 절차에 따라 Systems Manager가 태스크 1에서 생성한 Amazon SNS 주제에 Change Manager 알림을 게시할 수 있도록 Amazon SNS 액세스 정책을 업데이트합니다. 이 태스크를 완료하지 않으면 주제를 추가하는 이벤트에 대한 알림을 보낼 권한이 Change Manager에 없습니다.

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/sns/v3/home에서 Amazon SNS 콘솔을 엽니다.
2. 탐색 창에서 주제를 선택합니다.
3. 태스크 1에서 생성한 주제를 선택한 다음 편집(Edit)을 선택합니다.
4. 액세스 정책(Access policy)를 확장합니다.
5. 다음 `Sid` 블록을 기존 정책에 추가 및 업데이트하고 각 *`user input placeholder`*를 사용자의 정보로 바꿉니다.

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

기존 Sid 블록 뒤에 이 블록을 입력하고 *region*, *account-id*, *topic-name*을 생성한 주제에 대한 적절한 값으로 바꿉니다.

6. 변경 사항 저장(Save changes)을 선택합니다.

이제 시스템은 주제에 추가하는 이벤트 유형이 발생할 때 Amazon SNS 주제에 알림을 보냅니다.

> **Important**
>
> AWS Key Management Service(AWS KMS) 서버 측 암호화 키를 사용하여 Amazon SNS 주제를 구성하는 경우에는 태스크 3을 완료해야 합니다.

태스크 3: (옵션) AWS Key Management Service 액세스 정책 업데이트

Amazon SNS 주제에 대해 AWS Key Management Service(AWS KMS) 서버 측 암호화를 설정한 경우 주제를 구성할 때 선택한 AWS KMS key의 액세스 정책도 업데이트해야 합니다. 다음 절차에 따라 Systems Manager가 태스크 1에서 생성한 Amazon SNS 주제에 Change Manager 승인 알림을 게시할 수 있도록 액세스 정책을 업데이트합니다.

1. AWS KMS 콘솔(https://console.aws.amazon.com/kms)을 엽니다.
2. 탐색 창에서 고객 관리형 키를 선택합니다.

3. 주제를 생성할 때 선택한 고객 관리형 키의 ID를 선택합니다.
4. 키 정책(Key policy) 섹션에서 정책 보기로 전환(Switch to policy view)을 선택합니다.
5. 편집을 선택합니다.
6. 기존 정책의 기존 Sid 블록 중 하나의 뒤에 다음 Sid 블록을 입력합니다. *user input placeholder*를 사용자의 정보로 바꿉니다.

```
{
    "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. 이제 교차 서비스 혼동된 대리자 문제 방지를 도울 수 있도록 리소스 정책 내의 기존 Sid 블록 중 하나의 뒤에 다음 Sid 블록을 입력합니다.

   이 블록은 aws:SourceArn 및 aws:SourceAccount 글로벌 조건 컨텍스트 키를 사용하여 Systems Manager가 리소스에 다른 서비스를 부여하는 권한을 제한합니다.

   각 *user input placeholder*를 사용자의 정보로 바꿉니다.

```
{
  "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. 변경 사항 저장(Save changes)을 선택합니다.

Change Manager에 대한 역할 및 권한 구성

Change Manager는 기본적으로 리소스에서 작업을 수행할 권한이 없습니다. AWS Identity and Access Management(IAM) 서비스 역할 또는 수임 역할을 사용하여 액세스 권한을 부여해야 합니다. 이 역할은 사용자를 대신하여 승인된 변경 요청에 지정된 실행서 워크플로를 안전하게 실행하도록 Change Manager를 활성화합니다. 이 역할은 Change Manager에 AWS Security Token Service(AWS STS) AssumeRole 신뢰를 부여합니다.

조직의 사용자를 대신하여 작업할 역할에 이러한 권한을 제공함으로써 사용자에게 일련의 해당 사용 권한 자체를 직접 부여할 필요가 없습니다. 권한에 의해 허용되는 작업은 승인된 작업으로만 제한됩니다.

계정 또는 조직의 사용자가 변경 요청을 생성할 때 이 수임 역할을 선택하여 변경 작업을 수행할 수 있습니다.

Change Manager에 대한 새 수임 역할을 생성하거나 기존 역할에 필요한 권한이 포함되도록 업데이트합니다.

Change Manager에 대한 서비스 역할을 생성해야 하는 경우 다음 작업을 완료하세요.

Tasks

- 작업 1: Change Manager에 대한 수임 역할 정책 생성하기
- 작업 2: Change Manager에 대한 수임 역할 생성하기
- 작업 3: iam:PassRole 정책을 다른 역할에 연결하기
- 작업 4: 다른 AWS 서비스을(를) 호출하기 위해 수임 역할에 인라인 정책 추가하기
- 작업 5: Change Manager에 대한 사용자 액세스 구성하기

작업 1: Change Manager에 대한 수임 역할 정책 생성하기

다음 절차에 따라 Change Manager 수임 역할에 연결할 정책을 생성합니다.

Change Manager용 수임 역할 정책을 생성하려면

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 정책을 선택한 후 정책 생성을 선택합니다.
3. 정책 생성(Create policy) 페이지에서 JSON 탭을 선택하고 기본 내용을 다음과 같이 바꿉니다. 이 내용은 사용자가 자신의 Change Manager 작업에 맞게 다음 단계에서 수정할 수 있습니다.

> **Note**
>
> 단일 AWS 계정으로 사용할 정책을 생성하고 있으며 여러 계정과 AWS 리전가 있는 조직이 아닌 경우 첫 번째 문 블록을 생략할 수 있습니다. Change Manager를 사용한 단일 계정의 경우 `iam:PassRole` 사용 권한이 필요하지 않습니다.

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

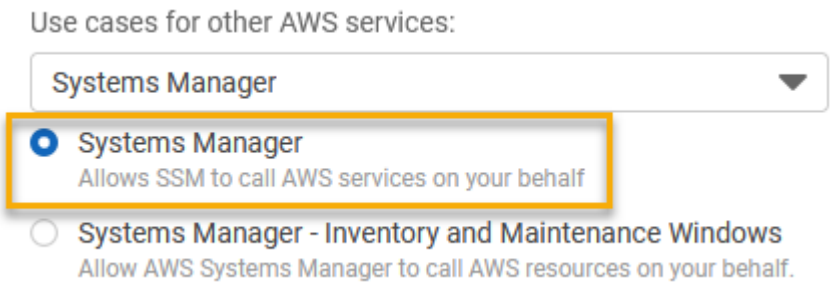
4. `iam:PassRole` 작업의 경우 실행서 워크플로를 시작할 수 있는 권한을 부여하려는 조직에 대해 정의된 모든 작업 기능의 ARN을 포함하도록 `Resource` 값을 업데이트합니다.
5. *region*, *account-id*, *template-name*, *delegated-admin-account-id*, *job-function* 자리표시자를 Change Manager 작업에 대한 값으로 교체합니다.
6. 두 번째 `Resource` 문의 경우 권한을 부여할 모든 변경 템플릿을 포함하도록 목록을 수정합니다. 또는 조직의 모든 변경 템플릿에 대한 권한을 부여하도록 `"Resource": "*"`를 지정합니다.
7. 다음: 태그(Next: Tags)를 선택합니다.
8. (선택 사항) 이 정책에 대한 액세스를 구성, 추적 또는 제어할 태그-키 값 페어를 하나 이상 추가합니다.
9. 다음: 검토를 선택합니다.
10. 정책 검토(Review Policy) 페이지에서 이름(Name) 상자에 **`MyChangeManagerAssumeRole`** 등의 이름을 입력한 다음 설명을 입력합니다(선택 사항).
11. 정책 생성(Create policy)을 선택하고 계속해서 작업 2: Change Manager에 대한 수임 역할 생성하기를 진행합니다.

작업 2: Change Manager에 대한 수임 역할 생성하기

다음 절차를 사용하여 Change Manager용 서비스 역할의 한 유형인 Change Manager 수임 역할을 생성합니다.

Change Manager용 수임 역할을 생성하려면

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할(Roles)을 선택한 후 역할 생성(Create role)을 선택합니다.
3. 신뢰할 수 있는 엔터티 선택(Select trusted entity)에서 다음을 선택합니다.
   1. 신뢰할 수 있는 엔터티 유형에서 AWS 서비스를 선택합니다.
   2. 다른 AWS 서비스의 사용 사례에서 Systems Manager를 선택합니다.
   3. 다음 이미지에 표시된 것과 같이 Systems Manager를 선택합니다.



4. Next(다음)를 선택합니다.
5. 연결된 권한 정책(Attached permissions policy) 페이지에서 작업 1: Change Manager에 대한 수임 역할 정책 생성하기에 생성한 수임 역할 정책(**MyChangeManagerAssumeRole** 등)을 검색합니다.
6. 수임 역할 정책 이름 옆에 있는 확인란을 선택한 후, 다음: 검토(Next: Review)를 선택합니다.
7. 역할 이름(Role name)에 새 인스턴스 프로파일의 이름(예: **MyChangeManagerAssumeRole**)을 입력합니다.
8. (선택) 설명(Description)에 이 인스턴스 역할에 대한 설명을 입력합니다.
9. (선택 사항) 이 역할에 대한 액세스를 구성, 추적 또는 제어할 태그-키 값 페어를 하나 이상 추가합니다.
10. 다음: 검토를 선택합니다.
11. (선택) 태그(Tags)에서 이 역할에 대한 액세스를 구성, 추적 또는 제어할 태그-키 값 페어를 하나 이상 추가한 후 역할 생성(Create role)을 선택합니다. 그러면 역할 페이지로 돌아갑니다.
12. 역할 생성(Create role)을 선택합니다. 그러면 역할 페이지로 돌아갑니다.

13. 역할(Roles) 페이지에서 방금 만든 역할을 선택하여 요약 페이지를 엽니다.

작업 3: **`iam:PassRole`** 정책을 다른 역할에 연결하기

다음 절차에 따라 `iam:PassRole` 정책을 IAM 인스턴스 프로파일 또는 IAM 서비스 역할에 연결합니다. (Systems Manager 서비스에서는 IAM 인스턴스 프로파일을 사용하여 EC2 인스턴스와 통신합니다. 하이브리드 및 멀티클라우드 환경의 비 EC2 관리형 노드의 경우 IAM 서비스 역할이 대신 사용됩니다.)

Change Manager 서비스는 `iam:PassRole` 정책을 연결함으로써 실행서 워크플로를 실행할 때 수임 역할 사용 권한을 다른 서비스 또는 Systems Manager 기능에 전달할 수 있습니다.

`iam:PassRole` 정책을 IAM 인스턴스 프로파일 또는 서비스 역할에 연결하는 방법

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할을 선택합니다.
3. 사용자가 생성한 Change Manager 수임 역할(예: **`MyChangeManagerAssumeRole`**)을 검색하여 그 이름을 선택합니다.
4. 수임 역할에 대한 요약(Summary) 페이지에서 권한(Permissions) 탭을 선택합니다.
5. 권한 추가, 인라인 정책 추가(Add permissions, Create inline policy)를 선택합니다.
6. 정책 생성(Create policy) 페이지에서 시각적 편집기(Visual editor) 탭을 선택합니다.
7. 서비스(Service)와 IAM을 차례대로 선택합니다.
8. 작업 필터링(Filter actions) 텍스트 상자에 **`PassRole`**을 입력하고 PassRole 옵션을 선택합니다.
9. 리소스(Resources)를 확장합니다. Specific(특정)이 선택되었는지 확인한 다음, ARN 추가(Add ARN)를 선택합니다.
10. 역할에 ARN 지정(Specify ARN for role) 필드에 수임 역할 권한을 전달하려는 IAM 인스턴스 프로파일 역할 또는 IAM 서비스 역할의 ARN을 입력합니다. 계정(Account) 및 역할 이름 및 경로(Role name with path) 필드에 값이 채워집니다.
11. 추가(Add)를 선택합니다.
12. 정책 검토(Review policy)를 선택합니다.
13. 이름(Name)에 이 정책을 알아볼 수 있는 이름을 입력하고, 정책 생성(Create policy)을 선택합니다.

추가 정보

- Systems Manager에 필요한 인스턴스 권한 구성

- 하이브리드 및 멀티클라우드 환경에서 Systems Manager에 필요한 IAM 서비스 역할 생성

작업 4: 다른 AWS 서비스을(를) 호출하기 위해 수임 역할에 인라인 정책 추가하기

변경 요청이 Change Manager 수임 역할을 사용하여 다른 AWS 서비스을(를) 호출하는 경우 해당 수임 역할이 다른 서비스를 호출할 권한이 있도록 구성되어야 합니다. 이 요구 사항은 `AWS-ConfigureS3BucketLogging`, `AWS-CreateDynamoDBBackup`, `AWS-RestartEC2Instance` 런북과 같이 변경 요청에 사용할 수 있는 모든 AWS Automation 런북(AWS-* 런북)에 적용됩니다. 또한 다른 서비스를 호출하는 작업을 사용하여 다른 AWS 서비스을(를) 호출하는 사용자 정의 실행서를 생성하는 경우에도 이 요구 사항이 항상 적용됩니다. 예를 들어, `aws:executeAwsApi`, `aws:CreateStack` 또는 `aws:copyImage` 작업을 사용하는 경우 이러한 서비스를 호출할 수 있는 권한을 포함하여 서비스 역할을 구성해야 합니다. 역할에 IAM 인라인 정책을 추가하여 다른 AWS 서비스에 대한 권한을 활성화할 수 있습니다.

다른 AWS 서비스을(를) 호출하기 위해 수임 역할에 인라인 정책을 추가하기(IAM 콘솔)

1. AWS Management Console에 로그인하여 https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할을 선택합니다.
3. 목록에서 업데이트하려는 수임 역할의 이름을 선택합니다(예:`MyChangeManagerAssumeRole`).
4. 권한 탭을 선택합니다.
5. 권한 추가, 인라인 정책 추가(Add permissions, Create inline policy)를 선택합니다.
6. JSON 탭을 선택합니다.
7. 호출하려는 AWS 서비스의 JSON 정책 문서를 입력합니다. 다음은 2개의 예제 JSON 정책 문서입니다.

   Amazon S3 **`PutObject`** 및 **`GetObject`** 예제

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
```

```
        }
    ]
}
```

Amazon EC2 **CreateSnapshot** 및 **DescribeSnapShots** 예제

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":"ec2:CreateSnapshot",
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":"ec2:DescribeSnapshots",
         "Resource":"*"
      }
   ]
}
```

IAM 정책 언어에 대한 자세한 내용은 IAM 사용 설명서의 IAM JSON 정책 참조를 참조하세요.

8. 작업이 완료되면 정책 검토(Review policy)를 선택합니다. 정책 검사기가 모든 구문 오류를 보고합니다.
9. 이름(Name)에 생성 중인 정책을 알아볼 수 있는 이름을 입력합니다. 정책 요약을 검토하여 정책이 부여한 권한을 확인합니다. 그런 다음 정책 생성을 선택하여 작업을 저장합니다.
10. 인라인 정책을 생성하면 이 정책이 역할에 자동으로 포함됩니다.

작업 5: Change Manager에 대한 사용자 액세스 구성하기

사용자, 그룹 또는 역할에 관리자 권한이 부여되어 있는 경우 Change Manager에 액세스할 수 있습니다. 관리자 권한이 없는 경우 관리자가 `AmazonSSMFullAccess` 관리형 정책 또는 비교 가능한 권한을 제공하는 정책을 해당 사용자, 그룹 또는 역할에 할당해야 합니다.

다음 절차에 따라 Change Manager를 사용할 수 있는 사용자를 구성합니다. 선택하는 사용자에게 Change Manager를 구성하고 실행할 수 있는 권한이 부여됩니다.

조직에서 사용 중인 ID 애플리케이션에 따라 사용자 액세스를 구성하는 데 사용할 수 있는 세 가지 옵션을 선택할 수 있습니다. 사용자 액세스를 구성하는 동안 다음을 할당하거나 추가합니다.

1. `AmazonSSMFullAccess` 정책 또는 Systems Manager에 액세스할 수 있는 권한을 부여하는 유사한 정책을 할당합니다.
2. `iam:PassRole` 정책을 할당합니다.
3. 작업 2: Change Manager에 대한 수임 역할 생성하기 종료 시 복사한 Change Manager 수임 역할에 대한 ARN을 추가합니다.

액세스 권한을 제공하려면 사용자, 그룹 또는 역할에 권한을 추가하세요:

- AWS IAM Identity Center의 사용자 및 그룹:

  권한 세트를 생성합니다. AWS IAM Identity Center 사용 설명서의 권한 세트 생성의 지침을 따르세요.

- ID 제공자를 통해 IAM에서 관리되는 사용자:

  ID 페더레이션을 위한 역할을 생성합니다. IAM 사용 설명서의 서드 파티 자격 증명 공급자의 역할 만들기(연합)의 지침을 따르세요.

- IAM 사용자:
  - 사용자가 맡을 수 있는 역할을 생성합니다. IAM 사용 설명서에서 IAM 사용자의 역할 생성의 지침을 따르세요.
  - (권장되지 않음)정책을 사용자에게 직접 연결하거나 사용자를 사용자 그룹에 추가합니다. IAM 사용 설명서에서 사용자(콘솔)에 권한 추가의 지침을 따르세요.

이렇게 해서 Change Manager에 역할 구성을 마쳤습니다. 이제 Change Manager 작업에서 Change Manager 수임 역할 ARN을 사용할 수 있습니다.

자동 승인 실행서 워크플로에 대한 액세스 제어

조직 또는 계정에 대해 생성된 각 변경 템플릿에 해당 템플릿에서 생성된 변경 요청을 자동 승인된 변경 요청으로 실행할 수 있는지 여부를 지정할 수 있습니다. 즉, 검토 단계 없이 자동으로 실행됩니다(변경 고정 이벤트 제외).

그러나 변경 템플릿에서 허용하는 경우에도 특정 사용자, 그룹 또는 AWS Identity and Access Management(IAM) 역할이 자동 승인된 변경 요청을 실행하지 못하도록 할 수 있습니다. 이렇게 하려

면 사용자, 그룹 또는 IAM 역할에 할당된 IAM 정책에서 `StartChangeRequestExecution` 작업에 `ssm:AutoApprove` 조건 키를 사용합니다.

다음 정책을 인라인 정책으로 추가할 수 있습니다. 여기서 조건이 `false`로 지정되어 사용자가 자동 승인 가능한 변경 요청을 실행하지 못하도록 할 수 있습니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
            {
            "Effect": "Allow",
            "Action": "ssm:StartChangeRequestExecution",
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "ssm:AutoApprove": "false"
                }
            }
        }
    ]
}
```

인라인 정책 지정에 대한 자세한 내용은 IAM User Guide의 Inline policies 및 Adding and removing IAM identity permissions를 참조하세요.

Systems Manager 정책의 조건 키에 대한 자세한 내용은 Systems Manager 조건 키를 참조하세요.

Change Manager 작업

AWS Systems Manager의 기능인 Change Manager를 사용하면 조직 전체 또는 단일 AWS 계정의 사용자가 필요한 권한이 부여된 변경 관련 태스크를 수행할 수 있습니다. Change Manager 태스크에는 다음이 포함됩니다.

- 변경 템플릿을 생성, 검토 및 승인하거나 거부합니다.

  변경 템플릿은 필수 승인, 사용 가능한 실행서 및 변경 요청에 대한 알림 옵션과 같은 항목을 정의하는 Change Manager의 구성 설정 모음입니다.
- 변경 요청을 생성, 검토 및 승인하거나 거부합니다.

변경 요청은 AWS 또는 온프레미스 환경에서 하나 이상의 리소스를 업데이트하는 Automation 실행 서를 실행하기 위한 Change Manager의 요청입니다. 변경 요청은 변경 템플릿을 사용하여 생성됩니다.

- 변경 템플릿 및 변경 요청에 대한 검토자가 될 수 있는 조직 또는 계정의 사용자를 지정합니다.
- Change Manager에서 사용자 자격 증명을 관리하는 방법과 Change Manager 작업에서 적용되는 사용 가능한 모범 사례 옵션과 같은 구성 설정을 편집합니다. 이러한 설정 구성에 대한 자세한 내용은 Change Manager 옵션 및 모범 사례 구성 섹션을 참조하세요.

주제

변경 템플릿 사용

변경 템플릿은 필수 승인, 사용 가능한 실행서 및 변경 요청에 대한 알림 옵션과 같은 항목을 정의하는 Change Manager의 구성 설정 모음입니다.

> **Note**
>
> AWS는 AWS Systems Manager의 기능인 Change Manager를 시도해 보는 데 사용할 수 있는 샘플 Hello World 변경 템플릿을 제공합니다. 그러나 조직 또는 계정의 리소스에 허용할 변경 사항을 정의하려면 변경 템플릿을 직접 생성합니다.

실행서 워크플로가 실행될 때 변경되는 내용은 Automation 실행서의 내용을 기반으로 합니다. 생성하는 각 변경 템플릿에 변경 요청을 하는 사용자가 업데이트 중 실행하도록 선택할 수 있는 하나 이상의 Automation 실행서를 포함할 수 있습니다. 또한 요청자가 변경 요청에 사용할 수 있는 Automation 실행서를 선택할 수 있도록 하는 변경 템플릿을 생성할 수도 있습니다.

변경 템플릿을 생성하려면 템플릿 생성(Create template) 콘솔 페이지의 빌더(Builder) 옵션을 사용하여 변경 템플릿을 구축합니다. 또는 편집기(Editor) 옵션을 사용하여 실행서 워크플로에 대해 원하는 구성으로 JSON 또는 YAML 콘텐츠를 직접 작성할 수 있습니다. 명령줄 도구를 사용하여 외부 파일에 저장된 변경 템플릿의 JSON 콘텐츠로 변경 템플릿을 생성할 수도 있습니다.

주제

AWS 관리형 **`Hello World`** 변경 템플릿 사용해 보기

AWS Systems Manager의 기능인 Change Manager 설정을 마친 후 샘플 Automation 실행서 `AWS-HelloWorld`를 사용하는 샘플 변경 템플릿 `AWS-HelloWorldChangeTemplate`으로 검토 및 승인 프로세스를 테스트할 수 있습니다. 이 템플릿은 구성된 권한, 승인자 할당 및 승인 프로세스를 테스트하거나 확인하기 위해 설계되었습니다. 조직 또는 계정에서 이 변경 템플릿을 사용하기 위한 승인은 이미 AWS에서 제공했습니다. 그러나 이 변경 템플릿을 기반으로 하는 모든 변경 요청은 여전히 조직 또는 계정의 검토자의 승인을 받아야 합니다.

리소스를 변경하는 대신 이 템플릿과 연결된 실행서 워크플로의 결과는 Automation단계의 출력에 메시지를 인쇄하는 것입니다.

시작하기 전 준비 사항

시작하기 전에 다음 태스크를 완료합니다.

- AWS Organizations를 사용하여 조직 전체의 변경 사항을 관리하는 경우 조직용 Change Manager 설정(관리 계정)에 설명된 조직 설정 태스크를 완료합니다.
- Change Manager 옵션 및 모범 사례 구성에 설명된 대로 위임된 관리자 계정 또는 단일 계정에 대해 Change Manager를 구성합니다.

> **Note**
>
> Change Manager 설정에서 모든 템플릿에 모니터 필요(Require monitors for all templates) 모범 사례 옵션을 설정한 경우 Hello World 변경 템플릿을 테스트하는 동안 해당 옵션을 일시적으로 해제합니다.

AWS 관리형 Hello World 변경 템플릿을 사용해 보려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.

2. 탐색 창에서 Change Manager를 선택합니다.
3. 요청 생성(Create request)을 선택합니다.
4. `AWS-HelloWorldChangeTemplate`이라는 변경 템플릿 선택하고 다음(Next)을 선택합니다.
5. 이름(Name)에 용도를 쉽게 식별할 수 있도록 변경 요청의 이름을 입력합니다(예: **`MyChangeRequestTest`**).
6. 변경 요청을 생성하는 나머지 단계는 변경 요청 생성 섹션을 참조하세요.

다음 단계

변경 요청 승인에 대한 자세한 내용은 변경 요청 검토 및 승인 또는 거부 섹션을 참조하세요.

변경 요청의 상태 및 결과를 보려면 Change Manager의 요청(Requests) 탭에서 변경 요청의 이름을 선택합니다.

변경 템플릿 생성

변경 템플릿은 필수 승인, 사용 가능한 실행서 및 변경 요청에 대한 알림 옵션과 같은 항목을 정의하는 Change Manager의 구성 설정 모음입니다.

빌더 및 편집기 옵션이 포함된 콘솔 또는 명령줄 도구를 사용하여 AWS Systems Manager의 기능인 Change Manager에서 작업에 대한 변경 템플릿을 생성할 수 있습니다.

주제

- 변경 템플릿의 승인 정보
- 빌더를 사용하여 변경 템플릿 생성
- 편집기를 사용하여 변경 템플릿 생성
- 명령줄 도구를 사용하여 변경 템플릿 생성

변경 템플릿의 승인 정보

생성하는 변경 템플릿마다 해당 템플릿에서 생성된 변경 요청에 대해 최대 5개의 승인 수준을 지정할 수 있습니다. 이러한 수준마다 최대 5명의 잠재적 승인자를 지정할 수 있습니다. 승인자는 단일 사용자로 제한되지 않습니다. IAM 그룹 또는 IAM 역할을 개별 승인자로 지정할 수도 있습니다. IAM 그룹 및 IAM 역할의 경우 그룹 또는 역할에 속한 한 명 이상의 사용자가 변경 요청에 필요한 총 승인 수를 받기 위한 승인을 제공할 수 있습니다. 변경 템플릿에 필요한 것보다 더 많은 승인자를 지정할 수도 있습니다.

Change Manager는 수준별 승인과 라인별 승인이라는 두 가지 주요 승인 방식을 지원합니다. 상황에 따라 두 가지 유형을 조합하여 사용할 수도 있습니다. Change Manager 작업에서는 수준별 승인만 사용하는 것이 좋습니다.

Per-level approvals

권장. 2023년 1월 23일부터 Change Manager에서 수준별 승인을 지원합니다. 이 모델에서는 먼저 변경 템플릿의 승인 수준별로 필요한 승인 수를 지정합니다. 그런 다음 수준에 대해 그 이상의 승인자를 지정합니다. 그러나 지정하는 수의 수준별 승인자만 변경 요청을 승인해야 합니다. 예를 들어, 5명의 승인자를 지정하고 3건의 승인을 요구할 수 있습니다.

이 승인 유형의 콘솔 보기와 JSON 샘플은 the section called “샘플 수준별 승인 구성” 섹션을 참조하세요.

Per-line approvals

이전 버전과의 호환성을 위해 지원됨. Change Manager의 원래 릴리스는 라인별 승인만 지원했습니다. 이 모델에서는 승인 수준에 지정된 모든 승인자가 승인 라인으로 표시됩니다. 각 승인자가 변경 요청을 승인해야 해당 수준에서 변경 요청이 승인되었습니다. 2023년 1월 23일 전에는 이것이 승인에 대해 지원되는 유일한 모델이었습니다. 이 날짜 전에 생성된 변경 템플릿은 라인별 승인을 계속 지원하지만 수준별 승인을 대신 사용하는 것이 좋습니다.

이 승인 유형의 콘솔 보기와 JSON 샘플은 the section called “샘플 라인별 승인 구성” 섹션을 참조하세요.

Combined per-line and per-level approvals

권장되지 않음. 콘솔의 빌더 탭에서 이제 라인별 승인 추가를 지원하지 않습니다. 그러나 경우에 따라 변경 템플릿에서 라인별 승인과 수준별 승인이 모두 이루어질 수 있습니다. 2023년 1월 23일 전에 생성된 변경 템플릿을 업데이트하거나 YAML 콘텐츠를 수동으로 편집하여 변경 템플릿을 생성 또는 업데이트하는 경우가 여기에 해당합니다.

이 승인 유형의 콘솔 보기와 JSON 샘플은 the section called “샘플 수준별 및 라인별 승인 결합 구성” 섹션을 참조하세요.

Important

라인별 승인과 수준별 승인을 조합하여 변경 템플릿을 생성할 수 있지만 이 구성이 권장되거나 필수는 아닙니다. 더 많은 승인이 필요한 승인 유형(라인별 승인 또는 수준별 승인)이 우선합니다. 예:

- 변경 템플릿이 수준별 승인을 3건 지정하고 라인별 승인을 5건 지정하는 경우 5건의 승인이 필요합니다.
- 변경 템플릿이 수준별 승인을 4건 지정하고 라인별 승인을 2건 지정하는 경우 4건의 승인이 필요합니다.

YAML 또는 JSON 콘텐츠를 수동으로 편집하여 라인별 승인과 수준별 승인을 모두 포함하는 수준을 생성할 수 있습니다. 그런 다음 빌더 탭에 수준과 개별 라인 모두에 필요한 승인 수를 지정하기 위한 컨트롤이 표시됩니다. 그러나 콘솔을 사용하여 추가하는 새 수준은 여전히 수준별 승인 구성만 지원합니다.

변경 요청 알림 및 거부

Amazon SNS 알림

변경 템플릿을 사용하여 변경 요청이 생성되면 해당 수준에서 승인 알림에 대해 지정된 Amazon Simple Notification Service(SNS) 주제의 구독자에게 알림이 전송됩니다. 변경 템플릿에 알림 주제를 지정하거나 변경 요청을 생성하는 사용자가 알림 주제를 지정하도록 허용할 수 있습니다.

한 수준에서 필요한 최소 승인 수가 수신되면 다음 수준에 대한 Amazon SNS 주제를 구독하는 승인자에게 알림이 전송됩니다.

> **Important**
>
> 함께 지정하는 IAM 역할, 그룹 및 사용자가 지정한 필요한 승인 수를 충족하기에 충분한 승인자를 제공하는지 확인하세요. 예를 들어, 3명의 사용자를 포함하는 단일 IAM 그룹만 승인자로 지정하는 경우 해당 수준에서 5건의 승인이 필수라고 지정할 수 없으며 3개 이하만 가능합니다.

변경 요청 거부

얼마나 많은 승인 수준과 승인자를 지정하는지에 관계없이 해당 요청에 대한 런북 워크플로가 발생하지 않도록 하려면 변경 요청에 대한 거부가 한 번만 필요합니다.

Change Manager 승인 유형 예제

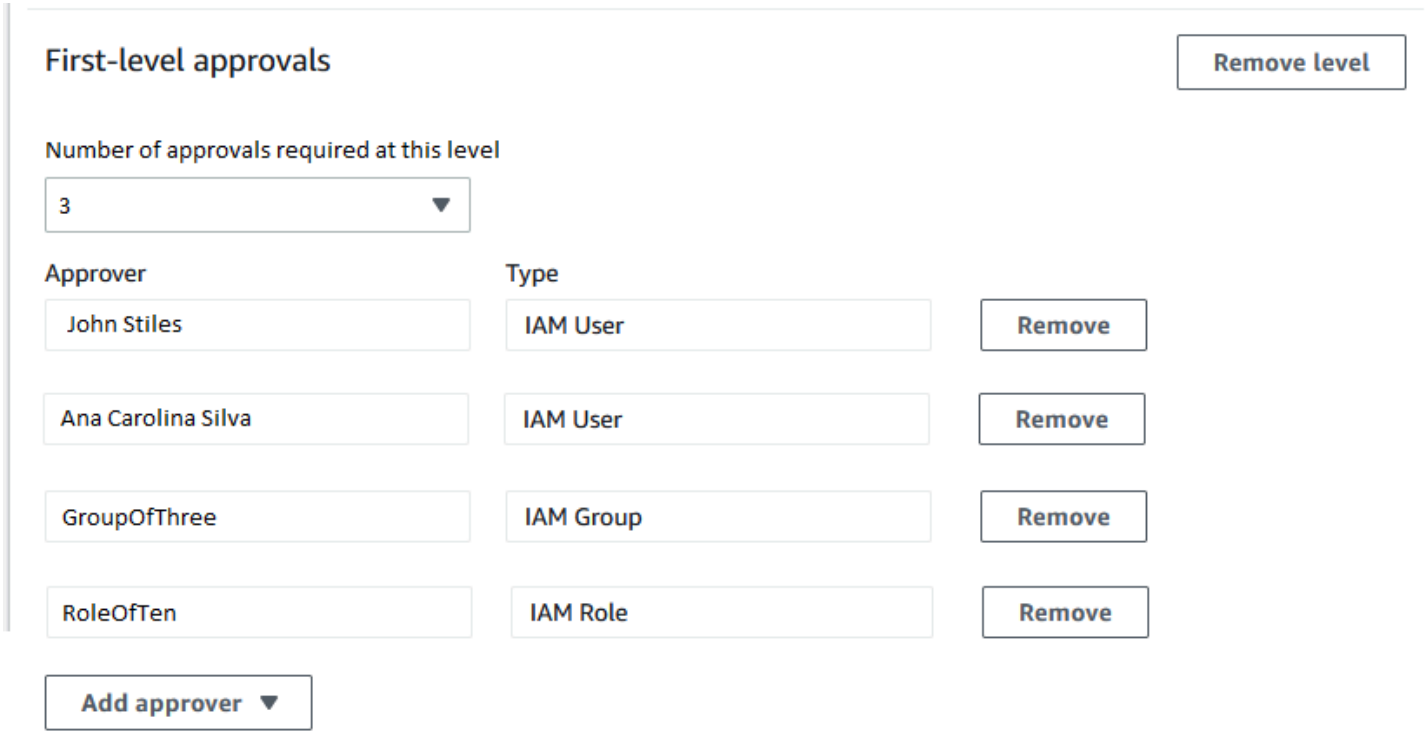
다음 샘플에서는 Change Manager의 세 가지 승인 유형에 대한 콘솔 보기와 JSON 콘텐츠를 보여줍니다.

주제

샘플 수준별 승인 구성

다음 이미지에 표시된 수준별 승인 수준 설정에서는 3건의 승인이 필요합니다. 승인자로 지정된 임의의 IAM 사용자, 그룹 및 역할 조합으로부터 이러한 승인을 받을 수 있습니다. 지정된 승인자에는 2명의 IAM 사용자(John Stiles 및 Ana Carolina Silva), 3명의 멤버로 구성된 사용자 그룹(`GroupOfThree`) 및 10명의 사용자를 나타내는 사용자 역할(`RoleOfTen`)이 포함됩니다.

`GroupOfThree` 그룹의 세 사용자 모두 변경 요청을 승인하면 해당 수준에 대해 승인됩니다. 각 사용자, 그룹 또는 역할에서 승인을 받을 필요는 없습니다. 임의의 지정된 승인자 조합으로부터 최소 수의 승인을 받을 수 있습니다. Change Manager 작업에는 수준별 승인을 사용하는 것이 좋습니다.



다음 샘플에서는 이 구성에 대한 YAML 코드의 일부를 보여줍니다.

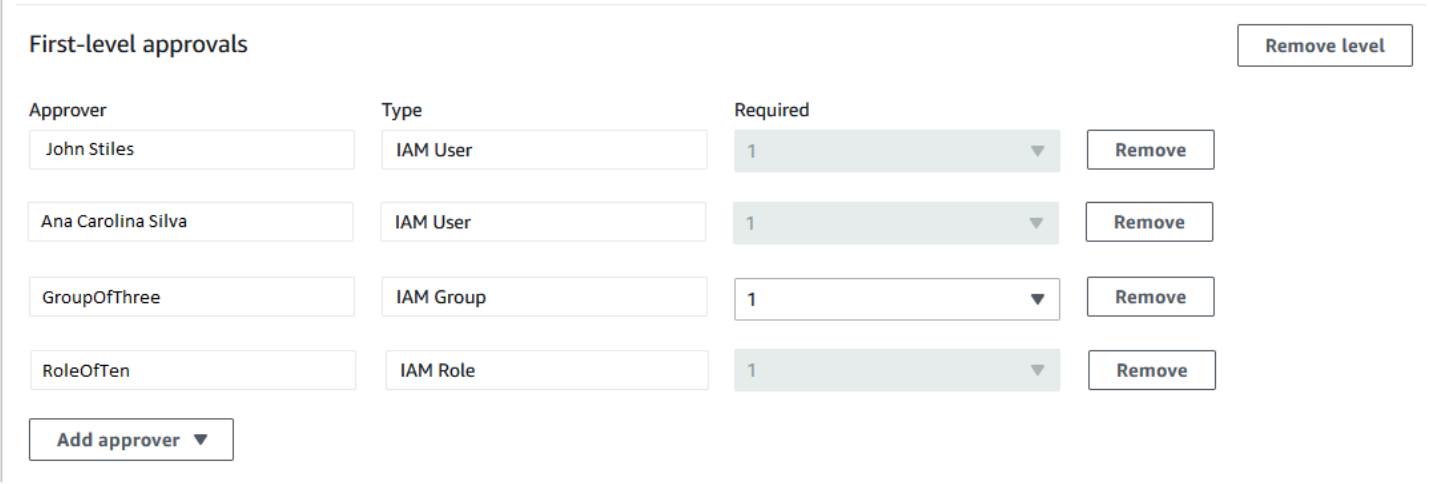
> **Note**
>
> 이 버전의 YAML 코드에는 추가 입력인 MinRequiredApprovals(대문자 M으로 시작)가 포함되어 있습니다. 이 입력 값은 사용 가능한 모든 검토자에게서 필요한 승인 수를 나타냅니다. 또한 Approvers 목록의 각 승인자에 대한 minRequiredApprovals(소문자 m으로 시작) 값은 0입니다. 이는 승인자가 전체 승인에 기여할 수 있지만 그렇게 할 필요는 없음을 나타냅니다.

```
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
```

샘플 라인별 승인 구성

다음 이미지에 표시된 승인 수준 설정에서 4명의 승인자가 지정되어 있습니다. 여기에는 2명의 IAM 사용자(John Stiles 및 Ana Carolina Silva), 3명의 멤버로 구성된 사용자 그룹(GroupOfThree) 및 10명

의 사용자를 나타내는 사용자 역할(RoleOfTen)이 포함됩니다. 이전 버전과의 호환성을 위해 라인별 승인이 지원되지만 권장되지는 않습니다.



이 라인별 승인 구성에서 변경 요청이 승인되려면 모든 승인자 라인(John Stiles, Ana Carolina Silva, GroupOfThree 그룹의 멤버 1명 및 RoleOfTen 역할의 멤버 1명)의 승인을 받아야 합니다.

다음 샘플에서는 이 구성에 대한 YAML 코드의 일부를 보여줍니다.

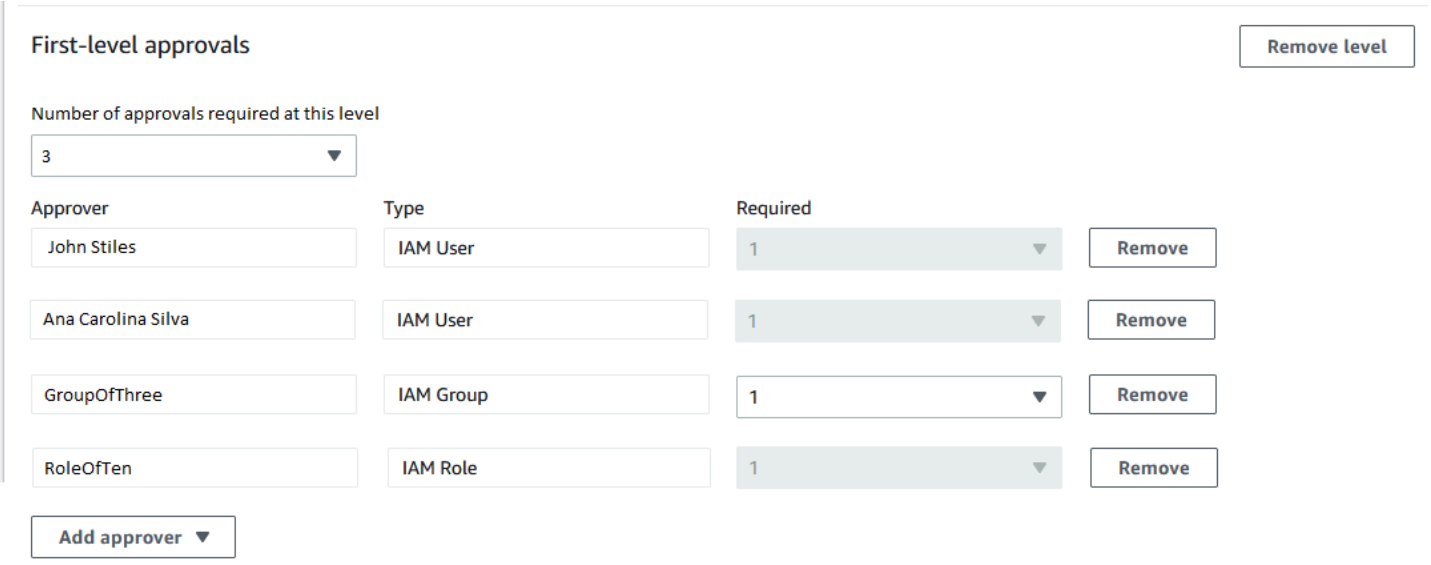
> **Note**
>
> 각 minRequiredApprovals 승인자의 값이 1인지 확인합니다. 이는 각 승인자로부터 승인이 한 건씩 필요함을 나타냅니다.

```
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
```

샘플 수준별 및 라인별 승인 결합 구성

다음 이미지에 표시된 수준별 및 라인별 승인 결합 설정에서는 수준에 대해 3건의 승인이 지정되어 있지만 라인 항목 승인에 대해서는 4건의 승인이 지정되어 있습니다. 더 많은 승인이 필요한 승인 유형이 다른 승인 유형보다 우선하므로 이 구성에는 4건의 승인이 필요합니다. 수준별 승인과 라인별 승인 결합은 권장되지 않습니다.




```
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
```

주제

- 빌더를 사용하여 변경 템플릿 생성
- 편집기를 사용하여 변경 템플릿 생성
- 명령줄 도구를 사용하여 변경 템플릿 생성

빌더를 사용하여 변경 템플릿 생성

AWS Systems Manager의 기능인 Change Manager의 변경 템플릿용 빌더를 사용하면 JSON 또는 YAML 구문을 사용하지 않고도 변경 템플릿에 정의된 실행서 워크플로를 구성할 수 있습니다. 옵션을 지정하면 Systems Manager가 실행서 워크플로를 실행하는 데 사용할 수 있는 YAML 형식으로 입력 내용이 변환됩니다.

빌더를 사용하여 변경 템플릿을 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Change Manager를 선택합니다.
3. 템플릿 생성(Create template)을 선택합니다.

4. 이름(Name)에 용도를 쉽게 식별할 수 있도록 템플릿의 이름을 입력합니다(예: **`UpdateEC2LinuxAMI`**).
5. 변경 템플릿 세부 정보(Change template details) 섹션에서 다음을 수행합니다.

   - 설명(Description)에 생성 중인 변경 템플릿이 사용되는 방법과 시기에 대한 간략한 설명을 제공합니다.

     이 설명은 변경 요청을 생성하는 사용자가 올바른 변경 템플릿을 사용하고 있는지 판단하는 데 도움이 됩니다. 또한 변경 요청을 검토하는 사람이 요청을 승인해야 하는지 여부를 이해하는 데 도움이 됩니다.
   - 변경 템플릿 유형(Change template type)에서 표준 변경 템플릿을 생성할지 아니면 긴급 변경 템플릿을 생성할지 지정합니다.

     긴급 변경 템플릿은 AWS Systems Manager Change Calendar가 사용 중인 캘린더의 이벤트로 인해 변경이 차단된 경우에도 변경해야 하는 상황에 사용됩니다. 긴급 변경 템플릿에서 생성된 변경 요청은 여전히 지정된 승인자의 승인을 받아야 하지만 일정이 차단된 경우에도 요청된 변경은 계속 실행될 수 있습니다.
   - 실행서 옵션(Runbook options)에서 변경 요청을 생성할 때 사용자가 선택할 수 있는 실행서를 지정합니다. 한 개 또는 여러 개의 실행서를 추가할 수 있습니다. 또는 요청자가 사용할 실행서를 지정하도록 허용할 수 있습니다. 이러한 경우 하나의 실행서만 변경 요청에 포함될 수 있습니다.
   - [실행서(Runbook)]에서 사용자가 변경 요청에 대해 선택할 수 있는 실행서의 이름과 버전을 선택합니다. 변경 템플릿에 추가하는 실행서의 수에 관계없이 변경 요청당 하나만 선택할 수 있습니다.

     이전에 [아무 실행서나 사용할 수 있음(Any runbook can be used)]을 선택한 경우 실행서를 지정하지 않습니다.

     > **Tip**
     >
     > 실행서와 실행서 버전을 선택한 다음 [보기]를 선택하여 Systems Manager Documents 인터페이스에서 실행서의 내용을 검사합니다.
6. [템플릿 정보(Template information)] 섹션에서 Markdown을 사용하여 이 변경 템플릿에서 변경 요청을 생성하는 사용자에 대한 정보를 입력합니다. 변경 요청을 생성하는 사용자를 위해 포함할 수 있는 일련의 질문이 이미 있거나 사용자가 대신 다른 정보와 질문을 추가할 수 있습니다.

> **Note**
>
> Markdown은 문서와 문서 내의 개별 단계에 wiki 스타일의 설명을 추가할 수 있는 마크업 언어입니다. Markdown 사용에 대한 자세한 내용은 AWS에서 Markdown 사용을 참조하세요.

변경 및 롤백 계획의 일부로 실행하는 데 필요한 수동 단계 나열과 같이 승인자가 각 변경 요청을 승인할지 여부를 결정하는 데 도움이 되도록 사용자가 변경 요청에 대해 답변할 수 있는 질문을 제공하는 것이 좋습니다.

> **Tip**
>
> [미리 보기 숨기기(Hide preview)]와 [미리 보기 표시(Show preview)]를 전환하여 작성 시 내용이 어떤 형태인지 확인합니다.

7. [변경 요청 승인(Change request approvals)] 섹션에서 다음을 수행합니다.

   - (옵션) 이 변경 템플릿에서 생성된 변경 요청이 승인자의 검토 없이 자동으로 실행되게 하려면 (변경 고정 이벤트 제외) [자동 승인 사용(Enable auto-approval)]을 선택합니다.

     > **Note**
     >
     > 변경 템플릿에서 자동 승인을 사용하도록 설정하면 사용자에게 검토자를 우회할 수 있는 옵션이 제공됩니다. 변경 요청을 생성할 때 검토자를 지정하도록 선택할 수 있습니다. 따라서 변경 템플릿에서 검토자 옵션을 계속 지정해야 합니다.

     > **Important**
     >
     > 변경 템플릿에 자동 승인을 사용하면 사용자는 실행하기 전에 검토자가 검토할 필요가 없는 해당 템플릿을 사용하여 변경 요청을 제출할 수 있습니다(변경 고정 이벤트 승인자는 제외). 특정 사용자, 그룹 또는 IAM 역할이 자동 승인 요청을 제출하지 못하도록 제한하려는 경우 이 용도로 IAM 정책의 조건을 사용할 수 있습니다. 자세한 내용은 자동 승인 실행서 워크플로에 대한 액세스 제어 단원을 참조하십시오.

- 이 수준에서 필요한 승인 수에서 이 변경 템플릿에서 생성된 변경 요청이 이 수준에 대해 받아야 하는 승인 수를 선택합니다.
- 필수 첫 번째 수준 승인자를 추가하려면 [승인자 추가(Add approver)]를 선택한 후 다음 중에서 선택합니다.
  - [템플릿 지정 승인자(Template specified approvers)] - 계정에서 하나 이상의 사용자, 그룹 또는 AWS Identity and Access Management(IAM) 역할을 선택하여 이 변경 템플릿에서 생성된 변경 요청을 승인합니다. 이 템플릿을 사용하여 생성된 모든 변경 요청은 지정한 각 승인자가 검토하고 승인해야 합니다.
  - 지정된 승인자 요청 - 변경 요청을 하는 사용자가 요청 시 검토자를 지정하며 계정의 사용자 목록에서 선택할 수 있습니다.

    [필수(Required)] 열에 입력한 숫자에 따라 이 변경 템플릿을 사용하는 변경 요청에 의해 지정되어야 하는 검토자 수가 결정됩니다.

> **Important**
>
> 2023년 1월 23일 이전에는 빌더 탭에서 라인별 승인만 지정할 수 있었습니다. 빌더 탭을 사용하여 기존 변경 템플릿에 추가하는 새 변경 템플릿과 새 수준은 수준별 승인만 지원합니다. Change Manager 작업에서는 수준별 승인만 사용하는 것이 좋습니다.
> 자세한 내용은 변경 템플릿의 승인 정보 단원을 참조하십시오.

- [승인자에게 알리는 SNS 주제(SNS topic to notify approvers)]에서 다음을 수행합니다.
  1. 다음 중 하나를 선택하여 변경 요청이 검토할 준비가 되었다는 알림을 승인자에게 보내는 데 사용할 계정의 Amazon Simple Notification Service(Amazon SNS) 주제를 지정합니다.
     - [SNS Amazon 리소스 이름(ARN) 입력(Enter an SNS Amazon Resource Name (ARN))] - [주제 ARN(Topic ARN)]에 기존 Amazon SNS 주제의 ARN을 입력합니다. 이 주제는 조직의 모든 계정에 있을 수 있습니다.
     - 기존 SNS 주제 선택(Select an existing SNS topic) - 대상 알림 주제(Target notification topic)에서 현재 AWS 계정의 기존 Amazon SNS 주제의 ARN을 선택합니다. (현재 AWS 계정 및 AWS 리전에서 Amazon SNS 주제를 아직 생성하지 않은 경우 이 옵션을 사용할 수 없습니다.)
     - [변경 요청 생성 시 SNS 주제 지정(Specify SNS topic when the change request is created)] - 변경 요청을 생성하는 사용자는 알림에 사용할 Amazon SNS 주제를 지정할 수 있습니다.

> **Note**
>
> 선택한 Amazon SNS 주제는 전송되는 알림과 알림이 전송되는 구독자를 지정하도록 구성되어야 합니다. Change Manager에서 알림을 보낼 수 있도록 액세스 정책도 Systems Manager에 권한을 부여해야 합니다. 자세한 내용은 Change Manager 알림에 대한 Amazon SNS 주제 구성 섹션을 참조하세요.

2. 알림 추가를 선택합니다.

8. (옵션) 수준을 추가하려면 [승인 수준 추가(Add approval level)]를 선택하고 이 수준에 대해 템플릿 지정 승인자와 요청 지정 승인자 중에서 선택합니다. 그런 다음 SNS 주제를 선택하여 이 수준의 승인자에게 알립니다.

   첫 번째 수준 승인자가 모든 승인을 받으면 두 번째 수준 승인자에게 알림이 전송됩니다.

   각 템플릿에 최대 5개 수준의 승인자를 추가할 수 있습니다. 예를 들어 첫 번째 수준에 대해 기술 역할의 사용자의 승인을 요구한 다음 두 번째 수준에 대한 관리 승인을 요구할 수 있습니다.

9. [모니터링(Monitoring)] 섹션에서 [모니터링할 CloudWatch 경보(CloudWatch alarm to monitor)]에 대해 현재 계정의 Amazon CloudWatch 경보 이름을 입력하여 이 템플릿을 기반으로 하는 실행서 워크플로의 진행 상황을 모니터링합니다.

> **Tip**
>
> 새 경보를 생성하거나 지정하려는 경보 설정을 검토하려면 [Amazon CloudWatch 콘솔 열기(Open the Amazon CloudWatch console)]를 선택합니다. CloudWatch 경보 사용에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 Amazon CloudWatch 경보 사용을 참조하세요.

10. Notifications(알림) 섹션에서 다음을 수행합니다.

    1. 다음 중 하나를 선택하여 이 변경 템플릿을 사용하여 생성된 변경 요청에 대한 알림을 보내는 데 사용할 계정의 Amazon SNS 주제를 지정합니다.

       - [SNS Amazon 리소스 이름(ARN) 입력(Enter an SNS Amazon Resource Name (ARN))] - [주제 ARN(Topic ARN)]에 기존 Amazon SNS 주제의 ARN을 입력합니다. 이 주제는 조직의 모든 계정에 있을 수 있습니다.
       - 기존 SNS 주제 선택(Select an existing SNS topic) - 대상 알림 주제(Target notification topic)에서 현재 AWS 계정의 기존 Amazon SNS 주제의 ARN을 선택합니다. (현재 AWS 계정

및 AWS 리전에서 Amazon SNS 주제를 아직 생성하지 않은 경우 이 옵션을 사용할 수 없습니다.)

> **Note**
>
> 선택한 Amazon SNS 주제는 전송되는 알림과 알림이 전송되는 구독자를 지정하도록 구성되어야 합니다. Change Manager에서 알림을 보낼 수 있도록 액세스 정책도 Systems Manager에 권한을 부여해야 합니다. 자세한 내용은 Change Manager 알림에 대한 Amazon SNS 주제 구성 섹션을 참조하세요.

2. 알림 추가를 선택합니다.

11. (옵션) [문서 태그(Document tags)] 섹션에서 변경 템플릿에 적용할 하나 이상의 태그 키 이름/값 페어를 입력합니다.

    태그는 리소스에 할당하는 선택적 메타데이터입니다. 태그를 사용하여 용도, 소유자 또는 환경을 기준으로 하는 등 리소스를 다양한 방식으로 분류할 수 있습니다. 예를 들어 변경 템플릿에 태그를 지정하여 변경 유형과 실행 환경을 식별할 수 있습니다. 다음 키 이름/값 페어를 지정할 수 있습니다.

    - `Key=TaskType,Value=InstanceRepair`
    - `Key=Environment,Value=Production`

    Systems Manager 리소스 태그 지정에 대한 자세한 내용은 Systems Manager 리소스에 태그 지정 섹션을 참조하세요.

12. [저장 후 미리 보기(Save and preview)]를 선택합니다.

13. 생성 중인 변경 템플릿의 세부정보를 검토합니다.

    검토를 위해 제출하기 전에 변경 템플릿을 변경하려면 [작업, 편집(Actions, Edit)]을 선택합니다.

    변경 템플릿의 내용에 만족하면 [검토를 위해 제출(Submit for review)]을 선택합니다. Change Manager의 [설정(Settings)] 탭에서 템플릿 검토자로 지정된 조직 또는 계정의 사용자는 새 변경 템플릿이 검토 보류 중이라는 알림을 받습니다.

    변경 템플릿에 대해 Amazon SNS 주제가 지정된 경우 변경 템플릿이 거부되거나 승인되면 알림이 전송됩니다. 이 변경 템플릿과 관련된 알림을 받지 못한 경우 나중에 Change Manager로 돌아가서 상태를 확인합니다.

편집기를 사용하여 변경 템플릿 생성

이 주제의 단계에 따라 콘솔 컨트롤을 사용하는 대신 JSON 또는 YAML을 입력하여 AWS Systems Manager의 기능인 Change Manager에서 변경 템플릿을 구성합니다.

편집기를 사용하여 변경 템플릿을 생성하려면

1. 탐색 창에서 Change Manager를 선택합니다.
2. 템플릿 생성(Create template)을 선택합니다.
3. [이름(Name)]에 용도를 쉽게 식별할 수 있도록 템플릿의 이름을 입력합니다(예: **`RestartEC2LinuxInstance`**).
4. [템플릿 세부 정보 변경(Change template details)] 위에서 [편집기(Editor)]를 선택합니다.
5. [문서 편집기(Document editor)] 섹션에서 [편집(Edit)]을 선택한 다음 변경 템플릿의 JSON 또는 YAML 콘텐츠를 입력합니다.

   다음은 예입니다.

   > **Note**
   >
   > 파라미터 `minRequiredApprovals`은 지정된 레벨에서 이 템플릿을 사용하여 작성된 변경 요청을 승인해야 하는 검토자 수를 지정하는 데 사용됩니다.
   > 이 예에서는 2가지 수준의 승인을 보여줍니다. 최대 5개의 승인 수준을 지정할 수 있지만 한 수준만 필요합니다.
   > 첫 번째 수준에서는 특정 사용자 “John-Doe”가 각 변경 요청을 승인해야 합니다. 그런 다음 IAM 역할 `Admin`의 멤버 3명이 변경 요청을 승인해야 합니다.
   > 변경 템플릿 승인에 대한 자세한 내용은 변경 템플릿의 승인 정보 섹션을 참조하세요.

   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. [저장 후 미리 보기(Save and preview)]를 선택합니다.
7. 생성 중인 변경 템플릿의 세부정보를 검토합니다.

   검토를 위해 제출하기 전에 변경 템플릿을 변경하려면 [작업, 편집(Actions, Edit)]을 선택합니다.

   변경 템플릿의 내용에 만족하면 [검토를 위해 제출(Submit for review)]을 선택합니다. Change Manager의 [설정(Settings)] 탭에서 템플릿 검토자로 지정된 조직 또는 계정의 사용자는 새 변경 템플릿이 검토 보류 중이라는 알림을 받습니다.

   변경 템플릿에 대해 Amazon Simple Notification Service(Amazon SNS) 주제가 지정된 경우 변경 템플릿이 거부되거나 승인되면 알림이 전송됩니다. 이 변경 템플릿과 관련된 알림을 받지 못한 경우 나중에 Change Manager로 돌아가서 상태를 확인합니다.

명령줄 도구를 사용하여 변경 템플릿 생성

다음 절차에서는 AWS Command Line Interface(AWS CLI)(Linux, macOS 또는 Windows에서) 또는 AWS Tools for Windows PowerShell을 사용하여 AWS Systems Manager의 기능인 Change Manager에서 변경 요청을 생성하는 방법을 설명합니다.

변경 템플릿을 생성하려면

1. 아직 하지 않은 경우 AWS CLI 또는 AWS Tools for PowerShell을 설치하고 구성합니다.

자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트 및 AWS Tools for PowerShell 설치를 참조하세요.

2. 로컬 시스템에서 MyChangeTemplate.json이라는 JSON 파일을 생성한 다음 이 파일에 변경 템플릿의 내용을 붙여 넣습니다.

> **Note**
>
> 변경 템플릿은 Automation 실행서와 동일한 지원을 모두 포함하지 않는 스키마 0.3 버전을 사용합니다.

다음은 예입니다.

> **Note**
>
> 파라미터 minRequiredApprovals은 지정된 레벨에서 이 템플릿을 사용하여 작성된 변경 요청을 승인해야 하는 검토자 수를 지정하는 데 사용됩니다.
> 이 예에서는 2가지 수준의 승인을 보여줍니다. 최대 5개의 승인 수준을 지정할 수 있지만 한 수준만 필요합니다.
> 첫 번째 수준에서는 특정 사용자 “John-Doe”가 각 변경 요청을 승인해야 합니다. 그런 다음 IAM 역할 Admin의 멤버 3명이 변경 요청을 승인해야 합니다.
> 변경 템플릿 승인에 대한 자세한 내용은 변경 템플릿의 승인 정보 섹션을 참조하세요.

```
{
   "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. 다음 명령을 실행하여 변경 템플릿을 생성합니다.

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"
```

지정할 수 있는 기타 옵션에 대한 자세한 내용은 create-document 섹션을 참조하세요.

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "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":[

      ]
   }
}
```

Change Manager의 [설정(Settings)] 탭에서 템플릿 검토자로 지정된 조직 또는 계정의 사용자는 새 변경 템플릿이 검토 보류 중이라는 알림을 받습니다.

변경 템플릿에 대해 Amazon Simple Notification Service(Amazon SNS) 주제가 지정된 경우 변경 템플릿이 거부되거나 승인되면 알림이 전송됩니다. 이 변경 템플릿과 관련된 알림을 받지 못한 경우 나중에 Change Manager로 돌아가서 상태를 확인합니다.

변경 템플릿 검토 및 승인 또는 거부

AWS Systems Manager의 기능인 Change Manager에서 변경 템플릿에 대한 검토자로 지정된 경우 새 변경 템플릿 또는 변경 템플릿의 새 버전이 검토를 기다리고 있을 때 알림을 받습니다. Amazon Simple Notification Service(Amazon SNS) 주제에서 알림을 보냅니다.

> **Note**
>
> 이 기능은 계정이 Amazon SNS 주제를 사용하여 변경 템플릿 검토 알림을 보내도록 구성되었는지 여부에 따라 다릅니다. 템플릿 검토자 알림 주제 지정에 대한 자세한 내용은 태스크 1: Change Manager 사용자 자격 증명 관리 및 템플릿 검토자 구성 섹션을 참조하세요.

변경 템플릿을 검토하려면 알림의 링크를 따라 AWS Management Console에 로그인한 다음 이 절차의 단계를 따릅니다.

변경 템플릿을 검토하고 승인 또는 거부하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Change Manager를 선택합니다.
3. [개요(Overview)] 탭 하단에 있는 [변경 템플릿(Change templates)] 섹션의 [검토 보류 중(Pending review)]에서 숫자를 선택합니다.
4. [변경 템플릿(Change templates)] 목록에서 검토할 변경 템플릿의 이름을 찾아 선택합니다.
5. 요약 페이지에서 제안된 변경 템플릿 내용을 검토하고 다음 중 하나를 수행합니다.
   - 변경 요청에 사용할 수 있도록 변경 템플릿을 승인하려면 승인(Approve)을 선택합니다.
   - 변경 요청에 사용하지 못하도록 변경 템플릿을 거부하려면 거부(Reject)를 선택합니다.

변경 템플릿 삭제

이 주제에서는 Systems Manager의 기능인 Change Manager에서 생성한 템플릿을 삭제하는 방법을 설명합니다. Change Manager를 조직에서 사용 중인 경우 위임된 관리자 계정에서 이 절차가 수행됩니다.

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Change Manager를 선택합니다.
3. 템플릿 탭을 선택합니다.
4. 삭제할 템플릿의 이름을 선택합니다.
5. 작업(Actions)과 템플릿 삭제(Delete template)를 차례로 선택합니다.
6. 확인 대화 상자에 **DELETE** 글자를 입력한 다음 삭제(Delete)를 선택합니다.

변경 요청 작업

변경 요청은 AWS 또는 온프레미스 환경에서 하나 이상의 리소스를 업데이트하는 Automation 실행서를 실행하기 위한 Change Manager의 요청입니다. 변경 요청은 변경 템플릿을 사용하여 생성됩니다.

AWS Systems Manager의 기능인 Change Manager에서 변경 요청을 생성할 때 조직 또는 계정에 있는 한 명 이상의 승인자가 요청을 검토하고 승인해야 합니다. 필요한 승인이 없으면 요청한 변경을 수행하는 실행서 워크플로를 실행할 수 없습니다.

주제

- 변경 요청 생성

- 변경 요청 검토 및 승인 또는 거부

변경 요청 생성

AWS Systems Manager의 기능인 Change Manager으로 변경 요청을 생성할 때 선택하는 변경 템플릿은 일반적으로 다음을 수행합니다.

- 변경 요청에 대한 승인자를 지정하거나 필요한 승인 수를 지정합니다.
- 승인자에게 변경 요청에 대해 알리는 데 사용할 Amazon Simple Notification Service(Amazon SNS) 주제를 지정합니다.
- 변경 요청에 대한 실행서 워크플로를 모니터링하도록 Amazon CloudWatch 경보를 지정합니다.
- 요청된 변경을 수행하기 위해 선택할 수 있는 Automation 실행서를 식별합니다.

경우에 따라 사용할 Automation 실행서를 지정하고 요청을 검토 및 승인해야 하는 사용자를 지정하도록 변경 템플릿을 구성할 수 있습니다.

> **Important**
>
> 조직 전체에서 Change Manager를 사용하는 경우 항상 위임된 관리자 계정에서 변경하는 것이 좋습니다. 조직의 다른 계정에서 변경할 수 있지만 이러한 변경 사항은 위임된 관리자 계정에서 보고되거나 볼 수 없습니다.

주제

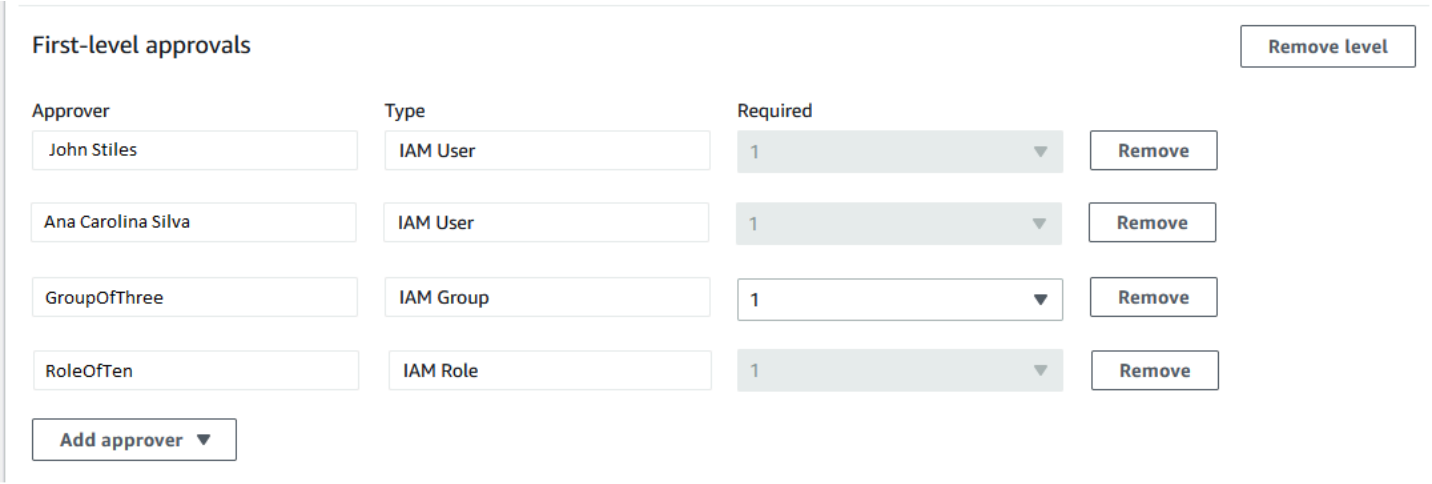
- 변경 요청 승인 정보
- 변경 요청 생성(콘솔)
- 변경 요청 생성(AWS CLI)

변경 요청 승인 정보

변경 템플릿에 지정된 요구 사항에 따라 변경 요청을 생성하는 경우 요청에 대한 런북 워크플로가 발생하기 전에 최대 5단계의 승인이 필요할 수 있습니다. 이러한 각 수준에 대해 템플릿 작성자는 최대 5명의 잠재적 승인자를 지정할 수 있습니다. 승인자는 단일 사용자로 제한되지 않습니다. 이러한 의미에서 승인자는 IAM 그룹 또는 IAM 역할일 수도 있습니다. IAM 그룹 및 IAM 역할의 경우 그룹 또는 역할에 속한 한 명 이상의 사용자가 변경 요청에 필요한 총 승인 수를 받기 위한 승인을 제공할 수 있습니다. 템플릿 작성자는 변경 템플릿에 필요한 것보다 더 많은 승인자를 지정할 수도 있습니다.

원본 승인 워크플로 및 업데이트 및/또는 승인

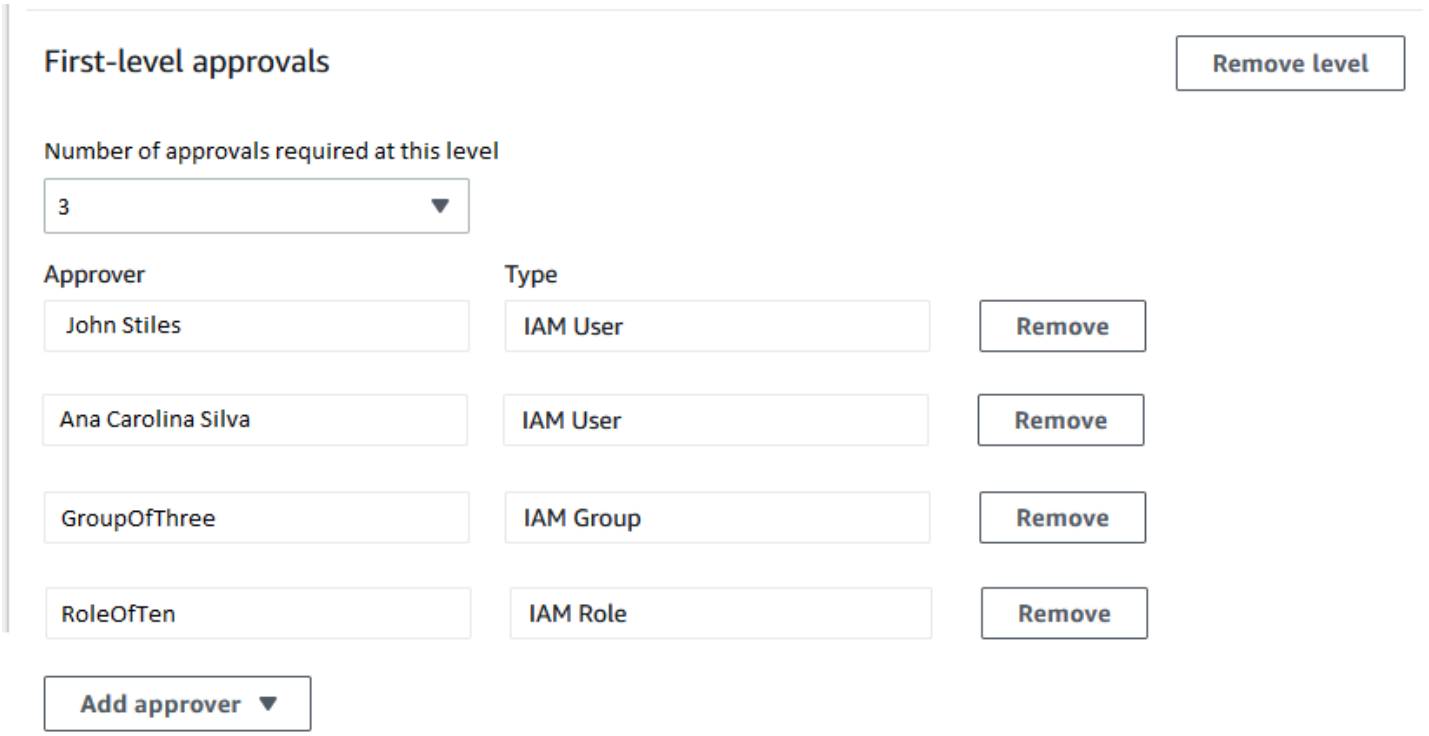
2023년 1월 23일 이전에 작성된 변경 템플릿을 사용하여 해당 수준에서 변경 요청 승인을 받으려면 지정된 각 승인자로부터 승인을 받아야 합니다. 예를 들어, 다음 이미지에 표시된 승인 수준 설정에서 4명의 승인자가 지정되어 있습니다. 지정된 승인자에는 2명의 사용자(John Stiles 및 Ana Carolina Silva), 3명의 멤버로 구성된 사용자 그룹(GroupOfThree) 및 10명의 사용자를 나타내는 사용자 역할(RoleOfTen)이 포함됩니다.



이 수준에서 변경 요청이 승인되려면 John Stiles, Ana Carolina Silva, `GroupOfThree` 그룹의 멤버 1명 및 `RoleOfTen` 역할의 멤버 1명의 승인을 받아야 합니다.

템플릿 작성자는 2023년 1월 23일 이후에 생성된 변경 템플릿을 사용하여 각 승인 수준에 대해 전체 필수 승인 수를 지정할 수 있습니다. 승인자로 지정된 임의의 사용자, 그룹 및 역할 조합으로부터 이러한 승인을 받을 수 있습니다. 변경 템플릿은 한 수준에 대해 단 하나의 승인만 필요로 할 수 있지만, 2명의 개별 사용자, 2개의 그룹 및 1개의 역할 등을 잠재적 승인자로 지정할 수 있습니다.

예를 들어, 다음 이미지에 표시된 승인 수준 영역에서는 3건의 승인이 필요합니다. 템플릿 지정 승인자에는 2명의 사용자(John Stiles 및 Ana Carolina Silva), 3명의 멤버로 구성된 사용자 그룹(`GroupOfThree`) 및 10명의 사용자를 나타내는 사용자 역할(`RoleOfTen`)이 포함됩니다.



`GroupOfThree` 그룹의 세 사용자 모두 변경 요청을 승인하면 해당 수준에 대해 승인됩니다. 각 사용자, 그룹 또는 역할에서 승인을 받을 필요는 없습니다. 임의의 잠재적 승인자 조합으로부터 최소 수의 승인을 받을 수 있습니다.

변경 요청이 생성되면 해당 수준에서 승인 알림에 대해 지정된 Amazon SNS 주제의 구독자에게 알림이 전송됩니다. 변경 템플릿 작성자가 사용해야 하는 알림 주제를 지정했거나 사용자가 지정하도록 허용했을 수 있습니다.

한 수준에서 필요한 최소 승인 수가 수신되면 다음 수준에 대한 Amazon SNS 주제를 구독하는 승인자에게 알림이 전송됩니다.

지정된 승인 수준 및 승인자 수에 관계없이 해당 요청에 대한 런북 워크플로가 발생하지 않도록 하려면 변경 요청에 대한 거부가 한 번만 필요합니다.

변경 요청 생성(콘솔)

다음 절차에서는 Systems Manager 콘솔을 사용하여 변경 요청을 생성하는 방법을 설명합니다.

변경 요청을 생성하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Change Manager를 선택합니다.

3. [요청 생성(Create request)]을 선택합니다.
4. 이 변경 요청에 사용할 변경 템플릿을 검색하고 선택합니다.
5. 다음을 선택합니다.
6. [이름(Name)]에 용도를 쉽게 식별할 수 있도록 변경 요청의 이름을 입력합니다(예: **`UpdateEC2LinuxAMI-us-east-2`**).
7. [실행서(Runbook)]에서 요청된 변경을 수행하는 데 사용할 실행서를 선택합니다.

> **Note**
>
> 실행서를 선택하는 옵션을 사용할 수 없는 경우 변경 템플릿 작성자가 사용해야 하는 실행서를 지정한 것입니다.

8. [변경 요청 정보(Change request information)]에서 검토자가 변경 요청을 승인할지 거부할지 결정하는 데 도움이 되도록 Markdown을 사용하여 변경 요청에 대한 추가 정보를 제공합니다. 사용 중인 템플릿의 작성자가 답변할 지침이나 질문을 제공했을 수 있습니다.

> **Note**
>
> Markdown은 문서와 문서 내의 개별 단계에 wiki 스타일의 설명을 추가할 수 있는 마크업 언어입니다. Markdown 사용에 대한 자세한 내용은 AWS에서 Markdown 사용을 참조하세요.

9. [워크플로 시작 시간(Workflow start time)] 섹션에서 다음 중 하나를 선택합니다.

   - [예약된 시간에 작업 실행(Run the operation at a scheduled time)] - [요청된 시작 시간(Requested start time)]에 이 요청에 대한 실행서 워크플로 실행을 위한 제안 날짜와 시간을 입력합니다. [예상 종료 시간(Estimated end time)]에 실행서 워크플로가 완료될 것으로 예상되는 날짜와 시간을 입력합니다. (이 시간은 검토자에게 제공하는 추정치일 뿐입니다.)

     > **Tip**
     >
     > [변경 일정 보기(View Change Calendar)]를 선택하여 지정한 시간에 차단 이벤트가 있는지 확인합니다.

   - [승인 후 최대한 빨리 작업 실행(Run the operation as soon as possible after approval)] – 변경 요청이 승인되면 변경을 수행할 수 있는 무제한 기간이 되는 즉시 실행서 워크플로가 실행됩니다.

10. [변경 요청 승인(Change request approvals)] 섹션에서 다음을 수행합니다.

    1. [승인 유형(Approval type)] 옵션이 표시되면 다음 중 하나를 선택합니다.

       - [자동 승인(Automatic approval)] – 선택한 변경 템플릿이 승인자의 검토 없이 변경 요청이 자동으로 실행되도록 구성되었습니다. 계속해서 11단계를 진행합니다.

         > **Note**
         >
         > Systems Manager 사용을 제어하는 IAM 정책에 지정된 권한이 자동 승인 변경 요청 제출을 제한하지 않아야 요청이 자동으로 실행됩니다.

       - [승인자 지정(Specify approvers)] - 이 변경 요청을 검토하고 승인하려면 하나 이상의 사용자, 그룹 또는 IAM 역할을 추가해야 합니다.

         > **Note**
         >
         > Systems Manager 사용을 제어하는 IAM 정책에 지정된 권한으로 자동 승인 변경 요청을 실행할 수 있는 경우에도 검토자를 지정하도록 선택할 수 있습니다.

    2. [승인자 추가(Add approver)]를 선택한 다음 사용 가능한 검토자 목록에서 하나 이상의 사용자, 그룹 또는 AWS Identity and Access Management(IAM) 역할을 선택합니다.

       > **Note**
       >
       > 하나 이상의 승인자가 이미 지정되었을 수 있습니다. 즉, 선택한 변경 템플릿에 필수 승인자가 이미 지정되어 있습니다. 요청에서 이러한 승인자를 제거할 수 없습니다. 승인자 추가 버튼을 사용할 수 없으면 선택한 템플릿에서 요청에 검토자 추가를 허용하지 않는 것입니다.

       변경 요청 승인에 대한 자세한 내용은 변경 요청 승인 정보 섹션을 참조하세요.

    3. [승인자에게 알릴 SNS 주제(SNS topic to notify approvers)]에서 다음 중 하나를 선택하여 이 변경 요청에 추가하려는 승인자에게 알림을 보내는 데 사용할 계정의 Amazon SNS 주제를 지정합니다.

> **Note**
>
> Amazon SNS 주제를 지정하는 옵션을 사용할 수 없는 경우 선택한 변경 템플릿이 사용할 Amazon SNS 주제를 이미 지정하고 있는 것입니다.

- [SNS Amazon 리소스 이름(ARN) 입력(Enter an SNS Amazon Resource Name (ARN))] - [주제 ARN(Topic ARN)]에 기존 Amazon SNS 주제의 ARN을 입력합니다. 이 주제는 조직의 모든 계정에 있을 수 있습니다.
- [기존 SNS 주제 선택(Select an existing SNS topic)] - [대상 알림 주제(Target notification topic)]에서 현재 계정의 기존 Amazon SNS 주제의 ARN을 선택합니다. (현재 AWS 계정 및 AWS 리전에서 Amazon SNS 주제를 아직 생성하지 않은 경우 이 옵션을 사용할 수 없습니다.)

> **Note**
>
> 선택한 Amazon SNS 주제는 전송되는 알림과 알림이 전송되는 구독자를 지정하도록 구성되어야 합니다. Change Manager에서 알림을 보낼 수 있도록 액세스 정책도 Systems Manager에 권한을 부여해야 합니다. 자세한 내용은 Change Manager 알림에 대한 Amazon SNS 주제 구성 섹션을 참조하세요.

4. 알림 추가를 선택합니다.

11. 다음을 선택합니다.
12. [IAM 역할(IAM role)]에서 이 변경 요청에 대해 지정된 실행서를 을 실행하는 데 필요한 권한이 있는 현재 계정의 IAM 역할을 선택합니다.

    이 역할을 Automation의 서비스 역할 또는 수임 역할이라고도 합니다. 이에 대한 자세한 내용은 Automation 설정 섹션을 참조하세요.
13. [배포 위치(Deployment location)] 섹션에서 다음 중 하나를 선택합니다.

> **Note**
>
> Change Manager를 단일 AWS 계정에만 사용하고 AWS Organizations에 설정된 조직에는 사용하지 않는 경우 배포 위치를 지정할 필요가 없습니다.

- [이 계정에 변경 사항 적용(Apply change to this account)] - 실행서 워크플로가 현재 계정에서만 실행됩니다. 조직의 경우 이는 위임된 관리자 계정을 의미합니다.
- [여러 조직 단위(OU) 에 변경 사항 적용(Apply change to multiple organizational units (OUs))] - 다음을 수행합니다.

  1. [계정 및 조직 단위(OU)(Accounts and organizational units (OUs))]에 조직의 멤버 계정 ID를 **123456789012** 형식으로 입력하거나 조직 단위의 ID를 **o-o96EXAMPLE** 형식으로 입력합니다.
  2. (옵션) [실행 역할 이름(Execution role name)]에 이 변경 요청에 대해 지정된 실행서를 실행하는 데 필요한 권한이 있는 대상 계정 또는 OU의 IAM 역할 이름을 입력합니다. 지정하는 OU의 모든 계정이 이 역할에 대해 동일한 이름을 사용해야 합니다.
  3. (옵션) 지정하려는 각 추가 계정 또는 OU에 대해 [다른 대상 위치 추가(Add another target location)]를 선택하고 단계 a와 b를 반복합니다.
  4. 대상 AWS 리전에서 변경할 리전을 선택합니다. 예를 들어 미국 동부(오하이오) 리전의 경우 `Ohio (us-east-2)`을 선택합니다.
  5. [속도 제어(Rate control)]를 확장합니다.

     [동시성(Concurrency)]에서 숫자를 입력한 다음 목록에서 이것이 실행서 워크플로를 동시에 실행할 수 있는 계정의 수를 나타내는지 아니면 백분율을 나타내는지 선택합니다.

     [오류 임계값(Error threshold)]에 숫자를 입력한 다음 목록에서 이것이 작업이 중지되기 전에 실행서 워크플로가 실패할 수 있는 계정의 수를 나타내는지 아니면 백분율을 나타내는지 선택합니다.

14. [배포 대상(Deployment targets)] 섹션에서 다음을 수행합니다.

    1. 다음 중 하나를 선택합니다.
       - [단일 리소스(Single resource)] - 한 리소스에 대해서만 변경이 수행됩니다. 예를 들어 이 변경 요청에 대한 런북에 정의된 작업에 따라 단일 노드 또는 단일 Amazon Machine Image(AMI)입니다.
       - [여러 리소스(Multiple resources)] - [파라미터(Parameter)]에서 이 변경 요청에 대한 실행서에서 사용 가능한 파라미터 중에서 선택합니다. 이 선택은 업데이트 중인 리소스의 유형을 반영합니다.

         예를 들어 이 변경 요청에 대한 실행서가 `AWS-RetartEC2Instance`이면 `InstanceId`를 선택한 후 다음 중에서 선택하여 업데이트할 인스턴스를 정의할 수 있습니다.

- [태그 지정(Specify tags)] - 업데이트할 모든 리소스에 태그가 지정되는 키-값 페어를 입력합니다.
- [리소스 그룹 선택(Choose a resource group)] - 업데이트할 모든 리소스가 속한 리소스 그룹의 이름을 선택합니다.
- [파라미터 값 지정(Specify parameter values)] - [실행서 파라미터(Runbook parameters)] 섹션에서 업데이트할 리소스를 식별합니다.
- 모든 인스턴스 대상(Target all instances) - 대상 위치의 모든 관리형 노드를 변경합니다.

2. [여러 리소스(Multiple resources)]를 선택한 경우 [속도 제어(Rate control)]를 확장합니다.

   [동시성(Concurrency)]에서 숫자를 입력한 다음 목록에서 이것이 실행서 워크플로가 동시에 업데이트할 수 있는 대상의 수를 나타내는지 아니면 백분율을 나타내는지 선택합니다.

   [오류 임계값(Error threshold)]에 숫자를 입력한 다음 목록에서 이것이 작업이 중지되기 전에 업데이트가 실패할 수 있는 대상의 수를 나타내는지 아니면 백분율을 나타내는지 선택합니다.

15. 이전 단계에서 여러 리소스를 업데이트하기 위해 [파라미터 값 지정(Specify parameter values)]을 선택한 경우 [실행서 파라미터(Runbook parameters)] 섹션에서 필수 입력 파라미터의 값을 지정합니다. 제공해야 하는 파라미터 값은 선택한 변경 템플릿과 연결된 Automation 실행서의 내용을 기반으로 합니다.

    예를 들어 변경 템플릿이 `AWS-RetartEC2Instance` 실행서를 사용하는 경우 InstanceId 파라미터에 대해 하나 이상의 인스턴스 ID를 입력해야 합니다. 또는 [대화형 인스턴스 선택기 표시(Show interactive instance picker)]를 선택하고 사용 가능한 인스턴스를 하나씩 선택합니다.

16. 다음을 선택합니다.

17. [검토 후 제출(Review and submit)] 페이지에서 이 변경 요청에 대해 지정한 리소스와 옵션을 다시 확인합니다.

    변경하려는 섹션에 대해 [편집(Edit)] 버튼을 선택합니다.

    변경 요청 세부 정보에 만족하면 [승인을 위해 제출(Submit for approval)]을 선택합니다.

요청에 대해 선택한 변경 템플릿에 Amazon SNS 주제가 지정된 경우 요청이 거부되거나 승인되면 알림이 전송됩니다. 요청에 대한 알림을 받지 못한 경우 Change Manager로 돌아가 요청 상태를 확인할 수 있습니다.

변경 요청 생성(AWS CLI)

AWS Command Line Interface(AWS CLI)를 사용하여 JSON 파일에 변경 요청에 대한 옵션 및 파라미터를 지정하고, 이를 명령에 포함시키는 `--cli-input-json` 옵션을 사용하여 변경 요청을 생성할 수 있습니다.

변경 요청을 생성하려면(AWS CLI)

1. 아직 하지 않은 경우 AWS CLI 또는 AWS Tools for PowerShell을 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트 및 AWS Tools for PowerShell 설치를 참조하세요.

2. 로컬 시스템에서 `MyChangeRequest.json`이라는 JSON 파일을 생성한 다음 이 파일에 다음 내용을 붙여넣습니다.

   `## ###`를 변경 요청에 대한 값으로 바꿉니다.

   > **Note**
   >
   > 이 샘플 JSON은 `AWS-HelloWorldChangeTemplate` 변경 요청 및 `AWS-HelloWorld` 실행서를 사용하여 변경 요청을 생성합니다. 이 샘플을 본인의 변경 요청에 맞게 조정하려면 AWS Systems Manager API 참조의 StartChangeRequestExecution에서 사용 가능한 모든 파라미터에 대한 정보를 참조하세요.
   > 변경 요청 승인에 대한 자세한 내용은 변경 요청 승인 정보 섹션을 참조하세요.

```
{
    "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. JSON 파일을 생성한 디렉터리에서 다음 명령을 실행합니다.

```
aws ssm start-change-request-execution --cli-input-json file://MyChangeRequest.json
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "AutomationExecutionId": "b3c1357a-5756-4839-8617-2d2a4EXAMPLE"
}
```

변경 요청 검토 및 승인 또는 거부

AWS Systems Manager의 기능인 Change Manager에서 변경 요청에 대한 검토자로 지정된 경우 새 변경 요청이 검토 대기 중일 때 Amazon Simple Notification Service(Amazon SNS) 주제를 통해 알림을 받습니다.

> **Note**
>
> 이 기능은 Amazon SNS가 검토 알림을 보내기 위한 변경 템플릿에 지정되었는지 여부에 따라 다릅니다. 자세한 설명은 Change Manager 알림에 대한 Amazon SNS 주제 구성을 참조하세요.

변경 요청을 검토하려면 알림의 링크를 따라 AWS Management Console에 바로 로그인하거나 이 절차의 단계를 따릅니다.

> **Note**
>
> Amazon SNS 주제가 변경 템플릿의 검토자에게 할당된 경우 변경 요청의 상태가 변경되면 주제의 구독자에게 알림이 전송됩니다.
> 변경 요청 승인에 대한 자세한 내용은 변경 요청 승인 정보 섹션을 참조하세요.

변경 요청 검토 및 승인 또는 거부(콘솔)

다음 절차에서는 Systems Manager 콘솔을 사용하여 변경 요청을 검토하고 승인하거나 거부하는 방법을 설명합니다.

단일 변경 요청을 검토하고 승인 또는 거부하려면

1. 받은 이메일 알림의 링크를 열고 AWS Management Console에 로그인하면 검토를 위한 변경 요청으로 이동합니다.
2. 요약 페이지에서 제안된 변경 요청 내용을 검토합니다.

   변경 요청을 승인하려면 [승인(Approve)]을 선택합니다. 대화 상자에서 이 승인에 대해 추가할 설명을 제공한 다음 [승인(Approve)]을 선택합니다. 이 요청이 나타내는 실행서 워크플로는 예약된 경우 또는 제한 사항으로 인해 변경 사항이 차단되지 않는 즉시 실행을 시작합니다.

   -또는-

변경 요청을 거부하려면 거부(Reject)를 선택합니다. 대화 상자에서 이 거부에 대해 추가할 설명을 제공한 다음 거부(Reject)를 선택합니다.

변경 요청을 일괄 검토하고 승인 또는 거부하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Change Manager를 선택합니다.
3. 승인(Approvals) 탭을 선택합니다.
4. (선택 사항) 각 요청의 이름을 선택하여 승인 대기 중인 요청의 세부 정보를 검토한 다음 승인(Approvals) 탭으로 돌아갑니다.
5. 승인할 각 변경 요청의 확인란을 선택합니다.

   -또는-

   거부할 각 변경 요청의 확인란을 선택합니다.
6. 대화 상자에서 이 승인 또는 거부에 대해 추가할 설명을 제공합니다.
7. 선택한 변경 요청을 승인 또는 거부하는지 여부에 따라 승인(Approve) 또는 거부(Reject)를 선택합니다.

변경 요청 검토 및 승인 또는 거부(명령줄)

다음 절차에서는 AWS Command Line Interface(AWS CLI)(Linux, macOS 또는 Windows에서)를 사용하여 변경 요청을 검토하고 승인 또는 거부하는 방법을 설명합니다.

변경 요청을 검토하고 승인 또는 거부하려면

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.
2. AWS CLI 호출에 대한 파라미터를 지정하는 JSON 파일을 로컬 시스템에 생성합니다.

```
{
  "OpsItemFilters":
  [
    {
      "Key": "OpsItemType",
      "Values": ["/aws/changerequest"],
```

```
      "Operator": "Equal"
    }
  ],
  "MaxResults": number
}
```

JSON 파일에서 승인자의 Amazon 리소스 이름(ARN)을 지정하여 특정 승인자의 결과를 필터링할 수 있습니다. 다음 예를 참고하세요

```
{
  "OpsItemFilters":
  [
    {
      "Key": "OpsItemType",
      "Values": ["/aws/changerequest"],
      "Operator": "Equal"
    },
    {
      "Key": "ChangeRequestByApproverArn",
      "Values": ["arn:aws:iam::account-id:user/user-name"],
      "Operator": "Equal"
    }
  ],
  "MaxResults": number
}
```

3. 다음 명령을 실행하여 JSON 파일에 지정한 최대 변경 요청 수를 확인합니다.

   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. 다음 명령을 실행하여 변경 요청을 승인하거나 거부합니다.

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"
```

요청에 대해 선택한 변경 템플릿에 Amazon SNS 주제가 지정된 경우 요청이 거부되거나 승인되면 알림이 전송됩니다. 요청에 대한 알림을 받지 못한 경우 Change Manager로 돌아가 요청 상태를 확인할 수 있습니다. 이 명령을 사용할 때 다른 옵션에 대한 자세한 내용은 AWS CLI 명령 레퍼런스의 AWS Systems Manager 섹션에서 send-automation-signal 항목을 참조하세요.

변경 요청 세부 정보, 태스크 및 타임라인 검토(콘솔)

AWS Systems Manager의 기능인 Change Manager의 대시보드에서 이미 변경이 처리된 요청을 포함하여 변경 요청에 대한 정보를 볼 수 있습니다. 이러한 세부 정보에는 변경을 수행하는 실행서를 실행하는 Automation 작업에 대한 링크가 포함됩니다. 요청이 생성될 때 Automation 실행 ID가 생성되지만 모든 승인이 제공되고 변경을 차단하기 위한 제한이 없을 때까지 프로세스가 실행되지 않습니다.

변경 요청 세부 정보, 태스크 및 타임라인을 검토하려면

1. 탐색 창에서 Change Manager를 선택합니다.
2. [요청(Requests)] 탭을 선택합니다.
3. [변경 요청(Change requests)] 섹션에서 검토하려는 변경 요청을 검색합니다.

   [날짜 범위 생성(Create date range)] 옵션을 사용하여 결과를 특정 기간으로 제한할 수 있습니다.

   다음 속성을 사용하여 요청을 필터링할 수 있습니다.

   - `Status`

- `Request ID`
- `Approver`
- `Requester`

예를 들어 지난 24시간 동안 성공적으로 완료된 모든 변경 요청에 대한 세부 정보를 보려면 다음을 수행합니다.

1. [날짜 범위 생성(Create date range)]에서 [1d]를 선택합니다.
2. 검색 상자에서 [상태, CompletedWithSuccess(Status, CompletedWithSuccess)]를 선택합니다.
3. 결과에서 결과를 검토할 성공적으로 완료된 변경 요청의 이름을 선택합니다.

4. 다음 탭에서 변경 요청에 대한 정보를 봅니다.

- [요청 세부 정보(Request details)] - 요청자, 변경 템플릿 및 변경에 대해 선택된 Automation 실행서를 포함하여 변경 요청에 대한 기본 세부 정보를 봅니다. Automation 작업 세부 정보에 대한 링크를 따라가서 요청에 지정된 실행서 파라미터, 변경 요청에 할당된 Amazon CloudWatch 경보, 요청에 대해 제공된 승인 및 설명에 대한 정보를 볼 수도 있습니다.
- [태스크(Task)] - 완료된 변경 요청의 태스크 상태, 대상 리소스, 연결된 Automation 실행서의 단계, 동시성 및 오류 임계값 세부 정보를 포함하여 변경 태스크에 대한 정보를 봅니다.
- [타임라인(Timeline)] - 변경 요청과 연관된 모든 이벤트의 요약(날짜 및 시간별로 나열됨)을 봅니다. 요약에는 변경 요청이 생성된 시기, 배정된 승인자의 작업, 승인된 변경 요청이 실행되도록 예약된 시기 메모, 실행서 워크플로 세부 정보, 전체 변경 프로세스 및 실행서의 각 단계에 대한 상태 변경 사항이 표시됩니다.
- 관련 이벤트 - AWS CloudTrail Lake에 기록된 변경 요청에 대한 감사 가능한 세부 정보를 볼 수 있습니다. 세부 정보에는 실행된 API 작업, 해당 작업에 포함된 요청 파라미터, 작업을 실행한 사용자 계정, 프로세스 중에 업데이트된 리소스 등이 포함됩니다.

  CloudTrail Lake 이벤트 추적을 활성화하면 CloudTrail Lake가 변경 요청과 관련한 이벤트에 사용할 이벤트 데이터 스토어를 생성합니다. 이벤트 세부 정보는 변경 요청이 실행된 계정 또는 조직에서 사용할 수 있습니다. 계정 또는 조직의 모든 변경 요청에서 CloudTrail Lake 이벤트 추적을 활성화할 수 있습니다. CloudTrail Lake 통합 활성화 및 이벤트 데이터 스토어 생성에 대한 자세한 내용은 변경 요청 이벤트 모니터링 섹션을 참조하세요.

> **Note**
>
> CloudTrail Lake를 사용하는 데 따른 요금이 부과됩니다. 자세한 내용은 AWS CloudTrail 요금을 참조하십시오.

변경 요청의 집계된 수 보기(명령줄)

GetOpsSummary API 작업을 사용하여 AWS Systems Manager의 기능인 Change Manager에서 집계된 변경 요청 수를 볼 수 있습니다. 이 API 작업은 단일 AWS 리전의 단일 AWS 계정 또는 여러 계정 및 여러 리전에 대한 수를 반환할 수 있습니다.

> **Note**
>
> 여러 AWS 계정와 여러 AWS 리전에 대한 집계된 변경 요청 수를 보려면 리소스 데이터 동기화를 설정하고 구성해야 합니다. 자세한 내용은 Inventory의 리소스 데이터 동기화 구성 단원을 참조하십시오.

다음 절차에서는 AWS Command Line Interface(AWS CLI)(Linux, macOS 또는 Windows에서)를 사용하여 집계된 변경 요청 수를 보는 방법을 설명합니다.

집계된 변경 요청 수를 보려면

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.
2. 다음 명령 중 하나를 실행합니다.

   단일 계정 및 리전

   이 명령은 AWS CLI 세션이 구성된 AWS 계정 및 AWS 리전에 대한 모든 변경 요청 수를 반환합니다.

   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
```

호출은 다음과 같은 정보를 반환합니다.

```
{
    "Entities": [
        {
            "Data": {
                "AWS:OpsItem": {
                    "Content": [
                        {
                            "Count": "38",
                            "Status": "Open"
                        }
                    ]
                }
            }
        }
    ]
}
```

여러 계정 및/또는 리전

이 명령은 리소스 데이터 동기화에 지정된 AWS 계정 및 AWS 리전에 대한 모든 변경 요청 수를 반환합니다.

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
```

호출은 다음과 같은 정보를 반환합니다.

```
{
    "Entities": [
        {
            "Data": {
                "AWS:OpsItem": {
                    "Content": [
                        {
                            "Count": "43",
                            "Status": "Open"
                        },
                        {
                            "Count": "2",
                            "Status": "Resolved"
                        }
                    ]
                }
            }
        }
    ]
}
```

여러 계정 및 특정 리전

이 명령은 리소스 데이터 동기화에 지정된 AWS 계정에 대한 모든 변경 요청 수를 반환합니다. 그러나 명령에 지정된 리전의 데이터만 반환합니다.

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
```

출력이 리전별로 그룹화된 여러 계정 및 리전

이 명령은 리소스 데이터 동기화에 지정된 AWS 계정 및 AWS 리전에 대한 모든 변경 요청 수를 반환합니다. 출력은 리전별 개수 정보를 표시합니다.

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"}]}]'
```

호출은 다음과 같은 정보를 반환합니다.

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

출력이 계정 및 리전별로 그룹화된 여러 계정 및 리전

이 명령은 리소스 데이터 동기화에 지정된 AWS 계정 및 AWS 리전에 대한 모든 변경 요청 수를 반환합니다. 출력은 계정 및 리전별로 개수 정보를 그룹화합니다.

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"}]}]}]
```

호출은 다음과 같은 정보를 반환합니다.

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

Change Manager 활동 감사 및 로깅

Amazon CloudWatch 및 AWS CloudTrail 경보를 사용하여 AWS Systems Manager의 기능인 Change Manager에서 활동을 감사할 수 있습니다.

Systems Manager에 대한 자세한 감사 및 로깅 옵션은 AWS Systems Manager 모니터링 섹션을 참조하세요.

CloudWatch 경보를 사용하여 Change Manager 활동을 감사합니다.

CloudWatch 경보를 구성하고 변경 템플릿에 할당할 수 있습니다. 경보에 정의된 조건이 충족되면 경보에 대해 지정된 작업이 수행됩니다. 경보 구성에서 경보 조건이 충족될 때 알리도록 Amazon Simple Notification Service(Amazon SNS) 주제를 지정할 수 있습니다.

Change Manager 템플릿 생성에 대한 자세한 내용은 변경 템플릿 사용 섹션을 참조하세요.

CloudWatch 경보 생성에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 Amazon CloudWatch 경보 사용을 참조하세요.

CloudTrail을 사용하여 Change Manager 활동 감사

CloudTrail에서는 Systems Manager 콘솔, AWS Command Line Interface(AWS CLI) 및 Systems Manager SDK에서 수행된 API 호출을 캡처합니다. 정보가 저장된 CloudTrail 콘솔 또는 Amazon Simple Storage Service(Amazon S3) 버킷에서 정보를 볼 수 있습니다. 계정에 대한 모든 CloudTrail 로그를 저장하는 데 버킷 하나가 사용됩니다.

Change Manager 작업의 로그에는 변경 템플릿 문서 생성, 변경 템플릿 및 변경 요청 승인 및 거부, Automation 실행서에서 생성된 활동 등이 표시됩니다. Systems Manager 활동의 CloudTrail 로그 보기 및 사용에 대한 자세한 내용은 AWS CloudTrail을 사용하여 AWS Systems ManagerAPI 호출을 로깅 섹션을 참조하세요.

Change Manager 문제 해결

다음 정보를 사용하면 AWS Systems Manager의 기능인 Change Manager 관련 문제를 해결하는 데 도움이 됩니다.

주제

- Active Directory 사용 시 변경 요청 승인 중 "그룹 {GUID}를 찾을 수 없음(Group {GUID} not found)" 오류 발생(그룹)

Active Directory 사용 시 변경 요청 승인 중 "그룹 ***{GUID}***를 찾을 수 없음(Group {GUID} not found)" 오류 발생(그룹)

문제: AWS IAM Identity Center(IAM Identity Center)이(가) 사용자 자격 증명 관리에 사용되는 경우 Change Manager에서 승인 권한이 부여된 Active Directory 그룹의 멤버가 "권한이 부여되지 않음(not authorized)" 또는 "그룹을 찾을 수 없음(group not found)" 오류를 수신합니다.

- 해결 방법: AWS Management Console에 액세스하기 위해 IAM Identity Center에서 Active Directory 그룹을 선택하면 시스템은 해당 Active Directory 그룹의 정보를 IAM Identity Center로 복사하는 주기적 동기화를 예약합니다. 이 프로세스를 완료해야 Active Directory 그룹 멤버십을 통해 권한이 부여된 사용자가 요청을 승인할 수 있습니다. 자세한 내용은 AWS IAM Identity Center 사용 설명서의 Microsoft AD 디렉터리에 연결을 참조하세요.

AWS Systems Manager 자동화

AWS Systems Manager의 기능인 Automation은 Amazon Elastic Compute Cloud(Amazon EC2), Amazon Relational Database Service(Amazon RDS), Amazon Redshift, Amazon Simple Storage

Service(Amazon S3) 등의 AWS 서비스에 대한 일반적인 유지 관리, 배포 및 수정 태스크를 단순화합니다. 자동화를 시작하려면 Systems Manager 콘솔을 엽니다. 왼쪽 탐색 창에서 Automation을 선택합니다.

Automation은 AWS 리소스를 대규모로 배포, 구성 및 관리하는 자동화된 솔루션을 구축하는 데 도움이 됩니다. Automation을 사용하면 자동화의 동시성을 세부적으로 제어할 수 있습니다 즉, 동시에 대상으로 지정할 리소스 수와 자동화가 중지되기 전에 발생할 수 있는 오류 수를 지정할 수 있습니다.

Automation을 시작하는 데 도움이 되도록 AWS는 미리 정의된 몇 가지 실행서를 개발하고 유지 관리합니다. 사용 사례에 따라 다양한 태스크를 수행하는 이러한 미리 정의된 실행서를 사용하거나 요구 사항에 더 잘 맞는 사용자 정의 실행서를 직접 생성할 수 있습니다. 자동화의 진행률과 상태를 모니터링하기 위해 Systems Manager Automation 콘솔 또는 선호하는 명령줄 도구를 사용할 수 있습니다. 또한 Automation은 Amazon EventBridge와 통합되어 대규모 이벤트 기반 아키텍처를 구축하는 데 도움이 됩니다.

Automation이 조직에 주는 이점은 무엇인가요?

Automation은 다음과 같은 이점을 제공합니다.

- 실행서 콘텐츠의 스크립팅 지원

  `aws:executeScript` 작업을 사용하면 실행서에서 직접 사용자 정의 Python 및 PowerShell 함수를 실행할 수 있습니다. 이렇게 하면 다른 Automation 작업이 지원하지 않는 다양한 태스크를 완료할 수 있으므로 더 유연하게 사용자 정의 실행서를 생성할 수 있습니다. 또한 실행서의 논리를 더 잘 제어할 수 있습니다. 이 작업을 사용하는 방법과 기존 자동화 솔루션을 개선하는 데 도움이 되는 방법의 예는 Automation 실행서 작성 섹션을 참조하세요.
- 중앙 위치에서 여러 AWS 계정 및 AWS 리전에 걸쳐 자동화 실행

  관리자는 Systems Manager 콘솔에서 여러 계정 및 리전에 걸쳐 리소스에 대한 자동화를 실행할 수 있습니다.
- 강화된 작업 보안

  관리자가 중앙 위치에서 실행서에 대한 액세스 권한을 부여하고 취소할 수 있습니다. AWS Identity and Access Management(IAM) 정책만 사용하는 경우에는 Automation을 사용할 수 있는 조직 내 개별 사용자 또는 그룹과 이들이 액세스할 수 있는 실행서를 제어할 수 있습니다.
- 일반 IT 태스크 자동화

  일반 태스크를 자동화하면 운영 효율성을 개선하고 조직 표준을 시행하고 운영자 오류를 줄이는 데 도움이 될 수 있습니다. 예를 들어 `AWS-UpdateCloudFormationStackWithApproval` 실행서를

사용하여 AWS CloudFormation 템플릿으로 배포된 리소스를 업데이트할 수 있습니다. 이 업데이트는 새 템플릿에 적용됩니다. 업데이트가 시작되기 전에 사용자 한 명 이상의 승인을 요청하도록 자동화를 구성할 수 있습니다.

- 중단 작업을 일괄적으로 안전하게 수행

  자동화에는 동시성 값과 오류 임계값을 지정하여 플릿에서 자동화 배포를 제어할 수 있는 속도 제어와 같은 기능이 포함됩니다. 속도 제어 작업에 대한 자세한 내용은 대규모로 자동화 실행 섹션을 참조하세요.

- 복잡한 태스크 간소화

  Automation은 골든 Amazon Machine Images(AMIs) 생성과 같이 복잡하고 시간 소모적인 태스크를 간소화하는 사전 정의된 실행서를 제공합니다. 예를 들어 `AWS-UpdateLinuxAmi` 및 `AWS-UpdateWindowsAmi` 실행서를 사용하여 소스 AMI에서 골든 AMIs를 생성할 수 있습니다. 이러한 실행서를 사용하여 업데이트가 적용되기 전과 후에 사용자 정의 스크립트를 실행할 수 있습니다. 또한 특정 소프트웨어 패키지를 포함하거나 설치에서 제외할 수 있습니다. 이러한 실행서를 사용하는 방법의 예제는 튜토리얼 섹션을 참조하세요.

- 입력에 대한 제약 조건 정의

  사용자 정의 실행서에 제약 조건을 정의하여 Automation이 특정 입력 파라미터에 대해 허용하는 값을 제한할 수 있습니다. 예를 들어, `allowedPattern`은 정의하는 정규식과 일치하는 입력 파라미터의 값만 허용합니다. 입력 파라미터에 대해 `allowedValues`를 지정하면 실행서에 지정한 값만 허용됩니다.

- Amazon CloudWatch Logs에 자동화 작업 출력 로그

  조직의 운영 또는 보안 요구 사항을 충족하기 위해 실행서 중 실행된 스크립트의 레코드를 제공해야 할 수 있습니다. CloudWatch Logs를 사용하면 다양한 AWS 서비스의 로그 파일을 모니터링, 저장 및 액세스할 수 있습니다. 디버깅 및 문제 해결을 위해 `aws:executeScript` 작업의 출력을 CloudWatch Logs 로그 그룹으로 전송할 수 있습니다. 로그 데이터는 KMS 키를 사용하여 AWS KMS 암호화를 적용하거나 적용하지 않은 상태로 로그 그룹으로 스트리밍할 수 있습니다. 자세한 내용은 CloudWatch Logs로 Automation 작업 출력 로깅 단원을 참조하십시오.

- Amazon EventBridge 통합

  Automation은 Amazon EventBridge 규칙의 대상 유형으로 지원됩니다. 즉, 이벤트를 사용하여 실행서를 트리거할 수 있습니다. 자세한 내용은 Amazon EventBridge로 Systems Manager 이벤트 모니터링 및 참조: Systems Manager용 Amazon EventBridge 이벤트 패턴 및 유형 단원을 참조하십시오.

- 조직의 모범 사례 공유

계정 및 리전 간에 공유하는 실행서에서 리소스 관리, 운영 태스크 등에 대한 모범 사례를 정의할 수 있습니다.

Automation은 누가 사용해야 하나요?

- 규모에 맞게 운영 효율성을 개선하고, 수동 개입과 관련된 오류를 줄이고, 일반적인 문제 해결 시간을 단축하고자 하는 AWS 고객
- 배포 및 구성 태스크를 자동화하려는 인프라 전문가
- 일반적인 문제를 안정적으로 해결하고, 문제 해결 효율성을 높이고, 반복 작업을 줄이려는 관리자
- 일반적으로 수동으로 수행하는 태스크를 자동화하려는 사용자

자동화란 무엇인가요?

자동화는 실행서에 정의되고 Automation 서비스에서 수행되는 모든 태스크로 구성됩니다. Automation은 다음 구성 요소를 사용하여 자동화를 실행합니다.

개념	Details
Automation 실행서	Systems Manager Automation 런북은 자동화(Systems Manager가 관리형 노드 및 AWS 리소스에 대해 수행하는 작업)를 정의합니다. Automation에는 사전 정의된 실행서가 몇 가지 포함되며, 이들 실행서를 사용하여 Amazon EC2 인스턴스를 하나 이상 다시 시작하거나 Amazon Machine Image(AMI)를 생성하는 등 일반적인 태스크를 수행할 수 있습니다. 사용자 고유의 실행서를 생성할 수도 있습니다. 실행서는 YAML 또는 JSON을 사용하며, 사용자가 지정하는 단계와 파라미터를 포함합니다. 단계는 순차적으로 실행됩니다. 자세한 내용은 사용자 런북 생성 단원을 참조하십시오. 실행서는 `Command`, `Policy`, `Session` 문서와 다른 `Automation` 유형의 Systems Manager 문서입니다. 실행서는 스키마 버전 0.3을 지원합

개념	Details
	니다. 명령 문서에는 스키마 버전 1.2, 2.0 또는 2.2가 사용됩니다. 정책 문서에는 스키마 버전 2.0 이상이 사용됩니다.
자동화 작업	실행서에 정의된 자동화에는 하나 이상의 단계가 포함됩니다. 각 단계는 특정 작업과 관련되어 있습니다. 작업은 해당 단계의 입력, 동작 및 출력을 결정합니다. 단계는 실행서의 `mainSteps` 섹션에 정의됩니다. 자동화는 20개의 고유한 작업 유형을 지원합니다. 자세한 내용은 Systems Manager Automation 작업 참조 단원을 참조하십시오.
Automation 할당량	각 AWS 계정는 100개의 자동화를 동시에 실행할 수 있습니다. 여기에는 하위 자동화(다른 자동화에 의해 시작된 자동화) 및 속도 제어 자동화가 포함됩니다. 이보다 많은 자동화를 실행하려고 하면 Systems Manager가 대기열에 추가 자동화를 추가하고 보류 중(Pending) 상태를 표시합니다. 적응형 동시성을 사용하여 이 할당량을 조정할 수 있습니다. 자세한 내용은 Automation이 동시성 요구 사항에 따라 조정되도록 허용 섹션을 참조하세요. 자동화 실행에 대한 자세한 내용은 자동화 실행 섹션을 참조하세요.
Automation 대기열 할당량	동시 자동화 제한보다 많은 자동화를 실행하려고 하면 후속 자동화가 대기열에 추가됩니다. AWS 계정마다 5,000개의 자동화를 대기열에 넣을 수 있습니다. 자동화가 완료되거나 종료 상태에 도달하면 대기열의 첫 번째 자동화가 시작됩니다.

개념	Details
속도 제어 자동화 할당량	각 AWS 계정는 25개의 속도 제어 자동화를 동시에 실행할 수 있습니다. 동시 속도 제어 자동화 제한보다 더 많은 속도 제어 자동화를 실행하려고 하면 Systems Manager가 후속 속도 제어 자동화를 대기열에 추가하고 보류 중(Pending) 상태를 표시합니다. 속도 제어 자동화 실행에 대한 자세한 내용은 대규모로 자동화 실행 섹션을 참조하세요.
속도 제어 자동화 대기열 할당량	동시 속도 제어 자동화 제한보다 많은 자동화를 실행하려고 하면 후속 자동화가 대기열에 추가됩니다. AWS 계정마다 1,000개의 속도 제어 자동화를 대기열에 넣을 수 있습니다. 자동화가 완료되거나 종료 상태에 도달하면 대기열의 첫 번째 자동화가 시작됩니다.

주제

Automation 설정

AWS Systems Manager의 기능인 Automation을 설정하려면 Automation 서비스에 대한 사용자 액세스 권한을 확인하고 역할을 구성하여 서비스에서 해당 리소스를 대신하여 작업을 수행할 수 있도록 해야 합니다. 또한 Automation 기본 설정에서 적응형 동시성 모드를 선택하는 것이 좋습니다. 적응형 동시성

은 사용자의 필요에 맞게 자동화 할당량을 자동으로 조정합니다. 자세한 내용은 Automation이 동시성 요구 사항에 따라 조정되도록 허용 단원을 참조하십시오.

AWS Systems Manager 자동화에 대한 액세스 권한이 적절한지 확인하려면 다음 사용자 및 서비스 역할 요구 사항을 검토하십시오.

실행서에 대한 사용자 액세스 확인

실행서를 사용할 권한이 있는지 확인합니다. 사용자, 그룹 또는 역할에 관리자 권한이 할당되어 있는 경우 Systems Manager Automation에 액세스할 수 있습니다. 관리자 권한이 없는 경우 관리자가 `AmazonSSMFullAccess` 관리형 정책 또는 비교 가능한 권한을 제공하는 정책을 해당 사용자, 그룹 또는 역할에 할당하여 사용자에게 권한을 부여해야 합니다.

> **Important**
>
> IAM 정책 `AmazonSSMFullAccess`는 Systems Manager 작업에 대해 권한을 부여합니다. 하지만 실행서 중에는 `ec2:ReleaseAddress`에 대한 IAM 권한이 필요한 실행서 `AWS-ReleaseElasticIP`와 같이, 다른 서비스에 대한 권한이 필요한 경우가 있습니다. 따라서 런북에서 수행할 작업을 검토하여 사용자, 그룹 또는 역할에 런북에 포함된 작업을 수행할 수 있는 필수 권한이 할당되어 있는지 확인해야 합니다.

자동화에 대한 서비스 역할(수임 역할) 액세스 구성

자동화는 서비스 역할(또는 수임 역할) 컨텍스트에 따라 시작할 수 있습니다. 그러면 서비스가 사용자 대신 작업을 수행할 수 있습니다. 수임 역할이 지정되어 있지 않으면 Automation은 자동화를 호출한 사용자 컨텍스트를 사용합니다.

하지만 다음과 같은 경우에는 자동화에 대한 서비스 역할을 지정해야 합니다.

- 리소스에 대한 사용자의 권한을 제한하려고 하지만 사용자가 더 높은 권한이 필요한 자동화를 실행하도록 하려는 경우입니다. 이 시나리오에서는 더 높은 권한이 있는 서비스 역할을 생성하고 사용자에게 자동화를 실행하도록 허용할 수 있습니다.
- 실행서를 실행하는 Systems Manager State Manager 연결을 만들 때.
- 작업을 12시간 이상 실행해야 하는 작업이 있는 경우
- `aws:executeScript` 작업을 사용하여 AWS API 작업을 호출하거나 AWS 리소스에 대한 작업을 수행하는 Amazon이 소유하지 않은 실행서를 실행하고 있는 경우. 자세한 설명은 실행서 사용 권한을 참조하세요.

자동화를 위한 서비스 역할을 생성해야 하는 경우 다음 방법 중 하나를 사용할 수 있습니다.

주제

방법 1: AWS CloudFormation을 사용하여 Automation을 위한 서비스 역할 구성

AWS CloudFormation 템플릿에서 AWS Systems Manager의 기능인 Automation을 위한 서비스 역할을 생성할 수 있습니다. 서비스 역할을 생성한 후에는 파라미터 `AutomationAssumeRole`을 사용하여 실행서에 서비스 역할을 지정할 수 있습니다.

AWS CloudFormation을 사용하여 서비스 역할 생성

다음 절차를 따라 AWS CloudFormation을 사용하여 Systems Manager Automation에 필요한 AWS Identity and Access Management(IAM) 역할을 생성합니다.

필요한 IAM 역할을 생성하려면

1. `AWS-SystemsManager-AutomationServiceRole.zip` 파일을 다운로드하고 압축을 해제합니다. 이 파일에는 `AWS-SystemsManager-AutomationServiceRole.yaml` AWS CloudFormation 템플릿 파일이 들어 있습니다.
2. AWS CloudFormation 콘솔(https://console.aws.amazon.com/cloudformation)을 엽니다.
3. 스택 생성을 선택합니다.
4. 템플릿 지정(Specify template) 섹션에서 템플릿 파일 업로드를 선택합니다.
5. 찾아보기를 선택한 다음 `AWS-SystemsManager-AutomationServiceRole.yaml` AWS CloudFormation 템플릿 파일을 선택합니다.
6. 다음을 선택합니다.
7. 스택 세부 정보 지정(Specify stack details) 페이지의 스택 이름 필드에 이름을 입력합니다.
8. 스택 옵션 구성(Configure stack options) 페이지에서는 아무 것도 선택할 필요가 없습니다. 다음을 선택합니다.
9. 검토 페이지에서 아래로 스크롤하여 AWS CloudFormation이 IAM 리소스를 생성할 수 있다는 것을 알고 있음 옵션을 선택합니다.

10. 생성(Create)을 선택합니다.

CloudFormation에 약 3분간 CREATE_IN_PROGRESS 상태가 표시됩니다. 스택이 생성되어 역할을 사용할 준비가 되면 상태가 CREATE_COMPLETE로 바뀝니다.

> **Important**
>
> AWS Identity and Access Management(IAM) 서비스 역할을 사용하여 다른 서비스를 호출하는 자동화 워크플로를 실행하는 경우 해당 서비스 역할이 다른 서비스를 호출할 권한이 있도록 구성되어야 합니다. 이 요구 사항은 `AWS-ConfigureS3BucketLogging`, `AWS-CreateDynamoDBBackup`, `AWS-RestartEC2Instance` 실행서 등의 모든 AWS Automation 실행서(`AWS-*` 실행서)에 적용됩니다. 또한 다른 서비스를 호출하는 작업을 사용하여 다른 AWS 서비스을(를) 호출하는 사용자 정의 Automation 실행서를 생성하는 경우에도 이 요구 사항이 항상 적용됩니다. 예를 들어 `aws:executeAwsApi`, `aws:createStack` 또는 `aws:copyImage` 작업을 사용하는 경우 이러한 서비스를 호출할 수 있는 권한을 포함하여 서비스 역할을 구성합니다. 역할에 IAM 인라인 정책을 추가하여 다른 AWS 서비스에 대한 권한을 부여할 수 있습니다. 자세한 내용은 (선택 사항) 다른 AWS 서비스를 간접적으로 호출할 Automation 인라인 정책과 고객 관리형 정책 추가 단원을 참조하십시오.

Automation을 위한 역할 정보 복사

다음 절차를 따라 AWS CloudFormation 콘솔에서 자동화 서비스 역할에 대한 정보를 복사합니다. 실행서를 사용할 때 이러한 역할을 지정해야 합니다.

> **Note**
>
> `AWS-UpdateLinuxAmi` 또는 `AWS-UpdateWindowsAmi` 실행서를 실행하는 경우 이 절차를 사용하여 역할 정보를 복사할 필요가 없습니다. 이러한 실행서에는 이미 기본값으로 지정된 필수 역할이 있습니다. 이 실행서에 지정된 역할은 IAM 관리형 정책을 사용합니다.

역할 이름을 복사하려면

1. AWS CloudFormation 콘솔(https://console.aws.amazon.com/cloudformation)을 엽니다.
2. 이전 절차에서 생성한 자동화 스택 이름을 선택합니다.
3. 리소스 탭을 선택합니다.

4. AutomationServiceRole에 대한 물리적 ID(Physical ID)를 선택합니다. IAM 콘솔이 열리고 Automation 서비스 역할의 요약 정보가 표시됩니다.
5. 역할 ARN 옆에 있는 Amazon 리소스 이름(ARN)을 복사합니다. ARN은 `arn:aws:iam::12345678:role/AutomationServiceRole`과 비슷합니다.
6. 나중에 사용할 수 있도록 ARN을 텍스트 파일에 붙여 넣습니다.

이렇게 해서 자동화에 필요한 서비스 역할 구성을 마쳤습니다. 이제 실행서에 Automation 서비스 역할의 ARN을 사용할 수 있습니다.

방법 2: IAM을 사용하여 Automation을 위한 역할 구성

AWS Systems Manager의 기능인 Automation용 서비스 역할을 생성해야 하는 경우 다음 태스크를 완료합니다. Automation을 위해 서비스 역할이 필요한 상황에 대한 자세한 내용은 Automation 설정 섹션을 참조하세요.

Tasks

- 작업 1: Automation을 위한 서비스 역할 생성
- 작업 2: Automation 역할에 iam:PassRole 정책 연결하기

작업 1: Automation을 위한 서비스 역할 생성

다음 절차를 사용하여 Systems Manager Automation에 대한 서비스 역할 또는 수임 역할을 생성합니다.

> **Note**
>
> `AWS-CreateManagedLinuxInstance` 실행서와 같은 실행서에서도 이 역할을 사용할 수도 있습니다. 실행서에서 이 역할 또는 AWS Identity and Access Management(IAM) 역할의 Amazon 리소스 이름(ARN)을 사용하면 Automation을 통해 환경에서 새 인스턴스 시작 및 사용자 대신 작업 수행과 같은 작업을 수행할 수 있습니다.

IAM 역할을 생성하고 Automation에서 이를 수임하도록 허용하려면

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할을 선택한 후 역할 생성을 선택합니다.

3. 신뢰할 수 있는 유형의 엔터티 선택(Select type of trusted entity) 아래에서 AWS 서비스를 선택합니다.
4. 사용 사례 선택 섹션에서 Systems Manager를 선택한 후 다음: 권한을 선택합니다.
5. Attach permissions policy(권한 정책 연결) 페이지에서 AmazonSSMAutomationRole 정책을 검색하여 선택한 후 다음: 검토를 선택합니다.
6. 검토 페이지에서 역할 이름 상자에 이름을 입력한 후 설명을 입력합니다.
7. 역할 생성(Create role)을 선택합니다. 그러면 역할 페이지로 돌아갑니다.
8. 역할 페이지에서 방금 만든 역할을 선택하여 요약 페이지를 엽니다. 역할 이름 및 역할 ARN을 메모합니다. 다음 절차에서 IAM 계정에 iam:PassRole 정책을 연결할 때 역할 ARN을 지정합니다. 실행서에서 역할 이름과 ARN을 지정할 수도 있습니다.

> **Note**
>
> `AmazonSSMAutomationRole` 정책은 계정 내 AWS Lambda 함수의 하위 집합에 Automation 역할 권한을 할당합니다. 이 함수는 "Automation"으로 시작합니다. Lambda 함수와 함께 Automation을 사용할 예정인 경우 Lambda ARN에서 다음 형식을 사용해야 합니다. `"arn:aws:lambda:*:*:function:Automation*"`
> ARN이 이 형식을 사용하지 않는 기존 Lambda 함수가 있는 경우 AWSLambdaRole 정책과 같은 추가 Lambda 정책도 자동화 역할에 연결해야 합니다. 추가 정책 또는 역할은 AWS 계정 내에서 Lambda 함수에 대한 보다 폭넓은 액세스를 제공해야 합니다.

서비스 역할을 생성한 후 교차 서비스 혼동된 대리자 문제를 방지하기 위해 신뢰 정책을 편집하는 것이 좋습니다. 혼동된 대리자 문제는 작업을 수행할 권한이 없는 엔터티가 권한이 더 많은 엔터티에 작업을 수행하도록 강요할 수 있는 보안 문제입니다. AWS에서는 교차 서비스 가장으로 인해 혼동된 대리자 문제가 발생할 수 있습니다. 교차 서비스 가장은 한 서비스(직접 호출하는 서비스)가 다른 서비스(직접 호출되는 서비스)를 직접 호출할 때 발생할 수 있습니다. 직접 호출하는 서비스는 다른 고객의 리소스에 대해 액세스 권한이 없는 방식으로 작동하게 권한을 사용하도록 조작될 수 있습니다. 이를 방지하기 위해 AWS에서는 계정의 리소스에 대한 액세스 권한이 부여된 서비스 보안 주체를 사용하여 모든 서비스에 대한 데이터를 보호하는 데 도움이 되는 도구를 제공합니다.

Automation이 리소스에 다른 서비스를 제공하는 권한을 제한하려면 리소스 정책에서 `aws:SourceArn` 및 `aws:SourceAccount` 전역 조건 컨텍스트 키를 사용하는 것이 좋습니다. 만약 `aws:SourceArn` 값에 Amazon S3 버킷 ARN과 같은 계정 ID가 포함되어 있지 않은 경우 권한을 제한하려면 두 전역 조건 컨텍스트 키를 모두 사용해야 합니다. 두 전역 조건 컨텍스트 키와 계정을 포함한 `aws:SourceArn` 값을 모두 사용하는 경우, `aws:SourceAccount` 값 및

aws:SourceArn 값의 계정은 동일한 정책 명령문에서 사용할 경우 반드시 동일한 계정 ID를 사용해야 합니다. 하나의 리소스만 교차 서비스 액세스와 연결되도록 허용하려는 경우 aws:SourceArn를 사용하십시오. 해당 계정의 모든 리소스가 교차 서비스 사용과 연결되도록 허용하려는 경우 aws:SourceAccount을 사용하세요. aws:SourceArn의 값은 자동화 실행을 위한 ARN이어야 합니다. 리소스의 전체 ARN을 모를 경우 또는 여러 리소스를 지정하는 경우, ARN의 알 수 없는 부분에 대해 와일드카드(*)를 포함한 aws:SourceArn 글로벌 조건 컨텍스트 키를 사용합니다. 예를 들면 arn:aws:ssm:*:123456789012:automation-execution/*입니다.

다음 예는 자동화시 aws:SourceArn 및 aws:SourceAccount 전역 조건 컨텍스트 키를 사용하여 혼동된 대리자 문제를 방지하는 방법을 보여줍니다.

```
{
  "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/*"
        }
      }
    }
  ]
}
```

역할의 신뢰 정책 수정

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할을 선택합니다.
3. 계정의 역할 목록에서 Automation 서비스 역할의 이름을 선택합니다.
4. 신뢰 관계 탭을 선택한 후 신뢰 관계 편집을 선택합니다.

5. 혼동된 대리자 문제를 방지하기 위해 Automation에 `aws:SourceArn` 및 `aws:SourceAccount` 전역 조건 컨텍스트 키를 사용하여 신뢰 정책을 편집합니다.
6. 신뢰 정책 업데이트(Update Trust Policy)를 선택하여 변경 사항을 저장합니다.

(선택 사항) 다른 AWS 서비스를 간접적으로 호출할 Automation 인라인 정책과 고객 관리형 정책 추가

IAM 서비스 역할을 사용하여 다른 AWS 서비스을(를) 호출하는 자동화를 실행하는 경우 해당 서비스 역할이 다른 서비스를 호출할 권한이 있도록 구성되어야 합니다. 이 요구 사항은 `AWS-ConfigureS3BucketLogging`, `AWS-CreateDynamoDBBackup`, `AWS-RestartEC2Instance` 실행서 등의 모든 AWS Automation 실행서(`AWS-*` 실행서)에 적용됩니다. 또한 다른 서비스를 호출하는 작업을 사용하여 다른 AWS 서비스을(를) 호출하는 사용자 정의 실행서를 생성하는 경우에도 이 요구 사항이 항상 적용됩니다. 예를 들어 `aws:executeAwsApi`, `aws:CreateStack` 또는 `aws:copyImage`를 사용하는 경우 이러한 서비스를 호출할 수 있는 권한을 포함하여 서비스 역할을 구성해야 합니다. 역할에 IAM 인라인 정책과 고객 관리형 정책을 추가하여 다른 AWS 서비스에 대한 권한을 부여할 수 있습니다.

서비스 역할의 인라인 정책을 포함하려면(IAM 콘솔)

1. AWS Management Console에 로그인하여 https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할을 선택합니다.
3. 목록에서 편집할 역할의 이름을 선택합니다.
4. 권한 탭을 선택합니다.
5. 권한 추가 드롭다운에서 정책 연결 또는 인라인 정책 생성을 선택합니다.
6. 정책 연결을 선택하는 경우 추가할 정책 옆의 확인란을 선택하고 권한 추가를 선택합니다.
7. 인라인 정책 생성을 선택하는 경우 JSON 탭을 선택합니다.
8. 호출하려는 AWS 서비스의 JSON 정책 문서를 입력합니다. 다음은 2개의 예제 JSON Policy 문서입니다.

   Amazon S3 PutObject 및 GetObject 예

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

```
                "s3:PutObject",
                "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
        }
    ]
}
```

Amazon EC2 CreateSnapshot 및 DescribeSnapShots 예

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":"ec2:CreateSnapshot",
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":"ec2:DescribeSnapshots",
         "Resource":"*"
      }
   ]
}
```

IAM 정책 언어에 대한 자세한 내용은 IAM User Guide의 IAM JSON Policy Reference를 참조하세요.

9. 작업이 완료되면 정책 검토(Review policy)를 선택합니다. 정책 검사기가 모든 구문 오류를 보고합니다.
10. 정책 검토(Review policy) 페이지에서 생성 중인 정책의 이름(Name)을 입력합니다. 정책 요약을 검토하여 정책이 부여한 권한을 확인합니다. 그런 다음 정책 생성을 선택하여 작업을 저장합니다.
11. 인라인 정책을 생성하면 이 정책이 역할에 자동으로 포함됩니다.

작업 2: Automation 역할에 iam:PassRole 정책 연결하기

다음 절차에 따라 자동화 서비스 역할에 `iam:PassRole` 정책을 연결합니다. 이렇게 하면 Automation 서비스가 자동화를 실행할 때 이 역할을 다른 서비스 또는 Systems Manager 기능에 전달할 수 있습니다.

iam:PassRole 정책을 자동화 역할에 연결하려면,

1. 방금 만든 역할에 대한 요약 페이지에서 권한 탭을 선택합니다.
2. 인라인 정책 추가(Add inline policy)를 선택합니다.
3. 정책 생성(Create policy) 페이지에서 시각적 편집기(Visual editor) 탭을 선택합니다.
4. 서비스(Service)와 IAM을 차례대로 선택합니다.
5. 작업 선택을 선택합니다.
6. 작업 필터링 텍스트 상자에 **PassRole**을 입력하고 PassRole 옵션을 선택합니다.
7. 리소스를 선택합니다. Specific(특정)이 선택되었는지 확인한 다음, ARN 추가(Add ARN)를 선택합니다.
8. Specify ARN for role(역할 ARN 지정) 필드에 작업 1의 마지막에 복사한 자동화 역할 ARN을 붙여넣습니다. 계정(Account) 및 역할 이름 및 경로(Role name with path) 필드에 값이 채워집니다.

   > **Note**
   >
   > Automation 서비스 역할에서 EC2 인스턴스에 IAM 인스턴스 프로파일 역할을 연결하려면 IAM 인스턴스 프로파일 역할의 ARN을 추가해야 합니다. 이렇게 하면 Automation 서비스 역할이 IAM 인스턴스 프로파일 역할을 대상 EC2 인스턴스로 전달할 수 있습니다.

9. 추가(Add)를 선택합니다.
10. 정책 검토(Review policy)를 선택합니다.
11. 정책 검토(Review Policy) 페이지에 이름을 입력한 다음 정책 생성(Create Policy)을 선택합니다.

Automation이 동시성 요구 사항에 따라 조정되도록 허용

기본적으로 Automation을 사용하면 한 번에 최대 100개의 동시 자동화를 실행할 수 있습니다. Automation은 동시성 자동화 할당량을 자동으로 조정하는 데 사용할 수 있는 선택적 설정도 제공합니다. 이 설정을 사용하면 사용 가능한 리소스에 따라 동시성 자동화 할당량이 최대 500개의 동시 자동화를 수용할 수 있습니다.

> **Note**
>
> 자동화가 API 작업을 호출하는 경우 대상에 맞게 조정하면 제한 예외가 발생할 수 있습니다. 적응형 동시성이 설정된 상태에서 자동화를 실행할 때 반복 제한 예외가 발생하면 사용 가능한 경우 API 작업에 대한 할당량 증가를 요청해야 할 수 있습니다.

적응형 동시성 설정(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 왼쪽 탐색 창에서 Automation을 선택합니다.
3. 기본 설정 탭을 선택하고 편집을 선택합니다.
4. 적응형 동시성 사용(Enable adaptive concurrency) 옆에 있는 확인란을 선택합니다.
5. Save(저장)를 선택합니다.

Automation을 위한 변경 제어 구현

기본적으로 Automation을 사용하면 날짜 및 시간 제한 없이 런북을 사용할 수 있습니다. Automation과 Change Calendar를 통합하여 AWS 계정의 모든 자동화에 대한 변경 제어를 구현할 수 있습니다. 이 설정을 사용하면 계정의 AWS Identity and Access Management(IAM) 보안 주체가 변경 일정에서 허용하는 기간 동안에만 자동화를 실행할 수 있습니다. Change Calendar 사용에 대한 자세한 내용은 Change Calendar 작업 섹션을 참조하세요.

변경 제어 켜기(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 왼쪽 탐색 창에서 Automation을 선택합니다.
3. 기본 설정 탭을 선택하고 편집을 선택합니다.
4. Change Calendar 통합 켜기 옆의 확인란을 선택합니다.
5. 변경 캘린더 선택 드롭다운 목록에서 Automation이 따를 변경 캘린더를 선택합니다.
6. Save(저장)를 선택합니다.

자동화 실행

이 섹션에는 Automation 실행서를 실행하는 방법에 대한 정보가 들어 있습니다. Automation은 AWS Systems Manager의 기능입니다. 사용 사례에 따라 자동화를 실행하는 방법을 설명하는 자습서는 튜토리얼에서 참조할 수 있습니다.

내용

- 대규모로 자동화 실행
- 여러 AWS 리전 및 계정에서 자동화 실행
- 이벤트를 기반으로 자동화 실행
- 수동으로 자동화 실행

자동화 실행

자동화를 실행하면 기본적으로 자동화를 시작한 사용자의 컨텍스트에서 자동화가 실행됩니다. 따라서 사용자가 관리자 권한을 가지고 있는 경우에는 관리자 권한과 자동화를 통해 구성 중인 리소스에 대한 모든 액세스 권한을 가지고 자동화가 실행됩니다. 보안 모범 사례에 따라 이 경우 AmazonSSMAutomationRole 관리형 정책으로 구성된 수임 역할로 알려진 IAM 서비스 역할을 사용하여 자동화를 실행하는 것이 좋습니다. 다양한 런북을 사용하려면 수임 역할에 IAM 정책을 추가해야 할 수 있습니다. IAM 서비스 역할을 사용하여 자동화를 실행하는 것을 위임된 관리라고 합니다.

서비스 역할을 사용하면 AWS 리소스에 대한 자동화 실행이 허용되지만, 자동화를 실행한 사용자는 해당 리소스에 대한 액세스 권한이 제한되거나 액세스 권한이 없습니다. 예를 들어 서비스 역할을 구성하고 Automation에서 이를 사용하여 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스를 하나 이상 다시 시작할 수 있습니다. Automation은 AWS Systems Manager의 기능입니다. 자동화는 인스턴스를 다시 시작하지만, 서비스 역할은 사용자에게 이들 인스턴스에 액세스할 수 있는 권한을 부여하지 않습니다.

자동화를 실행할 때 런타임 시 서비스 역할을 지정하거나, 사용자 정의 실행서를 생성하고 실행서에서 서비스 역할을 직접 지정할 수 있습니다. 런타임 시 또는 실행서에서 서비스 역할을 지정하면 지정된 서비스 역할의 컨텍스트에서 서비스가 실행됩니다. 서비스 역할을 지정하지 않으면 시스템이 사용자의 컨텍스트에서 임시 세션을 생성하고 자동화를 실행합니다.

> **Note**
>
> 자동화를 12시간 이상 실행해야 하는 경우에는 자동화를 위한 서비스 역할을 반드시 지정해야 합니다. 사용자의 컨텍스트에서 장기 실행 자동화를 시작하면 12시간 후에 사용자의 임시 세션이 만료됩니다.

위임된 관리를 통해 AWS 리소스에 대한 보안과 제어를 강화할 수 있습니다. 또한 여러 IAM 계정 대신 중앙의 서비스 역할이 리소스에 대한 작업을 수행하기 때문에 감사 기능을 강화할 수 있습니다.

시작하기 전 준비 사항

다음 절차를 완료하기 전에 먼저 IAM 서비스 역할을 생성하고 AWS Systems Manager의 기능인 Automation에 대해 신뢰 관계를 구성해야 합니다. 자세한 내용은 작업 1: Automation을 위한 서비스 역할 생성 단원을 참조하십시오.

다음 절차에서는 Systems Manager 콘솔 또는 원하는 명령줄 도구를 사용하여 간단한 자동화를 실행하는 방법을 설명합니다.

단순한 자동화 실행(콘솔)

다음 절차에서는 Systems Manager 콘솔을 사용하여 단순한 자동화를 실행하는 방법을 설명합니다.

단순한 자동화를 실행하려면

1. AWS Systems Manager 콘솔https://console.aws.amazon.com/systems-manager/을 엽니다.
2. 탐색 창에서 Automation(자동화)을 선택한 후 Execute automation(자동화 실행)을 선택합니다.
3. [Automation 문서(Automation document)] 목록에서 실행서를 선택합니다. 문서 카테고리 창에서 옵션을 1개 이상 선택하여 SSM 문서를 목적에 따라 필터링합니다. 자신이 소유한 실행서를 보려면 [내 소유(Owned by me)] 탭을 선택합니다. 자신의 계정과 공유하고 있는 실행서를 보려면 [나와 공유됨(Shared with me)] 탭을 선택합니다. 모든 실행서를 보려면 [모든 문서(All documents)] 탭을 선택합니다.

   > **Note**
   >
   > 실행서 이름을 선택하여 실행서에 대한 정보를 볼 수 있습니다.

4. 문서 세부 정보 섹션에서 문서 버전이 실행할 버전으로 설정되었는지 확인합니다. 이 시스템에는 다음 버전 옵션이 포함되어 있습니다.

   - 런타임 시 기본 버전 - Automation 런북이 정기적으로 업데이트되며 새 기본 버전이 할당된 경우 이 옵션을 선택합니다.
   - 런타임 시 최신 버전 - Automation 런북이 정기적으로 업데이트되며 최근에 업데이트된 버전을 실행하려는 경우 이 옵션을 선택합니다.
   - 1(기본값) - 문서의 최초 버전을 실행하려면 이 옵션을 선택합니다(기본값).

5. 다음을 선택합니다.
6. 실행 모드 섹션에서 Simple execution(단순 실행)을 선택합니다.
7. 입력 파라미터 섹션에서 필수 입력을 지정합니다. 필요에 따라 AutomationAssumeRole 목록에서 IAM 서비스 역할을 선택합니다.

8. (선택 사항) 모니터링을 위해 자동화에 적용할 CloudWatch 경보를 선택합니다. CloudWatch 경보를 자동화에 연결하려면 자동화를 시작하는 IAM 보안 주체에 `iam:createServiceLinkedRole` 작업에 대한 권한이 있어야 합니다. CloudWatch 경보에 대한 자세한 내용은 Amazon CloudWatch 경보 사용을 참조하세요. 경보가 활성화되면 자동화가 중지됩니다. AWS CloudTrail을 사용하면 추적에 API 호출이 표시됩니다.
9. 실행을 선택합니다.

콘솔에 자동화 상태가 표시됩니다. 자동화가 실행되지 않는 경우 Systems Manager Automation 문제 해결 섹션을 참조하세요.

단순한 자동화 실행(명령줄)

다음 절차에서는 AWS CLI(Linux 또는 Windows) 또는 AWS Tools for PowerShell을 사용하여 단순한 자동화를 실행하는 방법을 설명합니다.

단순한 자동화를 실행하려면

1. 아직 하지 않은 경우 AWS CLI 또는 AWS Tools for PowerShell을 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트 및 AWS Tools for PowerShell 설치를 참조하세요.
2. 단순한 자동화를 시작하려면 다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   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
```

여기에 실행서 AWS-RestartEC2Instance를 사용하여 지정된 EC2 인스턴스를 다시 시작하는 예가 있습니다.

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"}
```

시스템은 다음과 같은 정보를 반환합니다.

Linux & macOS

```
{
    "AutomationExecutionId": "4105a4fc-f944-11e6-9d32-0123456789ab"
}
```

Windows

```
{
    "AutomationExecutionId": "4105a4fc-f944-11e6-9d32-0123456789ab"
}
```

PowerShell

```
4105a4fc-f944-11e6-9d32-0123456789ab
```

3. 다음 명령을 실행하여 자동화 상태를 검색합니다.

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"}
```

시스템은 다음과 같은 정보를 반환합니다.

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                     : {}
```

승인자를 사용하여 자동화 실행

다음 절차에서는 단순 실행 모드에서 승인자를 사용하여 자동화를 실행하도록 AWS Systems Manager 콘솔과 AWS Command Line Interface(AWS CLI)를 사용하는 방법을 설명합니다. 자동화는 지정된 보안 주체가 작업을 승인 또는 거부할 때까지 자동화를 일시 중지하는 자동화 작업 `aws:approve`를 사용합니다. 자동화는 현재 사용자의 컨텍스트에서 실행됩니다. 이는 런북과 그 런북에 의해 호출되는 모든 작업을 사용할 권한이 있는 한 추가 IAM 권한을 구성할 필요가 없음을 뜻합니다. IAM에서 관리자 권한이 있는 경우 이 런북을 사용할 권한이 이미 있습니다.

시작하기 전 준비 사항

실행서에 요구되는 표준 입력 이외에 `aws:approve` 작업에는 다음 두 파라미터가 필요합니다.

- 승인자 목록. 승인자 목록은 사용자 이름 또는 사용자 ARN 형식의 승인자를 하나 이상 포함해야 합니다. 여러 승인자를 제공하는 경우 런북 내에 해당하는 최소 승인 횟수도 지정해야 합니다.
- Amazon Simple Notification Service(Amazon SNS) 주제 ARN Amazon SNS 주제 이름은 `Automation`으로 시작해야 합니다.

이 절차에서는 승인 요청을 전송하는 데 필요한 Amazon SNS 주제를 이미 생성한 것으로 가정합니다. 자세한 내용은 Amazon Simple Notification Service Developer Guide의 Create a Topic을 참조하세요.

승인자를 사용하여 자동화 실행(콘솔)

승인자를 사용하여 자동화를 실행하려면

다음 절차에서는 Systems Manager 콘솔에서 승인자를 사용하여 자동화를 실행하는 방법을 설명합니다.

1. AWS Systems Manager 콘솔https://console.aws.amazon.com/systems-manager/을 엽니다.
2. 탐색 창에서 Automation(자동화)을 선택한 후 Execute automation(자동화 실행)을 선택합니다.
3. [Automation 문서(Automation document)] 목록에서 실행서를 선택합니다. 문서 카테고리 창에서 옵션을 1개 이상 선택하여 SSM 문서를 목적에 따라 필터링합니다. 자신이 소유한 실행서를 보려면 [내 소유(Owned by me)] 탭을 선택합니다. 자신의 계정과 공유하고 있는 실행서를 보려면 [나와 공유됨(Shared with me)] 탭을 선택합니다. 모든 실행서를 보려면 [모든 문서(All documents)] 탭을 선택합니다.

   > **Note**
   >
   > 실행서 이름을 선택하여 실행서에 대한 정보를 볼 수 있습니다.

4. 문서 세부 정보 섹션에서 문서 버전이 실행할 버전으로 설정되었는지 확인합니다. 이 시스템에는 다음 버전 옵션이 포함되어 있습니다.

   - 런타임 시 기본 버전 - Automation 런북이 정기적으로 업데이트되며 새 기본 버전이 할당된 경우 이 옵션을 선택합니다.
   - 런타임 시 최신 버전 - Automation 런북이 정기적으로 업데이트되며 최근에 업데이트된 버전을 실행하려는 경우 이 옵션을 선택합니다.
   - 1(기본값) - 문서의 최초 버전을 실행하려면 이 옵션을 선택합니다(기본값).

5. 다음을 선택합니다.
6. Execute automation document(자동화 문서 실행) 페이지에서 Simple execution(단순 실행)을 선택합니다.
7. 입력 파라미터 섹션에서 필요한 입력 파라미터를 지정합니다.

   예를 들어 **`AWS-StartEC2InstanceWithApproval`** 런북을 선택한 경우 InstanceId 파라미터에 대해 인스턴스 ID를 지정하거나 선택해야 합니다.
8. 승인자 섹션에서 자동화 작업 승인자의 IAM 사용자 또는 사용자 ARN을 지정합니다.

9. SNSTopicARN 섹션에서 승인 알림을 전송하는 데 사용할 SNS 주제 ARN을 지정합니다. SNS 주제 이름은 Automation으로 시작해야 합니다.
10. 필요에 따라 [AutomationAssumeRole] 목록에서 IAM 서비스 역할을 선택합니다. 100개가 넘는 계정 및 리전을 대상으로 지정하는 경우 `AWS-SystemsManager-AutomationAdministrationRole`을 지정해야 합니다.
11. 자동화 실행(Execute automation)을 선택합니다.

지정된 승인자는 자동화를 승인 또는 거부하기 위한 세부 정보가 표함된 Amazon SNS 알림을 받습니다. 이 승인 작업은 발행 날짜로부터 7일간 유효하고 Systems Manager 콘솔 또는 AWS Command Line Interface(AWS CLI)를 사용하여 발행할 수 있습니다.

자동화를 승인하기로 결정하면 자동화가 지정된 실행서에 포함된 단계를 실행합니다. 콘솔에 자동화 상태가 표시됩니다. 자동화가 실행되지 않는 경우 Systems Manager Automation 문제 해결 섹션을 참조하세요.

자동화를 승인 또는 거부하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 [자동화(Automation)]를 선택하고 이전 절차에서 실행한 자동화를 선택합니다.
3. 작업을 선택하고 Approve/Deny(승인/거부)를 선택합니다.
4. 승인 또는 거부를 선택하고 선택적으로 설명을 입력합니다.
5. 제출을 선택합니다.

승인자를 사용하여 자동화 실행(명령줄)

다음 절차에서는 AWS CLI(Linux 또는 Windows) 또는 AWS Tools for PowerShell에서 승인자를 사용하여 자동화를 실행하는 방법을 설명합니다.

승인자를 사용하여 자동화를 실행하려면

1. 아직 하지 않은 경우 AWS CLI 또는 AWS Tools for PowerShell을 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트 및 AWS Tools for PowerShell 설치를 참조하세요.
2. 승인자를 사용하여 자동화를 실행하려면 다음 명령을 실행합니다. 각 *`example resource placeholder`*를 사용자의 정보로 바꿉니다. 문서 이름 섹션에 `aws:approve` 자동화 작업을 포함하는 런북을 지정합니다.

Approvers에서 작업 승인자의 사용자 이름 또는 사용자 ARN을 지정합니다. SNSTopic에서 승인 알림을 전송하는 데 사용할 SNS 주제 ARN을 지정합니다. Amazon SNS 주제 이름은 Automation으로 시작해야 합니다.

> **Note**
>
> 승인자 및 SNS 주제의 파라미터 값은 선택한 실행서에 지정된 값에 따라 특정한 이름이 다릅니다.

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"
    }
```

시스템은 다음과 같은 정보를 반환합니다.

Linux & macOS

```
{
    "AutomationExecutionId": "df325c6d-b1b1-4aa0-8003-6cb7338213c6"
}
```

Windows

```
{
    "AutomationExecutionId": "df325c6d-b1b1-4aa0-8003-6cb7338213c6"
}
```

PowerShell

```
df325c6d-b1b1-4aa0-8003-6cb7338213c6
```

자동화를 승인하려면

- 자동화를 승인하려면 다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

  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"}
```

명령이 성공해도 결과는 없습니다.

자동화를 거부하려면

- 자동화를 거부하려면 다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

  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"}
```

  명령이 성공해도 결과는 없습니다.

대규모로 자동화 실행

AWS Systems Manager Automation을 사용하면 대상을 사용하여 AWS 리소스 플릿에서 자동화를 실행할 수 있습니다. 또한 동시성 값과 오류 임계값을 지정하여 플릿 전체의 자동화 배포를 제어할 수도

있습니다. 동시성 및 오류 임계값 기능을 통칭하여 속도 제어라고 합니다. 동시성 값은 자동화를 동시에 실행하도록 허용된 리소스의 수를 결정합니다. 또한 Automation은 사용자가 선택할 수 있는 적응형 동시성 모드를 제공합니다. 적응형 동시성은 동시에 실행되는 자동화 100개에서 최대 500개로 자동화 할당량을 자동으로 조정합니다. 오류 임계값은 Systems Manager가 자동화를 다른 리소스로 보내는 것을 중지할 때까지 허용되는 자동화 실패 횟수를 결정합니다.

동시성 및 오류 임계값에 대한 자세한 내용은 대규모 자동화 제어 섹션을 참조하세요. 대상에 대한 자세한 내용은 자동화를 위한 대상 매핑 섹션을 참조하세요.

다음 절차에서는 적응형 동시성을 설정하는 방법과 Systems Manager 콘솔 및 AWS Command Line Interface(AWS CLI)에서 대상 및 속도 제어 기능을 사용하는 자동화를 실행하는 방법을 설명합니다.

대상 및 속도 제어를 사용하여 자동화 실행(콘솔)

다음 절차에서는 Systems Manager 콘솔에서 대상 및 속도 제어를 사용하여 자동화를 실행하는 방법을 설명합니다.

대상 및 속도 제어를 사용하여 자동화를 실행하려면

1. AWS Systems Manager 콘솔https://console.aws.amazon.com/systems-manager/을 엽니다.
2. 탐색 창에서 Automation(자동화)을 선택한 후 Execute automation(자동화 실행)을 선택합니다.
3. [Automation 문서(Automation document)] 목록에서 실행서를 선택합니다. 문서 카테고리 창에서 옵션을 1개 이상 선택하여 SSM 문서를 목적에 따라 필터링합니다. 자신이 소유한 실행서를 보려면 [내 소유(Owned by me)] 탭을 선택합니다. 자신의 계정과 공유하고 있는 실행서를 보려면 [나와 공유됨(Shared with me)] 탭을 선택합니다. 모든 실행서를 보려면 [모든 문서(All documents)] 탭을 선택합니다.

   > **Note**
   >
   > 실행서 이름을 선택하여 실행서에 대한 정보를 볼 수 있습니다.

4. 문서 세부 정보 섹션에서 문서 버전이 실행할 버전으로 설정되었는지 확인합니다. 이 시스템에는 다음 버전 옵션이 포함되어 있습니다.

   - 런타임 시 기본 버전 - Automation 런북이 정기적으로 업데이트되며 새 기본 버전이 할당된 경우 이 옵션을 선택합니다.
   - 런타임 시 최신 버전 - Automation 런북이 정기적으로 업데이트되며 최근에 업데이트된 버전을 실행하려는 경우 이 옵션을 선택합니다.
   - 1(기본값) - 문서의 최초 버전을 실행하려면 이 옵션을 선택합니다(기본값).

5. 다음을 선택합니다.
6. 실행 모드 섹션에서 Rate Control(속도 제어)을 선택합니다. 대상 및 속도 제어를 사용하려면 이 모드 또는 다중 계정 및 리전을 사용해야 합니다.
7. [대상(Targets)] 섹션에서 Automation을 실행할 AWS 리소스를 대상으로 지정할 방식을 선택합니다. 다음 옵션이 필요합니다.

   a. 파라미터를 선택하려면 파라미터 목록을 사용합니다. [파라미터(Parameter)] 목록의 항목은 이 절차의 시작 부분에서 선택한 Automation 실행서의 파라미터로 결정됩니다. 파라미터를 선택하여 자동화 워크플로에서 실행되는 리소스 유형을 정의할 수 있습니다.

   b. 리소스를 대상으로 지정하는 방식을 선택하려면 대상 목록을 사용합니다.

      i. 파라미터 값을 사용하여 리소스를 대상으로 지정하기로 선택한 경우 입력 파라미터 섹션에서 선택한 파라미터에 대한 값을 입력합니다.

      ii. AWS Resource Groups을 사용하여 리소스를 대상으로 지정하기로 한 경우 리소스 그룹 목록에서 그룹의 이름을 선택합니다.

      iii. 태그를 사용하여 리소스를 대상으로 지정하기로 한 경우 태그 키와 (선택 사항) 태그 값을 제공된 필드에 입력합니다. 추가를 선택합니다.

      iv. 현재 AWS 계정 및 AWS 리전의 전체 인스턴스에서 Automation 실행서를 실행하려면 [전체 인스턴스(All instances)]를 선택합니다.
8. 입력 파라미터 섹션에서 필수 입력을 지정합니다. 필요에 따라 [AutomationAssumeRole] 목록에서 IAM 서비스 역할을 선택합니다.

   > **Note**
   >
   > 입력 파라미터 섹션에서 일부 옵션은 선택할 필요가 없습니다. 그 이유는 태그 또는 리소스 그룹을 사용하여 리소스를 대상으로 지정했기 때문입니다. 예를 들어 `AWS-RestartEC2Instance` 실행서를 선택한 경우 [입력 파라미터(Input parameters)] 섹션에서 인스턴스 ID를 지정하거나 선택할 필요가 없습니다. Automation 실행 시, 지정된 태그 또는 리소스 그룹을 사용하여 다시 시작할 인스턴스를 찾을 수 있습니다.
9. 각 계정-리소스 페어 내에서 Automation을 실행할 수 있는 AWS 리소스 수를 제한하려면 속도 제어 섹션의 옵션을 사용합니다.

   동시성 섹션에서 옵션을 선택합니다.

   - 대상을 선택하여 자동화 워크플로를 동시에 실행할 수 있는 대상 수(절대 개수)를 입력합니다.
   - 백분율을 선택하여 자동화 워크플로를 동시에 실행할 수 있는 대상의 백분율을 입력합니다.

10. 오류 임계값 섹션에서 옵션을 선택합니다.

    - 오류를 선택하여 자동화를 통한 다른 리소스로의 워크플로 전송이 중지될 때까지 허용되는 오류 수(절대 개수)를 입력합니다.
    - 백분율을 선택하여 자동화를 통한 다른 리소스로의 워크플로 전송이 중지될 때까지 허용되는 백분율을 입력합니다.

11. (선택 사항) 모니터링을 위해 자동화에 적용할 CloudWatch 경보를 선택합니다. CloudWatch 경보를 자동화에 연결하려면 자동화를 시작하는 IAM 보안 주체에 `iam:createServiceLinkedRole` 작업에 대한 권한이 있어야 합니다. CloudWatch 경보에 대한 자세한 내용은 Amazon CloudWatch 경보 사용을 참조하세요. 경보가 활성화되면 자동화가 중지됩니다. AWS CloudTrail을 사용하면 추적에 API 호출이 표시됩니다.

12. 실행을 선택합니다.

속도 제어 자동화로 시작된 자동화를 보려면 탐색 창에서 Automation을 선택한 다음 [하위 자동화 표시(Show child automations)]를 선택합니다.

대상 및 속도 제어를 사용하여 자동화 실행(명령줄)

다음 절차에서는 AWS CLI(Linux 또는 Windows) 또는 AWS Tools for PowerShell에서 대상 및 속도 제어를 사용하여 자동화를 실행하는 방법을 설명합니다.

대상 및 속도 제어를 사용하여 자동화를 실행하려면

1. 아직 하지 않은 경우 AWS CLI 또는 AWS Tools for PowerShell을 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트 및 AWS Tools for PowerShell 설치를 참조하세요.

2. 다음 명령을 실행하여 문서 목록을 확인합니다.

   Linux & macOS

   ```
   aws ssm list-documents
   ```

   Windows

   ```
   aws ssm list-documents
   ```

PowerShell

```
Get-SSMDocumentList
```

사용할 실행서의 이름을 기록해 둡니다.

3. 다음 명령을 실행하여 런북에 대한 세부 정보를 봅니다. *### ##*을 세부 정보를 보려는 실행서 이름으로 바꿉니다. 또한 `--target-parameter-name` 옵션에 사용할 파라미터 이름(예: `InstanceId`)에 유의하세요. 이 파라미터는 자동화가 실행되는 리소스 유형을 결정합니다.

   Linux & macOS

   ```
   aws ssm describe-document \
       --name runbook name
   ```

   Windows

   ```
   aws ssm describe-document ^
       --name runbook name
   ```

   PowerShell

   ```
   Get-SSMDocumentDescription `
       -Name runbook name
   ```

4. 실행할 대상 및 속도 제어 옵션을 사용하는 명령을 생성합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   태그를 사용하여 대상 지정

   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%"
```

파라미터 값을 사용하여 대상 지정

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%"
```

AWS Resource Groups을 사용하여 대상 지정

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%"
```

현재 AWS 계정 및 AWS 리전의 모든 Amazon EC2 인스턴스를 대상으로 지정

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%"
```

이 명령으로 실행 ID가 반환됩니다. 클립보드에 이 ID를 복사합니다. 이 ID를 사용하여 자동화 상태를 확인할 수 있습니다.

Linux & macOS

```
{
    "AutomationExecutionId": "a4a3c0e9-7efd-462a-8594-01234EXAMPLE"
}
```

Windows

```
{
    "AutomationExecutionId": "a4a3c0e9-7efd-462a-8594-01234EXAMPLE"
}
```

PowerShell

```
a4a3c0e9-7efd-462a-8594-01234EXAMPLE
```

5. 다음 명령을 실행하여 자동화를 봅니다. 각 *### ## ID*를 자신의 정보를 바꿉니다.

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. 자동화 진행 상황에 대한 세부 정보를 보려면 다음 명령을 실행합니다. 각 *### ## ID*를 자신의 정보를 바꿉니다.

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
```

시스템은 다음과 같은 정보를 반환합니다.

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**
>
> 콘솔에서 자동화 상태를 모니터링할 수도 있습니다. 자동화 실행 목록에서 방금 실행한 자동화를 선택한 후 실행 단계 탭을 선택합니다. 이 탭은 자동화 작업의 상태를 보여줍니다.

자동화를 위한 대상 매핑

자동화의 대상 리소스를 빠르게 정의하려면 `Targets` 파라미터를 사용합니다. 예를 들어 관리형 인스턴스를 다시 시작하는 자동화를 실행하려는 경우 콘솔에서 인스턴스 ID 수십 개를 직접 선택하거나 명령에 이러한 ID를 입력하는 대신에, `Targets` 파라미터를 사용하여 Amazon Elastic Compute Cloud(Amazon EC2) 태그를 지정함으로써 대상 인스턴스를 지정할 수 있습니다.

대상을 사용하는 자동화를 실행하는 경우 AWS Systems Manager는 각 대상에 대해 하위 자동화를 생성합니다. 예를 들어 태그를 지정하여 Amazon Elastic Block Store(Amazon EBS) 볼륨을 대상으로 지정하고 해당 태그가 100개의 Amazon EBS 볼륨으로 해석되면 Systems Manager는 100개의 하위 자동화를 생성합니다. 모든 하위 자동화가 최종 상태에 도달하면 상위 자동화가 완료된 것입니다.

> **Note**
>
> 런타임 시 지정하는 모든 `input parameters`(콘솔의 [입력 파라미터(Input parameters)] 섹션에서 또는 명령줄에서 `parameters` 옵션 사용)는 모든 하위 자동화에서 자동으로 처리됩니다.

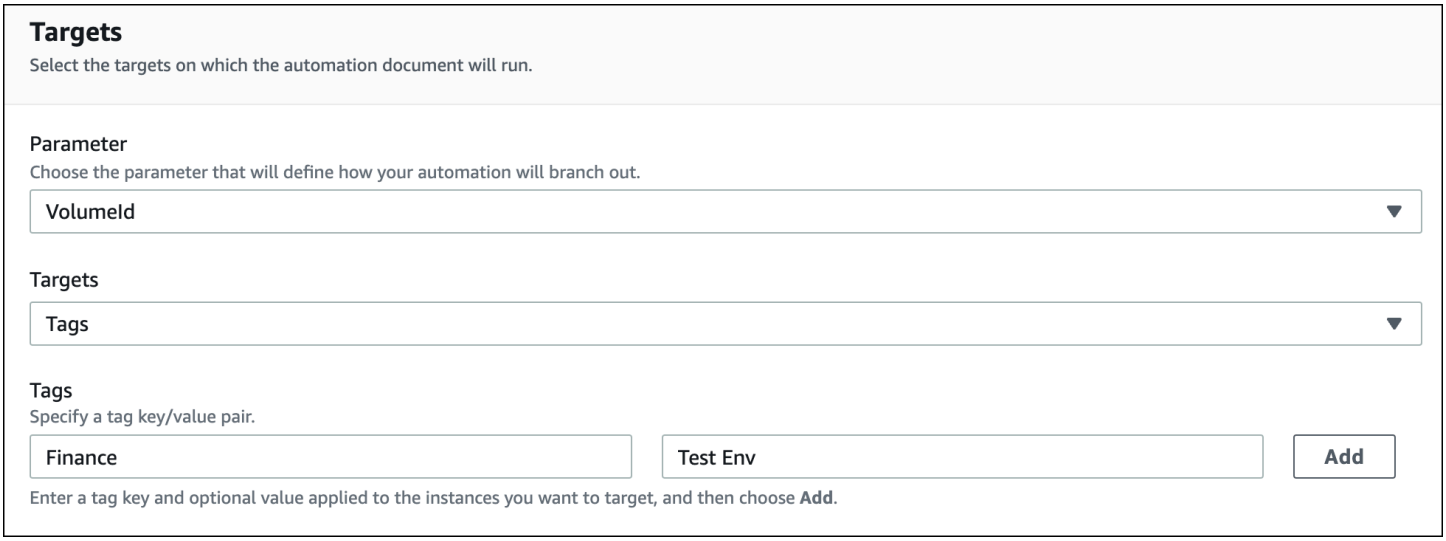
태그, Resource Groups 및 파라미터 값을 사용하여 자동화를 위한 대상 리소스를 지정할 수 있습니다. 또한 `TargetMaps` 옵션을 사용하여 명령줄 또는 파일에서 여러 파라미터 값을 대상으로 지정할 수 있습니다. 다음 단원에서는 이러한 각각의 대상 지정 옵션에 대해 자세히 설명합니다.

태그를 대상으로 지정

단일 태그를 자동화 대상으로 지정할 수 있습니다. Amazon Elastic Compute Cloud(Amazon EC2) 및 Amazon Relational Database Service(Amazon RDS) 인스턴스, Amazon Elastic Block Store(Amazon EBS) 볼륨 및 스냅샷, Resource Groups, Amazon Simple Storage Service(Amazon S3) 버킷을 비롯한 많은 AWS 리소스가 태그를 지원합니다. 태그를 대상으로 지정하여 AWS 리소스에서 자동화를 신속히 실행할 수 있습니다. 태그는 Operating_System:Linux 또는 Department:Finance와 같은 키-값 페어입니다. 리소스에 특정 이름을 할당할 경우 "Name"을 키로 사용하고 리소스 이름을 값으로 사용할 수도 있습니다.

자동화 대상으로 사용할 태그를 지정할 때 대상 파라미터도 지정합니다. 대상 파라미터에는 `TargetParameterName` 옵션이 사용됩니다. 대상 파라미터를 선택하면 자동화에서 실행되는 리소스 유형을 정의할 수 있습니다. 태그를 사용하여 지정하는 대상 파라미터는 실행서에 정의된 유효한 파라미터여야 합니다. 예를 들어 태그를 사용하여 수십 개 EC2 인스턴스를 대상으로 지정하려는 경우 `InstanceId` 대상 파라미터를 선택합니다. 이 파라미터를 선택하여 자동화에 대해 인스턴스를 리소스 유형으로 정의할 수 있습니다. 사용자 지정 런북을 생성할 때 대상 유형을 `/AWS::EC2::Instance`로 지정하여 인스턴스만 사용되도록 할 수 있습니다. 그렇지 않으면 태그가 동일한 모든 리소스가 대상이 됩니다. 태그를 사용하여 인스턴스를 대상으로 지정할 때 종료된 인스턴스가 포함될 수 있습니다.

다음 스크린샷에서는 `AWS-DetachEBSVolume`실행서를 사용합니다. 논리적 대상 파라미터는 `VolumeId`입니다.



`AWS-DetachEBSVolume` 실행서에는 `/AWS::EC2::Volume`으로 설정된 [대상 유형(Target type)]이라는 특수 속성도 포함되어 있습니다. 다시 말해서 태그-키 페어 `Finance:TestEnv`가 다양한 리소스 유형(예를 들면 EC2 인스턴스, Amazon EBS 볼륨, Amazon EBS 스냅샷)을 반환하는 경우 Amazon EBS 볼륨만 사용됩니다.

Important

대상 파라미터 이름은 대/소문자를 구분합니다. AWS Command Line Interface(AWS CLI) 또는 AWS Tools for Windows PowerShell을 사용하여 자동화를 실행할 경우 실행서에 정의된 것과 똑같은 대상 파라미터 이름을 입력해야 합니다. 그렇지 않으면 `InvalidAutomationExecutionParametersException` 오류가 반환됩니다. DescribeDocument API 작업을 사용하여 특정 실행서의 사용 가능한 대상 파라미터에 대한 정

보를 볼 수 있습니다. 다음은 AWS-DeleteSnapshot 문서에 대한 정보를 제공하는 AWS CLI 명령의 예입니다.

```
aws ssm describe-document \
    --name AWS-DeleteSnapshot
```

다음은 태그를 사용하여 대상 리소스를 지정하는 AWS CLI 명령의 예입니다.

예 1: 키-값 페어를 통해 태그를 대상으로 지정하여 Amazon EC2 인스턴스 다시 시작

이 예에서는 키가 Department이고 값이 HumanResources인 태그로 지정된 모든 Amazon EC2 인스턴스를 다시 시작합니다. 대상 파라미터에는 실행서의 InstanceId 파라미터가 사용됩니다. 이 예제에서는 자동화 서비스 역할을 사용(또는 역할 수임)함으로써 추가 파라미터를 사용하여 자동화를 실행합니다.

```
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"
```

예 2: 키-값 페어를 통해 태그를 대상으로 지정하여 Amazon EBS 스냅샷 삭제

다음 예에서는 AWS-DeleteSnapshot 실행서를 사용하여 키가 Name이고 값이 January2018Backups인 모든 스냅샷을 삭제합니다. 대상 파라미터에는 VolumeId 파라미터가 사용됩니다.

```
aws ssm start-automation-execution \
    --document-name AWS-DeleteSnapshot \
    --targets Key=tag:Name,Values=January2018Backups \
    --target-parameter-name VolumeId
```

AWS Resource Groups을 대상으로 지정

단일 AWS 리소스 그룹을 자동화 대상으로 지정할 수 있습니다. Systems Manager는 대상 리소스 그룹의 모든 객체에 대한 하위 자동화를 생성합니다.

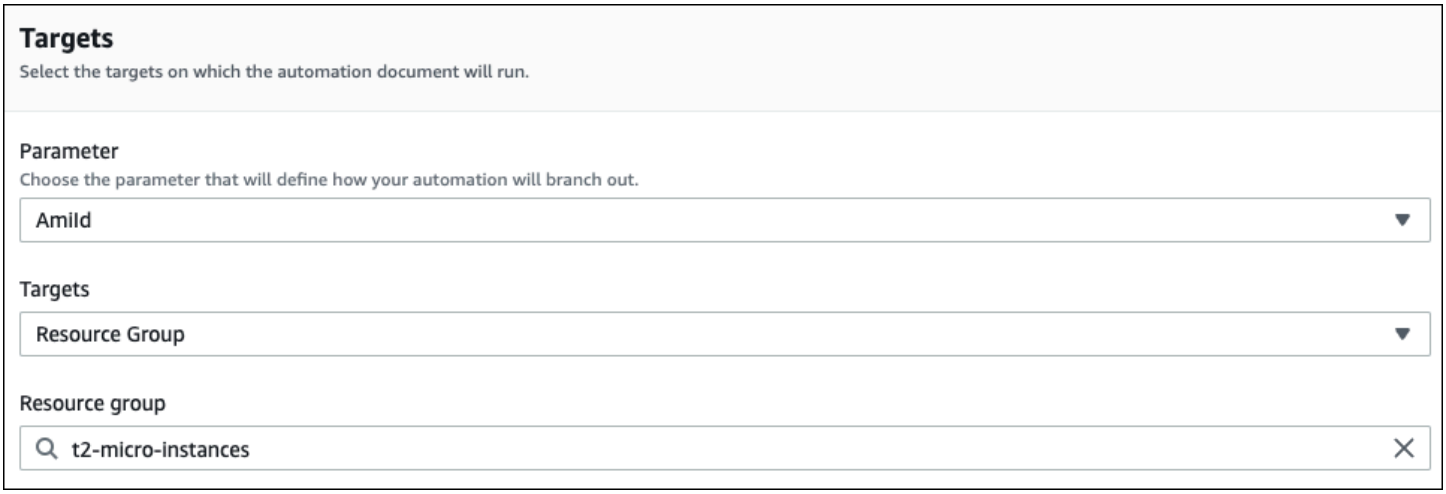
예를 들면 Resource Groups 중 하나가 PatchedAMIs라고 가정해 보겠습니다. 이 리소스 그룹에는 일상적으로 패치되는 25개 Windows Amazon Machine Images(AMIs)의 목록이 포함되어 있습니다. AWS-CreateManagedWindowsInstance 실행서를 사용하는 자동화를 실행하고 이 리소스 그룹을

대상으로 지정하는 경우 Systems Manager는 25개 AMIs 각각에 대해 하위 자동화를 생성합니다. 다시 말해서 이 자동화는 PatchedAMIs 리소스 그룹을 대상으로 지정하여 패치된 AMIs 목록에서 25개 인스턴스를 생성합니다. 모든 하위 자동화가 처리를 완료하거나 최종 상태에 도달하면 상위 자동화가 완료된 것입니다.

다음 AWS CLI 명령은 PatchAMIs 리소스 그룹 예제에 적용됩니다. 이 명령은 `--target-parameter-name` 옵션으로 AmiId 파라미터를 사용합니다. 이 명령은 각 AMI에서 생성할 인스턴스 유형을 정의하는 추가 파라미터를 포함하지 않습니다. `AWS-CreateManagedWindowsInstance` 실행서는 기본적으로 t2.medium 인스턴스 유형이므로 이 명령은 25개의 Windows Server용 t2.medium Amazon EC2 인스턴스를 생성합니다.

```
aws ssm start-automation-execution \
    --document-name AWS-CreateManagedWindowsInstance \
    --targets Key=ResourceGroup,Values=PatchedAMIs  \
    --target-parameter-name AmiId
```

다음 콘솔 예제에서는 t2-micro-instances라는 리소스 그룹이 사용됩니다.



파라미터 값을 대상으로 지정

파라미터 값을 대상으로 지정할 수도 있습니다. `ParameterValues`를 키로 입력하고 나서 자동화를 실행할 특정 리소스 값을 입력합니다. 여러 값을 지정할 경우 Systems Manager는 지정된 각 값에 대해 하위 자동화를 실행합니다.

예를 들어 실행서에 InstanceID 파라미터가 포함되어 있다고 가정해 보겠습니다. Automation 실행 시 InstanceID 파라미터의 값을 대상으로 지정할 경우 Systems Manager는 지정한 각 인스턴스 ID 값마다 하위 자동화를 하나 실행합니다. 이 자동화가 각각의 지정된 인스턴스 실행을 완료하거나 실행에 실패하면 상위 자동화가 완료된 것입니다. 최대 50개 파라미터 값을 대상으로 지정할 수 있습니다.

다음 예에서는 AWS-CreateImage 실행서를 사용합니다. 지정된 대상 파라미터 이름은 InstanceId입니다. 이 키에는 ParameterValues가 사용됩니다. 두 Amazon EC2 인스턴스 ID가 값이 됩니다. 이 명령은 각 인스턴스마다 자동화를 하나 생성하며, 이 워크플로는 각 인스턴스에서 AMI를 생성합니다.

```
aws ssm start-automation-execution
    --document-name AWS-CreateImage \
    --target-parameter-name InstanceId \
    --targets Key=ParameterValues,Values=i-02573cafcfEXAMPLE,i-0471e04240EXAMPLE
```

> **Note**
>
> AutomationAssumeRole은 유효한 파라미터가 아닙니다. 파라미터 값을 대상으로 지정하는 자동화를 실행할 때 이 항목을 선택하지 않습니다.

파라미터 값 맵을 대상으로 지정

TargetMaps 옵션은 ParameterValues를 대상으로 지정할 수 있는 기능을 확장합니다. 명령줄에서 TargetMaps를 사용하여 파라미터 값 배열을 입력할 수 있습니다. 명령줄에 최대 50개 파라미터 값을 지정할 수 있습니다. 50개가 넘는 파라미터 값을 지정하여 명령을 실행하려면 JSON 파일에 값을 입력할 수 있습니다. 그런 다음 명령줄에서 해당 파일을 호출할 수 있습니다.

> **Note**
>
> TargetMaps 옵션은 콘솔에서 지원되지 않습니다.

명령에서 TargetMaps 옵션을 사용하면 다음 형식을 사용하여 여러 파라미터 값을 지정할 수 있습니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
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”
```

TargetMaps 옵션에 대해 50개가 넘는 파라미터 값을 입력하려면 다음 JSON 형식을 사용하여 파일에 값을 지정합니다. JSON 파일을 사용하면 여러 파라미터 값 지정 시 가독성도 향상됩니다.

```
[

    {“parameter”: "value", “parameter 2”: "value", “parameter 3”: "value"},

    {“parameter 4”: "value", “parameter 5”: "value", "parameter 6": "value"}

]
```

파일을 .json 파일 확장명으로 저장합니다. 다음 명령을 사용하여 파일을 호출할 수 있습니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
aws ssm start-automation-execution \
    --document-name runbook name \
    --parameters input parameters \
    --target-maps path to file/file name.json
```

또한 버킷에서 데이터를 읽을 수 있는 권한이 있는 한 Amazon Simple Storage Service(Amazon S3) 버킷에서 파일을 다운로드할 수도 있습니다. 다음 명령 형식을 사용합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
aws ssm start-automation-execution \
    --document-name runbook name \
    --target-maps http://DOC-EXAMPLE-BUCKET.s3.amazonaws.com/file_name.json
```

다음은 TargetMaps 옵션을 이해하는 데 도움이 될 예제 시나리오입니다. 이 시나리오에서 사용자는 다양한 AMIs에서 다양한 유형의 Amazon EC2 인스턴스를 생성하려고 합니다. 이 태스크를 수행하기 위해 사용자는 AMI_Testing이라는 실행서를 생성합니다. 이 실행서는 instanceType 및 imageId라는 두 가지 입력 파라미터를 정의합니다.

```
{
  "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"
  ]
}
```

그런 다음 사용자는 AMI_instance_types.json이라는 파일에 다음과 같은 대상 파라미터 값을 지정합니다.

```
[
  {
    "instanceType" : ["t2.micro"],
    "imageId" : ["ami-b70554c8"]
  },
  {
    "instanceType" : ["t2.small"],
    "imageId" : ["ami-b70554c8"]
  },
  {
    "instanceType" : ["t2.medium"],
    "imageId" : ["ami-cfe4b2b0"]
  },
```

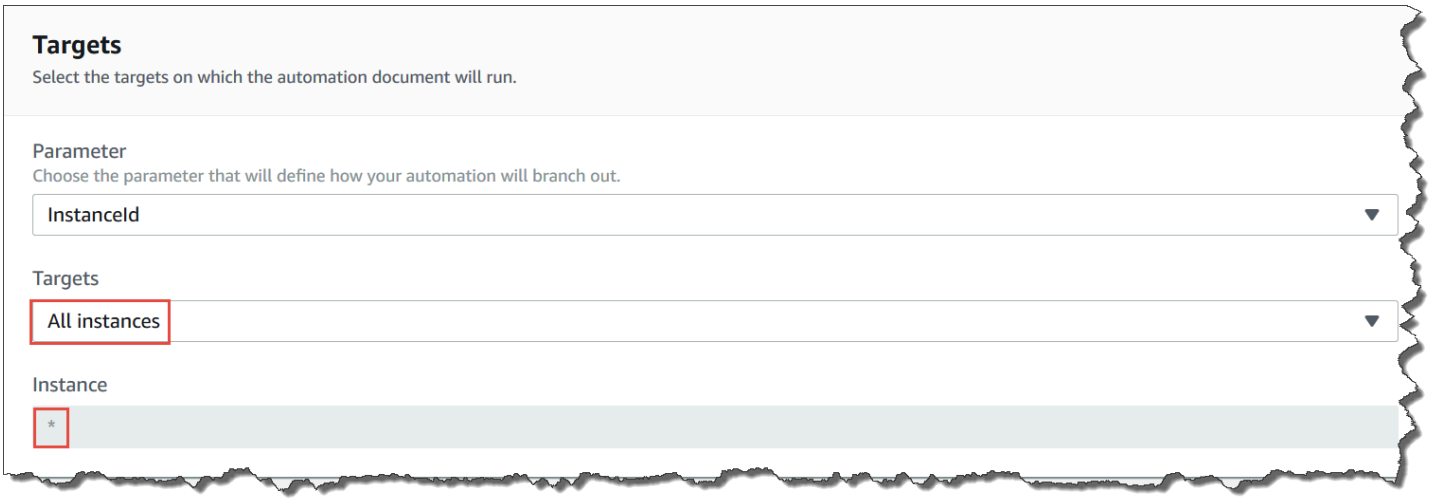
```
  {
    "instanceType" : ["t2.medium"],
    "imageId" : ["ami-cfe4b2b0"]
  },
  {
    "instanceType" : ["t2.medium"],
    "imageId" : ["ami-cfe4b2b0"]
  }
]
```

사용자는 다음 명령을 통해 이 자동화를 실행하고 `AMI_instance_types.json`에 정의된 5개 EC2 인스턴스를 생성할 수 있습니다.

```
aws ssm start-automation-execution \
    --document-name AMI_Testing \
    --target-parameter-name imageId \
    --target-maps file:///home/TestUser/workspace/runinstances/AMI_instance_types.json
```

모든 Amazon EC2 인스턴스를 대상으로 지정

대상 목록에서 모든 인스턴스를 선택하여 현재 AWS 계정 및 AWS 리전의 모든 Amazon EC2 인스턴스에 대해 자동화를 실행할 수 있습니다. 예를 들어 AWS 계정 및 현재 AWS 리전의 모든 Amazon EC2 인스턴스를 다시 시작하려면 **`AWS-RestartEC2Instance`** 런북을 선택한 다음에 대상 목록에서 모든 인스턴스를 선택하면 됩니다.



모든 인스턴스(All instances)를 선택하면 Systems Manager가 인스턴스(Instance) 필드를 별표(*)로 채우고 변경할 수 없게 됩니다(필드가 회색으로 표시됨). 입력 파라미터(Input parameters) 필드의 InstanceId 필드도 사용할 수 없게 됩니다. 모든 인스턴스를 대상으로 선택하면 이러한 필드는 변경할 수 없게 됩니다.

대규모 자동화 제어

동시성 값과 오류 임계값을 지정하여 AWS 리소스 플릿에서 자동화의 배포를 제어할 수도 있습니다. 동시성 및 오류 임계값을 통칭하여 속도 제어라고 합니다.

동시성

동시성을 통해 자동화를 동시에 실행하도록 허용된 리소스 수를 지정할 수 있습니다. 동시성은 자동화 처리 시 리소스에 대한 영향이나 중단 시간을 제한하는 데 도움이 됩니다. 리소스의 절대 개수(예: 20개)를 지정하거나 대상 집합의 비율(예: 10%)을 지정할 수 있습니다.

대기 중인 시스템은 단일 리소스로 자동화를 전달하고 최초 호출이 완료될 때까지 기다렸다가 이 자동화를 다른 두 리소스로 전송합니다. 시스템은 동시성 값이 충족될 때까지 기하급수적으로 이 자동화를 더 많은 리소스로 전송합니다.

오류 임계값

오류 임계값을 사용하여 AWS Systems Manager가 자동화를 다른 리소스로 전송하는 것을 중지하기 전에 실패할 수 있는 자동화 수를 지정합니다. 오류의 절대 개수(예: 10개)를 지정하거나 대상 집합의 비율(예: 10%)을 지정할 수 있습니다.

예를 들어 오류 수로 절대 개수 3을 지정할 경우 네 번째 오류가 수신되면 자동화 실행이 중지됩니다. 0을 지정하면 첫 번째 오류 결과가 반환된 후 추가 대상에서 자동화 실행이 중지됩니다.

예를 들어 자동화를 50개 인스턴스로 전송하고 오류 임계값을 10%로 설정하면 다섯 번째 오류가 수신되면 추가 인스턴스로의 명령 전송이 중지됩니다. 오류 임계값에 도달했을 때 자동화를 이미 실행 중인 호출은 완료될 수도 있지만, 이러한 자동화 중 일부가 실패할 수도 있습니다. 오류 수가 오류 임계값에 지정된 수보다 많지 않다고 확신하는 경우 [동시성(Concurrency)] 값을 1로 설정하면 자동화가 한 번에 하나씩 진행됩니다.

여러 AWS 리전 및 계정에서 자동화 실행

중앙 계정을 통해 여러 AWS 리전 및 AWS 계정 또는 AWS Organizations 조직 단위(OU)에서 AWS Systems Manager 자동화를 실행할 수 있습니다. Automation은 AWS Systems Manager의 기능입니다. 여러 리전 및 계정이나 OU에서 자동화를 실행하면 AWS 리소스를 관리하는 데 드는 시간이 절약될 뿐만 아니라 컴퓨팅 환경의 보안도 향상됩니다.

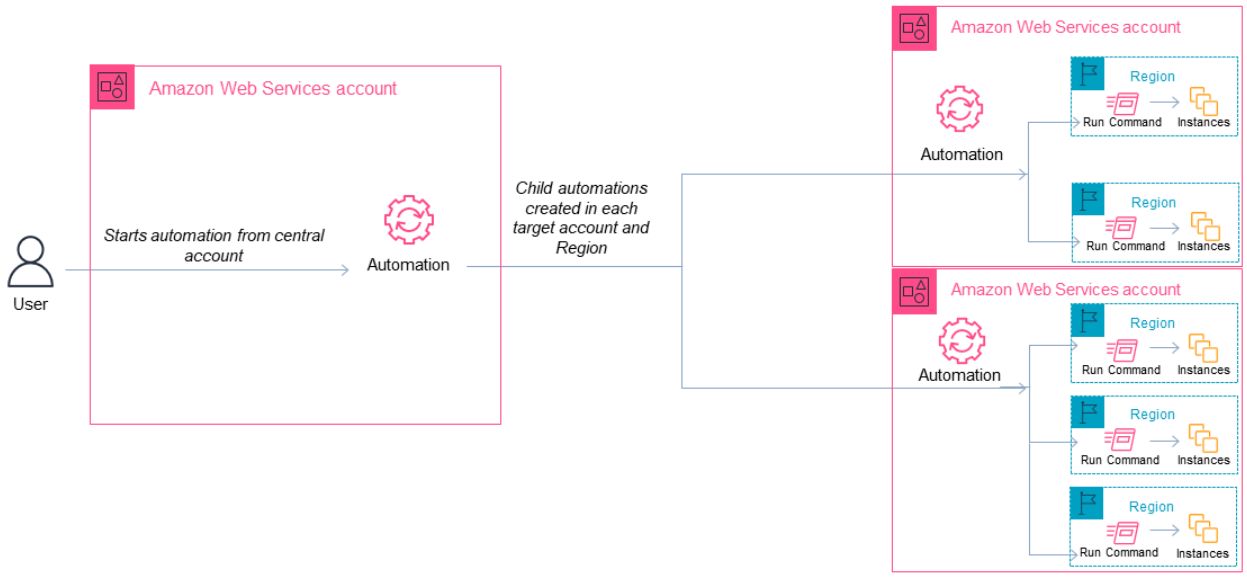
예를 들어, 자동화 런북을 사용하여 다음 작업을 수행할 수 있습니다.

- 패치 및 보안 업데이트를 중앙에서 구현합니다.
- VPC 구성 또는 Amazon S3 버킷 정책의 규정 준수 편차를 수정합니다.

- Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스와 같은 리소스를 대규모로 관리합니다.

다음 다이어그램은 중앙 계정 하나를 통해 여러 리전 및 계정에서 `AWS-RestartEC2Instances` 런북을 실행 중인 사용자 예를 보여줍니다. 이 자동화는 대상 리전 및 계정에서 지정된 태그를 사용하여 인스턴스를 찾아냅니다.

> **Note**
>
> 여러 리전 및 계정 간에 자동화를 실행할 경우 AWS 리소스 그룹의 이름이나 태그를 사용하여 리소스를 대상으로 지정합니다. 리소스 그룹은 각 대상 계정 및 리전에 있어야 합니다. 리소스 그룹 이름은 각 대상 계정 및 리전에서 동일해야 합니다. 태그가 지정되지 않았거나 지정된 리소스 그룹에 포함되지 않은 리소스에서는 자동화가 실행되지 않습니다.



Automation을 위한 중앙 계정 선택

OU에서 자동화를 실행하려면 중앙 계정에 OU의 모든 계정을 나열할 수 있는 권한이 있어야 합니다. 이는 위임된 관리자 계정 또는 조직의 관리 계정에서만 가능합니다. AWS Organizations 모범 사례를 따르고 위임된 관리자 계정을 사용하는 것이 좋습니다. AWS Organizations 모범 사례에 대한 자세한

내용은 AWS Organizations 사용 설명서의 관리 계정의 모범 사례를 참조하세요. Systems Manager에 대한 위임된 관리자 계정을 생성하려면 다음 예제와 같이 AWS CLI와 함께 `register-delegated-administrator` 명령을 사용할 수 있습니다.

```
aws organizations register-delegated-administrator \
    --account-id delegated admin account ID \
    --service-principal ssm.amazonaws.com
```

AWS Organizations에서 관리하지 않는 여러 계정에 걸쳐 자동화를 실행하려면 자동화 관리 전용 계정을 생성하는 것이 좋습니다. 전용 계정에서 모든 크로스 계정 자동화를 실행하면 IAM 권한 관리 및 문제 해결 작업이 간소화되고 운영과 관리를 구분할 수 있습니다. 이 접근 방식은 AWS Organizations를 사용하지만 OU가 아닌 개별 계정만 대상으로 하려는 경우에도 권장됩니다.

자동화 실행 방식

여러 리전 및 계정 또는 OU에서의 자동화 실행은 다음과 같이 작동합니다.

1. 모든 리전 및 계정 또는 OU에서 자동화를 실행하려는 모든 리소스에 동일한 태그가 사용되는지 확인합니다. 동일한 태그가 사용되지 않는 경우 이들 리소스를 AWS 리소스 그룹에 추가하고 해당 그룹을 대상으로 지정합니다. 자세한 내용은 AWS Resource Groups 및 태그 사용 설명서의 리소스 그룹이란 무엇인가요?를 참조하세요.
2. Automation 중앙 계정으로 구성할 계정에 로그인합니다.
3. 이 주제의 다중 리전 및 다중 계정 자동화를 위한 관리 계정 권한 설정 절차에 따라 다음 IAM 역할을 생성합니다.
   - **`AWS-SystemsManager-AutomationAdministrationRole`** - 이 역할은 사용자에게 여러 계정 및 OU에서 자동화를 실행할 수 있는 권한을 부여합니다.
   - **`AWS-SystemsManager-AutomationExecutionRole`** - 이 역할은 사용자에게 대상 계정에서 자동화를 실행할 수 있는 권한을 부여합니다.
4. 자동화를 실행할 런북, 리전 및 계정 또는 OU를 선택합니다.

   > **Note**
   >
   > Automation은 OU를 통해 재귀적으로 실행되지 않습니다. 대상 OU에 원하는 계정이 포함되어 있는지 확인합니다. 사용자 지정 런북을 선택하는 경우 런북을 모든 대상 계정과 공유해야 합니다. 실행서 공유에 대한 자세한 내용은 SSM 문서 공유 섹션을 참조하세요. 공유 런북 사용에 대한 자세한 내용은 공유 SSM 문서 사용 섹션을 참조하세요.

5. 자동화를 실행합니다.

> **Note**
>
> 여러 리전, 계정 또는 OU에서 자동화를 실행하는 경우 기본 계정에서 실행하는 자동화는 각 대상 계정에서 하위 자동화를 시작합니다. 기본 계정의 자동화에는 각 대상 계정에 대해 `aws:executeAutomation` 단계가 있습니다. 2019년 3월 20일 이후에 시작된 새 리전에서 자동화를 시작하고 기본적으로 활성화된 리전을 대상으로 하는 경우 자동화가 실패합니다. 기본적으로 활성화된 리전에서 자동화를 시작하고 활성화한 리전을 대상으로 하는 경우 자동화가 성공적으로 실행됩니다.

6. AWS Systems Manager 콘솔 또는 AWS CLI에서 GetAutomationExecution, DescribeAutomationStepExecutions, 및 DescribeAutomationExecutions API 작업을 사용하여 자동화 진행 상황을 모니터링합니다. 기본 계정의 자동화 단계 출력은 하위 자동화의 `AutomationExecutionId`가 됩니다. 대상 계정에서 생성된 하위 자동화의 출력을 보려면 요청에 적절한 계정, 리전 및 `AutomationExecutionId`를 지정해야 합니다.

다중 리전 및 다중 계정 자동화를 위한 관리 계정 권한 설정

다음 절차에 따라 AWS CloudFormation을 사용하여 Systems Manager Automation 다중 리전 및 다중 계정 실행에 필요한 IAM 역할을 생성합니다. 이 절차에서는 **`AWS-SystemsManager-AutomationAdministrationRole`** 역할을 생성하는 방법을 설명합니다. Automation 중앙 계정에서 이 역할을 생성하기만 하면 됩니다. 이 절차에서는 **`AWS-SystemsManager-AutomationExecutionRole`** 역할을 생성하는 방법도 설명합니다. 다중 리전 및 다중 계정 자동화를 실행할 대상으로 지정할 모든 계정에서 이 역할을 생성해야 합니다. 다중 리전 및 다중 계정 자동화를 실행하기 위해 대상으로 지정하려는 계정에서 **`AWS-SystemsManager-AutomationExecutionRole`** 역할을 생성하려면 AWS CloudFormation StackSets를 사용하는 것이 좋습니다.

AWS CloudFormation을 사용하여 다중 리전 및 다중 계정 자동화에 필요한 IAM 관리 역할 생성

1. AWS-SystemsManager-AutomationAdministrationRole.zip을 다운로드하고 압축을 해제합니다. 또는 AWS Organizations에서 계정을 관리하는 경우 AWS-SystemsManager-AutomationAdministrationRole (org).zip을 다운로드하고 압축을 풉니다. 이 파일에는 `AWS-SystemsManager-AutomationAdministrationRole.yaml` AWS CloudFormation 템플릿 파일이 들어 있습니다.
2. AWS CloudFormation 콘솔(https://console.aws.amazon.com/cloudformation)을 엽니다.
3. Create stack(스택 생성)을 선택합니다.

4. 템플릿 지정(Specify template) 섹션에서 템플릿 업로드(Upload a template)를 선택합니다.
5. 찾아보기(Choose file)를 선택한 다음 `AWS-SystemsManager-AutomationAdministrationRole.yaml` AWS CloudFormation 템플릿 파일을 선택합니다.
6. 다음을 선택합니다.
7. 스택 세부 정보 지정 페이지의 스택 이름 필드에 이름을 입력합니다.
8. Next(다음)를 선택합니다.
9. 구성 스택 옵션 페이지에서 사용할 옵션의 값을 입력합니다. 다음을 선택합니다.
10. 검토 페이지에서 아래로 스크롤하여 AWS CloudFormation이 IAM 리소스를 생성할 수 있다는 것을 알고 있음 옵션을 선택합니다.
11. 스택 생성을 선택합니다.

AWS CloudFormation에 약 3분간 CREATE_IN_PROGRESS 상태가 표시됩니다. 상태가 CREATE_COMPLETE로 변경됩니다.

다중 리전 및 다중 계정 자동화를 실행할 대상으로 지정할 모든 계정에서 다음 절차를 반복해야 합니다.

AWS CloudFormation을 사용하여 다중 리전 및 다중 계정 자동화에 필요한 IAM 자동화 역할 생성

1. AWS-SystemsManager-AutomationExecutionRole.zip를 다운로드합니다. 또는 AWS Organizations에서 계정을 관리하는 경우 AWS-SystemsManager-AutomationExecutionRole (org).zip을 다운로드하고 압축을 풉니다. 이 파일에는 `AWS-SystemsManager-AutomationExecutionRole.yaml` AWS CloudFormation 템플릿 파일이 들어 있습니다.
2. AWS CloudFormation 콘솔(https://console.aws.amazon.com/cloudformation)을 엽니다.
3. Create stack(스택 생성)을 선택합니다.
4. 템플릿 지정(Specify template) 섹션에서 템플릿 업로드(Upload a template)를 선택합니다.
5. 찾아보기(Choose file)를 선택한 다음 `AWS-SystemsManager-AutomationExecutionRole.yaml` AWS CloudFormation 템플릿 파일을 선택합니다.
6. 다음을 선택합니다.
7. 스택 세부 정보 지정 페이지의 스택 이름 필드에 이름을 입력합니다.
8. Parameters(파라미터) 섹션의 AdminAccountId 필드에 Automation 중앙 계정의 ID를 입력합니다.
9. AWS Organizations 환경에 대해 이 역할을 설정하는 경우 섹션에 OrganizationID라는 다른 필드가 있습니다. AWS 조직의 ID를 입력합니다.

10. 다음을 선택합니다.
11. 구성 스택 옵션 페이지에서 사용할 옵션의 값을 입력합니다. 다음을 선택합니다.
12. 검토 페이지에서 아래로 스크롤하여 AWS CloudFormation이 IAM 리소스를 생성할 수 있다는 것을 알고 있음 옵션을 선택합니다.
13. 스택 생성을 선택합니다.

AWS CloudFormation에 약 3분간 CREATE_IN_PROGRESS 상태가 표시됩니다. 상태가 CREATE_COMPLETE로 변경됩니다.

여러 리전 및 계정에서 Automation 실행(콘솔)

다음 절차에서는 Systems Manager 콘솔을 사용하여 Automation 관리 계정을 통해 여러 리전 및 계정에서 자동화를 실행하는 방법을 설명합니다.

시작하기 전 준비 사항

다음 절차를 완료하기 전에 다음 정보에 유의하세요.

- 다중 리전 또는 다중 계정 자동화를 실행하는 데 사용하는 사용자 또는 역할은 `AWS-SystemsManager-AutomationAdministrationRole` 역할에 대한 `iam:PassRole` 권한이 있어야 합니다.
- 자동화를 실행할 AWS 계정 ID 또는 OU.
- 자동화를 실행할 위치인 Systems Manager에서 지원하는 리전.
- 자동화를 실행할 위치인, 태그 키 및 태그 값 또는 리소스 그룹 이름.

여러 리전 및 계정에서 자동화를 실행하려면

1. AWS Systems Manager 콘솔https://console.aws.amazon.com/systems-manager/을 엽니다.
2. 탐색 창에서 Automation(자동화)을 선택한 후 Execute automation(자동화 실행)을 선택합니다.
3. [Automation 문서(Automation document)] 목록에서 실행서를 선택합니다. 문서 카테고리 창에서 옵션을 1개 이상 선택하여 SSM 문서를 목적에 따라 필터링합니다. 자신이 소유한 실행서를 보려면 [내 소유(Owned by me)] 탭을 선택합니다. 자신의 계정과 공유하고 있는 실행서를 보려면 [나와 공유됨(Shared with me)] 탭을 선택합니다. 모든 실행서를 보려면 [모든 문서(All documents)] 탭을 선택합니다.

> **Note**
>
> 실행서 이름을 선택하여 실행서에 대한 정보를 볼 수 있습니다.

4. 문서 세부 정보 섹션에서 문서 버전이 실행할 버전으로 설정되었는지 확인합니다. 이 시스템에는 다음 버전 옵션이 포함되어 있습니다.

   - 런타임 시 기본 버전 - Automation 런북이 정기적으로 업데이트되며 새 기본 버전이 할당된 경우 이 옵션을 선택합니다.
   - 런타임 시 최신 버전 - Automation 런북이 정기적으로 업데이트되며 최근에 업데이트된 버전을 실행하려는 경우 이 옵션을 선택합니다.
   - 1(기본값) - 문서의 최초 버전을 실행하려면 이 옵션을 선택합니다(기본값).

5. 다음을 선택합니다.
6. 자동화 문서 실행 페이지에서 다중 계정 및 리전을 선택합니다.
7. [대상 계정 및 리전(Target accounts and Regions)] 섹션에서 [계정 및 조직 단위(OU)(Accounts and organizational (OUs))] 필드를 사용하여 자동화를 실행할 서로 다른 AWS 계정 또는 AWS 조직 단위(OU)를 지정합니다. 쉼표를 사용하여 여러 계정 또는 OU를 구분합니다.
8. [AWS 리전] 목록을 사용하여 자동화를 실행할 하나 이상의 리전을 선택합니다.
9. [다중 리전 및 계정 속도 제어(Multi-Region and account rate control)] 옵션을 사용하여 자동화를 제한된 리전 수에서 실행 중인 제한된 계정 수로 제한합니다. 이러한 옵션은 자동화를 실행할 수 있는 AWS 리소스 수를 제한하지 않습니다.

   a. [위치(계정-리전 페어) 동시성(Location (account-Region pair) concurrency)] 섹션에서 옵션을 선택하여 여러 계정 및 리전에서 동시에 실행할 수 있는 자동화 수를 제한합니다. 예를 들어 4개 AWS 리전에 있는 5개 AWS 계정에서 자동화를 실행하도록 선택하는 경우 Systems Manager는 총 20개 계정-리전 페어에서 자동화를 실행합니다. 이 옵션을 사용하여 자동화가 2개 계정-리전 페어에서만 동시에 실행되도록 **2**와 같은 절대 개수를 지정할 수 있습니다. 또는 동시에 실행할 수 있는 계정-리전 페어 백분율을 지정할 수 있습니다. 예를 들어 20개 계정-리전 페어가 있을 때 20%를 지정할 경우 자동화가 최대 5개 계정-리전 페어에서 동시에 실행됩니다.

      - [대상(targets)]을 선택하여 자동화를 동시에 실행할 수 있는 계정-리전 페어 수(절대 개수)를 입력합니다.
      - [백분율(percent)]을 선택하여 자동화를 동시에 실행할 수 있는 총 계정-리전 페어 수의 백분율을 입력합니다.

b. Error threshold(오류 임계값) 섹션에서 옵션을 선택합니다.

- [오류(errors)]를 선택하여 Automation을 통한 다른 리소스로의 자동화 전송이 중지될 때까지 허용되는 오류 수(절대 개수)를 입력합니다.
- [퍼센트(percent)]를 선택하여 Automation을 통한 다른 리소스로의 자동화 전송이 중지될 때까지 허용되는 백분율을 입력합니다.

10. [대상(Targets)] 섹션에서 Automation을 실행할 AWS 리소스를 대상으로 지정할 방식을 선택합니다. 다음 옵션이 필요합니다.

a. 파라미터를 선택하려면 파라미터 목록을 사용합니다. [파라미터(Parameter)] 목록의 항목은 이 절차의 시작 부분에서 선택한 Automation 실행서의 파라미터로 결정됩니다. 파라미터를 선택하여 자동화 워크플로에서 실행되는 리소스 유형을 정의할 수 있습니다.

b. 리소스를 대상으로 지정하는 방식을 선택하려면 대상 목록을 사용합니다.

i. 파라미터 값을 사용하여 리소스를 대상으로 지정하기로 선택한 경우 입력 파라미터 섹션에서 선택한 파라미터에 대한 값을 입력합니다.

ii. AWS Resource Groups을 사용하여 리소스를 대상으로 지정하기로 한 경우 리소스 그룹 목록에서 그룹의 이름을 선택합니다.

iii. 태그를 사용하여 리소스를 대상으로 지정하기로 한 경우 태그 키와 (선택 사항) 태그 값을 제공된 필드에 입력합니다. 추가를 선택합니다.

iv. 현재 AWS 계정 및 AWS 리전의 전체 인스턴스에서 Automation 실행서를 실행하려면 [전체 인스턴스(All instances)]를 선택합니다.

11. 입력 파라미터 섹션에서 필수 입력을 지정합니다. AutomationAssumeRole 목록에서 `AWS-SystemsManager-AutomationAdministrationRole` IAM 서비스 역할을 선택합니다.

> **Note**
>
> [입력 파라미터(Input parameters)] 섹션에서 일부 옵션은 선택할 필요가 없습니다. 그 이유는 태그 및 리소스 그룹을 사용하여 여러 리전 및 계정의 리소스를 대상으로 지정했기 때문입니다. 예를 들어 `AWS-RestartEC2Instance` 실행서를 선택한 경우 [입력 파라미터(Input parameters)] 섹션에서 인스턴스 ID를 지정하거나 선택할 필요가 없습니다. 자동화는 지정된 태그를 사용하여 다시 시작할 인스턴스를 정확히 찾아낼 수 있습니다.

12. (선택 사항) 모니터링을 위해 자동화에 적용할 CloudWatch 경보를 선택합니다. CloudWatch 경보를 자동화에 연결하려면 자동화를 시작하는 IAM 보안 주체에 `iam:createServiceLinkedRole` 작업에 대한 권한이 있어야 합니다. CloudWatch 경보에 대

한 자세한 내용은 Amazon CloudWatch 경보 사용을 참조하세요. 경보가 활성화되면 자동화가 취소되고 정의한 `onCancel` 단계가 실행됩니다. AWS CloudTrail을 사용하면 추적에 API 호출이 표시됩니다.

13. 각 계정-리소스 페어 내에서 Automation을 실행할 수 있는 AWS 리소스 수를 제한하려면 속도 제어 섹션의 옵션을 사용합니다.

    동시성 섹션에서 옵션을 선택합니다.

    - 대상을 선택하여 자동화 워크플로를 동시에 실행할 수 있는 대상 수(절대 개수)를 입력합니다.
    - 백분율을 선택하여 자동화 워크플로를 동시에 실행할 수 있는 대상의 백분율을 입력합니다.

14. 오류 임계값 섹션에서 옵션을 선택합니다.

    - 오류를 선택하여 자동화를 통한 다른 리소스로의 워크플로 전송이 중지될 때까지 허용되는 오류 수(절대 개수)를 입력합니다.
    - 백분율을 선택하여 자동화를 통한 다른 리소스로의 워크플로 전송이 중지될 때까지 허용되는 백분율을 입력합니다.

15. Execute(실행)를 선택합니다.

여러 리전 및 계정에서 Automation 실행(명령줄)

다음 절차에서는 AWS CLI(Linux 또는 Windows) 또는 AWS Tools for PowerShell을 사용하여 Automation 관리 계정을 통해 여러 리전 및 계정에서 자동화를 실행하는 방법을 설명합니다.

시작하기 전 준비 사항

다음 절차를 완료하기 전에 다음 정보에 유의하세요.

- 자동화를 실행할 AWS 계정 ID 또는 OU.
- 자동화를 실행할 위치인 Systems Manager에서 지원하는 리전.
- 자동화를 실행할 위치인, 태그 키 및 태그 값 또는 리소스 그룹 이름.

여러 리전 및 계정에서 자동화를 실행하려면

1. 아직 하지 않은 경우 AWS CLI 또는 AWS Tools for PowerShell을 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트 및 AWS Tools for PowerShell 설치를 참조하세요.

2. 다음 형식을 사용하여 여러 리전 및 계정에서 자동화를 실행하기 위한 명령을 생성합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

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" }
```

다음은 몇 가지 예제입니다.

예제 1: 이 예제에서는 123456789012 및 987654321098 계정의 EC2 인스턴스를 다시 시작합니다. 이들 인스턴스는 us-east-2 및 us-west-1 리전에 위치합니다. 인스턴스에 태그 키 페어 값 Env-PROD가 태그로 지정되어야 합니다.

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" }
```

예제 2: 이 예제에서는 123456789012 및 987654321098 계정의 EC2 인스턴스를 다시 시작합니다. 이들 인스턴스는 eu-central-1 리전에 위치합니다. 인스턴스가 prod-instances AWS 리소스 그룹의 멤버여야 합니다.

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" }
```

예제 3: 이 예제에서는 ou-1a2b3c-4d5e6c AWS 조직 단위(OU)의 EC2 인스턴스를 다시 시작합니다. 이들 인스턴스는 us-west-1 및 us-west-2 리전에 위치합니다. 인스턴스가 WebServices AWS 리소스 그룹의 멤버여야 합니다.

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" }
```

시스템은 다음과 유사한 정보를 반환합니다.

Linux & macOS

```
{
        "AutomationExecutionId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE"
    }
```

Windows

```
{
        "AutomationExecutionId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE"
    }
```

PowerShell

```
4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE
```

3. 다음 명령을 실행하여 자동화에 대한 세부 정보를 봅니다. *### ## ID*를 자신의 정보를 바꿉니다.

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. 다음 명령을 실행하여 자동화 진행 상황에 대한 세부 정보를 봅니다.

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**
>
> 콘솔에서 자동화 상태를 모니터링할 수도 있습니다. 자동화 실행 목록에서 방금 실행한 자동화를 선택한 후 실행 단계 탭을 선택합니다. 이 탭은 자동화 작업의 상태를 보여줍니다.

추가 정보

AWS Systems Manager 자동화를 사용하여 중앙 집중식 다중 계정 및 다중 리전 패치 적용

이벤트를 기반으로 자동화 실행

실행서를 Amazon EventBridge 이벤트의 대상으로 지정하여 자동화를 시작할 수 있습니다. 일정에 따라 또는 특정 AWS 시스템 이벤트가 발생할 때 자동화를 시작할 수 있습니다. 예를 들어 인스턴스가 시작될 때 인스턴스에서 소프트웨어를 설치하는 BootStrapInstances라는 실행서를 생성한다고 가정합니다. BootStrapInstances 실행서와 해당 자동화를 EventBridge 이벤트의 대상으로 지정하려면 먼저 새 EventBridge 규칙을 생성합니다. (예제 규칙: 서비스 이름: EC2, 이벤트 유형: EC2 인스턴스 상태 변경 알림, 특정 상태: 실행 중, 모든 인스턴스.) 그리고 다음 절차에 따라 EventBridge 콘솔 및 AWS Command Line Interface(AWS CLI)를 사용하여 BootStrapInstances 실행서를 이벤트 대상으로 지정합니다. 새 인스턴스가 시작되면 시스템에서 자동화를 실행하고 소프트웨어를 설치합니다.

실행서 생성에 대한 자세한 내용은 사용자 런북 생성 섹션을 참조하세요.

실행서를 사용하는 EventBridge 이벤트 생성(콘솔)

다음 절차에 따라 실행서를 EventBridge 이벤트 대상으로 구성합니다.

실행서를 EventBridge 이벤트 규칙의 대상으로 구성하려면

1. https://console.aws.amazon.com/events/에서 Amazon EventBridge 콘솔을 엽니다.
2. 탐색 창에서 규칙을 선택합니다.
3. 규칙 생성을 선택합니다.
4. 규칙에 대해 이름과 설명을 입력하십시오.

   규칙은 동일한 지역과 동일한 이벤트 버스의 다른 규칙과 동일한 이름을 가질 수 없습니다.
5. 이벤트 버스에서 이 규칙과 연결할 이벤트 버스를 선택합니다. 이 규칙이 자신의 AWS 계정에서 오는 일치하는 이벤트에 응답하도록 하려면 default(기본)를 선택합니다. 계정의 AWS 서비스이(가) 이벤트를 출력하면 항상 계정의 기본 이벤트 버스로 이동합니다.
6. 규칙이 트리거되는 방식을 선택합니다.

다음을 기반으로 규칙 생성	수행할 작업	
Event	a. 규칙 유형에서 이벤트 패턴이 있는 규칙을 선택합니다. b. 다음을 선택합니다. c. 이벤트 소스(Event source)에서 AWS 이	

다음을 기반으로 규칙 생성	수행할 작업	
	벤트 또는 EventBridge 파트너 이벤트(Events or EventBridge partner events)를 선택합니다. d. Event pattern(이벤트 패턴) 섹션에서 다음을 수행합니다. • 템플릿을 사용하여 이벤트 패턴을 생성하려면 Event pattern form(이벤트 패턴 양식)에서 Event source(이벤트 소스), AWS service(서비스) 및 Event type(이벤트 유형)을 선택합니다. 이벤트 유형으로 All Events(모든 이벤트)를 선택하면 이 AWS 서비스에서 출력한 모든 이벤트가 규칙과 일치합니다. 템플릿을 사용자 정의하려면 사용자 정의 패턴(JSON 편집기)(Custom pattern (JSON editor))을 선택하고 변경 사항을 작성합니다. • 사용자 정의 이벤트 패턴을 사용하려면 사용자 정의 패턴(JSON 편집기)(Custom pattern (JSON editor))을 선택하고 이벤트 패턴을 만들 수 있습니다.	

다음을 기반으로 규칙 생성	수행할 작업	
일정	a. 규칙 유형(Rule type)에서 스케줄(Schedule)을 선택합니다. b. 다음을 선택합니다. c. 스케줄 패턴(Schedule pattern)에서 다음을 수행합니다. • cron 표현식을 사용하여 스케줄을 정의하려면 오전 8시와 같이 특정 시간에 실행되는 세분화된 일정(A fine-grained schedule that runs at a specific time, such as 8:00 a.m)을 선택합니다. 매월 첫째 월요일(PST)(PST on the first Monday of every month)과 cron 표현식을 입력합니다. • rate 표현식을 사용하여 일정을 정의하려면 10분마다와 같이 일반 속도로 실행되는 일정(A schedule that runs at a regular rate, such as every 10 minutes)을 선택하고 rate 표현식을 입력합니다.	

7. 다음을 선택합니다.
8. 대상 유형에서 AWS서비스를 선택합니다.
9. Target(대상)에서 Systems Manager Automation을 선택합니다.
10. [문서(Document)]에서 대상이 호출될 때 사용할 실행서를 선택합니다.

11. Configure automation parameter(s)(자동화 파라미터 구성) 섹션에서 기본 파라미터 값을 유지하거나(사용 가능한 경우) 값을 직접 입력할 수 있습니다.

> **Note**
>
> 대상을 생성하려면 각 필수 파라미터에 대한 값을 지정해야 합니다. 그렇지 않으면 시스템에서 규칙이 생성되지만 실행되지 않습니다.

12. 여러 대상 유형에 대해 EventBridge에서는 대상에 이벤트를 보낼 권한이 필요합니다. 이 경우 EventBridge는 규칙 실행에 필요한 IAM 역할을 생성할 수 있습니다. 다음 중 하나를 수행하십시오.

    - IAM 역할을 자동으로 생성하려면 이 특정 리소스에 대해 새 역할 생성을 선택합니다.
    - 이전에 생성한 IAM 역할을 사용하려면 기존 역할 사용(Use existing role)을 선택하고 드롭다운에서 기존 역할을 선택합니다. EventBridge를 포함하도록 IAM 역할에 대한 신뢰 정책을 업데이트해야 할 수도 있습니다. 다음은 그 예제입니다.

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

13. 다음을 선택합니다.
14. (선택 사항) 규칙에 대해 하나 이상의 태그를 입력하십시오. 자세한 내용은 Amazon EventBridge User Guide의 Tagging Your Amazon EventBridge Resources를 참조하세요.
15. 다음을 선택합니다.
16. 규칙의 세부 정보를 검토하고 규칙 생성을 선택합니다.

실행서를 사용하는 EventBridge 이벤트 생성(명령줄)

다음 절차에서는 AWS CLI(Linux 또는 Windows) 또는 AWS Tools for PowerShell을 사용하여 EventBridge 이벤트 규칙을 생성하고 실행서를 대상으로 구성하는 방법을 설명합니다.

실행서를 EventBridge 이벤트 규칙의 대상으로 구성하려면

1. 아직 하지 않은 경우 AWS CLI 또는 AWS Tools for PowerShell을 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트 및 AWS Tools for PowerShell 설치를 참조하세요.
2. 새 EventBridge 이벤트 규칙을 지정하기 위한 명령을 생성합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   일정을 기반으로 트리거

   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"
   ```

   다음 예에서는 매일 오전 9:00(UTC)에 트리거하는 EventBridge 이벤트 규칙을 생성합니다.

   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 * * ? *)"
```

이벤트를 기반으로 트리거

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"]}'
```

다음 예에서는 리전의 EC2 인스턴스가 상태가 변경될 때 시작되는 EventBridge 이벤트 규칙을 생성합니다.

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"]}'
```

이 명령은 다음과 비슷한 새 EventBridge 규칙의 세부 정보를 반환합니다.

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. 실행서를 2단계에서 생성한 EventBridge 이벤트 규칙의 대상으로 지정하기 위한 명령을 생성합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

Linux & macOS

```
aws events put-targets \
--rule rule name \
--targets '{"Arn": "arn:aws:ssm:region:account ID:automation-definition/runbook
 name","Input":"{\"input parameter\":[\"value\"],\"AutomationAssumeRole\":
[\"arn:aws:iam::123456789012:role/AutomationServiceRole\"]}","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":"{\"input parameter\":[\"value\"],\"AutomationAssumeRole\":
[\"arn:aws:iam::123456789012:role/AutomationServiceRole\"]}","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
```

다음 예에서는 실행서 AWS-StartEC2Instance를 사용하여 지정된 인스턴스 ID를 시작하는 EventBridge 이벤트 대상을 생성합니다.

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
```

시스템은 다음과 같은 정보를 반환합니다.

Linux & macOS

```
{
"FailedEntries": [],
"FailedEntryCount": 0
}
```

Windows

```
{
"FailedEntries": [],
"FailedEntryCount": 0
}
```

PowerShell

PowerShell에 대해 명령이 성공해도 결과는 없습니다.

수동으로 자동화 실행

다음 절차에서는 수동 실행 모드를 사용하여 자동화를 실행하도록 AWS Systems Manager 콘솔과 AWS Command Line Interface(AWS CLI)를 사용하는 방법을 설명합니다. 수동 실행 모드를 사용하면 자동화가 [대기 중(Waiting)] 상태로 시작하고 각 단계 간에 [대기 중(Waiting)] 상태로 일시 중지합니다. 그러면 사용자가 자동화를 진행할 시점을 제어할 수 있어 다음 단계로 넘어가기 전에 단계의 결과를 검토해야 할 경우 유용합니다.

자동화는 현재 사용자의 컨텍스트에서 실행됩니다. 이는 런북과 그 런북에 의해 호출되는 모든 작업을 사용할 권한이 있는 한 추가 IAM 권한을 구성할 필요가 없음을 뜻합니다. IAM에서 관리자 권한이 있는 경우 이 자동화를 실행할 권한이 이미 있습니다.

단계별 자동화 실행(콘솔)

다음 절차에서는 Systems Manager 콘솔을 사용하여 단계별 자동화를 수동으로 실행하는 방법을 보여 줍니다.

단계별 자동화를 실행하려면

1. AWS Systems Manager 콘솔https://console.aws.amazon.com/systems-manager/을 엽니다.
2. 탐색 창에서 Automation(자동화)을 선택한 후 Execute automation(자동화 실행)을 선택합니다.

3. [Automation 문서(Automation document)] 목록에서 실행서를 선택합니다. 문서 카테고리 창에서 옵션을 1개 이상 선택하여 SSM 문서를 목적에 따라 필터링합니다. 자신이 소유한 실행서를 보려면 [내 소유(Owned by me)] 탭을 선택합니다. 자신의 계정과 공유하고 있는 실행서를 보려면 [나와 공유됨(Shared with me)] 탭을 선택합니다. 모든 실행서를 보려면 [모든 문서(All documents)] 탭을 선택합니다.

   > **Note**
   >
   > 실행서 이름을 선택하여 실행서에 대한 정보를 볼 수 있습니다.

4. 문서 세부 정보 섹션에서 문서 버전이 실행할 버전으로 설정되었는지 확인합니다. 이 시스템에는 다음 버전 옵션이 포함되어 있습니다.

   - 런타임 시 기본 버전 - Automation 런북이 정기적으로 업데이트되며 새 기본 버전이 할당된 경우 이 옵션을 선택합니다.
   - 런타임 시 최신 버전 - Automation 런북이 정기적으로 업데이트되며 최근에 업데이트된 버전을 실행하려는 경우 이 옵션을 선택합니다.
   - 1(기본값) - 문서의 최초 버전을 실행하려면 이 옵션을 선택합니다(기본값).

5. 다음을 선택합니다.
6. 실행 모드 섹션에서 Manual execution(수동 실행)을 선택합니다.
7. 입력 파라미터 섹션에서 필수 입력을 지정합니다. 필요에 따라 [AutomationAssumeRole] 목록에서 IAM 서비스 역할을 선택합니다.
8. 실행을 선택합니다.
9. 자동화의 첫 번째 단계를 실행할 준비가 되면 [이 단계를 실행(Execute this step)]을 선택합니다. 자동화가 1단계를 실행하고 이 절차의 3단계에서 선택한 런북에 지정된 이후 단계를 실행하기 전에 일시 중지됩니다. 실행서에 여러 단계가 포함된 경우 각 단계마다 [이 단계 실행(Execute this step)]을 선택하여 자동화를 진행합니다. [이 단계 실행(Execute this step)]을 선택할 때마다 작업이 실행됩니다.

   > **Note**
   >
   > 콘솔에 자동화 상태가 표시됩니다. 자동화 단계가 실행되지 않는 경우 Systems Manager Automation 문제 해결 섹션을 참조하세요.

10. 실행서에 지정된 모든 단계를 마쳤으면 [완료 및 결과 보기(Complete and view results)]를 선택하여 자동화를 완료하고 결과를 확인합니다.

단계별 자동화 실행(명령줄)

다음 절차에서는 AWS CLI(Linux, macOS 또는 Windows에서) 또는 AWS Tools for PowerShell을 사용하여 단계별 자동화를 수동으로 실행하는 방법을 설명합니다.

단계별 자동화를 실행하려면

1. 아직 하지 않은 경우 AWS CLI 또는 AWS Tools for PowerShell을 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트 및 AWS Tools for PowerShell 설치를 참조하세요.

2. 수동 자동화를 시작하려면 다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   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
   ```

   여기에 실행서 `AWS-RestartEC2Instance`를 사용하여 지정된 EC2 인스턴스를 다시 시작하는 예가 있습니다.

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"}
```

시스템은 다음과 같은 정보를 반환합니다.

Linux & macOS

```
{
    "AutomationExecutionId": "ba9cd881-1b36-4d31-a698-0123456789ab"
}
```

Windows

```
{
    "AutomationExecutionId": "ba9cd881-1b36-4d31-a698-0123456789ab"
}
```

PowerShell

```
ba9cd881-1b36-4d31-a698-0123456789ab
```

3. 자동화의 첫 번째 단계를 실행할 준비가 되면 다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다. 자동화가 1단계를 실행하고 이 절차의 1단계에서 선택한 런북에 지정된 이후 단계를 실행하기 전에 일시 중지됩니다. 실행서에 여러 단계가 포함된 경우 각 단계마다 다음 명령을 실행하여 자동화를 진행합니다.

   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"}
   ```

   명령이 성공해도 결과는 없습니다.

4. 다음 명령을 실행하여 자동화의 각 단계 실행 상태를 검색합니다.

   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
```

시스템은 다음과 같은 정보를 반환합니다.

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. 선택한 런북 내에 지정된 모든 단계가 종료된 후에 다음 명령을 실행하여 자동화를 완료합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

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
```

명령이 성공해도 결과는 없습니다.

자동화 예약

다음 주제에는 특정 간격 또는 지정한 특정 시간에 실행되도록 자동화를 예약하는 방법에 대한 정보가 포함되어 있습니다.

내용

- State Manager 연결을 사용하여 자동화 예약
- 유지 관리 기간으로 자동화 예약

State Manager 연결을 사용하여 자동화 예약

실행서와 State Manager 연결을 생성하여 자동화를 시작할 수 있습니다. State Manager는 AWS Systems Manager의 기능입니다. 실행서와의 State Manager 연결을 생성하면 여러 유형의 AWS 리소스를 대상으로 지정할 수 있습니다. 예를 들어 다음과 같이 AWS 리소스에 원하는 상태를 적용하는 연결을 생성할 수 있습니다.

- Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에 연결하여 Systems Manager 역할을 관리형 인스턴스로 만듭니다.
- 보안 그룹에 원하는 인바운드 및 아웃바운드 규칙을 적용합니다.
- Amazon DynamoDB 백업을 생성하거나 삭제합니다.
- Amazon Elastic Block Store(Amazon EBS) 스냅샷을 생성하거나 삭제합니다.
- Amazon Simple Storage Service(Amazon S3) 버킷에 대한 읽기 및 쓰기 권한을 해제합니다.
- 관리형 인스턴스 및 Amazon Relational Database Service(Amazon RDS) 인스턴스를 시작, 재시작 또는 중지합니다.
- Linux, macOS 및 Window AMIs를 패치합니다.

AWS Systems Manager 콘솔과 AWS Command Line Interface(AWS CLI)에서 다음 절차를 사용하여 자동화를 실행하는 State Manager 연결을 생성합니다.

시작하기 전 준비 사항

State Manager를 사용하여 자동화를 실행하기 전에 다음과 같은 중요 세부 정보를 알고 있어야 합니다.

- 실행서를 사용하는 연결을 생성하기 전에 AWS Systems Manager의 기능인 Automation에 대해 구성된 권한이 있는지 확인합니다. 자세한 내용은 Automation 설정 단원을 참조하십시오.
- 실행서를 사용하는 State Manager 연결은 AWS 계정에서 동시에 실행되는 최대 자동화 수에 영향을 미칩니다. 최대 100개의 자동화를 동시에 실행할 수 있습니다. 자세한 내용은 Amazon Web Services 일반 참조의 Systems Manager 서비스 할당량을 참조하세요.
- 자동화를 실행하는 경우 State Manager는 AWS CloudTrail에서 자동화에 의해 시작된 API 작업을 기록하지 않습니다.
- Systems Manager는 자동으로 서비스 연결 역할을 생성하여 State Manager에 Systems Manager Automation API 작업을 호출할 권한을 부여합니다. 원할 경우, AWS CLI 또는 AWS Tools for PowerShell에서 다음 명령을 실행하여 서비스 연결 역할을 직접 생성할 수 있습니다.

  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
```

서비스 연결 역할에 대한 자세한 내용은 Systems Manager에 서비스 연결 역할 사용 섹션을 참조하세요.

자동화를 실행하는 연결 생성(콘솔)

다음 절차에서는 Systems Manager 콘솔을 사용하여 자동화를 실행하는 State Manager 연결을 생성하는 방법을 설명합니다.

자동화를 실행하는 State Manager 연결을 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 State Manager를 선택한 후 연결 생성을 선택합니다.
3. 이름(Name) 필드에 이름을 지정합니다. 이는 선택 사항이며, 권장 사항은 아닙니다.
4. [문서(Document)] 목록에서 실행서를 선택합니다. 검색 창에서 [문서 유형 : 같음 : Automation(Document type : Equal : Automation)]를 필터링합니다. 추가 실행서를 보려면 검색 창의 오른쪽에 있는 번호를 사용합니다.

   > **Note**
   >
   > 실행서 이름을 선택하여 실행서에 대한 정보를 볼 수 있습니다.

5. 대상의 리소스 ID를 지정하여 하나 이상의 대상에 대해 자동화를 실행하려면 Simple execution(단순 실행)을 선택합니다. 태그 또는 AWS Resource Groups와 같은 대상 지정 옵션을 지정하여 AWS 리소스 플릿에서 자동화를 실행하려면 [속도 제어(Rate control)]를 선택합니다. 또한 동시성 및 오류 임계값을 지정하여 리소스에 대한 자동화 작업을 제어할 수도 있습니다.

   [속도 제어(Rate control)]를 선택하면 [대상(Targets)] 섹션이 표시됩니다.
6. 대상 섹션에서 리소스를 대상으로 지정할 방법을 선택합니다.

a. (필수) 파라미터 목록에서 파라미터 하나를 선택합니다. [파라미터(Parameter)] 목록의 항목은 이 절차의 시작 부분에서 선택한 실행서의 파라미터로 결정됩니다. 파라미터를 선택하면 자동화가 실행되는 리소스 유형을 정의할 수 있습니다.

b. (필수) 대상 목록에서 리소스를 대상으로 지정할 방법을 선택합니다.

- 리소스 그룹: Resource Group(리소스 그룹) 목록에서 그룹의 이름을 선택합니다. 실행서에서 AWS Resource Groups를 대상으로 지정에 대한 자세한 내용은 AWS Resource Groups을 대상으로 지정 섹션을 참조하세요.
- 태그: 제공된 필드에 태그 키 및 (선택적으로) 태그 값을 입력합니다. 추가를 선택합니다. 실행서에서 태그를 대상으로 지정에 대한 자세한 내용은 태그를 대상으로 지정 섹션을 참조하세요.
- 파라미터 값: 입력 파라미터 섹션에 값을 입력합니다. 여러 값을 지정할 경우 Systems Manager는 지정된 각 값에 대해 하위 자동화를 실행합니다.

  예를 들어 실행서에 InstanceID 파라미터가 포함되어 있다고 가정해 보겠습니다. 자동화 실행 시 InstanceID 파라미터의 값을 대상으로 지정할 경우 Systems Manager는 지정한 각 인스턴스 ID 값마다 하위 자동화를 하나 실행합니다. 이 자동화가 각각의 지정된 인스턴스 실행을 완료하거나 실행에 실패하면 상위 자동화가 완료된 것입니다. 최대 50개 파라미터 값을 대상으로 지정할 수 있습니다. 실행서에서 파라미터 값을 대상으로 지정에 대한 자세한 내용은 파라미터 값을 대상으로 지정 섹션을 참조하세요.

7. 입력 파라미터 섹션에서 필요한 입력 파라미터를 지정합니다.

   태그나 리소스 그룹을 사용하여 리소스를 대상으로 지정하기로 한 경우 [입력 파라미터(Input parameters)] 섹션에서 옵션 중 일부는 선택하지 않아도 될 것입니다. 예를 들어 `AWS-RestartEC2Instance` 실행서를 선택했으며 태그를 사용하여 인스턴스를 대상으로 지정하기로 한 경우 [입력 파라미터(Input parameters)] 섹션에서 인스턴스 ID를 지정하거나 선택할 필요가 없습니다. 자동화는 지정된 태그를 사용하여 다시 시작할 인스턴스를 정확히 찾아낼 수 있습니다.

   > **Important**
   >
   > AutomationAssumeRole 필드에 역할 ARN을 지정해야 합니다. State Manager은(는) 수임 역할을 사용하여 사용자 대신 실행서에 지정된 AWS 서비스을(를) 호출하고 Automation 연결을 실행합니다.

8. 연결을 주기적으로 실행하려면 일정 지정 섹션에서 On Schedule(일정에 따라)을 선택합니다. 이 옵션을 선택할 경우 제공된 옵션을 사용하여 Cron 또는 Rate 표현식으로 일정을 생성합니다.

State Manager용 Cron 및 Rate 표현식에 대한 자세한 내용은 연결에 대한 Cron 및 Rate 표현식 섹션을 참조하세요.

> **Note**
>
> Rate 표현식은 실행서를 사용하는 State Manager 연결에 대해 선호되는 예약 메커니즘입니다. Rate 표현식을 사용하면 동시에 실행되는 자동화가 최대 수에 도달하는 경우 연결 실행 유연성이 더 큽니다. 속도 일정에서는 Systems Manager가 동시 자동화가 최대 수에 도달하여 조절되었다는 알림을 수신한 잠시 후에 자동화를 재시도할 수 있습니다.

연결을 한 번만 실행하려면 No schedule(일정 없음)을 선택합니다.

9. (선택 사항) 속도 제어(Rate Control) 섹션에서 동시성(Concurrency) 및 오류 임계값(Error threshold) 옵션을 선택하여 AWS 리소스에 대한 자동화 배포를 제어합니다.

   a. 동시성 섹션에서 옵션을 선택합니다.

      - [대상(targets)]을 선택하여 자동화를 동시에 실행할 수 있는 대상 수(절대 개수)를 입력합니다.
      - [백분율(percentage)]을 선택하여 자동화를 동시에 실행할 수 있는 대상의 백분율을 입력합니다.

   b. Error threshold(오류 임계값) 섹션에서 옵션을 선택합니다.

      - [오류(errors)]를 선택하여 Automation을 통한 다른 리소스로의 자동화 전송이 중지될 때까지 허용되는 오류 수(절대 개수)를 입력합니다.
      - [백분율(percentage)]을 선택하여 Automation을 통한 다른 리소스로의 자동화 전송이 중지될 때까지 허용되는 백분율을 입력합니다.

   자동화에서 대상 및 속도 제어 사용에 대한 자세한 내용은 대규모로 자동화 실행 섹션을 참조하세요.

10. 연결 생성을 선택합니다.

> **Important**
>
> 연결을 생성하면 즉시 지정된 대상에 대해 연결이 실행됩니다. 그런 다음 cron 또는 rate 표현식을 기반으로 연결이 실행됩니다. [일정 없음(No schedule)]을 선택한 경우 연결이 다시 실행되지 않습니다.

자동화를 실행하는 연결 생성(명령줄)

다음 절차에서는 AWS CLI(Linux 또는 Windows) 또는 AWS Tools for PowerShell을 사용하여 자동화를 실행하는 State Manager 연결을 생성하는 방법을 설명합니다.

시작하기 전 준비 사항

다음 절차를 완료하기 전에 먼저 Runbook을 실행하는 데 필요한 권한이 포함된 IAM 서비스 역할을 생성하고 AWS Systems Manager의 기능인 Automation에 대해 신뢰 관계를 구성했는지 확인하십시오. 자세한 내용은 작업 1: Automation을 위한 서비스 역할 생성 단원을 참조하십시오.

자동화를 실행하는 연결을 생성하려면

1. 아직 하지 않은 경우 AWS CLI 또는 AWS Tools for PowerShell을 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트 및 AWS Tools for PowerShell 설치를 참조하세요.
2. 다음 명령을 실행하여 문서 목록을 확인합니다.

   Linux & macOS

   ```
   aws ssm list-documents
   ```

   Windows

   ```
   aws ssm list-documents
   ```

   PowerShell

   ```
   Get-SSMDocumentList
   ```

연결에 사용할 실행서의 이름을 기록해 둡니다.

3. 다음 명령을 실행하여 런북에 대한 세부 정보를 봅니다. 다음 명령에서 *### ##*을 자신의 정보로 바꿉니다.

   Linux & macOS

   ```
   aws ssm describe-document \
   --name runbook name
   ```

   `--automation-target-parameter-name` 옵션에 사용할 파라미터 이름(예: `InstanceId`)에 유의하십시오. 이 파라미터는 자동화가 실행되는 리소스 유형을 결정합니다.

   Windows

   ```
   aws ssm describe-document ^
   --name runbook name
   ```

   `--automation-target-parameter-name` 옵션에 사용할 파라미터 이름(예: `InstanceId`)에 유의하십시오. 이 파라미터는 자동화가 실행되는 리소스 유형을 결정합니다.

   PowerShell

   ```
   Get-SSMDocumentDescription `
   -Name runbook name
   ```

   `AutomationTargetParameterName` 옵션에 사용할 파라미터 이름(예: `InstanceId`)에 유의하십시오. 이 파라미터는 자동화가 실행되는 리소스 유형을 결정합니다.

4. State Manager 연결을 사용하여 자동화를 실행하는 명령 생성 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   태그를 사용하여 대상 지정

   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**
>
> AWS CLI를 사용하여 연결을 생성한 경우 `--targets` 파라미터를 사용하여 연결에 대한 인스턴스를 대상으로 지정합니다. `--instance-id` 파라미터를 사용하지 마십시오. `--instance-id` 파라미터는 레거시 파라미터입니다.

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**
>
> AWS CLI를 사용하여 연결을 생성한 경우 `--targets` 파라미터를 사용하여 연결에 대한 인스턴스를 대상으로 지정합니다. `--instance-id` 파라미터를 사용하지 마십시오. `--instance-id` 파라미터는 레거시 파라미터입니다.

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**
>
> AWS Tools for PowerShell를 사용하여 연결을 생성한 경우 `Target` 파라미터를 사용하여 연결에 대한 인스턴스를 대상으로 지정합니다. `InstanceId` 파라미터를 사용하지 마십시오. `InstanceId` 파라미터는 레거시 파라미터입니다.

파라미터 값을 사용하여 대상 지정

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"
```

AWS Resource Groups을 사용하여 대상 지정

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"
```

여러 계정 및 리전을 대상으로 지정

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"]
```

이 명령은 다음과 비슷한 새 연결의 세부 정보를 반환합니다.

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**
>
> 태그를 사용하여 하나 이상의 대상 인스턴스에 대해 연결을 생성한 다음 인스턴스에서 태그를 제거하면 해당 인스턴스가 더 이상 연결을 실행하지 않습니다. 이러한 인스턴스는 State Manager 문서에서 연결 해제됩니다.

State Manager 연결에 의해 실행되는 자동화의 문제 해결

Systems Manager Automation은 리전별로 계정당 동시 자동화 100개 및 대기 중인 자동화 1,000개 제한을 시행합니다. 실행서를 사용하는 State Manager 연결에서 실패(Failed) 상태와 AutomationExecutionLimitExceeded 세부 상태를 표시할 경우 자동화가 한도에 도달한 것입니다. 결과적으로 자동화가 제한됩니다. 이 문제를 해결하려면 다음과 같이 실행합니다.

- 연결에 다른 rate 또는 cron 표현식을 사용합니다. 예를 들어 연결이 30분마다 실행하도록 지정된 경우 1시간 또는 2시간마다 실행하도록 표현식을 변경합니다.
- 보류 중(Pending) 상태인 기존 자동화를 삭제합니다. 이러한 자동화를 삭제하면 현재 대기열을 삭제하는 것입니다.

유지 관리 기간으로 자동화 예약

실행서를 유지 관리 기간의 등록된 태스크로 구성하여 자동화를 시작할 수 있습니다. 실행서를 등록된 태스크로 등록하면 유지 관리 기간은 예약된 유지 관리 기간 중에 자동화를 실행합니다.

예를 들어 유지 관리 기간에 대상으로 등록된 인스턴스의 Amazon Machine Image(AMI)를 만드는 `CreateAMI`라는 실행서를 생성한다고 가정해 보겠습니다. `CreateAMI` 실행서와 해당 자동화를 유지 관리 기간의 등록된 태스크로 지정하려면 먼저 유지 관리 기간을 생성하고 대상을 등록합니다. 그런 다

음, 다음 절차에 따라 유지 관리 기간에서 `CreateAMI` 문서를 등록된 태스크로 지정합니다. 예약된 기간 중에 유지 관리 기간이 시작되면 시스템은 자동화를 실행하여 등록된 대상의 AMI를 생성합니다.

Automation 실행서 생성에 대한 자세한 내용은 사용자 런북 생성 섹션을 참조하세요. Automation은 AWS Systems Manager의 기능입니다.

AWS Systems Manager 콘솔, AWS Command Line Interface(AWS CLI) 또는 AWS Tools for Windows PowerShell에서 다음 절차에 따라 자동화를 유지 관리 기간의 등록된 태스크로 구성합니다.

자동화 태스크를 유지 관리 기간으로 등록(콘솔)

다음 절차에서는 Systems Manager 콘솔을 사용하여 자동화를 유지 관리 기간의 등록된 태스크로 구성하는 방법을 설명합니다.

시작하기 전 준비 사항

다음 절차를 완료하기 전에 유지 관리 기간을 생성하고 하나 이상의 대상을 등록해야 합니다. 자세한 내용은 다음 절차를 참조하세요.

- 유지 관리 기간 생성(콘솔).
- 유지 관리 기간에 대상 할당(콘솔)

자동화를 유지 관리 기간의 등록된 태스크로 구성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 왼쪽 탐색 창에서 Maintenance Windows를 선택하고 자동화 작업을 등록하려는 유지 관리 기간을 선택합니다.
3. 작업을 선택합니다. 그런 다음 [자동화 태스크 등록(Register Automation task)]을 선택하여 실행서를 사용해 대상에서 원하는 자동화를 실행합니다.
4. 이름에 작업 이름을 입력합니다.
5. 설명에 설명을 입력합니다.
6. [문서(Document)]에서 실행할 태스크를 정의하는 실행서를 선택합니다.
7. [문서 버전(Document version)]에서 사용할 실행서 버전을 선택합니다.
8. [태스크 우선순위(Task priority)]에서 이 태스크의 우선순위를 지정합니다. 1이 가장 높은 우선순위입니다. 유지 관리 기간의 작업은 우선순위에 따라 예약되며, 우선순위가 같은 작업은 동시에 예약됩니다.
9. [대상(Targets)] 섹션에서 선택한 실행서가 리소스에 대한 태스크를 실행하는 경우 태그를 지정하거나 수동으로 인스턴스를 선택하여 이 자동화를 실행할 대상을 식별합니다.

> **Note**
>
> 대상 대신 입력 파라미터를 통해 리소스를 전달하려는 경우 유지 관리 기간 대상을 지정할 필요가 없습니다.
> 대부분의 경우 자동화 태스크의 대상을 명시적으로 지정할 필요가 없습니다. 예를 들어 `AWS-UpdateLinuxAmi` 실행서를 사용하여 Linux용 Amazon Machine Image(AMI)를 업데이트하는 Automation 유형 태스크를 생성한다고 가정해 보겠습니다. 태스크가 실행되면 AMI는 사용 가능한 최신 Linux 배포 패키지와 Amazon 소프트웨어로 업데이트됩니다. AMI에서 생성된 새 인스턴스에는 이러한 업데이트가 이미 설치되어 있습니다. 업데이트할 AMI의 ID가 실행서에 대한 입력 파라미터에 지정되어 유지 관리 기간 태스크에서 대상을 다시 지정할 필요가 없습니다.

대상이 필요하지 않은 유지 관리 기간 태스크에 대한 자세한 내용은 the section called “대상 없이 유지 관리 기간 태스크 등록” 섹션을 참조하세요.

10. (선택 사항) 속도 제어:

> **Note**
>
> 실 행중인 태스크가 대상을 지정하지 않으면 속도 제어를 지정할 필요가 없습니다.

- 동시성(Concurrency)에 자동화를 동시에 실행할 대상의 수 또는 백분율을 지정합니다.

  태크 키-값 페어를 선택하여 대상을 선택했지만 몇 개의 대상이 선택된 태그를 사용할지 확실치 않다면 백분율을 지정하여 동시에 실행할 수 있는 자동화 수를 제한합니다.

  유지 관리 기간이 실행되면 대상별로 새로운 자동화가 시작됩니다. AWS 계정당 동시 자동화는 100개로 제한됩니다. 100보다 높은 동시성을 지정할 경우 100개를 초과하는 동시 자동화는 자동화 대기열에 자동으로 추가됩니다. 자세한 내용은 Amazon Web Services 일반 참조의 Systems Manager 서비스 할당량을 참조하세요.

- [오류 임계값(Error threshold)]에서 대상 수 또는 백분율 때문에 실패한 후 언제 다른 대상에서 자동화 실행을 중지할지 지정합니다. 예를 들어 세 번의 오류를 지정할 경우 Systems Manager는 네 번째 오류가 수신하면 자동화 실행을 중지합니다. 여전히 자동화를 처리하는 대상도 오류를 보낼 수 있습니다.

11. [입력 파라미터(Input Parameters)] 섹션에서 실행서의 파라미터를 지정합니다. 실행서의 경우 시스템에서 일부 값을 자동으로 채웁니다. 이러한 값을 유지하거나 바꿀 수 있습니다.

    > **Important**
    >
    > 실행서의 경우 선택적으로 Automation 수임 역할을 지정할 수 있습니다. 이 파라미터에 역할을 지정하지 않으면 자동화가 11단계에서 선택한 유지 관리 기간 서비스 역할을 수임합니다. 그러므로 선택한 유지 관리 기간 서비스 역할이 실행서 내에 정의된 작업을 수행할 수 있는 적절한 AWS Identity and Access Management(IAM) 권한을 가지고 있는지 확인해야 합니다.
    >
    > 예를 들어 Systems Manager용 서비스 연결 역할은 실행서 `AWS-CopySnapshot`을 실행하는 데 필요한 IAM 권한 `ec2:CreateSnapshot`이 없습니다. 이 시나리오에서는 사용자 지정 유지 관리 기간 서비스 역할을 사용하거나 `ec2:CreateSnapshot` 권한이 있는 자동화 수임 역할을 지정해야 합니다. 자세한 설명은 Automation 설정을 참조하세요.

12. IAM 서비스 역할(IAM service role) 영역에서 역할을 선택하여 Systems Manager에 자동화를 시작할 수 있는 권한을 부여합니다.

    유지 관리 기간 작업을 위한 사용자 지정 서비스 역할을 생성하려면 콘솔을 사용하여 유지 관리 기간에 대한 권한 구성의 내용을 참조하세요.

13. 자동화 작업 등록을 선택합니다

Automation 태스크를 유지 관리 기간으로 등록(명령줄)

다음 절차에서는 AWS CLI(Linux 또는 Windows) 또는 AWS Tools for PowerShell을 사용하여 자동화를 유지 관리 기간의 등록된 태스크로 구성하는 방법을 설명합니다.

시작하기 전 준비 사항

다음 절차를 완료하기 전에 유지 관리 기간을 생성하고 하나 이상의 대상을 등록해야 합니다. 자세한 내용은 다음 절차를 참조하세요.

- 1단계: 유지 관리 기간 생성(AWS CLI).
- 2단계: 유지 관리 기간에 대상 노드 등록(AWS CLI)

자동화를 유지 관리 기간의 등록된 태스크로 구성하려면

1. 아직 하지 않은 경우 AWS CLI 또는 AWS Tools for PowerShell을 설치하고 구성합니다.

자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트 및 AWS Tools for PowerShell 설치를 참조하세요.

2. 자동화를 유지 관리 기간의 등록된 태스크로 구성하기 위한 명령을 생성합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

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**
>
> AWS CLI를 사용하여 등록된 태스크로 자동화를 구성하려면 `--Task-Invocation-Parameters` 파라미터를 사용하여 태스크 실행 시 태스크에 전달할 파라미터를 지정합니다. `--Task-Parameters` 파라미터를 사용하지 마십시오. `--Task-Parameters` 파라미터는 레거시 파라미터입니다.
> 대상이 지정되지 않은 유지 관리 기간 태스크의 경우 `--max-errors` 및 `--max-concurrency` 값을 제공할 수 없습니다. 그 대신에 시스템에서 describe-maintenance-window-tasks 및 get-maintenance-window-task와 같은 명령에 대한 응답으로 보고될 수 있는 자리 표시자 값 1을 삽입합니다. 이러한 값은 태스크 실행에 영향을 주지 않으며 무시할 수 있습니다.
> 대상이 필요하지 않은 유지 관리 기간 태스크에 대한 자세한 내용은 대상 없이 유지 관리 기간 태스크 등록 섹션을 참조하세요.

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**
>
> AWS CLI를 사용하여 등록된 태스크로 자동화를 구성하려면 `--task-invocation-parameters` 파라미터를 사용하여 태스크 실행 시 태스크에 전달할 파라미터를 지정합니다. `--task-parameters` 파라미터를 사용하지 마십시오. `--task-parameters` 파라미터는 레거시 파라미터입니다.
>
> 대상이 지정되지 않은 유지 관리 기간 태스크의 경우 `--max-errors` 및 `--max-concurrency` 값을 제공할 수 없습니다. 그 대신에 시스템에서 describe-maintenance-window-tasks 및 get-maintenance-window-task와 같은 명령에 대한 응답으로 보고될 수 있는 자리 표시자 값 1을 삽입합니다. 이러한 값은 태스크 실행에 영향을 주지 않으며 무시할 수 있습니다.
>
> 대상이 필요하지 않은 유지 관리 기간 태스크에 대한 자세한 내용은 대상 없이 유지 관리 기간 태스크 등록 섹션을 참조하세요.

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**
>
> AWS Tools for PowerShell를 사용하여 등록된 태스크로 자동화를 구성하려면 `-Automation_Parameter` 파라미터를 사용하여 태스크 실행 시 태스크에 전달할 파라미터를 지정합니다. `-TaskParameters` 파라미터를 사용하지 마십시오. `-TaskParameters` 파라미터는 레거시 파라미터입니다.
> 대상이 지정되지 않은 유지 관리 기간 태스크의 경우 `-MaxError` 및 `-MaxConcurrency` 값을 제공할 수 없습니다. 대신 시스템은 `Get-SSMMaintenanceWindowTaskList` 및 `Get-SSMMaintenanceWindowTask`와 같은 명령에 대한 응답으로 보고될 수 있는 자리 표시자 값 1을 삽입합니다. 이러한 값은 태스크 실행에 영향을 주지 않으며 무시할 수 있습니다.
> 대상이 필요하지 않은 유지 관리 기간 태스크에 대한 자세한 내용은 대상 없이 유지 관리 기간 태스크 등록 섹션을 참조하세요.

다음 예에서는 자동화를 우선순위 1의 유지 관리 기간의 등록된 태스크로 구성합니다. 또한 대상 없는 유지 관리 기간 태스크에 대해 `--targets`, `--max-errors` 및 `--max-concurrency` 옵션을 생략하는 방법도 보여줍니다. 자동화는 `AWS-StartEC2Instance` 실행서 및 지정된 Automation 수임 역할을 사용하여 유지 관리 기간의 대상으로 등록된 EC2 인스턴스를 시작합니다. 유지 관리 기간은 지정된 시간에 최대 5번의 인스턴스에서 동시에 자동화를 실행합니다. 또한 등록된 태스크는 오류 수가 1을 초과하는 경우 특정 간격에 대해 더 많은 인스턴스에서 실행을 중지합니다.

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
```

이 명령은 다음과 비슷한 새로운 등록된 작업의 세부 정보를 반환합니다.

Linux & macOS

```
{
"WindowTaskId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE"
}
```

Windows

```
{
"WindowTaskId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE"
}
```

PowerShell

```
4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE
```

3. 등록된 작업을 보려면 다음 명령을 실행합니다. *## ## ## ID*를 자신의 정보로 바꿉니다.

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
```

시스템은 다음과 같은 정보를 반환합니다.

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 작업 참조

이 참조는 Automation 실행서에서 지정할 수 있는 Automation 작업을 설명합니다. Automation은 AWS Systems Manager의 기능입니다. 이러한 작업은 다른 유형의 Systems Manager(SSM) 문서에는 사용할 수 없습니다. 다른 유형의 SSM 문서용 플러그인에 대한 자세한 내용은 Command 문서 플러그인 참조 섹션을 참조하세요.

Systems Manager Automation은 Automation 실행서에 정의된 단계를 실행합니다. 각 단계는 특정 작업과 관련되어 있습니다. 작업은 해당 단계의 입력, 동작 및 출력을 결정합니다. 단계는 실행서의 `mainSteps` 섹션에 정의됩니다.

작업 또는 단계의 출력은 지정할 필요가 없습니다. 출력은 단계와 관련된 작업에 의해 미리 결정되어 있습니다. 실행서에 단계 입력을 지정할 때 이전 단계로부터 하나 이상의 출력을 참조할 수 있습니다. 예를 들어 후속 `aws:runCommand` 작업에 `aws:runInstances` 출력을 사용 가능하게 할 수 있습니다. 또한 실행서의 `Output` 섹션에서 이전 단계의 출력을 참조할 수도 있습니다.

> **Important**
>
> AWS Identity and Access Management(IAM) 서비스 역할을 사용하여 다른 서비스를 호출하는 자동화 워크플로를 실행하는 경우 해당 서비스 역할이 다른 서비스를 호출할 권한이 있도록 구성되어야 합니다. 이 요구 사항은 `AWS-ConfigureS3BucketLogging`, `AWS-CreateDynamoDBBackup`, `AWS-RestartEC2Instance` 실행서 등의 모든 AWS Automation 실행서(`AWS-*` 실행서)에 적용됩니다. 또한 다른 서비스를 호출하는 작업을 사용하여 다른 AWS 서비스을(를) 호출하는 사용자 정의 Automation 실행서를 생성하는 경우에도 이 요구 사항이 항상 적용됩니다. 예를 들어 `aws:executeAwsApi`, `aws:createStack` 또는 `aws:copyImage` 작업을 사용하는 경우 이러한 서비스를 호출할 수 있는 권한을 포함하여 서비스 역할을 구성합니다. 역할에 IAM 인라인 정책을 추가하여 다른 AWS 서비스에 대한 권한을 부여할 수 있습니다. 자세한 내용은 (선택 사항) 다른 AWS 서비스를 간접적으로 호출할 Automation 인라인 정책과 고객 관리형 정책 추가 단원을 참조하십시오.

주제

- aws:copyImage - Amazon Machine Image 복사 또는 암호화
- aws:createImage - Amazon Machine Image 생성
- aws:createStack - AWS CloudFormation 스택 생성
- aws:createTags - AWS 리소스에 대한 태그 생성
- aws:deleteImage - Amazon Machine Image 삭제
- aws:deleteStack - AWS CloudFormation 스택 삭제
- aws:executeAutomation - 또 다른 자동화 실행
- aws:executeAwsApi - AWS API 작업 호출 및 실행
- aws:executeScript - 스크립트 실행
- aws:executeStateMachine - AWS Step Functions 상태 시스템을 실행합니다.
- aws:invokeWebhook - Automation 웹후크 통합 호출
- aws:invokeLambdaFunction - AWS Lambda 함수 호출
- aws:loop- 자동화의 여러 단계를 반복
- aws:pause - 자동화 일시 중지
- aws:runCommand - 관리형 인스턴스에서 명령 실행
- aws:runInstances - Amazon EC2 인스턴스 시작
- aws:sleep - 자동화 지연
- aws:updateVariable - 런북 변수 값을 업데이트
- aws:waitForAwsResourceProperty - AWS 리소스 속성 대기
- Automation 시스템 변수

모든 작업에서 공유하는 속성

공통 속성은 모든 작업에서 발견되는 파라미터 또는 옵션입니다. 일부 옵션은 단계가 완료될 때까지 기다리는 시간, 단계가 실패할 경우 수행할 작업 등과 같은 단계에 대한 동작을 정의합니다. 다음은 모든 작업에 공통적인 속성입니다.

description

런북 또는 단계의 용도를 설명하려고 제공하는 정보입니다.

타입: 문자열

필수 항목 여부: 아니요

name

실행서에 있는 모든 단계 이름을 통틀어 고유해야 하는 식별자.

타입: 문자열

허용 패턴: [a-zA-Z0-9_]+$

필수 항목 여부: 예

action

이 단계에서 실행해야 할 작업의 이름입니다. aws:runCommand - 관리형 인스턴스에서 명령 실행는 여기에서 지정할 수 있는 작업의 예입니다. 이 문서는 사용 가능한 모든 작업에 대한 세부 정보를 제공합니다.

타입: 문자열

필수 항목 여부: 예

maxAttempts

단계가 실패할 경우 재시도해야 하는 횟수. 지정된 값이 1보다 크면 모든 재시도가 실패할 때까지 해당 단계는 실패한 것으로 간주되지 않습니다. 기본값은 1입니다.

유형: 정수

필수 항목 여부: 아니요

timeoutSeconds

단계에 대한 시간 제한 값. 시간 제한에 도달했지만 maxAttempts 값이 1보다 큰 경우 단계는 모든 재시도 횟수가 시도될 때까지 시간 초과된 것으로 간주되지 않습니다.

유형: 정수

필수 항목 여부: 아니요

onFailure

실패 시 자동화가 중지되어야 하는지, 계속되어야 하는지 또는 다른 단계로 이동해야 하는지를 나타냅니다. 이 옵션의 기본값은 중단입니다.

타입: 문자열

유효 값: 중단 | 계속 | 단계:*step_name*

필수 항목 여부: 아니요

onCancel

사용자가 자동화를 취소하는 경우 자동화가 이동해야 하는 단계를 나타냅니다. Automation은 최대 2분 동안 취소 워크플로를 실행합니다.

타입: 문자열

유효 값: 중단 | 단계:*step_name*

필수 항목 여부: 아니요

onCancel 속성은 다음 작업으로 이동을 지원하지 않습니다.

- aws:approve
- aws:copyImage
- aws:createImage
- aws:createStack
- aws:createTags
- aws:loop
- aws:pause
- aws:runInstances
- aws:sleep

isEnd

이 옵션은 특정 단계 종료 시 자동화를 중지합니다. 단계가 실패하거나 성공한 경우 자동화가 중지됩니다. 기본값은 false입니다.

유형: Boolean

유효한 값: true | false

필수 항목 여부: 아니요

nextStep

한 단계를 성공적으로 완료한 후 다음에 처리할 자동화의 단계를 지정합니다.

타입: 문자열

필수 항목 여부: 아니요

isCritical

자동화의 성공적 완료에 대해 단계를 심각으로 지정합니다. 이 지정이 있는 단계가 실패하면 자동화는 자동화의 최종 상태를 실패로 보고합니다. 이 속성은 단계에서 명시적으로 정의하는 경우에만 평가됩니다. 단계에서 onFailure 속성이 Continue로 설정되면 기본값은 false입니다. 그렇지 않으면 이 옵션의 기본값은 true입니다.

타입: 부울

유효한 값: true | false

필수 항목 여부: 아니요

inputs

작업에 특정한 속성.

유형: 맵

필수 항목 여부: 예

예

```
---
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
    AavailabilityZone: "{{ 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` - 수동 승인을 위해 자동화 일시 중지

지정된 보안 주체가 작업을 승인하거나 거부할 때까지 자동화를 일시 중지합니다. 필요한 승인 횟수에 도달하면 자동화가 다시 시작됩니다. 런북의 `mainSteps` 섹션에 승인 단계를 삽입할 수 있습니다.

> **Note**
>
> 이 작업은 다중 계정 및 리전 자동화를 지원하지 않습니다. 이 작업에 대한 기본 제한 시간은 7일(604800초)이고 최대 값은 30일(2592000초)입니다. `aws:approve` 단계에서 `timeoutSeconds` 파라미터를 지정하여 제한 시간을 제한 또는 연장할 수 있습니다. 자동화 단계가 모든 필요한 승인 결정을 받기 전에 제한 시간에 도달한 경우 해당 단계 및 자동화가 실행을 중지하고 시간 초과 상태를 반환합니다.

다음 예에서 `aws:approve` 작업은 승인자 중 한 명이 자동화를 승인하거나 거부할 때까지 자동화를 일시 중지합니다. 승인하면 자동화는 간단한 PowerShell 명령을 실행합니다.

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

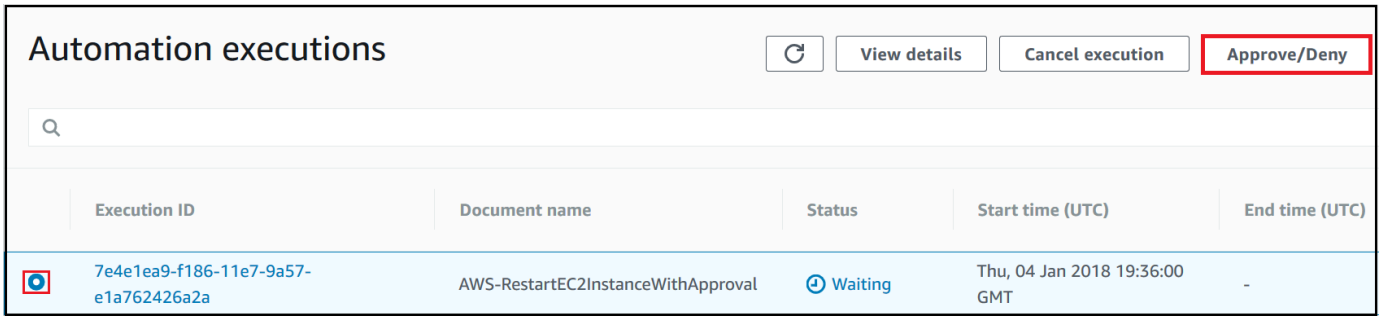
```
{
   "description":"RunInstancesDemo1",
   "schemaVersion":"0.3",
   "assumeRole":"{{ assumeRole }}",
   "parameters":{
      "assumeRole":{
         "type":"String"
      },
      "message":{
         "type":"String"
      }
   },
   "mainSteps":[
      {
         "name":"approve",
         "action":"aws:approve",
         "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"
               ]
            }
         }
      }
   ]
}
```

콘솔에서 승인을 기다리는 자동화를 승인 또는 거부할 수 있습니다.

대기 중인 자동화를 승인하거나 거부하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 왼쪽 탐색 창에서 Automation을 선택합니다.
3. 대기 중 상태와 함께 자동화 옆의 옵션을 선택합니다.



4. 승인/거부를 선택합니다.
5. 자동화의 세부 정보를 검토합니다.
6. 승인 또는 거부를 선택하고 필요에 따라 설명을 입력한 후 제출을 선택합니다.

입력 예제

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

Automation 승인을 위한 Amazon Simple Notification Service(Amazon SNS) 주제의 Amazon 리소스 이름(ARN)입니다. 실행서에서 `aws:approve` 단계를 지정하면 보안 주체가 Automation 단계를 승인하거나 거부해야 한다고 알려주는 메시지가 이 주제에 전송됩니다. Amazon SNS 주제의 제목에는 "Automation"을 접두사로 지정해야 합니다.

타입: 문자열

필수 항목 여부: 아니요

Message

승인 요청을 전송할 때 Amazon SNS 주제에 포함할 정보입니다. 최대 메시지 길이는 4,096자입니다.

타입: 문자열

필수 항목 여부: 아니요

MinRequiredApprovals

자동화를 다시 시작하는 데 필요한 최소 승인 횟수입니다. 값을 지정하지 않으면 기본값 1이 사용됩니다. 이 파라미터의 값은 양수여야 합니다. 이 파라미터의 값은 Approvers 파라미터에 정의된 승인자 수를 초과할 수 없습니다.

유형: 정수

필수 항목 여부: 아니요

승인자

작업을 승인하거나 거부할 수 있는 AWS 인증 보안 주체의 목록입니다. 최대 승인자자 수는 10명입니다. 다음 형식을 사용하여 보안 주체를 지정할 수 있습니다.

- 사용자 이름
- 사용자 ARN
- IAM 역할 ARN
- IAM 수임 역할 ARN

유형: StringList

필수 여부: 예

EnhancedApprovals

이 입력은 Change Manager 템플릿에서만 사용됩니다. 작업을 승인하거나 거부할 수 있는 AWS의 인증된 보안 주체, IAM 보안 주체 유형, 최소 승인자 수 목록. 다음은 그 예제입니다.

```
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
```

유형: StringList

필수 여부: 예

출력

ApprovalStatus

단계의 승인 상태입니다. 상태는 Approved, Rejected, Waiting 중 하나일 수 있습니다. Waiting은 승인자의 입력을 대기 중임을 나타냅니다.

타입: 문자열

ApproverDecisions

각 승인자의 승인 결정을 포함하는 JSON 맵입니다.

형식: MapList

`aws:assertAwsResourceProperty` - AWS 리소스 상태 또는 이벤트 상태 어설션

`aws:assertAwsResourceProperty` 작업은 특정 Automation 단계에 대해 특정 리소스 상태 또는 이벤트 상태를 어설션할 수 있게 합니다. 예를 들어 Automation 단계에서 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스가 시작할 때까지 대기하도록 지정할 수 있습니다. 그런 다음 DesiredValue 속성 `running`을 사용하여 Amazon EC2 DescribeInstanceStatus API 작업을 호출합니다. 이렇게 하면 자동화에서 실행 인스턴스를 기다렸다가 실제로 인스턴스가 실행 중일 때 계속 진행됩니다.

이 작업을 사용하는 방법에 관한 자세한 내용은 추가 런북 예제 섹션을 참조하세요.

Input

입력은 선택하는 API 작업에 의해 정의됩니다.

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"
    ]
  }
}
```

Service

실행하려는 API 작업을 포함하는 AWS 서비스 네임스페이스입니다. 예를 들면 Systems Manager의 네임스페이스는 ssm입니다. Amazon EC2의 네임스페이스는 ec2입니다. AWS CLI 명령 참조의 사용 가능한 서비스 섹션에서 지원되는 AWS 서비스 네임스페이스 목록을 볼 수 있습니다.

타입: 문자열

필수 항목 여부: 예

Api

실행할 API 작업의 이름입니다. 다음 [서비스 참조(Services Reference)] 페이지의 왼쪽 탐색 영역에서 서비스를 선택하여 API 작업(메서드라고도 함)을 볼 수 있습니다. 호출할 서비스에 대한 클라이언트 섹션에서 메서드를 선택합니다. 예를 들어 Amazon Relational Database Service(Amazon RDS)에 대한 모든 API 작업(메서드)이 Amazon RDS 메서드 페이지에 나열됩니다.

타입: 문자열

필수 항목 여부: 예

API 작업 입력

하나 이상의 API 작업 입력입니다. 다음 서비스 참조 페이지의 왼쪽 탐색 영역에서 서비스를 선택하여 사용할 수 있는 입력(파라미터)을 볼 수 있습니다. 호출할 서비스에 대한 클라이언트 섹션에서 메서드를 선택합니다. 예를 들어 Amazon RDS에 대한 모든 메서드는 Amazon RDS 메서드 페이지에 나열됩니다. describe_db_instances 메서드를 선택하고 아래로 스크롤하여 DBInstanceIdentifier, Name 및 Values 같은 사용할 수 있는 파라미터를 볼 수 있습니다. 두 개 이상의 입력을 지정하려면 다음 형식을 사용합니다.

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"
}
```

형식: 선택한 API 작업에 의해 결정됨

필수 항목 여부: 예

PropertySelector

응답 객체의 특정 속성에 대한 JSONPath입니다. 다음 서비스 참조 페이지의 왼쪽 탐색 영역에서 서비스를 선택하여 응답 객체를 볼 수 있습니다. 호출할 서비스에 대한 클라이언트 섹션에서 메서드를 선택합니다. 예를 들어 Amazon RDS에 대한 모든 메서드는 Amazon RDS 메서드 페이지에 나열됩니다. describe_db_instances 메서드를 선택하고 Response Structure(응답 구조) 섹션이 있는 아래쪽으로 스크롤합니다. DBInstances가 응답 객체로 나열됩니다.

타입: 문자열

필수 항목 여부: 예

DesiredValues

필요한 상태 또는 자동화를 계속 진행하기 위해 충족되어야 할 상태입니다. 부울 값을 지정할 경우 True 또는 False 같은 대문자를 사용해야 합니다.

유형: StringList

필수 항목 여부: 예

`aws:branch` - 조건부 자동화 단계 실행

`aws:branch` 작업을 통해 한 단계에서 여러 선택 항목을 평가한 다음 평가 결과에 따라 실행서의 다른 단계로 이동하는 동적 자동화를 생성할 수 있습니다.

단계에 대한 `aws:branch` 작업을 지정할 경우 해당 자동화에서 평가해야 하는 `Choices`를 지정할 수 있습니다. `Choices`는 실행서의 `Parameters` 섹션에 지정한 값이나 이전 단계의 출력으로 생성된 동적 값에 기초할 수 있습니다. 이 자동화는 부울 식을 사용하여 각 선택을 평가합니다. 첫 번째 선택 항목이 true이면 이 자동화는 해당 선택 항목에 지정된 단계로 이동합니다. 첫 번째 선택 항목이 false이면 이 자동화는 다음 선택 항목을 평가합니다. 이 자동화는 true 선택 항목을 처리할 때까지 각 선택 항목을 계속 평가합니다. 그런 다음 이 자동화는 true인 선택 항목에 지정된 단계로 이동합니다.

값이 true인 선택 항목이 없을 경우 이 자동화는 단계에 `default` 값이 포함되었는지 검사합니다. default 값은 선택 항목 중에 true인 항목이 없는 경우 자동화에서 이동해야 하는 단계를 정의합니다. 단계에 대한 `default` 값이 지정되지 않은 경우 자동화는 실행서의 다음 단계를 처리합니다.

`aws:branch` 작업은 `And`, `Not` 및 `Or` 연산자의 조합을 사용한 복합 선택 항목 평가를 지원합니다. 예제 실행서와 다양한 연산자를 사용하는 예제 등 `aws:branch`를 사용하는 방법에 대한 자세한 내용은 런북에서 조건문 사용 섹션을 참조하세요.

Input

단계에서 하나 이상의 Choices를 지정합니다. Choices는 실행서의 Parameters 섹션에 지정한 값이나 이전 단계의 출력으로 생성된 동적 값에 기초할 수 있습니다. 다음은 파라미터를 평가하는 YAML 샘플입니다.

```
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
```

다음은 이전 단계의 출력을 평가하는 YAML 샘플입니다.

```
mainSteps:
- name: chooseOS
  action: aws:branch
  inputs:
    Choices:
    - NextStep: runPowerShellCommand
      Variable: "{{Name of a response object. For example: GetInstance.platform}}"
      StringEquals: Windows
    - NextStep: runShellCommand
      Variable: "{{Name of a response object. For example: GetInstance.platform}}"
      StringEquals: Linux
    Default:
      sleep3
```

Choices

자동화에서 처리할 다음 단계를 결정할 때 평가해야 하는 하나 이상의 표현식입니다. Choices는 부울 식을 사용하여 평가됩니다. 각 선택 항목은 다음 옵션을 정의해야 합니다.

- NextStep: 지정된 선택 항목이 true인 경우 처리할 실행서의 다음 단계.

- Variable: 실행서의 `Parameters` 섹션에 정의된 파라미터의 이름을 지정합니다. 또는 실행서의 이전 단계에서 얻은 출력 객체를 지정합니다. `aws:branch`용 변수 생성에 대한 자세한 내용은 출력 변수 생성 정보 섹션을 참조하세요.
- Operation: 선택 항목(choice)을 평가하는 데 사용되는 기준입니다. `aws:branch` 작업에서는 다음 연산을 지원합니다.

  문자열 연산

  - StringEquals
  - EqualsIgnoreCase
  - StartsWith
  - EndsWith
  - 포함

  수치 연산

  - NumericEquals
  - NumericGreater
  - NumericLesser
  - NumericGreaterOrEquals
  - NumericLesser
  - NumericLesserOrEquals

  부울 연산

  - BooleanEquals

  > **Important**
  > 실행서를 생성하면 시스템에서 실행서의 각 작업을 검증합니다. 작업이 지원되지 않는 경우 실행서를 생성할 때 오류가 발생합니다.

기본값

`Choices` 중에 true인 항목이 없는 경우 자동화에서 이동해야 할 단계의 이름입니다.

타입: 문자열

필수 항목 여부: 아니요

> **Note**
>
> `aws:branch` 작업은 `And`, `Or` 및 `Not` 연산자를 지원합니다. 연산자를 사용하는 `aws:branch`의 예는 런북에서 조건문 사용 섹션을 참조하세요.

`aws:changeInstanceState` - 인스턴스 상태 변경 또는 어설션

인스턴스의 상태를 변경하거나 어설션합니다.

이 작업은 어설션 모드에서 사용될 수 있습니다(API를 실행하여 상태를 변경하는 것이 아니라 인스턴스가 원하는 상태에 있는지 확인함). 어설션 모드를 사용하려면 `CheckStateOnly` 파라미터를 true로 설정합니다. 이 모드는 배경에서 장시간 실행할 수 있는 비동기식 명령인 Sysprep 명령을 Windows에서 실행할 때 유용합니다. Amazon Machine Image(AMI)를 생성하기 전에 인스턴스가 중지되게 할 수 있습니다.

> **Note**
>
> 이 작업의 기본 제한 시간 값은 3600초(1시간)입니다. `aws:changeInstanceState` 단계에서 `timeoutSeconds` 파라미터를 지정하여 제한 시간을 제한 또는 연장할 수 있습니다.

입력

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

인스턴스의 ID.

유형: StringList

필수 항목 여부: 예

CheckStateOnly

false인 경우 인스턴스 상태를 원하는 상태로 설정합니다. true인 경우 폴링을 사용하여 원하는 상태를 어설션합니다.

기본값: `false`

타입: 부울

필수 항목 여부: 아니요

DesiredState

원하는 상태. `running`으로 설정하는 경우 이 작업은 완료하기 전에 Amazon EC2 상태가 `Running`, 인스턴스 상태가 `OK` 그리고 시스템 상태가 `OK`가 될 때까지 기다립니다.

타입: 문자열

유효한 값: `running` | `stopped` | `terminated`

필수 항목 여부: 예

Force

설정되면 인스턴스가 중지합니다. 인스턴스는 파일 시스템 캐시 또는 파일 시스템 메타데이터를 플러시하지 않습니다. 이 옵션을 사용하는 경우 파일 시스템 확인 및 복구 절차를 수행해야 합니다. 이 옵션은 Windows Server용 EC2 인스턴스에는 권장하지 않습니다.

타입: 부울

필수 항목 여부: 아니요

AdditionalInfo

예약.

타입: 문자열

필수 항목 여부: 아니요

출력

None

aws:copyImage - Amazon Machine Image 복사 또는 암호화

모든 AWS 리전의 Amazon Machine Image(AMI)를 현재 리전에 복사합니다. 이 작업은 새 AMI를 암호화할 수도 있습니다.

Input

이 작업은 대부분의 `CopyImage` 파라미터를 지원합니다. 자세한 내용은 CopyImage를 참조하십시오.

다음 예에서는 서울 리전의 AMI 복사본을 생성합니다(`SourceImageID`: ami-0fe10819. `SourceRegion`: ap-northeast-2). 새 AMI가 Automation 작업을 시작한 리전에 복사됩니다. `Encrypted` 플래그(옵션)가 `true`로 설정되어 있으므로 복사한 AMI가 암호화됩니다.

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

원본 AMI가 존재하는 리전입니다.

타입: 문자열

필수 항목 여부: 예

SourceImageId

원본 리전에서 복사할 AMI ID입니다.

타입: 문자열

필수 항목 여부: 예

ImageName

새 이미지의 이름입니다.

타입: 문자열

필수 항목 여부: 예

ImageDescription

대상 이미지에 대한 설명입니다.

타입: 문자열

필수 항목 여부: 아니요

Encrypted

대상 AMI를 암호화합니다.

타입: 부울

필수 항목 여부: 아니요

KmsKeyId

복사 작업 중에 이미지의 스냅샷을 암호화할 때 사용할 AWS KMS key의 전체 Amazon 리소스 이름(ARN)입니다. 자세한 내용은 CopyImage를 참조하십시오.

타입: 문자열

필수 항목 여부: 아니요

ClientToken

요청 멱등성을 보장하기 위해 제공하는 고유의 대/소문자 구분 식별자입니다. 자세한 내용은 CopyImage를 참조하십시오.

타입: 문자열

필수 항목 여부: 아니요

출력

ImageId

복사한 이미지의 ID입니다.

ImageState

복사한 이미지의 상태입니다.

유효한 값: `available` | `pending` | `failed`

`aws:createImage` - Amazon Machine Image 생성

실행 중이거나 중지 중이거나 중지된 인스턴스에서 Amazon Machine Image(AMI)를 생성합니다.

Input

이 작업은 다음 `CreateImage` 파라미터를 지원합니다. 자세한 내용은 CreateImage를 참조하십시오.

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

인스턴스의 ID

타입: 문자열

필수 항목 여부: 예

ImageName

이미지의 이름입니다.

타입: 문자열

필수 항목 여부: 예

ImageDescription

이미지의 설명.

타입: 문자열

필수 항목 여부: 아니요

NoReboot

부울 리터럴.

기본적으로 Amazon Elastic Compute Cloud(Amazon EC2)는 이미지를 생성하기 전에 인스턴스를 종료하고 재부팅하려고 시도합니다. [재부팅 안 함(No Reboot)] 옵션이 `true`로 설정된 경우 Amazon EC2는 이미지를 생성하기 전에 인스턴스를 종료하지 않습니다. 이 옵션을 사용하는 경우 생성된 이미지의 파일 시스템 무결성을 보장할 수 없습니다.

인스턴스에서 AMI를 생성한 후 인스턴스가 실행되지 않게 하려면 먼저 `aws:changeInstanceState` - 인스턴스 상태 변경 또는 어설션 작업을 사용하여 인스턴스를 중지한 다음 [NoReboot] 옵션을 `true`로 설정한 상태에서 이 `aws:createImage` 작업을 사용합니다.

타입: 부울

필수 항목 여부: 아니요

BlockDeviceMappings

해당 인스턴스용 블록 디바이스.

유형: 맵

필수 항목 여부: 아니요

출력

ImageId

새롭게 생성된 이미지의 ID.

타입: 문자열

ImageState

이미지의 현재 상태입니다. 상태가 사용 가능한 경우 이미지가 성공적으로 등록되고 인스턴스를 시작하는 데 사용할 수 있습니다.

타입: 문자열

aws:createStack - AWS CloudFormation 스택 생성

템플릿에서 AWS CloudFormation 스택을 생성합니다.

CloudFormation 스택 생성에 대한 추가 정보는 AWS CloudFormation API Reference의 CreateStack을 참조하세요.

입력

YAML

```
name: makeStack
action: aws:createStack
maxAttempts: 1
onFailure: Abort
inputs:
  Capabilities:
  - CAPABILITY_IAM
  StackName: myStack
  TemplateURL: http://s3.amazonaws.com/DOC-EXAMPLE-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/DOC-EXAMPLE-BUCKET/myStackTemplate",
        "TimeoutInMinutes": 5,
        "Parameters": [
          {
            "ParameterKey": "LambdaRoleArn",
            "ParameterValue": "{{LambdaAssumeRole}}"
          },
          {
            "ParameterKey": "createdResource",
            "ParameterValue": "createdResource-{{automation:EXECUTION_ID}}"
          }
    }
}
```

기능

CloudFormation에서 특정 스택을 생성할 수 있으려면 지정해야 하는 값의 목록입니다. 일부 스택 템플릿은 AWS 계정의 권한에 영향을 줄 수 있는 리소스를 포함합니다. 이러한 스택에서는 이 파라미터를 지정하여 스택의 기능을 명시적으로 확인해야 합니다.

유효값에는 CAPABILITY_IAM, CAPABILITY_NAMED_IAM 및 CAPABILITY_AUTO_EXPAND이 있습니다.

CAPABILITY_IAM 및 CAPABILITY_NAMED_IAM

IAM 리소스가 있는 경우 어느 기능이든 지정할 수 있습니다. 사용자 정의 이름을 갖는 IAM 리소스가 있는 경우 CAPABILITY_NAMED_IAM을 지정해야 합니다. 이 파라미터를 지정하지 않으면 이 작업은 InsufficientCapabilities 오류를 반환합니다. 다음 리소스의 경우 CAPABILITY_IAM 또는 CAPABILITY_NAMED_IAM을 지정해야 합니다.

- AWS::IAM::AccessKey
- AWS::IAM::Group
- AWS::IAM::InstanceProfile
- AWS::IAM::Policy
- AWS::IAM::Role

- AWS::IAM::User
- AWS::IAM::UserToGroupAddition

스택 템플릿에 이러한 리소스가 포함되어 있는 경우, 관련 권한을 모두 검토하고 필요에 따라 권한을 편집하는 것이 좋습니다.

자세한 내용은 AWS CloudFormation 템플릿에서 IAM 리소스 승인을 참조하세요.

CAPABILITY_AUTO_EXPAND

일부 템플릿에는 매크로가 포함되어 있습니다. 매크로는 템플릿에서 사용자 지정 처리를 수행합니다. 여기에는 찾기 및 바꾸기 작업과 같은 간단한 작업과 전체 템플릿의 광범위한 변환을 위한 모든 방법을 포함할 수 있습니다. 그러므로 사용자는 일반적으로 처리된 템플릿에서 변경 세트를 생성하여 실제로 스택을 생성하기 전에 매크로로 인한 변경 사항을 검토할 수 있습니다. 스택 템플릿에 매크로가 하나 이상 포함되어 있고 변경 세트의 결과 변경 사항을 먼저 검토하지 않고 처리된 템플릿에서 직접 스택을 생성하도록 선택한 경우 이 기능을 확인해야 합니다.

자세한 내용은 AWS CloudFormation 사용 설명서의 템플릿에서 사용자 정의 처리를 수행하는 AWS CloudFormation 매크로 사용을 참조하세요.

형식: 문자열 배열

유효 값: `CAPABILITY_IAM | CAPABILITY_NAMED_IAM | CAPABILITY_AUTO_EXPAND`

필수 항목 여부: 아니요

ClientRequestToken

이 CreateStack 요청에 대한 고유 식별자입니다. 이 단계에서 maxAttempts를 1보다 큰 값으로 설정한 경우 이 토큰을 지정하십시오. 이 토큰을 지정하면 사용자가 같은 이름으로 새 스택을 생성하려는 것이 아님을 CloudFormation에서 알게 됩니다.

타입: 문자열

필수 항목 여부: 아니요

길이 제약: 최소 길이 1. 최대 길이 128.

패턴: [a-zA-Z0-9][-a-zA-Z0-9]*

DisableRollback

스택 생성에 실패한 경우 스택의 롤백을 해제하려면 `true`로 설정합니다.

조건: DisableRollback 또는 OnFailure 파라미터를 지정할 수 있지만, 둘 다 지정하지 않을 수도 있습니다.

기본값: false

타입: 부울

필수 항목 여부: 아니요

NotificationARNs

스택 관련 이벤트를 게시할 Amazon Simple Notification Service(Amazon SNS) 주제 ARN입니다. Amazon SNS 콘솔(https://console.aws.amazon.com/sns/v3/home)을 사용하여 SNS 주제 ARN을 찾을 수 있습니다.

형식: 문자열 배열

배열 멤버: 최대 항목 수 5개.

필수 항목 여부: 아니요

OnFailure

스택 생성에 실패한 경우 취할 조치를 결정합니다. DO_NOTHING, ROLLBACK 또는 DELETE를 지정해야 합니다.

조건: OnFailure 또는 DisableRollback 파라미터를 지정할 수 있지만, 둘 다 지정하지 않을 수도 있습니다.

기본값: ROLLBACK

타입: 문자열

유효한 값: DO_NOTHING | ROLLBACK | DELETE

필수 항목 여부: 아니요

파라미터

스택에 대한 입력 파라미터를 지정하는 Parameter 구조의 목록. 자세한 내용은 Parameter 데이터 형식을 참조하십시오.

형식: Parameter 객체 배열

필수 항목 여부: 아니요

ResourceTypes

이 스택 생성 작업에 대해 함께 작업할 수 있는 권한을 갖는 템플릿 리소스 유형. 예: `AWS::EC2::Instance`, `AWS::EC2::*` 또는 `Custom::MyCustomInstance`. 템플릿 리소스 유형을 기술하려면 다음 구문을 사용합니다.

- 모든 AWS 리소스의 경우:

```
AWS::*
```

- 모든 사용자 지정 리소스의 경우:

```
Custom::*
```

- 특정 사용자 지정 리소스의 경우:

```
Custom::logical_ID
```

- 특정 AWS 서비스에 대한 모든 리소스의 경우:

```
AWS::service_name::*
```

- 특정 AWS 리소스의 경우:

```
AWS::service_name::resource_logical_ID
```

리소스 유형 목록에 사용자가 생성 중인 리소스가 포함되어 있지 않으면 스택 생성에 실패합니다. 기본적으로 CloudFormation은 모든 리소스 유형에 권한을 부여합니다. IAM은 IAM 정책의 CloudFormation별 조건 키에 이 파라미터를 사용합니다. 자세한 내용은 AWS Identity and Access Management을 통한 액세스 제어를 참조하십시오.

형식: 문자열 배열

길이 제약: 최소 길이 1. 최대 길이는 256.

필수 항목 여부: 아니요

RoleARN

CloudFormation이 스택을 생성하기 위해 수임하는 IAM 역할의 Amazon 리소스 이름(ARN)입니다. CloudFormation은 역할의 자격 증명을 사용하여 사용자를 대신하여 호출합니다. CloudFormation은 스택의 모든 향후 작업에 항상 이 역할을 사용합니다. 사용자에게 스택에서 작업할 수 있는 권한

이 있는 경우에는 역할을 전달할 수 있는 권한은 없더라도 CloudFormation에서 이 역할을 사용합니다. 역할이 최소한의 권한을 부여하는지 확인하십시오.

값을 지정하지 않으면 CloudFormation에서는 이전에 스택과 연결된 역할을 사용합니다. 사용 가능한 역할이 없으면 CloudFormation에서는 사용자 자격 증명으로부터 생성되는 임시 세션을 사용합니다.

타입: 문자열

길이 제약: 최소 길이는 20. 최대 길이는 2,048.

필수 항목 여부: 아니요

StackName

스택과 연결되어 있는 이름. 이름은 스택을 생성하는 리전에서 고유해야 합니다.

> **Note**
>
> 스택 이름에는 영숫자(대소문자 구분)와 하이픈만 사용할 수 있습니다. 영문자로 시작해야 하고 128자 이하여야 합니다.

타입: 문자열

필수 항목 여부: 예

StackPolicyBody

스택 정책 본문이 포함된 구조. 자세한 내용은 스택 리소스에 대한 업데이트 방지를 참조하십시오.

조건: `StackPolicyBody` 또는 `StackPolicyURL` 파라미터를 지정할 수 있지만, 둘 다 지정하지 않을 수도 있습니다.

유형: 문자열

길이 제약 조건: 최소 길이는 1입니다. 최대 길이는 16384입니다.

필수 항목 여부: 아니요

StackPolicyURL

스택 정책이 포함된 파일의 위치. URL은 스택과 동일한 리전의 S3 버킷에 있는 정책을 가리켜야 합니다. 스택 정책에 허용되는 최대 파일 크기는 16KB입니다.

조건: StackPolicyBody 또는 StackPolicyURL 파라미터를 지정할 수 있지만, 둘 다 지정하지 않을 수도 있습니다.

유형: 문자열

길이 제약: 최소 길이 1. 최대 길이 1350.

필수 항목 여부: 아니요

Tags

이 스택과 연결할 키-값 페어. 또한 CloudFormation은 이러한 태그를 해당 스택에 생성된 리소스로 전파합니다. 태그를 최대 10개까지 지정할 수 있습니다.

형식: Tag 객체 배열

필수 항목 여부: 아니요

TemplateBody

최소 길이가 1바이트이고 최대 길이가 51,200바이트인 템플릿 본문이 포함된 구조. 자세한 내용은 Template Anatomy를 참조하십시오.

조건: TemplateBody 또는 TemplateURL 파라미터를 지정할 수 있지만, 둘 다 지정하지 않을 수도 있습니다.

유형: 문자열

길이 제약: 최소 길이 1.

필수 항목 여부: 아니요

TemplateURL

템플릿 본문이 포함된 파일의 위치. URL은 S3 버킷에 있는 템플릿을 가리켜야 합니다. 템플릿에 허용되는 최대 크기는 460,800바이트입니다. 자세한 내용은 Template Anatomy를 참조하십시오.

조건: TemplateBody 또는 TemplateURL 파라미터를 지정할 수 있지만, 둘 다 지정하지 않을 수도 있습니다.

유형: 문자열

길이 제약: 최소 길이 1. 최대 길이 1024.

필수 항목 여부: 아니요

TimeoutInMinutes

스택 상태가 CREATE_FAILED가 되기 전에 경과할 수 있는 시간. DisableRollback이 설정되지 않거나 false로 설정되면 스택은 롤백됩니다.

타입: 정수

유효 범위: 최소값 1.

필수 항목 여부: 아니요

결과

StackId

스택의 고유 식별자.

타입: 문자열

StackStatus

스택의 현재 상태.

타입: 문자열

유효 값: 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

필수 여부: 예

StackStatusReason

스택 상태와 관련된 성공 또는 실패 메시지.

타입: 문자열

필수 항목 여부: 아니요

자세한 내용은 CreateStack을 참조하십시오.

보안 고려 사항

aws:createStack 작업을 사용하려면 먼저 IAM Automation 수임 역할에 다음 정책을 할당해야 합니다. assume role에 대한 자세한 내용은 작업 1: Automation을 위한 서비스 역할 생성 섹션을 참조하세요.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "sqs:*",
            "cloudformation:CreateStack",
            "cloudformation:DescribeStacks"
         ],
         "Resource":"*"
      }
   ]
}
```

aws:createTags - AWS 리소스에 대한 태그 생성

Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 또는 AWS Systems Manager 관리형 인스턴스에 대한 새 태그를 생성합니다.

Input

이 작업은 대부분의 Amazon EC2 CreateTags 및 Systems Manager AddTagsToResource 파라미터를 지원합니다. 자세한 내용은 CreateTags 및 AddTagsToResource를 참조하십시오.

다음 예에서는 Amazon Machine Image(AMI)와 인스턴스를 특정 부서의 프로덕션 리소스로 태그 지정하는 방법을 보여줍니다.

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

태그 지정할 리소스의 ID입니다. 리소스 유형이 "EC2"가 아니면 이 필드는 항목을 하나만 포함할 수 있습니다.

형식: 문자열 목록

필수 항목 여부: 예

Tags

리소스와 연결할 태그입니다.

형식: 맵 목록

필수 항목 여부: 예

ResourceType

태그 지정할 리소스의 유형입니다. 제공되지 않은 경우 기본값 "EC2"가 사용됩니다.

타입: 문자열

필수 항목 여부: 아니요

유효한 값: `EC2` | `ManagedInstance` | `MaintenanceWindow` | `Parameter`

출력

None

`aws:deleteImage` - Amazon Machine Image 삭제

지정된 Amazon Machine Image(AMI) 및 관련된 모든 스냅샷을 삭제합니다.

Input

이 작업은 하나의 파라미터만 지원합니다. 자세한 내용은 DeregisterImage and DeleteSnapshot 문서를 참조하십시오.

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

삭제되는 이미지의 ID.

타입: 문자열

필수 항목 여부: 예

출력

None

aws:deleteStack - AWS CloudFormation 스택 삭제

AWS CloudFormation 스택을 삭제합니다.

입력

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

이 DeleteStack 요청에 대한 고유 식별자입니다. 요청을 재시도하여 사용자가 동일한 이름의 스택을 삭제하지 않을 것임을 CloudFormation에 알릴 계획인 경우 이 토큰을 지정합니다. DeleteStack 요청을 재시도하여 CloudFormation에서 수신했음을 확인할 수 있습니다.

유형: 문자열

길이 제약: 최소 길이는 1. 최대 길이는 128.

패턴: [a-zA-Z][-a-zA-Z0-9]*

필수 항목 여부: 아니요

RetainResources.member.N

이 입력은 DELETE_FAILED 상태에 있는 스택에만 적용됩니다. 보유할 리소스에 대한 논리적 리소스 ID 목록입니다. 삭제하는 동안 CloudFormation은 스택을 삭제하지만 보유한 리소스는 삭제하지 않습니다.

비어 있지 않은 S3 버킷 같은 리소스를 삭제할 수 없지만, 스택을 삭제하고자 하는 경우에 리소스를 보유하면 유용합니다.

형식: 문자열 배열

필수 항목 여부: 아니요

RoleARN

CloudFormation이 스택을 생성하기 위해 수임하는 AWS Identity and Access Management(IAM) 역할의 Amazon 리소스 이름(ARN)입니다. CloudFormation은 역할의 자격 증명을 사용하여 사용자

를 대신하여 호출합니다. CloudFormation은 스택의 모든 향후 작업에 항상 이 역할을 사용합니다. 사용자에게 스택에서 작업할 수 있는 권한이 있는 경우에는 역할을 전달할 수 있는 권한은 없더라도 CloudFormation에서 이 역할을 사용합니다. 역할이 최소한의 권한을 부여하는지 확인하십시오.

값을 지정하지 않으면 CloudFormation에서는 이전에 스택과 연결된 역할을 사용합니다. 사용 가능한 역할이 없으면 CloudFormation에서는 사용자 자격 증명으로부터 생성되는 임시 세션을 사용합니다.

타입: 문자열

길이 제약: 최소 길이는 20. 최대 길이는 2,048.

필수 항목 여부: 아니요

StackName

스택이 연결되어 있는 이름 또는 고유한 스택 ID입니다.

타입: 문자열

필수 항목 여부: 예

보안 고려 사항

`aws:deleteStack` 작업을 사용하려면 먼저 IAM Automation 수임 역할에 다음 정책을 할당해야 합니다. assume role에 대한 자세한 내용은 작업 1: Automation을 위한 서비스 역할 생성 섹션을 참조하세요.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "sqs:*",
            "cloudformation:DeleteStack",
            "cloudformation:DescribeStacks"
         ],
         "Resource":"*"
      }
   ]
}
```

`aws:executeAutomation` - 또 다른 자동화 실행

보조 실행서를 호출하여 보조 자동화를 실행합니다. 이 작업을 수행하여 가장 일반적인 작업에 대한 실행서를 생성하고 자동화 중에 해당 실행서를 참조할 수 있습니다. 이 작업을 수행하면 비슷한 실행서 간에 복제 단계가 필요하지 않으므로 실행서를 간소화할 수 있습니다.

보조 자동화는 기본 자동화를 시작한 사용자의 컨텍스트에서 실행됩니다. 이는 보조 자동화가 첫 자동화를 시작한 사용자와 동일한 AWS Identity and Access Management(IAM) 역할 또는 사용자를 사용한다는 의미입니다.

> **Important**
>
> 수임 역할(iam:passRole 정책을 사용하는 역할)을 사용하는 보조 자동화에서 파라미터를 지정하는 경우 기본 자동화를 시작한 사용자 또는 역할에 보조 자동화에서 지정한 수임 역할을 전달할 권한이 있어야 합니다. 자동화의 assume role 설정에 대한 자세한 내용은 방법 2: IAM을 사용하여 Automation을 위한 역할 구성을 참조하십시오.

입력

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

단계 중에 실행할 보조 실행서의 이름입니다. 같은 AWS 계정에 있는 실행서의 경우 실행서 이름을 지정합니다. 다른 AWS 계정에서 공유되는 실행서의 경우 실행서의 Amazon 리소스 이름(ARN)을 지정합니다. 공유 실행서 사용에 대한 자세한 내용은 공유 SSM 문서 사용 섹션을 참조하세요.

타입: 문자열

필수 항목 여부: 예

DocumentVersion

실행할 보조 실행서의 버전입니다. 지정하지 않은 경우 Automation은 기본 실행서 버전을 실행합니다.

타입: 문자열

필수 항목 여부: 아니요

MaxConcurrency

이 태스크를 병렬로 실행할 수 있는 최대 대상 수입니다. 숫자(예: 10) 또는 백분율(예: 10%)로 지정할 수 있습니다.

타입: 문자열

필수 항목 여부: 아니요

MaxErrors

시스템에서 추가 대상에 대한 자동화 실행을 중지하기 전에 허용되는 오류 수입니다. 오류의 절대 개수(예: 10개)를 지정하거나 대상 집합의 비율(예: 10%)을 지정할 수 있습니다. 예를 들어 3을 지정할 경우 네 번째 오류가 수신되면 자동화 실행이 중지됩니다. 0을 지정하면 첫 번째 오류 결

과가 반환된 후 추가 대상에서 자동화 실행이 중지됩니다. 50개의 리소스에서 자동화를 실행하고 MaxErrors를 10%로 설정하면 6번째 오류 수신 시 추가 대상에서 자동화 실행이 중지됩니다.

MaxErrors 임계값에 도달했을 때 이미 실행 중인 자동화는 완료될 수 있지만 일부는 실패할 수도 있습니다. 지정된 MaxErrors보다 많이 자동화 실패가 발생하지 않게 자동화가 한 번에 하나씩 진행되도록 MaxConcurrency를 1로 설정합니다.

타입: 문자열

필수 항목 여부: 아니요

RuntimeParameters

보조 실행서에 필요한 파라미터입니다. 매핑은 {"parameter1" : "value1", "parameter2" : "value2" } 형식을 사용합니다.

유형: 맵

필수 항목 여부: 아니요

Tags

리소스에 할당하는 선택적 메타데이터입니다. 자동화에 대해 최대 5개의 태그를 지정할 수 있습니다.

형식: MapList

필수 항목 여부: 아니요

TargetLocations

위치는 자동화를 실행하려는 AWS 리전 및/또는 AWS 계정의 조합입니다. 최소 1개 항목을 지정해야 하며 최대 100개 항목을 지정할 수 있습니다.

형식: MapList

필수 항목 여부: 아니요

TargetMaps

대상 리소스에 대한 문서 파라미터의 키-값 매핑 목록입니다. Targets와 TargetMaps는 함께 지정할 수 없습니다.

형식: MapList

필수 항목 여부: 아니요

TargetParameterName

속도 제어 자동화의 대상 리소스로 사용되는 파라미터의 이름입니다. `Targets`를 지정한 경우에만 필요합니다.

타입: 문자열

필수 항목 여부: 아니요

대상

대상 리소스에 대한 키-값 매핑 목록입니다. `TargetParameterName`를 지정한 경우에만 필요합니다.

형식: MapList

필수 항목 여부: 아니요

출력

출력

보조 자동화에서 생성된 출력입니다. *`Secondary_Automation_Step_Name`*.Output 형식을 사용하여 출력을 참조할 수 있습니다.

유형: StringList

예:

```
- 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

보조 자동화의 ID입니다.

타입: 문자열

상태 표시기

보조 자동화의 상태입니다.

타입: 문자열

aws:executeAwsApi - AWS API 작업 호출 및 실행

AWS API 작업을 호출하여 실행합니다. 일부 API 작업이 테스트되지 않았더라도 대부분의 API 작업이 지원됩니다. GetObject 작업과 같은 스트리밍 API 작업은 지원되지 않습니다. 사용하려는 API 작업이 스트리밍 작업인지 확실하지 않은 경우 API에 스트리밍 입력 또는 출력이 필요한지 여부를 결정하는 서비스에 대한 Boto3 설명서를 검토하세요. 이 작업에 사용되는 Boto3 버전은 정기적으로 업데이트됩니다. 그러나 새 Boto3 버전 릴리스 후 변경 사항이 이 작업에 반영되려면 최대 몇 주가 걸릴 수 있습니

다. 각 aws:executeAwsApi 작업은 최대 25초까지 실행할 수 있습니다. 이 작업을 사용하는 방법에 관한 자세한 내용은 추가 런북 예제 섹션을 참조하세요.

입력

입력은 선택하는 API 작업에 의해 정의됩니다.

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"
      }
   ]
}
```

Service

실행하려는 API 작업을 포함하는 AWS 서비스 네임스페이스입니다. AWS SDK for Python (Boto3)의 사용 가능한 서비스에서 지원되는 AWS 서비스 네임스페이스 목록을 볼 수 있습니다. 네임스

페이스는 클라이언트 섹션에서 찾을 수 있습니다. 예를 들면 Systems Manager의 네임스페이스는 ssm입니다. Amazon Elastic Compute Cloud(Amazon EC2)의 네임스페이스는 ec2입니다.

타입: 문자열

필수 항목 여부: 예

Api

실행할 API 작업의 이름입니다. 다음 [서비스 참조(Services Reference)] 페이지의 왼쪽 탐색 영역에서 서비스를 선택하여 API 작업(메서드라고도 함)을 볼 수 있습니다. 호출할 서비스에 대한 클라이언트 섹션에서 메서드를 선택합니다. 예를 들어 Amazon Relational Database Service(Amazon RDS)에 대한 모든 API 작업(메서드)이 Amazon RDS 메서드 페이지에 나열됩니다.

타입: 문자열

필수 항목 여부: 예

API 작업 입력

하나 이상의 API 작업 입력입니다. 다음 서비스 참조 페이지의 왼쪽 탐색 영역에서 서비스를 선택하여 사용할 수 있는 입력(파라미터)을 볼 수 있습니다. 호출할 서비스에 대한 클라이언트 섹션에서 메서드를 선택합니다. 예를 들어 Amazon RDS에 대한 모든 메서드는 Amazon RDS 메서드 페이지에 나열됩니다. describe_db_instances 메서드를 선택하고 아래로 스크롤하여 DBInstanceIdentifier, Name 및 Values 같은 사용할 수 있는 파라미터를 볼 수 있습니다.

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"
```

```
}
```

형식: 선택한 API 작업에 의해 결정됨

필수 항목 여부: 예

결과

출력은 선택한 API 작업의 응답을 기반으로 사용자가 지정합니다.

명칭

출력의 이름입니다.

타입: 문자열

필수 항목 여부: 예

Selector

응답 객체의 특정 속성에 대한 JSONPath입니다. 다음 서비스 참조 페이지의 왼쪽 탐색 영역에서 서비스를 선택하여 응답 객체를 볼 수 있습니다. 호출할 서비스에 대한 클라이언트 섹션에서 메서드를 선택합니다. 예를 들어 Amazon RDS에 대한 모든 메서드는 Amazon RDS 메서드 페이지에 나열됩니다. describe_db_instances 메서드를 선택하고 Response Structure(응답 구조) 섹션이 있는 아래쪽으로 스크롤합니다. DBInstances가 응답 객체로 나열됩니다.

형식: Integer, Boolean, String, StringList, StringMap, MapList

필수 항목 여부: 예

유형

응답 요소의 데이터 유형입니다.

형식: 다양

필수 항목 여부: 예

`aws:executeScript` - 스크립트 실행

지정된 런타임 및 핸들러를 사용하여 제공된 Python 또는 PowerShell 스크립트를 실행합니다. 각 `aws:executeScript` 작업은 최대 600초(10분)까지 실행할 수 있습니다. `aws:executeScript` 단계에서 `timeoutSeconds` 파라미터를 지정하여 시간 제한을 제한할 수 있습니다.

함수에서 return 문을 사용하여 출력 페이로드에 출력을 추가합니다. `aws:executeScript` 작업에 대한 출력 정의 예시는 예제 2: 스크립팅된 실행서의 내용을 참조하세요. 실행서에 있는 `aws:executeScript` 작업의 출력을 지정하는 Amazon CloudWatch Logs 로그 그룹으로 전송할 수 있습니다. 자세한 내용은 CloudWatch Logs로 Automation 작업 출력 로깅 단원을 참조하십시오.

`aws:executeScript` 작업에서 CloudWatch Logs로 출력을 전송하거나 `aws:executeScript` 작업에 대해 지정한 스크립트가 AWS API 작업을 호출하는 경우, 런북을 실행하려면 AWS Identity and Access Management(IAM) 서비스 역할(또는 역할 수임)이 항상 필요합니다.

`aws:executeScript` 작업에는 다음과 같은 사전 설치된 PowerShell Core 모듈이 포함되어 있습니다.

- Microsoft.PowerShell.Host
- Microsoft.PowerShell.Management
- Microsoft.PowerShell.Security
- Microsoft.PowerShell.Utility
- PackageManagement
- PowerShellGet

사전 설치되지 않은 PowerShell Core 모듈을 사용하려면 스크립트에서 다음 명령과 같이 `-Force` 플래그를 사용하여 모듈을 설치해야 합니다. `AWSPowerShell.NetCore` 모듈은 지원되지 않습니다. *`ModuleName`*을 설치하려는 모듈로 바꿉니다.

```
Install-Module ModuleName -Force
```

스크립트에서 PowerShell Core cmdlet을 사용하려면 다음 명령과 같이 `AWS.Tools` 모듈을 사용하는 것이 좋습니다. 각 *`example resource placeholder`*를 사용자의 정보로 바꿉니다.

- Amazon S3 cmdlet입니다.

  ```
  Install-Module AWS.Tools.S3 -Force
  Get-S3Bucket -BucketName bucketname
  ```

- Amazon EC2 cmdlet.

  ```
  Install-Module AWS.Tools.EC2 -Force
  Get-EC2InstanceStatus -InstanceId instanceId
  ```

- 공통 또는 서비스 독립적 AWS Tools for Windows PowerShell cmdlet입니다.

```
Install-Module AWS.Tools.Common -Force
Get-AWSRegion
```

스크립트에서 PowerShell Core cmdlet을 사용하는 것 외에 새 객체를 초기화하는 경우 다음 명령과 같이 모듈도 가져와야 합니다.

```
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
```

`AWS.Tools` 모듈 설치와 가져오기 및 실행서의 PowerShell Core cmdlet 사용 예는 문서 빌더를 사용하여 런북 생성 섹션을 참조하세요.

Input

스크립트를 실행하는 데 필요한 정보를 입력합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

> **Note**
>
> Python 스크립트의 첨부 파일은 스크립트가 포함된 .py 파일 또는 .zip 파일일 수 있습니다. PowerShell 스크립트는.zip 파일에 저장해야 합니다.

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"
    }
}
```

런타임

제공된 스크립트를 실행하는 데 사용할 런타임 언어. `aws:executeScript`에서는 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) 스크립트를 지원합니다.

지원되는 값: **`python3.7`** | **`python3.8`** | **`python3.9`** | **`python3.10`** | **`python3.11`** | **`PowerShell Core 6.0`** | **`PowerShell 7.0`**

타입: 문자열

필수 항목 여부: 예

핸들러

함수의 이름입니다. 핸들러에 정의된 함수에 두 개의 파라미터 `events`과 `context`가 있는지 확인해야 합니다. PowerShell 런타임은 이 파라미터를 지원하지 않습니다.

타입: 문자열

필수: 예(Python) | 지원되지 않음(PowerShell)

InputPayload

핸들러의 첫 번째 파라미터로 전달되는 JSON 또는 YAML 객체입니다. 이 스크립트에 입력 데이터를 전달하는 데 사용할 수 있습니다.

타입: 문자열

필수 항목 여부: 아니요

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: "python3.8"
        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": "Example"
            }
            ec2.create_tags(
```

```
                Resources=[instanceId],
                Tags=[tag]
            )
```

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'
    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 = "Example"

          New-EC2Tag -Resource $input.instanceId -Tag $tag
```

Script

자동화 중 실행할 기본 제공 스크립트입니다.

타입: 문자열

필수: 아니오(Python) | 예(PowerShell)

연결

작업에서 호출될 수 있는 독립형 스크립트 파일 또는.zip 파일의 이름입니다. `Attachments` 요청 파라미터에서 지정한 문서 첨부 파일의 `Name`과 동일한 값을 지정합니다. 자세한 내용은 AWS Systems Manager API 참조의 첨부 파일을 참조하세요. 첨부 파일을 사용하여 스크립트를 제공하는 경우 런북의 최상위 수준 요소에서 `files` 섹션 또한 정의해야 합니다. 자세한 내용은 스키마 버전 0.3 단원을 참조하십시오.

Python용 파일을 호출하려면 `Handler`에서 `filename.method_name` 형식을 사용합니다.

> **Note**
>
> Python 스크립트의 첨부 파일은 스크립트가 포함된 .py 파일 또는 .zip 파일일 수 있습니다. PowerShell 스크립트는.zip 파일에 저장해야 합니다.

첨부 파일에 Python 라이브러리를 포함할 때 각 모듈 디렉터리에 빈 `__init__.py` 파일을 추가하는 것이 좋습니다. 이를 통해 스크립트 내용 내 첨부 파일의 라이브러리에서 모듈을 가져올 수 있습니다. 예: `from library import module`

타입: 문자열

필수 항목 여부: 아니요

출력

페이로드

함수에서 반환된 객체의 JSON 표현입니다. 최대 100KB가 반환됩니다. 목록을 출력하는 경우 최대 100개의 항목이 반환됩니다.

`aws:executeStateMachine` - AWS Step Functions 상태 시스템을 실행합니다.

AWS Step Functions 상태 시스템을 실행합니다.

입력

이 작업은 Step Functions StartExecution API 작업에서 대부분의 파라미터를 지원합니다.

필요한 AWS Identity and Access Management(IAM) 권한

- `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 상태 시스템의 Amazon 리소스 이름(ARN)입니다.

타입: 문자열

필수 항목 여부: 예

이름

실행의 이름입니다.

타입: 문자열

필수 항목 여부: 아니요

입력

실행을 위한 JSON 입력 문자를 포함하는 문자열입니다.

타입: 문자열

필수 항목 여부: 아니요

결과

이 작업의 경우 다음과 같은 출력이 사전 정의되어 있습니다.

executionArn

실행의 ARN입니다.

타입: 문자열

입력

실행의 JSON 입력 데이터를 포함하는 문자열입니다. 길이 제한은 페이로드 크기에 적용되며 UTF-8 인코딩에서 바이트로 표시됩니다.

타입: 문자열

이름

실행의 이름입니다.

타입: 문자열

output

실행의 JSON 출력 데이터입니다. 길이 제한은 페이로드 크기에 적용되며 UTF-8 인코딩에서 바이트로 표시됩니다.

타입: 문자열

startDate

실행이 시작된 날짜입니다.

타입: 문자열

stateMachineArn

실행된 상태 시스템의 ARN입니다.

타입: 문자열

status

실행의 현재 상태입니다.

타입: 문자열

stopDate

실행이 이미 종료된 경우, 실행이 중지된 날짜입니다.

타입: 문자열

`aws:invokeWebhook` - Automation 웹후크 통합 호출

지정된 Automation 웹후크 통합을 호출합니다. Automation 통합 생성에 대한 자세한 내용은 Automation을 위한 웹후크 통합 생성 섹션을 참조하세요.

> **Note**
>
> `aws:invokeWebhook` 작업을 사용하려면 사용자 또는 서비스 역할이 다음 작업을 허용해야 합니다.
>
> - ssm:GetParameter
> - kms:Decrypt
>
> AWS Key Management Service(AWS KMS) `Decrypt` 작업에 대한 권한은 고객 관리형 키를 사용하여 통합을 위한 파라미터를 암호화하는 경우에만 필요합니다.

Input

호출하려는 Automation 통합에 대한 정보를 제공합니다.

YAML

```
action: "aws:invokeWebhook"
inputs:
 IntegrationName: "exampleIntegration"
```

```
 Body: "Request body"
```

JSON

```
{
    "action": "aws:invokeWebhook",
    "inputs": {
        "IntegrationName": "exampleIntegration",
        "Body": "Request body"
    }
}
```

IntegrationName

Automation 통합의 이름입니다. 예를 들면 exampleIntegration입니다. 지정하는 통합은 이미 존재해야 합니다.

타입: 문자열

필수 항목 여부: 예

본문

웹후크 통합이 호출될 때 전송하려는 페이로드입니다.

타입: 문자열

필수 항목 여부: 아니요

출력

응답

웹후크 공급자 응답에서 받은 텍스트입니다.

ResponseCode

웹후크 공급자 응답에서 받은 HTTP 상태 코드입니다.

aws:invokeLambdaFunction - AWS Lambda 함수 호출

지정된 AWS Lambda 함수를 호출합니다.

> **Note**
>
> 각 `aws:invokeLambdaFunction` 작업은 최대 300초(5분)까지 실행할 수 있습니다. `aws:invokeLambdaFunction` 단계에서 `timeoutSeconds` 파라미터를 지정하여 시간 제한을 제한할 수 있습니다.

Input

이 작업은 Lambda 서비스에 대한 대부분의 호출된 파라미터를 지원합니다. 자세한 내용은 Invoke를 참조하십시오.

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 함수의 이름입니다. 이 함수가 존재해야 합니다.

타입: 문자열

필수 항목 여부: 예

Qualifier

함수 버전 또는 별칭 이름.

타입: 문자열

필수 항목 여부: 아니요

InvocationType

호출 유형. 기본 값은 RequestResponse입니다.

타입: 문자열

유효한 값: Event | RequestResponse | DryRun

필수 항목 여부: 아니요

LogType

기본값이 Tail인 경우 호출 유형은 RequestResponse여야 합니다. Lambda는 base64로 인코딩된 Lambda 함수에 의해 생성된 로그 데이터의 마지막 4KB를 반환합니다.

타입: 문자열

유효한 값: None | Tail

필수 항목 여부: 아니요

ClientContext

클라이언트 지정 정보.

필수 항목 여부: 아니요

InputPayload

핸들러의 첫 번째 파라미터로 전달되는 YAML 또는 JSON 객체입니다. 이 입력을 사용하여 함수에 데이터를 전달할 수 있습니다. 이 입력은 기존 Payload 입력보다 더 많은 유연성과 지원을 제공합니다. 작업에 대해 InputPayload와 Payload를 모두 정의하는 경우 InputPayload가 우선하며 Payload 값은 사용되지 않습니다.

유형: StringMap

필수 항목 여부: 아니요

페이로드

핸들러의 첫 번째 파라미터에 전달되는 JSON 문자열입니다. 함수에 입력 데이터를 전달하는 데 사용할 수 있습니다. 추가된 기능에는 InputPayload 입력을 사용하는 것이 좋습니다.

타입: 문자열

필수 항목 여부: 아니요

출력

StatusCode

HTTP 상태 코드.

FunctionError

존재하는 경우 함수 실행 중에 오류가 발생했음을 나타냅니다. 오류에 대한 세부 정보가 응답 페이로드에 포함됩니다.

LogResult

Lambda 함수 호출에 대한 base64 인코딩 로그입니다. 로그는 호출 유형이 RequestResponse이고 로그가 요청된 경우에만 표시됩니다.

페이로드

Lambda 함수에서 반환된 객체의 JSON 표현입니다. 호출 유형이 RequestResponse인 경우에만 페이로드가 있습니다. 최대 200KB가 반환됩니다.

다음은 aws:invokeLambdaFunction 작업의 출력을 참조하는 방법을 보여주는 AWS-PatchInstanceWithRollback 실행서의 일부입니다.

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- 자동화의 여러 단계를 반복

이 작업은 Automation 런북의 일부 하위 단계를 반복합니다. `do while` 또는 `for each` 스타일 루프를 선택할 수 있습니다. `do while` 루프를 구성하려면 `LoopCondition` 입력 파라미터를 사용합니다. `for each` 루프를 구성하려면 `Iterators` 및 `IteratorDataType` 입력 파라미터를 사용합니다. `aws:loop` 작업을 사용할 때는 `Iterators` 또는 `LoopCondition` 입력 파라미터 중 하나만 지정하십시오. 최대 반복 횟수는 100회입니다.

`onCancel` 속성은 루프 내에 정의된 단계에 대해서만 정의할 수 있습니다. 해당 `onCancel` 속성은 `aws:loop` 작업에 지원되지 않습니다.

예제

다음은 다양한 유형의 루프 작업을 구성하는 방법의 예시입니다.

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
```

Input

입력은 다음과 같습니다.

반복자

반복할 단계의 항목 목록입니다. 최대 반복자 수는 100입니다.

유형: StringList

필수 항목 여부: 아니요

IteratorDataType

`Iterators`의 데이터 유형을 지정하는 선택적 파라미터입니다. 이 파라미터의 값은 `Iterators` 입력 파라미터와 함께 제공될 수 있습니다. 이 파라미터와 `Iterators` 값을 지정하지 않으면 `LoopCondition` 파라미터의 값을 지정해야 합니다.

타입: 문자열

유효한 값: Boolean | Integer | String | StringMap

기본값: 문자열

필수 항목 여부: 아니요

LoopCondition

평가할 `Variable` 및 연산자 조건으로 구성됩니다. 이 파라미터의 값을 지정하지 않으면 `Iterators` 및 `IteratorDataType` 파라미터의 값을 지정해야 합니다. `And`, `Not`, `Or` 연산자를 조합하여 복잡한 연산자 평가를 사용할 수 있습니다. 조건은 루프의 단계가 완료된 후에 평가됩니다. 조건이 `true`에 해당하고 `MaxIterations` 값에 도달하지 않은 경우 루프의 단계가 다시 실행됩니다. 연산자 조건은 다음과 같습니다.

문자열 연산

- StringEquals
- EqualsIgnoreCase
- StartsWith
- EndsWith
- 포함

수치 연산

- NumericEquals
- NumericGreater

- NumericLesser
- NumericGreaterOrEquals
- NumericLesser
- NumericLesserOrEquals

부울 연산

- BooleanEquals

유형: StringMap

필수 항목 여부: 아니요

MaxIterations

루프의 최대 단계 실행 횟수입니다. 이 입력값에 지정된 값에 도달하면 `LoopCondition` 항목이 여전히 `true`에 해당하거나 `Iterators` 파라미터에 객체가 남아 있더라도 루프 실행이 중지됩니다.

유형: 정수

유효한 값은 1~100입니다.

필수 항목 여부: 아니요

단계

루프에서 실행할 단계 목록입니다. 이들은 중첩된 런북처럼 작동합니다. 이 단계에서 `{{loopStepName.CurrentIteratorValue}}` 구문을 사용하여 `for each` 루프의 현재 반복자 값에 액세스할 수 있습니다. `{{loopStepName.CurrentIteration}}` 구문을 사용하여 두 루프 유형에 대한 현재 반복의 정수 값에 액세스할 수도 있습니다.

유형: 단계 목록

필수 여부: 예

출력

CurrentIteration

현재 루프 반복을 정수로 나타낸 값입니다. 반복 값은 1에서 시작합니다.

유형: 정수

CurrentIteratorValue

문자열인 현재 반복자 값입니다. 이 출력은 `for each` 루프에만 표시됩니다.

타입: 문자열

`aws:pause` - 자동화 일시 중지

이 작업은 자동화를 일시 중지합니다. 자동화가 일시 중지되면 상태가 [대기 중(Waiting)]이 됩니다. 자동화를 계속하려면 신호 유형이 `Resume`인 SendAutomationSignal API 작업을 사용합니다. 작업 워크플로를 보다 세밀하게 제어할 수 있는 `aws:sleep` 또는`aws:approve` 작업을 사용하는 것이 좋습니다.

Input

입력은 다음과 같습니다.

YAML

```
name: pauseThis
action: aws:pause
inputs: {}
```

JSON

```
{
    "name": "pauseThis",
    "action": "aws:pause",
    "inputs": {}
}
```

출력

None

`aws:runCommand` - 관리형 인스턴스에서 명령 실행

지정된 명령을 실행합니다.

> **Note**
>
> Automation은 하나의 AWS Systems Manager Run Command 작업의 출력만 지원합니다. 런북에는 Run Command 작업이 여러 개 포함될 수 있지만 출력은 한 번에 한 작업에서만 지원됩니다.

Input

이 작업은 대부분의 send-command 파라미터를 지원합니다. 자세한 내용은 SendCommand를 참조하십시오.

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

명령 유형 문서가 사용자 또는 AWS의 소유인 경우 문서의 이름을 지정합니다. 다른 AWS 계정이 공유한 문서를 사용하는 경우 문서의 Amazon 리소스 이름(ARN)을 지정합니다. 공유 문서 사용에 대한 자세한 내용은 공유 SSM 문서 사용 섹션을 참조하세요.

타입: 문자열

필수 항목 여부: 예

InstanceIds

명령을 실행할 인스턴스 ID입니다. 최대 50개 ID를 지정할 수 있습니다.

또한 인스턴스 ID 대신 의사 파라미터 {{RESOURCE_ID}}를 사용하여 대상 그룹의 모든 인스턴스에서 명령을 실행할 수 있습니다. 가상 파라미터에 대한 자세한 내용은 유지 관리 기간 작업 등록 시 의사 파라미터 사용 단원을 참조하십시오.

또 다른 대안은 Targets 파라미터를 사용하여 인스턴스 플릿에 명령을 보내는 것입니다. Targets 파라미터는 Amazon Elastic Compute Cloud(Amazon EC2) 태그를 허용합니다. Targets 파라미터 사용 방법에 대한 자세한 내용은 대규모로 명령 실행 섹션을 참조하세요.

유형: StringList

필수 여부: 아니요(InstanceId를 지정하지 않거나 {{RESOURCE_ID}} 의사 파라미터를 사용하지 않는 경우 Targets 파라미터를 지정해야 함)

대상

지정한 키, 값 조합을 사용하여 인스턴스를 대상으로 하는 검색 기준의 배열입니다. Targets는 호출에 하나 이상의 인스턴스 ID를 제공하지 않는 경우 필요합니다. Targets 파라미터 사용 방법에 대한 자세한 내용은 대규모로 명령 실행 섹션을 참조하세요.

형식: MapList(목록의 맵 스키마가 객체와 일치해야 합니다.) 자세한 내용은 AWS Systems Manager API Reference의 Target을 참조하세요.

필수 여부: 아니요(Targets를 지정하지 않는 경우 InstanceId를 지정하거나 {{RESOURCE_ID}} 의사 파라미터를 사용해야 함)

다음은 한 예입니다.

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"
            ]
        }
    }
}
```

파라미터

문서에서 지정된 필수 및 선택 파라미터.

유형: 맵

필수 항목 여부: 아니요

CloudWatchOutputConfig

명령 출력을 Amazon CloudWatch Logs로 전송하기 위한 구성 옵션입니다. CloudWatch Logs로 명령 출력 전송에 대한 자세한 내용은 Run Command에 대한 Amazon CloudWatch Logs 구성 섹션을 참조하세요.

유형: StringMap(지도의 스키마는 객체와 일치해야 합니다. 자세한 내용은 AWS Systems Manager API Reference의 CloudWatchOutputConfig를 참조하세요.)

필수 항목 여부: 아니요

다음은 한 예입니다.

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
        }
    }
}
```

설명

명령에 관한 사용자 정의 정보.

타입: 문자열

필수 항목 여부: 아니요

DocumentHash

문서에 대한 해시.

타입: 문자열

필수 항목 여부: 아니요

DocumentHashType

해시의 유형.

타입: 문자열

유효한 값: Sha256 | Sha1

필수 항목 여부: 아니요

NotificationConfig

알림 전송에 대한 구성.

필수 항목 여부: 아니요

OutputS3BucketName

명령 출력 응답에 대한 S3 버킷의 이름.

타입: 문자열

필수 항목 여부: 아니요

OutputS3KeyPrefix

접두사입니다.

타입: 문자열

필수 항목 여부: 아니요

ServiceRoleArn

AWS Identity and Access Management(IAM) 역할의 ARN.

타입: 문자열

필수 항목 여부: 아니요

TimeoutSeconds

명령이 인스턴스의 AWS Systems Manager SSM Agent에 전송될 때까지 대기하는 시간(초)입니다. 지정된 값에 도달하기 전에 인스턴스의 SSM Agent에서 명령을 수신하지 않으면 명령 상태가 `Delivery Timed Out`으로 변경됩니다.

유형: 정수

필수 항목 여부: 아니요

유효한 값: 30~2592000

출력

CommandId

명령의 ID.

상태 표시기

명령의 상태.

ResponseCode

명령의 응답 코드입니다. 실행하는 문서의 단계가 두 개 이상인 경우 이 출력에는 값이 반환되지 않습니다.

출력

명령의 출력입니다. 명령으로 태그 또는 여러 인스턴스를 대상으로 지정하는 경우 출력 값이 반환되지 않습니다. `GetCommandInvocation` 및 `ListCommandInvocations` API 작업을 사용하여 개별 인스턴스의 출력을 검색할 수 있습니다.

aws:runInstances - Amazon EC2 인스턴스 시작

새로운 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스를 시작합니다.

Input

이 작업은 대다수 API 파라미터를 지원합니다. 자세한 내용은 RunInstances API 문서를 참조하십시오.

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

예약.

타입: 문자열

필수 항목 여부: 아니요

BlockDeviceMappings

해당 인스턴스용 블록 디바이스.

형식: MapList

필수 항목 여부: 아니요

ClientToken

요청의 멱등성을 보장하는 식별자.

타입: 문자열

필수 항목 여부: 아니요

DisableApiTermination

인스턴스 API 종료를 설정하거나 해제합니다.

타입: 부울

필수 항목 여부: 아니요

EbsOptimized

Amazon Elastic Block Store(Amazon EBS) 최적화를 설정하거나 해제합니다.

타입: 부울

필수 항목 여부: 아니요

IamInstanceProfileArn

인스턴스에 대한 AWS Identity and Access Management(IAM) 인스턴스 프로파일의 Amazon 리소스 이름(ARN)입니다.

타입: 문자열

필수 항목 여부: 아니요

IamInstanceProfileName

인스턴스용 IAM 인스턴스 프로파일의 이름입니다.

타입: 문자열

필수 항목 여부: 아니요

ImageId

Amazon Machine Image(AMI)의 ID입니다.

타입: 문자열

필수 항목 여부: 예

InstanceInitiatedShutdownBehavior

시스템 종료 시 인스턴스를 중지할지 종료할지 나타냅니다.

타입: 문자열

필수 항목 여부: 아니요

InstanceType

인스턴스 유형.

> **Note**
> 인스턴스 유형 값이 제공되지 않은 경우에는 m1 스몰 인스턴스 유형이 사용됩니다.

타입: 문자열

필수 항목 여부: 아니요

KernelId

커널 ID.

타입: 문자열

필수 항목 여부: 아니요

KeyName

키 페어 이름.

타입: 문자열

필수 항목 여부: 아니요

MaxInstanceCount

시작되는 인스턴스의 최대 수.

타입: 문자열

필수 항목 여부: 아니요

MetadataOptions

인스턴스에 대한 메타데이터 옵션입니다. 자세한 내용은 InstanceMetadataOptionsRequest를 참조하세요.

유형: StringMap

필수 항목 여부: 아니요

MinInstanceCount

시작되는 인스턴스의 최소 수.

타입: 문자열

필수 항목 여부: 아니요

모니터링

세부 모니터링을 설정하거나 해제합니다.

타입: 부울

필수 항목 여부: 아니요

NetworkInterfaces

네트워크 인터페이스.

형식: MapList

필수 항목 여부: 아니요

Placement

인스턴스 배치.

유형: StringMap

필수 항목 여부: 아니요

PrivateIpAddress

기본 IPv4 주소.

타입: 문자열

필수 항목 여부: 아니요

RamdiskId

RAM 디스크 ID.

타입: 문자열

필수 항목 여부: 아니요

SecurityGroupIds

인스턴스에 대한 보안 그룹의 ID.

유형: StringList

필수 항목 여부: 아니요

SecurityGroups

인스턴스에 대한 보안 그룹의 이름.

유형: StringList

필수 항목 여부: 아니요

SubnetId

서브넷 ID.

타입: 문자열

필수 항목 여부: 아니요

TagSpecifications

시작 시 리소스에 적용되는 태그. 시작 시에만 인스턴스와 볼륨에 태그를 지정할 수 있습니다. 지정된 태그는 시작 시 생성된 모든 인스턴스 또는 볼륨에 적용됩니다. 시작된 후 인스턴스에 태그를 지정하려면 `aws:createTags` - AWS 리소스에 대한 태그 생성 작업을 사용합니다.

형식: MapList. 자세한 내용은 TagSpecification을 참조하십시오.

필수 항목 여부: 아니요

UserData

문자열 리터럴 값으로 제공된 스크립트입니다. 리터럴 값이 입력되면 Base64로 인코딩되어야 합니다.

타입: 문자열

필수 항목 여부: 아니요

출력

InstanceIds

인스턴스의 ID.

InstanceStates

인스턴스의 현재 상태입니다.

`aws:sleep` - 자동화 지연

지정된 시간 동안 자동화를 지연시킵니다. 이 작업은 ISO(국제 표준화 기구) 8601 날짜 및 시간 형식을 사용합니다. 이 날짜 및 시간 형식에 대한 자세한 내용은 ISO 8601을 참조하십시오.

Input

지정된 기간 동안 자동화를 지연시킬 수 있습니다.

YAML

```
name: sleep
action: aws:sleep
inputs:
  Duration: PT10M
```

JSON

```
{
   "name":"sleep",
   "action":"aws:sleep",
   "inputs":{
      "Duration":"PT10M"
   }
```

```
}
```

또한 지정된 날짜와 시간까지 자동화를 지연시킬 수 있습니다. 지정된 날짜와 시간이 경과하면 작업이 즉시 진행됩니다.

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**
>
> Automation은 최대 지연을 604,799초(7일)까지 지원합니다.

지속 시간

ISO 8601 기간. 음의 기간을 지정할 수 없습니다.

타입: 문자열

필수 항목 여부: 아니요

Timestamp

ISO 8601 타임스탬프. 이 파라미터의 값을 지정하지 않으면 `Duration` 파라미터의 값을 지정해야 합니다.

타입: 문자열

필수 항목 여부: 아니요

출력

None

`aws:updateVariable` - 런북 변수 값을 업데이트

이 작업은 런북 변수 값을 업데이트합니다. 값의 데이터 유형은 업데이트하려는 변수의 데이터 유형과 일치해야 합니다. 데이터 유형 변환은 지원되지 않습니다. 해당 `onCancel` 속성은 `aws:updateVariable` 작업에 지원되지 않습니다.

Input

입력은 다음과 같습니다.

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"]
    }
}
```

명칭

값을 업데이트할 변수의 이름입니다. `variable:variable name` 형식을 사용해야 합니다.

타입: 문자열

필수 항목 여부: 예

값

변수에 할당할 새 값입니다. 값은 변수의 데이터 유형과 일치해야 합니다. 데이터 유형 변환은 지원되지 않습니다.

형식: Boolean | Integer | MapList | String | StringList | StringMap

필수 항목 여부: 예

제약 조건:

- MapList는 최대 200개의 항목을 포함할 수 있습니다.
- 키 길이는 최소 1자이고 최대 길이는 50자입니다.
- StringList는 최소 0개 항목에서 최대 50개 항목일 수 있습니다.
- 문자열의 최소 길이는 1자이고 최대 길이는 512자입니다.

출력

None

`aws:waitForAwsResourceProperty` - AWS 리소스 속성 대기

`aws:waitForAwsResourceProperty` 작업을 사용하면 자동화를 계속 진행하기 전에 자동화에서 특정 리소스 상태나 이벤트 상태를 대기할 수 있습니다. 이 작업을 사용하는 방법에 관한 자세한 내용은 추가 런북 예제 섹션을 참조하세요.

> **Note**
>
> 이 작업의 기본 제한 시간 값은 3600초(1시간)입니다. `aws:waitForAwsResourceProperty` 단계에서 `timeoutSeconds` 파라미터를 지정하여 제한 시간을 제한 또는 연장할 수 있습니다. 이 작업을 사용하는 방법에 관한 자세한 내용과 예제는 실행서에서 시간 제한 처리 섹션을 참조하세요.

Input

입력은 선택하는 API 작업에 의해 정의됩니다.

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"
    ]
  }
}
```

Service

실행하려는 API 작업을 포함하는 AWS 서비스 네임스페이스입니다. 예를 들면 AWS Systems Manager의 네임스페이스는 ssm입니다. Amazon Elastic Compute Cloud(Amazon EC2)의 네임스페이스는 ec2입니다. AWS CLI 명령 참조의 사용 가능한 서비스 섹션에서 지원되는 AWS 서비스 네임스페이스 목록을 볼 수 있습니다.

타입: 문자열

필수 항목 여부: 예

Api

실행할 API 작업의 이름입니다. 다음 [서비스 참조(Services Reference)] 페이지의 왼쪽 탐색 영역에서 서비스를 선택하여 API 작업(메서드라고도 함)을 볼 수 있습니다. 호출할 서비스에 대한 클라이언트 섹션에서 메서드를 선택합니다. 예를 들어 Amazon Relational Database Service(Amazon RDS)에 대한 모든 API 작업(메서드)이 Amazon RDS 메서드 페이지에 나열됩니다.

타입: 문자열

필수 항목 여부: 예

API 작업 입력

하나 이상의 API 작업 입력입니다. 다음 서비스 참조 페이지의 왼쪽 탐색 영역에서 서비스를 선택하여 사용할 수 있는 입력(파라미터)을 볼 수 있습니다. 호출할 서비스에 대한 클라이언트 섹션에서 메서드를 선택합니다. 예를 들어 Amazon RDS에 대한 모든 메서드는 Amazon RDS 메서드 페이지에 나열됩니다. describe_db_instances 메서드를 선택하고 아래로 스크롤하여 DBInstanceIdentifier, Name 및 Values 같은 사용할 수 있는 파라미터를 볼 수 있습니다.

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"
}
```

형식: 선택한 API 작업에 의해 결정됨

필수 항목 여부: 예

PropertySelector

응답 객체의 특정 속성에 대한 JSONPath입니다. 다음 서비스 참조 페이지의 왼쪽 탐색 영역에서 서비스를 선택하여 응답 객체를 볼 수 있습니다. 호출할 서비스에 대한 클라이언트 섹션에서 메서드를 선택합니다. 예를 들어 Amazon RDS에 대한 모든 메서드는 Amazon RDS 메서드 페이지에 나열됩니다. describe_db_instances 메서드를 선택하고 Response Structure(응답 구조) 섹션이 있는 아래쪽으로 스크롤합니다. DBInstances가 응답 객체로 나열됩니다.

타입: 문자열

필수 항목 여부: 예

DesiredValues

필요한 상태 또는 자동화를 계속 진행하기 위해 충족되어야 할 상태입니다.

유형: MapList, StringList

필수 항목 여부: 예

Automation 시스템 변수

AWS Systems Manager Automation 실행서는 다음 변수를 사용합니다. 이러한 변수를 사용하는 방법의 예는 `AWS-UpdateWindowsAmi` 실행서의 JSON 원본을 참조하세요.

`AWS-UpdateWindowsAmi` 실행서의 JSON 소스를 보려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. 문서 목록에서 검색 창을 사용하거나 검색 창 오른쪽의 숫자를 사용하여 실행서 **`AWS-UpdateWindowsAmi`**를 선택합니다.
4. 콘텐츠 탭을 선택합니다.

시스템 변수

Automation 실행서는 다음 시스템 변수를 지원합니다.

변수	Details
`global:ACCOUNT_ID`	Automation이 실행되는 사용자 또는 역할의 AWS 계정 ID입니다.
`global:DATE`	yyyy-MM-dd 형식의 날짜(런타임 시).
`global:DATE_TIME`	yyyy-MM-dd_HH.mm.ss 형식의 날짜와 시간(런타임 시).
`global:AWS_PARTITION`	리소스가 있는 파티션. 표준 AWS 리전에서 파티션은 `aws`입니다. 다른 파티션에 있는 리소스의 경우 파티션은 `aws-`*`partitionname`* 입니다. 예를 들어 AWS GovCloud(미국 서부) 리전의 리소스 파티션은 `aws-us-gov` 입니다.
`global:REGION`	실행서가 실행되는 리전입니다. 예: us-east-2.

Automation 변수

Automation 실행서는 다음 자동화 변수를 지원합니다.

변수	Details
`automation:EXECUTION_ID`	현재 자동화에 할당된 고유 식별자입니다. 예를 들면 `1a2b3c-1a2b3c-1a2b3c-1a2b3c 1a2b3c1a2b3c` 입니다.

주제

- 용어
- 지원되는 시나리오
- 지원되지 않는 시나리오

용어

다음 용어는 변수 및 파라미터 확인 방식을 설명합니다.

용어	정의	예
Constant ARN	변수가 없는 유효한 Amazon 리소스 이름(ARN)입니다.	`arn:aws:iam::123456789012:role/roleName`
실행서 파라미터	실행서 수준에서 정의된 파라미터(예:`instanceId`)입니다. 파라미터는 기본 문자열 대체 항목으로 사용됩니다. 값은 실행 시작 시점에 제공됩니다.	`{ "description": "Create Image Demo", "version": "0.3", "assumeRole": "Your_Automation_Assume_Role_ARN ", "parameters":{ "instanceId": { "type": "String", "description": "Instance to create image from" } }`
시스템 변수	실행서의 일부를 평가할 때 실행서에 대체 항목으로 제공되는 일반 변수입니다.	`"activities": [ { "id": "copyImage", "activityType": "AWS-CopyImage", "maxAttempts": 1, "onFailure": "Continue", "inputs": { "ImageName": "{{imageName}}", "SourceImageId": "{{sourceImageId}}", "SourceRegion": "{{sourceRegion}}",`

용어	정의	예
		"Encrypted": true, "ImageDescription": "Test CopyImage Description created on **{{global:DATE}}** " } }]

용어	정의	예
자동화 변수	실행서의 일부를 평가할 때 실행서에 대체 항목으로 제공되는 자동화 관련 변수입니다.	{ "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}}** ”] } } }

용어	정의	예
Systems Manager 파라미터	AWS Systems Manager Parameter Store에서 정의된 변수입니다. 단계 입력에서는 직접 참조할 수 없습니다. 파라미터에 액세스하려면 권한이 필요할 수 있습니다.	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

용어	정의	예
		timeoutSeconds: 1200 onFailure: Abort inputs: ImageId: '{{LatestAmi}}' ...

지원되는 시나리오

시나리오	설명	예
생성 시 상수 ARN `assumeRole` .	호출 사용자가 지정된 `assumeRole` 을 전달하도록 허용되는지 확인하는 권한 부여 확인이 수행됩니다.	{ "description": "Test all Automation resolvable parameters", "schemaVersion": "0.3", "assumeRole": ***"arn:aws:iam::123456789012:role/roleName"*** , "parameters": { ...
자동화가 시작될 때 `AssumeRole` 에 제공되는 실행서 파라미터입니다.	실행서의 파라미터 목록에서 정의되어야 합니다.	{ "description": "Test all Automation resolvable parameters", "schemaVersion": "0.3", "assumeRole": **"{{dynamicARN}}"** , "parameters": { ...

시나리오	설명	예
시작 시 실행서 파라미터에 제공된 값입니다.	고객이 파라미터에 사용할 값을 제공합니다. 시작 시 제공된 입력이 실행서의 파라미터 목록에 정의되어야 합니다.	<pre>... "parameters": { "amiId": { "type": "String", "default": "ami-12345678 ", "description": "list of commands to run as part of first step" }, ...</pre> 자동화 실행 시작에 입력되는 항목: {"amiId" : ["ami-12345678 "] }

시나리오	설명	예
실행서 내용 내에서 참조되는 Systems Manager 파라미터입니다.	변수는 고객의 계정 내에 존재하거나 공개적으로 액세스 가능한 파라미터이며 실행서의 `AssumeRole` 은 변수에 액세스할 수 있습니다. 생성 시 `AssumeRole` 에 액세스 권한이 있는지 확인이 이루어집니다. 이 파라미터는 단계 입력에서 직접 참조할 수 없습니다.	`...` `parameters:` `  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` `  timeoutSeconds: 1200` `  onFailure: Abort` `  inputs:` `  ImageId: '{{LatestAmi}}'` `...`

시나리오	설명	예
단계 정의 내에서 참조되는 시스템 변수	자동화가 시작될 때 시스템 변수가 실행서로 대체됩니다. 실행서에 입력된 값은 대체가 이루어진 시점과 관련이 있습니다. 즉, 그 사이의 단계를 실행하는 데 걸리는 시간 때문에 1단계에 입력된 시간 변수의 값은 3단계에 입력된 값과 다릅니다. 시스템 변수를 실행서의 파라미터 목록에 설정할 필요가 없습니다.	``` ... "mainSteps": [{ "name": "RunSomeCommands", "action": "aws:runCommand", "maxAttempts": 1, "onFailure": "Continue", "inputs": { "DocumentName": "AWS:RunPowerShell", "InstanceIds": ["{{LaunchInstance.InstanceIds}}"], "Parameters": { "commands" : ["echo {The time is now {{global:DATE_TIME}}}"] } } }, ... ```

시나리오	설명	예
단계 정의 내에서 참조되는 자동화 변수.	자동화 변수를 실행서의 파라미터 목록에 설정할 필요가 없습니다. 지원되는 유일한 자동화 변수는 automation:EXECUTION_ID입니다.	``` ... "mainSteps": [{ "name": "invokeLambdaFunction", "action": "aws:invokeLambdaFunction", "maxAttempts": 1, "onFailure": "Continue", "inputs": { "FunctionName": "Hello-World-LambdaFunction", "Payload" : "{ "executionId" : "{{automation:EXECUTION_ID}}" }" } } ... ```

시나리오	설명	예
다음 단계 정의에 포함된 이전 단계의 출력을 참조하십시오.	이는 파라미터 리디렉션입니다. 이전 단계의 출력은 구문 {{stepName.OutputName}} 을 이용해 참조됩니다. 고객은 이 구문을 실행서 파라미터에 사용할 수 없습니다. 이것은 참조 단계가 실행될 때 해결됩니다. 파라미터가 실행서의 파라미터에 열거되지 않습니다.	```... "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" } } ...```

지원되지 않는 시나리오

시나리오	설명	예
생성 시 `assumeRole` 에 제공되는 Systems Manager 파라미터입니다.	지원하지 않음.	`...` `{` `  "description":` ` "Test all Automation` ` resolvable parameter` `s",` `  "schemaVersion":` ` "0.3",` `  "assumeRole":` ` "{{ssm:administrato` `rRoleARN}}  ",` `  "parameters": {` `...`
Systems Manager 파라미터는 단계 입력에서 직접 참조할 수 있습니다.	생성 시 `InvalidDocumentContent` 예외를 반환합니다.	`...` `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}}'` `...`
변수 단계 정의	실행서에서 단계의 정의는 변수에 의해 구성됩니다.	`...`

시나리오	설명	예
		"mainSteps": [{ "name": "LaunchInstance", "action": "aws:runInstances", "**{{attemptModel}}** ": 1, "onFailure": "Continue", "inputs": { "ImageId": "*ami-12345678* ", "MinInstanceCount": 1, "MaxInstanceCount": 2 } ... User supplies input : { "attemptModel" : "**minAttempts** " }

시나리오	설명	예
실행서 파라미터 상호 참조	시작 시 사용자가 실행서 내 다른 파라미터에 대한 참조인 입력 파라미터를 제공합니다.	... "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} }" }, ...

시나리오	설명	예
다중 확장	실행서가 변수의 이름으로 평가되는 변수를 정의합니다. 이는 변수 구분 기호(즉 {{ }}) 내에 포함되고 해당 변수/파라미터의 값으로 확장됩니다.	... "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}}*** }}"] }

시나리오	설명	예
		`` ... Note: The customer intention here would be to run a command of "echo {Hello world}" ```

시나리오	설명	예
변수 유형이 다른 실행서 단계의 출력을 참조	사용자가 이후 단계에서 이전 실행서 단계의 출력을 참조합니다. 출력이 이후 단계 작업의 요구 사항을 충족하지 않는 변수 유형입니다.	``` ... 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 ```

시나리오	설명	예
		`outputs a "StringList" type variable.`

사용자 런북 생성

Automation 실행서는 자동화가 실행될 때 Systems Manager가 관리형 인스턴스 및 기타 AWS 리소스에 대해 수행하는 작업을 정의합니다. Automation은 AWS Systems Manager의 기능입니다. 실행서에는 순차적으로 실행되는 하나 이상의 단계가 포함되어 있습니다. 각 단계는 단일 작업을 중심으로 구축됩니다. 한 단계의 출력을 이후 단계에서 입력으로 사용할 수 있습니다.

이러한 작업과 해당 단계를 실행하는 프로세스를 자동화라고 합니다.

실행서에 지원되는 작업 유형을 사용하면 AWS 환경에서 다양한 작업을 자동화할 수 있습니다. 예를 들어 `executeScript` 작업 유형을 사용하면 실행서에 Python 또는 PowerShell 스크립트를 직접 포함할 수 있습니다. 사용자 정의 실행서를 만들 때 스크립트를 인라인으로 추가하거나 S3 버킷 또는 로컬 시스템에서 첨부할 수 있습니다. `createStack` 및 `deleteStack` 작업 유형을 사용하여 AWS CloudFormation 리소스 관리를 자동화할 수 있습니다. 또한 단계는 `executeAwsApi` 작업 유형을 사용하여 AWS 리소스 생성 또는 삭제, 다른 프로세스 시작, 알림 트리거 등 모든 AWS 서비스에서 모든 API 작업을 실행할 수 있습니다.

Automation에서 지원되는 20가지 작업 유형의 목록은 Systems Manager Automation 작업 참조 섹션을 참조하세요.

AWS Systems Manager Automation은 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스를 하나 이상 다시 시작하거나 Amazon Machine Image(AMI)를 생성하는 등 일반적인 태스크를 수행할 수 있는 단계가 사전 정의 되어 있는 몇 가지 실행서를 제공합니다. 자체 실행서를 생성하여 다른 AWS 계정와 공유하거나 모든 Automation 사용자에게 이를 공개할 수도 있습니다.

실행서는 YAML 또는 JSON을 사용하여 작성됩니다. 그러나 Systems Manager Automation 콘솔에서 [문서 빌더(Document Builder)]를 사용하면 기본 JSON 또는 YAML로 작성하지 않고도 실행서를 생성할 수 있습니다.

> **Important**
>
> AWS Identity and Access Management(IAM) 서비스 역할을 사용하여 다른 서비스를 호출하는 자동화 워크플로를 실행하는 경우 해당 서비스 역할이 다른 서비스를 호출할 권한이 있도록 구성되어야 합니다. 이 요구 사항은 `AWS-ConfigureS3BucketLogging`, `AWS-CreateDynamoDBBackup`, `AWS-RestartEC2Instance` 실행서 등의 모든 AWS Automation 실행서(`AWS-*` 실행서)에 적용됩니다. 또한 다른 서비스를 호출하는 작업을 사용하여 다른 AWS 서비스을(를) 호출하는 사용자 정의 Automation 실행서를 생성하는 경우에도 이 요구 사항이 항상 적용됩니다. 예를 들어 `aws:executeAwsApi`, `aws:createStack` 또는 `aws:copyImage` 작업을 사용하는 경우 이러한 서비스를 호출할 수 있는 권한을 포함하여 서비스 역할을 구성합니다. 역할에 IAM 인라인 정책을 추가하여 다른 AWS 서비스에 대한 권한을 부여할 수 있습니다. 자세한 내용은 (선택 사항) 다른 AWS 서비스를 간접적으로 호출할 Automation 인라인 정책과 고객 관리형 정책 추가 단원을 참조하십시오.

실행서에서 지정할 수 있는 작업에 대한 자세한 내용은 Systems Manager Automation 작업 참조 섹션을 참조하세요.

AWS Toolkit for Visual Studio Code를 사용하여 실행서 생성에 대한 자세한 내용은 AWS Toolkit for Visual Studio Code User Guide의 Working with Systems Manager Automation documents를 참조하세요.

비주얼 디자이너를 사용하여 사용자 정의 런북을 생성하는 방법에 대한 자세한 내용은 Automation 런북의 시각적 디자인 경험 섹션을 참조하세요.

목차

Automation 런북의 시각적 디자인 경험

AWS Systems Manager Automation은 Automation 런북을 만드는 데 도움이 되는 코드가 적은 시각적 디자인 환경을 제공합니다. 시각적 디자인 환경은 자체 코드를 추가할 수 있는 옵션이 포함된 드래그 앤 드롭 인터페이스를 제공하므로 런북을 더 쉽게 만들고 편집할 수 있습니다. 시각적 디자인 환경을 사용하여 다음 작업을 할 수 있습니다.

- 조건문 제어.
- 각 동작에 대해 입력과 출력이 필터링되거나 변환되는 방식을 제어할 수 있습니다.

- 오류 처리를 구성합니다.
- 새 런북의 프로토타입을 제작하십시오.
- AWS Toolkit for Visual Studio Code를 사용하여 프로토타입 런북을 로컬 개발의 출발점으로 활용하십시오.

런북을 만들거나 편집할 때 자동화 콘솔에서 시각적 디자인 환경에 액세스할 수 있습니다. 런북을 만들면 시각적 디자인 경험이 작업을 검증하고 코드를 자동으로 생성합니다. 생성된 코드를 검토하거나 로컬 개발을 위해 내보낼 수 있습니다. 작업을 마치면 Systems Manager Automation 콘솔에서 런북을 저장하고 실행하고 결과를 검토할 수 있습니다.

시작하기 전 준비 사항

시각적 디자인 경험을 사용하려면 사용하려는 모든 리소스에 대한 올바른 권한을 제공하는 AWS 계정 및 보안 인증이 필요합니다.

시각적 디자인 환경에서 Automation은 Amazon CodeGuru Security와 통합되어 Python 스크립트의 보안 정책 위반 및 취약성을 탐지하는 데 도움이 됩니다. 이 기능을 `aws:executeScript` 작업에 사용하려면 AWS Identity and Access Management(IAM) 정책에 다음 권한이 포함되어야 합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codeguru-security:CreateUploadUrl",
                "codeguru-security:CreateScan",
                "codeguru-security:GetScan",
                "codeguru-security:GetFindings"
            ]
        }
    ]
}
```

주제

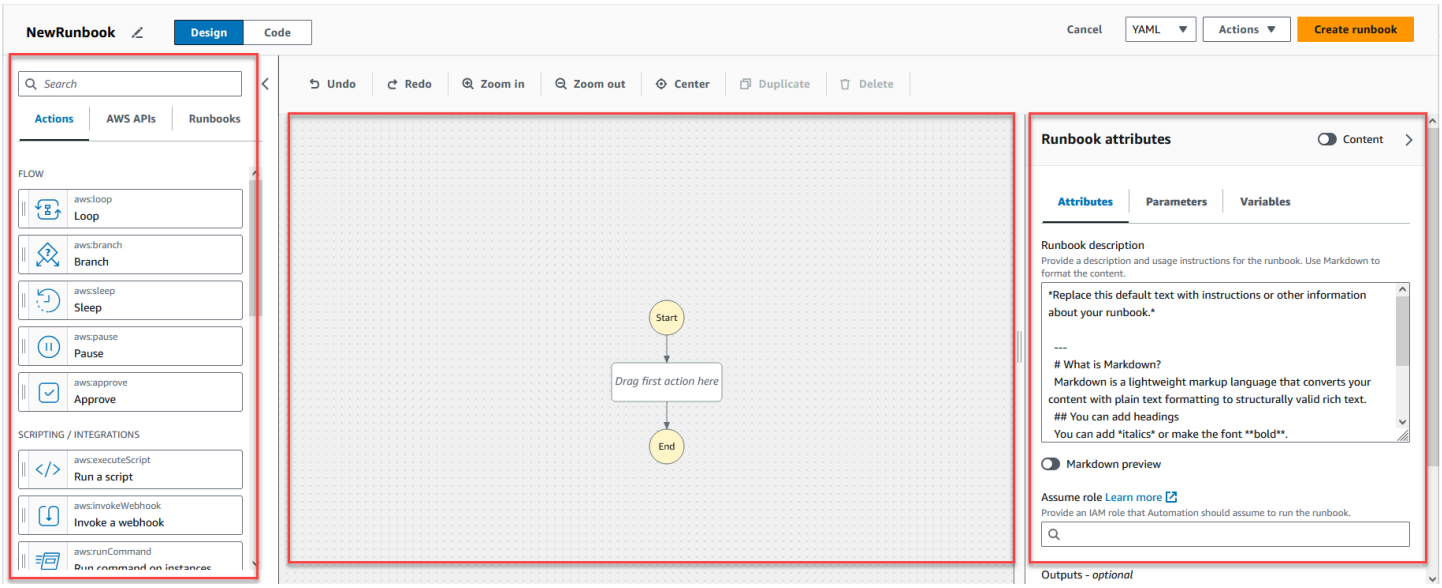
- 시각적 디자인 경험 인터페이스 개요
- 시각적 디자인 경험 사용
- 작업에 대한 입력 및 출력 구성

- 시각적 디자인 경험을 통한 오류 처리
- 자습서: 시각적 디자인 환경을 사용하여 런북을 생성합니다.

시각적 디자인 경험 인터페이스 개요

Systems Manager Automation의 시각적 디자인 환경은 Automation 런북을 만드는 데 도움이 되는 코드가 적은 시각적 워크플로 디자이너입니다.

인터페이스 구성 요소의 개요를 통해 시각적 디자인 경험에 대해 알아보십시오.



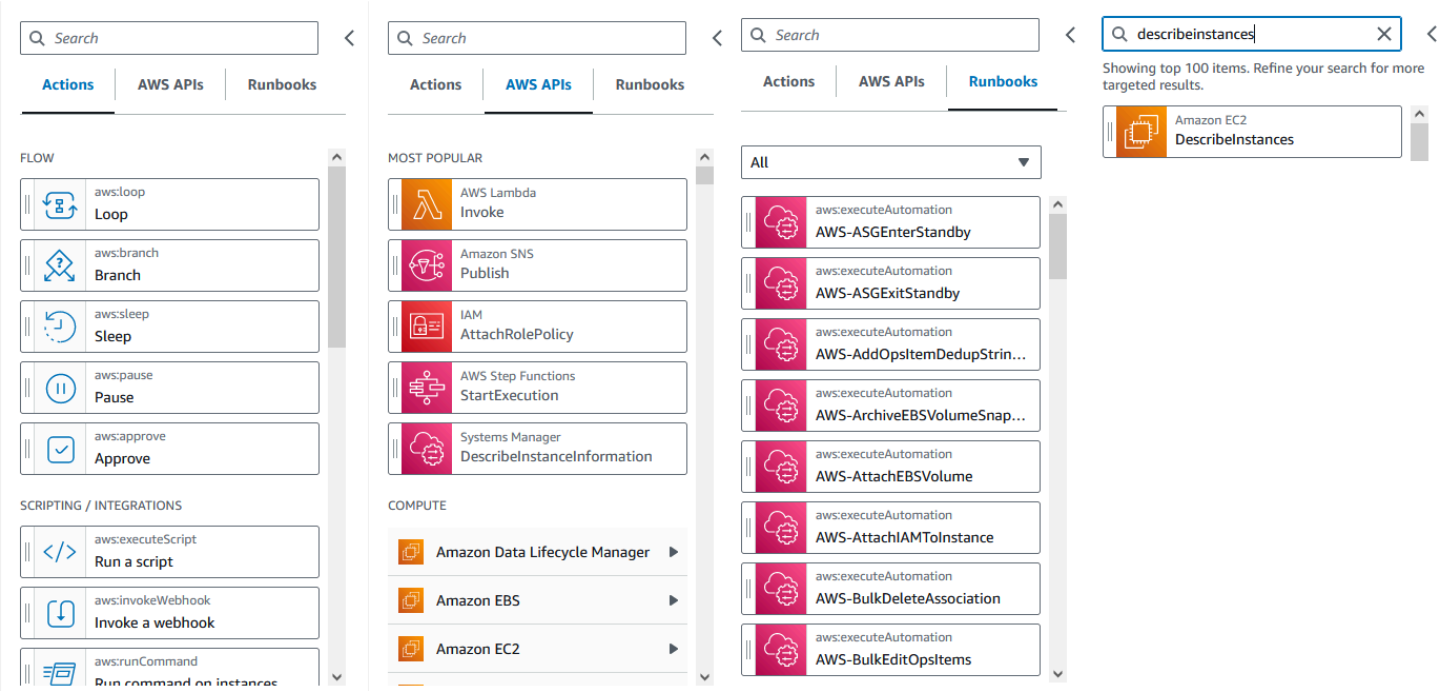
- 작업 브라우저에는 작업, AWS API 및 런북 탭이 있습니다.
- 캔버스에서는 작업을 워크플로 그래프로 끌어서 놓고, 작업 순서를 변경하고, 구성하거나 볼 작업을 선택할 수 있습니다.
- 양식 패널에서는 캔버스에서 선택한 작업의 속성을 보고 편집할 수 있습니다. 콘텐츠 토글을 선택하면 현재 선택한 작업이 강조 표시된 상태로 런북의 YAML 또는 JSON을 볼 수 있습니다.

도움이 필요할 때 정보 링크를 클릭하면 컨텍스트 정보가 포함된 패널이 열립니다. 이 패널에는 Systems Manager Automation 설명서의 관련 항목으로 연결되는 링크도 포함되어 있습니다.

작업 브라우저

작업 브라우저에서 워크플로 그래프로 드래그 앤 드롭할 작업을 선택할 수 있습니다. 작업 브라우저 상단의 검색 필드를 사용하여 모든 작업을 검색할 수 있습니다. 작업 브라우저에는 다음과 같은 탭이 있습니다.

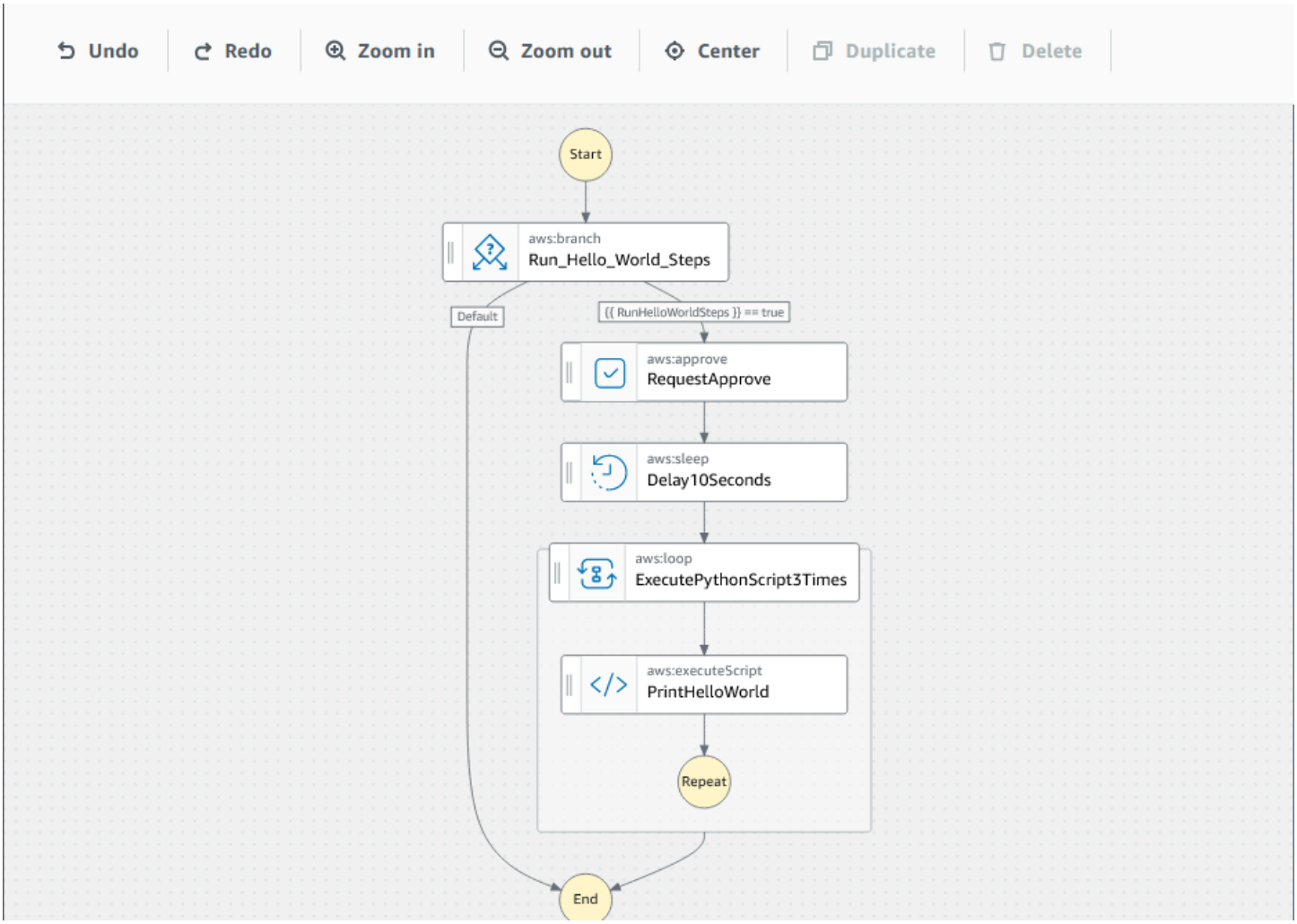
- 작업 탭은 캔버스에 있는 런북의 워크플로 그래프로 끌어서 놓을 수 있는 자동화 작업 목록을 제공합니다.
- AWS API 탭은 캔버스에 있는 런북의 워크플로 그래프로 드래그 앤 드롭할 수 있는 AWS API 목록을 제공합니다.
- 런북 탭에서는 다양한 사용 사례에 사용할 수 있는 구성 요소로 바로 사용할 수 있고 재사용이 가능한 여러 런북을 제공합니다. 예를 들어, 동일한 작업을 다시 생성할 필요 없이 런북을 사용하여 워크플로의 Amazon EC2 인스턴스에 대한 일반적인 수정 작업을 수행할 수 있습니다.



Canvas

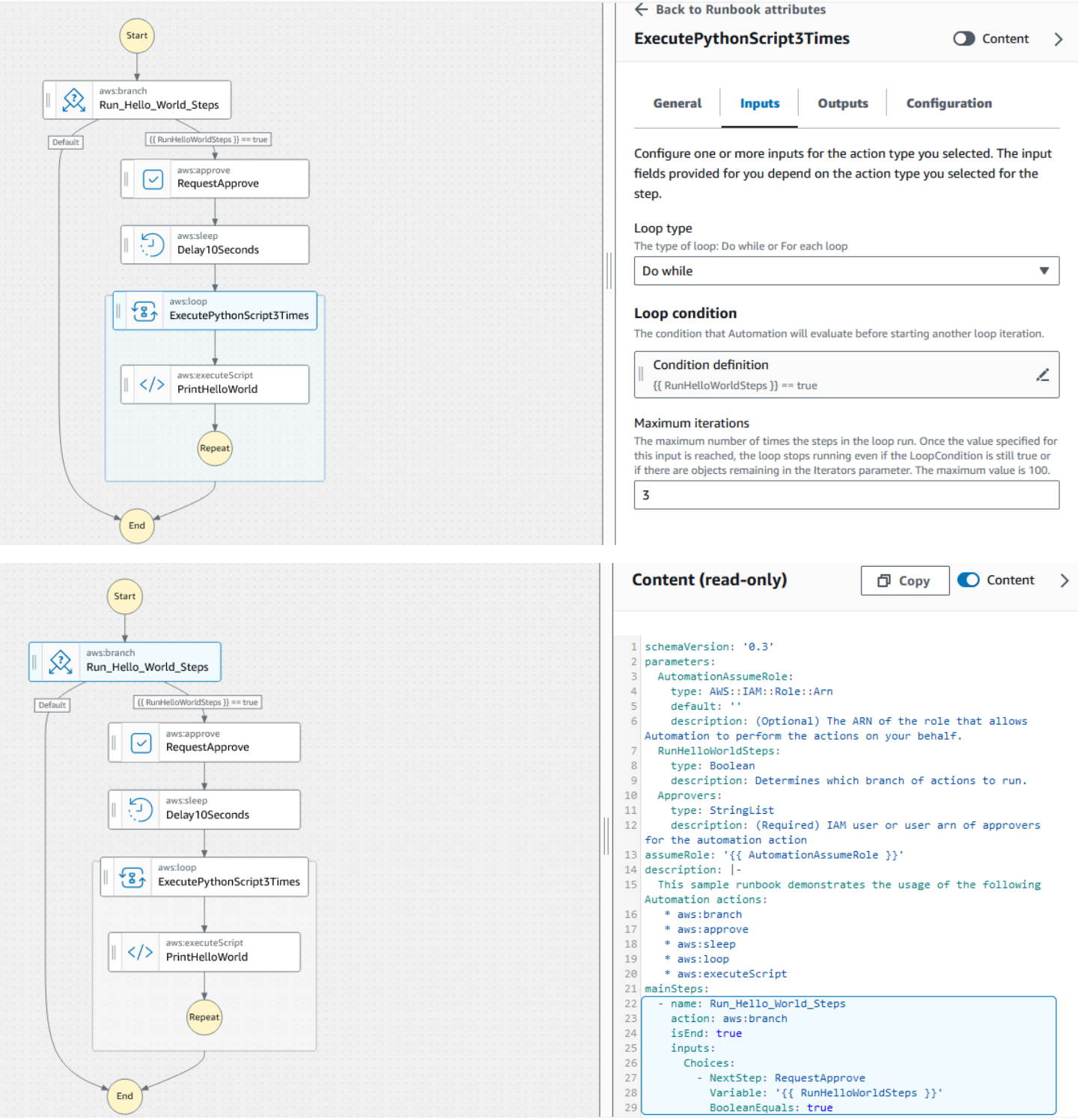
자동화에 추가할 작업을 선택한 후 캔버스로 드래그하여 워크플로 그래프에 놓습니다. 작업을 드래그 앤 드롭하여 런북 워크플로의 다른 위치로 이동시킬 수도 있습니다. 작업 과정이 복잡하면 캔버스 패널에서 전체 항목이 보이지 않을 수 있습니다. 캔버스 상단의 컨트롤을 사용하여 확대하거나 축소할 수 있습니다. 워크플로의 다른 부분을 보려면 캔버스에서 워크플로 그래프를 드래그하면 됩니다.

작업 브라우저에서 작업을 드래그하여 런북의 워크플로 그래프에 놓습니다. 선이 워크플로에서 배치될 위치를 표시해 줍니다. 작업 순서를 변경하려면 작업 순서를 워크플로의 다른 위치로 드래그할 수 있습니다. 새 작업이 워크플로에 추가되었으며 해당 코드가 자동으로 생성됩니다.



양식

런북 워크플로에 작업을 추가한 후 사용 사례에 맞게 구성할 수 있습니다. 구성하려는 작업을 선택하면 양식 패널에 해당 파라미터와 옵션이 표시됩니다. 콘텐츠 토글을 선택하여 YAML 또는 JSON 코드를 볼 수도 있습니다. 선택한 작업과 관련된 코드가 강조 표시됩니다.



키보드 바로 가기

시각적 디자인 환경은 다음 표에 표시된 키보드 단축키를 지원합니다.

환경 설정	노드 바로 가기
Ctrl + Z	막 작업의 실행을 취소합니다.
Ctrl + Shift + Z	막 작업을 다시 실행합니다.

참고

드바로가기

중요

스에서 워크플로를 중앙에 배치합니다.

취소 드바로가기

백그라운드로 전환 또는 상태를 제거합니다.

상태 한 모든 상태를 제거합니다.

참고

드바로가기

Ctrl+Tab

한 상태를 복제합니다.

시각적 디자인 경험 사용

시각적 디자인 경험을 사용하여 런북 워크플로를 만들고 편집하고 실행하는 방법을 알아봅니다. 워크플로가 준비되면 저장하거나 내보낼 수 있습니다. 또한 시각적 디자인 환경을 사용하여 프로토타입을 빠르게 만들 수 있습니다.

런북 워크플로 생성

1. Systems Manager Automation 콘솔에 로그인합니다.
2. 런북 생성을 선택합니다.
3. 이름 상자에 런북 이름을 입력합니다(예: *MyNewRunbook*).
4. 디자인 및 코드 토글 옆의 연필 아이콘을 선택하고 런북의 이름을 입력합니다.

이제 새 런북의 워크플로를 디자인할 수 있습니다.

런북 디자인

시각적 디자인 경험을 사용하여 런북 워크플로를 디자인하려면 작업 브라우저에서 캔버스로 자동화 작업을 드래그하여 런북의 워크플로에서 원하는 위치에 배치합니다. 워크플로에서 작업을 다른 위치로 드래그하여 작업을 재정렬할 수도 있습니다. 작업을 캔버스로 드래그하면 워크플로에서 작업을 놓을 수 있는 모든 위치에 선이 나타납니다. 작업을 캔버스에 놓으면 해당 코드가 자동 생성되어 런북의 콘텐츠에 추가됩니다.

추가하려는 작업의 이름을 알고 있는 경우 작업 브라우저 상단의 검색 상자를 사용하여 작업을 찾으세요.

캔버스에 작업을 드롭한 후 오른쪽에 있는 양식 패널을 사용하여 작업을 구성합니다. 이 패널에는 캔버스에 배치하는 각 자동화 작업 또는 API 작업에 대한 일반, 입력, 출력 및 구성 탭이 있습니다. 예를 들어 일반 탭은 다음 섹션을 포함합니다.

- 단계 이름은 단계를 식별합니다. 단계 이름에 고유한 값을 지정합니다.
- 설명은 런북의 워크플로에서 작업이 수행하는 작업을 설명하는 데 도움이 됩니다.

입력 탭에는 작업에 따라 달라지는 필드가 있습니다. 예를 들어, `aws:executeScript` automation 작업은 다음 섹션을 포함합니다.

- 런타임은 제공된 스크립트를 실행하는 데 사용하는 언어입니다.
- 핸들러는 함수의 이름입니다. 핸들러에 정의된 함수에 `events` 및 `context`의 두 개의 파라미터가 있는지 확인해야 합니다. PowerShell 런타임은 이 파라미터를 지원하지 않습니다.
- 스크립트는 워크플로 중에 실행할 기본 제공 스크립트입니다.
- (선택 사항) 첨부 파일은 작업에 의해 호출될 수 있는 독립형 스크립트 또는 .zip 파일용입니다. 이 파라미터는 JSON 런북에서 필수 항목입니다.

출력 탭에서는 작업에서 출력할 값을 지정할 수 있습니다. 워크플로의 이후 작업에서 출력 값을 참조하거나 로깅 목적으로 작업의 출력을 생성할 수 있습니다. 모든 작업이 출력을 지원하는 것은 아니므로 모든 작업에 출력 탭이 있는 것은 아닙니다. 예를 들어 `aws:pause` 작업은 출력을 지원하지 않습니다. 출력을 지원하는 작업의 경우 출력 탭은 다음 섹션으로 구성됩니다.

- 이름은 출력 값에 사용할 이름입니다. 워크플로의 이후 작업에서 출력을 참조할 수 있습니다.
- 선택기는 JSONPath 표현식 문자열로 JSON 요소 안에서 한 가지 이상의 구성 요소를 선택하는 데 사용되는 `"$."`로 시작됩니다.
- 유형은 출력 값의 데이터 유형입니다. 예를 들어 `String` 또는 `Integer` 데이터 유형입니다.

구성 탭에는 모든 자동화 작업에서 사용할 수 있는 속성과 옵션이 포함되어 있습니다. 이 템플릿은 다음 섹션으로 구성됩니다.

- Max attempts 속성은 작업이 실패할 경우 작업을 재시도하는 횟수입니다.
- Timeout seconds 속성은 작업의 제한 시간 값을 지정합니다.
- Is critical 속성은 작업 실패로 인해 전체 자동화가 중지되는지 여부를 결정합니다.
- Next step 속성은 런북에서 자동화가 다음으로 진행될 작업을 결정합니다.
- On failure 속성은 작업이 실패할 경우 자동화가 런북에서 다음으로 진행할 작업을 결정합니다.
- On cancel 속성은 사용자가 작업을 취소한 경우 자동화가 런북에서 다음 단계로 넘어갈 작업을 결정합니다.

작업을 삭제하려면 캔버스 위의 도구 모음인 백스페이스를 사용하거나 마우스 오른쪽 버튼을 클릭하고 작업 삭제를 선택하면 됩니다.

워크플로가 커지면 캔버스에 맞지 않을 수 있습니다. 워크플로를 캔버스에 맞게 만들려면 다음 옵션 중 하나를 시도할 수 있습니다.

- 측면 패널의 컨트롤을 사용하여 패널의 크기를 조정하거나 패널을 닫습니다.
- 캔버스 상단의 툴바를 사용하여 워크플로 그래프를 확대하거나 축소할 수 있습니다.

런북 업데이트

새 버전의 런북을 생성하여 기존 런북 워크플로를 업데이트할 수 있습니다. 시각적 디자인 환경을 사용하거나 코드를 직접 편집하여 런북을 업데이트할 수 있습니다. 기존 런북을 업데이트하려면 다음 절차를 따르십시오.

1. Systems Manager Automation 콘솔에 로그인합니다.
2. 업데이트할 런북을 선택합니다.
3. 새로운 버전 생성을 선택합니다.
4. 시각적 디자인 환경에는 코드 창과 시각적 워크플로 창이라는 두 개의 창이 있습니다. 시각적 워크플로 창에서 디자인을 선택하여 시각적 디자인 경험으로 워크플로를 편집합니다. 작업을 마쳤으면 새 버전 생성을 선택하여 변경 사항을 저장하고 종료합니다.
5. (선택 사항) 코드 패널을 사용하여 YAML 또는 JSON에서 런북 콘텐츠를 편집합니다.

런북 내보내기

런북의 워크플로 YAML 또는 JSON 코드와 워크플로 그래프를 내보내려면 다음 절차를 사용하십시오.

1. 문서 콘솔에서 런북을 선택합니다.
2. 새로운 버전 생성을 선택합니다.
3. 작업 드롭다운에서 그래프 또는 런북을 내보낼지 여부와 원하는 형식을 선택합니다.

작업에 대한 입력 및 출력 구성

각 자동화 작업은 수신한 입력에 따라 응답합니다. 대부분의 경우 출력을 후속 작업에 전달합니다. 시각적 디자인 환경에서는 양식 패널의 입력 및 출력 탭에서 작업의 입력 및 출력 데이터를 구성할 수 있습니다.

자동화 작업의 출력을 정의하고 사용하는 방법에 대한 자세한 내용은 작업 출력을 입력으로 사용 단원을 참조하십시오.

작업에 대한 입력 데이터 제공

자동화 작업마다 값을 제공해야 하는 한 가지 이상의 입력이 있습니다. 작업 입력에 제공하는 값은 작업에 허용되는 데이터 유형 및 형식에 따라 결정됩니다. 예를 들어 `aws:sleep` 작업의 `Duration` 입력에는 ISO 8601 형식의 문자열 값이 필요합니다.

일반적으로 런북의 워크플로에서는 후속 작업에서 사용하려는 출력을 반환하는 작업을 사용합니다. 런북 워크플로에서 오류가 발생하지 않도록 입력 값이 올바른지 확인하는 것이 중요합니다. 입력 값은 작업이 예상 출력을 반환하는지 여부를 결정하므로 중요합니다. 예를 들어 `aws:executeAwsApi` 작업을 사용할 때는 API 작업에 적합한 값을 제공하고 있는지 확인해야 합니다.

작업의 출력 데이터 정의

일부 자동화 동작은 정의된 작업을 수행한 후 출력을 반환합니다. 출력을 반환하는 작업에는 사전 정의된 출력이 있거나 사용자가 직접 출력을 정의할 수 있습니다. 예를 들어, `aws:createImage` 작업에는 `ImageId` 및 `ImageState`를 반환하는 사전 정의된 출력이 있습니다. 이에 비해 `aws:executeAwsApi` 작업을 사용하면 지정된 API 작업에서 원하는 출력을 정의할 수 있습니다. 따라서 단일 API 작업에서 하나 이상의 값을 반환하여 후속 작업에 사용할 수 있습니다.

자동화 작업에 대한 자체 출력을 정의하려면 출력 이름, 데이터 유형 및 출력 값을 지정해야 합니다. `aws:executeAwsApi` 작업을 계속 예로 들어 Amazon EC2에서 `DescribeInstances` API 작업을 직접적으로 호출한다고 가정해 보겠습니다. 이 예제에서는 `State` Amazon EC2 인스턴스를 반환하거

나 출력하고 출력을 기반으로 런북의 워크플로를 분기하려고 합니다. **InstanceState** 출력의 이름을 지정하고 **String** 데이터 유형을 사용하도록 선택합니다.

출력의 실제 값을 정의하는 프로세스는 작업에 따라 다릅니다. 예를 들어 aws:executeScript 작업을 사용하는 경우 함수에 return 명령문을 사용하여 출력값에 데이터를 제공해야 합니다. aws:executeAwsApi, aws:waitForAwsResourceProperty 및 aws:assertAwsResourceProperty과 같은 다른 작업에서는 Selector가 필수입니다. Selector 또는 일부 작업에서 참조하는 PropertySelector는 JSONPath 문자열로 API 작업의 JSON 응답을 처리하는 데 사용되는 문자열입니다. 출력에 적합한 값을 선택할 수 있으려면 API 작업의 JSON 응답 객체가 어떻게 구조화되는지 이해하는 것이 중요합니다. 앞서 언급한 DescribeInstances API 작업을 사용하는 방법은 다음 JSON 응답 예시를 참조하십시오.


```
{
  "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"
          },
```

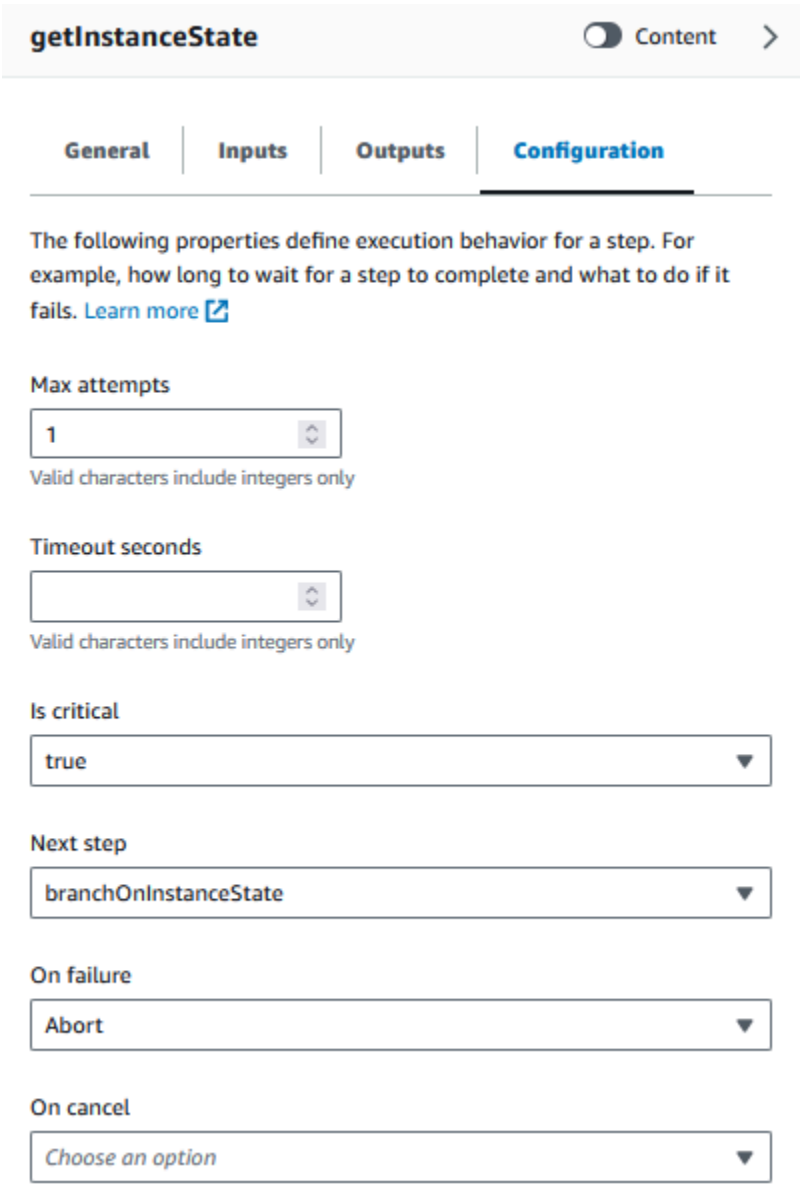
```
          "ebsOptimized": false,
          "cpuOptions": {
            "coreCount": 1,
            "threadsPerCore": 1
          }
        }
      }
    }
  }
}
```

JSON 응답 객체에서 State 인스턴스는 Instances 객체에 중첩되며, 해당 객체는 Reservations 객체에 중첩됩니다. State 인스턴스의 값을 반환하려면 출력에 해당 값을 사용할 수 있도록 Selector에 다음 문자열을 사용하십시오. **$.Reservations[0].Instances[0].State.Name**

런북 워크플로의 후속 작업에서 출력 값을 참조하려면 {{ *StepName*.*NameOfOutput* }} 형식이 사용됩니다. 예를 들면 **{{ GetInstanceState.InstanceState }}**입니다. 시각적 디자인 환경에서는 입력용 드롭다운을 사용하여 후속 작업에 사용할 출력 값을 선택할 수 있습니다. 후속 작업에서 출력을 사용할 때는 출력의 데이터 유형이 입력의 데이터 유형과 일치해야 합니다. 이 예시에서 InstanceState 출력은 String입니다. 따라서 후속 작업의 입력에서 값을 사용하려면 입력에서 String을 받아야 합니다.

시각적 디자인 경험을 통한 오류 처리

기본적으로 작업에서 오류가 보고되면 Automation은 런북의 워크플로를 완전히 중지합니다. 이는 모든 작업의 onFailure 속성 기본값이 Abort이기 때문입니다. Automation이 런북 워크플로의 오류를 처리하는 방법을 구성할 수 있습니다. 오류 처리를 구성했더라도 일부 오류로 인해 자동화가 실패할 수 있습니다. 자세한 내용은 Systems Manager Automation 문제 해결 단원을 참조하십시오. 시각적 디자인 환경에서는 구성 패널에서 오류 처리를 구성합니다.



오류 발생 시 조치 재시도

오류가 발생한 경우 작업을 재시도하려면 Max attempts 속성 값을 지정하십시오. 기본값은 1입니다. 지정된 값이 1보다 크면 모든 재시도가 실패할 때까지 해당 단계는 실패한 것으로 간주되지 않습니다.

시간 초과

작업 제한 시간을 구성하여 작업이 실패하기 전에 실행할 수 있는 최대 시간(초)을 설정할 수 있습니다. 제한 시간을 구성하려면 Timeout seconds 속성에 작업이 실패하기 전에 작업이 기다려야 하는 시간을 초로 입력합니다. 시간 제한에 도달했지만 작업의 `Max attempts` 값이 1보다 큰 경우 단계는 모든 재시도 횟수가 시도될 때까지 시간이 초과된 것으로 간주되지 않습니다.

실패한 작업

기본적으로 작업이 실패하면 Automation은 런북의 워크플로를 완전히 중지합니다. 런북에 있는 작업의 On failure 속성에 대체 값을 지정하여 이 동작을 수정할 수 있습니다. 워크플로를 런북의 다음 단계로 계속 진행하려면 계속을 선택합니다. 워크플로를 런북의 다른 후속 단계로 바로 이동시키려면 단계를 선택한 다음 단계 이름을 입력합니다.

취소된 작업

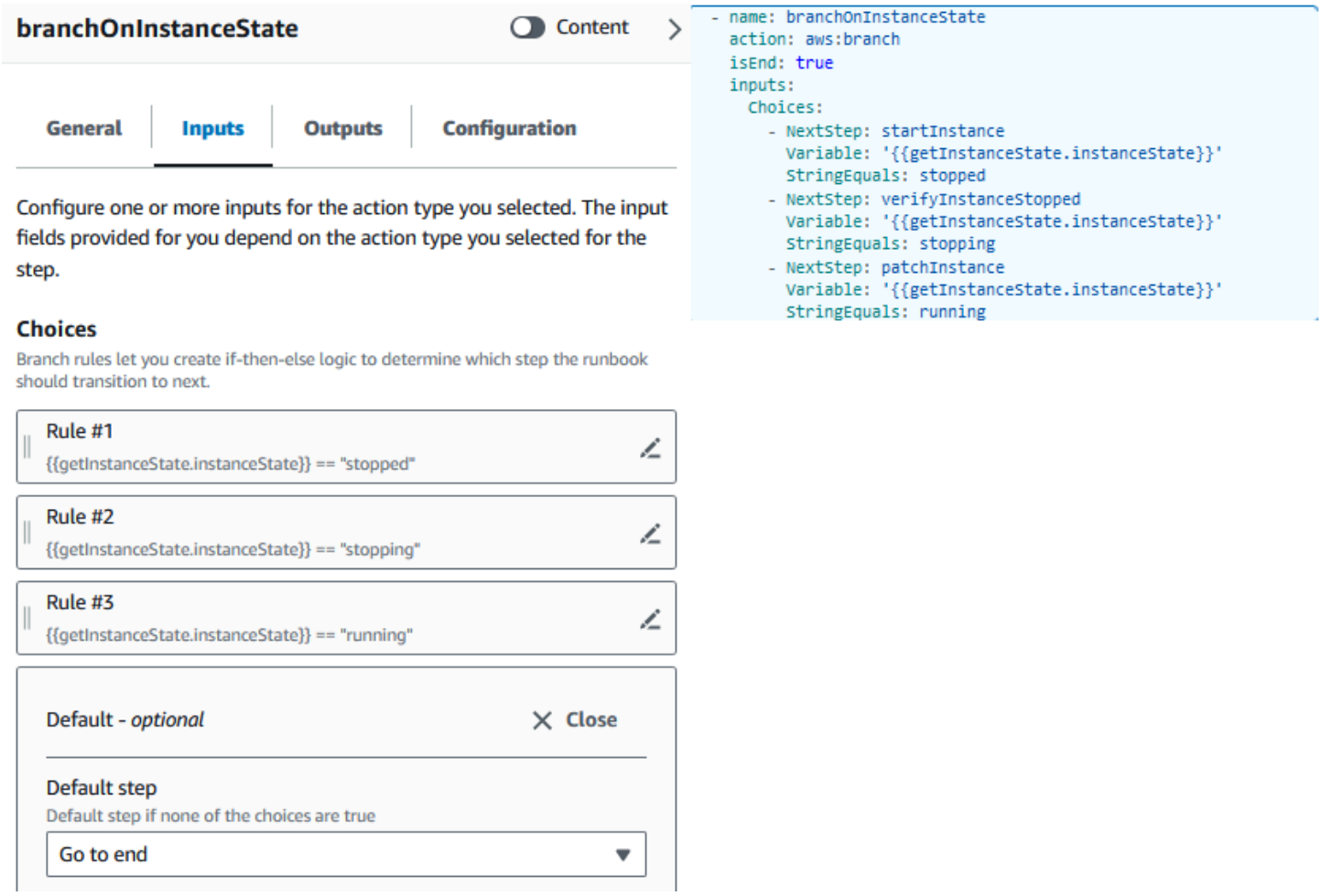
기본적으로 사용자가 작업을 취소하면 Automation은 런북의 워크플로를 완전히 중지합니다. 런북에 있는 작업의 On cancel 속성에 대체 값을 지정하여 이 동작을 수정할 수 있습니다. 워크플로를 런북의 다른 후속 단계로 바로 이동시키려면 단계를 선택한 다음 단계 이름을 입력합니다.

중요 작업

조치를 중요 조치로 지정할 수 있습니다. 즉, 해당 조치가 자동화의 전체 보고 상태를 결정합니다. 이 지정이 있는 단계 중 하나가 실패하면 Automation은 다른 작업의 성공 여부와 관계없이 `Failed` 상태로 최종 상태를 보고합니다. 작업을 중요한 작업으로 구성하려면 Is critical 속성의 기본값을 True로 그대로 두십시오.

종료 작업

Is end 속성은 지정된 작업 종료 시 자동화를 적절히 중지합니다. 이 속성의 기본값은 `false`입니다. 작업에 대해 이 속성을 구성하면 작업의 성공 여부에 관계없이 자동화가 중지됩니다. 이 속성은 예상치 못한 입력값이나 정의되지 않은 입력값을 처리하기 위한 `aws:branch` 작업과 함께 가장 자주 사용됩니다. 다음 예제는 인스턴스 상태가`running`, `stopping` 또는 `stopped`일 것으로 예상되는 런북을 보여줍니다. 인스턴스의 상태가 다른 경우 자동화가 종료됩니다.



자습서: 시각적 디자인 환경을 사용하여 런북을 생성합니다.

이 자습서에서는 Systems Manager Automation에서 제공하는 시각적 디자인 환경을 사용하여 작업하는 기본 사항을 학습합니다. 시각적 디자인 환경에서 여러 작업을 사용하는 런북을 만들 수 있습니다. 드래그 앤 드롭 기능을 사용하여 캔버스에 작업을 정렬할 수 있습니다. 또한 이러한 작업을 검색, 선택 및 구성할 수 있습니다. 그런 다음 런북의 워크플로에 대해 자동 생성된 YAML 코드를 확인하고, 시각적 디자인 환경을 종료하고, 런북을 실행하고, 실행 세부 정보를 검토할 수 있습니다.

이 자습서에서는 런북을 업데이트하고 새 버전을 보는 방법도 보여줍니다. 자습서가 끝나면 정리 단계를 수행하고 런북을 삭제하게 됩니다.

이 자습서를 완료하면 시각적 디자인 경험을 사용하여 런북을 만드는 방법을 알게 됩니다. 또한 런북을 업데이트, 실행 및 삭제하는 방법도 알게 됩니다.

> **Note**
>
> 이 자습서를 시작하기 전에 Automation 설정 항목을 완료했는지 확인하세요.

주제

1단계: 시각적 디자인 환경으로 이동

1. Systems Manager Automation 콘솔에 로그인합니다.
2. 자동화 생성을 선택합니다.

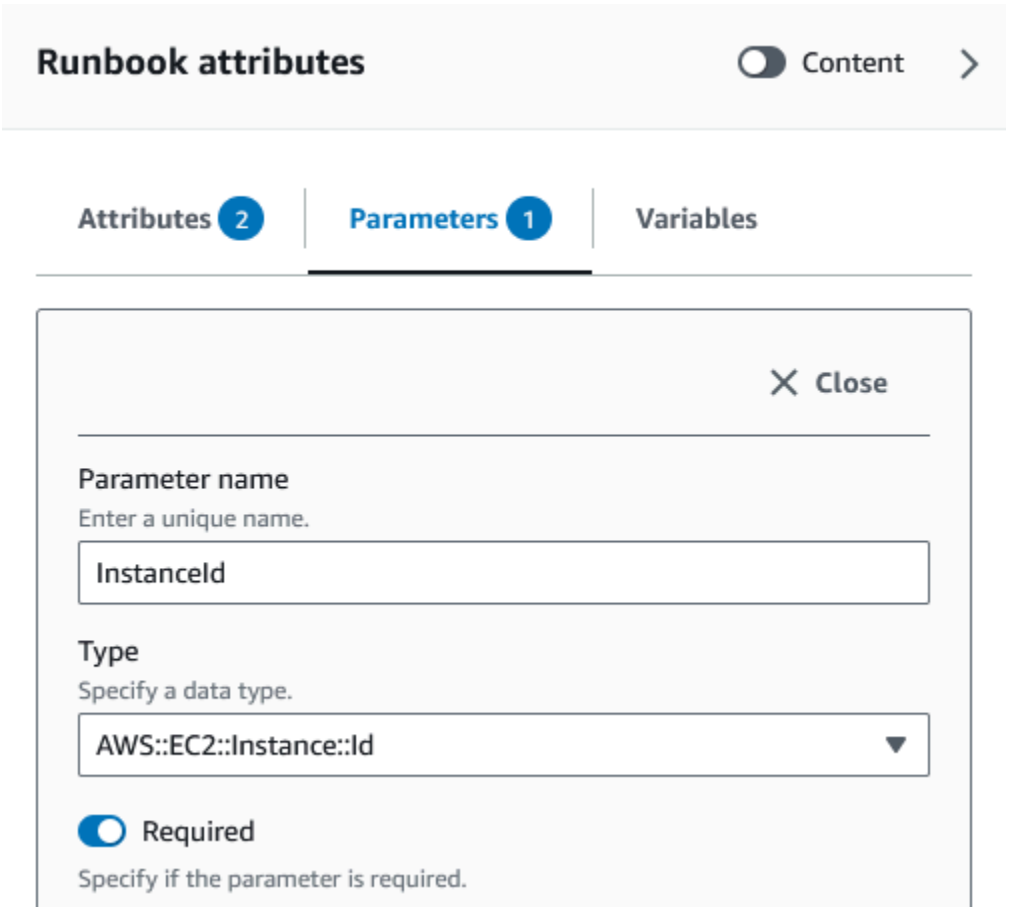
2단계: 워크플로 생성

시각적 디자인 환경에서 워크플로는 캔버스에 런북을 그래픽으로 표현한 것입니다. 시각적 디자인 경험을 사용하여 런북의 개별 작업을 정의, 구성 및 검토할 수 있습니다.

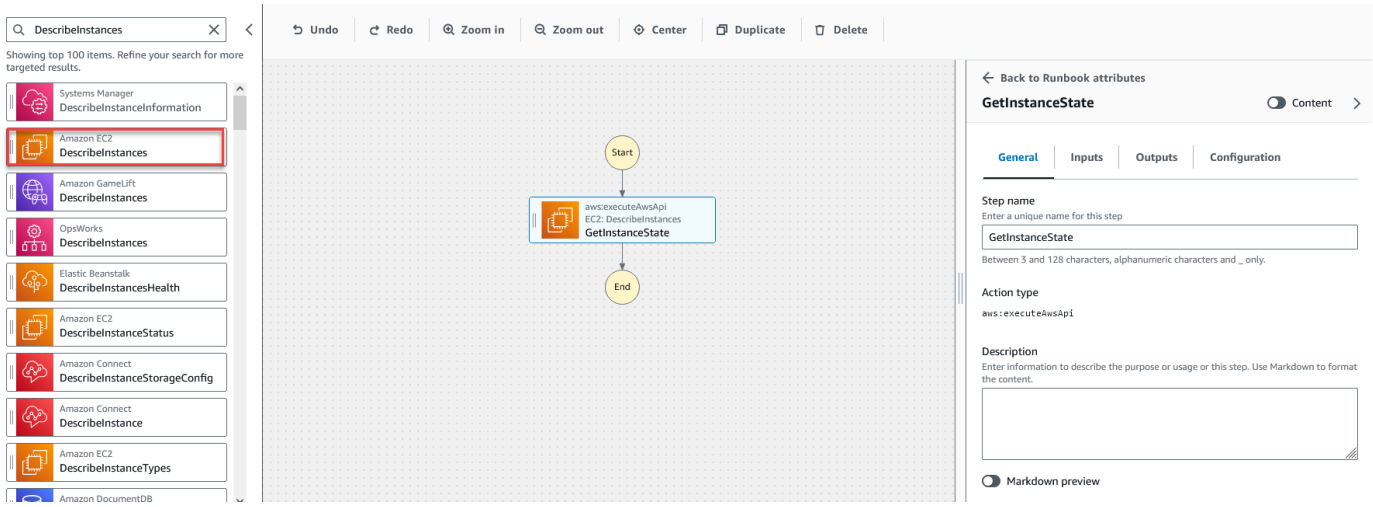
워크플로 생성 방법

1. 디자인 및 코드 토글 옆의 연필 아이콘을 선택하고 런북의 이름을 입력합니다. 이 자습서에서는 **`VisualDesignExperienceTutorial`**을 입력합니다.

2. 양식 패널의 문서 속성 섹션에서 입력 파라미터 드롭다운을 펼치고 파라미터 추가를 선택합니다.
   a. 파라미터 이름 필드에 **`InstanceId`**를 입력합니다.
   b. 유형 드롭다운에서 AWS::EC2::인스턴스를 선택합니다.
   c. 필수 토글을 선택합니다.



3. AWS API 브라우저의 검색 창에 **`DescribeInstances`**를 입력합니다.
4. Amazon EC2 — DescribeInstances 작업을 빈 캔버스로 드래그하십시오.
5. 단계 이름에 값을 입력합니다. 본 자습서에서는 **`GetInstanceState`** 이름을 사용할 수 있습니다.



   a. 추가 입력 드롭다운을 확장하고 입력 이름 필드에 **`InstanceIds`**를 입력합니다.

b. 입력 탭을 선택합니다.

c. 입력 값 필드에서 **InstanceId** 문서 입력을 선택합니다. 이 값은 프로시저를 시작할 때 생성한 입력 파라미터의 값을 참조합니다. `DescribeInstances` 작업에 대한 InstanceIds 입력은 `StringList` 값을 허용하므로 InstanceId 입력을 대괄호로 묶어야 합니다. 입력 값의 YAML은 다음과 일치해야 합니다. **`['{{ InstanceId }}']`**

d. 출력 탭에서 출력 추가를 선택하고 이름 필드에 **InstanceState**를 입력합니다.

e. 선택기 필드에 **$.Reservations[0].Instances[0].State.Name** 항목을 입력합니다.

f. 유형 드롭다운에서 문자열을 선택합니다.

6. 작업 브라우저에서 분기 작업을 드래그하여 **GetInstanceState**단계 아래에 놓습니다.

7. 단계 이름에 값을 입력합니다. 본 튜토리얼에서는 **BranchOnInstanceState**의 이름을 사용합니다.

   분기 로직을 정의하려면 다음을 수행합니다.

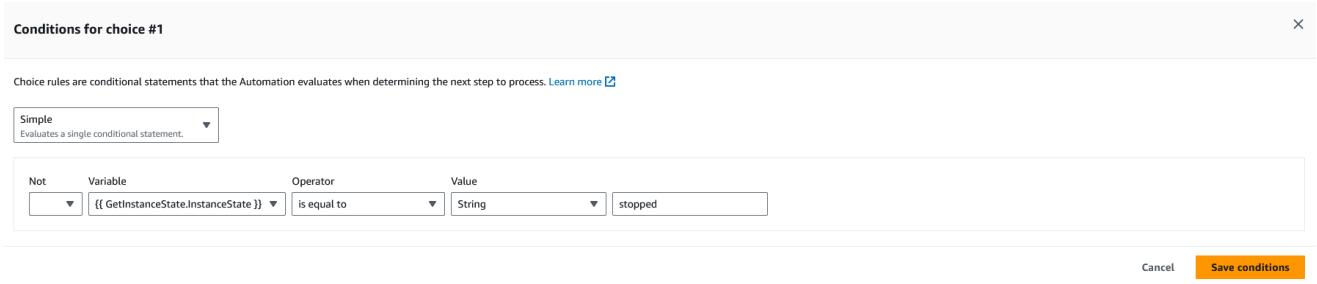
   a. 캔버스에서 **Branch**상태를 선택합니다. 그런 다음 입력 및 선택에서 연필 아이콘을 선택하여 규칙 #1 를 편집합니다.

   b. 조건 추가를 선택합니다.

   c. 규칙 #1 조건 대화 상자의 변수 드롭다운에서 **GetInstanceState.InstanceState** 단계 출력을 선택합니다.

   d. 연산자에서 같음을 선택합니다.

   e. 값의 경우. 드롭다운 목록에서 문자열을 선택합니다. **stopped**을 입력합니다.



   f. 조건 저장을 선택합니다.

   g. 새 선택 규칙 추가를 선택합니다.

   h. 규칙 #2 조건 추가를 선택합니다.

   i. 규칙 #2 조건 대화 상자의 변수 드롭다운에서 **GetInstanceState.InstanceState**단계 출력을 선택합니다.

   j. 연산자에서 같음을 선택합니다.

k. 값의 경우. 드롭다운 목록에서 문자열을 선택합니다. **stopping**을 입력합니다.

l. 조건 저장을 선택합니다.

m. 새 선택 규칙 추가를 선택합니다.

n. 규칙 #3 에서는 조건 추가를 선택합니다.

o. 규칙 #3 조건 대화 상자의 변수 드롭다운에서 **GetInstanceState.InstanceState**단계 출력을 선택합니다.

p. 연산자에서 같음을 선택합니다.

q. 값의 경우. 드롭다운 목록에서 문자열을 선택합니다. **running**을 입력합니다.

r. 조건 저장을 선택합니다.

s. 기본 규칙에서 기본 단계로 Go to end를 선택합니다.

8. 인스턴스 상태 변경 작업을 {{ GetInstanceState.InstanceState }} == "stopped" 조건 아래의 비어 있는 이곳으로 작업 드래그 상자로 드래그하십시오.

   a. 단계 이름에는 **StartInstance**를 입력합니다.

   b. 입력 탭의 인스턴스 ID 아래에 있는 드롭다운에서 InstanceID 문서 입력 값을 선택합니다.

   c. 원하는 상태에는 **running** 항목을 지정합니다.

9. AWS 리소스 대기 작업을 {{ GetInstanceState.InstanceState }} == "stopping" 조건 아래의 비어 있는 이곳으로 작업 드래그로 드래그하십시오.

10. 단계 이름에 값을 입력합니다. 본 튜토리얼에서는 **WaitForInstanceStop**의 이름을 사용합니다.

    a. 서비스 필드에서 Amazon EC2를 선택합니다.

    b. API 필드에서 DescribeInstances를 선택합니다.

    c. 속성 선택기 필드에 **$.Reservations[0].Instances[0].State.Name**를 입력합니다.

    d. 원하는 값 파라미터의 경우 **["stopped"]**를 입력합니다.

    e. WaitForInstanceStop 작업의 구성 탭에 있는 다음 단계 드롭다운에서 인스턴스 시작을 선택합니다.

11. 인스턴스에서 명령 실행 작업을 {{ GetInstanceState.InstanceState }} == "running" 조건의 비어 있는 이곳으로 작업 드래그 박스로 드래그하십시오.

12. 단계 이름에는 **SayHello**를 입력합니다.

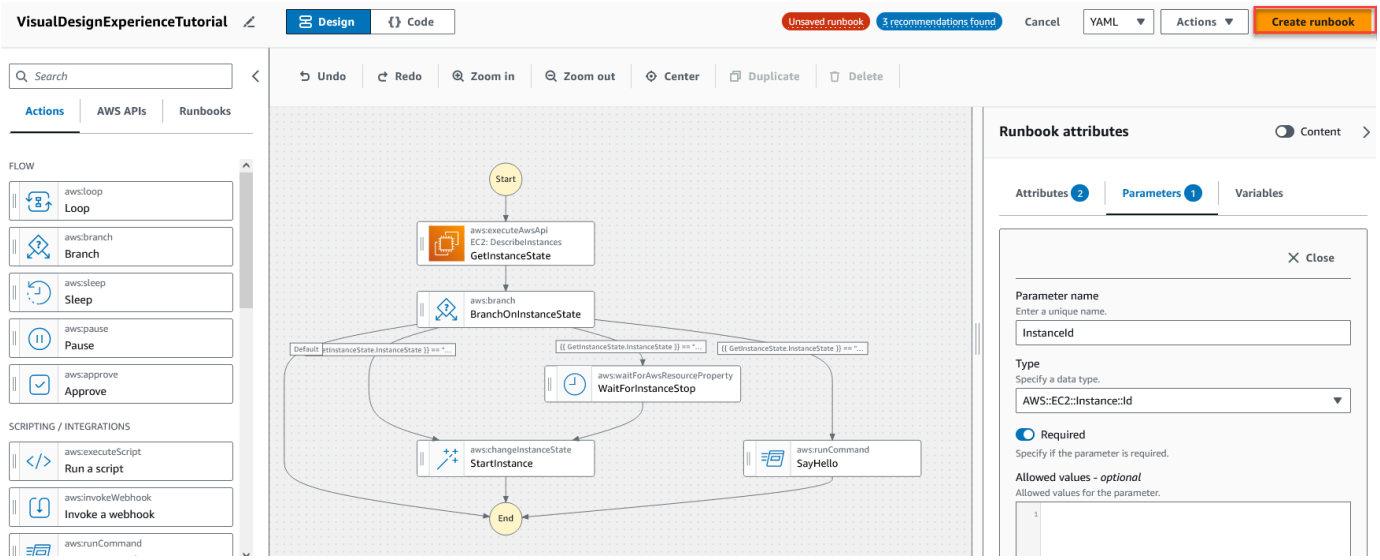
    a. 입력 탭에서 문서 이름 파라미터로 **AWS-RunShellScript**를 입력합니다.

    b. InstanceID의 경우 드롭다운에서 InstanceID 문서 입력 값을 선택합니다.

c. 추가 입력 드롭다운을 확장하고 이름 입력 드롭다운에서 파라미터를 선택합니다.

d. 입력 값 필드에 **{"commands": "echo 'Hello World'"}**를 입력합니다.

13. 캔버스에서 완성된 런북을 검토하고 런북 생성을 선택하여 튜토리얼 런북을 저장합니다.



3단계: 자동 생성 코드 검토

작업 브라우저에서 캔버스로 작업을 드래그 앤 드롭하면 시각적 디자인 경험이 런북의 YAML 또는 JSON 콘텐츠를 실시간으로 자동으로 작성합니다. 이 코드를 보고 편집할 수 있습니다. 자동 생성된 코드를 보려면 디자인용 코드 및 코드 토글을 선택합니다.

4단계: 새 런북 실행

런북을 만든 후 자동화를 실행할 수 있습니다.

새 Automation 런북 실행 방법

1. AWS Systems Manager 콘솔https://console.aws.amazon.com/systems-manager/을 엽니다.
2. 탐색 창에서 Automation(자동화)을 선택한 후 Execute automation(자동화 실행)을 선택합니다.
3. [Automation 문서(Automation document)] 목록에서 실행서를 선택합니다. 문서 카테고리 창에서 옵션을 1개 이상 선택하여 SSM 문서를 목적에 따라 필터링합니다. 자신이 소유한 실행서를 보려면 [내 소유(Owned by me)] 탭을 선택합니다. 자신의 계정과 공유하고 있는 실행서를 보려면 [나와 공유됨(Shared with me)] 탭을 선택합니다. 모든 실행서를 보려면 [모든 문서(All documents)] 탭을 선택합니다.

> **Note**
>
> 실행서 이름을 선택하여 실행서에 대한 정보를 볼 수 있습니다.

4. 문서 세부 정보 섹션에서 문서 버전이 실행할 버전으로 설정되었는지 확인합니다. 이 시스템에는 다음 버전 옵션이 포함되어 있습니다.
   - 런타임 시 기본 버전 - Automation 런북이 정기적으로 업데이트되며 새 기본 버전이 할당된 경우 이 옵션을 선택합니다.
   - 런타임 시 최신 버전 - Automation 런북이 정기적으로 업데이트되며 최근에 업데이트된 버전을 실행하려는 경우 이 옵션을 선택합니다.
   - 1(기본값) - 문서의 최초 버전을 실행하려면 이 옵션을 선택합니다(기본값).
5. 다음을 선택합니다.
6. 자동화 문서 실행 섹션에서 단순 실행을 선택합니다.
7. 입력 파라미터 섹션에서 필수 입력을 지정합니다. 필요에 따라 AutomationAssumeRole 목록에서 IAM 서비스 역할을 선택합니다.
8. (선택 사항) 모니터링을 위해 자동화에 적용할 Amazon CloudWatch 경보를 선택합니다. CloudWatch 경보를 자동화에 연결하려면 자동화를 시작하는 IAM 보안 주체에 `iam:createServiceLinkedRole` 작업에 대한 권한이 있어야 합니다. CloudWatch 경보에 대한 자세한 내용은 Amazon CloudWatch 경보 사용을 참조하세요. 경보가 활성화되면 자동화가 중지됩니다. AWS CloudTrail을 사용하면 추적에 API 호출이 표시됩니다.
9. 실행을 선택합니다.

5단계: 정리

런북 삭제 방법

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. 내 소유 탭을 선택합니다.
4. VisualDesignExperienceTutorial 런북을 찾습니다.
5. 문서 카드 페이지에서 버튼을 선택한 다음 작업 드롭다운에서 문서 삭제를 선택합니다.

Automation 실행서 작성

AWS Systems Manager의 기능인 Automation의 각 실행서는 자동화를 정의합니다. Automation 실행서는 자동화 중 수행되는 작업을 정의합니다. 실행서 내용에 Systems Manager가 관리형 인스턴스 및 AWS 리소스에 대해 수행하는 입력 파라미터, 출력 및 작업을 정의합니다.

Automation에는 사전 정의된 실행서가 몇 가지 포함되며, 이들 실행서를 사용하여 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스를 하나 이상 다시 시작하거나 Amazon Machine Image(AMI)를 생성하는 등 일반적인 태스크를 수행할 수 있습니다. 그러나 사용 사례가 미리 정의된 실행서의 기능 이상으로 확장될 수 있습니다. 이 경우 사용자 고유의 실행서를 생성하고 필요에 맞게 수정할 수 있습니다.

실행서는 자동화 작업, 이러한 작업에 대한 파라미터 및 사용자가 지정하는 입력 파라미터로 구성됩니다. 실행서의 내용은 YAML 또는 JSON으로 작성됩니다. YAML 또는 JSON에 익숙하지 않은 경우 비주얼 디자이너를 사용하거나 자체 런북을 작성하기 전에 마크업 언어를 자세히 알아보는 것이 좋습니다. 비주얼 디자이너에 대한 자세한 내용은 Automation 런북의 시각적 디자인 경험 섹션을 참조하세요.

다음 섹션은 첫 번째 실행서를 작성하는 데 도움이 됩니다.

사용 사례 식별

실행서 작성의 첫 번째 단계는 사용 사례를 식별하는 것입니다. 예를 들어 모든 프로덕션 Amazon EC2 인스턴스에서 매일 실행되도록 `AWS-CreateImage` 실행서를 예약했습니다. 월말에 복구 지점에 필요한 것보다 많은 이미지가 있다고 판단됩니다. 앞으로 새로운 AMI가 생성될 때 Amazon EC2 인스턴스의 가장 오래된 AMI를 자동으로 삭제하려고 합니다. 이를 위해 다음을 수행하는 새 실행서를 생성합니다.

1. `aws:createImage` 작업을 실행하고 이미지 설명에 인스턴스 ID를 지정합니다.
2. `aws:waitForAwsResourceProperty` 작업을 실행하여 `available` 상태가 될 때까지 이미지의 상태를 폴링합니다.
3. 이미지 상태가 `available`이면 `aws:executeScript` 작업은 Amazon EC2 인스턴스와 연결된 모든 이미지의 ID를 수집하는 사용자 정의 Python 스크립트를 실행합니다. 스크립트는 생성 시 지정한 이미지 설명의 인스턴스 ID로 필터링하여 이를 수행합니다. 그런 다음 스크립트는 이미지의 `creationDate`를 기준으로 이미지 ID 목록을 정렬하고 가장 오래된 AMI의 ID를 출력합니다.
4. 마지막으로 `aws:deleteImage` 작업이 실행되어 이전 단계의 출력에서 ID를 사용하여 가장 오래된 AMI를 삭제합니다.

이 시나리오에서는 이미 AWS-CreateImage 실행서를 사용하고 있었지만 사용 사례에 더 큰 유연성이 필요함을 알게 되었습니다. 이는 실행서와 자동화 작업 간에 겹칠 수 있기 때문에 일반적인 상황입니다. 따라서 사용 사례를 해결하기 위해 사용하는 실행서 또는 작업을 조정해야 할 수 있습니다.

예를 들어 aws:executeScript 및 aws:invokeLambdaFunction 작업을 모두 사용하면 자동화의 일부로 사용자 정의 스크립트를 실행할 수 있습니다. 추가로 지원되는 런타임 언어 때문에 aws:invokeLambdaFunction을 선호할 수 있습니다. 그러나 YAML 실행서에서 직접 스크립트 콘텐츠를 작성하고 JSON 콘텐츠에 대한 첨부 파일로 스크립트 콘텐츠를 제공할 수 있으므로 aws:executeScript를 선호할 수 있습니다. AWS Identity and Access Management(IAM) 설정 측면에서 aws:executeScript가 더 간단하다고 생각할 수도 있습니다. AutomationAssumeRole에서 제공하는 권한을 사용하기 때문에 aws:executeScript는 추가적인 AWS Lambda 기능 실행 역할이 필요하지 않습니다.

주어진 시나리오에서 한 작업은 다른 작업보다 더 많은 유연성을 제공하거나 기능을 추가할 수 있습니다. 따라서 사용 사례 및 기본 설정에 가장 적합한 것을 결정하기 위해 사용하려는 실행서 또는 작업에 사용 가능한 입력 파라미터를 검토하는 것이 좋습니다.

개발 환경 설정

실행서에서 사용하려는 사용 사례와 미리 정의된 실행서 또는 자동화 작업을 식별한 후에는 실행서의 콘텐츠에 대한 개발 환경을 설정할 차례입니다. 실행서 콘텐츠를 개발하려면 Systems Manager Documents 콘솔 대신 AWS Toolkit for Visual Studio Code를 사용하는 것이 좋습니다.

Toolkit for VS Code는 Systems Manager Documents 콘솔보다 더 많은 기능을 제공하는 Visual Studio Code(VS Code)용 오픈 소스 확장입니다. 유용한 기능에는 YAML 및 JSON 모두에 대한 스키마 검증, 자동화 작업 유형에 대한 코드 조각, YAML 및 JSON 모두의 다양한 옵션에 대한 자동 완성 지원이 있습니다.

Toolkit for VS Code 설치에 대한 자세한 내용은 AWS Toolkit for Visual Studio Code 설치를 참조하세요. 실행서 개발에 Toolkit for VS Code 사용에 대한 자세한 내용은 AWS Toolkit for Visual Studio Code User Guide의 Working with Systems Manager Automation documents를 참조하세요.

실행서 콘텐츠 개발

사용 사례가 식별되고 환경이 설정되면 실행서용 콘텐츠를 개발할 준비가 된 것입니다. 사용 사례와 기본 설정에 따라 실행서 콘텐츠에서 사용하는 자동화 작업 또는 실행서가 주로 결정됩니다. 일부 작업은 유사한 태스크를 수행할 수 있는 다른 작업과 비교할 때 입력 파라미터의 하위 집합만 지원합니다. 다른 작업에는 aws:createImage와 같은 특정 출력이 있으며, 일부 작업에서는 aws:executeAwsApi와 같이 고유한 출력을 정의할 수 있습니다.

실행서에서 특정 작업을 사용하는 방법을 잘 모르는 경우 Systems Manager Automation 작업 참조에서 작업에 대한 해당 항목을 검토하는 것이 좋습니다. 또한 미리 정의된 실행서의 콘텐츠를 검토하여 이러한 작업이 사용되는 실제 사례를 확인하는 것이 좋습니다. 실행서의 실제 적용에 대한 더 많은 예는 추가 런북 예제 섹션을 참조하세요.

실행서 콘텐츠가 제공하는 단순성과 유연성의 차이점을 보여주기 위해 다음 자습서에서는 Amazon EC2 인스턴스의 그룹에 단계별로 패치하는 방법의 예를 제공합니다.

- the section called “예 1: 상위-하위 실행서 생성” - 이 예에서는 2개의 실행서가 상위-하위 관계로 사용됩니다. 상위 실행서는 하위 실행서의 속도 제어 자동화를 시작합니다.
- the section called “예제 2: 스크립팅된 실행서” - 이 예에서는 콘텐츠를 단일 실행서로 압축하고 실행서에서 스크립트를 사용하여 예 1과 동일한 태스크를 수행하는 방법을 보여줍니다.

예 1: 상위-하위 실행서 생성

다음 예에서는 태그가 지정된 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 그룹에 단계별로 패치하는 2개의 실행서를 생성하는 방법을 보여줍니다. 이러한 실행서는 하위 실행서의 속도 제어 자동화를 시작하는 데 사용되는 상위 실행서와의 상위-하위 관계에 사용됩니다. 속도 제어 자동화에 대한 자세한 내용은 대규모로 자동화 실행 섹션을 참조하세요. 이 예에 사용되는 자동화 작업에 대한 자세한 내용은 Systems Manager Automation 작업 참조 섹션을 참조하세요.

하위 실행서 생성

이 예제 실행서는 다음 시나리오를 다룹니다. Emily는 AnyCompany Consultants, LLC의 시스템 엔지니어입니다. 그녀는 기본 및 보조 데이터베이스를 호스팅하는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 그룹에 대한 패치를 구성해야 합니다. 애플리케이션은 하루 24시간 이러한 데이터베이스에 액세스하므로 데이터베이스 인스턴스 중 하나를 항상 사용할 수 있어야 합니다.

그녀는 인스턴스에 단계적으로 패치하는 것이 가장 좋은 방법이라고 결정합니다. 데이터베이스 인스턴스의 주 그룹이 먼저 패치되고 데이터베이스 인스턴스의 보조 그룹이 패치됩니다. 또한 이전에 중지된 인스턴스를 계속 실행하여 추가 비용이 발생하지 않도록 Emily는 패치된 인스턴스가 패치되기 전의 원래 상태로 돌아가기를 원합니다.

Emily는 인스턴스와 연결된 태그로 데이터베이스 인스턴스의 주 그룹과 보조 그룹을 식별합니다. 그녀는 하위 실행서의 속도 제어 자동화를 시작하는 상위 실행서를 생성하기로 결정합니다. 이를 통해 데이터베이스 인스턴스의 주 및 보조 그룹과 연결된 태그를 대상으로 지정하고 하위 자동화의 동시성을 관리할 수 있습니다. 패치에 사용할 수 있는 Systems Manager(SSM) 문서를 검토한 후 그녀는 `AWS-RunPatchBaseline` 문서를 선택합니다. 이 SSM 문서를 사용하여 동료는 패치 작업이 완료된 후 관련 패치 규정 준수 정보를 검토할 수 있습니다.

실행서 콘텐츠 생성을 시작하기 위해 Emily는 사용 가능한 자동화 작업을 검토하고 다음과 같이 하위 실행서에 대한 콘텐츠 작성을 시작합니다.

1. 먼저 실행서의 스키마 및 설명 값을 제공하고 하위 실행서에 대한 입력 파라미터를 정의합니다.

   Emily와 동료는 AutomationAssumeRole 파라미터를 사용하여 Automation이 대신 실행서에서 작업을 수행할 수 있도록 하는 기존 IAM 역할을 사용할 수 있습니다. Emily는 InstanceId 파라미터를 사용하여 패치해야 하는 인스턴스를 결정합니다. 필요에 따라 Operation, RebootOption 및 SnapshotId 파라미터를 사용하여 AWS-RunPatchBaseline에 대한 문서 파라미터에 값을 제공 할 수 있습니다. 잘못된 값이 해당 문서 파라미터에 제공되지 않도록 필요에 따라 allowedValues를 정의합니다.

   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. 최상위 요소가 정의되면 Emily는 실행서의 mainSteps를 구성하는 작업 작성을 진행합니다. 첫 번째 단계는 aws:executeAwsApi 작업을 사용하여 InstanceId 입력 파라미터에 지정된 대상 인스턴스의 현재 상태를 출력합니다. 이 작업의 출력은 이후 작업에 사용됩니다.

   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. Emily는 패치가 필요한 모든 인스턴스의 원래 상태를 수동으로 시작하고 추적하는 대신 이전 작업의 출력을 사용하여 대상 인스턴스의 상태를 기반으로 자동화를 분기합니다. 이를 통해 자동화는 `aws:branch` 작업에 정의된 조건에 따라 다른 단계를 실행할 수 있으며 수동 개입 없이 자동화의 전반적인 효율성이 향상됩니다.

   인스턴스 상태가 이미 `running`이면 자동화는 `aws:runCommand` 작업을 사용하여 `AWS-RunPatchBaseline` 문서로 인스턴스 패치를 진행합니다.

   인스턴스 상태가 `stopping`이면 자동화는 인스턴스가 `aws:waitForAwsResourceProperty` 작업을 사용하여 `stopped` 상태에 도달하도록 폴링하고, `executeAwsApi` 작업을 사용하여 인스턴스를 시작하고, 인스턴스를 패치하기 전에 인스턴스가 `running` 상태에 도달하도록 폴링합니다.

   인스턴스 상태가 `stopped`이면 자동화는 인스턴스를 시작하고 동일한 작업을 사용하여 인스턴스를 패치하기 전에 인스턴스가 `running` 상태에 도달하도록 폴링합니다.

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. 패치 작업이 완료된 후 Emily는 자동화가 대상 인스턴스를 자동화가 시작되기 전과 동일한 상태로 되돌리기를 원합니다. 그녀는 첫 번째 작업의 출력을 다시 사용하여 이 작업을 수행합니다. `aws:branch` 작업을 사용하여 대상 인스턴스의 원래 상태를 기반으로 자동화가 분기합니다. 인스턴스가 이전에 `running`이 아닌 다른 상태에 있었다면 인스턴스가 중지됩니다. 그렇지 않고 인스턴스 상태가 `running`이면 자동화가 종료됩니다.

   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. Emily는 완성된 하위 실행서 콘텐츠를 검토하고 대상 인스턴스와 동일한 AWS 계정 및 AWS 리전에 실행서를 생성합니다. 이제 그녀는 계속해서 상위 실행서의 콘텐츠를 생성할 준비가 되었습니다. 다음은 완성된 하위 실행서 콘텐츠입니다.

   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}}"
             ]
          }
       }
    ]
}
```

이 예에 사용되는 자동화 작업에 대한 자세한 내용은 Systems Manager Automation 작업 참조 섹션을 참조하세요.

상위 실행서 생성

이 예제 실행서는 이전 섹션에서 설명한 시나리오를 계속합니다. 하위 실행서를 생성했으므로 이제 Emily는 다음과 같이 상위 실행서에 대한 콘텐츠 작성을 시작합니다.

1. 먼저 실행서의 스키마 및 설명 값을 제공하고 하위 실행서에 대한 입력 파라미터를 정의합니다.

   Emily와 동료는 `AutomationAssumeRole` 파라미터를 사용하여 Automation이 대신 실행서에서 작업을 수행할 수 있도록 하는 기존 IAM 역할을 사용할 수 있습니다. Emily는 `PatchGroupPrimaryKey` 및 `PatchGroupPrimaryValue` 파라미터를 사용하여 패치될 데이터베이스 인스턴스의 주 그룹과 연결된 태그를 지정합니다. 그녀는 `PatchGroupSecondaryKey` 및 `PatchGroupSecondaryValue` 파라미터를 사용하여 패치될 데이터베이스 인스턴스의 보조 그룹과 연결된 태그를 지정합니다.

   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. 최상위 요소가 정의되면 Emily는 실행서의 `mainSteps`를 구성하는 작업 작성을 진행합니다.

첫 번째 작업은 PatchGroupPrimaryKey 및 PatchGroupPrimaryValue 입력 파라미터에 지정된 태그와 연결된 인스턴스를 대상으로 하는 방금 생성한 하위 실행서를 사용하여 속도 제어 자동화를 시작합니다. 그녀는 입력 파라미터에 제공된 값을 사용하여 패치하려는 데이터베이스 인스턴스의 주 그룹과 연결된 태그의 키와 값을 지정합니다.

첫 번째 자동화가 완료된 후 두 번째 작업은 PatchGroupSecondaryKey 및 PatchGroupSecondaryValue 입력 파라미터에 지정된 태그와 연결된 인스턴스를 대상으로 하는 하위 실행서를 사용하여 다른 속도 제어 자동화를 시작합니다. 그녀는 입력 파라미터에 제공된 값을 사용하여 패치하려는 데이터베이스 인스턴스의 보조 그룹과 연결된 태그의 키와 값을 지정합니다.

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. Emily는 완성된 상위 실행서 콘텐츠를 검토하고 대상 인스턴스와 동일한 AWS 계정 및 AWS 리전에 실행서를 생성합니다. 이제 프로덕션 환경에 구현하기 전 실행서를 테스트하여 자동화가 원하는 대로 작동하는지 확인할 준비가 되었습니다. 다음은 완성된 상위 실행서 콘텐츠입니다.

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"
         }
      }
   ]
}
```

이 예에 사용되는 자동화 작업에 대한 자세한 내용은 Systems Manager Automation 작업 참조 섹션을 참조하세요.

예제 2: 스크립팅된 실행서

이 예제 실행서는 다음 시나리오를 다룹니다. Emily는 AnyCompany Consultants, LLC의 시스템 엔지니어입니다. 그녀는 이전에 주 및 보조 데이터베이스를 호스팅하는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스의 패치 그룹에 대한 상위-하위 관계에서 사용되는 2개의 실행서를 생성했습니다. 애플리케이션은 하루 24시간 이러한 데이터베이스에 액세스하므로 데이터베이스 인스턴스 중 하나를 항상 사용할 수 있어야 합니다.

이 요구 사항에 따라 그녀는 `AWS-RunPatchBaseline` Systems Manager(SSM) 문서를 사용하여 단계적으로 인스턴스를 패치하는 솔루션을 구축했습니다. 이 SSM 문서를 사용하여 동료는 패치 작업이 완료된 후 관련 패치 규정 준수 정보를 검토할 수 있습니다.

데이터베이스 인스턴스의 주 그룹이 먼저 패치된 다음 데이터베이스 인스턴스의 보조 그룹이 패치됩니다. 또한 이전에 중지된 인스턴스를 계속 실행하여 추가 비용이 발생하지 않도록 Emily는 자동화가 패치된 인스턴스를 패치되기 전의 원래 상태로 되돌리도록 했습니다. Emily는 데이터베이스 인스턴스의 주 및 보조 그룹과 연결된 태그를 사용하여 원하는 순서로 패치해야 하는 인스턴스를 식별했습니다.

기존 자동화 솔루션은 작동하지만 가능한 경우 솔루션을 개선하고자 합니다. 실행서 콘텐츠의 유지 관리를 지원하고 문제 해결을 용이하게 하기 위해 자동화를 단일 실행서로 축소하고 입력 파라미터의 수를 단순화하려고 합니다. 또한 여러 하위 자동화를 생성하지 않으려고 합니다.

Emily는 사용 가능한 자동화 작업을 검토한 후 `aws:executeScript` 작업으로 사용자 정의 Python 스크립트를 실행하여 솔루션을 개선할 수 있다고 판단합니다. 이제 다음과 같이 실행서의 콘텐츠를 작성하기 시작합니다.

1. 먼저 실행서의 스키마 및 설명 값을 제공하고 하위 실행서에 대한 입력 파라미터를 정의합니다.

   Emily와 동료는 `AutomationAssumeRole` 파라미터를 사용하여 Automation이 대신 실행서에서 작업을 수행할 수 있도록 하는 기존 IAM 역할을 사용할 수 있습니다. 예 1과 달리 `AutomationAssumeRole` 파라미터는 이제 옵션이 아닌 필수 항목입니다. 이 실행서에는 `aws:executeScript` 작업이 포함되어 있으므로 AWS Identity and Access Management(IAM) 서비스 역할(또는 수임 역할)이 항상 필요합니다. 이 요구 사항은 작업에 대해 지정된 일부 Python 스크립트가 AWS API 작업을 호출하기 때문에 필요합니다.

   Emily는 `PrimaryPatchGroupTag` 및 `SecondaryPatchGroupTag` 파라미터를 사용하여 패치될 데이터베이스 인스턴스의 주 및 보조 그룹과 연결된 태그를 지정합니다. 필요한 입력 파라미터를 단순화하기 위해 예제 1 실행서와 같이 `String` 파라미터를 여러 개 사용하는 대신 `StringMap` 파라

미터를 사용하기로 결정했습니다. 필요에 따라 Operation, RebootOption 및 SnapshotId 파라미터를 사용하여 AWS-RunPatchBaseline에 대한 문서 파라미터에 값을 제공 할 수 있습니다. 잘못된 값이 해당 문서 파라미터에 제공되지 않도록 필요에 따라 allowedValues를 정의합니다.

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. 최상위 요소가 정의되면 Emily는 실행서의 `mainSteps`를 구성하는 작업 작성을 진행합니다. 첫 번째 단계는 `PrimaryPatchGroupTag` 파라미터에 지정된 태그와 연결된 모든 인스턴스의 ID를 수집하고 인스턴스 ID와 인스턴스의 현재 상태를 포함하는 `StringMap` 파라미터를 출력합니다. 이 작업의 출력은 이후 작업에 사용됩니다.

   참고: `script` 입력 파라미터는 JSON 실행서에서 지원되지 않습니다. JSON 실행서는 `attachment` 입력 파라미터를 사용하여 스크립트 콘텐츠를 제공해야 합니다.

   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. Emily는 다른 `aws:executeScript` 작업에서 이전 작업의 출력을 사용하여 `PrimaryPatchGroupTag` 파라미터에 지정된 태그와 연결된 모든 인스턴스가 `running` 상태인지 확인합니다.

   인스턴스 상태가 이미 `running` 또는 `shutting-down`인 경우 스크립트는 나머지 인스턴스를 계속 반복합니다.

   인스턴스 상태가 `stopping`이면 스크립트는 `stopped` 상태에 도달하기 위해 인스턴스를 폴링하고 인스턴스를 시작합니다.

   인스턴스 상태가 `stopped`이면 스크립트는 인스턴스를 시작합니다.

   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. Emily는 PrimaryPatchGroupTag 파라미터에 지정된 태그와 연결된 모든 인스턴스가 시작되었거나 이미 running 상태인지 확인합니다. 그런 다음 다른 스크립트를 사용하여 이전 작업에서 시작된 인스턴스를 포함한 모든 인스턴스가 running 상태에 도달했는지 확인합니다.

   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. Emily는 2개의 스크립트를 더 사용하여 PrimaryPatchGroupTag 파라미터에 지정된 태그의 키와 값의 개별 String 값을 반환합니다. 이러한 작업에서 반환된 값을 통해 그녀는 AWS-RunPatchBaseline 문서의 Targets 파라미터에 직접 값을 제공할 수 있습니다. 그런 다음 자동화는 aws:runCommand 작업을 사용하여 AWS-RunPatchBaseline 문서로 인스턴스 패치를 진행합니다.

   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. 패치 작업이 완료된 후 Emily는 자동화가 PrimaryPatchGroupTag 파라미터에 지정된 태그와 연결된 대상 인스턴스를 자동화가 시작되기 전과 동일한 상태로 되돌리기를 원합니다. 그녀는 스크립트의 첫 번째 작업의 출력을 다시 사용하여 이 작업을 수행합니다. 대상 인스턴스의 원래 상태를 기준으로 인스턴스가 이전에 running 이외의 상태였으면 인스턴스를 중지합니다. 그렇지 않고 인스턴스 상태가 running이면 스크립트는 나머지 인스턴스를 계속 반복합니다.

   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. PrimaryPatchGroupTag 파라미터에 지정된 태그와 연결된 인스턴스에 대해 패치 작업이 완료됩니다. 이제 Emily는 SecondaryPatchGroupTag 파라미터에 지정된 태그와 연결된 인스턴스를 대상으로 하기 위해 실행서 콘텐츠의 모든 이전 작업을 복제합니다.

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. Emily는 완성된 스크립팅된 실행서 콘텐츠를 검토하고 대상 인스턴스와 동일한 AWS 계정 및 AWS 리전에 실행서를 생성합니다. 이제 프로덕션 환경에 구현하기 전 실행서를 테스트하여 자동화가 원하는 대로 작동하는지 확인할 준비가 되었습니다. 다음은 완성된 스크립팅된 실행서 콘텐츠입니다.

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":"..."
         }
      }
   ]
}
```

이 예에 사용되는 자동화 작업에 대한 자세한 내용은 Systems Manager Automation 작업 참조 섹션을 참조하세요.

추가 런북 예제

다음 예제 런북에서는 AWS Systems Manager Automation 작업을 사용하여 일반적인 배포, 문제 해결 및 유지 관리 태스크를 자동화하는 방법을 보여줍니다.

> **Note**
>
> 이 섹션의 예제 런북은 특정 운영 요구를 지원하기 위해 사용자 지정 런북을 생성하는 방법을 보여주기 위한 것입니다. 이러한 실행서는 프로덕션 환경에서 현재 그대로 사용할 수 없습니다. 하지만 자체적인 용도로 사용자 정의할 수 있습니다.

예제

- VPC 아키텍처 및 Microsoft Active Directory 도메인 컨트롤러 배포
- 최신 스냅샷에서 루트 볼륨 복원
- AMI 및 크로스 리전 복사본 생성

VPC 아키텍처 및 Microsoft Active Directory 도메인 컨트롤러 배포

효율성을 높이고 일반적인 작업을 표준화하기 위해 배포를 자동화하도록 선택할 수 있습니다. 이 기능은 여러 계정 및 AWS 리전에 동일한 아키텍처를 정기적으로 배포하는 경우에 유용합니다. 또한 아키텍처 배포를 자동화하면 수동으로 아키텍처를 배포할 때 발생할 수 있는 인적 오류의 가능성을 줄일 수 있습니다. AWS Systems Manager 이러한 경우 Automation 작업이 도움이 될 수 있습니다. Automation은 AWS Systems Manager의 기능입니다.

다음 예제 AWS Systems Manager 런북은 다음과 같은 작업을 수행합니다.

- 도메인 컨트롤러로 구성할 EC2 인스턴스를 시작할 때 사용할 Systems Manager Parameter Store를 통해 최신 Windows Server 2016 Amazon Machine Image(AMI)를 검색합니다. Parameter Store는 AWS Systems Manager의 기능입니다.
- `aws:executeAwsApi` Automation 작업으로 여러 AWS API 작업을 호출하여 VPC 아키텍처를 생성합니다. 도메인 컨트롤러 인스턴스는 프라이빗 서브넷에서 시작되고 NAT 게이트웨이를 사용하여 인터넷에 연결됩니다. 이렇게 하면 인스턴스의 SSM Agent가 필수 Systems Manager 엔드포인트에 액세스할 수 있습니다.
- `aws:waitForAwsResourceProperty` 자동화 작업을 사용하여 이전 작업으로 시작된 인스턴스가 AWS Systems Manager에 대해 `Online`인지 확인합니다.
- `aws:runCommand` 자동화 작업을 사용하여 Microsoft Active Directory 도메인 컨트롤러로 시작된 인스턴스를 구성합니다.

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[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 $_
              }
```

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 $_",
                "}"
              ]
            }
          }
        }
      ]
    }
```

최신 스냅샷에서 루트 볼륨 복원

루트 볼륨의 운영 체제는 여러 가지 이유로 손상될 수 있습니다. 예를 들어 패치 작업 후 손상된 커널 또는 레지스트리로 인해 인스턴스가 성공적으로 부팅되지 않을 수 있습니다. 패치 작업 전에 생성된 최신 스냅샷에서 루트 볼륨을 복원하는 것과 같은 일반적인 문제 해결 태스크를 자동화하면 가동 중지 시간을 줄이고 문제 해결 작업을 신속하게 수행할 수 있습니다. AWS Systems Manager 이러한 경우 Automation 작업이 도움이 될 수 있습니다. Automation은 AWS Systems Manager의 기능입니다.

다음 예제 AWS Systems Manager 런북은 다음과 같은 작업을 수행합니다.

- aws:executeAwsApi 자동화 작업을 사용하여 인스턴스의 루트 볼륨에서 세부 정보를 검색합니다.
- aws:executeScript 자동화 작업을 사용하여 루트 볼륨의 최신 스냅샷을 검색합니다.
- 루트 볼륨에 대한 스냅샷이 발견되면 aws:branch 자동화 작업을 사용하여 자동화를 계속합니다.

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"
                }
            }
        }
    }
```

AMI 및 크로스 리전 복사본 생성

인스턴스의 Amazon Machine Image(AMI) 생성은 백업 및 복구에 사용되는 일반적인 프로세스입니다. 재해 복구 아키텍처의 일부로 AMI를 다른 AWS 리전에 복사하도록 선택할 수도 있습니다. 일반적인 유지 관리 태스크를 자동화하면 문제에 장애 조치가 필요한 경우 가동 중지 시간을 줄일 수 있습니다. AWS Systems Manager 이러한 경우 Automation 작업이 도움이 될 수 있습니다. Automation은 AWS Systems Manager의 기능입니다.

다음 예제 AWS Systems Manager 런북은 다음과 같은 작업을 수행합니다.

- `aws:executeAwsApi` 자동화 작업을 사용하여 AMI를 생성합니다.
- `aws:waitForAwsResourceProperty` 자동화 작업을 사용하여 AMI의 가용성을 확인합니다.
- `aws:executeScript` 자동화 작업을 사용하여 AMI를 대상 리전으로 복사합니다.

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"
            }
        }
    }
```

```
    }
```

AWS 리소스를 채우는 입력 파라미터 생성

Systems Manager 매니저의 기능인 Automation은 입력 파라미터에 대해 정의한 리소스 타입과 일치하는 AWS Management Console의 AWS 리소스를 채웁니다. AWS 계정의 리소스 유형과 일치하는 리소스가 드롭다운 목록에 표시되어 선택할 수 있습니다. Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스, Amazon Simple Storage Service(Amazon S3) 버킷, AWS Identity and Access Management(IAM) 역할에 대한 입력 파라미터 유형을 정의할 수 있습니다. 지원되는 형식 정의와 일치하는 리소스를 찾는 데 사용되는 정규 표현식은 다음과 같습니다.

- `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/.*$`
- `List<AWS::IAM::Role::Arn>` - `^arn:(aws|aws-cn|aws-us-gov|aws-iso|aws-iso-b):iam::[0-9]{12}:role/.*$`

다음은 실행서 콘텐츠에 정의된 입력 파라미터 유형의 예제입니다.

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
      }
   ]
}
```

문서 빌더를 사용하여 런북 생성

AWS Systems Manager 퍼블릭 실행서가 AWS 리소스에서 수행하려는 모든 작업을 지원하지 않는 경우 자체 실행서를 생성할 수 있습니다. 사용자 정의 실행서를 생성하려면 적절한 자동화 작업을 사용하여 로컬 YAML 또는 JSON 형식 파일을 수동으로 생성할 수 있습니다. 또는 Systems Manager Automation 콘솔에서 문서 빌더를 사용하여 사용자 지정 런북을 빌드할 수도 있습니다.

문서 빌더를 사용하면 사용자 지정 런북에 자동화 작업을 추가하고 JSON 또는 YAML 구문을 사용하지 않고도 필요한 파라미터를 제공할 수 있습니다. 단계를 추가하고 실행서를 생성한 후에는 추가한 작업이 Systems Manager에서 자동화를 실행하는 데 사용할 수 있는 YAML 형식으로 변환됩니다.

실행서에서는 Markdown을 사용할 수 있습니다. 마크업 언어인 Markdown을 사용하면 실행서와 실행서 내의 개별 단계에 wiki 스타일의 설명을 추가할 수 있습니다. Markdown 사용에 대한 자세한 내용은 AWS에서 Markdown 사용을 참조하세요.

문서 빌더를 사용하여 런북 생성

시작하기 전 준비 사항

런북 내에서 사용할 수 있는 다양한 작업에 대해 읽어보는 것이 좋습니다. 자세한 내용은 Systems Manager Automation 작업 참조 단원을 참조하십시오.

문서 빌더를 사용하여 실행서를 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. Create automation(자동화 생성)을 선택합니다.
4. [이름(Name)]에 실행서를 설명하는 이름을 입력합니다.
5. [문서 설명(Document description)]에 실행서에 대한 Markdown 스타일 설명을 입력합니다. 실행서 사용 지침, 번호가 매겨진 단계 또는 기타 유형의 정보를 사용하여 실행서를 설명할 수 있도록 지침을 제공할 수 있습니다. 내용 서식 지정에 대한 자세한 내용은 기본 텍스트를 참조하십시오.

   > **Tip**
   >
   > Hide preview(미리보기 숨기기) 및 Show preview(미리보기 표시) 사이를 전환하여 작성 시 설명 내용이 어떤 형태인지 확인합니다.

6. (선택 사항) Assume role(역할 수임)에 사용자를 대신하여 작업을 수행할 서비스 역할의 이름 또는 ARN을 입력합니다. 역할을 지정하지 않으면 Automation은 자동화를 실행하는 사용자의 액세스 권한을 사용합니다.

   > **Important**
   >
   > `aws:executeScript` 작업을 사용하는 Amazon 소유가 아닌 실행서의 경우에는 역할을 지정해야 합니다. 자세한 설명은 실행서 사용 권한을 참조하세요.

7. (옵션) [출력(Outputs)]에 다른 프로세스에서 사용할 수 있도록 이 실행서의 자동화를 위한 출력을 입력합니다.

   예를 들어 실행서에서 새 AMI를 생성하는 경우 ["CreateImage.ImageId"]를 지정한 다음 이 출력을 사용하여 후속 자동화에서 새 인스턴스를 생성할 수 있습니다.

8. (선택 사항) Input parameter(입력 파라미터) 섹션을 확장하고 다음을 수행합니다.

   1. [파라미터 이름(Parameter name)]에 생성 중인 실행서 파라미터에 대한 설명이 포함된 이름을 입력합니다.
   2. 유형에서 파라미터 유형(예: `String` 또는 `MapList`)을 선택합니다.
   3. 필수에서 다음 중 하나를 수행합니다.
      - 런타임 시 이 실행서 파라미터의 값을 제공해야 하는 경우 [예(Yes)]를 선택합니다.

- 파라미터가 필수가 아닌 경우 [아니요(No)]를 선택하고 기본값에 기본 파라미터 값을 입력합니다(옵션).

4. [설명(Description)]에 실행서 파라미터에 대한 설명을 입력합니다.

> **Note**
>
> 실행서 파라미터를 더 추가하려면 [파라미터 추가(Add a parameter)]를 선택합니다. 실행서 파라미터를 제거하려면 [X](제거) 버튼을 선택합니다.

9. (옵션) [대상 유형(Target type)] 섹션을 확장하고 대상 유형을 선택하여 자동화를 실행할 수 있는 리소스의 종류를 정의합니다. 예를 들어 EC2 인스턴스에서 실행서를 사용하려면 `/AWS::EC2::Instance`을 선택합니다.

> **Note**
>
> '/' 값을 지정하면 모든 유형의 리소스에서 실행서를 실행할 수 있습니다. 유효한 리소스 유형 목록은 AWS CloudFormation 사용 설명서의 AWS 리소스 유형 참조를 참조하세요.

10. (옵션) [문서 태그(Document tags)] 섹션을 확장하고 실행서에 적용할 하나 이상의 태그 키-값 페어를 입력합니다. 태그를 사용하면 리소스를 쉽게 식별, 구성 및 검색할 수 있습니다. 자세한 내용은 Systems Manager 문서에 태그 지정 단원을 참조하십시오.
11. Step 1(1단계) 섹션에서 다음 정보를 제공합니다.

    - [단계 이름(Step name)]에 자동화의 첫 번째 단계에 대해 설명하는 이름을 입력합니다.
    - 작업 유형에서 이 단계에 사용할 작업 유형을 선택합니다.

      사용 가능한 작업 유형에 대한 목록과 자세한 내용은 Systems Manager Automation 작업 참조 섹션을 참조하세요.
    - 설명에 자동화 단계에 대한 설명을 입력합니다. Markdown을 사용하여 텍스트 서식을 지정할 수 있습니다.
    - 선택한 작업 유형에 따라 Step inputs(단계 입력) 섹션에 작업 유형에 필요한 입력을 입력합니다. 예를 들어 작업 `aws:approve`을 선택한 경우 `Approvers` 속성에 대해 값을 지정해야 합니다.

      단계 입력 필드에 대한 자세한 내용은 선택한 작업 유형에 대한 Systems Manager Automation 작업 참조의 항목을 참조하십시오. 예: `aws:executeStateMachine` - AWS Step Functions 상태 시스템을 실행합니다..

- (옵션) [추가 입력(Additional inputs)]에 실행서에 필요한 추가 입력 값을 입력합니다. 사용 가능한 입력 유형은 해당 단계에 대해 선택한 작업 유형에 따라 다릅니다 (일부 작업 유형에는 입력 값이 필요).

  > **Note**
  >
  > 입력을 추가하려면 Add optional input(선택적 입력 추가)을 선택합니다. 입력을 제거하려면 X(제거) 버튼을 선택합니다.

- (옵션) [출력(Outputs)]에 다른 프로세스에서 사용할 수 있도록 이 단계에 대한 출력을 입력합니다.

  > **Note**
  >
  > 모든 작업 유형에서출력을 사용할 수 없습니다.

- (옵션) [일반 속성(Common properties)] 섹션을 확장하고 모든 자동화 작업에 공통적인 작업의 속성을 지정합니다. 예를 들어 [시간 제한 초(Timeout seconds)]에 단계가 중지되기 전까지 실행될 수 있는 기간을 지정할 수 있도록 초 단위의 값을 입력합니다.

  자세한 내용은 모든 작업에서 공유하는 속성 단원을 참조하십시오.

> **Note**
>
> 단계를 더 추가하려면 [단계 추가(Add step)]를 선택하고 단계 생성 절차를 반복합니다. 단계를 제거하려면 [단계 제거(Remove step)]를 선택합니다.

12. [자동화 생성(Create automation)]을 선택하여 실행서를 저장합니다.

스크립트를 실행하는 런북 생성

다음 절차에서는 AWS Systems Manager Automation 콘솔에서 문서 빌더를 사용하여, 스크립트를 실행하는 사용자 지정 런북을 생성하는 방법을 보여줍니다.

생성하는 실행서의 첫 번째 단계에서는 스크립트를 실행하여 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스를 시작합니다. 두 번째 단계는 다른 스크립트를 실행하여 ok로 변경할 인스턴스 상태 확인을 모니터링합니다. 그러면 자동화에 대한 전체 상태가 Success로 보고됩니다.

시작하기 전 준비 사항

다음 단계를 완료했는지 확인합니다.

- 관리자 권한이 있는지, 아니면 AWS Identity and Access Management(IAM)에서 Systems Manager에 액세스할 수 있는 적절한 권한을 부여 받았는지 확인합니다.

  자세한 설명은 실행서에 대한 사용자 액세스 확인을 참조하세요.
- AWS 계정에 Automation을 위한 IAM 서비스 역할(수임 역할이라고도 함)이 있는지 확인합니다. 이 시연에서는 `aws:executeScript` 작업을 사용하기 때문에 이 역할이 필요합니다.

  이 역할을 생성하는 방법에 대한 상세 정보는 자동화에 대한 서비스 역할(수임 역할) 액세스 구성 섹션을 참조하세요.

  `aws:executeScript` 실행을 위한 IAM 서비스 역할 요구 사항에 대한 자세한 내용은 실행서 사용 권한 섹션을 참조하세요.
- EC2 인스턴스를 시작할 권한이 있는지 확인합니다.

  자세한 내용은 Amazon EC2 사용 설명서의 IAM 및 Amazon EC2를 참조하세요.

문서 빌더를 사용하여, 스크립트를 실행하는 사용자 지정 런북을 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. Create automation(자동화 생성)을 선택합니다.
4. [이름(Name)]에 실행서에 대한 설명이 포함된 이름을 입력합니다(예: **`LaunchInstanceAndCheckStatus`**).
5. (옵션) [문서 설명(Document description)]에서 Markdown을 사용하여 기본 텍스트를 이 실행서에 대한 설명으로 바꿉니다. 다음은 예입니다.

```
##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. [수임 역할(Assume role)]에 Automation용 IAM 서비스 역할(수임 역할)의 ARN을 **arn:aws:iam::111122223333:role/AutomationServiceRole** 형식으로 입력합니다. 111122223333을 사용자의 AWS 계정 ID로 대체합니다.

   사용자가 지정한 역할은 자동화를 시작하는 데 필요한 권한을 제공하는 데 사용됩니다.

   > **Important**
   >
   > aws:executeScript 작업을 사용하는 Amazon 소유가 아닌 실행서의 경우에는 역할을 지정해야 합니다. 자세한 설명은 실행서 사용 권한을 참조하세요.

7. 입력 파라미터를 확장하고 다음을 수행합니다.

   1. 파라미터 이름에 **imageId**를 입력합니다.
   2. 유형(Type)에서 **String**를 선택합니다.
   3. 필수에서 No를 선택합니다.
   4. 기본값에 다음을 입력합니다.

      ```
      {{ ssm:/aws/service/ami-amazon-linux-latest/amzn-ami-hvm-x86_64-gp2 }}
      ```

      > **Note**
      >
      > 이 값은 최신 Amazon Linux 1 Amazon Machine Image(AMI) ID를 사용하여 Amazon EC2 인스턴스를 시작합니다. 다른 AMI를 사용하려면 값을 AMI ID로 바꿉니다.

   5. 설명에 다음을 입력합니다.

      ```
      (Optional) The AMI ID to use for launching the instance. The default value uses
       the latest released Amazon Linux AMI ID.
      ```

8. 파라미터 추가를 선택하여 두 번째 파라미터 **tagValue**를 생성하고 다음 정보를 입력합니다.

1. 파라미터 이름에 **tagValue**를 입력합니다.
2. 유형(Type)에서 **String**를 선택합니다.
3. 필수에서 No를 선택합니다.
4. 기본값에 **LaunchedBySsmAutomation**를 입력합니다. 그러면 인스턴스에 태그 키 페어 값 Name:LaunchedBySsmAutomation이 추가됩니다.
5. 설명에 다음을 입력합니다.

```
(Optional) The tag value to add to the instance. The default value is
 LaunchedBySsmAutomation.
```

9. 파라미터 추가를 선택하여 세 번째 파라미터 **instanceType**를 생성하고 다음 정보를 입력합니다.
   1. 파라미터 이름에 **instanceType**를 입력합니다.
   2. 유형(Type)에서 **String**를 선택합니다.
   3. 필수에서 No를 선택합니다.
   4. 기본값에 **t2.micro**를 입력합니다.
   5. 파라미터 설명에 다음을 입력합니다.

```
(Optional) The instance type to use for the instance. The default value is
 t2.micro.
```

10. 대상 유형을 확장하고 **"/"**를 선택합니다.
11. (옵션) [문서 태그(Document tags)]를 확장하여 실행서에 리소스 태그를 적용합니다. 태그 키에 **Purpose**을 입력하고 태그 값에 **LaunchInstanceAndCheckState**를 입력합니다.
12. 1단계(Step 1) 섹션에서 다음 단계를 완료합니다.
    1. [단계 이름(Step name)]에 자동화의 첫 번째 단계에 대해 설명하는 단계 이름(**LaunchEc2Instance**)을 입력합니다.
    2. 작업 유형에서 Run a script(스크립트 실행)(**aws:executeScript**)를 선택합니다.
    3. 설명에 다음과 같은 자동화 단계에 대한 설명을 입력합니다.

```
**About This Step**
```

```
   This step first launches an EC2 instance using the ```aws:executeScript```
action and the provided script.
```

4. 입력을 확장합니다.
5. 런타임에서는 제공된 스크립트를 실행하는 데 사용할 런타임 언어를 선택합니다.
6. 핸들러에 **launch_instance**를 입력합니다. 이는 다음 스크립트에서 선언된 함수 이름입니다.

> **Note**
>
> PowerShell에는 필요하지 않습니다.

7. 스크립트에서 기본 내용을 다음과 같이 바꿉니다. 스크립트를 해당 런타임 값과 일치시켜야 합니다.

   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. Additional inputs(추가 입력)를 확장합니다.
9. Input name(입력 이름)에서 InputPayload를 선택합니다. Input value(입력 값)에 다음 YAML 데이터를 입력합니다.

```
image_id: "{{ imageId }}"
    tag_value: "{{ tagValue }}"
    instance_type: "{{ instanceType }}"
```

13. 출력을 확장하고 다음을 수행합니다.
    - 이름에 **payload**를 입력합니다.
    - Selector(선택기)에 **$.Payload**를 입력합니다.
    - 유형(Type)에서 StringMap를 선택합니다.
14. [단계 추가(Add step)]를 선택하여 실행서에 두 번째 단계를 추가합니다. 두 번째 단계는 1단계에서 시작된 인스턴스의 상태를 쿼리하고 반환된 상태가 ok가 될 때까지 기다립니다.
15. Step 2(2단계) 섹션에서 다음을 수행합니다.

1. [단계 이름(Step name)]에 자동화의 두 번째 단계에 대해 설명하는 이름(**`WaitForInstanceStatusOk`**)을 입력합니다.
2. 작업 유형에서 Run a script(스크립트 실행)(**`aws:executeScript`**)를 선택합니다.
3. 설명에 다음과 같은 자동화 단계에 대한 설명을 입력합니다.

```
**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. 런타임에서는 제공된 스크립트를 실행하는 데 사용되는 런타임 언어를 선택합니다.
5. 핸들러에 **`poll_instance`**를 입력합니다. 이는 다음 스크립트에서 선언된 함수 이름입니다.

> **Note**
>
> PowerShell에는 필요하지 않습니다.

6. 스크립트에서 기본 내용을 다음과 같이 바꿉니다. 스크립트를 해당 런타임 값과 일치시켜야 합니다.

   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. Additional inputs(추가 입력)를 확장합니다.
8. Input name(입력 이름)에서 InputPayload를 선택합니다. Input value(입력 값)에 다음을 입력합니다.

```
{{ LaunchEc2Instance.payload }}
```

16. [자동화 생성(Create automation)]을 선택하여 실행서를 저장합니다.

런북에서 스크립트 사용

Automation 실행서는 자동화의 일부로 스크립트 실행을 지원합니다. Automation은 AWS Systems Manager의 기능입니다. 실행서를 사용하면 스크립트를 실행하기 위한 별도의 컴퓨팅 환경을 생성하지 않고도 AWS에서 직접 스크립트를 실행할 수 있습니다. 실행서는 승인과 같은 다른 자동화 단계 유형과 함께 스크립트 단계를 실행할 수 있기 때문에, 중요하거나 모호한 상황에 수동으로 개입할 수 있습니다. 실행서의 `aws:executeScript` 작업 출력을 Amazon CloudWatch Logs로 보낼 수 있습니다. 자세한 내용은 CloudWatch Logs로 Automation 작업 출력 로깅 단원을 참조하십시오.

실행서 사용 권한

실행서를 사용하려면 Systems Manager에서 AWS Identity and Access Management(IAM) 역할의 권한을 사용해야 합니다. 자동화에서 사용할 역할의 권한을 결정하는 데 사용하는 방법은 몇 가지 요소와 단계에서 `aws:executeScript` 작업을 사용하는지 여부에 따라 달라집니다.

`aws:executeScript`를 사용하지 않는 실행서의 경우 Automation은 두 가지 권한 소스 중 하나를 사용합니다.

- 실행서에 지정되거나 파라미터로 전달되는 IAM 서비스 역할 또는 수임 역할의 권한입니다.
- IAM 서비스 역할이 지정되지 않은 경우에는 자동화를 시작한 사용자의 권한입니다.

그러나 실행서의 단계에 `aws:executeScript` 작업이 포함되어 있는 경우 작업에 대해 지정된 Python 또는 PowerShell 스크립트가 AWS API 작업을 호출하는 경우 항상 IAM 서비스 역할(수임 역할)이 필요합니다. 자동화는 다음 순서로 이 역할을 확인합니다.

- 실행서에 지정되거나 파라미터로 전달되는 IAM 서비스 역할 또는 수임 역할의 권한입니다.
- 역할이 없는 경우 자동화는 권한 없이 `aws:executeScript`에 대해 지정된 Python 또는 PowerShell 스크립트의 실행을 시도합니다. 스크립트가 AWS API 작업(예: Amazon EC2 `CreateImage` 작업)을 호출하거나 AWS 리소스(예: EC2 인스턴스)에서 작업을 시도하는 경우 스크립트를 포함하는 단계가 실패하고 Systems Manager가 오류를 보고하는 오류 메시지를 반환합니다.

실행서에 스크립트 추가

실행서의 단계의 일부로 스크립트를 인라인으로 포함시켜 실행서에 스크립트를 추가할 수 있습니다. 로컬 시스템에서 스크립트를 업로드하거나 스크립트가 위치하는 Amazon Simple Storage Service(Amazon S3) 버킷을 지정하여 실행서에 스크립트를 첨부할 수도 있습니다. 스크립트를 실행하는 단계가 완료되면 스크립트의 출력을 JSON 객체로 사용할 수 있습니다. 그런 다음 이를 실행서의 후속 단계에서 입력으로 사용할 수 있습니다.

실행서에 대한 스크립트 제약 조건

실행서는 첨부 파일을 5개로 제한합니다. 스크립트는 Python 스크립트(.py) 또는 PowerShell Core 스크립트(.ps1)의 형태이거나 .zip 파일 내의 내용으로 첨부될 수 있습니다.

런북에서 조건문 사용

기본적으로 실행서의 `mainSteps` 섹션에서 정의하는 단계는 순차적으로 실행됩니다. 한 작업이 완료된 후에는 `mainSteps` 섹션에 지정된 다음 작업이 시작됩니다. 또한 작업 실행에 실패할 경우 기본적으로 전체 자동화가 실패합니다. 이 섹션에 설명된 `aws:branch` 자동화 작업 및 실행서 옵션을 사용하여 조건부 분기를 수행하는 자동화를 생성할 수 있습니다. 다시 말해서 단계 완료 시 변경 내용에 동적으로 응답하거나 다양한 선택 항목을 평가한 후 다른 단계로 이동하는 자동화를 생성할 수 있습니다. 동적 자동화를 생성하는 데 사용할 수 있는 옵션 목록은 다음과 같습니다.

- **`aws:branch`**: 이 자동화 작업을 통해 한 단계에서 여러 선택 항목을 평가한 다음 평가 결과에 따라 실행서의 다른 단계로 이동하는 동적 자동화를 생성할 수 있습니다.
- **`nextStep`**: 이 옵션은 한 단계를 성공적으로 완료한 후 처리할 자동화의 단계를 지정합니다.
- **`isEnd`**: 이 옵션은 특정 단계 종료 시 자동화를 중지합니다. 이 옵션의 기본값은 false입니다.
- **`isCritical`**: 이 옵션은 자동화의 성공적 완료에 대해 단계를 심각으로 지정합니다. 이 지정이 있는 단계가 실패하면 Automation은 자동화의 최종 상태를 `Failed`로 보고합니다. 이 옵션의 기본값은 `true`입니다.
- **`onFailure`**: 이 옵션은 실패 시 자동화가 중지되어야 하는지, 계속되어야 하는지 또는 다른 단계로 이동해야 하는지를 나타냅니다. 이 옵션의 기본값은 중단입니다.

다음 섹션에서는 `aws:branch` 자동화 작업을 설명합니다. `nextStep`, `isEnd`, `isCritical` 및 `onFailure` 옵션에 대한 자세한 내용은 예제 aws:branch 실행서 섹션을 참조하세요.

`aws:branch` 작업 수행

`aws:branch` 작업은 자동화에 대한 가장 동적인 조건부 분기 옵션을 제공합니다. 앞에서 언급한 바와 같이, 이 작업을 사용하면 자동화가 한 단계에서 여러 조건을 평가한 다음 해당 평가의 결과에 따라 새 단계로 이동할 수 있습니다. `aws:branch` 작업은 프로그래밍의 `IF-ELIF-ELSE` 문처럼 작동합니다.

다음은 `aws:branch` 단계의 YAML 예제입니다.

```
- name: ChooseOSforCommands
  action: aws:branch
  inputs:
```

```
    Choices:
    - NextStep: runPowerShellCommand
      Variable: "{{GetInstance.platform}}"
      StringEquals: Windows
    - NextStep: runShellCommand
      Variable: "{{GetInstance.platform}}"
      StringEquals: Linux
    Default:
      PostProcessing
```

단계에 대한 aws:branch 작업을 지정할 경우 해당 자동화에서 평가해야 하는 Choices를 지정할 수 있습니다. 이 자동화는 실행서의 Parameters 섹션에 지정된 파라미터의 값에 따라 Choices를 평가할 수 있습니다. 이 자동화는 이전 단계의 출력에 따라 Choices를 평가할 수도 있습니다.

이 자동화는 부울 식을 사용하여 각 선택을 평가합니다. 평가에서 첫 번째 선택 항목이 true인 것으로 확인되면 자동화는 해당 선택 항목에 지정된 단계로 이동합니다. 평가에서 첫 번째 선택 항목이 false인 것으로 확인되면 자동화는 다음 선택 항목을 평가합니다. 단계에 세 개 이상의 Choices가 포함된 경우 자동화는 true인 선택 항목을 평가할 때까지 순서대로 각 선택 항목을 평가합니다. 그런 다음 이 워크플로는 true인 선택 항목에 지정된 단계로 이동합니다.

Choices 중에 true인 항목이 없는 경우 자동화는 단계에 Default 값이 포함되었는지 확인합니다. Default 값은 선택 항목 중에 true인 항목이 없는 경우 자동화에서 이동해야 하는 단계를 정의합니다. 단계에 대한 Default 값이 지정되지 않은 경우 자동화는 실행서의 다음 단계를 처리합니다.

다음은 chooseOSfromParameter라는 YAML의 aws:branch 단계입니다. 이 단계에는 NextStep: runWindowsCommand와 NextStep: runLinuxCommand라는 두 Choices가 포함되어 있습니다. 이 자동화는 이러한 Choices를 평가하여 해당 운영 체제에 대해 실행할 명령을 결정합니다. 각 선택 항목에 대한 Variable은 {{OSName}}을 사용하며, 이것은 실행서 작성자가 실행서의 Parameters 섹션에 정의한 파라미터입니다.

```
mainSteps:
- name: chooseOSfromParameter
  action: aws:branch
  inputs:
    Choices:
    - NextStep: runWindowsCommand
      Variable: "{{OSName}}"
      StringEquals: Windows
    - NextStep: runLinuxCommand
      Variable: "{{OSName}}"
      StringEquals: Linux
```

다음은 chooseOSfromOutput이라는 YAML의 aws:branch 단계입니다. 이 단계에는 NextStep: runPowerShellCommand와 NextStep: runShellCommand라는 두 Choices가 포함되어 있습니다. 이 자동화는 이러한 Choices를 평가하여 해당 운영 체제에 대해 실행할 명령을 결정합니다. 각 선택 항목의 Variable은 {{GetInstance.platform}}을 사용하며, 이것은 실행서의 이전 단계에서 얻은 출력입니다. 이 예제에는 Default라는 옵션도 포함되어 있습니다. 자동화에서 두 Choices를 평가했지만 어느 것도 true가 아니면 이 자동화는 PostProcessing이라는 단계로 이동합니다.

```
mainSteps:
- name: chooseOSfromOutput
  action: aws:branch
  inputs:
    Choices:
    - NextStep: runPowerShellCommand
      Variable: "{{GetInstance.platform}}"
      StringEquals: Windows
    - NextStep: runShellCommand
      Variable: "{{GetInstance.platform}}"
      StringEquals: Linux
    Default:
      PostProcessing
```

실행서에서 **aws:branch** 단계 생성

실행서에서 aws:branch 단계를 생성하는 경우 자동화에서 다음으로 이동해야 할 단계를 결정하기 위해 평가해야 하는 Choices를 정의합니다. 앞에서 언급한 바와 같이, Choices는 부울 식을 사용하여 평가됩니다. 각 선택 항목은 다음 옵션을 정의해야 합니다.

- NextStep: 지정된 선택 항목이 true인 경우 처리할 실행서의 다음 단계입니다.
- Variable: 런북의 Parameters 섹션에 정의된 파라미터(Variables 섹션에 정의된 변수)의 이름을 지정하거나 이전 단계에서 얻은 출력 객체를 지정합니다.

  다음 형식을 사용하여 변수 값을 지정합니다.

  `Variable: "{{variable name}}"`

  다음 형식을 사용하여 파라미터 값을 지정합니다.

  `Variable: "{{parameter name}}"`

다음 형식을 사용하여 출력 객체 변수를 지정합니다.

```
Variable: "{{previousStepName.outputName}}"
```

> **Note**
>
> 출력 변수 생성은 다음 섹션인 출력 변수 생성 정보에서 자세히 설명합니다.

- Operation: 선택 항목을 평가하는 데 사용되는 기준으로, 예를 들면 `StringEquals: Linux`입니다. `aws:branch` 작업에서는 다음 연산을 지원합니다.

  문자열 연산

  - StringEquals
  - EqualsIgnoreCase
  - StartsWith
  - EndsWith
  - 포함

  수치 연산

  - NumericEquals
  - NumericGreater
  - NumericLesser
  - NumericGreaterOrEquals
  - NumericLesser
  - NumericLesserOrEquals

  부울 연산

  - BooleanEquals

  > **Important**
  >
  > 실행서를 생성하면 시스템에서 실행서의 각 작업을 검증합니다. 작업이 지원되지 않는 경우 실행서를 생성할 때 오류가 발생합니다.

- Default: `Choices` 중 어떤 것도 `true`가 아니면 자동화에서 이동해야 할 폴백 단계를 지정합니다.

> **Note**
>
> Default 값을 지정하지 않으려면 isEnd 옵션을 지정할 수 있습니다. Choices 중 어느 것도 true가 아니며 Default 값을 지정하지 않은 경우 자동화는 단계의 끝에서 중지됩니다.

다음 템플릿을 사용하여 실행서에서 aws:branch 단계를 생성합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

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"
        }
    }
  ]
}
```

출력 변수 생성 정보

이전 단계의 출력을 참조하는 aws:branch 선택 항목을 생성하려면 이전 단계의 이름과 출력 필드의 이름을 식별해야 합니다. 그러고 나서 다음 형식을 사용하여 단계의 이름과 필드를 결합합니다.

`Variable: "{{previousStepName.outputName}}"`

예를 들어 다음 예에서 첫 번째 단계의 이름은 GetInstance입니다. 그 다음, outputs 아래에 platform이라는 필드가 있습니다. 두 번째 단계(ChooseOSforCommands)에서 작성자는 플랫폼 필드의 출력을 변수로 참조하려고 합니다. 변수를 생성하려면 단계 이름(GetInstance)과 출력 필드 이름(platform)을 결합하여 `Variable: "{{GetInstance.platform}}"`을 생성하면 됩니다.

```
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
```

다음은 이전 단계 및 출력에서 *"Variable": "{{ describeInstance.Platform }}"*이 어떻게 생성되는지 보여주는 예시입니다.

```
- 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
```

예제 **aws:branch** 실행서

다음은 aws:branch를 사용하는 몇 가지 예제 실행서입니다.

예제 1: **aws:branch**를 출력 변수와 함께 사용하여 운영 체제 유형에 따라 명령 실행

이 예의 첫 번째 단계(GetInstance)에서 실행서 작성자는 aws:executeAwsApi 작업을 사용하여 ssm DescribeInstanceInformation API 작업을 호출합니다. 작성자는 이 작업을 사용하여 인스턴스에서 사용 중인 운영 체제 유형을 확인합니다. aws:executeAwsApi 작업은 인스턴스 ID 및 플랫폼 유형을 출력합니다.

두 번째 단계(ChooseOSforCommands)에서 작성자는 aws:branch 작업과 두Choices인 NextStep: runPowerShellCommand 및 NextStep: runShellCommand를 함께 사용합니다. 이 자동화는 이전 단계(Variable: "{{GetInstance.platform}}")의 출력을 사용하여 인스턴스의 운영 체제를 평가합니다. 이 자동화는 지정된 운영 체제에 대한 단계로 이동합니다.

```
---
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
```

예 2: **aws:branch**를 파라미터 변수와 함께 사용하여 운영 체제 유형에 따라 명령 실행

실행서 작성자는 실행서의 시작 부분에 나오는 parameters 섹션에서 여러 파라미터 옵션을 정의합니다. 파라미터 하나의 이름은 OperatingSystemName입니다. 첫 번째 단계(ChooseOS)에서 작성자는 aws:branch 작업과 두Choices인 NextStep: runWindowsCommand 및 NextStep: runLinuxCommand를 함께 사용합니다. 이러한 Choices의 변수는 파라미터 섹션에 지정된 파라미터 옵션(Variable: "{{OperatingSystemName}}")을 참조합니다. 사용자는 이 실행서를 실행할 경우 런타임 시 OperatingSystemName에 대한 값을 지정합니다. 이 자동화는 Choices 평가 중에 런타임 파라미터를 사용합니다. 이 자동화는 OperatingSystemName에 지정된 런타임 파라미터에 따라 지정된 운영 체제에 해당하는 단계로 이동합니다.

```
---
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
```

연산자를 사용하여 복합 분기 자동화 생성

aws:branch 단계에서 And, Or 및 Not 연산자를 사용하여 복합 문기 자동화를 생성할 수 있습니다.

'And' 연산자

선택 항목에 대해 여러 변수가 true가 되도록 하려는 경우 And 연산자를 사용합니다. 다음 예제에서는 첫 번째 선택 항목이 인스턴스가 running이고 Windows 운영 체제를 사용하는지 여부를 평가합니다. 이들 변수 모두를 평가한 결과가 true이면 이 자동화는 runPowerShellCommand 단계로 이동합니다. 변수 중 하나 이상이 false이면 이 자동화는 두 번째 선택 항목에 대한 변수를 평가합니다.

```
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' 연산자

선택 항목에 대해 여러 변수 중 하나라도 true가 되도록 하려는 경우 Or 연산자를 사용합니다. 다음 예제에서 첫 번째 선택 항목은 파라미터 문자열이 Windows이고 AWS Lambda 단계의 출력이 true인지 여부를 평가합니다. 평가 결과 이들 변수 중 하나가 true인 것으로 확인되면 이 자동화는 RunPowerShellCommand 단계로 이동합니다. 두 변수가 모두 false이면 이 자동화는 두 번째 선택 항목에 대한 변수를 평가합니다.

```
- Or:
  - Variable: "{{parameter1}}"
    StringEquals: Windows
  - Variable: "{{BooleanParam1}}"
    BooleanEquals: true
  NextStep: RunPowershellCommand
- Or:
  - Variable: "{{parameter2}}"
    StringEquals: Linux
  - Variable: "{{BooleanParam2}}"
    BooleanEquals: true
  NextStep: RunShellScript
```

'Not' 연산자

변수가 true가 아닐 때 정의된 단계로 이동하려면 Not 연산자를 사용합니다. 다음 예제에서 첫 번째 선택 항목은 파라미터 문자열이 Not Linux인지 여부를 평가합니다. 평가 결과 이 변수가 Linux가 아니면 이 자동화는 sleep2 단계로 이동합니다. 첫 번째 선택 항목의 평가에서 Linux인 것으로 확인되면 이 자동화는 다음 선택 항목을 평가합니다.

```
mainSteps:
- name: switch
  action: aws:branch
  inputs:
    Choices:
    - NextStep: sleep2
      Not:
        Variable: "{{testParam}}"
        StringEquals: Linux
    - NextStep: sleep1
      Variable: "{{testParam}}"
      StringEquals: Windows
    Default:
      sleep3
```

조건 옵션 사용 방법의 예

이 섹션에는 실행서에서 동적 옵션 사용 방법의 다양한 예가 포함되어 있습니다. 이 섹션에 나오는 각 예는 다음 실행서를 확장합니다. 이 실행서에는 2개의 작업이 있습니다. 첫 번째 작업의 이름은 InstallMsiPackage입니다. aws:runCommand 작업을 사용하여 Windows Server 인스턴스에 애플리케이션을 설치합니다. 두 번째 작업의 이름은 TestInstall입니다. 이 작업은

aws:invokeLambdaFunction 작업을 사용하여 애플리케이션이 성공적으로 설치된 경우 설치된 애플리케이션의 테스트를 수행합니다. 1단계는 onFailure: Abort를 지정합니다. 따라서 애플리케이션이 성공적으로 설치되지 않은 경우 2단계 이전에 자동화가 중지됩니다.

예 1: 2개의 선형 작업이 있는 실행서

```
---
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
...
```

onFailure 옵션을 사용하여 다양한 단계로 이동하는 동적 자동화 생성

다음 예에서는 onFailure: step:*step name*, nextStep 및 isEnd 옵션을 사용하여 동적 자동화를 생성합니다. 이 예제에서 InstallMsiPackage 작업이 실패하면 자동화가 PostFailure(onFailure: step:PostFailure)라는 작업으로 이동하여 설치가 실패할 경우 특정 작업을 수행하는 AWS Lambda 함수를 실행합니다. 설치에 성공하면 이 자동화는 TestInstall 작업(nextStep: TestInstall)으로 이동합니다. 두 단계 중 하나가 완료될 때 자동화가 끝나도록 두 TestInstall 및 PostFailure 단계에 모두 isEnd 옵션(isEnd: true)이 사용됩니다.

> **Note**
>
> mainSteps 섹션의 마지막 단계에서 isEnd 옵션을 사용하는 것은 선택 사항입니다. 마지막 단계가 다른 단계로 건너뛰지 않으면 마지막 단계에서 작업을 실행한 후 자동화가 중지됩니다.

예 2: 다양한 단계로 이동하는 동적 자동화

```
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**
>
> 실행서를 처리하기 전에 시스템은 실행서가 무한 루프를 생성하지 않는지 확인합니다. 무한 루프가 감지되면 자동화는 오류와 순환 추적을 반환하여 루프를 생성하는 단계를 표시합니다.

중요 단계를 정의하는 동적 자동화 생성

단계가 자동화의 전체 성공에 중요한지를 지정할 수 있습니다. 중요 단계가 실패하면 하나 이상의 단계가 성공적으로 실행되더라도 Automation은 자동화 상태를 `Failed`로 보고합니다. 다음 예에서 사용자는 InstallMsiPackage 단계가 실패하는 경우(`onFailure: step:VerifyDependencies`) VerifyDependencies 단계를 식별합니다. 사용자는 `InstallMsiPackage` 단계가 중요하지 않다고 지정합니다(`isCritical: false`). 이 예에서는 애플리케이션 설치에 실패한 경우 자동화가 `VerifyDependencies` 단계를 처리하여 하나 이상의 종속성이 누락되어 결과적으로 애플리케이션 설치에 실패했는지 여부를 확인합니다.

예제 3: 자동화의 중요 단계 정의

```
---
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
...
```

작업 출력을 입력으로 사용

몇 가지 자동화 작업은 사전 정의된 출력을 반환합니다. {{*stepName*.*outputName*}} 형식을 사용하여 이러한 출력을 런북의 이후 단계에 입력으로 전달할 수 있습니다. 런북에서 다양한 자동화 작업

의 출력을 정의할 수 있습니다. 이를 통해 스크립트를 실행하거나 다른 AWS 서비스를 위한 API 작업을 한 번 호출하여 이후 작업에서 해당 값을 입력으로 재사용할 수 있습니다. 런북의 파라미터 유형은 정적입니다. 즉, 파라미터 유형을 정의한 후에는 변경할 수 없습니다. 단계 출력을 정의하려면 다음 필드를 제공하십시오.

- 이름: (필수) 이후 단계에서 출력 값을 참조하는 데 사용되는 출력 이름입니다.
- 선택기: (필수) 출력 값을 결정하는 데 사용되는 JSONPath 표현식입니다.
- 유형: (선택 사항) 선택기 필드에서 반환되는 값의 데이터 유형입니다. 유효한 유형 값은`String`, `Integer`, `Boolean`, `StringList`, `StringMap`, `MapList`입니다. 기본 값은 `String`입니다.

출력 값이 지정한 데이터 유형과 일치하지 않는 경우 Automation은 데이터 유형을 변환하려고 시도합니다. 예를 들어, 반환된 값이 `Integer`이지만 지정된 `Type`은 `String`인 경우 최종 출력 값은 `String` 값입니다. 다음과 같은 유형의 반환이 지원됩니다.

- `String` 값을 `StringList`, `Integer` 및 `Boolean`로 변환할 수 있습니다.
- `Integer` 값을 `String`, `StringList`로 변환할 수 있습니다.
- `Boolean` 값을 `String`, `StringList`로 변환할 수 있습니다.
- 한 가지 요소가 포함된 `StringList`, `IntegerList` 또는 `BooleanList` 값은 `String`, `Integer` 또는 `Boolean`으로 변환할 수 없습니다.

자동화 작업에 파라미터 또는 출력을 사용하면 데이터 유형을 작업의 입력 내에서 동적으로 변경할 수 없습니다.

다음은 작업 출력을 정의하고 해당 값을 이후 작업의 입력으로 참조하는 방법을 보여주는 예제 런북입니다. 이 런북은 다음 작업을 수행합니다.

- `aws:executeAwsApi` 작업을 사용하여 특정 Windows Server 2016 AMI의 이름을 가져오도록 Amazon EC2 DescribeImages API 작업을 호출합니다. 이미지 ID를 `ImageId`로 출력합니다.
- `aws:executeAwsApi` 작업을 사용하여 이전 단계의 `ImageId`를 사용하는 인스턴스 하나를 시작하도록 Amazon EC2 RunInstances API 작업을 호출합니다. 인스턴스 ID를 `InstanceId`로 출력합니다.
- `aws:waitForAwsResourceProperty` 작업을 사용하여 인스턴스가 `running` 상태에 도달할 때까지 기다리도록 Amazon EC2 DescribeInstanceStatus API 작업을 폴링합니다. 작업 제한 시간은 60초입니다. 폴링 후 60초가 경과되었지만 인스턴스가 `running` 상태에 도달하지 못하면 단계가 시간 초과됩니다.

- aws:assertAwsResourceProperty 작업으로 Amazon EC2 DescribeInstanceStatus API 작업을 호출하여 인스턴스가 running 상태에 있음을 어설션합니다. 인스턴스 상태가 running이 아니면 단계가 실패합니다.

```
---
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"
...
```

각각의 이전에 설명된 자동화 작업을 통해 서비스 네임스페이스, API 작업 이름, 입력 파라미터 및 출력 파라미터를 지정하여 특정 API 작업을 호출할 수 있습니다. 입력은 선택하는 API 작업에 의해 정의됩니다. 다음 [서비스 참조(Services Reference)] 페이지의 왼쪽 탐색 영역에서 서비스를 선택하여 API 작업(메서드라고도 함)을 볼 수 있습니다. 호출할 서비스에 대한 클라이언트 섹션에서 메서드를 선택합니다. 예를 들어 Amazon Relational Database Service(Amazon RDS)에 대한 모든 API 작업(메서드)이 Amazon RDS 메서드 페이지에 나열됩니다.

다음 위치에서 각 자동화 작업에 대한 스키마를 볼 수 있습니다.

- `aws:assertAwsResourceProperty` - AWS 리소스 상태 또는 이벤트 상태 어설션
- `aws:executeAwsApi` - AWS API 작업 호출 및 실행
- `aws:waitForAwsResourceProperty` - AWS 리소스 속성 대기

스키마에는 각 작업을 사용하기 위한 필수 필드에 대한 설명이 포함되어 있습니다.

Selector/PropertySelector 필드 사용

각 Automation 작업에서는 출력 `Selector`(`aws:executeAwsApi`의 경우) 또는 `PropertySelector`(`aws:assertAwsResourceProperty` 및 `aws:waitForAwsResourceProperty`의 경우)를 지정해야 합니다. 이러한 필드는 AWS API 작업의 JSON 응답을 처리하는 데 사용됩니다. 이러한 필드는 JSONPath 구문에 사용됩니다.

다음은 `aws:executeAwsAPi` 작업과 관련하여 이 개념을 설명하는 데 도움이 되는 예제입니다.

```
---
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"
...
```

이 자동화는 `aws:executeAwsApi` 단계인 `getImageId`에서 `DescribeImages` API 작업을 호출하고 `ec2`의 응답을 수신합니다. 그런 다음 이 자동화는 `Selector - "$.Images[0].ImageId"`를 API 응답에 적용하고 선택한 값을 `ImageId` 변수에 할당합니다. 동일한 자동화의 다른 단계에서는 `"{{ getImageId.ImageId }}"`를 지정하여 `ImageId`의 값을 사용할 수 있습니다.

다음은 `aws:waitForAwsResourceProperty` 작업과 관련하여 이 개념을 설명하는 데 도움이 되는 예제입니다.

```
---
- 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
...
```

이 자동화는 `aws:waitForAwsResourceProperty` 단계인 `waitUntilInstanceStateRunning`에서 `DescribeInstanceStatus` API 작업을 간접적으로 호출하고 ec2의 응답을 수신합니다. 그런 다음 이 자동화는 `PropertySelector - "$.InstanceStatuses[0].InstanceState.Name"`을 응답에 적용하여 지정된 반환 값이 `DesiredValues` 목록의 값(이 경우 `running`)과 일치하는지 확인합니다. 이 단계는 응답이 인스턴스 상태 `running`을 반환할 때까지 이 프로세스를 반복합니다.

런북에서 JSONPath 사용

JSONPath 식은 “$.”로 시작하는 문자열로 JSON 요소 내에서 하나 이상의 구성 요소를 선택하는 데 사용됩니다. 다음 목록에는 Systems Manager Automation에서 지원되는 JSONPath 연산자에 대한 정보가 포함되어 있습니다.

- 점으로 표기된 하위 객체(.): JSON 객체와 함께 사용합니다. 이 연산자는 특정 키의 값을 선택합니다.
- 상세 검색(..): JSON 요소와 함께 사용합니다. 이 연산자는 수준별로 JSON 요소 수준을 검색하고 특정 키가 지정된 값의 목록을 선택합니다. 이 연산자의 반환 유형은 항상 JSON 배열입니다. 자동화 작업 출력 유형의 컨텍스트에서 이 연산자는 StringList 또는 MapList일 수 있습니다.
- 배열-인덱스([ ]): JSON 배열과 함께 사용합니다. 이 연산자는 특정 인덱스의 값을 가져옵니다.
- 필터([?(***expression***)]): JSON 배열과 함께 사용합니다. 이 연산자는 필터 표현식에 정의된 기준과 일치하는 JSON 배열 값을 필터링합니다. 필터 표현식에는 이 연산자(==, !=, >, <, >= 또는 <=)만 사용할 수 있습니다. 여러 필터 표현식을 AND(&&) 또는 OR(||)과 결합하는 것은 지원되지 않습니다. 이 연산자의 반환 유형은 항상 JSON 배열입니다.

JSONPath 연산자를 더 명확히 이해하려면 ec2 `DescribeInstances` API 작업의 다음 JSON 응답을 검토합니다. 이 응답 아래에는 다양한 JSONPath 식을 `DescribeInstances` API 작업의 응답에 적용하여 다양한 결과를 표시하는 여러 예제가 나와 있습니다.

```
{
    "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 예제 1: JSON 응답에서 특정 문자열 가져오기

```
JSONPath:
$.Reservations[0].Instances[0].ImageId

Returns:
"ami-12345678"

Type: String
```

JSONPath 예제 2: JSON 응답에서 특정 부울 가져오기

```
JSONPath:
$.Reservations[0].Instances[0].BlockDeviceMappings[0].Ebs.DeleteOnTermination

Returns:
true

Type: Boolean
```

JSONPath 예제 3: JSON 응답에서 특정 정수 가져오기

```
JSONPath:
$.Reservations[0].Instances[0].State.Code

Returns:
16

Type: Integer
```

JSONPath 예제 4: JSON 응답을 상세 검색한 다음, VolumeId에 대한 모든 값을 StringList로 가져오기

```
JSONPath:
$.Reservations..BlockDeviceMappings..VolumeId

Returns:
```

```
[
   "vol-000000000000",
   "vol-111111111111"
]

Type: StringList
```

JSONPath 예제 5: 특정 BlockDeviceMappings 객체를 StringMap으로 가져오기

```
JSONPath:
$.Reservations[0].Instances[0].BlockDeviceMappings[0]

Returns:
{
   "Ebs" : {
      "DeleteOnTermination" : true,
      "Status" : "attached",
      "VolumeId" : "vol-000000000000"
   },
   "DeviceName" : "/dev/xvda"
}

Type: StringMap
```

JSONPath 예제 6: JSON 응답을 상세 검색한 후 모든 State 객체를 MapList로 가져오기

```
JSONPath:
$.Reservations..Instances..State

Returns:
[
   {
      "Code" : 16,
      "Name" : "running"
   },
   {
      "Code" : 80,
      "Name" : "stopped"
   }
]

Type: MapList
```

JSONPath 예제 7: **running** 상태에 있는 인스턴스에 대한 필터링

```
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 예제 8: **running** 상태에 있지 않은 **ImageId** 인스턴스의 반환

```
JSONPath:
$.Reservations..Instances[?(@.State.Name != 'running')].ImageId

Returns:
[
  "ami-12345678"
]

Type: StringList | String
```

Automation을 위한 웹후크 통합 생성

자동화 중 웹후크를 사용하여 메시지를 전송하려면 통합을 생성합니다. 실행서에서 `aws:invokeWebhook` 작업을 사용하여 자동화 중 통합을 호출할 수 있습니다. 웹후크를 아직 생성하지 않은 경우 통합을 위한 웹후크 생성 섹션을 참조하세요. `aws:invokeWebhook` 작업에 대한 자세한 내용은 `aws:invokeWebhook` - Automation 웹후크 통합 호출 섹션을 참조하세요.

다음 절차에서 볼 수 있듯이 Systems Manager 자동화 콘솔이나 선호하는 명령줄 도구를 사용하여 통합을 생성할 수 있습니다.

통합 생성(콘솔)

Automation을 위한 통합 생성(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 왼쪽 탐색 창에서 Automation을 선택합니다.
3. 통합(Integrations) 탭을 선택합니다.
4. 통합 추가(Add integration)를 선택하고 웹후크(Webhook)를 선택합니다.
5. 통합에 포함할 필수 값과 선택적 값을 입력합니다.
6. 추가(Add)를 선택하여 통합을 생성합니다.

통합 생성(명령줄)

명령줄 도구를 사용하여 통합을 생성하려면 통합에 필요한 `SecureString` 파라미터를 생성해야 합니다. Automation은 Systems Manager의 기능인 Parameter Store에서 예약된 네임스페이스를 사용하여 통합에 대한 정보를 저장합니다. AWS Management Console을 사용하여 통합을 생성하는 경우 Automation이 이 프로세스를 자동으로 처리합니다. 네임스페이스 다음에 생성하려는 통합 유형을 지정하고 통합 이름을 지정해야 합니다. 현재 Automation은 `webhook` 유형 통합을 지원합니다.

`webhook` 유형 통합에 지원되는 필드는 다음과 같습니다.

- 설명
- 헤더
- payload
- URL

시작하기 전 준비 사항

아직 하지 않은 경우 AWS Command Line Interface(AWS CLI) 또는 AWS Tools for PowerShell을 설치하고 구성합니다. 자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트 및 AWS Tools for PowerShell 설치를 참조하세요.

Automation을 위한 통합 생성(명령줄)

- 다음 명령을 실행하여 통합에 필요한 SecureString 파라미터를 생성합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다. /d9d01087-4a3f-49e0-b0b4-d568d7826553/ssm/integrations/webhook/ 네임스페이스는 통합을 위해 Parameter Store에 예약되어 있습니다. 파라미터 이름은 이 네임스페이스 다음에 통합 이름을 사용해야 합니다. 예: /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"}'
```

통합을 위한 웹후크 생성

공급자와 웹후크를 생성할 때는 다음 사항에 유의하세요.

- 프로토콜은 HTTPS여야 합니다.
- 사용자 정의 요청 헤더가 지원됩니다.
- 기본 요청 본문을 지정할 수 있습니다.
- aws:invokeWebhook 작업을 사용하여 통합을 호출할 때 기본 요청 본문을 재정의할 수 있습니다.

실행서에서 시간 제한 처리

timeoutSeconds 속성은 모든 자동화 작업에서 공유됩니다. 이 속성을 사용하여 작업에 대한 실행 제한 시간 값을 지정할 수 있습니다. 또한 작업 시간 제한이 자동화 및 전체 실행 상태에 미치는 영향을 변경할 수 있습니다. 또한 작업에 onFailure 및 isCritical 공유 속성을 정의하여 수행할 수도 있습니다.

예를 들어 작업 시간이 초과된 경우 사용 사례에 따라 자동화가 다른 작업을 계속 진행하고 자동화의 전체 상태에 영향을 주지 않도록 해야 할 경우가 있습니다. 이 예에서는 timeoutSeconds 속성을 사용하여 작업 시간이 초과되기 전에 대기할 시간을 지정합니다. 그런 다음 시간 제한이 있는 경우 자동화가 이동해야 하는 작업 또는 단계를 지정합니다. 기본값 Abort 대신 onFailure 속성에 대한 step:*step name* 형식을 사용하여 값을 지정합니다. 기본적으로 작업 시간이 초과되면 자동화 실행 상태는 Timed Out이 됩니다. 시간 제한이 자동화 실행 상태에 영향을 미치지 않도록 하려면 false 속성에 isCritical을 지정합니다.

다음 예에서는 이 시나리오에서 설명하는 작업의 공유 속성을 정의하는 방법을 보여줍니다.

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"
}
```

모든 자동화 작업에서 공유하는 속성에 대한 자세한 내용은 모든 작업에서 공유하는 속성 섹션을 참조하세요.

Systems Manager Automation 실행서 참조

AWS Systems Manager는 빠르게 시작할 수 있도록 미리 정의된 실행서를 제공합니다. 이러한 실행서는 Amazon Web Services, AWS Support 및 AWS Config에 의해 유지 관리됩니다. 실행서 참조에서는 Systems Manager, AWS Support 및 AWS Config에서 제공하는 미리 정의된 각 실행서를 설명합니다. 자세한 내용은 Systems Manager Automation 실행서 참조를 참조하세요.

튜토리얼

다음 자습서는 AWS Systems Manager Automation을 사용하여 일반적인 사용 사례를 해결하는 데 도움이 됩니다. 이 자습서에서는 자체 런북, Automation에서 제공하는 사전 정의된 런북 및 기타 Systems Manager 기능을 다른 AWS 서비스에 사용하는 방법을 보여줍니다.

목차

        - AWS CloudFormation 템플릿을 사용하여 권한 부여
    - 자동화 실행
- 입력 변환기를 사용하여 Automation에 데이터 전달

AMIs 업데이트

다음 자습서에서는 최신 패치를 포함하도록 Amazon Machine Image(AMIs)를 업데이트하는 방법을 설명합니다.

주제

- Linux AMI 업데이트
- Linux AMI(AWS CLI) 업데이트
- Windows Server AMI 업데이트
- Automation, AWS Lambda 및 Parameter Store를 사용하여 골든 AMI 업데이트
- Automation 및 Jenkins를 사용하여 AMIs 업데이트
- 오토 스케일링을 위해 AMIs 업데이트

Linux AMI 업데이트

이 Systems Manager Automation 시연에서는 AWS CLI와 `AWS-UpdateLinuxAmi` 런북을 사용하여, 지정한 최신 패키지의 패치로 Linux AMI를 자동으로 업데이트하는 방법을 보여줍니다. Automation은 AWS Systems Manager의 기능입니다. `AWS-UpdateLinuxAmi` 실행서는 사이트에 특정한 추가 패키지와 구성의 설치도 자동화합니다. 이 시연을 통해 Ubuntu Server, CentOS, RHEL, SLES 또는 Amazon Linux AMIs를 포함하여 다양한 Linux 배포를 업데이트할 수 있습니다. 지원되는 Linux 버전의 전체 목록은 Patch Manager 필수 조건 섹션을 참조하세요.

`AWS-UpdateLinuxAmi` 런북을 사용하면 JSON 또는 YAML로 런북을 만들지 않고도 이미지 유지 관리 태스크를 자동화할 수 있습니다. 다음과 같은 유형의 태스크를 수행하는 데 `AWS-UpdateLinuxAmi` 실행서를 사용할 수 있습니다.

- Amazon Linux, Red Hat Enterprise Linux, Ubuntu Server, SUSE Linux Enterprise Server 또는 CentOS Amazon Machine Image(AMI)에서 모든 배포 패키지와 Amazon 소프트웨어를 업그레이드합니다. 이는 기본 실행서 동작입니다.
- 기존 이미지에 AWS Systems Manager SSM Agent를 설치하여 Systems Manager 기능을 사용합니다. 예를 들면 AWS Systems Manager Run Command를 사용하여 원격 명령을 실행하거나 Inventory를 사용하여 소프트웨어 인벤토리를 수집할 수 있습니다.

- 추가 소프트웨어 패키지를 설치합니다.

시작하기 전 준비 사항

실행서로 작업을 시작하기 전에 역할을 구성하고, 선택적으로 Automation을 위한 EventBridge를 구성합니다. 자세한 내용은 Automation 설정 단원을 참조하십시오. 또한 이 시연에서는 AWS Identity and Access Management(IAM) 인스턴스 프로파일의 이름을 지정해야 합니다. IAM 인스턴스 프로파일 생성에 대한 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성을 참조하세요.

`AWS-UpdateLinuxAmi` 실행서는 다음 입력 파라미터를 수락합니다.

파라미터	유형	설명
SourceAmiId	String	(필수) 소스 AMI ID입니다.
IamInstanceProfileName	String	(필수) Systems Manager에 필요한 인스턴스 권한 구성에서 생성한 IAM 인스턴스 프로파일 역할의 이름입니다. 인스턴스 프로파일 역할은 인스턴스에 대해 명령 실행 또는 서비스 시작 및 중지와 같은 작업을 수행할 권한을 자동화 서비스에 부여합니다. 실행서에서는 이 인스턴스 프로파일 역할의 이름만 사용합니다. Amazon 리소스 이름(ARN)을 지정하면 자동화가 실패합니다.
AutomationAssumeRole	String	(필수) Automation 설정에서 생성한 IAM 서비스 역할의 이름입니다. 서비스 역할(수임 역할)은 IAM 역할을 수임하고 사용자를 대신하여 작업을 수행할 수 있는 Automation 권한을 부여합니다. 예를 들어 서비스 역할은 실행서에서 `aws:createImage` 작업을

파라미터	유형	설명
		실행할 때 Automation이 새로운 AMI를 생성할 수 있도록 합니다. 이 파라미터의 경우, 전체 ARN을 지정해야 합니다.
TargetAmiName	String	(옵션) 새로 생성된 AMI의 이름입니다. 기본 이름은 원본 AMI ID, 생성 시간 및 날짜가 포함된 시스템 생성 문자열입니다.
InstanceType	String	(선택 사항) 작업 영역의 호스트로 시작할 인스턴스의 유형입니다. 인스턴스 유형은 리전마다 다릅니다. 기본 형식은 t2.micro입니다.
PreUpdateScript	String	(선택 사항) 업데이트가 적용되기 전에 실행할 스크립트의 URL. 기본값(\"none\")은 스크립트를 실행하지 않는 것입니다.
PostUpdateScript	String	(선택 사항) 패키지 업데이트가 적용된 후에 실행할 스크립트의 URL. 기본값(\"none\")은 스크립트를 실행하지 않는 것입니다.
IncludePackages	String	(선택 사항) 이름 지정된 패키지만 업데이트합니다. 기본적으로(\"all\"), 사용 가능한 업데이트는 모두 적용합니다.

파라미터	유형	설명
ExcludePackages	String	(선택 사항) 어떤 조건에서나 업데이트를 보류할 패키지의 이름입니다. 기본적으로(\"none\"), 어떤 패키지도 제외되지 않습니다.

자동화 단계

AWS-UpdateLinuxAmi 실행서에는 기본적으로 다음과 같은 자동화 작업이 포함됩니다.

1단계: launchInstance(**aws:runInstances** 작업)

이 단계에서는 Amazon Elastic Compute Cloud(Amazon EC2) 사용자 데이터와 IAM 인스턴스 프로파일 역할을 사용하여 인스턴스를 시작합니다. 이때 사용자 데이터는 운영 체제에 따라 적절한 SSM Agent를 설치합니다. SSM Agent를 설치하면 Run Command, State Manager 및 Inventory와 같은 Systems Manager 기능을 사용할 수 있습니다.

2단계: updateOSSoftware(**aws:runCommand** 작업)

이 단계에서는 시작된 인스턴스에서 다음 명령을 실행합니다.

- Amazon S3에서 업데이트 스크립트를 다운로드합니다.
- 선택 사항인 업데이트 전 스크립트를 실행합니다.
- 배포 패키지와 Amazon 소프트웨어를 업데이트합니다.
- 선택 사항인 업데이트 후 스크립트를 실행합니다.

사용자가 나중에 볼 수 있도록 /tmp 폴더에 실행 로그가 저장됩니다.

특정한 패키지 세트를 업그레이드하려면 IncludePackages 파라미터를 사용하여 목록을 입력하면 됩니다. 목록이 입력되면 시스템에서 해당하는 패키지와 그 종속 항목만 업데이트하려고 시도합니다. 다른 업데이트는 수행하지 않습니다. 기본적으로, 포함 패키지를 지정하지 않으면 이 프로그램은 사용 가능한 패키지를 모두 업데이트합니다.

특정한 패키지 세트를 업그레이드에서 제외하려면 ExcludePackages 파라미터에 목록을 입력하면 됩니다. 목록이 입력되면 다른 어떤 옵션을 지정했든 간에 이러한 패키지는 현재 버전으로 유지됩니다. 기본적으로, 제외 패키지를 지정하지 않으면 어떤 패키지도 제외되지 않습니다.

3단계: stopInstance(**aws:changeInstanceState** 작업)

이 단계에서는 업데이트된 인스턴스를 중지합니다.

4단계: createImage(**aws:createImage** 작업)

이 단계에서는 원본 ID 및 생성 시간과 링크할 수 있는 설명 이름으로 새 AMI를 생성합니다. 예를 들어 "EC2 Automation이 {{global:DATE_TIME}}에 {{SourceAmiId}}에서 생성한 AMI"와 같습니다. 여기서 DATE_TIME 및 SourceID는 Automation의 변수를 나타냅니다.

5단계: terminateInstance(**aws:changeInstanceState** 작업)

이 단계에서는 실행 인스턴스를 종료하여 자동화를 정리합니다.

출력

자동화는 새로운 AMI ID를 출력으로 반환합니다.

> **Note**
>
> 기본적으로 Automation에서 `AWS-UpdateLinuxAmi` 실행서를 실행하면 시스템이 기본 VPC에 임시 인스턴스를 생성합니다(172.30.0.0/16). 기본 VPC를 삭제했다면 다음 오류 메시지를 받게 됩니다.
>
> `VPC not defined 400`
>
> 이 문제를 해결하려면 `AWS-UpdateLinuxAmi` 실행서의 사본을 만들고 서브넷 ID를 지정해야 합니다. 자세한 내용은 VPC not defined 400 단원을 참조하십시오.

Automation을 이용해 패치된 AMI를 생성하려면(AWS Systems Manager)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 왼쪽 탐색 창에서 Automation을 선택합니다.
3. 자동화 실행(Execute automation)을 선택합니다.
4. [Automation 문서(Automation document)] 목록에서 [**AWS-UpdateLinuxAmi**]를 선택합니다.
5. 문서 세부 정보 섹션에서 문서 버전이 런타임 시 기본 버전으로 설정되어 있는지 확인합니다.
6. Next(다음)를 선택합니다.
7. Execution mode(실행 모드) 섹션에서 Simple Execution(단순 실행)을 선택합니다.
8. 시작하기 전에 섹션에서 수집한 정보를 [입력 파라미터(Input parameters)] 섹션에 입력합니다.
9. 실행을 선택합니다. 콘솔에 자동화 실행의 상태가 표시됩니다.

워크플로가 종료된 후 업데이트된 AMI에서 테스트 인스턴스를 시작하여 변경을 확인합니다.

> **Note**
>
> 자동화 중 실패한 단계가 있으면 [Automation 실행(Automation Executions)] 페이지에 실패에 대한 정보가 나열됩니다. 자동화는 모든 태스크를 성공적으로 완료한 후 임시 인스턴스를 종료하도록 디자인되었습니다. 단계가 실패하면 시스템이 인스턴스를 종료하지 않을 수 있습니다. 따라서 단계가 실패하면 임시 인스턴스를 수동으로 종료합니다.

Linux AMI(AWS CLI) 업데이트

이 AWS Systems Manager Automation 시연은 AWS Command Line Interface(AWS CLI)와 Systems Manager `AWS-UpdateLinuxAmi` 실행서를 사용하여 지정한 최신 패키지 버전으로 Linux Amazon Machine Image(AMI)를 자동으로 패치하는 방법을 보여줍니다. Automation은 AWS Systems Manager의 기능입니다. `AWS-UpdateLinuxAmi` 실행서는 사이트에 특정한 추가 패키지와 구성의 설치도 자동화합니다. 이 시연을 통해 Ubuntu Server, CentOS, RHEL, SLES 또는 Amazon Linux AMIs를 포함하여 다양한 Linux 배포를 업데이트할 수 있습니다. 지원되는 Linux 버전의 전체 목록은 Patch Manager 필수 조건 섹션을 참조하세요.

`AWS-UpdateLinuxAmi` 실행서를 사용하면 JSON 또는 YAML로 실행서를 만들지 않고도 이미지 유지 관리 태스크를 자동화할 수 있습니다. 다음과 같은 유형의 태스크를 수행하는 데 `AWS-UpdateLinuxAmi` 실행서를 사용할 수 있습니다.

- Amazon Linux, Red Hat Enterprise Linux, Ubuntu Server, SLES 또는 Cent OS Amazon Machine Image(AMI)에서 모든 배포 패키지와 Amazon 소프트웨어를 업그레이드합니다. 이는 기본 실행서 동작입니다.
- 기존 이미지에 AWS Systems Manager SSM Agent를 설치하여 Systems Manager 기능을 사용합니다. 예를 들면 AWS Systems Manager Run Command를 사용하여 원격 명령을 실행하거나 Inventory를 사용하여 소프트웨어 인벤토리를 수집할 수 있습니다.
- 추가 소프트웨어 패키지를 설치합니다.

시작하기 전 준비 사항

실행서로 작업을 시작하기 전에 역할을 구성하고, 선택적으로 Automation을 위한 EventBridge를 구성합니다. 자세한 내용은 Automation 설정 단원을 참조하십시오. 또한 이 시연에서는 AWS Identity and Access Management(IAM) 인스턴스 프로파일의 이름을 지정해야 합니다. IAM 인스턴스 프로파일 생성에 대한 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성을 참조하세요.

AWS-UpdateLinuxAmi 실행서는 다음 입력 파라미터를 수락합니다.

파라미터	유형	설명
SourceAmiId	String	(필수) 소스 AMI ID입니다. AWS Systems Manager Parameter Store public 파라미터를 사용하여 Linux용 Amazon EC2 AMI의 최신 ID를 자동으로 참조할 수 있습니다. 자세한 내용은 Query for the latest Amazon Linux AMI IDs using AWS Systems Manager Parameter Store를 참조하세요.
IamInstanceProfileName	String	(필수) Systems Manager에 필요한 인스턴스 권한 구성에서 생성한 IAM 인스턴스 프로파일 역할의 이름입니다. 인스턴스 프로파일 역할은 인스턴스에 대해 명령 실행 또는 서비스 시작 및 중지와 같은 작업을 수행할 권한을 자동화 서비스에 부여합니다. 실행서에서는 이 인스턴스 프로파일 역할의 이름만 사용합니다.
AutomationAssumeRole	String	(필수) Automation 설정에서 생성한 IAM 서비스 역할의 이름입니다. 서비스 역할(수임 역할)은 IAM 역할을 수임하고 사용자를 대신하여 작업을 수행할 수 있는 Automation 권한을 부여합니다. 예를 들어 서비스 역할은 실행서에서 aws:createImage 작업을

파라미터	유형	설명
		실행할 때 Automation이 새로운 AMI를 생성할 수 있도록 합니다. 이 파라미터의 경우, 전체 ARN을 지정해야 합니다.
TargetAmiName	String	(옵션) 새로 생성된 AMI의 이름입니다. 기본 이름은 원본 AMI ID, 생성 시간 및 날짜가 포함된 시스템 생성 문자열입니다.
InstanceType	String	(선택 사항) 작업 영역의 호스트로 시작할 인스턴스의 유형입니다. 인스턴스 유형은 리전마다 다릅니다. 기본 형식은 t2.micro입니다.
PreUpdateScript	String	(선택 사항) 업데이트가 적용되기 전에 실행할 스크립트의 URL. 기본값(\"none\")은 스크립트를 실행하지 않는 것입니다.
PostUpdateScript	String	(선택 사항) 패키지 업데이트가 적용된 후에 실행할 스크립트의 URL. 기본값(\"none\")은 스크립트를 실행하지 않는 것입니다.
IncludePackages	String	(선택 사항) 이름 지정된 패키지만 업데이트합니다. 기본적으로(\"all\"), 사용 가능한 업데이트는 모두 적용합니다.

파라미터	유형	설명
ExcludePackages	String	(선택 사항) 어떤 조건에서나 업데이트를 보류할 패키지의 이름입니다. 기본적으로(\"none\"), 어떤 패키지도 제외되지 않습니다.

자동화 단계

AWS-UpdateLinuxAmi 실행서에는 기본적으로 다음과 같은 단계가 포함됩니다.

1단계: launchInstance(**aws:runInstances** 작업)

이 단계에서는 Amazon Elastic Compute Cloud(Amazon EC2) 사용자 데이터와 IAM 인스턴스 프로파일 역할을 사용하여 인스턴스를 시작합니다. 이때 사용자 데이터는 운영 체제에 따라 적절한 SSM Agent를 설치합니다. SSM Agent를 설치하면 Run Command, State Manager 및 Inventory와 같은 Systems Manager 기능을 사용할 수 있습니다.

2단계: updateOSSoftware(**aws:runCommand** 작업)

이 단계에서는 시작된 인스턴스에서 다음 명령을 실행합니다.

- Amazon Simple Storage Service(Amazon S3)에서 업데이트 스크립트를 다운로드합니다.
- 선택 사항인 업데이트 전 스크립트를 실행합니다.
- 배포 패키지와 Amazon 소프트웨어를 업데이트합니다.
- 선택 사항인 업데이트 후 스크립트를 실행합니다.

사용자가 나중에 볼 수 있도록 /tmp 폴더에 실행 로그가 저장됩니다.

특정한 패키지 세트를 업그레이드하려면 IncludePackages 파라미터를 사용하여 목록을 입력하면 됩니다. 목록이 입력되면 시스템에서 해당하는 패키지와 그 종속 항목만 업데이트하려고 시도합니다. 다른 업데이트는 수행하지 않습니다. 기본적으로, 포함 패키지를 지정하지 않으면 이 프로그램은 사용 가능한 패키지를 모두 업데이트합니다.

특정한 패키지 세트를 업그레이드에서 제외하려면 ExcludePackages 파라미터에 목록을 입력하면 됩니다. 목록이 입력되면 다른 어떤 옵션을 지정했든 간에 이러한 패키지는 현재 버전으로 유지됩니다. 기본적으로, 제외 패키지를 지정하지 않으면 어떤 패키지도 제외되지 않습니다.

3단계: stopInstance(**aws:changeInstanceState** 작업)

이 단계에서는 업데이트된 인스턴스를 중지합니다.

4단계: createImage(**aws:createImage** 작업)

이 단계에서는 원본 ID 및 생성 시간과 링크할 수 있는 설명 이름으로 새 AMI를 생성합니다. 예를 들면 "EC2 자동화가 {{global:DATE_TIME}}에 {{SourceAmiId}}에서 생성한 AMI"와 같습니다. 여기서 DATE_TIME 및 SourceID는 자동화의 변수를 나타냅니다.

5단계: terminateInstance(**aws:changeInstanceState** 작업)

이 단계에서는 실행 인스턴스를 종료하여 자동화를 정리합니다.

출력

자동화는 새로운 AMI ID를 출력으로 반환합니다.

> **Note**
>
> 기본적으로 Automation에서 `AWS-UpdateLinuxAmi` 실행서를 실행하면 시스템이 기본 VPC에 임시 인스턴스를 생성합니다(172.30.0.0/16). 기본 VPC를 삭제했다면 다음 오류 메시지를 받게 됩니다.
>
> `VPC not defined 400`
>
> 이 문제를 해결하려면 `AWS-UpdateLinuxAmi` 실행서의 사본을 만들고 서브넷 ID를 지정해야 합니다. 자세한 내용은 VPC not defined 400 단원을 참조하십시오.

Automation을 이용해 패치된 AMI를 생성하려면

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.

2. 다음 명령을 실행하여 `AWS-UpdateLinuxAmi` 실행서를 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
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'
```

이 명령으로 실행 ID가 반환됩니다. 클립보드에 이 ID를 복사합니다. 이 ID를 사용하여 자동화 상태를 확인합니다.

```
{
    "AutomationExecutionId": "automation execution ID"
}
```

3. AWS CLI를 사용하여 자동화를 보려면 다음 명령을 실행합니다.

```
aws ssm describe-automation-executions
```

4. 자동화 진행 상황에 대한 세부 정보를 보려면 다음 명령을 실행합니다. *### ## ID*를 자신의 정보를 바꿉니다.

```
aws ssm get-automation-execution --automation-execution-id automation execution ID
```

업데이트 프로세스가 완료될 때까지 30분 이상 걸릴 수 있습니다.

> **Note**
>
> 콘솔에서 자동화 상태를 모니터링할 수도 있습니다. 목록에서 방금 실행한 자동화를 선택한 후 [단계(Steps)] 탭을 선택합니다. 이 탭은 자동화 작업의 상태를 보여줍니다.

워크플로가 종료된 후 업데이트된 AMI에서 테스트 인스턴스를 시작하여 변경을 확인합니다.

> **Note**
>
> 자동화 중 실패한 단계가 있으면 [Automation 실행(Automation Executions)] 페이지에 실패에 대한 정보가 나열됩니다. 자동화는 모든 태스크를 성공적으로 완료한 후 임시 인스턴스를 종료하도록 디자인되었습니다. 단계가 실패하면 시스템이 인스턴스를 종료하지 않을 수 있습니다. 따라서 단계가 실패하면 임시 인스턴스를 수동으로 종료합니다.

Windows Server AMI 업데이트

`AWS-UpdateWindowsAmi` 실행서를 사용하면 JSON 또는 YAML로 실행서를 만들지 않고도 Amazon Windows Amazon Machine Image(AMI)를 기반으로 이미지 유지 관리 태스크를 자동화할 수 있습니다. 이 실행서는 Windows Server 2008 R2 이상에 지원됩니다. 다음과 같은 유형의 태스크를 수행하는 데 `AWS-UpdateWindowsAmi` 실행서를 사용할 수 있습니다.

- 모든 Windows 업데이트를 설치하고 Amazon 소프트웨어를 업그레이드합니다(기본 동작).
- 특정 Windows 업데이트를 설치하고 Amazon 소프트웨어를 업그레이드합니다.
- 스크립트를 이용해 AMI를 사용자 지정합니다.

시작하기 전 준비 사항

실행서 작업을 시작하기 전에 액세스 권한을 부여하려는 인스턴스 프로파일의 ARN을 참조하는 `iam:PassRole` 정책을 추가하도록 Automation에 대한 역할을 구성합니다. 필요에 따라 AWS Systems Manager의 기능인 Automation에 대해 Amazon EventBridge를 구성합니다. 자세한 내용은 Automation 설정 단원을 참조하십시오. 또한 이 시연에서는 AWS Identity and Access Management(IAM) 인스턴스 프로파일의 이름을 지정해야 합니다. IAM 인스턴스 프로파일 생성에 대한 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성을 참조하세요.

> **Note**
>
> AWS Systems Manager SSM Agent 업데이트는 일반적으로 서로 다른 시점에 여러 리전에 배포됩니다. AMI를 사용자 지정하거나 업데이트하려면 작업 중인 리전에 게시된 소스 AMI만 사용합니다. 그러면 해당 리전에 릴리스된 최신 SSM Agent로 작업하고 호환성 문제를 피할 수 있습니다.

`AWS-UpdateWindowsAmi` 실행서는 다음 입력 파라미터를 수락합니다.

파라미터	유형	설명
SourceAmiId	String	(필수) 소스 AMI ID입니다. Systems Manager Parameter Store public 파라미터를 사용하여 자동으로 최신 Windows Server AMI ID를 참조할 수

파라미터	유형	설명
		있습니다. 자세한 내용은 AWS Systems ManagerParameter Store를 사용하여 최신 Windows AMI ID 쿼리를 참조하세요.
SubnetId	String	(선택 사항) 임시 인스턴스를 시작할 서브넷입니다. 기본 VPC를 삭제한 경우 이 파라미터의 값을 지정해야 합니다.
IamInstanceProfileName	String	(필수) Systems Manager에 필요한 인스턴스 권한 구성에서 생성한 IAM 인스턴스 프로파일 역할의 이름입니다. 인스턴스 프로파일 역할은 인스턴스에 대해 명령 실행 또는 서비스 시작 및 중지와 같은 작업을 수행할 권한을 자동화 서비스에 부여합니다. 실행서에서는 이 인스턴스 프로파일 역할의 이름만 사용합니다.

파라미터	유형	설명
AutomationAssumeRole	String	(필수) Automation 설정에서 생성한 IAM 서비스 역할의 이름입니다. 서비스 역할(수임 역할)은 IAM 역할을 수임하고 사용자를 대신하여 작업을 수행할 수 있는 Automation 권한을 부여합니다. 예를 들어 서비스 역할은 실행서에서 `aws:createImage` 작업을 실행할 때 Automation이 새로운 AMI를 생성할 수 있도록 합니다. 이 파라미터의 경우, 전체 ARN을 지정해야 합니다.
TargetAmiName	String	(옵션) 새로 생성된 AMI의 이름입니다. 기본 이름은 원본 AMI ID, 생성 시간 및 날짜가 포함된 시스템 생성 문자열입니다.
InstanceType	String	(선택 사항) 작업 영역의 호스트로 시작할 인스턴스의 유형입니다. 인스턴스 유형은 리전마다 다릅니다. 기본 형식은 t2.medium입니다.
PreUpdateScript	String	(옵션) AMI를 업데이트하기 전에 실행할 스크립트. 실행서 또는 런타임에 파라미터로 스크립트를 입력합니다.
PostUpdateScript	String	(옵션) AMI를 업데이트한 후에 실행할 스크립트. 실행서 또는 런타임에 파라미터로 스크립트를 입력합니다.

파라미터	유형	설명
IncludeKbs	String	(선택 사항) 포함시킬 하나 이상의 Microsoft Knowledge Base(KB) 문서 ID를 지정합니다. 쉼표로 분리된 값을 사용하여 여러 ID를 설치할 수 있습니다. 유효한 형식: KB9876543 또는 9876543.
ExcludeKbs	String	(선택 사항) 제외할 하나 이상의 Microsoft Knowledge Base(KB) 문서 ID를 지정합니다. 쉼표로 분리된 값을 사용하여 여러 ID를 제외할 수 있습니다. 유효한 형식: KB9876543 또는 9876543.
카테고리	String	(선택 사항)하나 이상의 업데이트 범주를 지정합니다. 쉼표로 분리된 값을 사용하여 범주를 필터링할 수 있습니다. 옵션: 중요 업데이트, 보안 업데이트, 정의 업데이트, 업데이트 롤업, 서비스 팩, 도구, 업데이트 또는 드라이버. 유효한 형식에는 중요 업데이트 등 단일 입력이 포함됩니다. 또는 중요 업데이트, 보안 업데이트,정의 업데이트처럼 쉼표로 구분된 목록을 지정할 수 있습니다.

파라미터	유형	설명
SeverityLevels	String	(선택 사항) 업데이트와 연결되는 하나 이상의 MSRC 심각도를 지정합니다. 쉼표로 분리된 값을 사용하여 심각도를 필터링할 수 있습니다. 옵션: 심각, 중요, 낮음, 보통 또는 비지정. 유효한 형식에는 심각 등 단일 입력이 포함됩니다. 또는 심각,중요,낮음처럼 쉼표로 구분된 목록을 지정할 수 있습니다.

자동화 단계

`AWS-UpdateWindowsAmi` 실행서에는 기본적으로 다음과 같은 단계가 포함됩니다.

1단계: launchInstance(**`aws:runInstances`** 작업)

이 단계는 지정된 `SourceAmiID`의 IAM 인스턴스 프로파일 역할을 사용하여 인스턴스를 시작합니다.

2단계: runPreUpdateScript(**`aws:runCommand`** 작업)

이 단계에서는 업데이트가 설치되기 전에 실행되는 문자열로 스크립트를 지정할 수 있습니다.

3단계: updateEC2Config(**`aws:runCommand`** 작업)

이 단계는 `AWS-InstallPowerShellModule` 실행서를 사용하여 AWS 퍼블릭 PowerShell 모듈을 다운로드합니다. Systems Manager는 SHA-256 해시를 사용하여 모듈의 무결성을 확인합니다. 그런 다음 Systems Manager는 운영 체제를 확인하여 EC2Config 또는 EC2Launch를 업데이트할지 결정합니다. EC2Config는 Windows Server 2008 R2에서 Windows Server 2012 R2까지 실행됩니다. EC2Launch는 Windows Server 2016에서 실행됩니다.

4단계: updateSSMAgent(**`aws:runCommand`** 작업)

이 단계는 `AWS-UpdateSSMAgent` 실행서를 사용하여 SSM Agent를 업데이트합니다.

5단계: updateAWSPVDriver(**`aws:runCommand`** 작업)

이 단계는 `AWS-ConfigureAWSPackage` 실행서를 사용하여 AWS PV 드라이버를 업데이트합니다.

6단계: updateAwsEnaNetworkDriver **aws:runCommand** 작업)

이 단계는 AWS-ConfigureAWSPackage 실행서를 사용하여 AWS ENA 네트워크 드라이버를 업데이트합니다.

7단계: installWindowsUpdates(**aws:runCommand** 작업)

이 단계는 AWS-InstallWindowsUpdates 실행서를 사용하여 Windows 업데이트를 설치합니다. 기본적으로 Systems Manager는 누락된 모든 업데이트를 검색하고 설치합니다. 파라미터 IncludeKbs, ExcludeKbs, Categories 또는 SeverityLevels 중 하나를 지정하여 기본 동작을 변경할 수 있습니다.

8단계: runPostUpdateScript(**aws:runCommand** 작업)

이 단계에서는 업데이트가 설치된 후에 실행되는 문자열로 스크립트를 지정할 수 있습니다.

9단계: runSysprepGeneralize(**aws:runCommand** 작업)

이 단계는 AWS-InstallPowerShellModule 실행서를 사용하여 AWS 퍼블릭 PowerShell 모듈을 다운로드합니다. Systems Manager는 SHA-256 해시를 사용하여 모듈의 무결성을 확인합니다. 그런 다음 Systems Manager는 EC2Launch(Windows Server 2016) 또는 EC2Config(Windows Server 2008 R2~2012 R2)에 대해 AWS 지원 방법을 사용하여 sysprep을 실행합니다.

10단계: stopInstance(**aws:changeInstanceState** 작업)

이 단계에서는 업데이트된 인스턴스를 중지합니다.

11단계: createImage(**aws:createImage** 작업)

이 단계에서는 원본 ID 및 생성 시간과 링크할 수 있는 설명 이름으로 새 AMI를 생성합니다. 예를 들면 "EC2 자동화가 {{global:DATE_TIME}}에 {{SourceAmiId}}에서 생성한 AMI"와 같습니다. 여기서 DATE_TIME 및 SourceID는 자동화의 변수를 나타냅니다.

12단계: TerminateInstance(**aws:changeInstanceState** 작업)

이 단계에서는 실행 인스턴스를 종료하여 자동화를 정리합니다.

출력

이 섹션에서는 다양한 단계의 출력 또는 원하는 파라미터의 값을 자동화 출력으로 지정할 수 있습니다. 기본적으로 출력은 실행에 의해 생성되어 업데이트된 Windows AMI의 ID입니다.

> **Note**
>
> 기본적으로 Automation이 `AWS-UpdateWindowsAmi` 실행서를 실행하고 임시 인스턴스를 생성하면 시스템이 기본 VPC를 사용합니다(172.30.0.0/16). 기본 VPC를 삭제했다면 다음 오류 메시지를 받게 됩니다.
>
> VPC not defined 400
>
> 이 문제를 해결하려면 `AWS-UpdateWindowsAmi` 실행서의 사본을 만들고 서브넷 ID를 지정해야 합니다. 자세한 내용은 VPC not defined 400 단원을 참조하십시오.

Automation을 이용해 패치된 Windows AMI를 생성하려면

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.

2. 다음 명령을 실행하여 `AWS-UpdateWindowsAmi` 실행서를 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다. 아래 예제 명령은 최신 Amazon EC2 AMI를 사용하여 적용해야 하는 패치의 개수를 최소화합니다. 이 명령을 두 번 이상 실행하면 `targetAMIname`에 고유한 값을 지정해야 합니다. AMI 이름은 고유해야 합니다.

   ```
   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'
   ```

   이 명령으로 실행 ID가 반환됩니다. 클립보드에 이 ID를 복사합니다. 이 ID를 사용하여 자동화 상태를 확인합니다.

   ```
   {
       "AutomationExecutionId": "automation execution ID"
   }
   ```

3. AWS CLI를 사용하여 자동화를 보려면 다음 명령을 실행합니다.

   ```
   aws ssm describe-automation-executions
   ```

4. 자동화 진행 상황에 대한 세부 정보를 보려면 다음 명령을 실행합니다.

```
aws ssm get-automation-execution
    --automation-execution-id automation execution ID
```

> **Note**
>
> 적용된 패치 수에 따라 이 샘플 자동화에서 실행된 Windows 패치 프로세스가 완료되는 데 30분 이상의 시간이 소요될 수 있습니다.

Automation, AWS Lambda 및 Parameter Store를 사용하여 골든 AMI 업데이트

다음 예시에는 조직이 Amazon Elastic Compute Cloud(Amazon EC2) AMIs에서 구축하는 것이 아니라 자체적인 고유의 AMIs를 유지 관리하고 정기적으로 패치하는 모델을 사용합니다.

다음 절차에서는 이미 최신 AMI로 간주되는 AMI에 운영 체제(OS) 패치를 자동으로 적용하는 방법을 보여줍니다. 이 예제에서 `SourceAmiId` 파라미터의 기본값은 `latestAmi`라는 AWS Systems Manager Parameter Store 파라미터로 정의됩니다. `latestAmi` 값은 자동화가 끝날 때 호출되는 AWS Lambda 함수를 통해 업데이트됩니다. 이 Automation 프로세스의 결과로 최신 AMI에 패치가 항상 적용되기 때문에 AMIs를 패치하는 데 소비되는 시간과 노력이 최소화됩니다. Parameter Store와 Automation은 AWS Systems Manager의 기능입니다.

시작하기 전 준비 사항

Automation 역할 및 Automation을 위한 Amazon EventBridge(옵션)를 구성합니다. 자세한 내용은 Automation 설정 단원을 참조하십시오.

목차

태스크 1: Systems Manager Parameter Store에서 파라미터 생성

다음 정보를 사용하는 문자열 파라미터를 Parameter Store에 생성

- 이름: `latestAmi`.

- 값: AMI ID입니다. 예: `ami-188d6e0e`.

Parameter Store 문자열 파라미터를 생성하는 방법에 대한 자세한 내용은 Systems Manager 파라미터 생성 섹션을 참조하세요.

태스크 2: AWS Lambda용 IAM 역할 생성

다음 절차를 사용하여 AWS Lambda용 IAM 서비스 역할을 생성합니다. 이 정책은 Lambda 함수와 Systems Manager를 사용하여 `latestAmi` 파라미터의 값을 업데이트할 Lambda 권한을 부여합니다.

Lambda용 IAM 서비스 역할을 생성하려면

1. AWS Management Console에 로그인하여 https://console.aws.amazon.com/iam/ 에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 정책을 선택한 후 정책 생성을 선택합니다.
3. JSON 탭을 선택합니다.
4. 기본 콘텐츠를 다음과 같은 정책으로 바꿉니다. 각 *### ## ### #*를 자신의 정보로 바꿉니다.

```
{
    "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. 다음: 태그를 선택합니다.

6. (선택 사항) 이 정책에 대한 액세스를 구성, 추적 또는 제어할 태그-키 값 페어를 하나 이상 추가합니다.
7. 다음: 검토를 선택합니다.
8. 정책 검토(Review policy) 페이지에서 이름(Name)에 인라인 정책 이름을 입력합니다(예: **amiLambda**)
9. 정책 생성을 선택합니다.
10. 2단계와 3단계를 반복합니다.
11. 다음 정책을 붙여넣습니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
{
    "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. 다음: 태그를 선택합니다.
13. (선택 사항) 이 정책에 대한 액세스를 구성, 추적 또는 제어할 태그-키 값 페어를 하나 이상 추가합니다.
14. 다음: 검토를 선택합니다.
15. 정책 검토(Review policy) 페이지에서 이름(Name)에 인라인 정책 이름을 입력합니다(예: **amiParameter**)
16. 정책 생성을 선택합니다.
17. 탐색 창에서 역할을 선택한 후 역할 생성을 선택합니다.
18. 사용 사례에서 Lambda를 선택한 후 다음을 선택합니다.
19. 권한 추가 페이지의 검색 필드를 사용하여 앞서 생성한 두 정책을 찾습니다.
20. 정책 옆 확인란을 선택하고 다음을 선택합니다.

21. 역할 이름에 새 역할의 이름(예: **lambda-ssm-role** 또는 자신이 선호하는 다른 이름)을 입력합니다.

> **Note**
>
> 다양한 주체가 역할을 참조할 수 있기 때문에 역할이 생성된 후에는 역할 이름을 변경할 수 없습니다.

22. (선택 사항) 이 역할에 대한 액세스를 구성, 추적 또는 제어할 태그 키-값 페어를 하나 이상 추가한 후 역할 생성을 선택합니다.

작업 3: AWS Lambda 함수 생성

다음 절차를 사용하여 `latestAmi` 파라미터의 값을 자동으로 업데이트하는 Lambda 함수를 생성합니다.

Lambda 함수 생성하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/lambda/에서 AWS Lambda 콘솔을 엽니다.
2. 함수 생성을 선택합니다.
3. 함수 생성 페이지에서 처음부터 새로 작성을 선택합니다.
4. [함수 이름]에 **Automation-UpdateSsmParam**을 입력합니다.
5. 런타임에서 Python 3.8을 선택합니다.
6. 아키텍처에서 함수를 실행하는 데 사용할 Lambda용 컴퓨터 프로세서 유형(x86_64 또는 arm64)을 선택합니다.
7. 권한 섹션에서 기본 실행 역할 변경을 확장합니다.
8. [기존 역할 사용(Use an existing role)]을 선택하고 태스크 2에서 생성한 Lambda에 대한 서비스 역할을 선택합니다.
9. 함수 생성을 선택합니다.
10. 코드 소스 영역의 lambda_function 탭에서 필드에 미리 채워진 코드를 삭제하고 다음 코드 샘플을 붙여 넣습니다.

```
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. 파일, 저장을 선택합니다.
12. Lambda 함수를 테스트하려면 테스트 메뉴에서 테스트 이벤트 구성을 선택합니다.
13. 이벤트 이름에 테스트 이벤트의 이름을 입력합니다(예: **MyTestEvent**).
14. 기존 텍스트를 다음 JSON으로 바꿉니다. *AMI ID*를 자신의 정보로 바꾸어 latestAmi 파라미터 값을 설정합니다.

```
{
   "parameterName":"latestAmi",
   "parameterValue":"AMI ID"
}
```

15. Save(저장)를 선택합니다.
16. 함수를 테스트하려면 테스트를 선택합니다. 실행 결과 탭에서 업데이트에 대한 다른 세부 정보와 함께 상태가 성공으로 보고되어야 합니다.

태스크 4: 실행서 생성 및 AMI 패치

다음 절차를 사용하여 latestAmi 파라미터에 지정한 AMI를 패치하는 실행서를 생성하고 실행합니다. 자동화가 완료되면 latestAmi 값이 새로 패치된 AMI의 ID로 업데이트됩니다. 이후 자동화는 이전 실행에서 생성된 AMI를 사용합니다.

실행서를 생성하고 실행하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. 문서 생성에서 Automation을 선택합니다.
4. 이름에 **UpdateMyLatestWindowsAmi**를 입력합니다.
5. 편집기 탭을 선택하고 편집을 선택합니다.
6. 메시지가 나타나면 확인을 선택합니다.
7. 문서 편집기 필드에서 기본 콘텐츠를 다음 YAML 샘플 런북 콘텐츠로 바꿉니다.

```
---
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. Create automation(자동화 생성)을 선택합니다.
9. 탐색 창에서 Automation(자동화)을 선택한 후 Execute automation(자동화 실행)을 선택합니다.
10. 문서 선택(Choose document) 페이지에서 내 소유(Owned by me) 탭을 선택합니다.
11. UpdateMyLatestWindowsAmi 런북을 검색하고 UpdateMyLatestWindowsAmi 카드에서 버튼을 선택합니다.
12. 다음을 선택합니다.
13. Simple execution(단순 실행)을 선택합니다.
14. 입력 파라미터에 대한 값을 지정합니다.
15. 실행을 선택합니다.
16. 자동화가 완료된 후 탐색 창에서 Parameter Store를 선택하고 `latestAmi`의 새 값이 자동화에서 반환된 값과 일치하는지 확인합니다. 새 AMI ID가 Amazon EC2 콘솔의 [AMI(AMIs)] 섹션에 있는 Automation 출력과 일치하는지도 확인할 수 있습니다.

Automation 및 Jenkins를 사용하여 AMIs 업데이트

조직에서 CI/CD 파이프라인에 Jenkins 소프트웨어를 사용하는 경우 Automation을 사후 빌드 단계로 추가하여 Amazon Machine Images(AMIs)에 애플리케이션 릴리스를 미리 설치할 수 있습니다. Automation은 AWS Systems Manager의 기능입니다. 또한 Jenkins 예약 기능을 사용하여 Automation을 호출하고 고유의 OS(운영 체제) 패치 케이던스를 만들 수 있습니다.

아래 예제에서는 온프레미스 또는 Amazon Elastic Compute Cloud(Amazon EC2)에서 실행 중인 Jenkins 서버에서 Automation을 호출하는 방법을 보여줍니다. 인증을 위해 Jenkins 서버는 예제에서 생성하고 인스턴스 프로파일에 연결하는 IAM 정책을 기반으로 한 AWS 자격 증명을 사용합니다.

> **Note**
>
> 인스턴스를 구성할 때는 반드시 Jenkins 보안 모범 사례를 따르세요.

시작하기 전 준비 사항

Jenkins를 사용하여 Automation을 구성하기 전에 다음 작업을 완료합니다.

- Automation, AWS Lambda 및 Parameter Store를 사용하여 골든 AMI 업데이트 예제를 완료합니다. 다음 예제에서는 해당 예제에서 만든 UpdateMyLatestWindowsAmi 실행서를 사용합니다.

- Automation을 위한 IAM 역할을 구성합니다. 자동화 처리를 위해 Systems Manager에 인스턴스 프로파일 역할과 서비스 역할 ARN이 필요합니다. 자세한 내용은 Automation 설정 단원을 참조하십시오.

Jenkins 서버의 IAM 정책 생성

1. AWS Management Console에 로그인하여 https://console.aws.amazon.com/iam/ 에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 정책을 선택한 후 정책 생성을 선택합니다.
3. JSON 탭을 선택합니다.
4. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
{
        "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. Review policy(정책 검토)를 선택합니다.
6. 정책 검토(Review policy) 페이지에서 이름(Name)에 인라인 정책 이름을 입력합니다(예: **JenkinsPolicy**)
7. 정책 생성을 선택합니다.
8. 탐색 창에서 역할을 선택합니다.
9. Jenkins 서버에 연결된 인스턴스 프로파일을 선택합니다.
10. 권한 탭에서 권한 추가, 정책 연결을 선택합니다.
11. 기타 권한 정책 섹션에서 이전 단계에서 생성한 정책의 이름을 입력합니다. JenkinsPolicy를 예로 들 수 있습니다.
12. 정책 옆의 확인란을 선택한 다음 정책 연결을 선택합니다.

다음 절차를 사용하여 AWS CLI 서버에서 Jenkins를 구성합니다.

자동화를 위해 Jenkins 서버를 구성하려면

1. 기본 브라우저를 사용하여 포트 8080에서 Jenkins 서버에 연결하고 관리 인터페이스에 액세스합니다.
2. `/var/lib/jenkins/secrets/initialAdminPassword`에서 찾은 암호를 입력합니다. 암호를 표시하려면 다음 명령을 실행합니다.

   ```
   sudo cat /var/lib/jenkins/secrets/initialAdminPassword
   ```
3. Jenkins 설치 스크립트가 Jenkins 사용자 지정 페이지로 안내합니다. [제안된 플러그인 설치(Install suggested plugins)]를 선택합니다.
4. 설치가 완료되면 관리자 자격 증명, 자격 증명 저장, Jenkins를 사용하여 시작을 차례로 선택합니다.
5. 왼쪽 탐색 창에서 Jenkins 관리를 선택한 후 플러그인 관리를 선택합니다.
6. [사용 가능(Available)] 탭을 선택하고 **`Amazon EC2 plugin`**을 입력합니다.
7. **`Amazon EC2 plugin`** 확인란을 선택하고 [다시 시작하지 않고 설치(Install without restart)]를 선택합니다.
8. 설치가 완료되면 [맨 위 페이지로 돌아가기(Go back to the top page)]를 선택합니다.
9. Jenkins 관리를 선택한 다음 노드 및 클라우드 관리를 선택합니다.
10. 클라우드 구성 섹션에서 새 클라우드 추가를 선택한 다음 Amazon EC2를 선택합니다.
11. 나머지 필드에 정보를 입력합니다. EC2 인스턴스 프로파일을 사용하여 자격 증명 가져오기 옵션을 선택해야 합니다.

다음 절차를 사용하여 Automation을 호출하도록 Jenkins 프로젝트를 구성합니다.

Automation을 호출하도록 Jenkins 서버를 구성하려면

1. 웹 브라우저를 사용하여 Jenkins 콘솔을 엽니다.
2. 자동화로 구성할 프로젝트를 선택한 후 구성을 선택합니다.
3. 빌드 탭에서 빌드 단계 추가를 선택합니다.
4. Execute shell(셸 실행) 또는 Execute Windows batch command(Windows 배치 명령 실행)를 선택합니다(운영 체제에 따라 선택).

5. 명령(Command) 필드에서 다음과 같이 AWS CLI 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
aws ssm start-automation-execution \
        --document-name runbook name \
        --region AWS ## of your source AMI \
        --parameters runbook parameters
```

다음 예제 명령에서는 UpdateMyLatestWindowsAmi 실행서와 Automation, AWS Lambda 및 Parameter Store를 사용하여 골든 AMI 업데이트에서 생성한 Systems Manager 파라미터 `latestAmi`를 사용합니다.

```
aws ssm start-automation-execution \
        --document-name UpdateMyLatestWindowsAmi \
        --parameters \
            "sourceAMIid='{{ssm:latestAmi}}'"
        --region region
```

Jenkins에서 이 명령은 다음 스크린샷의 예제와 같이 표시됩니다.

6. Jenkins 프로젝트에서 지금 빌드를 선택합니다. Jenkins는 다음 예제와 비슷한 출력 결과를 반환합니다.

오토 스케일링을 위해 AMIs 업데이트

다음 예시는 Auto Scaling 그룹을 새로 패치된 AMI로 업데이트합니다. 이 접근 방법을 통해 Auto Scaling 그룹을 사용하는 다양한 컴퓨팅 환경에서 새로운 이미지를 자동으로 사용할 수 있습니다.

이 예시에서는 자동화의 최종 단계에서 새로 패치된 AMI를 사용하는 새로운 시작 템플릿을 Python 함수를 사용하여 생성합니다. 그런 다음 새 시작 템플릿을 사용하여 새 Auto Scaling 그룹을 생성합니다. 이 유형의 Auto Scaling 시나리오에서 사용자는 Auto Scaling 그룹의 기존 인스턴스를 종료하여 새로운 이미지를 사용하는 새로운 인스턴스를 강제로 시작할 수 있습니다. 또는 사용자는 기다렸다가 축소 또는 확장 이벤트를 통해 더 새로운 인스턴스를 자연스럽게 시작할 수 있습니다.

시작하기 전 준비 사항

이 예제를 시작하기 전에 다음 작업을 완료합니다.

- AWS Systems Manager의 기능인 Automation을 위한 IAM 역할을 구성합니다. 자동화 처리를 위해 Systems Manager에 인스턴스 프로파일 역할과 서비스 역할 ARN이 필요합니다. 자세한 내용은 Automation 설정 단원을 참조하십시오.

PatchAMIAndUpdateASG 런북 생성

다음 절차를 사용하여 SourceAMI 파라미터에 대해 지정한 AMI를 패치하는 PatchAMIAndUpdateASG 런북을 생성합니다. 런북은 또한 패치된 최신 AMI를 사용하도록 Auto Scaling 그룹을 업데이트합니다.

실행서를 생성하고 실행하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. [문서 생성(Create document)] 드롭다운에서 [Automation]을 선택합니다.
4. 이름 필드에 **`PatchAMIAndUpdateASG`**을 입력합니다.

5. 편집기(Editor) 탭을 선택하고 편집(Edit)을 선택합니다.
6. 대화 상자가 표시되면 확인(OK)을 선택하고 문서 편집기(Document editor) 필드에서 콘텐츠를 삭제합니다.
7. 문서 편집기(Document editor) 필드에 다음 YAML 샘플 런북 콘텐츠를 붙여넣습니다.

```
---
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. Create automation(자동화 생성)을 선택합니다.
9. 탐색 창에서 Automation(자동화)을 선택한 후 Execute automation(자동화 실행)을 선택합니다.
10. 문서 선택(Choose document) 페이지에서 내 소유(Owned by me) 탭을 선택합니다.
11. PatchAMIAndUpdateASG 런북을 찾고 PatchAMIAndUpdateASG 카드에서 버튼을 선택합니다.
12. 다음을 선택합니다.
13. Simple execution(단순 실행)을 선택합니다.
14. 입력 파라미터에 대한 값을 지정합니다. 지정한 `SubnetId` 및 `SecurityGroupIds`가 퍼블릭 Systems Manager 엔드포인트 또는 Systems Manager용 인터페이스 엔드포인트에 대한 액세스를 허용하도록 해야 합니다.
15. 실행을 선택합니다.
16. 자동화가 완료된 후 Amazon EC2 콘솔에서 Auto Scaling을 선택한 다음 시작 템플릿(Launch Templates)을 선택합니다. 새로운 시작 템플릿이 표시되고 새로운 AMI가 사용되는지 확인합니다.
17. [Auto Scaling]을 선택한 다음 [Auto Scaling 그룹(Auto Scaling Groups)]을 선택합니다. Auto Scaling 그룹에서 새로운 시작 템플릿이 사용되는지 확인합니다.
18. Auto Scaling 그룹에서 하나 이상의 인스턴스를 종료합니다. 대체 인스턴스가 새로운 AMI를 사용하여 시작됩니다.

AWS Support 셀프 서비스 런북 사용

이 섹션에서는 AWS Support 팀에서 만든 몇 가지 셀프 서비스 자동화를 사용하는 방법을 설명합니다. 이러한 자동화는 AWS 리소스를 관리하는 데 도움이 됩니다.

Support Automation Workflows

SAW(Support Automation Workflows)는 AWS Support 팀에서 작성하고 유지 관리하는 Automation 실행서입니다. 이러한 실행서는 AWS 리소스의 일반적인 문제를 해결하고, 네트워크 문제를 사전에 모니터링 및 식별하고, 로그를 수집 및 분석하는 등의 작업을 수행하는 데 도움이 됩니다.

SAW 실행서는 **AWSSupport** 접두사를 사용합니다. 예를 들면 AWSSupport-ActivateWindowsWithAmazonLicense입니다.

또한 AWS Enterprise 및 Business Support 고객은 **AWSPremiumSupport** 접두사를 사용하는 실행서에도 액세스할 수 있습니다. 예를 들면 AWSPremiumSupport-TroubleshootEC2DiskUsage입니다.

AWS Support에 대한 자세한 내용은 AWS Support 시작하기를 참조하세요.

주제

- 연결할 수 없는 인스턴스에서 EC2Rescue 도구 실행
- EC2 인스턴스에서 암호 및 SSH 키 재설정

연결할 수 없는 인스턴스에서 EC2Rescue 도구 실행

EC2Rescue를 사용하면 Linux 및 Windows Server용 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에 대한 문제를 진단하고 해결할 수 있습니다. Linux Server용 EC2Rescue 사용 및 Windows Server용 EC2Rescue 사용에서 설명한 대로 이 도구를 직접 실행할 수 있습니다. 또는 Systems Manager Automation 및 **AWSSupport-ExecuteEC2Rescue** 실행서를 사용하여 도구를 자동으로 실행할 수 있습니다. Automation은 AWS Systems Manager의 기능입니다. **AWSSupport-ExecuteEC2Rescue** 실행서는 EC2Rescue를 사용하는 데 일반적으로 필요한 단계를 자동화하는 Systems Manager 작업, AWS CloudFormation 작업 및 Lambda 함수를 조합하여 수행하도록 설계되었습니다.

AWSSupport-ExecuteEC2Rescue 실행서를 사용하여 다양한 유형의 운영 체제(OS) 문제를 해결하고, 수정할 수도 있습니다. 루트 볼륨이 암호화된 인스턴스는 지원되지 않습니다. 전체 목록은 다음 항목을 참조하십시오.

Windows: 명령줄에서 EC2Rescue for Windows Server 사용의 복구 작업을 참조하세요.

Linux 및 macOS: 일부 Linux용 EC2Rescue 모듈은 문제를 감지하여 수정을 시도합니다. 자세한 내용은 GitHub에서 각 모델에 대한 aws-ec2rescue-linux 설명서를 참조하세요.

작동 방식

Automation과 **AWSSupport-ExecuteEC2Rescue** 실행서를 사용한 인스턴스 문제 해결은 다음과 같이 수행됩니다.

- 연결할 수 없는 인스턴스의 ID를 지정하고 자동화를 시작합니다.

- 시스템에서 임시 VPC를 만든 다음, 일련의 Lambda 함수를 실행하여 VPC를 구성합니다.
- 시스템에서 원본 인스턴스와 동일한 가용 영역에 있는 임시 VPC의 서브넷을 식별합니다.
- 시스템에서 SSM이 활성화된 임시 헬퍼 인스턴스를 시작합니다.
- 시스템에서 원본 인스턴스를 중지한 후 백업을 만듭니다. 그런 다음 헬퍼 인스턴스에 원본 루트 볼륨을 연결합니다.
- 시스템에서 Run Command를 사용하여 헬퍼 인스턴스에서 EC2Rescue를 실행합니다. EC2Rescue는 연결된 원본 루트 볼륨의 문제를 확인하고 수정하려 시도합니다. 마치면 EC2Rescue가 원본 인스턴스에 루트 볼륨을 다시 연결합니다.
- 시스템에서 원본 인스턴스를 다시 시작하고, 임시 인스턴스를 종료합니다. 또한 자동화가 시작될 때 생성된 Lambda 함수와 임시 VPC를 종료합니다.

시작하기 전 준비 사항

다음 자동화를 실행하기 전에 다음을 수행합니다.

- 연결할 수 없는 인스턴스의 인스턴스 ID를 복사합니다. 절차에서 이 ID를 지정할 것입니다.
- 원할 경우, 연결할 수 없는 인스턴스와 동일한 가용 영역에 있는 서브넷 ID를 수집할 수도 있습니다. 이 서브넷에 EC2Rescue 인스턴스가 생성됩니다. 서브넷을 지정하지 않으면 Automation 프로세스에 따라 AWS 계정에 새로운 임시 VPC가 생성됩니다. AWS 계정에 사용 가능한 VPC가 하나 이상 있는지 확인합니다. 기본적으로 리전 하나에 VPC를 다섯 개 만들 수 있습니다. 리전에 이미 VPC를 다섯 개 만든 경우, 자동화가 인스턴스를 변경하지 않고 실패합니다. Amazon VPC 할당량에 대한 자세한 내용은 Amazon VPC 사용 설명서의 VPC 및 서브넷을 참조하세요.
- 필요에 따라 Automation의 AWS Identity and Access Management(IAM) 역할을 생성하고 지정할 수 있습니다. 이 역할을 지정하지 않으면 자동화가 자동화를 실행한 사용자의 맥락에서 실행됩니다.

AWSSupport-EC2Rescue에 인스턴스에서 작업을 수행할 권한 부여

EC2Rescue는 자동화 중에 인스턴스에서 일련의 작업을 수행할 권한이 필요합니다. 이러한 작업은 AWS Lambda, IAM 및 Amazon EC2 서비스를 호출하여 인스턴스 관련 문제를 안전하게 해결하려고 시도합니다. AWS 계정 및/또는 VPC에서 관리자 수준 권한이 있는 경우 이 섹션의 설명과 같이 권한을 구성하지 않고 자동화를 실행할 수 있습니다. 관리자 수준 권한이 없는 경우, 사용자 또는 관리자가 다음 옵션 중 하나를 사용하여 권한을 구성해야 합니다.

- IAM 정책을 사용하여 권한 부여
- AWS CloudFormation 템플릿을 사용하여 권한 부여

IAM 정책을 사용하여 권한 부여

사용자, 그룹 또는 역할에 다음 IAM 정책을 인라인 정책으로 연결하거나, 새 IAM 관리형 정책을 생성하고 이를 사용자, 그룹 또는 역할에 연결할 수 있습니다. 사용자, 그룹 또는 역할에 인라인 정책 추가에 대한 자세한 내용은 인라인 정책 작업을 참조하세요. 새 관리형 정책 만들기에 대한 자세한 내용은 관리형 정책 작업을 참조하십시오.

> **Note**
>
> 새 IAM 관리형 정책을 생성하는 경우 인스턴스가 Systems Manager API와 통신할 수 있도록 AmazonSSMAutomationRole 관리형 정책도 여기에 연결해야 합니다.

AWSSupport-EC2Rescue의 IAM 정책

*## ID*를 자신의 정보로 바꿉니다.

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

AWS CloudFormation 템플릿을 사용하여 권한 부여

AWS CloudFormation은 미리 구성된 템플릿을 사용하여 IAM 역할과 정책을 만드는 과정을 자동화합니다. 다음 절차에 따라 AWS CloudFormation을 사용하여 EC2Rescue Automation에 필요한 IAM 역할과 정책을 생성합니다.

EC2Rescue에 필요한 IAM 역할과 정책을 생성하려면

1. AWSSupport-EC2RescueRole.zip을 다운로드하여 로컬 시스템의 디렉터리에 AWSSupport-EC2RescueRole.json 파일의 압축을 풉니다.
2. AWS 계정가 특수 파티션인 경우 템플릿을 편집하여 ARN 값을 파티션 값으로 변경합니다.

   예를 들어 중국 리전의 경우 arn:aws의 모든 사례를 arn:aws-cn으로 변경합니다.
3. AWS Management Console에 로그인하여 https://console.aws.amazon.com/cloudformation에서 AWS CloudFormation 콘솔을 엽니다.
4. 스택 생성(Create stack), 새 리소스 사용(표준)(With new resources (standard))를 선택합니다.
5. 스택 생성 페이지에서 Prerequisite - Prepare template(사전 조건 - 템플릿 준비)에 대해 Template is ready(템플릿 준비됨)를 선택합니다.
6. 템플릿 지정에서 템플릿 파일 업로드를 선택합니다.
7. 파일 선택을 선택한 다음 압축을 푼 디렉터리에서 AWSSupport-EC2RescueRole.json 파일을 찾아 선택합니다.
8. 다음을 선택합니다.
9. Specify stack details(스택 세부 정보 지정) 페이지의 스택 이름 필드에 이 스택을 식별할 이름을 입력하고 다음을 선택합니다.
10. (선택 사항) 태그 영역에서 하나 이상의 태그 키 이름/값 쌍을 스택에 적용합니다.

    태그는 리소스에 할당하는 선택적 메타데이터입니다. 태그를 사용하면 용도, 소유자 또는 환경을 기준으로 하는 등 리소스를 다양한 방식으로 분류할 수 있습니다. 예를 들어, 스택에 태그를 지정하여 실행되는 작업 유형, 대상 유형이나 기타 관련 리소스, 해당 스택이 실행되는 환경을 식별할 수 있습니다.
11. 다음(Next)을 선택합니다.

12. 검토 페이지에서 스택 세부 정보를 검토하고 아래로 스크롤하여 AWS CloudFormation에서 IAM 리소스를 생성할 수 있음을 승인합니다 옵션을 선택합니다.
13. 스택 생성을 선택합니다.

    AWS CloudFormation에 몇 분간 CREATE_IN_PROGRESS 상태가 표시됩니다. 스택이 생성된 후 상태가 CREATE_COMPLETE로 바뀝니다. 새로 고침 아이콘을 선택하여 생성 프로세스의 상태를 확인할 수도 있습니다.
14. 스택 목록에서 방금 생성한 스택 옆의 옵션 버튼을 선택한 후 출력 탭을 선택합니다.
15. 값을 기록해 둡니다. 이는 AssumeRole의 ARN입니다. 자동화 실행 단원에 나온 절차에서 자동화를 실행할 때 이 ARN을 지정합니다.

자동화 실행

> **Important**
>
> 다음 자동화는 연결할 수 없는 인스턴스를 중지합니다. 인스턴스가 중지되면 연결된 인스턴스 스토어 볼륨(있는 경우)의 데이터가 손실될 수 있습니다. 또한 인스턴스가 중지되면 연결된 탄력적 IP가 없는 경우 퍼블릭 IP가 변경될 수도 있습니다.

AWSSupport-ExecuteEC2Rescue Automation을 실행하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 왼쪽 탐색 창에서 Automation을 선택합니다.
3. 자동화 실행(Execute automation)을 선택합니다.
4. 자동화 문서 섹션의 목록에서 Owned by Amazon(Amazon 소유)을 선택합니다.
5. 실행서 목록에서 [**AWSSupport-ExecuteEC2Rescue**]의 카드에 있는 버튼을 선택하고 [다음(Next)]을 선택합니다.
6. Execute automation document(자동화 문서 실행) 페이지에서 Simple execution(단순 실행)을 선택합니다.
7. 문서 세부 정보 섹션에서 문서 버전이 가장 높은 기본 버전으로 설정되어 있는지 확인합니다. 예를 들어, $DEFAULT 또는 3(기본값)입니다.
8. 입력 파라미터 섹션에서 다음 파라미터를 지정합니다.

    a. UnreachableInstanceId에서 연결할 수 없는 인스턴스의 ID를 지정합니다.

b. (선택 사항) EC2RescueInstanceType에서 EC2Rescue 인스턴스의 인스턴스 유형을 지정합니다. 기본 인스턴스 유형은 `t2.medium`입니다.

c. AutomationAssumeRole의 경우 이 주제의 앞부분에서 설명한 AWS CloudFormation 절차를 사용하여 이 자동화에 대한 역할을 생성한 경우 AWS CloudFormation 콘솔에서 생성한 AssumeRole의 ARN을 선택합니다.

d. (선택 사항) LogDestination에서 인스턴스의 문제를 해결하는 동안 운영 체제 수준 로그를 수집하려면 S3 버킷을 지정합니다. 지정된 버킷으로 로그가 자동으로 업로드됩니다.

e. SubnetId에서, 연결할 수 없는 인스턴스와 동일한 가용 영역에 있는 기존 VPC의 서브넷을 지정합니다. 기본적으로 Systems Manager는 새로운 VPC를 생성하지만, 원하는 사용자는 기존 VPC의 서브넷을 지정할 수 있습니다.

> **Note**
>
> 버킷이나 서브넷 ID를 지정하는 옵션이 표시되지 않는 경우 실행서의 최신 기본 버전을 사용하고 있는지 확인합니다.

9. (옵션) [태그(Tags)] 영역에서 하나 이상의 태그 키 이름/값 페어(예: `Key=Purpose,Value=EC2Rescue`)를 적용하여 자동화를 식별할 수 있습니다.
10. 실행을 선택합니다.

실행서가 자동화의 일부로 백업 AMI를 생성합니다. 자동화에 의해 생성된 다른 모든 리소스는 자동으로 삭제되지만, 이 AMI는 계정에 계속 유지됩니다. 다음 명명 규칙을 사용하여 AMI 이름이 지정됩니다.

백업 AMI: AWSSupport-EC2Rescue:*UnreachableInstanceId*

Automation 실행 ID를 검색하여 Amazon EC2 콘솔에 있는 이 AMI를 찾을 수 있습니다.

EC2 인스턴스에서 암호 및 SSH 키 재설정

`AWSSupport-ResetAccess` 실행서를 사용하여 Windows Server용 Amazon Elastic Compute Cloud Amazon EC2 인스턴스의 로컬 관리자 암호 생성을 자동으로 재활성화하고, Linux용 EC2 인스턴스에 새로운 SSH 키를 생성할 수 있습니다. `AWSSupport-ResetAccess` 실행서는 로컬 관리자 암호를 재설정하는 데 일반적으로 필요한 단계를 자동화하는 AWS Systems Manager 작업, AWS CloudFormation 작업 및 AWS Lambda 함수를 조합하여 수행하도록 설계되었습니다.

`AWSSupport-ResetAccess` 실행서와 함께 AWS Systems Manager의 기능인 Automation을 사용하여 다음 문제를 해결할 수 있습니다.

Windows

EC2 키 페어 분실: 이 문제를 해결하려면 AWSSupport-ResetAccess 실행서를 사용하여 현재 인스턴스에서 암호 지원 AMI를 생성한 후 이 AMI에서 인스턴스를 새로 시작하고 키 페어를 선택합니다.

로컬 관리자 암호 분실: 이 문제를 해결하려면 `AWSSupport-ResetAccess` 실행서를 사용하여, 현재 EC2 키 페어로 복호화할 수 있는 새로운 암호를 생성합니다.

Linux

EC2 키 페어를 분실한 경우 또는 분실한 키를 사용하여 인스턴스에 대한 SSH 액세스를 구성한 경우: 이 문제를 해결하려면 `AWSSupport-ResetAccess` 실행서를 사용하여 현재 인스턴스에 대한 SSH 키를 새로 생성합니다. 이렇게 하면 인스턴스에 다시 연결할 수 있습니다.

> **Note**
>
> Windows Server용 EC2 인스턴스가 Systems Manager용으로 구성된 경우 EC2Rescue 및 AWS Systems Manager Run Command를 사용하여 로컬 관리자 암호를 재설정할 수도 있습니다. 자세한 내용은 Amazon EC2 사용 설명서의 Systems Manager Run Command를 통해 Windows Server용 EC2Rescue 사용를 참조하세요.

관련 정보

Amazon EC2 사용 설명서의 PuTTY를 사용하여 Windows에서 Linux 인스턴스에 연결

작동 방식

Automation과 `AWSSupport-ResetAccess` 실행서를 사용한 인스턴스 문제 해결은 다음과 같이 수행됩니다.

- 인스턴스의 ID를 지정하고 실행서를 실행합니다.
- 시스템에서 임시 VPC를 만든 다음, 일련의 Lambda 함수를 실행하여 VPC를 구성합니다.
- 시스템에서 원본 인스턴스와 동일한 가용 영역에 있는 임시 VPC의 서브넷을 식별합니다.
- 시스템에서 SSM이 활성화된 임시 헬퍼 인스턴스를 시작합니다.
- 시스템에서 원본 인스턴스를 중지한 후 백업을 만듭니다. 그런 다음 헬퍼 인스턴스에 원본 루트 볼륨을 연결합니다.
- 시스템에서 Run Command를 사용하여 헬퍼 인스턴스에서 EC2Rescue를 실행합니다. Windows에서 EC2Rescue는 연결된, 원래 루트 볼륨에 있는 EC2Config 또는 EC2Launch를 사용하여 로컬 관리

자에 대한 암호 생성을 활성화합니다. Linux에서, EC2Rescue는 새로운 SSH 키를 생성하여 투입하고, 암호화된 프라이빗 키를 Parameter Store에 저장합니다. 마치면 EC2Rescue가 원본 인스턴스에 루트 볼륨을 다시 연결합니다.

- 암호 생성이 활성화되어 있으므로 시스템에서 인스턴스의 새 Amazon Machine Image(AMI)를 생성합니다. 이 AMI를 사용하여 새 EC2 인스턴스를 생성하고 필요 시 새 키 페어를 연결할 수 있습니다.
- 시스템에서 원본 인스턴스를 다시 시작하고, 임시 인스턴스를 종료합니다. 또한 자동화가 시작될 때 생성된 Lambda 함수와 임시 VPC를 종료합니다.
- Windows: 해당 인스턴스가 인스턴스에 지정된 현재 키 페어를 사용하여 Amazon EC2 콘솔로부터 디코딩할 수 있는 새로운 암호를 생성합니다.

  Linux: Systems Manager 파라미터 스토어에 /ec2rl/openssh/***instance ID***/key로 저장된 SSH 키를 사용하여 인스턴스에 SSH로 액세스할 수 있습니다.

시작하기 전 준비 사항

다음 자동화를 실행하기 전에 다음을 수행합니다.

- 관리자 암호를 재설정하려는 인스턴스의 인스턴스 ID를 복사합니다. 절차에서 이 ID를 지정할 것입니다.
- 원할 경우, 연결할 수 없는 인스턴스와 동일한 가용 영역에 있는 서브넷 ID를 수집할 수도 있습니다. 이 서브넷에 EC2Rescue 인스턴스가 생성됩니다. 서브넷을 지정하지 않으면 Automation 프로세스에 따라 AWS 계정에 새로운 임시 VPC가 생성됩니다. AWS 계정에 사용 가능한 VPC가 하나 이상 있는지 확인합니다. 기본적으로 리전 하나에 VPC를 다섯 개 만들 수 있습니다. 리전에 이미 VPC를 다섯 개 만든 경우, 자동화가 인스턴스를 변경하지 않고 실패합니다. Amazon VPC 할당량에 대한 자세한 내용은 Amazon VPC 사용 설명서의 VPC 및 서브넷을 참조하세요.
- 필요에 따라 Automation의 AWS Identity and Access Management(IAM) 역할을 생성하고 지정할 수 있습니다. 이 역할을 지정하지 않으면 자동화가 자동화를 실행한 사용자의 맥락에서 실행됩니다.

AWSSupport-EC2Rescue에 인스턴스에서 작업을 수행할 권한 부여

EC2Rescue는 자동화 중에 인스턴스에서 일련의 작업을 수행할 권한이 필요합니다. 이러한 작업은 AWS Lambda, IAM 및 Amazon EC2 서비스를 호출하여 인스턴스 관련 문제를 안전하게 해결하려고 시도합니다. AWS 계정 및/또는 VPC에서 관리자 수준 권한이 있는 경우 이 섹션의 설명과 같이 권한을 구성하지 않고 자동화를 실행할 수 있습니다. 관리자 수준 권한이 없는 경우, 사용자 또는 관리자가 다음 옵션 중 하나를 사용하여 권한을 구성해야 합니다.

- IAM 정책을 사용하여 권한 부여

- AWS CloudFormation 템플릿을 사용하여 권한 부여

IAM 정책을 사용하여 권한 부여

사용자, 그룹 또는 역할에 다음 IAM 정책을 인라인 정책으로 연결하거나, 새 IAM 관리형 정책을 생성하고 이를 사용자, 그룹 또는 역할에 연결할 수 있습니다. 사용자, 그룹 또는 역할에 인라인 정책 추가에 대한 자세한 내용은 인라인 정책 작업을 참조하세요. 새 관리형 정책 만들기에 대한 자세한 내용은 관리형 정책 작업을 참조하십시오.

> **Note**
>
> 새 IAM 관리형 정책을 생성하는 경우 인스턴스가 Systems Manager API와 통신할 수 있도록 AmazonSSMAutomationRole 관리형 정책도 여기에 연결해야 합니다.

`AWSSupport-ResetAccess`에 대한 IAM 정책

*## ID*를 자신의 정보로 바꿉니다.

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

AWS CloudFormation 템플릿을 사용하여 권한 부여

AWS CloudFormation은 미리 구성된 템플릿을 사용하여 IAM 역할과 정책을 만드는 과정을 자동화합니다. 다음 절차에 따라 AWS CloudFormation을 사용하여 EC2Rescue Automation에 필요한 IAM 역할과 정책을 생성합니다.

EC2Rescue에 필요한 IAM 역할과 정책을 생성하려면

1. AWSSupport-EC2RescueRole.zip을 다운로드하여 로컬 시스템의 디렉터리에 AWSSupport-EC2RescueRole.json 파일의 압축을 풉니다.
2. AWS 계정가 특수 파티션인 경우 템플릿을 편집하여 ARN 값을 파티션 값으로 변경합니다.

   예를 들어 중국 리전의 경우 arn:aws의 모든 사례를 arn:aws-cn으로 변경합니다.
3. AWS Management Console에 로그인하여 https://console.aws.amazon.com/cloudformation에서 AWS CloudFormation 콘솔을 엽니다.
4. 스택 생성(Create stack), 새 리소스 사용(표준)(With new resources (standard))를 선택합니다.
5. 스택 생성 페이지에서 Prerequisite - Prepare template(사전 조건 - 템플릿 준비)에 대해 Template is ready(템플릿 준비됨)를 선택합니다.
6. 템플릿 지정에서 템플릿 파일 업로드를 선택합니다.
7. 파일 선택을 선택한 다음 압축을 푼 디렉터리에서 AWSSupport-EC2RescueRole.json 파일을 찾아 선택합니다.
8. Next(다음)를 선택합니다.
9. Specify stack details(스택 세부 정보 지정) 페이지의 스택 이름 필드에 이 스택을 식별할 이름을 입력하고 다음을 선택합니다.
10. (선택 사항) 태그 영역에서 하나 이상의 태그 키 이름/값 쌍을 스택에 적용합니다.

    태그는 리소스에 할당하는 선택적 메타데이터입니다. 태그를 사용하면 용도, 소유자 또는 환경을 기준으로 하는 등 리소스를 다양한 방식으로 분류할 수 있습니다. 예를 들어, 스택에 태그를 지정하여 실행되는 작업 유형, 대상 유형이나 기타 관련 리소스, 해당 스택이 실행되는 환경을 식별할 수 있습니다.

11. 다음(Next)을 선택합니다.
12. 검토 페이지에서 스택 세부 정보를 검토하고 아래로 스크롤하여 AWS CloudFormation에서 IAM 리소스를 생성할 수 있음을 승인합니다 옵션을 선택합니다.
13. AWS CloudFormation에 몇 분간 CREATE_IN_PROGRESS 상태가 표시됩니다. 스택이 생성된 후 상태가 CREATE_COMPLETE로 바뀝니다. 새로 고침 아이콘을 선택하여 생성 프로세스의 상태를 확인할 수도 있습니다.
14. 스택 목록에서 방금 생성한 스택 옆에 있는 옵션을 선택한 다음, 출력 탭을 선택합니다.
15. 값을 복사합니다. 이는 AssumeRole의 ARN입니다. 자동화를 실행할 때 이 ARN을 지정합니다.

자동화 실행

다음 절차에서는 AWS Systems Manager 콘솔을 사용하여 `AWSSupport-ResetAccess` 실행서를 실행하는 방법을 설명합니다.

> **Important**
>
> 다음 자동화는 인스턴스를 중지합니다. 인스턴스가 중지되면 연결된 인스턴스 스토어 볼륨(있는 경우)의 데이터가 손실될 수 있습니다. 또한 인스턴스가 중지되면 연결된 탄력적 IP가 없는 경우 퍼블릭 IP가 변경될 수도 있습니다. 이렇게 구성이 변경되지 않도록 하려면 Run Command를 사용하여 액세스를 재설정하십시오. 자세한 내용은 Amazon EC2 사용 설명서의 Systems Manager Run Command를 통해 Windows Server용 EC2Rescue 사용를 참조하세요.

AWSSupport-ResetAccess 자동화를 실행하는 방법

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 왼쪽 탐색 창에서 Automation을 선택합니다.
3. 자동화 실행(Execute automation)을 선택합니다.
4. 자동화 문서 섹션의 목록에서 Owned by Amazon(Amazon 소유)을 선택합니다.
5. 실행서 목록에서 [AWSSupport-ResetAccess]의 카드에 있는 버튼을 선택하고 [다음(Next)]을 선택합니다.
6. Execute automation document(자동화 문서 실행) 페이지에서 Simple execution(단순 실행)을 선택합니다.
7. 문서 세부 정보 섹션에서 문서 버전이 가장 높은 기본 버전으로 설정되어 있는지 확인합니다. 예를 들어, $DEFAULT 또는 3(기본값)입니다.

8. 입력 파라미터 섹션에서 다음 파라미터를 지정합니다.

   a. InstanceID에서 연결할 수 없는 인스턴스의 ID를 지정합니다.

   b. SubnetId에서, 지정한 인스턴스와 동일한 가용 영역에 있는 기존 VPC의 서브넷을 지정합니다. 기본적으로 Systems Manager는 새로운 VPC를 생성하지만, 원하는 사용자는 기존 VPC의 서브넷을 지정할 수 있습니다.

   > **Note**
   >
   > 서브넷 ID를 지정하는 옵션이 표시되지 않는 경우 실행서의 최신 [기본(Default)] 버전을 사용하고 있는지 확인합니다.

   c. EC2RescueInstanceType에서 EC2Rescue 인스턴스의 인스턴스 유형을 지정합니다. 기본 인스턴스 유형은 `t2.medium`입니다.

   d. [AssumeRole]의 경우 이 주제의 앞부분에서 설명한 AWS CloudFormation 절차를 사용하여 이 Automation에 대한 역할을 생성한 경우 AWS CloudFormation 콘솔에서 기록한 AssumeRole ARN을 지정합니다.

9. (옵션) [태그(Tags)] 영역에서 하나 이상의 태그 키 이름/값 페어(예: `Key=Purpose,Value=ResetAccess`)를 적용하여 자동화를 식별할 수 있습니다.

10. 실행을 선택합니다.

11. 자동화 진행 상황을 모니터링하려면 실행 중인 자동화를 선택한 후 [단계(Steps)] 탭을 선택합니다. 자동화가 종료되면 [설명(Descriptions)] 탭을 선택한 후 [출력 보기(View output)]를 선택하여 결과를 봅니다. 개별 단계의 출력을 보려면 단계 탭을 선택한 다음, 단계 옆에 있는 출력 보기를 선택합니다.

실행서가 자동화의 일부로 백업 AMI와 암호 사용 AMI를 생성합니다. 자동화에 의해 생성된 다른 모든 리소스는 자동으로 삭제되지만, 이러한 AMIs는 계정에 계속 유지됩니다. AMIs는 다음 명명 규칙을 사용하여 이름이 지정됩니다.

- 백업 AMI: `AWSSupport-EC2Rescue:`*`InstanceID`*
- 암호가 설정된 AMI: AWSSupport-EC2Rescue: *`##### ID`*의 암호가 설정된 AMI

Automation 실행 ID를 검색하여 이 AMIs를 찾을 수 있습니다.

Linux의 경우 인스턴스의 새로운 SSH 프라이빗 키는 암호화되어 Parameter Store에 저장됩니다. 파라미터 이름은 /ec2rl/openssh/***`instance ID`***/key입니다.

입력 변환기를 사용하여 Automation에 데이터 전달

이 AWS Systems Manager Automation 자습서에서는 Amazon EventBridge의 입력 변환기 기능을 사용하여 인스턴스 상태 변경 이벤트에서 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스의 `instance-id`를 추출하는 방법을 보여줍니다. Automation은 AWS Systems Manager의 기능입니다. 입력 변환기를 사용하여 `AWS-CreateImage` 실행서 대상에 `InstanceId` 입력 파라미터로 해당 데이터를 전달합니다. 인스턴스 상태가 `stopped`로 변경될 때 규칙이 트리거됩니다.

입력 변환기 작업에 대한 자세한 내용은 Amazon EventBridge User Guide의 Tutorial: Use Input Transformer to Customize What is Passed to the Event Target을 참조하세요.

시작하기 전 준비 사항

Systems Manager Automation 서비스 역할에 EventBridge에 필요한 권한 및 신뢰 정책을 추가했는지 확인합니다. 자세한 내용은 Amazon EventBridge User Guide의 Overview of Managing Access Permissions to Your EventBridge Resources를 참조하세요.

자동화와 함께 입력 변환기를 사용하려면

1. https://console.aws.amazon.com/events/에서 Amazon EventBridge 콘솔을 엽니다.
2. 탐색 창에서 규칙을 선택합니다.
3. 규칙 생성을 선택합니다.
4. 규칙에 대해 이름과 설명을 입력하십시오.

   규칙은 동일한 지역과 동일한 이벤트 버스의 다른 규칙과 동일한 이름을 가질 수 없습니다.
5. 이벤트 버스에서 이 규칙과 연결할 이벤트 버스를 선택합니다. 이 규칙이 자신의 AWS 계정에서 오는 일치하는 이벤트에 응답하도록 하려면 default(기본)를 선택합니다. 계정의 AWS 서비스이(가) 이벤트를 출력하면 항상 계정의 기본 이벤트 버스로 이동합니다.
6. 규칙 유형에서 이벤트 패턴이 있는 규칙을 선택합니다.
7. 다음을 선택합니다.
8. 이벤트 소스(Event source)에서 AWS 이벤트 또는 EventBridge 파트너 이벤트(Events or EventBridge partner events)를 선택합니다.
9. 이벤트 패턴(Event pattern)섹션에서 이벤트 패턴 양식(Event pattern form)을 선택합니다.
10. 이벤트 소스(Event source)에서 AWS 서비스(services)를 선택합니다.
11. AWS service(서비스)에서 EC2를 선택합니다.
12. 이벤트 유형에서 EC2 인스턴스 상태 변경 알림을 선택합니다.
13. Specific state(특정 상태)에서 stopped(중지됨)을 선택합니다.

14. 다음을 선택합니다.
15. 대상 유형에서 AWS서비스를 선택합니다.
16. Target(대상)에서 Systems Manager Automation을 선택합니다.
17. 문서에 대해 AWS-CreateImage를 선택합니다.
18. Configure automation parameter(s)(자동화 파라미터 구성) 섹션에서 Input Transformer(입력 변환기)를 선택합니다.
19. Input path(입력 경로)에서 **`{"instance":"$.detail.instance-id"}`**을(를) 입력합니다.
20. Template(템플릿)에서 **`{"InstanceId":[<instance>]}`**을(를) 입력합니다.
21. Execution role(실행 역할)에서 Use existing role(기존 역할 사용)을 선택하고 Automation 서비스 역할을 선택합니다.
22. 다음을 선택합니다.
23. (선택 사항) 규칙에 대해 하나 이상의 태그를 입력하십시오. 자세한 내용은 Amazon EventBridge User Guide의 Tagging Your Amazon EventBridge Resources를 참조하세요.
24. 다음을 선택합니다.
25. 규칙의 세부 정보를 검토하고 규칙 생성을 선택합니다.

자동화 상태 이해

AWS Systems Manager Automation은 자동화를 실행할 때 및 전체 자동화에 대해 자동화 작업 또는 단계가 거치는 다양한 상태에 대한 자세한 상태 정보를 보고합니다. Automation은 AWS Systems Manager의 기능입니다. 다음 방법을 사용하여 자동화 상태를 모니터링할 수 있습니다.

- Systems Manager Automation 콘솔에서 [실행 상태(Execution status)]를 모니터링합니다.
- 선호하는 명령줄 도구를 사용합니다. AWS Command Line Interface(AWS CLI)에 대해 describe-automation-step-executions 또는 get-automation-execution을 사용할 수 있습니다. AWS Tools for Windows PowerShell에 대해 Get-SSMAutomationStepExecution 또는 Get-SSMAutomationExecution을 사용할 수 있습니다.
- 작업 또는 자동화 상태 변경에 응답하도록 Amazon EventBridge를 구성합니다.

자동화에서의 타임아웃 처리에 대한 자세한 내용은 실행서에서 시간 제한 처리 섹션을 참조하세요.

자동화 상태 정보

Automation은 전체 자동화 외에도 개별 자동화 작업에 대한 상태 세부 정보를 보고합니다.

전체 자동화 상태는 다음 표와 같이 개별 작업 또는 단계에서 보고되는 상태와 다를 수 있습니다.

작업의 세부 상태

상태 표시기	Details
보류중	단계 실행이 시작되지 않았습니다. 자동화에서 조건부 작업을 사용하는 경우 단계를 실행하기 위한 조건이 충족되지 않으면 자동화가 완료된 후 단계가 이 상태로 유지됩니다. 단계가 실행되기 전에 자동화가 취소된 경우에도 단계는 이 상태로 유지됩니다.
InProgress	작업이 실행 중입니다.
대기 중(Waiting)	단계가 입력을 기다리고 있습니다.
Success	단계가 성공적으로 완료되었습니다. 이것은 종료 상태입니다.
TimedOut	지정된 시간 제한 기간 전에 단계 또는 승인이 완료되지 않았습니다. 이것은 종료 상태입니다.
취소 중	요청자가 취소한 후 단계를 중지하는 중입니다.
취소됨	단계가 완료되기 전에 요청자가 단계를 중지했습니다. 이것은 종료 상태입니다.
Failed	단계가 성공적으로 완료되지 않았습니다. 이것은 종료 상태입니다.
Exited	`aws:loop` 작업에 의해서만 반환됩니다. 루프가 완전히 완료되지 않았습니다. `nextStep`, `onCancel` 또는 `onFailure` 속성을 사용하여 루프 내부의 단계가 외부 단계로 이동했습니다.

자동화에 대한 세부 상태

상태 표시기	Details
보류중	자동화 실행이 시작되지 않았습니다.
InProgress	자동화가 실행 중입니다.
대기 중(Waiting)	자동화가 입력을 기다리고 있습니다.
Success	자동화가 성공적으로 완료되었습니다. 이것은 종료 상태입니다.
TimedOut	지정된 시간 제한 기간 전에 단계 또는 승인이 완료되지 않았습니다. 이것은 종료 상태입니다.
취소 중	요청자가 취소한 후 자동화를 중지하는 중입니다.
취소됨	자동화가 완료되기 전에 요청자가 자동화를 중지했습니다. 이것은 종료 상태입니다.
Failed	자동화가 성공적으로 완료되지 않았습니다. 이것은 종료 상태입니다.

Systems Manager Automation 문제 해결

다음 정보를 사용하면 AWS Systems Manager의 기능인 AWS Systems Manager Automation 서비스 문제를 해결하는 데 도움이 됩니다. 이 주제에는 자동화 오류 메시지를 기반으로 문제를 해결하기 위한 특정 작업이 포함됩니다.

주제

일반적인 Automation 오류

이 섹션에는 일반적인 자동화 오류에 대한 정보가 나와 있습니다.

VPC not defined 400

기본적으로 Automation이 `AWS-UpdateLinuxAmi` 실행서 또는 `AWS-UpdateWindowsAmi` 실행서를 실행하면 시스템이 기본 VPC(172.30.0.0/16)에 임시 인스턴스를 생성합니다. 기본 VPC를 삭제했다면 다음 오류 메시지를 받게 됩니다.

```
VPC not defined 400
```

이 문제를 해결하려면 `SubnetId` 입력 파라미터의 값을 지정해야 합니다.

Automation 실행 시작이 실패함

Automation에 대한 AWS Identity and Access Management(IAM) 역할 및 정책을 제대로 구성하지 않은 경우 자동화는 액세스 거부 오류 또는 유효하지 않은 수임 역할 오류와 함께 실패할 수 있습니다.

액세스 거부됨

다음 예제는 자동화 시작이 액세스 거부 오류와 함께 실패한 경우에 대해 설명합니다.

Systems Manager API에 대한 액세스가 거부됨

오류 메시지: `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)`

- 가능한 원인 1: 자동화를 시작하려고 하는 사용자에게 `StartAutomationExecution` API를 호출할 수 있는 권한이 없습니다. 이 문제를 해결하려면 자동화를 시작하는 데 사용된 사용자에게 필요한 IAM 정책을 연결합니다.
- 가능한 원인 2: 자동화를 시작하려고 하는 사용자에게 `StartAutomationExecution` API를 호출할 수 있는 권한이 있지만, 특정 런북을 사용하여 API를 호출할 수 있는 권한은 없습니다. 이 문제를 해결하려면 자동화를 시작하는 데 사용된 사용자에게 필요한 IAM 정책을 연결합니다.

PassRole 권한이 누락되었기 때문에 액세스 거부됨

오류 메시지: `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)`

자동화를 시작하려고 하는 사용자에게 수임 역할에 대한 PassRole 권한이 없습니다. 이 문제를 해결하려면 자동화를 시작하려고 시도하는 사용자의 역할에 iam:PassRole 정책을 연결합니다. 자세한 내용은 작업 2: Automation 역할에 iam:PassRole 정책 연결하기 단원을 참조하십시오.

유효하지 않은 수임 역할

Automation을 실행할 때 assume 역할은 실행서에 제공되거나 해당 실행서에 대한 파라미터 값으로 전달됩니다. 수임 역할이 지정되지 않거나 제대로 구성되지 않으면 다양한 유형의 오류가 발생할 수 있습니다.

형식이 잘못된 Assume Role

오류 메시지: `The format of the supplied assume role ARN isn't valid.` assume role의 형식이 잘못 지정되었습니다. 이 문제를 해결하려면 유효한 수임 역할이 실행서에 지정되어 있는지 또는 자동화 시작 시 런타임 파라미터로 지정되어 있는지 확인합니다.

수임 역할을 수임할 수 없음

오류 메시지: `The defined assume role is unable to be assumed. (Service: AWSSimpleSystemsManagement; Status Code: 400; Error Code: InvalidAutomationExecutionParametersException; Request ID: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx)`

- 가능한 원인 1: 수임 역할이 존재하지 않습니다. 이 문제를 해결하려면 역할을 생성하십시오. 자세한 내용은 the section called “Automation 설정” 단원을 참조하십시오. 이 역할 생성에 대한 특정 정보는 작업 1: Automation을 위한 서비스 역할 생성 주제에 설명되어 있습니다.
- 가능한 원인 2: 수임 역할이 Systems Manager 서비스와 신뢰 관계를 갖지 않습니다. 이 문제를 해결하려면 신뢰 관계를 생성하십시오. 자세한 내용은 IAM User Guide의 I Can't Assume A Role을 참조하세요.

실행이 시작되었지만 상태가 실패인 경우

작업별 오류

실행서에는 단계와 단계 실행이 순서대로 포함되어 있습니다. 각 단계는 하나 이상의 AWS 서비스 API를 호출합니다. API는 해당 단계의 입력, 동작 및 출력을 결정합니다. 단계 실패를 유발할 수 있는 오류는 여러 위치에서 발생합니다. 오류 메시지는 오류가 언제 어디에서 발생했는지 나타냅니다.

Amazon Elastic Compute Cloud(Amazon EC2) 콘솔에서 오류 메시지를 보려면 실패한 단계의 [출력 보기(View Outputs)] 링크를 선택합니다. AWS CLI에서 오류 메시지를 보려면 `get-automation-execution`을 호출하고 실패한 `StepExecution`에서 `FailureMessage` 속성을 찾습니다.

다음 예제에서는 `aws:runInstance` 작업과 연결된 단계가 실패했습니다. 각 예제에서는 다른 유형의 오류를 살펴봅니다.

이미지 누락

오류 메시지: `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` 작업이 존재하지 않는 `ImageId`에 대한 입력을 받았습니다. 이 문제를 해결하려면 실행서 또는 파라미터 값을 정확한 AMI ID로 업데이트합니다.

수임 역할 정책에 충분한 권한이 없음

오류 메시지: `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.`

수임 역할에 EC2 인스턴스에서 `RunInstances` API를 호출할 수 있는 권한이 없습니다. 이 문제를 해결하려면 `RunInstances` API를 호출할 수 있는 권한이 있는 수임 역할에 IAM 정책을 연결합니다. 자세한 내용은 방법 2: IAM을 사용하여 Automation을 위한 역할 구성 단원을 참조하십시오.

예기치 않은 상태

오류 메시지: `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.`

- 가능한 원인 1: 인스턴스 또는 Amazon EC2 서비스에 문제가 있습니다. 이 문제를 해결하려면 인스턴스에 로그인하거나 인스턴스 시스템 로그를 검토하여 인스턴스가 종료되기 시작한 이유를 이해하십시오.
- 가능한 원인 2: `aws:runInstances` 작업에 지정된 사용자 데이터 스크립트에 문제가 있거나 구문이 부정확합니다. 사용자 데이터 스크립트의 구문을 확인하십시오. 또한 사용자 데이터 스크립트가 인스턴스를 종료시키지 않는지, 아니면 인스턴스를 종료시키는 다른 스크립트를 호출하지 않는지 확인하십시오.

작업 특정 오류 참조

단계가 실패하면 해당 오류 메시지가 오류 발생 시 호출 중이었던 서비스를 나타낼 수 있습니다. 다음 표에는 각 작업에 의해 호출된 서비스가 나와 있습니다. 또한 각 서비스에 대한 정보로 연결되는 링크도 제공합니다.

작업	이 작업에 의해 AWS 서비스이(가) 호출됨	이 서비스에 대한 정보	콘텐츠 문제 해결
`aws:runInstances`	Amazon EC2	Amazon EC2 사용 설명서	EC2 인스턴스 문제 해결
`aws:changeInstanceState`	Amazon EC2	Amazon EC2 사용 설명서	EC2 인스턴스 문제 해결
`aws:runCommand`	Systems Manager	AWS Systems Manager Run Command	Systems Manager Run Command 문제 해결
`aws:createImage`	Amazon EC2	Amazon Machine Images	
`aws:createStack`	AWS CloudFormation	AWS CloudFormation 사용 설명서	AWS CloudFormation 문제 해결
`aws:deleteStack`	AWS CloudFormation	AWS CloudFormation 사용 설명서	AWS CloudFormation 문제 해결
`aws:deleteImage`	Amazon EC2	Amazon 머신 이미지	

작업	이 작업에 의해 AWS 서비스이(가) 호출됨	이 서비스에 대한 정보	콘텐츠 문제 해결
aws:copyImage	Amazon EC2	Amazon Machine Images	
aws:createTag	Amazon EC2, Systems Manager	EC2 리소스 및 태그	
aws:invokeLambdaFunction	AWS Lambda	AWS Lambda 개발자 안내서	Lambda 문제 해결

Automation 서비스 내부 오류

오류 메시지: `Internal Server Error. Please refer to Automation Service Troubleshooting Guide for more diagnosis details.`

Automation 서비스 문제로 인해 지정된 실행서가 제대로 실행되지 않습니다. 이 문제를 해결하려면 AWS Support에 연락하십시오. 가능할 경우 실행 ID와 고객 ID를 제공하십시오.

실행이 시작되었지만 시간 초과됨

오류 메시지: `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` 작업의 단계가 시간 초과되었습니다. 이 오류는 단계 작업이 실행되는 시간이 해당 단계의 `timeoutSeconds`에 지정된 값보다 오래 걸리는 경우에 발생할 수 있습니다. 이 문제를 해결하려면 `aws:runInstances` 작업에서 `timeoutSeconds` 파라미터에 더 긴 값을 지정합니다. 이렇게 해도 문제가 해결되지 않으면 단계가 실행되는 시간이 예상보다 오래 걸리는 이유를 조사합니다.

AWS Systems Manager Change Calendar

AWS Systems Manager의 기능인 Change Calendar를 사용하면 지정한 작업(예: Systems Manager Automation 실행서에서)이 AWS 계정에서 수행되거나 수행되지 않을 때 날짜 및 시간 범위를 설정할 수 있습니다. Change Calendar에서는 이러한 범위를 이벤트라고 합니다. Change Calendar 항목을 생성하면 `ChangeCalendar` 유형의 Systems Manager 문서가 생성됩니다. Change Calendar에서 문서는 iCalendar 2.0 데이터를 일반 텍스트 형식으로 저장합니다. Change Calendar 항목에 추가하는 이벤

트는 문서의 일부가 됩니다. Change Calendar를 시작하려면 Systems Manager 콘솔을 엽니다. 탐색 창에서 Change Calendar를 선택합니다.

Systems Manager 콘솔에서 일정 및 해당 이벤트를 생성할 수 있습니다. 또한 지원되는 서드 파티 일정 공급자가 내보낸 iCalendar(`.ics`) 파일을 가져와 일정에 해당 이벤트를 추가할 수 있습니다. 지원되는 공급자에는 Google 캘린더, Microsoft Outlook, iCloud 캘린더가 있습니다.

Change Calendar 항목은 다음 두 가지 유형 중 하나가 될 수 있습니다.

`DEFAULT_OPEN` 또는 Open by default(기본적으로 열림)

일정 이벤트 중일 때를 제외하고 기본적으로 모든 작업이 실행될 수 있습니다. 이벤트 중 `DEFAULT_OPEN` 일정의 상태가 `CLOSED`이고 이벤트 실행이 차단됩니다.

`DEFAULT_CLOSED` 또는 Closed by default(기본적으로 닫힘)

일정 이벤트 중일 때를 제외하고 기본적으로 모든 작업이 차단됩니다. 이벤트 중 `DEFAULT_CLOSED` 일정의 상태가 `OPEN`이고 작업 실행이 허용됩니다.

예약된 모든 자동화 워크플로, 유지 관리 기간 및 State Manager 연결이 캘린더에 자동으로 추가되도록 선택할 수 있습니다. 캘린더 표시에서 해당 개별 유형을 모두 제거할 수도 있습니다.

Change Calendar는 누가 사용해야 하나요?

- 다음과 같은 작업 유형을 수행하는 AWS 고객:
  - 자동화 런북을 생성하거나 실행합니다.
  - Change Manager에서 변경 요청을 생성합니다.
  - 유지 관리 기간을 실행합니다.
  - State Manager에서 연결을 생성합니다.

  자동화, Change Manager, Maintenance Windows 및 State Manager가 AWS Systems Manager의 모든 기능입니다. 이러한 기능을 Change Calendar와 통합하여 각각에 연결하는 Change Calendar의 현재 상태에 따라 이러한 작업 유형을 허용하거나 차단할 수 있습니다.
- Systems Manager 관리형 노드의 구성을 일관되고 안정적이며 기능적으로 유지할 책임이 있는 관리자.

Change Calendar의 이점

Change Calendar를 사용하면 다음과 같은 이점이 있습니다.

- 적용 전 변경 사항 검토

  Change Calendar 항목을 사용하면 환경에 부정적인 영향을 줄 수 있는 변경 사항을 적용하기 전에 검토할 수 있습니다.

- 적절한 시간 동안에만 변경 사항 적용

  Change Calendar 항목을 사용하면 이벤트 시간 동안 환경을 안정적으로 유지할 수 있습니다. 예를 들어, 컨퍼런스 또는 공개 마케팅 프로모션 기간과 같이 리소스에 대한 수요가 높은 경우 변경을 차단하는 Change Calendar 항목을 생성할 수 있습니다. 일정 항목은 휴가 또는 휴일과 같이 제한된 관리자 지원이 필요한 경우에도 변경을 차단할 수 있습니다. 일정 항목을 사용하여 작업 실패 또는 배포 문제를 해결하는 관리자 지원이 제한된 특정 요일 또는 주를 제외하고 변경을 허용할 수 있습니다.

- 일정의 현재 또는 향후 상태 가져오기

  Systems Manager `GetCalendarState` API 작업을 실행하여 일정의 현재 상태, 지정된 시간의 상태 또는 일정 상태가 변경되도록 예약된 다음 시간을 표시할 수 있습니다.

- EventBridge 지원

  이 Systems Manager 기능은 Amazon EventBridge 규칙에서 이벤트 유형으로 지원됩니다. 자세한 내용은 Amazon EventBridge로 Systems Manager 이벤트 모니터링 및 참조: Systems Manager용 Amazon EventBridge 이벤트 패턴 및 유형 섹션을 참조하세요.

주제

Change Calendar 설정

AWS Systems Manager의 기능인 Change Calendar를 사용하여 다음을 완료합니다.

최신 명령줄 도구 설치

최신 명령줄 도구를 설치하여 일정에 대한 상태 정보를 가져옵니다.

요구 사항	설명
AWS CLI	(옵션) AWS Command Line Interface(AWS CLI)를 사용하여 일정에 대한 상태 정보를 가져오려면 로컬 컴퓨터에 AWS CLI의 최신 릴리스를 설치합니다. CLI를 설치하거나 업그레이드하는 방법에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서의 AWS CLI 설치, 업데이트 및 제거를 참조하세요.
AWS Tools for PowerShell	(옵션) Tools for PowerShell을 사용하여 일정에 대한 상태 정보를 가져오려면 로컬 컴퓨터에 Tools for PowerShell의 최신 릴리스를 설치합니다. Tools for PowerShell을 설치하거나 업그레이드하는 방법에 대한 자세한 내용은 AWS Tools for PowerShell 사용 설명서의 AWS Tools for PowerShell 설치를 참조하세요.

권한 설정

사용자, 그룹 또는 역할에 관리자 권한이 할당되어 있는 경우 Change Calendar에 대한 전체 액세스 권한이 있습니다. 관리자 권한이 없는 경우에는 관리자가 사용자, 그룹 또는 역할에 `AmazonSSMFullAccess` 관리형 정책을 할당하거나 필요한 권한을 제공하는 정책을 할당하여 사용자에게 권한을 부여해야 합니다.

Change Calendar로 작업하려면 다음과 같은 권한이 필요합니다.

Change Calendar 항목

항목에서 이벤트 추가하고 제거하는 것을 포함하여 Change Calendar 항목을 생성, 업데이트 또는 삭제하려면 사용자, 그룹 또는 역할에 연결된 정책에서 다음과 같은 작업이 허용되어야 합니다.

- `ssm:CreateDocument`
- `ssm:DeleteDocument`

- ssm:DescribeDocument
- ssm:DescribeDocumentPermission
- ssm:GetCalendar
- ssm:ListDocuments
- ssm:ModifyDocumentPermission
- ssm:PutCalendar
- ssm:UpdateDocument
- ssm:UpdateDocumentDefaultVersion

일정 상태

일정의 현재 또는 향후 상태에 대한 정보를 가져오려면 사용자, 그룹 또는 역할에 연결된 정책에서 다음과 같은 작업이 허용되어야 합니다.

- ssm:GetCalendarState

운영 이벤트

유지 관리 기간, 연결 및 계획된 자동화와 같은 운영 이벤트를 보려면 사용자, 그룹 또는 역할에 연결된 정책에서 다음과 같은 작업이 허용되어야 합니다.

- ssm:DescribeMaintenanceWindows
- ssm:DescribeMaintenanceWindowExecution
- ssm:DescribeAutomationExecutions
- ssm:ListAssociations

> **Note**
>
> 내 계정이 아닌 다른 계정이 소유한(생성한) Change Calendar 항목은 내 계정과 공유된 경우에도 읽기 전용입니다. 유지 관리 기간, State Manager 연결 및 자동화는 공유되지 않습니다.

Change Calendar 작업

AWS Systems Manager 콘솔을 사용하여 AWS Systems Manager의 기능인 Change Calendar에서 항목을 추가, 관리 또는 삭제할 수 있습니다. 또한 소스 일정으로부터 내보낸 iCalendar(.ics) 파일을 가져와서 지원되는 서드 파티 일정 공급자로부터 이벤트를 가져올 수도 있습니다. 그리고

`GetCalendarState` API 연산 또는 `get-calendar-state` AWS Command Line Interface(AWS CLI) 명령을 사용하여 특정 시간의 Change Calendar 상태에 대한 정보를 가져올 수 있습니다.

주제

Change Calendar 생성

AWS Systems Manager의 기능인 Change Calendar에서 항목을 만들면 `text` 형식을 사용하는 Systems Manager 문서(SSM 문서)가 생성됩니다.

Change Calendar를 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Change Calendar를 선택합니다.
3. Create calendar(일정 생성)를 선택합니다.

   -또는-

   Change Calendar 홈 페이지가 먼저 열리면 변경 일정 생성(Create change calendar)을 선택합니다.
4. Create calendar(일정 생성) 페이지의 Calendar details(일정 세부 정보)에 일정 항목의 이름을 입력합니다. 일정 항목 이름은 문자, 숫자, 마침표, 대시 및 밑줄을 포함할 수 있습니다. 이름은 일정 항목의 용도를 한눈에 파악할 수 있도록 구체적이어야 합니다. 예를 들면, **`support-off-hours`**입니다. 일정 항목을 생성한 후에는 이 이름을 업데이트할 수 없습니다.
5. (선택 사항) 설명(Description)에 일정 항목에 대한 설명을 입력합니다.
6. (선택 사항) 일정 가져오기(Import calendar) 영역에서 파일 선택(Choose file)을 선택하여 서드 파티 일정 공급자로부터 내보낸 iCalendar(`.ics`) 파일을 선택합니다. 파일을 가져오면 일정에 해당 이벤트가 추가됩니다.

지원되는 공급자에는 Google 캘린더, Microsoft Outlook, iCloud 캘린더가 있습니다.

자세한 내용은 서드 파티 일정 공급자로부터 이벤트 가져오기 단원을 참조하십시오.

7. Calendar type(일정 유형)에서 다음 중 하나를 선택합니다.
   - [기본적으로 열림(Open by default)] - 일정이 열리고(이벤트가 시작될 때까지 Automation 작업이 실행될 수 있음) 연결된 이벤트 기간 동안에는 닫혀 있습니다.
   - [기본적으로 닫힘(Closed by default)] - 일정이 닫히고(이벤트가 시작될 때까지 Automation 작업이 실행될 수 없음) 연결된 이벤트 기간 동안에는 열려 있습니다.
8. (선택 사항) 변경 관리 이벤트에서 일정에 변경 관리 이벤트 추가를 선택합니다. 이렇게 선택하면 예약된 모든 유지 관리 기간, State Manager 연결, Automation 워크플로 및 Change Manager 변경 요청이 월별 일정 표시에 표시됩니다.

   > **Tip**
   >
   > 나중에 일정 표시에서 이러한 이벤트 유형을 영구적으로 제거하려면 일정을 편집하고 이 확인란을 선택 취소한 다음에 저장을 선택합니다.

9. Create calendar(일정 생성)를 선택합니다.

   일정 항목이 생성되면 Systems Manager는 Change Calendar 목록에 일정 항목을 표시합니다. 열에는 일정 버전과 일정 소유자의 AWS 계정 번호가 표시됩니다. 일정 항목은 이벤트를 하나 이상 생성하거나 가져올 때까지 작업을 금지하거나 허용할 수 없습니다. 이벤트 생성에 대한 자세한 내용은 Change Calendar 이벤트 생성 섹션을 참조하세요. 이벤트 가져오기에 대한 자세한 내용은 서드 파티 일정 공급자로부터 이벤트 가져오기 섹션을 참조하세요.

Change Calendar에서의 이벤트 생성 및 관리

AWS Systems Manager Change Calendar에서 일정을 생성한 후 열려 있거나 닫혀 있는 일정에 포함된 이벤트를 생성, 업데이트 및 삭제할 수 있습니다. Change Calendar는 AWS Systems Manager의 기능입니다.

> **Tip**
>
> Systems Manager 콘솔에서 직접 이벤트를 생성하는 대신 지원되는 서드 파티 일정 앱으로부터 iCalendar(`.ics`) 파일을 가져올 수 있습니다. 자세한 내용은 서드 파티 일정으로부터 이벤트 가져오기 및 관리 섹션을 참조하세요.

주제

- Change Calendar 이벤트 생성
- Change Calendar 이벤트 업데이트
- Change Calendar 이벤트 삭제

Change Calendar 이벤트 생성

AWS Systems Manager의 기능인 Change Calendar의 항목에 이벤트를 추가할 때 일정 항목의 기본 작업이 일시 중지되는 기간을 지정합니다. 예를 들어, 일정 항목 유형이 기본적으로 닫힘인 경우 일정은 이벤트 기간에 변경되도록 열려 있습니다. (또는 일정에서만 정보 제공 역할을 하는 공지 이벤트를 생성할 수 있습니다.)

현재는 콘솔을 사용해야만 Change Calendar 이벤트를 생성할 수 있습니다. Change Calendar 항목 생성 시 생성된 Change Calendar 문서에 이벤트가 추가됩니다.

Change Calendar 이벤트를 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Change Calendar를 선택합니다.
3. 일정 목록에서 이벤트를 추가할 일정 항목의 이름을 선택합니다.
4. 일정 항목의 세부 정보 페이지에서 Create event(이벤트 생성)를 선택합니다.
5. Create scheduled event(예약된 이벤트 생성) 페이지의 Event details(이벤트 세부 정보)에서 이벤트의 표시 이름을 입력합니다. 이벤트 이름은 문자, 숫자, 마침표, 대시 및 밑줄을 포함할 수 있습니다. 이름은 이벤트의 용도를 파악할 수 있도록 구체적이어야 합니다. 예를 들면, **`nighttime-hours`**입니다.
6. 설명(Description)에 이벤트에 대한 설명을 입력합니다. 예를 들면 **`The support team isn't available during these hours`**입니다.

7. (선택 사항) 이 이벤트가 시각적 알림 또는 미리 알림으로만 제공되도록 하려면 공지(Advisory) 확인란을 선택합니다. 공지 이벤트는 일정에서 기능적인 역할을 하지 않습니다. 일정을 보는 사용자에게만 정보를 제공합니다.
8. 이벤트 시작 날짜(Event start date)에서 이벤트를 시작할 날짜를 MM/DD/YYYY 형식으로 입력하거나 선택하고 지정된 날짜에 이벤트를 시작할 시간을 hh:mm:ss(시, 분, 초) 형식으로 입력합니다.
9. 이벤트 종료 날짜(Event end date)에서 이벤트를 종료할 날짜를 MM/DD/YYYY 형식으로 입력하거나 선택하고 지정된 날짜에 이벤트를 종료할 시간을 hh:mm:ss(시, 분, 초) 형식으로 입력합니다.
10. 예약 시간대(Schedule time zone)에서 이벤트의 시작 및 종료 시간에 적용할 시간대를 선택합니다. 도시 이름의 일부 또는 그리니치 표준시(GMT)와의 시간대 차이를 입력하여 시간대를 더 빨리 찾을 수 있습니다. 기본값은 협정 세계 표준시(UTC)입니다.
11. (선택 사항) 매일, 매주 또는 매월 반복되는 이벤트를 생성하려면 반복(Recurrence)을 설정한 다음 반복 빈도 및 선택적 종료 날짜를 지정합니다.
12. Create scheduled event(예약된 이벤트 생성)를 선택합니다. 새 이벤트가 일정 항목에 추가되고 일정 항목의 세부 정보 페이지에 있는 이벤트 탭에 표시됩니다.

Change Calendar 이벤트 업데이트

AWS Systems Manager 콘솔에서 Change Calendar 이벤트를 업데이트하려면 다음 절차를 따릅니다. Change Calendar는 AWS Systems Manager의 기능입니다.

Change Calendar 이벤트를 업데이트하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Change Calendar를 선택합니다.
3. 일정 목록에서 이벤트를 편집할 일정 항목의 이름을 선택합니다.
4. 일정 항목의 세부 정보 페이지에서 이벤트를 선택합니다.
5. 일정 페이지에서 편집할 이벤트를 선택합니다.

> **Tip**
>
> 왼쪽 위에 있는 버튼을 사용하여 1년 뒤로 또는 앞으로 이동하거나 한 달 뒤로 또는 앞으로 이동합니다. 필요한 경우 오른쪽 상단의 목록에서 올바른 시간대를 선택하여 시간대를 변경합니다.

6. 이벤트 세부 정보(Event details)에서 편집(Edit)을 선택합니다.

이벤트 이름과 설명을 변경하려면 현재 텍스트 값을 추가하거나 바꿉니다.

7. 이벤트 시작 날짜(Event start date) 값을 변경하려면 현재 시작 날짜를 선택한 다음 일정에서 새 날짜를 선택합니다. 시작 시간을 변경하려면 현재 시작 시간을 선택한 다음 목록에서 새 시간을 선택합니다.
8. 이벤트 종료 날짜(Event end date) 값을 변경하려면 현재 날짜를 선택한 다음 일정에서 새 종료 날짜를 선택합니다. 종료 시간을 변경하려면 현재 종료 시간을 선택한 다음 목록에서 새 시간을 선택합니다.
9. 예약 시간대(Schedule time zone) 값을 변경하려면 이벤트의 시작 및 종료 시간에 적용할 시간대를 선택합니다. 도시 이름의 일부 또는 그리니치 표준시(GMT)와의 시간대 차이를 입력하여 시간대를 더 빨리 찾을 수 있습니다. 기본값은 협정 세계 표준시(UTC)입니다.
10. (선택 사항) 이 이벤트가 시각적 알림 또는 미리 알림으로만 제공되도록 하려면 공지(Advisory) 확인란을 선택합니다. 공지 이벤트는 일정에서 기능적인 역할을 하지 않습니다. 일정을 보는 사용자에게만 정보를 제공합니다.
11. Save(저장)를 선택합니다. 변경 사항은 일정 항목의 세부 정보 페이지에 있는 이벤트 탭에 표시됩니다. 변경 사항을 보려면 업데이트한 이벤트를 선택합니다.

Change Calendar 이벤트 삭제

AWS Management Console을 사용하여 AWS Systems Manager의 기능인 Change Calendar에서 이벤트를 한 번에 하나씩 삭제할 수 있습니다.

> **Tip**
>
> 일정을 생성할 때 일정에 변경 관리 이벤트 추가를 선택한 경우 다음과 같이 수행할 수 있습니다.
>
> - 일정 표시에서 변경 관리 이벤트 유형을 일시적으로 숨기려면 월별 미리 보기 상단에 있는 유형에 X를 선택합니다.
> - 일정 표시에서 이러한 유형을 영구적으로 제거하려면 일정을 편집하고 일정에 변경 관리 이벤트 추가 확인란을 선택 취소한 다음에 저장을 선택합니다. 유형은 일정 표시에서 제거해도 계정에서는 삭제되지 않습니다.

Change Calendar 이벤트를 삭제하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.

2. 탐색 창에서 Change Calendar를 선택합니다.
3. 일정 목록에서 이벤트를 삭제할 일정 항목의 이름을 선택합니다.
4. 일정 항목의 세부 정보 페이지에서 이벤트를 선택합니다.
5. 일정 페이지에서 삭제할 이벤트를 선택합니다.

> **Tip**
>
> 왼쪽 위에 있는 버튼을 사용하여 일정을 1년 뒤로 또는 앞으로 이동하거나 한 달 뒤로 또는 앞으로 이동합니다. 필요한 경우 오른쪽 상단의 목록에서 올바른 시간대를 선택하여 시간대를 변경합니다.

6. Event details(이벤트 세부 정보) 페이지에서 삭제를 선택합니다. 이벤트를 삭제할지 확인하는 메시지가 표시되면 확인(Confirm)을 선택합니다.

서드 파티 일정으로부터 이벤트 가져오기 및 관리

AWS Systems Manager 콘솔에서 직접 이벤트를 생성하는 대신 지원되는 서드 파티 일정 앱으로부터 iCalendar(`.ics`) 파일을 가져올 수 있습니다. 일정에는 가져온 이벤트와 사용자가 AWS Systems Manager의 기능인 Change Calendar에서 생성한 이벤트가 모두 포함될 수 있습니다.

시작하기 전 준비 사항

일정 파일을 가져오려고 시도하기 전에 다음 요구 사항 및 제약 조건을 검토하세요.

일정 파일 형식

유효한 iCalendar 파일(`.ics`)만 지원합니다.

지원되는 일정 공급자

다음 서드 파티 일정 공급자에서 내보낸 `.ics` 파일만 지원됩니다.

- Google 캘린더(내보내기 지침)
- Microsoft Outlook(내보내기 지침)
- iCloud 캘린더(내보내기 지침)

파일 크기

유효한 `.ics` 파일 수만큼 가져올 수 있습니다. 그러나 각 일정에 대해 가져온 모든 파일의 총 크기는 64KB를 초과할 수 없습니다.

> **Tip**
>
> .ics 파일의 크기를 최소화하기 위해, 일정 항목에 대한 기본 세부 정보만 내보내고 있는지 확인합니다. 필요한 경우 내보내는 시간 간격의 길이를 줄입니다.

시간대

내보낸 .ics 파일에는 일정 이름, 일정 공급자 및 하나 이상의 이벤트 외에 일정의 시간대도 표시되어야 합니다. 그렇지 않거나 표준 시간대를 식별하는 데 문제가 있는 경우 해당 파일을 가져온 후 표준 시간대를 지정하라는 메시지가 표시됩니다.

반복 이벤트 제한

사용자가 내보낸 .ics 파일에는 반복 이벤트가 포함될 수 있습니다. 그러나 소스 일정에서 반복 이벤트가 하나 이상 삭제된 경우 가져오기 작업이 실패합니다.

주제

- 서드 파티 일정 공급자로부터 이벤트 가져오기
- 서드 파티 일정 공급자의 모든 이벤트 업데이트
- 서드 파티 일정에서 가져온 모든 이벤트 삭제

서드 파티 일정 공급자로부터 이벤트 가져오기

지원되는 서드 파티 일정 앱의 iCalendar(.ics) 파일을 가져오려면 다음 절차를 따릅니다. 파일에 포함된 이벤트는 열려 있거나 닫혀 있는 일정의 규칙에 통합됩니다. Change Calendar(AWS Systems Manager의 기능)를 사용하여 생성하고 있는 새 일정으로 파일을 가져오거나 기존 일정으로 가져올 수 있습니다.

.ics 파일을 가져온 후 Change Calendar 인터페이스를 사용하여 개별 이벤트를 삭제할 수 있습니다. 자세한 설명은 Change Calendar 이벤트 삭제을 참조하세요. 또한 .ics 파일을 삭제하여 소스 일정의 모든 이벤트를 삭제할 수도 있습니다. 자세한 설명은 서드 파티 일정에서 가져온 모든 이벤트 삭제을 참조하세요.

서드 파티 일정 공급자로부터 이벤트를 가져오려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Change Calendar를 선택합니다.

3. 새 일정으로 시작하려면 일정 생성(Create calendar)을 선택합니다. 일정 가져오기(Import calendar) 영역에서 파일 선택(Choose file)을 선택합니다. 새 일정을 생성하는 다른 단계에 대한 자세한 내용은 Change Calendar 생성 섹션을 참조하세요.

   -또는-

   서드 파티 이벤트를 기존 일정으로 가져오려면 기존 일정의 이름을 선택하여 엽니다.
4. 작업, 편집(Actions, Edit)을 선택한 다음 일정 가져오기(Import calendar) 영역에서 파일 선택(Choose file)을 선택합니다.
5. 로컬 컴퓨터에서 내보낸 `.ics` 파일로 이동하고 선택합니다.
6. 메시지가 나타나면 표준 시간대 선택(Select a time zone)에서 일정에 적용할 시간대를 선택합니다.
7. Save(저장)를 선택합니다.

서드 파티 일정 공급자의 모든 이벤트 업데이트

iCalendar `.ics` 파일을 가져온 후 소스 일정에 여러 이벤트가 추가되거나 제거된 경우 이러한 변경 사항을 Change Calendar에 반영할 수 있습니다. 먼저 소스 일정을 다시 내보낸 다음 새 파일을 Change Calendar로 가져옵니다. 이는 AWS Systems Manager의 기능입니다. Change Calendar의 이벤트는 최신 파일의 내용을 반영하도록 업데이트됩니다.

서드 파티 일정 공급자의 모든 이벤트를 업데이트하려면

1. 서드 파티 일정에서 Change Calendar에 반영할 이벤트를 추가하거나 제거합니다. 그런 다음 해당 일정을 새 `.ics` 파일로 다시 내보냅니다.
2. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
3. 탐색 창에서 Change Calendar를 선택합니다.
4. 일정 목록에서 일정 이름을 선택합니다.
5. 파일 선택을 선택하고, 대체 `.ics` 파일로 이동하고 선택합니다.
6. 기존 파일 덮어쓰기에 대한 알림이 뜨면 확인(Confirm)을 선택합니다.

서드 파티 일정에서 가져온 모든 이벤트 삭제

서드 파티 공급자로부터 가져온 이벤트를 더 이상 일정에 포함시키지 않으려면, 가져온 iCalendar `.ics` 파일을 삭제할 수 있습니다.

서드 파티 일정에서 가져온 모든 이벤트를 삭제하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Change Calendar를 선택합니다.
3. 일정 목록에서 일정 이름을 선택합니다.
4. 일정 가져오기(Import calendar) 영역의 가져온 일정(My imported calendars)에서, 가져온 일정의 이름을 찾은 다음 X를 선택합니다.
5. Save(저장)를 선택합니다.

Change Calendar 업데이트

Change Calendar에 대한 설명은 업데이트할 수 있지만 이름은 업데이트할 수 없습니다. 일정의 기본 상태를 변경할 수 있지만 이 경우 일정과 연결된 이벤트 기간에 변경 작업의 동작이 반대로 수행됩니다. 예를 들어 일정의 상태를 [기본적으로 열림(Open by default)]에서 [기본적으로 닫힘(Closed by default)]으로 변경하면 연결된 이벤트를 생성한 사용자가 변경을 기대하지 않는 이벤트 기간에 원치 않은 변경이 발생할 수 있습니다.

Change Calendar를 업데이트할 때, 해당 항목을 생성할 때 생성한 Change Calendar 문서를 편집합니다. Change Calendar는 AWS Systems Manager의 기능입니다.

Change Calendar를 업데이트하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Change Calendar를 선택합니다.
3. 일정 목록에서 업데이트할 일정의 이름을 선택합니다.
4. 일정의 세부 정보 페이지에서 작업, 편집(Actions, Edit)을 선택합니다.
5. 설명에서 설명 텍스트를 변경할 수 있습니다. Change Calendar의 이름은 편집할 수 없습니다.
6. 일정 상태를 변경하려면 Calendar type(일정 유형)에서 다른 값을 선택합니다. 이 경우 일정과 연결된 이벤트 기간에 변경 작업의 동작이 반대로 수행됩니다. 일정 유형을 변경하기 전에 일정 유형을 변경해도 사용자가 생성한 이벤트 기간에 원치 않는 변경이 허용되지 않음을 다른 Change Calendar 사용자와 함께 확인해야 합니다.

   - 기본적으로 열림(Open by default) - 일정이 열리고(이벤트가 시작될 때까지 Automation 작업이 실행될 수 있음) 연결된 이벤트 기간 동안에는 닫혀 있습니다.
   - 기본적으로 닫힘(Closed by default) - 일정이 닫히고(이벤트가 시작될 때까지 Automation 작업이 실행될 수 없음) 연결된 이벤트 기간 동안에는 열려 있습니다.

7. Save(저장)를 선택합니다.

   일정은 이벤트를 하나 이상 추가할 때까지 작업을 금지하거나 허용할 수 없습니다. 이벤트를 추가하는 방법에 대한 자세한 내용은 Change Calendar 이벤트 생성 섹션을 참조하세요.

Change Calendar 공유

AWS Systems Manager 콘솔을 사용하여 AWS Systems Manager의 기능인 Change Calendar에서 Change Calendar를 다른 AWS 계정와 공유할 수 있습니다. 일정을 공유하면 공유 계정의 사용자가 일정을 읽기 전용으로 사용할 수 있습니다. 유지 관리 기간, State Manager 연결 및 자동화는 공유되지 않습니다.

Change Calendar를 공유하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Change Calendar를 선택합니다.
3. 일정 목록에서 공유할 일정의 이름을 선택합니다.
4. 일정의 세부 정보 페이지에서 공유(Sharing) 탭을 선택합니다.
5. 작업, 공유(Actions, Share)를 선택합니다.
6. [일정 공유(Share calendar)]에서 [계정 ID(Account ID)]에 유효한 AWS 계정의 ID 번호를 입력한 다음 [공유(Share)]를 선택합니다.

   공유 계정의 사용자는 Change Calendar를 읽을 수 있지만 변경할 수는 없습니다.

Change Calendar 삭제

Systems Manager 콘솔 또는 AWS Command Line Interface(AWS CLI)를 사용하여 AWS Systems Manager의 기능인 Change Calendar에서 일정을 삭제할 수 있습니다. Change Calendar를 삭제하면 연결된 모든 이벤트가 삭제됩니다.

Change Calendar를 삭제하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Change Calendar를 선택합니다.
3. 일정 목록에서 삭제할 일정의 이름을 선택합니다.
4. 일정의 세부 정보 페이지에서 작업, 삭제(Actions, Delete)를 선택합니다. 일정을 삭제할지 확인하는 메시지가 표시되면 삭제(Delete)를 선택합니다.

Change Calendar 상태 가져오기

AWS Systems Manager의 기능인 Change Calendar에서 일정의 전체 상태 또는 특정 시간의 일정 상태를 가져올 수 있습니다. 다음에 일정 상태가 OPEN에서 CLOSED로 또는 그 반대로 변경되는 시간을 표시할 수도 있습니다.

이 태스크는 `GetCalendarState` API 작업을 사용해야만 수행할 수 있습니다. 이 섹션의 절차에서는 AWS Command Line Interface(AWS CLI)를 사용합니다.

Change Calendar 상태를 가져오려면

- 다음 명령을 실행하여 특정 시간에 하나 이상의 일정에 대한 상태를 표시합니다. `--calendar-names` 파라미터는 필수 사항이지만 `--at-time`은 선택 사항입니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

  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"
  ```

  다음은 예입니다.

  ```
  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"
  ```

  다음은 예입니다.

  ```
  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"
```

명령은 다음과 같은 정보를 반환합니다.

```
{
    "State": "OPEN",
    "AtTime": "2020-07-30T16:18:18Z",
    "NextTransitionTime": "2020-07-31T00:00:00Z"
}
```

결과는 해당 계정이 소유하거나 공유한 지정된 일정 항목의 일정 상태(일정이 DEFAULT_OPEN인지 아니면 DEFAULT_CLOSED인지 여부)를 --at-time 값으로 지정된 시간과 다음 전환 시간으로 표시합니다. --at-time 파라미터를 추가하지 않으면 현재 시간이 사용됩니다.

> **Note**
>
> 요청에 일정을 2개 이상 지정하는 경우 요청의 모든 달력이 열려 있는 경우에만 OPEN 상태가 반환됩니다. 요청의 일정이 하나 이상 닫혀 있는 경우 CLOSED 상태가 반환됩니다.

Automation 실행서에 Change Calendar 종속성 추가

Automation 작업이 AWS Systems Manager의 기능인 Change Calendar를 준수하도록 하려면 해당 aws:assertAwsResourceProperty 작업을 사용하는 Automation 실행서에 단계를 추가합니다. GetCalendarState를 실행하여 지정된 일정 항목이 원하는 상태(OPEN 또는 CLOSED)인지 확인하도록 작업을 구성합니다. Automation 실행서는 일정 상태가 OPEN인 경우에만 다음 단계를 계속 진행할 수 있습니다. 다음은 일정 상태가 DesiredValues에 지정된 OPEN 상태가 아닌 경우 다음 단계인 LaunchInstance로 진행할 수 없는 Automation 실행서의 YAML 기반 샘플 발췌 내용입니다.

다음은 예입니다.

```
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 문제 해결

다음 정보를 사용하면 AWS Systems Manager의 기능인 Change Calendar 관련 문제를 해결하는 데 도움이 됩니다.

주제

- '일정 가져오기 실패(Calendar import failed)' 오류

'일정 가져오기 실패(Calendar import failed)' 오류

문제: iCalendar(`.ics`) 파일을 가져올 때 시스템이 일정 가져오기에 실패했음을 보고합니다.

- 솔루션 1 - 지원되는 서드 파티 일정 공급자에서 내보낸 파일을 가져오고 있는지 확인합니다. 여기에는 다음이 포함됩니다.
  - Google 캘린더(내보내기 지침)
  - Microsoft Outlook(내보내기 지침)
  - iCloud 캘린더(내보내기 지침)
- 솔루션 2 - 소스 일정에 반복 이벤트가 포함된 경우 이벤트의 개별 발생이 취소되거나 삭제되지 않았는지 확인합니다. 현재 Change Calendar는 개별 취소가 있는 반복 이벤트 가져오기를 지원하지 않습니다. 이 문제를 해결하려면 소스 일정에서 반복 이벤트를 제거하고, 해당 일정을 다시 내보내고 Change Calendar로 다시 가져온 다음, Change Calendar 인터페이스를 사용하여 반복 이벤트를 추가합니다. 자세한 내용은 Change Calendar 이벤트 생성 섹션을 참조하세요.

- 솔루션 3 - 소스 일정에 하나 이상의 이벤트가 포함되어 있는지 확인합니다. 이벤트가 포함되지 않은 `.ics` 파일은 업로드에 실패합니다.
- 솔루션 4 - `.ics` 파일이 너무 커서 시스템이 가져오는 데 실패했다고 보고하는 경우, 일정 항목에 대한 기본 세부 정보만 내보내고 있는지 확인하세요. 필요한 경우 내보내는 시간 간격의 길이를 줄입니다.
- 솔루션 5 - 내보낸 일정을 이벤트(Events) 탭에서 가져오려고 할 때 Change Calendar에서 해당 일정의 표준 시간대를 확인할 수 없는 경우, 다음과 같은 메시지가 나타날 수 있습니다. “일정 가져오기에 실패했습니다. Change Calendar가 유효한 시간대를 찾을 수 없습니다. 편집 메뉴에서 일정을 가져올 수 있습니다.” 이 경우, 작업, 편집(Actions, Edit)을 선택한 다음 일정 편집(Edit calendar) 페이지에서 파일을 가져옵니다.
- 솔루션 6 - `.ics` 파일을 가져오기 전에 편집하지 마세요. 파일의 내용을 수정하려고 하면 일정 데이터가 손상될 수 있습니다. 가져오기를 시도하기 전에 파일을 수정한 경우, 소스 일정에서 다시 일정을 내보낸 다음 업로드를 다시 시도합니다.

AWS Systems Manager Maintenance Windows

AWS Systems Manager의 기능인 Maintenance Windows를 사용하면 운영 체제 패치, 드라이버 업데이트, 소프트웨어 또는 패치 설치와 같이 노드에서 중단 가능성이 있는 작업 수행 시기에 대한 일정을 정의할 수 있습니다.

Maintenance Windows를 사용하면 Amazon Simple Storage Service(Amazon S3) 버킷, Amazon Simple Queue Service(Amazon SQS) 대기열, AWS Key Management Service(AWS KMS) 키 등의 수많은 다른 AWS 리소스 유형에 대한 작업을 예약할 수 있습니다.

유지 관리 기간 대상에 포함할 수 있는 지원되는 리소스 유형의 전체 목록은 AWS Resource Groups 사용 설명서의 AWS Resource Groups 및 Tag Editor와 사용할 수 있는 리소스를 참조하세요. Maintenance Windows를 시작하려면 Systems Manager 콘솔을 엽니다. 탐색 창에서 Maintenance Windows를 선택합니다.

> **Note**
>
> State Manager 및 Maintenance Windows가 관리형 노드에서 몇 가지 유사한 유형의 업데이트를 수행할 수 있습니다. 어느 유형의 업데이트를 선택할지는 시스템 규정 준수를 자동화해야 하는지 아니면 지정한 기간 동안 우선순위가 높고 시간에 민감한 태스크를 수행해야 하는지 여부에 따라 다릅니다.
> 자세한 내용은 State Manager와 Maintenance Windows 중에서 선택 단원을 참조하십시오.

각 유지 관리 기간에는 일정, 최대 기간, 일련의 등록된 대상(조치를 취한 관리형 노드 또는 기타 AWS 리소스) 및 일련의 등록된 태스크가 있습니다. 유지 관리 기간을 생성 또는 업데이트할 때 태그를 추가할 수 있습니다. (태그는 조직 내에서 리소스를 식별하고 분류하는 데 도움이 되는 키입니다.) 또한 이전 또는 이후에 유지 관리 기간이 실행되면 안 되는 날짜를 지정하고, 유지 관리 기간 일정의 기준이 될 국제 시간대를 지정할 수 있습니다.

유지 관리 기간에 대한 다양한 스케줄 관련 옵션이 서로 관련되는 방법에 대한 자세한 내용은 유지 관리 기간 예약 및 유효 기간 옵션 섹션을 참조하세요.

`--schedule` 옵션 작업에 대한 자세한 내용은 참조: Systems Manager용 Cron 및 Rate 표현식 섹션을 참조하세요.

지원되는 태스크 유형

유지 관리 기간에서는 다음 네 가지 유형의 태스크를 실행할 수 있습니다.

- Systems Manager의 기능인 Run Command의 명령

  Run Command에 대한 자세한 내용은 AWS Systems Manager Run Command 섹션을 참조하세요.
- Systems Manager의 기능인 Automation의 워크플로

  자동화 워크플로에 대한 자세한 내용은 AWS Systems Manager 자동화 섹션을 참조하세요.
- AWS Lambda의 함수

  Lambda 함수 생성에 대한 자세한 내용은 AWS Lambda개발자 안내서의 Lambda 시작하기를 참조하세요.
- AWS Step Functions의 태스크

> **Note**
>
> 유지 관리 기간 태스크는 Step Functions 표준 상태 머신 워크플로만 지원합니다. 익스프레스 상태 머신 워크플로는 지원하지 않습니다. 상태 머신 워크플로 유형에 대한 자세한 내용은 AWS Step Functions개발자 안내서의 표준 워크플로와 익스프레스 워크플로를 참조하세요.

  Step Functions에 대한 자세한 내용은 AWS Step Functions Developer Guide를 참조하세요.

> **Note**
>
> 유지 관리 기간 Run Command 유형 태스크에 대해 하나 이상의 대상을 지정해야 합니다. 태스크에 따라 대상은 다른 유지 관리 기간 태스크 유형(Automation, AWS Lambda 및 AWS Step Functions)에 대해 옵션입니다. 대상을 지정하지 않는 태스크 실행에 대한 자세한 내용은 대상 없이 유지 관리 기간 태스크 등록 섹션을 참조하세요.

즉, 유지 관리 기간을 사용하여 선택한 대상에서 다음과 같은 태스크를 수행할 수 있습니다.

- 애플리케이션을 설치하거나 업데이트합니다.
- 패치를 적용합니다.
- SSM Agent를 설치하거나 업데이트합니다.
- Systems Manager Run Command 태스크를 사용하여 PowerShell 명령과 Linux 셸 스크립트를 실행합니다.
- Amazon Machine Images(AMIs)를 구축하고, 소프트웨어를 부트스트랩하고, Systems Manager Automation 태스크를 사용하여 노드를 구성합니다.
- 패치 업데이트를 위한 노드 스캔 등의 추가 작업을 호출하는 AWS Lambda 함수를 실행합니다.
- AWS Step Functions 상태 머신을 실행하여 태스크를 수행합니다. 예를 들어 Elastic Load Balancing 환경에서 노드를 제거하고 노드를 패치한 후 Elastic Load Balancing 환경에 노드를 다시 추가할 수 있습니다.
- AWS 리소스 그룹을 대상으로 지정하여 오프라인 상태의 노드를 타겟팅합니다.

EventBridge 지원

이 Systems Manager 기능은 Amazon EventBridge 규칙에서 이벤트 유형으로 지원됩니다. 자세한 내용은 Amazon EventBridge로 Systems Manager 이벤트 모니터링 및 참조: Systems Manager용 Amazon EventBridge 이벤트 패턴 및 유형 섹션을 참조하세요.

내용

- Maintenance Windows 설정
- 유지 관리 기간 작업(콘솔)
- Systems Manager Maintenance Windows 튜토리얼(AWS CLI)
- 유지 관리 기간 연습

- 유지 관리 기간 작업 등록 시 의사 파라미터 사용
- 유지 관리 기간 예약 및 유효 기간 옵션
- 대상 없이 유지 관리 기간 태스크 등록
- 유지 관리 기간 문제 해결

Maintenance Windows 설정

AWS 계정의 사용자가 AWS Systems Manager의 기능인 Maintenance Windows를 사용하여 유지 관리 기간 태스크를 생성하고 예약하려면 먼저 필요한 권한이 부여되어야 합니다.

시작하기 전 준비 사항

섹션의 태스크를 완료하려면 이미 설정된 다음 리소스 중 하나 또는 두 가지 모두가 필요합니다.

- IAM 엔터티(사용자, 역할 또는 그룹)에 할당된 권한. 이러한 엔터티에는 유지 관리 기간 작업을 위한 일반 권한이 이미 부여되어 있어야 합니다. 이 작업은 사용자 또는 그룹에 IAM 정책 `AmazonSSMFullAccess`를 할당하거나, 유지 관리 기간 태스크를 다루는 Systems Manager를 위한 더 작은 액세스 권한 세트를 제공하는 다른 IAM 정책을 보장하여 수행할 수 있습니다.
- (선택) Run Command 태스크를 실행하는 유지 관리 기간의 경우 Amazon Simple Notification Service(Amazon SNS) 상태 알림을 보내도록 선택할 수 있습니다. Run Command는 Systems Manager의 기능입니다. 이 옵션을 사용하려면 이러한 설정 작업을 완료하기 전에 Amazon SNS 주제를 구성합니다. SNS 알림 전송에 사용할 IAM 역할 생성에 대한 정보를 포함하여 Systems Manager에 대한 Amazon SNS 알림을 구성하는 방법에 대한 내용은 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 섹션을 참조하세요.

설정 작업 개요

사용자가 유지 관리 기간을 등록하는 데 필요한 권한을 부여하기 위해 관리자는 다음 작업을 수행합니다. (전체 지침은 콘솔을 사용하여 유지 관리 기간에 대한 권한 구성에 제공됨).

작업 1: 사용자 지정 유지 관리 기간 역할에 사용할 정책 생성

유지 관리 기간 작업에는 대상 리소스에서 실행하는 데 필요한 권한을 제공하기 위해 IAM 역할이 필요합니다. 실행하는 작업 유형 및 기타 운영 요구 사항에 따라 이 정책의 내용이 결정됩니다.

작업 1: 사용자 지정 유지 관리 기간 서비스 역할에 대한 정책 생성 주제에서 적용할 수 있는 기본 정책을 제공합니다.

작업 2: 유지 관리 기간 작업을 위한 사용자 지정 서비스 역할 생성

작업 1에서 생성한 정책은 작업 2에서 생성하는 유지 관리 기간 역할에 연결됩니다. 사용자가 유지 관리 기간 작업을 등록할 때 이 사용자 지정 서비스 역할을 작업 구성의 일부로 지정합니다. 이 역할의 권한을 통해 Systems Manager가 사용자 대신 유지 관리 기간에서 작업을 실행할 수 있습니다.

> **Important**
>
> 이전에는 Systems Manager 콘솔에서 작업에 대한 유지 관리 역할로 사용하도록 AWS에서 관리하는 IAM 서비스 연결 역할인 `AWSServiceRoleForAmazonSSM`을 선택할 수 있었습니다. 유지 관리 기간에 이 역할 및 관련 정책인 `AmazonSSMServiceRolePolicy`를 사용하는 것은 더 이상 권장되지 않습니다. 현재 유지 관리 기간 작업에 이 역할을 사용하는 경우 사용을 중지하는 것이 좋습니다. 대신, 유지 관리 기간 작업 실행 시 Systems Manager와 다른 AWS 서비스 간에 통신을 가능하게 하는 IAM 역할을 생성하세요.

작업 3: 유지 관리 기간 작업을 등록하는 사용자에게 서비스 역할을 사용할 수 있는 권한 부여

사용자에게 사용자 지정 유지 관리 기간 역할에 액세스할 수 있는 권한을 제공하면 유지 관리 기간 작업에 이 역할을 사용할 수 있습니다. 이는 Maintenance Windows 기능에 대한 Systems Manager API 명령에 사용할 수 있도록 이미 부여한 권한에 추가됩니다. 이 역할은 유지 관리 기간 작업을 실행하는 데 필요한 권한을 전달합니다. 따라서 사용자는 이러한 IAM 권한을 전달할 수 없는 상태에서 사용자 정의 서비스 역할을 사용하여 유지 관리 기간에 작업을 할당할 수 없습니다.

작업 4: (선택) 유지 관리 기간 태스크를 등록할 수 없는 사용자에 대한 명시적 거부 권한

유지 관리 기간에 작업을 등록하지 않으려는 AWS 계정의 사용자에 대한 `ssm:RegisterTaskWithMaintenanceWindow` 권한을 거부할 수 있습니다. 이렇게 하면 유지 관리 기간 작업을 등록하지 않아야 하는 사용자에게 추가적인 예방 계층이 제공됩니다.

주제

- 콘솔을 사용하여 유지 관리 기간에 대한 권한 구성

콘솔을 사용하여 유지 관리 기간에 대한 권한 구성

다음 절차에서는 AWS Systems Manager 콘솔을 사용하여 유지 관리 기간에 대한 필수 역할 및 권한을 생성하는 방법을 설명합니다.

주제

- 작업 1: 사용자 지정 유지 관리 기간 서비스 역할에 대한 정책 생성
- 작업 2: 유지 관리 기간에 대한 사용자 지정 서비스 역할 생성(콘솔)
- 작업 3: 유지 관리 기간 작업을 등록할 수 있는 사용자에 대한 권한 구성(콘솔)
- 태스크 4: 유지 관리 기간 태스크를 등록할 수 없는 사용자에 대한 권한 구성

작업 1: 사용자 지정 유지 관리 기간 서비스 역할에 대한 정책 생성

JSON 형식의 다음 정책을 사용하여 유지 관리 기간 역할에 사용할 정책을 생성할 수 있습니다. 이후 작업 2: 유지 관리 기간에 대한 사용자 지정 서비스 역할 생성(콘솔)에서 이 정책을 해당 역할에 연결합니다.

> **Important**
>
> 유지 관리 기간이 실행되는 작업 및 작업 유형에 따라 이 정책의 일부 권한이 필요하지 않을 수 있으며, 추가 사용 권한을 포함해야 할 수도 있습니다.

사용자 지정 유지 관리 기간 서비스 역할에 대한 정책 생성

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 정책을 선택한 후 정책 생성을 선택합니다.
3. JSON 탭을 선택합니다.
4. 기본 콘텐츠를 다음으로 바꿉니다.

```
{
    "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": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ssm.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

5. 계정에서 실행하는 유지 관리 작업에 필요한 경우 JSON 콘텐츠를 수정합니다. 변경 사항은 계획된 작업에 따라 다릅니다.

   예:

   - 와일드카드(*) 한정자를 사용하는 대신 특정 함수 및 상태 머신에 Amazon 리소스 이름(ARN)을 제공할 수 있습니다.
   - AWS Step Functions 작업을 실행할 계획이 없는 경우 `states` 권한 및 ARN을 제거할 수 있습니다.
   - AWS Lambda 작업을 실행할 계획이 없는 경우 `lambda` 권한 및 ARN을 제거할 수 있습니다.
   - Automation 작업을 실행할 계획이 없는 경우 `ssm:GetAutomationExecution` 및 `ssm:StartAutomationExecution` 권한을 제거할 수 있습니다.
   - 작업을 실행하는 데 필요할 수 있는 권한을 추가합니다. 예를 들어 일부 자동화 작업은 AWS CloudFormation 스택과 함께 작동합니다. 따라서 `cloudformation:CreateStack`, `cloudformation:DescribeStacks` 및 `cloudformation:DeleteStack` 권한이 필요합니다.

     다른 예를 들어 Automation 실행서 `AWS-CopySnapshot`의 경우 Amazon Elastic Block Store(Amazon EBS) 스냅샷을 생성할 수 있는 권한이 필요합니다. 따라서 서비스 역할에 `ec2:CreateSnapshot` 권한이 필요합니다.

     Automation 실행서에 필요한 역할 권한에 대한 자세한 내용은 AWS Systems Manager Automation 실행서 참조의 실행서 설명을 참조하세요.

6. 정책 개정을 완료한 후 다음: 태그(Next: Tags)를 선택합니다.
7. (선택 사항) 이 정책에 대한 액세스를 구성, 추적 또는 제어할 태그-키 값 페어를 하나 이상 추가한 후 [다음: 검토]를 선택합니다.

8. 이름(Name)에서 생성한 Maintenance Windows 서비스 역할에서 사용하는 정책으로 식별되는 이름을 입력합니다. 예: **my-maintenance-window-role-policy**.
9. 정책 생성(Create policy)을 선택하고, 정책에 대해 지정한 이름을 적어둡니다. 다음 절차인 작업 2: 유지 관리 기간에 대한 사용자 지정 서비스 역할 생성(콘솔)에서 이를 참조합니다.

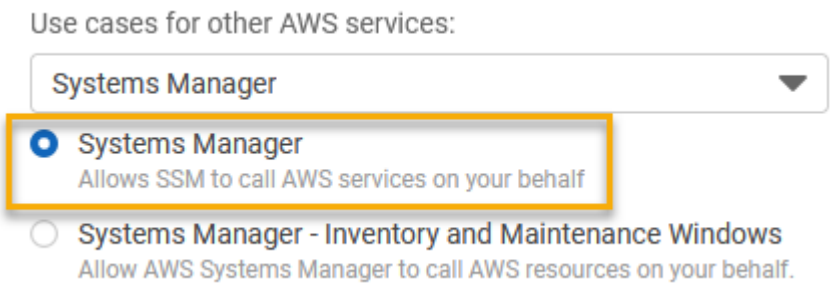
작업 2: 유지 관리 기간에 대한 사용자 지정 서비스 역할 생성(콘솔)

다음 절차를 사용하여 Systems Manager가 사용자 대신 Maintenance Windows 태스크를 실행할 수 있도록 Maintenance Windows에 대한 사용자 지정 서비스 역할을 생성합니다. 이전 태스크에서 생성한 정책을 생성한 사용자 지정 서비스 역할에 연결합니다.

> **Important**
>
> 이전에는 Systems Manager 콘솔에서 작업에 대한 유지 관리 역할로 사용하도록 AWS에서 관리하는 IAM 서비스 연결 역할인 `AWSServiceRoleForAmazonSSM`을 선택할 수 있었습니다. 유지 관리 기간에 이 역할 및 관련 정책인 `AmazonSSMServiceRolePolicy`를 사용하는 것은 더 이상 권장되지 않습니다. 현재 유지 관리 기간 작업에 이 역할을 사용하는 경우 사용을 중지하는 것이 좋습니다. 대신, 유지 관리 기간 작업 실행 시 Systems Manager와 다른 AWS 서비스 간에 통신을 가능하게 하는 IAM 역할을 생성하세요.

사용자 지정 서비스 역할을 생성하려면(콘솔)

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할(Roles)을 선택한 후 역할 생성(Create role)을 선택합니다.
3. 신뢰할 수 있는 엔터티 선택(Select trusted entity)에서 다음을 선택합니다.
    1. 신뢰할 수 있는 엔터티 유형에서 AWS 서비스를 선택합니다.
    2. 기타 AWS 서비스 사용 사례에서 Systems Manager를 선택합니다.
    3. 다음 이미지에 표시된 것과 같이 Systems Manager를 선택합니다.



4. 다음을 선택합니다.
5. 검색 상자에 작업 1: 사용자 지정 유지 관리 기간 서비스 역할에 대한 정책 생성에서 생성한 정책의 이름을 입력하고, 이름 옆에 있는 상자를 선택한 후 다음(Next)을 선택합니다.
6. 역할 이름(Role name)에 이 역할을 Maintenance Windows 역할로 식별하는 이름을 입력합니다. 예: **my-maintenance-window-role**.
7. (선택 사항) 이 역할의 용도를 나타내도록 기본 역할 설명을 변경합니다. 예: **Performs maintenance window tasks on your behalf**.
8. (선택 사항) 이 역할에 대한 액세스를 구성, 추적 또는 제어할 태그-키 값 페어를 하나 이상 추가한 후 다음: 검토(Next: Review)를 선택합니다.
9. 역할 생성(Create role)을 선택합니다. 그러면 역할 페이지로 돌아갑니다.
10. 방금 만든 역할 이름을 선택합니다.
11. 신뢰 관계(Trust relationships) 탭을 클릭한 후 다음 정책이 신뢰 엔터티(Trusted entities) 상자에 표시되는지 확인합니다.

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

12. 요약(Summary) 영역에서 역할 이름과 ARN 값을 적어둡니다. 계정의 사용자는 유지 관리 기간을 생성할 때 이 정보를 지정합니다.

작업 3: 유지 관리 기간 작업을 등록할 수 있는 사용자에 대한 권한 구성(콘솔)

유지 관리 기간에 태스크를 등록할 때 사용자 정의 서비스 역할 또는 Systems Manager 서비스 연결 역할을 지정하여 실제 태스크를 실행합니다. 이것은 서비스가 사용자를 대신하여 태스크를 실행할 때 수임하는 역할입니다. 그 전에 태스크 자체를 등록하려면 IAM 엔터티(사용자 또는 그룹 등)에 IAM PassRole 정책을 할당합니다. 이를 통해 IAM 엔터티(사용자 또는 그룹)는 유지 관리 기간에 해당 태스

크를 등록하는 과정에서 태스크를 실행할 때 사용해야 하는 역할을 지정할 수 있습니다. 자세한 내용은 IAM 사용 설명서에서 AWS 서비스에 역할을 전달할 사용자 권한 부여를 참조하세요.

유지 관리 기간 태스크를 등록할 수 있는 사용자에 대한 권한 구성

IAM 엔터티(사용자, 역할 또는 그룹)가 관리자 권한으로 설정된 경우 사용자 또는 역할은 Maintenance Windows에 액세스할 수 있습니다. 관리자 권한이 없는 IAM 엔터티의 경우 관리자가 IAM 엔터티에 다음 권한을 부여해야 합니다. 다음은 유지 관리 기간에 태스크를 등록하는 데 필요한 최소 권한입니다.

- `AmazonSSMFullAccess` 관리형 정책 또는 유사한 권한을 제공하는 정책.
- 다음 `iam:PassRole` 및 `iam:ListRoles` 권한.

```
{
    "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*은 앞에서 생성한 사용자 지정 유지 관리 기간 역할의 이름을 나타냅니다.

*account-id*는 AWS 계정의 ID를 나타냅니다. 리소스 `arn:aws:iam::account-id:role/`에 대해 이 권한을 추가하면 사용자가 유지 관리 기간 작업을 생성할 때 콘솔에서 고객 역할을 보고 선택할 수 있습니다. `arn:aws:iam::account-id:role/aws-service-role/`

ssm.amazonaws.com/에 대해 이 권한을 추가하면 사용자가 유지 관리 기간 태스크를 생성할 때 콘솔에서 Systems Manager 서비스 연결 역할을 선택할 수 있습니다.

액세스 권한을 제공하려면 사용자, 그룹 또는 역할에 권한을 추가하세요:

- AWS IAM Identity Center의 사용자 및 그룹:

  권한 세트를 생성합니다. AWS IAM Identity Center 사용 설명서의 권한 세트 생성의 지침을 따르세요.

- ID 제공자를 통해 IAM에서 관리되는 사용자:

  ID 페더레이션을 위한 역할을 생성합니다. IAM 사용 설명서의 서드 파티 자격 증명 공급자의 역할 만들기(연합)의 지침을 따르세요.

- IAM 사용자:
  - 사용자가 맡을 수 있는 역할을 생성합니다. IAM 사용 설명서에서 IAM 사용자의 역할 생성의 지침을 따르세요.
  - (권장되지 않음)정책을 사용자에게 직접 연결하거나 사용자를 사용자 그룹에 추가합니다. IAM 사용 설명서에서 사용자(콘솔)에 권한 추가의 지침을 따르세요.

유지 관리 기간 태스크를 등록할 수 있는 그룹에 대한 권한을 구성하려면(콘솔)

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 [사용자 그룹(User groups)]을 선택합니다.
3. 그룹 목록에서 iam:PassRole 권한을 할당하려는 그룹의 이름을 선택합니다.
4. 권한(Permissions) 탭에서 권한 추가, 인라인 정책 생성(Add permissions, Create inline policy)을 선택하고 JSON 탭을 선택합니다.
5. 상자의 기본 내용을 다음으로 바꿉니다.

```
{
    "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*은 앞에서 생성한 사용자 지정 유지 관리 기간 역할의 이름을 나타냅니다.

*account-id*는 AWS 계정의 ID를 나타냅니다. 리소스 `arn:aws:iam::account-id:role/`에 대해 이 권한을 추가하면 사용자가 유지 관리 기간 작업을 생성할 때 콘솔에서 고객 역할을 보고 선택할 수 있습니다. `arn:aws:iam::account-id:role/aws-service-role/ssm.amazonaws.com/`에 대해 이 권한을 추가하면 사용자가 유지 관리 기간 태스크를 생성할 때 콘솔에서 Systems Manager 서비스 연결 역할을 선택할 수 있습니다.

6. 정책 검토를 선택합니다.
7. [정책 검토(Review policy)] 페이지에서 [이름(Name)] 상자에 이 PassRole 정책을 식별하는 이름(예: **my-group-iam-passrole-policy**)을 입력한 후 [정책 생성(Create policy)]을 선택합니다.

태스크 4: 유지 관리 기간 태스크를 등록할 수 없는 사용자에 대한 권한 구성

`ssm:RegisterTaskWithMaintenanceWindow` 권한을 개별 사용자에 대해 거부할 것인지 아니면 그룹에 대해 거부할 것인지에 따라 다음 절차 중 하나를 사용하여 사용자가 유지 관리 기간에 태스크를 등록하지 못하게 합니다.

유지 관리 기간 태스크를 등록할 수 없는 사용자에 대한 권한 구성

- 관리자는 IAM 엔터티에 다음과 같은 제한을 추가해야 합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "ssm:RegisterTaskWithMaintenanceWindow",
```

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

유지 관리 기간 태스크를 등록할 수 없는 그룹에 대한 권한을 구성하려면(콘솔)

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 [사용자 그룹(User groups)]을 선택합니다.
3. 그룹 목록에서 `ssm:RegisterTaskWithMaintenanceWindow` 권한을 거부하려는 그룹의 이름을 선택합니다.
4. 권한(Permissions) 탭에서 권한 추가, 인라인 정책 생성(Add permissions, Create inline policy)을 선택합니다.
5. [JSON] 탭을 선택하고 상자의 기본 내용을 다음과 같이 바꿉니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "ssm:RegisterTaskWithMaintenanceWindow",
            "Resource": "*"
        }
    ]
}
```

6. 정책 검토를 선택합니다.
7. [정책 검토(Review policy)] 페이지에서 [이름(Name)] 상자에 이 정책을 식별하는 이름(예: **`my-groups-deny-mw-tasks-policy`**)을 입력한 후 [정책 생성(Create policy)]을 선택합니다.

유지 관리 기간 작업(콘솔)

이 섹션에서는 AWS Systems Manager 콘솔을 사용하여 유지 관리 기간을 생성, 구성, 업데이트 및 삭제하는 방법을 설명합니다. 또한 이 단원에서는 유지 관리 기간의 대상 및 작업 관리에 대한 정보를 제공합니다.

> **Important**
>
> 처음에는 테스트 환경에서 유지 관리 기간을 생성하고 구성하는 것이 좋습니다.

시작하기 전 준비 사항

유지 관리 기간을 생성하기 전에 AWS Systems Manager의 기능인 Maintenance Windows에 대한 액세스를 구성해야 합니다. 자세한 내용은 Maintenance Windows 설정 단원을 참조하십시오.

주제

- 유지 관리 기간 생성(콘솔)
- 유지 관리 기간에 대상 할당(콘솔)
- 유지 관리 기간에 작업 할당(콘솔)
- 유지 관리 기간 비활성화 또는 활성화
- 유지 관리 기간 리소스 업데이트 또는 삭제(콘솔)

유지 관리 기간 생성(콘솔)

이 절차에서는 AWS Systems Manager의 기능인 Maintenance Windows에서 유지 관리 기간을 생성합니다. 이름, 일정, 기간 등의 기본 옵션을 지정할 수 있습니다. 이후 단계에서는 유지 관리 기간 실행 중 실행되는 태스크 및 업데이트하는 대상 또는 리소스를 선택합니다.

> **Note**
>
> 유지 관리 기간에 대한 다양한 스케줄 관련 옵션이 서로 관련되는 방법에 대한 자세한 내용은 유지 관리 기간 예약 및 유효 기간 옵션 섹션을 참조하세요.
> `--schedule` 옵션 작업에 대한 자세한 내용은 참조: Systems Manager용 Cron 및 Rate 표현식 섹션을 참조하세요.

유지 관리 기간을 생성하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Maintenance Windows를 선택합니다.
3. 유지 관리 기간 생성을 선택합니다.
4. 이름(Name)에 이 유지 관리 기간을 식별하는 데 유용한 설명 이름을 입력합니다.

5. (선택 사항) 설명(Description)에 이 유지 관리 기간이 사용되는 방식을 식별하는 설명을 입력합니다.
6. (선택 사항) 관리형 노드가 대상으로 등록되어 있지 않은 경우에도 관리형 노드에 대해 유지 관리 기간 태스크를 실행하도록 허용하려면 등록되지 않은 대상 허용(Allow unregistered targets)을 선택합니다.

   이 옵션을 선택한 경우 작업을 유지 관리 기간에 등록할 때 등록되지 않은 노드를 노드 ID를 사용하여 선택할 수 있습니다.

   이 옵션을 선택하지 않은 경우 태스크를 유지 관리 기간에 등록할 때 이전에 등록된 대상을 선택해야 합니다.
7. 세 가지 예약 옵션 중 하나를 사용하여 유지 관리 기간에 대한 일정을 지정합니다.

   cron/rate 표현식 작성에 대한 자세한 내용은 참조: Systems Manager용 Cron 및 Rate 표현식 섹션을 참조하세요.
8. 기간에 유지 관리 기간을 실행할 시간을 입력합니다. 지정된 값은 유지 관리가 시작하는 시간을 기준으로 유지 관리 기간의 특정 종료 시간을 결정합니다. 다음 단계에서 해당 종료 시간에서 작업 개시 중지에 지정된 시간을 뺀 시간 이후부터는 유지 관리 기간 작업을 시작할 수 없습니다.

   예를 들어, 유지 관리 기간이 오후 3시에 시작되고, 기간이 3시간으로 설정되고, 작업 개시 중지 값이 1시간인 경우, 오후 5시 이후부터는 유지 관리 기간 작업을 시작할 수 없습니다.
9. 작업 개시 중지에 유지 관리 기간 종료 이전에 시스템에서 실행할 새 작업 예약을 중지해야 하는 시간을 입력합니다.
10. (선택 사항) 기간 시작일(Window start date)에 유지 관리 기간을 활성화하려는 날짜 및 시간을 ISO-8601 확장 형식으로 지정합니다. 이를 사용하면 향후 지정한 날짜까지 유지 관리 기간의 정품 인증을 지연시킬 수 있습니다.

    > **Note**
    >
    > 과거의 시작 날짜 및 시간을 지정할 수 없습니다.

11. (선택 사항) 기간 종료일(Window end date)에 유지 관리 기간을 비활성화하려는 날짜 및 시간을 ISO-8601 확장 형식으로 지정합니다. 이를 사용하면 유지 관리 기간이 더 이상 실행되지 않는 미래의 날짜 및 시간을 설정할 수 있습니다.
12. (선택 사항) 일정 시간대(Schedule timezone)에 예약된 유지 관리 기간 실행의 기준이 될 시간대를 IANA(Internet Assigned Numbers Authority) 형식으로 지정합니다. 예를 들어 "America/Los_Angeles", "etc/UTC" 또는 "Asia/Seoul"입니다.

올바른 형식에 대한 자세한 내용은 IANA 웹 사이트에서 표준 시간대 데이터베이스를 참조하십시오.

13. (선택 사항) 일정 오프셋(Schedule offset)에 유지 관리 기간을 실행하기 전에 cron 또는 rate 표현식에 의해 지정된 날짜 및 시간 이후에 대기할 일 수를 입력합니다. 1~6일을 지정할 수 있습니다.

> **Note**
>
> 이 옵션은 cron 또는 rate 표현식을 수동으로 입력하여 일정을 지정한 경우에만 사용할 수 있습니다.

14. (선택 사항) 태그 관리 영역에서 하나 이상의 태그 키 이름/값 페어를 유지 관리 기간에 적용합니다.

    태그는 리소스에 할당하는 선택적 메타데이터입니다. 태그를 사용하면 용도, 소유자 또는 환경을 기준으로 하는 등 리소스를 다양한 방식으로 분류할 수 있습니다. 예를 들어 유지 관리 기간에 태그를 지정하여 이 기간이 실행하는 작업 유형, 대상 유형, 이 기간이 실행되는 환경을 식별하려는 경우 다음 키 이름/값 페어를 지정할 수 있습니다.

    - `Key=TaskType,Value=AgentUpdate`
    - `Key=OS,Value=Windows`
    - `Key=Environment,Value=Production`

15. 유지 관리 기간 생성을 선택합니다. 그러면 유지 관리 기간 페이지로 돌아갑니다. 방금 생성한 유지 관리 기간의 상태는 활성입니다.

유지 관리 기간에 대상 할당(콘솔)

이 절차에서는 대상을 유지 관리 기간에 등록합니다. 다시 말해 유지 관리 기간이 작업을 수행하는 리소스를 지정합니다.

> **Note**
>
> 단일 유지 관리 기간 태스크가 여러 대상에 등록된 경우 해당 태스크 호출은 병렬이 아닌 순차적으로 발생합니다. 태스크를 여러 대상에서 동시에 실행해야 하는 경우 각 대상에 대해 태스크를 개별적으로 등록하고 각 태스크에 동일한 우선순위 수준을 할당합니다.

유지 관리 기간에 대상을 할당하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Maintenance Windows를 선택합니다.
3. 유지 관리 기간 목록에서 대상을 추가할 유지 관리 기간을 선택합니다.
4. 작업, 대상 등록을 차례로 선택합니다.
5. (선택 사항) 대상 이름에 대상의 이름을 입력합니다.
6. (선택 사항) 설명에 설명을 입력합니다.
7. (옵션) [소유자(Owner information)] 정보에서 이 유지 관리 기간 내에 이러한 대상에 대해 태스크를 실행하는 동안 발생한 모든 Amazon EventBridge 이벤트에 추가할 정보를 지정합니다.

   EventBridge를 사용하여 Systems Manager 이벤트 모니터링에 대한 자세한 내용은 Amazon EventBridge로 Systems Manager 이벤트 모니터링 섹션을 참조하세요.
8. 대상 영역에서 아래 표에 나열된 옵션 중 하나를 선택합니다.

옵션	설명
인스턴스 태그 지정	인스턴스 태그 지정(Specify instance tags)에 태그 키 1개 이상과 계정의 관리형 노드에 추가했거나 앞으로 추가할 태그 값(옵션)을 지정합니다. 유지 관리 기간이 실행되면 태그가 추가된 모든 관리형 노드에서 작업을 실행하려고 합니다. 두 개 이상의 태그 키를 지정하는 경우, 대상 그룹에 포함되도록 지정한 모든 태그 키 및 값으로 노드에 태그를 지정해야 합니다.
수동으로 인스턴스 선택	목록에서 유지 관리 기간 대상에 추가할 노드마다 확인란을 선택합니다. 계정에서 Systems Manager와 함께 사용하도록 구성된 노드가 모두 목록에 포함됩니다. 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

옵션	설명
	엣지 디바이스, 온프레미스 서버 및 가상 머신(VM)에 대한 자세한 내용은 하이브리드 및 멀티클라우드 환경에서 Systems Manager 사용하기 섹션을 참조하세요.

옵션	설명
리소스 그룹 선택	리소스 그룹(Resource group)에 있는 목록에서 계정의 기존 리소스 그룹 이름을 선택합니다. 리소스 그룹 생성 및 작업에 대한 자세한 내용은 다음 주제를 참조하십시오. • AWS Resource Groups 사용 설명서의 리소스 그룹이란 무엇인가요? • AWS 뉴스 블로그의 AWS에 대한 리소스 그룹 및 태깅 (선택 사항) 리소스 유형(Resource types)에서 사용 가능한 리소스 유형을 최대 5개까지 선택하거나 모든 리소스 유형(All resource types)을 선택합니다. 유지 관리 기간에 할당할 태스크가 대상에 추가한 리소스 유형 중 하나에서 유효하지 않을 경우에는 시스템이 오류를 보고할 수 있습니다. 이러한 오류가 발생하더라도 리소스 유형이 지원되는 작업은 계속해서 실행됩니다. 예를 들어 다음과 같은 리소스 유형을 대상에 추가한다고 가정하겠습니다. • `AWS::S3::Bucket` • `AWS::DynamoDB::Table` • `AWS::EC2::Instance` 하지만 나중에 작업을 유지 관리 기간에 추가하면서 패치 기준을 적용하거나 노드를 다시 부팅하는 등 추가하려는 작업을 노드 관련 작업으로 제한하려고 합니다. 그러면 유지 관리 기간 로그에서 Amazon Simple Storage

옵션	설명
	Service(Amazon S3) 버킷 또는 Amazon DynamoDB 테이블을 찾을 수 없다는 오류가 보고될 수 있습니다. 하지만 유지 관리 기간에서는 리소스 그룹의 노드에 대한 작업이 계속해서 실행됩니다.

9. 대상 등록을 선택합니다.

이 유지 관리 기간에 더 많은 대상을 할당하려면 대상 탭을 선택한 후 대상 등록을 선택합니다. 이 옵션을 사용하여 다른 대상 방법을 선택할 수 있습니다. 예를 들어, 이전에 노드 ID를 사용하여 노드를 대상으로 지정한 경우 관리형 노드에 적용되는 태그를 지정하거나, 리소스 그룹에서 리소스 유형을 선택하여 새로운 대상과 대상 노드를 등록할 수 있습니다.

유지 관리 기간에 작업 할당(콘솔)

이 절차에서는 유지 관리 기간에 작업을 추가합니다. 태스크는 유지 관리 기간이 실행될 때 수행되는 작업입니다.

다음 4가지 유형의 작업을 유지 관리 기간에 추가할 수 있습니다.

- AWS Systems Manager Run Command 명령
- Systems Manager Automation 워크플로
- AWS Step Functions 작업
- AWS Lambda 함수

> **Important**
>
> Maintenance Windows의 IAM 정책에서는 Lambda 함수 이름(또는 별칭) 앞에 접두사 `SSM`을 추가해야 합니다. 이러한 유형의 태스크를 등록하기 전에 AWS Lambda에서 `SSM`을 포함하도록 이름을 업데이트합니다. 예를 들어 Lambda 함수 이름이 `MyLambdaFunction`인 경우 `SSMMyLambdaFunction`으로 변경합니다.

유지 관리 기간에 작업을 할당하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.

2. 탐색 창에서 Maintenance Windows를 선택합니다.
3. 유지 관리 기간 목록에서 유지 관리 기간을 선택합니다.
4. [작업(Actions)]을 선택한 다음 유지 관리 기간에 등록할 태스크 유형에 대한 옵션을 선택합니다.

   - 실행 명령 작업 등록
   - 자동화 작업 등록
   - Lambda 작업 등록
   - Step Functions 작업 등록

   > **Note**
   >
   > 유지 관리 기간 태스크는 Step Functions 표준 상태 머신 워크플로만 지원합니다. 익스프레스 상태 머신 워크플로는 지원하지 않습니다. 상태 머신 워크플로 유형에 대한 자세한 내용은 AWS Step Functions개발자 안내서의 표준 워크플로와 익스프레스 워크플로를 참조하세요.

5. (옵션) [이름(Name)]에 태스크의 이름을 입력합니다.
6. (선택 사항) 설명에 설명을 입력합니다.
7. 새로운 작업 호출 컷오프에서는, 유지 관리 기간 컷오프 시간에 도달한 후 새 작업 호출을 시작하지 않으려면 활성화를 선택합니다.

   이 옵션이 비활성화된 경우, 컷오프 시간에 도달하면 작업이 계속 실행되고 완료될 때까지 새 작업 호출이 시작됩니다.

   > **Note**
   >
   > 이 옵션을 활성화할 때 완료되지 않은 작업 상태는 `TIMED_OUT`입니다.

8. 이 단계에서는 선택한 작업 유형의 하위 단계를 따르세요.

   Run Command

   1. 명령 문서 목록에서 실행할 작업을 정의하는 Systems Manager Command 문서(SSM 문서)를 선택합니다.
   2. 문서 버전에서 사용할 문서 버전을 선택합니다.

3. 작업 우선순위에서 이 작업의 우선순위를 지정합니다. 가장 높은 우선순위는 0입니다. 유지 관리 기간의 작업은 우선순위에 따라 예약되며, 같은 우선순위의 작업은 동시에 예약됩니다.

Automation

1. 자동화 문서 목록에서 실행할 작업을 정의하는 자동화 런북을 선택합니다.
2. [문서 버전(Document version)]에서 사용할 실행서 버전을 선택합니다.
3. 작업 우선순위에서 이 작업의 우선순위를 지정합니다. 가장 높은 우선순위는 0입니다. 유지 관리 기간의 작업은 우선순위에 따라 예약되며, 같은 우선순위의 작업은 동시에 예약됩니다.

Lambda

1. Lambda 파라미터 영역의 목록에서 Lambda 함수를 선택합니다.
2. (선택 사항) 포함하려는 페이로드(Payload), 클라이언트 컨텍스트(Client Context) 또는 한정자(Qualifier)에 대한 콘텐츠를 제공합니다.

   > **Note**
   >
   > 경우에 따라 의사 파라미터를 `Payload` 값의 일부로 사용할 수 있습니다. 그다음 유지 관리 기간 작업이 실행되면 의사 파라미터 자리표시자 대신 올바른 값을 전달합니다. 자세한 설명은 유지 관리 기간 작업 등록 시 의사 파라미터 사용을 참조하세요.

3. 작업 우선순위에서 이 작업의 우선순위를 지정합니다. 가장 높은 우선순위는 0입니다. 유지 관리 기간의 작업은 우선순위에 따라 예약되며, 같은 우선순위의 작업은 동시에 예약됩니다.

Step Functions

1. Step Functions 파라미터 영역의 목록에서 상태 머신을 선택합니다.
2. (선택 사항) 상태 머신 실행의 이름과 포함하려는 입력(Input)의 콘텐츠를 제공합니다.

> **Note**
>
> 경우에 따라 의사 파라미터를 `Input` 값의 일부로 사용할 수 있습니다. 그다음 유지 관리 기간 작업이 실행되면 의사 파라미터 자리표시자 대신 올바른 값을 전달합니다. 자세한 설명은 유지 관리 기간 작업 등록 시 의사 파라미터 사용을 참조하세요.

3. 작업 우선순위에서 이 작업의 우선순위를 지정합니다. 가장 높은 우선순위는 `0`입니다. 유지 관리 기간의 작업은 우선순위에 따라 예약되며, 같은 우선순위의 작업은 동시에 예약됩니다.

9. [대상(Targets)] 영역에서 다음 중 하나를 선택합니다.

   - [등록된 목표 그룹 선택(Selecting registered target groups)]: 현재 유지 관리 기간에 등록한 유지 관리 기간 대상을 하나 이상 선택합니다.
   - [등록되지 않은 대상 선택(Selecting unregistered targets)]: 사용 가능한 리소스를 태스크의 대상으로 하나씩 선택합니다.

     예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.
   - [태스크 대상 불필요(Task target not required)]: 태스크의 대상이 Run Command 유형 태스크를 제외한 모든 태스크에 대해 다른 기능에 이미 지정되었을 수 있습니다.

     유지 관리 기간 Run Command 유형 태스크에 대해 하나 이상의 대상을 지정합니다. 태스크에 따라 대상은 다른 유지 관리 기간 태스크 유형(Automation, AWS Lambda 및 AWS Step Functions)에 대해 옵션입니다. 대상을 지정하지 않는 태스크 실행에 대한 자세한 내용은 대상 없이 유지 관리 기간 태스크 등록 섹션을 참조하세요.

     > **Note**
     >
     > 대부분의 경우 자동화 태스크의 대상을 명시적으로 지정할 필요가 없습니다. 예를 들어 `AWS-UpdateLinuxAmi` 실행서를 사용하여 Linux용 Amazon Machine Image(AMI)를 업데이트하는 Automation 유형 태스크를 생성한다고 가정해 보겠습니다. 태스크가 실행되면 AMI는 사용 가능한 최신 Linux 배포 패키지와 Amazon 소프트웨어로 업데이트됩니다. AMI에서 생성된 새 인스턴스에는 이러한 업데이트가 이미 설치되어 있습니다. 업데이트할 AMI의 ID가 실행서에 대한 입력 파라미터에 지정되어 유지 관리 기간 태스크에서 대상을 다시 지정할 필요가 없습니다.

10. 자동화 작업만 해당:

입력 파라미터(Input parameters) 영역에서 태스크를 실행하는 데 필요한 필수 또는 선택적 파라미터에 대한 값을 제공합니다.

> **Note**
>
> 경우에 따라 특정 입력 파라미터 값에 의사 파라미터를 사용할 수 있습니다. 그다음 유지 관리 기간 작업이 실행되면 의사 파라미터 자리표시자 대신 올바른 값을 전달합니다. 자세한 설명은 유지 관리 기간 작업 등록 시 의사 파라미터 사용을 참조하세요.

11. Rate control(속도 제어)에서

    - Concurrency(동시성)에서 명령을 동시에 실행할 관리형 노드의 백분율 또는 개수를 지정합니다.

      > **Note**
      >
      > 관리형 노드에 적용할 태그를 지정하거나, AWS 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상 수를 제한합니다.

    - Error threshold(오류 임계값)에서, 명령이 노드의 개수 또는 백분율에서 실패한 후 다른 관리형 노드에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째 오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.

12. (선택 사항) IAM 서비스 역할의 경우 유지 관리 기간 작업을 실행할 때 Systems Manager에서 수임할 수 있는 권한을 제공할 역할을 선택합니다.

    서비스 역할 ARN을 지정하지 않으면 Systems Manager에서 계정의 서비스 연결 역할을 사용합니다. Systems Manager에 대한 적절한 서비스 연결 역할이 계정에 없는 경우 작업이 등록될 때 해당 역할이 생성됩니다.

    > **Note**
    >
    > 보안 태세를 개선하려면 유지 관리 기간 작업을 실행하기 위한 사용자 지정 정책 및 사용자 지정 서비스 역할을 생성하는 것이 좋습니다. 특정 유지 관리 기간 작업에 필요한 권한만 제공하도록 정책을 작성할 수 있습니다. 자세한 내용은 콘솔을 사용하여 유지 관리 기간에 대한 권한 구성 단원을 참조하십시오.

13. Run Command 작업만 해당:

    (선택 사항) 출력 옵션(Output options)에서 다음을 수행합니다.

    - S3에 쓰기 사용(Enable writing to S3) 확인란을 선택하여 파일에 명령 출력을 저장합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.
    - CloudWatch 출력(CloudWatch output) 확인란을 선택하여 Amazon CloudWatch Logs에 전체 출력을 씁니다. CloudWatch Logs 로그 그룹의 이름을 입력합니다.

    > **Note**
    >
    > 데이터를 S3 버킷 또는 CloudWatch Logs에 쓰는 기능을 부여하는 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아닌 노드에 할당된 인스턴스 프로파일의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성을 참조하세요. 또한 지정된 S3 버킷 또는 로그 그룹이 다른 AWS 계정에 있는 경우 노드와 연결된 인스턴스 프로파일은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

14. Run Command 작업만 해당:

    SNS notifications(SNS 알림) 섹션에서, 명령 실행 상태에 대한 알림이 전송되도록 하려면 Enable SNS notifications(SNS 알림 활성화) 확인란을 선택합니다.

    Run Command에 대한 Amazon SNS 알림 구성에 대한 자세한 내용은 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 섹션을 참조하세요.

15. Run Command 작업만 해당:

    [파라미터(Parameters)] 영역에서 문서의 파라미터를 지정합니다.

    > **Note**
    >
    > 경우에 따라 특정 입력 파라미터 값에 의사 파라미터를 사용할 수 있습니다. 그다음 유지 관리 기간 작업이 실행되면 의사 파라미터 자리표시자 대신 올바른 값을 전달합니다. 자세한 설명은 유지 관리 기간 작업 등록 시 의사 파라미터 사용을 참조하세요.

16. Run Command 및 자동화 작업만 해당:

    (선택 사항) CloudWatch 경보 영역에서 모니터링을 위해 작업에 적용할 기존 CloudWatch 경보를 경보 이름으로 선택합니다.

경보가 활성화되면 작업이 중지됩니다.

> **Note**
>
> CloudWatch 경보를 태스크에 연결하려면 태스크를 실행하는 IAM 보안 주체에 `iam:createServiceLinkedRole` 작업에 대한 권한이 있어야 합니다. CloudWatch 경보에 대한 자세한 내용은 Amazon CloudWatch 경보 사용을 참조하세요.

17. 작업 유형에 따라 다음 중 하나를 선택합니다.

- 실행 명령 작업 등록
- 자동화 작업 등록
- Lambda 작업 등록
- Step Functions 작업 등록

유지 관리 기간 비활성화 또는 활성화

AWS Systems Manager의 기능인 Maintenance Windows에서 유지 관리 기간을 비활성화하거나 활성화할 수 있습니다. 유지 관리 기간을 한 번에 하나씩 선택하여 유지 관리 기간 실행을 비활성화하거나 활성화할 수 있습니다. 유지 관리 기간을 여러 개 또는 모두 선택하여 활성화하거나 비활성화할 수도 있습니다.

이 섹션에서는 Systems Manager 콘솔을 사용하여 유지 관리 기간을 비활성화하거나 활성화하는 방법을 설명합니다. AWS Command Line Interface(AWS CLI)를 사용하여 이 작업을 수행하는 방법의 예는 자습서: 유지 관리 기간 업데이트(AWS CLI) 섹션을 참조하세요.

주제

- 유지 관리 기간 비활성화(콘솔)
- 유지 관리 기간 활성화(콘솔)

유지 관리 기간 비활성화(콘솔)

유지 관리 기간을 비활성화하여 지정된 기간의 작업을 일시 중지할 수 있으며, 나중에 다시 활성화하여 계속 사용할 수 있습니다.

유지 관리 기간 비활성화 방법

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Maintenance Windows를 선택합니다.
3. 비활성화하려는 유지 관리 기간 옆에 있는 확인란을 사용하여 유지 관리 기간을 하나 이상 선택합니다.
4. 작업 메뉴에서 유지 관리 기간 비활성화를 선택합니다. 시스템에 작업을 확인하라는 메시지가 표시됩니다.

유지 관리 기간 활성화(콘솔)

유지 관리 기간을 활성화하여 작업을 재개할 수 있습니다.

> **Note**
>
> 유지 관리 기간에 rate 일정을 사용하고 시작 날짜가 현재 과거 날짜 및 시간으로 설정된 경우 현재 날짜 및 시간이 유지 관리 기간의 시작 날짜로 사용됩니다. 유지 관리 기간을 활성화하기 전이나 후에 유지 관리 기간의 시작 날짜를 변경할 수 있습니다. 자세한 설명은 유지 관리 기간 리소스 업데이트 또는 삭제(콘솔)을 참조하세요.

유지 관리 기간 활성화 방법

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Maintenance Windows를 선택합니다.
3. 유지 관리 기간 옆의 확인란을 선택하여 활성화합니다.
4. 작업, 유지 관리 기간 활성화를 선택합니다. 시스템에 작업을 확인하라는 메시지가 표시됩니다.

유지 관리 기간 리소스 업데이트 또는 삭제(콘솔)

AWS Systems Manager의 기능인 Maintenance Windows에서 유지 관리 기간을 업데이트하거나 삭제할 수 있습니다. 유지 관리 기간의 대상 또는 작업을 업데이트하거나 삭제할 수도 있습니다. 유지 관리 기간에 대한 세부 정보를 편집할 경우 일정, 대상 및 작업을 변경할 수 있습니다. 또한 기간, 대상 및 작업의 용도를 쉽게 이해하고 기간의 대기열을 쉽게 관리할 수 있도록 기간, 대상 및 작업에 대한 이름 및 설명을 지정할 수 있습니다.

이 섹션에서는 Systems Manager 콘솔을 사용하여 유지 관리 기간, 대상 및 태스크를 업데이트하거나 삭제하는 방법을 설명합니다. AWS Command Line Interface(AWS CLI)를 사용하여 이 작업을 수행하는 방법의 예는 자습서: 유지 관리 기간 업데이트(AWS CLI) 섹션을 참조하세요.

주제

- 유지 관리 기간 업데이트 또는 삭제(콘솔)
- 유지 관리 기간 대상 업데이트 또는 등록 취소(콘솔)
- 유지 관리 기간 태스크 업데이트 또는 등록 취소(콘솔)

유지 관리 기간 업데이트 또는 삭제(콘솔)

유지 관리 기간의 이름, 설명, 일정 및 유지 관리 기간에서 등록되지 않은 대상을 허용할지 여부를 변경하도록 유지 관리 기간을 업데이트할 수 있습니다.

유지 관리 기간을 업데이트하거나 삭제하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Maintenance Windows를 선택합니다.
3. 업데이트 또는 삭제할 유지 관리 기간 옆에 있는 버튼을 선택하고 다음 중 하나를 수행합니다.
   - 삭제를 선택합니다. 시스템에 작업을 확인하라는 메시지가 표시됩니다.
   - 편집을 선택합니다. 유지 관리 기간 편집(Edit maintenance window) 페이지에서 원하는 값과 옵션을 변경한 다음 변경 사항 저장(Save changes)을 선택합니다.

     선택 가능한 구성에 대한 자세한 내용은 유지 관리 기간 생성(콘솔) 섹션을 참조하세요.

유지 관리 기간 대상 업데이트 또는 등록 취소(콘솔)

유지 관리 기간의 대상을 업데이트하거나 등록 취소할 수 있습니다. 유지 관리 기간 대상 업데이트를 선택하면 새 대상 이름, 설명 및 소유자를 지정할 수 있습니다. 다른 대상을 선택할 수도 있습니다.

유지 관리 기간의 대상을 업데이트하거나 삭제하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Maintenance Windows를 선택합니다.
3. 업데이트할 유지 관리 기간의 이름을 선택하고 대상(Targets) 탭을 선택한 후 다음 중 하나를 수행합니다.

- 대상을 업데이트하려면 업데이트할 대상 옆에 있는 버튼을 선택하고 편집(Edit)을 선택합니다.
- 대상 등록을 취소하려면 등록을 취소할 대상 옆에 있는 버튼을 선택하고 대상 등록 취소(Deregister target)를 선택합니다. [유지 관리 기간 대상 등록 취소(Deregister maintenance windows target)] 대화 상자에서 [등록 취소(Deregister)]를 선택합니다.

유지 관리 기간 태스크 업데이트 또는 등록 취소(콘솔)

유지 관리 기간의 태스크를 업데이트하거나 등록 취소할 수 있습니다. 업데이트를 선택하면 새 작업 이름, 설명 및 소유자를 지정할 수 있습니다. Run Command 및 Automation 태스크의 경우 태스크에 대한 다른 SSM 문서를 선택할 수 있습니다. 하지만 작업을 편집하여 유형을 변경할 수 없습니다. 예를 들어 자동화 작업을 생성한 경우 해당 작업을 편집하여 Run Command 작업으로 변경할 수 없습니다.

유지 관리 기간의 작업을 업데이트하거나 삭제하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Maintenance Windows를 선택합니다.
3. 업데이트할 유지 관리 기간의 이름을 선택합니다.
4. 태스크(Tasks) 탭을 선택하고 업데이트할 태스크 옆에 있는 버튼을 선택합니다.
5. 다음 중 하나를 수행하십시오.
   - 태스크를 등록 취소하려면 태스크 등록 취소(Deregister task)를 선택합니다.
   - 태스크를 편집하려면 [편집(Delete)]을 선택합니다. 원하는 값과 옵션을 변경한 다음 태스크 편집(Edit Task)을 선택합니다.

Systems Manager Maintenance Windows 튜토리얼(AWS CLI)

이 단원에는 AWS Command Line Interface(AWS CLI)를 사용하여 다음을 수행하는 방법을 배울 수 있는 자습서가 포함되어 있습니다.

- 유지 관리 기간 생성 및 구성
- 유지 관리 기간에 대한 정보 보기
- 유지 관리 기간 작업 및 작업 실행에 대한 정보 보기
- 유지 관리 기간 업데이트
- 유지 관리 기간 삭제

사전 조건 완료

해당 자습서를 시작하기 전에 다음 사전 조건을 완료하십시오.

- 로컬 컴퓨터에서 AWS CLI 구성 - AWS CLI 명령을 실행하기 전에 로컬 시스템에 CLI를 설치하고 구성해야 합니다. 자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트 및 AWS Tools for PowerShell 설치를 참조하세요.
- 유지 관리 기간 역할 및 사용 권한 확인 - 계정의 AWS 관리자는 CLI를 사용하여 유지 관리 기간을 관리하는 데 필요한 AWS Identity and Access Management(IAM) 권한을 부여해야 합니다. 자세한 설명은 Maintenance Windows 설정을 참조하세요.
- Systems Manager와 호환되는 인스턴스 생성 또는 구성: 자습서를 완료하려면 Systems Manager에 사용하도록 구성된 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스가 하나 이상 필요합니다. 즉, SSM Agent가 인스턴스에 설치되고 Systems Manager용 IAM 인스턴스 프로파일이 인스턴스에 연결됩니다.

  에이전트가 사전 설치된 1개의 AWS 관리형 Amazon Machine Image(AMI)에서 인스턴스를 실행하는 것이 좋습니다. 자세한 내용은 SSM Agent가 사전 설치된 상태로 AMIs 검색 단원을 참조하십시오.

  인스턴스에 SSM Agent를 설치하는 방법에 대한 자세한 내용은 다음 주제를 참조하십시오.

  - SSM Agent용 EC2 인스턴스에 수동으로 Windows Server 설치 및 제거
  - Linux용 EC2 인스턴스에 수동으로 SSM Agent 설치 및 제거

  인스턴스에 Systems Manager에 대한 IAM 권한 구성에 대한 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성을 참조하세요.
- 필요에 따라 추가 리소스 생성 - Systems Manager의 기능인 Run Command에는 이 사전 조건 주제에 나열된 것 이외의 리소스를 생성할 필요가 없는 많은 태스크를 포함합니다. 그래서 자습서 진행에 따라 처음 사용할 간단한 Run Command 작업을 제공합니다. 또한 이 주제의 앞부분에서 설명한 대로 Systems Manager에 사용하도록 구성된 EC2 인스턴스가 필요합니다. 해당 인스턴스를 구성한 후 간단한 Run Command 작업을 등록할 수 있습니다.

  Systems Manager Maintenance Windows 기능은 다음 4가지 유형의 태스크 실행을 지원합니다.

  - Run Command 명령
  - Systems Manager Automation 워크플로
  - AWS Lambda 함수
  - AWS Step Functions 작업

일반적으로 실행하려는 유지 관리 기간 작업에 추가 리소스가 필요한 경우 먼저 해당 리소스를 생성해야 합니다. 예를 들어 AWS Lambda 함수를 실행하는 유지 관리 기간을 생성하려면 시작하기 전에 Lambda 함수를 생성합니다. Run Command 태스크의 경우 명령 출력을 저장할 수 있는(저장할 계획이 있는 경우) S3 버킷을 생성합니다.

리소스 ID 추적

이 AWS CLI 자습서의 작업을 완료한 후에는 실행한 명령에 의해 생성된 리소스 ID를 추적합니다. 이러한 리소스 ID는 후속 명령의 입력으로 사용됩니다. 예를 들어 유지 관리 기간을 생성한 경우 시스템에서 다음 형식의 유지 관리 기간 ID를 제공합니다.

```
{
   "WindowId":"mw-0c50858d01EXAMPLE"
}
```

다음 시스템 생성 ID는 이 단원의 자습서에서 사용되므로 기록해 두십시오.

- `WindowId`
- `WindowTargetId`
- `WindowTaskId`
- `WindowExecutionId`
- `TaskExecutionId`
- `InvocationId`
- `ExecutionId`

또한 자습서에서 사용할 EC2 인스턴스의 ID가 필요합니다. 예: `i-02573cafcfEXAMPLE`

튜토리얼

- 자습서: 유지 관리 기간 생성 및 구성(AWS CLI)
- 자습서: 유지 관리 기간에 대한 정보 보기(AWS CLI)
- 자습서: 작업 및 작업 실행에 대한 정보 보기(AWS CLI)
- 자습서: 유지 관리 기간 업데이트(AWS CLI)
- 자습서: 유지 관리 기간 삭제(AWS CLI)

자습서: 유지 관리 기간 생성 및 구성(AWS CLI)

이 튜토리얼은 AWS Command Line Interface(AWS CLI)를 사용하여 유지 관리 기간 및 해당하는 대상과 태스크를 생성 및 구성하는 방법을 보여줍니다. 자습서의 주요 경로는 간단한 단계로 구성됩니다. 단일 유지 관리 기간을 만들고 단일 대상을 식별하고 유지 관리 기간을 실행할 간단한 작업을 설정합니다. 또한 더 복잡한 시나리오를 시도하는 데 사용할 수 있는 정보를 제공합니다.

이 튜토리얼의 단계를 수행하면서 *###* 기울임꼴 텍스트의 값을 원하는 옵션 및 ID로 바꿉니다. 예를 들어 유지 관리 기간 ID *mw-0c50858d01EXAMPLE*과 인스턴스 ID *i-02573cafcfEXAMPLE*을 생성하는 리소스의 ID로 바꿉니다.

내용

1단계: 유지 관리 기간 생성(AWS CLI)

이 단계에서는 유지 관리 기간을 생성하고 기본 옵션(예: 이름, 일정 및 기간)을 지정합니다. 이후 단계에서는 업데이트될 인스턴스와 실행될 작업을 선택합니다.

이 예제에서는 5분마다 실행되는 유지 관리 기간을 생성합니다. 일반적으로 유지 관리 기간을 이렇게 자주 실행하지는 않습니다. 그러나 이 속도로 설정하면 튜토리얼 결과를 빠르게 볼 수 있습니다. 태스크가 성공적으로 실행된 후 더 긴 주기로 변경하는 방법을 알려 드리겠습니다.

> **Note**
>
> 유지 관리 기간에 대한 다양한 스케줄 관련 옵션이 서로 관련되는 방법에 대한 자세한 내용은 유지 관리 기간 예약 및 유효 기간 옵션 섹션을 참조하세요.
> `--schedule` 옵션 작업에 대한 자세한 내용은 참조: Systems Manager용 Cron 및 Rate 표현식 섹션을 참조하세요.

유지 관리 기간을 생성하려면(AWS CLI)

1. AWS Command Line Interface(AWS CLI)를 열고 로컬 시스템에서 다음 명령을 실행하여 다음을 수행하는 유지 관리 기간을 생성합니다.

   - 필요한 경우 최대 2시간 동안 5분마다 실행합니다.

- 유지 관리 기간이 끝난 후 1시간 이내에 새 태스크가 시작되지 않도록 합니다.
- 연결되지 않은 대상(유지 관리 기간에 등록하지 않은 인스턴스)을 허용합니다.
- 사용자 지정 태그 사용을 통해 생성자가 자습서에서 사용하려는 목적임을 명시합니다.

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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "WindowId":"mw-0c50858d01EXAMPLE"
}
```

2. 이제 다음 명령을 실행하여 이 유지 관리 기간 및 계정에 이미 존재하는 다른 모든 유지 관리 기간에 대한 세부 정보를 확인합니다.

```
aws ssm describe-maintenance-windows
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
```

```
   "WindowIdentities":[
      {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Name": "My-First-Maintenance-Window",
            "Enabled": true,
            "Duration": 2,
            "Cutoff": 1,
            "NextExecutionTime": "2019-05-11T16:46:16.991Z"
      }
   ]
}
```

계속해서 2단계: 유지 관리 기간에 대상 노드 등록(AWS CLI)로 이동하십시오.

2단계: 유지 관리 기간에 대상 노드 등록(AWS CLI)

이 단계에서는 대상을 새 유지 관리 기간에 등록합니다. 이때 유지 관리 기간을 실행할 때 업데이트할 노드를 지정합니다.

노드 ID를 사용하여 여러 노드를 한 번에 등록하는 예제와 태그를 사용하여 여러 노드를 식별하는 예제, 그리고 리소스 그룹을 대상으로 지정하는 예제는 예제: 유지 관리 기간에 대상 등록 섹션을 참조하세요.

> **Note**
>
> Maintenance Windows 튜토리얼 사전 조건에서 설명한 대로 이 단계에서 사용할 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스를 이미 생성했어야 합니다.

유지 관리 기간에 대상 노드 등록(AWS CLI)

1. 로컬 시스템에서 다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "WindowTargetId":"e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
}
```

2. 이제 로컬 시스템에서 다음 명령을 실행하여 유지 관리 기간 대상의 세부 정보를 볼 수 있습니다.

Linux & macOS

```
aws ssm describe-maintenance-window-targets \
    --window-id "mw-0c50858d01EXAMPLE"
```

Windows

```
aws ssm describe-maintenance-window-targets ^
    --window-id "mw-0c50858d01EXAMPLE"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "Targets": [
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "WindowTargetId": "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE",
            "ResourceType": "INSTANCE",
            "Targets": [
                {
                    "Key": "InstanceIds",
                    "Values": [
                        "i-02573cafcfEXAMPLE"
                    ]
```

```
                }
            ]
        }
    ]
}
```

계속해서 3단계: 유지 관리 기간에 작업 등록(AWS CLI)로 이동하십시오.

예제: 유지 관리 기간에 대상 등록

2단계: 유지 관리 기간에 대상 노드 등록(AWS CLI)에 예시된 것처럼 노드 ID를 사용하여 단일 노드를 대상으로 등록할 수 있습니다. 또한 이 페이지의 명령 형식을 사용하여 하나 이상의 노드를 대상으로 등록할 수 있습니다.

일반적으로 유지 관리 기간 대상으로 사용할 노드를 식별하는 방법은 개별 노드 지정과 리소스 태그 사용 두 가지입니다. 리소스 태그를 사용하는 방법은 예제 2-3과 같이 더 많은 옵션이 제공됩니다.

또한 리소스 그룹을 1개 이상 유지 관리 기간 대상으로 지정할 수 있습니다. 리소스 그룹에는 노드를 비롯해 지원되는 AWS 리소스 유형이 다양하게 포함됩니다. 그런 다음 예제 4와 5에서는 리소스 그룹을 유지 관리 기간 대상에 추가하는 방법에 대해서 알아보겠습니다.

> **Note**
>
> 단일 유지 관리 기간 태스크가 여러 대상에 등록된 경우 해당 태스크 호출은 병렬이 아닌 순차적으로 발생합니다. 태스크를 여러 대상에서 동시에 실행해야 하는 경우 각 대상에 대해 태스크를 개별적으로 등록하고 각 태스크에 동일한 우선순위 수준을 할당합니다.

Resource Groups 생성 및 관리에 대한 자세한 내용은 AWS Resource Groups 사용 설명서의 Resource Groups란? 및 AWS 뉴스 블로그의 리소스 그룹 및 AWS를 위한 태깅을 참조하세요.

다음 예제에 지정된 경우 이외에 AWS Systems Manager의 기능인 Maintenance Windows에 대한 할당량에 대한 자세한 내용은 Amazon Web Services 일반 참조의 Systems Manager 서비스 할당량을 참조하세요.

예 1: 노드 ID를 사용하여 여러 대상 등록

로컬 시스템에서 다음 명령을 실행하여 여러 노드를 해당 노드 ID를 통해 대상으로 등록합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

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
```

권장 용도: 공통 노드 태그를 공유하지 않는 고유한 노드 그룹을 처음 유지 관리 기간에 등록할 때 매우 유용합니다.

할당량: 각 유지 관리 기간 대상에 최대 50개까지 노드를 지정할 수 있습니다.

예 2: 노드에 적용된 리소스 태그를 사용하여 대상 등록

로컬 시스템에서 다음 명령을 실행하여 사용자가 할당한 키 값 페어로 모두 태그 지정된 노드를 등록합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

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"
```

권장 용도: 공통 노드 태그를 공유하는 고유한 노드 그룹을 처음 유지 관리 기간에 등록할 때 매우 유용합니다.

할당량: 각 대상에 최대 5개까지 키-값 페어를 지정할 수 있습니다. 2개 이상의 키 값 페어를 지정하는 경우 대상 그룹에 포함되도록 지정한 모든 태그 키 및 값으로 노드에 태그를 지정해야 합니다.

> **Note**
>
> 태그-키 `Patch Group` 또는 `PatchGroup`으로 노드 그룹에 태그를 지정하고 노드에 `my-patch-group` 등의 공통 키 값을 할당할 수 있습니다. EC2 인스턴스 메타데이터에 태그를 허용한 경우 `PatchGroup`(공백 없음)을 사용해야 합니다. Systems Manager의 기능인 Patch Manager로 노드의 `Patch Group` 또는 `PatchGroup` 키를 평가하여 노드에 적용되는 패치 기준선을 결정할 수 있습니다. 태스크에서 `AWS-RunPatchBaseline` SSM 문서(또는 기존 `AWS-ApplyPatchBaseline` SSM 문서)를 실행하는 경우 대상을 유지 관리 기간에 등록할 때 동일한 `Patch Group` 또는 `PatchGroup` 키-값 페어를 지정할 수 있습니다. 예: `--target "Key=tag:PatchGroup,Values=`*`my-patch-group`*`"`. 이렇게 하면 유지 관리 기간을 사용하여 이미 동일한 패치 기준에 연결된 노드 그룹에 대한 패치를 업데이트할 수 있습니다. 자세한 내용은 패치 그룹 정보 단원을 참조하십시오.

예제 3: 태그 키 그룹(태그 값 없음)을 사용하여 대상 등록

로컬 시스템에서 다음 명령을 실행하여 키 값 여부와 상관없이 모두 하나 이상의 태그가 지정된 노드를 등록합니다. 각 *`example resource placeholder`*를 사용자의 정보로 바꿉니다.

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"
```

권장 용도: 단순한 단일 태그 키 또는 단일 태그 키 값 페어가 아닌 여러 태그 키(값 없음)를 지정하여 노드를 대상으로 지정하려는 경우 유용합니다.

할당량: 각 대상에 최대 5개까지 태그-키를 지정할 수 있습니다. 한 개 이상의 태그 키를 지정하는 경우, 대상 그룹에 포함되도록 지정한 모든 태그 키로 노드에 태그를 지정해야 합니다.

예제 4: 리소스 그룹 이름을 사용한 대상 등록

로컬 시스템에서 다음 명령을 실행하여 포함된 리소스 유형에 상관없이 지정된 리소스 그룹을 등록합니다. *mw-0c50858d01EXAMPLE*을 사용자의 정보로 바꿉니다. 유지 관리 기간에 할당할 태스크가 리소스 그룹에 추가한 리소스 유형에서 유효하지 않을 경우에는 시스템이 오류를 보고할 수 있습니다. 이러한 오류가 발생하더라도 리소스 유형이 지원되는 작업은 계속해서 실행됩니다.

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"
```

권장 용도: 유지 관리 기간에서 모든 리소스 유형의 대상 지정 여부를 평가하지 않고 리소스 그룹을 대상으로 빠르게 지정할 때, 혹은 리소스 그룹에 작업이 유효한 리소스 유형만 포함되어 있다는 사실을 알고 있을 때 유용합니다.

할당량: 대상으로 오직 하나의 리소스만 지정할 수 있습니다.

예제 5: 리소스 그룹의 리소스 유형을 필터링하여 대상 등록

로컬 시스템에서 다음 명령을 실행하여 지정한 리소스 그룹에 속해 있는 일부 리소스 유형만 등록합니다. *mw-0c50858d01EXAMPLE*을 사용자의 정보로 바꿉니다. 이 옵션을 사용하면 리소스 그룹에 속한 리소스 유형에 대해 작업을 추가하더라도 리소스 유형을 명시적으로 필터에 추가하지 않았다면 작업이 실행되지 않습니다.

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"
```

권장 용도: 유지 관리 기간에 작업을 실행할 수 있는 AWS 리소스 유형을 엄격하게 제어하려고 할 때, 혹은 리소스 그룹에 다수의 리소스 유형이 포함되어 있어서 유지 관리 기간 로그에서 불필요한 오류 보고서를 피하고 싶을 때 유용합니다.

할당량: 대상으로 오직 하나의 리소스만 지정할 수 있습니다.

3단계: 유지 관리 기간에 작업 등록(AWS CLI)

튜토리얼의 이 단계에서는 Linux용 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에서 `df` 명령을 실행하는 AWS Systems Manager Run Command 태스크를 등록합니다. 이 표준 Linux 명령의 결과는 사용 가능한 공간의 양과 인스턴스의 디스크 파일 시스템에서 사용되는 공간의 양을 표시합니다.

-또는-

예를 들어 Linux 대신 Windows Server용 Amazon EC2 인스턴스를 대상으로 지정하는 경우 다음 명령에서 df를 ipconfig로 바꿉니다. 이 명령의 출력에는 대상 인스턴스의 어댑터에 대한 IP 주소, 서브넷 마스크 및 기본 게이트웨이 세부 정보가 나열됩니다.

다른 태스크 유형을 등록할 준비가 되었거나 사용 가능한 Systems Manager Run Command 옵션 중 더 많은 옵션을 사용할 준비가 되면 예제: 유지 관리 기간에 작업 등록 섹션을 참조하세요. 여기서는 보다 광범위한 실제 시나리오를 계획하는 데 도움이 되는 네 가지 작업 유형 및 가장 중요한 옵션에 대한 자세한 정보를 제공합니다.

유지 관리 기간에 작업을 등록하려면

1. 로컬 시스템에서 다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다. 로컬 Windows 시스템에서 실행할 버전에는 명령줄 도구에서 명령을 실행하는 데 필요한 이스케이프 문자("/")가 포함됩니다.

   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\"]}}}
   ```

   시스템은 다음과 유사한 정보를 반환합니다.

   ```
   {
       "WindowTaskId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE"
   }
   ```

2. 이제 다음 명령을 실행하여 생성한 유지 관리 기간 작업에 대한 세부 정보를 봅니다.

   Linux & macOS

   ```
   aws ssm describe-maintenance-window-tasks \
   ```

```
    --window-id mw-0c50858d01EXAMPLE
```

Windows

```
aws ssm describe-maintenance-window-tasks ^
    --window-id mw-0c50858d01EXAMPLE
```

3. 시스템은 다음과 유사한 정보를 반환합니다.

```
{
    "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. 1단계: 유지 관리 기간 생성(AWS CLI) 단원에서 지정한 일정에 따라 작업을 실행할 시간이 될 때까지 기다립니다. 예를 들어 **--schedule "rate(5 minutes)"**를 지정한 경우 5분 동안 기다립니다. 기다린 후 다음 명령을 실행하여 이 작업에 대해 발생한 모든 실행에 대한 정보를 봅니다.

Linux & macOS

```
aws ssm describe-maintenance-window-executions \
    --window-id mw-0c50858d01EXAMPLE
```

Windows

```
aws ssm describe-maintenance-window-executions ^
    --window-id mw-0c50858d01EXAMPLE
```

시스템은 다음과 유사한 정보를 반환합니다.

```
{
    "WindowExecutions": [
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "WindowExecutionId": "14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE",
            "Status": "SUCCESS",
            "StartTime": 1557593493.096,
            "EndTime": 1557593498.611
        }
    ]
}
```

Tip

태스크가 성공적으로 실행되면 유지 관리 기간이 실행되는 속도를 줄일 수 있습니다. 예를 들어 빈도를 주 1회로 줄이려면 다음 명령을 실행합니다. *mw-0c50858d01EXAMPLE*을 사용자의 정보로 바꿉니다.

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)"
```

유지 관리 기간 일정 관리에 대한 자세한 내용은 참조: Systems Manager용 Cron 및 Rate 표현식 및 유지 관리 기간 예약 및 유효 기간 옵션 섹션을 참조하세요.
AWS Command Line Interface(AWS CLI)를 사용하여 유지 관리 기간을 수정하는 방법에 대한 자세한 내용은 자습서: 유지 관리 기간 업데이트(AWS CLI) 섹션을 참조하세요.

AWS CLI 명령을 실행하여 유지 관리 기간 작업 및 실행에 대한 자세한 내용을 보는 연습은 자습서: 작업 및 작업 실행에 대한 정보 보기(AWS CLI) 섹션을 참조하세요.

자습서 명령 출력 정보

유지 관리 기간 태스크 실행과 관련된 Run Command 명령의 출력을 보기 위해 AWS CLI를 사용하는 것은 이 튜토리얼에서 다루지 않습니다.

그러나 AWS CLI를 사용하여 이 데이터를 볼 수 있습니다. 명령 출력을 저장하는 유지 관리 기간을 구성한 경우 Systems Manager 콘솔 또는 Amazon Simple Storage Service(Amazon S3) 버킷에 저장된 로그 파일에서 출력을 볼 수도 있습니다. Linux용 EC2 인스턴스에서는 df 명령의 출력이 다음과 유사합니다.

```
Filesystem 1K-blocks Used Available Use% Mounted on

devtmpfs 485716 0 485716 0% /dev

tmpfs 503624 0 503624 0% /dev/shm

tmpfs 503624 328 503296 1% /run

tmpfs 503624 0 503624 0% /sys/fs/cgroup

/dev/xvda1 8376300 1464160 6912140 18% /
```

Windows Server용 EC2 인스턴스에 대한 ipconfig 명령의 출력은 다음과 유사합니다.

```
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  . :
```

예제: 유지 관리 기간에 작업 등록

유지 관리 기간에 태스크 등록에서 설명한 것처럼 AWS Command Line Interface(AWS CLI)를 사용하여 유지 관리 기간에 AWS Systems Manager의 기능인 Run Command의 태스크를 등록할 수 있습니다. 아래에 설명된 대로 Systems Manager Automation 워크플로용 태스크, AWS Lambda 함수 및 AWS Step Functions 태스크를 등록할 수도 있습니다.

> **Note**
>
> 유지 관리 기간 Run Command 유형 태스크에 대해 하나 이상의 대상을 지정합니다. 태스크에 따라 대상은 다른 유지 관리 기간 태스크 유형(Automation, AWS Lambda 및 AWS Step Functions)에 대해 옵션입니다. 대상을 지정하지 않는 태스크 실행에 대한 자세한 내용은 대상 없이 유지 관리 기간 태스크 등록 섹션을 참조하세요.

이 주제에서는 AWS Command Line Interface(AWS CLI) 명령 `register-task-with-maintenance-window`를 사용하여 4가지 지원되는 태스크 유형 각각을 유지 관리 기간에 등록하는 예제를 제공합니다. 예제는 데모용이지만 작업 등록 명령을 생성하도록 수정할 수 있습니다.

--cli-input-json 옵션 사용

작업 옵션을 보다 잘 관리하기 위해 JSON 파일에서 참조되는 옵션 값과 함께 명령 옵션 `--cli-input-json`을 사용할 수 있습니다.

다음 예제에서 제공하는 샘플 JSON 파일 콘텐츠를 사용하려면 로컬 시스템에서 다음을 수행하십시오.

1. `MyRunCommandTask.json`, `MyAutomationTask.json`과 같은 이름 또는 선호하는 다른 이름으로 파일을 생성합니다.
2. JSON 샘플의 콘텐츠를 파일에 복사합니다.
3. 작업 등록을 위해 파일 콘텐츠를 수정한 다음 파일을 저장합니다.
4. 파일을 저장한 디렉터리에서 다음 명령을 실행합니다. *`MyFile.json`*을 사용자의 파일 이름으로 바꿉니다.

   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
   ```

의사 파라미터 정보

일부 예제에서는 ID 정보를 작업에 전달하는 방법으로 의사 파라미터를 사용합니다. 예를 들어 `{{TARGET_ID}}` 및 `{{RESOURCE_ID}}`를 사용하여 AWS 리소스의 ID를 Automation, Lambda 및 Step Functions 태스크에 전달할 수 있습니다. `--task-invocation-parameters` 콘텐츠의 의사 파라미터에 대한 자세한 내용은 유지 관리 기간 작업 등록 시 의사 파라미터 사용 섹션을 참조하세요.

추가 정보

- register-task-with-maintenance-windows 옵션 정보.
- AWS CLI 명령 레퍼런스의 register-task-with-maintenance-window

- AWS Systems Manager API 참조의 RegisterTaskWithMaintenanceWindow

작업 등록 예제

다음 단원에서는 지원되는 작업 유형과, `--cli-input-json` 옵션과 함께 사용할 수 있는 JSON 샘플을 등록하기 위한 샘플 AWS CLI 명령을 제공합니다.

Systems Manager Run Command 태스크 등록

다음 예에서는 AWS CLI를 사용하여 유지 관리 기간에 Systems Manager Run Command 태스크를 등록하는 방법을 보여줍니다.

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\"]}}}"
```

`--cli-input-json` 파일 옵션과 함께 사용할 JSON 콘텐츠:

```
{
    "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": "DOC-EXAMPLE-BUCKET",
            "OutputS3KeyPrefix": "S3-PREFIX",
            "TimeoutSeconds": 3600
        }
    }
}
```

Systems Manager Automation 태스크 등록

다음 예에서는 AWS CLI를 사용하여 유지 관리 기간에 Systems Manager Automation 태스크를 등록하는 방법을 보여줍니다.

AWS CLI 명령:

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"
```

`--cli-input-json` 파일 옵션과 함께 사용할 JSON 콘텐츠:

```
{
    "WindowId": "mw-0c50858d01EXAMPLE",
        "TaskArn": "AWS-PatchInstanceWithRollback",
    "TaskType": "AUTOMATION","TaskInvocationParameters": {
        "Automation": {
            "DocumentVersion": "1",
            "Parameters": {
                "instanceId": [
                    "{{RESOURCE_ID}}"
                ]
            }
        }
    }
}
```

AWS Lambda 작업 등록

다음 예에서는 AWS CLI를 사용하여 유지 관리 기간에 Lambda 함수 태스크를 등록하는 방법을 보여 줍니다.

Lambda 함수를 생성한 사용자가 `SSMrestart-my-instances`라는 이름을 지정하고 `instanceId`와 `targetType`이라는 2개의 파라미터를 생성했습니다.

> **Important**
>
> Maintenance Windows의 IAM 정책에서는 Lambda 함수 이름(또는 별칭) 앞에 접두사 SSM을 추가해야 합니다. 이러한 유형의 태스크를 등록하기 전에 AWS Lambda에서 SSM을 포함하도록 이름을 업데이트합니다. 예를 들어 Lambda 함수 이름이 MyLambdaFunction인 경우 SSMMyLambdaFunction으로 변경합니다.

AWS CLI 명령:

Linux & macOS

> **Important**
>
> AWS CLI 버전 2를 사용하는 경우 Lambda 페이로드가 base64로 인코딩되지 않은 경우 다음 명령에 `--cli-binary-format raw-in-base64-out` 옵션을 포함해야 합니다. `cli_binary_format` 옵션은 버전 2에서만 사용할 수 있습니다. 이 파일 설정과 기타 AWS CLI `config` 파일 설정에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서의 지원되는 config 파일 설정을 참조하세요.

```
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**
>
> AWS CLI 버전 2를 사용하는 경우 Lambda 페이로드가 base64로 인코딩되지 않은 경우 다음 명령에 `--cli-binary-format raw-in-base64-out` 옵션을 포함해야 합니다. `cli_binary_format` 옵션은 버전 2에서만 사용할 수 있습니다. 이 파일 설정과 기타

AWS CLI config 파일 설정에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서의 지원되는 config 파일 설정을 참조하세요.

```
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\"}}'
```

--cli-input-json 파일 옵션과 함께 사용할 JSON 콘텐츠:

```
{
    "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
}
```

Step Functions 태스크 등록

다음 예에서는 AWS CLI를 사용하여 유지 관리 기간에 Step Functions 상태 머신 태스크를 등록하는 방법을 보여줍니다.

> **Note**
>
> 유지 관리 기간 태스크는 Step Functions 표준 상태 머신 워크플로만 지원합니다. 익스프레스 상태 머신 워크플로는 지원하지 않습니다. 상태 머신 워크플로 유형에 대한 자세한 내용은 AWS Step Functions개발자 안내서의 표준 워크플로와 익스프레스 워크플로를 참조하세요.

이러한 예의 경우 Step Functions 상태 머신을 생성한 사용자가 `instanceId` 파라미터가 있는 상태 시스템 `SSMMyStateMachine`을 생성했습니다.

> **Important**
>
> Maintenance Windows의 AWS Identity and Access Management(IAM) 정책에서는 Step Functions 상태 시스템 이름 앞에 접두사 `SSM`을 추가해야 합니다. 이러한 유형의 작업을 등록하기 전에 AWS Step Functions에서 `SSM`을 포함하도록 이름을 업데이트해야 합니다. 예를 들어 상태 시스템 이름이 `MyStateMachine`인 경우 `SSMMyStateMachine`으로 변경합니다.

AWS CLI 명령:

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"
```

`--cli-input-json` 파일 옵션과 함께 사용할 JSON 콘텐츠:

```
{
    "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 옵션 정보

register-task-with-maintenance-window 명령은 필요에 따라 작업을 구성하기 위한 몇 가지 옵션을 제공합니다. 일부는 필수 옵션이고 일부는 선택 옵션이며 일부는 단일 유지 관리 기간 작업 유형에만 적용됩니다.

이 주제에서는 이 자습서 단원의 샘플 작업에 도움이 되는 몇 가지 옵션에 대한 정보를 제공합니다. 모든 명령 옵션에 대한 자세한 내용은 AWS CLI Command Reference의 register-task-with-maintenance-window 섹션을 참조하세요.

`--task-arn` 옵션 정보

`--task-arn` 옵션은 태스크가 작업 중에 사용하는 리소스를 지정하는 데 사용됩니다. 지정하는 값은 다음 표에 설명된 대로 등록 중인 태스크 유형에 따라 달라집니다.

유지 관리 기간 작업을 위한 TaskArn 형식

유지 관리 기간 작업 유형	TaskArn 값
RUN_COMMAND 및 **AUTOMATION**	TaskArn은 SSM 문서 이름 또는 Amazon 리소스 이름(ARN)입니다. 예: `AWS-RunBatchShellScript` -또는- `arn:aws:ssm:` *`region`*`:11112222 3333:document/My-Document` .
LAMBDA	TaskArn은 함수 이름 또는 ARN입니다. 예: `SSMMy-Lambda-Function` -또는- `arn:aws:lambda:` *`region`*`:11112222 3333:function:SSMMyLambdaFu nction` .

유지 관리 기간 작업 유형	TaskArn 값
	Important Maintenance Windows의 IAM 정책에서는 Lambda 함수 이름(또는 별칭) 앞에 접두사 SSM을 추가해야 합니다. 이러한 유형의 태스크를 등록하기 전에 AWS Lambda에서 SSM을 포함하도록 이름을 업데이트합니다. 예를 들어 Lambda 함수 이름이 MyLambdaFunction 인 경우 SSMMyLambdaFunction 으로 변경합니다.
STEP_FUNCTIONS	TaskArn은 상태 시스템 ARN입니다. 예: arn:aws:states:us-east-2:111122223333:stateMachine:SSMMyStateMachine . **Important** 유지 관리 기간의 IAM 정책에서는 Step Functions 상태 시스템 이름 앞에 접두사 SSM을 추가해야 합니다. 이러한 유형의 작업을 등록하기 전에 AWS Step Functions에서 SSM을 포함하도록 이름을 업데이트해야 합니다. 예를 들어 상태 시스템 이름이 MyStateMachine 인 경우 SSMMyStateMachine 으로 변경합니다.

--service-role-arn 옵션 정보

유지 관리 기간 작업을 실행할 때 수임할 AWS Systems Manager 역할입니다.

자세한 내용은 Maintenance Windows 설정 단원을 참조하세요.

`--task-invocation-parameters` 옵션 정보

`--task-invocation-parameters` 옵션은 네 가지 작업 유형 각각에 고유한 파라미터를 지정하는 데 사용됩니다. 다음 표에서는 네 가지 작업 유형 각각에 대해 지원되는 파라미터를 설명합니다.

> **Note**
>
> {{TARGET_ID}} 등 `--task-invocation-parameters` 콘텐츠에서의 의사 파라미터 사용에 대한 자세한 내용은 유지 관리 기간 작업 등록 시 의사 파라미터 사용 섹션을 참조하세요.

유지 관리 기간 작업에 대한 작업 호출 파라미터 옵션

유지 관리 기간 작업 유형	사용 가능한 파라미터	예
RUN_COMMAND	설명 DocumentHash DocumentHashType NotificationConfig OutputS3BucketName OutputS3KeyPrefix 파라미터 ServiceRoleArn TimeoutSeconds	`"TaskInvocationParameters": { "RunCommand": { "Comment": "My Run Command task comment", "DocumentHash": "6554ed3d--truncated--5EXAMPLE", "DocumentHashType": "Sha256", "NotificationConfig": { "NotificationArn": "arn:aws:sns: region:123456789012:my-sns-topic-name", "NotificationEvents": [ "FAILURE" ],`

유지 관리 기간 작업 유형	사용 가능한 파라미터	예
		"NotificationType": "Invocation" }, "OutputS3BucketName": "DOC-EXAMPLE-BUCKET", "OutputS3KeyPrefix": " *S3-PREFIX* ", "Parameters": { "commands": ["Get-ChildItem$env:temp-Recurse|Remove-Item-Recurse-force"] }, "ServiceRoleArn": "arn:aws:iam::123456789012:role/MyMaintenanceWindowServiceRole", "TimeoutSeconds": 3600 } }

유지 관리 기간 작업 유형	사용 가능한 파라미터	예
자동화	DocumentVersion 파라미터	<pre>"TaskInvocationPar ameters": { "Automation": { "Document Version": "3", "Paramete rs": { "instanceid": ["{{TARGET_ID}}"] } } }</pre>
LAMBDA	ClientContext 페이로드 한정자	<pre>"TaskInvocationPar ameters": { "Lambda": { "ClientCo ntext": "ew0KICAi --truncated--0KIEX AMPLE", "Payload" : "{ \"targetId\": \"{{TARGET_ID}}\", \"targetType\": \"{{TARGET_TYPE}}\ " }", "Qualifie r": "$LATEST" } }</pre>

유지 관리 기간 작업 유형	사용 가능한 파라미터	예
STEP_FUNCTIONS	Input 명칭	<pre>"TaskInvocationParameters": { "StepFunctions": { "Input": "{ \"targetId\": \"{{TARGET_ID}}\" }", "Name": "{{INVOCATION_ID}}" } }</pre>

자습서: 유지 관리 기간에 대한 정보 보기(AWS CLI)

이 자습서에는 유지 관리 기간, 작업, 실행 및 호출에 대한 정보를 업데이트하거나 얻는 데 도움이 되는 명령이 포함되어 있습니다. 명령 옵션을 사용하여 보려는 세부 정보 유형을 필터링하는 방법을 보여주는 예제가 명령별로 정리되어 있습니다.

이 튜토리얼의 단계를 수행하면서 *###* 기울임꼴 텍스트의 값을 원하는 옵션 및 ID로 바꿉니다. 예를 들어 유지 관리 기간 ID *mw-0c50858d01EXAMPLE*과 인스턴스 ID *i-02573cafcfEXAMPLE*을 생성하는 리소스의 ID로 바꿉니다.

AWS Command Line Interface(AWS CLI) 설정 및 구성에 대한 자세한 내용은 AWS CLI 구성의 AWS CLI 설치, 업데이트 및 제거를 참조하세요.

명령 예제

- 'describe-maintenance-windows'에 대한 예제
- 'describe-maintenance-window-targets'에 대한 예제
- 'describe-maintenance-window-tasks'에 대한 예제
- 'describe-maintenance-windows-for-target'에 대한 예제
- 'describe-maintenance-window-executions'에 대한 예제
- 'describe-maintenance-window-schedule'에 대한 예제

'describe-maintenance-windows'에 대한 예제

AWS 계정에 모든 유지 관리 기간 나열

다음 명령을 실행합니다.

```
aws ssm describe-maintenance-windows
```

시스템은 다음과 같은 정보를 반환합니다.

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

활성화된 모든 유지 관리 기간 나열

다음 명령을 실행합니다.

```
aws ssm describe-maintenance-windows --filters "Key=Enabled,Values=true"
```

시스템은 다음과 같은 정보를 반환합니다.

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

비활성화된 모든 유지 관리 기간 나열

다음 명령을 실행합니다.

```
aws ssm describe-maintenance-windows --filters "Key=Enabled,Values=false"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "WindowIdentities": [
        {
            "WindowId": "mw-6e5c9d4b7cEXAMPLE",
            "Name": "My-Disabled-Maintenance-Window",
            "Enabled": false,
            "Duration": 2,
            "Cutoff": 1
        }
    ]
}
```

이름이 특정 접두사로 시작하는 모든 유지 관리 기간 나열

다음 명령을 실행합니다.

```
aws ssm describe-maintenance-windows --filters "Key=Name,Values=My"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "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'에 대한 예제

특정 소유자 정보 값과 일치하는 유지 관리 기간의 대상 표시

다음 명령을 실행합니다.

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**
>
> 지원되는 필터 키는 Type, WindowTargetId 및 OwnerInformation입니다.

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "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'에 대한 예제

SSM 명령 문서 **AWS-RunPowerShellScript**를 호출하는 모든 등록된 태스크 표시

다음 명령을 실행합니다.

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"
```

시스템은 다음과 같은 정보를 반환합니다.

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

우선순위가 “3”인 모든 등록된 작업 표시

다음 명령을 실행합니다.

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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
```

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

우선순위가 '1'인 모든 등록된 작업 표시 및 Run Command 사용

다음 명령을 실행합니다.

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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "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'에 대한 예제

특정 노드와 관련된 유지 관리 기간 대상 또는 작업에 대한 정보 나열

다음 명령을 실행합니다.

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
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "WindowIdentities": [
        {
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Name": "My-First-Maintenance-Window"
        },
        {
            "WindowId": "mw-9a8b7c6d5eEXAMPLE",
            "Name": "My-Second-Maintenance-Window"
        }
    ]
}
```

'describe-maintenance-window-executions'에 대한 예제

특정 날짜 이전에 실행된 모든 작업 나열

다음 명령을 실행합니다.

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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "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
        }
    ]
}
```

특정 날짜 이후에 실행된 모든 작업 나열

다음 명령을 실행합니다.

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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "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'에 대한 예제

특정 노드에 대해 예약된 다음 10개 유지 관리 기간 실행 표시

다음 명령을 실행합니다.

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
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "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"
}
```

특정 키 값 페어로 태그가 지정된 노드에 대한 유지 관리 기간 일정 표시

다음 명령을 실행합니다.

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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "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"
}
```

유지 관리 기간의 다음 4개 실행에 대한 시작 시간 표시

다음 명령을 실행합니다.

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"
```

시스템은 다음과 같은 정보를 반환합니다.

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

자습서: 작업 및 작업 실행에 대한 정보 보기(AWS CLI)

이 튜토리얼은 AWS Command Line Interface(AWS CLI)를 사용하여 완료된 유지 관리 기간 태스크에 대한 세부 정보를 보는 방법을 설명합니다.

자습서: 유지 관리 기간 생성 및 구성(AWS CLI)에서 바로 이어서 하는 경우 실행 결과를 보기 위해 유지 관리 기간이 적어도 한 번은 실행되도록 충분히 기다립니다.

이 튜토리얼의 단계를 수행하면서 *###* 기울임꼴 텍스트의 값을 원하는 옵션 및 ID로 바꿉니다. 예를 들어 유지 관리 기간 ID *mw-0c50858d01EXAMPLE*과 인스턴스 ID *i-02573cafcfEXAMPLE*을 생성하는 리소스의 ID로 바꿉니다.

작업 및 작업 실행에 대한 정보를 보려면(AWS CLI)

1. 다음 명령을 실행하여 특정 유지 관리 기간의 태스크 실행 목록을 확인합니다.

   Linux & macOS

   ```
   aws ssm describe-maintenance-window-executions \
       --window-id "mw-0c50858d01EXAMPLE"
   ```

   Windows

   ```
   aws ssm describe-maintenance-window-executions ^
       --window-id "mw-0c50858d01EXAMPLE"
   ```

   시스템은 다음과 같은 정보를 반환합니다.

```
{
    "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. 다음 명령을 실행하여 유지 관리 기간 태스크 실행에 대한 정보를 얻습니다.

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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "WindowExecutionId": "14bea65d-5ccc-462d-a2f3-e99c8EXAMPLE",
    "TaskIds": [
        "c9b05aba-197f-4d8d-be34-e73fbEXAMPLE"
    ],
    "Status": "SUCCESS",
    "StartTime": 1557593493.096,
    "EndTime": 1557593498.611
}
```

3. 다음 명령을 실행하여 유지 관리 기간 실행의 일부로 실행된 태스크를 나열합니다.

   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"
   ```

   시스템은 다음과 같은 정보를 반환합니다.

   ```
   {
       "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. 다음 명령을 실행하여 태스크 실행의 세부 정보를 파악합니다.

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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "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. 다음 명령을 실행하여 작업 실행을 위해 수행된 특정 작업 호출을 받습니다.

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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "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
        }
    ]
}
```

자습서: 유지 관리 기간 업데이트(AWS CLI)

이 튜토리얼은 AWS Command Line Interface(AWS CLI)를 사용하여 유지 관리 기간을 업데이트하는 방법을 설명합니다. 또한 AWS Systems Manager Run Command와 Automation, AWS Lambda 및 AWS Step Functions에 대한 태스크 유형을 포함하여 여러 태스크 유형을 업데이트하는 방법을 보여줍니다.

이 섹션의 예제에서는 다음 Systems Manager 작업을 사용하여 유지 관리 기간을 업데이트합니다.

- UpdateMaintenanceWindow
- UpdateMaintenanceWindowTarget
- UpdateMaintenanceWindowTask
- DeregisterTargetFromMaintenanceWindow

Systems Manager 콘솔을 사용하여 유지 관리 기간을 업데이트하는 방법에 대한 자세한 내용은 유지 관리 기간 리소스 업데이트 또는 삭제(콘솔) 섹션을 참조하세요.

이 튜토리얼의 단계를 수행하면서 *###* 기울임꼴 텍스트의 값을 원하는 옵션 및 ID로 바꿉니다. 예를 들어 유지 관리 기간 ID *mw-0c50858d01EXAMPLE*과 인스턴스 ID *i-02573cafcfEXAMPLE*을 생성하는 리소스의 ID로 바꿉니다.

유지 관리 기간을 업데이트하려면(AWS CLI)

1. AWS CLI를 열고 다음 명령을 실행하여 이름 및 설명을 포함하도록 대상을 업데이트합니다.

   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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "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. 다음 명령에 `replace` 옵션을 사용하여 실행하여 설명 필드를 제거하고 다른 대상을 추가합니다. 업데이트에 필드는 포함되지 않으므로(null 값) 설명 필드는 제거됩니다. Systems Manager에 사용하도록 구성된 추가 노드를 반드시 지정해야 합니다.

   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
   ```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "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 옵션을 사용하면 향후 지정한 날짜까지 유지 관리 기간의 활성화를 지연시킬 수 있습니다. end-date 옵션을 사용하면 유지 관리 기간이 더 이상 실행되지 않는 미래의 날짜 및 시간을 설정할 수 있습니다. 이 옵션은 ISO-8601 확장 형식으로 지정합니다.

   다음 명령을 실행하여 예약된 유지 관리 기간을 정기적으로 실행하는 날짜 및 시간 범위를 지정합니다.

   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. 다음 명령을 실행하여 Run Command 작업을 업데이트합니다.

> **Tip**
>
> 대상이 Windows Server용 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스인 경우 다음 명령에서 df를 ipconfig로, AWS-RunShellScript를 AWS-RunPowerShellScript로 변경합니다.

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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "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. 다음 명령을 변경 및 실행하여 Lambda 태스크를 업데이트합니다.

   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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "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. Step Functions 태스크를 업데이트하는 경우 다음 명령을 채택하고 실행하여 task-invocation-parameters를 업데이트합니다.

   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"
```

   시스템은 다음과 같은 정보를 반환합니다.

```
{
    "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. 다음 명령을 실행하여 유지 관리 기간에서 대상을 등록 취소합니다. 이 예에서는 `safe` 파라미터를 사용하여 대상이 다른 태스크에서 참조되고 따라서 안전하고 등록 취소할 수 있는지 확인합니다.

   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
   ```

시스템은 다음과 같은 정보를 반환합니다.

```
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. 다음 명령을 실행하여 작업에서 대상을 참조하는 경우에도 유지 관리 기간에서 대상을 등록 취소합니다. no-safe 파라미터를 사용하여 등록 취소 작업을 강제로 실행할 수 있습니다.

   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
   ```

   시스템은 다음과 같은 정보를 반환합니다.

   ```
   {
       "WindowId": "mw-0c50858d01EXAMPLE",
       "WindowTargetId": "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
   }
   ```

9. 다음 명령을 실행하여 Run Command 작업을 업데이트합니다. 이 예에서는 '{{ssm:UpdateLevel}}' 형식의 UpdateLevel이라는 Systems Manager Parameter Store 파라미터를 사용합니다.

   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}}'}}"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "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. 다음 명령을 실행하여 Automation 태스크를 업데이트하고 `task-invocation-parameters` 파라미터에 대한 `WINDOW_ID` 및 `WINDOW_TASK_ID` 파라미터를 지정합니다.

    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
    ```

    시스템은 다음과 같은 정보를 반환합니다.

    ```
    {
        "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"
}
```

자습서: 유지 관리 기간 삭제(AWS CLI)

이러한 튜토리얼에서 생성한 유지 관리 기간을 삭제하려면 다음 명령을 실행합니다.

```
aws ssm delete-maintenance-window --window-id "mw-0c50858d01EXAMPLE"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "WindowId":"mw-0c50858d01EXAMPLE"
}
```

유지 관리 기간 연습

이 섹션의 시연에서는 AWS Command Line Interface(AWS CLI) 또는 Systems Manager 콘솔을 사용하여 AWS Systems Manager 유지 관리 기간을 생성하는 방법을 보여줍니다. 생성된 유지 관리 기간이 관리형 노드에서 SSM Agent를 업데이트합니다.

내용

- 연습: SSM Agent를 업데이트할 유지 관리 기간 생성(AWS CLI)
- 연습: SSM Agent를 자동으로 업데이트할 유지 관리 기간 생성(콘솔)
- 패치에 대한 유지 관리 기간 생성(콘솔)

Systems Manager AWS CLI 참조에서도 샘플 명령을 볼 수 있습니다.

연습: SSM Agent를 업데이트할 유지 관리 기간 생성(AWS CLI)

다음 시연에서는 AWS Command Line Interface(AWS CLI)를 사용하여 AWS Systems Manager 유지 관리 기간을 생성하는 방법을 설명합니다. 또한 이 시연에서 관리형 노드를 대상으로 등록하고 SSM Agent를 업데이트할 Systems Manager Run Command 태스크를 등록하는 방법을 설명합니다.

시작하기 전 준비 사항

다음 절차를 수행하기 전에 구성할 노드에 대한 관리자 권한이 있거나 AWS Identity and Access Management(IAM)에서 적절한 권한을 부여받아야 합니다. 하이브리드 및 멀티클라우드 환경의 Systems Manager용으로 구성된 관리형 노드가 Linux 또는 Windows Server에서 하나 이상 실행되는지도 확인하세요. 자세한 내용은 AWS Systems Manager 설정 단원을 참조하십시오.

주제

- 1단계: 시작하기
- 2단계: 유지 관리 기간 생성
- 3단계: 유지 관리 기간 대상 등록(AWS CLI)
- 4단계: SSM Agent를 업데이트하도록 유지 관리 기간에 대한 Run Command 작업 등록

1단계: 시작하기

AWS CLI을 사용하여 명령을 실행하려면

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.

2. 노드가 유지 관리 기간의 대상으로 등록될 준비가 되었는지 확인합니다.

   다음 명령을 실행하여 어떤 노드가 온라인 상태인지 확인합니다.

   ```
   aws ssm describe-instance-information --query "InstanceInformationList[*]"
   ```

   다음 명령을 실행하여 특정 노드에 대한 세부 정보를 봅니다.

   ```
   aws ssm describe-instance-information --instance-information-filter-list
    key=InstanceIds,valueSet=instance-id
   ```

2단계: 유지 관리 기간 생성

다음 절차를 사용하여 유지 관리 기간을 생성하고 기본 옵션(예: 일정 및 기간)을 지정합니다.

유지 관리 기간 생성(AWS CLI)

1. AWS CLI를 열고 다음 명령을 실행하여 미국 태평양 시간대 기준 매주 일요일 오전 2시에 실행되는(1시간 차단) 유지 관리 기간을 생성합니다.

   Linux & macOS

   ```
   aws ssm create-maintenance-window \
       --name "My-First-Maintenance-Window" \
       --schedule "cron(0 2 ? * SUN *)" \
       --duration 2 \
       --schedule-timezone "America/Los_Angeles" \
       --cutoff 1 \
       --no-allow-unassociated-targets
   ```

   Windows

   ```
   aws ssm create-maintenance-window ^
       --name "My-First-Maintenance-Window" ^
       --schedule "cron(0 2 ? * SUN *)" ^
       --duration 2 ^
       --schedule-timezone "America/Los_Angeles" ^
       --cutoff 1 ^
   ```

```
    --no-allow-unassociated-targets
```

schedule 파라미터에 대한 cron 식을 생성하는 방법에 대한 자세한 내용은 참조: Systems Manager용 Cron 및 Rate 표현식 섹션을 참조하세요.

유지 관리 기간에 대한 다양한 스케줄 관련 옵션이 서로 관련되는 방법에 대한 자세한 내용은 유지 관리 기간 예약 및 유효 기간 옵션 섹션을 참조하세요.

--schedule 옵션 작업에 대한 자세한 내용은 참조: Systems Manager용 Cron 및 Rate 표현식 섹션을 참조하세요.

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "WindowId":"mw-0c50858d01EXAMPLE"
}
```

2. 이 기간을 포함하여 현재 AWS 리전의 AWS 계정에서 생성된 다른 모든 유지 관리 기간을 나열하려면 다음 명령을 실행합니다.

```
aws ssm describe-maintenance-windows
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "WindowIdentities": [
        {
            "Cutoff": 1,
            "Name": "My-First-Maintenance-Window",
            "NextExecutionTime": "2019-02-03T02:00-08:00",
            "Enabled": true,
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Duration": 2
        }
    ]
}
```

3단계: 유지 관리 기간 대상 등록(AWS CLI)

다음 절차를 사용하여 2단계에서 생성한 유지 관리 기간에 대상을 등록합니다. 대상을 등록한다는 것은 업데이트할 노드를 지정하는 것입니다.

유지 관리 기간 대상을 등록하려면(AWS CLI)

1. 다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   Linux & macOS

   ```
   aws ssm register-target-with-maintenance-window \
       --window-id "mw-0c50858d01EXAMPLE" \
       --target "Key=InstanceIds,Values=i-02573cafcfEXAMPLE" \
       --resource-type "INSTANCE"
   ```

   Windows

   ```
   aws ssm register-target-with-maintenance-window ^
       --window-id "mw-0c50858d01EXAMPLE" ^
       --target "Key=InstanceIds,Values=i-02573cafcfEXAMPLE" ^
       --resource-type "INSTANCE"
   ```

   시스템이 다음과 같은 정보를 반환합니다. 여기에 유지 관리 기간 대상 ID가 포함되어 있습니다. `WindowTargetId` 값을 복사하거나 적어둡니다. 이 유지 관리 기간에 대한 작업을 등록하려면 다음 단계에서 이 ID를 지정해야 합니다.

   ```
   {
      "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
   }
   ```

   다른 명령

   다음 명령을 사용하여 여러 관리형 노드를 등록합니다.

   Linux & macOS

   ```
   aws ssm register-target-with-maintenance-window \
       --window-id "mw-0c50858d01EXAMPLE" \
       --targets "Key=InstanceIds,Values=i-02573cafcfEXAMPLE,i-0471e04240EXAMPLE" \
   ```

```
    --resource-type "INSTANCE"
```

Windows

```
aws ssm register-target-with-maintenance-window ^
    --window-id "mw-0c50858d01EXAMPLE" ^
    --targets "Key=InstanceIds,Values=i-02573cafcfEXAMPLE,i-0471e04240EXAMPLE" ^
    --resource-type "INSTANCE"
```

다음 명령을 사용하여 태그를 지정해 노드를 등록합니다.

Linux & macOS

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-0c50858d01EXAMPLE" \
    --targets "Key=tag:Environment,Values=Prod" "Key=tag:Role,Values=Web" \
    --resource-type "INSTANCE"
```

Windows

```
aws ssm register-target-with-maintenance-window ^
    --window-id "mw-0c50858d01EXAMPLE" ^
    --targets "Key=tag:Environment,Values=Prod" "Key=tag:Role,Values=Web" ^
    --resource-type "INSTANCE"
```

2. 다음 명령을 실행하여 유지 관리 기간의 대상을 표시합니다.

```
aws ssm describe-maintenance-window-targets --window-id "mw-0c50858d01EXAMPLE"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "Targets": [
        {
            "ResourceType": "INSTANCE",
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Targets": [
                {
                    "Values": [
                        "i-02573cafcfEXAMPLE"
```

```
                    ],
                    "Key": "InstanceIds"
                }
            ],
            "WindowTargetId": "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
        },
        {
            "ResourceType": "INSTANCE",
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Targets": [
                {
                    "Values": [
                        "Prod"
                    ],
                    "Key": "tag:Environment"
                },
                {
                    "Values": [
                        "Web"
                    ],
                    "Key": "tag:Role"
                }
            ],
            "WindowTargetId": "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
        }
    ]
}
```

4단계: SSM Agent를 업데이트하도록 유지 관리 기간에 대한 Run Command 작업 등록

다음 절차를 사용하여 2단계에서 생성한 유지 관리 기간에 대한 Run Command 작업을 등록합니다. Run Command 작업은 등록된 대상에서 SSM Agent를 업데이트합니다.

SSM Agent를 업데이트하도록 유지 관리 기간에 대한 Run Command 작업을 등록하려면(AWS CLI)

1. 다음 명령을 통해 3단계의 `WindowTargetId` 값을 사용하여 유지 관리 기간에 대한 Run Command 태스크를 등록합니다. 각 *`example resource placeholder`*를 사용자의 정보로 바꿉니다. 이 작업은 `AWS-UpdateSSMAgent` 문서를 사용하여 SSM Agent를 업데이트합니다.

   Linux & macOS

   ```
   aws ssm register-task-with-maintenance-window \
   ```

```
    --window-id "mw-0c50858d01EXAMPLE" \
    --task-arn "AWS-UpdateSSMAgent" \
    --name "UpdateSSMAgent" \
    --targets "Key=WindowTargetIds,Values=e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
 \
    --service-role-arn "arn:aws:iam:account-id:role/MW-Role" \
    --task-type "RUN_COMMAND" \
    --max-concurrency 1 --max-errors 1 --priority 10
```

Windows

```
aws ssm register-task-with-maintenance-window ^
    --window-id "mw-0c50858d01EXAMPLE" ^
    --task-arn "AWS-UpdateSSMAgent" ^
    --name "UpdateSSMAgent" ^
    --targets "Key=WindowTargetIds,Values=e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
 ^
    --service-role-arn "arn:aws:iam:account-id:role/MW-Role" ^
    --task-type "RUN_COMMAND" ^
    --max-concurrency 1 --max-errors 1 --priority 10
```

> **Note**
>
> 이전 단계에서 등록한 대상이 Windows Server 2012 R2 이하인 경우 AWS-UpdateEC2Config 문서를 사용해야 합니다.

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "WindowTaskId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE"
}
```

2. 다음 명령을 실행하여 유지 관리 기간의 등록된 모든 작업을 나열합니다.

```
aws ssm describe-maintenance-window-tasks --window-id "mw-0c50858d01EXAMPLE"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "Tasks": [
        {
            "ServiceRoleArn": "arn:aws:iam::111122223333:role/MW-Role",
            "MaxErrors": "1",
            "TaskArn": "AWS-UpdateSSMAgent",
            "MaxConcurrency": "1",
            "WindowTaskId": "4f7ca192-7e9a-40fe-9192-5cb15EXAMPLE",
            "TaskParameters": {},
            "Priority": 10,
            "WindowId": "mw-0c50858d01EXAMPLE",
            "Type": "RUN_COMMAND",
            "Targets": [
                {
                    "Values": [
                        "e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE"
                    ],
                    "Key": "WindowTargetIds"
                }
            ],
            "Name": "UpdateSSMAgent"
        }
    ]
}
```

연습: SSM Agent를 자동으로 업데이트할 유지 관리 기간 생성(콘솔)

다음 시연에서는 AWS Systems Manager 콘솔을 사용하여 유지 관리 기간을 생성하는 방법을 설명합니다. 또한 이 시연에서 관리형 노드를 대상으로 등록하고 SSM Agent를 업데이트할 Systems Manager Run Command 태스크를 등록하는 방법을 설명합니다.

시작하기 전 준비 사항

다음 절차를 수행하기 전에 구성할 노드에 대한 관리자 권한이 있거나 AWS Identity and Access Management(IAM)에서 적절한 권한을 부여받아야 합니다. Systems Manager용으로 구성된 관리형 노드가 하이브리드 및 멀티클라우드 환경의 Linux 또는 Windows Server에서 하나 이상 실행되는지도 확인하세요. 자세한 내용은 AWS Systems Manager 설정 단원을 참조하십시오.

주제

- 1단계: 유지 관리 기간 생성(콘솔)

- 2단계: 유지 관리 기간 대상 등록(콘솔)
- 3단계: SSM Agent를 업데이트하도록 유지 관리 기간에 대한 Run Command 작업 등록(콘솔)

1단계: 유지 관리 기간 생성(콘솔)

유지 관리 기간을 생성하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Maintenance Windows를 선택합니다.
3. 유지 관리 기간 생성을 선택합니다.
4. 이름(Name)에 이 유지 관리 기간을 식별하는 데 유용한 설명 이름을 입력합니다.
5. (선택 사항) 설명에 설명을 입력합니다.
6. 관리형 노드가 대상으로 등록되어 있지 않은 경우에도 관리형 노드에 대해 유지 관리 기간 태스크를 실행하도록 허용하려면 등록되지 않은 대상 허용(Allow unregistered targets)을 선택합니다. 이 옵션을 선택한 경우 작업을 유지 관리 기간에 등록할 때 등록되지 않은 노드를 노드 ID를 사용하여 선택할 수 있습니다.

   이 옵션을 선택하지 않은 경우 작업을 유지 관리 기간에 등록할 때 이전에 등록된 대상을 선택해야 합니다.
7. 세 가지 예약 옵션 중 하나를 사용하여 유지 관리 기간에 대한 일정을 지정합니다.

   cron/rate 표현식 작성에 대한 자세한 내용은 참조: Systems Manager용 Cron 및 Rate 표현식 섹션을 참조하세요.
8. 기간에 유지 관리 기간이 실행되어야 하는 시간을 입력합니다.
9. 작업 개시 중지에 유지 관리 기간 종료 이전에 시스템에서 실행할 새 작업 예약을 중지해야 하는 시간을 입력합니다.
10. (옵션) [기간 시작일 - 옵션(Window start date - optional)]에 유지 관리 기간을 활성화하려는 날짜 및 시간을 ISO-8601 확장 형식으로 지정합니다. 이를 사용하면 향후 지정한 날짜까지 유지 관리 기간의 정품 인증을 지연시킬 수 있습니다.

> **Note**
>
> 과거의 시작 날짜 및 시간을 지정할 수 없습니다.

11. (옵션) [기간 종료일 - 옵션(Window end date - optional)]에 유지 관리 기간을 비활성화하려는 날짜 및 시간을 ISO-8601 확장 형식으로 지정합니다. 이를 사용하면 유지 관리 기간이 더 이상 실행되지 않는 미래의 날짜 및 시간을 설정할 수 있습니다.
12. (옵션) [예약 시간대 - 옵션(Schedule time zone - optional)]에 예약된 유지 관리 기간 실행의 기준이 될 시간대를 IANA(Internet Assigned Numbers Authority) 형식으로 지정합니다. 예를 들어 "America/Los_Angeles", "etc/UTC" 또는 "Asia/Seoul"입니다.

    올바른 형식에 대한 자세한 내용은 IANA 웹 사이트에서 표준 시간대 데이터베이스를 참조하십시오.
13. (선택 사항) 태그 관리 영역에서 하나 이상의 태그 키 이름/값 페어를 유지 관리 기간에 적용합니다.

    태그는 리소스에 할당하는 선택적 메타데이터입니다. 태그를 사용하면 용도, 소유자 또는 환경을 기준으로 하는 등 리소스를 다양한 방식으로 분류할 수 있습니다. 예를 들어 유지 관리 기간에 태그를 지정하여 이 기간이 실행하는 작업 유형, 대상 유형, 이 기간이 실행되는 환경을 식별하려는 경우 다음 키 이름/값 페어를 지정할 수 있습니다.

    - `Key=TaskType,Value=AgentUpdate`
    - `Key=OS,Value=Windows`
    - `Key=Environment,Value=Production`
14. 유지 관리 기간 생성을 선택합니다. 그러면 유지 관리 기간 페이지로 돌아갑니다. 방금 생성한 유지 관리 기간은 [(사용)Enabled] 상태입니다.

2단계: 유지 관리 기간 대상 등록(콘솔)

다음 절차를 사용하여 1단계에서 생성한 유지 관리 기간에 대상을 등록합니다. 대상을 등록한다는 것은 업데이트할 노드를 지정하는 것입니다.

유지 관리 기간에 대상을 할당하려면(콘솔)

1. 유지 관리 기간 목록에서 방금 생성한 유지 관리 기간을 선택합니다.
2. 작업, 대상 등록을 차례로 선택합니다.
3. (옵션) [대상 이름(Target name)]에 대상의 이름을 입력합니다.
4. (선택 사항) 설명에 설명을 입력합니다.

5. (선택 사항) 소유자 정보에 이름 또는 작업 별칭을 지정합니다. 소유자 정보는 이 유지 관리 기간 내에 이러한 대상에 대해 태스크를 실행하는 동안 발생한 모든 Amazon EventBridge 이벤트에 포함됩니다.

   EventBridge를 사용하여 Systems Manager 이벤트 모니터링에 대한 자세한 내용은 Amazon EventBridge로 Systems Manager 이벤트 모니터링 섹션을 참조하세요.

6. 대상 영역에서 아래 표에 나열된 옵션 중 하나를 선택합니다.

옵션	설명
인스턴스 태그 지정	인스턴스 태그 지정(Specify instance tags)에 태그 키 1개 이상과 계정의 관리형 노드에 추가했거나 앞으로 추가할 태그 값(옵션)을 지정합니다. 유지 관리 기간이 실행되면 태그가 추가된 모든 관리형 노드에서 작업을 실행하려고 합니다. 두 개 이상의 태그 키를 지정하는 경우, 대상 그룹에 포함되도록 지정한 모든 태그 키 및 값으로 노드에 태그를 지정해야 합니다.
수동으로 노드 선택	목록에서 유지 관리 기간 대상에 추가할 노드마다 확인란을 선택합니다. 계정에서 Systems Manager와 함께 사용하도록 구성된 노드가 모두 목록에 포함됩니다. 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요. 엣지 디바이스, 온프레미스 서버 및 가상 머신에 대한 자세한 내용은 하이브리드 및 멀티클라우드 환경에서 Systems Manager 사용하기 섹션을 참조하세요.

옵션	설명
리소스 그룹 선택	리소스 그룹(Resource group)에 있는 목록에서 계정의 기존 리소스 그룹 이름을 선택합니다. 리소스 그룹 생성 및 작업에 대한 자세한 내용은 다음 주제를 참조하십시오. • AWS Resource Groups 사용 설명서의 리소스 그룹이란 무엇인가요? • AWS 뉴스 블로그의 AWS에 대한 리소스 그룹 및 태깅 리소스 유형(Resource types)에서 사용 가능한 리소스 유형을 최대 5개까지 선택하거나, 모든 리소스 유형을 선택합니다. 유지 관리 기간에 할당할 태스크가 대상에 추가한 리소스 유형 중 하나에서 유효하지 않을 경우에는 시스템이 오류를 보고할 수 있습니다. 이러한 오류가 발생하더라도 리소스 유형이 지원되는 작업은 계속해서 실행됩니다. 예를 들어 다음과 같은 리소스 유형을 대상에 추가한다고 가정하겠습니다. • `AWS::S3::Bucket` • `AWS::DynamoDB::Table` • `AWS::EC2::Instance` 하지만 나중에 작업을 유지 관리 기간에 추가하면서 패치 기준을 적용하거나 노드를 다시 부팅하는 등 추가하려는 작업을 노드 관련 작업으로 제한하려고 합니다. 그러면 유지 관리 기간 로그에서 Amazon Simple Storage Service(Amazon S3) 버킷 또는 Amazon

옵션	설명
	DynamoDB 테이블을 찾을 수 없다는 오류가 보고될 수 있습니다. 하지만 유지 관리 기간에서는 리소스 그룹의 노드에 대한 작업이 계속해서 실행됩니다.

7. 대상 등록을 선택합니다.

3단계: SSM Agent를 업데이트하도록 유지 관리 기간에 대한 Run Command 작업 등록(콘솔)

다음 절차를 사용하여 1단계에서 생성한 유지 관리 기간에 대한 Run Command 작업을 등록합니다. Run Command 작업은 등록된 대상에서 SSM Agent를 업데이트합니다.

유지 관리 기간에 작업을 할당하려면(콘솔)

1. 유지 관리 기간 목록에서 방금 생성한 유지 관리 기간을 선택합니다.
2. 작업(Actions), Run command 태스크 등록(Register Run command task)을 차례로 선택합니다.
3. (옵션) [이름(Name)]에 태스크의 이름을 입력합니다(예: UpdateSSMAgent).
4. (선택 사항) 설명에 설명을 입력합니다.
5. [Command 문서(Command document)] 영역에서 SSM Command 문서 `AWS-UpdateSSMAgent`를 선택합니다.

   > **Note**
   >
   > 이전 단계에서 등록한 대상이 Windows Server 2012 R2 이하인 경우 `AWS-UpdateEC2Config` 문서를 사용해야 합니다.

6. 문서 버전에서 사용할 문서 버전을 선택합니다.
7. 작업 우선순위에서 이 작업의 우선순위를 지정합니다. 가장 높은 우선순위는 `0`입니다. 유지 관리 기간의 작업은 우선순위에 따라 예약되며, 같은 우선순위의 작업은 동시에 예약됩니다.
8. 대상(Targets) 섹션에서 등록된 대상 그룹 선택(Selecting registered target groups) 또는 등록되지 않은 대상 선택(Selecting unregistered targets)을 선택하여 이 작업을 실행할 노드를 식별합니다.
9. Rate control(속도 제어)에서

   - Concurrency(동시성)에서 명령을 동시에 실행할 관리형 노드의 백분율 또는 개수를 지정합니다.

> **Note**
>
> 관리형 노드에 적용할 태그를 지정하거나, AWS 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상 수를 제한합니다.

- Error threshold(오류 임계값)에서, 명령이 노드의 개수 또는 백분율에서 실패한 후 다른 관리형 노드에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째 오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.

10. (선택 사항) IAM 서비스 역할의 경우 유지 관리 기간 작업을 실행할 때 Systems Manager에서 수임할 수 있는 권한을 제공할 역할을 선택합니다.

    서비스 역할 ARN을 지정하지 않으면 Systems Manager에서 계정의 서비스 연결 역할을 사용합니다. Systems Manager에 대한 적절한 서비스 연결 역할이 계정에 없는 경우 작업이 등록될 때 해당 역할이 생성됩니다.

    > **Note**
    >
    > 보안 태세를 개선하려면 유지 관리 기간 작업을 실행하기 위한 사용자 지정 정책 및 사용자 지정 서비스 역할을 생성하는 것이 좋습니다. 특정 유지 관리 기간 작업에 필요한 권한만 제공하도록 정책을 작성할 수 있습니다. 자세한 내용은 콘솔을 사용하여 유지 관리 기간에 대한 권한 구성 단원을 참조하십시오.

11. (선택 사항) 출력 옵션(Output options)에서 다음 중 하나를 수행합니다.

    - S3에 쓰기 사용(Enable writing to S3) 확인란을 선택하여 파일에 명령 출력을 저장합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

      > **Note**
      >
      > 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 태스크를 수행하는 사용자의 권한이 아닌 노드에 할당된 인스턴스 프로파일의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 노드와 연결된 인스턴스 프로파일은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

- CloudWatch 출력(CloudWatch output) 확인란을 선택하여 Amazon CloudWatch Logs에 전체 출력을 씁니다. CloudWatch Logs 로그 그룹의 이름을 입력합니다.

12. [SNS 알림(SNS notifications)] 섹션에서 Systems Manager에서 Amazon Simple Notification Service(Amazon SNS)를 사용하여 명령 상태에 대한 알림을 보내도록 허용할 수 있습니다. 이 옵션을 설정하는 경우 다음을 지정해야 합니다.

    a. Amazon SNS 알림을 시작하는 IAM 역할입니다.

    b. 사용할 Amazon SNS 주제입니다.

    c. 알림을 받을 특정 이벤트 유형

    d. 명령 상태가 변경될 때 받을 알림 유형 여러 노드로 전송된 명령의 경우 각 호출 상태가 변경될 때 호출(노드 당)을 기준으로 알림을 받을 호출(Invocation)을 선택합니다.

13. 파라미터(Parameters) 섹션에서 선택적으로 설치할 SSM Agent의 특정 버전을 지정하거나 SSM Agent 서비스를 이전 버전으로 다운그레이드할 수 있습니다. 하지만 이 연습에서 특정 버전을 제공하지는 않습니다. 그러므로 SSM Agent가 최신 버전으로 업데이트됩니다.

14. [Run command 태스크 등록(Register Run command task)]을 선택합니다.

패치에 대한 유지 관리 기간 생성(콘솔)

> **Important**
>
> 이 레거시 주제를 계속 사용하여 패치 적용을 위한 유지 관리 기간을 생성할 수 있습니다. 하지만 버킷 정책을 사용하는 대신 것이 좋습니다. 자세한 내용은 Quick Setup 패치 정책 사용 및 Patch Manager 조직 패치 적용 구성 단원을 참조하세요.

서버 가용성에 미치는 영향을 최소화하려면 비즈니스 운영을 방해하지 않는 시간대에 패치를 실행할 수 있도록 유지 관리 기간을 구성하는 것이 좋습니다. 유지 관리 기간에 대한 자세한 내용은 AWS Systems Manager Maintenance Windows 섹션을 참조하세요.

이 절차를 시작하기 전에 AWS Systems Manager의 기능인 Maintenance Windows에 대한 역할과 권한을 구성해야 합니다. 자세한 내용은 Maintenance Windows 설정 단원을 참조하십시오.

패치 적용에 대한 유지 관리 기간을 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Maintenance Windows를 선택합니다.

3. 유지 관리 기간 생성을 선택합니다.
4. 이름 필드에서 필수 및 중요 업데이트에 패치를 적용하는 데 대한 유지 관리 기간으로 지정하는 이름을 입력합니다.
5. 설명에 설명을 입력합니다.
6. 관리형 노드가 대상으로 등록되어 있지 않은 경우에도 관리형 노드에 대해 유지 관리 기간 태스크를 실행하도록 허용하려면 등록되지 않은 대상 허용(Allow unregistered targets)을 선택합니다. 이 옵션을 선택한 경우 작업을 유지 관리 기간에 등록할 때 등록되지 않은 노드를 노드 ID를 사용하여 선택할 수 있습니다.

   이 옵션을 선택하지 않은 경우 작업을 유지 관리 기간에 등록할 때 이전에 등록된 대상을 선택해야 합니다.
7. 일정 섹션의 맨 위에서 세 가지 일정 옵션 중 하나를 사용하여 유지 관리 기간 일정을 지정합니다.

   cron/rate 표현식 작성에 대한 자세한 내용은 참조: Systems Manager용 Cron 및 Rate 표현식 섹션을 참조하세요.
8. 기간에 유지 관리 기간을 실행할 시간을 입력합니다. 지정된 값은 유지 관리가 시작하는 시간을 기준으로 유지 관리 기간의 특정 종료 시간을 결정합니다. 다음 단계에서 해당 종료 시간에서 작업 개시 중지에 지정된 시간을 뺀 시간 이후부터는 유지 관리 기간 작업을 시작할 수 없습니다.

   예를 들어, 유지 관리 기간이 오후 3시에 시작되고, 기간이 3시간으로 설정되고, 작업 개시 중지 값이 1시간인 경우, 오후 5시 이후부터는 유지 관리 기간 작업을 시작할 수 없습니다.
9. 작업 개시 중지에 유지 관리 기간 종료 이전에 시스템에서 실행할 새 작업 예약을 중지해야 하는 시간을 입력합니다.
10. (선택 사항) Start date (optional)(시작일(선택 사항))에 유지 관리 기간을 활성화하려는 날짜 및 시간을 ISO-8601 확장 형식으로 지정합니다. 이를 사용하면 향후 지정한 날짜까지 유지 관리 기간의 정품 인증을 지연시킬 수 있습니다.
11. (선택 사항) End date (optional)(종료일(선택 사항))에 유지 관리 기간을 비활성화하려는 날짜 및 시간을 ISO-8601 확장 형식으로 지정합니다. 이를 사용하면 유지 관리 기간이 더 이상 실행되지 않는 미래의 날짜 및 시간을 설정할 수 있습니다.
12. (선택 사항) 표준 시간대(선택 사항)에 예약된 유지 관리 기간 실행의 기준이 될 표준 시간대를 IANA(Internet Assigned Numbers Authority) 형식으로 지정합니다. 예를 들어 "America/Los_Angeles", "etc/UTC" 또는 "Asia/Seoul"입니다.

    올바른 형식에 대한 자세한 내용은 IANA 웹 사이트에서 표준 시간대 데이터베이스를 참조하십시오.
13. 유지 관리 기간 생성을 선택합니다.

14. 유지 관리 기간 목록에서 방금 만든 유지 관리 기간을 선택한 다음 작업, 대상 등록을 차례로 선택합니다.
15. (선택 사항) 유지 관리 기간 대상 세부 정보(Maintenance window target details) 섹션에서, 이 대상에 대한 이름, 설명 및 소유자 정보(이름이나 별칭)를 제공합니다.
16. 대상에서 Specifying instance tags(인스턴스 태그 지정)를 선택합니다.
17. Instance tags(인스턴스 태그)에 키와 태그 값을 입력하여 유지 관리 기간에 등록할 노드를 식별한 다음, 추가를 선택합니다.
18. 대상 등록을 선택합니다. 시스템에서 유지 관리 기간 대상이 생성됩니다.
19. 생성한 유지 관리 기간의 세부 정보 페이지에서 Actions(작업), Register Run command task(실행 명령 작업 등록)을 차례대로 선택합니다.
20. (선택 사항) Maintenance window task details(유지 관리 기간 작업 세부 정보)에 이 작업의 이름 및 설명을 제공합니다.
21. 명령 문서(Command document에서 `AWS-RunPatchBaseline`를 선택합니다.
22. 작업 우선순위(Task priority)에서 우선순위를 선택합니다. 가장 높은 우선순위는 `0`입니다.
23. 대상(Target)의 다음을 기준으로 대상 지정(Target by)에서, 이 절차의 앞부분에서 생성한 유지 관리 기간 대상을 선택합니다.
24. Rate control(속도 제어)에서

    - Concurrency(동시성)에서 명령을 동시에 실행할 관리형 노드의 백분율 또는 개수를 지정합니다.

      > **Note**
      >
      > 관리형 노드에 적용할 태그를 지정하거나, AWS 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상 수를 제한합니다.

    - Error threshold(오류 임계값)에서, 명령이 노드의 개수 또는 백분율에서 실패한 후 다른 관리형 노드에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째 오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.
25. (선택 사항) IAM 서비스 역할의 경우 유지 관리 기간 작업을 실행할 때 Systems Manager에서 수임할 수 있는 권한을 제공할 역할을 선택합니다.

서비스 역할 ARN을 지정하지 않으면 Systems Manager에서 계정의 서비스 연결 역할을 사용합니다. Systems Manager에 대한 적절한 서비스 연결 역할이 계정에 없는 경우 작업이 등록될 때 해당 역할이 생성됩니다.

> **Note**
>
> 보안 태세를 개선하려면 유지 관리 기간 작업을 실행하기 위한 사용자 지정 정책 및 사용자 지정 서비스 역할을 생성하는 것이 좋습니다. 특정 유지 관리 기간 작업에 필요한 권한만 제공하도록 정책을 작성할 수 있습니다. 자세한 내용은 콘솔을 사용하여 유지 관리 기간에 대한 권한 구성 단원을 참조하십시오.

26. (선택 사항) 출력 옵션에서 명령 출력을 파일에 저장하려면 S3 버킷에 쓰기 활성화 옆의 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

> **Note**
>
> 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아닌 관리형 노드에 할당된 인스턴스 프로파일의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

출력을 Amazon CloudWatch Logs 로그 그룹으로 스트리밍하려면 [CloudWatch 출력(CloudWatch output)] 상자를 선택합니다. 상자에 로그 그룹 이름을 입력합니다.

27. SNS notifications(SNS 알림) 섹션에서, 명령 실행 상태에 대한 알림이 전송되도록 하려면 Enable SNS notifications(SNS 알림 활성화) 확인란을 선택합니다.

    Run Command에 대한 Amazon SNS 알림 구성에 대한 자세한 내용은 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 섹션을 참조하세요.

28. 파라미터:

    - 작업에서 스캔을 선택하여 누락된 패치를 스캔하거나 설치를 선택하여 누락된 패치를 스캔 및 설치합니다.
    - Snapshot Id 필드에서 아무것도 입력할 필요가 없습니다. 이 시스템은 이 파라미터를 자동으로 생성하고 제공합니다.

- Patch Manager가 패치 기준에 지정된 것과 다른 패치 집합을 사용하도록 하려는 경우가 아니라면 [설치 재정의 목록(Install Override List)] 필드에 아무 것도 입력할 필요가 없습니다. 자세한 설명은 파라미터 이름: `InstallOverrideList`을 참조하세요.
- 재부팅 옵션(Reboot option)에서 `Install` 작업 중에 패치가 설치된 경우 노드를 재부팅할 것인지, 아니면 Patch Manager가 마지막 노드 재부팅 이후에 설치된 다른 패치를 검색할 것인지 지정합니다. 자세한 설명은 파라미터 이름: `RebootOption`을 참조하세요.
- (선택 사항) Comment에 이 명령에 대한 추적 정보 또는 알림 사항을 입력합니다.
- Timeout (seconds)에 시스템이 작업이 완료될 때까지 기다려야 하는 시간(초)을 입력합니다. 이 시간이 지나면 작업이 실패한 것으로 간주됩니다.

29. 실행 명령 작업 등록(Register run command task)를 선택합니다.

유지 관리 기간 태스크가 완료된 후 Systems Manager 콘솔의 관리형 인스턴스(Managed Instances) 페이지에서 패치 규정 준수 세부 정보를 볼 수 있습니다. 필터 막대에서 `AWS:PatchSummary` 및 `AWS:PatchCompliance` 필터를 사용합니다.

> **Note**
>
> 필터를 지정한 후 URL을 북마크하여 쿼리를 저장할 수 있습니다.

또한 관리형 인스턴스 페이지에서 특정 노드를 선택하여 해당 노드를 드릴다운한 다음 패치 탭을 선택할 수 있습니다. DescribePatchGroupState 및 DescribeInstancePatchStatesForPatchGroup API를 사용하여 규정 준수 세부 정보를 확인할 수도 있습니다. 패치 규정 준수 데이터에 대한 자세한 내용은 패치 규정 준수 정보 섹션을 참조하세요.

유지 관리 기간을 사용하는 패치 일정 정보

패치 기준선(및 선택적으로 패치 그룹)을 구성한 후 유지 관리 기간을 사용하여 패치를 노드에 적용할 수 있습니다. 유지 관리 기간은 비즈니스 운영을 방해하지 않는 범위 내에서 패치 프로세스 시간을 지정함으로써 서버 가용성에 미치는 영향을 최소화할 수 있습니다. 유지 관리 기간은 다음과 같이 실행됩니다.

1. 패치 적용 작업에 대한 일정에 따라 유지 관리 기간을 생성합니다.
2. 태그 이름에 `Patch Group` 또는 `PatchGroup` 태그를 지정하고 Amazon Elastic Compute Cloud(Amazon EC2) 태그를 정의한 값(예: '웹 서버' 또는 'US-EAST-PROD')을 지정하여 유지 관리 기간에 대한 대상을 선택합니다. EC2 인스턴스 메타데이터에 태그를 허용한 경우 `PatchGroup`(공백 없음)을 사용해야 합니다.

3. 새로운 유지 관리 기간 태스크를 생성하고 AWS-RunPatchBaseline 문서를 지정합니다.

작업을 구성할 때 노드를 스캔할 것인지 또는 노드에서 패치를 스캔하여 설치할 것인지 선택할 수 있습니다. 노드 스캔을 선택하는 경우 AWS Systems Manager의 기능인 Patch Manager는 이제 각 노드를 스캔하고 검토해야 할 누락된 패치 목록을 생성합니다.

패치를 스캔 및 설치하기로 선택하면 Patch Manager가 각 노드를 스캔하고 설치된 패치의 목록을 기준의 승인된 패치 목록과 비교합니다. Patch Manager는 누락된 패치를 식별한 다음 누락된 패치와 승인된 패치를 모두 다운로드하고 설치합니다.

한 번 검사하거나 설치하여 문제를 해결하려면 Run Command를 사용하여 AWS-RunPatchBaseline 문서를 직접 호출할 수 있습니다.

> **Important**
>
> 패치 설치를 마치면 Systems Manager가 각 노드를 재부팅합니다. 재부팅은 패치가 올바르게 설치되고 시스템이 노드를 잠재적으로 잘못된 상태로 두진 않았는지 확인하기 위해 필요합니다. (예외: AWS-RunPatchBaseline 문서에서 RebootOption 파라미터가 NoReboot로 설정되어 있으면 Patch Manager를 실행한 후 관리형 노드가 재부팅되지 않습니다. 자세한 내용은 파라미터 이름: RebootOption 섹션을 참조하세요.)

유지 관리 기간 작업 등록 시 의사 파라미터 사용

AWS Systems Manager의 기능인 Maintenance Windows에 작업을 등록할 때 4가지 각 작업 유형에 고유한 파라미터를 지정합니다. (CLI 명령에서는 --task-invocation-parameters 옵션을 사용하여 제공됩니다.)

{{RESOURCE_ID}}, {{TARGET_TYPE}}, {{WINDOW_TARGET_ID}} 등의 의사 파라미터 구문을 사용하여 특정 값을 참조할 수도 있습니다. 유지 관리 기간 작업이 실행되면 의사 파라미터 자리표시자 대신 올바른 값을 전달합니다. 사용할 수 있는 의사 파라미터의 전체 목록은 이 주제 후반부의 지원되는 가상 파라미터 섹션에 제공됩니다.

> **Important**
>
> 대상 유형 RESOURCE_GROUP의 경우 작업에 필요한 ID 형식에 따라 작업이 실행될 때 {{TARGET_ID}} 및 {{RESOURCE_ID}}를 사용하여 리소스를 참조하도록 선택할 수 있습니다. {{TARGET_ID}}는 리소스의 전체 ARN을 반환합니다. {{RESOURCE_ID}}는 다음 예와 같이 더 짧은 이름 또는 리소스 ID만 반환합니다.

- {{TARGET_ID}} 형식: arn:aws:ec2:us-east-1:123456789012:instance/i-02573cafcfEXAMPLE
- {{RESOURCE_ID}} 형식: i-02573cafcfEXAMPLE

대상 유형 INSTANCE의 경우 {{TARGET_ID}} 및 {{RESOURCE_ID}} 파라미터는 모두 인스턴스 ID만 산출합니다. 자세한 내용은 지원되는 가상 파라미터 단원을 참조하십시오. {{TARGET_ID}} 및 {{RESOURCE_ID}}를 사용하여 AWS 리소스의 ID만 Automation, Lambda 및 Step Functions 태스크에 전달할 수 있습니다. Run Command 태스크에는 이 2가지 의사 파라미터를 사용할 수 없습니다.

가상 파라미터 예제

AWS Lambda 태스크에 대한 페이로드가 ID를 기준으로 인스턴스를 참조해야 한다고 가정합니다.

INSTANCE 유지 관리 기간을 사용하든 RESOURCE_GROUP 유지 관리 기간 대상을 사용하든 관계없이, {{RESOURCE_ID}} 의사 파라미터를 사용하여 이 작업을 수행할 수 있습니다. 예제:

```
"TaskArn": "arn:aws:lambda:us-east-2:111122223333:function:SSMTestFunction",
    "TaskType": "LAMBDA",
    "TaskInvocationParameters": {
        "Lambda": {
            "ClientContext": "ew0KICAi--truncated--0KIEXAMPLE",
            "Payload": "{ \"instanceId\": \"{{RESOURCE_ID}}\" }",
            "Qualifier": "$LATEST"
        }
    }
```

Amazon DynamoDB 테이블 같은 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 외에 지원되는 다른 대상 유형에 대해 Lambda 태스크를 실행하려는 경우 동일한 구문을 사용할 수 있으며 {{RESOURCE_ID}}는 테이블 이름만 산출합니다. 그러나 테이블의 전체 ARN이 필요한 경우 다음 예와 같이 {{TARGET_ID}}를 사용합니다.

```
"TaskArn": "arn:aws:lambda:us-east-2:111122223333:function:SSMTestFunction",
    "TaskType": "LAMBDA",
    "TaskInvocationParameters": {
        "Lambda": {
            "ClientContext": "ew0KICAi--truncated--0KIEXAMPLE",
```

```
            "Payload": "{ \"tableArn\": \"{{TARGET_ID}}\" }",
            "Qualifier": "$LATEST"
        }
    }
```

인스턴스 또는 기타 리소스 유형을 대상으로 하는 경우에도 동일한 구문이 적용됩니다. 리소스 그룹에 여러 리소스 유형이 추가된 경우 해당 리소스 각각에 대해 작업이 실행됩니다.

> **Important**
>
> 리소스 그룹에 포함될 수 있는 모든 리소스 유형이 {{RESOURCE_ID}} 파라미터 값을 산출하는 것은 아닙니다. 지원되는 리소스 유형 목록은 지원되는 가상 파라미터 섹션을 참조하세요.

또 다른 예로, EC2 인스턴스를 중지하는 Automation 태스크를 실행하려면 AWS-StopEC2Instance Systems Manager 문서(SSM 문서)를 TaskArn 값으로 지정하고 {{RESOURCE_ID}} 의사 파라미터를 사용합니다.

```
"TaskArn": "AWS-StopEC2Instance",
    "TaskType": "AUTOMATION"
    "TaskInvocationParameters": {
        "Automation": {
            "DocumentVersion": "1",
            "Parameters": {
                "instanceId": [
                    "{{RESOURCE_ID}}"
                ]
            }
        }
    }
```

Amazon Elastic Block Store(Amazon EBS) 볼륨의 스냅샷을 복사하는 Automation 태스크를 실행하려면 AWS-CopySnapshot SSM 문서를 TaskArn 값으로 지정하고 {{RESOURCE_ID}} 의사 파라미터를 사용합니다.

```
"TaskArn": "AWS-CopySnapshot",
    "TaskType": "AUTOMATION"
    "TaskInvocationParameters": {
        "Automation": {
            "DocumentVersion": "1",
```

```
            "Parameters": {
                "SourceRegion": "us-east-2",
                "targetType":"RESOURCE_GROUP",
                "SnapshotId": [
                    "{{RESOURCE_ID}}"
                ]
            }
        }
    }
```

지원되는 가상 파라미터

다음 목록은 `--task-invocation-parameters` 옵션의 {{*PSEUDO_PARAMETER*}} 구문을 사용하여 지정할 수 있는 의사 파라미터를 설명합니다.

- **`WINDOW_ID`**: 대상 유지 관리 기간의 ID입니다.
- **`WINDOW_TASK_ID`**: 실행 중인 기간 태스크의 ID입니다.
- **`WINDOW_TARGET_ID`**: 대상을 포함하는 기간 대상의 ID(대상 ID)입니다.
- **`WINDOW_EXECUTION_ID`**: 현재 실행 기간의 ID입니다.
- **`TASK_EXECUTION_ID`**: 현재 실행 작업의 ID입니다.
- **`INVOCATION_ID`**: 현재 호출의 ID입니다.
- **`TARGET_TYPE`**: 대상의 유형입니다. 지원되는 유형에는 `RESOURCE_GROUP` 및 `INSTANCE`가 있습니다.
- **`TARGET_ID`**:

  지정한 대상 유형이 `INSTANCE`인 경우 `TARGET_ID` 의사 파라미터는 인스턴스의 ID로 바뀝니다. 예를 들면 `i-078a280217EXAMPLE`입니다.

  지정한 대상 유형이 `RESOURCE_GROUP`인 경우 태스크 실행에 참조되는 값은 리소스의 전체 ARN입니다. 예: `arn:aws:ec2:us-east-1:123456789012:instance/i-078a280217EXAMPLE`. 다음 표에서는 리소스 그룹의 특정 리소스 유형에 대한 샘플 `TARGET_ID` 값을 제공합니다.

> **Note**
>
> Run Command 태스크에는 `TARGET_ID`가 지원되지 않습니다.

리소스 유형	TARGET_ID 예	
AWS::CloudWatch::Alarm	arn:aws:cloudwatch:us-east-1:123456789012:alarm:MyCloudWatchAlarm i-078a280217EXAMPLE	
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::DynamoDB::Table	arn:aws:dynamodb:us-east-1:123456789012:table/MyTable	

리소스 유형	TARGET_ID 예
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::: DOC-EXAMPLE-BUCKET
AWS::SSM::ManagedInstance	arn:aws:ssm:us-east-1:123456789012:managed-instance/mi-0feadcf2d9EXAMPLE

- **RESOURCE_ID**: 리소스 그룹에 포함된 리소스 유형의 짧은 ID입니다. 다음 표에서는 리소스 그룹의 특정 리소스 유형에 대한 샘플 RESOURCE_ID 값을 제공합니다.

> **Note**
>
> Run Command 태스크에는 RESOURCE_ID가 지원되지 않습니다.

리소스 유형	RESOURCE_ID 예
AWS::CloudWatch::Alarm	MyCloudWatchAlarm
AWS::EC2::Instance	i-078a280217EXAMPLE
AWS::EC2::Image	ami-02250b3732EXAMPLE
AWS::EC2::SecurityGroup	sg-cEXAMPLE

리소스 유형	RESOURCE_ID 예
AWS::EC2::Snapshot	snap-03866bf003EXAMPLE
AWS::EC2::Volume	vol-0912e04d78EXAMPLE
AWS::DynamoDB::Table	MyTable
AWS::RDS::DBCluster	My-Cluster
AWS::RDS::DBInstance	My-SQL-Instance
AWS::S3::Bucket	DOC-EXAMPLE-BUCKET
AWS::SSM::ManagedInstance	mi-0feadcf2d9EXAMPLE

> **Note**
>
> 지정한 AWS 리소스 그룹에 RESOURCE_ID 값을 산출하지 않는 리소스 유형이 포함되어 있고 위 표에 나열되지 않은 경우 RESOURCE_ID 파라미터가 채워지지 않습니다. 해당 리소스에 대해 실행 호출이 계속 발생합니다. 이러한 경우 대신 TARGET_ID 의사 파라미터를 사용합니다. 이 파라미터는 리소스의 전체 ARN으로 대체됩니다.

유지 관리 기간 예약 및 유효 기간 옵션

유지 관리 기간을 생성할 때 Cron or rate expression(Cron 또는 Rate 표현식)을 사용하여 유지 관리 기간 실행 빈도를 지정해야 합니다. 선택적으로 유지 관리 기간을 정기 일정에 따라 실행할 수 있는 날짜 범위와 정규 일정의 기준으로 사용할 시간대를 지정할 수도 있습니다.

하지만 시간대 옵션과 시작 날짜 및 종료 날짜 옵션은 서로 영향을 주지 않습니다. 시간대에 대한 오프셋을 적용하거나 적용하지 않고 지정한 시작 날짜와 종료 날짜는 유지 관리 기간이 일정에 따라 실행될 수 있는 유효 기간만 결정합니다. 시간대 옵션은 유지 관리 기간 일정이 유효 기간 동안 기반으로 하는 국제 시간대를 결정합니다.

> **Note**
>
> 시작 날짜와 종료 날짜를 ISO-8601 타임스탬프 형식으로 지정합니다. 예: `2021-04-07T14:29:00-08:00`
> 시간대를 IANA(Internet Assigned Number Authority) 형식으로 지정합니다. 예: `America/Chicago`, `Europe/Berlin`, `Asia/Tokyo` 등

예제

예제 1: 유지 관리 기간 시작 날짜 지정

AWS Command Line Interface(AWS CLI)에서 다음 옵션을 사용하여 유지 관리 기간을 생성한다고 가정합니다.

- `--start-date 2021-01-01T00:00:00-08:00`
- `--schedule-timezone "America/Los_Angeles"`
- `--schedule "cron(0 09 ? * WED *)"`

예:

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 *)"
```

즉, 유지 관리 기간의 첫 번째 실행은 지정된 시작 날짜 및 시간, 즉 2021년 1월 1일 금요일 오전 12:00(미국 태평양 표준시) 후까지 수행되지 않습니다. 이 시간대는 UTC 시간보다 8시간 늦습니다. 이 경우 기간의 시작 날짜와 시간이 유지 관리 기간이 처음 실행되는 시간을 나타내지 않습니다. 종합하면 `--schedule-timezone` 및 `--schedule` 값은 유지 관리 기간을 매주 수요일 오전 9시(미국 태평양 표준시)(IANA 형식으로 "미국/로스앤젤레스"로 표시됨)에 실행합니다. 허용 기간 중 첫 번째 실행 시간은 2021년 1월 4일 수요일 오전 9시(미국 태평양 표준시)입니다.

예제 2: 유지 관리 기간 시작 날짜 및 종료 날짜 지정

다음에는 아래 옵션을 사용하여 유지 관리 기간을 생성한다고 가정합니다.

- `--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)"`

예:

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)"
```

이 유지 관리 기간에 대한 허용 기간은 2019년 1월 1일 오전 3시 15분(일본 표준시)에 시작됩니다. 이 유지 관리 기간에 대한 유효 기간은 2019년 6월 30일 일요일 오전 6시 15분(일본 표준시)에 종료됩니다. 이 시간대는 UTC 시간보다 9시간 빠릅니다. 종합하면 `--schedule-timezone` 및 `--schedule` 값은 유지 관리 기간을 매주 화요일 오전 3시 15분(일본 표준시)(IANA 형식으로 "아시아/도쿄"로 표시됨)에 실행합니다. 이유는 유지 관리 기간이 7일마다 실행되고 1월 1일 화요일 오전 3시 15분에 활성화되기 때문입니다. 마지막 실행 시간은 2019년 6월 25일 화요일 오전 3시 15분(일본 표준시)입니다. 이 시간은 허용된 유지 관리 기간이 종료되기 이전의 마지막 화요일이며 5일 후에 유효 기간이 종료됩니다.

예제 3: 한 번만 실행되는 유지 관리 기간 생성

이제 이 옵션으로 유지 관리 기간을 생성합니다.

- `--schedule "at(2020-07-07T15:55:00)"`

예:

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
```

이 유지 관리 기간은 2020년 7월 7일 오후 3시 55분(UTC 시간)에 한 번만 실행됩니다. 유지 관리 기간은 필요에 따라 최대 5시간 동안 실행하도록 허용되지만, 유지 관리 기간 종료 2시간 전에는 새 태스크를 시작할 수 없습니다

예제 4: 유지 관리 기간에 대한 일정 오프셋 일 수 지정

이제 이 옵션으로 유지 관리 기간을 생성합니다.

```
--schedule-offset 2
```

예:

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
```

일정 오프셋은 유지 관리 기간을 실행하기 전에 CRON 표현식에 의해 지정된 날짜 및 시간 이후에 대기할 일 수입니다.

위의 예에서 CRON 표현식은 매월 셋째 화요일 오후 11:30에 실행되도록 유지 관리 기간을 예약합니다.

```
--schedule "cron(0 30 23 ? * TUE#3 *)
```

그러나 `--schedule-offset 2`를 포함하면 매월 셋째 화요일 2일 후 오후 11시 30분까지 유지 관리 기간이 실행되지 않을 것임을 의미합니다.

일정 오프셋은 CRON 표현식에서만 지원됩니다.

추가 정보

- 참조: Systems Manager용 Cron 및 Rate 표현식
- 유지 관리 기간 생성(콘솔)
- 자습서: 유지 관리 기간 생성 및 구성(AWS CLI)
- AWS Systems Manager API Reference의 CreateMaintenanceWindow
- AWS CLI 명령 레퍼런스 AWS Systems Manager 섹션의 create-maintenance-window
- IANA 웹 사이트의 시간대 데이터베이스

대상 없이 유지 관리 기간 태스크 등록

생성하는 각 유지 관리 기간에 대해 유지 관리 기간이 실행될 때 수행할 하나 이상의 태스크를 지정할 수 있습니다. 대부분의 경우 태스크를 실행할 리소스 또는 대상을 지정해야 합니다. 그러나 태스크에서 대상을 명시적으로 지정할 필요가 없는 경우도 있습니다.

유지 관리 기간 Systems Manager Run Command 유형 태스크에 대해 하나 이상의 대상을 지정해야 합니다. 태스크 특성에 따라 대상은 다른 유지 관리 기간 태스크 유형(Systems Manager Automation, AWS Lambda 및 AWS Step Functions)에 대해 옵션입니다.

Lambda 및 Step Functions 태스크 유형의 경우 대상이 필요한지 여부는 생성한 기능 또는 상태 시스템의 콘텐츠에 따라 달라집니다.

대부분의 경우 자동화 태스크의 대상을 명시적으로 지정할 필요가 없습니다. 예를 들어 `AWS-UpdateLinuxAmi` 실행서를 사용하여 Linux용 Amazon Machine Image(AMI)를 업데이트하는 Automation 유형 태스크를 생성한다고 가정해 보겠습니다. 태스크가 실행되면 AMI는 사용 가능한 최신 Linux 배포 패키지와 Amazon 소프트웨어로 업데이트됩니다. AMI에서 생성된 새 인스턴스에는 이

러한 업데이트가 이미 설치되어 있습니다. 업데이트할 AMI의 ID가 실행서에 대한 입력 파라미터에 지정되어 유지 관리 기간 태스크에서 대상을 다시 지정할 필요가 없습니다.

마찬가지로 AWS Command Line Interface(AWS CLI)를 사용하여 `AWS-RestartEC2Instance` 런북을 사용하는 유지 관리 기간 Automation 태스크를 등록한다고 가정합니다. 다시 시작할 노드가 `--task-invocation-parameters` 인수에 지정되어 있으므로 `--targets` 옵션도 지정할 필요가 없습니다.

> **Note**
>
> 대상이 지정되지 않은 유지 관리 기간 태스크의 경우 `--max-errors` 및 `--max-concurrency` 값을 제공할 수 없습니다. 그 대신에 시스템에서 describe-maintenance-window-tasks 및 get-maintenance-window-task와 같은 명령에 대한 응답으로 보고될 수 있는 자리 표시자 값 1을 삽입합니다. 이러한 값은 태스크 실행에 영향을 주지 않으며 무시할 수 있습니다.

다음 예제에서는 대상 없는 유지 관리 기간 태스크에 대해 `--targets`, `--max-errors` 및 `--max-concurrency` 옵션을 생략하는 방법을 보여줍니다.

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**
>
> 2020년 12월 23일 이전에 등록된 유지 관리 기간 작업의 경우: 작업에 대상을 지정했고 하나가 더는 필요 없으면 Systems Manager 콘솔 또는 update-maintenance-window-task AWS CLI 명령으로 해당 작업을 업데이트하여 대상을 제거할 수 있습니다.

추가 정보

- 오류 메시지: “대상이 없는 유지 관리 기간 태스크는 MaxConcurrency 값을 지원하지 않음(Maintenance window tasks without targets don't support MaxConcurrency values)” 및 “대상이 없는 유지 관리 기간 태스크는 MaxErrors 값을 지원하지 않음(Maintenance window tasks without targets don't support MaxErrors values)”

유지 관리 기간 문제 해결

다음 정보를 사용하면 유지 관리 기간 관련 문제를 해결하는 데 도움이 됩니다.

주제

- 태스크 편집 오류: 유지 관리 기간 태스크 편집 페이지에서 IAM 역할 목록이 다음 오류 메시지를 반환합니다. "이 태스크에 지정된 IAM 유지 관리 기간 역할을 찾을 수 없습니다. 삭제되었거나 아직 생성되지 않았을 수 있습니다."
- 모든 유지 관리 기간 대상이 업데이트되지는 않습니다.
- 태스크 호출 상태가 “제공된 역할에 올바른 SSM 권한이 없습니다(The provided role does not contain the correct SSM permissions).”와 함께 태스크가 실패합니다.
- “단계 입력 검증 및 확인 시 단계 실패(Step fails when it is validating and resolving the step inputs)”라는 오류 메시지와 함께 태스크가 실패합니다.
- 오류 메시지: “대상이 없는 유지 관리 기간 태스크는 MaxConcurrency 값을 지원하지 않음(Maintenance window tasks without targets don't support MaxConcurrency values)” 및 “대상이 없는 유지 관리 기간 태스크는 MaxErrors 값을 지원하지 않음(Maintenance window tasks without targets don't support MaxErrors values)”

태스크 편집 오류: 유지 관리 기간 태스크 편집 페이지에서 IAM 역할 목록이 다음 오류 메시지를 반환합니다. "이 태스크에 지정된 IAM 유지 관리 기간 역할을 찾을 수 없습니다. 삭제되었거나 아직 생성되지 않았을 수 있습니다."

문제 1: 태스크 생성 후 원래 지정한 AWS Identity and Access Management(IAM) 유지 관리 기간 역할이 삭제되었습니다.

수정 방법: 1) 계정에 IAM 유지 관리 기간 역할이 하나 있으면 다른 해당 역할을 선택합니다. 아니면 새로 생성하고 작업을 위해 해당 역할을 선택합니다.

문제 2: AWS Command Line Interface(AWS CLI), AWS Tools for Windows PowerShell 또는 AWS SDK를 사용하여 태스크가 생성된 경우 존재하지 않는 IAM 유지 관리 기간 역할 이름이 지정되었을 수 있습니다. 예를 들어 태스크 생성 전에 IAM 유지 관리 기간 역할이 삭제되었거나, 역할 이름이 **my-role** 대신 **myrole**로 잘못 입력되었을 수 있습니다.

수정 방법: 사용할 IAM 유지 관리 기간 역할의 올바른 이름을 선택하거나, 새로 하나 생성하여 작업에 지정합니다.

모든 유지 관리 기간 대상이 업데이트되지는 않습니다.

문제: 유지 관리 기간의 대상이 되는 모든 리소스에서 유지 관리 기간 태스크가 실행되지 않았습니다. 예를 들어, 유지 관리 기간 실행 결과에서 해당 리소스에 대한 태스크가 실패 또는 시간 초과로 표시됩니다.

해결 방법: 대상 리소스에서 유지 관리 기간 태스크가 실행되지 않는 가장 일반적인 이유는 연결 및 가용성과 관련이 있습니다. 예:

- Systems Manager에서 유지 관리 기간 작업 전이나 중에 리소스에 대한 연결이 끊어졌습니다.
- 유지 관리 기간 동안 리소스가 오프라인 상태이거나 중지되었습니다.

리소스에 대한 태스크를 실행하기 위해 예약된 다음 유지 관리 기간을 기다릴 수 있습니다. 사용할 수 없거나 오프라인 상태였던 리소스에 대해 유지 관리 기간 태스크를 수동으로 실행할 수 있습니다.

태스크 호출 상태가 “제공된 역할에 올바른 SSM 권한이 없습니다(The provided role does not contain the correct SSM permissions).”와 함께 태스크가 실패합니다.

문제: 태스크에 대한 유지 관리 기간 서비스 역할을 지정했지만 태스크가 성공적으로 실행되지 않으며, 태스크 호출 상태가 “제공된 역할에 올바른 SSM 권한이 없습니다(The provided role does not contain the correct SSM permissions).”라고 보고됩니다.

- 해결 방법: 작업 1: 사용자 지정 유지 관리 기간 서비스 역할에 대한 정책 생성에서 사용자 지정 유지 관리 기간 서비스 역할에 연결할 수 있는 기본 정책을 제공합니다. 해당 정책에는 많은 태스크 시나리오에 필요한 권한이 포함되어 있습니다. 그러나 실행할 수 있는 태스크가 매우 다양하기 때문에 유지 관리 기간 역할에 대한 정책에서 추가 권한을 제공해야 할 수 있습니다.

  예를 들어 일부 자동화 작업은 AWS CloudFormation 스택과 함께 작동합니다. 따라서 유지 관리 기간 서비스 역할에 대한 정책에 추가 권한으로 `cloudformation:CreateStack`, `cloudformation:DescribeStacks`, 및 `cloudformation:DeleteStack`을 추가해야 할 수 있습니다.

  다른 예를 들어 Automation 실행서 `AWS-CopySnapshot`의 경우 Amazon Elastic Block Store(Amazon EBS) 스냅샷을 생성할 수 있는 권한이 필요합니다. 따라서 `ec2:CreateSnapshot` 권한을 추가해야 할 수 있습니다.

  AWS 관리형 Automation 런북을 사용하는 데 필요한 역할 권한에 대한 자세한 내용은 AWS Systems Manager Automation 런북 참조의 런북 설명을 참조하세요.

  AWS 관리형 SSM 문서에 따라 요구되는 역할 권한에 대한 자세한 내용은 Systems Manager 콘솔의 문서 섹션에서 문서 콘텐츠를 참조하세요.

  Step Functions 태스크, Lambda 태스크, 사용자 지정 Automation 런북 및 SSM 문서에 따라 요구되는 역할 권한에 대한 자세한 내용은 해당 리소스 작성자에게 문의하여 권한 요구 사항을 확인하세요.

“단계 입력 검증 및 확인 시 단계 실패(Step fails when it is validating and resolving the step inputs)”라는 오류 메시지와 함께 태스크가 실패합니다.

문제: 태스크에서 사용 중인 Automation 실행서 또는 Systems Manager Command 문서에 `InstanceId` 또는 `SnapshotId`와 같은 입력을 지정해야 하지만 값이 제공되지 않거나 제대로 제공되지 않습니다.

- 해결 방법 1: 태스크가 단일 노드 또는 단일 스냅샷과 같은 단일 리소스를 대상으로 하는 경우 태스크의 입력 파라미터에 해당 ID를 입력합니다.
- 해결 방법 2: 실행서 `AWS-CreateImage`를 사용할 때 여러 노드에서 이미지를 만드는 등 태스크가 여러 리소스를 대상으로 하는 경우 유지 관리 기간 태스크에 지원되는 의사 파라미터 중 하나를 입력 파라미터에 사용하여 명령에서 노드 ID를 지정할 수 있습니다.

  다음 명령은 AWS CLI를 사용하여 유지 관리 기간에 Systems Manager Automation 태스크를 등록합니다. `--targets` 값은 유지 관리 기간 대상 ID를 나타냅니다. 또한 `--targets` 파라미터가 기간 대

상 ID를 지정하더라도 Automation 실행서의 파라미터는 노드 ID를 제공해야 합니다. 이 경우 명령은 의사 파라미터 {{RESOURCE_ID}}를 InstanceId 값으로 사용합니다.

AWS CLI 명령:

Linux & macOS

다음 예제의 명령은 ID가 e32eecb2-646c-4f4b-8ed1-205fbEXAMPLE인 유지 관리 기간 대상 그룹에 속한 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스를 다시 시작합니다.

```
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"
```

유지 관리 기간 태스크를 위한 의사 파라미터 작업에 대한 자세한 내용은 유지 관리 기간 작업 등록 시 의사 파라미터 사용 및 작업 등록 예제 섹션을 참조하세요.

오류 메시지: “대상이 없는 유지 관리 기간 태스크는 MaxConcurrency 값을 지원하지 않음(Maintenance window tasks without targets don't support MaxConcurrency values)” 및 “대상이 없는 유지 관리 기간 태스크는 MaxErrors 값을 지원하지 않음(Maintenance window tasks without targets don't support MaxErrors values)”

문제: Run Command 유형 태스크를 등록할 때 태스크를 실행할 대상을 하나 이상 지정해야 합니다. 다른 태스크 유형(Automation, AWS Lambda 및 AWS Step Functions)의 경우 태스크의 특성에 따라 대상은 옵션입니다. 옵션 `MaxConcurrency`(태스크를 동시에 실행할 리소스 수) 및 `MaxErrors`(태스크가 실패하기 전에 대상 리소스에서 태스크 실행 실패 횟수)는 대상을 지정하지 않는 유지 관리 기간 태스크에 필요하지 않거나 지원되지 않습니다. 태스크 대상이 지정되지 않은 경우 이러한 옵션 중 하나에 값이 지정된 경우 시스템에서 이러한 오류 메시지를 생성합니다.

해결 방법: 이러한 오류 중 하나가 발생하면 유지 관리 기간 태스크를 계속 등록하거나 업데이트하기 전에 동시성 및 오류 임계값에 대한 값을 제거합니다.

대상을 지정하지 않는 태스크 실행에 대한 자세한 내용은 AWS Systems Manager 사용 설명서의 대상 없이 유지 관리 기간 태스크 등록 섹션을 참조하세요.

AWS Systems Manager 노드 관리

AWS Systems Manager는 다음과 같은 액세스, 관리 및 관리형 노드 구성 기능을 제공합니다. 관리형 노드는 하이브리드 및 멀티클라우드 환경에서 Systems Manager와 함께 사용하도록 구성된 모든 시스템입니다.

주제

- AWS Systems Manager Fleet Manager
- AWS Systems Manager Compliance
- AWS Systems Manager Inventory
- AWS Systems Manager 하이브리드 정품 인증
- AWS Systems Manager Session Manager
- AWS Systems Manager Run Command
- AWS Systems Manager State Manager
- AWS Systems Manager Patch Manager
- AWS Systems Manager Distributor

AWS Systems Manager Fleet Manager

AWS Systems Manager의 기능인 Fleet Manager는 AWS 또는 온프레미스에서 실행되는 관리형 노드를 원격으로 관리하는 데 도움이 되는 통합 사용자 인터페이스(UI) 환경입니다. Fleet Manager를 사용하면 하나의 콘솔에서 전체 서버 플릿의 상태 및 성능 상태를 볼 수 있습니다. 또한 개별 노드에서 데이터를 수집하여 콘솔에서 일반적인 문제 해결 및 관리 태스크를 수행할 수 있습니다. 여기에는 원격 데스크톱 프로토콜(RDF)을 사용하여 Windows 인스턴스에 연결, 폴더 및 파일 내용 보기, Windows 레지스트리 관리, 운영 체제 사용자 관리 등이 포함됩니다. Fleet Manager를 시작하려면 Systems Manager 콘솔을 엽니다. 탐색 창에서 Fleet Manager를 선택합니다.

Fleet Manager는 누가 사용해야 하나요?

노드 플릿을 중앙 집중식으로 관리하려는 모든 AWS 고객은 Fleet Manager를 사용해야 합니다.

Fleet Manager가 조직에 주는 이점은 무엇인가요?

Fleet Manager에서 제공하는 이점은 다음과 같습니다.

- 관리형 노드에 수동으로 연결할 필요 없이 다양한 일반적인 시스템 관리 태스크를 수행합니다.
- 단일 통합 콘솔에서 여러 플랫폼에서 실행되는 노드를 관리합니다.
- 단일 통합 콘솔에서 서로 다른 운영 체제를 실행하는 노드를 관리합니다.
- 시스템 관리의 효율성을 개선합니다.

Fleet Manager에는 어떤 기능이 있나요?

Fleet Manager의 주요 기능은 다음과 같습니다.

- Red Hat Knowledgebase 포털 액세스

  Red Hat Enterprise Linux(RHEL) 인스턴스를 통해 Red Hat Knowledgebase 포털에서 바이너리, 지식 공유 및 토론 포럼에 액세스하세요.

- 관리형 노드 상태

  어떤 관리형 인스턴스가 `running`이고 어떤 인스턴스가 `stopped`인지 봅니다. 중지된 인스턴스에 대한 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스 중지 및 시작을 참조하세요. AWS IoT Greengrass 코어 디바이스에서 `online`, `offline`을 볼 수 있고, `Connection lost`의 상태를 표시할 수 있습니다.

  > **Note**
  >
  > 2021년 7월 12일 이전에 관리형 인스턴스를 중지한 경우 `stopped` 마커가 표시되지 않습니다. 마커를 표시하려면 인스턴스를 시작하고 중지합니다.

- 인스턴스 정보 보기

  관리형 인스턴스에 연결된 볼륨에 저장된 폴더 및 파일 데이터에 대한 정보, 인스턴스에 대한 실시간 성능 데이터, 인스턴스에 저장된 로그 데이터에 대한 정보를 봅니다.

- 엣지 디바이스 정보 보기

  디바이스의 AWS IoT Greengrass 사물 이름, SSM Agent ping 상태 및 버전 등을 봅니다.

- 계정 및 레지스트리 관리

  인스턴스의 운영 체제(OS) 사용자 계정과 Windows 인스턴스의 레지스트리를 관리합니다.

- 서비스에 대한 액세스 제어

AWS Identity and Access Management(IAM) 정책을 사용하여 Fleet Manager 기능에 대한 액세스를 제어합니다. 이러한 정책을 통해 조직의 어떤 개별 사용자 또는 그룹이 다양한 Fleet Manager 기능을 사용할 수 있는지 그리고 어떤 관리형 노드를 관리할 수 있는지 제어할 수 있습니다.

주제

Fleet Manager 시작하기

AWS Systems Manager의 기능인 Fleet Manager를 사용하여 관리형 노드를 모니터링하고 관리하려면 먼저 다음 주제의 단계를 완료합니다.

주제

1단계: Fleet Manager 권한이 있는 IAM 정책 생성

AWS Systems Manager의 기능인 Fleet Manager를 사용하려면 AWS Identity and Access Management(IAM) 사용자 또는 역할에 필요한 권한이 있어야 합니다. 모든 Fleet Manager 기능에 대한 액세스를 제공하는 IAM 정책을 생성하거나 선택한 기능에 대한 액세스 권한을 부여하도록 정책을 수정할 수 있습니다.

아래의 정책 샘플에서는 모든 Fleet Manager 특성에 필요한 권한과 특성의 하위 집합에 필요한 권한이 제공됩니다.

IAM 정책 생성 및 편집에 대한 자세한 내용은 IAM User Guide의 Creating IAM Policies를 참조하세요.

주제

Fleet Manager 관리자 액세스 정책 샘플

다음 정책은 모든 Fleet Manager 기능에 대한 권한을 제공합니다. 즉, 사용자는 로컬 사용자 및 그룹을 생성 및 삭제할 수 있으며 모든 로컬 그룹에 대한 그룹 멤버십을 수정하고 Windows Server 레지스트리 키 또는 값을 수정할 수 있습니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
{
   "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"
         ],
         "Condition":{
            "BoolIfExists":{
               "ssm:SessionDocumentAccessCheck":"true"
            }
         }
      },
      {
         "Sid":"TerminateSession",
         "Effect":"Allow",
         "Action":[
            "ssm:TerminateSession"
         ],
         "Resource":"*",
         "Condition":{
            "StringLike":{
               "ssm:resourceTag/aws:ssmmessages:session-id":[
                  "${aws:userid}"
               ]
            }
         }
      },
      {
```

```
         "Sid":"KMS",
         "Effect":"Allow",
         "Action":[
            "kms:GenerateDataKey"
         ],
         "Resource":[
            "arn:aws:kms:region:account-id:key/key-name"
         ]
      }
   ]
}
```

Fleet Manager 읽기 전용 액세스 정책 샘플

다음 정책은 읽기 전용 Fleet Manager 기능에 대한 권한을 제공합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
{
   "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"
         ],
         "Condition":{
            "BoolIfExists":{
               "ssm:SessionDocumentAccessCheck":"true"
            }
         }
      },
      {
         "Sid":"TerminateSession",
         "Effect":"Allow",
         "Action":[
            "ssm:TerminateSession"
         ],
         "Resource":"*",
         "Condition":{
            "StringLike":{
               "ssm:resourceTag/aws:ssmmessages:session-id":[
                  "${aws:userid}"
```

```
                ]
            }
         }
      },
      {
         "Sid":"KMS",
         "Effect":"Allow",
         "Action":[
            "kms:GenerateDataKey"
         ],
         "Resource":[
            "arn:aws:kms:region:account-id:key/key-name"
         ]
      }
   ]
}
```

2단계: Systems Manager가 인스턴스 및 엣지 디바이스를 관리하는지 확인

Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에서, AWS Systems Manager의 기능인 Fleet Manager를 사용하여 AWS IoT Greengrass 코어 디바이스, 온프레미스 서버, 엣지 디바이스 및 가상 머신을 모니터링하고 관리하려면 Systems Manager 관리형 노드여야 합니다. 즉, 노드가 특정 전제 조건을 충족하고 AWS Systems Manager 에이전트(SSM Agent)로 구성되어야 합니다. 자세한 내용은 AWS Systems Manager 설정 단원을 참조하십시오.

AWS Systems Manager의 기능인 Quick Setup를 사용하여 Amazon EC2 인스턴스를 개별 계정에서 관리형 인스턴스로 빠르게 구성할 수 있습니다. 비즈니스 또는 조직이 AWS Organizations을 사용하는 경우, 여러 조직 구성 단위(OU)와 AWS 리전에 걸쳐 인스턴스를 구성할 수도 있습니다. Quick Setup을 사용하여 관리형 인스턴스 구성에 대한 자세한 내용은 Amazon EC2 호스트 관리 섹션을 참조하세요.

> **Note**
>
> AWS에서 실행되지 않는 비 EC2 시스템의 경우 하이브리드 정품 인증을 사용하여 하이브리드 및 멀티클라우드의 Systems Manager에서 사용할 시스템을 구성합니다. 하이브리드 정품 인증에 대한 자세한 내용은 AWS Systems Manager 하이브리드 정품 인증 섹션을 참조하세요.

Fleet Manager 작업

AWS Systems Manager의 기능인 Fleet Manager를 사용하여 AWS Systems Manager 콘솔에서 관리형 노드에 대한 다양한 태스크를 수행할 수 있습니다. 다음 주제에서는 Fleet Manager에서 제공하는 기능에 대해 설명합니다.

> **Note**
>
> macOS 인스턴스에 대해 지원되는 유일한 기능은 파일 시스템 보기입니다.

주제

관리형 노드 작업

관리형 노드는 AWS Systems Manager용으로 구성된 시스템입니다. 다음과 같은 시스템 유형을 관리형 노드로 구성할 수 있습니다.

- Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스
- 자체 프레미스의 서버(온프레미스 서버)
- AWS IoT Greengrass 코어 디바이스
- AWS IoT 및 비 AWS 엣지 디바이스
- 다른 클라우드 환경의 VM을 포함한 가상 머신

> **Note**
>
> Systems Manager 콘솔에서, "mi-"로 시작하는 시스템은 하이브리드 정품 인증을 사용하는 관리형 노드로 구성되어 있습니다. 엣지 디바이스는 AWS IoT 사물 이름을 보여줍니다.

AWS Systems Manager에서는 표준 인스턴스 티어와 고급 인스턴스 티어를 제공합니다. 둘 다 하이브리드 및 멀티클라우드 환경에서 관리형 노드를 지원합니다. 표준 인스턴스 티어를 통해 AWS 리전의 AWS 계정당 최대 1,000개의 시스템을 등록할 수 있습니다. 단일 계정 및 리전에 1,000개 이상의 시스템을 등록해야 하는 경우 고급 인스턴스 티어를 사용합니다. 고급 인스턴스 티어에서 원하는 만큼 관리형 노드를 생성할 수 있습니다. Systems Manager를 위해 구성된 모든 관리형 노드의 가격은 사용량에 따라 지불하는 방식으로 책정됩니다. 고급 인스턴스 티어 활성화에 대한 자세한 내용은 고급 인스턴스 티어 설정 섹션을 참조하세요. 요금에 대한 자세한 내용은 AWS Systems Manager 요금을 참조하세요.

> **Note**
>
> - 고급 인스턴스도 AWS Systems Manager Session Manager를 사용하여 하이브리드 및 멀티클라우드 환경의 비 EC2 노드에 연결할 수 있습니다. Session Manager에서는 인스턴스에 대한 대화형 쉘 액세스가 제공됩니다. 자세한 내용은 AWS Systems Manager Session Manager 단원을 참조하십시오.
> - 표준 인스턴스 할당량은 또한 Systems Manager 온프레미스 정품 인증을 사용하는 EC2 인스턴스에도 적용됩니다(일반 시나리오는 아님).
> - Microsoft에서 릴리스한 가상 머신 온프레미스 인스턴스 기반 애플리케이션을 패치하려면 고급 인스턴스 티어를 활성화합니다. 고급 인스턴스 티어 사용에는 요금이 따릅니다. Microsoft에서 릴리스한 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 기반 애플리케이션을 패치하는 데는 추가 요금이 없습니다. 자세한 내용은 Microsoft에서 릴리스한 Windows Server 기반 애플리케이션 패치 정보 단원을 참조하십시오.

관리형 노드 표시

콘솔에 관리형 노드가 표시되지 않으면 다음을 수행합니다.

1. 콘솔이 관리형 노드를 생성한 AWS 리전에서 열렸는지 확인합니다. 콘솔의 오른쪽 상단 모서리에 있는 목록을 사용하여 리전을 변경할 수 있습니다.
2. 관리형 노드 설정 단계에서 Systems Manager 요구 사항이 충족되는지 확인합니다. 자세한 설명은 AWS Systems Manager 설정을 참조하세요.

3. 비 EC2 시스템의 경우 하이브리드 정품 인증 프로세스를 완료했는지 확인합니다. 자세한 내용은 하이브리드 및 멀티클라우드 환경에서 Systems Manager 사용하기 단원을 참조하십시오.

> **Note**
>
> 다음 정보를 참고하세요.
>
> - Fleet Manager 콘솔에는 종료된 Amazon EC2 노드가 표시되지 않습니다.
> - Systems Manager는 시스템에서 작업을 수행하기 위해 정확한 시간 참조가 필요합니다. 노드의 날짜와 시간이 잘못 설정되어 있으면 API 요청의 서명 날짜와 일치하지 않을 수 있습니다. 자세한 내용은 사용 사례 및 모범 사례 단원을 참조하십시오.
> - 태그를 만들거나 편집할 때 시스템에서 테이블 필터에 변경 사항을 표시하는 데 최대 1시간이 걸릴 수 있습니다.
> - 관리형 노드의 상태가 30일 이상 `Connection Lost`한 후에는 해당 노드가 Fleet Manager 콘솔에 더 이상 나열되지 않을 수 있습니다. 목록으로 복원하려면 연결이 끊긴 문제를 해결해야 합니다. 문제 해결 팁은 관리형 노드 가용성 문제 해결 섹션을 참조하세요.

관리형 노드에서 Systems Manager 지원 확인

AWS Config은(는) AWS 관리형 규칙을 제공합니다. AWS Config은(는) 미리 정의되고 사용자 정의 가능한 이 규칙을 사용하여 AWS 리소스 구성이 공통 모범 사례를 준수하는지 평가합니다. AWS Config 관리형 규칙에는 ec2-instance-managed-by-systems-manager 규칙이 포함됩니다. 이 규칙은 해당 계정의 Amazon EC2 인스턴스가 Systems Manager에서 관리되는지를 확인합니다. 자세한 내용은 AWS Config 관리형 규칙을 참조하세요.

관리형 노드에서 보안 태세 강화

관리형 노드에서 무단 루트 수준 명령에 대한 보안 태세를 강화하는 방법은 SSM Agent를 통한 루트 수준 명령에 대한 액세스 제한 섹션을 참조하세요.

관리형 노드 등록 취소

관리형 노드는 언제든지 등록을 취소할 수 있습니다. 예를 들어 동일한 AWS Identity and Access Management(IAM) 역할의 노드를 여러 개 관리하면서 어떤 종류의 악의적인 동작을 감지하는 경우, 언제든지 원하는 수 컴퓨터를 등록 취소할 수 있습니다. 관리형 노드 등록 취소에 대한 자세한 내용은 하이브리드 및 멀티클라우드 환경의 관리형 노드 등록 취소 섹션을 참조하세요.

주제

- 인스턴스 티어 구성
- 관리형 노드의 암호 재설정
- 하이브리드 및 멀티클라우드 환경의 관리형 노드 등록 취소

인스턴스 티어 구성

이 주제에서는 고급 인스턴스 티어를 활성화해야 하는 시나리오에 대해 설명합니다.

AWS Systems Manager에서는 하이브리드 및 멀티클라우드 환경의 비 EC2 시스템에 대한 표준 인스턴스 티어와 고급 인스턴스 티어를 제공합니다.

추가 비용 없이 AWS 리전별로 계정당 최대 1,000개의 표준 하이브리드 정품 인증 노드를 등록할 수 있습니다. 그러나 1,000개 이상의 하이브리드 노드를 등록하려면 고급 인스턴스 티어를 활성화해야 합니다. 고급 인스턴스 티어를 사용하는 데는 요금이 부과됩니다. 자세한 내용은 AWS Systems Manager 요금을 참조하세요.

등록된 하이브리드 정품 인증 노드가 1,000개 미만인 경우에도 두 가지 다른 시나리오에는 고급 인스턴스 티어가 필요합니다.

- Session Manager를 사용하여 비EC2 노드에 연결하려고 합니다.
- Microsoft에서 릴리스한 비EC2 노드 기반 애플리케이션(운영 체제 아님)을 패치하려고 합니다.

> **Note**
>
> Microsoft에서 릴리스한 Amazon EC2 인스턴스 기반 애플리케이션을 패치하는 것은 무료입니다.

고급 인스턴스 티어 세부 시나리오

다음 정보는 고급 인스턴스 티어를 활성화해야 하는 세 가지 시나리오에 대한 세부 정보를 제공합니다.

시나리오 1: 1,000개가 넘는 하이브리드 정품 인증 노드를 등록하려는 경우

표준 인스턴스 티어를 사용하면 추가 비용 없이 특정 계정에서 AWS 리전당 최대 1,000개의 비EC2 노드를 하이브리드 및 멀티클라우드 환경에서 등록할 수 있습니다. 한 리전에 1,000개 이상의 비EC2 노드를 등록해야 하는 경우 고급 인스턴스 티어를 사용해야 합니다. 그런 다음에 하이브리드 및 멀티클라우드 환경의 시스템을 원하는 만큼 정품 인증할 수 있습니다. 고급 인스턴스 티어에

대한 요금은 Systems Manager 관리형 노드로 정품 인증된 고급 노드 수와 해당 노드가 실행되는 시간을 기준으로 합니다.

특정 계정의 한 리전에서 1,000개의 온프레미스 노드를 초과하는 경우 하이브리드 활성화를 생성하여 Systems Manager에 노드 등록에 설명된 활성화 프로세스를 사용하는 모든 Systems Manager 관리형 노드에 요금이 부과됩니다.

> **Note**
>
> 또한 Systems Manager 하이브리드 활성화를 사용하여 기존 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스를 활성화하고 테스트 등에 비EC2 인스턴스로 사용할 수 있습니다. 이러한 노드는 하이브리드 노드로 사용할 수도 있습니다. 이는 일반적인 시나리오가 아닙니다.

시나리오 2: Microsoft에서 릴리스한 하이브리드 활성 노드 기반 애플리케이션 패치

하이브리드 및 멀티클라우드 환경에 있는 Microsoft에서 릴리스한 비 EC2 노드의 애플리케이션에 패치를 적용하려는 경우에도 고급 인스턴스 티어가 필요합니다. 비EC2 노드 기반의 Microsoft 애플리케이션을 패치하기 위해 고급 인스턴스 티어를 활성화하면 1,000개 미만이더라도 모든 온프레미스 노드에 대해 요금이 부과됩니다.

Microsoft에서 릴리스한 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 기반 애플리케이션을 패치하는 데는 추가 요금이 없습니다. 자세한 내용은 Microsoft에서 릴리스한 Windows Server 기반 애플리케이션 패치 정보 단원을 참조하십시오.

시나리오 3: Session Manager를 사용하여 하이브리드 활성 노드에 연결

Session Manager는 인스턴스에 대한 대화형 셸 액세스를 제공합니다. Session Manager를 사용하여 하이브리드 정품 인증 관리형 노드에 연결하려면 고급 인스턴스 티어를 정품 인증해야 합니다. 그러면 1,000개 미만이라도 모든 하이브리드 활성 노드에 대해 요금이 부과됩니다.

요약: 고급 인스턴스 티어는 언제 필요한가요?

다음 표를 사용하여 고급 인스턴스 티어를 사용해야 하는 시기와 추가 요금이 적용되는 시나리오를 검토하세요.

시나리오	고급 인스턴스 티어가 필요한가요?	추가 요금이 발생하나요?
특정 계정에서 내 리전의 하이브리드 활성 노드 수가 1,000개 이상입니다.	예	예
하이브리드 활성 노드가 1,000개 미만이라도 Patch Manager를 사용하여 Microsoft에서 릴리스한 하이브리드 활성 노드 기반 애플리케이션을 패치하고 싶습니다.	예	예
하이브리드 활성 노드가 1,000개 미만이라도 Session Manager를 사용하여 하이브리드 활성 노드에 연결하고 싶습니다.	예	예
1. 특정 계정의 한 리전에서 하이브리드 활성 노드 수가 1,000개 이하입니다. 2. 하이브리드 활성 노드 기반 Microsoft 애플리케이션을 패치하고 있지 않습니다. 3. Session Manager를 사용하여 하이브리드 활성 노드에 연결하고 있지 않습니다.	아니요	아니요

주제

- 고급 인스턴스 티어 설정
- 고급 인스턴스 티어에서 표준 인스턴스 티어로 되돌리기

고급 인스턴스 티어 설정

AWS Systems Manager에서는 하이브리드 및 멀티클라우드 환경의 비 EC2 시스템에 대한 표준 인스턴스 티어와 고급 인스턴스 티어를 제공합니다. 표준 인스턴스 티어를 사용하면 한 AWS 리전의 AWS 계정당 최대 1,000개의 하이브리드 활성 머신을 등록할 수 있습니다. 또한 고급 인스턴스 티어는 Patch Manager를 사용하여 Microsoft에서 릴리스한 비EC2 노드 기반 애플리케이션을 패치하고 Session Manager를 사용하여 비EC2 노드에 연결하는 데 필요합니다. 자세한 내용은 인스턴스 티어 구성 단원을 참조하십시오.

이 섹션에서는 고급 인스턴스 티어를 사용하도록 하이브리드 및 멀티클라우드 환경을 구성하는 방법을 설명합니다.

시작하기 전 준비 사항

고급 인스턴스에 대한 요금 내역을 검토합니다. 고급 인스턴스는 종량 과금제로 이용 가능합니다. 자세한 내용은 AWS Systems Manager 요금을 참조하세요.

고급 인스턴스 티어를 설정하는 권한 구성

AWS Identity and Access Management(IAM)에 표준 인스턴트 티어에서 고급 인스턴스 티어로 환경을 변경할 수 있는 권한이 있는지 확인합니다. 사용자, 그룹 또는 역할에 `AdministratorAccess` IAM 정책이 연결되었거나 Systems Manager 활성화 티어 서비스 설정을 변경할 수 있는 권한이 있어야 합니다. 활성화 티어 설정은 다음 API 작업을 사용합니다.

- GetServiceSetting
- UpdateServiceSetting
- ResetServiceSetting

다음 절차를 사용하여 인라인 IAM 정책을 사용자 계정에 연결합니다. 이 정책을 통해 사용자는 현재 관리형 인스턴스 티어 설정을 볼 수 있습니다. 또한 이 정책을 통해 사용자는 지정된 AWS 계정와 AWS 리전에서 현재 설정을 변경하거나 재설정할 수 있습니다.

1. AWS Management Console에 로그인하여 https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 사용자를 선택합니다.
3. 목록에서 정책을 삽입할 사용자 이름을 선택합니다.
4. 권한 탭을 선택합니다.

5. 페이지 오른쪽에 있는 Permission policies(권한 정책)에서 Add inline policy(인라인 정책 추가)를 선택합니다.
6. JSON 탭을 선택합니다.
7. 기본 내용을 다음으로 바꿉니다.

```
{
    "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. 정책 검토를 선택합니다.
9. Review Policy(정책 검토) 페이지의 이름에 인라인 정책 이름을 입력합니다. 예: **Managed-Instances-Tier**.
10. 정책 생성을 선택합니다.

관리자는 다음 인라인 정책을 사용자에게 할당하여 읽기 전용 권한을 지정할 수 있습니다.

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

```
            "Action": [
                "ssm:GetServiceSetting"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Deny",
            "Action": [
                "ssm:ResetServiceSetting",
                "ssm:UpdateServiceSetting"
            ],
            "Resource": "*"
        }
    ]
}
```

IAM 정책 생성 및 편집에 대한 자세한 내용은 IAM User Guide의 Creating IAM Policies를 참조하세요.

고급 인스턴스 티어 설정(콘솔)

다음 절차에서는 Systems Manager 콘솔을 사용하여 지정된 AWS 계정 및 AWS 리전에서 관리형 인스턴스 정품 인증을 통해 추가된 모든 비 EC2 노드를 고급 인스턴스 티어 사용으로 변경하는 방법을 보여줍니다.

시작하기 전 준비 사항

콘솔이 관리형 인스턴스를 생성한 AWS 리전에서 열렸는지 확인합니다. 콘솔의 오른쪽 상단 모서리에 있는 목록을 사용하여 리전을 변경할 수 있습니다.

하이브리드 및 멀티클라우드 환경의 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 및 비 EC2 시스템에 대한 설정 요구 사항을 완료했는지 확인합니다. 자세한 설명은 AWS Systems Manager 설정을 참조하세요.

> **Important**
>
> 다음 절차는 계정 수준 설정을 변경하는 방법을 설명합니다. 이 변경 사항으로 인해 계정에 요금이 청구됩니다.

고급 인스턴스 티어를 설정하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.

2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 설정, 인스턴스 계층 설정 변경을 선택합니다.
4. 대화 상자에서 계정 설정 변경에 대한 정보를 검토하고 계속 진행합니다.
5. 승인하는 경우 수락할 옵션을 선택하고 설정 변경을 선택합니다.

시스템에서 모든 인스턴스를 표준 인스턴스 티어에서 고급 인스턴스 티어로 이동하는 프로세스를 완료하는 데 몇 분이 소요될 수 있습니다.

> **Note**
>
> 표준 인스턴스 티어로 다시 변경하는 방법에 대한 자세한 내용은 고급 인스턴스 티어에서 표준 인스턴스 티어로 되돌리기 섹션을 참조하세요.

고급 인스턴스 티어 설정(AWS CLI)

다음 절차에서는 AWS Command Line Interface를 사용하여 지정된 AWS 계정와 AWS 리전에서 관리형 인스턴스 활성화를 통해 추가된 모든 온프레미스 서버와 VM을 고급 인스턴스 티어 사용으로 변경하는 방법을 설명합니다.

> **Important**
>
> 다음 절차는 계정 수준 설정을 변경하는 방법을 설명합니다. 이 변경 사항으로 인해 계정에 요금이 청구됩니다.

AWS CLI를 사용하여 고급 인스턴스 티어를 설정하려면

1. AWS CLI을 열고 다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   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
```

명령이 성공해도 결과는 없습니다.

2. 다음 명령을 실행하여 현재 AWS 계정 및 AWS 리전의 관리형 노드에 대한 현재 서비스 설정을 확인합니다.

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
```

명령은 다음과 같은 정보를 반환합니다.

```
{
    "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"
    }
}
```

고급 인스턴스 티어 설정(PowerShell)

다음 절차에서는 AWS Tools for Windows PowerShell를 사용하여 지정된 AWS 계정와 AWS 리전에서 관리형 인스턴스 활성화를 통해 추가된 모든 온프레미스 서버와 VM을 고급 인스턴스 티어 사용으로 변경하는 방법을 설명합니다.

> **Important**
>
> 다음 절차는 계정 수준 설정을 변경하는 방법을 설명합니다. 이 변경 사항으로 인해 계정에 요금이 청구됩니다.

PowerShell을 사용하여 고급 인스턴스 티어를 설정하려면

1. AWS Tools for Windows PowerShell을 열고 다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   ```
   Update-SSMServiceSetting `
       -SettingId "arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-
   instance/activation-tier" `
       -SettingValue "advanced"
   ```

   명령이 성공해도 결과는 없습니다.

2. 다음 명령을 실행하여 현재 AWS 계정 및 AWS 리전의 관리형 노드에 대한 현재 서비스 설정을 확인합니다.

   ```
   Get-SSMServiceSetting `
       -SettingId "arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-
   instance/activation-tier"
   ```

   명령은 다음과 같은 정보를 반환합니다.

   ```
   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
   ```

시스템에서 모든 노드를 표준 인스턴스 티어에서 고급 인스턴스 티어로 이동하는 프로세스를 완료하는 데 몇 분이 소요될 수 있습니다.

> **Note**
>
> 표준 인스턴스 티어로 다시 변경하는 방법에 대한 자세한 내용은 고급 인스턴스 티어에서 표준 인스턴스 티어로 되돌리기 섹션을 참조하세요.

고급 인스턴스 티어에서 표준 인스턴스 티어로 되돌리기

이 섹션에서는 고급 인스턴스 티어에서 실행되는 하이브리드 정품 인증 노드를 표준 인스턴스 티어로 다시 변경하는 방법을 설명합니다. 이 구성은 AWS 계정 및 단일 AWS 리전의 모든 하이브리드 정품 인증 노드에 적용됩니다.

시작하기 전 준비 사항

다음과 같은 중요한 세부 정보를 검토합니다.

> **Note**
>
> - 계정과 리전에서 1,000개가 넘는 하이브리드 정품 인증 노드 실행하는 경우에는 표준 인스턴스 티어로 되돌릴 수 없습니다. 먼저 1,000개 이하가 될 때까지 노드의 등록을 취소해야 합니다. 이는 Systems Manager 하이브리드 정품 인증(일반적인 시나리오가 아님)을 사용하는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에도 적용됩니다. 자세한 내용은 하이브리드 및 멀티클라우드 환경의 관리형 노드 등록 취소 단원을 참조하십시오.
> - 되돌리면 AWS Systems Manager의 기능인 Session Manager를 사용하여 하이브리드 정품 인증 노드에 대화형으로 액세스할 수 없습니다.
> - 되돌리면 하이브리드 정품 인증 노드에서 AWS Systems Manager의 기능인 Patch Manager를 사용하여 Microsoft에서 릴리스한 애플리케이션에 패치를 적용할 수 없습니다.
> - 모든 하이브리드 정품 인증 노드를 표준 인스턴스 티어로 되돌리는 프로세스를 완료하는 데 30분 이상이 걸릴 수 있습니다.

이 섹션에서는 AWS 계정 및 AWS 리전의 모든 하이브리드 정품 인증 노드를 고급 인스턴스 티어에서 표준 인스턴스 티어로 되돌리는 방법을 설명합니다.

표준 인스턴스 티어로 되돌리기(콘솔)

다음 절차에서는 Systems Manager 콘솔을 사용하여 지정된 AWS 계정 및 AWS 리전에서 표준 인스턴스 티어를 사용하도록 하이브리드 및 멀티클라우드 환경의 모든 하이브리드 정품 인증 노드를 변경하는 방법을 보여줍니다.

표준 인스턴스 티어로 되돌리려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. [계정 설정(Account settings)] 드롭다운을 선택하고 [인스턴스 티어 설정(Instance tier settings)]을 선택합니다.
4. Change account settings(계정 설정 변경)를 선택합니다.
5. 팝업 창에 있는 계정 설정 변경 관련 정보를 검토하고 이를 승인하는 경우에는 수락 후 계속 진행할 수 있는 옵션을 선택합니다.

표준 인스턴스 티어로 되돌리기(AWS CLI)

다음 절차에서는 AWS Command Line Interface를 사용하여 지정된 AWS 계정 및 AWS 리전에서 표준 인스턴스 티어를 사용하도록 하이브리드 및 멀티클라우드 환경의 모든 하이브리드 정품 인증 노드를 변경하는 방법을 보여줍니다.

AWS CLI을 사용하여 표준 인스턴스 티어로 되돌리려면

1. AWS CLI을 열고 다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   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
```

명령이 성공해도 결과는 없습니다.

2. 30분 후에 다음 명령을 실행하여 현재 AWS 계정와 AWS 리전의 관리형 인스턴스에 대한 설정을 확인합니다.

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
```

명령은 다음과 같은 정보를 반환합니다.

```
{
    "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"
    }
}
```

요청이 승인되면 상태가 기본값으로 변경됩니다.

표준 인스턴스 티어로 되돌리기(PowerShell)

다음 절차에서는 AWS Tools for Windows PowerShell를 사용하여 지정된 AWS 계정 및 AWS 리전에서 표준 인스턴스 티어를 사용하도록 하이브리드 및 멀티클라우드 환경의 하이브리드 정품 인증 노드를 변경하는 방법을 보여줍니다.

PowerShell을 사용하여 표준 인스턴스 티어로 되돌리려면

1. AWS Tools for Windows PowerShell을 열고 다음 명령을 실행합니다.

   ```
   Update-SSMServiceSetting `
       -SettingId "arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-
   instance/activation-tier" `
       -SettingValue "standard"
   ```

   명령이 성공해도 결과는 없습니다.

2. 30분 후에 다음 명령을 실행하여 현재 AWS 계정와 AWS 리전의 관리형 인스턴스에 대한 설정을 확인합니다.

   ```
   Get-SSMServiceSetting `
       -SettingId "arn:aws:ssm:region:aws-account-id:servicesetting/ssm/managed-
   instance/activation-tier"
   ```

   명령은 다음과 같은 정보를 반환합니다.

   ```
   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
   ```

   요청이 승인되면 상태가 기본값으로 변경됩니다.

관리형 노드의 암호 재설정

관리형 노드에서 모든 사용자 암호를 재설정할 수 있습니다. 여기에는 AWS Systems Manager에서 관리하는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스, AWS IoT Greengrass 코어 디바

이스, 온프레미스 서버 및 가상 머신이 포함됩니다. 암호 재설정 기능은 AWS Systems Manager의 기능인 Session Manager에 바탕을 두고 있습니다. 이 기능을 사용하면 인바운드 포트를 개방하거나, Bastion Host를 유지하거나, SSH 키를 관리하지 않고도 관리형 노드에 연결할 수 있습니다.

암호 재설정은 사용자가 암호를 잊었거나, 혹은 RDP 또는 SSH를 관리형 노드에 연결하지 않고 빠르게 암호를 업데이트하고 싶을 때 매우 유용합니다.

필수 조건

관리형 노드 암호를 재설정하려면 먼저 다음 요건을 충족해야 합니다.

- 암호를 변경할 관리형 노드가 Systems Manager 관리형 노드여야 합니다. 또한,관리형 노드에 SSM Agent 버전 2.3.668.0 이상이 설치되어 있어야 합니다.) SSM Agent 설치 또는 업데이트에 대한 자세한 내용은 SSM Agent 작업 섹션을 참조하세요.
- 암호 재설정 기능은 계정에서 관리형 노드에 연결할 수 있도록 설정된 Session Manager 구성을 사용합니다. 따라서 Session Manager를 사용하려면 현재 AWS 리전에서 계정에 필요한 사전 조건이 충족되어야 합니다. 자세한 내용은 Session Manager 설정 단원을 참조하십시오.

  > **Note**
  >
  > 온프레미스 노드에 대한 Session Manager 지원은 고급 인스턴스 티어에만 제공됩니다. 자세한 내용은 고급 인스턴스 티어 설정 단원을 참조하십시오.

- 암호를 변경하는 AWS 사용자는 해당 관리형 노드에 대해 `ssm:SendCommand` 권한이 있어야 합니다. 자세한 내용은 태그를 기반으로 Run Command 액세스 제한 단원을 참조하십시오.

액세스 제한

암호를 재설정할 수 있는 사용자 권한을 특정 관리형 노드로 제한할 수 있습니다. Session Manager `ssm:StartSession` 작업에 `AWS-PasswordReset` SSM 문서와 함께 자격 증명 기반 정책을 사용하면 가능합니다. 자세한 내용은 인스턴스에 대한 사용자 세션 액세스 제어를 참조하세요.

데이터 암호화

관리형 노드에 대한 암호 재설정 옵션을 사용하려면 Session Manager 데이터에 대해 AWS Key Management Service(AWS KMS) 완전 암호화를 설정합니다. 자세한 내용은 세션 데이터의 KMS 키 암호화를 설정하려면(콘솔) 단원을 참조하십시오.

관리형 노드의 암호 재설정

Systems Manager 관리형 노드의 암호는 Systems Manager Fleet Manager 콘솔 또는 AWS Command Line Interface(AWS CLI)를 사용해 재설정할 수 있습니다.

관리형 노드의 암호 재설정(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 새로운 암호가 필요한 노드 옆에 있는 버튼을 선택합니다.
4. 인스턴스 작업, 암호 재설정을 선택합니다.
5. [사용자 이름(User name)]에 암호를 변경할 사용자 이름을 입력합니다. 해당 노드에 대한 계정을 소유하고 있는 사용자 이름도 될 수 있습니다.
6. 제출을 선택합니다.
7. Enter new password(새 암호 입력) 명령 창의 메시지에 따라 새로운 암호를 지정합니다.

> **Note**
>
> 관리형 노드에 설치된 SSM Agent 버전이 암호 재설정을 지원하지 않을 경우에는 AWS Systems Manager의 기능인 Run Command를 사용해 지원되는 버전을 설치하라는 메시지가 표시됩니다.

관리형 노드의 암호 재설정(AWS CLI)

1. 관리형 노드의 사용자 암호를 재설정하려면 다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

> **Note**
>
> AWS CLI를 사용해 암호를 재설정할 경우 Session Manager 플러그인이 로컬 시스템에 설치되어 있어야 합니다. 자세한 설명은 AWS CLI의 Session Manager 플러그인 설치을 참조하세요.

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. Enter new password(새 암호 입력) 명령 창의 메시지에 따라 새로운 암호를 지정합니다.

관리형 노드의 암호 재설정에 따른 문제 해결

암호를 재설정하면서 발생하는 문제는 대부분 암호 재설정 사전 조건만 충족해도 해결할 수 있습니다. 그 밖에는 다음 정보가 암호 재설정 문제를 해결하는 데 도움이 될 것입니다.

주제

- 관리형 노드 사용 불가
- SSM Agent가 최신 상태가 아닙니다(콘솔)
- 암호 재설정 옵션은 제공되지 않습니다(AWS CLI).
- ssm:SendCommand를 실행할 수 있는 권한이 없습니다
- Session Manager 오류 메시지

관리형 노드 사용 불가

문제: 관리형 인스턴스 콘솔 페이지에서 관리형 노드의 암호를 재설정하려고 하는데 노드가 목록에 없습니다.

- 솔루션: 연결하려는 관리형 노드가 Systems Manager에서 구성되지 않았을 수 있습니다. Systems Manager에서 EC2 인스턴스를 사용하려면 인스턴스에 대해 작업을 수행할 권한을 부여하는 AWS Identity and Access Management(IAM) 인스턴스 프로파일을 인스턴스에 연결해야 합니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성을 참조하세요.

  Systems Manager에서 EC2 시스템이 아닌 시스템을 사용하려면 시스템에서 관리형 노드의 작업을 수행하는 권한을 Systems Manager에 부여하는 IAM 서비스 역할을 생성합니다. 더 자세한 내용은 하이브리드 및 멀티클라우드 환경에서 Systems Manager에 필요한 IAM 서비스 역할 생성을 참조하

세요(온프레미스 서버 및 VM에 대한 Session Manager 지원은 고급 인스턴스 티어에만 제공됩니다. 자세한 내용은 고급 인스턴스 티어 설정 참조).

SSM Agent가 최신 상태가 아닙니다(콘솔)

문제: SSM Agent 버전이 암호 재설정 기능을 지원하지 않는다는 메시지가 표시됩니다.

- 해결 방법: 암호를 재설정하려면 SSM Agent 버전 2.3.668.0 이상이 필요합니다. 콘솔에서 업데이트 SSM Agent를 선택하여 관리형 노드의 에이전트를 업데이트할 수 있습니다.

  SSM Agent의 업데이트된 버전은 Systems Manager에 새 기능이 추가되거나 기존 기능에 업데이트가 발생할 때마다 릴리스됩니다. 최신 버전의 에이전트를 사용하지 못하면 관리형 노드에서 다양한 Systems Manager 기능을 사용하지 못할 수 있습니다. 이러한 이유로 시스템의 SSM Agent 상태를 최신 상태로 유지하는 프로세스를 자동화하는 것이 좋습니다. 자세한 설명은 SSM Agent 업데이트 자동화을 참조하세요. SSM Agent 업데이트에 대해 알림을 수신하려면 GitHub에서 SSM Agent 릴리스 정보 페이지를 구독합니다.

암호 재설정 옵션은 제공되지 않습니다(AWS CLI).

문제: AWS CLI `start-session` 명령을 사용해 관리형 노드에 연결했습니다. SSM 문서 `AWS-PasswordReset`을 지정하고 유효한 사용자 이름을 입력하였지만 암호를 변경할 수 있는 메시지가 표시되지 않습니다.

- 해결 방법: 관리형 노드에 설치된 SSM Agent 버전이 최신 상태가 아닙니다. 암호를 재설정하려면 버전 2.3.668.0 이상이 필요합니다.

  SSM Agent의 업데이트된 버전은 Systems Manager에 새 기능이 추가되거나 기존 기능에 업데이트가 발생할 때마다 릴리스됩니다. 최신 버전의 에이전트를 사용하지 못하면 관리형 노드에서 다양한 Systems Manager 기능을 사용하지 못할 수 있습니다. 이러한 이유로 시스템의 SSM Agent 상태를 최신 상태로 유지하는 프로세스를 자동화하는 것이 좋습니다. 자세한 설명은 SSM Agent 업데이트 자동화을 참조하세요. SSM Agent 업데이트에 대해 알림을 수신하려면 GitHub에서 SSM Agent 릴리스 정보 페이지를 구독합니다.

`ssm:SendCommand`를 실행할 수 있는 권한이 없습니다

문제: 관리형 노드에 연결하여 암호를 변경하려고 하지만 관리형 노드에서 `ssm:SendCommand`을 실행할 권한이 없다는 오류 메시지가 표시됩니다.

- 해결 방법: IAM 정책에 `ssm:SendCommand` 명령을 실행할 수 있는 권한을 추가해야 합니다. 자세한 설명은 태그를 기반으로 Run Command 액세스 제한을 참조하세요.

Session Manager 오류 메시지

문제: Session Manager에 관한 오류 메시지가 표시됩니다.

- 해결 방법: 암호 재설정 기능을 지원하려면 Session Manager가 올바르게 구성되어 있어야 합니다. 자세한 내용은 Session Manager 설정 및 Session Manager 문제 해결 섹션을 참조하세요.

하이브리드 및 멀티클라우드 환경의 관리형 노드 등록 취소

더 이상 AWS Systems Manager를 사용하여 온프레미스 서버, 엣지 디바이스 또는 가상 머신(VM)을 관리하고 싶지 않은 경우에는 등록을 취소할 수 있습니다. 하이브리드 정품 인증 등록을 취소하면 Systems Manager의 관리형 노드 목록에서 제거됩니다. AWS Systems Manager 하이브리드 정품 인증 노드에서 실행되는 에이전트(SSM Agent)는 더는 등록되지 않기 때문에 권한 부여 토큰 새로 고침을 진행할 수 없습니다. SSM Agent는 최대 절전 모드로 전환되고 클라우드에서 Systems Manager에 대한 핑 빈도가 시간당 한 번으로 줄어듭니다.

언제든지 온프레미스 서버, 엣지 디바이스 또는 VM을 다시 등록할 수 있습니다. Systems Manager는 등록 취소된 관리형 노드에 대한 명령 기록을 30일 동안 저장합니다.

다음 절차에서는 Systems Manager 콘솔을 사용하여 하이브리드 정품 인증 노드의 등록을 취소하는 방법을 설명합니다. AWS Command Line Interface를 사용하여 이를 수행하는 방법에 대한 자세한 내용은 deregister-managed-instance를 참조하세요.

하이브리드 정품 인증 노드의 등록을 취소하는 방법(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 등록 취소할 관리형 노드 옆에 있는 버튼을 선택합니다.
4. 노드 태스크, 도구, 이 관리 노드 등록 취소를 선택합니다.
5. 이 관리 노드 등록 취소 대화 상자의 정보를 검토하세요. 이상이 없으면 등록 취소를 선택합니다.

기본 호스트 관리 구성 설정 관리

기본 호스트 관리 구성 설정을 통해 AWS Systems Manager에서 Amazon EC2 인스턴스를 자동으로 관리형 인스턴스로 관리할 수 있습니다. 관리형 인스턴스는 Systems Manager에 사용하도록 구성된 EC2 인스턴스입니다.

Systems Manager로 인스턴스를 관리하는 경우의 이점은 다음과 같습니다.

- Session Manager를 사용하여 안전하게 EC2 인스턴스에 연결합니다.
- Patch Manager를 사용하여 자동 패치 스캔을 수행합니다.
- Systems Manager Inventory를 사용하여 인스턴스에 대한 세부 정보를 봅니다.
- Fleet Manager를 사용하여 인스턴스를 추적하고 관리합니다.
- SSM Agent를 자동으로 최신 상태로 유지합니다.

Fleet Manager, Inventory, Patch Manager 및 Session Manager는 Systems Manager의 기능입니다.

기본 호스트 관리 구성을 통해 AWS Identity and Access Management(IAM) 인스턴스 프로파일을 수동으로 생성하지 않고 EC2 인스턴스를 관리할 수 있습니다. 그 대신에 기본 IAM 역할이 활성화된 AWS 계정 및 AWS 리전의 모든 인스턴스를 관리하는 권한이 Systems Manager에 확보되도록 기본 호스트 관리 구성을 통해 해당 역할이 생성되고 적용됩니다.

제공된 권한이 사용 사례에 충분하지 않은 경우 기본 호스트 관리 구성에서 생성된 기본 IAM 역할에 정책을 추가할 수도 있습니다. 또는 기본 IAM 역할에서 제공하는 모든 기능에 대한 권한이 필요하지 않은 경우 사용자 지정 역할과 정책을 직접 생성할 수 있습니다. 기본 호스트 관리 구성에 대해 선택한 IAM 역할의 모든 변경 사항은 리전 및 계정의 모든 관리형 Amazon EC2 인스턴스에 적용됩니다.

기본 호스트 관리 구성에서 사용하는 정책에 대한 자세한 내용은 AWS 관리형 정책: AmazonSSMManagedEC2InstanceDefaultPolicy 섹션을 참조하세요.

최소 권한 액세스 구현

이 주제의 이 절차는 관리자만 수행할 수 있습니다. 따라서 관리자가 아닌 사용자가 기본 호스트 관리 구성을 구성하거나 수정하지 못하도록 하려면 최소 권한 액세스를 구현하는 것이 좋습니다. 기본 호스트 관리 구성에 대한 액세스를 제한하는 예제 정책을 보려면 이 주제의 기본 호스트 관리 구성의 최소 권한 정책 예제 이후를 참조하세요.

Important

기본 호스트 관리 구성을 사용하여 등록된 인스턴스의 등록 정보는 `var/lib/amazon/ssm` 또는 `C:\ProgramData\Amazon` 디렉터리에 로컬로 저장됩니다. 이러한 디렉토리 또는 해당 파일을 제거하면 인스턴스가 기본 호스트 관리 구성을 사용하여 시스템 관리자에 연결하는 데 필요한 보안 인증을 획득하지 못하게 됩니다. 이러한 경우 IAM 인스턴스 프로파일을 사용하여 인스턴스에 필요한 권한을 제공하거나 인스턴스를 다시 생성해야 합니다.

주제

필수 조건

설정을 활성화하는 AWS 리전 및 AWS 계정에서 기본 호스트 관리 구성을 사용하려면 다음과 같은 요구 사항을 충족해야 합니다.

- 관리할 인스턴스에서 IMDSv2(인스턴스 메타데이터 서비스 버전 2)를 사용해야 합니다.

  기본 호스트 관리 구성은 인스턴스 메타데이터 서비스 버전 1을 지원하지 않습니다. IMDSv2로 전환에 대한 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스 메타데이터 서비스 버전 2 사용으로 전환을 참조하세요.
- 인스턴스를 관리하려면 SSM Agent version 3.2.582.0 이상을 설치해야 합니다.

  인스턴스에 설치된 SSM Agent 버전 확인에 대한 자세한 내용은 SSM Agent 버전 번호 확인 섹션을 참조하세요.

  SSM Agent 업데이트에 대한 자세한 내용은 SSM Agent 자동 업데이트 섹션을 참조하세요.
- 이 주제의 작업을 수행하는 관리자라면 GetServiceSetting, ResetServiceSetting 및 UpdateServiceSetting API 작업에 대한 권한이 있어야 합니다. 또한 `AWSSystemsManagerDefaultEC2InstanceManagementRole` IAM 역할에 대한 `iam:PassRole` 권한이 있어야 합니다. 다음은 이러한 권한을 제공하는 정책 예제입니다. 각 *`example resource placeholder`*를 사용자의 정보로 바꿉니다.

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

- 시스템 관리자를 사용하여 관리할 EC2 인스턴스에 이미 IAM 인스턴스 프로파일이 연결되었으면 `ssm:UpdateInstanceInformation` 작업을 허용하는 권한을 제거해야 합니다. SSM Agent에서는 기본 호스트 관리 구성 권한을 사용하기 전에 인스턴스 프로파일 권한을 사용하려고 시도합니다. IAM 인스턴스 프로파일에서 `ssm:UpdateInstanceInformation` 작업을 허용하면 인스턴스에서 기본 호스트 관리 구성 권한을 사용하지 않습니다.

기본 호스트 관리 구성 설정 활성화

Fleet Manager 콘솔에서 또는 AWS Command Line Interface이나 AWS Tools for Windows PowerShell을 사용하여 기본 호스트 관리 구성을 활성화할 수 있습니다.

Amazon EC2 인스턴스를 이 설정으로 관리할 각 리전에서 기본 호스트 관리 구성을 하나씩 켜야 합니다.

기본 호스트 관리 구성을 켠 후 인스턴스가 다음 절차의 5단계에서 선택한 역할의 자격 증명을 사용하는 데 최대 30분이 걸릴 수 있습니다.

기본 호스트 관리 구성을 활성화하는 방법(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 계정 관리, 기본 호스트 관리 구성을 선택합니다.
4. 기본 호스트 관리 구성 활성화를 켭니다.
5. 인스턴스에 대해 Systems Manager 기능을 활성화하는 데 사용되는 AWS Identity and Access Management(IAM) 역할을 선택합니다. 기본 호스트 관리 구성에서 제공하는 기본 역할을 사용하는 것이 좋습니다. 여기에는 Systems Manager를 사용하여 Amazon EC2 인스턴스를 관리하는 데 필요한 최소 권한 세트가 포함되어 있습니다. 사용자 지정 역할을 사용하는 것을 선호하는 경우 역할의 신뢰 정책에서 Systems Manager를 신뢰할 수 있는 엔터티로 허용해야 합니다.
6. 구성을 선택하여 설정을 완료합니다.

기본 호스트 관리 구성을 활성화하는 방법(명령줄)

1. 다음 신뢰 관계 정책이 포함된 JSON 파일을 로컬 컴퓨터에 생성합니다.

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

2. AWS CLI 또는 Tools for Windows PowerShell을 열고 로컬 시스템의 운영 체제 유형에 따라 다음과 같은 명령 중 하나를 실행하여 계정에서 서비스 역할을 생성합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   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. 다음 명령을 실행하여 새로 생성한 역할에 AmazonSSMManagedEC2InstanceDefaultPolicy 관리형 정책을 연결합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   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. AWS CLI 또는 Tools for Windows PowerShell을 열고 다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   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"
   ```

   명령이 성공해도 출력은 없습니다.

5. 다음 명령을 실행하여 현재 AWS 계정 및 AWS 리전에서 기본 호스트 관리 구성에 대한 현재 서비스 설정을 봅니다.

   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"
```

명령은 다음과 같은 정보를 반환합니다.

```
{
    "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"
    }
}
```

기본 호스트 관리 구성 설정 비활성화

Fleet Manager 콘솔에서 또는 AWS Command Line Interface이나 AWS Tools for Windows PowerShell을 사용하여 기본 호스트 관리 구성을 비활성화할 수 있습니다.

Amazon EC2 인스턴스를 더는 이 구성으로 관리하지 않으려는 각 리전에서 기본 호스트 관리 구성 설정을 하나씩 꺼야 합니다. 한 리전에서 비활성화하면 모든 리전에서 비활성화되는 것이 아닙니다.

기본 호스트 관리 구성을 비활성화하고 Systems Manager에 대한 액세스를 허용하는 Amazon EC2 인스턴스에 인스턴스 프로파일을 연결하지 않으면 Systems Manager에서 해당 인스턴스를 더는 관리하지 않습니다.

기본 호스트 관리 구성을 비활성화하는 방법(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 계정 관리, 기본 호스트 관리 구성을 선택합니다.
4. 기본 호스트 관리 구성 활성화를 끕니다.
5. 구성을 선택하여 기본 호스트 관리 구성을 비활성화합니다.

기본 호스트 관리 구성을 비활성화하는 방법(명령줄)

- AWS CLI 또는 Tools for Windows PowerShell을 열고 다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

  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"
  ```

기본 호스트 관리 구성의 최소 권한 정책 예제

다음과 같은 샘플 정책에서는 소속 조직의 멤버가 계정의 기본 호스트 관리 구성 설정을 변경하지 못하도록 하는 방법을 보여줍니다.

AWS Organizations에 대한 서비스 제어 정책

다음 정책에서는 소속 AWS Organizations의 관리자 멤버가 아닌 멤버가 기본 호스트 관리 구성 설정을 업데이트하지 못하도록 하는 방법을 보여줍니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
{
    "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 보안 주체 정책

다음 정책에서는 소속 AWS Organizations의 IAM 그룹, 역할 또는 사용자가 기본 호스트 관리 구성 설정을 업데이트하지 못하도록 하는 방법을 보여줍니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

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

Remote Desktop을 사용하여 Windows Server 관리형 인스턴스에 연결

Remote Desktop Protocol(RDP)를 사용하여 Windows Server Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에 연결하는 데 AWS Systems Manager의 기능인 Fleet Manager를 사용할 수 있습니다. Fleet Manager NICE DCV로 구동되는 원격 데스크톱에서는 Systems Manager 콘솔에서 직접 Windows Server 인스턴스에 안전하게 연결할 수 있습니다. 단일 브라우저 창에서 4개까지 동시에 연결할 수 있습니다.

현재 Windows Server 2012 RTM 이상이 실행되는 원격 데스크톱에서만 인스턴스를 사용할 수 있습니다. 원격 데스크톱에서는 영어 입력만 지원됩니다.

> **Note**
>
> Fleet Manager 원격 데스크톱은 콘솔 전용 서비스이며 관리형 인스턴스에 대한 명령줄 연결을 지원하지 않습니다. 쉘을 통해 Windows Server 관리형 인스턴스에 연결하려면 AWS Systems Manager의 또 다른 기능인 Session Manager를 사용할 수 있습니다. 자세한 내용은 AWS Systems Manager Session Manager 단원을 참조하십시오.

인스턴스에 Systems Manager와의 상호 작용을 허용하는 AWS Identity and Access Management(IAM) 권한 구성에 대한 자세한 내용은 Systems Manager에 대한 인스턴스 권한 구성을 참조하세요.

주제

- 환경 설정
- 원격 데스크톱 IAM 권한 구성
- 원격 데스크톱 연결 인증

- 원격 연결 기간 및 동시성
- 원격 데스크톱을 사용하여 관리형 노드에 연결

환경 설정

원격 데스크톱 사용을 시작하기 전에 다음과 같은 요구 사항이 환경에서 충족되는지 확인하세요.

- 관리형 노드 구성

  Amazon EC2 인스턴스가 Systems Manager에서 관리형 노드로 구성되어 있는지 확인하세요.

- SSM Agent 최소 버전

  노드에서 SSM Agent 버전 3.0.222.0 이상이 실행되는지 확인하세요. 노드에서 실행되는 에이전트 버전을 확인하는 방법에 대한 내용은 SSM Agent 버전 번호 확인을 참조하세요. SSM Agent 설치 또는 업데이트에 대한 자세한 내용은 SSM Agent 작업 섹션을 참조하세요.

- RDP 포트 구성

  원격 연결을 수락하려면 Windows Server 노드의 Remote Desktop Services 서비스에서 기본 RDP 포트 3389를 사용해야 합니다. AWS에서 제공하는 Amazon Machine Images(AMIs)의 기본 구성입니다. 원격 데스크톱을 사용하려고 인바운드 포트를 명시적으로 열지 않아도 됩니다.

- 키보드 기능을 위한 PSReadLine 모듈 버전

  PowerShell에서 키보드가 제대로 작동하는지 확인하려면 Windows Server 2022가 실행되는 노드에 PSReadLine 모듈 버전 2.2.2 이상이 있는지 확인하세요. 이전 버전이 실행되고 있다면 다음 명령을 사용하여 필요한 버전을 설치할 수 있습니다.

  ```
  Install-Module `
      -Name PSReadLine `
      -Repository PSGallery `-MinimumVersion 2.2.2
  ```

- Session Manager 구성

  원격 데스크톱을 사용할 수 있으려면 Session Manager 설정 전제 조건을 완료해야 합니다. 원격 데스크톱을 사용하여 인스턴스에 연결하면 AWS 계정 및 AWS 리전에 대해 정의된 모든 세션 기본 설정이 적용됩니다. 자세한 내용은 Session Manager 설정 단원을 참조하십시오.

> **Note**
>
> Amazon Simple Storage Service(S3)를 사용하여 Session Manager 활동 로그를 기록하는 경우에는 원격 데스크톱 연결의 bucket_name/Port/stderr에서 다음과 같은 오류가 발생합니다. 이 오류는 예상되는 동작이며 무시해도 됩니다.
>
> ```
> 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>
> ```

원격 데스크톱 IAM 권한 구성

Systems Manager 및 Session Manager에 필요한 IAM 권한 외에도 콘솔에 액세스하는 데 사용하는 사용자 또는 역할이 다음과 같은 작업을 허용해야 합니다.

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

다음은 사용자 또는 역할을 연결하여 다양한 원격 데스크톱 상호 작용 유형을 허용할 수 있는 IAM 정책 예제입니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

EC2 인스턴스 연결을 위한 표준 정책

```
{
    "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": {
                "BoolIfExists": {
                    "ssm:SessionDocumentAccessCheck": "true"
                },
                "ForAnyValue:StringEquals": {
                    "aws:CalledVia": "ssm-guiconnect.amazonaws.com"
                }
```

```
            }
        },
        {
            "Sid": "GuiConnect",
            "Effect": "Allow",
            "Action": [
                "ssm-guiconnect:CancelConnection",
                "ssm-guiconnect:GetConnection",
                "ssm-guiconnect:StartConnection"
            ],
            "Resource": "*"
        }
    ]
}
```

특정 태그가 있는 EC2 인스턴스에 연결하기 위한 정책

> **Note**
>
> 다음 IAM 정책에서 SSMStartSession 섹션에는 ssm:StartSession 작업에 대한 Amazon 리소스 이름(ARN)이 필요합니다. 표시된 대로, 사용자가 지정한 ARN에는 AWS 계정 ID가 필요하지 않습니다. 계정 ID를 지정하면 Fleet Manager에서 AccessDeniedException을 반환합니다.
>
> 예제 정책 아래쪽에 있는 AccessTaggedInstances 섹션에도 ssm:StartSession에 대한 ARN이 필요합니다. 이 ARN의 경우 AWS 계정 ID를 지정합니다.

```
{
    "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": {
                "BoolIfExists": {
                    "ssm:SessionDocumentAccessCheck": "true"
                },
                "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"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS IAM Identity Center 사용자가 EC2 인스턴스에 연결하기 위한 정책

```
{
    "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: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": {
                "BoolIfExists": {
                    "ssm:SessionDocumentAccessCheck": "true"
                },
                "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"
            ],
            "Condition": {
                "BoolIfExists": {
                    "ssm:SessionDocumentAccessCheck": "true"
                }
            }
        },
        {
            "Sid": "GuiConnect",
            "Effect": "Allow",
            "Action": [
                "ssm-guiconnect:CancelConnection",
                "ssm-guiconnect:GetConnection",
                "ssm-guiconnect:StartConnection"
            ],
            "Resource": "*"
        }
    ]
}
```

원격 데스크톱 연결 인증

원격 연결을 설정할 때 Windows 보안 인증 정보 또는 인스턴스와 연결된 Amazon EC2 키 페어(`.pem` 파일)를 사용하여 인증할 수 있습니다. 키 페어에 대한 내용은 Amazon EC2 사용 설명서의 Amazon EC2 키 페어 및 Windows 인스턴스를 참조하세요.

또는 AWS IAM Identity Center을 사용하여 AWS Management Console 인증을 받은 경우 추가 보안 인증 정보를 제공하지 않고 인스턴스에 연결할 수 있습니다. IAM Identity Center를 통한 원격 연결 인증을 허용하는 정책의 예는 원격 데스크톱 IAM 권한 구성를 참조하세요.

시작하기 전 준비 사항

원격 데스크톱을 사용하여 연결을 시작하기 전에 다음과 같은 IAM ID 센터 인증을 사용하는 조건을 참고합니다.

- 원격 데스크톱에서는 IAM ID 센터를 활성화한 해당 AWS 리전에서 노드에 대한 IAM Identity Center 인증이 지원됩니다.
- 원격 데스크톱에서는 IAM Identity Center 사용자 이름이 16자까지 지원됩니다.

- 원격 데스크톱에서는 영숫자와 특수 문자(. - _)로 구성된 IAM Identity Center 사용자 이름이 지원됩니다.

> **Important**
>
> + = , @ 문자가 포함된 IAM Identity Center 사용자 이름은 연결되지 않습니다.
> IAM Identity Center에서는 이러한 문자가 사용자 이름에는 지원되지만 Fleet Manager RDP 연결에는 지원되지 않습니다.

- IAM Identity Center를 사용하는 연결이 인증되면 원격 데스크톱에서는 인스턴스의 로컬 관리자 그룹에 로컬 Windows 사용자가 생성됩니다. 이 사용자는 원격 연결이 종료된 후에도 유지됩니다.
- 원격 데스크톱에서는 Microsoft Active Directory 도메인 컨트롤러인 노드에 대한 IAM Identity Center 인증이 허용되지 않습니다.
- 원격 데스크톱에서는 Active Directory 도메인에 조인된 노드에 IAM Identity Center 인증을 사용하는 것이 허용되지만, 그렇게 하지 않는 것이 좋습니다. 이 인증 방법에서는 도메인에서 부여된 더 제한적인 권한이 재정의될 수도 있는 관리 권한이 사용자에게 부여됩니다.

IAM Identity Center 인증을 지원하는 리전

IAM Identity Center 인증을 사용한 Remote Desktop 연결은 다음 AWS 리전에서 지원됩니다.

- 미국 동부(오하이오)(us-east-2)
- 미국 동부(버지니아 북부)(us-east-1)
- 미국 서부(캘리포니아 북부) (us-west-1)
- 미국 서부(오레곤)(us-west-2)
- 아프리카(케이프타운)(af-south-1)
- 아시아 태평양(홍콩)(ap-east-1)
- 아시아 태평양(뭄바이)(ap-south-1)
- 아시아 태평양(도쿄)(ap-northeast-1)
- 아시아 태평양(서울)(ap-northeast-2)
- 아시아 태평양(오사카) (ap-northeast-3)
- 아시아 태평양(싱가포르)(ap-southeast-1)
- 아시아 태평양(시드니)(ap-southeast-2)
- 아시아 태평양(자카르타) (ap-southeast-3)

- 캐나다(중부)(ca-central-1)
- 유럽(프랑크푸르트)(eu-central-1)
- 유럽(스톡홀름)(eu-north-1)
- 유럽(아일랜드)(eu-west-1)
- 유럽(런던) (eu-west-2)
- 유럽(파리)(eu-west-3)
- 이스라엘(텔아비브) (il-central-1)
- 남아메리카(상파울루)(sa-east-1)
- 유럽(밀라노)(eu-south-1)
- 중동(바레인)(me-south-1)
- AWS GovCloud(미국 동부)(us-gov-east-1)
- AWS GovCloud(미국 서부)(us-gov-west-1)

원격 연결 기간 및 동시성

다음과 같은 조건이 활성 원격 데스크톱 연결에 적용됩니다.

- 연결 기간

  기본적으로 원격 데스크톱 연결은 60분 후에 연결이 해제됩니다. 연결 해제를 방지하려면 연결이 해제되기 전에 세션 갱신을 선택하여 기간 타이머를 재설정할 수 있습니다.
- 연결 제한 시간

  원격 데스크톱 연결의 유휴 상태가 10분을 넘기면 연결이 해제됩니다.
- 동시 연결

  기본적으로 동일한 AWS 계정 및 AWS 리전의 원격 데스크톱 연결을 5개까지 활성화할 수 있습니다. 최대 25개 동시 연결로 서비스 할당량 증가를 요청하려면 Service Quotas 사용 설명서의 할당량 증가 요청을 참조하세요.

원격 데스크톱을 사용하여 관리형 노드에 연결

브라우저의 텍스트 복사/붙여넣기 지원

Google Chrome 및 Microsoft Edge 브라우저를 사용하여 관리형 노드에서 로컬 시스템으로, 로컬 시스템에서 연결된 관리형 노드로 텍스트를 복사하고 붙여넣을 수 있습니다.

Mozilla Firefox 브라우저를 사용하면 관리형 노드의 텍스트를 로컬 시스템으로만 복사하고 붙여넣을 수 있습니다. 로컬 시스템에서 관리형 노드로의 복사는 지원되지 않습니다.

Fleet Manager 원격 데스크톱을 사용하여 관리형 노드에 연결

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 연결하려는 노드를 선택합니다. 확인란 또는 노드 이름을 선택할 수 있습니다.
4. 노드 작업 메뉴에서 원격 데스크톱과 연결을 선택합니다.
5. 원하는 인증 유형(Authentication type) 항목을 선택합니다. 사용자 보안 인증 정보를 선택하는 경우 연결하려는 노드의 Windows 사용자 계정에 대한 사용자 이름과 암호를 입력합니다. 키 페어를 선택하면 다음과 같은 방법 중 하나를 사용하여 인증을 제공할 수 있습니다.

   a. 로컬 파일 시스템에서 인스턴스와 연결된 PEM 키를 선택하려면 로컬 시스템 찾아보기를 선택합니다.

      – 또는 -

   b. PEM 파일의 콘텐츠를 복사하여 제공된 필드에 붙여넣으려면 키 페어 콘텐츠 붙여넣기를 선택합니다.
6. 연결(Connect)을 선택합니다.
7. 선호하는 디스플레이 해상도를 선택하려면 작업 메뉴에서 해상도를 선택한 후 다음 중에서 선택합니다.

   - 자동으로 조정
   - 1920 x 1080
   - 1400 x 900
   - 1366 x 768
   - 800 x 600

   자동으로 조정 옵션에서는 감지된 화면 크기에 따라 해상도가 설정됩니다.

관리형 인스턴스의 Amazon EBS 볼륨 관리

Amazon Elastic Block Store(Amazon EBS)는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스와 함께 사용할 수 있는 블록 스토리지 볼륨을 제공합니다. EBS 볼륨은 형식이 지정되지 않은 원시 블록 디바이스처럼 동작합니다. 이러한 볼륨을 인스턴스에 디바이스로 마운트할 수 있습니다.

AWS Systems Manager의 기능인 Fleet Manager를 사용하여 관리형 인스턴스에서 Amazon EBS 볼륨을 관리할 수 있습니다. 예를 들어 EBS 볼륨을 초기화하고, 파티션을 포맷하며, 볼륨을 마운트해 사용할 수 있습니다.

> **Note**
>
> Fleet Manager는 현재 Windows Server 인스턴스에 대해서만 Amazon EBS 볼륨 관리를 지원합니다.

EBS 볼륨 세부 정보 보기

Fleet Manager를 사용하여 EBS 볼륨의 세부 정보를 보는 방법

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. EBS 볼륨 세부 정보를 보려는 관리형 인스턴스 옆에 있는 버튼을 선택합니다.
4. 세부 정보 보기를 선택합니다.
5. 도구, EBS 볼륨을 선택합니다.
6. EBS 볼륨의 세부 정보를 보려면 볼륨 ID 열에서 해당 ID를 선택합니다.

EBS 볼륨 초기화 및 포맷

Fleet Manager를 사용하여 EBS 볼륨을 초기화하고 포맷하는 방법

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. EBS 볼륨을 초기화, 포맷 및 마운트할 관리형 인스턴스 옆에 있는 버튼을 선택합니다. 디스크가 비어 있는 경우에만 EBS 볼륨을 초기화할 수 있습니다.
4. 세부 정보 보기를 선택합니다.
5. 도구 메뉴에서 EBS 볼륨을 선택합니다.
6. 초기화하고 포맷할 EBS 볼륨 옆에 있는 버튼을 선택합니다.
7. 초기화 및 포맷을 선택합니다.
8. 파티션 스타일에서 EBS 볼륨에 사용할 파티션 스타일을 선택합니다.
9. (선택 사항) 파티션의 드라이브 문자를 선택합니다.

10. (선택 사항) 파티션을 식별할 파티션 이름을 입력합니다.
11. 파티션에 저장된 파일 및 데이터를 정리하는 데 사용할 파일 시스템을 선택합니다.
12. EBS 볼륨을 사용할 수 있게 하려면 확인을 선택합니다. 확인 후 AWS Management Console에서 파티션 구성을 변경할 수는 없지만 SSH 또는 RDP를 통해 인스턴스에 로그인하여 파티션 구성을 변경할 수 있습니다.

파일 시스템 작업

AWS Systems Manager의 기능인 Fleet Manager를 사용하여 관리형 노드에서 파일 시스템 작업을 할 수 있습니다. Fleet Manager를 사용하여 관리형 노드에 연결된 볼륨에 저장된 디렉터리 및 파일 데이터에 대한 정보를 볼 수 있습니다. 예를 들어 디렉터리 및 파일의 이름, 크기, 확장자, 소유자 및 권한을 볼 수 있습니다. Fleet Manager 콘솔에서 최대 10,000줄의 파일 데이터를 텍스트로 미리 볼 수 있습니다. 파일 `tail`에도 이 기능을 사용할 수 있습니다. `tail`을 사용하여 파일 데이터를 볼 때 파일의 마지막 10줄이 처음에 표시됩니다. 새 데이터 행이 파일에 기록되면 뷰가 실시간으로 업데이트됩니다. 결과적으로 콘솔에서 로그 데이터를 검토할 수 있어 문제 해결 및 시스템 관리의 효율성을 향상시킬 수 있습니다. 또한 디렉터리를 만들고 파일 및 디렉터리를 복사, 잘라내기, 붙여넣기, 이름 바꾸기 또는 삭제할 수 있습니다.

정기적인 백업을 생성하거나 관리형 노드에 연결된 Amazon Elastic Block Store(Amazon EBS) 볼륨의 스냅샷을 생성하는 것이 좋습니다. 파일을 복사하거나 잘라내어 붙여넣을 때, 새 파일 또는 디렉터리와 이름이 같은 대상 경로의 기존 파일 및 디렉터리가 대체됩니다. 시스템 파일 및 디렉터리를 바꾸거나 수정하면 심각한 문제가 발생할 수 있습니다. AWS는 이러한 문제가 해결될 수 있다고 보장하지 않습니다. 시스템 파일을 수정할 때는 사용자의 주의가 필요합니다. 모든 파일 및 디렉터리에 변경과 백업이 있는지 확인하는 것은 사용자의 책임입니다. 파일 및 디렉터리를 삭제하거나 바꾸려면 실행 취소할 수 없습니다.

> **Note**
>
> Fleet Manager는 AWS Systems Manager의 기능인 Session Manager를 사용하여 텍스트 미리 보기 및 `tail` 파일을 봅니다. Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에서, 인스턴스 프로파일은 관리형 인스턴스에 Session Manager가 이 기능을 사용할 수 있는 권한을 제공해야 합니다. 인스턴스 프로파일에 Session Manager 권한 추가에 대한 자세한 내용은 기존 IAM 역할에 Session Manager 권한 추가 섹션을 참조하세요. 또한 Fleet Manager 기능을 사용하려면 세션 기본 설정에서 AWS Key Management Service(AWS KMS) 암호화를 설정해야 합니다. Session Manager에 AWS KMS 암호화 사용에 대한 자세한 내용은 세션 데이터의 KMS 키 암호화를 설정하려면(콘솔) 섹션을 참조하세요.

Fleet Manager를 사용하여 파일 시스템을 보려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 보려고 하는 파일 시스템과 관리형 노드의 링크를 선택합니다.
4. 도구, 파일 시스템을 선택합니다.

Fleet Manager를 사용하여 파일의 텍스트 미리 보기를 보려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 미리 보려고 하는 파일과 관리형 노드의 링크를 선택합니다.
4. 도구, 파일 시스템을 선택합니다.
5. 미리 보려는 파일이 포함된 디렉터리의 파일 이름(File name)을 선택합니다.
6. 내용을 미리 보려는 파일 옆에 있는 버튼을 선택합니다.
7. 작업, 텍스트로 미리 보기를 선택합니다.

Fleet Manager으로 파일에 테일을 붙이려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 추적하려는 파일과 관리형 노드의 링크를 선택합니다.
4. 도구, 파일 시스템을 선택합니다.
5. 추적하려는 파일이 포함된 디렉터리의 파일 이름(File name)을 선택합니다.
6. 콘텐츠에 테일을 붙일 파일 옆에 있는 버튼을 선택합니다.
7. 작업, 테일 파일을 선택합니다.

Fleet Manager를 사용하여 파일 또는 디렉터리를 복사하거나 잘라내어 붙여넣으려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 복사하거나 잘라내어 붙여넣을 파일과 관리형 노드의 링크를 선택합니다.

4. 도구, 파일 시스템을 선택합니다.
5. 파일을 복사하거나 잘라내려면 복사하거나 잘라낼 파일이 들어 있는 디렉터리의 파일 이름(File name)을 선택합니다. 디렉터리를 복사하거나 잘라내려면 복사하거나 잘라낼 디렉터리 옆에 있는 버튼을 선택한 다음 8단계로 진행합니다.
6. 복사하거나 잘라낼 파일 옆에 있는 버튼을 선택합니다.
7. 작업(Actions) 메뉴에서 복사(Copy) 또는 잘라내기(Cut)를 선택합니다.
8. 파일 시스템(File system) 보기에서 파일을 붙여넣을 디렉터리 옆에 있는 버튼을 선택합니다.
9. 작업(Actions) 메뉴에서 붙여넣기(Paste)를 선택합니다.

Fleet Manager를 사용하여 파일 또는 디렉터리의 이름을 바꾸려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 이름을 바꿀 파일 또는 디렉터리의 관리형 노드의 링크를 선택합니다.
4. 도구, 파일 시스템을 선택합니다.
5. 파일 이름을 바꾸려면 이름을 바꾸려는 파일이 들어 있는 디렉터리의 파일 이름(File name)을 선택합니다. 디렉터리의 이름을 바꾸려면 이름을 바꿀 디렉터리 옆에 있는 버튼을 선택한 다음 8단계로 진행합니다.
6. 콘텐츠 이름을 바꿀 파일 옆에 있는 버튼을 선택합니다.
7. 작업, 이름 바꾸기를 선택합니다.
8. 파일 이름 필드에 파일의 새 이름을 입력하고 이름 바꾸기를 선택합니다.

Fleet Manager를 사용하여 파일 또는 디렉터리를 삭제하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 삭제하려는 파일 또는 디렉터리의 관리형 노드의 링크를 선택합니다.
4. 도구, 파일 시스템을 선택합니다.
5. 파일을 삭제하려면 삭제하려는 파일이 들어 있는 디렉터리의 파일 이름(File name)을 선택합니다. 디렉터리를 삭제하려면 삭제할 디렉터리 옆에 있는 버튼을 선택한 다음 7단계로 진행합니다.
6. 삭제할 콘텐츠의 파일 옆에 있는 버튼을 선택합니다.
7. 작업, 삭제를 선택합니다.

Fleet Manager으로 디렉터리를 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 디렉터리를 생성할 관리형 노드의 링크를 선택합니다.
4. 도구, 파일 시스템을 선택합니다.
5. 새 디렉터리를 생성할 디렉터리의 파일 이름(File name)을 선택합니다.
6. 디렉터리 생성(Create directory)을 선택합니다.
7. 디렉터리 이름 필드에 새 디렉터리 이름을 입력하고 디렉터리 생성을 선택합니다.

관리형 노드 성능 모니터링

AWS Systems Manager의 기능인 Fleet Manager를 사용하여 관리형 노드에 대한 성능 데이터를 실시간으로 볼 수 있습니다. 성능 데이터는 성능 카운터에서 검색됩니다.

Fleet Manager에서 사용할 수 있는 성능 카운터는 다음과 같습니다.

- CPU 사용률
- 디스크 입/출력(I/O) 활용도
- 네트워크 트래픽
- 메모리 사용량

> **Note**
>
> Fleet Manager는 AWS Systems Manager의 기능인 Session Manager를 사용하여 성능 데이터를 검색합니다. Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에서, 인스턴스 프로파일은 관리형 인스턴스에 Session Manager가 이 기능을 사용할 수 있는 권한을 제공해야 합니다. 인스턴스 프로파일에 Session Manager 권한 추가에 대한 자세한 내용은 기존 IAM 역할에 Session Manager 권한 추가 섹션을 참조하세요. 또한 Fleet Manager 기능을 사용하려면 세션 기본 설정에서 AWS Key Management Service(AWS KMS) 암호화를 설정해야 합니다. Session Manager에 대해 AWS KMS 암호화 설정에 대한 자세한 내용은 세션 데이터의 KMS 키 암호화를 설정하려면(콘솔) 섹션을 참조하세요.

Fleet Manager에서 성능 데이터를 보려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 성능을 모니터링하려는 관리형 노드 옆에 있는 버튼을 선택합니다.
4. 세부 정보 보기를 선택합니다.
5. 도구, 성능 카운터를 선택합니다.

프로세스 작업

AWS Systems Manager의 기능인 Fleet Manager를 사용하여 관리형 인스턴스에서 프로세스 작업을 할 수 있습니다. Fleet Manager를 사용하여, 프로세스에 대한 정보를 볼 수 있습니다. 예를 들어 프로세스의 핸들 및 스레드 외에 CPU 사용률 및 메모리 사용량을 확인할 수 있습니다. Fleet Manager와 함께 콘솔에서 프로세스를 시작하고 종료할 수 있습니다.

> **Note**
>
> Fleet Manager은(는) AWS Systems Manager의 기능인 Session Manager을(를) 사용하여 프로세스 데이터를 검색합니다. Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에서, 인스턴스 프로파일은 관리형 인스턴스에 Session Manager가 이 기능을 사용할 수 있는 권한을 제공해야 합니다. 인스턴스 프로파일에 Session Manager 권한 추가에 대한 자세한 내용은 기존 IAM 역할에 Session Manager 권한 추가 섹션을 참조하세요. 또한 Fleet Manager 기능을 사용하려면 세션 기본 설정에서 AWS Key Management Service(AWS KMS) 암호화를 설정해야 합니다. Session Manager에 대해 AWS KMS 암호화 설정에 대한 자세한 내용은 세션 데이터의 KMS 키 암호화를 설정하려면(콘솔) 섹션을 참조하세요.

Fleet Manager와 함께 프로세스 세부 사항 확인

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 프로세스를 보려는 파일의 인스턴스 링크를 선택합니다.
4. 도구, 프로세스를 선택합니다.

Fleet Manager와 함께 프로세스 시작

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 프로세스를 시작하려는 파일의 인스턴스 링크를 선택합니다.
4. 도구, 프로세스를 선택합니다.
5. 새 프로세스 시작(Start new process)을 선택합니다.
6. 프로세스 이름 또는 전체 경로 필드에 프로세스 이름 또는 실행 파일의 전체 경로를 입력합니다.
7. (선택 사항) 작업 디렉터리 필드에 프로세스를 실행할 디렉터리 경로를 입력합니다.

Fleet Manager와 함께 프로세스 종료

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 프로세스를 시작하려는 파일의 인스턴스 링크를 선택합니다.
4. 도구, 프로세스를 선택합니다.
5. 종료할 프로세스 옆에 있는 버튼을 선택합니다.
6. 작업, 프로세스 종료 또는 작업, 프로세스 트리 종료를 선택합니다.

> **Note**
>
> 프로세스 트리를 종료하면 해당 프로세스를 사용하는 모든 프로세스와 응용 프로그램도 종료됩니다.

관리형 노드의 로그 보기

AWS Systems Manager의 기능인 Fleet Manager를 사용하여 관리형 노드에 저장된 로그 데이터를 볼 수 있습니다. Windows 관리형 노드의 경우 콘솔에서 Windows 이벤트 로그를 보고 세부 정보를 복사할 수 있습니다. 이벤트 검색에 도움이 되도록 [이벤트 수준(Event level)], [이벤트 ID(Event ID)], [이벤트 소스(Event source)] 및 [생성 시간(Time created)]별로 Windows 이벤트 로그를 필터링합니다. 파일 시스템을 보는 절차를 사용하여 다른 로그 데이터를 볼 수도 있습니다. Fleet Manager를 사용하여 파일 시스템 보기에 대한 자세한 내용은 파일 시스템 작업 섹션을 참조하세요.

Fleet Manager를 사용하여 Windows 이벤트 로그를 보려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 이벤트 로그를 보려는 관리형 노드 옆에 있는 버튼을 선택합니다.
4. 세부 정보 보기를 선택합니다.
5. 도구, Windows 이벤트 로그를 선택합니다.
6. 보려는 이벤트가 포함된 [로그 이름(Log name)]을 선택합니다.
7. 보려는 [로그 이름(Log name)] 옆에 있는 버튼을 선택한 다음 [이벤트 보기(View events)]를 선택합니다.
8. 보려는 이벤트 옆에 있는 버튼을 선택한 다음 이벤트 세부 정보 보기(View event details)를 선택합니다.
9. (옵션) [JSON으로 복사(Copy as JSON)]를 선택하여 이벤트 세부 정보를 클립보드에 복사합니다.

관리형 노드의 OS 사용자 계정 관리

AWS Systems Manager의 기능인 Fleet Manager를 사용하여 관리형 노드의 운영 체제(OS) 사용자 계정을 관리할 수 있습니다. 예를 들어 사용자와 그룹을 생성하고 삭제할 수 있습니다. 또한 그룹 멤버십, 사용자 역할 및 상태와 같은 세부 정보를 볼 수 있습니다.

> **Important**
>
> Fleet Manager는 다양한 사용자 관리 작업을 위해 AWS Systems Manager의 기능인 Run Command와 Session Manager를 사용합니다. 결과적으로 사용자는 운영 체제 사용자 계정에 다른 방법으로는 불가능한 권한을 부여할 수 있습니다. 이는 AWS Systems Manager Agent(SSM Agent)가 루트 권한(Linux) 또는 SYSTEM 권한(Windows Server)을 사용하여 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에서 실행되기 때문입니다. SSM Agent를 통해 루트 수준 명령에 대한 액세스 제한에 대한 자세한 내용은 SSM Agent를 통한 루트 수준 명령에 대한 액세스 제한 섹션을 참조하세요. 이 기능에 대한 액세스를 제한하려면 정의한 작업에만 액세스를 허용하는 AWS Identity and Access Management(IAM) 정책을 사용자에 대해 생성하는 것이 좋습니다. Fleet Manager에 대한 IAM 정책 생성에 대한 자세한 내용은 1단계: Fleet Manager 권한이 있는 IAM 정책 생성 섹션을 참조하세요.

사용자 또는 그룹 생성

> **Note**
>
> Fleet Manager는 Session Manager를 사용하여 새 사용자의 암호를 설정합니다. Amazon EC2 인스턴스에서, 관리형 인스턴스에 연결된 인스턴스 프로파일은 Session Manager가 이 기능을 사용할 수 있는 권한을 제공해야 합니다. 인스턴스 프로파일에 Session Manager 권한 추가에 대한 자세한 내용은 기존 IAM 역할에 Session Manager 권한 추가 섹션을 참조하세요. 또한 Fleet Manager 기능을 사용하려면 세션 기본 설정에서 AWS Key Management Service(AWS KMS) 암호화를 설정해야 합니다. Session Manager에 AWS KMS 암호화 사용에 대한 자세한 내용은 세션 데이터의 KMS 키 암호화를 설정하려면(콘솔) 섹션을 참조하세요.

Fleet Manager를 사용하여 OS 사용자 계정을 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 새 사용자를 생성할 관리형 노드 옆에 있는 버튼을 선택합니다.
4. 세부 정보 보기를 선택합니다.
5. 도구, 사용자 및 그룹을 선택합니다.
6. [사용자(Users)] 탭을 선택한 다음 [사용자 생성(Create user)]을 선택합니다.
7. 새 사용자의 [이름(Name)] 값을 입력합니다.
8. (권장) [암호 설정(Set password)] 옆에 있는 확인란을 선택합니다. 절차가 끝나면 새 사용자의 암호를 입력하라는 메시지가 표시됩니다.
9. [사용자 생성(Create user)]을 선택합니다. 새 사용자의 암호를 생성하기 위해 확인란을 선택한 경우 암호 값을 입력하고 [완료(Done)]를 선택하라는 메시지가 나타납니다. 지정한 암호가 관리형 노드의 로컬 또는 도메인 정책에 지정된 요구 사항을 충족하지 않으면 오류가 반환됩니다.

Fleet Manager를 사용하여 OS 그룹을 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 새 그룹을 생성할 관리형 노드 옆에 있는 버튼을 선택합니다.
4. 세부 정보 보기를 선택합니다.

5. 도구, 사용자 및 그룹을 선택합니다.
6. 그룹 탭을 선택한 다음 그룹 생성을 선택합니다.
7. 새 그룹의 [이름(Name)] 값을 입력합니다.
8. (옵션) 새 그룹에 대한 [설명(Description)] 값을 입력합니다.
9. (옵션) 새 그룹의 [그룹 멤버(Group members)]에 추가할 사용자를 선택합니다.
10. [그룹 생성(Create group)]을 선택합니다.

사용자 또는 그룹 멤버십 업데이트

Fleet Manager를 사용하여 새 그룹에 OS 사용자 계정을 추가하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 업데이트하려는 사용자 계정이 있는 관리형 노드 옆에 있는 버튼을 선택합니다.
4. 세부 정보 보기를 선택합니다.
5. 도구, 사용자 및 그룹을 선택합니다.
6. 사용자 탭을 선택합니다.
7. 업데이트하려는 사용자 옆에 있는 버튼을 선택합니다.
8. 작업, 그룹에 사용자 추가를 선택합니다.
9. [그룹에 추가(Add to group)]에서 사용자를 추가할 그룹을 선택합니다.
10. [그룹에 사용자 추가(Add user to group)]를 선택합니다.

Fleet Manager를 사용하여 OS 그룹의 멤버십을 편집하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 업데이트하려는 그룹이 있는 관리형 노드 옆에 있는 버튼을 선택합니다.
4. 세부 정보 보기를 선택합니다.
5. 도구, 사용자 및 그룹을 선택합니다.
6. 그룹 탭을 선택합니다.
7. 업데이트하려는 그룹 옆에 있는 버튼을 선택합니다.

8. 작업, 그룹 수정을 선택합니다.
9. [그룹 멤버(Group members)]에서 추가하거나 제거하려는 사용자를 선택합니다.
10. [그룹 수정(Modify group)]을 선택합니다.

사용자 또는 그룹 삭제

Fleet Manager를 사용하여 OS 사용자 계정을 삭제하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 삭제하려는 사용자 계정이 있는 관리형 노드 옆에 있는 버튼을 선택합니다.
4. 세부 정보 보기를 선택합니다.
5. 사용자 및 그룹을 선택합니다.
6. 사용자 탭을 선택합니다.
7. 삭제하려는 사용자 옆에 있는 버튼을 선택합니다.
8. 작업, 로컬 사용자 삭제를 선택합니다.

Fleet Manager를 사용하여 OS 그룹을 삭제하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 삭제하려는 그룹이 있는 관리형 노드 옆에 있는 버튼을 선택합니다.
4. 세부 정보 보기를 선택합니다.
5. 도구, 사용자 및 그룹을 선택합니다.
6. [그룹(Group)] 탭을 선택합니다.
7. 업데이트하려는 그룹 옆에 있는 버튼을 선택합니다.
8. 작업, 로컬 그룹 삭제를 선택합니다.

관리형 노드의 Windows 레지스트리 관리

AWS Systems Manager의 기능인 Fleet Manager를 사용하여 Windows Server 관리형 노드에서 레지스트리를 관리할 수 있습니다. Fleet Manager 콘솔에서 레지스트리 항목과 값을 생성, 복사, 업데이트 및 삭제할 수 있습니다.

> **Important**
>
> 레지스트리를 수정하기 전에 레지스트리 백업을 생성하거나 관리형 노드에 연결된 루트 Amazon Elastic Block Store(Amazon EBS) 볼륨의 스냅샷을 생성하는 것이 좋습니다. 레지스트리를 잘못 수정하면 심각한 문제가 발생할 수 있습니다. 이러한 문제로 인해 운영 체제를 다시 설치하거나 스냅샷에서 관리형 노드의 루트 볼륨을 복원해야 할 수 있습니다. AWS는 이러한 문제가 해결될 수 있다고 보장하지 않습니다. 레지스트리를 수정할 때는 사용자의 주의가 필요합니다. 모든 레지스트리 변경과 백업이 있는지 확인하는 것은 사용자의 책임입니다.

Windows 레지스트리 키 또는 항목 생성

Fleet Manager를 사용하여 Windows 레지스트리 키를 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 레지스트리 키를 생성할 관리형 노드 옆에 있는 버튼을 선택합니다.
4. 세부 정보 보기를 선택합니다.
5. 도구, Windows 레지스트리를 선택합니다.
6. [레지스트리 이름(Registry name)]을 선택하여 새 레지스트리 키를 만들 Hive를 선택합니다.
7. 생성, 레지스트리 키 생성을 선택합니다.
8. 새 키를 생성할 레지스트리 항목 옆에 있는 버튼을 선택합니다.
9. [레지스트리 키 생성(Create registry key)]을 선택합니다.
10. 새 레지스트리 키의 [이름(Name)] 값을 입력하고 [제출(Submit)]을 선택합니다.

Fleet Manager를 사용하여 Windows 레지스트리 항목을 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 레지스트리 항목을 생성할 인스턴스 옆에 있는 버튼을 선택합니다.
4. 세부 정보 보기를 선택합니다.
5. 도구, Windows 레지스트리를 선택합니다.
6. [레지스트리 이름(Registry name)]을 선택하여 새 레지스트리 항목을 생성할 Hive 및 후속 레지스트리 키를 선택합니다.

7. 생성, 레지스트리 항목 생성을 선택합니다.
8. 새 레지스트리 항목의 [이름(Name)] 값을 입력합니다.
9. 레지스트리 항목에 대해 생성할 값의 [유형(Type)]을 선택합니다. 레지스트리 값 유형에 대한 자세한 내용은 레지스트리 값 유형을 참조하세요.
10. 새 레지스트리 항목의 [값(Value)]을 입력합니다.

Windows 레지스트리 항목 업데이트

Fleet Manager를 사용하여 Windows 레지스트리 항목을 업데이트하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 레지스트리 항목을 업데이트할 관리형 노드 옆에 있는 버튼을 선택합니다.
4. 세부 정보 보기를 선택합니다.
5. 도구, Windows 레지스트리를 선택합니다.
6. [레지스트리 이름(Registry name)]을 선택하여 업데이트할 Hive 및 후속 레지스트리 키를 선택합니다.
7. 업데이트할 레지스트리 항목 옆에 있는 버튼을 선택합니다.
8. 작업, 레지스트리 항목 업데이트를 선택합니다.
9. 레지스트리 항목의 [값(Value)]을 새로 입력합니다.
10. 업데이트를 선택합니다.

Windows 레지스트리 항목 또는 키 삭제

Fleet Manager를 사용하여 Windows 레지스트리 키를 삭제하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 레지스트리 키를 생성할 관리형 노드 옆에 있는 버튼을 선택합니다.
4. 도구, Windows 레지스트리를 선택합니다.
5. [레지스트리 이름(Registry name)]을 선택하여 삭제할 Hive 및 후속 레지스트리 키를 선택합니다.
6. 삭제할 레지스트리 키 옆에 있는 버튼을 선택합니다.

7. 작업, 레지스트리 키 삭제를 선택합니다.

Fleet Manager를 사용하여 Windows 레지스트리 항목을 삭제하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 레지스트리 항목을 생성할 관리형 노드 옆에 있는 버튼을 선택합니다.
4. 세부 정보 보기를 선택합니다.
5. 도구, Windows 레지스트리를 선택합니다.
6. [레지스트리 이름(Registry name)]을 선택하여 삭제할 항목이 들어 있는 Hive 및 후속 레지스트리 키를 선택합니다.
7. 삭제할 레지스트리 항목 옆에 있는 버튼을 선택합니다.
8. 작업, 레지스트리 항목 삭제를 선택합니다.

Red Hat Knowledgebase 액세스

Red Hat 고객이라면 AWS Systems Manager의 기능인 Fleet Manager를 사용하여 Knowledgebase에 액세스할 수 있습니다. Red Hat Enterprise Linux(RHEL) 인스턴스를 실행하거나 AWS의 RHEL 서비스를 이용한다면 Red Hat 고객으로 간주됩니다. Knowledgebase 포털에는 Red Hat 라이선스 고객에게만 제공되는 커뮤니티 지원을 위한 바이너리, 지식 공유 및 토론 포럼이 포함되어 있습니다.

Knowledgebase 포털에 액세스하려면 Systems Manager와 Fleet Manager에 대한 필수 AWS Identity and Access Management(IAM) 권한 외에도 콘솔에 액세스하는 데 사용하는 사용자 또는 역할은 `rhelkb:GetRhelURL` 작업을 허용해야 합니다.

Red Hat Knowledgebase 포털에 액세스

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. Red Hat Knowledgebase 포털에 연결하는 데 사용하려는 RHEL 인스턴스를 선택하세요.
4. 계정 관리, Red Hat Knowledgebase 액세스를 선택하여 Red Hat Knowledgebase 페이지를 엽니다.

완벽하게 지원되는 RHEL 워크로드를 실행하기 위해 AWS에서 RHEL을 사용하는 경우, AWS 자격 증명을 사용해 Red Hat 웹사이트를 통해 Red Hat Knowledgebase에 액세스할 수도 있습니다.

관리형 노드 가용성 문제 해결

Run Command, Distributor, Session Manager와 같은 여러 AWS Systems Manager 기능의 경우 작업을 실행할 관리형 노드를 수동으로 선택하도록 할 수 있습니다. 이러한 경우 노드를 수동으로 선택하도록 지정한 후, 작업을 실행할 수 있는 관리형 노드 목록이 표시됩니다.

이 주제에서는 실행 중인 것으로 확인된 관리형 노드가 Systems Manager의 관리형 노드 목록에 포함되지 않은 이유를 진단합니다.

노드가 Systems Manager에서 관리되고 관리형 노드 목록에서 사용 가능하게 하려면 다음 3가지 기본 요구 사항을 충족해야 합니다.

- SSM Agent는 지원되는 운영 체제가 있는 노드에 설치되어 실행 중이어야 합니다.

  > **Note**
  >
  > 일부 AWS 관리형 Amazon Machine Images(AMIs)는 SSM Agent가 사전 설치된 인스턴스를 시작하도록 구성됩니다. SSM Agent를 미리 설치하도록 사용자 정의 AMI를 구성할 수도 있습니다. 자세한 내용은 SSM Agent가 사전 설치된 상태로 AMIs 검색 단원을 참조하십시오.

- Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에서 AWS Identity and Access Management(IAM) 인스턴스에 대한 인스턴스 프로파일을 첨부해야 합니다. 인스턴스 프로파일을 사용하면 인스턴스가 Systems Manager 서비스와 통신할 수 있습니다. 인스턴스에 인스턴스 프로파일을 할당하지 않으면 일반적인 시나리오가 아닌 하이브리드 정품 인증을 사용하여 인스턴스를 등록합니다.
- SSM Agent는 서비스에 자신을 등록하기 위해 Systems Manager 엔드포인트에 연결할 수 있어야 합니다. 그 이후에는 서비스에서 관리형 노드를 사용할 수 있어야 하며, 이는 인스턴스 상태를 확인하기 위해 5분마다 신호를 보내는 서비스에 의해 확인됩니다.
- 관리형 노드의 상태가 30일 이상 `Connection Lost`한 후에는 해당 노드가 Fleet Manager 콘솔에 더 이상 나열되지 않을 수 있습니다. 목록으로 복원하려면 연결이 끊긴 문제를 해결해야 합니다.

관리형 노드가 실행 중인지 확인한 후 다음 명령을 사용하여 SSM Agent가 Systems Manager 서비스에 등록되었는지 확인할 수 있습니다. 이 명령은 성공적으로 등록될 때까지 결과를 반환하지 않습니다.

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
```

등록이 완료되고 이제 관리형 노드를 Systems Manager 작업에 사용할 수 있는 경우 명령은 다음과 유사한 결과를 반환합니다.

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

등록이 아직 완료되지 않았거나 실패한 경우 명령은 다음과 유사한 결과를 반환합니다.

```
{
    "InstanceAssociationStatusInfos": []
```

```
}
```

명령이 5분 정도 후에도 결과를 반환하지 않으면 다음 정보를 사용하여 관리형 노드의 문제를 해결하는 데 도움이 됩니다.

주제

- 해결 방법 1: SSM Agent가 관리형 노드에 설치되어 실행 중인지 확인
- 해결 방법 2: 인스턴스에 대해 IAM 인스턴스 프로파일이 지정되었는지 확인(EC2 인스턴스만 해당)
- 해결 방법 3: 서비스 엔드포인트 연결 확인
- 해결 방법 4: 대상 운영 체제 지원 확인
- 해결 방법 5: Amazon EC2 인스턴스와 동일한 AWS 리전에서 작업 중인지 확인
- 솔루션 6: 관리형 노드의 SSM Agent에 적용한 프록시 구성 확인
- 솔루션 7: 관리형 인스턴스에 TLS 인증서 설치
- ssm-cli를 사용하여 관리형 노드 가용성 문제 해결

해결 방법 1: SSM Agent가 관리형 노드에 설치되어 실행 중인지 확인

SSM Agent의 최신 버전이 관리형 노드에 설치되어 실행되고 있는지 확인합니다.

SSM Agent가 관리형 노드에 설치되어 실행 중인지 확인하려면 SSM Agent 상태 확인 및 에이전트 시작 섹션을 참조하세요.

관리형 노드에 SSM Agent를 설치하거나 다시 설치하려면 다음 주제를 참조하세요.

- Linux용 EC2 인스턴스에 수동으로 SSM Agent 설치 및 제거
- 하이브리드 Linux 노드에 SSM Agent를 설치하는 방법
- SSM Agent용 EC2 인스턴스에 수동으로 Windows Server 설치 및 제거
- 하이브리드 Windows 노드에 SSM Agent를 설치하는 방법

해결 방법 2: 인스턴스에 대해 IAM 인스턴스 프로파일이 지정되었는지 확인(EC2 인스턴스만 해당)

Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스가 Systems Manager API와 통신하도록 허용하는 AWS Identity and Access Management(IAM) 인스턴스 프로파일로 구성되어 있는지 확인합니다. 또한 사용자가 시스템 관리자 API와 통신할 수 있는 IAM 신뢰 정책이 있는지도 확인합니다.

> **Note**
>
> 온프레미스 서버, 엣지 디바이스 및 가상 머신(VM)은 인스턴스 프로파일 대신 IAM 서비스 역할을 사용합니다. 자세한 내용은 하이브리드 및 멀티클라우드 환경에서 Systems Manager에 필요한 IAM 서비스 역할 생성을 참조하세요.

필요한 권한이 있는 인스턴스 프로파일이 EC2 인스턴스에 연결되어 있는지 확인

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 Instances(인스턴스)를 선택합니다.
3. 인스턴스 프로파일을 확인할 인스턴스를 선택합니다.
4. 하단 창의 [설명(Description)] 탭에서 [IAM 역할(IAM role)]을 찾고 역할의 이름을 선택합니다.
5. 인스턴스 프로파일에 대한 역할 [요약(Summary)] 페이지의 [권한(Permissions)] 탭에서 `AmazonSSMManagedInstanceCore`가 [권한 정책(Permissions policies)] 아래에 나열되어 있는지 확인합니다.

   대신 사용자 정의 정책을 사용하는 경우 `AmazonSSMManagedInstanceCore`와 동일한 권한을 제공하는지 확인합니다.

   콘솔에서 `AmazonSSMManagedInstanceCore` 열기

   Systems Manager용 인스턴스 프로파일에 연결할 수 있는 다른 정책에 대한 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성을 참조하세요.

해결 방법 3: 서비스 엔드포인트 연결 확인

인스턴스가 Systems Manager 서비스 엔드포인트에 연결되어 있는지 확인합니다. 이러한 연결은 Systems Manager에 VPC 엔드포인트를 생성 및 구성하거나 서비스 엔드포인트에 대한 HTTPS(포트 443) 아웃바운드 트래픽을 허용함으로써 이루어집니다.

Amazon EC2 인스턴스에서는, AWS 리전에 대한 Systems Manager 서비스 엔드포인트는 Virtual Private Cloud(VPC) 구성에서 아웃바운드 트래픽을 허용하는 경우 인스턴스를 등록하는 데 사용됩니다. 그러나 인스턴스가 시작된 VPC 구성에서 아웃바운드 트래픽을 허용하지 않고 퍼블릭 서비스 엔드포인트에 대한 연결을 허용하도록 이 구성을 변경할 수 없는 경우, 대신 VPC에 대한 인터페이스 엔드포인트를 구성해야 합니다.

자세한 내용은 Systems Manager용 VPC 엔드포인트를 사용하여 EC2 인스턴스의 보안 개선을 참조하세요.

해결 방법 4: 대상 운영 체제 지원 확인

선택한 작업이 나열될 것으로 예상되는 관리형 노드 유형에서 실행될 수 있는지 확인합니다. 일부 Systems Manager 작업은 Windows 인스턴스 또는 Linux 인스턴스만 대상으로 할 수 있습니다. 예를 들어 Systems Manager(SSM) `AWS-InstallPowerShellModule` 및 `AWS-ConfigureCloudWatch`는 Windows 인스턴스에서만 실행할 수 있습니다. [명령 실행(Run a command)] 페이지에서 이러한 문서 중 하나를 선택하고 [수동으로 인스턴스 선택(Choose instances manually)]을 선택하면 Windows 인스턴스만 나열되고 선택할 수 있습니다.

해결 방법 5: Amazon EC2 인스턴스와 동일한 AWS 리전에서 작업 중인지 확인

Amazon EC2 인스턴스는 미국 동부(오하이오) 리전(us-east-2) 또는 유럽(아일랜드) 리전(eu-west-1)과 같은 특정 AWS 리전에서 생성 및 사용 가능합니다. 작업하려는 Amazon EC2 인스턴스와 동일한 AWS 리전에서 작업하고 있는지 확인합니다. 자세한 내용은 AWS Management Console 시작하기에서 리전 선택을 참조하세요.

솔루션 6: 관리형 노드의 SSM Agent에 적용한 프록시 구성 확인

관리형 노드의 SSM Agent에 적용한 프록시 구성이 올바른지 확인합니다. 프록시 구성이 올바르지 않으면 노드가 필요한 서비스 엔드포인트에 연결할 수 없거나 Systems Manager가 관리형 노드의 운영 체제를 잘못 식별할 수 있습니다. 자세한 내용은 SSM Agent를 구성하여 Linux 노드에 프록시 사용 및 Windows Server 인스턴스에 프록시를 사용하도록 SSM Agent 구성 단원을 참조하세요.

솔루션 7: 관리형 인스턴스에 TLS 인증서 설치

AWS Systems Manager와(과) 함께 사용하는 각 관리형 인스턴스에 TLS(전송 계층 보안) 인증서를 설치해야 합니다. AWS 서비스은(는) 이러한 인증서를 사용하여 다른 AWS 서비스에 대한 호출을 암호화합니다.

Amazon Machine Image(AMI)에서 생성된 각 Amazon EC2 인스턴스에는 TLS 인증서가 기본적으로 이미 설치되어 있습니다. 대부분의 최신 운영 체제에는 신뢰 저장소에 Amazon Trust Services CA의 필수 TLS 인증서가 포함되어 있습니다.

인스턴스에 필요한 인증서가 설치되어 있는지 확인하려면 인스턴스의 운영 체제에 따라 다음 명령을 실행합니다. 관리형 인스턴스가 위치한 URL ## 부분을 AWS 리전로 교체하세요.

Linux & macOS

```
curl -L https://ssm.region.amazonaws.com
```

Windows

```
Invoke-WebRequest -Uri https://ssm.region.amazonaws.com
```

명령이 UnknownOperationException 에러를 반환합니다. 대신 SSL/TLS 오류 메시지가 나타나면 필요한 인증서가 설치되지 않을 수 있습니다.

필수 Amazon Trust Services CA 인증서가 기본 운영 체제, Amazon에서 제공하지 않는 AMIs에서 생성된 인스턴스 또는 자체 온프레미스 서버와 VM에 설치되어 있지 않은 경우 Amazon Trust Services의 인증서를 설치하여 활성화하거나 AWS Certificate Manager(ACM)를 사용하여 지원되는 통합 서비스에 대한 인증서를 생성하고 관리해야 합니다.

각 관리형 인스턴스에 다음과 같은 TLS(전송 계층 보안) 인증서가 하나 설치되어 있어야 합니다.

- Amazon Root CA 1
- Starfield Services Root Certificate Authority - G2
- Starfield Class 2 인증 기관

ACM 사용에 대한 자세한 내용은 AWS Certificate Manager User Guide를 참조하세요.

GPO(그룹 정책 객체)로 인증서를 관리하는 컴퓨팅 환경이라면 이러한 인증서 중 하나가 포함되도록 그룹 정책을 구성해야 합니다.

Amazon Root 및 Starfield 인증서에 대한 자세한 내용은 블로그 게시물 How to Prepare for AWS's Move to Its Own Certificate Authority를 참조하세요.

`ssm-cli`를 사용하여 관리형 노드 가용성 문제 해결

`ssm-cli`는 SSM Agent 설치에 포함된 독립 실행형 명령줄 도구입니다. SSM Agent 3.1.501.0 이상을 컴퓨터에 설치하는 경우 해당 컴퓨터에서 `ssm-cli` 명령을 실행할 수 있습니다. 이러한 명령의 출력은 시스템이 AWS Systems Manager에서 관리되는 Amazon EC2 인스턴스 또는 비 EC2 시스템의 최소 요건 충족 여부를 판단하는 데 도움이 되며, 따라서 Systems Manager의 관리형 노드 목록에 추가 됩니다. (SSM Agent 버전 3.1.501.0은 2021년 11월에 출시되었습니다.)

최소 요구 사항

Amazon EC2 인스턴스 또는 비 EC2 시스템이 AWS Systems Manager에서 관리되고 관리형 노드 목록에서 사용 가능하려면 다음 세 가지 기본 요구 사항을 충족해야 합니다.

- 지원되는 운영 체제가 있는 시스템에 SSM Agent가 설치되어 실행 중이어야 합니다.

  EC2의 일부 AWS 관리형 Amazon Machine Images(AMIs)는 SSM Agent가 사전 설치된 인스턴스를 시작하도록 구성됩니다. SSM Agent를 미리 설치하도록 사용자 정의 AMI를 구성할 수도 있습니다. 자세한 내용은 SSM Agent가 사전 설치된 상태로 AMIs 검색 단원을 참조하십시오.
- Systems Manager 서비스와 통신하는 데 필요한 권한을 제공하는 AWS Identity and Access Management(IAM) 인스턴스 프로파일(EC2 인스턴스의 경우) 또는 IAM 서비스 역할(비 EC2 시스템의 경우)이 시스템에 연결되어 있어야 합니다.
- SSM Agent는 서비스에 자신을 등록하기 위해 Systems Manager 엔드포인트에 연결할 수 있어야 합니다. 그 이후에는 서비스에서 관리형 노드를 사용할 수 있어야 하며, 이는 관리형 노드의 상태를 확인하기 위해 5분마다 신호를 보내는 서비스에 의해 확인됩니다.

ssm-cli에서 사전 구성된 명령

실행 중인 것으로 확인된 시스템이 Systems Manager의 관리형 노드 목록에 포함되지 않은 이유를 진단하기 위해 필수 정보를 수집하는 사전 구성된 명령이 포함되어 있습니다. 이러한 명령은 `get-diagnostics` 옵션을 지정할 때 실행됩니다.

시스템에서 다음 명령을 실행하여 관리형 노드 가용성 문제 해결에 도움이 되는 `ssm-cli`를 사용합니다.

Linux & macOS

```
ssm-cli get-diagnostics --output table
```

Windows

Windows Server 시스템에서 명령을 실행하기 전에 `C:\Program Files\Amazon\SSM` 디렉터리로 이동해야 합니다.

```
ssm-cli.exe get-diagnostics --output table
```

PowerShell

Windows Server 시스템에서 명령을 실행하기 전에 `C:\Program Files\Amazon\SSM` 디렉터리로 이동해야 합니다.

```
.\ssm-cli.exe get-diagnostics --output table
```

명령을 통해 다음과 비슷한 테이블로 출력을 반환합니다.

> **Note**
>
> `ssmmessages`, `s3`, `kms`, `logs` 및 `monitoring` 엔드포인트와의 연결성 확인은 Amazon Simple Storage Service(Amazon S3) 또는 Amazon CloudWatch Logs, 그리고 AWS Key Management Service(AWS KMS) 암호화에 로그할 수 있는 Session Manager과 같은 추가적인 옵션 기능을 위한 것입니다.

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
                                      #
#########################################################################################
```

다음 표에는 ssm-cli에서 수행하는 각 검사에 대한 추가 세부 정보가 나와 있습니다.

ssm-cli 진단 검사

Check]를 선택합니다	Details
Amazon EC2 인스턴스 메타데이터 서비스	관리형 노드가 메타데이터 서비스에 도달할 수 있는지를 나타냅니다. 실패한 테스트는 로컬 경로, 프록시 또는 OS(운영 체제) 방화벽 및 프록시 구성으로 인해 발생할 수 있는 `http://169.254.169.254` 에 대한 연결 문제를 나타냅니다.
하이브리드 인스턴스 등록	SSM Agent가 하이브리드 정품 인증을 사용하여 등록되었는지를 나타냅니다.
`ssm` 엔드포인트에 연결성	노드가 TCP 포트 443에서 Systems Manager의 서비스 엔드포인트에 도달할 수 있는지를 나타냅니다. 실패한 테스트는 노드가 있는 `https://ssm.` *`region`*`.amazonaws.com` 에 따라 AWS 리전에 대한 연결 문제를 나타냅니다. 연결 문제는 보안 그룹, 네트워크 액세스 제어 목록, 라우팅 테이블 또는 OS 방화벽 및 프록시를 포함한 VPC 구성으로 인해 발생할 수 있습니다.
`ec2messages` 엔드포인트에 연결성	노드가 TCP 포트 443에서 Systems Manager의 서비스 엔드포인트에 도달할 수 있는지를 나타냅니다. 실패한 테스트는 노드가 있는 `https://ec2messages.` *`region`*`.amazonaws.com` 에 따라 AWS 리전에 대한 연결 문제를 나타냅니다. 연결 문제는 보안 그룹, 네트워크 액세스 제어 목록, 라우팅 테이블 또는 OS 방화벽 및 프록시를 포함한 VPC 구성으로 인해 발생할 수 있습니다.
`ssmmessages` 엔드포인트에 연결성	노드가 TCP 포트 443에서 Systems Manager의 서비스 엔드포인트에 도달할 수 있는지를 나타냅니다. 실패한 테스트는 노드가 있는 `https://ssmmessage`

Check]를 선택합니다	Details
	s. *region*.amazonaws.com 에 따라 AWS 리전에 대한 연결 문제를 나타냅니다. 연결 문제는 보안 그룹, 네트워크 액세스 제어 목록, 라우팅 테이블 또는 OS 방화벽 및 프록시를 포함한 VPC 구성으로 인해 발생할 수 있습니다.
s3 엔드포인트에 연결성	노드가 TCP 포트 443에서 Amazon Simple Storage Service의 서비스 엔드포인트에 도달할 수 있는지를 나타냅니다. 실패한 테스트는 노드가 있는 https://s3. *region*.amazonaws.com 에 따라 AWS 리전에 대한 연결 문제를 나타냅니다. 노드를 관리형 노드 목록에 표시하기 위해 이 엔드포인트에 연결할 필요는 없습니다.
kms 엔드포인트에 연결성	노드가 TCP 포트 443에서 AWS Key Management Service의 서비스 엔드포인트에 도달할 수 있는지를 나타냅니다. 실패한 테스트는 노드가 있는 https://kms. *region*.amazonaws.com 에 따라 AWS 리전에 대한 연결 문제를 나타냅니다. 노드를 관리형 노드 목록에 표시하기 위해 이 엔드포인트에 연결할 필요는 없습니다.
logs 엔드포인트에 연결성	노드가 TCP 포트 443에서 Amazon CloudWatch Logs의 서비스 엔드포인트에 도달할 수 있는지를 나타냅니다. 실패한 테스트는 노드가 있는 https://logs. *region*.amazonaws.com 에 따라 AWS 리전에 대한 연결 문제를 나타냅니다. 노드를 관리형 노드 목록에 표시하기 위해 이 엔드포인트에 연결할 필요는 없습니다.

Check]를 선택합니다	Details
`monitoring` 엔드포인트에 연결성	노드가 TCP 포트 443에서 Amazon CloudWatch의 서비스 엔드포인트에 도달할 수 있는지를 나타냅니다. 실패한 테스트는 노드가 있는 `https://monitoring.region.amazonaws.com` 에 따라 AWS 리전에 대한 연결 문제를 나타냅니다. 노드를 관리형 노드 목록에 표시하기 위해 이 엔드포인트에 연결할 필요는 없습니다.
AWS 자격 증명	시스템에 연결된 IAM 인스턴스 프로파일(EC2 인스턴스의 경우) 또는 IAM 서비스 역할(비 EC2 시스템의 경우)별로 필요한 보안 인증 정보가 SSM Agent에 있는지를 나타냅니다. 실패한 테스트는 인스턴스에 연결된 IAM 인스턴스 프로파일 또는 IAM 서비스 역할이 없거나 Systems Manager에 필요한 권한이 포함되어 있지 않음을 나타냅니다.
에이전트 서비스	SSM Agent 서비스가 실행 중인지 여부와 서비스가 Linux 또는 macOS용 루트 또는 Windows Server용 SYSTEM으로 실행 중인지 여부를 나타냅니다. 실패한 테스트는 SSM Agent 서비스가 실행되고 있지 않거나 루트 또는 SYSTEM으로 실행되고 있지 않음을 나타냅니다.
프록시 구성	프록시를 사용하도록 SSM Agent가 구성되었는지를 나타냅니다.
Sysprep 이미지 상태(Windows만 해당)	노드의 Sysprep 상태를 나타냅니다. Sysprep 상태가 `IMAGE_STATE_COMPLETE` 이외의 값이면 SSM Agent가 노드에서 시작되지 않습니다.
SSM Agent 버전	사용 가능한 최신 SSM Agent 버전이 설치되어 있는지를 나타냅니다.

AWS Systems Manager Compliance

AWS Systems Manager의 기능인 Compliance를 사용하여 관리형 노드 플릿에 대해 패치 규정 준수 및 구성 일관성을 스캔할 수 있습니다. 여러 AWS 계정 및 리전의 데이터를 수집하여 집계한 후 규정을 준수하지 않는 특정 리소스로 드릴다운할 수 있습니다. 기본적으로 Compliance는 Patch Manager에 패치에 대한 현재 준수 데이터를 표시하고 State Manager에 연결을 표시합니다. Patch Manager와 State Manager도 모두 AWS Systems Manager의 기능입니다. Compliance를 시작하려면 Systems Manager 콘솔을 엽니다. 탐색 창에서 Compliance를 선택합니다.

Patch Manager에서 패치 규정 준수 데이터를 AWS Security Hub로 보낼 수 있습니다. Security Hub에서는 우선순위가 높은 보안 알림 및 규정 준수 상태를 포괄적으로 파악할 수 있습니다. 또한 플릿의 패치 상태를 모니터링합니다. 자세한 내용은 Patch Manager와 AWS Security Hub 통합 단원을 참조하십시오.

Compliance에는 다음과 같은 추가적인 장점 및 기능이 있습니다.

- AWS Config를 사용하여 Patch Manager 패치 데이터와 State Manager 연결에 대한 규정 준수 기록 및 변경 사항 추적을 봅니다.
- IT 또는 비즈니스 요구 사항에 따라 규정 준수를 사용자 지정하여 자체 규정 준수 유형을 만들 수 있습니다.
- AWS Systems Manager, State Manager 또는 Amazon EventBridge의 또 다른 기능인 Run Command를 사용하여 문제를 해결합니다.
- Amazon Athena와 Amazon QuickSight로 데이터를 포팅하여 플릿 전체의 보고서를 생성합니다.

EventBridge 지원

이 Systems Manager 기능은 Amazon EventBridge 규칙에서 이벤트 유형으로 지원됩니다. 자세한 내용은 Amazon EventBridge로 Systems Manager 이벤트 모니터링 및 참조: Systems Manager용 Amazon EventBridge 이벤트 패턴 및 유형 섹션을 참조하세요.

Chef InSpec 통합

Systems Manager는 Chef InSpec과 통합됩니다. InSpec은 GitHub 또는 Amazon Simple Storage Service(S3)에 육안 판독 프로파일을 생성할 수 있는 오픈 소스 런타임 프레임워크입니다. Systems Manager를 사용하여 규정 준수 스캔을 수행하고 준수 및 비준수 노드를 확인할 수 있습니다. 자세한 내용은 Systems Manager Compliance와 함께 Chef InSpec 프로파일 사용 단원을 참조하십시오.

요금

Compliance는 추가 비용 없이 제공됩니다. 사용하는 AWS 리소스에 대해서만 비용을 지불하는 것입니다.

내용

Compliance 시작하기

AWS Systems Manager의 기능인 Compliance를 시작하려면 다음 태스크를 수행합니다.

작업	자세한 정보
Compliance는 Patch Manager의 패치 데이터와 State Manager의 연결을 사용합니다. Patch Manager와 State Manager도 모두 AWS Systems Manager의 기능입니다. Compliance는 Systems Manager를 사용하여 관리형 노드의 사용자 정의 규정 준수 유형도 사용합니다. 하이브리드 및 멀티클라우드 환경의 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 및 비 EC2 시스템에 대한 설정 요구 사항을 완료했는지 확인합니다.	AWS Systems Manager 설정
관리형 노드의 Systems Manager SSM Agent(SSM Agent)를 최신 버전으로 업데이트합니다.	SSM Agent 작업
패치 규정 준수를 모니터링하려는 경우 Patch Manager를 구성했는지 확인합니다. Compliance가 패치 규정 준수 데이터를 표시하려면 Patch	AWS Systems Manager Patch Manager

작업	자세한 정보
Manager를 사용하여 패치 작업을 수행해야 합니다.	
연결 규정 준수를 모니터링하려는 경우 State Manager 연결을 생성했는지 확인합니다. Compliance가 연결 규정 준수 데이터를 표시하려면 연결을 생성해야 합니다.	AWS Systems Manager State Manager
(선택 사항) 규정 준수 이력 및 변경 사항 추적을 보기 위해 시스템을 구성합니다.	규정 준수 구성 이력 및 변경 사항 추적 보기
(선택 사항) 사용자 지정 규정 준수 유형을 만듭니다.	Compliance 시연(AWS CLI)
(옵션) 리소스 데이터 동기화를 생성하여 대상 Amazon Simple Storage Service(Amazon S3) 버킷에서 모든 규정 준수 데이터를 집계합니다.	Compliance의 리소스 데이터 동기화 생성

Compliance의 리소스 데이터 동기화 생성

AWS Systems Manager의 리소스 데이터 동기화 기능을 사용하여 모든 관리형 노드의 규정 준수 데이터를 대상 Amazon Simple Storage Service(Amazon S3) 버킷으로 전송할 수 있습니다. 동기화를 생성할 때 여러 AWS 계정, AWS 리전 및 하이브리드 및 멀티클라우드 환경에서 관리형 노드를 지정할 수 있습니다. 그러면 새로운 규정 준수 데이터가 수집될 때마다 리소스 데이터 동기화가 중앙 데이터를 자동으로 업데이트합니다. 모든 규정 준수 데이터가 대상 S3 버킷에 저장되면 Amazon Athena와 Amazon QuickSight 같은 서비스를 사용하여 수집한 데이터에 대한 쿼리를 실행하거나 분석할 수 있습니다. Compliance의 리소스 데이터 동기화 구성은 일회성 작업입니다.

Compliance의 리소스 데이터 동기화는 AWS Management Console에서 다음 절차에 따라 생성합니다.

리소스 데이터 동기화를 위해 S3 버킷 생성 및 구성(콘솔)

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.

2. 집계된 규정 준수 데이터를 저장할 버킷을 생성합니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 버킷 생성을 참조하세요. 버킷 이름과 버킷을 생성한 AWS 리전을 따로 적어둡니다.
3. 버킷을 열고 권한 탭과 버킷 정책을 차례로 선택합니다.
4. 다음 버킷 정책을 복사하여 정책 편집기에 붙여 넣습니다. 이때 DOC-EXAMPLE-BUCKET 및 *Account-ID*를 사용자가 생성한 S3 버킷 이름 및 유효한 AWS 계정 ID로 바꿉니다. 원할 경우 *Bucket-Prefix*를 Amazon S3 접두사(하위 디렉터리) 이름으로 바꿉니다. 접두사를 생성하지 않았다면 정책의 ARN에서 *Bucket-Prefix*/를 제거하십시오.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "SSMBucketPermissionsCheck",
            "Effect": "Allow",
            "Principal": {
                "Service": "ssm.amazonaws.com"
            },
            "Action": "s3:GetBucketAcl",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
        },
        {
            "Sid": " SSMBucketDelivery",
            "Effect": "Allow",
            "Principal": {
                "Service": "ssm.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET/Bucket-Prefix/*/
accountid=Account_ID_number/*"],
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "bucket-owner-full-control"
                }
            }
        }
    ]
}
```

리소스 데이터 동기화를 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. [계정 관리(Account management)], [리소스 데이터 동기화(Resource Data Syncs)]를 선택한 다음 [리소스 데이터 동기화 생성(Create resource data sync)]을 선택합니다.
4. [동기화 이름(Sync name)] 필드에 동기화 구성 이름을 입력합니다.
5. [버킷 이름(Bucket name)] 필드에 절차를 시작할 때 생성한 Amazon S3 버킷 이름을 입력합니다.
6. (옵션) 버킷 접두사(Bucket prefix) 필드에 S3 버킷 접두사(하위 디렉터리)의 이름을 입력합니다.
7. 생성한 S3 버킷이 현재 AWS 리전에 위치하면 버킷 리전 필드에서 이 리전을 선택합니다. 버킷이 다른 AWS 리전에 위치하면 [다른 리전(Another region)]을 선택하고 리전 이름을 입력합니다.

   > **Note**
   >
   > 동기화와 대상 S3 버킷이 서로 다른 리전에 위치하는 경우에는 데이터 전송 요금이 부과될 수 있습니다. 자세한 내용은 Amazon S3 요금을 참조하세요.

8. 생성(Create)을 선택합니다.

Compliance 작업

AWS Systems Manager의 기능인 Compliance는 Patch Manager 패치의 패치 상태와 State Manager의 연결에 대한 데이터를 수집하고 보고합니다. Patch Manager와 State Manager도 모두 AWS Systems Manager의 기능입니다. 또한 Compliance는 관리형 노드에 대해 지정한 사용자 정의 규정 준수 유형에 대해 보고합니다. 이 섹션에는 이러한 규정 준수 유형 각각에 대한 세부 정보와 Systems Manager 규정 준수 데이터를 보는 방법이 나와 있습니다. 또한 규정 준수 이력과 변경 사항 추적을 확인하는 방법에 정보도 다룹니다.

> **Note**
>
> Systems Manager는 Chef InSpec과 통합됩니다. InSpec은 GitHub 또는 Amazon Simple Storage Service(S3)에 육안 판독 프로파일을 생성할 수 있는 오픈 소스 런타임 프레임워크입니다. Systems Manager를 사용하여 규정 준수 검사를 수행하고 준수 및 비준수 인스턴스를 확인할 수 있습니다. 자세한 내용은 Systems Manager Compliance와 함께 Chef InSpec 프로파일 사용 단원을 참조하십시오.

패치 규정 준수 정보

Patch Manager를 사용하여 인스턴스에 패치를 설치하고 나면 콘솔 또는 AWS Command Line Interface(AWS CLI) 명령에 대한 응답이나 해당하는 Systems Manager API 작업을 통해 규정 준수 상태에 대한 정보를 즉시 볼 수 있습니다.

패치 규정 준수 상태 값에 대한 자세한 내용은 패치 규정 준수 상태 값 이해 섹션을 참조하세요.

State Manager 연결 규정 준수 정보

State Manager 연결을 하나 이상 생성하고 나면 콘솔 또는 AWS CLI 명령에 대한 응답이나 해당하는 Systems Manager API 작업을 통해 규정 준수 상태에 대한 정보를 즉시 볼 수 있습니다. 연결의 경우 Compliance는 `Compliant` 또는 `Non-compliant`의 상태와 연결에 할당된 심각도(예: `Critical` 또는 `Medium`)를 보여줍니다.

사용자 지정 규정 준수 정보

관리형 노드에 규정 준수 메타데이터를 할당할 수 있습니다. 그러면 이 메타데이터를 규정 준수 보고를 위해 다른 규정 준수 데이터와 함께 집계할 수 있습니다. 예를 들어 여러분의 회사가 관리형 노드에서 소프트웨어 X의 버전 2.0, 3.0, 4.0을 실행한다고 가정하겠습니다. 이 회사는 버전 2.0과 3.0을 실행하는 인스턴스가 규정을 미준수하여 버전 4.0으로 표준화하려고 합니다. PutComplianceItems API 작업을 사용하여 소프트웨어 X의 이전 버전을 실행하는 관리형 노드를 명시적으로 기록할 수 있습니다. AWS CLI, AWS Tools for Windows PowerShell 또는 SDK를 사용하여 규정 준수 메타데이터만 할당할 수 있습니다. 다음 CLI 명령 샘플은 관리형 인스턴스에 규정 준수 메타데이터를 할당하고, 필요한 형식 `Custom:`으로 규정 준수 유형을 지정합니다. 각 *`example resource placeholder`*를 사용자의 정보로 바꿉니다.

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 파라미터는 ManagedInstance만 지원합니다. 관리형 AWS IoT Greengrass 코어 디바이스에 사용자 지정 규정 준수를 추가하는 경우 ManagedInstance의 ResourceType를 식별해야 합니다.

그러면 규정 준수 관리자는 요약을 보거나 규정 준수 또는 미준수 관리형 노드에 대한 보고서를 만들 수 있습니다. 관리형 노드 하나에 최대 10개의 사용자 지정 규정 준수 유형을 할당할 수 있습니다.

사용자 지정 규정 준수 유형을 만들고 규정 준수 데이터를 보는 방법의 예는 Compliance 시연(AWS CLI)을 참조하십시오.

현재의 규정 준수 데이터 보기

이 섹션에서는 AWS CLI를 사용하거나 Systems Manager 콘솔에서 규정 준수 데이터를 보는 방법을 설명합니다. 패치 및 연결의 규정 준수 이력과 변경 사항 추적을 확인하는 방법은 규정 준수 구성 이력 및 변경 사항 추적 보기 섹션을 참조하세요.

주제

- 현재의 규정 준수 데이터 보기(콘솔)
- 현재의 규정 준수 데이터 보기(AWS CLI)

현재의 규정 준수 데이터 보기(콘솔)

Systems Manager 콘솔에서 규정 준수 데이터를 보려면 다음 절차를 따릅니다.

Systems Manager 콘솔에서 현재의 규정 준수 보고서를 보려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Compliance를 선택합니다.

3. 규정 준수 대시보드 필터링 섹션에서 규정 준수 데이터를 필터링하는 옵션을 선택합니다. 규정 준수 리소스 요약 섹션에는 선택한 필터를 기반으로 규정 준수 데이터 개수가 표시됩니다.
4. 자세한 내용을 보기 위해 리소스로 드릴다운하려면 아래로 스크롤해 리소스에 대한 세부 정보 개요 영역을 선택하고 관리형 노드의 ID를 선택합니다.
5. 인스턴스 ID 또는 이름(Name) 세부 정보 페이지에서 구성 규정 준수(Configuration compliance) 탭을 선택하여 관리형 노드에 대한 자세한 구성 규정 준수 보고서를 확인합니다.

> **Note**
>
> 규정 준수 문제 수정에 대한 자세한 내용은 EventBridge를 사용하여 규정 준수 문제 해결을 참조하십시오.

현재의 규정 준수 데이터 보기(AWS CLI)

AWS CLI에서 다음 AWS CLI 명령을 사용하여 패치 적용, 연결, 사용자 지정 규정 준수 유형에 대한 규정 준수 데이터의 요약을 볼 수 있습니다.

list-compliance-summaries

지정한 필터에 따라 규정 준수 및 규정 미준수 연결 상태의 요약 개수를 반환합니다. (API: ListComplianceSummaries)

list-resource-compliance-summaries

리소스 수준 요약 개수를 반환합니다. 요약에는 지정한 필터 조건에 따라 규정 준수 및 규정 미준수 상태와 세부적인 규정 준수 항목 심각도 개수에 대한 정보가 포함되어 있습니다. (API: ListResourceComplianceSummaries)

다음 AWS CLI 명령을 사용하여 패치 적용에 대한 그 밖의 규정 준수 데이터를 볼 수 있습니다.

describe-patch-group-state

패치 그룹에 대한 높은 수준의 집계된 패치 규정 준수 상태를 반환합니다. (API: DescribePatchGroupState)

describe-instance-patch-states-for-patch-group

지정된 패치 그룹의 인스턴스에 대한 높은 수준의 패치 상태를 반환합니다. (API: DescribeInstancePatchStatesForPatchGroup)

> **Note**
>
> AWS CLI를 사용하여 패치를 구성하고 패치 규정 준수 세부 정보를 보는 방법은 자습서: 서버 환경에 패치 적용(AWS CLI) 섹션을 참조하세요.

규정 준수 구성 이력 및 변경 사항 추적 보기

Systems Manager Compliance는 관리형 노드에 대한 현재 패치 및 연결의 규정 준수 데이터를 표시합니다. AWS Config를 사용하여 패치와 연결의 규정 준수 기록 및 변경 사항 추적을 확인할 수 있습니다. AWS Config는 AWS 계정에 있는 AWS 리소스의 구성을 자세히 보여줍니다. 이러한 보기에는 리소스 간에 어떤 관계가 있는지와 리소스가 과거에 어떻게 구성되었는지도 포함되므로, 시간이 지나면서 구성과 관계가 어떻게 변하는지 확인할 수 있습니다. 패치 및 연결의 규정 준수 이력과 변경 사항 추적을 확인하려면 AWS Config에서 다음 리소스를 설정해야 합니다.

- `SSM:PatchCompliance`
- `SSM:AssociationCompliance`

AWS Config에서 이러한 특정 리소스를 선택하고 구성하는 방법에 대한 자세한 내용은 AWS Config Developer Guide의 Selecting Which Resources AWS Config Records를 참조하세요.

> **Note**
>
> AWS Config 요금에 대한 자세한 내용은 요금을 참조하세요.

Compliance의 리소스 데이터 동기화 삭제

더 이상 AWS Systems Manager Compliance를 사용하여 규정 준수 데이터를 보고 싶지 않은 경우, Compliance 데이터 수집에 사용되는 리소스 데이터 동기화를 삭제하는 것이 좋습니다.

Compliance 리소스 데이터 동기화를 삭제하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 계정 관리(Account management)에서 리소스 데이터 동기화(Resource data sync)를 선택합니다.
4. 목록에서 동기화를 선택합니다.

> **Important**
>
> Compliance에 사용되는 동기화를 선택해야 합니다. Systems Manager는 여러 기능에 대한 리소스 데이터 동기화를 지원합니다. 잘못된 동기화를 선택하면 Systems Manager Explorer 또는 Systems Manager Inventory의 데이터 집계가 중단될 수 있습니다.

5. 삭제를 선택합니다.
6. 데이터가 저장된 Amazon Simple Storage Service(Amazon S3) 버킷을 삭제합니다. S3 버킷 삭제에 대한 자세한 내용은 버킷 삭제를 참조하세요.

EventBridge를 사용하여 규정 준수 문제 해결

AWS Systems Manager의 기능인 Run Command를 사용하여 패치 및 연결 규정 준수 문제를 신속하게 해결할 수 있습니다. 인스턴스 또는 AWS IoT Greengrass 코어 디바이스 ID 또는 태그를 타겟팅할 수 있고 `AWS-RunPatchBaseline` 문서 또는 `AWS-RefreshAssociation` 문서를 실행할 수 있습니다. 연결을 새로 고치거나 패치 기준을 다시 실행해도 규정 준수 문제가 해결되지 않으면 연결, 패치 기준 또는 인스턴스 구성을 조사하여 Run Command 작업이 문제를 해결하지 못한 이유를 파악해야 합니다.

패치에 대한 자세한 내용은 AWS Systems Manager Patch Manager 및 `AWS-RunPatchBaseline` SSM 문서 정보 섹션을 참조하세요.

연결에 대한 자세한 내용은 Systems Manager에서 연결 작업 섹션을 참조하세요.

명령 실행에 대한 자세한 내용은 AWS Systems Manager Run Command을 참조하십시오.

Compliance를 EventBridge 이벤트 대상으로 지정

Systems Manager Compliance 이벤트에 대한 응답으로 작업을 수행하도록 Amazon EventBridge를 구성할 수도 있습니다. 예를 들어 하나 이상의 관리형 노드가 중요한 패치 업데이트를 설치하지 못했거나 바이러스 백신 소프트웨어를 설치하는 연결을 실행하지 못한 경우 Compliance 이벤트가 발생할 때 `AWS-RunPatchBaseline` 문서 또는 `AWS-RefreshAssocation` 문서를 실행하도록 EventBridge를 구성할 수 있습니다.

다음 절차에 따라 Compliance를 EventBridge 이벤트 대상으로 구성합니다.

Compliance를 EventBridge 이벤트 대상으로 구성하려면(콘솔)

1. https://console.aws.amazon.com/events/에서 Amazon EventBridge 콘솔을 엽니다.

2. 탐색 창에서 규칙을 선택합니다.
3. 규칙 생성을 선택합니다.
4. 규칙에 대해 이름과 설명을 입력하십시오.

   규칙은 동일한 AWS 리전과 동일한 이벤트 버스의 다른 규칙과 동일한 이름을 가질 수 없습니다.
5. 이벤트 버스에서 이 규칙과 연결할 이벤트 버스를 선택합니다. 이 규칙이 자신의 AWS 계정에서 오는 일치하는 이벤트에 응답하도록 하려면 default(기본)를 선택합니다. 계정의 AWS 서비스이(가) 이벤트를 출력하면 항상 계정의 기본 이벤트 버스로 이동합니다.
6. 규칙 유형에서 이벤트 패턴이 있는 규칙을 선택합니다.
7. 다음을 선택합니다.
8. 이벤트 소스(Event source)에서 AWS 이벤트 또는 EventBridge 파트너 이벤트(Events or EventBridge partner events)를 선택합니다.
9. 이벤트 패턴(Event pattern)섹션에서 이벤트 패턴 양식(Event pattern form)을 선택합니다.
10. 이벤트 소스에서 AWS 서비스를 선택합니다.
11. AWS service(서비스)에서 Systems Manager를 선택합니다.
12. [이벤트 유형(Event type)] 필드에서 [Configuration Compliance]를 선택합니다.
13. Specific detail type(s)(특정 상세 유형)에서 Configuration Compliance State Change(구성 준수 상태 변경)을 선택합니다.
14. 다음을 선택합니다.
15. 대상 유형에서 AWS서비스를 선택합니다.
16. Select a target(대상 선택)에서 Systems Manager Run Command을(를) 선택합니다.
17. [문서(Document)] 목록에서 대상이 호출될 때 실행할 Systems Manager 문서(SSM 문서)를 선택합니다. 예를 들어 규정 미준수 패치 이벤트의 경우 `AWS-RunPatchBaseline`을 선택하고, 규정 미준수 연결 이벤트의 경우 `AWS-RefreshAssociation`을 선택합니다.
18. 나머지 필드 및 파라미터의 정보를 지정합니다.

    > **Note**
    >
    > 필수 필드 및 파라미터는 이름 옆에 별표(*)가 있습니다. 대상을 생성하려면 각 필수 파라미터 또는 필드의 값을 지정해야 합니다. 그렇지 않으면 시스템에서 규칙이 생성되지만 실행되지 않습니다.
19. 다음을 선택합니다.

20. (선택 사항) 규칙에 대해 하나 이상의 태그를 입력하십시오. 자세한 내용은 Amazon EventBridge User Guide의 Tagging Your Amazon EventBridge Resources를 참조하세요.
21. 다음을 선택합니다.
22. 규칙의 세부 정보를 검토하고 규칙 생성을 선택합니다.

Compliance 시연(AWS CLI)

다음 절차에서는 AWS Command Line Interface(AWS CLI)를 통해 AWS Systems Manager PutComplianceItems API 작업을 호출하여 리소스에 사용자 정의 규정 준수 메타데이터를 할당하는 과정을 안내합니다. 다음 시연에서처럼 이 API 작업을 사용하여 관리형 노드에 패치 또는 연결 규정 준수 메타데이터를 수동으로 할당할 수도 있습니다. 사용자 정의 규정 준수에 대한 자세한 내용은 사용자 지정 규정 준수 정보를 참조하십시오.

관리형 인스턴스에 사용자 정의 규정 준수 메타데이터를 할당하려면(AWS CLI)

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.
2. 다음 명령을 실행하여 사용자 지정 규정 준수 메타데이터를 관리형 노드에 할당합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다. ResourceType 파라미터는 ManagedInstance의 값만 지원합니다. 관리형 AWS IoT Greengrass 코어 디바이스에 사용자 정의 규정 준수 메타데이터를 할당하더라도 이 값을 지정합니다.

   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. 이전 단계를 반복하여 하나 이상의 노드에 사용자 지정 규정 준수 메타데이터를 추가로 할당합니다. 다음 명령을 사용하여 관리형 노드에 패치 또는 연결 규정 준수 메타데이터를 수동으로 할당할 수도 있습니다.

   연결 규정 준수 메타데이터

   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
```

   패치 규정 준수 메타데이터

   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. 다음 명령을 실행하여 특정 관리형 노드의 규정 준수 항목 목록을 봅니다. 필터를 사용하여 특정 규정 준수 데이터로 드릴다운합니다.

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
```

다음 예제에서는 필터와 함께 이 명령을 사용하는 방법을 보여 줍니다.

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. 다음 명령을 실행하여 규정 준수 상태 요약을 봅니다. 필터를 사용하여 특정 규정 준수 데이터로 드릴다운합니다.

```
aws ssm list-resource-compliance-summaries --filters One or more filters.
```

다음 예제에서는 필터와 함께 이 명령을 사용하는 방법을 보여 줍니다.

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. 다음 명령을 실행하여 규정 준수 유형에 대한 규정 준수 및 규정 미준수 리소스의 요약 개수를 봅니다. 필터를 사용하여 특정 규정 준수 데이터로 드릴다운합니다.

```
aws ssm list-compliance-summaries --filters One or more filters.
```

다음 예제에서는 필터와 함께 이 명령을 사용하는 방법을 보여 줍니다.

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 Inventory

AWS Systems Manager Inventory는 AWS 컴퓨팅 환경에 대한 가시성을 제공합니다. Inventory를 사용하여 관리형 노드에서 메타데이터를 수집할 수 있습니다. 이 메타데이터를 중앙 Amazon Simple Storage Service(Amazon S3) 버킷에 저장한 후 기본 제공 도구를 사용하여 데이터를 쿼리하고 어느 노드에서 소프트웨어 정책이 요구하는 소프트웨어 및 구성을 실행 중인지, 어느 노드를 업데이트해야 하는지 빠르게 확인할 수 있습니다. 원클릭 절차를 사용하여 모든 관리형 노드에 대해 인벤토리를 구성할

수 있습니다. 또한 여러 AWS 리전 및 AWS 계정로부터 인벤토리 데이터를 구성하고 볼 수 있습니다. 인벤토리를 시작하려면 Systems Manager 콘솔을 엽니다. 탐색 창에서 [Inventory]를 선택합니다.

Systems Manager Inventory가 수집하는 미리 구성된 메타데이터 형식이 요구 사항을 충족하지 않을 경우 사용자 정의 Systems Manager Inventory를 생성할 수 있습니다. 사용자 지정 인벤토리는 사용자가 제공하고 특정 디렉터리에서 관리형 노드에 추가하는 정보를 포함하는 단순한 JSON 파일입니다. Systems Manager Inventory가 데이터를 수집할 때 이 사용자 정의 인벤토리 데이터를 캡처합니다. 예를 들어 대규모 데이터 센터를 운영하는 경우 각 서버의 랙 위치를 사용자 지정 인벤토리로 지정할 수 있습니다. 그러면 다른 인벤토리 데이터를 볼 때 랙 공간 데이터를 볼 수 있습니다.

> **Important**
>
> Systems Manager Inventory는 관리형 노드에서만 메타데이터를 수집합니다. Inventory는 독점 정보 또는 데이터에 액세스하지 않습니다.

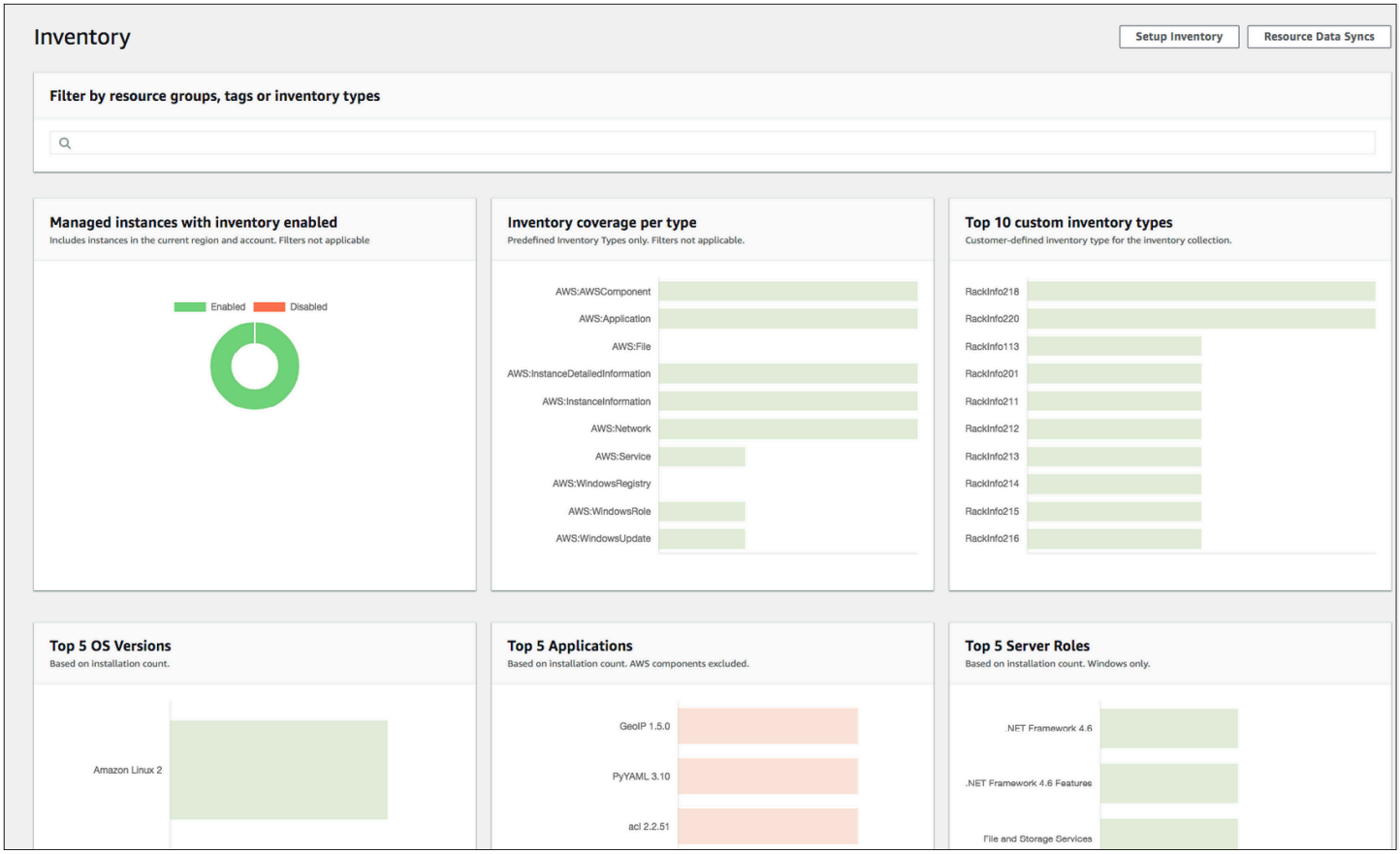
다음 표에서는 Systems Manager Inventory로 수집할 수 있는 데이터 형식을 설명합니다. 또한 이 표에서는 노드를 대상으로 하는 다양한 오퍼링과 지정할 수 있는 수집 간격에 대해 설명합니다.

구성	Details
메타데이터 형식	다음 유형의 데이터를 수집할 인벤토리를 구성할 수 있습니다. • 애플리케이션: 애플리케이션 이름, 게시자, 버전 등 • AWS 구성 요소: EC2 드라이버, 에이전트, 버전 등 • 파일: 이름, 크기, 버전, 설치한 날짜, 수정 및 마지막 액세스 시각 등 • 네트워크 구성: IP 주소, MAC 주소, DNS, 게이트웨이, 서브넷 마스크 등 • Windows 업데이트: Hotfix ID, 설치한 사람, 설치한 날짜 등 • 인스턴스 세부 정보: 시스템 이름, 운영 체제(OS) 이름, OS 버전, DNS, 도메인, 작업 그룹, OS 아키텍처 등

구성	Details
	• 서비스: 이름, 표시 이름, 상태, 종속 서비스, 서비스 유형, 시작 유형 등 • 태그: 노드에 할당된 태그 • Windows 레지스트리: 레지스트리 키 경로, 값 이름, 값 유형 및 값 • Windows 역할: 이름, 표시 이름, 경로, 기능 유형, 설치된 상태 등 • 사용자 지정 인벤토리: 사용자 정의 인벤토리 작업에 설명된 대로 관리형 노드에 할당된 메타데이터 **Note** 인벤토리에서 수집한 모든 메타데이터의 목록을 보려면 인벤토리에서 수집한 메타데이터 섹션을 참조하세요.
대상 노드	AWS 계정에서 관리되는 모든 노드의 인벤토리를 선택하고, 태그를 사용하여 개별적으로 노드 또는 대상 노드 그룹을 선택할 수 있습니다. 모든 관리형 노드에서 인벤토리 데이터를 수집하는 방법에 대한 자세한 내용은 AWS 계정의 모든 관리형 노드에 대한 인벤토리 작성 섹션을 참조하세요.
정보 수집 시기	수집 간격을 분, 시간 및 일 단위로 지정할 수 있습니다. 가장 짧은 수집 간격은 30분입니다.

> **Note**
>
> 수집하는 데이터의 용량에 따라 시스템이 데이터를 지정된 출력 경로에 보고하는 데 몇 분이 걸릴 수 있습니다. 정보가 수집된 후 보안 HTTPS 채널을 통해 AWS 계정에서만 액세스할 수 있는 일반 텍스트 AWS 스토어로 데이터가 전송됩니다.

Systems Manager 콘솔의 [Inventory] 페이지에서 데이터를 볼 수 있으며, 손쉽게 데이터를 쿼리할 수 있도록 여러 개의 미리 정의된 카드가 포함됩니다.



> **Note**
>
> Inventory 카드가 [종료됨(Terminated)] 및 [중지됨(Stopped)] 상태의 Amazon EC2 관리형 인스턴스를 자동으로 필터링하여 제외합니다. 온프레미스와 AWS IoT Greengrass 코어 디바이스 관리형 노드의 경우 인벤토리 카드가 종료됨 상태의 노드를 자동으로 필터링하여 제외합니다.

리소스 데이터 동기화를 생성하여 단일 Amazon S3 버킷에 모든 데이터를 저장하고 동기화하는 경우 [인벤토리 세부 정보 뷰(Inventory Detailed View)] 페이지에서 데이터를 세부적으로 확인할 수 있습니다. 자세한 내용은 여러 리전 및 계정에서 인벤토리 데이터 쿼리 단원을 참조하십시오.

EventBridge 지원

이 Systems Manager 기능은 Amazon EventBridge 규칙에서 이벤트 유형으로 지원됩니다. 자세한 내용은 Amazon EventBridge로 Systems Manager 이벤트 모니터링 및 참조: Systems Manager용 Amazon EventBridge 이벤트 패턴 및 유형 섹션을 참조하세요.

내용

Systems Manager Inventory에 대해 자세히 알아보기

AWS Systems Manager 인벤토리를 구성할 때 수집할 메타데이터의 유형, 메타데이터를 수집할 관리형 노드, 메타데이터 모음에 대한 일정을 지정해야 합니다. 이러한 구성은 AWS 계정에 AWS Systems Manager State Manager 연결로 저장됩니다. 연결은 단순히 구성일 뿐입니다.

> **Note**
>
> 인벤토리는 메타데이터만 수집합니다. 개인 정보나 독점 정보는 수집하지 않습니다.

주제

인벤토리에서 수집한 메타데이터

다음 샘플은 각 AWS Systems Manager Inventory 플러그인에서 수집한 메타데이터의 전체 목록을 보여줍니다.

```
{
    "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:Process",
    "version": "1.0",
    "attributes":[
      { "name": "StartTime",          "dataType": "STRING"},
      { "name": "CommandLine",        "dataType": "STRING"},
      { "name": "User",               "dataType": "STRING"},
      { "name": "FileName",           "dataType": "STRING"},
      { "name": "FileVersion",        "dataType": "STRING"},
      { "name": "FileDescription",    "dataType": "STRING"},
      { "name": "FileSize",           "dataType": "STRING"},
      { "name": "CompanyName",        "dataType": "STRING"},
      { "name": "ProductName",        "dataType": "STRING"},
      { "name": "ProductVersion",     "dataType": "STRING"},
      { "name": "InstalledDate",      "dataType": "STRING"},
      { "name": "InstalledDir",       "dataType": "STRING"},
      { "name": "UsageId",            "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

- `"typeName": "AWS:InstanceInformation"`의 경우에, `InstanceStatus`는 Active, ConnectionLost, Stopped, Terminated 중 하나일 수 있습니다.
- 버전 2.5의 릴리스에서 RPM Package Manager는 일련 속성을 Epoch로 교체했습니다. Epoch 속성은 일련처럼 단조 증가하는 정수입니다. `AWS:Application` 유형을 이용해 인벤토리를 작성할 때 큰 Epoch 값은 새 버전을 의미합니다. Epoch 값이 동일하거나 빈 경우에는 버전 또는 릴리스 속성의 값으로 새 버전을 판단합니다.

- 일부 메타데이터는 Linux 인스턴스에서 제공되지 않습니다. 특히 "typeName": "AWS:Network"의 경우 다음과 같은 메타데이터 유형은 아직 Linux 인스턴스에 대해 지원되지 않습니다. Windows의 경우에는 지원됩니다.
  - { "name": "SubnetMask", "dataType": "STRING"},
  - { "name": "DHCPServer", "dataType": "STRING"},
  - { "name": "DNSServer", "dataType": "STRING"},
  - { "name": "Gateway", "dataType": "STRING"},

파일 및 Windows 레지스트리 인벤토리 관련 작업

AWS Systems Manager Inventory를 통해 Windows, Linux 및 macOS 운영 체제에서 파일을 검색하고 인벤토리로 만들 수 있습니다. 또한 Windows 레지스트리를 검색하고 인벤토리로 만들 수 있습니다.

Files: 몇 가지 예를 들자면 파일 이름, 파일 생성 시각, 파일을 마지막으로 수정하고 액세스한 시간, 파일 크기 등 파일에 관한 메타데이터 정보를 수집할 수 있습니다. 파일 인벤토리 수집을 시작하려면 인벤토리를 수행할 파일 경로, 인벤토리로 만들고자 하는 파일의 유형을 정의하는 패턴 한 개 이상, 그리고 그 경로가 반복적으로 통과하는지 여부를 지정합니다. Systems Manager는 패턴과 일치하는 지정된 경로의 파일에 대한 모든 파일 메타데이터의 인벤토리를 작성합니다. 파일 인벤토리에서는 다음 파라미터 입력을 사용합니다.

```
{
"Path": string,
"Pattern": array[string],
"Recursive": true,
"DirScanLimit" : number // Optional
}
```

- Path: 파일을 인벤토리로 만들고자 하는 디렉터리 경로. Windows에서는 해당 환경 변수가 단일 디렉터리 경로와 매핑되는 경우에 한해 %PROGRAMFILES%와 같은 환경 변수를 사용할 수 있습니다. 예를 들어 여러 디렉터리 경로에 매핑되는 %PATH%를 사용하는 경우 Inventory에는 오류가 발생합니다.
- Pattern: 파일을 식별할 패턴의 어레이
- Recursive: Inventory가 디렉터리를 반복적으로 통과하는지 여부를 나타내는 부울 값
- DirScanLimit: 몇 개의 디렉터리를 검사할 수 있는지 지정하는 값(선택 사항). 이 파라미터를 사용하여 해당 노드의 성능 저하를 최소화합니다. 기본적으로 Inventory는 최소 5,000개의 디렉터리를 검사합니다.

> **Note**
>
> Inventory는 지정된 모든 경로에서 최소 500개 파일에 대해 메타데이터를 수집합니다.

다음은 파일에 대한 인벤토리 작업을 수행할 때 파라미터를 지정하는 방법을 보여주는 예제입니다.

- Linux 및 macOS에서는 `/home/ec2-user` 디렉터리(하위 디렉터리는 모두 제외)에 .sh 파일에 대한 메타데이터를 수집합니다.

```
[{"Path":"/home/ec2-user","Pattern":["*.sh", "*.sh"],"Recursive":false}]
```

- Windows에서는 모든 ".exe" 파일에 대한 메타데이터를 Program Files 폴더(하위 디렉터리 포함)에 반복적으로 수집합니다.

```
[{"Path":"C:\Program Files","Pattern":["*.exe"],"Recursive":true}]
```

- Windows에서는 특정 로그 패턴의 메타데이터를 수집합니다.

```
[{"Path":"C:\ProgramData\Amazon","Pattern":["*amazon*.log"],"Recursive":true}]
```

- 반복 수집을 수행할 때는 디렉터리 개수를 제한합니다.

```
[{"Path":"C:\Users","Pattern":["*.ps1"],"Recursive":true, "DirScanLimit": 1000}]
```

Windows 레지스트리: Windows 레지스트리 키 및 값을 수집할 수 있습니다. 키 경로를 선택하고 모든 키와 값을 반복적으로 수집할 수 있습니다. 특정 경로에 대해 특정 레지스트리 키와 그 값을 수집할 수도 있습니다. 인벤토리는 키 경로, 이름 및 값을 수집합니다.

```
{
"Path": string,
"Recursive": true,
"ValueNames": array[string] // optional
}
```

- Path: 레지스트리 키의 경로
- Recursive: 인벤토리가 레지스트리 경로를 반복적으로 통과하는지 여부를 나타내는 부울 값

- ValueNames: 레지스트리 키에 대한 인벤토리 작업을 위한 값 이름의 어레이. 이 파라미터를 사용하는 경우 Systems Manager는 지정된 경로에 대한 지정된 값 이름만 인벤토리로 만듭니다.

> **Note**
>
> 인벤토리는 지정된 모든 경로에 대해 최소 250개 레지스트리 키 값을 수집합니다.

다음은 Windows 레지스트리에 대한 인벤토리 작업을 수행할 때 파라미터를 지정하는 방법을 보여주는 예제입니다.

- 특정 경로에 대해 모든 키와 값을 반복적으로 수집합니다.

```
[{"Path":"HKEY_LOCAL_MACHINE\SOFTWARE\Amazon","Recursive": true}]
```

- 특정 경로에 대해 모든 키와 값을 수집합니다(반복 검색은 해제됨).

```
[{"Path":"HKEY_LOCAL_MACHINE\SOFTWARE\Intel\PSIS\PSIS_DECODER", "Recursive": false}]
```

- `ValueNames` 옵션을 사용하여 특정 키를 수집합니다.

```
{"Path":"HKEY_LOCAL_MACHINE\SOFTWARE\Amazon\MachineImage","ValueNames":["AMIName"]}
```

관련 AWS 서비스

AWS Systems Manager 인벤토리는 현재 인벤토리의 스냅샷을 제공하여, 소프트웨어 정책을 관리하고 전체 집합의 보안 태세를 개선하는 데 도움을 줍니다. 다음 AWS 서비스을(를) 사용하여 인벤토리 관리와 마이그레이션 기능을 확장할 수 있습니다.

- AWS Config는 구성 항목이 변경될 경우 알림을 발생시키는 규칙을 수립할 수 있는 기능과 함께 인벤토리 변경에 대한 기록 레코드를 제공합니다. 자세한 내용은 AWS Config Developer Guide의 Recording Amazon EC2 managed instance inventory를 참조하세요.
- AWS Application Discovery Service는 AWS로의 성공적인 마이그레이션을 지원하기 위해 온프레미스 VM에서 OS 유형, 애플리케이션 인벤토리, 프로세스, 연결, 서버 성능 지표에 대한 인벤토리를 수집하도록 만들어졌습니다. 자세한 내용은 Application Discovery Service User Guide를 참조하세요.

Systems Manager Inventory 설정

AWS Systems Manager Inventory를 사용하여 관리형 노드에서 실행 중인 애플리케이션, 서비스, AWS 구성 요소 등에 대한 메타데이터를 수집하기 전에 단일 Amazon Simple Storage Service(Amazon S3) 버킷의 인벤토리 데이터 스토리지를 중앙 집중화하도록 리소스 데이터 동기화를 구성하는 것이 좋습니다. 또한 인벤토리 이벤트의 Amazon EventBridge 모니터링을 구성하는 것이 좋습니다. 이러한 프로세스를 통해 인벤토리 데이터 및 수집을 보다 쉽게 보고 관리할 수 있습니다.

주제

Inventory의 리소스 데이터 동기화 구성

이 주제에서는 AWS Systems Manager 인벤토리에 대한 리소스 데이터 동기화를 설정하고 구성하는 방법에 대해 설명합니다. Systems Manager Explorer의 리소스 데이터 동기화에 대한 자세한 내용은 여러 계정 및 리전에서 데이터를 표시하도록 Systems Manager Explorer 설정 단원을 참조하세요.

리소스 데이터 동기화 정보

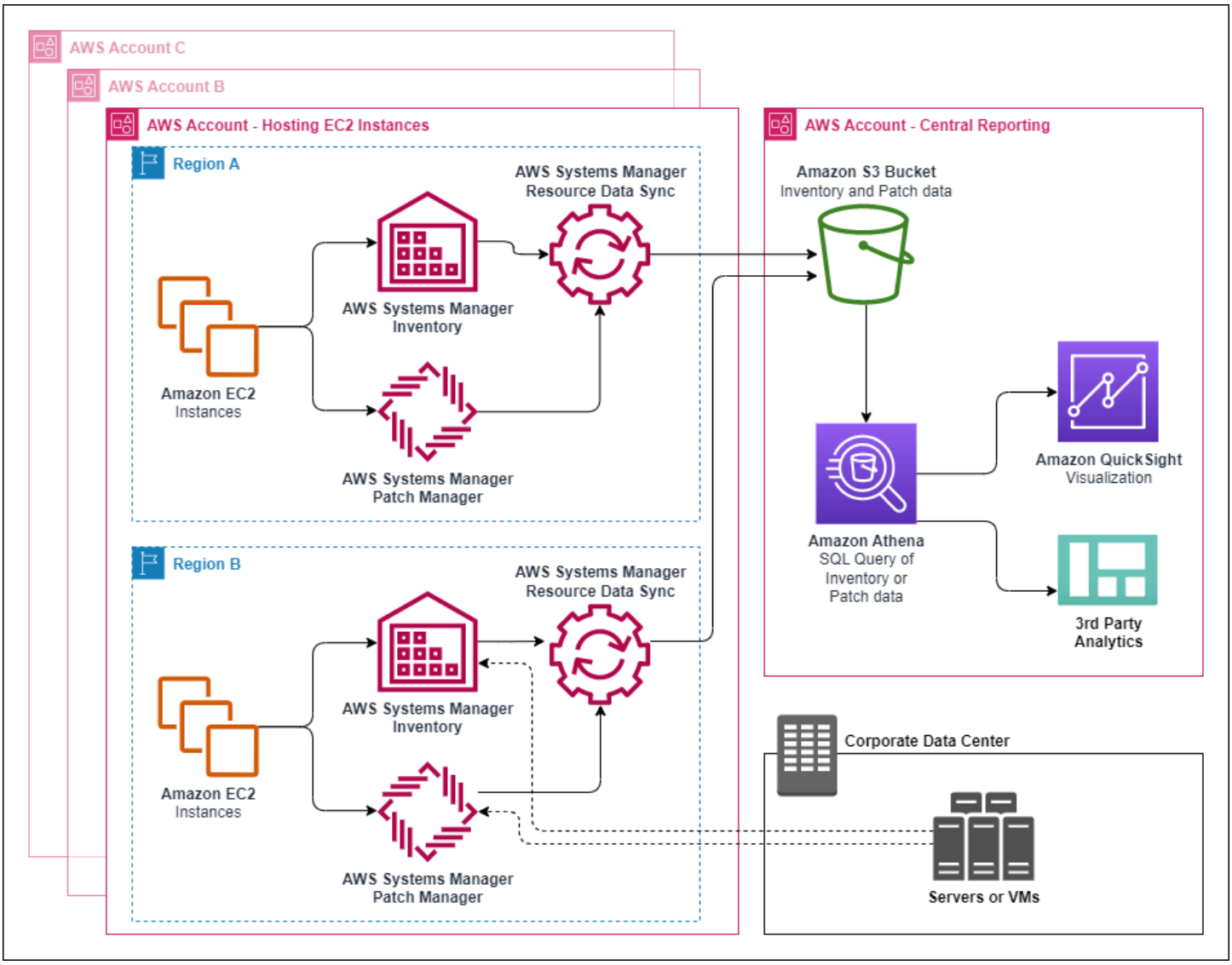
Systems Manager 리소스 데이터 동기화를 사용하여 모든 관리형 노드에서 수집된 인벤토리 데이터를 단일 Amazon Simple Storage Service(Amazon S3) 버킷으로 전송합니다. 이제 새로운 인벤토리 데이터가 수집될 때마다 리소스 데이터 동기화가 중앙 데이터를 자동으로 업데이트합니다. 모든 인벤토리 데이터가 대상 Amazon S3 버킷에 저장되면 Amazon Athena와 Amazon QuickSight 같은 서비스를 사용하여 수집한 데이터에 대한 쿼리를 실행하거나 분석할 수 있습니다.

예를 들어 150개 관리형 노드 플릿에서 실행 중인 운영 체제(OS) 및 애플리케이션에 대한 데이터를 수집할 수 있도록 인벤토리를 구성하였다고 가정하겠습니다. 이러한 노드 중 일부는 노드는 온프레미스 데이터 센터에 있고, 다른 노드는 여러 AWS 리전의 Amazon Elastic Compute Cloud(Amazon EC2)에서 실행되고 있습니다. 리소스 데이터 동기화를 구성하지 않았다면 각 노드마다 수집된 인벤토리 데이터를 수동으로 수집하거나, 혹은 스크립트를 생성하여 이 정보를 수집해야 합니다. 그런 다음 쿼리를 통해 데이터를 분석하려면 데이터를 애플리케이션으로 포팅해야 합니다.

리소스 데이터 동기화를 사용하면 작업 한 번으로 모든 관리형 노드에서 수집된 인벤토리 데이터를 모두 동기화할 수 있습니다. 동기화가 성공적으로 생성되면 Systems Manager가 모든 인벤토리 데이터의 기준을 만들어서 대상 Amazon S3 버킷에 저장합니다. 이후 새로운 인벤토리 데이터가 수집되면 Systems Manager가 Amazon S3 버킷의 데이터를 자동 업데이트합니다. 이후부터는 데이터를 빠르고 경제적으로 Amazon Athena 및 Amazon QuickSight에 포팅할 수 있습니다.

다이어그램 1에서는 리소스 데이터 동기화를 통해 하이브리드 및 멀티클라우드 환경의 Amazon EC2 및 기타 시스템 유형에서 대상 Amazon S3 버킷으로 인벤토리 데이터를 집계하는 방법을 보여줍니다. 또한 리소스 데이터 동기화가 여러 AWS 계정 및 AWS 리전에서 어떻게 작동하는지도 보여줍니다.

다이어그램 1: 여러 AWS 계정 및 AWS 리전에서 리소스 데이터 동기화



관리형 노드가 삭제되더라도 리소스 데이터 동기화가 삭제된 노드의 인벤토리 파일을 보존합니다. 하지만 실행 노드의 경우에는 새로운 파일이 생성되어 Amazon S3 버킷에 작성되면 리소스 데이터 동기화가 이전 인벤토리 파일을 자동으로 덮어씁니다. 시간 경과에 따른 인벤토리 변경 사항을 추적하고 싶다면 AWS Config 서비스를 사용하여 `SSM:ManagedInstanceInventory` 리소스 유형을 추적할 수 있습니다. 자세한 내용은 AWS Config 시작하기 섹션을 참조하세요.

이 섹션의 절차에 따라 Amazon S3 및 AWS Systems Manager 콘솔을 사용하여 Inventory에 대한 리소스 데이터 동기화를 생성할 수 있습니다. AWS CloudFormation을 사용하여 리소스 데이터 동기화를 생

성하거나 삭제할 수도 있습니다. AWS CloudFormation을 사용하려면 AWS CloudFormation 템플릿에 `AWS::SSM::ResourceDataSync` 리소스를 추가합니다. 자세한 내용은 다음 설명서 리소스 중 하나를 참조하십시오.

- AWS CloudFormation resource for resource data sync in AWS Systems Manager(블로그)
- AWS CloudFormation 사용 설명서의 AWS CloudFormation 템플릿 작업

> **Note**
>
> AWS Key Management Service(AWS KMS)를 사용하여 Amazon S3 버킷의 인벤토리 데이터를 암호화할 수 있습니다. AWS Command Line Interface(AWS CLI)를 사용하여 암호화된 동기화를 생성하는 방법과 Amazon Athena 및 Amazon QuickSight에서 중앙 데이터를 사용하는 방법에 대한 예는 시연: 리소스 데이터 동기화를 사용하여 인벤토리 데이터 집계 섹션을 참조하세요.

시작하기 전 준비 사항

리소스 데이터 동기화를 생성하기 전에 다음 절차를 사용하여 집계된 인벤토리 데이터를 저장할 중앙 Amazon S3 버킷을 생성합니다. 이 절차에서는 Systems Manager가 여러 계정에서 버킷에 인벤토리 데이터를 쓸 수 있는 버킷 정책을 할당하는 방법을 설명합니다. 리소스 데이터 동기화를 위해 인벤토리 데이터를 집계하는 데 사용할 Amazon S3 버킷이 이미 있는 경우 다음 절차에서 정책을 사용하도록 버킷을 구성해야 합니다.

> **Note**
>
> 지정된 Amazon S3 버킷이 객체 잠금을 사용하도록 구성된 경우 Systems Manager Inventory는 해당 버킷에 데이터를 추가할 수 없습니다. 리소스 데이터 동기화를 위해 생성하거나 선택한 Amazon S3 버킷이 Amazon S3 객체 잠금을 사용하도록 구성되지 않았는지 확인합니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 Amazon S3 객체 잠금 작동 방식을 참조하세요.

리소스 데이터 동기화를 위해 Amazon S3 버킷을 생성하고 구성하려면

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.

2. 수집한 인벤토리 데이터를 저장할 버킷을 생성합니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 버킷 생성을 참조하세요. 버킷 이름과 버킷을 생성한 AWS 리전을 따로 적어둡니다.
3. 권한 탭을 선택한 후 버킷 정책을 선택합니다.
4. 다음 버킷 정책을 복사하여 정책 편집기에 붙여 넣습니다. 이때 DOC-EXAMPLE-BUCKET 및 *account-id*를 사용자가 생성한 S3 버킷 이름 및 유효한 AWS 계정 ID로 바꿉니다.

   여러 AWS 계정에서 중앙 Amazon S3 버킷으로 인벤토리 데이터를 보내려면 다음 `Resource` 샘플에 표시된 것처럼 정책에서 각 계정을 지정합니다.

```
"Resource": [
    "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*/accountid=123456789012/*",
    "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*/accountid=444455556666/*",
    "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*/accountid=777788889999/*"
],
"Condition": {
    "StringEquals": {
        "s3:x-amz-acl": "bucket-owner-full-control",
        "aws:SourceAccount": [
                "123456789012",
                "444455556666",
                "777788889999"
        ]
     },
      "ArnLike": {
          "aws:SourceArn": [
            "arn:aws:ssm:*:123456789012:resource-data-sync/*",
            "arn:aws:ssm:*:444455556666:resource-data-sync/*",
            "arn:aws:ssm:*:777788889999:resource-data-sync/*"
          ]
     }
}
```

> **Note**
>
> AWS 계정 ID 보기에 대한 자세한 내용은 IAM User Guide의 Your Amazon Web Services Account ID and Its Alias를 참조하세요.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
        "Sid": "SSMBucketPermissionsCheck",
        "Effect": "Allow",
        "Principal": {
            "Service": "ssm.amazonaws.com"
        },
        "Action": "s3:GetBucketAcl",
        "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
        },
        {
        "Sid": " SSMBucketDelivery",
        "Effect": "Allow",
        "Principal": {
            "Service": "ssm.amazonaws.com"
        },
        "Action": "s3:PutObject",
        "Resource": [
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*/accountid=ID_number/*",
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*/accountid=ID_number/*",
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*/accountid=ID_number/*",
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*/accountid=ID_number/*"
        ],
        "Condition": {
            "StringEquals": {
                "s3:x-amz-acl": "bucket-owner-full-control",
                "aws:SourceAccount": "ID_number"
            },
            "ArnLike": {
                "aws:SourceArn": "arn:aws:ssm:*:ID_number:resource-data-sync/*"
            }
        }
      }
    ]
}
```

Inventory의 리소스 데이터 동기화 생성

Systems Manager Inventory의 리소스 데이터 동기화는 Systems Manager 콘솔에서 다음 절차에 따라 생성합니다. AWS CLI를 사용하여 리소스 데이터 동기화를 만드는 방법에 대한 자세한 내용은 연습: CLI를 사용하여 인벤토리를 위한 관리형 노드 구성 섹션을 참조하세요.

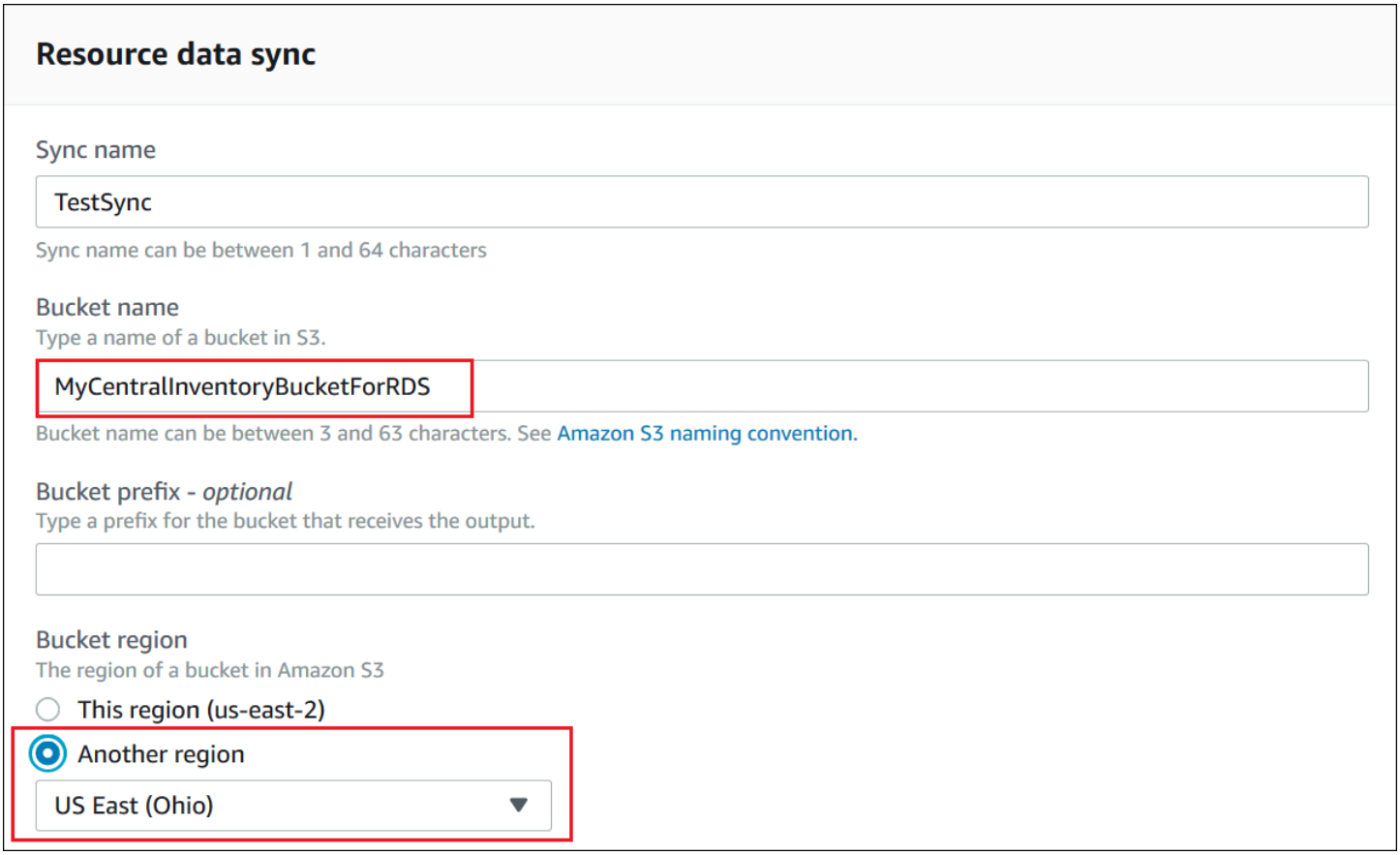
리소스 데이터 동기화를 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. [계정 관리(Account management)] 메뉴에서 [리소스 데이터 동기화(Resource data sync)]를 선택합니다.
4. Create resource data sync(리소스 데이터 동기화 생성)를 선택합니다.
5. [동기화 이름(Sync name)] 필드에 동기화 구성 이름을 입력합니다.
6. [버킷 이름(Bucket name)] 필드에 리소스 데이터 동기화를 위한 Amazon S3 버킷을 생성하고 구성하려면 절차를 사용하여 생성한 Amazon S3 버킷의 이름을 입력합니다.
7. (옵션) [버킷 접두사(Bucket prefix)] 필드에 Amazon S3 버킷 접두사(하위 디렉터리)의 이름을 입력합니다.
8. 생성한 Amazon S3 버킷이 현재 AWS 리전에 위치하면 [버킷 리전(Bucket region)] 필드에서 [이 리전(This region)]을 선택합니다. 버킷이 다른 AWS 리전에 위치하면 [다른 리전(Another region)]을 선택하고 리전 이름을 입력합니다.

   > **Note**
   >
   > 동기화와 대상 Amazon S3 버킷이 서로 다른 리전에 위치하는 경우에는 데이터 전송 요금이 부과될 수 있습니다. 자세한 내용은 Amazon S3 요금을 참조하세요.

9. (옵션) [KMS Key ARN] 필드에 Amazon S3의 인벤토리 데이터를 암호화할 KMS 키 ARN을 입력하거나 붙여 넣습니다.
10. 생성(Create)을 선택합니다.

여러 AWS 리전의 인벤토리 데이터를 동기화하려면 각 리전에서 리소스 데이터 동기화를 생성해야 합니다. 인벤토리 데이터를 수집해 중앙 Amazon S3 버킷으로 보내려는 각 AWS 리전에서 이 절차를 반복합니다. 각 리전에서 동기화를 생성할 때 [버킷 이름(Bucket name)] 필드에 중앙 Amazon S3 버킷을 지정합니다. 그런 다음 [버킷 리전(Bucket region)] 옵션을 사용하여 다음 스크린샷에 표시된 것처럼 중앙 Amazon S3 버킷을 생성한 리전을 선택합니다. 연결이 실행되어 인벤토리 데이터를 수집하고 나면 Systems Manager가 중앙 Amazon S3 버킷에 데이터를 저장합니다.



AWS Organizations에 정의된 계정에 대한 인벤토리 리소스 데이터 동기화 생성

AWS Organizations에 정의된 AWS 계정의 인벤토리 데이터를 중앙 Amazon S3 버킷으로 동기화할 수 있습니다. 다음 절차를 완료하면 인벤토리 데이터가 중앙 버킷의 개별 Amazon S3 키 접두사와 동기화됩니다. 각 키 접두사는 서로 다른 AWS 계정 ID를 나타냅니다.

시작하기 전 준비 사항

시작하기 전에 AWS Organizations에서 AWS 계정를 설정하고 구성했는지 확인합니다. 자세한 내용은 AWS Organizations User Guide를 참조하세요.

또한 AWS Organizations에 정의된 AWS 리전과 AWS 계정 각각에 대해 조직 기반 자원 데이터 동기화를 생성해야 합니다.

중앙 Amazon S3 버킷 생성

다음 절차를 사용하여 집계된 인벤토리 데이터를 저장할 중앙 Amazon S3 버킷을 생성합니다. 이 절차에서는 Systems Manager가 AWS Organizations 계정 ID에서 버킷에 인벤토리 데이터를 쓸 수 있는 버킷 정책을 할당하는 방법을 설명합니다. 리소스 데이터 동기화를 위해 인벤토리 데이터를 집계하는 데 사용할 Amazon S3 버킷이 이미 있는 경우 다음 절차에서 정책을 사용하도록 버킷을 구성해야 합니다.

AWS Organizations에 정의된 여러 계정의 리소스 데이터 동기화를 위한 Amazon S3 버킷을 생성하고 구성하려면

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 집계된 인벤토리 데이터를 저장할 버킷을 생성합니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 버킷 생성을 참조하세요. 버킷 이름과 버킷을 생성한 AWS 리전을 따로 적어둡니다.
3. 권한 탭을 선택한 후 버킷 정책을 선택합니다.
4. 다음 버킷 정책을 복사하여 정책 편집기에 붙여 넣습니다. 이때 DOC-EXAMPLE-BUCKET 및 *organization-id*를 사용자가 생성한 Amazon S3 버킷 이름 및 유효한 AWS Organizations 계정 ID로 바꿉니다.

   원할 경우 *bucket-prefix*를 Amazon S3 접두사(하위 디렉터리) 이름으로 바꿉니다. 접두사를 생성하지 않았을 경우에는 다음 정책의 ARN에서 *bucket-prefix/*를 제거하십시오.

```
{
  "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:::DOC-EXAMPLE-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:::DOC-EXAMPLE-BUCKET/bucket-prefix/*/accountid=*/*"
      ]
    }
  ]
}
```

AWS Organizations에 정의된 계정에 대한 인벤토리 리소스 데이터 동기화 생성

다음 절차에서는 AWS CLI를 사용하여 AWS Organizations에 정의된 계정에 대한 리소스 데이터 동기화를 만드는 방법을 설명합니다. 이 작업을 수행하려면 AWS CLI를 사용해야 합니다. AWS Organizations에 정의된 AWS 리전과 AWS 계정에 대해서도 각각 이 절차를 수행해야 합니다.

AWS Organizations(AWS CLI)에 정의된 계정에 대한 리소스 데이터 동기화를 생성하려면

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.

2. 다음 명령을 실행하여 다른 리소스 데이터 동기화가 없는지 확인합니다. 조직 기반 리소스 데이터 동기화는 하나만 가질 수 있습니다.

   ```
   aws ssm list-resource-data-sync
   ```

   명령이 다른 리소스 데이터 동기화를 반환하는 경우 이를 삭제하거나 새 동기화를 생성하지 않도록 선택해야 합니다.

3. 다음 명령을 실행하여 AWS Organizations에 정의된 계정에 대한 리소스 데이터 동기화를 생성합니다. DOC-EXAMPLE-BUCKET에 이 주제의 앞부분에서 생성한 Amazon S3 버킷의 이름을 지정합니다. 버킷의 접두사(하위 디렉터리)를 생성한 경우 *prefix-name*에 이 정보를 지정합니다.

```
aws ssm create-resource-data-sync --sync-name name --s3-
destination "BucketName=DOC-EXAMPLE-BUCKET,Prefix=prefix-
name,SyncFormat=JsonSerDe,Region=AWS ##, for example us-
east-2,DestinationDataSharing={DestinationDataSharingType=Organization}"
```

4. 중앙 Amazon S3 버킷에 데이터를 동기화하려는 모든 AWS 리전 및 AWS 계정에 대해 2단계와 3단계를 반복합니다.

리소스 데이터 동기화 관리

각 AWS 계정에서는 AWS 리전당 리소스 데이터 동기화가 5회씩 발생할 수 있습니다. AWS Systems Manager Fleet Manager콘솔을 사용하여 리소스 데이터 동기화를 관리할 수 있습니다.

리소스 데이터 동기화를 보는 방법

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 계정 관리 드롭다운에서 리소스 데이터 동기화를 선택합니다.
4. 테이블에서 리소스 데이터 동기화를 선택한 다음에 세부 정보 보기를 선택하여 리소스 데이터 동기화에 대한 정보를 봅니다.

리소스 데이터 동기화를 삭제하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 계정 관리 드롭다운에서 리소스 데이터 동기화를 선택합니다.
4. 테이블에서 리소스 데이터 동기화를 선택한 다음에 삭제를 선택합니다.

Inventory 이벤트의 EventBridge 모니터링 정보

AWS Systems Manager Inventory 리소스 상태 변경에 대한 응답으로 이벤트를 생성하도록 Amazon EventBridge에서 규칙을 구성할 수 있습니다. EventBridge는 다음과 같은 Inventory 상태 변경에 대한 이벤트를 지원합니다. 모든 이벤트는 최선의 작업을 기반으로 전송됩니다.

특정 인스턴스에 대해 사용자 정의 인벤토리 유형이 삭제됨: 이 이벤트를 모니터링하도록 규칙이 구성된 경우 특정 관리형의 사용자 정의 인벤토리 유형이 삭제될 때 EventBridge가 이벤트를 생성합니다.

EventBridge는 사용자 정의 인벤토리 유형별로 노드당 하나의 이벤트를 보냅니다. 다음은 샘플 이벤트 패턴입니다.

```
{
    "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"
    }
}
```

모든 인스턴스에 대한 사용자 정의 인벤토리 유형이 삭제됨: 이 이벤트를 모니터링하도록 규칙이 구성된 경우 모든 관리형 노드의 사용자 정의 인벤토리 유형이 삭제될 때 EventBridge가 이벤트를 생성합니다. 다음은 샘플 이벤트 패턴입니다.

```
{
    "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"
    }
}
```

이전 스키마 버전으로 PutInventory 호출: 이 이벤트를 모니터링하도록 규칙이 구성된 경우 현재 스키마보다 낮은 스키마 버전을 사용하는 PutInventory 호출이 수행될 때 EventBridge가 이벤트를 생성합니다. 이 이벤트는 모든 인벤토리 유형에 적용됩니다. 다음은 샘플 이벤트 패턴입니다.

```
{
    "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"
    }
}
```

이러한 이벤트를 모니터링하도록 EventBridge 구성하는 방법에 대한 자세한 내용은 Systems Manager 이벤트에 대해 EventBridge 구성 섹션을 참조하세요.

인벤토리 수집 구성

이 섹션에서는 Systems Manager 콘솔을 사용하여 관리형 노드 1개 이상에 대한 AWS Systems Manager Inventory 수집을 구성하는 방법에 대해서 설명합니다. AWS Command Line Interface(AWS

CLI)를 사용하여 인벤토리 수집을 구성하는 방법의 예는 Systems Manager Inventory 시연 섹션을 참조하세요.

인벤토리 수집을 구성할 때 AWS Systems Manager State Manager 연결을 생성하는 것으로 시작합니다. Systems Manager는 연결이 실행될 때 인벤토리 데이터를 수집합니다. 연결을 먼저 생성하지 않고 AWS Systems Manager Run Command 등을 사용하여 `aws:softwareInventory` 플러그 인을 호출하려고 하면 시스템이 다음 오류를 반환합니다. `The aws:softwareInventory plugin can only be invoked via ssm-associate.`

> **Note**
>
> 관리형 노드에 대해 여러 인벤토리 연결을 만드는 경우 다음 동작을 유의하세요.
>
> - 각 노드에 모든 노드(--targets "Key=InstanceIds,Values=*")를 대상으로 하는 인벤토리 연결이 할당될 수 있습니다.
> - 태그 키/값 페어 또는 AWS 리소스 그룹을 사용하는 특정 연결이 각 노드에 할당될 수도 있습니다.
> - 노드에 여러 인벤토리 연결이 할당된 경우 실행되지 않은 연결에 대해 상태가 건너뜀(Skipped)으로 표시됩니다. 가장 최근에 실행된 연결은 인벤토리 연결의 실제 상태를 표시합니다.
> - 노드에 여러 인벤토리 연결이 할당되고 각각 태그 키/값 페어를 사용하는 경우 태그 충돌로 인해 해당 인벤토리 연결이 노드에서 실행되지 않습니다. 태그 키/값 충돌이 없는 노드에서 연결이 계속 실행됩니다.

시작하기 전

인벤토리 수집을 구성하기 전에 다음 작업을 수행하십시오.

- 인벤토리로 만들고 싶은 노드에서 AWS Systems Manager SSM Agent를 업데이트합니다. SSM Agent 최신 버전을 실행하여 지원되는 모든 인벤토리 유형에 대한 메타데이터를 수집할 수 있는지 확인합니다. SSM Agent를 사용해 State Manager 를 업데이트하는 방법에 대한 자세한 내용은 시연: SSM Agent(CLI) 자동 업데이트 섹션을 참조하세요.
- 하이브리드 및 멀티클라우드 환경의 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 및 비 EC2 시스템에 대한 설정 요구 사항을 완료했는지 확인합니다. 자세한 설명은 AWS Systems Manager 설정을 참조하세요.

- Microsoft Windows 노드의 경우 노드가 Windows PowerShell 3.0 이상으로 구성되었는지 확인합니다. SSM Agent는 PowerShell에서 `ConvertTo-Json` cmdlet을 사용하여 Windows 업데이트 인벤토리 데이터를 필요한 형식으로 변환합니다.
- (옵션) 리소스 데이터 동기화를 생성하여 Amazon S3 버킷에 인벤토리 데이터를 중앙 집중식으로 저장합니다. 그러면 새 인벤토리 데이터가 수집될 때 리소스 데이터 동기화에 의해 중앙 데이터가 자동으로 업데이트됩니다. 자세한 내용은 Inventory의 리소스 데이터 동기화 구성 단원을 참조하십시오.
- (선택 사항) JSON 파일을 생성하여 사용자 정의 인벤토리를 수집합니다. 자세한 내용은 사용자 정의 인벤토리 작업 단원을 참조하십시오.

AWS 계정의 모든 관리형 노드에 대한 인벤토리 작성

전역 인벤토리 연결을 생성하면 AWS 계정의 모든 관리형 노드에 대한 인벤토리를 쉽게 작성할 수 있습니다. 전역 인벤토리 연결은 다음 작업을 수행합니다.

- AWS 계정에 있는 모든 관리형 노드에 전역 인벤토리 구성(연결)을 자동으로 적용합니다. 전역 인벤토리 연결이 적용되어 실행될 경우, 이미 인벤토리 연결이 있는 관리형 노드는 건너뜁니다. 노드를 건너뛰면 `Overridden By Explicit Inventory Association`이라는 자세한 상태 메시지가 표시됩니다. 이러한 노드는 전역 연결에서 건너뛰지만 각자 할당된 인벤토리 연결을 실행할 때는 인벤토리를 보고합니다.
- AWS 계정에 생성된 새로운 노드를 전역 인벤토리 연결에 자동으로 추가합니다.

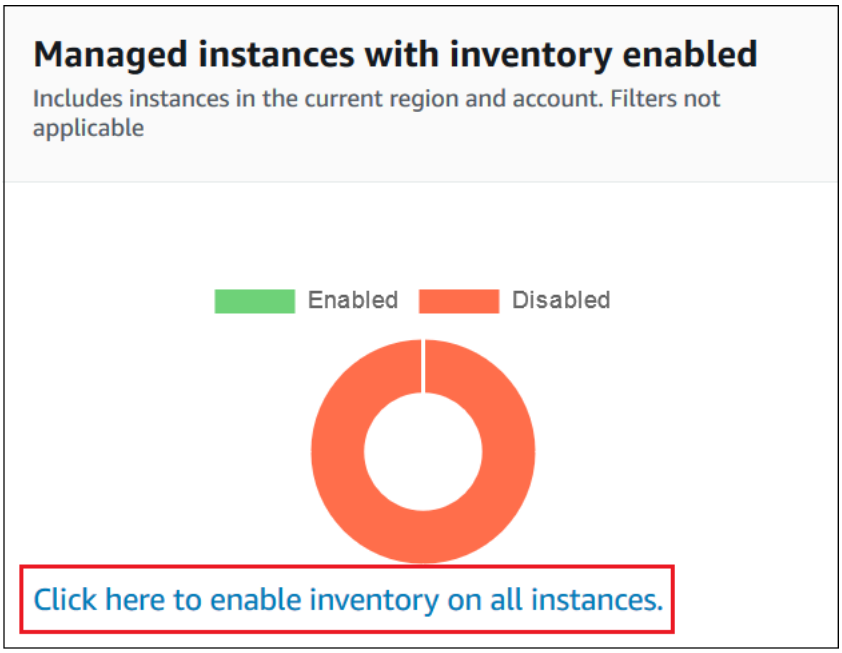
> **Note**
>
> - 전역 인벤토리 연결을 구성한 관리형 노드에 특정 연결을 할당하면 Systems Manager Inventory에서 전역 연결의 우선순위를 무시하고 특정 연결을 적용합니다.
> - 전역 인벤토리 연결은 SSM Agent 버전 2.0.790.0 이상에서 사용할 수 있습니다. 노드에서 SSM Agent를 업데이트하는 자세한 방법은 Run Command를 사용하여 SSM Agent 업데이트 섹션을 참조하세요.

원클릭 절차로 인벤토리 수집 구성(콘솔)

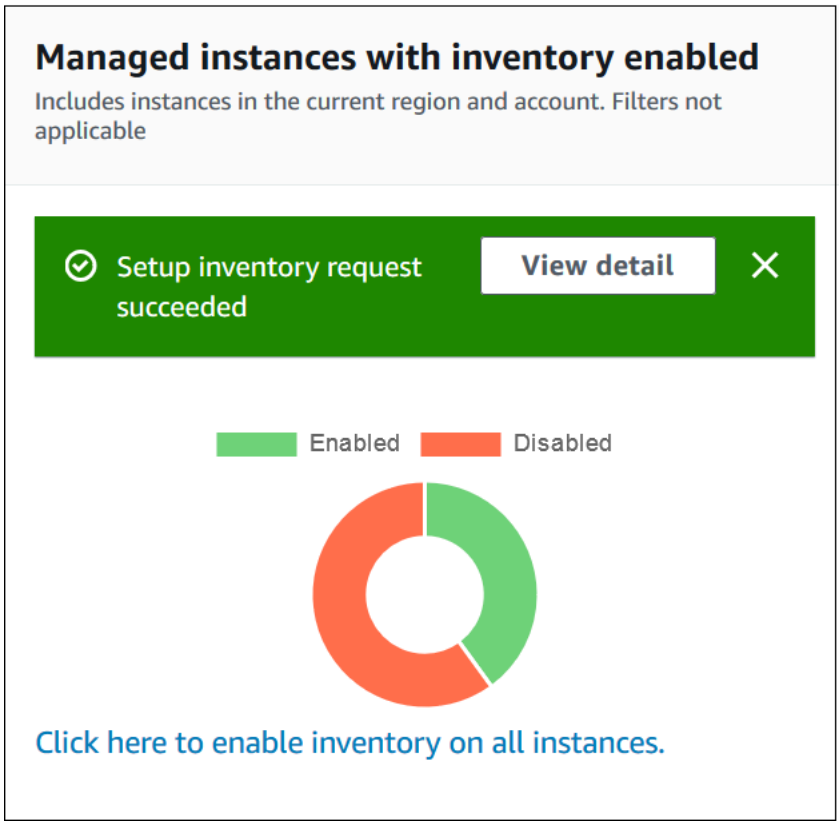
다음 절차에 따라 AWS 계정 및 단일 AWS 리전의 모든 관리형 노드에 대해 Systems Manager Inventory를 구성합니다.

Systems Manager Inventory의 현재 리전에서 모든 관리형 노드 구성

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 [Inventory]를 선택합니다.
3. Managed instances with inventory enabled(인벤토리가 활성화된 관리형 인스턴스) 카드에서 Click here to enable inventory on all instances(모든 인스턴스에서 인벤토리를 활성화하려면 여기를 클릭)를 선택합니다.



성공하면 콘솔에 다음 메시지가 표시됩니다.



계정에 포함된 관리형 노드 수에 따라 전역 인벤토리 연결이 적용되는 데 몇 분이 걸릴 수 있습니다. 몇 분 동안 기다린 다음 페이지를 새로 고칩니다. 그래픽이 바뀌어 모든 관리형 노드에서 인벤토리가 구성된 것으로 나타내는지 확인합니다.

콘솔을 사용하여 수집 구성

이 섹션에는 Systems Manager 콘솔을 사용하여 관리형 노드에서 메타데이터를 수집하도록 Systems Manager Inventory를 구성하는 방법이 나와 있습니다. 특정 AWS 계정의 모든 노드(및 그 계정에서 만들 수 있는 향후의 모든 노드)에서 빠르게 메타데이터를 수집하거나 태그 또는 노드 ID를 사용하여 선택적으로 인벤토리 데이터를 수집할 수 있습니다.

> **Note**
>
> 이 절차를 완료하기 전에 전역 인벤토리 연결이 이미 존재하는지 확인합니다. 전역 인벤토리 연결이 이미 존재하는 경우 새 인스턴스를 시작할 때마다 연결이 해당 인스턴스에 적용되고 새 인스턴스가 인벤토리에 추가됩니다.

인벤토리 수집을 구성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 [Inventory]를 선택합니다.
3. 인벤토리 설정을 선택합니다.
4. 대상 섹션에서, 다음 중 하나를 선택하여 이 작업을 실행할 노드를 식별합니다.

   - 이 계정의 모든 관리형 인스턴스 선택 - 이 옵션은 인벤토리 연결이 존재하지 않는 모든 관리형 노드를 선택합니다. 이 옵션을 선택하면 인벤토리 연결이 이미 있는 노드들이 인벤토리 수집에서 무시되고, 인벤토리 결과에 건너뜀 상태로 표시됩니다. 자세한 내용은 AWS 계정의 모든 관리형 노드에 대한 인벤토리 작성 단원을 참조하십시오.
   - 태그 지정(Specifying a tag) - 태그 하나를 지정하여 계정에서 인벤토리를 수집할 노드를 식별하려면 이 옵션을 사용합니다. 태그를 사용하는 경우 그 이후로 이와 동일한 태그를 사용하여 생성되는 노드도 인벤토리를 보고합니다. 모든 노드와 연결된 인벤토리 연결이 존재하는 경우, 태그를 사용하여 특정 노드를 여러 인벤토리에 대한 대상으로 선택하면 모든 관리형 인스턴스 대상 그룹의 노드 멤버십을 무시합니다. 지정한 태그를 가진 관리형 노드는 이후에 수행되는 모든 관리형 인스턴스의 인벤토리 수집에서 무시됩니다.
   - 수동으로 인스턴스 선택(Manually selecting instances) - 계정에서 특정 관리형 노드를 선택하려면 이 옵션을 사용합니다. 이 옵션을 사용하여 특정 노드를 명시적으로 선택하면 모든 관리형 인스턴스 대상의 인벤토리 연결을 무시합니다. 노드는 이후에 수행되는 모든 관리형 인스턴스의 인벤토리 수집에서 무시됩니다.

   > **Note**
   >
   > 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

5. 일정 섹션에서 시스템이 노드로부터 인벤토리 메타데이터를 수집하는 간격을 선택합니다.
6. [파라미터(Parameters)] 섹션에서 목록을 사용하여 다양한 유형의 인벤토리 수집을 설정하거나 해제합니다. 파일 또는 Windows 레지스트리에 대한 인벤토리 검색을 생성하려면 다음 샘플을 참조하십시오.

   파일

   - Linux 및 macOS에서는 `/home/ec2-user` 디렉터리(하위 디렉터리는 모두 제외)에 .sh 파일에 대한 메타데이터를 수집합니다.

```
[{"Path":"/home/ec2-user","Pattern":["*.sh", "*.sh"],"Recursive":false}]
```

- Windows에서는 모든 ".exe" 파일에 대한 메타데이터를 Program Files 폴더(하위 디렉터리 포함)에 반복적으로 수집합니다.

```
[{"Path":"C:\Program Files","Pattern":["*.exe"],"Recursive":true}]
```

- Windows에서는 특정 로그 패턴의 메타데이터를 수집합니다.

```
[{"Path":"C:\ProgramData\Amazon","Pattern":["*amazon*.log"],"Recursive":true}]
```

- 반복 수집을 수행할 때는 디렉터리 개수를 제한합니다.

```
[{"Path":"C:\Users","Pattern":["*.ps1"],"Recursive":true, "DirScanLimit": 1000}]
```

Windows 레지스트리

- 특정 경로에 대해 모든 키와 값을 반복적으로 수집합니다.

```
[{"Path":"HKEY_LOCAL_MACHINE\SOFTWARE\Amazon","Recursive": true}]
```

- 특정 경로에 대해 모든 키와 값을 수집합니다(반복 검색은 해제됨).

```
[{"Path":"HKEY_LOCAL_MACHINE\SOFTWARE\Intel\PSIS\PSIS_DECODER", "Recursive":
 false}]
```

- `ValueNames` 옵션을 사용하여 특정 키를 수집합니다.

```
{"Path":"HKEY_LOCAL_MACHINE\SOFTWARE\Amazon\MachineImage","ValueNames":
["AMIName"]}
```

파일 및 Windows 레지스트리 인벤토리를 수집하는 방법에 대한 자세한 내용은 파일 및 Windows 레지스트리 인벤토리 관련 작업 섹션을 참조하세요.

7. Amazon S3 버킷에 연결 실행 상태를 저장하고 싶으면 [고급(Advanced)] 섹션에서 [인벤토리 실행 로그를 Amazon S3 버킷에 동기화(Sync inventory execution logs to an Amazon S3 bucket)]를 선택합니다.

8. 인벤토리 설정을 선택합니다. Systems Manager가 State Manager 연결을 생성하고 노드에 대해 Inventory를 즉시 실행합니다.
9. 탐색 창에서 State Manager를 선택합니다. **AWS-GatherSoftwareInventory** 문서를 사용하는 새로운 연결이 생성되었는지 확인합니다. 연결 일정표는 rate 표현식을 사용합니다. 또한 상태 필드에 성공이 표시되는지 확인합니다. [Amazon S3 버킷으로 인벤토리 실행 로그 동기화(Sync inventory execution logs to an Amazon S3 bucket)] 옵션을 선택하는 경우 몇 분 후 Amazon S3에서 로그 데이터를 볼 수 있습니다. 특정 노드에 대한 인벤토리 데이터를 보려는 경우 탐색 창에서 관리형 인스턴스를 선택합니다.
10. 노드를 선택한 후 세부 정보 보기(View detail)를 선택합니다.
11. 노드 세부 정보 페이지에서 인벤토리(Inventory)를 선택합니다. 인벤토리 유형 목록을 사용하여 인벤토리를 필터링합니다.

Systems Manager 인벤토리 데이터 작업

이 섹션에는 AWS Systems Manager Inventory 데이터를 쿼리하고 집계하는 방법을 설명하는 주제가 포함되어 있습니다.

주제

여러 리전 및 계정에서 인벤토리 데이터 쿼리

AWS Systems Manager Inventory가 Amazon Athena와 통합되어 여러 AWS 리전 및 AWS 계정에서 인벤터리 데이터를 쿼리하는 데 도움이 됩니다. Athena 통합은 리소스 데이터 동기화를 사용하므로 AWS Systems Manager 콘솔의 세부 정보 보기 페이지에서 모든 관리형 노드의 인벤토리 데이터를 볼 수 있습니다.

> **Important**
>
> 이 기능은 AWS Glue를 사용하여 Amazon Simple Storage Service(Amazon S3) 버킷의 데이터를 크롤링하고 Amazon Athena를 사용하여 데이터를 쿼리합니다. 탐색하는 데이터의 양에 따라 이러한 서비스 사용에 대한 비용이 청구될 수 있습니다. AWS Glue는 시간당 요금제이며, 크롤러(데이터 검색) 및 ETL 작업(데이터 처리 및 로드)의 경우 초 단위로 비용이 청구됩니다. Athena에서는 각 쿼리가 검사한 데이터의 양을 기준으로 요금이 청구됩니다. Systems

Manager Inventory와 Amazon Athena의 통합을 사용하기 전에 이러한 서비스에 대한 요금 자침을 확인하는 것이 좋습니다. 자세한 내용은 Amazon Athena 요금 및 AWS Glue 요금을 참조하세요.

Amazon Athena를 사용할 수 있는 모든 AWS 리전의 세부 정보 보기(Detailed View) 페이지에서 인벤토리 데이터를 볼 수 있습니다. 지원되는 리전 목록은 Amazon Web Services 일반 참조의 Amazon Athena 서비스 엔드포인트를 참조하세요.

시작하기 전 준비 사항

Athena 통합은 리소스 데이터 동기화를 사용합니다. 이 기능을 사용하려면 리소스 데이터 동기화를 설정 및 구성해야 합니다. 자세한 내용은 Inventory의 리소스 데이터 동기화 구성 단원을 참조하십시오.

또한 세부 정보 보기(Detailed View) 페이지에 리소스 데이터 동기화에서 사용하는 중앙 Amazon S3 버킷의 소유자에 대한 인벤토리 데이터가 표시됩니다. 중앙 Amazon S3 버킷의 소유자가 아닌 경우에는 세부 정보 보기(Detailed View) 페이지에서 인벤토리 데이터가 표시되지 않습니다.

액세스 구성

Systems Manager 콘솔의 세부 정보 보기 페이지에서 여러 계정 및 리전의 데이터를 쿼리하고 보려면 먼저 데이터를 볼 수 있는 권한이 있는 IAM 엔터티를 구성해야 합니다.

인벤토리 데이터가 AWS Key Management Service(AWS KMS) 암호화를 사용하는 Amazon S3 버킷에 저장된 경우 AWS KMS 암호화를 위해 IAM 엔터티와 `Amazon-GlueServiceRoleForSSM` 서비스 역할 또한 구성해야 합니다.

주제

- 세부 정보 보기 페이지에 액세스하도록 IAM 엔터티 구성
- (선택 사항) AWS KMS 암호화 데이터를 볼 수 있는 권한 구성

세부 정보 보기 페이지에 액세스하도록 IAM 엔터티 구성

다음은 상세 보기 페이지에서 인벤토리 데이터를 보는 데 필요한 최소 권한에 대한 설명입니다.

`AWSQuicksightAthenaAccess` 관리형 정책

다음 `PassRole` 및 추가 필수 권한 블록

```
{
    "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": "*",
            "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/*"
        }
    ]
}
```

(선택 사항) 인벤토리 데이터를 저장하는 데 사용되는 Amazon S3 버킷이 AWS KMS를 사용하여 암호화된 경우 다음 블록도 정책에 추가해야 합니다.

```
{
    "Effect": "Allow",
    "Action": [
        "kms:Decrypt"
    ],
    "Resource": [
        "arn:aws:kms:Region:account_ID:key/key_ARN"
    ]
}
```

액세스 권한을 제공하려면 사용자, 그룹 또는 역할에 권한을 추가하세요:

- AWS IAM Identity Center의 사용자 및 그룹:

  권한 세트를 생성합니다. AWS IAM Identity Center 사용 설명서의 권한 세트 생성의 지침을 따르세요.
- ID 제공자를 통해 IAM에서 관리되는 사용자:

  ID 페더레이션을 위한 역할을 생성합니다. IAM 사용 설명서의 서드 파티 자격 증명 공급자의 역할 만들기(연합)의 지침을 따르세요.
- IAM 사용자:
  - 사용자가 맡을 수 있는 역할을 생성합니다. IAM 사용 설명서에서 IAM 사용자의 역할 생성의 지침을 따르세요.
  - (권장되지 않음)정책을 사용자에게 직접 연결하거나 사용자를 사용자 그룹에 추가합니다. IAM 사용 설명서에서 사용자(콘솔)에 권한 추가의 지침을 따르세요.

(선택 사항) AWS KMS 암호화 데이터를 볼 수 있는 권한 구성

인벤토리 데이터를 저장하는 데 사용되는 Amazon S3 버킷이 AWS Key Management Service(AWS KMS)를 사용하여 암호화된 경우, AWS KMS 키에 대한 `kms:Decrypt` 권한으로 IAM 엔터티 및 Amazon-GlueServiceRoleForSSM 역할을 구성해야 합니다.

시작하기 전 준비 사항

AWS KMS 키에 대한 `kms:Decrypt` 권한을 제공하려면 IAM 엔터티에 다음 정책 블록을 추가합니다.

```
{
```

```
    "Effect": "Allow",
    "Action": [
        "kms:Decrypt"
    ],
    "Resource": [
        "arn:aws:kms:Region:account_ID:key/key_ARN"
    ]
}
```

아직 수행하지 않은 경우 해당 절차를 완료하고 AWS KMS 키에 대해 `kms:Decrypt` 권한을 추가합니다.

다음 절차에 따라 AWS KMS 키에 대한 `kms:Decrypt` 권한으로 Amazon-GlueServiceRoleForSSM 역할을 구성합니다.

`kms:Decrypt` 권한을 가진 Amazon-GlueServiceRoleForSSM 역할을 구성하려면

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할(Roles)을 선택한 다음 검색 필드를 사용하여 Amazon-GlueServiceRoleForSSM 역할을 찾습니다. 요약 페이지가 열립니다.
3. 검색 필드를 사용하여 Amazon-GlueServiceRoleForSSM 역할을 찾습니다. 역할 이름을 선택합니다. 요약 페이지가 열립니다.
4. 역할 이름을 선택합니다. 요약 페이지가 열립니다.
5. 인라인 정책 추가(Add inline policy)를 선택합니다. 정책 생성 페이지가 열립니다.
6. JSON 탭을 선택합니다.
7. 편집기에서 기존 JSON 텍스트를 삭제하고 다음 정책을 복사한 다음 JSON 편집기에 붙여 넣습니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:Region:account_ID:key/key_ARN"
            ]
        }
```

```
    ]
}
```

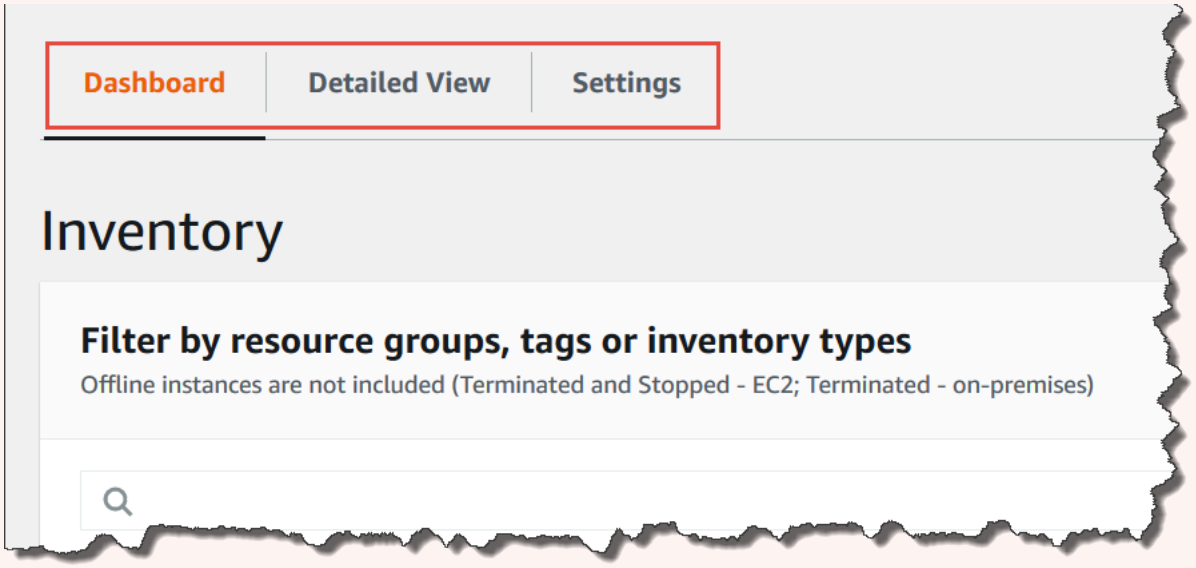
8. [정책 검토(Review policy)]를 선택합니다.
9. 정책 검토 페이지에서 이름 필드에 이름을 입력합니다.
10. 정책 생성을 선택합니다.

인벤토리 세부 정보 보기(Inventory Detail View) 페이지에서 데이터 쿼리

다음 절차에 따라 Systems Manager Inventory [세부 정보 뷰(Detailed View)] 페이지에서 여러 AWS 리전과 AWS 계정의 인벤토리 데이터를 봅니다.

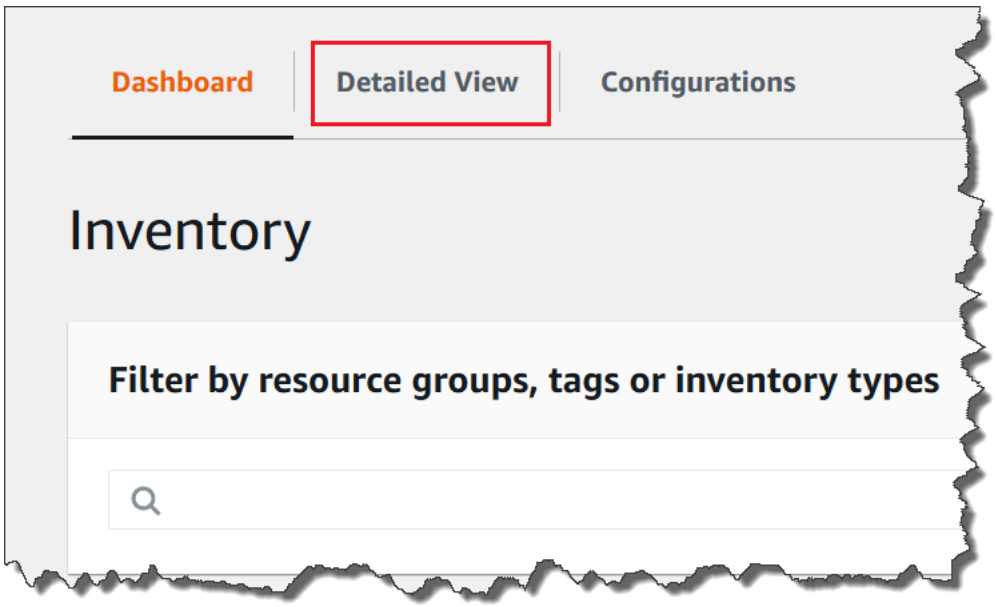
Important

Inventory [세부 정보 뷰(Detailed View)] 페이지는 Amazon Athena를 제공하는 AWS 리전에서만 사용할 수 있습니다. 다음 탭이 Systems Manager Inventory 페이지에 표시되지 않는 경우 Athena는 리전에서 사용할 수 없으며 세부 정보 보기(Detailed View)를 사용하여 데이터를 쿼리할 수 없습니다.

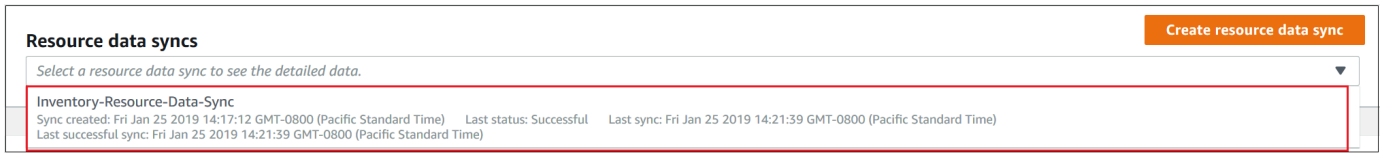


AWS Systems Manager 콘솔에서 여러 리전과 계정의 인벤토리 데이터를 보려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 [Inventory]를 선택합니다.
3. 세부 정보 보기(Detailed View) 탭을 선택합니다.

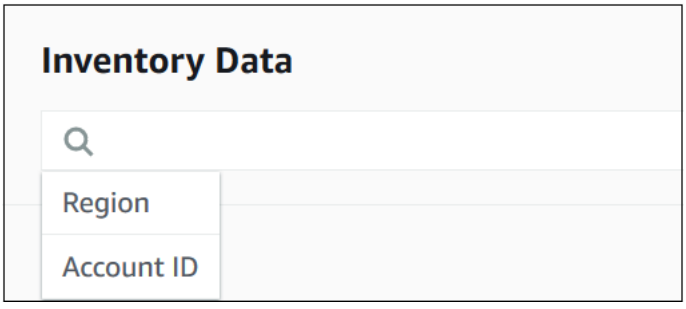


4. 데이터를 쿼리할 리소스 데이터 동기화를 선택합니다.



5. 인벤토리 유형 목록에서 쿼리하려는 인벤토리 데이터의 유형을 선택한 후 Enter를 누릅니다.

6. 데이터를 필터링하려면 필터 막대를 선택한 후 필터 옵션을 선택합니다.



CSV로 내보내기 버튼을 사용해 현재 쿼리 세트를 Microsoft Excel 등과 같은 스프레드시트 애플리케이션에서 볼 수 있습니다. [쿼리 기록(Query History)] 및 [고급 쿼리 실행(Run Advanced Queries)] 버튼을 사용해 Amazon Athena에서 기록 세부 정보를 보고 데이터와 상호 작용할 수도 있습니다.

AWS Glue 크롤러 예약 편집

AWS Glue는 기본적으로 중앙 Amazon S3 버킷에서 매일 두 번씩 인벤토리 데이터를 크롤링합니다. 노드에서 수집하는 데이터 유형을 자주 변경하는 경우 다음 절차의 설명에 따라 데이터를 더욱 자주 탐색할 수 있습니다.

> **Important**
>
> AWS Glue는 AWS 계정에 시간당 요금을 부과하며, 크롤러(데이터 검색) 및 ETL 작업(데이터 처리 및 로드)의 경우 초 단위로 비용이 청구됩니다. 크롤러 일정을 변경하기 전에 AWS Glue 요금 페이지를 확인하십시오.

인벤토리 데이터 크롤러 일정을 변경하려면

1. https://console.aws.amazon.com/glue/에서 AWS Glue 콘솔을 엽니다.
2. 탐색 창에서 크롤러를 선택합니다.
3. 크롤러 목록에서 Systems Manager Inventory 데이터 크롤러 옆의 옵션을 선택합니다. 크롤러 이름의 형식은 다음과 같습니다.

   `AWSSystemsManager-DOC-EXAMPLE-BUCKET-`*`Region`*`-`*`account_ID`*
4. 작업을 선택한 후 크롤러 편집을 선택합니다.
5. 탐색 창에서 일정을 선택합니다.
6. Cron 표현식 필드에서 cron 형식을 사용하여 새 일정을 지정합니다. cron 형식에 대한 자세한 내용은 AWS Glue 개발자 안내서의 크롤러와 작업을 위한 시간 기반 일정을 참조하세요.

> **Important**
>
> AWS Glue에서 더 이상 요금이 발생하지 않도록 크롤러를 일시 중지할 수 있습니다. 크롤러를 일시 중지하거나 데이터 크롤링 횟수를 줄이도록 빈도를 변경하면 인벤토리 [세부 정보 뷰(Detailed View)]에 최신이 아닌 데이터가 표시될 수 있습니다.

필터를 사용하여 인벤토리 수집 쿼리

인벤토리 데이터를 수집했으면 AWS Systems Manager의 필터 기능을 사용하여 특정 필터 조건에 맞는 관리형 노드 목록을 쿼리할 수 있습니다.

인벤토리 필터를 기반으로 노드 쿼리

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 [Inventory]를 선택합니다.
3. Filter by resource groups, tags or inventory types(리소스 그룹, 태그 또는 인벤토리 유형별로 필터링) 섹션에서 필터 상자를 선택합니다. 미리 정의된 필터의 목록이 표시됩니다.
4. 필터를 적용할 속성을 선택합니다. 이 예에서는 [**AWS:Application**]을 선택합니다. 메시지가 나타나면 필터를 적용할 두 번째 속성을 선택합니다. 이 예에서는 [**AWS:Application.Name**]을 선택합니다.
5. 목록에서 구분 기호를 선택합니다. 예를 들어 Begin with(다음으로 시작)를 선택합니다. 필터에 텍스트 상자가 표시됩니다.
6. 텍스트 상자에 값을 입력합니다. 예를 들어 Amazon을 입력합니다(SSM Agent의 이름은 Amazon SSM Agent로 지정됨).
7. Enter을 누릅니다. Amazon이라는 단어로 시작하는 애플리케이션 이름이 들어 있는 관리형 노드 목록이 반환됩니다.

> **Note**
>
> 여러 필터를 결합하여 검색 범위를 좁힐 수 있습니다.

인벤토리 데이터 집계

AWS Systems Manager 인벤토리용 관리형 노드를 구성한 뒤에는 집계된 인벤토리 데이터 수를 볼 수 있습니다. 예를 들어 `AWS:Application` 인벤토리 유형을 수집하기 위해 관리형 노드를 수십, 수백 개 정도 구성했다고 가정하겠습니다. 그 경우 이 섹션의 정보를 이용하여 이 데이터를 수집하도록 구성된 노드가 몇 개인지 정확한 수를 알아볼 수 있습니다.

데이터 유형으로 집계하여 특정한 인벤토리 세부 정보를 확인할 수도 있습니다. 예를 들어 `AWS:InstanceInformation` 인벤토리 유형은 `Platform` 데이터 형식의 운영 체제 플랫폼 정보를 수집합니다. `Platform` 데이터 형식으로 데이터를 집계함으로써 Windows, Linux 및 macOS를 실행 노드가 각각 몇 개인지 빠르게 파악할 수 있습니다.

이 섹션의 절차에서는 AWS Command Line Interface(AWS CLI)를 사용하여 집계된 인벤토리 데이터 수를 보는 방법을 설명합니다. AWS Systems Manager 콘솔의 인벤토리 페이지에서 사전 구성된 집계 개수를 볼 수도 있습니다. 이렇게 사전 구성된 대시보드를 인벤토리 인사이트라고 부르며, 이를 통해 인벤토리 구성 문제를 클릭 한 번으로 해결할 수 있습니다.

다음은 인벤토리 데이터 집계 개수에 관한 중요한 세부 정보입니다.

- 인벤토리 데이터를 수집하도록 구성된 관리형 노드를 종료하면 Systems Manager는 인벤토리 데이터를 30일 동안 보관한 다음 삭제합니다. 실행 노드의 경우 30일이 경과한 인벤토리 데이터가 삭제됩니다. 인벤토리 데이터를 30일 이상 저장해야 하는 경우 AWS Config를 사용하여 이력을 기록하거나 정기적으로 데이터를 쿼리하여 Amazon Simple Storage Service(Amazon S3) 버킷에 업로드하면 됩니다.
- 노드가 이전에 `AWS:Network`와 같이 특정한 인벤토리 데이터 형식을 보고하도록 구성된 경우 나중에 구성을 변경하여 해당 유형의 수집을 중지하도록 하더라도, 해당 노드를 종료하고 30일이 지날 때까지 집계 개수에는 여전히 `AWS:Network` 데이터가 표시됩니다.

특정 AWS 계정의 모든 노드(및 그 계정에서 만들 수 있는 향후의 모든 노드)에서 인벤토리 데이터를 빠르게 구성하고 수집하는 방법에 대한 자세한 내용은 콘솔을 사용하여 수집 구성 섹션을 참조하세요.

주제

- 인벤토리 데이터를 집계하여 지정된 유형의 데이터를 수집하는 노드 개수를 확인
- 인벤토리 유형을 수집하도록 구성된 노드와 그렇지 않은 노드를 파악하기 위해 인벤토리 데이터를 그룹으로 집계

인벤토리 데이터를 집계하여 지정된 유형의 데이터를 수집하는 노드 개수를 확인

AWS Systems Manager GetInventory API 작업을 사용하여 하나 이상의 Inventory 유형 및 데이터 형식을 수집하는 노드의 집계 개수를 볼 수 있습니다. 예를 들어 `AWS:InstanceInformation` Inventory 유형을 사용하면 `AWS:InstanceInformation.PlatformType` 데이터 형식과 함께 GetInventory API 작업을 사용하여 운영 체제의 집계를 볼 수 있습니다. 다음은 예 AWS CLI 명령과 출력입니다.

```
aws ssm get-inventory --aggregators "Expression=AWS:InstanceInformation.PlatformType"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "Entities":[
      {
         "Data":{
            "AWS:InstanceInformation":{
               "Content":[
                  {
```

```
                    "Count":"7",
                    "PlatformType":"windows"
                },
                {
                    "Count":"5",
                    "PlatformType":"linux"
                }
            ]
        }
    }
  }
]
}
```

시작하기

개수를 보려는 인벤토리 유형과 데이터 유형을 결정합니다. AWS CLI에서 다음 명령을 실행하여 집계가 지원되는 인벤토리 유형 및 데이터 유형의 목록을 볼 수 있습니다.

```
aws ssm get-inventory-schema --aggregator
```

이 명령은 집계가 지원되는 인벤토리 유형 및 데이터 유형의 JSON 목록을 반환합니다. TypeName 필드에는 지원되는 인벤토리 유형이 표시됩니다. 그리고 Name 필드에는 데이터 유형이 각각 표시됩니다. 예를 들어 아래의 목록에서 `AWS:Application` 인벤토리 유형에는 `Name` 및 `Version` 데이터 형식이 포함됩니다.

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

다음 구문을 사용하는 명령을 만들어 목록의 인벤토리 유형에 대한 데이터를 집계할 수 있습니다.

```
aws ssm get-inventory --aggregators "Expression=InventoryType.DataType"
```

여기 몇 가지 예가 있습니다.

예 1

이 예에서는 해당 노드에서 사용하는 Windows 역할 개수를 집계합니다.

```
aws ssm get-inventory --aggregators "Expression=AWS:WindowsRole.Name"
```

예제 2

이 예에서는 해당 노드에 설치된 애플리케이션 개수를 집계합니다.

```
aws ssm get-inventory --aggregators "Expression=AWS:Application.Name"
```

여러 집계자 결합

명령 하나에 여러 가지 인벤토리 유형과 데이터 유형을 결합하면 데이터를 더 잘 이해할 수 있습니다. 여기 몇 가지 예가 있습니다.

예 1

이 예에서는 해당 노드에서 사용하는 운영 체제 유형의 개수를 집계합니다. 또한 운영 체제의 구체적인 이름도 반환합니다.

```
aws ssm get-inventory --aggregators '[{"Expression":
 "AWS:InstanceInformation.PlatformType", "Aggregators":[{"Expression":
 "AWS:InstanceInformation.PlatformName"}]}]'
```

예제 2

이 예에서는 해당 노드에서 실행 중인 애플리케이션 개수와 각 애플리케이션의 특정 버전을 집계합니다.

```
aws ssm get-inventory --aggregators '[{"Expression": "AWS:Application.Name",
 "Aggregators":[{"Expression": "AWS:Application.Version"}]}]'
```

원한다면 하나 이상의 인벤토리 유형과 데이터 유형으로 된 집계 표현식을 JSON 파일로 만들고 AWS CLI에서 그 파일을 호출할 수 있습니다. 이 파일의 JSON은 다음 구문을 사용해야 합니다.

```
[
       {
           "Expression": "string",
```

```
            "Aggregators": [
                {
                   "Expression": "string"
                }
            ]
        }
]
```

.json 파일 확장자로 파일을 저장해야 합니다.

다음은 여러 가지 인벤토리 유형과 데이터 유형을 사용하는 예제입니다.

```
[
        {
            "Expression": "AWS:Application.Name",
            "Aggregators": [
                {
                    "Expression": "AWS:Application.Version",
                    "Aggregators": [
                      {
                      "Expression": "AWS:InstanceInformation.PlatformType"
                      }
                    ]
                }
            ]
        }
]
```

다음 명령을 사용하여 AWS CLI에서 해당 파일을 호출합니다.

```
aws ssm get-inventory --aggregators file://file_name.json
```

명령은 다음과 같은 정보를 반환합니다.

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

인벤토리 유형을 수집하도록 구성된 노드와 그렇지 않은 노드를 파악하기 위해 인벤토리 데이터를 그룹으로 집계

Systems Manager Inventory의 그룹으로 관리형 노드 중 하나 이상의 Inventory 유형을 수집하도록 구성된 인스턴스 수와 그렇지 않은 인스턴스 수를 빠르게 파악할 수 있습니다. 그룹을 통해 하나 이상의 인벤토리 유형과 `exists` 연산자를 사용하는 필터를 지정합니다.

예를 들어, 다음 인벤토리 유형을 수집하도록 구성한 관리형 노드가 4개 있다고 가정하겠습니다.

- 노드 1: `AWS:Application`
- 노드 2: `AWS:File`
- 노드 3: `AWS:Application`, `AWS:File`
- 노드 4: `AWS:Network`

AWS CLI에서 다음 명령을 실행하여 AWS:Application과 AWS:File inventory 인벤토리 유형을 둘 다 수집하도록 구성된 노드가 몇 개인지 확인할 수 있습니다. 응답에는 이러한 두 가지 인벤토리 유형을 모두 수집하도록 구성되지 않은 노드 개수도 반환됩니다.

```
aws ssm get-inventory --aggregators
 'Groups=[{Name=ApplicationAndFile,Filters=[{Key=TypeName,Values=[AWS:Application],Type=Exists}
{Key=TypeName,Values=[AWS:File],Type=Exists}]}]'
```

명령의 응답은 AWS:Application 및 AWS:File 인벤토리 유형을 둘 다 수집하도록 구성된 관리형 노드가 하나뿐임을 보여줍니다.

```
{
   "Entities":[
      {
         "Data":{
            "ApplicationAndFile":{
               "Content":[
                  {
                     "notMatchingCount":"3"
                  },
                  {
                     "matchingCount":"1"
                  }
               ]
            }
         }
      }
   ]
}
```

> **Note**
>
> 그룹은 데이터 유형 개수를 반환하지 않습니다. 또한 결과를 심층 분석하여 인벤토리 유형을 수집하도록 구성된 노드와 그렇지 않은 노드 ID를 파악할 수도 없습니다.

원한다면 하나 이상의 인벤토리 유형으로 된 집계 표현식을 JSON 파일로 만들고 AWS CLI에서 그 파일을 호출할 수 있습니다. 이 파일의 JSON은 다음 구문을 사용해야 합니다.

```
{
   "Aggregators":[
```

```
      {
         "Groups":[
            {
               "Name":"Name",
               "Filters":[
                  {
                     "Key":"TypeName",
                     "Values":[
                        "Inventory_type"
                     ],
                     "Type":"Exists"
                  },
                  {
                     "Key":"TypeName",
                     "Values":[
                        "Inventory_type"
                     ],
                     "Type":"Exists"
                  }
               ]
            }
         ]
      }
   ]
}
```

.json 파일 확장자로 파일을 저장해야 합니다.

다음 명령을 사용하여 AWS CLI에서 해당 파일을 호출합니다.

```
aws ssm get-inventory --cli-input-json file://file_name.json
```

추가 예제

다음 예에서는 지정된 인벤토리 유형을 수집하도록 구성된 관리형 노드와 그렇지 않은 노드를 파악하기 위해 인벤토리 데이터를 집계하는 방법을 보여줍니다. 이 예제에서는 AWS CLI를 사용합니다. 각 예제에 전체 명령과 필터가 나와 있으므로 정보를 파일에 입력하는 방법을 선호한다면 명령줄에서 이 명령과 샘플 input.json 파일을 실행하면 됩니다.

예 1

이 예에서는 `AWS:Application` 또는 `AWS:File` 인벤토리 유형을 수집하도록 구성된 노드와 그렇지 않은 노드의 개수를 집계합니다.

AWS CLI에서 다음 명령을 실행합니다.

```
aws ssm get-inventory --aggregators
 'Groups=[{Name=ApplicationORFile,Filters=[{Key=TypeName,Values=[AWS:Application,
 AWS:File],Type=Exists}]}]'
```

파일을 사용하려면 다음 샘플을 복사하여 파일에 붙여 넣은 다음 이 파일을 input.json으로 저장합니다.

```
{
   "Aggregators":[
      {
         "Groups":[
            {
               "Name":"ApplicationORFile",
               "Filters":[
                  {
                     "Key":"TypeName",
                     "Values":[
                        "AWS:Application",
                        "AWS:File"
                     ],
                     "Type":"Exists"
                  }
               ]
            }
         ]
      }
   ]
}
```

AWS CLI에서 다음 명령을 실행합니다.

```
aws ssm get-inventory --cli-input-json file://input.json
```

명령은 다음과 같은 정보를 반환합니다.

```
{
   "Entities":[
      {
         "Data":{
            "ApplicationORFile":{
```

```
                "Content":[
                   {
                      "notMatchingCount":"1"
                   },
                   {
                      "matchingCount":"3"
                   }
                ]
             }
          }
       }
    ]
}
```

예제 2

이 예에서는 AWS:Application, AWS:File 및 AWS:Network 인벤토리 유형을 수집하도록 구성된 노드와 그렇지 않은 노드의 개수를 집계합니다.

AWS CLI에서 다음 명령을 실행합니다.

```
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}]}]'
```

파일을 사용하려면 다음 샘플을 복사하여 파일에 붙여 넣은 다음 이 파일을 input.json으로 저장합니다.

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

AWS CLI에서 다음 명령을 실행합니다.

```
aws ssm get-inventory --cli-input-json file://input.json
```

명령은 다음과 같은 정보를 반환합니다.

```
{
   "Entities":[
      {
         "Data":{
            "Application":{
               "Content":[
                  {
```

```
                    "notMatchingCount":"2"
                 },
                 {
                    "matchingCount":"2"
                 }
              ]
           },
           "File":{
              "Content":[
                 {
                    "notMatchingCount":"2"
                 },
                 {
                    "matchingCount":"2"
                 }
              ]
           },
           "Network":{
              "Content":[
                 {
                    "notMatchingCount":"3"
                 },
                 {
                    "matchingCount":"1"
                 }
              ]
           }
        }
     }
  ]
}
```

사용자 정의 인벤토리 작업

AWS Systems Manager Inventory 사용자 정의 인벤토리를 생성하여 원하는 메타데이터를 노드에 할당할 수 있습니다. 예를 들어 데이터 센터에서 여러 랙에 다수의 서버를 설치하여 관리하고 있으며, 이 서버들은 Systems Manager 관리형 노드로 구성되었다고 가정하겠습니다. 현재는 서버 랙 위치에 대한 정보를 스프레드시트에 저장하고 있습니다. 하지만 사용자 정의 인벤토리를 사용하면 각 노드의 랙 위치를 노드에 대한 메타데이터로 지정할 수 있습니다. Systems Manager를 사용하여 인벤토리를 수집할 경우에는 메타데이터가 다른 인벤토리 메타데이터와 함께 수집됩니다. 그런 다음 리소스 데이터 동기화를 사용하여 모든 인벤토리 메타데이터를 중앙 Amazon S3 버킷으로 포팅하고 데이터를 쿼리할 수 있습니다.

> **Note**
>
> Systems Manager는 AWS 계정당 최대 20개의 사용자 정의 인벤토리 유형을 지원합니다.

노드에 사용자 정의 인벤토리를 할당하려면 시연: 관리형 노드에 사용자 지정 인벤토리 메타데이터 할당에 설명된 대로 Systems Manager PutInventory API 작업을 사용할 수 있습니다. 아니면, 사용자 정의 인벤토리 JSON 파일을 생성하여 노드에 업로드하는 방법이 있습니다. 이번 단원에서는 JSON 파일을 생성하는 방법에 대해서 설명합니다.

다음 예제 JSON 파일은 온프레미스 서버의 랙 정보를 지정하는 사용자 지정 인벤토리를 포함하고 있습니다. 이 예제는 `Content` 아래에 데이터를 설명하는 여러 항목을 포함하는 한 사용자 지정 인벤토리 데이터 형식(`"TypeName": "Custom:RackInformation"`)을 지정합니다.

```
{
    "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"
      }
 }
```

다음 예제에 나온 것처럼 `Content` 섹션에 개별 항목을 지정할 수도 있습니다.

```
{
"SchemaVersion": "1.0",
"TypeName": "Custom:PuppetModuleInfo",
    "Content": [{
        "Name": "puppetlabs/aws",
        "Version": "1.0"
      },
      {
        "Name": "puppetlabs/dsc",
        "Version": "2.0"
      }
    ]
}
```

사용자 정의 인벤토리에 사용되는 JSON 스키마는 SchemaVersion TypeName 및 Content 섹션이 필요하지만 이 섹션의 정보를 정의할 수 있습니다.

```
{
    "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 값은 100자로 제한됩니다. 또한 TypeName 값은 대문자로 된 Custom 단어로 시작해야 합니다. 예를 들면 Custom:PuppetModuleInfo입니다. 따라서 이 예시(CUSTOM:PuppetModuleInfo, custom:PuppetModuleInfo)에서는 예외가 발생합니다.

Content 섹션에는 속성과 *data*가 포함됩니다. 이들 항목은 대/소문자를 구분하지 않습니다. 하지만 속성을 정의하는 경우에는(예: "Vendor": "DELL") 사용자 정의 인벤토리 파일에서 이 속성을 일관적으로 참조해야 합니다. 한 파일에서 "Vendor": "DELL"(vendor에서 대문자 “V” 사용)을 지정한 다음 다른 파일에서 "vendor": "DELL"(vendor에서 소문자 “v” 사용)을 지정하면 시스템이 오류 메시지를 반환합니다.

> **Note**
>
> 파일을 .json 확장자로 저장해야 하며 정의한 인벤토리는 문자열 값으로만 구성되어야 합니다.

파일을 생성한 후에는 노드에 저장해야 합니다. 다음은 노드에서 사용자 정의 인벤토리 JSON 파일을 저장해야 하는 위치를 나타낸 표입니다.

운영 체제	경로
Linux	/var/lib/amazon/ssm/*node-id*/inventory/custom
macOS	/opt/aws/ssm/data/ *node-id*/ inventory/custom

운영 체제	경로
Windows	%SystemDrive%\ProgramData\Amazon\SSM\InstanceData*node-id*\inventory\custom

사용자 지정 인벤토리를 사용하는 방법의 예제는 EC2 Systems Manager 사용자 지정 인벤토리 유형을 사용하여 플릿의 디스크 사용률 얻기를 참조하십시오.

사용자 지정 인벤토리 삭제

DeleteInventory API 작업을 사용하여 사용자 정의 Inventory 유형 및 해당 유형과 연결된 데이터를 삭제할 수 있습니다. AWS Command Line Interface(AWS CLI)를 사용하여 delete-inventory 명령을 호출함으로써 인벤토리 유형의 모든 데이터를 삭제합니다. `SchemaDeleteOption`과 함께 delete-inventory 명령을 호출하여 사용자 지정 인벤토리 유형을 삭제합니다.

> **Note**
>
> 인벤토리 유형을 인벤토리 스키마라고도 합니다.

`SchemaDeleteOption` 파라미터에는 다음과 같은 옵션이 있습니다.

- DeleteSchema: 이 옵션은 지정한 사용자 지정 유형 및 해당 유형과 연결된 모든 데이터를 삭제합니다. 원한다면 나중에 스키마를 다시 만들 수 있습니다.
- [DisableSchema]: 이 옵션을 선택하면 시스템에서 현재 버전을 해제하고, 현재 버전에 대한 모든 데이터를 삭제하고, 해당 버전이 해제된 버전과 같거나 이전 버전일 경우 새 데이터를 모두 무시합니다. 해제된 버전보다 높은 버전에 대해 PutInventory 작업을 호출함으로써 이 Inventory 유형을 다시 허용할 수 있습니다.

AWS CLI를 사용하여 사용자 정의 인벤토리를 삭제하거나 해제하려면

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.
2. `dry-run` 옵션으로 시스템에서 삭제할 데이터를 확인하려면 다음 명령을 실행하십시오. 이 명령은 어떤 데이터도 삭제하지 않습니다.

```
aws ssm delete-inventory --type-name "Custom:custom_type_name" --dry-run
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "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"
}
```

인벤토리 삭제 요약을 이해하는 방법은 인벤토리 삭제 요약 이해하기 섹션을 참조하세요.

3. 다음 명령을 실행하여 사용자 지정 인벤토리 유형의 데이터를 모두 삭제합니다.

```
aws ssm delete-inventory --type-name "Custom:custom_type_name"
```

> **Note**
>
> 이 명령의 출력에는 삭제 진행 상황이 표시되지 않습니다. 그러므로 [총 개수(TotalCount)]와 [남은 개수(Remaining Count)]는 항상 동일합니다. 아직 아무것도 삭제되지 않았기 때문입니다. 이 주제 후반부에서 설명하는 방법으로 describe-inventory-deletions 명령을 사용하여 삭제 진행 상황을 표시할 수 있습니다.

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "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"
}
```

그러면 Systems Manager Inventory 서비스에서 지정된 사용자 정의 Inventory 유형의 데이터가 모두 삭제됩니다.

4. 다음 명령을 실행합니다. 이 명령은 현재 버전을 해제하고, 해당하는 데이터를 모두 삭제하고, 해제된 버전 이하의 버전에서는 신규 데이터를 모두 무시하는 등의 작업을 현재 버전의 인벤토리 유형에 대해 수행합니다.

```
aws ssm delete-inventory --type-name "Custom:custom_type_name" --schema-delete-
option "DisableSchema"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "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"
}
```

다음 명령으로 해제된 인벤토리 유형을 볼 수 있습니다.

```
aws ssm get-inventory-schema --type-name Custom:custom_type_name
```

5. 다음 명령을 실행하여 인벤토리 유형을 삭제합니다.

```
aws ssm delete-inventory --type-name "Custom:custom_type_name" --schema-delete-
option "DeleteSchema"
```

지정된 사용자 지정 유형의 스키마와 인벤토리 데이터가 모두 삭제됩니다.

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "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"
}
```

삭제 상태 보기

`describe-inventory-deletions` AWS CLI 명령으로 삭제 작업의 상태를 확인할 수 있습니다. 특정한 삭제 작업의 상태를 보려면 삭제 ID를 지정하십시오. 아니면 삭제 ID를 생략하고 최근 30일간 실행된 전체 삭제 작업 목록을 볼 수도 있습니다.

1. 다음 명령을 실행하여 삭제 작업의 상태를 확인합니다. 시스템에서는 delete-inventory 요약을 통해 삭제 ID를 반환했습니다.

   ```
   aws ssm describe-inventory-deletions --deletion-id system_generated_deletion_ID
   ```

   시스템에서 최신 상태를 반환합니다. 삭제 작업이 아직 종료되지 않았을 수 있습니다. 시스템은 다음과 같은 정보를 반환합니다.

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

   삭제 작업이 성공하면 `LastStatusMessage` 상태: 삭제 성공이 반환됩니다.

```
{"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. 다음 명령을 실행하여 최근 30일간 실행된 전체 삭제 작업 목록을 볼 수도 있습니다.

```
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"
```

인벤토리 삭제 요약 이해하기

인벤토리 삭제 요약을 이해하려면 다음 예제를 생각해 보십시오. 사용자가 노드 세 개에 Custom:RackSpace 인벤토리를 할당했습니다. 인벤토리 항목 1과 2는 사용자 지정 유형 버전 1.0을 사용합니다("SchemaVersion":"1.0"). 인벤토리 항목 3은 사용자 지정 유형 버전 2.0을 사용합니다("SchemaVersion":"2.0").

RackSpace 사용자 지정 인벤토리 1

```
{
   "CaptureTime":"2018-02-19T10:48:55Z",
   "TypeName":"CustomType:RackSpace",
   "InstanceId":"i-1234567890",
   "SchemaVersion":"1.0"   "Content":[
      {
         content of custom type omitted
      }
   ]
}
```

RackSpace 사용자 지정 인벤토리 2

```
{
   "CaptureTime":"2018-02-19T10:48:55Z",
   "TypeName":"CustomType:RackSpace",
   "InstanceId":"i-1234567891",
   "SchemaVersion":"1.0"   "Content":[
      {
         content of custom type omitted
      }
   ]
}
```

RackSpace 사용자 지정 인벤토리 3

```
{
   "CaptureTime":"2018-02-19T10:48:55Z",
   "TypeName":"CustomType:RackSpace",
   "InstanceId":"i-1234567892",
   "SchemaVersion":"2.0"   "Content":[
      {
         content of custom type omitted
      }
   ]
}
```

사용자는 다음 명령을 실행하여 삭제할 데이터를 미리 봅니다.

```
aws ssm delete-inventory --type-name "Custom:RackSpace" --dry-run
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "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"
}
```

사용자는 다음 명령을 실행하여 Custom:RackSpace 인벤토리를 삭제합니다.

> **Note**
>
> 이 명령의 출력에는 삭제 진행 상황이 표시되지 않습니다. 그러므로 `TotalCount`와 `RemainingCount`는 항상 동일합니다. 아직 아무것도 삭제되지 않았기 때문입니다. `describe-inventory-deletions` 명령을 사용하여 삭제 진행 상황을 표시합니다.

```
aws ssm delete-inventory --type-name "Custom:RackSpace"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
```

```
   "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"
}
```

EventBridge에서 인벤토리 삭제 작업 보기

사용자가 사용자 정의 인벤토리를 삭제할 때마다 이벤트를 생성하도록 Amazon EventBridge를 구성할 수 있습니다. EventBridge는 사용자 정의 인벤토리 삭제 작업에 대해 다음과 같은 3가지 유형의 이벤트를 제공합니다.

- Delete action for an instance(인스턴스에 대한 삭제 작업): 지정된 관리형 노드에 대한 사용자 지정 인벤토리가 무사히 삭제되었는지 여부.
- Delete action summary(삭제 작업 요약): 삭제 작업의 요약.
- [해제된 사용자 정의 Inventory 유형 경고(Warning for turned off custom inventory type)]: 이전에 해제한 사용자 정의 Inventory 유형 버전에 PutInventory API 작업을 호출할 경우 등장하는 경고 이벤트.

다음은 각 이벤트의 예입니다.

Delete action for an instance(인스턴스에 대한 삭제 작업)

```
{
   "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"
   }
}
```


Delete action summary(삭제 작업 요약)


```
{
   "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"
   }
}
```

해제된 사용자 정의 인벤토리 유형 경고

```
{
   "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"
   }
}
```

사용자 정의 인벤토리 삭제 작업에 EventBridge 규칙을 생성하려면 다음 절차를 따릅니다. 이 절차는 사용자 정의 인벤토리 삭제 작업에 대한 알림을 Amazon SNS 주제에 전송하도록 규칙을 생성하는 방법입니다. 시작하기 전에 Amazon SNS 주제가 있는지 확인하고 없으면 새로 생성합니다. 자세한 내용은 Amazon Simple Notification Service Developer Guide의 Getting Started를 참조하세요.

인벤토리 삭제 작업에 EventBridge를 구성하려면

1. https://console.aws.amazon.com/events/에서 Amazon EventBridge 콘솔을 엽니다.
2. 탐색 창에서 규칙을 선택합니다.
3. 규칙 생성을 선택합니다.
4. 규칙에 대해 이름과 설명을 입력하십시오.

   규칙은 동일한 지역과 동일한 이벤트 버스의 다른 규칙과 동일한 이름을 가질 수 없습니다.
5. 이벤트 버스에서 이 규칙과 연결할 이벤트 버스를 선택합니다. 이 규칙이 자신의 AWS 계정에서 오는 일치하는 이벤트에 응답하도록 하려면 default(기본)를 선택합니다. 계정의 AWS 서비스이(가) 이벤트를 출력하면 항상 계정의 기본 이벤트 버스로 이동합니다.

6. 규칙 유형에서 이벤트 패턴이 있는 규칙을 선택합니다.
7. 다음을 선택합니다.
8. 이벤트 소스(Event source)에서 AWS 이벤트 또는 EventBridge 파트너 이벤트(Events or EventBridge partner events)를 선택합니다.
9. 이벤트 패턴(Event pattern)섹션에서 이벤트 패턴 양식(Event pattern form)을 선택합니다.
10. 이벤트 소스에서 AWS 서비스를 선택합니다.
11. AWS service(서비스)에서 Systems Manager를 선택합니다.
12. [이벤트 유형(Event type)]으로 [인벤토리]를 선택합니다.
13. Specific detail type(s)(특정 상세 유형)에서 Inventory Resource State Change(인벤토리 리소스 상태 변경)을 선택합니다.
14. 다음을 선택합니다.
15. 대상 유형에서 AWS서비스를 선택합니다.
16. Select a target(대상 선택)에서 SNS topic(SNS 주제)을 선택한 후 Topic(주제)에서 주제를 선택합니다.
17. Additional settings(추가 설정) 섹션의 Configure target input(대상 입력 구성)에서 Matched event(일치하는 이벤트)가 선택되었는지 확인합니다.
18. 다음을 선택합니다.
19. (선택 사항) 규칙에 대해 하나 이상의 태그를 입력하십시오. 자세한 내용은 Amazon EventBridge User Guide의 Tagging Your Amazon EventBridge Resources를 참조하세요.
20. 다음을 선택합니다.
21. 규칙의 세부 정보를 검토하고 규칙 생성을 선택합니다.

인벤토리 이력 및 변경 사항 추적 보기

AWS Config를 사용하여 모든 관리형 노드에 대한 AWS Systems Manager Inventory 기록 및 변경 사항 추적을 확인할 수 있습니다. AWS Config에서는 AWS 계정에 있는 AWS 리소스의 구성을 자세히 보여줍니다. 이러한 보기에는 리소스 간에 어떤 관계가 있는지와 리소스가 과거에 어떻게 구성되었는지도 포함되므로, 시간이 지나면서 구성과 관계가 어떻게 변하는지 확인할 수 있습니다. 인벤토리 기록과 변경 사항 추적을 보려면 AWS Config에서 다음 리소스를 설정해야 합니다.

- SSM:ManagedInstanceInventory
- SSM:PatchCompliance

- SSM:AssociationCompliance
- SSM:FileData

> **Note**
>
> Inventory 기록 및 변경 추적에 대한 다음과 같은 중요한 세부 정보에 유의합니다.
>
> - AWS Config를 사용하여 시스템의 변경 사항을 추적하는 경우 AWS Config(`SSM:FileData`)에서 파일 변경 사항을 볼 수 있도록 `AWS:File` 메타데이터를 수집하도록 Systems Manager Inventory를 구성해야 합니다. 그렇지 않으면 AWS Config가 시스템의 파일 변경 사항을 추적하지 않습니다.
> - SSM:PatchCompliance 및 SSM:AssociationCompliance를 설정하여 Systems Manager Patch Manager 패치와 Systems Manager State Manager 연결의 Compliance 기록 및 변경 사항 추적을 확인할 수 있습니다. 이러한 리소스의 규정 준수 관리에 대한 자세한 내용은 Compliance 작업 섹션을 참조하세요.

다음 절차에서는 AWS Command Line Interface(AWS CLI)를 사용하여 AWS Config에서 인벤토리 기록 및 변경 사항 추적의 기록을 설정하는 방법을 설명합니다. AWS Config에서 이러한 리소스를 선택하고 구성하는 방법에 대한 자세한 내용은 AWS Config Developer Guide의 Selecting Which Resources AWS Config Records를 참조하세요. AWS Config 요금에 대한 자세한 내용은 요금을 참조하세요.

시작하기 전에

AWS Config에 AWS Identity and Access Management(IAM) 권한이 있어야 Systems Manager 리소스에 대한 구성 세부 정보를 가져올 수 있습니다. 다음 절차에서는 IAM 역할의 Amazon 리소스 이름(ARN)을 지정하여 Systems Manager 리소스에 대한 권한을 AWS Config에 부여해야 합니다. `AWS_ConfigRole` 관리형 정책을 AWS Config에 할당한 IAM 역할에 연결할 수 있습니다. 이 역할에 대한 자세한 내용은 AWS Config 개발자 안내서의 AWS 관리형 정책: AWS_ConfigRole을 참조하세요. IAM 역할을 생성하고 해당 역할에 `AWS_ConfigRole` 관리형 정책을 할당하는 방법에 대한 자세한 내용은 IAM 사용 설명서의 AWS 서비스에 권한을 위임하는 역할 생성을 참조하세요.

AWS Config에서 인벤토리 기록 및 변경 사항 추적의 기록을 설정하려면

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.

2. 다음 JSON 샘플을 복사하여 단순한 텍스트 파일에 붙여 넣고 recordingGroup.json으로 저장합니다.

```
{
   "allSupported":false,
   "includeGlobalResourceTypes":false,
   "resourceTypes":[
      "AWS::SSM::AssociationCompliance",
      "AWS::SSM::PatchCompliance",
      "AWS::SSM::ManagedInstanceInventory",
      "AWS::SSM::FileData"
   ]
}
```

3. 다음 명령을 실행하여 recordingGroup.json 파일을 AWS Config에 로드합니다.

```
aws configservice put-configuration-recorder --configuration-recorder
 name=myRecorder,roleARN=arn:aws:iam::123456789012:role/myConfigRole --recording-
group file://recordingGroup.json
```

4. 다음 명령을 실행하여 인벤토리 이력 및 변경 사항 추적의 기록을 시작합니다.

```
aws configservice start-configuration-recorder --configuration-recorder-
name myRecorder
```

기록 및 변경 내용 추적을 구성한 후 Systems Manager 콘솔에서 AWS Config 버튼을 선택하여 특정 관리형 노드에 대한 기록을 드릴다운할 수 있습니다. 관리형 인스턴스(Managed Instances) 페이지 또는 Inventory 페이지에서 AWS Config 버튼에 액세스할 수 있습니다. 모니터 크기에 따라 페이지 오른쪽으로 스크롤해야 이 버튼이 보일 수도 있습니다.

데이터 수집 중지 및 인벤토리 데이터 삭제

더 이상 AWS Systems Manager Inventory를 사용하여 AWS 리소스에 대한 메타데이터를 보고 싶지 않다면, 데이터 수집을 중지하고 이미 수집된 데이터를 삭제할 수 있습니다. 이 섹션에는 다음 정보가 포함됩니다.

주제

- 데이터 수집 중지
- Inventory 리소스 데이터 동기화 삭제

데이터 수집 중지

인벤토리 데이터를 수집하도록 Systems Manager를 처음 구성하면 시스템에서 일정 및 메타데이터를 수집할 리소스를 정의하는 State Manager 연결을 생성합니다. `AWS-GatherSoftwareInventory` 문서를 사용하는 State Manager 연결을 삭제하여 데이터 수집을 중지할 수 있습니다.

Inventory 연결을 삭제하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 State Manager를 선택합니다.
3. `AWS-GatherSoftwareInventory` 문서를 사용하는 연결을 선택한 다음 삭제(Delete)를 선택합니다.
4. `AWS-GatherSoftwareInventory` 문서를 사용하는 나머지 연결에 대해 3단계를 반복합니다.

Inventory 리소스 데이터 동기화 삭제

더 이상 AWS Systems Manager Inventory를 사용하여 AWS 리소스에 대한 메타데이터를 보고 싶지 않다면, 인벤토리 데이터 수집에 사용되는 리소스 데이터 동기화를 삭제하는 것이 좋습니다.

Inventory 리소스 데이터 동기화를 삭제하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 [Inventory]를 선택합니다.
3. 리소스 데이터 동기화(Resource data sync)를 선택합니다.
4. 목록에서 동기화를 선택합니다.

> **Important**
>
> Inventory에 사용되는 동기화를 선택해야 합니다. Systems Manager는 여러 기능에 대한 리소스 데이터 동기화를 지원합니다. 잘못된 동기화를 선택하면 Systems Manager Explorer 또는 Systems Manager Compliance의 데이터 집계가 중단될 수 있습니다.

5. 삭제(Delete)를 선택합니다.
6. 삭제할 나머지 리소스 데이터 동기화에 대해 이 단계를 반복합니다.
7. 데이터가 저장된 Amazon Simple Storage Service(Amazon S3) 버킷을 삭제합니다. Amazon S3 버킷 삭제에 대한 자세한 내용은 버킷 삭제를 참조하세요.

Systems Manager Inventory 시연

다음 시연에 따라 AWS Systems Manager Inventory를 사용하여 인벤토리 데이터를 수집하고 관리합니다. 처음에는 관리형 노드가 테스트 환경일 때 연습하는 것이 좋습니다.

시작하기 전 준비 사항

이 시연을 시작하기 전에 먼저 다음 작업을 완료합니다.

- 인벤토리로 만들고 싶은 노드에서 AWS Systems Manager SSM Agent를 업데이트합니다. SSM Agent 최신 버전을 실행하여 지원되는 모든 인벤토리 유형에 대한 메타데이터를 수집할 수 있는지 확인합니다. SSM Agent를 사용해 State Manager 를 업데이트하는 방법에 대한 자세한 내용은 시연: SSM Agent(CLI) 자동 업데이트 섹션을 참조하세요.
- 하이브리드 및 멀티클라우드 환경의 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 및 비 EC2 시스템에 대한 설정 요구 사항을 완료했는지 확인합니다. 자세한 설명은 AWS Systems Manager 설정을 참조하세요.
- (선택 사항) JSON 파일을 생성하여 사용자 정의 인벤토리를 수집합니다. 자세한 내용은 사용자 정의 인벤토리 작업 단원을 참조하십시오.

목차

시연: 관리형 노드에 사용자 지정 인벤토리 메타데이터 할당

다음 절차는 AWS Systems Manager PutInventory API 작업을 사용하여 관리형 노드에 사용자 정의 인벤토리 메타데이터를 할당하는 프로세스를 안내합니다. 이번 예에서는 노드에 랙 위치 정보를 할당합니다. 사용자 지정 인벤토리에 대한 자세한 내용은 사용자 정의 인벤토리 작업 섹션을 참조하세요.

사용자 지정 인벤토리 메타데이터를 노드에 할당

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.
2. 다음 명령을 실행하여 노드에 랙 위치 정보를 할당합니다.

   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. 다음 명령을 실행하여 이 노드의 사용자 정의 인벤토리 항목을 확인합니다.

```
aws ssm list-inventory-entries --instance-id ID --type-name "Custom:RackInfo"
```

시스템은 다음과 같은 정보로 응답합니다.

```
{
    "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. 다음 명령을 실행하여 사용자 정의 인벤토리 스키마를 확인합니다.

```
aws ssm get-inventory-schema --type-name Custom:RackInfo
```

시스템은 다음과 같은 정보로 응답합니다.

```
{
    "Schemas": [
        {
            "TypeName": "Custom:RackInfo",
            "Version": "1.0",
            "Attributes": [
```

```
                {
                    "DataType": "STRING",
                    "Name": "RackLocation"
                }
            ]
        }
    ]
}
```

연습: CLI를 사용하여 인벤토리를 위한 관리형 노드 구성

다음 절차에서는 관리형 노드에서 메타데이터를 수집하도록 AWS Systems Manager 인벤토리를 구성하는 프로세스를 설명합니다. 인벤토리 수집을 구성할 때 Systems Manager State Manager 연결을 생성하는 것으로 시작합니다. Systems Manager는 연결이 실행될 때 인벤토리 데이터를 수집합니다. 연결을 먼저 생성하지 않고 Systems Manager Run Command 등을 사용하여 `aws:softwareInventory` 플러그인을 호출하려고 하면 시스템이 다음 오류를 반환합니다.

`The aws:softwareInventory plugin can only be invoked via ssm-associate.`

> **Note**
>
> 노드에는 한 번에 하나의 인벤토리 연결만 구성할 수 있습니다. 둘 이상의 인벤토리 연결로 노드를 구성할 경우 연결이 실행되지 않으며 인벤토리 데이터가 수집되지 않습니다.

빠르게 인벤토리에 대한 모든 관리형 노드 구성(CLI)

AWS 계정 및 현재 리전의 모든 관리형 노드에서 인벤토리 데이터를 수집하도록 빠르게 구성할 수 있습니다. 이를 전역 인벤토리 연결 생성이라고 합니다. AWS CLI를 사용하여 전역 인벤토리 연결을 생성하려면 다음 절차에 나와 있듯이 `instanceIds` 값에 와일드카드 옵션을 사용하십시오.

AWS 계정 및 현재 리전의 모든 관리형 노드에서 인벤토리 구성(CLI)

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.
2. 다음 명령을 실행합니다.

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**
>
> 이 명령은 Inventory가 Windows 레지스트리 또는 파일에 대한 메타데이터를 수집하는 것을 허용하지 않습니다. 이러한 데이터 형식을 인벤토리하려면 다음 절차를 사용하십시오.

관리형 노드에서 수동으로 인벤토리 구성(CLI)

다음 절차를 사용하여 수동으로 인스턴스 ID 또는 태그를 사용해 관리형 노드에 AWS Systems Manager 인벤토리를 구성합니다.

수동으로 관리형 노드에서 인벤토리 구성(CLI)

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.

2. 다음 명령을 실행하여 노드에서 Systems Manager Inventory를 실행하는 State Manager 연결을 생성합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다. 이 명령은 6시간마다 서비스를 실행하고 노드에서 네트워크 구성, Windows 업데이트, 애플리케이션 메타데이터를 수집하도록 구성합니다.

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\": \"DOC-EXAMPLE-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\": \"DOC-EXAMPLE-BUCKET\",
 \"OutputS3KeyPrefix\": \"Test\" } }" ^
--parameters "networkConfig=Enabled,windowsUpdates=Enabled,applications=Enabled"
```

시스템은 다음과 같은 정보로 응답합니다.

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

대규모 노드 그룹을 대상으로 EC2 태그와 Targets 파라미터를 사용할 수 있습니다. 다음 예를 참조하세요.

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\": \"DOC-EXAMPLE-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\": \"DOC-EXAMPLE-BUCKET\", \"OutputS3KeyPrefix\": \"Test
\" } }" ^
--parameters "networkConfig=Enabled,windowsUpdates=Enabled,applications=Enabled"
```

표현식과 함께 `files` 및 `windowsRegistry` 유형의 인벤토리를 사용하여 Windows Server 노드에서 파일 및 Windows 레지스트리 키를 인벤토리로 만들 수도 있습니다. 이러한 인벤토리 유형에 대한 자세한 내용은 파일 및 Windows 레지스트리 인벤토리 관련 작업를 참조하십시오.

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. 다음 명령을 실행하여 연결 상태를 확인합니다.

```
aws ssm describe-instance-associations-status --instance-id an_instance_ID
```

시스템은 다음과 같은 정보로 응답합니다.

```
{
"InstanceAssociationStatusInfos": [
         {
            "Status": "Pending",
            "DetailedStatus": "Associated",
            "Name": "reInvent2016PolicyDocumentTest",
            "InstanceId": "i-1a2b3c4d5e6f7g",
            "AssociationId": "1a2b3c4d5e6f7g-1a2b3c-1a2b3c-1a2b3c-1a2b3c4d5e6f7g",
            "DocumentVersion": "1"
        }
]
}
```

시연: 리소스 데이터 동기화를 사용하여 인벤토리 데이터 집계

다음 시연에서는 AWS Command Line Interface(AWS CLI)를 사용하여 AWS Systems Manager Inventory에 대한 리소스 데이터 동기화 구성을 생성하는 방법을 설명합니다. 리소스 데이터 동기화는 모든 관리형 노드의 인벤토리 데이터를 중앙의 Amazon Simple Storage Service(Amazon S3) 버킷으로 자동 포팅합니다. 새로운 인벤토리 데이터가 발견될 때마다 동기화를 통해 중앙의 Amazon S3 버킷 데이터가 자동으로 업데이트됩니다.

이번 시연에서는 Amazon Athena 및 Amazon QuickSight를 사용하여 수집한 데이터에 대해 쿼리를 실행하고 분석하는 방법에 대해서 설명합니다. AWS Management Console에서 Systems Manager를 사용하여 리소스 데이터 동기화를 생성하는 방법에 대한 자세한 내용은 Inventory의 리소스 데이터 동기화 구성 섹션을 참조하세요. AWS Management Console의 Systems Manager를 사용하여 여러 AWS 리전 및 계정의 인벤토리를 쿼리하는 방법에 대한 자세한 내용은 여러 리전 및 계정에서 인벤토리 데이터 쿼리 섹션을 참조하세요.

> **Note**
>
> 이 연습에는 AWS Key Management Service(AWS KMS)를 사용하여 동기화를 암호화하는 방법에 대한 정보가 포함되어 있습니다. 암호화는 옵션이므로 인벤토리에서는 사용자별 데이터, 독점 데이터 또는 민감한 데이터를 수집하지 않습니다. AWS KMS에 대한 자세한 내용은 AWS Key Management Service Developer Guide를 참조하세요.

시작하기 전 준비 사항

이 섹션에서 연습을 시작하기 전에 다음 작업을 검토하거나 완료합니다.

- 관리형 노드에서 인벤토리 데이터를 수집합니다. 이번 시연의 Amazon Athena 및 Amazon QuickSight 섹션 목적에 따라 애플리케이션 데이터의 수집을 권장합니다. 인벤토리 데이터를 수집하는 방법에 대한 자세한 내용은 인벤토리 수집 구성 또는 연습: CLI를 사용하여 인벤토리를 위한 관리형 노드 구성 섹션을 참조하세요.
- (선택 사항) 인벤토리 데이터가 AWS Key Management Service(AWS KMS) 암호화를 사용하는 Amazon Simple Storage Service(Amazon S3) 버킷에 저장된 경우, AWS KMS 암호화를 위해 IAM 계정과 `Amazon-GlueServiceRoleForSSM` 서비스 역할 또한 구성해야 합니다. IAM 계정과 이 역할을 구성하지 않는 경우 콘솔의 세부 정보 보기(Detailed View) 탭을 선택하면 Systems Manager가 `Cannot load Glue tables`를 표시합니다. 자세한 내용은 (선택 사항) AWS KMS 암호화 데이터를 볼 수 있는 권한 구성 단원을 참조하십시오.
- (선택 사항) AWS KMS를 사용하여 리소스 데이터 동기화를 암호화하려면 다음 정책을 포함하는 새 키를 생성하거나 기존 키를 업데이트하고 이 정책을 키에 추가해야 합니다.

```
{
    "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/
*"
                }
            }
        }
    ]
```

```
}
```

인벤토리의 리소스 데이터 동기화를 생성하려면

1. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
2. 집계된 인벤토리 데이터를 저장할 버킷을 생성합니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 버킷 생성을 참조하세요. 버킷 이름과 버킷을 생성한 AWS 리전을 따로 적어둡니다.
3. 버킷을 생성한 후 권한 탭, 버킷 정책을 차례로 선택합니다.
4. 다음 버킷 정책을 복사하여 정책 편집기에 붙여 넣습니다. DOC-EXAMPLE-BUCKET 및 *account-id*를 사용자가 생성한 Amazon S3 버킷 이름 및 유효한 AWS 계정 ID로 바꿉니다. 여러 계정을 추가할 때 계정마다 조건 문자열과 ARN을 추가합니다. 계정 하나를 추가할 때 예제에서 추가 자리 표시자를 제거합니다. 원할 경우 *bucket-prefix*를 Amazon S3 접두사(하위 디렉터리) 이름으로 바꿉니다. 접두사를 생성하지 않았으면 정책의 ARN에서 *bucket-prefix/*를 제거합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": " SSMBucketDelivery",
      "Effect": "Allow",
      "Principal": {
        "Service": "ssm.amazonaws.com"
      },
      "Action": "s3:PutObject",
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-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. (선택 사항) 동기화를 암호화하려면 앞 단계에 나열된 정책에 다음 조건을 추가해야 합니다. `StringEquals` 섹션에 추가하세요.

```
"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"
```

예:

```
"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. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.

7. (옵션) 동기화를 암호화하려면 다음 명령을 실행하여 버킷 정책에서 AWS KMS 키 요구 사항을 실행하는지 확인합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   Linux & macOS

```
aws s3 cp ./A_file_in_the_bucket s3://DOC-EXAMPLE-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://DOC-EXAMPLE-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. 다음 명령을 실행하여 이 절차를 시작할 때 생성한 Amazon S3 버킷에서 리소스 데이터 동기화 구성을 생성합니다. 이 명령은 로그인되어 있는 AWS 리전에서 동기화를 생성합니다.

> **Note**
>
> 동기화와 대상 Amazon S3 버킷이 서로 다른 리전에 위치하는 경우에는 데이터 전송 요금이 부과될 수 있습니다. 자세한 내용은 Amazon S3 요금을 참조하세요.

Linux & macOS

```
aws ssm create-resource-data-sync \
--sync-name a_name \
--s3-destination "BucketName=DOC-EXAMPLE-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=DOC-EXAMPLE-BUCKET,Prefix=prefix_name,
 if_specified,SyncFormat=JsonSerDe,Region=bucket_region"
```

`region` 파라미터를 사용하여 동기화 구성을 생성할 리전을 지정할 수 있습니다. 다음 예에서는 us-west-1 리전의 인벤토리 데이터가 us-west-2 리전에 속한 Amazon S3 버킷에서 동기화됩니다.

Linux & macOS

```
aws ssm create-resource-data-sync \
    --sync-name InventoryDataWest \
```

```
    --s3-destination "BucketName=DOC-EXAMPLE-
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=DOC-EXAMPLE-
BUCKET,Prefix=HybridEnv,SyncFormat=JsonSerDe,Region=us-west-2" ^ --region us-
west-1
```

(선택 사항) AWS KMS를 사용하여 동기화를 암호화하려면 다음 명령을 실행하여 동기화를 생성합니다. 동기화를 암호화한 경우 AWS KMS 키와 Amazon S3 버킷이 같은 리전에 있어야 합니다.

Linux & macOS

```
aws ssm create-resource-data-sync \
--sync-name sync_name \
--s3-destination "BucketName=DOC-EXAMPLE-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=DOC-EXAMPLE-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. 다음 명령을 실행하여 동기화 구성 상태를 확인합니다.

```
aws ssm list-resource-data-sync
```

동기화 구성을 다른 리전에 생성한 경우에는 아래 예제와 같이 `region` 파라미터를 지정해야 합니다.

```
aws ssm list-resource-data-sync --region us-west-1
```

10. 동기화 구성이 성공적으로 생성된 후에는 Amazon S3의 대상 버킷을 검사합니다. 몇 분 내에 인벤토리 데이터가 표시되어야 합니다.

Amazon Athena의 데이터 작업

아래 섹션에서는 Amazon Athena에서 데이터를 확인하거나 쿼리를 실행하는 방법에 대해 설명합니다. 시작하기 전에 Athena에 대해 알아보는 것이 좋습니다. 자세한 내용은 Amazon Athena User Guide의 What is Amazon Athena? 및 Working with Data를 참조하세요.

Amazon Athena에서 데이터를 보고 쿼리하려면

1. https://console.aws.amazon.com/athena/에서 Athena 콘솔을 엽니다.
2. 다음 문을 복사하여 쿼리 편집기에 붙여 넣은 다음 쿼리 실행을 선택합니다.

   ```
   CREATE DATABASE ssminventory
   ```

   시스템이 ssminventory라는 이름의 데이터베이스를 생성합니다.
3. 다음 문을 복사하여 쿼리 편집기에 붙여 넣은 다음 쿼리 실행을 선택합니다. DOC-EXAMPLE-BUCKET 및 *bucket_prefix*를 Amazon S3 대상의 이름 및 접두사로 바꿉니다.

   ```
   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://DOC-EXAMPLE-BUCKET/bucket_prefix/AWS:Application/'
   ```

4. 다음 문을 복사하여 쿼리 편집기에 붙여 넣은 다음 쿼리 실행을 선택합니다.

```
MSCK REPAIR TABLE ssminventory.AWS_Application
```

시스템이 테이블을 분할합니다.

> **Note**
>
> 추가되는 AWS 리전 또는 AWS 계정에서 리소스 데이터 동기화를 생성하는 경우에는 위 명령을 다시 실행하여 파티션을 업데이트해야 합니다. Amazon S3 버킷 정책도 업데이트해야 할 수 있습니다.

5. 데이터를 미리 보려면 AWS_Application 테이블 옆에 있는 보기 아이콘을 선택합니다.

6. 다음 문을 복사하여 쿼리 편집기에 붙여 넣은 다음 쿼리 실행을 선택합니다.

```
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
```

쿼리는 Linux, macOS 및 Windows Server용 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에 있는 AWS 애플리케이션인 aws-cfn-bootstrap의 여러 버전 수를 반환합니다.

7. 다음 문을 개별적으로 복사하여 쿼리 편집기에 붙여 넣고 DOC-EXAMPLE-BUCKET 및 *bucket-prefix*를 Amazon S3에 대한 정보로 바꾼 다음, 쿼리 실행을 선택합니다. 아래 문들은 Athena의 인벤토리 테이블을 추가로 설정합니다.

```
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://DOC-EXAMPLE-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://DOC-EXAMPLE-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://DOC-EXAMPLE-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://DOC-EXAMPLE-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://DOC-EXAMPLE-BUCKET/bucket-prefix/AWS:PatchSummary/'
```

```
MSCK REPAIR TABLE ssminventory.AWS_PatchSummary
```

Amazon QuickSight의 데이터 작업

아래 섹션에서는 Amazon QuickSight의 시각화 빌드를 위해 참조할 수 있는 링크에 대해서 간략히 소개합니다.

Amazon QuickSight에서 시각화를 구축하려면

1. Amazon QuickSight에 가입한 후 QuickSight 콘솔에 로그인합니다.
2. `AWS_Application` 테이블과 생성한 다른 테이블에서 데이터 집합을 생성합니다. 자세한 내용은 Amazon Athena 데이터를 사용하여 데이터 집합 생성을 참조하세요.
3. 테이블을 조인합니다. 예를 들어 다른 인벤토리 테이블의 `resourceid` 열과 일치하기 때문에 `AWS_InstanceInformation`의 `instanceid` 열을 조인할 수 있습니다. 테이블 조인에 대한 자세한 내용은 Joining Tables 섹션을 참조하세요.
4. 시각화를 빌드합니다. 자세한 내용은 Working with Amazon QuickSight Visuals 섹션을 참조하세요.

Systems Manager Inventory 관련 문제 해결

여기서는 AWS Systems Manager 인벤토리와 관련된 일반적인 오류나 문제를 해결하는 방법을 설명합니다. Systems Manager에서 노드를 보는 데 문제가 있는 경우 관리형 노드 가용성 문제 해결 섹션을 참조하세요.

주제

- 문서 'AWS-GatherSoftwareInventory'와의 다중 적용 모든 연결은 지원되지 않습니다.
- Inventory 실행 상태가 계속 보류 중임
- AWS-ListWindowsInventory 문서 실행 실패
- 콘솔은 인벤토리 대시보드 | 세부 정보 보기 | 설정 탭을 표시하지 않습니다.
- UnsupportedAgent
- 건너뜀
- Failed
- Amazon EC2 인스턴스에 대한 인벤토리 규정 준수 실패

- S3 버킷 객체에 이전 데이터가 포함되어 있음

문서 '**AWS-GatherSoftwareInventory**'와의 다중 적용 모든 연결은 지원되지 않습니다.

`Multiple apply all associations with document 'AWS-GatherSoftwareInventory' are not supported` 오류는 모든 노드에 대한 Inventory 연결을 구성하려는 하나 이상의 AWS 리전이 모든 노드에 대한 인벤토리 연결로 이미 구성되었음을 의미합니다. 필요한 경우 모든 노드에 대한 기존 인벤토리 연결을 삭제한 다음 새로 생성할 수 있습니다. 기존 인벤토리 연결을 보려면 Systems Manager 콘솔에서 State Manager를 선택한 다음 `AWS-GatherSoftwareInventory` SSM 문서를 사용하는 연결을 찾습니다. 모든 노드에 대한 기존 인벤토리 연결이 여러 리전에 걸쳐 생성되었고 새로 생성하려는 경우 기존 연결이 있는 각 리전에서 기존 연결을 삭제해야 합니다.

Inventory 실행 상태가 계속 보류 중임

인벤토리 수집이 계속 `Pending` 상태인 이유는 두 가지입니다.

- 선택된 AWS 리전에 노드가 없습니다.

  Systems Manager Quick Setup을 사용하여 글로벌 인벤토리 연결을 생성하는 경우 선택한 리전에 사용 가능한 노드가 없으면 인벤토리 연결(`AWS-GatherSoftwareInventory` 문서)의 상태가 `Pending`으로 표시됩니다.
- 권한 부족

  하나 이상의 노드에 Systems Manager Inventory를 실행할 권한이 없는 경우 인벤토리 연결이 `Pending`으로 표시됩니다. AWS Identity and Access Management(IAM) 인스턴스 프로파일에 AmazonSSMManagedInstanceCore 관리형 정책이 포함되어 있는지 확인합니다. 인스턴스 프로파일에 이 정책을 추가하는 방법에 대한 자세한 내용은 EC2 인스턴스 권한에 대한 대체 구성 섹션을 참조하세요.

  최소한 인스턴스 프로파일에 다음 IAM 권한이 있어야 합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeAssociation",
```

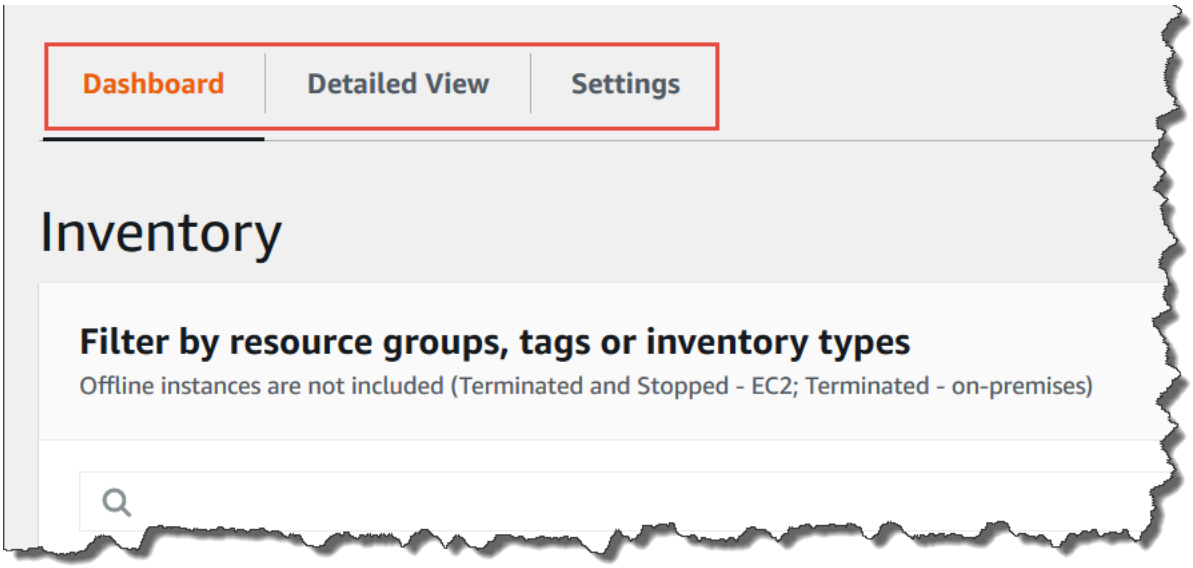
```
                "ssm:ListAssociations",
                "ssm:ListInstanceAssociations",
                "ssm:PutInventory",
                "ssm:PutComplianceItems",
                "ssm:UpdateAssociationStatus",
                "ssm:UpdateInstanceAssociationStatus",
                "ssm:UpdateInstanceInformation",
                "ssm:GetDocument",
                "ssm:DescribeDocument"
            ],
            "Resource": "*"
        }
    ]
}
```

`AWS-ListWindowsInventory` 문서 실행 실패

`AWS-ListWindowsInventory` 문서는 더 이상 사용되지 않습니다. 이 문서를 사용하여 인벤토리를 수집하지 마십시오. 대신 인벤토리 수집 구성에 설명된 프로세스 중 하나를 사용하십시오.

콘솔은 인벤토리 대시보드 | 세부 정보 보기 | 설정 탭을 표시하지 않습니다.

Inventory [세부 정보 뷰(Detailed View)] 페이지는 Amazon Athena를 제공하는 AWS 리전에서만 사용할 수 있습니다. 다음 탭이 인벤토리(Inventory) 페이지에 표시되지 않는 경우 Athena는 리전에서 사용할 수 없으며 세부 정보 보기(Detailed View)를 사용하여 데이터를 쿼리할 수 없습니다.



UnsupportedAgent

인벤토리 연결의 세부 상태가 UnsupportedAgent이고, 연결 상태(Association status)에 실패(Failed)라고 표시되는 경우 관리형 노드의 AWS Systems Manager SSM Agent 버전이 올바르지 않은 것입니다. 예를 들어 전역 인벤토리 연결을 생성하여 해당 AWS 계정의 모든 노드에 대한 인벤토리를 작성하려면 SSM Agent 버전 2.0.790.0 이상을 사용해야 합니다. 각 노드에서 실행 중인 에이전트 버전은 관리형 인스턴스 페이지의 에이전트 버전 열에서 볼 수 있습니다. 노드에서 SSM Agent를 업데이트하는 자세한 방법은 Run Command를 사용하여 SSM Agent 업데이트 섹션을 참조하세요.

건너뜀

노드에 대한 인벤토리 연결의 상태가 건너뜀(Skipped)이면 모든 노드에서 인벤토리를 수집하기 위해 전역 인벤토리 연결을 생성했지만 건너뛴 노드에 이미 할당된 인벤토리 연결이 있음을 의미합니다. 이 노드에는 전역 인벤토리 연결이 할당되지 않고 전역 인벤토리 연결에서 수집한 인벤토리가 없습니다. 하지만 이 노드는 기존 인벤토리 연결 실행 시에는 인벤토리 데이터를 보고합니다.

글로벌 인벤토리 연결에서 노드를 건너뛰지 않도록 하려면 기존 인벤토리 연결을 삭제해야 합니다. 기존 인벤토리 연결을 보려면 Systems Manager 콘솔에서 State Manager를 선택한 다음 `AWS-GatherSoftwareInventory` SSM 문서를 사용하는 연결을 찾습니다.

Failed

인스턴스의 인벤토리 연결 상태가 실패(Failed)인 경우, 해당 인스턴스에 여러 개의 인벤토리 연결이 할당되었을 수 있습니다. 하나의 노드는 한 번에 한 개의 인벤토리 연결만 가질 수 있습니다. 인벤토리 연결은 `AWS-GatherSoftwareInventory` AWS Systems Manager 문서(SSM 문서)를 사용합니다. AWS Command Line Interface(AWS CLI)에서 다음 명령을 실행하여 노드에 대한 연결 목록을 볼 수 있습니다.

```
aws ssm describe-instance-associations-status
                --instance-id instance ID
```

Amazon EC2 인스턴스에 대한 인벤토리 규정 준수 실패

인스턴스에 여러 인벤토리 연결을 할당하면 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에 대한 인벤토리 규정 준수가 실패할 수 있습니다.

이 문제를 해결하려면 인스턴스에 할당된 인벤토리 연결을 하나 이상 삭제합니다. 자세한 내용은 연결 삭제를 참조하세요.

> **Note**
>
> 관리형 노드에 대해 여러 인벤토리 연결을 만드는 경우 다음 동작을 유의하세요.
>
> - 각 노드에 모든 노드(--targets "Key=InstanceIds,Values=*")를 대상으로 하는 인벤토리 연결이 할당될 수 있습니다.
> - 태그 키-값 페어 또는 AWS 리소스 그룹을 사용하는 특정 연결이 각 노드에 할당될 수도 있습니다.
> - 노드에 여러 인벤토리 연결이 할당된 경우 실행되지 않은 연결에 대해 상태가 건너뜀(Skipped)으로 표시됩니다. 가장 최근에 실행된 연결은 인벤토리 연결의 실제 상태를 표시합니다.
> - 노드에 여러 인벤토리 연결이 할당되고 각각 태그 키-값 페어를 사용하는 경우 태그 충돌로 인해 해당 인벤토리 연결이 노드에서 실행되지 않습니다. 태그 키-값 충돌이 없는 노드에서 연결이 계속 실행됩니다.

S3 버킷 객체에 이전 데이터가 포함되어 있음

Amazon S3 버킷 객체 내부의 데이터는 인벤토리 연결이 성공하고 새 데이터가 발견되면 업데이트됩니다. Amazon S3 버킷 객체는 연결이 실행되고 실패할 때 노드마다 업데이트되지만, 이 경우 객체 내부의 데이터는 업데이트되지 않습니다. Amazon S3 버킷 객체 내부의 데이터는 연결이 성공적으로 실행될 때만 업데이트됩니다. 인벤토리 연결에 실패하면 Amazon S3 버킷 객체에 이전 데이터가 표시됩니다.

AWS Systems Manager 하이브리드 정품 인증

하이브리드 및 멀티클라우드 환경의 AWS Systems Manager에서 사용하도록 비 EC2 시스템을 구성하려면 하이브리드 정품 인증을 생성합니다. 관리형 노드로 지원되는 비 EC2 시스템 유형에는 다음이 포함됩니다.

- 자체 프레미스의 서버(온프레미스 서버)
- AWS IoT Greengrass 코어 디바이스
- AWS IoT 및 비 AWS 엣지 디바이스
- 다른 클라우드 환경의 VM을 포함한 가상 머신

create-activation 명령을 실행하여 하이브리드 정품 인증 프로세스를 시작하면 명령 응답으로 활성화 코드와 ID가 수신됩니다. 그런 다음에 하이브리드 및 멀티클라우드 환경에서 Systems Manager 사용하기의 3단계에 설명된 대로 시스템에 SSM Agent를 설치하는 명령에 정품 인증 코드 및 ID를 포함합니다. 이 정품 인증 프로세스는 AWS IoT Greengrass 코어 디바이스를 제외한 모든 비 EC2 시스템 유형에 적용됩니다. AWS IoT Greengrass Systems Manager용 핵심 장치 구성에 대한 자세한 내용은 관리자용 핵심 장치는 Systems Manager를 통한 엣지 디바이스 관리 섹션을 참조하세요.

> **Note**
>
> 현재 비 EC2 macOS 시스템에 대한 지원은 제공되지 않습니다.

Systems Manager 인스턴스 티어 정보

AWS Systems Manager에서는 표준 인스턴스 티어와 고급 인스턴스 티어를 제공합니다. 둘 다 하이브리드 및 멀티클라우드 환경에서 관리형 노드를 지원합니다. 표준 인스턴스 티어를 통해 AWS 리전의 AWS 계정당 최대 1,000개의 시스템을 등록할 수 있습니다. 단일 계정 및 리전에 1,000개 이상의 시스템을 등록해야 하는 경우 고급 인스턴스 티어를 사용합니다. 고급 인스턴스 티어에서 원하는 만큼 관리형 노드를 생성할 수 있습니다. Systems Manager를 위해 구성된 모든 관리형 노드의 가격은 사용량에 따라 지불하는 방식으로 책정됩니다. 고급 인스턴스 티어 활성화에 대한 자세한 내용은 고급 인스턴스 티어 설정 섹션을 참조하세요. 요금에 대한 자세한 내용은 AWS Systems Manager 요금을 참조하세요.

> **Note**
>
> - 고급 인스턴스도 AWS Systems Manager Session Manager를 사용하여 하이브리드 및 멀티클라우드 환경의 비 EC2 노드에 연결할 수 있습니다. Session Manager에서는 인스턴스에 대한 대화형 쉘 액세스가 제공됩니다. 자세한 내용은 AWS Systems Manager Session Manager 단원을 참조하십시오.
> - 표준 인스턴스 할당량은 또한 Systems Manager 온프레미스 정품 인증을 사용하는 EC2 인스턴스에도 적용됩니다(일반 시나리오는 아님).
> - Microsoft에서 릴리스한 가상 머신 온프레미스 인스턴스 기반 애플리케이션을 패치하려면 고급 인스턴스 티어를 활성화합니다. 고급 인스턴스 티어 사용에는 요금이 따릅니다. Microsoft에서 릴리스한 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 기반 애플리케이션을 패치하는 데는 추가 요금이 없습니다. 자세한 내용은 Microsoft에서 릴리스한 Windows Server 기반 애플리케이션 패치 정보 단원을 참조하십시오.

AWS Systems Manager Session Manager

Session Manager는 종합 관리형 AWS Systems Manager 기능입니다. Session Manager를 사용하여 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스, 엣지 디바이스, 온프레미스 서버 및 가상 머신을 관리할 수 있습니다. 대화형 원클릭 브라우저 기반 셸 또는 AWS Command Line Interface(AWS CLI)을 사용할 수 있습니다. Session Manager는 인바운드 포트를 열고, Bastion Host를 유지하고, SSH 키를 관리할 필요 없이 보안성과 감사 가능성을 갖춘 노드 관리 기능을 제공합니다. 또한 Session Manager를 통해 노드에 대한 제어된 액세스를 요구하는 회사 정책, 엄격한 보안 관행을 손쉽게 준수하고, 관리형 노드 액세스 세부 정보가 포함된 완전히 감사 가능한 로그를 제공하는 동시에 최종 사용자에게 관리형 노드에 대한 간단한 원 클릭 교차 플랫폼 액세스를 제공합니다. Session Manager를 시작하려면 Systems Manager 콘솔을 엽니다. 탐색 창에서 Session Manager를 선택합니다.

Session Manager가 조직에 주는 이점은 무엇인가요?

Session Manager에서 제공하는 이점은 다음과 같습니다.

- IAM 정책을 사용하여 관리형 노드에 대한 중앙 집중식 액세스 제어

  관리자가 한 곳에서 관리형 노드에 대한 액세스 권한을 부여하고 취소할 수 있습니다. AWS Identity and Access Management(IAM) 정책만 사용하는 경우에는 Session Manager를 사용할 수 있는 조직 내 개별 사용자 또는 그룹과 이들이 액세스할 수 있는 관리형 노드를 제어할 수 있습니다.

- 인바운드 포트를 열 필요가 없고 Bastion Host 또는 SSH 키를 관리할 필요가 없음

  관리형 노드에 인바운드 PowerShell 포트와 원격 PowerShell 포트를 열어 두면 관리형 노드에서 개체가 권한이 없거나 악의적인 명령을 실행할 위험이 매우 높아집니다. Session Manager에서는 이러한 인바운드 포트를 닫도록 하고, SSH 키 및 인증서, Bastion Host 및 점프박스를 관리할 필요가 없도록 해 보안 태세를 강화하도록 합니다.

- 클릭 한 번으로 콘솔 및 CLI에서 관리형 노드에 액세스

  AWS Systems Manager 콘솔 또는 Amazon EC2 콘솔을 사용하여 한 번의 클릭으로 세션을 시작할 수 있습니다. AWS CLI를 사용하여 단일 명령 또는 일련의 명령을 실행하는 세션을 시작할 수도 있습니다. 관리형 노드에 대한 권한이 SSH 키 또는 기타 메커니즘을 대신 IAM 정책을 통해 제공되기 때문에 연결 시간이 크게 줄어듭니다.

- 하이브리드 및 멀티클라우드 환경의 Amazon EC2 인스턴스와 비 EC2 관리형 노드에 모두 연결

  하이브리드 및 멀티클라우드 환경의 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스와 비 EC2 관리형 노드에 모두 연결할 수 있습니다.

Session Manager를 사용하여 비EC2 노드에 연결하려면 먼저 고급 인스턴스 티어를 활성화해야 합니다. 고급 인스턴스 티어를 사용하는 데는 요금이 부과됩니다. 그러나 Session Manager를 사용하여 EC2 인스턴스에 연결하는 데는 추가 요금이 부과되지 않습니다. 자세한 내용은 인스턴스 티어 구성을 참조하세요.

- 포트 전달

  원격 관리형 노드 내부의 포트를 클라이언트의 로컬 포트로 리디렉션합니다. 그런 다음 로컬 포트에 연결하고 노드 내부에서 실행 중인 서버 애플리케이션에 액세스합니다.

- Windows, Linux 및 macOS에 대한 크로스 플랫폼 지원

  Session Manager에서는 단일 도구의 Windows, Linux 및 macOS에 대한 지원을 제공합니다. 예를 들면, Windows Server 관리형 노드의 Linux 및 macOS 관리형 노드 또는 RDP 연결에 SSH 클라이언트를 사용할 필요가 없습니다.

- 세션 활동 로깅 및 감사

  조직의 작업 또는 보안 요구 사항을 충족하기 위해 관리형 노드에 대한 연결과 해당 인스턴스에서 실행된 명령에 대한 기록을 제공해야 할 수 있습니다. 또한 조직의 사용자가 세션 활동을 시작 또는 종료할 때 알림을 받을 수도 있습니다.

  로깅 및 감사 기능은 다음 AWS 서비스와(과)의 통합을 통해 제공됩니다.

  - AWS CloudTrail - AWS CloudTrail에서는 AWS 계정에서 수행된 Session Manager API 호출에 대한 정보를 캡처해 지정한 Amazon Simple Storage Service(Amazon S3) 버킷에 저장되는 로그 파일에 기록합니다. 계정에 대한 모든 CloudTrail 로그를 저장하는 데 버킷 하나가 사용됩니다. 자세한 내용은 AWS CloudTrail을 사용하여 AWS Systems ManagerAPI 호출을 로깅 단원을 참조하십시오.
  - Amazon Simple Storage Service - 디버깅 및 문제 해결을 위해 선택한 Amazon S3 버킷에 세션 로그 데이터를 저장하도록 선택할 수 있습니다. 로그 데이터는 AWS KMS key를 사용하여 암호화를 적용하거나 적용하지 않은 상태로 Amazon S3 버킷으로 전송할 수 있습니다. 자세한 내용은 Amazon S3를 사용하여 세션 데이터 로깅(콘솔) 단원을 참조하십시오.
  - Amazon CloudWatch Logs - CloudWatch Logs를 사용하면 다양한 AWS 서비스의 로그 파일을 모니터링, 저장 및 액세스할 수 있습니다. 디버깅 및 문제 해결을 위해 CloudWatch Logs 로그 그룹으로 세션 로그 데이터를 전송할 수 있습니다. 로그 데이터는 KMS 키를 사용하여 AWS KMS 암호화를 적용하거나 적용하지 않은 상태로 로그 그룹으로 스트리밍할 수 있습니다. 자세한 내용은 Amazon CloudWatch Logs를 사용하여 세션 데이터 로깅(콘솔) 단원을 참조하십시오.
  - Amazon EventBridge 및 Amazon Simple Notification Service - EventBridge를 사용하면 지정한 AWS 리소스에 변경 사항이 발생할 때 이를 감지하는 규칙을 설정할 수 있습니다. 조직 내 사용자

가 세션을 시작 또는 중지하면 이를 감지하는 규칙을 생성하면 Amazon SNS를 통해 해당 이벤트에 대한 알림(예: 문자 메시지 또는 이메일 메시지)을 받을 수 있습니다. 또한 다른 응답을 시작하도록 CloudWatch 이벤트를 구성할 수도 있습니다. 자세한 내용은 Amazon EventBridge를 사용하여 세션 활동 모니터링(콘솔) 단원을 참조하십시오.

> **Note**
>
> 포트 전달 또는 SSH를 통해 연결하는 Session Manager 세션에는 로깅을 사용할 수 없습니다. SSH는 모든 세션 데이터를 암호화하고 Session Manager는 SSH 연결을 위한 터널 역할만 하기 때문입니다.

Session Manager는 누가 사용해야 하나요?

- 보안 및 감사 상태를 강화하고, 관리형 노드의 액세스 제어를 중앙에서 관리하여 운영 오버헤드를 절감하고, 인바운드 노드를 줄이고자 하는 모든 AWS 고객
- 관리형 노드 액세스와 활동을 모니터링하고 추적하며, 관리형 노드에서 인바운드 포트를 닫거나 퍼블릭 IP 없이 관리형 노드와 연결하기를 원하는 정보 보안 전문가
- 단일 위치의 액세스 권한을 부여하고 취소하며 Linux, macOS 및 Windows Server 관리형 노드의 솔루션 하나를 사용자에게 제공하려는 관리자입니다.
- SSH 키를 제공할 필요 없이 브라우저 또는 AWS CLI에서 한 번의 클릭으로 관리형 노드에 연결하고자 하는 사용자

Session Manager의 주요 기능은 무엇입니까?

- Windows Server, Linux 및 macOS 관리형 노드 지원

  Session Manager를 사용하면 Amazon Elastic Compute Cloud(EC2) 인스턴스, 엣지 디바이스, 온프레미스 서버 및 가상 머신과의 보안 연결을 설정할 수 있습니다. 지원되는 운영 체제 유형의 목록은 Session Manager 설정 섹션을 참조하세요.

  > **Note**
  >
  > Session Manager온프레미스 머신에 대한 지원은 고급 인스턴스 티어에만 제공됩니다. 자세한 설명은 고급 인스턴스 티어 설정을 참조하세요.

- 콘솔 CLI 및 SDK를 통해 Session Manager 기능에 액세스

다음과 같은 방식으로 Session Manager 작업을 수행할 수 있습니다.

AWS Systems Manager 콘솔에서는 관리자 및 최종 사용자 둘 다를 위한 모든 Session Manager 기능에 액세스할 수 있습니다. Systems Manager 콘솔을 사용하여 세션에 연결된 모든 태스크를 수행할 수 있습니다.

Amazon EC2 콘솔에서는 최종 사용자가 세션 권한이 부여된 EC2 인스턴스에 연결할 수 있는 기능을 제공합니다.

AWS CLI에서는 최종 사용자를 위한 Session Manager 기능에 액세스할 수 있습니다. AWS CLI를 사용해 세션을 시작하고, 세션 목록을 보고, 세션을 영구히 종료할 수 있습니다.

> **Note**
>
> AWS CLI를 사용하여 세션 명령을 실행하려면 CLI의 버전 1.16.12(또는 이상)를 사용하고 로컬 시스템에 Session Manager 플러그인을 설치해야 합니다. 자세한 설명은 AWS CLI의 Session Manager 플러그인 설치을 참조하세요. GitHub의 플러그인을 보려면 session-manager-plugin을 참조하세요.

- IAM 액세스 제어

  IAM 정책 사용을 통해 관리형 노드에 대한 세션을 시작할 조직 내 멤버와 해당 멤버가 액세스할 수 있는 노드를 제어할 수 있습니다. 또한 관리형 노드에 대한 임시 액세스를 제공할 수도 있습니다. 예를 들어, 대기 중인 엔지니어(또는 대기 중인 엔지니어 그룹)에게 교대 근무 중에만 프로덕션 서버에 액세스할 수 있는 권한을 부여하려고 할 수 있습니다.

- 로깅 및 감사 기능 지원

  Session Manager에서는 여러 가지 다른 AWS 서비스와(과)의 통합을 통해 AWS 계정 계정 내 세션 기록을 감사 및 로깅할 수 있는 옵션을 제공합니다. 자세한 내용은 세션 활동 감사 및 세션 활동 로깅 활성화 및 비활성화 단원을 참조하세요.

- 구성 가능한 셸 프로파일

  Session Manager는 세션 내에서 기본 설정을 구성할 수 있는 옵션을 제공합니다. 이러한 사용자 정의 가능한 프로파일을 사용하면 셸 기본 설정, 환경 변수, 작업 디렉터리 및 세션 시작 시 여러 명령 실행과 같은 기본 설정을 정의할 수 있습니다.

- 고객 키 데이터 암호화 지원

Amazon Simple Storage Service(Amazon S3) 버킷으로 전송하거나 CloudWatch Logs 로그 그룹으로 스트리밍하는 세션 데이터 로그를 암호화하도록 Session Manager를 구성할 수 있습니다. 세션 중에 클라이언트 시스템과 관리형 노드 사이에 전송되는 데이터를 추가로 암호화하도록 Session Manager을 구성할 수도 있습니다. 자세한 내용은 세션 활동 로깅 활성화 및 비활성화 및 세션 기본 설정 구성을 참조하세요.

- 퍼블릭 IP 주소 없이 관리형 노드에 대한 AWS PrivateLink 지원

  AWS PrivateLink로 Systems Manager용 VPC 엔드포인트를 설정하여 세션을 더욱 안전하게 보호할 수도 있습니다. AWS PrivateLink는 관리형 노드, Systems Manager 및 Amazon EC2 간의 모든 네트워크 트래픽을 Amazon 네트워크로 제한합니다. 자세한 내용은 Systems Manager용 VPC 엔드포인트를 사용하여 EC2 인스턴스의 보안 개선을 참조하세요.

- 터널링

  세션에서 세션 유형 AWS Systems Manager(SSM) 문서를 사용하여 트래픽을 터널링합니다(예: 클라이언트 시스템의 로컬 포트와 관리형 노드의 원격 포트 간 http 또는 사용자 정의 프로토콜).

- 대화형 명령

  세션을 사용하여 단일 명령을 대화형으로 실행하고 사용자가 관리형 노드에서 할 수 있는 것을 관리할 방법을 제공하는 세션 유형 SSM 문서를 생성합니다.

세션이란 무엇입니까?

세션은 Session Manager를 사용한 관리형 노드 연결입니다. 세션은 클라이언트(사용자) 와 명령에 대한 입력 및 출력을 스트리밍하는 원격 관리형 노드 간의 안전한 양방향 통신 채널을 기반으로 합니다. 클라이언트와 관리형 노드 간의 트래픽은 TLS 1.2를 사용하여 암호화되며, 연결을 생성하기 위한 요청은 SigV4를 사용하여 서명됩니다. 이 양방향 통신을 통해 관리형 노드에 대한 대화형 bash 및 PowerShell 액세스가 가능합니다. AWS Key Management Service(AWS KMS) 키를 사용하여 기본 TLS 암호화 이상으로 데이터를 추가로 암호화할 수도 있습니다.

예를 들어, John이 IT 부서의 대기 중인 엔지니어라고 생각해 보겠습니다. John은 원격으로 관리형 노드에 액세스해야 하는 문제에 대한 알림을 받습니다(예: 문제 해결 또는 노드에 대한 간단한 구성 옵션을 변경하는 지침이 필요한 장애). John은 AWS Systems Manager 콘솔, Amazon EC2 콘솔 또는 AWS CLI를 사용해 관리형 노드에 연결하는 세션을 시작하고, 노드에서 태스크를 완료하는 데 필요한 명령을 실행한 다음 세션을 종료합니다.

John이 세션을 시작하라는 첫 번째 명령을 보내면 Session Manager서비스에서 John의 ID를 인증하고, IAM 정책에 따라 John에게 부여된 권한을 확인한 다음 구성 설정(예: 세션에 대해 허용된 제한 확

인)을 확인하고 SSM Agent에 양방향 연결을 열라는 메시지를 보냅니다. 연결이 설정되어 John이 다음 명령을 입력하면 SSM Agent의 명령 출력이 이 통신 채널로 업로드되고 다시 John의 로컬 시스템으로 전송됩니다.

주제

Session Manager 설정

AWS Systems Manager를 사용하여 계정 내 Session Manager 관리형 노드에 연결하기 전에 다음 주제의 단계를 완료합니다.

주제

1단계: Session Manager 사전 조건 완료

Session Manager 사용을 시작하기 전에 환경이 다음 요구 사항을 충족하는지 확인하십시오.

Session Manager 필수 조건

요구 사항	설명
지원되는 운영 체제	Session Manager에서는 고급 인스턴스 티어를 사용하는 하이브리드 및 멀티클라우드 환경의 비 EC2 시스템 외에 Amazon Elastic Compute Cloud(Amazon EC2) 연결도 지원합니다. Session Manager에서 지원하는 운영 체제 버전은 다음과 같습니다. **Note** Session Manager에서는 고급 인스턴스 티어를 사용하는 하이브리드 및 멀티클라우드 환경의 EC2 인스턴스, 엣지 디바이스, 온프레미스 서버 및 가상 머신을 지원합니다. 전용 인스턴스에 대한 자세한 내용은 인스턴스 티어 구성 섹션을 참조하세요. Linux 및 macOS Session Manager에서는 AWS Systems Manager에서 지원되는 모든 Linux 및 macOS버전을 지원합니다. 자세한 설명은 지원되는 운영 체제 및 시스템 유형을 참조하세요. Windows Session Manager는 Windows Server 2012부터 Windows Server 2022까지 지원합니다.

요구 사항	설명
	Note Microsoft Windows Server 2016 Nano는 지원되지 않습니다.

요구 사항	설명
SSM Agent	세션을 통해 연결하려는 관리형 노드에 AWS Systems Manager SSM Agent 버전 2.3.68.0 이상이 설치되어 있어야 합니다. AWS Key Management Service(AWS KMS)에서 생성한 키를 사용하여 세션 데이터를 암호화하는 옵션을 사용하려면 버전 2.3.539.0 이상의 SSM Agent가 관리형 노드에 설치되어 있어야 합니다. 세션에서 셸 프로파일을 사용하려면 관리형 노드에 SSM Agent 버전 3.0.161.0 이상이 설치되어 있어야 합니다. Session Manager 포트 전달 또는 SSH 세션을 시작하려면 관리형 노드에 SSM Agent 버전 3.0.222.0 이상이 설치되어 있어야 합니다. Amazon CloudWatch Logs를 사용하여 세션 데이터를 스트리밍하려면 관리형 노드에 SSM Agent 버전 3.0.284.0 이상이 설치되어 있어야 합니다. 인스턴스에서 실행 중인 버전 번호를 확인하는 방법에 대한 자세한 내용은 SSM Agent 버전 번호 확인 섹션을 참조하세요. SSM Agent 수동 설치 또는 자동 업데이트에 대한 자세한 내용은 SSM Agent 작업 섹션을 참조하세요. ssm-user 계정 정보 SSM Agent의 버전 2.3.50.0부터 에이전트는 `ssm-user`라는 루트 또는 관리자 권한으로 관리형 노드에 사용자 계정을 생성합니다. (2.3.612.0 이전 버전에서는 SSM Agent가 시작되거나 다시 시작될 때 계정이 생성됩니다. 버전 2.3.612.0 이상에서는 관리형 노드에서 세션이 처음 시작

요구 사항	설명
	될 때 `ssm-user`가 생성됩니다.) 세션은 이 사용자 계정의 관리자 자격 증명을 사용하여 시작됩니다. 이 계정에 대한 관리 제어 제한에 대한 자세한 내용은 ssm-user 계정 관리 권한 설정 또는 해제를 참조하세요. Windows Server 도메인 컨트롤러의 ssm-user SSM Agent 버전 2.3.612.0부터 `ssm-user` 계정은 Windows Server 도메인 컨트롤러로 사용되는 관리형 노드에 자동으로 생성되지 않습니다. Windows Server 시스템에서 Session Manager를 도메인 컨트롤러로 사용하려면 `ssm-user` 계정이 없는 경우 수동으로 생성하고 사용자에게 도메인 관리자 권한을 할당해야 합니다. Windows Server에서 SSM Agent는 세션이 시작될 때마다 `ssm-user` 계정에 대한 새 암호를 설정하므로 계정을 만들 때 암호를 지정할 필요가 없습니다.

요구 사항	설명
엔드포인트에 연결	연결하려는 관리형 노드는 다음 엔드포인트에 대한 HTTPS(포트 443) 아웃바운드 트래픽도 허용해야 합니다. • ec2messages.*region*.amazonaws.com • ssm.*region*.amazonaws.com • ssmmessages.*region*.amazonaws.com 자세한 정보는 다음 주제를 참조하세요. • 참조: ec2messages, ssmmessages 및 기타 API 작업 • Systems Manager를 사용하여 인터넷 액세스 없이 프라이빗 EC2 인스턴스를 관리할 수 있도록 VPC 엔드포인트를 생성하려면 어떻게 해야 합니까?(AWS re:Post 지식 센터) 또는 인터페이스 엔드포인트를 사용하여 필요한 엔드포인트에 연결할 수 있습니다. 자세한 내용은 6단계: (옵션) AWS PrivateLink를 사용하여 Session Manager에 대한 VPC 엔드포인트 설정 단원을 참조하십시오.

요구 사항	설명
AWS CLI	(옵션) AWS Systems Manager 콘솔 또는 Amazon EC2 콘솔 대신 AWS Command Line Interface(AWS CLI)를 사용하여 세션을 시작한 경우 로컬 시스템에 CLI 버전 1.16.12 이상이 설치되어 있어야 합니다. `aws --version` 을 호출해 버전을 확인할 수 있습니다. CLI를 설치하거나 업그레이드해야 하는 경우 AWS Command Line Interface 사용 설명서의 AWS Command Line Interface 설치를 참조하세요. **Important** SSM Agent의 업데이트된 버전은 Systems Manager에 새 기능이 추가되거나 기존 기능에 업데이트가 발생할 때마다 릴리스됩니다. 최신 버전의 에이전트를 사용하지 못하면 관리형 노드에서 다양한 Systems Manager 기능을 사용하지 못할 수 있습니다. 이러한 이유로 시스템의 SSM Agent 상태를 최신 상태로 유지하는 프로세스를 자동화하는 것이 좋습니다. 자세한 설명은 SSM Agent 업데이트 자동화을 참조하세요. SSM Agent 업데이트에 대해 알림을 수신하려면 GitHub에서 SSM Agent 릴리스 정보 페이지를 구독합니다. 또한 CLI를 사용해 Session Manager로 노드를 관리하려면 먼저, 로컬 시스템에 Session Manager 플러그 인을 설치해야 합니다. 자세한

요구 사항	설명
	설명은 AWS CLI의 Session Manager 플러그인 설치을 참조하세요.
advanced-instances 티어 활성화(하이브리드 및 멀티클라우드 환경)	Session Manager를 사용하여 비 EC2 시스템에 연결하려면 하이브리드 정품 인증을 생성하여 비 EC2 시스템을 관리형 인스턴스로 등록하는 AWS 계정 및 AWS 리전에서 advanced-instances 티어를 활성화해야 합니다. 고급 인스턴스 티어를 사용하는 데는 요금이 부과됩니다. 고급 인스턴스 티어에 대한 자세한 내용은 인스턴스 티어 구성 섹션을 참조하세요.
IAM 서비스 역할 권한 확인(하이브리드 및 멀티클라우드 환경)	하이브리드 정품 인증 노드에서는 하이브리드 정품 인증에 지정된 AWS Identity and Access Management(IAM) 서비스 역할을 사용하여 Systems Manager API 작업과 통신합니다. 이 서비스 역할에는 Session Manager를 사용하여 하이브리드 및 멀티클라우드 시스템에 연결하는 데 필요한 권한이 포함되어야 합니다. 서비스 역할이 AWS 관리형 정책 `AmazonSSMManagedInstanceCore` 를 포함한다면, Session Manager에 대한 필수 권한이 이미 제공되어 있습니다. 서비스 역할에 필수 권한이 포함되어 있지 않은 경우 관리형 인스턴스를 등록 취소하고 필수 권한이 있는 IAM 서비스 역할을 사용하는 새 하이브리드 활성화에 등록해야 합니다. 관리형 인스턴스 등록 취소에 대한 자세한 내용은 하이브리드 및 멀티클라우드 환경의 관리형 노드 등록 취소 섹션을 참조하세요. Session Manager 권한이 있는 IAM 정책 생성에 대한 자세한 내용은 2단계: Session Manager의 인스턴스 권한 확인 또는 추가를 참조하세요.

2단계: Session Manager의 인스턴스 권한 확인 또는 추가

AWS Systems Manager는 기본적으로 인스턴스에서 작업을 수행할 권한이 없습니다. AWS Identity and Access Management(IAM) 역할을 사용하여 계정 수준에서 또는 인스턴스 프로파일을 사용하여 인스턴스 수준에서 인스턴스 권한을 제공할 수 있습니다. 사용 사례에서 허용하는 경우 기본 호스트 관리 구성을 사용하여 계정 수준에서 액세스 권한을 부여하는 것이 좋습니다. `AmazonSSMManagedEC2InstanceDefaultPolicy` 정책을 사용하여 계정의 기본 호스트 관리 구성을 이미 설정했으면 다음 단계를 진행할 수 있습니다. 기본 호스트 관리 구성에 대한 자세한 내용은 기본 호스트 관리 구성 설정 관리 섹션을 참조하세요.

또는 인스턴스 프로파일을 사용하여 인스턴스에 필요한 권한을 제공할 수 있습니다. 인스턴스 프로파일을 사용하여 Amazon EC2 인스턴스에 IAM 역할을 전달합니다. IAM 인스턴스 프로파일은 Amazon EC2 인스턴스를 시작할 때 해당 인스턴스에 연결하거나 이전에 시작한 인스턴스에 연결할 수 있습니다. 자세한 내용은 인스턴스 프로파일 사용 섹션을 참조하세요.

온프레미스 서버 또는 가상 머신(VM)의 경우, Systems Manager에 온프레미스 서버 및 VM을 등록하는 데 사용되는 하이브리드 활성화와 연결된 IAM 서비스 역할에 의해 권한이 제공됩니다. 온프레미스 서버 및 VM에서는 인스턴스 프로파일을 사용하지 않습니다.

다른 Systems Manager 기능(예: Run Command 또는 Parameter Store)을 이미 사용하고 있는 경우 Session Manager에 대한 필수 기본 권한을 가진 인스턴스 프로파일이 Amazon EC2 인스턴스에 이미 연결되어 있을 수 있습니다. AWS 관리형 정책 `AmazonSSMManagedInstanceCore`를 포함한 인스턴스 프로파일이 이미 인스턴스에 연결되어 있는 경우에는 Session Manager에 대한 필수 권한이 이미 제공되어 있습니다. 하이브리드 활성화에 사용된 IAM 서비스 역할이 `AmazonSSMManagedInstanceCore` 관리형 정책을 포함하는 경우에도 마찬가지입니다.

> **Important**
>
> 하이브리드 활성화와 연결된 IAM 서비스 역할은 변경할 수 없습니다. 서비스 역할에 필수 권한이 포함되어 있지 않은 경우 관리형 인스턴스를 등록 취소하고 필수 권한이 있는 서비스 역할을 사용하는 새 하이브리드 활성화에 등록해야 합니다. 관리형 인스턴스 등록 취소에 대한 자세한 내용은 하이브리드 및 멀티클라우드 환경의 관리형 노드 등록 취소 섹션을 참조하세요. 온프레미스 시스템용 IAM 서비스 역할을 생성하는 방법에 대한 자세한 내용은 하이브리드 및 멀티클라우드 환경에서 Systems Manager에 필요한 IAM 서비스 역할 생성을 참조하세요.

그러나 경우에 따라 인스턴스 프로파일에 연결된 권한을 수정해야 할 수 있습니다. 예를 들어 더 좁은 인스턴스 권한 집합을 제공하려 하고 인스턴스 프로파일에 대해 사용자 정의 정책을 생성했거나 또는

세션 데이터를 확보하기 위한 Amazon Simple Storage Service(Amazon S3) 암호화 또는 AWS Key Management Service(AWS KMS) 암호화 옵션을 사용하려고 합니다. 이러한 경우 인스턴스에 Session Manager 작업이 수행되도록 다음 중 하나를 실시할 수 있습니다.

- 사용자 지정 IAM 역할의 Session Manager 작업용 임베드된 권한

  AWS 제공 기본 정책 `AmazonSSMManagedInstanceCore`에 의존하지 않는 기존 IAM 역할에 Session Manager 작업에 대한 권한을 추가하려면 기존 IAM 역할에 Session Manager 권한 추가의 단계를 따르세요.

- Session Manager 권한만 있는 사용자 지정 IAM 역할 생성

  Session Manager 작업에 대한 권한만 있는 IAM 역할을 생성하려면 Session Manager용 사용자 지정 IAM 역할 생성의 단계를 수행합니다.

- 모든 Systems Manager 작업에 대한 권한이 있는 새 IAM 역할 생성 및 사용

  모든 Systems Manager 권한을 부여하기 위해 AWS에서 제공하는 기본 정책을 사용하는 Systems Manager 관리형 인스턴스에 대한 IAM 역할을 생성하려면 Systems Manager에 필요한 인스턴스 권한 구성의 단계를 수행합니다.

주제

기존 IAM 역할에 Session Manager 권한 추가

다음 절차를 사용하여 기존 AWS Identity and Access Management(IAM) 역할에 Session Manager 권한을 추가합니다. 기존 역할에 권한을 추가하면 인스턴스 권한에 대한 AWS `AmazonSSMManagedInstanceCore` 정책을 사용하지 않고 컴퓨팅 환경의 보안을 강화할 수 있습니다.

> **Note**
>
> 다음과 같은 정보를 참고합니다.
>
> - 이 절차에서는 기존 역할에 액세스를 허용하려는 작업에 대한 다른 Systems Manager `ssm` 권한이 이미 포함되어 있다고 가정합니다. 이 정책만으로는 Session Manager를 사용하는 데 충분하지 않습니다.

- 다음 정책 예제에는 `s3:GetEncryptionConfiguration` 작업이 포함되어 있습니다. Session Manager 로깅 기본 설정에서 S3 로그 암호화 적용 옵션을 선택한 경우 이 작업이 필요합니다.

Session Manager 권한을 기존 역할에 추가(콘솔)

1. AWS Management Console에 로그인하여 https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할을 선택합니다.
3. 권한을 추가하려는 역할의 이름을 선택합니다.
4. 권한 탭을 선택합니다.
5. 권한 추가를 선택하고 인라인 정책 추가를 선택합니다.
6. JSON 탭을 선택합니다.
7. 기본 정책 콘텐츠를 다음과 같은 콘텐츠로 바꿉니다. *key-name*을 사용하려는 AWS Key Management Service 키(AWS KMS key)의 Amazon 리소스 이름(ARN)으로 바꿉니다.

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

KMS 키를 사용하여 세션 데이터를 암호화하는 방법에 대한 자세한 내용은 세션 데이터의 KMS 키 암호화를 설정하려면(콘솔) 섹션을 참조하세요.

세션 데이터에 AWS KMS 암호화를 사용하지 않으려면 정책에서 다음 콘텐츠를 제거할 수 있습니다.

```
,
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "key-name"
        }
```

8. 다음: 태그를 선택합니다.
9. (선택 사항) 태그 추가(Add tag)를 선택하고 정책에 대한 기본 설정 태그를 입력하여 태그를 추가합니다.
10. 다음: 검토를 선택합니다.
11. 정책 검토(Review policy) 페이지에서 이름(Name)에 인라인 정책 이름을 입력합니다(예: **`SessionManagerPermissions`**)
12. (선택 사항) 설명에 정책에 대한 설명을 입력합니다.

    정책 생성을 선택합니다.

`ssmmessages` 작업에 대한 자세한 내용은 참조: ec2messages, ssmmessages 및 기타 API 작업 섹션을 참조하세요.

Session Manager용 사용자 지정 IAM 역할 생성

Amazon EC2 관리형 인스턴스에서 작업을 수행하는 권한을 Session Manager에 부여하는 AWS Identity and Access Management(IAM) 역할을 생성할 수 있습니다. 세션 로그를 Amazon Simple

Storage Service(S3) 및 Amazon CloudWatch Logs로 보내는 데 필요한 권한을 부여하는 정책도 포함할 수도 있습니다.

IAM 역할을 생성한 후에 인스턴스에 역할을 연결하는 방법에 대한 내용은 AWS re:Post 웹사이트에서 인스턴스 프로파일 연결 또는 바꾸기를 참조하세요. IAM 인스턴스 프로파일 및 역할에 대한 자세한 내용은 IAM 사용 설명서의 인스턴스 프로파일 사용과 Linux 인스턴스용 Amazon Elastic Compute Cloud 사용 설명서의 Amazon EC2에 대한 IAM 역할을 참조하세요. 온프레미스 시스템용 IAM 서비스 역할을 생성하는 방법에 대한 자세한 내용은 하이브리드 및 멀티클라우드 환경에서 Systems Manager에 필요한 IAM 서비스 역할 생성을 참조하세요.

주제

- 최소 Session Manager 권한을 사용하여 IAM 역할 생성(콘솔)
- Session Manager, Amazon S3 및 CloudWatch Logs에 대한 권한이 있는 IAM 역할 생성(콘솔)

최소 Session Manager 권한을 사용하여 IAM 역할 생성(콘솔)

다음 절차에 따라 인스턴스에 대해 Session Manager 작업만 가능한 권한을 제공하는 정책을 사용해 사용자 지정 IAM 역할을 생성합니다.

최소 Session Manager 권한을 사용하여 인스턴스 프로파일을 생성하려면(콘솔)

1. AWS Management Console에 로그인하여 https://console.aws.amazon.com/iam/ 에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 정책을 선택한 후 정책 생성을 선택합니다. ([시작하기(Get Started)] 버튼이 표시되면 이 버튼을 선택한 후 [정책 생성(Create Policy)]을 선택합니다.)
3. JSON 탭을 선택합니다.
4. 기본 콘텐츠를 다음과 같은 정책으로 바꿉니다. AWS Key Management Service(AWS KMS)를 사용하여 세션을 암호화하려면 사용하려는 AWS KMS key의 Amazon 리소스 이름(ARN)으로 *key-name*을 바꿉니다.

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

KMS 키를 사용하여 세션 데이터를 암호화하는 방법에 대한 자세한 내용은 세션 데이터의 KMS 키 암호화를 설정하려면(콘솔) 섹션을 참조하세요.

세션 데이터에 AWS KMS 암호화를 사용하지 않으려면 정책에서 다음 콘텐츠를 제거할 수 있습니다.

```
,
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "key-name"
        }
```

5. 다음: 태그를 선택합니다.
6. (선택 사항) 태그 추가(Add tag)를 선택하고 정책에 대한 기본 설정 태그를 입력하여 태그를 추가합니다.
7. 다음: 검토를 선택합니다.
8. 정책 검토(Review policy) 페이지에서 이름(Name)에 인라인 정책 이름을 입력합니다(예: **`SessionManagerPermissions`**)
9. (선택 사항) 설명에 정책에 대한 설명을 입력합니다.
10. 정책 생성을 선택합니다.
11. 탐색 창에서 역할을 선택한 후 역할 생성을 선택합니다.

12. 역할 생성 페이지에서 AWS 서비스(service)를 선택하고 사용 사례(Use case)에 EC2를 선택합니다.
13. 다음(Next)을 선택합니다.
14. 권한 추가 페이지에서 방금 생성한 정책의 이름(예: **SessionManagerPermissions**) 왼쪽에 있는 확인란을 선택합니다.
15. Next(다음)를 선택합니다.
16. 이름, 검토 및 생성 페이지에서 역할 이름(Role name)에 IAM 역할의 이름을 입력합니다(예: **MySessionManagerRole**).
17. (선택 사항) 역할 설명에 인스턴스 프로파일에 대한 설명을 입력합니다.
18. (선택 사항) 태그 추가(Add tag)를 선택하고 역할에 대한 기본 설정 태그를 입력하여 태그를 추가합니다.

    역할 생성을 선택합니다.

`ssmmessages` 작업에 대한 자세한 내용은 참조: ec2messages, ssmmessages 및 기타 API 작업 섹션을 참조하세요.

Session Manager, Amazon S3 및 CloudWatch Logs에 대한 권한이 있는 IAM 역할 생성(콘솔)

다음 절차에 따라 인스턴스에 대해 Session Manager 작업이 가능한 권한을 제공하는 정책을 사용해 사용자 지정 IAM 역할을 생성합니다. 이 정책은 세션 로그를 Amazon Simple Storage Service(Amazon S3) 버킷 및 Amazon CloudWatch Logs 로그 그룹에 저장하는 데 필요한 권한도 제공합니다.

> **Important**
>
> 다른 AWS 계정에서 소유하는 Amazon S3 버킷에 세션 로그를 출력하려면 정책에 IAM 역할 정책에 `s3:PutObjectAcl` 권한을 추가해야 합니다. 소유하는 계정에서 사용하는 IAM 역할에 버킷 정책에서 크로스 계정 액세스 권한을 부여하여 관리형 인스턴스에 대한 Systems Manager 권한을 부여하는지도 확인해야 합니다. 버킷에서 Key Management Service(KMS) 암호화를 사용하는 경우에는 버킷의 KMS 정책에서도 이 크로스 계정 액세스 권한을 부여해야 합니다. Amazon S3의 크로스 계정 버킷 권한 구성에 대한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 크로스 계정 버킷 권한 부여를 참조하세요. 크로스 계정 권한이 추가되지 않으면 Amazon S3 버킷을 소유하는 계정에서 세션 출력 로그에 액세스할 수 없습니다.

세션 로그 저장에 대한 기본 설정 지정에 관한 자세한 내용은 세션 활동 로깅 활성화 및 비활성화 섹션을 참조하세요.

Session Manager, Amazon S3 및 CloudWatch Logs에 대한 권한이 있는 IAM 역할 생성(콘솔)

1. AWS Management Console에 로그인하여 https://console.aws.amazon.com/iam/ 에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 정책을 선택한 후 정책 생성을 선택합니다. ([시작하기(Get Started)] 버튼이 표시되면 이 버튼을 선택한 후 [정책 생성(Create Policy)]을 선택합니다.)
3. JSON 탭을 선택합니다.
4. 기본 콘텐츠를 다음과 같은 정책으로 바꿉니다. 각 *### ## ### #*를 자신의 정보로 바꿉니다.

```
{
    "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:::DOC-EXAMPLE-BUCKET/s3-prefix/*"
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetEncryptionConfiguration"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "key-name"
        },
        {
            "Effect": "Allow",
            "Action": "kms:GenerateDataKey",
            "Resource": "*"
        }
    ]
}
```

5. 다음: 태그를 선택합니다.
6. (선택 사항) 태그 추가(Add tag)를 선택하고 정책에 대한 기본 설정 태그를 입력하여 태그를 추가합니다.
7. 다음: 검토를 선택합니다.
8. 정책 검토(Review policy) 페이지에서 이름(Name)에 인라인 정책 이름을 입력합니다(예: **SessionManagerPermissions**)
9. (선택 사항) 설명에 정책에 대한 설명을 입력합니다.
10. 정책 생성을 선택합니다.
11. 탐색 창에서 역할을 선택한 후 역할 생성을 선택합니다.
12. 역할 생성 페이지에서 AWS 서비스(service)를 선택하고 사용 사례(Use case)에 EC2를 선택합니다.
13. 다음(Next)을 선택합니다.
14. 권한 추가 페이지에서 방금 생성한 정책의 이름(예: **SessionManagerPermissions**) 왼쪽에 있는 확인란을 선택합니다.
15. Next(다음)를 선택합니다.

16. 이름, 검토 및 생성 페이지에서 역할 이름(Role name)에 IAM 역할의 이름을 입력합니다(예: **MySessionManagerRole**).
17. (선택 사항) Role description(역할 설명)에 역할에 대한 설명을 입력합니다.
18. (선택 사항) 태그 추가(Add tag)를 선택하고 역할에 대한 기본 설정 태그를 입력하여 태그를 추가합니다.
19. 역할 생성을 선택합니다.

3단계: 관리형 노드에 대한 세션 액세스 제어

AWS Identity and Access Management(IAM) 정책을 사용하여 관리형 노드에 대한 Session Manager 액세스 권한을 부여하거나 취소할 수 있습니다. 정책을 생성하여 사용자 또는 그룹이 연결할 수 있는 관리형 노드를 지정하는 IAM 사용자 또는 그룹에 해당 정책을 연결할 수 있습니다. 사용자 또는 그룹이 해당 관리 노드에서 수행할 수 있는 Session Manager API 작업을 지정할 수도 있습니다.

Session Manager IAM 권한 정책을 시작하는 데 도움이 되도록 최종 사용자와 관리자 사용자를 위한 샘플 정책을 생성했습니다. 이러한 정책을 약간 변경하여 사용할 수 있습니다. 사용자 지정 IAM 정책을 생성하는 안내서로 사용할 수도 있습니다. 자세한 내용은 Session Manager에 대한 샘플 IAM 정책 단원을 참조하십시오. 정책을 생성하고 IAM 사용자 또는 그룹에 연결하는 방법에 대한 자세한 내용은 IAM 사용 설명서의 IAM 정책 생성과 IAM 정책 추가 및 제거를 참조하세요.

세션 ID ARN 형식 정보

Session Manager 액세스를 위한 IAM 정책을 생성할 때 Amazon 리소스 이름(ARN)의 일부로 세션 ID를 지정합니다. 세션 ID에는 사용자 이름이 변수로 포함됩니다. 설명하는 데 도움이 되는 Session Manager ARN 형식과 예제가 있습니다.

```
arn:aws:ssm:region-id:account-id:session/session-id
```

예:

```
arn:aws:ssm:us-east-2:123456789012:session/JohnDoe-1a2b3c4d5eEXAMPLE
```

IAM 정책에서 변수 사용에 대한 자세한 내용은 IAM 정책 요소: 변수를 참조하세요.

주제

- IAM 정책에서 기본 세션 문서를 지정하여 기본 쉘 세션을 시작합니다.
- IAM 정책에서 세션 문서를 지정하여 문서로 세션 시작

- Session Manager에 대한 샘플 IAM 정책
- Session Manager에 대한 추가 IAM 정책 샘플

IAM 정책에서 기본 세션 문서를 지정하여 기본 쉘 세션을 시작합니다.

Systems Manager 콘솔에서 AWS 계정의 Session Manager를 구성하거나 세션 기본 설정을 변경하면 시스템에서 `SSM-SessionManagerRunShell`이라는 SSM 세션 문서가 생성됩니다. 이 문서가 기본 세션 문서입니다. Session Manager에서는 이 문서를 사용하여 다음과 같은 정보가 포함된 세션 기본 설정을 저장합니다.

- 세션 데이터를 저장하려는 위치(예: Amazon Simple Storage Service(S3) 버킷 또는 Amazon CloudWatch Logs 로그 그룹)입니다.
- 세션 데이터 암호화용 AWS Key Management Service(AWS KMS) 키 ID입니다.
- 세션에 Run As 지원이 허용되는지 여부입니다.

다음은 `SSM-SessionManagerRunShell` 세션 기본 설정 문서에 포함된 정보의 예입니다.

```
{
  "schemaVersion": "1.0",
  "description": "Document to hold regional settings for Session Manager",
  "sessionType": "Standard_Stream",
  "inputs": {
    "s3BucketName": "DOC-EXAMPLE-BUCKET",
    "s3KeyPrefix": "MyS3Prefix",
    "s3EncryptionEnabled": true,
    "cloudWatchLogGroupName": "MyCWLogGroup",
    "cloudWatchEncryptionEnabled": false,
    "kmsKeyId": "1a2b3c4d",
    "runAsEnabled": true,
    "runAsDefaultUser": "RunAsUser"
  }
}
```

기본적으로 Session Manager에서는 사용자가 AWS Management Console에서 세션을 시작할 때 기본 세션 문서를 사용합니다. 이는 Systems Manager 콘솔의 Fleet Manager 또는 Session Manager나 Amazon EC2 콘솔의 EC2 Connect에 적용됩니다. Session Manager에서는 사용자가 다음 예제와 같은 AWS CLI 명령을 사용하여 세션을 시작할 때에도 기본 세션 문서를 사용합니다.

```
aws ssm start-session \
```

```
    --target i-02573cafcfEXAMPLE
```

기본 쉘 세션을 시작하려면 다음 예제와 같이 IAM 정책에서 기본 세션 문서를 지정해야 합니다.

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

IAM 정책에서 세션 문서를 지정하여 문서로 세션 시작

기본 세션 문서를 사용하여 start-session AWS CLI 명령을 사용하는 경우 문서 이름을 생략할 수 있습니다. 시스템에서 자동으로 `SSM-SessionManagerRunShell` 세션 문서를 호출합니다.

그 밖의 경우에는 `document-name` 파라미터의 값을 지정해야 합니다. 사용자가 명령에서 세션 문서의 이름을 지정하면 시스템에서는 IAM 정책을 점검하여 문서에 액세스할 권한이 있는지 확인합니다. 권한이 없으면 연결 요청에 실패합니다. 다음 예제에는 `document-name` 파라미터가 `AWS-StartPortForwardingSession` 세션 문서와 함께 포함되어 있습니다.

```
aws ssm start-session \
    --target i-02573cafcfEXAMPLE \
    --document-name AWS-StartPortForwardingSession \
    --parameters '{"portNumber":["80"], "localPortNumber":["56789"]}'
```

세션을 시작할 때 세션 문서 권한 확인 적용

`AWS-StartPortForwardingSession` 세션 문서에 대한 액세스를 제한하려면 사용자가 세션 문서에 명시적으로 액세스할 수 있는지 여부를 확인하는 조건 요소를 사용자의 IAM 정책에 추가할 수 있습

니다. 이 조건이 적용되면 사용자가 start-session 명령의 document-name 옵션에 값을 지정해야 합니다. IAM 정책의 ssm:StartSession 작업에 추가되면 다음 조건 요소는 Session 문서 액세스 확인을 수행합니다.

```
"Condition": {
    "BoolIfExists": {
        "ssm:SessionDocumentAccessCheck": "true"
    }
}
```

사용자가 세션을 시작할 수 있도록 이 조건 요소를 true로 설정하여 IAM 정책에서 세션 문서에 대한 액세스 권한을 명시적으로 부여해야 합니다. 조건 요소가 적용되도록 하려면 ssm:StartSession 작업을 허용하는 모든 정책 설명에 포함되어야 합니다. 의 예는 다음과 같습니다.

```
{
    "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::document/AWS-StartPortForwardingSession"
            ],
            "Condition": {
                "BoolIfExists": {
                    "ssm:SessionDocumentAccessCheck": "true"
                }
            }
        }
    ]
}
```

이 IAM 정책을 적용하면 SessionDocumentAccessCheck 조건 요소가 true로 설정된 경우 사용자가 AWS CLI를 사용하여 세션을 시작할 때 명령에 document-name 파라미터를 입력해야 합니다. document-name의 값은 IAM 정책의 Resource 섹션에 지정된 문서여야 합니다. 사용자가 다른 문서 이름을 입력하거나 document-name 파라미터를 지정하지 않으면 요청에 실패합니다.

`SessionDocumentAccessCheck` 조건 요소가 `false`로 설정되어 있으면 IAM 정책 평가에 영향을 미치지 않습니다.

IAM 정책에서 Session Manager 세션 문서를 지정하는 예제는 Session Manager에 대한 빠른 시작 최종 사용자 정책 섹션을 참조하세요.

기타 시나리오

SSH로 세션을 시작하려면 대상 관리형 노드와 사용자의 로컬 시스템 모두에서 구성 단계를 완료해야 합니다. 자세한 내용은 (선택 사항) Session Manager를 통한 SSH 연결에 대한 권한 허용 및 제어를 참조하세요.

Session Manager에 대한 샘플 IAM 정책

이 섹션에 나오는 샘플은 Session Manager 액세스에 가장 일반적으로 필요한 권한을 제공하는 AWS Identity and Access Management(IAM) 정책을 생성하는 데 유용합니다.

> **Note**
>
> 또한 AWS KMS key 정책을 사용하면 KMS 키에 대한 액세스 권한이 부여될 IAM 엔터티(사용자 또는 역할) 및 AWS 계정를 제어할 수 있습니다. 자세한 내용은 AWS Key Management Service Developer Guide의 Overview of Managing Access to Your AWS KMS Resources 및 Using Key Policies in AWS KMS를 참조하세요.

주제

- Session Manager에 대한 빠른 시작 최종 사용자 정책
- Session Manager에 대한 빠른 시작 관리자 정책

Session Manager에 대한 빠른 시작 최종 사용자 정책

다음 예를 사용하여 Session Manager에 대한 IAM 최종 사용자 정책을 생성합니다.

사용자가 Session Manager 콘솔과 AWS Command Line Interface(AWS CLI)에서만, Amazon Elastic Compute Cloud(Amazon EC2) 콘솔에서만 또는 세 가지 모두에서 세션을 시작할 수 있도록 허용하는 정책을 생성할 수 있습니다.

이러한 정책은 최종 사용자에게 특정 관리형 노드에 대한 세션을 시작할 수 있는 권한과 자신의 세션만 종료할 수 있는 권한을 제공합니다. 정책에 대해 수행하려는 사용자 지정에 대한 예는 Session Manager에 대한 추가 IAM 정책 샘플 섹션을 참조하세요.

다음 샘플 정책에서 각 *### ## ### #*를 자신의 정보로 바꿉니다.

제공할 세션 액세스 범위에 대한 샘플 정책을 보려면 다음 섹션을 참조하세요.

세션 관리자 and Fleet Manager

사용자에게 Session Manager 콘솔과 Fleet Manager에서만 세션을 시작하고 재개할 수 있는 권한을 제공하려면 이 샘플 정책을 사용합니다.

```
{
    "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)
            ],
            "Condition": {
                "BoolIfExists": {
                    "ssm:SessionDocumentAccessCheck":
 "true" (2)
                }
            }
        },
        {
            "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" 3
            ],
            "Resource": "key-name"
        }
    ]
}
```

Amazon EC2

사용자에게 Amazon EC2 콘솔에서만 세션을 시작하고 재개할 수 있는 권한을 제공하려면 이 샘플 정책을 사용합니다. 이 정책은 Session Manager 콘솔과 AWS CLI에서 세션을 시작하는 데 필요한 모든 권한을 제공하지 않습니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession",

 "ssm:SendCommand" 4
            ],
            "Resource": [
                "arn:aws:ec2:region:account-id:instance/instance-id",
                "arn:aws:ssm:region:account-id:document/SSM-
SessionManagerRunShell" 1
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetConnectionStatus",
```

```
                "ssm:DescribeInstanceInformation"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:TerminateSession",
                "ssm:ResumeSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:userid}-*"
            ]
        }
    ]
}
```

AWS CLI

사용자에게 AWS CLI에서만 세션을 시작하고 재개할 수 있는 권한을 제공하려면 이 샘플 정책을 사용합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession",

 "ssm:SendCommand" 4
            ],
            "Resource": [
                "arn:aws:ec2:region:account-id:instance/instance-id",
                "arn:aws:ssm:region:account-id:document/SSM-
SessionManagerRunShell" 1
            ],
            "Condition": {
                "BoolIfExists": {
                    "ssm:SessionDocumentAccessCheck":
 "true" 2
                }
            }
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:TerminateSession",
                "ssm:ResumeSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:userid}-*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [

 "kms:GenerateDataKey" 3
            ],
            "Resource": "key-name"
        }
    ]
}
```

[1] SSM-SessionManagerRunShell은 Session Manager에서 세션 구성 설정을 저장할 목적으로 생성되는 SSM 문서의 기본 이름입니다. 그 밖에 사용자 정의 Session 문서를 생성한 후 정책에서 지정할 수도 있습니다. 또한 AWS에서 제공하는 문서인 AWS-StartSSHSession을 SSH로 세션을 시작하는 사용자에게 지정하는 방법도 있습니다. SSH로 세션을 지원하는 데 필요한 구성 단계에 대한 자세한 내용은 (선택 사항) Session Manager를 통한 SSH 연결에 대한 권한 허용 및 제어를 참조하세요.

[2] 조건 요소 ssm:SessionDocumentAccessCheck를 true로 지정한 경우 시스템은 세션을 설정하기 전에 이 예제 SSM-SessionManagerRunShell에서 정의된 Session 문서에 사용자가 명시적으로 액세스할 수 있는지 확인합니다. 자세한 내용은 세션을 시작할 때 세션 문서 권한 확인 적용 단원을 참조하십시오.

[3] kms:GenerateDataKey 권한을 사용하면 세션 데이터를 암호화하는 데 사용할 데이터 암호화 키를 생성할 수 있습니다. 세션 데이터에 AWS Key Management Service(AWS KMS) 암호화를 사용하는 경우 *key-name*을 arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-12345EXAMPLE 형식으로 사용하려는 KMS 키의 Amazon 리소스 이름(ARN)으로 바꿉니다. 세션 데이터에 KMS 키 암호화를 사용하지 않으려면 정책에서 다음 콘텐츠를 제거합니다.

```
,
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey"
            ],
            "Resource": "key-name"
        }
```

세션 데이터 암호화에 AWS KMS 사용에 대한 내용은 세션 데이터의 KMS 키 암호화를 설정하려면(콘솔) 섹션을 참조하세요.

[4] 사용자가 Amazon EC2 콘솔에서 세션을 시작하려고 시도하는 경우 SendCommand에 대한 권한이 필요하지만, 먼저 Session Manager에 필요한 최소 버전으로 SSM Agent를 업데이트해야 합니다. Run Command를 사용하여 에이전트를 업데이트하라는 명령을 인스턴스에 전달합니다.

Session Manager에 대한 빠른 시작 관리자 정책

다음 예를 사용하여 Session Manager에 대한 IAM 관리자 정책을 생성합니다.

이러한 정책은 관리자에게 `Key=Finance,Value=WebServers`로 태그가 지정된 관리형 노드에 대한 세션을 시작하는 권한, 기본 설정을 생성, 업데이트 및 삭제하는 권한, 자신의 세션만 종료할 수 있는 권한을 제공합니다. 정책에 대해 수행하려는 사용자 지정에 대한 예는 Session Manager에 대한 추가 IAM 정책 샘플 섹션을 참조하세요.

관리자가 Session Manager 콘솔과 AWS CLI에서만, Amazon EC2 콘솔에서만 또는 세 가지 모두에서 이러한 태스크를 수행할 수 있도록 허용하는 정책을 생성할 수 있습니다.

다음 샘플 정책에서 각 *### ## ### #*를 자신의 정보로 바꿉니다.

세 가지 권한 시나리오에 대한 샘플 정책을 보려면 다음 섹션을 참조하세요.

세션 관리자 and CLI

관리자에게 Session Manager 콘솔과 AWS CLI에서만 세션 관련 태스크를 수행할 수 있는 권한을 제공하려면 이 샘플 정책을 사용합니다. 이 정책은 Amazon EC2 콘솔에서 세션 관련 태스크를 수행하는 데 필요한 모든 권한을 제공하지 않습니다.

```
{
    "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": [
                "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": [
                "ssm:TerminateSession",
                "ssm:ResumeSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:userid}-*"
```

```
            ]
        }
    ]
}
```

Amazon EC2

관리자에게 Amazon EC2 콘솔에서만 세션 관련 태스크를 수행할 수 있는 권한을 제공하려면 이 샘플 정책을 사용합니다. 이 정책은 Session Manager 콘솔 및 AWS CLI에서 세션 관련 작업을 수행하는 데 필요한 모든 권한을 제공하지 않습니다.

```
{
    "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": [
                "ssm:GetConnectionStatus",
                "ssm:DescribeInstanceInformation"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:TerminateSession",
                "ssm:ResumeSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:userid}-*"
            ]
        }
    ]
}
```

세션 관리자, CLI, and Amazon EC2

관리자에게 Session Manager 콘솔, AWS CLI 및 Amazon EC2 콘솔에서 세션 관련 태스크를 수행할 수 있는 권한을 제공하려면 이 샘플 정책을 사용합니다.

```
{
    "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: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] 사용자가 Amazon EC2 콘솔에서 세션을 시작하려고 하는 경우 SendCommand에 대한 권한이 필요하지만 먼저 SSM Agent를 업데이트하도록 명령을 보내야 합니다.

Session Manager에 대한 추가 IAM 정책 샘플

다음 예제 정책을 참조하면 지원하려는 모든 Session Manager 사용자 액세스 시나리오에 대한 사용자 정의 AWS Identity and Access Management(IAM) 정책을 생성할 수 있습니다.

주제

예제 1: 콘솔의 문서 액세스 권한 부여

사용자가 Session Manager 콘솔을 사용하여 세션을 시작할 때 사용자 지정 문서를 지정하도록 허용할 수 있습니다. 다음 예제 IAM 정책에서는 지정된 AWS 리전 및 AWS 계정에서 이름이 **`SessionDocument-`**로 시작하는 문서에 액세스하는 권한을 부여합니다.

이 정책을 사용하려면 각 *`## ### ## ###`*를 자신의 정보로 바꿉니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetDocument",
                "ssm:ListDocuments"
            ],
            "Resource": [
                "arn:aws:ssm:region:account-id:document/SessionDocument-*"

            ],
            "Condition": {
                "BoolIfExists": {
                    "ssm:SessionDocumentAccessCheck": "true"
                }
            }
        }
    ]
```

```
}
```

> **Note**
>
> Session Manager 콘솔에서는 세션 기본 설정 정의에 사용되는 `Standard_Stream`의 `sessionType`이 있는 세션 문서만 지원됩니다. 자세한 내용은 Session 문서 스키마 단원을 참조하십시오.

예제 2: 특정 관리형 노드에 대한 액세스 제한

Session Manager로 사용자가 연결할 수 있는 관리형 노드를 정의하는 IAM 정책을 생성할 수 있습니다. 예를 들어, 다음 정책은 사용자에게 3개의 특정 노드에서 세션을 시작, 종료 및 재개할 수 있는 권한을 부여합니다. 이 정책은 사용자가 지정된 노드 이외의 노드에 연결하지 못하도록 제한합니다.

> **Note**
>
> 페더레이션 사용자의 경우 예제 4: 사용자에게 자신이 시작한 세션만 종료하도록 허용 섹션을 참조하세요.

```
{
    "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": [
```

```
                "ssm:TerminateSession",
                "ssm:ResumeSession"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:userid}-*"
            ]
        }
    ]
}
```

예제 3: 태그에 따라 액세스 제한

특정 태그를 기준으로 관리형 노드에 대한 액세스 권한을 제한할 수 있습니다. 다음 예제에서 사용자는 관리형 노드가 Finance WebServer(`ssm:resourceTag/Finance: WebServer`)인 상태에서 모든 관리형 노드(`Resource: arn:aws:ec2:region:987654321098:instance/*`)에 대해 세션(`Effect: Allow, Action: ssm:StartSession, ssm:ResumeSession`)을 시작하고 재개할 수 있습니다. 사용자가 태그 지정되지 않은 관리형 노드에 명령을 전송하거나 `Finance: WebServer` 이외의 태그가 있는 경우 명령 결과에 `AccessDenied`가 포함됩니다.

```
{
    "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": [
                "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"
            ]
        }
    ]
}
```

사용자가 여러 태그로 지정된 관리형 노드에 대해 세션을 시작하도록 허용하는 IAM 정책을 생성할 수 있습니다. 다음 정책에서는 사용자가 관리형 노드에 적용된 태그를 둘 다 가지고 있는 인스턴스에 대한 세션을 시작하도록 합니다. 사용자가 두 태그 모두에 지정되지 않은 관리형 노드에 명령을 전송할 경우 명령 결과에 AccessDenied가 포함됩니다.

```
{
   "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": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ssm:us-east-2:123456789012:document/SSM-
SessionManagerRunShell"
            ]
      }
   ]
}
```

IAM 정책 생성에 대한 자세한 내용은 IAM 사용 설명서의 관리형 정책과 인라인 정책을 참조하세요. 관리형 노드 태그 지정에 대한 자세한 내용은 Amazon EC2 사용 설명서의 관리형 노드 태깅 및 Amazon EC2 리소스에 태그 지정을 참조하세요(콘텐츠는 Windows 및 Linux 관리형 노드에 적용됨). 관리형 노드에서 무단 루트 수준 명령에 대한 보안 태세를 강화하는 방법은 SSM Agent를 통한 루트 수준 명령에 대한 액세스 제한 섹션을 참조하세요.

예제 4: 사용자에게 자신이 시작한 세션만 종료하도록 허용

Session Manager에서는 AWS 계정의 페더레이션 사용자가 종료할 수 있는 세션을 제어하는 두 가지 방법이 제공됩니다.

- AWS Identity and Access Management(IAM) 권한 정책에서 `{aws:userid}` 변수를 사용합니다. 페더레이션 사용자는 자신이 시작한 세션만 종료할 수 있습니다. 페더레이션 사용자가 아닌 사용자는 `{aws:userid}` 대신에 변수 `{aws:username}`을 사용합니다.
- IAM 권한 정책에서 AWS 태그에서 제공하는 태그를 사용합니다. 정책에서, 사용자가 AWS에서 제공한 특정 태그로 태그 달린 세션만 종료할 수 있는 조건을 포함시킵니다. 이 방법은 페더레이션 ID를 사용하여 AWS에 대한 액세스 권한을 부여하는 계정을 포함하여 모든 계정에 작동합니다.

방법 1: 변수 **`{aws:username}`**를 사용하여 TerminateSession에 권한 부여

다음 IAM 정책은 사용자가 계정에 있는 모든 세션의 ID를 볼 수 있도록 합니다. 그러나 사용자는 자신이 시작한 세션을 통해서만 관리형 노드와 상호 작용할 수 있습니다. 다음 정책이 할당된 사용자는 다른 사용자의 세션에 연결하거나 해당 세션을 종료할 수 없습니다. 이 정책은 변수 `{aws:username}`를 사용하여 이와 같이 제한합니다.

> **Note**
>
> 이 방법은 페더레이션 ID를 사용하여 AWS에 대한 액세스 권한을 부여하는 계정에는 적용되지 않습니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ssm:DescribeSessions"
            ],
            "Effect": "Allow",
            "Resource": [
                "*"
            ]
        },
        {
            "Action": [
                "ssm:TerminateSession"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:ssm:*:*:session/${aws:username}-*"
            ]
        }
    ]
}
```

방법 2: AWS에서 제공한 태그를 사용하여 TerminateSession에 권한 부여

IAM 정책에 조건부 태그 키 변수를 포함하여 사용자가 종료할 수 있는 세션을 제어할 수 있습니다. 조건은 사용자가 이러한 특정 태그 키 변수와 지정된 값 중 하나 또는 둘 다로 태그가 지정된 세션만 종료할 수 있도록 지정합니다.

AWS 계정의 사용자가 세션을 시작하면 Session Manager에서는 두 개의 리소스 태그를 세션에 적용합니다. 첫 번째 리소스 태그는 `aws:ssmmessages:target-id`이며, 이 태그에서는 사용자가 종료할 수 있는 대상의 ID를 지정합니다. 다른 리소스 태그는 `aws:ssmmessages:session-id`이며, *`role-id`*`:`*`caller-specified-role-name`* 형식의 값입니다.

> **Note**
>
> Session Manager에서는 이 IAM 액세스 제어 정책에 대한 사용자 정의 태그를 지원하지 않습니다. 아래에 설명된 AWS에서 제공하는 리소스 태그를 사용해야 합니다.

aws:ssmmessages:target-id

이 태그 키를 사용하여 관리형 노드 ID를 값으로 정책에 포함시킵니다. 다음 정책 블록에서, 조건문을 사용하여 사용자가 i-02573cafcfEXAMPLE 노드만 종료할 수 있습니다.

```
{
     "Version": "2012-10-17",
     "Statement": [
         {
             "Effect": "Allow",
             "Action": [
                "ssm:TerminateSession"
             ],
             "Resource": "*",
             "Condition": {
                 "StringLike": {
                     "ssm:resourceTag/aws:ssmmessages:target-id": [
                        "i-02573cafcfEXAMPLE"
                     ]
                 }
             }
         }
     ]
}
```

사용자가 이 TerminateSession 권한이 부여되지 않은 세션을 종료하려고 시도하면 AccessDeniedException 오류가 발생합니다.

aws:ssmmessages:session-id

이 태그 키는 세션 ID에 대한 변수를 값으로 세션 시작 요청에 포함시킵니다.

다음 예제에서는 호출자 유형이 User인 경우에 대한 정책을 보여줍니다. aws:ssmmessages:session-id에 대해 제공하는 값은 사용자의 ID입니다. 이 예에서 AIDIODR4TAW7CSEXAMPLE은 AWS 계정에 있는 사용자의 ID를 나타냅니다. AWS 계정에서 사

용자의 ID를 검색하려면 IAM 명령 `get-user`를 사용합니다. 자세한 내용은 IAM User Guide에서 AWS Identity and Access Management 섹션의 get-user를 참조하세요.

```
{
     "Version": "2012-10-17",
     "Statement": [
         {
             "Effect": "Allow",
             "Action": [
                "ssm:TerminateSession"
             ],
             "Resource": "*",
             "Condition": {
                 "StringLike": {
                     "ssm:resourceTag/aws:ssmmessages:session-id": [
                        "AIDIODR4TAW7CSEXAMPLE"
                     ]
                 }
             }
         }
     ]
}
```

다음 예제에서는 호출자 유형이 `AssumedRole`인 경우에 대한 정책을 보여줍니다. `aws:ssmmessages:session-id`에서 제공한 값에 `{aws:userid}` 변수를 사용할 수 있습니다. 또는 `aws:ssmmessages:session-id`에서 제공한 값에 대해 역할 ID를 하드코딩할 수 있습니다. 역할 ID를 하드코딩하는 경우 값을 *`role-id`*`:`*`caller-specified-role-name`* 형식으로 제공해야 합니다. 예를 들면 `AIDIODR4TAW7CSEXAMPLE:MyRole`입니다.

> **Important**
>
> 시스템 태그를 적용하려면 제공하는 역할 ID에 다음 문자만 포함할 수 있습니다. 유니코드 문자, 0-9, 공백, _, ., :, /, =, +, -, @ 및 \.

AWS 계정의 역할에 대한 역할 ID를 검색하려면 `get-caller-identity` 명령을 사용합니다. 자세한 내용은 AWS CLI Command Reference의 get-caller-identity를 참조하세요.

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

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "ssm:TerminateSession"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "ssm:resourceTag/aws:ssmmessages:session-id": [
                       "${aws:userid}*"
                    ]
                }
            }
        }
    ]
}
```

사용자가 이 TerminateSession 권한이 부여되지 않은 세션을 종료하려고 시도하면 AccessDeniedException 오류가 발생합니다.

aws:ssmmessages:target-id 및 **aws:ssmmessages:session-id**

또한 이 예제에 표시된 대로 사용자가 두 시스템 태그로 태그가 지정된 세션을 종료할 수 있도록 하는 IAM 정책을 생성할 수도 있습니다.

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

```
            }
        }
    ]
}
```

예제 5: 모든 세션에 대한 전체(관리) 액세스 권한 허용

다음 IAM 정책은 사용자가 다른 모든 사용자가 생성한 관리형 노드 및 세션 모두와 완벽하게 상호 작용하도록 허용합니다. 이러한 권한은 조직의 Session Manager 활동을 완벽하게 제정해야 하는 관리자에게만 부여해야 합니다.

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

4단계: 세션 기본 설정 구성

AWS Identity and Access Management(IAM) 정책의 관리 권한이 부여된 사용자는 다음을 포함하여 세션 기본 설정을 구성할 수 있습니다.

- Linux 관리형 노드에 대한 Run As 지원을 활성화합니다. 그러면 AWS Systems Manager Session Manager가 관리형 노드에서 생성할 수 있는 `ssm-user` 계정의 자격 증명이 아닌 특정 운영 체제 사용자의 자격 증명을 사용해 세션을 시작할 수 있습니다.
- AWS KMS key 암호화를 사용해 클라이언트 시스템과 관리형 노드 사이에서 전송되는 데이터를 추가로 보호할 수 있도록 Session Manager를 구성합니다.

- 세션 기록 로그를 생성하여 Amazon Simple Storage Service(Amazon S3) 버킷 또는 Amazon CloudWatch Logs 로그 그룹으로 보내도록 Session Manager를 구성합니다. 저장된 로그 데이터는 세션 중 인스턴스에 대한 세션 연결과 관리형 노드에 대한 명령 실행을 감사 또는 보고하는 데 사용할 수 있습니다.
- 세션 시간 제한을 구성합니다. 이 설정을 사용하여 일정 기간 동안 활동이 없으면 세션을 종료할 시기를 지정할 수 있습니다.
- 구성 가능한 셸 프로파일을 사용하도록 Session Manager를 구성합니다. 이러한 사용자 정의 가능한 프로파일을 사용하면 셸 기본 설정, 환경 변수, 작업 디렉터리 및 세션 시작 시 여러 명령 실행과 같은 세션 내 기본 설정을 정의할 수 있습니다.

Session Manager 기본 설정 구성에 필요한 권한에 대한 자세한 내용은 the section called “Session Manager 기본 설정을 업데이트하도록 사용자 권한 부여 또는 거부” 섹션을 참조하세요.

주제

- Session Manager 기본 설정을 업데이트하도록 사용자 권한 부여 또는 거부
- 유휴 세션 시간 제한 값 지정
- 최대 세션 기간 지정
- 구성 가능한 셸 프로파일 허용
- Linux 및 macOS 관리형 노드에 대한 Run As 지원 활성화
- 세션 데이터의 KMS 키 암호화를 설정하려면(콘솔)
- Session Manager 기본 설정 문서 생성(명령줄)
- Session Manager 기본 설정 업데이트(명령줄)

Systems Manager 콘솔에서 세션 데이터 로깅 옵션을 구성하는 방법에 대한 자세한 내용은 다음 주제를 참조하세요.

- Amazon S3를 사용하여 세션 데이터 로깅(콘솔)
- Amazon CloudWatch Logs를 사용하여 세션 데이터 스트리밍(콘솔)
- Amazon CloudWatch Logs를 사용하여 세션 데이터 로깅(콘솔)

Session Manager 기본 설정을 업데이트하도록 사용자 권한 부여 또는 거부

계정 기본 설정은 각 AWS 리전에 대해 AWS Systems Manager(SSM) 문서로 저장됩니다. 사용자가 계정 내 세션에 대한 계정 기본 설정을 업데이트할 수 있으려면 기본 설정이 저장되는 SSM 문서 유형

에 액세스할 수 있는 필수 권한을 부여 받은 상태여야 합니다. 이러한 권한은 AWS Identity and Access Management(IAM) 정책을 통해 부여됩니다.

기본 설정이 생성 및 업데이트되도록 허용하는 관리자 정책

관리자는 언제든지 기본 설정을 생성 및 업데이트할 수 있도록 다음 정책을 가지고 있을 수 있습니다. 다음 정책은 us-east-2 계정 123456789012에서 `SSM-SessionManagerRunShell` 문서에 액세스하고 해당 문서를 업데이트하는 권한을 허용합니다.

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

기본 설정 업데이트를 방지하는 사용자 정책

다음 정책을 사용하면 계정의 최종 사용자가 Session Manager 기본 설정을 업데이트 또는 재정의할 수 없습니다.

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

유휴 세션 시간 제한 값 지정

AWS Systems Manager의 기능인 Session Manager를 사용하여 시스템이 세션을 끝내기 전에 사용자의 비활성 상태를 허용하는 시간을 지정할 수 있습니다. 기본적으로 20분 동안 활동이 없으면 세션이 시간 초과됩니다. 이 설정을 수정하여 1~60분 동안 비활성 상태이면 세션이 시간 초과되도록 지정할 수 있습니다. 일부 전문 컴퓨팅 보안 기관에서는 유휴 세션 시간 제한을 최대 15분으로 설정할 것을 권장합니다.

유휴 세션 시간 제한을 허용하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Session Manager를 선택합니다.
3. 기본 설정 탭을 선택하고 편집을 선택합니다.
4. [유휴 세션 시간 제한(Idle session timeout)] 아래의 [분(minutes)] 필드에 세션이 종료되기 전 사용자가 비활성 상태일 수 있는 시간을 지정합니다.
5. Save(저장)를 선택합니다.

최대 세션 기간 지정

AWS Systems Manager의 기능 Session Manager를 사용하면 세션이 종료되기 전의 최대 지속 시간을 지정할 수 있습니다. 기본적으로 세션에는 최대 기간이 없습니다. 최대 세션 기간에 대해 지정하는 값은 1~1,440분 사이여야 합니다.

최대 세션 기간 지정(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Session Manager를 선택합니다.
3. 기본 설정 탭을 선택하고 편집을 선택합니다.

4. 최대 세션 기간 활성화(Enable maximum session duration) 옆의 체크박스를 선택합니다.
5. 최대 세션 기간(Maximum session duration) 밑의 분(minutes) 필드에서 세션이 종료되기 전의 최대 지속 시간을 지정합니다.
6. Save(저장)를 선택합니다.

구성 가능한 셸 프로파일 허용

기본적으로 Linux용 EC2 인스턴스의 세션은 Bourne 쉘(sh)을 사용하여 시작합니다. 그러나 bash 등의 다른 셸을 사용하고자 할 수 있습니다. 구성 가능한 셸 프로파일을 허용하여 세션 내에서 셸 기본 설정, 환경 변수, 작업 디렉터리 및 세션이 시작될 때 여러 명령 실행과 같은 기본 설정을 사용자 지정할 수 있습니다.

> **Important**
>
> Systems Manager 는 셸 프로파일의 명령이나 스크립트를 검사하여 실행되기 전에 인스턴스에 어떤 변화가 있는지 확인하지 않습니다. 사용자가 셸 프로파일에 입력한 명령이나 스크립트를 수정할 수 있는 권한을 제한하려면 다음을 권장합니다.
>
> - AWS Identity and Access Management(IAM) 사용자 및 역할에 대한 사용자 지정된 세션 유형 문서를 생성합니다. 그런 다음 이러한 사용자 및 역할에 대한 IAM 정책을 수정하여 `StartSession` API 작업이 해당 사용자에 대해 생성한 세션 유형 문서만 사용할 수 있도록 합니다. 자세한 내용은 Session Manager 기본 설정 문서 생성(명령줄) 및 Session Manager에 대한 빠른 시작 최종 사용자 정책 섹션을 참조하세요.
> - IAM 사용자 및 역할에 대한 IAM 정책을 수정하여 생성한 세션 유형 문서 리소스에 대한 `UpdateDocument` API 작업 액세스를 거부합니다. 이렇게 하면 사용자와 역할이 설정을 수정하지 않고도 세션 기본 설정에 대해 생성한 문서를 사용할 수 있습니다.

구성 가능한 셸 프로파일을 설정하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Session Manager를 선택합니다.
3. 기본 설정 탭을 선택하고 편집을 선택합니다.
4. 해당 운영 체제의 필드에서 세션이 시작될 때 실행할 환경 변수, 셸 기본 설정 또는 명령을 지정합니다.
5. Save(저장)를 선택합니다.

다음은 셸 프로파일에 추가할 수 있는 몇 가지 예제 명령입니다.

bash 쉘로 변경하고 Linux 인스턴스의 /usr 디렉터리로 변경합니다.

```
exec /bin/bash
cd /usr
```

세션 시작 시 타임스탬프 및 시작 메시지를 출력합니다.

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."
```

세션 시작 시 동적 시스템 활동을 봅니다.

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 "-------  ------    -----      ----- -----   ------     -- -----------"}
```

Linux 및 macOS 관리형 노드에 대한 Run As 지원 활성화

기본적으로 Session Manager는 관리형 노드에 생성된 시스템 생성 `ssm-user` 계정의 보안 인증 정보를 사용하여 연결을 인증합니다. Linux 및 macOS 시스템에서는 이 계정이 `/etc/sudoers/`에 추가됩니다. 선택하는 경우 대신 운영 체제(OS) 사용자 계정의 보안 인증 정보를 사용하여 세션을 인증할 수 있습니다. 이 경우 Session Manager는 세션을 시작하기 전에 지정한 OS 계정이 노드에 존재하는지 확인합니다. 노드에 존재하지 않는 OS 계정을 사용하여 세션을 시작하려고 하면 연결이 실패합니다.

> **Note**
>
> Session Manager는 운영 체제의 `root` 사용자 계정을 사용한 연결 인증을 지원하지 않습니다. OS 사용자 계정을 사용하여 인증된 세션의 경우 로그인 제한 또는 시스템 리소스 사용 제한과 같은 노드의 OS 수준 및 디렉터리 정책이 적용되지 않을 수 있습니다.

작동 방식

세션에서 Run As 지원을 설정하면 시스템이 다음과 같이 액세스 권한 유무를 검사합니다.

1. 세션을 시작하는 사용자의 경우 IAM 엔터티(사용자 또는 역할)에 `SSMSessionRunAs = os user account name`으로 태그가 지정되어 있나요?

   그렇다면 관리형 노드에 OS 사용자 이름이 존재하나요? 그렇다면 세션을 시작합니다. 그렇지 않다면 세션 시작을 허용하지 않습니다.

   IAM 엔터티에 `SSMSessionRunAs = os user account name`으로 태그가 지정되지 않은 경우 2단계로 계속 진행합니다.
2. IAM 엔터티에 `SSMSessionRunAs = os user account name`로 태그가 지정되어 있지 않은 경우 AWS 계정의 Session Manager 기본 설정에서 OS 사용자 이름이 지정되어 있나요?

   그렇다면 관리형 노드에 OS 사용자 이름이 존재하나요? 그렇다면 세션을 시작합니다. 그렇지 않다면 세션 시작을 허용하지 않습니다.

> **Note**
>
> Run As 지원을 활성화하면 Session Manager가 관리형 노드에서 `ssm-user` 계정을 사용하여 세션을 시작할 수 없습니다. 즉, Session Manager가 지정된 OS 사용자 계정을 사용하여 연결하지 못하는 경우 기본 방법을 사용하는 연결로 되돌아가지 않습니다.

OS 계정을 지정하거나 IAM 엔터티에 태그를 지정하지 않고 Run As를 활성화하고 Session Manager 기본 설정에서 OS 계정을 지정하지 않은 경우 세션 연결 시도가 실패합니다.

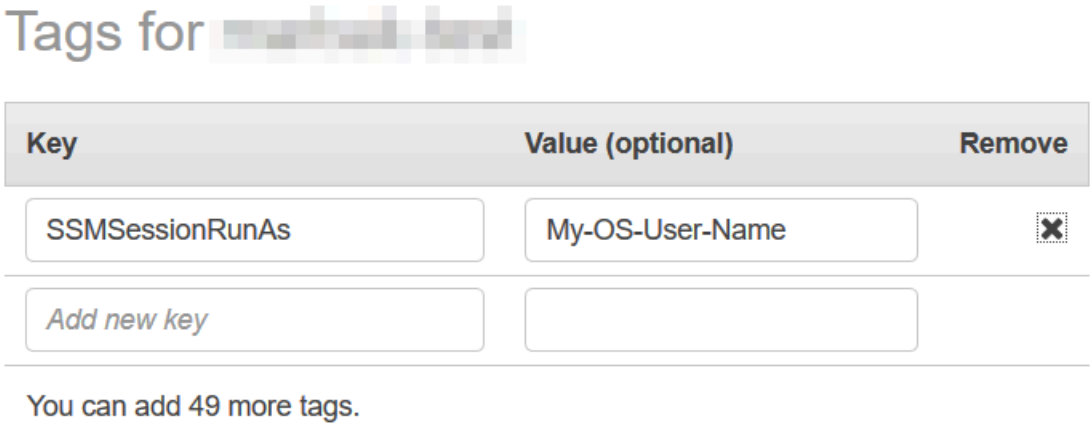
Linux 및 macOS 관리형 노드에 대한 Run As 지원을 활성화하는 방법

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Session Manager를 선택합니다.
3. 기본 설정 탭을 선택하고 편집을 선택합니다.
4. Linux 인스턴스에 대한 Run As 지원 활성화 옆에 있는 확인란을 선택합니다.
5. 다음 중 하나를 수행하십시오.

   - 옵션 1: 운영 체제 사용자 이름 필드에 세션을 시작할 때 사용할 OS 사용자 계정의 이름을 입력합니다. 이 옵션을 사용하면 Session Manager를 사용하여 연결하는 AWS 계정의 모든 사용자에 대해 동일한 OS 사용자가 모든 세션을 실행합니다.
   - 옵션 2(권장): IAM 콘솔 링크를 선택합니다. 탐색 창에서 사용자 또는 역할을 선택합니다. 태그를 추가할 개체(사용자 또는 역할)와 태그 탭을 차례대로 선택합니다. 키 이름으로 `SSMSessionRunAs`를 입력합니다. 키 값에 대한 OS 사용자 계정의 이름을 입력합니다. Save changes(변경 사항 저장)를 선택합니다.

     이 옵션을 사용하면 원하는 경우 서로 다른 IAM 엔터티에 대해 고유한 OS 사용자를 지정할 수 있습니다. IAM 엔터티(사용자 또는 역할) 태그 지정에 대한 자세한 내용은 IAM 사용 설명서의 IAM 리소스 태그 지정을 참조하세요.

     다음은 예입니다.



6. Save(저장)를 선택합니다.

세션 데이터의 KMS 키 암호화를 설정하려면(콘솔)

키를 생성하고 관리하려면 AWS Key Management Service(AWS KMS)을 사용합니다. AWS KMS로 다양한 AWS 서비스 및 애플리케이션의 암호화 사용을 제어할 수 있습니다. KMS 키 암호화를 사용하여 관리형 노드와 AWS 계정의 사용자 로컬 머신 사이에 전송되는 세션 데이터가 암호화되도록 지정할 수 있습니다. (이는 AWS에서 기본적으로 제공하는 TLS 1.2 암호화에 대한 추가 사항입니다.) Session Manager 세션 데이터를 암호화하려면 AWS KMS를 사용하여 대칭 KMS 키를 생성합니다.

`Standard_Stream`, `InteractiveCommands`, `NonInteractiveCommands` 세션 유형에 AWS KMS 암호화를 사용할 수 있습니다. AWS KMS(AWS Systems Manager)에서 생성한 키를 사용하여 세션 데이터를 암호화하는 옵션을 사용하려면 버전 2.3.539.0 이상의 SSM Agent가 관리형 노드에 설치되어 있어야 합니다.

> **Note**
>
> AWS Systems Manager 콘솔에서 관리형 노드의 암호를 재설정하려면 AWS KMS 암호화를 허용해야 합니다. 자세한 내용은 관리형 노드의 암호 재설정 단원을 참조하십시오.

AWS 계정에서 생성한 키를 사용할 수 있습니다. 다른 AWS 계정에서 생성된 키를 사용할 수도 있습니다. 다른 AWS 계정에 있는 키 생성자는 키를 사용하는 데 필요한 권한을 제공해야 합니다.

세션 데이터에 대해 KMS 키 암호화를 설정하면 세션을 시작하는 사용자와 거기에 연결하는 관리형 노드 모두에 키를 사용할 권한이 있어야 합니다. Session Manager과 함께 AWS Identity and Access Management(IAM) 정책을 통해 KMS 키를 사용할 수 있는 권한을 제공합니다. 자세한 내용은 다음 주제를 참조하세요.

- 계정의 사용자에 대한 AWS KMS 권한 추가: Session Manager에 대한 샘플 IAM 정책.
- 계정의 관리형 노드에 대한 AWS KMS 권한 추가: 2단계: Session Manager의 인스턴스 권한 확인 또는 추가.

KMS 키 생성 및 관리에 대한 자세한 내용은 AWS Key Management Service Developer Guide를 참조하세요.

계정에서 AWS CLI를 사용하여 세션 데이터의 KMS 키 암호화를 설정하는 방법에 대한 자세한 내용은 Session Manager 기본 설정 문서 생성(명령줄) 또는 Session Manager 기본 설정 업데이트(명령줄) 섹션를 참조하세요.

> **Note**
>
> KMS 키 사용에 요금이 부과됩니다. 자세한 내용은 AWS Key Management Service 요금을 참조하십시오.

세션 데이터의 KMS 키 암호화를 설정하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Session Manager를 선택합니다.
3. 기본 설정 탭을 선택하고 편집을 선택합니다.
4. KMS 암호화 활성화(Enable KMS encryption) 옆의 체크박스를 선택합니다.
5. 다음 중 하나를 수행하십시오.

   - [내 현재 계정에서 KMS 키 선택(Select a KMS key in my current account)] 옆에 있는 버튼을 선택한 다음 목록에서 키를 선택합니다.

     -또는-

     Enter a KMS key alias or KMS key ARN(KMS 키 별칭 또는 KMS 키 ARN 입력) 옆의 버튼을 선택하십시오. 현재 계정에서 생성한 키의 KMS 키 별칭을 수동으로 입력하거나 다른 계정의 키에 대한 키 Amazon 리소스 이름(ARN)을 입력합니다. 예를 들어, 다음과 같습니다.

     - 키 별칭: `alias/my-kms-key-alias`
     - 키 ARN: `arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-12345EXAMPLE`

     -또는-

     [새 키 생성(Create new key)]을 선택하여 계정에서 새 KMS 키를 생성합니다. 새 키를 생성한 후 기본 설정 탭으로 돌아가서 계정의 세션 데이터를 암호화할 키를 선택하십시오.

   키 공유에 대한 자세한 내용은 AWS Key Management Service Developer Guide의 Allowing External AWS 계정 to Access a key를 참조하세요.
6. Save(저장)를 선택합니다.

Session Manager 기본 설정 문서 생성(명령줄)

다음 절차를 사용하여 AWS Systems Manager Session Manager 세션의 기본 설정을 정의하는 SSM 문서를 생성합니다. 문서를 사용하여 데이터 암호화, 세션 기간 및 로깅을 포함한 세션 옵션을 구성할 수 있습니다. 예를 들면, 기본 설정을 사용하여 Amazon Simple Storage Service(S3) 버킷 또는 Amazon CloudWatch Logs 로그 그룹에 세션 로그 데이터를 저장할지를 지정할 수 있습니다. AWS 계정 및 AWS 리전의 모든 세션에 대한 일반 기본 설정을 정의하거나 개별 세션의 기본 설정을 정의하는 문서를 생성할 수 있습니다.

> **Note**
>
> Session Manager 콘솔을 사용하여 일반 세션 기본 설정을 구성할 수도 있습니다.

Session Manager 기본 설정을 설정하는 데 사용하는 문서에는 `Standard_Stream`의 `sessionType`이 있어야 합니다. 세션 문서에 대한 자세한 내용은 the section called “Session 문서 스키마” 섹션을 참조하세요.

명령줄을 사용하여 기존 Session Manager 기본 설정을 업데이트하는 방법에 대한 자세한 내용은 Session Manager 기본 설정 업데이트(명령줄) 섹션을 참조하세요.

AWS CloudFormation을 사용하여 세션 환경설정을 생성하는 방법의 예제는 AWS CloudFormation 사용 설명서의 Session Manager 기본 설정에 대한 Systems Manager 문서 생성을 참조하세요.

> **Note**
>
> 이 절차에서는 AWS 계정 수준에서 Session Manager 기본 설정을 설정하는 문서를 생성하는 방법을 설명합니다. 세션 수준 기본 설정을 설정하는 데 사용할 문서를 생성하려면 파일 이름 관련 명령 입력에 대한 `SSM-SessionManagerRunShell` 이외의 값을 지정합니다.
> 문서를 사용하여 AWS Command Line Interface(AWS CLI)에서 시작하는 세션의 기본 설정을 설정하려면 문서 이름을 `--document-name` 파라미터 값으로 제공합니다. Session Manager 콘솔에서 시작하는 세션의 기본 설정을 설정하려면 문서 이름을 입력하거나 목록에서 선택할 수 있습니다.

Session Manager 기본 설정을 생성하려면(명령줄)

1. 로컬 시스템에서 `SessionManagerRunShell.json`이라는 JSON 파일을 생성한 다음 이 파일에 다음 내용을 붙여 넣습니다.

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

다음 예와 같이 값을 하드코딩하는 대신 파라미터를 사용하여 세션 기본 설정에 값을 전달할 수도 있습니다.

```
{
   "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. 세션 데이터를 보낼 위치를 지정하십시오. S3 버킷 이름(선택적 접두사 사용) 또는 CloudWatch Logs 로그 그룹 이름을 지정할 수 있습니다. 로컬 클라이언트와 관리형 노드 사이에 데이터를 추가로 암호화하려면 암호화에 사용할 KMS 키를 제공하세요. 다음은 예입니다.

```
{
  "schemaVersion": "1.0",
  "description": "Document to hold regional settings for Session Manager",
  "sessionType": "Standard_Stream",
  "inputs": {
    "s3BucketName": "DOC-EXAMPLE-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**
>
> 세션 로그 데이터를 암호화하지 않으려는 경우 `s3EncryptionEnabled`의 `true`를 `false`로 변경합니다.
> Amazon S3 버킷 또는 CloudWatch Logs 로그 그룹에 로그를 보내지 않거나, 활성 세션 데이터를 암호화하지 않거나, 계정 세션에서 Run As 지원을 설정하지 않으려면 해당 옵션에 대한 행을 삭제할 수 있습니다. `inputs` 섹션의 마지막 줄은 쉼표로 끝나면 안 됩니다.
> KMS 키 ID를 추가하여 세션 데이터를 암호화하면 세션을 시작하는 사용자와 거기에 연결하는 관리형 노드 모두에 키를 사용할 권한이 있어야 합니다. Session Manager와 함께 IAM 정책을 통해 KMS 키를 사용할 수 있는 권한을 제공합니다. 자세한 내용은 다음 주제를 참조하세요.
>
> - 계정의 사용자에 대한 AWS KMS 권한 추가: Session Manager에 대한 샘플 IAM 정책
> - 계정의 관리형 노드에 대한 AWS KMS 권한 추가: 2단계: Session Manager의 인스턴스 권한 확인 또는 추가.

3. 파일을 저장합니다.
4. JSON 파일을 생성한 디렉터리에서 다음 명령을 실행합니다.

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
```

이 작업이 성공하면 다음과 비슷한 출력이 반환됩니다.

```
{
    "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"
    }
}
```

Session Manager 기본 설정 업데이트(명령줄)

다음 절차에서는 기본 설정된 명령줄 도구를 사용하여 선택한 AWS 리전에서 AWS 계정에 대한 AWS Systems Manager Session Manager 기본 설정을 변경하는 방법을 설명합니다. Session Manager 기본 설정을 사용하여 Amazon Simple Storage Service(Amazon S3) 버킷 또는 Amazon CloudWatch Logs 로그 그룹에서 세션 데이터를 로그하는 옵션을 지정합니다. Session Manager 기본 설정을 사용하여 세션 데이터를 암호화할 수도 있습니다.

Session Manager 기본 설정을 업데이트하려면(명령줄)

1. 로컬 시스템에서 SessionManagerRunShell.json이라는 JSON 파일을 생성한 다음 이 파일에 다음 내용을 붙여 넣습니다.

```
{
    "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. 세션 데이터를 보낼 위치를 지정하십시오. S3 버킷 이름(선택적 접두사 사용) 또는 CloudWatch Logs 로그 그룹 이름을 지정할 수 있습니다. 로컬 클라이언트와 관리형 노드 사이에 데이터를 추가로 암호화하려면 암호화에 사용할 AWS KMS key를 제공하세요. 다음은 예입니다.

```
{
  "schemaVersion": "1.0",
  "description": "Document to hold regional settings for Session Manager",
  "sessionType": "Standard_Stream",
  "inputs": {
    "s3BucketName": "DOC-EXAMPLE-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**
>
> 세션 로그 데이터를 암호화하지 않으려는 경우 `s3EncryptionEnabled`의 `true`를 `false`로 변경합니다.
> Amazon S3 버킷 또는 CloudWatch Logs 로그 그룹에 로그를 보내지 않거나, 활성 세션 데이터를 암호화하지 않거나, 계정 세션에서 Run As 지원을 설정하지 않으려면 해당 옵션에 대한 행을 삭제할 수 있습니다. `inputs` 섹션의 마지막 줄은 쉼표로 끝나면 안 됩니다.
> KMS 키 ID를 추가하여 세션 데이터를 암호화하면 세션을 시작하는 사용자와 거기에 연결하는 관리형 노드 모두에 키를 사용할 권한이 있어야 합니다. Session Manager과 함께 AWS Identity and Access Management(IAM) 정책을 통해 KMS 키를 사용할 수 있는 권한을 제공합니다. 자세한 내용은 다음 주제를 참조하세요.
>
> - 계정의 사용자에 대한 AWS KMS 권한 추가: Session Manager에 대한 샘플 IAM 정책.
> - 계정의 관리형 노드에 대한 AWS KMS 권한 추가: 2단계: Session Manager의 인스턴스 권한 확인 또는 추가.

3. 파일을 저장합니다.
4. JSON 파일을 생성한 디렉터리에서 다음 명령을 실행합니다.

   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'
```

이 작업이 성공하면 다음과 비슷한 출력이 반환됩니다.

```
{
    "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"
    }
}
```

5단계: (선택 사항) 세션의 명령에 대한 액세스 제한

사용자 지정 `Session` 유형 AWS Systems Manager(SSM) 문서를 사용하여 사용자가 AWS Systems Manager Session Manager세션에서 실행할 수 있는 명령을 제한할 수 있습니다. 문서에서 사용자가 세션을 시작할 때 실행할 수 있는 명령과 사용자가 명령에 제공할 수 있는 파라미터를 정의합니다. `Session` 문서 `schemaVersion`은 1.0이어야 하고 문서의 `sessionType`은 `InteractiveCommands`여야 합니다. 그러면 직접 정의하는 `Session` 문서에만 사용자가 액세스할 수 있도록 허용하는 AWS Identity and Access Management(IAM) 정책을 생성할 수 있습니다. IAM 정책을 사용하여 세션의 명령에 대한 액세스를 제한하는 방법에 대한 자세한 내용은 대화형 명령에 대한 IAM 정책 예제 섹션을 참조하세요.

`InteractiveCommands`의 `sessionType`이 포함된 문서는 AWS Command Line Interface(AWS CLI) 에서 시작된 세션의 경우에만 지원됩니다. 사용자는 사용자 지정 문서 이름을 `--document-name` 파라미터 값으로 제공하고 `--parameters` 옵션을 사용하여 모든 명령 파라미터 변수 값을 제공합니다. 대화형 명령 실행에 대한 자세한 내용은 세션 시작(대화형 및 비대화형 명령) 섹션을 참조하세요.

다음 절차를 사용하여 사용자가 실행할 수 있는 명령을 정의하는 사용자 지정 `Session` 유형 SSM 문서를 생성합니다.

세션의 명령에 대한 액세스 제한(콘솔)

사용자가 Session Manager 세션에서 실행할 수 있는 명령을 제한하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. 명령 또는 세션 생성을 선택합니다.
4. [이름(Name)]에 문서에 대한 설명이 포함된 이름을 입력합니다.
5. 문서 유형에서 세션 문서를 선택합니다.
6. 다음 예제와 같이 사용자가 JSON 또는 YAML을 사용하여 Session Manager 세션에서 실행할 수 있는 명령을 정의하는 문서 내용을 입력합니다.

   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. 문서 생성을 선택합니다.

세션의 명령에 대한 액세스 제한(명령줄)

시작하기 전 준비 사항

아직 하지 않은 경우 AWS Command Line Interface(AWS CLI) 또는 AWS Tools for PowerShell을 설치하고 구성합니다. 자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트 및 AWS Tools for PowerShell 설치를 참조하세요.

사용자가 Session Manager 세션에서 실행할 수 있는 명령을 제한하려면(명령줄)

1. 다음 예제와 같이 사용자가 Session Manager 세션에서 실행할 수 있는 명령을 정의하는 문서 내용에 JSON 또는 YAML 파일을 생성합니다.

   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. 다음 명령을 실행하여 사용자가 Session Manager 세션에서 실행할 수 있는 명령을 정의하는 내용을 사용하여 SSM 문서를 생성합니다.

   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"
   ```

대화형 명령 파라미터 및 AWS CLI

AWS CLI를 사용할 때 대화형 명령 파라미터를 제공할 수 있는 다양한 방법이 있습니다. AWS CLI가 있는 관리형 노드에 연결하는 데 사용하는 클라이언트 시스템의 운영 체제(OS)에 따라 특수 문자나 이스케이프 문자가 포함된 명령에 제공하는 구문이 다를 수 있습니다. 다음 예는 AWS CLI 사용 시 명령 파라미터를 제공할 수 있는 몇 가지 다른 방법과 특수 문자 또는 이스케이프 문자를 처리하는 방법을 보여줍니다.

Parameter Store에 저장된 파라미터는 다음 예와 같이 명령 파라미터에 대해 AWS CLI에서 참조할 수 있습니다.

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}}"]}'
```

다음 예에서는 AWS CLI에 약식 구문을 사용하여 파라미터를 전달하는 방법을 보여줍니다.

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"
```

다음 예와 같이 JSON에서 파라미터를 제공할 수도 있습니다.

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"]}'
```

파라미터를 JSON 파일에 저장하고 다음 예와 같이 AWS CLI에 제공할 수도 있습니다. 파일에서 AWS CLI 파라미터 사용에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서의 파일에서 AWS CLI 파라미터 로드를 참조하세요.

```
{
    "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
```

다음 예와 같이 JSON 입력 파일에서 AWS CLI 스켈레톤을 생성할 수도 있습니다. JSON 입력 파일에서 AWS CLI 스켈레톤 생성에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서의 JSON 또는 YAML 입력 파일에서 AWS CLI 스켈레톤 및 입력 파라미터 생성을 참조하세요.

```
{
    "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
```

따옴표 안의 문자를 이스케이프 처리하려면 다음 예와 같이 이스케이프 문자에 백슬래시를 추가해야 합니다.

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\""]}'
```

AWS CLI에서 명령 파라미터에 따옴표를 사용에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서의 AWS CLI에서 문자열에 따옴표 사용을 참조하세요.

대화형 명령에 대한 IAM 정책 예제

직접 정의한 Session 문서에만 사용자가 액세스할 수 있도록 허용하는 IAM정책을 생성할 수 있습니다. 이렇게 하면 사용자가 Session Manager 세션에서 실행할 수 있는 명령이 사용자 정의 Session 유형 SSM 문서에 정의된 명령으로만 제한됩니다.

사용자가 단일 관리형 노드에서 대화형 명령을 실행하도록 허용

```
{
   "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"
         ],
         "Condition":{
            "BoolIfExists":{
               "ssm:SessionDocumentAccessCheck":"true"
            }
         }
      }
   ]
}
```

사용자가 모든 관리형 노드에서 대화형 명령을 실행하도록 허용

```
{
   "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"
         ],
         "Condition":{
            "BoolIfExists":{
               "ssm:SessionDocumentAccessCheck":"true"
```

```
            }
         }
      }
   ]
}
```

사용자가 모든 관리형 노드에서 여러 대화형 명령을 실행하도록 허용

```
{
   "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"
         ],
         "Condition":{
            "BoolIfExists":{
               "ssm:SessionDocumentAccessCheck":"true"
            }
         }
      }
   ]
}
```

6단계: (옵션) AWS PrivateLink를 사용하여 Session Manager에 대한 VPC 엔드포인트 설정

인터페이스 Virtual Private Cloud(VPC) 엔드포인트를 사용하도록 AWS Systems Manager를 구성하여 관리형 노드의 보안 태세를 개선할 수 있습니다. 인터페이스 엔드포인트는 프라이빗 IP 주소를 사용하여 Amazon Elastic Compute Cloud(Amazon EC2) 및 Systems Manager API에 비공개로 액세스할 수 있는 기술인 AWS PrivateLink로 구동됩니다.

AWS PrivateLink는 관리형 노드, Systems Manager 및 Amazon EC2 간의 모든 네트워크 트래픽을 Amazon 네트워크로 제한합니다. (관리형 노드는 인터넷에 액세스할 수 없음) 또한 인터넷 게이트웨이, NAT 디바이스 또는 가상 프라이빗 게이트웨이가 필요 없습니다.

VPC 엔드포인트 생성에 대한 자세한 내용은 Systems Manager용 VPC 엔드포인트를 사용하여 EC2 인스턴스의 보안 개선을 참조하세요.

VPC 엔드포인트 사용의 대체 방법은 관리형 노드에서 아웃바운드 인터넷 액세스를 허용하는 것입니다. 이 경우 관리형 노드는 다음 엔드포인트에 대한 HTTPS(포트 443) 아웃바운드 트래픽도 허용해야 합니다.

- `ec2messages.region.amazonaws.com`
- `ssm.region.amazonaws.com`
- `ssmmessages.region.amazonaws.com`

Systems Manager는 이러한 엔드포인트 중 마지막 엔드포인트인 `ssmmessages.region.amazonaws.com`을 사용하여 SSM Agent에서 클라우드의 Session Manager 서비스를 호출합니다.

AWS Key Management Service(AWS KMS)와 같은 선택적 기능을 사용하려면 암호화, Amazon CloudWatch Logs(CloudWatch 로그)로 로그를 스트리밍하고, Amazon Simple Storage Service(Amazon S3)로 로그를 전송하려면 HTTPS(포트 443) 아웃바운드 트래픽을 다음 엔드포인트로 허용해야 합니다.

- `kms.region.amazonaws.com`
- `logs.region.amazonaws.com`
- `s3.region.amazonaws.com`

Systems Manager의 필수 엔드포인트에 대한 자세한 내용은 참조: ec2messages, ssmmessages 및 기타 API 작업 섹션을 참조하세요.

7단계: (옵션) ssm-user 계정 관리 권한 설정 또는 해제

AWS Systems Manager SSM Agent 버전 2.3.50.0부터 에이전트에서 `ssm-user`라는 로컬 사용자 계정을 생성하여 `/etc/sudoers`(Linux 및 macOS) 또는 관리자 그룹(Windows)에 추가합니다. 2.3.612.0 이전 버전의 에이전트에서 SSM Agent 최초 시작 또는 설치 후 재시작 시 계정이 생성됩니다. 2.3.612.0 이후 버전에서 `ssm-user` 계정은 노드에서 세션을 처음 시작할 때 생성됩니다. AWS Systems Manager Session Manager 세션이 시작될 때 `ssm-user`가 기본 운영 체제(OS) 사용자입니다. SSM Agent 버전 2.3.612.0은 2019년 5월 8일에 릴리스되었습니다.

Session Manager 사용자가 노드에서 관리 명령을 실행하지 못하도록 하려는 경우 `ssm-user` 계정 권한을 업데이트할 수 있습니다. 또한 제거한 후 이러한 권한을 복원할 수도 있습니다.

주제

Linux 및 macOS에서 ssm-user sudo 계정 권한 관리

다음과 같은 절차 중 하나를 사용하여 Linux 및 macOS 관리형 노드에 대한 ssm-user 계정 sudo 권한을 설정하거나 해제합니다.

Run Command을 사용하여 ssm-user sudo 권한 수정(콘솔)

- 다음 값을 사용해 콘솔에서 명령 실행의 절차를 따릅니다.
  - 명령 문서(Command document에서 `AWS-RunShellScript`를 선택합니다.
  - sudo 액세스 권한을 제거하려면 [명령 파라미터(Command parameters)] 영역에서 [명령(Commands)] 상자에 다음을 붙여 넣습니다.

    ```
    cd /etc/sudoers.d
    echo "#User rules for ssm-user" > ssm-agent-users
    ```

    -또는-

    sudo 액세스를 복원하려면 명령 파라미터(Commands) 영역에서 명령(Commands) 상자에 다음을 붙여 넣습니다.

    ```
    cd /etc/sudoers.d
    echo "ssm-user ALL=(ALL) NOPASSWD:ALL" > ssm-agent-users
    ```

명령줄을 사용해 ssm-user sudo 권한 수정(AWS CLI)

1. 관리형 노드에 연결하고 다음 명령을 실행합니다.

   ```
   sudo -s
   ```

2. 다음 명령을 사용하여 작업 디렉터리를 변경합니다.

   ```
   cd /etc/sudoers.d
   ```

3. 편집하기 위해 `ssm-agent-users` 파일을 엽니다.
4. sudo 액세스를 제거하려면 다음 줄을 삭제합니다.

```
ssm-user ALL=(ALL) NOPASSWD:ALL
```

-또는-

sudo 액세스를 복원하려면 다음 줄을 추가합니다.

```
ssm-user ALL=(ALL) NOPASSWD:ALL
```

5. 파일을 저장합니다.

Windows Server에서 ssm-user 관리자 계정 권한 관리

다음 절차 중 하나를 수행해 Windows Server 관리형 노드에 대한 ssm-user 계정 관리자 권한을 설정하거나 해제합니다.

Run Command를 사용하여 관리자 권한 수정(콘솔)

- 다음 값을 사용해 콘솔에서 명령 실행의 절차를 따릅니다.

  명령 문서(Command document에서 `AWS-RunPowerShellScript`를 선택합니다.

  관리 액세스를 제거하려면 명령 파라미터(Command parameters) 영역의 명령(Commands) 상자에 다음 내용을 붙여 넣습니다.

```
net localgroup "Administrators" "ssm-user" /delete
```

  -또는-

  관리 액세스를 복원하려면 명령 파라미터(Command parameters) 영역의 명령(Commands) 상자에 다음 내용을 붙여 넣습니다.

```
net localgroup "Administrators" "ssm-user" /add
```

PowerShell 또는 명령 프롬프트 창을 사용하여 관리자 권한 수정

1. 관리형 노드에 연결하고 PowerShell 또는 명령 프롬프트 창을 엽니다.

2. 관리 액세스를 제거하려면 다음 명령을 실행합니다.

```
net localgroup "Administrators" "ssm-user" /delete
```

-또는-

관리 액세스를 복원하려면 다음 명령을 실행합니다.

```
net localgroup "Administrators" "ssm-user" /add
```

Windows 콘솔을 사용하여 관리자 권한 수정

1. 관리형 노드에 연결하고 PowerShell 또는 명령 프롬프트 창을 엽니다.
2. 명령줄에서 `lusrmgr.msc`를 실행하여 Local Users and Groups(로컬 사용자 및 그룹) 콘솔을 엽니다.
3. 사용자 디렉터리를 열고 ssm-user를 엽니다.
4. 소속 그룹(Member Of) 탭에서 다음 중 한 가지를 수행합니다.
   - 관리 액세스를 제거하려면 관리자를 선택하고 제거를 선택합니다.

     -또는-

     관리 액세스를 복원하려면 텍스트 상자에 **Administrators**를 입력한 다음 [추가(Add)]를 선택합니다.
5. 확인을 선택합니다.

8단계: (선택 사항) Session Manager를 통한 SSH 연결에 대한 권한 허용 및 제어

AWS Command Line Interface(AWS CLI)를 사용하여 AWS Systems Manager Session Manager를 통해 관리형 노드에 SSH(Secure Shell) 연결을 설정하도록 AWS 계정의 사용자를 허용할 수 있습니다. SSH로 연결하는 사용자는 로컬 시스템과 관리형 노드 사이에서 SCP(Secure Copy Protocol)를 사용하여 파일을 복사하는 것도 가능합니다. 이 기능을 사용하면 인바운드 포트를 개방하거나 Bastion 호스트를 유지하지 않고도 관리형 노드에 연결할 수 있습니다.

SSH 연결을 허용한 후 AWS Identity and Access Management(IAM) 정책을 사용하여 사용자, 그룹 또는 역할이 Session Manager를 사용한 SSH 연결을 명시적으로 허용하거나 거부하도록 할 수 있습니다.

> **Note**
>
> 포트 전달 또는 SSH를 통해 연결하는 Session Manager 세션에는 로깅을 사용할 수 없습니다. SSH는 모든 세션 데이터를 암호화하고 Session Manager는 SSH 연결을 위한 터널 역할만 하기 때문입니다.

주제

- Session Manager의 SSH 연결 허용
- Session Manager를 통해 SSH 연결에 대한 사용자 권한 제어

Session Manager의 SSH 연결 허용

다음 단계를 사용하여 관리형 노드에서 Session Manager를 통해 SSH 연결을 허용합니다.

Session Manager의 SSH 연결 허용

1. SSH 연결을 허용할 관리형 노드에서 다음과 같이 실행합니다.

   - 관리형 노드에서 SSH가 실행 중인지 확인합니다. (노드에서 인바운드 포트를 폐쇄할 수 있습니다)
   - SSM Agent 2.3.672.0 이상이 관리형 노드에 설치되어 있는지 확인합니다.

     관리형 노드에 SSM Agent를 설치하거나 업데이트하는 방법에 대한 자세한 내용은 다음 주제를 참조하세요.

     - SSM Agent용 EC2 인스턴스에 수동으로 Windows Server 설치 및 제거.
     - Linux용 EC2 인스턴스에 수동으로 SSM Agent 설치 및 제거
     - SSM Agent용 EC2 인스턴스에 수동으로 macOS 설치 및 제거
     - 하이브리드 Windows 노드에 SSM Agent를 설치하는 방법
     - 하이브리드 Linux 노드에 SSM Agent를 설치하는 방법

     > **Note**
     >
     > 관리형 노드로 활성화한 온프레미스 서버, 엣지 디바이스, 가상 머신에서 Session Manager를 사용하려면 고급 인스턴스 티어를 사용해야 합니다. 전용 인스턴스에 대한 자세한 내용은 인스턴스 티어 구성 섹션을 참조하세요.

2. SSH로 관리형 노드에 연결할 로컬 시스템에서 다음과 같이 실행합니다.

   - Session Manager 플러그인 버전 1.1.23.0 이상이 설치되어 있는지 확인합니다.

     Session Manager 플러그인 설치에 대한 자세한 내용은 AWS CLI의 Session Manager 플러그인 설치 섹션을 참조하세요.

   - 프록시 명령 실행을 허용하여 Session Manager 세션을 시작할 수 있도록 SSH 구성 파일을 업데이트한 후 SSH 연결을 통해 모든 데이터를 전송합니다.

     Linux 및 macOS

     > **Tip**
     >
     > SSH 구성 파일은 일반적으로 `~/.ssh/config`에 있습니다.

     로컬 시스템에서 다음과 같이 구성 파일에 추가합니다.

     ```
     # SSH over Session Manager
     host i-* mi-*
         ProxyCommand sh -c "aws ssm start-session --target %h --document-name AWS-
     StartSSHSession --parameters 'portNumber=%p'"
     ```

     Windows

     > **Tip**
     >
     > SSH 구성 파일은 일반적으로 `C:\Users\<username>\.ssh\config`에 있습니다.

     로컬 시스템에서 다음과 같이 구성 파일에 추가합니다.

     ```
     # 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"
     ```

   - 관리형 노드에 연결 설정 시 사용할 Privacy Enhanced Mail 인증서(PEM 파일) 또는 최소 퍼블릭 키를 생성하거나 갖추고 있는지 확인합니다. 이 키는 관리형 노드와 이미 연결된 키여야 합니

다. 사용자만 읽을 수 있도록 프라이빗 키 파일의 권한을 설정해야 합니다. 다음 명령을 사용하여 사용자만 읽을 수 있도록 프라이빗 키 파일의 권한을 설정할 수 있습니다.

```
chmod 400 <my-key-pair>.pem
```

예를 들어 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스의 경우 인스턴스를 생성할 때 선택 또는 생성한 키 페어 파일입니다. (세션을 시작하는 명령의 일부로 인증서 또는 키에 대한 경로를 지정합니다. SSH를 사용하여 세션 시작에 대한 자세한 내용은 세션 시작(SSH) 섹션을 참조하세요.)

Session Manager를 통해 SSH 연결에 대한 사용자 권한 제어

Session Manager를 통해 관리형 노드에서 SSH 연결을 활성화한 후 IAM 정책을 사용하여 사용자, 그룹 또는 역할이 Session Manager를 통해 SSH 연결을 수립할 수 있는 기능을 허용하거나 거부할 수 있습니다.

Session Manager를 통한 SSH 연결을 허용하는 IAM 정책을 사용하려면

- 다음 옵션 중 하나를 사용하십시오.
  - 옵션 1: https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.

    탐색 창에서 정책을 선택한 다음 Session Manager을 통해 SSH 연결을 시작할 수 있도록 허용할 사용자 또는 역할에 대한 권한 정책을 업데이트합니다.

    예를 들어 Session Manager에 대한 빠른 시작 최종 사용자 정책에서 생성한 빠른 시작 정책에 다음 요소를 추가합니다. 각 *### ## ### #*를 자신의 정보로 바꿉니다.

```
{
    "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"
            ],
            "Condition": {
                "BoolIfExists": {
```

```
                    "ssm:SessionDocumentAccessCheck": "true"
                }
            }
        }
    ]
}
```

- 옵션 2: AWS Management Console, AWS CLI 또는 AWS API를 사용하여 인라인 정책을 사용자 정책에 연결합니다.

  선택한 방법을 사용하여 옵션 1의 정책 문을 AWS 사용자, 그룹 또는 역할에 대한 정책에 연결합니다.

  자세한 내용은 IAM User Guide의 Adding and Removing IAM Identity Permissions를 참조하세요.

Session Manager를 통한 SSH 연결을 거부하는 IAM 정책을 사용하려면

- 다음 옵션 중 하나를 사용하십시오.

  - 옵션 1: https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다. 탐색 창에서 정책을 선택한 다음, Session Manager 세션의 시작을 차단하도록 사용자 또는 역할에 대한 권한 정책을 업데이트합니다.

    예를 들어 Session Manager에 대한 빠른 시작 최종 사용자 정책에서 생성한 빠른 시작 정책에 다음 요소를 추가합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor1",
            "Effect": "Deny",
            "Action": "ssm:StartSession",
            "Resource": "arn:aws:ssm:*:*:document/AWS-StartSSHSession"
        }
    ],
            "Condition": {
                "BoolIfExists": {
                    "ssm:SessionDocumentAccessCheck": "true"
                }
            }
```

```
}
```

- 옵션 2: AWS Management Console, AWS CLI 또는 AWS API를 사용하여 인라인 정책을 사용자 정책에 연결합니다.

  선택한 방법을 사용하여 옵션 1의 정책 문을 AWS 사용자, 그룹 또는 역할에 대한 정책에 연결합니다.

  자세한 내용은 IAM User Guide의 Adding and Removing IAM Identity Permissions를 참조하세요.

Session Manager 작업

AWS Systems Manager 콘솔, Amazon Elastic Compute Cloud(Amazon EC2) 콘솔 또는 AWS Command Line Interface(AWS CLI)로 시스템 관리자가 AWS Identity and Access Management(IAM) 정책을 사용하여 액세스 권한을 부여한 관리형 노드에 연결하는 세션을 시작할 수 있습니다. 권한에 따라 세션에 대한 정보를 보거나, 시간이 초과되지 않은 비활성 세션을 다시 시작하고, 세션을 종료할 수 있습니다. 세션이 설정된 후에는 IAM 역할 세션 기간의 영향을 받지 않습니다. Session Manager의 세션 기간 제한에 대한 자세한 내용은 유휴 세션 시간 제한 값 지정 및 최대 세션 기간 지정의 내용을 참조하세요.

세션에 대한 자세한 내용은 세션이란 무엇입니까? 섹션을 참조하세요.

주제

- AWS CLI의 Session Manager 플러그인 설치
- 세션 시작
- 세션 종료
- 세션 기록 보기

AWS CLI의 Session Manager 플러그인 설치

AWS Command Line Interface(AWS CLI)를 사용하여 관리형 노드로 Session Manager 세션을 시작하려면 로컬 시스템에 Session Manager 플러그인을 설치해야 합니다. Microsoft Windows Server, macOS, Linux, Ubuntu Server의 지원되는 버전에 플러그인을 설치할 수 있습니다.

> **Note**
>
> AWS CLI 플러그인을 사용하려면 로컬 시스템에 Session Manager 버전 1.16.12 이상이 설치되어 있어야 합니다. 자세한 내용은 최신 버전의 AWS Command Line Interface 설치 또는 업데이트를 참조하세요.

주제

Session Manager 플러그인 최신 버전 및 릴리스 기록

로컬 시스템에서 Session Manager 플러그인의 지원되는 버전이 실행 중이어야 합니다. 현재 지원되는 최소 버전은 1.1.17.0입니다. 이전 버전을 실행 중인 경우 Session Manager 작업에 실패할 수 있습니다.

최신 버전이 설치되어 있는지 확인하려면 AWS CLI에서 다음 명령을 실행합니다.

> **Note**
>
> 이 명령은 플러그인이 운영 체제 유형의 기본 설치 디렉터리에 있는 경우에만 결과를 반환합니다. 또한 플러그인을 설치한 디렉터리에 있는 `VERSION` 파일의 내용에서 버전을 확인할 수도 있습니다.

```
session-manager-plugin --version
```

다음 표에는 Session Manager 플러그인의 모든 릴리스와 각 릴리스에 포함된 기능 및 기능 향상이 나와 있습니다.

버전	릴리스 날짜	Details
1.2.633.0	2024년 5월 30일	향상된 기능: Amazon Elastic Container Registry(Amazon ECR) 이미지를 사용하도록 Dockerfile이 업데이트되었습니다.
1.2.553.0	2024년 1월 10일	향상된 기능: aws-sdk-go 및 종속 Golang 패키지가 업그레이드되었습니다.
1.2.536.0	2023년 12월 4일	향상된 기능: StartSession API 응답을 session-manager-plugin에 환경 변수로 전달하는 지원이 추가되었습니다.
1.2.497.0	2023년 8월 1일	향상된 기능: Go SDK를 v1.44.302로 업그레이드했습니다.
1.2.463.0	2023년 3월 15일	향상된 기능: macOS 번들 설치 관리자 및 서명된 설치 관리자에 Apple Mac(M1)에 대한 Mac with Apple silicon 지원이 추가되었습니다.
1.2.398.0	2022년 10월 14일	개선 사항: golang 버전 1.17을 지원합니다. macOS의 기본 세션 관리자 플러그인 실행기가 python3을 사용하도록 업데이트합니다. SSMCLI에서 세션 관리자 플러그인으로 가져오기 경로를 업데이트합니다.
1.2.339.0	2022년 6월 16일	버그 수정: 포트 세션의 유휴 세션 시간 초과를 수정했습니다.
1.2.331.0	2022년 5월 27일	버그 수정: 시간 초과 전에 로컬 서버가 연결되지 않을 때 포트 세션이 조기에 닫히는 문제를 수정했습니다.
1.2.323.0	2022년 5월 19일	버그 수정: 유휴 세션 시간 초과 기능을 사용하기 위해 smux 유지 기능을 비활성화합니다.
1.2.312.0	2022년 3월 31일	기능 향상: 더 많은 출력 메시지 페이로드 유형을 지원합니다.
1.2.295.0	2022년 1월 12일	버그 수정: 에이전트가 비활성화되고 `start_publication` 및 `pause_publication` 메시지에 대한 로그가 잘못되었을 때 클라이언트가 스트림 데이터를 재전송하여 세션이 중단되었습니다.

버전	릴리스 날짜	Details
1.2.279.0	2021년 10월 27일	향상된 기능: Windows 플랫폼용 Zip 패키징입니다.
1.2.245.0	2021년 8월 19일	기능 향상: `aws-sdk-go` 을(를) AWS IAM Identity Center을(를) 지원하는 최신 버전(v1.40.17)으로 업그레이드합니다.
1.2.234.0	2021년 7월 26일	버그 수정: 대화형 세션 유형에서 세션이 갑자기 종료된 시나리오를 처리합니다.
1.2.205.0	2021년 6월 10일	기능 향상: 서명된 macOS 설치 관리자에 대한 지원이 추가되었습니다.
1.2.54.0	2021년 1월 29일	기능 향상: NonInteractiveCommands 실행 모드에서 세션 실행에 대한 지원이 추가되었습니다.
1.2.30.0	2020년 11월 24일	기능 향상: (포트 전달 세션만 해당) 전반적인 성능이 향상되었습니다.
1.2.7.0	2020년 10월 15일	기능 향상: (포트 전달 세션에만 해당) 대기 시간이 줄고 전반적인 성능이 향상되었습니다.
1.1.61.0	2020년 4월 17일	기능 향상: Linux 및 Ubuntu에 대한 ARM 지원이 추가되었습니다.
1.1.54.0	2020년 1월 6일	버그 수정: Session Manager 플러그인이 준비되지 않은 경우 패킷이 삭제되는 교착 상태 시나리오를 처리합니다.
1.1.50.0	2019년 11월 19일	기능 향상: 포트를 로컬 Unix 소켓에 전달하기 위한 지원이 추가되었습니다.
1.1.35.0	2019년 11월 7일	기능 향상: (포트 전달 세션에만 해당) 로컬 사용자가 SSM Agent을 누를 때 TerminateSession 명령을 `Ctrl+C`에 보냅니다.
1.1.33.0	2019년 9월 26일	기능 향상: (포트 전달 세션만 해당) 클라이언트에서 TCP 연결이 해제되면 연결 해제 신호가 서버에 전송됩니다.
1.1.31.0	2019년 9월 6일	기능 향상: 원격 서버가 연결을 종료할 때까지 포트 전달 세션을 열어 둘 수 있도록 업데이트되었습니다.

버전	릴리스 날짜	Details
1.1.26.0	2019년 7월 30일	기능 향상: 세션 도중 데이터 전송 속도를 제한할 수 있도록 업데이트되었습니다.
1.1.23.0	2019년 7월 9일	기능 향상: Session Manager를 사용하여 SSH 세션을 실행할 수 있도록 지원이 추가되었습니다.
1.1.17.0	2019년 4월 4일	기능 향상: AWS Key Management Service(AWS KMS)를 사용하여 세션 데이터를 추가적으로 암호화할 수 있도록 지원이 추가되었습니다.
1.0.37.0	2018년 9월 20일	향상된 기능: Windows 버전의 버그를 수정했습니다.
1.0.0.0	2018년 9월 11일	Session Manager 플러그인의 최초 릴리스.

Windows에 Session Manager 플러그 인 설치

독립 실행형 설치 관리자를 사용하여 Windows Vista 이상에 Session Manager 플러그인을 설치할 수 있습니다.

업데이트가 릴리스되면 설치 프로세스를 반복하여 최신 버전의 Session Manager 플러그인을 가져와야 합니다.

> **Note**
>
> 최상의 결과를 위해 Windows PowerShell 버전 5 이상을 사용하여 Windows에서 세션을 시작하는 것이 좋습니다. Windows 10의 Command 쉘을 사용할 수도 있습니다. Session Manager 플러그인에서는 PowerShell 및 Command 쉘만 지원됩니다. 서드 파티 명령줄 도구는 플러그인과 호환되지 않을 수 있습니다.

EXE 설치 관리자를 사용하여 Session Manager 플러그인을 설치하려면

1. 다음 URL을 사용하여 설치 관리자를 다운로드합니다.

```
https://s3.amazonaws.com/session-manager-downloads/plugin/latest/windows/
SessionManagerPluginSetup.exe
```

또는 다음 URL을 사용하여 압축 버전의 설치 프로그램을 다운로드할 수 있습니다.

```
https://s3.amazonaws.com/session-manager-downloads/plugin/latest/windows/
SessionManagerPlugin.zip
```

2. 다운로드한 설치 관리자를 실행하고 화면의 지침을 따릅니다. 압축된 버전의 설치 프로그램을 다운로드한 경우 먼저 설치 프로그램의 압축을 풀어야 합니다.

   기본 디렉터리에 플러그인을 설치하려면 설치 위치 상자를 비워 둡니다.

   - `%PROGRAMFILES%\Amazon\SessionManagerPlugin\bin\`

3. 설치가 성공적인지 확인합니다. 자세한 내용은 Session Manager 플러그인 설치 확인을 참조하세요.

> **Note**
>
> Windows에서 실행 파일을 찾을 수 없는 경우 명령 프롬프트를 다시 열거나 설치 디렉터리를 PATH 환경 변수에 수동으로 추가하는 것이 좋습니다. 자세한 내용은 문제 해결 주제 명령줄 경로에 자동으로 추가되지 않는 Session Manager 플러그인(Windows) 섹션을 참조하세요.

macOS에 Session Manager 플러그 인 설치

다음 주제 중 하나를 선택하여 macOS에 Session Manager 플러그인을 설치합니다. 번들 설치 프로그램은 ZIP 파일을 사용합니다. 압축을 풀고 나면 이진수를 사용하여 플러그인을 설치할 수 있습니다. 서명된 설치 프로그램은 서명된 .pkg 파일입니다.

주제

- macOS에 Session Manager 플러그 인 설치
- 서명된 설치 관리자로 macOS에 Session Manager 플러그인 설치

macOS에 Session Manager 플러그 인 설치

이 섹션에서는 번들 설치 관리자를 사용하여 macOS에 Session Manager 플러그인을 설치하는 방법을 설명합니다.

> **Important**
>
> 번들 설치 관리자는 공백을 포함하는 경로에 설치하는 것을 지원하지 않습니다.

번들 설치 관리자를 사용하여 Session Manager 플러그인을 설치하려면(macOS)

1. 번들 설치 관리자를 다운로드합니다.

   x86_64

   ```
   curl "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/mac/
   sessionmanager-bundle.zip" -o "sessionmanager-bundle.zip"
   ```

   Apple 실리콘 탑재 Mac

   ```
   curl "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/
   mac_arm64/sessionmanager-bundle.zip" -o "sessionmanager-bundle.zip"
   ```

2. 패키지의 압축을 풉니다.

   ```
   unzip sessionmanager-bundle.zip
   ```

3. 설치 명령을 실행합니다.

   ```
   sudo ./sessionmanager-bundle/install -i /usr/local/sessionmanagerplugin -b /usr/
   local/bin/session-manager-plugin
   ```

   > **Note**
   >
   > 플러그인에 Python 2.6.5 이상 또는 Python 3.3 이상이 필요합니다. 기본적으로 설치 스크립트는 시스템 기본 버전의 Python에서 실행됩니다. 대체 버전의 Python을 설치하고 이를 사용하여 Session Manager 플러그인을 설치하려는 경우 Python 실행 파일에 대한 절대 경로로 해당 버전의 설치 스크립트를 실행합니다. 다음은 예입니다.
   >
   > ```
   > sudo /usr/local/bin/python3.8 sessionmanager-bundle/install -i /usr/local/
   > sessionmanagerplugin -b /usr/local/bin/session-manager-plugin
   > ```

설치 관리자는 /usr/local/sessionmanagerplugin에서 Session Manager 플러그인을 설치하고 /usr/local/bin 디렉터리에 symlink session-manager-plugin을 생성합니다. 이렇게 하면 사용자의 $PATH 변수에 설치 디렉터리를 지정할 필요가 없습니다.

-i 및 -b 옵션에 대한 설명을 보려면 -h 옵션을 사용합니다.

```
./sessionmanager-bundle/install -h
```

4. 설치가 성공적인지 확인합니다. 자세한 설명은 Session Manager 플러그인 설치 확인을 참조하세요.

> **Note**
>
> 플러그인을 제거하려면 다음 명령 2개를 표시된 순서로 실행합니다.
>
> ```
> sudo rm -rf /usr/local/sessionmanagerplugin
> ```
>
> ```
> sudo rm /usr/local/bin/session-manager-plugin
> ```

서명된 설치 관리자로 macOS에 Session Manager 플러그인 설치

이 섹션에서는 서명된 설치 관리자를 사용하여 macOS에 Session Manager 플러그인을 설치하는 방법을 설명합니다.

서명된 설치 관리자를 사용하여 Session Manager 플러그인을 설치하려면(macOS)

1. 서명된 설치 관리자를 다운로드합니다.

   x86_64

   ```
   curl "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/mac/
   session-manager-plugin.pkg" -o "session-manager-plugin.pkg"
   ```

Apple 실리콘 탑재 Mac

```
curl "https://s3.amazonaws.com/session-manager-downloads/plugin/latest/
mac_arm64/session-manager-plugin.pkg" -o "session-manager-plugin.pkg"
```

2. 설치 명령을 실행합니다.

```
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. 설치가 성공적인지 확인합니다. 자세한 설명은 Session Manager 플러그인 설치 확인을 참조하세요.

Amazon Linux 2 및 Red Hat Enterprise Linux 배포에 Session Manager 플러그인 설치

RHEL 배포판에 Session Manager 플러그인을 설치하려면 다음 절차를 사용하십시오.

> **Note**
>
> Amazon Linux 1에서는 Session Manager 플러그인이 지원되지 않습니다. Amazon Linux 2 이상에 기반한 배포판에서 지원됩니다.

1. Session Manager 플러그인 RPM 패키지를 다운로드하여 설치합니다.

   x86_64

   RHEL 7에서 다음 명령을 실행합니다.

   ```
   sudo yum install -y https://s3.amazonaws.com/session-manager-downloads/plugin/
   latest/linux_64bit/session-manager-plugin.rpm
   ```

   RHEL 8 및 9에서 다음 명령을 실행합니다.

   ```
   sudo dnf install -y https://s3.amazonaws.com/session-manager-downloads/plugin/
   latest/linux_64bit/session-manager-plugin.rpm
   ```

x86

RHEL 7에서 다음 명령을 실행합니다.

```
sudo yum install -y https://s3.amazonaws.com/session-manager-downloads/plugin/
latest/linux_32bit/session-manager-plugin.rpm
```

RHEL 8 및 9에서 다음 명령을 실행합니다.

```
sudo dnf install -y https://s3.amazonaws.com/session-manager-downloads/plugin/
latest/linux_32bit/session-manager-plugin.rpm
```

ARM64

RHEL 7에서 다음 명령을 실행합니다.

```
sudo yum install -y https://s3.amazonaws.com/session-manager-downloads/plugin/
latest/linux_arm64/session-manager-plugin.rpm
```

RHEL 8 및 9에서 다음 명령을 실행합니다.

```
sudo dnf install -y https://s3.amazonaws.com/session-manager-downloads/plugin/
latest/linux_arm64/session-manager-plugin.rpm
```

2. 설치가 성공적인지 확인합니다. 자세한 내용은 Session Manager 플러그인 설치 확인을 참조하세요.

> **Note**
>
> 플러그인을 제거하려는 경우 `sudo yum erase session-manager-plugin -y`를 실행합니다.

Debian Server 및 Ubuntu Server에 Session Manager 플러그인 설치

1. Session Manager 플러그인 deb 패키지를 다운로드합니다.

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. 설치 명령을 실행합니다.

```
sudo dpkg -i session-manager-plugin.deb
```

3. 설치가 성공적인지 확인합니다. 자세한 내용은 Session Manager 플러그인 설치 확인을 참조하세요.

> **Note**
>
> 플러그인을 제거하려는 경우 `sudo dpkg -r session-manager-plugin`를 실행합니다.

Session Manager 플러그인 설치 확인

다음 명령을 실행하여 Session Manager 플러그인이 성공적으로 설치되었는지 확인합니다.

```
session-manager-plugin
```

설치에 성공하면 다음 메시지가 반환됩니다.

```
The Session Manager plugin is installed successfully. Use the AWS CLI to start a
 session.
```

AWS Command Line Interface(AWS CLI)에서 start-session 명령을 실행하여 설치를 테스트할 수도 있습니다. 다음 명령에서 *instance-id*를 자신의 정보로 바꿉니다.

```
aws ssm start-session --target instance-id
```

이 명령은 AWS CLI를 설치 및 구성했으며 Session Manager 관리자가 Session Manager를 사용하여 대상 관리형 노드에 액세스하는 데 필요한 IAM 권한을 부여한 경우에만 작동합니다.

GitHub에서 Session Manager 플러그인 켜기

Session Manager 플러그인의 소스 코드는 GitHub에 제공되므로 필요에 따라 플러그인을 조정할 수 있습니다. 포함하고 싶은 변경에 대해서는 풀 요청을 제출하는 것이 좋습니다. 단, Amazon Web Services에서 이 소프트웨어의 수정된 사본 실행을 지원하지 않습니다.

(옵션) Session Manager 플러그인 로깅 설정

Session Manager 플러그인에는 실행하는 세션에 대한 로깅을 허용하는 옵션이 있습니다. 기본적으로 로깅은 해제되어 있습니다.

로깅을 허용하면 Session Manager 플러그인에서는 로컬 시스템에서 발생한 애플리케이션 활동(`session-manager-plugin.log`) 및 오류(`errors.log`)에 대한 로그 파일이 생성됩니다.

주제

- Session Manager 플러그인에 대한 로깅 켜기(Windows)
- Session Manager 플러그인에 대한 로깅 활성화(Linux 및 macOS)

Session Manager 플러그인에 대한 로깅 켜기(Windows)

1. 플러그인에 대한 `seelog.xml.template` 파일을 찾습니다.

   기본 위치는 `C:\Program Files\Amazon\SessionManagerPlugin\seelog.xml.template`입니다.
2. 이 파일의 이름을 `seelog.xml`로 변경합니다.
3. 파일을 열고 `minlevel="off"`를 `minlevel="info"` 또는 `minlevel="debug"`로 변경합니다.

> **Note**
>
> 기본적으로 데이터 채널 열기 및 세션 다시 연결에 대한 항목은 INFO 수준에서 기록됩니다. 데이터 흐름(패킷 및 승인) 항목은 DEBUG 수준에서 기록됩니다.

4. 수정하려는 다른 구성 옵션을 변경합니다. 변경 가능한 옵션은 다음과 같습니다.

   - 디버그 수준: 디버그 수준은 `formatid="fmtinfo"`에서 `formatid="fmtdebug"`로 변경할 수 있습니다.
   - 로그 파일 옵션: 로그 파일 이름을 제외하고 로그가 저장되는 위치를 포함해 로그 파일 옵션을 변경할 수 있습니다.

   > **Important**
   >
   > 파일 이름을 변경하지 않습니다. 그렇지 않으면 로깅이 제대로 작동하지 않습니다.

   ```
   <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. 파일을 저장합니다.

Session Manager 플러그인에 대한 로깅 활성화(Linux 및 macOS)

1. 플러그인에 대한 `seelog.xml.template` 파일을 찾습니다.

   기본 위치는 `/usr/local/sessionmanagerplugin/seelog.xml.template`입니다.

2. 이 파일의 이름을 `seelog.xml`로 변경합니다.
3. 파일을 열고 `minlevel="off"`를 `minlevel="info"` 또는 `minlevel="debug"`로 변경합니다.

   > **Note**
   >
   > 기본적으로 데이터 채널 열기 및 세션 다시 연결에 대한 로그 항목은 INFO 수준에 기록됩니다. 데이터 흐름(패킷 및 승인) 항목은 DEBUG 수준에서 기록됩니다.

4. 수정하려는 다른 구성 옵션을 변경합니다. 변경 가능한 옵션은 다음과 같습니다.

   - 디버그 수준: 디버그 수준은 `formatid="fmtinfo"`에서 `outputs formatid="fmtdebug"`로 변경할 수 있습니다.

- 로그 파일 옵션: 로그 파일 이름을 제외하고 로그가 저장되는 위치를 포함해 로그 파일 옵션을 변경할 수 있습니다.

  > **Important**
  >
  > 파일 이름을 변경하지 않습니다. 그렇지 않으면 로깅이 제대로 작동하지 않습니다.

```
<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**
  >
  > 로그를 저장하는 데 지정된 기본 디렉터리를 사용하는 경우 sudo를 사용하여 세션 명령을 실행하거나 플러그인이 설치된 디렉터리에 전체 읽기 및 쓰기 권한을 부여해야 합니다. 이러한 제한 사항을 우회하려면 로그가 저장되는 위치를 바꿉니다.

5. 파일을 저장합니다.

세션 시작

AWS Systems Manager콘솔, Amazon Elastic Compute Cloud(Amazon EC2) 콘솔, AWS Command Line Interface(AWS CLI) 또는 SSH를 사용하여 세션을 시작할 수 있습니다.

주제

- 세션 시작(Systems Manager 콘솔)
- 세션 시작(Amazon EC2 콘솔)
- 세션 시작(AWS CLI)
- 세션 시작(SSH)
- 세션 시작(포트 전달)
- 세션 시작(원격 호스트로 포트 전달)
- 세션 시작(대화형 및 비대화형 명령)

세션 시작(Systems Manager 콘솔)

AWS Systems Manager 콘솔을 사용하여 계정의 관리형 노드와 관련된 세션을 시작할 수 있습니다.

> **Note**
>
> 세션을 시작하기 전에 Session Manager에 대한 설정 단계를 완료했는지 확인합니다. 자세한 설명은 Session Manager 설정을 참조하세요.

세션 시작(Systems Manager 콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Session Manager를 선택합니다.
3. 세션 시작을 선택합니다.
4. (선택 사항) 세션 이유 필드에 세션 설명을 입력합니다.
5. 대상 인스턴스의 경우, 연결하려는 관리형 노드 왼쪽에 있는 옵션 버튼을 선택합니다.

   원하는 노드가 목록에 없거나 노드를 선택했는데 구성 오류가 표시되는 경우 관리형 노드를 사용할 수 없거나 Session Manager에 대해 구성되지 않음에서 문제 해결 단계를 참조하세요.
6. 세션을 즉시 시작하려면 세션 시작을 선택합니다.

   -또는-

   세션 옵션을 보려면 다음을 선택합니다.
7. (선택 사항) 세션 문서의 경우 세션이 시작될 때 실행하려는 문서를 선택합니다. 문서에서 런타임 파라미터를 지원하는 경우 각 파라미터 필드에 쉼표로 구분된 값을 하나 이상 입력할 수 있습니다.
8. Next(다음)를 선택합니다.
9. 세션 시작을 선택합니다.

연결된 후 다른 연결 유형을 사용할 때 bash 명령(Linux 및 macOS) 또는 PowerShell 명령(Windows)을 실행할 수 있습니다.

> **Important**
>
> Session Manager 콘솔에서 세션을 시작할 때 사용자가 문서를 지정할 수 있도록 하려면 다음을 참고하세요.

- 사용자에게 해당 IAM 정책의 `ssm:GetDocument` 및 `ssm:ListDocuments` 권한을 부여해야 합니다. 자세한 내용은 콘솔의 사용자 지정 세션 문서 액세스 권한 부여 단원을 참조하십시오.
- 콘솔에서는 `Standard_Stream`으로 정의된 `sessionType`이 있는 세션 문서만 지원합니다. 자세한 내용은 Session 문서 스키마 단원을 참조하십시오.

세션 시작(Amazon EC2 콘솔)

Amazon Elastic Compute Cloud(Amazon EC2) 콘솔을 사용하여 계정의 인스턴스로 세션을 시작할 수 있습니다.

> **Note**
>
> 하나 이상의 Systems Manager 작업(`ssm:`*`command-name`*)을 수행할 권한이 없다는 오류가 수신되면 관리자에게 문의하여 도움을 받아야 합니다. 관리자는 로그인 보안 인증 정보를 제공한 사람입니다. Amazon EC2 콘솔에서 세션을 시작할 수 있도록 정책을 업데이트할 것을 관리자에게 요청합니다. 관리자인 경우 Session Manager에 대한 샘플 IAM 정책 섹션에서 자세한 내용을 참조하세요.

세션을 시작하려면(Amazon EC2 콘솔)

1. https://console.aws.amazon.com/ec2/에서 Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 Instances(인스턴스)를 선택합니다.
3. 인스턴스를 선택한 다음 연결을 선택합니다.
4. 연결 방법에서 Session Manager를 선택합니다.
5. 연결을 선택합니다.

연결된 후 다른 연결 유형을 사용할 때 bash 명령(Linux 및 macOS) 또는 PowerShell 명령(Windows)을 실행할 수 있습니다.

세션 시작(AWS CLI)

아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.

세션을 시작하기 전에 Session Manager에 대한 설정 단계를 완료했는지 확인합니다. 자세한 설명은 Session Manager 설정을 참조하세요.

AWS CLI를 사용하여 세션 명령을 실행하려면 Session Manager 플러그인도 로컬 시스템에 설치해야 합니다. 자세한 설명은 AWS CLI의 Session Manager 플러그인 설치을 참조하세요.

AWS CLI를 사용하여 세션을 시작하려면 다음 명령을 실행하여 *instance-id*를 자신의 정보로 바꿉니다.

```
aws ssm start-session \
    --target instance-id
```

start-session 명령과 함께 사용할 수 있는 다른 옵션에 대한 자세한 내용은 AWS CLI 명령 레퍼런스의 AWS Systems Manager 섹션에 있는 start-session을 참조하세요.

세션 시작(SSH)

Session Manager SSH 세션을 시작하려면 관리형 노드에 SSM Agent 버전 2.3.672.0 이상이 설치되어 있어야 합니다.

SSH 연결 요구 사항

SSH를 사용하는 세션 연결에 있어 다음과 같은 요구 사항 및 제한 사항에 유의합니다.

- SSH 연결을 지원하도록 대상 관리형 노드를 구성해야 합니다. 자세한 내용은 (선택 사항) Session Manager를 통한 SSH 연결에 대한 권한 허용 및 제어를 참조하세요.
- 다른 유형의 세션 연결에 사용되는 `ssm-user` 계정이 아닌 PEM(Privacy Enhanced Mail) 인증서와 연결된 관리형 노드 계정을 사용해서 연결해야 합니다. 예를 들면, Linux 및 macOS용 EC2 인스턴스의 기본 사용자는 `ec2-user`입니다. 각 인스턴스 유형의 기본 사용자를 식별하는 것에 대한 자세한 내용은 Amazon EC2 사용 설명서의 인스턴스에 대한 정보 가져오기를 참조하세요.
- 포트 전달 또는 SSH를 통해 연결하는 Session Manager 세션에는 로깅을 사용할 수 없습니다. SSH는 모든 세션 데이터를 암호화하고 Session Manager는 SSH 연결을 위한 터널 역할만 하기 때문입니다.

> **Note**
>
> 세션을 시작하기 전에 Session Manager에 대한 설정 단계를 완료했는지 확인합니다. 자세한 설명은 Session Manager 설정을 참조하세요.

SSH를 사용하여 세션을 시작하려면 다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
ssh -i /path/my-key-pair.pem username@instance-id
```

> **Tip**
>
> SSH를 사용해 세션을 시작할 때는 다음 명령 형식을 사용해 로컬 파일을 대상 관리형 노드에 복사할 수 있습니다.
>
> ```
> scp -i /path/my-key-pair.pem /path/ExampleFile.txt username@instance-id:~
> ```

start-session 명령과 함께 사용할 수 있는 다른 옵션에 대한 자세한 내용은 AWS CLI 명령 레퍼런스의 AWS Systems Manager 섹션에 있는 start-session를 참조하세요.

세션 시작(포트 전달)

Session Manager 포트 전달 세션을 시작하려면 관리형 노드에 SSM Agent 버전 2.3.672.0 이상이 설치되어 있어야 합니다.

> **Note**
>
> 세션을 시작하기 전에 Session Manager에 대한 설정 단계를 완료했는지 확인합니다. 자세한 설명은 Session Manager 설정을 참조하세요.
> AWS CLI를 사용하여 세션 명령을 실행하려면 Session Manager 플러그인을 로컬 시스템에 설치해야 합니다. 자세한 설명은 AWS CLI의 Session Manager 플러그인 설치을 참조하세요.
> 운영 체제 및 명령줄 도구에 따라 따옴표의 배치가 다를 수 있으며 이스케이프 문자가 필요할 수 있습니다.

포트 전달 세션을 시작하려면 CLI에서 다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

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`는 세션 트래픽을 리디렉션하려는 관리형 노드의 원격 포트입니다. 예를 들면, RDP(원격 데스크톱 프로토콜)를 사용하여 Windows 노드 연결에 3389 포트를 지정할 수도 있습니다. `portNumber` 파라미터를 지정하지 않으면 Session Manager에서는 80을 기본값으로 사용합니다.

`localPortNumber`는 트래픽이 시작되는 로컬 컴퓨터의 포트입니다(예: 56789). 이 값은 클라이언트를 사용하여 관리형 노드에 연결할 때 입력하는 값입니다. 예를 들면 **`localhost:56789`**입니다.

start-session 명령과 함께 사용할 수 있는 다른 옵션에 대한 자세한 내용은 AWS CLI 명령 레퍼런스의 AWS Systems Manager 섹션에 있는 start-session을 참조하세요.

포트 전달 세션에 대한 자세한 내용은 AWS News Blog의 Port Forwarding Using AWS Systems ManagerSession Manager를 참조하세요.

세션 시작(원격 호스트로 포트 전달)

원격 호스트로의 Session Manager 포트 전달 세션을 시작하려면 관리형 노드에 SSM Agent 버전 3.1.1374.0 이상이 설치되어 있어야 합니다. 원격 호스트는 Systems Manager에서 관리할 필요가 없습니다.

> **Note**
>
> 세션을 시작하기 전에 Session Manager에 대한 설정 단계를 완료했는지 확인합니다. 자세한 설명은 Session Manager 설정을 참조하세요.
> AWS CLI를 사용하여 세션 명령을 실행하려면 Session Manager 플러그인을 로컬 시스템에 설치해야 합니다. 자세한 설명은 AWS CLI의 Session Manager 플러그인 설치을 참조하세요.
> 운영 체제 및 명령줄 도구에 따라 따옴표의 배치가 다를 수 있으며 이스케이프 문자가 필요할 수 있습니다.

포트 전달 세션을 시작하려면 AWS CLI에서 다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

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 값은 연결하려는 원격 호스트의 호스트 이름 또는 IP 주소를 나타냅니다. 관리형 노드와 원격 호스트 간의 일반적인 연결 및 이름 확인 요구 사항은 여전히 적용됩니다.

portNumber는 세션 트래픽을 리디렉션하려는 관리형 노드의 원격 포트입니다. 예를 들면, RDP(원격 데스크톱 프로토콜)를 사용하여 Windows 노드 연결에 3389 포트를 지정할 수도 있습니다. portNumber 파라미터를 지정하지 않으면 Session Manager에서는 80을 기본값으로 사용합니다.

localPortNumber는 트래픽이 시작되는 로컬 컴퓨터의 포트입니다(예: 56789). 이 값은 클라이언트를 사용하여 관리형 노드에 연결할 때 입력하는 값입니다. 예를 들면 **localhost:56789**입니다.

start-session 명령과 함께 사용할 수 있는 다른 옵션에 대한 자세한 내용은 AWS CLI 명령 레퍼런스의 AWS Systems Manager 섹션에 있는 start-session를 참조하세요.

Amazon ECS 작업을 사용하여 세션 시작

Session Manager는 Amazon Elastic Container Service(Amazon ECS) 클러스터 내의 작업을 사용하여 포트 전달 세션을 시작하는 것을 지원합니다. 이렇게 하려면 다음 권한을 포함하도록 IAM의 작업 역할을 업데이트해야 합니다.

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

```
        "Action": [
             "ssmmessages:CreateControlChannel",
             "ssmmessages:CreateDataChannel",
             "ssmmessages:OpenControlChannel",
             "ssmmessages:OpenDataChannel"
        ],
       "Resource": "*"
       }
    ]
}
```

Amazon ECS 작업을 사용하여 포트 전달 세션을 시작하려면 AWS CLI에서 다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

> **Note**
>
> `target` 파라미터에서 < and > 기호를 제거합니다. 이러한 기호는 독자의 설명을 위한 용도로만 제공됩니다.

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"
```

세션 시작(대화형 및 비대화형 명령)

세션을 시작하기 전에 Session Manager에 대한 설정 단계를 완료했는지 확인합니다. 자세한 설명은 Session Manager 설정을 참조하세요.

AWS CLI를 사용하여 세션 명령을 실행하려면 Session Manager 플러그인도 로컬 시스템에 설치해야 합니다. 자세한 설명은 AWS CLI의 Session Manager 플러그인 설치을 참조하세요.

대화형 명령 세션을 시작하려면 다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

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"
```

start-session 명령과 함께 사용할 수 있는 다른 옵션에 대한 자세한 내용은 AWS CLI 명령 레퍼런스의 AWS Systems Manager 섹션에 있는 start-session를 참조하세요.

추가 정보

- AWS Systems ManagerSession Manager에서 포트 전달을 사용하여 원격 호스트에 연결
- AWS Systems Manager을 사용한 Amazon EC2 인스턴스 포트 전달
- Session Manager 포트 전달을 사용하여 AWS Managed Microsoft AD 리소스 관리
- AWS News Blog의 Port Forwarding Using AWS Systems ManagerSession Manager.

세션 종료

AWS Systems Manager 콘솔 또는 AWS Command Line Interface(AWS CLI)를 사용하여 계정의 시작한 세션을 종료할 수 있습니다. 20분이 지나도록 사용자 활동이 없으면 세션이 종료됩니다. 세션이 종료되면 다시 시작할 수 없습니다.

주제

- 세션 종료(콘솔)
- 세션 종료(AWS CLI)

세션 종료(콘솔)

AWS Systems Manager 콘솔을 사용하여 계정의 세션을 종료할 수 있습니다.

세션을 종료하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Session Manager를 선택합니다.
3. 세션에서 종료할 세션 왼쪽에 있는 옵션 버튼을 선택합니다.
4. 종료를 선택합니다.

세션 종료(AWS CLI)

AWS CLI를 사용하여 세션을 종료하려면 다음 명령을 실행합니다. *session-id*를 자신의 정보로 바꿉니다.

```
aws ssm terminate-session \
    --session-id session-id
```

terminate-session 명령에 대한 자세한 내용은 AWS CLI 명령 레퍼런스 AWS Systems Manager 섹션의 terminate-session을 참조하세요.

세션 기록 보기

AWS Systems Manager 콘솔이나 AWS Command Line Interface(AWS CLI)를 사용하여 계정의 세션에 대한 정보를 볼 수 있습니다. 콘솔에서 다음과 같은 세션 세부 정보를 확인할 수 있습니다.

- 세션 ID
- 세션을 통해 관리형 노드에 연결된 사용자
- 관리형 노드의 ID
- 세션이 시작 및 종료된 시점
- 세션의 상태
- (설정된 경우) 세션 로그를 저장하도록 지정된 위치

AWS CLI를 사용하여 계정의 세션 목록을 볼 수 있지만 콘솔에서 사용할 수 있는 추가 세부 정보를 확인할 수 없습니다.

세션 기록 정보 로깅에 대한 자세한 내용은 세션 활동 로깅 활성화 및 비활성화 섹션을 참조하세요.

주제

- 세션 기록 보기(콘솔)
- 세션 기록 보기(AWS CLI)

세션 기록 보기(콘솔)

AWS Systems Manager 콘솔을 사용하여 계정의 세션에 대한 세부 정보를 볼 수 있습니다.

세션 기록을 보려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Session Manager를 선택합니다.
3. 세션 기록 탭을 선택합니다.

   -또는-

   Session Manager 홈 페이지가 먼저 열리면 기본 설정 구성, 세션 기록 탭을 차례로 선택합니다.

세션 기록 보기(AWS CLI)

AWS CLI를 사용하여 계정의 세션 목록을 보려면 다음 명령을 실행합니다.

```
aws ssm describe-sessions \
    --state History
```

> **Note**
>
> 이 명령은 Session Manager를 사용하여 시작된 대상에 대한 연결 결과만 반환합니다. RDP(Remote Desktop Protocol) 또는 SSH(Secure Shell Protocol)와 같은 다른 수단을 통해 이루어진 연결은 나열되지 않습니다.

describe-sessions 명령과 함께 사용할 수 있는 다른 옵션에 대한 자세한 내용은 AWS CLI 명령 레퍼런스의 AWS Systems Manager 섹션에 있는 describe-sessions를 참조하세요.

세션 활동 감사

Session Manager는 Systems Manager 콘솔에서 현재 세션과 완료된 세션에 대한 정보를 제공하는 것 외에 AWS CloudTrail을 사용하는 AWS 계정의 세션 활동을 감사하는 기능을 제공합니다.

CloudTrail에서는 Systems Manager 콘솔, AWS Command Line Interface(AWS CLI) 및 Systems Manager SDK를 통해 API 호출을 캡처합니다. CloudTrail 콘솔에서 정보를 보고 지정된 Amazon Simple Storage Service(Amazon S3) 버킷에 저장할 수 있습니다. 계정에 대한 모든 CloudTrail 로그를 저장하는 데 Amazon S3 버킷 하나가 사용됩니다. 자세한 내용은 AWS CloudTrail을 사용하여 AWS Systems ManagerAPI 호출을 로깅 단원을 참조하십시오.

> **Note**
>
> 로그 파일의 반복 분석, 기록 분석 및 이론적 분석을 위해 CloudTrail Lake 또는 유지 관리하는 테이블을 사용하여 CloudTrail 로그를 쿼리하는 방법을 고려합니다. 자세한 내용은 AWS CloudTrail 사용 설명서의 AWS CloudTrail 로그 쿼리를 참조하세요.

Amazon EventBridge를 사용하여 세션 활동 모니터링(콘솔)

EventBridge를 사용하면 AWS 리소스 변경 시 이를 감지하기 위한 규칙을 설정할 수 있습니다. 조직 내 사용자가 세션을 시작 또는 종료하면 이를 감지하는 규칙을 생성하면 예를 들어 Amazon SNS를 통해 해당 이벤트에 대한 알림을 받을 수 있습니다.

Session Manager에 대한 EventBridge 지원은 CloudTrail에서 기록한 API 작업 기록에 의존합니다. CloudTrail과 EventBridge 통합을 사용하여 대부분의 AWS Systems Manager 이벤트에 응답할 수 있습니다. API 호출을 수행하지 않는 `exit` 명령과 같이 세션 내에서 발생하는 작업은 EventBridge에서 감지되지 않습니다.

다음 단계에서는 StartSession과 같은 Session Manager API 이벤트가 발생할 때 Amazon Simple Notification Service(Amazon SNS)를 통해 알림을 시작하는 방법을 간략하게 설명합니다.

Amazon EventBridge를 사용하여 세션 활동을 모니터링하려면(콘솔)

1. 추적하려는 Session Manager 이벤트 발생 시 알림을 보내는 데 사용할 Amazon SNS 주제를 생성합니다.

   자세한 내용은 Amazon Simple Notification Service Developer Guide의 Create a Topic을 참조하세요.

2. 추적하려는 Session Manager 이벤트 유형의 Amazon SNS 대상을 호출할 EventBridge 규칙을 생성합니다.

   규칙을 생성하는 방법에 대한 자세한 내용은 Amazon EventBridge 사용 설명서의 이벤트에 대응하는 Amazon EventBridge 규칙 생성을 참조하세요.

   규칙 생성 단계를 수행할 때 다음과 같이 선택하십시오.

   - AWS service(서비스)에서 Systems Manager를 선택합니다.
   - Event Type(이벤트 유형)에서 AWS API Call through CloudTrail(CloudTrail을 통한 API 호출)을 선택합니다.
   - 특정 작업을 선택한 후 알림을 받고 싶은 Session Manager 명령을 한 번에 하나씩 입력합니다. StartSession, ResumeSession 및 TerminateSession을 선택할 수 있습니다. (EventBridge는 `Get*`, `List*` 및 `Describe*` 명령을 지원하지 않습니다.)
   - 대상 선택에서 SNS 주제를 선택합니다. [주제(Topic)]에서 1단계에서 생성한 Amazon SNS 주제의 이름을 선택합니다.

자세한 내용은 Amazon EventBridge User Guide 및 Amazon Simple Notification Service Getting Started Guide를 참조하세요.

세션 활동 로깅 활성화 및 비활성화

Session Manager는 Systems Manager 콘솔에서 현재 세션 및 완료된 세션에 대한 정보를 제공하는 것 외에 AWS 계정의 세션 활동을 감사 및 로깅하기 위한 옵션을 제공합니다. 따라서 다음을 수행할 수 있습니다.

- 보관을 위해 세션 로그를 생성해 저장합니다.
- 지난 30일 동안 Session Manager를 사용해 관리형 노드에 설정한 모든 연결에 대한 세부 정보를 보여주는 보고서를 생성합니다.
- Amazon Simple Notification Service(Amazon SNS) 알림과 같이 AWS 계정의 세션 활동 알림을 생성합니다.
- 세션 활동의 결과로 AWS 리소스에 대한 다른 작업이 자동으로 시작됩니다(AWS Lambda 기능 실행, AWS CodePipeline 파이프라인 시작 또는 AWS Systems Manager Run Command 문서 실행).

> **Important**
>
> Session Manager에 대해 다음과 같은 요구 사항 및 제한 사항에 유의합니다.

- Session Manager는 세션 기본 설정에 따라 세션 중에 입력한 명령과 출력을 기록합니다. 암호와 같은 민감한 데이터가 세션 로그에 표시되지 않도록 하려면 세션 중에 민감한 데이터를 입력할 때 다음 명령을 사용하는 것이 좋습니다.

  Linux & macOS

  ```
  stty -echo; read passwd; stty echo;
  ```

  Windows

  ```
  $Passwd = Read-Host -AsSecureString
  ```

- Windows Server 2012 이하를 사용하는 경우 로그의 데이터가 최적 상태로 지정되지 않을 수 있습니다. 최적 로그 형식을 위해 Windows Server 2012 R2 이상을 사용하는 것이 좋습니다.
- Linux 또는 macOS 관리형 노드를 사용하는 경우 screen 유틸리티가 설치되어 있어야 합니다. 설치되지 않은 경우 로그 데이터가 잘릴 수 있습니다. Amazon Linux 1, Amazon Linux 2, AL2023, Ubuntu Server에는 기본적으로 화면 유틸리티가 설치됩니다. screen을 수동으로 설치하려면 Linux 버전에 따라 `sudo yum install screen` 또는 `sudo apt-get install screen`을 실행합니다.
- 포트 전달 또는 SSH를 통해 연결하는 Session Manager 세션에는 로깅을 사용할 수 없습니다. SSH는 모든 세션 데이터를 암호화하고 Session Manager는 SSH 연결을 위한 터널 역할만 하기 때문입니다.

세션 데이터 로그에 Amazon S3 또는 Amazon CloudWatch Logs를 사용하는 데 필요한 권한에 대한 자세한 내용은 Session Manager, Amazon S3 및 CloudWatch Logs에 대한 권한이 있는 IAM 역할 생성(콘솔) 섹션을 참조하세요.

Session Manager의 로깅 옵션에 대한 자세한 내용은 다음 주제를 참조하세요.

주제

- Amazon CloudWatch Logs를 사용하여 세션 데이터 스트리밍(콘솔)
- Amazon S3를 사용하여 세션 데이터 로깅(콘솔)
- Amazon CloudWatch Logs를 사용하여 세션 데이터 로깅(콘솔)
- CloudWatch Logs 및 Amazon S3의 Session Manager 활동 로깅 비활성화

Amazon CloudWatch Logs를 사용하여 세션 데이터 스트리밍(콘솔)

세션 데이터 로그의 연속 스트림을 Amazon CloudWatch Logs로 보낼 수 있습니다. 사용자가 세션에서 실행한 명령, 명령을 실행한 사용자의 ID, 세션 데이터가 CloudWatch Logs로 스트리밍되는 시간에 대한 타임스탬프와 같은 필수 세부 정보는 세션 데이터를 스트리밍할 때 포함됩니다. 세션 데이터를 스트리밍할 때 로그는 JSON 형식이므로 기존 로깅 솔루션과 통합할 수 있습니다. 대화형 명령에는 세션 데이터 스트리밍이 지원되지 않습니다.

> **Note**
>
> Windows Server 관리형 노드에서 세션 데이터를 스트리밍하려면 PowerShell 5.1 이상이 설치되어 있어야 합니다. 기본적으로 Windows Server 2016 이상에는 필요한 PowerShell 버전이 설치되어 있습니다. 그러나 Windows Server 2012 및 2012 R2에는 필요한 PowerShell 버전이 기본적으로 설치되어 있지 않습니다. 아직 Windows Server 2012 또는 2012 R2 관리형 노드에서 PowerShell을 업데이트하지 않은 경우 Run Command를 사용하여 업데이트할 수 있습니다. Run Command를 사용하는 PowerShell 업데이트에 대한 자세한 내용은 Run Command를 사용하여 PowerShell 업데이트 섹션을 참조하세요.

> **Important**
>
> Windows Server 관리형 노드에 PowerShell 트랜스크립션(PowerShell Transcription) 정책 설정이 구성되어 있으면 세션 데이터를 스트리밍할 수 없습니다.

Amazon CloudWatch Logs를 사용하여 세션 데이터를 스트리밍하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Session Manager를 선택합니다.
3. 기본 설정 탭을 선택하고 편집을 선택합니다.
4. [CloudWatch 로깅(CloudWatch logging)]에서 [사용(Enable)] 확인란을 선택합니다.
5. [세션 로그 스트리밍] 옵션을 선택합니다.
6. (권장) [암호화된 CloudWatch 로그 그룹만 허용(Allow only encrypted CloudWatch log groups)] 옆의 확인란을 선택합니다. 이 옵션을 설명하면 로그 데이터가 로그 그룹에 지정된 서버 측 암호화 키를 사용하여 암호화됩니다. CloudWatch Logs로 전송되는 로그 데이터를 암호화하지 않으려면 확인란을 선택 취소합니다. 로그 그룹에서 암호화가 허용되지 않는 경우에도 확인란의 선택을 취소해야 합니다.

7. [CloudWatch 로그(CloudWatch logs)]에 대해 세션 로그를 업로드할 AWS 계정의 기존 CloudWatch Logs 로그 그룹을 지정하려면 다음 중 하나를 선택합니다.
   - 세션 로그 데이터를 저장하기 위해 계정에 이미 생성된 로그 그룹의 이름을 텍스트 상자에 입력합니다.
   - [로그 그루 찾아보기(Browse log groups)]: 계정에서 이미 생성한 로그 그룹을 선택하여 세션 로그 데이터를 저장합니다.
8. Save(저장)를 선택합니다.

Amazon S3를 사용하여 세션 데이터 로깅(콘솔)

디버깅 및 문제 해결을 위해 지정된 Amazon Simple Storage Service(Amazon S3) 버킷에 세션 로그 데이터를 저장하도록 선택할 수 있습니다. 기본 옵션은 암호화된 Amazon S3 버킷으로 전송할 로그에 적용됩니다. 암호화는 버킷에 지정한 키인 AWS KMS key 또는 Amazon S3 서버 측 암호화(SSE) 키(AES-256)를 사용하여 수행됩니다.

> **Important**
>
> Secure Sockets Layer(SSL)와 함께 가상 호스팅 방식의 버킷을 사용할 경우, SSL 와일드카드 인증서는 마침표가 포함되지 않은 버킷에만 연결됩니다. 이 문제를 해결하려면 HTTP를 사용하거나, 인증서 확인 로직을 직접 작성합니다. 가상 호스팅 방식의 버킷을 사용할 경우 버킷 이름에 마침표(".")를 사용하지 않는 것이 좋습니다.

Amazon S3 버킷 암호화

로그를 암호화된 Amazon S3 버킷으로 전송하려면 해당 버킷에서 암호화가 허용되어야 합니다. Amazon S3 버킷 암호화에 대한 자세한 내용은 S3 버킷에 대한 Amazon S3 기본 암호화를 참조하세요.

고객 관리형 키

사용자가 스스로 관리하는 KMS 키를 사용해 버킷을 암호화하는 경우 인스턴스에 연결된 IAM 인스턴스 프로파일에는 키를 읽을 수 있는 명시적인 권한이 있어야 합니다. AWS 관리형 키을(를) 사용하는 경우 인스턴스에는 이러한 명시적 권한이 필요하지 않습니다. 인스턴스 프로파일에 키를 사용할 수 있는 액세스 권한 제공에 대한 자세한 내용은 AWS Key Management Service Developer Guide의 Allows Key Users to Use the key를 참조하세요.

다음 단계에 따라 Amazon S3 버킷에 세션 로그를 저장하도록 Session Manager를 구성합니다.

> **Note**
>
> 또한 AWS CLI를 사용하여 세션 데이터가 전송되는 Amazon S3 버킷을 지정 또는 변경할 수 있습니다. 자세한 설명은 Session Manager 기본 설정 업데이트(명령줄)을 참조하세요.

Amazon S3를 사용하여 세션 데이터를 로그하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Session Manager를 선택합니다.
3. 기본 설정 탭을 선택하고 편집을 선택합니다.
4. [S3 로깅(S3 logging)]에서 [사용(Enable)] 확인란을 선택합니다.
5. (권장) [암호화된 S3 버킷만 허용(Allow only encrypted S3 buckets)] 옆의 확인란을 선택합니다. 이 옵션을 설명하면 로그 데이터가 버킷에 지정된 서버 측 암호화 키를 사용하여 암호화됩니다. Amazon S3로 전송되는 로그 데이터를 암호화하지 않으려면 확인란을 선택 취소합니다. S3 버킷에서 암호화가 허용되지 않는 경우에도 확인란의 선택을 취소해야 합니다.
6. S3 버킷 이름에 대해 다음 중 한 가지를 선택합니다.

   > **Note**
   >
   > 가상 호스팅 방식의 버킷을 사용할 경우 버킷 이름에 마침표(".")를 사용하지 않는 것이 좋습니다. Amazon S3 버킷 명명 규칙에 대한 자세한 내용은 Amazon Simple Storage Service 개발자 안내서의 버킷 규제 및 제한을 참조하세요.

   - 목록에서 버킷 이름 선택: 계정에서 이미 생성한 Amazon S3 버킷을 선택하여 세션 로그 데이터를 저장합니다.
   - 텍스트 상자에 버킷 이름 입력: 계정에서 이미 생성한 Amazon S3 버킷의 이름을 입력하여 세션 로그 데이터를 저장합니다.
7. (선택 사항) S3 키 접두사에 대해 선택한 버킷에서 로그를 저장할 기존 폴더 또는 새 폴더의 이름을 입력합니다.
8. Save(저장)를 선택합니다.

Amazon S3 및 Amazon S3 버킷 작업에 대한 자세한 내용은 Amazon Simple Storage Service 사용 설명서 및 Amazon Simple Storage Service 사용 설명서를 참조하세요.

Amazon CloudWatch Logs를 사용하여 세션 데이터 로깅(콘솔)

Amazon CloudWatch Logs를 사용하면 다양한 AWS 서비스의 로그 파일을 모니터링, 저장 및 액세스할 수 있습니다. 디버깅 및 문제 해결을 위해 CloudWatch Logs 로그 그룹으로 세션 로그 데이터를 전송할 수 있습니다. 기본 옵션은 KMS 키를 사용하여 암호화된 로그 데이터를 전송하는 것이지만 암호화 여부에 상관없이 로그 그룹에 데이터를 보낼 수 있습니다.

다음 단계에 따라 세션 종료 시 세션 로그 데이터를 CloudWatch Logs 로그 그룹으로 보내도록 AWS Systems Manager Session Manager를 구성합니다.

> **Note**
>
> 또한 AWS CLI를 사용하여 세션 데이터가 전송되는 CloudWatch Logs 로그 그룹을 지정할 수 있습니다. 자세한 설명은 Session Manager 기본 설정 업데이트(명령줄)을 참조하세요.

Amazon CloudWatch Logs를 사용하여 세션 데이터를 로그하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Session Manager를 선택합니다.
3. 기본 설정 탭을 선택하고 편집을 선택합니다.
4. [CloudWatch 로깅(CloudWatch logging)]에서 [사용(Enable)] 확인란을 선택합니다.
5. [세션 로그 업로드(Upload session logs)] 옵션을 선택합니다.
6. (권장) [암호화된 CloudWatch 로그 그룹만 허용(Allow only encrypted CloudWatch log groups)] 옆의 확인란을 선택합니다. 이 옵션을 설명하면 로그 데이터가 로그 그룹에 지정된 서버 측 암호화 키를 사용하여 암호화됩니다. CloudWatch Logs로 전송되는 로그 데이터를 암호화하지 않으려면 확인란을 선택 취소합니다. 로그 그룹에서 암호화가 허용되지 않는 경우에도 확인란의 선택을 취소해야 합니다.
7. [CloudWatch 로그(CloudWatch logs)]에 대해 세션 로그를 업로드할 AWS 계정의 기존 CloudWatch Logs 로그 그룹을 지정하려면 다음 중 하나를 선택합니다.
   - 목록에서 로그 그룹 선택: 계정에서 이미 생성한 로그 그룹을 선택하여 세션 로그 데이터를 저장합니다.
   - 텍스트 상자에 로그 그룹 이름 입력: 계정에서 이미 생성한 로그 그룹의 이름을 입력하여 세션 로그 데이터를 저장합니다.
8. Save(저장)를 선택합니다.

CloudWatch Logs에 대한 자세한 내용은 Amazon CloudWatch Logs User Guide를 참조하세요.

CloudWatch Logs 및 Amazon S3의 Session Manager 활동 로깅 비활성화

Systems Manager 콘솔 또는 AWS CLI를 사용하여계정의 세션 활동 로깅을 비활성화할 수 있습니다.

세션 활동 로깅을 비활성화하는 방법(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Session Manager를 선택합니다.
3. 기본 설정 탭을 선택하고 편집을 선택합니다.
4. CloudWatch 로깅을 비활성화하려면 CloudWatch 로깅 섹션에서 활성화 확인란 선택을 취소합니다.
5. S3 로깅을 비활성화하려면 S3 로깅 섹션에서 활성화 확인란 선택을 취소합니다.
6. Save(저장)를 선택합니다.

세션 활동 로깅을 비활성화하는 방법(AWS CLI)

AWS CLI를 사용하여 세션 활동 로깅을 비활성화하려면 Session Manager 기본 설정 업데이트(명령줄)의 지침을 따릅니다.

JSON 파일에서 `s3BucketName` 및 `cloudWatchLogGroupName` 입력에 값이 없는지 확인합니다. 예:

```
"inputs": {
        "s3BucketName": "",
        ...
        "cloudWatchLogGroupName": "",
        ...
    }
```

JSON 파일에서 모든 `S3*` 및 `cloudWatch*` 입력을 제거하여 로깅을 비활성화할 수도 있습니다.

Session 문서 스키마

다음 정보는 Session 문서의 스키마 요소에 대해 설명합니다. AWS Systems Manager Session Manager는 Session 문서를 사용하여 표준 세션, 포트 전달 세션 또는 대화형 명령을 실행하는 세션 등 시작할 세션 유형을 결정합니다.

schemaVersion

Session 문서의 스키마 버전입니다. Session 문서는 버전 1.0만 지원합니다.

타입: 문자열

필수 항목 여부: 예

description

Session 문서에 대해 지정하는 설명입니다. 예를 들면 “Session Manager로 포트 전달 세션을 시작하기 위한 문서“입니다.

타입: 문자열

필수 항목 여부: 아니요

sessionType

Session 문서가 설정하는 데 사용되는 세션 유형입니다.

타입: 문자열

필수 항목 여부: 예

유효한 값: `InteractiveCommands` | `NonInteractiveCommands` | `Port` | `Standard_Stream`

inputs

이 Session 문서를 사용하여 설정된 세션에 사용할 세션 기본 설정입니다. 이 요소는 `Standard_Stream` 세션을 생성하는 데 사용되는 Session 문서에 필요합니다.

유형: StringMap

필수 항목 여부: 아니요

s3BucketName

세션 종료 시 세션 로그를 보내려는 Amazon Simple Storage Service(Amazon S3) 버킷입니다.

타입: 문자열

필수 항목 여부: 아니요

s3KeyPrefix

`s3BucketName` 입력에 지정한 Amazon S3 버킷으로 로그를 보낼 때 사용할 접두사입니다. Amazon S3에 저장된 객체에 공유 접두사 사용에 대한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 S3 버킷에서 폴더를 어떻게 사용하나요?를 참조하세요.

타입: 문자열

필수 항목 여부: 아니요

s3EncryptionEnabled

`true`로 설정하면 `s3BucketName` 입력에 지정한 Amazon S3 버킷을 암호화해야 합니다.

타입: 부울

필수 여부: 예

cloudWatchLogGroupName

세션 종료 시 세션 로그를 보내려는 Amazon CloudWatch Logs(CloudWatch Logs) 그룹의 이름입니다.

타입: 문자열

필수 항목 여부: 아니요

cloudWatchEncryptionEnabled

`true`로 설정하면 `cloudWatchLogGroupName` 입력에 지정한 로그 그룹을 암호화해야 합니다.

타입: 부울

필수 여부: 예

cloudWatchStreamingEnabled

`true`로 설정하면 세션 데이터 로그의 연속 스트림이 `cloudWatchLogGroupName` 입력에 지정한 로그 그룹으로 전송됩니다. `false`로 설정하면 세션 로그가 세션 종료 시 `cloudWatchLogGroupName` 입력에 지정한 로그 그룹으로 전송됩니다.

타입: 부울

필수 여부: 예

kmsKeyId

로컬 클라이언트 시스템과 연결하는 Amazon Elastic Compute Cloud(Amazon EC2) 관리형 노드 간의 데이터를 추가로 암호화하는 데 사용할 AWS KMS key의 ID입니다.

타입: 문자열

필수 항목 여부: 아니요

runAsEnabled

`true`로 설정하면 `runAsDefaultUser` 입력에 연결할 관리형 노드에 존재하는 사용자 계정을 지정해야 합니다. 그렇지 않으면 세션이 시작되지 않습니다. 기본적으로 세션은 AWS Systems Manager SSM Agent에서 생성된 `ssm-user` 계정을 사용하여 시작됩니다. Run As 특성은 Linux 관리형 노드에 연결하는 경우에만 지원됩니다.

타입: 부울

필수 여부: 예

runAsDefaultUser

`runAsEnabled` 입력이 `true`로 설정된 경우 Linux 관리형 노드에서 세션을 시작할 사용자 계정의 이름입니다. 이 입력에 대해 지정한 사용자 계정은 연결할 관리형 노드에 있어야 합니다. 그렇지 않으면 세션이 시작되지 않습니다.

타입: 문자열

필수 항목 여부: 아니요

idleSessionTimeout

세션이 종료되기 전에 허용할 비활성 시간입니다. 이 입력은 분 단위로 측정됩니다.

타입: 문자열

유효한 값은 1~60입니다.

필수 항목 여부: 아니요

maxSessionDuration

세션이 종료되기 전에 허용할 최대 시간입니다. 이 입력은 분 단위로 측정됩니다.

타입: 문자열

유효한 값은 1~1,440입니다.

필수 항목 여부: 아니요

shellProfile

셸 기본 설정, 환경 변수, 작업 디렉터리 및 세션 시작 시 여러 명령 실행과 같이 세션 내에서 적용할 운영 체제별로 지정하는 기본 설정입니다.

유형: StringMap

필수 항목 여부: 아니요

windows

Windows 관리형 노드의 세션에 대해 지정하는 쉘 기본 설정, 환경 변수, 작업 디렉터리 및 명령입니다.

타입: 문자열

필수 항목 여부: 아니요

linux

Linux 관리형 노드의 세션에 대해 지정하는 쉘 기본 설정, 환경 변수, 작업 디렉터리 및 명령입니다.

타입: 문자열

필수 항목 여부: 아니요

parameters

문서가 허용하는 파라미터를 정의하는 객체입니다. 문서 파라미터 정의에 대한 자세한 내용은 최상위 데이터 요소의 파라미터를 참조하세요. 자주 참조하는 파라미터의 경우에는 먼저 Systems Manager Parameter Store에 저장한 후 참조하는 것이 좋습니다. 이 문서 섹션에서는 `String` 및 `StringList` Parameter Store 파라미터를 참조할 수 있습니다. 이 문서 섹션에서는 `SecureString` Parameter Store 파라미터를 참조할 수 없습니다. 다음 형식을 사용하여 Parameter Store 파라미터를 참조할 수 있습니다.

```
{{ssm:parameter-name}}
```

Parameter Store에 대한 자세한 내용은 AWS Systems Manager Parameter Store 섹션을 참조하세요.

유형: StringMap

필수 항목 여부: 아니요

properties

`StartSession` API 작업에 사용되는 값을 지정하는 객체입니다.

`InteractiveCommands` 세션에 사용되는 Session 문서의 경우 속성 객체에는 지정한 운영 체제에서 실행할 명령이 포함됩니다. `runAsElevated` 부울 속성을 사용하여 명령이 `root`로 실행되는지도 판단할 수 있습니다. 자세한 내용은 세션의 명령에 대한 액세스 제한을 참조하세요.

`Port` 세션에 사용되는 Session 문서의 경우 속성 객체에는 트래픽이 리디렉션되어야 하는 포트 번호가 포함됩니다. 예를 들어 이 주제 뒷부분의 `Port` 유형 Session 문서 예를 참조하세요.

유형: StringMap

필수 항목 여부: 아니요

`Standard_Stream` 유형 Session 문서 예

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` 유형 Session 문서 예

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 유형 Session 문서 예

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"
    }
}
```

특수 문자가 있는 Session 문서 예

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 문제 해결

다음 정보를 사용하면 AWS Systems Manager Session Manager 관련 문제를 해결하는 데 도움이 됩니다.

주제

- Session Manager는 Amazon EC2 콘솔에서 연결할 수 없습니다.
- 세션을 시작할 권한 없음
- 세션 기본 설정을 변경할 권한 없음
- 관리형 노드를 사용할 수 없거나 Session Manager에 대해 구성되지 않음
- Session Manager 플러그인을 찾을 수 없음
- 명령줄 경로에 자동으로 추가되지 않는 Session Manager 플러그인(Windows)
- Session Manager 플러그 인이 응답하지 않음
- TargetNotConnected
- 세션 시작 후 빈 화면 표시
- 장기 실행 세션 동안 관리형 노드가 응답하지 않음

- StartSession 작업을 호출할 때 오류가 발생합니다(InvalidDocument)

Session Manager는 Amazon EC2 콘솔에서 연결할 수 없습니다.

문제: 새 인스턴스를 생성한 후 Amazon Elastic Compute Cloud(Amazon EC2) 콘솔의 Session Manager 탭에 연결 옵션이 표시되지 않습니다.

솔루션 A: 인스턴스 프로파일 생성: 아직 그렇게(EC2 콘솔의 Session Manager 탭에 있는 정보의 지침대로) 하지 않았다면 Quick Setup을 사용하여 AWS Identity and Access Management(IAM) 인스턴스 프로파일을 생성합니다. Quick Setup은 AWS Systems Manager의 기능입니다.

인스턴스에 연결하려면 IAM 인스턴스 프로파일이 Session Manager에 필요합니다. Quick Setup으로 호스트 관리 구성을 생성하면 인스턴스 프로파일을 생성하여 인스턴스에 할당할 수 있습니다. 호스트 관리 구성에서는 필요한 권한이 있는 인스턴스 프로파일을 생성하여 인스턴스에 할당합니다. 또한 호스트 관리 구성에서는 다른 Systems Manager 기능을 활성화하고 해당 기능을 실행할 IAM 역할을 생성합니다. 호스트 관리 구성을 통해 활성화된 Quick Setup 또는 기능을 사용하는 요금은 없습니다. Quick Setup을 열고 호스트 관리 구성을 생성합니다.

> **Important**
>
> 호스트 관리 구성을 생성한 후 Amazon EC2에서 변경 사항을 등록하고 Session Manager 탭을 새로 고치는 데 몇 분 정도 걸릴 수 있습니다. 2분 후에도 탭에 연결 버튼이 표시되지 않으면 인스턴스를 재부팅 해보세요. 재부팅 후에도 연결 옵션이 표시되지 않으면 빠른 설정을 열고 호스트 관리 구성이 하나뿐인지 확인합니다. 구성이 2개라면 오래된 구성을 삭제하고 몇 분 정도 기다립니다.

호스트 관리 구성을 생성한 후에도 여전히 연결할 수 없거나 오류(SSM Agent에 대한 오류 포함)가 표시되는 경우 다음과 같은 솔루션 중 하나를 참조하세요.

- 솔루션 방법 B: 오류가 없으나 여전히 연결할 수 없음
- 솔루션 C: 누락 SSM Agent에 대한 오류

솔루션 방법 B: 오류가 없으나 여전히 연결할 수 없음

호스트 관리 구성을 생성하고 연결을 시도하기 전에 몇 분 정도 기다렸는데도 연결할 수 없으면 호스트 관리 구성을 인스턴스에 수동으로 적용해야 할 수도 있습니다. 다음 절차를 사용하여 Quick Setup 호스트 관리 구성을 업데이트하고 인스턴스에 변경 사항을 적용합니다.

Quick Setup을 사용하여 호스트 관리 구성을 업데이트하는 방법

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Quick Setup를 선택합니다.
3. 생성한 호스트 관리 구성을 구성 목록에서 선택합니다.
4. 작업을 선택한 다음에 구성 편집을 선택합니다.
5. 대상 섹션에서 수동을 선택합니다.
6. 생성한 인스턴스를 인스턴스 섹션에서 선택합니다.
7. 업데이트를 선택합니다.

EC2에서 Session Manager 탭 새로 고침이 진행되는 동안 몇 분 정도 기다립니다. 그래도 연결할 수 없거나 오류가 표시되는 경우 이 문제에 대한 나머지 솔루션을 검토합니다.

솔루션 C: 누락 SSM Agent에 대한 오류

Quick Setup을 사용하여 호스트 관리 구성을 생성할 수 없거나 설치되지 않은 SSM Agent에 대한 오류가 표시되면 인스턴스에 수동으로 SSM Agent를 설치해야 할 수 있습니다. SSM Agent는 Session Manager를 사용하여 인스턴스에 연결하도록 Systems Manager를 활성화하는 Amazon 소프트웨어입니다. SSM Agent는 대부분의 Amazon Machine Image(AMI)에 기본적으로 설치되어 있습니다. 비표준 AMI 또는 오래된 AMI에서 인스턴스를 생성한 경우 에이전트를 수동으로 설치해야 할 수도 있습니다. SSM Agent 설치 절차는 인스턴스 운영 체제에 해당하는 다음 주제를 참조하세요.

- Windows Server
- macOS
- AlmaLinux
- Amazon Linux 1
- Amazon Linux 2 및 AL2023
- CentOS
- CentOS Stream
- Debian Server
- Oracle Linux
- Red Hat Enterprise Linux
- Rocky Linux
- SUSE Linux Enterprise Server

- Ubuntu Server

SSM Agent 관련 문제는 SSM Agent 문제 해결를 참조하세요.

세션을 시작할 권한 없음

문제: 세션을 시작하려고 했지만 필요한 권한이 없다고 표시됩니다.

- 해결 방법: 시스템 관리자가 Session Manager 세션을 시작할 수 있는 AWS Identity and Access Management(IAM) 정책 권한을 부여하지 않았습니다. 자세한 내용은 인스턴스에 대한 사용자 세션 액세스 제어를 참조하세요.

세션 기본 설정을 변경할 권한 없음

문제: 조직에 대한 전역 세션 기본 설정을 업데이트하려고 했는데 필요한 권한이 없다는 메시지가 표시됩니다.

- 해결 방법: 시스템 관리자가 Session Manager 기본 설정을 지정할 수 있는 IAM 정책 권한을 부여하지 않았습니다. 자세한 설명은 Session Manager 기본 설정을 업데이트하도록 사용자 권한 부여 또는 거부을 참조하세요.

관리형 노드를 사용할 수 없거나 Session Manager에 대해 구성되지 않음

문제 1: 세션 시작(Start a session) 콘솔 페이지에서 세션을 시작하려고 했지만 관리형 노드가 목록에 없습니다.

- 솔루션 A: 연결하려는 관리형 노드가 AWS Systems Manager에서 구성되지 않았을 수 있습니다. 자세한 내용은 AWS Systems Manager 설정 단원을 참조하십시오.

  > **Note**
  >
  > IAM 인스턴스 프로파일을 연결할 때 AWS Systems Manager SSM Agent가 이미 실행 중이면 세션 시작(Start a session) 콘솔 페이지에 관리형 노드가 나열되기 전에 에이전트를 다시 시작해야 할 수 있습니다.

- 해결 방법 B: 관리형 노드의 SSM Agent에 적용한 프록시 구성이 올바르지 않을 수 있습니다. 프록시 구성이 올바르지 않으면 관리형 노드가 필요한 서비스 엔드포인트에 연결할 수 없거나 노드가 Systems Manager에 다른 운영 체제로 보고할 수 있습니다. 자세한 내용은 SSM Agent를 구성하여

Linux 노드에 프록시 사용 및 Windows Server 인스턴스에 프록시를 사용하도록 SSM Agent 구성 단원을 참조하세요.

문제 2: 연결하려는 관리형 노드가 세션 시작(Start a session) 콘솔 페이지의 목록에 있지만 이 페이지에 "선택한 인스턴스가 Session Manager를 사용하도록 구성되지 않았습니다.(The instance you selected isn't configured to use Session Manager.)"라는 메시지가 표시됩니다.

- 해결 방법 A: 관리형 노드가 Systems Manager 서비스에서 사용하도록 구성되어 있지만 노드에 연결된 IAM 인스턴스 프로파일에 Session Manager 기능에 대한 권한이 없을 수 있습니다. 자세한 내용은 Session Manager 권한이 있는 IAM 인스턴스 프로파일 확인 또는 생성(Verify or Create an IAM Instance Profile with Session Manager Permissions)을 참조하세요.
- 해결 방법 B: 관리형 노드가 Session Manager를 지원하는 SSM Agent 버전을 실행하지 않습니다. 노드에 대한 SSM Agent를 버전 2.3.68.0 이상으로 업데이트합니다.

  운영 체제에 따라 SSM Agent용 EC2 인스턴스에 수동으로 Windows Server 설치 및 제거, Linux용 EC2 인스턴스에 수동으로 SSM Agent 설치 및 제거 또는 SSM Agent용 EC2 인스턴스에 수동으로 macOS 설치 및 제거의 단계를 수행하여 관리형 노드에 SSM Agent를 업데이트합니다.

  또는 Run Command 문서 `AWS-UpdateSSMAgent`를 사용하여 한 번에 하나 이상의 관리형 노드에서 에이전트 버전을 업데이트하세요. 자세한 설명은 Run Command를 사용하여 SSM Agent 업데이트을 참조하세요.

  > **Tip**
  >
  > 에이전트를 항상 최신 상태로 유지하려면 다음 방법 중 하나를 사용하여 정의한 자동화된 일정에 따라 SSM Agent를 최신 버전으로 업데이트하는 것이 좋습니다.
  >
  > - State Manager 연결의 일부로 `AWS-UpdateSSMAgent`를 실행합니다. 자세한 설명은 시연: SSM Agent(CLI) 자동 업데이트을 참조하세요.
  > - `AWS-UpdateSSMAgent`를 유지 관리 기간의 일부로 실행합니다. 유지 관리 기간 작업에 대한 자세한 내용은 유지 관리 기간 작업(콘솔) 및 자습서: 유지 관리 기간 생성 및 구성(AWS CLI) 섹션을 참조하세요.

- 해결 방법 C: 관리형 노드가 필수 서비스 엔드포인트에 연결할 수 없습니다. AWS PrivateLink에서 제공하는 인터페이스 엔드포인트를 사용하여 Systems Manager 엔드포인트에 연결하여 관리형 노드의 보안 태세를 개선할 수 있습니다. 인터페이스 엔드포인트 사용의 대체 방법은 관리형 노드에서 아웃바운드 인터넷 액세스를 허용하는 것입니다. 자세한 내용은 PrivateLink를 사용하여 Session Manager에 대한 VPC 엔드포인트 설정을 참조하세요.

- 솔루션 D: 관리형 노드에 사용 가능한 CPU 또는 메모리 리소스가 제한되어 있습니다. 관리형 노드가 작동할 수는 있지만 노드에 사용 가능한 리소스가 충분하지 않은 경우 세션을 설정할 수 없습니다. 자세한 내용은 연결할 수 없는 인스턴스 문제 해결을 참조하십시오.

Session Manager 플러그인을 찾을 수 없음

AWS CLI를 사용하여 세션 명령을 실행하려면 Session Manager 플러그인도 로컬 시스템에 설치해야 합니다. 자세한 설명은 AWS CLI의 Session Manager 플러그인 설치을 참조하세요.

명령줄 경로에 자동으로 추가되지 않는 Session Manager 플러그인(Windows)

Windows에 Session Manager 플러그인을 설치하면 `session-manager-plugin` 실행 파일이 운영 체제의 PATH 환경 변수에 자동으로 추가되어야 합니다. Session Manager 플러그인이 제대로 설치되었는지 확인하기 위해 이 환경 변수를 실행한 후 명령에 실패하면(`aws ssm start-session --target` *`instance-id`*) 다음 절차를 수행하여 수동으로 설정해야 할 수 있습니다.

PATH 변수(Windows)를 수정하려면

1. Windows 키를 누르고 **`environment variables`**를 입력합니다.
2. 계정의 환경 변수 편집을 선택합니다.
3. PATH를 선택한 후 편집을 선택합니다.
4. 다음 예제에 표시된 대로 세미콜론으로 구분하여 경로를 변수 값 필드에 추가합니다. *`C:\existing\path`*;***`C:\new\path`***

   다음 예제들에 표시된 대로*`C:\existing\path`*는 필드에 이미 있는 값을 나타내고, ***`C:\new\path`***는 추가하려는 경로를 나타냅니다.

   - 64비트 시스템: `C:\Program Files\Amazon\SessionManagerPlugin\bin\`
   - 32비트 시스템: `C:\Program Files (x86)\Amazon\SessionManagerPlugin\bin\`
5. 확인을 두 번 선택하여 새 설정을 적용합니다.
6. 실행 중인 명령 프롬프트를 모두 닫았다가 다시 엽니다.

Session Manager 플러그 인이 응답하지 않음

포트 전달 세션 중에 로컬 컴퓨터에 바이러스 백신 소프트웨어가 설치되어 있는 경우 트래픽 전달을 중지할 수 있습니다. 어떤 경우에는 Session Manager 플러그 인이 포함된 바이러스 백신 소프트웨어

가 프로세스 교착 상태를 일으킵니다. 이 문제를 해결하려면 바이러스 백신 소프트웨어에서 Session Manager 플러그 인을 허용하거나 제외합니다. Session Manager 플러그 인의 기본 설치 경로에 대한 자세한 내용은 AWS CLI의 Session Manager 플러그인 설치 섹션을 참조하세요.

TargetNotConnected

문제: 세션을 시작하려고 하지만 시스템에서 “StartSession 작업을 호출할 때 (targetNotConnected) 오류가 발생했습니다. *InstanceID*가 연결되어 있지 않습니다.(An error occurred (TargetNotConnected) when calling the StartSession operation: InstanceID isn't connected.)” 오류 메시지를 반환합니다.

- 해결 방법 A: 세션에 대해 지정된 대상 관리형 노드가 세션 관리자에서 사용하도록 완전히 구성되지 않은 경우 이 오류가 반환됩니다. 자세한 설명은 Session Manager 설정을 참조하세요.
- 해결 방법 B: 서로 다른 AWS 계정 또는 AWS 리전에 위치한 관리형 노드에서 세션을 시작하려고 시도하는 경우에도 이 오류가 반환됩니다.

세션 시작 후 빈 화면 표시

문제: 세션을 시작하면 Session Manager에 빈 화면이 표시됩니다.

- 해결 방법 A: 이 문제는 관리형 노드의 루트 볼륨이 가득 찬 경우 발생할 수 있습니다. 디스크 공간 부족으로 인해 노드에서 SSM Agent 작동이 중지됩니다. 이 문제를 해결하려면 Amazon CloudWatch를 사용하여 운영 체제의 지표 및 로그를 수집합니다. 자세한 내용은 Amazon CloudWatch 사용 설명서의 CloudWatch 에이전트를 사용하여 지표, 로그 및 추적 수집을 참조하세요.
- 해결 방법 B: 일치하지 않는 엔드포인트 및 리전 페어가 포함된 링크를 사용하여 콘솔에 액세스한 경우 빈 화면이 표시될 수 있습니다. 예를 들어 다음 콘솔 URL에서 `us-west-2`는 지정된 엔드포인트이지만 `us-west-1`이 지정된 AWS 리전입니다.

```
https://us-west-2.console.aws.amazon.com/systems-manager/session-manager/sessions?
region=us-west-1
```

- 해결 방법 C: 관리형 노드가 VPC 엔드포인트를 사용하여 Systems Manager에 연결하고 있고 Session Manager 기본 설정에서 세션 출력을 Amazon S3 버킷 또는 Amazon CloudWatch Logs 로그 그룹에 기록하지만 `s3` 게이트웨이 엔드포인트 또는 `logs` 인터페이스 엔드포인트는 VPC에 존재하지 않습니다. 관리형 노드가 VPC 엔드포인트를 사용하여 Systems Manager에 연결하고 Session Manager 기본 설정이 Amazon S3 버킷에 세션 출력을 쓰는 경우 **`com.amazonaws.`*`region`*`.s3`** 형식의 `s3` 엔드포인트가 필요합니다. 대신 관리형 노드가 VPC 엔드포인트를 사용하여 Systems Manager에 연결하고 Session Manager 기본 설정이 CloudWatch Logs 로그 그룹에 세션 출력을 쓰

는 경우 **com.amazonaws.*region*.logs** 형식의 logs 엔드포인트가 필요합니다. 자세한 내용은 Systems Manager용 VPC 엔드포인트 생성 단원을 참조하십시오.

- 해결 방법 D: 세션 기본 설정에서 지정한 로그 그룹 또는 Amazon S3 버킷이 삭제되었습니다. 이 문제를 해결하려면 유효한 로그 그룹 또는 S3 버킷으로 세션 기본 설정을 업데이트합니다.
- 해결 방법 E: 세션 기본 설정에서 지정한 로그 그룹 또는 Amazon S3 버킷은 암호화되지 않았지만 `cloudWatchEncryptionEnabled` 또는 `s3EncryptionEnabled` 입력을 `true`로 설정했습니다. 이 문제를 해결하려면 암호화된 로그 그룹 또는 Amazon S3 버킷으로 세션 기본 설정을 업데이트하거나 `cloudWatchEncryptionEnabled` 또는 `s3EncryptionEnabled` 입력을 `false`로 설정합니다. 이 시나리오는 명령줄 도구를 사용하여 세션 기본 설정을 생성하는 고객에게만 적용됩니다.

장기 실행 세션 동안 관리형 노드가 응답하지 않음

문제: 장기 실행 세션 중에 관리형 노드가 응답하지 않거나 충돌합니다.

해결 방법: Session Manager에 대한 SSM Agent 로그 보존 기간을 줄입니다.

세션에 대한 SSM Agent 로그 보존 기간을 줄이려면

1. Linux의 `/etc/amazon/ssm/` 디렉터리 또는 Windows의 `C:\Program Files\Amazon\SSM`에서 `amazon-ssm-agent.json.template`을 찾습니다.
2. `amazon-ssm-agent.json.template`의 내용을 동일한 디렉터리의 `amazon-ssm-agent.json`이라는 새 파일에 복사합니다.
3. `SSM` 속성에서 `SessionLogsRetentionDurationHours` 값의 기본값을 줄이고 파일을 저장합니다.
4. SSM Agent를 다시 시작합니다.

StartSession 작업을 호출할 때 오류가 발생합니다(InvalidDocument)

문제: AWS CLI을(를) 사용하여 세션을 시작할 때 다음 오류가 발생합니다.

```
An error occurred (InvalidDocument) when calling the StartSession operation: Document
 type: 'Command' is not supported. Only type: 'Session' is supported for Session
 Manager.
```

해결 방법: `--document-name` 파라미터에 지정한 SSM 문서는 세션 문서가 아닙니다. 다음 절차에 따라 AWS Management Console에서 세션 문서의 목록을 봅니다.

세션 문서 목록 보기

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. 범주 목록에서 세션 문서를 선택합니다.

AWS Systems Manager Run Command

AWS Systems Manager의 기능인 Run Command를 사용하여 관리형 노드의 구성을 원격으로 안전하게 관리할 수 있습니다. 관리형 노드는 Systems Manager용으로 구성된 하이브리드 및 멀티클라우드 환경의 모든 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 또는 비 EC2 시스템입니다. Run Command를 사용하면 일반적인 관리 작업을 자동화하고 대규모로 일회성 구성 변경을 수행할 수 있습니다. AWS Management Console, AWS Command Line Interface(AWS CLI), AWS Tools for Windows PowerShell 또는 AWS SDK에서 Run Command를 사용할 수 있습니다. Run Command는 무료로 제공됩니다. Run Command를 시작하려면 Systems Manager 콘솔을 엽니다. 탐색 창에서 Run Command를 선택합니다.

관리자는 Run Command를 사용하여 부트스트랩 애플리케이션 설치, 배포 파이프라인 구축, Auto Scaling 그룹에서 인스턴스가 제거될 때 로그 파일 캡처, 인스턴스를 Windows 도메인에 조인합니다.

시작하기

아래 표에는 Run Command를 처음 사용하는 데 도움이 되는 정보가 나와 있습니다.

주제	Details
AWS Systems Manager 설정	하이브리드 및 멀티클라우드 환경의 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 및 비 EC2 시스템에 대한 설정 요구 사항을 완료했는지 확인합니다.
하이브리드 및 멀티클라우드 환경에서 Systems Manager 사용하기	(옵션) Run Command를 사용하여 관리할 수 있도록 온프레미스 서버와 VM을 AWS에 등록합니다.
the section called “Systems Manager를 통한 엣지 디바이스 관리”	(선택 사항) Run Command를 사용하여 관리할 수 있도록 엣지 디바이스를 구성합니다.

주제	Details
관리형 노드에서 명령 실행	AWS Management Console을 사용하여 하나 이상의 관리되는 노드를 대상으로 하는 명령을 실행하는 방법에 대해 알아봅니다.
Run Command 연습	Tools for Windows PowerShell 또는 AWS CLI를 사용하여 명령을 실행하는 방법에 대해 알아봅니다.

EventBridge 지원

이 Systems Manager 기능은 Amazon EventBridge 규칙에서 이벤트 유형과 대상 유형으로 모두 지원됩니다. 자세한 내용은 Amazon EventBridge로 Systems Manager 이벤트 모니터링 및 참조: Systems Manager용 Amazon EventBridge 이벤트 패턴 및 유형 섹션을 참조하세요.

추가 정보

- EC2 인스턴스에서 원격으로 Run Command 실행(10분 자습서)
- Amazon Web Services 일반 참조의 Systems Manager 서비스 할당량
- AWS Systems Manager API 참조

주제

Run Command 설정

AWS Systems Manager의 기능인 Run Command를 사용하여 노드를 관리하려면 먼저 명령을 실행할 사용자에 대해 AWS Identity and Access Management(IAM) 정책을 구성합니다.

Systems Manager에 대한 노드를 구성해야 합니다. 자세한 내용은 AWS Systems Manager 설정 단원을 참조하십시오.

또한 다음과 같은 선택적 설정 작업을 완료하여 관리형 노드의 보안 상태 및 일상적인 관리를 최소화하는 것이 좋습니다.

Amazon EventBridge를 사용하여 명령 실행 모니터링

EventBridge를 사용하여 명령 실행 상태 변경 사항을 기록할 수 있습니다. 상태가 달라질 때마다 또는 관심이 있는 상태로 변경될 때 실행되는 규칙을 만들면 됩니다. EventBridge 이벤트가 발생할 때 Run Command를 대상 작업으로 지정할 수도 있습니다. 자세한 내용은 Systems Manager 이벤트에 대해 EventBridge 구성 단원을 참조하십시오.

Amazon CloudWatch Logs를 사용하여 명령 실행 모니터링

모든 명령 출력 및 오류 로그를 정기적으로 Amazon CloudWatch 로그 그룹에 보내도록 Run Command를 구성할 수 있습니다. 거의 실시간으로 이러한 출력 로그를 모니터링하고, 특정 구문, 값 또는 패턴을 검색하며, 검색을 기반으로 경보를 생성할 수 있습니다. 자세한 내용은 Run Command에 대한 Amazon CloudWatch Logs 구성 단원을 참조하십시오.

특정 관리형 노드 Run Command 액세스 제한

AWS Identity and Access Management(IAM)을 사용하여 관리되는 노드에서 명령을 실행하는 사용자의 기능을 제한할 수 있습니다. 특히 사용자가 특정 태그에 지정된 관리형 노드에서만 명령을 실행할 수 있도록 하는 조건이 포함된 IAM 정책을 생성할 수 있습니다. 자세한 내용은 태그를 기반으로 Run Command 액세스 제한 단원을 참조하십시오.

태그를 기반으로 Run Command 액세스 제한

이 섹션에서는 IAM 정책에서 태그 조건을 지정하여 관리형 노드에서 명령을 실행하는 사용자의 기능을 제한하는 방법에 대해 설명합니다. 관리형 노드에는 Systems Manager용으로 구성된 하이브리드 및 멀티클라우드 환경의 Amazon EC2 인스턴스 및 비 EC2 노드가 포함됩니다. 정보가 명시적으로 표시되지 않지만 관리형 AWS IoT Greengrass 코어 디바이스에 대한 액세스를 제한할 수도 있습니다. 시작하려면 AWS IoT Greengrass 디바이스에 태그를 지정해야 합니다. 자세한 내용은 AWS IoT Greengrass Version 2 개발자 안내서에서 AWS IoT Greengrass Version 2 리소스 태그하기 섹션을 참조하세요.

사용자가 특정 태그에 지정된 관리형 노드에 대해서만 명령을 실행할 수 있도록 하는 조건이 포함된 IAM 정책을 생성하여 명령 실행을 특정 노드로 제한할 수 있습니다. 다음 예에서 사용자는 노드가 Finance WebServer(`ssm:resourceTag/Finance: WebServer`)인 상태에서 노드(`Resource: arn:aws:ec2:*:*:instance/*`)의 SSM 문서(`Resource: arn:aws:ssm:*:*:document/*`)를

통해 Run Command(Effect: Allow, Action: ssm:SendCommand)를 사용할 수 있습니다. 사용자가 태그 지정되지 않았거나 Finance: WebServer 이외의 태그가 있는 노드에 명령을 보내는 경우 실행 결과는 AccessDenied로 표시됩니다.

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

사용자가 여러 태그로 지정된 관리형 노드에 대해 명령을 실행하도록 허용하는 IAM 정책을 생성할 수 있습니다. 다음 정책은 두 태그를 포함하는 관리형 노드에서 사용자가 명령을 실행하도록 허용합니다. 사용자가 두 태그 모두에 지정되지 않은 노드에 명령을 전송할 경우 실행 결과가 AccessDenied로 표시됩니다.

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

사용자가 여러 태그로 지정된 관리형 노드 그룹에 대해 명령을 실행하도록 허용하는 IAM 정책을 생성할 수도 있습니다. 다음 정책 예시는 태그 지정된 노드 그룹 중 하나 또는 두 그룹 모두에 대해 명령을 실행하도록 허용합니다.

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

IAM 정책 생성에 대한 자세한 내용은 IAM 사용 설명서의 관리형 정책과 인라인 정책을 참조하세요. 관리형 노드 태그 지정에 대한 자세한 내용은 AWS Resource Groups 사용 설명서의 태그 에디터를 참조하세요.

관리형 노드에서 명령 실행

이 섹션에는 AWS Systems Manager 콘솔에서 관리형 노드로 명령을 전송하는 방법에 대한 정보가 포함되어 있습니다. 이 단원에는 명령을 취소하는 방법에 대한 정보도 나와 있습니다.

Windows PowerShell을 사용하여 명령을 보내는 방법에 대한 자세한 내용은 AWS Tools for PowerShell Cmdlet Reference의 AWS Systems Manager 섹션에 있는 연습: Run Command에서 AWS Tools for Windows PowerShell 사용 또는 예를 참조하세요. AWS Command Line Interface(AWS CLI)를 사용하여 명령을 보내는 방법에 대한 자세한 내용은 SSM CLI Reference의 예 또는 연습: Run Command에서 AWS CLI 사용를 참조하세요.

> **Important**
>
> Run Command를 사용해 명령을 보낼 때 암호, 구성 데이터 또는 기타 보안 암호와 같이 민감한 정보는 일반 텍스트 형식으로 포함하지 않습니다. 계정의 모든 Systems Manager API 활동은 AWS CloudTrail 로그를 위해 S3 버킷에 기록됩니다. 즉, 해당 S3 버킷에 대한 액세스 권한이 있는 사용자는 그러한 보안 암호의 일반 텍스트 값을 볼 수 있습니다. 따라서 `SecureString` 파라미터를 생성하고 사용하여 Systems Manager 작업에 사용하는 민감한 데이터를 암호화하는 것이 좋습니다.
> 자세한 내용은 IAM 정책을 사용하여 Systems Manager 파라미터에 대한 액세스 제한 단원을 참조하십시오.

목차

콘솔에서 명령 실행

AWS Management Console에서 AWS Systems Manager의 기능인 Run Command를 사용하여 로그인하지 않고도 관리형 노드를 구성할 수 있습니다. 이 주제에는 Run Command를 사용하여 관리형 노드에서 SSM Agent를 업데이트하는 방법을 보여주는 예가 포함되어 있습니다.

시작하기 전 준비 사항

Run Command를 사용하여 명령을 보내기 전에 관리형 노드에서 모든 Systems Manager 설정 요구 사항을 충족하는지 확인합니다.

Run Command를 사용하여 명령을 전송하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택합니다.
3. Run command(Run 명령)를 선택합니다.
4. Command 문서(Command document) 목록에서 Systems Manager 문서를 선택합니다.
5. 명령 파라미터 섹션에서 필요한 파라미터의 값을 지정합니다.
6. Targets(대상) 섹션에서, 태그를 지정하거나, 수동으로 인스턴스나 엣지 디바이스를 선택하거나, 리소스 그룹을 지정하여 이 작업을 실행할 관리형 노드를 식별합니다.

   > **Tip**
   >
   > 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

7. Other parameters(다른 파라미터):
   - Comment(설명)에 명령에 대한 정보를 입력합니다.
   - 제한 시간(초)에서 전체 명령 실행이 실패할 때까지 시스템이 기다리는 시간을 초 단위로 지정합니다.

8. Rate control(속도 제어)에서

   - Concurrency(동시성)에서 명령을 동시에 실행할 관리형 노드의 백분율 또는 개수를 지정합니다.

     > **Note**
     >
     > 관리형 노드에 적용할 태그를 지정하거나, AWS 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상 수를 제한합니다.

   - Error threshold(오류 임계값)에서, 명령이 노드의 개수 또는 백분율에서 실패한 후 다른 관리형 노드에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째 오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.

9. (선택 사항) 모니터링을 위해 명령에 적용할 CloudWatch 경보를 선택합니다. CloudWatch 경보를 명령에 연결하려면 명령을 실행하는 IAM 보안 주체에 `iam:createServiceLinkedRole` 작업에 대한 권한이 있어야 합니다. CloudWatch 경보에 대한 자세한 내용은 Amazon CloudWatch 경보 사용을 참조하세요. 경보가 활성화되면 보류 중인 명령 호출이 실행되지 않습니다.

10. (선택 사항) Output options(출력 옵션)에서 명령 출력을 파일에 저장하려면 Write command output to an S3 bucket(S3 버킷에 명령 출력 쓰기) 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

    > **Note**
    >
    > 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아니라 인스턴스에 할당된 인스턴스 프로파일(EC2 인스턴스용) 또는 IAM 서비스 역할(하이브리드 정품 인증 시스템)의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

11. SNS notifications(SNS 알림) 섹션에서, 명령 실행 상태에 대한 알림이 전송되도록 하려면 Enable SNS notifications(SNS 알림 활성화) 확인란을 선택합니다.

    Run Command에 대한 Amazon SNS 알림 구성에 대한 자세한 내용은 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 섹션을 참조하세요.

12. Run(실행)을 선택합니다.

명령을 취소하는 방법에 대한 자세한 내용은 the section called “명령 취소” 섹션을 참조하세요.

명령 실행

Systems Manager에는 Systems Manager 콘솔의 Run Command 페이지에서 명령을 다시 실행하는 2가지 옵션이 있습니다.

- Rerun(다시 실행): 이 버튼을 사용하면 명령을 변경하지 않고 동일한 명령을 실행할 수 있습니다.
- 새로 복사(Copy to new): 이 버튼은 한 명령의 설정을 새 명령으로 복사하고 실행하기 전에 해당 설정을 편집할 수 있는 옵션을 제공합니다.

명령을 다시 실행하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택합니다.
3. 다시 실행할 명령을 선택합니다. 명령 세부 정보 페이지에서 명령을 실행한 직후 명령을 다시 실행할 수 있습니다. 또는 명령 기록(Command history) 탭에서 이전에 실행한 명령을 선택할 수 있습니다.
4. Rerun(다시 실행)을 선택하여 동일한 명령을 변경 없이 실행하거나, 명령을 실행하기 전에 새로 복사를 선택하여 명령 설정을 편집합니다.

특정 문서 버전을 사용하여 명령 실행

문서 버전 파라미터를 사용하여 명령 실행 시 사용할 AWS Systems Manager 문서 버전을 지정할 수 있습니다. 이 파라미터에는 다음 옵션 중 하나를 지정할 수 있습니다.

- $DEFAULT
- $LATEST
- 버전 번호

문서 버전 파라미터를 실행하여 명령을 실행하려면 다음 절차를 사용합니다.

Linux

로컬 Linux 시스템에서 AWS CLI를 사용하여 명령을 실행하려면

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.
2. 사용할 수 있는 모든 문서 나열

   이 명령을 실행하면 AWS Identity and Access Management(IAM) 권한에 따라 계정에 사용할 수 있는 문서를 모두 나열합니다.

   ```
   aws ssm list-documents
   ```
3. 다음 명령을 실행하여 문서의 여러 버전을 확인합니다. *document name*을 사용자의 정보로 바꿉니다.

   ```
   aws ssm list-document-versions \
       --name "document name"
   ```
4. 다음 명령을 실행하여 SSM 문서 버전을 사용하는 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   ```
   aws ssm send-command \
       --document-name "AWS-RunShellScript" \
       --parameters commands="echo Hello" \
       --instance-ids instance-ID \
       --document-version '$LATEST'
   ```

Windows

로컬 Windows 시스템에서 AWS CLI를 사용하여 명령을 실행하려면

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.
2. 사용할 수 있는 모든 문서 나열

   이 명령을 실행하면 AWS Identity and Access Management(IAM) 권한에 따라 계정에 사용할 수 있는 문서를 모두 나열합니다.

```
aws ssm list-documents
```

3. 다음 명령을 실행하여 문서의 여러 버전을 확인합니다. *document name*을 사용자의 정보로 바꿉니다.

```
aws ssm list-document-versions ^
    --name "document name"
```

4. 다음 명령을 실행하여 SSM 문서 버전을 사용하는 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
aws ssm send-command ^
    --document-name "AWS-RunShellScript" ^
    --parameters commands="echo Hello" ^
    --instance-ids instance-ID ^
    --document-version "$LATEST"
```

PowerShell

Tools for PowerShell을 사용하여 명령을 실행하려면

1. 아직 설치하지 않은 경우 AWS Tools for PowerShell(Tools for Windows PowerShell)을 설치하고 구성합니다.

   자세한 내용은 AWS Tools for PowerShell 설치를 참조하세요.

2. 사용할 수 있는 모든 문서 나열

   이 명령을 실행하면 AWS Identity and Access Management(IAM) 권한에 따라 계정에 사용할 수 있는 문서를 모두 나열합니다.

```
Get-SSMDocumentList
```

3. 다음 명령을 실행하여 문서의 여러 버전을 확인합니다. *document name*을 사용자의 정보로 바꿉니다.

```
Get-SSMDocumentVersionList `
    -Name "document name"
```

4. 다음 명령을 실행하여 SSM 문서 버전을 사용하는 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
Send-SSMCommand `
    -DocumentName "AWS-RunShellScript" `
    -Parameter @{commands = "echo helloWorld"} `
    -InstanceIds "instance-ID" `
    -DocumentVersion $LATEST
```

대규모로 명령 실행

AWS Systems Manager의 기능인 Run Command에서는 `targets`을 사용하여 관리형 노드의 플릿에 대해 명령을 실행할 수 있습니다. `targets` 파라미터는 관리형 노드에 대해 지정한 태그를 바탕으로 `Key,Value` 조합을 허용합니다. 명령을 실행하면 시스템에서 지정된 태그와 일치하는 모든 관리형 노드를 찾아 이러한 인스턴스에 대해 명령 실행을 시도합니다. 관리형 인스턴스의 태그 지정에 대한 자세한 내용은 AWS 리소스 태그 지정 사용 설명서의 AWS 리소스 태그 지정을 참조하세요. 관리형 IoT 디바이스 태그 지정에 대한 자세한 내용은 AWS IoT Greengrass Version 2개발자 안내서의 AWS IoT Greengrass Version 2 리소스 태그하기 섹션을 참조하세요.

다음 섹션에 설명된 대로 `targets` 파라미터를 사용하여 특정 관리형 노드 ID 목록을 대상으로 지정할 수도 있습니다.

수백 또는 수천 개의 관리형 노드에 걸쳐 명령 실행을 제어할 수 있도록 Run Command에는 하나의 요청을 동시에 처리할 수 있는 노드의 수와 명령이 취소되기 전에 명령에서 발생시킬 수 있는 오류의 수를 제한하기 위한 파라미터가 포함되어 있습니다.

내용

다중 관리형 노드 대상 지정

태그, AWS 리소스 그룹 이름, 또는 관리형 노드 ID를 지정하여 명령 및 대상 관리 노드를 실행할 수 있습니다.

다음 예시에서는 AWS Command Line Interface(AWS CLI)에서 Run Command를 사용할 경우의 명령 형식을 보여줍니다. *example resource placeholder*를 사용자의 정보로 바꿉니다. 이번 단원의 샘플 명령은 `[...]`를 사용해 잘립니다.

예제 1: 태그를 대상으로 지정

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 ^
    [...]
```

예제 2: AWS 리소스 그룹을 대상으로 지정

명령 한 번마다 지정할 수 있는 리소스 그룹 이름은 최대 1개입니다. 리소스 그룹을 생성할 때는 그룹화 기준에서 `AWS::SSM:ManagedInstance`와 `AWS::EC2::Instance`를 리소스 유형으로 추가하는 것이 좋습니다.

> **Note**
>
> 리소스 그룹을 대상으로 지정하는 명령을 전송하려면 먼저 해당 그룹에 속한 리소스를 나열하거나 볼 수 있는 AWS Identity and Access Management(IAM) 권한을 부여해야 합니다. 자세한 내용은 AWS Resource Groups 사용 설명서의 권한 설정을 참조하세요.

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 ^
    [...]
```

예제 3: 리소스 유형별로 AWS 리소스 그룹 태그 지정

명령 한 번마다 지정할 수 있는 리소스 그룹 유형은 최대 5개입니다. 리소스 그룹을 생성할 때는 그룹화 기준에서 `AWS::SSM:ManagedInstance`와 `AWS::EC2::Instance`를 리소스 유형으로 추가하는 것이 좋습니다.

> **Note**
>
> 리소스 그룹을 대상으로 지정하는 명령을 전송하려면 먼저 해당 그룹에 속한 리소스를 나열하거나 볼 수 있는 IAM 권한을 부여해야 합니다. 자세한 내용은 AWS Resource Groups 사용 설명서의 권한 설정을 참조하세요.

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 ^
    [...]
```

예제 4: 인스턴스 ID를 대상으로 지정

다음 예에서는 `instanceids` 키와 `targets` 파라미터를 사용하여 관리형 노드를 대상으로 지정하는 방법을 보여줍니다. 각 디바이스에 mi-*ID_number*가 지정되어 있기 때문에 이 키를 사용하여 관리형 AWS IoT Greengrass 코어 디바이스를 대상으로 지정할 수 있습니다. AWS Systems Manager의 기능인 Fleet Manager에서 디바이스 ID를 확인할 수 있습니다.

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 ^
    [...]
```

`Environment`라는 `Key`와 `Development`, `Test`, `Pre-production` 및 `Production`의 `Values`를 사용하여 다양한 환경에 대해 관리형 노드에 태그를 지정한 경우에는 다음 구문과 함께 `targets` 파라미터를 사용해 이러한 환경 중 하나의 모든 관리형 노드로 명령을 보낼 수 있습니다.

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 ^
    [...]
```

`Values` 목록에 추가하여 다른 환경의 추가 관리형 노드를 대상으로 지정할 수 있습니다. 쉼표를 사용하여 항목을 구분합니다.

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 ^
    [...]
```

번형: Key 기준을 여러 개 사용하여 대상 세분화

Key 조건을 여러 개 포함시켜 명령의 대상 수를 세분화할 수 있습니다. Key 조건을 두 개 이상 포함시키면 시스템에서 모든 조건을 충족하는 관리형 노드를 대상으로 지정합니다. 다음 명령은 재무 부서에 대해 태그가 지정되고 또한 데이터베이스 서버 역할에 대해 태그가 지정된 모든 관리형 노드를 대상으로 지정합니다.

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 ^
    [...]
```

번형: 여러 Key 및 Value 기준 사용

앞의 예제를 확장해 Values 조건에 항목을 추가로 포함시켜 여러 부서 및 여러 서버 역할을 대상으로 지정할 수 있습니다.

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 ^
    [...]
```

번형: 여러 `Values` 기준을 사용하여 태그가 지정된 관리형 노드를 대상으로 지정

`Department`라는 `Key`와 `Sales`, `Finance`의 `Values`를 사용하여 다양한 환경에 대해 관리형 노드에 태그를 지정한 경우에는 다음 구문과 함께 `targets` 파라미터를 사용해 이러한 환경의 모든 노드로 명령을 보낼 수 있습니다.

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 ^
    [...]
```

최대 5개의 키와 각 키에 대해 5개의 값을 지정할 수 있습니다.

태그 키(태그 이름) 또는 태그 값에 공백이 포함된 경우 다음 예와 같이 태그 키 또는 값을 인용 부호로 묶습니다.

예: `Value` 태그의 공백

Linux & macOS

```
aws ssm send-command \
    --document-name document-name \
    --targets Key=tag:OS,Values="Windows Server 2016 Nano" \
    [...]
```

Windows

```
aws ssm send-command ^
    --document-name document-name ^
    --targets Key=tag:OS,Values="Windows Server 2016 Nano" ^
    [...]
```

예: tag 키 및 Value의 공백

Linux & macOS

```
aws ssm send-command \
    --document-name document-name \
    --targets Key="tag:Operating System",Values="Windows Server 2016 Nano" \
    [...]
```

Windows

```
aws ssm send-command ^
    --document-name document-name ^
    --targets Key="tag:Operating System",Values="Windows Server 2016 Nano" ^
    [...]
```

예: Values 목록에서 한 항목의 공백

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" ^
    [...]
```

비율 제어 사용

동시성 제어와 오류 제어를 사용하여 그룹의 관리형 노드로 명령이 전송되는 비율을 제어할 수 있습니다.

주제

- 동시성 제어 사용
- 오류 제어 사용

동시성 제어 사용

max-concurrency 파라미터(Run a command(명령 실행) 페이지의 Concurrency(동시성) 옵션))로 명령을 동시에 실행하는 관리형 노드의 수를 제어할 수 있습니다. 관리형 노드의 절대 개수(예: **10**)를 지정하거나 대상 집합의 비율(예: **10%**)을 지정할 수 있습니다. 대기 중인 시스템은 단일 노드에 명령을 전송하고 시스템이 초기 호출을 승인할 때까지 기다렸다가 명령을 2개 더 많은 노드에 보냅니다. 시스템은 시스템이 max-concurrency 값을 충족할 때까지 기하급수적으로 더 많은 노드에 명령을 보냅니다. max-concurrency의 기본값은 50입니다. 다음 예는 max-concurrency 파라미터에 대한 값을 지정하는 방법을 보여줍니다.

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 ^
    [...]
```

오류 제어 사용

max-errors 파라미터(명령 실행 페이지의 오류 임계값(Error threshold) 필드)로 오류 제한을 설정하여 수백 또는 수천 개의 관리형 노드에 대한 명령 실행을 제어할 수도 있습니다. 이 파라미터는 시스템이 추가 관리형 노드로의 명령 전송을 중지하기까지 허용되는 오류 횟수를 지정합니다. 오류의 절대 개수(예: **10**)를 지정하거나 대상 집합의 비율(예: **10%**)을 지정할 수 있습니다. 예를 들어 **3**을 지정하면 네 번째 오류를 받았을 때 명령 전송이 중지됩니다. **0**을 지정하면 첫 번째 오류 결과가 반환된 후 추가 관리형 노드로의 명령 전송이 중지됩니다. 50개의관리형 노드로 명령을 보내고 max-errors을 **10%**로 설정하면 여섯 번째 오류를 받았을 때 추가 노드로의 명령 전송이 중지됩니다.

max-errors에 도달할 때 이미 명령을 실행 중인 호출은 완료될 수 있지만, 이런 호출 중 일부는 실패할 수도 있습니다. 실패한 호출 수가 max-errors보다 많지 않도록 해야 할 경우에는 호출이 한 번에 한 개를 초과하지 않도록 max-concurrency를 **1**로 설정하십시오. max-errors의 기본값은 0입니다. 다음 예는 max-errors 파라미터에 대한 값을 지정하는 방법을 보여줍니다.

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 ^
    [...]
```

명령 취소

서비스에 명령이 보류 중 또는 실행 중 상태로 표시되는 경우 명령을 취소할 수 있습니다. 그러나 명령이 그러한 상태 중 하나인 경우에도 명령이 취소되고 기본 프로세스가 중지되는 것을 보장할 수 없습니다.

콘솔을 사용하여 명령을 취소하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택합니다.
3. 취소하고 싶은 명령 호출을 선택합니다.
4. 명령 취소를 선택합니다.

AWS CLI를 사용하여 명령을 취소하려면

다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

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"
```

취소된 명령의 상태에 대한 자세한 내용은 명령 상태 이해 섹션을 참조하세요.

명령에 종료 코드 사용

경우에 따라 종료 코드를 사용하여 명령을 처리하는 방법을 관리해야 할 수도 있습니다.

명령에 종료 코드 지정

AWS Systems Manager의 기능인 Run Command를 사용하여 종료 코드를 지정함으로써 명령 처리 방법을 결정할 수 있습니다. 기본적으로 스크립트에서 실행된 마지막 명령의 종료 코드는 전체 스크립트의 종료 코드로 보고됩니다. 예를 들어 세 개의 명령이 포함된 스크립트가 있습니다. 첫 번째 명령은 실패하지만 다음 명령은 성공합니다. 마지막 명령이 성공했기 때문에 실행 상태는 succeeded로 보고됩니다.

Shell 스크립트

첫 번째 명령 실패 시 전체 스크립트를 실패하려면, 마지막 명령 이전에 명령이 실패하는 경우 셸 조건문을 포함하여 스크립트를 종료할 수 있습니다. 다음과 같은 접근 방식을 사용하지 마십시오.

```
<command 1>
    if [ $? != 0 ]
    then
        exit <N>
    fi
    <command 2>
    <command 3>
```

다음 예제에서는 첫 번째 명령이 실패하면 전체 스크립트가 실패합니다.

```
cd /test
    if [ $? != 0 ]
    then
        echo "Failed"
        exit 1
    fi
    date
```

PowerShell 스크립트

PowerShell에서 종료 코드를 성공적으로 캡처하려면 Run Command에 대한 스크립트에서 `exit`를 명시적으로 호출해야 합니다.

```
<command 1>
    if ($?) {<do something>}
    else {exit <N>}
    <command 2>
    <command 3>
    exit <N>
```

예:

```
cd C:\
    if ($?) {echo "Success"}
    else {exit 1}
    date
```

명령 실행 시 재부팅 처리

AWS Systems Manager의 기능인 Run Command을 사용하여 관리형 노드를 재부팅하는 스크립트를 실행할 수 있으며, 스크립트에 엑시트 코드를 지정하는 것이 좋습니다. 다른 방식으로 스크립트에서 노드를 재부팅하려고 시도할 경우, 재부팅이 스크립트의 마지막 단계라 하더라도 스크립트 실행 상태가 올바로 업데이트되지 않을 수 있습니다. Windows 관리형 노드의 경우 스크립트에서 `exit 3010`을 지정합니다. Linux 및 macOS 관리형 노드의 경우 `exit 194`를 지정합니다. 종료 코드는 AWS Systems Manager 에이전트(SSM Agent)에 관리형 노드를 재부팅하라고 지시한 다음 재부팅이 완료되면 스크립트를 재시작합니다. 재부팅을 시작하기 전에 SSM Agent는 클라우드 내 Systems Manager 서비스에 서버를 재부팅하는 동안 통신이 중단됨을 알립니다.

> **Note**
>
> 재부팅 스크립트는 `aws:runDocument` 플러그 인의 일부일 수 없습니다. 문서에 재부팅 스크립트가 포함되어 있고 다른 문서가 `aws:runDocument` 플러그 인을 통해 해당 문서를 실행하려고 할 경우 SSM Agent에 오류가 발생합니다.

idempotent 스크립트 생성

관리형 노드를 재부팅하는 스크립트를 개발할 때는 재부팅 후 떠난 곳에서 스크립트 실행을 계속하도록 idempotent 스크립트를 만듭니다. idempotent 스크립트는 상태를 관리하며 작업 수행 여부를 검증합니다. 이렇게 하면 한 번만 실행되도록 되어 있는 단계가 여러 번 실행되는 것을 방지할 수 있습니다.

다음은 관리형 노드를 여러 번 재부팅하는 idempotent 스크립트의 대략적인 예입니다.

```
$name = Get current computer name
If ($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
    }
```

예시

다음 스크립트 샘플은 종료 코드를 사용하여 관리형 노드를 재시작합니다. Linux 예제에서는 Amazon Linux에 패키지 업데이트를 설치한 다음 노드를 다시 시작합니다. Windows Server 예시에서는 노드에 Telnet-Client를 설치한 다음 노드를 다시 시작합니다.

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
    }
```

명령 상태 이해

AWS Systems Manager의 기능인 Run Command는 어떤 명령을 처리하는 동안과 명령을 처리한 각 관리형 노드에 대해 그 명령이 거치게 되는 다양한 상태에 대한 자세한 상태 정보를 보고합니다. 다음 방법을 사용하여 명령 상태를 모니터링할 수 있습니다.

- Run Command 콘솔 인터페이스의 Commands(명령) 탭에서 Refresh(새로 고침) 아이콘을 선택합니다.

- AWS Command Line Interface(AWS CLI)를 사용하여 list-commands 또는 list-command-invocations를 호출합니다. AWS Tools for Windows PowerShell을 사용하여 Get-SSMCommand 또는 Get-SSMCommandInvocation을 호출합니다.
- 상태 변경에 응답하도록 Amazon EventBridge를 구성합니다.
- 모든 상태 변경 또는 `Failed`, `TimedOut` 등의 특정 상태에 대해 알림을 보내도록 Amazon Simple Notification Service(SNS)를 구성합니다.

Run Command 상태

Run Command는 플러그인, 호출, 전체 명령 상태라는 세 가지 영역에 대한 자세한 상태 정보를 보고합니다. 플러그인은 명령의 SSM 문서에 정의된 코드 실행 블록입니다. 플러그인에 대한 자세한 내용은 Command 문서 플러그인 참조 섹션을 참조하세요.

동시에 여러 관리형 노드에 명령을 보내는 경우, 각 노드를 대상으로 하는 명령 사본 각각을 명령 호출이라고 합니다. 예를 들어 `AWS-RunShellScript` 문서를 사용하여 `ifconfig` 명령을 20개의 Linux 인스턴스에 보내는 경우 이 명령에는 20개의 호출이 있습니다. 각 명령 호출은 개별적으로 상태를 보고합니다. 지정된 명령 호출에 대한 플러그인 역시 상태를 개별적으로 보고합니다.

마지막으로, Run Command에는 모든 플러그인과 호출에 대한 총체적인 명령 상태가 포함되어 있습니다. 다음 표에 나와 있는 것처럼, 총체적인 명령 상태는 플러그인이나 호출에서 보고되는 상태와 다를 수 있습니다.

> **Note**
>
> `max-concurrency` 또는 `max-errors` 파라미터를 사용하여 많은 수의 관리형 노드에 명령을 실행할 경우 다음 표의 설명과 같이 명령 상태는 해당 파라미터에 따른 제한을 반영합니다. 이런 파라미터에 대한 자세한 내용은 대규모로 명령 실행 섹션을 참조하세요.

명령 플러그인 및 호출의 세부 상태

상태 표시기	Details
보류중	명령이 아직 관리형 노드로 전송되지 않았거나 SSM Agent에 의해 수신되지 않았습니다. 시간 제한(초)(Timeout (seconds)) 파라미터와 실행 시간 제한(Execution timeout) 파라미터의 합에 해당하는 시간이 경과하기 전에 에이전트

상태 표시기	Details
	가 명령을 수신하지 않으면 상태가 `Delivery Timed Out`로 변경됩니다.
InProgress	Systems Manager가 관리형 노드에 명령을 보내려고 시도하거나 SSM Agent에서 명령을 수신하고 인스턴스에서 실행을 시작했습니다. 모든 명령 플러그인의 결과에 따라 상태는 `Success`, `Failed`, `Delivery Timed Out` 또는 `Execution Timed Out`으로 변경됩니다. 예외: 에이전트가 노드에서 실행 중이 아니거나 사용 가능한 경우 명령 상태는 에이전트를 다시 사용할 수 있거나 실행 시간 제한에 도달할 때까지 `In Progress`으로 유지됩니다. 상태는 종료 상태로 변경됩니다.
Delayed	시스템에서 관리형 노드로 명령 전송을 시도했지만 완료하지 못했습니다. 시스템이 다시 시도합니다.

상태 표시기	Details
Success	이 상태는 다양한 조건에서 반환됩니다. 이 상태가 노드에서 명령이 처리되었다는 뜻은 아닙니다. 예를 들어 관리형 노드의 SSM Agent에 명령이 수신된 후 PowerShell `Execution Policy` 에서 명령 실행을 차단한 결과로 종료 코드 0이 반환될 수 있습니다. 이것은 종료 상태입니다. 명령에서 `Success` 상태를 반환하는 조건은 다음과 같습니다. • 단일 인스턴스를 타겟팅할 때 명령이 관리형 노드의 SSM Agent에 수신되었고 종료 코드 0이 반환되었습니다. • 여러 인스턴스를 타겟팅할 때 실패한 호출 수가 명령에 지정된 오류 임계값을 초과하지 않았습니다. • 여러 인스턴스를 타겟팅할 때 한 번 이상의 호출이 성공했지만 나머지 호출은 제한 시간이 초과되었습니다. 지정된 오류 임계값은 여전히 적용됩니다. • 태그를 타겟팅할 때 태그와 관련된 인스턴스가 발견되지 않았습니다. • 태그를 타켓팅할 때 실패한 호출 수가 명령에 지정된 오류 임계값을 초과하지 않았습니다. • 태그를 타켓팅할 때 한 번 이상의 호출이 성공했지만 나머지 호출은 제한 시간이 초과되었습니다. 지정된 오류 임계값은 여전히 적용됩니다. • 명령 실행을 방지하거나 재정의하여 종료 코드 0이 반환되는 애플리케이션 또는 정책을 OS 수준에서 적용되고 있습니다.

상태 표시기	Details
	Note 리소스 그룹을 타겟팅할 때도 동일한 조건이 적용됩니다. 오류를 해결하거나 명령 실행에 대한 자세한 정보를 얻으려면 적절한 종료 코드(명령 실패에 대한 0 이외의 종료 코드)를 반환해 오류나 예외를 처리하는 명령을 전송합니다.
DeliveryTimedOut	총 시간 제한 만료 전까지 명령이 관리형 노드로 전송되지 않았습니다. 상위 명령의 `max-errors` 제한에 총 시간 초과 횟수가 계산되지는 않지만, 상위 명령 상태를 `Success`, `Incomplete` 또는 `Delivery Timed Out`으로 표시할지 여부에는 영향을 미칩니다. 이것은 종료 상태입니다.
ExecutionTimedOut	관리형 노드에서 명령 자동화가 시작되었지만, 실행 시간제한 만료 전까지 명령이 완료되지 않았습니다. 실행 시간 초과는 실패로 계산되며, 이 경우 0이 아닌 응답이 전송되고 Systems Manager는 명령 자동화 실행 시도를 종료하고 실패 상태를 보고합니다.
Failed	관리형 노드에서 명령 실행을 완료하지 못했습니다. 플러그인의 경우 이는 결과 코드가 0이 아니었음을 나타냅니다. 명령 호출의 경우 이는 하나 이상의 플러그인에 대한 결과 코드가 0이 아니었음을 나타냅니다. 호출 실패는 상위 명령의 `max-errors` 제한에 계산됩니다. 이것은 종료 상태입니다.
취소됨	명령이 완료되기 전에 취소되었습니다. 이것은 종료 상태입니다.

상태 표시기	Details
Undeliverable	관리형 노드로 명령을 전달할 수 없습니다. 해당 노드가 없거나 응답하지 않는 것일 수 있습니다. 배달할 수 없는 호출은 상위 명령의 max-errors 제한에 계산되지 않지만 상위 명령 상태를 Success 또는 Incomplete 으로 표시할지 여부에 영향을 미치지도 않습니다. (예를 들어 명령의 모든 호출이 Undeliverable 상태인 경우 반환된 명령 상태는 Failed입니다. 그러나 명령에 5개 호출이 있는 경우 그 중 4개가 Undeliverable 상태를 반환하고 나머지 1개가 Success 상태를 반환하는 경우 상위 명령의 상태는 Success입니다.) 이것은 종료 상태입니다.
Terminated	상위 명령이 max-errors 제한을 초과해 시스템에서 이후의 명령 호출을 취소했습니다. 이것은 종료 상태입니다.
InvalidPlatform	선택한 문서에 지정된 필수 플랫폼과 일치하지 않는 관리형 노드로 명령이 전송되었습니다. Invalid Platform은 상위 명령의 최대 오류 제한에 포함되지 않지만 상위 명령 상태가 성공인지 실패인지 여부에 영향을 줍니다. (예를 들어 명령의 모든 호출이 Invalid Platform 상태인 경우 반환된 명령 상태는 Failed입니다. 그러나 명령에 5개 호출이 있는 경우 그 중 4개가 Invalid Platform 상태를 반환하고 나머지 1개가 Success 상태를 반환하는 경우 상위 명령의 상태는 Success입니다.) 이것은 종료 상태입니다.

상태 표시기	Details
AccessDenied	명령을 시작하는 AWS Identity and Access Management(IAM) 사용자 또는 역할은 대상 지정된 관리형 노드에 액세스할 수 없습니다. `Access Denied`는 상위 명령의 `max-errors` 한도에 불리하게 작용하지 않지만 상위 명령 상태가 `Success` 또는 `Failed`인지 여부에 원인을 제공합니다. (예를 들어 명령의 모든 호출이 `Access Denied` 상태인 경우 반환된 명령 상태는 `Failed`입니다. 그러나 명령에 5개 호출이 있는 경우 그 중 4개가 `Access Denied` 상태를 반환하고 나머지 1개가 `Success` 상태를 반환하는 경우 상위 명령의 상태는 `Success`입니다.) 이것은 종료 상태입니다.

명령의 세부 상태

상태 표시기	Details
보류중	어떤 관리형 노드에서도 에이전트가 아직 명령을 받지 않았습니다.
InProgress	1개 이상의 관리형 노드로 명령을 전송했지만 모든 노드에서 최종 상태에 도달하지는 못했습니다.
Delayed	시스템에서 노드로 명령 전송을 시도했지만 완료하지 못했습니다. 시스템이 다시 시도합니다.
Success	지정되었거나 대상이 된 모든 관리형 노드의 SSM Agent에 명령이 수신되었고 종료 코드 0을 반환했습니다. 모든 명령 호출이 터미널 상태에 도달했고 `max-errors` 값에 도달하지 않았습니다. 이 상태가 지정되었거나 대상이 된 모든 관리형 노드에서 명령이 처리되었다는 뜻은 아닙니다. 이것은 종료 상태입니다.

상태 표시기	Details
	Note 오류를 해결하거나 명령 실행에 대한 자세한 정보를 얻으려면 적절한 종료 코드(명령 실패에 대한 0 이외의 종료 코드)를 반환해 오류나 예외를 처리하는 명령을 전송합니다.
DeliveryTimedOut	총 시간 제한 만료 전까지 명령이 관리형 노드로 전송되지 않았습니다. `max-errors` 이상의 명령 호출 값이 `Delivery Timed Out` 상태를 표시합니다. 이것은 종료 상태입니다.
Failed	관리형 노드에서 명령 실행을 완료하지 못했습니다. `max-errors` 이상의 명령 호출 값이 `Failed` 상태를 표시합니다. 이것은 종료 상태입니다.
불완전	모든 관리형 노드에서 명령을 시도했고 값이 `Success`가 아닌 호출이 한 개 이상입니다. 하지만 `Failed` 상태가 될 정도로 많은 호출에 실패한 것은 아닙니다. 이것은 종료 상태입니다.
취소됨	명령이 완료되기 전에 취소되었습니다. 이것은 종료 상태입니다.
RateExceeded	명령의 대상이 되는 관리형 노드 수가 보류 중인 호출에 대한 계정 할당량을 초과했습니다. 시스템에서 이 명령을 어떤 노드에서 실행하기 전에 취소했습니다. 이것은 종료 상태입니다.

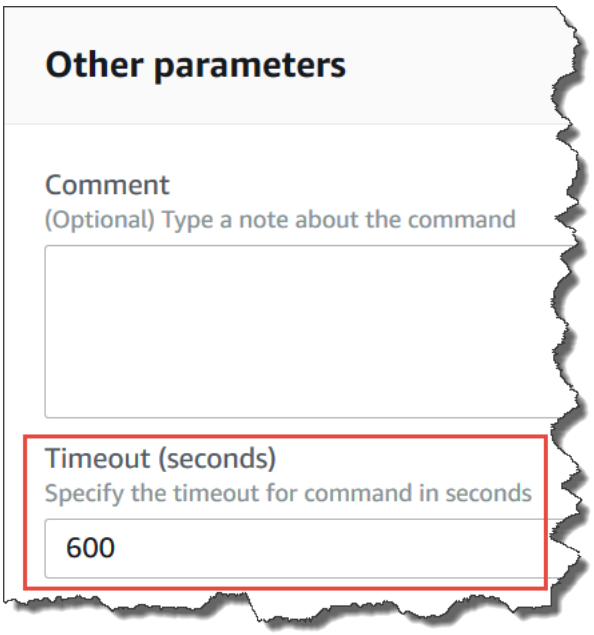
상태 표시기	Details
AccessDenied	명령을 시작하는 사용자 또는 역할은 대상으로 지정된 리소스 그룹에 액세스할 수 없습니다. `AccessDenied` 는 상위 명령의 `max-errors` 한도에 불리하게 작용하지 않지만 상위 명령 상태가 `Success` 또는 `Failed`인지 여부에 원인을 제공합니다. (예를 들어 명령의 모든 호출이 `AccessDenied` 상태인 경우 반환된 명령 상태는 `Failed`입니다. 그러나 명령에 5번의 호출이 있고 그 중 4번은 상태 `AccessDenied` 를 반환하고 1번은 `Success` 상태를 반환하는 경우 상위 명령의 상태는 `Success`입니다.) 이것은 종료 상태입니다.
태그에 인스턴스 없음	명령으로 대상 지정된 태그 키 페어 또는 리소스 그룹은 관리형 노드와 일치하지 않습니다. 이것은 종료 상태입니다.

명령 제한 시간 값 이해

Systems Manager는 명령을 실행할 때 다음 시간 제한 값을 적용합니다.

총 제한 시간

Systems Manager 콘솔에서 시간 제한(초)(Timeout (seconds)) 필드에 시간 제한 값을 지정합니다. 명령이 전송된 후 Run Command는 명령이 만료되었는지 여부를 확인합니다. 명령이 명령 만료 제한(총 시간 제한)에 도달하면 상태가 `InProgress`, `Pending` 또는 `Delayed`인 모든 호출에 대해 상태가 `DeliveryTimedOut`으로 변경됩니다.



보다 기술적인 수준에서 총 시간 제한(시간 제한(초)(Timeout (seconds)))은 다음과 같이 두 가지 시간 제한 값의 조합입니다.

`Total timeout = "Timeout(seconds)" from the console + "timeoutSeconds": "{{ executionTimeout }}" from your SSM document`

예를 들어 Systems Manager 콘솔에서 시간 제한(초)(Timeout (seconds))의 기본값은 600초입니다. `AWS-RunShellScript`SSM 문서를 사용하여 명령을 실행하는 경우 다음 문서 샘플과 같이 "timeoutSeconds": "{{ executionTimeout }}"의 기본값은 3,600초입니다.

```
"executionTimeout": {
    "type": "String",
    "default": "3600",

"runtimeConfig": {
  "aws:runShellScript": {
    "properties": [
      {
        "timeoutSeconds": "{{ executionTimeout }}"
```

즉, 시스템이 명령 상태를 `DeliveryTimedOut`으로 설정하기 전에 명령이 4,200초(70분) 동안 실행됩니다.

실행 제한 시간

Systems Manager 콘솔의 실행 시간 제한(Execution Timeout) 필드에 실행 시간 제한 값을 지정합니다(사용 가능한 경우). 모든 SSM 문서에 실행 시간 제한을 지정해야 하는 것은 아닙니다. Execution Timeout(실행 시간 초과) 필드는 SSM 문서에 해당 입력 파라미터가 정의되어 있는 경우에만 표시됩니다. 지정된 경우 이 시간 내에 명령이 완료되어야 합니다.

> **Note**
>
> Run Command는 명령이 에이전트에 전송되었는지 여부를 판별하기 위해 SSM Agent 문서 터미널 응답에 의존합니다. SSM Agent는 `ExecutionTimedOut`으로 표시되는 호출 또는 명령에 대한 `ExecutionTimedOut` 신호를 보내야 합니다.

기본 실행 제한 시간

SSM 문서에서 실행 제한 시간 값을 명시적으로 지정할 필요가 없는 경우 Systems Manager는 하드 코딩된 기본 실행 제한 시간을 적용합니다.

Systems Manager가 시간 제한을 보고하는 방법

Systems Manager가 대상의 SSM Agent로부터 `execution timeout` 응답을 수신한 경우 Systems Manager는 명령 호출을 `executionTimeout`으로 표시합니다.

Run Command가 SSM Agent로부터 문서 터미널 응답을 받지 못하면 명령 호출은 `deliveryTimeout`으로 표시됩니다.

대상의 시간 제한 상태를 확인하기 위해 SSM Agent는 모든 파라미터와 SSM 문서의 콘텐츠를 결합하여 `executionTimeout`을 계산합니다. SSM Agent에서 명령이 시간 초과되었다고 판단하면 `executionTimeout`을 서비스로 전송합니다.

시간 제한(초)(Timeout (seconds))의 기본값은 3,600초입니다. 실행 시간 제한(Execution Timeout)의 기본값도 3,600초입니다. 따라서 명령의 총 기본 시간 제한은 7,200초입니다.

> **Note**
>
> SSM Agent는 SSM 문서 유형 및SSM 문서 버전에 따라 `executionTimeout`을 다르게 처리합니다.

Run Command 연습

이 섹션의 시연에서는 AWS Command Line Interface(AWS CLI) 또는 AWS Tools for Windows PowerShell을 사용하여 AWS Systems Manager의 기능인 Run Command로 명령을 실행하는 방법을 보여줍니다.

내용

- Run Command를 사용하여 소프트웨어 업데이트
- 연습: Run Command에서 AWS CLI 사용
- 연습: Run Command에서 AWS Tools for Windows PowerShell 사용

다음 참조에서도 명령 샘플을 볼 수 있습니다.

- Systems Manager AWS CLI 참조
- AWS Tools for Windows PowerShell - AWS Systems Manager

Run Command를 사용하여 소프트웨어 업데이트

다음 절차에서는 관리형 노드에서 소프트웨어를 업데이트하는 방법을 설명합니다.

Run Command를 사용하여 SSM Agent 업데이트

다음 절차에서는 관리형 노드에서 실행 중인 SSM Agent를 업데이트하는 방법을 설명합니다. SSM Agent의 최신 버전으로 업데이트하거나 이전 버전으로 다운그레이드할 수 있습니다. 명령을 실행하면 시스템은 AWS에서 버전을 다운로드하여 설치한 다음, 명령을 실행하기 이전에 있었던 버전을 제거합니다. 이 프로세스 중에 오류가 발생하면, 서버에서 명령을 실행하기 이전의 버전으로 롤백하며 명령 상태에 명령이 실패했다고 표시됩니다.

> **Note**
>
> 인스턴스가 macOS 버전 11.0(Big Sur) 이상을 실행하는 경우 AWS-UpdateSSMAgent 문서를 실행하려면 인스턴스의 SSM Agent 버전이 3.1.941.0 이상이어야 합니다. 인스턴스

가 3.1.941.0 이전에 릴리스된 SSM Agent 버전을 실행 중인 경우 `brew update` 및 `brew upgrade amazon-ssm-agent` 명령을 실행하여 AWS-UpdateSSMAgent 문서를 실행하도록 SSM Agent를 업데이트할 수 있습니다.

SSM Agent 업데이트에 대해 알림을 수신하려면 GitHub에서 SSM Agent 릴리스 정보 페이지를 구독합니다.

Run Command를 사용하여 SSM Agent를 업데이트하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택합니다.
3. Run command를(Run 명령) 선택합니다.
4. Command 문서(Command document) 목록에서 **AWS-UpdateSSMAgent**를 선택합니다.
5. 명령 파라미터 섹션에서, 원하는 경우 다음 파라미터의 값을 지정합니다.
   a. (옵션) 버전(Version)에 설치할 SSM Agent의 버전을 입력합니다. 이전 버전의 에이전트를 설치할 수 있습니다. 버전을 지정하지 않으면 최신 버전으로 설치됩니다.
   b. (옵션) 다운그레이드 허용(Allow Downgrade)에서 true를 선택하여 이전 버전의 SSM Agent를 설치합니다. 이 옵션을 선택하는 경우 이전 버전 번호를 지정해야 합니다. 최신 서비스 버전을 설치하려면 false를 선택합니다.
6. 대상(Targets) 섹션에서, 태그를 지정하거나, 수동으로 인스턴스나 엣지 디바이스를 선택하거나, 리소스 그룹을 지정하여 이 작업을 실행할 관리형 노드를 식별합니다.

   > **Tip**
   > 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

7. Other parameters(다른 파라미터):
   - Comment(설명)에 명령에 대한 정보를 입력합니다.
   - 제한 시간(초)에서 전체 명령 실행이 실패할 때까지 시스템이 기다리는 시간을 초 단위로 지정합니다.
8. Rate control(속도 제어)에서

- Concurrency(동시성)에서 명령을 동시에 실행할 관리형 노드의 백분율 또는 개수를 지정합니다.

> **Note**
>
> 관리형 노드에 적용할 태그를 지정하거나, AWS 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상 수를 제한합니다.

- Error threshold(오류 임계값)에서, 명령이 노드의 개수 또는 백분율에서 실패한 후 다른 관리형 노드에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째 오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.

9. (선택 사항) Output options(출력 옵션)에서 명령 출력을 파일에 저장하려면 Write command output to an S3 bucket(S3 버킷에 명령 출력 쓰기) 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

> **Note**
>
> 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아니라 인스턴스에 할당된 인스턴스 프로파일(EC2 인스턴스용) 또는 IAM 서비스 역할(하이브리드 정품 인증 시스템)의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

10. SNS notifications(SNS 알림) 섹션에서, 명령 실행 상태에 대한 알림이 전송되도록 하려면 Enable SNS notifications(SNS 알림 활성화) 확인란을 선택합니다.

    Run Command에 대한 Amazon SNS 알림 구성에 대한 자세한 내용은 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 섹션을 참조하세요.

11. Run(실행)을 선택합니다.

Run Command를 사용하여 PowerShell 업데이트

다음 절차에서는 Windows Server 2012 및 2012 R2 관리형 노드에서 PowerShell을 버전 5.1로 업데이트하는 방법을 설명합니다. 이 절차에서 제공하는 스크립트는 Windows Management

Framework(WMF) 버전 5.1 업데이트를 다운로드하고 업데이트 설치를 시작합니다. WMF 5.1을 설치할 때 필요하기 때문에 이 프로세스 중 노드가 재부팅됩니다. 업데이트 다운로드 및 설치를 완료하는 데 5분 정도 걸립니다.

Run Command를 사용하여 PowerShell을 업데이트하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택합니다.
3. Run command를(Run 명령) 선택합니다.
4. Command 문서(Command document) 목록에서 **AWS-RunPowerShellScript**를 선택합니다.
5. 명령(Commands) 섹션에 운영 체제에 대해 다음 명령을 붙여 넣습니다.

   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. 대상(Targets) 섹션에서, 태그를 지정하거나, 수동으로 인스턴스나 엣지 디바이스를 선택하거나, 리소스 그룹을 지정하여 이 작업을 실행할 관리형 노드를 식별합니다.

   > **Tip**
   >
   > 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

7. Other parameters(다른 파라미터):

   - Comment(설명)에 명령에 대한 정보를 입력합니다.
   - 제한 시간(초)에서 전체 명령 실행이 실패할 때까지 시스템이 기다리는 시간을 초 단위로 지정합니다.

8. Rate control(속도 제어)에서

   - Concurrency(동시성)에서 명령을 동시에 실행할 관리형 노드의 백분율 또는 개수를 지정합니다.

     > **Note**
     >
     > 관리형 노드에 적용할 태그를 지정하거나, AWS 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상 수를 제한합니다.

   - Error threshold(오류 임계값)에서, 명령이 노드의 개수 또는 백분율에서 실패한 후 다른 관리형 노드에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째 오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.

9. (선택 사항) Output options(출력 옵션)에서 명령 출력을 파일에 저장하려면 Write command output to an S3 bucket(S3 버킷에 명령 출력 쓰기) 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

   > **Note**
   >
   > 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아니라 인스턴스에 할당된 인스턴스 프로파일(EC2 인스턴스용) 또는 IAM 서비스 역할(하이브리드 정품 인증 시스템)의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

10. SNS notifications(SNS 알림) 섹션에서, 명령 실행 상태에 대한 알림이 전송되도록 하려면 Enable SNS notifications(SNS 알림 활성화) 확인란을 선택합니다.

    Run Command에 대한 Amazon SNS 알림 구성에 대한 자세한 내용은 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 섹션을 참조하세요.

11. Run(실행)을 선택합니다.

관리형 노드가 재부팅되고 업데이트 설치가 완료되면 관리형 노드에 연결하여 PowerShell이 버전 5.1로 업그레이드되었는지 확인합니다. 노드에서 PowerShell의 버전을 확인하려면 PowerShell을 열고 `$PSVersionTable`을 입력합니다. 출력 테이블의 `PSVersion` 값은 업그레이드가 성공한 경우 5.1을 표시합니다.

`PSVersion` 값이 5.1과 다른 경우(예: 3.0 또는 4.0) Windows 로그(Windows Logs) 아래에 있는 이벤트 뷰어의 설치(Setup) 로그를 검토합니다. 이러한 로그에서 업데이트 설치가 실패한 이유를 확인할 수 있습니다.

연습: Run Command에서 AWS CLI 사용

다음 예제 시연에서는 AWS Command Line Interface(AWS CLI)를 사용하여 명령 및 명령 파라미터에 대한 정보를 보는 방법, 명령을 실행하는 방법, 해당 명령의 상태를 보는 방법을 보여줍니다.

> **Important**
>
> 신뢰할 수 있는 관리자만 이번 주제에서 언급하는 AWS Systems Manager 사전 구성 문서를 사용할 수 있도록 허용해야 합니다. Systems Manager 문서에서 지정하는 명령 또는 스크립트는 관리형 노드에 대한 관리자 권한으로 실행됩니다. 미리 정의된 Systems Manager 문서(`AWS-`로 시작하는 모든 문서)를 실행할 권한이 있는 사용자는 해당 노드에 대한 관리자 권한도 보유합니다. 그 밖의 모든 사용자들에 대해서는 제한된 문서를 생성하여 그 문서를 특정 사용자와 공유해야 합니다.

주제

1단계: 시작하기

구성할 관리형 노드에 대한 관리자 권한이 있거나 AWS Identity and Access Management(IAM)에서 적절한 권한을 부여받아야 합니다. 또한 이 예제에서는 미국 동부(오하이오) 리전(us-east-2)을 사용합니

다. Run Command는 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 나열된 AWS 리전에서 사용할 수 있습니다. 자세한 내용은 AWS Systems Manager 설정 단원을 참조하십시오.

AWS CLI을 사용하여 명령을 실행하려면

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.
2. 사용할 수 있는 모든 문서를 나열합니다.

   이 명령을 실행하면 IAM 권한에 따라 계정에 사용할 수 있는 문서를 모두 나열합니다.

   ```
   aws ssm list-documents
   ```
3. 관리형 노드가 명령을 수신할 준비가 되었는지 확인합니다.

   다음 명령의 출력은 관리형 노드가 온라인 상태인지 여부를 보여줍니다.

   Linux & macOS

   ```
   aws ssm describe-instance-information \
       --output text --query "InstanceInformationList[*]"
   ```

   Windows

   ```
   aws ssm describe-instance-information ^
       --output text --query "InstanceInformationList[*]"
   ```
4. 다음 명령을 실행하여 특정 관리형 노드에 대한 세부 정보를 봅니다.

> **Note**
>
> 이 시연에서 명령을 실행하려면 인스턴스 및 명령 ID를 바꿉니다. 관리형 AWS IoT Greengrass 코어 디바이스에서, 인스턴스 ID로 mi-*ID_number*를 사용합니다. send-command의 응답으로 명령 ID가 반환됩니다. AWS Systems Manager의 기능인 Fleet Manager에서 인스턴스 ID를 사용할 수 있습니다.

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
```

2단계: 셸 스크립트를 실행하여 리소스 세부 정보 보기

Run Command 및 `AWS-RunShellScript` 문서를 사용하면 로컬로 로그인한 것처럼 관리형 노드에서 명령이나 스크립트를 실행할 수 있습니다.

설명 및 사용 가능한 파라미터 보기

다음 명령을 실행하여 Systems Manager JSON 문서에 대한 설명을 봅니다.

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]"
```

다음 명령을 실행하여 사용 가능한 파라미터 및 해당 파라미터에 대한 세부 정보를 봅니다.

Linux & macOS

```
aws ssm describe-document \
    --name "AWS-RunShellScript" \
    --query "Document.Parameters[*]"
```

Windows

```
aws ssm describe-document ^
    --name "AWS-RunShellScript" ^
    --query "Document.Parameters[*]"
```

3단계: **AWS-RunShellScript** 문서를 사용하여 간단한 명령 전송

다음 명령을 실행하여 관리형 노드에 대한 IP 정보를 가져옵니다.

Windows Server 관리형 노드를 대상으로 지정했다면, `document-name`을 `AWS-RunPowerShellScript`로 바꾸고 `command`를 `ifconfig`에서 `ipconfig`로 바꿉니다.

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
```

응답 데이터와 함께 명령 정보 가져오기

다음 명령은 이전 명령에서 반환된 명령 ID를 사용하여 명령 실행의 세부 정보와 응답 데이터를 가져옵니다. 명령이 완료되면 시스템에서 응답 데이터가 반환됩니다. 명령 실행에서 `"Pending"` 또는 `"InProgress"`가 표시되는 경우, 이 명령을 다시 실행하여 응답 데이터를 확인하십시오.

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
```

사용자 식별

다음 명령은 명령을 실행하는 기본 사용자를 표시합니다.

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")
```

명령 상태 가져오기

다음 명령은 명령 ID를 사용하여 관리형 노드에서 명령 실행 상태를 가져옵니다. 이 예에서는 이전 명령에서 반환된 명령 ID를 사용합니다.

Linux & macOS

```
aws ssm list-commands \
    --command-id "command-ID"
```

Windows

```
aws ssm list-commands ^
    --command-id "command-ID"
```

명령 세부 정보 가져오기

다음 명령은 이전 명령의 명령 ID를 사용하여 관리형 노드 단위로 명령 실행 상태를 가져옵니다.

Linux & macOS

```
aws ssm list-command-invocations \
    --command-id "command-ID" \
    --details
```

Windows

```
aws ssm list-command-invocations ^
    --command-id "command-ID" ^
    --details
```

특정 관리형 노드에 대한 응답 데이터로 명령 정보 가져오기

다음 명령은 특정 관리형 노드에 대한 원래 `aws ssm send-command` 요청을 반환합니다.

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
```

Python 버전 표시

다음 명령은 노드에서 실행되는 Python의 버전을 반환합니다.

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}"'
```

4단계: Run Command을 사용하여 간단한 Python 스크립트 실행

다음 명령은 Run Command을 사용하여 간단한 Python “Hello World” 스크립트를 실행합니다.

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}"'
```

5단계: Run Command를 사용하여 Bash 스크립트 실행

이 섹션의 예에서는 Run Command를 사용하여 다음 bash 스크립트를 실행하는 방법을 보여줍니다.

Run Command를 사용하여 원격 위치에 저장된 스크립트를 실행하는 예는 Amazon S3에서 스크립트 실행 및 GitHub에서 스크립트 실행를 참조하세요.

```
#!/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
```

이 스크립트는 AWS CodeDeploy User Guide의 Create an Amazon EC2 instance for CodeDeploy에 설명된 대로 Amazon Linux 및 Red Hat Enterprise Linux(RHEL) 인스턴스에 AWS CodeDeploy 에이전트를 설치합니다.

이 스크립트는 미국 동부(오하이오) 리전(us-east-2)인 `aws-codedeploy-us-east-2`의 AWS 관리형 S3 버킷에서 CodeDeploy 에이전트를 설치합니다.

AWS CLI 명령에서 bash 스크립트 실행

다음 샘플은 `--parameters` 옵션을 사용하여 CLI 명령에 bash 스크립트를 포함하는 방법을 보여줍니다.

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"]}'
```

JSON 파일에서 bash 스크립트 실행

다음 예에서는 bash 스크립트의 내용을 JSON 파일로 저장하고, `--cli-input-json` 옵션을 사용하여 해당 파일을 명령에 포함합니다.

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
```

참조되는 `installCodeDeployAgent.json` 파일의 내용이 다음 예제에 나와 있습니다.

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

연습: Run Command에서 AWS Tools for Windows PowerShell 사용

다음 예에서는 AWS Tools for Windows PowerShell를 사용하여 명령 및 명령 파라미터에 대한 정보를 보는 방법, 명령을 실행하는 방법, 해당 명령의 상태를 보는 방법을 보여 줍니다. 이 연습에는 각 사전 정의 AWS Systems Manager 문서의 예가 포함되어 있습니다.

> **Important**
>
> 신뢰할 수 있는 관리자만 이번 주제에서 언급하는 Systems Manager 사전 구성 문서를 사용할 수 있도록 허용해야 합니다. Systems Manager 문서에서 지정하는 명령 또는 스크립트는 관리형 노드에 대한 관리자 권한으로 실행됩니다. 미리 정의된 Systems Manager 문서(AWS로 시작하는 모든 문서)를 실행할 권한이 있는 사용자는 해당 노드에 대한 관리자 권한도 보유합니다. 그 밖의 모든 사용자들에 대해서는 제한된 문서를 생성하여 그 문서를 특정 사용자와 공유해야 합니다.

주제

- AWS Tools for Windows PowerShell 세션 설정 구성
- 사용할 수 있는 모든 문서 나열
- PowerShell 명령 또는 스크립트 실행
- AWS-InstallApplication 문서를 사용하여 애플리케이션 설치
- AWS-InstallPowerShellModule JSON 문서를 사용하여 PowerShell 모듈 설치

- AWS-JoinDirectoryServiceDomain JSON 문서를 사용하여 도메인에 관리형 노드 조인
- AWS-ConfigureCloudWatch 문서를 사용하여 Amazon CloudWatch Logs로 Windows 지표 전송
- AWS-UpdateEC2Config 문서를 사용하여 EC2Config 업데이트
- AWS-ConfigureWindowsUpdate 문서를 사용하여 Windows 자동 업데이트 설정 또는 해제
- Run Command를 사용하여 Windows 업데이트 관리

AWS Tools for Windows PowerShell 세션 설정 구성

자격 증명 지정

로컬 컴퓨터에서 Tools for Windows PowerShell을 열고 다음 명령을 실행하여 자격 증명을 지정합니다. 구성할 관리형 노드에 대한 관리자 권한이 있거나 AWS Identity and Access Management(IAM)에서 적절한 권한을 부여받아야 합니다. 자세한 내용은 AWS Systems Manager 설정 단원을 참조하십시오.

```
Set-AWSCredentials -AccessKey key-name -SecretKey key-name
```

기본 AWS 리전 설정

다음 명령을 실행하여 PowerShell 세션의 리전을 설정합니다. 예제에서는 미국 동부(오하이오) 리전(us-east-2)을 사용합니다. Run Command는 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 나열된 AWS 리전에서 사용할 수 있습니다.

```
Set-DefaultAWSRegion `
    -Region us-east-2
```

사용할 수 있는 모든 문서 나열

이 명령을 실행하면 계정에 사용 가능한 모든 문서가 나열됩니다.

```
Get-SSMDocumentList
```

PowerShell 명령 또는 스크립트 실행

Run Command 및 `AWS-RunPowerShell` 문서를 사용하면 로컬로 로그인한 것처럼 관리형 노드에서 명령이나 스크립트를 실행할 수 있습니다. 명령을 내리거나 로컬 스크립트의 경로를 입력하여 명령을 실행할 수 있습니다.

> **Note**
>
> Run Command를 사용할 때 서버와 관리형 노드를 재부팅하여 스크립트를 호출하는 방법은 명령 실행 시 재부팅 처리 섹션을 참조하세요.

설명 및 사용 가능한 파라미터 보기

```
Get-SSMDocumentDescription `
    -Name "AWS-RunPowerShellScript"
```

파라미터에 대한 자세한 내용 보기

```
Get-SSMDocumentDescription `
    -Name "AWS-RunPowerShellScript" | Select -ExpandProperty Parameters
```

AWS-RunPowerShellScript 문서를 사용하여 명령 전송

다음 명령은 두 개의 관리형 노드에 대해 "C:\Users" 디렉터리의 내용과 "C:\" 디렉터리의 내용을 보여줍니다.

```
$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:\')}
```

명령 요청 세부 정보 가져오기

다음 명령은 CommandId를 사용하여 두 관리형 노드에서 명령 실행 상태를 가져옵니다. 이 예에서는 이전 명령에서 반환된 CommandId를 사용합니다.

```
Get-SSMCommand `
    -CommandId $runPSCommand.CommandId
```

이 예에서 명령 상태는 성공, 보류 중 또는 진행 중일 수 있습니다.

관리형 노드당 명령 정보 가져오기

다음 명령은 이전 명령의 CommandId를 사용하여 관리형 노드 단위로 명령 실행 상태를 가져옵니다.

```
Get-SSMCommandInvocation `
    -CommandId $runPSCommand.CommandId
```

특정 관리형 노드에 대한 응답 데이터로 명령 정보 가져오기

다음 명령은 특정 관리형 노드에 대한 원래 Send-SSMCommand 출력을 반환합니다.

```
Get-SSMCommandInvocation `
    -CommandId $runPSCommand.CommandId `
    -Details $true `
    -InstanceId instance-ID | Select -ExpandProperty CommandPlugins
```

명령 취소

다음 명령은 AWS-RunPowerShellScript 문서에 대한 Send-SSMCommand를 취소합니다.

```
$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
```

명령 상태 확인

다음 명령은 Cancel 명령의 상태를 확인합니다.

```
Get-SSMCommand `
    -CommandId $cancelCommand.CommandId
```

AWS-InstallApplication 문서를 사용하여 애플리케이션 설치

Run Command 및 AWS-InstallApplication 문서를 사용하면 관리형 노드에 애플리케이션을 설치, 복구 또는 제거할 수 있습니다. 이 명령에는 MSI 경로 또는 주소가 필요합니다.

> **Note**
>
> Run Command를 사용할 때 서버와 관리형 노드를 재부팅하여 스크립트를 호출하는 방법은 명령 실행 시 재부팅 처리 섹션을 참조하세요.

설명 및 사용 가능한 파라미터 보기

```
Get-SSMDocumentDescription `
    -Name "AWS-InstallApplication"
```

파라미터에 대한 자세한 내용 보기

```
Get-SSMDocumentDescription `
    -Name "AWS-InstallApplication" | Select -ExpandProperty Parameters
```

AWS-InstallApplication 문서를 사용하여 명령 전송

다음 명령은 무인 모드로 관리형 노드에 Python 버전을 설치하고 C: 드라이브의 로컬 텍스트 파일에 출력을 로깅합니다.

```
$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'}
```

관리형 노드당 명령 정보 가져오기

다음 명령은 CommandId를 사용하여 명령 실행 상태를 가져옵니다.

```
Get-SSMCommandInvocation `
    -CommandId $installAppCommand.CommandId `
    -Details $true
```

특정 관리형 노드에 대한 응답 데이터로 명령 정보 가져오기

다음 명령은 Python 설치 결과를 반환합니다.

```
Get-SSMCommandInvocation `
    -CommandId $installAppCommand.CommandId `
    -Details $true `
    -InstanceId instance-ID | Select -ExpandProperty CommandPlugins
```

AWS-InstallPowerShellModule JSON 문서를 사용하여 PowerShell 모듈 설치

Run Command를 사용하여 관리형 노드에 PowerShell 모듈을 설치할 수 있습니다. PowerShell 모듈에 대한 자세한 내용은 Windows PowerShell Modules 섹션을 참조하세요.

설명 및 사용 가능한 파라미터 보기

```
Get-SSMDocumentDescription `
    -Name "AWS-InstallPowerShellModule"
```

파라미터에 대한 자세한 내용 보기

```
Get-SSMDocumentDescription `
    -Name "AWS-InstallPowerShellModule" | Select -ExpandProperty Parameters
```

PowerShell 모듈 설치

다음 명령은 EZOut.zip 파일을 다운로드하여 설치한 다음, 명령을 추가로 실행하여 XPS 뷰어를 설치합니다. 마지막으로 이 명령의 출력이 “demo-ssm-output-bucket”이라는 S3 버킷으로 업로드됩니다.

```
$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 demo-ssm-output-bucket
```

관리형 노드당 명령 정보 가져오기

다음 명령은 CommandId를 사용하여 명령 실행 상태를 가져옵니다.

```
Get-SSMCommandInvocation `
    -CommandId $installPSCommand.CommandId `
    -Details $true
```

특정 관리형 노드에 대한 응답 데이터로 명령 정보 가져오기

다음 명령은 특정 CommandId에 대한 원래 Send-SSMCommand 출력을 반환합니다.

```
Get-SSMCommandInvocation `
```

```
    -CommandId $installPSCommand.CommandId `
    -Details $true | Select -ExpandProperty CommandPlugins
```

AWS-JoinDirectoryServiceDomain JSON 문서를 사용하여 도메인에 관리형 노드 조인

Run Command를 사용하여 AWS Directory Service 도메인에 관리형 노드를 빠르게 조인할 수 있습니다. 이 명령을 실행하기 전에 디렉터리를 생성합니다. 또한 AWS Directory Service에 대해 자세히 알아보는 것이 좋습니다. 자세한 내용은 AWS Directory Service 관리 안내서를 참조하세요.

도메인에 관리형 노드를 조인하는 것만 가능합니다. 도메인에서 노드를 제거할 수는 없습니다.

> **Note**
>
> Run Command를 사용하여 스크립트를 호출할 관리형 노드에 대한 자세한 내용은 명령 실행 시 재부팅 처리 섹션을 참조하세요.

설명 및 사용 가능한 파라미터 보기

```
Get-SSMDocumentDescription `
    -Name "AWS-JoinDirectoryServiceDomain"
```

파라미터에 대한 자세한 내용 보기

```
Get-SSMDocumentDescription `
    -Name "AWS-JoinDirectoryServiceDomain" | Select -ExpandProperty Parameters
```

관리형 노드를 도메인에 조인

다음 명령은 관리되는 노드를 지정된 AWS Directory Service 도메인에 조인하고, 생성된 모든 아웃풋을 예시 Amazon Simple Storage Service(Amazon S3) 버킷에 업로드합니다.

```
$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 demo-ssm-output-bucket
```

관리형 노드당 명령 정보 가져오기

다음 명령은 CommandId를 사용하여 명령 실행 상태를 가져옵니다.

```
Get-SSMCommandInvocation `
    -CommandId $domainJoinCommand.CommandId `
    -Details $true
```

특정 관리형 노드에 대한 응답 데이터로 명령 정보 가져오기

이 명령은 특정 CommandId에 대한 원래 Send-SSMCommand의 출력을 반환합니다.

```
Get-SSMCommandInvocation `
    -CommandId $domainJoinCommand.CommandId `
    -Details $true | Select -ExpandProperty CommandPlugins
```

AWS-ConfigureCloudWatch 문서를 사용하여 Amazon CloudWatch Logs로 Windows 지표 전송

Windows용 이벤트 추적(ETW) 로그 및 시스템, 보안, 애플리케이션의 Windows Server 메시지를 Amazon CloudWatch Logs로 보낼 수 있습니다. 로그 기록을 처음 활성화하면 Systems Manager는 애플리케이션, 보안, ETW 로그를 업로드하기 시작한 시간부터 1분 내에 생성된 모든 로그를 전송합니다. 시작한 시간보다 1분 전에 발생한 로그는 포함되지 않습니다. 로깅을 해제했다가 나중에 다시 설정하면 Systems Manager는 중단된 시간부터 로그를 전송합니다. 사용자 정의 로그 파일 및 인터넷 정보 서비스(IIS) 로그의 경우 Systems Manager는 시작 지점부터 로그 파일을 읽습니다. 또한 Systems Manager는 성능 카운터 데이터를 CloudWatch Logs로 전송할 수도 있습니다.

이전에 EC2Config에서 CloudWatch 통합을 설정한 경우 Systems Manager 설정이 C:\Program Files\Amazon\EC2ConfigService\Settings\AWS.EC2.Windows.CloudWatch.json 파일의 관리형 노드에 로컬로 저장된 모든 설정을 재정의합니다. EC2Config를 사용하여 단일 관리형 노드에서 성능 카운터 및 로그를 관리하는 방법에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 CloudWatch 에이전트를 사용하여 Amazon EC2 인스턴스 및 온프레미스 서버로부터 지표 및 로그 수집을 참조하세요.

설명 및 사용 가능한 파라미터 보기

```
Get-SSMDocumentDescription `
    -Name "AWS-ConfigureCloudWatch"
```

파라미터에 대한 자세한 내용 보기

```
Get-SSMDocumentDescription `
    -Name "AWS-ConfigureCloudWatch" | Select -ExpandProperty Parameters
```

CloudWatch로 애플리케이션 로그 전송

다음 명령은 관리형 노드를 구성하고 Windows 애플리케이션 로그를 CloudWatch로 이동합니다.

```
$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"]}}}'}
```

관리형 노드당 명령 정보 가져오기

다음 명령은 CommandId를 사용하여 명령 실행 상태를 가져옵니다.

```
Get-SSMCommandInvocation `
    -CommandId $cloudWatchCommand.CommandId `
    -Details $true
```

특정 관리형 노드에 대한 응답 데이터로 명령 정보 가져오기

다음 명령은 Amazon CloudWatch 구성 결과를 반환합니다.

```
Get-SSMCommandInvocation `
    -CommandId $cloudWatchCommand.CommandId `
    -Details $true `
    -InstanceId instance-ID | Select -ExpandProperty CommandPlugins
```

AWS-ConfigureCloudWatch 문서를 사용하여 CloudWatch로 성능 카운터 전송

다음 명령 예는 CloudWatch로 성능 카운터를 업로드합니다. 자세한 내용은 CloudWatch 사용 설명서를 참조하세요.

```
$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"]}}}'}
```

AWS-UpdateEC2Config 문서를 사용하여 EC2Config 업데이트

Run Command 및 AWS-EC2ConfigUpdate 문서를 사용하면 Windows Server 관리형 노드에서 실행 중인 EC2Config 서비스를 업데이트할 수 있습니다. 이 명령은 EC2Config 서비스를 최신 버전 또는 사용자가 지정하는 버전으로 업데이트할 수 있습니다.

설명 및 사용 가능한 파라미터 보기

```
Get-SSMDocumentDescription `
    -Name "AWS-UpdateEC2Config"
```

파라미터에 대한 자세한 내용 보기

```
Get-SSMDocumentDescription `
    -Name "AWS-UpdateEC2Config" | Select -ExpandProperty Parameters
```

EC2Config를 최신 버전으로 업데이트

```
$ec2ConfigCommand = Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-UpdateEC2Config"
```

특정 관리형 노드에 대한 응답 데이터로 명령 정보 가져오기

이 명령은 이전의 Send-SSMCommand에서 지정된 명령의 출력을 반환합니다.

```
Get-SSMCommandInvocation `
    -CommandId $ec2ConfigCommand.CommandId `
    -Details $true `
    -InstanceId instance-ID | Select -ExpandProperty CommandPlugins
```

EC2Config를 특정 버전으로 업데이트

다음 명령은 인스턴스에서 EC2Config를 이전 버전으로 다운그레이드합니다.

```
Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-UpdateEC2Config" `
    -Parameter @{'version'='4.9.3519'; 'allowDowngrade'='true'}
```

AWS-ConfigureWindowsUpdate 문서를 사용하여 Windows 자동 업데이트 설정 또는 해제

Run Command 및 AWS-ConfigureWindowsUpdate 문서를 사용하면 Windows Server 관리형 노드에서 Windows 자동 업데이트를 설정 또는 해제할 수 있습니다. 이 명령은 Windows Update 에이전트를 구성하여 사용자가 지정하는 요일과 시간에 Windows Update를 다운로드 및 설치합니다. 업데이트에 재부팅이 필요한 경우, 업데이트가 설치되고 15분 후 관리형 노드가 자동으로 재부팅됩니다. 이 명령을 사용하여 업데이트 설치가 아닌 확인을 위해 Windows Update를 구성할 수도 있습니다. AWS-ConfigureWindowsUpdate 문서는 Windows Server 2008, 2008 R2, 2012, 2012 R2 및 2016과 호환됩니다.

설명 및 사용 가능한 파라미터 보기

```
Get-SSMDocumentDescription `
    -Name "AWS-ConfigureWindowsUpdate"
```

파라미터에 대한 자세한 내용 보기

```
Get-SSMDocumentDescription `
    -Name "AWS-ConfigureWindowsUpdate" | Select -ExpandProperty Parameters
```

Windows 자동 업데이트 설정

다음 명령은 Windows Update를 구성하여 매일 오후 10시에 업데이트를 자동으로 다운로드하고 설치합니다.

```
$configureWindowsUpdateCommand = Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-ConfigureWindowsUpdate" `
    -Parameters @{'updateLevel'='InstallUpdatesAutomatically';
 'scheduledInstallDay'='Daily'; 'scheduledInstallTime'='22:00'}
```

Windows 자동 업데이트 허용을 위한 명령 상태 보기

다음 명령은 CommandId를 사용하여 Windows 자동 업데이트 허용을 위한 명령 실행 상태를 가져옵니다.

```
Get-SSMCommandInvocation `
    -Details $true `
    -CommandId $configureWindowsUpdateCommand.CommandId | Select -ExpandProperty
 CommandPlugins
```

Windows 자동 업데이트 해제

다음 명령은 시스템에서 업데이트를 확인하지만 관리형 노드를 자동으로 업데이트하지 않도록 Windows 업데이트 알림 수준을 낮춥니다.

```
$configureWindowsUpdateCommand = Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-ConfigureWindowsUpdate" `
    -Parameters @{'updateLevel'='NeverCheckForUpdates'}
```

Windows 자동 업데이트 해제를 위한 명령 상태 보기

다음 명령은 CommandId를 사용하여 Windows 자동 업데이트 해제를 위한 명령 실행 상태를 가져옵니다.

```
Get-SSMCommandInvocation `
    -Details $true `
    -CommandId $configureWindowsUpdateCommand.CommandId | Select -ExpandProperty
 CommandPlugins
```

Run Command를 사용하여 Windows 업데이트 관리

Run Command 및 AWS-InstallWindowsUpdates 문서를 사용하면 Windows Server 관리형 노드에 대한 업데이트를 관리할 수 있습니다. 이 명령은 관리형 노드에 누락된 업데이트를 스캔하거나 설치하고 다음 설치를 선택적으로 재부팅합니다. 또한 사용자 환경에 설치할 업데이트에 대한 적절한 분류 및 심각도 수준을 지정할 수 있습니다.

> **Note**
>
> Run Command를 사용할 때 서버와 관리형 노드를 재부팅하여 스크립트를 호출하는 방법은 명령 실행 시 재부팅 처리 섹션을 참조하세요.

다음 예제에서는 지정된 Windows 업데이트 관리 작업을 수행하는 방법을 보여 줍니다.

누락된 Windows 업데이트를 모두 검색

```
Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-InstallWindowsUpdates" `
    -Parameters @{'Action'='Scan'}
```

특정 Windows 업데이트 설치

```
Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-InstallWindowsUpdates" `
    -Parameters @{'Action'='Install';'IncludeKbs'='kb-ID-1,kb-ID-2,kb-
ID-3';'AllowReboot'='True'}
```

누락된 중요 Windows 업데이트 설치

```
Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-InstallWindowsUpdates" `
    -Parameters @{'Action'='Install';'SeverityLevels'='Important';'AllowReboot'='True'}
```

누락된 Windows 업데이트를 특정 항목을 배제하고 설치

```
Send-SSMCommand `
    -InstanceId instance-ID `
    -DocumentName "AWS-InstallWindowsUpdates" `
    -Parameters @{'Action'='Install';'ExcludeKbs'='kb-ID-1,kb-
ID-2';'AllowReboot'='True'}
```

Systems Manager Run Command 문제 해결

AWS Systems Manager의 기능인 Run Command는 각 명령 실행에 대한 상태 정보를 제공합니다. 명령 상태에 대한 자세한 내용은 명령 상태 이해 섹션을 참조하세요. 이 단원의 내용을 참조하여 Run Command와 관련된 문제를 해결할 수도 있습니다.

주제

- 일부 관리형 노드가 누락됨
- 스크립트의 단계가 실패했지만 전체 상태는 'succeeded(성공)'입니다.

- SSM Agent가 제대로 실행되지 않음

일부 관리형 노드가 누락됨

명령 실행(Run a command) 페이지에서 실행할 SSM 문서를 선택하고 대상(Targets) 섹션에서 수동으로 인스턴스 선택(Manually selecting instances)을 선택한 후 명령을 실행하도록 선택할 수 있는 관리형 노드 목록이 표시됩니다.

예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

관리형 노드를 생성, 활성화, 재부팅 또는 재시작하거나, 노드에 Run Command를 설치하거나, AWS Identity and Access Management(IAM) 인스턴스 프로파일을 노드에 연결한 후, 목록에 관리형 노드가 추가되는 데 몇 분 정도 걸릴 수 있습니다.

스크립트의 단계가 실패했지만 전체 상태는 'succeeded(성공)'입니다.

Run Command를 사용하여 스크립트에서 종료 코드를 처리하는 방법을 정의할 수 있습니다. 기본적으로 스크립트에서 실행된 마지막 명령의 종료 코드는 전체 스크립트의 종료 코드로 보고됩니다. 그러나 마지막 명령 이전에 명령이 실패하는 경우 셸 조건문을 포함하여 스크립트를 종료할 수 있습니다. 자세한 내용 및 예제는 명령에 종료 코드 지정 섹션을 참조하세요.

SSM Agent가 제대로 실행되지 않음

Run Command를 사용하여 명령을 실행하는 데 문제가 있는 경우 SSM Agent에 문제가 있을 수 있습니다. SSM Agent 문제 조사에 대한 자세한 내용은 SSM Agent 문제 해결 섹션을 참조하세요.

AWS Systems Manager State Manager

AWS Systems Manager의 기능인 State Manager는 안전하고 확장 가능한 구성 관리 서비스로서 관리형 노드 및 다른 AWS 리소스를 사용자가 정의한 상태로 유지하는 프로세스를 자동화합니다. State Manager를 시작하려면 Systems Manager 콘솔을 엽니다. 탐색 창에서 State Manager를 선택합니다.

> **Note**
>
> State Manager 및 Maintenance Windows가 관리형 노드에서 몇 가지 유사한 유형의 업데이트를 수행할 수 있습니다. 어느 유형의 업데이트를 선택할지는 시스템 규정 준수를 자동화해야 하는지 아니면 지정한 기간 동안 우선순위가 높고 시간에 민감한 태스크를 수행해야 하는지 여부에 따라 다릅니다.
> 자세한 내용은 State Manager와 Maintenance Windows 중에서 선택 섹션을 참조하세요.

State Manager가 조직에 주는 이점은 무엇인가요?

사전 구성된 Systems Manager 문서(SSM 문서)를 사용하면 State Manager에서는 노드를 관리할 때 다음과 같은 이점을 제공합니다.

- 시작 시 특정 소프트웨어로 노드 부트스트랩
- 정의된 일정에 따라 SSM Agent를 포함하여 에이전트 다운로드 및 업데이트
- 네트워크 설정 구성
- 노드를 Microsoft Active Directory 도메인에 조인합니다.
- 수명 주기 동안 Linux, macOS 및 Windows 관리형 노드에서 스크립트 실행

다른 AWS 리소스 간에 구성 드리프트를 관리하기 위해서는 State Manager를 통해 Systems Manager 기능인 Automation을 사용하여 다음과 같은 유형의 작업을 수행할 수 있습니다.

- Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에 연결하여 Systems Manager 역할을 관리형 노드로 만듭니다.
- 보안 그룹에 원하는 인바운드 및 아웃바운드 규칙을 적용합니다.
- Amazon DynamoDB 백업을 생성하거나 삭제합니다.
- Amazon Elastic Block Store(Amazon EBS) 스냅샷을 생성하거나 삭제합니다.
- Amazon Simple Storage Service(Amazon S3) 버킷에 대한 읽기 및 쓰기 권한을 해제합니다.
- 관리형 노드 및 Amazon Relational Database Service(Amazon RDS) 인스턴스를 시작, 재시작 또는 중지합니다.
- Linux, macOS 및 Window AMIs를 패치합니다.

Automation 실행서를 통한 State Manager 사용 방법에 대한 자세한 내용은 State Manager 연결을 사용하여 자동화 예약 섹션을 참조하세요.

State Manager는 누가 사용해야 하나요?

State Manager는 AWS 리소스의 관리 및 거버넌스를 개선하고 구성 드리프트를 줄이려는 모든 AWS 사용자에게 적합합니다.

State Manager에는 어떤 기능이 있나요?

State Manager의 주요 기능은 다음과 같습니다.

- State Manager 연결

  State Manager 연결은 사용자가 자신의 AWS 리소스에 할당하는 구성입니다. 이러한 구성은 리소스에서 관리하려는 상태를 정의합니다. 예를 들어, 연결은 관리형 노드에서 안티바이러스 소프트웨어가 설치되어 실행 중이어야 하는지 또는 특정 포트가 닫혀 있어야 하는지를 지정할 수 있습니다.

  연결은 구성 및 연결 대상을 적용할 시점의 일정을 지정합니다. 예를 들어, 안티바이러스 소프트웨어에 대한 연결은 AWS 계정의 관리형 노드에서 하루에 한 번 실행할 수 있습니다. 노드에 소프트웨어가 설치되어 있지 않으면 연결은 State Manager에 소프트웨어를 설치하도록 지시할 수 있습니다. 소프트웨어가 설치되어 있으나 서비스가 실행 중이 아닌 경우 연결이 State Manager에 해당 서비스의 시작을 지시할 수 있습니다.

- 유연한 예약 옵션

  State Manager에서는 연결이 실행될 때 다음과 같은 예약 옵션을 제공합니다.

  - 즉각적인 처리 또는 지연된 처리

    연결을 생성하면 기본적으로 시스템은 지정된 리소스에서 연결을 즉시 실행합니다. 최초 실행 후 연결은 정의한 일정에 따른 간격으로 실행됩니다.

    콘솔에서 지정된 다음 Cron 간격에서만 연결 적용(Apply association only at the next specified Cron interval) 옵션 또는 명령줄에서 `ApplyOnlyAtCronInterval` 파라미터를 사용하여 즉시 연결을 실행하지 말라고 State Manager에 지시할 수 있습니다.

  - Cron 및 Rate 표현식

    연결을 생성하는 경우 State Manager이(가) 구성을 적용하는 시기의 일정을 지정합니다. State Manager은(는) 연결이 실행될 때 예약에 대한 가장 표준적인 cron 및 rate 표현식을 지원합니다. State Manager은(는) 또한 요일 및 숫자 기호(#)를 포함하는 cron 표현식을 지원하여 매달 n 번째 일을 지정하고, 연결 및 (L) 기호를 실행하고, 매월 마지막 X일을 지정합니다.

    > **Note**
    >
    > State Manager은(는) 현재 연결에 대한 cron 표현식에서 월 지정을 지원하지 않습니다.

    예를 들어, 화요일을 패치한 지 2일 후 연결을 실행하려는 경우 연결 실행 시기를 추가로 제어하려면 오프셋을 지정할 수 있습니다. 오프셋을 통해 연결을 실행하도록 예약된 날짜 이후에 대기할 일수를 정의할 수 있습니다.

cron 및 rate 표현식 작성에 대한 자세한 내용은 참조: Systems Manager용 Cron 및 Rate 표현식을(를) 참조하세요.

- 여러 대상 지정 옵션

  연결은 연결 대상도 지정합니다. State Manager는 태그, AWS Resource Groups, 개별 노드 ID 또는 현재 AWS 리전 및 AWS 계정의 모든 관리형 노드를 사용하여 AWS 리소스를 지원합니다.

- Amazon S3 지원

  선택한 Amazon S3 버킷에 연결 실행의 명령 출력을 저장합니다. 자세한 내용은 Systems Manager에서 연결 작업 단원을 참조하십시오.

- EventBridge 지원

  이 Systems Manager 기능은 Amazon EventBridge 규칙에서 이벤트 유형과 대상 유형으로 모두 지원됩니다. 자세한 내용은 Amazon EventBridge로 Systems Manager 이벤트 모니터링 및 참조: Systems Manager용 Amazon EventBridge 이벤트 패턴 및 유형 섹션을 참조하세요.

State Manager를 사용하는 데 비용이 듭니까?

State Manager는 추가 요금 없이 사용할 수 있습니다.

State Manager를 시작하는 방법

다음 작업을 완료하여 State Manager를 시작합니다.

작업	자세한 정보
Systems Manager 설정	AWS Systems Manager 설정
State Manager에 대해 자세히 알아보기	State Manager 정보
노드에 State Manager 연결 생성 및 할당	Systems Manager에서 연결 작업

추가 정보

- Amazon EC2 Systems Manager 및 Windows PowerShell DSC를 사용하여 구성 불일치 방지
- State Manager를 사용하여 Auto Scaling 그룹에서 Amazon EC2 인스턴스 구성

주제

State Manager 정보

AWS Systems Manager의 기능인 State Manager는 하이브리드 및 멀티클라우드 인프라에서 관리형 노드를 사용자가 정의하는 상태로 유지하는 프로세스를 자동화하는 안전하고 확장 가능한 서비스입니다.

State Manager의 작동 방식은 다음과 같습니다.

1. AWS 리소스에 적용하려는 상태를 결정합니다.

   안티바이러스 또는 멀웨어 애플리케이션 등의 특정 애플리케이션에서 관리형 노드를 구성하도록 하시겠나요? SSM Agent 또는 다른 AWS 패키지(예: `AWSPVDriver`) 업데이트 프로세스를 자동화하고 싶으신가요? 특정 포트가 열려 있거나 닫혀 있도록 해야 하나요? State Manager로 시작하려면 AWS 리소스에 적용하려는 상태를 결정합니다. 적용하려는 상태에 따라 State Manager 연결을 생성하는 데 사용할 SSM 문서가 결정됩니다.

   State Manager 연결은 사용자가 자신의 AWS 리소스에 할당하는 구성입니다. 이러한 구성은 리소스에서 관리하려는 상태를 정의합니다. 예를 들어, 연결은 관리형 노드에서 안티바이러스 소프트웨어가 설치되어 실행 중이어야 하는지 또는 특정 포트가 닫혀 있어야 하는지를 지정할 수 있습니다.

   연결은 구성 및 연결 대상을 적용할 시점의 일정을 지정합니다. 예를 들어, 안티바이러스 소프트웨어에 대한 연결은 AWS 계정의 관리형 노드에서 하루에 한 번 실행할 수 있습니다. 노드에 소프트웨어가 설치되어 있지 않으면 연결은 State Manager에 소프트웨어를 설치하도록 지시할 수 있습니다. 소프트웨어가 설치되어 있으나 서비스가 실행 중이 아닌 경우 연결이 State Manager에 해당 서비스의 시작을 지시할 수 있습니다.
2. 사전 구성된 SSM 문서가 AWS 리소스에서 원하는 상태를 생성하는 데 도움이 되는지 확인합니다.

   Systems Manager에는 연결을 생성하는 데 사용할 수 있는 수십 개의 사전 구성된 SSM 문서가 있습니다. 사전 구성된 문서는 애플리케이션 설치, Amazon CloudWatch 구성, AWS Systems Manager 자동화, PowerShell 및 셸 스크립트 실행, 관리형 노드를 Active Directory의 디렉터리 서비스 도메인에 조인 등의 일반적인 태스크를 수행할 준비가 되어 있습니다.

Systems Manager 콘솔에서 모든 SSM 문서를 볼 수 있습니다. 문서의 이름을 선택하여 각 문서에 대해 자세히 알아봅니다. `AWS-ConfigureAWSPackage`와 `AWS-InstallApplication`의 두 가지 예가 있습니다.

3. 연결을 생성합니다.

Systems Manager 콘솔, AWS Command Line Interface(AWS CLI), AWS Tools for Windows PowerShell(Tools for Windows PowerShell) 또는 Systems Manager API를 사용하여 연결을 생성할 수 있습니다. 연결을 생성할 때 다음 정보를 지정할 수 있습니다.

- 연결의 이름입니다.
- SSM 문서의 파라미터(예: 설치할 애플리케이션 경로 또는 노드에서 실행할 스크립트)입니다.
- 연결 대상. 태그를 지정하거나, 개별 인스턴스 ID를 선택하거나, AWS Resource Groups에서 그룹을 선택하여 관리형 노드를 대상으로 지정할 수 있습니다. 현재 AWS 리전 및 AWS 계정의 모든 관리형 노드를 대상으로 지정할 수도 있습니다.
- 상태를 적용할 시간 또는 빈도에 대한 일정. cron 또는 rate 표현식을 지정할 수 있습니다. cron 및 rate 표현식을 사용하여 일정 생성에 대한 자세한 내용은 연결에 대한 Cron 및 Rate 표현식 섹션을 참조하세요.

> **Note**
>
> State Manager은(는) 현재 연결에 대한 cron 표현식에서 월 지정을 지원하지 않습니다.

연결 생성을 위한 명령을 실행하면 Systems Manager는 지정한 정보(일정, 대상, SSM 문서 및 파라미터)를 대상으로 지정된 리소스로 바인딩합니다. 시스템에서 모든 대상에 연결하고 연결에 지정된 상태를 즉시 적용하려고 하기 때문에 연결 상태는 처음에 "보류 중"으로 표시됩니다.

> **Note**
>
> 이전 연결이 실행 중인 상태에서 실행하도록 예정된 새 연결을 생성하면 이전 연결 시간이 초과되고 새 연결이 실행됩니다.

Systems Manager가 리소스에서 연결을 생성하기 위해 요청 상태를 보고합니다. DescribeInstanceAssociationsStatus API 작업을 사용하여 콘솔 또는 (관리형 노드)의 상태 세부 정보를 볼 수 있습니다. 연결을 생성할 때 명령의 출력을 Amazon Simple Storage Service(Amazon S3)에 쓰기로 선택한 경우 지정한 Amazon S3 버킷에서 출력을 볼 수도 있습니다.

자세한 내용은 Systems Manager에서 연결 작업 단원을 참조하십시오.

> **Note**
>
> 연결 실행 중에 SSM 문서에 의해 시작된 API 작업은 AWS CloudTrail에 로깅되지 않습니다.

4. 모니터링 및 업데이트합니다.

   연결을 생성하면 State Manager에서 연결에 정의된 일정에 따라 구성을 다시 적용합니다. 연결 상태는 콘솔의 State Manager 페이지에서 보거나 연결 생성 시 Systems Manager가 생성한 연결 ID를 직접 호출하여 볼 수 있습니다. 자세한 내용은 연결 내역 보기 단원을 참조하십시오. 연결 문서를 업데이트하고 필요에 따라 다시 적용할 수 있습니다. 여러 버전의 연결을 생성할 수도 있습니다. 자세한 내용은 새로운 연결 버전 편집 및 생성 단원을 참조하십시오.

연결은 리소스에 언제 적용되나요?

연결을 생성할 때 구성, 대상 리소스 목록 및 구성 적용 일정을 정의하는 SSM 문서를 지정합니다. 기본적으로 State Manager에서는 연결을 생성할 때와 이후에는 일정에 따라 연결을 실행합니다. State Manager는 또한 다음과 같은 상황에서 연결을 실행하려고 시도합니다.

- 연결 편집 - State Manager는 사용자가 `DOCUMENT_VERSION`, `PARAMETERS`, `SCHEDULE_EXPRESSION`, `OUTPUT_S3_LOCATION` 연결 필드를 편집하고 변경 사항을 저장한 후 연결을 실행합니다.
- 문서 편집 - State Manager는 사용자가 연결 구성 상태를 정의하는 SSM 문서를 편집하고 변경 사항을 저장한 이후에 연결을 실행합니다. 특히 연결은 문서를 다음과 같이 편집한 후에 실행됩니다.
  - 사용자가 새 $DEFAULT 문서 버전을 지정하고 $DEFAULT 버전을 사용하여 연결을 생성했습니다.
  - 사용자가 문서를 업데이트하고 $LATEST 버전을 사용하여 연결을 생성했습니다.
  - 사용자는 연결을 생성할 때 지정된 문서를 삭제합니다.
- Parameter Store 파라미터 값 변경 - State Manager는 사용자가 연결에 정의된 파라미터 값을 편집한 후 연결을 실행합니다.
- 수동 시작 - State Manager가 사용자가 Systems Manager 콘솔에서 또는 프로그래밍 방식으로 시작한 경우에 연결을 실행합니다.
- 대상 변경 - State Manager는 대상 노드에서 다음 활동이 발생한 이후에 연결을 실행합니다.
  - 관리형 노드가 처음 온라인 상태로 전환됩니다.

- 예약된 연결 실행을 지나친 후 관리형 노드가 온라인 상태로 전환됩니다.
- 30일 넘게 중지된 경우 이후 관리형 노드가 온라인 상태로 전환됩니다.

> **Note**
>
> 대상 업데이트는 Systems Manager Automation을 사용하여 생성된 연결에 영향을 미치지 않습니다.

Systems Manager에서 연결 작업

이 섹션에서는 AWS Systems Manager 콘솔, AWS Command Line Interface(AWS CLI) 및 AWS Tools for PowerShell을 사용하여 State Manager 연결을 생성하고 관리하는 방법을 설명합니다.

주제

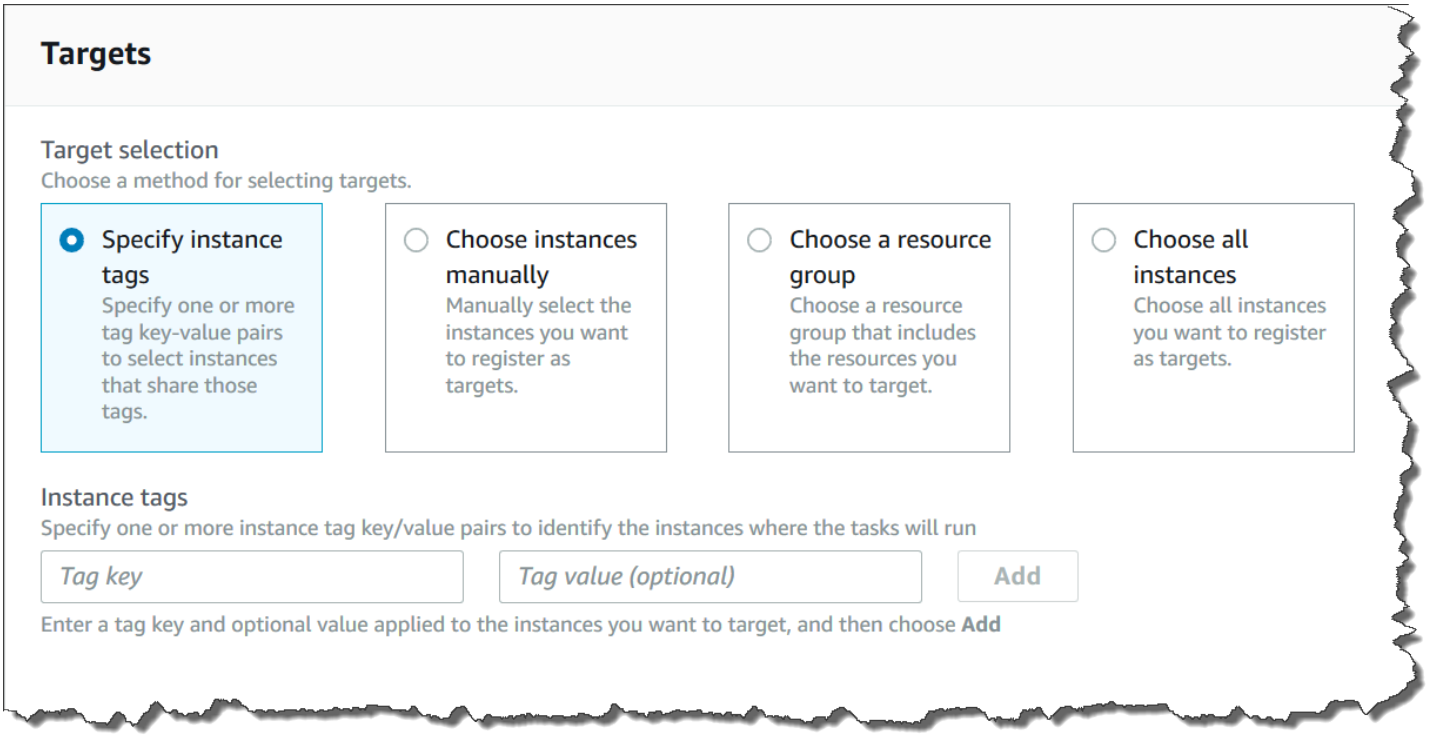
- State Manager 연결에서의 대상 및 속도 제어 정보
- 연결 생성
- 새로운 연결 버전 편집 및 생성
- 연결 삭제
- 연결을 사용하여 Auto Scaling 그룹 실행
- 연결 내역 보기
- IAM을 사용한 연결 작업

State Manager 연결에서의 대상 및 속도 제어 정보

이 주제에서는 예약된 시간에 연결을 실행하는 노드 수를 제어하면서 수십 또는 수백 개의 노드에 대한 연결을 배포하는 데 도움이 되는 AWS Systems Manager의 기능인 State Manager에 대해 설명합니다.

대상

State Manager 연결을 생성할 때 여기에 표시된 대로 Systems Manager 콘솔의 대상(Targets) 섹션에서 연결로 구성할 노드를 선택합니다.



AWS Command Line Interface(AWS CLI)와 같은 명령줄 도구를 사용하여 연결을 생성하는 경우 `targets` 파라미터를 지정합니다. 노드를 대상으로 지정하면 개별 노드 ID를 지정하거나 선택할 필요 없이 수십, 수백 또는 수천 개의 노드를 연결로 구성할 수 있습니다.

각 관리형 노드는 최대 20개 연결의 대상이 될 수 있습니다.

연결 생성 시 State Manager에는 다음과 같은 대상 옵션이 포함되어 있습니다.

태그 지정

이 옵션을 사용하여 태그 키와 노드에 지정된 태그 값(옵션)을 지정합니다. 요청을 실행하면 시스템은 지정된 태그 키 및 값과 일치하는 모든 노드를 찾아 해당 인스턴스에서 연결을 생성하려고 시도합니다. 여러 태그 값을 지정한 경우 연결은 해당 태그 값 중 적어도 하나 이상이 있는 모든 노드를 대상으로 합니다. 시스템은 처음 연결을 생성할 때 연결을 실행합니다. 이러한 최초 실행 후, 시스템은 사용자가 지정한 일정에 따라 연결을 실행합니다.

새 노드를 생성하고 지정된 태그 키 및 값을 해당 노드에 할당하면, 시스템은 자동으로 연결을 적용하고 즉시 실행하며 그 다음에는 일정에 따라 실행합니다. 이는 연결에서 Command 또는 Policy 문서를 사용하는 경우 적용되며 연결에서 Automation 실행서를 사용하는 경우에는 적용되지 않습니다. 노드에서 지정된 태그를 삭제하면 시스템은 더 이상 해당 노드에서 연결을 실행하지 않습니다.

> **Note**
>
> State Manager(으)로 Automation 실행서를 사용하며 태깅 제한으로 인해 특정 목표를 달성할 수 없는 경우 Amazon EventBridge로 Automation 실행서를 사용하는 것을 고려하세요. 자세한 내용은 이벤트를 기반으로 자동화 실행 단원을 참조하십시오. State Manager(으)로 실행서를 사용하는 방법에 대한 자세한 내용은 State Manager 연결을 사용하여 자동화 예약을(를) 참조하세요.

명령 또는 정책 문서를 사용하는 연결을 생성할 때 태그를 사용하는 것이 가장 좋습니다. 오토 스케일링을 실행하는 연결을 생성할 때 태그를 사용하는 것도 좋습니다. 자세한 내용은 연결을 사용하여 Auto Scaling 그룹 실행 단원을 참조하십시오.

> **Note**
>
> 다음 정보를 참고하세요.
>
> - 콘솔에서 연결을 생성하면서 태그를 사용하여 노드를 대상으로 지정할 때에는 태그 키를 하나만 지정할 수 있습니다. 콘솔을 사용하고 2개 이상의 태그 키를 사용하여 노드를 대상으로 지정하려면 AWS Resource Groups 그룹에 태그 키를 할당하고 노드를 추가합니다. 그러면 State Manager 연결을 생성할 때 대상 목록에서 리소스 그룹 옵션을 선택할 수 있습니다.
> - AWS CLI를 사용하여 최대 5개의 태그 키를 지정할 수 있습니다. AWS CLI를 사용하는 경우 `create-association` 명령에 지정된 모든 태그 키가 현재 노드에 할당되어 있어야 합니다. 그렇지 않으면 State Manager에서 노드를 연결 대상으로 지정하지 못합니다. 노드에 태그를 할당하는 방법에 대한 자세한 내용은 Systems Manager 리소스에 태그 지정 섹션을 참조하세요.

수동으로 노드 선택

이 옵션을 사용하면 연결을 생성하려는 노드를 수동으로 선택할 수 있습니다. 인스턴스(Instances) 창에는 현재 AWS 계정 및 AWS 리전의 모든 Systems Manager 관리형 노드가 표시됩니다. 노드를 원하는 수만큼 수동으로 선택할 수 있습니다. 시스템은 처음 연결을 생성할 때 연결을 실행합니다. 이러한 최초 실행 후, 시스템은 사용자가 지정한 일정에 따라 연결을 실행합니다.

> **Note**
>
> 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

리소스 그룹 선택

이 옵션을 사용하면 AWS Resource Groups 태그 기반 또는 AWS CloudFormation 스택 기반 쿼리에서 반환된 모든 노드에서 연결을 생성할 수 있습니다.

다음은 연결을 위해 리소스 그룹을 대상으로 지정하는 방법에 대한 세부 정보입니다.

- 새 노드를 그룹에 추가하면 시스템은 리소스 그룹을 대상으로 하는 연결에 해당 노드를 자동으로 매핑합니다. 변경 사항이 발견되면 시스템은 노드에 연결을 적용합니다. 이러한 최초 실행 후, 시스템은 사용자가 지정한 일정에 따라 연결을 실행합니다.
- 리소스 그룹을 대상으로 지정하는 연결을 생성하고 해당 그룹에 대해 `AWS::SSM::ManagedInstance` 리소스 유형을 지정된 경우 하이브리드 및 멀티클라우드 환경에서 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스와 비 EC2 노드에서 모두 연결이 실행되도록 설계되어 있습니다.
- 리소스 그룹을 대상으로 하는 연결을 생성하는 경우 리소스 그룹에 5개 이상의 태그 키가 할당되어 있거나 하나의 태그 키에 대해 5개 이상의 값이 지정되지 않아야 합니다. 이러한 조건 중 하나가 리소스 그룹에 할당된 태그 및 키에 적용되는 경우 연결이 실행되지 않고 `InvalidTarget` 오류를 반환합니다.
- 리소스 그룹을 삭제하면 해당 그룹의 모든 인스턴스는 더 이상 연결을 실행하지 않습니다. 그룹을 대상으로 하는 연결을 삭제하는 것이 좋습니다.
- 연결에 대해 단일 리소스 그룹만 대상으로 지정할 수 있습니다. 여러 그룹 또는 중첩된 그룹은 지원되지 않습니다.
- 연결을 생성한 후 State Manager는 리소스 그룹의 리소스에 대한 정보를 이용해 연결을 주기적으로 업데이트합니다. 새 리소스를 리소스 그룹에 추가하는 경우, 시스템에서 새 리소스에 연결을 적용할 때의 일정은 몇 가지 요소에 따라 다릅니다. Systems Manager 콘솔의 State Manager 페이지에서 연결 상태를 확인할 수 있습니다.

> **Warning**
>
> Amazon EC2 인스턴스의 리소스 그룹을 대상으로 하는 연결을 생성할 수 있는 권한을 가진 AWS Identity and Access Management(IAM) 사용자, 그룹 또는 역할은 해당 그룹의 모든 인스

턴스에 대한 루트 수준 제어 권한을 자동으로 갖습니다. 신뢰할 수 있는 관리자에게만 연결 생성이 허용되어야 합니다.

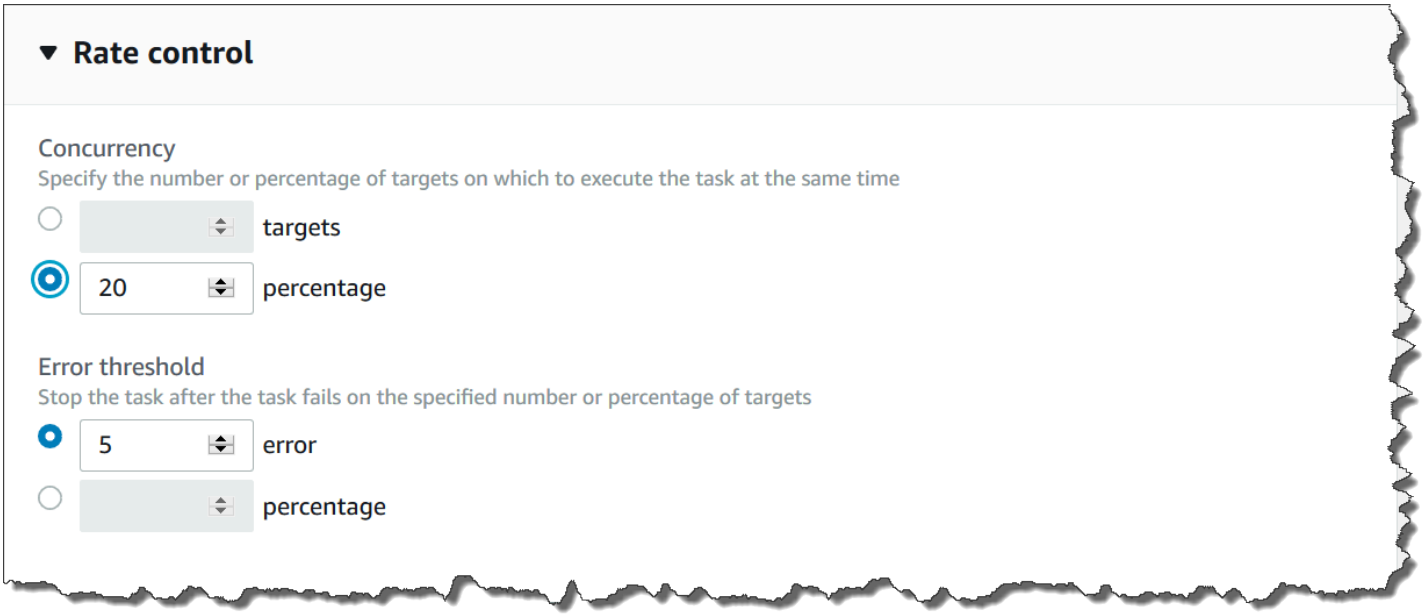
Resource Groups에 대한 자세한 내용은 AWS Resource Groups User Guide의 What Is AWS Resource Groups?를 참조하세요.

모든 노드 선택

이 옵션을 사용하면 현재 AWS 계정 및 AWS 리전의 모든 노드를 대상으로 지정할 수 있습니다. 요청을 실행하면 시스템은 현재 AWS 계정 및 AWS 리전의 모든 노드를 찾아 해당 노드에서 연결을 생성하려고 시도합니다. 시스템은 처음 연결을 생성할 때 연결을 실행합니다. 이러한 최초 실행 후, 시스템은 사용자가 지정한 일정에 따라 연결을 실행합니다. 새 노드를 생성하면, 시스템은 자동으로 연결을 적용하고 즉시 실행하며 그 다음에는 일정에 따라 실행합니다.

속도 제어

동시성 값과 오류 임계값을 지정하여 노드에서 연결 실행을 제어할 수 있습니다. 동시성 값은 연결을 동시에 실행할 수 있는 노드의 수를 지정합니다. 오류 임계값은 Systems Manager가 연결 실행을 중지하기 위해 해당 연결로 구성된 각 노드에 명령을 보내기 전에 실패할 수 있는 연결 실행 수를 지정합니다. 이 명령은 다음 예정된 실행까지 연결이 실행되는 것을 중지합니다. 동시성 및 오류 임계값 기능을 통칭하여 속도 제어라고 합니다.



동시성

동시성은 한 번에 특정한 수의 노드만 연결을 처리하도록 지정하여 노드에 대한 영향을 제한할 수 있습니다. 노드의 절대 개수(예: 20)를 지정하거나 대상 집합의 노드 비율(예: 10%)을 지정할 수 있습니다.

State Manager 동시성에는 다음과 같은 제한 사항이 있습니다.

- 대상을 사용하여 연결을 생성하도록 선택했으나 동시성 값을 지정하지 않은 경우 State Manager는 최대 동시성으로 노드 50개를 자동으로 적용합니다.
- 동시성을 사용하는 연결이 실행 중인데 대상 기준을 충족하는 새 노드가 온라인 상태가 되면 동시성 값을 초과하지 않는 경우 새 노드가 연결을 실행합니다. 동시성 값이 초과되면 현재 연결 실행 간격 중에는 새 노드가 무시됩니다. 동시성 요구 사항을 준수할 때 노드는 예약된 다음 간격 중 연결을 실행합니다.
- 동시성을 사용하는 연결을 업데이트하고 업데이트 시 하나 이상의 노드가 연결을 처리 중이면 연결을 실행 중인 모든 노드를 완료할 수 있습니다. 시작되지 않은 연결은 중지됩니다. 연결이 업데이트되었기 때문에 연결 실행이 완료되면 모든 대상 노드가 즉시 연결을 실행합니다. 연결이 다시 실행되면 동시성 값이 적용됩니다.

오류 임계값

오류 임계값은 Systems Manager가 해당 연결로 구성된 각 노드에 명령을 보내기 전에 실패가 허용되는 연결 실행 수를 지정합니다. 이 명령은 다음 예정된 실행까지 연결이 실행되는 것을 중지합니다. 오류의 절대 개수(예: 10개)를 지정하거나 대상 집합의 비율(예: 10%)을 지정할 수 있습니다.

예를 들어, 오류 절대 개수를 3으로 지정한 경우 4번째 오류가 반환되면 State Manager에서는 중지 명령을 전송합니다. 0을 지정하면 첫 번째 오류 결과가 반환된 후 State Manager는 중지 명령을 전송합니다.

예를 들어, 오류 임계값을 50개 연결의 10%로 지정한 경우 6번째 오류가 반환되면 State Manager에서는 중지 명령을 전송합니다. 오류 임계값에 도달했을 때 자동화를 이미 실행 중인 연결은 완료될 수도 있지만, 이러한 연결 중 일부가 실패할 수 있습니다. 오류 수가 오류 임계값에 지정된 수보다 많지 않도록 하려면 연결이 한 번에 하나씩 진행되도록 동시성 값을 1로 설정합니다.

State Manager 오류 임계값에는 다음과 같은 제한 및 제약이 있습니다.

- 오류 임계값은 현재 간격에 대해 적용됩니다.
- 단계 수준 세부 정보를 비롯해 각 오류에 대한 정보는 연결 내역에 기록됩니다.
- 대상을 사용하여 연결을 생성하도록 선택했으나 오류 임계값을 지정하지 않은 경우 State Manager에서는 임계값으로 100% 실패를 자동으로 적용합니다.

연결 생성

AWS Systems Manager의 기능인 State Manager는 AWS 리소스를 구성 드리프트를 정의하고 줄일 수 있는 상태로 유지하는 데 도움이 됩니다. 이렇게 하기 위해 State Manager는 연결을 사용합니다. 연결은 사용자가 자신의 AWS 리소스에 할당하는 구성입니다. 이러한 구성은 리소스에서 관리하려는 상태를 정의합니다. 예를 들어, 연결은 관리형 노드에서 안티바이러스 소프트웨어가 설치되어 실행 중이어야 하는지 또는 특정 포트가 닫혀 있어야 하는지를 지정할 수 있습니다.

연결은 구성 및 연결 대상을 적용할 시점의 일정을 지정합니다. 예를 들어, 안티바이러스 소프트웨어에 대한 연결은 AWS 계정의 관리형 노드에서 하루에 한 번 실행할 수 있습니다. 노드에 소프트웨어가 설치되어 있지 않으면 연결은 State Manager에 소프트웨어를 설치하도록 지시할 수 있습니다. 소프트웨어가 설치되어 있으나 서비스가 실행 중이 아닌 경우 연결이 State Manager에 해당 서비스의 시작을 지시할 수 있습니다.

> **Note**
>
> AWS CLI 또는 AWS Tools for PowerShell 등의 명령줄 도구를 사용하여 연결을 생성할 때 연결에 태그를 할당할 수 있습니다. Systems Manager 콘솔을 사용하여 연결에 태그를 추가하는 것은 지원되지 않습니다. 태그에 대한 자세한 내용은 Systems Manager 리소스에 태그 지정 단원을 참조하세요.

다음 절차에서는 `Command` 또는 `Policy` 문서를 사용하는 연결을 생성하여 관리형 노드를 대상 지정하는 방법에 대해 설명합니다. Automation 실행서를 사용하는 연결을 생성하여 노드 또는 다른 유형의 AWS 리소스를 대상 지정하는 방법에 대한 자세한 내용은 State Manager 연결을 사용하여 자동화 예약 섹션을 참조하세요.

연결 대상 및 속도 제어

연결은 연결을 수신해야 하는 관리형 노드 또는 대상을 지정할 수 있습니다. State Manager에는 관리형 노드를 대상으로 지정하고 해당 대상으로 연결을 배포하는 방법을 제어하는 데 도움이 되는 몇 가지 기능이 포함되어 있습니다. 대상 및 속도 제어에 대한 자세한 내용은 State Manager 연결에서의 대상 및 속도 제어 정보 섹션을 참조하세요.

연결 실행

기본적으로 State Manager는 연결을 생성한 직후 정의한 일정에 따라 실행합니다.

또한 시스템은 다음 규칙에 따라 연결을 실행합니다.

- 간격 중에 State Manager는 지정되거나 대상으로 지정된 모든 노드에서 연결을 실행하려고 시도합니다.
- (예를 들어, 동시성 값이 한 번에 연결을 처리할 수 있는 노드 수를 제한했기 때문에) 간격 동안 연결이 실행되지 않은 경우 State Manager에서는 다음 간격 중 해당 연결을 실행하려고 합니다.
- State Manager는 연결의 구성, 대상 노드, 문서 또는 파라미터 변경 후 연결을 실행합니다. 자세한 내용은 연결은 리소스에 언제 적용되나요? 단원을 참조하세요.
- State Manager는 건너 뛴 모든 간격을 기록합니다. 이러한 기록은 실행 내역 탭에서 확인할 수 있습니다.

연결 예약

10시간마다와 같은 기본 간격으로 실행되도록 연결을 예약하거나 사용자 지정 cron 및 rate 표현식을 사용하여 고급 일정을 생성할 수 있습니다. 연결을 처음 생성할 때 연결이 실행되지 않도록 할 수도 있습니다.

cron 및 rate 표현식을 사용하여 연결 실행 예약

표준 cron 또는 rate 표현식 외에도 State Manager는 또한 요일 및 숫자 기호(#)를 포함하는 cron 표현식을 지원하여 연결을 실행할 매월 n번째 일을 지정할 수 있습니다. 매월 셋째 주 화요일 UTC 23:30에 cron 일정을 실행하는 예는 다음과 같습니다.

```
cron(30 23 ? * TUE#3 *)
```

매월 두 번째 목요일 UTC 자정에 실행하는 예는 다음과 같습니다.

```
cron(0 0 ? * THU#2 *)
```

또한 State Manager는 (L) 기호를 지원하여 매월 마지막 X 일을 표시합니다. 매월 마지막 화요일 UTC 자정에 cron 일정을 실행하는 예는 다음과 같습니다.

```
cron(0 0 ? * 3L *)
```

예를 들어, 화요일을 패치한 지 2일 후 연결을 실행하려는 경우 연결 실행 시기를 추가로 제어하려면 오프셋을 지정할 수 있습니다. 오프셋을 통해 연결을 실행하도록 예약된 날짜 이후에 대기할 일 수를 정의할 수 있습니다. 예를 들어, `cron(0 0 ? * THU#2 *)`의 cron 일정을 지정한 경우 일정 오프셋 필드에 숫자 3을 지정하여 매월 두 번째 목요일 이후 매주 일요일에 연결을 실행할 수 있습니다.

> **Note**
>
> 오프셋을 사용하려면 콘솔에서 지정된 다음 Cron 간격에서만 연결 적용을 선택하거나 명령줄에서 `ApplyOnlyAtCronInterval` 파라미터를 지정해야 합니다. 이러한 옵션 중 하나가 활성화되면 연결을 생성한 직후 State Manager가 연결을 실행하지 않습니다.

Cron 및 Rate 표현식에 대한 자세한 내용은 참조: Systems Manager용 Cron 및 Rate 표현식 섹션을 참조하세요.

연결 생성(콘솔)

다음 절차에서는 Systems Manager 콘솔을 사용하여 State Manager 연결을 생성하는 방법을 설명합니다.

> **Warning**
>
> 연결을 생성하는 경우 연결의 대상으로 관리형 노드의 AWS 리소스 그룹을 선택할 수 있습니다. AWS Identity and Access Management(IAM) 사용자, 그룹 또는 역할이 관리형 노드의 리소스 그룹을 대상으로 하는 연결을 생성할 수 있는 권한을 가진 경우, 해당 사용자, 그룹 또는 역할에는 그룹의 모든 노드에 대한 루트 수준 제어 권한이 자동으로 생깁니다. 신뢰할 수 있는 관리자에게만 연결 생성이 허용됩니다.

State Manager 연결 생성

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 State Manager를 선택합니다.
3. 연결 생성을 선택합니다.
4. 이름(Name) 필드에 이름을 지정합니다.
5. 문서 목록에서 문서 이름 옆에 있는 옵션을 선택합니다. 문서 유형을 적어둡니다. 이 절차는 `Command` 및 `Policy` 문서에 적용됩니다. Automation 런북을 사용하는 연결 생성에 대한 자세한 내용은 State Manager 연결을 사용하여 자동화 예약 섹션을 참조하세요.

   > **Important**
   >
   > State Manager는 해당 문서가 다른 계정에서 공유되는 경우 새 버전의 문서를 사용하는 연결 실행을 지원하지 않습니다. Systems Manager 콘솔이 새로운 버전이 처리되었음을

보여주더라도 상태 관리자는 언제나 다른 계정에서 공유된 문서의 `default` 버전을 실행합니다. 다른 계정에서 새 버전의 문서 공유 양식을 사용하여 연결을 실행하려면 문서 버전을 `default`로 설정해야 합니다.

6. 파라미터에서 필요한 입력 파라미터를 지정합니다.
7. (선택 사항) 모니터링을 위해 연결에 적용할 CloudWatch 경보를 선택합니다.

> **Note**
>
> 이 단계에 대한 다음 정보를 참조하십시오.
>
> - 알람 목록에는 최대 100개의 알람이 표시됩니다. 목록에 해당 경보가 표시되지 않으면 AWS Command Line Interface를 사용하여 연결을 생성합니다. 자세한 내용은 연결 생성(명령줄) 단원을 참조하십시오.
> - CloudWatch 경보를 명령에 연결하려면 연결을 생성하는 IAM 보안 주체에 `iam:createServiceLinkedRole` 작업에 대한 권한이 있어야 합니다. CloudWatch 경보에 대한 자세한 내용은 Amazon CloudWatch 경보 사용을 참조하세요.
> - 경보가 활성화되면 보류 중인 명령 호출 또는 자동화가 실행되지 않습니다.

8. 대상에서 옵션을 선택합니다. 태그 사용에 대한 자세한 내용은 State Manager 연결에서의 대상 및 속도 제어 정보 섹션을 참조하세요.
9. 일정 지정 섹션에서 On Schedule(일정이 있을 때) 또는 No schedule(일정이 없을 때)을 선택합니다. On Schedule(일정이 있을 때)을 선택한 경우 제공된 버튼을 사용하여 연결에 대한 cron 또는 rate 일정을 생성합니다.

   연결을 생성한 직후에 연결을 실행하지 않으려면 지정된 다음 Cron 간격에서만 연결 적용을선택합니다.
10. (선택 사항) 일정 오프셋(Schedule offset) 필드에 1에서 6 사이의 숫자를 지정합니다.
11. [고급 옵션(Advanced options)] 섹션에서 [규정 준수 심각도(Compliance severity)]를 사용하여 연결의 심각도 수준을 선택하고 [변경 일정(Change Calendars)]을 사용하여 연결의 변경 일정을 선택합니다.

    규정 준수 보고는 여기서 지정한 심각도 수준과 함께 연결 상태가 준수인지 아니면 미준수인지를 나타냅니다. 자세한 내용은 State Manager 연결 규정 준수 정보 단원을 참조하십시오.

변경 일정에 따라 연결 실행 시기가 결정됩니다. 일정이 닫혀 있으면 연결이 적용되지 않습니다. 일정이 열려 있으면 그에 따라 연결이 실행됩니다. 자세한 내용은 AWS Systems Manager Change Calendar 단원을 참조하십시오.

12. 속도 제어(Rate control) 섹션의 여러 노드에서 연결을 실행하는 방법을 제어하는 옵션을 선택합니다. 속도 제어 사용에 대한 자세한 내용은 State Manager 연결에서의 대상 및 속도 제어 정보 섹션을 참조하세요.

    동시성 섹션에서 옵션을 선택합니다.

    - 대상을 선택하여 연결을 동시에 실행할 수 있는 대상 수(절대 개수)를 입력합니다.
    - 백분율을 선택하여 연결을 동시에 실행할 수 있는 대상의 백분율을 입력합니다.

    오류 임계값 섹션에서 옵션을 선택합니다.

    - 오류를 선택하여 State Manager에서 추가 대상에 대한 연결 실행을 중지하기 전에 허용되는 절대 오류 수를 입력합니다.
    - 백분율을 선택하여 State Manager에서 추가 대상에 대한 연결 실행을 중지하기 전에 허용되는 오류 비율을 입력합니다.

13. (선택 사항) 출력 옵션에서 명령 출력을 파일에 저장하려면 S3 버킷에 쓰기 활성화 옆의 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

    > **Note**
    >
    > 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아닌 관리형 노드에 할당된 인스턴스 프로파일의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

    다음은 연결에 대해 Amazon S3 출력을 설정하는 데 필요한 최소 권한입니다. IAM 정책을 계정 내의 사용자 또는 역할에 연결하여 액세스를 추가로 제한할 수 있습니다. 최소한 Amazon EC2 인스턴스 프로파일에는 `AmazonSSMManagedInstanceCore` 관리형 정책과 다음 인라인 정책이 포함된 IAM 역할이 있어야 합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:PutObjectAcl"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
        }
    ]
}
```

최소 권한을 위해 내보내는 Amazon S3 버킷에 Amazon S3 콘솔에서 정의한 기본 설정이 있어야 합니다. Amazon S3 버킷 생성에 대한 자세한 내용은 Amazon S3 사용 설명서의 버킷 생성을 참조하세요.

> **Note**
>
> 연결 실행 중에 SSM 문서에 의해 시작된 API 작업은 AWS CloudTrail에 로깅되지 않습니다.

14. 연결 생성을 선택합니다.

> **Note**
>
> 생성한 연결을 삭제하면 연결은 더 이상 해당 연결의 대상에서 실행되지 않습니다.

연결 생성(명령줄)

다음 절차에서는 AWS CLI(Linux 또는 Windows) 또는 Tools for PowerShell을 사용하여 State Manager 연결을 생성하는 방법을 설명합니다. 이 섹션에는 대상 및 속도 제어를 사용하는 방법을 보여주는 몇 가지 예제가 포함되어 있습니다. 대상 및 속도 제어를 사용하면 수십 또는 수백 개의 노드에 연결을 할당하면서 이러한 연결의 실행을 제어할 수 있습니다. 대상 및 속도 제어에 대한 자세한 내용은 State Manager 연결에서의 대상 및 속도 제어 정보 섹션을 참조하세요.

시작하기 전 준비 사항

targets 파라미터는 지정한 Key-Value 조합을 사용하여 노드를 대상으로 지정하는 검색 조건의 배열입니다. targets 파라미터를 사용하여 수십 또는 수백 개의 노드에서 연결을 생성하려는 경우, 절차를 시작하기 전에 다음의 대상 지정 옵션을 검토하세요.

ID를 지정하여 특정 노드를 대상으로 지정

```
--targets Key=InstanceIds,Values=instance-id-1,instance-id-2,instance-id-3
```

```
--targets
 Key=InstanceIds,Values=i-02573cafcfEXAMPLE,i-0471e04240EXAMPLE,i-07782c72faEXAMPLE
```

태그를 사용하여 인스턴스를 대상으로 지정

```
--targets Key=tag:tag-key,Values=tag-value-1,tag-value-2,tag-value-3
```

```
--targets Key=tag:Environment,Values=Development,Test,Pre-production
```

AWS Resource Groups를 사용하여 노드 대상 지정

```
--targets Key=resource-groups:Name,Values=resource-group-name
```

```
--targets Key=resource-groups:Name,Values=WindowsInstancesGroup
```

현재 AWS 계정 및 AWS 리전의 모든 인스턴스를 대상으로 지정

```
--targets Key=InstanceIds,Values=*
```

> **Note**
>
> 다음 정보를 참고하세요.
>
> - State Manager는 해당 문서가 다른 계정에서 공유되는 경우 새 버전의 문서를 사용하는 연결 실행을 지원하지 않습니다. Systems Manager 콘솔이 새로운 버전이 처리되었음을 보여주더라도 상태 관리자는 언제나 다른 계정에서 공유된 문서의 default 버전을 실행합

니다. 다른 계정에서 새 버전의 문서 공유 양식을 사용하여 연결을 실행하려면 문서 버전을 `default`로 설정해야 합니다.

- AWS CLI를 사용하여 최대 5개의 태그 키를 지정할 수 있습니다. AWS CLI를 사용하는 경우 `create-association` 명령에 지정된 모든 태그 키가 현재 노드에 할당되어 있어야 합니다. 그렇지 않으면 State Manager에서 노드를 연결 대상으로 지정하지 못합니다. 노드에 태그를 할당하는 방법에 대한 자세한 내용은 Systems Manager 리소스에 태그 지정 섹션을 참조하세요.
- 연결을 생성할 때 언제 일정을 실행할지 지정합니다. Cron 및 Rate 표현식을 사용하여 일정을 지정합니다. Cron 및 Rate 표현식에 대한 자세한 내용은 연결에 대한 Cron 및 Rate 표현식 섹션을 참조하세요.

연결을 생성하려면

1. 아직 하지 않은 경우 AWS CLI 또는 AWS Tools for PowerShell을 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트 및 AWS Tools for PowerShell 설치를 참조하세요.

2. 다음 형식을 사용하여 State Manager 연결을 생성하는 명령을 만듭니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   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"
```

다음 예제에서는 "Environment,Linux" 태그가 지정된 노드에 대해 연결을 생성합니다. 이 연결은 AWS-UpdateSSMAgent 문서를 사용하여 매주 일요일 오전 2시(협정 세계 표준시(UTC))에 대상 노드에 대해 SSM Agent를 업데이트합니다. 이 연결은 주어진 시간에 최대 10개의 노드에서 동시에 실행됩니다. 또한 이 연결은 오류 수가 5를 초과하는 경우 특정 실행 간격에 대해 더 많은 노드에서 실행을 중지합니다. 규정 준수 보고를 위해 이 연결에는 중간 심각도 수준이 할당됩니다.

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
```

다음 예는 와일드카드 값(*)을 지정하여 노드 ID를 대상으로 지정합니다. 이를 통해 Systems Manager는 현재 AWS 계정 및 AWS 리전의 모든 노드에 대한 연결을 생성할 수 있습니다. 이 연결은 주어진 시간에 최대 10개의 노드에서 동시에 실행됩니다. 또한 이 연결은 오류 수가 5를 초과하는 경우 특정 실행 간격에 대해 더 많은 노드에서 실행을 중지합니다. 규정 준수 보고를 위해 이 연결에는 중간 심각도 수준이 할당됩니다. 이 연결에서는 일정 오프셋을 사용합니다. 즉, 지정된 cron 일정 2일 후에 실행됩니다. 또한 `ApplyOnlyAtCronInterval` 파라미터를 포함하며, 이 파라미터는 일정 오프셋을 사용하는 데 필요합니다. 즉, 연결은 생성된 직후에는 실행되지 않습니다.

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
```

다음 예에서는 Resource Groups의 노드에 연결을 생성합니다. 그룹 이름은 "HR-Department"입니다. 이 연결은 AWS-UpdateSSMAgent 문서를 사용하여 매주 일요일 오전 2시(협정 세계 표준시(UTC))에 대상 노드에 대해 SSM Agent를 업데이트합니다. 이 연결은 주어진 시간에 최대 10개의 노드에서 동시에 실행됩니다. 또한 이 연결은 오류 수가 5를 초과하는 경우 특정 실행 간격에 대해 더 많은 노드에서 실행을 중지합니다. 규정 준수 보고를 위해 이 연결에는 중간 심각도 수준이 할당됩니다. 이 연결은 지정된 cron 일정에서 실행됩니다. 연결이 생성된 직후에는 실행되지 않습니다.

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
```

다음 예에서는 특정 노드 ID로 태그가 지정된 노드에서 실행되는 연결을 생성합니다. 연결은 변경 일정이 열려 있을 때 SSM Agent 문서를 사용하여 대상 노드에서 SSM Agent를 한 번 업데이트합니다. 연결은 실행 시 일정 상태를 확인합니다. 시작 시 일정이 닫혀 있고 연결이 한 번만 실행되면 연결 실행 기간이 지나므로 열결이 다시 실행되지 않습니다. 일정이 열려 있으면 그에 따라 연결이 실행됩니다.

> **Note**
>
> 변경 일정이 닫힐 때 연결이 작동하는 태그 또는 리소스 그룹에 새 노드를 추가하는 경우 변경 일정이 열리면 연결이 해당 노드에 적용됩니다.

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)"
```

다음 예에서는 특정 노드 ID로 태그가 지정된 노드에서 실행되는 연결을 생성합니다. 이 연결은 SSM Agent 문서를 사용하여 매주 일요일 오전 2시에 대상 노드에 대해 SSM Agent를 업데이트합니다. 이 연결은 변경 일정이 열려 있을 때 지정된 cron 일정에서만 실행됩니다. 연결은 생성 시 일정 상태를 확인합니다. 일정이 닫혀 있으면 연결이 적용되지 않습니다. 연결 적용 간격이 일요일 오전 2시에 시작되면 연결은 달력이 열려 있는지 확인합니다. 일정이 열려 있으면 그에 따라 연결이 실행됩니다.

> **Note**
>
> 변경 일정이 닫힐 때 연결이 작동하는 태그 또는 리소스 그룹에 새 노드를 추가하는 경우 변경 일정이 열리면 연결이 해당 노드에 적용됩니다.

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**
>
> 생성한 연결을 삭제하면 연결은 더 이상 해당 연결의 대상에서 실행되지 않습니다. 또한 `apply-only-at-cron-interval` 파라미터를 지정한 경우 이 옵션을 재설정할 수 있습니다. 이렇게 하려면 명령줄에서 연결을 업데이트할 때 `no-apply-only-at-cron-interval` 파라미터를 지정합니다. 이 파라미터는 연결을 업데이트한 직후 지정된 간격에 따라 연결을 강제로 실행합니다.

새로운 연결 버전 편집 및 생성

State Manager 연결을 편집하여 새 이름, 일정, 심각도 수준 또는 대상을 지정할 수 있습니다. 명령의 출력을 Amazon Simple Storage Service(Amazon S3) 버킷에 쓰도록 선택할 수도 있습니다. 연결을 편집하면 State Manager에서 새 버전을 생성합니다. 다음 절차에 설명된 대로 편집 후에 다른 버전이 표시될 수 있습니다.

다음 절차에서는 Systems Manager 콘솔, AWS Command Line Interface(AWS CLI) 및 AWS Tools for PowerShell(Tools for PowerShell)을 사용하여 새로운 연결 버전을 편집하고 생성하는 방법을 설명합니다.

> **Important**
>
> State Manager는 해당 문서가 다른 계정에서 공유되는 경우 새 버전의 문서를 사용하는 연결 실행을 지원하지 않습니다. State Manager는 Systems Manager 콘솔이 새 버전이 처리되었음을 보여주더라도 다른 계정에서 공유된 경우 문서의 `default` 버전을 항상 실행합니다. 다른 계정에서 새 버전의 문서 공유 양식을 사용하여 연결을 실행하려면 문서 버전을 `default`로 설정해야 합니다.

연결 편집(콘솔)

다음 절차에서는 Systems Manager 콘솔을 사용하여 새로운 연결 버전을 편집하고 생성하는 방법을 설명합니다.

> **Note**
>
> 이 절차를 수행하려면 기존 Amazon S3 버킷에 대한 쓰기 액세스 권한이 있어야 합니다. 이전에 Amazon S3를 사용해 본 적이 없는 경우 Amazon S3 사용 요금이 부과되므로 주의하세요. 버킷을 생성하는 방법에 대한 자세한 내용은 버킷 생성 섹션을 참조하세요.

State Manager 연결을 편집하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 State Manager를 선택합니다.
3. 연결 생성(명령줄)에서 생성한 연결을 선택한 후 [편집(Edit)]을 선택합니다.
4. [이름(Name)] 필드에 새 이름을 입력합니다.
5. 일정 지정 섹션에서 새 옵션을 선택합니다.
6. (선택 사항) 출력 옵션에서 명령 출력을 파일에 저장하려면 S3 버킷에 쓰기 활성화 옆의 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

   > **Note**
   >
   > 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아닌 관리형 노드에 할당된 인스턴스 프로파일의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

7. Edit association(연결 편집)을 선택합니다. 현재 요구 사항을 충족하도록 연결을 구성합니다.
8. [연결(Associations)] 페이지에서 편집한 연결의 이름을 선택한 후 [버전(Versions)] 탭을 선택합니다. 생성하고 편집한 연결의 각 버전이 시스템에 나열됩니다.
9. https://console.aws.amazon.com/s3/에서 Amazon S3 콘솔을 엽니다.
10. 명령 출력을 저장하도록 지정한 Amazon S3 버킷의 이름을 선택한 후, 연결을 실행한 노드의 ID로 이름을 지정한 폴더를 선택합니다. (버킷의 폴더에 출력을 저장하도록 선택한 경우, 해당 폴더를 먼저 여십시오.)
11. 몇 가지 수준을 드릴다운하여 `awsrunPowerShell` 폴더의 `stdout` 파일을 찾습니다.
12. 열기 또는 다운로드를 선택하여 호스트 이름을 확인합니다.

연결 편집(명령줄)

다음 절차에서는 AWS CLI(Linux 또는 Windows) 또는 AWS Tools for PowerShell을 사용하여 새로운 연결 버전을 편집 및 생성하는 방법을 설명합니다.

State Manager 연결을 편집하려면

1. 아직 하지 않은 경우 AWS CLI 또는 AWS Tools for PowerShell을 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트 및 AWS Tools for PowerShell 설치를 참조하세요.

2. 다음 형식을 사용하여 기존 State Manager 연결의 새로운 버전을 편집 및 생성할 명령을 만듭니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   > **Important**
   >
   > UpdateAssociation을 호출할 때, 시스템은 요청의 모든 부가적인 파라미터를 드롭하고 이 파라미터를 null 값으로 덮어씁니다. 이것은 설계에 따른 것입니다. 파라미터를 변경하지 않더라도 호출에서 모든 선택적 파라미터를 지정해야 합니다. 여기에는 Name 파라미터가 포함됩니다. 이 API 작업을 호출하기 전에, DescribeAssociation API 작업을 호출하고 UpdateAssociation 호출에 필요한 모든 부가적인 파라미터를 기억하는 것이 좋습니다.

   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
```

다음 예제에서는 이름을 TestHostnameAssociation2로 변경하도록 기존 연결을 업데이트합니다. 새로운 연결 버전은 매 시간마다 실행되고 명령의 출력을 지정된 Amazon S3 버킷에 씁니다.

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=DOC-EXAMPLE-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=DOC-EXAMPLE-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 DOC-EXAMPLE-BUCKET `
  -S3Location_OutputS3KeyPrefix logs `
  -S3Location_OutputS3Region us-east-1 `
  -ScheduleExpression "cron(0 */1 * * ? *)"
```

다음 예제에서는 이름을 CalendarAssociation로 변경하도록 기존 연결을 업데이트합니다. 새로운 연결은 달력이 열릴 때 실행되고 명령 출력을 지정된 Amazon S3 버킷에 씁니다.

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=DOC-EXAMPLE-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=DOC-EXAMPLE-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 DOC-EXAMPLE-BUCKET `
  -CalendarNames "arn:aws:ssm:us-east-1:123456789012:document/testCalendar2"
```

다음 예제에서는 이름을 MultiCalendarAssociation로 변경하도록 기존 연결을 업데이트합니다. 새로운 연결은 달력이 열려 있을 때 실행되고 명령 출력을 지정된 Amazon S3 버킷에 씁니다.

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=DOC-EXAMPLE-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=DOC-EXAMPLE-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 DOC-EXAMPLE-BUCKET `
  -CalendarNames "arn:aws:ssm:us-east-1:123456789012:document/testCalendar1"
 "arn:aws:ssm:us-east-2:123456789012:document/testCalendar2"
```

3. 새로운 연결 버전을 보려면 다음 명령을 실행합니다.

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 *
```

시스템은 다음과 같은 정보를 반환합니다.

Linux & macOS

```
{
    "AssociationDescription": {
        "ScheduleExpression": "cron(0 */1 * * ? *)",
        "OutputLocation": {
            "S3Location": {
                "OutputS3KeyPrefix": "logs",
                "OutputS3BucketName": "DOC-EXAMPLE-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": "DOC-EXAMPLE-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}
```

연결 삭제

다음 절차에서는 AWS Systems Manager 콘솔을 사용하여 State Manager 연결을 삭제하는 방법을 설명합니다.

연결 삭제

다음 절차에 따라 AWS Systems Manager 콘솔을 사용하여 연결을 삭제합니다.

연결을 삭제하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 State Manager를 선택합니다.
3. 연결을 선택하고 삭제를 선택합니다.

연결을 사용하여 Auto Scaling 그룹 실행

연결을 사용하여 Auto Scaling 그룹을 실행할 때 모범 사례는 태그 대상을 사용하는 것입니다. 태그를 사용하지 않으면 연결 제한에 도달하게 될 수 있습니다.

모든 노드에 동일한 키와 값으로 태그가 지정된 경우 Auto Scaling 그룹을 실행하는 데 하나의 연결만 필요합니다. 다음 절차에서는 이러한 연결을 생성하는 방법을 설명합니다.

Auto Scaling 그룹을 실행하는 연결을 생성하려면

1. Auto Scaling 그룹의 모든 노드에 동일한 키와 값으로 태그가 지정되었는지 확인합니다. 노드에 태그를 지정하는 자세한 지침은 AWS Auto Scaling 사용 설명서의 Auto Scaling 그룹 및 인스턴스 태그 지정을 참조하세요.
2. Systems Manager에서 연결 작업의 절차를 사용하여 연결을 생성합니다.

   콘솔에서 작업 중인 경우 [대상(Targets)] 필드에서 [인스턴스 태그 지정(Specify instance tags)]을 선택합니다. [인스턴스 태그(Instance tags)]에 Auto Scaling 그룹의 [태그(Tag)] 키와 값을 입력합니다.

   AWS Command Line Interface(AWS CLI)를 사용하는 경우 키와 값이 노드에 태그를 지정하는 데 사용한 것과 일치하는 `--targets Key=tag:tag-key,Values=tag-value`를 지정합니다.

연결 내역 보기

DescribeAssociationExecutions API 작업을 사용하여 특정 연결 ID에 대한 모든 실행을 볼 수 있습니다. 이 작업을 사용하여 State Manager 연결의 상태, 세부 상태, 결과, 마지막 실행 시간 및 추가 정보를 봅니다. State Manager는 AWS Systems Manager의 기능입니다. 이 API 작업에는 지정한 기준에 따라 연결을 찾는 데 도움이 되는 필터도 포함됩니다. 예를 들어 정확한 날짜 및 시간을 지정하고 GREATER_THAN 필터를 사용하여 지정한 날짜 및 시간 이후에 처리된 실행을 볼 수 있습니다.

예를 들어 연결 실행에 실패한 경우 DescribeAssociationExecutionTargets API 작업을 사용하여 특정 실행의 세부 정보를 심층 분석할 수 있습니다. 이 작업은 연결이 실행된 노드 ID와 같은 리소스 및 다양한 연결 상태를 보여줍니다. 그러면 연결 실행에 실패한 리소스나 노드를 확인할 수 있습니다. 그런 다음 리소스 ID를 사용하여 명령 실행 세부 정보를 보고 명령의 어떤 단계가 실패했는지를 확인할 수 있습니다.

이 섹션의 예에는 StartAssociationsOnce API 작업을 사용하여 생성 시 한 번 연결을 실행하는 방법에 대한 정보도 포함되어 있습니다. 실패한 연결 실행을 조사할 때 이 API 작업을 사용할 수 있습니다. 연

결이 실패했음을 확인하는 경우 리소스를 변경한 다음 연결을 즉시 실행하여 리소스 변경으로 인해 연결을 성공적으로 실행할 수 있는지 여부를 확인할 수 있습니다.

> **Note**
>
> 연결 실행 중에 SSM 문서에 의해 시작된 API 작업은 AWS CloudTrail에 로깅되지 않습니다.

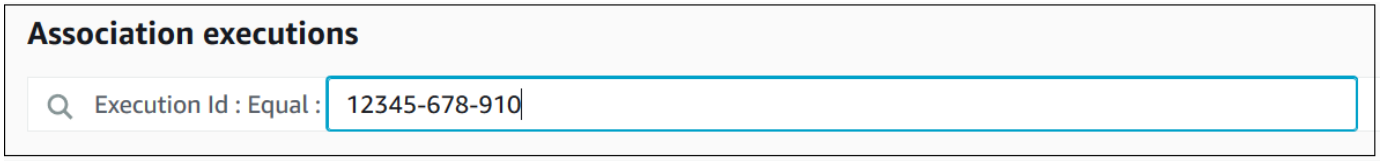
연결 내역 보기(콘솔)

다음 절차를 사용하여 특정 연결 ID에 대한 실행 내역을 본 다음 하나 이상의 리소스에 대한 실행 세부 정보를 봅니다.

특정 연결 ID에 대한 실행 내역을 보려면

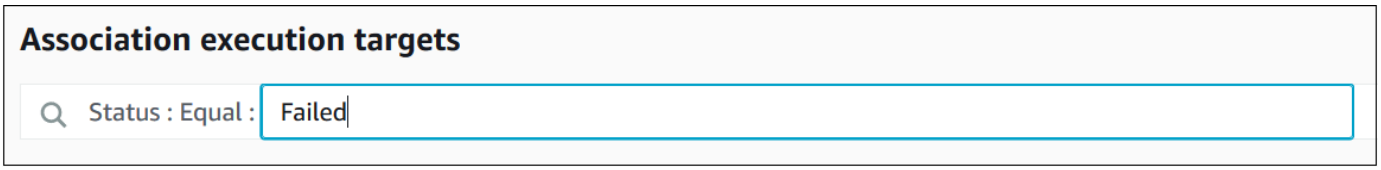
1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. State Manager를 선택합니다.
3. Association id(연결 ID) 필드에서 내역을 보려는 연결을 선택합니다.
4. 세부 정보 보기(View details) 버튼을 선택합니다.
5. Execution history(실행 내역) 탭을 선택합니다.
6. 리소스 레벨 실행 세부 정보를 보려는 연결을 선택합니다. 예를 들어, Failed(실패) 상태를 보여 주는 연결을 선택합니다. 그런 다음 연결을 실행하는 데 실패한 노드에 대한 실행 세부 정보를 볼 수 있습니다.

   검색 상자 필터를 사용하여 세부 정보를 보려는 실행을 찾습니다.

7. 실행 ID를 선택합니다. Association execution targets(연결 실행 대상) 페이지가 열립니다. 이 페이지에는 연결을 실행한 모든 리소스가 표시됩니다.
8. 리소스 ID를 선택하여 해당 리소스에 대한 구체적인 정보를 봅니다.

   검색 상자 필터를 사용하여 세부 정보를 보려는 리소스를 찾습니다.

9. 실행에 실패한 연결을 조사하는 경우 [지금 연결 적용(Apply association now)] 버튼을 사용하여 생성 시 연결을 한 번 실행할 수 있습니다. 연결 실행에 실패한 리소스를 변경한 후 탐색 이동 경로에서 Association ID(연결 ID) 랭크를 선택합니다.
10. Apply association now(지금 연결 적용) 버튼을 선택합니다. 실행이 완료된 후 해당 연결 실행이 성공했는지를 확인합니다.

연결 내역 보기(명령줄)

다음 절차에서는 AWS Command Line Interface(AWS CLI)(Linux 또는 Windows) 또는 AWS Tools for PowerShell을 사용하여 특정 연결 ID에 대한 실행 내역을 보는 방법을 설명합니다. 그런 다음, 하나 이상의 리소스에 대한 실행 세부 정보를 보는 방법을 설명합니다.

특정 연결 ID에 대한 실행 내역을 보려면

1. 아직 하지 않은 경우 AWS CLI 또는 AWS Tools for PowerShell을 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트 및 AWS Tools for PowerShell 설치를 참조하세요.
2. 다음 명령을 실행하여 특정 연결 ID의 작업 실행 목록을 봅니다.

   Linux & macOS

   ```
   aws ssm describe-association-executions \
     --association-id ID \
     --filters Key=CreatedTime,Value="2018-04-10T19:15:38.372Z",Type=GREATER_THAN
   ```

   > **Note**
   >
   > 이 명령에는 특정 날짜 및 시간 이후에 발생한 실행으로만 결과를 제한하는 필터가 포함됩니다. 특정 연결 ID의 모든 실행을 보려는 경우 `--filters` 파라미터 및 `Key=CreatedTime,Value="2018-04-10T19:15:38.372Z",Type=GREATER_THAN` 값을 제거합니다.

   Windows

   ```
   aws ssm describe-association-executions ^
     --association-id ID ^
   ```

```
  --filters Key=CreatedTime,Value="2018-04-10T19:15:38.372Z",Type=GREATER_THAN
```

> **Note**
>
> 이 명령에는 특정 날짜 및 시간 이후에 발생한 실행으로만 결과를 제한하는 필터가 포함됩니다. 특정 연결 ID의 모든 실행을 보려는 경우 `--filters` 파라미터 및 `Key=CreatedTime,Value="2018-04-10T19:15:38.372Z",Type=GREATER_THAN` 값을 제거합니다.

PowerShell

```
Get-SSMAssociationExecution `
  -AssociationId ID `
  -Filter
 @{"Key"="CreatedTime";"Value"="2019-06-01T19:15:38.372Z";"Type"="GREATER_THAN"}
```

> **Note**
>
> 이 명령에는 특정 날짜 및 시간 이후에 발생한 실행으로만 결과를 제한하는 필터가 포함됩니다. 특정 연결 ID의 모든 실행을 보려는 경우 `-Filter` 파라미터 및 `@{"Key"="CreatedTime";"Value"="2019-06-01T19:15:38.372Z";"Type"="GREAT` 값을 제거합니다.

시스템은 다음과 같은 정보를 반환합니다.

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
```

하나 이상의 필터를 사용하려 결과를 제한할 수 있습니다. 다음 예는 특정 날짜 및 시간 이전에 실행된 모든 연결을 반환합니다.

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"}
```

다음은 특정 날짜 및 시간 이후에 성공적으로 실행된 모든 연결을 반환합니다.

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. 다음 명령을 실행하여 특정 실행이 실행된 모든 대상을 봅니다.

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
```

하나 이상의 필터를 사용하려 결과를 제한할 수 있습니다. 다음 예는 특정 연결 실행에 실패한 모든 대상에 대한 정보를 반환합니다.

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"
    }
```

다음 예는 연결 실행에 실패한 특정 관리형 노드에 대한 정보를 반환합니다.

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. 실행에 실패한 연결을 조사하는 경우 StartAssociationsOnce API 작업을 사용하여 연결을 한 번만 즉시 실행할 수 있습니다. 연결 실행에 실패한 리소스를 변경한 후 다음 명령을 실행하여 연결을 한 번만 즉시 실행합니다.

   Linux & macOS

   ```
   aws ssm start-associations-once \
     --association-id ID
   ```

   Windows

   ```
   aws ssm start-associations-once ^
     --association-id ID
   ```

   PowerShell

   ```
   Start-SSMAssociationsOnce `
     -AssociationId ID
   ```

IAM을 사용한 연결 작업

State Manager의 기능인 AWS Systems Manager는 대상을 사용하여 연결을 구성할 인스턴스를 선택합니다. 원래 연결은 문서 이름(Name)과 인스턴스 ID(InstanceId)를 지정하여 생성되었습니다. 그러면 문서와 인스턴스 또는 관리형 노드드 간 연결이 생성됩니다. 이러한 파라미터에 의해 식별되는 데 사용되는 연결입니다. 이러한 파라미터는 이제 더 이상 사용되지 않지만 여전히 지원됩니다. 리소스

instance 및 managed-instance가 Name 및 InstanceId와 함께 작업에 리소스로 추가되었습니다.

AWS Identity and Access Management(IAM) 정책 시행 동작은 지정된 리소스 유형에 따라 다릅니다. State Manager 작업을 위한 리소스는 전달된 요청에 따라서만 시행됩니다. State Manager는 사용자 계정에 있는 리소스의 속성에 대한 정밀 검사를 수행하지 않습니다. 요청 파라미터에 지정된 정책 리소스가 포함된 경우에만 요청이 정책 리소스에 대해 검증됩니다. 예를 들어 리소스 블록에 인스턴스를 지정하면 요청에서 InstanceId 파라미터를 사용하는 경우 정책이 시행됩니다. 계정의 각 리소스에 대한 Targets 파라미터는 해당 InstanceId에 대해 확인되지 않습니다.

다음은 혼란스러운 동작이 포함된 몇 가지 경우입니다.

- DescribeAssociation, DeleteAssociation 및 UpdateAssociation은 instance, managed-instance 및 document 리소스를 사용하여 더 이상 사용되지 않는 연결 참조 방법을 지정합니다. 여기에는 더 이상 사용되지 않는 InstanceId 파라미터로 생성된 모든 연결이 포함됩니다.
- CreateAssociation, CreateAssociationBatch 및 UpdateAssociation은 instance 및 managed-instance 리소스를 사용하여 더 이상 사용되지 않는 연결 참조 방법을 지정합니다. 여기에는 더 이상 사용되지 않는 InstanceId 파라미터로 생성된 모든 연결이 포함됩니다. document 리소스 유형은 더 이상 사용되지 않는 연결 참조 방법의 일부이며 연결의 실제 속성입니다. 즉, 문서 이름을 기반으로 Create 및 Update 작업 모두에 대해 Allow 또는 Deny 권한이 있는 IAM 정책을 구성할 수 있습니다.

Systems Manager에서 IAM 정책을 사용하는 방법에 대한 자세한 내용은 Service Authorization Reference의 AWS Systems Manager의 I자격 증명 및 액세스 관리 또는 Actions, resources, and condition keys for AWS Systems Manager를 참조하세요.

AWS Systems ManagerState Manager 연습

다음 시연에서는 Systems Manager 콘솔 또는 AWS Command Line Interface(AWS CLI)를 사용하여 State Manager 연결을 생성하고 구성하는 방법을 보여줍니다. 또한 AWS Systems Manager의 기능인 State Manager를 사용하여 일반 관리 태스크를 자동으로 수행하는 방법을 보여줍니다.

주제

- 시연: MOF 파일을 실행하는 연결 생성
- 시연: Ansible 플레이북을 실행하는 연결 생성
- 시연: Chef 레시피를 실행하는 연결 생성
- 시연: SSM Agent(CLI) 자동 업데이트

- 연습: Windows Server용 EC2 인스턴스에서 PV 드라이버 자동 업데이트(콘솔)

시연: MOF 파일을 실행하는 연결 생성

MOF(Management Object Format) 파일을 실행하여 AWS-ApplyDSCMofs SSM 문서를 사용해 AWS Systems Manager의 기능인 State Manager에서 Windows Server 관리형 노드에 대해 원하는 상태를 적용할 수 있습니다. AWS-ApplyDSCMofs 문서에는 2가지 실행 모드가 있습니다. 첫 번째 모드에서는 관리형 노드가 지정된 MOF 파일에 정의된 원하는 상태인지 검사해 보고하도록 연결을 구성할 수 있습니다. 두 번째 모드에서는 MOF 파일에 정의된 리소스와 그 값을 기반으로 MOF 파일을 실행하고 노드의 구성을 변경할 수 있습니다. AWS-ApplyDSCMofs 문서를 사용하면 로컬 공유인 Amazon Simple Storage Service(Amazon S3) 또는 HTTPS 도메인을 사용하는 안전한 웹 사이트에서 MOF 구성 파일을 다운로드하고 실행할 수 있습니다.

State Manager는 각 연결 실행 중 개별 MOF 파일의 상태를 기록 및 보고합니다. State Manager는 또한 각 MOF 파일 실행의 출력도 규정 준수 이벤트로 보고하며, 이러한 내용은 AWS Systems Manager 규정 준수 페이지에서 확인할 수 있습니다.

MOF 파일 실행은 PowerShell DSC(Windows PowerShell Desired State Configuration)에서 빌드됩니다. PowerShell DSC는 Windows 시스템 구성, 배포 및 관리에 사용되는 서술식 플랫폼입니다. PowerShell DSC를 사용해 관리자는 DSC 구성이라는 간단한 텍스트 문서의 형식으로 서버 구성 방식을 설명할 수 있습니다. PowerShell DSC 구성은 할 일이 명시된 특수화된 PowerShell 스크립트이지만 여기에는 할 일 수행 방식은 나와 있지 않습니다. 구성을 실행하면 MOF 파일이 생성됩니다. MOF 파일은 하나 이상의 서버에 적용해 해당 서버에 대해 원하는 구성을 얻을 수 있습니다. PowerShell DSC 리소스는 구성 적용의 실제 작업을 수행합니다. 자세한 내용은 Windows PowerShell Desired State Configuration 개요를 참조하십시오.

주제

- Amazon S3를 사용하여 아티팩트 저장
- MOF 파일에서 자격 증명 확인
- MOF 파일에서 토큰 사용
- 필수 조건
- MOF 파일을 실행하는 연결 생성
- 문제 해결
- DSC 리소스 규정 준수 세부 정보 보기

Amazon S3를 사용하여 아티팩트 저장

Amazon S3를 사용하여 PowerShell 모듈, MOF 파일, 규정 준수 보고서 또는 상태 보고서를 저장하는 경우 AWS Systems Manager SSM Agent에서 사용하는 AWS Identity and Access Management(IAM) 역할에는 버킷에 대한 `GetObject` 및 `ListBucket` 권한이 있어야 합니다. 이러한 권한을 제공하지 않으면 시스템에서는 액세스 거부됨 오류가 반환됩니다. 다음은 Amazon S3에 아티팩트 저장에 대한 중요 정보입니다.

- 버킷이 다른 AWS 계정에 있는 경우 해당 계정(또는 IAM 역할)에 `GetObject` 및 `ListBucket` 권한을 부여하는 버킷 리소스 정책을 생성합니다.
- 사용자 정의 DSC 리소스를 사용하려는 경우 Amazon S3 버킷에서 이러한 리소스를 다운로드할 수 있습니다. 또한 PowerShell 갤러리에서 자동으로 설치할 수도 있습니다.
- Amazon S3를 모듈 소스로 사용하는 경우 다음 대/소문자를 구분하는 *ModuleName_ModuleVersion*.zip 형식의 Zip 파일로 모듈을 업로드해야 합니다. 예: MyModule_1.0.0.zip.
- 모든 파일이 버킷 루트에 있어야 합니다. 폴더 구조는 지원되지 않습니다.

MOF 파일에서 자격 증명 확인

자격 증명은 AWS Secrets Manager 또는 AWS Systems Manager Parameter Store를 사용하여 확인합니다. 그러면 자동 자격 증명 교체를 설정할 수 있습니다. 또한 덕분에 DSC가 MOF를 다시 배포하지 않고 서버에 자격 증명을 자동으로 전파할 수 있습니다.

구성에 AWS Secrets Manager 암호를 사용하려면 사용자 이름이 자격 증명을 포함한 암호의 SecretId 또는 SecretARN인 PSCredential 객체를 생성합니다. 암호에 대해 모든 값을 지정할 수 있습니다. 이 값은 무시됩니다. 다음은 한 예입니다.

```
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
```

```
        }
    }
}
```

구성 데이터의 PsAllowPlaintextPassword 설정을 사용하여 MOF를 컴파일합니다. 자격 증명에는 레이블만 포함되기 때문에 괜찮습니다.

Secrets Manager에서 노드가 IAM Managed 관리형 정책과 선택적으로 암호 리소스 정책(있는 경우)에 GetSecretValue 액세스를 가지고 있는지 확인합니다. DSC 작업을 수행하려면 암호가 다음과 같은 형식이어야 합니다.

```
{ 'Username': 'a_name', 'Password': 'a_password' }
```

암호에는 다른 속성(예: 교체에 사용되는 속성)이 있을 수 있지만 최소한 사용자 이름 및 암호 속성은 있어야 합니다.

두 개의 다른 사용자 및 암호가 있고 교체 AWS Lambda 함수가 이러한 사용자 및 암호 간에 전환하는 다중 사용자 교체 방법을 사용하는 것이 좋습니다. 이 방법을 사용하면 활성 계정을 여러 개 가질 수 있고, 교체 중 사용자를 잠글 위험이 사라집니다.

MOF 파일에서 토큰 사용

토큰은 MOF 파일 컴파일 후 리소스 속성 값을 수정할 수 있는 기능을 제공합니다. 따라서 유사한 구성이 필요한 여러 서버에서 공통 MOF 파일을 다시 사용할 수 있습니다.

토큰 대체는 유형 `String`의 리소스 속성에 대해서만 작동합니다. 그러나 리소스에 중첩된 CIM 노드 속성이 있는 경우 해당 CIM 노드의 `String` 속성에서 토큰을 확인합니다. 숫자 또는 배열에는 토큰 대체를 사용할 수 없습니다.

예를 들어, xComputerManagement 리소스를 사용하고 DSC를 사용하여 컴퓨터의 이름을 바꾸려고 하는 시나리오에 대해 생각해 보십시오. 일반적으로 해당 컴퓨터 전용 MOF 파일이 필요합니다. 그러나 토큰 지원을 통해 단일 MOF 파일을 생성해 모든 노드에 적용할 수 있습니다. MOF에 컴퓨터 이름을 하드 코딩하는 대신 `ComputerName` 속성에서 인스턴스 태그 유형 토큰을 사용할 수 있습니다. 이 값은 MOF 구문 분석 중 확인됩니다. 다음 예를 참조하세요.

```
Configuration MyConfig
{
    xComputer Computer
    {
        ComputerName = '{tag:ComputerName}'
    }
```

```
}
```

그런 다음 Systems Manager 콘솔에서 관리형 노드에 태그를 설정하거나 Amazon EC2 콘솔에서 Amazon Elastic Compute Cloud(Amazon EC2) 태그를 설정합니다. 문서를 실행할 때 스크립트가 인스턴스 태그의 값에 대한 {tag:ComputerName} 토큰을 대체합니다.

또한 다음 예와 같이 여러 태그를 단일 속성으로 통합할 수도 있습니다.

```
Configuration MyConfig
{
    File MyFile
    {
        DestinationPath = '{env:TMP}\{tag:ComputerName}'
        Type = 'Directory'
    }
}
```

사용 가능한 다음과 같은 5가지 다른 유형의 토큰이 있습니다.

- tag: Amazon EC2 또는 관리형 노드 태그입니다.
- tagb64: tag와 동일하지만 시스템에서 값을 디코딩하는 데 base64를 사용합니다. 따라서 태그 값에 특수 문자를 사용할 수 있습니다.
- env: 환경 변수를 확인합니다.
- ssm: Parameter Store 값입니다. String 및 Secure String 유형만 지원됩니다.
- tagssm: tag와 동일하지만 tag가 노드에 대해 설정되지 않은 경우 시스템에서는 이름이 동일한 Systems Manager 파라미터에서 값을 확인하려고 합니다. 이는 '기본 전역 값'이 필요하지만 단일 노드에서 이 값을 재정의하길 원하는 경우 유용합니다(예: 단일 시스템 배포).

다음은 `ssm` 토큰 유형을 사용하는 Parameter Store 예입니다.

```
File MyFile
{
    DestinationPath = "C:\ProgramData\ConnectionData.txt"
    Content = "{ssm:%servicePath%/ConnectionData}"
}
```

토큰은 MOF 파일을 일반적이고 재사용 가능하게 만들어 중복 코드를 줄이는 데 중요한 역할을 합니다. 서버별 MOF 파일을 피할 수 있는 경우 MOF 구축 서비스가 필요 없습니다. MOF 컴파일 시 빌드 서

버에 다른 모듈 버전이 설치되므로 MOF 구축 서비스를 사용하면 비용이 증가하고, 프로비저닝 시간이 늘어나고, 그룹화된 노드 간에 구성 편차 가능성이 커집니다.

필수 조건

MOF 파일을 실행하는 연결을 생성하기 전에 관리형 노드에 다음 사전 조건이 설치되어 있는지 확인합니다.

- Windows PowerShell 버전 5.0 이상. 자세한 내용은 Microsoft.com에서 Windows PowerShell System Requirements를 참조하십시오.
- AWS Tools for Windows PowerShell 버전 3.3.261.0 이상
- SSM Agent 버전 2.2 이상

MOF 파일을 실행하는 연결 생성

MOF 파일을 실행하는 연결을 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 State Manager를 선택합니다.
3. [State Manager]를 선택하고 [연결 생성(Create association)]을 선택합니다.
4. 이름(Name) 필드에 이름을 지정합니다. 이는 선택 사항이며, 권장 사항은 아닙니다. 이 이름으로 연결 생성 시 연결의 용도를 파악할 수 있습니다. 공백은 이름에 사용할 수 없습니다.
5. [문서(Document)] 목록에서 **AWS-ApplyDSCMofs**를 선택합니다.
6. 파라미터 섹션에서 필요한 선택적 입력 파라미터를 지정합니다.

   a. Mofs To Apply(적용할 MOF): 연결 실행 시 실행할 MOF 파일을 하나 이상 지정합니다. 쉼표를 사용해 MOF 파일 목록을 구분합니다. MOF 파일을 찾기 위해 다음 옵션을 지정할 수 있습니다.

      - Amazon S3 버킷 이름입니다. 버킷 이름은 소문자여야 합니다. 다음 형식을 사용하여 이 정보를 지정합니다.

        ```
        s3:DOC-EXAMPLE-BUCKET:MOF_file_name.mof
        ```

        AWS 리전을 지정하려면 다음 형식을 사용합니다.

        ```
        s3:bucket_Region:DOC-EXAMPLE-BUCKET:MOF_file_name.mof
        ```

- 안전한 웹 사이트 다음 형식을 사용하여 이 정보를 지정합니다.

```
https://domain_name/MOF_file_name.mof
```

다음 예를 참고하세요

```
https://www.example.com/TestMOF.mof
```

- 로컬 공유의 파일 시스템. 다음 형식을 사용하여 이 정보를 지정합니다.

```
\server_name\shared_folder_name\MOF_file_name.mof
```

다음 예를 참고하세요

```
\StateManagerAssociationsBox\MOFs_folder\MyMof.mof
```

b. [서비스 경로(Service Path)]: (옵션) 서비스 경로는 보고서 및 상태 정보를 작성하려는 Amazon S3 버킷 접두사이거나 Parameter Store 파라미터 기반 태그의 경로입니다. 파라미터 기반 태그를 확인할 때 시스템에서는 {ssm:%servicePath%/*parameter_name*을 사용하여 파라미터 이름에 servicePath 값을 삽입합니다. 예를 들어, 서비스 경로가 "WebServers/Production"인 경우 시스템에서는 이 파라미터를 WebServers/Production/*parameter_name*으로 확인합니다. 이는 동일한 계정에서 여러 환경을 실행하는 경우 유용합니다.

c. [보고서 버킷 이름(Report Bucket Name)]: (옵션) 규정 준수 데이터를 쓰려는 Amazon S3 버킷의 이름을 입력합니다. 보고서는 이 버킷에 JSON 형식으로 저장됩니다.

> **Note**
>
> 버킷이 있는 리전으로 버킷 이름의 접두사를 지정할 수 있습니다. 예: us-west-2:MyMOFBucket. 특정 리전(us-east-1 제외)에서 Amazon S3 엔드포인트에 프록시를 사용하는 경우 리전으로 버킷 이름의 접두사를 지정합니다. 버킷 이름에 접두사가 지정되지 않은 경우 us-east-1 엔드포인트를 사용하여 자동으로 버킷 리전을 검색합니다.

d. [Mof 작업 모드(Mof Operation Mode)]: **AWS-ApplyDSCMofs** 연결을 실행할 때 State Manager 동작을 선택합니다.

- 적용: 규정을 준수하지 않는 노드 구성을 수정합니다.

- ReportOnly: 노드 구성은 수정하지 않지만 대신 모든 규정 준수 데이터를 로깅하고 규정을 준수하지 않는 노드를 보고합니다.

e. [상태 버킷 이름(Status Bucket Name)]: (옵션) MOF 실행 상태 정보를 쓰려는 Amazon S3 버킷의 이름을 입력합니다. 이러한 상태 보고서는 노드의 최신 규정 준수 실행의 singleton 요약입니다. 즉, 이 보고서는 다음에 연결이 MOF 파일을 실행할 때 덮어씁니다.

> **Note**
>
> 버킷이 있는 리전으로 버킷 이름의 접두사를 지정할 수 있습니다. 예를 들면 `us-west-2:DOC-EXAMPLE-BUCKET`입니다. 특정 리전(us-east-1 제외)에서 Amazon S3 엔드포인트에 프록시를 사용하는 경우 리전으로 버킷 이름의 접두사를 지정합니다. 버킷 이름에 접두사가 지정되지 않은 경우 us-east-1 엔드포인트를 사용하여 자동으로 버킷 리전을 검색합니다.

f. [모듈 소스 버킷 이름(Module Source Bucket Name)]: (옵션) PowerShell 모듈 파일이 포함된 Amazon S3 버킷의 이름을 입력합니다. [없음(None)]을 지정하는 경우 [PS 갤러리 모듈 소스 허용(Allow PS Gallery Module Source)] 옵션에 대해 [True]를 선택합니다.

> **Note**
>
> 버킷이 있는 리전으로 버킷 이름의 접두사를 지정할 수 있습니다. 예를 들면 `us-west-2:DOC-EXAMPLE-BUCKET`입니다. 특정 리전(us-east-1 제외)에서 Amazon S3 엔드포인트에 프록시를 사용하는 경우 리전으로 버킷 이름의 접두사를 지정합니다. 버킷 이름에 접두사가 지정되지 않은 경우 us-east-1 엔드포인트를 사용하여 자동으로 버킷 리전을 검색합니다.

g. [PS 갤러리 모듈 소스 허용(Allow PS Gallery Module Source)]: (옵션) https://www.powershellgallery.com/에서 PowerShell 모듈을 다운로드하려면 True를 선택합니다. [False]를 선택하는 경우 이전 옵션인 ModuleSourceBucketName에 대해 소스를 지정합니다.

h. Proxy Uri(프록시 Uri): (선택 사항) 프록시 서버에서 MOF 파일을 다운로드하려면 이 옵션을 사용합니다.

i. Reboot Behavior(재부팅 동작): (선택 사항) MOF 파일을 실행하려면 재부팅해야 하는 경우 다음 재부팅 동작 중 하나를 지정합니다.

- AfterMof: 모든 MOF 실행이 완료되면 노드를 재부팅합니다. 여러 MOF 실행 요청이 재부팅되더라도 시스템에서는 모든 MOF 실행이 재부팅을 완료할 때까지 대기합니다.

- 즉시(Immediately): MOF 실행 시 요청할 때마다 노드를 재부팅합니다. 재부팅을 요청하는 MOF 파일을 여러 개 실행하는 경우 노드가 여러 번 재부팅됩니다.
- 안 함(Never): MOF 실행이 명시적으로 재부팅을 요청하더라도 노드가 재부팅되지 않습니다.

j. [보고에 컴퓨터 이름 사용(Use Computer Name For Reporting)]: (옵션) 규정 준수 정보를 보고할 때 컴퓨터의 이름을 사용하려면 이 옵션을 설정합니다. 기본값은 false로, 규정 준수 정보를 보고할 때 시스템에서 노드 ID를 사용함을 의미합니다.

k. [상세 정보 로깅 활성화(Enable Verbose Logging)]: (옵션) 처음으로 MOF 파일을 배포하는 경우 상세 정보 로깅을 설정하는 것이 좋습니다.

> **Important**
>
> 허용되면 상세 정보 로깅은 표준 연결 실행 로깅보다 Amazon S3 버킷에 더 많은 데이터를 씁니다. 이로 인해 성능이 저하되고 Amazon S3에 대한 스토리지 비용이 높아질 수 있습니다. 스토리지 크기 문제를 완화하기 위해 Amazon S3 버킷에 대해 수명 주기 정책을 설정하는 것이 좋습니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 S3 버킷에 대한 수명 주기 정책을 생성하려면 어떻게 해야 하나요? 섹션을 참조하세요.

l. [Enable Debug Logging(디버그 로깅 설정)]: (옵션) MOF 실패 문제를 해결하려면 디버그 로깅을 설정하는 것이 좋습니다. 또한 일반적인 사용에는 이 옵션을 비활성화하는 것이 좋습니다.

> **Important**
>
> 허용되면 디버그 로깅은 표준 연결 실행 로깅보다 Amazon S3 버킷에 더 많은 데이터를 씁니다. 이로 인해 성능이 저하되고 Amazon S3에 대한 스토리지 비용이 높아질 수 있습니다. 스토리지 크기 문제를 완화하기 위해 Amazon S3 버킷에 대해 수명 주기 정책을 설정하는 것이 좋습니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 S3 버킷에 대한 수명 주기 정책을 생성하려면 어떻게 해야 하나요? 섹션을 참조하세요.

m. 규정 준수 유형: (선택 사항) 규정 준수 정보를 보고할 때 사용할 규정 준수 유형을 지정합니다. 기본 규정 준수 유형은 Custom:DSC입니다. MOF 파일을 실행하는 연결을 여러 개 생성하는 경우 각 연결에 대해 규정 준수 유형을 다르게 지정해야 합니다. 그렇지 않은 경우 Custom:DSC를 사용하는 각각의 추가 연결이 기존 규정 준수 데이터를 덮어씁니다.

n. Pre Reboot Script(재부팅 전 스크립트): (선택 사항) 구성에 재부팅이 필요하다고 표시된 경우 실행할 스크립트를 지정합니다. 이 스크립트는 재부팅 전에 실행됩니다. 이 스크립트는 한 줄이어야 합니다. 세미콜론을 사용하여 추가 행을 구분합니다.

7. 대상 섹션에서 태그 지정 또는 수동으로 인스턴스 선택을 선택합니다. 태그를 사용하여 리소스를 대상으로 지정하기로 한 경우 태그 키와 태그 값을 제공된 필드에 입력합니다. 태그 사용에 대한 자세한 내용은 State Manager 연결에서의 대상 및 속도 제어 정보 섹션을 참조하세요.
8. 일정 지정 섹션에서 On Schedule(일정이 있을 때) 또는 No schedule(일정이 없을 때)을 선택합니다. On Schedule(일정이 있을 때)을 선택한 경우 제공된 버튼을 사용하여 연결에 대한 cron 또는 rate 일정을 생성합니다.
9. 고급 옵션 섹션에서:

   - 규정 준수 심각도에서 연결에 대한 심각도를 선택합니다. 규정 준수 보고는 여기서 지정한 심각도 수준과 함께 연결 상태가 준수인지 아니면 미준수인지를 나타냅니다. 자세한 내용은 State Manager 연결 규정 준수 정보 단원을 참조하십시오.

10. 속도 제어(Rate control) 섹션에서 관리형 노드 플릿 간에 State Manager 연결을 실행하기 위한 옵션을 구성합니다. 이러한 옵션에 대한 자세한 내용은 State Manager 연결에서의 대상 및 속도 제어 정보 섹션을 참조하세요.

    동시성 섹션에서 옵션을 선택합니다.

    - 대상을 선택하여 연결을 동시에 실행할 수 있는 대상 수(절대 개수)를 입력합니다.
    - 백분율을 선택하여 연결을 동시에 실행할 수 있는 대상의 백분율을 입력합니다.

    오류 임계값 섹션에서 옵션을 선택합니다.

    - 오류를 선택하여 State Manager에서 추가 대상에 대한 연결 실행을 중지하기 전에 허용되는 절대 오류 수를 입력합니다.
    - 백분율을 선택하여 State Manager에서 추가 대상에 대한 연결 실행을 중지하기 전에 허용되는 오류 비율을 입력합니다.

11. (선택 사항) 출력 옵션에서 명령 출력을 파일에 저장하려면 S3 버킷에 쓰기 활성화 옆의 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

> **Note**
>
> 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아닌 관리형 노드에 할당된 인스턴스 프로파일의 권한입니다. 자세한 내용은

Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

12. 연결 생성을 선택합니다.

State Manager는 지정된 노드 또는 대상에 대해 연결을 생성하고 즉시 실행합니다. 최초 실행 후 연결은 정의한 일정에 따른 간격으로 다음 규칙에 따라 실행됩니다.

- State Manager는 간격이 시작될 때 온라인 상태인 노드에서 연결을 실행하고 오프라인 노드를 건너뜁니다.
- State Manager에서는 간격 중 구성된 모든 노드에 대해 연결을 실행하려고 합니다.
- (예를 들어, 동시성 값이 한 번에 연결을 처리할 수 있는 노드 수를 제한했기 때문에) 간격 동안 연결이 실행되지 않은 경우 State Manager에서는 다음 간격 중 해당 연결을 실행하려고 합니다.
- State Manager는 건너 뛴 모든 간격을 기록합니다. 이러한 기록은 실행 내역 탭에서 확인할 수 있습니다.

> **Note**
>
> `AWS-ApplyDSCMofs`는 Systems Manager Command 문서입니다. 즉, AWS Systems Manager의 기능인 Run Command를 사용하여 이 문서를 실행할 수도 있습니다. 자세한 내용은 AWS Systems Manager Run Command 단원을 참조하십시오.

문제 해결

이 단원에는 MOF 파일을 실행하는 연결 생성과 관련된 문제를 해결하기 위한 정보가 포함되어 있습니다.

향상된 로깅 설정

문제 해결의 첫 번째 단계로, 고급 로깅을 설정합니다. 보다 구체적으로 다음을 수행하십시오.

1. Amazon S3 또는 Amazon CloudWatch Logs(CloudWatch)에 명령 출력을 쓰도록 연결이 구성되었는지 확인합니다.
2. Enable Verbose Logging(상세 정보 로깅 활성화) 파라미터를 True로 설정합니다.

3. Enable Debug Logging(디버그 로깅 활성화) 파라미터를 True로 설정합니다.

상세 정보 및 디버그 로깅이 설정되면 Stdout 출력 파일에 스크립트 실행에 대한 세부 정보가 포함됩니다. 이 출력 파일은 스크립트가 실패한 위치를 식별하는 데 유용할 수 있습니다. Stderr 출력 파일에는 스크립트 실행 중 발생한 오류가 포함됩니다.

공통 문제

이 단원에는 MOF 파일을 실행하는 연결 생성 시 발생할 수 있는 일반적인 문제에 대한 정보와 이러한 문제 해결 단계가 나와 있습니다.

내 MOF가 적용되지 않음

State Manager에서 노드에 연결을 적용하지 못하면 Stderr 출력 파일을 검토하는 것으로 시작합니다. 이 파일은 문제의 근본 원인을 파악하는 데 도움이 될 수 있습니다. 또한 다음을 확인합니다.

- 노드에는 모든 MOF 관련 Amazon S3 버킷에 필요한 액세스 권한이 있습니다. 구체적으로 설명하면 다음과 같습니다.
  - s3:GetObject 권한: 프라이빗 Amazon S3 버킷의 MOF 파일과 Amazon S3 버킷의 사용자 정의 모듈에 필요합니다.
  - s3:PutObject 권한: Amazon S3 버킷에 규정 준수 보고서 및 규정 준수 상태를 쓰려면 필요합니다.
- 태그를 사용하는 경우 노드에 필요한 IAM 정책이 있는지 확인합니다. 태그를 사용하려면 정책이 `ec2:DescribeInstances` 및 `ssm:ListTagsForResource` 작업을 허용하도록 하기 위해 인스턴스 IAM 역할이 필요합니다.
- 노드에 예상한 태그가 있거나 SSM 파라미터가 할당되어 있는지 확인합니다.
- 태그 또는 SSM 파라미터에 오탈자가 없는지 확인합니다.
- MOF 파일 자체와 관련된 문제가 없는지 확인하기 위해 노드에서 MOF를 로컬로 적용해 보세요.

내 MOF가 실패한 것 같지만 Systems Manager 실행에는 성공함

`AWS-ApplyDSCMofs` 문서가 성공적으로 실행되면 Systems Manager 실행 상태에 [성공(Success)]이라고 표시됩니다. 이 상태는 MOF 파일의 구성 요구 사항을 기준으로 한 노드의 규정 준수 상태를 반영하지 않습니다. 노드의 규정 준수 상태를 확인하려면 규정 준수 보고서를 확인합니다. JSON 보고서는 Amazon S3 보고서 버킷에서 볼 수 있습니다. 이 내용은 Run Command 및 State Manager 실행에 적용됩니다. 또한 State Manager의 경우 Systems Manager Compliance 페이지에서 규정 준수 세부 정보를 확인할 수 있습니다.

Stderr 상태: 서비스 연결 시도 시 이름 확인 실패

이 오류는 스크립트가 원격 서비스에 연결할 수 없음을 나타냅니다. 이 스크립트는 Amazon S3에 연결하지 못할 가능성이 매우 큽니다. 대부분 이 문제는 스크립트가 문서 파라미터에 제공된 Amazon S3 버킷에 규정 준수 보고서 또는 규정 준수 상태를 쓰려고 할 때 발생합니다. 일반적으로 이 오류는 컴퓨팅 환경이 방화벽 또는 허용 목록을 포함한 투명한 프록시를 사용하는 경우 발생합니다. 이 문제를 해결하려면:

- 모든 Amazon S3 버킷 파라미터에 대해 리전별 버킷 구문을 사용합니다. 예를 들어, Mofs to Apply(적용할 Mof) 파라미터의 형식은 다음과 같아야 합니다.

  s3:*bucket-region*:*bucket-name*:*mof-file-name*.mof.

  예: `s3:us-west-2:DOC-EXAMPLE-BUCKET:my-mof.mof`

  보고서, 상태 및 모듈 소스 버킷 이름의 형식은 다음과 같아야 합니다.

  bucket-region:*bucket-name*. 단순 예시: `us-west-1:DOC-EXAMPLE-BUCKET;`
- 리전별 구문으로 문제가 해결되지 않으면 대상 노드가 원하는 리전에서 Amazon S3에 액세스할 수 있는지 확인합니다. 이를 확인하려면 다음을 수행합니다.
  1. 적절한 Amazon S3 리전에서 Amazon S3의 엔드포인트 이름을 찾습니다. 자세한 내용은 Amazon Web Services 일반 참조의 Amazon S3 서비스 엔드포인트를 참조하세요.
  2. 대상 노드에 로그온해 다음 ping 명령을 실행합니다.

     ```
     ping s3.s3-region.amazonaws.com
     ```

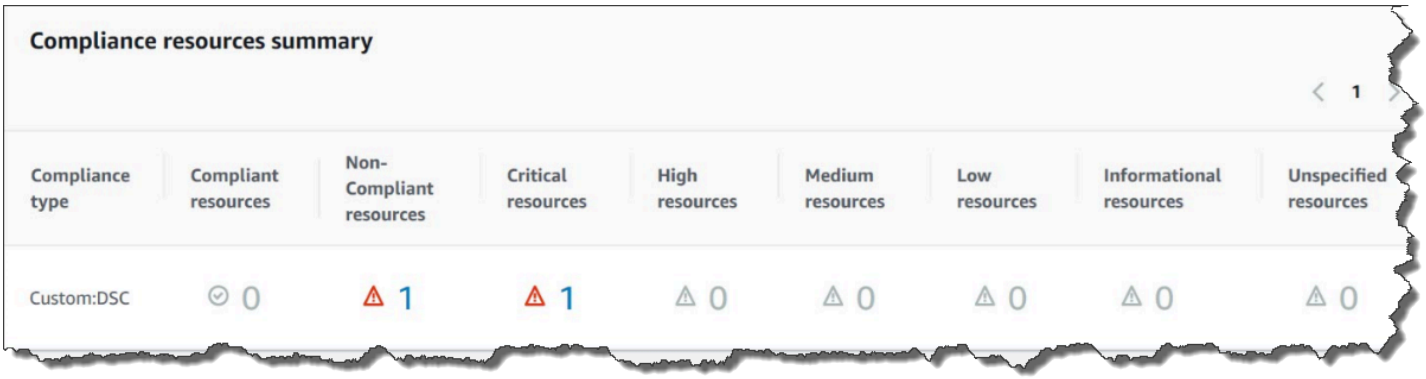
     ping에 실패하면 이는 Amazon S3이 다운되었거나, 방화벽/투명한 프록시가 Amazon S3 리전에 대한 액세스를 차단했거나, 노드가 인터넷에 액세스할 수 없는 것입니다.

DSC 리소스 규정 준수 세부 정보 보기

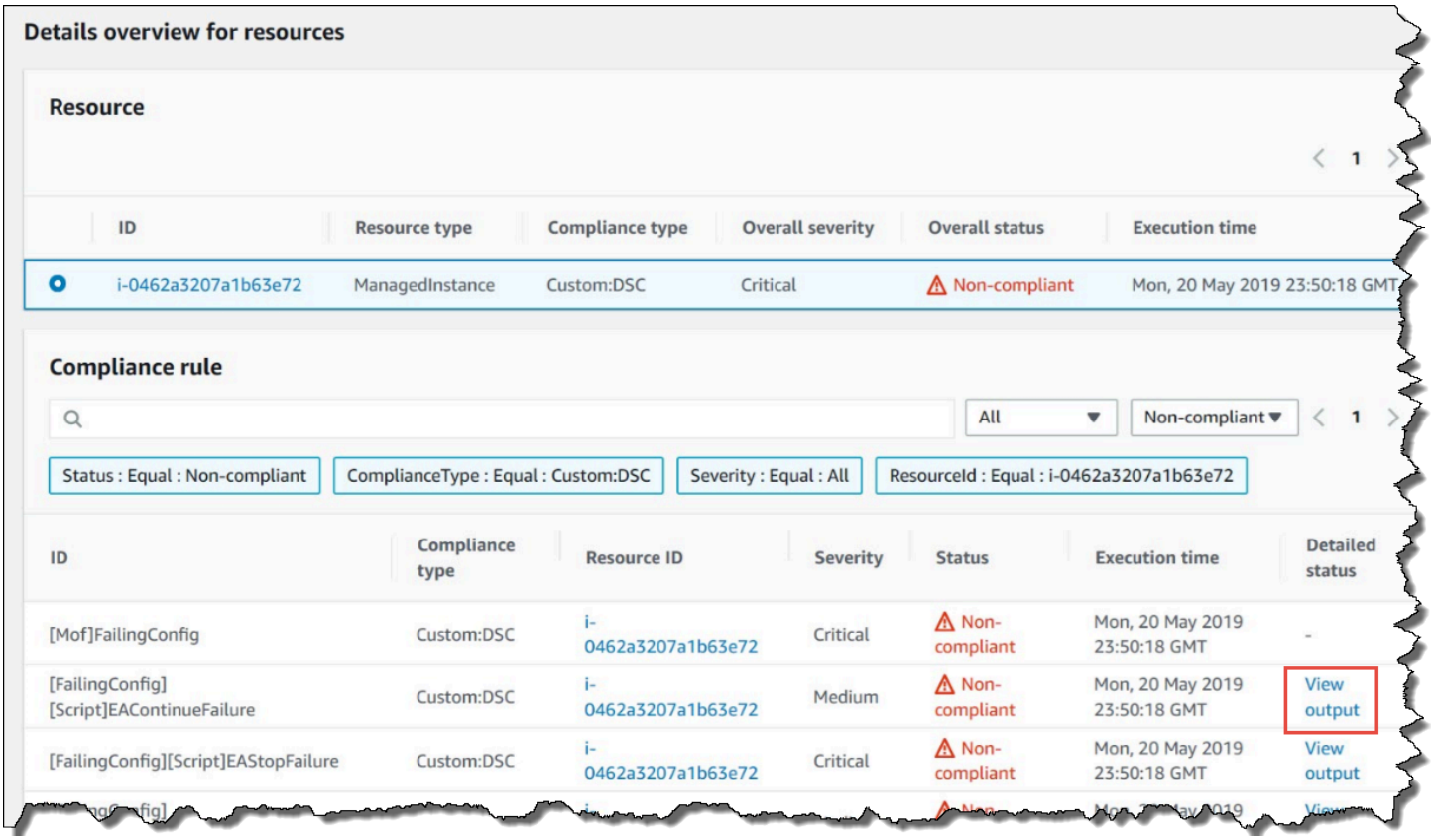
Systems Manager는 `AWS-ApplyDSCMofs` 문서를 실행했을 때 지정한 Amazon S3 [상태 버킷(Status Bucket)]에서 DSC 리소스 실패에 대한 규정 준수 정보를 수집합니다. Amazon S3 버킷에서 DSC 리소스 실패에 대한 정보를 검색하는 데는 시간이 많이 걸릴 수 있습니다. 대신, Systems Manager [Compliance] 페이지에서 이 정보를 볼 수 있습니다.

규정 준수 리소스 요약 섹션에 실패한 리소스의 수가 표시됩니다. 다음 예에서 ComplianceType은 Custom:DSC이고 한 개의 리소스가 규정 미준수입니다.

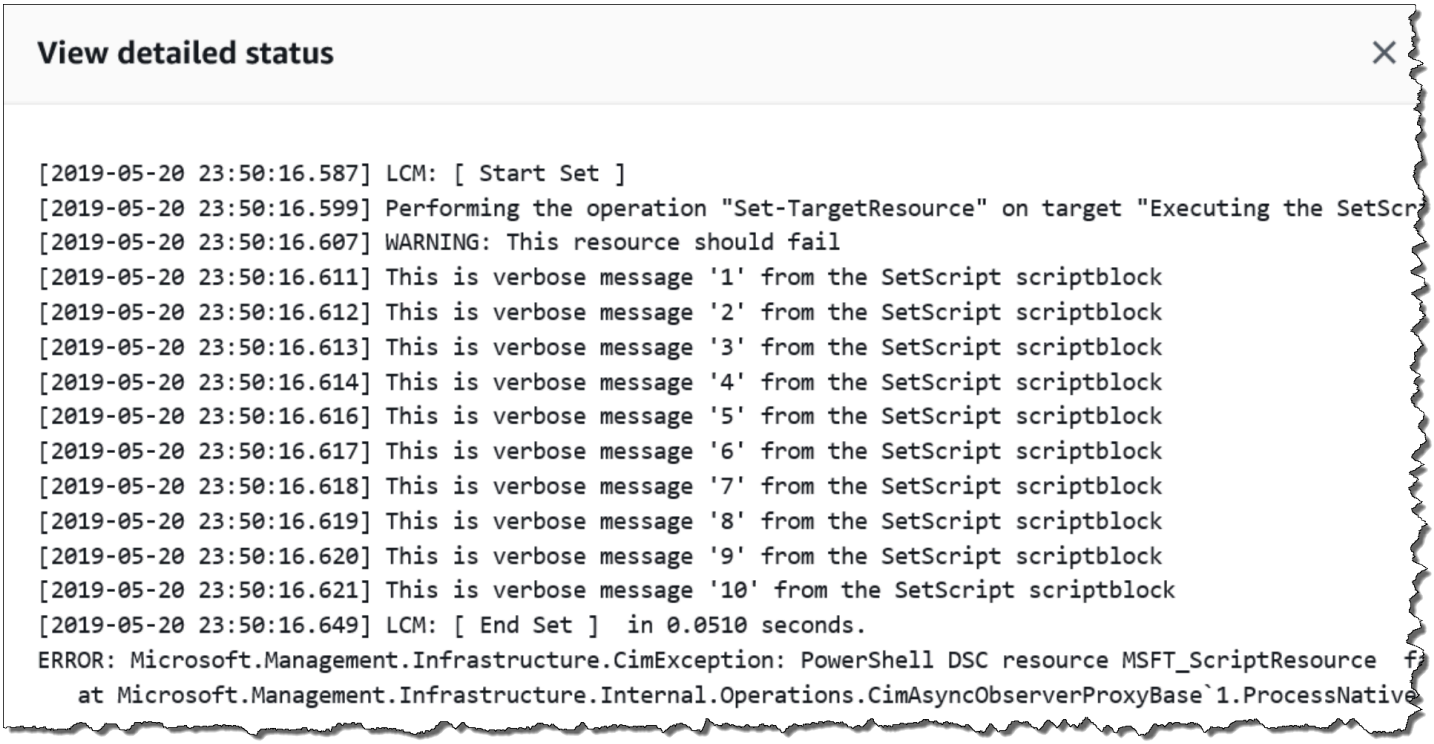
> **Note**
>
> Custom:DSC는 `AWS-ApplyDSCMofs` 문서에서 기본 [ComplianceType] 값입니다. 이 값은 사용자 지정 가능합니다.



[리소스에 대한 세부 정보 개요(Details overview for resources)] 섹션에는 규정 미준수 DSC 리소스가 있는 AWS 리소스에 대한 정보가 표시됩니다. 이 섹션에는 또한 MOF 이름, 스크립트 실행 단계, 및 세부 상태 정보를 볼 수 있는 출력 보기 링크(적용되는 경우)도 포함됩니다.



[출력 보기(View output)] 링크는 세부 상태의 마지막 4,000자를 표시합니다. Systems Manager는 첫 번째 요소로 예외를 시작한 다음 자세한 메시지를 다시 검사하고 4,000자 할당량에 도달할 때까지 가능한 한 많이 추가합니다. 이 프로세스는 예외가 발생되기 전에 출력된 긴 메시지(문제 해결을 위해 가장 관련성이 높은 메시지)를 표시합니다.



규정 준수 정보를 보는 방법에 대한 자세한 내용은 AWS Systems Manager Compliance 섹션을 참조하세요.

규정 준수 보고에 영향을 미치는 상황

State Manager 연결이 실패한 경우에는 규정 준수 데이터가 보고되지 않습니다. 보다 구체적으로, MOF가 처리되지 못한다면 연결이 실패하기 때문에 Systems Manager가 어떤 규정 준수 항목도 보고하지 않습니다. 예를 들어, Systems Manager가 노드에 액세스할 권한이 없는 Amazon S3 버킷에서 MOF를 다운로드하려고 시도하면 연결이 실패하고 어떠한 규정 준수 데이터도 보고되지 않습니다.

두 번째 MOF의 리소스가 실패하면 Systems Manager가 규정 준수 데이터를 보고합니다. 예를 들어 MOF가 존재하지 않는 드라이브에서 파일을 생성하려고 하면 `AWS-ApplyDSCMofs` 문서가 완전히 처리될 수 있기 때문에(즉, 연결이 성공적으로 실행될 수 있기 때문에) Systems Manager가 규정 준수를 보고합니다.

시연: Ansible 플레이북을 실행하는 연결 생성

AWS-ApplyAnsiblePlaybooks SSM 문서를 사용하여 Ansible 플레이북을 실행하는 State Manager 연결을 생성할 수 있습니다. State Manager는 AWS Systems Manager의 기능입니다. 이 문서는 플레이북 실행을 위해 다음과 같은 이점을 제공합니다.

- 복잡한 플레이북 실행 지원
- GitHub 및 Amazon Simple Storage Service(S3)에서 플레이북 다운로드 지원
- 압축된 플레이북 구조 지원
- 향상된 로깅
- 플레이북이 번들로 제공될 때 실행할 플레이북 지정 가능

> **Note**
>
> Systems Manager에는 Ansible 플레이북을 실행하는 State Manager 연결을 생성하는 데 사용할 수 있는 SSM 문서가 2개(AWS-RunAnsiblePlaybook, AWS-ApplyAnsiblePlaybooks)가 포함되어 있습니다. AWS-RunAnsiblePlaybook 문서는 더 이상 사용되지 않습니다. Systems Manager에서 레거시용으로 제공됩니다. AWS-ApplyAnsiblePlaybooks 문서에 여기서 설명한 기능 향상 부분이 있으므로 이 문서를 사용하는 것이 좋습니다.
>
> Ansible 플레이북을 실행하는 연결은 macOS에서 지원되지 않습니다.

복잡한 플레이북 실행 지원

AWS-ApplyAnsiblePlaybooks 문서는 지정된 주요 플레이북을 실행하기 전에 먼저 로컬 디렉터리에 전체 파일 구조를 복사하므로 번들로 제공되는 복잡한 플레이북을 지원합니다. 소스 플레이북은 zip 파일 또는 디렉터리 구조로 제공할 수 있습니다. Zip 파일이나 디렉터리는 GitHub 또는 Amazon S3에 저장할 수 있습니다.

GitHub에서 플레이북 다운로드 지원

AWS-ApplyAnsiblePlaybooks 문서는 aws:downloadContent 플러그인을 사용하여 플레이북 파일을 다운로드합니다. 파일은 GitHub에 한 개의 파일로 또는 결합된 플레이북 세트로 저장할 수 있습니다. GitHub에서 콘텐츠를 다운로드하려면 GitHub 리포지토리에 대한 정보를 JSON 형식으로 지정합니다. 다음 예를 참고하세요

```
{
```

```
    "owner":"TestUser",
    "repository":"GitHubTest",
    "path":"scripts/python/test-script",
    "getOptions":"branch:master",
    "tokenInfo":"{{ssm-secure:secure-string-token}}"
}
```

Amazon S3에서 플레이북 다운로드 지원

Amazon S3에서 Ansible 플레이북을 단일 .zip 파일 또는 디렉터리 구조로 저장하고 다운로드할 수도 있습니다. Amazon S3에서 콘텐츠를 다운로드하려면 파일에 대한 경로를 지정합니다. 다음은 두 가지 예입니다.

예 1: 특정 플레이북 파일 다운로드

```
{
    "path":"https://s3.amazonaws.com/DOC-EXAMPLE-BUCKET/playbook.yml"
}
```

예 2: 디렉터리 콘텐츠 다운로드

```
{
    "path":"https://s3.amazonaws.com/DOC-EXAMPLE-BUCKET/ansible/webservers/"
}
```

> **Important**
>
> Amazon S3를 지정하면 `AmazonS3ReadOnlyAccess` 정책을 사용하여 관리형 노드에 있는 AWS Identity and Access Management(IAM) 노드 프로파일을 구성해야 합니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성을 참조하세요.

압축된 플레이북 구조 지원

`AWS-ApplyAnsiblePlaybooks` 문서를 사용하면 다운로드한 번들에서 압축된 .zip 파일을 실행할 수 있습니다. 문서는 압축된 파일이 다운로드한 파일에 .zip 형식으로 포함되어 있는지 점검합니다. .zip 파일이 있으면 문서는 자동으로 파일 압축을 푼 다음 지정된 Ansible 자동화를 실행합니다.

향상된 로깅

AWS-ApplyAnsiblePlaybooks 문서에는 다양한 로깅 수준을 지정하는 데 필요한 파라미터 옵션이 포함되어 있습니다. 세부 수준이 낮으면 -v를, 중간이면 -vvv를, 디버그 레벨 로깅 수준이면 -vvvv를 지정합니다. 이러한 옵션은 Ansible 세부 수준 옵션에 직접 매핑됩니다.

플레이북이 번들로 제공될 때 실행할 플레이북 지정 가능

AWS-ApplyAnsiblePlaybooks 문서에는 여러 개의 플레이북이 번들로 제공될 때 실행할 플레이북을 지정하는 데 필요한 파라미터가 포함되어 있습니다. 이 옵션을 사용하면 다양한 사용 사례를 지원하도록 플레이북을 유연하게 실행할 수 있습니다.

설치된 종속성

InstallDependencies 파라미터에 True를 지정하면 Systems Manager는 노드에 다음 종속성이 설치되어 있는지 확인합니다.

- Ubuntu Server/Debian Server: Apt-get(패키지 관리), Python 3, Ansible, Unzip
- Amazon Linux: Ansible
- RHEL: Python 3, Ansible, Unzip

이 종속성 중 하나 이상이 없다면 Systems Manager는 해당 종속성을 자동으로 설치합니다.

Ansible 플레이북을 실행하는 연결 생성(콘솔)

다음 절차에서는 Systems Manager 콘솔을 사용하여 AWS-ApplyAnsiblePlaybooks 문서로 Ansible 플레이북을 실행하는 State Manager 연결을 생성하는 방법을 설명합니다.

Ansible 플레이북을 실행하는 연결을 생성하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 State Manager를 선택합니다.
3. [State Manager]를 선택하고 [연결 생성(Create association)]을 선택합니다.
4. 이름에 연결의 목적을 기억하는 데 도움이 되는 이름을 지정합니다.
5. [문서(Document)] 목록에서 **AWS-ApplyAnsiblePlaybooks**를 선택합니다.
6. 파라미터 섹션의 소스 유형에서 GitHub 또는 S3를 선택합니다.

   GitHub

   GitHub를 선택하는 경우 다음 형식으로 리포지토리 정보를 입력합니다.

```
{
   "owner":"user_name",
   "repository":"name",
   "path":"path_to_directory_or_playbook_to_download",
   "getOptions":"branch:branch_name",
   "tokenInfo":"{{(Optional)_token_information}}"
}
```

S3

[S3]를 선택하는 경우 다음 형식으로 경로 정보를 입력합니다.

```
{
   "path":"https://s3.amazonaws.com/path_to_directory_or_playbook_to_download"
}
```

7. 종속성 설치에서 옵션을 선택합니다.
8. (선택 사항) Playbook File(플레이북 파일)에 파일 이름을 입력합니다. Zip 파일에 플레이북이 포함된 경우 Zip 파일에 대한 상대 경로를 지정합니다.
9. (선택 사항) 추가 변수에서 실행 시간 동안 State Manager에서 Ansible에 전송할 변수를 입력합니다.
10. (선택 사항) 확인에서 옵션을 선택합니다.
11. (선택 사항) 상세 표시에서 옵션을 선택합니다.
12. 대상에서 옵션을 선택합니다. 태그 사용에 대한 자세한 내용은 State Manager 연결에서의 대상 및 속도 제어 정보 섹션을 참조하세요.
13. 일정 지정 섹션에서 On Schedule(일정이 있을 때) 또는 No schedule(일정이 없을 때)을 선택합니다. On Schedule(일정이 있을 때)을 선택한 경우 제공된 버튼을 사용하여 연결에 대한 cron 또는 rate 일정을 생성합니다.
14. 고급 옵션 섹션의 규정 준수 심각도에서 연결에 대한 심각도 수준을 선택합니다. 규정 준수 보고는 여기서 지정한 심각도 수준과 함께 연결 상태가 준수인지 아니면 미준수인지를 나타냅니다. 자세한 내용은 State Manager 연결 규정 준수 정보 단원을 참조하십시오.
15. Rate control(속도 제어) 섹션에서 관리형 노드 플릿 간에 State Manager 연결을 실행하기 위한 옵션을 구성합니다. 속도 제어 사용에 대한 자세한 내용은 State Manager 연결에서의 대상 및 속도 제어 정보 섹션을 참조하세요.

    동시성 섹션에서 옵션을 선택합니다.

- 대상을 선택하여 연결을 동시에 실행할 수 있는 대상 수(절대 개수)를 입력합니다.
- 백분율을 선택하여 연결을 동시에 실행할 수 있는 대상의 백분율을 입력합니다.

오류 임계값 섹션에서 옵션을 선택합니다.

- 오류를 선택하여 State Manager에서 추가 대상에 대한 연결 실행을 중지하기 전에 허용되는 절대 오류 수를 입력합니다.
- 백분율을 선택하여 State Manager에서 추가 대상에 대한 연결 실행을 중지하기 전에 허용되는 오류 비율을 입력합니다.

16. (선택 사항) 출력 옵션에서 명령 출력을 파일에 저장하려면 S3 버킷에 쓰기 활성화 옆의 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

> **Note**
>
> 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아닌 관리형 노드에 할당된 인스턴스 프로파일의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

17. 연결 생성을 선택합니다.

> **Note**
>
> 태그를 사용하여 하나 이상의 대상 노드에 대해 연결을 생성한 다음 노드에서 태그를 제거하면 해당 노드가 더 이상 연결을 실행하지 않습니다. 이러한 노드는 State Manager 문서에서 연결 해제됩니다.

Ansible 플레이북을 실행하는 연결 생성(CLI)

다음 절차에서는 AWS Command Line Interface(AWS CLI)을 사용하여 `AWS-ApplyAnsiblePlaybooks` 문서로 Ansible 플레이북을 실행하는 State Manager 연결을 생성하는 방법을 설명합니다.

Ansible 플레이북을 실행하는 연결을 생성하려면(CLI)

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.

2. 다음 명령 중 하나를 실행하면 태그로 노드를 대상 지정하여 Ansible 플레이북을 실행하는 연결을 생성할 수 있습니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다. Command (A)는 소스 유형으로 GitHub를 지정합니다. Command (B)는 소스 유형으로 Amazon S3를 지정합니다.

   (A) GitHub 소스

   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"
```

   다음 예를 참고하세요

```
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 소스

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"
```

다음 예를 참고하세요

```
aws ssm create-association --name "AWS-ApplyAnsiblePlaybooks" \
```

```
    --targets "Key=tag:OS,Values=Linux" \
    --parameters '{"SourceType":["S3"],"SourceInfo":["{\"path\":\"https://
s3.amazonaws.com/DOC-EXAMPLE-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 연결은 cron 및 rate 표현식 중 일부를 지원하지 않습니다. 연결에 대한 cron 및 rate 표현식을 생성하는 방법에 대한 자세한 내용은 참조: Systems Manager용 Cron 및 Rate 표현식 섹션을 참조하세요.

시스템은 해당 노드에 연결을 생성하고 그 상태를 즉시 적용하려고 합니다.

3. 다음 명령을 실행하여 방금 생성한 연결의 업데이트된 상태를 확인합니다.

```
aws ssm describe-association --association-id "ID"
```

시연: Chef 레시피를 실행하는 연결 생성

Chef SSM 문서를 사용하여 AWS Systems Manager 레시피를 실행하는 State Manager 연결을 생성할 수 있습니다. State Manager는 `AWS-ApplyChefRecipes`의 기능입니다. `AWS-ApplyChefRecipes` SSM 문서를 사용하여 Linux 기반 Systems Manager 관리형 노드를 대상으로 지정할 수 있습니다. 이 문서는 Chef 레시피 실행을 위해 다음과 같은 이점을 제공합니다.

- 여러 Chef 릴리스(Chef 11~Chef 18)를 지원합니다.
- 대상 노드에 Chef 클라이언트 소프트웨어를 자동으로 설치합니다.
- 선택적으로 대상 노드에 대해 Systems Manager 규정 준수 점검을 실행하고 규정 준수 점검 결과를 Amazon Simple Storage Service(Amazon S3) 버킷에 저장합니다.
- 문서를 한 번 실행할 때 여러 쿡북과 레시피를 실행합니다.
- 선택적으로 `why-run` 모드에서 레시피를 실행하여 변경하지 않고 대상 노드에서 변경되는 레시피를 표시합니다.
- 선택적으로 사용자 지정 JSON 속성을 `chef-client` 실행에 적용합니다.
- 사용자가 지정하는 위치에 저장된 소스 파일의 사용자 지정 JSON 속성을 선택적으로 적용합니다.

Git, GitHub, HTTP 또는 Amazon S3 버킷을 AWS-ApplyChefRecipes 문서에서 사용자가 지정하는 Chef 쿡북 및 레시피의 다운로드 소스로 사용할 수 있습니다.

> **Note**
>
> Chef 레시피를 실행하는 연결은 macOS에서 지원되지 않습니다.

사전 조건: 연결, 리포지토리 및 쿡북 설정

AWS-ApplyChefRecipes 문서를 만들기 전에 Chef 쿡북과 쿡북 리포지토리를 준비합니다. 사용하려는 Chef 쿡북이 없는 경우 AWS에서 준비한 테스트 HelloWorld 쿡북을 사용하여 시작할 수 있습니다. AWS-ApplyChefRecipes 문서는 이미 기본적으로 이 쿡북을 가리키고 있습니다. 쿡북은 다음 디렉터리 구조와 유사하게 설정되어야 합니다. 다음 예에서 jenkins 및 nginx는 Chef 웹사이트의 Chef Supermarket에서 사용할 수 있는 Chef 쿡북의 예입니다.

AWS에서 Chef Supermarket 웹사이트의 쿡북을 공식적으로 지원할 수는 없지만 많은 사람들이 AWS-ApplyChefRecipes 문서를 사용합니다. 다음은 커뮤니티 쿡북을 테스트할 때 확인할 기준의 예입니다.

- 쿡북은 대상으로 하는 Systems Manager 관리형 노드의 Linux 기반 운영 체제를 지원해야 합니다.
- 쿡북은 사용자가 사용하는 Chef 클라이언트 버전(Chef 11~Chef 18)에 대해 유효해야 합니다.
- 쿡북은 Chef Infra Client와 호환되며 Chef 서버가 필요하지 않습니다.

Systems Manager 문서(SSM 문서)가 실행될 때 실행 목록에 지정된 쿡북을 설치할 수 있도록 Chef.io 웹사이트에 연결할 수 있는지 확인합니다. 중첩된 cookbooks 폴더 사용은 지원되지만 필수는 아닙니다. 루트 수준 바로 아래에 쿡북을 저장할 수 있습니다.

```
<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**
>
> Chef 레시피를 실행하는 State Manager 연결을 생성하기 전에 Chef 클라이언트 버전 값을 None으로 설정하지 않는 한 문서 실행 시 Systems Manager 관리형 노드에 Chef 클라이언트 소프트웨어가 설치된다는 점에 유의하세요. 이 작업은 Chef의 설치 스크립트를 사용하여 사용자 대신 Chef 구성 요소를 설치합니다. `AWS-ApplyChefRecipes` 문서를 실행하기 전에 기업이 Chef 소프트웨어 사용에 적용되는 라이선스 조건을 포함하여 적용 가능한 법적 요구 사항을 준수할 수 있는지 확인합니다. 자세한 내용은 Chef 웹사이트를 참조하세요.

Systems Manager는 규정 준수 보고서를 Systems Manager 콘솔인 S3 버킷에 제공하거나, Systems Manager API 명령에 대한 응답에서 규정 준수 결과를 사용하도록 만들 수 있습니다. Systems Manager 규정 준수 보고서를 실행하려면 Systems Manager 관리형 노드에 연결된 인스턴스 프로파일에 S3 버킷에 대한 쓰기 권한이 있어야 합니다. 인스턴스 프로파일에는 Systems Manager `PutComplianceItem` API를 사용할 권한이 있어야 합니다. Systems Manager 규정 준수에 대한 자세한 내용은 AWS Systems Manager Compliance 섹션을 참조하세요.

실행 문서 로깅

State Manager 연결을 사용하여 Systems Manager 문서(SSM 문서)를 실행할 때 문서 실행의 출력을 선택하도록 연결을 구성할 수 있으며 출력을 Amazon S3 또는 Amazon CloudWatch Logs(CloudWatch Logs)로 전송할 수 있습니다. 연결 실행이 완료될 때 문제 해결을 쉽게 하려면 연결이 Amazon S3 버킷 또는 CloudWatch Logs에 명령 출력을 쓰도록 구성되어 있는지 확인합니다. 자세한 내용은 Systems Manager에서 연결 작업 단원을 참조하십시오.

레시피 실행 시 대상에 JSON 속성 적용

연결 실행 중에 대상 노드에 적용할 Chef 클라이언트의 JSON 속성을 지정할 수 있습니다. 연결을 설정할 때 원시 JSON을 제공하거나 Amazon S3에 저장된 JSON 파일의 경로를 제공할 수 있습니다.

레시피가 실행되는 방식을 사용자 정의하려는 경우 레시피 자체를 수정할 필요 없이 JSON 속성을 사용합니다. 예를 들면 다음과 같습니다.

- 소수의 속성 재정의

  사용자 지정 JSON을 사용하면 사소한 차이를 수용하기 위해 레시피를 여러 버전으로 유지하지 않아도 됩니다.
- 변수 값 제공

사용자 지정 JSON을 사용하여, 실행할 때마다 변경될 수 있는 값을 지정합니다. 예를 들어 Chef 쿡북이 결제를 허용하는 타사 애플리케이션을 구성하는 경우 사용자 지정 JSON을 사용하여 결제 엔드포인트 URL을 지정할 수 있습니다.

원시 JSON에 속성 지정

다음은 Chef 레시피의 사용자 지정 JSON 속성 지정에 사용할 수 있는 형식의 예제입니다.

```
{"filepath":"/tmp/example.txt", "content":"Hello, World!"}
```

JSON 파일 경로 지정

다음은 Chef 레시피의 사용자 지정 JSON 속성 경로 지정에 사용할 수 있는 형식의 예제입니다.

```
{"sourceType":"s3", "sourceInfo":"someS3URL1"}, {"sourceType":"s3",
 "sourceInfo":"someS3URL2"}
```

쿡북 소스로 Git 사용

AWS-ApplyChefRecipes 문서는 aws:downloadContent 플러그인을 사용하여 Chef 쿡북을 다운로드합니다. Git에서 콘텐츠를 다운로드하려면 Git 리포지토리에 대한 정보를 다음 예제와 같이 JSON 형식으로 지정합니다. 각 *example-resource-placeholder*를 자신의 정보로 바꿉니다.

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

GitHub을 쿡북 소스로 사용

AWS-ApplyChefRecipes 문서는aws:downloadContent 플러그인을 사용하여 쿡북을 다운로드합니다. GitHub에서 콘텐츠를 다운로드하려면 GitHub 리포지토리에 대한 정보를 다음 예제와 같이 JSON 형식으로 지정합니다. 각 *example-resource-placeholder*를 자신의 정보로 바꿉니다.

```
{
```

```
    "owner":"TestUser",
    "repository":"GitHubCookbookRepository",
    "path":"cookbooks/HelloWorld",
    "getOptions":"branch:refs/head/main",
    "tokenInfo":"{{ssm-secure:token-secure-string-parameter}}"
}
```

쿡북 소스로 HTTP 사용

Chef 쿡북을 사용자 지정 HTTP 위치에 단일 `.zip` 또는 `tar.gz` 파일이나 디렉터리 구조로 저장할 수 있습니다. HTTP에서 콘텐츠를 다운로드하려면 파일 또는 디렉터리 경로를 다음 예제와 같이 JSON 형식으로 지정합니다. 각 *example-resource-placeholder*를 자신의 정보로 바꿉니다.

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

Amazon S3를 쿡북 소스로 사용

Amazon S3에서 Chef 쿡북을 단일 `.zip` 또는 `tar.gz` 파일이나 디렉터리 구조로 저장하고 다운로드할 수도 있습니다. Amazon S3에서 콘텐츠를 다운로드하려면 파일 경로를 다음 예제와 같이 JSON 형식으로 지정합니다. 각 *example-resource-placeholder*를 자신의 정보로 바꿉니다.

예제 1: 특정 쿡북 다운로드

```
{
    "path":"https://s3.amazonaws.com/chef-cookbooks/HelloWorld.zip"
}
```

예 2: 디렉터리 콘텐츠 다운로드

```
{
    "path":"https://s3.amazonaws.com/chef-cookbooks-test/HelloWorld"
}
```

> **Important**
>
> Amazon S3를 지정하면 `AmazonS3ReadOnlyAccess` 정책을 사용하여 관리형 노드에 있는 AWS Identity and Access Management(IAM) 노드 프로파일을 구성해야 합니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성을 참조하세요.

주제

- Chef 레시피를 실행하는 연결 생성(콘솔)
- Chef 레시피를 실행하는 연결 생성(CLI)
- Chef 리소스 규정 준수 세부 정보 보기

Chef 레시피를 실행하는 연결 생성(콘솔)

다음 절차에서는 Systems Manager 콘솔을 사용하여 `AWS-ApplyChefRecipes` 문서로 Chef 쿡북을 실행하는 State Manager 연결을 생성하는 방법을 설명합니다.

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 State Manager를 선택합니다.
3. [State Manager]를 선택하고 [연결 생성(Create association)]을 선택합니다.
4. 이름에 연결의 목적을 기억하는 데 도움이 되는 이름을 입력합니다.
5. [문서(Document)] 목록에서 **`AWS-ApplyChefRecipes`**를 선택합니다.
6. 파라미터에서 소수 유형은 Git, GitHub, HTTP 또는 S3 중에서 선택합니다.
7. 소스 정보에는 6단계에서 선택한 소스 유형에 적합한 형식을 사용하여 쿡북 소스 정보를 입력합니다. 자세한 정보는 다음 주제를 참조하세요.
   - the section called “쿡북 소스로 Git 사용”
   - the section called “GitHub을 쿡북 소스로 사용”
   - the section called “쿡북 소스로 HTTP 사용”
   - the section called “Amazon S3를 쿡북 소스로 사용”
8. Run list(실행 목록)에서 다음과 같이 각 레시피를 쉼표로 구분하여 실행할 레시피를 나열합니다. 쉼표 뒤에 공백을 넣지 않습니다. 각 *`example-resource-placeholder`*를 자신의 정보로 바꿉니다.

```
recipe[cookbook-name1::recipe-name],recipe[cookbook-name2::recipe-name]
```

9. (선택 사항) Chef 클라이언트를 통해 대상 노드에 전달하려는 사용자 지정 JSON 속성을 지정합니다.

   a. JSON 속성 콘텐츠에서 Chef 클라이언트를 통해 대상 노드에 전달하려는 모든 속성을 추가합니다.

   b. JSON 속성 소스에서 Chef 클라이언트를 통해 대상 노드에 전달하려는 모든 속성의 경로를 추가합니다.

   자세한 내용은 the section called “레시피 실행 시 대상에 JSON 속성 적용” 단원을 참조하십시오.

10. Chef 클라이언트 버전에서 Chef 버전을 지정합니다. 유효한 값은 11~18 또는 None입니다. 11부터 18까지의 숫자를 지정하면 Systems Manager에서는 대상 노드에 올바른 Chef 클라이언트 버전을 설치합니다. None을 지정한 경우 Systems Manager는 문서의 레시피를 실행하기 전에 대상 노드에 Chef 클라이언트를 설치하지 않습니다.

11. (옵션) Chef 클라이언트 인수에서 사용 중인 Chef 버전에 대해 지원되는 추가 인수를 지정합니다. 지원되는 인수에 대한 자세한 내용을 보려면 Chef 클라이언트를 실행하는 노드에서 `chef-client -h`를 실행하세요.

12. (옵션) Why-run을 설정하여 대상 노드를 실제로 변경하지 않고 레시피를 실행할 경우 대상 노드에 적용되는 변경 사항을 표시합니다.

13. [규정 준수 심각도(Compliance severity)]에서 보고할 Systems Manager Compliance 결과의 심각도를 선택합니다. 규정 준수 보고는 지정한 심각도 수준과 함께 연결 상태가 규정을 준수하는지 여부를 나타냅니다. Compliance 보고서는 [Compliance 보고서 버킷(Compliance report bucket)] 파라미터 값(14단계)으로 지정하는 S3 버킷에 저장됩니다. Compliance에 대한 자세한 내용은 이 가이드의 Compliance 작업 섹션을 참조하세요.

    규정 준수 검사는 Chef 레시피 및 노드 리소스에 지정된 구성 간의 드리프트를 측정합니다. 유효한 값은 `Critical`, `High`, `Medium`, `Low`, `Informational`, `Unspecified` 또는 None입니다. 규정 준수 보고를 건너뛰려면 None을 선택합니다.

14. 규정 준수 유형에서 결과를 보고할 규정 준수 유형을 지정합니다. 유효한 값은 State Manager 연결의 경우 `Association` 또는 `Custom:`*`custom_type`*입니다. 기본 값은 `Custom:Chef`입니다.

15. Compliance 보고서 버킷의 경우 리소스 구성 및 Compliance 결과를 포함하여 이 문서에서 수행한 모든 Chef에 대한 정보를 저장할 S3 버킷의 이름을 입력합니다.

16. Rate control(속도 제어)에서 관리형 노드 플릿 간에 State Manager 연결을 실행하기 위한 옵션을 구성합니다. 속도 제어 사용에 대한 자세한 내용은 State Manager 연결에서의 대상 및 속도 제어 정보 섹션을 참조하세요.

    동시성에서 옵션을 선택합니다.

    - 대상을 선택하여 연결을 동시에 실행할 수 있는 대상 수(절대 개수)를 입력합니다.
    - 백분율을 선택하여 연결을 동시에 실행할 수 있는 대상의 백분율을 입력합니다.

    Error threshold(오류 임계값)에서 옵션을 선택합니다.

    - 오류를 선택하여 State Manager에서 추가 대상에 대한 연결 실행을 중지하기 전에 허용되는 절대 오류 수를 입력합니다.
    - 백분율을 선택하여 State Manager에서 추가 대상에 대한 연결 실행을 중지하기 전에 허용되는 오류 비율을 입력합니다.

17. (선택 사항) 출력 옵션에서 명령 출력을 파일에 저장하려면 S3 버킷에 쓰기 활성화 옆의 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

    > **Note**
    >
    > 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아닌 관리형 노드에 할당된 인스턴스 프로파일의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

18. 연결 생성을 선택합니다.

Chef 레시피를 실행하는 연결 생성(CLI)

다음 절차에서는 AWS Command Line Interface(AWS CLI)를 사용하여 `AWS-ApplyChefRecipes` 문서로 Chef 쿡북을 실행하는 State Manager 연결을 생성하는 방법을 설명합니다.

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

    자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.

2. 다음과 같은 명령 중 하나를 실행하여 지정된 태그가 있는 대상 노드에서 Chef 쿡북을 실행하는 연결을 생성합니다. 쿡북 소스 유형 및 운영 체제에 적합한 명령을 사용합니다. 각 *example-resource-placeholder*를 자신의 정보로 바꿉니다.

   a. Git 소스

   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 소스

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"
```

다음 예를 참고하세요

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 소스

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 소스

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": ["DOC-EXAMPLE-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": ["DOC-EXAMPLE-BUCKET"]}' ^
    --association-name "name" ^
    --schedule-expression "cron_or_rate_expression"
```

다음 예를 참고하세요

Linux & macOS

```
aws ssm create-association --name "AWS-ApplyChefRecipes" \
    --targets "Key=tag:OS,Values= Linux" \
    --parameters '{"SourceType":["S3"],"SourceInfo":["{\"path
\":\"https://s3.amazonaws.com/DOC-EXAMPLE-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/DOC-EXAMPLE-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 *)"
```

시스템에서 연결이 생성되며, 지정된 cron 또는 rate 표현식이 이를 방해하지 않는 한 시스템에서는 대상 노드에서 연결을 실행합니다.

> **Note**
>
> State Manager 연결은 cron 및 rate 표현식 중 일부를 지원하지 않습니다. 연결에 대한 cron 및 rate 표현식을 생성하는 방법에 대한 자세한 내용은 참조: Systems Manager용 Cron 및 Rate 표현식 섹션을 참조하세요.

3. 다음 명령을 실행하여 방금 생성한 연결의 상태를 봅니다.

```
aws ssm describe-association --association-id "ID"
```

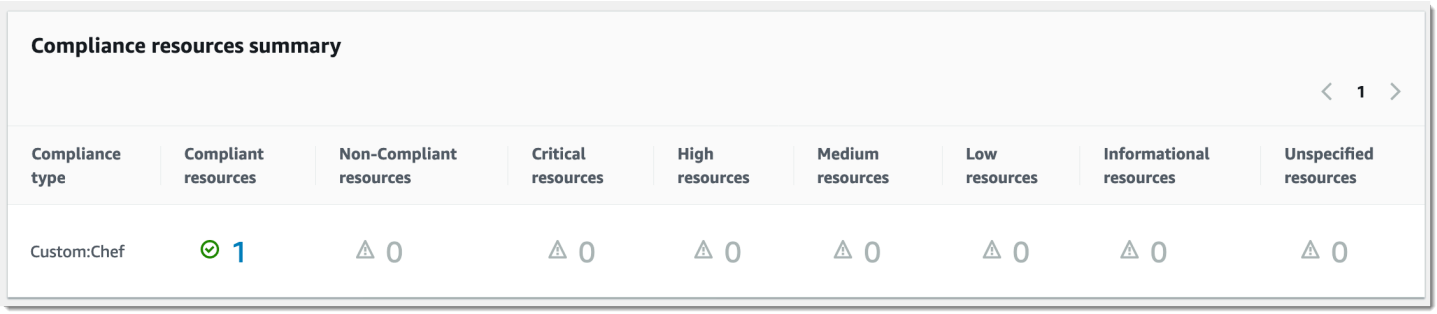
Chef 리소스 규정 준수 세부 정보 보기

Systems Manager는 `AWS-ApplyChefRecipes` 문서를 실행했을 때 지정한 Amazon S3 Compliance 보고서 버킷 값에서 Chef 관리형 리소스에 대한 규정 준수 정보를 수집합니다. S3 버킷에서 Chef 리소스 실패에 대한 정보를 검색하는 데는 시간이 많이 걸릴 수 있습니다. 대신, Systems Manager [Compliance] 페이지에서 이 정보를 볼 수 있습니다.

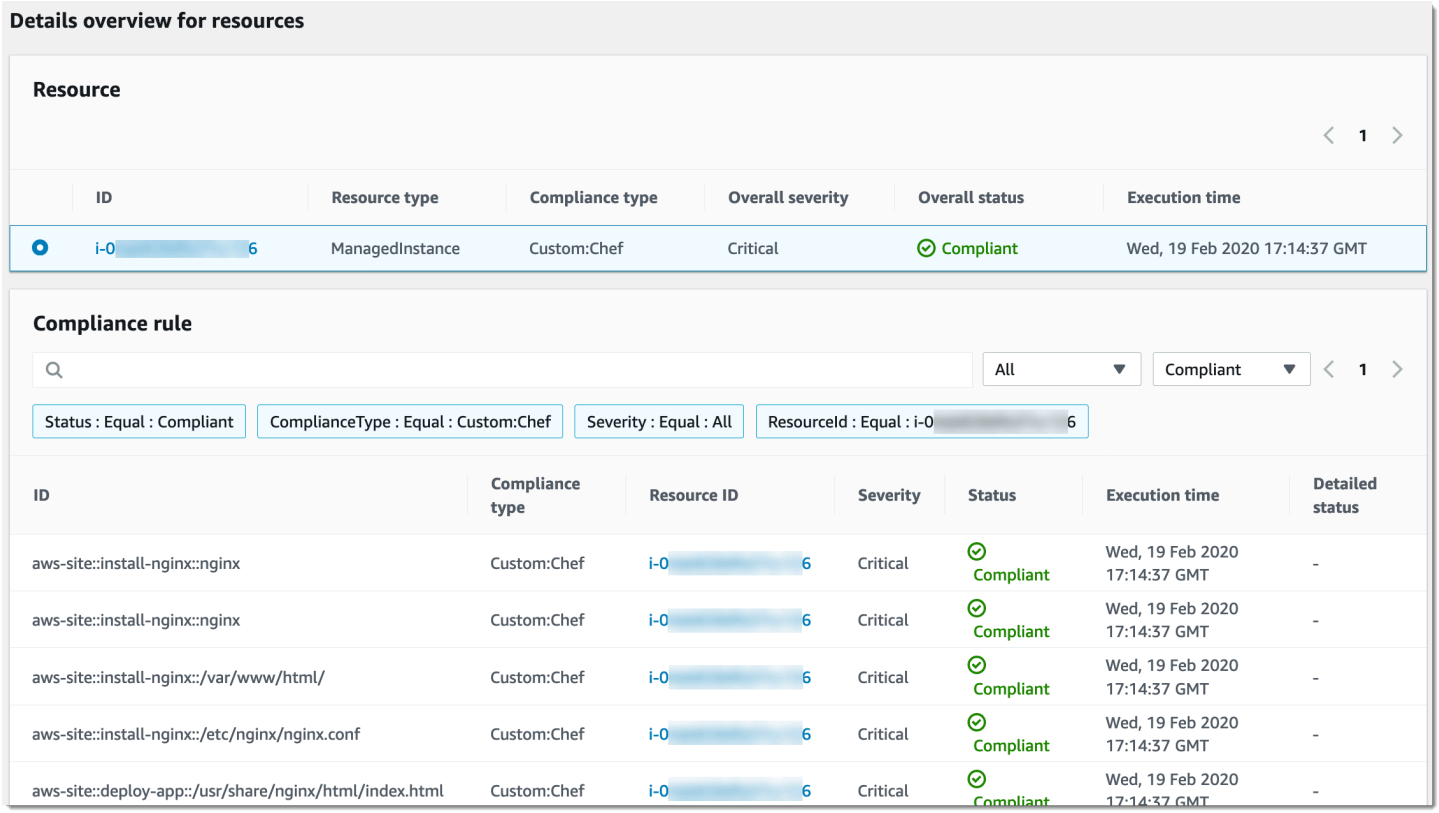
Systems Manager Compliance 검사는 가장 최근의 Chef 실행에서 생성되거나 확인된 관리형 노드의 리소스에 대한 정보를 수집합니다. 리소스에는 파일, 디렉터리, systemd 서비스, yum 패키지, 템플릿 파일, gem 패키지 및 종속 쿡북 등이 포함될 수 있습니다.

규정 준수 리소스 요약 섹션에 실패한 리소스의 수가 표시됩니다. 다음 예에서 ComplianceType은 Custom:Chef이고 한 개의 리소스가 규정 미준수입니다.

> **Note**
>
> Custom:Chef는 AWS-ApplyChefRecipes 문서의 기본 ComplianceType 값입니다. 이 값은 사용자 지정 가능합니다.



[리소스에 대한 세부 정보 개요(Details overview for resources)] 섹션에 규정 미준수 AWS 리소스에 대한 정보가 표시됩니다. 이 섹션에는 규정 준수가 실행된 Chef 리소스 유형, 문제의 심각도, 규정 준수 상태 및 추가 정보(해당하는 경우)에 대한 링크도 포함됩니다.



[출력 보기(View output)]는 세부 상태의 마지막 4,000자를 표시합니다. Systems Manager는 예외를 첫 번째 요소로 시작하여 자세한 메시지를 찾아 4,000자 할당량에 도달할 때까지 표시합니다. 이 프로세스는 예외가 발생되기 전에 출력된 긴 메시지(문제 해결을 위해 가장 관련성이 높은 메시지)를 표시합니다.

규정 준수 정보를 보는 방법에 대한 자세한 내용은 AWS Systems Manager Compliance 섹션을 참조하세요.

연결 실패가 규정 준수 보고에 미치는 영향

State Manager 연결이 실패한 경우에는 규정 준수 데이터가 보고되지 않습니다. 예를 들어, Systems Manager가 노드에 액세스할 권한이 없는 S3 버킷에서 Chef 쿡북을 다운로드하려고 하면 연결이 실패하고 Systems Manager가 어떠한 규정 준수 데이터도 보고되지 않습니다.

시연: SSM Agent(CLI) 자동 업데이트

다음 절차는 AWS Command Line Interface를 사용하여 State Manager 연결을 생성하는 과정을 안내합니다. 연결하면 지정한 일정에 따라 SSM Agent가 자동으로 업데이트됩니다. SSM Agent에 대한 자세한 내용은 SSM Agent 작업 섹션을 참조하세요. 콘솔을 사용하여 SSM Agent의 업데이트 일정을 사용자 지정하려면 SSM Agent 자동 업데이트 섹션을 참조하세요.

SSM Agent 업데이트에 대해 알림을 수신하려면 GitHub에서 SSM Agent 릴리스 정보 페이지를 구독합니다.

시작하기 전 준비 사항

다음 절차를 완료하기 전에 Systems Manager에 대해 구성된 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스(Linux, macOS 또는 Windows Server용)가 1개 이상 실행되고 있는지 확인합니다. 자세한 내용은 AWS Systems Manager 설정 단원을 참조하십시오.

AWS CLI 또는 AWS Tools for Windows PowerShell 사용을 통해 연결을 생성한 경우 다음 예제에 표시된 것과 같이 `--Targets` 파라미터를 사용하여 인스턴스를 대상으로 지정합니다. `--InstanceID` 파라미터를 사용하지 마십시오. `--InstanceID` 파라미터는 레거시 파라미터입니다.

SSM Agent를 자동으로 업데이트하기 위해 연결을 생성하려면

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.
2. 다음 명령을 실행하면 Amazon Elastic Compute Cloud(Amazon EC2) 태그로 인스턴스를 대상으로 지정하여 연결을 생성합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다. `Schedule` 파라미터는 매주 일요일 오전 2시(UTC)에 연결을 실행하는 일정을 설정합니다.

   State Manager 연결은 cron 및 rate 표현식 중 일부를 지원하지 않습니다. 연결에 대한 cron 및 rate 표현식을 생성하는 방법에 대한 자세한 내용은 참조: Systems Manager용 Cron 및 Rate 표현식 섹션을 참조하세요.

   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 *)"
   ```

쉼표로 구분된 목록으로 인스턴스 ID를 지정하여 여러 인스턴스를 대상으로 지정할 수도 있습니다.

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 *)"
```

업데이트하려는 SSM Agent의 버전을 지정할 수 있습니다.

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
```

시스템은 다음과 같은 정보를 반환합니다.

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

시스템은 해당 인스턴스에 연결 생성을 시도하고 생성 후 상태를 적용합니다. 연결 상태는 `Pending`으로 표시됩니다.

3. 다음 명령을 실행하여 생성한 연결의 업데이트된 상태를 확인합니다.

```
aws ssm list-associations
```

인스턴스에서 최신 버전의 SSM Agent를 실행하고 있지 않은 경우 상태가 `Failed`로 표시됩니다. 최신 버전의 SSM Agent가 발행되면 이 연결을 통해 새 에이전트가 자동으로 설치되고 상태가 `Success`로 표시됩니다.

연습: Windows Server용 EC2 인스턴스에서 PV 드라이버 자동 업데이트(콘솔)

Amazon Windows Amazon Machine Images(AMIs)는 가상화 하드웨어에 대한 액세스를 허용하는 드라이버 집합을 포함하고 있습니다. 이러한 드라이버는 Amazon Elastic Compute Cloud(Amazon EC2)에서 인스턴스 스토어 및 Amazon Elastic Block Store(Amazon EBS) 볼륨을 디바이스에 매핑하는 데

사용됩니다. Windows Server용 EC2 인스턴스의 안정성과 성능을 개선하려면 최신 드라이버를 설치하는 것이 좋습니다. PV 드라이버에 대한 자세한 내용은 AWS PV 드라이버를 참조하세요.

다음 시연에서는 드라이버를 사용할 수 있는 경우 새 AWS PV 드라이버를 자동으로 다운로드하여 설치하도록 State Manager 연결을 구성하는 방법을 보여줍니다. State Manager는 AWS Systems Manager의 기능입니다.

시작하기 전 준비 사항

다음 절차를 완료하기 전에 Systems Manager에 대해 구성된 Windows Server용 Amazon EC2 인스턴스가 1개 이상 실행되고 있는지 확인합니다. 자세한 내용은 AWS Systems Manager 설정 단원을 참조하십시오.

PV 드라이버를 자동으로 업데이트하는 State Manager 연결을 생성하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 State Manager를 선택합니다.
3. 연결 생성을 선택합니다.
4. 이름 필드에 연결에 대한 설명이 포함된 이름을 입력합니다.
5. [문서(Document)] 목록에서 `AWS-ConfigureAWSPackage`를 선택합니다.
6. 파라미터 섹션에서 다음을 수행합니다.
   - 작업에서 설치를 선택합니다.
   - [설치 유형(Installation type)]에서 [제거 및 다시 설치(Uninstall and reinstall)]를 선택합니다.

     > **Note**
     > 이 패키지에는 현재 위치 업그레이드가 지원되지 않습니다. 제거한 후 다시 설치해야 합니다.

   - 이름에 **`AWSPVDriver`**를 입력합니다.

     버전 및 추가 인수에는 아무 것도 입력할 필요가 없습니다.
7. Targets(대상) 섹션에서, 태그를 지정하거나, 수동으로 인스턴스나 엣지 디바이스를 선택하거나, 리소스 그룹을 지정하여 이 작업을 실행할 관리형 노드를 식별합니다.

> **Tip**
>
> 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

> **Note**
>
> 태그를 사용하여 인스턴스 대상을 지정하고 Linux 인스턴스에 매핑되는 태그를 지정할 경우 Windows 인스턴스에서는 연결이 성공하지만 Linux 인스턴스에서는 실패합니다. 전체 연결 상태는 실패로 표시됩니다.

8. 일정 지정 영역에서 구성한 일정에 따라 연결을 실행할지 아니면 한 번만 실행할지 선택합니다. 업데이트된 PV 드라이버는 1년에 몇 번 릴리스되므로 원하는 경우 연결을 한 달에 한 번 실행하도록 예약할 수 있습니다.
9. 고급 옵션의 규정 준수 심각도에서 연결의 심각도 수준을 선택합니다. 규정 준수 보고는 여기서 지정한 심각도 수준과 함께 연결 상태가 준수인지 아니면 미준수인지를 나타냅니다. 자세한 내용은 State Manager 연결 규정 준수 정보 단원을 참조하십시오.
10. Rate control(속도 제어)에서

    - Concurrency(동시성)에서 명령을 동시에 실행할 관리형 노드의 백분율 또는 개수를 지정합니다.

      > **Note**
      >
      > 관리형 노드에 적용할 태그를 지정하거나, AWS 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상 수를 제한합니다.

    - Error threshold(오류 임계값)에서, 명령이 노드의 개수 또는 백분율에서 실패한 후 다른 관리형 노드에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째 오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.
11. (선택 사항) 출력 옵션에서 명령 출력을 파일에 저장하려면 S3 버킷에 쓰기 활성화 옆의 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

> **Note**
>
> 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아닌 관리형 노드에 할당된 인스턴스 프로파일의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

12. (선택 사항) CloudWatch 경보 섹션에서 모니터링을 위해 연결에 적용할 CloudWatch 경보를 경보 이름으로 선택합니다.

> **Note**
>
> 이 단계에 대한 다음 정보를 참조하십시오.
>
> - 알람 목록에는 최대 100개의 알람이 표시됩니다. 목록에 해당 경보가 표시되지 않으면 AWS Command Line Interface를 사용하여 연결을 생성합니다. 자세한 내용은 연결 생성(명령줄) 단원을 참조하십시오.
> - CloudWatch 경보를 명령에 연결하려면 연결을 생성하는 IAM 보안 주체에 `iam:createServiceLinkedRole` 작업에 대한 권한이 있어야 합니다. CloudWatch 경보에 대한 자세한 내용은 Amazon CloudWatch 경보 사용을 참조하세요.
> - 경보가 활성화되면 보류 중인 명령 호출 또는 자동화가 실행되지 않습니다.

13. [연결 생성(Create association)], [닫기(Close)]를 차례로 선택합니다. 시스템은 해당 인스턴스에 연결을 생성하고 그 상태를 즉시 적용하려고 합니다.

    1개 이상의 Windows Server용 Amazon EC2 인스턴스에 대해 연결을 생성한 경우 상태가 [성공(Success)]으로 변경됩니다. Systems Manager에 대해 인스턴스를 구성하지 않았거나 실수로 Linux 인스턴스를 대상으로 지정한 경우 상태가 [실패(Failed)]로 표시됩니다.

    상태가 [실패(Failed)]인 경우 연결 ID와 [리소스(Resources)] 탭을 차례로 선택하고 Windows Server용 EC2 인스턴스에서 연결이 성공적으로 생성되었는지 확인합니다. Windows Server용 EC2 인스턴스의 상태가 [실패(Failed)]로 표시될 경우 SSM Agent가 인스턴스에서 실행 중이고 Systems Manager에 대한 AWS Identity and Access Management(IAM) 역할로 인스턴스가 구성되었는지 확인합니다. 자세한 내용은 AWS Systems Manager 설정 단원을 참조하십시오.

AWS Systems Manager Patch Manager

AWS Systems Manager의 기능인 Patch Manager는 보안 관련 업데이트 및 기타 유형의 업데이트로 관리형 노드를 패치하는 프로세스를 자동화합니다.

> **Important**
>
> 2022년 12월 22일부터 Systems Manager는 새롭고 권장되는 패치 작업 구성 방법인 패치 정책에 대한 지원을 제공합니다. 단일 패치 정책 구성을 사용하여 조직 내 모든 리전의 모든 계정이나, 선택한 계정 및 리전이나, 단일 계정-리전 쌍에 대한 패치 적용을 정의할 수 있습니다. 자세한 내용은 Quick Setup 패치 정책 사용 단원을 참조하십시오.

Patch Manager를 사용하면 운영 체제와 애플리케이션 모두를 패치할 수 있습니다. (Windows Server에서 애플리케이션 지원은 Microsoft에서 릴리스한 애플리케이션의 업데이트로 제한됩니다.) Patch Manager를 사용하여 Windows 노드에 서비스 팩을 설치하고 Linux 노드에서 부 버전 업그레이드를 실행할 수 있습니다. 운영 체제 유형별로 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스, 엣지 디바이스, 온프레미스 서버 및 가상 머신의 플릿에 패치를 적용할 수 있습니다. 여기에는 Patch Manager 필수 조건에 나와 있는 여러 운영 체제의 지원되는 버전이 포함됩니다. 인스턴스를 스캔하여 누락된 패치 보고서만 보거나, 혹은 스캔 후 누락된 패치를 모두 자동으로 설치할 수도 있습니다. Patch Manager를 시작하려면 Systems Manager 콘솔을 엽니다. 탐색 창에서 Patch Manager를 선택합니다.

> **Note**
>
> AWS는 Patch Manager에서 사용 가능성 여부를 확인하기 전에는 패치를 테스트하지 않습니다. 또한 Patch Manager는 주요 버전의 운영 체제 업그레이드를 지원하지 않습니다(예: Windows Server 2016에서 Windows Server 2019로 또는 (SUSE Linux Enterprise Server) (SLES) 12.0에서 SLES 15.0으로).
>
> 패치의 심각도 수준을 보고하는 Linux 기반 운영 체제 유형의 경우 Patch Manager는 업데이트 알림 또는 개별 패치에 대해 소프트웨어 게시자가 보고한 심각도 수준을 사용합니다. Patch Manager는 CVSS(Common Vulnerability Scoring System)와 같은 서드 파티 소스 또는 NVD(National Vulnerability Database)에서 발표한 지표에서 심각도 수준을 도출하지 않습니다.

패치 기준

Patch Manager는 승인 및 거부된 패치 목록(선택 사항)과 함께 릴리스 후 며칠 내에 패치를 자동 승인하는 규칙을 포함하는 패치 기준선을 사용합니다. 패치 작업이 실행되면 Patch Manager는 관리형 노드에 현재 적용된 패치를 패치 기준선에 설정된 규칙에 따라 적용해야 하는 패치와 비교합니다. Patch Manager가 누락된 패치에 대한 보고서만 표시하도록 선택하거나(`Scan` 작업), Patch Manager가 관리형 노드에서 누락된 모든 패치를 자동으로 설치하도록 선택할 수 있습니다(`Scan and install` 작업).

패치 작업 방법

Patch Manager는 현재 실행 `Scan` 및 `Scan and install` 작업을 위한 네 가지 방법을 제공합니다.

- (권장) Quick Setup에 구성된 패치 정책 - AWS Organizations와의 통합을 기반으로 하는 단일 패치 정책으로 여러 AWS 계정 계정 및 해당 계정을 운영하는 모든 AWS 리전 계정을 비롯하여 전체 조직의 패치 적용 일정과 패치 기준선을 정의할 수 있습니다. 패치 정책은 조직의 일부 조직 단위(OU)만 대상으로 할 수도 있습니다. 단일 패치 정책을 사용하여 다양한 일정에 따라 스캔하고 설치할 수 있습니다. 자세한 내용은 Patch Manager 조직 패치 적용 구성 및 Quick Setup 패치 정책 사용 단원을 참조하세요.
- Quick Setup에 구성된 호스트 관리 옵션 - 호스트 관리 구성도 AWS Organizations와의 통합을 통해 지원되므로, 최대 전체 조직을 대상으로 패치 작업을 실행할 수 있습니다. 단, 이 옵션은 현재 기본 패치 기준선을 사용하여 누락된 패치를 스캔하고 규정 준수 보고서에 결과를 제공하는 것으로 제한됩니다. 이 작업 방법으로 패치를 설치할 수는 없습니다. 자세한 내용은 Amazon EC2 호스트 관리 단원을 참조하십시오.
- 패치 **`Scan`** 또는 **`Install`** 태스크를 실행하기 위한 유지 관리 기간 - Maintenance Windows라는 Systems Manager 기능에서 설정하는 유지 관리 기간은 사용자가 정의한 일정에 따라 다양한 유형의 태스크를 실행하도록 구성할 수 있습니다. Run Command 유형 태스크는 선택한 관리형 노드 세트에 대해 `Scan` 또는 `Scan and install` 태스크를 실행하는 데 사용할 수 있습니다. 각 유지 관리 기간 태스크는 단일 AWS 계정-AWS 리전 쌍의 관리형 노드만 대상으로 할 수 있습니다. 자세한 내용은 패치에 대한 유지 관리 기간 생성(콘솔) 단원을 참조하십시오.
- Patch Manager의 온디맨드 'Patch now(지금 패치)' 작업 - Patch now(지금 패치) 옵션을 사용하면 관리형 노드를 최대한 신속하게 패치해야 할 때 일정 설정을 무시할 수 있습니다. Patch now(지금 패치)를 사용하여, `Scan` 또는 `Scan and install` 작업을 실행할지 여부와 작업을 실행할 대상 관리형 노드를 지정합니다. 패치 작업 중에 Systems Manager 문서(SSM 문서)를 수명 주기 후크로 실행하도록 선택할 수도 있습니다. 각각의 Patch now(지금 패치) 작업은 단일 AWS 계정-AWS 리전 쌍의 관리형 노드만 대상으로 할 수 있습니다. 자세한 내용은 관리형 노드 온디맨드 패치 단원을 참조하십시오.

규정 준수 보고

Scan작업이 끝나면 Systems Manager 콘솔을 사용하여 어떤 관리형 노드가 패치 규정을 위반하지 않는지, 그리고 각 노드에서 어떤 패치가 누락되었는지에 대한 정보를 확인할 수 있습니다. 선택한 Amazon Simple Storage Service(S3) 버킷으로 전송되는 패치 규정 준수 보고서를 .csv 형식으로 생성할 수도 있습니다. 일회성 보고서를 생성하거나 정기적인 일정에 따라 보고서를 생성할 수 있습니다. 단일 관리형 노드의 경우 보고서에 노드에 대한 모든 패치의 세부 정보가 포함됩니다. 모든 관리형 노드에 대한 보고서의 경우 누락된 패치 수에 대한 요약만 제공됩니다. 보고서가 생성된 후 Amazon QuickSight와 같은 도구를 사용하여 데이터를 가져오고 분석할 수 있습니다. 자세한 내용은 패치 규정 준수 보고서 작업 단원을 참조하십시오.

> **Note**
>
> 패치 정책을 사용하여 생성된 규정 준수 항목의 실행 유형은 PatchPolicy입니다. 패치 정책 작업에서 생성도되지 않은 규정 준수 항목의 실행 유형은 Command입니다.

통합

Patch Manager는 다음과 같은 다른 AWS 서비스와 통합됩니다.

- AWS Identity and Access Management(IAM) - IAM을 사용하여 Patch Manager 작업에 액세스할 수 있는 사용자, 그룹 및 역할을 제어할 수 있습니다. 자세한 내용은 AWS Systems Manager에서 IAM을 사용하는 방식 및 Systems Manager에 필요한 인스턴스 권한 구성을 참조하세요.
- AWS CloudTrail - CloudTrail을 사용하여 사용자, 역할 또는 그룹에 의해 시작된 패치 작업 이벤트의 감사 가능한 기록을 기록할 수 있습니다. 자세한 내용은 AWS CloudTrail을 사용하여 AWS Systems ManagerAPI 호출을 로깅 단원을 참조하십시오.
- AWS Security Hub - Patch Manager에서 패치 규정 준수 데이터를 AWS Security Hub로 보낼 수 있습니다. Security Hub에서는 우선순위가 높은 보안 알림 및 규정 준수 상태를 포괄적으로 파악할 수 있습니다. 또한 플릿의 패치 상태를 모니터링합니다. 자세한 내용은 Patch Manager와 AWS Security Hub 통합 단원을 참조하십시오.
- AWS Config - Patch Manager 대시보드에서 Amazon EC2 인스턴스 관리 데이터를 볼 수 있도록 AWS Config에서 기록을 설정합니다. 자세한 내용은 패치 대시보드 요약 보기 단원을 참조하십시오.

주제

- Quick Setup 패치 정책 사용
- Patch Manager 필수 조건
- Patch Manager 작업 작동 방식

- 관리형 노드 패치를 위한 SSM 문서 정보
- 패치 기준 정보
- Amazon Linux 2 관리형 노드에서 Kernel Live Patching 사용
- Patch Manager 작업(콘솔)
- Patch Manager(AWS CLI) 작업
- AWS Systems Manager Patch Manager 자습서
- Patch Manager 문제 해결

Quick Setup 패치 정책 사용

2022년 12월 22일부터 Patch Manager는 패치 정책을 사용하여 조직과 AWS 계정의 패치 적용을 구성할 수 있는 새로운 권장 방법을 제공합니다.

패치 정책은 AWS Systems Manager의 기능인 Quick Setup을 사용하여 설정하는 구성입니다. 패치 정책은 이전 패치 구성 방법에서 제공되는 것보다 더 광범위하고 중앙 집중화된 패치 작업 제어 기능을 제공합니다. 패치 정책은 지원되는 Linux, macOS 및 Windows Server 버전을 포함하여 Patch Manager에서 지원되는 모든 운영 체제에서 사용할 수 있습니다. 패치 정책에 대한 자세한 내용은 Patch Manager 조직 패치 적용 구성 섹션을 참조하세요.

패치 정책의 주요 기능

노드를 패치하는 다른 방법을 사용하는 대신, 패치 정책을 사용하여 다음과 같은 주요 기능의 이점을 활용하세요.

- 단일 설정 - 유지 관리 기간 또는 State Manager 연결을 사용하여 패치 작업을 설정하려면 Systems Manager 콘솔의 여러 부분에서 여러 태스크를 수행해야 할 수 있습니다. 패치 정책을 사용하면 모든 패치 작업을 단일 마법사에서 설정할 수 있습니다.
- 다중 계정/다중 리전 지원 - Patch Manager에서 유지 관리 기간, State Manager 연결 또는 Patch now(지금 패치) 기능을 사용하면 단일 AWS 계정-AWS 리전 쌍의 관리형 노드에 대해서만 작업을 수행할 수 있습니다. 따라서 여러 계정과 여러 리전을 사용하는 경우, 각 계정-리전 쌍에서 설정 작업을 수행해야 하므로 설정 및 유지 관리 작업에 시간이 많이 소요될 수 있습니다. 반면, AWS Organizations를 사용하면 모든 AWS 리전에서 모든 AWS 계정의 모든 관리형 노드에 적용되는 단일 패치 정책을 설정할 수 있습니다. 또는 원하는 경우 계정 및 리전에서 선택한 일부 조직 단위(OU)에만 패치 정책을 적용할 수도 있습니다. 원하는 경우 패치 정책을 단일 로컬 계정에도 적용할 수 있습니다.

- 조직 차원의 설치 지원 - Quick Setup의 기존 호스트 관리 구성 옵션은 관리형 노드의 패치 규정 준수를 매일 검사할 수 있도록 지원합니다. 하지만 이 검사은 미리 정해진 시간에 수행되며, 패치 규정 준수 정보만 제공합니다. 패치 설치 작업은 수행되지 않습니다. 패치 정책을 사용하여 다양한 스캐닝 및 설치 일정을 지정할 수 있습니다. 사용자 지정 CRON 또는 Rate 표현식을 사용하여 이들 작업의 빈도와 시간을 선택할 수도 있습니다. 예를 들어 매일 누락된 패치를 검사하여 정기적으로 업데이트되는 규정 준수 정보를 제공할 수 있습니다. 하지만 원치 않는 가동 중단을 방지하기 위해, 설치 일정을 일주일에 한 번으로 정할 수도 있습니다.
- 간소화된 패치 기준선 선택 - 패치 정책에도 패치 기준선이 포함되어 있으며, 패치 기준선이 구성되는 방식에는 변화가 없습니다. 단, 패치 정책을 생성하거나 업데이트할 때, 각 운영 체제(OS) 유형에 사용할 AWS 관리형 기준선 또는 사용자 지정 기준선을 단일 목록에서 선택할 수 있습니다. 태스크마다 각 OS 유형에 대한 기본 기준선을 지정할 필요는 없습니다.

> **Note**
>
> 패치 정책 실행을 기반으로 패치 작업을 수행할 때는 `AWS-RunPatchBaseline` SSM 문서를 사용합니다. 자세한 내용은 `AWS-RunPatchBaseline` SSM 문서 정보 단원을 참조하십시오.

관련 정보

Systems Manager Quick Setup을 사용하여 중앙에서 AWS 조직 전체에 패치 적용 작업 배포(AWS 클라우드 운영 및 마이그레이션 블로그)

패치 정책을 사용할 경우의 기타 차이점

이전 패치 구성 방법 대신 패치 정책을 사용할 때 주의해야 할 몇 가지 다른 차이점은 다음과 같습니다.

- 패치 그룹이 필요 없음 - 이전 패치 작업에서는 패치 그룹에 속하도록 여러 노드에 태그를 지정한 다음, 해당 패치 그룹에 사용할 패치 기준선을 지정할 수 있었습니다. 패치 그룹이 없으면 Patch Manager가 OS 유형의 현재 기본 패치 기준선을 사용하여 인스턴스에 패치를 적용했습니다. 패치 정책을 사용하면 더 이상 패치 그룹을 설정하고 유지 관리할 필요가 없습니다.
- 'Configure patching(패치 구성)' 페이지 제거 - 패치 정책이 릴리스되기 전에는 Configure patching(패치 구성) 페이지에서 패치를 적용할 노드, 패치 일정 및 패치 작업에 대한 기본값을 지정할 수 있었습니다. 이 페이지가 Patch Manager에서 제거되었습니다. 이들 옵션은 이제 패치 정책에서 지정합니다.
- 'Patch now(지금 패치)' 지원 안 함 - 온디맨드로 노드를 패치 기능은 여전히 한 번에 하나의 AWS 리전-AWS 계정 쌍으로 제한됩니다. 자세한 설명은 관리형 노드 온디맨드 패치을 참조하세요.

- 패치 정책 및 규정 준수 정보 - 패치 정책 구성에 따라 관리형 노드의 규정 준수를 검사하면, 규정 준수 데이터가 사용자에게 제공됩니다. 다른 규정 준수 검사 방법과 동일한 방식으로 데이터를 이 보고 관련 작업을 수행할 수 있습니다. 전체 조직 또는 여러 조직 단위에 대해 패치 정책을 설정할 수 있지만, 규정 준수 정보는 각 AWS 계정-AWS 리전 쌍에 대해 개별적으로 보고됩니다. 자세한 내용은 패치 규정 준수 보고서 작업 단원을 참조하십시오.
- 연결 규정 준수 상태 및 패치 정책-Quick Setup 패치 정책이 적용되는 관리형 노드의 패치 상태는 해당 노드의 State Manager 연결 실행 상태와 일치합니다. 연결 실행 상태가 `Compliant`인 경우 관리형 노드의 패치 상태가 `Compliant`로 표시됩니다. 연결 실행 상태가 `Non-Compliant`인 경우 관리형 노드의 패치 상태가 `Non-Compliant`로 표시됩니다.

패치 정책이 지원되는 AWS 리전

Quick Setup의 패치 정책 구성은 현재 다음 리전에서 지원됩니다.

- 미국 동부(오하이오)(us-east-2)
- 미국 동부(버지니아 북부)(us-east-1)
- 미국 서부(캘리포니아 북부) (us-west-1)
- 미국 서부(오레곤)(us-west-2)
- 아시아 태평양(뭄바이)(ap-south-1)
- 아시아 태평양(서울)(ap-northeast-2)
- 아시아 태평양(싱가포르)(ap-southeast-1)
- 아시아 태평양(시드니)(ap-southeast-2)
- 아시아 태평양(도쿄)(ap-northeast-1)
- 캐나다(중부)(ca-central-1)
- 유럽(프랑크푸르트)(eu-central-1)
- 유럽(아일랜드)(eu-west-1)
- 유럽(런던) (eu-west-2)
- 유럽(파리)(eu-west-3)
- 유럽(스톡홀름)(eu-north-1)
- 남아메리카(상파울루)(sa-east-1)

Patch Manager 필수 조건

AWS Systems Manager의 기능인 Patch Manager를 사용하기 전에 필수 사전 조건을 충족했는지 확인합니다.

주제

SSM Agent 버전

버전 2.0.834.0 이상의 SSM Agent가 Patch Manager로 관리형 노드에서 실행 중이어야 합니다.

> **Note**
>
> SSM Agent의 업데이트된 버전은 Systems Manager에 새 기능이 추가되거나 기존 기능에 업데이트가 발생할 때마다 릴리스됩니다. 최신 버전의 에이전트를 사용하지 못하면 관리형 노드에서 다양한 Systems Manager 기능을 사용하지 못할 수 있습니다. 이러한 이유로 시스템의 SSM Agent 상태를 최신 상태로 유지하는 프로세스를 자동화하는 것이 좋습니다. 자세한 설명은 SSM Agent 업데이트 자동화을 참조하세요. SSM Agent 업데이트에 대해 알림을 수신하려면 GitHub에서 SSM Agent 릴리스 정보 페이지를 구독합니다.

Python 버전

macOS 및 대부분의 리눅스 운영체제(OS)의 경우 Patch Manager에서는 현재 Python 버전 2.6~3.10이 지원됩니다. AlmaLinux, Debian Server, Raspberry Pi OS 및 Ubuntu Server OS에는 지원되는 Python 3 버전(3.0~3.10)이 필요합니다.

패치 소스에 대한 연결

관리형 노드가 인터넷에 직접 연결되어 있지 않고 VPC 엔드포인트가 있는 Amazon Virtual Private Cloud(Amazon VPC)를 사용하는 경우 노드가 소스 패치 리포지토리에 액세스할 수 있는지 확인해야

합니다. Linux 노드에서 패치 업데이트는 일반적으로 노드에 구성된 원격 리포지토리에서 다운로드됩니다. 따라서, 노드에서 리포지토리에 연결할 수 있어야 패치를 적용할 수 있습니다. 자세한 내용은 보안 패치 선택 방법 단원을 참조하십시오.

Windows Server 관리형 노드는 Windows 업데이트 카탈로그 또는 Windows Server Update Services(WSUS)에 연결할 수 있어야 합니다. 노드가 인터넷 게이트웨이, NAT 게이트웨이 또는 NAT 인스턴스를 통해 Microsoft Update 카탈로그에 연결되어 있는지 확인합니다. WSUS를 사용하는 경우 노드가 환경의 WSUS 서버에 연결되어 있는지 확인합니다. 자세한 내용은 문제: 관리형 노드에 Windows 업데이트 카탈로그 또는 WSUS에 대한 액세스 권한이 없습니다. 단원을 참조하십시오.

S3 엔드포인트 액세스

관리형 노드가 사설 네트워크에서 작동하는지 아니면 퍼블릭 네트워크에서 작동하는지에 관계없이 필수 AWS 관리형 Amazon Simple Storage Service(Amazon S3) 버킷에 액세스하지 못하면 패치 작업이 실패합니다. 관리형 노드가 액세스할 수 있어야 하는 S3 버킷에 대한 자세한 내용은 AWS 관리형 S3 버킷과 SSM Agent 통신 및 Systems Manager용 VPC 엔드포인트를 사용하여 EC2 인스턴스의 보안 개선 섹션을 참조하세요.

Patch Manager에 지원되는 운영 체제

Patch Manager 기능이 다른 Systems Manager 기능이 지원했던 동일한 운영 체제 버전을 모두 지원하는 것은 아닙니다. 예를 들어, Patch Manager는 CentOS 6.3 또는 Raspberry Pi OS 8(Jessie)을 지원하지 않습니다. Systems Manager에서 지원되는 운영 체제의 전체 목록은 Systems Manager가 지원되는 운영 체제 섹션을 참조하세요. 따라서 Patch Manager에서 사용하려는 관리형 노드에 다음 표에 나열된 운영 체제 중 하나를 실행 중인지 확인합니다.

> **Note**
>
> Patch Manager는 Windows용 Windows Update 카탈로그 및 Windows Server 업데이트 서비스와 같이 관리형 노드에 구성된 패치 리포지토리에 의존하여 설치 가능한 패치를 검색합니다. 따라서 EOL(For End of Life) 운영 체제 버전의 경우 사용 가능한 새 업데이트가 없으면 Patch Manager에서 새 업데이트를 보고하지 못할 수 있습니다. 이는 Linux 배포 관리자, Microsoft 또는 Apple에서 새 업데이트를 릴리스하지 않았기 때문이거나 관리형 노드에 새 업데이트에 액세스할 수 있는 적절한 라이선스가 없기 때문일 수 있습니다.
>
> Patch Manager는 관리형 노드에서 사용 가능한 패치를 기준으로 규정 준수 상태를 보고합니다. 따라서 인스턴스가 EOL 운영 체제를 실행 중이고 사용 가능한 업데이트가 없는 경우 패치 작업에 대해 구성된 패치 기준에 따라 Patch Manager가 노드를 규정 준수 상태로 보고할 수 있습니다.

운영 체제	Details
Linux	• AlmaLinux 8.3~8.7, 9.0~9.2 • Amazon Linux 2012.03~2018.03 • Amazon Linux 2 버전 2.0 및 모든 이후 버전 • Amazon Linux 2022 • Amazon Linux 2023 • CentOS 6.5~7.9, 8.0~8.5 • CentOS Stream 8 • Debian Server 8.x, 9.x, 10.x, 11.x, 12.x • Oracle Linux 7.5~8.7, 9.0~9.2 • Raspberry Pi OS(이전 명칭: Raspbian) 9(Stretch) • Red Hat Enterprise Linux(RHEL) 6.5~8.9, 9.0~9.3 • Rocky Linux 8.4~8.7, 9.0~9.2 • SUSE Linux Enterprise Server(SLES) 12.0 이상 12.x 버전, 15.0~15.5 • Ubuntu Server 14.04 LTS, 16.04 LTS, 18.04 LTS, 20.04 LTS, 20.10 STR, 22.04 LTS, 23.04
macOS	11.3.1, 11.4~11.7(Big Sur) 12.0~12.6(Monterey) 13.0~13.5(Ventura) 14.0(Sonoma) macOS OS 업데이트 Patch Manager은(는) 12.x~13.x 또는 13.1~13.2과 같은 macOS을(를) 위한 운영 체제(OS) 업데이트 또는 업그레이드를 지원하지 않습니다. macOS에서 OS 버전 업데이트를 수행하려면

운영 체제	Details
	Apple에 내장된 OS 업그레이드 메커니즘을 사용하는 것이 좋습니다. 자세한 내용은 Apple 개발자 문서 웹사이트의 장치 관리를 참조하세요. Homebrew 지원 Homebrew 오픈 소스 소프트웨어 패키지 관리 시스템에서는 macOS 10.14.x(Mojave) 및 10.15.x(Catalina) 지원이 중단되었습니다. 따라서 이러한 버전의 패치 적용 작업은 현재 지원되지 않습니다. 리전 지원 일부 AWS 리전에서는 macOS가 지원되지 않습니다. macOS용 Amazon EC2 지원에 대한 자세한 내용은 Amazon EC2 사용 설명서의 Amazon EC2 Mac 인스턴스를 참조하세요. macOS 엣지 디바이스 AWS IoT Greengrass 코어 디바이스용 SSM Agent는 macOS에서 지원되지 않습니다. macOS 엣지 디바이스를 패치하기 위해 Patch Manager를 사용할 수 없습니다.

운영 체제	Details
Windows	R2 버전을 포함해 Windows Server 2008부터 Windows Server 2022까지 해당됩니다. **Note** AWS IoT Greengrass 코어 디바이스용 SSM Agent는 Windows 10에서 지원되지 않습니다. Windows 10 엣지 디바이스를 패치하기 위해 Patch Manager를 사용할 수 없습니다. Windows Server 2008 지원 정보 2020년 1월 14일부로 Windows Server 2008은 Microsoft의 기능 또는 보안 업데이트가 더 이상 지원되지 않습니다. Windows Server 2008 및 2008 R2용 레거시 Amazon Machine Images(AMIs)에는 여전히 SSM Agent 버전 2가 사전 설치된 상태로 포함되어 있지만 Systems Manager는 더 이상 2008 버전을 공식적으로 지원하지 않으며 이러한 Windows Server 버전용 에이전트를 더 이상 업데이트하지 않습니다. 또한 SSM Agent 버전 3은 Windows Server 2008 및 2008 R2의 일부 작업과 호환되지 않을 수 있습니다. Windows Server 2008 버전에 대해 공식적으로 지원되는 최종 SSM Agent 버전은 2.3.1644.0입니다. Windows Server 2012 및 2012 R2 지원 정보 Windows Server 2012 및 2012 R2는 2023년 10월 10일 이후로 지원이 종료되었습니다. 이러한 버전과 함께 Patch Manager를 사용하려면 Microsoft의 확장 보안 업데이트(ESU)도 사용하는 것이 좋습니다. 자세한 내용은 Microsoft 웹

운영 체제	Details
	사이트에서 Windows Server 2012 및 2012 R2에 대한 지원 종료를 참조하세요.

Patch Manager 작업 작동 방식

이 섹션에서는 AWS Systems Manager의 기능인 Patch Manager가 설치할 패치를 결정하는 방법 및 지원되는 각 운영 체제에 이러한 패치가 설치되는 방법에 대한 기술적인 정보를 제공합니다. Linux 운영 체제의 경우 관리형 노드에 구성된 기본 소스 리포지토리 이외의 패치를 위해 사용자 지정 패치 기준에서 소스 리포지토리를 지정하는 방법에 대한 정보도 제공합니다. 이 단원에서는 Linux 운영 체제의 서로 다른 배포에서 패치 기준 규칙이 작동하는 방법에 대해서도 설명합니다.

> **Note**
>
> 다음 항목의 정보는 패치 작업에 사용하는 구성 방법 또는 유형에 관계없이 적용됩니다.
>
> - Quick Setup에서 구성된 패치 정책
> - Quick Setup에서 구성된 호스트 관리 옵션
> - 패치 `Scan` 또는 `Install` 태스크를 실행하기 위한 유지 관리 기간
> - 온디맨드 Patch now(지금 패치) 작업

주제

- 패키지 릴리스 날짜 및 업데이트 날짜 계산 방법
- 보안 패치 선택 방법
- 대체 패치 소스 리포지토리를 지정하는 방법(Linux)
- 패치 설치 방법
- Linux 기반 시스템에서 패치 기준 규칙 사용 방법
- Linux 패치와 Windows 패치의 주요 차이점

패키지 릴리스 날짜 및 업데이트 날짜 계산 방법

> **Important**
>
> 이 페이지의 정보는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스용 Amazon Linux 1, Amazon Linux 2, Amazon Linux 2022, Amazon Linux 2023 운영 체제(OS)에 적용됩니다. Amazon Web Services에서 이러한 OS 유형에 대한 패키지를 생성하고 유지 관리합니다. 다른 운영 체제의 제조업체가 패키지와 리포지토리를 관리하는 방법은 릴리스 날짜와 업데이트 날짜를 계산하는 방법에 영향을 줍니다. Amazon Linux, Amazon Linux 2, Amazon Linux 2022 및 Amazon Linux 2023 이외의 OS(예: Red Hat Enterprise Linux(RHEL) 및 SUSE Linux Enterprise Server(SLES))의 경우 패키지 업데이트 및 유지 관리 방법에 대한 자세한 내용은 제조업체 설명서를 참조하세요.

생성하는 사용자 지정 패치 기준선 설정에서 대부분의 OS 유형에 대해 특정 일 수 후 설치를 위해 패치가 자동 승인되도록 지정할 수 있습니다. AWS는 7일의 자동 승인 날짜를 포함하는 미리 정의된 여러 패치 기준선을 제공합니다.

자동 승인 지연은 패치가 릴리스된 후 패치 적용을 위해 자동 승인되기까지 대기하는 일 수입니다. 예를 들어, `CriticalUpdates` 분류를 사용하여 규칙을 생성하고 7일 자동 승인 지연으로 구성합니다. 결과적으로 릴리스 날짜 또는 마지막 업데이트 날짜가 7월 7일인 새로운 중요 패치는 7월 14일에 자동으로 승인됩니다.

Amazon Linux 1, Amazon Linux 2, Amazon Linux 2022, Amazon Linux 2023에서 자동 승인 지연으로 인한 예기치 않은 결과를 방지하려면 릴리스 날짜와 업데이트 날짜가 계산되는 방식을 이해하는 것이 중요합니다.

대부분의 경우 패치가 설치되기 전 자동 승인 대기 시간은 `Release Date` 값이 아닌 `updateinfo.xml`의 `Updated Date` 값에서 계산됩니다. 다음은 이러한 날짜 계산에 대한 중요한 세부 정보입니다.

- `Release Date`는 공지가 릴리스된 날짜입니다. 이는 아직 패키지가 관련 리포지토리에서 반드시 사용 가능함을 의미하지는 않습니다.
- `Update Date`는 공지가 업데이트된 마지막 날짜입니다. 공지 업데이트는 텍스트 또는 설명 업데이트와 같은 작은 것을 나타낼 수 있습니다. 이는 아직 패키지가 해당 날짜부터 릴리스되었거나 관련 리포지토리에서 반드시 사용 가능함을 의미하지는 않습니다.

  이는 패키지의 `Update Date` 값이 7월 7일 수 있지만 7월 13일(예)까지 설치할 수 없음을 의미합니다. 이 경우 7일 자동 승인 지연을 지정하는 패치 기준선이 7월 14일에 `Install` 작업에서 실행된다

고 가정합니다. `Update Date` 값이 실행 날짜 7일 전이므로 패키지의 패치 및 업데이트는 7월 14일에 설치됩니다. 패키지가 실제 설치 가능한 상태가 되고 단 하루가 지났는데 설치가 진행됩니다.

- 운영 체제 또는 애플리케이션 패치를 포함하는 패키지는 최초 릴리스 후 두 번 이상 업데이트할 수 있습니다.
- 패키지는 AWS 관리형 리포지토리로 릴리스될 수 있지만 나중에 문제가 발견되면 롤백됩니다.

일부 패치 작업에서는 이러한 요인이 중요하지 않을 수 있습니다. 예를 들어, 심각도 값이 `Low` 및 `Medium`이고 분류가 `Recommended`인 패치를 설치하도록 패치 기준선이 구성된 경우 자동 승인 지연은 작업에 거의 영향을 미치지 않을 수 있습니다.

그러나 중요하거나 심각도가 높은 패치의 타이밍이 더 중요한 경우 패치 설치 시기를 더 세밀하게 제어할 수 있습니다. 이를 수행하는 데 권장되는 방법은 관리 노드에서 작업을 패치하기 위해 기본 리포지토리 대신 대체 패치 소스 리포지토리를 사용하는 것입니다.

사용자 지정 패치 기준을 만들 때 대체 패치 소스 리포지토리를 지정할 수 있습니다. 각 사용자 지정 패치 기준에서 지원되는 최대 20개의 Linux 운영 체제 버전에 대해 패치 소스 구성을 지정할 수 있습니다. 자세한 내용은 대체 패치 소스 리포지토리를 지정하는 방법(Linux) 단원을 참조하십시오.

보안 패치 선택 방법

AWS Systems Manager의 기능인 Patch Manager의 주 업무는 관리형 노드에 운영 체제 보안 관련 업데이트를 설치하는 것입니다. 기본적으로 Patch Manager는 사용 가능한 모든 패치를 설치하기 보다는 보안에 관련된 소규모의 패치만 설치합니다.

패치의 심각도 수준을 보고하는 Linux 기반 운영 체제 유형의 경우 Patch Manager는 업데이트 알림 또는 개별 패치에 대해 소프트웨어 게시자가 보고한 심각도 수준을 사용합니다. Patch Manager는 CVSS(Common Vulnerability Scoring System)와 같은 서드 파티 소스 또는 NVD(National Vulnerability Database)에서 발표한 지표에서 심각도 수준을 도출하지 않습니다.

> **Note**
>
> Patch Manager에서 지원하는 모든 Linux 기반 시스템에서 관리형 노드에 대해 구성된 다른 소스 리포지토리를 선택하여 비보안 업데이트를 설치할 수도 있습니다. 자세한 설명은 대체 패치 소스 리포지토리를 지정하는 방법(Linux)을 참조하세요.

이 섹션의 나머지 부분에서는 Patch Manager가 다른 지원 운영 체제에 맞는 보안 패치를 선택하는 방법을 설명합니다.

Amazon Linux 1, Amazon Linux 2, Amazon Linux 2022, and Amazon Linux 2023

사전 구성된 리포지토리는 Amazon Linux 1 및 Amazon Linux 2에서 Amazon Linux 2022 및 Amazon Linux 2023과 다르게 처리됩니다.

Amazon Linux 1 및 Amazon Linux 2의 경우 Systems Manager 패치 기준 서비스는 관리형 노드의 사전 구성된 리포지토리를 사용합니다. 하나의 노드에는 보통 다음과 같은 두 개의 사전 구성된 리포지토리가 있습니다.

Amazon Linux 1

- 리포지토리 ID: `amzn-main/latest`

  리포지토리 이름: `amzn-main-Base`
- 리포지토리 ID: `amzn-updates/latest`

  리포지토리 이름: `amzn-updates-Base`

Amazon Linux 2

- 리포지토리 ID: `amzn2-core/2/architecture`

  리포지토리 이름: `Amazon Linux 2 core repository`
- 리포지토리 ID: `amzn2extra-docker/2/architecture`

  리포지토리 이름: `Amazon Extras repo for docker`

> **Note**
>
> ##### x86_64 또는 arch64일 수 있습니다.

처음에는 Amazon Linux 2023(AL2023) 인스턴스에 AL2023 버전 및 선택한 AMI에서 사용할 수 있었던 업데이트가 포함됩니다. 기본적으로 AL2023 인스턴스에는 시작 시 심각하고 중요한 추가 보안 업데이트가 자동으로 수신되지 않습니다. 그 대신에 기본적으로 켜져 있는 AL2023의 버전 지정 리포지토리 특성을 통한 결정적 업그레이드를 통해 특정한 필요성을 충족하는 업데이트를 일정에 따라 적용할 수 있습니다. 자세한 내용은 Amazon Linux 2023 사용 설명서의 버전 지정 리포지토리를 통한 결정적 업그레이드를 참조하세요.

Amazon Linux 2022에서는 사전 구성된 리포지토리가 패키지 업데이트의 잠긴 버전에 연결됩니다. 새로운 Amazon Linux 2022용 Amazon Machine Images(AMIs)가 릴리스될 때 특정 버전으로 잠겨 있습니다. 패치 업데이트의 경우 Patch Manager는 패치 업데이트 리포지토리의 최신 잠긴 버전을 검색한 다음 잠긴 버전의 콘텐츠를 기반으로 관리 노드의 패키지를 업데이트합니다.

AL2023의 사전 구성된 리포지토리는 다음과 같습니다.

- 리포지토리 ID: `amazonlinux`

  리포지토리 이름: Amazon Linux 2023 리포지토리

Amazon Linux 2022(평가판 릴리스)의 사전 구성된 리포지토리는 패키지 업데이트의 잠긴 버전에 연결됩니다. 새로운 Amazon Linux 2022용 Amazon Machine Images(AMIs)가 릴리스될 때 특정 버전으로 잠겨 있습니다. 패치 업데이트의 경우 Patch Manager는 패치 업데이트 리포지토리의 최신 잠긴 버전을 검색한 다음 잠긴 버전의 콘텐츠를 기반으로 관리 노드의 패키지를 업데이트합니다.

Amazon Linux 2022에서 사전 구성된 리포지토리는 다음과 같습니다.

- 리포지토리 ID: `amazonlinux`

  리포지토리 이름: Amazon Linux 2022 리포지토리

> **Note**
>
> 모든 업데이트는 관리형 노드에 구성되어 있는 원격 리포지토리로부터 다운로드됩니다. 따라서, 패치 적용을 수행할 수 있도록 리포지토리에 연결하려면 인터넷에 대한 아웃바운드 액세스 권한이 노드에 있어야 합니다.

Amazon Linux 1 및 Amazon Linux 2 관리형 노드에서는 Yum을 패키지 관리자로 사용합니다. Amazon Linux 2022 및 Amazon Linux 2023에서는 DNF를 패키지 관리자로 사용합니다.

두 패키지 관리자 모두 업데이트 알림 개념을 `updateinfo.xml`이라는 파일로 사용합니다. 업데이트 알림은 단순히 특정 문제를 수정하는 패키지 모음입니다. Patch Manager는 업데이트 알림에 있는 모든 패키지를 보안으로 간주합니다. 개별 패키지에는 분류 또는 심각도 수준이 할당되지 않습니다. 따라서 Patch Manager는 업데이트 알림의 속성을 관련 패키지에 할당합니다.

> **Note**
>
> 패치 기준선 생성 페이지에서 비보안 업데이트 포함 확인란을 선택하면 `updateinfo.xml` 파일로 분류되지 않은 패키지(또는 적절한 형식의 분류, 심각도 및 날짜 값이 없는 파일이 포함된 패키지)가 미리 필터링된 패치 목록에 포함될 수 있습니다. 그러나 패치가 적용되려면 패치가 사용자 지정 패치 기준 규칙을 충족해야 합니다.

CentOS and CentOS Stream

CentOS 및 CentOS Stream의 경우 Systems Manager 패치 기준선 서비스가 관리형 노드에 있는 사전 구성된 리포지토리를 사용합니다. 다음 목록에는 가상의 CentOS 8.2 Amazon Machine Image(AMI)에 대한 예가 나와 있습니다.

- 리포지토리 ID: `example-centos-8.2-base`

  리포지토리 이름: `Example CentOS-8.2 - Base`
- 리포지토리 ID: `example-centos-8.2-extras`

  리포지토리 이름: `Example CentOS-8.2 - Extras`
- 리포지토리 ID: `example-centos-8.2-updates`

  리포지토리 이름: `Example CentOS-8.2 - Updates`
- 리포지토리 ID: `example-centos-8.x-examplerepo`

  리포지토리 이름: `Example CentOS-8.x – Example Repo Packages`

> **Note**
>
> 모든 업데이트는 관리형 노드에 구성되어 있는 원격 리포지토리로부터 다운로드됩니다. 따라서, 패치 적용을 수행할 수 있도록 리포지토리에 연결하려면 인터넷에 대한 아웃바운드 액세스 권한이 노드에 있어야 합니다.

CentOS 6 및 7 관리형 노드는 Yum을 패키지 관리자로 사용합니다. CentOS 8 및 CentOS Stream 노드는 DNF를 패키지 관리자로 사용합니다. 두 패키지 관리자 모두 업데이트 알림 개념을 사용합니다. 업데이트 알림은 단순히 특정 문제를 수정하는 패키지 모음입니다.

하지만 CentOS 및 CentOS Stream 기본 리포지토리는 업데이트 알림으로 구성되지 않습니다. 따라서 Patch Manager는 기본 CentOS 및 CentOS Stream 리포지토리의 패키지를 감지하지 못합니다. Patch Manager가 업데이트 알림에 포함되지 않은 패키지를 처리하게 하려면 패치 기준 규칙에서 `EnableNonSecurity` 플래그를 설정해야 합니다.

> **Note**
>
> CentOS 및 CentOS Stream 업데이트 알림은 지원되지 않습니다. 업데이트 알림을 포함하는 리포지토리는 시작 후 다운로드할 수 있습니다.

Debian 서버 and Raspberry Pi OS

Debian Server 및 Raspberry Pi OS(구 Raspbian)에서 Systems Manager 패치 기준 서비스는 인스턴스에 있는 사전 구성된 리포지토리를 사용합니다. 사용 가능한 패키지 업그레이드의 업데이트된 목록을 가져오기 위해 3개의 사전 구성된 리포지토리가 사용됩니다. 이를 위해, Systems Manager는 `sudo apt-get update` 명령과 동등한 명령을 수행합니다.

그런 다음 패키지는 `debian-security` *`codename`* 리포지토리에서 필터링됩니다. 즉, 각 Debian Server 버전에서 Patch Manager는 다음과 같이 해당 버전의 관련 리포지토리에 속하는 업그레이드만 식별합니다.

- 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**
>
> Debian Server 8에만 해당: 일부 Debian Server 8.* 관리형 노드는 더 이상 사용하지 않는 패키지 리포지토리(`jessie-backports`)를 참조하기 때문에 Patch Manager는 패치 작업이 성공할 수 있도록 추가 단계를 수행해야 합니다. 자세한 내용은 패치 설치 방법 단원을 참조하십시오.

Oracle Linux

Oracle Linux에서, Systems Manager 패치 기준 서비스가 관리형 노드에 있는 사전 구성된 리포지토리를 사용합니다. 하나의 노드에는 보통 다음과 같은 두 개의 사전 구성된 리포지토리가 있습니다.

Oracle Linux 7:

- 리포지토리 ID: `ol7_UEKR5/x86_64`

  리포지토리 이름: `Latest Unbreakable Enterprise Kernel Release 5 for Oracle Linux 7Server (x86_64)`
- 리포지토리 ID: `ol7_latest/x86_64`

  리포지토리 이름: `Oracle Linux 7Server Latest (x86_64)`

Oracle Linux 8:

- 리포지토리 ID: `ol8_baseos_latest`

  리포지토리 이름: `Oracle Linux 8 BaseOS Latest (x86_64)`
- 리포지토리 ID: `ol8_appstream`

  리포지토리 이름: `Oracle Linux 8 Application Stream (x86_64)`
- 리포지토리 ID: `ol8_UEKR6`

  리포지토리 이름: `Latest Unbreakable Enterprise Kernel Release 6 for Oracle Linux 8 (x86_64)`

Oracle Linux 9:

- 리포지토리 ID: `ol9_baseos_latest`

  리포지토리 이름: `Oracle Linux 9 BaseOS Latest (x86_64)`
- 리포지토리 ID: `ol9_appstream`

  리포지토리 이름: `Oracle Linux 9 Application Stream Packages(x86_64)`
- 리포지토리 ID: `ol9_UEKR7`

리포지토리 이름: `Oracle Linux UEK Release 7 (x86_64)`

> **Note**
>
> 모든 업데이트는 관리형 노드에 구성되어 있는 원격 리포지토리로부터 다운로드됩니다. 따라서, 패치 적용을 수행할 수 있도록 리포지토리에 연결하려면 인터넷에 대한 아웃바운드 액세스 권한이 노드에 있어야 합니다.

Oracle Linux 관리형 노드는 Yum을 패키지 관리자로 사용하며, Yum은 업데이트 알림의 개념을 `updateinfo.xml`이라는 파일로 사용합니다. 업데이트 알림은 단순히 특정 문제를 수정하는 패키지 모음입니다. 개별 패키지에는 분류 또는 심각도 수준이 할당되지 않습니다. 따라서 Patch Manager는 업데이트 알림의 속성을 관련 패키지에 할당하고 패치 기준에 지정된 분류 필터를 기반으로 패키지를 설치합니다.

> **Note**
>
> 패치 기준선 생성 페이지에서 비보안 업데이트 포함 확인란을 선택하면 `updateinfo.xml` 파일로 분류되지 않은 패키지(또는 적절한 형식의 분류, 심각도 및 날짜 값이 없는 파일이 포함된 패키지)가 미리 필터링된 패치 목록에 포함될 수 있습니다. 그러나 패치가 적용되려면 패치가 사용자 지정 패치 기준 규칙을 충족해야 합니다.

AlmaLinux, RHEL, and Rocky Linux

AlmaLinux, Red Hat Enterprise Linux 및 Rocky Linux의 Systems Manager 패치 기준선 서비스에서는 관리형 노드의 사전 구성된 리포지토리(repos)를 사용합니다. 하나의 노드에는 보통 다음과 같은 세 개의 사전 구성된 리포지토리가 있습니다.

모든 업데이트는 관리형 노드에 구성되어 있는 원격 리포지토리로부터 다운로드됩니다. 따라서, 패치 적용을 수행할 수 있도록 리포지토리에 연결하려면 인터넷에 대한 아웃바운드 액세스 권한이 노드에 있어야 합니다.

> **Note**
>
> 패치 기준선 생성 페이지에서 비보안 업데이트 포함 확인란을 선택하면 `updateinfo.xml` 파일로 분류되지 않은 패키지(또는 적절한 형식의 분류, 심각도 및 날짜 값이 없는 파일이

포함된 패키지)가 미리 필터링된 패치 목록에 포함될 수 있습니다. 그러나 패치가 적용되려면 패치가 사용자 지정 패치 기준 규칙을 충족해야 합니다.

Red Hat Enterprise Linux 7 관리형 노드에서는 Yum을 패키지 관리자로 사용합니다. AlmaLinux, Red Hat Enterprise Linux 8 및 Rocky Linux 관리형 노드에서는 DNF를 패키지 관리자로 사용합니다. 두 패키지 관리자 모두 업데이트 알림 개념을 `updateinfo.xml`이라는 파일로 사용합니다. 업데이트 알림은 단순히 특정 문제를 수정하는 패키지 모음입니다. 개별 패키지에는 분류 또는 심각도 수준이 할당되지 않습니다. 따라서 Patch Manager는 업데이트 알림의 속성을 관련 패키지에 할당하고 패치 기준에 지정된 분류 필터를 기반으로 패키지를 설치합니다.

RHEL 7

> **Note**
>
> 다음 리포지토리 ID는 RHUI 2와 연결되어 있습니다. RHUI 3는 2019년 12월에 출시되었으며 Yum 리포지토리 ID에 대한 다른 이름 지정 체계를 도입했습니다. 관리형 노드를 생성하는 RHEL-7 AMI에 따라 명령을 업데이트해야 할 수도 있습니다. 자세한 내용은 Red Hat Customer Portal의 Repository IDs for RHEL 7 in AWS Have Changed를 참조하세요.

- 리포지토리 ID: `rhui-REGION-client-config-server-7/x86_64`

  리포지토리 이름: `Red Hat Update Infrastructure 2.0 Client Configuration Server 7`
- 리포지토리 ID: `rhui-REGION-rhel-server-releases/7Server/x86_64`

  리포지토리 이름: `Red Hat Enterprise Linux Server 7 (RPMs)`
- 리포지토리 ID: `rhui-REGION-rhel-server-rh-common/7Server/x86_64`

  리포지토리 이름: `Red Hat Enterprise Linux Server 7 RH Common (RPMs)`

AlmaLinux, 8 RHEL 8 및 Rocky Linux 8

- 리포지토리 ID: `rhel-8-appstream-rhui-rpms`

  리포지토리 이름: `Red Hat Enterprise Linux 8 for x86_64 - AppStream from RHUI (RPMs)`
- 리포지토리 ID: `rhel-8-baseos-rhui-rpms`

리포지토리 이름: `Red Hat Enterprise Linux 8 for x86_64 - BaseOS from RHUI (RPMs)`

- 리포지토리 ID: `rhui-client-config-server-8`

  리포지토리 이름: `Red Hat Update Infrastructure 3 Client Configuration Server 8`

AlmaLinux 9, RHEL 9 및 Rocky Linux 9

- 리포지토리 ID: `rhel-9-appstream-rhui-rpms`

  리포지토리 이름: `Red Hat Enterprise Linux 9 for x86_64 - AppStream from RHUI (RPMs)`

- 리포지토리 ID: `rhel-9-baseos-rhui-rpms`

  리포지토리 이름: `Red Hat Enterprise Linux 9 for x86_64 - BaseOS from RHUI (RPMs)`

- 리포지토리 ID: `rhui-client-config-server-9`

  리포지토리 이름: `Red Hat Enterprise Linux 9 Client Configuration`

SLES

SUSE Linux Enterprise Server(SLES) 관리형 노드에 있는 ZYPP 라이브러리는 다음 위치에서 사용 가능한 패치 목록(패키지 모음)을 가져옵니다.

- 리포지토리 목록: `etc/zypp/repos.d/*`
- 패키지 정보: `/var/cache/zypp/raw/*`

SLES 관리형 노드는 Zypper를 패키지 관리자로 사용하며, Zypper는 패치 개념을 사용합니다. 패치는 단순히 특정 문제를 수정하는 패키지 모음입니다. Patch Manager는 패치에서 참조하는 모든 패키지를 보안 관련으로 처리합니다. 개별 패키지에는 분류 또는 심각도가 지정되지 않았기 때문에 Patch Manager는 해당 패키지를 이들이 속한 패치의 속성에 할당합니다.

Ubuntu 서버

Ubuntu Server에서, Systems Manager 패치 기준 서비스가 관리형 노드에 있는 사전 구성된 리포지토리를 사용합니다. 사용 가능한 패키지 업그레이드의 업데이트된 목록을 가져오기 위해 3개의

사전 구성된 리포지토리가 사용됩니다. 이를 위해, Systems Manager는 `sudo apt-get update` 명령과 동등한 명령을 수행합니다.

그런 다음 *codename*`-security` 리포지토리에서 패키지가 필터링됩니다. 여기서 코드명은 Ubuntu Server 14의 경우 `trusty`와 같이 릴리스 버전마다 고유합니다. Patch Manager는 이러한 리포지토리의 일부인 업그레이드만 식별합니다.

- 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`)

Windows Server

Microsoft Windows 운영 체제에서 Patch Manager는 Microsoft가 Microsoft Update에 게시하고 WSUS(Windows Server Update Services)에서 자동으로 사용할 수 있는 사용 가능한 업데이트 목록을 검색합니다.

Patch Manager는 모든 AWS 리전에서 새로운 업데이트를 지속적으로 모니터링합니다. 사용 가능한 업데이트 목록은 최소한 하루에 한 번씩 리전별로 갱신됩니다. Microsoft의 패치 정보가 처리되면 Patch Manager는 해당 패치 목록에서 최신 업데이트로 대체된 업데이트를 제거합니다. 그러므로 최신 업데이트만 표시되며 설치 시 사용 가능한 상태가 됩니다. 예를 들어 `KB4012214`가 `KB3135456`을 대체하는 경우 `KB4012214`만 Patch Manager에서 업데이트로 사용됩니다.

Patch Manager는 Patch Manager에 지원되는 Windows Server 운영 체제 버전에 대해서만 패치를 제공합니다. 예를 들어 Patch Manager는 Windows RT 패치에 사용할 수 없습니다.

> **Note**
>
> 경우에 따라 Microsoft는 업데이트된 날짜와 시간을 지정하지 않는 애플리케이션에 대한 패치를 릴리스합니다. 이 경우 업데이트된 `01/01/1970`의 날짜 및 시간은 기본적으로 제공됩니다.

대체 패치 소스 리포지토리를 지정하는 방법(Linux)

패치 작업에 대해 관리형 노드에 구성된 기본 리포지토리를 사용하는 경우 AWS Systems Manager의 기능인 Patch Manager는 보안 관련 패치를 찾거나 설치합니다. 이는 Patch Manager의 기본 동작입니다. Patch Manager가 보안 패치를 선택 및 설치하는 방법에 대한 모든 정보는 보안 패치 선택 방법 섹션을 참조하세요.

그러나 Linux 시스템에서는 Patch Manager를 사용하여 보안과 관련되지 않은 패치 또는 관리형 노드에 구성된 기본 소스 리포지토리가 아닌 다른 소스 리포지토리에 있는 패치를 설치할 수도 있습니다. 사용자 지정 패치 기준을 만들 때 대체 패치 소스 리포지토리를 지정할 수 있습니다. 각 사용자 지정 패치 기준에서 지원되는 최대 20개의 Linux 운영 체제 버전에 대해 패치 소스 구성을 지정할 수 있습니다.

예를 들어, Ubuntu Server 플릿에 Ubuntu Server 14.04 및 Ubuntu Server 16.04 관리형 노두가 모두 포함된다고 가정해 보겠습니다. 이 경우, 동일한 사용자 지정 패치 기준으로 각 버전에 대체 리포지토리를 지정할 수 있습니다. 각 버전에 대해 이름을 입력하고, 운영 체제 버전 유형(제품)을 지정하고, 리포지토리 구성을 제공합니다. 또한 지원되는 모든 운영 체제 버전에 적용되는 단일의 대체 소스 리포지토리를 지정할 수도 있습니다.

> **Note**
>
> 관리형 노드에 대한 대체 패치 리포지토리를 지정하는 사용자 정의 패치 기준을 실행해도 해당 리포지토리가 운영체제의 새 기본 리포지토리로 지정되지 않습니다. 패치 작업이 완료된 후 이전에 노드의 운영체제에서 기본값으로 구성된 리포지토리는 기본값으로 유지됩니다.

이 옵션 사용에 대한 예제 시나리오 목록은 이 주제 후반부의 대체 패치 소스 리포지토리의 사용 샘플 섹션을 참조하세요.

기본 및 사용자 지정 패치 기준에 대한 자세한 내용은 미리 정의된 패치 및 사용자 지정 패치 기준 정보 섹션을 참조하세요.

예: 콘솔 사용

Systems Manager 콘솔에서 작업 중인 경우 대체 패치 소스 리포지토리를 지정하려면 패치 기준 생성(Create patch baseline) 페이지의 패치 소스(Patch sources) 섹션을 사용합니다. 패치 소스(Patch sources) 옵션 사용에 대한 자세한 내용은 사용자 정의 패치 기준 생성(Linux) 섹션을 참조하세요.

예: AWS CLI 사용

AWS Command Line Interface(AWS CLI)와 함께 `--sources` 옵션을 사용하는 예는 다른 OS 버전에 대한 사용자 지정 리포지토리가 있는 패치 기준 생성 섹션을 참조하세요.

주제

대체 리포지토리에 대한 중요 고려 사항

대체 패치 리포지토리를 사용하여 대치 전략을 계획할 경우 다음 사항에 유의하십시오.

지정된 리포지토리만 패치를 적용하는 데 사용됩니다.

대체 리포지토리를 지정하는 것이 추가 리포지토리를 지정하는 것을 의미하지는 않습니다. 관리형 노드에 대해 기본으로 구성되지 않은 리포지토리를 지정할 수 있습니다. 하지만 업데이트를 적용하려면 기본 리포지토리도 대체 패치 소스 구성의 일부로 지정해야 합니다.

예를 들어 Amazon Linux 2 관리형 노드의 기본 리포지토리는 `amzn2-core` 및 `amzn2extra-docker`입니다. 패치 적용 작업에 EPEL(Extra Packages for Enterprise Linux) 리포지토리를 포함하려면 세 리포지토리를 모두 대체 리포지토리로 지정해야 합니다.

> **Note**
>
> 관리형 노드에 대한 대체 패치 리포지토리를 지정하는 사용자 정의 패치 기준을 실행해도 해당 리포지토리가 운영체제의 새 기본 리포지토리로 지정되지 않습니다. 패치 작업이 완료된 후 이전에 노드의 운영체제에서 기본값으로 구성된 리포지토리는 기본값으로 유지됩니다.

YUM 기반 배포에 대한 패치 동작은 updateinfo.xml 매니페스트에 따라 다릅니다.

YUM 기반 배포(예: Amazon Linux 1 또는 Amazon Linux 2, Red Hat Enterprise Linux 또는 CentOS)에 대해 대체 패치 리포지토리를 지정할 경우 패치 동작은 완전하고 올바른 형식의 `updateinfo.xml` 파일 형태의 업데이트 매니페스트가 리포지토리에 포함되는지 여부에 따라 달라집니다. 이 파일은 다양한 패키지의 릴리스 날짜, 분류 및 심각도를 지정합니다. 패치 동작에 영향을 주는 항목은 다음과 같습니다.

- 분류(Classification)와 심각도(Severity)를 기준으로 필터링하지만 `updateinfo.xml`에 지정되어 있지 않은 경우 해당 패키지는 필터에 포함되지 않습니다. 즉, `updateinfo.xml` 파일이 없는 패키지는 패치에 포함되지 않습니다.
- ApprovalAfterDays를 기준으로 필터링하지만 패키지 릴리스 날짜가 Unix Epoch 형식이 아니거나 릴리스 날짜가 지정되지 않은 경우 해당 패키지는 필터에 포함되지 않습니다.

- 패치 기준선 생성 페이지에서 비보안 업데이트 포함 확인란을 선택한 경우는 예외입니다. 이 경우 `updateinfo.xml` 파일이 없거나 적절하게 서식 지정된 분류(Classification), 보안(Severity) 및 날짜(Date) 값이 없는 이 파일을 포함하는 패키지는 사전 필터링된 패치 목록에 포함됩니다. 이 경우에도 설치를 위한 다른 패치 기준 규칙 요건을 충족해야 합니다.

대체 패치 소스 리포지토리의 사용 샘플

예제 1 - Ubuntu Server에 대한 비보안 업데이트

이미 Patch Manager를 사용하여 AWS에서 제공하는 미리 정의된 패치 기준인 `AWS-UbuntuDefaultPatchBaseline`을 사용하는 Ubuntu Server 관리형 노드의 플릿에 보안 패치를 설치했습니다. 이 기본 패치 기준을 기반으로 하는 새 패치 기준을 생성하되, 승인 규칙에서 기본 배포의 일부인 비보안 관련 업데이트도 설치하도록 지정할 수 있습니다. 이 패치 기준이 노드에 대해 실행될 때 보안 및 비보안 문제에 대한 패치가 적용됩니다. 또한 기준에 대해 지정하는 패치 예외의 비보안 패치를 승인할 수도 있습니다.

예제 2 - Ubuntu Server의 PPA(Personal Package Archives)

Ubuntu Server 관리형 노드가 Ubuntu의 PPA(Personal Package Archives)를 통해 배포된 소프트웨어를 실행합니다. 이 경우 관리형 노드에 구성한 PPA 리포지토리를 패치 적용 작업에 대한 소스 리포지토리로 지정하는 패치 기준을 생성합니다. 그런 다음 Run Command를 사용하여 노드에서 패치 기준 문서를 실행합니다.

예제 3 - Amazon Linux의 사내 애플리케이션

Amazon Linux 관리형 노드에서 산업 규정 준수에 필요한 애플리케이션을 실행해야 합니다. 노드에 이러한 애플리케이션의 리포지토리를 구성하거나, YUM을 사용하여 애플리케이션을 처음 설치한 후 업데이트하거나, 이 새로운 기업 리포지토리를 포함하도록 새 패치 기준을 생성할 수 있습니다. 그런 다음 Run Command를 사용하여 `Scan` 옵션으로 `AWS-RunPatchBaseline` 문서를 실행하여 기업 패키지가 설치된 패키지로 나열되고 관리형 노드에서 최신 상태인지 확인할 수 있습니다. 최신이 아닌 경우 애플리케이션을 업데이트하도록 `Install` 옵션을 사용하여 문서를 다시 실행하면 됩니다.

패치 설치 방법

AWS Systems Manager의 기능인 Patch Manager는 운영 체제 유형에 적합한 내장 메커니즘을 사용하여 관리형 노드에 업데이트를 설치합니다. 예를 들어 Windows Server의 경우 Windows Update API가 사용되며 Amazon Linux 2의 경우에는 `yum` 패키지 관리자가 사용됩니다.

이 섹션의 나머지 부분에서는 Patch Manager가 운영 체제에 패치를 설치하는 방법을 설명합니다.

Amazon Linux 1, Amazon Linux 2, Amazon Linux 2022, and Amazon Linux 2023

Amazon Linux 1, Amazon Linux 2, Amazon Linux 2022, Amazon Linux 2023 관리형 노드의 패치 설치 워크플로는 다음과 같습니다.

1. `AWS-RunPatchBaseline` 또는 `AWS-RunPatchBaselineAssociation` 문서에 대해 `InstallOverrideList` 파라미터를 사용하여 https URL 또는 Amazon Simple Storage Service(Amazon S3) 경로 스타일 URL을 사용하여 패치 목록을 지정한 경우 나열된 패치가 설치되고 2~7단계를 건너뜁니다.
2. 패치 기준에 지정된 대로 GlobalFilters를 적용하여 자격을 갖춘 패키지만 향후 프로세스에 사용되도록 유지됩니다.
3. 패치 기준에 지정된 대로 ApprovalRules를 적용합니다. 각 승인 규칙은 패키지를 승인된 것으로 정의할 수 있습니다.

   그러나 승인 규칙은 패치 기준을 생성하거나 마지막으로 업데이트할 때 비보안 업데이트 포함(Include nonsecurity updates) 확인란을 선택했는지 여부에 따라 달라집니다.

   비보안 업데이트가 제외되면 보안 리포지토리에서 업그레이드가 있는 패키지만 선택할 수 있도록 묵시적인 규칙이 적용됩니다. 패키지마다, 후보 패키지 버전(일반적으로 최신 버전)이 보안 리포지토리에 속해 있어야 합니다.

   비보안 업데이트가 포함된 경우 다른 리포지토리의 패치도 고려됩니다.
4. 패치 기준에 지정된 대로 ApprovedPatches를 적용합니다. 승인된 패치는 GlobalFilters를 통해 폐기되었거나 ApprovalRules에 지정된 승인 규칙이 이를 승인하지 않더라도 업데이트가 승인됩니다.
5. 패치 기준에 지정된 대로 RejectedPatches를 적용합니다. 거부된 패치는 승인된 패치 목록에서 제거되고 적용되지 않습니다.
6. 여러 패치 버전이 승인되는 경우 최신 버전이 적용됩니다.
7. YUM 업데이트 API(Amazon Linux 1, Amazon Linux 2) 또는 DNF 업데이트 API(Amazon Linux 2022, Amazon Linux 2023)는 다음과 같이 승인된 패치에 적용됩니다.
   - AWS에서 제공하는 미리 정의된 기본 패치 기준선의 경우 `updateinfo.xml`에 지정된 패치만 적용됩니다(보안 업데이트만 해당). 비보안 업데이트 포함 확인란이 선택되지 않았기 때문입니다. 미리 정의된 기준선은 다음과 같은 사용자 지정 기준선과 동일합니다.
     - 비보안 업데이트 포함 확인란은 선택되지 않음
     - `[Critical, Important]`의 심각도 목록
     - `[Security, Bugfix]`의 분류 목록

Amazon Linux 1 및 Amazon Linux 2의 경우 이 워크플로와 동등한 yum 명령은 다음과 같습니다.

```
sudo yum update-minimal --sec-severity=critical,important --bugfix -y
```

Amazon Linux 2022 및 Amazon Linux 2023의 경우 이 워크플로와 동등한 dnf 명령은 다음과 같습니다.

```
sudo dnf upgrade-minimal --sec-severity=critical --sec-severity=important --
bugfix -y
```

비보안 업데이트 포함 확인란이 선택된 경우 `updateinfo.xml`에 포함된 패치 및 `updateinfo.xml`에 포함되지 않은 패치가 모두 적용됩니다(보안 및 비보안 업데이트).

Amazon Linux 1 및 Amazon Linux 2의 경우 비보안 업데이트 포함이 선택된 기준선에 `[Critical, Important]`의 심각도 목록 및 `[Security, Bugfix]`의 분류 목록이 포함된 경우 이와 동등한 yum 명령은 다음과 같습니다.

```
sudo yum update --security --sec-severity=critical,important --bugfix -y
```

Amazon Linux 2022 및 Amazon Linux 2023의 경우 이와 동등한 dnf 명령은 다음과 같습니다.

```
sudo dnf upgrade --security --sec-severity=critical --sec-severity=important --
bugfix -y
```

> **Note**
>
> Amazon Linux 2022 및 Amazon Linux 2023의 경우 `Medium`의 패치 심각도 수준은 일부 외부 리포지토리에 정의될 수 있는 `Moderate`의 심각도와 동일합니다. 패치 기준에 `Medium` 심각도 패치를 포함하면 외부 패치의 `Moderate` 심각도 패치도 인스턴스에 설치됩니다.
>
> API 작업 DescribeInstancePatches를 사용하여 규정 준수 데이터를 쿼리하는 경우 심각도 수준 `Medium`을 필터링하면 심각도 수준이 `Medium` 및 `Moderate`인 패치가 모두 보고됩니다.

Amazon Linux 2022 및 Amazon Linux 2023은 DNF 패키지 관리자가 인식하는 패치 심각도 수준 None도 지원합니다.

8. 업데이트가 설치되어 있는 경우 관리형 노드가 재부팅됩니다. (예외: AWS-RunPatchBaseline 문서에서 RebootOption 파라미터가 NoReboot로 설정되어 있으면 Patch Manager를 실행한 후 관리형 노드가 재부팅되지 않습니다. 자세한 내용은 파라미터 이름: RebootOption 섹션을 참조하세요.)

CentOS and CentOS Stream

CentOS 및 CentOS Stream 관리형 노드에서 패치 설치 워크플로는 다음과 같습니다.

1. AWS-RunPatchBaseline 또는 AWS-RunPatchBaselineAssociation 문서에 대해 InstallOverrideList 파라미터를 사용하여 https URL 또는 Amazon Simple Storage Service(Amazon S3) 경로 스타일 URL을 사용하여 패치 목록을 지정한 경우 나열된 패치가 설치되고 2~7단계를 건너뜁니다.

   패치 기준에 지정된 대로 GlobalFilters를 적용하여 자격을 갖춘 패키지만 향후 프로세스에 사용되도록 유지됩니다.

2. 패치 기준에 지정된 대로 ApprovalRules를 적용합니다. 각 승인 규칙은 패키지를 승인된 것으로 정의할 수 있습니다.

   그러나 승인 규칙은 패치 기준을 생성하거나 마지막으로 업데이트할 때 비보안 업데이트 포함(Include nonsecurity updates) 확인란을 선택했는지 여부에 따라 달라집니다.

   비보안 업데이트가 제외되면 보안 리포지토리에서 업그레이드가 있는 패키지만 선택할 수 있도록 묵시적인 규칙이 적용됩니다. 패키지마다, 후보 패키지 버전(일반적으로 최신 버전)이 보안 리포지토리에 속해 있어야 합니다.

   비보안 업데이트가 포함된 경우 다른 리포지토리의 패치도 고려됩니다.

3. 패치 기준에 지정된 대로 ApprovedPatches를 적용합니다. 승인된 패치는 GlobalFilters를 통해 폐기되었거나 ApprovalRules에 지정된 승인 규칙이 이를 승인하지 않더라도 업데이트가 승인됩니다.

4. 패치 기준에 지정된 대로 RejectedPatches를 적용합니다. 거부된 패치는 승인된 패치 목록에서 제거되고 적용되지 않습니다.

5. 여러 패치 버전이 승인되는 경우 최신 버전이 적용됩니다.

6. YUM 업데이트 API(CentOS 6.x 및 7.x 버전) 또는 DNF 업데이트(CentOS 8 및 CentOS Stream)는 승인된 패치에 적용됩니다.
7. 업데이트가 설치되어 있는 경우 관리형 노드가 재부팅됩니다. (예외: `AWS-RunPatchBaseline` 문서에서 `RebootOption` 파라미터가 `NoReboot`로 설정되어 있으면 Patch Manager를 실행한 후 관리형 노드가 재부팅되지 않습니다. 자세한 내용은 파라미터 이름: `RebootOption` 섹션을 참조하세요.)

Debian 서버 and Raspberry Pi OS

Debian Server 및 Raspberry Pi OS(구 Raspbian) 인스턴스에서 패치 설치 워크플로는 다음과 같습니다.

1. `AWS-RunPatchBaseline` 또는 `AWS-RunPatchBaselineAssociation` 문서에 대해 `InstallOverrideList` 파라미터를 사용하여 https URL 또는 Amazon Simple Storage Service(Amazon S3) 경로 스타일 URL을 사용하여 패치 목록을 지정한 경우 나열된 패치가 설치되고 2~7단계를 건너뜁니다.
2. `python3-apt`(`libapt`에 대한 Python 라이브러리 인터페이스)에 대한 업데이트가 있는 경우 최신 버전으로 업그레이드됩니다. ([비보안 업데이트 포함(Include nonsecurity updates)] 옵션을 선택하지 않은 경우에도 이 비보안 패키지가 업그레이드됩니다.)

> **Important**
>
> Debian Server 8에만 해당: 일부 Debian Server 8.* 관리형 노드는 더 이상 사용하지 않는 패키지 리포지토리(`jessie-backports`)를 참조하기 때문에 Patch Manager는 패치 작업이 성공할 수 있도록 다음과 같은 추가 단계를 수행해야 합니다.
>
> a. 관리형 노드에서 `jessie-backports` 리포지토리에 대한 참조는 소스 위치 목록(`/etc/apt/sources.list.d/jessie-backports`)에서 코멘트 아웃 처리됩니다. 따라서 해당 위치에서 패치를 다운로드하려고 시도하지 않습니다.
>
> b. Stretch 보안 업데이트 서명 키를 가져옵니다. 이 키는 Debian Server 8.* 배포의 업데이트 및 설치 작업에 필요한 권한을 제공합니다.
>
> c. 이 시점에서 패치 프로세스가 시작되기 전에 `python3-apt`의 최신 버전이 설치되도록 `apt-get` 작업이 실행됩니다.
>
> d. 설치 프로세스가 완료되면 `jessie-backports` 리포지토리에 대한 참조가 복원되고 서명 키가 apt 소스 키 링에서 제거됩니다. 이 작업은 패치 적용 작업 이전의 시스템 구성을 그대로 두기 위해 수행됩니다.

다음에 Patch Manager가 시스템을 업데이트하면 동일한 프로세스가 반복됩니다.

3. 패치 기준에 지정된 대로 GlobalFilters를 적용하여 자격을 갖춘 패키지만 향후 프로세스에 사용되도록 유지됩니다.
4. 패치 기준에 지정된 대로 ApprovalRules를 적용합니다. 각 승인 규칙은 패키지를 승인된 것으로 정의할 수 있습니다.

> **Note**
>
> Debian Server에 대한 업데이트 패키지의 릴리스 날짜를 확실히 결정할 수 없으므로 이 운영 체제에서는 자동 승인 옵션이 지원되지 않습니다.

그러나 승인 규칙은 패치 기준을 생성하거나 마지막으로 업데이트할 때 비보안 업데이트 포함(Include nonsecurity updates) 확인란을 선택했는지 여부에 따라 달라집니다.

비보안 업데이트가 제외되면 보안 리포지토리에서 업그레이드가 있는 패키지만 선택할 수 있도록 묵시적인 규칙이 적용됩니다. 패키지마다, 후보 패키지 버전(일반적으로 최신 버전)이 보안 리포지토리에 속해 있어야 합니다.

비보안 업데이트가 포함된 경우 다른 리포지토리의 패치도 고려됩니다.

> **Note**
>
> Debian Server 및 Raspberry Pi OS의 경우 패치 후보 버전은 `debian-security`에 포함된 패치로 제한됩니다.

5. 패치 기준에 지정된 대로 ApprovedPatches를 적용합니다. 승인된 패치는 GlobalFilters를 통해 폐기되었거나 ApprovalRules에 지정된 승인 규칙이 이를 승인하지 않더라도 업데이트가 승인됩니다.
6. 패치 기준에 지정된 대로 RejectedPatches를 적용합니다. 거부된 패치는 승인된 패치 목록에서 제거되고 적용되지 않습니다.
7. APT 라이브러리가 사용되어 패키지가 업그레이드됩니다.

> **Note**
>
> Patch Manager에서는 APT `Pin-Priority` 옵션을 사용하여 패키지에 우선순위를 지정하는 기능을 지원하지 않습니다. Patch Manager에서는 활성화된 모든 리포지토리에

서 사용 가능한 업데이트를 집계하고 설치된 각 패키지의 기준과 일치하는 최신 업데이트를 선택합니다.

8. 업데이트가 설치되어 있는 경우 관리형 노드가 재부팅됩니다. (예외: `AWS-RunPatchBaseline` 문서에서 `RebootOption` 파라미터가 `NoReboot`로 설정되어 있으면 Patch Manager를 실행한 후 관리형 노드가 재부팅되지 않습니다. 자세한 내용은 파라미터 이름: `RebootOption` 섹션을 참조하세요.)

macOS

macOS 관리형 노드에서 패치 설치 워크플로는 다음과 같습니다.

1. `/Library/Receipts/InstallHistory.plist` 속성 목록은 `softwareupdate` 및 `installer` 패키지 관리자를 사용하여 설치 및 업그레이드된 소프트웨어의 레코드입니다. `pkgutil` 명령줄 도구(`installer`용)와 `softwareupdate` 패키지 관리자를 사용하여 CLI 명령을 실행하여 이 목록을 구문 분석합니다.

   `installer`의 경우 CLI 명령에 대한 응답에 `package name`, `version`, `volume`, `location` 및 `install-time` 세부 정보가 포함되지만 Patch Manager는 `package name`과 `version`만 사용합니다.

   `softwareupdate`의 경우 CLI 명령에 대한 응답에 패키지 이름(`display name`), `version` 및 `date`가 포함되지만 패치 관리자는 패키지 이름과 버전만 사용합니다.

   Brew 및 Brew Cask의 경우 Homebrew는 루트 사용자로 실행되는 명령을 지원하지 않습니다. 결과적으로 Patch Manager는 Homebrew 디렉터리의 소유자 또는 Homebrew 디렉터리의 소유자 그룹에 속하는 유효한 사용자로 Homebrew 명령을 쿼리하고 실행합니다. 명령은 `softwareupdate` 및 `installer`와 유사하며 Python 하위 프로세스를 통해 실행되어 패키지 데이터를 수집하고 출력을 구문 분석하여 패키지 이름 및 버전을 식별합니다.
2. 패치 기준에 지정된 대로 GlobalFilters를 적용하여 자격을 갖춘 패키지만 향후 프로세스에 사용되도록 유지됩니다.
3. 패치 기준에 지정된 대로 ApprovalRules를 적용합니다. 각 승인 규칙은 패키지를 승인된 것으로 정의할 수 있습니다.
4. 패치 기준에 지정된 대로 ApprovedPatches를 적용합니다. 승인된 패치는 GlobalFilters를 통해 폐기되었거나 ApprovalRules에 지정된 승인 규칙이 이를 승인하지 않더라도 업데이트가 승인됩니다.

5. 패치 기준에 지정된 대로 RejectedPatches를 적용합니다. 거부된 패치는 승인된 패치 목록에서 제거되고 적용되지 않습니다.
6. 여러 패치 버전이 승인되는 경우 최신 버전이 적용됩니다.
7. 관리형 노드에서 적절한 패키지 CLI를 호출하여 다음과 같이 승인된 패치를 처리합니다.

> **Note**
>
> `installer`에는 업데이트를 확인하고 설치하는 기능이 없습니다. 따라서 `installer`의 경우 Patch Manager는 설치된 패키지만 보고합니다. 그 결과 `installer` 패키지는 `Missing`으로 보고되지 않습니다.

   - AWS에서 제공하는 미리 정의된 기본 패치 기준선 및 비보안 업데이트 포함 확인란이 선택되지 않은 사용자 지정 패치 기준선의 경우 보안 업데이트만 적용됩니다.
   - 비보안 업데이트 포함 확인란이 선택된 사용자 지정 패치 기준선의 경우 보안 및 비보안 업데이트가 모두 적용됩니다.
8. 업데이트가 설치되어 있는 경우 관리형 노드가 재부팅됩니다. (예외: `AWS-RunPatchBaseline` 문서에서 `RebootOption` 파라미터가 `NoReboot`로 설정되어 있으면 Patch Manager를 실행한 후 관리형 노드가 재부팅되지 않습니다. 자세한 내용은 파라미터 이름: `RebootOption` 섹션을 참조하세요.)

Oracle Linux

Oracle Linux 관리형 노드에서 패치 설치 워크플로는 다음과 같습니다.

1. `AWS-RunPatchBaseline` 또는 `AWS-RunPatchBaselineAssociation` 문서에 대해 `InstallOverrideList` 파라미터를 사용하여 https URL 또는 Amazon Simple Storage Service(Amazon S3) 경로 스타일 URL을 사용하여 패치 목록을 지정한 경우 나열된 패치가 설치되고 2~7단계를 건너뜁니다.
2. 패치 기준에 지정된 대로 GlobalFilters를 적용하여 자격을 갖춘 패키지만 향후 프로세스에 사용되도록 유지됩니다.
3. 패치 기준에 지정된 대로 ApprovalRules를 적용합니다. 각 승인 규칙은 패키지를 승인된 것으로 정의할 수 있습니다.

   그러나 승인 규칙은 패치 기준을 생성하거나 마지막으로 업데이트할 때 비보안 업데이트 포함(Include nonsecurity updates) 확인란을 선택했는지 여부에 따라 달라집니다.

비보안 업데이트가 제외되면 보안 리포지토리에서 업그레이드가 있는 패키지만 선택할 수 있도록 묵시적인 규칙이 적용됩니다. 패키지마다, 후보 패키지 버전(일반적으로 최신 버전)이 보안 리포지토리에 속해 있어야 합니다.

비보안 업데이트가 포함된 경우 다른 리포지토리의 패치도 고려됩니다.

4. 패치 기준에 지정된 대로 ApprovedPatches를 적용합니다. 승인된 패치는 GlobalFilters를 통해 폐기되었거나 ApprovalRules에 지정된 승인 규칙이 이를 승인하지 않더라도 업데이트가 승인됩니다.
5. 패치 기준에 지정된 대로 RejectedPatches를 적용합니다. 거부된 패치는 승인된 패치 목록에서 제거되고 적용되지 않습니다.
6. 여러 패치 버전이 승인되는 경우 최신 버전이 적용됩니다.
7. 버전 7 관리형 노드에서 다음과 같이 YUM 업데이트 API가 승인된 패치에 적용됩니다.
   - AWS에서 제공하는 미리 정의된 기본 패치 기준선 및 비보안 업데이트 포함 확인란이 선택되지 않은 사용자 지정 패치 기준선의 경우 `updateinfo.xml`에 지정한 패치만 적용됩니다(보안 업데이트만 해당).

     이 워크플로와 동등한 YUM 명령은 다음과 같습니다.

     ```
     sudo yum update-minimal --sec-severity=Important,Moderate --bugfix -y
     ```

   - 비보안 업데이트 포함 확인란을 선택한 사용자 지정 패치 기준선의 경우 `updateinfo.xml`에 포함된 패치 및 `updateinfo.xml`에 포함되지 않은 패치가 모두 적용됩니다(보안 및 비보안 업데이트).

     이 워크플로와 동등한 YUM 명령은 다음과 같습니다.

     ```
     sudo yum update --security --bugfix -y
     ```

     버전 8 및 9 관리형 노드에서 다음과 같이 DNF 업데이트 API가 승인된 패치에 적용됩니다.

   - AWS에서 제공하는 미리 정의된 기본 패치 기준선 및 비보안 업데이트 포함 확인란이 선택되지 않은 사용자 지정 패치 기준선의 경우 `updateinfo.xml`에 지정한 패치만 적용됩니다(보안 업데이트만 해당).

     이 워크플로와 동등한 YUM 명령은 다음과 같습니다.

```
sudo dnf upgrade-minimal --security --sec-severity=Moderate --sec-
severity=Important
```

- 비보안 업데이트 포함 확인란을 선택한 사용자 지정 패치 기준선의 경우 `updateinfo.xml`에 포함된 패치 및 `updateinfo.xml`에 포함되지 않은 패치가 모두 적용됩니다(보안 및 비보안 업데이트).

  이 워크플로와 동등한 YUM 명령은 다음과 같습니다.

```
sudo dnf upgrade --security --bugfix
```

8. 업데이트가 설치되어 있는 경우 관리형 노드가 재부팅됩니다. (예외: `AWS-RunPatchBaseline` 문서에서 `RebootOption` 파라미터가 `NoReboot`로 설정되어 있으면 Patch Manager를 실행한 후 관리형 노드가 재부팅되지 않습니다. 자세한 내용은 파라미터 이름: `RebootOption` 섹션을 참조하세요.)

AlmaLinux, RHEL, and Rocky Linux

Red Hat Enterprise Linux 및 Rocky Linux 관리형 노드의 패치 설치 워크플로는 다음과 같습니다.

1. `AWS-RunPatchBaseline` 또는 `AWS-RunPatchBaselineAssociation` 문서에 대해 `InstallOverrideList` 파라미터를 사용하여 https URL 또는 Amazon Simple Storage Service(Amazon S3) 경로 스타일 URL을 사용하여 패치 목록을 지정한 경우 나열된 패치가 설치되고 2~7단계를 건너뜁니다.
2. 패치 기준에 지정된 대로 GlobalFilters를 적용하여 자격을 갖춘 패키지만 향후 프로세스에 사용되도록 유지됩니다.
3. 패치 기준에 지정된 대로 ApprovalRules를 적용합니다. 각 승인 규칙은 패키지를 승인된 것으로 정의할 수 있습니다.

   그러나 승인 규칙은 패치 기준을 생성하거나 마지막으로 업데이트할 때 비보안 업데이트 포함(Include nonsecurity updates) 확인란을 선택했는지 여부에 따라 달라집니다.

   비보안 업데이트가 제외되면 보안 리포지토리에서 업그레이드가 있는 패키지만 선택할 수 있도록 묵시적인 규칙이 적용됩니다. 패키지마다, 후보 패키지 버전(일반적으로 최신 버전)이 보안 리포지토리에 속해 있어야 합니다.

   비보안 업데이트가 포함된 경우 다른 리포지토리의 패치도 고려됩니다.

4. 패치 기준에 지정된 대로 ApprovedPatches를 적용합니다. 승인된 패치는 GlobalFilters를 통해 폐기되었거나 ApprovalRules에 지정된 승인 규칙이 이를 승인하지 않더라도 업데이트가 승인됩니다.
5. 패치 기준에 지정된 대로 RejectedPatches를 적용합니다. 거부된 패치는 승인된 패치 목록에서 제거되고 적용되지 않습니다.
6. 여러 패치 버전이 승인되는 경우 최신 버전이 적용됩니다.
7. YUM 업데이트 API(RHEL 7의 경우) 또는 DNF 업데이트 API(AlmaLinux 8 및 9, RHEL 8 및 9, Rocky Linux 8 및 9의 경우)는 다음과 같이 승인된 패치에 적용됩니다.
   - AWS에서 제공하는 미리 정의된 기본 패치 기준선 및 비보안 업데이트 포함 확인란이 선택되지 않은 사용자 지정 패치 기준선의 경우 `updateinfo.xml`에 지정한 패치만 적용됩니다(보안 업데이트만 해당).

     RHEL 7의 경우 이 워크플로와 동등한 yum 명령은 다음과 같습니다.

     ```
     sudo yum update-minimal --sec-severity=Critical,Important --bugfix -y
     ```

     AlmaLinux, RHEL 8 및 Rocky Linux의 경우 이 워크플로와 동등한 dnf 명령은 다음과 같습니다.

     ```
     sudo dnf update-minimal --sec-severity=Critical --bugfix -y ; \
     sudo dnf update-minimal --sec-severity=Important --bugfix -y
     ```
   - 비보안 업데이트 포함 확인란을 선택한 사용자 지정 패치 기준선의 경우 `updateinfo.xml`에 포함된 패치 및 `updateinfo.xml`에 포함되지 않은 패치가 모두 적용됩니다(보안 및 비보안 업데이트).

     RHEL 7의 경우 이 워크플로와 동등한 yum 명령은 다음과 같습니다.

     ```
     sudo yum update --security --bugfix -y
     ```

     AlmaLinux 8 및 9, RHEL 8 및 9 Rocky Linux 8 및 9의 경우 이 워크플로와 동등한 dnf 명령은 다음과 같습니다.

     ```
     sudo dnf update --security --bugfix -y
     ```
8. 업데이트가 설치되어 있는 경우 관리형 노드가 재부팅됩니다. (예외: `AWS-RunPatchBaseline` 문서에서 `RebootOption` 파라미터가 `NoReboot`로 설정되어 있으면 Patch Manager를 실행한

후 관리형 노드가 재부팅되지 않습니다. 자세한 내용은 파라미터 이름: RebootOption 섹션을 참조하세요.)

SLES

:SUSE Linux Enterprise Server(SLES) 관리형 노드, 패치 설치 워크플로는 다음과 같습니다.

1. AWS-RunPatchBaseline 또는 AWS-RunPatchBaselineAssociation 문서에 대해 InstallOverrideList 파라미터를 사용하여 https URL 또는 Amazon Simple Storage Service(Amazon S3) 경로 스타일 URL을 사용하여 패치 목록을 지정한 경우 나열된 패치가 설치되고 2~7단계를 건너뜁니다.
2. 패치 기준에 지정된 대로 GlobalFilters를 적용하여 자격을 갖춘 패키지만 향후 프로세스에 사용되도록 유지됩니다.
3. 패치 기준에 지정된 대로 ApprovalRules를 적용합니다. 각 승인 규칙은 패키지를 승인된 것으로 정의할 수 있습니다.

   그러나 승인 규칙은 패치 기준을 생성하거나 마지막으로 업데이트할 때 비보안 업데이트 포함(Include nonsecurity updates) 확인란을 선택했는지 여부에 따라 달라집니다.

   비보안 업데이트가 제외되면 보안 리포지토리에서 업그레이드가 있는 패키지만 선택할 수 있도록 묵시적인 규칙이 적용됩니다. 패키지마다, 후보 패키지 버전(일반적으로 최신 버전)이 보안 리포지토리에 속해 있어야 합니다.

   비보안 업데이트가 포함된 경우 다른 리포지토리의 패치도 고려됩니다.
4. 패치 기준에 지정된 대로 ApprovedPatches를 적용합니다. 승인된 패치는 GlobalFilters를 통해 폐기되었거나 ApprovalRules에 지정된 승인 규칙이 이를 승인하지 않더라도 업데이트가 승인됩니다.
5. 패치 기준에 지정된 대로 RejectedPatches를 적용합니다. 거부된 패치는 승인된 패치 목록에서 제거되고 적용되지 않습니다.
6. 여러 패치 버전이 승인되는 경우 최신 버전이 적용됩니다.
7. Zypper 업데이트 API가 승인된 패치에 적용됩니다.
8. 업데이트가 설치되어 있는 경우 관리형 노드가 재부팅됩니다. (예외: AWS-RunPatchBaseline 문서에서 RebootOption 파라미터가 NoReboot로 설정되어 있으면 Patch Manager를 실행한 후 관리형 노드가 재부팅되지 않습니다. 자세한 내용은 파라미터 이름: RebootOption 섹션을 참조하세요.)

Ubuntu 서버

Ubuntu Server 관리형 노드에서 패치 설치 워크플로는 다음과 같습니다.

1. `AWS-RunPatchBaseline` 또는 `AWS-RunPatchBaselineAssociation` 문서에 대해 `InstallOverrideList` 파라미터를 사용하여 https URL 또는 Amazon Simple Storage Service(Amazon S3) 경로 스타일 URL을 사용하여 패치 목록을 지정한 경우 나열된 패치가 설치되고 2~7단계를 건너뜁니다.
2. `python3-apt`(`libapt`에 대한 Python 라이브러리 인터페이스)에 대한 업데이트가 있는 경우 최신 버전으로 업그레이드됩니다. ([비보안 업데이트 포함(Include nonsecurity updates)] 옵션을 선택하지 않은 경우에도 이 비보안 패키지가 업그레이드됩니다.)
3. 패치 기준에 지정된 대로 GlobalFilters를 적용하여 자격을 갖춘 패키지만 향후 프로세스에 사용되도록 유지됩니다.
4. 패치 기준에 지정된 대로 ApprovalRules를 적용합니다. 각 승인 규칙은 패키지를 승인된 것으로 정의할 수 있습니다.

> **Note**
>
> Ubuntu Server에 대한 업데이트 패키지의 릴리스 날짜를 확실히 결정할 수 없으므로 이 운영 체제에서는 자동 승인 옵션이 지원되지 않습니다.

그러나 승인 규칙은 패치 기준을 생성하거나 마지막으로 업데이트할 때 비보안 업데이트 포함(Include nonsecurity updates) 확인란을 선택했는지 여부에 따라 달라집니다.

비보안 업데이트가 제외되면 보안 리포지토리에서 업그레이드가 있는 패키지만 선택할 수 있도록 묵시적인 규칙이 적용됩니다. 패키지마다, 후보 패키지 버전(일반적으로 최신 버전)이 보안 리포지토리에 속해 있어야 합니다.

비보안 업데이트가 포함된 경우 다른 리포지토리의 패치도 고려됩니다.

그러나 승인 규칙은 패치 기준을 생성하거나 마지막으로 업데이트할 때 [비보안 업데이트 포함(Include nonsecurity updates)] 확인란을 선택했는지 여부에 따라 달라집니다.

> **Note**
>
> Ubuntu Server의 각 버전에서 패치 후보 버전은 다음과 같이 해당 버전의 관련 리포지토리에 속하는 패치로 제한됩니다.

- 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-lobster`

5. 패치 기준에 지정된 대로 ApprovedPatches를 적용합니다. 승인된 패치는 GlobalFilters를 통해 폐기되었거나 ApprovalRules에 지정된 승인 규칙이 이를 승인하지 않더라도 업데이트가 승인됩니다.
6. 패치 기준에 지정된 대로 RejectedPatches를 적용합니다. 거부된 패치는 승인된 패치 목록에서 제거되고 적용되지 않습니다.
7. APT 라이브러리가 사용되어 패키지가 업그레이드됩니다.

> **Note**
>
> Patch Manager에서는 APT `Pin-Priority` 옵션을 사용하여 패키지에 우선순위를 지정하는 기능을 지원하지 않습니다. Patch Manager에서는 활성화된 모든 리포지토리에서 사용 가능한 업데이트를 집계하고 설치된 각 패키지의 기준과 일치하는 최신 업데이트를 선택합니다.

8. 업데이트가 설치되어 있는 경우 관리형 노드가 재부팅됩니다. (예외: `AWS-RunPatchBaseline` 문서에서 `RebootOption` 파라미터가 `NoReboot`로 설정되어 있으면 Patch Manager를 실행한 후 관리형 노드가 재부팅되지 않습니다. 자세한 내용은 파라미터 이름: `RebootOption` 섹션을 참조하세요.)

Windows Server

Windows Server 관리형 노드에서 패치 작업이 수행되는 경우 노드가 Systems Manager에서 해당되는 패치 기준 스냅샷을 요청합니다. 이 스냅샷에는 배포하도록 승인된 패치 기준에 있는 사용 가능한 모든 업데이트 목록이 들어 있습니다. 이 업데이트 목록이 Windows 업데이트 API로 전송되며, 여기서 관리형 노드에 적용 가능한 업데이트를 결정하고 필요 시 이를 설치합니다. 업데이트가 설치되면, 필요한 모든 패치 적용 작업을 완료하기 위해 필요한 만큼 관리형 노드가 재부팅됩니다. (예외: `AWS-RunPatchBaseline` 문서에서 `RebootOption` 파라미터가

NoReboot로 설정되어 있으면 Patch Manager를 실행한 후 관리형 노드가 재부팅되지 않습니다. 자세한 내용은 파라미터 이름: RebootOption 섹션을 참조하세요.) 패치 적용 작업에 대한 정보는 Run Command 요청의 결과에 요약되어 표시됩니다. 추가 로그는 %PROGRAMDATA%\Amazon\PatchBaselineOperations\Logs 폴더의 관리형 노드에서 찾을 수 있습니다.

패치를 다운로드하고 설치하는 데 Windows Update API가 사용되므로 Windows Update용 모든 그룹 정책 설정이 유지됩니다. Patch Manager를 사용하는 데는 그룹 정책 설정이 필요하지 않지만, 관리형 노드가 Windows Server Update Services(WSUS) 서버로 향하도록 하는 것과 같은 정의된 설정은 적용됩니다.

> **Note**
>
> Patch Manager는 Windows Update API를 사용하여 패치의 다운로드 및 설치를 진행하므로 기본적으로 Windows는 Microsoft의 Windows Update 사이트로부터 모든 패치를 다운로드합니다. 따라서 관리형 노드에서 Microsoft Windows Update 사이트에 연결할 수 있어야 하며, 그렇지 않으면 패치가 적용되지 않습니다. 또는 WSUS 서버가 패치 리포지토리 역할을 수행하도록 구성하고, 관리형 노드가 그룹 정책을 사용하는 해당 WSUS 서버를 대상으로 지정하도록 구성할 수 있습니다.

Linux 기반 시스템에서 패치 기준 규칙 사용 방법

Linux 배포용 패치 기준의 규칙은 배포 유형에 따라 다르게 작동합니다. Windows Server 관리형 노드의 패치 업데이트와 달리 규칙은 인스턴스의 구성된 리포지토리를 고려하기 위해 각 노드에서 평가됩니다. AWS Systems Manager의 기능인 Patch Manager는 기본 패키지 관리자를 사용하여 패치 기준에서 승인한 패치 설치를 진행합니다.

패치의 심각도 수준을 보고하는 Linux 기반 운영 체제 유형의 경우 Patch Manager는 업데이트 알림 또는 개별 패치에 대해 소프트웨어 게시자가 보고한 심각도 수준을 사용합니다. Patch Manager는 CVSS(Common Vulnerability Scoring System)와 같은 서드 파티 소스 또는 NVD(National Vulnerability Database)에서 발표한 지표에서 심각도 수준을 도출하지 않습니다.

주제

- Amazon Linux 1, Amazon Linux 2, Amazon Linux 2022, Amazon Linux 2023의 패치 기준 규칙 작동 방식
- CentOS 및 CentOS Stream에서 패치 기준선 규칙의 작동 방법
- Debian Server 및 Raspberry Pi OS에서 패치 기준 규칙이 작동하는 방법
- macOS에서 패치 기준 규칙 작동 방법

- Oracle Linux에서 패치 기준 규칙 작동 방법
- AlmaLinux, RHEL 및 Rocky Linux의 패치 기준선 규칙 작동 방식
- SUSE Linux Enterprise Server에서 패치 기준 규칙 작동 방법
- Ubuntu Server에서 패치 기준 규칙 작동 방법

Amazon Linux 1, Amazon Linux 2, Amazon Linux 2022, Amazon Linux 2023의 패치 기준 규칙 작동 방식

Amazon Linux 1, Amazon Linux 2, Amazon Linux 2022, Amazon Linux 2023의 패치 선택 프로세스는 다음과 같습니다.

1. 관리형 노드에서 YUM 라이브러리(Amazon Linux 1, Amazon Linux 2) 또는 DNF 라이브러리(Amazon Linux 2022 및 Amazon Linux 2023)는 구성된 리포지토리마다 `updateinfo.xml` 파일에 액세스합니다.

   > **Note**
   >
   > `updateinfo.xml` 파일이 없는 경우 비보안 업데이트 포함 및 자동 승인 설정에 따라 패치 설치 여부가 결정됩니다. 예를 들어 비보안 업데이트가 허용된 경우 자동 승인 시간이 되면 설치됩니다.

2. `updateinfo.xml`에 있는 각 업데이트 알림에는 다음 표에 설명된 대로, 패키지의 속성을 알림에 나타내는 여러 속성이 들어 있습니다.

   업데이트 알림 속성

속성	설명
형식	패치 기준의 PatchFilter 데이터 유형에 있는 분류 키 속성의 값에 해당합니다. 업데이트 알림에 들어 있는 패키지 유형을 나타냅니다. AWS CLI 명령 describe-patch-properties 또는 API 작업 DescribePatchProperties를 사용하여 지원되는 값 목록을 볼 수 있습니다. 또한 Systems Manager 콘솔의 [패치 기준 생성(Create patch baseline)] 페이지 또는 [패치 기준 편집(Edit patch baseline)] 페이지의 [승인

속성	설명
	규칙(Approval rules)] 영역에서 목록을 볼 수 있습니다.
severity	패치 기준의 PatchFilter 데이터 유형에 있는 심각도 키 속성의 값에 해당합니다. 업데이트 알림에 들어 있는 패키지의 심각도를 나타냅니다. 보통 심각도 업데이트 알림에만 적용 가능합니다. AWS CLI 명령 describe-patch-properties 또는 API 작업 DescribePatchProperties를 사용하여 지원되는 값 목록을 볼 수 있습니다. 또한 Systems Manager 콘솔의 [패치 기준 생성(Create patch baseline)] 페이지 또는 [패치 기준 편집(Edit patch baseline)] 페이지의 [승인 규칙(Approval rules)] 영역에서 목록을 볼 수 있습니다.
update_id	ALAS-2017-867과 같은 자문 ID를 나타냅니다. 자문 ID는 패치 기준의 ApprovedPatches 또는 RejectedPatches 속성에서 사용될 수 있습니다.
references	CVE ID(형식: CVE-2017-1234567)와 같은 업데이트 알림에 대한 추가 정보가 들어 있습니다. CVE ID는 패치 기준의 ApprovedPatches 또는 RejectedPatches 속성에서 사용될 수 있습니다.
updated	패치 기준의 ApproveAfterDays에 해당됩니다. 업데이트 알림에 들어 있는 패키지의 릴리스 날짜(업데이트된 날짜)를 나타냅니다. 현재 타임스탬프와 이 속성의 값에 `ApproveAfterDays` 를 더한 수를 비교하여 배포 승인을 받은 패치인지 확인합니다.

> **Note**
>
> 승인된 패치 및 거부된 패치 목록의 승인된 형식에 대한 자세한 내용은 승인 패치 및 거부 패치 목록의 패키지 이름 형식 정보 섹션을 참조하세요.

3. SSM Agent는 관리형 노드의 프로덕트를 확인합니다. 이 속성은 패치 기준의 PatchFilter 데이터 유형에 있는 상품 키 속성의 값에 해당합니다.
4. 패키지는 다음 지침에 따라 업데이트 대상으로 선택됩니다.

보안 옵션	패치 선택
AWS에서 제공하는 미리 정의된 기본 패치 기준선 및 비보안 업데이트 포함이 선택되지 않은 사용자 지정 패치 기준선	`updateinfo.xml` 에 있는 업데이트 알림마다, 패치 기준이 필터로 사용되어, 자격을 갖춘 패키지만 업데이트에 포함되도록 허용합니다. 패치 기준 정의를 적용한 후에도 여러 패키지가 남아 있으면 최신 버전이 사용됩니다. Amazon Linux 1 및 Amazon Linux 2의 경우 이 워크플로와 동등한 yum 명령은 다음과 같습니다. `sudo yum update-minimal --sec-severity=Critical,Important --bugfix -y` Amazon Linux 2022 및 Amazon Linux 2023의 경우 이 워크플로와 동등한 dnf 명령은 다음과 같습니다. `sudo dnf upgrade-minimal --sec-severity=Critical --sec-severity=Important --bugfix -y`
`[Critical, Important]` 의 심각도 목록 및 `[Security, Bugfix]` 의 분류 목록	Patch Manager는 `updateinfo.xml` 에서 선택한 보안 업데이트를 적용할 뿐만 아니라, 그

보안 옵션	패치 선택
과 함께 비보안 업데이트 포함 확인란이 선택된 사용자 지정 패치 기준선	렇지 않을 경우 패치 필터링 규칙을 준수하는 비보안 업데이트를 적용합니다. Amazon Linux 및 Amazon Linux 2의 경우 이 워크플로와 동등한 yum 명령은 다음과 같습니다. `sudo yum update-minimal --security --sec-severity=Critical,Important --bugfix -y` Amazon Linux 2022 및 Amazon Linux 2023의 경우 이 워크플로와 동등한 dnf 명령은 다음과 같습니다. `sudo dnf upgrade-minimal --security --sec-severity=Critical --sec-severity=Imporant --bugfix -y`

패치 규정 준수 상태 값에 대한 자세한 내용은 패치 규정 준수 상태 값 이해 섹션을 참조하세요.

CentOS 및 CentOS Stream에서 패치 기준선 규칙의 작동 방법

CentOS 및 CentOS Stream 기본 리포지토리에는 `updateinfo.xml` 파일이 포함되지 않습니다. 하지만 만들거나 사용하는 사용자 지정 리포지토리에는 이 파일이 포함될 수 있습니다. 이 항목에서 `updateinfo.xml` 참조는 이러한 사용자 지정 리포지토리에만 적용됩니다.

CentOS 및 CentOS Stream에서 패치 선택 프로세스는 다음과 같습니다.

1. 관리형 노드에서 YUM 라이브러리(CentOS 6.x 및 7.x 버전) 또는 DNF 라이브러리(CentOS 8.x 및 CentOS Stream)는 사용자 지정 리포지토리에 있는 경우 구성된 각 리포지토리의 `updateinfo.xml` 파일에 액세스합니다.

   항상 기본 리포지토리를 포함하는 `updateinfo.xml`이 없는 경우 비보안 업데이트 포함 및 자동 승인 설정에 따라 패치 설치 여부가 결정됩니다. 예를 들어 비보안 업데이트가 허용된 경우 자동 승인 시간이 되면 설치됩니다.

2. `updateinfo.xml`이 있는 경우 해당 파일의 각 업데이트 알림에는 다음 표에 설명된 대로, 패키지의 속성을 알림에 나타내는 여러 속성이 들어 있습니다.

   업데이트 알림 속성

속성	설명
형식	패치 기준의 PatchFilter 데이터 유형에 있는 분류 키 속성의 값에 해당합니다. 업데이트 알림에 들어 있는 패키지 유형을 나타냅니다. AWS CLI 명령 describe-patch-properties 또는 API 작업 DescribePatchProperties를 사용하여 지원되는 값 목록을 볼 수 있습니다. 또한 Systems Manager 콘솔의 [패치 기준 생성(Create patch baseline)] 페이지 또는 [패치 기준 편집(Edit patch baseline)] 페이지의 [승인 규칙(Approval rules)] 영역에서 목록을 볼 수 있습니다.
severity	패치 기준의 PatchFilter 데이터 유형에 있는 심각도 키 속성의 값에 해당합니다. 업데이트 알림에 들어 있는 패키지의 심각도를 나타냅니다. 보통 심각도 업데이트 알림에만 적용 가능합니다. AWS CLI 명령 describe-patch-properties 또는 API 작업 DescribePatchProperties를 사용하여 지원되는 값 목록을 볼 수 있습니다. 또한 Systems Manager 콘솔의 [패치 기준 생성(Create patch baseline)] 페이지 또는 [패치 기준 편집(Edit patch baseline)] 페이지의 [승인 규칙(Approval rules)] 영역에서 목록을 볼 수 있습니다.

속성	설명
update_id	CVE-2019-17055와 같은 자문 ID를 나타냅니다. 자문 ID는 패치 기준의 ApprovedPatches 또는 RejectedPatches 속성에서 사용될 수 있습니다.
references	CVE ID(형식: CVE-2019-17055) 또는 Bugzilla ID(형식: 1463241)와 같은 업데이트 알림에 대한 추가 정보가 들어 있습니다. CVE ID 및 Bugzilla ID는 패치 기준의 ApprovedPatches 또는 RejectedPatches 속성에서 사용될 수 있습니다.
updated	패치 기준의 ApproveAfterDays에 해당됩니다. 업데이트 알림에 들어 있는 패키지의 릴리스 날짜(업데이트된 날짜)를 나타냅니다. 현재 타임스탬프와 이 속성의 값에 `ApproveAfterDays` 를 더한 수를 비교하여 배포 승인을 받은 패치인지 확인합니다.

> **Note**
>
> 승인된 패치 및 거부된 패치 목록의 승인된 형식에 대한 자세한 내용은 승인 패치 및 거부 패치 목록의 패키지 이름 형식 정보 섹션을 참조하세요.

3. 모든 경우 SSM Agent는 관리형 노드의 프로덕트를 확인합니다. 이 속성은 패치 기준의 PatchFilter 데이터 유형에 있는 상품 키 속성의 값에 해당합니다.
4. 패키지는 다음 지침에 따라 업데이트 대상으로 선택됩니다.

보안 옵션	패치 선택
AWS에서 제공하는 미리 정의된 기본 패치 기준선 및 비보안 업데이트 포함이 선택되지 않은 사용자 지정 패치 기준선	사용자 지정 리포지토리에 있는 경우 `updateinfo.xml` 에 있는 업데이트 알림마다, 패치 기준이 필터로 사용되어, 자격을 갖춘 패키지만 업데이트에 포함되도록 허용합니다.

보안 옵션	패치 선택
	패치 기준 정의를 적용한 후에도 여러 패키지가 남아 있으면 최신 버전이 사용됩니다. `updateinfo.xml` 이 있는 CentOS 6 및 7의 경우 이 워크플로와 동등한 yum 명령은 다음과 같습니다. `sudo yum update-minimal --sec-severity=Critical,Important --bugfix -y` `updateinfo.xml` 이 있는 CentOS 8 및 CentOS Stream의 경우 이 워크플로와 동등한 yum 명령은 다음과 같습니다. `sudo dnf upgrade-minimal --sec-severity=Critical --sec-severity=Important --bugfix -y`

보안 옵션	패치 선택
[Critical, Important] 의 심각도 목록 및 [Security, Bugfix] 의 분류 목록과 함께 비보안 업데이트 포함 확인란이 선택된 사용자 지정 패치 기준선	사용자 지정 리포지토리에 있는 경우 Patch Manager는 updateinfo.xml 에서 선택한 보안 업데이트를 적용할 뿐만 아니라, 그렇지 않을 경우 패치 필터링 규칙을 준수하는 비보안 업데이트를 적용합니다. updateinfo.xml 이 있는 CentOS 6 및 7의 경우 이 워크플로와 동등한 yum 명령은 다음과 같습니다. `sudo yum update --sec-severity=Critical,Important --bugfix -y` updateinfo.xml 이 있는 CentOS 8 및 CentOS Stream의 경우 이 워크플로와 동등한 yum 명령은 다음과 같습니다. `sudo dnf upgrade --security --sec-severity=Critical --sec-severity=Important --bugfix -y` 기본 리포지토리와 updateinfo.xml 이 없는 사용자 지정 리포지토리의 경우 운영 체제(OS) 패키지를 업데이트하려면 비보안 업데이트 포함 확인란을 선택해야 합니다.

패치 규정 준수 상태 값에 대한 자세한 내용은 패치 규정 준수 상태 값 이해 섹션을 참조하세요.

Debian Server 및 Raspberry Pi OS에서 패치 기준 규칙이 작동하는 방법

Debian Server 및 Raspberry Pi OS(구 Raspbian)에서 패치 기준 서비스는 우선순위 및 섹션 필드에 필터링을 제공합니다. 이러한 필드는 일반적으로 모든 Debian Server 및 Raspberry Pi OS 패키지에 존재합니다. 패치가 패치 기준에 의해 선택된 것인지를 확인하기 위해 Patch Manager는 다음 작업을 수행합니다.

1. Debian Server 및 Raspberry Pi OS 시스템에서 `sudo apt-get update`와 동등한 명령이 실행되어 사용 가능한 패키지 목록을 새로 고칩니다. 리포지토리는 구성되지 않으며 데이터는 `sources` 목록에 구성되어 있는 리포지토리로부터 가져옵니다.
2. `python3-apt`(`libapt`에 대한 Python 라이브러리 인터페이스)에 대한 업데이트가 있는 경우 최신 버전으로 업그레이드됩니다. ([비보안 업데이트 포함(Include nonsecurity updates)] 옵션을 선택하지 않은 경우에도 이 비보안 패키지가 업그레이드됩니다.)

> **Important**
>
> Debian Server 8에만 해당: Debian Server 8.* 운영 체제는 더 이상 사용하지 않는 패키지 리포지토리(`jessie-backports`)를 참조하기 때문에 Patch Manager는 패치 작업이 성공할 수 있도록 다음과 같은 추가 단계를 수행해야 합니다.
>
> a. 관리형 노드에서 `jessie-backports` 리포지토리에 대한 참조는 소스 위치 목록(`/etc/apt/sources.list.d/jessie-backports`)에서 코멘트 아웃 처리됩니다. 따라서 해당 위치에서 패치를 다운로드하려고 시도하지 않습니다.
>
> b. Stretch 보안 업데이트 서명 키를 가져옵니다. 이 키는 Debian Server 8.* 배포의 업데이트 및 설치 작업에 필요한 권한을 제공합니다.
>
> c. 이 시점에서 패치 프로세스가 시작되기 전에 `python3-apt`의 최신 버전이 설치되도록 `apt-get` 작업이 실행됩니다.
>
> d. 설치 프로세스가 완료되면 `jessie-backports` 리포지토리에 대한 참조가 복원되고 서명 키가 apt 소스 키 링에서 제거됩니다. 이 작업은 패치 적용 작업 이전의 시스템 구성을 그대로 두기 위해 수행됩니다.

3. 다음으로 GlobalFilters, ApprovalRules, ApprovedPatches 및 RejectedPatches 목록이 적용됩니다.

> **Note**
>
> Debian Server에 대한 업데이트 패키지의 릴리스 날짜를 확실히 결정할 수 없으므로 이 운영 체제에서는 자동 승인 옵션이 지원되지 않습니다.

그러나 승인 규칙은 패치 기준을 생성하거나 마지막으로 업데이트할 때 비보안 업데이트 포함(Include nonsecurity updates) 확인란을 선택했는지 여부에 따라 달라집니다.

비보안 업데이트가 제외되면 보안 리포지토리에서 업그레이드가 있는 패키지만 선택할 수 있도록 묵시적인 규칙이 적용됩니다. 패키지마다, 후보 패키지 버전(일반적으로 최신 버전)이 보안 리포지

토리에 속해 있어야 합니다. 이 경우 Debian Server의 패치 후보 버전은 다음 리포지토리에 포함된 패치로 제한됩니다.

이러한 리포지토리는 다음과 같이 이름이 지정됩니다.

- Debian Server 8: `debian-security jessie`
- Debian Server 및 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`

비보안 업데이트가 포함된 경우 다른 리포지토리의 패치도 고려됩니다.

> **Note**
>
> 승인된 패치 및 거부된 패치 목록의 승인된 형식에 대한 자세한 내용은 승인 패치 및 거부 패치 목록의 패키지 이름 형식 정보 섹션을 참조하세요.

우선순위 및 섹션 필드의 내용을 보려면 다음 `aptitude` 명령을 실행하세요.

> **Note**
>
> 먼저 Debian Server 시스템에 Aptitude를 설치해야 할 수 있습니다.

```
aptitude search -F '%p %P %s %t %V#' '~U'
```

이 명령에 대한 응답으로, 업그레이드 가능한 모든 패키지가 다음 형식으로 보고됩니다.

```
name, priority, section, archive, candidate version
```

패치 규정 준수 상태 값에 대한 자세한 내용은 패치 규정 준수 상태 값 이해 섹션을 참조하세요.

macOS에서 패치 기준 규칙 작동 방법

macOS에서 패치 선택 프로세스는 다음과 같습니다.

1. 관리형 노드에서 Patch Manager는 `InstallHistory.plist` 파일의 구문 분석된 내용에 액세스하고 패키지 이름과 버전을 식별합니다.

   구문 분석 프로세스에 대한 자세한 내용은 패치 설치 방법의 macOS 섹션을 참조하세요.

2. SSM Agent는 관리형 노드의 프로덕트를 확인합니다. 이 속성은 패치 기준의 PatchFilter 데이터 유형에 있는 상품 키 속성의 값에 해당합니다.

3. 패키지는 다음 지침에 따라 업데이트 대상으로 선택됩니다.

보안 옵션	패치 선택
AWS에서 제공하는 미리 정의된 기본 패치 기준선 및 비보안 업데이트 포함이 선택되지 않은 사용자 지정 패치 기준선	사용 가능한 각 패키지 업데이트에 대해 패치 기준이 필터로 사용되어, 자격을 갖춘 패키지만 업데이트에 포함되도록 허용합니다. 패치 기준 정의를 적용한 후에도 여러 패키지가 남아 있으면 최신 버전이 사용됩니다.
비보안 업데이트 포함이 선택된 사용자 지정 패치 기준선	패치 관리자는 `InstallHistory.plist`를 사용하여 식별된 보안 업데이트를 적용할 뿐만 아니라, 그렇지 않을 경우 패치 필터링 규칙을 준수하는 비보안 업데이트를 적용합니다.

패치 규정 준수 상태 값에 대한 자세한 내용은 패치 규정 준수 상태 값 이해 섹션을 참조하세요.

Oracle Linux에서 패치 기준 규칙 작동 방법

Oracle Linux에서 패치 선택 프로세스는 다음과 같습니다.

1. 관리형 노드에서 YUM 라이브러리는 구성된 각 리포지토리에 대한 `updateinfo.xml` 파일에 액세스합니다.

> **Note**
>
> 리포지토리가 Oracle에서 관리하는 것이 아닌 경우 `updateinfo.xml` 파일을 사용할 수 없을 수 있습니다. `updateinfo.xml`이 없는 경우 비보안 업데이트 포함 및 자동 승인 설정에 따라 패치 설치 여부가 결정됩니다. 예를 들어 비보안 업데이트가 허용된 경우 자동 승인 시간이 되면 설치됩니다.

2. `updateinfo.xml`에 있는 각 업데이트 알림에는 다음 표에 설명된 대로, 패키지의 속성을 알림에 나타내는 여러 속성이 들어 있습니다.

   업데이트 알림 속성

속성	설명
형식	패치 기준의 PatchFilter 데이터 유형에 있는 분류 키 속성의 값에 해당합니다. 업데이트 알림에 들어 있는 패키지 유형을 나타냅니다. AWS CLI 명령 describe-patch-properties 또는 API 작업 DescribePatchProperties를 사용하여 지원되는 값 목록을 볼 수 있습니다. 또한 Systems Manager 콘솔의 [패치 기준 생성(Create patch baseline)] 페이지 또는 [패치 기준 편집(Edit patch baseline)] 페이지의 [승인 규칙(Approval rules)] 영역에서 목록을 볼 수 있습니다.
severity	패치 기준의 PatchFilter 데이터 유형에 있는 심각도 키 속성의 값에 해당합니다. 업데이트 알림에 들어 있는 패키지의 심각도를 나타냅니다. 보통 심각도 업데이트 알림에만 적용 가능합니다. AWS CLI 명령 describe-patch-properties 또는 API 작업 DescribePatchProperties를 사용하여 지원되는 값 목록을 볼 수 있습니다. 또한 Systems Manager 콘솔의 [패치 기준 생성(Create patch baseline)] 페이지 또는 [패치 기준 편집(Edit patch baseline)] 페이지의 [승인 규칙(Approval rules)] 영역에서 목록을 볼 수 있습니다.

속성	설명
update_id	CVE-2019-17055와 같은 자문 ID를 나타냅니다. 자문 ID는 패치 기준의 ApprovedPatches 또는 RejectedPatches 속성에서 사용될 수 있습니다.
references	CVE ID(형식: CVE-2019-17055) 또는 Bugzilla ID(형식: 1463241)와 같은 업데이트 알림에 대한 추가 정보가 들어 있습니다. CVE ID 및 Bugzilla ID는 패치 기준의 ApprovedPatches 또는 RejectedPatches 속성에서 사용될 수 있습니다.
updated	패치 기준의 ApproveAfterDays에 해당됩니다. 업데이트 알림에 들어 있는 패키지의 릴리스 날짜(업데이트된 날짜)를 나타냅니다. 현재 타임스탬프와 이 속성의 값에 `ApproveAfterDays` 를 더한 수를 비교하여 배포 승인을 받은 패치인지 확인합니다.

> **Note**
>
> 승인된 패치 및 거부된 패치 목록의 승인된 형식에 대한 자세한 내용은 승인 패치 및 거부 패치 목록의 패키지 이름 형식 정보 섹션을 참조하세요.

3. SSM Agent는 관리형 노드의 프로덕트를 확인합니다. 이 속성은 패치 기준의 PatchFilter 데이터 유형에 있는 상품 키 속성의 값에 해당합니다.
4. 패키지는 다음 지침에 따라 업데이트 대상으로 선택됩니다.

보안 옵션	패치 선택
AWS에서 제공하는 미리 정의된 기본 패치 기준선 및 비보안 업데이트 포함이 선택되지 않은 사용자 지정 패치 기준선	`updateinfo.xml` 에 있는 업데이트 알림마다, 패치 기준이 필터로 사용되어, 자격을 갖춘 패키지만 업데이트에 포함되도록 허용합니다.

보안 옵션	패치 선택
	패치 기준 정의를 적용한 후에도 여러 패키지가 남아 있으면 최신 버전이 사용됩니다. 버전 7 관리형 노드의 경우 이 워크플로와 동등한 yum 명령은 다음과 같습니다. `sudo yum update-minimal --sec-severity=Important,Moderate --bugfix -y` 버전 8 및 9 관리형 노드의 경우 이 워크플로와 동등한 dnf 명령은 다음과 같습니다. `sudo dnf upgrade-minimal --security --sec-severity=Moderate --sec-severity=Important`
`[Critical, Important]` 의 심각도 목록 및 `[Security, Bugfix]` 의 분류 목록과 함께 비보안 업데이트 포함이 선택된 사용자 지정 패치 기준선	Patch Manager는 `updateinfo.xml` 에서 선택한 보안 업데이트를 적용할 뿐만 아니라, 그렇지 않을 경우 패치 필터링 규칙을 준수하는 비보안 업데이트를 적용합니다. 버전 7 관리형 노드의 경우 이 워크플로와 동등한 yum 명령은 다음과 같습니다. `sudo yum update --security --sec-severity=Critical,Important --bugfix -y` 버전 8 및 9 관리형 노드의 경우 이 워크플로와 동등한 dnf 명령은 다음과 같습니다. `sudo dnf upgrade --security --sec-severity=Critical, --sec-severity=Important --bugfix y`

패치 규정 준수 상태 값에 대한 자세한 내용은 패치 규정 준수 상태 값 이해 섹션을 참조하세요.

AlmaLinux, RHEL 및 Rocky Linux의 패치 기준선 규칙 작동 방식

AlmaLinux, Red Hat Enterprise Linux (RHEL) 및 Rocky Linux의 패치 선택 프로세스는 다음과 같습니다.

1. 관리형 노드의 YUM 라이브러리(RHEL 7) 또는 DNF 라이브러리(AlmaLinux 8 및 9, RHEL 8 및 9, Rocky Linux 8 및 9)에서는 구성된 리포지토리마다 `updateinfo.xml` 파일에 액세스합니다.

   > **Note**
   >
   > 리포지토리가 Red Hat에서 관리하는 것이 아닌 경우 `updateinfo.xml` 파일을 사용할 수 없을 수 있습니다. `updateinfo.xml`을 찾을 수 없는 경우, 패치가 적용되지 않습니다.

2. `updateinfo.xml`에 있는 각 업데이트 알림에는 다음 표에 설명된 대로, 패키지의 속성을 알림에 나타내는 여러 속성이 들어 있습니다.

   업데이트 알림 속성

속성	설명
형식	패치 기준의 PatchFilter 데이터 유형에 있는 분류 키 속성의 값에 해당합니다. 업데이트 알림에 들어 있는 패키지 유형을 나타냅니다. AWS CLI 명령 describe-patch-properties 또는 API 작업 DescribePatchProperties를 사용하여 지원되는 값 목록을 볼 수 있습니다. 또한 Systems Manager 콘솔의 [패치 기준 생성(Create patch baseline)] 페이지 또는 [패치 기준 편집(Edit patch baseline)] 페이지의 [승인 규칙(Approval rules)] 영역에서 목록을 볼 수 있습니다.
severity	패치 기준의 PatchFilter 데이터 유형에 있는 심각도 키 속성의 값에 해당합니다. 업데이트 알림에 들어 있는 패키지의 심각도를 나타냅니다. 보통 심각도 업데이트 알림에만 적용 가능합니다.

속성	설명
	AWS CLI 명령 describe-patch-properties 또는 API 작업 DescribePatchProperties를 사용하여 지원되는 값 목록을 볼 수 있습니다. 또한 Systems Manager 콘솔의 [패치 기준 생성(Create patch baseline)] 페이지 또는 [패치 기준 편집(Edit patch baseline)] 페이지의 [승인 규칙(Approval rules)] 영역에서 목록을 볼 수 있습니다.
update_id	RHSA-2017:0864와 같은 자문 ID를 나타냅니다. 자문 ID는 패치 기준의 ApprovedPatches 또는 RejectedPatches 속성에서 사용될 수 있습니다.
references	CVE ID(형식: CVE-2017-1000371) 또는 Bugzilla ID(형식: 1463241)와 같은 업데이트 알림에 대한 추가 정보가 들어 있습니다. CVE ID 및 Bugzilla ID는 패치 기준의 ApprovedPatches 또는 RejectedPatches 속성에서 사용될 수 있습니다.
updated	패치 기준의 ApproveAfterDays에 해당됩니다. 업데이트 알림에 들어 있는 패키지의 릴리스 날짜(업데이트된 날짜)를 나타냅니다. 현재 타임스탬프와 이 속성의 값에 `ApproveAfterDays` 를 더한 수를 비교하여 배포 승인을 받은 패치인지 확인합니다.

> **Note**
>
> 승인된 패치 및 거부된 패치 목록의 승인된 형식에 대한 자세한 내용은 승인 패치 및 거부 패치 목록의 패키지 이름 형식 정보 섹션을 참조하세요.

3. SSM Agent는 관리형 노드의 프로덕트를 확인합니다. 이 속성은 패치 기준의 PatchFilter 데이터 유형에 있는 상품 키 속성의 값에 해당합니다.
4. 패키지는 다음 지침에 따라 업데이트 대상으로 선택됩니다.

보안 옵션	패치 선택
AWS에서 제공하는 미리 정의된 기본 패치 기준선 및 비보안 업데이트 포함 확인란이 어떤 규칙에서도 선택되지 않은 사용자 지정 패치 기준선	`updateinfo.xml` 에 있는 업데이트 알림마다, 패치 기준이 필터로 사용되어, 자격을 갖춘 패키지만 업데이트에 포함되도록 허용합니다. 패치 기준 정의를 적용한 후에도 여러 패키지가 남아 있으면 최신 버전이 사용됩니다. RHEL 7의 경우 이 워크플로와 동등한 yum 명령은 다음과 같습니다. `sudo yum update-minimal --sec-severity=Critical,Important --bugfix -y` AlmaLinux 8 및 9, RHEL 8 및 9 Rocky Linux 8 및 9의 경우 이 워크플로와 동등한 dnf 명령은 다음과 같습니다. `sudo dnf upgrade-minimal --sec-severity=Critical --sec-severity=Important --bugfix -y`
`[Critical, Important]` 의 심각도 목록 및 `[Security, Bugfix]` 의 분류 목록과 함께 비보안 업데이트 포함 확인란이 선택된 사용자 지정 패치 기준선	Patch Manager는 `updateinfo.xml` 에서 선택한 보안 업데이트를 적용할 뿐만 아니라, 그렇지 않을 경우 패치 필터링 규칙을 준수하는 비보안 업데이트를 적용합니다. RHEL 7의 경우 이 워크플로와 동등한 yum 명령은 다음과 같습니다. `sudo yum update --security --sec-severity=Critical,Important --bugfix -y`

보안 옵션	패치 선택
	AlmaLinux 8 및 9, RHEL 8 및 9 Rocky Linux 8 및 9의 경우 이 워크플로와 동등한 dnf 명령은 다음과 같습니다. `sudo dnf upgrade --sec-severity=Critical --sec-severity=Important --bugfix -y`

패치 규정 준수 상태 값에 대한 자세한 내용은 패치 규정 준수 상태 값 이해 섹션을 참조하세요.

SUSE Linux Enterprise Server에서 패치 기준 규칙 작동 방법

SLES에서 각 패치에는 패치에 있는 패키지 속성을 나타내는 다음 속성이 포함되어 있습니다.

- 카테고리: 패치 기준선의 PatchFilter 데이터 유형에 있는 분류 키 속성의 값에 해당합니다. 업데이트 알림에 들어 있는 패치 유형을 나타냅니다.

  AWS CLI 명령 describe-patch-properties 또는 API 작업 DescribePatchProperties를 사용하여 지원되는 값 목록을 볼 수 있습니다. 또한 Systems Manager 콘솔의 [패치 기준 생성(Create patch baseline)] 페이지 또는 [패치 기준 편집(Edit patch baseline)] 페이지의 [승인 규칙(Approval rules)] 영역에서 목록을 볼 수 있습니다.
- 심각도: 패치 기준선의 PatchFilter 데이터 유형에 있는 심각도 키 속성의 값에 해당합니다. 패치의 심각도를 나타냅니다.

  AWS CLI 명령 describe-patch-properties 또는 API 작업 DescribePatchProperties를 사용하여 지원되는 값 목록을 볼 수 있습니다. 또한 Systems Manager 콘솔의 [패치 기준 생성(Create patch baseline)] 페이지 또는 [패치 기준 편집(Edit patch baseline)] 페이지의 [승인 규칙(Approval rules)] 영역에서 목록을 볼 수 있습니다.

SSM Agent는 관리형 노드의 프로덕트를 확인합니다. 이 속성은 패치 기준선의 PatchFilter 데이터 유형에 있는 제품 키 속성의 값에 해당합니다.

각 패치에 대해 패치 기준이 필터로 사용되어, 자격을 갖춘 패키지만 업데이트에 포함되도록 허용합니다. 패치 기준 정의를 적용한 후에도 여러 패키지가 남아 있으면 최신 버전이 사용됩니다.

> **Note**
>
> 승인된 패치 및 거부된 패치 목록의 승인된 형식에 대한 자세한 내용은 승인 패치 및 거부 패치 목록의 패키지 이름 형식 정보 섹션을 참조하세요.

Ubuntu Server에서 패치 기준 규칙 작동 방법

Ubuntu Server에서 패치 기준 서비스는 우선 순위 및 섹션 필드에 필터링을 제공합니다. 이러한 필드는 일반적으로 모든 Ubuntu Server 패키지에 존재합니다. 패치가 패치 기준에 의해 선택된 것인지를 확인하기 위해 Patch Manager는 다음 작업을 수행합니다.

1. Ubuntu Server 시스템의 경우, `sudo apt-get update`와 동등한 명령이 실행되어 사용 가능한 패키지 목록을 새로 고칩니다. 리포지토리는 구성되지 않으며 데이터는 `sources` 목록에 구성되어 있는 리포지토리로부터 가져옵니다.
2. `python3-apt`(`libapt`에 대한 Python 라이브러리 인터페이스)에 대한 업데이트가 있는 경우 최신 버전으로 업그레이드됩니다. ([비보안 업데이트 포함(Include nonsecurity updates)] 옵션을 선택하지 않은 경우에도 이 비보안 패키지가 업그레이드됩니다.)
3. 다음으로 GlobalFilters, ApprovalRules, ApprovedPatches 및 RejectedPatches 목록이 적용됩니다.

> **Note**
>
> Ubuntu Server에 대한 업데이트 패키지의 릴리스 날짜를 확실히 결정할 수 없으므로 이 운영 체제에서는 자동 승인 옵션이 지원되지 않습니다.

그러나 승인 규칙은 패치 기준을 생성하거나 마지막으로 업데이트할 때 비보안 업데이트 포함(Include nonsecurity updates) 확인란을 선택했는지 여부에 따라 달라집니다.

비보안 업데이트가 제외되면 보안 리포지토리에서 업그레이드가 있는 패키지만 선택할 수 있도록 묵시적인 규칙이 적용됩니다. 패키지마다, 후보 패키지 버전(일반적으로 최신 버전)이 보안 리포지토리에 속해 있어야 합니다. 이 경우 Ubuntu Server의 패치 후보 버전은 다음 리포지토리에 포함된 패치로 제한됩니다.

- 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`)

비보안 업데이트가 포함된 경우 다른 리포지토리의 패치도 고려됩니다.

> **Note**
>
> 승인된 패치 및 거부된 패치 목록의 승인된 형식에 대한 자세한 내용은 승인 패치 및 거부 패치 목록의 패키지 이름 형식 정보 섹션을 참조하세요.

우선순위 및 섹션 필드의 내용을 보려면 다음 `aptitude` 명령을 실행하세요.

> **Note**
>
> 먼저 Ubuntu Server 16 시스템에 Aptitude를 설치해야 할 수 있습니다.

```
aptitude search -F '%p %P %s %t %V#' '~U'
```

이 명령에 대한 응답으로, 업그레이드 가능한 모든 패키지가 다음 형식으로 보고됩니다.

```
name, priority, section, archive, candidate version
```

패치 규정 준수 상태 값에 대한 자세한 내용은 패치 규정 준수 상태 값 이해 섹션을 참조하세요.

Linux 패치와 Windows 패치의 주요 차이점

이 주제에서는 AWS Systems Manager의 기능인 Patch Manager의 Linux 패치와 Windows 패치 간 주요 차이점에 대해 설명합니다.

> **Note**
>
> Linux 관리형 노드에 패치를 실행하려면 노드가 SSM Agent 버전 2.0.834.0 이상을 실행 중이어야 합니다.
>
> SSM Agent의 업데이트된 버전은 Systems Manager에 새 기능이 추가되거나 기존 기능에 업데이트가 발생할 때마다 릴리스됩니다. 최신 버전의 에이전트를 사용하지 못하면 관리형 노드

에서 다양한 Systems Manager 기능을 사용하지 못할 수 있습니다. 이러한 이유로 시스템의 SSM Agent 상태를 최신 상태로 유지하는 프로세스를 자동화하는 것이 좋습니다. 자세한 설명은 SSM Agent 업데이트 자동화을 참조하세요. SSM Agent 업데이트에 대해 알림을 수신하려면 GitHub에서 SSM Agent 릴리스 정보 페이지를 구독합니다.

차이점 1: 패치 평가

Linux

Linux 패치의 경우 Systems Manager는 패치 기준 규칙과 각 관리형 노드에서 승인 및 거부된 패치 목록을 평가합니다. 서비스가 관리형 노드에 구성된 리포지토리에서 알려진 패치 및 업데이트 목록을 검색하기 때문에 Systems Manager는 각 노드에서 패치를 평가해야 합니다.

Windows

Patch Manager는 Windows 관리형 노드와 Linux 관리형 노드에 적용되는 패치를 평가할 때 서로 다른 프로세스를 사용합니다. Windows 패치의 경우 Systems Manager는 패치 기준 규칙과 승인/거부된 패치 목록을 직접 서비스에서 평가합니다. Windows 패치는 단일 리포지토리(Windows 업데이트)에서 가져오기 때문에 이것이 가능합니다.

차이점 2: **`Not Applicable`** 패치

Linux 운영 체제에서는 사용할 수 있는 패키지가 매우 많기 때문에 Systems Manager가 [해당 사항 없음(Not Applicable)] 상태의 패치에 대해서는 세부 정보를 보고하지 않습니다. 예를 들어 `Not Applicable` 패치는 인스턴스에 Apache가 설치되어 있지 않은 경우 Apache 소프트웨어용 패치입니다. Systems Manager는 요약에서 `Not Applicable` 패치 수를 보고하지만 관리형 노드에 대해 DescribeInstancePatches API를 호출하면 상태가 `Not Applicable`인 패치는 반환된 데이터에 포함되지 않습니다. Windows에서는 이러한 동작이 다릅니다.

차이점 3: SSM 문서 지원

`AWS-ApplyPatchBaseline` Systems Manager 문서(SSM 문서)는 Linux 관리형 노드를 지원하지 않습니다. Linux, macOS 및 Windows Server 관리형 노드에 패치 기준을 적용하기 위해 권장되는 SSM 문서는 `AWS-RunPatchBaseline`입니다. 자세한 내용은 관리형 노드 패치를 위한 SSM 문서 정보 및 `AWS-RunPatchBaseline` SSM 문서 정보 단원을 참조하세요.

차이점 4: 애플리케이션 패치

Patch Manager는 운영 체제를 패치하는 데 중점을 둡니다. 그러나 Patch Manager를 사용하여 관리형 노드의 일부 애플리케이션을 패치할 수도 있습니다.

Linux

Linux 운영 체제에서 Patch Manager는 업데이트를 위해 구성된 리포지토리를 사용하며 운영 체제와 애플리케이션 패치를 구분하지 않습니다. Patch Manager를 사용하여 업데이트를 가져올 리포지토리를 정의할 수 있습니다. 자세한 내용은 대체 패치 소스 리포지토리를 지정하는 방법(Linux) 단원을 참조하십시오.

Windows

Windows Server 관리형 노드에서는 Microsoft Word 2016, Microsoft Exchange Server 2016과 같이 Microsoft에서 출시한 애플리케이션에 대해 승인된 패치와 거부된 패치 예외는 물론, 승인 규칙을 적용할 수 있습니다. 자세한 내용은 사용자 정의 패치 기준 작업 단원을 참조하십시오.

관리형 노드 패치를 위한 SSM 문서 정보

이 주제에서는 보안과 관련된 최신 업데이트를 통해 관리형 노드를 계속 패치하는 데 활용할 수 있는 9개의 Systems Manager 문서(SSM 문서)에 대해 설명합니다.

패치 작업에서는 이들 문서 중 5개만 사용할 것을 권장하고 있습니다. 5개의 SSM 문서를 함께 참조하면 AWS Systems Manager를 사용한 다양한 패치 옵션에 대해 배울 수 있습니다. 이 문서 중 4개는 대체된 4개의 레거시 SSM 문서보다 나중에 발표되었으며 기능의 확장 또는 통합을 나타냅니다.

패치 작업에 권장되는 SSM 문서

패치 작업에서는 다음과 같은 5개의 SSM 문서를 사용할 것을 권장합니다.

- `AWS-ConfigureWindowsUpdate`
- `AWS-InstallWindowsUpdates`
- `AWS-RunPatchBaseline`
- `AWS-RunPatchBaselineAssociation`
- `AWS-RunPatchBaselineWithHooks`

패치 작업을 위한 기존 SSM 문서

다음 4개의 기존 SSM 문서는 일부 AWS 리전에서 계속 사용할 수 있지만 더 이상 업데이트되지 않고 모든 시나리오에서의 작동이 보장되지 않으며 향후 더 이상 지원되지 않을 수 있습니다. 패치 작업에 사용하지 않는 것이 좋습니다.

- `AWS-ApplyPatchBaseline`

- `AWS-FindWindowsUpdates`
- `AWS-InstallMissingWindowsUpdates`
- `AWS-InstallSpecificWindowsUpdates`

패치 적용 작업 시 이러한 SSM 문서를 사용하는 방법에 대한 자세한 내용은 다음 섹션을 참조하세요.

주제

- 관리형 노드 패치를 위한 권장 SSM 문서 정보
- 관리형 노드 패치를 위한 SSM 레거시
- AWS-RunPatchBaseline SSM 문서 정보
- AWS-RunPatchBaselineAssociation SSM 문서 정보
- AWS-RunPatchBaselineWithHooks SSM 문서 정보
- AWS-RunPatchBaseline 또는 AWS-RunPatchBaselineAssociation에 InstallOverrideList 파라미터를 사용하기 위한 샘플 시나리오
- BaselineOverride 파라미터 사용

관리형 노드 패치를 위한 권장 SSM 문서 정보

관리형 노드 패치 작업에서는 다음과 같은 5개의 SSM 문서를 사용할 것을 권장합니다.

권장되는 SSM 문서

- AWS-ConfigureWindowsUpdate
- AWS-InstallWindowsUpdates
- AWS-RunPatchBaseline
- AWS-RunPatchBaselineAssociation
- AWS-RunPatchBaselineWithHooks

`AWS-ConfigureWindowsUpdate`

기본적인 Windows Update 기능을 구성하고 이들을 사용하여 업데이트를 자동 설치하거나 자동 업데이트를 해제할 수 있도록 돕습니다. 모든 AWS 리전에서 사용 가능합니다.

이 SSM 문서에서는 Windows 업데이트에게 지정된 업데이트를 다운로드 및 설치하고 필요에 따라 관리형 노드를 재부팅할 것을 요구합니다. AWS Systems Manager의 기능인 State Manager에서 이 문서

를 사용하여 Windows Update에서 구성을 유지할 수 있습니다. 또한 AWS Systems Manager의 기능인 Run Command를 사용하여 이를 수동으로 실행하여 Windows Update 구성을 변경할 수도 있습니다.

이 문서에서 사용 가능한 파라미터는 설치할 업데이트의 범주나 자동 업데이트의 해제 여부를 지정하는 것은 물론이고, 패치 작업이 실행되는 요일과 시간을 지정할 수 있도록 지원합니다. 이 SSM 문서는 Windows 업데이트를 엄격하게 제어할 필요가 없고 규정 준수 정보를 수집할 필요가 없는 경우에 매우 유용합니다.

기존 SSM 문서 대체:

- None(없음)

AWS-InstallWindowsUpdates

Windows Server 관리형 노드에 업데이트를 설치합니다. 모든 AWS 리전에서 사용 가능합니다.

이 SSM 문서는 특정 업데이트를 설치하고 싶은 경우(`Include Kbs` 파라미터 사용), 또는 특정 분류 또는 범주에 따라 패치를 설치하되, 패치 규정 준수 정보가 필요하지 않은 경우에 기본적인 패치 기능을 제공합니다.

기존 SSM 문서 대체:

- `AWS-FindWindowsUpdates`
- `AWS-InstallMissingWindowsUpdates`
- `AWS-InstallSpecificWindowsUpdates`

3개의 레거시 문서는 서로 다른 역할을 수행하지만, 최신 SSM 문서인 `AWS-InstallWindowsUpdates`에서는 서로 다른 파라미터 설정을 사용해도 동일한 결과를 얻을 수 있습니다. 이러한 파라미터 설정은 관리형 노드 패치를 위한 SSM 레거시에 설명되어 있습니다.

AWS-RunPatchBaseline

관리형 노드에 패치를 설치하거나 노드를 스캔하여 기준에 부합하는 패치의 누락 여부를 판단합니다. 모든 AWS 리전에서 사용 가능합니다.

`AWS-RunPatchBaseline`을 사용하면 운영 체제 유형에 대해 "기본값"으로 지정된 패치 기준을 사용하여 패치 승인을 제어할 수 있습니다. Systems Manager Compliance 도구를 사용하여 확인할 수 있는 패치 규정 준수 정보를 보고합니다. 이러한 도구들은 어떤 노드에서 패치가 누락되었는지, 어떤

패치가 누락되었는지와 같이 관리형 노드의 패치 규정 준수 상태에 대한 통찰력을 제공합니다. AWS-RunPatchBaseline을 사용하면 PutInventory API 명령을 사용하여 패치 규정 준수 정보가 기록됩니다. Linux 운영 체제의 경우 관리형 노드에 구성된 기본 소스 리포지토리와 사용자 지정 패치 기준에서 지정한 대체 소스 리포지토리의 패치에 대한 규정 준수 정보가 제공됩니다. 대체 소스 리포지토리에 대한 자세한 내용은 대체 패치 소스 리포지토리를 지정하는 방법(Linux) 섹션을 참조하세요. Systems Manager Compliance 도구에 대한 자세한 내용은 AWS Systems Manager Compliance 섹션을 참조하세요.

기존 문서 대체:

- AWS-ApplyPatchBaseline

레거시 문서 AWS-ApplyPatchBaseline은 Windows Server 관리형 노드에만 적용되며 애플리케이션 패치를 지원하지 않습니다. 최신 버전의 AWS-RunPatchBaseline은 Windows 및 Linux 시스템 모두를 동일하게 지원합니다. AWS-RunPatchBaseline 문서를 사용하려면 SSM Agent 버전 2.0.834.0 이상이 필요합니다.

AWS-RunPatchBaseline SSM 문서에 대한 자세한 내용은 AWS-RunPatchBaseline SSM 문서 정보 섹션을 참조하세요.

AWS-RunPatchBaselineAssociation

인스턴스에 패치를 설치하거나 인스턴스를 스캔하여 기준에 부합하는 패치의 누락 여부를 판단합니다. 모든 상용 AWS 리전에서 사용 가능합니다.

AWS-RunPatchBaselineAssociation은 다음과 같은 몇 가지 중요한 면에서 AWS-RunPatchBaseline과 다릅니다.

- AWS-RunPatchBaselineAssociation은 주로 AWS Systems Manager의 기능인 Quick Setup을 사용하여 생성된 State Manager 연결에 사용하도록 만들어졌습니다. 특히 Quick Setup 호스트 관리 구성 유형을 사용하는 경우, Scan instances for missing patches daily((매일 인스턴스에서 누락된 패치 스캔) 옵션을 선택하면 작업에 AWS-RunPatchBaselineAssociation이 사용됩니다.

  그러나 대부분의 경우 패치 작업을 직접 설정할 때 AWS-RunPatchBaselineAssociation 대신 AWS-RunPatchBaseline 또는 AWS-RunPatchBaselineWithHooks를 선택해야 합니다.

  자세한 정보는 다음 주제를 참조하세요.

  - AWS Systems Manager Quick Setup
  - AWS-RunPatchBaselineAssociation SSM 문서 정보

- AWS-RunPatchBaselineAssociation은 실행 시 대상 집합에 사용할 패치 기준 식별을 위해 태그 사용을 지원합니다.
- AWS-RunPatchBaselineAssociation을 사용하는 패치 작업의 경우 패치 규정 준수 데이터는 특정 State Manager 연결 측면에서 컴파일됩니다. AWS-RunPatchBaselineAssociation 실행 시 수집된 패치 규정 준수 데이터는 PutInventory 명령 대신 PutComplianceItems API 명령을 사용하여 기록됩니다. 이렇게 하면 이 특정 연결과 연결되지 않은 규정 준수 데이터를 덮어쓰는 것을 방지할 수 있습니다.

  Linux 운영 체제의 경우 인스턴스에 구성된 기본 소스 리포지토리와 사용자 지정 패치 기준에서 지정한 대체 소스 리포지토리의 패치에 대한 규정 준수 정보가 제공됩니다. 대체 소스 리포지토리에 대한 자세한 내용은 대체 패치 소스 리포지토리를 지정하는 방법(Linux) 섹션을 참조하세요. Systems Manager Compliance 도구에 대한 자세한 내용은 AWS Systems Manager Compliance 섹션을 참조하세요.

기존 문서 대체:

- None(없음)

AWS-RunPatchBaselineAssociation SSM 문서에 대한 자세한 내용은 AWS-RunPatchBaselineAssociation SSM 문서 정보 섹션을 참조하세요.

AWS-RunPatchBaselineWithHooks

관리형 노드에 패치를 설치하거나 노드를 스캔하여 패치 주기 동안 세 지점에서 SSM 문서를 실행하는 데 사용할 수 있는 옵션 후크로 정규화된 패치의 누락 여부를 판단합니다. 모든 상용 AWS 리전에서 사용 가능합니다.

AWS-RunPatchBaselineWithHooks는 Install 작업에 있어 AWS-RunPatchBaseline과 다릅니다.

AWS-RunPatchBaselineWithHooks는 관리형 노드 패치 중 지정된 지점에서 실행되는 수명 주기 후크를 지원합니다. 패치 설치 시 관리형 노드를 재부팅해야 하는 경우가 있기 때문에 패치 작업은 사용자 정의 기능을 지원하는 총 3개의 후크에 대한 2개의 이벤트로 나뉩니다. 첫 번째 후크는 Install with NoReboot 작업 전입니다. 두 번째 후크는 Install with NoReboot 작업 후입니다. 세 번째 후크는 노드 재부팅 후 사용할 수 있습니다.

기존 문서 대체:

- None(없음)

AWS-RunPatchBaselineWithHooks SSM 문서에 대한 자세한 내용은 AWS-RunPatchBaselineWithHooks SSM 문서 정보 섹션을 참조하세요.

관리형 노드 패치를 위한 SSM 레거시

다음과 같은 4개의 SSM 문서는 일부 AWS 리전에서 계속 사용할 수 있습니다. 그러나 더 이상 업데이트되지 않으며 향후 지원되지 않을 수 있기 때문에 사용하지 않는 것이 좋습니다. 대신에 관리형 노드 패치를 위한 권장 SSM 문서 정보에 설명되어 있는 문서를 사용하십시오.

기존 SSM 문서

- AWS-ApplyPatchBaseline
- AWS-FindWindowsUpdates
- AWS-InstallMissingWindowsUpdates
- AWS-InstallSpecificWindowsUpdates

AWS-ApplyPatchBaseline

Windows Server 관리형 노드만 지원하지만, 이를 대체하는 AWS-RunPatchBaseline에 있는 패치 애플리케이션에 대한 지원은 포함하지 않습니다. 2017년 8월 이후에 출시된 AWS 리전에서는 사용할 수 없습니다.

> **Note**
>
> 이 SSM 문서 AWS-RunPatchBaseline을 대체하려면 버전 2.0.834.0 이상의 SSM Agent가 필요합니다. AWS-UpdateSSMAgent 문서를 사용하여 최신 버전의 에이전트로 관리형 노드를 업데이트할 수 있습니다.

AWS-FindWindowsUpdates

AWS-InstallWindowsUpdates에 의해 대체되지만, 동일한 모든 작업을 수행할 수 있습니다. 2017년 4월 이후에 출시된 AWS 리전에서는 사용할 수 없습니다.

이 레거시 SSM 문서에서와 동일한 결과를 얻으려면 권장되는 대체 문서인 AWS-InstallWindowsUpdates에서 다음과 같이 파라미터 구성을 사용합니다.

- Action = Scan
- Allow Reboot = False

AWS-InstallMissingWindowsUpdates

AWS-InstallWindowsUpdates에 의해 대체되지만, 동일한 모든 작업을 수행할 수 있습니다. 2017년 4월 이후에 출시된 AWS 리전에서는 사용할 수 없습니다.

이 레거시 SSM 문서에서와 동일한 결과를 얻으려면 권장되는 대체 문서인 AWS-InstallWindowsUpdates에서 다음과 같이 파라미터 구성을 사용합니다.

- Action = Install
- Allow Reboot = True

AWS-InstallSpecificWindowsUpdates

AWS-InstallWindowsUpdates에 의해 대체되지만, 동일한 모든 작업을 수행할 수 있습니다. 2017년 4월 이후에 출시된 AWS 리전에서는 사용할 수 없습니다.

이 레거시 SSM 문서에서와 동일한 결과를 얻으려면 권장되는 대체 문서인 AWS-InstallWindowsUpdates에서 다음과 같이 파라미터 구성을 사용합니다.

- Action = Install
- Allow Reboot = True
- Include Kbs = *KB ### ### ### ##*

AWS-RunPatchBaseline SSM 문서 정보

AWS Systems Manager는 AWS Systems Manager의 기능인 Patch Manager용 Systems Manager 문서(SSM 문서)인 AWS-RunPatchBaseline을 지원합니다. 이 SSM 문서는 보안 관련 업데이트 및 기타 유형의 업데이트 모두에 대해 관리형 노드에서 패치 작업을 수행합니다. 문서가 실행될 때 패치 그룹이 지정되지 않은 경우 운영 체제 유형에 대해 "기본값"으로 지정된 패치 기준을 사용합니다 그렇지 않으면 패치 그룹과 연결된 패치 기준을 사용합니다. 패치 그룹에 대한 자세한 내용은 패치 그룹 정보 섹션을 참조하세요.

문서 AWS-RunPatchBaseline을 사용하면 운영 체제와 애플리케이션 모두를 패치할 수 있습니다. (Windows Server에서 애플리케이션 지원은 Microsoft에서 릴리스한 애플리케이션의 업데이트로 제한됩니다.)

이 문서는 Linux, macOS, 및 Windows Server관리형 노드를 지원합니다. 문서에서 각 플랫폼에 적합한 작업을 수행합니다.

> **Note**
>
> Patch Manager는 레거시 SSM 문서 `AWS-ApplyPatchBaseline`도 지원합니다. 단, 이 문서는 Windows 관리형 노드에 대한 패치 적용 작업만 지원합니다. `AWS-RunPatchBaseline`이 Linux, macOS 및 Windows Server 관리형 노드 모두에서 패치 작업을 지원하므로 이를 대신 사용하는 것이 좋습니다. `AWS-RunPatchBaseline` 문서를 사용하려면 SSM Agent 버전 2.0.834.0 이상이 필요합니다.

Windows Server

Windows Server 관리형 노드에서는 `AWS-RunPatchBaseline` 문서가 PowerShell 모듈을 다운로드하고 호출합니다. 그런 다음, 해당 관리형 노드에 적용되는 패치 기준의 스냅샷을 다운로드합니다. 이 패치 기준 스냅샷에는 Windows Server Update Services(WSUS) 서버에 대해 패치 기준을 쿼리하여 컴파일된 승인된 패치 목록이 포함되어 있습니다. 이 목록은 Windows Update API에 전달되며, 이 API가 승인된 패치를 적절하게 다운로드하고 설치하는 작업을 제어합니다.

Linux

Linux 관리형 노드에서는 `AWS-RunPatchBaseline` 문서가 Python 모듈을 호출합니다. 그런 다음, 해당 관리형 노드에 적용되는 패치 기준의 스냅샷을 다운로드합니다. 이 패치 기준 스냅샷은 정의된 규칙과 승인 및 차단된 패치 목록을 사용하여 각 노드 유형에 적합한 패키지 관리자를 실행합니다.

- Amazon Linux 1, Amazon Linux 2, CentOS, Oracle Linux, RHEL 7 관리형 노드는 YUM을 사용합니다. YUM 작업의 경우 `Python 2.6` 이상의 지원되는 버전(2.6~3.10)이 Patch Manager에 필요합니다.
- RHEL 8 관리형 노드는 DNF를 사용합니다. DNF 작업의 경우 지원되는 `Python 2` 또는 `Python 3` 버전(2.6~3.10)이 Patch Manager에 필요합니다. (두 버전 모두 RHEL 8에 기본적으로 설치되어 있지 않습니다. 둘 중 하나를 수동으로 설치해야 합니다.)
- Debian Server, Raspberry Pi OS 및 Ubuntu Server 인스턴스는 APT를 사용합니다. APT 작업의 경우 지원되는 `Python 3` 버전(3.0~3.10)이 Patch Manager에 필요합니다.
- SUSE Linux Enterprise Server 관리형 노드는 Zypper를 사용합니다. Zypper 작업의 경우 `Python 2.6` 이상의 지원되는 버전(2.6~3.10)이 Patch Manager에 필요합니다.

macOS

macOS 관리형 노드에서는 AWS-RunPatchBaseline 문서가 Python 모듈을 호출합니다. 그런 다음, 해당 인스턴스에 적용되는 패치 기준의 스냅샷을 다운로드합니다. 다음으로 Python 하위 프로세스는 노드에서 AWS Command Line Interface(AWS CLI)를 호출하여 지정된 패키지 관리자에 대한 설치 및 업데이트 정보를 검색하고 각 업데이트 패키지에 대해 적절한 패키지 관리자를 구동합니다.

각 스냅샷은 AWS 계정, 패치 그룹, 운영 체제 및 스냅샷 ID에 따라 다릅니다. 스냅샷은 스냅샷 생성 후 24시간이 지나면 만료되는 미리 서명된 Amazon Simple Storage Service(Amazon S3) URL을 통해 전송됩니다. 그러나 URL이 만료된 후 동일한 스냅샷 콘텐츠를 다른 관리형 노드에 적용하려는 경우 스냅샷이 생성된 후 최대 3일 이내에 미리 서명된 Amazon S3 URL을 새로 생성할 수 있습니다. 이렇게 하려면 get-deployable-patch-snapshot-for-instance 명령을 사용합니다.

승인되고 적용 가능한 모든 업데이트가 설치되고 필요한 만큼 재부팅이 수행되면, 패치 규정 준수 정보가 관리형 노드에 생성되며 Patch Manager에 다시 보고됩니다.

> **Note**
>
> AWS-RunPatchBaseline 문서에서 RebootOption 파라미터가 NoReboot로 설정되어 있으면 Patch Manager를 실행한 후 관리형 노드가 재부팅되지 않습니다. 자세한 내용은 파라미터 이름: RebootOption 단원을 참조하십시오.

패치 규정 준수 데이터를 보는 방법에 대한 자세한 내용은 패치 규정 준수 정보 섹션을 참조하세요.

AWS-RunPatchBaseline 파라미터

AWS-RunPatchBaseline은 5개 파라미터를 지원합니다 Operation 파라미터가 필요합니다. InstallOverrideList, BaselineOverride 및 RebootOption 파라미터는 선택 항목입니다. Snapshot-ID는 엄밀히 말해 옵션이지만, 유지 관리 기간 이외에서 AWS-RunPatchBaseline을 실행하는 경우 이에 대한 사용자 정의 값을 제공하여, 유지 관리 기간 작업 동안에 문서가 실행되는 경우 Patch Manager가 값을 자동으로 제공하도록 하는 것이 좋습니다.

파라미터

- 파라미터 이름: Operation
- 파라미터 이름: AssociationId
- 파라미터 이름: Snapshot ID

- 파라미터 이름: InstallOverrideList
- 파라미터 이름: RebootOption
- 파라미터 이름: BaselineOverride

파라미터 이름: **`Operation`**

사용 여부: 필수.

옵션: `Scan` | `Install`.

스캔

`Scan` 옵션을 선택하는 경우 `AWS-RunPatchBaseline`에 따라 관리형 노드의 패치 규정 준수 상태가 결정되며 이 정보가 Patch Manager에게 다시 보고됩니다. `Scan`은 업데이트를 설치하거나 관리형 노드를 재부팅하라는 메시지를 표시하지 않습니다. 대신, 작업이 승인되고 노드에 적용 가능한 업데이트가 누락된 위치를 식별해 줍니다.

Install

`Install` 옵션을 선택하는 경우 `AWS-RunPatchBaseline`이 관리형 노드에서 누락된 승인되고 적용 가능한 업데이트를 설치하려고 시도합니다. `Install` 작업의 일부로 생성된 패치 규정 준수 정보에는 누락된 업데이트가 나열되지 않지만, 어떠한 이유로든 업데이트 설치가 성공하지 못하면 실패 상태에 있는 업데이트를 보고할 수 있습니다. 관리형 노드에 업데이트가 설치될 때마다 업데이트가 설치되었는지 및 활성 상태인지를 확인하기 위해 노드가 재부팅됩니다. (예외: `AWS-RunPatchBaseline` 문서에서 `RebootOption` 파라미터가 `NoReboot`로 설정되어 있으면 Patch Manager를 실행한 후 관리형 노드가 재부팅되지 않습니다. 자세한 내용은 파라미터 이름: `RebootOption` 섹션을 참조하세요.)

> **Note**
>
> Patch Manager가 관리형 노드를 업데이트하기 전에 기준 규칙에 의해 지정된 패치가 설치된 경우 시스템이 예상대로 재부팅되지 않을 수도 있습니다. 이는 패치가 사용자에 의해 수동으로 설치되어 있거나 Ubuntu Server의 `unattended-upgrades` 패키지와 같은 다른 프로그램에 의해 자동으로 설치되어 있는 경우에 발생할 수 있습니다.

파라미터 이름: **`AssociationId`**

사용 여부: 선택.

AssociationId는 AWS Systems Manager의 기능인 State Manager에 있는 기존 연결의 ID입니다. 이 ID는 Patch Manager에서 지정된 연결에 규정 준수 데이터를 추가하는 데 사용됩니다. 이 연결은 의 Quick Setup에서 패치 정책에 설정한 패치 작업과 관련이 있습니다.

> **Note**
>
> AWS-RunPatchBaseline을 사용하면 패치 정책 기준선 재정의와 함께 AssociationId 값이 제공될 경우, 패치 적용이 PatchPolicy 작업으로 수행되며, AWS:ComplianceItem에 보고되는 ExecutionType 값도 PatchPolicy가 됩니다. AssociationId 값이 제공되지 않으면, 패치 적용이 Command 작업으로 수행되며 제출된 AWS:ComplianceItem에 대해 보고되는 ExecutionType 값도 Command가 됩니다.

사용하려는 연결이 아직 없는 경우 create-association 명령을 실행하여 하나를 생성할 수 있습니다.

파라미터 이름: **Snapshot ID**

사용 여부: 선택.

Snapshot ID는 Patch Manager가 단일 작업으로 패치된 관리형 노드 집합 모두가 승인된 패치 집합과 정확히 일치하는지를 확인하기 위해 사용하는 고유 ID(GUID)입니다. 이 파라미터가 옵션으로 정의되어 있긴 하지만, 다음 표에 설명된 대로 유지 관리 기간에서 AWS-RunPatchBaseline을 실행하는지 여부에 따라 모범 사례 권장 사항이 달라집니다.

AWS-RunPatchBaseline 모범 사례

Mode	모범 사례	Details
유지 관리 기간 내 AWS-RunPatchBaseline 실행	스냅샷 ID를 제공하지 않습니다. Patch Manager가 제공합니다.	유지 관리 기간을 사용하여 AWS-RunPatchBaseline 을 실행하는 경우 자체 생성한 스냅샷 ID를 제공해선 안 됩니다. 이러한 경우에는 Systems Manager가 유지 관리 기간 실행 ID를 기반으로 GUID 값을 제공합니다. 이렇게 하면 해당 유지 관리 기간에 모든 AWS-RunPatchBaseline 호출에 올바른 ID가 사용됩니다.

Mode	모범 사례	Details
		이러한 경우에 값을 지정하는 경우 패치 기준선의 스냅샷이 3일 이상 적용되지 않을 수 있습니다. 해당 시간 경과 후에는 스냅샷 만료 후 동일한 ID를 지정하더라도 새로운 스냅샷이 생성됩니다.

Mode	모범 사례	Details
유지 관리 기간 외 AWS-RunPatchBaseline 실행	스냅샷 ID에 대해 사용자 정의 GUID 값을 생성하고 지정합니다.[1]	AWS-RunPatchBaseline 을 실행하는 데 유지 관리 기간을 사용하고 있지 않은 경우 각 패치 기준에 대해 고유한 스냅샷 ID를 생성하고 지정하는 것이 좋습니다. 특히 동일한 작업에서 여러 관리형 노드에 AWS-RunPatchBaseline 문서를 실행하고 있는 경우 더욱 그러합니다. 이러한 경우에 ID를 지정하지 않으면 Systems Manager가 명령을 보내는 각 관리형 노드에 다른 스냅샷 ID를 생성합니다. 이렇게 되면 관리형 노드 간에 다양한 패치 집합이 지정될 수 있습니다. 예를 들어 AWS Systems Manager의 기능인 Run Command를 통해 직접 AWS-RunPatchBaseline 문서를 실행하고 50개의 관리형 노드로 이루어진 그룹을 대상으로 한다고 가정해 보겠습니다. 사용자 지정 스냅샷 ID를 지정하면 모든 노드를 평가하고 패치를 적용하는 데 사용되는 기준 스냅샷이 하나가 생성되어 일관적인 상태를 유지하도록 해줍니다.

[1] GUID를 생성할 수 있는 도구라면 어떤 도구를 사용해서든 스냅샷 ID 파라미터의 값을 생성할 수 있습니다. 예를 들어 PowerShell의 경우 New-Guid cmdlet을 사용하여 12345699-9405-4f69-bc5e-9315aEXAMPLE 형식으로 GUID를 생성할 수 있습니다.

파라미터 이름: **InstallOverrideList**

사용 여부: 선택.

InstallOverrideList를 사용하면 설치하려는 패치 목록에 대해 https URL 또는 Amazon S3 경로 스타일 URL을 지정할 수 있습니다. YAML 형식으로 관리되는 이 패치 설치 목록은 현재 기본 패치 기준으로 지정된 패치를 재정의합니다. 그러면 관리형 노드에 설치되는 패치를 세부적으로 제어할 수 있습니다.

InstallOverrideList 파라미터를 사용할 때의 패치 작업 동작은 Linux 및 macOS 관리형 노드와 Windows Server 관리형 노드가 다릅니다. Linux 및 macOS에서는 Patch Manager가 패치가 패치 기준 규칙과 일치하는지 여부와 관계없이 노드에 활성화된 모든 리포지토리의 InstallOverrideList 패치 목록에 포함된 패치를 적용하려고 시도합니다. 그러나 Windows Server 노드에서는 InstallOverrideList 패치 목록의 패치가 패치 기준 규칙과도 일치하는 경우에만 적용됩니다.

규정 준수 보고서에서는 InstallOverrideList 패치 목록에 지정된 항목이 아니라 패치 기준선에 지정된 항목에 따라 패치 상태를 반영합니다. 다시 말해 스캔 작업에서 InstallOverrideList 파라미터를 무시합니다. 그래야만 규정 준수 보고서에서 특정 패치 적용 작업에 대해 승인된 내용이 아닌 정책에 따라 패치 상태를 일관되게 반영합니다.

InstallOverrideList 파라미터를 사용하여 단일 패치 기준을 계속 사용하면서 서로 다른 유형의 유지 관리 기간 일정에 따라 대상 그룹에 서로 다른 유형의 패치를 적용하는 방법에 대한 설명은 AWS-RunPatchBaseline 또는 AWS-RunPatchBaselineAssociation에 InstallOverrideList 파라미터를 사용하기 위한 샘플 시나리오 섹션을 참조하세요.

유효한 URL 형식

> **Note**
>
> 파일이 공개적으로 사용 가능한 버킷에 저장된 경우 https URL 형식 또는 Amazon S3 경로 스타일 URL을 지정할 수 있습니다. 파일이 프라이빗 버킷에 저장된 경우 Amazon S3 경로 스타일 URL을 지정해야 합니다.

- https URL 형식:

```
https://s3.aws-api-domain/DOC-EXAMPLE-BUCKET/my-windows-override-list.yaml
```

- Amazon S3 경로 스타일 URL:

```
s3://DOC-EXAMPLE-BUCKET/my-windows-override-list.yaml
```

유효한 YAML 콘텐츠 형식

목록에서 패치를 지정하는 데 사용되는 형식은 관리형 노드의 운영 체제에 따라 다릅니다. 일반적인 형식은 다음과 같습니다.

```
patches:
    -
        id: '{patch-d}'
        title: '{patch-title}'
        {additional-fields}:{values}
```

YAML 파일에 추가 필드를 제공할 수 있지만 해당 필드는 패치 작업 중에 무시됩니다.

또한 S3 버킷에서 목록을 추가하거나 업데이트하기 전에 YAML 파일 형식이 올바른지 확인하는 것이 좋습니다. YAML 형식에 대한 자세한 내용은 yaml.org를 참조하십시오. 유효한 도구 옵션을 보려면 웹에서 "yaml format validators"를 검색하십시오.

Linux

id

id 필드는 필수입니다. 패키지 이름 및 아키텍처를 사용하여 패치를 지정하는 데 사용됩니다. 예: `'dhclient.x86_64'`. id에서 와일드카드를 사용하여 여러 패키지를 나타낼 수 있습니다. 예를 들면 `'dhcp*'` 및 `'dhcp*1.*'` 등입니다.

Title

제목 필드는 선택 사항이지만 Linux 시스템의 경우 이 필드는 추가 필터링 기능을 제공합니다. title을 사용할 경우 패키지 버전 정보를 다음 중 한 가지 형식으로 포함해야 합니다.

YUM/SUSE Linux Enterprise Server(SLES):

```
{name}.{architecture}:{epoch}:{version}-{release}
```

APT

```
{name}.{architecture}:{version}
```

Linux 패치 제목의 경우 임의의 위치에서 하나 이상의 와일드카드를 사용하여 패키지 일치 수를 확장할 수 있습니다. 예: `'*32:9.8.2-0.*.rc1.57.amzn1'`.

예:

- apt 패키지 버전 1.2.25가 관리형 노드에 현재 설치되어 있지만 버전 1.2.27을 지금 사용할 수 있습니다.
- apt.amd64 버전 1.2.27을 패치 목록에 추가합니다. apt utils.amd64 버전 1.2.27을 사용하지만 apt-utils.amd64 버전 1.2.25가 목록에 지정되어 있습니다.

이 경우 apt 버전 1.2.27은 설치 시 차단되고 “Failed-NonCompliant”로 보고됩니다.

Windows Server

id

id 필드는 필수입니다. id 필드는 Microsoft Knowledge Base ID(예: KB2736693)와 Microsoft Security Bulletin ID(예: MS17-023)를 사용하여 패치를 지정하는 데 사용됩니다.

Windows에 대한 패치 목록에 제공하려는 다른 필드는 선택 사항이며 정보 제공의 목적으로만 사용됩니다. title, classification, severity 등과 같은 추가 필드를 사용하여 지정된 패치에 대한 자세한 정보를 제공할 수 있습니다.

macOS

id

id 필드는 필수입니다. id 필드의 값은 `{package-name}.{package-version}` 형식 또는 {package_name} 형식을 사용하여 제공할 수 있습니다.

샘플 패치 목록

- 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'
```

파라미터 이름: **RebootOption**

사용 여부: 선택.

옵션: RebootIfNeeded | NoReboot

기본값: RebootIfNeeded

> **Warning**
>
> 기본 옵션은 RebootIfNeeded입니다. 사용 사례에 맞는 올바른 옵션을 선택해야 합니다. 예를 들어 구성 프로세스를 완료하기 위해 관리형 노드를 즉시 재부팅해야 하는 경우, RebootIfNeeded를 선택합니다. 또는 예약된 재부팅 시간까지 관리형 노드 가용성을 유지해야 하는 경우, NoReboot를 선택합니다.

> **Important**
>
> Amazon EMR(이전 명칭: Amazon Elastic MapReduce)에서 클러스터 인스턴스 패치를 적용하는 데는 Patch Manager를 사용하지 않는 것이 좋습니다. 특히 RebootOption 파라미터에 RebootIfNeeded 옵션을 선택하지 마세요. (이 옵션은 AWS-RunPatchBaseline, AWS-RunPatchBaselineAssociation 및 AWS-RunPatchBaselineWithHooks 패치 적용에 대한 SSM 명령 문서에서 제공되지 않습니다.)

Patch Manager를 사용하는 패치 적용에 대한 기본 명령에서는 yum 및 dnf 명령을 사용합니다. 따라서 패키지 설치 방식 때문에 작업이 호환되지 않을 수 있습니다. Amazon EMR 클러스터의 소프트웨어 업데이트용으로 기본 설정된 방법에 대한 자세한 내용은 Amazon EMR 관리 안내서의 Amazon EMR용 기본 AMI 사용을 참조하세요.

RebootIfNeeded

RebootIfNeeded 옵션을 선택하면 다음과 같은 경우 관리형 노드가 재부팅됩니다.

- Patch Manager가 하나 이상의 패치를 설치했습니다.

  Patch Manager가 패치에 재부팅이 필요한지 여부를 평가하지 않습니다. 패치에 재부팅이 필요하지 않은 경우에도 시스템이 재부팅됩니다.

- Patch Manager가 Install 작업 중 INSTALLED_PENDING_REBOOT 상태의 패치를 하나 이상 감지합니다.

  INSTALLED_PENDING_REBOOT 상태는 Install 작업이 마지막으로 실행될 때 NoReboot 옵션이 선택되었거나 관리형 노드를 마지막으로 재부팅한 이후에 Patch Manager 외부에 패치가 설치되었다는 의미일 수 있습니다.

이 두 가지 경우에 관리형 노드를 재부팅하면 업데이트된 패키지가 메모리에서 플러시되고 모든 운영 체제에서 패치 및 재부팅 동작이 일관되게 유지됩니다.

NoReboot

NoReboot 옵션을 선택하면 Patch Manager가 Install 작업 중에 패치를 설치한 경우에도 관리형 노드를 재부팅하지 않습니다. 이 옵션은 패치된 후 관리형 노드를 재부팅할 필요가 없다는 것을 알고 있거나 패치 작업 재부팅으로 인해 중단되지 않아야 하는 노드에서 실행되는 애플리케이션 또는 프로세스가 있는 경우에 유용합니다. 유지 관리 기간을 사용하는 등 관리형 노드 재부팅 시간을 보다 정확하게 제어하려는 경우에도 유용합니다.

> **Note**
>
> NoReboot 옵션을 선택하고 패치가 설치되면 패치에 InstalledPendingReboot 상태가 할당됩니다. 그러나 관리형 노드 자체는 Non-Compliant로 표시됩니다. 재부팅이 발생하고 Scan 작업이 실행되면 관리형 노드 상태가 Compliant로 업데이트됩니다.

패치 설치 추적 파일(Patch installation tracking file): 패치 설치, 특히 마지막 시스템 재부팅 이후에 설치된 패치를 추적하기 위해 Systems Manager는 관리형 노드에서 파일을 유지 관리합니다.

> **Important**
>
> 추적 파일을 삭제하거나 수정하지 않습니다. 이 파일이 삭제되거나 손상된 경우 관리형 노드에 대한 패치 준수 보고서가 부정확하게 됩니다. 이 경우 노드를 재부팅하고 패치 스캔 작업을 실행하여 파일을 복원합니다.

이 추적 파일은 관리형 노드의 다음 위치에 저장됩니다.

- Linux 운영 체제:
  - `/var/log/amazon/ssm/patch-configuration/patch-states-configuration.json`
  - `/var/log/amazon/ssm/patch-configuration/patch-inventory-from-last-operation.json`
- Windows Server 운영 체제:
  - `C:\ProgramData\Amazon\PatchBaselineOperations\State\PatchStatesConfiguration.json`
  - `C:\ProgramData\Amazon\PatchBaselineOperations\State\PatchInventoryFromLastOperation.json`

파라미터 이름: **`BaselineOverride`**

사용 여부: 선택.

`BaselineOverride` 파라미터를 사용하여 런타임에 패치 기본 설정을 정의할 수 있습니다. 이 기준 재정의는 S3 버킷에서 JSON 객체로 유지됩니다. 이는 패치 작업이 기본 패치 기준의 규칙을 적용하는 대신 호스트 운영 체제와 일치하는 제공된 기준을 사용하도록 합니다.

`BaselineOverride` 파라미터 사용 방법에 대한 자세한 내용은 BaselineOverride 파라미터 사용 섹션을 참조하세요.

`AWS-RunPatchBaselineAssociation` SSM 문서 정보

`AWS-RunPatchBaseline` 문서와 마찬가지로 `AWS-RunPatchBaselineAssociation`은 보안 관련 업데이트 및 기타 유형의 업데이트 모두에 대해 인스턴스에서 패치 작업을 수행합니다. 문서 `AWS-RunPatchBaselineAssociation`을 사용하면 운영 체제와 애플리케이션 모두를 패치할 수도 있습

니다. (Windows Server에서 애플리케이션 지원은 Microsoft에서 릴리스한 애플리케이션의 업데이트로 제한됩니다.)

이 문서는 Linux, macOS 및 Windows Server용 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스를 지원합니다. 하이브리드 및 멀티클라우드 환경의 비 EC2 노드는 지원되지 않습니다. 이 문서는 Linux 및 macOS 인스턴스에서 Python 모듈을 호출하고 Windows 인스턴스에서 PowerShell 모듈을 호출하여 각 플랫폼에 대해 적절한 작업을 수행합니다.

그러나 `AWS-RunPatchBaselineAssociation`은 다음과 같은 점에서 `AWS-RunPatchBaseline`과 다릅니다.

- `AWS-RunPatchBaselineAssociation`은 주로 AWS Systems Manager의 기능인 Quick Setup을 사용하여 생성된 State Manager 연결에 사용하도록 만들어졌습니다. 특히 Quick Setup 호스트 관리 구성 유형을 사용하는 경우, Scan instances for missing patches daily((매일 인스턴스에서 누락된 패치 스캔) 옵션을 선택하면 작업에 `AWS-RunPatchBaselineAssociation`이 사용됩니다.

  그러나 대부분의 경우 패치 작업을 직접 설정할 때 `AWS-RunPatchBaselineAssociation` 대신 `AWS-RunPatchBaseline` 또는 `AWS-RunPatchBaselineWithHooks`를 선택해야 합니다.
- `AWS-RunPatchBaselineAssociation` 문서를 사용할 때 문서의 `BaselineTags` 파라미터 필드에 태그 키 페어를 지정할 수 있습니다. AWS 계정의 사용자 정의 패치 기준이 이러한 태그를 공유하는 경우 AWS Systems Manager의 기능인 Patch Manager는 운영 체제 유형에 대해 현재 지정된 "기본" 패치 기준 대신 대상 인스턴스에서 실행할 때 태그가 지정된 기준을 사용합니다.

  > **Important**
  >
  > Quick Setup을 사용하여 설정한 작업 이외의 패치 작업에서 `AWS-RunPatchBaselineAssociation`을 사용하도록 선택하고 옵션 `BaselineTags` 파라미터를 사용하려면 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스의 인스턴스 프로파일에 몇 가지 추가 권한을 제공해야 합니다. 자세한 내용은 파라미터 이름: `BaselineTags` 단원을 참조하십시오.

  다음 두 형식 모두 `BaselineTags` 파라미터에 대해 유효합니다.

  `Key=`*`tag-key`*`,Values=`*`tag-value`*

  `Key=`*`tag-key`*`,Values=`*`tag-value1`*`,`*`tag-value2`*`,`*`tag-value3`*
- `AWS-RunPatchBaselineAssociation` 실행 시 `AWS-RunPatchBaseline`에서 사용하는 `PutInventory` 명령 대신 `PutComplianceItems` API 명령을 사용하여 수집된 패치 규정 준수 데

이터가 기록됩니다. 이 차이는 특정 연결별로 저장 및 보고되는 패치 규정 준수 정보를 의미합니다. 이 연결 외부에서 생성된 패치 규정 준수 데이터는 덮어쓰지 않습니다.

- AWS-RunPatchBaselineAssociation 실행 후 보고되는 패치 규정 준수 정보는 인스턴스가 준수하는지 여부를 나타냅니다. 다음 AWS Command Line Interface(AWS CLI) 명령의 출력에서 알 수 있듯이 패치 수준 세부 정보는 포함되지 않습니다. 이 명령은 Association을 규정 준수 유형으로 필터링합니다.

```
aws ssm list-compliance-items \
    --resource-ids "i-02573cafcfEXAMPLE" \
    --resource-types "ManagedInstance" \
    --filters "Key=ComplianceType,Values=Association,Type=EQUAL" \
    --region us-east-2
```

시스템은 다음과 같은 정보를 반환합니다.

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

태그 키 페어 값이 AWS-RunPatchBaselineAssociation 문서의 파라미터로 지정된 경우 Patch Manager는 운영 체제 유형과 일치하고 동일한 태그 키 페어로 태그가 지정된 사용자 정의 패치 기준을 검색합니다. 이 검색은 현재 지정된 기본 패치 기준 또는 패치 그룹에 할당된 기준으로 제한되지 않습

니다. 지정된 태그에 기준이 없으면 AWS-RunPatchBaselineAssociation을 실행하는 명령에 패치 그룹이 지정된 경우 Patch Manager는 다음으로 패치 그룹을 찾습니다. 일치하는 패치 그룹이 없으면 Patch Manager는 운영 체제 계정의 현재 기본 패치 기준으로 폴백합니다.

AWS-RunPatchBaselineAssociation 문서에 지정된 태그와 함께 둘 이상의 패치 기준이 발견되면 Patch Manager는 작업을 계속하기 위해 해당 키-값 페어로 하나의 패치 기준에만 태그를 지정할 수 있음을 나타내는 오류 메시지를 반환합니다.

> **Note**
>
> Linux 인스턴스에서는 각 인스턴스 유형에 대한 적절한 패키지 관리자가 패키지를 설치하는 데 사용됩니다.
>
> - Amazon Linux 1, Amazon Linux 2, CentOS, Oracle Linux 및 RHEL 인스턴스는 YUM을 사용합니다. YUM 작업의 경우 Python 2.6 이상의 지원되는 버전(2.6~3.10)이 Patch Manager에 필요합니다.
> - Debian Server, Raspberry Pi OS 및 Ubuntu Server 인스턴스는 APT를 사용합니다. APT 작업의 경우 지원되는 Python 3 버전(3.0~3.10)이 Patch Manager에 필요합니다.
> - SUSE Linux Enterprise Server 인스턴스는 Zypper를 사용합니다. Zypper 작업의 경우 Python 2.6 이상의 지원되는 버전(2.6~3.10)이 Patch Manager에 필요합니다.

검사가 완료된 후 또는 승인되고 적용 가능한 모든 업데이트가 설치된 후 필요에 따라 재부팅이 수행되면 패치 규정 준수 정보가 인스턴스에서 생성되고 Patch Compliance 서비스에 다시 보고됩니다.

> **Note**
>
> AWS-RunPatchBaselineAssociation 문서에서 RebootOption 파라미터가 NoReboot로 설정되어 있으면 Patch Manager를 실행한 후 인스턴스가 재부팅되지 않습니다. 자세한 내용은 파라미터 이름: RebootOption 단원을 참조하십시오.

패치 규정 준수 데이터를 보는 방법에 대한 자세한 내용은 패치 규정 준수 정보 섹션을 참조하세요.

AWS-RunPatchBaselineAssociation 파라미터

AWS-RunPatchBaselineAssociation에서는 4개의 파라미터를 지원합니다. Operation 및 AssociationId 파라미터가 필요합니다. InstallOverrideList, RebootOption 및 BaselineTags 파라미터는 옵션입니다.

파라미터

- 파라미터 이름: Operation
- 파라미터 이름: BaselineTags
- 파라미터 이름: AssociationId
- 파라미터 이름: InstallOverrideList
- 파라미터 이름: RebootOption

파라미터 이름: **`Operation`**

사용 여부: 필수.

옵션: `Scan` | `Install`.

스캔

`Scan` 옵션을 선택하는 경우 `AWS-RunPatchBaselineAssociation`에 따라 인스턴스의 패치 규정 준수 상태가 결정되며 이 정보가 Patch Manager에게 다시 보고됩니다. `Scan`은 업데이트를 설치하거나 인스턴스를 재부팅하라는 메시지를 표시하지 않습니다. 대신, 승인되고 인스턴스에 적용 가능한 업데이트가 누락된 위치를 식별해 줍니다.

Install

`Install` 옵션을 선택하는 경우 `AWS-RunPatchBaselineAssociation`이 인스턴스에서 누락된 승인되고 적용 가능한 업데이트를 설치하려고 시도합니다. `Install` 작업의 일부로 생성된 패치 규정 준수 정보에는 누락된 업데이트가 나열되지 않지만, 어떠한 이유로든 업데이트 설치가 성공하지 못하면 실패 상태에 있는 업데이트를 보고할 수 있습니다. 인스턴스에 업데이트가 설치될 때마다 업데이트가 설치되었는지 및 활성 상태인지를 확인하기 위해 인스턴스가 재부팅됩니다. (예외: `AWS-RunPatchBaselineAssociation` 문서에서 `RebootOption` 파라미터가 `NoReboot`로 설정되어 있으면 Patch Manager를 실행한 후 인스턴스가 재부팅되지 않습니다. 자세한 내용은 파라미터 이름: `RebootOption` 섹션을 참조하세요.)

> **Note**
>
> Patch Manager가 인스턴스를 업데이트하기 전에 기준 규칙에 의해 지정된 패치가 설치된 경우 시스템이 예상대로 재부팅되지 않을 수도 있습니다. 이는 패치가 사용자에 의해 수동으로 설치되어 있거나 Ubuntu Server의 `unattended-upgrades` 패키지와 같은 다른 프로그램에 의해 자동으로 설치되어 있는 경우에 발생할 수 있습니다.

파라미터 이름: **BaselineTags**

사용 여부: 선택.

BaselineTags는 사용자가 선택하여 개별 사용자 정의 패치 기준에 할당하는 고유한 태그 키-값 페어입니다. 이 파라미터에 대해 값을 하나 이상 지정할 수 있습니다. 다음 형식이 모두 유효합니다.

Key=*tag-key*,Values=*tag-value*

Key=*tag-key*,Values=*tag-value1*,*tag-value2*,*tag-value3*

BaselineTags 값은 Patch Manager가 단일 작업으로 패치된 인스턴스 집합 모두가 승인된 패치 집합과 정확히 일치하는지를 확인하기 위해 사용합니다. 패치 작업이 실행될 때 Patch Manager는 운영 체제 유형에 대한 패치 기준에 사용자가 BaselineTags에 대해 지정한 동일한 키-값 페어로 태그가 지정되었는지 확인합니다. 일치 항목이 있는 경우 이 사용자 정의 패치 기준이 사용됩니다. 일치 항목이 없는 경우 패치 작업에 대해 지정된 패치 그룹에 따라 패치 기준이 식별됩니다. 아무 것도 없는 경우 해당 운영 체제에 대해 미리 정의된 AWS 관리형 패치 기준이 사용됩니다.

추가 권한 요구 사항

Quick Setup을 사용하여 설정한 작업 이외의 패치 작업에서 AWS-RunPatchBaselineAssociation을 사용하는 경우 옵션 BaselineTags 파라미터를 사용하려면 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스의 인스턴스 프로파일에 다음 권한을 제공해야 합니다.

> **Note**
>
> Quick Setup 및 AWS-RunPatchBaselineAssociation은 온프레미스 서버와 가상 머신(VM)을 지원하지 않습니다.

```
{
    "Effect": "Allow",
    "Action": [
        "ssm:DescribePatchBaselines",
        "tag:GetResources"
    ],
    "Resource": "*"
},
{
    "Effect": "Allow",
    "Action": [
```

```
        "ssm:GetPatchBaseline",
        "ssm:DescribeEffectivePatchesForPatchBaseline"
    ],
    "Resource": "patch-baseline-arn"
}
```

*patch-baseline-arn*을 arn:aws:ssm:us-east-2:123456789012:patchbaseline/pb-0c10e65780EXAMPLE 형식으로 액세스를 제공하려는 패치 기준의 Amazon 리소스 이름(ARN)으로 바꿉니다.

파라미터 이름: **AssociationId**

사용 여부: 필수.

AssociationId는 AWS Systems Manager의 기능인 State Manager에 있는 기존 연결의 ID입니다. 이 ID는 Patch Manager에서 지정된 연결에 규정 준수 데이터를 추가하는 데 사용됩니다. 이 연결은 Quick Setup에서 생성한 호스트 관리 구성에 활성화되어 있는 패치 Scan 작업과 관련이 있습니다. 패치 결과를 인벤토리 규정 준수 데이터 대신 연결 규정 준수 데이터로 전송하면 패치 작업 후 또는 다른 연결 ID에 대해 인스턴스의 기존 인벤토리 규정 준수 정보를 덮어쓰지 않습니다. 사용하려는 연결이 아직 없는 경우 create-association 명령을 실행하여 하나를 생성할 수 있습니다. 예:

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
```

파라미터 이름: **`InstallOverrideList`**

사용 여부: 선택.

`InstallOverrideList`를 사용하여 설치하려는 패치 목록에 대해 https URL 또는 Amazon Simple Storage Service(Amazon S3) 경로 스타일 URL을 지정합니다. YAML 형식으로 관리되는 이 패치 설치 목록은 현재 기본 패치 기준으로 지정된 패치를 재정의합니다. 그러면 인스턴스에 설치되는 패치를 세부적으로 제어할 수 있습니다.

`InstallOverrideList` 파라미터를 사용할 때의 패치 작업 동작은 Linux 및 macOS 관리형 노드와 Windows Server 관리형 노드가 다릅니다. Linux 및 macOS에서는 Patch Manager가 패치가 패치 기준 규칙과 일치하는지 여부와 관계없이 노드에 활성화된 모든 리포지토리의 `InstallOverrideList` 패치 목록에 포함된 패치를 적용하려고 시도합니다. 그러나 Windows Server 노드에서는 `InstallOverrideList` 패치 목록의 패치가 패치 기준 규칙과도 일치하는 경우에만 적용됩니다.

규정 준수 보고서에서는 `InstallOverrideList` 패치 목록에 지정된 항목이 아니라 패치 기준선에 지정된 항목에 따라 패치 상태를 반영합니다. 다시 말해 스캔 작업에서 `InstallOverrideList` 파라미터를 무시합니다. 그래야만 규정 준수 보고서에서 특정 패치 적용 작업에 대해 승인된 내용이 아닌 정책에 따라 패치 상태를 일관되게 반영합니다.

유효한 URL 형식

> **Note**
>
> 파일이 공개적으로 사용 가능한 버킷에 저장된 경우 https URL 형식 또는 Amazon S3 경로 스타일 URL을 지정할 수 있습니다. 파일이 프라이빗 버킷에 저장된 경우 Amazon S3 경로 스타일 URL을 지정해야 합니다.

- https URL 형식 예시:

```
https://s3.amazonaws.com/DOC-EXAMPLE-BUCKET/my-windows-override-list.yaml
```

- Amazon S3 경로 스타일 URL 예시:

```
s3://DOC-EXAMPLE-BUCKET/my-windows-override-list.yaml
```

유효한 YAML 콘텐츠 형식

목록에서 패치를 지정하는 데 사용되는 형식은 인스턴스의 운영 체제에 따라 다릅니다. 일반적인 형식은 다음과 같습니다.

```
patches:
    -
        id: '{patch-d}'
        title: '{patch-title}'
        {additional-fields}:{values}
```

YAML 파일에 추가 필드를 제공할 수 있지만 해당 필드는 패치 작업 중에 무시됩니다.

또한 S3 버킷에서 목록을 추가하거나 업데이트하기 전에 YAML 파일 형식이 올바른지 확인하는 것이 좋습니다. YAML 형식에 대한 자세한 내용은 yaml.org를 참조하십시오. 유효한 도구 옵션을 보려면 웹에서 "yaml format validators"를 검색하십시오.

- Microsoft Windows

  id

  id 필드는 필수입니다. id 필드는 Microsoft Knowledge Base ID(예: KB2736693)와 Microsoft Security Bulletin ID(예: MS17-023)를 사용하여 패치를 지정하는 데 사용됩니다.

  Windows에 대한 패치 목록에 제공하려는 다른 필드는 선택 사항이며 정보 제공의 목적으로만 사용됩니다. title, classification, severity 등과 같은 추가 필드를 사용하여 지정된 패치에 대한 자세한 정보를 제공할 수 있습니다.

- Linux

  id

  id 필드는 필수입니다. 패키지 이름 및 아키텍처를 사용하여 패치를 지정하는 데 사용됩니다. 예: `'dhclient.x86_64'`. id에서 와일드카드를 사용하여 여러 패키지를 나타낼 수 있습니다. 예를 들면 `'dhcp*'` 및 `'dhcp*1.*'` 등입니다.

  title

제목 필드는 선택 사항이지만 Linux 시스템의 경우 이 필드는 추가 필터링 기능을 제공합니다. title을 사용할 경우 패키지 버전 정보를 다음 중 한 가지 형식으로 포함해야 합니다.

YUM/SUSE Linux Enterprise Server(SLES):

```
{name}.{architecture}:{epoch}:{version}-{release}
```

APT

```
{name}.{architecture}:{version}
```

Linux 패치 제목의 경우 임의의 위치에서 하나 이상의 와일드카드를 사용하여 패키지 일치 수를 확장할 수 있습니다. 예: `'*32:9.8.2-0.*.rc1.57.amzn1'`.

예:

- apt 패키지 버전 1.2.25가 인스턴스에 현재 설치되어 있지만 버전 1.2.27을 지금 사용할 수 있습니다.
- apt.amd64 버전 1.2.27을 패치 목록에 추가합니다. apt utils.amd64 버전 1.2.27을 사용하지만 apt-utils.amd64 버전 1.2.25가 목록에 지정되어 있습니다.

이 경우 apt 버전 1.2.27은 설치 시 차단되고 “Failed-NonCompliant”로 보고됩니다.

기타 필드

Linux에 대한 패치 목록에 제공하려는 다른 필드는 선택 사항이며 정보 제공의 목적으로만 사용됩니다. classification, severity 등과 같은 추가 필드를 사용하여 지정된 패치에 대한 자세한 정보를 제공할 수 있습니다.

샘플 패치 목록

- 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'
```

파라미터 이름: **RebootOption**

사용 여부: 선택.

옵션: RebootIfNeeded | NoReboot

기본값: RebootIfNeeded

> **Warning**
>
> 기본 옵션은 RebootIfNeeded입니다. 사용 사례에 맞는 올바른 옵션을 선택해야 합니다. 예를 들어 구성 프로세스를 완료하기 위해 인스턴스를 즉시 재부팅해야 하는 경우, RebootIfNeeded를 선택합니다. 또는 예약된 재부팅 시간까지 인스턴스 가용성을 유지해야 하는 경우, NoReboot를 선택합니다.

> **Important**
>
> Amazon EMR(이전 명칭: Amazon Elastic MapReduce)에서 클러스터 인스턴스 패치를 적용하는 데는 Patch Manager를 사용하지 않는 것이 좋습니다. 특히 RebootOption 파라미터에 RebootIfNeeded 옵션을 선택하지 마세요. (이 옵션은 AWS-RunPatchBaseline, AWS-RunPatchBaselineAssociation 및 AWS-RunPatchBaselineWithHooks 패치 적용에 대한 SSM 명령 문서에서 제공되지 않습니다.)
> Patch Manager를 사용하는 패치 적용에 대한 기본 명령에서는 yum 및 dnf 명령을 사용합니다. 따라서 패키지 설치 방식 때문에 작업이 호환되지 않을 수 있습니다. Amazon EMR 클러스터의 소프트웨어 업데이트용으로 기본 설정된 방법에 대한 자세한 내용은 Amazon EMR 관리 안내서의 Amazon EMR용 기본 AMI 사용을 참조하세요.

RebootIfNeeded

RebootIfNeeded 옵션을 선택하면 다음과 같은 경우 인스턴스가 재부팅됩니다.

- Patch Manager가 하나 이상의 패치를 설치했습니다.

  Patch Manager가 패치에 재부팅이 필요한지 여부를 평가하지 않습니다. 패치에 재부팅이 필요하지 않은 경우에도 시스템이 재부팅됩니다.

- Patch Manager가 Install 작업 중 INSTALLED_PENDING_REBOOT 상태의 패치를 하나 이상 감지합니다.

  INSTALLED_PENDING_REBOOT 상태는 Install 작업이 마지막으로 실행될 때 NoReboot 옵션이 선택되었거나 관리형 노드를 마지막으로 재부팅한 이후에 Patch Manager 외부에 패치가 설치되었다는 의미일 수 있습니다.

이 두 가지 경우에 인스턴스를 재부팅하면 업데이트된 패키지가 메모리에서 플러시되고 모든 운영 체제에서 패치 및 재부팅 동작이 일관되게 유지됩니다.

NoReboot

NoReboot 옵션을 선택하면 Patch Manager가 Install 작업 중에 패치를 설치한 경우에도 인스턴스를 재부팅하지 않습니다. 이 옵션은 패치된 후 인스턴스를 재부팅할 필요가 없다는 것을 알고 있거나 패치 작업 재부팅으로 인해 중단되지 않아야 하는 인스턴스에서 실행되는 애플리케이션 또는 프로세스가 있는 경우에 유용합니다. 유지 관리 기간을 사용하는 등 인스턴스 재부팅 시간을 보다 정확하게 제어하려는 경우에도 유용합니다.

[패치 설치 추적 파일(Patch installation tracking file)]: 패치 설치, 특히 마지막 시스템 재부팅 이후에 설치된 패치를 추적하기 위해 Systems Manager는 관리형 인스턴스에서 파일을 유지 관리합니다.

> **Important**
>
> 추적 파일을 삭제하거나 수정하지 않습니다. 이 파일이 삭제되거나 손상된 경우 인스턴스에 대한 패치 준수 보고서가 부정확하게 됩니다. 이 경우 인스턴스를 재부팅하고 패치 스캔 작업을 실행하여 파일을 복원합니다.

이 추적 파일은 관리형 인스턴스의 다음 위치에 저장됩니다.

- Linux 운영 체제:
  - `/var/log/amazon/ssm/patch-configuration/patch-states-configuration.json`
  - `/var/log/amazon/ssm/patch-configuration/patch-inventory-from-last-operation.json`
- Windows Server 운영 체제:
  - `C:\ProgramData\Amazon\PatchBaselineOperations\State\PatchStatesConfiguration.json`
  - `C:\ProgramData\Amazon\PatchBaselineOperations\State\PatchInventoryFromLastOperation.json`

AWS-RunPatchBaselineWithHooks SSM 문서 정보

AWS Systems Manager는 AWS Systems Manager의 기능인 Patch Manager용 Systems Manager 문서(SSM 문서)인 AWS-RunPatchBaselineWithHooks을 지원합니다. 이 SSM 문서는 보안 관련 업데이트 및 기타 유형의 업데이트 모두에 대해 관리형 노드에서 패치 작업을 수행합니다.

AWS-RunPatchBaselineWithHooks는 다음과 같은 점에서 AWS-RunPatchBaseline과 다릅니다.

- 래퍼 문서 - AWS-RunPatchBaselineWithHooks는 AWS-RunPatchBaseline의 래퍼이며 일부 작업에 대해 AWS-RunPatchBaseline에 의존합니다.
- **Install** 작업 – AWS-RunPatchBaselineWithHooks은 관리형 노드 패치 중 지정된 지점에서 실행되는 수명 주기 후크를 지원합니다. 패치 설치 시 관리형 노드를 재부팅해야 하는 경우가 있기 때문에 패치 작업은 사용자 정의 기능을 지원하는 총 3개의 후크에 대한 2개의 이벤트로 나뉩니다. 첫 번째 후크는 Install with NoReboot 작업 전입니다. 두 번째 후크는 Install with NoReboot 작업 후입니다. 세 번째 후크는 관리형 노드 재부팅 후 사용할 수 있습니다.
- 사용자 정의 패치 목록 지원 없음 - AWS-RunPatchBaselineWithHooks가 InstallOverrideList 파라미터를 지원하지 않습니다.
- SSM Agent 지원 - AWS-RunPatchBaselineWithHooks이 SSM Agent를 사용하려면 패치할 관리형 노드에 3.0.502 이상 버전이 설치되어 있어야 합니다.

문서가 실행될 때 패치 그룹이 지정되지 않은 경우 운영 체제 유형에 대해 "기본값"으로 현재 지정된 패치 기준을 사용합니다 그렇지 않으면 패치 그룹과 연결된 패치 기준을 사용합니다. 패치 그룹에 대한 자세한 내용은 패치 그룹 정보 섹션을 참조하세요.

문서 AWS-RunPatchBaselineWithHooks을 사용하면 운영 체제와 애플리케이션 모두를 패치할 수 있습니다. (Windows에서 애플리케이션 지원은 Microsoft에서 릴리스한 애플리케이션의 업데이트로 제한됩니다.)

이 문서는 Linux, macOS, 및 Windows Server 관리형 노드를 지원합니다. 문서에서 각 플랫폼에 적합한 작업을 수행합니다.

Linux

Linux 관리형 노드에서는 AWS-RunPatchBaselineWithHooks 문서가 Python 모듈을 호출합니다. 그런 다음, 해당 관리형 노드에 적용되는 패치 기준의 스냅샷을 다운로드합니다. 이 패치 기준 스냅샷은 정의된 규칙과 승인 및 차단된 패치 목록을 사용하여 각 노드 유형에 적합한 패키지 관리자를 실행합니다.

- Amazon Linux 1, Amazon Linux 2, CentOS, Oracle Linux, RHEL 7 관리형 노드는 YUM을 사용합니다. YUM 작업의 경우 Python 2.6 이상의 지원되는 버전(2.6~3.10)이 Patch Manager에 필요합니다.

- RHEL 8 관리형 노드는 DNF를 사용합니다. DNF 작업의 경우 지원되는 `Python 2` 또는 `Python 3` 버전(2.6~3.10)이 Patch Manager에 필요합니다. (두 버전 모두 RHEL 8에 기본적으로 설치되어 있지 않습니다. 둘 중 하나를 수동으로 설치해야 합니다.)
- Debian Server, Raspberry Pi OS 및 Ubuntu Server 인스턴스는 APT를 사용합니다. APT 작업의 경우 지원되는 `Python 3` 버전(3.0~3.10)이 Patch Manager에 필요합니다.
- SUSE Linux Enterprise Server 관리형 노드는 Zypper를 사용합니다. Zypper 작업의 경우 `Python 2.6` 이상의 지원되는 버전(2.6~3.10)이 Patch Manager에 필요합니다.

macOS

macOS 관리형 노드에서는 `AWS-RunPatchBaselineWithHooks` 문서가 Python 모듈을 호출합니다. 그런 다음, 해당 인스턴스에 적용되는 패치 기준의 스냅샷을 다운로드합니다. 다음으로 Python 하위 프로세스는 노드에서 CLI를 호출하여 지정된 패키지 관리자에 대한 설치 및 업데이트 정보를 검색하고 각 업데이트 패키지에 대해 적절한 패키지 관리자를 구동합니다.

Windows Server

Windows Server 관리형 노드에서는 `AWS-RunPatchBaselineWithHooks` 문서가 PowerShell 모듈을 다운로드하고 호출합니다. 그런 다음, 해당 관리형 노드에 적용되는 패치 기준의 스냅샷을 다운로드합니다. 이 패치 기준 스냅샷에는 Windows Server Update Services(WSUS) 서버에 대해 패치 기준을 쿼리하여 컴파일된 승인된 패치 목록이 포함되어 있습니다. 이 목록은 Windows Update API에 전달되며, 이 API가 승인된 패치를 적절하게 다운로드하고 설치하는 작업을 제어합니다.

각 스냅샷은 AWS 계정, 패치 그룹, 운영 체제 및 스냅샷 ID에 따라 다릅니다. 스냅샷은 스냅샷 생성 후 24시간이 지나면 만료되는 미리 서명된 Amazon Simple Storage Service(Amazon S3) URL을 통해 전송됩니다. 그러나 URL이 만료된 후 동일한 스냅샷 콘텐츠를 다른 관리형 노드에 적용하려는 경우 스냅샷이 생성된 후 최대 3일 이내에 미리 서명된 Amazon S3 URL을 새로 생성할 수 있습니다. 이렇게 하려면 get-deployable-patch-snapshot-for-instance 명령을 사용합니다.

승인되고 적용 가능한 모든 업데이트가 설치되고 필요한 만큼 재부팅이 수행되면, 패치 규정 준수 정보가 관리형 노드에 생성되며 Patch Manager에 다시 보고됩니다.

> **Note**
>
> `AWS-RunPatchBaselineWithHooks` 문서에서 `RebootOption` 파라미터가 `NoReboot`로 설정되어 있으면 Patch Manager를 실행한 후 관리형 노드가 재부팅되지 않습니다. 자세한 내용은 파라미터 이름: `RebootOption` 단원을 참조하십시오.

패치 규정 준수 데이터를 보는 방법에 대한 자세한 내용은 패치 규정 준수 정보 섹션을 참조하세요.

AWS-RunPatchBaselineWithHooks 운영 단계

AWS-RunPatchBaselineWithHooks가 실행되면 다음 단계가 수행됩니다.

1. 스캔 - AWS-RunPatchBaseline을 사용하는 Scan 작업이 관리형 노드에서 실행되고 규정 준수 보고서가 생성되고 업로드됩니다.
2. 로컬 패치 상태 확인 - 선택한 작업과 1단계의 Scan 결과를 기반으로 수행할 단계를 결정하기 위해 스크립트가 실행됩니다.
   a. 선택한 작업이 Scan이면 작업이 완료된 것으로 표시됩니다. 작업이 종료됩니다.
   b. 선택한 작업이 Install이면 Patch Manager는 1단계의 Scan 결과를 평가하여 다음에 실행할 작업을 결정합니다.
      i. 누락된 패치가 발견되지 않고 보류 중인 재부팅이 필요하지 않으면 사용자가 제공한 후크가 포함된 최종 단계(8단계)로 작업이 바로 진행됩니다. 그 사이의 모든 단계는 건너뜁니다.
      ii. 누락된 패치가 발견되지 않았지만 보류 중인 재부팅이 필요하고 선택한 재부팅 옵션이NoReboot이면 사용자가 제공한 후크가 포함된 최종 단계(8단계)로 작업이 바로 진행됩니다. 그 사이의 모든 단계는 건너뜁니다.
      iii. 그렇지 않으면 작업은 다음 단계로 진행됩니다.
3. 패치 전 후크 작업 - 첫 번째 수명 주기 후크인 PreInstallHookDocName에 대해 제공한 SSM 문서가 관리형 노드에서 실행됩니다.
4. NoReboot로 설치 - 재부팅 옵션이 NoReboot이고 AWS-RunPatchBaseline을 사용하는 Install 작업이 관리형 노드에서 실행되고 규정 준수 보고서가 생성되어 업로드됩니다.
5. 설치 후 후크 작업 - 두 번째 수명 주기 후크인 PostInstallHookDocName에 대해 제공한 SSM 문서가 관리형 노드에서 실행됩니다.
6. 재부팅 확인 - 관리형 노드에 재부팅이 필요한지 여부와 실행할 단계를 결정하기 위해 스크립트가 실행됩니다.
   a. 선택한 재부팅 옵션이 NoReboot인 경우 사용자가 제공한 후크가 포함된 최종 단계(8단계)로 작업이 바로 진행됩니다. 그 사이의 모든 단계는 건너뜁니다.
   b. 선택한 재부팅 옵션이 RebootIfNeeded인 경우 Patch Manager는 4단계에서 수집된 인벤토리에서 필요한 보류 중인 재부팅이 있는지 확인합니다. 즉, 작업이 7단계로 계속 진행되고 다음과 같은 경우 관리형 노드가 재부팅됩니다.
      i. Patch Manager가 하나 이상의 패치를 설치한 경우(Patch Manager가 패치에 재부팅이 필요한지 여부를 평가하지 않습니다. 패치에 재부팅이 필요하지 않은 경우에도 시스템이 재부팅됩니다.)

ii. Patch Manager가 설치 작업 중 INSTALLED_PENDING_REBOOT 상태의 패치를 하나 이상 감지합니다. INSTALLED_PENDING_REBOOT 상태는 설치 작업이 마지막으로 실행될 때 NoReboot 옵션이 선택되었거나 관리형 노드를 마지막으로 재부팅한 이후에 Patch Manager 외부에 패치가 설치되었다는 의미일 수 있습니다.

이러한 기준을 충족하는 패치가 없으면 관리형 노드 패치 작업이 완료되고 작업은 사용자가 제공한 후크가 포함된 최종 단계(8단계)로 바로 진행됩니다. 그 사이의 모든 단계는 건너뜁니다.

7. 재부팅 및 보고 - 재부팅 옵션이 RebootIfNeeded인 설치 작업이 AWS-RunPatchBaseline을 사용하여 관리형 노드에서 실행되고 규정 준수 보고서가 생성되어 업로드됩니다.
8. 재부팅 후 후크 작업 - 세 번째 수명 주기 후크인 OnExitHookDocName에 대해 제공한 SSM 문서가 관리형 노드에서 실행됩니다.

Scan 작업의 경우 1단계가 실패하면 문서 실행 프로세스가 중지되고 단계가 실패로 보고되지만 후속 단계는 성공으로 보고됩니다.

Install 작업의 경우 작업 중 aws:runDocument 단계 중 하나라도 실패하면 해당 단계는 실패로 보고되고 작업은 사용자가 제공한 후크를 포함하는 최종 단계(8단계)로 바로 진행됩니다. 그 사이의 모든 단계는 건너뜁니다. 이 단계는 실패로 보고되고, 마지막 단계는 작업 결과의 상태를 보고하며, 그 사이의 모든 단계는 성공으로 보고됩니다.

AWS-RunPatchBaselineWithHooks 파라미터

AWS-RunPatchBaselineWithHooks는 6개 파라미터를 지원합니다.

Operation 파라미터가 필요합니다.

RebootOption, PreInstallHookDocName, PostInstallHookDocName 및 OnExitHookDocName 파라미터는 옵션입니다.

Snapshot-ID는 기술적으로 옵션이지만 유지 관리 기간 외에 AWS-RunPatchBaselineWithHooks를 실행할 때 사용자 정의 값을 제공하는 것이 좋습니다. 유지 관리 기간 작업의 일부로 문서가 실행될 때 Patch Manager가 자동으로 값을 제공하도록 합니다.

파라미터

- 파라미터 이름: Operation
- 파라미터 이름: Snapshot ID
- 파라미터 이름: RebootOption

- 파라미터 이름: PreInstallHookDocName
- 파라미터 이름: PostInstallHookDocName
- 파라미터 이름: OnExitHookDocName

파라미터 이름: **`Operation`**

사용 여부: 필수.

옵션: `Scan` | `Install`.

스캔

`Scan` 옵션을 선택하는 경우 시스템이 `AWS-RunPatchBaseline` 문서를 사용하여 따라 관리형 노드의 패치 규정 준수 상태를 결정하며 이 정보가 Patch Manager에게 다시 보고됩니다. `Scan`은 업데이트를 설치하거나 관리형 노드를 재부팅하라는 메시지를 표시하지 않습니다. 대신, 작업이 승인되고 노드에 적용 가능한 업데이트가 누락된 위치를 식별해 줍니다.

Install

`Install` 옵션을 선택하는 경우 `AWS-RunPatchBaselineWithHooks`이 관리형 노드에서 누락된 승인되고 적용 가능한 업데이트를 설치하려고 시도합니다. `Install` 작업의 일부로 생성된 패치 규정 준수 정보에는 누락된 업데이트가 나열되지 않지만, 어떠한 이유로든 업데이트 설치가 성공하지 못하면 실패 상태에 있는 업데이트를 보고할 수 있습니다. 관리형 노드에 업데이트가 설치될 때마다 업데이트가 설치되었는지 및 활성 상태인지를 확인하기 위해 노드가 재부팅됩니다. (예외: `AWS-RunPatchBaselineWithHooks` 문서에서 `RebootOption` 파라미터가 `NoReboot`로 설정되어 있으면 Patch Manager를 실행한 후 관리형 노드가 재부팅되지 않습니다. 자세한 내용은 파라미터 이름: `RebootOption` 섹션을 참조하세요.)

> **Note**
>
> Patch Manager가 관리형 노드를 업데이트하기 전에 기준 규칙에 의해 지정된 패치가 설치된 경우 시스템이 예상대로 재부팅되지 않을 수도 있습니다. 이는 패치가 사용자에 의해 수동으로 설치되어 있거나 Ubuntu Server의 `unattended-upgrades` 패키지와 같은 다른 프로그램에 의해 자동으로 설치되어 있는 경우에 발생할 수 있습니다.

파라미터 이름: **`Snapshot ID`**

사용 여부: 선택.

Snapshot ID는 Patch Manager가 단일 작업으로 패치된 관리형 노드 집합 모두가 승인된 패치 집합과 정확히 일치하는지를 확인하기 위해 사용하는 고유 ID(GUID)입니다. 이 파라미터가 옵션으로 정의되어 있긴 하지만, 다음 표에 설명된 대로 유지 관리 기간에서 AWS-RunPatchBaselineWithHooks을 실행하는지 여부에 따라 모범 사례 권장 사항이 달라집니다.

AWS-RunPatchBaselineWithHooks 모범 사례

Mode	모범 사례	Details
유지 관리 기간 내 AWS-RunPatchBaselineWithHooks 실행	스냅샷 ID를 제공하지 않습니다. Patch Manager가 제공합니다.	유지 관리 기간을 사용하여 AWS-RunPatchBaselineWithHooks 을 실행하는 경우 자체 생성한 스냅샷 ID를 제공해선 안 됩니다. 이러한 경우에는 Systems Manager가 유지 관리 기간 실행 ID를 기반으로 GUID 값을 제공합니다. 이렇게 하면 해당 유지 관리 기간에 모든 AWS-RunPatchBaselineWithHooks 호출에 올바른 ID가 사용됩니다. 이러한 경우에 값을 지정하는 경우 패치 기준선의 스냅샷이 3일 이상 적용되지 않을 수 있습니다. 해당 시간 경과 후에는 스냅샷 만료 후 동일한 ID를 지정하더라도 새로운 스냅샷이 생성됩니다.
유지 관리 기간 외 AWS-RunPatchBaselineWithHooks 실행	스냅샷 ID에 대해 사용자 정의 GUID 값을 생성하고 지정합니다.[1]	AWS-RunPatchBaselineWithHooks 을 실행하는 데 유지 관리 기간을 사용하고 있지 않은 경우 각 패치 기준에 대해 고유한 스냅샷 ID를 생성하고 지정하는 것이 좋습니다. 특히 동일한 작업에서 여

Mode	모범 사례	Details
		러 관리형 노드에 AWS-RunPatchBaselineWithHooks 문서를 실행하고 있는 경우 더욱 그러합니다. 이러한 경우에 ID를 지정하지 않으면 Systems Manager가 명령을 보내는 각 관리형 노드에 다른 스냅샷 ID를 생성합니다. 이렇게 되면 노드 간에 다양한 패치 집합이 지정될 수 있습니다. 예를 들어 AWS Systems Manager의 기능인 Run Command를 통해 직접 AWS-RunPatchBaselineWithHooks 문서를 실행하고 50개의관리형 노드로 이루어진 그룹을 대상으로 한다고 가정해 보겠습니다. 사용자 지정 스냅샷 ID를 지정하면 모든 관리형 노드를 평가하고 패치를 적용하는 데 사용되는 기준 스냅샷이 하나가 생성되어 일관적인 상태를 유지하도록 해줍니다.

[1] GUID를 생성할 수 있는 도구라면 어떤 도구를 사용해서든 스냅샷 ID 파라미터의 값을 생성할 수 있습니다. 예를 들어 PowerShell의 경우 New-Guid cmdlet을 사용하여 12345699-9405-4f69-bc5e-9315aEXAMPLE 형식으로 GUID를 생성할 수 있습니다.

파라미터 이름: **RebootOption**

사용 여부: 선택.

옵션: RebootIfNeeded | NoReboot

기본값: RebootIfNeeded

> **Warning**
>
> 기본 옵션은 RebootIfNeeded입니다. 사용 사례에 맞는 올바른 옵션을 선택해야 합니다. 예를 들어 구성 프로세스를 완료하기 위해 관리형 노드를 즉시 재부팅해야 하는 경우, RebootIfNeeded를 선택합니다. 또는 예약된 재부팅 시간까지 관리형 노드 가용성을 유지해야 하는 경우, NoReboot를 선택합니다.

> **Important**
>
> Amazon EMR(이전 명칭: Amazon Elastic MapReduce)에서 클러스터 인스턴스 패치를 적용하는 데는 Patch Manager를 사용하지 않는 것이 좋습니다. 특히 RebootOption 파라미터에 RebootIfNeeded 옵션을 선택하지 마세요. (이 옵션은 AWS-RunPatchBaseline, AWS-RunPatchBaselineAssociation 및 AWS-RunPatchBaselineWithHooks 패치 적용에 대한 SSM 명령 문서에서 제공되지 않습니다.)
> Patch Manager를 사용하는 패치 적용에 대한 기본 명령에서는 yum 및 dnf 명령을 사용합니다. 따라서 패키지 설치 방식 때문에 작업이 호환되지 않을 수 있습니다. Amazon EMR 클러스터의 소프트웨어 업데이트용으로 기본 설정된 방법에 대한 자세한 내용은 Amazon EMR 관리 안내서의 Amazon EMR용 기본 AMI 사용을 참조하세요.

RebootIfNeeded

RebootIfNeeded 옵션을 선택하면 다음과 같은 경우 관리형 노드가 재부팅됩니다.

- Patch Manager가 하나 이상의 패치를 설치했습니다.

  Patch Manager가 패치에 재부팅이 필요한지 여부를 평가하지 않습니다. 패치에 재부팅이 필요하지 않은 경우에도 시스템이 재부팅됩니다.

- Patch Manager가 Install 작업 중 INSTALLED_PENDING_REBOOT 상태의 패치를 하나 이상 감지합니다.

  INSTALLED_PENDING_REBOOT 상태는 Install 작업이 마지막으로 실행될 때 NoReboot 옵션이 선택되었거나 관리형 노드를 마지막으로 재부팅한 이후에 Patch Manager 외부에 패치가 설치되었다는 의미일 수 있습니다.

이 두 가지 경우에 관리형 노드를 재부팅하면 업데이트된 패키지가 메모리에서 플러시되고 모든 운영 체제에서 패치 및 재부팅 동작이 일관되게 유지됩니다.

NoReboot

NoReboot 옵션을 선택하면 Patch Manager가 Install 작업 중에 패치를 설치한 경우에도 관리형 노드를 재부팅하지 않습니다. 이 옵션은 패치된 후 관리형 노드를 재부팅할 필요가 없다는 것을 알고 있거나 패치 작업 재부팅으로 인해 중단되지 않아야 하는 노드에서 실행되는 애플리케이션 또는 프로세스가 있는 경우에 유용합니다. 유지 관리 기간을 사용하는 등 관리형 노드 재부팅 시간을 보다 정확하게 제어하려는 경우에도 유용합니다.

> **Note**
>
> NoReboot 옵션을 선택하고 패치가 설치되면 패치에 InstalledPendingReboot 상태가 할당됩니다. 그러나 관리형 노드 자체는 Non-Compliant로 표시됩니다. 재부팅이 발생하고 Scan 작업이 실행되면 노드 상태가 Compliant로 업데이트됩니다.

패치 설치 추적 파일(Patch installation tracking file): 패치 설치, 특히 마지막 시스템 재부팅 이후에 설치된 패치를 추적하기 위해 Systems Manager는 관리형 노드에서 파일을 유지 관리합니다.

> **Important**
>
> 추적 파일을 삭제하거나 수정하지 않습니다. 이 파일이 삭제되거나 손상된 경우 관리형 노드에 대한 패치 준수 보고서가 부정확하게 됩니다. 이 경우 노드를 재부팅하고 패치 스캔 작업을 실행하여 파일을 복원합니다.

이 추적 파일은 관리형 노드의 다음 위치에 저장됩니다.

- Linux 운영 체제:
  - /var/log/amazon/ssm/patch-configuration/patch-states-configuration.json
  - /var/log/amazon/ssm/patch-configuration/patch-inventory-from-last-operation.json
- Windows Server 운영 체제:
  - C:\ProgramData\Amazon\PatchBaselineOperations\State\PatchStatesConfiguration.json

- C:\ProgramData\Amazon\PatchBaselineOperations\State\PatchInventoryFromLastOperation.json

파라미터 이름: **PreInstallHookDocName**

사용 여부: 선택.

기본값: AWS-Noop.

PreInstallHookDocName 파라미터에 제공할 값은 선택한 SSM 문서의 이름 또는 Amazon 리소스 이름(ARN)입니다. AWS 관리형 문서의 이름이나 생성했거나 공유한 사용자 정의 SSM 문서의 이름 또는 ARN을 제공할 수 있습니다. (다른 AWS 계정에서 공유한 SSM 문서의 경우 arn:aws:ssm:us-east-2:123456789012:document/MySharedDocument과 같이 전체 리소스 ARN을 지정해야 합니다.)

지정한 SSM 문서는 Install 작업 전에 실행되며 관리형 노드에서 패치가 수행되기 전에 애플리케이션 상태 확인을 점검하는 셸 스크립트와 같이 SSM Agent에서 지원하는 모든 작업을 수행합니다. 작업 목록은 Command 문서 플러그인 참조 섹션을 참조하세요. 기본 SSM 문서 이름은 AWS-Noop이며 관리형 노드에서 작업을 수행하지 않습니다.

사용자 정의 SSM 문서 생성에 대한 자세한 내용은 SSM 문서 콘텐츠 생성 섹션을 참조하세요.

파라미터 이름: **PostInstallHookDocName**

사용 여부: 선택.

기본값: AWS-Noop.

PostInstallHookDocName 파라미터에 제공할 값은 선택한 SSM 문서의 이름 또는 Amazon 리소스 이름(ARN)입니다. AWS 관리형 문서의 이름이나 생성했거나 공유한 사용자 정의 SSM 문서의 이름 또는 ARN을 제공할 수 있습니다. (다른 AWS 계정에서 공유한 SSM 문서의 경우 arn:aws:ssm:us-east-2:123456789012:document/MySharedDocument과 같이 전체 리소스 ARN을 지정해야 합니다.)

지정한 SSM 문서는 Install with NoReboot 작업 후에 실행되며 재부팅 전에 서드 파티 업데이트를 설치하기 위한 셸 스크립트와 같이 SSM Agent가 지원하는 모든 작업을 수행합니다. 작업 목록은 Command 문서 플러그인 참조 섹션을 참조하세요. 기본 SSM 문서 이름은 AWS-Noop이며 관리형 노드에서 작업을 수행하지 않습니다.

사용자 정의 SSM 문서 생성에 대한 자세한 내용은 SSM 문서 콘텐츠 생성 섹션을 참조하세요.

파라미터 이름: **`OnExitHookDocName`**

사용 여부: 선택.

기본값: `AWS-Noop`.

`OnExitHookDocName` 파라미터에 제공할 값은 선택한 SSM 문서의 이름 또는 Amazon 리소스 이름(ARN)입니다. AWS 관리형 문서의 이름이나 생성했거나 공유한 사용자 정의 SSM 문서의 이름 또는 ARN을 제공할 수 있습니다. (다른 AWS 계정에서 공유한 SSM 문서의 경우 `arn:aws:ssm:us-east-2:123456789012:document/MySharedDocument`과 같이 전체 리소스 ARN을 지정해야 합니다.)

지정한 SSM 문서는 관리형 노드 재부팅 작업 후에 실행되며 패치 작업이 완료된 후 노드 상태를 확인하기 위한 셸 스크립트와 같이 SSM Agent에서 지원하는 모든 작업을 수행합니다. 작업 목록은 Command 문서 플러그인 참조 섹션을 참조하세요. 기본 SSM 문서 이름은 `AWS-Noop`이며 관리형 노드에서 작업을 수행하지 않습니다.

사용자 정의 SSM 문서 생성에 대한 자세한 내용은 SSM 문서 콘텐츠 생성 섹션을 참조하세요.

`AWS-RunPatchBaseline` 또는 **`AWS-RunPatchBaselineAssociation`**에 InstallOverrideList 파라미터를 사용하기 위한 샘플 시나리오

AWS Systems Manager의 기능인 Patch Manager에 현재 기본 패치 기준선으로 지정된 패치를 재정의하려는 경우 `InstallOverrideList` 파라미터를 사용할 수 있습니다. 이 주제에서는 이 파라미터를 사용하여 다음을 수행하는 방법을 보여주는 예를 제공합니다.

- 대상 관리형 노드 그룹에 서로 다른 패치 세트를 적용합니다.
- 이러한 패치 세트를 다른 빈도에 적용합니다.
- 두 작업 모두에 동일한 패치 기준선을 사용합니다.

Amazon Linux 2 관리형 노드에 서로 다른 두 가지 범주의 패치를 설치한다고 가정해 보겠습니다. 유지 관리 기간을 사용하여 이러한 패치를 다른 일정에 설치하려고 합니다. 매주 하나의 유지 관리 기간을 실행하고 모든 `Security` 패치를 설치하려고 합니다. 한 달에 한 번 다른 유지 관리 기간을 실행하고 사용 가능한 모든 패치 또는 `Security` 이외 범주의 패치를 설치하려고 합니다.

그러나 한 번에 하나의 패치 기준선만 운영 체제의 기본값으로 정의할 수 있습니다. 이 요구 사항은 한 패치 기준이 패치를 승인하는 반면 다른 패치 기준선이 패치를 차단하여, 충돌하는 버전 간에 문제가 발생할 수 있는 상황을 피하는 데 도움이 됩니다.

다음 전략으로 InstallOverrideList 파라미터를 사용하여 동일한 패치 기준선을 계속 사용하면서 서로 다른 일정에 따라 대상 그룹에 서로 다른 유형의 패치를 적용할 수 있습니다.

1. 기본 패치 기준선에서 Security 업데이트만 지정되었는지 확인합니다.
2. 매주 AWS-RunPatchBaseline 또는 AWS-RunPatchBaselineAssociation이 실행되는 유지 관리 기간을 생성합니다. 재정의 목록을 지정하지 않습니다.
3. 매월 적용하려는 모든 유형의 패치에 대한 재정의 목록을 생성하여 Amazon Simple Storage Service(Amazon S3) 버킷에 저장합니다.
4. 한 달에 한 번 실행되는 두 번째 유지 관리 기간을 생성합니다. 그러나 이 유지 관리 기간에 등록한 Run Command 태스크의 경우 재정의 목록의 위치를 지정합니다.

결과: 기본 패치 기준선에 정의된 Security 패치만 매주 설치됩니다. 사용 가능한 모든 패치 또는 정의한 패치의 하위 집합은 매월 설치됩니다.

자세한 내용과 샘플 목록은 파라미터 이름: InstallOverrideList 섹션을 참조하세요.

BaselineOverride 파라미터 사용

AWS Systems Manager의 기능인 Patch Manager의 기준 재정의 기능을 사용하여 런타임 시 패치 기본 설정을 정의할 수 있습니다. 이렇게 하려면 패치 기준 목록과 함께 JSON 객체를 포함하는 Amazon Simple Storage Service(Amazon S3) 버킷을 지정합니다. 패치 작업은 기본 패치 기준의 규칙을 적용하는 대신 호스트 운영 체제와 일치하는 JSON 객체에 제공된 기준을 사용합니다.

> **Note**
>
> 패치 작업이 패치 정책을 사용하는 경우를 제외하고 BaselineOverride 파라미터를 사용해도 파라미터에 제공된 기준선의 패치 규정 준수를 덮어쓰지 않습니다. 출력 결과는 AWS Systems Manager의 기능인 Run Command의 Stdout 로그에 기록됩니다. 결과는 NON_COMPLIANT로 표시된 패키지만 인쇄합니다. 이는 패키지가 Missing, Failed, InstalledRejected 또는 InstalledPendingReboot로 표시되었음을 의미합니다. 그러나 패치 작업이 패치 정책을 사용하는 경우 시스템은 관련 S3 버킷의 재정의 파라미터를 전달하고 관리형 노드에 대한 규정 준수 값이 업데이트됩니다. 패치 정책 동작에 대한 자세한 내용은 Quick Setup 패치 정책 사용 섹션을 참조하세요.

스냅샷 ID 또는 설치 재정의 목록 파라미터에 패치 기준 재정의 사용

패치 기준 재정의에 주목할 만한 동작이 있는 두 가지 경우가 있습니다.

기준 재정의 및 스냅샷 ID 동시 사용

스냅샷 ID는 특정 패치 명령의 모든 관리형 노드가 모두 동일한 것을 적용하도록 합니다. 예를 들어 한 번에 1,000개의 노드를 패치하는 경우 패치는 동일합니다.

스냅샷 ID와 패치 기준 재정의를 모두 사용하는 경우 스냅샷 ID가 패치 기준 재정의보다 우선합니다. 기준 재정의 규칙은 계속 사용되지만 한 번만 평가됩니다. 이전 예에서 1,000개 관리형 노드의 패치는 여전히 항상 동일합니다. 패치 중간에 참조된 S3 버킷의 JSON 파일을 다른 파일로 변경한 경우 적용되는 패치는 여전히 동일합니다. 스냅샷 ID가 제공되었기 때문입니다.

기준 재정의 및 설치 재정의 목록 동시 사용

이 두 파라미터를 동시에 사용할 수 없습니다. 두 파라미터가 모두 제공되면 패치 문서가 실패하고 관리형 노드에서 스캔이나 설치를 수행하지 않습니다.

코드 예시

Python의 다음 코드 예제에서는 패치 기준 재정의를 생성하는 방법을 보여줍니다.

```
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)
```

이렇게 하면 다음과 같은 패치 기준 재정의가 생성됩니다.

```
[
    {
        "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": []
}
```

```
]
```

패치 기준 정보

이 섹션의 주제에서는 관리형 에서 `Scan` 또는 `Install` 작업을 실행할 때, AWS Systems Manager의 기능인 Patch Manager에서 패치 기준의 작동 방식에 대한 정보를 제공합니다.

주제

- 미리 정의된 패치 및 사용자 지정 패치 기준 정보
- 승인 패치 및 거부 패치 목록의 패키지 이름 형식 정보
- 패치 그룹 정보
- Microsoft에서 릴리스한 Windows Server 기반 애플리케이션 패치 정보

미리 정의된 패치 및 사용자 지정 패치 기준 정보

AWS Systems Manager의 기능인 Patch Manager는 Patch Manager에서 지원하는 운영 체제마다 미리 정의된 패치 기준을 제공합니다. 이러한 기준은 현재 구성된 대로 사용하거나(사용자 지정할 수 없음) 고유한 사용자 지정 패치 기준을 생성할 수 있습니다. 사용자 정의 패치 기준을 사용하면 환경에 대해 승인 또는 거부되는 패치를 더욱 효과적으로 제어할 수 있습니다. 또한 미리 정의된 기준은 해당 기준을 사용하여 설치된 모든 패치에 `Unspecified`의 규정 준수 수준을 할당합니다. 규정 준수 값을 할당하려면 미리 정의된 기준의 복사본을 생성하고 패치에 할당할 규정 준수 값을 지정할 수 있습니다. 자세한 내용은 사용자 지정 기준 정보 및 사용자 정의 패치 기준 작업 섹션을 참조하세요.

> **Note**
>
> 이 항목의 정보는 패치 작업에 사용하는 구성 방법 또는 유형에 관계없이 적용됩니다.
>
> - Quick Setup에서 구성된 패치 정책
> - Quick Setup에서 구성된 호스트 관리 옵션
> - 패치 `Scan` 또는 `Install` 태스크를 실행하기 위한 유지 관리 기간
> - 온디맨드 Patch now(지금 패치) 작업

주제

- 미리 정의된 패치 기준 정보
- 사용자 지정 기준 정보

미리 정의된 패치 기준 정보

다음은 Patch Manager와 함께 제공되는 미리 정의된 패치 기준을 설명하는 표입니다.

Patch Manager가 지원하는 각 운영 체제 버전에 대한 자세한 내용은 Patch Manager 필수 조건 섹션을 참조하세요.

명칭	지원되는 운영 체제	Details
AWS-AlmaLinuxDefaultPatchBaseline	AlmaLinux	"Security"로 분류되고, 심각도 수준이 "Critical" 또는 "Important"로 분류되는 모든 운영 체제 패치를 승인합니다. 또한 'Bugfix'로 분류되는 모든 패치도 승인합니다. 패치는 릴리스되거나 업데이트되고 7일 후 자동 승인됩니다.[1]
AWS-AmazonLinuxDefaultPatchBaseline	Amazon Linux 1	"Security"로 분류되고, 심각도 수준이 "Critical" 또는 "Important"로 분류되는 모든 운영 체제 패치를 승인합니다. 또한 'Bugfix'로 분류되는 모든 패치도 자동 승인합니다. 패치는 릴리스되거나 업데이트되고 7일 후 자동 승인됩니다.[1]
AWS-AmazonLinux2DefaultPatchBaseline	Amazon Linux 2	"Security"로 분류되고, 심각도 수준이 "Critical" 또는 "Important"로 분류되는 모든 운영 체제 패치를 승인합니다. 또한 'Bugfix'로 분류되는 모든 패치도 승인합니다. 패치는 릴리스 7일 후 자동 승인됩니다.[1]
AWS-AmazonLinux2022DefaultPatchBaseline	Amazon Linux 2022	"Security"로 분류되고, 심각도 수준이 "Critical" 또는 "Important"로 분류되는 모든

명칭	지원되는 운영 체제	Details
		운영 체제 패치를 승인합니다. 패치는 출시 7일 후 자동 승인됩니다. 또한 릴리스 후 7일간 "Bugfix"로 분류되는 모든 패치도 승인합니다.
AWS-AmazonLinux2023DefaultPatchBaseline	Amazon Linux 2023	"Security"로 분류되고, 심각도 수준이 "Critical" 또는 "Important"로 분류되는 모든 운영 체제 패치를 승인합니다. 패치는 출시 7일 후 자동 승인됩니다. 또한 릴리스 후 7일간 "Bugfix"로 분류되는 모든 패치도 승인합니다.
AWS-CentOSDefaultPatchBaseline	CentOS 및 CentOS Stream	업데이트가 제공되고 7일 후에 모든 업데이트를 승인합니다(비보안 업데이트 포함).
AWS-DebianDefaultPatchBaseline	Debian Server	우선순위가 "Required", "Important", "Standard," "Optional" 또는 "Extra"로 분류되는 모든 운영 체제 보안 관련 패치를 즉시 승인합니다. 신뢰할 수 있는 릴리스 날짜가 리포지토리에 제공되지 않으므로 승인되기까지 대기 시간이 없습니다.
AWS-MacOSDefaultPatchBaseline	macOS	“보안”으로 분류되는 모든 운영 체제 패치를 승인합니다. 또한 현재 업데이트가 있는 모든 패키지를 승인합니다.

명칭	지원되는 운영 체제	Details
AWS-OracleLinuxDefaultPatchBaseline	Oracle Linux	"Security"로 분류되고, 심각도 수준이 "Important" 또는 "Moderate"로 분류되는 모든 운영 체제 패치를 승인합니다. 또한 릴리스 후 7일간 'Bugfix'로 분류되는 모든 패치도 승인합니다. 패치는 릴리스되거나 업데이트되고 7일 후 자동 승인됩니다.[1]
AWS-DefaultRaspbianPatchBaseline	Raspberry Pi OS	우선순위가 "Required", "Important", "Standard," "Optional" 또는 "Extra"로 분류되는 모든 운영 체제 보안 관련 패치를 즉시 승인합니다. 신뢰할 수 있는 릴리스 날짜가 리포지토리에 제공되지 않으므로 승인되기까지 대기 시간이 없습니다.
AWS-RedHatDefaultPatchBaseline	Red Hat Enterprise Linux (RHEL)	"Security"로 분류되고, 심각도 수준이 "Critical" 또는 "Important"로 분류되는 모든 운영 체제 패치를 승인합니다. 또한 'Bugfix'로 분류되는 모든 패치도 승인합니다. 패치는 릴리스되거나 업데이트되고 7일 후 자동 승인됩니다.[1]

명칭	지원되는 운영 체제	Details
AWS-RockyLinuxDefaultPatchBaseline	Rocky Linux	"Security"로 분류되고, 심각도 수준이 "Critical" 또는 "Important"로 분류되는 모든 운영 체제 패치를 승인합니다. 또한 'Bugfix'로 분류되는 모든 패치도 승인합니다. 패치는 릴리스되거나 업데이트되고 7일 후 자동 승인됩니다.[1]
AWS-SuseDefaultPatchBaseline	SUSE Linux Enterprise Server (SLES)	"Security"로 분류되고, 심각도가 "Critical" 또는 "Important"로 분류되는 모든 운영 체제 패치를 승인합니다. 패치는 릴리스되거나 업데이트되고 7일 후 자동 승인됩니다.[1]
AWS-UbuntuDefaultPatchBaseline	Ubuntu Server	우선순위가 "Required", "Important", "Standard," "Optional" 또는 "Extra"로 분류되는 모든 운영 체제 보안 관련 패치를 즉시 승인합니다. 신뢰할 수 있는 릴리스 날짜가 리포지토리에 제공되지 않으므로 승인되기까지 대기 시간이 없습니다.
AWS-DefaultPatchBaseline	Windows Server	"CriticalUpdates" 또는 "SecurityUpdates"로 분류되고, MSRC 심각도가 "Critical" 또는 "Important"로 분류되는 모든 Windows Server 운영 체제 패치를 승인합니다. 패치는 릴리스되거나 업데이트되고 7일 후 자동 승인됩니다.[2]

명칭	지원되는 운영 체제	Details
AWS-WindowsPredefinedPatchBaseline-OS	Windows Server	"CriticalUpdates" 또는 "SecurityUpdates"로 분류되고, MSRC 심각도가 "Critical" 또는 "Important"로 분류되는 모든 Windows Server 운영 체제 패치를 승인합니다. 패치는 릴리스되거나 업데이트되고 7일 후 자동 승인됩니다.[2]
AWS-WindowsPredefinedPatchBaseline-OS-Applications	Windows Server	Windows Server 운영 체제의 경우, "CriticalUpdates" 또는 "SecurityUpdates"로 분류되고, MSRC 심각도가 "Critical" 또는 "Important"로 분류되는 모든 운영 체제 패치를 승인합니다. Microsoft에서 릴리스한 애플리케이션의 경우 모든 패치를 승인합니다. OS 및 애플리케이션 모두에 해당하는 패치는 릴리스 또는 업데이트 7일 후 자동 승인됩니다.[2]

[1] Amazon Linux 1 및 Amazon Linux 2의 경우 패치가 자동 승인되기까지의 7일 대기 시간은 `Release Date` 값이 아니라 `updateinfo.xml`의 `Updated Date` 값에서 계산됩니다. 다양한 요인이 `Updated Date` 값에 영향을 줄 수 있습니다. 다른 운영 체제에서는 릴리스 날짜와 업데이트 날짜를 다르게 처리합니다. 자동 승인 지연으로 인한 예상치 못한 결과를 방지하는 데 도움이 되는 정보를 알아보려면 패키지 릴리스 날짜 및 업데이트 날짜 계산 방법 섹션을 참조하세요.

[2] Windows Server의 경우 기본 기준선에는 7일 자동 승인 지연이 포함됩니다. 릴리스 후 7일 내에 패치를 설치하려면 사용자 지정 기준선을 생성해야 합니다.

사용자 지정 기준 정보

자체 패치 기준선을 생성하는 경우 다음 카테고리를 사용하여 자동 승인할 패치를 선택할 수 있습니다.

- 운영 체제: Windows Server, Amazon Linux, Ubuntu Server 등

- 제품 이름(운영 체제용): 예를 들어 RHEL 6.5, Amazon Linux 2014.09, Windows Server 2012, Windows Server 2012 R2 등
- 제품 이름(Microsoft에서 릴리스한 Windows Server 기반 애플리케이션만 해당): 예를 들어 Word 2016, BizTalk Server 등
- 분류: 예를 들어 필수 업데이트, 보안 업데이트 등
- 심각도: 예를 들어 Critical, Important 등

생성한 각 승인 규칙에 대해 자동 승인 지연을 지정하거나 패치 승인 마감 날짜를 지정할 수 있습니다.

> **Note**
>
> Ubuntu Server에 대한 업데이트 패키지의 릴리스 날짜를 확실히 결정할 수 없으므로 이 운영 체제에서는 자동 승인 옵션이 지원되지 않습니다.

자동 승인 지연은 패치가 릴리스되거나 마지막으로 업데이트된 후 패치 적용을 위해 자동 승인되기까지 대기하는 일 수입니다. 예를 들어, `CriticalUpdates` 분류를 사용하여 규칙을 생성하고 자동 승인 지연 기간을 7일로 구성하면 7월 7일에 릴리스된 새 필수 패치가 7월 14일에 자동으로 승인됩니다.

> **Note**
>
> Linux 리포지토리가 패키지 릴리스 날짜 정보를 제공하지 않는 경우에는 Systems Manager가 패키지 빌드 시간을 Amazon Linux 1, Amazon Linux 2, RHEL, CentOS에 대한 자동 승인 지연 시간으로 사용합니다. 이때 시스템이 패키지 빌드 시간을 찾지 못하면 Systems Manager가 자동 승인 지연 시간을 0으로 처리합니다.

자동 승인 마감 날짜를 지정하면 Patch Manager는 해당 날짜 또는 그 이전에 릴리스되거나 마지막으로 업데이트된 모든 패치를 자동으로 적용합니다. 예를 들어 2023년 7월 7일을 마감 날짜로 지정하면 2023년 7월 8일 당일 또는 이후에 릴리스되거나 마지막으로 업데이트된 모든 패치가 자동으로 설치되지 않습니다.

> **Note**
>
> 사용자 지정 패치 기준선을 생성할 때 해당 패치 기준선에서 승인된 패치의 규정 준수 심각도 수준을 지정할 수 있습니다(예: `Critical` 또는 `High`). 승인된 패치의 패치 상태가

Missing으로 보고되는 경우 패치 기준선의 전반적인 보고된 규정 준수 심각도는 사용자가 지정한 심각도 수준입니다.

패치 기준을 생성하는 경우 다음 사항에 유의하십시오.

- Patch Manager는 지원되는 운영 체제에 따라 미리 정의된 패치 기준이 있습니다. 고유한 패치 기준을 생성하고 이를 해당 운영 체제 유형의 기본값으로 지정하지 않는 한, 이 미리 정의된 패치 기준이 각 운영 체제 유형의 기본 패치 기준으로 사용됩니다.

  > **Note**
  >
  > Windows Server에는 3개의 미리 정의된 패치 기준이 제공됩니다. 패치 기준 AWS-DefaultPatchBaseline 및 AWS-WindowsPredefinedPatchBaseline-OS는 Windows 운영 체제 자체에서 운영 체제 업데이트만 지원합니다. AWS-DefaultPatchBaseline은 다른 패치 기준을 지정하지 않는 한 Windows Server 관리형 노드의 기본 패치 기준으로 사용됩니다. 이러한 두 패치 기준의 구성 설정은 동일합니다. 둘 중 더 새로운 AWS-WindowsPredefinedPatchBaseline-OS는 Windows Server에 대해 미리 정의된 세 번째 패치 기준과 구별하기 위해 생성되었습니다. 해당 패치 기준인 AWS-WindowsPredefinedPatchBaseline-OS-Applications는 Windows Server 운영 체제 및 Microsoft에서 릴리스한 지원되는 애플리케이션을 패치하는 데 사용될 수 있습니다.

- 온프레미스 서버 및 가상 머신(VM)의 경우 Patch Manager가 사용자 정의 기본 패치 기준을 사용하려고 시도합니다. 사용자 지정 기본 패치 기준선이 없는 경우 시스템은 해당 운영 체제에 사전 정의되어 있는 패치 기준선을 사용합니다.
- 패치가 동일한 패치 기준선에 승인 및 거부로 나열되면 패치가 거부됩니다.
- 관리형 노드마다 정의할 수 있는 패치 기준선은 각각 1개로 제한됩니다.
- 패치 기준에 대한 승인된 패치 및 거부된 패치 목록에 추가할 수 있는 패키지 이름의 형식은 패치를 적용하는 운영 체제의 유형에 따라 다릅니다.

  승인된 패치 및 거부된 패치 목록의 승인된 형식에 대한 자세한 내용은 승인 패치 및 거부 패치 목록의 패키지 이름 형식 정보 섹션을 참조하세요.
- Quick Setup에서 패치 정책 구성을 사용하는 경우 사용자 지정 패치 기준선에 대한 업데이트는 한 시간에 한 번씩 Quick Setup과 동기화됩니다.

  패치 정책에서 참조된 사용자 지정 패치 기준선이 삭제되면 해당 패치 정책의 Quick Setup Configuration details(구성 세부 정보) 페이지에 배너가 표시됩니다. 배너는 패치 정책에서 더 이상

존재하지 않는 패치 기준선을 참조하고 있으며 후속 패치 작업이 실패할 것임을 알려줍니다. 이 경우 Quick Setup Configurations(구성) 페이지로 돌아가서 Patch Manager 구성을 선택하고 Actions(작업), Edit configuration(구성 편집)을 선택합니다. 삭제된 패치 기준선 이름이 강조 표시되며, 영향을 받는 운영 체제의 새 패치 기준선을 선택해야 합니다.

패치 기준 생성에 대한 자세한 내용은 사용자 정의 패치 기준 작업 및 자습서: 서버 환경에 패치 적용 (AWS CLI) 섹션을 참조하세요.

승인 패치 및 거부 패치 목록의 패키지 이름 형식 정보

승인된 패치 및 거부된 패치 목록에 추가할 수 있는 패키지 이름의 형식은 패치를 적용하는 운영 체제의 유형에 따라 다릅니다.

Linux 운영 체제의 패키지 이름 형식

패치 기준의 승인 패치와 거부 패치에 지정할 수 있는 형식은 Linux 유형별로 다릅니다. 즉, 지원되는 형식은 Linux 운영 체제의 유형에서 사용하는 패키지 관리자에 따라 다릅니다.

주제

- Amazon Linux 1, Amazon Linux 2, Amazon Linux 2022, Amazon Linux 2023, CentOS, Oracle Linux, Red Hat Enterprise Linux(RHEL)
- Debian Server, Raspberry Pi OS(구 Raspbian) 및 Ubuntu Server
- SUSE Linux Enterprise Server (SLES)

Amazon Linux 1, Amazon Linux 2, Amazon Linux 2022, Amazon Linux 2023, CentOS, Oracle Linux, Red Hat Enterprise Linux(RHEL)

패키지 관리자: DNF를 패키지 관리자로 사용하는 Amazon Linux 2022, Amazon Linux 2023, RHEL 8 및 CentOS 8을 제외한 YUM

승인된 패치: 승인 패치에 다음을 지정할 수 있습니다.

- Bugzilla ID, `1234567` 형식(숫자로만 구성된 문자열만 Bugzilla ID로 처리됨)
- CVE IDs, `CVE-2018-1234567` 형식
- Advisory ID, `RHSA-2017:0864` 및 `ALAS-2018-123` 형식
- 전체 패키지 이름, 형식 예:

  - `example-pkg-0.710.10-2.7.abcd.x86_64`
  - `pkg-example-EE-20180914-2.2.amzn1.noarch`
- 와일드카드 한 개를 포함하는 패키지 이름, 형식 예:
  - `example-pkg-*.abcd.x86_64`
  - `example-pkg-*-20180914-2.2.amzn1.noarch`
  - `example-pkg-EE-2018*.amzn1.noarch`

거부된 패치: 거부 패치에 다음을 입력할 수 있습니다.

- 전체 패키지 이름, 형식 예:
  - `example-pkg-0.710.10-2.7.abcd.x86_64`
  - `pkg-example-EE-20180914-2.2.amzn1.noarch`
- 와일드카드 한 개를 포함하는 패키지 이름, 형식 예:
  - `example-pkg-*.abcd.x86_64`
  - `example-pkg-*-20180914-2.2.amzn1.noarch`
  - `example-pkg-EE-2018*.amzn1.noarch`

Debian Server, Raspberry Pi OS(구 Raspbian) 및 Ubuntu Server

패키지 관리자: APT

승인된 패치와 거부된 패치: 승인 패치와 거부 패치 모두에 다음을 지정할 수 있습니다.

- 패키지 이름, `ExamplePkg33` 형식

> **Note**
>
> Debian Server 목록, Raspberry Pi OS 목록 및 Ubuntu Server 목록의 경우 아키텍처나 버전 등의 요소가 포함되지 않습니다. 예를 들어 패치 목록에 다음을 모두 포함시키려면 `ExamplePkg33` 패키지 이름을 지정합니다.
>
> - `ExamplePkg33.x86.1`
> - `ExamplePkg33.x86.2`
> - `ExamplePkg33.x64.1`
> - `ExamplePkg33.3.2.5-364.noarch`

SUSE Linux Enterprise Server (SLES)

패키지 관리자: Zypper

승인된 패치와 거부된 패치: 승인 패치와 거부 패치 목록 모두에 다음을 지정할 수 있습니다.

- 전체 패키지 이름, 형식 예:
  - `SUSE-SLE-Example-Package-12-2018-123`
  - `example-pkg-2018.11.4-46.17.1.x86_64.rpm`
- 와일드카드 한 개를 포함하는 패키지 이름, 예:
  - `SUSE-SLE-Example-Package-12-2018-*`
  - `example-pkg-2018.11.4-46.17.1.*.rpm`

macOS의 패키지 이름 형식

지원되는 패키지 관리자: 소프트웨어 업데이트, 설치 관리자, Brew, Brew Cask

[승인된 패치(Approved patches)] 및 [거부된 패치(rejected patches)]: : 승인된 패치 목록과 거부된 패치 목록 모두에 대해 다음과 같은 형식으로 전체 패키지 이름을 지정합니다.

- `XProtectPlistConfigData`
- `MRTConfigData`

macOS에 대한 승인된 패치 및 거부된 패치 목록에서는 와일드카드가 지원되지 않습니다.

Windows 운영 체제의 패키지 이름 형식

Windows 운영 체제의 경우 Microsoft Knowledge Base ID와 Microsoft Security Bulletin ID를 사용하여 패치를 지정합니다. 다음 예를 참조하십시오.

```
KB2032276,KB2124261,MS10-048
```

패치 그룹 정보

> **Important**
>
> 패치 그룹은 패치 정책을 기반으로 하는 패치 적용 작업에 사용되지 않습니다. 패치 정책 작업에 대한 자세한 내용은 Quick Setup 패치 정책 사용 섹션을 참조하세요.

패치 그룹을 사용하여 AWS Systems Manager의 기능인 Patch Manager에서 관리형 노드를 특정 패치 기준과 연결할 수 있습니다. 패치 그룹을 사용하면 연결된 패치 기준 규칙을 기반으로 적절한 패치를 정확하게 해당 노드 집합에 배포할 수 있습니다. 패치 그룹은 거치기 전에 패치를 배포하는 것을 방지하는 데도 도움이 됩니다. 예를 들어 다양한 환경(예: 개발, 테스트 및 프로덕션)에 필요한 패치 그룹을 만들고 적절한 패치 기준에 각 패치 그룹을 등록할 수 있습니다.

`AWS-RunPatchBaseline`을 실행하면 ID 또는 태그를 사용하여 관리형 노드를 대상으로 지정할 수 있습니다. SSM Agent와 Patch Manager는 관리형 노드에 추가한 패치 그룹 값에 따라 사용할 패치 기준을 판단합니다.

Amazon Elastic Compute Cloud(Amazon EC2) 태그를 사용하여 패치 그룹을 생성합니다. Systems Manager를 통한 다른 태깅 시나리오와 달리 패치 그룹은 반드시 태그 키 `Patch Group` 또는 `PatchGroup`으로 정의되어야 합니다. 키는 대/소문자를 구분합니다. '웹 서버' 또는 'US-EAST-PROD'와 같이 해당 그룹의 리소스를 식별하고 대상으로 지정하는 데 도움이 되는 값을 지정할 수 있지만 키는 `Patch Group` 또는 `PatchGroup`이어야 합니다.

패치 그룹 및 태그 관리형 노드를 생성한 이후 패치 그룹을 패치 기준선에 등록할 수 있습니다. 패치 기준으로 패치 그룹을 등록하면 패치 그룹 내부의 노드가 연결된 패치 기준에 정의된 규칙을 사용합니다.

패치 그룹을 생성하고 패치 기준에 연결하는 방법에 대한 자세한 내용은 패치 그룹 작업 및 패치 기준에 패치 그룹 추가를 참조하세요.

AWS Command Line Interface(AWS CLI)를 사용하여 패치 기준 및 패치 그룹을 생성하는 방법을 보려면 자습서: 서버 환경에 패치 적용(AWS CLI) 섹션을 참조하세요. Amazon EC2 태그에 대한 자세한 내용은 Amazon EC2 사용 설명서의 Amazon EC2 리소스에 태그 지정을 참조하세요.

작동 방식

시스템이 패치 기준을 관리형 노드에 적용하기 위해 태스크를 실행하면 SSM Agent는 패치 그룹 값이 해당 노드에 대해 정의되었는지 확인합니다. 노드가 패치 그룹에 할당되는 경우 Patch Manager는 이제 그 그룹에 등록되는 패치 기준을 확인합니다. 해당 그룹에 대한 패치 기준이 검색되면 Patch Manager는 SSM Agent에 해당 패치 기준 사용을 알립니다. 노드가 패치 그룹에 대해 구성되지 않은 경우 Patch Manager는 SSM Agent에 현재 구성된 기본 패치 기준을 자동으로 사용함을 알립니다.

> **Important**
>
> 관리형 노드는 하나의 패치 그룹에만 있을 수 있습니다.
> 패치 그룹은 각 운영 체제 유형에 대해 하나의 패치 기준에만 등록할 수 있습니다.
> Allow tags in instance metadata(인스턴스 메타데이터의 태그 허용) 옵션이 인스턴스에서 활성화된 경우 `Patch Group` 태그(공백 포함)를 Amazon EC2 인스턴스에 적용할 수 없습니다. 인

스턴스 메타데이터에서 태그를 허용하면 태그 키 이름에 공백이 포함되지 않습니다. EC2 인스턴스 메타데이터에 태그를 허용한 경우 태그 키 PatchGroup(공백 없음)을 사용해야 합니다.

다음 다이어그램은 Patch Manager를 사용하여 패치하기 위해 Run Command 태스크를 서버 플릿으로 보낼 때 Systems Manager가 수행하는 프로세스의 일반적인 예를 보여줍니다. 유지 관리 기간이 Patch Manager를 사용하여 패치할 명령을 보내도록 구성된 경우에도 유사한 프로세스가 사용됩니다.

이 예제에서는 다음 태그가 적용된 세 개의 Windows Server용 EC2 인스턴스 그룹이 있습니다.

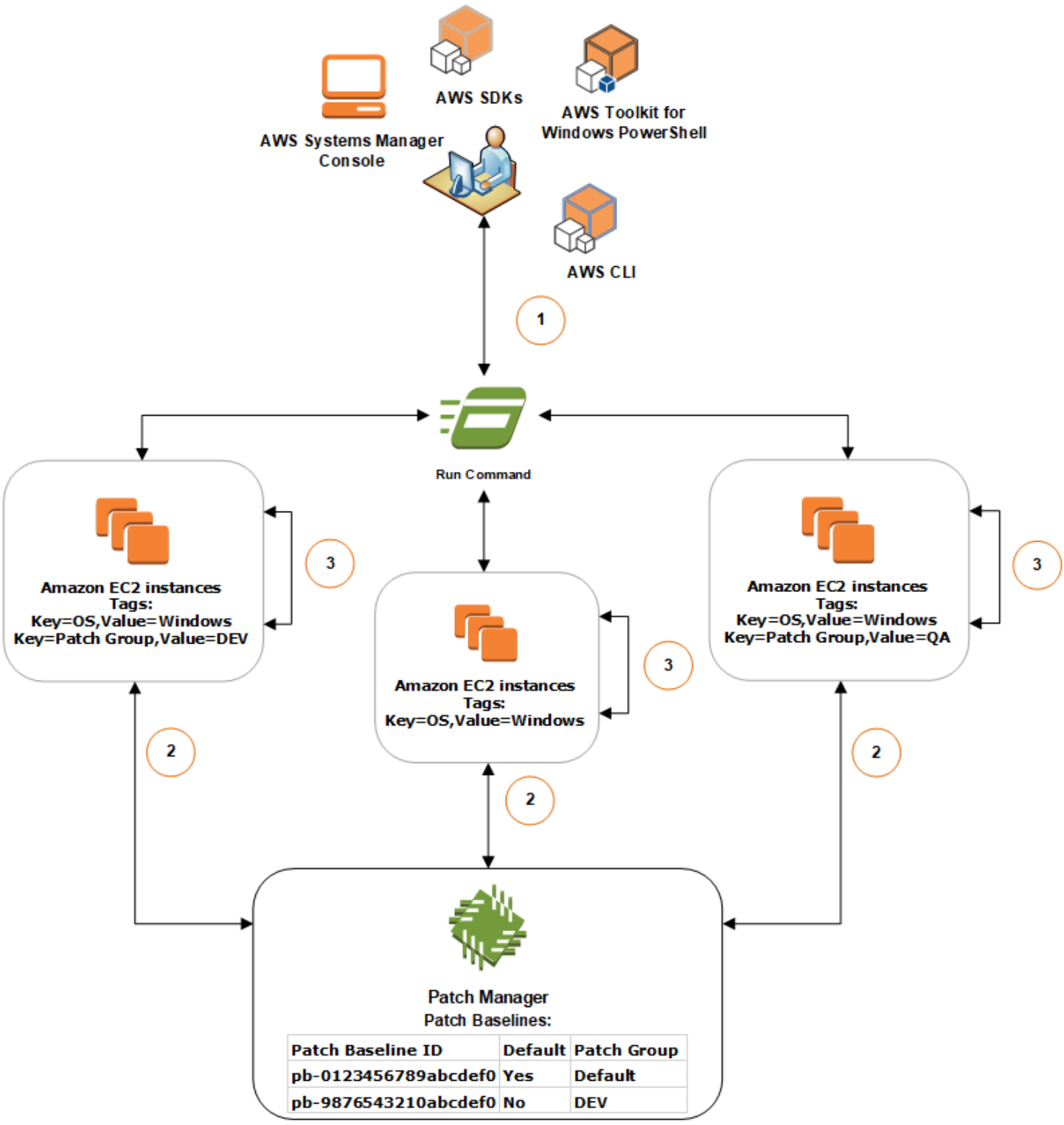
EC2 인스턴스 그룹	Tags
그룹 1	key=OS,value=Windows key=PatchGroup,value=DEV
그룹 2	key=OS,value=Windows
그룹 3	key=OS,value=Windows key=PatchGroup,value=QA

이 예에서는 2개의 Windows Server 패치 기준도 있습니다.

패치 기준 ID	기본값	연결된 패치 그룹
pb-0123456789abcdef0	예	Default
pb-9876543210abcdef0	아니요	DEV

다이어그램 1: 패치 작업 프로세스 흐름의 일반적인 예

다음 다이어그램에서는 Patch Manager가 패치 작업에 사용할 패치 기준선을 결정하는 방법을 보여줍니다.



AWS Systems Manager의 기능인 Run Command와 Patch Manager를 사용하여 패치를 검색하거나 설치하는 일반적인 프로세스는 다음과 같습니다.

1. 패치 명령 보내기: Systems Manager 콘솔, SDK, AWS Command Line Interface(AWS CLI) 또는 AWS Tools for Windows PowerShell을 통해 문서 `AWS-RunPatchBaseline`을 사용하여 Run Command 태스크를 보냅니다. 이 다이어그램은 태그 `key=OS,value=Windows`를 대상으로 관리된 인스턴스를 패치하기 위한 Run Command 태스크를 보여줍니다.
2. 패치 기준 결정: SSM Agent는 EC2 인스턴스에 적용된 패치 그룹 태그를 확인하고 Patch Manager에게 해당 패치 기준을 쿼리합니다.
   - 패치 기준과 연결된 일치하는 패치 그룹 값:
     1. 그룹 1의 EC2 인스턴스에 설치된 SSM Agent는 1단계에서 지시된 명령을 수신하여 패치 작업을 시작합니다. SSM Agent는 EC2 인스턴스에 패치 그룹 태그 값 `DEV`가 적용되었는지 확인하고 Patch Manager에게 연결된 패치 기준을 쿼리합니다.
     2. Patch Manager는 패치 기준 `pb-9876543210abcdef0`에 패치 그룹 `DEV`가 연결되어 있는지 확인하고 SSM Agent에 알립니다.
     3. SSM Agent는 `pb-9876543210abcdef0`에 구성된 승인 규칙 및 예외를 기반으로 Patch Manager에서 패치 기준 스냅샷을 검색하고 다음 단계로 진행합니다.
   - 패치 그룹 태그가 인스턴스에 추가되지 않은 경우:
     1. 그룹 2의 EC2 인스턴스에 설치된 SSM Agent는 1단계에서 지시된 명령을 수신하여 패치 작업을 시작합니다. SSM Agent는 EC2 인스턴스에 `Patch Group` 또는 `PatchGroup` 태그가 적용되지 않았음을 확인하고 그에 따라 SSM Agent는 Patch Manager에 기본 Windows 패치 기준선을 쿼리합니다.
     2. Patch Manager는 기본 Windows Server 패치 기준이 `pb-0123456789abcdef0`인지 확인하고 SSM Agent에 알립니다.
     3. SSM Agent는 기본 패치 기준 `pb-0123456789abcdef0`에 구성된 승인 규칙 및 예외를 기반으로 Patch Manager에서 패치 기준 스냅샷을 검색하고 다음 단계로 진행합니다.
   - 패치 기준과 연결된 일치하는 패치 그룹 값이 없는 경우:
     1. 그룹 3의 EC2 인스턴스에 설치된 SSM Agent는 1단계에서 지시된 명령을 수신하여 패치 작업을 시작합니다. SSM Agent는 EC2 인스턴스에 패치 그룹 태그 값 `QA`가 적용되었는지 확인하고 Patch Manager에게 연결된 패치 기준을 쿼리합니다.
     2. Patch Manager는 패치 그룹 `QA`가 연결되어 있는 패치 기준을 찾지 못합니다.
     3. Patch Manager는 SSM Agent에게 기본 Windows 패치 기준 `pb-0123456789abcdef0`을 사용할 것을 알립니다.
     4. SSM Agent는 기본 패치 기준 `pb-0123456789abcdef0`에 구성된 승인 규칙 및 예외를 기반으로 Patch Manager에서 패치 기준 스냅샷을 검색하고 다음 단계로 진행합니다.

3. 패치 스캔 또는 설치: 사용할 적절한 패치 기준을 결정한 후 SSM Agent는 1단계에서 지정한 작업 값을 기준으로 패치를 스캔하거나 설치합니다. 검사 또는 설치되는 패치는 Patch Manager가 제공하는 패치 기준 스냅샷에 정의된 승인 규칙 및 패치 예외에 의해 결정됩니다.

추가 정보

- 패치 규정 준수 상태 값 이해

Microsoft에서 릴리스한 Windows Server 기반 애플리케이션 패치 정보

이 주제의 정보를 사용하여 AWS Systems Manager의 기능인 Patch Manager로 Windows Server 기반 애플리케이션을 패치할 준비를 합니다.

Microsoft 애플리케이션 패치

Windows Server 관리형 노드의 애플리케이션에 대한 패치 지원은 Microsoft에서 릴리스한 애플리케이션으로 제한됩니다.

> **Note**
>
> 경우에 따라 Microsoft는 업데이트된 날짜와 시간을 지정하지 않는 애플리케이션에 대한 패치를 릴리스합니다. 이 경우 업데이트된 `01/01/1970`의 날짜 및 시간은 기본적으로 제공됩니다.

Microsoft에서 릴리스한 애플리케이션 패치를 위한 패치 기준

Windows Server에는 3개의 미리 정의된 패치 기준이 제공됩니다. 패치 기준 `AWS-DefaultPatchBaseline` 및 `AWS-WindowsPredefinedPatchBaseline-OS`는 Windows 운영 체제 자체에서 운영 체제 업데이트만 지원합니다. `AWS-DefaultPatchBaseline`은 다른 패치 기준을 지정하지 않는 한 Windows Server 관리형 노드의 기본 패치 기준으로 사용됩니다. 이러한 두 패치 기준의 구성 설정은 동일합니다. 둘 중 더 새로운 `AWS-WindowsPredefinedPatchBaseline-OS`는 Windows Server에 대해 미리 정의된 세 번째 패치 기준과 구별하기 위해 생성되었습니다. 해당 패치 기준인 `AWS-WindowsPredefinedPatchBaseline-OS-Applications`는 Windows Server 운영 체제 및 Microsoft에서 릴리스한 지원되는 애플리케이션을 패치하는 데 사용될 수 있습니다.

Microsoft에서 릴리스한 Windows Server 시스템 기반 애플리케이션을 업데이트하는 사용자 정의 패치 기준을 생성할 수도 있습니다.

Microsoft에서 릴리스한 온프레미스 서버, 엣지 디바이스, VM 및 기타 비EC2 노드 기반 애플리케이션 패치에 대한 지원

Microsoft에서 릴리스한 가상 머신 및 기타 비 EC2 관리형 노드의 애플리케이션에 패치를 적용하려면 고급 인스턴스 티어를 설정해야 합니다. 고급 인스턴스 티어를 사용하는 데는 요금이 부과됩니다. 그러나 Microsoft에서 릴리스한 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 기반 애플리케이션을 패치하는 데는 추가 요금이 부과되지 않습니다. 자세한 내용은 인스턴스 티어 구성 단원을 참조하십시오.

"다른 Microsoft 제품"에 대한 Windows 업데이트 옵션

Patch Manager가 Microsoft에서 릴리스한 Windows Server 관리형 노드 기반 애플리케이션을 패치할 수 있도록 관리형 노드에서 Windows 업데이트 옵션 Windows를 업데이트할 때 다른 Microsoft 제품에 대한 업데이트 제공(Give me updates for other Microsoft products when I update Windows)을 활성화해야 합니다.

단일 관리형 노드에서 이 옵션 허용에 대한 자세한 내용은 Microsoft Support 웹 사이트의 Microsoft 업데이트를 사용하여 Office 업데이트를 참조하세요.

Windows Server 2016 이상을 실행하는 관리형 노드 플릿의 경우 그룹 정책 객체(GPO)를 사용하여 설정을 켤 수 있습니다. 그룹 정책 관리 편집기에서 [컴퓨터 구성(Computer Configuration)], [관리 템플릿(Administrative Templates)], [Windows 구성 요소(Windows Components)], [Windows 업데이트(Windows Updates)]로 이동하고 [다른 Microsoft 제품에 대한 업데이트 설치(Install updates for other Microsoft products)]를 선택합니다. 또한 계획되지 않은 자동 업데이트 및 Patch Manager 외부 재부팅을 방지하는 추가 파라미터를 사용하여 GPO를 구성하는 것이 좋습니다. 자세한 내용은 Microsoft 기술 문서 웹 사이트의 Non-Active Directory 환경에서 자동 업데이트 구성을 참조하세요.

Windows Server 2012 또는 2012 R2를 실행하는 관리형 노드 플릿의 경우 Microsoft Docs 블로그 웹 사이트의 Enabling and Disabling Microsoft Update in Windows 7 via Script에 설명된 대로 스크립트를 사용하여 옵션을 설정할 수 있습니다. 예를 들어 다음을 수행할 수 있습니다.

1. 블로그 게시물의 스크립트를 파일로 저장합니다.
2. 파일을 Amazon Simple Storage Service(Amazon S3) 버킷 또는 기타 액세스 가능한 위치에 업로드합니다.
3. AWS Systems Manager의 기능인 Run Command로 다음과 같은 명령에 Systems Manager 문서(SSM 문서) `AWS-RunPowerShellScript`를 사용하여 관리형 노드에서 스크립트를 실행합니다.

```
Invoke-WebRequest `
    -Uri "https://s3.aws-api-domain/DOC-EXAMPLE-BUCKET/script.vbs" `
    -Outfile "C:\script.vbs" cscript c:\script.vbs
```

최소 파라미터 요구 사항

사용자 정의 패치 기준에 Microsoft에서 릴리스한 애플리케이션을 포함하려면 최소한 패치할 제품을 지정해야 합니다. 다음 AWS Command Line Interface(AWS CLI) 명령은 Microsoft Office 2016과 같이 제품을 패치하는 데 필요한 최소 요구 사항을 보여줍니다.

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}]"
```

Microsoft 애플리케이션 제품군을 지정하는 경우, 지정하는 각 제품은 선택한 제품군의 지원 구성원이어야 합니다. 예를 들어 "Active Directory 권한 관리 서비스 클라이언트 2.0" 제품을 패치하려면 해당 제품군을 "Office"또는 "SQL Server"가 아닌 "Active Directory"로 지정해야 합니다. 다음 AWS CLI 명령은 제품군 및 제품으로 구성된 일치된 페어를 보여줍니다.

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**
>
> 일치하지 않는 제품 및 제품군 페어링에 대한 오류 메시지가 표시되는 경우 문제: 일치하지 않는 제품군/제품 페어에서 문제 해결을 위한 도움말을 찾아보세요.

Amazon Linux 2 관리형 노드에서 Kernel Live Patching 사용

Amazon Linux 2용 Kernel Live Patching을 사용하면 실행 중인 애플리케이션을 재부팅하거나 중단하지 않고 실행 중인 Linux 커널에 보안 취약성 및 중요 버그 패치를 적용할 수 있습니다. 이를 통해 서비스 및 애플리케이션 가용성을 향상하는 동시에 인프라를 최신 상태로 안전하게 유지할 수 있습니다. Kernel Live Patching은 Amazon EC2 인스턴스와 Amazon Linux 2를 실행하는 AWS IoT Greengrass 코어 디바이스, 온프레미스 가상 머신에서 지원됩니다.

Kernel Live Patching에 대한 일반적인 내용은 Amazon EC2 사용 설명서의 Amazon Linux 2의 Kernel Live Patching을 참조하세요.

Amazon Linux 2 관리형 노드에서 Kernel Live Patching을 설정한 후 AWS Systems Manager의 기능인 Patch Manager를 사용하여 커널 라이브 패치를 인스턴스에 적용할 수 있습니다. 노드에서 기존 yum 워크플로를 사용하여 업데이트를 적용하는 대신 Patch Manager를 사용할 수 있습니다.

시작하기 전 준비 사항

Patch Manager를 사용하여 커널 라이브 패치를 Amazon Linux 2 관리형 노드에 적용하려면 노드가 올바른 아키텍처와 커널 버전을 기반으로 해야 합니다. 자세한 내용은 Amazon EC2 사용 설명서의 지원되는 구성 및 사전 조건을 참조하세요.

주제

- Kernel Live Patching 및 Patch Manager 정보
- 작동 방식
- Run Command를 사용하여 Kernel Live Patching 설정
- Run Command을 사용하여 커널 라이브 패치 적용
- Run Command를 사용하여 Kernel Live Patching을 해제하려면

Kernel Live Patching 및 Patch Manager 정보

커널 버전 업데이트

커널 라이브 패치 업데이트를 적용한 후 관리형 노드를 재부팅할 필요가 없습니다. 그러나 AWS는 릴리스 후 최대 3개월 동안 Amazon Linux 2 커널 버전에 대한 커널 라이브 패치를 제공합니다. 3개월 후 커널 라이브 패치를 계속 받으려면 최신 커널 버전으로 업데이트해야 합니다. 유지 관리 기간을 사용하여 커널 버전 업데이트를 요청하는 메시지를 표시하도록 최소 3개월에 한 번 노드 재부팅을 예약하는 것이 좋습니다.

커널 라이브 패치 제거

커널 라이브 패치는 Patch Manager를 사용하여 제거할 수 없습니다. 대신 Kernel Live Patching을 해제하여 적용된 커널 라이브 패치에 대한 RPM 패키지를 제거할 수 있습니다. 자세한 내용은 Run Command를 사용하여 Kernel Live Patching을 해제하려면 단원을 참조하십시오.

커널 규정 준수

경우에 따라 현재 커널 버전에 대한 라이브 패치에서 모든 CVE 수정 사항을 설치하면 해당 커널이 최신 커널 버전과 동일한 규정 준수 상태가 될 수 있습니다. 이 경우 최신 버전은 `Installed`으로 보고되고 관리형 노드는 `Compliant`로 보고됩니다. 그러나 최신 커널 버전에서는 설치 시간이 보고되지 않습니다.

하나의 커널 라이브 패치, 여러 CVE

커널 라이브 패치가 여러 CVE를 해결하고 해당 CVE가 다양한 분류 및 심각도 값을 갖는 경우, CVE에서 가장 높은 분류 및 심각도만 해당 패치에 대해 보고됩니다.

이 섹션의 나머지 부분에서는 Patch Manager를 사용하여 이러한 요구 사항을 충족하는 관리형 노드에 커널 라이브 패치를 적용하는 방법을 설명합니다.

작동 방식

AWS는 Amazon Linux 2를 위한 2가지 유형의 커널 라이브 패치인 보안 업데이트와 버그 수정 사항을 릴리스합니다. 이러한 유형의 패치를 적용하려면 다음 표에 나열된 분류 및 심각도만 대상으로 하는 패치 기준 문서를 사용합니다.

분류	심각도
`Security`	`Critical`, `Important`

분류	심각도
Bugfix	All

이러한 패치만 대상으로 하는 사용자 지정 패치 기준을 생성하거나, 미리 정의된 AWS-AmazonLinux2DefaultPatchBaseline 패치 기준을 사용할 수 있습니다. 즉, Kernel Live Patching이 설정된 Amazon Linux 2 관리형 노드에서 AWS-AmazonLinux2DefaultPatchBaseline을 사용할 수 있으며 커널 라이브 업데이트가 적용됩니다.

> **Note**
>
> AWS-AmazonLinux2DefaultPatchBaseline 구성은 패치가 릴리스되거나 마지막으로 업데이트된 후 자동으로 설치되기 전에 7일의 대기 기간을 지정합니다. 7일 동안 커널 라이브 패치 자동 승인을 기다리지 않으려면 사용자 정의 패치 기준선을 생성하여 사용합니다. 패치 기준에서 자동 승인 대기 기간을 지정하거나 더 짧거나 긴 대기 기간을 지정할 수 있습니다. 자세한 내용은 사용자 정의 패치 기준 작업 단원을 참조하십시오.

커널 라이브 업데이트로 관리형 노드를 패치하려면 다음 전략을 사용하는 것이 좋습니다.

1. Amazon Linux 2 관리형 노드에서 Kernel Live Patching 활성화
2. AWS Systems Manager의 기능인 Run Command를 사용하여 미리 정의된 AWS-AmazonLinux2DefaultPatchBaseline 또는 심각도가 Critical과 Important로 분류되었고 Bugfix 심각도가 All인 Security 업데이트만 대상으로 하는 사용자 정의 패치 기준을 사용하여 관리형 노드에서 Scan 작업을 실행합니다.
3. AWS Systems Manager의 기능인 규정 준수(Compliance)를 사용하여 스캔된 관리형 노드에 대해 패치 규정 미준수가 보고되는지 검토합니다. 그렇다면 노드 규정 준수 세부 정보를 보고, 관리형 노드에서 누락된 커널 라이브 패치가 있는지 확인합니다.
4. 누락된 커널 라이브 패치를 설치하려면 이전에 지정한 것과 동일한 패치 기준에서 Run Command을 사용합니다. 하지만 이번에는 Scan 작업 대신 Install 작업을 실행합니다.

   커널 라이브 패치는 재부팅할 필요 없이 설치되므로 이 작업에 대해 NoReboot 재부팅 옵션을 선택할 수 있습니다.

> **Note**
>
> 노드에 설치된 다른 유형의 패치가 필요하거나 최신 커널로 업데이트하려는 경우에도 관리형 노드를 재부팅할 수 있습니다. 이러한 경우 `RebootIfNeeded` 재부팅 옵션을 대신 선택합니다.

5. 규정 준수로 돌아가서 커널 라이브 패치가 설치되었는지 확인합니다.

Run Command를 사용하여 Kernel Live Patching 설정

Kernel Live Patching을 설정하기 위해 관리형 노드에서 `yum` 명령을 실행하거나 Run Command와 생성한 사용자 정의 Systems Manager 문서(SSM 문서)를 사용할 수 있습니다.

관리형 노드에서 직접 `yum` 명령을 실행하여 Kernel Live Patching을 켜는 것에 대한 자세한 내용은 Amazon EC2 사용 설명서의 Kernel Live Patching 활성화를 참조하세요.

> **Note**
>
> 커널 라이브 패칭을 설정할 때 관리형 노드에서 이미 실행 중인 커널이 `kernel-4.14.165-131.185.amzn2.x86_64`(지원되는 최소 버전)보다 이전 버전이면 프로세스가 사용 가능한 최신 커널 버전을 설치하고 관리형 노드를 재부팅합니다. 노드가 `kernel-4.14.165-131.185.amzn2.x86_64` 이상을 이미 실행하고 있는 경우 프로세스가 최신 버전을 설치하지 않고 노드를 재부팅하지 않습니다.

Run Command를 사용하여 Kernel Live Patching을 설정하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택합니다.
3. Run command(Run 명령)를 선택합니다.
4. [Command 문서(Command document)] 목록에서 사용자 정의 SSM 문서 `AWS-ConfigureKernelLivePatching`을 선택합니다.
5. 명령 파라미터 섹션에서 이 작업의 일부로 관리형 노드를 재부팅할지 여부를 지정합니다.
6. 이 페이지의 나머지 컨트롤 작업에 대한 자세한 내용은 콘솔에서 명령 실행 섹션을 참조하세요.
7. Run(실행)을 선택합니다.

Kernel Live Patching을 설정하려면(AWS CLI)

- 로컬 시스템에서 다음 명령을 실행합니다.

  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"
  ```

  *instance-id*를 기능을 설정할 Amazon Linux 2 관리형 노드의 ID로 바꿉니다(예: i-02573cafcfEXAMPLE). 여러 관리형 노드에서 기능을 설정하기 위해 다음 형식 중 하나를 사용할 수 있습니다.

  - `--targets "Key=instanceids,Values=instance-id1,instance-id2"`
  - `--targets "Key=tag:tag-key,Values=tag-value"`

  명령에서 사용할 수 있는 기타 옵션에 대한 자세한 내용은 AWS CLI 명령 레퍼런스의 send-command 를 참조하세요.

Run Command을 사용하여 커널 라이브 패치 적용

커널 라이브 패치를 적용하기 위해 관리형 노드에서 yum 명령을 실행하거나, Run Command 및 SSM 문서 AWS-RunPatchBaseline을 사용할 수 있습니다.

관리형 노드에서 직접 yum 명령을 실행하여 커널 라이브 패치를 적용하는 것에 대한 자세한 내용은 Amazon EC2 사용 설명서의 커널 라이브 패치 적용을 참조하세요.

Run Command을 사용하여 커널 라이브 패치를 적용하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택합니다.
3. Run command(Run 명령)를 선택합니다.
4. [Command 문서(Command document)] 목록에서 SSM 문서 `AWS-RunPatchBaseline`을 선택합니다.
5. 명령 파라미터 섹션에서 다음 중 하나를 수행합니다.
   - 새 커널 라이브 패치가 사용 가능한지 확인하려는 경우 [작업(Operation)]에서 [Scan]을 선택합니다. 이 작업 후에 관리형 노드를 재부팅하지 않으려면 재부팅 옵션(Reboot Option)에서 `NoReboot`를 선택합니다. 작업이 완료되면 Compliance에서 새 패치 및 규정 준수 상태를 확인할 수 있습니다.
   - 패치 규정 준수를 이미 확인하고 사용 가능한 커널 라이브 패치를 적용할 준비가 된 경우 작업에서 `Install`를 선택합니다. 이 작업 후에 관리형 노드를 재부팅하지 않으려면 재부팅 옵션(Reboot Option)에서 `NoReboot`를 선택합니다.
6. 이 페이지의 나머지 컨트롤 작업에 대한 자세한 내용은 콘솔에서 명령 실행 섹션을 참조하세요.
7. Run(실행)을 선택합니다.

Run Command을 사용하여 커널 라이브 패치를 적용하려면(AWS CLI)

1. Compliance에서 결과를 확인하기 전에 `Scan` 작업을 수행하려면 로컬 시스템에서 다음 명령을 실행합니다.

   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
\"]}
```

명령에서 사용할 수 있는 기타 옵션에 대한 자세한 내용은 AWS CLI 명령 레퍼런스의 send-command 를 참조하세요.

2. Compliance에서 결과를 확인한 후 `Install` 작업을 수행하려면 로컬 시스템에서 다음 명령을 실행합니다.

   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\"]}
   ```

앞의 두 명령 모두에서 *instance-id*를 커널 라이브 패치를 적용할 Amazon Linux 2 관리형 노드의 ID로 바꿉니다(예: i-02573cafcfEXAMPLE). 여러 관리형 노드에서 기능을 설정하기 위해 다음 형식 중 하나를 사용할 수 있습니다.

- `--targets "Key=instanceids,Values=`*instance-id1*`,`*instance-id2*`"`
- `--targets "Key=tag:`*tag-key*`,Values=`*tag-value*`"`

이러한 명령에서 사용할 수 있는 기타 옵션에 대한 자세한 내용은 AWS CLI 명령 레퍼런스의 send-command 를 참조하세요.

Run Command를 사용하여 Kernel Live Patching을 해제하려면

Kernel Live Patching을 해제하기 위해 관리형 노드에서 `yum` 명령을 실행하거나, Run Command 및 사용자 정의 SSM 문서 `AWS-ConfigureKernelLivePatching`을 사용할 수 있습니다.

> **Note**
>
> 커널 라이브 패치를 더 이상 사용할 필요가 없는 경우 언제든지 해제할 수 있습니다. 대부분의 경우 이 기능을 해제할 필요가 없습니다.

관리형 노드에서 직접 yum 명령을 실행하여 Kernel Live Patching 설정을 비활성화하는 것에 대한 자세한 내용은 Amazon EC2 사용 설명서의 Kernel Live Patching 활성화를 참조하세요.

> **Note**
>
> Kernel Live Patching을 해제하면 프로세스가 Kernel Live Patching 플러그인을 제거하고 관리형 노드를 재부팅합니다.

Run Command를 사용하여 Kernel Live Patching을 해제하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택합니다.
3. Run command(Run 명령)를 선택합니다.
4. [Command 문서(Command document)] 목록에서 SSM 문서 `AWS-ConfigureKernelLivePatching`을 선택합니다.
5. 명령 파라미터 섹션에서 필요한 파라미터의 값을 지정합니다.
6. 이 페이지의 나머지 컨트롤 작업에 대한 자세한 내용은 콘솔에서 명령 실행 섹션을 참조하세요.
7. Run(실행)을 선택합니다.

Kernel Live Patching을 해제하려면(AWS CLI)

- 다음과 유사한 명령을 실행합니다.

  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"
```

*instance-id*를 기능을 비활성화할 Amazon Linux 2 관리형 노드의 ID로 바꿉니다(예: i-02573cafcfEXAMPLE). 여러 관리형 노드에서 기능을 비활성화하기 위해 다음 형식 중 하나를 사용할 수 있습니다.

- `--targets "Key=instanceids,Values=instance-id1,instance-id2"`
- `--targets "Key=tag:tag-key,Values=tag-value"`

명령에서 사용할 수 있는 기타 옵션에 대한 자세한 내용은 AWS CLI 명령 레퍼런스의 send-command 를 참조하세요.

Patch Manager 작업(콘솔)

AWS Systems Manager의 기능인 Patch Manager를 사용하려면 다음 태스크를 완료합니다. 이러한 작업은 이 단원에서 더 자세히 설명합니다.

1. 사용 중인 각 운영 체제 유형에 따라 AWS 미리 정의된 패치 기준이 필요를 충족하는지 확인하십시오. 그렇지 않은 경우 해당 관리형 노드 유형에 대한 표준 패치 집합을 정의하는 패치 기준을 생성하고, 기본값으로 설정합니다.
2. Amazon Elastic Compute Cloud(Amazon EC2) 태그를 사용하여 관리형 노드를 패치 그룹으로 구성합니다(옵션이지만 권장됨).
3. 다음 중 하나를 수행하십시오.
   - (권장) Systems Manager의 기능인 Quick Setup에서 패치 정책을 구성하여 전체 조직, 일부 조직 단위 또는 단일 AWS 계정의 일정에 따라 누락된 패치를 설치할 수 있습니다. 자세한 내용은 Patch Manager 조직 패치 적용 구성 단원을 참조하십시오.
   - Run Command 태스크 유형에서 Systems Manager 문서(SSM 문서) `AWS-RunPatchBaseline`을 사용하는 유지 관리 기간을 생성합니다. 자세한 내용은 패치에 대한 유지 관리 기간 생성(콘솔) 단원을 참조하십시오.

- Run Command 작업에서 `AWS-RunPatchBaseline`을 수동으로 실행합니다. 자세한 내용은 콘솔에서 명령 실행 단원을 참조하십시오.
- Patch now(지금 패치) 기능을 사용하여 필요에 따라 노드를 수동으로 패치합니다. 자세한 내용은 관리형 노드 온디맨드 패치 단원을 참조하십시오.

4. 패치 적용을 모니터링하여 규정 준수 여부를 확인하고 오류를 조사합니다.

주제

패치 정책 생성

패치 정책은 AWS Systems Manager의 기능인 Quick Setup을 사용하여 설정하는 구성입니다. 패치 정책은 다른 패치 구성 방법에서 제공되는 것보다 더 광범위하고 중앙 집중화된 패치 작업 제어 기능을 제공합니다. 패치 정책은 노드와 애플리케이션에 자동으로 패치를 적용할 때 사용할 일정과 기준선을 정의합니다.

자세한 정보는 다음 주제를 참조하십시오:

- Quick Setup 패치 정책 사용
- Patch Manager 조직 패치 적용 구성

패치 대시보드 요약 보기

Patch Manager의 대시보드(Dashboard) 탭에서는 통합 보기에서 패치 작업을 모니터링하는 데 사용할 수 있는 콘솔의 요약 보기를 제공합니다. Patch Manager는 AWS Systems Manager의 기능입니다. 대시보드(Dashboard) 탭에서 다음을 볼 수 있습니다.

- 패치 규정을 준수 및 준수하지 않는 관리형 노드의 수를 보여주는 스냅샷입니다.
- 관리형 노드에 대한 패치 규정 준수 결과 기간에 대한 스냅샷입니다.
- 규정 미준수의 가장 일반적인 사유 각각에 대해 규정을 준수하지 않는 관리형 노드의 연결 수입니다.
- 최신 패치 작업의 연결된 목록입니다.
- 설정된 반복 패치 작업의 연결된 목록입니다.

패치 대시보드 요약을 보려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Patch Manager를 선택합니다.
3. 대시보드(Dashboard) 탭을 선택합니다.
4. 보고자 하는 요약 데이터가 포함된 섹션으로 스크롤합니다.
    - Amazon EC2 인스턴스 관리
    - 규정 준수 요약
    - 미준수 건수
    - 규정 준수 보고서
    - 비패치 정책 기반 작업
    - 비패치 정책 기반 반복 태스크

패치 규정 준수 보고서 작업

다음 주제의 정보를 사용하여 패치 규정 준수 보고서를 AWS Systems Manager의 기능인 Patch Manager에 생성하고 작업할 수 있습니다.

다음 항목의 정보는 패치 작업에 사용하는 구성 방법 또는 유형에 관계없이 적용됩니다.

- Quick Setup에서 구성된 패치 정책
- Quick Setup에서 구성된 호스트 관리 옵션
- 패치 `Scan` 또는 `Install` 태스크를 실행하기 위한 유지 관리 기간
- 온디맨드 Patch now(지금 패치) 작업

> **Important**
>
> 인스턴스에서 패치 규정 준수를 검사하기 위해 여러 유형의 작업을 수행하는 경우, 스캔할 때마다 이전 스캔의 패치 규정 준수 데이터를 덮어씁니다. 따라서 패치 규정 준수 데이터에 예상치 못한 결과가 발생할 수 있습니다. 자세한 내용은 의도치 않은 패치 규정 준수 데이터 덮어쓰기 방지 단원을 참조하십시오.
>
> 최신 규정 준수 정보를 생성하는 데 어떤 패치 기준선이 사용되었는지 확인하려면, Patch Manager의 규정 준수 보고 탭으로 이동하여 정보를 얻으려는 관리형 노드에 해당하는 행을 찾은 다음, 사용된 기준선 ID 열에서 기준선 ID를 선택합니다.

주제

- 패치 규정 준수 결과 보기
- .csv 패치 규정 준수 보고서 생성
- Patch Manager로 규정 미준수 관리형 노드 문제 해결
- 의도치 않은 패치 규정 준수 데이터 덮어쓰기 방지

패치 규정 준수 결과 보기

다음 절차를 사용하여 관리형 노드에 대한 패치 규정 준수 정보를 봅니다.

이 절차는 `AWS-RunPatchBaseline` 문서를 사용하는 패치 작업에 적용됩니다. `AWS-RunPatchBaselineAssociation` 문서를 사용하는 패치 작업에 대한 패치 규정 준수 정보 보기에 대한 자세한 내용은 규정 미준수 관리형 노드 식별 섹션을 참조하세요.

> **Note**
>
> Quick Setup 및 Explorer에 대한 패치 검사 작업에서는 `AWS-RunPatchBaselineAssociation` 문서를 사용합니다. Quick Setup 및 Explorer는 둘 다 AWS Systems Manager의 기능입니다.

특정 CVE 문제에 대한 패치 솔루션 식별(Linux)

많은 Linux 기반 운영 체제의 경우 패치 규정 준수 결과에 따라 어느 일반적인 취약성 및 노출도(CVE) 게시판 문제가 어느 패치로 해결되는지 확인할 수 있습니다. 이 정보는 누락되거나 실패한 패치를 얼마나 긴급하게 설치해야 하는지를 결정하는 데 도움이 됩니다.

다음 운영 체제 유형의 지원되는 버전에 대한 CVE 세부 정보가 포함되어 있습니다.

- 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**
>
> 기본적으로 CCentOS와 CentOS Stream은 업데이트에 대한 CVE 정보를 제공하지 않습니다. 그러나 Fedora가 게시하는 EPEL(Extra Packages for Enterprise Linux) 리포지토리와 같은 서드 파티 리포지토리를 사용하여 이러한 지원을 허용할 수 있습니다. 자세한 내용은 Fedora Wiki의 EPEL을 참조하세요.
>
> 현재 CVE ID 값은 `Missing` 또는 `Failed` 상태인 패치에 대해서만 보고됩니다.

상황과 패치 목표가 보장하는 대로 패치 기준의 승인 또는 거부된 패치 목록에 CVE ID를 추가할 수도 있습니다.

승인 및 거부된 패치 목록 작업에 대한 자세한 내용은 다음 주제를 참조하세요.

- 사용자 정의 패치 기준 작업
- 승인 패치 및 거부 패치 목록의 패키지 이름 형식 정보
- Linux 기반 시스템에서 패치 기준 규칙 사용 방법
- 패치 설치 방법

> **Note**
>
> 경우에 따라 Microsoft는 업데이트된 날짜와 시간을 지정하지 않는 애플리케이션에 대한 패치를 릴리스합니다. 이 경우 업데이트된 `01/01/1970`의 날짜 및 시간은 기본적으로 제공됩니다.

패치 규정 준수 결과 보기

AWS Systems Manager 콘솔에서 패치 규정 준수 결과를 보려면 다음 절차를 따릅니다.

> **Note**
>
> Amazon Simple Storage Service(Amazon S3) 버킷에 다운로드되는 패치 규정 준수 보고서 생성에 대한 자세한 내용은 .csv 패치 규정 준수 보고서 생성 섹션을 참조하세요.

패치 규정 준수 결과를 보려면

1. 다음 중 하나를 수행하세요.

   옵션 1(권장) - AWS Systems Manager의 기능인 Patch Manager에서 이동합니다.

   - 탐색 창에서 Patch Manager를 선택합니다.
   - 규정 준수 보고(Compliance reporting) 탭을 선택합니다.
   - 노드 패치 적용 세부 정보에서 패치 규정 준수 결과를 검토할 관리형 노드의 노드 ID를 선택합니다.
   - 세부 정보 영역의 속성 목록에서 패치를 선택합니다.

   옵션 2 - AWS Systems Manager의 기능인 Compliance에서 이동합니다.

   - 탐색 창에서 [Compliance]를 선택합니다.
   - [Compliance 리소스 요약(Compliance resources summary)]에서 검토하려는 패치 리소스 유형의 열(예: [규정 미준수 리소스(Non-Compliant resources)])에서 숫자를 선택합니다.
   - 아래의 리소스 목록에서 패치 규정 준수 결과를 검토할 관리형 노드의 ID를 선택합니다.
   - 세부 정보 영역의 속성 목록에서 패치를 선택합니다.

   옵션 3 - AWS Systems Manager의 기능인 Fleet Manager에서 이동합니다.

- 탐색 창에서 Fleet Manager를 선택합니다.
- 관리형 인스턴스 영역에서 패치 규정 준수 결과를 검토할 관리형 노드의 ID를 선택합니다.
- 세부 정보 영역의 속성 목록에서 패치를 선택합니다.

2. (옵션) 검색 상자
()
에서 사용 가능한 필터 중에서 선택합니다.

   예를 들어 Red Hat Enterprise Linux(RHEL)의 경우 다음 중에서 선택합니다.

   - 명칭
   - 분류
   - State
   - 심각도

   Windows Server에 대해 다음 중에서 선택합니다.

   - KB
   - 분류
   - State
   - 심각도

3. 선택한 필터 유형에 사용할 수 있는 값 중 하나를 선택합니다. 예를 들어 [상태(State)]를 선택한 경우 [InstalledPendingReboot], [실패(Failed)], [누락(Missing)] 등의 규정 준수 상태를 선택합니다.

   > **Note**
   >
   > 현재 CVE ID 값은 `Missing` 또는 `Failed` 상태인 패치에 대해서만 보고됩니다.

4. 관리형 노드의 규정 준수 상태에 따라 규정을 준수하지 않는 노드를 수정하기 위해 수행할 작업을 선택할 수 있습니다.

   예를 들어 미준수 관리형 노드를 즉시 패치하도록 선택할 수 있습니다. 온디맨드로 관리형 노드 패치에 대한 자세한 내용은 관리형 노드 온디맨드 패치 섹션을 참조하세요.

   패치 규정 준수 상태에 대한 자세한 내용은 패치 규정 준수 상태 값 이해 섹션을 참조하세요.

.csv 패치 규정 준수 보고서 생성

AWS Systems Manager 콘솔을 사용하여 선택한 Amazon Simple Storage Service(Amazon S3) 버킷에 .csv 파일로 저장되는 패치 규정 준수 보고서를 생성할 수 있습니다. 단일 온디맨드 보고서를 생성하거나 보고서를 자동으로 생성하는 일정을 지정할 수 있습니다.

단일 관리형 노드 또는 선택한 모든 AWS 계정 및 AWS 리전에 대해 보고서를 생성할 수 있습니다. 단일 노드의 경우 보고서의 경우 규정을 준수하지 않는 노드와 관련된 패치의 ID를 비롯한 포괄적인 세부 정보가 포함됩니다. 모든 관리형 노드에 대한 보고서의 경우 비준수 노드의 패치 수와 요약 정보만 제공됩니다.

보고서가 생성된 후 Amazon QuickSight와 같은 도구를 사용하여 데이터를 가져오고 분석할 수 있습니다. Amazon QuickSight는 대화형 시각적 환경에서 정보를 탐색하고 해석하는 데 사용할 수 있는 비즈니스 인텔리전스(BI) 서비스입니다. 자세한 내용은 Amazon QuickSight User Guide를 참조하세요.

> **Note**
>
> 사용자 지정 패치 기준선을 생성할 때 해당 패치 기준선에서 승인된 패치의 규정 준수 심각도 수준을 지정할 수 있습니다(예: `Critical` 또는 `High`). 승인된 패치의 패치 상태가 `Missing`으로 보고되는 경우 패치 기준선의 전반적인 보고된 규정 준수 심각도는 사용자가 지정한 심각도 수준입니다.

보고서가 생성될 때 알림을 보내는 데 사용할 Amazon Simple Notification Service(Amazon SNS) 주제를 지정할 수도 있습니다.

패치 규정 준수 보고서 생성을 위한 서비스 역할

보고서를 처음 생성할 때 Systems Manager에서는 S3로 내보내기 프로세스에 사용할 `AWS-SystemsManager-PatchSummaryExportRole`이라는 Automation 수임 역할을 생성합니다.

> **Note**
>
> 규정 준수 데이터를 암호화된 S3 버킷으로 내보내는 경우 연결된 AWS KMS 키 정책을 업데이트하여 필요한 `AWS-SystemsManager-PatchSummaryExportRole` 권한을 제공해야 합니다. 인스턴스의 경우 이와 비슷한 권한을 S3 버킷의 AWS KMS 정책에 추가합니다.
>
> ```
> {
>     "Effect": "Allow",
>     "Action": [
> ```

```
            "kms:GenerateDataKey"
        ],
        "Resource": "role-arn"
    }
```

*role-arn*을 계정에서 생성된 Amazon 리소스 이름(ARN)으로 바꿉니다(형식: arn:aws:iam::*111222333444*:role/service-role/AWS-SystemsManager-PatchSummaryExportRole).
자세한 내용은 AWS Key Management Service 개발자 안내서의 AWS KMS의 키 정책을 참조하세요.

일정에 따라 보고서를 처음 생성할 때 Systems Manager는 내보내기 프로세스에 사용할 서비스 역할 AWS-SystemsManager-PatchSummaryExportRole(아직 생성되지 않은 경우)과 함께 AWS-EventBridge-Start-SSMAutomationRole이라는 다른 서비스 역할을 생성합니다. AWS-EventBridge-Start-SSMAutomationRole은 Amazon EventBridge가 실행서 AWS-ExportPatchReportToS3을 사용하여 자동화를 시작할 수 있도록 합니다.

이러한 정책 및 역할을 수정하지 않는 것이 좋습니다. 이렇게 하면 패치 규정 준수 보고서 생성이 실패할 수 있습니다. 자세한 내용은 패치 규정 준수 보고서 생성 문제 해결 단원을 참조하십시오.

주제

생성된 패치 규정 준수 보고서에는 무엇이 포함되어 있나요?

이 주제에서는 지정된 S3 버킷에 생성되어 다운로드되는 패치 규정 준수 보고서에 포함된 콘텐츠 유형에 대한 정보를 제공합니다.

단일 관리형 노드에 대한 보고서 형식

단일 관리형 노드에 대해 생성된 보고서는 요약 정보와 세부 정보를 모두 제공합니다.

샘플 보고서 다운로드(단일 노드)

단일 관리형 노드에 대한 요약 정보에는 다음이 포함됩니다.

- 인덱스
- 인스턴스 ID
- 인스턴스 이름
- 인스턴스 IP
- 플랫폼 이름
- 플랫폼 버전
- SSM Agent 버전
- 패치 기준
- 패치 그룹
- 규정 준수 상태
- 규정 준수 심각도
- 규정 미준수 중요 심각도 패치 수
- 규정 미준수 높음 심각도 패치 수
- 규정 미준수 중간 심각도 패치 수
- 규정 미준수 낮음 심각도 패치 수
- 규정 미준수 정보 심각도 패치 수
- 규정 미준수 지정되지 않음 심각도 패치 수

단일 관리형 노드에 대한 세부 정보에는 다음이 포함됩니다.

- 인덱스
- 인스턴스 ID
- 인스턴스 이름
- 패치 이름
- KB ID/패치 ID
- 패치 상태
- 최근 보고 시간
- Compliance 수준

- 패치 심각도
- 패치 분류
- CVE ID
- 패치 기준
- 로그 URL
- 인스턴스 IP
- 플랫폼 이름
- 플랫폼 버전
- SSM Agent 버전

> **Note**
>
> 사용자 지정 패치 기준선을 생성할 때 해당 패치 기준선에서 승인된 패치의 규정 준수 심각도 수준을 지정할 수 있습니다(예: `Critical` 또는 `High`). 승인된 패치의 패치 상태가 `Missing`으로 보고되는 경우 패치 기준선의 전반적인 보고된 규정 준수 심각도는 사용자가 지정한 심각도 수준입니다.

모든 관리형 노드에 대한 보고서 형식

모든 관리형 노드에 대해 생성된 보고서는 요약 정보만 제공합니다.

샘플 보고서 다운로드(모든 관리형 노드)

모든 관리형 노드에 대한 요약 정보에는 다음이 포함됩니다.

- 인덱스
- 인스턴스 ID
- 인스턴스 이름
- 인스턴스 IP
- 플랫폼 이름
- 플랫폼 버전
- SSM Agent 버전
- 패치 기준
- 패치 그룹

- 규정 준수 상태
- 규정 준수 심각도
- 규정 미준수 중요 심각도 패치 수
- 규정 미준수 높음 심각도 패치 수
- 규정 미준수 중간 심각도 패치 수
- 규정 미준수 낮음 심각도 패치 수
- 규정 미준수 정보 심각도 패치 수
- 규정 미준수 지정되지 않음 심각도 패치 수

단일 관리형 노드에 대한 패치 규정 준수 보고서 생성

다음 절차에 따라 AWS 계정의 단일 관리형 노드에 대한 패치 요약 보고서를 생성합니다. 단일 관리형 노드에 대한 보고서는 패치 이름 및 ID를 포함하여 규정을 위반하는 각 패치에 대한 세부 정보를 제공합니다.

단일 관리형 노드에 대한 패치 규정 준수 보고서 생성

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Patch Manager를 선택합니다.
3. 규정 준수 보고(Compliance reporting) 탭을 선택합니다.
4. 보고서를 생성할 관리형 노드의 행에 대한 버튼을 선택한 다음 세부 정보 보기(View detail)를 선택합니다.
5. 패치 요약(Patch summary) 섹션에서 S3로 내보내기(Export to S3)를 선택합니다.
6. [보고서 이름(Report name)]에 나중에 보고서를 식별하는 데 도움이 되는 이름을 입력합니다.
7. [보고 빈도(Reporting frequency)]에서 다음 중 하나를 선택합니다.
   - [온디맨드(On demand)] - 일회성 보고서를 생성합니다. 9단계로 건너뜁니다.
   - [일정에 따라(On a schedule)] - 보고서를 자동으로 생성하기 위한 반복 일정을 지정합니다. 계속해서 8단계를 진행합니다.
8. [일정 유형(Schedule type)]에서 3일마다와 같은 비율 표현식을 지정하거나 cron 표현식을 제공하여 보고서 빈도를 설정합니다.

   Cron 표현식에 대한 자세한 내용은 참조: Systems Manager용 Cron 및 Rate 표현식 섹션을 참조하세요.
9. [버킷 이름(Bucket name)]에서 .csv 보고서 파일을 저장할 S3 버킷의 이름을 선택합니다.

> **Important**
>
> 2019년 3월 20일 이후에 시작된 AWS 리전에서 작업하는 경우 동일한 리전의 S3 버킷을 선택해야 합니다. 이 날짜 이후에 시작된 리전은 기본적으로 해제되어 있습니다. 자세한 내용과 이러한 리전의 목록은 Amazon Web Services 일반 참조의 리전 활성화를 참조하세요.

10. (옵션) 보고서가 생성될 때 알림을 보내려면 다음 위치에서 SNS 주제(SNS topic) 섹션을 확장하고, Amazon Resource Name(ARN)에서 기존 Amazon SNS 주제를 선택합니다.
11. 제출을 선택합니다.

생성된 보고서의 기록 보기에 대한 자세한 내용은 패치 규정 준수 보고 기록 보기 섹션을 참조하세요.

생성한 보고 일정의 세부 정보 보기에 대한 자세한 내용은 패치 규정 준수 보고 일정 보기 섹션을 참조하세요.

단일 관리형 노드에 대한 패치 규정 준수 보고서 생성

다음 절차에 따라 AWS 계정의 모든 관리형 노드에 대한 패치 요약 보고서를 생성합니다. 모든 관리형 노드에 대한 보고서에는 규정을 위반하는 노드와 규정 미준수 패치 수가 나와 있습니다. 패치의 이름이나 다른 식별자는 제공하지 않습니다. 이러한 추가 세부 정보를 보려면 단일 관리형 노드에 대한 패치 규정 준수 보고서를 생성합니다. 자세한 내용은 이번 주제 전반부의 단일 관리형 노드에 대한 패치 규정 준수 보고서 생성 섹션을 참조하세요.

모든 관리형 노드에 대한 패치 규정 준수 보고서 생성

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Patch Manager를 선택합니다.
3. 규정 준수 보고(Compliance reporting) 탭을 선택합니다.
4. [S3로 내보내기(Export to S3)]를 선택합니다. (노드 ID를 먼저 선택하지 마세요.)
5. [보고서 이름(Report name)]에 나중에 보고서를 식별하는 데 도움이 되는 이름을 입력합니다.
6. [보고 빈도(Reporting frequency)]에서 다음 중 하나를 선택합니다.
   - [온디맨드(On demand)] - 일회성 보고서를 생성합니다. 8단계로 건너뜁니다.
   - [일정에 따라(On a schedule)] - 보고서를 자동으로 생성하기 위한 반복 일정을 지정합니다. 계속해서 7단계를 진행합니다.

7. [일정 유형(Schedule type)]에서 3일마다와 같은 비율 표현식을 지정하거나 cron 표현식을 제공하여 보고서 빈도를 설정합니다.

   Cron 표현식에 대한 자세한 내용은 참조: Systems Manager용 Cron 및 Rate 표현식 섹션을 참조하세요.

8. [버킷 이름(Bucket name)]에서 .csv 보고서 파일을 저장할 S3 버킷의 이름을 선택합니다.

   > **Important**
   >
   > 2019년 3월 20일 이후에 시작된 AWS 리전에서 작업하는 경우 동일한 리전의 S3 버킷을 선택해야 합니다. 이 날짜 이후에 시작된 리전은 기본적으로 해제되어 있습니다. 자세한 내용과 이러한 리전의 목록은 Amazon Web Services 일반 참조의 리전 활성화를 참조하세요.

9. (옵션) 보고서가 생성될 때 알림을 보내려면 다음 위치에서 SNS 주제(SNS topic) 섹션을 확장하고, Amazon Resource Name(ARN)에서 기존 Amazon SNS 주제를 선택합니다.
10. 제출을 선택합니다.

생성된 보고서의 기록 보기에 대한 자세한 내용은 패치 규정 준수 보고 기록 보기 섹션을 참조하세요.

생성한 보고 일정의 세부 정보 보기에 대한 자세한 내용은 패치 규정 준수 보고 일정 보기 섹션을 참조하세요.

패치 규정 준수 보고 기록 보기

이 주제의 정보를 사용하여 AWS 계정에서 생성된 패치 규정 준수 보고서에 대한 세부 정보를 볼 수 있습니다.

패치 규정 준수 보고 기록을 보려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Patch Manager를 선택합니다.
3. 규정 준수 보고(Compliance reporting) 탭을 선택합니다.
4. [모든 S3 내보내기 보기(View all S3 exports)]를 선택하고 [내보내기 기록(Export history)] 탭을 선택합니다.

패치 규정 준수 보고 일정 보기

이 주제의 정보를 사용하여 AWS 계정에서 생성된 패치 규정 준수 보고 일정에 대한 세부 정보를 볼 수 있습니다.

패치 규정 준수 보고 기록을 보려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Patch Manager를 선택합니다.
3. 규정 준수 보고(Compliance reporting) 탭을 선택합니다.
4. 모든 S3 내보내기 보기(View all S3 exports)를 선택하고 보고서 예약 역할(Report schedule rules) 탭을 선택합니다.

패치 규정 준수 보고서 생성 문제 해결

다음 정보를 사용하여 AWS Systems Manager의 기능인 Patch Manager에서 패치 규정 준수 보고서 생성 문제를 해결할 수 있습니다.

주제

- AWS-SystemsManager-PatchManagerExportRolePolicy 정책이 손상되었다는 메시지가 나타납니다.
- 패치 규정 준수 정책 또는 역할을 삭제한 후 예약된 보고서가 성공적으로 생성되지 않습니다.

AWS-SystemsManager-PatchManagerExportRolePolicy 정책이 손상되었다는 메시지가 나타납니다.

문제: `AWS-SystemsManager-PatchManagerExportRolePolicy`가 손상되었음을 나타내는 다음과 비슷한 오류 메시지가 나타납니다.

```
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.
```

- 솔루션: 새 패치 규정 준수 보고서를 생성하기 전에 Patch Manager 콘솔 또는 AWS CLI를 사용하여 영향을 받는 역할과 정책을 삭제합니다.

콘솔을 사용하여 손상된 정책을 삭제하는 방법

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 다음 중 하나를 수행하십시오.

   온디맨드 보고서 - 일회성 온디맨드 보고서를 생성하는 동안 문제가 발생한 경우 왼쪽 탐색에서 [정책(Policies)]을 선택하고 AWS-SystemsManager-PatchManagerExportRolePolicy를 검색한 다음 정책을 삭제합니다. 그런 다음 [역할(Roles)]을 선택하고 AWS-SystemsManager-PatchSummaryExportRole을 검색한 다음 역할을 삭제합니다.

   예약된 보고서 - 일정에 따라 보고서를 생성하는 동안 문제가 발생한 경우 왼쪽 탐색에서 정책을 선택하고, 한 번에 하나씩 AWS-EventBridge-Start-SSMAutomationRolePolicy 및 AWS-SystemsManager-PatchManagerExportRolePolicy를 검색하고, 각 정책을 삭제합니다. 그런 다음 [역할(Roles)]을 선택하고 한 번에 하나씩 AWS-EventBridge-Start-SSMAutomationRole 및 AWS-SystemsManager-PatchSummaryExportRole을 검색한 다음 각 역할을 삭제합니다.

AWS CLI를 사용하여 손상된 정책을 삭제하는 방법

*## ### #*을 계정 ID로 바꿉니다.

- 일회성 온디맨드 보고서를 생성하는 동안 문제가 발생한 경우 다음과 같은 명령을 실행합니다.

```
aws iam delete-policy --policy-arn arn:aws:iam::account-id:policy/AWS-
SystemsManager-PatchManagerExportRolePolicy
```

```
aws iam delete-role --role-name AWS-SystemsManager-PatchSummaryExportRole
```

일정에 따라 보고서를 생성하는 동안 문제가 발생한 경우 다음과 같은 명령을 실행합니다.

```
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
```

두 절차 중 하나를 완료한 후 단계에 따라 새 패치 규정 준수 보고서를 생성하거나 예약합니다.

패치 규정 준수 정책 또는 역할을 삭제한 후 예약된 보고서가 성공적으로 생성되지 않습니다.

문제: 보고서를 처음 생성할 때 Systems Manager는 내보내기 프로세스에 사용할 서비스 역할과 정책을 생성합니다(`AWS-SystemsManager-PatchSummaryExportRole` 및 `AWS-SystemsManager-PatchManagerExportRolePolicy`). 일정에 따라 보고서를 처음 생성할 때 Systems Manager는 다른 서비스 역할과 정책을 생성합니다(`AWS-EventBridge-Start-SSMAutomationRole` 및 `AWS-EventBridge-Start-SSMAutomationRolePolicy`). 이를 통해 Amazon EventBridge는 실행서 AWS-ExportPatchReportToS3을 사용하여 자동화를 시작할 수 있습니다.

이러한 정책 또는 역할을 삭제하면 일정과 지정된 S3 버킷 및 Amazon SNS 주제 간의 연결이 끊어질 수 있습니다.

- 해결 방법: 이 문제를 해결하려면 이전 일정을 삭제하고 문제가 발생한 일정을 대체할 새 일정을 만드는 것이 좋습니다.

Patch Manager로 규정 미준수 관리형 노드 문제 해결

이 섹션의 주제에서는 패치 규정을 위반하는 관리형 노드를 식별하는 방법과 노드가 규정을 준수하도록 하는 방법을 간략히 소개합니다.

주제

- 규정 미준수 관리형 노드 식별
- 패치 규정 준수 상태 값 이해
- 규정 미준수 관리형 노드 패치 적용

규정 미준수 관리형 노드 식별

2개의 AWS Systems Manager 문서(SSM 문서) 중 하나를 실행하면 규정 위반 관리형 노드가 식별됩니다. 이들 SSM 문서는 AWS Systems Manager의 기능인 Patch Manager의 각 관리형 노드에 대한 적

절한 패치 기준을 참조합니다. 그런 다음 관리형 노드의 패치 상태를 평가하고 규정 준수 결과를 제공합니다.

규정 미준수 관리형 노드를 식별하거나 업데이트하는 데 사용하는 두 가지 SSM 문서가 있습니다(`AWS-RunPatchBaseline` 및 `AWS-RunPatchBaselineAssociation`). 각 문서는 서로 다른 프로세스에서 사용되며 규정 준수 결과는 여러 채널을 통해 제공됩니다. 다음 표에는 이러한 문서 간의 차이점이 요약되어 있습니다.

> **Note**
>
> Patch Manager에서 패치 규정 준수 데이터를 AWS Security Hub로 보낼 수 있습니다. Security Hub에서는 우선순위가 높은 보안 알림 및 규정 준수 상태를 포괄적으로 파악할 수 있습니다. 또한 플릿의 패치 상태를 모니터링합니다. 자세한 내용은 Patch Manager와 AWS Security Hub 통합 단원을 참조하십시오.

	AWS-RunPatchBaseline	**AWS-RunPatchBaselineAssociation**
문서를 사용하는 프로세스	온디맨드로 패치 - 지금 패치(Patch now) 옵션을 사용하여 온디맨드로 관리형 노드를 스캔하거나 패치할 수 있습니다. 자세한 설명은 관리형 노드 온디맨드 패치을 참조하세요. Systems Manager Quick Setup 패치 정책 - AWS Systems Manager의 기능인 Quick Setup에서 전체 조직, 일부 조직 단위 또는 단일 AWS 계정에 대해 별도의 일정에 따라 누락된 패치를 검색하거나 설치할 수 있는 패치 구성 기능을 생성할 수 있습니다. 자세한 설명은 Patch Manager 조직 패치 적용 구성을 참조하세요.	Systems Manager Quick Setup 호스트 관리 - Quick Setup에서 Systems Manager 호스트 관리 구성 옵션을 활성화하여 관리형 인스턴스의 패치 규정 준수 여부를 검사할 수 있습니다. 자세한 설명은 Amazon EC2 호스트 관리을 참조하세요. Systems Manager Explorer - AWS Systems Manager의 기능인 Explorer를 허용하면 정기적으로 관리형 인스턴스의 패치 규정 준수 여부를 검사하여 Explorer 대시보드에 결과를 보고합니다.

	AWS-RunPatchBaseline	AWS-RunPatchBaselineAssociation
	명령 실행 - AWS Systems Manager의 기능인 Run Command의 작업에서 AWS-RunPatchBaseline 을 수동으로 실행할 수 있습니다. 자세한 설명은 콘솔에서 명령 실행을 참조하세요. 유지 관리 기간 - Run Command 태스크 유형에서 SSM 문서 AWS-RunPatchBaseline 을 사용하는 유지 관리 기간을 생성할 수 있습니다. 자세한 설명은 패치에 대한 유지 관리 기간 생성(콘솔)을 참조하세요.	
패치 검사 결과 데이터의 형식	AWS-RunPatchBaseline 실행 후 Patch Manager가 AWS Systems Manager의 기능인 Inventory에 AWS:PatchSummary 객체를 전송합니다.	AWS-RunPatchBaselineAssociation 실행 후 Patch Manager가 Systems Manager Inventory에 AWS:ComplianceItem 객체를 전송합니다.

	AWS-RunPatchBaseline	AWS-RunPatchBaselineAssociation
콘솔에서 패치 규정 준수 보고서 보기	Systems Manager Configuration Compliance 및 관리형 노드 작업에서 AWS-RunPatchBaseline 을 사용하는 프로세스에 대한 패치 규정 준수 정보를 볼 수 있습니다. 자세한 내용은 패치 규정 준수 결과 보기 단원을 참조하십시오.	Quick Setup을 사용하여 관리형 인스턴스의 패치 규정 준수 여부를 검사하는 경우 Quick Setup의 [결과 보기(View results)] 버튼을 사용하여 액세스할 수 있는 Systems Manager State Manager에서 규정 준수 보고서를 볼 수 있습니다. Explorer를 사용하여 관리형 인스턴스의 패치 규정 준수 여부를 검사하는 경우 Explorer와 Systems Manager OpsCenter 모두에서 규정 준수 보고서를 볼 수 있습니다.
패치 규정 준수 결과를 보기 위한 AWS CLI 명령	AWS-RunPatchBaseline 을 사용하는 프로세스의 경우 다음 AWS CLI 명령을 사용하여 관리형 노드의 패치에 대한 요약 정보를 볼 수 있습니다. • describe-instance-patch-states • describe-instance-patch-states-for-patch-group • describe-patch-group-state	AWS-RunPatchBaselineAssociation 을 사용하는 프로세스의 경우 다음 AWS CLI 명령을 사용하여 인스턴스의 패치에 대한 요약 정보를 볼 수 있습니다. • list-compliance-items

	AWS-RunPatchBaseline	AWS-RunPatchBaselineAssociation
패치 작업	AWS-RunPatchBaseline 을 사용하는 프로세스의 경우 작업에서 Scan 작업만 실행할지 아니면 Scan and install 작업을 실행할지 지정합니다. (규정 미준수 관리형 노드를 식별하고 문제를 해결하지는 않는 것이 목표라면 Scan 작업만 실행합니다.)	AWS-RunPatchBaselineAssociation 을 사용하는 Quick Setup 및 Explorer 프로세스는 Scan 작업만 실행합니다.
추가 정보	AWS-RunPatchBaseline SSM 문서 정보	AWS-RunPatchBaselineAssociation SSM 문서 정보

보고되는 다양한 패치 규정 준수 상태에 대한 자세한 내용은 패치 규정 준수 상태 값 이해 섹션을 참조하세요.

패치 규정을 위반하는 관리형 노드 수정에 대한 자세한 내용은 규정 미준수 관리형 노드 패치 적용 섹션을 참조하세요.

패치 규정 준수 상태 값 이해

관리형 노드의 패치에 대한 정보에는 각 개별 패치의 상태에 대한 보고서가 포함됩니다.

> **Note**
>
> 관리형 노드에 특정 패치 규정 준수 상태를 할당하려면 put-compliance-items AWS Command Line Interface(AWS CLI) 명령 또는 PutComplianceItems API 작업을 사용합니다. 콘솔에서는 규정 준수 상태를 할당할 수 없습니다.

다음 표의 정보를 사용하여 관리형 노드가 패치 규정을 위반하는 이유를 식별할 수 있습니다.

Debian Server, Raspberry Pi OS 및 Ubuntu Server의 패치 규정 준수 값

Debian Server, Raspberry Pi OS 및 Ubuntu Server의 경우 다양한 규정 준수 상태로 패키지를 분류하는 규칙은 다음 표에 설명되어 있습니다.

> **Note**
>
> 설치됨, 설치됨 기타 및 누락됨 상태 값을 평가할 때 다음 사항을 유의하세요. 패치 기준선을 생성하거나 업데이트할 때 비보안 업데이트 포함 확인란을 선택하지 않은 경우 패치 후보 버전은 `trusty-security` (Ubuntu Server 14.04 LTS), `xenial-security` (Ubuntu Server 16.04 LTS), `bionic-security` (Ubuntu Server 18.04 LTS), `focal-security` (Ubuntu Server 20.04 LTS), `groovy-security` (Ubuntu Server 20.10 STR), `jammy-security` (Ubuntu Server 22.04 LTS) 또는 `debian-security` (Debian Server 및 Raspberry Pi OS)에 포함된 패치로 제한됩니다. [비보안 업데이트 포함(Include nonsecurity updates)] 확인란을 선택하면 다른 리포지토리의 패치도 고려됩니다.

패치 상태	설명	규정 준수 상태
`INSTALLED`	패치가 패치 기준에 나열되고 관리형 노드에 설치됩니다. 개인이 수동으로 설치하거나 `AWS-RunPatchBaseline` 문서가 관리형 노드에서 실행되었을 때 개인이나 Patch Manager가 자동으로 설치했을 수 있습니다.	규정 준수
`INSTALLED_OTHER`	패치가 기준에 포함되지 않았거나 기준에서 승인되지 않았지만 관리형 노드에 설치되었습니다. 패치가 수동으로 설치되었거나 패키지가 승인된 다른 패치의 필수 종속성이거나 패치가 InstallOverrideList 작업에 포함되었을 수 있습니다. [거부된 패치(Rejected patches)]	규정 준수

패치 상태	설명	규정 준수 상태
	작업으로 Block을 지정하지 않으면 Installed_Other 패치에는 설치되었지만 거부된 패치도 포함됩니다.	
INSTALLED_PENDING_REBOOT	INSTALLED_PENDING_REBOOT 는 다음 중 하나를 의미할 수 있습니다. • Patch Manager Install 작업에서 관리형 노드에 패치를 적용했지만 패치가 적용된 이후로 노드가 재부팅되지 않았습니다. 이는 일반적으로 AWS-RunPatchBaseline 문서가 관리형 노드에서 마지막으로 실행되었을 때 RebootOption 파라미터에 대해 NoReboot 옵션이 선택되었음을 의미합니다. 자세한 내용은 파라미터 이름: RebootOption 단원을 참조하십시오. • 관리형 노드를 마지막으로 재부팅한 이후 Patch Manager 외부에 패치가 설치되었습니다.	규정 미준수

패치 상태	설명	규정 준수 상태
INSTALLED_REJECTED	패치가 관리형 노드에 설치되었지만 거부된 패치(Rejected patches) 목록에 지정되었습니다. 이는 일반적으로 패치가 거부된 패치 목록에 추가되기 이전에 설치된 경우에 해당합니다.	규정 미준수
MISSING	기준을 통해 필터링되고 아직 설치되지 않은 패키지입니다.	규정 미준수
FAILED	패치 작업 중에 설치에 실패한 패키지입니다.	규정 미준수

다른 운영 체제의 패치 규정 준수 값

Debian Server, Raspberry Pi OS 및 Ubuntu Server 외에 모든 운영 체제의 경우 다양한 규정 준수 상태로 패키지를 분류하는 규칙은 다음 표에 설명되어 있습니다.

패치 상태	설명	Compliance 값
INSTALLED	패치가 패치 기준에 나열되고 관리형 노드에 설치됩니다. 개인이 수동으로 설치하거나 `AWS-RunPatchBaseline` 문서가 노드에서 실행되었을 때 개인이나 Patch Manager가 자동으로 설치했을 수 있습니다.	규정 준수
INSTALLED_OTHER [1]	패치가 기준에 없지만 관리형 노드에 설치되었습니다. 패치가 수동으로 설치되었거나 패키지가 다른 승인된 패치의 필수 종속성일 수 있습니다. [거부	규정 준수

패치 상태	설명	Compliance 값
	된 패치(Rejected patches)] 작업으로 `Block`을 지정하지 않으면 `Installed_Other` 패치에는 설치되었지만 거부된 패치도 포함됩니다.	
`INSTALLED_REJECTED`	패치가 관리형 노드에 설치되었지만 거부된 패치(Rejected patches) 목록에 지정되었습니다. 이는 일반적으로 패치가 거부된 패치 목록에 추가되기 이전에 설치된 경우에 해당합니다.	규정 미준수

패치 상태	설명	Compliance 값
INSTALLED_PENDING_REBOOT	INSTALLED_PENDING_REBOOT 는 다음 중 하나를 의미할 수 있습니다. • Patch Manager Install 작업에서 관리형 노드에 패치를 적용했지만 패치가 적용된 이후로 노드가 재부팅되지 않았습니다. 이는 일반적으로 AWS-RunPatchBaseline 문서가 관리형 노드에서 마지막으로 실행되었을 때 RebootOption 파라미터에 대해 NoReboot 옵션이 선택되었음을 의미합니다. 자세한 내용은 파라미터 이름: RebootOption 단원을 참조하십시오. • 관리형 노드를 마지막으로 재부팅한 이후 Patch Manager 외부에 패치가 설치되었습니다.	규정 미준수
MISSING	패치가 기준에 승인되었지만 관리형 노드에 설치되지 않았습니다. AWS-RunPatchBaseline 문서 태스크를 (설치 대신) 검사하도록 구성하는 경우 시스템은 검사 도중에 있었지만 설치되지는 않은 패치에 대해 이 상태를 보고합니다.	규정 미준수

패치 상태	설명	Compliance 값
NOT_APPLICABLE [1]	패치가 기준에 승인되었지만 패치를 사용하는 서비스 또는 기능이 관리형 노드에 설치되지 않았습니다. 예를 들어 인터넷 정보 서비스(IIS)와 같은 웹 서버 서비스에 대한 패치는 해당 패치가 기준에 승인되었지만 웹 서비스가 관리형 노드에 설치되지 않은 경우 NOT_APPLICABLE 로 표시됩니다. 패치가 후속 업데이트로 대체된 경우에도 패치를 NOT_APPLICABLE 로 표시할 수 있습니다. 즉, 이후 업데이트가 설치되고 NOT_APPLICABLE 업데이트가 더 이상 필요하지 않습니다. **Note** 이 규정 준수 상태는 Windows Server 운영 체제에서만 보고됩니다.	해당 사항 없음
FAILED	패치가 기준에 승인되었지만 인스턴스에 설치될 수 없었습니다. 이 상황을 해결하려면 문제를 이해하는 데 도움이 될 수 있는 정보에 대한 명령 출력을 검토합니다.	규정 미준수

[1] 상태가 INSTALLED_OTHER 및 NOT_APPLICABLE인 패치의 경우 Patch Manager에서는 describe-instance-patches 명령을 기반으로 하는 쿼리 결과에서 일부 데이터를 생략합니다(예:

Classification 및 Severity의 값). 이 작업은 AWS Systems Manager의 기능인 인벤토리의 개별 노드에 대한 데이터 제한을 초과하는 것을 방지하기 위해 수행됩니다. 모든 패치 세부 정보를 보려면 describe-available-patches 명령을 사용합니다.

규정 미준수 관리형 노드 패치 적용

관리형 노드의 패치 규정 준수 여부를 확인하는 데 사용할 수 있는 것과 동일한 AWS Systems Manager 도구와 프로세스를 사용하여 노드가 현재 적용되는 패치 규정을 준수하도록 할 수 있습니다. 관리형 노드가 패치 규정을 준수하도록 하려면 AWS Systems Manager의 기능인 Patch Manager가 Scan and install 작업을 실행해야 합니다. (규정 미준수 관리형 노드를 식별만 하고 해결은 하지 않는 것이 목표라면 그 대신에 Scan 작업을 실행합니다. 자세한 내용은 규정 미준수 관리형 노드 식별 섹션을 참조하세요.)

Systems Manager 사용하여 패치 설치

여러 도구 중에서 선택하여 Scan and install 작업을 실행할 수 있습니다.

- (권장) Systems Manager의 기능인 Quick Setup에서 패치 정책을 구성하여 전체 조직, 일부 조직 단위 또는 단일 AWS 계정의 일정에 따라 누락된 패치를 설치할 수 있습니다. 자세한 내용은 Patch Manager 조직 패치 적용 구성 단원을 참조하십시오.
- Run Command 태스크 유형에서 Systems Manager 문서(SSM 문서) AWS-RunPatchBaseline을 사용하는 유지 관리 기간을 생성합니다. 자세한 설명은 패치에 대한 유지 관리 기간 생성(콘솔)을 참조하세요.
- Run Command 작업에서 AWS-RunPatchBaseline을 수동으로 실행합니다. 자세한 설명은 콘솔에서 명령 실행을 참조하세요.

- [지금 패치(Patch now)] 옵션을 사용하여 온디맨드로 패치를 설치합니다. 자세한 내용은 관리형 노드 온디맨드 패치 섹션을 참조하세요.

의도치 않은 패치 규정 준수 데이터 덮어쓰기 방지

인스턴스에서 패치 규정 준수를 검사하기 위해 여러 유형의 작업을 수행하는 경우, 스캔할 때마다 이전 스캔의 패치 규정 준수 데이터를 덮어씁니다. 따라서 패치 규정 준수 데이터에 예상치 못한 결과가 발생할 수 있습니다.

예를 들어 현지 시간으로 매일 오전 2시에 패치 규정 준수를 스캔하는 패치 정책을 생성한다고 가정해 보겠습니다. 이 패치 정책은 심각도가 Critical, Important 및 Moderate로 표시된 패치를 대상으로 하는 패치 기준선을 사용합니다. 이 패치 기준선은 몇 가지 명시적으로 거부된 패치도 지정합니다.

또한 현지 시간으로 매일 오전 4시에 동일한 관리형 노드 세트를 스캔하도록 유지 관리 기간이 이미 설정되어 있고, 사용자가 이 설정은 삭제하거나 비활성화하지 않는다고 가정해 보겠습니다. 해당 유지 관리 기간의 태스크에는 심각도 `Critical`인 패치만 대상으로 하고 특정 패치를 제외하지 않는 다른 패치 기준선을 사용합니다.

유지 관리 기간에 의해 이 두 번째 스캔이 수행되면 첫 번째 스캔의 패치 규정 준수 데이터가 삭제되고 두 번째 스캔의 패치 규정 준수로 교체됩니다.

따라서 패치 작업에서 스캔 및 설치에는 자동화된 한 가지 방법만 사용하는 것이 좋습니다. 패치 정책을 설정하는 경우 패치 규정 준수를 검사하는 다른 방법을 삭제하거나 비활성화해야 합니다. 자세한 정보는 다음 주제를 참조하세요.

- 유지 관리 기간에서 패치 작업을 제거하려면 - 유지 관리 기간 태스크 업데이트 또는 등록 취소(콘솔)
- State Manager 연결을 삭제하려면 – 연결 삭제.

호스트 관리 구성에서 일일 패치 규정 준수 검사를 비활성화하려면 Quick Setup에서 다음을 수행합니다.

1. 탐색 창에서 Quick Setup를 선택합니다.
2. 업데이트할 호스트 관리 구성을 선택합니다.
3. Actions(작업), Edit configuration(구성 편집)을 차례로 선택합니다.
4. Scan instances for missing patches daily(매일 누락된 패치에 대한 인스턴스 스캔) 확인란의 선택을 취소합니다.
5. 업데이트를 선택합니다.

> **Note**
>
> Patch now(지금 패치) 옵션을 사용하여 관리형 노드의 규정 준수를 검사하면 패치 규정 준수 데이터도 덮어쓰여집니다.

관리형 노드 온디맨드 패치

AWS Systems Manager의 기능인 Patch Manager의 [지금 패치(Patch now)] 옵션을 사용하여 Systems Manager 콘솔에서 온디맨드 패치 작업을 실행할 수 있습니다. 즉, 관리형 노드의 규정 준수 상태를 업데이트하거나 비준수 노드에 패치를 설치하기 위해 일정을 생성할 필요가 없습니다. 또한

예약된 패치 기간을 설정하거나 수정하기 위해 AWS Systems Manager의 기능인 Patch Manager와 Maintenance Windows 사이에서 Systems Manager 콘솔을 전환할 필요가 없습니다.

지금 패치(Patch now)는 가능한 한 빨리 관리형 노드에 제로데이 업데이트를 적용하거나 다른 중요한 패치를 설치해야 할 때 특히 유용합니다.

> **Note**
>
> 온디맨드 패치는 한 번에 하나의 AWS 리전-AWS 계정 쌍에 대해서만 지원됩니다. 패치 정책을 기반으로 한 패치 작업에는 사용할 수 없습니다. 모든 관리형 노드를 규정 준수 상태로 유지하려면 패치 정책을 사용하는 것이 좋습니다. 패치 정책 작업에 대한 자세한 내용은 Quick Setup 패치 정책 사용 섹션을 참조하세요.

주제

'지금 패치(Patch now)' 작동 방식

지금 패치(Patch now)를 실행하려면 2가지 필수 설정만 지정하면 됩니다.

- 누락된 패치만 스캔할지 아니면 관리형 노드에서 패치를 스캔 및 설치할지 여부
- 작업을 실행할 관리형 노드

지금 패치(Patch now) 작업이 실행되면 다른 패치 작업에 대해 선택한 것과 동일한 방식으로 사용할 패치 기준선을 결정합니다. 관리형 노드가 패치 그룹과 연결된 경우 해당 그룹에 대해 지정된 패치 기준선이 사용됩니다. 관리형 노드가 패치 그룹에 연결되어 있지 않은 경우 작업은 관리형 노드의 운영 체제 유형에 대한 기본값으로 현재 설정된 패치 기준을 사용합니다. 미리 정의된 기준 또는 기본값으로 설정한 사용자 정의 기준일 수 있습니다. 패치 기준 선택에 대한 자세한 내용은 패치 그룹 정보 섹션을 참조하세요.

지금 패치(Patch now)에 대해 지정할 수 있는 옵션으로는 패치 후 관리형 노드 재부팅 시기 또는 재부팅 여부 선택, 패치 작업을 위해 로그 데이터를 저장할 Amazon Simple Storage Service(Amazon S3) 버킷 지정, 패치 중 수명 주기 후크로 Systems Manager 문서(SSM 문서) 실행 등이 있습니다.

'지금 패치'에 대한 동시성 및 오류 임계값

패치 지금(Patch now) 작업의 경우 동시성 및 오류 임계값 옵션은 Patch Manager에서 처리합니다. 한 번에 패치할 관리형 노드 수 또는 작업이 실패하기 전에 허용되는 오류 수를 지정할 필요가 없습니다. Patch Manager는 온디맨드로 패치할 때 다음 표에 설명된 동시성 및 오류 임계값 설정을 적용합니다.

> **Important**
>
> 다음 임계값은 `Scan and install` 작업에만 적용됩니다. `Scan` 작업의 경우 Patch Manager는 최대 1,000개의 노드를 동시에 스캔하고 최대 1,000개의 오류가 발생할 때까지 계속 스캔합니다.

동시성: 설치 작업

지금 패치(Patch now) 작업의 총 관리형 노드 수	한 번에 스캔 또는 패치되는 관리형 노드 수
25개 미만	1
25~100개	5%
101~1,000개	8%
1,000개 이상	10%

오류 임계값: 설치 작업

지금 패치(Patch now) 작업의 총 관리형 노드 수	작업이 실패하기 전에 허용되는 오류 수
25개 미만	1
25~100개	5
101~1,000개	10
1,000개 이상	10

'지금 패치' 수명 주기 후크 사용

지금 패치(Patch now)에서는 패치 작업 `Install` 중에 SSM 명령 문서를 수명 주기 후크로 실행할 수 있는 기능을 제공합니다. 패치를 적용한 후 또는 재부팅 후 애플리케이션에 패치를 적용하거나 상태 확인을 실행하기 전에 애플리케이션을 종료하는 등의 작업에 이러한 후크를 사용할 수 있습니다.

수명 주기 후크 사용에 대한 자세한 내용은 `AWS-RunPatchBaselineWithHooks` SSM 문서 정보 섹션을 참조하세요.

다음 표에는 각 후크에 대한 샘플 사용 외에도 세 가지 지금 패치(Patch now) 재부팅 옵션 각각에 사용할 수 있는 수명 주기 후크가 나열됩니다.

수명 주기 후크 및 샘플 사용

재부팅 옵션	후크: 설치 전	후크: 설치 후	후크: 종료 시	후크: 예약된 재부팅 후
필요한 경우 재부팅	패치가 시작되기 전에 SSM 문서를 실행합니다. 사용 예: 패치 프로세스가 시작되기 전에 애플리케이션을 안전하게 종료합니다.	패치 작업 종료 시 및 관리형 노드 재부팅 전에 SSM 문서를 실행합니다. 사용 예: 잠재적 재부팅 전에 서드파티 애플리케이션 설치와 같은 작업을 실행합니다.	패치 적용 작업이 완료되고 인스턴스가 재부팅되면 SSM 문서를 실행합니다. 사용 예: 패치 후 애플리케이션이 예상대로 실행되고 있는지 확인합니다.	사용할 수 없음
인스턴스 재부팅 안 함	위와 동일합니다.	패치 작업 종료 시 SSM 문서를 실행합니다. 사용 예시: 패치 후 애플리케이션이 예상대로 실행되고 있는지 확인합니다.	사용할 수 없음	사용할 수 없음

재부팅 옵션	후크: 설치 전	후크: 설치 후	후크: 종료 시	후크: 예약된 재부팅 후
재부팅 시간 예약	위와 동일합니다.	인스턴스 재부팅 안 함과 동일합니다.	사용할 수 없음	예약된 재부팅이 완료된 후 바로 SSM 문서를 실행합니다. 사용 예: 재부팅 후 애플리케이션이 예상대로 실행되고 있는지 확인합니다.

'지금 패치(Patch now)' 실행

다음 절차에 따라 온디맨드로 관리형 노드를 패치합니다.

'지금 패치(Patch now)'를 실행하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Patch Manager를 선택합니다.
3. [AWS Systems Manager Patch Manager] 페이지 또는 [패치 기준(Patch baselines)] 페이지에서 [지금 패치(Patch now)]를 선택합니다.
4. [패치 작업(Patching operation)]에서 다음 중 하나를 선택합니다.
   - 스캔(Scan): Patch Manager가 관리형 노드에서 누락된 패치를 찾지만 설치하지는 않습니다. [Compliance] 대시보드 또는 패치 규정 준수를 보는 데 사용하는 다른 도구에서 결과를 볼 수 있습니다.
   - 스캔 및 설치(Scan and install): Patch Manager가 관리형 노드에서 누락된 패치를 찾아 설치합니다.
5. 이전 단계에서 [검사 후 설치(Scan and install)]를 선택한 경우에만 이 단계를 사용합니다. [재부팅 옵션(Reboot option)]에서 다음 중 하나를 선택합니다.
   - 필요한 경우 재부팅(Reboot if needed): 설치 후 패치 설치를 완료하는 데 필요한 경우에만 Patch Manager가 관리형 노드를 재부팅합니다.

- 인스턴스 재부팅 안 함(Don't reboot my instances): 설치 후 Patch Manager가 관리형 노드를 재부팅하지 않습니다. Patch Manager 밖에서 재부팅을 선택하거나 관리할 때 노드를 수동으로 재부팅할 수 있습니다.
- 재부팅 시간 예약(Schedule a reboot time): Patch Manager가 관리형 노드를 재부팅할 날짜, 시간 및 UTC 시간대를 지정합니다. 지금 패치(Patch now) 작업을 실행한 후, 예약된 재부팅이 `AWS-PatchRebootAssociation`이라는 이름과 함께 State Manager에 연결로 나열됩니다.

6. [패치를 적용할 인스턴스(Instances to patch)] 섹션에서 다음 중 하나를 선택합니다.

   - 모든 인스턴스 패치(Patch all instances): Patch Manager가 현재 AWS 리전의 AWS 계정에서 모든 관리형 노드에 대해 지정된 작업을 실행합니다.
   - 지정한 대상 인스턴스만 패치(Patch only the target instances I specify): 다음 단계에서 대상으로 지정할 관리형 노드를 지정합니다.

7. 이전 단계에서 [지정한 대상 인스턴스만 패치(Patch only the target instances I specify)]를 선택한 경우에만 이 단계를 사용합니다. 대상 선택(Target selection) 섹션에서 태그를 지정하거나, 수동으로 노드를 선택하거나, 리소스 그룹을 지정하여 이 작업을 실행할 노드를 식별합니다.

   > **Note**
   >
   > 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.
   > 리소스 그룹을 대상으로 선택하는 경우 AWS CloudFormation 스택을 기반으로 하는 리소스 그룹은 여전히 기본 `aws:cloudformation:`*`stack-id`* 태그로 태그를 지정해야 합니다. 제거된 경우 Patch Manager가 리소스 그룹에 속하는 관리형 노드를 확인하지 못할 수 있습니다.

8. (옵션) 이 패치 작업에서 로그를 생성하고 저장하려면 [패치 로그 스토리지(Patching log storage)]에서 로그를 저장할 S3 버킷을 선택합니다.

   > **Note**
   >
   > 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아니라 인스턴스에 할당된 인스턴스 프로파일(EC2 인스턴스용) 또는 IAM 서비스 역할(하이브리드 정품 인증 시스템)의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성 또는 하이브리드 및 멀티클라우드 환경에서 Systems Manager에 필요한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에

있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

9. (옵션) 패치 작업의 특정 지점 동안 SSM 문서를 수명 주기 후크로 실행하려면 다음을 수행합니다.

   - [수명 주기 후크 사용(Use lifecycle hooks)]을 선택합니다.
   - 사용 가능한 각 후크에 대해 작업의 지정된 지점에서 실행할 SSM 문서를 선택합니다.
     - 설치 전
     - 설치 후
     - 종료 시
     - 예약된 재부팅 후

   > **Note**
   >
   > 기본 문서인 AWS-Noop은 작업을 실행하지 않습니다.

10. [지금 패치(Patch now)]를 선택합니다.

    [연결 실행 요약(Association execution summary)] 페이지가 열립니다. (이제 패치 작업에 AWS Systems Manager의 기능인 State Manager의 연결이 사용됩니다.) 작업 요약(Operation summary) 영역에서 지정한 관리형 노드의 스캔 또는 패치 상태를 모니터링할 수 있습니다.

패치 기준 작업

AWS Systems Manager의 기능인 Patch Manager의 패치 기준선은 관리형 노드에 대한 설치가 승인되는 패치를 정의합니다. 패치 승인 또는 거부는 하나씩 지정할 수 있습니다. 또한 자동 승인 규칙을 생성하여 특정 유형의 업데이트(예: 필수 업데이트)가 자동 승인되도록 지정할 수도 있습니다. 거부된 목록은 규칙 및 승인 목록을 모두 재정의합니다. 승인된 패치 목록을 사용하여 특정 패키지를 설치하려면 먼저 모든 자동 승인 규칙을 제거하십시오. 임의 패치를 명시적으로 거부로 구분하면 해당 패치는 자동 승인 규칙의 모든 기준을 만족하더라도 승인 또는 설치되지 않습니다. 또한 패치가 해당 관리형 노드에 대해 승인되었더라도 노드의 소프트웨어에 적용되는 경우에만 패치가 관리형 노드에 설치됩니다.

주제

- AWS가 미리 정의한 패치 기준 보기
- 사용자 정의 패치 기준 작업
- 기존 패치 기준을 기본값으로 설정

추가 정보

- 패치 기준 정보

AWS가 미리 정의한 패치 기준 보기

AWS Systems Manager의 기능인 Patch Manager에는 Patch Manager가 지원하는 각 운영 체제에 대해 미리 정의된 패치 기준이 있습니다. 이 패치 기준선을 사용하거나(기본 패치 기준선은 사용자 지정할 수 없음), 혹은 자체 기준선을 생성할 수도 있습니다. 다음 절차에서는 미리 정의된 패치 기준이 해당 요건을 충족하는지 확인하는 방법을 설명합니다. 패치 기준선에 대한 자세한 내용은 미리 정의된 패치 및 사용자 지정 패치 기준 정보 섹션을 참조하세요.

AWS가 미리 정의한 패치 기준을 보려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Patch Manager를 선택합니다.
3. 패치 기준 목록에서 미리 정의된 패치 기준 중 하나의 기준 ID를 선택하십시오.

   -또는-

   현재 AWS 리전에서 Patch Manager에 처음으로 액세스하는 경우 개요로 시작을 선택하고, 패치 기준선을 선택한 다음에 사전 정의된 패치 기준선 중 하나의 기준선 ID를 선택합니다.

   > **Note**
   >
   > Windows Server에는 3개의 미리 정의된 패치 기준이 제공됩니다. 패치 기준 `AWS-DefaultPatchBaseline` 및 `AWS-WindowsPredefinedPatchBaseline-OS`는 Windows 운영 체제 자체에서 운영 체제 업데이트만 지원합니다. `AWS-DefaultPatchBaseline`은 다른 패치 기준을 지정하지 않는 한 Windows Server 관리형 노드의 기본 패치 기준으로 사용됩니다. 이러한 두 패치 기준의 구성 설정은 동일합니다. 둘 중 더 새로운 `AWS-WindowsPredefinedPatchBaseline-OS`는 Windows Server에 대해 미리 정의된 세 번째 패치 기준과 구별하기 위해 생성되었습니다. 해당 패치 기준인 `AWS-WindowsPredefinedPatchBaseline-OS-Applications`는 Windows Server 운영 체제 및 Microsoft에서 릴리스한 지원되는 애플리케이션을 패치하는 데 사용될 수 있습니다.
   > 자세한 내용은 기존 패치 기준을 기본값으로 설정 단원을 참조하십시오.

4. 승인 규칙 섹션에서 패치 기준선 구성을 검토합니다.
5. 구성이 관리형 노드에 적합한 경우 패치 그룹 작업 절차로 건너뛸 수 있습니다.

-또는-

자체적으로 기본 패치 기준선을 생성하려면 사용자 정의 패치 기준 작업 주제를 계속 진행합니다.

사용자 정의 패치 기준 작업

AWS Systems Manager의 기능인 Patch Manager에는 Patch Manager가 지원하는 각 운영 체제에 대해 미리 정의된 패치 기준이 있습니다. 이 패치 기준선을 사용하거나(기본 패치 기준선은 사용자 지정할 수 없음), 혹은 자체 기준선을 생성할 수도 있습니다.

다음 절차에서는 사용자 정의 패치 기준을 직접 생성, 업데이트 및 삭제하는 방법을 설명합니다. 패치 기준선에 대한 자세한 내용은 미리 정의된 패치 및 사용자 지정 패치 기준 정보 섹션을 참조하세요.

주제

- 사용자 정의 패치 기준 생성(Linux)
- 사용자 정의 패치 기준 생성(macOS)
- 사용자 정의 패치 기준 생성(Windows)
- 사용자 지정 패치 기준선 업데이트 또는 삭제

사용자 정의 패치 기준 생성(Linux)

AWS Systems Manager의 기능인 Patch Manager에서 Linux 관리형 노드에 대한 사용자 정의 패치 기준을 생성하려면 다음 절차를 따릅니다.

macOS 관리형 노드의 패치 기준 생성에 대한 자세한 내용은 사용자 정의 패치 기준 생성(macOS) 섹션을 참조하세요. Windows 관리형 노드의 패치 기준 생성에 대한 자세한 내용은 사용자 정의 패치 기준 생성(Windows) 섹션을 참조하세요.

Linux 관리형 노드의 사용자 정의 패치 기준 생성

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Patch Manager를 선택합니다.
3. 패치 기준선 탭을 선택한 다음 패치 기준선 생성을 선택합니다.

   -또는-

   현재 AWS 리전에서 처음으로 Patch Manager에 액세스한 경우 개요를 통한 시작을 선택하고 패치 기준선 탭을 선택한 다음, 패치 기준선 생성을 선택합니다.

4. Name(이름) 필드에 새로운 패치 기준 이름(예: `MyRHELPatchBaseline`)을 입력합니다.
5. (선택 사항) 설명에 이 패치 기준에 대한 설명을 입력합니다.
6. Operating system에서 운영 체제(예: `Red Hat Enterprise Linux`)를 선택합니다.
7. 이 패치 기준을 생성하는 즉시 선택한 운영 체제의 기본값으로 사용하려면 Set this patch baseline as the default patch baseline for ***operating system name*** instances(이 패치 기준을 operating system name 인스턴스의 기본 패치 기준으로 설정) 옆에 있는 확인란을 선택합니다.

> **Note**
>
> 이 옵션은 2022년 12월 22일에 패치 정책이 릴리스되기 전에 Patch Manager에 처음 액세스한 경우에만 사용할 수 있습니다.
> 기존 패치 기준을 기본값으로 설정하는 방법에 대한 자세한 내용은 기존 패치 기준을 기본값으로 설정 섹션을 참조하세요.

8. Approval rules for operating systems(운영 체제에 대한 승인 규칙) 섹션에서 필드를 사용하여 자동 승인 규칙을 한 개 이상 생성합니다.

   - 제품: 승인 규칙이 적용되는 운영 체제 버전입니다(예: `RedhatEnterpriseLinux7.4`). 기본 선택은 `All`입니다.
   - Classification(분류): 승인 규칙이 적용되는 패치 유형입니다(예: `Security` 또는 `Enhancement`). 기본 선택은 `All`입니다.

   > **Tip**
   >
   > 패치 기준을 구성하여 RHEL 7.8과 같이 Linux용 부 버전 업그레이드의 설치 여부를 제어할 수 있습니다. 해당 리포지토리에서 업데이트할 수 있는 경우 Patch Manager를 통해 부 버전 업그레이드를 자동 설치할 수 있습니다.
   > Linux 운영 체제의 경우 부 버전 업그레이드는 일관되게 분류되지 않습니다. 동일한 커널 버전 내에서도 버그 수정 또는 보안 업데이트로 분류되거나 분류되지 않을 수 있습니다. 다음은 패치 기준으로 패치 설치 여부를 제어하는 몇 가지 옵션입니다.
   >
   > - 옵션 1: 마이너 버전 업그레이드가 가능한 경우 설치되도록 하기 위한 가장 광범위한 승인 규칙은 Classification(분류)을 `All`(*)로 지정하고 Include nonsecurity updates(비보안 업데이트 포함) 옵션을 선택하는 것입니다.
   > - 옵션 2: 운영 체제 버전에 대한 패치가 설치되도록 하려면 와일드카드(*) 를 사용하여 기준의 Patch exceptions(패치 예외) 섹션에서 커널 형식을 지정할 수 있습니다. 예를 들어 RHEL 7.*의 커널 형식은 `kernel-3.10.0-*.el7.x86_64`입니다.

부 버전 업그레이드를 포함한 모든 패치가 RHEL 7.* 관리형 노드에 적용되도록 하려면 패치 기준의 Approved patches(승인된 패치) 목록에 `kernel-3.10.0-*.el7.x86_64`를 입력합니다. (부 버전 패치의 정확한 패키지 이름을 알고 있다면 대신 입력할 수 있음)

- 옵션 3: `AWS-RunPatchBaseline` 문서에서 InstallOverrideList 파라미터를 사용하여 부 버전 업그레이드 등, 관리형 노드에 적용할 패치를 가장 많이 제어할 수 있습니다. 자세한 내용은 AWS-RunPatchBaseline SSM 문서 정보 단원을 참조하십시오.

- Severity: 규칙이 적용될 패치의 심각도 값입니다(예: `Critical`). 기본 선택은 `All`입니다.
- Auto-approval: 자동 승인을 위해 패치를 선택하는 방법입니다.

> **Note**
>
> Ubuntu Server에 대한 업데이트 패키지의 릴리스 날짜를 확실히 결정할 수 없으므로 이 운영 체제에서는 자동 승인 옵션이 지원되지 않습니다.

  - Approve patches after a specified number of days(지정된 일 수 후 패치 승인): 패치가 릴리스 또는 마지막으로 업데이트된 후 자동으로 승인되기까지 Patch Manager가 대기하는 일 수입니다. 0~360의 정수를 입력할 수 있습니다. 대부분의 시나리오에서 100일 이상 기다리지 않는 것이 좋습니다.
  - Approve patches released up to a specific date(특정 날짜까지 릴리스된 패치 승인): Patch Manager가 해당 날짜 또는 그 이전에 릴리스 또는 업데이트된 모든 패치를 자동으로 적용하는 패치 릴리스 날짜입니다. 예를 들어 2023년 7월 7일을 지정하면 2023년 7월 8일 또는 그 이후에 릴리스되거나 마지막으로 업데이트된 패치가 자동으로 설치되지 않습니다.
- (선택 사항) 규정 준수 보고: 기준선에 따라 승인된 패치에 할당하려는 심각도 수준입니다(예: `Critical` 또는 `High`).

> **Note**
>
> 규정 준수 보고 수준을 지정하고 승인된 패치의 패치 상태가 `Missing`으로 보고되는 경우 패치 기준선의 전반적인 보고된 규정 준수 심각도는 사용자가 지정한 심각도 수준입니다.

- Include non-security updates(비보안 업데이트 포함): 보안 관련 패치 외에, 소스 리포지토리에서 사용 가능한 비보안 Linux 운영 체제 패치도 설치하려면 확인란을 선택합니다.

> **Note**
>
> SUSE Linux Enterprise Server(SLES)의 경우 보안 및 비보안 문제에 대한 패치가 SLES 관리형 노드에 기본적으로 설치되어 있기 때문에 확인란을 선택하지 않아도 됩니다. 자세한 내용은 보안 패치 선택 방법에서 SLES에 대한 내용을 참조하십시오.

사용자 지정 패치 기준의 승인 규칙 작업에 대한 자세한 내용은 사용자 지정 기준 정보 섹션을 참조하세요.

9. 승인 규칙을 준수하는 패치 이외에 모든 패치를 명시적으로 승인하려면 패치 예외 섹션에서 다음을 수행합니다.

   - Approved patches(승인된 패치)에 승인할 패치의 쉼표로 구분된 목록을 입력합니다.

     > **Note**
     >
     > 승인된 패치 및 거부된 패치 목록의 승인된 형식에 대한 자세한 내용은 승인 패치 및 거부 패치 목록의 패키지 이름 형식 정보 섹션을 참조하세요.

   - (선택 사항) Approved patches compliance level(승인된 패치 규정 준수 수준)에서 규정 준수 수준을 목록의 패치에 할당합니다.
   - 지정하는 승인된 패치가 보안과 관련되지 않은 경우 Linux 운영 체제에 이러한 패치를 설치하려면 비보안 업데이트 포함 확인란을 선택합니다.

10. 승인 규칙을 준수하는 패치를 명시적으로 거부하려면 패치 예외 섹션에서 다음을 수행합니다.

    - Rejected patches(거부된 패치)에 거부할 패치의 쉼표로 구분된 목록을 입력합니다.

      > **Note**
      >
      > 승인된 패치 및 거부된 패치 목록의 승인된 형식에 대한 자세한 내용은 승인 패치 및 거부 패치 목록의 패키지 이름 형식 정보 섹션을 참조하세요.

    - [거부된 패치 작업(Rejected patches action)]에서 Patch Manager가 [거부된 패치(Rejected patches)] 목록에 포함된 패치에 대해 수행할 작업을 선택합니다.
      - [종속성으로 허용(Allow as dependency)]: [거부된 패치(Rejected patches)] 목록에 있는 패키지는 다른 패키지에 종속성을 가질 때만 설치됩니다. 이 경우 패치 기준을 준수하는 것으로 간

주되고 상태가 InstalledOther로 보고됩니다. 이는 옵션을 지정하지 않은 경우의 기본 작업입니다.

- 차단: 거부된 패치 목록에 있는 패키지와 종속적으로 그것들을 포함하는 패키지는 어떠한 경우에도 Patch Manager에서 설치되지 않습니다. 패키지가 거부된 패치 목록에 추가되기 전에 설치되거나 이후에 Patch Manager 외부에 설치된 경우 패치 기준을 준수하지 않는 것으로 간주되고 상태가 InstalledRejected로 보고됩니다.

11. (선택 사항) AmazonLinux2016.03 및 AmazonLinux2017.09 등의 다른 운영 체제 버전에 대해 대체 패치 리포지토리를 지정하려면 패치 소스 섹션에서 각 제품에 대해 다음을 수행합니다.

    - Name에 소스 구성을 식별하는 데 도움이 되는 이름을 입력합니다.
    - Product에서 패치 소스 리포지토리의 운영 체제 버전을 선택합니다(예: `RedhatEnterpriseLinux7.4`).
    - [구성(Configuration)]에 사용할 yum 리포지토리 구성의 값을 다음 형식으로 입력합니다.

    ```
    [main]
    name=MyCustomRepository
    baseurl=https://my-custom-repository
    enabled=1
    ```

    > **Tip**
    >
    > yum 리포지토리 구성에 사용할 수 있는 기타 옵션에 대한 자세한 내용은 dnf.conf(5)를 참조하세요.

    각 추가 운영 체제 버전(최대 20개)에 대해 소스 리포지토리를 지정하려면 Add another source(다른 소스 추가)를 선택합니다.

    대체 소스 패치 리포지토리에 대한 자세한 내용은 대체 패치 소스 리포지토리를 지정하는 방법(Linux) 섹션을 참조하세요.

12. (선택 사항) Manage tags(태그 관리)의 경우, 하나 이상의 태그 키 이름/값 페어를 패치 기준에 적용합니다.

    태그는 리소스에 할당하는 선택적 메타데이터입니다. 태그를 사용하면 용도, 소유자 또는 환경을 기준으로 하는 등 리소스를 다양한 방식으로 분류할 수 있습니다. 예를 들어, 패치 기준에 태그를 지정하여 패치의 심각도 수준, 해당 패치가 적용되는 운영 체제 제품군 및 환경 유형을 식별할 수 있습니다. 이 경우 다음 키 이름/값 페어와 비슷한 태그를 지정할 수 있습니다.

- `Key=PatchSeverity,Value=Critical`
- `Key=OS,Value=RHEL`
- `Key=Environment,Value=Production`

13. 패치 기준 생성을 선택합니다.

사용자 정의 패치 기준 생성(macOS)

AWS Systems Manager의 기능인 Patch Manager에서 macOS 관리형 노드에 대한 사용자 정의 패치 기준을 생성하려면 다음 절차를 따릅니다.

Windows Server 관리형 노드의 패치 기준 생성에 대한 자세한 내용은 사용자 정의 패치 기준 생성 (Windows) 섹션을 참조하세요. Linux 관리형 노드의 패치 기준 생성에 대한 자세한 내용은 사용자 정의 패치 기준 생성(Linux) 섹션을 참조하세요.

> **Note**
>
> 일부 AWS 리전에서는 macOS가 지원되지 않습니다. macOS용 Amazon EC2 지원에 대한 자세한 내용은 Amazon EC2 사용 설명서의 Amazon EC2 Mac 인스턴스를 참조하세요.

macOS 관리형 노드의 사용자 정의 패치 기준 생성

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Patch Manager를 선택합니다.
3. 패치 기준선 탭을 선택한 다음 패치 기준선 생성을 선택합니다.

   -또는-

   현재 AWS 리전에서 처음으로 Patch Manager에 액세스한 경우 개요를 통한 시작을 선택하고 패치 기준선 탭을 선택한 다음, 패치 기준선 생성을 선택합니다.
4. Name(이름) 필드에 새로운 패치 기준 이름(예: `MymacOSPatchBaseline`)을 입력합니다.
5. (선택 사항) 설명에 이 패치 기준에 대한 설명을 입력합니다.
6. 운영 체제에서 macOS를 선택합니다.
7. 이 패치 기준을 생성하는 즉시 macOS의 기본값으로 사용하려면 [이 패치 기준을 macOS 인스턴스의 기본 패치 기준으로 설정(Set this patch baseline as the default patch baseline for macOS instances)] 옆에 있는 확인란을 선택합니다.

> **Note**
>
> 이 옵션은 2022년 12월 22일에 패치 정책이 릴리스되기 전에 Patch Manager에 처음 액세스한 경우에만 사용할 수 있습니다.
> 기존 패치 기준을 기본값으로 설정하는 방법에 대한 자세한 내용은 기존 패치 기준을 기본값으로 설정 섹션을 참조하세요.

8. Approval rules for operating systems(운영 체제에 대한 승인 규칙) 섹션에서 필드를 사용하여 자동 승인 규칙을 한 개 이상 생성합니다.

   - 제품: 승인 규칙이 적용되는 운영 체제 버전입니다(예: `Mojave10.14.1` 또는 `Catalina10.15.1`). 기본 선택은 `All`입니다.

     > **Note**
     >
     > Homebrew 오픈 소스 소프트웨어 패키지 관리 시스템에서는 macOS 10.14.x(Mojave) 및 10.15.x(Catalina) 지원이 중단되었습니다. 따라서 이러한 버전의 패치 적용 작업은 현재 지원되지 않습니다.

   - 분류: 패치 작업 중 패키지를 적용하려는 하나 이상의 패키지 관리자입니다. 사용자는 다음 중에서 선택할 수 있습니다.

     - softwareupdate
     - 설치 관리자
     - brew
     - brew cask

     기본 선택은 `All`입니다.

   - (선택 사항) 규정 준수 보고: 기준선에 따라 승인된 패치에 할당하려는 심각도 수준입니다(예: `Critical` 또는 `High`).

     > **Note**
     >
     > 규정 준수 보고 수준을 지정하고 승인된 패치의 패치 상태가 `Missing`으로 보고되는 경우 패치 기준선의 전반적인 보고된 규정 준수 심각도는 사용자가 지정한 심각도 수준입니다.

- [비보안 업데이트 포함(Include non-security updates)]: 보안 관련 패치 외에, 소스 리포지토리에서 사용 가능한 비보안 운영 체제 패치도 설치하려면 확인란을 선택합니다.

사용자 지정 패치 기준의 승인 규칙 작업에 대한 자세한 내용은 사용자 지정 기준 정보 섹션을 참조하세요.

9. 승인 규칙을 준수하는 패치 이외에 모든 패치를 명시적으로 승인하려면 패치 예외 섹션에서 다음을 수행합니다.

   - Approved patches(승인된 패치)에 승인할 패치의 쉼표로 구분된 목록을 입력합니다.

     > **Note**
     >
     > 승인된 패치 및 거부된 패치 목록의 승인된 형식에 대한 자세한 내용은 승인 패치 및 거부 패치 목록의 패키지 이름 형식 정보 섹션을 참조하세요.

   - (선택 사항) Approved patches compliance level(승인된 패치 규정 준수 수준)에서 규정 준수 수준을 목록의 패치에 할당합니다.
   - 지정하는 승인된 패치가 보안과 관련되지 않은 경우 macOS 운영 체제에 이러한 패치를 설치하려면 비보안 업데이트 포함 확인란을 선택합니다.

10. 승인 규칙을 준수하는 패치를 명시적으로 거부하려면 패치 예외 섹션에서 다음을 수행합니다.

    - Rejected patches(거부된 패치)에 거부할 패치의 쉼표로 구분된 목록을 입력합니다.

      > **Note**
      >
      > 승인된 패치 및 거부된 패치 목록의 승인된 형식에 대한 자세한 내용은 승인 패치 및 거부 패치 목록의 패키지 이름 형식 정보 섹션을 참조하세요.

    - [거부된 패치 작업(Rejected patches action)]에서 Patch Manager가 [거부된 패치(Rejected patches)] 목록에 포함된 패치에 대해 수행할 작업을 선택합니다.
      - [종속성으로 허용(Allow as dependency)]: [거부된 패치(Rejected patches)] 목록에 있는 패키지는 다른 패키지에 종속성을 가질 때만 설치됩니다. 이 경우 패치 기준을 준수하는 것으로 간주되고 상태가 InstalledOther로 보고됩니다. 이는 옵션을 지정하지 않은 경우의 기본 작업입니다.
      - 차단: 거부된 패치 목록에 있는 패키지와 종속적으로 그것들을 포함하는 패키지는 어떠한 경우에도 Patch Manager에서 설치되지 않습니다. 패키지가 거부된 패치 목록에 추가되기 전에

설치되거나 이후에 Patch Manager 외부에 설치된 경우 패치 기준을 준수하지 않는 것으로 간주되고 상태가 InstalledRejected로 보고됩니다.

11. (선택 사항) Manage tags(태그 관리)의 경우, 하나 이상의 태그 키 이름/값 페어를 패치 기준에 적용합니다.

    태그는 리소스에 할당하는 선택적 메타데이터입니다. 태그를 사용하면 용도, 소유자 또는 환경을 기준으로 하는 등 리소스를 다양한 방식으로 분류할 수 있습니다. 예를 들어 패치 기준에 태그를 지정하여 지정한 패치의 심각도 수준, 적용되는 패키지 관리자 및 환경 유형을 식별할 수 있습니다. 이 경우 다음 키 이름/값 페어와 비슷한 태그를 지정할 수 있습니다.

    - `Key=PatchSeverity,Value=Critical`
    - `Key=PackageManager,Value=softwareupdate`
    - `Key=Environment,Value=Production`
12. 패치 기준 생성을 선택합니다.

사용자 정의 패치 기준 생성(Windows)

AWS Systems Manager의 기능인 Patch Manager에서 Windows 관리형 노드에 대한 사용자 정의 패치 기준을 생성하려면 다음 절차를 따릅니다.

Linux 관리형 노드의 패치 기준 생성에 대한 자세한 내용은 사용자 정의 패치 기준 생성(Linux) 섹션을 참조하세요. macOS 관리형 노드의 패치 기준 생성에 대한 자세한 내용은 사용자 정의 패치 기준 생성(macOS) 섹션을 참조하세요.

Windows 서비스 팩 설치에만 제한되는 패치 기준 생성의 예는 자습서: Windows 서비스 팩 설치를 위한 패치 기준선 생성(콘솔) 섹션을 참조하세요.

사용자 지정 패치 기준을 생성하려면(Windows)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Patch Manager를 선택합니다.
3. 패치 기준선 탭을 선택한 다음 패치 기준선 생성을 선택합니다.

   -또는-

   현재 AWS 리전에서 처음으로 Patch Manager에 액세스한 경우 개요를 통한 시작을 선택하고 패치 기준선 탭을 선택한 다음, 패치 기준선 생성을 선택합니다.
4. Name(이름) 필드에 새로운 패치 기준 이름(예: `MyWindowsPatchBaseline`)을 입력합니다.

5. (선택 사항) 설명에 이 패치 기준에 대한 설명을 입력합니다.
6. 운영 체제에서 `Windows`를 선택합니다.
7. 이 패치 기준을 생성하는 즉시 Windows의 기본값으로 사용하려면 Set this patch baseline as the default patch baseline for Windows Server instances(이 패치 기준을 Windows Server 인스턴스의 기본 패치 기준으로 설정)를 선택하십시오.

> **Note**
>
> 이 옵션은 2022년 12월 22일에 패치 정책이 릴리스되기 전에 Patch Manager에 처음 액세스한 경우에만 사용할 수 있습니다.
> 기존 패치 기준을 기본값으로 설정하는 방법에 대한 자세한 내용은 기존 패치 기준을 기본값으로 설정 섹션을 참조하세요.

8. Approval rules for operating systems(운영 체제에 대한 승인 규칙) 섹션에서 필드를 사용하여 자동 승인 규칙을 한 개 이상 생성합니다.

   - 제품: 승인 규칙이 적용되는 운영 체제 버전입니다(예: `WindowsServer2012`). 기본 선택은 `All`입니다.
   - Classification(분류): 승인 규칙이 적용되는 패치 유형입니다(예: `CriticalUpdates`, `Drivers` 및 `Tools`). 기본 선택은 `All`입니다.

     > **Tip**
     >
     > `ServicePacks`를 포함하거나 [분류(Classification)] 목록에서 `All`을 선택하여 Windows 서비스 팩 설치를 승인 규칙에 포함할 수 있습니다. 예시는 자습서: Windows 서비스 팩 설치를 위한 패치 기준선 생성(콘솔) 섹션을 참조하세요.

   - Severity: 규칙이 적용될 패치의 심각도 값입니다(예: `Critical`). 기본 선택은 `All`입니다.
   - Auto-approval: 자동 승인을 위해 패치를 선택하는 방법입니다.
     - Approve patches after a specified number of days(지정된 일 수 후 패치 승인): 패치가 릴리스 또는 업데이트된 후 자동으로 승인되기까지 Patch Manager가 대기하는 일 수입니다. 0~360의 정수를 입력할 수 있습니다. 대부분의 시나리오에서 100일 이상 기다리지 않는 것이 좋습니다.
     - Approve patches released up to a specific date(특정 날짜까지 릴리스된 패치 승인): Patch Manager가 해당 날짜 또는 그 이전에 릴리스 또는 업데이트된 모든 패치를 자동으로 적용하

는 패치 릴리스 날짜입니다. 예를 들어 2023년 7월 7일을 지정하면 2023년 7월 8일 또는 그 이후에 릴리스되거나 마지막으로 업데이트된 패치가 자동으로 설치되지 않습니다.

- (선택 사항) Compliance reporting(규정 준수 보고): 기준에서 승인된 패치에 할당할 심각도 수준입니다(예: `High`).

> **Note**
>
> 규정 준수 보고 수준을 지정하고 승인된 패치의 패치 상태가 `Missing`으로 보고되는 경우 패치 기준선의 전반적인 보고된 규정 준수 심각도는 사용자가 지정한 심각도 수준입니다.

9. (옵션) [애플리케이션에 대한 승인 규칙(Approval rules for applications)] 섹션에서 필드를 사용하여 자동 승인 규칙을 1개 이상 생성합니다.

> **Note**
>
> 승인 규칙을 지정하는 대신 승인된 패치 및 거부된 패치 목록을 패치 예외로 지정할 수 있습니다. 10단계와 11단계를 참조하세요.

- Product family: 규칙을 지정하려는 일반 Microsoft 제품군입니다(예: `Office` 또는 `Exchange Server`).
- 제품: 승인 규칙이 적용되는 애플리케이션 버전입니다(예: `Office 2016` 또는 `Active Directory Rights Management Services Client 2.0 2016`). 기본 선택은 `All`입니다.
- Classification: 승인 규칙이 적용되는 패치 유형입니다(예: `CriticalUpdates`). 기본 선택은 `All`입니다.
- Severity: 규칙이 적용되는 패치의 심각도 값입니다(예: `Critical`). 기본 선택은 `All`입니다.
- Auto-approval: 자동 승인을 위해 패치를 선택하는 방법입니다.
  - Approve patches after a specified number of days(지정된 일 수 후 패치 승인): 패치가 릴리스 또는 업데이트된 후 자동으로 승인되기까지 Patch Manager가 대기하는 일 수입니다. 0~360의 정수를 입력할 수 있습니다. 대부분의 시나리오에서 100일 이상 기다리지 않는 것이 좋습니다.
  - Approve patches released up to a specific date(특정 날짜까지 릴리스된 패치 승인): Patch Manager가 해당 날짜 또는 그 이전에 릴리스 또는 업데이트된 모든 패치를 자동으로 적용하

는 패치 릴리스 날짜입니다. 예를 들어 2023년 7월 7일을 지정하면 2023년 7월 8일 또는 그 이후에 릴리스되거나 마지막으로 업데이트된 패치가 자동으로 설치되지 않습니다.

- (선택 사항) 규정 준수 보고: 기준선에 따라 승인된 패치에 할당하려는 심각도 수준입니다(예: `Critical` 또는 `High`).

> **Note**
>
> 규정 준수 보고 수준을 지정하고 승인된 패치의 패치 상태가 `Missing`으로 보고되는 경우 패치 기준선의 전반적인 보고된 규정 준수 심각도는 사용자가 지정한 심각도 수준입니다.

10. (옵션) 승인 규칙에 따라 패치를 선택하는 대신 패치를 명시적으로 승인하려면 [패치 예외(Patch exceptions)] 섹션에서 다음을 수행합니다.

    - Approved patches(승인된 패치)에 승인할 패치의 쉼표로 구분된 목록을 입력합니다.

    > **Note**
    >
    > 승인된 패치 및 거부된 패치 목록의 승인된 형식에 대한 자세한 내용은 승인 패치 및 거부 패치 목록의 패키지 이름 형식 정보 섹션을 참조하세요.

    - (선택 사항) Approved patches compliance level(승인된 패치 규정 준수 수준)에서 규정 준수 수준을 목록의 패치에 할당합니다.

11. 승인 규칙을 준수하는 패치를 명시적으로 거부하려면 패치 예외 섹션에서 다음을 수행합니다.

    - Rejected patches(거부된 패치)에 거부할 패치의 쉼표로 구분된 목록을 입력합니다.

    > **Note**
    >
    > 승인된 패치 및 거부된 패치 목록의 승인된 형식에 대한 자세한 내용은 승인 패치 및 거부 패치 목록의 패키지 이름 형식 정보 섹션을 참조하세요.

    - [거부된 패치 작업(Rejected patches action)]에서 Patch Manager가 [거부된 패치(Rejected patches)] 목록에 포함된 패치에 대해 수행할 작업을 선택합니다.
      - [종속성으로 허용(Allow as dependency)]: [거부된 패치(Rejected patches)] 목록에 있는 패키지는 다른 패키지에 종속성을 가질 때만 설치됩니다. 이 경우 패치 기준을 준수하는 것으로 간주되고 상태가 InstalledOther로 보고됩니다. 이는 옵션을 지정하지 않은 경우의 기본 작업입니다.

- 차단: 거부된 패치 목록에 있는 패키지와 종속적으로 그것들을 포함하는 패키지는 어떠한 경우에도 Patch Manager에서 설치되지 않습니다. 패키지가 거부된 패치 목록에 추가되기 전에 설치되거나 이후에 Patch Manager 외부에 설치된 경우 패치 기준을 준수하지 않는 것으로 간주되고 상태가 InstalledRejected로 보고됩니다.

12. (선택 사항) Manage tags(태그 관리)의 경우, 하나 이상의 태그 키 이름/값 페어를 패치 기준에 적용합니다.

    태그는 리소스에 할당하는 선택적 메타데이터입니다. 태그를 사용하면 용도, 소유자 또는 환경을 기준으로 하는 등 리소스를 다양한 방식으로 분류할 수 있습니다. 예를 들어, 패치 기준에 태그를 지정하여 패치의 심각도 수준, 해당 패치가 적용되는 운영 체제 제품군 및 환경 유형을 식별할 수 있습니다. 이 경우 다음 키 이름/값 페어와 비슷한 태그를 지정할 수 있습니다.

    - `Key=PatchSeverity,Value=Critical`
    - `Key=OS,Value=RHEL`
    - `Key=Environment,Value=Production`

13. 패치 기준 생성을 선택합니다.

사용자 지정 패치 기준선 업데이트 또는 삭제

AWS Systems Manager의 기능인 Patch Manager에 생성한 사용자 정의 패치 기준선을 업데이트하거나 삭제할 수 있습니다. 패치 기준선을 업데이트할 때 이름 또는 설명, 승인 규칙 및 승인된 패치와 거부된 패치 예외를 변경할 수 있습니다. 패치 기준선에 적용된 태그를 업데이트할 수도 있습니다. 패치 기준선이 생성된 운영 체제 유형은 변경할 수 없으며 AWS에서 제공한 미리 정의된 패치 기준은 변경할 수 없습니다.

패치 기준선 업데이트 또는 삭제

다음 단계에 따라 패치 기준선을 업데이트하거나 삭제하십시오.

> **Important**
>
> Quick Setup의 패치 정책 구성에 사용될 수 있는 사용자 지정 패치 기준선을 삭제할 때는 주의해야 합니다.
> Quick Setup에서 패치 정책 구성을 사용하는 경우 사용자 지정 패치 기준선에 대한 업데이트는 한 시간에 한 번씩 Quick Setup과 동기화됩니다.
> 패치 정책에서 참조된 사용자 지정 패치 기준선이 삭제되면 해당 패치 정책의 Quick Setup Configuration details(구성 세부 정보) 페이지에 배너가 표시됩니다. 배너는 패치 정책에서 더

이상 존재하지 않는 패치 기준선을 참조하고 있으며 후속 패치 작업이 실패할 것임을 알려줍니다. 이 경우 Quick Setup Configurations(구성) 페이지로 돌아가서 Patch Manager 구성을 선택하고 Actions(작업), Edit configuration(구성 편집)을 선택합니다. 삭제된 패치 기준선 이름이 강조 표시되며, 영향을 받는 운영 체제의 새 패치 기준선을 선택해야 합니다.

패치 기준선을 업데이트 또는 삭제하는 방법

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Patch Manager를 선택합니다.
3. 업데이트 또는 삭제할 패치 기준선을 선택하고 다음 중 하나를 수행합니다.

   - AWS 계정에서 패치 기준선을 제거하려면 [삭제(Delete)]를 선택합니다. 시스템에 작업을 확인하라는 메시지가 표시됩니다.
   - 패치 기준선 이름 또는 설명, 승인 규칙 또는 패치 예외를 변경하려면 편집을 선택하십시오. 패치 기준선 편집 페이지에서 원하는 값과 옵션을 변경한 다음 변경 사항 저장을 선택하십시오.
   - 패치 기준선에 적용된 태그를 추가, 변경 또는 삭제하려면 태그 탭을 선택한 다음 태그 편집을 선택하십시오. Edit patch baseline tags(패치 기준선 태그 편집) 페이지에서 패치 기준 태그를 업데이트한 다음 변경 사항 저장을 선택합니다.

   선택 가능한 구성에 대한 자세한 내용은 사용자 정의 패치 기준 작업 섹션을 참조하세요.

기존 패치 기준을 기본값으로 설정

> **Important**
>
> 여기서 선택한 기본 패치 기준선은 패치 정책을 기반으로 하는 패치 작업에 적용되지 않습니다. 패치 정책에서는 고유한 패치 기준선 사양을 사용합니다. 패치 정책에 대한 자세한 내용은 Quick Setup 패치 정책 사용 섹션을 참조하세요.

AWS Systems Manager의 기능인 Patch Manager에 사용자 정의 패치 기준을 생성할 때, 기준을 생성하자마자 연결된 운영 체제 유형의 기본값으로 설정할 수 있습니다. 자세한 설명은 사용자 정의 패치 기준 작업을 참조하세요.

기존 패치 기준을 운영 체제 유형의 기본값으로 설정할 수도 있습니다.

> **Note**
>
> 2022년 12월 22일의 패치 정책 릴리스 이전 또는 이후에 처음 Patch Manager에 액세스했는지에 따라 사용자가 따르는 단계가 달라집니다. 해당 날짜 이전에 Patch Manager를 사용했다면 콘솔 절차를 사용할 수 있습니다. 그렇지 않으면 AWS CLI 절차를 사용합니다. 콘솔 절차에서 참조되는 작업 메뉴는 패치 정책 릴리스 이전에 Patch Manager가 사용되지 않은 리전에는 표시되지 않습니다.

패치 기준을 기본값으로 설정하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Patch Manager를 선택합니다.
3. [패치 기준(Patch baselines)] 탭을 선택합니다.
4. 패치 기준 목록에서 현재 운영 체제 유형의 기본값으로 설정되지 않은 패치 기준의 버튼을 선택합니다.

   Default baseline(기본 기준) 열은 현재 어떤 기준이 기본값으로 설정되어 있는지 나타냅니다.
5. 작업 메뉴에서 Set default patch baseline(기본 패치 기준 설정)을 선택하십시오.

   > **Important**
   >
   > 2022년 12월 22일 이전에 현재 AWS 계정 및 리전에서 Patch Manager로 작업하지 않은 경우 작업 메뉴를 사용할 수 없습니다. 자세한 내용은 이 주제 앞부분의 참고를 참조하세요.
6. 확인 대화 상자가 나타나면 Set default(기본값으로 설정)를 선택합니다.

패치 기준선을 기본값으로 설정하는 방법(AWS CLI)

1. 사용 가능한 패치 기준선과 해당 ID, Amazon 리소스 이름(ARN) 목록을 보려면 describe-patch-baselines 명령을 실행합니다.

   ```
   aws ssm describe-patch-baselines
   ```

2. 기준선을 연결된 운영 체제의 기본값으로 설정하려면 register-default-patch-baseline 명령을 실행합니다. *baseline-id-or-ARN*을 사용할 사용자 지정 패치 기준선 또는 사전 정의된 기준선의 ID로 바꿉니다.

   Linux & macOS

   ```
   aws ssm register-default-patch-baseline \
       --baseline-id baseline-id-or-ARN
   ```

   다음은 사용자 지정 기준선을 기본값으로 설정하는 예제입니다.

   ```
   aws ssm register-default-patch-baseline \
       --baseline-id pb-abc123cf9bEXAMPLE
   ```

   다음은 AWS에서 관리하는 사전 정의된 기준선을 기본값으로 설정하는 예제입니다.

   ```
   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
   ```

   다음은 사용자 지정 기준선을 기본값으로 설정하는 예제입니다.

   ```
   aws ssm register-default-patch-baseline ^
       --baseline-id pb-abc123cf9bEXAMPLE
   ```

   다음은 AWS에서 관리하는 사전 정의된 기준선을 기본값으로 설정하는 예제입니다.

   ```
   aws ssm register-default-patch-baseline ^
       --baseline-id arn:aws:ssm:us-east-2:733109147000:patchbaseline/
   pb-071da192df1226b63
   ```

사용 가능한 패치 보기

AWS Systems Manager의 기능인 Patch Manager로 지정된 운영 체제 및 특정 운영 체제 버전(옵션)에 대해 사용 가능한 패치를 모두 볼 수 있습니다.

> **Tip**
>
> 사용 가능한 패치 목록을 생성하여 파일에 저장하려면 describe-available-patches 명령을 사용하고 기본 설정 출력을 지정합니다.

사용 가능한 패치를 보려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Patch Manager를 선택합니다.
3. [패치(Patches)] 탭을 클릭합니다.

   -또는-

   현재 AWS 리전에서 처음으로 Patch Manager에 액세스한 경우 개요를 통한 시작을 선택하고 패치 탭을 선택합니다.

   > **Note**
   >
   > Windows Server의 경우 Windows Server 업데이트 서비스(WSUS)에서 사용할 수 있는 업데이트가 패치 탭에 표시됩니다.

4. [운영 체제(Operating system)]에서 사용 가능한 패치를 보려는 운영 체제(예: `Windows` 또는 `Amazon Linux`)를 선택합니다.
5. (옵션) [제품(Product)]에서 OS 버전(예 :`WindowsServer2019` 또는 `AmazonLinux2018.03`)을 선택합니다.
6. (옵션) 결과에 대한 정보 열을 추가하거나 제거하려면 [패치(Patches)] 목록 오른쪽 위에 있는 구성 버튼
   (⚙)
   을 선택합니다. (기본적으로 [패치(Patches)] 탭에는 사용 가능한 패치 메타데이터 중 일부에 대한 열만 표시됩니다.)

뷰에 추가할 수 있는 메타데이터 유형에 대한 자세한 내용은 AWS Systems Manager API Reference의 Patch를 참조하세요.

패치 그룹 작업

작업에서 패치 정책을 사용하지 않는 경우 태그를 사용하여 패치 그룹에 관리형 노드를 추가하면 패치 적용 작업을 구성할 수 있습니다.

> **Important**
>
> 패치 그룹은 패치 정책을 기반으로 하는 패치 작업에 사용되지 않습니다. 패치 정책 작업에 대한 자세한 내용은 Quick Setup 패치 정책 사용 섹션을 참조하세요.

패치 적용 작업에 태그를 사용하려면 태그 키 `Patch Group` 또는 `PatchGroup`을 관리형 노드에 적용해야 합니다. 패치 그룹에 태그 값으로 제공할 이름도 지정해야 합니다. 값을 지정하는 데는 제한이 없지만 태그 키는 `Patch Group` 또는 `PatchGroup`이어야 합니다.

EC2 인스턴스 메타데이터에서 태그를 허용한 경우 `PatchGroup`(스페이스 사용 안 함)이 필요합니다.

태그를 사용하여 관리형 노드를 그룹화한 후에 패치 기준에 패치 그룹 값을 추가해야 합니다. 패치 그룹을 패치 기준선에 등록하여 패치 적용 작업 중 올바른 패치가 설치되는지 확인할 수 있습니다. 패치 그룹에 대한 자세한 내용은 패치 그룹 정보 섹션을 참조하세요.

이 주제의 작업을 완료하여 태그를 노드 및 패치 기준선과 함께 사용하여 패치를 적용할 관리형 노드를 준비합니다. 작업 1은 Amazon EC2 인스턴스에 패치를 적용하는 경우에만 필요합니다. 작업 2는 하이브리드 및 멀티클라우드 환경에서 비 EC2 인스턴스에 패치를 적용하는 경우에만 필요합니다. 작업 3은 모든 관리형 노드에 필요합니다.

> **Tip**
>
> AWS CLI 명령 `add-tags-to-resource` 또는 Systems Manager API 작업 `AddTagsToResource`를 사용하여 관리형 노드에 태그를 추가할 수도 있습니다.

Tasks

- 태스크 1: 태그를 사용하여 패치 그룹에 EC2 인스턴스 추가

- 태스크 2: 태그를 사용하여 패치 그룹에 관리형 인스턴스 추가
- 작업 3: 패치 기준에 패치 그룹 추가

태스크 1: 태그를 사용하여 패치 그룹에 EC2 인스턴스 추가

Systems Manager 콘솔 또는 Amazon EC2 콘솔을 사용하여 EC2 인스턴스에 태그를 추가할 수 있습니다. 이 작업은 Amazon EC2 인스턴스에 패치를 적용하는 경우에만 필요합니다.

> **Important**
>
> Allow tags in instance metadata(인스턴스 메타데이터의 태그 허용) 옵션이 인스턴스에서 활성화된 경우 `Patch Group` 태그(공백 포함)를 Amazon EC2 인스턴스에 적용할 수 없습니다. 인스턴스 메타데이터에서 태그를 허용하면 태그 키 이름에 공백이 포함되지 않습니다. EC2 인스턴스 메타데이터에 태그를 허용한 경우 태그 키 `PatchGroup`(공백 없음)을 사용해야 합니다.

옵션 1: 패치 그룹에 EC2 인스턴스를 추가하는 방법(Systems Manager 콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 관리형 인스턴스 목록에서 패치 적용을 위해 구성하려는 관리형 EC2 인스턴스의 ID를 선택합니다. EC2 인스턴스의 노드 ID는 `i-`으로 시작합니다.

   > **Note**
   >
   > Amazon EC2 콘솔과 AWS CLI를 사용하는 경우 Systems Manager에 사용하도록 아직 구성되지 않은 인스턴스에 `Key = Patch Group` 또는 `Key = PatchGroup` 태그를 적용할 수 있습니다.
   > 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

4. 태그 탭을 선택한 다음에 편집을 선택합니다.
5. 왼쪽 열에 **`Patch Group`** 또는 **`PatchGroup`**을 입력합니다. EC2 인스턴스 메타데이터에 태그를 허용한 경우 `PatchGroup`(공백 없음)을 사용해야 합니다.
6. 오른쪽 열에서 패치 그룹의 이름으로 사용할 태그 값을 입력합니다.
7. Save(저장)를 선택합니다.

8. 이 절차를 반복하여 동일한 패치 그룹에 다른 EC2 인스턴스를 추가합니다.

옵션 2: 패치 그룹에 EC2 인스턴스를 추가하는 방법(Amazon EC2 콘솔)

1. Amazon EC2 콘솔을 열고 탐색 창에서 [인스턴스(Instances)]를 선택합니다.
2. 인스턴스 목록에서 패치로 구성할 인스턴스를 선택합니다.
3. 작업 메뉴에서 인스턴스 설정, 태그 관리를 선택합니다.
4. 새 태그 추가를 선택합니다.
5. Key(키)에 **`Patch Group`** 또는 **`PatchGroup`**을 입력합니다. EC2 인스턴스 메타데이터에 태그를 허용한 경우 `PatchGroup`(공백 없음)을 사용해야 합니다.
6. 값의 경우 패치 그룹의 이름으로 사용할 값을 입력합니다.
7. Save(저장)를 선택합니다.
8. 이 절차를 반복하여 동일한 패치 그룹에 다른 인스턴스를 추가합니다.

태스크 2: 태그를 사용하여 패치 그룹에 관리형 인스턴스 추가

이 주제의 단계에 따라 AWS IoT Greengrass 코어 디바이스와 비 EC2 하이브리드 정품 인증 관리형 노드에 태그를 추가합니다(mi-*). 이 작업은 하이브리드 및 멀티클라우드 환경에서 비 EC2 인스턴스에 패치를 적용하는 경우에만 필요합니다.

> **Note**
>
> Amazon EC2 콘솔을 사용하여 비 EC2 관리형 노드의 태그를 추가할 수 없습니다.

패치 그룹에 비 EC2 관리형 노드를 추가하는 방법(Systems Manager 콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 관리형 노드 목록에서 패치로 구성할 관리형 노드의 이름을 선택합니다.

> **Note**
>
> 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

4. 태그 탭을 선택한 다음에 편집을 선택합니다.
5. 왼쪽 열에 **`Patch Group`** 또는 **`PatchGroup`**을 입력합니다. EC2 인스턴스 메타데이터에 태그를 허용한 경우 `PatchGroup`(공백 없음)을 사용해야 합니다.
6. 오른쪽 열에서 패치 그룹의 이름으로 사용할 태그 값을 입력합니다.
7. Save(저장)를 선택합니다.
8. 이 절차를 반복하여 동일한 패치 그룹에 다른 관리형 노드를 추가합니다.

작업 3: 패치 기준에 패치 그룹 추가

특정 패치 기준을 관리형 노드와 연결하려면 패치 기준에 패치 그룹 값을 추가해야 합니다. 패치 그룹을 패치 기준선에 등록하여 패치 적용 작업 중 올바른 패치가 설치되는지 확인할 수 있습니다. 이 작업 EC2 인스턴스, 비 EC2 관리형 노드 또는 둘 다에 패치를 적용하든 상관없이 필요합니다.

패치 그룹에 대한 자세한 내용은 패치 그룹 정보 섹션을 참조하세요.

> **Note**
>
> 2022년 12월 22일의 패치 정책 릴리스 이전 또는 이후에 처음 Patch Manager에 액세스했는지에 따라 사용자가 따르는 단계가 달라집니다.

패치 기준선에 패치 그룹을 추가하는 방법(Systems Manager 콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Patch Manager를 선택합니다.
3. 현재 AWS 리전에서 처음으로 Patch Manager에 액세스하고 Patch Manager 시작 페이지가 열리면 개요로 시작을 선택합니다.
4. 패치 기준선 탭을 선택한 다음에 패치 기준선 목록에서 패치 그룹에 대해 구성하려는 패치 기준선의 이름을 선택합니다.

   패치 정책 릴리스 이후까지 처음 Patch Manager에 액세스하지 않았다면 직접 생성한 사용자 지정 기준선을 선택해야 합니다.
5. 기준선 ID 세부 정보 페이지에 작업 메뉴가 포함되어 있으면 다음을 수행합니다.

   - 작업, Modify patch groups(패치 그룹 수정)을 선택합니다.
   - 태스크 2: 태그를 사용하여 패치 그룹에 관리형 인스턴스 추가에서 관리형 노드에 추가한 태그 값을 입력한 다음에 추가를 선택합니다.

기준선 ID 세부 정보 페이지에 작업 메뉴가 포함되어 있지 않으면 콘솔에서 패치 그룹을 구성할 수 없습니다. 그 대신에 다음 중 하나를 수행할 수 있습니다.

- (권장) AWS Systems Manager의 기능인 Quick Setup에서 패치치 정책을 설정하여 패치 기준선을 하나 이상의 EC2 인스턴스에 매핑합니다.

  자세한 내용은 Quick Setup 패치 정책 사용 및 Quick Setup 패치 정책을 사용하여 조직 전반의 패치 적용 자동화를 참조하세요.
- AWS Command Line Interface(AWS CLI) 의 register-patch-baseline-for-patch-group명령을 사용하여 패치 그룹을 구성합니다.

Patch Manager 설정 작업

주제

- Patch Manager와 AWS Security Hub 통합

Patch Manager와 AWS Security Hub 통합

AWS Security Hub는 AWS 내의 보안 상태에 대한 포괄적인 뷰를 제공합니다. Security Hub는 AWS 계정, AWS 서비스 및 지원되는 서드 파티 파트너 제품에서 보안 데이터를 수집합니다. Security Hub를 사용하면 환경에서 보안 업계 표준 및 모범 사례를 준수하는지 확인할 수 있습니다. Security Hub는 보안 추세를 분석하고 우선순위가 가장 높은 보안 문제를 식별하는 데 도움이 됩니다.

AWS Systems Manager의 기능인 Patch Manager와 Security Hub 간 통합을 사용하여 Patch Manager에서 Security Hub로 규정 미준수 노드에 관한 조사 결과를 보낼 수 있습니다. 결과는 보안 점검 또는 보안 관련 감지의 관찰 가능한 기록입니다. 그러면 Security Hub는 보안 태세 분석에 이러한 패치 관련 결과를 포함할 수 있습니다.

다음 항목의 정보는 패치 작업에 사용하는 구성 방법 또는 유형에 관계없이 적용됩니다.

- Quick Setup에서 구성된 패치 정책
- Quick Setup에서 구성된 호스트 관리 옵션
- 패치 `Scan` 또는 `Install` 태스크를 실행하기 위한 유지 관리 기간
- 온디맨드 Patch now(지금 패치) 작업

목차

Patch Manager에서 Security Hub로 조사 결과를 보내는 방법

Security Hub의 경우 보안 문제를 조사 결과와 같이 추적합니다. 일부 결과는 다른 AWS 서비스 또는 서드 파티에서 감지한 문제에서 비롯됩니다. Security Hub에는 보안 문제를 감지하고 결과를 생성하는 데 사용하는 규칙 집합도 있습니다.

Patch Manager는 결과를 Security Hub로 보내는 Systems Manager 기능 중 하나입니다. SSM 문서(`AWS-RunPatchBaseline`, `AWS-RunPatchBaselineAssociation` 또는 `AWS-RunPatchBaselineWithHooks`)를 실행하여 패치 작업을 수행하면 패치 정보가 AWS Systems Manager의 기능인 Inventory나 Compliance 또는 둘 다로 전송됩니다. Inventory나 Compliance 또는 둘 다 데이터를 수신한 후 Patch Manager가 알림을 수신합니다. 그런 다음 Patch Manager가 데이터의 정확성, 형식 및 규정 준수 여부를 평가합니다. 모든 조건이 충족되면 Patch Manager는 데이터를 Security Hub로 전달합니다.

Security Hub는 이러한 모든 출처를 총망라하여 결과를 관리할 도구를 제공합니다. 사용자는 조사 결과 목록을 조회하고 필터링할 수 있으며 주어진 조사 결과의 세부 정보를 조회할 수도 있습니다. 자세한 내용은 AWS Security Hub User Guide의 Viewing findings를 참조하세요. 또한 주어진 결과에 대한 조사 상태를 추적할 수도 있습니다. 자세한 내용은 AWS Security Hub User Guide의 Taking action on findings를 참조하세요.

Security Hub의 모든 결과는 표준 JSON 형식을 사용합니다. 이를 AWS Security Finding Format(ASFF)이라고 합니다. ASFF에는 문제의 출처, 영향을 받은 리소스와 결과의 현재 상태 등에 관한 세부 정보가 포함됩니다. 자세한 내용은 AWS Security Hub User Guide의 AWS Security Finding Format (ASFF)을 참조하세요.

Patch Manager이(가) 보내는 결과의 유형

Patch Manager는 AWS Security Finding Format(ASFF)을 사용하여 결과를 Security Hub로 보냅니다. ASFF의 경우, Types 필드가 결과 유형을 제공합니다. Patch Manager의 결과는 Types의 값이 다음과 같습니다.

- 소프트웨어 및 구성 확인/패치 관리

Patch Manager는 규정을 준수하지 않는 관리형 노드당 하나의 검색 결과를 보냅니다. 결과는 리소스 유형 AwsEc2Instance로 보고되므로 결과는 AwsEc2Instance 리소스 유형을 보고하는 다른 Security Hub 통합과 상호 연관될 수 있습니다. Patch Manager는 작업에서 관리형 노드가 비준수임을 발견한 경우에만 결과를 Security Hub로 전달합니다. 결과에는 패치 요약 결과가 포함됩니다.

> **Note**
>
> 규정 미준수 노드를 Security Hub에 보고한 후에 노드가 규정을 준수하면 Patch Manager는 Security Hub에 업데이트를 보내지 않습니다. 필요한 패치를 관리형 노드에 적용한 후 Security Hub에서 조사 결과를 수동으로 확인할 수 있습니다.

규정 준수 정의에 대한 자세한 내용은 패치 규정 준수 상태 값 이해 섹션을 참조하세요. PatchSummary에 대한 자세한 내용은 AWS Security Hub API Reference의 PatchSummary를 참조하세요.

조사 결과 전송 지연 시간

Patch Manager가 새로운 결과를 생성하면 일반적으로 몇 초에서 2시간 이내에 Security Hub로 결과가 전송됩니다. 속도는 해당 시간에 처리 중인 AWS 리전의 트래픽에 따라 달라집니다.

Security Hub 사용할 수 없을 때 다시 시도

서비스 중단이 발생하면 AWS Lambda 기능이 실행되어 서비스가 다시 실행된 후 메시지를 기본 대기열에 다시 넣습니다. 메시지가 기본 대기열에 있으면 다시 시도는 자동으로 수행됩니다.

Security Hub를 사용할 수 없는 경우 Patch Manager는 결과가 수신될 때까지 결과 전송을 재시도합니다.

Security Hub에서 조사 결과 보기

이 절차에서는 패치 규정을 준수하지 않는 플릿의 관리형 노드에 대한 Security Hub의 조사 결과를 보는 방법을 설명합니다.

패치 규정 준수에 대한 Security Hub 조사 결과를 검토하는 방법

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/securityhub/에서 AWS Security Hub 콘솔을 엽니다.
2. 탐색 창에서 결과를 선택합니다.
3. 필터 추가() 상자를 선택합니다.
4. 메뉴의 필터에서 제품 이름을 선택합니다.
5. 이때 열리는 대화 상자의 첫 번째 필드에서 is를 선택한 다음에 두 번째 필드에서 **Systems Manager Patch Manager**를 입력합니다.
6. 적용을 선택합니다.
7. 결과 범위를 좁히는 데 도움이 되도록 원하는 추가 필터를 추가합니다.
8. 자세한 내용을 알아보려는 조사 결과의 제목을 결과 목록에서 선택합니다.

   화면 오른쪽에 리소스, 발견된 문제 및 권장 문제 해결에 대한 자세한 정보가 포함된 창이 열립니다.

> **Important**
>
> 현재 Security Hub에서는 모든 관리형 노드의 리소스 유형을 `EC2 Instance`로 보고합니다. 여기에는 Systems Manager에서 사용하려고 등록한 온프레미스 서버와 가상 머신이 포함됩니다.

심각도 분류

Systems Manager Patch Manager의 결과 목록에는 조사 결과의 심각도에 대한 보고서가 포함됩니다. 심각도 수준에는 최저부터 최고까지 다음이 포함됩니다.

- INFORMATIONAL – 찾은 문제가 없습니다.
- LOW – 문제 해결이 필요하지 않습니다.
- MEDIUM – 문제를 해결해야 하지만 긴급하지는 않습니다.
- HIGH – 우선적으로 해결해야 할 문제입니다.
- CRITICAL – 문제가 에스컬레이션되지 않도록 즉시 해결해야 합니다.

심각도는 인스턴스의 가장 심각한 규정 미준수 패키지에 의해 결정됩니다. 심각도 수준 여러 개인 패치 기준선이 여러 개 있을 수 있으므로 모든 규정 미준수 패키지 중에서 최고 심각도가 보고됩니다. 예를 들면, 패키지 A의 심각도는 "심각"이고 패키지 B의 심각도는 "낮음"인 2개의 규정 미준수 패키지가 있을 수 있습니다. "심각"이 심각도로 보고됩니다.

심각도 필드는 Patch Manager `Compliance` 필드와 직접적으로 연관됩니다. 이 필드는 규칙과 일치하는 개별 패치에 할당되도록 설정하는 필드입니다. 이 `Compliance` 필드는 개별 패치에 할당되므로 패치 요약 수준에서는 반영되지 않습니다.

관련 콘텐츠

- AWS Security Hub 사용 설명서의 조사 결과
- AWS 관리 및 거버넌스 블로그의 Patch Manager 및 Security Hub로 다중 계정 패치 규정 준수

Patch Manager의 일반적 결과

Patch Manager는 AWS Security Finding Format(ASFF)을 사용하여 결과를 Security Hub로 보냅니다.

다음은 Patch Manager의 일반적인 결과를 예시로 나타낸 것입니다.

```
{
  "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/managed-
instances/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"
  }
}
```

통합 설정 및 구성

Patch Manager와 Security Hub 통합을 사용하려면 Security Hub를 설정해야 합니다. Security Hub를 설정하는 방법에 대한 자세한 내용은 AWS Security Hub User Guide의 Setting up Security Hub를 참조하세요.

다음 절차에서는 Security Hub가 이미 활성화되어 있지만 Patch Manager 통합이 해제된 경우 Patch Manager와 Security Hub를 통합하는 방법을 설명합니다 통합이 수동으로 해제된 경우에만 이 절차를 수행합니다.

Security Hub 통합에 Patch Manager를 추가하려면

1. 탐색 창에서 Patch Manager를 선택합니다.
2. 설정 탭을 선택합니다.

   -또는-

   현재 AWS 리전에서 처음으로 Patch Manager에 액세스한 경우 개요를 통한 시작을 선택하고 설정 탭을 선택합니다.
3. [Security Hub로 내보내기(Export to Security Hub)] 섹션 아래의 [패치 규정 준수 결과가 Security Hub로 내보내지지 않음(Patch compliance findings aren't being exported to Security Hub)] 오른쪽에서 [사용(Enable)]을 선택합니다.

결과 전송을 중지하는 방법

Security Hub로 결과를 전송하는 작업을 중지하려면 Security Hub 콘솔 또는 API를 사용하면 됩니다.

자세한 내용은 AWS Security Hub 사용 설명서에서 다음 주제를 참조하세요.

- 통합에서 결과의 흐름 사용 중지 및 사용 설정(콘솔)
- 통합에서 결과의 흐름 사용 중지(Security Hub API, AWS CLI)

Patch Manager(AWS CLI) 작업

이 섹션에는 AWS Systems Manager의 기능인 Patch Manager에 대한 구성 태스크를 수행하는 데 사용할 수 있는 AWS Command Line Interface(AWS CLI) 명령의 예가 포함되어 있습니다.

AWS CLI를 사용하여 사용자 맞춤 패치 기준선에 따라 서버 환경을 패치하는 방법은 자습서: 서버 환경에 패치 적용(AWS CLI) 섹션을 참조하세요.

AWS Systems Manager 태스크에 AWS CLI 사용에 대한 자세한 내용은 AWS CLI Command Reference의 AWS Systems Manager 섹션을 참조하세요.

주제

패치 기준을 위한 AWS CLI 명령

패치 기준을 위한 샘플 명령

패치 기준선 생성

다음 명령은 릴리스되고 5일 후 Windows Server 2012 R2에 대한 필수 및 중요 보안 업데이트를 모두 승인하는 패치 기준선을 생성합니다. 또한 승인된 패치 및 거부된 패치 목록에도 패치가 지정되었습니다. 또한 패치 기준에는 프로덕션 환경용임을 나타내는 태그가 지정되어 있습니다.

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}]"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "BaselineId":"pb-0c10e65780EXAMPLE"
}
```

다른 OS 버전에 대한 사용자 지정 리포지토리가 있는 패치 기준 생성

Linux 관리형 노드에만 적용됩니다. 다음 명령은 Amazon Linux 운영 체제의 특정 버전에 사용할 패치 리포지토리를 지정하는 방법을 보여줍니다. 이 샘플에서는 Amazon Linux 2017.09에서 기본적으로 활

성화되어 있는 소스 리포지토리를 사용하지만, 관리형 노드에 대해 구성한 다른 소스 리포지토리에도 적용할 수 있습니다.

> **Note**
>
> 이 복잡한 명령을 보다 잘 설명하기 위해 여기서는 `--cli-input-json` 옵션과 외부 JSON 파일에 저장된 추가 옵션을 사용합니다.

1. `my-patch-repository.json`과 같은 이름의 JSON 파일을 만든 후 다음 내용을 파일에 추가합니다.

```
{
    "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. 파일을 저장한 디렉터리에서 다음 명령을 실행합니다.

```
aws ssm create-patch-baseline --cli-input-json file://my-patch-repository.json
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "BaselineId": "pb-0c10e65780EXAMPLE"
}
```

패치 기준선 업데이트

다음 명령은 두 개의 패치를 거부된 것으로 추가하고 하나의 패치를 기존 패치 기준선에 승인된 대로 추가합니다.

> **Note**
>
> 승인된 패치 및 거부된 패치 목록의 승인된 형식에 대한 자세한 내용은 승인 패치 및 거부 패치 목록의 패키지 이름 형식 정보 섹션을 참조하세요.

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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "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"
}
```

패치 기준선 이름 변경

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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "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"
}
```

패치 기준선 삭제

```
aws ssm delete-patch-baseline --baseline-id "pb-0c10e65780EXAMPLE"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "BaselineId":"pb-0c10e65780EXAMPLE"
}
```

모든 패치 기준선 나열

```
aws ssm describe-patch-baselines
```

시스템은 다음과 같은 정보를 반환합니다.

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

다음은 AWS 리전의 모든 패치 기준을 나열하는 또 다른 명령입니다.

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]"
```

시스템은 다음과 같은 정보를 반환합니다.

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

AWS가 제공하는 모든 패치 기준 나열

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]"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "BaselineIdentities":[
      {
         "BaselineName":"AWS-DefaultPatchBaseline",
         "DefaultBaseline":true,
         "BaselineDescription":"Default Patch Baseline Provided by AWS.",
         "BaselineId":"arn:aws:ssm:us-east-2:111122223333:patchbaseline/
pb-0c10e65780EXAMPLE"
      }
   ]
}
```

사용자의 패치 기준선 나열

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]"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "BaselineIdentities":[
      {
         "BaselineName":"Windows-Server-2012R2",
         "DefaultBaseline":false,
         "BaselineDescription":"Windows Server 2012 R2, Important and Critical security
 updates",
         "BaselineId":"pb-0c10e65780EXAMPLE"
      }
   ]
}
```

패치 기준선 표시

```
aws ssm get-patch-baseline --baseline-id pb-0c10e65780EXAMPLE
```

> **Note**
>
> 사용자 정의 패치 기준의 경우 패치 기준 ID 또는 전체 Amazon 리소스 이름(ARN)을 지정할 수 있습니다. AWS에서 제공한 패치 기준의 경우 전체 ARN을 지정해야 합니다. 예를 들면 `arn:aws:ssm:us-east-2:075727635805:patchbaseline/pb-0c10e65780EXAMPLE`입니다.

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "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"
}
```

기본 패치 기준선 받기

```
aws ssm get-default-patch-baseline --region us-east-2
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "BaselineId":"arn:aws:ssm:us-east-2:111122223333:patchbaseline/pb-0c10e65780EXAMPLE"
}
```

사용자 지정 패치 기준을 기본값으로 설정

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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "BaselineId":"pb-0c10e65780EXAMPLE"
}
```

AWS 패치 기준을 기본값으로 재설정

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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "BaselineId":"pb-0c10e65780EXAMPLE"
}
```

패치 기준선 태그 지정

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"
```

패치 기준선에 대한 태그 나열

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"
```

패치 기준선에서 태그 삭제

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"
```

패치 그룹을 위한 AWS CLI 명령

패치 그룹을 위한 샘플 명령

- 패치 그룹 생성
- 패치 그룹 '웹 서버'를 패치 기준선에 등록
- 패치 그룹 'Backend'를 AWS가 제공하는 패치 기준에 등록
- 패치 그룹 등록 표시
- 패치 기준선에서 패치 그룹 등록 취소

패치 그룹 생성

패치 작업을 쉽게 구성하려면 태그를 사용하여 관리형 노드를 패치 그룹에 추가하는 것이 좋습니다. 패치 그룹은 태그 키 `Patch Group` 또는 `PatchGroup`을 사용해야 합니다. EC2 인스턴스 메타데이터에 태그를 허용한 경우 `PatchGroup`(공백 없음)을 사용해야 합니다. 값을 지정하는 데는 제한이 없지만 태그 키는 `Patch Group` 또는 `PatchGroup`이어야 합니다. 패치 그룹에 대한 자세한 내용은 패치 그룹 정보 섹션을 참조하세요.

태그를 사용하여 관리형 노드를 그룹화한 후에 패치 기준에 패치 그룹 값을 추가해야 합니다. 패치 그룹을 패치 기준선에 등록하여 패치 적용 작업 중 올바른 패치가 설치되는지 확인할 수 있습니다.

태스크 1: 태그를 사용하여 패치 그룹에 EC2 인스턴스 추가

> **Note**
>
> Amazon Elastic Compute Cloud(Amazon EC2) 콘솔과 AWS CLI를 사용하는 경우 Systems Manager에 사용하도록 아직 구성되지 않은 인스턴스에 `Key = Patch Group` 또는 `Key = PatchGroup` 태그를 적용할 수 있습니다. `Patch Group` 또는 `Key = PatchGroup` 태그 적용 후 Patch Manager에서 예상되는 EC2 인스턴스가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 확인하세요.

다음 명령을 실행해 EC2 인스턴스에 `PatchGroup` 태그를 추가합니다.

```
aws ec2 create-tags --resources "i-1234567890abcdef0" --tags
 "Key=PatchGroup,Value=GroupValue"
```

태스크 2: 태그를 사용하여 패치 그룹에 관리형 인스턴스 추가

다음 명령을 실행해 관리형 노드에 `PatchGroup` 태그를 추가합니다.

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"
```

작업 3: 패치 기준에 패치 그룹 추가

다음 명령을 실행하여 `PatchGroup` 태그 값을 지정된 패치 기준에 연결하십시오.

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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
  "PatchGroup": "Development",
  "BaselineId": "pb-0c10e65780EXAMPLE"
}
```

패치 그룹 '웹 서버'를 패치 기준선에 등록

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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "PatchGroup":"Web Servers",
   "BaselineId":"pb-0c10e65780EXAMPLE"
```

```
}
```

패치 그룹 'Backend'를 AWS가 제공하는 패치 기준에 등록

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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "PatchGroup":"Backend",
   "BaselineId":"arn:aws:ssm:us-east-2:111122223333:patchbaseline/pb-0c10e65780EXAMPLE"
}
```

패치 그룹 등록 표시

```
aws ssm describe-patch-groups --region us-east-2
```

시스템은 다음과 같은 정보를 반환합니다.

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

패치 기준선에서 패치 그룹 등록 취소

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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "PatchGroup":"Production",
   "BaselineId":"arn:aws:ssm:us-east-2:111122223333:patchbaseline/pb-0c10e65780EXAMPLE"
}
```

패치 요약 및 세부 정보 보기를 위한 AWS CLI 명령

패치 요약 및 세부 정보 보기를 위한 샘플 명령

- 패치 기준선이 정의한 모든 패치 받기
- 분류 SECURITY 및 심각도 Critical의 AmazonLinux2018.03에 대한 모든 패치 가져오기
- MSRC 심각도가 Critical인 Windows Server 2012에 대한 모든 패치 가져오기
- 사용 가능한 모든 패치 받기
- 관리형 노드별 패치 요약 상태 받기
- 관리형 노드에 대한 패치 규정 준수 세부 정보 받기
- 패치 규정 준수 결과 보기(AWS CLI)

패치 기준선이 정의한 모든 패치 받기

> **Note**
>
> 이 명령은 Windows Server 패치 기준에만 지원됩니다.

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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "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---
```

분류 **SECURITY** 및 심각도 **Critical**의 AmazonLinux2018.03에 대한 모든 패치 가져오기

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
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "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---
```

MSRC 심각도가 **Critical**인 Windows Server 2012에 대한 모든 패치 가져오기

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
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "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---
```

사용 가능한 모든 패치 받기

```
aws ssm describe-available-patches --region us-east-2
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "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---
```

관리형 노드별 패치 요약 상태 받기

관리형 노드별 요약은 패치 그룹에 대해 다음 'NotApplicable', 'Missing', 'Failed', 'InstalledOther', 'Installed' 상태에 있는 패치의 수를 알려 줍니다.

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
```

시스템은 다음과 같은 정보를 반환합니다.


```
{
   "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---
```

관리형 노드에 대한 패치 규정 준수 세부 정보 받기

```
aws ssm describe-instance-patches --instance-id i-08ee91c0b17045407
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "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---
```

패치 규정 준수 결과 보기(AWS CLI)

단일 관리형 노드에 대한 패치 규정 준수 결과 확인

단일 관리형 노드에 대한 패치 규정 준수 결과를 보려면 AWS Command Line Interface(AWS CLI)에서 다음 명령을 실행합니다.

```
aws ssm describe-instance-patch-states --instance-id instance-id
```

*instance-id*를 i-02573cafcfEXAMPLE 또는 mi-0282f7c436EXAMPLE 형식으로 결과를 보려는 관리형 노드의 ID로 바꿉니다.

시스템은 다음과 같은 정보를 반환합니다.

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

리전의 모든 EC2 인스턴스에 대한 패치 수 요약 확인

describe-instance-patch-states는 한 번에 하나의 관리형 인스턴스에 대한 결과 검색만 지원합니다. 그러나 describe-instance-patch-states 명령에 사용자 정의 스크립트를 사용하면 보다 세부적인 보고서를 생성할 수 있습니다.

예를 들어 jq 필터 도구가 로컬 시스템에 설치된 경우 다음 명령을 실행하여 특정 AWS 리전의 어느 EC2 인스턴스가 InstalledPendingReboot 상태인지 식별할 수 있습니다.

```
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}'
```

##은 미국 동부(오하이오) 리전의 us-east-2 같이 AWS Systems Manager가 지원하는 AWS 리전의 식별자를 나타냅니다. 지원되는 ## 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

예:

```
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}'
```

시스템은 다음과 같은 정보를 반환합니다.

```
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
```

InstalledPendingRebootCount 외에도 검색할 수 있는 카운트 유형 목록은 다음과 같습니다.

- CriticalNonCompliantCount
- SecurityNonCompliantCount
- OtherNonCompliantCount
- UnreportedNotApplicableCount
- InstalledPendingRebootCount
- FailedCount

- NotApplicableCount
- InstalledRejectedCount
- InstalledOtherCount
- MissingCount
- InstalledCount

관리형 노드 스캔 및 패치를 위한 AWS CLI 명령

다음 명령을 실행하여 패치 규정 준수 여부를 검사하거나 패치를 설치한 후 패치 요약 및 세부 정보 보기를 위한 AWS CLI 명령 섹션의 명령을 사용하여 패치 상태 및 규정 준수에 대한 정보를 볼 수 있습니다.

샘플 명령

- 패치 규정 준수를 위해 관리형 노드 스캔(AWS CLI)
- 관리형 노드에 패치 설치(AWS CLI)

패치 규정 준수를 위해 관리형 노드 스캔(AWS CLI)

패치 규정 준수를 위해 특정 관리형 노드 스캔

다음 명령을 실행합니다.

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
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "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---

    }
}
```

패치 그룹 태그로 관리형 노드에서 패치 규정 준수 여부 스캔

다음 명령을 실행합니다.

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
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "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---

    }
```

```
}
```

관리형 노드에 패치 설치(AWS CLI)

특정 관리형 노드에 패치 설치

다음 명령을 실행합니다.

> **Note**
>
> 패치 설치를 완료하기 위해 필요에 따라 대상 관리형 노드가 재부팅됩니다. 자세한 내용은 AWS-RunPatchBaseline SSM 문서 정보 단원을 참조하십시오.

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
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "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---

    }
}
```

특정 패치 그룹의 관리형 노드에 패치 설치

다음 명령을 실행합니다.

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
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
    "Command": {
        "CommandId": "fa44b086-7d36-4ad5-ac8d-627ecEXAMPLE",
        "DocumentName": "AWS-RunPatchBaseline",
        "DocumentVersion": "$DEFAULT",
        "Comment": "",
        "ExpiresAfter": 1621975407.865,
        "Parameters": {
            "Operation": [
                "Install"
            ]
        },
        "InstanceIds": [],
        "Targets": [
            {
                "Key": "tag:PatchGroup",
                "Values": [
                    "Web servers"
                ]
            }
        ],
        "RequestedDateTime": 1621953207.865,
        "Status": "Pending",
        "StatusDetails": "Pending",
        "TimeoutSeconds": 600,

    ---output truncated---

    }
}
```

AWS Systems Manager Patch Manager 자습서

이 섹션의 자습서에서는 여러 패치 적용 시나리오에 AWS Systems Manager의 기능인 Patch Manager를 사용하는 방법을 보여줍니다.

주제

- 자습서: Windows 서비스 팩 설치를 위한 패치 기준선 생성(콘솔)
- 자습서: 애플리케이션 종속성 업데이트, 관리형 노드 패치 및 애플리케이션별 상태 확인 수행

- 자습서: 서버 환경에 패치 적용(AWS CLI)

자습서: Windows 서비스 팩 설치를 위한 패치 기준선 생성(콘솔)

사용자 정의 패치 기준을 만들 때 지원되는 모든 패치 또는 일부 또는 한 가지 유형의 패치만 설치하도록 지정할 수 있습니다.

Windows의 패치 기준에서 패치 업데이트를 서비스 팩으로만 제한하기 위해 `ServicePacks`를 유일한 분류 옵션으로만 선택할 수 있습니다. Windows Update 또는 WSUS(Windows Server Update Services)에서 업데이트를 사용할 수 있는 경우 AWS Systems Manager의 기능인 Patch Manager로 서비스 팩을 자동 설치할 수 있습니다.

모든 Windows 버전에 대한 서비스 팩의 설치 여부를 제어하거나 Windows 7 또는 Windows Server 2016과 같은 특정 버전에 대한 서비스 팩만 설치할지 여부를 제어하도록 패치 기준을 구성할 수 있습니다.

Windows 관리형 노드에 모든 서비스 팩을 설치하는 데만 사용할 사용자 정의 패치 기준을 만들려면 다음 절차를 따르세요.

Windows 서비스 팩 설치를 위한 패치 기준선을 생성하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Patch Manager를 선택합니다.
3. 패치 기준선 탭을 선택한 다음 패치 기준선 생성을 선택합니다.
4. Name(이름) 필드에 새로운 패치 기준 이름(예: `MyWindowsServicePackPatchBaseline`)을 입력합니다.
5. (선택 사항) 설명에 이 패치 기준에 대한 설명을 입력합니다.
6. 운영 체제에서 `Windows`를 선택합니다.
7. 이 패치 기준을 생성하는 즉시 Windows의 기본값으로 사용하려면 Set this patch baseline as the default patch baseline for Windows Server instances(이 패치 기준을 Windows Server 인스턴스의 기본 패치 기준으로 설정)를 선택하십시오.

> **Note**
>
> 이 옵션은 2022년 12월 22일에 패치 정책이 릴리스되기 전에 Patch Manager에 처음 액세스한 경우에만 사용할 수 있습니다.

기존 패치 기준을 기본값으로 설정하는 방법에 대한 자세한 내용은 기존 패치 기준을 기본값으로 설정 섹션을 참조하세요.

8. Approval rules for operating systems(운영 체제에 대한 승인 규칙) 섹션에서 필드를 사용하여 자동 승인 규칙을 한 개 이상 생성합니다.

   - 제품: 승인 규칙이 적용되는 운영 체제 버전입니다(예: `WindowsServer2012`). 지원되는 Windows 버전을 하나 또는 둘 이상 또는 모두 선택할 수 있습니다. 기본 선택은 `All`입니다.
   - Classification(분류): `ServicePacks`를 선택합니다.
   - Severity(심각도): 규칙이 적용될 패치의 심각도 값입니다. 모든 서비스 팩이 규칙에 포함되도록 하려면 `All`을 선택합니다.
   - Auto-approval: 자동 승인을 위해 패치를 선택하는 방법입니다.
     - Approve patches after a specified number of days(지정된 일 수 후 패치 승인): 패치가 릴리스 또는 업데이트된 후 자동으로 승인되기까지 Patch Manager가 대기하는 일 수입니다. 0~360의 정수를 입력할 수 있습니다. 대부분의 시나리오에서 100일 이상 기다리지 않는 것이 좋습니다.
     - Approve patches released up to a specific date(특정 날짜까지 릴리스된 패치 승인): Patch Manager가 해당 날짜 또는 그 이전에 릴리스 또는 업데이트된 모든 패치를 자동으로 적용하는 패치 릴리스 날짜입니다. 예를 들어 2023년 7월 7일을 지정하면 2023년 7월 8일 또는 그 이후에 릴리스되거나 마지막으로 업데이트된 패치가 자동으로 설치되지 않습니다.
   - (선택 사항) Compliance reporting(규정 준수 보고): 기준에서 승인된 서비스 팩에 할당할 심각도 수준입니다(예: `High`).

   > **Note**
   >
   > 규정 준수 보고 수준을 지정하고 승인된 서비스 패치의 패치 상태가 `Missing`으로 보고되는 경우 패치 기준선의 전반적인 보고된 규정 준수 심각도는 사용자가 지정한 심각도 수준입니다.

9. (선택 사항) Manage tags(태그 관리)의 경우, 하나 이상의 태그 키 이름/값 페어를 패치 기준에 적용합니다.

   태그는 리소스에 할당하는 선택적 메타데이터입니다. 태그를 사용하면 용도, 소유자 또는 환경을 기준으로 하는 등 리소스를 다양한 방식으로 분류할 수 있습니다. 서비스 팩 업데이트 전용인 이 패치 기준의 경우 다음과 같은 키-값 쌍을 지정할 수 있습니다.

   - `Key=OS,Value=Windows`

- `Key=Classification,Value=ServicePacks`

10. 패치 기준 생성을 선택합니다.

자습서: 애플리케이션 종속성 업데이트, 관리형 노드 패치 및 애플리케이션별 상태 확인 수행

대부분의 경우 관리형 노드는 최신 소프트웨어 업데이트로 패치된 후 재부팅해야 합니다. 그러나 안전 장치 없이 프로덕션 환경에서 노드를 재부팅하면 경보 호출, 잘못된 지표 데이터 기록, 데이터 동기화 중단과 같은 여러 문제가 발생할 수 있습니다.

이 자습서에서는 AWS Systems Manager 문서(SSM 문서) `AWS-RunPatchBaselineWithHooks`를 사용하여 다음을 수행하는 복잡한 다단계 패치 작업을 수행하여 이러한 문제를 피하는 방법을 보여줍니다.

1. 애플리케이션에 새로 연결 방지
2. 운영 체제 업데이트 설치
3. 애플리케이션의 패키지 종속성 업데이트
4. 시스템 다시 시작
5. 애플리케이션별 상태 확인 수행

이 예에서는 인프라를 다음과 같이 설정했습니다.

- 대상 가상 머신은 Systems Manager에 관리형 노드로 등록됩니다.
- `Iptables`는 로컬 방화벽으로 사용됩니다.
- 관리형 노드에서 호스팅되는 애플리케이션이 포트 443에서 실행되고 있습니다.
- 관리형 노드에서 호스팅되는 애플리케이션이 `nodeJS` 애플리케이션입니다.
- 관리형 노드에서 호스팅되는 애플리케이션이 pm2 프로세스 관리자에 의해 관리됩니다.
- 애플리케이션에 이미 지정된 상태 확인 엔드포인트가 있습니다.
- 애플리케이션의 상태 확인 엔드포인트에 최종 사용자 인증이 필요하지 않습니다. 엔드포인트를 사용하면 조직의 가용성 설정 요구 사항을 충족하는 상태 확인을 수행할 수 있습니다. (사용자 환경에서는 `nodeJS` 애플리케이션이 실행 중이고 요청을 수신할 수 있는지 확인하는 것으로 충분할 수 있습니다. 다른 경우에는 캐싱 계층 또는 데이터베이스 계층에 대한 연결이 이미 설정되었는지도 확인해야 할 수 있습니다.)

이 자습서의 예제는 데모용으로만 제공되며 프로덕션 환경에 있는 그대로 구현되지 않습니다. 또한 AWS-RunPatchBaselineWithHooks 문서와 함께 Systems Manager의 기능인 Patch Manager의 수명 주기 후크 기능은 다양한 다른 시나리오를 지원할 수 있습니다. 다음은 몇 가지 예제입니다.

- 지표 보고 에이전트를 패치 전에 중지하고 관리형 노드 재부팅 후 다시 시작합니다.
- 관리형 노드를 패치 전에 CRM 또는 PCS 클러스터에서 분리하고 재부팅한 후 다시 연결합니다.
- 운영 체제(OS) 업데이트가 적용된 후 관리형 노드가 재부팅되기 전에 Windows Server 시스템에서 서드 파티 소프트웨어(예: Java, Tomcat, Adobe 애플리케이션 등)를 업데이트합니다.

애플리케이션 종속성 업데이트, 관리형 노드 패치 및 애플리케이션별 상태 확인 수행

1. 다음 내용으로 사전 설치 스크립트에 대한 SSM 문서를 생성하고 이름을 NodeJSAppPrePatch로 지정합니다. *your_application*을 애플리케이션 이름으로 바꿉니다.

   이 스크립트는 새로운 수신 요청을 즉시 차단하고 패치 작업을 시작하기 전에 이미 활성화된 요청이 완료될 때까지 5초간 기다립니다. sleep 옵션에 대해 수신 요청을 완료하는 데 일반적으로 걸리는 시간보다 긴 시간(초)을 지정합니다.

   ```
   # 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
   ```

   SSM 문서 생성에 대한 자세한 내용은 SSM 문서 콘텐츠 생성 섹션을 참조하세요.

2. 설치 후 스크립트에 대해 다음 내용으로 다른 SSM 문서를 생성하여 애플리케이션 종속성을 업데이트하고 이름을 NodeJSAppPostPatch로 지정합니다. */your/application/path*를 애플리케이션의 경로로 바꿉니다.

   ```
   cd /your/application/path
   npm update
   # you can use npm-check-updates if you want to upgrade major versions
   ```

3. onExit 스크립트에 대한 다음 내용으로 다른 SSM 문서를 생성하여 애플리케이션을 다시 불러오고 상태 확인을 수행합니다. 이 SSM 문서의 이름을 NodeJSAppOnExitPatch로 지정합니다. *your_application*을 애플리케이션 이름으로 바꿉니다.

```
# 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. 다음 단계에 따라 AWS Systems Manager의 기능인 State Manager에 연결을 생성하여 작업을 실행합니다.

   1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
   2. 탐색 창에서 State Manager를 선택한 후 연결 생성을 선택합니다.
   3. [이름(Name)]에 연결의 목적을 식별하는 데 도움이 되는 이름을 입력합니다.
   4. [문서(Document)] 목록에서 AWS-RunPatchBaselineWithHooks를 선택합니다.
   5. [작업(Operation)]에서 [설치(Install)]를 선택합니다.
   6. (옵션) [스냅샷 ID(Snapshot Id)]에 작업 속도를 높이고 일관성을 보장하기 위해 생성하는 GUID를 제공합니다. 00000000-0000-0000-0000-111122223333과 같이 단순한 GUID 값을 사용할 수 있습니다.
   7. [설치 전 후크 문서 이름(Pre Install Hook Doc Name)]에 NodeJSAppPrePatch를 입력합니다.
   8. [설치 후 후크 문서 이름(Post Install Hook Doc Name)]에 NodeJSAppPostPatch를 입력합니다.
   9. [종료 시 후크 문서 이름(On ExitHook Doc Name)]에 NodeJSAppOnExitPatch를 입력합니다.

5. 대상(Targets)에서, 태그를 지정하거나, 노드를 수동으로 선택하거나, 리소스 그룹을 선택하거나, 모든 관리형 노드를 선택하여 관리형 노드를 식별할 수 있습니다.
6. [일정 지정(Specify schedule)]에서 연결을 실행할 빈도를 지정합니다. 예를 들어 관리형 노드 패치의 경우 일주일에 한 번이 일반적입니다.

7. 속도 제어(Rate control) 섹션에서 여러 관리형 노드에서 연결을 실행하는 방법을 제어하는 옵션을 선택합니다. 한 번에 관리형 노드의 일부만 업데이트되는지 확인합니다. 그렇지 않으면 전체 또는 대부분의 플릿이 한 번에 오프라인 상태가 될 수 있습니다. 속도 제어 사용에 대한 자세한 내용은 State Manager 연결에서의 대상 및 속도 제어 정보 섹션을 참조하세요.
8. (선택 사항) 출력 옵션에서 명령 출력을 파일에 저장하려면 S3 버킷에 쓰기 활성화 옆의 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

> **Note**
>
> 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아닌 관리형 노드에 할당된 인스턴스 프로파일의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

9. 연결 생성을 선택합니다.

자습서: 서버 환경에 패치 적용(AWS CLI)

다음 절차는 사용자 지정 패치 기준, 패치 그룹 및 유지 관리 기간을 사용하여 서버 환경에 패치를 적용하는 방법에 대해 설명합니다.

시작하기 전 준비 사항

- 관리형 노드에서 SSM Agent 설치 또는 업데이트 Linux 관리형 노드에 패치를 실행하려면 노드가 SSM Agent 버전 2.0.834.0 이상을 실행 중이어야 합니다. 자세한 내용은 Run Command를 사용하여 SSM Agent 업데이트 단원을 참조하십시오.
- AWS Systems Manager의 기능인 Maintenance Windows에 대한 역할 및 권한을 구성합니다. 자세한 내용은 Maintenance Windows 설정 단원을 참조하십시오.
- 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

  자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.

Patch Manager 및 패치 관리형 노드 구성(명령줄)

1. 다음 명령을 실행하여 이름이 `Production-Baseline`인 Windows 패치 기준을 생성합니다. 이 패치 기준선은 프로덕션 환경의 패치가 릴리스되거나 마지막으로 업데이트되고 7일 후에 승인합니다. 패치 기준에는 프로덕션 환경용임을 나타내는 태그가 지정되어 있습니다.

> **Note**
>
> `OperatingSystem` 파라미터 및 `PatchFilters`는 패치 기준이 적용되는 대상 관리형 노드의 운영 체제에 따라 달라집니다. 자세한 내용은 OperatingSystem 및 PatchFilter를 참조하십시오.

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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "BaselineId":"pb-0c10e65780EXAMPLE"
}
```

2. 다음 명령을 실행하여 두 패치 그룹에 대해 "Production-Baseline" 패치 기준을 등록합니다. 그룹의 이름은 "Database Servers" 및 "Front-End Servers"로 지정됩니다.

   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"
   ```

   시스템은 다음과 같은 정보를 반환합니다.

   ```
   {
      "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"
   ```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "PatchGroup":"Front-End Servers",
   "BaselineId":"pb-0c10e65780EXAMPLE"
}
```

3. 다음 명령을 실행하여 프로덕션 서버에 대한 두 개의 유지 관리 기간을 생성합니다. 첫 번째 기간은 매주 화요일 오후 10시에 실행합니다. 두 번째 Window는 매주 토요일 오후 10시에 실행합니다. 또한 유지 관리 기간에는 프로덕션 환경용임을 나타내는 태그가 지정되어 있습니다.

   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
   ```

   시스템은 다음과 같은 정보를 반환합니다.

   ```
   {
      "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
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "WindowId":"mw-9a8b7c6d5eEXAMPLE"
}
```

4. 다음 명령을 실행하여 `Database` 및 `Front-End` 서버 패치 그룹을 해당 유지 관리 기간에 등록합니다.

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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "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"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "WindowTargetId":"faa01c41-1d57-496c-ba77-ff9caEXAMPLE"
}
```

5. 다음 명령을 실행하여 각 유지 관리 기간 동안 `Database` 및 `Front-End` 서버에 누락된 업데이트를 설치하는 패치 작업을 등록합니다.

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}}"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "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}}"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "WindowTaskId":"8a5c4629-31b0-4edd-8aea-33698EXAMPLE"
}
```

6. 다음 명령을 실행하여 패치 그룹에 대한 높은 수준의 패치 규정 준수 요약을 확인합니다. 상위 수준의 패치 규정 준수 요약에는 각 패치 상태의 패치가 있는 관리형 노드 수가 포함됩니다.

> **Note**
>
> 첫 번째 유지 관리 기간 동안 패치 태스크가 실행될 때까지 요약에는 관리형 노드 수가 0으로 표시됩니다.

Linux & macOS

```
aws ssm describe-patch-group-state \
```

```
    --patch-group "Database Servers"
```

Windows Server

```
aws ssm describe-patch-group-state ^
    --patch-group "Database Servers"
```

시스템은 다음과 같은 정보를 반환합니다.

```
{
   "Instances": number,
   "InstancesWithFailedPatches": number,
   "InstancesWithInstalledOtherPatches": number,
   "InstancesWithInstalledPatches": number,
   "InstancesWithInstalledPendingRebootPatches": number,
   "InstancesWithInstalledRejectedPatches": number,
   "InstancesWithMissingPatches": number,
   "InstancesWithNotApplicablePatches": number,
   "InstancesWithUnreportedNotApplicablePatches": number
}
```

7. 다음 명령을 실행하여 패치 그룹에 대한 관리형 노드별 패치 요약 상태를 확인합니다. 관리형 노드별 요약에는 패치 그룹에 대한 관리형 노드당 각 패치 상태의 여러 패치가 포함됩니다.

   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"
   ```

   시스템은 다음과 같은 정보를 반환합니다.

   ```
   {
      "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
        }
    ]
}
```

Patch Manager 구성 태스크에 사용할 수 있는 다른 AWS CLI 명령의 예를 보려면 Patch Manager(AWS CLI) 작업 섹션을 참조하세요.

Patch Manager 문제 해결

다음 정보를 사용하면 AWS Systems Manager의 기능인 Patch Manager 관련 문제를 해결하는 데 도움이 됩니다.

주제

- 문제: baseline_overrides.json에 대한 "Invoke-PatchBaselineOperation: 액세스 거부됨" 오류 또는 "S3에서 파일을 다운로드할 수 없음" 오류
- 문제: 명백한 원인 또는 오류 메시지 없이 패치 적용 실패
- 문제: 예기치 않은 패치 규정 준수 결과
- Linux에서 AWS-RunPatchBaseline 실행 시 오류
- Windows Server에서 AWS-RunPatchBaseline 실행 시 오류
- AWS Support 문의

문제: **baseline_overrides.json**에 대한 "Invoke-PatchBaselineOperation: 액세스 거부됨" 오류 또는 "S3에서 파일을 다운로드할 수 없음" 오류

문제: 패치 정책에 따라 지정된 패치 적용 작업을 실행하면 다음 예제와 비슷한 오류가 표시됩니다.

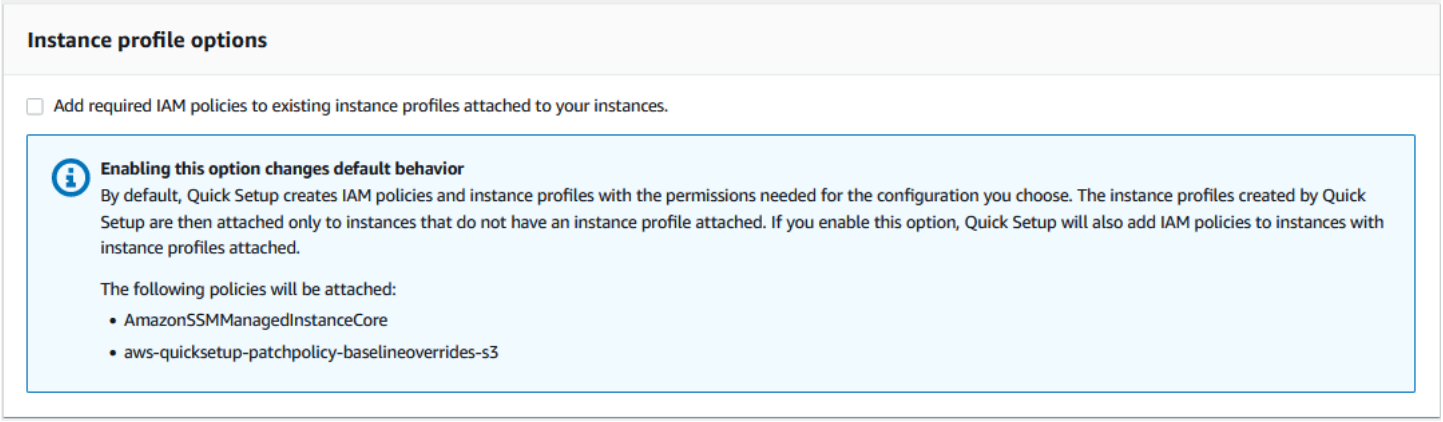
Example error on Windows Server

```
----------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.
```

원인: Quick Setup에서 패치 정책을 생성했으며, 연결된 인스턴스 프로파일(EC2 인스턴스의 경우) 또는 연결된 서비스 역할(비 EC2 시스템의 경우)이 이미 일부 관리형 노드에 있었습니다. 그러나 다음 이미지와 같이 인스턴스에 연결된 기존 인스턴스 프로파일에 필수 IAM 정책 추가 확인란을 선택하지 않았습니다.



패치 정책을 생성하면 정책의 구성 `baseline_overrides.json` 파일을 저장하는 Amazon S3 버킷도 생성됩니다. 정책을 생성할 때 인스턴스에 연결된 기존 인스턴스 프로파일에 필수 IAM 정책 추가 확인란을 선택하지 않으면 S3 버킷의 `baseline_overrides.json`에 액세스하는 데 필요한 IAM 정책 및 리소스 태그가 기존 IAM 인스턴스 프로파일 및 서비스 역할에 자동으로 추가되지 않습니다.

솔루션 1: 기존 패치 정책 구성을 삭제한 다음에 대체 패치 정책 구성을 생성하고, 인스턴스에 연결된 기존 인스턴스 프로파일에 필수 IAM 정책 추가 확인란을 반드시 선택합니다. 이렇게 선택하면 연결된 인스턴스 프로파일 또는 서비스 역할이 이미 있는 노드에 이 Quick Setup 구성을 통해 생성된 IAM 정책이 적용됩니다. (기본적으로 Quick Setup에서는 인스턴스 프로파일 또는 서비스 역할이 아직 없는 인스턴스와 노드에 필수 정책을 추가합니다.) 자세한 내용은 Quick Setup 패치 정책을 사용하여 조직 전반의 패치 적용 자동화를 참조하세요.

솔루션 2: Quick Setup에서 사용하는 각 IAM 인스턴스 프로파일 및 IAM 서비스 역할에 필수 권한과 태그를 수동으로 추가합니다. 지침은 패치 정책 S3 버킷에 대한 권한 섹션을 참조하세요.

문제: 명백한 원인 또는 오류 메시지 없이 패치 적용 실패

문제: 오류 메시지가 반환되지 않고 패치 적용 작업이 실패합니다.

가능한 원인: `AWS-RunPatchBaseline` 간접 호출이 한 번에 두 번 이상 발생하면 간접 호출이 서로 충돌하여 패치 적용 작업이 실패할 수 있습니다. 패치 적용 로그에는 이 내용이 표시되지 않을 수 있습니다.

동시 패치 적용 작업이 서로 방해될 수 있었는지 확인하려면 AWS Systems Manager의 기능인 Run Command에서 명령 기록을 검토합니다. 패치 적용에 실패한 관리형 노드의 경우 여러 작업에서 서로 2분 이내에 시스템 패치 적용이 시도되었는지 확인합니다. 이 시나리오가 오류의 원인일 수도 있습니다.

다음 명령을 사용하면 AWS Command Line Interface(AWS CLI)를 사용하여 동시 패치 시도를 확인할 수도 있습니다. *`node-id`* 값을 관리형 노드의 ID로 바꿉니다.

```
aws ssm list-commands \
    --filter "key=DocumentName,value=AWS-RunPatchBaseline" \
    --query 'Commands[*].
{CommandId:CommandId,RequestedDateTime:RequestedDateTime,Status:Status}' \
    --instance-id node-id \
    --output table
```

솔루션: 동일한 관리형 노드에서 경쟁하는 패치 적용 작업 때문에 패치 적용에 실패했다고 판단되면 이 오류가 다시 발생하지 않도록 패치 적용 구성을 조정합니다. 예를 들어, 두 유지 관리 기간에 겹치는 패치 적용 시간이 지정되어 있는 경우 둘 중 하나를 제거하거나 수정합니다. 유지 관리 기간에는 하나의 패치 적용 작업이 지정되어 있지만 패치 정책에는 같은 시간에 다른 패치 적용 작업이 지정되어 있으면 유지 관리 기간에서 작업을 제거하는 것이 좋습니다.

이 시나리오에서 충돌하는 패치 적용 작업이 실패의 원인이 아니었다고 판단되면 AWS Support에 문의하는 것이 좋습니다.

문제: 예기치 않은 패치 규정 준수 결과

문제: `Scan` 작업 후에 생성된 패치 규정 준수 세부 정보를 검토할 때, 결과에 패치 기준선에 설정된 규칙에 부합하지 않는 정보가 포함되어 있습니다. 패치 기준선의 Rejected patches(거부된 패치) 목록에 추가한 예외가 `Missing`으로 표시되는 경우를 예로 들 수 있습니다. 또는 패치 기준선에서 `Critical` 패치만 지정했는데도 `Important`로 분류된 패치가 누락된 것으로 표시됩니다.

원인: Patch Manager는 현재 다양한 `Scan` 작업 실행 방법을 지원합니다.

- Quick Setup에서 구성된 패치 정책
- Quick Setup에서 구성된 호스트 관리 옵션
- 패치 `Scan` 또는 `Install` 태스크를 실행하기 위한 유지 관리 기간
- 온디맨드 Patch now(지금 패치) 작업

`Scan` 작업이 실행되면 가장 최근 검사의 규정 준수 세부 정보를 덮어씁니다. `Scan` 작업을 실행하도록 설정된 방법이 2가지 이상이고, 해당 방법에서 서로 다른 규칙의 서로 다른 패치 기준선을 사용하는 경우 패치 규정 준수 결과가 달라집니다.

해결 방법: 예기치 않은 패치 규정 준수 결과를 방지하려면 Patch Manager `Scan` 작업을 실행할 때 한 번에 한 가지 방법만 사용하는 것이 좋습니다. 자세한 내용은 의도치 않은 패치 규정 준수 데이터 덮어쓰기 방지 단원을 참조하십시오.

Linux에서 **AWS-RunPatchBaseline** 실행 시 오류

주제

문제: '해당 파일 또는 디렉터리가 없음(No such file or directory)' 오류

문제: `AWS-RunPatchBaseline` 실행 시 다음 오류 중 하나와 함께 패치가 실패합니다.

```
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
```

원인 1: AWS-RunPatchBaseline을 실행하는 2개의 명령이 동일한 관리형 노드에서 동시에 실행되었습니다. 이로 인해 임시 file patch-baseline-operations*가 제대로 생성 또는 액세스되지 않는 경쟁 조건이 발생합니다.

원인 2: /var 디렉터리의 저장 공간이 부족합니다.

해결 방법 1: 유지 관리 기간에 동일한 우선순위 수준으로 AWS-RunPatchBaseline을 실행하고 동일한 대상 ID에서 실행되는 둘 이상의 Run Command 태스크가 없는지 확인합니다. 이 경우 우선순위를 다시 지정합니다. Run Command는 AWS Systems Manager의 기능입니다.

해결 방법 2: 한 번에 하나의 유지 관리 기간만 동일한 대상 및 동일한 일정에서 AWS-RunPatchBaseline을 사용하는 Run Command 태스크를 실행하고 있는지 확인합니다. 이 경우 일정을 변경합니다.

해결 방법 3: 하나의 State Manager 연결만 동일한 일정으로 AWS-RunPatchBaseline을 실행하고 동일한 관리형 노드를 대상으로 하는지 확인합니다. State Manager는 AWS Systems Manager의 기능입니다.

해결 방법 4: 업데이트 패키지를 위해 /var 디렉터리에 충분한 저장 공간을 확보합니다.

문제: '다른 프로세스가 yum 잠금을 획득함(another process has acquired yum lock)' 오류

문제: AWS-RunPatchBaseline 실행 시 다음 오류와 함께 패치가 실패합니다.

```
12/20/2019 21:41:48 root [INFO]: another process has acquired yum lock, waiting 2 s and
 retry.
```

원인: AWS-RunPatchBaseline 문서가 이미 다른 작업에서 실행 중이고 패키지 관리자 yum 프로세스를 획득한 관리형 노드에서 실행을 시작했습니다.

해결 방법: 일정에 따라 AWS-RunPatchBaseline을 실행하는 State Manager 연결, 유지 관리 기간 태스크 또는 기타 구성이 거의 같은 시간에 동일한 관리형 노드를 대상으로 하지 않도록 합니다.

문제: '권한 거부됨/명령 실행 실패(Permission denied / failed to run commands)'오류

문제: AWS-RunPatchBaseline 실행 시 다음 오류와 함께 패치가 실패합니다.

```
sh:
```

```
/var/lib/amazon/ssm/instanceid/document/orchestration/commandid/PatchLinux/_script.sh:
 Permission denied
failed to run commands: exit status 126
```

원인: /var/lib/amazon/이 noexec 권한으로 탑재되었을 수 있습니다. 이것은 SSM Agent가 /var/lib/amazon/ssm에 페이로드 스크립트를 다운로드하고 해당 위치에서 실행하기 때문에 문제입니다.

해결 방법: /var/log/amazon 및 /var/lib/amazon에 대한 배타적 파티션을 구성하고 exec 권한으로 탑재했는지 확인합니다.

문제: '페이로드를 다운로드할 수 없음(Unable to download payload)' 오류

문제: AWS-RunPatchBaseline 실행 시 다음 오류와 함께 패치가 실패합니다.

```
Unable to download payload: https://s3.DOC-EXAMPLE-BUCKET.region.amazonaws.com/
aws-ssm-region/patchbaselineoperations/linux/payloads/patch-baseline-operations-
X.XX.tar.gz.failed to run commands: exit status 156
```

원인: 지정된 Amazon Simple Storage Service(Amazon S3) 버킷에 액세스하는 데 필요한 권한이 관리형 노드에 없습니다.

해결 방법: S3 엔드포인트에 연결할 수 있도록 네트워크 구성을 업데이트합니다. 자세한 내용은 AWS 관리형 S3 버킷과 SSM Agent 통신의 Patch Manager에 대한 S3 버킷에 대한 필수 액세스에 대한 정보를 참조하세요.

문제: '지원되지 않는 패키지 관리자 및 python 버전 조합(unsupported package manager and python version combination)' 오류

문제: AWS-RunPatchBaseline 실행 시 다음 오류와 함께 패치가 실패합니다.

```
An unsupported package manager and python version combination was found. Apt requires
 Python3 to be installed.
failed to run commands: exit status 1
```

원인: Debian Server, Raspberry Pi OS 또는 Ubuntu Server 인스턴스에 지원되는 python3 버전이 설치되어 있지 않습니다.

솔루션: Debian Server, Raspberry Pi OS 및 Ubuntu Server 관리형 노드에 필요한 지원되는 python3 버전(3.0~3.10)을 서버에 설치합니다.

문제: Patch Manager가 특정 패키지를 제외하도록 지정된 규칙을 적용하지 않음

문제: `/etc/yum.conf` 파일에 `exclude=`*`package-name`* 형식으로 지정하여 특정 패키지를 제외하려고 했지만 Patch Manager `Install` 작업 중에 제외되지 않았습니다.

원인: Patch Manager는 `/etc/yum.conf` 파일에 지정된 제외를 포함하지 않습니다.

해결 방법: 특정 패키지를 제외하려면 사용자 정의 패치 기준을 생성하고 설치하지 않으려는 패키지를 제외하는 규칙을 생성합니다.

문제: 패치가 실패하고 Patch Manager가 TLS에 대한 서버 이름 표시 확장을 사용할 수 없다고 보고함

문제: 패치 작업이 다음 메시지를 표시합니다.

```
/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
```

원인: 이 메시지는 오류를 나타내지 않습니다. 대신 운영 체제와 함께 배포된 이전 버전의 Python이 TLS 서버 이름 표시를 지원하지 않는다는 경고입니다. Systems Manager 패치 페이로드 스크립트는 SNI를 지원하는 AWS API에 연결할 때 이 경고를 표시합니다.

해결 방법: 이 메시지가 보고될 때 패치 실패 문제를 해결하려면 `stdout` 및 `stderr` 파일의 내용을 검토합니다. 이러한 파일을 S3 버킷 또는 Amazon CloudWatch Logs 저장하도록 패치 기준선을 구성하지 않은 경우 Linux 관리형 노드의 다음 위치에서 파일을 찾을 수 있습니다.

`/var/lib/amazon/ssm/`*`instance-id`*`/document/orchestration/`*`Run-Command-execution-id`*`/awsrunShellScript/PatchLinux`

문제: Patch Manager가 '시도할 미러가 더 이상 없음(No more mirrors to try)' 보고

문제: 패치 작업이 다음 메시지를 표시합니다.

```
[Errno 256] No more mirrors to try.
```

원인: 관리형 노드에 구성된 리포지토리가 제대로 작동하지 않습니다. 이에 대한 가능한 원인은 다음과 같습니다.

- yum 캐시가 손상되었습니다.
- 네트워크 관련 문제로 인해 리포지토리 URL에 연결할 수 없습니다.

해결 방법: Patch Manager는 관리형 노드의 기본 패키지 관리자를 사용하여 패치 작업을 수행합니다. 리포지토리가 제대로 구성되고 작동하는지 다시 확인합니다.

문제: 'curl에서 반환된 오류 코드는 23'이라는 메시지와 함께 패치 적용 실패

문제: 다음과 비슷한 오류로 AWS-RunPatchBaseline을 사용하는 패치 적용 작업이 실패합니다.

```
05/01/2023 17:04:30 root [ERROR]: Error code returned from curl is 23
```

원인: 파일 시스템에 쓰는 데 필요한 권한이 시스템에서 사용 중인 curl 도구에 없습니다. 이는 패키지 관리자의 기본 curl 도구가 다른 버전(예: snap으로 설치된 도구)으로 대체된 경우에 발생할 수 있습니다.

솔루션: 다른 버전이 설치되었을 때 패키지 관리자가 제공한 curl 버전이 제거되었다면 다시 설치합니다.

여러 curl 버전을 설치된 상태로 유지해야 하는 경우 패키지 관리자와 연결된 버전이 PATH 변수에 나열된 첫 번째 디렉터리에 있는지 확인합니다. echo $PATH 명령을 실행하여 시스템에서 실행 파일을 검사한 디렉터리의 현재 순서를 참조하면 이를 확인할 수 있습니다.

문제: 'rpm 패키지 압축 해제 오류...'라는 메시지와 함께 패치 적용 실패

문제: 다음과 비슷한 오류로 패치 적용 작업이 실패합니다.

```
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
```

원인 1: 특정 패키지가 여러 패키지 설치 관리자에 있으면(예: pip 및 yum 둘 다 또는 dnf) 기본 패키지 관리자를 사용할 때 충돌이 발생할 수 있습니다.

pip, yum 및 dnf에서 찾을 수 있는 일반적인 예가 urllib3 패키지에서 발생합니다.

원인 2: python-urllib3 패키지가 손상되었습니다. rpm이 이전에 yum 또는 dnf를 통해 설치된 패키지였던 이후에 pip을 통해 패키지 파일이 설치되거나 업데이트되었으면 이 문제가 발생할 수 있습니다.

솔루션: 기본 패키지 관리자(yum 또는 dnf)에서만 패키지가 유지되도록 sudo pip uninstall urllib3 명령을 실행하여 pip에서 python-urllib3 패키지를 제거합니다.

문제: '패키지를 다운로드하는 동안 오류가 발생했습니다'라는 메시지와 함께 패치 적용 실패

문제: 패치를 적용하는 동안 다음과 비슷한 오류가 표시됩니다.

```
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',
```

원인: 이 오류는 관리형 노드에 사용할 수 있는 메모리가 부족할 때 발생할 수 있습니다.

솔루션: 스왑 메모리를 구성하거나 인스턴스를 다른 유형으로 업그레이드하여 메모리 지원을 늘립니다. 그런 다음에 새 패치 적용 작업을 시작합니다.

문제: '퍼블릭 키를 사용할 수 없어 다음 서명을 확인할 수 없습니다'라는 메시지와 함께 패치 적용 실패

문제: 다음과 비슷한 오류로 Ubuntu Server에서 패치 적용에 실패합니다.

```
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
```

원인: GPG(GNU Privacy Guard) 키가 만료되었거나 누락되었습니다.

솔루션: GPG 키를 새로 고치거나 키를 다시 추가합니다.

예를 들어 이전에 표시된 오류를 사용하면 467B942D3A79BD29 키가 누락되었으며 추가해야 한다는 것을 알 수 있습니다. 이렇게 하려면 다음과 같은 명령 중 하나를 실행합니다.

```
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
```

또는 모든 키를 새로 고치려면 다음 명령을 실행합니다.

```
sudo apt-key adv --keyserver hkps://keyserver.ubuntu.com --refresh-keys
```

이후에도 오류가 다시 발생하면 리포지토리를 유지 관리하는 조직에 문제를 보고하는 것이 좋습니다. 수정이 가능할 때까지 패치 적용 프로세스 중에 `/etc/apt/sources.list` 파일을 편집하여 리포지토리를 생략할 수 있습니다.

이렇게 하려면 편집할 `sources.list` 파일을 열고, 리포지토리의 줄을 찾고, 줄 시작 부분에 # 문자를 삽입하여 주석을 추가합니다. 그런 다음에 파일을 저장하고 닫습니다.

문제: 'NoMoreMirrorsRepoError' 메시지와 함께 패치 적용 실패

문제: 다음과 비슷한 오류가 표시됩니다.

```
NoMoreMirrorsRepoError: failure: repodata/repomd.xml from pgdg94: [Errno 256] No more
 mirrors to try.
```

원인: 소스 리포지토리에 오류가 있습니다.

솔루션: 리포지토리를 유지 관리하는 조직에 문제를 보고하는 것이 좋습니다. 오류가 수정될 때까지 운영 체제 수준에서 리포지토리를 비활성화할 수 있습니다. 이렇게 하려면 다음 명령을 실행하여 *repo-name*의 값을 리포지토리 이름으로 바꿉니다.

```
yum-config-manager --disable repo-name
```

다음은 한 예입니다.

```
yum-config-manager --disable pgdg94
```

이 명령을 실행한 후 다른 패치 적용 작업을 실행합니다.

문제: '페이로드를 다운로드할 수 없음' 메시지와 함께 패치 적용 실패

문제: 다음과 비슷한 오류가 표시됩니다.

```
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
```

원인: 관리형 노드 구성에 오류가 있거나 불완전합니다.

솔루션: 관리형 노드에 다음이 구성되었는지 확인합니다.

- 보안 그룹의 아웃바운드 TCP 443 규칙.
- NACL의 송신 TCP 443 규칙.
- NACL의 수신 TCP 1024-65535 규칙.
- S3 엔드포인트에 대한 연결을 제공하는 라우팅 테이블의 NAT/IGW 인스턴스에 인터넷 액세스 권한이 없으면 S3 엔드포인트와 연결을 제공합니다. 이렇게 하려면 S3 게이트웨이 엔드포인트를 VPC에서 추가하고 관리형 노드의 라우팅 테이블과 통합합니다.

문제: '설치 오류: dpkg: 오류: dpkg 프런트엔드가 다른 프로세스에 의해 잠겼습니다'라는 메시지와 함께 패치 적용 실패

문제: 다음과 비슷한 오류로 패치 적용에 실패합니다.

```
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
```

원인: 운영 체제 수준에서 관리형 노드의 다른 프로세스가 이미 패키지 관리자에서 실행되고 있습니다. 다른 프로세스를 완료하는 데 시간이 오래 걸리는 경우 Patch Manager 패치 적용 작업이 시간 초과로 실패할 수 있습니다.

솔루션: 패키지 관리자를 사용 중인 다른 프로세스가 완료된 후 새 패치 적용 작업을 실행합니다.

문제: 'dpkg가 중단되었습니다' 오류로 Ubuntu Server의 패치 적용 실패

문제: Ubuntu Server에서 다음과 비슷한 오류로 패치 적용에 실패합니다.

```
E: dpkg was interrupted, you must manually run
'dpkg --configure -a' to correct the problem.
```

원인: 하나 이상의 패키지가 잘못 구성되었습니다.

해결 방법: 다음 단계를 수행합니다.

1. 다음과 같은 명령을 한 번에 하나씩 실행하여 영향을 받는 패키지와 각 패키지의 문제를 확인합니다.

```
sudo apt-get check
```

```
sudo dpkg -C
```

```
dpkg-query -W -f='${db:Status-Abbrev} ${binary:Package}\n' | grep -E ^.[^nci]
```

2. 다음 명령을 실행하여 문제가 있는 패키지를 수정합니다.

```
sudo dpkg --configure -a
```

3. 이전 명령으로 문제가 완전히 해결되지 않으면 다음 명령을 실행합니다.

```
sudo apt --fix-broken install
```

문제: 패키지 관리자 유틸리티를 통해 패키지 종속성을 해결할 수 없음

문제: 관리형 노드의 네이티브 패키지 관리자를 통해 패키지 종속성을 해결할 수 없어 패치 적용에 실패합니다. 다음 오류 메시지 예제는 yum을 패키지 관리자로 사용하는 운영 체제의 이 오류 유형을 나타냅니다.

```
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']
```

원인: Linux 운영 체제의 Patch Manager에서는 시스템의 네이티브 패키지 관리자를 사용하여 패치 적용 작업을 실행합니다(예: yum, dnf, apt 및 zypper). 애플리케이션에서는 필요에 따라 종속 패키지를 자동으로 감지, 설치, 업데이트 또는 제거합니다. 그러나 다음과 같은 일부 조건에서는 패키지 관리자가 종속성 작업을 완료하지 못할 수 있습니다.

- 충돌하는 리포지토리가 운영 체제에 여러 개 구성되어 있습니다.
- 네트워크 관련 문제로 인해 원격 리포지토리 URL에 액세스할 수 없습니다.
- 리포지토리에서 잘못된 아키텍처에 대한 패키지를 찾았습니다.

솔루션: 아주 다양한 사유의 종속성 문제 때문에 패치 적용에 실패할 수 있습니다. 따라서 AWS Support에 문의하여 문제 해결 도움을 받는 것이 좋습니다.

Windows Server에서 **AWS-RunPatchBaseline** 실행 시 오류

주제

문제: 일치하지 않는 제품군/제품 페어

문제: Systems Manager 콘솔에서 패치 기준을 생성할 때 제품군 및 제품을 지정합니다. 예를 들면 다음을 선택할 수 있습니다.

- 제품군: `Office`

  제품: `Office 2016`

원인: 일치하지 않는 제품군/제품 페어로 패치 기준을 만들려고 하면 오류 메시지가 표시됩니다. 가능한 이유는 다음과 같습니다.

- 유효한 제품군 및 제품 페어를 선택했지만 제품군 선택을 제거했습니다.
- [사용 가능 및 일치하는 옵션(Available and matching options)] 하위 목록 대신 [폐기되었거나 일치하지 않는 옵션(Obsolete or mismatched options)] 하위 목록에서 제품을 선택했습니다.

  실수로 제품 [사용되지 않거나 일치하지 않는 옵션(Obsolete or mismatched options)] 하위 목록의 항목이 SDK 또는 AWS Command Line Interface(AWS CLI) `create-patch-baseline` 명령을 통해 입력되었을 수 있습니다. 오타가 있거나 제품이 잘못된 제품군에 할당되었다는 뜻일 수 있습니다. 이전 패치 기준에 지정되었지만 Microsoft에서 제공하는 패치가 없는 경우에도 [사용되지 않거나 일치하지 않는 옵션(Obsolete or mismatched options)] 하위 목록에 제품이 포함됩니다.

해결 방법: 콘솔에서 이 문제를 방지하려면 항상 [현재 사용 가능한 옵션(Currently available options)] 하위 목록에서 옵션을 선택합니다.

AWS CLI에서 describe-patch-properties 명령을 사용하거나 DescribePatchProperties 명령을 사용하여 사용 가능한 패치가 있는 제품을 볼 수도 있습니다.

문제: **AWS-RunPatchBaseline** 출력에서 **HRESULT**(Windows Server) 반환

문제: 다음과 같은 오류가 표시됩니다.

```
----------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
```

원인: 이 출력은 기본 Windows Update API가 패치 작업을 실행할 수 없음을 나타냅니다.

해결 방법: 다음 microsoft.com 항목에서 HResult 코드를 확인하여 오류 해결을 위한 문제 해결 단계를 식별합니다.

- 구성 요소별 Windows 업데이트 오류 코드
- Windows 업데이트의 일반적인 오류 및 해결 방법

문제: 관리형 노드에 Windows 업데이트 카탈로그 또는 WSUS에 대한 액세스 권한이 없습니다.

문제: 다음과 같은 오류가 표시됩니다.

```
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----
```

원인: 이 오류는 Windows Update 구성 요소, Windows Update 카탈로그 또는 Windows Server Update Services(WSUS)에 대한 연결 부족과 관련이 있을 수 있습니다.

해결 방법: 관리형 노드가 인터넷 게이트웨이, NAT 게이트웨이 또는 NAT 인스턴스를 통해 Microsoft Update 카탈로그에 연결되어 있는지 확인합니다. WSUS를 사용하는 경우 관리형 노드가 환경의 WSUS 서버에 연결되어 있는지 확인합니다. 의도한 대상에 연결할 수 있는 경우 Microsoft 설명서에서 `HResult 0x80072EE2`의 다른 잠재적 원인을 확인합니다. 이는 운영 체제 수준 문제를 나타낼 수 있습니다.

문제: PatchBaselineOperations PowerShell 모듈을 다운로드할 수 없음

문제: 다음과 같은 오류가 표시됩니다.

```
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
```

해결 방법: Amazon Simple Storage Service(Amazon S3)에 대한 관리형 노드 연결 및 권한을 확인합니다. 관리형 노드의 AWS Identity and Access Management(IAM) 역할이 AWS 관리형 S3 버킷과 SSM Agent 통신에 언급된 최소 권한을 사용해야 합니다. 노드는 Amazon S3 게이트웨이 엔드포인트, NAT 게이트웨이 또는 인터넷 게이트웨이를 통해 Amazon S3 엔드포인트와 통신해야 합니다. AWS Systems ManagerSSM Agent (SSM Agent)에 대한 VPC 엔드포인트 요구 사항에 대한 자세한 내용은 Systems Manager용 VPC 엔드포인트를 사용하여 EC2 인스턴스의 보안 개선 섹션을 참조하세요.

문제: 패치 누락

문제: `AWS-RunPatchbaseline`이 성공적으로 완료되었지만 일부 누락된 패치가 있습니다.

다음은 몇 가지 일반적인 원인과 해결 방법입니다.

원인 1: 기준이 유효하지 않습니다.

해결 방법 1: 이것이 원인인지 확인하려면 다음 절차를 따릅니다.

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택합니다.
3. [명령 기록(Command history)] 탭을 선택하고 기준을 확인할 명령을 선택합니다.
4. 패치가 누락된 관리형 노드를 선택합니다.
5. [1단계 - 출력(Step 1 - Output)]을 선택하고 `BaselineId` 값을 찾습니다.
6. 할당된 패치 기준 구성, 즉 운영 체제, 제품 이름, 분류 및 패치 기준의 심각도를 확인합니다.
7. Microsoft Update 카탈로그로 이동합니다.
8. Microsoft 기술 자료(KB) 문서 ID(예: KB3216916)를 검색합니다.
9. 제품(Product) 아래의 값이 관리형 노드의 값과 일치하는지 확인하고 해당 제목(Title)을 선택합니다. 새로운 [세부 정보 업데이트(Update Details)] 창이 열립니다.
10. [개요(Overview)] 탭의 [분류(classification)] 및 [MSRC 심각도(MSRC severity)]가 이전에 찾은 패치 기준 구성과 일치해야 합니다.

원인 2: 패치가 교체되었습니다.

해결 방법 2: 이것이 맞는지 확인하려면 다음 절차를 따릅니다.

1. Microsoft Update 카탈로그로 이동합니다.
2. Microsoft 기술 자료(KB) 문서 ID(예: KB3216916)를 검색합니다.
3. 제품(Product) 아래의 값이 관리형 노드의 값과 일치하는지 확인하고 해당 제목(Title)을 선택합니다. 새로운 [세부 정보 업데이트(Update Details)] 창이 열립니다.
4. [패키지 세부 정보(Package Details)] 탭으로 이동합니다. [이 업데이트는 다음 업데이트로 대체되었습니다.(This update has been replaced by the following updates:)] 헤더 아래에서 항목을 찾습니다.

원인 3: WSUS 및 Window 온라인 업데이트는 Microsoft에서 독립적인 릴리스 채널로 처리하기 때문에 동일한 패치의 KB 번호가 다를 수 있습니다.

해결 방법 3: 패치 적격성을 확인합니다. WSUS에서 패키지를 사용할 수 없는 경우 OS 빌드 14393.3115를 설치합니다. 모든 운영 체제 빌드에 패키지를 사용할 수 있는 경우 OS 빌드 18362.1256 및 18363.1256을 설치합니다.

AWS Support 문의

이 섹션이나 AWS re:Post의 Systems Manager 개발자 포럼에서 문제 해결 방법을 찾을 수 없고 Developer, Business 또는 Enterprise AWS Support 플랜이 있는 경우 AWS Support에서 기술 지원 사례를 생성할 수 있습니다.

AWS Support에 연락하기 전에 다음 항목을 수집합니다.

- SSM Agent 로그
- Run Command 명령 ID, 유지 관리 기간 ID 또는 Automation 실행 ID
- Windows Server 관리형 노드를 위해서는 다음 항목도 수집합니다.
  - 패치 설치 방법ㅂ의 [Windows] 탭에 설명된 `%PROGRAMDATA%\Amazon\PatchBaselineOperations\Logs`
  - Windows 업데이트 로그: Windows Server 2012 R2 이상의 경우 `%windir%/WindowsUpdate.log`를 사용합니다. Windows Server 2016 이상에서는 `%windir%/WindowsUpdate.log`를 사용하기 전에 먼저 PowerShell 명령 `Get-WindowsUpdateLog`를 실행합니다.
- Linux 관리형 노드를 위해서는 다음 항목도 수집합니다.
  - 디렉터리 `/var/lib/amazon/ssm/`*`instance-id`*`/document/orchestration/`*`Run-Command-execution-id`*`/awsrunShellScript/PatchLinux`의 내용

AWS Systems Manager Distributor

Distributor의 기능인 AWS Systems Manager를 사용하면 AWS Systems Manager 관리형 노드에 소프트웨어를 패키징하고 게시할 수 있습니다. 자체 소프트웨어를 패키징하고 게시하거나 Distributor을 사용해 AWS 제공 에이전트 소프트웨어 패키지를 찾거나 게시할 수 있습니다(예: AmazonCloudWatchAgent 또는 Trend Micro와 같은 서드 파티 패키지). 패키지 게시는 패키지 문서의 특정한 버전을 노드 ID, AWS 계정 ID, 태그, 또는 AWS 리전를 사용해 인식하는 관리형 노드에 보급합니다. Distributor를 시작하려면 Systems Manager 콘솔을 엽니다. 탐색 창에서 Distributor를 선택합니다.

Distributor에서 패키지를 생성하면, 다음 중 한 가지 방법으로 해당 패키지를 설치할 수 있습니다.

- AWS Systems Manager Run Command를 사용하여 한 번만
- AWS Systems Manager State Manager를 사용하여 일정에 따라

> **Important**
>
> 타사 판매자가 배포한 패키지는 AWS가 관리하지 않으며 패키지 공급업체가 게시합니다. 내부 보안 통제를 준수하기 위해 추가 실사를 수행하는 것이 좋습니다. 보안은 AWS과 사용자의 공동 책임입니다. 이것을 공동 책임 모델이라고 합니다. 자세한 내용은 공동 책임 모델을 참조하세요.

Distributor가 조직에 주는 이점은 무엇인가요?

Distributor에서 제공하는 이점은 다음과 같습니다.

- 한 패키지에 여러 플랫폼

  Distributor에서 패키지를 생성할 때, 시스템은 AWS Systems Manager 문서(SSM 문서)를 생성합니다. 이 문서에.zip 파일을 첨부할 수 있습니다. Distributor를 실행할 때, 시스템은 SSM 문서의 지침을 처리하고 지정된 대상의.zip 파일에 소프트웨어 패키지를 설치합니다. Distributor는 Windows, Ubuntu Server, Debian Server 및 Red Hat Enterprise Linux 등 여러 운영 체제를 지원합니다. 지원되는 플랫폼에 대한 자세한 내용은 지원되는 패키지 플랫폼 및 아키텍처 섹션을 참조하세요.

- 관리형 인스턴스 그룹 간에 패키지 액세스 제어

  Run Command 또는 State Manager를 사용해 패키지를 얻을 관리형 노드와 해당 패키지의 버전을 제어할 수 있습니다. Run Command와 State Manager는 AWS Systems Manager의 기능입니다. 관리형 노드는 인스턴스, 장치 ID, AWS 계정 번호, 태그 또는 AWS 리전별로 그룹화할 수 있습니다. State Manager 연결을 사용해 패키지의 여러 버전을 여러 인스턴스 그룹으로 전달할 수 있습니다.

- 포함되어 바로 사용 가능한 많은 AWS 에이전트

  Distributor에는 관리형 노드에 바로 배포할 수 있는 AWS 에이전트 패키지가 많이 포함되어 있습니다. Distributor `Packages` 목록 페이지에서 `Amazon`이 발행한 패키지를 찾습니다. 예를 들면 `AmazonCloudWatchAgent`나 `AWSPVDriver`와 같습니다.

- 배포 자동화

  환경을 최신 상태로 유지하려면 State Manager를 사용하여 노드를 처음 시작할 때 대상 관리형 노드에 자동으로 배포하도록 패키지를 예약합니다.

Distributor는 누가 사용해야 하나요?

- 새로 생성한 소프트웨어 패키지 또는 기존 소프트웨어 패키지(AWS에서 게시한 패키지 포함)를 여러 Systems Manager 관리형 노드에 한 번에 배포하려는 모든 AWS 고객.
- 소프트웨어 패키지를 생성하는 소프트웨어 개발자
- 최신 소프트웨어 패키지를 사용해 Systems Manager 관리형 노드를 최신 상태로 유지해야 하는 관리자.

Distributor에는 어떤 기능이 있나요?

- Windows 및 Linux 인스턴스 둘 다에 패키지 배포

  Distributor를 사용하여 Linux 및 Windows Server용 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 및 AWS IoT Greengrass 코어 디바이스 소프트웨어 패키지를 배포할 수 있습니다. 지원되는 인스턴스 운영 체제 유형의 목록은 the section called “지원되는 패키지 플랫폼 및 아키텍처” 섹션을 참조하세요.

  > **Note**
  >
  > Distributor는 macOS 운영 체제에서 지원됩니다.

- 한 번만 또는 자동화된 일정에 따라 패키지 배포

  패키지는 한 번만, 정기적으로 또는 기본 패키지 버전이 다른 버전으로 바뀔 때마다 배포하도록 선택할 수 있습니다.

- 패키지를 완전히 다시 설치하거나 인플레이스 업데이트 수행

  새 패키지 버전을 설치하려면 제공하는 업데이트 스크립트에 따라 현재 버전을 완전히 제거하고 새 버전을 설치하거나 새 구성 요소와 업데이트된 구성 요소로만 현재 버전을 업데이트할 수 있습니다. 재설치 동안에는 패키지 애플리케이션을 사용할 수 없지만 인플레이스 업데이트 중에는 계속 사용할 수 있습니다. 인플레이스 업데이트는 보안 모니터링 애플리케이션이나 애플리케이션 다운타임을 피해야 하는 기타 시나리오에 특히 유용합니다.

- 콘솔, CLI, PowerShell 및 SDK에서 Distributor 기능에 액세스

  선택한 Systems Manager 콘솔, AWS Command Line Interface(AWS CLI), AWS Tools for PowerShell 또는 AWS SDK를 사용하여 Distributor로 작업할 수 있습니다.

- IAM 액세스 제어

AWS Identity and Access Management(IAM) 정책을 사용해 패키지 또는 패키지 버전을 생성, 배포 또는 삭제할 수 있는 조직의 멤버를 제어할 수 있습니다. 예를 들어, 관리자에게 패키지를 배포할 수 있는 권한은 부여하지만 패키지를 변경하거나 새 패키지 버전을 생성할 수 있는 권한은 부여하지 않을 수 있습니다.

- 로깅 및 감사 기능 지원

  다른 AWS 서비스와(과)의 통합을 통해 AWS 계정에서 Distributor 사용자 작업을 감사하고 로그할 수 있습니다. 자세한 내용은 Distributor 활동 감사 및 로깅 단원을 참조하십시오.

패키지란 무엇입니까

패키지는 다음을 포함해 설치 가능한 소프트웨어 또는 자산 모음입니다.

- 대상 운영 체제 플랫폼별 소프트웨어 .zip 파일 각 .zip 파일에는 다음 스크립트가 포함되어 있어야 합니다.
  - install 및 uninstall 스크립트. Windows Server 기반 관리형 노드에는 PowerShell 스크립트(스크립트 이름이 `install.ps1` 및 `uninstall.ps1`)가 필요합니다. Linux 기반 관리형 노드에는 셸 스크립트(스크립트 이름: `install.sh` 및 `uninstall.sh`)가 필요합니다. AWS Systems Manager SSM Agent가 install 및 uninstall 스크립트의 명령을 읽고 수행합니다.
  - 실행 파일. 대상 관리형 노드에 패키지를 설치하려면 SSM Agent가 이 실행 파일을 찾아야 합니다.
- 패키지 콘텐츠를 설명하는 JSON 형식 매니페스트 파일. 매니페스트는 ZIP 파일에 포함되어 있지 않지만 패키지를 구성하는 ZIP 파일과 동일한 Amazon Simple Storage Service(Amazon S3) 버킷에 저장됩니다. 매니페스트는 패키지 버전을 식별하고, 패키지의 ZIP 파일을 대상 관리형 노드 속성(예: 운영 체제 버전 또는 아키텍처)에 매핑합니다. 매니페스트를 생성하는 방법은 2단계: JSON 패키지 매니페스트 생성 섹션을 참조하세요.

Distributor 콘솔에서 단순(Simple) 패키지 생성을 선택하면 Distributor가 소프트웨어 실행 파일 이름과 대상 플랫폼 및 아키텍처를 기반으로 설치 및 제거 스크립트, 파일 해시 및 JSON 패키지 매니페스트를 생성합니다.

지원되는 패키지 플랫폼 및 아키텍처

Distributor을 사용해 다음 Systems Manager 관리형 노드 플랫폼에 패키지를 게시할 수 있습니다. 버전 값은 대상 운영 체제 Amazon Machine Image(AMI)의 정확한 릴리스 버전과 일치해야 합니다. 이 버전 확인에 대한 자세한 내용은 2단계: JSON 패키지 매니페스트 생성의 4단계 참조하십시오.

> **Note**
>
> Systems Manager는 다음 AWS IoT Greengrass 코어 디바이스를 위한 운영 체제 중 일부를 지원하지 않습니다. 자세한 내용은 AWS IoT Greengrass Version 2 개발자 안내서의 AWS IoT Greengrass 코어 디바이스 설정 섹션을 참조하세요.

플랫폼	매니페스트 파일의 코드 값	아키텍처
Windows Server	`windows`	`x86_64` 또는 `386`
Debian Server	`debian`	`x86_64` 또는 `386`
Ubuntu Server	`ubuntu`	`x86_64` 또는 `386` `arm64`(Ubuntu Server 16 이상, A1 인스턴스 유형)
Red Hat Enterprise Linux (RHEL)	`redhat`	`x86_64` 또는 `386` `arm64`(RHEL 7.6 이상, A1 인스턴스 유형)
CentOS	`centos`	`x86_64` 또는 `386`
Amazon Linux 1, Amazon Linux 2, Amazon Linux 2023	`amazon`	`x86_64` 또는 `386` `arm64`(Amazon Linux 2 및 AL2023, A1 인스턴스 유형)
SUSE Linux Enterprise Server (SLES)	`suse`	`x86_64` 또는 `386`
openSUSE	`opensuse`	`x86_64` 또는 `386`
openSUSE Leap	`opensuseleap`	`x86_64` 또는 `386`
Oracle Linux	`oracle`	`x86_64`

주제

Distributor 설정

AWS Systems Manager의 기능인 Distributor를 사용해 소프트웨어 패키지를 생성, 관리 및 배포하기 전에 다음 단계를 수행합니다.

주제

1단계: Distributor 사전 조건 완료

AWS Systems Manager의 기능인 Distributor를 사용하기 전에 환경이 다음 요구 사항을 충족하는지 확인합니다.

Distributor 필수 조건

요구 사항	설명
SSM Agent	배포하려는 관리형 노드 또는 패키지를 제거하려는 노드에 AWS Systems Manager SSM Agent 버전 2.3.274.0 이상이 설치되어 있어야 합니다. SSM Agent를 설치 또는 업데이트하려면 SSM Agent 작업 섹션을 참조하세요.
AWS CLI	(옵션) Systems Manager 콘솔 대신 AWS Command Line Interface(AWS CLI)를 사용하여

요구 사항	설명
	패키지를 생성하고 관리하려면 로컬 컴퓨터에서 AWS CLI의 최신 릴리스를 설치합니다. CLI를 설치하거나 업그레이드하는 방법에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서의 AWS Command Line Interface 설치를 참조하세요.
AWS Tools for PowerShell	(옵션) Systems Manager 콘솔 대신 Tools for PowerShell을 사용하여 패키지를 생성하고 관리하려면 로컬 컴퓨터에 Tools for PowerShell의 최신 릴리스를 설치합니다. Tools for PowerShell을 설치하거나 업그레이드하는 방법에 대한 자세한 내용은 AWS Tools for Windows PowerShell 사용 설명서의 AWS Tools for Windows PowerShell 또는 AWS Tools for PowerShell Core 설정을 참조하세요.

> **Note**
>
> Systems Manager에서는 Distributor를 사용하여 Oracle Linux 관리형 노드에 패키지를 배포하는 작업을 지원하지 않습니다.

2단계: Distributor 권한을 사용하여 IAM 인스턴스 프로파일 확인 또는 생성

AWS Systems Manager는 기본적으로 인스턴스에서 작업을 수행할 권한이 없습니다. AWS Identity and Access Management(IAM) 인스턴스 프로파일을 사용하여 액세스 권한을 부여해야 합니다. 인스턴스 프로파일은 시작 시 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에 IAM 역할 정보를 전달하는 컨테이너입니다. 이러한 요구 사항은 AWS Systems Manager의 기능인 Distributor만이 아니라 모든 Systems Manager 기능에 대한 권한에 적용됩니다.

> **Note**
>
> 엣지 디바이스가 AWS IoT Greengrass 코어 소프트웨어와 SSM Agent를 실행하도록 구성할 때 Systems Manager가 사용해 작업을 수행하는 IAM 서비스 역할을 특정하세요. 인스턴스 프로파일을 사용하여 관리형 에지 디바이스를 구성할 필요가 없습니다.

Run Command와 State Manager 등의 다른 Systems Manager 기능을 이미 사용하고 있는 경우에는 Distributor에 대해 필요한 권한이 있는 인스턴스 프로파일이 이미 인스턴스에 연결되어 있습니다. Distributor 태스크를 수행할 수 있는 권한이 있는지 확인하는 가장 간단한 방법은 AmazonSSMManagedInstanceCore 정책을 인스턴스 프로파일에 연결하는 것입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성을 참조하세요.

3단계: 패키지에 대한 사용자 액세스 제어

AWS Identity and Access Management(IAM) 정책을 사용하면 패키지를 생성, 배포 및 관리할 수 있는 사람을 제어할 수 있습니다. 또한 관리형 노드에서 수행할 수 있는 Run Command 및 State Manager API 작업을 제어할 수도 있습니다. Distributor와 마찬가지로 Run Command와 State Manager도 모두 AWS Systems Manager의 기능입니다.

ARN 형식

사용자 정의 패키지는 문서 Amazon 리소스 이름(ARN)과 연결되어 있고 다음과 같은 형식을 갖습니다.

```
arn:aws:ssm:region:account-id:document/document-name
```

다음은 예입니다.

```
arn:aws:ssm:us-west-1:123456789012:document/ExampleDocumentName
```

각각 최종 사용자와 관리자를 위한 AWS 제공 기본 IAM 정책 페어를 사용하여 Distributor 작업에 대한 권한을 부여할 수 있습니다. 또는 권한 요구 사항에 적절한 사용자 정의 IAM 정책을 생성할 수 있습니다.

IAM 정책에서 변수 사용에 대한 자세한 내용은 IAM 정책 요소: 변수를 참조하세요.

정책을 생성하고 사용자 또는 그룹에 연결하는 방법에 대한 자세한 내용은 IAM 사용 설명서의 IAM 정책 생성 및 IAM 정책 추가 및 제거를 참조하세요.

4단계: Amazon S3 버킷 생성 또는 선택

AWS Systems Manager 콘솔에서 단순(Simple) 워크플로를 사용해 패키지를 생성할 경우 Distributor에서 소프트웨어를 업로드하던 기존 Amazon Simple Storage Service(Amazon S3) 버킷을 선택합니다. Distributor는 AWS Systems Manager의 기능입니다. 고급(Advanced) 워크플로에서는 시작하기 전에 소프트웨어 또는 자산의 ZIP 파일을 Amazon S3 버킷에 업로드해야 합니다. 콘솔에서 단순(Simple) 또는 고급(Advanced) 워크플로를 선택하든, API를 사용하든 상관없이 패키지 생성을 시작하려면 Amazon S3 버킷이 필요합니다. 패키지 생성 프로세스가 시작되면 Distributor가 설치할 수 있는 소프트웨어 및 자산을 이 버킷에서 내부 Systems Manager 스토어로 복사합니다. 자산이 내부 스토어로 복사되기 때문에 패키지 생성을 마치면 Amazon S3 버킷을 삭제하거나 다른 용도로 사용할 수 있습니다.

버킷 생성 방법에 대한 자세한 내용은 Amazon Simple Storage Service 시작 안내서의 버킷 생성을 참조하세요. AWS CLI 명령을 실행하여 버킷을 생성하는 방법에 대한 자세한 내용은 AWS CLI Command Reference의 mb를 참조하세요.

Distributor 작업

AWS Systems Manager 콘솔, AWS 명령줄 도구(AWS CLI 및 AWS Tools for PowerShell), AWS SDK를 사용하여 Distributor에서 패키지를 추가, 관리 또는 배포할 수 있습니다. Distributor는 AWS Systems Manager의 기능입니다. Distributor에 패키지를 추가하기 전에:

- 설치 가능한 자산을 생성해 압축합니다.
- (선택 사항) 패키지의 JSON 매니페스트 파일을 생성합니다. Distributor 콘솔에서 단순(Simple) 패키지 생성 프로세스를 사용하는 경우에는 필요하지 않습니다. 심플 패키지 생성 프로세스에서는 JSON 매니페스트 파일이 자동으로 생성되기 때문입니다.

  매니페스트 파일은 AWS Systems Manager 콘솔이나 텍스트 또는 JSON 편집기를 사용해 생성할 수 있습니다.
- 설치 가능한 자산 또는 소프트웨어를 저장할 Amazon Simple Storage Service(Amazon S3) 버킷을 준비합니다. 고급(Advanced) 패키지 생성 프로세스를 사용한다면 시작하기 전에 자산을 Amazon S3 버킷에 업로드합니다.

> **Note**
>
> 패키지 생성 프로세스에서 Distributor가 패키지 내용을 내부 Systems Manager 버킷에 저장하기 때문에 패키지 생성을 마친 후 이 버킷을 삭제하거나 다른 용도로 사용할 수 있습니다.

AWS에서 게시한 패키지는 이미 패키징되어 바로 배포할 수 있습니다. AWS에서 게시한 패키지를 관리형 노드에 배포하려면 패키지 설치 또는 업데이트 섹션을 참조하세요.

AWS 계정 간에 Distributor 패키지를 공유할 수 있습니다. AWS CLI 명령에서 다른 계정에서 공유한 패키지를 사용하는 경우 패키지 이름 대신 Amazon 리소스 이름(ARN) 패키지를 사용합니다.

주제

패키지 보기

설치 가능한 패키지를 보려면 AWS Systems Manager 콘솔이나 선호하는 AWS 명령줄 도구를 사용합니다. Distributor는 AWS Systems Manager의 기능입니다. Distributor에 액세스하려면 AWS Systems Manager 콘솔을 열고 왼쪽 탐색 창에서 Distributor를 선택합니다. 사용 가능한 모든 패키지를 볼 수 있습니다.

다음 섹션에서는 선호하는 명령줄 도구를 사용하여 Distributor 패키지를 보는 방법을 설명합니다.

패키지 보기(명령줄)

이 섹션에는 기본 명령줄 도구를 사용하여 제공된 명령을 사용하여 Distributor 패키지를 보는 방법에 대한 정보가 들어 있습니다.

Linux & macOS

Linux에서 AWS CLI를 사용하여 패키지를 보려면

- 공유 패키지를 제외한 모든 패키지를 보려면 다음 명령을 실행합니다.

```
aws ssm list-documents \
```

```
    --filters Key=DocumentType,Values=Package
```

- Amazon이 소유한 모든 패키지를 보려면 다음 명령을 실행합니다.

```
aws ssm list-documents \
    --filters Key=DocumentType,Values=Package Key=Owner,Values=Amazon
```

- 서드 파티에서 소유한 모든 패키지를 보려면 다음 명령을 실행합니다.

```
aws ssm list-documents \
    --filters Key=DocumentType,Values=Package Key=Owner,Values=ThirdParty
```

Windows

Windows에서 AWS CLI를 사용하여 패키지를 보려면

- 공유 패키지를 제외한 모든 패키지를 보려면 다음 명령을 실행합니다.

```
aws ssm list-documents ^
    --filters Key=DocumentType,Values=Package
```

- Amazon이 소유한 모든 패키지를 보려면 다음 명령을 실행합니다.

```
aws ssm list-documents ^
    --filters Key=DocumentType,Values=Package Key=Owner,Values=Amazon
```

- 서드 파티에서 소유한 모든 패키지를 보려면 다음 명령을 실행합니다.

```
aws ssm list-documents ^
    --filters Key=DocumentType,Values=Package Key=Owner,Values=ThirdParty
```

PowerShell

Tools for PowerShell을 사용하여 패키지를 보려면

- 공유 패키지를 제외한 모든 패키지를 보려면 다음 명령을 실행합니다.

```
$filter = New-Object Amazon.SimpleSystemsManagement.Model.DocumentKeyValuesFilter
$filter.Key = "DocumentType"
$filter.Values = "Package"
```

```
Get-SSMDocumentList `
    -Filters @($filter)
```

- Amazon이 소유한 모든 패키지를 보려면 다음 명령을 실행합니다.

```
$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)
```

- 서드 파티에서 소유한 모든 패키지를 보려면 다음 명령을 실행합니다.

```
$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)
```

패키지 생성

패키지를 생성하려면 설치 가능한 소프트웨어 또는 자산을 운영 체제 플랫폼마다 파일 1개씩 준비합니다. 파일이 1개 이상 있어야만 패키지를 생성할 수 있습니다.

여러 플랫폼에서 동일한 파일을 사용하는 경우도 있을 수 있지만 패키지에 연결한 모든 파일은 매니페스트의 `Files` 섹션에 나열되어 있어야 합니다. 콘솔에서 단순 워크플로를 사용해 패키지를 생성하는 경우에는 매니페스트가 자동으로 생성됩니다. 단일 문서에 연결할 수 있는 파일의 최대 개수는 20개입

니다. 각 파일의 최대 크기는 1GB입니다. 지원되는 플랫폼에 대한 자세한 내용은 지원되는 패키지 플랫폼 및 아키텍처 섹션을 참조하세요.

패키지를 생성할 때 시스템에서 새 SSM 문서를 생성합니다. 이 문서를 사용하면 관리형 노드에 패키지를 배포할 수 있습니다.

데모용인 예제 패키지 ExamplePackage.zip을 웹 사이트에서 다운로드할 수 있습니다. 예제 패키지에는 완료된 JSON 매니페스트와 PowerShell v7.0.0용 설치 프로그램이 포함된 .zip 파일 3개가 들어 있습니다. 설치 및 제거 스크립트에는 유효한 명령이 포함되어 있지 않습니다. 고급(Advanced) 워크플로에서는 소프트웨어 설치 파일과 스크립트를 .zip 파일로 압축해야 패키지를 생성할 수 있지만 단순(Simple) 워크플로에서는 설치 가능한 자산을 압축하지 않습니다.

주제

- 패키지 생성(단순)
- 패키지 생성(고급)

패키지 생성(단순)

이 섹션에서는 Distributor 콘솔을 사용해 단순(Simple) 패키지 생성 워크플로를 선택하여 Distributor에서 패키지를 생성하는 방법에 대해 알아봅니다. Distributor는 AWS Systems Manager의 기능입니다. 패키지를 생성하려면 설치 가능한 자산을 운영 체제 플랫폼마다 파일 1개씩 준비합니다. 파일이 1개 이상 있어야만 패키지를 생성할 수 있습니다. 단순(Simple) 패키지 생성 프로세스에서는 설치 및 제거 스크립트와 파일 해시, 그리고 JSON 형식의 매니페스트가 자동으로 생성됩니다. 단순(Simple) 워크플로우를 선택하면 설치 파일을 업로드하여 ZIP 파일로 압축하는 프로세스와 새로운 패키지를 비롯해 연결된 SSM 문서를 생성하는 프로세스가 처리됩니다. 지원되는 플랫폼에 대한 자세한 내용은 지원되는 패키지 플랫폼 및 아키텍처 섹션을 참조하세요.

심플 메서드를 사용하여 패키지를 만들면 Distributor에서 `install` 및 `uninstall` 스크립트가 생성됩니다. 그러나 인플레이스 업데이트를 위한 패키지를 생성하는 경우에는 업데이트 스크립트(Update script) 탭에서 자체 `update` 스크립트 내용을 제공해야 합니다. `update` 스크립트에 대한 입력 명령을 추가할 때 Distributor은 `install` 및 `uninstall` 스크립트와 함께 생성된 .zip 패키지에 이 스크립트를 포함합니다.

> **Note**
>
> `In-place` 업데이트 옵션을 사용하여 연결된 애플리케이션을 오프라인으로 전환하지 않고도 기존 패키지 설치에 새 파일이나 업데이트된 파일을 추가합니다.

패키지를 생성하려면(단순)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Distributor를 선택합니다.
3. Distributor 홈페이지에서 패키지 생성(Create package)과 단순(Simple)을 차례대로 선택합니다.
4. 패키지 생성(Create package) 페이지에서 패키지 이름을 입력합니다. 패키지 이름은 문자, 숫자, 마침표, 대시 및 밑줄을 포함할 수 있습니다. 이 이름은 모든 버전의 패키지 연결에 적용할 수 있을 정도로 충분히 일반적이어야 하지만 패키지의 목적을 식별할 수 있을 정도로 구체적이어야 합니다.
5. (선택 사항) 버전 이름(Version name)에 버전 이름을 입력합니다. 버전 이름은 최대 512자로 구성되며, 여기에 특수 문자가 포함될 수 없습니다.
6. 위치(Location)에서 버킷 이름 및 접두사를 사용하거나 버킷 URL을 사용하여 버킷을 선택합니다.
7. 소프트웨어 업로드(Upload software)에서 소프트웨어 추가(Add software)를 선택한 다음, `.rpm`, `.msi`, `.deb` 확장을 사용하여 설치 가능한 소프트웨어 파일을 검색합니다. 파일 이름에 공백이 있으면 업로드가 실패합니다. 소프트웨어 파일은 한 번에 1개 이상 업로드할 수 있습니다.
8. 대상 플랫폼(Target platform)에서 설치 파일마다 표시된 대상 운영 체제 플랫폼이 정확한지 확인합니다. 운영 체제가 잘못 표시되어 있으면 드롭다운 목록에서 정확한 운영 체제를 선택합니다.

   단순(Simple) 패키지 생성 워크플로의 경우, 설치 가능한 파일마다 한 번씩 업로드하기 때문에 Distributor에서 다수의 운영 체제에 단일 파일을 대상으로 지정하도록 명령하기 위한 추가 단계가 필요합니다. 예를 들어 이름이 `Logtool_v1.1.1.rpm`인 소프트웨어 설치 파일을 업로드하는 경우에는 단순(Simple) 워크플로우에서 기본값을 몇 가지 변경해야만 동일한 소프트웨어를 Amazon Linux 운영 체제와 Ubuntu 운영 체제에 업로드할 수 있습니다. 여러 플랫폼을 대상으로 지정할 때는 다음 중 하나를 수행합니다.

   - 시작하기 전에 고급 워크플로를 사용해 각 설치 파일을 zip 파일로 압축한 다음 설치 파일을 다수의 운영 체제 플랫폼 또는 버전에 업로드할 수 있도록 매니페스트를 직접 작성합니다. 자세한 내용은 패키지 생성(고급) 단원을 참조하십시오.
   - zip 파일이 다수의 운영 체제 플랫폼 또는 버전으로 업로드될 수 있도록 심플 워크플로에서 매니페스트 파일을 직접 편집합니다. 자세한 내용은 2단계: JSON 패키지 매니페스트 생성 단원에서 4단계 마지막을 참조하십시오.
9. 플랫폼 버전(Platform version)에서 표시된 운영 체제 플랫폼 버전이 **_any**, 와일드카드 뒤에 오는 주 릴리스 버전(7.*) 또는 소프트웨어를 적용하려는 정확한 운영 체제 릴리스 버전인지 확인합니다. 운영 체제 플랫폼 버전을 지정하는 방법에 대한 자세한 내용은 2단계: JSON 패키지 매니페스트 생성 단원에서 4단계를 참조하십시오.

10. 아키텍처(Architecture)의 드롭다운 목록에서 설치 파일마다 정확한 프로세서 아키텍처를 선택합니다. 지원되는 프로세서 아키텍처에 대한 자세한 내용은 지원되는 패키지 플랫폼 및 아키텍처 섹션을 참조하세요.
11. (선택 사항) 스크립트를 확장하고 설치 가능한 소프트웨어에 대해 Distributor에서 생성되는 스크립트를 검토합니다.
12. (선택 사항) 인플레이스 업데이트에 사용할 업데이트 스크립트를 제공하려면 스크립트를 확장하고 Update script(업데이트 스크립트) 탭을 선택한 다음, 업데이트 스크립트 명령을 입력합니다.

    Systems Manager는 사용자를 대신하여 업데이트 스크립트를 생성하지 않습니다.
13. 소프트웨어 설치 파일을 추가하려면 Add software(소프트웨어 추가)를 선택합니다. 아니면 다음 단계로 이동합니다.
14. (선택 사항) Manifest(매니페스트)를 확장하여 Distributor에서 설치 가능한 소프트웨어에 생성한 JSON 패키지 매니페스트를 살펴봅니다. 이번 절차를 시작한 이후 플랫폼 버전, 대상 플랫폼 등 소프트웨어에 대한 정보를 변경하였다면 Generate manifest(매니페스트 생성)을 선택하여 업데이트된 패키지 매니페스트를 확인합니다.

    8단계에서 설명한 것처럼 소프트웨어 설치 파일을 다수의 운영 체제에 업로드하는 경우에는 매니페스트를 수동으로 편집할 수 있습니다. 매니페스트 편집에 대한 자세한 내용은 2단계: JSON 패키지 매니페스트 생성 섹션을 참조하세요.
15. Create package(패키지 생성)를 선택합니다.

Distributor에서 소프트웨어 업로드 및 패키지 생성을 마칠 때까지 기다립니다. 각 설치 파일의 업로드 상태가 Distributor에 표시됩니다. 이 작업은 추가하는 패키지 수와 크기에 따라 몇 분까지 걸릴 수 있습니다. Distributor에서 새로운 패키지의 [패키지 세부 정보(Package details)] 페이지로 자동 리디렉션되지만 소프트웨어가 업로드된 후에도 이 페이지를 열도록 직접 선택할 수 있습니다. Distributor가 패키지 생성 프로세스를 마칠 때까지 [패키지 세부 정보(Package details)] 페이지에 패키지에 대한 모든 정보가 표시되지 않습니다. 업로드 및 패키지 생성 프로세스를 중지하려면 취소를 선택합니다.

Distributor에서 일부 소프트웨어 설치 파일을 업로드하지 못하는 경우에는 [업로드 실패(Upload failed)] 메시지가 표시됩니다. 업로드를 다시 시도하려면 Retry upload(업로드 재시도)를 선택합니다. 패키지 생성 실패에 따른 문제 해결 방법에 대한 자세한 내용은 AWS Systems ManagerDistributor 문제 해결 섹션을 참조하세요.

패키지 생성(고급)

이 섹션에서는 고급 사용자가 설치 및 제거 스크립트와 JSON 매니페스트 파일로 설치 가능한 자산을 압축하여 Amazon S3 버킷으로 업로드한 후 Distributor에서 패키지를 생성할 수 있는 방법에 대해서 알아보겠습니다.

패키지를 생성하려면 설치 가능한 자산의 .zip 파일을 운영 체제 플랫폼당 하나의 .zip 파일로 준비합니다. 패키지를 생성하려면 하나 이상의 .zip 파일이 필요합니다. 다음으로, JSON 매니페스트를 생성합니다. 매니페스트에는 패키지 코드 파일에 대한 포인터가 포함되어 있습니다. 필요한 코드 파일을 폴더 또는 디렉터리에 추가했고 매니페스트가 올바른 값으로 채워진 경우 S3 버킷에 패키지를 업로드합니다.

예제 패키지 ExamplePackage.zip은 Amazon 웹사이트에서 다운로드할 수 있습니다. 예제 패키지에는 완료된 JSON 매니페스트와 .zip 파일 3개가 들어 있습니다.

주제

- 1단계: ZIP 파일 생성
- 2단계: JSON 패키지 매니페스트 생성
- 3단계: Amazon S3 버킷에 패키지 및 매니페스트 업로드
- 4단계: Distributor에 패키지 추가

1단계: ZIP 파일 생성

소프트웨어 또는 설치 가능한 자산의 .zip 파일 하나 이상이 패키지의 바탕을 이룹니다. 여러 운영 체제에 하나의 .zip 파일을 설치할 수 없는 경우 패키지에는 지원하려는 운영 체제당 .zip 파일이 하나씩 포함됩니다. 예를 들어 Red Hat Enterprise Linux 및 Amazon Linux 인스턴스는 일반적으로 동일한 .RPM 실행 파일을 실행할 수 있으므로 두 운영 체제를 지원하려면 이 패키지에 .zip 파일을 하나만 연결해야 합니다.

필수 파일

각 .zip 파일에 다음 항목이 필요합니다.

- install 및 uninstall 스크립트. Windows Server 기반 관리형 노드에는 PowerShell 스크립트(스크립트 이름이 `install.ps1` 및 `uninstall.ps1`)가 필요합니다. Linux 기반 관리형 노드에는 셸 스크립트(스크립트 이름: `install.sh` 및 `uninstall.sh`)가 필요합니다. SSM Agent가 install 및 uninstall 스크립트의 명령을 실행합니다.

예를 들어, 설치 스크립트는 설치 프로그램(예: .rpm 또는 .msi)을 실행할 수 있고, 파일을 복사하거나 구성 설정을 지정할 수 있습니다.

- 실행 파일, 설치 프로그램 패키지(rpm, .deb, .msi 등), 기타 스크립트 또는 구성 파일 등

옵션 파일

각 .zip 파일에서 다음 항목은 선택 사항입니다.

- update 스크립트입니다. 업데이트 스크립트를 제공하면 `In-place update` 옵션을 사용하여 패키지를 설치할 수 있습니다. 기존 패키지 설치에 새 파일이나 업데이트된 파일을 추가하려는 경우 `In-place update` 옵션은 업데이트를 수행하는 동안 패키지 애플리케이션을 오프라인으로 전환하지 않습니다. Windows Server 기반 관리형 노드에는 PowerShell 스크립트(`update.ps1`라는 스크립트)가 필요합니다. Linux 기반 관리형 노드에는 셸 스크립트(`update.sh`라는 이름의 스크립트)가 필요합니다. SSM Agent은 update 스크립트의 명령을 실행합니다.

패키지 설치 또는 업데이트에 대한 자세한 내용은 패키지 설치 또는 업데이트 섹션을 참조하세요.

샘플 install 및 uninstall 스크립트를 포함한 .zip 파일의 예를 살펴보려면 예제 패키지인 ExamplePackage.zip을 다운로드하십시오.

2단계: JSON 패키지 매니페스트 생성

설치 파일을 준비하여 ZIP 파일로 압축하고 JSON 매니페스트를 생성합니다. 다음은 템플릿입니다. 이 단원의 절차에는 매니페스트 템플릿의 일부에 대한 설명이 나와 있습니다. JSON 편집기를 사용해 매니페스트를 별도의 파일로 생성할 수 있습니다. 또한 패키지를 생성할 때 AWS Systems Manager 콘솔에서 매니페스트를 작성할 수 있습니다.

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

JSON 패키지 매니페스트를 생성하려면

1. 매니페스트에 스키마 버전을 추가합니다. 이 릴리스에서 스키마 버전은 항상 `2.0`입니다.

   ```
   { "schemaVersion": "2.0",
   ```

2. 매니페스트에 사용자 정의 패키지 버전을 추가합니다. Distributor에 패키지를 추가할 때 지정하는 [버전 이름(Version name)]의 값이기도 합니다. 이 값은 패키지를 추가할 때 Distributor에서 생성한 AWS Systems Manager 문서의 일부가 됩니다. 또한 최신 패키지 이외의 패키지 버전을 설치하려면 `AWS-ConfigureAWSPackage` 문서에 입력값으로 이 값을 제공합니다. `version` 값에는 문자, 숫자, 밑줄, 하이픈 및 마침표를 사용할 수 있으며 길이는 최대 128자입니다. 배포 시 정확한 패키지 버전을 쉽게 지정할 수 있으려면 사람이 읽을 수 있는 패키지 버전을 사용하는 것이 좋습니다. 다음은 예입니다.

```
"version": "1.0.1",
```

3. (선택 사항) 게시자 이름을 추가합니다. 다음은 예입니다.

```
"publisher": "MyOrganization",
```

4. 패키지를 추가합니다. "packages" 단원에는 패키지의 .zpi 파일에서 지원하는 플랫폼, 릴리스 버전 및 아키텍처에 대한 설명이 들어 있습니다. 자세한 내용은 지원되는 패키지 플랫폼 및 아키텍처 단원을 참조하십시오.

   *platform-version*은 와일드카드 값인 _any일 수 있습니다. 이 값을 사용하면 .zip 파일이 모든 플랫폼 릴리스를 지원함을 나타냅니다. 모든 부 버전이 지원되도록 주 릴리스 버전 뒤에 와일드카드를 지정할 수도 있습니다(예: 7.*). 특정 운영 체제 버전에 대한 *platform-version* 값을 지정하도록 선택한 경우 대상 운영 체제 AMI의 정확한 릴리스 버전과 일치하는지 확인합니다. 다음은 운영 체제의 올바른 값을 얻을 수 있도록 제안되는 리소스입니다.

   - Windows Server 기반 관리형 노드에서 릴리스 버전은 WMI(Windows Management Instrumentation) 데이터로 사용할 수 있습니다. 다음 명령 프롬프트에서 명령을 실행해 버전 정보를 얻은 다음 version에 대한 결과를 구문 분석할 수 있습니다. 이 명령은 Windows Server Nano의 버전은 표시하지 않습니다. Windows Server Nano의 버전 값은 nano입니다.

     ```
     wmic OS get /format:list
     ```

   - Linux 기반 관리형 노드에서는 운영 체제 릴리스를 처음 스캔하여 버전을 확인할 수 있습니다(다음 명령). VERSION_ID의 값을 찾아보십시오.

     ```
     cat /etc/os-release
     ```

     필요한 값을 반환하지 않은 경우 다음 명령을 실행해 /etc/lsb-release 파일에서 LSB 릴리스 정보를 가져오고 DISTRIB_RELEASE의 값을 찾아봅니다.

     ```
     lsb_release -a
     ```

     이러한 방법이 실패하면 일반적으로 배포를 기준으로 릴리스를 확인할 수 있습니다. 예를 들어, Debian Server에서는 /etc/debian_version 파일을, Red Hat Enterprise Linux에서는 /etc/redhat-release 파일을 스캔할 수 있습니다.

```
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"
        }
      }
    }
  }
```

다음은 예입니다. 이 예제에서 운영 체제 플랫폼은 amazon이고, 지원되는 릴리스 버전은 2016.09이고, 아키텍처는 x86_64이고, 이 플랫폼을 지원하는 .zip 파일은 test.zip입니다.

```
{
    "amazon": {
        "2016.09": {
            "x86_64": {
                "file": "test.zip"
            }
        }
    }
},
```

패키지가 상위 요소의 모든 버전을 지원함을 나타내기 위해 와일드카드 값 _any를 추가할 수 있습니다. 예를 들어 패키지가 Amazon Linux의 모든 릴리스 버전에서 지원된다고 나타내려면 패키지 구문이 다음과 비슷해야 합니다. 버전 또는 아키텍처 수준에서_any 와일드카드를 사용하여 플랫폼의 모든 버전, 버전의 모든 아키텍처 또는 플랫폼의 모든 버전 및 아키텍처를 지원할 수 있습니다.

```
{
    "amazon": {
        "_any": {
            "x86_64": {
                "file": "test.zip"
            }
        }
    }
},
```

다음 예에서는 _any를 추가해 첫 번째 패키지 data1.zip이 Amazon Linux 2016.09의 모든 아키텍처에 대해 지원됨을 보여줍니다. 두 번째 패키지 data2.zip은 Amazon Linux의 모든 릴리스에 대해 지원되지만 아키텍처가 x86_64인 관리형 노드만 지원합니다. 2016.09 및 _any 버전은 둘 다 amazon 아래에 있는 항목입니다. 여기에는 지원되는 버전, 아키텍처 및 연결된 .zip 파일이 아니라 하나의 플랫폼(Amazon Linux)이 있습니다.

```
{
    "amazon": {
        "2016.09": {
            "_any": {
                "file": "data1.zip"
            }
        },
        "_any": {
            "x86_64": {
                "file": "data2.zip"
            }
        }
    }
}
```

.zip 파일이 플랫폼을 두 개 이상 지원하는 경우에는 매니페스트의 "packages" 섹션에서 .zip 파일을 두 번 이상 참조할 수 있습니다. 예를 들어 Red Hat Enterprise Linux 7.x 버전과 Amazon

Linux를 둘 다 지원하는 .zip 파일이 있는 경우 다음 예제에 표시된 것처럼 "packages" 섹션에는 동일한 .zip 파일을 가리키는 항목이 2개 있습니다.

```
{
    "amazon": {
        "2018.03": {
            "x86_64": {
                "file": "test.zip"
            }
        }
    },
    "redhat": {
        "7.*": {
            "x86_64": {
                "file": "test.zip"
            }
        }
    }
},
```

5. 4단계에서 이 패키지의 일부인 .zip 파일 목록을 추가합니다. 각 파일 항목에는 파일 이름과 sha256 해시 값 체크섬이 필요합니다. 설치 실패를 방지하기 위해 매니페스트의 체크섬 값은 압축된 자산에 sha256 해시 값과 일치해야 합니다.

   설치 가능 항목에서 정확한 체크섬을 얻기 위해 다음 명령을 실행할 수 있습니다. Linux에서 `shasum -a 256 file-name.zip` 또는 `openssl dgst -sha256 file-name.zip`을 실행합니다. Windows의 경우 PowerShell에서 `Get-FileHash -Path path-to-.zip-file` cmdlet을 실행합니다.

   매니페스트의 "files" 섹션에는 패키지의 각 .zip 파일에 대한 참조가 하나씩 들어 있습니다.

```
"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. 패키지 정보를 추가한 후에는 매니페스트 파일을 저장한 다음 닫습니다.

다음은 완성된 매니페스트의 예입니다. 이 예제에서는 플랫폼을 두 개 이상 지원하지만 "files" 섹션에서 한 번만 참조되는 .zip 파일인 NewPackage_LINUX.zip이 있습니다.

```
{
    "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"
            }
        }
```

```
    }
}
```

패키지 예제

예제 패키지 ExamplePackage.zip은 Amazon 웹사이트에서 다운로드할 수 있습니다. 예제 패키지에는 완료된 JSON 매니페스트와 .zip 파일 3개가 들어 있습니다.

3단계: Amazon S3 버킷에 패키지 및 매니페스트 업로드

모든 .zip 파일을 폴더 또는 디렉터리로 복사하거나 이동하여 패키지를 준비합니다. 유효한 패키지에는 2단계: JSON 패키지 매니페스트 생성에서 생성한 매니페스트 및 매니페스트 파일 목록에 지정된 모든 .zip 파일이 필요합니다.

Amazon S3에 패키지 및 매니페스트를 업로드하려면

1. 매니페스트에서 지정한 모든 .zip 아카이브 파일을 폴더 또는 디렉터리로 복사하거나 이동합니다. .zip 아카이브 파일 및 매니페스트 파일을 이동하려는 폴더 또는 디렉터리를 압축하지 않습니다.
2. 버킷을 생성하거나 기존 버킷을 선택합니다. 자세한 내용은 Amazon Simple Storage Service 시작 안내서의 버킷 생성을 참조하세요. AWS CLI 명령을 실행하여 버킷을 생성하는 방법에 대한 자세한 내용은 AWS CLI Command Reference의 mb를 참조하세요.
3. 버킷에 폴더나 디렉터리를 업로드합니다. 자세한 내용은 Amazon Simple Storage Service 시작 안내서의 버킷에 객체 추가를 참조하세요. JSON 매니페스트를 AWS Systems Manager 콘솔에 붙여 넣으려는 경우 매니페스트를 업로드하지 않습니다. AWS CLI 명령을 실행하여 파일을 업로드하는 방법에 대한 자세한 내용은 AWS CLI Command Reference의 mv를 참조하세요.
4. 버킷의 홈 페이지에서 업로드한 폴더나 디렉터리를 선택합니다. 파일을 버킷의 하위 폴더로 업로드하였다면 해당하는 하위 폴더(접두사로도 알 수 있음)를 기록해야 합니다. 패키지를 Distributor에 추가할 때 접두사가 필요하기 때문입니다.

4단계: Distributor에 패키지 추가

AWS Systems Manager 콘솔, AWS 명령줄 도구(AWS CLI 및 AWS Tools for PowerShell) 또는 AWS SDK를 사용하여 Distributor에 새 패키지를 추가할 수 있습니다. 패키지를 추가할 때 새 SSM 문서를 추가합니다. 이 문서를 사용하면 관리형 노드에 패키지를 배포할 수 있습니다.

주제

- 패키지 추가(콘솔)

- 패키지 추가(AWS CLI)

패키지 추가(콘솔)

AWS Systems Manager 콘솔을 사용하여 패키지를 만들 수 있습니다. 3단계: Amazon S3 버킷에 패키지 및 매니페스트 업로드에서 패키지를 업로드한 버킷 이름을 준비합니다.

Distributor에 패키지를 추가하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Distributor를 선택합니다.
3. Distributor 홈페이지에서 Create package(패키지 생성)과 고급을 차례대로 선택합니다.
4. 패키지 생성(Create package) 페이지에서 패키지 이름을 입력합니다. 패키지 이름은 문자, 숫자, 마침표, 대시 및 밑줄을 포함할 수 있습니다. 이 이름은 모든 버전의 패키지 연결에 적용할 수 있을 정도로 충분히 일반적이어야 하지만 패키지의 목적을 식별할 수 있을 정도로 구체적이어야 합니다.
5. Version name(버전 이름)에 매니페스트 파일 내 `version` 항목의 정확한 값을 입력합니다.
6. [S3 버킷 이름(S3 bucket name)]에서 the section called “3단계: Amazon S3 버킷에 패키지 및 매니페스트 업로드”에서 .zip 파일과 매니페스트를 업로드한 버킷의 이름을 선택합니다.
7. S3 키 접두사에 .zip 파일과 매니페스트가 저장되는 버킷의 하위 폴더를 입력합니다.
8. .zip 파일이 저장된 Amazon S3 버킷에 업로드한 매니페스트를 사용하려면 [매니페스트(Manifest)]에서 [패키지에서 추출(Extract from package)]을 선택합니다.

   (옵션) .zip 파일이 저장된 S3 버킷에 JSON 매니페스트를 업로드하지 않았다면 [새 매니페스트(New manifest)]를 선택합니다. JSON 편집기 필드에서 전체 매니페스트를 작성하거나 붙여넣을 수 있습니다. JSON 매니페스트를 생성하는 자세한 방법은 2단계: JSON 패키지 매니페스트 생성 섹션을 참조하세요.
9. 매니페스트 작성을 마치면 [패키지 생성(Create package)]을 선택합니다.
10. Distributor가 .zip 파일과 매니페스트에서 패키지를 생성할 때까지 기다립니다. 이 작업은 추가하는 패키지 수와 크기에 따라 몇 분까지 걸릴 수 있습니다. Distributor에서 새로운 패키지의 Package details(패키지 세부 정보) 페이지로 자동 리디렉션되지만 소프트웨어가 업로드된 후에도 이 페이지를 열도록 직접 선택할 수 있습니다. Distributor가 패키지 생성 프로세스를 마칠 때까지 [패키지 세부 정보(Package details)] 페이지에 패키지에 대한 모든 정보가 표시되지 않습니다. 업로드 및 패키지 생성 프로세스를 중지하려면 취소를 선택합니다.

패키지 추가(AWS CLI)

AWS CLI을 사용하여 패키지를 생성할 수 있습니다. 3단계: Amazon S3 버킷에 패키지 및 매니페스트 업로드에서 패키지를 업로드한 버킷에서 해당 URL을 사용할 준비가 되었습니다.

Amazon S3에 패키지 추가(AWS CLI)

1. AWS CLI를 사용하여 패키지를 생성하려면 다음 명령을 실행합니다. *package-name*을 해당 패키지 이름으로 바꾸고 *path-to-manifest-file*은 JSON 매니페스트 파일의 파일 경로로 바꾸십시오. DOC-EXAMPLE-BUCKET은 전체 패키지가 저장되는 Amazon S3 버킷의 URL입니다. Distributor에서 create-document 명령을 실행할 때 `--document-type`에 대해 `Package` 값을 지정합니다.

   Amazon S3 버킷에 매니페스트 파일을 추가하지 않은 경우 `--content` 파라미터 값은 JSON 매니페스트 파일에 대한 파일 경로입니다.

   ```
   aws ssm create-document \
       --name "package-name" \
       --content file://path-to-manifest-file \
       --attachments Key="SourceUrl",Values="DOC-EXAMPLE-BUCKET" \
       --version-name version-value-from-manifest \
       --document-type Package
   ```

   다음은 예입니다.

   ```
   aws ssm create-document \
       --name "ExamplePackage" \
       --content file://path-to-manifest-file \
       --attachments Key="SourceUrl",Values="https://s3.amazonaws.com/DOC-EXAMPLE-
   BUCKET/ExamplePackage" \
       --version-name 1.0.1 \
       --document-type Package
   ```

2. 패키지가 추가되었는지 확인하고 다음 명령을 실행하되 *package-name*은 패키지의 이름으로 바꿔 패키지 매니페스트를 표시합니다. (패키지의 버전과 같지 않은) 문서의 특정 버전을 확인하려면 `--document-version` 파라미터를 추가할 수 있습니다.

   ```
   aws ssm get-document \
       --name "package-name"
   ```

create-document 명령과 함께 사용할 수 있는 다른 옵션에 대한 자세한 내용은 AWS CLI Command Reference의 AWS Systems Manager 섹션에 있는 create-document를 참조하세요. get-document 명령과 함께 사용할 수 있는 다른 옵션에 대한 자세한 내용은 get-document를 참조하십시오.

패키지 권한 편집(콘솔)

AWS Systems Manager의 기능인 Distributor에 패키지를 추가한 후 Systems Manager 콘솔에서 패키지의 권한을 편집할 수 있습니다. 패키지의 권한에 다른 AWS 계정를 추가할 수 있습니다. 동일한 AWS 리전에서만 다른 계정과 패키지를 공유할 수 있습니다. 크로스 리전 공유는 지원되지 않습니다. 기본적으로 패키지는 [프라이빗(Private)]으로 설정되어 있습니다. 즉, 패키지 생성자의 AWS 계정에 대한 액세스 권한을 가진 사용자만 패키지 정보를 보고 패키지를 업데이트 또는 삭제할 수 있습니다. 프라이빗 권한을 허용 가능한 경우 이 절차를 건너뛸 수 있습니다.

> **Note**
>
> 20개 이하의 계정과 공유되는 패키지의 권한을 업데이트할 수 있습니다.

패키지 권한을 편집하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Distributor를 선택합니다.
3. 패키지 페이지에서 권한을 편집하려는 패키지를 선택합니다.
4. Package details(패키지 세부 정보) 탭에서 권한 편집을 선택해 권한을 변경합니다.
5. Edit permissions(권한 편집)에서 Shared with specific accounts(특정 계정과 공유됨)를 선택합니다.
6. [특정 계정과 공유됨(Shared with specific accounts)]에서 AWS 계정 번호를 한 번에 하나씩 추가합니다. 작업을 마쳤으면 저장을 선택합니다.

패키지 태그 편집(콘솔)

AWS Systems Manager의 기능인 Distributor에 패키지를 추가한 후에는 Systems Manager 콘솔에서 해당 패키지의 태그를 편집할 수 있습니다. 이러한 태그는 패키지에 적용되며, 패키지를 배포하려는 관리형 노드의 태그에는 연결되지 않습니다. 태그는 대소문자를 구분하는 키와 값 페어로, 조직과 관련된 기준으로 패키지를 그룹화 및 필터링하는 데 유용합니다. 태그를 추가하지 않으려는 경우 패키지를 설치하거나 새 버전을 추가할 준비가 된 것입니다.

패키지 태그를 편집하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Distributor를 선택합니다.
3. 패키지 페이지에서 태그를 편집할 패키지를 선택합니다.
4. 패키지 세부 정보 탭의 태그에서 편집을 선택합니다.
5. 태그 추가에서 태그 키나, 태그 키 및 값 페어를 입력한 다음 추가를 선택합니다. 태그를 더 추가하고 싶은 경우 이 단계를 반복합니다. 태그를 삭제하려면 창 아래에서 태그에 표시된 X를 선택합니다.
6. 패키지에 태그 추가가 완료되면 [저장(Save)]을 선택합니다.

Distributor에 패키지 버전 추가

패키지 버전을 추가하려면 패키지를 생성한 후 Distributor를 사용해 이전 버전에 이미 존재하는 AWS Systems Manager(SSM) 문서에 항목을 추가하여 패키지 버전을 추가합니다. Distributor는 AWS Systems Manager의 기능입니다. 시간을 절약하기 위해 패키지의 이전 버전에 대한 매니페스트를 업데이트하고 매니페스트에서 `version` 항목의 값을 변경한 다음(예: `Test_1.0`에서 `Test_2.0`으로 변경) 새 버전의 매니페스트로 저장합니다. Distributor 콘솔의 간단한 [버전 추가(Add version)] 워크플로가 매니페스트 파일을 업데이트합니다.

새 패키지 버전은 다음 작업을 수행할 수 있습니다.

- 현재 버전에 연결된 설치 파일을 하나 이상 바꿉니다.
- 추가 플랫폼을 지원하기 위해 새로운 설치 파일을 추가합니다.
- 특정 플랫폼에 대한 지원을 중단하기 위해 파일을 삭제합니다.

새 버전에서 동일한 Amazon Simple Storage Service(Amazon S3) 버킷을 사용할 수 있지만 맨 끝에 표시되는 URL의 파일 이름은 달라야 합니다. Systems Manager 콘솔이나 AWS Command Line Interface(AWS CLI)를 사용하여 새 버전을 추가할 수 있습니다. 기존에 Amazon S3 버킷에 저장되어 있던 설치 파일과 동일한 이름으로 설치 파일을 업로드할 경우 기존 파일을 덮어쓰게 됩니다. 설치 파일은 이전 버전에서 새로운 버전으로 복사할 수 없습니다. 따라서 새로운 버전에 추가하려면 이전 버전에서 설치 파일을 업로드해야 합니다. Distributor가 새로운 패키지 버전 생성을 마치면 Amazon S3 버킷을 삭제하거나 다른 용도로 사용할 수 있습니다. Distributor가 버전 관리 프로세스의 일환으로 소프트웨어를 내부 Systems Manager 버킷에 복사하기 때문입니다.

> **Note**
>
> 각 패키지는 최대 25개 버전으로 유지됩니다. 더 이상 필요하지 않은 버전을 삭제할 수 있습니다.

주제

- 패키지 버전 추가(콘솔)
- 패키지 버전 추가(AWS CLI)

패키지 버전 추가(콘솔)

이러한 단계를 수행하기 전에 패키지 생성의 지침에 따라 해당 버전에 대한 새 패키지를 생성합니다. 그런 다음 Systems Manager 콘솔을 사용하여 Distributor에 새 패키지 버전을 추가합니다.

패키지 버전 추가(심플)

Simple(단순) 워크플로를 사용해 패키지 버전을 추가할 때는 업데이트된 설치 가능한 파일을 준비하거나, 더욱 많은 플랫폼과 아키텍처를 지원할 수 있는 설치 파일을 추가합니다. 그런 다음 Distributor를 사용해 새롭거나 업데이트된 설치 파일을 업로드하고 패키지 버전을 추가합니다. Distributor 콘솔의 간단한 [버전 추가(Add version)] 워크플로가 매니페스트 파일과 관련 SSM 문서를 업데이트합니다.

패키지 버전을 추가하려면(단순)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Distributor를 선택합니다.
3. Distributor 홈 페이지에서 다른 버전을 추가하려는 패키지를 선택합니다.
4. Add version(버전 추가) 페이지에서 Simple(단순)을 선택합니다.
5. Version name(버전 이름)에 버전 이름을 입력합니다. 새로운 버전 이름은 이전 버전 이름과 달라야 합니다. 버전 이름은 최대 512자로 구성되며, 여기에 특수 문자가 포함될 수 없습니다.
6. S3 버킷 이름의 목록에서 기존 S3 버킷을 선택합니다. 이전 버전에서 설치 파일을 저장했던 버킷을 그대로 선택할 수도 있지만 이때는 버킷에 저장된 기존 설치 파일을 덮어쓰지 않도록 설치 파일 이름이 달라야 합니다.
7. S3 키 접두사에 설치 가능한 자산이 저장되는 버킷의 하위 폴더를 입력합니다.
8. [소프트웨어 업로드(Upload software)]에서 새로운 버전에 추가할 소프트웨어 설치 파일을 찾습니다. 기존 버전의 설치 파일은 새로운 버전으로 자동 복사되지 않습니다. 따라서 새로운 버전에 동

일한 설치 파일을 추가하려면 이전 패키지 버전에서 설치 파일을 업로드해야 합니다. 소프트웨어 파일은 한 번에 1개 이상 업로드할 수 있습니다.

9. 대상 플랫폼(Target platform)에서 설치 파일마다 표시된 대상 운영 체제 플랫폼이 정확한지 확인합니다. 운영 체제가 잘못 표시되어 있으면 드롭다운 목록에서 정확한 운영 체제를 선택합니다.

   Simple(단순) 버전 관리 워크플로우에서는 설치 파일마다 한 번씩 업로드하기 때문에 다수의 운영 체제를 대상으로 단일 파일을 업로드해야 한다면 추가 단계가 필요합니다. 예를 들어 이름이 `Logtool_v1.1.1.rpm`인 소프트웨어 설치 파일을 업로드하는 경우에는 Simple(단순) 워크플로우에서 기본값을 몇 가지 변경해야만 Distributor에서 동일한 소프트웨어를 Amazon Linux 운영 체제와 Ubuntu 운영 체제에 업로드할 수 있습니다. 이러한 제약은 다음 중 한 가지를 사용해 해결할 수 있습니다.

   - 시작하기 전에 고급 버전 관리 워크플로우를 사용해 각 설치 파일을 .zip 파일로 압축한 다음 설치 파일을 다수의 운영 체제 플랫폼 또는 버전에 업로드할 수 있도록 매니페스트를 직접 작성합니다. 자세한 내용은 패키지 버전 추가(고급) 단원을 참조하십시오.
   - zip 파일이 다수의 운영 체제 플랫폼 또는 버전으로 업로드될 수 있도록 심플 워크플로에서 매니페스트 파일을 직접 편집합니다. 자세한 내용은 2단계: JSON 패키지 매니페스트 생성 단원에서 4단계 마지막을 참조하십시오.

10. [플랫폼 버전(Platform version)]에서 표시된 운영 체제 플랫폼 버전이 **_any**, 와일드카드 뒤에 오는 주 릴리스 버전(7.*) 또는 소프트웨어를 적용하려는 정확한 운영 체제 릴리스 버전인지 확인합니다. 플랫폼 버전을 지정하는 방법에 대한 자세한 내용은 2단계: JSON 패키지 매니페스트 생성 단원에서 4단계를 참조하십시오.

11. 아키텍처의 드롭다운 목록에서 설치 파일마다 정확한 프로세서 아키텍처를 선택합니다. 지원되는 아키텍처에 대한 자세한 내용은 지원되는 패키지 플랫폼 및 아키텍처 섹션을 참조하세요.

12. (선택 사항) 스크립트를 확장하여 Distributor에서 설치 가능한 소프트웨어에 대해 생성된 설치 및 제거 스크립트를 검토합니다.

13. 소프트웨어 설치 파일을 새로운 버전에 추가하려면 Add software(소프트웨어 추가)를 선택합니다. 아니면 다음 단계로 이동합니다.

14. (선택 사항) Manifest(매니페스트)를 확장하여 Distributor에서 설치 가능한 소프트웨어에 생성한 JSON 패키지 매니페스트를 살펴봅니다. 이번 절차를 시작한 이후 플랫폼 버전, 대상 플랫폼 등 설치 가능한 소프트웨어에 대한 정보를 변경하였다면 Generate manifest(매니페스트 생성)을 선택하여 업데이트된 패키지 매니페스트를 확인합니다.

    9단계에서 설명한 것처럼 소프트웨어 설치 파일을 다수의 운영 체제에 업로드하는 경우에는 매니페스트를 수동으로 편집할 수 있습니다. 매니페스트 편집에 대한 자세한 내용은 2단계: JSON 패키지 매니페스트 생성 섹션을 참조하세요.

15. 소프트웨어 추가를 비롯해 대상 플랫폼, 버전 및 아키텍처 데이터에 대한 검토까지 마쳤다면 이제 Add version(버전 추가)를 선택합니다.
16. Distributor에서 소프트웨어 업로드 및 새로운 패키지 버전 생성을 마칠 때까지 기다립니다. 각 설치 파일의 업로드 상태가 Distributor에 표시됩니다. 이 작업은 추가하는 패키지 수와 크기에 따라 몇 분까지 걸릴 수 있습니다. Distributor에서 패키지의 Package details(패키지 세부 정보) 페이지로 자동 리디렉션되지만 소프트웨어가 업로드된 후에도 이 페이지를 열도록 직접 선택할 수 있습니다. Distributor가 패키지 새 패키지 버전 생성을 마칠 때까지 [Package details(패키지 세부 정보)] 페이지에 패키지에 대한 모든 정보가 표시되지 않습니다. 업로드 및 패키지 버전 생성을 중지하려면 Stop upload(업로드 중지)를 선택합니다.
17. Distributor에서 일부 소프트웨어 설치 파일을 업로드하지 못하는 경우에는 [업로드 실패(Upload failed)] 메시지가 표시됩니다. 업로드를 다시 시도하려면 Retry upload(업로드 재시도)를 선택합니다. 패키지 버전 생성 실패에 따른 문제 해결 방법에 대한 자세한 내용은 AWS Systems ManagerDistributor 문제 해결 섹션을 참조하세요.
18. Distributor가 새로운 패키지 버전 생성을 마치면 해당 패키지 세부 정보 페이지의 버전 탭에서 사용 가능한 패키지 버전 목록에 새로운 버전이 있는지 확인합니다. 버전을 추가한 다음 기본 버전 설정을 선택하여 패키지의 기본 버전을 설정합니다.

    기본 버전을 설정하지 않으면 최신 패키지 버전이 기본 버전입니다.

패키지 버전 추가(고급)

패키지 버전을 추가하려면 패키지를 생성한 후 Distributor를 사용해 이전 버전에 존재하는 SSM 문서에 항목을 추가하여 패키지 버전을 추가합니다. 시간을 절약하기 위해 패키지의 이전 버전에 대한 매니페스트를 업데이트하고 매니페스트에서 `version` 항목의 값을 변경한 다음(예: `Test_1.0`에서 `Test_2.0`으로 변경) 새 버전의 매니페스트로 저장합니다. 고급 워크플로우에서 새로운 패키지 버전을 추가하려면 업데이트된 매니패스트가 필요합니다.

패키지 버전을 추가하려면(고급)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Distributor를 선택합니다.
3. Distributor 홈페이지에서 다른 버전을 추가할 패키지와 Add version(버전 추가)를 차례대로 선택합니다.
4. Version name(버전 이름)에 매니페스트 파일의 `version` 항목에 있는 값을 입력합니다.

5. S3 버킷 이름의 목록에서 기존 S3 버킷을 선택합니다. 이전 버전에서 설치 파일을 저장했던 버킷을 그대로 선택할 수도 있지만 이때는 버킷에 저장된 기존 설치 파일을 덮어쓰지 않도록 설치 파일 이름이 달라야 합니다.
6. S3 키 접두사에 설치 가능한 자산이 저장되는 버킷의 하위 폴더를 입력합니다.
7. .zip 파일과 함께 S3 버킷에 업로드한 매니페스트를 사용하려면 Manifest(매니페스트)에서 Extract from package(패키지에서 추출)를 선택합니다.

   (옵션) .zip 파일이 저장된 Amazon S3 버킷에 수정된 JSON 매니페스트를 업로드하지 않았다면 [새 매니페스트(New manifest)]를 선택합니다. JSON 편집기 필드에서 전체 매니페스트를 작성하거나 붙여넣을 수 있습니다. JSON 매니페스트를 생성하는 자세한 방법은 2단계: JSON 패키지 매니페스트 생성 섹션을 참조하세요.
8. 매니페스트 작성을 마치면 [패키지 버전 추가(Add package version)]를 선택합니다.
9. 패키지 세부 정보 페이지의 버전 탭에서 사용 가능한 패키지 버전 목록의 새 버전을 확인합니다. 버전을 추가한 다음 기본 버전 설정을 선택하여 패키지의 기본 버전을 설정합니다.

   기본 버전을 설정하지 않으면 최신 패키지 버전이 기본 버전입니다.

패키지 버전 추가(AWS CLI)

AWS CLI을 사용하여 Distributor에 새 패키지 버전을 추가할 수 있습니다. 이러한 명령을 실행하기 전에 이 주제 시작 부분에서 설명한 것처럼 새 패키지 버전을 생성해 S3에 업로드해야 합니다.

패키지 버전을 추가하려면(AWS CLI)

1. 다음 명령을 실행하여 새 패키지 버전에 대한 항목으로 AWS Systems Manager 문서를 편집합니다. *document-name*을 문서 이름으로 바꿉니다. *DOC-EXAMPLE-BUCKET*을 3단계: Amazon S3 버킷에 패키지 및 매니페스트 업로드에서 복사한 JSON 매니페스트의 URL로 바꿉니다. *S3-bucket-URL-of-package*는 전체 패키지가 저장된 Amazon S3 버킷의 URL입니다. *version-name-from-updated-manifest*를 매니페스트의 `version` 값으로 바꿉니다. `--document-version` 파라미터를 `$LATEST`로 설정해 이 패키지 버전과 연결된 문서를 최신 버전 문서로 지정합니다.

```
aws ssm update-document \
    --name "document-name" \
    --content "S3-bucket-URL-to-manifest-file" \
    --attachments Key="SourceUrl",Values="DOC-EXAMPLE-BUCKET" \
    --version-name version-name-from-updated-manifest \
    --document-version $LATEST
```

다음은 예입니다.

```
aws ssm update-document \
    --name ExamplePackage \
    --content "https://s3.amazonaws.com/DOC-EXAMPLE-BUCKET/ExamplePackage/
manifest.json" \
    --attachments Key="SourceUrl",Values="https://s3.amazonaws.com/DOC-EXAMPLE-
BUCKET/ExamplePackage" \
    --version-name 1.1.1 \
    --document-version $LATEST
```

2. 다음 명령을 실행하여 패키지가 업데이트되었는지 확인하고 패키지 매니페스트를 표시합니다. *package-name*은 패키지의 이름으로 바꾸고, 경우에 따라 *document-version*은 업데이트한 문서의 버전 번호(패키지 버전과 다름)로 바꿉니다. 이 패키지 버전이 문서의 최신 버전과 연결된 경우 선택적 `--document-version` 파라미터의 값에 대해 `$LATEST`를 지정할 수 있습니다.

```
aws ssm get-document \
    --name "package-name" \
    --document-version "document-version"
```

update-document 명령과 함께 사용할 수 있는 다른 옵션에 대한 자세한 내용은 AWS CLI Command Reference의 AWS Systems Manager 섹션에 있는 update-document를 참조하세요.

패키지 설치 또는 업데이트

AWS Systems Manager의 기능인 Distributor를 사용해 AWS Systems Manager 관리형 노드에 패키지를 배포할 수 있습니다. 패키지를 배포하려면 AWS Management Console 또는 AWS Command Line Interface(AWS CLI)를 사용합니다. 명령당 한 패키지의 버전을 하나만 배포할 수 있습니다. 새 패키지를 설치하거나 기존 설치를 인플레이스 업데이트할 수 있습니다. 특정 버전을 배포하도록 선택하거나 항상 패키지의 최신 버전을 배포하도록 선택할 수 있습니다. AWS Systems Manager의 기능인 State Manager를 사용하여 패키지를 설치하는 것이 좋습니다. State Manager를 사용하면 관리형 노드에서 항상 최신 버전의 패키지를 실행하도록 할 수 있습니다.

기본 설정	AWS Systems Manager 작업	추가 정보
패키지를 즉시 설치하거나 업데이트합니다.	Run Command	• 패키지를 한 번만 설치 또는 업데이트(콘솔)

기본 설정	AWS Systems Manager 작업	추가 정보
		• 패키지를 한 번만 설치(AWS CLI) • 패키지를 한 번만 업데이트(AWS CLI)
설치에 항상 기본 버전이 포함되도록 정기적으로 패키지를 설치 또는 업데이트합니다.	State Manager	• 패키지 설치 또는 업데이트 예약(콘솔) • 패키지 설치 예약(AWS CLI) • 패키지 업데이트 예약(AWS CLI)
특정 태그 또는 태그 세트가 있는 새 관리형 노드에 패키지를 자동으로 설치합니다. 예를 들어 새 인스턴스에 Amazon CloudWatch 에이전트를 설치합니다.	State Manager	이렇게 하기 위한 한 가지 방법은 새 인스턴스에 태그를 적용한 다음 State Manager 연결에서 해당 태그를 대상으로 지정하는 것입니다. 그러면 State Manager 태그가 일치하는 인스턴스에서 연결에 패키지를 자동으로 설치합니다. State Manager 연결에서의 대상 및 속도 제어 정보 섹션을 참조하세요.

주제

- 패키지를 한 번만 설치 또는 업데이트(콘솔)
- 패키지 설치 또는 업데이트 예약(콘솔)
- 패키지를 한 번만 설치(AWS CLI)
- 패키지를 한 번만 업데이트(AWS CLI)
- 패키지 설치 예약(AWS CLI)
- 패키지 업데이트 예약(AWS CLI)

패키지를 한 번만 설치 또는 업데이트(콘솔)

AWS Systems Manager 콘솔을 사용하여 패키지를 한 번만 설치 또는 업데이트할 수 있습니다. 일회성 설치를 구성하는 경우 Distributor는 AWS Systems Manager의 기능인 AWS Systems Manager Run Command를 사용하여 설치를 수행합니다.

패키지를 한 번 설치 또는 업데이트하려면 (콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Distributor를 선택합니다.
3. Distributor 홈 페이지에서 설치할 패키지를 선택합니다.
4. Install one time(한 번만 설치)을 선택합니다.

   이 명령은 Run Command를 열며, 명령 문서 `AWS-ConfigureAWSPackage` 및 Distributor 패키지가 이미 선택되어 있습니다.
5. 문서 버전에서 실행할 `AWS-ConfigureAWSPackage` 문서의 버전을 선택합니다.
6. 작업에서 설치를 선택합니다.
7. Installation type(설치 유형)에서 다음 중 하나를 선택합니다.

   - Uninstall and reinstall(제거 및 재설치): 패키지가 완전히 제거되었다가 다시 설치됩니다. 재설치가 완료될 때까지 애플리케이션을 사용할 수 없습니다.
   - [인플레이스 업데이트(In-place update)]: `update` 스크립트에서 제공한 지침에 따라 새 파일이나 변경된 파일만 기존 설치에 추가됩니다. 애플리케이션은 업데이트 프로세스 전체에서 사용할 수 있습니다. 이 옵션은 `AWSEC2Launch-Agent` 패키지를 제외한 AWS 게시 패키지에 대해 지원되지 않습니다.
8. 이름에 선택한 패키지 이름이 입력되었는지 확인합니다.
9. (선택 사항) 버전에 패키지의 버전 이름 값을 입력합니다. 이 필드를 비워 두면 Run Command에서는 Distributor에서 선택한 기본 버전을 설치합니다.
10. 대상 섹션에서, 태그를 지정하거나, 수동으로 관리형 노드를 선택하거나, 리소스 그룹을 지정하여 이 작업을 실행할 인스턴스 또는 장치를 식별합니다.

> **Note**
>
> 목록에 관리형 노드가 표시되지 않은 경우에는 관리형 노드 가용성 문제 해결 섹션을 참조하세요.

11. Other parameters(다른 파라미터):

    - Comment(설명)에 명령에 대한 정보를 입력합니다.
    - 제한 시간(초)에서 전체 명령 실행이 실패할 때까지 시스템이 기다리는 시간을 초 단위로 지정합니다.

12. 속도 제어(Rate control)에서:

    - 동시성(Concurrency)에서 명령을 동시에 실행할 대상의 백분율 또는 개수를 지정합니다.

      > **Note**
      >
      > 태그나 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상의 수를 제한합니다.

    - 오류 임계값에서, 명령이 관리형 노드의 개수 또는 백분율에서 실패한 후 다른 대상에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째 오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.

13. (선택 사항) Output options(출력 옵션)에서 명령 출력을 파일에 저장하려면 Write command output to an S3 bucket(S3 버킷에 명령 출력 쓰기) 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

    > **Note**
    >
    > 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아니라 인스턴스에 할당된 인스턴스 프로파일(EC2 인스턴스용) 또는 IAM 서비스 역할(하이브리드 정품 인증 시스템)의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

14. SNS notifications(SNS 알림) 섹션에서, 명령 실행 상태에 대한 알림이 전송되도록 하려면 Enable SNS notifications(SNS 알림 활성화) 확인란을 선택합니다.

    Run Command에 대한 Amazon SNS 알림 구성에 대한 자세한 내용은 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 섹션을 참조하세요.

15. 패키지를 설치할 준비가 되면 [실행(Run)]을 선택합니다.

16. Command status(명령 상태) 영역은 실행 진행률을 보고합니다. 명령이 계속 진행 중인 경우 Overall status(전체 상태) 또는 세부 상태 열에 성공 또는 실패가 표시될 때까지 콘솔의 왼쪽 위 모서리에 있는 새로 고침 아이콘을 선택합니다.
17. 대상 및 출력(Targets and output) 영역에서 관리형 노드 이름 옆에 있는 버튼을 선택한 후 출력 보기(View output)를 선택합니다.

    명령 출력 페이지에 명령 실행 결과가 표시됩니다.
18. (옵션) Amazon S3 버킷에 명령 출력을 쓰도록 선택한 경우 출력 로그 데이터를 볼 수 있도록 [Amazon S3]를 선택합니다.

패키지 설치 또는 업데이트 예약(콘솔)

AWS Systems Manager 콘솔을 사용하여 패키지 설치 또는 업데이트를 예약할 수 있습니다. 패키지 설치 또는 업데이트를 예약할 때 Distributor는 AWS Systems Manager State Manager을 사용해 설치 또는 업데이트합니다.

패키지 설치를 예약하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Distributor를 선택합니다.
3. Distributor 홈 페이지에서 설치 또는 업데이트할 패키지를 선택합니다.
4. Package(패키지)에서 Install on a schedule(일정에 따라 설치)을 선택합니다.

   이 명령은 자동으로 생성된 새 연결에 대한 State Manager를 엽니다.
5. 이름에 이름(예: **`Deploy-test-agent-package`**)을 입력합니다. 이는 선택 사항이며, 권장 사항은 아닙니다. 공백은 이름에 사용할 수 없습니다.
6. 문서 목록에는 문서 이름 `AWS-ConfigureAWSPackage`가 이미 선택되어 있습니다.
7. 작업에서 설치가 선택되어 있는지 확인합니다.
8. Installation type(설치 유형)에서 다음 중 하나를 선택합니다.

   - Uninstall and reinstall(제거 및 재설치): 패키지가 완전히 제거되었다가 다시 설치됩니다. 재설치가 완료될 때까지 애플리케이션을 사용할 수 없습니다.
   - [인플레이스 업데이트(In-place update)]: `update` 스크립트에서 제공한 지침에 따라 새 파일이나 변경된 파일만 기존 설치에 추가됩니다. 애플리케이션은 업데이트 프로세스 전체에서 사용할 수 있습니다.
9. 이름에 패키지 이름이 입력되었는지 확인합니다.

10. 최신 게시 버전 이외의 패키지 버전을 설치하려면 버전에 버전 식별자를 입력합니다.
11. 대상에서 이 계정의 모든 관리형 인스턴스 선택, 태그 지정 또는 수동으로 인스턴스 선택을 선택합니다. 태그를 사용하여 리소스 대상을 지정하는 경우 제공된 필드에 태그 키 및 태그 값을 입력합니다.

    > **Note**
    >
    > 이 계정의 모든 관리형 인스턴스 선택 또는 인스턴스 수동 선택 중 하나를 통해 관리형 AWS IoT Greengrass 코어 디바이스를 선택할 수 있습니다.

12. 일정 지정에서 정기적으로 연결을 실행하려면 일정이 있을 때를, 연결을 한 번만 실행하려면 일정이 없을 때를 선택합니다. 이러한 옵션에 대한 자세한 내용은 Systems Manager에서 연결 작업 섹션을 참조하세요. 컨트롤을 사용하여 cron 또는 연결에 대한 일정을 생성합니다.
13. 연결 생성을 선택합니다.
14. 연결 페이지에서 생성된 연결 옆에 있는 단추를 선택한 다음, 지금 연결 적용을 선택합니다.

    State Manager는 지정된 대상에 대해 연결을 생성하고 즉시 실행합니다. 실행 중인 연결의 결과에 대한 자세한 내용은 이 설명서의 Systems Manager에서 연결 작업 섹션을 참조하세요.

고급 옵션, 비율 제어(Rate control) 및 출력 옵션(Output options)에서의 옵션 작업에 대한 자세한 내용은 Systems Manager에서 연결 작업 섹션을 참조하세요.

패키지를 한 번만 설치(AWS CLI)

AWS CLI에서 send-command를 실행하여 Distributor 패키지를 한 번 설치할 수 있습니다. 패키지가 이미 설치되어 있으면 패키지가 제거되고 새 버전이 설치되는 동안 애플리케이션이 오프라인으로 전환됩니다.

패키지를 한 번 설치하려면(AWS CLI)

- AWS CLI에서 다음과 같은 명령을 실행합니다.

```
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**
>
> installationType에 대한 기본 동작은 Uninstall and reinstall입니다. 전체 패키지를 설치할 때 이 명령에서 "installationType":["Uninstall and reinstall"]을 생략할 수 있습니다.

다음은 예입니다.

```
aws ssm send-command \
    --document-name "AWS-ConfigureAWSPackage" \
    --instance-ids "i-00000000000000" \
    --parameters '{"action":["Install"],"installationType":["Uninstall and
 reinstall"],"name":["ExamplePackage"]}'
```

send-command 명령과 함께 사용할 수 있는 다른 옵션에 대한 자세한 내용은 AWS CLI Command Reference의 AWS Systems Manager 섹션에 있는 send-command를 참조하세요.

패키지를 한 번만 업데이트(AWS CLI)

AWS CLI에서 send-command을 실행하면 관련 애플리케이션을 오프라인으로 전환하지 않고 Distributor 패키지를 업데이트할 수 있습니다. 패키지의 새 파일이나 업데이트된 파일만 바뀝니다.

패키지를 한 번 업데이트하려면(AWS CLI)

- AWS CLI에서 다음과 같은 명령을 실행합니다.

```
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**
>
> 새 파일이나 변경된 파일을 추가할 때는 명령에 "installationType":["In-place update"]를 포함해야 합니다.

다음은 예입니다.

```
aws ssm send-command \
    --document-name "AWS-ConfigureAWSPackage" \
    --instance-ids "i-02573cafcfEXAMPLE" \
    --parameters '{"action":["Install"],"installationType":["In-place
 update"],"name":["ExamplePackage"]}'
```

send-command 명령과 함께 사용할 수 있는 다른 옵션에 대한 자세한 내용은 AWS CLI Command Reference의 AWS Systems Manager 섹션에 있는 send-command를 참조하세요.

패키지 설치 예약(AWS CLI)

AWS CLI에서 create-association을 실행하여 일정에 따라 Distributor 패키지를 설치할 수 있습니다. `--name` 값 즉, 문서 이름은 항상 `AWS-ConfigureAWSPackage`입니다. 다음 명령은 키 `InstanceIds`를 사용하여 대상 관리형 노드를 지정합니다. 패키지가 이미 설치되어 있으면 패키지가 제거되고 새 버전이 설치되는 동안 애플리케이션이 오프라인으로 전환됩니다.

```
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`에 대한 기본 동작은 `Uninstall and reinstall`입니다. 전체 패키지를 설치할 때 이 명령에서 `"installationType":["Uninstall and reinstall"]`을 생략할 수 있습니다.

다음은 예입니다.

```
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\"]}]
```

create-association 명령과 함께 사용할 수 있는 다른 옵션에 대한 자세한 내용은 AWS CLI Command Reference의 AWS Systems Manager 섹션에 있는 create-association를 참조하세요.

패키지 업데이트 예약(AWS CLI)

AWS CLI에서 create-association을 실행하면 관련 애플리케이션을 오프라인으로 전환하지 않고 일정에 따라 Distributor 패키지를 업데이트할 수 있습니다. 패키지의 새 파일이나 업데이트된 파일만 바뀝니다. `--name` 값 즉, 문서 이름은 항상 `AWS-ConfigureAWSPackage`입니다. 다음 명령은 키 `InstanceIds`를 사용하여 대상 인스턴스를 지정합니다.

```
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**
>
> 새 파일이나 변경된 파일을 추가할 때는 명령에 `"installationType":["In-place update"]`를 포함해야 합니다.

다음은 예입니다.

```
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\"]}]
```

create-association 명령과 함께 사용할 수 있는 다른 옵션에 대한 자세한 내용은 AWS CLI Command Reference의 AWS Systems Manager 섹션에 있는 create-association를 참조하세요.

패키지 제거

AWS Management Console 또는 AWS Command Line Interface(AWS CLI)를 사용하여 Run Command로 AWS Systems Manager 관리형 노드에서 Distributor 패키지를 제거할 수 있습니다. Distributor와 Run Command는 AWS Systems Manager의 기능입니다. 이 릴리스에서는 명령당 패키지 하나의 버전을 하나씩만 제거할 수 있습니다. 특정 버전이나 기본 버전을 제거할 수 있습니다.

주제

패키지 제거(콘솔)

Systems Manager 콘솔에서 Run Command을 사용하여 패키지를 한 번 제거할 수 있습니다. Distributor에서는 AWS Systems Manager Run Command를 사용하여 패키지를 제거합니다.

패키지를 제거하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택합니다.
3. Run Command 홈 페이지에서 [Run command]를 선택합니다.
4. `AWS-ConfigureAWSPackage` 명령 문서를 선택합니다.
5. 작업에서 제거를 선택합니다.
6. 이름에 제거하려는 패키지 이름을 입력합니다.
7. 대상(Target)에서 관리형 노드를 대상으로 지정하는 방식을 선택합니다. 대상에서 공유하는 태그 키와 값을 지정할 수 있습니다. ID, 플랫폼 및 SSM Agent 버전 같은 속성을 선택하여 대상을 지정할 수도 있습니다.
8. 고급 옵션을 사용하여 작업에 대한 설명을 추가하거나, [속도 제어(Rate control)]에서 [동시성(Concurrency)] 및 [Error threshold(오류 임계값)] 변경하거나, 출력 옵션을 지정하거나, Amazon Simple Notification Service(Amazon SNS) 알림을 구성할 수 있습니다. 자세한 내용은 이 설명서의 콘솔에서 명령 실행을 참조하십시오.
9. 패키지를 제거할 준비가 되면 [실행(Run)]을 선택한 후 [결과 보기(View results)]를 선택합니다.
10. 명령 목록에서 실행한 `AWS-ConfigureAWSPackage` 명령을 선택합니다. 명령이 아직 진행 중인 경우 콘솔 오른쪽 위 모서리의 새로 고침 아이콘을 선택합니다.
11. 상태(Status) 열에 성공(Success) 또는 실패(Failed)가 표시되면 출력(Output) 탭을 선택합니다.

12. [출력 보기(View output)]를 선택합니다. 명령 출력 페이지에 명령 실행 결과가 표시됩니다.

패키지 제거(AWS CLI)

AWS CLI에서 Run Command를 사용하여 관리형 노드에서 Distributor 패키지를 제거할 수 있습니다.

패키지를 제거하려면(AWS CLI)

- AWS CLI에서 다음과 같은 명령을 실행합니다.

```
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)"]}'
```

다음은 예입니다.

```
aws ssm send-command \
    --document-name "AWS-ConfigureAWSPackage" \
    --instance-ids "i-02573cafcfEXAMPLE" \
    --parameters '{"action":["Uninstall"],"name":["Test-ConfigureAWSPackage"]}'
```

send-command 명령과 함께 사용할 수 있는 다른 옵션에 대한 자세한 내용은 AWS CLI Command Reference의 AWS Systems Manager 섹션에 있는 send-command를 참조하세요.

패키지 삭제

이 섹션은 패키지를 삭제하는 방법을 설명합니다. 패키지 버전은 삭제할 수 없으며 전체 패키지만 삭제할 수 있습니다.

패키지 삭제(콘솔)

AWS Systems Manager 콘솔을 사용하여 AWS Systems Manager의 기능인 Distributor에서 패키지 또는 패키지 버전을 삭제할 수 있습니다. 이 패키지를 삭제하면 Distributor에서 모든 패키지 버전이 삭제됩니다.

패키지를 삭제하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.

2. 탐색 창에서 Distributor를 선택합니다.
3. Distributor 홈 페이지에서 삭제할 패키지를 선택합니다.
4. 패키지 세부 정보 페이지에서 패키지 삭제를 선택합니다.
5. 삭제를 확인하라는 메시지가 표시되면 [패키지 삭제(Delete package)]를 선택합니다.

패키지 버전 삭제(콘솔)

Systems Manager 콘솔을 사용하여 Distributor에서 패키지 버전을 삭제할 수 있습니다.

패키지 버전을 삭제하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Distributor를 선택합니다.
3. Distributor 홈 페이지에서 삭제할 패키지 버전을 선택합니다.
4. 패키지의 버전 페이지에서 삭제할 버전을 선택하고 버전 삭제를 선택합니다.
5. 삭제를 확인하라는 메시지가 표시되면 [패키지 버전 삭제(Delete package version)]를 선택합니다.

패키지 삭제(명령줄)

선호하는 명령줄 도구를 사용하여 Distributor에서 패키지를 삭제할 수 있습니다.

Linux & macOS

패키지를 삭제하려면(AWS CLI)

1. 다음 명령을 실행하여 특정 패키지에 대한 문서를 나열합니다. 이 명령의 결과에서 삭제하려는 패키지를 찾습니다.

```
aws ssm list-documents \
    --filters Key=Name,Values=package-name
```

2. 다음 명령을 실행하여 패키지를 삭제합니다. *package-name*을 패키지 파일 이름으로 바꿉니다.

```
aws ssm delete-document \
    --name "package-name"
```

3. list-documents 명령을 다시 실행하여 패키지가 삭제되었는지 확인합니다. 삭제한 패키지가 목록에 포함되어서는 안 됩니다.

```
aws ssm list-documents \
    --filters Key=Name,Values=package-name
```

Windows

패키지를 삭제하려면(AWS CLI)

1. 다음 명령을 실행하여 특정 패키지에 대한 문서를 나열합니다. 이 명령의 결과에서 삭제하려는 패키지를 찾습니다.

```
aws ssm list-documents ^
    --filters Key=Name,Values=package-name
```

2. 다음 명령을 실행하여 패키지를 삭제합니다. *package-name*을 패키지 파일 이름으로 바꿉니다.

```
aws ssm delete-document ^
    --name "package-name"
```

3. list-documents 명령을 다시 실행하여 패키지가 삭제되었는지 확인합니다. 삭제한 패키지가 목록에 포함되어서는 안 됩니다.

```
aws ssm list-documents ^
    --filters Key=Name,Values=package-name
```

PowerShell

패키지를 삭제하려면(Tools for PowerShell)

1. 다음 명령을 실행하여 특정 패키지에 대한 문서를 나열합니다. 이 명령의 결과에서 삭제하려는 패키지를 찾습니다.

```
$filter = New-Object
 Amazon.SimpleSystemsManagement.Model.DocumentKeyValuesFilter
$filter.Key = "Name"
$filter.Values = "package-name"
```

```
Get-SSMDocumentList `
    -Filters @($filter)
```

2. 다음 명령을 실행하여 패키지를 삭제합니다. *package-name*을 패키지 파일 이름으로 바꿉니다.

```
Remove-SSMDocument `
    -Name "package-name"
```

3. Get-SSMDocumentList 명령을 다시 실행하여 패키지가 삭제되었는지 확인합니다. 삭제한 패키지가 목록에 포함되어서는 안 됩니다.

```
$filter = New-Object
 Amazon.SimpleSystemsManagement.Model.DocumentKeyValuesFilter
$filter.Key = "Name"
$filter.Values = "package-name"

Get-SSMDocumentList `
    -Filters @($filter)
```

패키지 버전 삭제(명령줄)

선호하는 명령줄 도구를 사용하여 Distributor에서 패키지 버전을 삭제할 수 있습니다.

Linux & macOS

패키지 버전을 삭제하려면(AWS CLI)

1. 다음 명령을 실행하여 패키지 버전을 나열합니다. 이 명령의 결과에서 삭제하려는 패키지 버전을 찾습니다.

```
aws ssm list-document-versions \
    --name "package-name"
```

2. 다음 명령을 실행하여 패키지 버전을 삭제합니다. *package-name*을 패키지 이름으로 바꾸고 *version*을 버전 번호로 바꿉니다.

```
aws ssm delete-document \
    --name "package-name" \
```

```
    --document-version version
```

3. list-document-versions 명령을 실행하여 패키지 버전이 삭제되었는지 확인합니다. 삭제한 패키지 버전은 검색되지 않아야 합니다.

```
aws ssm list-document-versions \
    --name "package-name"
```

Windows

패키지 버전을 삭제하려면(AWS CLI)

1. 다음 명령을 실행하여 패키지 버전을 나열합니다. 이 명령의 결과에서 삭제하려는 패키지 버전을 찾습니다.

```
aws ssm list-document-versions ^
    --name "package-name"
```

2. 다음 명령을 실행하여 패키지 버전을 삭제합니다. *package-name*을 패키지 이름으로 바꾸고 *version*을 버전 번호로 바꿉니다.

```
aws ssm delete-document ^
    --name "package-name" ^
    --document-version version
```

3. list-document-versions 명령을 실행하여 패키지 버전이 삭제되었는지 확인합니다. 삭제한 패키지 버전은 검색되지 않아야 합니다.

```
aws ssm list-document-versions ^
    --name "package-name"
```

PowerShell

패키지 버전을 삭제하려면(Tools for PowerShell)

1. 다음 명령을 실행하여 패키지 버전을 나열합니다. 이 명령의 결과에서 삭제하려는 패키지 버전을 찾습니다.

```
Get-SSMDocumentVersionList `
```

```
    -Name "package-name"
```

2. 다음 명령을 실행하여 패키지 버전을 삭제합니다. *package-name*을 패키지 이름으로 바꾸고 *version*을 버전 번호로 바꿉니다.

```
Remove-SSMDocument `
    -Name "package-name" `
    -DocumentVersion version
```

3. Get-SSMDocumentVersionList 명령을 실행하여 패키지 버전이 삭제되었는지 확인합니다. 삭제한 패키지 버전은 검색되지 않아야 합니다.

```
Get-SSMDocumentVersionList `
    -Name "package-name"
```

list-documents 명령과 함께 사용할 수 있는 다른 옵션에 대한 자세한 내용은 AWS CLI Command Reference의 AWS Systems Manager 섹션에 있는 list-documents를 참조하세요. delete-document 명령과 함께 사용할 수 있는 다른 옵션에 대한 자세한 내용은 delete-document를 참조하십시오.

Distributor 활동 감사 및 로깅

AWS CloudTrail을 사용하여 AWS Systems Manager의 기능인 Distributor와 관련된 활동을 감사할 수 있습니다. Systems Manager에 대한 자세한 감사 및 로깅 옵션은 AWS Systems Manager 모니터링 섹션을 참조하세요.

CloudTrail을 사용하여 Distributor 활동 감사

CloudTrail에서는 AWS Systems Manager 콘솔, AWS Command Line Interface(AWS CLI) 및 Systems Manager SDK에서 수행된 API 호출을 캡처합니다. CloudTrail 콘솔에서 정보를 보거나 Amazon Simple Storage Service(Amazon S3) 버킷에 정보를 저장할 수 있습니다. 계정에 대한 모든 CloudTrail 로그를 저장하는 데 버킷 하나가 사용됩니다.

Run Command 및 State Manager 작업 로그는 문서 생성, 패키지 설치 및 패키지 제거 활동을 보여줍니다. Run Command와 State Manager는 AWS Systems Manager의 기능입니다. Systems Manager 활동의 CloudTrail 로그 보기 및 사용에 대한 자세한 내용은 AWS CloudTrail을 사용하여 AWS Systems ManagerAPI 호출을 로깅 섹션을 참조하세요.

AWS Systems ManagerDistributor 문제 해결

다음 정보는 AWS Systems Manager의 기능인 Distributor 사용 시 발생할 수 있는 문제를 해결하는 데 유용합니다.

주제

이름이 같은 잘못된 패키지가 설치됨

문제: 패키지를 설치했으나 Distributor가 대신 다른 패키지를 설치했습니다.

원인: 설치 중 Systems Manager에서 사용자 정의 외부 패키지보다 앞서 AWS 게시 패키지를 결과로 찾았습니다. 사용자 정의 패키지 이름이 AWS 게시 패키지 이름과 동일한 경우 사용자 패키지 대신 AWS 패키지가 설치됩니다.

해결 방법: 이 문제를 피하려면 패키지의 이름을 AWS 게시 패키지의 이름과 다르게 지정해야 합니다.

오류: 매니페스트를 조회하지 못함: 패키지의 최신 버전을 찾을 수 없음

문제: 다음과 같은 오류가 표시됩니다.

```
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
```

원인: Distributor에서 2.3.274.0 버전 이전의 SSM Agent를 사용합니다.

솔루션: SSM Agent 버전을 2.3.274.0 버전 이상으로 업데이트하십시오. 자세한 내용은 Run Command를 사용하여 SSM Agent 업데이트 또는 시연: SSM Agent(CLI) 자동 업데이트 섹션을 참조하세요.

오류: 매니페스트를 검색하는 데 실패함: 검증 예외

문제: 다음과 같은 오류가 표시됩니다.

```
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
```

원인: Distributor에서 2.3.274.0 버전 이전의 SSM Agent를 사용합니다.

솔루션: SSM Agent 버전을 2.3.274.0 버전 이상으로 업데이트하십시오. 자세한 내용은 Run Command를 사용하여 SSM Agent 업데이트 또는 시연: SSM Agent(CLI) 자동 업데이트 섹션을 참조하세요.

패키지가 지원되지 않음(패키지에 설치 작업이 없음)

문제: 다음과 같은 오류가 표시됩니다.

```
Package is not supported (package is missing install action)
```

원인: 패키지 디렉터리 구조가 올바르지 않습니다.

해결 방법: 소프트웨어 및 필수 스크립트가 포함된 상위 디렉터리를 압축하지 않습니다. 대신 절대 경로에 직접 필요한 모든 콘텐츠의 `.zip` 파일을 생성합니다. `.zip` 파일이 제대로 생성되었는지 확인하려면 대상 플랫폼 디렉터리의 압축을 풀고 디렉터리 구조를 검토합니다. 예를 들어 설치 스크립트 절대 경로는 */ExamplePackage_targetPlatform*`/install.sh`여야 합니다.

오류: 매니페스트를 다운로드하지 못했습니다. 이름이 있는 문서가 존재하지 않습니다.

문제: 다음과 같은 오류가 표시됩니다.

```
Failed to download manifest - failed to retrieve package document description:
 InvalidDocument: Document with name filename does not exist.
```

원인: 다른 계정에서 Distributor 패키지를 공유할 때 Distributor가 패키지 이름으로 패키지를 찾을 수 없습니다.

솔루션: 다른 계정에서 패키지를 공유할 때 패키지 이름뿐만 아니라 패키지의 전체 Amazon 리소스 이름(ARN)도 사용합니다.

업로드에 실패했습니다.

문제: 다음과 같은 오류가 표시됩니다.

```
Upload failed. At least one of your files was not successfully uploaded to your S3
 bucket.
```

원인: 소프트웨어 패키지 이름에 공백이 있습니다. 예를 들어, `Hello World.msi` 업로드에 실패할 수 있습니다.

AWS Systems Manager 공유 리소스

Systems Manager에서 AWS 리소스 관리 및 구성에 사용하는 공유 리소스는 다음과 같습니다.

주제

- AWS Systems Manager Documents

AWS Systems Manager Documents

AWS Systems Manager 문서(SSM 문서)는 Systems Manager가 관리형 인스턴스에서 실행하는 작업을 정의합니다. Systems Manager에는 런타임에 파라미터를 지정하여 사용할 수 있는 사전 구성 문서가 100개 이상 포함되어 있습니다. Amazon 소유(Owned by Amazon) 탭을 선택하거나 `ListDocuments` API 작업 호출 시 `Owner` 필터에 대해 Amazon을 지정하여 Systems Manager 문서(Systems Manager Documents) 콘솔에서 사전 구성된 문서를 찾을 수 있습니다. 문서는 JavaScript Object Notation(JSON) 또는 YAML을 사용하며 사용자가 지정하는 단계와 파라미터를 포함합니다. SSM 문서를 시작하려면 Systems Manager 콘솔을 엽니다. 탐색 창에서 Documents를 선택합니다.

Documents 기능이 조직에 주는 이점은 무엇인가요?

AWS Systems Manager의 기능인 Documents는 다음과 같은 이점을 제공합니다.

- 문서 범주

  필요한 문서를 찾는 데 도움이 되도록 검색 중인 문서 유형에 따라 범주를 선택합니다. 검색 범위를 넓히기 위해 동일한 문서 유형의 여러 범주를 선택할 수 있습니다. 서로 다른 문서 유형의 범주 선택은 지원되지 않습니다. 범주는 Amazon이 소유한 문서에만 지원됩니다.

- 문서 버전

  다양한 버전의 문서를 작성하여 저장할 수 있습니다. 그런 다음 각 문서의 기본 버전을 지정할 수 있습니다. 문서의 기본 버전은 새로운 버전으로 업데이트하거나 이전 버전으로 되돌릴 수 있습니다. 문서의 콘텐츠를 변경하면 Systems Manager가 자동으로 문서의 버전을 높입니다. 콘솔, AWS Command Line Interface(AWS CLI) 명령 또는 API 호출에서 문서 버전을 지정하여 문서의 모든 버전을 검색하거나 사용할 수 있습니다.

- 필요에 맞게 문서 사용자 지정

문서에서 각 단계 및 작업을 사용자 정의하고 싶다면 자신만의 고유한 문서를 생성할 수 있습니다. 시스템은 문서를 생성한 AWS 리전의 AWS 계정으로 문서를 저장합니다. SSM 문서 생성 방법에 대한 자세한 내용은 SSM 문서 콘텐츠 생성 섹션을 참조하세요.

- 문서에 태그 지정

  문서에 태그를 지정하면 문서에 지정한 태그를 토대로 하나 이상의 문서를 빠르게 식별하는 데 도움이 될 수 있습니다. 예를 들어 특정 환경, 부서, 사용자, 그룹 또는 기간의 문서에 태그를 지정할 수 있습니다. 또한 사용자나 그룹이 액세스할 수 있는 태그를 지정하는 AWS Identity and Access Management(IAM) 정책을 생성하여 문서에 대한 액세스를 제한할 수도 있습니다. 자세한 내용은 Systems Manager 문서에 태그 지정 단원을 참조하십시오.

- 문서 공유

  문서를 퍼블릭으로 설정하거나 동일한 AWS 리전에서 특정 AWS 계정와 공유할 수 있습니다. 고객 또는 직원에게 공급하는 모든 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스의 구성이 동일하기를 바라는 경우 계정 간 문서 공유가 유용할 수 있습니다. 인스턴스의 애플리케이션 및 패치를 최신 상태를 유지하는 것 외에도 고객 인스턴스의 특정 활동을 제한하고자 할 수도 있습니다. 또는 조직 전체의 직원 계정에서 사용하는 인스턴스가 특정 내부 리소스에 대해 액세스가 허용되도록 하고자 할 수도 있습니다. 자세한 내용은 SSM 문서 공유 단원을 참조하십시오.

누가 Documents를 사용해야 하나요?

- Systems Manager 기능을 사용하여 규모에 맞게 운영 효율성을 개선하고, 수동 개입과 관련된 오류를 줄이고, 일반적인 문제 해결 시간을 단축하고자 하는 AWS 고객
- 배포 및 구성 태스크를 자동화하려는 인프라 전문가
- 일반적인 문제를 안정적으로 해결하고, 문제 해결 효율성을 높이고, 반복 작업을 줄이려는 관리자
- 일반적으로 수동으로 수행하는 태스크를 자동화하려는 사용자

SSM 문서의 유형은 무엇인가요?

다음 표에서는 여러 유형의 SSM 문서와 그 용도에 대해 설명합니다.

유형	용도	Details
ApplicationConfiguration	AWS AppConfig	AWS Systems Manager의 기능 AWS AppConfig를 사용하

유형	용도	Details
ApplicationConfigurationSchema		여 애플리케이션 구성을 생성, 관리 및 신속하게 배포합니다. `ApplicationConfiguration` 문서 유형을 사용하는 문서를 만들어 SSM 문서에 구성 데이터를 저장할 수 있습니다. 자세한 내용은 AWS AppConfig 사용 설명서의 자유형 구성 단원을 참조하십시오. SSM 문서에서 구성을 생성하는 경우 해당 JSON 스키마를 지정해야 합니다. 스키마는 `ApplicationConfigurationSchema` 문서 유형을 사용하여, 규칙 세트와 같이 각 애플리케이션 구성 설정에 대해 허용 가능한 속성을 정의합니다. 자세한 내용은 AWS AppConfig 사용 설명서의 검사기 정보 단원을 참조하십시오.

유형	용도	Details
Automation 실행서	자동화 State Manager Maintenance Windows	Automation 실행서는 Amazon Machine Image(AMI) 생성 또는 업데이트와 같은 일반적인 유지 관리 및 배포 태스크를 실행할 때 사용됩니다. State Manager는 Automation 실행서를 사용하여 구성을 적용합니다. 이 작업은 인스턴스 수명 주기 중 언제든지 하나 이상의 대상에서 실행할 수 있습니다. Maintenance Windows에서는 Automation 실행서를 사용하여 지정된 일정을 기반으로 일반 유지 관리 및 배포 태스크를 수행합니다. Linux 기반 운영 체제에서 지원되는 모든 Automation 실행서는 macOS용 EC2 인스턴스에서도 지원됩니다.

유형	용도	Details
Change Calendar 문서	Change Calendar	AWS Systems Manager의 기능인 Change Calendar는 `ChangeCalendar` 문서 유형을 사용합니다. Change Calendar 문서는 자동화 작업에서 환경 변경을 허용하거나 금지할 수 있는 일정 항목 및 연관된 이벤트를 저장합니다. Change Calendar에서 문서는 iCalendar 2.0 데이터를 일반 텍스트 형식으로 저장합니다. Change Calendar는 macOS용 EC2 인스턴스에서 지원되지 않습니다.

유형	용도	Details
AWS CloudFormation 템플릿	AWS CloudFormation	AWS CloudFormation 템플릿은 CloudFormation 스택에서 프로비저닝할 리소스에 대해 설명합니다. CloudFormation 템플릿을 Systems Manager 문서로 저장하여 Systems Manager 문서 기능을 활용할 수 있습니다. 여기에는 여러 버전의 템플릿 생성 및 비교와 같은 AWS 리전에 있는 다른 계정과 템플릿 공유가 포함됩니다. Systems Manager의 기능인 Application Manager를 사용하여 CloudFormation 템플릿 및 스택을 생성하고 편집할 수 있습니다. 자세한 내용은 Application Manager에서 AWS CloudFormation 템플릿 및 스택 작업 단원을 참조하십시오.

유형	용도	Details
명령 문서	Run Command State Manager Maintenance Windows	AWS Systems Manager의 기능인 Run Command는 Command 문서를 사용하여 명령을 실행합니다. AWS Systems Manager의 기능인 State Manager는 Command 문서를 사용하여 구성을 적용합니다. 이 작업은 인스턴스 수명 주기 중 언제든지 하나 이상의 대상에서 실행할 수 있습니다. AWS Systems Manager의 기능인 Maintenance Windows에서는 Command 문서를 사용하여 지정된 일정을 기반으로 구성을 적용합니다. 대부분의 Command 문서는 모든 Linux 및 Systems Manager에서 지원하는 Windows Server 운영 체제에서 지원됩니다. 다음 Command 문서는 macOS용 EC2 인스턴스에서 지원됩니다. • `AWS-ConfigureAWSPackage` • `AWS-RunPatchBaseline` • `AWS-RunPatchBaselineAssociation` • `AWS-RunShellScript`

유형	용도	Details
AWS Config 규정 준수 팩 템플릿	AWS Config	AWS Config 규정 준수 팩 템플릿은 AWS Config 관리형 또는 사용자 지정 규칙 및 수정 작업 목록을 포함하는 규정 준수 팩을 생성하는 데 사용되는 YAML 형식의 문서입니다. 자세한 내용을 알아보려면 규정 준수 팩을 참조하세요.
패키지 문서	Distributor	AWS Systems Manager의 기능인 Distributor에서 패키지는 SSM 문서로 표현됩니다. 패키지 문서에는 관리형 인스턴스에 설치할 소프트웨어 또는 자산이 포함된 연결된 ZIP 아카이브 파일이 포함됩니다. Distributor에서 패키지를 생성하면 패키지 문서가 생성됩니다. Distributor는 Oracle Linux와 macOS 관리형 인스턴스에서 지원되지 않습니다.

유형	용도	Details
정책 문서	State Manager	AWS Systems Manager의 기능인 Inventory는 State Manager 연결이 있는 `AWS-GatherSoftware Inventory` Policy 문서를 사용하여 관리형 인스턴스에서 인벤토리 데이터를 수집합니다. 자체 SSM 문서 생성 시 Automation 실행서와 Command 문서는 관리형 인스턴스에서 정책을 적용하는 데 선호되는 방법입니다. Systems Manager Inventory 및 `AWS-GatherSoftware Inventory` Policy 문서는 Systems Manager에서 지원하는 모든 운영 체제에서 지원됩니다.
인시던트 후 분석 템플릿	Incident Manager 인시던트 후 분석	Incident Manager는 인시던트 후 분석 템플릿을 사용하여 AWS 운영 관리 모범 사례를 기반으로 분석을 생성합니다. 템플릿을 사용하여 팀에서 인시던트 대응에 대한 개선 사항을 식별하는 데 사용할 수 있는 분석을 생성합니다.

유형	용도	Details
세션 문서	Session Manager	AWS Systems Manager의 기능인 Session Manager에서는 Session 문서를 사용하여 시작할 세션의 유형(예: 포트 전달 세션, 대화형 명령을 실행할 세션, SSH 터널을 생성할 세션)을 결정합니다. Session 문서는 모든 Linux 및 Systems Manager에서 지원하는 Windows Server 운영 체제에서 지원됩니다. 다음 Command 문서는 macOS용 EC2 인스턴스에서 지원됩니다. • `AWS-PasswordReset` • `AWS-StartInteractiveCommand` • `AWS-StartPortForwardingSession` • `AWS-StartPortForwardingSessionToSocket` • `AWS-StartSSHSession`

SSM 문서 할당량

SSM 문서 할당량에 대한 내용은 Amazon Web Services 일반 참조의 Systems Manager 서비스 할당량을 참조하세요.

주제

- 문서 구성 요소
- SSM 문서 콘텐츠 생성

- 문서 작업

문서 구성 요소

이 섹션에는 SSM 문서를 구성하는 구성 요소에 대한 정보가 포함되어 있습니다.

내용

- 스키마, 특성 및 예제
- 데이터 요소 및 파마미터
- Command 문서 플러그인 참조

스키마, 특성 및 예제

AWS Systems Manager 문서는 다음 스키마 버전을 사용합니다.

- `Command` 유형의 문서는 스키마 버전 1.2, 2.0 및 2.2을 사용해야 합니다. 1.2 문서를 사용하는 경우 스키마 버전 2.2를 사용하는 문서 생성을 권장합니다.
- `Policy` 유형의 문서는 스키마 버전 2.0 이상을 사용해야 합니다.
- `Automation` 유형의 문서는 스키마 버전 0.3을 사용해야 합니다.
- JSON 또는 YAML에서 문서를 만들 수 있습니다.

`Command` 및 `Policy` 문서에 최신 스키마 버전을 사용하여 다음 기능을 활용할 수 있습니다.

스키마 버전 2.2 문서 기능

기능	Details
문서 편집	이제 문서를 업데이트할 수 있습니다. 버전 1.2에서는 문서를 업데이트하려면 다른 이름으로 저장해야 했습니다.
자동 버전 관리	문서로 업데이트하면 새로운 버전이 만들어집니다. 이것은 스키마가 아닌 문서 버전입니다.
기본 버전	문서 버전이 다수인 경우에는 기본 문서로 사용할 버전을 지정할 수 있습니다.

기능	Details
시퀀싱	문서의 플러그인 또는 단계는 지정된 순서로 실행됩니다.
교차 플랫폼 지원	크로스 플랫폼 지원으로 동일한 SSM 문서 내에서 플러그인마다 다른 운영 체제를 지정할 수 있습니다. 교차 플랫폼 지원 기능은 단계 내에서 `precondition` 파라미터를 사용합니다.

> **Note**
>
> 새로운 Systems Manager 기능과 SSM 문서 기능을 사용하려면 인스턴스의 AWS Systems Manager SSM Agent를 최신 버전으로 업데이트하여 최신 상태를 유지해야 합니다. 자세한 내용은 Run Command를 사용하여 SSM Agent 업데이트 단원을 참조하십시오.

다음 표에는 주요 스키마 버전의 차이점이 나열되어 있습니다.

버전 1.2	버전 2.2(최신 버전)	Details
runtimeConfig	mainSteps	버전 2.2에서는 `mainSteps` 섹션이 `runtimeConfig` 섹션으로 대체되었습니다. `mainSteps` 섹션을 사용하면 Systems Manager가 단계를 순서대로 실행할 수 있습니다.
properties	inputs	버전 2.2에서는 `properties` 섹션이 `inputs` 섹션으로 대체되었습니다. `inputs` 섹션은 각 단계의 파라미터를 허용합니다.
명령	runCommand	버전 2.2에서는 `inputs` 섹션이 `commands` 파라미터 대신

버전 1.2	버전 2.2(최신 버전)	Details
		runCommand 파라미터를 사용합니다.
id	작업	버전 2.2에서는 ID가 Action으로 대체되었습니다. 여기에서는 이름만 변경되었을 뿐입니다.
해당 사항 없음	이름	버전 2.2 에서 name은 단계를 나타내는 사용자 정의 이름입니다.

Precondition 파라미터 사용

스키마 버전 2.2 이상에서는 precondition 파라미터를 사용하여 플러그인마다 대상 운영 체제를 지정하거나 SSM 문서에 정의한 입력 파라미터를 검증할 수 있습니다. precondition 파라미터는 SSM 문서의 입력 파라미터 참조 및 platformType의 Linux, MacOS, Windows 값 사용을 지원합니다. StringEquals 연산자만 지원됩니다.

스키마 버전 2.2 이상을 사용하는 문서에서 precondition 파라미터를 지정하지 않으면 플러그인과 운영 체제의 호환성에 따라 각 플러그인이 실행되거나 생략됩니다. 운영 체제와의 플러그인 호환성은 precondition보다 먼저 평가됩니다. 반대로 스키마 버전 2.0 이하를 사용하는 문서에서는 비호환 플러그인을 실행하려고 하면 오류가 발생합니다.

예를 들어 스키마 버전 2.2 문서에서 precondition이 지정되지 않고 aws:runShellScript 플러그인이 나열되면 Linux 인스턴스에서는 단계가 실행되지만, aws:runShellScript가 Windows Server 인스턴스와 호환되지 않기 때문에 Windows Server 인스턴스에서는 이 단계를 건너뜁니다. 하지만 스키마 버전 2.0 문서에서는 aws:runShellScript 플러그인을 지정하더라도 Windows Server 인스턴스에서 문서를 실행하면 실행이 실패합니다. 이 섹션 후반부의 SSM 문서에 있는 precondition 파라미터의 예를 참조하세요.

스키마 버전 2.2

최상위 수준 요소

다음은 스키마 버전 2.2를 사용하여 SSM 문서의 최상위 요소를 나타낸 예제입니다.

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 }}"
```

```
         }
      }
   ]
}
```

스키마 버전 2.2 예제

다음 예제에서는 aws:runPowerShellScript 플러그인을 사용하여 대상 인스턴스에서 PowerShell 명령을 실행합니다.

YAML

```
---
schemaVersion: "2.2"
description: "Example document"
parameters:
  Message:
    type: "String"
    description: "Example parameter"
    default: "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"
      }
   },
   "mainSteps": [
      {
```

```
         "action": "aws:runPowerShellScript",
         "name": "example",
         "inputs": {
            "timeoutSeconds": "60",
            "runCommand": [
               "Write-Output {{Message}}"
            ]
         }
      }
   ]
}
```

스키마 버전 2.2 precondition 파라미터 예제

스키마 버전 2.2는 교차 플랫폼 지원 기능을 제공합니다. 이 말은 단일 SSM 문서 내에서 플러그인마다 다른 운영 체제를 지정할 수 있다는 것을 의미합니다. 교차 플랫폼 지원 기능은 다음 예제와 같이 단계 내에서 precondition 파라미터를 사용합니다. precondition 파라미터를 사용하여 SSM 문서에 정의한 입력 파라미터를 검증할 수도 있습니다. 다음 예 중 두 번째 예에서 이를 확인할 수 있습니다.

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"
            ]
         }
      }
   ]
```

```
}
```

스키마 버전 2.2 State Manager 예제

State Manager에서 Systems Manager의 기능인 다음과 같은 SSM 문서를 사용하여 ClamAV 안티바이러스 소프트웨어를 설치할 수 있습니다. State Manager에서는 특정 구성만 사용 가능합니다. 즉, State Manager 연결이 실행될 때마다 시스템이 ClamAV 소프트웨어가 설치되어 있는지 확인합니다. 설치되어 있지 않은 경우 State Manager은 이 문서를 다시 실행합니다.

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"
            ]
         }
      }
   ]
```

```
}
```

스키마 버전 2.2 인벤토리 예제

State Manager에서 다음과 같은 SSM 문서를 사용하여 해당 인스턴스에 대한 인벤토리 메타데이터를 수집할 수 있습니다.

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 }}"
            }
         }
      ]
}
```

스키마 버전 2.2 **AWS-ConfigureAWSPackage** 예제

다음은 AWS-ConfigureAWSPackage 문서를 나타낸 예제입니다. mainSteps 섹션에는 action 단계의 aws:configurePackage 플러그인이 포함되어 있습니다.

> **Note**
>
> Linux 운영 체제에서는 AmazonCloudWatchAgent 및 AWSSupport-EC2Rescue 패키지만 지원됩니다.

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 }}"
            }
         }
      ]
}
```

스키마 버전 1.2

다음은 스키마 버전 1.2 문서에서 최상위 요소를 나타낸 예제입니다.

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

스키마 버전 1.2 **aws:runShellScript** 예제

다음 예에서는 AWS-RunShellScript SSM 문서를 보여줍니다. runtimeConfig 섹션에는 aws:runShellScript 플러그인이 포함되어 있습니다.

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

스키마 버전 0.3

최상위 수준 요소

다음 예에서는 스키마 버전이 0.3인 JSON 형식 Automation 실행서의 최상위 요소를 보여줍니다.

```
{
    "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 실행서 예

다음 샘플은 Automation 실행서의 내용을 YAML 형식으로 보여줍니다. 문서 스키마 버전 0.3에 대한 이 작업 예제는 Markdown을 사용하여 문서 설명의 서식을 지정하는 방법을 보여줍니다.

```
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
```

데이터 요소 및 파마미터

이 주제에서는 SSM 문서에 사용되는 데이터 요소를 설명합니다. 문서를 생성하는 데 사용되는 스키마 버전은 문서에서 허용하는 구문과 데이터 요소를 정의합니다. Command 문서에는 스키마 버전 2.2 이상을 사용하는 것이 좋습니다. Automation 런북은 스키마 버전 0.3을 사용합니다. 또한 Automation 런북에서는 Markdown을 사용할 수 있습니다. 마크업 언어인 Markdown을 사용하면 문서와 문서 내의 개별 단계에 wiki 스타일의 설명을 추가할 수 있습니다. 마크다운 사용에 대한 자세한 내용은 AWS Management Console 시작 안내서의 콘솔에서 마크다운 사용을 참조하세요.

다음 섹션에서는 SSM 문서에 포함할 수 있는 데이터 요소에 대해 설명합니다.

최상위 데이터 요소

schemaVersion

사용할 스키마 버전입니다.

형식: 버전

필수 여부: 예

설명

문서 목적을 설명하는 정보입니다. 또한 이 필드를 사용하여 파라미터에 문서 실행 값이 필요한지 또는 파라미터 값을 제공하는 것이 선택 사항인지 여부를 지정할 수 있습니다. 필수 파라미터와 선택적 파라미터는 이 주제의 예에서 볼 수 있습니다.

타입: 문자열

필수 항목 여부: 아니요

파라미터

문서가 허용하는 파라미터를 정의하는 구조입니다.

자주 사용하는 파라미터의 경우 AWS Systems Manager의 기능인 Parameter Store에 해당 파라미터를 저장하는 것이 좋습니다. 그런 다음 Parameter Store 파라미터를 기본값으로 참조하는 파라미터를 문서에서 정의할 수 있습니다. Parameter Store 파라미터를 참조하려면 다음 구문을 사용합니다.

```
{{ssm:parameter-name}}
```

다른 문서 파라미터와 동일한 방식으로 Parameter Store 파라미터를 참조하는 파라미터를 사용할 수 있습니다. 다음 예제에서 `commands` 파라미터의 기본값은 Parameter Store 파라미터 `myShellCommands`입니다. `commands` 파라미터를 `runCommand` 문자열로 지정하면 문서에서 `myShellCommands` 파라미터에 저장된 명령을 실행합니다.

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**
>
> 문서의 `parameters` 섹션에서 `String` 및 `StringList` Parameter Store 파라미터를 참조할 수 있습니다. `SecureString` Parameter Store 파라미터를 참조할 수 없습니다.

Parameter Store에 대한 자세한 내용은 AWS Systems Manager Parameter Store 섹션을 참조하세요.

형식: 구조

`parameters` 구조는 다음의 필드 및 값을 수락합니다.

- `type`: (필수) 허용되는 값은 다음과 같습니다. `String`, `StringList`, `Integer` `Boolean`, `MapList`, `StringMap`. 각 유형의 예를 보려면 다음 단원의 SSM 문서 파라미터 `type` 예제를 참조하십시오.

> **Note**
>
> Command 유형 문서에서는 `String` 및 `StringList` 파라미터 유형만 지원합니다.

- `description`: (선택 사항) 파라미터에 대한 설명입니다.
- `default`: (선택 사항) 파라미터의 기본값 또는 Parameter Store 내 파라미터에 대한 참조입니다.
- `allowedValues`: (선택 사항) 파라미터에 허용되는 값의 배열입니다. 파라미터에 허용된 값을 정의하면 사용자 입력이 검증됩니다. 사용자가 허용되지 않는 값을 입력하면 실행이 시작되지 않습니다.

  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`: (선택 사항) 사용자 입력이 파라미터에 대해 정의된 패턴과 일치하는지 여부를 확인하는 정규 표현식입니다. 사용자 입력이 허용된 패턴과 일치하지 않으면 실행이 시작되지 않습니다.

> **Note**
>
> Systems Manager는 `allowedPattern`에 대한 두 가지 검증을 수행합니다. 첫 번째 유효성 검사는 문서를 사용할 때 API 레벨에서 Java regex 라이브러리를 사용하여 수행됩니다. 두 번째 유효성 검사는 문서를 처리하기 전에 GO regexp 라이브러리를 사용하여 SSM Agent에서 수행됩니다.

YAML

```
InstanceId:
  type: String
  description: "(Required) The instance ID to target."
  allowedPattern: "^i-[a-z0-9]{8,17}$"
  default: ''
```

JSON

```
"InstanceId": {
  "type": "String",
  "description": "(Required) The instance ID to target.",
  "allowedPattern": "^i-[a-z0-9]{8,17}$",
  "default": ""
}
```

- `displayType`: (선택 사항) AWS Management Console에서 `textfield` 또는 `textarea`를 표시하는 데 사용됩니다. `textfield`는 한 줄 텍스트 상자이고, `textarea`는 여러 줄 텍스트 상자입니다.
- `minItems`: (선택 사항) 허용되는 최소 항목 수입니다.
- `maxItems`: (선택 사항) 허용되는 최대 항목 수입니다.
- `minChars`: (선택 사항) 허용되는 파라미터 문자 최소 개수입니다.
- `maxChars`: (선택 사항) 허용되는 파라미터 문자 최대 개수입니다.

필수 항목 여부: 아니요

variables

(스키마 버전 0.3만 해당) Automation 런북의 전체 단계에서 참조하거나 업데이트할 수 있는 값입니다. 변수는 파라미터와 비슷하지만 매우 중요한 점에서 다릅니다. 런북의 컨텍스트에서는 파라미터

값이 고정되어 있지만 런북의 컨텍스트에서 변수 값을 변경할 수 있습니다. 변수 값을 업데이트할 때 데이터 유형은 정의된 데이터 유형과 일치해야 합니다. 자동화에서 변수 값을 업데이트하는 방법에 대한 자세한 내용은 aws:updateVariable - 런북 변수 값을 업데이트 섹션을 참조하세요.

형식: Boolean | Integer | MapList | String | StringList | StringMap

필수 항목 여부: 아니요

YAML

```
variables:
    payload:
        type: StringMap
        default: "{}"
```

JSON

```
{
    "variables": [
        "payload": {
            "type": "StringMap",
            "default": "{}"
        }
    ]
}
```

runtimeConfig

(스키마 버전 1.2만 해당) 하나 이상의 Systems Manager 플러그인에 의해 적용되는 인스턴스 구성입니다. 플러그인은 순서대로 실행되지 않습니다.

형식: Dictionary<string,PluginConfiguration>

필수 항목 여부: 아니요

mainSteps

(스키마 버전 0.3, 2.0 및 2.2만 해당) 여러 단계(플러그인)를 포함할 수 있는 객체입니다. 플러그인은 단계 내에서 정의됩니다. 단계는 이 문서에 나열된 순서대로 실행됩니다.

형식: Dictionary<string,PluginConfiguration>

필수 여부: 예

출력

(스키마 버전 0.3에만 해당) 이 문서를 실행하여 생성되었고 다른 프로세스에서 사용할 수 있는 데이터입니다. 예를 들어 문서에서 새 AMI를 생성하는 경우 출력 값으로 "CreateImage.ImageId"를 지정한 다음, 이 출력을 사용하여 후속 자동화 실행에서 새 인스턴스를 생성할 수 있습니다. 출력에 대한 자세한 내용은 작업 출력을 입력으로 사용 섹션을 참조하세요.

형식: Dictionary<string,OutputConfiguration>

필수 항목 여부: 아니요

files

(스키마 버전 0.3에만 해당) 문서에 첨부되어 자동화 실행 중에 실행되는 스크립트 파일(및 해당 체크섬)입니다. `aws:executeScript` 작업을 포함하고 하나 이상의 단계에서 첨부 파일이 지정된 문서에만 적용됩니다.

스크립트 런타임 지원의 경우 Automation 런북에서는 Python 3.7, Python 3.8, PowerShell Core 6.0 및 PowerShell 7.0용 스크립트가 지원합니다. Automation 런북에 스크립트 포함에 대한 자세한 내용은 런북에서 스크립트 사용 및 문서 빌더를 사용하여 런북 생성 섹션을 참조하세요.

첨부 파일이 포함된 Automation 실행서를 생성할 때 옵션 `--attachments`(AWS CLI용) 또는 `Attachments`(API 및 SDK용)를 사용하여 첨부 파일을 지정해야 합니다. 로컬 파일과 Amazon Simple Storage Service(Amazon S3) 버킷에 저장된 파일 모두에 대해 파일 위치를 지정할 수 있습니다. 자세한 내용은 AWS Systems Manager API 참조의 첨부 파일을 참조하세요.

YAML

```
---
files:
  launch.py:
    checksums:
      sha256: 18871b1311b295c43d0f...
[truncated]...772da97b67e99d84d342ef4aEXAMPLE
```

JSON

```
"files": {
    "launch.py": {
        "checksums": {
            "sha256": "18871b1311b295c43d0f...
[truncated]...772da97b67e99d84d342ef4aEXAMPLE"
        }
```

```
    }
}
```

형식: Dictionary<string,FilesConfiguration>

필수 항목 여부: 아니요

SSM 문서 파라미터 **type** 예제

SSM 문서의 파라미터 유형은 정적입니다. 즉, 파라미터 유형을 정의한 후에는 변경할 수 없습니다. SSM 문서 플러그인과 함께 파라미터를 사용하는 경우 파라미터 유형을 플러그인의 입력 내에서 동적으로 변경할 수 없습니다. 예를 들어 이러한 입력은 문자열이나 문자열 목록을 허용하기 때문에 `aws:runShellScript` 플러그인의 `runCommand` 입력 내에서 `Integer` 파라미터를 참조할 수 없습니다. 플러그인 입력에 파라미터를 사용하려면 파라미터 유형이 허용되는 유형과 일치해야 합니다. 예를 들어 `aws:updateSsmAgent` 플러그인의 `allowDowngrade` 입력에 대해 `Boolean` 유형 파라미터를 지정해야 합니다. 파라미터 유형이 플러그인의 입력 유형과 일치하지 않으면 SSM 문서 검증에 실패하고 시스템에서는 문서를 생성하지 않습니다. 이는 다른 플러그 인 또는 AWS Systems Manager Automation 작업에 대한 입력 내에서 파라미터 다운스트림을 사용할 때도 마찬가지입니다. 예를 들어 `aws:runDocument` 플러그 인의 `documentParameters` 입력 내에서 `StringList` 파라미터를 참조할 수 없습니다. `documentParameters` 입력은 다운스트림 SSM 문서 파라미터 유형이 `StringList` 파라미터이고 참조하는 파라미터와 일치하더라도 문자열 맵을 허용합니다.

Automation 작업에서 파라미터를 사용할 때, 대부분의 경우 SSM 문서 생성 시 파라미터 유형을 검증하지 않습니다. `aws:runCommand` 작업을 사용하는 경우에만 SSM 문서 생성 시 파라미터 유형이 검증됩니다. 다른 모든 경우에는 작업을 실행하기 전에 작업의 입력이 확인되면 자동화 실행 중에 파라미터 검증이 수행됩니다. 예를 들어 입력 파라미터가 `String`이고 이를 `aws:runInstances` 작업의 `MaxInstanceCount` 입력 값으로 참조하면 SSM 문서가 생성됩니다. 그러나 문서 실행 시 `MaxInstanceCount` 입력에 `Integer`가 필요하기 때문에 `aws:runInstances` 작업을 검증하는 중에 자동화가 실패합니다.

다음은 각 파라미터 `type`의 예입니다.

String

인용 부호로 묶인 0개 이상의 유니코드 문자 시퀀스입니다. 예를 들면 "i-1234567890abcdef0"입니다. 이스케이프하려면 백슬래시를 사용합니다.

YAML

```
---
```

```
InstanceId:
  type: String
  description: "(Optional) The target EC2 instance ID."
```

JSON

```
"InstanceId":{
  "type":"String",
  "description":"(Optional) The target EC2 instance ID."
}
```

StringList

쉼표로 구분된 문자열 항목의 목록입니다. 예를 들면 ["cd ~", “pwd”]입니다.

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"
}
```

불

`true` 또는 `false`만 허용됩니다. "true" 또는 0은 허용되지 않습니다.

YAML

```
---
canRun:
  type: Boolean
  description: ''
```

```
  default: true
```

JSON

```
"canRun": {
  "type": "Boolean",
  "description": "",
  "default": true
}
```

Integer

정수입니다. 십진수(예: 3.14159) 또는 인용 부호로 묶인 숫자(예: "3")는 허용되지 않습니다.

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

키-값 매핑입니다. 키와 값은 문자열이어야 합니다. 예를 들면 {"Env”: “Prod”}입니다.

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

SfortMar 객체의 목록입니다.

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
}
```

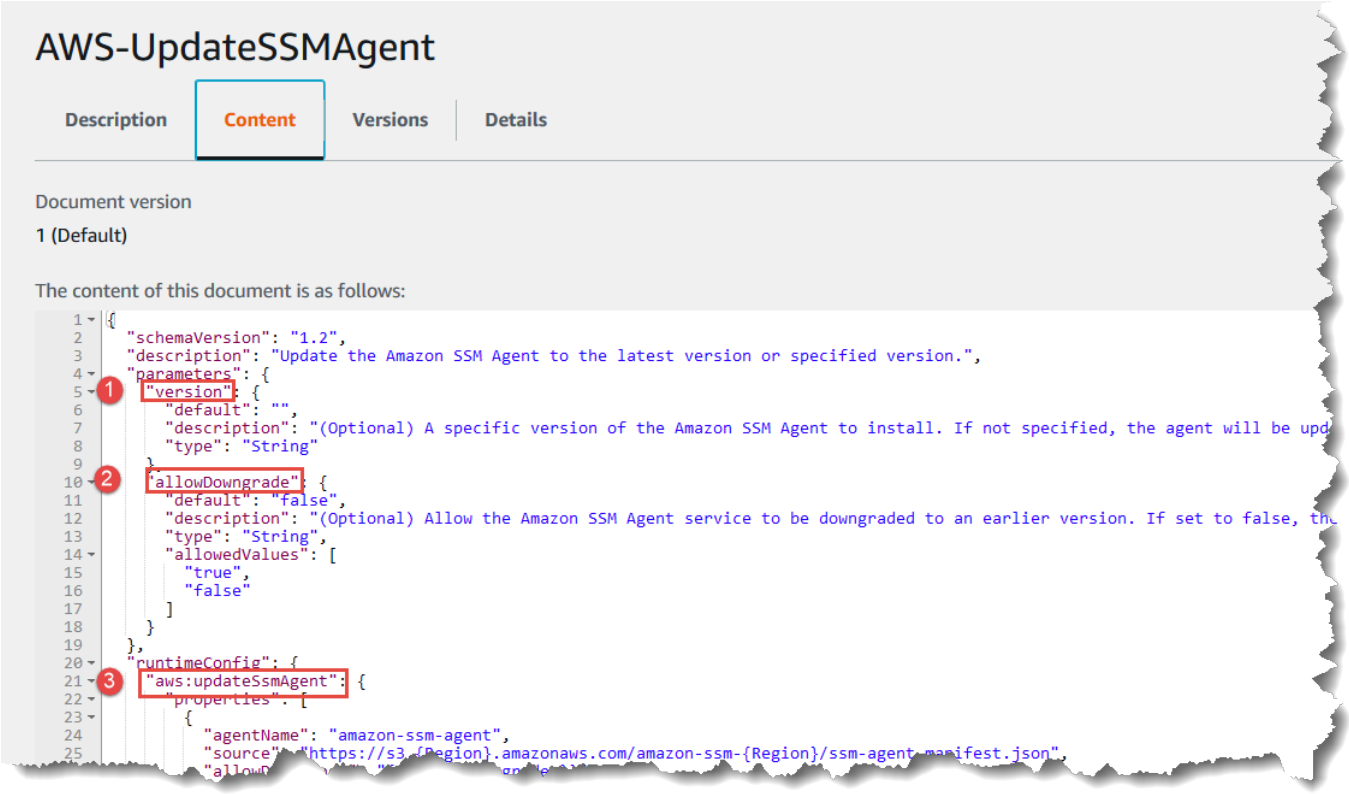
SSM Command 문서 내용 보기

AWS Systems Manager(SSM) 문서에 대한 필수 및 옵션 파라미터를 미리 보기 위해 문서에서 실행하는 작업 외에도 Systems Manager 콘솔에서 문서의 내용을 볼 수 있습니다.

SSM Command 문서 내용을 보려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. 검색 상자에서 [문서 유형(Document type)]을 선택한 다음 [Command]를 선택합니다.
4. 문서의 이름을 선택한 후 [콘텐츠(Content)] 탭을 선택합니다.
5. 콘텐츠 필드에서 문서에 사용 가능한 파라미터와 작업 단계를 검토합니다.

   예를 들어 다음 이미지는 (1) `version` 및 (2) `allowDowngrade`가 `AWS-UpdateSSMAgent` 문서에 대한 옵션 파라미터이고 문서에서 실행되는 첫 번째 작업이 (3) `aws:updateSsmAgent`임을 보여줍니다.



Command 문서 플러그인 참조

이 참조는 AWS Systems Manager(SSM) Command 유형 문서에서 지정할 수 있는 플러그인을 설명합니다. 이러한 플러그인은 Automation 작업을 사용하는 SSM Automation 실행서에서는 사용할 수 없습니다. AWS Systems Manager 자동화 작업에 대한 자세한 내용은 Systems Manager Automation 작업 참조 섹션을 참조하세요.

Systems Manager는 SSM 문서의 콘텐츠를 읽어 관리형 인스턴스에서 수행할 작업을 결정합니다. 각 문서에는 코드 실행 섹션이 포함됩니다. 문서의 스키마 버전에 따라 이 코드 실행 섹션이 하나 이상의 플러그인 또는 단계를 포함할 수 있습니다. 이 도움말 주제의 목적을 위해, 플러그인 및 단계를 플러그인으로 부릅니다. 이 섹션에는 각 Systems Manager 플러그인에 대한 정보가 포함되어 있습니다. 문서 생성 및 스키마 버전 간 차이를 비롯해 문서에 대한 자세한 내용은 AWS Systems Manager Documents 섹션을 참조하세요.

> **Note**
>
> 여기서 설명하는 일부 플러그인은 Windows Server 인스턴스 또는 Linux 인스턴스 중 하나에서만 실행할 수 있습니다. 플러그인마다 플랫폼 종속성이 표시되어 있습니다.

다음 문서 플러그인은 macOS용 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에서 지원됩니다.

- `aws:refreshAssociation`
- `aws:runShellScript`
- `aws:runPowerShellScript`
- `aws:softwareInventory`
- `aws:updateSsmAgent`

내용

공유 입력

(SSM Agent 버전 3.0.502 이상 한정) 모든 플러그 인은 다음 입력을 사용할 수 있습니다.

finallyStep

문서를 실행할 마지막 단계입니다. 이 입력이 단계에 대해 정의된 경우 onFailure 또는 onSuccess 입력에 지정된 exit 값보다 우선합니다. 이 입력이 있는 단계가 예상대로 실행되려면 해당 단계가 문서의 mainSteps에 정의된 마지막 단계여야 합니다.

타입: 부울

유효한 값: true | false

필수 항목 여부: 아니요

onFailure

exit 값이 있는 플러그인에 대해 이 입력을 지정하고 단계가 실패하면 단계 상태는 실패를 반영하고 문서는 finallyStep이 정의되지 않는 한 나머지 단계를 실행하지 않습니다. successAndExit 값이 있는 플러그인에 대해 이 입력을 지정하고 단계가 실패하면 단계 상태는 성공으로 표시되고 문서는 finallyStep이 정의되지 않는 한 나머지 단계를 실행하지 않습니다.

타입: 문자열

유효한 값: exit | successAndExit

필수 항목 여부: 아니요

onSuccess

플러그인에 대해 이 입력을 지정하고 단계가 성공적으로 실행되면 finallyStep이 정의되지 않는 한 문서는 나머지 단계를 실행하지 않습니다.

타입: 문자열

유효값: exit

필수 항목 여부: 아니요

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

EC2 인스턴스에서 애플리케이션을 설치, 복구 또는 제거합니다. 이 플러그인은 Windows Server 운영 체제에서만 실행됩니다.

구문

스키마 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 }}"
      }
    }
  ]
}
```

스키마 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 }}"
            }
         ]
      }
   }
}
```

속성

작업

수행할 작업입니다.

형식: 열거형

유효한 값: `Install` | `Repair` | `Uninstall`

필수 여부: 예

파라미터

설치 프로그램에 대한 파라미터입니다.

타입: 문자열

필수 항목 여부: 아니요

source

애플리케이션용 `.msi` 파일의 URL입니다.

타입: 문자열

필수 항목 여부: 예

sourceHash

`.msi` 파일의 SHA256 해시입니다.

타입: 문자열

필수 항목 여부: 아니요

`aws:cloudWatch`

Windows Server에서 Amazon CloudWatch 또는 Amazon CloudWatch Logs로 데이터를 내보내고 CloudWatch 지표를 사용하여 데이터를 모니터링합니다. 이 플러그인은 Windows Server 운영 체제에서만 실행됩니다. Amazon Elastic Compute Cloud(Amazon EC2)와의 CloudWatch 통합 구성에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 CloudWatch 에이전트를 사용하여 지표, 로그 및 추적 수집을 참조하세요.

> **Important**
>
> 통합 CloudWatch 에이전트가 Amazon CloudWatch Logs에 로그 데이터를 전송하기 위한 도구로 SSM Agent를 대체했습니다. SSM Agent aws:cloudWatch 플러그인은 지원되지 않습니다. 로그 수집 프로세스에 통합된 CloudWatch 에이전트만 사용하는 것이 좋습니다. 자세한 정보는 다음 주제를 참조하세요.
>
> - 통합 CloudWatch Logs로 노드 로그 전송(CloudWatch 에이전트)
> - Windows 서버 노드 로그 수집을 CloudWatch 에이전트로 마이그레이션
> - Amazon CloudWatch 사용 설명서의 CloudWatch 에이전트를 사용하여 지표, 로그 및 추적 수집.

다음의 데이터 형식을 내보내고 모니터링할 수 있습니다.

ApplicationEventLog

CloudWatch Logs로 애플리케이션 이벤트 로그 데이터를 보냅니다.

CustomLogs

Amazon CloudWatch Logs로 텍스트 기반 로그 파일을 보냅니다. CloudWatch 플러그인은 로그 파일의 지문을 생성합니다. 그러면 시스템이 각 지문에 데이터 오프셋을 연결합니다. 플러그인은 변

경 사항이 있을 경우 파일을 업로드하고, 오프셋을 기록하고, 지문에 오프셋을 연결합니다. 이 방법은 사용자가 플러그인을 설정하고, 대량 파일을 포함하는 디렉터리에 서비스를 연결하고, 시스템이 모든 파일을 업로드하는 상황을 방지하기 위해 사용됩니다.

> **Warning**
>
> 애플리케이션이 폴링 도중 로그를 자르거나 정리하려 할 경우 `LogDirectoryPath`에 지정된 모든 로그에서 항목이 손실될 수 있음을 유의하십시오. 예를 들어 로그 파일 크기를 제한하려는 경우, 해당 제한에 도달하면 새 로그 파일을 생성한 후 새 파일에 데이터를 계속 기록합니다.

ETW

CloudWatch Logs로 Windows용 이벤트 추적(ETW) 데이터를 보냅니다.

IIS

CloudWatch Logs로 IIS 로그 데이터를 전송합니다.

PerformanceCounter

CloudWatch로 Windows 성능 카운터를 보냅니다. CloudWatch에 지표로 업로드할 여러 범주를 선택할 수 있습니다. 업로드하려는 각 성능 카운터에 대해 고유한 ID(예: "PerformanceCounter2", "PerformanceCounter3" 등)를 지정하여 PerformanceCounter 섹션을 생성하고 해당 속성을 구성합니다.

> **Note**
>
> AWS Systems Manager SSM Agent 또는 CloudWatch 플러그 인이 중지되고 성능 카운터 데이터가 에 로그되지 않습니다. 이 동작은 사용자 지정 로그나 Windows 이벤트 로그와 다릅니다. 사용자 정의 로그와 Windows 이벤트 로그는 성능 측정 데이터를 보관했다가 SSM Agent 또는 CloudWatch 플러그인이 사용 가능해지면 CloudWatch에 업로드합니다.

SecurityEventLog

CloudWatch Logs로 보안 이벤트 로그 데이터를 보냅니다.

SystemEventLog

CloudWatch Logs로 시스템 이벤트 로그 데이터를 전송합니다.

데이터에 다음 대상을 정의할 수 있습니다.

CloudWatch

성능 카운터 지표 데이터가 보내지는 대상입니다. 고유한 ID(예: "CloudWatch2", CloudWatch3" 등)를 갖는 섹션을 추가하고 새 ID마다 다른 리전을 지정하여 동일한 데이터를 여러 위치로 보낼 수 있습니다.

CloudWatchLogs

로그 데이터가 보내지는 대상입니다. 고유한 ID(예: "CloudWatchLogs2", CloudWatchLogs3" 등)를 갖는 섹션을 추가하고 새 ID마다 다른 리전을 지정하여 동일한 데이터를 여러 위치로 보낼 수 있습니다.

구문

```
"runtimeConfig":{
        "aws:cloudWatch":{
            "settings":{
                "startType":"{{ status }}"
            },
            "properties":"{{ properties }}"
        }
    }
```

설정 및 속성

AccessKey

사용자의 액세스 키 ID입니다. 이 속성은 IAM 역할을 사용해 인스턴스를 시작하지 않은 한 필수입니다. 이 속성은 SSM과 함께 사용할 수 없습니다.

타입: 문자열

필수 항목 여부: 아니요

CategoryName

Performance Monitor의 성능 카운터 범주입니다.

타입: 문자열

필수 항목 여부: 예

CounterName

Performance Monitor의 성능 카운터 이름입니다.

타입: 문자열

필수 항목 여부: 예

CultureName

타임스탬프가 기록되는 로캘입니다. CultureName이 공백이면 기본적으로 Windows Server 인스턴스에서 사용 중인 것과 동일한 로캘로 설정됩니다.

타입: 문자열

유효한 값: 지원되는 값 목록은 Microsoft 웹 사이트에서 National Language Support(NLS)를 참조하십시오. div, div-MV, hu, hu-HU 값은 지원되지 않습니다.

필수 항목 여부: 아니요

DimensionName

Amazon CloudWatch 지표의 차원입니다. `DimensionName`를 지정할 경우 `DimensionValue`을 지정해야 합니다. 이런 파라미터는 메트릭 나열 시 또 다른 보기를 제공합니다. 특정 차원에 속하는 모든 지표를 볼 수 있도록 여러 지표에서 같은 차원을 사용할 수 있습니다.

타입: 문자열

필수 항목 여부: 아니요

DimensionValue

Amazon CloudWatch 지표의 차원 값입니다.

타입: 문자열

필수 항목 여부: 아니요

인코딩

사용할 파일 인코딩입니다(예: UTF-8). 표시 이름이 아니라 인코딩 이름을 사용하세요.

타입: 문자열

유효한 값: 지원되는 값 목록은 Microsoft Learn Library의 Encoding Class를 참조하세요.

필수 여부: 예

필터

로그 이름의 접두사입니다. 모든 파일을 모니터링하려면 이 파라미터를 공백으로 둡니다.

타입: 문자열

유효한 값: 지원되는 값 목록은 MSDN 라이브러리에서 FileSystemWatcherFilter Property 항목을 참조하십시오.

필수 항목 여부: 아니요

흐름

데이터의 대상(CloudWatch 또는 CloudWatch Logs)과 함께 업로드할 각 데이터 형식입니다. 예를 들어 `"Id": "PerformanceCounter"`에서 정의된 성능 카운터를 `"Id": "CloudWatch"`에서 정의된 CloudWatch 대상으로 보내려면 "PerformanceCounter,CloudWatch"를 입력합니다. 마찬가지로, 사용자 정의 로그, ETW 로그 및 시스템 로그를 `"Id": "ETW"`에서 정의된 CloudWatch Logs 대상으로 보내려면 "(ETW),CloudWatchLogs"를 입력합니다. 또한, 같은 성능 카운터 또는 로그 파일을 두 개 이상의 대상으로 보낼 수 있습니다. 예를 들어, 애플리케이션 로그를 `"Id": "CloudWatchLogs"` 및 `"Id": "CloudWatchLogs2"`에서 정의된 두 개의 대상으로 보내려면 "ApplicationEventLog,(CloudWatchLogs, CloudWatchLogs2)"를 입력합니다.

타입: 문자열

유효한 값(원본): `ApplicationEventLog` | `CustomLogs` | `ETW` | `PerformanceCounter` | `SystemEventLog` | `SecurityEventLog`

유효한 값(대상): `CloudWatch` | `CloudWatchLogs` | `CloudWatch`*n* | `CloudWatchLogs`*n*

필수 여부: 예

FullName

구성 요소의 전체 이름입니다.

타입: 문자열

필수 항목 여부: 예

Id

데이터 원본 또는 대상을 식별합니다. 이 식별자는 구성 파일 내에서 고유해야 합니다.

타입: 문자열

필수 항목 여부: 예

InstanceName

성능 카운터 인스턴스의 이름입니다. 각 성능 카운터 구성 요소가 메트릭을 하나씩만 지원하므로, 모든 인스턴스를 나타내기 위해 별표(*)를 사용하지 않도록 합니다. 하지만 _Total을 사용할 수는 있습니다.

타입: 문자열

필수 항목 여부: 예

수준

Amazon CloudWatch로 전송할 메시지의 유형입니다.

타입: 문자열

유효 값:

- 1 - 오류 메시지만 업로드됩니다.
- 2 - 경고 메시지만 업로드됩니다.
- 4 - 정보 메시지만 업로드됩니다.

이들 값을 적절히 더하여 두 가지 이상의 메시지 유형을 포함할 수 있습니다. 예를 들어 3은 오류 메시지(1)와 경고 메시지(2)가 포함된다는 의미입니다. 값 7은 오류 메시지(1), 경고 메시지(2) 및 정보 메시지(4)가 포함된다는 의미입니다.

필수 여부: 예

> **Note**
>
> Windows 보안 로그는 수준을 7로 설정해야 합니다.

LineCount

로그 파일을 식별하는 헤더의 행 수입니다. 예를 들어 IIS 로그 파일에 있는 헤더들은 사실상 동일합니다. 3을 입력하면 로그 파일 헤더에서 처음 나오는 세 줄을 읽어 식별하는 식입니다. IIS 로그 파일에서 세 번째 줄은 날짜와 타임스탬프(로그 파일 간에 차이가 있음)입니다.

유형: 정수

필수 항목 여부: 아니요

LogDirectoryPath

CustomLogs의 경우, 로그가 EC2 인스턴스에서 저장되는 경로입니다. IIS 로그의 경우, 개별 사이트에 대해 IIS 로그가 저장되는 폴더입니다(예: C:\\inetpub\\logs\\LogFiles\\W3SVC*n*). IIS 로그에는 W3C 로그 형식만 지원됩니다. IIS, NCSA 및 사용자 정의 형식은 지원되지 않습니다.

타입: 문자열

필수 항목 여부: 예

LogGroup

로그 그룹의 이름. 이 이름은 CloudWatch 콘솔에서 [로그 그룹(Log Groups)] 화면에 표시됩니다.

타입: 문자열

필수 항목 여부: 예

LogName

로그 파일의 이름입니다.

1. 로그 이름을 찾으려면 이벤트 뷰어의 탐색 창에서 [애플리케이션 및 서비스 로그(Applications and Services Logs)]를 선택합니다.
2. 로그 목록에서 업로드하려는 로그(예: Microsoft>Windows>Backup>Operational)를 마우스 오른쪽 버튼으로 클릭한 후 [사용자 정의 뷰 생성(Create Custom View)]을 선택합니다.
3. [사용자 정의 뷰 생성(Create Custom View)] 대화 상자에서 XML 탭을 선택합니다. LogName은 <Select Path=> 태그 안에 있습니다(예: `Microsoft-Windows-Backup`). 이 텍스트를 LogName 파라미터에 복사합니다.

타입: 문자열

유효한 값: `Application` | `Security` | `System` | `Microsoft-Windows-WinINet/Analytic`

필수 여부: 예

LogStream

대상 로그 스트림입니다. 기본값인 {instance_id}를 사용하는 경우 이 인스턴스의 인스턴스 ID가 로그 스트림 이름으로 사용됩니다.

타입: 문자열

유효한 값: `{instance_id}` | `{hostname}` | `{ip_address}` *`<log_stream_name>`*

미리 존재하지 않는 로그 스트림 이름을 입력하면 CloudWatch Logs에서 이 이름을 자동으로 생성합니다. 리터럴 문자열이나 사전 정의된 변수({instance_id}, {hostname}, {ip_address}) 또는 세 변수 모두의 조합을 사용하여 로그 스트림 이름을 정의할 수 있습니다.

이 파라미터에 지정된 로그 스트림 이름은 CloudWatch 콘솔의 로그 그룹 > ***<YourLogStream>***의 스트림(Log Groups > Streams for <YourLogStream>) 화면에 표시됩니다.

필수 여부: 예

MetricName

성능 데이터를 포함할 CloudWatch 지표입니다.

> **Note**
>
> 이름에 특수 문자를 사용할 수 없습니다. 사용하면 측정치 및 연결된 경보가 작동하지 않을 수 있습니다.

타입: 문자열

필수 항목 여부: 예

NameSpace

성능 카운터 데이터가 기록될 지표 네임스페이스입니다.

타입: 문자열

필수 항목 여부: 예

PollInterval

새 성능 카운터 및 로그 데이터가 업로드되기 전에 경과해야 하는 시간(초)입니다.

유형: 정수

유효한 값: 이 값을 5초 이상으로 설정합니다. 15초(00:00:15)를 권장합니다.

필수 여부: 예

리전

로그 데이터를 보내려는 AWS 리전입니다. 로그 데이터를 보내는 다른 리전으로 성능 카운터를 보낼 수 있지만, 이 파라미터를 인스턴스가 실행 중인 것과 같은 리전으로 설정하는 것이 좋습니다.

타입: 문자열

유효한 값: Systems Manager와 CloudWatch Logs에서 모두 지원하는 AWS 리전의 리전 ID입니다(예: `us-east-2`, `eu-west-1`, 및 `ap-southeast-1`). 각 서비스에서 지원하는 AWS 리전 목록은 Amazon Web Services 일반 참조의 Amazon CloudWatch Logs 서비스 엔드포인트 및 Systems Manager 서비스 엔드포포인트를 참조하세요.

필수 여부: 예

SecretKey

보안 액세스 키입니다. 이 속성은 IAM 역할을 사용해 인스턴스를 시작하지 않은 한 필수입니다.

타입: 문자열

필수 항목 여부: 아니요

startType

인스턴스에서 CloudWatch를 설정하거나 해제합니다.

타입: 문자열

유효한 값: `Enabled` | `Disabled`

필수 여부: 예

TimestampFormat

사용하려는 타임스탬프 형식입니다. 지원되는 값 목록은 MSDN 라이브러리에서 Custom Date and Time Format Strings 항목을 참조하십시오.

타입: 문자열

필수 항목 여부: 예

TimeZoneKind

로그의 타임스탬프에 시간대 정보가 포함되어 있지 않을 때 시간대 정보를 제공합니다. 이 파라미터가 공백으로 남겨져 있고 타임스탬프에 시간대 정보가 포함되어 있지 않으면 CloudWatch Logs

가 기본적으로 현지 시간대로 설정됩니다. 타임스탬프에 표준 시간대 정보가 이미 포함되어 있는 경우 이 파라미터는 무시됩니다.

타입: 문자열

유효한 값: `Local` | `UTC`

필수 항목 여부: 아니요

단위

지표의 측정 단위입니다.

타입: 문자열

유효한 값: Seconds | Microseconds | Milliseconds | Bytes | Kilobytes | Megabytes | Gigabytes | Terabytes | Bits | Kilobits | Megabits | Gigabits | Terabits | Percent | Count | Bytes/Second | Kilobytes/Second | Megabytes/Second | Gigabytes/Second | Terabytes/Second | Bits/Second | Kilobits/Second | Megabits/Second | Gigabits/Second | Terabits/Second | Count/Second | None

필수 여부: 예

`aws:configureDocker`

(스키마 버전 2.0 이상) 컨테이너 및 도커에서 사용할 인스턴스를 구성합니다. 이 플러그인은 Linux 및 Windows Server 운영 체제에서 지원됩니다.

구문

스키마 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 }}"
      }
    }
  ]
}
```

입력

작업

수행할 작업의 유형입니다.

형식: 열거형

유효한 값: `Install` | `Uninstall`

필수 여부: 예

aws:configurePackage

(스키마 버전 2.0 이상) AWS Systems Manager Distributor 패키지를 설치하거나 제거합니다. 최신 버전, 기본 버전 또는 지정한 패키지 버전을 설치할 수 있습니다. AWS에서 제공하는 패키지도 지원됩니다. 이 플러그인은 Windows Server 및 Linux 운영 체제에서 실행해야 하지만 사용 가능한 일부 패키지는 Linux 운영 체제에서 지원되지 않습니다.

Windows Server에 사용할 수 있는 AWS 패키지는 `AWSPVDriver`, `AWSNVMe`, `AwsEnaNetworkDriver`, `AwsVssComponents`, `AmazonCloudWatchAgent`, `CodeDeployAgent` 및 `AWSSupport-EC2Rescue.`입니다.

Linux 운영 체제에 사용할 수 있는 AWS 패키지는 `AmazonCloudWatchAgent`, `CodeDeployAgent` 및 `AWSSupport-EC2Rescue`입니다.

구문

스키마 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\"}"
          }
       }
    ]
}
```

입력

이름

설치 또는 제거할 AWS 패키지의 이름입니다. AWSPVDriver, AwsEnaNetworkDriver, AwsVssComponents, AmazonCloudWatchAgent 등의 패키지를 사용할 수 있습니다.

타입: 문자열

필수 항목 여부: 예

작업

패키지를 설치 또는 제거합니다.

형식: 열거형

유효한 값: Install | Uninstall

필수 여부: 예

installationType

수행할 설치 유형입니다. Uninstall and reinstall을 지정하면 패키지가 완전히 제거되었다가 다시 설치됩니다. 재설치가 완료될 때까지 애플리케이션을 사용할 수 없습니다. In-place update를 지정하면 업데이트 스크립트에서 제공한 지침에 따라 새 파일이나 변경된 파일만 기존 설치에 추가됩니다. 애플리케이션은 업데이트 프로세스 전체에서 사용할 수 있습니다. AWS 게시된 패키지에는 In-place update 옵션이 지원되지 않습니다. Uninstall and reinstall이 기본값입니다.

형식: 열거형

유효한 값: Uninstall and reinstall | In-place update

필수 항목 여부: 아니요

additionalArguments

스크립트 설치, 제거 또는 업데이트에 제공되는 추가 파라미터의 JSON 문자열입니다. 각 파라미터에는 SSM_가 접두사로 붙어야 합니다. 규칙 {{ssm:*parameter-name*}}을 사용하여 추가 인수에서 Parameter Store 파라미터를 참조할 수 있습니다. 설치, 제거 또는 업데이트 스크립트에서 추가 파라미터를 사용하려면 운영 체제에 적합한 구문을 사용하여 파라미터를 환경 변수로 참조해야 합니다. 예를 들어 PowerShell에서 SSM_arg 인수를 $Env:SSM_arg로 참조합니다. 정의하는 인수 수에는 제한이 없지만 추가 인수 입력에는 4,096자 제한이 있습니다. 이 제한에는 정의한 모든 키와 값이 포함됩니다.

유형: StringMap

필수 항목 여부: 아니요

version

설치 또는 제거할 패키지의 특정 버전입니다. 설치하는 경우 기본적으로 시스템이 최근에 게시된 버전을 설치합니다. 제거하는 경우 기본적으로 시스템이 현재 설치된 버전을 제거합니다. 설치된 버전이 발견되지 않으면 최근에 게시된 버전이 다운로드되고 제거 작업이 실행됩니다.

타입: 문자열

필수 항목 여부: 아니요

aws:domainJoin

도메인에 EC2 인스턴스를 조인합니다. 이 플러그인은 Linux 및 Windows Server 운영 체제에서 실행됩니다. 이 플러그인은 Linux 인스턴스의 호스트 이름을 EC2AMAZ-*XXXXXXX* 형식으로 변경합니다. EC2 인스턴스 조인에 대한 자세한 내용은 AWS Directory Service Administration Guide의 Join an EC2 Instance to Your AWS Managed Microsoft AD Directory를 참조하세요.

구문

스키마 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
mainSteps:
- action: aws:domainJoin
  name: domainJoin
  inputs:
    directoryId: "{{ directoryId }}"
    directoryName: "{{ directoryName }}"
    directoryOU: "{{ directoryOU }}"
    dnsIpAddresses: "{{ dnsIpAddresses }}"
```

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"
    },
  },
  "mainSteps": [
    {
      "action": "aws:domainJoin",
      "name": "domainJoin",
      "inputs": {
        "directoryId": "{{ directoryId }}",
        "directoryName": "{{ directoryName }}",
        "directoryOU":"{{ directoryOU }}",
        "dnsIpAddresses":"{{ dnsIpAddresses }}"
      }
    }
  ]
}
```

스키마 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 }}"
```

```
            }
        }
    }
}
```

속성

directoryId

디렉터리의 ID입니다.

타입: 문자열

필수 항목 여부: 예

예: "directoryId": "d-1234567890"

directoryName

도메인 이름.

타입: 문자열

필수 항목 여부: 예

예: "directoryName": "example.com"

directoryOU

조직 단위(OU)입니다.

타입: 문자열

필수 항목 여부: 아니요

예: "directoryOU": "OU=test,DC=example,DC=com"

dnsIpAddresses

DNS 서버의 IP 주소입니다.

유형: StringList

필수 여부: 예

예: "dnsIpAddresses": ["198.51.100.1","198.51.100.2"]

예제

예시는 AWS Directory Service 관리 안내서의 AWS Managed Microsoft AD에 Amazon EC2 인스턴스 조인을 참조하세요.

aws:downloadContent

(스키마 버전 2.0 이상) 원격 위치에서 SSM 문서 및 스크립트를 다운로드합니다. GitHub Enterprise 리포지토리는 지원되지 않습니다. 이 플러그인은 Linux 및 Windows Server 운영 체제에서 지원됩니다.

구문

스키마 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 }}"
      }
    }
  ]
}
```

입력

sourceType

다운로드 소스입니다. Systems Manager는 스크립트 및 SSM 문서 다운로드를 위해 `GitHub`, `Git`, `HTTP`, `S3` 및 `SSMDocument` 소스 유형을 지원합니다.

타입: 문자열

필수 항목 여부: 예

sourceInfo

필요한 원본에서 콘텐츠를 검색하는 데 필요한 정보입니다.

유형: StringMap

필수 여부: 예

sourceType **GitHub,**에 대해 다음을 지정합니다.

- owner: 리포지토리 소유자입니다.
- repository: 리포지토리의 이름입니다.
- path: 다운로드할 파일 또는 디렉터리에 대한 경로입니다.
- getOptions: 마스터가 아닌 분기 또는 리포지토리의 특정 커밋에서 내용을 검색하는 추가 옵션입니다. 마스터 분기에서 최신 커밋을 사용하는 경우 getOptions를 생략할 수 있습니다. 리포지토리가 2020년 10월 1일 이후에 생성된 경우 기본 분기의 이름은 master 대신 main이 될 수 있습니다. 이 경우 getOptions 파라미터에 대한 값을 지정해야 합니다.

  이 파라미터는 다음 형식을 사용합니다.

  - branch:refs/heads/*branch_name*

    기본값은 `master`입니다.

    기본이 아닌 브랜치를 지정하려면 다음 형식을 사용합니다.

    branch:refs/heads/*branch_name*

  - commitID:*commitID*

    기본값은 `head`입니다.

    최신 버전이 아닌 커밋에서 SSM 문서의 버전을 사용하려면 전체 커밋 ID를 지정합니다. 예:

    ```
    "getOptions": "commitID:bbc1ddb94...b76d3bEXAMPLE",
    ```

- tokenInfo: GitHub 액세스 토큰 정보를 {{ssm-secure:*secure-string-token-name*}} 형식으로 저장하는 Systems Manager 파라미터(SecureString 파라미터)입니다.

  > **Note**
  >
  > 이 `tokenInfo` 필드는 SecureString 파라미터를 지원하는 유일한 SSM 문서 플러그인 필드입니다. SecureString 파라미터는 다른 SSM 문서 플러그인이나 다른 필드에서 지원되지 않습니다.

```
{
    "owner":"TestUser",
    "repository":"GitHubTest",
    "path":"scripts/python/test-script",
```

```
    "getOptions":"branch:master",
    "tokenInfo":"{{ssm-secure:secure-string-token}}"
}
```

sourceType **Git**에 대해 다음을 지정해야 합니다.

- 리포지토리

  다운로드할 파일 또는 디렉터리의 Git 리포지토리 URL입니다.

  타입: 문자열

또한 다음과 같은 파라미터(옵션)를 지정할 수 있습니다.

- getOptions

  마스터가 아닌 분기 또는 리포지토리의 특정 커밋에서 내용을 검색하는 추가 옵션입니다. 마스터 분기에서 최신 커밋을 사용하는 경우 getOptions를 생략할 수 있습니다.

  타입: 문자열

  이 파라미터는 다음 형식을 사용합니다.

  - branch:refs/heads/*branch_name*

    기본값은 `master`입니다.

    `"branch"`는 SSM 문서가 `master`가 아닌 분기에 저장된 경우에만 필요합니다. 예:

    ```
    "getOptions": "branch:refs/head/main"
    ```

  - commitID:*commitID*

    기본값은 `head`입니다.

    최신 버전이 아닌 커밋에서 SSM 문서의 버전을 사용하려면 전체 커밋 ID를 지정합니다. 예:

    ```
    "getOptions": "commitID:bbc1ddb94...b76d3bEXAMPLE",
    ```

- privateSSHKey

  지정하는 `repository`에 연결할 때 사용할 SSH 키입니다. `{{ssm-secure:`*your-secure-string-parameter*`}}` 형식을 사용하여 SSH 키 값에 대한 `SecureString` 파라미터를 참조할 수 있습니다.

타입: 문자열

- 건너뛰기 키 검사

  지정한 `repository`에 연결할 때 StrictHostKeyChecking 옵션의 값을 결정합니다. 기본 값은 `false`입니다.

  타입: 부울

- 사용자 이름

  HTTP를 사용하여 지정한 `repository`에 연결할 때 사용할 사용자 이름입니다. `{{ssm-secure:`*`your-secure-string-parameter`*`}}` 형식을 사용하여 사용자 이름 값에 대한 SecureString 파라미터를 참조할 수 있습니다.

  타입: 문자열

- 비밀번호

  HTTP를 사용하여 지정한 `repository`에 연결할 때 사용할 암호입니다. `{{ssm-secure:`*`your-secure-string-parameter`*`}}` 형식을 사용하여 암호 값에 대한 SecureString 파라미터를 참조할 수 있습니다.

  타입: 문자열

sourceType **HTTP**에 대해 다음을 지정해야 합니다.

- url

  다운로드할 파일 또는 디렉터리의 URL입니다.

  타입: 문자열

또한 다음과 같은 파라미터(옵션)를 지정할 수 있습니다.

- allowInsecureDownload

  보안 소켓 계층(SSL) 또는 전송 계층 보안(TLS)으로 암호화되지 않은 연결을 통해 다운로드를 수행할 수 있는지 여부를 결정합니다. 기본 값은 `false`입니다. 암호화 없이 다운로드를 수행하지 않는 것이 좋습니다. 그렇게 하기로 선택하는 경우 모든 관련 위험을 감수해야 합니다. 보안은 AWS와 사용자의 공동 책임입니다. 이것을 공동 책임 모델이라고 합니다. 자세한 내용은 공동 책임 모델을 참조하세요.

  타입: 부울

- authMethod

  지정한 `url`에 연결할 때 사용자 이름과 암호를 인증에 사용할지 여부를 결정합니다. `Basic` 또는 `Digest`를 지정하는 경우 `username` 및 `password` 파라미터에 대한 값을 제공해야 합니다. `Digest` 메서드를 사용하려면 인스턴스에 SSM Agent 버전 3.0.1181.0 이상이 설치되어 있어야 합니다. `Digest` 메서드는 MD5 및 SHA256 암호화를 지원합니다.

  타입: 문자열

  유효한 값: `None` | `Basic` | `Digest`

- 사용자 이름

  `Basic` 인증을 사용하여 지정한 `url`에 연결할 때 사용할 사용자 이름입니다. `{{ssm-secure:`*`your-secure-string-parameter`*`}}` 형식을 사용하여 사용자 이름 값에 대한 `SecureString` 파라미터를 참조할 수 있습니다.

  타입: 문자열

- 비밀번호

  `Basic` 인증을 사용하여 지정한 `url`에 연결할 때 사용할 암호입니다. `{{ssm-secure:`*`your-secure-string-parameter`*`}}` 형식을 사용하여 암호 값에 대한 `SecureString` 파라미터를 참조할 수 있습니다.

  타입: 문자열

sourceType **`S3`**에 대해 다음을 지정합니다.

- path: Amazon S3에서 다운로드할 파일 또는 디렉터리의 URL입니다.

```
{
    "path": "https://s3.amazonaws.com/DOC-EXAMPLE-BUCKET/powershell/
helloPowershell.ps1"
}
```

sourceType **`SSMDocument`**에 대해 다음 중 하나를 지정합니다.

- name: 문서의 이름 및 버전입니다. 형식: `name:version`. 버전은 선택 항목입니다.

```
{
    "name": "Example-RunPowerShellScript:3"
}
```

- name: 문서에 대한 ARN입니다 (`arn:aws:ssm:region:account_id:document/document_name` 형식).

```
{
   "name":"arn:aws:ssm:us-east-2:3344556677:document/MySharedDoc"
}
```

destinationPath

파일을 다운로드하고자 할 때 선택할 수 있는 인스턴스의 로컬 경로 옵션입니다. 경로를 지정하지 않은 경우에는 명령 ID와 관련된 경로에 콘텐츠가 다운로드됩니다.

타입: 문자열

필수 항목 여부: 아니요

aws:psModule

Amazon EC2 인스턴스에 PowerShell 모듈을 설치합니다. 이 플러그인은 Windows Server 운영 체제에서만 실행됩니다.

구문

스키마 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 }}"
      }
    }
  ]
}
```

스키마 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 }}"
            }
         ]
      }
   }
}
```

속성

runCommand

모듈 설치 후 실행될 PowerShell 명령입니다.

유형: StringList

필수 항목 여부: 아니요

source

애플리케이션 `.zip` 파일에 대한 URL 또는 인스턴스 상 로컬 경로입니다.

타입: 문자열

필수 항목 여부: 예

sourceHash

`.zip` 파일의 SHA256 해시입니다.

타입: 문자열

필수 항목 여부: 아니요

timeoutSeconds

명령이 실패로 간주되기 전에 완료되어야 할 시간(초)입니다.

타입: 문자열

필수 항목 여부: 아니요

workingDirectory

인스턴스 상의 작업 디렉터리에 대한 경로.

타입: 문자열

필수 항목 여부: 아니요

aws:refreshAssociation

(스키마 버전 2.0 이상) 온디맨드로 연결을 새로 고칩니다(강제 적용). 이 작업은 대상에 바운딩된 선택된 연결 또는 모든 연결에서 정의된 대로 시스템 상태를 변경합니다. 이 플러그인은 Linux 및 Microsoft Windows Server 운영 체제에서 실행됩니다.

구문

스키마 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 }}"
        ]
      }
    }
  ]
}
```

입력

associationIds

연결 ID의 목록입니다. 비어 있을 경우, 지정된 대상에 바인딩된 모든 연결이 적용됩니다.

유형: StringList

필수 항목 여부: 아니요

aws:runDockerAction

(스키마 버전 2.0 이상) 컨테이너에서 도커 작업을 실행합니다. 이 플러그인은 Linux 및 Microsoft Windows Server 운영 체제에서 실행됩니다.

구문

스키마 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 }}"
```

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 }}"
         }
      }
   ]
}
```

입력

작업

수행할 작업의 유형입니다.

타입: 문자열

필수 항목 여부: 예

컨테이너

도커 컨테이너 ID입니다.

타입: 문자열

필수 항목 여부: 아니요

image

도커 이미지 이름입니다.

타입: 문자열

필수 항목 여부: 아니요

cmd

컨테이너 명령입니다.

타입: 문자열

필수 항목 여부: 아니요

메모리

컨테이너 메모리 제한입니다.

타입: 문자열

필수 항목 여부: 아니요

cpuShares

컨테이너 CPU 공유입니다(상대 가중치).

타입: 문자열

필수 항목 여부: 아니요

볼륨

컨테이너 볼륨 마운트입니다.

유형: StringList

필수 항목 여부: 아니요

env

컨테이너 환경 변수입니다.

타입: 문자열

필수 항목 여부: 아니요

사용자

컨테이너 사용자 이름입니다.

타입: 문자열

Required: No

publish

컨테이너 게시 포트입니다.

타입: 문자열

필수 항목 여부: 아니요

aws:runDocument

(스키마 버전 2.0 이상) Systems Manager 또는 로컬 공유에 저장된 SSM 문서를 실행합니다. 이 플러그인을 aws:downloadContent 플러그인과 함께 사용하면 원격 위치에서 로컬 공유로 SSM 문서를 다운로드하여 실행할 수 있습니다. 이 플러그인은 Linux 및 Windows Server 운영 체제에서 지원됩니다. 이 플러그인은 aws:updateSsmAgent 플러그 인을 사용하는 AWS-UpdateSSMAgent 문서 또는 다른 문서 실행을 지원하지 않습니다.

구문

스키마 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 }}"
      }
    }
  ]
}
```

입력

documentType

실행할 문서 유형입니다. 로컬 문서(`LocalPath`) 또는 Systems Manager에 저장된 문서(`SSMDocument`)를 실행할 수 있습니다.

타입: 문자열

필수 항목 여부: 예

documentPath

문서에 대한 경로입니다. documentType이 LocalPath일 경우, 로컬 공유에 저장된 문서에 대한 경로를 지정합니다. documentType이 SSMDocument일 경우, 문서의 이름을 지정합니다.

타입: 문자열

필수 항목 여부: 아니요

documentParameters

문서에 대한 파라미터입니다.

유형: StringMap

필수 항목 여부: 아니요

aws:runPowerShellScript

PowerShell 스크립트를 실행하거나 실행할 스크립트에 대한 경로를 지정합니다. 이 플러그인은 Microsoft Windows Server 및 Linux 운영 체제에서 실행됩니다.

구문

스키마 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 }}"
        ]
      }
    }
  ]
}
```

스키마 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 }}"
            }
         ]
      }
   }
}
```

속성

runCommand

실행할 명령 또는 인스턴스의 기존 스크립트에 대한 경로를 지정합니다.

유형: StringList

필수 여부: 예

timeoutSeconds

명령이 실패로 간주되기 전에 완료되어야 할 시간(초)입니다. 시간 제한에 도달하면 Systems Manager가 명령 실행을 멈춥니다.

타입: 문자열

필수 항목 여부: 아니요

workingDirectory

인스턴스 상의 작업 디렉터리에 대한 경로.

타입: 문자열

필수 항목 여부: 아니요

aws:runShellScript

Linux 셸 스크립트를 실행하거나 실행할 스크립트에 대한 경로를 지정합니다. 이 플러그인은 Linux 운영 체제에서만 실행됩니다.

구문

스키마 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 }}"
        ]
      }
    }
  ]
}
```

스키마 1.2

YAML

```
---
runtimeConfig:
  aws:runShellScript:
    properties:
    - runCommand: "{{ commands }}"
      workingDirectory: "{{ workingDirectory }}"
      timeoutSeconds: "{{ executionTimeout }}"
```

JSON

```
{
   "runtimeConfig":{
      "aws:runShellScript":{
         "properties":[
            {
               "runCommand":"{{ commands }}",
               "workingDirectory":"{{ workingDirectory }}",
               "timeoutSeconds":"{{ executionTimeout }}"
            }
         ]
      }
   }
```

```
}
```

속성

runCommand

실행할 명령 또는 인스턴스의 기존 스크립트에 대한 경로를 지정합니다.

유형: StringList

필수 여부: 예

timeoutSeconds

명령이 실패로 간주되기 전에 완료되어야 할 시간(초)입니다. 시간 제한에 도달하면 Systems Manager가 명령 실행을 멈춥니다.

타입: 문자열

필수 항목 여부: 아니요

workingDirectory

인스턴스 상의 작업 디렉터리에 대한 경로.

타입: 문자열

필수 항목 여부: 아니요

`aws:softwareInventory`

(스키마 버전 2.0 이상) 관리형 인스턴스의 애플리케이션, 파일 및 구성에 대한 메타데이터를 수집합니다. 이 플러그인은 Linux 및 Microsoft Windows Server 운영 체제에서 실행됩니다. 인벤토리 수집을 구성할 때 AWS Systems Manager State Manager 연결을 생성하는 것으로 시작합니다. Systems Manager는 연결이 실행될 때 인벤토리 데이터를 수집합니다. 연결을 먼저 생성하지 않고 `aws:softwareInventory` 플러그인을 호출하려고 하면 시스템이 다음 오류를 반환합니다.

```
The aws:softwareInventory plugin can only be invoked via ssm-associate.
```

인스턴스에는 한 번에 하나의 인벤토리 연결만 구성할 수 있습니다. 둘 이상의 연결로 인스턴스를 구성할 경우 인벤토리 연결이 실행되지 않으며 인벤토리 데이터가 수집되지 않습니다. 인벤토리 수집에 대한 자세한 내용은 AWS Systems Manager Inventory 섹션을 참조하세요.

구문

스키마 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 }}"
         }
      }
   ]
```

```
}
```

입력

애플리케이션

(선택 사항) 설치한 애플리케이션에 대한 메타데이터를 수집합니다.

타입: 문자열

필수 항목 여부: 아니요

awsComponents

(옵션) amazon-ssm-agent 같은 AWS 구성 요소의 메타데이터를 수집합니다.

타입: 문자열

필수 항목 여부: 아니요

files

(옵션, SSM Agent 버전 2.2.64.0 이상 필요) 몇 가지 예를 들자면 파일 이름, 파일 생성 시각, 파일을 마지막으로 수정하고 액세스한 시간, 파일 크기 등 파일에 관한 메타데이터 정보를 수집합니다. 파일 인벤토리 수집에 대한 자세한 내용은 파일 및 Windows 레지스트리 인벤토리 관련 작업 섹션을 참조하세요.

타입: 문자열

필수 항목 여부: 아니요

networkConfig

(선택 사항) 네트워크 구성에 대한 메타데이터를 수집합니다.

타입: 문자열

필수 항목 여부: 아니요

windowsUpdates

(선택 사항) 모든 Windows 업데이트에 대한 메타데이터를 수집합니다.

타입: 문자열

필수 항목 여부: 아니요

InstanceDetailedInformation

(옵션) CPU 모델, 속도, 코어 수를 포함하여 기본 인벤토리 플러그인 (`aws:instanceInformation`)에서 제공하는 것 이외의 인스턴스 정보를 수집합니다.

타입: 문자열

필수 항목 여부: 아니요

서비스

(옵션, Windows OS에만 해당, SSM Agent 버전 2.2.64.0 이상 필요) 서비스 구성에 대한 메타데이터를 수집합니다.

타입: 문자열

필수 항목 여부: 아니요

windowsRegistry

(옵션, Windows OS에만 해당, SSM Agent 버전 2.2.64.0 이상 필요) Windows 레지스트리 키 및 값을 수집합니다. 키 경로를 선택하고 모든 키와 값을 반복적으로 수집할 수 있습니다. 특정 경로에 대해 특정 레지스트리 키와 그 값을 수집할 수도 있습니다. 인벤토리는 키 경로, 이름 및 값을 수집합니다. Windows 레지스트리 인벤토리를 수집하는 방법에 대한 자세한 내용은 파일 및 Windows 레지스트리 인벤토리 관련 작업 섹션을 참조하세요.

타입: 문자열

필수 항목 여부: 아니요

windowsRoles

(옵션, Windows OS에만 해당, SSM Agent 버전 2.2.64.0 이상 필요) Microsoft Windows 역할 구성에 대한 메타데이터를 수집합니다.

타입: 문자열

필수 항목 여부: 아니요

customInventory

(선택 사항) 사용자 지정 인벤토리 데이터를 수집합니다. 사용자 정의 인벤토리에 대한 자세한 내용은 사용자 정의 인벤토리 작업 섹션을 참조하세요.

타입: 문자열

필수 항목 여부: 아니요

aws:updateAgent

EC2Config 서비스를 최신 버전으로 업데이트하거나 이전 버전을 지정합니다. 이 플러그인은 Microsoft Windows Server 운영 체제에서만 실행됩니다. EC2Config 서비스에 대한 자세한 내용은 Amazon EC2 사용 설명서의 EC2Config 서비스(레거시)를 사용하여 Windows 인스턴스 구성을 참조하세요.

구문

스키마 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"
      }
    }
  ]
}
```

스키마 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 }}"
         }
      }
   }
}
```

속성

agentName

EC2Config. 이것은 EC2Config 서비스를 실행하는 에이전트의 이름입니다.

타입: 문자열

필수 항목 여부: 예

allowDowngrade

EC2Config 서비스를 이전 버전으로 다운그레이드할 수 있게 허용합니다. false로 설정할 경우, 서비스가 새 버전으로 업그레이드만 될 수 있습니다(기본값). true로 설정하는 경우, 이전 버전을 지정합니다.

타입: 부울

필수 항목 여부: 아니요

source

Systems Manager가 설치할 EC2Config의 버전을 복사하는 위치입니다. 이 위치는 변경할 수 없습니다.

타입: 문자열

필수 항목 여부: 예

targetVersion

설치할 EC2Config 서비스의 특정 버전입니다. 버전을 지정하지 않으면 서비스는 최신 버전으로 업데이트됩니다.

타입: 문자열

필수 항목 여부: 아니요

aws:updateSsmAgent

SSM Agent를 최신 버전으로 업데이트하거나 이전 버전을 지정합니다. 이 플러그인은 Linux 및 Windows Server 운영 체제에서 실행됩니다. 자세한 내용은 SSM Agent 작업 단원을 참조하십시오.

구문

스키마 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 }}"
      }
    }
  ]
}
```

스키마 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 }}"
            }
         ]
      }
   }
}
```

속성

agentName

amazon-ssm-agent. 요청을 처리하고 인스턴스에서 명령을 실행하는 Systems Manager Agent의 이름입니다.

타입: 문자열

필수 항목 여부: 예

allowDowngrade

SSM Agent를 이전 버전으로 다운그레이드할 수 있게 허용합니다. false로 설정할 경우, 에이전트가 새 버전으로 업그레이드만 될 수 있습니다(기본값). true로 설정하는 경우, 이전 버전을 지정합니다.

타입: 부울

필수 여부: 예

source

Systems Manager가 설치할 SSM Agent 버전을 복사하는 위치입니다. 이 위치는 변경할 수 없습니다.

타입: 문자열

필수 항목 여부: 예

targetVersion

설치할 SSM Agent의 특정 버전입니다. 버전을 지정하지 않으면 에이전트는 최신 버전으로 업데이트됩니다.

타입: 문자열

필수 항목 여부: 아니요

SSM 문서 콘텐츠 생성

AWS Systems Manager 퍼블릭 문서가 AWS 리소스에서 수행하려는 모든 작업을 수행하지 않는 경우 자체 SSM 문서를 생성할 수 있습니다. 콘솔을 사용하여 SSM 문서를 복제할 수도 있습니다. 문서를 복제하면 기존 문서의 내용이 수정 가능한 새 문서로 복사됩니다. 문서를 생성하거나 복제할 경우, 문서

의 콘텐츠는 64KB를 초과할 수 없습니다. 이 할당량에는 런타임 시 입력 파라미터에 지정된 콘텐츠도 포함됩니다. 새 `Policy` 또는 `Command` 문서를 생성할 때는 문서 편집, 자동 버전 관리, 시퀀싱 등과 같은 최신 기능을 활용할 수 있도록 스키마 버전 2.2 이상을 사용하는 것이 좋습니다.

SSM 문서 콘텐츠 작성

사용자 고유의 SSM 문서 콘텐츠를 생성하려면 SSM 문서에 사용할 수 있는 다양한 스키마, 기능, 플러그인 및 구문을 이해하는 것이 중요합니다. 다음 리소스를 숙지하는 것이 좋습니다.

- 나만의 AWS Systems Manager 문서 작성
- 데이터 요소 및 파마미터
- 스키마, 특성 및 예제
- Command 문서 플러그인 참조
- Systems Manager Automation 작업 참조
- Automation 시스템 변수
- 추가 런북 예제
- AWS Toolkit for Visual Studio Code를 사용하여 Systems Manager Automation 실행서로 작업
- 문서 빌더를 사용하여 런북 생성
- 런북에서 스크립트 사용

사전 정의된 AWS SSM 문서는 필요한 일부 작업을 수행할 수 있습니다. 문서 유형에 따라 사용자 정의 SSM 문서 내에서 `aws:runDocument`, `aws:runCommand` 또는 `aws:executeAutomation` 플러그인을 사용하여 이러한 문서를 호출할 수 있습니다. 이러한 문서의 일부를 사용자 정의 SSM 문서로 복사하고 요구 사항에 맞게 콘텐츠를 편집할 수도 있습니다.

> **Tip**
>
> SSM 문서 콘텐츠를 생성할 때 테스트하는 동안 SSM 문서의 콘텐츠를 변경하고 여러 번 업데이트할 수 있습니다. 다음 명령은 최신 콘텐츠로 SSM 문서를 업데이트하고 문서의 기본 버전을 최신 버전의 문서로 업데이트합니다.
>
> > **Note**
> >
> > Linux 및 Windows 명령은 `jq` 명령줄 도구를 사용하여 JSON 응답 데이터를 필터링합니다.

Linux & macOS

```
latestDocVersion=$(aws ssm update-document \
    --content file://path/to/file/documentContent.json \
    --name "ExampleDocument" \
    --document-format JSON \
    --document-version '$LATEST' \
    | jq -r '.DocumentDescription.LatestVersion')

aws ssm update-document-default-version \
    --name "ExampleDocument" \
    --document-version $latestDocVersion
```

Windows

```
latestDocVersion=$(aws ssm update-document ^
    --content file://C:\path\to\file\documentContent.json ^
    --name "ExampleDocument" ^
    --document-format JSON ^
    --document-version "$LATEST" ^
    | jq -r '.DocumentDescription.LatestVersion')

aws ssm update-document-default-version ^
    --name "ExampleDocument" ^
    --document-version $latestDocVersion
```

PowerShell

```
$content = Get-Content -Path "C:\path\to\file\documentContent.json" | Out-
String
$latestDocVersion = Update-SSMDocument `
    -Content $content `
    -Name "ExampleDocument" `
    -DocumentFormat "JSON" `
    -DocumentVersion '$LATEST' `
    | Select-Object -ExpandProperty LatestVersion

Update-SSMDocumentDefaultVersion `
    -Name "ExampleDocument" `
```

```
    -DocumentVersion $latestDocVersion
```

SSM 문서 복제

Systems Manager Documents 콘솔을 사용하여 AWS Systems Manager 문서를 복제하여 SSM 문서를 생성할 수 있습니다. SSM 문서를 복제하면 기존 문서의 내용이 수정 가능한 새 문서로 복사됩니다. 64KB보다 큰 문서는 복제할 수 없습니다.

SSM 문서를 복제하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. 검색 상자에 복제하려는 문서의 이름을 입력합니다.
4. 복제할 문서의 이름을 선택한 다음 [작업(Actions)] 드롭다운에서 [문서 복제(Clone document)]를 선택합니다.
5. 원하는 대로 문서를 수정한 다음 [문서 생성(Create document)]을 선택하여 문서를 저장합니다.

다음 방법 중 하나를 통해 SSM 문서 콘텐츠를 작성한 후 해당 콘텐츠를 사용하는 SSM 문서를 생성할 수 있습니다.

SSM 문서 생성

- 복합 문서 생성

복합 문서 생성

복합 AWS Systems Manager(SSM) 문서는 하나 이상의 보조 SSM 문서를 실행하여 일련의 작업을 수행하는 사용자 정의 문서입니다. 복합 문서는 부트스트랩 소프트웨어 또는 도메인 조인 인스턴스와 같은 일반적인 태스크에 대한 표준 SSM 문서 집합을 생성할 수 있도록 하여 코드형 인프라를 승격합니다. 그러면 이들 문서를 동일한 AWS 리전에서 AWS 계정 간에 공유하여 SSM 문서 유지 관리를 줄이고 일관성을 보장할 수 있습니다.

예를 들어 다음 작업을 수행하는 복합 문서를 생성할 수 있습니다.

1. 허용 목록에 모든 패치를 설치합니다.
2. 안티바이러스 소프트웨어를 설치
3. GitHub에서 스크립트를 다운로드하고 실행합니다.

이 예에서는 사용자 정의 SSM 문서가 이러한 작업을 수행하기 위해 다음과 같은 플러그인을 포함합니다.

1. `aws:runDocument` 플러그인은 나열된 모든 허용 패치를 설치하는 `AWS-RunPatchBaseline` 문서를 실행합니다.
2. 안티바이러스 소프트웨어를 설치하는 `AWS-InstallApplication` 문서를 실행하기 위한 `aws:runDocument` 플러그인.
3. `aws:downloadContent` 플러그인은 GitHub에서 스크립트를 다운로드하고 실행합니다.

복합 및 보조 문서는 Systems Manager, GitHub(퍼블릭 및 프라이빗 리포지토리) 또는 Amazon S3에 저장할 수 있습니다. 복합 문서 및 보조 문서는 JSON 또는 YAML으로 생성할 수 있습니다.

> **Note**
>
> 복합 문서는 최대 3개 문서 깊이까지만 실행될 수 있습니다. 즉, 복합 문서가 하위 문서 하나를 호출하고, 이 하위 문서가 마지막 문서를 호출할 수 있습니다.

복합 문서를 생성하려면 사용자 정의 SSM 문서에서 `aws:runDocument` 플러그인을 추가하고 필요한 입력을 지정합니다. 다음은 다음 작업을 수행하는 복합 문서의 예입니다.

1. `aws:downloadContent` 플러그인을 실행하여 GitHub 퍼블릭 리포지토리에서 SSM 문서를 부트스트랩이라고 하는 로컬 디렉터리로 다운로드합니다. SSM 문서는 StateManagerBootstrap.yml이라고 합니다(YAML 문서).
2. `aws:runDocument` 플러그인을 실행하여 StateManagerBootstrap.yml 문서를 실행합니다. 파라미터는 지정하지 않습니다.
3. `aws:runDocument` 플러그인을 실행하여 `AWS-ConfigureDocker pre-defined` SSM 문서를 실행합니다. 지정된 파라미터가 인스턴스에 도커를 설치합니다.

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

추가 정보

- Run Command를 사용할 때 서버와 인스턴스를 재부팅하여 스크립트를 호출하는 방법은 명령 실행 시 재부팅 처리 섹션을 참조하세요.
- 사용자 정의 SSM 문서에 추가할 수 있는 플러그인에 대한 자세한 내용은 Command 문서 플러그인 참조 섹션을 참조하세요.
- (복합 문서를 생성하지 않고) 원격 위치에서 문서를 실행하려는 경우, 원격 위치에서 문서 실행 섹션을 참조하세요.

문서 작업

이 섹션에는 SSM 문서 사용 및 작업 방법에 대한 정보가 포함되어 있습니다.

내용

State Manager 연결에서 SSM 문서 사용

AWS Systems Manager의 기능인 State Manager용 SSM 문서를 생성하는 경우 문서를 시스템에 추가한 후 관리형 인스턴스에 연결해야 합니다. 자세한 내용은 Systems Manager에서 연결 작업 단원을 참조하십시오.

State Manager 연결에서 SSM 문서를 사용할 때는 다음 세부 사항에 유의하세요.

- 다양한 문서를 사용하는 서로 다른 State Manager 연결을 생성하여 대상에 여러 문서를 할당할 수 있습니다.
- 서로 충돌하는 플러그인(예: 도메인 조인과 도메인으로부터 제거) 문서를 생성하면 마지막으로 실행된 플러그인이 최종 상태가 됩니다. State Manager는 문서 내에서 명령이나 플러그인의 논리적 순서나 무결성을 확인하지 않습니다.
- 문서를 처리할 때 인스턴스 연결이 먼저 적용된 다음 태그 지정된 그룹 연결이 적용됩니다. 인스턴스가 여러 태그 지정된 그룹의 일부인 경우 태그 지정된 그룹의 일부인 문서는 특정 순서로 실행되지 않습니다. 인스턴스가 인스턴스 ID별로 여러 문서를 통해 직접 대상으로 지정되면 특정한 실행 순서가 없습니다.
- State Manager에 대한 SSM Policy 문서의 기본 버전을 변경할 경우 이 문서를 사용하는 연결은 다음 번에 Systems Manager가 인스턴스에 연결을 적용할 때 새로운 기본 버전을 사용하기 시작합니다.

- 공유되는 SSM 문서를 사용하여 연결을 생성하고, 소유자가 문서 공유를 중지한 경우 연결에서 해당 문서에 더 이상 액세스할 수 없습니다. 하지만 소유자가 나중에 동일한 SSM 문서를 공유하는 경우 연결이 자동으로 다시 매핑됩니다.

SSM 문서 버전 비교

Systems Manager Documents 콘솔에서 AWS Systems Manager(SSM) 문서 버전 간의 내용 차이를 비교할 수 있습니다. SSM 문서의 버전을 비교할 때 버전 내용 간의 차이가 강조 표시됩니다.

SSM 문서 내용을 비교하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. 문서 목록에서 내용을 비교하려는 문서를 선택합니다.
4. [콘텐츠(Content)] 탭에서 [버전 비교(Compare versions)]를 선택하고 콘텐츠를 비교하려는 문서의 버전을 선택합니다.

SSM 문서 생성(콘솔)

SSM 문서 콘텐츠 작성 섹션에 설명된 대로 사용자 정의 SSM 문서에 대한 콘텐츠를 생성한 후 Systems Manager 콘솔을 통해 콘텐츠를 사용하는 SSM 문서를 생성할 수 있습니다.

SSM 문서를 생성하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. 명령 또는 세션 생성을 선택합니다.
4. 문서 이름을 서술식으로 입력합니다.
5. (선택 사항) Target type(대상 유형)에는 문서를 실행할 수 있는 리소스 유형을 지정합니다.
6. 문서 유형 목록에서 생성할 문서의 유형을 선택합니다.
7. 콘텐츠 필드에서 괄호를 삭제한 후 앞서 생성한 문서 콘텐츠를 붙여 넣습니다.
8. (선택 사항) Document tags(문서 태그) 섹션에서 문서에 적용할 하나 이상의 태그 키 이름/값 페어를 입력합니다.

   태그는 리소스에 할당하는 선택적 메타데이터입니다. 태그를 사용하면 용도, 소유자 또는 환경을 기준으로 하는 등 리소스를 다양한 방식으로 분류할 수 있습니다. 예를 들어 문서에 태그를 지정하

여 실행하는 작업 유형, 대상 운영 체제 유형 및 실행 환경을 식별할 수 있습니다. 다음 키 이름/값 페어를 지정할 수 있습니다.

- `Key=TaskType,Value=MyConfigurationUpdate`
- `Key=OS,Value=AMAZON_LINUX_2`
- `Key=Environment,Value=Production`

Systems Manager 리소스 태그 지정에 대한 자세한 내용은 Systems Manager 리소스에 태그 지정 섹션을 참조하세요.

9. 문서 생성을 선택하여 문서를 저장합니다.

SSM 문서 생성(명령줄)

SSM 문서 콘텐츠 작성 섹션에 설명된 대로 사용자 정의 AWS Systems Manager(SSM) 문서에 대한 콘텐츠를 생성한 후 AWS Command Line Interface(AWS CLI) 또는 AWS Tools for PowerShell을 통해 콘텐츠를 사용하는 SSM 문서를 생성할 수 있습니다. 다음 명령을 참조하십시오.

시작하기 전 준비 사항

아직 하지 않은 경우 AWS CLI 또는 AWS Tools for PowerShell을 설치하고 구성합니다. 자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트 및 AWS Tools for PowerShell 설치를 참조하세요.

다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

Linux & macOS

```
aws ssm create-document \
--content file://path/to/file/documentContent.json \
--name "document-name" \
--document-type "Command" \
--tags "Key=tag-key,Value=tag-value"
```

Windows

```
aws ssm create-document ^
--content file://C:\path\to\file\documentContent.json ^
--name "document-name" ^
--document-type "Command" ^
```

```
--tags "Key=tag-key,Value=tag-value"
```

PowerShell

```
$json = Get-Content -Path "C:\path\to\file\documentContent.json" | Out-String
New-SSMDocument `
-Content $json `
-Name "document-name" `
-DocumentType "Command" `
-Tags "Key=tag-key,Value=tag-value"
```

이 작업이 성공하면 명령에서 다음과 비슷한 응답이 반환됩니다.

```
{
"DocumentDescription":{
  "CreatedDate":1.585061751738E9,
  "DefaultVersion":"1",
  "Description":"MyCustomDocument",
  "DocumentFormat":"JSON",
  "DocumentType":"Command",
  "DocumentVersion":"1",
  "Hash":"0d3d879b3ca072e03c12638d0255ebd004d2c65bd318f8354fcde820dEXAMPLE",
  "HashType":"Sha256",
  "LatestVersion":"1",
  "Name":"Example",
  "Owner":"111122223333",
  "Parameters":[
     --truncated--
  ],
  "PlatformTypes":[
     "Windows",
     "Linux"
  ],
  "SchemaVersion":"0.3",
  "Status":"Creating",
  "Tags": [
        {
            "Key": "Purpose",
            "Value": "Test"
        }
    ]
}
```

```
}
```

SSM 문서 생성(API)

SSM 문서 콘텐츠 작성 섹션에 설명된 대로 사용자 정의 AWS Systems Manager(SSM) 문서에 대한 콘텐츠를 생성한 후 원하는 SDK를 통해 AWS Systems Manager CreateDocument API 작업을 호출하여 콘텐츠를 사용하는 SSM 문서를 생성할 수 있습니다. `Content` 요청 파라미터에 대한 JSON 또는 YAML 문자열은 일반적으로 파일에서 읽어옵니다. 다음 샘플 함수는 SDK for Python, SDK for Go 및 SDK for Java를 사용하여 SSM 문서를 생성합니다.

Python

```
import boto3

ssm = boto3.client('ssm')
filepath = '/path/to/file/documentContent.yaml'

def createDocumentApiExample():
with open(filepath) as openFile:
    documentContent = openFile.read()
    createDocRequest = ssm.create_document(
        Content = documentContent,
        Name = 'createDocumentApiExample',
        DocumentType = 'Automation',
        DocumentFormat = 'YAML'
    )
    print(createDocRequest)

createDocumentApiExample()
```

Go

```
package main

import (
"github.com/aws/aws-sdk-go/aws"
"github.com/aws/aws-sdk-go/aws/session"
"github.com/aws/aws-sdk-go/service/ssm"

"fmt"
```

```
"io/ioutil"
"log"
)


func main() {
openFile, err := ioutil.ReadFile("/path/to/file/documentContent.yaml")
if err != nil {
    log.Fatal(err)
}
documentContent := string(openFile)
sesh := session.Must(session.NewSessionWithOptions(session.Options{
    SharedConfigState: session.SharedConfigEnable}))

ssmClient := ssm.New(sesh)
createDocRequest, err := ssmClient.CreateDocument(&ssm.CreateDocumentInput{
    Content: &documentContent,
    Name:  aws.String("createDocumentApiExample"),
    DocumentType: aws.String("Automation"),
    DocumentFormat: aws.String("YAML"),
})
result := *createDocRequest
fmt.Println(result)
}
```

Java

```


import java.io.IOException;
import java.nio.charset.Charset;
import java.nio.charset.StandardCharsets;
import java.nio.file.Files;
import java.nio.file.Paths;

import com.amazonaws.AmazonClientException;
import com.amazonaws.AmazonServiceException;
import com.amazonaws.auth.AWSCredentials;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.simplesystemsmanagement.AWSSimpleSystemsManagement;
import
 com.amazonaws.services.simplesystemsmanagement.AWSSimpleSystemsManagementClientBuilder;
import com.amazonaws.services.simplesystemsmanagement.model.*;
```

```
public class createDocumentApiExample {
public static void main(String[] args) {
try {
  createDocumentMethod(getDocumentContent());
}
catch (IOException e) {
  e.printStackTrace();
}
}
public static String getDocumentContent() throws IOException {
  String filepath = new String("/path/to/file/documentContent.yaml");
    byte[] encoded = Files.readAllBytes(Paths.get(filepath));
    String documentContent = new String(encoded, StandardCharsets.UTF_8);
    return documentContent;
}

public static void createDocumentMethod (final String documentContent) {
  AWSSimpleSystemsManagement ssm =
 AWSSimpleSystemsManagementClientBuilder.defaultClient();
  final CreateDocumentRequest createDocRequest = new CreateDocumentRequest()
    .withContent(documentContent)
    .withName("createDocumentApiExample")
    .withDocumentType("Automation")
    .withDocumentFormat("YAML");
  final CreateDocumentResult result = ssm.createDocument(createDocRequest);
}
}
```

사용자 정의 문서 콘텐츠 생성에 대한 자세한 내용은 데이터 요소 및 파마미터 섹션을 참조하세요.

사용자 지정 SSM 문서 삭제 중

사용자 지정 SSM 문서를 더 이상 사용하지 않을 경우 AWS Command Line Interface(AWS CLI) 또는 AWS Systems Manager 콘솔을 사용하여 삭제할 수 있습니다.

SSM 문서를 삭제하려면(AWS CLI)

1. 문서를 삭제하기 전에 문서와 연결된 모든 인스턴스의 연결을 해제하는 것이 좋습니다.

   다음 명령을 실행하여 문서에서 인스턴스의 연결을 해제합니다.

```
aws ssm delete-association --instance-id "123456789012" --name "documentName"
```

명령이 성공해도 결과는 없습니다.

2. 다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

Linux

```
aws ssm delete-document \
    --name "document-name" \
    --document-version "document-version" \
    --version-name "version-name"
```

Windows

```
aws ssm delete-document ^
    --name "document-name" ^
    --document-version "document-version" ^
    --version-name "version-name"
```

PowerShell

```
Delete-SSMDocument `
    -Name "document-name" `
    -DocumentVersion 'document-version' `
    -VersionName 'version-name'
```

명령이 성공해도 결과는 없습니다.

> **Important**
>
> 만약 `document-version` 또는 `version-name`이 제공되지 않으면 문서의 모든 버전이 삭제됩니다.

SSM 문서를 삭제하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.

2. 탐색 창에서 Documents를 선택합니다.
3. 삭제할 문서를 선택합니다.
4. 삭제(Delete)를 선택합니다. 문서를 삭제할 것인지 묻는 메시지가 나타나면 삭제(Delete)를 선택합니다.

원격 위치에서 문서 실행

`AWS-RunDocument` 사전 정의 AWS Systems Manager(SSM) 문서를 사용하면 원격 위치로부터 SSM 문서를 실행할 수 있습니다. 이 문서는 다음 위치에 저장된 SSM 문서 실행을 지원합니다.

- 퍼블릭 및 프라이빗 GitHub 리포지토리(GitHub Enterprise는 지원되지 않음)
- Amazon S3 버킷
- Systems Manager

AWS Systems Manager의 기능인 State Manager 또는 Automation을 사용하여 원격 문서를 실행할 수도 있지만 다음 절차에서는 Systems Manager 콘솔에서 AWS Systems Manager Run Command를 사용하여 원격 SSM 문서를 실행하는 방법만 설명합니다.

> **Note**
>
> `AWS-RunDocument`는 Automation 실행서 등의 다른 유형이 아닌 명령 유형 SSM 문서만 실행하는 데 사용할 수 있습니다. `AWS-RunDocument`는 `aws:downloadContent` 플러그인을 사용합니다. `aws:downloadContent` 플러그인에 대한 자세한 내용은 `aws:downloadContent` 섹션을 참조하세요.

시작하기 전 준비 사항

원격 문서를 실행하기 전에 다음 작업을 완료해야 합니다.

- SSM Command 문서를 생성하여 원격 위치에 저장합니다. 자세한 내용은 SSM 문서 콘텐츠 생성 단원을 참조하세요.
- 프라이빗 GitHub 리포지토리에 저장된 원격 문서를 실행하려는 경우 GitHub 보안 액세스 토큰에 대한 Systems Manager `SecureString` 파라미터를 생성해야 합니다. 수동으로 SSH를 통해 토큰을 전달해서는 프라이빗 GitHub 리포지토리에 저장된 원격 문서에 액세스할 수 없습니다. 액세스 토큰은 Systems Manager `SecureString` 파라미터로 전달되어야 합니다. `SecureString` 파라미터 생성에 대한 자세한 내용은 Systems Manager 파라미터 생성 섹션을 참조하세요.

원격 문서 실행(콘솔)

원격 문서를 실행하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택합니다.
3. Run command(Run 명령)를 선택합니다.
4. [문서(Document)] 목록에서 **AWS-RunDocument**를 선택합니다.
5. Command parameters(명령 파라미터)의 소스 유형에서 옵션을 선택합니다.

   - GitHub를 선택하는 경우 다음 형식으로 소스 정보의 정보를 지정합니다.

```
{
    "owner": "owner_name",
    "repository": "repository_name",
    "path": "path_to_document",
    "getOptions":"branch:branch_name",
    "tokenInfo": "{{ssm-secure:secure-string-token}}"
}
```

   예:

```
{
    "owner":"TestUser",
    "repository":"GitHubTestExamples",
    "path":"scripts/python/test-script",
    "getOptions":"branch:exampleBranch",
    "tokenInfo":"{{ssm-secure:my-secure-string-token}}"
}
```

> **Note**
>
> getOptions는 마스터가 아닌 분기 또는 리포지토리의 특정 커밋에서 내용을 검색하는 추가 옵션입니다. 마스터 분기에서 최신 커밋을 사용하는 경우 getOptions를 생략할 수 있습니다. branch 파라미터는 SSM 문서가 master가 아닌 분기에 저장된 경우에만 필요합니다.
> 리포지토리의 특정 커밋에 있는 SSM 문서의 버전을 사용하려면 branch대신 getOptions에서 commitID를 사용합니다. 예:

```
"getOptions": "commitID:bbc1ddb94...b76d3bEXAMPLE",
```

- S3를 선택하는 경우 다음 형식으로 Source Info(소스 정보) 정보를 지정합니다.

```
{"path":"URL_to_document_in_S3"}
```

예:

```
{"path":"https://s3.amazonaws.com/DOC-EXAMPLE-BUCKET/scripts/ruby/mySSMdoc.json"}
```

- SSMDocument를 선택하는 경우 다음 형식으로 Source Info(소스 정보) 정보를 지정합니다.

```
{"name": "document_name"}
```

예:

```
{"name": "mySSMdoc"}
```

6. [문서 파라미터(Document Parameters)] 필드에서 원격 SSM 문서에 대한 파라미터를 입력합니다. 예를 들어 AWS-RunPowerShell 문서를 실행하는 경우 다음을 지정할 수 있습니다.

```
{"commands": ["date", "echo \"Hello World\""]}
```

AWS-ConfigureAWSPack 문서를 실행하는 경우 다음을 지정할 수 있습니다.

```
{
   "action":"Install",
   "name":"AWSPVDriver"
}
```

7. 대상(Targets) 섹션에서, 태그를 지정하거나, 수동으로 인스턴스나 엣지 디바이스를 선택하거나, 리소스 그룹을 지정하여 이 작업을 실행할 관리형 노드를 식별합니다.

> **Tip**
>
> 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

8. Other parameters(다른 파라미터):

   - Comment(설명)에 명령에 대한 정보를 입력합니다.
   - 제한 시간(초)에서 전체 명령 실행이 실패할 때까지 시스템이 기다리는 시간을 초 단위로 지정합니다.

9. Rate control(속도 제어)에서

   - Concurrency(동시성)에서 명령을 동시에 실행할 관리형 노드의 백분율 또는 개수를 지정합니다.

     > **Note**
     >
     > 관리형 노드에 적용할 태그를 지정하거나, AWS 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상 수를 제한합니다.

   - Error threshold(오류 임계값)에서, 명령이 노드의 개수 또는 백분율에서 실패한 후 다른 관리형 노드에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째 오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.

10. (선택 사항) Output options(출력 옵션)에서 명령 출력을 파일에 저장하려면 Write command output to an S3 bucket(S3 버킷에 명령 출력 쓰기) 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

    > **Note**
    >
    > 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아니라 인스턴스에 할당된 인스턴스 프로파일(EC2 인스턴스용) 또는 IAM 서비스 역할(하이브리드 정품 인증 시스템)의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

11. SNS notifications(SNS 알림) 섹션에서, 명령 실행 상태에 대한 알림이 전송되도록 하려면 Enable SNS notifications(SNS 알림 활성화) 확인란을 선택합니다.

    Run Command에 대한 Amazon SNS 알림 구성에 대한 자세한 내용은 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 섹션을 참조하세요.

12. Run(실행)을 선택합니다.

> **Note**
>
> Run Command를 사용할 때 서버와 인스턴스를 재부팅하여 스크립트를 호출하는 방법은 명령 실행 시 재부팅 처리 섹션을 참조하세요.

SSM 문서 공유

동일한 AWS 리전에서 계정과 비공개적으로 또는 공개적으로 AWS Systems Manager(SSM) 문서를 공유할 수 있습니다. 문서를 사적으로 공유하려면 문서 권한을 수정하여 특정 개인들이 자신의 AWS 계정 ID에 따라 액세스할 수 있도록 허용합니다. SSM 문서를 공개적으로 공유하려면 문서 권한을 수정하여 `All`로 지정합니다. 문서는 공개 및 비공개로 동시에 공유할 수 없습니다.

> **Warning**
>
> 신뢰할 수 있는 소스에서 공유된 SSM 문서만 사용합니다. 공유 문서를 사용할 때는 그 전에 문서의 내용을 미리 신중하게 검토하여 해당 문서가 자신의 인스턴스 구성을 어떻게 바꾸어놓을지 이해해야 합니다. 공유 문서 모범 사례에 대한 자세한 내용은 공유 SSM 문서에 대한 모범 사례 섹션을 참조하세요.

제한 사항

SSM 문서로 작업을 시작할 때 다음과 같은 제한 사항에 유의합니다.

- 소유자만이 문서를 공유할 수 있습니다.
- 문서를 삭제하려면 먼저 문서 공유를 중단해야 합니다. 자세한 내용은 공유 SSM 문서에 대한 권한 수정 단원을 참조하십시오.
- 최대 1,000개의 AWS 계정와 문서를 공유할 수 있습니다. AWS Support Center에 이 제한에 대한 증가를 요청할 수 있습니다. [제한 유형(Limit type)]에서 [EC2 Systems Manager]를 선택하고 요청 이유를 설명합니다.
- 최대 5건의 SSM 문서를 공개적으로 공유할 수 있습니다. AWS Support Center에 이 제한에 대한 증가를 요청할 수 있습니다. [제한 유형(Limit type)]에서 [EC2 Systems Manager]를 선택하고 요청 이유를 설명합니다.

- 동일한 AWS 리전에서만 다른 계정과 문서를 공유할 수 있습니다. 크로스 리전 공유는 지원되지 않습니다.

Systems Manager Service Quotas에 대한 자세한 내용은 AWS Systems Manager Service Quotas 사용 설명서를 참조하세요.

내용

공유 SSM 문서에 대한 모범 사례

문서를 공유하기 전에 또는 공유된 문서를 사용하기 전에 다음 지침을 읽어보십시오.

민감한 정보의 삭제

AWS Systems Manager(SSM) 문서를 주의 깊게 살펴보고 민감한 정보는 모두 삭제합니다. 예를 들어 문서에 AWS 자격 증명이 포함되지 않도록 합니다. 특정 개인들과 문서를 공유하는 경우, 그 사용자들은 문서에 있는 정보를 볼 수 있습니다. 문서를 공개적으로 공유하는 경우에는 모든 사용자가 그 문서에 있는 정보를 볼 수 있습니다.

문서의 퍼블릭 공유 차단

사용 사례에서 퍼블릭 공유를 설정해야 하는 경우가 아니면 Systems Manager Documents 콘솔의 [기본 설정(Preferences)] 섹션에서 Systems Manager Documents의 퍼블릭 공유 차단 설정을 켜는 것이 좋습니다.

IAM 신뢰 정책을 사용하여 Run Command 작업 제한

문서에 대한 액세스 권한을 얻게 될 사용자들에 대해 제한적인 AWS Identity and Access Management(IAM) 정책을 생성합니다. IAM 정책은 사용자가 Amazon Elastic Compute Cloud(Amazon EC2) 콘솔에서 또는 AWS Command Line Interface(AWS CLI)나 AWS Tools for Windows PowerShell로 `ListDocuments`을 호출하여 볼 수 있는 SSM 문서를 결정합니다. 또한 이 정책은 사용자가 SSM 문서에서 수행할 수 있는 작업을 제한합니다. 제한적인 정책을 생성하여 사용자가 특정 문서만 사용하도록 할 수 있습니다. 자세한 내용은 고객 관리형 정책 예 단원을 참조하십시오.

공유 SSM 문서를 사용할 때는 주의해야 합니다.

자신에게 공유된 모든 문서, 특히 퍼블릭 문서의 내용을 검토하여 자신의 인스턴스에서 실행될 명령을 이해합니다. 문서가 실행된 후에는 의도적이든 비의도적이든 부정적인 영향을 미칠 수 있습니다. 문서가 외부 네트워크를 참조하는 경우, 그 문서를 사용하기 전에 외부 소스를 검토하십시오.

문서 해시를 사용해 명령 전송하기

문서를 공유하면 시스템이 SHA256 해시를 생성해 그것을 문서에 할당합니다. 또한 시스템은 문서 내용에 대한 스냅샷을 저장합니다. 공유된 문서를 사용해 명령을 전송할 때 명령에 해시를 지정해 다음 조건이 참이 되도록 할 수 있습니다.

- 정확한 Systems Manager 문서에서 명령을 실행하고 있습니다.
- 문서가 공유된 이후 그 내용이 변경되지 않았습니다.

해시가 지정된 문서와 일치하지 않거나 공유된 문서의 내용이 변경된 경우 명령은 `InvalidDocument` 예외를 반환합니다. 해시는 외부 위치에서 온 문서 내용은 확인할 수 없습니다.

SSM 문서의 퍼블릭 공유 차단

사용 사례에서 퍼블릭 공유를 설정해야 하는 경우가 아니면 AWS Systems Manager(SSM) 문서에 대해 퍼블릭 공유 차단 설정을 켜는 것이 좋습니다. 이 설정을 켜면 SSM 문서에 대한 원치 않는 액세스를 방지할 수 있습니다. 퍼블릭 공유 차단 설정은 AWS 리전마다 다를 수 있는 계정 수준 설정입니다. SSM 문서의 퍼블릭 공유를 차단하려면 다음 태스크를 수행합니다.

퍼블릭 공유 차단(콘솔)

SSM 문서의 퍼블릭 공유를 차단하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. [기본 설정(Preferences)]을 선택하고 [퍼블릭 공유 차단(Block public sharing)] 섹션에서 [편집(Edit)]을 선택합니다.
4. [퍼블릭 공유 차단(Block public sharing)] 확인란을 선택하고 [저장(Save)]을 선택합니다.

퍼블릭 공유 차단(명령줄)

로컬 컴퓨터에서 AWS Command Line Interface(AWS CLI) 또는 AWS Tools for Windows PowerShell을 열고 다음 명령을 실행하여 SSM 문서의 퍼블릭 공유를 차단합니다.

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
```

다음 명령을 사용하여 설정 값이 업데이트되었는지 확인합니다.

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
```

액세스를 제한하여 IAM과의 퍼블릭 공유 차단

사용자가 퍼블릭 공유 차단 설정을 수정하지 못하도록 제한하는 AWS Identity and Access Management(IAM) 정책을 생성할 수 있습니다. 이렇게 하면 사용자가 SSM 문서에 대한 원치 않는 액세스를 허용할 수 없습니다.

다음은 사용자가 퍼블릭 공유 차단 설정을 업데이트하지 못하도록 하는 IAM 정책의 예입니다. 이 예를 사용하려면 예제 Amazon Web Services 계정 ID를 사용자의 계정 ID로 바꿔야 합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "ssm:UpdateServiceSetting",
            "Resource": "arn:aws:ssm:*:987654321098:servicesetting/ssm/documents/
console/public-sharing-permission"
        }
    ]
}
```

SSM 문서 공유

Systems Manager 콘솔을 사용하여 AWS Systems Manager(SSM) 문서를 공유할 수 있습니다. 콘솔의 문서를 공유하는 경우 문서의 기본 버전만 공유할 수 있습니다. AWS Command Line Interface(AWS CLI), AWS Tools for Windows PowerShell 또는 AWS SDK를 사용하여 `ModifyDocumentPermission` API 작업을 호출하여 프로그래밍 방식으로 SSM 문서를 공유할 수도 있습니다. 문서를 공유하기 전에 먼저 공유하고자 하는 사람들의 AWS 계정 ID가 있어야 합니다. 문서를 공유할 때 이들 계정 ID를 지정해야 합니다.

문서 공유(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. 문서 목록에서 공유하고자 하는 문서를 선택한 후 세부 정보 보기(View details)를 선택합니다. [권한(Permissions)] 탭에서 자신이 문서의 소유자인지 확인합니다. 문서 소유자만이 문서를 공유할 수 있습니다.

4. 편집을 선택합니다.
5. 명령을 공개적으로 공유하려면 퍼블릭을 선택한 후, 저장을 선택합니다. 명령을 사적으로 공유하려면 [프라이빗(Private)]을 선택하고 AWS 계정 ID를 입력하고 [권한 추가(Add permission)]를 선택한 후 [저장(Save)]을 선택합니다.

문서 공유(명령줄)

다음 절차를 수행하려면 명령줄 세션에 대해 AWS 리전을 지정해야 합니다.

1. 로컬 컴퓨터에서 AWS CLI 또는 AWS Tools for Windows PowerShell을 열고 다음 명령을 실행하여 자격 증명을 지정합니다.

   다음 명령에서 *region*을 자신의 정보로 바꿉니다. 지원되는 *##* 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

   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. 다음 명령을 실행하여 사용 가능한 모든 SSM 문서를 나열합니다. 목록에는 자신이 생성한 문서와 자신에게 공유된 문서가 포함됩니다.

Linux & macOS

```
aws ssm list-documents
```

Windows

```
aws ssm list-documents
```

PowerShell

```
Get-SSMDocumentList
```

3. 다음 명령을 사용하여 특정 문서를 얻습니다.

Linux & macOS

```
aws ssm get-document \
    --name document name
```

Windows

```
aws ssm get-document ^
    --name document name
```

PowerShell

```
Get-SSMDocument `
    -Name document name
```

4. 다음 명령을 사용하여 문서에 대한 설명을 봅니다.

Linux & macOS

```
aws ssm describe-document \
    --name document name
```

Windows

```
aws ssm describe-document ^
```

```
    --name document name
```

PowerShell

```
Get-SSMDocumentDescription `
    -Name document name
```

5. 다음 명령을 사용하여 문서에 대한 권한을 확인합니다.

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. 다음 명령을 사용하여 문서에 대한 권한을 수정한 후 문서를 공유합니다. 권한을 편집하기 위해서는 문서의 소유자이어야 합니다. 선택적으로 `--shared-document-version` 파라미터를 사용하여 공유하려는 문서 버전을 지정할 수 있습니다. 버전을 지정하지 않으면 시스템에서 문서의 `Default` 버전을 공유합니다. 예제 명령에서는 해당자의 AWS 계정 ID에 따라 특정 개인과 비공개로 문서를 공유합니다.

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. 다음 명령을 사용하여 문서를 공개적으로 공유합니다.

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')
```

공유 SSM 문서에 대한 권한 수정

명령을 공유하면 사용자들은 AWS Systems Manager(SSM) 문서에 대한 액세스 권한을 제거하거나 SSM 문서를 삭제할 때까지 그 명령을 보고 사용할 수 있습니다. 그러나 문서가 공유되고 있는 한 삭제는 할 수 없습니다. 먼저 공유를 중단하고 나서 삭제해야 합니다.

문서 공유 중지(콘솔)

문서 공유 중단

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. 문서 목록에서 공유를 중단하고자 하는 문서를 선택한 후 세부 정보를 선택합니다. 권한 섹션에서 자신이 문서의 소유자인지 확인합니다. 문서 소유자만이 문서 공유를 중단할 수 있습니다.
4. 편집을 선택합니다.
5. X를 선택하여 명령에 대한 액세스 권한이 있어서는 안 되는 AWS 계정 ID를 삭제한 다음 [저장(Save)]을 선택합니다.

문서 공유 중지(명령줄)

로컬 컴퓨터에서 AWS CLI 또는 AWS Tools for Windows PowerShell을 열고 다음 명령을 실행하여 명령 공유를 중지합니다.

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
```

공유 SSM 문서 사용

AWS Systems Manager(SSM) 문서를 공유하면 시스템은 Amazon 리소스 이름(ARN)을 생성하고 명령에 할당합니다. Systems Manager 콘솔에서 공유 문서를 선택하고 실행하면 ARN이 보이지 않습니다. 그러나 공유 SSM 문서를 Systems Manager 콘솔이 아닌 다른 방법으로 실행하고자 한다면 DocumentName 요청 파라미터에 대한 문서의 전체 ARN을 지정해야 합니다. 이 명령을 실행해 문서를 나열하면 SSM 문서에 대한 전체 ARN이 나타납니다.

> **Note**
>
> AWS 퍼블릭 문서(AWS-*로 시작하는 문서)나 자신이 소유한 문서에 대해서는 ARN을 지정할 필요가 없습니다.

공유 SSM 문서 사용(명령줄)

퍼블릭 SSM 문서를 모두 나열

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)
```

자신에게 공유된 프라이빗 SSM 문서를 나열

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)
```

사용할 수 있는 모든 SSM 문서를 나열

Linux & macOS

```
aws ssm list-documents
```

Windows

```
aws ssm list-documents
```

PowerShell

```
Get-SSMDocumentList
```

자신에게 공유된 SSM 문서에 대한 정보 확인

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
```

공유 SSM 문서 실행

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
```

SSM 문서 검색

자유 텍스트 검색 또는 필터 기반 검색을 사용하여 AWS Systems Manager(SSM) 문서 저장소에서 SSM 문서를 검색할 수 있습니다. 자주 사용하는 SSM 문서를 찾는 데 도움이 되도록 문서를 즐겨찾기에 추가할 수도 있습니다. 다음 섹션에서는 이러한 기능을 사용하는 방법을 설명합니다.

자유 텍스트 검색 사용

Systems Manager [Documents] 페이지의 검색 상자에서 자유 텍스트 검색을 지원합니다. 자유 텍스트 검색은 입력한 검색어를 각 SSM 문서의 문서 이름과 비교합니다. 단일 검색어(예: **`ansible`**)를 입력하면 Systems Manager는 이 용어가 발견된 모든 SSM 문서를 반환합니다. 여러 검색어를 입력하면 Systems Manager는 OR 문을 사용하여 검색합니다. 예를 들어 **`ansible`** 및 **`linux`**을 지정하면 검색 시 이름에 두 키워드 중 하나가 포함된 모든 문서가 반환됩니다.

자유 텍스트 검색어를 입력하고 [플랫폼 유형(Platform type)]과 같은 검색 옵션을 선택하면 검색에 AND 문이 사용되고 이름에 키워드를 포함하는 지정된 플랫폼 유형의 모든 문서가 반환됩니다.

> **Note**
>
> 자유 텍스트 검색에 대한 다음 세부 정보에 유의합니다.
>
> - 자유 텍스트 검색은 대/소문자를 구분하지 않습니다.
> - 검색어는 최소 3자, 최대 20자여야 합니다.
> - 자유 텍스트 검색은 최대 5개의 검색어를 허용합니다.
> - 검색어 사이에 공백을 입력하면 검색 시 공백이 포함됩니다.
> - 자유 텍스트 검색을 [문서 유형(Document type)] 또는 [플랫폼 유형(Platform type)]과 같은 다른 검색 옵션과 결합할 수 있습니다.
> - [문서 이름 접두사(Document Name Prefix)] 필터와 자유 텍스트 검색은 상호 배타적이므로 함께 사용할 수 없습니다.

SSM 문서를 검색하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. 검색 상자에 검색어를 입력하고 Enter 키를 누릅니다.

AWS CLI를 사용하여 자유 텍스트 문서 검색 수행

CLI를 사용하여 자유 텍스트 문서 검색을 수행하려면

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.
2. 한 용어로 자유 텍스트 문서 검색을 수행하려면 다음 명령을 실행합니다. 이 명령에서 *search_term*을 자신의 정보로 바꿉니다.

   ```
   aws ssm list-documents --filters Key="SearchKeyword",Values="search_term"
   ```

   다음은 그 예입니다.

   ```
   aws ssm list-documents --filters Key="SearchKeyword",Values="aws-asg" --region us-
   east-2
   ```

   AND 문을 생성하는 여러 용어를 사용하여 검색하려면 다음 명령을 실행합니다. 이 명령에서 *search_term_1*과 *search_term_2*를 자신의 정보로 바꿉니다.

   ```
   aws ssm list-documents --filters
    Key="SearchKeyword",Values="search_term_1","search_term_2","search_term_3" --
   region us-east-2
   ```

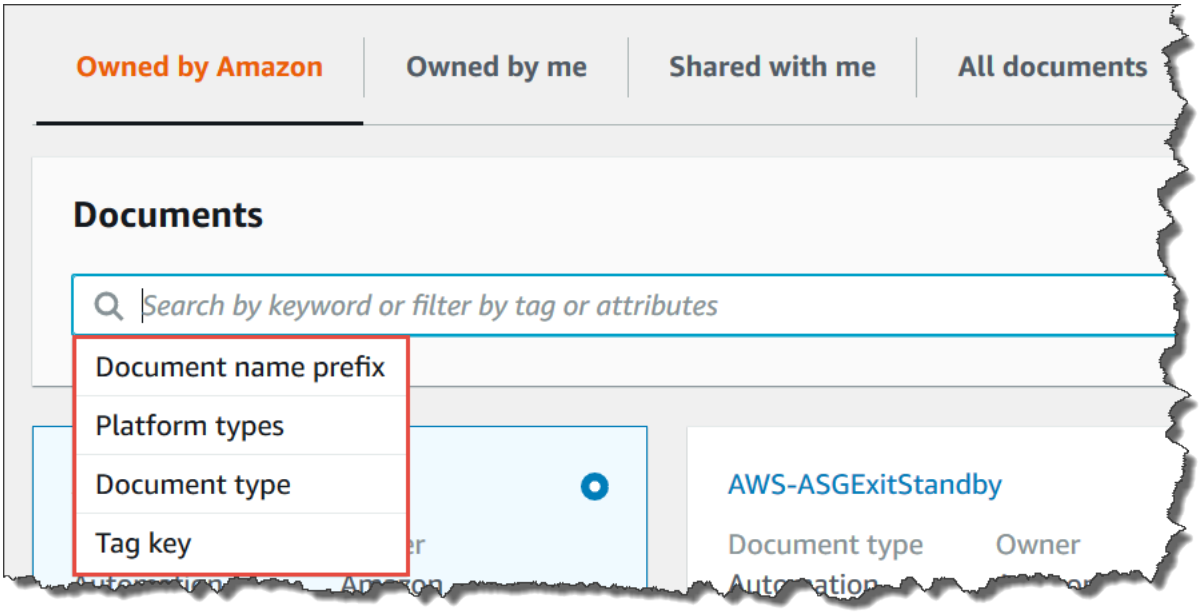
   다음은 그 예입니다.

   ```
   aws ssm list-documents --filters Key="SearchKeyword",Values="aws-asg","aws-
   ec2","restart" --region us-east-2
   ```

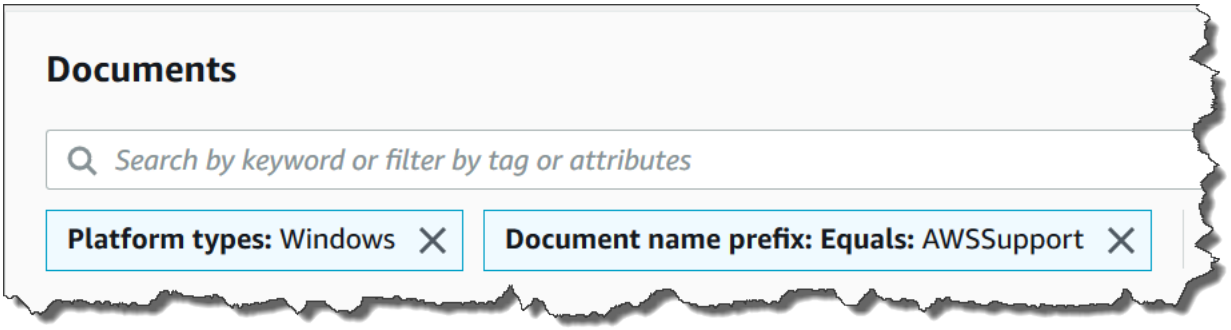
필터 사용

검색 상자를 선택하면 Systems Manager [Documents] 페이지에 자동으로 다음 필터가 표시됩니다.

- 문서 이름 접두사
- 플랫폼 유형
- 문서 유형
- 태그 키



단일 필터를 사용하여 SSM 문서를 검색할 수 있습니다. 좀 더 구체적인 SSM 문서 집합을 반환하려는 경우 여러 필터를 적용할 수 있습니다. 다음은 [플랫폼 유형(Platform types)] 및 [문서 이름 접두사(Document name prefix)] 필터를 사용하는 검색의 예입니다.



여러 필터를 적용하는 경우 Systems Manager는 선택한 필터에 따라 다른 검색 문을 생성합니다.

- 동일한 필터를 여러 번 적용하는 경우(예: [문서 이름 접두사(Document name prefix)]) Systems Manager는 OR 문을 사용하여 검색합니다. 예를 들어 [문서 이름 접두사(Document name prefix)]=**AWS** 필터 하나와 [문서 이름 접두사(Document name prefix)]=**Lambda** 필터를 또 하나 지정하는 경우 검색 시 접두사 "AWS"가 있는 모든 문서와 접두사 "Lambda"가 있는 모든 문서가 반환됩니다.
- 여러 필터를 적용하는 경우(예: 문서 이름 접두사(Document name prefix) 및 플랫폼 유형(Platform types)) Systems Manager는 AND 명령문을 사용하여 검색합니다. 예를 들어 문서 이름 접두사=**AWS** 필터와 플랫폼 유형(Platform types)=**Linux** 필터를 지정하면 검색 시 Linux 플랫폼에 특정한 접두사 "AWS"가 있는 모든 문서가 반환됩니다.

> **Note**
>
> 필터를 사용하는 검색은 대/소문자를 구분합니다.

즐겨찾기에 문서 추가

자주 사용하는 SSM 문서를 찾는 데 도움이 되도록 문서를 즐겨찾기에 추가합니다. 문서 유형, AWS 계정 및 AWS 리전로 최대 20개의 문서를 즐겨찾기에 추가할 수 있습니다. 문서 AWS Management Console에서 즐겨찾기를 선택, 수정 및 확인할 수 있습니다. 다음 절차에서는 즐겨찾기를 선택하고 수정하고 보는 방법을 설명합니다.

즐겨찾기에 SSM 문서 추가

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. 즐겨찾기에 추가하려는 문서 이름 옆의 별표 아이콘을 선택합니다.

즐겨찾기에서 SSM 문서 제거

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. 즐겨찾기에서 제거하려는 문서 이름 옆의 별표 아이콘 선택을 취소합니다.

문서 AWS Management Console에서 즐겨찾기 보기

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. 즐겨찾기 탭을 선택합니다.

AWS Systems Manager의 보안

Amazon Web Services에서 가장 우선순위가 높은 것이 클라우드 보안입니다. AWS 고객은 보안에 가장 보안에 민감한 조직의 요구 사항에 부합하도록 구축된 데이터 센터 및 네트워크 아키텍처의 혜택을 누릴 수 있습니다.

보안은 AWS와 여러분의 공동 책임입니다. 공동 책임 모델은 이 사항을 클라우드의 보안 및 클라우드 내 보안으로 설명합니다.

- 클라우드의 보안 - AWS는 AWS 클라우드에서 AWS 서비스를 실행하는 인프라를 보호합니다. AWS는 또한 안전하게 사용할 수 있는 서비스를 제공합니다. 서드 파티 감사자는 AWS 규정 준수 프로그램의 일환으로 보안 효과를 정기적으로 테스트하고 검증합니다. AWS Systems Manager에 적용되는 규정 준수 프로그램에 대한 자세한 내용은 규정 준수 프로그램의 범위에 속하는 AWS 서비스을(를) 참조하세요.
- 클라우드 내 보안 – 사용자의 책임은 사용자가 사용하는 AWS 서비스에 의해 결정됩니다. 또한 귀하는 데이터의 민감도, 회사 요구 사항, 관련 법률 및 규정을 비롯한 기타 요소에 대해서도 책임이 있습니다.

이 설명서는 AWS Systems Manager 사용 시 공동 책임 모델을 적용하는 방법을 이해하는 데 도움이 됩니다. 다음 주제에서는 보안 및 규정 준수 목표를 충족하도록 Systems Manager을(를) 구성하는 방법을 보여줍니다. 또한 Systems Manager 리소스를 모니터링하고 보호하는 데 도움이 되는 다른 AWS 서비스을(를) 사용하는 방법을 배우게 됩니다.

주제

AWS Systems Manager의 데이터 보호

데이터 보호란 전송 중(Systems Manager 안팎으로 데이터가 이동 중)과 저장된(AWS 데이터 센터에 데이터가 저장됨) 동안 데이터를 보호하는 것을 말합니다.

AWS 공동 책임 모델은 AWS Systems Manager의 데이터 보호에 적용됩니다. 이 모델에서 설명하는 것처럼 AWS는 모든 AWS 클라우드를 실행하는 글로벌 인프라를 보호할 책임이 있습니다. 사용자는 인프라에서 호스팅되는 콘텐츠를 관리해야 합니다. 사용하는 AWS 서비스의 보안 구성과 관리 작업에 대한 책임도 사용자에게 있습니다. 데이터 프라이버시에 대한 자세한 내용은 데이터 프라이버시 FAQ를 참조하세요. 유럽의 데이터 보호에 대한 자세한 내용은 AWS 보안 블로그의 AWS 공동 책임 모델 및 GDPR 블로그 게시물을 참조하세요.

데이터를 보호하려면 AWS 계정보안 인증 정보를 보호하고 AWS IAM Identity Center또는 AWS Identity and Access Management(IAM)를 통해 개별 사용자 계정을 설정하는 것이 좋습니다. 이렇게 하면 개별 사용자에게 자신의 직무를 충실히 이행하는 데 필요한 권한만 부여됩니다. 또한 다음과 같은 방법으로 데이터를 보호하는 것이 좋습니다.

- 각 계정에 멀티 팩터 인증 설정(MFA)을 사용하세요.
- SSL/TLS를 사용하여 AWS 리소스와 통신하세요. TLS 1.2는 필수이며 TLS 1.3를 권장합니다.
- AWS CloudTrail로 API 및 사용자 활동 로깅을 설정하세요.
- AWS 암호화 솔루션을 AWS 서비스 내의 모든 기본 보안 컨트롤과 함께 사용하세요.
- Amazon S3에 저장된 민감한 데이터를 검색하고 보호하는 데 도움이 되는 Amazon Macie와 같은 고급 관리형 보안 서비스를 사용하세요.
- 명령줄 인터페이스 또는 API를 통해 AWS에 액세스할 때 FIPS 140-2 검증된 암호화 모듈이 필요한 경우, FIPS 엔드포인트를 사용합니다. 사용 가능한 FIPS 엔드포인트에 대한 자세한 내용은 FIPS(Federal Information Processing Standard) 140-2를 참조하세요.

고객의 이메일 주소와 같은 기밀 정보나 중요한 정보는 태그나 이름 필드와 같은 자유 양식 필드에 입력하지 않는 것이 좋습니다. 여기에는 Systems Manager 또는 기타 AWS 서비스에서 콘솔, API, AWS CLI 또는 AWS SDK를 사용하여 작업하는 경우가 포함됩니다. 이름에 사용되는 태그 또는 자유 형식 텍스트 필드에 입력하는 모든 데이터는 청구 또는 진단 로그에 사용될 수 있습니다. 외부 서버에 URL을 제공할 때 해당 서버에 대한 요청을 검증하기 위해 보안 인증 정보를 URL에 포함해서는 안 됩니다.

데이터 암호화

저장 중 암호화

Parameter Store 파라미터

AWS Systems Manager의 기능인 Parameter Store에서 생성할 수 있는 파라미터 유형에는 `String`, `StringList` 및 `SecureString`이 포함됩니다.

`SecureString` 파라미터 값을 암호화하기 위해 Parameter Store는 AWS Key Management Service(AWS KMS)에서 AWS KMS key를 사용합니다. AWS KMS는 고객 관리형 키 또는 AWS 관리형 키를 사용하여 AWS 관리형 데이터베이스의 파라미터 값을 암호화합니다.

> **Important**
>
> `String` 또는 `StringList` 파라미터에 중요한 데이터를 저장하지 않습니다. 암호화된 상태로 유지해야 하는 모든 중요한 데이터의 경우 `SecureString` 파라미터 유형만 사용하십시오. 자세한 내용은 파라미터란 무엇인가요? 및 IAM 정책을 사용하여 Systems Manager 파라미터에 대한 액세스 제한 단원을 참조하세요.

S3 버킷의 콘텐츠

Systems Manager 작업의 일부로 데이터를 하나 이상의 Amazon Simple Storage Service(Amazon S3) 버킷에 업로드하거나 저장하도록 선택할 수 있습니다.

S3 버킷 암호화에 대한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 암호화를 사용하여 데이터 보호 및 Amazon S3에서 데이터 보호를 참조하세요.

다음은 Systems Manager 활동의 일부로 S3 버킷에 업로드하거나 저장할 수 있는 데이터 형식입니다.

- AWS Systems Manager의 기능인 Run Command의 명령 출력
- AWS Systems Manager의 기능인 Distributor의 패키지
- AWS Systems Manager의 기능인 Patch Manager의 패치 작업 로그
- Patch Manager 패치 재정의 목록
- AWS Systems Manager의 기능인 Automation의 런북 워크플로에서 실행할 스크립트 또는 Ansible 플레이북
- AWS Systems Manager의 기능인 Compliance의 검사에 사용하기 위한 Chef InSpec 프로파일

- AWS CloudTrail 로그
- AWS Systems Manager의 기능인 Session Manager의 세션 기록 로그
- AWS Systems Manager의 기능인 Explorer의 보고서
- AWS Systems Manager의 기능인 OpsCenter의 OpsData
- 자동화 워크플로에 사용할 AWS CloudFormation 템플릿
- 리소스 데이터 동기화 검사의 규정 준수 데이터
- 관리형 노드에서 AWS Systems Manager의 기능인 State Manager의 연결 생성 또는 편집 요청 출력
- AWS 관리형 SSM 문서 `AWS-RunDocument`를 사용하여 실행할 수 있는 사용자 정의 Systems Manager 문서(SSM 문서)

CloudWatch Logs 로그 그룹

Systems Manager 작업의 일부로 데이터를 하나 이상의 Amazon CloudWatch Logs 로그 그룹으로 스트리밍하도록 선택할 수 있습니다.

CloudWatch Logs 로그 그룹 암호화에 대한 자세한 내용은 Amazon CloudWatch Logs 사용 설명서의 AWS Key Management Service를 사용하여 CloudWatch Logs에서 로그 데이터 암호화를 참조하세요.

다음은 Systems Manager 활동의 일부로 CloudWatch Logs 로그 그룹으로 스트리밍할 수 있는 데이터 형식입니다.

- Run Command 명령의 출력
- 스크립트 출력은 Automation 실행서의 `aws:executeScript` 작업을 사용하여 실행됩니다.
- Session Manager 세션 기록 로그
- 관리형 노드의 SSM Agent의 로그

전송 중 암호화

클라이언트와 노드 간에 전송되는 중요한 데이터를 암호화하려면 전송 계층 보안(TLS)과 같은 암호화 프로토콜을 사용하는 것이 좋습니다.

Systems Manager는 전송 중인 데이터의 암호화에 대해 다음과 같은 지원을 제공합니다.

Systems Manager API 엔드포인트에 연결

Systems Manager API 엔드포인트는 HTTPS를 통한 보안 연결만을 지원합니다. AWS Management Console, AWS SDK 또는 Systems Manager API를 사용하여 Systems Manager 리소

스를 관리하면 모든 통신이 TLS(전송 계층 보안)로 암호화됩니다. API 엔드포인트의 전체 목록은 Amazon Web Services 일반 참조의 AWS 서비스 엔드포인트를 참조하세요.

관리형 인스턴스

AWS에서는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 간에 안전한 프라이빗 연결을 제공합니다. 또한 256비트 암호화의 AEAD 알고리즘을 사용하여 동일한 Virtual Private Cloud(VPC) 또는 피어 VPC의 지원되는 인스턴스 간 전송 중인 트래픽을 자동으로 암호화합니다. 이 암호화 기능은 기본 하드웨어의 오프라인 기능을 사용하며 네트워크 성능에는 영향을 주지 않습니다. 지원되는 인스턴스는 C5n, G4, I3en, M5dn, M5n, P3dn, R5dn 및 R5n입니다.

Session Manager 세션

기본적으로 Session Manager에서 TLS 1.2를 사용하여 계정 내 사용자의 로컬 머신과 EC2 인스턴스 사이에 전송되는 세션 데이터를 암호화합니다. AWS KMS에서 생성된 AWS KMS key를 사용하여 전송 중인 데이터를 추가로 암호화하도록 선택할 수도 있습니다. AWS KMS 암호화는 `Standard_Stream`, `InteractiveCommands`, 및 `NonInteractiveCommands` 세션 유형에 사용할 수 있습니다.

Run Command 액세스

기본적으로 Run Command를 사용하는 노드에 대한 원격 액세스는 TLS 1.2를 사용하여 암호화되며, 연결을 생성하기 위한 요청은 SigV4를 사용하여 서명됩니다.

인터네트워크 트래픽 개인 정보

Amazon Virtual Private Cloud(Amazon VPC)를 사용하여 관리형 노드의 리소스 간 경계를 생성하고 리소스, 온프레미스 네트워크 및 인터넷 간의 트래픽을 제어할 수 있습니다. 자세한 내용은 Systems Manager용 VPC 엔드포인트를 사용하여 EC2 인스턴스의 보안 개선을 참조하세요.

Amazon Virtual Private Cloud 보안에 대한 자세한 내용은 Amazon VPC 사용 설명서의 Amazon VPC의 인터네트워크 트래픽 개인 정보 보호를 참조하세요.

AWS Systems Manager의 I자격 증명 및 액세스 관리

AWS Identity and Access Management(IAM)는 관리자가 AWS 리소스에 대한 액세스를 안전하게 통제할 수 있도록 지원하는 AWS 서비스입니다. IAM 관리자는 어떤 사용자가 Systems Manager 리소스를 사용할 수 있는 인증(로그인) 및 권한(권한 있음)을 받을 수 있는지 제어합니다. IAM은 추가 비용 없이 사용할 수 있는 AWS 서비스입니다.

주제

고객

AWS Identity and Access Management(IAM)을 사용하는 방법은 Systems Manager에서 수행하는 작업에 따라 달라집니다.

서비스 사용자 - Systems Manager 서비스를 사용하여 작업을 수행하는 경우 필요한 보안 인증과 권한을 관리자가 제공합니다. 더 많은 Systems Manager 기능을 사용하여 작업을 수행하게 되면 추가 권한이 필요할 수 있습니다. 액세스 권한 관리 방식을 이해하면 적절한 권한을 관리자에게 요청할 수 있습니다. Systems Manager의 기능에 액세스할 수 없는 경우 AWS Systems Manager ID 및 액세스 문제 해결 섹션을 참조하세요.

서비스 관리자 - 회사에서 Systems Manager 리소스를 책임지고 있는 담당자라면 Systems Manager에 대한 전체 액세스 권한을 가지고 있을 것입니다. 서비스 관리자는 서비스 사용자가 액세스해야 하는 Systems Manager 기능과 리소스를 결정합니다. 그런 다음, IAM 관리자에게 요청을 제출하여 서비스 사용자의 권한을 변경해야 합니다. 이 페이지의 정보를 검토하여 IAM의 기본 개념을 이해합니다. 회사가 Systems Manager에서 IAM을 사용하는 방법에 대해 자세히 알아보려면 AWS Systems Manager에서 IAM을 사용하는 방식 섹션을 참조하세요.

IAM 관리자 - IAM 관리자라면 Systems Manager에 대한 액세스 권한 관리 정책 작성 방법을 자세히 알고 싶을 것입니다. IAM에서 사용할 수 있는 Systems Manager 자격 증명 기반 정책 예제를 보려면 AWS Systems Manager 자격 증명 기반 정책 예시 섹션을 참조하세요.

보안 인증을 통한 인증

인증은 ID 보안 인증을 사용하여 AWS에 로그인하는 방식입니다. AWS 계정 루트 사용자나 IAM 사용자로 또는 IAM 역할을 수임하여 인증(AWS에 로그인)되어야 합니다.

자격 증명 소스 AWS IAM Identity Center를 통해 제공된 보안 인증 정보를 사용하여 연동 ID로 AWS에 로그인할 수 있습니다. (IAM Identity Center) 사용자, 회사의 Single Sign-On 인증, Google 또는

Facebook 자격 증명이 연동 ID의 예입니다. 연동 ID로 로그인할 때 관리자가 이전에 IAM 역할을 사용하여 자격 연동을 설정했습니다. 연동을 사용하여 AWS에 액세스하면 간접적으로 역할을 수임합니다.

사용자 유형에 따라 AWS Management Console 또는AWS 액세스 포털에 로그인할 수 있습니다. AWS에 로그인하는 방법에 대한 자세한 내용은 AWS 로그인 사용 설명서의 AWS 계정에 로그인하는 방법을 참조하세요.

AWS에 프로그래밍 방식으로 액세스하는 경우, AWS에서는 보안 인증 정보를 사용하여 요청에 암호화 방식으로 서명할 수 있는 소프트웨어 개발 키트(SDK) 및 명령줄 인터페이스(CLI)를 제공합니다. AWS 도구를 사용하지 않는 경우 요청에 직접 서명해야 합니다. 권장 방법을 사용하여 요청에 직접 서명하는 방법에 대한 자세한 내용은 IAM 사용자 설명서의 AWS API 요청에 서명을 참조하세요.

사용하는 인증 방법에 상관없이 추가 보안 정보를 제공해야 할 수도 있습니다. 예를 들어, AWS은 다중 인증(MFA)을 사용하여 계정의 보안을 강화하는 것을 권장합니다. 자세한 내용은 AWS IAM Identity Center 사용 설명서의 다중 인증 및 IAM 사용자 설명서의 AWS에서 다중 인증(MFA) 사용을 참조하세요.

AWS 계정 루트 사용자

AWS 계정를 생성할 때는 해당 계정의 모든 AWS 서비스 및 리소스에 대한 완전한 액세스 권한이 있는 단일 로그인 ID로 시작합니다. 이 ID는 AWS 계정 루트 사용자라고 하며, 계정을 생성할 때 사용한 이메일 주소와 암호로 로그인하여 액세스합니다. 일상적인 작업에 루트 사용자를 사용하지 않을 것을 강력히 권장합니다. 루트 사용자 보안 인증 정보를 보호하고 루트 사용자만 수행할 수 있는 작업을수행하는 데 사용합니다. 루트 사용자로 로그인해야 하는 전체 작업 목록은 IAM 사용 설명서의 루트 사용자 보안 인증이 필요한 태스크를 참조하세요.

IAM 사용자 및 그룹

IAM 사용자는 단일 개인 또는 애플리케이션에 대한 특정 권한을 가지고 있는 AWS 계정에 속하는 ID입니다. 가능하면 암호 및 액세스 키와 같은 장기 자격 증명이 있는 IAM 사용자를 생성하는 대신 임시 자격 증명을 사용하는 것이 좋습니다. 하지만 IAM 사용자의 장기 자격 증명이 필요한 특정 사용 사례가 있는 경우 액세스 키를 교체하는 것이 좋습니다. 자세한 내용은 IAM 사용자 설명서의 장기 보안 인증이 필요한 사용 사례의 경우 정기적으로 액세스 키 교체를 참조하세요.

IAM 그룹은 IAM 사용자 컬렉션을 지정하는 자격 증명입니다. 사용자는 그룹으로 로그인할 수 없습니다. 그룹을 사용하여 여러 사용자의 권한을 한 번에 지정할 수 있습니다. 그룹을 사용하면 대규모 사용자 집합의 권한을 더 쉽게 관리할 수 있습니다. 예를 들어 IAMAdmins라는 그룹이 있고 이 그룹에 IAM 리소스를 관리할 권한을 부여할 수 있습니다.

사용자는 역할과 다릅니다. 사용자는 한 사람 또는 애플리케이션과 고유하게 연결되지만, 역할은 해당 역할이 필요한 사람이라면 누구나 수임할 수 있습니다. 사용자는 영구적인 보안 인증 정보를 가지고 있지만, 역할은 임시 보안 인증 정보만 제공합니다. 자세한 정보는 IAM 사용자 설명서의 IAM 사용자를 만들어야 하는 경우(역할이 아님)를 참조하세요.

IAM 역할

IAM 역할은 특정 권한을 가지고 있는 AWS 계정 내의 ID입니다. IAM 사용자와 유사하지만, 특정 개인과 연결되지 않습니다. 역할 전환하여 AWS Management Console에서 IAM 역할을 임시로 수임할 수 있습니다. AWS CLI 또는 AWS API 작업을 호출하거나 사용자 지정 URL을 사용하여 역할을 수임할 수 있습니다. 역할 사용 방법에 대한 자세한 정보는 IAM 사용자 설명서의 IAM 역할 사용을 참조하세요.

임시 보안 인증 정보가 있는 IAM 역할은 다음과 같은 상황에서 유용합니다.

- 연동 사용자 액세스 - 연동 자격 증명에 권한을 부여하려면 역할을 생성하고 해당 역할의 권한을 정의합니다. 연동 자격 증명이 인증되면 역할이 연결되고 역할에 정의된 권한이 부여됩니다. 연동 역할에 대한 자세한 내용은 IAM 사용자 설명서의 서드 파티 자격 증명 공급자의 역할 만들기 부분을 참조하세요. IAM Identity Center를 사용하는 경우 권한 집합을 구성합니다. 인증 후 자격 증명이 액세스할 수 있는 항목을 제어하기 위해 IAM Identity Center는 권한 세트를 IAM의 역할과 연관짓습니다. 권한 세트에 대한 자세한 내용은 AWS IAM Identity Center 사용 설명서의 권한 세트를 참조하세요.
- 임시 IAM 사용자 권한 - IAM 사용자 또는 역할은 IAM 역할을 수임하여 특정 작업에 대한 다양한 권한을 임시로 받을 수 있습니다.
- 계정 간 액세스 - IAM 역할을 사용하여 다른 계정의 사용자(신뢰할 수 있는 보안 주체)가 내 계정의 리소스에 액세스하도록 허용할 수 있습니다. 역할은 계정 간 액세스를 부여하는 기본적인 방법입니다. 그러나 일부 AWS 서비스를 사용하면 역할을(역할을 프록시로 사용하는 대신) 리소스에 정책을 직접 연결할 수 있습니다. 계정 간 액세스를 위한 역할과 리소스 기반 정책의 차이점을 알아보려면 IAM 사용자 설명서의 IAM 역할과 리소스 기반 정책의 차이를 참조하세요.
- 교차 서비스 액세스 – 일부 AWS 서비스는 다른 AWS 서비스의 특성을 사용합니다. 예를 들어 서비스에서 직접 호출을 수행하면 일반적으로 해당 서비스는 EC2에서 애플리케이션을 실행하거나 S3에 개체를 저장합니다. 서비스는 호출하는 보안 주체의 권한을 사용하거나, 서비스 역할을 사용하거나, 또는 서비스 연결 역할을 사용하여 이 작업을 수행할 수 있습니다.
  - 전달 액세스 세션(FAS) - IAM 사용자 또는 역할을 사용하여 AWS에서 작업을 수행하는 사람은 보안 주체로 간주됩니다. 일부 서비스를 사용하는 경우 다른 서비스에서 다른 작업을 시작하는 작업을 수행할 수 있습니다. FAS는 AWS 서비스를 직접 호출하는 보안 주체의 권한과 요청하는 AWS 서비스를 함께 사용하여 다운스트림 서비스에 대한 요청을 수행합니다. FAS 요청은 서비스에서 완료를 위해 다른 AWS 서비스 또는 리소스와의 상호 작용이 필요한 요청을 받은 경우에만 이루어

집니다. 이 경우 두 작업을 모두 수행할 수 있는 권한이 있어야 합니다. FAS 요청 시 정책 세부 정보는 전달 액세스 세션을 참조하세요.

- 서비스 역할 - 서비스 역할은 서비스가 사용자를 대신하여 태스크를 수행하기 위해 맡는 IAM 역할입니다. IAM 관리자는 IAM 내에서 서비스 역할을 생성, 수정 및 삭제할 수 있습니다. 자세한 정보는 IAM 사용자 설명서의 AWS 서비스에 대한 권한을 위임할 역할 생성을 참조하세요.
- 서비스 연결 역할 - 서비스 연결 역할은 AWS 서비스에 연결된 서비스 역할의 한 유형입니다. 서비스는 사용자를 대신하여 작업을 수행하기 위해 역할을 수임할 수 있습니다. 서비스 링크 역할은 AWS 계정에 나타나고, 서비스가 소유합니다. IAM 관리자는 서비스 연결 역할의 권한을 볼 수 있지만 편집할 수는 없습니다.

- Amazon EC2에서 실행 중인 애플리케이션 – IAM 역할을 사용하여 EC2 인스턴스에서 실행되고 AWS CLI 또는 AWS API 요청을 수행하는 애플리케이션의 임시 보안 인증을 관리할 수 있습니다. 이는 EC2 인스턴스 내에 액세스 키를 저장할 때 권장되는 방법입니다. EC2 인스턴스에 AWS역할을 할당하고 해당 역할을 모든 애플리케이션에서 사용할 수 있도록 하려면 인스턴스에 연결된 인스턴스 프로파일을 생성합니다. 인스턴스 프로파일에는 역할이 포함되어 있으며 EC2 인스턴스에서 실행되는 프로그램이 임시 보안 인증을 얻을 수 있습니다. 자세한 정보는 IAM 사용자 설명서의 IAM 역할을 사용하여 Amazon EC2 인스턴스에서 실행되는 애플리케이션에 권한 부여를 참조하세요.

IAM 역할을 사용할지 또는 IAM 사용자를 사용할지를 알아보려면 IAM 사용자 설명서의 IAM 역할(사용자 대신)을 생성하는 경우를 참조하세요.

정책을 사용한 액세스 관리

정책을 생성하고 AWS 자격 증명 또는 리소스에 연결하여 AWS 내 액세스를 제어합니다. 정책은 ID 또는 리소스와 연결될 때 해당 권한을 정의하는 AWS의 객체입니다. AWS는 보안 주체(사용자, 루트 사용자 또는 역할 세션)가 요청을 보낼 때 이러한 정책을 평가합니다. 정책에서 권한은 요청이 허용되거나 거부되는 지를 결정합니다. 대부분의 정책은 AWS에 JSON 문서로 저장됩니다. JSON 정책 문서의 구조와 콘텐츠에 대한 자세한 정보는 IAM 사용자 설명서의 JSON 정책 개요를 참조하세요.

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는 지를 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는 지를 지정할 수 있습니다.

기본적으로, 사용자와 역할에는 어떠한 권한도 없습니다. 사용자에게 사용자가 필요한 리소스에서 작업을 수행할 권한을 부여하려면 IAM 관리자가 IAM 정책을 생성하면 됩니다. 그런 다음 관리자가 IAM 정책을 역할에 추가하고, 사용자가 역할을 수임할 수 있습니다.

IAM 정책은 작업을 수행하기 위해 사용하는 방법과 상관없이 작업에 대한 권한을 정의합니다. 예를 들어, `iam:GetRole` 작업을 허용하는 정책이 있다고 가정합니다. 해당 정책이 있는 사용자는 AWS Management Console, AWS CLI또는 AWSAPI에서 역할 정보를 가져올 수 있습니다.

ID 기반 정책

ID 기반 정책은 IAM 사용자, 사용자 그룹 또는 역할과 같은 자격 증명에 연결할 수 있는 JSON 권한 정책 문서입니다. 이러한 정책은 사용자와 역할이 어떤 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는 지를 제어합니다. 자격 증명 기반 정책을 생성하는 방법을 알아보려면 IAM 사용자 설명서의 IAM 정책 생성을 참조하세요.

ID 기반 정책은 인라인 정책 또는 관리형 정책으로 한층 더 분류할 수 있습니다. 인라인 정책은 단일 사용자, 그룹 또는 역할에 직접 포함됩니다. 관리형 정책은 AWS 계정에 속한 다수의 사용자, 그룹 및 역할에 독립적으로 추가할 수 있는 정책입니다. 관리형 정책에는 AWS관리형 정책과 고객 관리형 정책이 포함되어 있습니다. 관리형 정책 또는 인라인 정책을 선택하는 방법을 알아보려면 IAM 사용자 설명서의 관리형 정책과 인라인 정책 사이의 선택을 참조하세요.

AWS의 Systems Manager 관리형 정책에 대한 자세한 내용은 AWS Systems Manager 관리형 정책 섹션을 참조하세요.

리소스 기반 정책

리소스 기반 정책은 리소스에 연결하는 JSON 정책 설명서입니다. 리소스 기반 정책의 예는 IAM 역할 신뢰 정책과 Amazon S3 버킷 정책입니다. 리소스 기반 정책을 지원하는 서비스에서 서비스 관리자는 이러한 정책을 사용하여 특정 리소스에 대한 액세스를 통제할 수 있습니다. 정책이 연결된 리소스의 경우 정책은 지정된 보안 주체가 해당 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는 지를 정의합니다. 리소스 기반 정책에서 보안 주체를 지정해야 합니다. 보안 주체에는 계정, 사용자, 역할, 연동 사용자 또는 AWS 서비스가 포함될 수 있습니다.

리소스 기반 정책은 해당 서비스에 있는 인라인 정책입니다. 리소스 기반 정책에서는 IAM의 AWS 관리형 정책을 사용할 수 없습니다.

액세스 제어 목록(ACLs)

액세스 제어 목록(ACLs)은 어떤 보안 주체(계정 멤버, 사용자 또는 역할)가 리소스에 액세스할 수 있는 권한을 가지고 있는 지를 제어합니다. ACLs는 JSON 정책 문서 형식을 사용하지 않지만 리소스 기반 정책과 유사합니다.

Amazon S3, AWS WAF 및 Amazon VPC는 ACL을 지원하는 대표적인 서비스입니다. ACL에 대해 자세히 알아보려면 Amazon Simple Storage Service 개발자 안내서의 액세스 제어 목록(ACL) 개요를 참조하세요.

기타 정책 타입

AWS는 비교적 일반적이지 않은 추가 정책 유형을 지원합니다. 이러한 정책 타입은 더 일반적인 정책 타입에 따라 사용자에게 부여되는 최대 권한을 설정할 수 있습니다.

- 권한 경계 – 권한 경계는 자격 증명 기반 정책에 따라 IAM 엔터티(IAM 사용자 또는 역할)에 부여할 수 있는 최대 권한을 설정하는 고급 기능입니다. 개체에 대한 권한 경계를 설정할 수 있습니다. 그 결과로 얻는 권한은 엔터티의 자격 증명 기반 정책과 그 권한 경계의 교집합입니다. `Principal` 필드에서 사용자나 역할을 보안 주체로 지정하는 리소스 기반 정책은 권한 경계를 통해 제한되지 않습니다. 이러한 정책 중 하나에 포함된 명시적 거부는 허용을 재정의합니다. 권한 경계에 대한 자세한 정보는 IAM 사용자 설명서의 IAM 엔터티에 대한 권한 경계를 참조하세요.
- 서비스 제어 정책(SCPs) – SCPs는 AWS Organizations에서 조직 또는 조직 단위(OU)에 최대 권한을 지정하는 JSON 정책입니다. AWS Organizations는 기업이 소유하는 여러 개의 AWS 계정를 그룹화하고 중앙에서 관리하기 위한 서비스입니다. 조직에서 모든 특성을 활성화할 경우 서비스 제어 정책(SCP)을 임의의 또는 모든 계정에 적용할 수 있습니다. SCP는 각 AWS 계정 루트 사용자을 비롯하여 멤버 계정의 엔터티에 대한 권한을 제한합니다. 조직 및 SCPs에 대한 자세한 정보는 AWS Organizations사용 설명서의 SCP 작동 방식을 참조하세요.
- 세션 정책 – 세션 정책은 역할 또는 연동 사용자에 대해 임시 세션을 프로그래밍 방식으로 생성할 때 파라미터로 전달하는 고급 정책입니다. 결과적으로 얻는 세션의 권한은 사용자 또는 역할 자격 증명 기반 정책의 교차 및 세션 정책입니다. 또한 권한을 리소스 기반 정책에서 가져올 수도 있습니다. 이러한 정책 중 하나에 포함된 명시적 거부는 허용을 재정의합니다. 자세한 정보는 IAM 사용자 설명서의 세션 정책을 참조하세요.

여러 정책 타입

여러 정책 타입이 요청에 적용되는 경우 결과 권한은 이해하기가 더 복잡합니다. 여러 정책 유형이 관련될 때 AWS가 요청을 허용할지를 결정하는 방법을 알아보려면 IAM 사용 설명서의 정책 평가 로직을 참조하세요.

AWS Systems Manager에서 IAM을 사용하는 방식

AWS Identity and Access Management(IAM)를 사용하여 AWS Systems Manager에 대한 액세스를 관리하려면 먼저 어떤 IAM 기능을 Systems Manager에 사용할 수 있는지를 이해해야 합니다. Systems Manager 및 기타 AWS 서비스에서 IAM을 사용하는 방법을 전체적으로 알아보려면 IAM 사용 설명서의 IAM으로 작업하는 AWS 서비스을(를) 참조하세요.

주제

- Systems Manager ID 기반 정책
- Systems Manager 리소스 기반 정책
- Systems Manager 태그 기반 인증
- Systems Manager IAM 역할

Systems Manager ID 기반 정책

IAM 자격 증명 기반 정책을 사용하면 허용되거나 거부되는 작업 및 리소스를 지정할 수 있을 뿐 아니라 작업이 허용되거나 거부되는 조건도 지정할 수 있습니다. Systems Manager는 특정 작업, 리소스 및 조건 키를 지원합니다. JSON 정책에서 사용하는 모든 요소에 대해 알아보려면 IAM 사용 설명서의 IAM JSON 정책 요소 참조를 참조하십시오.

작업

관리자는 AWSJSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지를 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

JSON 정책의 `Action`요소는 정책에서 액세스를 허용하거나 거부하는 데 사용할 수 있는 태스크를 설명합니다. 일반적으로 정책 작업의 이름은 연결된 AWSAPI 작업의 이름과 동일합니다. 일치하는 API 작업이 없는 권한 전용 작업 같은 몇 가지 예외도 있습니다. 정책에서 여러 작업이 필요한 몇 가지 작업도 있습니다. 이러한 추가 작업을 일컬어 종속 작업이라고 합니다.

연결된 작업을 수행할 수 있는 권한을 부여하기 위한 정책에 작업을 포함하십시오.

Systems Manager의 정책 작업은 작업 앞에 `ssm:` 접두사를 사용합니다. 예를 들어 Systems Manager `PutParameter` API 작업으로 Systems Manager 파라미터(SSM 파라미터)를 생성할 수 있는 권한을 부여하려면 해당 정책에 `ssm:PutParameter` 작업을 포함합니다. 정책 문에는 `Action` 또는 `NotAction` 요소가 포함되어야 합니다. Systems Manager는 이 서비스로 수행할 수 있는 작업을 설명하는 고유한 작업 집합을 정의합니다.

명령문 하나에 여러 작업을 지정하려면 다음과 같이 쉼표로 구분합니다.

```
"Action": [
      "ssm:action1",
      "ssm:action2"
```

> **Note**
>
> AWS Systems Manager의 다음 기능에서는 작업 앞에 다른 접두사를 사용합니다.

- AWS AppConfig는 작업 앞에 `appconfig:` 접두사를 사용합니다.
- Incident Manager는 작업 앞에 접두사 `ssm-incidents:` 또는 `ssm-contacts:`를 사용합니다.
- Systems Manager GUI Connect는 작업 앞에 `ssm-guiconnect` 접두사를 사용합니다.

와일드카드(*)를 사용하여 여러 작업을 지정할 수 있습니다. 예를 들어, `Describe`라는 단어로 시작하는 모든 태스크를 지정하려면 다음 태스크를 포함합니다.

```
"Action": "ssm:Describe*"
```

Systems Manager 작업 목록을 보려면 Service Authorization Reference의 Actions Defined by AWS Systems Manager를 참조하세요.

리소스

관리자는 AWSJSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지를 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지 지정할 수 있습니다.

`Resource` JSON 정책 요소는 작업이 적용되는 하나 이상의 개체를 지정합니다. 문장에는 `Resource`또는 `NotResource`요소가 반드시 추가되어야 합니다. 모범 사례에 따라 Amazon 리소스 이름(ARN)을 사용하여 리소스를 지정합니다. 리소스 수준 권한이라고 하는 특정 리소스 타입을 지원하는 작업에 대해 이 작업을 수행할 수 있습니다.

작업 나열과 같이 리소스 수준 권한을 지원하지 않는 작업의 경우, 와일드카드(*)를 사용하여 해당 문이 모든 리소스에 적용됨을 나타냅니다.

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

예를 들어, Systems Manager 유지 관리 기간 리소스에는 다음과 같은 ARN 형식이 있습니다.

```
arn:aws:ssm:region:account-id:maintenancewindow/window-id
```

미국 동부(오하이오) 리전에서 명령문에 mw-0c50858d01EXAMPLE 유지 관리 기간을 지정하려면 다음과 유사한 ARN을 사용합니다.

```
"Resource": "arn:aws:ssm:us-east-2:123456789012:maintenancewindow/mw-0c50858d01EXAMPLE"
```

특정 계정에 속하는 모든 유지 관리 기간을 지정하려면 와일드카드(*)를 사용합니다.

```
"Resource": "arn:aws:ssm:region:123456789012:maintenancewindow/*"
```

Parameter Store API 작업의 경우 다음과 같이 AWS Identity and Access Management(IAM) 정책과 계층적 이름을 사용하여 계층 구조의 한 수준 내에 있는 모든 파라미터에 대한 액세스를 제공하거나 제한할 수 있습니다.

```
"Resource": "arn:aws:ssm:region:123456789012:parameter/Dev/ERP/Oracle/*"
```

리소스를 생성하기 위한 작업과 같은 일부 Systems Manager 작업은 특정 리소스에서 수행할 수 없습니다. 이러한 경우, 와일드카드(*)를 사용해야 합니다.

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

일부 Systems Manager API 작업에서는 여러 리소스 사용을 허용합니다. 명령문 하나에 여러 리소스를 지정하려면 다음과 같이 각 ARN을 쉼표로 구분합니다.

```
"Resource": [
      "resource1",
      "resource2"
```

> **Note**
>
> 대부분의 AWS 서비스은(는) 콜론(:) 또는 슬래시(/)를 ARN에서 동일한 문자로 처리합니다. 하지만 Systems Manager에서는 리소스 패턴 및 규칙에 정확히 일치하는 항목이 있어야 합니다. 이벤트 패턴을 만들 때 리소스의 ARN이 일치하도록 정확한 ARN 문자를 사용해야 합니다.

아래 표에서는 Systems Manager에서 지원하는 리소스 유형의 ARN 형식을 설명합니다.

> **Note**
>
> ARN 형식에는 다음과 같은 예외가 있습니다.

- AWS Systems Manager의 다음 기능에서는 작업 앞에 다른 접두사를 사용합니다.
  - AWS AppConfig는 작업 앞에 `appconfig:` 접두사를 사용합니다.
  - Incident Manager는 작업 앞에 접두사 `ssm-incidents:` 또는 `ssm-contacts:`를 사용합니다.
  - Systems Manager GUI Connect는 작업 앞에 `ssm-guiconnect` 접두사를 사용합니다.
- Amazon 소유의 문서 및 자동화 정의 리소스뿐 아니라 Amazon 및 타사 소스에서 제공하는 공용 파라미터에는 ARN 형식의 계정 ID가 포함되지 않습니다. 예:
  - SSM 문서 `AWS-RunPatchBaseline`:

    ```
    arn:aws:ssm:us-east-2:::document/AWS-RunPatchBaseline
    ```
  - Automation 런북 `AWS-ConfigureMaintenanceWindows`:

    ```
    arn:aws:ssm:us-east-2:::automation-definition/AWS-
    ConfigureMaintenanceWindows
    ```
  - 공용 파라미터 `/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
    ```

  이러한 리소스 유형에 대한 자세한 내용은 다음 주제를 참조하세요.
  - 문서 작업
  - 자동화 실행
  - 퍼블릭 파라미터 작업

리소스 유형	ARN 형식
애플리케이션(AWS AppConfig)	arn:aws:appconfig:*region*:*account-id* :application/*application-id*
연결	arn:aws:ssm:*region*:*account-id* :association/*association-id*
자동화 실행	arn:aws:ssm:*region*:*account-id* :automation-execution/*automation-execution-id*

리소스 유형	ARN 형식
자동화 정의(버전 하위 리소스 사용)	arn:aws:ssm:*region*:*account-id* :automation-definition/*automation-definition-id* :*version-id* [1]
구성 프로파일(AWS AppConfig)	arn:aws:appconfig:*region*:*account-id* :application/*application-id* /configurationprofile/*configurationprofile-id*
연락처(Incident Manager)	arn:aws:ssm-contacts:*region*:*account-id* :contact/*contact-alias*
배포 전략(AWS AppConfig)	arn:aws:appconfig:*region*:*account-id* :deploymentstrategy/*deploymentstrategy-id*
문서	arn:aws:ssm:*region*:*account-id* :document/*document-name*
환경(AWS AppConfig)	arn:aws:appconfig:*region*:*account-id* :application/*application-id* /environment/*environment-id*
인시던트	arn:aws:ssm-incidents:*region*:*account-id* :incident-record/*response-plan-name* /*incident-id*
유지보수 윈도우	arn:aws:ssm:*region*:*account-id* :maintenancewindow/*window-id*
관리형 노드	arn:aws:ssm:*region*:*account-id* :managed-instance/*managed-node-id*
관리형 노드 인벤토리	arn:aws:ssm:*region*:*account-id* :managed-instance-inventory/*managed-node-id*
OpsItem	arn:aws:ssm:*region*:*account-id* :opsitem/*OpsItem-id*

리소스 유형	ARN 형식
파라미터	단일 수준 파라미터: • arn:aws:ssm:*region*:*account-id* :parameter/*parameter-name*/ 계층 구조적 구성으로 이름이 지정된 파라미터: • arn:aws:ssm:*region*:*account-id* :parameter/*parameter-name-root* /*level-2*/*level-3*/*level-4*/*level-5* [2]
패치 기준	arn:aws:ssm:*region*:*account-id* :patchbaseline/*patch-baseline-id*
대응 계획	arn:aws:ssm-incidents:*region*:*account-id* :response-plan/*response-plan-name*
세션	arn:aws:ssm:*region*:*account-id* :session/*session-id* [3]
모든 Systems Manager 리소스	arn:aws:ssm:*
지정한 AWS 리전에서 지정한 AWS 계정가 소유한 모든 Systems Manager 리소스	arn:aws:ssm:*region*:*account-id* :*

[1]
자동화 정의의 경우, Systems Manager는 두 번째 수준 리소스인 버전 ID를 지원합니다. AWS에서는 이러한 두 번째 수준 리소스를 하위 리소스라고 합니다. 자동화 정의 리소스에 대해 버전 하위 리소스를 지정하면 자동화 정의의 특정 버전에 대한 액세스를 제공할 수 있습니다. 예를 들어, 자동화 정의의 최신 버전만 노드 관리에 사용되도록 하고자 할 수 있습니다.

[2]
파라미터를 구성하고 관리하려면 계층적 구조로 파라미터의 이름을 생성합니다. 계층 구조적 구성을

사용하면, 슬래시를 사용하여 정의한 경로를 파라미터 이름에 포함할 수 있습니다. 최대 15개의 수준으로 파라미터 리소스의 이름을 지정할 수 있습니다. 현재 환경의 기존 계층 구조를 반영하는 계층 구조를 생성하는 것이 좋습니다. 자세한 내용은 Systems Manager 파라미터 생성 단원을 참조하십시오.

3

대부분의 경우 세션 ID는 세션을 시작한 계정 사용자의 ID와 영숫자 접미사로 구성됩니다. 예:

```
arn:aws:us-east-2:111122223333:session/JohnDoe-1a2b3c4sEXAMPLE
```

단, 사용자 ID를 사용할 수 없는 경우 ARN은 다음과 같은 방법으로 구성됩니다.

```
arn:aws:us-east-2:111122223333:session/session-1a2b3c4sEXAMPLE
```

ARN 형식에 대한 자세한 내용은 Amazon Web Services 일반 참조의 Amazon 리소스 이름(ARN)을 참조하세요.

Systems Manager 리소스 유형 및 해당 ARN의 목록을 보려면 Service Authorization Reference의 Resources Defined by AWS Systems Manager를 참조하세요. 각 리소스의 ARN을 지정할 수 있는 작업을 알아보려면 AWS Systems Manager가 정의한 작업을 참조하세요.

Systems Manager에 사용되는 조건 키

관리자는 AWSJSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지를 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지 지정할 수 있습니다.

`Condition` 요소(또는 `Condition` 블록)를 사용하면 정책이 발효되는 조건을 지정할 수 있습니다. `Condition` 요소는 옵션입니다. 같거나 작음과 같은 조건 연산자를 사용하여 정책의 조건을 요청의 값과 일치시키는 조건식을 생성할 수 있습니다.

한 문에서 여러 `Condition`요소를 지정하거나 단일 `Condition`요소에서 여러 키를 지정하는 경우 AWS는 논리적 `AND`태스크를 사용하여 평가합니다. 단일 조건 키의 여러 값을 지정하는 경우 AWS는 논리적 `OR`태스크를 사용하여 조건을 평가합니다. 명문의 권한을 부여하기 전에 모든 조건을 충족해야 합니다.

조건을 지정할 때 자리 표시자 변수를 사용할 수도 있습니다. 예를 들어, IAM 사용자에게 IAM 사용자 이름으로 태그가 지정된 경우에만 리소스에 액세스할 수 있는 권한을 부여할 수 있습니다. 자세한 정보는 IAM 사용 설명서의 IAM 정책 요소: 변수 및 태그를 참조하세요.

AWS는 전역 조건 키와 서비스별 조건 키를 지원합니다. 모든 AWS 전역 조건 키를 보려면 IAM 사용 설명서의 AWS 전역 조건 컨텍스트 키를 참조하세요.

Systems Manager 조건 키 목록을 보려면 Service Authorization Reference의 Condition Keys for AWS Systems Manager를 참조하세요. 조건 키를 사용할 수 있는 작업과 리소스를 알아보려면 AWS Systems Manager가 정의한 작업 섹션을 참조하세요.

`ssm:resourceTag/*` 조건 키 사용에 대한 자세한 내용은 다음 주제를 참조하십시오.

- SSM Agent를 통한 루트 수준 명령에 대한 액세스 제한
- 태그를 기반으로 Run Command 액세스 제한
- 인스턴스 태그를 기준으로 세션 액세스 제한

`ssm:Recursive` 및 `ssm:Overwrite` 조건 키 사용에 관한 자세한 내용은 파라미터 계층 구조 작업 섹션을 참조하세요.

예제

Systems Manager 자격 증명 기반 정책의 예를 보려면 AWS Systems Manager 자격 증명 기반 정책 예시을 참조하세요.

Systems Manager 리소스 기반 정책

Amazon Simple Storage Service(Amazon S3)와 같은 다른 AWS 서비스도 리소스 기반 권한 정책을 지원합니다. 예를 들어, 권한 정책을 S3 버킷에 연결하여 해당 버킷에 대한 액세스 권한을 관리할 수 있습니다.

Systems Manager에서는 리소스 기반 정책을 지원하지 않습니다.

Systems Manager 태그 기반 인증

태그를 Systems Manager 리소스에 연결하거나 요청을 통해 태그를 Systems Manager에 전달할 수 있습니다. 태그를 기반으로 액세스를 제어하려면 `ssm:resourceTag/`*`key-name`*, `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`* 또는 `aws:TagKeys` 조건 키를 사용하여 정책의 조건 요소에 태그 정보를 제공합니다. 다음 리소스 유형을 생성하거나 업데이트할 때 태그를 추가할 수 있습니다.

- 문서
- 관리형 노드
- 유지보수 윈도우

- 파라미터
- 패치 기준
- OpsItem

Systems Manager 리소스 태그 지정에 대한 자세한 내용은 Systems Manager 리소스에 태그 지정 섹션을 참조하세요.

리소스의 태그를 기반으로 리소스에 대한 액세스를 제한하는 자격 증명 기반 정책의 예제는 태그 기준으로 Systems Manager 문서 보기 섹션에서 확인할 수 있습니다.

Systems Manager IAM 역할

IAM 역할은 특정 권한을 가지고 있는 AWS 계정 계정 내 엔터티입니다.

Systems Manager에서 임시 보안 인증 사용

임시 보안 인증을 사용하여 페더레이션을 통해 로그인하거나, IAM 역할을 맡거나, 교차 계정 역할을 맡을 수 있습니다. AssumeRole 또는 GetFederationToken 같은 AWS Security Token Service(AWS STS) API 작업을 호출하여 임시 보안 자격 증명을 가져옵니다.

Systems Manager는 임시 보안 인증 사용을 지원합니다.

서비스 연결 역할

서비스 연결 역할을 사용하면 AWS 서비스이(가) 다른 서비스의 리소스에 액세스하여 사용자 대신 태스크를 완료할 수 있습니다. 서비스 연결 역할은 IAM 계정에 나열되고, 서비스가 소유합니다. 관리자는 서비스 연결 역할의 권한을 볼 수 있지만 편집할 수는 없습니다.

Systems Manager에서는 서비스 연결 역할을 지원합니다. Systems Manager 서비스 연결 역할을 생성 또는 관리하는 방법에 대한 자세한 내용은 Systems Manager에 서비스 연결 역할 사용을 참조하세요.

서비스 역할

이 기능을 사용하면 서비스가 사용자를 대신하여 서비스 역할을 수임할 수 있습니다. 이 역할을 사용하면 서비스가 다른 서비스의 리소스에 액세스해 사용자를 대신해 작업을 완료할 수 있습니다. 서비스 역할은 IAM 계정에 표시되고, 해당 계정이 소유합니다. 즉, 관리자가 이 역할에 대한 권한을 변경할 수 있습니다. 그러나 권한을 변경하면 서비스의 기능이 손상될 수 있습니다.

Systems Manager는 서비스 역할을 지원합니다.

Systems Manager에서 IAM 역할 선택

Systems Manager에서 관리형 노드와 상호 작용하려면 Systems Manager에서 사용자를 대신하여 노드에 액세스할 수 있는 역할을 선택해야 합니다. 이전에 서비스 역할 또는 서비스 연결 역할을 생성한 경우 Systems Manager는 선택할 수 있는 역할 목록을 제공합니다. 관리형 노드 시작 및 중지에 대한 액세스를 허용하는 역할을 선택하는 것이 중요합니다.

EC2 인스턴스에 액세스하려면 인스턴스 권한을 구성해야 합니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성을 참조하세요.

하이브리드 및 멀티클라우드에서 비 EC2 노드에 액세스하려는 AWS 계정에 필요한 역할은 IAM 서비스 역할입니다. 자세한 내용은 하이브리드 및 멀티클라우드 환경에서 Systems Manager에 필요한 IAM 서비스 역할 생성을 참조세요.

자동화 워크플로는 서비스 역할(또는 수임 역할) 컨텍스트에 따라 시작할 수 있습니다. 그러면 서비스가 사용자 대신 작업을 수행할 수 있습니다. 수임 역할이 지정되어 있지 않으면 Automation은 실행을 호출한 사용자 컨텍스트를 사용합니다. 그러나 특정 상황에서는 자동화를 위한 서비스 역할을 지정해야 합니다. 자세한 내용은 자동화에 대한 서비스 역할(수임 역할) 액세스 구성 단원을 참조하십시오.

AWS Systems Manager 관리형 정책

AWS는 AWS에서 생성하고 관리하는 독립형 IAM 정책을 제공하여 많은 일반 사용 사례를 처리합니다. 이러한 AWS 관리형 정책은 사용자가 필요한 권한을 조사할 필요가 없도록 일반 사용 사례에 필요한 권한을 부여합니다. Systems Manager 작업 및 리소스에 대한 권한을 허용하는 고유의 사용자 정의 IAM 정책을 생성할 수도 있습니다.

Systems Manager의 관리형 정책에 대한 자세한 내용은 AWS Systems Manager의 AWS 관리형 정책을 참조하세요.

관리형 정책에 대한 일반 정보는 IAM 사용자 설명서의 AWS 관리형 정책을 참조하세요.

AWS Systems Manager 자격 증명 기반 정책 예시

기본적으로 AWS Identity and Access Management(IAM) 엔터티(사용자 및 역할)에는 AWS Systems Manager 리소스를 생성하거나 수정할 수 있는 권한이 없습니다. 이 사용자와 역할은 Systems Manager 콘솔, AWS Command Line Interface(AWS CLI) 또는 AWS API를 사용하여 태스크를 수행할 수 없습니다. 관리자는 지정된 리소스에서 특정 API 태스크를 수행할 수 있는 권한을 사용자와 역할에게 부여하는 IAM 정책을 생성해야 합니다. 그런 다음 관리자는 해당 권한이 필요한 사용자 또는 그룹에 이러한 정책을 연결해야 합니다.

다음은 사용자가 미국 동부(오하이오)(us-east-2) AWS 리전에서 **MyDocument-**로 시작하는 이름의 문서를 삭제하는 것을 허용하는 권한 정책의 예입니다.

```
{
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "ssm:DeleteDocument"
      ],
      "Resource" : [
        "arn:aws:ssm:us-east-2:111122223333:document/MyDocument-*"
      ]
    }
  ]
}
```

이러한 예제 JSON 정책 문서를 사용하여 IAM 자격 증명 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서의 IAM 정책 생성을 참조하세요.

주제

- 정책 모범 사례
- Systems Manager 콘솔 사용
- 사용자가 자신이 권한을 볼 수 있도록 허용
- 교차 서비스 혼동된 대리인 방지
- 고객 관리형 정책 예
- 태그 기준으로 Systems Manager 문서 보기

정책 모범 사례

자격 증명 기반 정책에 따라 계정에서 사용자가 Systems Manager 리소스를 생성, 액세스 또는 삭제할 수 있는지 여부가 결정됩니다. 이 작업으로 인해 AWS 계정에 비용이 발생할 수 있습니다. ID 기반 정책을 생성하거나 편집할 때는 다음 지침과 권장 사항을 따릅니다.

- AWS 관리형 정책으로 시작하고 최소 권한을 향해 나아가기 - 사용자 및 워크로드에 권한 부여를 시작하려면 많은 일반 사용 사례에 대한 권한을 부여하는 AWS관리형 정책을 사용합니다. AWS 계정에서 사용할 수 있습니다. 사용 사례에 고유한 AWS고객 관리형 정책을 정의하여 권한을 줄이는 것

이 좋습니다. 자세한 정보는 IAM 사용자 설명서의 AWS 관리형 정책 또는 AWS 직무에 대한 관리형 정책을 참조하세요.

- 최소 권한 적용 – IAM 정책을 사용하여 권한을 설정하는 경우 작업을 수행하는 데 필요한 권한만 부여합니다. 이렇게 하려면 최소 권한으로 알려진 특정 조건에서 특정 리소스에 대해 수행할 수 있는 작업을 정의합니다. IAM을 사용하여 권한을 적용하는 방법에 대한 자세한 정보는 IAM 사용자 설명서에 있는 IAM의 정책 및 권한을 참조하세요.
- Use conditions in IAM policies to further restrict access(IAM 정책의 조건을 사용하여 액세스 추가 제한) – 정책에 조건을 추가하여 작업 및 리소스에 대한 액세스를 제한할 수 있습니다. 예를 들어 SSL을 사용하여 모든 요청을 전송해야 한다고 지정하는 정책 조건을 작성할 수 있습니다. AWS CloudFormation와 같이, 특정 AWS 서비스를 통해 사용되는 경우에만 서비스 작업에 대한 액세스 권한을 부여할 수도 있습니다. 자세한 정보는 IAM 사용자 설명서의 IAM JSON 정책 요소: 조건을 참조하세요.
- IAM Access Analyzer를 통해 IAM 정책을 검증하여 안전하고 기능적인 권한 보장 – IAM Access Analyzer에서는 IAM 정책 언어(JSON)와 IAM 모범 사례가 정책에서 준수되도록 신규 및 기존 정책을 검증합니다. IAM Access Analyzer는 100개 이상의 정책 확인 항목과 실행 가능한 권장 사항을 제공하여 안전하고 기능적인 정책을 생성하도록 돕습니다. 자세한 정보는 IAM 사용자 설명서의 IAM Access Analyzer 정책 검증을 참조하세요.
- 다중 인증(MFA) 필요 – AWS 계정계정에 IAM 사용자 또는 루트 사용자가 필요한 시나리오가 있는 경우 추가 보안을 위해 MFA를 설정합니다. API 작업을 직접 호출할 때 MFA가 필요하면 정책에 MFA 조건을 추가합니다. 자세한 정보는 IAM 사용자 설명서의 MFA 보호 API 액세스 구성을 참조하세요.

IAM의 모범 사례에 대한 자세한 내용은 IAM 사용 설명서의 IAM의 보안 모범 사례를 참조하세요.

Systems Manager 콘솔 사용

Systems Manager 콘솔에 액세스하려면 최소 권한 집합이 있어야 합니다. 이러한 권한은 AWS 계정에서 Systems Manager 리소스 및 기타 리소스에 대한 세부 정보를 나열하고 볼 수 있도록 허용해야 합니다.

Systems Manager 콘솔에서 Systems Manager를 완전히 사용하려면 다음 서비스의 권한이 있어야 합니다.

- AWS Systems Manager
- Amazon Elastic Compute Cloud(Amazon EC2)
- AWS Identity and Access Management (IAM)

다음 정책 문을 사용하여 필요한 권한을 부여할 수 있습니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:*",
                "ec2:describeInstances",
                "iam:ListRoles"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ssm.amazonaws.com"
                }
            }
        }
    ]
}
```

최소 필수 권한보다 더 제한적인 자격 증명 기반 정책을 생성하면 콘솔이 해당 정책을 사용하는 IAM 엔티티(사용자 또는 역할)에 대해 의도한 대로 작동하지 않습니다.

AWS CLI 또는 AWSAPI만 호출하는 사용자에게 최소 콘솔 권한을 허용할 필요가 없습니다. 그 대신, 수행하려는 API 작업과 일치하는 작업에만 액세스할 수 있도록 합니다.

사용자가 자신이 권한을 볼 수 있도록 허용

이 예시는 IAM 사용자가 자신의 사용자 자격 증명에 연결된 인라인 및 관리형 정책을 볼 수 있도록 허용하는 정책을 생성하는 방법을 보여줍니다. 이 정책에는 콘솔에서 또는 AWS CLI나 AWS API를 사용하여 프로그래밍 방식으로 이 작업을 완료할 수 있는 권한이 포함됩니다.

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

교차 서비스 혼동된 대리인 방지

혼동된 대리자 문제는 작업을 수행할 권한이 없는 엔터티가 권한이 더 많은 엔터티에게 작업을 수행하도록 강요할 수 있는 보안 문제입니다. AWS에서는 교차 서비스 가장으로 인해 혼동된 대리자 문제가 발생할 수 있습니다. 교차 서비스 가장은 한 서비스(직접 호출하는 서비스)가 다른 서비스(직접 호출되는 서비스)를 직접 호출할 때 발생할 수 있습니다. 직접 호출하는 서비스는 다른 고객의 리소스에 대해 액세스 권한이 없는 방식으로 작동하게 권한을 사용하도록 조작될 수 있습니다. 이를 방지하기 위해 AWS에서는 계정의 리소스에 대한 액세스 권한이 부여된 서비스 보안 주체를 사용하여 모든 서비스에 대한 데이터를 보호하는 데 도움이 되는 도구를 제공합니다.

`aws:SourceArn`이 리소스에 다른 서비스를 제공하는 권한을 제한하려면 리소스 정책에서 `aws:SourceAccount` 및 AWS Systems Manager 글로벌 조건 컨텍스트 키를 사용하는 것이 좋

습니다. aws:SourceArn 값에 S3 버킷의 Amazon 리소스 이름(ARN)과 같은 계정 ID가 포함되지 않은 경우 권한을 제한하려면 두 전역 조건 컨텍스트 키를 모두 사용해야 합니다. 두 전역 조건 컨텍스트 키와 계정을 포함한 aws:SourceArn 값을 모두 사용하는 경우, aws:SourceAccount 값 및 aws:SourceArn 값의 계정은 동일한 정책 명령문에서 사용할 경우 반드시 동일한 계정 ID를 사용해야 합니다. 하나의 리소스만 교차 서비스 액세스와 연결되도록 허용하려는 경우 aws:SourceArn을 사용하세요. 해당 계정의 모든 리소스가 교차 서비스 사용과 연결되도록 허용하려는 경우 aws:SourceAccount을(를) 사용하세요.

다음 섹션에서는 AWS Systems Manager 기능에 대한 예시 정책을 제공합니다.

하이브리드 정품 인증 정책 예제

하이브리드 정품 인증에 사용되는 서비스 역할의 경우 aws:SourceArn의 값은 AWS 계정의 ARN이어야 합니다. 하이브리드 정품 인증을 생성한 ARN에서 AWS 리전을 지정해야 합니다. 리소스의 전체 ARN을 모를 경우 또는 여러 리소스를 지정하는 경우, ARN의 알 수 없는 부분에 대해 와일드카드(*)를 포함한 aws:SourceArn 글로벌 조건 컨텍스트 키를 사용합니다. 예를 들면 arn:aws:ssm:*:*region*:*123456789012*:*입니다.

다음 예제에서는 미국 동부(오하이오) 리전(us-east-2) 에서 혼동된 대리자 문제를 방지하기 위해 자동화에 대한 aws:SourceArn 및 aws:SourceAccount 전역 조건 컨텍스트 키를 사용하는 방법을 보여줍니다.

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

```
}
```

리소스 데이터 동기화 정책 예제

Systems Manager 인벤토리, Explorer, 및 규정 준수를 사용하면 리소스 데이터 동기화를 생성하여 중앙 Amazon Simple Storage 서비스 버킷에서 운영 데이터(OpsData) 의 스토리지를 중앙 집중화할 수 있습니다. (선택 사항) AWS Key Management Service(AWS KMS)을 사용하여 리소스 데이터 동기화를 암호화하려면 다음 정책을 포함하는 새 키를 생성하거나 기존 키를 업데이트하고 이 정책을 키에 추가해야 합니다. 이 정책의 `aws:SourceArn` 및 `aws:SourceAccount` 조건 키는 혼동된 대리자 문제를 방지합니다. 다음은 예시 정책입니다.

```
{
    "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**
>
> 정책 예시의 ARN을 사용하면 시스템이 AWS Security Hub을 제외한 모든 소스에서 OpsData를 암호화할 수 있습니다. Security Hub 데이터를 암호화해야 하는 경우, 예를 들어 Explorer을

사용하여 Security Hub 데이터를 수집해야 하는 경우, 다음 ARN을 지정하는 추가 정책을 연결해야 합니다.

```
"aws:SourceArn": "arn:aws:ssm:*:account-id:role/
aws-service-role/opsdatasync.ssm.amazonaws.com/
AWSServiceRoleForSystemsManagerOpsDataSync"
```

고객 관리형 정책 예

사용자는 자체 AWS 계정에서 관리할 독립형 정책을 생성할 수 있습니다. 이를 고객 관리형 정책이라고 합니다. 이러한 정책은 AWS 계정에 속한 다수의 보안 주체 객체에 추가할 수 있습니다. 정책을 보안 주체 엔터티에 추가할 경우 정책에서 정의한 권한까지 엔터티에게 부여하게 됩니다. 자세한 내용은 IAM 사용 설명서의 고객 관리형 정책 예제를 참조하세요.

다음은 다양한 Systems Manager 작업에 대한 권한을 부여하는 사용자 정책의 예입니다. 이러한 정책을 사용하여 IAM 엔터티(사용자 및 역할)에 대한 Systems Manager 액세스를 제한할 수 있습니다. 이러한 정책은 Systems Manager API, AWS SDK 또는 AWS CLI에서 작업을 수행할 때 유효합니다. 콘솔을 사용하는 사용자는 콘솔에 특정한 추가 권한을 부여해야 합니다. 자세한 내용은 Systems Manager 콘솔 사용 단원을 참조하십시오.

> **Note**
>
> 모든 예에서는 미국 서부(오레곤) 리전(us-west-2)을 사용하며 가상의 계정 ID를 포함합니다. AWS 퍼블릭 문서(AWS-*로 시작하는 문서)의 경우 Amazon 리소스 이름(ARN)에 계정 ID를 지정해서는 안 됩니다.

예제

- 예제 1: 사용자가 단일 리전에서 Systems Manager 작업을 수행하도록 허용
- 예제 2: 사용자가 단일 리전의 문서를 나열하도록 허용

예제 1: 사용자가 단일 리전에서 Systems Manager 작업을 수행하도록 허용

다음 예에서는 미국 동부(오하이오) 리전(us-east-2)에서만 Systems Manager 작업을 수행하는 권한을 부여합니다.

```
{
```

```
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "ssm:*"
      ],
      "Resource" : [
        "arn:aws:ssm:us-east-2:aws-account-ID:*"
      ]
    }
  ]
}
```

예제 2: 사용자가 단일 리전의 문서를 나열하도록 허용

다음 예에서는 미국 동부(오하이오) 리전(us-east-2)에서 **Update**로 시작하는 모든 문서 이름을 나열할 수 있는 권한을 부여합니다.

```
{
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "ssm:ListDocuments"
      ],
      "Resource" : [
        "arn:aws:ssm:us-east-2:aws-account-ID:document/Update*"
      ]
    }
  ]
}
```

예제 3: 사용자가 특정 SSM 문서를 사용하여 특정 노드에서 명령을 실행하도록 허용

다음은 IAM 정책 예제에서는 사용자가 미국 동부(오하이오) 리전(us-east-2)에서 다음을 수행할 수 있도록 허용합니다.

- Systems Manager 문서(SSM 문서) 및 문서 버전을 나열합니다.
- 문서에 대한 세부 정보를 봅니다.
- 정책에 지정된 문서를 사용하여 명령을 보냅니다. 문서 이름은 다음 항목에 의해 결정됩니다.

```
arn:aws:ssm:us-east-2:aws-account-ID:document/Systems-Manager-document-name
```

- 명령을 세 개의 노드로 보냅니다. 노드는 두 번째 Resource 섹션의 다음 항목에 의해 결정됩니다.

```
"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"
```

- 명령이 전송된 후 해당 명령에 대한 세부 정보를 봅니다.
- AWS Systems Manager의 기능인 Automation에서 워크플로를 시작하고 중지합니다.
- Automation 워크플로에 대한 정보를 가져옵니다.

사용자에게 이 문서를 사용하여 액세스 권한을 갖는 노드에 대한 명령을 보낼 수 있는 권한을 부여하려는 경우 Resource 섹션에 다음 항목과 유사한 항목을 지정하고 다른 노드 항목을 제거할 수 있습니다. 다음 예제에서는 미국 동부(오하이오) 리전(us-east-2)을 사용합니다.

```
"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": [
                "*"
            ]
        }
```

```
    ]
}
```

태그 기준으로 Systems Manager 문서 보기

자격 증명 기반 정책의 조건을 사용하여 태그를 기반으로 Systems Manager 리소스에 대한 액세스를 제어할 수 있습니다. 이 예에서는 SSM 문서 보기를 허용하는 정책을 생성할 수 있는 방법을 보여줍니다. 하지만 문서 태그 `Owner`가 해당 사용자의 사용자 이름 값을 가지고 있는 경우에만 권한이 부여됩니다. 이 정책은 콘솔에서 이 작업을 완료하는 데 필요한 권한도 부여합니다.

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

이 정책을 계정의 사용자에게 연결할 수 있습니다. 이름이 `richard-roe`인 사용자가 Systems Manager 문서를 보려고 하면 문서에 `Owner=richard-roe` 또는 `owner=richard-roe` 태그를 지정해야 합니다. 그렇지 않으면 액세스가 거부됩니다. 조건 키 이름은 대/소문자를 구분하지 않기 때문에 태그 키 `Owner`는 `Owner` 및 `owner` 모두와 일치합니다. 자세한 정보는 IAM 사용 설명서의 IAM JSON 정책 요소: 조건을 참조하세요.

AWS Systems Manager의 AWS 관리형 정책

AWS 관리형 정책은 AWS에서 생성되고 관리되는 독립 실행형 정책입니다. AWS 관리형 정책은 사용자, 그룹 및 역할에 권한 할당을 시작할 수 있도록 많은 일반 사용 사례에 대한 권한을 제공하도록 설계되었습니다.

AWS 관리형 정책은 모든 AWS 고객이 사용할 수 있기 때문에 특정 사용 사례에 대해 최소 권한을 부여하지 않을 수 있습니다. 사용 사례에 고유한 고객 관리형 정책을 정의하여 권한을 줄이는 것이 좋습니다.

AWS 관리형 정책에서 정의한 권한은 변경할 수 없습니다. 만약 AWS가 AWS 관리형 정책에 정의된 권한을 업데이트할 경우 정책이 연결되어 있는 모든 보안 주체 엔터티(사용자, 그룹 및 역할)에도 업데이트가 적용됩니다. 새로운 AWS 서비스를 시작하거나 새로운 API 작업을 기존 서비스에 이용하는 경우 AWS가 AWS 관리형 정책을 업데이트할 가능성이 높습니다.

자세한 내용은 IAM 사용자 설명서의 AWS 관리형 정책을 참조하세요.

AWS 관리형 정책: AmazonSSMServiceRolePolicy

`AmazonSSMServiceRolePolicy`를 AWS Identity and Access Management(IAM) 엔터티에 연결할 수 없습니다. 이 정책은 AWS Systems Manager에서 사용자를 대신하여 작업을 수행할 수 있도록 서비스 연결 역할에 연결됩니다. 자세한 내용은 역할을 사용하여 인벤토리 수집 및 OpsData 보기 단원을 참조하십시오.

`AmazonSSMServiceRolePolicy`에서는 지정된 경우를 제외하고 Systems Manager에서 모든 관련 리소스(`"Resource": "*"`)에 대한 다음과 같은 작업을 완료하도록 허용합니다.

- `ssm:CancelCommand`
- `ssm:GetCommandInvocation`
- `ssm:ListCommandInvocations`
- `ssm:ListCommands`
- `ssm:SendCommand`
- `ssm:GetAutomationExecution`
- `ssm:GetParameters`
- `ssm:StartAutomationExecution`
- `ssm:StopAutomationExecution`

- ssm:ListTagsForResource
- ssm:GetCalendarState
- ssm:UpdateServiceSetting [1]
- ssm:GetServiceSetting [1]
- ec2:DescribeInstanceAttribute
- ec2:DescribeInstanceStatus
- ec2:DescribeInstances
- lambda:InvokeFunction [2]
- states:DescribeExecution [3]
- states:StartExecution [3]
- resource-groups:ListGroups
- resource-groups:ListGroupResources
- resource-groups:GetGroupQuery
- tag:GetResources
- config:SelectResourceConfig
- config:DescribeComplianceByConfigRule
- config:DescribeComplianceByResource
- config:DescribeRemediationConfigurations
- config:DescribeConfigurationRecorders
- cloudwatch:DescribeAlarms
- compute-optimizer:GetEC2InstanceRecommendations
- compute-optimizer:GetEnrollmentStatus
- support:DescribeTrustedAdvisorChecks
- support:DescribeTrustedAdvisorCheckSummaries
- support:DescribeTrustedAdvisorCheckResult
- support:DescribeCases
- iam:PassRole [4]
- cloudformation:DescribeStacks
- cloudformation:ListStackResources

- cloudformation:ListStackInstances [5]
- cloudformation:DescribeStackSetOperation [5]
- cloudformation:DeleteStackSet [5]
- cloudformation:DeleteStackInstances [6]
- events:PutRule [7]
- events:PutTargets [7]
- events:RemoveTargets [8]
- events:DeleteRule [8]
- events:DescribeRule
- securityhub:DescribeHub

[1] ssm:UpdateServiceSetting 및 ssm:GetServiceSetting 작업은 다음 리소스에 대해서만 허용되는 권한입니다.

```
arn:aws:ssm:*:*:servicesetting/ssm/opsitem/*
arn:aws:ssm:*:*:servicesetting/ssm/opsdata/*
```

[2] lambda:InvokeFunction 작업은 다음 리소스에 대해서만 허용되는 권한입니다.

```
arn:aws:lambda:*:*:function:SSM*
arn:aws:lambda:*:*:function:*:SSM*
```

[3] states: 작업은 다음 리소스에서만 허용되는 권한입니다.

```
arn:aws:states:*:*:stateMachine:SSM*
arn:aws:states:*:*:execution:SSM*
```

[4] iam:PassRole 작업은 Systems Manager 서비스에 대해서만 다음 조건에 의해 허용됩니다.

```
"Condition": {
   "StringEquals": {
      "iam:PassedToService": [
         "ssm.amazonaws.com"
      ]
   }
```

```
}
```

[5] `cloudformation:ListStackInstances`, `cloudformation:DescribeStackSetOperation` 및 `cloudformation:DeleteStackSet` 작업은 다음 리소스에서만 허용되는 권한입니다.

```
arn:aws:cloudformation:*:*:stackset/AWS-QuickSetup-SSM*:*
```

[6] `cloudformation:DeleteStackInstances` 작업은 다음 리소스에서만 허용되는 권한입니다.

```
arn:aws:cloudformation:*:*:stackset/AWS-QuickSetup-SSM*:*
arn:aws:cloudformation:*:*:stackset-target/AWS-QuickSetup-SSM*:*
arn:aws:cloudformation:*:*:type/resource/*
```

[7] `events:PutRule` 및 `events:PutTargets` 작업은 Systems Manager 서비스에 대해서만 다음 조건에 의해 허용됩니다.

```
"Condition": {
    "StringEquals": {
        "events:ManagedBy": "ssm.amazonaws.com"
    }
}
```

[8] `events:RemoveTargets` 및 `events:DeleteRule` 작업은 다음 리소스에서만 허용되는 권한입니다.

```
arn:aws:events:*:*:rule/SSMExplorerManagedRule
```

최신 버전의 JSON 정책 문서를 포함하여 정책에 대한 추가 세부 정보를 보려면 AWS 관리형 정책 참조 안내서의 AmazonSSMServiceRolePolicy를 참조하세요.

AWS 관리형 정책: AmazonSSMReadOnlyAccess

`AmazonSSMReadOnlyAccess` 정책을 IAM 보안 인증에 연결할 수 있습니다. 이 정책에서는 `Describe*`, `Get*` 및 `List*`를 포함하여 AWS Systems Manager API 작업에 대한 읽기 전용 액세스 권한을 부여합니다.

최신 버전의 JSON 정책 문서를 포함하여 정책에 대한 추가 세부 정보를 보려면 AWS 관리형 정책 참조 안내서의 AmazonSSMReadOnlyAccess를 참조하세요.

AWS 관리형 정책: AWSSystemsManagerOpsDataSyncServiceRolePolicy

AWSSystemsManagerOpsDataSyncServiceRolePolicy를 IAM 엔터티에 연결할 수 없습니다. 이 정책은 Systems Manager에서 사용자를 대신하여 작업을 수행할 수 있도록 서비스 연결 역할에 연결됩니다. 자세한 내용은 역할을 사용하여 Explorer용 OpsData 및 OpsItems 생성 단원을 참조하십시오.

AWSSystemsManagerOpsDataSyncServiceRolePolicy - AWSServiceRoleForSystemsManagerOpsDataSync 서비스 연결 역할이 AWS Security Hub 결과에서 OpsItems 및 OpsData를 생성 및 업데이트하도록 허용합니다.

정책에서는 지정된 경우를 제외하고 Systems Manager에서 모든 관련 리소스("Resource": "*")에 대한 다음과 같은 작업을 완료하도록 허용합니다.

- 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] ssm:GetOpsItem 및 ssm:UpdateOpsItem 작업은 Systems Manager 서비스에 대해서만 다음 조건에 의해 허용됩니다.

```
"Condition": {
    "StringEquals": {
        "aws:ResourceTag/ExplorerSecurityHubOpsItem": "true"
    }
}
```

[2] ssm:AddTagsToResource 작업은 다음 리소스에 대해서만 허용되는 권한입니다.

```
arn:aws:ssm:*:*:opsitem/*
```

[3] ssm:UpdateServiceSetting 및 ssm:GetServiceSetting 작업은 다음 리소스에 대해서만 허용되는 권한입니다.

```
arn:aws:ssm:*:*:servicesetting/ssm/opsitem/*
arn:aws:ssm:*:*:servicesetting/ssm/opsdata/*
```

[4] securityhub:BatchUpdateFindings는 Systems Manager 서비스에 대해서만 다음 조건에 의해 거부되는 권한입니다.

```
{
   "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
     }
    }
```

최신 버전의 JSON 정책 문서를 포함하여 정책에 대한 추가 세부 정보를 보려면 AWS 관리형 정책 참조 안내서의 AWSSystemsManagerOpsDataSyncServiceRolePolicy를 참조하세요.

AWS 관리형 정책: AmazonSSMManagedEC2InstanceDefaultPolicy

Systems Manager 기능 사용 권한을 원하는 Amazon EC2 인스턴스의 IAM 역할에만 `AmazonSSMManagedEC2InstanceDefaultPolicy`을 연결해야 합니다. 이 역할을 IAM 사용자 및 IAM 그룹과 같은 다른 IAM 엔티티나 용도가 다른 IAM 역할에 연결해서는 안 됩니다. 자세한 내용은 기본 호스트 관리 구성 설정 관리 단원을 참조하십시오.

이 정책은 Amazon EC2 인스턴스의 SSM Agent가 Documents를 검색하고, Run Command를 사용하여 명령을 실행하고, Session Manager를 사용하여 세션을 설정하고, 인스턴스 인벤토리를 수집하고, Patch Manager를 사용하여 패치 및 패치 규정 준수를 검색할 수 있는 권한을 부여합니다.

Systems Manager에서는 각 인스턴스에 대해 개인화된 권한 부여 토큰을 사용하여 SSM Agent가 올바른 인스턴스에서 API 작업을 수행하도록 합니다. Systems Manager에서는 API 작업에 제공된 인스턴스의 Amazon 리소스 이름(ARN)에 대해 개인화된 권한 부여 토큰을 검증합니다.

`AmazonSSMManagedEC2InstanceDefaultPolicy` 역할 권한 정책은 Systems Manager가 모든 관련 리소스에서 다음 작업을 수행하도록 허용합니다.

- ssm:DescribeAssociation
- ssm:GetDeployablePatchSnapshotForInstance
- ssm:GetDocument
- ssm:DescribeDocument
- ssm:GetManifest
- ssm:ListAssociations
- ssm:ListInstanceAssociations
- ssm:PutInventory
- ssm:PutComplianceItems
- ssm:PutConfigurePackageResult
- ssm:UpdateAssociationStatus
- ssm:UpdateInstanceAssociationStatus
- ssm:UpdateInstanceInformation
- ssmmessages:CreateControlChannel
- ssmmessages:CreateDataChannel
- ssmmessages:OpenControlChannel
- ssmmessages:OpenDataChannel
- ec2messages:AcknowledgeMessage
- ec2messages:DeleteMessage
- ec2messages:FailMessage
- ec2messages:GetEndpoint
- ec2messages:GetMessages
- ec2messages:SendReply

최신 버전의 JSON 정책 문서를 포함하여 정책에 대한 추가 세부 정보를 보려면 AWS 관리형 정책 참조 안내서의 AmazonSSMManagedEC2InstanceDefaultPolicy를 참조하세요.

AWS 관리형 정책으로 Systems Manager 업데이트

다음 테이블에는 2021년 3월 12일 해당 서비스가 변경 사항을 추적하기 시작한 이후 Systems Manager의 AWS 관리형 정책 업데이트에 대한 세부 정보가 나와 있습니다. Systems Manager 서비스

의 다른 관리형 정책에 대한 자세한 내용은 이 항목의 Systems Manager에 대한 추가 관리형 정책 뒷부분을 참조하세요. 이 페이지의 변경 사항에 대한 자동 알림을 받아보려면 Systems Manager 문서 이력 페이지에서 RSS 피드를 구독하세요.

변경 사항	설명	날짜
AWSSystemsManagerOpsDataSyncServiceRolePolicy - 기존 정책에 대한 업데이트	OpsCenter에서는 정책을 업데이트하여 OpsData 관련 작업을 관리하는 Explorer에 대한 서비스 연결 역할 내 서비스 코드의 보안을 개선했습니다.	2023년 6월 28일
AmazonSSMManagedEC2InstanceDefaultPolicy — 새 정책.	Systems Manager가 IAM 인스턴스 프로파일을 사용하지 않고 Amazon EC2 인스턴스에서 Systems Manager 기능을 허용하는 새로운 정책을 추가했습니다.	2022년 8월 18일
AmazonSSMServiceRolePolicy - 기존 정책에 대한 업데이트입니다.	Systems Manager가 Explorer 또는 OpsCenter에서 Security Hub를 설정할 때 Explorer가 관리형 규칙을 생성할 수 있도록 새로운 권한을 추가했습니다. 구성 및 Compute Optimizer가 OpsData를 허용하기 전에 필요한 요구 사항을 충족하는지 확인하기 위해 새로운 권한이 추가되었습니다.	2021년 4월 27일
AWSSystemsManagerOpsDataSyncServiceRolePolicy — 새 정책.	Systems Manager는 Explorer 및 OpsCenter의 Security Hub 결과에서 OpsItems 및 OpsData를 생성하고 업데이트하는 새로운 정책을 추가했습니다.	2021년 4월 27일

변경 사항	설명	날짜
AmazonSSMServiceRolePolicy - 기존 정책에 대한 업데이트	Systems Manager가 여러 계정의 집계 OpsData 및 OpsItems 세부 정보 및 Explorer의 AWS 리전을 볼 수 있도록 새로운 권한을 추가했습니다.	2021년 3월 24일
Systems Manager에서 변경 사항 추적 시작	Systems Manager가 AWS 관리형 정책에 대한 변경 내용 추적을 시작했습니다.	2021년 3월 12일

Systems Manager에 대한 추가 관리형 정책

이 주제의 앞부분에서 설명한 관리형 정책 외에도 Systems Manager에서는 다음과 같은 정책도 지원합니다.

- AmazonSSMAutomationApproverAccess-자동화 실행을 보고, 승인 대기 중인 자동화로 승인 결정을 전송하는 액세스 권한을 부여하는 AWS 관리형 정책.
- AmazonSSMAutomationRole-Automation 런북에 정의된 작업을 실행할 수 있는 권한을 Systems Manager Automation 서비스에 부여하는 AWS 관리형 정책. 관리자 및 신뢰할 수 있는 고급 사용자에게 이 정책을 할당합니다.
- AmazonSSMDirectoryServiceAccess-SSM Agent가 관리형 노드의 도메인 조인 요청에 대해 사용자 대신 AWS Directory Service에 액세스하도록 허용하는 AWS 관리형 정책.
- AmazonSSMFullAccess-Systems Manager API 및 문서에 대한 전체 액세스 권한을 부여하는 AWS 관리형 정책.
- AmazonSSMMaintenanceWindowRole-유지 관리 기간에 Systems Manager API에 대한 권한을 제공하는 AWS 관리형 정책.
- AmazonSSMManagedInstanceCore – 노드가 Systems Manager 서비스 코어 기능을 사용하도록 허용하는 AWS 관리형 정책.
- AmazonSSMPatchAssociation-패치 연결 작업을 위해 하위 인스턴스에 대한 액세스를 제공하는 AWS 관리형 정책.
- AmazonSSMReadOnlyAccess-Systems Manager 읽기 전용 API 작업(예: `Get*`, `List*`)에 대한 액세스 권한을 부여하는 AWS 관리형 정책.

- AWSSSMOpsInsightsServiceRolePolicy-Systems Manager에서 운영 인사이트 OpsItems를 생성하고 업데이트할 수 있는 권한을 부여하는 AWS 관리형 정책. 서비스 연결 역할 AWSServiceRoleForAmazonSSM_OpsInsights을 통해 권한을 제공하는 데 사용됩니다.
- AWSSystemsManagerAccountDiscoveryServicePolicy-Systems Manager에 AWS 계정 정보를 검색할 수 있는 권한을 부여하는 AWS 관리형 정책.
- AWSSystemsManagerChangeManagementServicePolicy-Systems Manager 변경 관리 프레임워크에서 관리 또는 사용하고 서비스 연결 역할 AWSServiceRoleForSystemsManagerChangeManagement에서 사용하는 AWS 리소스에 대한 액세스를 제공하는 AWS 관리형 정책.
- AmazonEC2RoleforSSM-이 정책은 더 이상 지원되지 않으므로 사용해서는 안 됩니다. 대신 EC2 인스턴스에서 AmazonSSMManagedInstanceCore 정책을 사용하여 Systems Manager 서비스 핵심 기능을 허용합니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성을 참조하세요.

AWS Systems Manager ID 및 액세스 문제 해결

다음 정보를 사용하여 AWS Systems Manager 및 AWS Identity and Access Management(IAM)로 작업할 때 발생할 수 있는 일반적인 문제를 진단하고 수정할 수 있습니다.

주제

- Systems Manager에서 작업을 수행할 권한이 없음
- iam:PassRole을 수행하도록 인증되지 않음
- 내 AWS 계정 외부의 사람이 내 Systems Manager 리소스에 액세스할 수 있게 허용하기를 원합니다.

Systems Manager에서 작업을 수행할 권한이 없음

AWS Management Console에서 태스크를 수행할 권한이 없다는 메시지가 나타나는 경우, 관리자에게 문의하여 도움을 받아야 합니다. 관리자는 로그인 보안 인증 정보를 제공한 사람입니다.

다음 예제 오류는 mateojackson 사용자가 콘솔을 사용하여 문서에 대한 세부 정보를 보려고 하지만 ssm:GetDocument 권한이 없는 경우에 발생합니다.

```
User: arn:aws:ssm::123456789012:user/mateojackson isn't authorized to perform:
 ssm:GetDocument on resource: MyExampleDocument
```

이 경우 Mateo는 `MyExampleDocument`작업을 사용하여 `ssm:GetDocument`리소스에 액세스하도록 허용하는 정책을 업데이트하라고 관리자에게 요청합니다.

iam:PassRole을 수행하도록 인증되지 않음

`iam:PassRole` 작업을 수행할 수 있는 권한이 없다는 오류가 수신되면 Systems Manager에 역할을 전달할 수 있도록 정책을 업데이트해야 합니다.

일부 AWS 서비스에서는 새 서비스 역할 또는 서비스 연결 역할을 생성하는 대신, 해당 서비스에 기존 역할을 전달할 수 있습니다. 이렇게 하려면 사용자가 서비스에 역할을 전달할 수 있는 권한을 가지고 있어야 합니다.

다음 예 오류는 `marymajor`라는 IAM 사용자가 콘솔을 사용하여 Systems Manager에서 작업을 수행하려고 하는 경우에 발생합니다. 하지만 작업을 수행하려면 서비스 역할이 부여한 권한이 서비스에 있어야 합니다. Mary는 서비스에 역할을 전달할 수 있는 권한을 가지고 있지 않습니다.

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

이 경우 Mary가 `iam:PassRole`작업을 수행할 수 있도록 Mary의 정책을 업데이트해야 합니다.

도움이 필요한 경우 AWS 관리자에게 문의합니다. 관리자는 로그인 보안 인증 정보를 제공하는 사람입니다.

내 AWS 계정 외부의 사람이 내 Systems Manager 리소스에 액세스할 수 있게 허용하기를 원합니다.

다른 계정의 사용자 또는 조직 외부의 사람이 리소스에 액세스할 때 사용할 수 있는 역할을 생성할 수 있습니다. 역할을 수임할 신뢰할 수 있는 사람을 지정할 수 있습니다. 리소스 기반 정책 또는 액세스 제어 목록(ACLs)을 지원하는 서비스의 경우 이러한 정책을 사용하여 다른 사람에게 리소스에 대한 액세스 권한을 부여할 수 있습니다.

자세히 알아보려면 다음을 참조하세요.

- Systems Manager에서 이러한 기능을 지원하는지 여부를 알아보려면 AWS Systems Manager에서 IAM을 사용하는 방식 섹션을 참조하세요.
- 소유하고 있는 AWS 계정의 리소스에 대한 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 자신이 소유한 다른 AWS 계정의 IAM 사용자에 대한 액세스 권한 제공을 참조하세요.
- 리소스에 대한 액세스 권한을 서드 파티 AWS 계정에게 제공하는 방법을 알아보려면 IAM 사용자 설명서의 서드 파티가 소유한 AWS 계정에 대한 액세스 제공을 참조하세요.

- 자격 증명 연동을 통해 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용자 설명서의 외부에서 인증된 사용자에게 액세스 권한 제공(자격 증명 연동)을 참조하세요.
- 교차 계정 액세스를 위한 역할과 리소스 기반 정책 사용의 차이점을 알아보려면 IAM 사용자 설명서의 IAM 역할과 리소스 기반 정책의 차이를 참조하십시오.

Systems Manager에 서비스 연결 역할 사용

AWS Systems Manager는 AWS Identity and Access Management(IAM) 서비스 연결 역할을 사용합니다. 서비스 링크 역할은 Systems Manager에 직접 연결된 고유한 유형의 IAM 역할입니다. 서비스 연결 역할은 Systems Manager에서 사전 정의하며 서비스에서 다른 AWS 서비스을(를) 자동으로 호출하기 위해 필요한 모든 권한을 포함합니다.

> **Note**
>
> 서비스 역할 역할은 서비스 연결 역할과 다릅니다. 서비스 역할은 AWS 리소스에 액세스할 수 있는 권한을 AWS 서비스에 부여하는 AWS Identity and Access Management(IAM) 역할의 한 유형입니다. 몇 가지 Systems Manager 시나리오에만 서비스 역할이 필요합니다. Systems Manager용 서비스 역할을 생성하는 경우 이 역할이 다른 AWS 리소스에 액세스하거나 상호 작용하도록 부여할 권한을 선택합니다.

서비스 연결 역할을 통해 Systems Manager 설정이 쉬워지는데 필요한 권한을 수동으로 추가할 필요가 없기 때문입니다. Systems Manager에서 서비스 연결 역할 권한을 정의하므로, 달리 정의되지 않은 한 Systems Manager에서만 해당 역할을 맡을 수 있습니다. 정의된 권한에는 신뢰 정책과 권한 정책이 포함되며, 이 권한 정책은 다른 IAM 엔터티에 연결할 수 없습니다.

먼저 관련 리소스를 삭제한 후에만 서비스 연결 역할을 삭제할 수 있습니다. 이렇게 하면 리소스에 대한 액세스 권한을 부주의로 삭제할 수 없기 때문에 Systems Manager 리소스가 보호됩니다.

> **Note**
>
> 하이브리드 및 멀티클라우드 환경의 비 EC2 노드에는 해당 시스템에서 Systems Manager 서비스와 통신할 수 있는 추가 IAM 역할이 필요합니다. 이것이 Systems Manager용 IAM 서비스 역할입니다. 이 역할은 AWS Security Token Service(AWS STS) AssumeRole 신뢰를 Systems Manager 서비스에 부여합니다. `AssumeRole` 작업은 액세스 키 ID, 보안 액세스 키 및 보안 토큰으로 구성된 일련의 임시 보안 자격 증명을 반환합니다. 이러한 임시 자격 증명을 사용하여 평소에는 액세스 권한이 없을 수 있는 AWS 리소스에 액세스할 수 있습니다. 자세한 내용

은 AWS Security Token Service API 참조의 하이브리드 및 멀티클라우드 환경에서 Systems Manager에 필요한 IAM 서비스 역할 생성 및 AssumeRole을 참조하세요.

서비스 연결 역할을 지원하는 기타 서비스에 대한 자세한 내용은 IAM으로 작업하는 AWS 서비스을(를) 참조하고 서비스 연결 역할 열에 예가 있는 서비스를 찾아보세요. 해당 서비스에 대한 서비스 연결 역할 설명서를 보려면 링크가 있는 예를 선택합니다.

주제

- 역할을 사용하여 인벤토리 수집 및 OpsData 보기
- 역할을 사용하여 OpsCenter 및 Explorer에 대한 AWS 계정 정보 수집
- 역할을 사용하여 Explorer용 OpsData 및 OpsItems 생성
- 역할을 사용하여 Systems Manager OpsCenter에서 운영 인사이트 OpsItems 생성
- 역할을 사용하여 Explorer OpsData 내보내기

역할을 사용하여 인벤토리 수집 및 OpsData 보기

Systems Manager는 **AWSServiceRoleForAmazonSSM**이라는 서비스 연결 역할을 사용합니다. AWS Systems Manager는 이 IAM 서비스 역할을 사용하여 AWS 리소스를 대신 관리합니다.

인벤토리, OpsData 및 OpsItems에 대한 서비스 연결 역할 권한

`AWSServiceRoleForAmazonSSM` 서비스 연결 역할은 이 역할을 맡을 `ssm.amazonaws.com`만 신뢰합니다.

다음과 같은 경우에 Systems Manager 서비스 연결 역할 `AWSServiceRoleForAmazonSSM`을 사용할 수 있습니다.

- Systems Manager Inventory 기능은 서비스 연결 역할 `AWSServiceRoleForAmazonSSM`을 사용하여 태그와 리소스 그룹에서 인벤토리 메타데이터를 수집합니다.
- Explorer 기능은 서비스 연결 역할 `AWSServiceRoleForAmazonSSM`을 사용하여 여러 계정에서 OpsData와 OpsItems 보기를 사용합니다. 이 서비스 연결 역할을 통해 Explorer 또는 OpsCenter에서 Security Hub를 데이터 원본으로 사용하면 Explorer이 관리형 규칙을 생성할 수도 있습니다.

> **Important**
>
> 이전에는 Systems Manager 콘솔에서 작업에 대한 유지 관리 역할로 사용하도록 AWS에서 관리하는 IAM 서비스 연결 역할인 `AWSServiceRoleForAmazonSSM`을 선택할 수 있었습니다. 유지 관리 기간에 이 역할 및 관련 정책인 `AmazonSSMServiceRolePolicy`를 사용하는 것은 더 이상 권장되지 않습니다. 현재 유지 관리 기간 작업에 이 역할을 사용하는 경우 사용을 중지하는 것이 좋습니다. 대신, 유지 관리 기간 작업 실행 시 Systems Manager와 다른 AWS 서비스 간에 통신을 가능하게 하는 IAM 역할을 생성하세요.
> 자세한 내용은 Maintenance Windows 설정 단원을 참조하십시오.

`AWSServiceRoleForAmazonSSM` 역할에 대한 권한을 제공하는 데 사용되는 관리형 정책은 `AmazonSSMServiceRolePolicy`입니다. 부여되는 권한에 대한 자세한 내용은 AWS 관리형 정책: AmazonSSMServiceRolePolicy 섹션을 참조하세요.

Systems Manager에 대한 **`AWSServiceRoleForAmazonSSM`** 서비스 연결 역할 생성

IAM 콘솔을 사용하여 EC2 사용 사례에서 서비스 연결 역할을 생성할 수 있습니다. AWS Command Line Interface(AWS CLI)에서 IAM에 대한 명령을 사용하거나 IAM API를 사용하여 `ssm.amazonaws.com` 서비스 이름으로 서비스 연결 역할을 생성합니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 생성을 참조하세요.

이 서비스 연결 역할을 삭제했다가 다시 생성해야 하는 경우 동일한 프로세스를 사용하여 계정에서 역할을 다시 생성할 수 있습니다.

Systems Manager에 대한 **`AWSServiceRoleForAmazonSSM`** 서비스 연결 역할 편집

Systems Manager에서는 `AWSServiceRoleForAmazonSSM` 서비스 연결 역할을 편집하도록 허용하지 않습니다. 서비스 연결 역할을 생성한 후에는 다양한 엔터티가 역할을 참조할 수 있기 때문에 역할 이름을 변경할 수 없습니다. 하지만 IAM을 사용하여 역할의 설명을 편집할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 편집을 참조하세요.

Systems Manager에 대한 **`AWSServiceRoleForAmazonSSM`** 서비스 연결 역할 삭제

서비스 연결 역할이 요구되는 기능 또는 서비스가 더 이상 필요 없어지면 해당 역할을 삭제하는 것이 좋습니다. 이렇게 하면 적극적으로 모니터링하거나 유지 관리하지 않는 미사용 엔터티가 없게 됩니다. IAM 콘솔, AWS CLI 또는 IAM API를 사용하여 서비스 연결 역할을 수동으로 삭제할 수 있습니다. 단, 서비스 연결 역할에 대한 리소스를 먼저 정리해야 수동으로 삭제할 수 있습니다.

AWSServiceRoleForAmazonSSM 서비스 연결 역할은 여러 기능에서 사용할 수 있기 때문에, 삭제하기 전에 기능에서 역할을 사용하고 있지 않은지 확인합니다.

- 인벤토리: 인벤토리 기능에서 사용하는 서비스 연결 역할을 삭제하면 태그 및 리소스 그룹에 대한 인벤토리 데이터가 더는 동기화되지 않습니다. 서비스 연결 역할에 대한 리소스를 먼저 정리해야 수동으로 삭제할 수 있습니다.
- Explorer: Explorer 기능에서 사용하는 서비스 연결 역할을 삭제하면 크로스 계정 및 크로스 리전 OpsData와 OpsItems가 더 이상 표시되지 않습니다.

> **Note**
>
> 태그 또는 리소스 그룹을 삭제하려 할 때 Systems Manager 서비스가 역할을 사용 중이면 삭제에 실패할 수 있습니다. 이 문제가 발생하면 몇 분 기다렸다가 작업을 다시 시도하세요.

AWSServiceRoleForAmazonSSM에서 사용하는 Systems Manager 리소스를 삭제하려면

1. 태그를 삭제하려면 개별 리소스에서 태그 추가 및 삭제를 참조하세요.
2. 리소스 그룹을 삭제하려면 AWS Resource Groups에서 그룹 삭제를 참조하세요.

IAM을 사용하여 수동으로 **AWSServiceRoleForAmazonSSM** 서비스 연결 역할 삭제

IAM 콘솔, AWS CLI 또는 IAM API를 사용하여 AWSServiceRoleForAmazonSSM 서비스 연결 역할을 삭제합니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 삭제를 참조하십시오.

Systems Manager**AWSServiceRoleForAmazonSSM** 서비스 연결 역할이 지원되는 리전

Systems Manager에서는 서비스를 사용할 수 있는 모든 AWS 리전에서 AWSServiceRoleForAmazonSSM 서비스 연결 역할 사용을 지원합니다. 자세한 내용은 AWS Systems Manager 엔드포인트 및 할당량을 참조하세요.

역할을 사용하여 OpsCenter 및 Explorer에 대한 AWS 계정 정보 수집

Systems Manager은(는) **AWSServiceRoleForAmazonSSM_AccountDiscovery**(이)라는 서비스 연결 역할을 사용합니다. AWS Systems Manager은(는) 이 IAM 서비스 역할을 사용하여 AWS 계정 정보를 검색하기 위해 다른 AWS 서비스을(를) 호출합니다.

Systems Manager 계정 검색을 위한 서비스 연결 역할 권한

`AWSServiceRoleForAmazonSSM_AccountDiscovery` 서비스 연결 역할은 역할을 수임하기 위해 다음 서비스를 신뢰합니다.

- `accountdiscovery.ssm.amazonaws.com`

역할 권한 정책은 Systems Manager가 지정된 리소스에서 다음 작업을 완료하도록 허용합니다.

- `organizations:DescribeAccount`
- `organizations:DescribeOrganizationalUnit`
- `organizations:DescribeOrganization`
- `organizations:ListAccounts`
- `organizations:ListAWSServiceAccessForOrganization`
- `organizations:ListChildren`
- `organizations:ListParents`
- `organizations:ListDelegatedServicesForAccount`
- `organizations:ListDelegatedAdministrators`
- `organizations:ListRoots`

IAM 엔터티(사용자, 그룹, 역할 등)가 서비스 링크 역할을 생성하고 편집하거나 삭제할 수 있도록 권한을 구성할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 권한을 참조하세요.

Systems Manager에 대한 **`AWSServiceRoleForAmazonSSM_AccountDiscovery`** 서비스 연결 역할 생성

Systems Manager의 Explorer 및 OpsCenter 기능을 여러 AWS 계정에서 사용하려면 서비스 연결 역할을 생성해야 합니다. OpsCenter의 경우 서비스 연결 역할을 수동으로 생성해야 합니다. 자세한 내용은 (선택 사항) 여러 계정에서 OpsItems를 중앙 집중식으로 관리하도록 OpsCenter 설정 단원을 참조하십시오.

Explorer의 경우 AWS Management Console에서 Systems Manager를 사용하여 리소스 데이터 동기화를 생성할 때 Create role(역할 생성) 버튼을 선택하여 서비스 연결 역할을 생성할 수 있습니다. 프로그래밍 방식으로 리소스 데이터 동기화를 생성하려면 리소스 데이터 동기화를 생성 전에 역할을 생성해야 합니다. CreateServiceLinkedRole API 작업을 사용하여 역할을 생성할 수 있습니다.

Systems Manager에 대한 **AWSServiceRoleForAmazonSSM_AccountDiscovery** 서비스 연결 역할 편집

Systems Manager에서는 AWSServiceRoleForAmazonSSM_AccountDiscovery 서비스 연결 역할을 편집하도록 허용하지 않습니다. 서비스 연결 역할을 생성한 후에는 다양한 엔터티가 역할을 참조할 수 있기 때문에 역할 이름을 변경할 수 없습니다. 하지만 IAM을 사용하여 역할의 설명을 편집할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 편집을 참조하세요.

Systems Manager에 대한 **AWSServiceRoleForAmazonSSM_AccountDiscovery** 서비스 연결 역할 삭제

서비스 연결 역할이 필요한 기능 또는 서비스가 더 이상 필요 없는 경우에는 해당 역할을 삭제하는 것이 좋습니다. 이렇게 하면 적극적으로 모니터링하거나 유지 관리하지 않는 미사용 엔터티가 없게 됩니다. 단, 서비스 연결 역할을 정리해야 수동으로 삭제할 수 있습니다.

AWSServiceRoleForAmazonSSM_AccountDiscovery 서비스 연결 역할 정리

IAM을 사용하여 AWSServiceRoleForAmazonSSM_AccountDiscovery 서비스 연결 역할을 삭제하기 전에 먼저 Explorer 리소스 데이터 동기화를 모두 삭제해야 합니다. 자세한 내용은 Systems Manager Explorer 리소스 데이터 동기화 삭제 단원을 참조하십시오.

> **Note**
>
> 리소스를 삭제하려 할 때 Systems Manager 서비스가 역할을 사용 중이면 삭제에 실패할 수 있습니다. 이 문제가 발생하면 몇 분 기다렸다가 작업을 다시 시도하십시오.

수동으로 **AWSServiceRoleForAmazonSSM_AccountDiscovery** 서비스 연결 역할 삭제

IAM 콘솔, AWS CLI 또는 AWS API를 사용하여 AWSServiceRoleForAmazonSSM_AccountDiscovery 서비스 연결 역할을 삭제할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 삭제를 참조하십시오.

Systems Manager**AWSServiceRoleForAmazonSSM_AccountDiscovery** 서비스 연결 역할이 지원되는 리전

Systems Manager는 서비스가 제공되는 모든 리전에서 서비스 연결 역할을 사용하도록 지원합니다. 자세한 내용은 AWS Systems Manager 엔드포인트 및 할당량을 참조하세요.

AWSServiceRoleForAmazonSSM_AccountDiscovery 서비스 연결 역할 업데이트

이 서비스에서 이러한 변경 사항 추적을 시작한 이후 AWSServiceRoleForAmazonSSM_AccountDiscovery service-linked 서비스 연결 역할 업데이트에 대한 세부 정보를 봅니다. 이 페이지의 변경 사항에 대한 자동 알림을 받아보려면 Systems Manager 문서 이력 페이지에서 RSS 피드를 구독하세요.

변경 사항	설명	날짜
새 권한이 추가됨	이제 이 서비스 연결 역할에는 `organizations:DescribeOrganizationalUnit` 및 `organizations:ListRoots` 권한이 포함됩니다. 이러한 권한을 통해 AWS Organizations 관리 계정 또는 Systems Manager 위임 관리자 계정에서 여러 계정에 결쳐 OpsItems 작업을 수행할 수 있습니다. 자세한 내용은 (선택 사항) 여러 계정에서 OpsItems를 중앙 집중식으로 관리하도록 OpsCenter 설정 단원을 참조하십시오.	2022년 10월 17일

역할을 사용하여 Explorer용 OpsData 및 OpsItems 생성

Systems Manager는 **`AWSServiceRoleForSystemsManagerOpsDataSync`**라는 서비스 연결 역할을 사용합니다. AWS Systems Manager는 Explorer에 대해 이 IAM 서비스 역할을 사용하여 OpsData와 OpsItems를 생성합니다.

Systems Manager OpsData 동기화에 대한 서비스 연결 역할 권한

`AWSServiceRoleForSystemsManagerOpsDataSync` 서비스 연결 역할은 역할을 수임하기 위해 다음 서비스를 신뢰합니다.

- `opsdatasync.ssm.amazonaws.com`

역할 권한 정책은 Systems Manager가 지정된 리소스에서 다음 작업을 완료하도록 허용합니다.

- Systems Manager Explorer를 사용하려면 서비스 연결 역할이 OpsItem이 업데이트될 때 보안 결과를 업데이트하고, OpsItem을 생성 및 업데이트하고, 고객이 SSM 관리형 규칙을 삭제할 때 Security Hub 데이터 원본을 해제할 수 있는 권한을 부여해야 합니다.

`AWSServiceRoleForSystemsManagerOpsDataSync` 역할에 대한 권한을 제공하는 데 사용되는 관리형 정책은 `AWSSystemsManagerOpsDataSyncServiceRolePolicy`입니다. 부여되는 권한에 대한 자세한 내용은 AWS 관리형 정책: AWSSystemsManagerOpsDataSyncServiceRolePolicy 섹션을 참조하세요.

IAM 엔터티(사용자, 그룹, 역할 등)가 서비스 링크 역할을 생성하고 편집하거나 삭제할 수 있도록 권한을 구성할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 권한을 참조하세요.

Systems Manager에 대한 **`AWSServiceRoleForSystemsManagerOpsDataSync`** 서비스 연결 역할 생성

서비스 링크 역할은 수동으로 생성할 필요가 없습니다. AWS Management Console 콘솔에서 Explorer를 사용하면 Systems Manager에서 서비스 연결 역할을 생성합니다.

> **Important**
>
> 이 서비스 연결 역할은 이 역할이 지원하는 기능을 사용하는 다른 서비스에서 작업을 완료했을 경우 계정에 표시될 수 있습니다. 또한 Systems Manager 서비스가 서비스 연결 역할을 지원하기 시작한 2017년 1월 1일 이전에 이 서비스를 사용 중이었다면 Systems Manager에서 사용자 계정에 `AWSServiceRoleForSystemsManagerOpsDataSync` 역할을 이미 생성했습니다. 자세한 내용은 내 IAM 계정에 표시되는 새 역할을 참조하세요.

이 서비스 연결 역할을 삭제했다가 다시 생성해야 하는 경우 동일한 프로세스를 사용하여 계정에서 역할을 다시 생성할 수 있습니다. AWS Management Console에서 Explorer를 사용하면 Systems Manager에서 서비스 연결 역할을 다시 생성합니다.

IAM 콘솔을 사용하여 Explorer에 OpsData 및 OpsItems 사용 사례 생성을 허용하는 AWS 서비스 역할 사용 사례로 서비스 연결 역할을 생성할 수도 있습니다. AWS CLI 또는 AWS API에서 `opsdatasync.ssm.amazonaws.com` 서비스 이름의 서비스 연결 역할을 생성합니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 생성을 참조하십시오. 이 서비스 연결 역할을 삭제하면 동일한 프로세스를 사용하여 역할을 다시 생성할 수 있습니다.

Systems Manager에 대한 **AWSServiceRoleForSystemsManagerOpsDataSync** 서비스 연결 역할 편집

Systems Manager에서는 AWSServiceRoleForSystemsManagerOpsDataSync 서비스 연결 역할을 편집하도록 허용하지 않습니다. 서비스 연결 역할을 생성한 후에는 다양한 엔터티가 역할을 참조할 수 있기 때문에 역할 이름을 변경할 수 없습니다. 하지만 IAM을 사용하여 역할의 설명을 편집할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 편집을 참조하세요.

Systems Manager에 대한 **AWSServiceRoleForSystemsManagerOpsDataSync** 서비스 연결 역할 삭제

서비스 연결 역할이 필요한 기능 또는 서비스가 더 이상 필요 없는 경우에는 해당 역할을 삭제하는 것이 좋습니다. 이렇게 하면 적극적으로 모니터링하거나 유지 관리하지 않는 미사용 엔터티가 없게 됩니다. 단, 서비스 링크 역할에 대한 리소스를 먼저 정리해야 수동으로 삭제할 수 있습니다.

> **Note**
>
> 리소스를 삭제하려 할 때 Systems Manager 서비스가 역할을 사용 중이면 삭제에 실패할 수 있습니다. 이 문제가 발생하면 몇 분 기다렸다가 작업을 다시 시도하십시오.

AWSServiceRoleForSystemsManagerOpsDataSync 역할에서 사용하는 Systems Manager 리소스를 삭제하는 절차는 Explorer 또는 OpsCenter를 Security Hub와 통합하도록 구성했는지 여부에 따라 다릅니다.

AWSServiceRoleForSystemsManagerOpsDataSync 역할에서 사용하는 Systems Manager 리소스를 삭제하려면

- Explorer에서 Security Hub 결과에 대해 새 OpsItems을 생성하지 않게 하려면 결과 수신을 중지하는 방법 섹션을 참조하세요.
- OpsCenter에서 Security Hub 결과에 대해 새 OpsItems를 생성하지 않게 하려면 다음 섹션을 참조하세요.

IAM을 사용하여 수동으로 **AWSServiceRoleForSystemsManagerOpsDataSync** 서비스 연결 역할 삭제

IAM 콘솔, AWS CLI 또는 AWS API를 사용하여 `AWSServiceRoleForSystemsManagerOpsDataSync` 서비스 연결 역할을 삭제합니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 삭제를 참조하십시오.

Systems Manager**AWSServiceRoleForSystemsManagerOpsDataSync** 서비스 연결 역할이 지원되는 리전

Systems Manager에서는 서비스를 사용할 수 있는 모든 리전에서 서비스 링크 역할 사용을 지원합니다. 자세한 내용은 AWS Systems Manager 엔드포인트 및 할당량을 참조하세요.

Systems Manager에서는 서비스가 제공되는 모든 리전에서 서비스 연결 역할을 사용하도록 지원하지 않습니다. 다음 리전에서 `AWSServiceRoleForSystemsManagerOpsDataSync` 역할을 사용할 수 있습니다.

AWS 리전 이름	리전 자격 증명	Systems Manager의 지원
미국 동부(버지니아 북부)	us-east-1	예
미국 동부(오하이오)	us-east-2	예
미국 서부(캘리포니아 북부)	us-west-1	예
미국 서부(오레곤)	us-west-2	예
아시아 태평양(뭄바이)	ap-south-1	예
아시아 태평양(오사카)	ap-northeast-3	예
아시아 태평양(서울)	ap-northeast-2	예
아시아 태평양(싱가포르)	ap-southeast-1	예
아시아 태평양(시드니)	ap-southeast-2	예
아시아 태평양(도쿄)	ap-northeast-1	예
캐나다(중부)	ca-central-1	예
유럽(프랑크푸르트)	eu-central-1	예
유럽(아일랜드)	eu-west-1	예

AWS 리전 이름	리전 자격 증명	Systems Manager의 지원
유럽(런던)	eu-west-2	예
유럽(파리)	eu-west-3	예
유럽(스톡홀름)	eu-north-1	예
남아메리카(상파울루)	sa-east-1	예
AWS GovCloud (US)	us-gov-west-1	아니요

역할을 사용하여 Systems Manager OpsCenter에서 운영 인사이트 OpsItems 생성

Systems Manager는 **AWSServiceRoleForAmazonSSM_OpsInsights**라는 서비스 연결 역할을 사용합니다. AWS Systems Manager는 이 IAM 서비스 역할을 사용하여 Systems Manager OpsCenter에서 운영 인사이트 OpsItems를 생성하고 업데이트합니다.

Systems Manager 운영 인사이트 OpsItems에 대한 **AWSServiceRoleForAmazonSSM_OpsInsights** 서비스 연결 역할 권한

AWSServiceRoleForAmazonSSM_OpsInsights 서비스 연결 역할은 역할을 수임하기 위해 다음 서비스를 신뢰합니다.

- opsinsights.ssm.amazonaws.com

역할 권한 정책은 Systems Manager가 지정된 리소스에서 다음 작업을 완료하도록 허용합니다.

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

IAM 엔터티(사용자, 그룹, 역할 등)가 서비스 링크 역할을 생성하고 편집하거나 삭제할 수 있도록 권한을 구성할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 권한을 참조하세요.

Systems Manager에 대한 **AWSServiceRoleForAmazonSSM_OpsInsights** 서비스 연결 역할 생성

서비스 연결 역할을 생성해야 합니다. AWS Management Console에서 Systems Manager를 사용하여 운영 인사이트를 사용하면 [사용(Enable)] 버튼을 선택하여 서비스 연결 역할을 생성할 수 있습니다.

Systems Manager에 대한 **AWSServiceRoleForAmazonSSM_OpsInsights** 서비스 연결 역할 편집

Systems Manager에서는 AWSServiceRoleForAmazonSSM_OpsInsights 서비스 연결 역할을 편집하도록 허용하지 않습니다. 서비스 링크 역할을 생성한 후에는 다양한 개체가 역할을 참조할 수 있기 때문에 역할 이름을 변경할 수 없습니다. 하지만 IAM을 사용하여 역할의 설명을 편집할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 편집을 참조하세요.

Systems Manager에 대한 **AWSServiceRoleForAmazonSSM_OpsInsights** 서비스 연결 역할 삭제

서비스 연결 역할이 필요한 기능 또는 서비스가 더 이상 필요 없는 경우에는 해당 역할을 삭제하는 것이 좋습니다. 따라서 적극적으로 모니터링하거나 유지하지 않는 미사용 엔터티가 없도록 합니다. 단, 서비스 연결 역할을 정리해야 수동으로 삭제할 수 있습니다.

AWSServiceRoleForAmazonSSM_OpsInsights 서비스 연결 역할 정리

IAM을 사용하여 AWSServiceRoleForAmazonSSM_OpsInsights 서비스 연결 역할을 삭제하려면 먼저 Systems Manager OpsCenter에서 운영 인사이트를 비활성화해야 합니다. 자세한 내용은 운영 인사이트를 분석하여 OpsItems 줄이기 단원을 참조하십시오.

수동으로 **AWSServiceRoleForAmazonSSM_OpsInsights** 서비스 연결 역할 삭제

IAM 콘솔, AWS CLI 또는 AWS API를 사용하여 AWSServiceRoleForAmazonSSM_OpsInsights 서비스 연결 역할을 삭제할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 삭제를 참조하십시오.

Systems Manager**AWSServiceRoleForAmazonSSM_OpsInsights** 서비스 연결 역할이 지원되는 리전

Systems Manager에서는 서비스가 제공되는 모든 리전에서 서비스 연결 역할을 사용하도록 지원하지 않습니다. 다음 리전에서 AWSSSMOpsInsightsServiceRolePolicy 역할을 사용할 수 있습니다.

지역명	리전 자격 증명	Systems Manager의 지원
미국 동부(버지니아 북부)	us-east-1	예
미국 동부(오하이오)	us-east-2	예
미국 서부(캘리포니아 북부)	us-west-1	예
미국 서부(오레곤)	us-west-2	예
아시아 태평양(뭄바이)	ap-south-1	예
아시아 태평양(도쿄)	ap-northeast-1	예
아시아 태평양(서울)	ap-northeast-2	예

지역명	리전 자격 증명	Systems Manager의 지원
아시아 태평양(싱가포르)	ap-southeast-1	예
아시아 태평양(시드니)	ap-southeast-2	예
아시아 태평양(홍콩)	ap-east-1	예
캐나다(중부)	ca-central-1	예
유럽(프랑크푸르트)	eu-central-1	예
유럽(아일랜드)	eu-west-1	예
유럽(런던)	eu-west-2	예
유럽(파리)	eu-west-3	예
유럽(스톡홀름)	eu-north-1	예
유럽(밀라노)	eu-south-1	예
남아메리카(상파울루)	sa-east-1	예
중동(바레인)	me-south-1	예
아프리카(케이프타운)	af-south-1	예
AWS GovCloud (US)	us-gov-west-1	예
AWS GovCloud (US)	us-gov-east-1	예

역할을 사용하여 Explorer OpsData 내보내기

AWS Systems Manager Explorer는 AmazonSSMExplorerExportRole 서비스 역할을 통해 `AWS-ExportOpsDataToS3` 자동화 런북을 사용하여 작업 데이터(OpsData)를 내보냅니다.

Explorer에 대한 서비스 링크 역할 권한

`AmazonSSMExplorerExportRole` 서비스 연결 역할은 이 역할을 맡을 `ssm.amazonaws.com`만 신뢰합니다.

AmazonSSMExplorerExportRole 서비스 연결 역할을 사용하면 AWS-ExportOpsDataToS3 자동화 런북을 통해 운영 데이터(OpsData)를 내보낼 수 있습니다. Explorer에서 OpsData 항목 5,000개를 쉼표로 구분된 값(.csv) 파일로 Amazon Simple Storage Service(Amazon S3) 버킷에 내보낼 수 있습니다.

역할 권한 정책은 Systems Manager가 지정된 리소스에서 다음 작업을 완료하도록 허용합니다.

- s3:PutObject
- s3:GetBucketAcl
- s3:GetBucketLocation
- sns:Publish
- logs:DescribeLogGroups
- logs:DescribeLogStreams
- logs:CreateLogGroup
- logs:PutLogEvents
- logs:CreateLogStream
- ssm:GetOpsSummary

IAM 엔터티(사용자, 그룹, 역할 등)가 서비스 링크 역할을 생성하고 편집하거나 삭제할 수 있도록 권한을 구성할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 권한을 참조하세요.

Systems Manager에 대한 **AmazonSSMExplorerExportRole** 서비스 연결 역할 생성

Systems Manager는 Systems Manager 콘솔에서 Explorer를 사용하여 OpsData를 내보낼 때 AmazonSSMExplorerExportRole 서비스 연결 역할을 생성합니다. 자세한 내용은 Systems Manager Explorer에서 OpsData 내보내기 단원을 참조하십시오.

이 서비스 연결 역할을 삭제했다가 다시 생성해야 하는 경우 동일한 프로세스를 사용하여 계정에서 역할을 다시 생성할 수 있습니다.

Systems Manager에 대한 **AmazonSSMExplorerExportRole** 서비스 연결 역할 편집

Systems Manager에서는 AmazonSSMExplorerExportRole 서비스 연결 역할을 편집하도록 허용하지 않습니다. 서비스 연결 역할을 생성한 후에는 다양한 엔터티가 역할을 참조할 수 있기 때문에 역할 이름을 변경할 수 없습니다. 하지만 IAM을 사용하여 역할의 설명을 편집할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 편집을 참조하세요.

Systems Manager에 대한 **AmazonSSMExplorerExportRole** 서비스 연결 역할 삭제

서비스 연결 역할이 요구되는 기능 또는 서비스가 더 이상 필요 없어지면 해당 역할을 삭제하는 것이 좋습니다. 이렇게 하면 적극적으로 모니터링하거나 유지 관리하지 않는 미사용 엔터티가 없게 됩니다. IAM 콘솔, AWS CLI 또는 IAM API를 사용하여 서비스 연결 역할을 수동으로 삭제할 수 있습니다. 단, 서비스 연결 역할에 대한 리소스를 먼저 정리해야 수동으로 삭제할 수 있습니다.

> **Note**
>
> 태그 또는 리소스 그룹을 삭제하려 할 때 Systems Manager 서비스가 역할을 사용 중이면 삭제에 실패할 수 있습니다. 이 문제가 발생하면 몇 분 기다렸다가 작업을 다시 시도하세요.

AmazonSSMExplorerExportRole에서 사용하는 Systems Manager 리소스를 삭제하려면

1. 태그를 삭제하려면 개별 리소스에서 태그 추가 및 삭제를 참조하세요.
2. 리소스 그룹을 삭제하려면 AWS Resource Groups에서 그룹 삭제를 참조하세요.

IAM을 사용하여 수동으로 **AmazonSSMExplorerExportRole** 서비스 연결 역할 삭제

IAM 콘솔, AWS CLI 또는 IAM API를 사용하여 AmazonSSMExplorerExportRole 서비스 연결 역할을 삭제합니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 삭제를 참조하십시오.

Systems Manager**AmazonSSMExplorerExportRole** 서비스 연결 역할이 지원되는 리전

Systems Manager에서는 서비스를 사용할 수 있는 모든 AWS 리전에서 AmazonSSMExplorerExportRole 서비스 연결 역할 사용을 지원합니다. 자세한 내용은 AWS Systems Manager 엔드포인트 및 할당량을 참조하세요.

AWS Systems Manager의 로깅 및 모니터링

모니터링은 AWS Systems Manager와 사용자 AWS 솔루션의 신뢰성, 가용성 및 성능을 유지하는 중요한 역할을 합니다. 다중 지점 실패가 발생할 경우 보다 디버깅할 수 있도록 AWS 솔루션의 모든 부분으로부터 모니터링 데이터를 수집해야 합니다. AWS는 Systems Manager 및 다른 리소스를 모니터링하고 잠재적 인시던트에 대응하기 위한 여러 도구를 제공합니다.

AWS CloudTrail 로그

CloudTrail은 Systems Manager에서 사용자, 역할 또는 AWS 서비스이(가) 수행한 작업 기록을 제공합니다. CloudTrail에서 수집한 정보를 사용하여 Systems Manager에 수행된 요청, 요청이 수행된 IP 주소, 요청을 수행한 사람, 요청이 수행된 시간 및 추가 세부 정보를 확인할 수 있습니다. 자세한 내용은 AWS CloudTrail을 사용하여 AWS Systems ManagerAPI 호출을 로깅 단원을 참조하십시오.

Amazon CloudWatch 경보

Amazon CloudWatch 경보를 사용하여 지정한 기간 동안 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 및 기타 리소스에 대한 단일 지표를 감시합니다. 지표가 지정된 임계값을 초과하면 Amazon Simple Notification Service(Amazon SNS) 주제 또는 AWS Auto Scaling 정책으로 알림이 전송됩니다. CloudWatch 경보는 특정 상태에 있다고 해서 작업을 호출하지 않습니다. 대신, 상태가 변경되어 지정된 기간 동안 유지되어야 합니다. 자세한 내용은 Amazon CloudWatch 사용 설명서의 Amazon CloudWatch 경보 사용을 참조하세요.

Amazon CloudWatch 대시보드

Amazon CloudWatch 대시보드는 CloudWatch 콘솔에서 사용자 지정이 가능한 홈 페이지로, 다른 AWS 리전에 분산되어 있는 리소스를 비롯하여 단일 보기에서 리소스를 모니터링하는 데 사용할 수 있습니다. CloudWatch 대시보드를 사용해 AWS 리소스에 대한 지표 및 경보를 보여주는 사용자 정의 뷰를 생성할 수 있습니다. 자세한 내용은 Systems Manager에서 호스팅하는 Amazon CloudWatch 대시보드 단원을 참조하십시오.

Amazon EventBridge

Amazon EventBridge를 사용하여 Systems Manager 리소스의 변경 사항을 알리고 EventBridge가 해당 이벤트의 내용에 따라 조치를 취하도록 지시하는 규칙을 구성할 수 있습니다. EventBridge는 다양한 Systems Manager 기능에서 발생하는 여러 이벤트에 대한 지원을 제공합니다. 자세한 내용은 Amazon EventBridge로 Systems Manager 이벤트 모니터링 단원을 참조하십시오.

Amazon CloudWatch Logs 및 SSM Agent 로그

SSM Agent는 실행, 예약된 작업, 오류 및 상태에 대한 정보를 각 노드의 로그 파일에 씁니다. 노드에 수동으로 연결하여 로그 파일을 볼 수 있습니다. 분석을 위해 CloudWatch Logs의 로그 그룹에 에이전트 로그 데이터를 자동으로 보내는 것이 좋습니다. 자세한 내용은 통합 CloudWatch Logs로 노드 로그 전송(CloudWatch 에이전트) 및 SSM Agent 로그 보기 단원을 참조하세요.

AWS Systems Manager 규정 준수

AWS Systems Manager의 기능인 Compliance를 사용하여 관리형 노드 플릿에 대해 패치 규정 준수 및 구성 일관성을 스캔할 수 있습니다. 여러 AWS 계정 및 AWS 리전의 데이터를 수집하여 집

계한 후 규정을 준수하지 않는 특정 리소스로 드릴다운할 수 있습니다. 기본적으로 규정 준수는 AWS Systems Manager의 기능인 Patch Manager에서 패치에 대한 현재 규정 준수 데이터를 표시하고 AWS Systems Manager의 기능인 State Manager에 연결을 표시합니다. 자세한 내용은 AWS Systems Manager Compliance 단원을 참조하십시오.

AWS Systems Manager Explorer

AWS Systems Manager의 기능인 Explorer는 AWS 리소스에 대한 정보를 보고하는 사용자 정의 가능한 운영 대시보드입니다. Explorer에는 AWS 계정 및 AWS 리전의 운영 데이터(OpsData)에 대한 집계 뷰가 표시됩니다. Explorer에서 OpsData에는 EC2 인스턴스에 대한 메타데이터, 패치 규정 준수 세부 정보 및 운영 작업 항목(OpsItems)이 포함됩니다. Explorer은 사업부 또는 애플리케이션에 OpsItems이 분배되는 되는 방식, 시간 경과에 따른 추세 및 범주별 차이점에 대한 컨텍스트를 제공합니다. Explorer에서 정보를 그룹화 및 필터링하여 사용자와 관련이 있고 작업이 필요한 항목에 초점을 맞출 수 있습니다. 자세한 내용은 AWS Systems Manager Explorer 단원을 참조하십시오.

AWS Systems Manager OpsCenter

AWS Systems Manager의 기능인 OpsCenter는 운영 엔지니어와 IT 전문가가 AWS 리소스와 관련된 운영 작업 항목(OpsItems)을 보고, 조사하고, 해결할 수 있는 중앙 위치를 제공합니다. OpsCenter는 각 OpsItem, 관련 OpsItems 및 관련 리소스에 대한 컨텍스트별 조사 데이터를 제공하면서 서비스 전반에 걸쳐 OpsItems를 집계하고 표준화합니다. 또한 OpsCenter는 문제를 신속하게 해결하는 데 사용할 수 있는 AWS Systems Manager의 기능인 Automation 실행서를 제공합니다. OpsCenter는 Amazon EventBridge와 통합되어 있습니다. 즉, EventBridge에 이벤트를 게시하는 모든 AWS 서비스에 대해 OpsItems을(를) 자동으로 만드는 EventBridge 규칙을 생성할 수 있습니다. 자세한 내용은 AWS Systems Manager OpsCenter 단원을 참조하십시오.

Amazon Simple Notification Service

AWS Systems Manager의 기능인 Run Command 또는 Maintenance Windows를 사용하여 보내는 명령의 상태에 대한 알림을 보내도록 Amazon Simple Notification Service(Amazon SNS)를 구성할 수 있습니다. Amazon SNS는 Amazon SNS 주제를 구독하는 클라이언트 또는 엔드포인트에 알림을 보내고 전송하는 작업을 조정하고 관리합니다. 명령이 새로운 상태로 바뀌거나 `Failed` 또는 `Timed Out` 같은 특정 상태가 될 때마다 알림을 받을 수 있습니다. 여러 노드에 명령을 보내는 경우, 특정 노드로 보낸 각각의 명령 사본에 대해 알림을 받을 수 있습니다. 자세한 내용은 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 단원을 참조하십시오.

AWS Trusted Advisor 및 AWS Health Dashboard

Trusted Advisor는 수십만 명의 AWS 고객에게 서비스를 제공하면서 익힌 모범 사례를 활용합니다. Trusted Advisor는 AWS 환경을 검사한 후 비용 절감, 시스템 가용성 및 성능 향상 또는 보안 격차를 해결할 기회가 있을 때 권장 사항을 제시합니다. 모든 AWS 고객은 5개의 Trusted Advisor 점검 항

목에 액세스할 수 있습니다. AWS Support Business 또는 Enterprise Support 플랜을 보유한 고객은 모든 Trusted Advisor 점검 항목을 볼 수 있습니다. 자세한 내용은 AWS Support 사용 설명서의 AWS Trusted Advisor 및 AWS Health 사용 설명서를 참조하세요.

추가 정보

- AWS Systems Manager 모니터링

AWS Systems Manager의 규정 준수 확인

이 주제에서는 타사 보증 프로그램을 규정 준수하는 AWS Systems Manager에 대해 설명합니다. 관리형 노드의 규정 준수 데이터를 보는 방법에 대한 자세한 내용은 섹션을 참조하세요.AWS Systems Manager Compliance

서드 파티 감사자는 여러 AWS 규정 준수 프로그램의 일환으로 Systems Manager 서비스의 보안 및 규정 준수를 평가합니다. 여기에는 SOC, PCI, FedRAMP, HIPAA 등이 포함됩니다.

특정 규정 준수 프로그램의 범위 내에 있는 AWS 서비스 목록은 규정 준수 프로그램 제공 범위 내 AWS 서비스 을(를) 참조하세요. 일반적인 정보는 AWS 규정 준수 프로그램을 참조합니다.

AWS Artifact를 사용하여 제3자 감사 보고서를 다운로드할 수 있습니다. 자세한 내용은 AWS Artifact에서 보고서 다운로드를 참조하세요.

Systems Manager 사용 시 규정 준수 책임은 데이터의 민감도, 회사의 규정 준수 목표 및 관련 법률과 규정에 따라 결정됩니다.AWS에서는 규정 준수를 지원할 다음과 같은 리소스를 제공합니다.

- 보안 및 규정 준수 빠른 시작 안내서 – 이 배포 안내서에서는 아키텍처 고려 사항에 관해 설명하고 AWS에서 보안 및 규정 준수에 중점을 둔 기본 환경을 배포하기 위한 단계를 제공합니다.
- HIPAA 보안 및 규정 준수 백서 설계 - 이 백서에서는 기업에서 AWS를 사용하여 HIPAA를 준수하는 애플리케이션을 생성하는 방법을 설명합니다.
- AWS 규정 준수 리소스 - 고객 조직이 속한 산업 및 위치에 적용될 수 있는 워크북 및 안내서 콜렉션입니다.
- AWS Config 개발자 안내서의 규칙을 사용하여 리소스 평가 – AWS Config 서비스는 내부 사례, 산업 지침 및 규제에 대한 리소스 구성의 준수 상태를 평가합니다.
- AWS Security Hub - 이 AWS 서비스는 보안 산업 표준 및 모범 사례 규정 준수 여부를 확인하는 데 도움이 되도록 AWS 내 보안 상태를 종합적으로 보여줍니다.

AWS Systems Manager의 복원성

AWS 글로벌 인프라는 AWS 리전 및 가용 영역을 중심으로 구축됩니다. AWS 리전에서는 물리적으로 분리되고 격리된 다수의 가용 영역을 제공하며 이러한 가용 영역은 짧은 대기 시간, 높은 처리량 및 높은 중복성을 갖춘 네트워크에 연결되어 있습니다. 가용 영역을 사용하면 중단 없이 영역 간에 자동으로 장애 극복 조치가 이루어지는 애플리케이션 및 데이터베이스를 설계하고 운영할 수 있습니다. 가용 영역은 기존의 단일 또는 다중 데이터 센터 인프라보다 가용성, 내결함성, 확장성이 뛰어납니다.

AWS 리전 및 가용 영역에 대한 자세한 내용은 AWS 글로벌 인프라 섹션을 참조하세요.

AWS Systems Manager에서 인프라 보안

관리형 서비스인 AWS Systems Manager은 AWS 글로벌 네트워크 보안으로 보호됩니다. AWS 보안 서비스와 AWS의 인프라 보호 방법에 대한 자세한 내용은 AWS 클라우드 보안을 참조하세요. 인프라 보안에 대한 모범 사례를 사용하여 AWS 환경을 설계하려면 보안 원칙 AWS Well-Architected 프레임워크의 인프라 보호를 참조하세요.

AWS에서 게시한 API 직접 호출을 사용하여 네트워크를 통해 Systems Manager에 액세스합니다. 고객은 다음을 지원해야 합니다.

- 전송 계층 보안(TLS) TLS 1.2는 필수이며 TLS 1.3을 권장합니다.
- DHE(Ephemeral Diffie-Hellman) 또는 ECDHE(Elliptic Curve Ephemeral Diffie-Hellman)와 같은 완전 전송 보안(PFS)이 포함된 암호 제품군 Java 7 이상의 최신 시스템은 대부분 이러한 모드를 지원합니다.

또한 요청은 액세스 키 ID 및 IAM 주체와 관련된 비밀 액세스 키를 사용하여 서명해야 합니다. 또는 AWS Security Token Service(AWS STS)를 사용하여 임시 보안 인증 정보를 생성하여 요청에 서명할 수 있습니다.

AWS Systems Manager의 구성 및 취약성 분석

AWS는 방화벽 구성, 재해 복구와 같은 기본 보안 태스크를 처리합니다. 적합한 제3자가 이 절차를 검토하고 인증하였습니다. 자세한 내용은 다음 리소스를 참조하세요.

- AWS Systems Manager의 규정 준수 확인
- 공동 책임 모델
- 보안, 자격 증명 및 규정 준수를 위한 모범 사례

Systems Manager의 보안 모범 사례

AWS Systems Manager는 자체 보안 정책을 개발하고 구현할 때 고려해야 할 여러 보안 기능을 제공합니다. 다음 모범 사례는 일반적인 지침이며 완벽한 보안 솔루션을 나타내지는 않습니다. 이러한 모범 사례는 환경에 적절하지 않거나 충분하지 않을 수 있으므로 참고용으로만 사용해 주십시오.

주제

Systems Manager 예방적 보안 모범 사례

다음과 같은 Systems Manager 모범 사례를 통해 보안 사고를 예방할 수 있습니다.

최소 권한 액세스 구현

권한을 부여할 때 누가 어떤 Systems Manager 리소스에 대해 어떤 권한을 갖는지 결정합니다. 해당 리소스에서 허용할 특정 작업을 허용합니다. 따라서 작업을 수행하는 데 필요한 권한만 부여해야 합니다. 최소 권한 액세스를 구현하는 것이 오류 또는 악의적인 의도로 인해 발생할 수 있는 보안 위험과 영향을 최소화할 수 있는 근본적인 방법입니다.

다음과 같은 도구를 사용하여 최소 권한 액세스를 구현할 수 있습니다.

- IAM 정책 및 IAM 엔터티에 대한 권한 경계
- 서비스 제어 정책

프록시를 사용하도록 구성된 경우 SSM Agent의 권장 설정 사용

프록시를 사용하도록 SSM Agent를 구성하는 경우 Systems Manager 인스턴스 메타데이터 서비스의 IP 주소와 함께 `no_proxy` 변수를 사용하여 Systems Manager에 대한 직접 호출 시 프록시 서비스의 자격 증명을 를 사용하지 않도록 합니다.

자세한 내용은 SSM Agent를 구성하여 Linux 노드에 프록시 사용 및 Windows Server 인스턴스에 프록시를 사용하도록 SSM Agent 구성 단원을 참조하세요.

SecureString 파라미터를 사용하여 보안 암호 데이터 암호화 및 보호

AWS Systems Manager의 기능인 Parameter Store에서 `SecureString` 파라미터는 안전한 방식으로 저장되고 참조되어야 하는 모든 민감한 데이터를 뜻합니다. 암호나 라이선스 키처럼 사용자가 일반 텍스트로 변경하거나 참조하지 않아야 하는 데이터가 있는 경우 `SecureString` 데이터 형식을 사용하여 해당 파라미터를 생성합니다. Parameter Store는 AWS Key Management

Service(AWS KMS)의 AWS KMS key를 사용하여 파라미터 값을 암호화합니다. AWS KMS는 파라미터 값을 암호화할 때 고객 관리형 키 또는 AWS 관리형 키를 사용합니다. 보안을 최대한 강화하기 위해서는 자체 KMS 키를 사용하는 것이 좋습니다. AWS 관리형 키를 사용하는 경우 계정에서 GetParameter 및 GetParameters 작업을 실행할 권한이 있는 사용자는 모든 SecureString 파라미터의 콘텐츠를 보거나 검색할 수 있습니다. 고객 관리형 키를 사용하여 보안 SecureString 값을 암호화하는 경우 IAM 정책 및 키 정책을 사용하여 파라미터의 암호화 및 복호화를 위한 권한을 관리할 수 있습니다. 고객 관리형 키를 사용할 때는 이러한 작업에 대한 액세스 제어 정책을 설정하기가 더 어렵습니다. 예를 들어 AWS 관리형 키를 사용하여 SecureString 파라미터를 암호화하고 사용자가 SecureString 파라미터로 작업하지 않도록 하려면, IAM 정책에서 기본 키에 대한 액세스를 명시적으로 거부해야 합니다.

자세한 내용은 AWS Key Management Service Developer Guide의 IAM 정책을 사용하여 Systems Manager 파라미터에 대한 액세스 제한 및 How AWS Systems ManagerParameter Store Uses AWS KMS를 참조하세요.

문서 파라미터에 대한 allowedValues 및 allowedPattern 정의

allowedValues 및 allowedPattern을 정의하여 Systems Manager 문서(SSM 문서)의 파라미터에 대한 사용자 입력을 검증할 수 있습니다. allowedValues의 경우, 파라미터에서 허용되는 값 배열을 정의합니다. 사용자가 허용되지 않는 값을 입력하면 실행이 시작되지 않습니다. allowedPattern의 경우, 사용자 입력이 파라미터에 대해 정의된 패턴과 일치하는지 여부를 확인하는 정규 표현식을 정의합니다. 사용자 입력이 허용된 패턴과 일치하지 않으면 실행이 시작되지 않습니다.

allowedValues 및 allowedPattern에 대한 자세한 내용은 데이터 요소 및 파마미터 섹션을 참조하세요.

문서의 퍼블릭 공유 차단

사용 사례에서 퍼블릭 공유를 허용해야 하는 경우가 아니면 Systems Manager Documents 콘솔의 [기본 설정(Preferences)] 섹션에서 SSM 문서의 퍼블릭 공유 차단 설정을 켜는 것이 좋습니다.

Amazon Virtual Private Cloud(Amazon VPC) 및 VPC 엔드포인트 사용

Amazon VPC를 사용하여 사용자가 정의한 가상 네트워크에서 AWS 리소스를 시작할 수 있습니다. 이 가상 네트워크는 AWS의 확장 가능한 인프라를 사용한다는 이점과 함께 고객의 자체 데이터 센터에서 운영하는 기존 네트워크와 매우 유사합니다.

VPC 엔드포인트를 구현하면 인터넷 게이트웨이, NAT 디바이스, VPN 연결 또는 AWS Direct Connect 연결 없이도 AWS PrivateLink을(를) 통해 지원되는 AWS 서비스 및 VPC 엔드포인트 서비스에 VPC를 비공개로 연결할 수 있습니다. VPC의 인스턴스는 서비스의 리소스와 통신하는 데 퍼

블릭 IP 주소를 필요로 하지 않습니다. VPC와 기타 서비스 간의 트래픽은 Amazon 네트워크를 벗어나지 않습니다.

Amazon VPC 보안에 대한 자세한 내용은 Amazon VPC 사용 설명서의 Systems Manager용 VPC 엔드포인트를 사용하여 EC2 인스턴스의 보안 개선 및 Amazon VPC의 네트워크간 트래픽 개인 정보 보호를 참조하세요.

대화형 명령 및 특정 SSM 세션 문서를 사용하여 Session Manager 사용자를 세션으로 제한

AWS Systems Manager의 기능인 Session Manager는 관리형 노드에 대한 세션을 시작하는 몇 가지 방법을 제공합니다. 가장 안전한 연결을 위해 사용자가 대화형 명령 방법을 통해 연결하여 사용자 상호 작용을 특정 명령 또는 명령 시퀀스로 제한하도록 요구할 수 있습니다. 이렇게 하면 사용자가 수행할 수 있는 대화형 작업을 관리할 수 있습니다. 자세한 내용은 세션 시작(대화형 및 비대화형 명령) 단원을 참조하십시오.

보안 강화를 위해 특정 Amazon EC2 인스턴스 및 특정 Session Manager 세션 문서에 대한 Session Manager의 액세스를 제한할 수 있습니다. AWS Identity and Access Management(IAM) 정책을 사용하여 이 방식으로 Session Manager 액세스 권한을 부여하거나 취소할 수 있습니다. 자세한 내용은 3단계: 관리형 노드에 대한 세션 액세스 제어 단원을 참조하십시오.

Automation 워크플로에 대한 임시 노드 권한 제공

AWS Systems Manager의 기능인 Automation의 워크플로 중 노드에 해당 실행에만 필요하고 다른 Systems Manager 작업에는 필요하지 않은 권한이 필요할 수 있습니다. 예를 들어 Automation 워크플로에서 노드가 특정 API 작업을 호출하거나 워크플로 중에 특별히 AWS 리소스에 액세스해야 할 수 있습니다. 이러한 호출이나 리소스에 대한 액세스를 제한하려는 경우 IAM 인스턴스 프로파일에 권한을 추가하는 대신 Automation 런북 자체 내에서 노드에 대한 임시 보충 권한을 제공할 수 있습니다. Automation 워크플로가 끝나면 임시 권한이 제거됩니다. 자세한 내용은 AWS Management and Governance Blog의 Providing temporary instance permissions with AWS Systems Manager Automations를 참조하세요.

최신 상태로 AWS 및 Systems Manager도구 유지

AWS는 AWS 및 Systems Manager 작업에 사용할 수 있는 업데이트된 버전의 도구 및 플러그인을 정기적으로 출시합니다. 이러한 리소스를 최신 상태로 유지하면 해당 계정의 사용자와 노드가 이러한 도구의 최신 기능과 보안 기능에 액세스할 수 있습니다.

- SSM Agent - AWS Systems Manager 에이전트(SSM Agent)는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스, 온프레미스 서버 또는 가상 머신(VM)에 설치 및 구성할 수 있는 Amazon 소프트웨어입니다. SSM Agent를 사용하면 Systems Manager에서 이러한 리소스를 업데이트, 관리 및 구성할 수 있습니다. 최소한 2주마다 새 버전을 확인하거나 에이전트 업데이트를 자동화하는 것이 좋습니다. 자세한 설명은 SSM Agent 업데이트 자동화을 참조하세요. 또한 업데

이트 프로세스의 일부로 SSM Agent 서명을 확인하는 것이 좋습니다. 자세한 설명은 SSM Agent의 서명 확인을 참조하세요.

- AWS CLI - AWS Command Line Interface(AWS CLI)은(는) 명령줄 셸의 명령을 사용하여 AWS 서비스와(과) 상호 작용할 수 있는 오픈 소스 도구입니다. AWS CLI를 업데이트하려면 AWS CLI를 설치하는 데 사용한 것과 동일한 명령을 실행합니다. 로컬 컴퓨터에서 예약된 작업을 생성하여 운영 체제에 적합한 명령을 최소한 2주에 한 번씩 실행하는 것이 좋습니다. 설치 명령에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서의 AWS CLI 버전 2 설치를 참조하세요.
- AWS Tools for Windows PowerShell - Tools for Windows PowerShell은 AWS SDK for .NET에서 공개하는 기능을 기반으로 구축된 PowerShell 모듈 집합입니다. AWS Tools for Windows PowerShell을 사용하면 PowerShell 명령줄에서 AWS 리소스에 대한 작업을 스크립팅할 수 있습니다. 정기적으로 Tools for Windows PowerShell의 업데이트된 버전이 릴리스될 때 로컬로 실행 중인 버전을 업데이트해야 합니다. 자세한 내용은 IAM 정책 시뮬레이터 사용 설명서의 Windows에서 AWS Tools for Windows PowerShell 업데이트 또는 Linux macOS에서 AWS Tools for Windows PowerShell 업데이트를 참조하세요.
- Session Manager 플러그 인 – 조직에서 Session Manager에 대한 사용 권한이 있는 사용자가 AWS CLI를 사용하여 노드에 연결하려는 경우 먼저 로컬 시스템에 Session Manager 플러그 인을 설치해야 합니다. 플러그인을 업데이트하려면 플러그인을 설치하는 데 사용한 것과 동일한 명령을 실행합니다. 로컬 컴퓨터에서 예약된 작업을 생성하여 운영 체제에 적합한 명령을 최소한 2주에 한 번씩 실행하는 것이 좋습니다. 자세한 설명은 AWS CLI의 Session Manager 플러그인 설치을 참조하세요.
- CloudWatch 에이전트 - CloudWatch 에이전트를 구성하고 사용하여 EC2 인스턴스, 온프레미스 인스턴스 및 가상 머신(VM)에서 지표와 로그를 수집할 수 있습니다. 이러한 로그는 모니터링 및 분석을 위해 Amazon CloudWatch Logs로 전송할 수 있습니다. 최소한 2주마다 새 버전을 확인하거나 에이전트 업데이트를 자동화하는 것이 좋습니다. 간단하게 업데이트하려면 AWS Systems Manager 빠른 설치를 사용하십시오. 자세한 설명은 AWS Systems Manager Quick Setup을 참조하세요.

Systems Manager 모니터링 및 감사 모범 사례

다음과 같은 Systems Manager 모범 사례가 잠재적 보안 약점과 사고를 탐지하는 데 도움이 됩니다.

모든 Systems Manager 리소스 식별 및 감사

IT 자산 식별은 거버넌스와 보안의 중요한 측면입니다. 모든 Systems Manager 리소스를 식별하여 보안 상태를 평가하고 잠재적 취약 영역에 대해 조치를 취해야 합니다.

Tag Editor를 사용하여 보안이나 감사에 민감한 리소스를 식별한 후, 이 리소스를 검색해야 할 때 태그를 이용하세요. 자세한 내용은 AWS Resource Groups 사용 설명서의 태그를 지정할 리소스 찾기를 참조하세요.

Systems Manager 리소스의 리소스 그룹을 만드십시오. 자세한 내용은 Resource Groups이란 무엇인가요?를 참조하세요.

Amazon CloudWatch 모니터링 도구를 사용하여 모니터링 구현

모니터링은 Systems Manager와 사용자 AWS 솔루션의 안정성, 보안, 가용성 및 성능을 유지하는 중요한 역할을 합니다. Amazon CloudWatch는 Systems Manager와(과) 다른 AWS 서비스을(를) 모니터링하는 데 도움이 되는 몇 가지 도구와 서비스를 제공합니다. 자세한 내용은 통합 CloudWatch Logs로 노드 로그 전송(CloudWatch 에이전트) 및 Amazon EventBridge로 Systems Manager 이벤트 모니터링을(를) 참조하세요.

CloudTrail 사용

AWS CloudTrail은(는) Systems Manager에서 사용자, 역할 또는 AWS 서비스이(가) 수행한 작업에 대한 기록을 제공합니다. CloudTrail에서 수집한 정보를 사용하여 Systems Manager에 수행된 요청, 요청이 수행된 IP 주소, 요청을 수행한 사람, 요청이 수행된 시간 및 추가 세부 정보를 확인할 수 있습니다. 자세한 내용은 AWS CloudTrail을 사용하여 AWS Systems ManagerAPI 호출을 로깅 단원을 참조하십시오.

AWS Config 켜기

AWS Config를 사용하면 AWS 리소스의 구성을 평가, 감사 및 측정할 수 있습니다. AWS Config는 리소스 구성을 모니터링하여 필요한 보안 구성을 기준으로 기록된 구성을 평가할 수 있게 합니다. AWS Config를 사용하면 AWS 리소스 간 구성 및 관계 변화를 검토하고, 자세한 리소스 구성 기록을 조사하고, 내부 지침에 지정되어 있는 구성을 기준으로 전반적인 규정 준수 여부를 확인할 수 있습니다. 이를 사용하면 규정 준수 감사, 보안 분석, 변경 관리 및 운영 문제 해결 작업을 간소화할 수 있습니다. 자세한 내용은 AWS Config 개발자 안내서의 콘솔을 통해 AWS Config 설정을 참조하세요. 기록할 리소스 유형을 지정할 때 Systems Manager 리소스를 포함해야 합니다.

AWS 보안 공지 모니터링

AWS 계정의 Trusted Advisor에 게시되는 보안 권고 사항을 정기적으로 확인해야 합니다. 이는 describe-trusted-advisor-checks를 사용하여 프로그래밍 방식으로 수행하면 됩니다.

뿐만 아니라, 각 AWS 계정에 등록된 기본 이메일 주소를 적극적으로 모니터링하세요. 사용자에게 영향을 줄 수 있는 보안 문제가 생기면 AWS에서 이 이메일 주소를 사용하여 연락 드립니다.

널리 영향을 미치는 AWS 운영 문제는 AWS Service Health Dashboard에 게시됩니다. Personal Health Dashboard를 통해 개별 계정에도 운영 문제가 게시됩니다. 자세한 내용은 AWS Health 설명서를 참조하세요.

추가 정보

- 보안, 자격 증명 및 규정 준수를 위한 모범 사례
- Getting Started: Follow Security Best Practices as You Configure Your AWS Resources(AWS Security Blog)
- IAM의 보안 모범 사례
- AWS CloudTrail의 보안 모범 사례
- Amazon S3의 보안 모범 사례
- AWS Key Management Service의 보안 모범 사례

AWS SDK를 사용한 Systems Manager의 코드 예제

다음 코드 예제에서는 Systems Manager를 AWS 소프트웨어 개발 키트(SDK)와 함께 사용하는 방법을 보여줍니다.

작업은 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 작업은 개별 서비스 함수를 호출하는 방법을 보여 주며 관련 시나리오와 교차 서비스 예시에서 컨텍스트에 맞는 작업을 볼 수 있습니다.

시나리오는 동일한 서비스 내에서 여러 함수를 호출하여 특정 태스크를 수행하는 방법을 보여주는 코드 예시입니다.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 섹션을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

시작하기

Hello Systems Manager

다음 코드 예제에서는 Systems Manager를 사용하여 시작하는 방법을 보여줍니다.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS코드 예시 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
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);
    }
}
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 listThings를 참조하세요.

코드 예시

- AWS SDK를 사용한 Systems Manager의 작업
  - AWS SDK 또는 CLI와 함께 AddTagsToResource 사용
  - AWS SDK 또는 CLI와 함께 CancelCommand 사용
  - AWS SDK 또는 CLI와 함께 CreateActivation 사용
  - AWS SDK 또는 CLI와 함께 CreateAssociation 사용
  - AWS SDK 또는 CLI와 함께 CreateAssociationBatch 사용
  - AWS SDK 또는 CLI와 함께 CreateDocument 사용
  - AWS SDK 또는 CLI와 함께 CreateMaintenanceWindow 사용
  - AWS SDK 또는 CLI와 함께 CreateOpsItem 사용
  - AWS SDK 또는 CLI와 함께 CreatePatchBaseline 사용
  - AWS SDK 또는 CLI와 함께 DeleteActivation 사용
  - AWS SDK 또는 CLI와 함께 DeleteAssociation 사용
  - AWS SDK 또는 CLI와 함께 DeleteDocument 사용
  - AWS SDK 또는 CLI와 함께 DeleteMaintenanceWindow 사용
  - AWS SDK 또는 CLI와 함께 DeleteParameter 사용
  - AWS SDK 또는 CLI와 함께 DeletePatchBaseline 사용
  - AWS SDK 또는 CLI와 함께 DeregisterManagedInstance 사용
  - AWS SDK 또는 CLI와 함께 DeregisterPatchBaselineForPatchGroup 사용
  - AWS SDK 또는 CLI와 함께 DeregisterTargetFromMaintenanceWindow 사용
  - AWS SDK 또는 CLI와 함께 DeregisterTaskFromMaintenanceWindow 사용
  - AWS SDK 또는 CLI와 함께 DescribeActivations 사용

- AWS SDK 또는 CLI와 함께 DescribeAssociation 사용
- AWS SDK 또는 CLI와 함께 DescribeAssociationExecutionTargets 사용
- AWS SDK 또는 CLI와 함께 DescribeAssociationExecutions 사용
- AWS SDK 또는 CLI와 함께 DescribeAutomationExecutions 사용
- AWS SDK 또는 CLI와 함께 DescribeAutomationStepExecutions 사용
- AWS SDK 또는 CLI와 함께 DescribeAvailablePatches 사용
- AWS SDK 또는 CLI와 함께 DescribeDocument 사용
- AWS SDK 또는 CLI와 함께 DescribeDocumentPermission 사용
- AWS SDK 또는 CLI와 함께 DescribeEffectiveInstanceAssociations 사용
- AWS SDK 또는 CLI와 함께 DescribeEffectivePatchesForPatchBaseline 사용
- AWS SDK 또는 CLI와 함께 DescribeInstanceAssociationsStatus 사용
- AWS SDK 또는 CLI와 함께 DescribeInstanceInformation 사용
- AWS SDK 또는 CLI와 함께 DescribeInstancePatchStates 사용
- AWS SDK 또는 CLI와 함께 DescribeInstancePatchStatesForPatchGroup 사용
- AWS SDK 또는 CLI와 함께 DescribeInstancePatches 사용
- AWS SDK 또는 CLI와 함께 DescribeMaintenanceWindowExecutionTaskInvocations 사용
- AWS SDK 또는 CLI와 함께 DescribeMaintenanceWindowExecutionTasks 사용
- AWS SDK 또는 CLI와 함께 DescribeMaintenanceWindowExecutions 사용
- AWS SDK 또는 CLI와 함께 DescribeMaintenanceWindowTargets 사용
- AWS SDK 또는 CLI와 함께 DescribeMaintenanceWindowTasks 사용
- AWS SDK 또는 CLI와 함께 DescribeMaintenanceWindows 사용
- AWS SDK 또는 CLI와 함께 DescribeOpsItems 사용
- AWS SDK 또는 CLI와 함께 DescribeParameters 사용
- AWS SDK 또는 CLI와 함께 DescribePatchBaselines 사용
- AWS SDK 또는 CLI와 함께 DescribePatchGroupState 사용
- AWS SDK 또는 CLI와 함께 DescribePatchGroups 사용
- AWS SDK 또는 CLI와 함께 GetAutomationExecution 사용
- AWS SDK 또는 CLI와 함께 GetCommandInvocation 사용
- AWS SDK 또는 CLI와 함께 GetConnectionStatus 사용
- AWS SDK 또는 CLI와 함께 GetDefaultPatchBaseline 사용

- AWS SDK 또는 CLI와 함께 GetDeployablePatchSnapshotForInstance 사용
- AWS SDK 또는 CLI와 함께 GetDocument 사용
- AWS SDK 또는 CLI와 함께 GetInventory 사용
- AWS SDK 또는 CLI와 함께 GetInventorySchema 사용
- AWS SDK 또는 CLI와 함께 GetMaintenanceWindow 사용
- AWS SDK 또는 CLI와 함께 GetMaintenanceWindowExecution 사용
- AWS SDK 또는 CLI와 함께 GetMaintenanceWindowExecutionTask 사용
- AWS SDK 또는 CLI와 함께 GetParameterHistory 사용
- AWS SDK 또는 CLI와 함께 GetParameters 사용
- AWS SDK 또는 CLI와 함께 GetPatchBaseline 사용
- AWS SDK 또는 CLI와 함께 GetPatchBaselineForPatchGroup 사용
- AWS SDK 또는 CLI와 함께 ListAssociationVersions 사용
- AWS SDK 또는 CLI와 함께 ListAssociations 사용
- AWS SDK 또는 CLI와 함께 ListCommandInvocations 사용
- AWS SDK 또는 CLI와 함께 ListCommands 사용
- AWS SDK 또는 CLI와 함께 ListComplianceItems 사용
- AWS SDK 또는 CLI와 함께 ListComplianceSummaries 사용
- AWS SDK 또는 CLI와 함께 ListDocumentVersions 사용
- AWS SDK 또는 CLI와 함께 ListDocuments 사용
- AWS SDK 또는 CLI와 함께 ListInventoryEntries 사용
- AWS SDK 또는 CLI와 함께 ListResourceComplianceSummaries 사용
- AWS SDK 또는 CLI와 함께 ListTagsForResource 사용
- AWS SDK 또는 CLI와 함께 ModifyDocumentPermission 사용
- AWS SDK 또는 CLI와 함께 PutComplianceItems 사용
- AWS SDK 또는 CLI와 함께 PutInventory 사용
- AWS SDK 또는 CLI와 함께 PutParameter 사용
- AWS SDK 또는 CLI와 함께 RegisterDefaultPatchBaseline 사용
- AWS SDK 또는 CLI와 함께 RegisterPatchBaselineForPatchGroup 사용
- AWS SDK 또는 CLI와 함께 RegisterTargetWithMaintenanceWindow 사용
- AWS SDK 또는 CLI와 함께 RegisterTaskWithMaintenanceWindow 사용

- AWS SDK 또는 CLI와 함께 RemoveTagsFromResource 사용
- AWS SDK 또는 CLI와 함께 SendCommand 사용
- AWS SDK 또는 CLI와 함께 StartAutomationExecution 사용
- AWS SDK 또는 CLI와 함께 StopAutomationExecution 사용
- AWS SDK 또는 CLI와 함께 UpdateAssociation 사용
- AWS SDK 또는 CLI와 함께 UpdateAssociationStatus 사용
- AWS SDK 또는 CLI와 함께 UpdateDocument 사용
- AWS SDK 또는 CLI와 함께 UpdateDocumentDefaultVersion 사용
- AWS SDK 또는 CLI와 함께 UpdateMaintenanceWindow 사용
- AWS SDK 또는 CLI와 함께 UpdateManagedInstanceRole 사용
- AWS SDK 또는 CLI와 함께 UpdateOpsItem 사용
- AWS SDK 또는 CLI와 함께 UpdatePatchBaseline 사용

- AWS SDK를 사용한 Systems Manager 시나리오
  - AWS SDK를 사용하여 Systems Manager 시작하기

AWS SDK를 사용한 Systems Manager의 작업

다음 코드 예제에서는 AWS SDK를 통해 개별 Systems Manager 작업을 수행하는 방법을 보여줍니다. 이 발췌문은 Systems Manager API를 직접 호출하며, 컨텍스트에서 실행되어야 하는 더 큰 프로그램에서 발췌한 코드입니다. 각 예제에는 GitHub에 대한 링크가 포함되어 있습니다. 여기에서 코드 설정 및 실행에 대한 지침을 찾을 수 있습니다.

다음 예제에는 가장 일반적으로 사용되는 작업만 포함되어 있습니다. 전체 목록은 AWS Systems ManagerAPI 참조를 참조하세요.

예제

- AWS SDK 또는 CLI와 함께 AddTagsToResource 사용
- AWS SDK 또는 CLI와 함께 CancelCommand 사용
- AWS SDK 또는 CLI와 함께 CreateActivation 사용
- AWS SDK 또는 CLI와 함께 CreateAssociation 사용
- AWS SDK 또는 CLI와 함께 CreateAssociationBatch 사용
- AWS SDK 또는 CLI와 함께 CreateDocument 사용
- AWS SDK 또는 CLI와 함께 CreateMaintenanceWindow 사용

- AWS SDK 또는 CLI와 함께 CreateOpsItem 사용
- AWS SDK 또는 CLI와 함께 CreatePatchBaseline 사용
- AWS SDK 또는 CLI와 함께 DeleteActivation 사용
- AWS SDK 또는 CLI와 함께 DeleteAssociation 사용
- AWS SDK 또는 CLI와 함께 DeleteDocument 사용
- AWS SDK 또는 CLI와 함께 DeleteMaintenanceWindow 사용
- AWS SDK 또는 CLI와 함께 DeleteParameter 사용
- AWS SDK 또는 CLI와 함께 DeletePatchBaseline 사용
- AWS SDK 또는 CLI와 함께 DeregisterManagedInstance 사용
- AWS SDK 또는 CLI와 함께 DeregisterPatchBaselineForPatchGroup 사용
- AWS SDK 또는 CLI와 함께 DeregisterTargetFromMaintenanceWindow 사용
- AWS SDK 또는 CLI와 함께 DeregisterTaskFromMaintenanceWindow 사용
- AWS SDK 또는 CLI와 함께 DescribeActivations 사용
- AWS SDK 또는 CLI와 함께 DescribeAssociation 사용
- AWS SDK 또는 CLI와 함께 DescribeAssociationExecutionTargets 사용
- AWS SDK 또는 CLI와 함께 DescribeAssociationExecutions 사용
- AWS SDK 또는 CLI와 함께 DescribeAutomationExecutions 사용
- AWS SDK 또는 CLI와 함께 DescribeAutomationStepExecutions 사용
- AWS SDK 또는 CLI와 함께 DescribeAvailablePatches 사용
- AWS SDK 또는 CLI와 함께 DescribeDocument 사용
- AWS SDK 또는 CLI와 함께 DescribeDocumentPermission 사용
- AWS SDK 또는 CLI와 함께 DescribeEffectiveInstanceAssociations 사용
- AWS SDK 또는 CLI와 함께 DescribeEffectivePatchesForPatchBaseline 사용
- AWS SDK 또는 CLI와 함께 DescribeInstanceAssociationsStatus 사용
- AWS SDK 또는 CLI와 함께 DescribeInstanceInformation 사용
- AWS SDK 또는 CLI와 함께 DescribeInstancePatchStates 사용
- AWS SDK 또는 CLI와 함께 DescribeInstancePatchStatesForPatchGroup 사용
- AWS SDK 또는 CLI와 함께 DescribeInstancePatches 사용
- AWS SDK 또는 CLI와 함께 DescribeMaintenanceWindowExecutionTaskInvocations 사용

- AWS SDK 또는 CLI와 함께 DescribeMaintenanceWindowExecutionTasks 사용
- AWS SDK 또는 CLI와 함께 DescribeMaintenanceWindowExecutions 사용
- AWS SDK 또는 CLI와 함께 DescribeMaintenanceWindowTargets 사용
- AWS SDK 또는 CLI와 함께 DescribeMaintenanceWindowTasks 사용
- AWS SDK 또는 CLI와 함께 DescribeMaintenanceWindows 사용
- AWS SDK 또는 CLI와 함께 DescribeOpsItems 사용
- AWS SDK 또는 CLI와 함께 DescribeParameters 사용
- AWS SDK 또는 CLI와 함께 DescribePatchBaselines 사용
- AWS SDK 또는 CLI와 함께 DescribePatchGroupState 사용
- AWS SDK 또는 CLI와 함께 DescribePatchGroups 사용
- AWS SDK 또는 CLI와 함께 GetAutomationExecution 사용
- AWS SDK 또는 CLI와 함께 GetCommandInvocation 사용
- AWS SDK 또는 CLI와 함께 GetConnectionStatus 사용
- AWS SDK 또는 CLI와 함께 GetDefaultPatchBaseline 사용
- AWS SDK 또는 CLI와 함께 GetDeployablePatchSnapshotForInstance 사용
- AWS SDK 또는 CLI와 함께 GetDocument 사용
- AWS SDK 또는 CLI와 함께 GetInventory 사용
- AWS SDK 또는 CLI와 함께 GetInventorySchema 사용
- AWS SDK 또는 CLI와 함께 GetMaintenanceWindow 사용
- AWS SDK 또는 CLI와 함께 GetMaintenanceWindowExecution 사용
- AWS SDK 또는 CLI와 함께 GetMaintenanceWindowExecutionTask 사용
- AWS SDK 또는 CLI와 함께 GetParameterHistory 사용
- AWS SDK 또는 CLI와 함께 GetParameters 사용
- AWS SDK 또는 CLI와 함께 GetPatchBaseline 사용
- AWS SDK 또는 CLI와 함께 GetPatchBaselineForPatchGroup 사용
- AWS SDK 또는 CLI와 함께 ListAssociationVersions 사용
- AWS SDK 또는 CLI와 함께 ListAssociations 사용
- AWS SDK 또는 CLI와 함께 ListCommandInvocations 사용
- AWS SDK 또는 CLI와 함께 ListCommands 사용
- AWS SDK 또는 CLI와 함께 ListComplianceItems 사용

- AWS SDK 또는 CLI와 함께 ListComplianceSummaries 사용
- AWS SDK 또는 CLI와 함께 ListDocumentVersions 사용
- AWS SDK 또는 CLI와 함께 ListDocuments 사용
- AWS SDK 또는 CLI와 함께 ListInventoryEntries 사용
- AWS SDK 또는 CLI와 함께 ListResourceComplianceSummaries 사용
- AWS SDK 또는 CLI와 함께 ListTagsForResource 사용
- AWS SDK 또는 CLI와 함께 ModifyDocumentPermission 사용
- AWS SDK 또는 CLI와 함께 PutComplianceItems 사용
- AWS SDK 또는 CLI와 함께 PutInventory 사용
- AWS SDK 또는 CLI와 함께 PutParameter 사용
- AWS SDK 또는 CLI와 함께 RegisterDefaultPatchBaseline 사용
- AWS SDK 또는 CLI와 함께 RegisterPatchBaselineForPatchGroup 사용
- AWS SDK 또는 CLI와 함께 RegisterTargetWithMaintenanceWindow 사용
- AWS SDK 또는 CLI와 함께 RegisterTaskWithMaintenanceWindow 사용
- AWS SDK 또는 CLI와 함께 RemoveTagsFromResource 사용
- AWS SDK 또는 CLI와 함께 SendCommand 사용
- AWS SDK 또는 CLI와 함께 StartAutomationExecution 사용
- AWS SDK 또는 CLI와 함께 StopAutomationExecution 사용
- AWS SDK 또는 CLI와 함께 UpdateAssociation 사용
- AWS SDK 또는 CLI와 함께 UpdateAssociationStatus 사용
- AWS SDK 또는 CLI와 함께 UpdateDocument 사용
- AWS SDK 또는 CLI와 함께 UpdateDocumentDefaultVersion 사용
- AWS SDK 또는 CLI와 함께 UpdateMaintenanceWindow 사용
- AWS SDK 또는 CLI와 함께 UpdateManagedInstanceRole 사용
- AWS SDK 또는 CLI와 함께 UpdateOpsItem 사용
- AWS SDK 또는 CLI와 함께 UpdatePatchBaseline 사용

AWS SDK 또는 CLI와 함께 **AddTagsToResource** 사용

다음 코드 예시는 AddTagsToResource의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 유지 관리 기간에 태그를 추가하는 방법

다음 add-tags-to-resource 예제에서는 지정된 유지 관리 기간에 태그를 추가합니다.

```
aws ssm add-tags-to-resource \
    --resource-type "MaintenanceWindow" \
    --resource-id "mw-03eb9db428EXAMPLE" \
    --tags "Key=Stack,Value=Production"
```

이 명령은 출력을 생성하지 않습니다.

예제 2: 파라미터에 태그를 추가하는 방법

다음 add-tags-to-resource 예제에서는 지정된 파라미터에 두 개의 태그를 추가합니다.

```
aws ssm add-tags-to-resource \
    --resource-type "Parameter" \
    --resource-id "My-Parameter" \
    --tags '[{"Key":"Region","Value":"East"},{"Key":"Environment",
 "Value":"Production"}]'
```

이 명령은 출력을 생성하지 않습니다.

예제 3: SSM 문서에 태그를 추가하는 방법

다음 add-tags-to-resource 예제에서는 지정된 문서에 태그를 추가합니다.

```
aws ssm add-tags-to-resource \
    --resource-type "Document" \
    --resource-id "My-Document" \
    --tags "Key=Quarter,Value=Q322"
```

이 명령은 출력을 생성하지 않습니다.

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager 리소스 태그 지정을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 AddTagsToResource를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 유지 관리 기간을 새 태그로 업데이트합니다. 명령이 성공해도 출력은 없습니다. 이 예제에서 사용하는 구문에는 PowerShell 버전 3 이상이 필요합니다.

```
$option1 = @{Key="Stack";Value=@("Production")}
Add-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
 "MaintenanceWindow" -Tag $option1
```

예제 2: PowerShell 버전 2에서 각 태그를 생성하려면 New-Object를 사용해야 합니다. 명령이 성공해도 출력은 없습니다.

```
$tag1 = New-Object Amazon.SimpleSystemsManagement.Model.Tag
$tag1.Key = "Stack"
$tag1.Value = "Production"

Add-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
 "MaintenanceWindow" -Tag $tag1
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 AddTagsToResource를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 `CancelCommand` 사용

다음 코드 예시는 `CancelCommand`의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 모든 인스턴스에 대한 명령을 취소하는 방법

다음 cancel-command 예제에서는 모든 인스턴스에 대해 이미 실행 중인 지정된 명령을 취소하려고 시도합니다.

```
aws ssm cancel-command \
    --command-id "662add3d-5831-4a10-b64a-f2ff3EXAMPLE"
```

이 명령은 출력을 생성하지 않습니다.

예제 2: 특정 인스턴스의 명령을 취소하는 방법

다음 cancel-command 예제에서는 지정된 인스턴스에 대한 명령만 취소하려고 시도합니다.

```
aws ssm cancel-command \
    --command-id "662add3d-5831-4a10-b64a-f2ff3EXAMPLE"
    --instance-ids "i-02573cafcfEXAMPLE"
```

이 명령은 출력을 생성하지 않습니다.

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager 파라미터 태그 지정을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 CancelCommand를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 명령 취소를 시도합니다. 작업이 성공해도 출력은 없습니다.

```
Stop-SSMCommand -CommandId "9ded293e-e792-4440-8e3e-7b8ec5feaa38"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 CancelCommand를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **CreateActivation** 사용

다음 코드 예시는 CreateActivation의 사용 방법을 보여줍니다.

CLI

AWS CLI

관리형 인스턴스 활성화를 생성하는 방법

다음 `create-activation` 예제에서는 관리형 인스턴스 활성화를 생성합니다.

```
aws ssm create-activation \
    --default-instance-name "HybridWebServers" \
    --iam-role "HybridWebServersRole" \
    --registration-limit 5
```

출력:

```
{
    "ActivationId": "5743558d-563b-4457-8682-d16c3EXAMPLE",
    "ActivationCode": "dRmgnYaFv567vEXAMPLE"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 4단계: 하이브리드 환경을 위한 관리형 인스턴스 활성화 생성을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 CreateActivation을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 관리형 인스턴스를 생성합니다.

```
New-SSMActivation -DefaultInstanceName "MyWebServers" -IamRole
 "SSMAutomationRole" -RegistrationLimit 10
```

출력:

```
ActivationCode       ActivationId
--------------       ------------
KWChhOxBTiwDcKE9BlKC 08e51e79-1e36-446c-8e63-9458569c1363
```

- API에 대한 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 CreateActivation을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **CreateAssociation** 사용

다음 코드 예시는 CreateAssociation의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 인스턴스 ID를 사용하여 문서를 연결하는 방법

이 예제에서는 인스턴스 ID를 사용하여 구성 문서를 인스턴스와 연결합니다.

```
aws ssm create-association \
    --instance-id "i-0cb2b964d3e14fd9f" \
    --name "AWS-UpdateSSMAgent"
```

출력:

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

자세한 내용은 AWS Systems Manager API 참조의 CreateAssociation을 참조하세요.

예제 2: 대상을 사용하여 문서를 연결하는 방법

이 예제에서는 대상을 사용하여 구성 문서를 인스턴스와 연결합니다.

```
aws ssm create-association \
    --name "AWS-UpdateSSMAgent" \
    --targets "Key=instanceids,Values=i-0cb2b964d3e14fd9f"
```

출력:

```
{
    "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"
            }
```

```
        ]
    }
}
```

자세한 내용은 AWS Systems Manager API 참조의 CreateAssociation을 참조하세요.

예제 3: 한 번만 실행되는 연결을 생성하는 방법

이 예제에서는 지정된 날짜 및 시간에 한 번만 실행되는 새 연결을 생성합니다. 과거 또는 현재 날짜(처리 시점을 기준으로 해당 날짜가 과거임)에 생성된 연결은 즉시 실행됩니다.

```
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
```

출력:

```
{
    "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"
            }
```

```
        ]
    }
}
```

자세한 내용은 AWS Systems Manager API 참조의 CreateAssociation 또는 AWS Systems Manager 사용 설명서의 참조: Systems Manager의 Cron 및 Rate 표현식을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 CreateAssociation을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 인스턴스 ID를 사용하여 구성 문서를 인스턴스와 연결합니다.

```
New-SSMAssociation -InstanceId "i-0cb2b964d3e14fd9f" -Name "AWS-UpdateSSMAgent"
```

출력:

```
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 :
```

예제 2: 이 예제에서는 대상을 사용하여 구성 문서를 인스턴스와 연결합니다.

```
$target = @{Key="instanceids";Values=@("i-0cb2b964d3e14fd9f")}
New-SSMAssociation -Name "AWS-UpdateSSMAgent" -Target $target
```

출력:

```
Name                  : AWS-UpdateSSMAgent
InstanceId            :
Date                  : 3/1/2017 6:22:21 PM
Status.Name           :
Status.Date           :
Status.Message        :
Status.AdditionalInfo :
```

예제 3: 이 예제는 대상과 파라미터를 사용하여 구성 문서를 인스턴스와 연결합니다.

```
$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
```

출력:

```
Name                  : Configure-CloudWatch
InstanceId            :
Date                  : 5/17/2018 3:17:44 PM
Status.Name           :
Status.Date           :
Status.Message        :
Status.AdditionalInfo :
```

예제 4: 이 예제에서는 **AWS-GatherSoftwareInventory**를 사용하여 리전에 있는 모든 인스턴스와의 연결을 생성합니다. 또한 파라미터에서 수집할 사용자 지정 파일 및 레지스트리 위치도 제공합니다.

```
$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)"
```

출력:

```
Name                  : AWS-GatherSoftwareInventory
InstanceId            :
```

```
Date                  : 6/9/2019 8:57:56 AM
Status.Name           :
Status.Date           :
Status.Message        :
Status.AdditionalInfo :
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 CreateAssociation을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **CreateAssociationBatch** 사용

다음 코드 예시는 CreateAssociationBatch의 사용 방법을 보여줍니다.

CLI

AWS CLI

다중 연결을 생성하는 방법

이 예제에서는 구성 문서를 여러 인스턴스와 연결합니다. 출력은 해당하는 경우 성공한 작업과 실패한 작업의 목록을 반환합니다.

명령:

```
aws ssm create-association-batch --entries "Name=AWS-
UpdateSSMAgent,InstanceId=i-1234567890abcdef0" "Name=AWS-
UpdateSSMAgent,InstanceId=i-9876543210abcdef0"
```

출력:

```
{
  "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": []
}
```

- API 세부 정보는 AWS CLI 명령 참조의 CreateAssociationBatch를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 구성 문서를 여러 인스턴스와 연결합니다. 출력은 해당하는 경우 성공한 작업과 실패한 작업의 목록을 반환합니다.

```
$option1 = @{InstanceId="i-0cb2b964d3e14fd9f";Name=@("AWS-UpdateSSMAgent")}
$option2 = @{InstanceId="i-0000293ffd8c57862";Name=@("AWS-UpdateSSMAgent")}
New-SSMAssociationFromBatch -Entry $option1,$option2
```

출력:

```
Failed  Successful
------  ----------
{}      {Amazon.SimpleSystemsManagement.Model.FailedCreateAssociation,
 Amazon.SimpleSystemsManagement.Model.FailedCreateAsso...
```

예제 2: 이 예제에서는 성공한 작업의 전체 세부 정보를 보여줍니다.

```
$option1 = @{InstanceId="i-0cb2b964d3e14fd9f";Name=@("AWS-UpdateSSMAgent")}
$option2 = @{InstanceId="i-0000293ffd8c57862";Name=@("AWS-UpdateSSMAgent")}
(New-SSMAssociationFromBatch -Entry $option1,$option2).Successful
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 CreateAssociationBatch를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **CreateDocument** 사용

다음 코드 예시는 CreateDocument의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- Systems Manager 시작하기

CLI

AWS CLI

문서를 생성하는 방법

다음 create-document 예제에서는 Systems Manager 문서를 생성합니다.

```
aws ssm create-document \
    --content file://exampleDocument.yml \
    --name "Example" \
    --document-type "Automation" \
    --document-format YAML
```

출력:

```
{
    "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": []
    }
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager 문서 생성을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 CreateDocument를 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS코드 예시 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    // Create an AWS SSM document to use in this scenario.
    public static void createSSMDoc(SsmClient ssmClient, String docName) {
```

```
        // Create JSON for the content
        String jsonData = """
            {
                "schemaVersion": "2.2",
                "description": "Run a simple shell command",
                "mainSteps": [
                    {
                        "action": "aws:runShellScript",
                        "name": "runEchoCommand",
                        "inputs": {
                          "runCommand": [
                            "echo 'Hello, world!'"
                          ]
                        }
                      }
                    ]
                }
            """;

        try {
            CreateDocumentRequest request = CreateDocumentRequest.builder()
                .content(jsonData)
                .name(docName)
                .documentType(DocumentType.COMMAND)
                .build();

            // Create the document.
            CreateDocumentResponse response = ssmClient.createDocument(request);
            System.out.println("The status of the document is " +
response.documentDescription().status());

        } catch (DocumentAlreadyExistsException e) {
            System.err.println("The document already exists. Moving on." );
        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 CreateDocument를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제는 계정에서 문서를 생성합니다. 문서는 JSON 형식이어야 합니다. 구성 문서 작성에 대한 자세한 내용은 SSM API 참조의 구성 문서를 참조하세요.

```
New-SSMDocument -Content (Get-Content -Raw "c:\temp\RunShellScript.json") -Name
 "RunShellScript" -DocumentType "Command"
```

출력:

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 CreateDocument를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **CreateMaintenanceWindow** 사용

다음 코드 예시는 CreateMaintenanceWindow의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- Systems Manager 시작하기

CLI

AWS CLI

예제 1: 유지 관리 기간을 생성하는 방법

다음 create-maintenance-window 예제에서는 필요한 경우 5분마다 최대 2시간 동안 유지 관리 기간 실행 종료 1시간 이내에 새 작업 시작을 방지하는 새 유지 관리 기간을 생성하고, 연결되지 않은 대상(유지 관리 기간에 등록되지 않은 인스턴스)을 허용하며, 생성자가 자습서에서 사용하려는 사용자 지정 태그 사용을 통해 이를 나타냅니다.

```
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"
```

출력:

```
{
    "WindowId": "mw-0c50858d01EXAMPLE"
}
```

예제 2: 한 번만 실행되는 유지 관리 기간을 생성하는 방법

다음 create-maintenance-window 예제에서는 지정된 날짜 및 시간에 한 번만 실행되는 새 유지 관리 기간을 생성합니다.

```
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"
```

출력:

```
{
    "WindowId": "mw-01234567890abcdef"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 유지 관리 기간을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 CreateMaintenanceWindow를 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS코드 예시 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
   public static String createMaintenanceWindow(SsmClient ssmClient, String
winName) {
       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();

       try {
           CreateMaintenanceWindowResponse response =
ssmClient.createMaintenanceWindow(request);
           String maintenanceWindowId = response.windowId();
           System.out.println("The maintenance window id is " +
maintenanceWindowId);
           return maintenanceWindowId;

       } catch (DocumentAlreadyExistsException e) {
           System.err.println("The maintenance window already exists. Moving
on.");
```

```
        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }

        MaintenanceWindowFilter filter = MaintenanceWindowFilter.builder()
            .key("name")
            .values(winName)
            .build();

        DescribeMaintenanceWindowsRequest winRequest =
DescribeMaintenanceWindowsRequest.builder()
            .filters(filter)
            .build();

        String windowId = "";
        DescribeMaintenanceWindowsResponse response =
ssmClient.describeMaintenanceWindows(winRequest);
        List<MaintenanceWindowIdentity> windows = response.windowIdentities();
        if (!windows.isEmpty()) {
            windowId = windows.get(0).windowId();
            System.out.println("Window ID: " + windowId);
        } else {
            System.out.println("Window not found.");
        }
       return windowId;
   }
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 CreateMaintenanceWindow를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 지정된 이름의 새 유지 관리 기간을 생성합니다. 이 유지 관리 기간은 매주 화요일 오후 4시에 4시간 동안 실행되며, 마감 시간은 1시간이고, 연결되지 않은 대상을 허용합니다.

```
New-SSMMaintenanceWindow -Name "MyMaintenanceWindow" -Duration 4 -Cutoff 1 -
AllowUnassociatedTarget $true -Schedule "cron(0 16 ? * TUE *)"
```

출력:

```
mw-03eb53e1ea7383998
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 CreateMaintenanceWindow를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 `CreateOpsItem` 사용

다음 코드 예시는 `CreateOpsItem`의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- Systems Manager 시작하기

CLI

AWS CLI

OpsItems를 생성하는 방법

다음 `create-ops-item` 예시에서는 OperationalData의 /aws/resources 키를 사용하여 Amazon DynamoDB 관련 리소스가 있는 OpsItem을 생성합니다.

```
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"
```

출력:

```
{
    "OpsItemId": "oi-1a2b3c4d5e6f"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 Creating OpsItems를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 CreateOpsItem을 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
   // Create an SSM OpsItem
   public static String createSSMOpsItem(SsmClient ssmClient, String title,
String source, String category, String severity) {
       try {
           CreateOpsItemRequest opsItemRequest = CreateOpsItemRequest.builder()
               .description("Created by the Systems Manager Java API")
               .title(title)
               .source(source)
               .category(category)
               .severity(severity)
               .build();

           CreateOpsItemResponse itemResponse =
ssmClient.createOpsItem(opsItemRequest);
           return itemResponse.opsItemId();

       } catch (SsmException e) {
           System.err.println(e.getMessage());
           System.exit(1);
       }
      return "";
   }
```

- API에 대한 세부 정보는 AWS SDK for Java 2.x API 참조의 CreateOpsItem을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **CreatePatchBaseline** 사용

다음 코드 예시는 CreatePatchBaseline의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 자동 승인을 사용하여 패치 기준을 생성하는 방법

다음 create-patch-baseline 예제에서는 Microsoft에서 릴리스하고 7일 후에 프로덕션 환경에 대한 패치를 승인하는 Windows Server용 패치 기준을 생성합니다.

```
aws ssm create-patch-baseline \
    --name "Windows-Production-Baseline-AutoApproval" \
    --operating-system "WINDOWS" \
    --approval-rules
 "PatchRules=[{PatchFilterGroup={PatchFilters=[{Key=MSRC_SEVERITY,Values=[Critical,Import
{Key=CLASSIFICATION,Values=[SecurityUpdates,Updates,UpdateRollups,CriticalUpdates]}]},App
 \
    --description "Baseline containing all updates approved for Windows Server
 production systems"
```

출력:

```
{
    "BaselineId": "pb-045f10b4f3EXAMPLE"
}
```

예제 2: 승인 마감일이 포함된 패치 기준을 생성하는 방법

다음 create-patch-baseline 예제에서는 2020년 7월 7일을 포함하여 해당 날짜 이전에 릴리스된 프로덕션 환경에 대한 패치를 승인하는 Windows Server용 패치 기준을 생성합니다.

```
aws ssm create-patch-baseline \
```

```
    --name "Windows-Production-Baseline-AutoApproval" \
    --operating-system "WINDOWS" \
    --approval-rules
 "PatchRules=[{PatchFilterGroup={PatchFilters=[{Key=MSRC_SEVERITY,Values=[Critical,Import
{Key=CLASSIFICATION,Values=[SecurityUpdates,Updates,UpdateRollups,CriticalUpdates]}]},App
 \
    --description "Baseline containing all updates approved for Windows Server
 production systems"
```

출력:

```
{
    "BaselineId": "pb-045f10b4f3EXAMPLE"
}
```

예제 3: JSON 파일에 저장된 승인 규칙을 사용하여 패치 기준을 생성하는 방법

다음 create-patch-baseline 예제에서는 Amazon Linux 2017.09용 패치 기준을 생성합니다. 여기에서는 릴리스하고 7일 후에 프로덕션 환경에 대한 패치를 승인하고 패치 기준에 대한 승인 규칙을 지정하며 패치에 대한 사용자 지정 리포지토리를 지정합니다.

```
aws ssm create-patch-baseline \
    --cli-input-json file://my-amazon-linux-approval-rules-and-repo.json
```

my-amazon-linux-approval-rules-and-repo.json의 콘텐츠:

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

예제 4: 승인된 패치와 거부된 패치를 지정하는 패치 기준을 생성하는 방법

다음 create-patch-baseline 예제에서는 기본 승인 규칙의 예외로 승인 및 거부할 패치를 명시적으로 지정합니다.

```
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"
```

자세한 내용은 AWS Systems Manager 사용 설명서의 사용자 지정 패치 기준 생성을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 CreatePatchBaseline을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 Microsoft에서 릴리스하고 7일 후에 프로덕션 환경에서 Windows Server 2019를 실행하는 관리형 인스턴스에 대한 패치를 승인하는 패치치 기준을 생성합니다.

```
$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
```

출력:

```
pb-0z4z6221c4296b23z
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 CreatePatchBaseline을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DeleteActivation** 사용

다음 코드 예시는 DeleteActivation의 사용 방법을 보여줍니다.

CLI

AWS CLI

관리형 인스턴스 활성화를 삭제하는 방법

다음 delete-activation 예제에서는 관리형 인스턴스 활성화를 삭제합니다.

```
aws ssm delete-activation \
```

```
    --activation-id "aa673477-d926-42c1-8757-1358cEXAMPLE"
```

이 명령은 출력을 생성하지 않습니다.

자세한 내용은 AWS Systems Manager 사용 설명서의 하이브리드 환경에 대한 AWS Systems Manager 설정을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DeleteActivation을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 활성화를 삭제합니다. 명령이 성공해도 출력은 없습니다.

```
Remove-SSMActivation -ActivationId "08e51e79-1e36-446c-8e63-9458569c1363"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DeleteActivation을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DeleteAssociation** 사용

다음 코드 예시는 DeleteAssociation의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 연결 ID를 사용하여 연결을 삭제하는 방법

다음 delete-association 예제에서는 지정된 연결 ID의 연결을 삭제합니다. 명령이 성공해도 출력은 없습니다.

```
aws ssm delete-association \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

이 명령은 출력을 생성하지 않습니다.

자세한 내용은 AWS Systems Manager 사용 설명서의 새 연결 버전 편집 및 생성을 참조하세요.

예제 2: 연결을 삭제하는 방법

다음 delete-association 예제에서는 인스턴스와 문서 간 연결을 삭제합니다. 명령이 성공해도 출력은 없습니다.

```
aws ssm delete-association \
    --instance-id "i-1234567890abcdef0" \
    --name "AWS-UpdateSSMAgent"
```

이 명령은 출력을 생성하지 않습니다.

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager에서 연결 작업을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DeleteAssociation을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 인스턴스와 문서 간의 연결을 삭제합니다. 명령이 성공해도 출력은 없습니다.

```
Remove-SSMAssociation -InstanceId "i-0cb2b964d3e14fd9f" -Name "AWS-
UpdateSSMAgent"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DeleteAssociation을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DeleteDocument** 사용

다음 코드 예시는 DeleteDocument의 사용 방법을 보여줍니다.

CLI

AWS CLI

문서를 삭제하는 방법

다음 `delete-document` 예제에서는 Systems Manager 문서를 삭제합니다.

```
aws ssm delete-document \
    --name "Example"
```

이 명령은 출력을 생성하지 않습니다.

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager 문서 생성을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DeleteDocument를 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS코드 예시 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
   // Deletes an AWS Systems Manager document.
   public static void deleteDoc(SsmClient ssmClient, String documentName) {
       try {
           DeleteDocumentRequest documentRequest =
DeleteDocumentRequest.builder()
               .name(documentName)
               .build();

           ssmClient.deleteDocument(documentRequest);
           System.out.println("The Systems Manager document was successfully
deleted.");

       } catch (SsmException e) {
```

```
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 DeleteDocument를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 문서를 삭제합니다. 명령이 성공해도 출력은 없습니다.

```
Remove-SSMDocument -Name "RunShellScript"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DeleteDocument를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DeleteMaintenanceWindow** 사용

다음 코드 예시는 DeleteMaintenanceWindow의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- Systems Manager 시작하기

CLI

AWS CLI

유지 관리 기간을 삭제하는 방법

이 delete-maintenance-window 예제에서는 지정된 유지 관리 기간을 제거합니다.

```
aws ssm delete-maintenance-window \
```

```
    --window-id "mw-1a2b3c4d5e6f7g8h9"
```

출력:

```
{
    "WindowId":"mw-1a2b3c4d5e6f7g8h9"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 유지 관리 기간 삭제(AWS CLI)를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DeleteMaintenanceWindow를 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS코드 예시 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
   public static void deleteMaintenanceWindow(SsmClient ssmClient, String winId)
{
       try {
           DeleteMaintenanceWindowRequest windowRequest =
DeleteMaintenanceWindowRequest.builder()
               .windowId(winId)
               .build();

           ssmClient.deleteMaintenanceWindow(windowRequest);
           System.out.println("The maintenance window was successfully
deleted.");

       } catch (SsmException e) {
           System.err.println(e.getMessage());
           System.exit(1);
       }
   }
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 DeleteMaintenanceWindow를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 유지 관리 기간을 제거합니다.

```
Remove-SSMMaintenanceWindow -WindowId "mw-06d59c1a07c022145"
```

출력:

```
mw-06d59c1a07c022145
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DeleteMaintenanceWindow를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DeleteParameter** 사용

다음 코드 예시는 DeleteParameter의 사용 방법을 보여줍니다.

CLI

AWS CLI

파라미터를 삭제하는 방법

다음 delete-parameter 예제에서는 지정된 단일 파라미터를 삭제합니다.

```
aws ssm delete-parameter \
    --name "MyParameter"
```

이 명령은 출력을 생성하지 않습니다.

자세한 내용은 AWS Systems Manager 사용 설명서의 Parameter Store 작업을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DeleteParameter를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 파라미터를 삭제합니다. 명령이 성공해도 출력은 없습니다.

```
Remove-SSMParameter -Name "helloWorld"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DeleteParameter를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DeletePatchBaseline** 사용

다음 코드 예시는 DeletePatchBaseline의 사용 방법을 보여줍니다.

CLI

AWS CLI

패치 기준을 삭제하는 방법

다음 delete-patch-baseline 예제에서는 지정된 패치 기준을 삭제합니다.

```
aws ssm delete-patch-baseline \
    --baseline-id "pb-045f10b4f382baeda"
```

출력:

```
{
    "BaselineId": "pb-045f10b4f382baeda"
}
```

자세한 내용은 AWSSystems Manager 사용 설명서의 패치 기준 업데이트 또는 삭제(콘솔)를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DeletePatchBaseline을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 패치 기준을 삭제합니다.

```
Remove-SSMPatchBaseline -BaselineId "pb-045f10b4f382baeda"
```

출력:

```
pb-045f10b4f382baeda
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DeletePatchBaseline을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DeregisterManagedInstance** 사용

다음 코드 예시는 DeregisterManagedInstance의 사용 방법을 보여줍니다.

CLI

AWS CLI

관리형 인스턴스를 등록 취소하는 방법

다음 deregister-managed-instance 예제에서는 지정된 관리형 인스턴스를 등록 취소합니다.

```
aws ssm deregister-managed-instance
    --instance-id "mi-08ab247cdfEXAMPLE"
```

이 명령은 출력을 생성하지 않습니다.

자세한 내용은 AWS Systems Manager 사용 설명서의 하이브리드 환경에서 관리형 인스턴스 등록 취소를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DeregisterManagedInstance를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 관리형 인스턴스를 등록 취소합니다. 명령이 성공해도 출력은 없습니다.

```
Unregister-SSMManagedInstance -InstanceId "mi-08ab247cdf1046573"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DeregisterManagedInstance를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 `DeregisterPatchBaselineForPatchGroup` 사용

다음 코드 예시는 `DeregisterPatchBaselineForPatchGroup`의 사용 방법을 보여줍니다.

CLI

AWS CLI

패치 기준에서 패치 그룹을 등록 취소하는 방법

다음 `deregister-patch-baseline-for-patch-group` 예제에서는 지정된 패치 기준에서 지정된 패치 그룹을 등록 취소합니다.

```
aws ssm deregister-patch-baseline-for-patch-group \
    --patch-group "Production" \
    --baseline-id "pb-0ca44a362fEXAMPLE"
```

출력:

```
{
  "PatchGroup":"Production",
  "BaselineId":"pb-0ca44a362fEXAMPLE"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 패치 기준에 패치 그룹 추가를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DeregisterPatchBaselineForPatchGroup을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 패치 기준에서 패치 그룹을 등록 취소합니다.

```
Unregister-SSMPatchBaselineForPatchGroup -BaselineId "pb-045f10b4f382baeda" -
PatchGroup "Production"
```

출력:

```
BaselineId           PatchGroup
----------           ----------
pb-045f10b4f382baeda Production
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DeregisterPatchBaselineForPatchGroup을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DeregisterTargetFromMaintenanceWindow** 사용

다음 코드 예시는 DeregisterTargetFromMaintenanceWindow의 사용 방법을 보여줍니다.

CLI

AWS CLI

유지 관리 기간에서 대상을 제거하는 방법

다음 deregister-target-from-maintenance-window 예제에서는 지정된 유지 관리 기간에서 지정된 대상을 제거합니다.

```
aws ssm deregister-target-from-maintenance-window \
    --window-id "mw-ab12cd34ef56gh78" \
    --window-target-id "1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
```

출력:

```
{
    "WindowId":"mw-ab12cd34ef56gh78",
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 유지 관리 기간 업데이트(AWS CLI)를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DeregisterTargetFromMaintenanceWindow를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 유지 관리 기간에서 대상을 제거합니다.

```
Unregister-SSMTargetFromMaintenanceWindow -WindowTargetId
 "6ab5c208-9fc4-4697-84b7-b02a6cc25f7d" -WindowId "mw-06cf17cbefcb4bf4f"
```

출력:

```
WindowId             WindowTargetId
--------             --------------
mw-06cf17cbefcb4bf4f 6ab5c208-9fc4-4697-84b7-b02a6cc25f7d
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DeregisterTargetFromMaintenanceWindow를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DeregisterTaskFromMaintenanceWindow** 사용

다음 코드 예시는 DeregisterTaskFromMaintenanceWindow의 사용 방법을 보여줍니다.

CLI

AWS CLI

유지 관리 기간에서 작업을 제거하는 방법

다음 deregister-task-from-maintenance-window 예제에서는 지정된 유지 관리 기간에서 지정된 작업을 제거합니다.

```
aws ssm deregister-task-from-maintenance-window \
    --window-id "mw-ab12cd34ef56gh78" \
    --window-task-id "1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d5e6c"
```

출력:

```
{
    "WindowTaskId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d5e6c",
    "WindowId":"mw-ab12cd34ef56gh78"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager 유지 관리 기간 자습서(AWS CLI)를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DeregisterTaskFromMaintenanceWindow를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 유지 관리 기간에서 작업을 제거합니다.

```
Unregister-SSMTaskFromMaintenanceWindow -WindowTaskId "f34a2c47-ddfd-4c85-
a88d-72366b69af1b" -WindowId "mw-03a342e62c96d31b0"
```

출력:

```
WindowId             WindowTaskId
--------             ------------
mw-03a342e62c96d31b0 f34a2c47-ddfd-4c85-a88d-72366b69af1b
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DeregisterTaskFromMaintenanceWindow를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeActivations** 사용

다음 코드 예시는 DescribeActivations의 사용 방법을 보여줍니다.

CLI

AWS CLI

활성화를 설명하는 방법

다음 describe-activations 예제에서는 AWS 계정의 활성화에 대한 세부 정보를 나열합니다.

```
aws ssm describe-activations
```

출력:

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

자세한 내용은 AWS Systems Manager 사용 설명서의 4단계: 하이브리드 환경을 위한 관리형 인스턴스 활성화 생성을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribeActivations를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 계정 활성화에 대한 세부 정보를 제공합니다.

```
Get-SSMActivation
```

출력:

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeActivations를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeAssociation** 사용

다음 코드 예시는 DescribeAssociation의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 연결 세부 정보를 가져오는 방법

다음 describe-association 예제에서는 지정된 연결 ID의 연결을 설명합니다.

```
aws ssm describe-association \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

출력:

```
{
    "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"
    }
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 새 연결 버전 편집 및 생성을 참조하세요.

예제 2: 특정 인스턴스 및 문서에 대한 연결 세부 정보를 가져오는 방법

다음 describe-association 예제에서는 인스턴스와 문서 간 연결을 설명합니다.

```
aws ssm describe-association \
    --instance-id "i-1234567890abcdef0" \
    --name "AWS-UpdateSSMAgent"
```

출력:

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

자세한 내용은 AWS Systems Manager 사용 설명서의 새 연결 버전 편집 및 생성을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribeAssociation을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 인스턴스와 문서 간 연결을 설명합니다.

```
Get-SSMAssociation -InstanceId "i-0000293ffd8c57862" -Name "AWS-UpdateSSMAgent"
```

출력:

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeAssociation을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeAssociationExecutionTargets** 사용

다음 코드 예시는 DescribeAssociationExecutionTargets의 사용 방법을 보여줍니다.

CLI

AWS CLI

연결 실행의 세부 정보를 가져오는 방법

다음 describe-association-execution-targets 예제에서는 지정된 연결 실행을 설명합니다.

```
aws ssm describe-association-execution-targets \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab" \
    --execution-id "7abb6378-a4a5-4f10-8312-0123456789ab"
```

출력:

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

자세한 내용은 AWS Systems Manager 사용 설명서의 연결 기록 보기를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribeAssociationExecutionTargets를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 연결 실행 대상의 일부인 리소스 ID와 해당 실행 상태를 표시합니다.

```
Get-SSMAssociationExecutionTarget -AssociationId 123a45a0-
c678-9012-3456-78901234db5e -ExecutionId 123a45a0-c678-9012-3456-78901234db5e |
 Select-Object ResourceId, Status
```

출력:

```
ResourceId           Status
----------           ------
i-0b1b2a3456f7a890b  Success
i-01c12a45d6fc7a89f  Success
i-0a1caf234f56d7dc8  Success
i-012a3fd45af6dbcfe  Failed
i-0ddc1df23c4a5fb67  Success
```

예제 2: 이 명령은 어제 이후 명령 문서가 연결된 명령 자동화의 특정 실행을 확인합니다. 연결 실행이 실패했는지 추가로 확인하고, 실패한 경우 인스턴스 ID와 함께 실행에 대한 명령 간접 호출 세부 정보를 표시합니다.

```
$AssociationExecution= Get-SSMAssociationExecutionTarget -
AssociationId 1c234567-890f-1aca-a234-5a678d901cb0 -ExecutionId
 12345ca12-3456-2345-2b45-23456789012 |
    Where-Object {$_.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
    }
}
```

출력:

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeAssociationExecutionTargets를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeAssociationExecutions** 사용

다음 코드 예시는 DescribeAssociationExecutions의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 연결에 대한 모든 실행 세부 정보를 가져오는 방법

다음 describe-association-executions 예제에서는 지정된 연결의 모든 실행을 설명합니다.

```
aws ssm describe-association-executions \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

출력:

```
{
    "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}"
        },
        ...
    ]
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 연결 기록 보기를 참조하세요.

예제 2: 특정 날짜 및 시간 이후 연결에 대한 모든 실행의 세부 정보를 보는 방법

다음 `describe-association-executions` 예제에서는 지정된 날짜 및 시간 이후 연결의 모든 실행을 설명합니다.

```
aws ssm describe-association-executions \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab" \
    --filters "Key=CreatedTime,Value=2019-02-18T16:00:00Z,Type=GREATER_THAN"
```

출력:

```
{
    "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}"
        },
        ...
    ]
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 연결 기록 보기를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribeAssociationExecutions를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 제공된 연결 ID에 대한 실행을 반환합니다.

```
Get-SSMAssociationExecution -AssociationId 123a45a0-c678-9012-3456-78901234db5e
```

출력:

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeAssociationExecutions를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeAutomationExecutions** 사용

다음 코드 예시는 DescribeAutomationExecutions의 사용 방법을 보여줍니다.

CLI

AWS CLI

자동화 실행을 설명하는 방법

다음 describe-automation-executions 예제에서는 자동화 실행에 대한 세부 정보를 표시합니다.

```
aws ssm describe-automation-executions \
    --filters Key=ExecutionId,Values=73c8eef8-f4ee-4a05-820c-e354fEXAMPLE
```

출력:

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

자세한 내용은 AWS Systems Manager 사용 설명서의 단순 자동화 워크플로 실행을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribeAutomationExecutions를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 계정과 연결된 모든 활성 및 종료된 자동화 실행을 설명합니다.

```
Get-SSMAutomationExecutionList
```

출력:

```
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]]}
```

예제 2: 이 예제에서는 자동화 실행 상태가 '성공'이 아닌 실행의 ExecutionID, 문서, 실행 시작 및 종료 타임스탬프를 표시합니다.

```
Get-SSMAutomationExecutionList | Where-Object AutomationExecutionStatus
 -ne "Success" | Select-Object AutomationExecutionId, DocumentName,
 AutomationExecutionStatus, ExecutionStartTime, ExecutionEndTime | Format-Table -
AutoSize
```

출력:

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeAutomationExecutions를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeAutomationStepExecutions** 사용

다음 코드 예시는 DescribeAutomationStepExecutions의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 자동화 실행의 모든 단계를 설명하는 방법

다음 describe-automation-step-executions 예제에서는 자동화 실행 단계에 대한 세부 정보를 표시합니다.

```
aws ssm describe-automation-step-executions \
    --automation-execution-id 73c8eef8-f4ee-4a05-820c-e354fEXAMPLE
```

출력:

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

예제 2: 자동화 실행의 특정 단계를 설명하는 방법

다음 `describe-automation-step-executions` 예제에서는 자동화 실행의 특정 단계에 대한 세부 정보를 표시합니다.

```
aws ssm describe-automation-step-executions \
    --automation-execution-id 73c8eef8-f4ee-4a05-820c-e354fEXAMPLE \
    --filters Key=StepExecutionId,Values=95e70479-cf20-4d80-8018-7e4e2EXAMPLE
```

자세한 내용은 AWS Systems Manager 사용 설명서의 자동화 워크플로 단계별 실행(명령줄)을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribeAutomationStepExecutions를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 자동화 워크플로에서 모든 활성 및 종료된 단계 실행에 대한 정보를 표시합니다.

```
Get-SSMAutomationStepExecution -AutomationExecutionId e1d2bad3-4567-8901-
ae23-456c7c8901be | Select-Object StepName, Action, StepStatus
```

출력:

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeAutomationStepExecutions를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeAvailablePatches** 사용

다음 코드 예시는 DescribeAvailablePatches의 사용 방법을 보여줍니다.

CLI

AWS CLI

사용 가능한 패치를 가져오는 방법

다음 describe-available-patches 예제에서는 MSRC 심각도가 위험인 Windows Server 2019에서 사용 가능한 모든 패치에 대한 세부 정보를 검색합니다.

```
aws ssm describe-available-patches \
    --filters "Key=PRODUCT,Values=WindowsServer2019"
 "Key=MSRC_SEVERITY,Values=Critical"
```

출력:

```
{
    "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"
        },
        ...
    ]
}
```


특정 패치의 세부 정보를 가져오는 방법

다음 describe-available-patches 예제에서는 지정된 패치에 대한 세부 정보를 검색합니다.

```
aws ssm describe-available-patches \
    --filters "Key=PATCH_ID,Values=KB4480979"
```

출력:

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

자세한 내용은 AWS Systems Manager 사용 설명서의 Patch Manager 작업 작동 방법을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribeAvailablePatches를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 MSRC 심각도가 위험인 Windows Server 2012에서 사용 가능한 모든 패치를 가져옵니다. 이 예제에서 사용하는 구문에는 PowerShell 버전 3 이상이 필요합니다.

```
$filter1 = @{Key="PRODUCT";Values=@("WindowsServer2012")}
$filter2 = @{Key="MSRC_SEVERITY";Values=@("Critical")}

Get-SSMAvailablePatch -Filter $filter1,$filter2
```

출력:

```
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
...
```

예제 2: PowerShell 버전 2에서 각 필터를 생성하려면 New-Object를 사용해야 합니다.

```
$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
```

예제 3: 이 예제에서는 지난 20일 동안 릴리스되고 WindowsServer2019와 일치하는 제품에 적용할 수 있는 모든 업데이트를 가져옵니다.

```
Get-SSMAvailablePatch | Where-Object ReleaseDate -ge (Get-Date).AddDays(-20)
 | Where-Object Product -eq "WindowsServer2019" | Select-Object ReleaseDate,
 Product, Title
```

출력:

```
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)
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeAvailablePatches를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeDocument** 사용

다음 코드 예시는 DescribeDocument의 사용 방법을 보여줍니다.

CLI

AWS CLI

문서 세부 정보를 표시하는 방법

다음 describe-document 예제에서는 사용자 AWS 계정의 Systems Manager 문서에 대한 세부 정보를 표시합니다.

```
aws ssm describe-document \
    --name "Example"
```

출력:

```
{
    "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": []
    }
```

```
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager 문서 생성을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribeDocument를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 문서에 대한 정보를 반환합니다.

```
Get-SSMDocumentDescription -Name "RunShellScript"
```

출력:

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeDocument를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeDocumentPermission** 사용

다음 코드 예시는 DescribeDocumentPermission의 사용 방법을 보여줍니다.

CLI

AWS CLI

문서 권한을 설명하는 방법

다음 describe-document-permission 예제에서는 공개적으로 공유되는 Systems Manager 문서에 대한 권한 세부 정보를 표시합니다.

```
aws ssm describe-document-permission \
    --name "Example" \
    --permission-type "Share"
```

출력:

```
{
    "AccountIds": [
        "all"
    ],
    "AccountSharingInfoList": [
        {
            "AccountId": "all",
            "SharedDocumentVersion": "$DEFAULT"
        }
    ]
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager 문서 공유를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribeDocumentPermission을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 문서의 모든 버전을 나열합니다.

```
Get-SSMDocumentVersionList -Name "RunShellScript"
```

출력:

```
CreatedDate          DocumentVersion IsDefaultVersion Name
-----------          --------------- ---------------- ----
2/24/2017 5:25:13 AM 1               True             RunShellScript
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeDocumentPermission을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeEffectiveInstanceAssociations** 사용

다음 코드 예시는 DescribeEffectiveInstanceAssociations의 사용 방법을 보여줍니다.

CLI

AWS CLI

인스턴스에 대한 유효한 연결의 세부 정보를 가져오는 방법

다음 describe-effective-instance-associations 예제에서는 인스턴스에 대한 유효한 연결의 세부 정보를 검색합니다.

명령:

```
aws ssm describe-effective-instance-associations --instance-id
 "i-1234567890abcdef0"
```

출력:

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

- API 세부 정보는 AWS CLI 명령 참조의 DescribeEffectiveInstanceAssociations를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 인스턴스에 대한 유효한 연결을 설명합니다.

```
Get-SSMEffectiveInstanceAssociationList -InstanceId "i-0000293ffd8c57862" -
MaxResult 5
```

출력:

```
AssociationId                        Content
-------------                        -------
d8617c07-2079-4c18-9847-1655fc2698b0 {...
```

예제 2: 이 예제에서는 인스턴스에 대한 유효한 연결의 콘텐츠를 설명합니다.

```
(Get-SSMEffectiveInstanceAssociationList -InstanceId "i-0000293ffd8c57862" -
MaxResult 5).Content
```

출력:

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

```
        }
    }
}
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeEffectiveInstanceAssociations를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeEffectivePatchesForPatchBaseline** 사용

다음 코드 예시는 DescribeEffectivePatchesForPatchBaseline의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 사용자 지정 패치 기준에서 정의한 모든 패치를 가져오는 방법

다음 describe-effective-patches-for-patch-baseline 예제에서는 현재 AWS 계정의 사용자 지정 패치 기준으로 정의된 패치를 반환합니다. 사용자 지정 기준의 경우 --baseline-id에는 ID만 필요합니다.

```
aws ssm describe-effective-patches-for-patch-baseline \
    --baseline-id "pb-08b654cf9b9681f04"
```

출력:

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

예제 2: AWS 관리형 패치 기준에서 정의한 모든 패치를 가져오는 방법

다음 `describe-effective-patches-for-patch-baseline` 예제에서는 AWS 관리형 패치 기준으로 정의된 패치를 반환합니다. AWS 관리형 기준의 경우 `--baseline-id`에는 전체 기준 ARN이 필요합니다.

```
aws ssm describe-effective-patches-for-patch-baseline \
    --baseline-id "arn:aws:ssm:us-east-2:733109147000:patchbaseline/
pb-020d361a05defe4ed"
```

샘플 출력은 예 1을 참조하세요.

자세한 내용은 AWS Systems Manager 사용 설명서의 보안 패치 선택 방법을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribeEffectivePatchesForPatchBaseline을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 최대 결과 목록이 1인 모든 패치 기준을 나열합니다.

```
Get-SSMEffectivePatchesForPatchBaseline -BaselineId "pb-0a2f1059b670ebd31" -
MaxResult 1
```

출력:

```
Patch                                       PatchStatus
-----                                       -----------
Amazon.SimpleSystemsManagement.Model.Patch
 Amazon.SimpleSystemsManagement.Model.PatchStatus
```

예제 2: 이 예제에서는 최대 결과 목록이 1인 모든 패치 기준의 패치 상태를 표시합니다.

```
(Get-SSMEffectivePatchesForPatchBaseline -BaselineId "pb-0a2f1059b670ebd31" -
MaxResult 1).PatchStatus
```

출력:

```
ApprovalDate          DeploymentStatus
------------          ----------------
12/21/2010 6:00:00 PM APPROVED
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeEffectivePatchesForPatchBaseline을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeInstanceAssociationsStatus** 사용

다음 코드 예시는 DescribeInstanceAssociationsStatus의 사용 방법을 보여줍니다.

CLI

AWS CLI

인스턴스 연결 상태를 설명하는 방법

이 예제에서는 인스턴스 연결의 세부 정보를 보여줍니다.

명령:

```
aws ssm describe-instance-associations-status --instance-id "i-1234567890abcdef0"
```

출력:

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

- API 세부 정보는 AWS CLI 명령 참조의 DescribeInstanceAssociationsStatus를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 인스턴스 연결의 세부 정보를 보여줍니다.

```
Get-SSMInstanceAssociationsStatus -InstanceId "i-0000293ffd8c57862"
```

출력:

```
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
```

예제 2: 이 예제에서는 지정된 인스턴스 ID의 인스턴스 연결 상태를 확인하고 더 나아가 해당 연결의 실행 상태를 표시합니다.

```
Get-SSMInstanceAssociationsStatus -InstanceId i-012e3cb4df567e8aa | ForEach-
Object {Get-SSMAssociationExecution -AssociationId .AssociationId}
```

출력:

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeInstanceAssociationsStatus를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeInstanceInformation** 사용

다음 코드 예시는 DescribeInstanceInformation의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 관리형 인스턴스 정보를 설명하는 방법

다음 describe-instance-information 예제에서는 각 관리형 인스턴스의 세부 정보를 검색합니다.

```
aws ssm describe-instance-information
```

예제 2: 특정 관리형 인스턴스에 대한 정보를 설명하는 방법

다음 describe-instance-information 예제에서는 관리형 인스턴스 i-028ea792daEXAMPLE의 세부 정보를 보여줍니다.

```
aws ssm describe-instance-information \
    --filters "Key=InstanceIds,Values=i-028ea792daEXAMPLE"
```

예제 3: 특정 태그 키를 사용하는 관리형 인스턴스에 대한 정보를 설명하는 방법

다음 describe-instance-information 예제에서는 태그 키 DEV가 있는 관리형 인스턴스의 세부 정보를 보여줍니다.

```
aws ssm describe-instance-information \
    --filters "Key=tag-key,Values=DEV"
```

출력:

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

자세한 내용은 AWS Systems Manager 사용 설명서의 관리형 인스턴스를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribeInstanceInformation을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 각 인스턴스의 세부 정보를 보여줍니다.

```
Get-SSMInstanceInformation
```

출력:

```
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
```

예제 2: 이 예제에서는 -Filter 파라미터를 사용하여 **AgentVersion**이 **2.2.800.0**인 **us-east-1** 리전의 AWS Systems Manager 인스턴스로만 결과를 필터링하는 방법을 보여줍니다. InstanceInformation API 참조 주제(https://docs.aws.amazon.com/systems-manager/latest/APIReference/API_InstanceInformation.html#systemsmanager-Type-InstanceInformation-ActivationId)에서 유효한 -Filter 키 값 목록을 찾을 수 있습니다.

```
$Filters = @{
    Key="AgentVersion"
    Values="2.2.800.0"
}
Get-SSMInstanceInformation -Region us-east-1 -Filter $Filters
```

출력:

```
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
```

예제 3: 이 예제에서는 -InstanceInformationFilterList 파라미터를 사용하여 **PlatformTypes**가 **Windows** 또는 **Linux**인 **us-east-1** 리전의 AWS Systems Manager 인스턴스로만 결과를 필터링하는 방법을 보여줍니다. InstanceInformationFilter API 참조 주제(https://docs.aws.amazon.com/systems-manager/latest/APIReference/API_InstanceInformationFilter.html)에서 유효한 -InstanceInformationFilterList 키 값 목록을 찾을 수 있습니다.

```
$Filters = @{
   Key="PlatformTypes"
   ValueSet=("Windows","Linux")
}
Get-SSMInstanceInformation -Region us-east-1 -InstanceInformationFilterList
 $Filters
```

출력:

```
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
```

예제 4: 이 예제에서는 ssm 관리형 인스턴스를 나열하고 InstanceId, PingStatus, LastPingDateTime 및 PlatformName을 csv 파일로 내보냅니다.

```
Get-SSMInstanceInformation | Select-Object InstanceId, PingStatus,
 LastPingDateTime, PlatformName | Export-Csv Instance-details.csv -
NoTypeInformation
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeInstanceInformation을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeInstancePatchStates** 사용

다음 코드 예시는 DescribeInstancePatchStates의 사용 방법을 보여줍니다.

CLI

AWS CLI

인스턴스의 패치 요약 상태를 가져오는 방법

이 describe-instance-patch-states 예제에서는 인스턴스의 패치 요약 상태를 가져옵니다.

```
aws ssm describe-instance-patch-states \
    --instance-ids "i-1234567890abcdef0"
```

출력:

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

자세한 내용은 AWS Systems Manager 사용 설명서의 패치 규정 준수 정보를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribeInstancePatchStates를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 인스턴스의 패치 요약 상태를 가져옵니다.

```
Get-SSMInstancePatchState -InstanceId "i-08ee91c0b17045407"
```

예제 2: 이 예제에서는 두 인스턴스의 패치 요약 상태를 가져옵니다.

```
Get-SSMInstancePatchState -InstanceId "i-08ee91c0b17045407","i-09a618aec652973a9"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeInstancePatchStates를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeInstancePatchStatesForPatchGroup** 사용

다음 코드 예시는 DescribeInstancePatchStatesForPatchGroup의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 패치 그룹의 인스턴스 상태를 가져오는 방법

다음 describe-instance-patch-states-for-patch-group 예제에서는 지정된 패치 그룹의 인스턴스당 패치 요약 상태에 대한 세부 정보를 검색합니다.

```
aws ssm describe-instance-patch-states-for-patch-group \
    --patch-group "Production"
```

출력:

```
{
    "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
        }
    ]
}
```

예제 2: 패치가 5개 넘게 누락된 패치 그룹의 인스턴스 상태를 가져오는 방법

다음 describe-instance-patch-states-for-patch-group 예제에서는 패치가 5개 넘게 누락된 인스턴스에서 지정된 패치 그룹의 패치 요약 상태에 대한 세부 정보를 검색합니다.

```
aws ssm describe-instance-patch-states-for-patch-group \
    --filters Key=MissingCount,Type=GreaterThan,Values=5 \
    --patch-group "Production"
```

출력:

```
{
    "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
        }
    ]
}
```

예제 3: 재부팅이 필요한 인스턴스가 10개 미만인 패치 그룹의 인스턴스 상태를 가져오는 방법

다음 describe-instance-patch-states-for-patch-group 예제에서는 리부팅해야 하는 패치가 10개 미만인 인스턴스에서 지정된 패치 그룹의 패치 요약 상태에 대한 세부 정보를 검색합니다.

```
aws ssm describe-instance-patch-states-for-patch-group \
    --filters Key=InstalledPendingRebootCount,Type=LessThan,Values=10 \
    --patch-group "Production"
```

출력:

```
{
    "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
        }
    ]
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 패치 규정 준수 상태 값 이해를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribeInstancePatchStatesForPatchGroup을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 패치 그룹의 인스턴스당 패치 요약 상태를 가져옵니다.

```
Get-SSMInstancePatchStatesForPatchGroup -PatchGroup "Production"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeInstancePatchStatesForPatchGroup을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeInstancePatches** 사용

다음 코드 예시는 DescribeInstancePatches의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 인스턴스의 패치 상태 세부 정보를 가져오는 방법

다음 describe-instance-patches 예제에서는 지정된 인스턴스의 패치에 대한 세부 정보를 검색합니다.

```
aws ssm describe-instance-patches \
```

```
    --instance-id "i-1234567890abcdef0"
```

출력:

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

예제 2: 인스턴스에서 누락 상태의 패치 목록을 가져오는 방법

다음 describe-instance-patches 예제에서는 지정된 인스턴스에서 누락 상태인 패치에 대한 정보를 검색합니다.

```
aws ssm describe-instance-patches \
    --instance-id "i-1234567890abcdef0" \
    --filters Key=State,Values=Missing
```

출력:

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

자세한 내용은 AWS Systems Manager 사용 설명서의 패치 규정 준수 상태 정보를 참조하세요.

예제 3: 인스턴스의 지정된 설치 시간 이후에 설치된 패치 목록을 가져오는 방법

다음 `describe-instance-patches` 예제에서는 `--filters` 및 `--query`의 사용을 조합하여 지정된 인스턴스에 대해 지정된 시간 이후에 설치된 패치에 대한 정보를 검색합니다.

```
aws ssm describe-instance-patches \
    --instance-id "i-1234567890abcdef0" \
    --filters Key=State,Values=Installed \
    --query "Patches[?InstalledTime >= `2023-01-01T16:00:00`]"
```

출력:

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

- API 세부 정보는 AWS CLI 명령 참조의 DescribeInstancePatches를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 인스턴스에 대한 패치 규정 준수 세부 정보를 가져옵니다.

```
Get-SSMInstancePatch -InstanceId "i-08ee91c0b17045407"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeInstancePatches를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeMaintenanceWindowExecutionTaskInvocations** 사용

다음 코드 예시는 DescribeMaintenanceWindowExecutionTaskInvocations의 사용 방법을 보여줍니다.

CLI

AWS CLI

유지 관리 기간 작업 실행을 위해 수행된 특정 작업 간접 호출을 가져오는 방법

다음 describe-maintenance-window-execution-task-invocations 예제에서는 지정된 유지 관리 기간 실행의 일부로 실행된 작업에 간접 호출을 나열합니다.

```
aws ssm describe-maintenance-window-execution-task-invocations \
    --window-execution-id "518d5565-5969-4cca-8f0e-da3b2a638355" \
    --task-id "ac0c6ae1-daa3-4a89-832e-d384503b6586"
```

출력:

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

자세한 내용은 AWS Systems Manager 사용 설명서의 작업 및 작업 실행에 대한 정보 보기 (AWS CLI)를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribeMaintenanceWindowExecutionTaskInvocations를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 유지 관리 기간 실행의 일부로 실행된 작업에 대한 간접 호출을 나열합니다.

```
Get-SSMMaintenanceWindowExecutionTaskInvocationList -TaskId "ac0c6ae1-
daa3-4a89-832e-d384503b6586" -WindowExecutionId "518d5565-5969-4cca-8f0e-
da3b2a638355"
```

출력:

```
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    :
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeMaintenanceWindowExecutionTaskInvocations를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeMaintenanceWindowExecutionTasks** 사용

다음 코드 예시는 DescribeMaintenanceWindowExecutionTasks의 사용 방법을 보여줍니다.

CLI

AWS CLI

유지 관리 기간 실행과 연결된 모든 작업을 나열하는 방법

다음 ssm describe-maintenance-window-execution-tasks 예제에서는 지정된 유지 관리 기간 실행과 연결된 작업을 나열합니다.

```
aws ssm describe-maintenance-window-execution-tasks \
    --window-execution-id "518d5565-5969-4cca-8f0e-da3b2EXAMPLE"
```

출력:

```
{
    "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"
```

```
        }
    ]
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 작업 및 작업 실행에 대한 정보 보기(AWS CLI)를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribeMaintenanceWindowExecutionTasks를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 유지 관리 기간 실행과 연결된 작업을 나열합니다.

```
Get-SSMMaintenanceWindowExecutionTaskList -WindowExecutionId
 "518d5565-5969-4cca-8f0e-da3b2a638355"
```

출력:

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeMaintenanceWindowExecutionTasks를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeMaintenanceWindowExecutions** 사용

다음 코드 예시는 DescribeMaintenanceWindowExecutions의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 유지 관리 기간의 모든 실행을 나열하는 방법

다음 describe-maintenance-window-executions 예제에서는 지정된 유지 관리 기간의 모든 실행을 나열합니다.

```
aws ssm describe-maintenance-window-executions \
    --window-id "mw-ab12cd34eEXAMPLE"
```

출력:

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

예제 2: 지정된 날짜 이전의 유지 관리 기간에 대한 모든 실행을 나열하는 방법

다음 describe-maintenance-window-executions 예제에서는 지정된 날짜 이전에 지정된 유지 관리 기간의 모든 실행을 나열합니다.

```
aws ssm describe-maintenance-window-executions \
    --window-id "mw-ab12cd34eEXAMPLE" \
    --filters "Key=ExecutedBefore,Values=2021-08-03T00:00:00Z"
```

출력:

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

예제 3: 지정된 날짜 이후 유지 관리 기간에 대한 모든 실행을 나열하는 방법

다음 describe-maintenance-window-executions 예제에서는 지정된 날짜 이후에 지정된 유지 관리 기간의 모든 실행을 나열합니다.

```
aws ssm describe-maintenance-window-executions \
    --window-id "mw-ab12cd34eEXAMPLE" \
    --filters "Key=ExecutedAfter,Values=2021-08-04T00:00:00Z"
```

출력:

```
{
    "WindowExecutions": [
        {
        "WindowId": "mw-ab12cd34eEXAMPLE",
        "WindowExecutionId": "6027b513-64fe-4cf0-be7d-1191aEXAMPLE",
        "Status": "IN_PROGRESS",
        "StartTime": "2021-08-04T11:00:00.000000-07:00"
        }
```

```
    ]
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 작업 및 작업 실행에 대한 정보 보기 (AWS CLI)를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribeMaintenanceWindowExecutions를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 유지 관리 기간의 모든 실행을 나열합니다.

```
Get-SSMMaintenanceWindowExecutionList -WindowId "mw-03eb9db42890fb82d"
```

출력:

```
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
```

예제 2: 이 예제에서는 지정된 날짜 이전에 유지 관리 기간의 모든 실행을 나열합니다.

```
$option1 = @{Key="ExecutedBefore";Values=@("2016-11-04T05:00:00Z")}
Get-SSMMaintenanceWindowExecutionList -WindowId "mw-03eb9db42890fb82d" -Filter
 $option1
```

예제 3: 이 예제에서는 지정된 날짜 이후에 유지 관리 기간의 모든 실행을 나열합니다.

```
$option1 = @{Key="ExecutedAfter";Values=@("2016-11-04T05:00:00Z")}
Get-SSMMaintenanceWindowExecutionList -WindowId "mw-03eb9db42890fb82d" -Filter
 $option1
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeMaintenanceWindowExecutions를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeMaintenanceWindowTargets** 사용

다음 코드 예시는 DescribeMaintenanceWindowTargets의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 유지 관리 기간의 모든 대상을 나열하는 방법

다음 describe-maintenance-window-targets 예제에서는 유지 관리 기간의 모든 대상을 나열합니다.

```
aws ssm describe-maintenance-window-targets \
    --window-id "mw-06cf17cbefEXAMPLE"
```

출력:

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

예제 2: 특정 소유자 정보 값과 일치하는 유지 관리 기간의 대상을 나열하는 방법

이 describe-maintenance-window-targets 예제에서는 특정 값이 있는 유지 관리 기간의 모든 대상을 나열합니다.

```
aws ssm describe-maintenance-window-targets \
    --window-id "mw-0ecb1226ddEXAMPLE" \
    --filters "Key=OwnerInformation,Values=CostCenter1"
```

출력:

```
{
    "Targets": [
        {
            "WindowId": "mw-0ecb1226ddEXAMPLE",
            "WindowTargetId": "da89dcc3-7f9c-481d-ba2b-edcb7d0057f9",
            "ResourceType": "INSTANCE",
            "Targets": [
                {
                    "Key": "tag:Environment",
                    "Values": [
                        "Prod"
                    ]
                }
            ],
            "OwnerInformation": "CostCenter1",
            "Name": "ProdTarget1"
        }
```

```
    ]
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 유지 관리 기간에 대한 정보 보기(AWS CLI)를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribeMaintenanceWindowTargets를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 유지 관리 기간의 모든 대상을 나열합니다.

```
Get-SSMMaintenanceWindowTarget -WindowId "mw-06cf17cbefcb4bf4f"
```

출력:

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeMaintenanceWindowTargets를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeMaintenanceWindowTasks** 사용

다음 코드 예시는 DescribeMaintenanceWindowTasks의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 유지 관리 기간의 모든 작업을 나열하는 방법

다음 describe-maintenance-window-tasks 예제에서는 지정된 유지 관리 기간의 모든 작업을 나열합니다.

```
aws ssm describe-maintenance-window-tasks \
    --window-id "mw-06cf17cbefEXAMPLE"
```

출력:

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

예제 2: AWS-RunPowerShellScript 명령 문서를 간접 호출하는 유지 관리 기간에 대한 모든 작업을 나열하는 방법

다음 describe-maintenance-window-tasks 예제에서는 AWS-RunPowerShellScript 명령 문서를 간접 호출하는 지정된 유지 관리 기간의 모든 작업을 나열합니다.

```
aws ssm describe-maintenance-window-tasks \
    --window-id "mw-ab12cd34eEXAMPLE" \
    --filters "Key=TaskArn,Values=AWS-RunPowerShellScript"
```

출력:

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

예제 3: 우선순위가 3인 유지 관리 기간의 모든 작업을 나열하는 방법

다음 describe-maintenance-window-tasks 예제에서는 3이 Priority인 지정된 유지 관리 기간의 모든 작업을 나열합니다.

```
aws ssm describe-maintenance-window-tasks \
    --window-id "mw-ab12cd34eEXAMPLE" \
    --filters "Key=Priority,Values=3"
```

출력:

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

예제 4: 우선순위가 1이고 Run Command를 사용하는 유지 관리 기간의 모든 작업을 나열하는 방법

이 describe-maintenance-window-tasks 예제에서는 1이 Priority이고 Run Command를 사용하는 지정된 유지 관리 기간의 모든 작업을 나열합니다.

```
aws ssm describe-maintenance-window-tasks \
    --window-id "mw-ab12cd34eEXAMPLE" \
    --filters "Key=Priority,Values=1" "Key=TaskType,Values=RUN_COMMAND"
```

출력:

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

자세한 내용은 AWS Systems Manager 사용 설명서의 유지 관리 기간에 대한 정보 보기(AWS CLI)를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribeMaintenanceWindowTasks를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 유지 관리 기간의 모든 작업을 나열합니다.

```
Get-SSMMaintenanceWindowTaskList -WindowId "mw-06cf17cbefcb4bf4f"
```

출력:

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeMaintenanceWindowTasks를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeMaintenanceWindows** 사용

다음 코드 예시는 DescribeMaintenanceWindows의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 모든 유지 관리 기간을 나열하는 방법

다음 describe-maintenance-windows 예제에서는 현재 리전 내 AWS 계정의 모든 유지 관리 기간을 나열합니다.

```
aws ssm describe-maintenance-windows
```

출력:

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

예제 2: 활성화된 모든 유지 관리 기간을 나열하는 방법

다음 `describe-maintenance-windows` 예제에서는 활성화된 모든 유지 관리 기간을 나열합니다.

```
aws ssm describe-maintenance-windows \
    --filters "Key=Enabled,Values=true"
```

예제 3: 특정 이름과 일치하는 유지 관리 기간을 나열하는 방법

이 `describe-maintenance-windows` 예제에서는 지정된 이름의 모든 유지 관리 기간을 나열합니다.

```
aws ssm describe-maintenance-windows \
    --filters "Key=Name,Values=MyMaintenanceWindow"
```

자세한 내용은 AWS Systems Manager 사용 설명서의 유지 관리 기간에 대한 정보 보기(AWS CLI)를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribeMaintenanceWindows를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 계정의 모든 유지 관리 기간을 나열합니다.

```
Get-SSMMaintenanceWindowList
```

출력:

```
Cutoff   : 1
Duration : 4
Enabled  : True
Name     : My-First-Maintenance-Window
WindowId : mw-06d59c1a07c022145
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeMaintenanceWindows를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeOpsItems** 사용

다음 코드 예시는 DescribeOpsItems의 사용 방법을 보여줍니다.

CLI

AWS CLI

OpsItems 세트를 나열하는 방법

다음 describe-ops-items 예시에서는 AWS 계정에서 열려 있는 모든 OpsItems의 목록을 표시합니다.

```
aws ssm describe-ops-items \
    --ops-item-filters "Key=Status,Values=Open,Operator=Equal"
```

출력:

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

자세한 내용은 AWS Systems Manager 사용 설명서의 OpsItems 작업을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribeOpsItems를 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    public static void describeOpsItems(SsmClient ssmClient, String key) {
        try {
            OpsItemFilter filter = OpsItemFilter.builder()
                .key(OpsItemFilterKey.OPS_ITEM_ID)
                .values(key)
                .operator(OpsItemFilterOperator.EQUAL)
                .build();
```

```
            DescribeOpsItemsRequest itemsRequest =
DescribeOpsItemsRequest.builder()
                .maxResults(10)
                .opsItemFilters(filter)
                .build();

            DescribeOpsItemsResponse itemsResponse =
ssmClient.describeOpsItems(itemsRequest);
            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());
            }

        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

- API에 대한 세부 정보는 AWS SDK for Java 2.x API 참조의 DescribeOpsItems를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribeParameters** 사용

다음 코드 예시는 DescribeParameters의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 모든 파라미터를 나열하는 방법

다음 describe-parameters 예시에서는 현재 AWS 계정 및 리전의 모든 파라미터를 나열합니다.

```
aws ssm describe-parameters
```

출력:

```
{
    "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": []
        }
    ]
}
```

예 2: 특정 메타데이터와 일치하는 모든 파라미터를 나열하는 방법

이 `describe-parameters` 예시에서는 필터와 일치하는 모든 파라미터를 나열합니다.

aws ssm describe-parameters --filters "Key=Type,Values=StringList"

출력:

```
{
    "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": []
        }
    ]
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager 파라미터 검색을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribeParameters를 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
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);
        }
    }
}
```

- API에 대한 세부 정보는 AWS SDK for Java 2.x API 참조의 DescribeParameters를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 모든 파라미터를 나열합니다.

```
Get-SSMParameterList
```

출력:

```
Description      :
KeyId            :
LastModifiedDate : 3/3/2017 6:58:23 PM
LastModifiedUser : arn:aws:iam::123456789012:user/admin
Name             : Welcome
Type             : String
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribeParameters를 참조하세요.

Rust

SDK for Rust

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS코드 예시 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
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(())
}
```

- API 세부 정보는 AWS SDK for Rust API 참조의 DescribeParameters를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribePatchBaselines** 사용

다음 코드 예시는 DescribePatchBaselines의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 모든 패치 기준을 나열하는 방법

다음 describe-patch-baselines 예제에서는 현재 리전의 계정에서 모든 패치 기준에 대한 세부 정보를 가져옵니다.

```
aws ssm describe-patch-baselines
```

출력:

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

예제 2: AWS에서 제공하는 모든 패치 기준을 나열하는 방법

다음 `describe-patch-baselines` 예제에서는 AWS에서 제공하는 모든 패치 기준을 나열합니다.

```
aws ssm describe-patch-baselines \
    --filters "Key=OWNER,Values=[AWS]"
```

예제 3: 소유한 모든 패치 기준을 나열하는 방법

다음 `describe-patch-baselines` 예제에서는 현재 리전의 계정에서 생성된 모든 사용자 지정 패치 기준을 나열합니다.

```
aws ssm describe-patch-baselines \
    --filters "Key=OWNER,Values=[Self]"
```

자세한 내용은 AWS Systems Manager 사용 설명서의 사전 정의된 패치 기준 및 사용자 지정 패치 기준 정보를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribePatchBaselines를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 모든 패치 기준을 나열합니다.

```
Get-SSMPatchBaseline
```

출력:

```
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...
```

예제 2: 이 예제에서는 AWS에서 제공하는 모든 패치 기준을 나열합니다. 이 예제에서 사용하는 구문에는 PowerShell 버전 3 이상이 필요합니다.

```
$filter1 = @{Key="OWNER";Values=@("AWS")}
```

출력:

```
Get-SSMPatchBaseline -Filter $filter1
```

예제 3: 이 예제에서는 사용자가 소유자인 모든 패치 기준을 나열합니다. 이 예제에서 사용하는 구문에는 PowerShell 버전 3 이상이 필요합니다.

```
$filter1 = @{Key="OWNER";Values=@("Self")}
```

출력:

```
Get-SSMPatchBaseline -Filter $filter1
```

예제 4: PowerShell 버전 2에서 각 태그를 생성하려면 New-Object를 사용해야 합니다.

```
$filter1 = New-Object
 Amazon.SimpleSystemsManagement.Model.PatchOrchestratorFilter
$filter1.Key = "OWNER"
$filter1.Values = "AWS"

Get-SSMPatchBaseline -Filter $filter1
```

출력:

```
BaselineDescription                     BaselineId
                            BaselineName            DefaultBaselin

                                                    e
-------------------                     ----------
                            ------------            --------------
Default Patch Baseline Provided by AWS. arn:aws:ssm:us-
west-2:123456789012:patchbaseline/pb-04fb4ae6142167966 AWS-DefaultPatchBaseline
 True
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribePatchBaselines를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribePatchGroupState** 사용

다음 코드 예시는 DescribePatchGroupState의 사용 방법을 보여줍니다.

CLI

AWS CLI

패치 그룹의 상태를 가져오는 방법

다음 describe-patch-group-state 예제에서는 패치 그룹에 대한 개요 수준의 패치 규정 준수 요약을 검색합니다.

```
aws ssm describe-patch-group-state \
    --patch-group "Production"
```

출력:

```
{
    "Instances": 21,
    "InstancesWithCriticalNonCompliantPatches": 1,
    "InstancesWithFailedPatches": 2,
```

```
    "InstancesWithInstalledOtherPatches": 3,
    "InstancesWithInstalledPatches": 21,
    "InstancesWithInstalledPendingRebootPatches": 2,
    "InstancesWithInstalledRejectedPatches": 1,
    "InstancesWithMissingPatches": 3,
    "InstancesWithNotApplicablePatches": 4,
    "InstancesWithOtherNonCompliantPatches": 1,
    "InstancesWithSecurityNonCompliantPatches": 1,
    "InstancesWithUnreportedNotApplicablePatches": 2
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 패치 그룹 정보(<https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-patchgroups.html>__)와 패치 규정 준수 상태 값 이해를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribePatchGroupState를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 패치 그룹에 대한 개요 수준의 패치 규정 준수 요약을 가져옵니다.

```
Get-SSMPatchGroupState -PatchGroup "Production"
```

출력:

```
Instances                          : 4
InstancesWithFailedPatches         : 1
InstancesWithInstalledOtherPatches : 4
InstancesWithInstalledPatches      : 3
InstancesWithMissingPatches        : 0
InstancesWithNotApplicablePatches  : 0
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribePatchGroupState를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **DescribePatchGroups** 사용

다음 코드 예시는 DescribePatchGroups의 사용 방법을 보여줍니다.

CLI

AWS CLI

패치 그룹 등록을 표시하는 방법

다음 describe-patch-groups 예제에서는 패치 그룹 등록을 나열합니다.

```
aws ssm describe-patch-groups
```

출력:

```
{
    "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
            }
        },
        ...
    ]
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 패치 그룹 생성(<https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-group-tagging.html>__)과 패치 기준에 패치 그룹 추가를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 DescribePatchGroups를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 패치 그룹 등록을 나열합니다.

```
Get-SSMPatchGroup
```

출력:

```
BaselineIdentity                                                PatchGroup
----------------                                                ----------
Amazon.SimpleSystemsManagement.Model.PatchBaselineIdentity Production
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 DescribePatchGroups를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **GetAutomationExecution** 사용

다음 코드 예시는 GetAutomationExecution의 사용 방법을 보여줍니다.

CLI

AWS CLI

자동화 실행에 대한 세부 정보를 표시하는 방법

다음 get-automation-execution 예제에서는 자동화 실행에 대한 세부 정보를 표시합니다.

```
aws ssm get-automation-execution \
    --automation-execution-id 73c8eef8-f4ee-4a05-820c-e354fEXAMPLE
```

출력:

```
{
    "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
        }
    }
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 연습: Linux AMI 패치(AWS CLI)를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 GetAutomationExecution을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 자동화 실행의 세부 정보를 표시합니다.

```
Get-SSMAutomationExecution -AutomationExecutionId "4105a4fc-
f944-11e6-9d32-8fb2db27a909"
```

출력:

```
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...}
```

예제 2: 이 예제에서는 지정된 자동화 실행 ID에 대한 단계 세부 정보를 나열합니다.

```
Get-SSMAutomationExecution -AutomationExecutionId e1d2bad3-4567-8901-
ae23-456c7c8901be | Select-Object -ExpandProperty StepExecutions | Select-Object
 StepName, Action, StepStatus, ValidNextSteps
```

출력:

```
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    {}
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 GetAutomationExecution을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **GetCommandInvocation** 사용

다음 코드 예시는 GetCommandInvocation의 사용 방법을 보여줍니다.

CLI

AWS CLI

명령 간접 호출의 세부 정보를 표시하는 방법

다음 get-command-invocation 예제에서는 지정된 인스턴스에서 지정된 명령의 모든 간접 호출을 나열합니다.

```
aws ssm get-command-invocation \
    --command-id "ef7fdfd8-9b57-4151-a15c-db9a12345678" \
    --instance-id "i-1234567890abcdef0"
```

출력:

```
{
    "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
    }
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 명령 상태 이해를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 GetCommandInvocation을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 인스턴스에서 실행된 명령의 세부 정보를 표시합니다.

```
Get-SSMCommandInvocationDetail -InstanceId "i-0cb2b964d3e14fd9f" -CommandId
 "b8eac879-0541-439d-94ec-47a80d554f44"
```

출력:

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 GetCommandInvocation을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **GetConnectionStatus** 사용

다음 코드 예시는 GetConnectionStatus의 사용 방법을 보여줍니다.

CLI

AWS CLI

관리형 인스턴스의 연결 상태를 표시하는 방법

이 get-connection-status 예제에서는 지정된 관리형 인스턴스의 연결 상태를 반환합니다.

```
aws ssm get-connection-status \
    --target i-1234567890abcdef0
```

출력:

```
{
    "Target": "i-1234567890abcdef0",
    "Status": "connected"
}
```

- API 세부 정보는 AWS CLI 명령 참조의 GetConnectionStatus를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 인스턴스의 세션 관리자 연결 상태를 검색하여 인스턴스가 연결되어 있고 세션 관리자 연결을 수신할 준비가 되었는지 확인합니다.

```
Get-SSMConnectionStatus -Target i-0a1caf234f12d3dc4
```

출력:

```
Status    Target
```

```
------    ------
Connected i-0a1caf234f12d3dc4
```

- API 세부 정보는 AWS Tools for PowerShell 명령 참조의 GetConnectionStatus를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **GetDefaultPatchBaseline** 사용

다음 코드 예시는 GetDefaultPatchBaseline의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 기본 Windows 패치 기준을 표시하는 방법

다음 get-default-patch-baseline 예제에서는 Windows Server의 기본 패치 기준에 대한 세부 정보를 검색합니다.

```
aws ssm get-default-patch-baseline
```

출력:

```
{
  "BaselineId": "pb-0713accee01612345",
  "OperatingSystem": "WINDOWS"
}
```

예제 2: Amazon Linux의 기본 패치 기준을 표시하는 방법

다음 get-default-patch-baseline 예제에서는 Amazon Linux의 기본 패치 기준에 대한 세부 정보를 검색합니다.

```
aws ssm get-default-patch-baseline \
    --operating-system AMAZON_LINUX
```

출력:

```
{
    "BaselineId": "pb-047c6eb9c8fc12345",
    "OperatingSystem": "AMAZON_LINUX"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 사전 정의된 패치 기준 및 사용자 지정 패치 기준 정보(<https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-baselines.html>__)와 기존 패치 기준을 기본값으로 설정을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 GetDefaultPatchBaseline을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 기본 패치 기준을 표시합니다.

```
Get-SSMDefaultPatchBaseline
```

출력:

```
arn:aws:ssm:us-west-2:123456789012:patchbaseline/pb-04fb4ae6142167966
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 GetDefaultPatchBaseline을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **GetDeployablePatchSnapshotForInstance** 사용

다음 코드 예시는 GetDeployablePatchSnapshotForInstance의 사용 방법을 보여줍니다.

CLI

AWS CLI

인스턴스에서 사용하는 패치 기준에 대한 현재 스냅샷을 검색하는 방법

다음 get-deployable-patch-snapshot-for-instance 예제에서는 인스턴스에서 사용하는 지정된 패치 기준의 현재 스냅샷에 대한 세부 정보를 검색합니다. 이 명령은 인스턴스 자격 증명을 사용하여 인스턴스에서 실행해야 합니다. 인스턴스 자격 증명을 사용하도록 하려면 aws configure를 실행하고 인스턴스의 리전만 지정합니다. Access Key 및 Secret Key 필드는 비워 둡니다.

팁: uuidgen을 사용하여 snapshot-id를 생성합니다.

```
aws ssm get-deployable-patch-snapshot-for-instance \
    --instance-id "i-1234567890abcdef0" \
    --snapshot-id "521c3536-930c-4aa9-950e-01234567abcd"
```

출력:

```
{
    "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"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 파라미터 이름: 스냅샷 ID를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 GetDeployablePatchSnapshotForInstance를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 인스턴스에서 사용하는 패치 기준의 현재 스냅샷을 표시합니다. 이 명령은 인스턴스 자격 증명을 사용하여 인스턴스에서 실행해야 합니다. 이 예제에서는 인스턴스 자격 증명을 사용하는지 확인하기 위해 자격 증명 파라미터에 **Amazon.Runtime.InstanceProfileAWSCredentials** 객체를 전달합니다.

```
$credentials = [Amazon.Runtime.InstanceProfileAWSCredentials]::new()
Get-SSMDeployablePatchSnapshotForInstance -SnapshotId "4681775b-098f-4435-
a956-0ef33373ac11" -InstanceId "i-0cb2b964d3e14fd9f" -Credentials $credentials
```

출력:

```
InstanceId          SnapshotDownloadUrl
----------          -------------------
i-0cb2b964d3e14fd9f https://patch-baseline-snapshot-us-west-2.s3-us-
west-2.amazonaws.com/853d0d3db0f0cafe...1692/4681775b-098f-4435...
```

예제 2: 이 예제에서는 전체 SnapshotDownloadUrl을 가져오는 방법을 보여줍니다. 이 명령은 인스턴스 자격 증명을 사용하여 인스턴스에서 실행해야 합니다. 인스턴스 자격 증명을 사용하는지 확인하기 위해 이 예제에서는 **`Amazon.Runtime.InstanceProfileAWSCredentials`** 객체를 사용하도록 PowerShell 세션을 구성합니다.

```
Set-AWSCredential -Credential
 ([Amazon.Runtime.InstanceProfileAWSCredentials]::new())
(Get-SSMDeployablePatchSnapshotForInstance -SnapshotId "4681775b-098f-4435-
a956-0ef33373ac11" -InstanceId "i-0cb2b964d3e14fd9f").SnapshotDownloadUrl
```

출력:

```
https://patch-baseline-snapshot-us-west-2.s3-us-
west-2.amazonaws.com/853d0d3db0f0cafe...
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 GetDeployablePatchSnapshotForInstance를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **`GetDocument`** 사용

다음 코드 예시는 `GetDocument`의 사용 방법을 보여줍니다.

CLI

AWS CLI

문서 콘텐츠를 가져오는 방법

다음 `get-document` 예제에서는 Systems Manager 문서의 콘텐츠를 표시합니다.

```
aws ssm get-document \
    --name "AWS-RunShellScript"
```

출력:

```
{
    "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"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 AWS Systems Manager 문서를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 GetDocument를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 문서의 콘텐츠를 반환합니다.

```
Get-SSMDocument -Name "RunShellScript"
```

출력:

```
Content
-------
{...
```

예제 2: 이 예제에서는 문서의 전체 콘텐츠를 표시합니다.

```
(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 }}"
         }
      }
   ]
}
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 GetDocument를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **GetInventory** 사용

다음 코드 예시는 GetInventory의 사용 방법을 보여줍니다.

CLI

AWS CLI

인벤토리 페이지를 보는 방법

이 예제에서는 인벤토리의 사용자 지정 메타데이터를 가져옵니다.

명령:

```
aws ssm get-inventory
```

출력:

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

- API 세부 정보는 AWS CLI 명령 참조의 GetInventory를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 인벤토리의 사용자 지정 메타데이터를 가져옵니다.

```
Get-SSMInventory
```

출력:

```
Data
     Id
----
     --
{[AWS:InstanceInformation,
 Amazon.SimpleSystemsManagement.Model.InventoryResultItem]} i-0cb2b964d3e14fd9f
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 GetInventory를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **GetInventorySchema** 사용

다음 코드 예시는 GetInventorySchema의 사용 방법을 보여줍니다.

CLI

AWS CLI

인벤토리 스키마를 보는 방법

이 예제에서는 계정의 인벤토리 유형 이름 목록을 반환합니다.

명령:

```
aws ssm get-inventory-schema
```

출력:

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

특정 인벤토리 유형의 인벤토리 스키마를 보는 방법

이 예제에서는 AWS:AWS 구성 요소 인벤토리 유형에 대한 인벤토리 스키마를 반환합니다.

명령:

```
aws ssm get-inventory-schema --type-name "AWS:AWSComponent"
```

- API 세부 정보는 AWS CLI 명령 참조의 GetInventorySchema를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 계정의 인벤토리 유형 이름 목록을 반환합니다.

```
Get-SSMInventorySchema
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 GetInventorySchema를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **GetMaintenanceWindow** 사용

다음 코드 예시는 GetMaintenanceWindow의 사용 방법을 보여줍니다.

CLI

AWS CLI

유지 관리 기간에 대한 정보를 가져오는 방법

다음 get-maintenance-window 예제에서는 지정된 유지 관리 기간에 대한 세부 정보를 검색합니다.

```
aws ssm get-maintenance-window \
    --window-id "mw-03eb9db428EXAMPLE"
```

출력:

```
{
    "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"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 유지 관리 기간에 대한 정보 보기(AWS CLI)를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 GetMaintenanceWindow를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 유지 관리 기간에 대한 세부 정보를 가져옵니다.

```
Get-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d"
```

출력:

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 GetMaintenanceWindow를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **GetMaintenanceWindowExecution** 사용

다음 코드 예시는 `GetMaintenanceWindowExecution`의 사용 방법을 보여줍니다.

CLI

AWS CLI

유지 관리 기간 작업 실행에 대한 정보를 가져오는 방법

다음 `get-maintenance-window-execution` 예제에서는 지정된 유지 관리 기간 실행의 일부로 실행된 작업에 대한 정보를 나열합니다.

```
aws ssm get-maintenance-window-execution \
    --window-execution-id "518d5565-5969-4cca-8f0e-da3b2EXAMPLE"
```

출력:

```
{
```

```
    "Status": "SUCCESS",
    "TaskIds": [
        "ac0c6ae1-daa3-4a89-832e-d3845EXAMPLE"
    ],
    "StartTime": 1487692834.595,
    "EndTime": 1487692835.051,
    "WindowExecutionId": "518d5565-5969-4cca-8f0e-da3b2EXAMPLE",
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 작업 및 작업 실행에 대한 정보 보기 (AWS CLI)를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 GetMaintenanceWindowExecution을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 유지 관리 기간 실행의 일부로 실행된 작업에 대한 정보를 나열합니다.

```
Get-SSMMaintenanceWindowExecution -WindowExecutionId "518d5565-5969-4cca-8f0e-
da3b2a638355"
```

출력:

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 GetMaintenanceWindowExecution을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **GetMaintenanceWindowExecutionTask** 사용

다음 코드 예시는 GetMaintenanceWindowExecutionTask의 사용 방법을 보여줍니다.

CLI

AWS CLI

유지 관리 기간 작업 실행에 대한 정보를 가져오는 방법

다음 get-maintenance-window-execution-task 예제에서는 지정된 유지 관리 기간 실행의 일부인 작업에 대한 정보를 나열합니다.

```
aws ssm get-maintenance-window-execution-task \
    --window-execution-id "518d5565-5969-4cca-8f0e-da3b2EXAMPLE" \
    --task-id "ac0c6ae1-daa3-4a89-832e-d3845EXAMPLE"
```

출력:

```
{
    "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"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 작업 및 작업 실행에 대한 정보 보기(AWS CLI)를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 GetMaintenanceWindowExecutionTask를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 유지 관리 기간 실행의 일부였던 작업에 대한 정보를 나열합니다.

```
Get-SSMMaintenanceWindowExecutionTask -TaskId "ac0c6ae1-daa3-4a89-832e-
d384503b6586" -WindowExecutionId "518d5565-5969-4cca-8f0e-da3b2a638355"
```

출력:

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 GetMaintenanceWindowExecutionTask를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **GetParameterHistory** 사용

다음 코드 예시는 GetParameterHistory의 사용 방법을 보여줍니다.

CLI

AWS CLI

파라미터 값 기록을 가져오는 방법

다음 get-parameter-history 예제에서는 해당 값을 포함하여 지정된 파라미터의 변경 기록을 나열합니다.

```
aws ssm get-parameter-history \
    --name "MyStringParameter"
```

출력:

```
{
    "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": []
        }
```

```
    ]
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 파라미터 버전 작업을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 GetParameterHistory를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 파라미터 값 기록을 나열합니다.

```
Get-SSMParameterHistory -Name "Welcome"
```

출력:

```
Description      :
KeyId            :
LastModifiedDate : 3/3/2017 6:55:25 PM
LastModifiedUser : arn:aws:iam::123456789012:user/admin
Name             : Welcome
Type             : String
Value            : helloWorld
```

- API 세부 정보는 AWS Tools for PowerShell 명령 참조의 GetParameterHistory를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **GetParameters** 사용

다음 코드 예시는 GetParameters의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 파라미터 값을 나열하는 방법

다음 get-parameters 예제에서는 지정된 세 개의 파라미터 값을 나열합니다.

```
aws ssm get-parameters \
    --names "MyStringParameter" "MyStringListParameter" "MyInvalidParameterName"
```

출력:

```
{
    "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"
    ]
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 Parameter Store 작업을 참조하세요.

예제 2: "--query" 옵션을 사용하여 여러 파라미터의 이름과 값을 나열하는 방법

다음 get-parameters 예제에서는 지정된 파라미터의 이름 및 값을 나열합니다.

```
aws ssm get-parameters \
```

```
    --names MyStringParameter MyStringListParameter \
    --query "Parameters[*].{Name:Name,Value:Value}"
```

출력:

```
[
    {
        "Name": "MyStringListParameter",
        "Value": "alpha,beta,gamma"
    },
    {
        "Name": "MyStringParameter",
        "Value": "Vidi"
    }
]
```

자세한 내용은 AWS Systems Manager 사용 설명서의 Parameter Store 작업을 참조하세요.

예제 3: 레이블을 사용하여 파라미터 값을 표시하는 방법

다음 `get-parameter` 예제에서는 지정된 레이블을 포함하는 지정된 단일 파라미터 값을 나열합니다.

```
aws ssm get-parameter \
    --name "MyParameter:label"
```

출력:

```
{
    "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": []
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 파라미터 레이블 작업을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 GetParameters를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 파라미터 값을 나열합니다.

```
Get-SSMParameterValue -Name "Welcome"
```

출력:

```
InvalidParameters Parameters
----------------- ----------
{}                {Welcome}
```

예제 2: 이 예제에서는 값의 세부 정보를 나열합니다.

```
(Get-SSMParameterValue -Name "Welcome").Parameters
```

출력:

```
Name    Type   Value
----    ----   -----
Welcome String Good day, Sunshine!
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 GetParameters를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **GetPatchBaseline** 사용

다음 코드 예시는 GetPatchBaseline의 사용 방법을 보여줍니다.

CLI

AWS CLI

패치 기준을 표시하는 방법

다음 get-patch-baseline 예제에서는 지정된 패치 기준에 대한 세부 정보를 검색합니다.

```
aws ssm get-patch-baseline \
    --baseline-id "pb-0123456789abcdef0"
```

출력:

```
{
    "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": []
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 패치 기준 정보를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 GetPatchBaseline을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 패치 기준의 세부 정보를 표시합니다.

```
Get-SSMPatchBaselineDetail -BaselineId "pb-03da896ca3b68b639"
```

출력:

```
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 : {}
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 GetPatchBaseline을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **GetPatchBaselineForPatchGroup** 사용

다음 코드 예시는 GetPatchBaselineForPatchGroup의 사용 방법을 보여줍니다.

CLI

AWS CLI

패치 그룹의 패치 기준을 표시하는 방법

다음 get-patch-baseline-for-patch-group 예제에서는 지정된 패치 그룹의 패치 기준에 대한 세부 정보를 검색합니다.

```
aws ssm get-patch-baseline-for-patch-group \
    --patch-group "DEV"
```

출력:

```
{
    "PatchGroup": "DEV",
    "BaselineId": "pb-0123456789abcdef0",
    "OperatingSystem": "WINDOWS"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 패치 그룹 생성(<https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-group-tagging.html>__)과 패치 기준에 패치 그룹 추가를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 GetPatchBaselineForPatchGroup을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 패치 그룹의 패치 기준을 표시합니다.

```
Get-SSMPatchBaselineForPatchGroup -PatchGroup "Production"
```

출력:

```
BaselineId           PatchGroup
----------           ----------
pb-045f10b4f382baeda Production
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 GetPatchBaselineForPatchGroup을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **ListAssociationVersions** 사용

다음 코드 예시는 ListAssociationVersions의 사용 방법을 보여줍니다.

CLI

AWS CLI

특정 연결 ID의 모든 연결 버전을 가져오는 방법

다음 list-association-versions 예제에서는 지정된 연결의 모든 버전을 나열합니다.

```
aws ssm list-association-versions \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab"
```

출력:

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

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager에서 연결 작업을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 ListAssociationVersions를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 제공된 연결의 모든 버전을 검색합니다.

```
Get-SSMAssociationVersionList -AssociationId 123a45a0-c678-9012-3456-78901234db5e
```

출력:

```
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}
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 ListAssociationVersions를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **ListAssociations** 사용

다음 코드 예시는 ListAssociations의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 특정 인스턴스의 연결을 나열하는 방법

다음 list-associations 예제에서는 AssociationName인 UpdateSSMAgent인 모든 연결을 나열합니다.

```
aws ssm list-associations /
    --association-filter-list "key=AssociationName,value=UpdateSSMAgent"
```

출력:

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

자세한 내용은 Systems Manager 사용 설명서의 Systems Manager에서 연결 작업을 참조하세요.

예제 2: 특정 문서의 연결을 나열하는 방법

다음 list-associations 예제에서는 지정된 문서의 모든 연결을 나열합니다.

```
aws ssm list-associations /
    --association-filter-list "key=Name,value=AWS-UpdateSSMAgent"
```

출력:

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

자세한 내용은 Systems Manager 사용 설명서의 Systems Manager에서 연결 작업을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 ListAssociations를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 인스턴스의 모든 연결을 나열합니다. 이 예제에서 사용하는 구문에는 PowerShell 버전 3 이상이 필요합니다.

```
$filter1 = @{Key="InstanceId";Value=@("i-0000293ffd8c57862")}
Get-SSMAssociationList -AssociationFilterList $filter1
```

출력:

```
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}
```

예제 2: 이 예제에서는 구성 문서의 모든 연결을 나열합니다. 이 예제에서 사용하는 구문에는 PowerShell 버전 3 이상이 필요합니다.

```
$filter2 = @{Key="Name";Value=@("AWS-UpdateSSMAgent")}
```

```
Get-SSMAssociationList -AssociationFilterList $filter2
```

출력:

```
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}
```

예제 3: PowerShell 버전 2에서 각 필터를 생성하려면 New-Object를 사용해야 합니다.

```
$filter1 = New-Object Amazon.SimpleSystemsManagement.Model.AssociationFilter
$filter1.Key = "InstanceId"
$filter1.Value = "i-0000293ffd8c57862"

Get-SSMAssociationList -AssociationFilterList $filter1
```

출력:

```
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}
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 ListAssociations를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **ListCommandInvocations** 사용

다음 코드 예시는 ListCommandInvocations의 사용 방법을 보여줍니다.

CLI

AWS CLI

특정 명령의 간접 호출을 나열하는 방법

다음 `list-command-invocations` 예제에서는 명령의 모든 간접 호출을 나열합니다.

```
aws ssm list-command-invocations \
    --command-id "ef7fdfd8-9b57-4151-a15c-db9a12345678" \
    --details
```

출력:

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

자세한 내용은 AWS Systems Manager 사용 설명서의 명령 상태 이해를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 ListCommandInvocations를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 명령의 모든 간접 호출을 나열합니다.

```
Get-SSMCommandInvocation -CommandId "b8eac879-0541-439d-94ec-47a80d554f44" -
Detail $true
```

출력:

```
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         :
```

예제 2: 이 예제에서는 명령 ID e1eb2e3c-ed4c-5123-45c1-234f5612345f의 간접 호출에 대한 CommandPlugins를 나열합니다.

```
Get-SSMCommandInvocation -CommandId e1eb2e3c-ed4c-5123-45c1-234f5612345f -Detail:
$true | Select-Object -ExpandProperty CommandPlugins
```

출력:

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 ListCommandInvocations를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **ListCommands** 사용

다음 코드 예시는 ListCommands의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 특정 명령의 상태를 가져오는 방법

다음 list-commands 예제에서는 지정된 명령의 상태를 검색하고 표시합니다.

```
aws ssm list-commands \
    --command-id "0831e1a8-a1ac-4257-a1fd-c831bEXAMPLE"
```

예제 2: 특정 날짜 이후에 요청된 명령의 상태를 가져오는 방법

다음 list-commands 예제에서는 지정된 날짜 이후에 요청된 명령의 세부 정보를 검색합니다.

```
aws ssm list-commands \
    --filter "key=InvokedAfter,value=2020-02-01T00:00:00Z"
```

예제 3: AWS 계정에서 요청한 모든 명령을 나열하는 방법

다음 list-commands 예제에서는 현재 AWS 계정 및 리전의 사용자가 요청한 모든 명령을 나열합니다.

```
aws ssm list-commands
```

출력:

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

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager Run Command를 사용하여 명령 실행을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 ListCommands를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 요청된 모든 명령을 나열합니다.

```
Get-SSMCommand
```

출력:

```
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            : {}
```

예제 2: 이 예제는 특정 명령의 상태를 가져옵니다.

```
Get-SSMCommand -CommandId "4b75a163-d39a-4d97-87c9-98ae52c6be35"
```

예제 3: 이 예제에서는 2019-04-01T00:00:00Z 이후에 간접 호출된 모든 SSM 명령을 검색합니다.

```
Get-SSMCommand -Filter @{Key="InvokedAfter";Value="2019-04-01T00:00:00Z"} |
 Select-Object CommandId, DocumentName, Status, RequestedDateTime | Sort-Object -
Property RequestedDateTime -Descending
```

출력:

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 ListCommands를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **ListComplianceItems** 사용

다음 코드 예시는 ListComplianceItems의 사용 방법을 보여줍니다.

CLI

AWS CLI

특정 인스턴스의 규정 준수 항목을 나열하는 방법

이 예제에서는 지정된 인스턴스의 모든 규정 준수 항목을 나열합니다.

명령:

```
aws ssm list-compliance-items --resource-ids "i-1234567890abcdef0" --resource-
types "ManagedInstance"
```

출력:

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

특정 인스턴스 및 연결 ID에 대한 규정 준수 항목을 나열하는 방법

이 예제에서는 지정된 인스턴스 및 연결 ID의 모든 규정 준수 항목을 나열합니다.

명령:

```
aws ssm list-compliance-items --resource-ids "i-1234567890abcdef0" --resource-
types "ManagedInstance" --filters
 "Key=ComplianceType,Values=Association,Type=EQUAL"
 "Key=Id,Values=e4c2ed6d-516f-41aa-aa2a-0123456789ab,Type=EQUAL"
```

특정 날짜 및 시간 이후 인스턴스의 규정 준수 항목을 나열하는 방법

이 예제에서는 지정된 날짜 및 시간 이후 인스턴스에 대한 모든 규정 준수 항목을 나열합니다.

명령:

```
aws ssm list-compliance-items --resource-ids "i-1234567890abcdef0" --resource-
types "ManagedInstance" --filters
 "Key=ExecutionTime,Values=2019-02-18T16:00:00Z,Type=GREATER_THAN"
```

- API 세부 정보는 AWS CLI 명령 참조의 ListComplianceItems를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 지정된 리소스 ID 및 유형에 대한 규정 준수 항목 목록을 나열하며, 이때 필터링 규정 준수 유형은 '연결'로 필터링됩니다.

```
Get-SSMComplianceItemList -ResourceId i-1a2caf345f67d0dc2 -ResourceType
 ManagedInstance -Filter @{Key="ComplianceType";Values="Association"}
```

출력:

```
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            :
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 ListComplianceItems를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **ListComplianceSummaries** 사용

다음 코드 예시는 `ListComplianceSummaries`의 사용 방법을 보여줍니다.

CLI

AWS CLI

모든 규정 준수 유형에 대한 규정 준수 요약을 나열하는 방법

이 예제에서는 계정의 모든 규정 준수 유형에 대한 규정 준수 요약을 나열합니다.

명령:

```
aws ssm list-compliance-summaries
```

출력:

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

특정 규정 준수 유형에 대한 규정 준수 요약을 나열하는 방법

이 예제에서는 패치 규정 준수 유형에 대한 규정 준수 요약을 나열합니다.

명령:

```
aws ssm list-compliance-summaries --filters
 "Key=ComplianceType,Values=Patch,Type=EQUAL"
```

- API 세부 정보는 AWS CLI 명령 참조의 ListComplianceSummaries를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 모든 규정 준수 유형에 대한 규정 준수 및 규정 미준수 리소스의 요약 개수를 반환합니다.

```
Get-SSMComplianceSummaryList
```

출력:

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 ListComplianceSummaries를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **ListDocumentVersions** 사용

다음 코드 예시는 ListDocumentVersions의 사용 방법을 보여줍니다.

CLI

AWS CLI

문서 버전을 나열하는 방법

다음 list-document-versions 예제에서는 Systems Manager 문서의 모든 버전을 나열합니다.

```
aws ssm list-document-versions \
    --name "Example"
```

출력:

```
{
```

```
    "DocumentVersions": [
        {
            "Name": "Example",
            "DocumentVersion": "1",
            "CreatedDate": 1583257938.266,
            "IsDefaultVersion": true,
            "DocumentFormat": "YAML",
            "Status": "Active"
        }
    ]
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 문서 버전 파라미터를 사용하는 명령 전송을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 ListDocumentVersions를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 문서에 대한 권한 목록을 반환합니다.

```
Get-SSMDocumentPermission -Name "RunShellScript" -PermissionType "Share"
```

출력:

```
all
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 ListDocumentVersions를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **ListDocuments** 사용

다음 코드 예시는 ListDocuments의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 문서를 나열하는 방법

다음 `list-documents` 예제에서는 사용자 지정 태그로 지정된 요청 계정에서 소유한 문서를 나열합니다.

```
aws ssm list-documents \
    --filters Key=Owner,Values=Self Key=tag:DocUse,Values=Testing
```

출력:

```
{
    "DocumentIdentifiers": [
        {
            "Name": "Example",
            "Owner": "29884EXAMPLE",
            "PlatformTypes": [
                "Windows",
                "Linux"
            ],
            "DocumentVersion": "1",
            "DocumentType": "Automation",
            "SchemaVersion": "0.3",
            "DocumentFormat": "YAML",
            "Tags": [
                {
                    "Key": "DocUse",
                    "Value": "Testing"
                }
            ]
        }
    ]
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 AWS Systems Manager 문서를 참조하세요.

예제 2: 공유 문서를 나열하는 방법

다음 `list-documents` 예제에서는 AWS에서 소유하지 않은 프라이빗 공유 문서를 포함한 공유 문서를 나열합니다.

```
aws ssm list-documents \
    --filters Key=Name,Values=sharedDocNamePrefix  Key=Owner,Values=Private
```

출력:

```
{
    "DocumentIdentifiers": [
        {
            "Name": "Example",
            "Owner": "12345EXAMPLE",
            "PlatformTypes": [
                "Windows",
                "Linux"
            ],
            "DocumentVersion": "1",
            "DocumentType": "Command",
            "SchemaVersion": "0.3",
            "DocumentFormat": "YAML",
            "Tags": []
        }
    ]
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 AWS Systems Manager 문서를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 ListDocuments를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 계정의 모든 구성 문서를 나열합니다.

```
Get-SSMDocumentList
```

출력:

```
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
...
```

예제 2: 이 예제에서는 이름이 'Platform'과 일치하는 모든 자동화 문서를 검색합니다.

```
Get-SSMDocumentList -DocumentFilterList @{Key="DocumentType";Value="Automation"}
 | Where-Object Name -Match "Platform"
```

출력:

```
DocumentFormat  : JSON
DocumentType    : Automation
DocumentVersion : 7
Name            : KT-Get-Platform
Owner           : 987654123456
PlatformTypes   : {Windows, Linux}
SchemaVersion   : 0.3
Tags            : {}
TargetType      :
VersionName     :
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 ListDocuments를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **ListInventoryEntries** 사용

다음 코드 예시는 ListInventoryEntries의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 인스턴스의 특정 인벤토리 유형 항목을 보는 방법

다음 list-inventory-entries 예제에서는 특정 인스턴스의 AWS:Application 인벤토리 유형에 대한 인벤토리 항목을 나열합니다.

```
aws ssm list-inventory-entries \
    --instance-id "i-1234567890abcdef0" \
    --type-name "AWS:Application"
```

출력:

```
{
  "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"
```

```
    }
  ]
}
```

예제 2: 인스턴스에 할당된 사용자 지정 인벤토리 항목을 보는 방법

다음 `list-inventory-entries` 예제에서는 인스턴스에 할당된 사용자 지정 인벤토리 항목을 나열합니다.

```
aws ssm list-inventory-entries \
    --instance-id "i-1234567890abcdef0" \
    --type-name "Custom:RackInfo"
```

출력:

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

- API 세부 정보는 AWS CLI 명령 참조의 ListInventoryEntries를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 인스턴스의 모든 사용자 지정 인벤토리 항목을 나열합니다.

```
Get-SSMInventoryEntriesList -InstanceId "i-0cb2b964d3e14fd9f" -TypeName
 "Custom:RackInfo"
```

출력:

```
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
```

예제 2: 이 예제에서는 세부 정보를 나열합니다.

```
(Get-SSMInventoryEntriesList -InstanceId "i-0cb2b964d3e14fd9f" -TypeName
 "Custom:RackInfo").Entries
```

출력:

```
Key          Value
---          -----
RackLocation Bay B/Row C/Rack D/Shelf E
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 ListInventoryEntries를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **ListResourceComplianceSummaries** 사용

다음 코드 예시는 ListResourceComplianceSummaries의 사용 방법을 보여줍니다.

CLI

AWS CLI

리소스 수준 규정 준수 요약 수를 나열하는 방법

이 예제에서는 리소스 수준 규정 준수 요약 수를 나열합니다.

명령:

```
aws ssm list-resource-compliance-summaries
```

출력:

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

특정 규정 준수 유형에 대한 리소스 수준 규정 준수 요약을 나열하는 방법

이 예제에서는 패치 규정 준수 유형에 대한 리소스 수준 규정 준수 요약을 나열합니다.

명령:

```
aws ssm list-resource-compliance-summaries --filters
 "Key=ComplianceType,Values=Patch,Type=EQUAL"
```

- API 세부 정보는 AWS CLI 명령 참조의 ListResourceComplianceSummaries를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 리소스 수준 요약 수를 가져옵니다. 요약에는 'Windows10'과 일치하는 제품의 규정 준수 및 비준수 상태에 대한 정보와 자세한 규정 준수 항목의 심각도 수가 포함됩니다. 파라미터가 지정되지 않은 경우 MaxResult의 기본값은 100이지만 이 값은 유효하지 않으므로 MaxResult 파라미터가 추가되고 값은 50으로 설정됩니다.

```
$FilterValues = @{
  "Key"="Product"
        "Type"="EQUAL"
        "Values"="Windows10"
}
        Get-SSMResourceComplianceSummaryList -Filter $FilterValues -MaxResult 50
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 ListResourceComplianceSummaries를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **ListTagsForResource** 사용

다음 코드 예시는 ListTagsForResource의 사용 방법을 보여줍니다.

CLI

AWS CLI

패치 기준에 적용된 태그를 나열하는 방법

다음 list-tags-for-resource 예제에서는 패치 기준의 태그를 나열합니다.

```
aws ssm list-tags-for-resource \
    --resource-type "PatchBaseline" \
    --resource-id "pb-0123456789abcdef0"
```

출력:

```
{
    "TagList": [
        {
            "Key": "Environment",
            "Value": "Production"
        },
        {
            "Key": "Region",
            "Value": "EMEA"
        }
    ]
}
```

자세한 내용은 AWS 일반 참조의 Tagging AWS Resources를 참조하세요.

- API 세부 정보는 AWS CLI Command Reference의 ListTagsForResource를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 유지 관리 기간의 태그를 나열합니다.

```
Get-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
 "MaintenanceWindow"
```

출력:

```
Key   Value
---   -----
Stack Production
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 ListTagsForResource를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **ModifyDocumentPermission** 사용

다음 코드 예시는 ModifyDocumentPermission의 사용 방법을 보여줍니다.

CLI

AWS CLI

문서 권한을 수정하는 방법

다음 modify-document-permission 예제에서는 Systems Manager 문서를 공개적으로 공유합니다.

```
aws ssm modify-document-permission \
    --name "Example" \
    --permission-type "Share" \
    --account-ids-to-add "All"
```

이 명령은 출력을 생성하지 않습니다.

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager 문서 공유를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 ModifyDocumentPermission을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 문서의 모든 계정에 '공유' 권한을 추가합니다. 명령이 성공해도 출력은 없습니다.

```
Edit-SSMDocumentPermission -Name "RunShellScript" -PermissionType "Share" -
AccountIdsToAdd all
```

예제 2: 이 예제에서는 문서의 특정 계정에 '공유' 권한을 추가합니다. 명령이 성공해도 출력은 없습니다.

```
Edit-SSMDocumentPermission -Name "RunShellScriptNew" -PermissionType "Share" -
AccountIdsToAdd "123456789012"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 ModifyDocumentPermission을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **PutComplianceItems** 사용

다음 코드 예시는 PutComplianceItems의 사용 방법을 보여줍니다.

CLI

AWS CLI

지정된 인스턴스에 규정 준수 유형 및 규정 준수 세부 정보를 등록하는 방법

이 예제에서는 지정된 관리형 인스턴스에 규정 준수 유형 Custom:AVCheck를 등록합니다. 명령이 성공해도 출력은 없습니다.

명령:

```
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"
```

- API 세부 정보는 AWS CLI 명령 참조의 PutComplianceItems를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 지정된 관리형 인스턴스에 대한 사용자 지정 규정 준수 항목을 작성합니다.

```
$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"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 PutComplianceItems를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **PutInventory** 사용

다음 코드 예시는 PutInventory의 사용 방법을 보여줍니다.

CLI

AWS CLI

인스턴스에 사용자 지정 메타데이터를 할당하는 방법

이번 예에서는 인스턴스에 랙 위치 정보를 할당합니다. 명령이 성공해도 출력은 없습니다.

명령(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"}]}]'
```

명령(Windows):

```
aws ssm put-inventory --instance-id "i-016648b75dd622dab" --items
 "TypeName=Custom:RackInfo,SchemaVersion=1.0,CaptureTime=2019-01-22T10:01:01Z,Content=[{R
 B/Row C/Rack D/Shelf F'}]"
```

- API 세부 정보는 AWS CLI 명령 참조의 PutInventory를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 인스턴스에 랙 위치 정보를 할당합니다. 명령이 성공해도 출력은 없습니다.

```
$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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 PutInventory를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **PutParameter** 사용

다음 코드 예시는 PutParameter의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 파라미터 값을 변경하는 방법

다음 put-parameter 예시에서는 지정된 파라미터의 값을 변경합니다.

```
aws ssm put-parameter \
    --name "MyStringParameter" \
    --type "String" \
    --value "Vici" \
    --overwrite
```

출력:

```
{
    "Version": 2,
    "Tier": "Standard"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager 파라미터 생성(AWS CLI), '파라미터 티어 관리<https://docs.aws.amazon.com/systems-manager/latest/userguide/parameter-store-advanced-parameters.html>'__ 및 파라미터 정책 작업을 참조하세요.

예 2: 고급 파라미터를 생성하는 방법

다음 put-parameter 예시에서는 고급 파라미터를 생성합니다.

```
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
```

출력:

```
{
    "Version": 1,
    "Tier": "Advanced"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager 파라미터 생성(AWS CLI), '파라미터 티어 관리<https://docs.aws.amazon.com/systems-manager/latest/userguide/parameter-store-advanced-parameters.html>'__ 및 파라미터 정책 작업을 참조하세요.

예 3: 표준 파라미터를 고급 파라미터로 변환하는 방법

다음 `put-parameter` 예시에서는 기존 표준 파라미터를 고급 파라미터로 변환합니다.

```
aws ssm put-parameter \
    --name "MyConvertedParameter" \
    --value "abc123" \
    --type "String" \
    --tier Advanced \
    --overwrite
```

출력:

```
{
    "Version": 2,
    "Tier": "Advanced"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager 파라미터 생성(AWS CLI), '파라미터 티어 관리<https://docs.aws.amazon.com/systems-manager/latest/userguide/parameter-store-advanced-parameters.html>'__ 및 파라미터 정책 작업을 참조하세요.

예 4: 정책이 연결된 파라미터를 생성하는 방법

다음 `put-parameter` 예시에서는 파라미터 정책이 연결된 고급 파라미터를 생성합니다.

```
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\"}}]"
```

출력:

```
{
    "Version": 1,
    "Tier": "Advanced"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager 파라미터 생성(AWS CLI), '파라미터 티어 관리<https://docs.aws.amazon.com/systems-manager/latest/userguide/parameter-store-advanced-parameters.html>'__ 및 파라미터 정책 작업을 참조하세요.

예 5: 기존 파라미터에 정책을 추가하는 방법

다음 `put-parameter` 예시에서는 정책을 기존 고급 파라미터에 연결합니다.

```
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
```

출력:

```
{
    "Version": 2,
    "Tier": "Advanced"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager 파라미터 생성(AWS CLI), '파라미터 티어 관리<https://docs.aws.amazon.com/systems-manager/latest/userguide/parameter-store-advanced-parameters.html>'__ 및 파라미터 정책 작업을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 PutParameter를 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
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);
        }
    }
}
```

- API에 대한 세부 정보는 AWS SDK for Java 2.x API 참조의 PutParameter를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 파라미터를 생성합니다. 명령이 성공해도 출력은 없습니다.

```
Write-SSMParameter -Name "Welcome" -Type "String" -Value "helloWorld"
```

예제 2: 이 예제에서는 파라미터를 변경합니다. 명령이 성공해도 출력은 없습니다.

```
Write-SSMParameter -Name "Welcome" -Type "String" -Value "Good day, Sunshine!" -
Overwrite $true
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 PutParameter를 참조하세요.

Rust

SDK for Rust

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS코드 예시 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
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(())
}
```

- API 세부 정보는 AWS SDK for Rust API 참조의 PutParameter를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **RegisterDefaultPatchBaseline** 사용

다음 코드 예시는 RegisterDefaultPatchBaseline의 사용 방법을 보여줍니다.

CLI

AWS CLI

기본 패치 기준을 설정하는 방법

다음 `register-default-patch-baseline` 예제에서는 지정된 사용자 지정 패치 기준을 지원하는 운영 체제 유형의 기본 패치 기준으로 등록합니다.

```
aws ssm register-default-patch-baseline \
    --baseline-id "pb-abc123cf9bEXAMPLE"
```

출력:

```
{
    "BaselineId":"pb-abc123cf9bEXAMPLE"
}
```

다음 `register-default-patch-baseline` 예제에서는 CentOS용 AWS에서 제공하는 기본 패치 기준을 기본 패치 기준으로 등록합니다.

```
aws ssm register-default-patch-baseline \
    --baseline-id "arn:aws:ssm:us-east-2:733109147000:patchbaseline/
pb-0574b43a65ea646ed"
```

출력:

```
{
    "BaselineId":"pb-abc123cf9bEXAMPLE"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 사전 정의된 패치 기준 및 사용자 지정 패치 기준 정보를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 RegisterDefaultPatchBaseline을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 패치 기준을 기본 패치 기준으로 등록합니다.

```
Register-SSMDefaultPatchBaseline -BaselineId "pb-03da896ca3b68b639"
```

출력:

```
pb-03da896ca3b68b639
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 RegisterDefaultPatchBaseline을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **RegisterPatchBaselineForPatchGroup** 사용

다음 코드 예시는 RegisterPatchBaselineForPatchGroup의 사용 방법을 보여줍니다.

CLI

AWS CLI

패치 그룹에 대해 패치 기준을 등록하는 방법

다음 register-patch-baseline-for-patch-group 예제에서는 패치 그룹의 패치 기준을 등록합니다.

```
aws ssm register-patch-baseline-for-patch-group \
    --baseline-id "pb-045f10b4f382baeda" \
    --patch-group "Production"
```

출력:

```
{
    "BaselineId": "pb-045f10b4f382baeda",
    "PatchGroup": "Production"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 패치 그룹 생성(<https://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-patch-group-tagging.html>__)과 패치 기준에 패치 그룹 추가를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 RegisterPatchBaselineForPatchGroup을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 패치 그룹의 패치 기준을 등록합니다.

```
Register-SSMPatchBaselineForPatchGroup -BaselineId "pb-03da896ca3b68b639" -
PatchGroup "Production"
```

출력:

```
BaselineId           PatchGroup
----------           ----------
pb-03da896ca3b68b639 Production
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 RegisterPatchBaselineForPatchGroup을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **RegisterTargetWithMaintenanceWindow** 사용

다음 코드 예시는 RegisterTargetWithMaintenanceWindow의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 유지 관리 기간에 단일 대상을 등록하는 방법

다음 register-target-with-maintenance-window 예제에서는 유지 관리 기간에 인스턴스를 등록합니다.

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-ab12cd34ef56gh78" \
    --target "Key=InstanceIds,Values=i-0000293ffd8c57862" \
    --owner-information "Single instance" \
    --resource-type "INSTANCE"
```

출력:

```
{
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

예제 2: 인스턴스 ID를 사용하여 유지 관리 기간에 여러 대상을 등록하는 방법

다음 register-target-with-maintenance-window 예제에서는 인스턴스 ID를 지정하여 유지 관리 기간에 두 인스턴스를 등록합니다.

```
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"
```

출력:

```
{
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

예제 3: 리소스 태그를 사용하여 유지 관리 기간에 대상을 등록하는 방법

다음 register-target-with-maintenance-window 예제에서는 인스턴스에 적용되는 리소스 태그를 지정하여 유지 관리 기간에 인스턴스를 등록합니다.

```
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"
```

출력:

```
{
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

예제 4: 태그 키 그룹을 사용하여 대상을 등록하는 방법

다음 `register-target-with-maintenance-window` 예제에서는 키 값에 상관없이 모두 하나 이상의 태그가 지정된 인스턴스를 등록합니다.

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-0c50858d01EXAMPLE" \
    --resource-type "INSTANCE" \
    --target "Key=tag-key,Values=Name,Instance-Type,CostCenter"
```

출력:

```
{
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
}
```

예제 5: 리소스 그룹 이름을 사용하여 대상을 등록하는 방법

다음 `register-target-with-maintenance-window` 예제에서는 포함된 리소스 유형에 상관없이 지정된 리소스 그룹을 등록합니다.

```
aws ssm register-target-with-maintenance-window \
    --window-id "mw-0c50858d01EXAMPLE" \
    --resource-type "RESOURCE_GROUP" \
    --target "Key=resource-groups:Name,Values=MyResourceGroup"
```

출력:

```
{
    "WindowTargetId":"1a2b3c4d-1a2b-1a2b-1a2b-1a2b3c4d-1a2"
```

```
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 유지 관리 기간에 대상 인스턴스 등록(AWS CLI)을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 RegisterTargetWithMaintenanceWindow를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 유지 관리 기간에 한 인스턴스를 등록합니다.

```
$option1 = @{Key="InstanceIds";Values=@("i-0000293ffd8c57862")}
Register-SSMTargetWithMaintenanceWindow -WindowId "mw-06cf17cbefcb4bf4f" -Target
 $option1 -OwnerInformation "Single instance" -ResourceType "INSTANCE"
```

출력:

```
d8e47760-23ed-46a5-9f28-927337725398
```

예제 2: 이 예제에서는 유지 관리 기간에 여러 인스턴스를 등록합니다.

```
$option1 =
 @{Key="InstanceIds";Values=@("i-0000293ffd8c57862","i-0cb2b964d3e14fd9f")}
Register-SSMTargetWithMaintenanceWindow -WindowId "mw-06cf17cbefcb4bf4f" -Target
 $option1 -OwnerInformation "Single instance" -ResourceType "INSTANCE"
```

출력:

```
6ab5c208-9fc4-4697-84b7-b02a6cc25f7d
```

예제 3: 이 예제에서는 EC2 태그를 사용하여 유지 관리 기간에 인스턴스를 등록합니다.

```
$option1 = @{Key="tag:Environment";Values=@("Production")}
Register-SSMTargetWithMaintenanceWindow -WindowId "mw-06cf17cbefcb4bf4f" -Target
 $option1 -OwnerInformation "Production Web Servers" -ResourceType "INSTANCE"
```

출력:

```
2994977e-aefb-4a71-beac-df620352f184
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 RegisterTargetWithMaintenanceWindow를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **RegisterTaskWithMaintenanceWindow** 사용

다음 코드 예시는 RegisterTaskWithMaintenanceWindow의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 유지 관리 기간에 자동화 작업을 등록하는 방법

다음 register-task-with-maintenance-window 예제에서는 인스턴스에서 대상으로 지정된 유지 관리 기간에 자동화 작업을 등록합니다.

```
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"
```

출력:

```
{
```

```
    "WindowTaskId":"11144444-5555-6666-7777-88888888"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 유지 관리 기간에 작업 등록(AWS CLI)을 참조하세요.

예제 2: 유지 관리 기간에 Lambda 작업을 등록하는 방법

다음 `register-task-with-maintenance-window` 예제에서는 인스턴스에서 대상으로 지정된 유지 관리 기간에 Lambda 작업을 등록합니다.

```
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"
```

출력:

```
{
    "WindowTaskId":"22244444-5555-6666-7777-88888888"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 유지 관리 기간에 작업 등록(AWS CLI)을 참조하세요.

예제 3: 유지 관리 기간에 Run Command 작업을 등록하는 방법

다음 `register-task-with-maintenance-window` 예제에서는 인스턴스에서 대상으로 지정된 유지 관리 기간에 Run Command 작업을 등록합니다.

```
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
```

출력:

```
{
    "WindowTaskId":"33344444-5555-6666-7777-88888888"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 유지 관리 기간에 작업 등록(AWS CLI)을 참조하세요.

예제 4: 유지 관리 기간에 Step Functions 작업을 등록하는 방법

다음 `register-task-with-maintenance-window` 예제에서는 인스턴스에서 대상으로 지정된 유지 관리 기간에 Step Functions 작업을 등록합니다.

```
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"
```

출력:

```
{
    "WindowTaskId":"44444444-5555-6666-7777-88888888"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 유지 관리 기간에 작업 등록(AWS CLI)을 참조하세요.

예제 5: 유지 관리 기간 대상 ID를 사용하여 작업을 등록하는 방법

다음 `register-task-with-maintenance-window` 예제에서는 유지 관리 기간 대상 ID를 사용하여 작업을 등록합니다. 유지 관리 기간 대상 ID는 `aws ssm register-target-with-maintenance-window` 명령 출력에 포함되어 있습니다. `aws ssm describe-maintenance-window-targets` 명령의 출력에서 검색할 수도 있습니다.

```
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
```

출력:

```
{
    "WindowTaskId":"33344444-5555-6666-7777-88888888"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 유지 관리 기간에 작업 등록(AWS CLI)을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 RegisterTaskWithMaintenanceWindow를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 인스턴스 ID를 사용하여 유지 관리 기간에 작업을 등록합니다. 출력은 작업 ID입니다.

```
$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
```

출력:

```
f34a2c47-ddfd-4c85-a88d-72366b69af1b
```

예제 2: 이 예제에서는 대상 ID를 사용하여 유지 관리 기간에 작업을 등록합니다. 출력은 작업 ID입니다.

```
$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
```

출력:

```
f34a2c47-ddfd-4c85-a88d-72366b69af1b
```

예제 3: 이 예제에서는 Run Command 문서 **AWS-RunPowerShellScript**에 대한 파라미터 객체를 생성하고 대상 ID를 사용하여 지정된 유지 관리 기간을 포함하는 작업을 생성합니다. 반환 출력은 작업 ID입니다.

```
$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
```

출력:

```
f1e2ef34-5678-12e3-456a-12334c5c6cbe
```

예제 4: 이 예제에서는 이름이 **Create-Snapshots**인 문서를 사용하여 AWS Systems Manager 자동화 작업을 등록합니다.

```
$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"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 RegisterTaskWithMaintenanceWindow를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **RemoveTagsFromResource** 사용

다음 코드 예시는 RemoveTagsFromResource의 사용 방법을 보여줍니다.

CLI

AWS CLI

패치 기준에서 태그를 삭제하는 방법

다음 remove-tags-from-resource 예제에서는 패치 기준에서 태그를 제거합니다.

```
aws ssm remove-tags-from-resource \
    --resource-type "PatchBaseline" \
    --resource-id "pb-0123456789abcdef0" \
    --tag-keys "Region"
```

이 명령은 출력을 생성하지 않습니다.

자세한 내용은 AWS 일반 참조의 Tagging AWS Resources를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 RemoveTagsFromResource를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 유지 관리 기간에서 태그를 제거합니다. 명령이 성공해도 출력은 없습니다.

```
Remove-SSMResourceTag -ResourceId "mw-03eb9db42890fb82d" -ResourceType
 "MaintenanceWindow" -TagKey "Production"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 RemoveTagsFromResource를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **SendCommand** 사용

다음 코드 예시는 SendCommand의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- Systems Manager 시작하기

CLI

AWS CLI

예제 1: 하나 이상의 원격 인스턴스에서 명령을 실행하는 방법

다음 send-command 예제에서는 대상 인스턴스에서 echo 명령을 실행합니다.

```
aws ssm send-command \
    --document-name "AWS-RunShellScript" \
    --parameters 'commands=["echo HelloWorld"]' \
    --targets "Key=instanceids,Values=i-1234567890abcdef0" \
    --comment "echo HelloWorld"
```

출력:

```
{
    "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
        }
    }
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager Run Command를 사용하여 명령 실행을 참조하세요.

예제 2: 인스턴스에 대한 IP 정보를 가져오는 방법

다음 send-command 예제에서는 인스턴스에 대한 IP 정보를 검색합니다.

```
aws ssm send-command \
```

```
    --instance-ids "i-1234567890abcdef0" \
    --document-name "AWS-RunShellScript" \
    --comment "IP config" \
    --parameters "commands=ifconfig"
```

샘플 출력은 예 1을 참조하세요.

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager Run Command를 사용하여 명령 실행을 참조하세요.

예제 3: 특정 태그를 사용하는 인스턴스에서 명령을 실행하는 방법

다음 send-command 예제에서는 태그 키가 'ENV'이고 값이 'Dev'인 인스턴스에서 명령을 실행합니다.

```
aws ssm send-command \
    --targets "Key=tag:ENV,Values=Dev" \
    --document-name "AWS-RunShellScript" \
    --parameters "commands=ifconfig"
```

샘플 출력은 예 1을 참조하세요.

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager Run Command를 사용하여 명령 실행을 참조하세요.

예제 4: SNS 알림을 보내는 명령을 실행하는 방법

다음 send-command 예제에서는 모든 알림 이벤트 및 Command 알림 유형에 대해 SNS 알림을 보내는 명령을 실행합니다.

```
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"
```

샘플 출력은 예 1을 참조하세요.

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager Run Command를 사용하여 명령 실행을 참조하세요.

예제 5: S3 및 CloudWatch로 출력하는 명령을 실행하는 방법

다음 send-command 예제에서는 명령 세부 정보를 S3 버킷 및 CloudWatch Logs 로그 그룹에 출력하는 명령을 실행합니다.

```
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"
```

샘플 출력은 예 1을 참조하세요.

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager Run Command를 사용하여 명령 실행을 참조하세요.

예제 6: 태그가 서로 다른 여러 인스턴스에서 명령을 실행하는 방법

다음 send-command 예제는 서로 다른 두 개의 태그 키와 값을 가진 인스턴스에서 명령을 실행합니다.

```
aws ssm send-command \
    --document-name "AWS-RunPowerShellScript" \
    --parameters commands=["echo helloWorld"] \
    --targets Key=tag:Env,Values=Dev Key=tag:Role,Values=WebServers
```

샘플 출력은 예 1을 참조하세요.

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager Run Command를 사용하여 명령 실행을 참조하세요.

예제 7: 태그 키가 같은 여러 인스턴스를 대상으로 지정하는 방법

다음 send-command 예제에서는 태그 키는 같지만 값이 다른 인스턴스에서 명령을 실행합니다.

```
aws ssm send-command \
    --document-name "AWS-RunPowerShellScript" \
    --parameters commands=["echo helloWorld"] \
```

```
    --targets Key=tag:Env,Values=Dev,Test
```

샘플 출력은 예 1을 참조하세요.

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager Run Command를 사용하여 명령 실행을 참조하세요.

예제 8: 공유 문서를 사용하는 명령을 실행하는 방법

다음 send-command 예제에서는 대상 인스턴스에서 공유 문서를 실행합니다.

```
aws ssm send-command \
    --document-name "arn:aws:ssm:us-east-1:123456789012:document/ExampleDocument"
 \
    --targets "Key=instanceids,Values=i-1234567890abcdef0"
```

샘플 출력은 예 1을 참조하세요.

자세한 내용은 AWS Systems Manager 사용 설명서의 공유 SSM 문서 사용을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 SendCommand를 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS코드 예시 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
   // Sends a SSM command to a managed node.
   public static String sendSSMCommand(SsmClient ssmClient, String documentName,
String instanceId) throws InterruptedException {
        // Before we use Document to send a command - make sure it is active.
        boolean isDocumentActive = false;
        DescribeDocumentRequest request = DescribeDocumentRequest.builder()
            .name(documentName)
            .build();

        while (!isDocumentActive) {
```

```
            DescribeDocumentResponse response =
ssmClient.describeDocument(request);
            String documentStatus = response.document().statusAsString();
            if (documentStatus.equals("Active")) {
                System.out.println("The Systems Manager document is active and
ready to use.");
                isDocumentActive = true;
            } else {
                System.out.println("The Systems Manager document is not active.
Status: " + documentStatus);
                try {
                    // Add a delay to avoid making too many requests.
                    Thread.sleep(5000); // Wait for 5 seconds before checking
again
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        }

        // Create the SendCommandRequest.
        SendCommandRequest commandRequest = SendCommandRequest.builder()
            .documentName(documentName)
            .instanceIds(instanceId)
            .build();

        // Send the command.
        SendCommandResponse commandResponse =
ssmClient.sendCommand(commandRequest);
        String commandId = commandResponse.command().commandId();
        System.out.println("The command Id is " + commandId);

        // Wait for the command execution to complete.
        GetCommandInvocationRequest invocationRequest =
GetCommandInvocationRequest.builder()
            .commandId(commandId)
            .instanceId(instanceId)
            .build();

        System.out.println("Wait 5 secs");
        TimeUnit.SECONDS.sleep(5);

        // Retrieve the command execution details.
```

```
        GetCommandInvocationResponse commandInvocationResponse =
 ssmClient.getCommandInvocation(invocationRequest);

        // 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);
        }
        return commandId;
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 SendCommand를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 대상 인스턴스에서 echo 명령을 실행합니다.

```
Send-SSMCommand -DocumentName "AWS-RunPowerShellScript" -Parameter @{commands =
 "echo helloWorld"} -Target @{Key="instanceids";Values=@("i-0cb2b964d3e14fd9f")}
```

출력:

```
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}
```

예제 2: 이 예제에서는 중첩된 파라미터를 수락하는 명령을 실행하는 방법을 보여줍니다.

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 SendCommand를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **StartAutomationExecution** 사용

다음 코드 예시는 StartAutomationExecution의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 자동화 문서를 실행하는 방법

다음 start-automation-execution 예제에서는 자동화 문서를 실행합니다.

```
aws ssm start-automation-execution \
    --document-name "AWS-UpdateLinuxAmi" \
    --parameters "AutomationAssumeRole=arn:aws:iam::123456789012:role/
SSMAutomationRole,SourceAmiId=ami-EXAMPLE,IamInstanceProfileName=EC2InstanceRole"
```

출력:

```
{
  "AutomationExecutionId": "4105a4fc-f944-11e6-9d32-0a1b2EXAMPLE"
```

```
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 수동으로 자동화 워크플로 실행을 참조하세요.

예제 2: 공유 자동화 문서를 실행하는 방법

다음 `start-automation-execution` 예제에서는 공유 자동화 문서를 실행합니다.

```
aws ssm start-automation-execution \
    --document-name "arn:aws:ssm:us-east-1:123456789012:document/ExampleDocument"
```

출력:

```
{
  "AutomationExecutionId": "4105a4fc-f944-11e6-9d32-0a1b2EXAMPLE"
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 공유 SSM 문서 사용을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 StartAutomationExecution을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 자동화 역할, AMI 소스 ID 및 Amazon EC2 인스턴스 역할을 지정하는 문서를 실행합니다.

```
Start-SSMAutomationExecution -DocumentName AWS-UpdateLinuxAmi -
Parameter @{'AutomationAssumeRole'='arn:aws:iam::123456789012:role/
SSMAutomationRole';'SourceAmiId'='ami-
f173cc91';'InstanceIamRole'='EC2InstanceRole'}
```

출력:

```
3a532a4f-0382-11e7-9df7-6f11185f6dd1
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 StartAutomationExecution을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **StopAutomationExecution** 사용

다음 코드 예시는 StopAutomationExecution의 사용 방법을 보여줍니다.

CLI

AWS CLI

자동화 실행을 중지하는 방법

다음 stop-automation-execution 예제에서는 자동화 문서를 중지합니다.

```
aws ssm stop-automation-execution
    --automation-execution-id "4105a4fc-f944-11e6-9d32-0a1b2EXAMPLE"
```

이 명령은 출력을 생성하지 않습니다.

자세한 내용은 AWS Systems Manager 사용 설명서의 수동으로 자동화 워크플로 실행을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 StopAutomationExecution을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 자동화 실행을 중지합니다. 명령이 성공해도 출력은 없습니다.

```
Stop-SSMAutomationExecution -AutomationExecutionId "4105a4fc-
f944-11e6-9d32-8fb2db27a909"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 StopAutomationExecution을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **UpdateAssociation** 사용

다음 코드 예시는 UpdateAssociation의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 문서 연결을 업데이트하는 방법

다음 update-association 예제에서는 새 문서 버전과의 연결을 업데이트합니다.

```
aws ssm update-association \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab" \
    --document-version "\$LATEST"
```

출력:

```
{
    "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
    }
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 새 연결 버전 편집 및 생성을 참조하세요.

예제 2: 연결의 일정 표현식을 업데이트하는 방법

다음 update-association 예제에서는 지정된 연결의 일정 표현식을 업데이트합니다.

```
aws ssm update-association \
    --association-id "8dfe3659-4309-493a-8755-0123456789ab" \
    --schedule-expression "cron(0 0 0/4 1/1 * ? *)"
```

출력:

```
{
    "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
    }
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 새 연결 버전 편집 및 생성을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 UpdateAssociation을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 연결을 새 문서 버전으로 업데이트합니다.

```
Update-SSMAssociation -AssociationId "93285663-92df-44cb-9f26-2292d4ecc439" -
DocumentVersion "1"
```

출력:

```
Name                 : AWS-UpdateSSMAgent
InstanceId           :
Date                 : 3/1/2017 6:22:21 PM
Status.Name          :
Status.Date          :
Status.Message       :
Status.AdditionalInfo :
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 UpdateAssociation을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **UpdateAssociationStatus** 사용

다음 코드 예시는 UpdateAssociationStatus의 사용 방법을 보여줍니다.

CLI

AWS CLI

연결 상태를 업데이트하는 방법

다음 update-association-status 예제에서는 인스턴스와 문서 간 연결의 연결 상태를 업데이트합니다.

```
aws ssm update-association-status \
    --name "AWS-UpdateSSMAgent" \
```

```
    --instance-id "i-1234567890abcdef0" \
    --association-status
 "Date=1424421071.939,Name=Pending,Message=temp_status_change,AdditionalInfo=Additional-
Config-Needed"
```

출력:

```
{
    "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
    }
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 Systems Manager에서 연결 작업을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 UpdateAssociationStatus를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 인스턴스 및 구성 문서 간 연결 상태를 업데이트합니다.

```
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"
```

출력:

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 UpdateAssociationStatus를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **UpdateDocument** 사용

다음 코드 예시는 UpdateDocument의 사용 방법을 보여줍니다.

CLI

AWS CLI

문서의 새 버전을 생성하는 방법

다음 update-document 예제에서는 Windows 컴퓨터에서 실행 시 문서의 새 버전을 생성합니다. --document에서 지정한 문서는 JSON 형식이어야 합니다. 콘텐츠 파일 경로 앞에서 file://을 참조해야 합니다. --document-version 파라미터의 시작 위치에 $이 있으므로 Windows에서는 값을 큰따옴표로 묶어야 합니다. Linux, MacOS 또는 PowerShell 프롬프트에서는 값을 작은따옴표로 묶어야 합니다.

Windows 버전:

```
aws ssm update-document \
    --name "RunShellScript" \
    --content "file://RunShellScript.json" \
    --document-version "$LATEST"
```

Linux 및 Mac 버전:

```
aws ssm update-document \
    --name "RunShellScript" \
    --content "file://RunShellScript.json" \
    --document-version '$LATEST'
```

출력:

```
{
  "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"
  }
}
```

- API 세부 정보는 AWS CLI 명령 참조의 UpdateDocument를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 지정한 json 파일의 업데이트된 콘텐츠를 포함하는 문서의 새 버전을 생성합니다. 문서는 JSON 형식이어야 합니다. 'Get-SSMDocumentVersionList' cmdlet을 사용하여 문서 버전을 얻을 수 있습니다.

```
Update-SSMDocument -Name RunShellScript -DocumentVersion "1" -Content (Get-
Content -Raw "c:\temp\RunShellScript.json")
```

출력:

```
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
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 UpdateDocument를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **UpdateDocumentDefaultVersion** 사용

다음 코드 예시는 UpdateDocumentDefaultVersion의 사용 방법을 보여줍니다.

CLI

AWS CLI

문서의 기본 버전을 업데이트하는 방법

다음 update-document-default-version 예제에서는 Systems Manager 문서의 기본 버전을 업데이트합니다.

```
aws ssm update-document-default-version \
    --name "Example" \
    --document-version "2"
```

출력:

```
{
    "Description": {
        "Name": "Example",
        "DefaultVersion": "2"
    }
}
```

자세한 내용은 AWS Systems Manager 사용 설명서의 SSM 문서 콘텐츠 작성을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 UpdateDocumentDefaultVersion을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 여기에서는 문서의 기본 버전을 업데이트합니다. 'Get-SSMDocumentVersionList' cmdlet을 사용하여 사용 가능한 문서 버전을 얻을 수 있습니다.

```
Update-SSMDocumentDefaultVersion -Name "RunShellScript" -DocumentVersion "2"
```

출력:

```
DefaultVersion Name
-------------- ----
2              RunShellScript
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 UpdateDocumentDefaultVersion을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **UpdateMaintenanceWindow** 사용

다음 코드 예시는 UpdateMaintenanceWindow의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 유지 관리 기간을 업데이트하는 방법

다음 update-maintenance-window 예제에서는 유지 관리 기간의 이름을 업데이트합니다.

```
aws ssm update-maintenance-window \
    --window-id "mw-1a2b3c4d5e6f7g8h9" \
    --name "My-Renamed-MW"
```

출력:

```
{
    "Cutoff": 1,
    "Name": "My-Renamed-MW",
    "Schedule": "cron(0 16 ? * TUE *)",
    "Enabled": true,
    "AllowUnassociatedTargets": true,
```

```
    "WindowId": "mw-1a2b3c4d5e6f7g8h9",
    "Duration": 4
}
```

예제 2: 유지 관리 기간을 비활성화하는 방법

다음 update-maintenance-window 예제에서는 유지 관리 기간을 비활성화합니다.

```
aws ssm update-maintenance-window \
    --window-id "mw-1a2b3c4d5e6f7g8h9" \
    --no-enabled
```

예제 3: 유지 관리 기간을 활성화하는 방법

다음 update-maintenance-window 예제에서는 유지 관리 기간을 활성화합니다.

```
aws ssm update-maintenance-window \
    --window-id "mw-1a2b3c4d5e6f7g8h9" \
    --enabled
```

자세한 내용은 AWS Systems Manager 사용 설명서의 유지 관리 기간 업데이트(AWS CLI)를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 UpdateMaintenanceWindow를 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS코드 예시 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    // Update the maintenance window schedule
    public static void updateSSMMaintenanceWindow(SsmClient ssmClient, String id,
String name) {
        try {
```

```
            UpdateMaintenanceWindowRequest updateRequest =
 UpdateMaintenanceWindowRequest.builder()
                .windowId(id)
                .allowUnassociatedTargets(true)
                .duration(24)
                .enabled(true)
                .name(name)
                .schedule("cron(0 0 ? * MON *)")
                .build();

            ssmClient.updateMaintenanceWindow(updateRequest);
            System.out.println("The Systems Manager maintenance window was
 successfully updated.");

        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 UpdateMaintenanceWindow를 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 유지 관리 기간의 이름을 업데이트합니다.

```
Update-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d" -Name "My-Renamed-
MW"
```

출력:

```
AllowUnassociatedTargets : False
Cutoff                   : 1
Duration                 : 2
Enabled                  : True
Name                     : My-Renamed-MW
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d
```

예제 2: 이 예제에서는 유지 관리 기간을 활성화합니다.

```
Update-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d" -Enabled $true
```

출력:

```
AllowUnassociatedTargets : False
Cutoff                   : 1
Duration                 : 2
Enabled                  : True
Name                     : My-Renamed-MW
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d
```

예제 3: 이 예제에서는 유지 관리 기간을 비활성화합니다.

```
Update-SSMMaintenanceWindow -WindowId "mw-03eb9db42890fb82d" -Enabled $false
```

출력:

```
AllowUnassociatedTargets : False
Cutoff                   : 1
Duration                 : 2
Enabled                  : False
Name                     : My-Renamed-MW
Schedule                 : cron(0 */30 * * * ? *)
WindowId                 : mw-03eb9db42890fb82d
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 UpdateMaintenanceWindow를 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **UpdateManagedInstanceRole** 사용

다음 코드 예시는 UpdateManagedInstanceRole의 사용 방법을 보여줍니다.

CLI

AWS CLI

관리형 인스턴스의 IAM 역할을 업데이트하는 방법

다음 `update-managed-instance-role` 예제에서는 관리형 인스턴스의 IAM 인스턴스 프로파일을 업데이트합니다.

```
aws ssm update-managed-instance-role \
    --instance-id "mi-08ab247cdfEXAMPLE" \
    --iam-role "ExampleRole"
```

이 명령은 출력을 생성하지 않습니다.

자세한 내용은 AWS Systems Manager 사용 설명서의 4단계: Systems Manager에 대한 IAM 인스턴스 프로파일 생성을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 UpdateManagedInstanceRole을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 관리형 인스턴스의 역할을 업데이트합니다. 명령이 성공해도 출력은 없습니다.

```
Update-SSMManagedInstanceRole -InstanceId "mi-08ab247cdf1046573" -IamRole
 "AutomationRole"
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 UpdateManagedInstanceRole을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **UpdateOpsItem** 사용

다음 코드 예시는 `UpdateOpsItem`의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- Systems Manager 시작하기

CLI

AWS CLI

OpsItem을 업데이트하는 방법

다음 `update-ops-item` 예시에서는 OpsItem의 설명, 우선순위 및 범주를 업데이트합니다. 또한 이 명령은 이 OpsItem을 편집하거나 변경할 때 알림이 전송되는 SNS 주제를 지정합니다.

```
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"
```

출력:

```
This command produces no output.
```

자세한 내용은 AWS Systems Manager 사용 설명서의 OpsItems 작업을 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 UpdateOpsItem을 참조하세요.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    public static void resolveOpsItem(SsmClient ssmClient, String opsID) {
```

```
        try {
            UpdateOpsItemRequest opsItemRequest = UpdateOpsItemRequest.builder()
                .opsItemId(opsID)
                .status(OpsItemStatus.RESOLVED)
                .build();

            ssmClient.updateOpsItem(opsItemRequest);

        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

- API에 대한 세부 정보는 AWS SDK for Java 2.x API 참조의 UpdateOpsItem을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 CLI와 함께 **UpdatePatchBaseline** 사용

다음 코드 예시는 UpdatePatchBaseline의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 패치 기준을 업데이트하는 방법

다음 update-patch-baseline 예제에서는 지정된 두 개의 패치를 거부된 패치로 추가하고 하나의 패치를 기존 패치 기준에 승인된 패치로 추가합니다.

```
aws ssm update-patch-baseline \
        --baseline-id "pb-0123456789abcdef0" \
        --rejected-patches "KB2032276" "MS10-048" \
        --approved-patches "KB2124261"
```

출력:

```
{
```

```
    "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": []
}
```

예제 2: 패치 기준의 이름을 바꾸는 방법

다음 update-patch-baseline 예제에서는 지정된 패치 기준의 이름을 바꿉니다.

```
aws ssm update-patch-baseline \
    --baseline-id "pb-0713accee01234567" \
    --name "Windows-Server-2012-R2-Important-and-Critical-Security-Updates"
```

자세한 내용은 AWS Systems Manager 사용 설명서의 패치 기준 업데이트 또는 삭제(<https://docs.aws.amazon.com/systems-manager/latest/userguide/patch-baseline-update-or-delete.html>`__)를 참조하세요.

- API 세부 정보는 AWS CLI 명령 참조의 UpdatePatchBaseline을 참조하세요.

PowerShell

PowerShell용 도구

예제 1: 이 예제에서는 기존 패치 기준에 두 개의 패치를 거부된 패치로 추가하고 한 개의 패치를 승인된 패치로 추가합니다.

```
Update-SSMPatchBaseline -BaselineId "pb-03da896ca3b68b639" -RejectedPatch
 "KB2032276","MS10-048" -ApprovedPatch "KB2124261"
```

출력:

```
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}
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조의 UpdatePatchBaseline을 참조하세요.

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK를 사용한 Systems Manager 시나리오

다음 코드 예제는 AWS SDK를 사용하여 Systems Manager에서 일반적인 시나리오를 구현하는 방법을 보여줍니다. 이러한 시나리오에서는 Systems Manager 내에서 여러 함수를 직접 호출하여 특정 작업을 수행하는 방법을 보여줍니다. 각 시나리오에는 GitHub에 대한 링크가 포함되어 있습니다. 여기에서 코드를 설정하고 실행하는 방법에 대한 지침을 찾을 수 있습니다.

예제

- AWS SDK를 사용하여 Systems Manager 시작하기

AWS SDK를 사용하여 Systems Manager 시작하기

다음 코드 예제는 Systems Manager 유지 관리 기간, 문서 및 OpsItems로 작업하는 방법을 보여줍니다.

Java

SDK for Java 2.x

> **Note**
>
> GitHub에 더 많은 내용이 있습니다. AWS코드 예시 리포지토리에서 전체 예시를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.ssm.SsmClient;
import software.amazon.awssdk.services.ssm.model.CommandInvocation;
import software.amazon.awssdk.services.ssm.model.CommandInvocationStatus;
import software.amazon.awssdk.services.ssm.model.CreateDocumentRequest;
import software.amazon.awssdk.services.ssm.model.CreateDocumentResponse;
import software.amazon.awssdk.services.ssm.model.CreateMaintenanceWindowRequest;
import software.amazon.awssdk.services.ssm.model.CreateMaintenanceWindowResponse;
import software.amazon.awssdk.services.ssm.model.CreateOpsItemRequest;
import software.amazon.awssdk.services.ssm.model.CreateOpsItemResponse;
import software.amazon.awssdk.services.ssm.model.DeleteDocumentRequest;
import software.amazon.awssdk.services.ssm.model.DeleteMaintenanceWindowRequest;
import software.amazon.awssdk.services.ssm.model.DeleteOpsItemRequest;
import software.amazon.awssdk.services.ssm.model.DescribeDocumentRequest;
import software.amazon.awssdk.services.ssm.model.DescribeDocumentResponse;
```

```
import
 software.amazon.awssdk.services.ssm.model.DescribeMaintenanceWindowsRequest;
import
 software.amazon.awssdk.services.ssm.model.DescribeMaintenanceWindowsResponse;
import software.amazon.awssdk.services.ssm.model.DescribeOpsItemsRequest;
import software.amazon.awssdk.services.ssm.model.DescribeOpsItemsResponse;
import software.amazon.awssdk.services.ssm.model.DocumentAlreadyExistsException;
import software.amazon.awssdk.services.ssm.model.DocumentType;
import software.amazon.awssdk.services.ssm.model.GetCommandInvocationRequest;
import software.amazon.awssdk.services.ssm.model.GetCommandInvocationResponse;
import software.amazon.awssdk.services.ssm.model.GetOpsItemRequest;
import software.amazon.awssdk.services.ssm.model.GetOpsItemResponse;
import software.amazon.awssdk.services.ssm.model.ListCommandInvocationsRequest;
import software.amazon.awssdk.services.ssm.model.ListCommandInvocationsResponse;
import software.amazon.awssdk.services.ssm.model.MaintenanceWindowFilter;
import software.amazon.awssdk.services.ssm.model.MaintenanceWindowIdentity;
import software.amazon.awssdk.services.ssm.model.OpsItemDataValue;
import software.amazon.awssdk.services.ssm.model.OpsItemFilter;
import software.amazon.awssdk.services.ssm.model.OpsItemFilterKey;
import software.amazon.awssdk.services.ssm.model.OpsItemFilterOperator;
import software.amazon.awssdk.services.ssm.model.OpsItemStatus;
import software.amazon.awssdk.services.ssm.model.OpsItemSummary;
import software.amazon.awssdk.services.ssm.model.SendCommandRequest;
import software.amazon.awssdk.services.ssm.model.SendCommandResponse;
import software.amazon.awssdk.services.ssm.model.SsmException;
import software.amazon.awssdk.services.ssm.model.UpdateMaintenanceWindowRequest;
import software.amazon.awssdk.services.ssm.model.UpdateOpsItemRequest;
import java.time.ZoneId;
import java.time.format.DateTimeFormatter;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Scanner;
import java.util.concurrent.TimeUnit;

/**
 * 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/setup.html
 *
 *
```

```
 * This Java program performs these tasks:
 * 1. Creates an AWS Systems Manager maintenance window with a default name or a
 user-provided name.
 * 2. Modifies the maintenance window schedule.
 * 3. Creates a Systems Manager document with a default name or a user-provided
 name.
 * 4. Sends a command to a specified EC2 instance using the created Systems
 Manager document and displays the time when the command was invoked.
 * 5. Creates a Systems Manager OpsItem with a predefined title, source,
 category, and severity.
 * 6. Updates and resolves the created OpsItem.
 * 7. Deletes the Systems Manager maintenance window, OpsItem, and document.
 */

public class SSMScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
    public static void main(String[] args) throws InterruptedException {
        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);
        String documentName;
        String windowName;
        String instanceId = args[0];
        String title = "Disk Space Alert" ;
```

```
        String source = "EC2" ;
        String category = "Performance" ;
        String severity = "2" ;

        Region region = Region.US_EAST_1;
        SsmClient ssmClient = SsmClient.builder()
            .region(region)
            .build();

        System.out.println(DASHES);
        System.out.println("""
            Welcome to the AWS Systems Manager SDK Getting Started scenario.
            This program demonstrates how to interact with Systems Manager using
the AWS SDK for Java (v2).
            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...
            Please hit Enter
            """);
        scanner.nextLine();
        System.out.println(DASHES);

        System.out.println("Create a Systems Manager 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 = createMaintenanceWindow(ssmClient, windowName);
        System.out.println(DASHES);

        System.out.println("Modify the maintenance window by changing the
schedule");
        System.out.println("Please hit Enter");
        scanner.nextLine();
        updateSSMMaintenanceWindow(ssmClient, winId, windowName);
        System.out.println(DASHES);

        System.out.println("Create a document that defines the actions that
Systems Manager performs on your EC2 instance.");
```

```
        System.out.println("Please enter the document name (default is
ssmdocument):");
        String doc = scanner.nextLine();
        documentName = doc.isEmpty() ? "ssmdocument" : doc;
        createSSMDoc(ssmClient, documentName);

        System.out.println("Now we are going to run a command on an EC2 instance
that echoes 'Hello, world!'");
        System.out.println("Please hit Enter");
        scanner.nextLine();
        String commandId = sendSSMCommand(ssmClient, documentName, instanceId);
        System.out.println(DASHES);

        System.out.println("Lets get the time when the specific command was sent
to the specific managed node");
        System.out.println("Please hit Enter");
        scanner.nextLine();
        displayCommands(ssmClient, commandId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
             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.
            """);

        System.out.println("Please hit Enter");
        scanner.nextLine();
        String opsItemId = createSSMOpsItem(ssmClient, title, source, category,
severity);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Now we will update  the OpsItem "+opsItemId);
        System.out.println("Please hit Enter");
```

```
        scanner.nextLine();
        String description = "An update to "+opsItemId ;
        updateOpsItem(ssmClient, opsItemId, title, description);
        System.out.println("Now we will get the status of the OpsItem
"+opsItemId);
        System.out.println("Please hit Enter");
        scanner.nextLine();
        describeOpsItems(ssmClient, opsItemId);
        System.out.println("Now we will resolve the OpsItem "+opsItemId);
        System.out.println("Please hit Enter");
        scanner.nextLine();
        resolveOpsItem(ssmClient, opsItemId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Would you like to delete the Systems Manager
resources? (y/n)");
        String delAns = scanner.nextLine().trim();
        if (delAns.equalsIgnoreCase("y")) {
            System.out.println("You selected to delete the resources.");
            System.out.print("Press Enter to continue...");
            scanner.nextLine();
            deleteOpsItem(ssmClient, opsItemId);
            deleteMaintenanceWindow(ssmClient, winId);
            deleteDoc(ssmClient, documentName);
        } else {
            System.out.println("The Systems Manager resources will not be
deleted");
        }
        System.out.println(DASHES);

        System.out.println("This concludes the Systems Manager SDK Getting
Started scenario.");
        System.out.println(DASHES);
    }

    // Displays the date and time when the specific command was invoked.
    public static void displayCommands(SsmClient ssmClient, String commandId) {
        try {
            ListCommandInvocationsRequest commandInvocationsRequest =
ListCommandInvocationsRequest.builder()
                .commandId(commandId)
                .build();
```

```
            ListCommandInvocationsResponse response =
ssmClient.listCommandInvocations(commandInvocationsRequest);
            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()));
            }

        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    // Create an SSM OpsItem
    public static String createSSMOpsItem(SsmClient ssmClient, String title,
String source, String category, String severity) {
        try {
            CreateOpsItemRequest opsItemRequest = CreateOpsItemRequest.builder()
                .description("Created by the Systems Manager Java API")
                .title(title)
                .source(source)
                .category(category)
                .severity(severity)
                .build();

            CreateOpsItemResponse itemResponse =
ssmClient.createOpsItem(opsItemRequest);
            return itemResponse.opsItemId();

        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
       return "";
    }

    // Update the AWS SSM OpsItem.
    public static void updateOpsItem(SsmClient ssmClient, 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());

        try {
            UpdateOpsItemRequest request = UpdateOpsItemRequest.builder()
                .opsItemId(opsItemId)
                .title(title)
                .operationalData(operationalData)
                .status(getOpsItem(ssmClient, opsItemId))
                .description(description)
                .build();

            ssmClient.updateOpsItem(request);

        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public static void resolveOpsItem(SsmClient ssmClient, String opsID) {
        try {
            UpdateOpsItemRequest opsItemRequest = UpdateOpsItemRequest.builder()
                .opsItemId(opsID)
                .status(OpsItemStatus.RESOLVED)
                .build();

            ssmClient.updateOpsItem(opsItemRequest);

        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    // Gets a specific OpsItem.
    private static OpsItemStatus getOpsItem(SsmClient ssmClient, String
opsItemId) {
        GetOpsItemRequest itemRequest = GetOpsItemRequest.builder()
            .opsItemId(opsItemId)
            .build();
```

```
        try {
            GetOpsItemResponse response = ssmClient.getOpsItem(itemRequest);
            return response.opsItem().status();

        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
        return null;
    }

    // Sends a SSM command to a managed node.
    public static String sendSSMCommand(SsmClient ssmClient, String documentName,
String instanceId) throws InterruptedException {
        // Before we use Document to send a command - make sure it is active.
        boolean isDocumentActive = false;
        DescribeDocumentRequest request = DescribeDocumentRequest.builder()
            .name(documentName)
            .build();

        while (!isDocumentActive) {
            DescribeDocumentResponse response =
ssmClient.describeDocument(request);
            String documentStatus = response.document().statusAsString();
            if (documentStatus.equals("Active")) {
                System.out.println("The Systems Manager document is active and
ready to use.");
                isDocumentActive = true;
            } else {
                System.out.println("The Systems Manager document is not active.
Status: " + documentStatus);
                try {
                    // Add a delay to avoid making too many requests.
                    Thread.sleep(5000); // Wait for 5 seconds before checking
again
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        }

        // Create the SendCommandRequest.
        SendCommandRequest commandRequest = SendCommandRequest.builder()
            .documentName(documentName)
```

```
            .instanceIds(instanceId)
            .build();

        // Send the command.
        SendCommandResponse commandResponse =
ssmClient.sendCommand(commandRequest);
        String commandId = commandResponse.command().commandId();
        System.out.println("The command Id is " + commandId);

        // Wait for the command execution to complete.
        GetCommandInvocationRequest invocationRequest =
GetCommandInvocationRequest.builder()
            .commandId(commandId)
            .instanceId(instanceId)
            .build();

        System.out.println("Wait 5 secs");
        TimeUnit.SECONDS.sleep(5);

        // Retrieve the command execution details.
        GetCommandInvocationResponse commandInvocationResponse =
ssmClient.getCommandInvocation(invocationRequest);

        // 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);
        }
        return commandId;
    }

    // Deletes an AWS Systems Manager document.
    public static void deleteDoc(SsmClient ssmClient, String documentName) {
        try {
            DeleteDocumentRequest documentRequest =
DeleteDocumentRequest.builder()
                .name(documentName)
                .build();

            ssmClient.deleteDocument(documentRequest);
            System.out.println("The Systems Manager document was successfully
deleted.");
```

```
        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public static void deleteMaintenanceWindow(SsmClient ssmClient, String winId)
{
        try {
            DeleteMaintenanceWindowRequest windowRequest =
DeleteMaintenanceWindowRequest.builder()
                .windowId(winId)
                .build();

            ssmClient.deleteMaintenanceWindow(windowRequest);
            System.out.println("The maintenance window was successfully
deleted.");

        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    // Update the maintenance window schedule
    public static void updateSSMMaintenanceWindow(SsmClient ssmClient, String id,
String name) {
        try {
            UpdateMaintenanceWindowRequest updateRequest =
UpdateMaintenanceWindowRequest.builder()
                .windowId(id)
                .allowUnassociatedTargets(true)
                .duration(24)
                .enabled(true)
                .name(name)
                .schedule("cron(0 0 ? * MON *)")
                .build();

            ssmClient.updateMaintenanceWindow(updateRequest);
            System.out.println("The Systems Manager maintenance window was
successfully updated.");

        } catch (SsmException e) {
```

```
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public static String createMaintenanceWindow(SsmClient ssmClient, String
winName) {
        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();

        try {
            CreateMaintenanceWindowResponse response =
ssmClient.createMaintenanceWindow(request);
            String maintenanceWindowId = response.windowId();
            System.out.println("The maintenance window id is " +
maintenanceWindowId);
            return maintenanceWindowId;

        } catch (DocumentAlreadyExistsException e) {
            System.err.println("The maintenance window already exists. Moving
on.");
        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }

        MaintenanceWindowFilter filter = MaintenanceWindowFilter.builder()
            .key("name")
            .values(winName)
            .build();

        DescribeMaintenanceWindowsRequest winRequest =
DescribeMaintenanceWindowsRequest.builder()
            .filters(filter)
            .build();

        String windowId = "";
```

```
        DescribeMaintenanceWindowsResponse response =
ssmClient.describeMaintenanceWindows(winRequest);
        List<MaintenanceWindowIdentity> windows = response.windowIdentities();
        if (!windows.isEmpty()) {
            windowId = windows.get(0).windowId();
            System.out.println("Window ID: " + windowId);
        } else {
            System.out.println("Window not found.");
        }
       return windowId;
   }

   // Create an AWS SSM document to use in this scenario.
   public static void createSSMDoc(SsmClient ssmClient, String docName) {
        // Create JSON for the content
        String jsonData = """
            {
                "schemaVersion": "2.2",
                "description": "Run a simple shell command",
                "mainSteps": [
                    {
                        "action": "aws:runShellScript",
                        "name": "runEchoCommand",
                        "inputs": {
                          "runCommand": [
                            "echo 'Hello, world!'"
                          ]
                        }
                      }
                    ]
                }
            """;

        try {
            CreateDocumentRequest request = CreateDocumentRequest.builder()
                .content(jsonData)
                .name(docName)
                .documentType(DocumentType.COMMAND)
                .build();

            // Create the document.
            CreateDocumentResponse response = ssmClient.createDocument(request);
            System.out.println("The status of the document is " +
response.documentDescription().status());
```

```
        } catch (DocumentAlreadyExistsException e) {
            System.err.println("The document already exists. Moving on." );
        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public static void describeOpsItems(SsmClient ssmClient, String key) {
        try {
            OpsItemFilter filter = OpsItemFilter.builder()
                .key(OpsItemFilterKey.OPS_ITEM_ID)
                .values(key)
                .operator(OpsItemFilterOperator.EQUAL)
                .build();

            DescribeOpsItemsRequest itemsRequest =
DescribeOpsItemsRequest.builder()
                .maxResults(10)
                .opsItemFilters(filter)
                .build();

            DescribeOpsItemsResponse itemsResponse =
ssmClient.describeOpsItems(itemsRequest);
            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());
            }

        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public static void deleteOpsItem(SsmClient ssmClient, String opsId) {
        try {
            DeleteOpsItemRequest deleteOpsItemRequest =
DeleteOpsItemRequest.builder()
                .opsItemId(opsId)
                .build();
```

```
            ssmClient.deleteOpsItem(deleteOpsItemRequest);
            System.out.println(opsId +" Opsitem was deleted");

        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 다음 주제를 참조하십시오.
  - CommandInvocations
  - CreateDocument
  - CreateMaintenanceWindow
  - CreateOpsItem
  - DeleteMaintenanceWindow
  - SendCommand
  - UpdateOpsItem

AWS SDK 개발자 가이드 및 코드 예시의 전체 목록은 AWS SDK를 사용하여 Systems Manager 사용 단원을 참조하세요. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS Systems Manager 모니터링

모니터링은 AWS Systems Manager와 사용자 AWS 솔루션의 신뢰성, 가용성 및 성능을 유지하는 중요한 역할을 합니다. 발생하는 다중 지점 실패를 디버깅할 수 있도록 AWS 솔루션의 모든 부분으로부터 모니터링 데이터를 수집해야 합니다. Systems Manager 모니터링을 시작하기 전에 다음 질문에 대한 답변을 포함하는 모니터링 계획을 생성합니다.

- 모니터링의 목표
- 모니터링할 리소스
- 이러한 리소스를 모니터링하는 빈도
- 사용할 모니터링 도구
- 누가 모니터링 작업을 수행합니까?
- 문제 발생 시 알려야 할 대상

모니터링 목표를 정의하고 모니터링 계획을 생성했으면, 다음 단계는 환경에서 Systems Manager 성능의 기준을 설정하는 것입니다. 다양한 시간과 다양한 로드 조건에서 Systems Manager 성능을 측정해야 합니다. Systems Manager를 모니터링할 때 수집한 모니터링 데이터의 기록을 저장해야 합니다. 현재 Systems Manager 성능을 이 기록 데이터와 비교하면 일반적인 성능 패턴과 성능 이상을 식별하고 이를 해결할 방법을 생성할 수 있습니다.

예를 들어 자동화 워크플로, 패치 기준선 적용, 유지 관리 기간 이벤트 및 구성 규정 준수 등과 같은 작업의 성공 또는 실패를 모니터링할 수 있습니다. Automation은 AWS Systems Manager의 기능입니다.

또한 관리형 노드의 CPU 사용률, 디스크 I/O 및 네트워크 사용률을 모니터링할 수도 있습니다. 설정한 기준 이하로 성능이 떨어지면 노드를 재구성하거나 최적화하여 CPU 사용률을 줄이거나 디스크 I/O를 개선하거나 네트워크 트래픽을 줄일 수 있습니다. EC2 인스턴서 모니터링에 대한 자세한 내용은 Amazon EC2 사용 설명서의 Amazon EC2 모니터링을 참조하세요.

주제

- AWS CloudTrail을 사용하여 AWS Systems ManagerAPI 호출을 로깅
- CloudWatch Logs로 Automation 작업 출력 로깅
- Run Command에 대한 Amazon CloudWatch Logs 구성
- Amazon EventBridge로 Systems Manager 이벤트 모니터링
- Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링

모니터링 도구

이 장의 내용은 Systems Manager 및 기타 AWS 리소스를 모니터링하는 데 사용할 수 있는 도구 사용 정보를 제공합니다. 전체 도구 목록은 AWS Systems Manager의 로깅 및 모니터링 섹션을 참조하세요.

통합 CloudWatch Logs로 노드 로그 전송(CloudWatch 에이전트)

Amazon CloudWatch 에이전트를 구성하고 사용하면, 이러한 태스크에 대해 AWS Systems Manager Agent(SSM Agent)를 사용하는 대신, 노드로부터 지표 및 로그를 수집할 수 있습니다. CloudWatch 에이전트를 사용하면 SSM Agent를 사용할 때보다 EC2 인스턴스에 대한 지표를 더 많이 수집할 수 있습니다. 또한 CloudWatch 에이전트를 사용하면 온프레미스 서버로부터 지표를 수집할 수 있습니다.

CloudWatch 에이전트에 사용하기 위해 Systems Manager Parameter Store에 에이전트 구성 설정을 저장할 수도 있습니다. Parameter Store는 AWS Systems Manager의 기능입니다.

> **Note**
>
> AWS Systems Manager는 64비트 버전의 Windows에서만 로그 및 지표를 수집하기 위해 SSM Agent에서 통합 CloudWatch 에이전트로의 마이그레이션을 지원합니다. 다른 운영 체제에서 통합 CloudWatch 에이전트 설정에 대한 자세한 내용과 CloudWatch 에이전트 사용에 대한 전체 내용은 Amazon CloudWatch 사용 설명서의 CloudWatch 에이전트를 사용하여 Amazon EC2 인스턴스 및 온프레미스 서버로부터 지표 및 로그 수집을 참조하세요.
> 지원되는 다른 운영 체제에서 CloudWatch 에이전트를 사용할 수 있지만 Systems Manager를 사용하여 도구 마이그레이션을 수행할 수는 없습니다.

SSM Agent는 실행, 예약된 작업, 오류 및 상태에 대한 정보를 각 노드의 로그 파일에 씁니다. 노드에 수동으로 연결하여 로그 파일을 보고 SSM Agent 문제를 해결하려면 많은 시간이 소요됩니다. 보다 효율적으로 노드를 모니터링하려면 SSM Agent 자체 또는 CloudWatch 에이전트가 이 로그 데이터를 Amazon CloudWatch Logs에 보내도록 구성합니다.

Important

통합 CloudWatch 에이전트가 Amazon CloudWatch Logs에 로그 데이터를 전송하기 위한 도구로 SSM Agent를 대체했습니다. SSM Agent aws:cloudWatch 플러그인은 지원되지 않습니다. 로그 수집 프로세스에 통합된 CloudWatch 에이전트만 사용하는 것이 좋습니다. 자세한 정보는 다음 주제를 참조하세요.

- 통합 CloudWatch Logs로 노드 로그 전송(CloudWatch 에이전트)
- Windows 서버 노드 로그 수집을 CloudWatch 에이전트로 마이그레이션
- Amazon CloudWatch 사용 설명서의 CloudWatch 에이전트를 사용한 지표, 로그 및 추적 수집.

CloudWatch Logs를 사용하면 로그 데이터를 실시간으로 모니터링하고, 하나 이상의 지표 필터를 생성하여 로그 데이터를 검색 및 필터링하고, 필요 시 기록 데이터를 보관 및 가져올 수 있습니다. CloudWatch Logs에 대한 자세한 내용은 Amazon CloudWatch Logs User Guide를 참조하세요.

로그 데이터를 Amazon CloudWatch Logs로 보내도록 에이전트를 구성하면 다음과 같은 이점을 얻을 수 있습니다.

- 모든 SSM Agent 로그 파일에 대한 로그 파일 스토리지 중앙 집중화
- 파일에 더 빠르게 액세스하여 오류 확인
- 무기한 로그 파일 보존(구성 가능)
- 노드의 상태에 상관없이 로그 유지 관리 및 액세스 가능
- 지표, 경보 등의 다른 CloudWatch 기능에 액세스

Session Manager 활동 모니터링에 대한 자세한 내용은 세션 활동 감사 및 세션 활동 로깅 활성화 및 비활성화 섹션을 참조하세요.

Windows 서버 노드 로그 수집을 CloudWatch 에이전트로 마이그레이션

지원되는 Windows Server 노드에서 SSM Agent를 사용하여 SSM Agent 로그 파일을 Amazon CloudWatch Logs로 전송하고 있는 경우 Systems Manager를 사용하면 로그 수집 도구를 SSM Agent에서 CloudWatch 에이전트로 마이그레이션하고 구성 설정을 마이그레이션할 수 있습니다.

CloudWatch 에이전트는 Windows Server의 32비트 버전에서 지원되지 않습니다.

Windows Server용 64비트 EC2 인스턴스의 경우 CloudWatch 에이전트로의 마이그레이션을 자동 또는 수동으로 수행할 수 있습니다. 온프레미스 서버 및 가상 머신의 경우, 수동으로만 프로세스를 수행해야 합니다.

> **Note**
>
> 마이그레이션 프로세스 중 CloudWatch로 전송된 데이터가 중단되거나 중복될 수 있습니다. 마이그레이션이 완료되면 지표 및 로그 데이터가 CloudWatch에 다시 정확하게 기록됩니다.

전체 플릿을 CloudWatch 에이전트에 마이그레이션하기 전에 제한된 수의 노드에 대해 마이그레이션을 테스트해 보는 것이 좋습니다. 마이그레이션한 후 SSM Agent를 사용하여 로그를 수집하려면 이를 다시 사용하도록 설정을 변경하면 됩니다.

> **Important**
>
> 다음과 같은 경우에는 이 주제에서 설명하는 단계를 사용하여 CloudWatch 에이전트에 마이그레이션을 수행할 수 없습니다.
>
> - SSM Agent에 대한 기존 구성이 여러 리전이 지정되어 있는 경우
> - SSM Agent에 대한 기존 구성에 여러 세트의 액세스/보안 키 자격 증명이 지정되어 있는 경우
>
> 이러한 경우 마이그레이션 프로세스 없이 SSM Agent의 로그 수집을 해제하고 CloudWatch 에이전트를 설치해야 합니다. 자세한 내용은 Amazon CloudWatch User Guide의 다음 주제를 참조하세요.
>
> - CloudWatch 에이전트 설치
> - 온프레미스 서버에 CloudWatch 에이전트 설치

시작하기 전 준비 사항

로그 수집을 위해 CloudWatch 에이전트로 마이그레이션을 시작하기 전에, 마이그레이션을 수행할 노드가 다음 요건을 충족하는지 확인합니다.

- OS가 Windows 서버 64비트 버전입니다.
- SSM Agent 2.2.93.0 이상이 노드에 설치되어 있습니다.

- SSM Agent가 노드에서 모니터링하도록 구성되어 있습니다.

주제

- CloudWatch 에이전트로 자동 마이그레이션
- CloudWatch 에이전트로 수동 마이그레이션

CloudWatch 에이전트로 자동 마이그레이션

Windows Server용 EC2 인스턴스에서만 AWS Systems Manager 콘솔 또는 AWS Command Line Interface(AWS CLI)를 사용하여 로그 수집 도구로 CloudWatch 에이전트에 자동으로 마이그레이션할 수 있습니다.

> **Note**
>
> AWS Systems Manager는 64비트 버전의 Windows에서만 로그 및 지표를 수집하기 위해 SSM Agent에서 통합 CloudWatch 에이전트로의 마이그레이션을 지원합니다. 다른 운영 체제에서 통합 CloudWatch 에이전트 설정에 대한 자세한 내용과 CloudWatch 에이전트 사용에 대한 전체 내용은 Amazon CloudWatch 사용 설명서의 CloudWatch 에이전트를 사용하여 Amazon EC2 인스턴스 및 온프레미스 서버로부터 지표 및 로그 수집을 참조하세요.
> 지원되는 다른 운영 체제에서 CloudWatch 에이전트를 사용할 수 있지만 Systems Manager를 사용하여 도구 마이그레이션을 수행할 수는 없습니다.

마이그레이션이 성공하면 CloudWatch에서 결과를 확인하여 원하는 지표, 로그 또는 Windows 이벤트 로그를 전송 받고 있는지 확인합니다. 결과에 만족하는 경우 선택적으로 Parameter Store에 CloudWatch 에이전트 구성 설정 저장를 수행할 수 있습니다. 마이그레이션에 실패하거나 원하는 결과가 아닌 경우 SSM Agent를 사용한 로그 수집으로 롤백을 시도할 수 있습니다.

> **Note**
>
> `{hostname}` 항목을 포함하는 소스 구성 파일을 마이그레이션하려는 경우 마이그레이션이 완료된 이후에 `{hostname}` 항목으로 인해 필드 값이 변경될 수 있다는 사실을 염두에 두어야 합니다. 예를 들어 다음 `"LogStream": "{hostname}"` 항목이 MyLogServer001 서버로 매핑된다고 가정합니다.
>
> ```
> {
> "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}"
     }
}
```

마이그레이션 후 이 항목은 도메인(예: ip-11-1-1-11.production)에 매핑됩니다. ExampleCompany.com. 로컬 호스트 이름 값을 유지하려면 `{hostname}` 대신 `{local_hostname}`을 지정합니다.

CloudWatch 에이전트로 자동으로 마이그레이션하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택한 후 명령 실행을 선택합니다.
3. Command 문서(Command document) 목록에서 `AmazonCloudWatch-MigrateCloudWatchAgent`를 선택합니다.
4. 상태에서 활성을 선택합니다.
5. Targets(대상) 섹션에서, 태그를 지정하거나, 수동으로 인스턴스나 엣지 디바이스를 선택하거나, 리소스 그룹을 지정하여 이 작업을 실행할 관리형 노드를 식별합니다.

   > **Tip**
   >
   > 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

6. Rate control(속도 제어)에서

   - Concurrency(동시성)에서 명령을 동시에 실행할 관리형 노드의 백분율 또는 개수를 지정합니다.

> **Note**
>
> 관리형 노드에 적용할 태그를 지정하거나, AWS 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상 수를 제한합니다.

- Error threshold(오류 임계값)에서, 명령이 노드의 개수 또는 백분율에서 실패한 후 다른 관리형 노드에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째 오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.

7. (선택 사항) Output options(출력 옵션)에서 명령 출력을 파일에 저장하려면 Write command output to an S3 bucket(S3 버킷에 명령 출력 쓰기) 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

> **Note**
>
> 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아니라 인스턴스에 할당된 인스턴스 프로파일(EC2 인스턴스용) 또는 IAM 서비스 역할(하이브리드 정품 인증 시스템)의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

8. SNS notifications(SNS 알림) 섹션에서, 명령 실행 상태에 대한 알림이 전송되도록 하려면 Enable SNS notifications(SNS 알림 활성화) 확인란을 선택합니다.

   Run Command에 대한 Amazon SNS 알림 구성에 대한 자세한 내용은 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 섹션을 참조하세요.

9. Run(실행)을 선택합니다.

CloudWatch 에이전트로 자동으로 마이그레이션하려면(AWS CLI)

- 다음 명령을 실행합니다.

```
aws ssm send-command --document-name AmazonCloudWatch-MigrateCloudWatchAgent --
targets Key=instanceids,Values=ID1,ID2,ID3
```

ID1, *ID2* 및 *ID3*은 i-02573cafcfEXAMPLE와 같이 업데이트하려는 노드의 ID를 나타냅니다.

CloudWatch 에이전트로 수동 마이그레이션

온프레미스 Windows Server 노드 또는 Windows Server용 EC2 인스턴스에서 로그 수집을 Amazon CloudWatch 에이전트로 수동 마이그레이션하려면 다음 단계를 따릅니다.

> **Note**
>
> {hostname} 항목을 포함하는 소스 구성 파일을 마이그레이션하려는 경우 마이그레이션이 완료된 이후에 {hostname} 항목으로 인해 필드 값이 변경될 수 있다는 사실을 염두에 두어야 합니다. 예를 들어 다음 "LogStream": "{hostname}" 항목이 MyLogServer001 서버로 매핑된다고 가정합니다.
>
> ```
> {
> "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}"
>     }
> }
> ```
>
> 마이그레이션 후 이 항목은 도메인(예: ip-11-1-1-11.production.ExampleCompany.com)에 매핑됩니다. 로컬 호스트 이름 값을 유지하려면 {hostname} 대신 {local_hostname}을 지정합니다.

1단계: CloudWatch 에이전트를 설치하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택한 후 명령 실행을 선택합니다.
3. Command 문서(Command document) 목록에서 AWS-ConfigureAWSPackage를 선택합니다.
4. 작업(Action)에서 Install을(를) 선택합니다.

5. 이름에 **AmazonCloudWatchAgent**를 입력합니다.
6. 버전(Version)에 **latest**를 입력합니다(기본적으로 아직 제공되지 않은 경우).
7. Targets(대상) 섹션에서, 태그를 지정하거나, 수동으로 인스턴스나 엣지 디바이스를 선택하거나, 리소스 그룹을 지정하여 이 작업을 실행할 관리형 노드를 식별합니다.

> **Tip**
>
> 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

8. Rate control(속도 제어)에서

   - Concurrency(동시성)에서 명령을 동시에 실행할 관리형 노드의 백분율 또는 개수를 지정합니다.

     > **Note**
     >
     > 관리형 노드에 적용할 태그를 지정하거나, AWS 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상 수를 제한합니다.

   - Error threshold(오류 임계값)에서, 명령이 노드의 개수 또는 백분율에서 실패한 후 다른 관리형 노드에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째 오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.

9. (선택 사항) Output options(출력 옵션)에서 명령 출력을 파일에 저장하려면 Write command output to an S3 bucket(S3 버킷에 명령 출력 쓰기) 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

> **Note**
>
> 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아니라 인스턴스에 할당된 인스턴스 프로파일(EC2 인스턴스용) 또는 IAM 서비스 역할(하이브리드 정품 인증 시스템)의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

10. SNS notifications(SNS 알림) 섹션에서, 명령 실행 상태에 대한 알림이 전송되도록 하려면 Enable SNS notifications(SNS 알림 활성화) 확인란을 선택합니다.

    Run Command에 대한 Amazon SNS 알림 구성에 대한 자세한 내용은 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 섹션을 참조하세요.

11. Run(실행)을 선택합니다.

2단계: 구성 데이터 JSON 형식 업데이트

- CloudWatch 에이전트에 대한 기존 구성 설정의 JSON 형식을 업데이트하려면 AWS Systems Manager의 기능인 Run Command를 사용하거나 RDP 연결로 노드에 직접 로그인하여 다음 Windows PowerShell 명령을 노드에 한 번에 하나씩 실행합니다.

```
cd ${Env:ProgramFiles}\\Amazon\\AmazonCloudWatchAgent
```

```
.\\amazon-cloudwatch-agent-config-wizard.exe --isNonInteractiveWindowsMigration
```

  *{Env:ProgramFiles}*는 CloudWatch 에이전트가 포함되어 있는 Amazon 디렉터리를 찾을 수 있는 위치를 나타내며, 위치는 주로 `C:\Program Files`입니다.

3단계: CloudWatch 에이전트를 구성하고 시작하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택한 후 명령 실행을 선택합니다.
3. Command 문서(Command document) 목록에서 `AWS-RunPowerShellScript`를 선택합니다.
4. 명령(Commands)에 다음 두 명령을 입력합니다.

```
cd ${Env:ProgramFiles}\Amazon\AmazonCloudWatchAgent
```

```
.\amazon-cloudwatch-agent-ctl.ps1 -a fetch-config -m ec2 -c file:config.json -s
```

   *{Env:ProgramFiles}*는 CloudWatch 에이전트가 포함되어 있는 Amazon 디렉터리를 찾을 수 있는 위치를 나타내며, 위치는 주로 `C:\Program Files`입니다.

5. Targets(대상) 섹션에서, 태그를 지정하거나, 수동으로 인스턴스나 엣지 디바이스를 선택하거나, 리소스 그룹을 지정하여 이 작업을 실행할 관리형 노드를 식별합니다.

> **Tip**
>
> 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

6. Rate control(속도 제어)에서

   - Concurrency(동시성)에서 명령을 동시에 실행할 관리형 노드의 백분율 또는 개수를 지정합니다.

     > **Note**
     >
     > 관리형 노드에 적용할 태그를 지정하거나, AWS 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상 수를 제한합니다.

   - Error threshold(오류 임계값)에서, 명령이 노드의 개수 또는 백분율에서 실패한 후 다른 관리형 노드에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째 오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.

7. (선택 사항) Output options(출력 옵션)에서 명령 출력을 파일에 저장하려면 Write command output to an S3 bucket(S3 버킷에 명령 출력 쓰기) 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

   > **Note**
   >
   > 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아니라 인스턴스에 할당된 인스턴스 프로파일(EC2 인스턴스용) 또는 IAM 서비스 역할(하이브리드 정품 인증 시스템)의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

8. SNS notifications(SNS 알림) 섹션에서, 명령 실행 상태에 대한 알림이 전송되도록 하려면 Enable SNS notifications(SNS 알림 활성화) 확인란을 선택합니다.

Run Command에 대한 Amazon SNS 알림 구성에 대한 자세한 내용은 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 섹션을 참조하세요.

9. Run(실행)을 선택합니다.

4단계: SSM Agent에서 로그 수집을 해제하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택한 후 명령 실행을 선택합니다.
3. Command 문서(Command document) 목록에서 `AWS-ConfigureCloudWatch`를 선택합니다.
4. 상태(Status)에서 사용 중지됨(Disabled)을 선택합니다.
5. Targets(대상) 섹션에서, 태그를 지정하거나, 수동으로 인스턴스나 엣지 디바이스를 선택하거나, 리소스 그룹을 지정하여 이 작업을 실행할 관리형 노드를 식별합니다.

> **Tip**
>
> 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

6. 상태(Status)에서 `Disabled`를 선택합니다.
7. Rate control(속도 제어)에서

   - Concurrency(동시성)에서 명령을 동시에 실행할 관리형 노드의 백분율 또는 개수를 지정합니다.

     > **Note**
     >
     > 관리형 노드에 적용할 태그를 지정하거나, AWS 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상 수를 제한합니다.

   - Error threshold(오류 임계값)에서, 명령이 노드의 개수 또는 백분율에서 실패한 후 다른 관리형 노드에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째 오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.

8. (선택 사항) Output options(출력 옵션)에서 명령 출력을 파일에 저장하려면 Write command output to an S3 bucket(S3 버킷에 명령 출력 쓰기) 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

   > **Note**
   >
   > 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아니라 인스턴스에 할당된 인스턴스 프로파일(EC2 인스턴스용) 또는 IAM 서비스 역할(하이브리드 정품 인증 시스템)의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

9. SNS notifications(SNS 알림) 섹션에서, 명령 실행 상태에 대한 알림이 전송되도록 하려면 Enable SNS notifications(SNS 알림 활성화) 확인란을 선택합니다.

   Run Command에 대한 Amazon SNS 알림 구성에 대한 자세한 내용은 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 섹션을 참조하세요.

10. Run(실행)을 선택합니다.

    이러한 단계를 수행했으면 CloudWatch에서 결과를 확인하여 원하는 지표, 로그 또는 Windows 이벤트 로그를 전송 받고 있는지 확인합니다. 결과에 만족하는 경우 선택적으로 Parameter Store에 CloudWatch 에이전트 구성 설정 저장를 수행할 수 있습니다. 마이그레이션에 실패하거나 원하는 결과가 아닌 경우 SSM Agent를 사용한 로그 수집으로 롤백을 수행할 수 있습니다.

Parameter Store에 CloudWatch 에이전트 구성 설정 저장

Parameter Store에 CloudWatch 에이전트 구성 파일의 내용을 저장할 수 있습니다. 이러한 구성 데이터를 파라미터에 유지하면 여러 노드가 그로부터 구성 설정을 가져올 수 있으며, 사용자가 노드에 구성 파일을 생성하거나 수동으로 업데이트하지 않아도 됩니다. 예를 들어 Run Command를 사용하여 여러 노드의 구성 파일에 파라미터 내용을 쓰거나 AWS Systems Manager의 기능인 State Manager을 사용하여 노드 플릿에 대해 CloudWatch 에이전트 구성 설정의 구성 편차를 방지할 수 있습니다.

CloudWatch 에이전트 구성 마법사를 실행하면 마법사가 Parameter Store에 새로운 파라미터로 구성 설정을 저장하도록 선택할 수 있습니다. CloudWatch 에이전트 구성 마법사를 실행하는 방법에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 마법사로 CloudWatch 에이전트 구성 파일 생성을 참조하세요.

마법사를 실행했지만 설정을 파라미터로 저장하도록 하는 옵션을 선택하지 않았거나 CloudWatch 에이전트 구성 파일을 수동으로 생성한 경우 다음 파일에 노드의 파라미터로 저장할 데이터를 가져올 수 있습니다.

```
${Env:ProgramFiles}\Amazon\AmazonCloudWatchAgent\config.json
```

*{Env:ProgramFiles}*는 CloudWatch 에이전트가 포함되어 있는 Amazon 디렉터리를 찾을 수 있는 위치를 나타내며, 위치는 주로 `C:\Program Files`입니다.

노드 자체가 아닌 다른 위치의 이 파일에 JSON 백업을 유지하는 것이 좋습니다.

파라미터 생성에 대한 자세한 내용은 Systems Manager 파라미터 생성 섹션을 참조하세요.

CloudWatch 에이전트에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 CloudWatch 에이전트를 사용하여 Amazon EC2 인스턴스 및 온프레미스 서버로부터 지표 및 로그 수집을 참조하세요.

SSM Agent를 사용한 로그 수집으로 롤백

SSM Agent를 사용하여 로그를 수집하도록 다시 되돌리려면 다음 단계를 따르십시오.

1단계: SSM Agent로부터 구성 데이터 가져오기

1. SSM Agent를 사용하여 로그를 수집하도록 되돌리려는 노드에서 SSM Agent 구성 파일의 내용을 찾습니다. 이 JSON 파일은 일반적으로 다음 위치에서 찾을 수 있습니다.

   ```
   ${Env:ProgramFiles}\\Amazon\\SSM\\Plugins\\awsCloudWatch\
   \AWS.EC2.Windows.CloudWatch.json
   ```

   *{Env:ProgramFiles}*는 `Amazon` 디렉터리를 찾을 수 있는 위치를 나타내며, 위치는 주로 `C:\Program Files`입니다.
2. 추후 단계에서 사용할 수 있도록 이 데이터를 텍스트 파일에 복사합니다.

   노드 자체가 아닌 다른 위치에 JSON 백업을 저장해 두는 것이 좋습니다.

2단계: CloudWatch 에이전트를 제거하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택한 후 명령 실행을 선택합니다.
3. Command 문서(Command document) 목록에서 `AWS-ConfigureAWSPackage`를 선택합니다.

4. 작업(Action)에서 제거(Uninstall)를 선택합니다.
5. 이름에 **AmazonCloudWatchAgent**를 입력합니다.
6. Targets(대상) 섹션에서, 태그를 지정하거나, 수동으로 인스턴스나 엣지 디바이스를 선택하거나, 리소스 그룹을 지정하여 이 작업을 실행할 관리형 노드를 식별합니다.

> **Tip**
>
> 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

7. Rate control(속도 제어)에서

   - Concurrency(동시성)에서 명령을 동시에 실행할 관리형 노드의 백분율 또는 개수를 지정합니다.

   > **Note**
   >
   > 관리형 노드에 적용할 태그를 지정하거나, AWS 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상 수를 제한합니다.

   - Error threshold(오류 임계값)에서, 명령이 노드의 개수 또는 백분율에서 실패한 후 다른 관리형 노드에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째 오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.

8. (선택 사항) Output options(출력 옵션)에서 명령 출력을 파일에 저장하려면 Write command output to an S3 bucket(S3 버킷에 명령 출력 쓰기) 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

> **Note**
>
> 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아니라 인스턴스에 할당된 인스턴스 프로파일(EC2 인스턴스용) 또는 IAM 서비스 역할(하이브리드 정품 인증 시스템)의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

9. SNS notifications(SNS 알림) 섹션에서, 명령 실행 상태에 대한 알림이 전송되도록 하려면 Enable SNS notifications(SNS 알림 활성화) 확인란을 선택합니다.

   Run Command에 대한 Amazon SNS 알림 구성에 대한 자세한 내용은 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 섹션을 참조하세요.

10. Run(실행)을 선택합니다.

3단계: SSM Agent에 로그 수집을 다시 설정하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택한 후 명령 실행을 선택합니다.
3. Command 문서(Command document) 목록에서 `AWS-ConfigureCloudWatch`를 선택합니다.
4. 상태(Status)에서 `Enabled`를 선택합니다.
5. 속성(Properties) 상자에 저장한 이전 구성 데이터의 내용을 텍스트 파일에 붙여 넣습니다.
6. Targets(대상) 섹션에서, 태그를 지정하거나, 수동으로 인스턴스나 엣지 디바이스를 선택하거나, 리소스 그룹을 지정하여 이 작업을 실행할 관리형 노드를 식별합니다.

   > **Tip**
   >
   > 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

7. Rate control(속도 제어)에서

   - Concurrency(동시성)에서 명령을 동시에 실행할 관리형 노드의 백분율 또는 개수를 지정합니다.

     > **Note**
     >
     > 관리형 노드에 적용할 태그를 지정하거나, AWS 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상 수를 제한합니다.

   - Error threshold(오류 임계값)에서, 명령이 노드의 개수 또는 백분율에서 실패한 후 다른 관리형 노드에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째 오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.

8. (선택 사항) Output options(출력 옵션)에서 명령 출력을 파일에 저장하려면 Write command output to an S3 bucket(S3 버킷에 명령 출력 쓰기) 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

   > **Note**
   >
   > 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아니라 인스턴스에 할당된 인스턴스 프로파일(EC2 인스턴스용) 또는 IAM 서비스 역할(하이브리드 정품 인증 시스템)의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

9. SNS notifications(SNS 알림) 섹션에서, 명령 실행 상태에 대한 알림이 전송되도록 하려면 Enable SNS notifications(SNS 알림 활성화) 확인란을 선택합니다.

   Run Command에 대한 Amazon SNS 알림 구성에 대한 자세한 내용은 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 섹션을 참조하세요.

10. Run(실행)을 선택합니다.

CloudWatch Logs에 SSM Agent 로그 전송

AWS Systems Manager Agent(SSM Agent)는 EC2 인스턴스와 엣지 디바이스, 온프레미스 서버 및 Systems Manager용으로 구성된 가상 머신에서 실행되는 Amazon 소프트웨어입니다. SSM Agent는 클라우드에서 Systems Manager 서비스의 요청을 처리하고 요청에 지정된 대로 머신을 구성합니다. SSM Agent에 대한 자세한 내용은 SSM Agent 작업 섹션을 참조하세요.

또한 아래 단계에 따라 로그 데이터를 Amazon CloudWatch Logs에 보내도록 SSM Agent를 구성할 수 있습니다.

시작하기 전 준비 사항

CloudWatch Logs의 로그 그룹을 생성합니다. 자세한 내용은 Amazon CloudWatch Logs 사용 설명서의 CloudWatch Logs 시작하기 섹션을 참조하세요.

CloudWatch에 로그를 전송하도록 SSM Agent를 구성하려면

1. 노드에 로그인하여 다음 파일을 찾습니다.

Linux

대부분의 Linux 노드 유형: `/etc/amazon/ssm/seelog.xml.template`.

Ubuntu Server 20.10 STR 및 20.04, 18.04 및 16.04 LTS: `/snap/amazon-ssm-agent/current/seelog.xml.template`

macOS

`/opt/aws/ssm/seelog.xml.template`

Windows

`%ProgramFiles%\Amazon\SSM\seelog.xml.template`

2. 파일 이름을 `seelog.xml.template`에서 `seelog.xml`로 변경합니다.

> **Note**
>
> Ubuntu Server 20.10 STR 및 20.04, 18.04 및 16.04 LTS에서 파일 `seelog.xml`은 `/etc/amazon/ssm/` 디렉터리에 생성되어야 합니다. 다음 명령을 실행하여 이 디렉터리와 파일을 생성할 수 있습니다.
>
> ```
> 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. 텍스트 편집기로 `seelog.xml` 파일을 열고 다음 섹션을 찾습니다.

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. 파일을 편집하고 닫기 </filter> 태그 뒤에 사용자 정의 이름 요소를 추가합니다. 다음 예제에서는 `cloudwatch_receiver`로 지정된 사용자 정의 이름입니다.

   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. 변경 사항을 저장한 후 SSM Agent 또는 노드를 다시 시작합니다.
6. https://console.aws.amazon.com/cloudwatch/에서 CloudWatch 콘솔을 엽니다.
7. 탐색 창에서 로그 그룹을 선택한 다음, 로그 그룹의 이름을 선택합니다.

> **Tip**
>
> SSM Agent 로그 파일 데이터에 대한 로그 스트림은 노드 ID별로 구성됩니다.

변경 요청 이벤트 모니터링

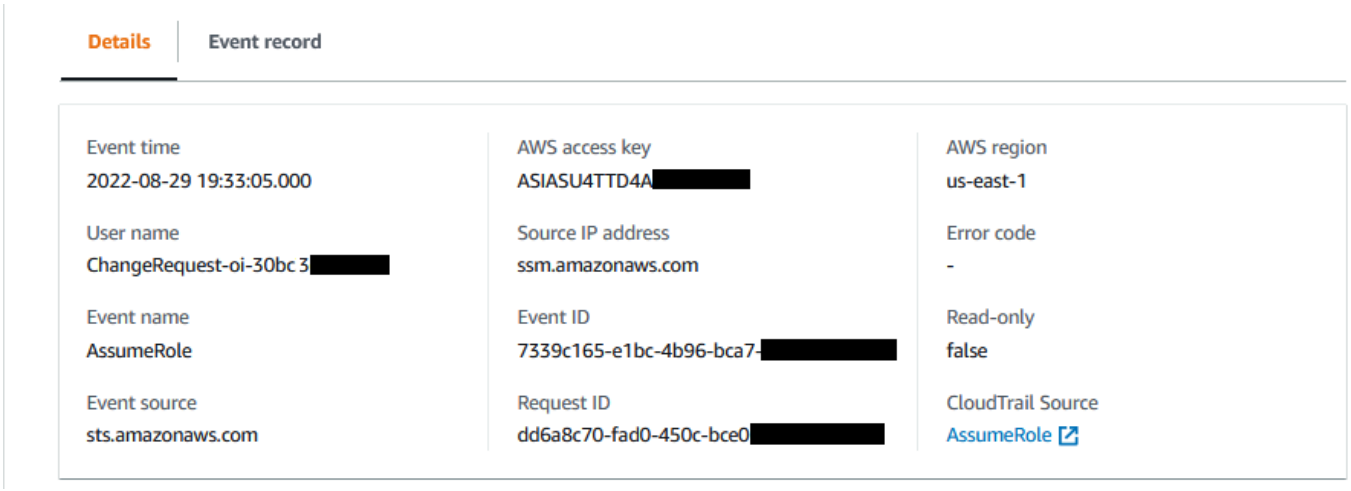
AWS CloudTrail Lake와의 통합을 활성화하고 이벤트 데이터 스토어를 생성한 후에는 계정 또는 조직에서 실행되는 변경 요청에 대한 감사 가능한 세부 정보를 볼 수 있습니다. 여기에는 다음과 같은 세부 정보가 포함됩니다.

- 변경 요청을 시작한 사용자의 ID
- 변경이 이루어진 AWS 리전
- 요청의 소스 IP 주소
- 요청에 사용된 AWS 액세스 키
- 변경 요청에 대해 실행된 API 작업
- 해당 작업에 대해 포함된 요청 파라미터
- 프로세스 실행 중에 업데이트된 리소스

다음은 AWS CloudTrail Lake에서 이벤트 데이터 스토어를 생성한 후 볼 수 있는 변경 요청과 관련한 이벤트 세부 정보의 샘플입니다.

Details

다음 이미지는 Details(세부 정보) 탭에서 제공되는 변경 요청에 대한 개략적인 정보를 보여줍니다. 이러한 세부 정보에는 변경 요청 작업이 시작된 시간, 변경 요청을 시작한 사용자의 ID, 영향을 받은 AWS 리전, 요청과 관련한 이벤트 ID 및 요청 ID 등의 정보가 포함됩니다.



Event record

다음 이미지는 변경 요청 이벤트에 대해 CloudTrail Lake에서 제공하는 JSON 콘텐츠의 구조를 보여줍니다. 이 데이터는 변경 요청의 Event record(이벤트 레코드) 탭에서 제공됩니다.

Important

조직에서 Change Manager를 사용하는 경우 Change Manager의 관리 계정 또는 위임된 관리자 계정에 로그인한 상태에서 다음 절차를 완료할 수 있습니다.

단, 위임된 관리자 계정을 사용하여 이 단계를 완료하려면 CloudTrail과 Change Manager에 대해 동일한 위임된 관리자 계정을 지정해야 합니다.
Change Manager의 관리 계정에 로그인하면 CloudTrail Settings(설정) 페이지에서 CloudTrail의 위임된 관리자 계정을 추가하거나 변경할 수 있습니다. 위임된 관리자 계정에서 전체 조직에 사용할 이벤트 데이터 스토어를 생성하려면 먼저 이 작업을 수행해야 합니다.

Change Manager에서 CloudTrail Lake 이벤트 추적을 활성화하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Change Manager를 선택합니다.
3. [요청(Requests)] 탭을 선택합니다.
4. 기존 변경 요청을 선택한 다음 Associated events(관련 이벤트) 탭을 선택합니다.
5. Enable CloudTrail Lake(CloudTrail Lake 활성화)를 선택합니다.
6. AWS CloudTrail 사용 설명서의 CloudTrail 이벤트에 대한 이벤트 데이터 스토어 생성에 있는 단계를 따릅니다.

   변경 요청에 대한 이벤트 데이터가 저장되도록 하려면 절차를 완료할 때 다음을 선택합니다.

   - 이벤트 유형의 경우 기본 AWS 이벤트와 CloudTrail 이벤트를 선택한 상태로 둡니다.
   - 조직에서 Change Manager를 사용하는 경우 조직의 모든 계정에 대해 활성화를 선택합니다.
   - 관리 이벤트의 경우 쓰기 확인란을 선택 취소하지 않습니다.

   이벤트 데이터 스토어를 생성할 때 선택하는 다른 옵션은 변경 요청에 대한 이벤트 데이터를 저장하는 데 영향을 미치지 않습니다.

자동화 모니터링

지표는 Amazon CloudWatch의 기본 개념입니다. 지표는 CloudWatch에 게시된 시간 순서별 데이터 요소 집합을 나타냅니다. 지표는 모니터링할 변수로, 데이터 요소는 시간에 따른 변수의 값을 나타내는 것으로 간주합니다.

Automation은 AWS Systems Manager의 기능입니다. Systems Manager는 Automation 사용에 대한 지표를 CloudWatch에 게시합니다. 이를 통해 해당 지표를 기반으로 경보를 설정할 수 있습니다.

CloudWatch 콘솔에서 Automation 지표 보기

1. https://console.aws.amazon.com/cloudwatch/에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 지표를 선택합니다.
3. SSM을 선택합니다.
4. 지표(Metrics) 탭에서 사용량(Usage)을 선택한 다음 AWS 리소스별(By Resource)을 선택합니다.
5. 지표 목록 근처의 검색 상자에 SSM을 입력합니다.

AWS CLI를 사용하여 Automation 지표 보기

명령 프롬프트를 열고 다음 명령을 사용합니다.

```
aws cloudwatch list-metrics \
    --namespace "AWS/Usage"
```

Automation 지표

Systems Manager는 다음 Automation 지표를 CloudWatch로 전송합니다.

지표	설명
`ConcurrentAutomationUsage`	현재 AWS 계정과 AWS 리전에서 동시에 실행 중인 자동화 수입니다.
`QueuedAutomationUsage`	시작되지 않고 `Pending` 상태이며 대기 중인 자동화의 수입니다.

CloudWatch 지표 작업에 대한 자세한 내용은 Amazon CloudWatch User Guide의 다음 주제를 참조하세요.

- 지표
- Amazon CloudWatch 지표 사용
- Amazon CloudWatch 경보 사용

Amazon CloudWatch를 사용한 Run Command 지표 모니터링

지표는 Amazon CloudWatch의 기본 개념입니다. 지표는 CloudWatch에 게시된 시간 순서별 데이터 요소 집합을 나타냅니다. 지표는 모니터링할 변수로, 데이터 요소는 시간에 따른 변수의 값을 나타내는 것으로 간주합니다.

AWS Systems Manager에서는 Run Command 명령 상태에 대한 지표를 CloudWatch에 게시하여 해당 지표를 기반으로 경보를 설정할 수 있습니다. Run Command는 AWS Systems Manager의 기능입니다. 이러한 통계는 장기간 동안 기록되므로 기록 정보에 액세스하여 AWS 계정에서 실행되는 명령의 성공률을 더 잘 파악할 수 있습니다.

지표를 추적할 수 있는 명령의 터미널 상태 값에는 `Success`, `Failed` 및 `Delivery Timed Out`이 있습니다. 예를 들어 1시간마다 실행되는 SSM Command 문서 집합의 경우 해당 시간 동안 `Success`의 상태가 보고되지 않을 때 알림을 보내도록 경보를 구성할 수 있습니다. 명령 상태 값에 대한 자세한 내용은 명령 상태 이해 섹션을 참조하세요.

CloudWatch 콘솔에서 지표 보기

1. https://console.aws.amazon.com/cloudwatch/에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 지표를 선택합니다.
3. AWS 서비스별 경보(Alarms by service) 지역의 서비스(Services)에서 SSM-Run Command을 선택합니다.

AWS CLI를 사용하여 지표를 보려면

명령 프롬프트를 열고 다음 명령을 사용합니다.

```
aws cloudwatch list-metrics --namespace "AWS/SSM-RunCommand"
```

사용 가능한 측정치를 모두 나열하려면 다음 명령을 사용합니다.

```
aws cloudwatch list-metrics
```

Systems Manager Run Command 지표 및 차원

Systems Manager는 1분마다 한 번씩 Run Command 명령 지표를 CloudWatch에 보냅니다.

Systems Manager는 다음 명령 지표를 CloudWatch로 전송합니다.

> **Note**
>
> 이러한 지표는 Count를 단위로 사용하므로 Sum과 SampleCount는 가장 유용한 통계가 아닙니다.

지표	설명
CommandsDeliveryTimedOut	터미널 상태가 Delivery Timed Out인 명령 수입니다.
CommandsFailed	터미널 상태가 Failed인 명령 수입니다.
CommandsSucceeded	터미널 상태가 Success인 명령 수입니다.

CloudWatch 지표 작업에 대한 자세한 내용은 Amazon CloudWatch User Guide의 다음 주제를 참조하세요.

- 지표
- Amazon CloudWatch 지표 사용
- Amazon CloudWatch 경보 사용

AWS CloudTrail을 사용하여 AWS Systems ManagerAPI 호출을 로깅

AWS Systems Manager는 사용자, 역할 또는 AWS 서비스가 수행한 작업의 레코드를 제공하는 서비스인 AWS CloudTrail과 통합됩니다. CloudTrail은 Systems Manager에 대한 API 호출을 이벤트로 캡처합니다. 캡처되는 호출에는 Systems Manager 콘솔에서 수행한 호출과 Systems Manager API 작업에 대한 코드 호출이 포함됩니다. CloudTrail에서 수집한 정보를 사용하여 Systems Manager에 수행된 요청, 요청이 수행된 IP 주소, 요청이 수행된 시간, 추가 세부 정보를 확인할 수 있습니다.

모든 이벤트 또는 로그 항목에는 누가 요청했는지 확인하는 데 도움이 되는 정보가 포함되어 있습니다.

- AWS 계정 루트 사용자
- AWS Identity and Access Management(IAM) 역할 또는 페더레이션 사용자에 대한 임시 보안 자격 증명

- IAM 사용자의 장기 보안 보안 인증.
- IAM Identity Center 사용자를 대신한 요청입니다.
- 또 다른 AWS 서비스입니다.

자세한 설명은 CloudTrail userIdentity 요소를 참조하십시오.

계정을 생성할 때 AWS 계정에서 CloudTrail이 활성 상태이며, CloudTrail 이벤트 기록에 자동으로 액세스할 수 있습니다. CloudTrail 이벤트 기록은 지난 90일 간 AWS 리전의 관리 이벤트에 대해 보기, 검색 및 다운로드가 가능하고, 수정이 불가능한 레코드를 제공합니다. 자세한 설명은 AWS CloudTrail 사용 설명서의 CloudTrail 이벤트 기록 작업을 참조하세요. Event history(이벤트 기록) 보기는 CloudTrail 요금이 부과되지 않습니다.

지난 90일 동안 AWS 계정에서 진행 중인 이벤트 기록을 보려면 추적 또는 CloudTrail Lake 이벤트 데이터 스토어를 생성합니다.

CloudTrail 추적

CloudTrail은 추적을 사용하여 Amazon S3 버킷으로 로그 파일을 전송할 수 있습니다. AWS Management Console을 사용하여 만든 추적은 모두 다중 리전입니다. AWS CLI를 사용하여 단일 리전 또는 다중 리전 추적을 생성할 수 있습니다. 계정의 모든 AWS 리전에서 활동을 캡처하므로, 다중 리전 추적 생성이 권장됩니다. 단일 리전 추적을 생성하는 경우 추적의 AWS 리전에 로깅된 이벤트만 볼 수 있습니다. 추적에 대한 자세한 내용은 AWS CloudTrail 사용 설명서의 Creating a trail for your AWS 계정 및 Creating a trail for an organization을 참조하세요.

CloudTrail에서 추적을 생성하여 진행 중인 관리 이벤트의 사본 하나를 Amazon S3 버킷으로 무료로 전송할 수는 있지만, Amazon S3 스토리지 요금이 부과됩니다. CloudTrail 요금에 대한 자세한 내용은 AWS CloudTrail 요금을 참조하세요. Amazon S3 요금에 대한 자세한 내용은 Amazon S3 요금을 참조하세요.

CloudTrail Lake 이벤트 데이터 스토어

CloudTrail Lake를 사용하면 이벤트에 대해 SQL 기반 쿼리를 실행할 수 있습니다. CloudTrail Lake는 행 기반 JSON 형식의 기존 이벤트를 Apache ORC 형식으로 변환합니다. ORC는 빠른 데이터 검색에 최적화된 열 기반 스토리지 형식입니다. 이벤트는 이벤트 데이터 스토어로 집계되며, 이벤트 데이터 스토어는 고급 이벤트 선택기를 적용하여 선택한 기준을 기반으로 하는 변경 불가능한 이벤트 컬렉션입니다. 이벤트 데이터 스토어에 적용하는 선택기는 어떤 이벤트가 지속되고 쿼리할 수 있는지 제어합니다. CloudTrail Lake에 대한 자세한 내용은 AWS CloudTrail 사용 설명서의 Working with AWS CloudTrail Lake를 참조하세요.

CloudTrail Lake 이벤트 데이터 스토어 및 쿼리에는 비용이 발생합니다. 이벤트 데이터 스토어를 생성할 때 이벤트 데이터 스토어에 사용할 요금 옵션을 선택합니다. 요금 옵션에 따라 이벤트 모으기 및 저장 비용과 이벤트 데이터 스토어의 기본 및 최대 보존 기간이 결정됩니다. CloudTrail 요금에 대한 자세한 내용은 AWS CloudTrail 요금을 참조하세요.

CloudTrail의 Systems Manager 데이터 이벤트

데이터 이벤트는 리소스 기반 또는 리소스에서 수행된 리소스 작업에 대한 정보를 제공합니다(예: 컨트롤 채널 생성 또는 열기). 이를 데이터 영역 작업이라고도 합니다. 데이터 이벤트가 대량 활동인 경우도 있습니다. 기본적으로 CloudTrail은 데이터 이벤트를 로깅하지 않습니다. CloudTrail 이벤트 기록은 데이터 이벤트를 기록하지 않습니다.

데이터 이벤트에는 추가 요금이 적용됩니다. CloudTrail 요금에 대한 자세한 내용은 AWS CloudTrail 요금을 참조하세요.

CloudTrail 콘솔, AWS CLI 또는 CloudTrail API 작업을 사용하여 Systems Manager 리소스 유형에 대한 데이터 이벤트를 로깅할 수 있습니다. 데이터 이벤트를 로깅하는 방법에 대한 자세한 내용은 AWS CloudTrail 사용 설명서의 Logging data events with the AWS Management Console 및 Logging data events with the AWS Command Line Interface를 참조하세요.

다음 표에는 데이터 이벤트를 로깅할 수 있는 Systems Manager 리소스 유형이 나열되어 있습니다. 데이터 이벤트 유형(콘솔) 열에는 CloudTrail 콘솔의 데이터 이벤트 유형 목록에서 선택할 값이 표시됩니다. resources.type 값 열에는 AWS CLI 또는 CloudTrail API를 사용하여 고급 이벤트 선택기를 구성할 때 지정하는 `resources.type` 값이 표시됩니다. CloudTrail에 로깅되는 데이터 API 열에는 리소스 유형에 대해 CloudTrail에 로깅된 API 호출이 표시됩니다.

데이터 이벤트 유형(콘솔)	resources.type 값	CloudTrail에 로깅되는 데이터 API
Systems Manager	`AWS::SSMMessages::ControlChannel`	• `CreateControlChannel` • `OpenControlChannel` 이러한 작업에 대한 자세한 내용은 서비스 승인 참조의 Amazon Message Gateway

데이터 이벤트 유형(콘솔)	resources.type 값	CloudTrail에 로깅되는 데이터 API
		Service에서 정의한 작업을 참조하세요.
Systems Manager 관리형 노드	`AWS::SSM::ManagedNode`	• `RequestManagedInstanceRoleToken` -이 이벤트는 Systems Manager에서 관리하는 노드에서 실행되는 Systems Manager Agent(SSM Agent)가 Systems Manager 자격 증명 서비스에 자격 증명을 요청할 때 생성됩니다.

`eventName`, `readOnly` 및 `resources.ARN` 필드를 필터링하여 중요한 이벤트만 로깅하도록 고급 이벤트 선택기를 구성할 수 있습니다. 이러한 필드에 대한 자세한 내용은 AWS CloudTrail API 참조의 AdvancedFieldSelector 섹션을 참조하세요.

CloudTrail의 Systems Manager 관리 이벤트

관리 이벤트는 AWS 계정의 리소스에 대해 수행되는 관리 작업에 대한 정보를 제공합니다. 이를 제어 영역 작업이라고도 합니다. 기본적으로 CloudTrail은 관리 이벤트를 로깅합니다.

Systems Manager는 CloudTrail에 모든 컨트롤 플레인 작업을 CloudTrail에 관리 이벤트로 기록합니다. Systems Manager API 작업은 AWS Systems Manager API 참조에 문서화됩니다. 예를 들어 `CreateMaintenanceWindows`, `PutInventory`, `SendCommand` 및 `StartSession` 작업을 호출하면 CloudTrail 로그 파일에 항목이 생성됩니다. Systems Manager API 호출을 모니터링하기 위해 CloudTrail을 설정하는 예는 Amazon EventBridge를 사용하여 세션 활동 모니터링(콘솔) 섹션을 참조하세요.

Systems Manager 이벤트 예제

이벤트는 모든 소스로부터의 단일 요청을 나타내며 요청된 API 작업, 작업 날짜와 시간, 요청 파라미터 등에 대한 정보가 들어 있습니다. CloudTrail 로그 파일은 퍼블릭 API 직접 호출의 주문 스택 추적이 아니므로 이벤트가 특정 순서로 표시되지 않습니다.

예:

- 관리 이벤트 예제
- 데이터 이벤트 예제

관리 이벤트 예제

예 1: **DeleteDocument**

다음 예제는 미국 동부(오하이오)(us-east-2)의 example-Document라는 문서에 대한 DeleteDocument 작업을 알리는 CloudTrail 이벤트를 보여줍니다.


```
{
    "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"
}
```

예 2: **StartConnection**

다음 예제는 미국 동부(오하이오) 리전(us-east-2)에서 Fleet Manager를 사용하여 RDP 연결을 시작하는 사용자의 CloudTrail 이벤트를 보여줍니다. 기본 API 작업은 `StartConnection`입니다.

```
{
    "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"
}
```

데이터 이벤트 예제

예 1: **CreateControlChannel**

다음 예제는 CreateControlChannel 작업을 시연하는 CloudTrail 이벤트를 보여줍니다.

```
{
  "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"
```

```
}
```

예 2: **RequestManagedInstanceRoleToken**

다음 예제는 RequestManagedInstanceRoleToken 작업을 시연하는 CloudTrail 이벤트를 보여줍니다.

```
{
    "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"
}
```


CloudTrail 레코드 콘텐츠에 대한 자세한 내용은 AWS CloudTrail 사용 설명서의 CloudTrail record contents를 참조하세요.

CloudWatch Logs로 Automation 작업 출력 로깅

AWS Systems Manager의 기능인 Automation이 Amazon CloudWatch Logs와 통합됩니다. 실행서에 있는 `aws:executeScript` 작업의 출력을 지정하는 로그 그룹으로 전송할 수 있습니다. Systems Manager는 `aws:executeScript` 작업을 사용하지 않는 문서에 대한 로그 그룹이나 로그 스트림을 생성하지 않습니다. 문서에서 `aws:executeScript`를 사용하는 경우 CloudWatch Logs로 전송되는 출력은 해당 작업에만 적용됩니다. 디버깅 및 문제 해결을 위해 CloudWatch Logs 로그 그룹에 저장된 `aws:executeScript` 작업 출력을 사용할 수 있습니다. 암호화된 로그 그룹을 선택하면 `aws:executeScript` 작업 출력도 암호화됩니다. `aws:executeScript` 작업의 로깅 출력은 계정 수준 설정입니다.

작업 출력을 Amazon 소유 런북에 대한 CloudWatch Logs로 전송하려면 자동화를 실행하는 사용자 또는 역할에 다음 작업에 대한 권한이 부여되어 있어야 합니다.

- `logs:CreateLogGroup`
- `logs:CreateLogStream`
- `logs:DescribeLogGroups`
- `logs:DescribeLogStreams`
- `logs:PutLogEvents`

자신이 소유한 런북의 경우, 런북을 실행하는 데 사용하는 IAM 서비스 역할(또는 AssumeRole)에 동일한 권한을 추가해야 합니다.

작업 출력을 CloudWatch Logs로 전송하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 왼쪽 탐색 창에서 Automation을 선택합니다.
3. 기본 설정 탭을 선택하고 편집을 선택합니다.
4. [Send output to CloudWatch Logs(CloudWatch Logs로 출력 전송)] 옆에 있는 확인란을 선택합니다.
5. (권장) [로그 데이터 암호화(Encrypt log data)] 옆에 있는 확인란을 선택합니다. 이 옵션을 설명하면 로그 데이터가 로그 그룹에 지정된 서버 측 암호화 키를 사용하여 암호화됩니다. CloudWatch Logs로 전송되는 로그 데이터를 암호화하지 않으려면 확인란을 선택 취소합니다. 로그 그룹에서 암호화가 허용되지 않는 경우에도 확인란의 선택을 취소합니다.
6. [CloudWatch Logs 로그 그룹(CloudWatch Logs log group)]에 대해 작업 출력을 전송할 AWS 계정의 기존 CloudWatch Logs 로그 그룹을 지정하려면 다음 중 하나를 선택합니다.
   - [기본 로그 그룹으로 출력 전송(Send output to the default log group)] - 기본 로그 그룹이 존재하지 않는 경우(`/aws/ssm/automation/executeScript`) Automation에서 자동으로 생성합니다.
   - [로그 그룹 목록에서 선택(Choose from a list of log groups)]: 작업 출력을 저장할 계정에 이미 생성된 로그 그룹을 선택합니다.
   - [로그 그룹 이름 입력(Enter a log group name)]: 작업 출력을 저장하기 위해 계정에 이미 생성된 로그 그룹의 이름을 텍스트 상자에 입력합니다.
7. Save(저장)를 선택합니다.

작업 출력을 CloudWatch Logs로 전송하려면(명령줄)

1. 원하는 명령줄 도구를 열고 다음 명령을 실행하여 작업 출력 대상을 업데이트합니다.

   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"
```

명령이 성공해도 결과는 없습니다.

2. 다음 명령을 실행하여 작업 출력을 전송할 로그 그룹을 지정합니다.

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"
```

명령이 성공해도 결과는 없습니다.

3. 다음 명령을 실행하여 현재 AWS 계정 및 AWS 리전의 Automation 작업 로깅 기본 설정에 대한 현재 서비스 설정을 확인합니다.

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"
```

명령은 다음과 같은 정보를 반환합니다.

```
{
    "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"
    }
}
```

Run Command에 대한 Amazon CloudWatch Logs 구성

AWS Systems Manager의 기능인 Run Command를 사용하여 명령을 전송할 때는 명령 출력을 전송할 위치를 지정할 수 있습니다. 기본적으로 Systems Manager는 명령 출력의 첫 24,000자만 반환합니다. 명령 출력의 전체 세부 정보를 보려면 Amazon Simple Storage Service(Amazon S3) 버킷을 지정할 수 있습니다. 또는 Amazon CloudWatch Logs를 지정할 수 있습니다. CloudWatch Logs를 지정할 경우 Run Command가 모든 명령 출력과 오류 로그를 CloudWatch Logs에 주기적으로 전송합니다. 거의 실시간으로 출력 로그를 모니터링하고, 특정 구문, 값 또는 패턴을 검색하며, 검색을 기반으로 경보를 생성할 수 있습니다.

AWS Identity and Access Management(IAM) 관리형 정책 `AmazonSSMManagedInstanceCore` 및 `CloudWatchAgentServerPolicy`를 사용하도록 관리형 노드를 구성한 경우 CloudWatch Logs로 출력을 보낼 추가 노드 구성이 필요하지 않습니다. 콘솔에서 명령을 전송하는 경우 이 옵션을 선택합니다. 혹은 AWS Command Line Interface (AWS CLI), AWS Tools for Windows PowerShell 또는 API 작업을 사용하는 경우 `cloud-watch-output-config` 섹션과 `CloudWatchOutputEnabled` 파라미터를 추가합니다. `cloud-watch-output-config` 섹션과 `CloudWatchOutputEnabled` 파라미터에 대해서는 나중에 이 주제 안에서 자세히 설명합니다.

EC2 인스턴스용 인스턴스 프로파일에 정책을 추가하는 것에 대한 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성을 참조하세요. 관리형 노드로 사용할 예정인 온프레미스 서버 및 가상 머신에 대한 서비스 역할에 정책을 추가하는 것에 대한 정보는 하이브리드 및 멀티클라우드 환경에서 Systems Manager에 필요한 IAM 서비스 역할 생성을 참조하세요.

노드에 사용자 정의 정책을 사용하는 경우라면 Systems Manager가 CloudWatch Logs에 출력 및 로그를 전송하는 것을 허용하도록 각 노드의 정책을 업데이트합니다. 사용자 지정 정책에 다음 정책 객체를 추가하십시오. IAM 정책 업데이트에 대한 자세한 내용은 IAM User Guide의 Editing IAM policies를 참조하세요.

```
{
    "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/*"
},
```

명령을 전송할 때 CloudWatch Logs 지정

AWS Management Console에서 명령을 보낼 때 CloudWatch Logs를 출력으로 지정하려면 [출력 옵션(Output options)] 섹션에서 [CloudWatch 출력(CloudWatch Output)]을 선택합니다. 선택에 따라 명령 출력을 전송할 CloudWatch Logs 그룹의 이름을 지정할 수 있습니다. 그룹 이름을 지정하지 않으면 Systems Manager에서 로그 그룹을 자동으로 생성합니다. 로그 그룹은 `/aws/ssm/`*`SystemsManagerDocumentName`* 명명 형식을 사용합니다.

AWS CLI를 사용하여 명령을 실행하는 경우 명령에서 `cloud-watch-output-config` 섹션을 지정합니다. 이 섹션에서 `CloudWatchOutputEnabled` 파라미터를 지정할 수 있고, 선택에 따라 `CloudWatchLogGroupName` 파라미터도 지정할 수 있습니다. 다음 예를 참고하세요

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}'
```

CloudWatch Logs에서 명령 출력 보기

명령 실행을 시작하자마자 Systems Manager는 거의 실시간으로 CloudWatch Logs로 출력을 보냅니다. CloudWatch Logs에서 출력은 다음 형식을 사용합니다.

`CommandID``/`*`InstanceID`*`/`*`PluginID`*`/stdout`

CommandID/*InstanceID*/*PluginID*/stderr

30초마다 혹은 버퍼가 200KB를 초과할 때, 어느 것에든 먼저 도달하면 실행에서 출력을 업로드합니다.

> **Note**
>
> 출력 데이터를 사용할 수 있을 때에만 로그 스트림이 생성됩니다. 예를 들어 실행에 대한 오류 데이터가 없는 경우 stderr 스트림이 생성되지 않습니다.

다음은 CloudWatch Logs에 표시되는 명령 출력의 예입니다.

```
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
```

Amazon EventBridge로 Systems Manager 이벤트 모니터링

Amazon EventBridge는 애플리케이션을 다양한 소스의 데이터와 연결할 수 있는 서버리스 이벤트 버스 서비스입니다. EventBridge는 자체 애플리케이션, 서비스형 소프트웨어(SaaS) 애플리케이션 및 AWS 서비스의 실시간 데이터 스트림을 제공한 다음, 해당 데이터를 AWS Lambda 등의 대상으로 라우팅합니다. 데이터를 전송할 대상을 결정하는 라우팅 규칙을 설정하여 모든 데이터 소스에 실시간으로 대응하는 애플리케이션 아키텍처를 구축할 수 있습니다. EventBridge를 사용하면 느슨하게 결합되고 분산되는 이벤트 기반 아키텍처를 구축할 수 있습니다.

이전에는 EventBridge를 Amazon CloudWatch Events라고 했습니다. EventBridge에는 SaaS 파트너 및 자체 애플리케이션에서 이벤트를 수신할 수 있는 새로운 기능이 포함되어 있습니다. 기존 CloudWatch Events 사용자는 새 EventBridge 콘솔과 CloudWatch 이벤트 콘솔에서 기존 기본 버스, 규칙 및 이벤트에 액세스할 수 있습니다. EventBridge는 동일한 CloudWatch Events API를 사용하므로 기존의 모든 CloudWatch Events API 사용량이 동일하게 유지됩니다.

EventBridge는 수십 개의 AWS 서비스에서 이벤트를 규칙에 추가하고 20개 이상의 AWS 서비스에서 대상을 추가할 수 있습니다.

EventBridge는 AWS Systems Manager 이벤트와 Systems Manager 대상을 모두 지원합니다.

지원되는 Systems Manager 이벤트 유형

다음은 EventBridge가 감지할 수 있는 여러 유형의 Systems Manager 이벤트입니다.

- 유지 관리 기간이 해제되어 있습니다.
- Automation 워크플로가 성공적으로 완료되었습니다. Automation은 AWS Systems Manager의 기능입니다.
- 관리형 노드가 패치 규정을 준수하지 않습니다.
- 파라미터 값을 업데이트 중입니다.

EventBridge는 다음 AWS Systems Manager 기능의 이벤트를 지원합니다.

- Automation(이벤트가 최선의 작업을 기반으로 발생.)
- Change Calendar(이벤트는 최선의 작업을 기반으로 발생됩니다.)
- 규정 준수
- Inventory(이벤트는 최선의 작업을 기반으로 발생됩니다.)
- Maintenance Windows(이벤트는 최선의 작업을 기반으로 발생됩니다.)
- Parameter Store(이벤트는 최선의 작업을 기반으로 발생됩니다.)
- Run Command(이벤트는 최선의 작업을 기반으로 발생됩니다.)
- State Manager(이벤트는 최선의 작업을 기반으로 발생됩니다.)

지원되는 Systems Manager 이벤트 유형에 대한 자세한 내용은 참조: Systems Manager용 Amazon EventBridge 이벤트 패턴 및 유형 및 Systems Manager에 대한 Amazon EventBridge 이벤트 예 섹션을 참조하세요.

지원되는 Systems Manager 대상 유형

EventBridge는 이벤트 규칙의 대상으로 다음 3가지 Systems Manager 기능을 지원합니다.

- Automation 워크플로 실행
- Run Command Command 문서 실행(최선의 작업을 기반으로 이벤트 방출)
- OpsCenter OpsItem 생성

이러한 대상을 사용할 수 있는 권장 방법은 샘플 시나리오: Amazon EventBridge 규칙의 Systems Manager 대상 섹션을 참조하세요.

EventBridge 시작 및 규칙 설정 방법에 대한 자세한 내용은 Amazon EventBridge 사용 설명서의 Amazon EventBridge 시작하기를 참조하세요. EventBridge 작업에 대한 전체 내용은 Amazon EventBridge User Guide를 참조하세요.

주제

Systems Manager 이벤트에 대해 EventBridge 구성

Amazon EventBridge를 사용하여 지원되는 AWS Systems Manager 상태 변경, 상태 변경 또는 기타 조건이 발생할 때 대상 이벤트를 수행할 수 있습니다. 상태가 달라질 때마다 또는 관심이 있는 상태로 변경될 때 실행되는 규칙을 만들면 됩니다.

다음 절차에서는 Systems Manager가 지정된 이벤트를 발생시킬 때 적용되는 EventBridge 규칙을 생성하는 일반적인 단계를 제공합니다. 이 사용 설명서에서 특정 시나리오를 다루는 절차에 대한 목록은 이 주제 끝부분의 추가 정보를 참조하세요.

> **Note**
>
> AWS 계정의 서비스가 이벤트를 출력하면 항상 계정의 기본 이벤트 버스로 이동합니다. 계정의 AWS 서비스에서 이벤트에 응답하는 규칙을 작성하려면 기본 이벤트 버스와 연결합니다. AWS 서비스의 이벤트를 찾는 사용자 정의 이벤트 버스에서 규칙을 생성할 수 있지만, 이 규칙은 크로스 계정 이벤트 제공을 통해 다른 계정에서 이러한 이벤트를 수신할 때만 적용됩니다. 자세한 내용은 Amazon EventBridge 사용 설명서의 AWS 계정 간 이벤트 전송 및 수신을 참조하세요.

Systems Manager 이벤트에 대해 EventBridge를 구성하려면

1. https://console.aws.amazon.com/events/에서 Amazon EventBridge 콘솔을 엽니다.
2. 탐색 창에서 규칙을 선택합니다.
3. 규칙 생성을 선택합니다.
4. 규칙에 대해 이름과 설명을 입력하십시오.

   규칙은 동일한 AWS 리전과 동일한 이벤트 버스의 다른 규칙과 동일한 이름을 가질 수 없습니다.
5. 이벤트 버스에서 이 규칙과 연결할 이벤트 버스를 선택합니다. 이 규칙이 자신의 AWS 계정에서 오는 일치하는 이벤트에 응답하도록 하려면 default(기본)를 선택합니다. 계정의 AWS 서비스이(가) 이벤트를 출력하면 항상 계정의 기본 이벤트 버스로 이동합니다.

6. 규칙 유형에서 이벤트 패턴이 있는 규칙을 선택합니다.
7. 다음을 선택합니다.
8. 이벤트 소스(Event source)에서 AWS 이벤트 또는 EventBridge 파트너 이벤트(Events or EventBridge partner events)를 선택합니다.
9. 이벤트 패턴(Event pattern)섹션에서 이벤트 패턴 양식(Event pattern form)을 선택합니다.
10. 이벤트 소스에서 AWS 서비스를 선택합니다.
11. AWS service(서비스)에서 Systems Manager를 선택합니다.
12. 이벤트 유형(Event type)에서 다음 중 하나를 수행합니다.

    - 모든 이벤트(All Events)를 선택합니다.

      모든 이벤트(All Events)를 선택하면 Systems Manager에서 출력한 모든 이벤트가 규칙과 일치합니다. 이 옵션을 사용하면 많은 이벤트 대상 작업이 발생할 수 있습니다.
    - 이 규칙에 사용할 Systems Manager 이벤트의 유형을 선택합니다. EventBridge는 다음 AWS Systems Manager 기능의 이벤트를 지원합니다.
      - 자동화
      - Change Calendar
      - 규정 준수
      - 인벤토리
      - Maintenance Windows
      - Parameter Store
      - Run Command
      - State Manager

    > **Note**
    >
    > EventBridge에서 지원하지 않는 Systems Manager 작업의 경우 CloudTrail을 통해 AWS API 호출을 선택하여 CloudTrail에서 기록하는 API 호출을 기반으로 하는 이벤트 규칙을 생성할 수 있습니다. 예시는 Amazon EventBridge를 사용하여 세션 활동 모니터링(콘솔) 섹션을 참조하세요.

13. (선택 사항) 규칙을 더 구체적으로 만들려면 필터 값을 추가합니다. 예를 들어 State Manager을(를) 선택하고 연결의 대상이 되는 단일 관리형 인스턴스의 상태로 규칙을 제한하려는 경우 Specific type(s)(특정 유형)에서 EC2 State Manager Instance Association State Change(EC2 상태 관리자 인스턴스 연결 상태 변경)를 선택합니다.

지원되는 세부 유형에 대한 자세한 내용은 참조: Systems Manager용 Amazon EventBridge 이벤트 패턴 및 유형 섹션을 참조하세요.

일부 상세 유형에는 상태와 같은 다른 지원 옵션이 있습니다. 사용 가능한 옵션은 선택한 기능에 따라 다릅니다.

14. 다음을 선택합니다.
15. 대상 유형에서 AWS서비스를 선택합니다.
16. 대상 선택에서 Amazon SNS 주제 또는 AWS Lambda 함수와 같은 대상을 선택합니다. 규칙에 정의된 이벤트 패턴과 일치하는 이벤트를 수신할 때 대상이 트리거됩니다.
17. 여러 대상 유형에 대해 EventBridge에서는 대상에 이벤트를 보낼 권한이 필요합니다. 이 경우 EventBridge는 이벤트 실행에 필요한 AWS Identity and Access Management(IAM) 역할을 생성할 수 있습니다.

    - IAM 역할을 자동으로 생성하려면 이 특정 리소스에 대해 새 역할 생성을 선택합니다.
    - 이전에 생성한 IAM 역할을 사용하려면 기존 역할 사용(Use existing role)을 선택합니다.
18. (선택 사항) 이 규칙에 다른 대상을 추가하려면 Add another target(다른 대상 추가)을 선택합니다.
19. 다음을 선택합니다.
20. (선택 사항) 규칙에 대해 하나 이상의 태그를 입력하십시오. 자세한 정보는 Amazon EventBridge 사용 설명서의 Amazon EventBridge 태그를 참조하십시오.
21. 다음을 선택합니다.
22. 규칙의 세부 정보를 검토하고 규칙 생성을 선택합니다.

추가 정보

- 실행서를 사용하는 EventBridge 이벤트 생성(콘솔)
- 입력 변환기를 사용하여 Automation에 데이터 전달
- EventBridge를 사용하여 규정 준수 문제 해결
- EventBridge에서 인벤토리 삭제 작업 보기
- OpsItems를 생성하도록 EventBridge 규칙 구성
- 파라미터 및 파라미터 정책에 대해 EventBridge 구성

Systems Manager에 대한 Amazon EventBridge 이벤트 예

다음은 AWS Systems Manager에 대해 지원되는 EventBridge 이벤트의 JSON 형식 예입니다.

Systems Manager 이벤트 유형

- AWS Systems Manager Automation 이벤트
- AWS Systems Manager이벤트Change Calendar
- AWS Systems Manager이벤트Change Manager
- AWS Systems Manager Compliance 이벤트
- AWS Systems Manager이벤트Maintenance Windows
- AWS Systems Manager이벤트Parameter Store
- AWS Systems Manager이벤트OpsCenter
- AWS Systems Manager이벤트Run Command
- AWS Systems Manager이벤트State Manager

AWS Systems Manager Automation 이벤트

자동화 단계 상태 변경 알림

```
{
  "version": "0",
  "id": "eeca120b-a321-433e-9635-dab369006a6b",
  "detail-type": "EC2 Automation Step Status-change Notification",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2016-11-29T19:43:35Z",
  "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, 2016 7:43:25 PM",
    "StartTime": "Nov 29, 2016 7:43:23 PM",
    "Time": 2630.0,
    "StepName": "runFixedCmds",
    "Action": "aws:runCommand"
  }
}
```

자동화 실행 상태 변경 알림

```
{
  "version": "0",
  "id": "d290ece9-1088-4383-9df6-cd5b4ac42b99",
  "detail-type": "EC2 Automation Execution Status-change Notification",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2016-11-29T19:43:35Z",
  "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, 2016 7:43:20 PM",
    "EndTime": "Nov 29, 2016 7:43:26 PM",
    "Time": 5753.0,
    "ExecutedBy": "arn:aws:iam::123456789012:user/userName"
  }
}
```

AWS Systems Manager이벤트Change Calendar

다음은 AWS Systems Manager Change Calendar 이벤트의 예입니다.

> **Note**
>
> 다른 AWS 계정에서 공유된 일정의 상태 변경은 현재 지원되지 않습니다.

일정 OPEN

```
{
    "version": "0",
    "id": "47a3f03a-f30d-1011-ac9a-du3bdEXAMPLE",
    "detail-type": "Calendar State Change",
    "source": "aws.ssm",
    "account": "123456789012",
```

```
    "time": "2020-09-19T18:00:07Z",
    "region": "us-east-2",
    "resources": [
        "arn:aws:ssm:us-east-2:123456789012:document/MyCalendar"
    ],
    "detail": {
        "state": "OPEN",
        "atTime": "2020-09-19T18:00:07Z",
        "nextTransitionTime": "2020-10-11T18:00:07Z"
    }
}
```

일정 CLOSED

```
{
    "version": "0",
    "id": "f30df03a-1011-ac9a-47a3-f761eEXAMPLE",
    "detail-type": "Calendar State Change",
    "source": "aws.ssm",
    "account": "123456789012",
    "time": "2020-09-17T21:40:02Z",
    "region": "us-east-2",
    "resources": [
        "arn:aws:ssm:us-east-2:123456789012:document/MyCalendar"
    ],
    "detail": {
        "state": "CLOSED",
        "atTime": "2020-08-17T21:40:00Z",
        "nextTransitionTime": "2020-09-19T18:00:07Z"
    }
}
```

AWS Systems Manager이벤트Change Manager

변경 요청 상태 업데이트 알림 - 예제 1

```
{
  "version": "0",
  "id": "feab80c1-a8ff-c721-b8b1-96ce70939696",
  "detail-type": "Change Request Status Update",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2023-10-24T10:51:52Z",
```

```
  "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": "2023-10-24T10:50:33.180334Z",
    "ops-item-modified-by": "arn:aws:iam::123456789012:user/JohnDoe",
    "ops-item-modified-time": "2023-10-24T10:50:33.180340Z",
    "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"
    ]
  }
}
```

변경 요청 상태 업데이트 알림 - 예제 2

```
{
  "version": "0",
  "id": "25ce6b03-2e4e-1a2b-2a8f-6c9de8d278d2",
  "detail-type": "Change Request Status Update",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2023-10-24T10:51:52Z",
  "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": "2023-10-24T10:50:33.180334Z",
    "ops-item-modified-by": "arn:aws:iam::123456789012:user/JohnDoe",
    "ops-item-modified-time": "2023-10-24T10:50:33.997163Z",
    "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 Compliance 이벤트

다음은 AWS Systems Manager Compliance의 예입니다.

연결 규정 준수

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "Configuration Compliance State Change",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2017-07-17T19:03:26Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:ssm:us-east-2:123456789012:managed-instance/i-01234567890abcdef"
  ],
  "detail": {
    "last-runtime": "2017-01-01T10:10:10Z",
    "compliance-status": "compliant",
    "resource-type": "managed-instance",
    "resource-id": "i-01234567890abcdef",
    "compliance-type": "Association"
  }
}
```

연결 규정 미준수

```
{
```

```
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "Configuration Compliance State Change",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2017-07-17T19:02:31Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:ssm:us-east-2:123456789012:managed-instance/i-01234567890abcdef"
  ],
  "detail": {
    "last-runtime": "2017-01-01T10:10:10Z",
    "compliance-status": "non_compliant",
    "resource-type": "managed-instance",
    "resource-id": "i-01234567890abcdef",
    "compliance-type": "Association"
  }
}
```

패치 규정 준수

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "Configuration Compliance State Change",
  "source": "aws.123456789012",
  "account": "123456789012",
  "time": "2017-07-17T19:03:26Z",
  "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"
  }
}
```

패치 규정 미준수

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "Configuration Compliance State Change",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2017-07-17T19:02:31Z",
  "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 Manager이벤트Maintenance Windows

다음은 Systems Manager Maintenance Windows 이벤트의 예입니다.

대상 등록

다른 유효한 상태 값은 DEREGISTERED입니다.

```
{
   "version":"0",
   "id":"01234567-0123-0123-0123-0123456789ab",
   "detail-type":"Maintenance Window Target Registration Notification",
   "source":"aws.ssm",
   "account":"123456789012",
   "time":"2016-11-16T00:58:37Z",
   "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"
   }
}
```

Window 실행 유형

다른 유효한 상태 값은 PENDING, IN_PROGRESS, SUCCESS, FAILED, TIMED_OUT 및 SKIPPED_OVERLAPPING입니다.

```
{
   "version":"0",
   "id":"01234567-0123-0123-0123-0123456789ab",
   "detail-type":"Maintenance Window Execution State-change Notification",
   "source":"aws.ssm",
   "account":"123456789012",
   "time":"2016-11-16T01:00:57Z",
   "region":"us-east-2",
   "resources":[
      "arn:aws:ssm:us-east-2:123456789012:maintenancewindow/mw-123456789012345678"
   ],
   "detail":{
      "start-time":"2016-11-16T01:00:56.427Z",
      "end-time":"2016-11-16T01:00:57.070Z",
      "window-id":"mw-0ed7251d3fcf6e0c2",
      "window-execution-id":"b60fb56e-776c-4e5c-84ee-123456789012",
      "status":"TIMED_OUT"
   }
}
```

작업 실행 유형

다른 유효한 상태 값은 IN_PROGRESS, SUCCESS, FAILED 및 TIMED_OUT입니다.

```
{
   "version":"0",
   "id":"01234567-0123-0123-0123-0123456789ab",
   "detail-type":"Maintenance Window Task Execution State-change Notification",
   "source":"aws.ssm",
   "account":"123456789012",
   "time":"2016-11-16T01:00:56Z",
   "region":"us-east-2",
```

```
   "resources":[
      "arn:aws:ssm:us-east-2:123456789012:maintenancewindow/mw-123456789012345678"
   ],
   "detail":{
      "start-time":"2016-11-16T01:00:56.759Z",
      "task-execution-id":"6417e808-7f35-4d1a-843f-123456789012",
      "end-time":"2016-11-16T01:00:56.847Z",
      "window-id":"mw-0ed7251d3fcf6e0c2",
      "window-execution-id":"b60fb56e-776c-4e5c-84ee-123456789012",
      "status":"TIMED_OUT"
   }
}
```

처리된 작업 대상

다른 유효한 상태 값은 IN_PROGRESS, SUCCESS, FAILED 및 TIMED_OUT입니다.

```
{
   "version":"0",
   "id":"01234567-0123-0123-0123-0123456789ab",
   "detail-type":"Maintenance Window Task Target Invocation State-change Notification",
   "source":"aws.ssm",
   "account":"123456789012",
   "time":"2016-11-16T01:00:57Z",
   "region":"us-east-2",
   "resources":[
      "arn:aws:ssm:us-east-2:123456789012:maintenancewindow/mw-123456789012345678"
   ],
   "detail":{
      "start-time":"2016-11-16T01:00:56.427Z",
      "end-time":"2016-11-16T01:00:57.070Z",
      "window-id":"mw-0ed7251d3fcf6e0c2",
      "window-execution-id":"b60fb56e-776c-4e5c-84ee-123456789012",
      "task-execution-id":"6417e808-7f35-4d1a-843f-123456789012",
      "window-target-id":"e7265f13-3cc5-4f2f-97a9-123456789012",
      "status":"TIMED_OUT",
      "owner-information":"Owner"
   }
}
```

기간 상태 변경

다른 유효한 상태 값은 ENABLED 및 DISABLED입니다.

```
{
   "version":"0",
   "id":"01234567-0123-0123-0123-0123456789ab",
   "detail-type":"Maintenance Window State-change Notification",
   "source":"aws.ssm",
   "account":"123456789012",
   "time":"2016-11-16T00:58:37Z",
   "region":"us-east-2",
   "resources":[
      "arn:aws:ssm:us-east-2:123456789012:maintenancewindow/mw-123456789012345678"
   ],
   "detail":{
      "window-id":"mw-123456789012",
      "status":"DISABLED"
   }
}
```

AWS Systems Manager이벤트Parameter Store

다음은 Systems Manager Parameter Store 이벤트의 예입니다.

파라미터 생성

```
{
  "version": "0",
  "id": "6a7e4feb-b491-4cf7-a9f1-bf3703497718",
  "detail-type": "Parameter Store Change",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2017-05-22T16:43:48Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:ssm:us-east-2:123456789012:parameter/MyExampleParameter"
  ],
  "detail": {
    "operation": "Create",
    "name": "MyExampleParameter",
    "type": "String",
    "description": "Sample Parameter"
  }
}
```

파라미터 업데이트

```
{
  "version": "0",
  "id": "9547ef2d-3b7e-4057-b6cb-5fdf09ee7c8f",
  "detail-type": "Parameter Store Change",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2017-05-22T16:44:48Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:ssm:us-east-2:123456789012:parameter/MyExampleParameter"
  ],
  "detail": {
    "operation": "Update",
    "name": "MyExampleParameter",
    "type": "String",
    "description": "Sample Parameter"
  }
}
```

파라미터 삭제

```
{
  "version": "0",
  "id": "80e9b391-6a9b-413c-839a-453b528053af",
  "detail-type": "Parameter Store Change",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2017-05-22T16:45:48Z",
  "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 Manager이벤트OpsCenter

OpsCenter OpsItem 알림 생성


```
{
  "version": "0",
  "id": "aae66adc-7aac-f0c0-7854-7691e8c079b8",
  "detail-type": "OpsItem Create",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2023-10-19T02:48:11Z",
  "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": "2023-10-19T02:46:53.629361Z",
    "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 업데이트 알림


```
{
  "version": "0",
  "id": "2fb5b168-b725-41dd-a890-29311200089c",
  "detail-type": "OpsItem Update",
  "source": "aws.ssm",
  "account": "123456789012",
  "time": "2023-10-19T02:48:11Z",
  "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": "2023-10-19T02:46:54.049271Z",
```

```
    "modified-by": "arn:aws:iam::123456789012:user/JohnDoe",
    "modified-time": "2023-10-19T02:46:54.337354Z",
    "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 Manager이벤트Run Command

Run Command 상태 변경 알림

```
{
    "version": "0",
    "id": "51c0891d-0e34-45b1-83d6-95db273d1602",
    "detail-type": "EC2 Command Status-change Notification",
    "source": "aws.ssm",
    "account": "123456789012",
    "time": "2016-07-10T21:51:32Z",
    "region": "us-east-2",
    "resources": ["arn:aws:ec2:us-east-2:123456789012:instance/i-abcd1111"],
    "detail": {
        "command-id": "e8d3c0e4-71f7-4491-898f-c9b35bee5f3b",
        "document-name": "AWS-RunPowerShellScript",
        "expire-after": "2016-07-14T22:01:30.049Z",
        "parameters": {
            "executionTimeout": ["3600"],
            "commands": ["date"]
        },
        "requested-date-time": "2016-07-10T21:51:30.049Z",
        "status": "Success"
    }
}
```

Run Command 호출 상태 변경 알림

```
{
    "version": "0",
    "id": "4780e1b8-f56b-4de5-95f2-95db273d1602",
    "detail-type": "EC2 Command Invocation Status-change Notification",
```

```
    "source": "aws.ssm",
    "account": "123456789012",
    "time": "2016-07-10T21:51:32Z",
    "region": "us-east-2",
    "resources": ["arn:aws:ec2:us-east-2:123456789012:instance/i-abcd1111"],
    "detail": {
        "command-id": "e8d3c0e4-71f7-4491-898f-c9b35bee5f3b",
        "document-name": "AWS-RunPowerShellScript",
        "instance-id": "i-9bb89e2b",
        "requested-date-time": "2016-07-10T21:51:30.049Z",
        "status": "Success"
    }
}
```

AWS Systems Manager이벤트State Manager

State Manager 연결 상태 변경

```
{
   "version":"0",
   "id":"db839caf-6f6c-40af-9a48-25b2ae2b7774",
   "detail-type":"EC2 State Manager Association State Change",
   "source":"aws.ssm",
   "account":"123456789012",
   "time":"2017-05-16T23:01:10Z",
   "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":"2017-02-13T17:22:54.458Z",
      "last-successful-execution-date":"2017-05-16T23:00:01Z",
      "last-execution-date":"2017-05-16T23:00:01Z",
      "last-updated-date":"2017-02-13T17:22:54.458Z",
      "status":"Success",
      "association-status-aggregated-count":"{\"Success\":1}",
      "schedule-expression":"cron(0 */30 * * * ? *)",
      "association-cwe-version":"1.0"
```

```
    }
}
```

State Manager 인스턴스 연결 상태 변경

```
{
   "version":"0",
   "id":"6a7e8feb-b491-4cf7-a9f1-bf3703467718",
   "detail-type":"EC2 State Manager Instance Association State Change",
   "source":"aws.ssm",
   "account":"123456789012",
   "time":"2017-02-23T15:23:48Z",
   "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-12345678",
      "document-name":"my-custom-document",
      "document-version":"1",
      "targets":"[{\"key\":\"instanceids\",\"values\":[\"i-12345678\"]}]",
      "creation-date":"2017-02-23T15:23:48Z",
      "last-successful-execution-date":"2017-02-23T16:23:48Z",
      "last-execution-date":"2017-02-23T16:23:48Z",
      "status":"Success",
      "detailed-status":"",
      "error-code":"testErrorCode",
      "execution-summary":"testExecutionSummary",
      "output-url":"sampleurl",
      "instance-association-cwe-version":"1"
   }
}
```

샘플 시나리오: Amazon EventBridge 규칙의 Systems Manager 대상

Amazon EventBridge 규칙에서 호출할 대상을 지정할 때 20개 이상의 대상 유형 중에서 선택하고 각 규칙에 최대 5개의 대상을 추가할 수 있습니다.

다양한 대상 중 EventBridge 이벤트 발생 시 대상 동작으로 AWS Systems Manager의 기능인 Automation, OpsCenter 및 Run Command 중에서 선택할 수 있습니다.

다음은 이러한 기능을 EventBridge 규칙의 대상으로 사용할 수 있는 방법의 몇 가지 예입니다.

Automation 예

다음과 같은 이벤트가 발생할 때 Automation 워크플로를 시작하도록 EventBridge 규칙을 구성할 수 있습니다.

- Amazon CloudWatch 경보에서 관리형 노드가 상태 확인(`StatusCheckFailed_Instance=1`)에 실패했다고 보고하면 노드에서 `AWSSupport-ExecuteEC2Rescue` Automation 실행서를 실행합니다.
- 새 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스가 실행 중이어서 `EC2 Instance State-change Notification` 이벤트가 발생하면 인스턴스에서 `AWS-AttachEBSVolume` Automation 실행서를 실행합니다.
- Amazon Elastic Block Store(Amazon EBS) 볼륨을 생성하고 사용할 수 있는 볼륨이 생성되어 사용 가능하게 되면 해당 볼륨에서 `AWS-CreateSnapshot` Automation 실행서를 실행합니다.

OpsCenter 예제

다음과 같은 인시던트가 발생할 때 새 OpsItem을 생성하도록 EventBridge 규칙을 구성할 수 있습니다.

- Amazon DynamoDB에 대한 제한 이벤트가 발생하거나 Amazon EBS 볼륨 성능이 저하되었습니다.
- Amazon EC2 Auto Scaling 그룹이 노드를 시작하지 못하거나 Systems Manager Automation 워크플로가 실패합니다.
- EC2 인스턴스가 `Running`에서 `Stopped`로 상태를 변경합니다.

Run Command 예제

다음과 같은 이벤트가 발생할 때 Run Command에서 Systems Manager Command 문서를 실행하도록 EventBridge 규칙을 구성할 수 있습니다.

- Auto Scaling 그룹이 종료될 때 Run Command 스크립트는 종료되기 전에 노드에서 로그 파일을 캡처할 수 있습니다.
- Auto Scaling 그룹에 새 노드가 생성될 때 Run Command 대상 작업은 웹 서버 역할을 설정하거나 노드에 소프트웨어를 설치할 수 있습니다.
- 관리형 노드가 규정을 준수하지 않는 것으로 확인되면 Run Command 대상 작업은 `AWS-RunPatchBaseline` 문서를 실행하여 노드의 패치를 업데이트할 수 있습니다.

Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링

> **Note**
>
> Amazon Simple Notification Service FIFO 주제는 지원되지 않습니다.

AWS Systems Manager의 기능인 Run Command 또는 Maintenance Windows를 사용하여 보내는 명령의 상태에 대한 알림을 보내도록 Amazon Simple Notification Service(Amazon SNS)를 구성할 수 있습니다. Amazon SNS는 Amazon SNS 주제를 구독하는 클라이언트 또는 엔드포인트에 알림을 보내고 전송하는 작업을 조정하고 관리합니다. 명령이 새로운 상태로 바뀌거나 실패 또는 시간 초과 같은 특정 상태가 될 때마다 알림을 받을 수 있습니다. 여러 노드에 명령을 보내는 경우, 특정 노드로 보낸 각각의 명령 사본에 대해 알림을 받을 수 있습니다. 이때 각 사본을 호출이라고 합니다.

Amazon SNS는 HTTP 또는 HTTPS POST, 이메일(SMTP, 일반 텍스트 또는 JSON 형식) 또는 Amazon Simple Queue Service(Amazon SQS) 대기열에 게시된 메시지로 알림을 배달할 수 있습니다. 자세한 내용은 Amazon Simple Notification Service Developer Guide의 What is Amazon SNS를 참조하세요. Run Command 및 Maintenance Windows에서 제공한 Amazon SNS 알림에 포함된 JSON 데이터 구조의 예는 AWS Systems Manager에 대한 Amazon SNS 알림 예 섹션을 참조하세요.

AWS Systems Manager에 대한 Amazon SNS 알림 구성

유지 관리 기간에 등록된 Run Command 및 Maintenance Windows 태스크는 다음 상태를 시작하는 명령 태스크에 대한 Amazon SNS 알림을 보낼 수 있습니다.

- 진행 중
- Success
- Failed
- 시간 초과
- 취소됨

명령을 다음 상태로 만드는 조건에 대한 자세한 내용은 명령 상태 이해 섹션을 참조하세요.

> **Note**
>
> 또한 Run Command를 사용해 전송된 명령은 취소 중 및 보류 중 상태를 보고합니다. 이러한 상태는 Amazon SNS 알림에서 캡처하지 않습니다.

명령 요약 Amazon SNS 알림

유지 관리 기간에 Amazon SNS 알림을 보내도록 Run Command 또는 Run Command 태스크를 구성한 경우 Amazon SNS는 다음 정보가 포함된 요약 메시지를 전송합니다.

필드	유형	설명
eventTime	String	이벤트가 시작된 시간입니다. Amazon SNS는 메시지의 배달 순서를 보장하지 않기 때문에 타임스탬프가 중요합니다. 예: 2016-04-26T13:15:30Z
documentName	String	이 명령을 실행하는 데 사용된 SSM 문서의 이름입니다.
commandId	String	명령을 전송한 후 Run Command에서 생성한 ID입니다.
expiresAfter	날짜	이 시간에 도달할 때까지 명령 실행이 아직 시작되지 않은 경우 명령이 실행되지 않습니다.
outputS3BucketName	String	명령 실행에 대한 응답을 저장해야 하는 Amazon Simple Storage Service(Amazon S3) 버킷입니다.
outputS3KeyPrefix	String	명령 실행에 대한 응답을 저장해야 하는 버킷 내부의

필드	유형	설명
		Amazon S3 디렉터리 경로입니다.
requestedDateTime	String	이 특정 노드로 요청을 보낸 날짜와 시간입니다.
instanceIds	StringList	명령의 대상이 되는 노드입니다. **Note** Run Command 작업이 직접 인스턴스 ID를 대상으로 한 경우에만 인스턴스 ID가 요약 메시지에 포함됩니다. Run Command 태스크가 태그 기반 대상 지정을 사용하여 실행된 경우 인스턴스 ID는 요약 메시지에 포함되지 않습니다.
status	String	명령의 상태입니다.

호출 기반 Amazon SNS 알림

여러 노드에 명령을 보내는 경우 Amazon SNS는 각각의 명령 사본 또는 명령 호출에 대해 메시지를 보내 줄 수 있습니다. 메시지에 포함하는 정보는 다음과 같습니다.

필드	유형	설명
eventTime	String	이벤트가 시작된 시간입니다. Amazon SNS는 메시지의 배달 순서를 보장하지 않기 때문에

필드	유형	설명
		타임스탬프가 중요합니다. 예: 2016-04-26T13:15:30Z
documentName	String	이 명령을 실행하는 데 사용된 Systems Manager 문서(SSM 문서)의 이름입니다.
requestedDateTime	String	이 특정 노드로 요청을 보낸 날짜와 시간입니다.
commandId	String	명령을 전송한 후 Run Command에서 생성한 ID입니다.
instanceId	String	명령의 대상이 된 인스턴스입니다.
status	String	이 호출의 명령 상태입니다.

명령의 상태가 바뀔 때 Amazon SNS 알림을 보내도록 설정하려면 다음 태스크를 완료합니다.

> **Note**
>
> 유지 관리 기간에 대해 Amazon SNS 알림을 구성하지 않는 경우 이 주제 뒷부분의 태스크 5를 건너뛸 수 있습니다.

주제

- 태스크 1: Amazon SNS 주제 생성 및 구독
- 태스크 2: Amazon SNS 알림을 위한 IAM 정책 생성
- 태스크 3: Amazon SNS 알림을 위한 IAM 역할 생성
- 작업 4: 사용자 액세스 구성
- 작업 5: 유지 관리 기간 역할에 iam:PassRole 정책 연결

태스크 1: Amazon SNS 주제 생성 및 구독

Amazon SNS 주제는 유지 관리 기간에 등록된 Run Command 및 Run Command 태스크가 명령 상태에 대한 알림을 보내는 데 사용하는 통신 채널입니다. Amazon SNS는 HTTP/S, 이메일 및 Amazon Simple Queue Service(Amazon SQS)와 같은 기타 AWS 서비스을(를) 비롯한 다양한 통신 프로토콜을 지원합니다. 시작하려면 이메일 프로토콜부터 시작하는 것이 좋습니다. 주제 생성 방법에 대한 자세한 내용은 Amazon Simple Notification Service 개발자 안내서의 Amazon SNS 주제 생성을 참조하세요.

> **Note**
>
> 주제를 생성한 뒤 주제 ARN을 복사하거나 기록해 둡니다. 상태 알림을 반환하도록 구성된 명령을 보낼 때 이 ARN을 지정해야 합니다.

주제를 생성한 후에는 엔드포인트를 지정하여 이를 구독합니다. 이메일 프로토콜을 선택한 경우에는 알림을 수신할 이메일 주소가 엔드포인트가 됩니다. 주제 구독 방법에 대한 자세한 내용은 Amazon Simple Notification Service 개발자 안내서의 Amazon SNS 주제 구독을 참조하세요.

Amazon SNS는 AWS 알림에서 지정한 이메일 주소로 확인 이메일을 보냅니다. 이메일을 열고 구독 확인 링크를 선택합니다.

AWS에서 확인 메시지를 받게 됩니다. 이제 Amazon SNS가 지정된 이메일 주소로 이메일 형식의 알림을 주고받을 수 있도록 구성됩니다.

태스크 2: Amazon SNS 알림을 위한 IAM 정책 생성

다음 절차에 따라 Amazon SNS 알림을 시작할 수 있는 권한을 제공하는 사용자 정의 AWS Identity and Access Management(IAM) 정책을 생성합니다.

Amazon SNS 알림을 위한 사용자 정의 IAM 정책을 생성하려면

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 정책을 선택한 후 정책 생성을 선택합니다. ([시작하기(Get Started)] 버튼이 나타나면 이 버튼을 선택한 후 [정책 생성(Create Policy)]을 선택합니다.)
3. JSON 탭을 선택합니다.
4. 기본 내용을 다음으로 바꿉니다.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sns:Publish"
            ],
            "Resource": "arn:aws:sns:region:account-id:sns-topic-name"
        }
    ]
}
```

*##*은 미국 동부(오하이오) 리전의 `us-east-2` 같이 AWS Systems Manager가 지원하는 AWS 리전의 식별자를 나타냅니다. 지원되는 *##* 값 목록은 Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트에 있는 리전 열을 참조하세요.

*account-id*는 AWS 계정에 대한 `123456789012` 형식의 12자리 식별자를 나타냅니다.

*sns-topic-name*은 알림 게시에 사용할 Amazon SNS 주제의 이름을 나타냅니다.

5. 다음: 태그를 선택합니다.
6. (선택 사항) 이 정책에 대한 액세스를 구성, 추적 또는 제어할 태그-키 값 페어를 하나 이상 추가합니다.
7. 다음: 검토를 선택합니다.
8. Review Policy(정책 검토) 페이지의 이름에 인라인 정책 이름을 입력합니다. 예: **my-sns-publish-permissions**.
9. (선택 사항) 설명에 정책에 대한 설명을 입력합니다.
10. 정책 생성을 선택합니다.

태스크 3: Amazon SNS 알림을 위한 IAM 역할 생성

다음 절차를 사용하여 Amazon SNS 알림을 위한 IAM 역할을 생성합니다. 이 서비스 역할은 Systems Manager에서 Amazon SNS 알림을 시작하는 데 사용됩니다. 이후의 모든 절차에서 이 역할을 Amazon SNS IAM 역할이라고 합니다.

Amazon SNS 알림을 위한 IAM 서비스 역할을 생성하려면

1. AWS Management Console에 로그인하고 https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.

2. IAM 콘솔의 탐색 창에서 역할을 선택하고 역할 생성을 선택합니다.
3. AWS 서비스 역할 유형을 선택한 후 Systems Manager를 선택합니다.
4. Systems Manager 사용 사례를 선택합니다. 그리고 다음을 선택합니다.
5. Attach permissions policies(권한 정책 연결) 페이지에서 작업 2에서 생성한 사용자 지정 정책의 이름 왼쪽에 있는 확인란을 선택합니다. 예: **`my-sns-publish-permissions`**.
6. (선택 사항) 권한 경계를 선택합니다. 이는 서비스 역할에서 가능한 고급 기능이며 서비스 링크된 역할은 아닙니다.

   권한 경계 섹션을 열고 최대 역할 권한을 관리하기 위한 권한 경계 사용을 선택합니다. IAM은 계정의 AWS 관리형 또는 고객 관리형 정책 목록을 포함합니다. 권한 경계를 사용하기 위한 정책을 선택하거나 정책 생성을 선택하여 새 브라우저 탭을 열고 완전히 새로운 정책을 생성합니다. 자세한 내용은 IAM 사용자 설명서에서 IAM 정책 생성을 참조하세요. 정책을 생성하면 탭을 닫고 원래 탭으로 돌아와 권한 경계에 사용할 정책을 선택합니다.
7. 다음을 선택합니다.
8. 가능한 경우 이 역할의 목적을 식별하는 데 도움이 되는 역할 이름이나 역할 이름 접미사를 입력합니다. 역할 이름은 AWS 계정 내에서 고유해야 합니다. 대소문자는 구별하지 않습니다. 예를 들어, 이름이 **`PRODROLE`**과 **`prodrole`**, 두 가지로 지정된 역할을 만들 수는 없습니다. 다양한 주체가 역할을 참조할 수 있기 때문에 역할이 생성된 후에는 역할 이름을 편집할 수 없습니다.
9. (선택 사항)설명에 새 역할에 대한 설명을 입력합니다.
10. 1단계: 신뢰할 수 있는 엔터티 선택(Step 1: Select trusted entities) 또는 2단계: 권한 선택(Step 2: Select permissions) 섹션에서 편집(Edit)을 선택하여 역할에 대한 사용 사례와 권한을 편집합니다.
11. (선택 사항) 태그를 키 값 페어로 연결하여 메타데이터를 사용자에게 추가합니다. IAM에서 태그 사용에 대한 자세한 내용을 알아보려면 IAM 사용 설명서의 IAM 리소스에 태그 지정을 참조하십시오.
12. 역할을 검토한 다음 역할 생성을 선택합니다.
13. 역할 이름을 선택한 다음 역할 ARN 값을 복사하거나 기록해 둡니다. 이 역할에 대한 Amazon Resource Name(ARN)은 Amazon SNS 알림을 반환하도록 구성된 명령을 보낼 때 사용됩니다.
14. [요약(Summary)] 페이지가 열립니다.

작업 4: 사용자 액세스 구성

IAM 엔터티(사용자, 역할 또는 그룹)에 관리자 권한이 할당된 경우 사용자 또는 역할은 AWS Systems Manager의 기능인 Run Command와 Maintenance Windows에 액세스할 수 있습니다.

관리자 권한이 없는 엔터티의 경우 관리자가 IAM 엔터티에 다음 권한을 부여해야 합니다.

- `AmazonSSMFullAccess` 관리형 정책 또는 유사한 권한을 제공하는 정책.
- `iam:PassRole`에서 생성된 역할에 대한 태스크 3: Amazon SNS 알림을 위한 IAM 역할 생성 권한. 예:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::account-id:role/sns-role-name"
        }
    ]
}
```

액세스 권한을 제공하려면 사용자, 그룹 또는 역할에 권한을 추가하세요:

- AWS IAM Identity Center의 사용자 및 그룹:

  권한 세트를 생성합니다. AWS IAM Identity Center 사용 설명서의 권한 세트 생성의 지침을 따르세요.
- ID 제공자를 통해 IAM에서 관리되는 사용자:

  ID 페더레이션을 위한 역할을 생성합니다. IAM 사용 설명서의 서드 파티 자격 증명 공급자의 역할 만들기(연합)의 지침을 따르세요.
- IAM 사용자:
  - 사용자가 맡을 수 있는 역할을 생성합니다. IAM 사용 설명서에서 IAM 사용자의 역할 생성의 지침을 따르세요.
  - (권장되지 않음)정책을 사용자에게 직접 연결하거나 사용자를 사용자 그룹에 추가합니다. IAM 사용 설명서에서 사용자(콘솔)에 권한 추가의 지침을 따르세요.

사용자 액세스를 구성하고 사용자 계정에 **`iam:PassRole`** 정책을 연결하려면

1. IAM 탐색 창에서 [사용자(Users)]를 선택한 후 구성할 사용자 계정을 선택합니다.

2. [권한(Permissions)] 탭의 정책 목록에서 **AmazonSSMFullAccess** 정책이 있는지, 아니면 Systems Manager에 액세스할 수 있는 권한을 계정에 부여하는 유사한 정책이 있는지 확인합니다.
3. 인라인 정책 추가(Add inline policy)를 선택합니다.
4. 정책 생성(Create policy) 페이지에서 시각적 편집기(Visual editor) 탭을 선택합니다.
5. [서비스 선택(Choose a service)]을 선택한 다음 [IAM]을 선택합니다.
6. 작업(Actions)에서 작업 필터링(Filter actions) 텍스트 상자에 **PassRole**을 입력하고 PassRole 옆의 확인란을 선택합니다.
7. [리소스(Resources)]에 대해 [특정(Specific)]이 선택되었는지 확인한 다음 [ARN 추가(Add ARN)]를 선택합니다.
8. [역할에 대한 ARN 지정(Specify ARN for role)] 필드에서 태스크 3의 마지막에 복사한 Amazon SNS IAM 역할 ARN을 붙여 넣습니다. 계정 및 Role name with path(역할 이름 및 경로) 필드에 값이 자동으로 입력됩니다.
9. 추가(Add)를 선택합니다.
10. 정책 검토(Review policy)를 선택합니다.
11. 정책 검토(Review Policy) 페이지에 이름을 입력한 다음 정책 생성(Create Policy)을 선택합니다.

작업 5: 유지 관리 기간 역할에 iam:PassRole 정책 연결

유지 관리 기간으로 Run Command 작업을 등록할 때 서비스 역할 Amazon 리소스 이름(ARN)을 지정합니다. 이 서비스 역할은 Systems Manager가 유지 관리 기간에 등록된 태스크를 실행하는 데 사용됩니다. 등록된 Run Command 태스크에 대한 Amazon SNS 알림을 구성하려면 지정된 유지 관리 기간 서비스 역할에 `iam:PassRole` 정책을 연결합니다. Amazon SNS 알림을 위해 등록된 태스크를 구성하지 않으려면 이 태스크를 건너뛸 수 있습니다.

`iam:PassRole` 정책은 Maintenance Windows 서비스 역할이 태스크 3에서 생성된 Amazon SNS IAM 역할을 Amazon SNS 서비스로 전달하도록 허용합니다. 다음 절차에서는 `iam:PassRole` 정책을 Maintenance Windows 서비스 역할에 연결하는 방법을 보여줍니다.

> **Note**
>
> 등록된 Run Command 태스크와 관련된 알림을 보내려면 유지 관리 기간의 사용자 정의 서비스 역할을 사용합니다. 자세한 설명은 Maintenance Windows 설정을 참조하세요.
> 유지 관리 기간 작업을 위해 사용자 지정 서비스 역할을 생성한 경우 콘솔을 사용하여 유지 관리 기간에 대한 권한 구성의 내용을 참조하세요.

Maintenance Windows 역할에 **`iam:PassRole`** 정책을 연결하려면

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 엽니다.
2. 탐색 창에서 역할(Roles)을 선택하고, 태스크 3에서 생성한 Amazon SNS IAM 역할을 선택합니다.
3. [역할 ARN(Role ARN)]을 복사하거나 기록하고 IAM 콘솔의 [역할(Roles)] 섹션으로 돌아갑니다.
4. 역할 이름(Role name) 목록에서 생성한 사용자 지정 Maintenance Windows 서비스 역할을 선택합니다.
5. 권한(Permissions) 탭에서 `AmazonSSMMaintenanceWindowRole` 정책이 나열되는지 또는 유지 관리 기간에 Systems Manager API에 대한 권한을 부여하는 유사한 정책이 있는지 확인합니다. 그렇지 않은 경우 권한 추가, 정책 연결을 선택하여 연결합니다.
6. 권한 추가, 인라인 정책 추가(Add permissions, Create inline policy)를 선택합니다.
7. [시각적 편집기(Visual Editor)] 탭을 선택합니다.
8. [서비스(Service)]에서 [IAM]을 선택합니다.
9. 작업(Actions)에서 작업 필터링(Filter actions) 텍스트 상자에 **`PassRole`**을 입력하고 PassRole 옆의 확인란을 선택합니다.
10. 리소스에서 Specific(특정)을 선택한 다음 Add ARN(ARN 추가)를 선택합니다.
11. [역할의 ARN 지정(Specify ARN for role)] 상자에서 태스크 3에서 생성된 Amazon SNS IAM 역할의 ARN을 붙여 넣은 다음, [추가(Add)]를 선택합니다.
12. 정책 검토를 선택합니다.
13. 정책 검토 페이지에서 `PassRole` 정책의 이름을 지정한 다음 정책 생성을 선택합니다.

AWS Systems Manager에 대한 Amazon SNS 알림 예

AWS Systems Manager의 기능인 Run Command 또는 Maintenance Windows를 사용하여 보내는 명령의 상태에 대한 알림을 보내도록 Amazon Simple Notification Service(Amazon SNS)를 구성할 수 있습니다.

> **Note**
>
> 이 설명서에서는 Run Command 또는 Maintenance Windows 알림을 구성하는 방법을 다루지 않습니다. 명령 상태에 대한 Amazon SNS 알림을 보내기 위한 Run Command 또는 Maintenance Windows 구성에 대한 자세한 내용은 AWS Systems Manager에 대한 Amazon SNS 알림 구성 섹션을 참조하세요.

다음 예에서는 Run Command 또는 Maintenance Windows에 대해 구성된 경우 Amazon SNS 알림에 의해 반환된 JSON 출력의 구조를 보여줍니다.

인스턴스 ID 타겟팅을 사용하여 명령 요약 메시지의 샘플 JSON 출력

```
{
    "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": "DOC-EXAMPLE-BUCKET",
    "outputS3KeyPrefix": "runcommand",
    "status": "InProgress",
    "eventTime": "2019-04-25T17:57:09.236Z"
}
```

태그 기반 타겟팅을 사용하여 명령 요약 메시지의 샘플 JSON 출력

```
{
    "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"
}
```

호출 메시지를 위한 샘플 JSON 출력

```
{
    "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"
}
```

Run Command을 사용하여 상태 알림을 반환하는 명령 전송

다음 절차에서는 상태 경보를 반환하도록 구성된 AWS Systems Manager의 기능인 Run Command를 통해 AWS Command Line Interface(AWS CLI) 또는 AWS Systems Manager 콘솔을 사용하여 명령을 보내는 방법을 보여줍니다.

알림을 반환하는 Run Command 전송(콘솔)

Systems Manager 콘솔을 사용하여 상태 알림을 반환하도록 구성된 Run Command를 통해 명령을 보내려면 다음 절차를 사용합니다.

알림을 반환하는 명령을 전송하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택합니다.
3. Run command(Run 명령)를 선택합니다.
4. Command 문서(Command document) 목록에서 Systems Manager 문서를 선택합니다.
5. 명령 파라미터 섹션에서 필요한 파라미터의 값을 지정합니다.
6. Targets(대상) 섹션에서, 태그를 지정하거나, 수동으로 인스턴스나 엣지 디바이스를 선택하거나, 리소스 그룹을 지정하여 이 작업을 실행할 관리형 노드를 식별합니다.

   > **Tip**
   >
   > 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

7. Other parameters(다른 파라미터):
   - Comment(설명)에 명령에 대한 정보를 입력합니다.
   - 제한 시간(초)에서 전체 명령 실행이 실패할 때까지 시스템이 기다리는 시간을 초 단위로 지정합니다.
8. Rate control(속도 제어)에서
   - Concurrency(동시성)에서 명령을 동시에 실행할 관리형 노드의 백분율 또는 개수를 지정합니다.

> **Note**
>
> 관리형 노드에 적용할 태그를 지정하거나, AWS 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상 수를 제한합니다.

- Error threshold(오류 임계값)에서, 명령이 노드의 개수 또는 백분율에서 실패한 후 다른 관리형 노드에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째 오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.

9. (선택 사항) Output options(출력 옵션)에서 명령 출력을 파일에 저장하려면 Write command output to an S3 bucket(S3 버킷에 명령 출력 쓰기) 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

> **Note**
>
> 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아니라 인스턴스에 할당된 인스턴스 프로파일(EC2 인스턴스용) 또는 IAM 서비스 역할(하이브리드 정품 인증 시스템)의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

10. SNS 알림 섹션에서 SNS 알림 활성화를 선택합니다.
11. IAM 역할(IAM role)에서 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링의 작업 3에서 생성한 Amazon SNS IAM 역할 ARN을 선택합니다.
12. [SNS 주제(SNS topic)]에 사용할 Amazon SNS 주제 ARN을 입력합니다.
13. [이벤트 알림(Event notifications)]에서 알림을 받을 이벤트를 선택합니다.
14. 변경 사항 알림(Change notifications)에서 명령 요약에 대한 알림만 수신하거나(명령 상태 변경(Command status changes)) 또는 여러 노드로 전송된 명령 사본마다 알림을 수신하도록 선택합니다(각 인스턴스 변경에 대한 명령 상태(Command status on each instance changes)).
15. Run(실행)을 선택합니다.
16. 이메일에서 Amazon SNS의 메시지를 확인하고 이메일 메시지를 엽니다. Amazon SNS에서 이메일 메시지를 보내는 데 몇 분 정도 걸릴 수 있습니다.

알림을 반환하는 Run Command 보내기(CLI)

AWS CLI를 사용하여 상태 알림을 반환하도록 구성된 Run Command 을 통해 명령을 보내려면 다음 절차를 사용하십시오.

상태 알림을 반환하는 명령 보내기(CLI)

1. AWS CLI을 엽니다.
2. 관리형 노드 ID를 기반으로 대상을 지정하려면 다음 명령에서 파라미터를 지정합니다.

```
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"}'
```

다음은 한 예입니다.

```
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"}'
```

다른 명령

태그를 사용하여 관리형 인스턴스를 대상으로 지정하려면 다음 명령에서 파라미터를 지정합니다.

```
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"}'
```

다음은 한 예입니다.

```
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. Enter를 누릅니다.
4. 이메일에서 Amazon SNS의 메시지를 확인하고 이메일 메시지를 엽니다. Amazon SNS에서 이메일 메시지를 보내는 데 몇 분 정도 걸릴 수 있습니다.

자세한 내용은 AWS CLI 명령 레퍼런스의 send-command 섹션을 참조하세요.

유지 관리 기간을 사용하여 상태 알림을 반환하는 명령 전송

다음 절차에서는 AWS Systems Manager 콘솔 또는 AWS Command Line Interface(AWS CLI)를 사용하여 Run Command 태스크를 유지 관리 기간에 등록하는 방법을 보여줍니다. Run Command는 AWS Systems Manager의 기능입니다. 이 절차에서는 상태 알림을 반환하도록 Run Command 작업을 구성하는 방법에 대해서도 설명합니다.

시작하기 전 준비 사항

유지 관리 기간을 생성하거나 대상을 등록하지 않은 경우, 유지 관리 기간 작업(콘솔)에서 유지 관리 기간을 생성하고 대상을 등록하는 방법에 대한 단계를 참조하십시오.

Amazon Simple Notification Service(Amazon SNS) 서비스에서 알림을 수신하려면 등록된 태스크에 지정된 Maintenance Windows 서비스 역할에 `iam:PassRole` 정책을 연결합니다. Maintenance Windows 서비스 역할에 `iam:PassRole` 권한을 추가하지 않은 경우 작업 5: 유지 관리 기간 역할에 iam:PassRole 정책 연결 섹션을 참조하세요.

알림을 반환하는 유지 관리 기간에 Run Command 작업 등록(콘솔)

Systems Manager 콘솔을 사용하여 유지 관리 기간에 상태 알림을 반환하도록 구성된 Run Command 태스크를 등록하려면 다음 절차를 사용합니다.

알림을 반환하는 유지 관리 기간에 Run Command 작업을 등록하려면(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Maintenance Windows를 선택합니다.
3. Amazon Simple Notification Service(Amazon SNS) 알림을 보내도록 구성된 Run Command 태스크를 등록할 유지 관리 기간을 선택합니다.
4. [작업(Actions)], [Run command 태스크 등록(Register Run command task)]을 차례로 선택합니다.

5. (옵션) [이름(Name)] 필드에 태스크의 이름을 입력합니다.
6. (옵션) [설명(Description)]에 설명을 입력합니다.
7. 명령 문서(Command document)에서 명령 문서를 선택합니다.
8. 작업 우선순위에서 이 작업의 우선순위를 지정합니다. 가장 높은 우선순위는 0입니다. Maintenance Window의 작업은 우선순위에 따라 예약됩니다. 우선순위가 같은 작업은 동시에 일정이 수립됩니다.
9. [대상(Targets)] 섹션에서 등록된 대상 그룹을 선택하거나 등록되지 않은 대상을 선택합니다.
10. Rate control(속도 제어)에서

    - Concurrency(동시성)에서 명령을 동시에 실행할 관리형 노드의 백분율 또는 개수를 지정합니다.

      > **Note**
      >
      > 관리형 노드에 적용할 태그를 지정하거나, AWS 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상 수를 제한합니다.

    - Error threshold(오류 임계값)에서, 명령이 노드의 개수 또는 백분율에서 실패한 후 다른 관리형 노드에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째 오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.
11. [IAM 서비스 역할(IAM service role)] 영역에서 SNS 역할에 대한 `iam:PassRole` 권한이 있는 Maintenance Windows 서비스 역할을 선택합니다.

    > **Note**
    >
    > Systems Manager가 SNS 역할을 Amazon SNS에 전달할 수 있도록 Maintenance Windows 역할에 `iam:PassRole` 권한을 추가합니다. `iam:PassRole` 권한을 추가하지 않은 경우에는 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 주제의 작업 5를 참조하십시오.
12. (선택 사항) 출력 옵션에서 명령 출력을 파일에 저장하려면 S3 버킷에 쓰기 활성화 옆의 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

> **Note**
>
> 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아닌 관리형 노드에 할당된 인스턴스 프로파일의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

13. [SNS 알림(SNS notifications)] 섹션에서 다음을 수행합니다.
    - [SNS 알림 선택(Enable SNS Notifications)]을 선택합니다.
    - [IAM 역할(IAM role)]에서 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링의 태스크 3에서 생성한 Amazon SNS IAM 역할 Amazon 리소스 이름(ARN)을 선택하여 Amazon SNS를 시작합니다.
    - [SNS 주제(SNS topic)]에 사용할 Amazon SNS 주제 ARN을 입력합니다.
    - [이벤트 유형(Event type)] 섹션에서 알림을 받을 이벤트를 선택합니다.
    - 알림 유형(Notification type)에서 여러 노드로 보낸 명령의 각 사본(호출)에 대해 알림을 받을 것인지 아니면 명령 요약에 대해 알림을 받을 것인지 선택합니다.
14. [파라미터(Parameters)] 섹션에 선택한 Command 문서를 기반으로 필요한 파라미터를 입력합니다.
15. [Run command 태스크 등록(Register Run command task)]을 선택합니다.
16. 다음에 유지 관리 기간이 실행되면 이메일에서 Amazon SNS의 메시지를 확인하고 이메일 메시지를 엽니다. Amazon SNS에서 이메일 메시지를 보내는 데 몇 분 정도 걸릴 수 있습니다.

알림을 반환하는 유지 관리 기간에 Run Command 작업 등록(CLI)

AWS CLI를 사용하여 유지 관리 기간에 상태 알림을 반환하도록 구성된 Run Command 작업을 등록하려면 다음 절차를 사용하십시오.

알림을 반환하는 유지 관리 기간으로 Run Command 작업을 등록하려면(CLI)

> **Note**
>
> 이 절차는 작업 옵션을 보다 잘 관리하기 위해 `--cli-input-json` 명령 옵션을 사용하며 옵션 값은 JSON 파일에 저장됩니다.

1. 로컬 시스템에 `RunCommandTask.json`이라는 파일을 생성하십시오.
2. 다음 내용을 파일에 붙여 넣습니다.

```
{
    "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. 예제 값을 고유한 리소스에 대한 정보로 바꾸십시오.

   이 예제에서 생략한 옵션을 사용하려는 경우 복원할 수도 있습니다. 예를 들어 명령 출력을 S3 버킷에 저장할 수 있습니다.

   자세한 정보는 AWS CLI 명령 참조의 register-task-with-maintenance-window 섹션을 참조하세요.
4. 파일을 저장합니다.
5. 파일을 저장한 로컬 시스템 디렉터리에서 다음 명령을 실행합니다.

```
aws ssm register-task-with-maintenance-window --cli-input-json file://
RunCommandTask.json
```

> **Important**
>
> 파일 이름 앞에 `file://`를 포함해야 합니다. 이 명령에 필수적입니다.

   명령이 제대로 실행되면 다음과 비슷한 정보를 반환합니다.

```
{
    "WindowTaskId": "j2l8d5b5c-mw66-tk4d-r3g9-1d4d1EXAMPLE"
}
```

6. 유지 관리 기간의 다음 실행 후 이메일에서 Amazon SNS의 메시지를 확인하고 이메일 메시지를 엽니다. Amazon SNS에서 이메일 메시지를 보내는 데 몇 분 정도 걸릴 수 있습니다.

명령줄에서 유지 관리 기간에 태스크 등록에 대한 자세한 내용은 유지 관리 기간에 태스크 등록을 참조하세요.

Systems Manager와 제품 및 서비스 통합

기본적으로 AWS Systems Manager은(는) AWS 서비스 및 기타 제품 및 서비스와 통합됩니다. 다음 정보를 통해 사용 중인 제품과 서비스를 통합할 Systems Manager를 구성할 수 있습니다.

- AWS 서비스와의 통합
- 다른 제품 및 서비스와 통합

AWS 서비스와의 통합

Systems Manager Command 문서(SSM 문서)와 Automation 실행서 사용을 통해 AWS Systems Manager을(를) 사용하여 AWS 서비스와(과) 통합할 수 있습니다. 이러한 리소스에 대한 자세한 내용은 AWS Systems Manager Documents 섹션을 참조하세요.

Systems Manager는 다음 AWS 서비스와(과) 통합되어 있습니다.

컴퓨팅

Amazon Elastic Compute Cloud(Amazon EC2)	Amazon EC2는 AWS 클라우드에서 확장 가능한 컴퓨팅 용량을 제공합니다. Amazon EC2를 사용하면 하드웨어에 사전 투자할 필요가 없어 더 빠르게 애플리케이션을 개발하고 배포할 수 있습니다. Amazon EC2를 사용하여 원하는 수의 가상 서버를 구축하고 보안 및 네트워킹을 구성하며 스토리지를 관리할 수 있습니다. Systems Manager를 사용하면 EC2 인스턴스에서 여러 태스크를 수행할 수 있습니다. 예를 들어 EC2 인스턴스를 시작, 구성, 관리, 유지 관리, 문제 해결 및 안전하게 연결할 수 있습니다. 또한 Systems Manager를 사용하여 소프트웨어를 배포하고, 규정 준수 상태를 확인하고, EC2 인스턴스에서 인벤토리를 수집할 수 있습니다. 자세히 알아보기 • 관리형 노드 작업

	• AWS Systems Manager State Manager • AWS Systems Manager Run Command • AWS Systems Manager Patch Manager • AWS Systems Manager Session Manager • AWS Systems Manager Distributor • AWS Systems Manager Compliance • AWS Systems Manager Inventory
Amazon EC2 Auto Scaling	Auto Scaling을 사용하면 애플리케이션의 로드를 처리할 수 있는 정확한 수의 EC2 인스턴스를 유지할 수 있습니다. Auto Scaling 그룹이라는 EC2 인스턴스 모음을 생성합니다. Systems Manager를 사용하면 Auto Scaling 그룹의 Auto Scaling 템플릿에 사용된 Amazon Machine Image(AMI) 패치와 같은 일반적인 절차를 자동화할 수 있습니다. 자세히 알아보기 오토 스케일링을 위해 AMIs 업데이트

Amazon Elastic Container Service(Amazon ECS)	Amazon ECS는 클러스터에서 도커 컨테이너를 손쉽게 실행, 중지 및 관리할 수 있게 하는 컨테이너 관리 서비스로서 확장성과 속도가 뛰어납니다. Systems Manager를 사용하면 Systems Manager의 기능인 Parameter Store의 파라미터에 민감한 데이터를 저장한 다음 컨테이너 정의에서 참조하여 컨테이너 인스턴스를 원격으로 관리하고 민감한 데이터를 컨테이너에 주입할 수 있습니다. 자세히 알아보기 • AWS Systems Manager를 사용하여 원격으로 컨테이너 인스턴스 관리 • Systems Manager Parameter Store를 사용하여 민감한 데이터 지정

AWS Lambda	Lambda는 서버를 프로비저닝하거나 관리하지 않고도 코드를 실행할 수 있게 하는 컴퓨팅 서비스입니다. Lambda는 필요 시에만 코드를 실행하며, 일일 몇 개의 요청에서 초당 수천 개의 요청까지 자동으로 규모를 조정합니다. Systems Manager에서는 `aws:invokeLambdaFunction` 작업을 사용하여 Automation 실행서 콘텐츠 내에서 Lambda 함수를 사용할 수 있습니다. AWS Lambda 함수에서 Parameter Store의 파라미터를 사용하려면 AWS 파라미터 및 보안 Lambda 익스텐션을 사용하여 파라미터 값을 검색하고 나중에 사용하기 위해 캐시할 수 있습니다. 자세히 알아보기 Automation, AWS Lambda 및 Parameter Store를 사용하여 골든 AMI 업데이트 AWS Lambda 함수에서 Parameter Store 파라미터 사용

사물 인터넷(IoT)

AWS IoT Greengrass 코어 디바이스	AWS IoT Greengrass은 디바이스에서 IoT 애플리케이션을 구축, 배포 및 관리하는 데 도움이 되는 오픈 소스 IoT 에지 런타임 및 클라우드 서비스입니다. Systems Manager는AWS IoT Greengrass 코어 디바이스를 위한 네이티브 지원을 제공합니다. 자세히 알아보기 Systems Manager를 통한 엣지 디바이스 관리

AWS IoT 코어 디바이스	AWS IoT는 IoT 디바이스를 다른 디바이스에 연결하는 클라우드 서비스와 AWS 클라우드 서비스를 제공합니다. AWS IoT는 IoT 디바이스를 AWS IoT 기반 솔루션에 통합하는 데 도움이 되는 디바이스 소프트웨어를 제공합니다. 디바이스를 AWS IoT에 연결할 수 있는 경우 AWS IoT는 AWS가 제공하는 클라우드 서비스에 디바이스를 연결할 수 있습니다. Systems Manager에서는 해당 디바이스가 하이브리드 및 멀티클라우드 환경의 관리형 노드로 구성된 경우에만 AWS IoT 코어 디바이스가 지원됩니다. 자세히 알아보기 하이브리드 및 멀티클라우드 환경에서 Systems Manager 사용하기

스토리지

Amazon Simple Storage Service(S3)	Amazon S3는 인터넷 스토리지입니다. 이 서비스는 개발자가 더 쉽게 웹 규모 컴퓨팅 작업을 수행할 수 있도록 설계되었습니다. Amazon S3에서 제공하는 단순한 웹 서비스 인터페이스를 사용하여 언제든지 웹상 어디서나 원하는 양의 데이터를 저장하고 검색할 수 있습니다. Systems Manager를 사용하면 Amazon S3에 저장된 원격 스크립트 및 SSM 문서를 실행할 수 있습니다. AWS Systems Manager의 기능인 Distributor는 Amazon S3를 사용하여 패키지를 저장합니다. AWS Systems Manager의 기능인 Run Command 및 Session Manager에 대한 출력을 Amazon S3로 보낼 수도 있습니다. 자세히 알아보기 • Amazon S3에서 스크립트 실행

	• 원격 위치에서 문서 실행 • AWS Systems Manager Distributor • Amazon S3를 사용하여 세션 데이터 로깅(콘솔)

개발자 도구

AWS CodeBuild	CodeBuild는 클라우드상의 완전 관리형 빌드 서비스입니다. CodeBuild는 소스 코드를 컴파일하고 단위 테스트를 실행하며 배포 준비가 완료된 아티팩트를 생성합니다. CodeBuild에서는 자체 빌드 서버를 프로비저닝, 관리 및 확장할 필요가 없습니다. Parameter Store를 사용하면 빌드 사양 및 프로젝트에 대한 민감한 정보를 저장할 수 있습니다. 자세히 알아보기 • CodeBuild의 빌드 사양 참조 • AWS CodeBuild에서 빌드 프로젝트 생성
AWS CDK	AWS Cloud Development Kit (AWS CDK)는 프로그래밍 언어를 사용하여 클라우드 인프라를 코드로 정의하고 AWS CloudFormation을 통해 배포하는 프레임워크입니다. Application Manager에서는 Application Manager 콘솔에서 애플리케이션으로 그룹화된 CDK 구성를 보고, 기본 리소스가 포함된 애플리케이션 구조를 보고, 알림을 보고, 운영 문제를 조사 및 해결하고, 비용을 추적할 수 있습니다. 자세히 알아보기 • 애플리케이션에 대한 개요 정보 보기

	• 애플리케이션 리소스 보기

보안, 자격 증명 및 규정 준수

AWS Identity and Access Management (IAM)	IAM은 AWS 리소스에 대한 액세스를 안전하게 제어할 수 있는 웹 서비스입니다. IAM을 사용하여 리소스를 사용하도록 인증(로그인) 및 권한 부여(권한 있음)된 대상을 제어합니다. Systems Manager를 사용하면 IAM으로 서비스에 대한 액세스를 제어할 수 있습니다. 자세히 알아보기 • AWS Systems Manager에서 IAM을 사용하는 방식 • AWS Systems Manager에 사용되는 작업, 리소스 및 조건 키 • Systems Manager에 필요한 인스턴스 권한 구성
AWS Secrets Manager	Secrets Manager를 사용하면 보안 암호를 더 쉽게 관리할 수 있습니다. 보안 암호는 데이터베이스 보안 인증 정보, 암호, 타사 API 키 및 임의 텍스트가 될 수 있습니다. Parameter Store을(를) 통해 이미 Parameter Store 파라미터 참조를 지원하는 다른 AWS 서비스을(를) 사용할 때 Secrets Manager 암호를 검색할 수 있습니다. 자세히 알아보기 Parameter Store 파라미터에서 AWS Secrets Manager 암호 참조
AWS Security Hub	Security Hub에서는 우선순위가 높은 보안 알림과 AWS 계정의 규정 준수 상태를 포괄적으로

파악할 수 있습니다. Security Hub는 여러 AWS 서비스의 보안 알림 또는 결과를 집계 및 구성하고 우선순위를 지정합니다.

Security Hub와 AWS Systems Manager의 기능인 Patch Manager 간의 통합을 설정하면 Security Hub가 보안 관점에서 플릿의 패치 상태를 모니터링합니다. 패치 규정 준수 세부 정보는 자동으로 Security Hub로 내보내집니다. 이를 통해 단일 뷰를 사용하여 패치 규정 준수 상태를 중앙에서 모니터링하고 다른 보안 결과를 추적할 수 있습니다. 플릿의 노드가 패치 규정 준수를 벗어나면 알림을 수신하고 Security Hub 콘솔에서 패치 규정 준수 결과를 검토할 수 있습니다.

Security Hub를 AWS Systems Manager의 기능인 Explorer 및 OpsCenter와 통합할 수도 있습니다. Security Hub와의 통합을 통해 Explorer 및 OpsCenter에서 Security Hub의 결과를 받을 수 있습니다. Security Hub 결과는 Explorer 및 OpsCenter에서 사용할 수 있는 보안 정보를 제공하여 AWS Systems Manager의 보안, 성능 및 운영 문제를 집계하고 조치를 취합니다.

Security Hub 사용 시 요금이 부과됩니다. 자세한 내용은 Security Hub 요금을 참조하세요.

자세히 알아보기

- Explorer에서 AWS Security Hub의 결과 받기
- AWS Security Hub
- Patch Manager와 AWS Security Hub 통합

암호화 및 PKI

AWS Key Management Service (AWS KMS)	AWS KMS는 데이터 암호화에 사용하는 암호화 키인 고객 관리형 키를 생성하고 제어할 수 있게 하는 관리형 서비스입니다. Systems Manager에서는 AWS KMS를 사용하여 `SecureString` 파라미터를 생성하고 Session Manager 세션 데이터를 암호화할 수 있습니다. 자세히 알아보기 • AWS Systems ManagerParameter Store의 AWS KMS 활용 방식 • 세션 데이터의 KMS 키 암호화를 설정하려면 (콘솔)

관리 및 거버넌스

AWS CloudFormation	AWS CloudFormation은 Amazon Web Services 리소스를 모델링하고 설정하여 리소스 관리 시간을 줄이고 AWS에서 실행되는 애플리케이션에 더 많은 시간을 사용하도록 하는 서비스입니다. Parameter Store는 동적 참조의 소스입니다. 동적 참조는 AWS CloudFormation 스택 템플릿의 다른 서비스에서 저장 및 관리되는 외부 값을 지정하는 간결하면서 강력한 방법을 제공합니다. 자세히 알아보기 동적 참조를 사용하여 템플릿 값 지정
AWS CloudTrail	CloudTrail은 AWS 계정에 대한 거버넌스, 규정 준수, 운영 및 위험 감사 권한을 부여하는

	데 도움이 되는 AWS 서비스입니다. 사용자, 역할 또는 AWS 서비스이(가) 수행하는 작업은 CloudTrail에 이벤트로 기록됩니다. 이벤트에는 AWS Management Console, AWS Command Line Interface(AWS CLI) 및 AWS SDK와 API에서 수행되는 작업이 포함됩니다. Systems Manager는 대부분의 Systems Manager API 직접 호출을 이벤트로 캡처하는 CloudTrail과 통합됩니다. 여기에는 Systems Manager 콘솔의 API 호출, Systems Manager API에 대한 호출 등이 포함됩니다. 자세히 알아보기 AWS CloudTrail을 사용하여 AWS Systems ManagerAPI 호출을 로깅
Amazon CloudWatch Logs	Amazon CloudWatch Logs로 사용하는 모든 시스템, 애플리케이션 및 AWS 서비스의 로그를 중앙 집중화할 수 있습니다. 그런 다음 로그를 보고, 특정 오류 코드 또는 패턴이 있는지 검색하고, 특정 필드를 기반으로 필터링하거나, 향후 분석을 위해 안전하게 보관할 수 있습니다. Systems Manager를 사용하면 SSM Agent, Run Command 및 Session Manager에 대한 로그를 CloudWatch Logs로 전송할 수 있습니다. 자세히 알아보기 • 통합 CloudWatch Logs로 노드 로그 전송(CloudWatch 에이전트) • Run Command에 대한 Amazon CloudWatch Logs 구성 • Amazon CloudWatch Logs를 사용하여 세션 데이터 로깅(콘솔)

Amazon EventBridge	EventBridge는 Amazon Web Services 리소스의 변경 사항을 설명하는 시스템 이벤트의 스트림을 거의 실시간으로 제공합니다. 신속하게 설정할 수 있는 단순 규칙을 사용하여 일치하는 이벤트를 검색하고 하나 이상의 대상 함수 또는 스트림으로 이를 라우팅할 수 있습니다. EventBridge는 운영 변경 사항이 발생할 때 이를 인식하게 됩니다. EventBridge는 이러한 운영 변경 사항에 대응하고 필요에 따라 시정 조치를 취합니다. 이러한 작업에는 환경에 대응을 위한 메시지 전송, 기능 활성화 및 상태 정보 캡처가 포함됩니다. Systems Manager에는 EventBridge에서 지원하는 여러 이벤트가 있어 해당 이벤트의 내용을 기반으로 작업을 수행할 수 있습니다. 자세히 알아보기 Amazon EventBridge로 Systems Manager 이벤트 모니터링 **Note** Amazon EventBridge는 이벤트를 관리하는 데 선호되는 방법입니다. CloudWatch Events와 EventBridge는 기본 서비스 및 API가 동일하지만 EventBridge가 더 많은 기능을 제공합니다. CloudWatch 또는 EventBridge에서 변경한 내용은 각 콘솔에 반영됩니다. 자세한 내용은 Amazon EventBridge 사용 설명서를 참조하세요.

AWS Config	AWS Config는 AWS 계정에 있는 AWS 리소스의 구성을 자세히 보여줍니다. 여기에는 리소스가 서로 관련되는 방식과 리소스가 구성된 방식이 포함됩니다. 이를 통해 구성과 관계가 시간 경과에 따라 어떻게 변하는지 확인할 수 있습니다. Systems Manager는 AWS Config과 통합되어 EC2 인스턴스에 대한 가시성을 확보하는 데 도움이 되는 여러 규칙을 제공합니다. 이러한 규칙은 Systems Manager에서 관리하는 EC2 인스턴스, 운영 체제 구성, 시스템 수준 업데이트, 설치된 애플리케이션, 네트워크 구성 등을 식별하는 데 도움이 됩니다. 자세히 알아보기 • AWS Config 지원되는 리소스 유형 • 관리형 인스턴스에 대한 소프트웨어 구성 기록 • 인벤토리 이력 및 변경 사항 추적 보기
AWS Trusted Advisor	Trusted Advisor는 AWS 모범 사례에 따라 리소스를 제공하는 데 도움이 되는 실시간 지침을 제공하는 온라인 도구입니다. Systems Manager는 Trusted Advisor를 호스팅하고 Explorer에서 Trusted Advisor 데이터를 볼 수 있습니다. 자세히 알아보기 • AWS Systems Manager Explorer • AWS Trusted Advisor 시작하기

AWS Organizations	Organizations는 여러 AWS 계정를 사용자가 생성하고 중앙에서 관리하는 단일 조직으로 통합할 수 있는 계정 관리 서비스입니다. Organizations에는 계정 관리 및 통합 결제 기능이 포함되어 있어 비즈니스의 예산, 보안 및 규정 준수 요구 사항을 보다 잘 충족할 수 있습니다. AWS Systems Manager의 기능인 Change Manager를 Organizations와 통합하면 위임된 관리자 계정을 사용하여 이 단일 계정을 통해 전체 조직에 대한 변경 요청, 변경 템플릿 및 승인을 관리할 수 있습니다. Organizations를 AWS Systems Manager의 기능인 Inventory 및 Explorer와 통합하면 여러 AWS 리전 및 AWS 계정에서 인벤토리 및 운영 데이터(OpsData)를 집계할 수 있습니다. AWS Systems Manager의 기능인 Quick Setup과 Organizations를 통합하면 일반적인 서비스 설정 태스크가 자동화되고 조직 단위(OU) 전반에 모범 사례를 기반으로 서비스 구성이 배포됩니다.

네트워킹 및 콘텐츠 전송

AWS PrivateLink	AWS PrivateLink을(를) 통해 인터넷 게이트웨이, NAT 디바이스, VPN 연결 또는 AWS Direct Connect 연결 없이도 지원되는 AWS 서비스 및 VPC 엔드포인트 서비스에 Virtual Private Cloud(VPC)를 비공개로 연결할 수 있습니다. Systems Manager는 AWS PrivateLink를 사용하여 Systems Manager API에 연결하는 관리형 노드를 지원합니다. AWS PrivateLink는 관리형 노드, Systems Manager 및 Amazon EC2 간의 모

	든 네트워크 트래픽을 Amazon 네트워크로 제한하기 때문에 이를 통해 관리형 노드의 보안 태세가 향상됩니다. 즉, 관리형 노드가 인터넷에 액세스할 필요가 없습니다. 자세히 알아보기 Systems Manager용 VPC 엔드포인트를 사용하여 EC2 인스턴스의 보안 개선

분석

Amazon Athena	Athena는 표준 SQL을 사용하여 Amazon Simple Storage Service(Amazon S3)에 있는 데이터를 직접 간편하게 분석할 수 있는 대화형 쿼리 서비스입니다. AWS Management Console에서 몇 가지 작업을 수행하면 Amazon S3에 저장된 데이터에서 Athena를 가리키고, 표준 SQL을 사용하여 일회성 쿼리를 실행하고, 몇 초 안에 결과를 얻을 수 있습니다. Systems Manager Inventory가 Athena와 통합되어 여러 AWS 리전 및 AWS 계정에서 인벤토리 데이터를 쿼리하는 데 도움이 됩니다. Athena 통합은 리소스 데이터 동기화를 사용하므로 Systems Manager Inventory 콘솔의 세부 정보 보기 페이지에서 모든 관리형 노드의 인벤토리 데이터를 볼 수 있습니다. 자세히 알아보기 • 여러 리전 및 계정에서 인벤토리 데이터 쿼리 • 시연: 리소스 데이터 동기화를 사용하여 인벤토리 데이터 집계
AWS Glue	AWS Glue는 완전 관리형 추출, 변환 및 로드(ETL) 서비스로서 간단하고 경제적으로 데이터

	를 분류, 정리, 보강하고, 다양한 데이터 스토어와 데이터 스트림 간에 안정적으로 이동할 수 있습니다. Systems Manager는 AWS Glue를 사용하여 S3 버킷의 Inventory 데이터를 크롤링합니다. 자세히 알아보기 여러 리전 및 계정에서 인벤토리 데이터 쿼리
Amazon QuickSight	Amazon QuickSight는 데이터를 사용하여 시각적 객체를 빌드하고, 일회성 분석을 수행하고, 비즈니스 관련 인사이트를 얻을 수 있는 비즈니스 분석 서비스입니다. AWS 데이터 원본을 자동으로 검색할 수 있으며 사용자의 데이터 원본도 사용할 수 있습니다. Systems Manager 리소스 데이터 동기화는 모든 관리형 노드에서 수집된 인벤토리 데이터를 단일 S3 버킷으로 전송합니다. Amazon QuickSight를 사용하여 집계된 데이터를 쿼리하고 분석할 수 있습니다. 자세히 알아보기 • Inventory의 리소스 데이터 동기화 구성 • 시연: 리소스 데이터 동기화를 사용하여 인벤토리 데이터 집계

애플리케이션 통합

Amazon Simple Notification Service(Amazon SNS)	Amazon SNS는 구독 중인 엔드포인트 또는 클라이언트에 메시지 전달 또는 전송을 조정 및 관리하는 웹 서비스입니다.

	Systems Manager는 Amazon SNS 알림으로 캡처할 수 있는 여러 서비스에 대한 상태를 생성합니다. 자세히 알아보기 • Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 • Parameter Store 이벤트 기반 알림 설정 또는 작업 트리거

AWS Management Console

AWS Resource Groups	Resource Groups는 AWS 리소스를 구성합니다. 리소스 그룹을 사용하여 많은 리소스에 대한 작업을 한 번에 관리, 모니터링 및 자동화할 수 있습니다. 관리형 노드, SSM 문서, 유지 관리 기간, Parameter Store 파라미터 및 패치 기준과 같은 Systems Manager 리소스 유형을 Resource Groups에 추가할 수 있습니다. 자세히 알아보기 AWS Resource Groups란 무엇입니까?

주제

- Amazon S3에서 스크립트 실행
- Parameter Store 파라미터에서 AWS Secrets Manager 암호 참조
- AWS Lambda 함수에서 Parameter Store 파라미터 사용

Amazon S3에서 스크립트 실행

이 섹션에서는 Amazon Simple Storage Service(Amazon S3)에서 스크립트를 다운로드하고 실행하는 방법을 설명합니다. 다음 주제에는 Amazon S3와 관련된 정보와 용어가 포함되어 있습니다. Amazon S3에 대한 자세한 내용은 Amazon S3 설명서를 참조하세요. Ansible Playbooks, Python, Ruby, PowerShell을 포함한 다양한 유형의 스크립트를 실행할 수 있습니다.

여러 스크립트가 포함된 디렉터리를 다운로드할 수도 있습니다. 디렉터리에서 기본 스크립트를 실행할 때 AWS Systems Manager는 디렉터리에 포함된 참조되는 스크립트도 모두 실행합니다.

다음은 Amazon S3로부터 스크립트 실행에 대한 중요 세부 정보입니다.

- Systems Manager는 스크립트가 노드에서 실행 가능한지 확인하지 않습니다. 스크립트를 다운로드하여 실행하기 전에 노드에 필요한 소프트웨어가 설치되어 있는지 확인합니다. 아니면 AWS Systems Manager의 기능인 Run Command 또는 State Manager를 사용하여 소프트웨어를 설치한 다음, 스크립트를 다운로드해 실행하는 복합 문서를 생성할 수 있습니다.
- 사용자, 역할 또는 그룹에 S3 버킷에서 읽는 데 필요한 AWS Identity and Access Management(IAM) 권한이 부여되었는지 확인합니다.
- Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스의 인스턴스 프로파일에 `s3:ListBucket` 및 `s3:GetObject` 권한이 있는지 확인합니다. 인스턴스 프로파일에 이러한 권한이 없으면 시스템은 S3 버킷에서 스크립트를 다운로드하지 못합니다. 자세한 내용은 IAM User Guide의 Using instance profiles를 참조하세요.

Amazon S3에서 셸 스크립트 실행

다음 정보에는 AWS Systems Manager 콘솔 또는 AWS Command Line Interface(AWS CLI)를 사용하여 Amazon Simple Storage Service(Amazon S3)에서 Ruby 스크립트를 실행할 수 있는 절차가 포함되어 있습니다. 이 예시에서는 셸 스크립트가 사용되지만 다른 유형의 스크립트로 대체할 수 있습니다.

Amazon S3에서 셸 스크립트 실행(콘솔)

Amazon S3에서 셸 스크립트 실행

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택합니다.
3. Run command를(Run 명령) 선택합니다.
4. [Command 문서(Command document)] 목록에서 **`AWS-RunRemoteScript`**를 선택합니다.
5. 명령 파라미터에서 다음을 수행합니다.

- 소스 유형에서 S3를 선택합니다.
- [소스 정보(Source Info)] 텍스트 상자에 소스에 액세스하는 데 필요한 정보를 다음 형식으로 입력합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

> **Note**
>
> https://s3.*aws-api-domain*을 버킷 URL로 교체합니다. Objects(객체) 탭에서 Amazon S3의 버킷 URL을 복사할 수 있습니다.

```
{"path":"https://s3.aws-api-domain/path to script"}
```

다음은 예입니다.

```
{"path":"https://DOC-EXAMPLE-BUCKET.s3.us-west-2.amazonaws.com/scripts/shell/
helloWorld.sh"}
```

- [명령줄(Command Line)] 필드에 스크립트 실행을 위한 파라미터를 입력합니다. 다음 예를 참고하세요

```
helloWorld.sh argument-1 argument-2
```

- (옵션) 작업 디렉터리(Working Directory) 필드에 스크립트를 다운로드하여 실행하려는 노드에 있는 디렉터리의 이름을 입력합니다.
- (선택 사항) Execution Timeout(실행 제한 시간)에서 스크립트 명령 실행이 실패할 때까지 시스템이 기다리는 시간(단위: 초)을 지정합니다.

6. 대상 섹션에서, 태그를 지정하거나, 수동으로 인스턴스나 엣지 디바이스를 선택하거나, 리소스 그룹을 지정하여 이 작업을 실행할 관리형 노드를 식별합니다.

> **Tip**
>
> 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

7. Other parameters(다른 파라미터):

- Comment(설명)에 명령에 대한 정보를 입력합니다.

- 제한 시간(초)에서 전체 명령 실행이 실패할 때까지 시스템이 기다리는 시간을 초 단위로 지정합니다.

8. Rate control(속도 제어)에서

   - Concurrency(동시성)에서 명령을 동시에 실행할 관리형 노드의 백분율 또는 개수를 지정합니다.

     > **Note**
     >
     > 관리형 노드에 적용할 태그를 지정하거나, AWS 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상 수를 제한합니다.

   - Error threshold(오류 임계값)에서, 명령이 노드의 개수 또는 백분율에서 실패한 후 다른 관리형 노드에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째 오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.

9. (선택 사항) Output options(출력 옵션)에서 명령 출력을 파일에 저장하려면 Write command output to an S3 bucket(S3 버킷에 명령 출력 쓰기) 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

   > **Note**
   >
   > 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아니라 인스턴스에 할당된 인스턴스 프로파일(EC2 인스턴스용) 또는 IAM 서비스 역할(하이브리드 정품 인증 시스템)의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

10. SNS notifications(SNS 알림) 섹션에서, 명령 실행 상태에 대한 알림이 전송되도록 하려면 Enable SNS notifications(SNS 알림 활성화) 확인란을 선택합니다.

    Run Command에 대한 Amazon SNS 알림 구성에 대한 자세한 내용은 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 섹션을 참조하세요.

11. Run(실행)을 선택합니다.

Amazon S3에서 셸 스크립트 실행(명령줄)

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.

2. 다음 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

> **Note**
>
> https://s3.*aws-api-domain*을 버킷 URL로 교체합니다. Objects(객체) 탭에서 Amazon S3의 버킷 URL을 복사할 수 있습니다.

Linux & macOS

```
aws ssm send-command \
    --document-name "AWS-RunRemoteScript" \
    --output-s3-bucket-name "bucket-name" \
    --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 "bucket-name" ^
    --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 "bucket-name" `
    -OutputS3KeyPrefix "key-prefix" `
    -Target @{Key="InstanceIds";Values=@("instance-id")} `
```

```
    -Parameter @{ sourceType="S3";sourceInfo='{"path": "https://s3.aws-api-
domain/script path"}',; "commandLine"="script name and arguments"}
```

Parameter Store 파라미터에서 AWS Secrets Manager 암호 참조

AWS Secrets Manager을(를) 사용하면 자격 증명, 암호 및 라이선스 키 등과 같이 중요한 구성 데이터를 정리 및 관리할 수 있습니다. AWS Systems Manager의 기능인 Parameter Store은(는) 이제 Secrets Manager에 통합되어 Parameter Store 파라미터에 대한 참조를 이미 지원하는 다른 AWS 서비스을(를) 사용하는 경우 Secrets Manager 암호를 가져올 수 있습니다. 이러한 서비스에는 Amazon Elastic Compute Cloud(Amazon EC2), Amazon Elastic Container Service(Amazon ECS), AWS Lambda, AWS CloudFormation, AWS CodeBuild, AWS CodeDeploy 및 기타 Systems Manager 기능이 포함됩니다. Parameter Store를 통해 Secrets Manager 암호를 참조하면 코드 및 구성 스크립트에서 일관성 있고 안전하게 암호 및 참조 데이터를 호출하고 사용할 수 있는 프로세스가 마련됩니다.

Secrets Manager에 대한 자세한 내용은 AWS Secrets Manager User Guide의 What Is AWS Secrets Manager?를 참조하세요.

제한 사항

Parameter Store를 사용하여 Secrets Manager 암호를 참조할 때는 다음과 같은 제한 사항이 있습니다.

- GetParameter 및 GetParameters API 작업을 사용하여 Secrets Manager 보안 암호만 검색할 수 있습니다. DescribeParameters, GetParametersByPath 등의 고급 쿼리 API 작업과 수정 작업은 Secrets Manager에 사용할 수 없습니다.
- AWS Command Line Interface(AWS CLI), AWS Tools for Windows PowerShell 및 SDK에서 Parameter Store를 사용하여 암호를 검색할 수 있습니다.
- Parameter Store에서 Secrets Manager 암호를 검색하는 경우 이름이 예약된 경로인 /aws/reference/secretsmanager/*secret-_ID*로 시작되어야 합니다.

  단순 예시: `/aws/reference/secretsmanager/CFCreds1`
- Parameter Store는 Secrets Manager 보안 암호에 연결된 AWS Identity and Access Management(IAM) 정책을 준수합니다. 예를 들어 사용자 1에게 암호 A에 대한 액세스 권한이 없으면 사용자 1은 Parameter Store를 사용하여 암호 A를 검색할 수 없습니다.
- Secrets Manager 암호를 참조하는 파라미터는 Parameter Store의 버전 관리 기능이나 기록 기능을 사용할 수 없습니다.

- Parameter Store는 Secrets Manager 버전 단계를 준수합니다. 버전 단계를 참조할 때는 문자, 숫자, 마침표(.), 하이픈(-) 또는 밑줄(_)을 사용합니다. 버전 단계에 그 밖의 모든 기호를 지정하면 참조가 실패하게 됩니다.

Parameter Store를 사용하여 Secrets Manager 보안 암호를 참조하는 방법

다음 절차에서는 Parameter Store API를 사용하여 Secrets Manager 암호를 참조하는 방법을 설명합니다. 여기서는 AWS Secrets Manager User Guide의 다른 절차를 참조합니다.

> **Note**
>
> 시작하기 전에 Parameter Store 파라미터에서 Secrets Manager 암호를 참조할 권한이 있는지 확인합니다. Secrets Manager 및 Systems Manager에서 관리자 권한이 있는 경우 Parameter Store API를 사용하여 보안 암호를 참조하거나 검색할 수 있습니다. Parameter Store 파라미터에 Secrets Manager 암호를 참조했는데 그 암호에 액세스할 권한이 없으면 참조가 실패합니다. 자세한 내용은 AWS Secrets Manager User Guide의 Authentication and access control for AWS Secrets Manager를 참조하세요.

> **Important**
>
> Parameter Store는 Secrets Manager 암호 참조를 위한 패스스루 서비스의 역할을 합니다. Parameter Store는 암호에 대한 데이터 또는 메타데이터를 보존하지 않습니다. 참조는 상태 비저장입니다.

Parameter Store를 사용하여 Secrets Manager 보안 암호를 참조하려면

1. Secrets Manager에서 보안 암호를 생성합니다. 자세한 내용은 AWS Secrets Manager로 암호 생성 및 관리를 참조하세요.
2. AWS CLI, AWS Tools for Windows PowerShell 또는 SDK를 사용하여 암호를 참조합니다. Secrets Manager 암호를 참조할 때는 이름이 예약된 경로인 `/aws/reference/secretsmanager/`로 시작되어야 합니다. 이 경로를 지정하면 Parameter Store가 아니라 Secrets Manager에서 암호를 검색해야 한다는 것을 Systems Manager에서 알게 됩니다. 다음은 Parameter Store를 사용하여 Secrets Manager 보안 암호 `CFCreds1` 및 `DBPass`를 올바르게 참조하는 몇 가지 예제 이름입니다.

   - `/aws/reference/secretsmanager/CFCreds1`

- `/aws/reference/secretsmanager/DBPass`

다음은 Secrets Manager에 저장된 액세스 키와 보안 키를 참조하는 Java 코드 예제입니다. 이 코드 예제는 Amazon DynamoDB 클라이언트를 설정합니다. 이 코드는 Parameter Store에서 구성 데이터와 자격 증명을 검색합니다. 구성 데이터는 Parameter Store에 문자열 파라미터로 저장되고, 자격 증명은 Secrets Manager에 저장됩니다. 구성 데이터와 자격 증명이 별도의 서비스에 저장되지만 두 데이터 집합 모두 Parameter Store에서 `GetParameter` API를 사용하여 액세스할 수 있습니다.

```
/**
* 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();
}
```

여기 몇 가지 AWS CLI 예제가 있습니다. `aws secretsmanager list-secrets` 명령을 사용하여 보안 암호의 이름을 찾습니다.

AWS CLI 예제 1: 암호의 이름을 사용하여 참조

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
```

명령은 다음과 같은 정보를 반환합니다.

```
{
    "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 예제 2: 버전 ID를 포함하는 참조

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
```

명령은 다음과 같은 정보를 반환합니다.

```
{
    "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 예제 3: 버전 단계를 포함하는 참조

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
```

명령은 다음과 같은 정보를 반환합니다.

```
{
    "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 Lambda 함수에서 Parameter Store 파라미터 사용

AWS Systems Manager의 기능인 Parameter Store는 구성 데이터 관리 및 암호 관리를 위한 안전한 계층적 스토리지를 제공합니다. 암호, 데이터베이스 문자열, Amazon Machine Image(AMI) ID, 라이선스 코드와 같은 데이터를 파라미터 값으로 저장할 수 있습니다.

SDK를 사용하지 않고 AWS Lambda 함수에서 Parameter Store의 파라미터를 사용하려면 AWS 파라미터 및 보안 Lambda 익스텐션을 사용할 수 있습니다. 이 익스텐션은 파라미터 값을 검색한 후 나중에 사용할 수 있도록 캐시합니다. Lambda 익스텐션을 사용하면 Parameter Store에 대한 API 호출 수를 줄여 비용을 절감할 수 있습니다. 익스텐션을 사용하면 Parameter Store에서 검색할 때보다 캐시된 파라미터를 더 빠르게 검색할 수 있으므로 지연 시간을 줄일 수 있습니다.

Lambda 익스텐션은 Lambda 함수의 기능을 강화하는 동반 프로세스입니다. 익스텐션은 Lambda 호출과 병렬로 실행되는 클라이언트와 같습니다. 이 병렬 클라이언트는 수명 주기 중 언제든지 함수와 연결될 수 있습니다. Lambda 익스텐션에 대한 자세한 내용은 AWS Lambda 개발자 안내서의 Lambda 익스텐션 API를 참조하세요.

AWS 파라미터 및 보안 Lambda 익스텐션은 Parameter Store 및 AWS Secrets Manager 모두에서 작동합니다. Lambda 익스텐션을 Secrets Manager의 보안 암호와 함께 사용하는 방법은 AWS Secrets Manager사용자 안내서의 AWS Lambda 함수에 AWS Secrets Manager 보안 암호 사용을 참조하세요.

관련 정보

AWS 파라미터 및 Secrets Lambda 확장을 사용하여 파라미터와 보안 암호 캐싱(AWS 컴퓨팅 블로그)

익스텐션 작동 방법

Lambda 익스텐션 없이 Lambda 함수에서 파라미터를 사용하려면 Parameter Store의 `GetParameter` API 작업과 통합하여 구성 업데이트를 수신하도록 Lambda 함수를 구성해야 합니다.

AWS 파라미터 및 보안 Lambda 익스텐션을 사용하는 경우 익스텐션은 Parameter Store에서 파라미터 값을 검색한 후 로컬 캐시에 저장합니다. 그런 다음 캐시된 값은 만료될 때까지 이후 호출에 사용됩니다. 캐시된 값은 TTL(time-to-live)이 경과한 이후에 만료됩니다. 이 항목의 뒷부분에 설명된 대로 `SSM_PARAMETER_STORE_TTL` 환경 변수를 사용하여 TTL 값을 구성할 수 있습니다.

구성된 캐시 TTL이 만료되지 않은 경우 캐시된 파라미터 값이 사용됩니다. 시간이 만료된 경우 캐시된 값은 무효화되고 파라미터 값이 Parameter Store에서 검색됩니다.

또한 자주 사용되는 파라미터 값을 감지하여 캐시에서 유지하고 만료되거나 사용되지 않는 값은 지웁니다.

구현 세부 정보

다음 세부 정보를 사용하여 AWS 파라미터 및 보안 Lambda 익스텐션을 구성할 수 있습니다.

인증

Parameter Store 요청을 승인하고 인증하기 위해 익스텐션은 Lambda 함수를 실행하는 데 사용된 것과 동일한 보안 인증을 사용합니다. 따라서 함수를 실행하는 데 사용되는AWS Identity and Access Management(IAM) 역할에 다음과 같은 권한이 있어야 Parameter Store와 상호 작용할 수 있습니다.

- `ssm:GetParameter` - Parameter Store에서 파라미터를 검색하는 데 필요합니다.
- `kms:Decrypt` – Parameter Store에서 `SecureString` 파라미터를 검색하는 경우에 필요합니다.

자세한 내용은 AWS Lambda 개발자 가이드의 AWS Lambda 실행 역할을 참조하세요.

인스턴스화

Lambda는 함수에 필요한 동시성 레벨에 해당하는 별도의 인스턴스를 인스턴스화합니다. 각 인스턴스는 격리되며 구성 데이터의 자체 로컬 캐시를 유지합니다. Lambda 인스턴스 및 동시성에 대한 자세한 내용은 AWS Lambda 개발자 안내서의 예약된 동시성 구성을 참조하세요.

SDK에 종속되지 않음

AWS 파라미터 및 보안 Lambda 익스텐션은 AWS SDK 언어 라이브러리와 독립적으로 작동합니다. Parameter Store에 대한 GET 요청을 생성하는 데 AWS SDK가 필요하지 않습니다.

Localhost port

GET 요청에는 `localhost`를 사용합니다. 이 익스텐션은 localhost 포트 2773에 요청합니다. 익스텐션을 사용하기 위해 외부 또는 내부 엔드포인트를 지정할 필요가 없습니다. 환경 변수 `PARAMETERS_SECRETS_EXTENSION_HTTP_PORT`를 설정하여 포트를 구성할 수 있습니다.

예를 들어 Python에서 GET URL은 다음 예제와 비슷합니다.

```
parameter_url = ('http://localhost:' + port + '/systemsmanager/parameters/get/?
name=' + ssm_parameter_path)
```

TTL 만료 전 파라미터 값 변경

익스텐션은 파라미터 값에 대한 변경 사항을 감지하지 못하며 TTL이 만료되기 전에 자동 새로 고침을 수행하지 않습니다. 파라미터 값을 변경하면 캐시된 파라미터 값을 사용하는 작업이 다음에 캐시를 새로 고칠 때까지 실패할 수 있습니다. 파라미터 값을 자주 변경할 경우 TTL 값을 더 짧게 설정하는 것이 좋습니다.

헤더 요구 사항

익스텐션 캐시에서 파라미터를 검색하려면 GET 요청의 헤더에 `X-Aws-Parameters-Secrets-Token` 참조가 포함되어야 합니다. 모든 실행 중인 함수에 대해 Lambda에서 제공하는 `AWS_SESSION_TOKEN`으로 토큰을 설정합니다. 이 헤더를 사용하면 호출자가 Lambda 환경 내에 있음을 나타냅니다.

예

Python의 다음 예제는 캐시된 파라미터 값을 검색하기 위한 기본 요청을 보여줍니다.

```
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 지원

확장에서는 x86_64 및 x86 아키텍처가 지원되는 모든 동일한 AWS 리전의 ARM 아키텍처를 지원하지 않습니다.

익스텐션 ARN의 전체 목록은 AWS 파라미터 및 보안 암호 Lambda 익스텐션 ARN 단원을 참조하세요.

로깅

Lambda는 Amazon CloudWatch Logs를 사용하여 익스텐션에 대한 실행 정보를 함수와 함께 기록합니다. 기본적으로 이 익스텐션은 CloudWatch에 최소한의 정보를 기록합니다. 세부 정보를 기록하려면 환경 변수 PARAMETERS_SECRETS_EXTENSION_LOG_LEVEL을 DEBUG로 설정합니다.

Lambda 함수에 익스텐션 추가

AWS 파라미터 및 보안 Lambda 익스텐션을 사용하려면 Lambda 함수에 익스텐션을 계층으로 추가합니다.

다음 중 한 가지 방법으로 함수에 익스텐션을 추가합니다.

AWS Management Console(계층 추가 옵션)

1. https://console.aws.amazon.com/lambda/에서 AWS Lambda 콘솔을 엽니다.
2. 함수를 선택합니다. Layers(계층) 영역에서 Add a layer(계층 추가)를 선택합니다.
3. 계층 선택 영역에서 AWS 계층 옵션을 선택합니다.
4. AWS 계층의 경우 AWS-Parameters-and-Secrets-Lambda-Extension을 선택하고, 버전을 선택한 다음에 추가를 선택합니다.

AWS Management Console(ARN 옵션 지정)

1. https://console.aws.amazon.com/lambda/에서 AWS Lambda 콘솔을 엽니다.
2. 함수를 선택합니다. Layers(계층) 영역에서 Add a layer(계층 추가)를 선택합니다.
3. Choose a layer(계층 선택) 영역에서 Specify an ARN(ARN 지정) 옵션을 선택합니다.
4. Specify an ARN(ARN 지정)에서 AWS 리전 및 아키텍처에 대한 익스텐션 ARN을 입력한 다음 Add(추가)를 선택합니다.

AWS Command Line Interface

AWS CLI에서 다음과 같은 명령을 실행합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
aws lambda update-function-configuration \
    --function-name function-name \
    --layers layer-ARN
```

관련 정보

Lambda 함수에서 계층 사용

익스텐션 구성(.zip 파일 아카이브)

AWS 파라미터 및 보안 암호 Lambda 익스텐션 환경 변수

다음 환경 변수를 변경하여 익스텐션을 구성할 수 있습니다. 현재 설정을 보려면 PARAMETERS_SECRETS_EXTENSION_LOG_LEVEL을 DEBUG로 설정합니다. 자세한 내용은 AWS Lambda 개발자 안내서의 AWS Lambda 환경 변수 사용을 참조하세요.

> **Note**
>
> AWS Lambda는 Lambda 익스텐션 및 Lambda 함수에 대한 작업 세부 정보를 Amazon CloudWatch Logs에 기록합니다.

환경 변수	Details	필수	유효값	기본값
SSM_PARAMETER_STORE_TIMEOUT_MILLIS	Parameter Store에 대한 요청 제한 시간(밀리초)입니다. 0 값은 시간 제한이 없는 것을 나타냅니다.	아니요	모든 정수	0(제로)
SECRETS_MANAGER_TIMEOUT_MILLIS	Secrets Manager에 대한 요청 제한 시간(밀리초)입니다. 값이 0이면 시간 제한이 없습니다.	아니요	모든 정수	0(제로)
SSM_PARAMETER_STORE_TTL	캐시에서 파라미터가 무효화되기 전의 최대 유효 기간(초)입니	아니요	0~300초(5분)	300초(5분)

환경 변수	Details	필수	유효값	기본값
	다. 값이 0이면 캐시가 우회됩니다. PARAMETERS_SECRETS_EXTENSION_CACHE_SIZE 에 대한 값이 0이면 이 변수는 무시됩니다.			
SECRETS_MANAGER_TTL	캐시에서 암호가 무효화되기 전의 최대 유효 기간(초)입니다. 값이 0이면 캐시가 우회됩니다. PARAMETERS_SECRETS_EXTENSION_CACHE_SIZE 에 대한 값이 0이면 이 변수는 무시됩니다.	아니요	0~300초(5분)	300초(5분)
PARAMETERS_SECRETS_EXTENSION_CACHE_ENABLED	확장에 대한 캐시가 활성화되었는지 여부를 결정합니다. 유효한 값: TRUE | FALSE	아니요	TRUE | FALSE	TRUE

환경 변수	Details	필수	유효값	기본값
PARAMETERS_SECRETS_EXTENSION_CACHE_SIZE	항목 수를 기준으로 한 캐시의 최대 크기입니다. 값이 0이면 캐시가 우회됩니다. 두 캐시 TTL 값이 모두 0인 경우 이 변수는 무시됩니다.	아니요	0~1000	1000
PARAMETERS_SECRETS_EXTENSION_HTTP_PORT	로컬 HTTP 서버의 포트입니다.	아니요	1~65535	2773
PARAMETERS_SECRETS_EXTENSION_MAX_CONNECTIONS	익스텐션에서 Parameter Store 또는 Secrets Manager에 요청하는 데 사용되는 HTTP 클라이언트의 최대 연결 수입니다. Secrets Manager 클라이언트와 Parameter Store 클라이언트가 모두 백엔드 서비스에 연결하는 개수에 대한 클라이언트별 구성입니다.	아니요	최소값은 1이고 최대 제한은 없습니다.	3

환경 변수	Details	필수	유효값	기본값
PARAMETERS_SECRETS_EXTENSION_LOG_LEVEL	익스텐션에 대해 로그에 보고된 세부 정보 수준입니다. 익스텐션을 설정하고 테스트할 때 캐시 구성에 대한 자세한 내용을 보려면 DEBUG를 사용하는 것이 좋습니다. Lambda 작업에 대한 로그는 연결된 CloudWatch Logs 로그 그룹에 자동으로 푸시됩니다.	아니요	DEBUG | WARN | ERROR | NONE | INFO	INFO

AWS Systems Manager Parameter Store 및 AWS Secrets Manager 익스텐션 사용을 위한 샘플 명령

이 단원의 예제는 AWS Systems Manager Parameter Store 및 AWS Secrets Manager 익스텐션에 사용하기 위한 API 작업을 보여줍니다.

Parameter Store에 대한 샘플 명령

Lambda 익스텐션은 GetParameter API 작업에 대한 읽기 전용 액세스를 사용합니다.

이 작업을 호출하려면 다음과 비슷한 HTTP GET 호출을 수행합니다.

```
GET http://localhost:port/systemsmanager/parameters/get?name=parameter-path&version=version&label=label&withDecryption={true|false}
```

이 예에서 *parameter-path*는 전체 파라미터 이름을 나타냅니다. *version* 및 *label*는 GetParameter 작업에 사용할 수 있는 선택기입니다. 이 명령 형식은 표준 파라미터 계층의 파라미터에 대한 액세스를 제공합니다.

> **Note**
>
> GET 호출을 사용하는 경우 특수 문자를 보존하려면 파라미터 값을 HTTP용으로 인코딩해야 합니다. 예를 들어 계층적 경로 형식(예: /a/b/c)을 지정하는 대신 URL의 일부로 해석될 수 있는 문자(예: %2Fa%2Fb%2Fc)를 인코딩합니다.

```
GET http://localhost:port/systemsmanager/parameters/get/?name=MyParameter&version=5
```

계층 구조로 파라미터를 호출하려면 HTTP GET을 다음과 비슷하게 호출합니다.

```
GET http://localhost:port/systemsmanager/parameters/get?name=%2Fa%2Fb%2F&label=release
```

퍼블릭(글로벌) 파라미터를 호출하려면 HTTP GET을 다음과 비슷하게 호출합니다.

```
GET http://localhost:port/systemsmanager/parameters/get/?name=%2Faws%2Fservice%20list
%2F…
```

Parameter Store 참조를 사용하여 Secrets Manager 암호에 대한 HTTP GET 호출을 생성하려면 HTTP GET을 다음과 비슷하게 호출합니다.

```
GET http://localhost:port/systemsmanager/parameters/get?name=%2Faws%2Freference
%2Fsecretsmanager%2F…
```

파라미터에 대해 Amazon Resource Name(ARN)을 사용하여 호출하려면 HTTP GET을 다음과 비슷하게 호출합니다.

```
GET http://localhost:port/systemsmanager/parameters/get?name=arn:aws:ssm:us-
east-1:123456789012:parameter/MyParameter
```

암호 해독을 통해 SecureString 파라미터에 액세스하는 호출을 실행하려면 HTTP GET을 다음과 비슷하게 호출합니다.

```
GET http://localhost:port/systemsmanager/parameters/get?
name=MyParameter&withDecryption=true
```

`withDecryption`을 생략하거나 `false`로 명시적으로 설정하여 파라미터가 해독되지 않도록 지정할 수 있습니다. 버전, 레이블 또는 둘 모두를 지정할 수도 있습니다. 그러면 URL에서 물음표(?) 뒤에 있는 첫 번째 항목만 사용됩니다.

AWS 파라미터 및 보안 암호 Lambda 익스텐션 ARN

다음 표는 지원되는 아키텍처 및 리전에 대한 익스텐션 ARN을 제공합니다.

주제

- x86_64 및 x86 아키텍처용 익스텐션 ARN
- ARM64 및 Mac with Apple silicon 아키텍처용 익스텐션 ARN

x86_64 및 x86 아키텍처용 익스텐션 ARN

리전	ARN
미국 동부(오하이오)	arn:aws:lambda:us-east-2:590474943231:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11
미국 동부(버지니아 북부)	arn:aws:lambda:us-east-1:177933569100:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11
미국 서부(캘리포니아 북부)	arn:aws:lambda:us-west-1:997803712105:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11
미국 서부(오레곤)	arn:aws:lambda:us-west-2:345057560386:layer:AWS-Parame

리전	ARN
	ters-and-Secrets-Lambda-Ext ension:11
아프리카(케이프타운)	arn:aws:lambda:af-south-1:3 17013901791:layer:AWS-Param eters-and-Secrets-Lambda-Ex tension:11
아시아 태평양(홍콩)	arn:aws:lambda:ap-east-1:76 8336418462:layer:AWS-Parame ters-and-Secrets-Lambda-Ext ension:11
아시아 태평양(하이데라바드) 리전	arn:aws:lambda:ap-south-2:0 70087711984:layer:AWS-Param eters-and-Secrets-Lambda-Ex tension:8
아시아 태평양(자카르타)	arn:aws:lambda:ap-southeast -3:490737872127:layer:AWS-P arameters-and-Secrets-Lambda- Extension:11
아시아 태평양(멜버른)	arn:aws:lambda:ap-southeast -4:090732460067:layer:AWS-P arameters-and-Secrets-Lambda- Extension:1
아시아 태평양(뭄바이)	arn:aws:lambda:ap-south-1:1 76022468876:layer:AWS-Param eters-and-Secrets-Lambda-Ex tension:11

리전	ARN
아시아 태평양(오사카)	arn:aws:lambda:ap-northeast-3:576959938190:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11
아시아 태평양(서울)	arn:aws:lambda:ap-northeast-2:738900069198:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11
아시아 태평양(싱가포르)	arn:aws:lambda:ap-southeast-1:044395824272:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11
아시아 태평양(시드니)	arn:aws:lambda:ap-southeast-2:665172237481:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11
아시아 태평양(도쿄)	arn:aws:lambda:ap-northeast-1:133490724326:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11
캐나다(중부)	arn:aws:lambda:ca-central-1:200266452380:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11
캐나다 서부(캘거리)	arn:aws:lambda:ca-west-1:243964427225:layer:AWS-Parameters-and-Secrets-Lambda-Extension:1

리전	ARN
중국(베이징)	arn:aws-cn:lambda:cn-north-1:287114880934:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11
중국(닝샤)	arn:aws-cn:lambda:cn-northwest-1:287310001119:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11
유럽(프랑크푸르트)	arn:aws:lambda:eu-central-1:187925254637:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11
유럽(아일랜드)	arn:aws:lambda:eu-west-1:015030872274:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11
유럽(런던)	arn:aws:lambda:eu-west-2:133256977650:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11
유럽(밀라노)	arn:aws:lambda:eu-south-1:325218067255:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11
유럽(파리)	arn:aws:lambda:eu-west-3:780235371811:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11

리전	ARN
유럽(스페인) 리전	arn:aws:lambda:eu-south-2:524103009944:layer:AWS-Parameters-and-Secrets-Lambda-Extension:8
유럽(스톡홀름)	arn:aws:lambda:eu-north-1:427196147048:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11
이스라엘(텔아비브)	arn:aws:lambda:il-central-1:148806536434:layer:AWS-Parameters-and-Secrets-Lambda-Extension:1
유럽(취리히) 리전	arn:aws:lambda:eu-central-2:772501565639:layer:AWS-Parameters-and-Secrets-Lambda-Extension:8
중동(바레인)	arn:aws:lambda:me-south-1:832021897121:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11
중동(UAE)	arn:aws:lambda:me-central-1:858974508948:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11
남아메리카(상파울루)	arn:aws:lambda:sa-east-1:933737806257:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11

리전	ARN
AWS GovCloud(미국 동부)	`arn:aws-us-gov:lambda:us-gov-east-1:129776340158:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11`
AWS GovCloud(미국 서부)	`arn:aws-us-gov:lambda:us-gov-west-1:127562683043:layer:AWS-Parameters-and-Secrets-Lambda-Extension:11`

ARM64 및 Mac with Apple silicon 아키텍처용 익스텐션 ARN

리전	ARN
미국 동부(오하이오)	`arn:aws:lambda:us-east-2:590474943231:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:11`
미국 동부(버지니아 북부)	`arn:aws:lambda:us-east-1:177933569100:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:11`
US West (N. California) Region	`arn:aws:lambda:us-west-1:997803712105:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:8`
미국 서부(오레곤)	`arn:aws:lambda:us-west-2:345057560386:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:11`

리전	ARN
아프리카(케이프타운) 리전	arn:aws:lambda:af-south-1:317013901791:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:8
Asia Pacific (Hong Kong) Region	arn:aws:lambda:ap-east-1:768336418462:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:8
Asia Pacific (Jakarta) Region	arn:aws:lambda:ap-southeast-3:490737872127:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:8
아시아 태평양(뭄바이)	arn:aws:lambda:ap-south-1:176022468876:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:11
아시아 태평양(오사카)	arn:aws:lambda:ap-northeast-3:576959938190:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:8
Asia Pacific (Seoul) Region	arn:aws:lambda:ap-northeast-2:738900069198:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:8
아시아 태평양(싱가포르)	arn:aws:lambda:ap-southeast-1:044395824272:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:11

리전	ARN
아시아 태평양(시드니)	arn:aws:lambda:ap-southeast-2:665172237481:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:11
아시아 태평양(도쿄)	arn:aws:lambda:ap-northeast-1:133490724326:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:11
캐나다(중부) 리전	arn:aws:lambda:ca-central-1:200266452380:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:8
유럽(프랑크푸르트)	arn:aws:lambda:eu-central-1:187925254637:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:11
유럽(아일랜드)	arn:aws:lambda:eu-west-1:015030872274:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:11
유럽(런던)	arn:aws:lambda:eu-west-2:133256977650:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:11
Europe (Milan) Region	arn:aws:lambda:eu-south-1:325218067255:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:8

리전	ARN
Europe (Paris) Region	`arn:aws:lambda:eu-west-3:780235371811:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:8`
유럽(스톡홀름) 리전	`arn:aws:lambda:eu-north-1:427196147048:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:8`
Middle East (Bahrain) Region	`arn:aws:lambda:me-south-1:832021897121:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:8`
South America (São Paulo) Region	`arn:aws:lambda:sa-east-1:933737806257:layer:AWS-Parameters-and-Secrets-Lambda-Extension-Arm64:8`

다른 제품 및 서비스와 통합

AWS Systems Manager는 다음 표의 제품 및 서비스와 기본적으로 통합되어 있습니다.

Ansible	Ansible은 애플리케이션과 시스템을 더 쉽게 배포할 수 있게 하는 IT 자동화 플랫폼입니다. Systems Manager는 Ansible 플레이북을 실행하는 State Manager 연결을 생성할 수 있는 Systems Manager 문서(SSM 문서) `AWS-ApplyAnsiblePlaybooks` 를 제공합니다. 자세히 알아보기 시연: Ansible 플레이북을 실행하는 연결 생성

Chef	Chef는 애플리케이션과 시스템을 더 쉽게 배포할 수 있게 하는 IT 자동화 도구입니다. Systems Manager는 Chef 레시피를 실행하는 AWS Systems Manager의 기능인 State Manager에 연결을 생성할 수 있는 `AWS-ApplyChefRecipes` SSM 문서를 제공합니다. 자세히 알아보기 시연: Chef 레시피를 실행하는 연결 생성 Systems Manager는 또한 Chef InSpec 프로파일과 통합되어 규정 준수 스캔을 실행하고 규정 준수 및 비준수 노드를 볼 수 있습니다. 자세히 알아보기 Systems Manager Compliance와 함께 Chef InSpec 프로파일 사용
GitHub	GitHub는 소프트웨어 개발 버전 제어 및 협업을 위한 호스팅을 제공합니다. Systems Manager는 GitHub에 저장된 다른 SSM 문서를 실행할 수 있는 SSM 문서 `AWS-RunDocument` 와 GitHub에 저장된 스크립트를 실행할 수 있는 SSM 문서 `AWS-RunRemoteScript` 를 제공합니다. 자세히 알아보기 • 원격 위치에서 문서 실행 • GitHub에서 스크립트 실행

Jenkins	Jenkins는 개발자가 소프트웨어를 안정적으로 빌드, 테스트 및 배포할 수 있는 오픈 소스 자동화 서버입니다. Systems Manager의 기능인 Automation은 Amazon Machine Images(AMIs)에 애플리케이션 릴리스를 사전 설치하기 위한 구축 후 단계로 사용할 수 있습니다. 자세히 알아보기 Automation 및 Jenkins를 사용하여 AMIs 업데이트
ServiceNow	ServiceNow는 IT 서비스 및 운영을 관리할 수 있는 엔터프라이즈 서비스 관리 시스템입니다. Systems Manager의 모든 기능인 Automation, Change Manager, Incident Manager, OpsCenter는 AWS Service Management Connector를 사용하여 ServiceNow와 통합됩니다. 이 통합을 통해 ServiceNow에서 AWS Support 사례를 보고, 생성하고, 업데이트하고, 대응을 추가하고, 해결할 수 있습니다. 자세히 알아보기 ServiceNow과 통합

주제

- GitHub에서 스크립트 실행
- Systems Manager Compliance와 함께 Chef InSpec 프로파일 사용
- ServiceNow과 통합

GitHub에서 스크립트 실행

이 주제에서는 사전 정의된 Systems Manager 문서(SSM 문서) `AWS-RunRemoteScript`를 사용해 GitHub에서 Ansible Playbooks, Python, Ruby, PowerShell 스크립트를 비롯한 스크립트를 다운로드하는 방법을 설명합니다. 이 SSM 문서를 사용하면 더 이상 수동으로 스크립트를 Amazon Elastic Compute Cloud(Amazon EC2)로 포팅하거나 SSM 문서에 래핑할 필요가 없습니다. AWS Systems Manager과 GitHub의 통합은 코드형 인프라를 지원하여, 노드 관리에 소요되는 시간을 단축하고 플릿에서 구성을 표준화합니다.

또한 원격 위치에서 스크립트 또는 다른 SSM 문서를 다운로드할 수 있는 사용자 정의 SSM 문서를 생성할 수도 있습니다. 자세한 내용은 복합 문서 생성 단원을 참조하십시오.

여러 스크립트가 포함된 디렉터리를 다운로드할 수도 있습니다. 디렉터리에서 기본 스크립트를 실행할 때 Systems Manager는 디렉터리에 포함된 참조되는 스크립트도 모두 실행합니다.

다음은 GitHub로부터 스크립트 실행에 대한 중요 세부 정보입니다.

- Systems Manager는 스크립트가 노드에서 실행 가능한지 확인하지 않습니다. 스크립트를 다운로드하여 실행하기 전에 노드에 필요한 소프트웨어가 설치되어 있는지 확인합니다. 아니면 AWS Systems Manager의 기능인 Run Command 또는 State Manager를 사용하여 소프트웨어를 설치한 다음, 스크립트를 다운로드해 실행하는 복합 문서를 생성할 수 있습니다.
- 사용자는 모든 GitHub 요구 사항이 충족되도록 해야 합니다. 여기에는 필요에 따라 액세스 토큰을 새로 고치는 절차가 포함됩니다. 인증 또는 미인증된 요청의 수를 초과하지 않도록 해야 합니다. 자세한 내용은 GitHub 설명서를 참조하세요.
- GitHub Enterprise 리포지토리는 지원되지 않습니다.

주제

GitHub에서 Ansible Playbooks 실행

이 섹션에는 콘솔 또는 AWS Command Line Interface(AWS CLI)을 사용해 GitHub에서 Ansible Playbooks를 실행할 수 있는 절차가 포함되어 있습니다.

시작하기 전 준비 사항

프라이빗 GitHub 리포지토리에 저장된 스크립트를 실행하려는 경우 GitHub 보안 액세스 토큰에 대한 AWS Systems Manager SecureString 파라미터를 생성합니다. 수동으로 SSH를 통해 토큰을 전달해서는 프라이빗 GitHub 리포지토리에 저장된 스크립트에 액세스할 수 없습니다. 액세스 토큰은 Systems Manager SecureString 파라미터로 전달되어야 합니다. SecureString 파라미터 생성에 대한 자세한 내용은 Systems Manager 파라미터 생성 섹션을 참조하세요.

GitHub(콘솔)에서 Ansible Playbook 실행

GitHub에서 Ansible Playbook 실행

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택합니다.
3. Run command를(Run 명령) 선택합니다.
4. [Command 문서(Command document)] 목록에서 **AWS-RunRemoteScript**를 선택합니다.
5. 명령 파라미터에서 다음을 수행합니다.
   - 소스 유형에서 GitHub를 선택합니다.
   - [소스 정보(Source Info)] 상자에 소스에 액세스하는 데 필요한 정보를 다음 형식으로 입력합니다.

```
{
  "owner": "owner_name",
  "repository": "repository_name",
  "getOptions": "branch:branch_name",
  "path": "path_to_scripts_or_directory",
  "tokenInfo": "{{ssm-secure:SecureString_parameter_name}}"
}
```

이 예제는 webserver.yml이라는 파일을 다운로드합니다.

```
{
    "owner": "TestUser1",
    "repository": "GitHubPrivateTest",
    "getOptions": "branch:myBranch",
    "path": "scripts/webserver.yml",
    "tokenInfo": "{{ssm-secure:mySecureStringParameter}}"
}
```

> **Note**
>
> "branch"는 SSM 문서가 master가 아닌 분기에 저장된 경우에만 필요합니다. 리포지토리의 특정 커밋에 있는 스크립트 버전을 사용하려면 branch대신 getOptions에서 commitID를 사용합니다. 예: "getOptions": "commitID:bbc1ddb94...b76d3bEXAMPLE",

- [명령줄(Command Line)] 필드에 스크립트 실행을 위한 파라미터를 입력합니다. 다음 예를 참고하세요

  ansible-playbook -i “localhost,” --check -c local webserver.yml

- (옵션) 작업 디렉터리(Working Directory) 필드에 스크립트를 다운로드하여 실행하려는 노드에 있는 디렉터리의 이름을 입력합니다.
- (선택 사항) Execution Timeout(실행 제한 시간)에서 스크립트 명령 실행이 실패할 때까지 시스템이 기다리는 시간(단위: 초)을 지정합니다.

6. 대상 섹션에서, 태그를 지정하거나, 수동으로 인스턴스나 엣지 디바이스를 선택하거나, 리소스 그룹을 지정하여 이 작업을 실행할 관리형 노드를 식별합니다.

> **Tip**
>
> 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

7. Other parameters(다른 파라미터):

   - Comment(설명)에 명령에 대한 정보를 입력합니다.
   - 제한 시간(초)에서 전체 명령 실행이 실패할 때까지 시스템이 기다리는 시간을 초 단위로 지정합니다.

8. Rate control(속도 제어)에서

   - Concurrency(동시성)에서 명령을 동시에 실행할 관리형 노드의 백분율 또는 개수를 지정합니다.

> **Note**
>
> 관리형 노드에 적용할 태그를 지정하거나, AWS 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상 수를 제한합니다.

- Error threshold(오류 임계값)에서, 명령이 노드의 개수 또는 백분율에서 실패한 후 다른 관리형 노드에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째 오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.

9. (선택 사항) Output options(출력 옵션)에서 명령 출력을 파일에 저장하려면 Write command output to an S3 bucket(S3 버킷에 명령 출력 쓰기) 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

> **Note**
>
> 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아니라 인스턴스에 할당된 인스턴스 프로파일(EC2 인스턴스용) 또는 IAM 서비스 역할(하이브리드 정품 인증 시스템)의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

10. SNS notifications(SNS 알림) 섹션에서, 명령 실행 상태에 대한 알림이 전송되도록 하려면 Enable SNS notifications(SNS 알림 활성화) 확인란을 선택합니다.

    Run Command에 대한 Amazon SNS 알림 구성에 대한 자세한 내용은 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 섹션을 참조하세요.

11. Run(실행)을 선택합니다.

AWS CLI를 사용하여 GitHub에서 Ansible Playbook 실행

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.

2. 다음 명령을 실행하여 GitHub에서 스크립트를 다운로드하고 실행합니다.

```
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"]}'
```

다음은 로컬 Linux 시스템에서 실행하는 명령의 예입니다.

```
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"]}'
```

GitHub에서 Python 스크립트 실행

이 섹션에는 AWS Systems Manager 콘솔 또는 AWS Command Line Interface(AWS CLI)를 사용해 GitHub에서 Python 스크립트를 실행할 수 있는 절차가 포함되어 있습니다.

GitHub에서 Python 스크립트 실행(콘솔)

GitHub에서 Python 스크립트 실행

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Run Command를 선택합니다.
3. Run command를(Run 명령) 선택합니다.
4. [Command 문서(Command document)] 목록에서 **AWS-RunRemoteScript**를 선택합니다.
5. 명령 파라미터에서 다음을 수행합니다.
   - 소스 유형에서 GitHub를 선택합니다.
   - [소스 정보(Source Info)] 상자에 소스에 액세스하는 데 필요한 정보를 다음 형식으로 입력합니다.

```
{
    "owner": "owner_name",
    "repository": "repository_name",
    "getOptions": "branch:branch_name",
    "path": "path_to_document",
    "tokenInfo": "{{ssm-secure:SecureString_parameter_name}}"
}
```

다음 예에서는 complex-script라는 스크립트의 디렉터리를 다운로드합니다.

```
{
    "owner": "TestUser1",
    "repository": "SSMTestDocsRepo",
    "getOptions": "branch:myBranch",
    "path": "scripts/python/complex-script",
    "tokenInfo": "{{ssm-secure:myAccessTokenParam}}"
}
```

> **Note**
>
> "branch"는 스크립트가 master가 아닌 분기에 저장된 경우에만 필요합니다. 리포지토리의 특정 커밋에 있는 스크립트 버전을 사용하려면 branch대신 getOptions에서 commitID를 사용합니다. 예: "getOptions": "commitID:bbc1ddb94...b76d3bEXAMPLE",

- [명령줄(Command Line)]에 스크립트 실행을 위한 파라미터를 입력합니다. 다음 예를 참고하세요

```
mainFile.py argument-1 argument-2
```

이 예에서는 mainFile.py를 실행합니다. 그러면 이 파일이 complex-script 디렉터리의 다른 스크립트를 실행할 수 있습니다.

- (옵션) 작업 디렉터리(Working Directory)에 스크립트를 다운로드하여 실행하려는 노드 상의 디렉터리의 이름을 입력합니다.
- (선택 사항) Execution Timeout(실행 제한 시간)에서 스크립트 명령 실행이 실패할 때까지 시스템이 기다리는 시간(초)을 지정합니다.

6. 대상(Targets) 섹션에서, 태그를 지정하거나, 수동으로 인스턴스나 엣지 디바이스를 선택하거나, 리소스 그룹을 지정하여 이 작업을 실행할 관리형 노드를 식별합니다.

> **Tip**
>
> 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

7. Other parameters(다른 파라미터):

   - Comment(설명)에 명령에 대한 정보를 입력합니다.
   - 제한 시간(초)에서 전체 명령 실행이 실패할 때까지 시스템이 기다리는 시간을 초 단위로 지정합니다.

8. Rate control(속도 제어)에서

   - Concurrency(동시성)에서 명령을 동시에 실행할 관리형 노드의 백분율 또는 개수를 지정합니다.

   > **Note**
   >
   > 관리형 노드에 적용할 태그를 지정하거나, AWS 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상 수를 제한합니다.

   - Error threshold(오류 임계값)에서, 명령이 노드의 개수 또는 백분율에서 실패한 후 다른 관리형 노드에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째 오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.

9. (선택 사항) Output options(출력 옵션)에서 명령 출력을 파일에 저장하려면 Write command output to an S3 bucket(S3 버킷에 명령 출력 쓰기) 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

> **Note**
>
> 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아니라 인스턴스에 할당된 인스턴스 프로파일(EC2 인스턴스용) 또는 IAM 서비스 역할(하이브리드 정품 인증 시스템)의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세

요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

10. SNS notifications(SNS 알림) 섹션에서, 명령 실행 상태에 대한 알림이 전송되도록 하려면 Enable SNS notifications(SNS 알림 활성화) 확인란을 선택합니다.

    Run Command에 대한 Amazon SNS 알림 구성에 대한 자세한 내용은 Amazon SNS 알림을 사용하여 Systems Manager 상태 변경 모니터링 섹션을 참조하세요.
11. Run(실행)을 선택합니다.

AWS CLI를 사용하여 GitHub에서 Python 스크립트 실행

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.
2. 다음 명령을 실행하여 GitHub에서 스크립트를 다운로드하고 실행합니다.

```
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"]}'
```

   다음 예를 참고하세요

```
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 "]}'
```

   이 예에서는 `complex-script`라는 스크립트의 디렉터리를 다운로드합니다. `commandLine` 항목은 `mainFile.py`를 실행합니다. 그러면 이 파일이 `complex-script` 디렉터리의 다른 스크립트를 실행할 수 있습니다.

Systems Manager Compliance와 함께 Chef InSpec 프로파일 사용

AWS Systems Manager은 Chef InSpec과 통합됩니다. Chef InSpec은 GitHub 또는 Amazon Simple Storage Service(S3)에 저장할 육안 판독 프로파일을 생성할 수 있는 오픈 소스 테스트 프레임워크입

니다. Systems Manager를 사용하여 규정 준수 스캔을 수행하고 준수 및 비준수 노드를 확인할 수 있습니다. 프로파일은 컴퓨팅 환경에 대한 보안, 규정 준수 또는 정책 요구 사항입니다. 예를 들어 AWS Systems Manager의 기능인 Compliance를 통해 노드를 스캔할 때 다음을 확인하는 프로파일을 생성할 수 있습니다.

- 특정 포트가 열려 있는지 또는 닫혀 있는지 확인
- 특정 애플리케이션이 실행 중인지 확인
- 특정 패키지가 설치되었는지 확인
- 특정 속성에 대한 Windows 레지스트리 키 확인

Systems Manager를 사용하여 관리하는 온프레미스 서버 또는 가상 머신 및 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스에 대해 InSpec 프로파일을 생성할 수 있습니다. 다음의 Chef InSpec 프로파일 예제는 포트 22가 열려 있는지 확인합니다.

```
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
```

InSpec에는 검사 및 감사 컨트롤을 빠르게 작성할 수 있는 리소스 모음이 들어 있습니다. InSpec은 InSpec 도메인 전용 언어(DSL)를 사용하여 Ruby로 이러한 컨트롤을 작성합니다. InSpec 대규모 사용자 커뮤니티에서 만든 프로파일을 사용할 수도 있습니다. 예를 들어 GitHub에 대한 DevSec chef-os-hardening 프로젝트에는 노드 보안에 도움이 되는 수십 개의 프로파일이 포함되어 있습니다. GitHub 또는 Amazon S3에서 프로파일을 작성하고 저장할 수 있습니다.

작동 방식

다음은 Compliance에서 InSpec 프로파일을 사용하는 프로세스가 어떻게 이루어지는지를 보여줍니다.

1. 미리 정의된 InSpec 프로파일을 지정하여 사용하거나 직접 만들 수 있습니다. GitHub에서 사전 정의 프로파일을 사용하여 시작할 수 있습니다. 고유한 InSpec 프로파일을 생성하는 방법에 대한 자세한 내용은 ChefChef InSpec 프로파일을 참조하세요.
2. 프로파일을 퍼블릭 또는 프라이빗 GitHub 리포지토리 또는 S3 버킷에 저장합니다.

3. Systems Manager 문서(SSM 문서) `AWS-RunInspecChecks`를 사용하여 InSpec 프로파일로 Compliance를 실행합니다. AWS Systems Manager의 기능인 Run Command(온디맨드 검사의 경우)를 사용하여 Compliance를 시작하거나, AWS Systems Manager의 기능인 State Manager를 사용하여 정규 Compliance 검사를 예약할 수 있습니다.
4. 규정 준수 API 또는 규정 준수 콘솔을 사용하여 비준수 노드를 식별합니다.

> **Note**
>
> 다음 정보를 참고하세요.
>
> - Chef는 노드의 클라이언트를 사용하여 프로파일을 처리합니다. 클라이언트를 설치할 필요가 없습니다. Systems Manager가 SSM 문서 `AWS-RunInspecChecks`를 실행할 때 시스템에서 클라이언트가 설치되었는지 확인합니다. 설치되어 있지 않은 경우 Systems Manager는 검사 중 Chef 클라이언트를 설치하고, 검사가 완료되면 클라이언트를 제거합니다.
> - 이 주제에 설명된 대로 SSM 문서 `AWS-RunInspecChecks`를 실행하면 각 대상 노드에 `Custom:Inspec` 유형의 규정 준수 항목이 할당됩니다. 이 규정 준수 유형을 할당하기 위해 문서는 PutComplianceItems API 작업을 호출합니다.

InSpec 규정 준수 검사 실행

이 섹션에서는 Systems Manager 콘솔과 AWS Command Line Interface(AWS CLI)를 사용하여 InSpec 규정 준수 검사를 실행하는 방법을 설명합니다. 콘솔 절차는 State Manager를 구성하여 검사를 실행하는 방법을 보여줍니다. AWS CLI 절차는 Run Command를 구성하여 검사를 실행하는 방법을 보여줍니다.

State Manager(콘솔)로 InSpec 규정 준수 검사 실행

AWS Systems Manager 콘솔을 사용해 State Manager에서 InSpec 규정 준수를 실행하려면

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 State Manager를 선택합니다.
3. 연결 생성을 선택합니다.
4. 연결 세부 정보 제공 섹션에 이름을 입력합니다.
5. [문서(Document)] 목록에서 **`AWS-RunInspecChecks`**를 선택합니다.
6. 문서 버전 목록에서 실행 시간 시 최신 버전을 선택합니다.

7. 파라미터 섹션의 소스 유형 목록에서 GitHub 또는 S3를 선택합니다.

   GitHub를 선택한 후 소스 정보 필드에 퍼블릭 또는 프라이빗 GitHub 리포지토리의 InSpec 프로파일 경로를 입력합니다. 다음은 Systems Manager 팀이 https://github.com/awslabs/amazon-ssm/tree/master/Compliance/InSpec/PortCheck에서 제공하는 퍼블릭 프로파일 경로의 예입니다.

   ```
   {"owner":"awslabs","repository":"amazon-ssm","path":"Compliance/InSpec/
   PortCheck","getOptions":"branch:master"}
   ```

   S3를 선택한 경우 소스 정보 필드의 S3 버킷에 InSpec 프로파일에 대한 유효한 URL을 입력합니다.

   Systems Manager가 GitHub 및 Amazon S3와 통합되는 방법에 대한 자세한 내용은 GitHub에서 스크립트 실행 섹션을 참조하세요.

8. Targets(대상) 섹션에서, 태그를 지정하거나, 수동으로 인스턴스나 엣지 디바이스를 선택하거나, 리소스 그룹을 지정하여 이 작업을 실행할 관리형 노드를 식별합니다.

   > **Tip**
   >
   > 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

9. [일정 지정(Specify schedule)] 섹션에서 일정 작성기 옵션을 사용하여 Compliance 검사를 실행할 때를 지정하는 일정을 생성합니다.

10. Rate control(속도 제어)에서

    - Concurrency(동시성)에서 명령을 동시에 실행할 관리형 노드의 백분율 또는 개수를 지정합니다.

      > **Note**
      >
      > 관리형 노드에 적용할 태그를 지정하거나, AWS 리소스 그룹을 지정하여 대상을 선택하였지만 대상으로 지정할 관리형 노드 수를 잘 모를 경우에는 백분율을 지정하여 동시에 문서를 실행할 수 있는 대상 수를 제한합니다.

    - Error threshold(오류 임계값)에서, 명령이 노드의 개수 또는 백분율에서 실패한 후 다른 관리형 노드에서 해당 명령의 실행을 중지할 시간을 지정합니다. 예를 들어 세 오류를 지정하면 네 번째

오류를 받았을 때 Systems Manager가 명령 전송을 중지합니다. 여전히 명령을 처리 중인 관리형 노드도 오류를 전송할 수 있습니다.

11. (선택 사항) Output options(출력 옵션)에서 명령 출력을 파일에 저장하려면 Write command output to an S3 bucket(S3 버킷에 명령 출력 쓰기) 상자를 선택합니다. 상자에 버킷 및 접두사(폴더) 이름을 입력합니다.

> **Note**
>
> 데이터를 S3 버킷에 쓰는 기능을 부여하는 S3 권한은 이 작업을 수행하는 IAM 사용자의 권한이 아니라 인스턴스에 할당된 인스턴스 프로파일(EC2 인스턴스용) 또는 IAM 서비스 역할(하이브리드 정품 인증 시스템)의 권한입니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성이나 하이브리드 환경을 위한 IAM 서비스 역할 생성을 참조하세요. 또한 지정된 S3 버킷이 다른 AWS 계정에 있는 경우 관리형 노드와 연결된 인스턴스 프로파일 또는 IAM 서비스 역할은 해당 버킷에 쓸 수 있는 권한이 있어야 합니다.

12. 연결 생성을 선택합니다. 시스템에서 연결을 생성하고 규정 준수 검사를 자동으로 실행합니다.
13. 검사가 완료될 때까지 몇 분 정도 기다린 후 탐색 창에서 규정 준수를 선택합니다.
14. 해당 관리형 인스턴스에서 Compliance Type 열이 Custom:Inspec인 노드를 찾습니다.
15. 노드 ID를 선택하여 비준수 상태 정보를 봅니다.

Run Command(AWS CLI)로 InSpec 규정 준수 검사 실행

1. 아직 하지 않은 경우 AWS Command Line Interface(AWS CLI)를 설치하고 구성합니다.

   자세한 내용은 최신 버전의 AWS CLI 설치 또는 업데이트를 참조하세요.
2. 다음 명령 중 하나를 실행하여 GitHub 또는 Amazon S3에서 InSpec 프로파일을 실행합니다.

   명령은 다음 파라미터를 사용합니다.

   - sourceType: GitHub 또는 Amazon S3
   - sourceInfo: GitHub 또는 S3 버킷에 있는 InSpec 프로파일 폴더에 대한 URL. 이 폴더에는 기본 InSpec 파일(*.yml)과 기타 모든 관련 컨트롤(*.rb)이 있어야 합니다.

   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"}"]}'
```

다음 예를 참고하세요

```
aws ssm send-command --document-name "AWS-RunInspecChecks" --targets
 '[{"Key":"tag:testEnvironment","Values":["webServers"]}]' --parameters
 '{"sourceType":["GitHub"],"getOptions":"branch:master","sourceInfo":["{\"owner\":
\"awslabs\", \"repository\":\"amazon-ssm\", \"path\": \"Compliance/InSpec/PortCheck
\"}"]}'
```

Amazon S3

```
aws ssm send-command --document-name "AWS-RunInspecChecks" --targets
 '[{"Key":"tag:tag_name","Values":["tag_value"]}]' --parameters'{"sourceType":
["S3"],"sourceInfo":["{\"path\":\"https://s3.aws-api-domain/DOC-EXAMPLE-
BUCKET/Inspec.yml_file\"}"]}'
```

다음 예를 참고하세요

```
aws ssm send-command --document-name "AWS-RunInspecChecks" --targets
 '[{"Key":"tag:testEnvironment","Values":["webServers"]}]' --
parameters'{"sourceType":["S3"],"sourceInfo":["{\"path\":\"https://s3.aws-api-
domain/DOC-EXAMPLE-BUCKET/InSpec/PortCheck.yml\"}"]}'
```

3. 다음 명령을 실행하여 규정 준수 검사에 대한 요약 정보를 봅니다.

```
aws ssm list-resource-compliance-summaries --filters
 Key=ComplianceType,Values=Custom:Inspec
```

4. 다음 명령을 실행하여 규정을 준수하지 않는 노드의 세부 정보를 확인합니다.

```
aws ssm list-compliance-items --resource-ids node_ID --resource-type
 ManagedInstance --filters Key=DocumentName,Values=AWS-RunInspecChecks
```

ServiceNow과 통합

ServiceNow는 IT 서비스, 티켓팅 시스템, 지원과 같은 조직 수준의 워크플로를 생성하고 관리하기 위한 클라우드 기반 서비스 관리 시스템을 제공합니다. AWS Service Management Connector는

ServiceNow와 Systems Manager를 통합하여 ServiceNow에서 AWS 리소스를 프로비저닝, 관리 및 운영합니다. AWS Service Management Connector를 사용하여 ServiceNow를 AWS Systems Manager의 모든 기능인 Automation, Change Manager, Incident Manager, OpsCenter와 통합할 수 있습니다.

ServiceNow를 사용하여 다음 작업을 수행할 수 있습니다.

- Systems Manager에서 자동화 플레이북을 실행합니다.
- Systems Manager OpsItems에서 인시던트를 보고, 업데이트하고, 해결합니다.
- Systems Manager OpsCenter를 통해 인시던트와 같은 운영 항목을 보고 관리합니다.
- 사전 승인된 변경 템플릿의 선별된 목록에서 Systems Manager 변경 요청을 보고 실행합니다.
- Incident Manager와 통합하여 AWS 호스팅 애플리케이션과 관련된 인시던트를 관리하고 해결합니다.

> **Note**
>
> ServiceNow와 통합하는 방법에 대한 자세한 내용은 AWS Service Management Connector 관리자 안내서의 AWS 서비스 통합 구성을 참조하세요.

Systems Manager 리소스에 태그 지정

태그는 AWS 리소스에 할당하는 레이블입니다. 각 태그는 사용자가 정의하는 키와 값으로 구성됩니다.

태그를 사용하면 용도, 소유자 또는 환경을 기준으로 하는 것과 같이 AWS 리소스를 다양한 방식으로 분류할 수 있습니다. 예를 들어 개발 또는 프로덕션에 사용되는지 여부에 따라 리소스를 구성하고 관리하려는 경우, 일부 리소스에 `Environment` 키와 `Production` 값을 태그로 지정할 수 있습니다. 그런 다음 태그가 `"Key=Environment,Values=Production"`로 지정된 리소스에 대해 다양한 유형의 쿼리를 수행할 수 있습니다. 예를 들어, 계정의 관리형 노드에 대한 태그 집합을 정의하여 `development`, `staging` 및 `production`으로 그룹화된 SUSE Linux Enterprise Server 등의 운영 체제 및 환경별로 노드를 추적하거나 대상으로 지정할 수 있습니다. 또한 명령에 이 키 값 페어를 지정하여 그룹의 모든 노드에서 업데이트 스크립트를 실행하거나 해당 노드의 상태를 검토하는 등의 리소스 작업을 수행할 수 있습니다.

여러 작업에서 AWS Systems Manager 리소스에 적용된 태그를 사용할 수 있습니다. 예를 들어 명령을 실행하거나 유지 관리 기간에 대상을 할당할 때 지정된 태그 키 값 페어로 태그가 지정된 관리형 노드만 대상으로 지정할 수 있습니다. 리소스에 적용된 태그를 기준으로 리소스에 대한 액세스를 제한할 수도 있습니다.

동일한 유형 외에도 다양한 유형의 AWS 리소스에 대해 동일한 태그를 지정하여 리소스 그룹을 생성할 수 있습니다. 그런 다음 Resource Groups를 사용하여 그룹 내 어떤 리소스가 호환되고 올바르게 작동하고 있는지, 어떤 리소스에 작업이 필요한지에 대한 정보를 확인할 수 있습니다. 사용자가 보는 정보는 지원되는 Systems Manager 리소스 유형 외에도 리소스 그룹에 추가할 수 있는 모든 유형의 AWS 리소스와 관련이 있습니다. 자세한 내용은 AWS Resource Groups User Guide의 What are AWS Resource Groups? 섹션을 참조하세요.

이 장의 나머지 부분에서는 Systems Manager 리소스에서 태그를 추가하거나 제거하는 방법에 대해 설명합니다().

주제

- OpsItems 태그 지정
- Systems Manager 파라미터에 태그 지정
- 패치 기준 태깅

태그를 지정할 수 있는 Systems Manager 리소스

다음 AWS Systems Manager 리소스에 태그를 적용할 수 있습니다.

- Associations
- 자동화
- 문서
- 유지 관리 기간
- 관리형 노드
- OpsItems
- OpsMetadata
- 파라미터
- 패치 기준

OpsItems와 OpsMetadata를 제외한 이러한 각 유형을 리소스 그룹에 추가할 수 있습니다.

리소스 유형에 따라 태그를 사용하여 작업에 포함해야 하는 리소스를 식별할 수 있습니다. 예를 들어 관리형 노드 그룹에 태그를 지정한 다음, 해당 키 값 페어가 있는 노드만 대상으로 하는 유지 관리 기간 작업을 실행할 수 있습니다.

사용자가 액세스할 수 있는 태그를 지정하는 AWS Identity and Access Management(IAM) 정책을 생성하거나 IAM 엔터티(사용자, 역할 또는 그룹)에 정책을 연결하여 이러한 리소스 유형에 대한 사용자 액세스를 제한할 수도 있습니다. 다음은 태그를 사용하여 리소스 액세스를 제한하는 몇 가지 예입니다.

- 사용자 정의 Systems Manager 문서(SSM 문서) 세트에 태그를 적용한 다음, 해당 태그가 있지만 다른 태그는 없는 문서에 대한 액세스 권한을 부여하거나 해당 문서에 대한 액세스만 금지하도록 IAM 정책을 생성하고 적용할 수 있습니다.
- OpsItems에 태그를 지정한 다음 해당 리소스를 보거나 업데이트할 수 있는 액세스 권한이 있는 사용자 또는 그룹을 제한하는 IAM 정책을 생성할 수 있습니다. 예를 들어 조직 담당자는 모든 OpsItems에 대한 전체 액세스 권한을 부여받을 수 있지만, 소프트웨어 개발자와 지원 엔지니어는 자신이 담당하는 프로젝트 또는 클라이언트 세그먼트에 대해서만 액세스 권한을 부여받을 수 있습니다.

- 지원되는 6가지 유형의 모든 리소스에 공통 태그를 적용하고 `Key=Project,Value=ProjectA` 또는 `Key=Environment,Value=Development` 등의 리소스에 대한 액세스 권한을 부여하는 IAM 정책을 생성할 수 있습니다. 두 태그 페어가 모두 할당된 리소스에만 액세스 권한을 부여할 수도 있습니다. 예를 들어 개발 환경에서 ProjectA에 대한 리소스만 작업하도록 사용자를 제한할 수 있습니다.

Systems Manager Resource Groups 콘솔, 지원되는 리소스 유형에 대한 콘솔(예: Maintenance Windows 콘솔 또는 OpsCenter 콘솔), AWS Command Line Interface(AWS CLI) 및 AWS Tools for PowerShell을 사용할 수 있습니다. 리소스를 생성 또는 업데이트할 때 태그를 추가할 수 있습니다. 예를 들면, AWS CLI add-tags-to-resource 명령을 사용하여 지원되는 Systems Manager 리소스 유형을 생성한 후에 태그를 추가할 수 있습니다. remove-tags-from-resource 명령을 사용하여 태그를 제거할 수도 있습니다.

Systems Manager 연결에 태그 지정

이 섹션의 주제에서는 State Manager 연결에 태그를 사용하는 방법을 설명합니다. State Manager는 AWS Systems Manager의 구성 요소입니다.

주제

태그를 사용하여 연결 생성

AWS CLI를 사용하여 State Manager 연결을 생성할 때 태그를 추가할 수 있습니다. Systems Manager 콘솔을 사용하여 연결을 생성할 때 태그를 추가하는 것은 지원되지 않습니다. 자세한 설명은 연결 생성(명령줄)을 참조하세요.

기존 연결에 태그 추가

다음 절차에 따라 명령줄을 사용하여 기존 State Manager 연결에 태그를 추가합니다.

주제

기존 연결에 태그 추가(AWS CLI)

1. AWS CLI로 다음 명령을 실행하여 태그를 지정할 수 있는 연결을 나열합니다.

```
aws ssm list-associations
```

   태그를 지정할 연결의 이름을 기록해 둡니다.

2. 다음 명령을 실행하여 연결에 태그를 지정합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
aws ssm add-tags-to-resource \
    --resource-type "Association" \
    --resource-id "association-ID" \
    --tags "Key=tag-key,Value=tag-value"
```

   명령이 성공하면 출력이 표시되지 않습니다.

3. 다음 명령을 실행하여 연결 태그를 확인합니다.

```
aws ssm list-tags-for-resource --resource-type "Association" --resource-id
 "association-ID"
```

기존 연결에 태그 추가(AWS Tools for PowerShell)

1. 다음 명령을 실행하여 태그를 지정할 수 있는 연결을 나열합니다.

```
Get-SSMAssociationList
```

2. 다음 명령을 실행하여 파라미터에 태그를 지정합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
$tag = New-Object Amazon.SimpleSystemsManagement.Model.Tag
```

```
$tag.Key = "tag-key"
```

```
$tag.Value = "tag-value"
```

```
Add-SSMResourceTag `
    -ResourceType "Association" `
    -ResourceId "association-ID" `
    -Tag $tag `
    -Force
```

3. 다음 명령을 실행하여 연결 태그를 확인합니다.

```
Get-SSMResourceTag `
    -ResourceType "Association" `
    -ResourceId "association-ID"
```

연결에서 태그 제거

명령줄을 사용하여 State Manager 연결에서 태그를 제거할 수 있습니다.

연결에서 태그 제거(명령줄)

1. 원하는 명령줄 도구로 다음 명령을 실행하여 계정의 연결을 나열합니다.

   Linux & macOS

   ```
   aws ssm list-associations
   ```

   Windows

   ```
   aws ssm list-associations
   ```

   PowerShell

   ```
   Get-SSMAssociationList
   ```

   태그를 제거할 연결의 이름을 확인합니다.

2. 다음 명령을 실행하여 연결에서 태그를 제거합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

Linux & macOS

```
aws ssm remove-tags-from-resource \
    --resource-type "Association" \
    --resource-id "association-ID" \
    --tag-key "tag-key"
```

Windows

```
aws ssm remove-tags-from-resource ^
    --resource-type "Association" ^
    --resource-id "association-ID" ^
    --tag-key "tag-key"
```

PowerShell

```
Remove-SSMResourceTag
    -ResourceId "association-ID"
    -ResourceType "Association"
    -TagKey "tag-key
```

명령이 성공하면 출력이 표시되지 않습니다.

3. 다음 명령을 실행하여 연결 태그를 확인합니다.

Linux & macOS

```
aws ssm list-tags-for-resource \
    --resource-type "Association" \
    --resource-id "association-ID"
```

Windows

```
aws ssm list-tags-for-resource ^
    --resource-type "Association" ^
    --resource-id "association-ID"
```

PowerShell

```
Get-SSMResourceTag `
    -ResourceType "Association" `
    -ResourceId "association-ID"
```

태깅 자동화

이 섹션의 주제에서는 자동화에 태그를 사용하는 방법을 설명합니다. AWS Systems Manager 자동화에 최대 5개의 태그를 추가할 수 있습니다. 콘솔 또는 명령줄에서 자동화를 시작할 때 또는 명령줄을 사용하여 자동화를 실행한 후에 태그를 추가할 수 있습니다.

자동화에 태그 추가(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 왼쪽 탐색 창에서 Automation을 선택합니다.
3. 실행할 자동화 실행서를 선택합니다.
4. Execute automation(자동화 실행)을 선택합니다.
5. 태그(Tags) 섹션에서 편집(Edit)을 선택한 다음 하나 이상의 키 값 태그 페어를 추가합니다.
6. 저장(Save)을 선택합니다.

자동화에 태그 추가(명령줄)

원하는 명령줄 도구를 사용하여 다음 명령을 실행하여 자동화가 시작될 때 태그를 추가합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

Linux & macOS

```
aws ssm start-automation-execution \
    --document-name DocumentName \
    --parameters ParametersRequiredByDocument \
    --tags "Key=ExampleKey,Value=ExampleValue"
```

Windows

```
aws ssm start-automation-execution ^
```

```
    --document-name DocumentName ^
    --parameters ParametersRequiredByDocument ^
    --tags "Key=ExampleKey,Value=ExampleValue"
```

PowerShell

```
$exampleTag = New-Object Amazon.SimpleSystemsManagement.Model.Tag
$exampleTag.Key = "ExampleKey"
$exampleTag.Value = "ExampleValue"

Start-SSMAutomationExecution `
    -DocumentName DocumentName `
    -Parameter ParametersRequiredByDocument
    -Tag $exampleTag
```

1. 원하는 명령줄 도구를 사용하여 자동화를 실행한 후에 태깅할 수도 있습니다. 다음 명령을 실행하여 자동화에 태그를 추가합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   Linux & macOS

   ```
   aws ssm add-tags-to-resource \
       --resource-type "Automation" \
       --resource-id "automation-execution-id" \
       --tags "Key=ExampleKey,Value=ExampleValue"
   ```

   Windows

   ```
   aws ssm add-tags-to-resource ^
       --resource-type "Automation" ^
       --resource-id "automation-execution-id" ^
       --tags "Key=ExampleKey,Value=ExampleValue"
   ```

   PowerShell

   ```
   $exampleTag = New-Object Amazon.SimpleSystemsManagement.Model.Tag
   $exampleTag.Key = "ExampleKey"
   $exampleTag.Value = "ExampleValue"

   Add-SSMResourceTag `
   ```

```
    -ResourceType "Automation" `
    -ResourceId "automation-execution-id" `
    -Tag $exampleTag `
    -Force
```

명령이 성공하면 출력이 표시되지 않습니다.

2. 다음 명령을 실행하여 자동화의 태그를 확인합니다.

Linux & macOS

```
aws ssm list-tags-for-resource \
    --resource-type "Automation" \
    --resource-id "automation-execution-id"
```

Windows

```
aws ssm list-tags-for-resource ^
    --resource-type "Automation" ^
    --resource-id "automation-execution-id"
```

PowerShell

```
Get-SSMResourceTag `
    -ResourceType "Automation" `
    -ResourceId "automation-execution-id"
```

자동화에서 태그 제거

명령줄 도구를 사용하여 자동화에서 태그를 제거할 수 있습니다.

자동화에서 태그 제거(명령줄)

1. 원하는 명령줄 도구를 사용하여 다음 명령을 실행하여 자동화에서 태그를 제거합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

Linux & macOS

```
aws ssm remove-tags-from-resource \
    --resource-type "Automation" \
```

```
    --resource-id "automation-execution-id" \
    --tag-key "tag-key"
```

Windows

```
aws ssm remove-tags-from-resource ^
    --resource-type "Automation" ^
    --resource-id "automation-execution-id" ^
    --tag-key "tag-key"
```

PowerShell

```
Remove-SSMResourceTag `
    -ResourceId "automation-execution-id" `
    -ResourceType "Automation" `
    -TagKey "tag-key" `
    -Force
```

2. 다음 명령을 실행하여 자동화의 태그를 확인합니다.

Linux & macOS

```
aws ssm list-tags-for-resource \
    --resource-type "Automation" \
    --resource-id "automation-execution-id"
```

Windows

```
aws ssm list-tags-for-resource ^
    --resource-type "Automation" ^
    --resource-id "automation-execution-id"
```

PowerShell

```
Get-SSMResourceTag `
    -ResourceType "Automation" `
    -ResourceId "automation-execution-id"
```

Systems Manager 문서에 태그 지정

이 섹션의 주제에서는 Systems Manager 문서(SSM 문서)에서 태그를 사용하는 방법을 설명합니다.

주제

태그를 지정하여 문서 만들기

사용자 정의 SSM 문서를 생성할 때 태그를 추가할 수 있습니다.

자세한 내용은 다음 주제를 참조하세요.

- SSM 문서 생성(콘솔)
- SSM 문서 생성(명령줄)

기존 문서에 태그 추가

Systems Manager 콘솔 또는 명령줄을 사용하여 소유한 사용자 정의 SSM 문서에 태그를 추가할 수 있습니다.

주제

기존 SSM 문서에 태그 추가(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. 내 소유(Owned by me) 탭을 선택합니다.
4. 태그를 추가할 문서의 이름을 선택한 다음 세부 정보(Details) 탭을 선택합니다.
5. 태그(Tags) 섹션에서 편집(Edit)을 선택한 다음 하나 이상의 키 값 태그 페어를 추가합니다.

6. Save(저장)를 선택합니다.

기존 SSM 문서에 태그 추가(명령줄)

기존 SSM 문서에 태그를 추가하려면(명령줄)

1. 원하는 명령줄 도구를 사용하여 다음 명령을 실행함으로써 태그를 지정할 수 있는 문서 목록을 확인합니다.

   Linux & macOS

   ```
   aws ssm list-documents
   ```

   Windows

   ```
   aws ssm list-documents
   ```

   PowerShell

   ```
   Get-SSMDocumentList
   ```

   태그를 지정할 문서의 이름을 기록해 둡니다.

2. 다음 명령을 실행하여 문서에 태그를 지정합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   Linux & macOS

   ```
   aws ssm add-tags-to-resource \
       --resource-type "Document" \
       --resource-id "document-name" \
       --tags "Key=tag-key,Value=tag-value"
   ```

   Windows

   ```
   aws ssm add-tags-to-resource ^
       --resource-type "Document" ^
       --resource-id "document-name" ^
       --tags "Key=tag-key,Value=tag-value"
   ```

PowerShell

```
$tag = New-Object Amazon.SimpleSystemsManagement.Model.Tag
```

```
$tag.Key = "tag-key"
```

```
$tag.Value = "tag-value"
```

```
Add-SSMResourceTag `
    -ResourceType "Document" `
    -ResourceId "document-name" `
    -Tag $tag `
    -Force
```

명령이 성공하면 출력이 표시되지 않습니다.

3. 다음 명령을 실행하여 문서 태그를 확인합니다.

Linux & macOS

```
aws ssm list-tags-for-resource \
    --resource-type "Document" \
    --resource-id "document-name"
```

Windows

```
aws ssm list-tags-for-resource ^
    --resource-type "Document" ^
    --resource-id "document-name"
```

PowerShell

```
Get-SSMResourceTag `
    -ResourceType "Document" `
    -ResourceId "document-name"
```

SSM 문서에서 태그 제거

Systems Manager 콘솔이나 명령줄을 사용하여 SSM 문서에서 태그를 제거할 수 있습니다.

주제

SSM 문서에서 태그 제거(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Documents를 선택합니다.
3. 내 소유(Owned by me) 탭을 선택합니다.
4. 태그를 제거할 문서의 이름을 선택한 다음 세부 정보(Details) 탭을 선택합니다.
5. 태그(Tags) 섹션에서 편집(Edit)을 선택한 다음, 더 이상 필요하지 않은 태그 페어 옆에 있는 제거(Remove)를 선택합니다.
6. Save(저장)를 선택합니다.

SSM 문서에서 태그 제거(명령줄)

1. 원하는 명령줄 도구를 사용하여 다음 명령을 실행함으로써 계정의 문서를 나열합니다.

   Linux & macOS

   ```
   aws ssm list-documents
   ```

   Windows

   ```
   aws ssm list-documents
   ```

   PowerShell

   ```
   Get-SSMDocumentList
   ```

   태그를 제거할 문서의 이름을 확인합니다.

2. 다음 명령을 실행하여 문서에서 태그를 제거합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   Linux & macOS

   ```
   aws ssm remove-tags-from-resource \
       --resource-type "Document" \
       --resource-id "document-name" \
       --tag-key "tag-key"
   ```

   Windows

   ```
   aws ssm remove-tags-from-resource ^
       --resource-type "Document" ^
       --resource-id "document-name" ^
       --tag-key "tag-key"
   ```

   PowerShell

   ```
   Remove-SSMResourceTag `
       -ResourceId "document-name" `
       -ResourceType "Document" `
       -TagKey "tag-key" `
       -Force
   ```

   명령이 성공하면 출력이 표시되지 않습니다.

3. 다음 명령을 실행하여 문서 태그를 확인합니다.

   Linux & macOS

   ```
   aws ssm list-tags-for-resource \
       --resource-type "Document" \
       --resource-id "document-name"
   ```

   Windows

   ```
   aws ssm list-tags-for-resource ^
       --resource-type "Document" ^
       --resource-id "document-name"
   ```

PowerShell

```
Get-SSMResourceTag `
    -ResourceType "Document" `
    -ResourceId "document-name"
```

유지 관리 기간 태깅

이 섹션의 주제에서는 유지 관리 기간에 태그를 지정하는 방법에 대해 설명합니다.

주제

- 태그를 지정하여 유지 관리 기간 만들기
- 기존 유지 관리 기간에 태그 추가
- 유지 관리 기간에서 태그 제거

태그를 지정하여 유지 관리 기간 만들기

태그를 생성할 때 유지 관리 기간에 태그를 추가할 수 있습니다.

자세한 내용은 다음 주제를 참조하세요.

- 유지 관리 기간 생성(콘솔)
- 자습서: 유지 관리 기간 생성 및 구성(AWS CLI)

기존 유지 관리 기간에 태그 추가

AWS Systems Manager 콘솔 또는 명령줄을 사용하여 소유한 유지 관리 기간에 태그를 추가할 수 있습니다.

주제

- 기존 유지 관리 기간에 태그 추가(콘솔)
- 기존 유지 관리 기간에 태그 추가(AWS CLI)
- 유지 관리 기간에 태그 지정(AWS Tools for PowerShell)

기존 유지 관리 기간에 태그 추가(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Maintenance Windows를 선택합니다.
3. 이미 생성한 유지 관리 기간의 이름을 선택한 후 태그(Tags) 탭을 선택합니다.
4. 태그 편집(Edit tags)을 선택한 다음 태그 추가(Add tag)를 선택합니다.
5. 키(Key)에 태그 키를 입력합니다(예: **`Environment`**).
6. 값(Value)에 태그 값을 입력합니다(예: **`Test`**).
7. Save changes(변경 사항 저장)를 선택합니다.

기존 유지 관리 기간에 태그 추가(AWS CLI)

1. 원하는 명령줄 도구를 사용하여 다음 명령을 실행함으로써 태그를 지정할 수 있는 유지 관리 기간 목록을 확인합니다.

   ```
   aws ssm describe-maintenance-windows
   ```

   태그를 지정할 유지 관리 기간의 ID를 확인합니다.
2. 다음 명령을 실행하여 유지 관리 기간에 태그를 지정합니다. 각 *`example resource placeholder`*를 사용자의 정보로 바꿉니다.

   Linux & macOS

   ```
   aws ssm add-tags-to-resource \
       --resource-type "MaintenanceWindow" \
       --resource-id "window-id" \
       --tags "Key=tag-key,Value=tag-value"
   ```

   Windows

   ```
   aws ssm add-tags-to-resource ^
       --resource-type "MaintenanceWindow" ^
       --resource-id "window-id" ^
       --tags "Key=tag-key,Value=tag-value"
   ```

   명령이 성공하면 출력이 표시되지 않습니다.

3. 다음 명령을 실행하여 유지 관리 기간의 태그를 확인합니다.

   Linux & macOS

   ```
   aws ssm list-tags-for-resource \
       --resource-type "MaintenanceWindow" \
       --resource-id "window-id"
   ```

   Windows

   ```
   aws ssm list-tags-for-resource ^
       --resource-type "MaintenanceWindow" ^
       --resource-id "window-id"
   ```

유지 관리 기간에 태그 지정(AWS Tools for PowerShell)

1. 다음 명령을 실행하여 태그를 지정할 수 있는 유지 관리 기간을 나열합니다.

   ```
   Get-SSMMaintenanceWindow
   ```

2. 다음 명령을 실행하여 유지 관리 기간에 태그를 지정합니다.

   ```
   $tag = New-Object Amazon.SimpleSystemsManagement.Model.Tag
   ```

   ```
   $tag.Key = "tag-key"
   ```

   ```
   $tag.Value = "tag-value"
   ```

   ```
   Add-SSMResourceTag `
       -ResourceType "MaintenanceWindow" `
       -ResourceId "window-id" `
       -Tag $tag
   ```

   *windows-id*는 태그를 지정할 유지 관리 기간의 ID입니다.

   *tag-key*는 제공한 사용자 정의 키의 이름입니다. 예를 들면 Environment 또는 Project입니다.

*tag-value*는 해당 키에 대해 제공할 값의 사용자 정의 콘텐츠입니다. 예를 들면 Production 또는 Q321입니다.

3. 다음 명령을 실행하여 유지 관리 기간의 태그를 확인합니다.

```
Get-SSMResourceTag `
    -ResourceType "MaintenanceWindow" `
    -ResourceId "window-id"
```

유지 관리 기간에서 태그 제거

Systems Manager 콘솔이나 명령줄을 사용하여 유지 관리 기간에서 태그를 제거할 수 있습니다.

주제

- 유지 관리 기간에서 태그 제거(콘솔)
- 유지 관리 기간에서 태그 제거(명령줄)

유지 관리 기간에서 태그 제거(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Maintenance Windows를 선택합니다.
3. 태그를 제거할 유지 관리 기간의 이름을 선택한 다음, 태그(Tags) 탭을 선택합니다.
4. 태그 편집(Edit tags)을 선택한 다음, 더 이상 필요하지 않은 태그 페어 옆에 있는 태그 제거(Remove tag)를 선택합니다.
5. Save changes(변경 사항 저장)를 선택합니다.

유지 관리 기간에서 태그 제거(명령줄)

1. 원하는 명령줄 도구를 사용하여 다음 명령을 실행함으로써 계정의 유지 관리 기간을 나열합니다.

   Linux & macOS

```
aws ssm describe-maintenance-windows
```

Windows

```
aws ssm describe-maintenance-windows
```

PowerShell

```
Get-SSMMaintenanceWindows
```

태그를 제거할 유지 관리 기간의 ID를 확인합니다.

2. 다음 명령을 실행하여 유지 관리 기간에서 태그를 제거합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

Linux & macOS

```
aws ssm remove-tags-from-resource \
    --resource-type "MaintenanceWindow" \
    --resource-id "window-id" \
    --tag-key "tag-key"
```

Windows

```
aws ssm remove-tags-from-resource ^
    --resource-type "MaintenanceWindow" ^
    --resource-id "window-id" ^
    --tag-key "tag-key"
```

PowerShell

```
Remove-SSMResourceTag `
    -ResourceType "MaintenanceWindow" `
    -ResourceId "window-id" `
    -TagKey "tag-key
```

명령이 성공하면 출력이 표시되지 않습니다.

3. 다음 명령을 실행하여 유지 관리 기간의 태그를 확인합니다.

Linux & macOS

```
aws ssm list-tags-for-resource \
    --resource-type "MaintenanceWindow" \
    --resource-id "window-id"
```

Windows

```
aws ssm list-tags-for-resource ^
    --resource-type "MaintenanceWindow" ^
    --resource-id "window-id"
```

PowerShell

```
Get-SSMResourceTag `
    -ResourceType "MaintenanceWindow" `
    -ResourceId "window-id"
```

관리형 노드 태깅

이 섹션의 주제에서는 관리형 노드에 태그를 사용하는 방법을 설명합니다.

관리형 노드는 AWS Systems Manager용으로 구성된 머신입니다. 여기에는 Systems Manager용으로 구성된 하이브리드 및 멀티클라우드 환경의 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스 및 비 EC2 시스템이 포함됩니다.

이 주제의 지침은 Systems Manager를 사용하여 관리하는 모든 시스템에 적용될 수 있습니다.

주제

- 태그를 지정하여 관리형 노드 생성 또는 활성화
- 기존 관리형 노드에 태그 추가
- 관리형 노드에서 태그 제거

태그를 지정하여 관리형 노드 생성 또는 활성화

EC2 인스턴스를 생성할 때 태그를 추가할 수 있습니다. 온프레미스 서버 및 VM(가상 머신)을 활성화할 때 태그를 추가할 수 있습니다.

자세한 내용은 다음 주제를 참조하세요.

- EC2 인스턴스는 Amazon EC2 사용 설명서의 Amazon EC2 리소스에 태그 지정 섹션을 참조하세요. (콘텐츠는 Linux와 Windows용 EC2 인스턴스에 모두 적용됨)
- 온프레미스 서버 및 VM은 하이브리드 활성화를 생성하여 Systems Manager에 노드 등록을 참조하세요.

기존 관리형 노드에 태그 추가

Systems Manager 콘솔 또는 명령줄을 사용하여 관리형 노드에 태그를 추가할 수 있습니다.

주제

기존 관리형 노드에 태그 추가(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 태그를 추가할 관리형 노드의 ID를 선택한 다음 태그(Tags) 탭을 선택합니다.

> **Note**
>
> 예상한 관리형 노드가 목록에 없으면 관리형 노드 가용성 문제 해결에서 문제 해결 팁을 참조하세요.

4. 태그(Tags) 섹션에서 편집(Edit)을 선택한 다음 하나 이상의 키 값 태그 페어를 추가합니다.
5. Save(저장)를 선택합니다.

기존 관리형 노드에 태그 추가(명령줄)

기존 관리형 노드에 태그 추가(명령줄)

1. 원하는 명령줄 도구를 사용하여 다음 명령을 실행함으로써 태그를 지정할 수 있는 관리형 노드 목록을 확인합니다.

Linux & macOS

```
aws ssm describe-instance-information
```

Windows

```
aws ssm describe-instance-information
```

PowerShell

```
Get-SSMInstanceInformation
```

태그를 지정할 관리형 노드의 ID를 확인합니다.

> **Note**
>
> 하이브리드 및 멀티클라우드 환경에서 Systems Manager와 함께 사용하려고 등록한 비 EC2 시스템은 `mi-`로 시작합니다(예: `mi-0471e04240EXAMPLE`). EC2 인스턴스에는 `i-`로 시작하는 ID가 있습니다(예: `i-02573cafcfEXAMPLE`).

2. 다음 명령을 실행하여 관리형 노드에 태그를 지정합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

Linux & macOS

```
aws ssm add-tags-to-resource \
    --resource-type "ManagedInstance" \
    --resource-id "instance-id" \
    --tags Key=tag-key,Value=tag-value
```

Windows

```
aws ssm add-tags-to-resource ^
    --resource-type "ManagedInstance" ^
    --resource-id "instance-id" ^
    --tags "Key=tag-key,Value=tag-value"
```

PowerShell

```
$tag = New-Object Amazon.SimpleSystemsManagement.Model.Tag
```

```
$tag.Key = "tag-key"
```

```
$tag.Value = "tag-value"
```

```
Add-SSMResourceTag `
    -ResourceType "ManagedInstance" `
    -ResourceId "instance-id" `
    -Tag $tag `
    -Force
```

명령이 성공하면 출력이 표시되지 않습니다.

3. 다음 명령을 실행하여 관리형 노드 태그를 확인합니다.

Linux & macOS

```
aws ssm list-tags-for-resource \
    --resource-type "ManagedInstance" \
    --resource-id "instance-id"
```

Windows

```
aws ssm list-tags-for-resource ^
    --resource-type "ManagedInstance" ^
    --resource-id "instance-id"
```

PowerShell

```
Get-SSMResourceTag `
    -ResourceType "ManagedInstance" `
    -ResourceId "instance-id"
```

관리형 노드에서 태그 제거

Systems Manager 콘솔이나 명령줄을 사용하여 관리형 노드에서 태그를 제거할 수 있습니다.

주제

관리형 노드에서 태그 제거(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Fleet Manager를 선택합니다.
3. 태그를 제거할 관리형 노드의 이름을 선택한 다음, 태그(Tags) 탭을 선택합니다.
4. 태그(Tags) 섹션에서 편집(Edit)을 선택한 다음, 더 이상 필요하지 않은 태그 페어 옆에 있는 제거(Remove)를 선택합니다.
5. Save(저장)를 선택합니다.

관리형 노드에서 태그 제거(명령줄)

1. 원하는 명령줄 도구로 다음 명령을 실행하여 계정의 관리형 노드를 나열합니다.

   Linux & macOS

   ```
   aws ssm describe-instance-information
   ```

   Windows

   ```
   aws ssm describe-instance-information
   ```

   PowerShell

   ```
   Get-SSMInstanceInformation
   ```

   태그를 제거할 관리형 노드의 이름을 확인합니다.

2. 다음 명령을 실행하여 관리형 노드에서 태그를 제거합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   Linux & macOS

   ```
   aws ssm remove-tags-from-resource \
       --resource-type "ManagedInstance" \
       --resource-id "instance-id" \
       --tag-key "tag-key"
   ```

   Windows

   ```
   aws ssm remove-tags-from-resource ^
       --resource-type "ManagedInstance" ^
       --resource-id "instance-id" ^
       --tag-key "tag-key"
   ```

   PowerShell

   ```
   Remove-SSMResourceTag `
       -ResourceId "instance-id" `
       -ResourceType "ManagedInstance" `
       -TagKey "tag-key" `
       -Force
   ```

   명령이 성공하면 출력이 표시되지 않습니다.

3. 다음 명령을 실행하여 관리형 노드 태그를 확인합니다.

   Linux & macOS

   ```
   aws ssm list-tags-for-resource \
       --resource-type "ManagedInstance" \
       --resource-id "instance-id"
   ```

   Windows

   ```
   aws ssm list-tags-for-resource ^
       --resource-type "ManagedInstance" ^
       --resource-id "instance-id"
   ```

PowerShell

```
Get-SSMResourceTag `
    -ResourceType "ManagedInstance" `
    -ResourceId "instance-id"
```

OpsItems 태그 지정

이 섹션의 주제에서는 OpsItems에 태그를 사용하는 방법을 설명합니다.

주제

- 태그로 OpsItems 생성
- 기존 OpsItems에 태그 추가
- Systems Manager OpsItems에서 태그 제거

태그로 OpsItems 생성

명령줄 도구를 사용하는 경우 태그를 만들 때 사용자 정의 AWS Systems Manager OpsItems에 태그를 추가할 수 있습니다.

자세한 내용은 다음 주제를 참조하십시오.

기존 OpsItems에 태그 추가

명령줄 도구를 사용하여 OpsItems에 태그를 추가할 수 있습니다.

주제

- 기존 OpsItem에 태그 추가(명령줄)

기존 OpsItem에 태그 추가(명령줄)

기존 OpsItem에 태그를 추가하려면(명령줄)

1. 원하는 명령줄 도구를 사용하여 다음 명령을 실행함으로써 태그를 지정할 수 있는 OpsItem 목록을 확인합니다.

Linux & macOS

```
aws ssm describe-ops-items
```

Windows

```
aws ssm describe-ops-items
```

PowerShell

```
Get-SSMOpsItemSummary
```

태그를 지정할 OpsItem의 ID를 확인합니다.

2. 다음 명령을 실행하여 OpsItem에 태그를 지정합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

Linux & macOS

```
aws ssm add-tags-to-resource \
    --resource-type "OpsItem" \
    --resource-id "ops-item-id" \
    --tags "Key=tag-key,Value=tag-value"
```

Windows

```
aws ssm add-tags-to-resource ^
    --resource-type "OpsItem" ^
    --resource-id "ops-item-id" ^
    --tags "Key=tag-key,Value=tag-value"
```

PowerShell

```
$tag = New-Object Amazon.SimpleSystemsManagement.Model.Tag
```

```
$tag.Key = "tag-key"
```

```
$tag.Value = "tag-value"
```

```
Add-SSMResourceTag `
    -ResourceType "OpsItem" `
    -ResourceId "ops-item-id" `
    -Tag $tag `
    -Force
```

명령이 성공하면 출력이 표시되지 않습니다.

3. 다음 명령을 실행하여 OpsItem 태그를 확인합니다.

Linux & macOS

```
aws ssm list-tags-for-resource \
    --resource-type "OpsItem" \
    --resource-id "ops-item-id"
```

Windows

```
aws ssm list-tags-for-resource ^
    --resource-type "OpsItem" ^
    --resource-id "ops-item-id"
```

PowerShell

```
Get-SSMResourceTag `
    -ResourceType "OpsItem" `
    -ResourceId "ops-item-id"
```

Systems Manager OpsItems에서 태그 제거

명령줄 도구를 사용하여 Systems Manager OpsItems에서 태그를 제거할 수 있습니다.

주제

- OpsItems에서 태그 제거(명령줄)

OpsItems에서 태그 제거(명령줄)

1. 원하는 명령줄 도구로 다음 명령을 실행하여 계정의 OpsItems을 나열합니다.

   Linux & macOS

   ```
   aws ssm describe-ops-items
   ```

   Windows

   ```
   aws ssm describe-ops-items
   ```

   PowerShell

   ```
   Get-SSMOpsItemSummary
   ```

   태그를 제거할 OpsItem의 이름을 확인합니다.

2. 다음 명령을 실행하여 OpsItem에서 태그를 제거합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

   Linux & macOS

   ```
   aws ssm remove-tags-from-resource \
       --resource-type "OpsItem" \
       --resource-id "ops-item-id" \
       --tag-key "tag-key"
   ```

   Windows

   ```
   aws ssm remove-tags-from-resource ^
       --resource-type "OpsItem" ^
       --resource-id "ops-item-id" ^
       --tag-key "tag-key"
   ```

   PowerShell

   ```
   Remove-SSMResourceTag `
       -ResourceId "ops-item-id" `
   ```

```
    -ResourceType "OpsItem" `
    -TagKey "tag-key" `
    -Force
```

명령이 성공하면 출력이 표시되지 않습니다.

3. 다음 명령을 실행하여 OpsItem 태그를 확인합니다.

Linux & macOS

```
aws ssm list-tags-for-resource \
    --resource-type "OpsItem" \
    --resource-id "ops-item-id"
```

Windows

```
aws ssm list-tags-for-resource ^
    --resource-type "OpsItem" ^
    --resource-id "ops-item-id"
```

PowerShell

```
Get-SSMResourceTag `
    -ResourceType "OpsItem" `
    -ResourceId "ops-item-id"
```

Systems Manager 파라미터에 태그 지정

이 섹션의 주제에서는 AWS Systems Manager 파라미터(SSM 파라미터)에 태그를 사용하는 방법을 설명합니다.

주제

- 태그를 지정하여 파라미터 만들기
- 기존 파라미터에 태그 추가
- SSM 파라미터에서 태그 제거

태그를 지정하여 파라미터 만들기

SSM 파라미터를 생성할 때 태그를 추가할 수 있습니다.

자세한 내용은 다음 주제를 참조하세요.

- Systems Manager 파라미터 생성(콘솔)
- Systems Manager 파라미터 생성(AWS CLI)
- Systems Manager 파라미터 생성(Tools for Windows PowerShell)

기존 파라미터에 태그 추가

Systems Manager 콘솔 또는 명령줄을 사용하여 소유한 사용자 정의 SSM 파라미터에 태그를 추가할 수 있습니다.

주제

- 기존 파라미터에 태그 추가(콘솔)
- 기존 파라미터에 태그 추가(AWS CLI)
- 기존 파라미터에 태그 추가(AWS Tools for PowerShell)

기존 파라미터에 태그 추가(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Parameter Store를 선택합니다.
3. 이미 생성한 파라미터 이름을 선택한 후 태그 탭을 선택합니다.
4. 첫 번째 상자에서 태그의 키를 입력합니다(예: **`Environment`**).
5. 두 번째 상자에서 태그의 값을 입력합니다(예: **`Test`**).
6. Save(저장)를 선택합니다.

기존 파라미터에 태그 추가(AWS CLI)

1. 원하는 명령줄 도구를 사용하여 다음 명령을 실행함으로써 태그를 지정할 수 있는 파라미터 목록을 확인합니다.

```
aws ssm describe-parameters
```

태그를 지정할 파라미터의 이름을 기록해 둡니다.

2. 다음 명령을 실행하여 파라미터에 태그를 지정합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
aws ssm add-tags-to-resource \
    --resource-type "Parameter" \
    --resource-id "parameter-name" \
    --tags "Key=tag-key,Value=tag-value"
```

명령이 성공하면 출력이 표시되지 않습니다.

3. 다음 명령을 실행하여 파라미터 태그를 확인합니다.

```
aws ssm list-tags-for-resource --resource-type "Parameter" --resource-id
 "parameter-name"
```

기존 파라미터에 태그 추가(AWS Tools for PowerShell)

1. 다음 명령을 실행하여 태그를 지정할 수 있는 파라미터를 나열합니다.

```
Get-SSMParameterList
```

2. 다음 명령을 실행하여 파라미터에 태그를 지정합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
$tag = New-Object Amazon.SimpleSystemsManagement.Model.Tag
```

```
$tag.Key = "tag-key"
```

```
$tag.Value = "tag-value"
```

```
Add-SSMResourceTag `
    -ResourceType "Parameter" `
    -ResourceId "parameter-name" `
    -Tag $tag `
    -Force
```

3. 다음 명령을 실행하여 파라미터 태그를 확인합니다.

```
Get-SSMResourceTag `
    -ResourceType "Parameter" `
    -ResourceId "parameter-name"
```

SSM 파라미터에서 태그 제거

Systems Manager 콘솔이나 명령줄을 사용하여 SSM 파라미터에서 태그를 제거할 수 있습니다.

주제

- SSM 파라미터에서 태그 제거(콘솔)
- SSM 파라미터에서 태그 제거(명령줄)

SSM 파라미터에서 태그 제거(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Parameter Store를 선택합니다.
3. 태그를 제거할 파라미터의 이름을 선택한 다음, 태그(Tags) 탭을 선택합니다.
4. 더 이상 필요하지 않은 태그 페어 옆에 있는 제거(Remove)를 선택합니다.
5. Save(저장)를 선택합니다.

SSM 파라미터에서 태그 제거(명령줄)

1. 원하는 명령줄 도구로 다음 명령을 실행하여 계정의 파라미터를 나열합니다.

   Linux & macOS

   ```
   aws ssm describe-parameters
   ```

   Windows

   ```
   aws ssm describe-parameters
   ```

PowerShell

```
Get-SSMParameterList
```

태그를 제거할 파라미터의 이름을 확인합니다.

2. 다음 명령을 실행하여 파라미터에서 태그를 제거합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

Linux & macOS

```
aws ssm remove-tags-from-resource \
    --resource-type "Parameter" \
    --resource-id "parameter-name" \
    --tag-key "tag-key"
```

Windows

```
aws ssm remove-tags-from-resource ^
    --resource-type "Parameter" ^
    --resource-id "parameter-name" ^
    --tag-key "tag-key"
```

PowerShell

```
Remove-SSMResourceTag
    -ResourceId "parameter-name"
    -ResourceType "Parameter"
    -TagKey "tag-key
```

명령이 성공하면 출력이 표시되지 않습니다.

3. 다음 명령을 실행하여 문서 태그를 확인합니다.

Linux & macOS

```
aws ssm list-tags-for-resource \
    --resource-type "Parameter" \
    --resource-id "parameter-name"
```

Windows

```
aws ssm list-tags-for-resource ^
    --resource-type "Parameter" ^
    --resource-id "parameter-name"
```

PowerShell

```
Get-SSMResourceTag `
    -ResourceType "Parameter" `
    -ResourceId "parameter-name"
```

패치 기준 태깅

이 단원의 주제에서는 패치 기준에 태그를 지정하는 방법에 대해 설명합니다.

주제

- 태그를 지정하여 패치 기준 만들기
- 기존 패치 기준에 태그 추가
- 패치 기준에서 태그 제거

태그를 지정하여 패치 기준 만들기

AWS Systems Manager 패치 기준을 생성할 때 태그를 추가할 수 있습니다.

자세한 내용은 다음 주제를 참조하세요.

- 사용자 정의 패치 기준 작업
- 패치 기준선 생성
- 다른 OS 버전에 대한 사용자 지정 리포지토리가 있는 패치 기준 생성

기존 패치 기준에 태그 추가

Systems Manager 콘솔 또는 명령줄을 사용하여 소유한 패치 기준에 태그를 추가할 수 있습니다.

주제

- 기존 패치 기준에 태그 추가(콘솔)
- 기존 패치 기준(AWS CLI)에 태그 추가
- 패치 기준 태그 지정(AWS Tools for PowerShell)

기존 패치 기준에 태그 추가(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Patch Manager를 선택합니다.
3. 이미 만든 사용자 정의 패치 기준의 이름을 선택하고 테이블 태그 지정(Tags table) 섹션까지 아래로 스크롤한 다음, 태그 편집(Edit tags)을 선택합니다.
4. 태그 추가(Add tag)를 선택합니다.
5. 키(Key)에 태그 키를 입력합니다(예: **`Environment`**).
6. 값(Value)에 태그 값을 입력합니다(예: **`Test`**).
7. Save changes(변경 사항 저장)를 선택합니다.

기존 패치 기준(AWS CLI)에 태그 추가

1. 원하는 명령줄 도구를 사용하여 다음 명령을 실행함으로써 태그를 지정할 수 있는 패치 기준 목록을 확인합니다.

   ```
   aws ssm describe-patch-baselines --filters "Key=OWNER,Values=[Self]"
   ```

   태그를 지정할 패치 기준의 ID를 확인합니다.

2. 다음 명령을 실행하여 패치 기준에 태그를 지정합니다. 각 *`example resource placeholder`*를 사용자의 정보로 바꿉니다.

   Linux & macOS

   ```
   aws ssm add-tags-to-resource \
       --resource-type "PatchBaseline" \
       --resource-id "baseline-id" \
       --tags "Key=tag-key,Value=tag-value"
   ```

Windows

```
aws ssm add-tags-to-resource ^
    --resource-type "PatchBaseline" ^
    --resource-id "baseline-id" ^
    --tags "Key=tag-key,Value=tag-value"
```

명령이 성공하면 출력이 표시되지 않습니다.

3. 다음 명령을 실행하여 패치 기준 태그를 확인합니다.

Linux & macOS

```
aws ssm list-tags-for-resource \
    --resource-type "PatchBaseline" \
    --resource-id "baseline-id"
```

Windows

```
aws ssm list-tags-for-resource ^
    --resource-type "PatchBaseline" ^
    --resource-id "patchbaseline-id"
```

패치 기준 태그 지정(AWS Tools for PowerShell)

1. 다음 명령을 실행하여 태그를 지정할 수 있는 패치 기준을 나열합니다.

```
Get-SSMPatchBaseline
```

2. 다음 명령을 실행하여 패치 기준에 태그를 지정합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

```
$tag = New-Object Amazon.SimpleSystemsManagement.Model.Tag
```

```
$tag.Key = "tag-key"
```

```
$tag.Value = "tag-value"
```

```
Add-SSMResourceTag `
    -ResourceType "PatchBaseline" `
    -ResourceId "baseline-id" `
    -Tag $tag `
    -Force
```

3. 다음 명령을 실행하여 패치 기준 태그를 확인합니다.

```
Get-SSMResourceTag `
    -ResourceType "PatchBaseline" `
    -ResourceId "baseline-id"
```

패치 기준에서 태그 제거

Systems Manager 콘솔이나 명령줄을 사용하여 패치 기준에서 태그를 제거할 수 있습니다.

주제

- 패치 기준에서 태그 제거(콘솔)
- 패치 기준에서 태그 제거(명령줄)

패치 기준에서 태그 제거(콘솔)

1. AWS Systems Manager 콘솔(https://console.aws.amazon.com/systems-manager/)을 엽니다.
2. 탐색 창에서 Patch Manager를 선택합니다.
3. 제거할 패치 기준의 이름을 선택하고 테이블 태그 지정(Tags table) 섹션까지 아래로 스크롤한 다음, 태그 편집(Edit tags) 탭을 선택합니다.
4. 더 이상 필요하지 않은 태그 페어 옆에 있는 태그 제거(Remove tag)를 선택합니다.
5. Save changes(변경 사항 저장)를 선택합니다.

패치 기준에서 태그 제거(명령줄)

1. 원하는 명령줄 도구로 다음 명령을 실행하여 계정의 패치 기준을 나열합니다.

Linux & macOS

```
aws ssm describe-patch-baselines
```

Windows

```
aws ssm describe-patch-baselines
```

PowerShell

```
Get-SSMPatchBaseline
```

태그를 제거할 패치 기준의 ID를 확인합니다.

2. 다음 명령을 실행하여 패치 기준에서 태그를 제거합니다. 각 *example resource placeholder*를 사용자의 정보로 바꿉니다.

Linux & macOS

```
aws ssm remove-tags-from-resource \
    --resource-type "PatchBaseline" \
    --resource-id "baseline-id" \
    --tag-key "tag-key"
```

Windows

```
aws ssm remove-tags-from-resource ^
    --resource-type "PatchBaseline" ^
    --resource-id "baseline-id" ^
    --tag-key "tag-key"
```

PowerShell

```
Remove-SSMResourceTag `
    -ResourceType "PatchBaseline" `
    -ResourceId "baseline-id" `
    -TagKey "tag-key
```

명령이 성공하면 출력이 표시되지 않습니다.

3. 다음 명령을 실행하여 패치 기준 태그를 확인합니다.

Linux & macOS

```
aws ssm list-tags-for-resource \
    --resource-type "PatchBaseline" \
    --resource-id "baseline-id"
```

Windows

```
aws ssm list-tags-for-resource ^
    --resource-type "PatchBaseline" ^
    --resource-id "baseline-id"
```

PowerShell

```
Get-SSMResourceTag `
    -ResourceType "PatchBaseline" `
    -ResourceId "baseline-id"
```

AWS Systems Manager 참조

다음 정보와 항목은 AWS Systems Manager 솔루션을 더욱 잘 구현하는 데 도움이 될 수 있습니다.

보안 주체

AWS Identity and Access Management(IAM)에서는 보안 주체 정책 요소를 사용하여 리소스에 대한 서비스 액세스를 부여하거나 거부할 수 있습니다. Systems Manager의 보안 주체 정책 요소 값은 `ssm.amazonaws.com`입니다.

지원되는 AWS 리전 및 엔드포인트

Amazon Web Services 일반 참조의 Systems Manager 서비스 엔드포인트를 참조하세요.

Service Quotas

Amazon Web Services 일반 참조의 Systems Manager 서비스 할당량을 참조하세요.

API Reference

AWS Systems Manager API Reference를 참조하세요.

AWS CLI 명령 참조

AWS CLI Command Reference의 AWS Systems Manager 섹션을 참조하세요.

AWS Tools for PowerShell Cmdlet 참조

AWS Tools for PowerShell Cmdlet Reference의 AWS Systems Manager 섹션을 참조하세요.

GitHub의 SSM Agent 리포지토리

aws/amazon-ssm-agent를 참조하십시오.

질문하기

AWS re:Post에서 Systems Manager 문제

AWS 뉴스 블로그

관리 도구

기타 참조 주제

- 참조: Systems Manager용 Amazon EventBridge 이벤트 패턴 및 유형

- 참조: Systems Manager용 Cron 및 Rate 표현식
- 참조: ec2messages, ssmmessages 및 기타 API 작업
- 참조: Systems Manager용 형식이 지정된 날짜 및 시간 문자열 생성

참조: Systems Manager용 Amazon EventBridge 이벤트 패턴 및 유형

> **Note**
>
> Amazon EventBridge는 이벤트를 관리하는 데 선호되는 방법입니다. CloudWatch Events와 EventBridge는 기본 서비스 및 API가 동일하지만 EventBridge가 더 많은 기능을 제공합니다. CloudWatch 또는 EventBridge에서 변경한 내용은 각 콘솔에 반영됩니다. 자세한 내용은 Amazon EventBridge 사용 설명서를 참조하세요.

Amazon EventBridge를 사용하여 수신 이벤트와 일치하는 규칙을 생성하고 처리 대상으로 라우팅할 수 있습니다.

이벤트는 자체 애플리케이션, 서비스형 소프트웨어(SaaS) 애플리케이션 또는 AWS 서비스의 환경 변경을 나타냅니다. 이벤트는 최선의 작업을 기반으로 생성됩니다. 규칙에 지정된 이벤트 유형이 감지된 후 EventBridge는 처리를 위해 이를 지정된 대상으로 라우팅합니다. 대상에는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스, AWS Lambda 기능, Amazon Kinesis Streams, Amazon Elastic Container Service(Amazon ECS) 태스크, AWS Step Functions 상태 시스템, Amazon Simple Notification Service(Amazon SNS) 주제, Amazon Simple Queue Service(Amazon SQS) 대기열, 기본 제공 대상 등이 포함될 수 있습니다.

EventBridge 규칙 생성에 대한 자세한 내용은 다음 주제를 참조하세요.

- Amazon EventBridge로 Systems Manager 이벤트 모니터링
- Systems Manager에 대한 Amazon EventBridge 이벤트 예
- Amazon EventBridge 사용 설명서의 Amazon EventBridge 시작하기를 참조하세요.

이 주제의 나머지 부분에서는 EventBridge 규칙에 포함할 수 있는 Systems Manager 이벤트 유형에 대해 설명합니다.

이벤트 유형: Automation

이벤트 유형 이름	규칙에 추가할 수 있는 이벤트에 대한 설명
EC2 Automation 실행 상태 변경 알림	Automation 워크플로의 전체 상태가 변경됩니다. 이벤트 규칙에 다음 상태 변경 중 하나를 추가할 수 있습니다. • Approved • 취소됨 • Failed • PendingApproval • PendingChangeCalendarOverride • 거부됨 • 예약됨 • Success • TimedOut
EC2 Automation 단계 상태 변경 알림	Automation 워크플로의 특정 단계 상태가 변경됩니다. 이벤트 규칙에 다음 상태 변경 중 하나를 추가할 수 있습니다. • 취소됨 • Failed • Success • TimedOut

이벤트 유형: Change Calendar

이벤트 유형 이름	규칙에 추가할 수 있는 이벤트에 대한 설명
일정 상태 변경	Change Calendar의 상태가 변경됩니다. 이벤트 규칙에 다음 상태 변경 중 하나 또는 둘 다를 추가할 수 있습니다.

이벤트 유형 이름	규칙에 추가할 수 있는 이벤트에 대한 설명
	• OPEN • CLOSED 다른 AWS 계정에서 공유된 일정의 상태 변경은 지원되지 않습니다.

이벤트 유형: Change Manager

이벤트 유형 이름	규칙에 추가할 수 있는 이벤트에 대한 설명
변경 요청 상태 업데이트	Change Manager 변경 요청의 상태입니다. 이벤트 규칙에는 다음 상태를 사용할 수 있습니다. • Approved • 거부됨 • InProgress

이벤트 유형: Configuration Compliance

이벤트 유형 이름	규칙에 추가할 수 있는 이벤트에 대한 설명
Configuration Compliance 상태 변경	연결 규정 준수 또는 패치 규정 준수에 대한 관리형 노드의 상태가 변경됩니다. 이벤트 규칙에 다음 상태 변경 중 하나를 추가할 수 있습니다. • compliant • non_compliant

이벤트 유형: Inventory

이벤트 유형 이름	규칙에 추가할 수 있는 이벤트에 대한 설명
인벤토리 리소스 상태 변경	사용자 정의 인벤토리 삭제 및 이전 스키마 버전을 사용하는 PutInventory 호출입니다. 이벤트 규칙에 다음 상태 변경 중 하나를 추가할 수 있습니다. • 특정 노드의 사용자 정의 인벤토리 유형 삭제 이벤트입니다. EventBridge는 사용자 정의 인벤토리 유형별로 노드당 하나의 이벤트를 보냅니다. • 모든 노드에 대한 사용자 정의 인벤토리 유형 삭제 이벤트입니다. • 이전 스키마 버전 이벤트가 있는 PutInventory 호출입니다. EventBridge는 스키마 버전이 현재 스키마보다 낮을 때 이 이벤트를 보냅니다. 이 이벤트는 모든 인벤토리 유형에 적용됩니다. 자세한 내용은 Inventory 이벤트의 EventBridge 모니터링 정보 단원을 참조하십시오.

이벤트 유형: Maintenance Window

이벤트 유형 이름	규칙에 추가할 수 있는 이벤트에 대한 설명
Maintenance Window 상태 변경 알림	하나 이상의 유지 관리 기간의 전체 상태가 변경됩니다. 이벤트 규칙에 다음 상태 변경 중 하나를 추가할 수 있습니다. • DISABLED • ENABLED

이벤트 유형 이름	규칙에 추가할 수 있는 이벤트에 대한 설명
Maintenance Window 대상 등록 알림	하나 이상의 유지 관리 기간 대상 상태가 변경됩니다. 이벤트 규칙에 다음 상태 변경 중 하나를 추가할 수 있습니다. • DEREGISTERED • REGISTERED • UPDATED
Maintenance Window 실행 상태 변경 알림	유지 관리 기간이 실행되는 동안 전체 상태가 변경됩니다. 이벤트 규칙에 다음 상태 변경 중 하나를 추가할 수 있습니다. • 취소됨 • 취소 중 • 실패함 • IN_PROGRESS • 대기 중 • SKIPPED_OVERLAPPING • 성공 • TIMED_OUT
Maintenance Window 태스크 실행 상태 변경 알림	유지 관리 기간이 실행되는 동안 태스크의 상태가 변경됩니다. 이벤트 규칙에 다음 상태 변경 중 하나를 추가할 수 있습니다. • 취소됨 • 취소 중 • 실패함 • IN_PROGRESS • 성공 • TIMED_OUT

이벤트 유형 이름	규칙에 추가할 수 있는 이벤트에 대한 설명
Maintenance Window 태스크 대상 호출 상태 변경 알림	특정 대상에 대한 유지 관리 기간 태스크의 상태가 변경됩니다. 이 알림은 Run Command 작업에 대해서만 완전히 지원됩니다. 이런 유형의 태스크에서, 이벤트 규칙에 다음 상태 변경 중 하나를 추가할 수 있습니다. • 취소됨 • 취소 중 • 실패함 • IN_PROGRESS • 성공 • TIMED_OUT 자동화를 위해 AWS Lambda, AWS Step Functions 태스크, EventBridge는 `IN_PROGRESS` 과 `COMPLETE`의 상태만 보고합니다. `COMPLETE`는 태스크의 성공 여부를 보고합니다.
Maintenance Window 태스크 등록 알림	하나 이상의 유지 관리 기간 태스크의 상태가 변경됩니다. 이벤트 규칙에 다음 상태 변경 중 하나를 추가할 수 있습니다. • DEREGISTERED • REGISTERED • UPDATED

이벤트 유형: OpsCenter

이벤트 유형 이름	규칙에 추가할 수 있는 이벤트에 대한 설명
OpsItem 생성	OpsItem이 생성될 때 발생합니다. 다음 OpsItem 유형 중 하나에 규칙을 추가할 수 있습니다. • /aws/issue • /aws/task • /aws/insight • /aws/actionitem
OpsItem 업데이트	OpsItem이 업데이트될 때 발생합니다. 다음 OpsItem 유형 중 하나에 규칙을 추가할 수 있습니다. • /aws/issue • /aws/task • /aws/insight • /aws/actionitem

이벤트 유형: Parameter Store

이벤트 유형 이름	규칙에 추가할 수 있는 이벤트에 대한 설명
파라미터 스토어 변경	파라미터의 상태가 변경됩니다. 이벤트 규칙에 다음 상태 변경 중 하나를 추가할 수 있습니다. • 생성 • 업데이트 • 삭제 • LabelParameterVersion

이벤트 유형 이름	규칙에 추가할 수 있는 이벤트에 대한 설명
	자세한 내용은 파라미터 및 파라미터 정책에 대해 EventBridge 구성 단원을 참조하십시오.
파라미터 스토어 정책 작업	고급 파라미터 정책 변경 조건이 충족됩니다. 이벤트 규칙에 다음 상태 변경 중 하나를 추가할 수 있습니다. • Expiration • ExpirationNotification • NoChangeNotification 자세한 내용은 파라미터 및 파라미터 정책에 대해 EventBridge 구성 단원을 참조하십시오.

이벤트 유형: Run Command

이벤트 유형 이름	규칙에 추가할 수 있는 이벤트에 대한 설명
EC2 명령 호출 상태 변경 알림	개별 관리형 인스턴스로 전송된 명령의 상태가 변경됩니다. 이벤트 규칙에 다음 상태 변경 중 하나를 추가할 수 있습니다. • Success • InProgress • TimedOut • 취소됨 • Failed
EC2 명령 상태 변경 알림	명령의 전체 상태가 변경됩니다. 이벤트 규칙에 다음 상태 변경 중 하나를 추가할 수 있습니다. • Success • InProgress • TimedOut

이벤트 유형 이름	규칙에 추가할 수 있는 이벤트에 대한 설명
	• 취소됨 • Failed

이벤트 유형: State Manager

이벤트 유형 이름	규칙에 추가할 수 있는 이벤트에 대한 설명
EC2 State Manager 연결 상태 변경	연결이 적용되면서 전체 연결 상태가 변경됩니다. 이벤트 규칙에 다음 상태 변경 중 하나를 추가할 수 있습니다. • Failed • 보류중 • Success
EC2 State Manager 인스턴스 연결 상태 변경	연결의 대상이 되는 단일 관리형 인스턴스의 상태가 변경됩니다. 이벤트 규칙에 다음 상태 변경 중 하나를 추가할 수 있습니다. • Failed • 보류중 • Success

참조: Systems Manager용 Cron 및 Rate 표현식

State Manager에 유지 관리 기간 또는 AWS Systems Manager 연결을 생성할 때는 창 또는 연결이 작동해야 할 시점에 대한 일정을 지정합니다. cron 표현식이라는 시간 기반 항목이나 rate 표현식이라는 빈도 기반 항목으로 일정을 지정할 수 있습니다.

Cron 및 Rate 표현식에 대한 일반 정보

다음 정보는 유지 관리 기간 및 연결 모두에 대해 cron 및 rate 표션식에 적용됩니다.

단일 실행 일정

연결 유지 관리 기간을 생성할 때 지정된 시간에 한 번 실행되도록 협정 세계 표준시(UTC) 형식으로 타임스탬프를 지정할 수 있습니다. 예: `"at(2020-07-07T15:55:00)"`

일정 오프셋

연결 및 유지 관리 기간은 cron 표현식에 대해서만 일정 오프셋을 지원합니다. 일정 오프셋은 연결 또는 유지 관리 기간을 실행하기 전에 cron 표현식에 의해 지정된 날짜 및 시간 이후에 대기할 일 수입니다.

Maintenance window example

다음 명령에서 cron 표현식은 매월 셋째 화요일 오후 11:30에 실행되도록 유지 관리 기간을 예약합니다. 그러나, 일정 오프셋이 2인 경우 2일 후 오후 11시 30분까지 유지 관리 기간이 실행되지 않습니다.

```
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

다음 명령에서 cron 표현식은 매월 두 번째 목요일에 실행되도록 연결을 예약합니다. 하지만 스케줄 오프셋이 3이므로 3일 후인 다음 일요일까지 연결이 실행되지 않습니다.

```
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**
>
> 연결과 함께 오프셋을 사용하려면 `--apply-only-at-cron-interval` 옵션을 지정해야 합니다. 이 옵션은 시스템에 연결을 만든 직후에 연결을 실행하지 말라고 알려줍니다.

현재 기간에 이미 경과한 날짜를 대상으로 하는 cron 표현식을 사용하여 연결 또는 유지 관리 기간을 생성하지만, 미래에 해당하는 일정 오프셋 날짜를 추가하면 연결 또는 유지 관리 기간이 해당 기간에 실행되지 않습니다. 다음 기간에 적용됩니다. 예를 들어 어제 유지 관리 기간을 실행한 cron 표현식을 지정하고 일정 오프셋을 2일로 추가하면 유지 관리 기간은 내일 실행되지 않습니다.

필수 필드

유지 관리 기간에 대한 cron 표현식에는 필수 필드가 6개 있습니다. 연결에 대한 cron 표현식에는 5개가 있습니다. (State Manager은(는) 현재 연결에 대한 cron 표현식에서 월 지정을 지원하지 않습니다.) 추가 필드인 Seconds 필드(cron 표현식의 첫 번째 필드)는 선택 사항입니다. 각 필드는 공백으로 구분됩니다.

Cron 표현식의 예

분	시간	일	월	요일	연도	의미
0	10	*	*	?	*	매일 오전 10시(UTC)에 실행
15	12	*	*	?	*	매일 오후 12시 15분(UTC)에 실행
0	18	?	*	월-금	*	매주 월요일부터 금요일까지 오후 6시(UTC)에 실행
0	8	1	*	?	*	매월 1일 오전 8시(UTC)에 실행

지원되는 값

다음 표에는 필수적인 cron 항목에 대해 지원되는 값이 나와 있습니다.

Cron 표현식에 대해 지원되는 값

필드	값	와일드카드
분	0-59	, - * /
시간	0-23	, - * /
날짜	1-31	, - * ? / L W
월(유지 관리 기간에만 해당)	1–12 또는 JAN-DEC	, - * /
요일	1–7 또는 SUN-SAT	, - * ? / L #
연도	1970-2199	, - * /

> **Note**
>
> 동일한 cron 표현식에서 월 필드와 주 필드 모두에 값을 지정할 수는 없습니다. 이들 필드 중 하나에 값을 지정하는 경우에는 다른 필드에서 반드시 ?(물음표)를 사용합니다.

cron 표현식에 대한 와일드카드

다음 표에는 cron 표현식에서 지원되는 와일드카드 값이 나와 있습니다.

> **Note**
>
> 5분보다 빠른 속도로 이어지는 cron 표현식은 지원되지 않습니다. day-of-week 및 day-of-month 값을 모두 지정하기 위한 지원 기능은 완전하지 않습니다. 이들 필드 중 하나에 ?(물음표)를 사용합니다.

Cron 표현식에 대해 지원되는 와일드카드

와일드카드	설명
,	,(쉼표) 와일드카드는 추가 값을 포함합니다. 예컨대, Month 필드에서 JAN, FEB, MAR은 1월, 2월, 3월을 포함한다는 의미입니다.
-	-(대시) 와일드카드는 범위를 지정합니다. 예컨대, Day 필드에서 1-15는 지정된 달의 1일에서 15일까지 포함한다는 의미입니다.
*	*(별표) 와일드카드는 필드의 모든 값을 포함합니다. 예컨대, 시간 필드에서 *는 모든 시간을 포함한다는 의미입니다.
/	/(슬래시) 와일드카드로 증분을 지정합니다. 분 필드에서 1/10을 입력하면 지정한 시간의 1분부터 시작해서 매 10분 간격을 지정할 수 있습니다. 따라서 1/10은 첫 번째, 11번째, 21번째, 31번째 분 등을 지정합니다.
?	?(물음표) 와일드카드는 어떤 한 가지나 다른 것을 지정합니다. Day-of-month 필드에 7을 입력하고 Day-of-week 필드에는 ?을 입력하면 매월 7일이 무슨 요일이든 상관없이 7번째 되는 날을 지정한다는 의미입니다.
L	월(Day-of-month) 필드 또는 주(Day-of-week) 필드에는 L 와일드카드로 해당 월 또는 주의 마지막 날을 지정할 수 있습니다.
W	'날짜' 필드에서는 W 와일드카드로 어떤 한 평일을 지정할 수 있습니다. 예컨대, Day-of-month 필드에 3W를 입력하면 해당 월의 세 번째 평일에 가장 가까운 날을 지정할 수 있습니다.

와일드카드	설명
#	요일 필드의 # 와일드카드 뒤에 1~5 사이의 숫자는 해당 월의 지정된 요일을 지정합니다. 5#3은 해당 월의 셋째 목요일을 지정합니다.

rate 표현식

rate 식에는 다음과 같은 필수 필드가 2개 있습니다. 각 필드는 공백으로 구분됩니다.

Rate 표현식의 필수 필드

필드	값
값	양수(예: 1 또는 15)
단위	`minute` `minutes` `hour` `hours` `day` `days`

값이 1(와)과 같을 경우에는 단위가 단수여야 합니다. 마찬가지로, 1보다 큰 값에 대해서는 단위가 복수여야 합니다. 예를 들어 `rate(1 hours)`와 `rate(5 hour)`는 유효하지 않으며 `rate(1 hour)`와 `rate(5 hours)`는 유효합니다.

주제

- 연결에 대한 Cron 및 Rate 표현식
- 유지 관리 기간에 대한 Cron 및 Rate 표현식

연결에 대한 Cron 및 Rate 표현식

이 단원에는 State Manager 연결에 대한 cron 및 rate 표현식의 예제가 포함되어 있습니다. 이러한 표현식 중 하나를 생성하기 전에 다음 정보에 유의해야 합니다.

- 연결은 1/2, 1, 2, 4, 8 또는 12시간마다, 매일, 매주, 또는 매주 특정 날짜 및 시간, 매월 특정 주의 특정 날짜나 달의 마지막 x일의 특정 시간과 같은 cron 표현식을 지원합니다.
- 연결은 30분 이상 및 31일 미만의 간격과 같은 rate 표현식을 지원합니다.
- 옵션인 Seconds 필드를 지정한 경우 값으로 0을 사용할 수 있습니다. 예: `cron(0 */30 * * * ? *)`
- AWS Systems Manager의 기능인 인벤토리에 대한 메타데이터를 수집하는 연결의 경우 rate 표현식을 사용하는 것이 좋습니다.
- State Manager은(는) 현재 연결에 대한 cron 표현식에서 월 지정을 지원하지 않습니다.

연결은 요일 및 숫자 기호(#)를 포함하는 cron 표현식을 지원함으로써 매달 n 번째 일을 지정하여 연결을 실행합니다. 매월 셋째 주 화요일 UTC 23:30에 cron 일정을 실행하는 예는 다음과 같습니다.

```
cron(30 23 ? * TUE#3 *)
```

매월 두 번째 목요일 UTC 자정에 실행하는 예는 다음과 같습니다.

```
cron(0 0 ? * THU#2 *)
```

또한 연결은 (L) 기호를 지원하여 매월 마지막 X 일을 표시합니다. 매월 마지막 화요일 UTC 자정에 cron 일정을 실행하는 예는 다음과 같습니다.

```
cron(0 0 ? * 3L *)
```

예를 들어, 화요일을 패치한 지 2일 후 연결을 실행하려는 경우 연결 실행 시기를 추가로 제어하려면 오프셋을 지정할 수 있습니다. 오프셋을 통해 연결을 실행하도록 예약된 날짜 이후에 대기할 일 수를 정의할 수 있습니다. 예를 들어, `cron(0 0 ? * THU#2 *)`의 cron 일정을 지정한 경우 일정 오프셋 필드에 숫자 3을 지정하여 매월 두 번째 목요일 이후 매주 일요일에 연결을 실행할 수 있습니다.

오프셋을 사용하려면 콘솔에서 지정된 다음 Cron 간격에서만 연결 적용(Apply association only at the next specified Cron interval) 옵션을 선택하거나 명령줄에서 `--apply-only-at-cron-interval` 파라미터를 지정해야 합니다. 이 옵션은 State Manager에게 연결을 만든 직후에 연결을 실행하지 말라고 알려줍니다.

다음 표에는 연결에 대한 cron 예제가 나와 있습니다.

연결에 대한 Cron 예제

예	Details
cron(0/30 * * * ? *)	30분마다
cron(0 0/1 * * ? *)	매 시간
cron(0 0/2 * * ? *)	2시간마다
cron(0 0/4 * * ? *)	4시간마다
cron(0 0/8 * * ? *)	8시간마다
cron(0 0/12 * * ? *)	12시간마다
cron(15 13 ? * * *)	매일 오후 1시 15분
cron(15 13 ? * MON *)	매주 월요일 오후 1시 15분
cron(30 23 ? * TUE#3 *)	매월 세 번째 화요일 오후 11:30

다음은 연결에 대한 몇 가지 rate 예제입니다.

연결에 대한 Rate 예제

예	Details
rate(30 minutes)	30분마다
rate(1 hour)	매 시간
rate(5 hours)	5시간마다
rate(15 days)	15일마다

연결에 대한 AWS CLI 예제

AWS CLI를 사용하여 State Manager 연결을 생성하려면 `--schedule-expression` 파라미터를 cron 또는 rate 표현식과 함께 포함합니다. 다음 예에서는 로컬 Linux 시스템에서 AWS CLI를 사용합니다.

> **Note**
>
> 기본적으로 새 연결을 만들면 새 연결이 생성된 직후에 실행되고 그 후에 지정된 일정에 따라 실행됩니다. 연결을 생성한 직후 연결이 실행되지 않도록 `--apply-only-at-cron-interval`을 지정합니다. 이 파라미터는 rate 표현식에 대해 지원되지 않습니다.

```
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
```

유지 관리 기간에 대한 Cron 및 Rate 표현식

이 단원에는 유지 관리 기간에 대한 cron 및 rate 표현식의 예제가 포함되어 있습니다.

State Manager 연결과 달리 유지 관리 기간은 cron 및 rate 표현식 중 일부를 지원하지 않습니다. 여기에는 초 필드의 값에 대한 지원이 포함됩니다.

예를 들어 다음 6필드 cron 표현식은 매일 오전 9시 30분에 유지 관리 기간을 실행합니다.

```
cron(30 09 ? * * *)
```

Seconds 필드에 값을 추가하여 다음 7필드 cron 표현식은 매일 오전 9시 30분 24초에 유지 관리 기간을 실행합니다.

```
cron(24 30 09 ? * * *)
```

다음 표에는 유지 관리 기간에 대한 추가 6필드 cron 예제가 나와 있습니다.

유지 관리 기간에 대한 Cron 예제

예	Details
cron(0 2 ? * THU#3 *)	매월 셋째 목요일 오전 2시
cron(15 10 ? * * *)	매일 오전 10시 15분
cron(15 10 ? * MON-FRI *)	매주 월요일, 화요일, 수요일, 목요일, 금요일 오전 10시 15분
cron(0 2 L * ? *)	매월 말일 오전 2시
cron(15 10 ? * 6L *)	매월 마지막 금요일 오전 10시 15분

다음 표에는 유지 관리 기간에 대한 rate 예제가 나와 있습니다.

유지 관리 기간에 대한 Rate 예제

예	Details
rate(30 minutes)	30분마다
rate(1 hour)	매 시간
rate(5 hours)	5시간마다
rate(25 days)	25일마다

유지 관리 기간에 대한 예제AWS CLI

AWS CLI를 사용하여 유지 관리 기간을 생성하려면 `--schedule` 파라미터를 cron 또는 rate 표현식 또는 타임스탬프와 함께 포함합니다. 다음 예에서는 로컬 Linux 시스템에서 AWS CLI를 사용합니다.

```
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
```

추가 정보

위키백과 웹사이트의 CRON 표현식

참조: ec2messages, ssmmessages 및 기타 API 작업

API 작업을 모니터링하는 경우 다음 작업에 대한 직접 호출을 볼 수 있습니다.

- 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

이 주제의 나머지 부분에서 설명하는 대로, AWS Systems Manager에서 사용하는 특별한 작업입니다.

에이전트 관련 API 작업(ssmmessages및 ec2messages 엔드포인트)

ssmmessages API 작업

Systems Manager는 ssmmessages 엔드포인트를 사용하여 다음 두 가지 유형의 API 작업을 수행합니다.

- SSM Agent가 클라우드에서 수행하는 AWS Systems Manager의 기능인 Session Manager 작업입니다. 이 엔드포인트는 클라우드의 Session Manager 서비스를 사용하여 세션 채널을 생성하고 삭제하는 데 필요합니다. 또한 연결이 허용되는 경우 Amazon Message Gateway Service를 통해 Command 문서가 SSM Agent에 수신됩니다. 연결이 허용되지 않는 경우 Amazon Message Delivery Service를 통해 Command 문서가 SSM Agent에 수신됩니다. 자세한 내용은 Amazon Session Manager Message Gateway Service를 위한 작업, 리소스 및 조건 키를 참조하세요.
- Systems Manager Agent(SSM Agent)가 클라우드에서 수행하는 Systems Manager 서비스 작업입니다.

ec2messages API 작업

ec2messages:* API 작업은 Amazon Message Delivery Service 엔드포인트를 수행합니다. Systems Manager는 이 Systems Manager Agent(SSM Agent)에서 API 작업용 엔드포인트를 클라우드의 Systems Manager 서비스로 사용합니다.

> **Important**
>
> ec2messages:* API 작업은 2024년 이전에 릴리스된 AWS 리전에서만 지원됩니다. 2024년 이후 출시된 리전에서는 ssmmessages:* API 작업만 지원됩니다.

엔드포인트 연결 우선순위

SSM Agent 버전 3.3.40.0부터 Systems Manager는 가능한 경우 ec2messages:* 엔드포인트(Amazon Message Delivery Service) 대신 ssmmessages:* 엔드포인트(Amazon Message Gateway Service)를 사용하기 시작했습니다.

AWS Identity and Access Management (IAM) 권한 정책에서 ssmmessages:*에 대한 액세스를 제공하는 경우, IAM 인스턴스 프로파일이 두 엔드포인트를 모두 허용하도록 구성되어 있더라도 SSM Agent가 ssmmessages:* 엔드포인트에 연결됩니다. 여기에는 사용자가 직접 생성한 IAM 인스턴스 프로파일과 IAM 서비스 역할, 그리고 Quick Setup 호스트 관리 구성 및 기본 호스트 관리 구성으로 생성한 IAM 인스턴스 프로파일에 대한 정책이 포함됩니다.

두 엔드포인트에 대한 권한을 부여하고, 예를 들어 CloudWatch 지표를 사용하여 API 작업을 모니터링하는 경우 ec2messages:*에 대한 호출이 표시되지 않습니다.

2024년 이전에 출시된 AWS 리전의 경우: 현재 정책에서 ec2messages:* 권한을 안전하게 제거할 수 있습니다.

엔드포인트 연결 장애 조치

2024년 이전에 출시된 AWS 리전의 경우에만: 에이전트를 시작할 때 IAM 인스턴스 프로파일에서 ssmmessages:*에 대한 권한을 제공하지 않고 ec2messages:*만 제공하는 경우 SSM Agent에서는 ec2messages:* 엔드포인트에 연결합니다. SSM Agent 시작 시점에 ssmmessages:* 및 ec2messages:* 둘 다 있지만 에이전트가 시작된 후에 ssmmessages:*가 제거될 경우, SSM Agent는 곧 연결이 ec2messages:* 엔드포인트로 전환됩니다. 2024년 이후에 출시된 리전의 경우 ssmmessages:* 엔드포인트만 지원됩니다.

ssmmessages 및 ec2messages:* 엔드포인트에 대한 자세한 내용은 AWS 서비스 권한 부여 참조에서 다음 주제를 참조하세요.

- Amazon Message Gateway Service에 사용되는 작업, 리소스 및 조건 키(ssmmessages).
- Amazon Message Delivery Service에 사용되는 작업, 리소스 및 조건 키(ec2messages:*)

ssm:* 네임스페이스 인스턴스 관련 API 작업

`DescribeDocumentParameters`

Systems Manager는 이 API 작업을 실행하여 Amazon EC2 콘솔에서 특정 노드를 렌더링합니다. `DescribeDocumentParameters` 작업의 결과가 문서 노드에 표시됩니다.

`DescribeInstanceProperties`

Systems Manager는 이 API 작업을 실행하여 Amazon EC2 콘솔에서 특정 노드를 렌더링합니다. `DescribeInstanceProperties` 작업의 결과가 Fleet Manager 노드에 표시됩니다.

`GetCalendar`

Systems Manager는 이 API 작업을 실행하여 Change Calendar 콘솔에서 Change Calendar 유형 문서를 렌더링합니다.

`GetManifest`

SSM Agent에서는 이 API 작업을 실행하여 AWS Systems Manager Distributor 패키지의 지정된 버전 설치 또는 업데이트에 대한 시스템 요구 사항을 결정합니다. 이는 레거시 API 작업이며 2017년 이후에 시작된 AWS 리전에서는 사용할 수 없습니다.

`ListInstanceAssociations`

SSM Agent에서는 이 API를 실행하여 새 State Manager 연결이 사용 가능한지 확인합니다. 이 API 작업은 State Manager 작동에 필요합니다.

`PutCalendar`

Systems Manager는 이 API 작업을 실행하여 Change Calendar 콘솔에서 Change Calendar 유형 문서를 업데이트합니다.

`PutConfigurePackageResult`

SSM Agent에서는 이 API 작업을 실행하여 퍼블릭 배포자 패키지의 설치 오류 및 지연 지표를 패키지 소유자의 계정에 게시합니다.

`RegisterManagedInstance`

SSM Agent에서는 다음 시나리오에서 이 API 작업을 실행합니다.

- Systems Manager에서 활성화 코드 및 ID를 사용하여 온프레미스 서버 또는 가상 머신(VM)을 관리형 인스턴스로 등록하는 경우.
- AWS IoT Greengrass Version 2 자격 증명을 등록하는 경우.

이 작업은 SSM Agent 버전 3.1.x 이상을 실행하는 Amazon EC2 인스턴스에서도 호출됩니다.

`RequestManagedInstanceRoleToken`

SSM Agent에서는 이 API 작업을 실행하여 관리 노드에 액세스하기 위한 임시 자격 증명을 검색합니다.

`UpdateInstanceAssociationStatus`

SSM Agent에서는 이 API 작업을 실행하여 연결을 업데이트합니다. 이 API 작업은 AWS Systems Manager의 기능인 State Manager가 작동하는 데 필요합니다.

`UpdateInstanceInformation`

SSM Agent에서는 5분마다 클라우드의 Systems Manager 서비스를 호출하여 하트비트 정보를 제공합니다. 이 호출은 에이전트가 예상한 대로 작동하고 있음을 해당 서비스에서 알도록 에이전트와 함께 하트비트를 유지 관리하는 데 필요합니다.

`UpdateManagedInstancePublicKey`

SSM Agent에서는 이 API 작업을 실행하여 관리형 노드에서 키 쌍을 교체한 후 퍼블릭 키를 제공합니다. 퍼블릭 키는 프라이빗 키로 서명된 요청을 인증하여 Systems Manager에서 임시 자격 증명을 가져오는 데 사용됩니다.

참조: Systems Manager용 형식이 지정된 날짜 및 시간 문자열 생성

AWS Systems Manager API 작업은 필터를 허용하여 요청에서 반환하는 결과 수를 제한합니다. 이러한 API 작업 중 일부는 특정 날짜 및 시간을 나타내도록 서식이 지정된 문자열이 필요한 필터를 허용합니다. 예를 들어 `DescribeSessions` API 작업은 `InvokedAfter` 및 `InvokedBefore` 키를 `SessionFilter` 객체에 유효한 일부 값으로 허용합니다. 또 다른 예는 `StartTimeBefore` 및 `StartTimeAfter` 키를 `AutomationExecutionFilter` 객체에 유효한 일부 값으로 허용하는 `DescribeAutomationExecutions` API 작업입니다. 요청을 필터링할 때 이러한 키에 제공하는 값은 ISO 8601 표준과 일치해야 합니다. ISO 8601에 대한 자세한 내용은 ISO 8601을 참조하십시오.

서식이 지정된 이러한 날짜 및 시간 문자열은 필터에만 제한되지 않습니다. 요청 파라미터에 값을 제공할 때 특정 날짜 및 시간을 나타내는 ISO 8601 형식의 문자열이 필요한 API 작업도 있습니다.

GetCalendarState 작업의 AtTime 요청 파라미터가 그 예입니다. 이러한 문자열은 만들기 어렵습니다. 이 주제의 예를 사용하여 Systems Manager API 작업에 사용할 형식이 지정된 날짜 및 시간 문자열을 생성합니다.

Systems Manager의 날짜 및 시간 문자열 형식 지정

다음은 ISO 8601 형식의 날짜 및 시간 문자열의 예입니다.

```
2020-05-08T15:16:43Z
```

2020년 5월 8일 15시 16분(협정 세계 표준시(UTC))을 나타냅니다. 문자열의 날짜 부분은 네 자리 연도, 두 자리 월 및 하이픈으로 구분된 두 자리 날짜로 표시됩니다. 다음과 같은 형식으로 표시될 수 있습니다.

```
YYYY-MM-DD
```

문자열의 시간 부분은 구분 기호인 문자 “T”로 시작한 다음, 콜론으로 구분된 두 자리 시간, 두 자리 분 및 두 자리 초로 표시됩니다. 다음과 같은 형식으로 표시될 수 있습니다.

```
hh:mm:ss
```

문자열의 시간 부분은 UTC 표준을 나타내는 문자 “Z”로 끝납니다.

Systems Manager에 대한 사용자 정의 날짜 및 시간 문자열 생성

원하는 명령줄 도구를 사용하여 로컬 컴퓨터에서 사용자 지정 날짜 및 시간 문자열을 만들 수 있습니다. ISO 8601 형식의 날짜 및 시간 문자열을 만들 때 사용하는 구문은 로컬 컴퓨터의 운영 체제에 따라 다릅니다. 다음은 Windows의 PowerShell 또는 Linux의 GNU coreutils의 date를 사용하여 ISO 8601 형식의 날짜 및 시간 문자열을 만드는 방법에 대한 예입니다.

coreutils

```
date '+%Y-%m-%dT%H:%M:%SZ'
```

PowerShell

```
(Get-Date).ToString("yyyy-MM-ddTH:mm:ssZ")
```

Systems Manager API 작업을 실행할 때 보고 또는 문제 해결할 수 있도록 기록 날짜 및 시간 문자열을 만들어야 할 수 있습니다. 다음은 AWS Tools for PowerShell 및 AWS Command Line Interface(AWS CLI)에 대해 사용자 정의 기록 ISO 8601 형식의 날짜 및 시간 문자열을 생성하고 사용하는 방법의 예입니다.

AWS CLI

- SSM 문서에 대한 마지막 주의 명령 기록을 검색합니다.

```
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
```

- 자동화 실행 기록의 마지막 주를 검색합니다.

```
lastWeekStamp=$(date '+%Y-%m-%dT%H:%M:%SZ' -d '7 days ago')

aws ssm describe-automation-executions \
    --filters Key=StartTimeAfter,Values=$lastWeekStamp
```

- 세션 기록의 마지막 달을 검색합니다.

```
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

- SSM 문서에 대한 마지막 주의 명령 기록을 검색합니다.

```
$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
```

- 자동화 실행 기록의 마지막 주를 검색합니다.

```
$lastWeekStamp = (Get-Date).AddDays(-7).ToString("yyyy-MM-ddTH:mm:ssZ")

Get-SSMAutomationExecutionList `
    -Filters @{Key="StartTimeAfter";Values=$lastWeekStamp}
```

- 세션 기록의 마지막 달을 검색합니다.

```
$lastWeekStamp = (Get-Date).AddDays(-30).ToString("yyyy-MM-ddTH:mm:ssZ")

Get-SSMSession `
    -State History `
    -Filters @{Key="InvokedAfter";Value=$lastWeekStamp}
```

사용 사례 및 모범 사례

이 주제는 AWS Systems Manager 기능에 대한 일반 사용 사례 및 모범 사례를 다룹니다. 경우에 따라 이 주제에는 관련 블로그 게시물과 기술 문서의 링크도 포함됩니다.

> **Note**
>
> 여기에서 각 섹션의 제목은 기술 문서의 해당 섹션으로 연결되는 활성 링크입니다.

자동화

- 인프라용 셀프 서비스 Automation 런북을 생성합니다.
- AWS Systems Manager의 기능인 Automation을 사용하여 퍼블릭 Systems Manager 문서(SSM 문서)를 사용하거나 고유한 워크플로를 작성하여 AWS Marketplace 또는 사용자 정의 AMIs에서 Amazon Machine Images(AMIs)를 간단하게 생성합니다.
- `AWS-UpdateLinuxAmi` 및 `AWS-UpdateWindowsAmi` Automation 런북을 사용하거나 직접 생성한 사용자 정의 Automation 런북을 사용하여 AMIs를 구축하고 유지 관리합니다.

인벤토리

- AWS Config와 함께 AWS Systems Manager의 기능인 Inventory를 사용하여 시간 경과에 따른 애플리케이션 구성을 감사합니다.

Maintenance Windows

- 운영 체제(OS) 패치, 드라이버 업데이트 또는 소프트웨어 설치 등 노드에 방해가 될 가능성이 있는 작업의 일정을 정의합니다.
- AWS Systems Manager의 기능인 State Manager와 Maintenance Windows의 차이점에 대한 자세한 내용은 State Manager와 Maintenance Windows 중에서 선택 섹션을 참조하세요.

Parameter Store

- AWS Systems Manager의 기능인 Parameter Store를 사용하여 글로벌 구성 설정을 중앙에서 관리합니다.
- AWS Systems ManagerParameter Store의 AWS KMS 활용 방식

- Parameter Store 파라미터에서 AWS Secrets Manager 암호 참조.

Patch Manager

- AWS Systems Manager의 기능인 Patch Manager를 사용하여 대규모로 패치를 롤아웃하고 여러 노드에 걸쳐 플릿 규정 준수 가시성을 높입니다.
- Patch Manager를 AWS Security Hub와 통합하여 플릿의 노드가 규정을 준수하지 않을 때 알림을 받고 보안 관점에서 플릿의 패치 상태를 모니터링합니다. Security Hub 사용 시 요금이 부과됩니다. 자세한 내용은 요금을 참조하세요.
- 관리형 노드의 패치 규정 준수를 검사할 때는 한 번에 한 가지 방법만 사용하여 규정 준수 데이터를 실수로 덮어쓰지 않도록 하세요.

Run Command

- EC2 Run Command를 이용해 SSH 액세스 없이 대규모 인스턴스를 관리합니다.
- AWS CloudTrail을 사용하여 AWS Systems Manager의 기능인 Run Command에 의해 또는 이를 대신하여 실행된 모든 API 호출을 감사합니다.
- Run Command를 사용해 명령을 보낼 때 암호, 구성 데이터 또는 기타 보안 암호와 같이 민감한 정보는 일반 텍스트 형식으로 포함하지 않습니다. 계정의 모든 Systems Manager API 활동은 AWS CloudTrail 로그를 위해 S3 버킷에 기록됩니다. 즉, 해당 S3 버킷에 대한 액세스 권한이 있는 사용자는 그러한 보안 암호의 일반 텍스트 값을 볼 수 있습니다. 따라서 `SecureString` 파라미터를 생성하고 사용하여 Systems Manager 작업에 사용하는 민감한 데이터를 암호화하는 것이 좋습니다.

  자세한 내용은 IAM 정책을 사용하여 Systems Manager 파라미터에 대한 액세스 제한 단원을 참조하십시오.

  > **Note**
  >
  > 기본적으로 CloudTrail이 버킷에 제공하는 로그 파일은 Amazon S3가 관리하는 암호화 키(SSE-S3)를 사용하는 서버 측 암호화를 사용하여 암호화됩니다. 직접 관리할 수 있는 보안 계층을 제공하려면 CloudTrail 로그 파일에 대한 AWS KMS 관리형 키(SSE-KMS)를 사용하는 서버 측 암호화를 대신 사용하면 됩니다.
  > 자세한 내용은 AWS CloudTrail 사용 설명서의 AWS KMS–관리형 키(SSE-KMS)로 CloudTrail 로그 파일 암호화를 참조하세요.

- Run Command의 대상 및 속도 제어 기능을 사용하여 단계별 명령 작업을 수행합니다.

- AWS Identity and Access Management(IAM) 정책을 사용하여 Run Command 및 모든 Systems Manager 기능에 대해 세분화된 액세스 권한을 사용합니다.

Session Manager

- AWS CloudTrail을 사용하여 AWS 계정의 세션 활동을 감사합니다.
- Amazon CloudWatch Logs 또는 Amazon S3를 사용하여 AWS 계정의 세션 데이터를 로그합니다.
- 인스턴스에 대한 사용자 세션 액세스를 제어합니다.
- 세션의 명령에 대한 액세스를 제한합니다.
- ssm-user 계정 관리 권한을 설정하거나 해제합니다.

State Manager

- 사전 구성된 `AWS-UpdateSSMAgent` 문서를 사용하여 적어도 한 달에 한 번 SSM Agent를 업데이트합니다.
- (Windows) PowerShell 또는 DSC 모듈을 Amazon Simple Storage Service(Amazon S3)에 업로드하고 `AWS-InstallPowerShellModule`을 사용합니다.
- 태그를 이용해 노드에 대한 애플리케이션 그룹을 생성합니다. 그런 다음 개별 노드 ID를 지정하는 대신 `Targets` 파라미터를 사용하여 노드를 대상으로 합니다.
- Systems Manager를 사용하여 Amazon Inspector에서 생성된 결과를 자동으로 수정합니다.
- SSM 문서에 중앙 집중식 구성 리포지토리를 사용하고 조직 전체에 걸쳐 문서를 공유합니다.
- State Manager와 Maintenance Windows의 차이에 대한 자세한 내용은 State Manager와 Maintenance Windows 중에서 선택 섹션을 참조하세요.

관리형 노드

- Systems Manager는 작업을 수행하기 위해 정확한 시간 참조가 필요합니다. 노드의 날짜와 시간이 잘못 설정되어 있으면 API 요청의 서명 날짜와 일치하지 않을 수 있습니다. 이로 인해 오류 또는 불완전한 기능이 발생할 수 있습니다. 예를 들어 시간 설정이 올바르지 않은 노드는 관리형 노드 목록에 포함되지 않게 됩니다.

  노드에 대한 자세한 정보는 Amazon EC2 인스턴스의 시간 설정을 참조하세요.
- Linux 관리 노드에서는 서명을 확인합니다 SSM Agent.

추가 정보

- Systems Manager의 보안 모범 사례

Systems Manager 리소스 및 아티팩트 삭제

더 이상 해당 리소스에 대한 데이터를 보거나 아티팩트를 사용할 필요가 없는 경우 Systems Manager 리소스 및 아티팩트를 삭제하는 것이 좋습니다. 다음 표에는 각 Systems Manager 기능 또는 아티팩트와 Systems Manager에서 만든 리소스 또는 아티팩트 삭제에 대한 자세한 정보가 있는 링크가 나열됩니다.

기능 또는 아티팩트	Details
Application Manager	Application Manager에서 애플리케이션을 삭제할 수 없지만, 기본 태그, Resource Groups 또는 AWS CloudFormation 스택을 삭제하여 서비스에서 애플리케이션을 제거할 수 있습니다.
자동화	Systems Manager 자동화를 사용하여 AWS 리소스를 만드는 경우, 해당 AWS Management Console을 사용하여 이러한 리소스를 수동으로 삭제해야 합니다. 사용자 지정 실행서를 만든 경우 기본 SSM 문서를 삭제할 수 있습니다. 자세한 내용은 사용자 지정 SSM 문서 삭제 중 단원을 참조하십시오.
Change Calendar	change calendar 및 change calendar 이벤트를 삭제할 수 있습니다. 자세한 내용은 Change Calendar 삭제 및 Change Calendar 이벤트 삭제 섹션을 참조하세요.
Change Manager	변경 템플릿을 삭제할 수 있습니다. 자세한 내용은 변경 템플릿 삭제 단원을 참조하십시오.
규정 준수	Systems Manager Compliance는 Patch Manager 패치 및 State Manager 연결에 대한 규정 준수 데이터를 자동으로 표시합니다. 이 데이터는 삭제할 수 없습니다. S3 버킷에서 규정 준

기능 또는 아티팩트	Details
	수 데이터를 중앙 집중화하도록 리소스 데이터 동기화를 구성한 경우, 동기화를 삭제할 수 있습니다. 자세한 내용은 Compliance의 리소스 데이터 동기화 삭제 단원을 참조하십시오.
Distributor	Distributor에서 패키지를 삭제할 수 있습니다. 자세한 내용은 패키지 삭제 단원을 참조하십시오.
Explorer	Explorer가 OpsData를 수집하는 소스를 연결 중지시킬 수 있습니다. 자세한 내용은 Systems Manager Explorer 데이터 원본 편집 단원을 참조하십시오. Explorer에서 여러 AWS 리전 및 계정의 OpsData 및 OpsItems을(를) 집계하는 데 사용하는 리소스 데이터 동기화를 Amazon Simple Storage Service(Amazon S3) 버킷 하나로 삭제할 수도 있습니다. 자세한 내용은 Systems Manager Explorer 리소스 데이터 동기화 삭제 단원을 참조하십시오. S3 버킷 삭제에 대한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 버킷 삭제를 참조하세요.
Fleet Manager	Fleet Manager를 사용하여 관리형 노드를 삭제할 수 없습니다. Amazon Elastic Compute Cloud(Amazon EC2)를 사용해야 합니다. 자세한 내용은 인스턴스 종료(Linux) 및 인스턴스 종료(Windows)를 참조하세요.

기능 또는 아티팩트	Details
인벤토리	메타데이터를 수집할 일정 및 리소스를 정의하는 State Manager 연결을 삭제하여 Inventory 데이터 수집을 중지할 수 있습니다. 자세한 내용은 데이터 수집 중지 및 인벤토리 데이터 삭제 단원을 참조하십시오. 더 이상 AWS Systems Manager Inventory를 사용하여 AWS 리소스에 대한 메타데이터를 보고 싶지 않다면, 인벤토리 데이터 수집에 사용되는 리소스 데이터 동기화를 삭제하는 것이 좋습니다. 자세한 내용은 Inventory 리소스 데이터 동기화 삭제 단원을 참조하십시오.
Maintenance Windows	유지 관리 기간, 유지 관리 기간 대상, 유지 관리 기간 작업을 삭제할 수 있습니다. 자세한 내용은 유지 관리 기간 리소스 업데이트 또는 삭제(콘솔) 단원을 참조하십시오.
OpsCenter	AWS Command Line Interface 또는 AWS SDK를 사용하여 API 작업 삭제OpsItem를 호출하면 개인 OpsItem을(를) 삭제할 수 있습니다. AWS Management Console에서는 OpsItem을(를) 삭제할 수 없습니다. 자세한 내용은 OpsItems 삭제 단원을 참조하십시오.
Parameter Store	생성한 파라미터를 삭제할 수 있습니다. 자세한 내용은 Systems Manager 파라미터 삭제 단원을 참조하십시오.
Patch Manager	사용자 지정 패치 기준을 삭제할 수 있습니다. 자세한 내용은 사용자 지정 패치 기준선 업데이트 또는 삭제 단원을 참조하십시오.

기능 또는 아티팩트	Details
빠른 설정	빠른 설정으로 생성된 연결을 삭제할 수 있습니다. 연결은 State Manager에서 저장 및 처리됩니다. 자세한 내용은 연결 삭제 단원을 참조하십시오.
Run Command	명령이 처리를 완료하면 해당 명령에 대한 정보가 명령 기록(Command history) 탭에 저장됩니다. 명령 기록(Command history) 탭에서 정보를 삭제할 수 없습니다.
서비스 연결 역할	Systems Manager는 일부 기능에 대한 서비스 연결 역할을 자동으로 생성합니다. 이러한 역할을 삭제할 수 있습니다. 자세한 내용은 Systems Manager에 대한 AWSServiceRoleForAmazonSSM 서비스 연결 역할 삭제 단원을 참조하십시오.
Session Manager	세션을 종료한 후에는 Session Manager가 리소스에 대한 데이터를 보존하지 않습니다. 세션을 종료하려면 세션 종료 섹션을 참조하세요.
SSM Agent	노드에서 SSM Agent를 수동으로 제거할 수 있습니다. 자세한 내용은 다음 항목을 참조하십시오. • Linux: Linux용 EC2 인스턴스에 수동으로 SSM Agent 설치 및 제거 • macOS: SSM Agent용 EC2 인스턴스에 수동으로 macOS 설치 및 제거 • Windows Server: 제어판(Control panel)을 연 다음 프로그램 추가/제거(Add/remove programs)를 선택합니다.
State Manager	연결을 삭제할 수 있습니다. 자세한 내용은 연결 삭제 단원을 참조하십시오.

기능 또는 아티팩트	Details
Systems Manager 문서 서비스	AWS 또는 AWS Support에서 제공한 실행서는 삭제할 수 없지만, 사용자 지정 실행서는 삭제할 수 있습니다. 자세한 내용은 사용자 지정 SSM 문서 삭제 중 단원을 참조하십시오.

State Manager와 Maintenance Windows 중에서 선택

State Manager와 Maintenance Windows 모두 AWS Systems Manager의 기능으로 관리되는 노드에서 몇 가지 유사한 유형의 업데이트를 수행할 수 있습니다. 어느 유형의 업데이트를 선택할지는 시스템 규정 준수를 자동화해야 하는지 아니면 지정한 기간 동안 우선순위가 높고 시간에 민감한 태스크를 수행해야 하는지 여부에 따라 다릅니다.

State Manager 및 Maintenance Windows: 주요 사용 사례

AWS Systems Manager의 기능인 State Manager는 AWS 계정 내의 관리형 노드와 AWS 리소스에 대한 목표 상태 구성을 설정하고 유지 관리합니다. 구성 및 대상의 조합을 연결 객체로 정의할 수 있습니다. State Manager는 계정의 모든 관리형 노드를 일관된 상태로 유지하려는 경우 Amazon EC2 Auto Scaling을 사용하여 새 노드를 생성하려는 경우 또는 계정의 관리형 노드에 대한 엄격한 규정 준수 보고 요구 사항이 있는 경우에 권장되는 기능입니다.

State Manager의 주요 사용 사례는 다음과 같습니다.

- Auto Scaling 시나리오: State Manager는 계정 내에서 시작된 모든 새 노드를 수동으로 또는 Auto Scaling 그룹을 통해 모니터링할 수 있습니다. 새 노드를 대상으로 하는 계정에 연결이 있는 경우(태그 또는 모든 노드를 통해) 해당 특정 연결이 새 노드에 자동으로 적용됩니다.
- 규정 준수 보고: State Manager는 계정의 리소스에 대한 필요한 상태의 준수 보고를 유도할 수 있습니다.
- 모든 노드 지원: State Manager는 지정된 계정 내의 모든 노드를 대상으로 지정할 수 있습니다.

유지 관리 기간은 지정된 기간 내에 AWS 리소스에 대해 하나 이상의 작업을 수행합니다. 시작 및 종료 시간으로 단일 유지 관리 기간을 정의할 수 있습니다. 이 유지 관리 기간 내에서 실행할 여러 태스크를 지정할 수 있습니다. 우선순위가 높은 작업에 관리형 노드 패치, 업데이트 기간 동안 노드에서 여러 유형의 태스크 실행 또는 노드에서 업데이트 작업을 실행할 수 있는 시점 제어가 포함되는 경우 AWS Systems Manager의 기능인 Maintenance Windows를 사용합니다.

Maintenance Windows의 주요 사용 사례는 다음과 같습니다.

- 여러 문서 실행: 유지 관리 기간은 여러 태스크를 실행할 수 있습니다. 태스크마다 다른 문서 유형을 사용할 수 있습니다. 따라서 단일 유지 관리 기간 내에서 여러 태스크를 사용하여 복잡한 워크플로를 구축할 수 있습니다.
- 패치 적용: 유지 관리 기간에서는 특정 태그 또는 리소스 그룹으로 태그가 지정된 단일 리전의 모든 관리형 노드에 대한 패치 적용 지원이 제공될 수 있습니다. 패치에는 일반적으로 노드 중단(예: 로드 밸런서에서 노드 제거), 패치 및 사후 처리(노드를 프로덕션으로 다시 배치)가 포함되기 때문에 지정된 패치 기간 내에서 일련의 태스크로 패치를 수행할 수 있습니다.

  > **Note**
  >
  > 유지 관리 기간을 사용하면 패치 적용 작업이 단일 계정의 단일 리전으로 제한됩니다. Systems Manager의 기능인 Quick Setup에서 생성된 패치 정책을 사용하여, AWS Organizations에서 생성된 조직의 일부 또는 모든 계정 및 리전에 대한 패치 적용을 구성할 수도 있습니다. 자세한 내용은 Quick Setup 패치 정책 사용 단원을 참조하십시오.

- 기간 작업: 유지 관리 기간은 특정 기간 내에 시작되는 하나 이상의 작업 집합을 만들 수 있습니다. 해당 기간 외에는 유지 관리 기간이 시작되지 않습니다. 이미 시작된 작업은 해당 기간 외에도 완료될 때까지 계속됩니다.

다음 표에서는 State Manager와 Maintenance Windows의 주요 기능을 비교합니다.

기능	State Manager	Maintenance Windows
AWS CloudFormation 통합	AWS CloudFormation 템플릿은 State Manager 연결을 지원합니다.	AWS CloudFormation 템플릿은 유지 관리 기간, 기간 대상 및 기간 태스크를 지원합니다.
규정 준수	모든 State Manager 연결은 대상 리소스의 필요한 상태에 대한 규정 준수를 보고합니다. Compliance 대시보드를 사용하여 보고된 규정 준수를 집계하고 볼 수 있습니다.	해당 사항 없음.
구성 관리 통합	State Manager는 Microsoft PowerShell Desired State	해당 사항 없음.

기능	State Manager	Maintenance Windows
	Configuration(DSC), Ansible 플레이북, Chef 레시피와 같은 외부의 목표 상태 솔루션을 지원합니다. State Manager 연결을 사용하여 구성 관리 솔루션이 작동하는지 테스트하고 준비가 되면 해당 구성 변경 사항을 노드에 적용할 수 있습니다.	
문서	State Manager 구성은 Amazon Simple Storage Service(Amazon S3) 버킷과 같은 AWS 리소스에 대한 Policy 문서(인벤토리 정보 수집용), Automation 실행서 또는 관리형 노드용 Systems Manager Command 문서(SSM 문서)로 정의할 수 있습니다.	Maintenance Windows 구성은 자동화 문서(선택적 승인 워크플로가 있는 다단계 작업) 또는 SSM 문서(관리형 노드의 필요한 상태)로 정의할 수 있습니다.
모니터링(Monitoring)	State Manager는 노드의 구성, 연결 또는 상태의 변경 사항(예: 새 노드가 온라인 상태가 됨)을 모니터링합니다. State Manager가 이러한 변경 사항을 감지하면 해당 연결로 원래 대상이 지정된 노드에 지정된 연결이 다시 적용됩니다.	해당 사항 없음.

기능	State Manager	Maintenance Windows
태스크 내 우선순위	해당 사항 없음.	유지 관리 기간 내의 태스크에 우선순위를 할당할 수 있습니다. 우선순위가 같은 태스크는 모두 병렬로 실행됩니다. 우선순위가 낮은 태스크는 우선순위가 높은 태스크가 최종 상태에 도달한 후에 실행됩니다. 조건부로 태스크를 실행할 수 있는 방법은 없습니다. 우선순위가 높은 태스크가 최종 상태에 도달하면 이전 태스크의 상태에 관계없이 다음 우선순위 태스크가 실행됩니다.
안전 제어	State Manager는 대규모 플릿 전체에 구성을 배포할 때 두 가지 안전 제어를 지원합니다. 최대 동시성을 사용하여 구성을 적용해야 하는 동시 노드 또는 리소스 수를 정의할 수 있습니다. 플릿 전체에서 특정 수 또는 비율의 오류가 발생하는 경우 State Manager 연결을 일시 중지하는 데 사용할 수 있는 최대 오류율을 정의할 수 있습니다.	유지 관리 기간은 대규모 플릿 전체에 구성을 배포할 때 두 가지 안전 제어를 지원합니다. 최대 동시성을 사용하여 구성을 적용해야 하는 동시 노드 또는 리소스 수를 정의할 수 있습니다. 플릿 전체에서 특정 수 또는 비율의 오류가 발생하는 경우 유지 관리 기간에서 작업을 일시 중지하는 데 사용할 수 있는 최대 오류율을 정의할 수 있습니다.

기능	State Manager	Maintenance Windows
일정 예약	온디맨드로, 특정 cron 간격으로, 지정된 속도로 또는 생성된 후 한 번 State Manager 연결을 실행할 수 있습니다. 이는 일관되고 시기 적절한 방식으로 필요한 리소스 상태를 유지하려는 경우에 유용합니다. **Important** State Manager 연결에 대한 Cron 표현식은 월 필드를 지원하지 않습니다(예: 3월 한 달 동안 `03` 또는`MAR`) 월별 또는 분기별 구성 업데이트가 필요한 경우 유지 관리 기간이 요구 사항을 가장 잘 충족할 수 있습니다. 자세한 내용은 참조: Systems Manager용 Cron 및 Rate 표현식 단원을 참조하십시오.	유지 관리 기간은 `at` 표현식(예: `"at(2021-07-07T13:15:30)"`), cron 및 rate 표현식, 오프셋이 있는 cron, 유지 관리 기간이 실행되어야 하는 시작 및 종료 시간, 지정된 기간 내에 예약을 중지할 시점을 지정하는 마감 시간을 비롯한 여러 예약 옵션을 지원합니다.
대상 지정	State Manager 연결은 노드 ID, 태그 또는 리소스 그룹을 사용하여 하나 이상의 노드를 대상으로 할 수 있습니다. State Manager는 지정된 계정 내의 모든 관리형 노드를 대상으로 할 수 있습니다.	유지 관리 기간은 노드 ID, 태그 또는 리소스 그룹을 사용하여 하나 이상의 노드를 대상으로 할 수 있습니다.

기능	State Manager	Maintenance Windows
유지 관리 기간 내의 태스크	해당 사항 없음.	유지 관리 기간은 각 태스크가 특정 Automation 실행서 또는 Command 문서 작업을 대상으로 하는 하나 이상의 태스크를 지원할 수 있습니다. 태스크마다 다른 우선순위가 설정되지 않는 한 유지 관리 기간 내의 모든 태스크가 병렬로 실행됩니다. 전반적으로 유지 관리 기간은 다음과 같은 4가지 유형의 태스크를 지원합니다. • AWS Systems Manager Run Command 명령 • AWS Systems Manager Automation 워크플로 • AWS Lambda 함수 • AWS Step Functions 작업

관련 정보

다음의 관련 리소스는 이 서비스를 이용할 때 도움이 될 수 있습니다.

요금

일부 Systems Manager 기능에는 요금이 부과됩니다. 자세한 내용은 AWS Systems Manager 요금을 참조하십시오.

AWS Systems Manager 설명서 라이브러리

AWS Systems Manager 설명서 – SAP의 AWS AppConfig, 인시던트 관리자 및 AWS Systems Manager를 포함하여 Systems Manager의 모든 사용자 설명서를 이용하세요.

AWS re:Post

AWS re:Post – 크라우드 소싱을 거쳐 전문가가 검토한 기술적 질문에 대한 답변을 제공하는 AWS 관리형 Q&A(질의응답) 서비스입니다.

AWS 블로그 및 팟캐스트

AWS 관리 도구 범주의 Systems Manager에 대한 블로그 게시물과 `#Systems Manager`로 태그가 지정된 다른 게시물을 읽어보세요.

Service quotas

Amazon Web Services 일반 참조의 Systems Manager 서비스 할당량을 검토하세요. 별도로 명시되지 않는 한, AWS 계정의 단일 리전에 각 할당량이 적용됩니다.

Systems Manager용 Service Authorization Reference

AWS Service Authorization Reference에서 Systems Manager의 AWS Identity and Access Management(IAM) 정책에서 사용할 수 있는 작업, 리소스 및 조건 컨텍스트 키에 대한 내용을 확인하세요.

AWS Systems Manager 서비스 수준에 관한 계약

AWS Systems Manager 서비스 수준에 관한 계약(SLA)은 Systems Manager 사용을 관리하는 정책이며 Systems Manager를 사용하는 각 AWS 계정에 별도로 적용됩니다.

일반 AWS 리소스

AWS로 작업할 때 다음과 같은 일반 리소스가 도움이 될 수 있습니다.

- Classes & Workshops(교육 및 워크숍) – 역할 기반의 과정 및 전문 과정은 물론 자습형 실습에 대한 링크를 통해 AWS 기술을 연마하고 실무에 도움이 되는 경험을 쌓을 수 있습니다.
- AWS 개발자 센터 – 튜토리얼을 살펴보고, 도구를 다운로드하고, AWS 개발자 이벤트에 대해 알아봅니다.
- AWSDeveloper Tools(개발자 도구) – AWS 애플리케이션을 개발 및 관리하기 위한 개발자 도구, SDK, IDE 도구 키트 및 명령줄 도구 링크입니다.
- 시작하기 리소스 센터 - AWS 계정을(를) 설정하고 AWS 커뮤니티에 가입하고 첫 번째 애플리케이션을 시작하는 방법을 알아봅니다.
- 실습 튜토리얼 – 단계별 튜토리얼에 따라 AWS에서 첫 번째 애플리케이션을 시작합니다.
- AWSWhitepapers(백서) – AWS 솔루션 아키텍트 또는 기타 기술 전문가가 아키텍처, 보안 및 경제 등의 토픽에 대해 작성한 포괄적 AWS 기술 백서 목록의 링크입니다.
- AWS SupportCenter(센터) – AWS Support 사례를 생성하고 관리할 수 있는 허브입니다. 또한 포럼, 기술 FAQ, 서비스 상태 및 AWS Trusted Advisor 등의 기타 유용한 자료에 대한 링크가 있습니다.
- AWS Support – 클라우드에서 1대 1로 애플리케이션을 구축 및 실행하도록 지원하는 빠른 응답 지원 채널인 AWS Support에 대한 정보가 포함된 기본 웹 페이지입니다.
- Contact Us(문의처) - AWS 결제, 계정, 이벤트, 침해 및 기타 문제에 대해 문의할 수 있는 중앙 연락 창구입니다.
- AWSSite Terms(사이트 약관) – 저작권 및 상표, 사용자 계정, 라이선스 및 사이트 액세스와 기타 토픽에 대한 세부 정보입니다.

문서 이력

다음 표에서는 AWS Systems Manager의 최신 릴리스가 발표된 이후 이 설명서에서 변경된 중요 사항에 대해 설명합니다. 이 설명서에 대한 업데이트 알림을 받으려면 RSS feed를 구독하세요.

- API 버전: 2014-11-06

변경 사항	설명	날짜
업데이트: `/aws/service/global-infrastructure` 파라미터 경로의 리전별 가용성	다른 상업 AWS 리전에서 작업하는 경우 어느 상업 리전에서 `/aws/service/global-infrastructure` 퍼블릭 파라미터 경로를 쿼리할 수 있는지 명확히 했습니다. 자세한 내용은 AWS 서비스, 리전, 엔드포인트, 가용 영역, 로컬 영역 및 Wavelength Zone에 대한 퍼블릭 파라미터 호출을 참조하세요.	2024년 6월 12일
신규: 코드 예제 장	새 AWS SDK를 사용한 Systems Manager의 코드 예제 장에서는 Systems Manager 서비스를 사용하는 방법에 대한 예제를 다양한 SDK 언어로 제공합니다.	2024년 5월 8일
`ec2messages:*` 엔드포인트 지원 변경	2024년 이후 출시된 AWS 리전의 경우 `ec2messages:*` 엔드포인트는 SSM Agent에서 상태 및 실행 정보를 Systems Manager 서비스로 다시 전송하는 기능을 지원하지 않습니다. 이 리전의 계정은 `ssmmessag`	2024년 5월 3일

	es:* 를 사용해야 합니다. 2024년 이전에 출시된 리전의 경우 ssmmessages:* 및 ec2messages:* 모두 여전히 지원되지만, ssmmessages:* 엔드포인트(Amazon Message Gateway Service)만 사용하는 것이 좋습니다. 현재 정책에서 ec2messages:* 권한을 안전하게 제거할 수 있습니다. 자세한 내용은 SSM Agent 작업 및 에이전트 관련 API 작업(ssmmessages 및 ec2messages 엔드포인트)을 참조하세요.	
Automation 런북에서 스크립트를 실행하는 데 사용할 수 있는 추가 런타임	이제 aws:executeScript 작업은 Python 3.9, 3.10, 3.11 런타임을 지원합니다. 이 작업을 사용하는 방법에 대한 자세한 내용은 aws:executeScript 섹션을 참조하세요.	2024년 4월 23일
8.8 및 8.9 버전에 대한 지원: AlmaLinux, Oracle Linux, Rocky Linux	이제 Systems Manager는 이제 8.x 이전 버전 외에도 AlmaLinux의 8.8 및 8.9, Oracle Linux, Rocky Linux를 지원합니다. 지원되는 OS 및 버전의 전체 목록은 Systems Manager에서 지원되는 운영 체제를 참조하세요.	2024년 4월 22일

Patch Manager: 패치 작업 상태를 'INSTALLED_PENDING_REBOOT'로 변경	이전에는 Patch Manager에서 설치한 패치만 `INSTALLED_PENDING_REBOOT` 로 표시할 수 있었습니다. Patch Manager 외부에 설치된 패치의 상태는 이 상태로 지정되지 않습니다. 이제 `INSTALLED_PENDING_REBOOT` 는 관리형 노드가 마지막으로 재부팅된 이후 관리형 노드에 적용된 모든 패치에 적용할 수 있습니다. 여기에는 `NoReboot` 옵션을 선택하여 Patch Manager에서 설치한 패치와 노드의 가장 최근 재부팅 이후 Patch Manager 외부에서 설치한 패치가 포함됩니다. 모든 Patch Manager 패치 작업 상태 값에 대한 설명은 패치 규정 준수 상태 값 이해를 참조하세요.	2024년 4월 16일
RHEL 8.9 및 9.3 지원	Patch Manager를 비롯한 Systems Manager는 이제 이전 8.x 및 9.x 버전 외에도 Red Hat Enterprise Linux(RHEL) 버전 8.9 및 9.3을 지원합니다.	2024년 3월 26일

주제 업데이트: AWS Systems Manager용 AWS 관리형 정책	AWS Systems Manager용 AWS 관리형 정책 항목에서는 2021년 3월 12일 이후 도입되거나 업데이트된 Systems Manager의 4가지 관리형 정책에 대한 정보를 제공합니다. 해당 날짜 이전에 생성되었거나 마지막으로 업데이트된 Systems Manager와 함께 사용할 수 있는 12개의 다른 관리형 정책에 대한 정보가 포함된 섹션을 이 항목에 추가했습니다. 자세한 내용은 Systems Manager의 추가 관리형 정책을 참조하세요.	2024년 3월 1일
Parameter Store의 계정 간 공유 지원	이제 리소스 공유를 설정하여 AWS 계정 전체 또는 AWS 조직 내에서 고급 파라미터를 안전하고 효율적으로 공유할 수 있습니다. 리소스 공유를 통해 애플리케이션 구성 관리를 중앙 집중화하고 소유한 모든 계정과 파라미터를 공유하는 데 따른 운영 오버헤드를 줄일 수 있습니다. Parameter Store 콘솔, AWS RAM 콘솔 또는 AWS CLI를 사용하여 계정 간에 파라미터를 공유할 수 있습니다. 자세한 내용은 공유 파라미터 작업을 참조하세요.	2024년 2월 21일

자동화 작업 개선	이제 `aws:approve` 작업과 함께 `onFailure` 및 `isCritical` 속성을 사용할 수 있습니다. `aws:approve` 작업에 대한 자세한 내용은 aws:approve-Pause an automation for manual approval을 참조하세요.	2024년 2월 12일
Patch Manager에 대한 추가 운영 버전 지원	Patch Manager에 대해 지원되는 운영 체제 버전 목록에 항목이 추가되었습니다. 다음에 대한 지원이 추가되었습니다. • Debian Server 11.x 및 12.x • macOS 14.0(Sonoma) • SUSE Linux Enterprise Server(SLES) 15.5 • Ubuntu Server 23.04	2024년 1월 4일
Application Manager 콘솔을 사용하여 자동 SSM Agent 업데이트 구성	이제 Application Manager 콘솔을 사용하여 애플리케이션 인스턴스의 SSM Agent 업데이트를 자동화할 수 있습니다. 자세한 내용은 애플리케이션 인스턴스 작업을 참조하세요.	2023년 12월 21일

하이브리드 및 멀티클라우드 환경에서 Amazon EC2 이외 시스템을 등록하는 프로세스가 업데이트됨	이제 Systems Manager는 하이브리드 및 멀티클라우드 환경에서 Amazon Elastic Compute Cloud(Amazon EC2) 이외 시스템을 등록하는 데 도움이 되는 `ssm-setup-cli` 기능을 제공합니다. 자세한 내용은 하이브리드 Linux 노드에 SSM Agent를 설치하는 방법 및 하이브리드 Windows 노드에SSM Agent를 설치하는 방법을 참조하세요.	2023년 12월 20일
Fleet Manager를 사용하여 Amazon EBS 볼륨 관리	이제 AWS Systems Manager의 기능인 Fleet Manager를 사용하여 관리형 인스턴스에서 Amazon Elastic Block Store 볼륨을 관리할 수 있습니다. 예를 들어 EBS 볼륨을 초기화하고, 파티션을 포맷하며, 볼륨을 마운트해 사용할 수 있습니다. 자세한 내용은 EBS 볼륨 관리를 참조하세요.	2023년 12월 14일
Session Manager 플러그인 향상된 기능	StartSession API 응답을 session-manager-plugin에 환경 변수로 전달하는 지원이 추가되었습니다.	2023년 12월 4일

Automation 런북을 위한 새로운 시각적 디자인 경험	이제 Systems Manager Automation에서 개발한 새로운 시각적 디자인 환경을 사용하여 런북을 만들고 편집할 수 있습니다. 시각적 디자인 환경은 코드가 적은 드래그 앤 드롭 인터페이스를 제공하므로 런북을 더 쉽게 만들고 편집할 수 있습니다. 자세한 내용은 Automation 런북의 시각적 디자인 환경을 참조하십시오.	2023년 11월 26일

새로운 Systems Manager Automation 작업, 데이터 요소 및 런북의 기능 향상	이제 aws:loop 작업을 사용하여 런북의 여러 작업을 반복할 수 있습니다. 이 새로운 작업은 do while 및 for each 스타일 루프를 지원합니다. 또한 새 변수 데이터 요소를 사용하여 런북 컨텍스트 내에서 값을 동적으로 정의, 참조 및 업데이트할 수 있습니다. 런북의 변수 값을 업데이트하려면 새 aws:updateVariable 작업을 사용하십시오. Automation으로 출력의 동적 데이터 유형 변환에 대한 지원도 추가되었습니다. 즉, 출력 값이 지정한 데이터 유형과 일치하지 않으면 Automation이 데이터 유형 변환을 시도합니다. 예를 들어, 반환된 값이 Integer이지만 지정된 Type은 String인 경우 최종 출력 값은 String 값입니다. 마지막으로, Automation은 이제 선택기에 대한 JSONPath 필터 표현식을 지원합니다. 자세한 정보는 다음 주제를 참조하세요. • aws:loop - 자동화의 여러 단계를 반복 • aws:updateVariable - 런북 변수 값을 업데이트 • 데이터 요소 및 파라미터 - 최상위 데이터 요소 • 작업 출력을 입력으로 사용.	2023년 11월 17일

	• 런북에서 JSONPath 사용.	
Remote Desktop Protocol(RDP) 연결에 대한 리전 지원 업데이트	NICE DCV로 구동되는 Fleet Manager 원격 데스크톱에서는 Systems Manager 콘솔에서 직접 Windows Server 인스턴스에 안전하게 연결할 수 있습니다. 다음 세 개의 추가 리전이 Fleet Manager 원격 데스크톱 연결을 사용할 수 있게 되었습니다. • 아프리카(케이프타운)(af-south-1) • 아시아 태평양(자카르타) (ap-southeast-3) • 이스라엘(텔아비브) (il-central-1)	2023년 11월 15일
Patch Manager: RHEL 및 macOS에 대한 확장된 OS 버전 지원	이제 Patch Manager에서 지원하는 추가적인 운영 체제 버전은 다음과 같습니다. • Red Hat Enterprise Linux: 버전 8.8 • macOS: 11.5~11:7(Big Sur) • macOS: 12.0~12.6 (Monterey) • macOS: 13.0~13.5(Ventura)	2023년 10월 23일
새 OpsCenter API - 삭제 OpsItem	이제 OpsCenter에서 개인 OpsItems을(를) 삭제하기 위한 삭제OpsItem API를 제공합니다. 자세한 내용은 AWS Systems Manager API 참조의 DeleteOpsItem을 참조하세요.	2023년 10월 20일

새로운 Quick Setup 구성 유형: 전체 조직에 대한 SSM Agent 업데이트	새로운 구성 유형인 기본 호스트 관리 구성을 사용하면 AWS Organizations에 정의된 대로 조직 관리자가 조직의 계정 및 리전에 있는 모든 EC2 인스턴스에 대한 SSM Agent의 자동 확인 및 업데이트를 신속하게 요청할 수 있습니다. 자세한 내용은 조직의 기본 호스트 관리를 참조하세요.	2023년 10월 16일
CloudWatch Application Insights에서 생성된 OpsItems의 새로운 제목 및 설명 형식	2023년 10월 16일부터 CloudWatch Application Insights에서 생성된 OpsItems의 제목과 설명이 개선된 형식으로 변경됩니다. 새 형식을 보려면 Amazon CloudWatch Application Insights를 참조하세요.	2023년 9월 29일

Fleet Manager RDP 연결의 여러 디스플레이 해상도 지원	이제는 Fleet Manager에서 RDP(원격 데스크톱 프로토콜) 옵션을 사용하여 Windows Server 관리형 노드에 연결하면 디스플레이 해상도를 선택할 수 있습니다. 이전에는 모든 연결에 고정된 720P(1366 x 768) 해상도가 사용되었습니다. 이제는 각 연결에 대해 다음 중에서 선택할 수 있습니다. • 자동으로 조정(감지된 화면 크기에 따라 최적 해상도 결정) • 1920 x 1080 • 1400 x 900 • 1366 x 768 • 800 x 600 자세한 내용은 원격 데스크톱을 사용하여 관리형 노드에 연결을 참조하세요.	2023년 9월 22일
새 주제: 패치 정책 작업의 무작위 패치 기준선 ID	Quick Setup 패치 정책에서 `AWS-RunPatchBaseline` SSM Command 문서의 `BaselineOverride` 파라미터를 사용하여 패치 정책 작업이 실행될 때마다 패치 기준선에 대한 무작위 ID를 생성하는 방식을 설명하는 콘텐츠를 추가했습니다. 자세한 내용은 패치 정책 작업의 무작위 패치 기준선 ID를 참조하세요.	2023년 9월 22일

OpsItems 관리를 위한 새로운 운영 인사이트	OpsCenter에는 이제 대부분의 OpsItem이 생성되는 리소스라는 운영 인사이트가 포함됩니다. AWS 리소스에 10개를 초과하는 미결 OpsItems이 있으면 이 유형의 인사이트가 생성됩니다. 이 인사이트를 사용하여 문제가 있는 리소스를 찾아보세요. 인사이트 내에서 나온 `AWS-BulkResolveOpsItems` 런북을 사용하여 리소스와 관련된 OpsItems 문제를 빠르게 해결하세요. 자세한 내용은 운영 인사이트를 분석하여 OpsItems 줄이기를 참조하세요.	2023년 9월 22일
GPG 퍼블릭 키 업데이트됨	SSM Agent의 서명을 확인하는 새로운 퍼블릭 키가 생성되었습니다. 자세한 내용은 SSM Agent의 서명 확인을 참조하세요.	2023년 9월 5일
AlmaLinux, Oracle Linux, RHEL 및 Rocky Linux의 추가 버전 지원 추가됨	다음과 같은 추가 OS 버전 지원을 반영하도록 AWS Systems Manager 및 Patch Manager가 지원되는 운영 체제 목록이 업데이트되었습니다. • AlmaLinux: 9.2 • Oracle Linux: 8.7 및 9.2 • Red Hat Enterprise Linux(RHEL): 8.7, 9.1 및 9.2 • Rocky Linux: 8.6 및 8.7, 9.0~9.2	2023년 8월 30일

OpsCenter를 통해 OpsItem 설명 필드의 마크다운 서식 지원이 추가되었습니다.	이제는 OpsCenter에서 OpsItem 설명 필드의 마크다운 서식이 지원합니다. 지원되는 마크다운 서식 유형은 다음과 같습니다. • 단락 • 줄 간격 • 가로줄 • 제목 • 텍스트 서식 지정 • Links • List 자세한 내용은 AWS Management Console 시작하기 시작 안내서의 콘솔에서 마크다운 사용을 참조하세요.	2023년 8월 18일
AWS 파라미터 및 Secrets Lambda 확장의 새 버전	이제는 AWS 파라미터 및 Secrets Lambda 확장의 새로운 버전을 사용할 수 있습니다. 아시아 태평양(멜버른)(ap-southeast-4) 및 이스라엘(텔아비브)(il-central-1) 리전(x86_64 및 x86 아키텍처만 해당)에 대한 확장 지원도 추가되었습니다. 자세한 내용은 AWS Lambda 함수에서 Parameter Store 파라미터 사용을 참조하세요.	2023년 8월 16일

업데이트: Quick Setup 패치 정책 버킷에 필요한 권한에 대한 정보 추가됨	패치 정책을 생성할 때 `baseline_overrides.json` 이라는 파일이 포함된 Amazon S3 버킷을 Quick Setup에서 생성합니다. 이 파일에는 패치 정책에 지정한 패치 기준선에 대한 정보가 저장됩니다. 패치 정책을 구성할 때 인스턴스에 인스턴스에 연결된 기존 인스턴스 프로파일에 필수 IAM 정책 추가 확인란을 선택하는 옵션이 있습니다. 이 옵션을 선택하지 않기로 선택하는 경우에는 이 버킷에 액세스하는 권한을 특정 리소스에 제공해야 합니다. 그렇지 않으면 정책 작업이 실패할 수 있습니다. 자세한 정보는 다음 주제를 참조하세요. • 패치 정책 S3 버킷에 대한 권한 • 문제: `baseline_overrides.json` 에 대한 "Invoke-PatchBaselineOperation: 액세스 거부됨" 오류 또는 "S3에서 파일을 다운로드할 수 없음" 오류	2023년 7월 6일

Quick Setup을 사용하여 다중 계정 OpsItem 관리용 OpsCenter 구성	OpsCenter의 Quick Setup은 여러 계정의 OpsItems를 관리하는 다음과 같은 작업을 완료하는 데 도움이 됩니다. • 위임된 관리자 계정 지정 • 필수 AWS Identity and Access Management(IAM) 정책 및 역할 생성 • 위임된 관리자가 여러 계정의 OpsItems를 관리할 수 있는 AWS Organizations 조직 또는 멤버 계정 하위 집합 지정 자세한 내용은 (선택 사항) Quick Setup을 사용하여 여러 계정의 OpsItems을 관리하도록 OpsCenter 구성을 참조하세요.	2023년 6월 19일
Quick Setup을 사용하여 Amazon EC2 시작 에이전트 업데이트	이제는 Systems Manager에서 인스턴스에 설치된 시작 에이전트의 새 버전을 30일마다 확인하도록 허용할 수 있습니다. 새 버전을 사용할 수 있는 경우 Systems Manager에서 인스턴스의 에이전트를 업데이트합니다. 자세한 내용은 Quick Setup 호스트 관리를 참조하세요.	2023년 6월 19일

Patch Manager에서 이제 Ubuntu Server 22.04 LTS 지원	이제는 Patch Manager를 사용하여 Ubuntu Server 22.04 LTS 노드에 패치를 적용할 수 있습니다. 지원되는 다른 Ubuntu Server 버전과 마찬가지로 22.04 LTS 버전에서도 AWS 관리형 `AWS-UbuntuDefaultPatchBaseline` 패치 기준선을 사용합니다.	2023년 5월 15일
Systems Manager에서 이제 Patch Manager를 포함한 AlmaLinux 지원	이제는 Systems Manager를 사용하여 AlmaLinux 8.3~8.7, 9.0~9.1 노드를 관리할 수 있습니다. 패치 적용을 위해 RHEL 8에 적용되는 규칙이 AlmaLinux에도 많이 적용됩니다. AlmaLinux에서는 새 `AWS-DefaultAlmaLinuxPatchBaseline` 을 사용합니다. 자세한 정보는 다음 주제를 참조하세요. • AlmaLinux 인스턴스에 수동으로 SSM Agent 설치 • 보안 패치 선택 방법 • 패치 설치 방법 • AlmaLinux, RHEL 및 Rocky Linux의 패치 기준선 규칙 작동 방식입니다.	2023년 5월 8일
Quick Setup을 사용하여 EC2Launch v2 에이전트 배포	이제는 Quick Setup을 사용하여 EC2Launch v2 에이전트를 배포할 수 있습니다. 자세한 내용을 알아보려면 Quick Setup으로 Distributor 패키지 배포를 참조하세요.	2023년 4월 13일

Systems Manager의 Amazon Linux 2023 지원	이제는 Systems Manager에서 Patch Manager 작업 지원을 포함한 새 Amazon Linux 2023(AL2023) EC2 인스턴스 유형을 지원합니다. Amazon Linux 2에 적용되는 패치 적용 규칙이 Amazon Linux 2023에도 많이 적용됩니다(Patch Manager에서도 Amazon Linux 2022 평가판 릴리스를 계속 지원합니다). 자세한 정보는 다음 주제를 참조하세요. • 보안 패치 선택 방법 • 패치 설치 방법 • Amazon Linux 1, Amazon Linux 2, Amazon Linux 2022, Amazon Linux 2023의 패치 기준 규칙 작동 방식	2023년 3월 23일
Amazon EC2 인스턴스에 대한 설정 콘텐츠 수정됨	Amazon EC2 인스턴스에 대한 설정 콘텐츠를 수정했습니다. 이제 새로 릴리스된 기본 호스트 관리 구성을 인스턴스 권한에 사용하는 것이 좋습니다. 자세한 내용은 Systems Manager에 필요한 인스턴스 권한 구성을 참조하세요.	2023년 2월 15일
기본 호스트 관리 구성을 사용한 자동 인스턴스 관리	이제 Systems Manager를 사용하여 전체 AWS 리전에서 Amazon EC2 인스턴스를 자동으로 관리할 수 있습니다. 자세한 내용은 기본 호스트 관리 구성을 참조하세요.	2023년 2월 15일

즐겨찾기에 SSM 문서 추가	자주 사용하는 SSM 문서를 찾는 데 도움이 되도록 이제 문서를 즐겨찾기에 추가할 수 있습니다. 문서 유형, AWS 계정 및 AWS 리전로 최대 20개의 문서를 즐겨찾기에 추가할 수 있습니다. Systems Manager 문서 콘솔에서 즐겨찾기를 선택, 수정 및 확인할 수 있습니다. 자세한 내용은 즐겨찾기에 문서 추가를 참조하세요.	2023년 2월 7일
Change Calendar를 사용하여 Automation을 위한 변경 제어 구현	Automation과 Change Calendar를 통합하여 이제 AWS 계정의 모든 자동화에 대한 변경 제어를 구현할 수 있습니다. 자세한 내용은 Automation을 위한 변경 제어 구현을 참조하세요.	2023년 1월 24일

새 Change Manager 승인 워크플로	Change Manager 승인 워크플로는 이제 라인별 승인 대신 수준별 승인을 지원합니다. 이전에는 승인 수준에 추가한 모든 승인자가 변경 요청을 승인해야 했습니다. 그렇지 않으면 수준이 승인되지 않았습니다. 이제 수준에 필요한 승인 수를 지정하고 승인자를 그 이상 추가할 수 있습니다. 예를 들어, 한 수준에 대해 3건의 승인을 요구하고 최대 5명의 승인자를 지정할 수 있습니다. 승인자 3명 중 1명의 승인만 있으면 수준을 승인할 수 있습니다. 자세한 내용은 변경 템플릿의 승인 정보를 참조하세요.	2023년 1월 23일

신규: Quick Setup에서 패치 정책을 사용하여 전체 조직에 대한 패치 구성	이제 Systems Manager의 기능인 Quick Setup을 사용하여 Patch Manager 기반 패치 정책을 생성할 수 있습니다. 패치 정책은 관리형 노드에 자동으로 패치를 적용할 때 사용할 일정과 패치 기준선을 정의합니다. 단일 패치 정책 구성을 사용하여 조직 내 모든 리전의 모든 계정이나, 선택한 계정 및 리전이나, 단일 계정-리전 쌍에 대한 패치 적용을 정의할 수 있습니다. 자세한 내용은 다음 항목을 참조하십시오. • Quick Setup 패치 정책 사용 • Quick Setup 패치 정책을 사용하여 조직 전반의 패치 적용 자동화	2022년 12월 22일

Application Manager는 Amazon EC2와 통합되어 애플리케이션 컨텍스트에서 인스턴스에 대한 정보를 표시합니다.	Application Manager는 선택한 애플리케이션의 인스턴스 상태, 상태 및 Amazon EC2 Auto Scaling 상태를 그래픽 형식으로 표시합니다. Instances(인스턴스) 탭에는 애플리케이션의 각 인스턴스에 대한 다음 정보가 포함된 테이블도 있습니다. • 인스턴스 상태(보류 중, 중지 중, 실행 중, 중지됨) • SSM Agent의 Ping 상태 • 인스턴스에서 대해 마지막으로 처리된 Systems Manager Automation 런북의 상태 및 이름 • 상태별 Amazon CloudWatch Logs 경보의 수입니다. • ALARM-지표 또는 표현식이 정의된 임곗값을 벗어났습니다. • OK-지표 또는 표현식이 정의된 임곗값 내에 있습니다. • INSUFFICIENT_DATA - 경보가 방금 시작되었거나 지표를 사용할 수 없거나 지표에서 경보 상태를 결정하는 데 사용할 수 있는 데이터가 충분하지 않습니다. • 상위 및 개별 오토 스케일링 그룹의 Auto Scaling 상태	2022년 12월 22일

Quick Setup을 사용하여 Amazon EC2 인스턴스의 시작 및 중지 예약	이제 Resource Scheduler 솔루션을 배포하여 Quick Setup에서 Amazon EC2 인스턴스의 시작 및 중지를 자동화할 수 있습니다. 자세한 내용은 Resource Scheduler를 참조하세요.	2022년 12월 19일
이제 OpsCenter에서 여러 계정에 걸친 OpsItems 작업 지원	OpsCenter는 관리 계정(AWS Organizations 관리 계정 또는 Systems Manager 위임 관리자 계정) 및 멤버 계정에서 세션을 실행하는 동안 OpsItems 작업을 지원합니다. 구성되면 사용자는 다음 유형의 작업을 수행할 수 있습니다. • 멤버 계정의 OpsItems 생성, 보기 및 업데이트 • 멤버 계정의 OpsItems에 지정된 AWS 리소스에 대한 세부 정보 보기 • Systems Manager Automation 런북을 시작하여 멤버 계정의 AWS 리소스 관련 문제 해결 자세한 내용은 여러 계정의 OpsItems으로 작업할 수 있도록 OpsCenter 설정을 참조하세요.	2022년 11월 16일

AWS CloudTrail Lake를 사용하여 Change Manager 변경 요청의 세부 정보 추적	이제 AWS CloudTrail Lake의 이벤트 데이터 스토어를 사용하여 Change Manager에서 조직 또는 계정에 대해 실행되는 변경 요청에 대한 세부 정보를 캡처하고 검토할 수 있습니다. 이 정보에는 변경 요청을 생성한 사용자 ID, 요청이 실행된 IP 주소, 변경이 수행된 AWS 리전, 대상 리소스 등에 대한 감사 가능한 세부 정보가 포함됩니다. 자세한 내용은 변경 요청 이벤트 모니터링 및 변경 요청 세부 정보, 태스크 및 타임라인 검토를 참조하세요.	2022년 11월 11일
CloudWatch 경보를 사용한 추가 Systems Manager Automation 태스크 제어	이제 CloudWatch 경보를 사용하여 여러 계정 및 리전에 걸쳐 자동화를 실행할 때 추가 제어를 구현할 수 있습니다. 자동화에 지표 또는 복합 CloudWatch 경보를 적용하여 정의한 지표를 기반으로 자동화가 중지되는 시기를 제어할 수 있습니다. 여러 계정 및 리전에서 실행되는 자동화에 CloudWatch 경보를 적용하는 방법에 대한 자세한 내용은 여러 리전 및 계정에서 자동화 실행(콘솔)을 참조하세요.	2022년 11월 9일

업데이트됨: 'AWS Lambda 함수에서 Parameter Store 파라미터 사용'	AWS 파라미터 및 보안 Lambda 익스텐션을 사용하여 파라미터 값을 검색하고 Lambda 함수에서 나중에 사용하기 위해 캐시하는 데 도움이 되는 추가 정보를 제공했습니다. Lambda 익스텐션을 사용하면 Parameter Store에 대한 API 호출 수를 줄여 비용을 절감할 수 있습니다. 자세한 내용은 AWS Lambda함수에서 Parameter Store 파라미터 사용을 참조하세요.	2022년 10월 25일
CloudWatch 경보를 사용한 추가 Systems Manager 태스크 제어	이제 CloudWatch 경보를 사용하여 자동화 및 명령을 실행할 때 추가 제어를 구현할 수 있습니다. CloudWatch 경보는 State Manager 연결 또는 유지 관리 기간 태스크에 등록된 경우 자동화 또는 명령에 추가할 수도 있습니다. 자동화 또는 명령에 복합 CloudWatch 경보를 적용하여 정의한 지표를 기반으로 자동화 또는 명령이 중지되는 시기를 제어할 수 있습니다. CloudWatch 경보를 자동화 또는 명령에 적용하는 방법에 대한 자세한 내용은 다음 절차를 참조하세요. • 보안 패치 선택 방법 • 패치 설치 방법 • Amazon Linux 1, Amazon Linux 2, Amazon Linux 2022의 패치 기준 규칙 작동 방식	2022년 9월 26일

CloudWatch 경보를 사용한 추가 Systems Manager 태스크 제어	이제 CloudWatch 경보를 사용하여 자동화 및 명령을 실행할 때 추가 제어를 구현할 수 있습니다. CloudWatch 경보는 State Manager 연결 또는 유지 관리 기간 태스크에 등록된 경우 자동화 또는 명령에 추가할 수도 있습니다. 자동화 또는 명령에 복합 CloudWatch 경보를 적용하여 정의한 지표를 기반으로 자동화 또는 명령이 중지되는 시기를 제어할 수 있습니다. CloudWatch 경보를 자동화 또는 명령에 적용하는 방법에 대한 자세한 내용은 다음 절차를 참조하세요. • 단순한 자동화 실행 • 콘솔에서 명령 실행 • 연결 생성 • 유지 관리 기간에 태스크 할당	2022년 9월 26일
고급 인스턴스 티어 요구 사항 명확화	고객 피드백을 바탕으로 인스턴스 티어 구성에서 고급 인스턴스 티어를 활성화해야 하는 시나리오를 명확히 했습니다.	2022년 9월 21일
Quick Setup을 사용하여 Amazon CloudWatch 에이전트 배포	이제 Quick Setup을 사용하여 Amazon CloudWatch 에이전트를 배포할 수 있습니다. 자세한 내용을 알아보려면 Quick Setup으로 Distributor 패키지 배포를 참조하세요.	2022년 9월 20일

이제 EC2 인스턴스 메타데이터가 허용되는 경우 패치 그룹에 대해 'PatchGroup' 키가 지원됨	EC2 인스턴스 메타데이터에서 태그를 허용할 때 생성하는 태그 키에 공백이 없어야 합니다. 이전에는 태그 키 `Patch Group`을 인스턴스에 적용해야 했기 때문에 이로 인해 고객이 일부 EC2 인스턴스를 Patch Manager의 패치 그룹에 추가할 수 없었습니다. Patch Manager는 이제 패치 그룹의 인스턴스를 식별하기 위한 태그 키로 `Patch Group`(공백 포함)과 `PatchGroup` (공백 없음)을 모두 지원합니다. 이제 인스턴스 메타데이터에서 태그가 허용되는 EC2 인스턴스를 Patch Manager의 패치 그룹에 추가할 수 있습니다. 자세한 내용은 패치 그룹 정보를 참조하세요.	2022년 8월 31일

새 주제: '패키지 릴리스 날짜 및 업데이트 날짜 계산 방법'	AWS에서 관리하는 패치 기준선에서 새 패치는 릴리스 또는 업데이트되고 7일 후 자동 승인됩니다. 생성한 사용자 지정 패치 기준선에서 설치를 릴리스되거나 업데이트된 후 자동 승인되기까지 대기할 일 수를 지정할 수도 있습니다. Amazon Linux 1 및 Amazon Linux 2의 경우 다양한 요인이 최신 릴리스 날짜 및 업데이트 날짜를 계산하는 방법에 영향을 줍니다. 자동 승인 지연을 선택할 때 예기치 않은 결과를 방지하는 데 도움이 되도록 How package release dates and update dates are calculated(패키지 릴리스 날짜 및 업데이트 날짜 계산 방법) 주제에 이러한 요소가 설명되어 있습니다.	2022년 8월 24일
업데이트된 콘텐츠: AMI 패치 및 Auto Scaling 그룹 업데이트	시작 구성 대신 시작 템플릿을 사용하도록 auto scaling 그룹을 위해 AMIs 업데이트 단계를 업데이트했습니다. 또한 런북 콘텐츠에 최신 자동화 작업 및 런타임을 구현했습니다.	2022년 6월 22일

Change Manager: 사용자가 자동 승인 가능한 요청을 생성하지 못하도록 방지	자동 승인을 지원하도록 Change Manager에서 변경 템플릿을 구성할 수 있습니다. 즉, 필요한 IAM 권한이 있는 사용자가 추가 승인 없이 변경 요청을 시작하도록 선택할 수 있습니다. 이제 변경 템플릿이 지원하는 경우에도 개별 사용자, 그룹 또는 IAM 역할이 자동 승인 요청을 제출하지 못하도록 제한할 수도 있습니다. 이는 새로운 IAM 조건 키인 `ssm:AutoApprove` 사용을 통해 가능합니다. 자세한 내용은 자동 승인 실행서 워크플로에 대한 액세스 제어를 참조하세요.	2022년 6월 15일
유지 관리 기간 작업 역할에 대한 업데이트된 지침	이전에는 Systems Manager 콘솔에서 작업에 대한 유지 관리 역할로 사용하도록 AWS에서 관리하는 IAM 서비스 연결 역할인 `AWSServiceRoleForAmazonSSM` 을 선택할 수 있었습니다. 유지 관리 기간에 이 역할 및 관련 정책인 `AmazonSSMServiceRolePolicy` 를 사용하는 것은 더 이상 권장되지 않습니다. 대신 유지 관리 기간 작업을 위한 사용자 지정 정책 및 역할을 생성해야 합니다. 자세한 내용은 Maintenance Windows 설정을 참조하세요.	2022년 6월 9일

Session Manager에 대한 원격 호스트로의 포트 전달 지원	이제 Session Manager에서 원격 호스트로의 포트 전달 세션을 지원합니다. 원격 호스트는 Systems Manager에서 관리할 필요가 없습니다. 자세한 내용은 세션 시작(원격 호스트로의 포트 전달)을 참조하세요.	2022년 5월 25일
업데이트된 콘텐츠: Amazon EC2 Linux 인스턴스에서의 SSM Agent 수동 설치 관련 지침	고객 피드백에 대한 응답으로 Amazon EC2 인스턴스에서의 SSM Agent 수동 설치에 대한 지침을 제공하는 주제를 점검했습니다. 이제 이 주제에서는 임의의 AWS 리전에 있는 EC2 인스턴스에 빠르게 설치하기 위해 복사 및 붙여넣을 수 있는 전역적으로 사용 가능한 파일을 사용하는 명령을 제공합니다. 이 주제에서는 사용 중인 리전에서 사용할 수 있는 파일을 사용하는 설치 명령을 생성하는 데 도움이 되는 정보도 제공합니다. 후자의 방법은 스크립트나 템플릿을 사용하여 여러 인스턴스에 에이전트를 설치할 때 권장됩니다. 자세한 내용은 Linux용 EC2 인스턴스에서 SSM Agent 수동 설치 섹션의 Linux 운영 체제 관련 지침을 참조하세요.	2022년 5월 9일

새로운 주제: Amazon Machine Images(AMIs), SSM Agent 사전 설치	고객 피드백에 대한 응답으로 SSM Agent가 사전 설치된 AWS에서 관리하는 AMIs에 관한 정보를 모았습니다. 이 주제에서는 이러한 AMIs에서 생성된 Amazon EC2 인스턴스가 성공적으로 설치되었고 실행 중인지 확인하는 방법에 관한 지침도 제공합니다. 에이전트가 성공적으로 설치되지 않았거나 설치되었지만 시작되지 않는 경우 이러한 인스턴스에서 에이전트를 시작하거나 수동으로 설치하는 방법에 대한 정보도 제공합니다. 자세한 내용은 Amazon Machine Images(AMIs), SSM Agent 사전 설치 단원을 참조하세요.	2022년 5월 8일
새로운 State Manager 섹션	State Manager에서 연결을 실행할 때의 세부 정보를 설명하는 새 섹션이 추가되었습니다. 자세한 내용은 연결 일정 예약 정보의 내용을 참조하세요.	2022년 4월 27일

Patch Manager가 이제 Rocky Linux을 지원합니다.	이제 Patch Manager를 사용하여 Rocky Linux 노드를 패치할 수 있습니다. 패치를 위해 RHEL 8에 적용되는 규칙 대부분은 Rocky Linux에도 적용됩니다. Rocky Linux 8은 새 `AWS-DefaultRockyLinuxPatchBaseline` 을 사용합니다. 자세한 정보는 다음 주제를 참조하세요. • 보안 패치 선택 방법 • 패치 설치 방법 • RHEL, CentOS Stream 및 Rocky Linux에서 패치 기준 규칙 작동 방법	2022년 4월 14일
Patch Manager에서 이제 CentOS Stream 8 지원	이제 Patch Manager를 사용하여 CentOS Stream 8 및 Red Hat Enterprise Linux(RHEL) 4.4~4.5 인스턴스를 패치할 수 있습니다. 패치를 위해 RHEL 8에 적용되는 규칙 대부분은 CentOS Stream 8에도 적용됩니다. CentOS Stream 8은 새 `AWS-DefaultCentOSPatchBaseline` 을 사용합니다. 자세한 정보는 다음 주제를 참조하세요. • 보안 패치 선택 방법 • 패치 설치 방법 • RHEL 및 CentOS Stream에서 패치 기준 규칙 작동 방법	2022년 4월 4일

Change Manager용 수임 역할 생성	새 섹션에서는 Change Manager용 수임 역할의 생성 및 구현에 대한 요구 사항을 명확히 설명합니다. 수임 역할은 사용자를 대신하여 승인된 변경 요청에 지정된 실행서 워크플로를 안전하게 실행하도록 Change Manager를 활성화하는 AWS Identity and Access Management(IAM) 서비스 역할입니다. 이 역할은 Change Manager에 AWS Systems Manager(AWS STS) `AssumeRole` 신뢰를 부여합니다. 자세한 내용은 Change Manager에 대한 역할 및 권한 구성을 참조하세요.	2022년 3월 18일
Change Manager 변경 요청 일괄 승인 또는 거부	이제 Systems Manager 콘솔에서 한 번의 작업으로 승인 또는 거부할 여러 변경 요청을 선택할 수 있습니다. 자세한 내용은 변경 요청 검토 및 승인 또는 거부(콘솔)를 참조하세요.	2022년 3월 8일

Rocky Linux 및 Windows Server 2022 관리형 노드 지원	Systems Manager는 온프레미스 또는 다른 클라우드 공급자와 함께 위치한 엣지 디바이스 및 하이브리드 머신을 포함한 Rocky Linux 및 Windows Server 2022 관리형 노드를 지원합니다. 이러한 운영 체제에서 Systems Manager를 사용하려면 하이브리드 환경 또는 엣지 디바이스에 대한 절차를 포함하여 필요한 모든 Systems Manager 설정 절차를 완료해야 합니다(해당하는 경우). 자세한 내용은 Systems Manager 설정을 참조하세요. Rocky Linux 시스템의 경우 SSM Agent도 수동으로 설치해야 합니다. 자세한 내용은 Rocky Linux 인스턴스에 수동으로 SSM Agent 설치를 참조하세요. Windows Server 2022 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스의 경우 SSM Agent가 기본적으로 설치되어 있습니다.	2022년 3월 1일

Automation이 동시성 요구 사항에 따라 조정되고 Automation 사용 지표를 볼 수 있도록 허용	이제 Automation이 동시 자동화 할당량을 자동으로 조정하고 CloudWatch에 게시된 자동화 사용 지표를 보도록 허용할 수 있습니다. 적응형 동시성에 대한 자세한 내용은 Automation이 동시성 요구 사항에 따라 조정되도록 허용을 참조하세요. Automation 사용 지표를 보는 방법에 대한 자세한 내용은 Amazon CloudWatch를 사용하여 Automation 지표 모니터링을 참조하세요.	2022년 1월 27일
Automation이 동시성 요구 사항에 따라 조정되고 Automation 사용 지표를 볼 수 있도록 허용	이제 Automation이 동시 자동화 할당량을 자동으로 조정하고 CloudWatch에 게시된 자동화 사용 지표를 보도록 허용할 수 있습니다. 적응형 동시성에 대한 자세한 내용은 Automation이 동시성 요구 사항에 따라 조정되도록 허용을 참조하세요. Automation 사용 지표를 보는 방법에 대한 자세한 내용은 Amazon CloudWatch를 사용하여 Automation 지표 모니터링을 참조하세요.	2022년 1월 27일
범주별로 정리된 Systems Manager 문서	Amazon 소유 Systems Manager 문서는 이제 필요한 문서를 찾는 데 도움이 되도록 유형 및 범주별로 구성됩니다.	2022년 1월 13일

Automation을 위한 통합 생성 및 호출	이제 통합을 생성하여 자동화 중 웹후크를 사용하여 메시지를 전송할 수 있습니다. 런북에서 새로운 `aws:invokeWebhook` 작업을 사용하여 자동화 중에 통합을 호출할 수 있습니다. 통합 생성에 대한 자세한 내용은 Automation을 위한 웹후크 통합 생성을 참조하세요. `aws:invokeWebhook` 작업에 대해 자세히 알아보려면 `aws:invokeWebhook` - Automation 웹후크 통합 호출을 참조하세요.	2022년 1월 13일
새로운 AWS 리전에서 사용 불가능한 기능	다음 Systems Manager 기능은 현재 새로운 아시아 태평양(자카르타) 리전에서 사용할 수 없습니다. • Application Manager • Change Calendar • Change Manager • Explorer • Fleet Manager • Incident Manager • Quick Setup	2021년 12월 13일

애플리케이션에 대한 리소스 비용 세부 정보 보기	Application Manager는 Cost Explorer 위젯을 통해 AWS Billing and Cost Managemen t와 통합됩니다. Billing and Cost Management 콘솔에서 Cost Explorer를 활성화하면, Application Manager의 Cost Explorer 위젯은 특정 비컨테이너 애플리케이션 또는 애플리케이션 구성 요소에 대한 비용 데이터를 보여줍니다. 위젯의 필터를 사용하여 막대 또는 꺾은선형 차트에서 다양한 기간, 세부 기간 및 비용 유형에 따라 비용 데이터를 볼 수 있습니다. 자세한 내용은 애플리케이션에 대한 개요 보기 섹션을 참조하세요.	2021년 12월 7일
Fleet Manager를 사용해 프로세스 관리	이제 Fleet Manager를 사용하여 노드에서 프로세스를 관리할 수 있습니다. 자세한 내용은 프로세스로 작업을 참조하세요.	2021년 12월 6일
용어 변경: 관리형 인스턴스는 이제 관리형 노드입니다	AWS IoT Greengrass 코어 디바이스 지원에 대해서는, 대부분의 Systems Manager 설명서에서 문구 관리형 인스턴스가 관리형 노드로 바뀌었습니다. Systems Manager 콘솔, API 호출, 오류 메시지 및 SSM 문서는 여전히 인스턴스라는 용어를 사용합니다.	2021년 11월 29일

엣지 디바이스 지원	Systems Manager는 다음과 같은 엣지 디바이스 구성을 지원합니다 • AWS IoT Greengrass: 이제 Systems Manager는 AWS IoT Greengrass용으로 구성된 모든 장치를 지원하며 AWS IoT Greengrass 핵심 소프트웨어를 실행합니다. AWS IoT Greengrass 코어 디바이스를 온보딩하려면 AWS Identity and Access Management(IAM) 서비스 역할을 생성해야 합니다. 또한 AWS IoT Greengrass 콘솔을 사용하여 디바이스의 AWS IoT Greengrass 구성 요소로써 SSM Agent를 배포해야 합니다. 자세한 내용은 AWS Systems Manager 엣지 디바이스의 보안 그룹 섹션을 참조하세요. • 하이브리드 환경의 엣지 디바이스: Systems Manager는 AWS IoT 코어 디바이스 및 비 AWS IoT 디바이스를 이들을 온프레미스 머신으로 구성한 후에 지원합니다. 디바이스를 온보딩하려면 IAM 서비스 역할을 생성하고, 하이브리드 환경에 대한 관리형 노드 활성화를 생성하고, 수동으로 장치에 SSM Agent를 설치해야 합니다. 자세한 내용은 하이브리드 환경에서	2021년 11월 29일

	AWS Systems Manager 설정을 참조하세요.	
원격 데스크톱을 사용하여 인스턴스에 연결합니다.	이제 Fleet Manager를 사용하여 원격 데스크톱 프로토콜(RDP)을 사용하여 관리형 Windows 인스턴스에 연결할 수 있습니다. NICE DCV로 구동되는 이러한 원격 데스크톱 세션은 브라우저에서 직접 인스턴스에 대한 보안 연결을 제공합니다. 자세한 내용은 원격 데스크톱을 사용하여 연결하기 섹션을 참조하세요.	2021년 11월 23일
최대 세션 기간 지정 및 세션 이유 제공	이제 AWS 계정의 AWS 리전에서 모든 Session Manager 세션 지속 시간을 지정할 수 있습니다. 세션이 지정한 기간에 도달하면 세션이 종료됩니다. 이제 세션을 시작할 때 이유를 선택적으로 추가할 수도 있습니다. 자세한 내용은 최대 세션 기간 지정 섹션을 참조하세요.	2021년 11월 16일

Patch Manager에서 이제 Raspberry Pi OS 운영 체제 지원	이제 Patch Manager을 사용하여 Raspberry Pi OS 인스턴스를 패치할 수 있습니다. Patch Manager는 Raspberry Pi OS 9(Stretch) 및 10(Buster) 패치를 지원합니다. Raspberry Pi OS는 Debian 기반 OS이기 때문에 Debian Server와 동일한 패치 적용 규칙이 많이 적용됩니다. 자세한 정보는 다음 주제를 참조하세요. • 보안 패치 선택 방법 • 패치 설치 방법 • Debian Server 및 Raspberry Pi OS에서 패치 기준 규칙이 작동하는 방법	2021년 11월 16일
Red Hat Knowledgebase 포털 액세스	Red Hat 제품을 사용하는 솔루션, 아티클, 문서 및 영상을 찾기 위해 RHEL Knowledgebase 포털에 액세스할 때 Fleet Manager를 사용합니다. 자세한 내용은 Red Hat 지식베이스 포털 액세스 섹션을 참조하세요.	2021년 11월 3일
OpsItems 대량 편집	OpsCenter에서 OpsItems 대량 편집을 지원합니다. 여러 OpsItems를 선택하고 Status(상태), Priority(우선순위), Severity(심각도), Category(카테고리) 필드 중 하나를 편집합니다. 자세한 내용은 OpsItems 편집을 참조하세요.	2021년 10월 15일

AWS 리소스를 채우는 입력 파라미터 생성	이제 AWS Management Console에서 AWS 리소스를 채우는 Automation 실행서의 입력 파라미터를 생성할 수 있습니다. 자세한 내용은 AWS 리소스를 채우는 입력 파라미터 생성을 참조하세요.	2021년 10월 14일
유지 관리 기간에 대한 새로운 작업 호출 컷오프 옵션	이제 유지 관리 기간에 지정된 컷오프 시간에 도달한 후 새 작업 호출이 시작되지 않도록 차단할 수 있습니다. 자세한 내용은 유지 관리 기간에 작업 할당(콘솔)을 참조하세요.	2021년 10월 13일
macOS 11.3.1 및 11.4에 대한 Patch Manager 지원(Big Sur)	이제 macOS 11.3.1 및 11.4(Big Sur)에 대한 Amazon Elastic Compute Cloud(Amazon EC2)에서Patch Manager를 사용하여 패치할 수 있습니다. 이는 macOS 10.14.x(Mojave) 및 10.15.x(Catalina)에 대한 기존 지원에 추가되는 사항입니다. Patch Manager로 하는 작업에 대한 자세한 정보는 AWS Systems ManagerPatch Manager 섹션을 참조하세요.	2021년 10월 1일

Application Manager의 Application Insights	Application Manager는 Amazon CloudWatch Application Insights와 통합됩니다. Application Insights는 애플리케이션 리소스와 기술 스택 전반에 걸쳐 주요 지표, 로그, 경보를 식별하고 설정합니다. Application Insights는 지표와 로그를 지속적으로 모니터링하여 이상과 오류를 감지하고 연결합니다. 시스템이 오류 및 이상을 감지하면 Application Insights에서 알림을 설정하고 작업을 수행할 수 있는 CloudWatch Events를 생성합니다. Application Manager의 개요(Overview) 및 모니터링(Monitoring) 탭에서 Application Insights를 활성화하고 볼 수 있습니다. Application Insights에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 Amazon CloudWatch Application Insights란 무엇인가요?를 참조하세요.	2021년 9월 21일

다른 캘린더에서 Change Calendar로 이벤트 가져오기	이제 서드 파티 캘린더에서 Change Calendar의 캘린더로 이벤트를 가져올 수 있습니다. 이전에는 각 이벤트를 캘린더에 수동으로 입력해야 했습니다. 지원되는 서드 파티 캘린더 제공자에서 iCalendar (`.ics`) 파일로 캘린더를 내보낸 후 Change Calendar로 가져오고 해당 이벤트는 Systems Manager의 공개 또는 비공개 달력에 대한 규칙에 포함됩니다. 지원되는 공급자에는 iCloud 캘린더, Google 캘린더, Microsoft Outlook가 있습니다. 자세한 내용은 서드 파티 일정으로부터 이벤트 가져오기 및 관리를 참조하세요.	2021년 9월 8일
Application Manager에서 새로운 태그 지정 및 실행서 기능	태그 지정 개선 사항에는 Application Manager 애플리케이션에서 특정 리소스 또는 모든 리소스에서 태그를 추가하거나 삭제할 수 있는 기능이 포함됩니다. 실행서 개선 사항에는 특정 리소스 유형에 대해 필터링된 실행서 목록을 보거나 동일한 유형의 모든 리소스에서 실행서를 시작하는 기능이 포함됩니다. 자세한 내용은 Application Manager에서 태그로 작업 및 Application Manager에서 실행서로 작업을 참조하세요.	2021년 8월 31일

새 예제: AWS CLI를 사용하여 변경 요청 생성	AWS CLI를 사용하여 변경 요청을 작성하는 예제가 Change Manager 챕터에 추가되었습니다. 이 예제에서는 샘플 `AWS-HelloWorldChangeTemplate` 변경 템플릿 및 `AWS-HelloWorld` runbook을 사용합니다. • 변경 요청 생성(AWS CLI)	2021년 8월 20일
새 섹션: Amazon EKS에서 파라미터 사용	새 섹션이 Parameter Store 챕터에 추가되었습니다. 이 주제에서는 Amazon EKS 클러스터에서 파라미터를 사용하는 방법에 대해 시연합니다. 자세한 내용은 Amazon Elastic Kubernetes Service에서 Parameter Store 파라미터 사용을 참조하세요.	2021년 8월 19일
업데이트된 Patch Manager 수명 주기 후크	Patch Manager에서 이제 지금 패치 패치 적용 작업 동안 추가 포인트가 되는 수명 주기 후크를 제공하여 Systems Manager 명령 문서를 실행할 수 있습니다. 지금 패치 실행 후 인스턴스 재부팅을 예약하는 경우 재부팅이 완료된 후 실행할 수명 주기 후크를 지정할 수 있습니다. 자세한 내용은 '지금 패치' 수명 주기 후크 사용 및 `AWS-RunPatchBaselineWithHooks` SSM 문서 정보를 참조하세요.	2021년 8월 9일

이제Change Manager 요청에 대해 자동 승인이 지원됩니다.	이제 자동 승인을 지원하도록 Change Manager에서 변경 템플릿을 구성할 수 있습니다. 즉, 필요한 IAM 권한이 있는 사용자가 추가 승인 없이 변경 요청을 시작하도록 선택할 수 있습니다. 자동 승인 템플릿에 대한 액세스 권한이 있는 사용자는 원하는 경우 승인자를 지정하도록 선택할 수 있습니다. Change Manager 프로세스를 제어할 수 있도록 변경 고정 기간 동안 모든 요청에 대해 승인이 여전히 필요합니다. 자세한 정보는 다음 주제를 참조하세요. • 변경 템플릿 생성 • 변경 요청 생성 • AWS 관리형 Hello World 변경 템플릿 사용해 보기	2021년 7월 30일
OpsCenter 운영 인사이트	OpsCenter는 사용자의 계정에 있는 OpsItems를 자동으로 분석하여 인사이트를 생성합니다. 인사이트에는 계정에 얼마나 많은 중복 OpsItems가 있고 어떤 소스에서 중복 OpsItems가 생성되는지 이해하는 데 도움이 되는 정보가 포함되어 있습니다. 인사이트는 또한 중복 OpsItems를 해결하는 데 도움이 되는 권장 모범 사례와 Automation 실행서도 제공합니다. 자세한 내용은 운영 인사이트 작업을 참조하세요.	2021년 7월 13일

Fleet Manager에서 중지된 인스턴스 보기	이제 Fleet Manager 콘솔에서 어떤 인스턴스가 `running`이고 어떤 인스턴스가 `stopped`인지 볼 수 있습니다. 자세한 내용은 AWS Systems ManagerFleet Manager 섹션을 참조하세요.	2021년 7월 12일
새로운 주제: Automation 실행서 작성	새로운 주제인 Automation 실행서 작성에서는 사용자 정의 Automation 실행서의 콘텐츠를 작성하는 방법에 대한 지침과 설명 예제를 제공합니다.	2021년 7월 8일

Application Manager에서 AWS CloudFormation 스택 및 템플릿 생성	Application Manager는 CloudFormation과 통합하여 애플리케이션에 대한 리소스를 프로비저닝하고 관리하는 데 도움이 됩니다. Application Manager에서는 AWS CloudFormation 템플릿과 스택을 생성, 수정, 삭제할 수 있습니다. Application Manager에는 템플릿을 복제, 생성 및 저장할 수 있는 템플릿 라이브러리도 포함되어 있습니다. Application Manager와 CloudFormation은 스택의 현재 상태에 대해 동일한 정보를 표시합니다. 템플릿 및 템플릿 업데이트는 스택을 프로비저닝할 때까지 Systems Manager에 저장되며, 이때 변경 사항은 CloudFormation에도 표시됩니다. 자세한 내용은 Application Manager에서 AWS CloudFormation 스택 작업을 참조하세요.	2021년 7월 8일
새 주제: 하이브리드 인스턴스에서 SSM Agent에 대한 프라이빗 키 자동 교체	새로운 주제인 프라이빗 키 자동 교체 설정에서는 하이브리드 환경 프라이빗 키를 자동으로 교체하도록 SSM Agent를 구성하여 보안 태세를 강화하는 방법에 관한 지침을 제공합니다.	2021년 6월 15일

AWS CLI 버전 1.2.205.0용 Session Manager 플러그인	AWS CLI용 Session Manager 플러그인의 새 버전이 릴리스되었습니다. 자세한 내용은 Session Manager 플러그인 최신 버전 및 릴리스 기록을 참조하세요.	2021년 6월 10일
새 IAM 서비스 연결 역할	OpsCenter 운영 인사이트를 사용하면 Systems Manager가 `AWSSSMOpsInsightsServiceRolePolicy` 라는 새로운 AWS Identity and Access Management(IAM) 서비스 연결 역할을 생성합니다. 이 역할에 대한 자세한 내용은 역할을 사용하여 Systems Manager OpsCenter에서 운영 인사이트 OpsItems 생성: `AWSSSMOpsInsightsServiceRolePolicy` 를 참조하세요.	2021년 6월 9일
Linux용 새로운 Patch Manager 문제 해결 콘텐츠	새로운 주제인 Linux에서 `AWS-RunPatchBaseline` 실행 시 오류 발생은 Linux 운영 체제로 관리형 인스턴스를 패치할 때 발생할 수 있는 여러 가지 문제에 대한 설명과 해결 방법을 제공합니다.	2021년 6월 8일

지정된 대상이 필요하지 않은 유지 관리 기간 태스크에 대한 지원 개선(콘솔)	이제 필요하지 않은 경우 태스크에서 대상을 지정하지 않고도 콘솔에서 유지 관리 기간 태스크를 생성할 수 있습니다. 이전에는 AWS CLI 또는 API를 사용할 때만 이 옵션을 사용할 수 있었습니다. 이 옵션은 Automation, AWS Lambda 및 AWS Step Functions 태스크 유형에 적용됩니다. 예를 들어 Automation 태스크를 생성하고 업데이트할 리소스가 Automation 문서 파라미터에 지정된 경우 더 이상 태스크 자체에서 대상을 지정할 필요가 없습니다. 자세한 내용은 대상 없이 유지 관리 기간 태스크 등록, 유지 관리 기간에 태스크 할당(콘솔) 및 유지 관리 기간으로 자동화 예약을 참조하세요.	2021년 5월 28일
Automation 런북 참조 재배치됨	Automation 실행서 참조가 새 위치로 이동되었습니다. 자세한 내용은 Systems Manager Automation 실행서 참조를 참조하세요.	2021년 5월 10일
AWS Systems Manager Incident Manager 시작	Incident Manager는 사용자가 AWS 호스팅 애플리케이션에 영향을 미치는 인시던트를 완화하고 복구하는 데 도움이 되도록 설계된 인시던트 관리 콘솔입니다. 자세한 내용은 AWS Systems Manager Incident Manager 사용 설명서를 참조하십시오.	2021년 5월 10일

State Manager는 Change Calendar를 지원합니다.	이제 State Manager 연결을 생성하거나 업데이트할 때 Change Calendar 이름 또는 Amazon 리소스 이름(ARN)을 지정할 수 있습니다. State Manager는 변경 일정이 닫혀 있을 때가 아니라 열려 있을 때만 연결을 적용합니다. 자세한 내용은 연결 생성 및 새로운 연결 버전 편집 및 생성을 참조하세요.	2021년 5월 6일
Clone Systems Manager 문서	이제 Systems Manager Documents 콘솔을 사용하여 기존 문서의 내용을 수정할 수 있는 새 문서로 복사할 수 있습니다. 자세한 내용은 SSM 문서 복제를 참조하세요.	2021년 5월 4일

Explorer 및 OpsCenter와 Security Hub 통합	이제 Explorer 및 OpsCenter를 AWS Security Hub와 통합할 수 있습니다. Security Hub에서는 AWS에서 보안 상태를 포괄적으로 파악할 수 있으며 보안 업계 표준 및 모범 사례와 비교하여 환경을 확인할 수 있습니다. Explorer와 통합하면 Explorer 대시보드의 Security Hub 위젯에서 보안 결과를 볼 수 있습니다. OpsCenter와 통합하면 Security Hub 결과에 대해 OpsItems를 생성할 수 있습니다. 자세한 내용은 Explorer에서 AWS Security Hub로부터 결과 수신 및 OpsCenter에서 AWS Security Hub로부터 결과 수신을 참조하세요.	2021년 4월 27일
새로운 주제: 문서 규칙	사용자가 AWS Systems Manager User Guide의 일반적인 인쇄 규칙을 이해하는 데 도움이 되도록 새로운 주제가 추가되었습니다. 자세한 내용은 문서 규칙을 참조하세요.	2021년 4월 21일

업데이트된 주제: Microsoft에서 릴리스한 Windows Server 기반 애플리케이션 패치 정보	이제 Patch Manager가 Windows Server 관리형 인스턴스에서 Microsoft가 릴리스한 애플리케이션을 패치할 수 있도록 인스턴스에서 Windows 업데이트 옵션 Windows를 업데이트할 때 다른 Microsoft 제품에 대한 업데이트 제공(Give me updates for other Microsoft products when I update Windows)을 허용해야 함이 Windows Server에서 Microsoft가 릴리스한 애플리케이션 패치 정보 주제에 명시되어 있습니다.	2021년 4월 12일
Automation 런북 참조 재구성	필요한 실행서를 찾고 참조를 보다 효율적으로 탐색할 수 있도록 관련 AWS 서비스별 Automation 실행서 참조의 콘텐츠가 재구성되었습니다. 이러한 변경 사항을 확인하려면 Systems Manager Automation 런북 참조를 참조하세요.	2021년 4월 12일

Patch Manager: .csv 패치 규정 준수 보고서 생성	Patch Manager는 이제 인스턴스에 대한 패치 규정 준수 보고서를 생성하고 선택한 S3 버킷에 .csv 형식으로 보고서를 저장하는 기능을 지원합니다. 그런 다음 Amazon QuickSight와 같은 도구를 사용하여 패치 규정 준수 보고서 데이터를 분석할 수 있습니다. 단일 인스턴스 또는 AWS 계정의 모든 인스턴스에 대한 패치 규정 준수 보고서를 생성할 수 있습니다. 온디맨드로 일회성 보고서를 생성하거나 보고서가 자동으로 생성되도록 일정을 설정할 수 있습니다. 보고서가 생성될 때 알림을 제공하도록 Amazon Simple Notification Service 주제를 지정할 수도 있습니다. 자세한 내용은 CSV 패치 규정 준수 보고서 생성을 참조하세요.	2021년 4월 9일
Parameter Store 파라미터 레이블 삭제	이제 Systems Manager 콘솔이나 AWS CLI를 사용하여 Parameter Store 파라미터 레이블을 삭제할 수 있습니다. 자세한 내용은 파라미터 레이블 작업을 참조하세요.	2021년 4월 6일

지금 패치 사용 시 인스턴스 재부팅 예약	Patch Manager는 이제 지금 패치 기능을 사용하여 패치를 설치한 후 인스턴스를 재부팅할 시간을 예약할 수 있도록 지원합니다. 이는 패치 설치를 완료하기 위해 필요한 경우에만 인스턴스를 재부팅하거나 패치 작업 후 모든 재부팅을 건너뛰는 기존 옵션에 추가됩니다. 자세한 내용은 온디맨드로 인스턴스 패치를 참조하세요.	2021년 4월 1일
새로운 주제: 퍼블릭 파라미터 검색	이제 AWS CLI 또는 Systems Manager 콘솔을 사용하여 Parameter Store 퍼블릭 파라미터를 찾을 수 있습니다. 자세한 내용은 퍼블릭 파라미터 찾기를 참조하세요.	2021년 4월 1일
패치 지금 업데이트: S3에 로그 저장 및 수명 주기 후크 실행	Patch Manager Patch now(지금 패치) 작업을 실행할 때 패치 로그를 자동으로 저장할 S3 버킷을 선택할 수 있습니다. 또한 작업 중 설치 전, 설치 후, 종료 시, 이렇게 세 지점에서 수명 주기 후크로 Systems Manager Command 문서(SSM 문서)를 실행하도록 선택할 수 있습니다. 자세한 내용은 온디맨드로 인스턴스 패치를 참조하세요.	2021년 3월 31일

Systems Manager에서 이제 AWS 관리형 정책에 대한 변경 사항 보고 가능	2021년 3월 24일부터 AWS 관리형 정책에 대한 Systems Manager 업데이트 주제에서 관리형 정책의 변경 사항을 보고합니다. 나열된 첫 번째 변경 사항은 여러 계정 및 리전에서 OpsData 및 OpsItems를 보고하는 Explorer 기능에 대한 지원 추가입니다.	2021년 3월 24일
Explorer는 AWS Organizations의 계정을 기반으로 리소스 데이터 동기화를 위한 모든 OpsData 소스를 자동으로 허용합니다.	리소스 데이터 동기화를 생성할 때 AWS Organizations 옵션 중 하나를 선택하면 Systems Manager는 조직 또는 선택한 조직 단위의 모든 AWS 계정에 대해 선택한 AWS 리전의 모든 OpsData 소스를 자동으로 허용합니다. 즉, 예를 들어 AWS 리전에서 Explorer를 허용하지 않은 경우에도 리소스 데이터 동기화에 대해 AWS Organizations 옵션을 선택하면 Systems Manager가 자동으로 해당 리전에서 OpsData를 수집합니다. 자세한 내용은 여러 계정 및 리전 리소스 데이터 동기화 정보를 참조하세요.	2021년 3월 24일
Systems Manager Automation은 실행서를 위한 새로운 시스템 변수 제공	새로운 `global:AWS_PARTITION` 시스템 변수를 사용하여 실행서를 작성할 때 리소스가 있는 AWS 파티션을 지정할 수 있습니다. 자세한 내용은 Automation 시스템 변수를 참조하세요.	2021년 3월 18일

Change Manager 변경 요청에 대해 여러 수준의 승인 허용	Change Manager 변경 템플릿을 생성할 때 이제 두 수준 이상의 승인자가 변경 요청 실행에 대한 권한을 부여하도록 할 수 있습니다. 예를 들어 기술 검토자가 먼저 변경 템플릿에서 생성된 변경 요청을 승인한 다음 한 명 이상의 관리자에게 두 번째 수준의 승인을 요구할 수 있습니다. 자세한 내용은 변경 템플릿 생성을 참조하세요.	2021년 3월 4일
Patch Manager에서 이제 Oracle Linux 8.x 지원	이제 Patch Manager를 사용하여 버전 8.3을 통해 Oracle Linux 8.x 인스턴스를 패치할 수 있습니다. 자세한 정보는 다음 주제를 참조하세요. • 보안 패치 선택 방법 • 패치 설치 방법 • Oracle Linux에서 패치 기준 규칙 작동 방법	2021년 3월 1일
OpsCenter에서 선택한 리소스에 대해 다른 OpsItems 표시	문제를 조사하고 문제에 대한 컨텍스트를 제공하는 데 도움이 되도록 특정 AWS 리소스에 대한 OpsItems 목록을 볼 수 있습니다. 목록에는 각 OpsItem의 상태, 심각도 및 제목이 표시됩니다. 목록에는 각 OpsItem에 대한 딥 링크도 포함되어 있습니다. 자세한 내용은 특정 리소스에 대한 다른 OpsItems 보기를 참조하세요.	2021년 3월 1일

런타임 시 패치 기본 설정 정의	이제 기준 재정의 기능을 사용하여 런타임 시 패치 기본 설정을 정의할 수 있습니다. 자세한 내용은 BaselineOverride 파라미터 사용을 참조하세요.	2021년 2월 25일
새 Systems Manager 문서 유형	이제 AWS CloudFormation 템플릿을 Systems Manager 문서로 저장할 수 있습니다. CloudFormation 템플릿을 Systems Manager 문서로 저장하면 버전 관리, 버전 콘텐츠 비교, 계정과 공유 등의 Systems Manager 문서 기능을 활용할 수 있습니다. 자세한 내용은 AWS Systems Manager 문서를 참조하세요.	2021년 2월 9일
옵션 후크를 사용하여 인스턴스 패치	새로운 SSM 문서 `AWS-RunPatchBaselineWithHooks` 는 인스턴스 패치 주기 동안 세 지점에서 SSM 문서를 실행하는 데 사용할 수 있는 후크를 제공합니다. `AWS-RunPatchBaselineWithHooks` 에 대한 자세한 내용은 `AWS-RunPatchBaselineWithHooks` SSM 문서 정보를 참조하세요. 세 후크를 모두 사용하는 패치 작업의 샘플 시연은 시연: 애플리케이션 종속성 업데이트, 인스턴스 패치 및 애플리케이션별 상태 확인 수행을 참조하세요.	2021년 2월 2일

새로운 주제: 하드웨어 지문을 사용하여 온프레미스 서버 및 가상 머신 검증	SSM Agent는 계산된 지문을 사용하여 서비스에 등록한 온프레미스 서버와 가상 머신 및 VM의 자격 증명을 확인합니다. 지문은 에이전트가 특정 Systems Manager API에 전달하는 볼트에 저장된 불투명 문자열입니다. 하드웨어 지문에 대한 정보와 시스템 검증을 지원하기 위한 유사성 임계값 구성 지침은 하드웨어 지문을 사용하여 온프레미스 서버 및 가상 머신 검증을 참조하세요.	2021년 1월 25일
새로운 주제: SSM Agent 기술 참조	SSM Agent 기술 참조 주제에서는 AWS Systems Manager SSM Agent를 구현하고 에이전트 작동 방식을 이해하는 데 도움이 되는 정보를 제공합니다. 이 주제에는 완전히 새로운 섹션인 AWS 리전의 SSM Agent 롤링 업데이트가 포함되어 있습니다.	2021년 1월 21일
Windows Server 2008의 SSM Agent	2020년 1월 14일부로 Windows Server 2008은 Microsoft의 기능 또는 보안 업데이트에 대해 더 이상 지원되지 않습니다. Windows Server 2008 AMIs에는 SSM Agent가 포함되어 있지만 에이전트는 이 운영 체제에 대해 더 이상 업데이트되지 않습니다.	2021년 1월 5일

지정된 대상이 필요하지 않은 유지 관리 기간 태스크에 대한 지원 개선(AWS CLI 및 API만 해당)	이제 필요하지 않은 경우 태스크에서 대상을 지정하지 않고도 유지 관리 기간 태스크를 생성할 수 있습니다(AWS CLI 및 API만 해당). 이는 Automation, AWS Lambda 및 AWS Step Functions 태스크 유형에 적용됩니다. 예를 들어 Automation 태스크를 생성하고 업데이트할 리소스가 Automation 런북 파라미터에 지정된 경우 더 이상 태스크 자체에서 대상을 지정할 필요가 없습니다. 자세한 내용은 대상 없이 유지 관리 기간 태스크 등록 및 유지 관리 기간으로 자동화 예약을 참조하세요.	2020년 12월 23일
새로운 Automation 기능	Systems Manager Automation 실행서에 새로운 공유 속성이 추가되었습니다. `onCancel` 속성을 사용하면 사용자가 자동화를 취소하는 경우 자동화가 수행해야 하는 단계를 지정할 수 있습니다. 자세한 내용은 모든 작업에서 공유하는 속성을 참조하세요.	2020년 12월 21일
새로운 주제: IAM을 사용한 연결 작업	IAM을 사용하여 연결 생성에 대한 모범 사례를 설명하는 새로운 주제가 Systems Manager State Manager 장에 추가되었습니다. 자세한 내용은 IAM을 사용한 연결 작업을 참조하십시오.	2020년 12월 18일

State Manager에서 이제 다중 리전과 다중 계정 지원	이제 여러 리전 또는 계정으로 연결을 생성하거나 업데이트할 수 있습니다. 자세한 내용은 연결 생성을 참조하세요.	2020년 12월 15일
새로운 기능: Fleet Manager	AWS Systems Manager의 기능인 Fleet Manager는 AWS 또는 온프레미스에서 실행되는 서버 플릿을 원격으로 관리하는 데 도움이 되는 통합 사용자 인터페이스(UI) 환경입니다. Fleet Manager를 사용하면 하나의 콘솔에서 전체 서버 플릿의 상태 및 성능 상태를 볼 수 있습니다. 또한 개별 인스턴스에서 데이터를 수집하여 콘솔에서 일반적인 문제 해결 및 관리 태스크를 수행할 수 있습니다. 자세한 내용은 AWS Systems ManagerFleet Manager 섹션을 참조하세요.	2020년 12월 15일

새로운 기능: Change Manager	Amazon Web Services에서 애플리케이션 구성 및 인프라에 대한 운영 변경을 요청, 승인, 구현 및 보고하기 위한 엔터프라이즈 변경 관리 프레임워크인 Change Manager를 릴리스했습니다. 하나의 위임된 관리자 계정에서 AWS Organizations를 사용하면 여러 AWS 리전에 있는 여러 AWS 계정의 변경사항을 관리할 수 있습니다. 또는 로컬 계정을 사용하여 하나의 AWS 계정에 대한 변경 사항을 관리할 수 있습니다. AWS 리소스와 온프레미스 리소스 모두에 대한 변경 사항을 관리하려면 Change Manager를 사용합니다. 자세한 내용은 AWS Systems ManagerChange Manager 섹션을 참조하세요.	2020년 12월 15일
새로운 기능: Application Manager	Application Manager은(는) 애플리케이션의 컨텍스트에서 AWS 리소스의 문제를 조사하고 수정하는 데 도움이 됩니다. Application Manager은(는) 여러 AWS 서비스 및 Systems Manager 기능의 작업 정보를 하나의 AWS Management Console(으)로 집계합니다. 자세한 내용은 AWS Systems ManagerApplication Manager 섹션을 참조하세요.	2020년 12월 15일

AWS Systems Manager에서 macOS용 Amazon EC2 인스턴스를 지원합니다.	macOS 인스턴스에 대한 Amazon Elastic Compute Cloud(Amazon EC2) 지원 릴리스와 함께 Systems Manager는 이제 macOS용 EC2 인스턴스에 대한 많은 작업을 지원합니다. macOS 10.14.x(Mojave) 및 10.15.x(Catalina) 버전이 지원됩니다. 자세한 내용은 다음 항목을 참조하십시오. • macOS의 EC2 인스턴스에 SSM Agent 설치에 대한 자세한 내용은 macOS의 EC2 인스턴스에 SSM Agent 설치 및 구성을 참조하세요. • macOS에서 EC2 인스턴스 패치에 대한 자세한 내용은 패치 설치 방법 및 사용자 지정 패치 기준 생성(macOS)을 참조하세요. • macOS용 EC2 인스턴스 지원에 대한 자세한 내용은 Amazon EC2 사용 설명서의 Amazon EC2 Mac 인스턴스를 참조하세요.	2020년 11월 30일

유지 관리 기간 유사 파라미터: {{TARGET_ID}} 및 {{RESOURCE_ID}} 에 대해 지원되는 새 리소스 유형	이제 의사 파라미터 {{TARGET_ID}} 및 {{RESOURCE_ID}} 에 추가 리소스 유형을 사용할 수 있습니다. 이제 이 두 의사 파라미터에 리소스 유형 AWS::RDS::DBCluster 를 사용할 수 있습니다. 유지 관리 기간 의사 파라미터에 대한 자세한 내용은 유지 관리 기간 작업 등록 시 의사 파라미터 사용을 참조하세요.	2020년 11월 27일
AWS CLI 버전 1.2.30.0용 Session Manager 플러그인	AWS CLI용 Session Manager 플러그인의 새 버전이 릴리스되었습니다. 자세한 내용은 Session Manager 플러그인 최신 버전 및 릴리스 기록을 참조하세요.	2020년 11월 24일
새로운 주제: SSM 문서 버전 비교	이제 Systems Manager Documents 콘솔에서 SSM 문서 버전 간의 내용 차이를 비교할 수 있습니다. 자세한 내용은 SSM 문서 버전 비교를 참조하세요.	2020년 11월 24일
Systems Manager에서 이제 VPC 엔드포인트 정책 지원	이제 Systems Manager용 VPC 인터페이스 엔드포인트에 대한 정책을 생성할 수 있습니다. 자세한 내용은 인터페이스 대한 VPC 엔드포인트 정책 생성을 참조하세요.	2020년 11월 18일

새로운 주제: 유휴 세션 시간 제한 값 지정	이제 Session Manager로 세션이 끝나기 전에 사용자의 비활성 상태를 허용하는 시간을 지정할 수 있습니다. 자세한 내용은 유휴 세션 시간 제한 값 지정을 참조하세요.	2020년 11월 18일
새로운 Session Manager 로깅 기능	이제 JSON 형식의 세션 데이터 로그의 연속 스트림을 Amazon CloudWatch Logs로 보낼 수 있습니다. 자세한 내용은 Amazon CloudWatch Logs를 사용하여 세션 데이터 스트리밍을 참조하세요.	2020년 11월 18일
새로운 주제: SSM Agent의 서명 확인	이제 Linux 인스턴스의 SSM Agent에 대한 설치 관리자 패키지의 암호화 서명을 확인할 수 있습니다. 자세한 내용은 SSM 문서 스키마 및 기능을 참조하세요.	2020년 11월 17일
새로운 주제: 자동화 상태 이해	작업 및 자동화의 상태를 설명하는 새로운 주제가 Systems Manager Automation 장에 추가되었습니다. 자세한 내용은 자동화 상태 이해를 참조하세요.	2020년 11월 17일
`aws:downloadContent` 플러그에 대한 새로운 소스 유형	이제 Git 및 HTTP가 `aws:downloadContent` 플러그인의 소스 유형으로 지원됩니다. 자세한 내용은 `aws:downloadContent` 단원을 참조하십시오.	2020년 11월 17일

새로운 Systems Manager 문서(SSM 문서) 스키마 기능	스키마 버전 2.2 이상의 SSM 문서에서 `precondition` 파라미터는 이제 문서의 입력 파라미터 참조를 지원합니다. 자세한 내용은 SSM 문서 스키마 및 기능을 참조하세요.	2020년 11월 17일
Explorer의 새 데이터 원본: AWS Config	이제 Explorer에 준수 및 비준수 AWS Config 규칙의 전체 요약, 준수 및 비준수 리소스 수, 각각에 대한 특정 세부 정보(비준수 규칙 또는 리소스로 드릴다운할 때) 등의 AWS Config 규정 준수 정보가 표시됩니다. 자세한 내용은 Systems Manager Explorer 데이터 원본 편집을 참조하세요.	2020년 11월 11일
새로운 주제: 연결을 사용하여 Auto Scaling 그룹 실행	Auto Scaling 그룹 실행을 위해 연결을 생성하는 모범 사례를 설명하는 새 섹션이 State Manager에 추가되었습니다. 자세한 내용은 연결을 사용하여 Auto Scaling 그룹 실행을 참조하세요.	2020년 11월 10일
Quick Setup에서 이제 리소스 그룹 대상 지정 지원	이제 Quick Setup에서 리소스 그룹을 로컬 설정 유형의 대상으로 선택할 수 있습니다. 자세한 내용은 Quick Setup 대상 선택을 참조하세요.	2020년 11월 5일

Patch Manager에서 Debian Server 10 LTS, Oracle Linux 7.9 LTS 및 Ubuntu Server 20.10 STR 지원 추가	이제 Patch Manager를 사용하여 Debian Server 10 LTS, Oracle Linux 7.9 LTS 및 Ubuntu Server 20.10 STR 인스턴스를 패치할 수 있습니다. 자세한 정보는 다음 주제를 참조하세요. • Patch Manager 필수 조건 • 보안 패치 선택 방법 • 패치 설치 방법 • Debian Server에서 패치 기준 규칙 작동 방법 • Oracle Linux에서 패치 기준 규칙 작동 방법 • Ubuntu Server에서 패치 기준 규칙 작동 방법	2020년 11월 4일
AWS Systems ManagerChange Calendar에 대한 새로운 EventBridge 지원	Amazon EventBridge는 이제 이벤트 규칙에서 Change Calendar 이벤트를 지원합니다. 일정 상태가 변경되면 EventBridge는 EventBridge 규칙을 정의한 대상 작업을 시작할 수 있습니다. EventBridge 및 Systems Manager 이벤트 작업에 대한 자세한 내용은 다음 주제를 참조하세요. • Systems Manager 이벤트에 대해 EventBridge 구성 • 참조: Systems Manager용 Amazon EventBridge 이벤트 패턴 및 유형	2020년 11월 4일

경보에서 OpsItems를 생성하도록 CloudWatch 구성	경보가 ALARM 상태가 되면 Systems Manager OpsCenter에서 OpsItem을 자동으로 생성하도록 Amazon CloudWatch를 구성할 수 있습니다. 이렇게 하면 단일 콘솔에서 AWS 리소스 문제를 빠르게 진단하고 수정할 수 있습니다. 자세한 내용은 경보에서 OpsItems를 생성하도록 CloudWatch 구성을 참조하세요.	2020년 11월 4일
Ubuntu Server 20.10 지원	AWS Systems Manager에서 이제 Ubuntu Server 20.10 단기 릴리스(STR)를 지원합니다. 자세한 정보는 다음 주제를 참조하세요. • 지원되는 운영 체제 • 하이브리드 환경을 위한 SSM Agent 설치(Linux) • Ubuntu Server 인스턴스에 SSM Agent 수동 설치 • SSM Agent 상태 확인 및 에이전트 시작	2020년 10월 22일

새로운 주제: 구성 가능한 셸 프로파일 허용	이제 Session Manager로 사용하여 구성 가능한 셸 프로파일을 허용할 수 있습니다. 구성 가능한 셸 프로파일을 허용하여 세션 내에서 셸 기본 설정, 환경 변수, 작업 디렉터리 및 세션이 시작될 때 여러 명령 실행과 같은 기본 설정을 사용자 지정할 수 있습니다. 자세한 내용은 구성 가능한 셸 프로파일 허용을 참조하세요.	2020년 10월 21일
패치 규정 준수 결과에서 이제 어느 CVE가 어느 패치로 해결되는지 보고 가능	지원되는 대부분의 Linux 시스템에서 관리형 인스턴스에 대한 패치 규정 준수 결과를 볼 때 제공되는 세부 정보에서 이제 어느 일반적인 취약성 및 노출도(CVE) 게시판 문제가 사용 가능한 어느 패치로 해결되는지 보고합니다. 이 정보는 누락되거나 실패한 패치를 얼마나 긴급하게 설치해야 하는지를 결정하는 데 도움이 됩니다. 자세한 내용은 패치 규정 준수 결과 보기를 참조하세요.	2020년 10월 20일

Linux 패치 메타데이터에 대한 지원 확대	이제 Patch Manager에서 사용 가능한 Linux 패치에 대한 많은 세부 정보를 볼 수 있습니다. 아키텍처, epoch, 버전, CVE ID, Advisory ID, Bugzilla ID, 리포지토리 등의 패치 데이터를 보도록 선택할 수 있습니다. 또한 새로 사용 가능한 패치 메타데이터 유형에 따른 필터링과 Linux 운영 체제를 지원하도록 DescribeAvailablePatches API 작업이 업데이트되었습니다. 자세한 정보는 다음 주제를 참조하세요. • 사용 가능한 패치 보기 • AWS Systems Manager API 참조의 DescribeAvailablePatches 및 패치 • AWS CLI 명령 레퍼런스 AWS Systems Manager 섹션의 describe-available-patches	2020년 10월 16일
AWS CLI 버전 1.2.7.0용 Session Manager 플러그인	AWS CLI용 Session Manager 플러그인의 새 버전이 릴리스되었습니다. 자세한 내용은 Session Manager 플러그인 최신 버전 및 릴리스 기록을 참조하세요.	2020년 10월 15일

새로운 주제: Session 문서 스키마	새로운 주제 Session 문서 스키마에서는 Session 문서의 스키마 요소를 설명합니다. 이 정보는 Session Manager에 사용하는 세션 유형에 대한 기본 설정을 지정하는 사용자 정의 Session 문서를 만드는 데 도움이 될 수 있습니다.	2020년 10월 15일
새로운 주제: SSM 문서에 대한 무료 텍스트 검색	Systems Manager 문서 페이지의 검색 상자에서 이제 자유 텍스트 검색을 지원합니다. 자유 텍스트 검색은 입력한 검색어를 각 SSM 문서의 문서 이름과 비교합니다. 자세한 내용은 자유 텍스트 검색 사용을 참조하세요.	2020년 10월 15일
새로운 주제: Amazon EC2 관리형 인스턴스 가용성 문제 해결	새로운 주제 Amazon EC2 관리형 인스턴스 가용성 문제 해결은 실행 중임을 확인한 Amazon EC2 인스턴스를 Systems Manager의 사용 가능한 관리형 인스턴스 목록에서 사용할 수 없는 이유를 조사하는 데 도움이 됩니다.	2020년 10월 6일

Parameter Store 장 재구성	필요한 정보를 보다 효율적으로 찾을 수 있도록 AWS Systems Manager User Guide의 Parameter Store 장이 재구성되었습니다. 이제 대부분의 내용이 Setting up Parameter Store 및 Working with Parameter Store 섹션에 정리되어 있습니다. 또한 다음 섹션을 포함하도록 AWS Systems ManagerParameter Store 주제가 확장되었습니다. • Parameter Store가 조직에 주는 이점은 무엇인가요? • Parameter Store는 누가 사용해야 하나요? • Parameter Store에는 어떤 기능이 있나요? • 파라미터란 무엇인가요?	2020년 10월 1일
새로운 패치 규정 준수 관련 주제	패치 규정을 위반하는 관리형 인스턴스를 식별하고, 다양한 유형의 패치 규정 준수 검사를 이해하고, 인스턴스가 규정을 준수하도록 하기 위한 적절한 단계를 수행하는 데 도움이 되도록 다음 주제가 추가되었습니다. • 규정 미준수 인스턴스 식별 • 규정 미준수 인스턴스 패치 적용 • 패치 규정 준수 결과 보기	2020년 9월 24일

SSM Agent 버전 3.0	Systems Manager가 새 버전의 SSM Agent를 시작했습니다.	2020년 9월 21일
새로운 주제 및 업데이트된 주제: Amazon EventBridge가 CloudWatch Events의 이벤트 관리 기능 대체	CloudWatch Events 및 EventBridge는 동일한 기본 서비스 및 API이지만 EventBridge가 더 많은 기능을 제공하며 이제 AWS에서 이벤트를 관리하는 데 선호되는 방법입니다. CloudWatch 또는 EventBridge에서 변경한 내용은 각 콘솔에 반영됩니다. AWS Systems Manager User Guide 전체에 걸쳐 CloudWatch Events 및 기존 프로시저에 대한 참조가 EventBridge 지원을 반영하도록 업데이트되었습니다. 추가로 다음과 같은 새로운 주제가 추가되었습니다. • Systems Manager 이벤트 모니터링 • Systems Manager 이벤트에 대해 EventBridge 구성 • Systems Manager 대상 유형 예 • 참조: Systems Manager용 Amazon EventBridge 이벤트 패턴 및 유형	2020년 9월 18일

AWS Security Hub 및 Patch Manager 통합	이제 Patch Manager를 AWS Security Hub와 통합할 수 있습니다. Security Hub에서는 AWS에서 보안 상태를 포괄적으로 파악할 수 있으며 보안 업계 표준 및 모범 사례와 비교하여 환경을 확인할 수 있습니다. Patch Manager와 통합되면 Security Hub는 보안 관점에서 플릿의 패치 상태를 모니터링합니다. 자세한 내용은 Patch Manager와 AWS Security Hub 통합을 참조하세요.	2020년 9월 17일

유지 관리 기간 유사 파라미터: {{TARGET_ID}} 및 {{RESOURCE_ID}} 에 대해 지원되는 새 리소스 유형	유지 관리 기간 태스크를 등록할 때 --task-invocation-parameters 옵션을 사용하여 4가지 각 태스크 유형에 고유한 파라미터를 지정합니다. {{TARGET_ID}} , {{RESOURCE_ID}} 등의 의사 파라미터 구문을 사용하여 특정 값을 참조할 수도 있습니다. 유지 관리 기간 작업이 실행되면 의사 파라미터 자리표시자 대신 올바른 값을 전달합니다. 2개의 추가 리소스 유형을 이제 의사 파라미터 {{TARGET_ID}} 및 {{RESOURCE_ID}} 에 사용할 수 있습니다. 이제 이 두 의사 파라미터에 리소스 유형 AWS::RDS::DBInstance 및 AWS::SSM::ManagedInstance 를 사용할 수 있습니다. 유지 관리 기간 의사 파라미터에 대한 자세한 내용은 유지 관리 기간 작업 등록 시 의사 파라미터 사용을 참조하세요.	2020년 9월 14일

새로운 '지금 패치(Patch now)' 옵션을 사용하여 온디맨드로 인스턴스 패치	이제 Systems Manager 콘솔을 사용하여 언제든지 인스턴스를 패치하거나 누락된 패치를 검사할 수 있습니다. 일정을 생성하거나 수정하지 않고도 이 작업을 수행할 수 있으며, 즉시 패치를 위한 전체 패치 작업 구성 옵션을 지정할 수 있습니다. 패치를 검사할지 아니면 설치할지 지정하고 작업의 대상 인스턴스를 식별하기만 하면 됩니다. Patch Manager는 인스턴스 유형에 대한 현재 기본 패치 기준을 자동으로 적용하고 한 번에 패치되는 인스턴스 수와 작업이 실패하기 전에 허용되는 오류 수에 대한 모범 사례 옵션을 적용합니다. 자세한 내용은 온디맨드로 인스턴스 패치를 참조하세요.	2020년 9월 9일
새로운 주제: SSM Agent 상태 확인 및 에이전트 시작	새로운 주제 SSM Agent 상태 확인 및 에이전트 시작에서는 지원되는 각 운영 체제에서 SSM Agent가 실행 중인지 확인하는 명령을 제공합니다. 에이전트가 실행 중이 아닌 경우 에이전트를 시작하는 명령도 제공합니다.	2020년 9월 7일

Patch Manager가 이제 Ubuntu Server 20.04 LTS 지원	이제 Patch Manager를 사용하여 Ubuntu Server 20.04 LTS 인스턴스를 패치할 수 있습니다. 자세한 정보는 다음 주제를 참조하세요. • 보안 패치 선택 방법 • 패치 설치 방법 • Ubuntu Server에서 패치 기준 규칙 작동 방법	2020년 8월 31일
사용 사례 및 모범 사례에 대한 새로운 주제	사용자가 Maintenance Windows와 State Manager의 차이점을 쉽게 이해할 수 있도록 새로운 주제가 추가되었습니다. 자세한 내용은 State Manager와 Maintenance Windows 중에서 선택을 참조하세요.	2020년 8월 28일
새로운 OpsCenter 기능	OpsCenter에는 Automation 실행서를 신속하게 찾고 실행하여 문제를 해결하는 데 도움이 되는 새로운 기능이 포함되어 있습니다. 자세한 내용은 OpsCenter의 Automation 실행서 기능을 참조하세요.	2020년 8월 19일
Explorer의 새 데이터 원본: AWS Support 사례	이제 Explorer에서 AWS Support 사례에 대한 정보를 표시합니다. AWS Support에서 설정된 Enterprise 또는 Business 계정이 있어야 합니다. 자세한 내용은 Systems Manager Explorer 데이터 원본 편집을 참조하세요.	2020년 8월 13일

Distributor가 이제 Trend Micro의 서드 파티 패키지를 제공합니다.	이제 Distributor에 Trend Micro의 서드 파티 패키지가 포함됩니다. Distributor를 사용하여 관리형 인스턴스에 Trend Micro Cloud One 에이전트를 설치할 수 있습니다. Trend Micro Cloud One을 사용하면 클라우드에서 워크로드를 보호할 수 있습니다. 자세한 내용은 AWSDistributor 단원을 참조하십시오.	2020년 8월 12일
이제 aws:configurePackage 문서 플러그 인에 additionalArguments 파라미터가 포함됩니다.	Systems Manager Command 문서 플러그인 aws:configurePackage 는 이제 새 additionalArguments 파라미터를 사용하여 스크립트에 추가 파라미터 제공을 지원합니다(설치, 제거 및 업데이트). 자세한 내용은 aws:configurePackage 주제를 참조하세요.	2020년 8월 11일
AppConfig 콘텐츠가 별도의 사용 설명서로 이동	AWS AppConfig에 대한 내용은 별도의 사용 설명서로 이동되었습니다. 자세한 내용은 AWSAppConfig란 무엇인가요?를 참조하세요. AppConfig에는 사용 설명서, AppConfig API 참조 및 새로운 AppConfig 워크숍에 대한 링크가 포함된 별도의 설명서 시작 페이지도 있습니다.	2020년 8월 3일

Quick Setup가 이제 AWS Organizations을 지원합니다.	Quick Setup에서 이제 AWS Organizations를 지원하여 여러 계정 및 리전에서 필요한 보안 역할과 일반적으로 사용되는 Systems Manager 기능을 빠르게 구성할 수 있습니다. 자세한 내용은 AWS Systems ManagerQuick Setup 섹션을 참조하세요.	2020년 7월 23일
Explorer의 새 데이터 원본: 연결 규정 준수	이제 Explorer에서 State Manager의 연결 규정 준수 데이터를 표시합니다. 자세한 내용은 Systems Manager Explorer 데이터 원본 편집을 참조하세요.	2020년 7월 23일
Kernel Live Patching을 설정하고 해제하는 새로운 Systems Manager Command 문서	이제 Amazon Linux 2 인스턴스에서 Kernel Live Patching을 설정하거나 해제할 때 Run Command에 문서 `AWS-ConfigureKernelLivePatching` 을 사용할 수 있습니다. 이 문서를 사용하면 이러한 태스크를 위해 사용자 정의 Command 문서를 생성할 필요가 없습니다. 자세한 내용은 Amazon Linux 2 인스턴스에서 커널 라이브 패치 사용을 참조하세요.	2020년 7월 22일

업데이트된 Automation 할당량	속도 제어 자동화를 위한 별도의 대기열을 포함하여 Automation에 대한 서비스 할당량이 업데이트되었습니다. 자세한 내용은 AWS Systems Manager Automation을 참조하세요.	2020년 7월 20일
콘솔을 사용하여 유지 관리 기간에 대한 일정 오프셋 일 수 지정	Systems Manager 콘솔을 사용하여 유지 관리 기간을 실행하기 전에 CRON 표현식에 의해 지정된 날짜 및 시간 이후에 대기할 일 수를 지정할 수 있습니다. (이전에는 AWS SDK 또는 명령줄 도구를 사용할 때만 이 옵션을 사용할 수 있었습니다.) 예를 들어 CRON 표현식이 매월 셋째 화요일 오후 11:30(`cron(0 30 23 ? * TUE#3 *)`)에 실행되도록 유지 관리 기간을 예약하고 일정 오프셋을 2로 지정하면 이 유지 관리 기간은 2일 후 오후 11시 30분까지 실행되지 않습니다. 자세한 내용은 https://docs.aws.amazon.com/systems-manager/latest/userguide/reference-cron-and-rate-expressions.htmlSystems Manager에 대한 Cron 및 Rate 표현식 및 유지 관리 기간에 대한 일정 오프셋 일 수 지정을 참조하세요.	2020년 7월 17일

Run Command를 사용하여 PowerShell 업데이트	Windows Server 2012 및 2012 R2 인스턴스에서 PowerShell을 버전 5.1로 업데이트하는 데 도움이 되도록 AWS Systems Manager User Guide에 시연이 추가되었습니다. 자세한 내용은 Run Command를 사용한 PowerShell 업데이트를 참조하세요.	2020년 6월 30일
Patch Manager가 이제 CentOS 8.0 및 8.1을 지원합니다.	이제 Patch Manager를 사용하여 CentOS 8.0 및 8.1 인스턴스를 패치할 수 있습니다. 자세한 내용은 다음 주제를 참조하세요. • 보안 패치 선택 방법 • 패치 설치 방법 • CentOS에서 패치 기준 규칙의 작동 방법 • CentOS 인스턴스에 SSM Agent 수동 설치 • 하이브리드 Linux 노드에 SSM Agent를 설치하는 방법	2020년 6월 27일

AppConfig와 AWS CodePipeline 통합	AppConfig는 AWS CodePipeline(CodePipeline)에 대한 통합 배포 작업입니다. CodePipeline은 빠르고 안정적인 애플리케이션 및 인프라 업데이트를 위해 릴리스 파이프라인을 자동화하는 데 사용할 수 있는 완전관리형 지속적 제공 서비스입니다. CodePipeline은 정의한 릴리스 모델을 기반으로 코드 변경이 있을 때마다 릴리스 프로세스의 구축, 테스트 및 배포 단계를 자동화합니다. AppConfig와 CodePipeline의 통합은 다음과 같은 이점을 제공합니다. 자세한 내용은 AppConfig와 CodePipeline 통합을 참조하세요. • CodePipeline을 사용하여 오케스트레이션을 관리하는 고객은 이제 전체 코드베이스를 배포할 필요 없이 애플리케이션에 구성 변경 사항을 간단히 배포할 수 있습니다. • AppConfig를 사용하여 구성 배포를 관리하고 싶지만 AppConfig가 현재 코드 또는 구성 스토어를 지원하지 않아 제한을 받는 고객에게 이제 추가 옵션이 있습니다. CodePipeline은 AWS CodeCommit, GitHub, BitBucket 등을 지원합니다.	2020년 6월 25일

새로운 장: 제품 및 서비스 통합	Systems Manager가 AWS 서비스 및 기타 제품 및 서비스와 통합되는 방식을 이해하는 데 도움이 되도록 AWS Systems Manager 사용 설명서에 새로운 장이 추가되었습니다. 자세한 내용은 Systems Manager와 제품 및 서비스 통합을 참조하세요.	2020년 6월 23일
Automation 장 재구성	필요한 항목을 쉽게 찾을 수 있도록 AWS Systems Manager User Guide의 Automation 장의 주제가 재구성되었습니다. 예를 들어 Automation 작업과 Automation 실행서 참조가 이제 장의 최상위 섹션입니다. 자세한 내용은 AWS Systems Manager Automation을 참조하세요.	2020년 6월 23일

유지 관리 기간에 대한 일정 오프셋 일 수 지정	이제 명령줄 도구 또는 AWS SDK를 사용하여 유지 관리 기간을 실행하기 전에 CRON 표현식에 의해 지정된 날짜 및 시간 이후에 대기할 일 수를 지정할 수 있습니다. 예를 들어 CRON 표현식이 매월 셋째 화요일 오후 11:30(`cron(0 30 23 ? * TUE#3 *)`)에 실행되도록 유지 관리 기간을 예약하고 일정 오프셋을 2로 지정하면 이 유지 관리 기간은 2일 후 오후 11시 30분까지 실행되지 않습니다. 자세한 내용은 https://docs.aws.amazon.com/systems-manager/latest/userguide/reference-cron-and-rate-expressions.htmlSystems Manager에 대한 Cron 및 Rate 표현식 및 유지 관리 기간에 대한 일정 오프셋 일 수 지정을 참조하세요.	2020년 6월 19일
Amazon Linux 2 인스턴스의 커널 라이브 패치에 대한 Patch Manager 지원	Amazon Linux 2의 커널 라이브 패치를 사용하면 실행 중인 애플리케이션을 재부팅하거나 중단하지 않고 실행 중인 Linux 커널에 보안 취약성 및 중요 버그 패치를 적용할 수 있습니다. 이제 Patch Manager를 사용하여 이 기능을 허용하고 커널 라이브 패치를 적용할 수 있습니다. 자세한 내용은 Amazon Linux 2 인스턴스에서 커널 라이브 패치 사용을 참조하세요.	2020년 6월 16일

Patch Manager는 Oracle Linux 버전 지원 증가	이전에는 Patch Manager가 Oracle Linux 버전 7.6만 지원했습니다. 이제 Patch Manager 사전 조건에 나열된 대로 버전 7.5~7.8이 지원됩니다.	2020년 6월 16일
패치 작업에서 `InstallOverrideList` 파라미터 사용을 위한 샘플 시나리오	새로운 주제인 `InstallOverrideList` 파라미터를 사용하기 위한 샘플 시나리오는 `AWS-RunPatchBaseline` 문서의 `InstallOverrideList` 파라미터를 사용하여 단일 패치 기준을 계속 사용하면서 서로 다른 유형의 유지 관리 기간 일정에 따라 대상 그룹에 서로 다른 유형의 패치를 적용하기 위한 전략을 설명합니다.	2020년 6월 11일
AppConfig에 대한 미리 정의된 배포 전략	AppConfig는 이제 미리 정의된 배포 전략을 제공합니다. 자세한 내용은 배포 전략 생성을 참조하세요.	2020년 6월 10일

Patch Manager가 이제 Red Hat Enterprise Linux(RHEL) 7.8~8.2를 지원합니다.	이제 Patch Manager를 사용하여 RHEL 7.8~8.2 인스턴스에 패치를 적용할 수 있습니다. 자세한 내용은 다음 주제를 참조하세요. • 보안 패치 선택 방법 • 패치 설치 방법 • RHEL에서 패치 기준 규칙 작동 방법 • Red Hat Enterprise Linux 인스턴스에 SSM Agent 수동 설치 • 하이브리드 Linux 노드에 SSM Agent를 설치하는 방법	2020년 6월 9일
Explorer는 위임된 관리자 지원	AWS Organizations에서 리소스 데이터 동기화를 사용하여 여러 AWS 리전 및 AWS 계정에서 Explorer 데이터를 집계하는 경우 Explorer에 대한 위임된 관리자를 구성하는 것이 좋습니다. 위임된 관리자는 다중 계정 및 리전 리소스 데이터 동기화를 생성하거나 삭제할 수 있는 Explorer 관리자 수를 한 명으로 제한하여 Explorer 보안을 강화합니다. 또한 Explorer에서 리소스 데이터 동기화를 관리하기 위해 더 이상 AWS Organizations 관리 계정에 로그인할 필요가 없습니다. 자세한 내용은 위임된 관리자 구성을 참조하세요.	2020년 6월 3일

지정된 다음 Cron 간격에서만 State Manager 연결 적용	State Manager 연결을 생성한 직후에 연결을 실행하지 않으려면 Systems Manager 콘솔에서 지정된 다음 Cron 간격에서만 연결 적용(Apply association only at the next specified Cron interval) 옵션을 선택합니다. 자세한 내용은 연결 생성을 참조하세요.	2020년 6월 3일
Explorer의 새 데이터 원본: AWS Compute Optimizer	Explorer는 이제 AWS Compute Optimizer의 데이터를 표시합니다. 여기에는 미달 프로비저닝되거나 과다 프로비저닝된 EC2 인스턴스 수, 최적화 결과, 온디맨드 요금 내역, 인스턴스 유형 및 요금에 대한 권장 사항이 포함됩니다. 자세한 내용은 관련 서비스 설정의 AWS Compute Optimizer 설정에 대한 세부 정보를 참조하세요.	2020년 5월 26일

새로운 장: Systems Manager 리소스에 태그 지정	새로운 장인 Systems Manager 리소스 태그 지정에서는 Systems Manager의 태그 지정 가능한 리소스 유형 6가지와 함께 태그를 사용하는 방법에 대해 간략하게 설명합니다. 또한 이 장에서는 이러한 리소스 유형에서 태그를 추가하고 제거하는 방법에 대한 포괄적인 지침을 제공합니다. • 문서 • 유지 관리 기간 • 관리형 인스턴스 • OpsItems • 파라미터 • 패치 기준	2020년 5월 25일
Patch Manager를 사용하여 Windows 서비스 팩 및 Linux 부 버전 업그레이드 설치	새로운 주제인 자습서: Windows 서비스 팩 설치용 패치 기준선 생성(콘솔)에서는 Windows 서비스 팩 설치에만 사용되는 패치 기준선을 생성할 수 있는 방법을 보여줍니다. 사용자 정의 패치 기준 만들기(Linux) 주제는 패치 기준에 Linux 운영 체제의 부 버전 업그레이드를 포함하는 것에 대한 정보로 업데이트되었습니다.	2020년 5월 21일

Parameter Store 장 재구성	Parameter Store 작업의 옵션 구성 또는 설정에 대한 모든 주제는 Parameter Store 설정하기 섹션에 통합되었습니다. 여기에는 이 장의 다른 부분에서 다룬 파라미터 티어 관리 및 Parameter Store 처리량 증가 주제가 포함됩니다.	2020년 5월 18일
Systems Manager API 작업과 상호 작용하기 위한 날짜 및 시간 문자열을 만드는 새로운 주제입니다.	새로운 주제인 Systems Manager용 형식이 지정된 날짜 및 시간 문자열 생성에서는 Systems Manager API 작업과 상호 작용하기 위한 형식이 지정된 날짜 및 시간 문자열을 생성하는 방법을 설명합니다.	2020년 5월 13일
SecureString 파라미터의 암호화 권한에 대한 정보	새로운 주제인 IAM 정책을 사용하여 Systems Manager 파라미터에 대한 액세스 제한에서 AWS KMS key를 사용한 `SecureString` 파라미터 암호화와 AWS에서 제공한 AWS 관리형 키 사용의 차이점을 설명합니다.	2020년 5월 13일

Patch Manager에서 이제 Debian Server 및 Oracle Linux 7.6 운영 체제 지원	이제 Patch Manager를 사용하여 Debian Server 및 Oracle Linux 인스턴스를 패치할 수 있습니다. Patch Manager는 Debian Server 8.x, 9.x 및 Oracle Linux 7.6 버전 패치를 지원합니다. 자세한 정보는 다음 주제를 참조하세요. • 보안 패치 선택 방법 • 패치 설치 방법 • Debian Server에서 패치 기준 규칙 작동 방법 • Oracle Linux에서 패치 기준 규칙 작동 방법	2020년 5월 7일
AWS Resource Groups를 대상으로 하는 State Manager 연결 생성	태그, 개별 인스턴스 및 AWS 계정의 모든 인스턴스를 대상으로 지정하는 것 외에도, 이제 AWS Resource Groups의 인스턴스를 대상으로 하는 State Manager 연결을 생성할 수 있습니다. 자세한 내용은 State Manager 연결에서의 대상 및 속도 제어 정보를 참조하세요.	2020년 5월 7일

AMI ID 검증을 위한 Parameter Store의 새로운 aws:ec2:image 데이터 형식	String 파라미터를 생성할 때 이제 데이터 형식을 aws:ec2:image 로 지정하여 입력한 파라미터 값이 유효한 Amazon Machine Image(AMI) ID 형식인지 확인할 수 있습니다. AMI ID 형식을 지원하므로 프로세스에서 사용하려는 AMI가 변경될 때마다 모든 스크립트 및 템플릿을 새 ID로 업데이트하지 않아도 됩니다. aws:ec2:image 데이터 형식을 사용하여 파라미터를 생성하고 해당 값에 AMI의 ID를 입력할 수 있습니다. 이것은 새 인스턴스를 생성하려는 AMI입니다. 그런 다음 템플릿 및 명령에서 이 파라미터를 참조합니다. 다른 AMI를 사용할 준비가 되면 파라미터 값을 업데이트합니다. Parameter Store에서 새 AMI ID를 검증하므로 스크립트 및 템플릿을 업데이트할 필요가 없습니다. 자세한 내용은 Amazon Machine Image ID에 대한 기본 파라미터 지원을 참조하세요.	2020년 5월 5일

Run Command 명령으로 종료 코드 관리	Run Command를 사용하면 스크립트에서 종료 코드를 처리하는 방법을 정의할 수 있습니다. 기본적으로 스크립트에서 실행된 마지막 명령의 종료 코드는 전체 스크립트의 종료 코드로 보고됩니다. 그러나 다음 접근 방식을 사용하면 마지막 명령 이전에 명령이 실패하는 경우 셸 조건문을 포함하여 스크립트를 종료할 수 있습니다. 예를 들어 새로운 주제인 Run Command 명령으로 종료 코드 관리를 참조하세요.	2020년 5월 5일
가용 영역 및 로컬 영역에 대한 새로운 퍼블릭 파라미터 출시	AWS 가용 영역 및 로컬 영역에 대한 정보를 프로그래밍 방식으로 사용할 수 있도록 하기 위한 퍼블릭 파라미터가 릴리스되었습니다. 이는 AWS 서비스 및 AWS 리전에 대한 기존 글로벌 인프라 퍼블릭 파라미터에도 추가됩니다. 자세한 내용은 AWS 서비스, 리전, 엔드포인트, 가용 영역, 로컬 영역 및 Wavelength Zone에 대한 퍼블릭 파라미터 호출을 참조하세요.	2020년 5월 4일

Explorer의 새 데이터 원본: AWS Trusted Advisor	Explorer는 이제 AWS Trusted Advisor의 데이터를 표시합니다. 이러한 데이터에는 비용 최적화, 보안, 내결함성, 성능 및 서비스 한도에 대한 권장 사항 및 모범 사례 점검 항목의 상태가 포함됩니다. 자세한 내용은 관련 서비스 설정의 Trusted Advisor 설정에 대한 세부 정보를 참조하세요.	2020년 5월 4일

Chef 레시피를 실행하는 State Manager 연결 생성	`AWS-ApplyChefRecipes` 문서를 사용하여 Chef 쿡북 및 레시피를 실행하는 State Manager 연결을 생성할 수 있습니다. 이 문서는 Chef 레시피 실행을 위해 다음과 같은 이점을 제공합니다. • 여러 Chef 릴리스(Chef 11~Chef 14)를 지원합니다. • 대상 인스턴스에 Chef 클라이언트 소프트웨어를 자동으로 설치합니다. • 선택적으로 대상 인스턴스에 대해 Systems Manager 규정 준수 점검을 실행하고 규정 준수 점검 결과를 S3 버킷에 저장합니다. • 문서를 한 번 실행할 때 여러 쿡북과 레시피를 실행합니다. • 선택적으로 `why-run` 모드에서 레시피를 실행하여 변경하지 않고 대상 인스턴스에서 변경할 레시피를 표시합니다. • 선택적으로 사용자 지정 JSON 속성을 `chef-client` 실행에 적용합니다. 자세한 내용은 Chef 레시피를 실행하는 연결 생성을 참조하세요.	2020년 3월 19일

여러 AWS 계정의 인벤토리 데이터를 중앙 Amazon S3 버킷으로 동기화	여러 AWS 계정의 Systems Manager Inventory 데이터를 중앙 S3 버킷으로 동기화할 수 있습니다. AWS Organizations에서 계정을 정의해야 합니다. 자세한 내용은 AWS Organizations에 정의된 여러 계정에 대한 Inventory 리소스 데이터 동기화 생성을 참조하세요.	2020년 3월 16일
Amazon S3에 AppConfig 구성 저장	이전에는 AppConfig에서 Systems Manager(SSM) 문서 또는 Parameter Store 파라미터에 저장된 애플리케이션 구성만 지원했습니다. 이제 AppConfig에서는 이러한 옵션 외에도 Amazon S3에 구성 저장을 지원합니다. 자세한 내용은 Amazon S3에 저장된 구성 정보를 참조하세요.	2020년 3월 13일
Amazon ECS에 최적화된 AMIs에 기본적으로 SSM Agent 설치	이제 Amazon ECS 최적화 AMIs에 SSM Agent가 기본적으로 설치됩니다. 자세한 내용은 SSM Agent 작업을 참조하세요.	2020년 2월 25일
콘솔에서 AppConfig 구성 생성	이제 AppConfig에서 구성 프로파일을 생성할 때 콘솔에서 애플리케이션 구성을 생성할 수 있습니다. 자세한 내용은 구성 및 구성 프로필 생성을 참조하세요.	2020년 2월 13일

지정된 날짜까지 릴리스된 패치만 자동 승인	Patch Manager는 릴리스 후 지정된 일 수 동안 설치를 위해 자동으로 패치를 승인하는 옵션 외에도, 이제 지정한 날짜 또는 그 이전에 릴리스된 패치만 자동 승인하는 기능을 지원합니다. 예를 들어, 2020년 7월 7일을 패치 기준의 마감 날짜로 지정하면 2020년 7월 8일 당일 또는 이후에 릴리스된 패치가 자동으로 설치되지 않습니다. 자세한 내용은 사용자 지정 기준 정보 및 사용자 지정 패치 기준으로 작업(콘솔)을 참조하세요.	2020년 2월 12일

유지 관리 기간 작업에서 {{RESOURCE_ID}} 의사 파라미터 사용	유지 관리 기간 작업을 등록할 때 작업 유형에 고유한 파라미터를 지정합니다. {{TARGET_ID}} , {{TARGET_TYPE}} 및 {{WINDOW_TARGET_ID}} 와 같은 의사 파라미터 구문을 사용하여 특정 값을 참조할 수 있습니다. 유지 관리 기간 작업이 실행되면 의사 파라미터 자리표시자 대신 올바른 값을 전달합니다. 리소스 그룹의 일부인 리소스를 대상으로 지원하려면 {{RESOURCE_ID}} 가상 파라미터를 사용하여 DynamoDB 테이블, S3 버킷 및 기타 지원되는 유형과 같은 리소스 값을 전달할 수 있습니다. 자세한 내용은 자습서: 유지 관리 기간 생성 및 구성 (AWS CLI)의 다음 주제를 참조하세요. • 유지 관리 기간 작업 등록 시 의사 파라미터 사용 • 예제: 유지 관리 기간에 작업 등록	2020년 2월 6일

신속하게 명령 다시 실행	Systems Manager에는 AWS Systems Manager 콘솔의 Run Command 페이지에서 명령을 다시 실행하는 2가지 옵션이 있습니다. 다시 실행(Rerun): 이 버튼을 사용하면 명령을 변경하지 않고 동일한 명령을 실행할 수 있습니다. 새로 복사(Copy to new): 이 버튼은 한 명령의 설정을 새 명령으로 복사하고 실행하기 전에 해당 설정을 편집할 수 있는 옵션을 제공합니다. 자세한 내용은 명령 재실행을 참조하세요.	2020년 2월 5일
고급 인스턴스 티어에서 표준 인스턴스 티어로 되돌리기	이전에 하이브리드 환경에서 실행 중인 모든 온프레미스 인스턴스가 고급 인스턴스 티어를 사용하도록 구성한 경우에는 표준 인스턴스 티어를 사용하도록 해당 인스턴스를 신속하게 구성할 수 있습니다. 표준 인스턴스 티어로 되돌리기는 AWS 계정 및 단일 AWS 리전의 모든 하이브리드 인스턴스에 적용됩니다. 표준 인스턴스 티어로 되돌리기는 일부 Systems Manager 기능의 가용성에 영향을 미칩니다. 자세한 내용은 고급 인스턴스 티어에서 표준 인스턴스 티어로 되돌리기를 참조하세요.	2020년 1월 16일

패치 설치 후 인스턴스 재부팅을 건너뛰는 새로운 옵션	이전에는 Patch Manager가 패치를 설치하고 난 후에 관리형 인스턴스가 항상 재부팅되었습니다. SSM 문서 `AWS-RunPatchBaseline` 의 새 `RebootOption` 파라미터를 사용하면 새 패치가 설치된 이후에 인스턴스를 자동으로 재부팅할지 여부를 지정할 수 있습니다. 자세한 내용은 SSM 문서 `AWS-RunPatchBaseline` 정보 주제의 파라미터 이름: RebootOption을 참조하세요.	2020년 1월 15일
새로운 주제: 'Linux 인스턴스에서 PowerShell 스크립트 실행'	Run Command를 사용하여 Linux 인스턴스에서 PowerShell 스크립트를 실행하는 방법을 설명하는 새로운 주제입니다. 자세한 내용은 Linux 인스턴스에서 PowerShell 스크립트 실행을 참조하세요.	2020년 1월 10일
'프록시를 사용하도록 SSM Agent 구성' 업데이트	프록시를 사용하도록 SSM Agent을 구성할 때 지정하는 값이 HTTP 프록시 서버와 HTTPS 프록시 서버 모두에 대한 옵션을 반영하도록 업데이트되었습니다. 자세한 내용은 프록시를 사용하도록 SSM Agent 구성을 참조하세요.	2020년 1월 9일

새로운 "보안" 장에서는 Systems Manager 리소스 보안을 위한 간략한 실습 제공	AWS Systems Manager 사용 설명서의 새로운 보안 장은 Systems Manager를 사용할 때 공동 책임 모델을 적용하는 방법을 이해하는 데 도움이 됩니다. 이 장의 주제에서는 보안 및 규정 준수 목표를 충족하도록 Systems Manager를 구성하는 방법을 보여줍니다. 또한 Systems Manager 리소스를 모니터링하고 보호하는 데 도움이 되는 다른 AWS 서비스을(를) 사용하는 방법을 알아봅니다. **Note** 이번 업데이트에서는 사용 설명서의 '인증 및 액세스 제어' 장이 더 새롭고 간단하게 AWS Systems Manager 에 대한 Identity and Access Management 섹션으로 대체되었습니다.	2019년 12월 24일

새로운 샘플 사용자 정의 Automation 실행서	사용자 정의 Automation 실행서 샘플 집합이 사용 설명서에 추가되었습니다. 이러한 샘플은 다양한 Automation 작업을 사용하여 배포, 문제 해결 및 유지 관리 태스크를 단순화하는 방법을 보여주며 자체 사용자 정의 Automation 실행서를 작성하는 데 도움이 됩니다. 자세한 내용은 사용자 정의 Automation 실행서 샘플을 참조하세요. Systems Manager 콘솔에서 Amazon 관리형 Automation 실행서 콘텐츠를 볼 수도 있습니다. 자세한 내용은 Systems Manager Automation 실행서 참조를 참조하세요.	2019년 12월 23일
Oracle Linux 지원	Systems Manager에서 이제 Oracle Linux 7.5 및 7.7을 지원합니다. Oracle Linux 인스턴스용 EC2 인스턴스에 SSM Agent를 수동으로 설치하는 방법에 대한 자세한 내용은 Oracle Linux를 참조하세요. 하이브리드 환경에서 Oracle Linux 서버에 SSM Agent를 설치하는 방법에 대한 자세한 내용은 하이브리드 Linux 노드에 SSM Agent를 설치하는 방법을 참조하세요.	2019년 12월 19일

Amazon EC2 콘솔에서 Session Manager 세션 시작	이제 Amazon Elastic Compute Cloud(Amazon EC2) 콘솔에서 Session Manager 세션을 시작할 수 있습니다. Amazon EC2 콘솔에서 세션 관련 태스크를 수행하려면 사용자와 관리자 모두에게 서로 다른 IAM 권한이 필요합니다. Session Manager 콘솔 및 AWS CLI만 사용하거나, Amazon EC2 콘솔만 사용하거나, 3가지 도구를 모두 사용하기 위한 권한을 제공할 수 있습니다. 자세한 내용은 다음 항목을 참조하십시오. • Session Manager에 대한 빠른 시작 기본 IAM 정책 • 세션 시작(Amazon EC2 콘솔)	2019년 12월 18일
CloudWatch는 Run Command 지표 및 경보 지원	AWS Systems Manager에서 이제 Run Command 명령 상태에 대한 지표를 CloudWatch에 게시하여 해당 지표를 기반으로 경보를 설정할 수 있습니다. 지표를 추적할 수 있는 명령의 터미널 상태 값에는 `Success`, `Failed` 및 `Delivery Timed Out`이 있습니다. 자세한 내용은 Amazon CloudWatch를 사용하여 Run Command 지표 모니터링을 참조하세요.	2019년 12월 17일

새로운 Systems Manager 기능: Change Calendar	Systems Manager Change Calendar를 사용하여 리소스에 대한 코드 변경(예: Systems Manager Automation 실행서 또는 AWS Lambda 함수에서)을 제한하거나 방지하려는 기간(이벤트)을 지정할 수 있습니다. 변경 일정은 iCalendar 2.0 데이터를 일반 텍스트 형식으로 저장하는 새로운 Systems Manager 문서 유형입니다. 자세한 내용은 AWS Systems Manager Change Calendar를 참조하세요.	2019년 12월 11일

새로운 Systems Manager 기능: AWSAppConfig	AppConfig을 사용하여 애플리케이션 구성을 생성, 관리 및 신속하게 배포합니다. AppConfig는 모든 규모의 애플리케이션에 대한 제어된 배포를 지원합니다. EC2 인스턴스, AWS Lambda, 컨테이너, 모바일 애플리케이션 또는 IoT 디바이스에서 호스팅되는 애플리케이션에서 AppConfig를 사용할 수 있습니다. 애플리케이션 구성을 배포할 때 오류를 방지하기 위해 AppConfig에는 유효성 검사기가 포함됩니다. 유효성 검사기는 구문 또는 의미 검사를 제공하여 배포하려는 구성이 의도 한대로 작동하는지 확인합니다. 구성 배포 중에 AppConfig은 애플리케이션을 모니터링하여 배포가 완료되었는지 확인합니다. 시스템에 오류가 발생하거나 배포가 경보를 시작하면 애플리케이션 사용자에 대한 영향을 최소화하기 위해 AppConfig이 변경 사항을 롤백합니다. 자세한 내용은 AWSAppConfig 단원을 참조하십시오.	2019년 11월 25일

새로운 Systems Manager 기능: Systems Manager Explorer	AWS Systems Manager Explorer는 AWS 리소스에 대한 정보를 보고하는 사용자 지정 가능한 운영 대시보드입니다. Explorer에는 AWS 계정 및 AWS 리전의 운영 데이터(OpsData)에 대한 집계 보기가 표시됩니다. Explorer에서 OpsData에는 EC2 인스턴스에 대한 메타데이터, 패치 규정 준수 세부 정보 및 운영 작업 항목(OpsItems)이 포함됩니다. Explorer은 사업부 또는 애플리케이션에 OpsItems이 분배되는 되는 방식, 시간 경과에 따른 추세 및 범주별 차이점에 대한 컨텍스트를 제공합니다. Explorer에서 정보를 그룹화 및 필터링하여 사용자와 관련이 있고 작업이 필요한 항목에 초점을 맞출 수 있습니다. 우선 순위가 높은 문제를 식별하면 Systems Manager OpsCenter를 사용하여 Automation 실행 서를 실행하고 이러한 문제를 신속하게 해결할 수 있습니다. 자세한 정보는 AWS Systems ManagerExplorer 섹션을 참조하세요. **Note** Systems Manager OpsCenter 설정이 Explorer 설정에 통합되어 있습니다. 이미 OpsCenter를 설정한	2019년 11월 18일

	경우, 통합 설정을 완료하여 설정 및 옵션을 확인해야 합니다. OpsCenter를 설정하지 않은 경우에는 통합 설정을 사용하여 두 기능을 모두 시작할 수 있습니다. 자세한 내용은 Explorer 및 OpsCenter 시작하기를 참조하세요.	
파라미터 검색 기능 향상	이제 파라미터를 검색하는 도구를 사용하면 계정에 파라미터가 많거나 파라미터의 정확한 이름을 기억하지 못하는 경우에 쉽게 파라미터를 찾을 수 있습니다. 검색 도구를 사용하여 `contains`로 필터링합니다. 이전에는 검색 도구에서 `equals` 및 `begins-with` 에 따라서만 파라미터 이름을 검색할 수 있도록 지원했습니다. 자세한 내용은 Systems Manager 파라미터 검색을 참조하세요.	2019년 11월 15일

Automation을 위한 새로운 콘솔 기반 문서 빌더 | Automation 단계에서 스크립트 실행 지원	이제 Systems Manager Automation을 사용하여 표준화된 운영 플레이북을 빌드 및 공유함으로써 사용자, AWS 계정 및 AWS 리전 전반에서 일관성을 유지할 수 있습니다. Markdown을 사용하여 스크립트를 실행하고 Automation 실행서에 인라인 문서를 추가하는 기능을 사용하면 오류를 줄이고 Wiki에서 작성된 절차를 탐색하거나 터미널 명령을 실행하는 등의 수동 단계를 제거할 수 있습니다. 자세한 내용은 다음 항목을 참조하십시오. • 시연: 문서 빌더를 사용하여 사용자 지정 Automation 런북 생성 • `aws:executeScript` (Automation 작업 참조) • 문서 빌더를 사용하여 Automation 런북 생성 • AWS News Blog의 New Automation Features In Systems Manager	2019년 11월 14일

을 사용하여 인플레이스 패키지 업데이트 수행Distributor	이전에는 Distributor을 사용하여 패키지에 업데이트를 설치하려는 경우 전체 패키지를 제거하고 새 버전을 다시 설치하는 것이 유일한 방법이었습니다. 이제는 대신해 인플레이스 업데이트를 수행할 수 있습니다. 인플레이스 업데이트 중에는 Distributor가 패키지에 포함된 업데이트 스크립트에 따라 마지막 설치 이후 새 파일이나 변경된 파일만 설치합니다. 이 옵션을 사용하면 업데이트 중에 패키지 패키지 업데이트를 계속 사용할 수 있고 오프라인으로 전환할 수 없습니다. 자세한 내용은 다음 항목을 참조하십시오. • 패키지 생성 • 패키지 설치 또는 업데이트	2019년 11월 11일
새로운 SSM Agent 자동 업데이트 기능	클릭 한 번으로 새 버전의 SSM Agent를 자동으로 확인하고 다운로드하도록 AWS 계정의 모든 인스턴스를 구성할 수 있습니다. 이렇게 하려면 AWS Systems Manager 콘솔의 관리형 인스턴스 페이지에서 에이전트 자동 업데이트(Agent auto update)를 선택합니다. 자세한 내용은 SSM Agent에 대한 업데이트 자동화를 참조하세요.	2019년 11월 5일

AWS 제공 태그를 사용하여 Session Manager 액세스 제한	이제 세션 작업에 대한 사용자 액세스를 제어하는 두 번째 방법을 사용할 수 있습니다. 이 새로운 방법을 사용하면 `{aws:username}` 변수를 사용하는 대신 AWS 제공 세션 태그를 사용하여 IAM 액세스 정책을 생성할 수 있습니다. 이러한 AWS 제공 세션 태그를 사용하면 조직이 페더레이션 ID를 사용하여 세션에 대한 사용자 액세스를 제어할 수 있게 됩니다. 자세한 내용은 사용자가 자신이 시작한 세션만 종료하도록 허용 섹션을 참조하세요.	2019년 10월 2일

Ansible 플레이북을 적용하기 위한 새로운 SSM Command 문서	`AWS-ApplyAnsiblePlaybooks` 문서를 사용하여 Ansible 플레이북을 실행하는 State Manager 연결을 생성할 수 있습니다. 이 문서는 플레이북 실행을 위해 다음과 같은 이점을 제공합니다. • 복잡한 플레이북 실행 지원 • GitHub 및 Amazon Simple Storage Service(S3)에서 플레이북 다운로드 지원 • 압축된 플레이북 구조 지원 • 향상된 로깅 • 플레이북이 번들로 제공될 때 실행할 플레이북 지정 가능 자세한 내용은 Ansible 플레이북을 실행하는 연결 생성을 참조하세요.	2019년 9월 24일

Session Manager에 대한 포트 전달 지원	이제 Session Manager은 포트 전달 세션을 지원합니다. 포트 전달을 사용하면, 서버에서 SSH 서비스를 시작하거나 보안 그룹에서 SSH 포트를 열거나 Bastion Host를 사용할 필요 없이, 프라이빗 서브넷에 배포된 인스턴스 간에 터널을 안전하게 생성할 수 있습니다. SSH 터널과 비슷한 포트 전달을 사용하면 랩톱 간 트래픽을 인스턴스의 개방 포트에 전달할 수 있습니다. 포트 전달이 구성되면 로컬 포트에 연결하고 인스턴스 내부에서 실행 중인 서버 애플리케이션에 액세스할 수 있습니다. 자세한 내용은 다음 주제를 참조하세요. • AWS News Blog의 Port Forwarding Using AWS Systems ManagerSession Manager • 세션 시작(포트 전달)	2019년 8월 29일

기본 파라미터 티어 지정 또는 티어 선택 자동화	이제 요청에 사용할 기본 파라미터 티어를 지정하여 티어를 지정하지 않는 파라미터를 생성하거나 업데이트할 수 있습니다. 기본 티어를 표준 파라미터, 고급 파라미터 또는 새 옵션인 지능형 계층화에 설정할 수 있습니다. 지능형 계층화는 각 PutParameter 요청을 평가하고 필요할 때에만 고급 파라미터를 생성합니다. (고급 파라미터는 파라미터 값의 크기가 4KB 이상이고 파라미터 정책이 파라미터와 연결되어 있거나 표준 티어에 지원되는 최대 10,000개 파라미터가 이미 생성된 경우 필요합니다.) 기본 티어 지정과 지능형 계층화 사용에 대한 자세한 내용은 기본 파라미터 티어 지정을 참조하세요.	2019년 8월 27일
CLI 및 PowerShell 절차로 업데이트된 연결 작업 섹션	연결 작업 섹션은 AWS CLI 또는 AWS Tools for PowerShell을 사용하여 연결을 관리하기 위한 절차 문서를 포함하도록 업데이트되었습니다. 자세한 내용은 Systems Manager에서 연결 작업을 참조하세요.	2019년 8월 26일

CLI 및 PowerShell 절차로 업데이트된 Automation 실행 작업 섹션	자동화 실행 작업 섹션은 AWS CLI 또는 AWS Tools for PowerShell을 사용하여 자동화 워크플로를 실행하기 위한 절차 문서를 포함하도록 업데이트되었습니다. 자세한 내용은 Automation 실행 작업을 참조하세요.	2019년 8월 20일
애플리케이션 인사이트와 OpsCenter 통합	OpsCenter는 .NET 및 SQL Server용 Amazon CloudWatch Application Insights와 통합됩니다. 즉 애플리케이션에서 감지된 문제에 대한 OpsItems를 자동으로 생성할 수 있습니다. OpsItems를 생성하도록 Application Insights를 구성하는 방법에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 모니터링을 위해 애플리케이션 설정, 구성, 관리를 참조하세요.	2019년 8월 7일

새로운 콘솔 기능: AWS Systems Manager Quick Setup	Quick Setup은 EC2 인스턴스에서 여러 Systems Manager 구성 요소를 빠르게 구성할 수 있는 Systems Manager 콘솔의 새 기능입니다. 특히, 빠른 설정을 사용하면 태그를 사용하여 선택하거나 대상을 지정하는 인스턴스에 다음 구성 요소를 구성할 수 있습니다. • Systems Manager에 대한 AWS Identity and Access Management(IAM) 인스턴스 프로파일 역할. • SSM Agent의 예정된 격월 업데이트. • 30분마다 인벤토리 메타데이터의 예정된 수집. • 누락된 패치를 식별하도록 매일 인스턴스 스캔. • Amazon CloudWatch 에이전트의 일회 설치 및 구성. • CloudWatch 에이전트의 예정된 월간 업데이트. 자세한 내용은 AWS Systems Manager 빠른 설정을 참조하세요.	2019년 8월 7일

리소스 그룹을 유지 관리 기간 대상으로 등록	관리형 인스턴스를 유지 관리 기간의 대상으로 등록하는 것 외에도 이제 리소스 그룹을 유지 관리 기간 대상으로 등록할 수 있습니다. Maintenance Windows는 AWS Resource Groups에서 지원하는 `AWS::EC2::Instance` , `AWS::DynamoDB::Table` , `AWS::OpsWorks::Instance` , `AWS::Redshift::Cluster` 등의 모든 AWS 리소스 유형을 지원합니다. 또한 이번 릴리스에서는 예를 들어 Run Command 콘솔 또는 AWS CLI send-command 명령을 사용해 명령을 리소스 그룹에게 전송할 수도 있습니다. 자세한 정보는 다음 주제를 참조하세요. • 유지 관리 기간에 대상 할당(콘솔) • 예제: 유지 관리 기간에 대상 등록 • 대상 및 비율 제어를 사용하여 플릿에 명령 전송	2019년 7월 23일
AWS Systems ManagerDistributor를 사용한 단순 패키지 생성 및 버전 관리	Distributor에는 패키지 매니페스트, 스크립트 및 파일 해시를 자동으로 생성해주는 단순 패키지 생성 워크플로가 있습니다. 단순 워크플로는 기존 패키지에 버전을 추가할 때도 사용할 수 있습니다.	2019년 7월 22일

Systems Manager Automation을 위한 새로운 문서 범주 창	콘솔에서 Automation을 실행할 때 새로운 문서 범주 창이 Systems Manager에 추가되었습니다. 이 창을 사용하면 목적에 따라 Automation 실행서를 필터링할 수 있습니다.	2019년 7월 18일
사용자가 기본 Session Manager 구성 문서에 액세스할 수 있는 권한 확인	계정에 속한 사용자가 AWS CLI를 사용해 Session Manager 세션을 시작하면서 명령에서 구성 문서를 지정하지 않을 경우 Systems Manager는 기본 구성 문서인 `SSM-SessionManagerRunShell` 을 사용합니다. 이제 `ssm:SessionDocumentAccessCheck` 에 대한 조건 요소를 AWS Identity and Access Management의 정책에 추가하면 이 문서에 대한 액세스 권한이 사용자에게 부여되었는지 확인할 수 있습니다. (IAM) 엔터티(사용자, 그룹 또는 역할). 자세한 내용은 문서 권한 검사를 기본 CLI에 적용하는 시나리오 를 참조하세요.	2019년 7월 9일

운영 체제 사용자 자격 증명을 사용해 Session Manager 세션을 시작할 수 있도록 지원	기본적으로 Session Manager 세션은 시스템을 통해 관리형 인스턴스에서 생성되는 `ssm-user` 계정의 자격 증명을 사용해 시작됩니다. 이제 Linux 시스템에서 운영 체제 계정의 자격 증명을 사용해 세션을 시작할 수 있습니다. 자세한 내용은 Linux 인스턴스에 대한 Run As 지원 설정을 참조하세요.	2019년 7월 9일
SSH로 Session Manager 세션을 시작할 수 있도록 지원	이제 Session Manager에서 AWS CLI를 사용해 SSH 세션을 관리형 인스턴스에서 시작할 수 있습니다. Session Manager에서 SSH 세션 허용에 대한 자세한 내용은 (옵션) SSH Session Manager 세션 설정을 참조하세요. Session Manager를 사용해 SSH 세션을 시작하는 방법에 대한 자세한 내용은 세션 시작(SSH)을 참조하세요.	2019년 7월 9일
관리형 인스턴스의 암호 변경 지원	이제 Systems Manager를 사용해 관리하는 시스템(관리형 인스턴스)의 암호를 재설정할 수 있습니다. 암호 재설정은 Systems Manager 콘솔 또는 AWS CLI를 사용하면 가능합니다. 자세한 내용은 관리형 인스턴스의 암호 재설정을 참조하세요.	2019년 7월 9일

"AWS Systems Manager이란 무엇입니까?"에 대한 개정된 내용	AWS Systems Manager란 무엇인가요?의 소개 내용이 서비스에 대해 더 폭넓은 소개를 제공하고 최근에 릴리스된 Systems Manager 기능을 반영하기 위해 확장되었습니다. 뿐만 아니라, 해당 섹션의 다른 내용이 더 잘 검색될 수 있도록 개별 항목으로 이동되었습니다.	2019년 6월 10일
새로운 Systems Manager 기능: OpsCenter	OpsCenter는 운영 엔지니어 및 IT 전문가가 AWS 리소스와 관련된 운영 작업 항목(OpsItems)을 보고, 조사하고, 해결할 수 있는 중앙 위치를 제공합니다. OpsCenter는 AWS 리소스에 영향을 미치는 문제의 평균 해결 시간을 단축하도록 설계되었습니다. 이 Systems Manager는 각 OpsItem, 관련 OpsItems 및 관련 리소스에 대한 컨텍스트 조사 데이터를 제공하면서 서비스 전반에 걸쳐 OpsItems를 집계하고 표준화합니다. 또한 OpsCenter는 신속하게 문제를 해결하는 데 사용할 수 있는 Systems Manager Automation 실행서를 제공합니다. 각 OpsItem에 대해 검색 가능한 사용자 지정 데이터를 지정할 수 있습니다. 상태 및 소스별로 OpsItems에 대한 자동 생성 요약 보고서를 볼 수도 있습니다. 자세한 내용은 AWS Systems ManagerOpsCenter 섹션을 참조하세요.	2019년 6월 6일

AWS Management Console의 Systems Manager 왼쪽 탐색 창 변경 사항	AWS Management Console의 Systems Manager 왼쪽 탐색 창에는 Ops Center의 새 제목을 비롯한 새로운 제목이 포함되어 있어 Systems Manager 기능의 논리적 그룹화를 제공합니다.	2019년 6월 6일
AWS CLI를 사용한 유지 관리 기간 생성 및 구성에 대한 개정된 자습서	자습서: 유지 관리 기간 생성 및 구성(AWS CLI)은 연습 단계를 통해 간단한 경로를 제공하기 위해 철저히 점검되었습니다. 단일 유지 관리 기간을 만들고 단일 대상을 식별하고 유지 관리 기간을 실행할 간단한 작업을 설정합니다. 또한 {{TARGET_ID}} 과 같은 의사 파라미터 사용에 대한 정보를 포함하여 자체 작업 등록 명령을 생성하는 데 사용할 수 있는 정보와 예제를 제공합니다. 자세한 내용 및 예제는 다음 주제를 참조하세요. • 예제: 유지 관리 기간에 대상 등록 • 예제: 유지 관리 기간에 작업 등록 • register-task-with-maintenance-windows 옵션 정보 • 유지 관리 기간 작업 등록 시 의사 파라미터 사용	2019년 5월 31일

SSM Agent 업데이트 알림	SSM Agent 업데이트에 대해 알림을 수신하려면 GitHub에서 SSM Agent 릴리스 정보 페이지를 구독합니다.	2019년 5월 24일
Parameter Store의 변경 사항을 기반으로 알림 수신 또는 작업 트리거	Parameter Store 이벤트 기반 알림 설정 또는 작업 트리거 주제를 참조하여 Parameter Store의 변경 사항에 응답하는 Amazon EventBridge 규칙을 설정할 수 있습니다. 다음 중 하나가 발생할 때 알림을 받거나 다른 작업을 트리거할 수 있습니다. • 파라미터 생성, 업데이트 또는 삭제 • 파라미터 레이블 버전 생성, 업데이트 또는 삭제 • 파라미터 만료, 만료 예정 또는 지정된 기간 동안 변경되지 않음	2019년 5월 22일

설정 및 시작 콘텐츠에 대한 주요 개정 사항	AWS Systems Manager User Guide의 Setting Up 및 Getting Started의 내용이 확대 및 재구성되었습니다. 설정 콘텐츠는 두 섹션으로 구성됩니다. 한 섹션에서는 EC2 인스턴스를 구성 및 관리하기 위해 Systems Manager를 설정하는 태스크에 중점을 둡니다. 다른 섹션에서는 하이브리드 환경에서 온프레미스 서버 및 가상 머신(VM)을 구성하고 관리하기 위해 Systems Manager를 설정하는 태스크에 중점을 둡니다. 이제 두 섹션 모두 모든 설정 주제를 권장되는 완료 순서에 따라 주요 번호 매기기 단계로 표시합니다. 새로운 시작하기 장에서는 계정 및 서비스 구성 태스크가 완료된 후 최종 사용자가 Systems Manager를 시작하도록 돕는 데 중점을 둡니다. • AWS Systems Manager 설정 • 하이브리드 환경을 위한 AWS Systems Manager 설정 • AWS Systems Manager 시작하기	2019년 5월 15일

패치 기준에 따라 Microsoft에서 릴리스한 애플리케이션에 대한 패치 포함(Windows)	Patch Manager에서 이제 Microsoft에서 릴리스한 Windows Server 인스턴스 기반 애플리케이션에 대한 패치 업데이트를 지원합니다. 이전에는 Windows Server 운영 체제용 패치만 지원되었습니다. Patch Manager는 Windows Server 인스턴스에 대해 미리 정의된 두 가지 패치 기준을 제공합니다. 패치 기준 `AWS-WindowsPredefinedPatchBaseline-OS` 는 운영 체제 패치에만 적용됩니다. `AWS-WindowsPredefinedPatchBaseline-OS-Applications` 는 Microsoft에서 릴리스한 Windows Server 운영 체제 및 Windows 기반 애플리케이션 모두에 적용됩니다. Microsoft 릴리스 애플리케이션 패치를 포함하는 사용자 정의 패치 기준 생성에 대한 자세한 내용은 사용자 정의 패치 기준 생성의 첫 번째 절차를 참조하세요. 또한 이번 업데이트의 일환으로 AWS에서 제공하는 미리 정의된 패치 기준의 이름이 변경됩니다. 자세한 내용은 사전 정의된 기준을 참조하세요.	2019년 5월 7일

AWS CLI를 사용하여 유지 관리 기간 대상을 등록하는 예제	새로운 주제인 예제: 유지 관리 기간에 대상 등록에서는 AWS CLI를 사용할 때 유지 관리 기간에 대한 대상을 각기 다른 방법으로 지정할 수 있는 3개의 샘플 명령을 제공합니다. 이 주제에서는 또한 각 샘플 명령의 모범 사용 사례를 설명합니다.	2019년 5월 3일
패치 그룹 주제 업데이트	업데이트되어 관리형 인스턴스에서 패칭 작업 도중 사용할 적절한 패치 기준을 결정하는 방법에 관한 섹션이 포함되도록 패치 그룹 정보 주제가 업데이트되었습니다. 또한 AWS CLI 또는 Systems Manager 콘솔을 사용하여 관리형 인스턴스에 Patch Group 또는 PatchGroup 태그를 추가하는 방법과 패치 기준선에 Patch Group 또는 PatchGroup을 추가하는 방법에 대한 지침이 추가되었습니다. EC2 인스턴스 메타데이터에 태그를 허용한 경우 **`PatchGroup`** (공백 없음)을 사용해야 합니다. 자세한 내용은 패치 그룹 생성 및 패치 기준에 패치 그룹 추가를 참조하세요.	2019년 5월 1일

새로운 Parameter Store 기능	Parameter Store에서 다음과 같은 새 기능을 제공합니다. • 고급 파라미터: Parameter Store 사용을 통해 표준 파라미터 티어(기본 티어) 또는 고급 파라미터 티어에서 사용하도록 파라미터를 개별적으로 구성할 수 있습니다. 고급 파라미터는 파라미터 값에 대한 크기 제한 증가, AWS 계정 및 AWS 리전별로 생성할 수 있는 파라미터 수 제한 증가 및 파라미터 정책 사용 기능을 제공합니다. 고급 파라미터에 대한 자세한 내용은 Systems Manager 고급 파라미터 정보를 참조하세요. • 파라미터 정책: 파라미터 정책은 만료 날짜 또는 TTL(Time to Live)과 같이 파라미터에 대한 특정 기준을 지정할 수 있도록 허용함으로써 증가하는 파라미터 집합을 관리하는 데 도움이 됩니다. 파라미터 정책은 특히 Parameter Store에 저장된 암호 및 구성 데이터를 강제로 업데이트 또는 삭제하도록 하는 데 유용합니다. 파라미터 정책은 고급 파라미터 티어를 사용하는 파라미터에 대해서만 이용 가능합니다. 자세한 내용은 파라미터 정책 작업을 참조하세요.	2019년 4월 25일

	• 처리량 증가: Parameter Store 처리량 제한을 최대 초당 1,000개의 트랜잭션으로 증가시킬 수 있습니다. 자세한 내용은 Parameter Store 처리량 증가를 참조하세요.	
Automation 섹션 업데이트	자동화 섹션이 업데이트되어 검색 가능성이 개선되었습니다. 추가로 세 개의 새로운 주제가 Automation 단원에 추가되었습니다. • 수동으로 자동화 실행 • 승인자를 사용하여 자동화 실행 • 자동화 예약	2019년 4월 17일

AWS KMS 키를 사용한 세션 데이터 암호화	기본적으로 Session Manager에서 TLS 1.2를 사용하여 계정 내 사용자의 로컬 머신과 EC2 인스턴스 사이에 전송되는 세션 데이터를 암호화합니다. 이제 AWS Key Management Service에서 생성된 AWS KMS key를 사용하여 데이터를 암호화할 수 있습니다. AWS 계정에 생성된 KMS 키를 사용하거나 다른 계정에서 공유되는 키를 사용할 수 있습니다. 세션 데이터 암호화를 위한 KMS 지정에 대한 자세한 내용은 세션 데이터의 AWS KMS 키 암호화 설정(콘솔), Session Manager 기본 설정 생성(AWS CLI) 또는 Session Manager 기본 설정 업데이트(AWS CLI)를 참조하세요.	2019년 4월 4일
AWS Systems Manager에 대한 Amazon SNS 알림 구성	유지 관리 기간에 등록된 Run Command 및 Run Command 태스크에 대해 Amazon SNS 알림을 구성하기 위한 AWS CLI 또는 Systems Manager 콘솔 사용 지침이 추가되었습니다. 자세한 내용은 AWS Systems Manager에 대한 Amazon SNS 알림 구성을 참조하세요.	2019년 3월 6일

하이브리드 환경의 서버 및 VM에 대한 고급 인스턴스	AWS Systems Manager에서는 하이브리드 환경의 서버 및 VM에 대해 표준 인스턴스 티어 및 고급 인스턴스 티어를 제공합니다. 표준 인스턴스 티어를 통해 AWS 리전의 AWS 계정당 최대 1,000개의 서버 또는 VM을 등록할 수 있습니다. 단일 계정 및 리전에 1,000개 이상의 서버 또는 VM을 등록해야 하는 경우 고급 인스턴스 티어를 사용하세요. 고급 인스턴스 티어에서 원하는 만큼 인스턴스를 생성할 수 있지만 Systems Manager에 대해 구성된 모든 인스턴스는 종량 과금제로 제공됩니다. 또한 고급 인스턴스에서는 AWS Systems Manager Session Manager 사용을 통해 하이브리드 시스템에 연결할 수 있습니다. Session Manager는 인스턴스에 대한 대화형 셸 액세스를 제공합니다. 고급 인스턴스 허용에 대한 자세한 내용은 고급 인스턴스 티어 사용을 참조하세요.	2019년 3월 4일

공유 SSM 문서를 사용하는 State Manager 연결 생성	다른 AWS 계정에서 공유되는 SSM 명령 및 Automation 실행 서를 사용하는 State Manager 연결을 생성할 수 있습니다. 공유 SSM 문서를 사용함으로써 연결을 생성하면 인스턴스가 동일한 계정에 있는 경우가 아니어도 Amazon EC2 및 하이브리드 인프라를 일관적인 상태로 유지하는 데 도움이 됩니다. SSM 문서에 대한 자세한 내용은 AWS Systems Manager 문서를 참조하세요. State Manager 연결 생성에 대한 자세한 내용은 연결 생성을 참조하세요.	2019년 2월 28일
Amazon EventBridge 규칙에 대해 지원되는 Systems Manager 이벤트 목록 보기	새로운 주제인 Amazon EventBridge를 사용한 Systems Manager 이벤트 모니터링에서는 Systems Manager에서 전송된 다양한 이벤트 요약을 제공하며, 이에 대해 EventBridge에서 이벤트 모니터링 규칙을 생성할 수 있습니다.	2019년 2월 25일

Systems Manager 리소스를 생성할 때 태그 추가	Systems Manager에서는 특정 리소스 유형 생성 시 태그를 추가할 수 있는 기능을 지원합니다. AWS CLI 또는 SDK를 사용하여 생성할 때 태그를 지정할 수 있는 리소스에는 유지 관리 기간, 패치 기준, Parameter Store 파라미터 및 SSM 문서 등이 있습니다. 또한 관리형 인스턴스에 대한 활성화 생성 시 관리형 인스턴스에 태그를 지정할 수도 있습니다. Systems Manager 콘솔을 사용할 때 유지 관리 기간, 패치 기준 및 파라미터에 태그를 추가할 수 있습니다.	2019년 2월 24일

Systems Manager Inventory에 대한 자동 IAM 역할 생성	이전에는 AWS Identity and Access Management(IAM) 역할을 생성하고 이 역할에 별도의 정책을 연결해야 콘솔의 인벤토리 세부 정보 보기 페이지에서 인벤토리 데이터를 볼 수 있었습니다. 더 이상 이 역할을 생성하거나 역할에 정책을 연결할 필요가 없습니다. 인벤토리 세부 정보 보기(Inventory Detail View) 페이지에서 원격 데이터 동기화(Remote Data Sync)를 선택하면 Systems Manager에서 자동으로 `Amazon-GlueService PolicyForSSM` 역할을 생성하고 Amazon-GlueService PolicyForSSM-{S3 bucket name} 정책과 AWSGlueServiceRole 정책을 이 역할에 할당합니다. 자세한 내용은 여러 리전 및 계정에서 인벤토리 데이터 쿼리를 참조하세요.	2019년 2월 14일
SSM Agent 업데이트를 위한 Maintenance Windows 시연	새로운 연습 두 개를 Maintenance Windows 설명서에 추가했습니다. 시연에서는 Systems Manager 콘솔 또는 AWS CLI를 사용하여 SSM Agent의 최신 상태를 자동으로 유지하는 유지 관리 기간 생성 방법을 구체적으로 설명합니다. 자세한 내용은 Maintenance Windows 연습을 참조하세요.	2019년 2월 11일

Parameter Store 퍼블릭 파라미터 사용	Parameter Store 퍼블릭 파라미터를 설명하는 짧은 섹션이 추가되었습니다. 자세한 내용은 Systems Manager 퍼블릭 파라미터 사용을 참조하세요.	2019년 1월 31일
AWS CLI를 사용하여 Session Manager 기본 설정 생성	AWS CLI를 사용하여 CloudWatch Logs, S3 버킷 로깅 옵션 및 세션 암호화 설정과 같은 Session Manager 기본 설정을 생성하는 방법에 대한 지침이 추가되었습니다. 자세한 내용은 AWS CLI를 사용하여 Session Manager 기본 설정 생성을 참조하세요.	2019년 1월 22일
State Manager를 사용하여 Systems Manager 자동화 워크플로 실행	AWS Systems Manager State Manager에서는 SSM Automation 실행서를 사용하는 연결 생성을 지원합니다. State Manager의 경우 이전에는 `command` 및 `policy` 문서만 지원했으며, 따라서 관리형 인스턴스를 대상으로 하는 연결만을 생성할 수 있었습니다. SSM Automation 실행서에 대한 지원을 통해 이제 다른 유형의 AWS 리소스를 대상으로 하는 연결을 생성할 수 있습니다. 자세한 내용은 State Manager 사용을 통한 Systems Manager Automation 워크플로 실행을 참조하세요.	2019년 1월 22일

Cron 및 Rate 표현식과 유지 관리 기간 예약 옵션에 대한 참조 업데이트	참조 주제 Systems Manager용 Cron 및 Rate 표현식이 업데이트되었습니다. 새 버전에서는 Cron 및 Rate 표현식을 사용하여 유지 관리 기간 및 State Manager 연결을 예약하는 방법에 대한 추가 예제와 개선된 설명을 제공합니다. 또한 새로운 주제인 Maintenance Windows 예약 및 활성 기간 옵션에서는 유지 관리 기간에 대한 다양한 일정 관련 옵션(예: 시작 날짜, 종료 날짜, 시간대, 일정 빈도)이 서로 어떤 관계를 갖는지 설명합니다.	2018년 12월 6일
SSM Agent 디버그 로깅을 설정합니다.	관리형 인스턴스에서 seelog.xml.template 파일을 편집해 SSM Agent 디버그 로깅을 설정할 수 있습니다. 자세한 내용은 SSM Agent 디버그 로깅 설정을 참조하세요.	2018년 11월 30일
ARM64 프로세서 아키텍처에 대한 지원	AWS Systems Manager에서 이제 Amazon Linux 2, Red Hat Enterprise Linux 7.6 및 Ubuntu Server(18.04 LTS 및 16.04 LTS) 운영 체제의 ARM64 버전을 지원합니다. 자세한 내용은 Amazon Linux 2, RHEL 및 Ubuntu Server 18.04 및 16.04 LTS(Snap 패키지 포함)의 설치 지침을 참조하세요. A1 인스턴스 유형에 대한 자세한 내용은 Amazon EC2 사용 설명서의 범용 인스턴스를 참조하세요.	2018년 11월 26일

AWS Systems ManagerDistributor를 사용하여 패키지 생성 및 배포	AWS Systems Manager Distributor는 자체 소프트웨어를 패키징하거나, AWS에서 제공한 에이전트 소프트웨어 패키지(예: `AmazonCloudWatchAgent`)를 찾아 AWS Systems Manager 관리형 인스턴스에 설치할 수 있습니다. Distributor는 소프트웨어 패키지 등의 리소스를 AWS Systems Manager 관리형 인스턴스에 게시합니다. 패키지를 게시하면 Distributor에서 패키지를 추가할 때 생성한 Systems Manager 문서인 패키지 문서의 특정 버전을 관리형 인스턴스 ID, AWS 계정 ID, 태그 또는 AWS 리전으로 식별하는 관리형 인스턴스에 알립니다. 자세한 내용은 AWS Systems ManagerDistributor 단원을 참조하십시오.	2018년 11월 20일

현재, 하나의 중앙 계정을 통해 여러 AWS 리전 및 AWS 계정 계정에서 AWS Systems Manager 자동화 워크플로를 실행합니다.	하나의 Automation 관리 계정을 통해 여러 AWS 리전 및 AWS 계정 또는 AWS 조직 단위(OU)에서 AWS Systems Manager 자동화 워크플로를 동시에 실행할 수 있습니다. 여러 리전 및 계정이나 OU에서 Automation을 동시에 실행하면 AWS 리소스를 관리하는 데 드는 시간이 절약될 뿐만 아니라 컴퓨팅 환경의 보안도 향상됩니다. 자세한 내용은 여러 AWS 리전 및 AWS 계정에서 Automation 워크플로 실행을 참조하세요.	2018년 11월 19일
여러 AWS 리전 및 AWS 계정에서 인벤토리 데이터 쿼리	Systems Manager Inventory가 Amazon Athena와 통합되어 여러 AWS 리전 및 AWS 계정에서 인벤터리 데이터를 쿼리하는 데 도움이 됩니다. Athena 통합은 리소스 데이터 동기화를 사용하므로 AWS Systems Manager 콘솔의 인벤토리 세부 정보 보기 페이지에서 모든 관리형 인스턴스의 인벤토리 데이터를 볼 수 있습니다. 자세한 내용은 여러 리전 및 계정에서 인벤토리 데이터 쿼리를 참조하세요.	2018년 11월 15일

MOF 파일을 실행하는 State Manager 연결 생성	MOF(Management Object Format) 파일을 실행하여 AWS-ApplyDSCMofs SSM 문서를 사용해 State Manager에서 Windows Server 관리형 인스턴스에 대해 목표 상태를 적용할 수 있습니다. AWS-ApplyDSCMofs 문서에는 2가지 실행 모드가 있습니다. 첫 번째 모드에서는 관리형 인스턴스가 현재, 지정된 MOF 파일에 정의된 목표 상태인지 스캔해 보고하도록 연결을 구성할 수 있습니다. 두 번째 모드에서는 MOF 파일에 정의된 리소스와 그 값을 기반으로 MOF 파일을 실행하고 인스턴스의 구성을 변경할 수 있습니다. AWS-ApplyDSCMofs 문서를 사용하면 로컬 공유인 Amazon Simple Storage Service(Amazon S3) 또는 HTTPS 도메인을 사용하는 안전한 웹 사이트에서 MOF 구성 파일을 다운로드하고 실행할 수 있습니다. 자세한 내용은 MOF 파일을 실행하는 연결 생성을 참조하세요.	2018년 11월 15일

Session Manager 세션에서 관리 액세스 제한	Session Manager 세션은 `ssm-user`라는 기본 루트 또는 관리자 권한을 사용하여 생성된 사용자 계정의 자격 증명을 사용하여 시작됩니다. 이 계정에 대한 관리 제어 제한에 대한 자세한 내용은 주제 ssm-user 계정 관리 권한 설정 또는 해제에서 확인할 수 있습니다.	2018년 11월 13일
Automation 작업 참조의 YAML 예	Automation 작업 참조에는 이제 JSON 샘플이 이미 포함된 각 작업에 대한 YAML 샘플이 포함되어 있습니다.	2018년 10월 31일
연결에 규정 준수 심각도 수준 할당	이제 State Manager 연결에 규정 준수 심각도 수준을 할당할 수 있습니다. 이러한 심각도 수준은 규정 준수 대시보드에서 보고되며 규정 준수 보고서를 필터링하는 데에도 사용됩니다. 할당할 수 있는 심각도 수준에는 심각, 높음, 중간, 낮음 및 지정 안 함이 있습니다. 자세한 내용은 연결 생성(콘솔)을 참조하세요.	2018년 10월 26일
자동화 및 State Manager와 함께 대상 및 속도 제어 사용	대상, 동시성 및 오류 임계값을 사용하여 리소스의 플릿 간에 Automations 및 State Manager 연결의 실행을 제어할 수 있습니다. 자세한 내용은 플릿에서 대상 및 비율 제어를 사용하여 Automation 워크플로 실행 및 State Manager 연결에 대상 및 비율 제어 사용을 참조하세요.	2018년 10월 23일

유지 관리 기간에 대한 활성 시간 범위 및 국제 시간대 지정	또한 이전 또는 이후에 유지 관리 기간이 실행되면 안 되는 날짜(시작 날짜 및 종료 날짜)를 지정하고 유지 관리 기간 일정의 기준이 되는 국제 시간대를 지정할 수 있습니다. 자세한 내용은 유지 관리 기간 생성(콘솔) 및 유지 관리 기간 업데이트(AWS CLI)를 참조하세요.	2018년 10월 9일
S3 버킷에서 패치 기준에 대한 사용자 지정 패치 목록 관리	SSM 명령 문서 `AWS-RunPatchBaseline` 에 새로운 'InstallOverrideList' 파라미터를 사용하여 설치하려는 패치 목록에 대해 https URL 또는 Amazon Simple Storage Service(Amazon S3) 경로 스타일 URL을 지정할 수 있습니다. S3 버킷에서 YAML 형식으로 관리되는 이 패치 설치 목록은 기본 패치 기준으로 지정된 패치를 재정의합니다. 자세한 내용은 파라미터 이름: InstallOverrideList를 참조하세요.	2018년 10월 5일
패치 종속성 설치 여부에 대한 확장된 제어	이전에 거부된 패치 목록의 패치가 다른 패치의 종속성으로 식별된 경우 설치되어 있을 수 있습니다. 이제, 이러한 종속성을 설치하거나 설치되지 않도록 차단할지 결정할 수 있습니다. 자세한 내용은 패치 기준 생성을 참조하세요.	2018년 10월 5일

조건부 분기를 사용하여 동적 Automation 워크플로 생성	`aws:branch` Automation 작업을 통해 한 단계에서 여러 선택 항목을 평가한 다음 평가 결과에 따라 Automation 런북의 다른 단계로 이동하는 동적 Automation 워크플로를 생성할 수 있습니다. 자세한 내용은 런북에서 조건문 사용을 참조하세요.	2018년 9월 26일
AWS CLI를 사용하여 Session Manager 기본 설정 업데이트	CLI를 사용하여 Session Manager 기본 설정(예: CloudWatch Logs 및 S3 버킷 로깅 옵션)을 업데이트하는 지침이 AWS Systems Manager User Guide에 추가되었습니다. 자세한 내용은 AWS CLI를 사용하여 Session Manager 기본 설정 업데이트를 참조하세요.	2018년 9월 25일
Session Manager에 대한 SSM Agent 요구 사항이 업데이트됨	Session Manager에는 이제 SSM Agent 버전 2.3.68.0 이상이 필요합니다. Session Manager 사전 조건에 대한 자세한 내용은 Session Manager 사전 조건 완료를 참조하세요.	2018년 9월 17일

Session Manager를 사용하여 인바운드 포트를 열거나 Bastion Host를 관리하지 않고 인스턴스 관리	이제 AWS Systems Manager의 완전 관리형 기능인 Session Manager를 사용하여 대화형 원클릭 브라우저 기반 셸 또는 AWS CLI를 통해 EC2 인스턴스에 빠르고 안전하게 액세스할 수 있습니다. Session Manager는 인바운드 포트를 열고, Bastion Host를 유지하고, SSH 키를 관리할 필요 없이 보안성과 감사 가능성을 갖춘 인스턴스 관리 기능을 제공합니다. 또한 Session Manager를 통해 인스턴스에 대한 제한된 액세스를 요구하는 회사 정책, 엄격한 보안 관행을 손쉽게 준수하고, 인스턴스 액세스 세부 정보가 포함된 완전히 감사 가능한 로그를 제공하고 동시에 최종 사용자에게 EC2 인스턴스에 대한 간단한 원 클릭 크로스 플랫폼 액세스를 제공합니다. 자세한 내용은 Session Manager에 대해 자세히 알아보기를 참조하세요.	2018년 9월 11일
Systems Manager Automation 워크플로에서 다른 AWS 서비스 호출	Automation 런북의 3가지 새로운 Automation 작업(또는 플러그인)을 사용하여 Automation 워크플로에서 다른 AWS 서비스와(과) 다른 Systems Manager 기능을 호출할 수 있습니다. 자세한 내용은 작업 출력을 입력으로 사용을 참조하세요.	2018년 8월 28일

IAM 정책에 Systems Manager 별 조건 키 사용	정책에 통합할 수 있는 Systems Manager의 IAM 조건 키를 나열하도록 정책에 조건 지정 주제가 업데이트되었습니다. 이러한 키를 사용하여 정책 적용의 조건을 지정할 수 있습니다. 이 주제에는 예제 정책과 기타 관련 주제에 대한 링크도 나와 있습니다.	2018년 8월 18일
인벤토리 유형을 수집하도록 구성된 인스턴스와 그렇지 않은 인스턴스를 파악하기 위해 인벤토리 데이터를 그룹으로 집계	그룹을 만들면 관리형 인스턴스 중 하나 이상의 Inventory 유형을 수집하도록 구성된 인스턴스 수와 그렇지 않은 인스턴스 수를 빠르게 파악할 수 있습니다. 그룹을 통해 하나 이상의 인벤토리 유형과 `exists` 연산자를 사용하는 필터를 지정합니다. 자세한 내용은 인벤토리 데이터 집계를 참조하세요.	2018년 8월 16일
인벤토리 및 구성 규정 준수를 위한 기록 보기 및 변경 사항 추적	이제 관리형 인스턴스에서 수집한 인벤토리의 기록을 보고 변경 사항을 추적할 수 있습니다. 또한 Configuration Compliance에 의해 보고된 State Manager 연결과 Patch Manager 패치에 대한 기록을 보고 변경을 추적할 수도 있습니다. 자세한 내용은 인벤토리 이력 및 변경 사항 추적 보기를 참조하세요.	2018년 8월 9일

Parameter Store가 Secrets Manager와 통합됩니다	이미 Parameter Store 파라미터 참조를 지원하는 다른 AWS 서비스을(를) 사용할 때 Secrets Manager 암호를 검색할 수 있도록 Parameter Store을(를) AWS Secrets Manager와(과) 통합했습니다. 이러한 서비스에는 Amazon EC2, Amazon Elastic Container Service, AWS Lambda, AWS CloudFormation, AWS CodeBuild, AWS CodeDeploy 및 기타 Systems Manager 기능이 포함됩니다. Parameter Store를 통해 Secrets Manager 암호를 참조하면 코드 및 구성 스크립트에서 일관성 있고 안전하게 암호 및 참조 데이터를 호출하고 사용할 수 있는 프로세스가 마련됩니다. 자세한 내용은 Parameter Store 파라미터에서 AWS Secrets Manager 암호 참조를 참조하세요.	2018년 7월 26일
Parameter Store 파라미터에 레이블 연결	파라미터 레이블이란 다양한 버전의 파라미터를 관리하는 데 도움이 되는 사용자 정의 별칭입니다. 파라미터를 수정하면 Systems Manager에서 버전 번호를 하나씩 늘려서 새 버전을 저장합니다. 레이블이 있으면 파라미터 버전이 여러 개일 때 버전의 용도를 기억하기 쉽습니다. 자세한 내용은 파라미터 레이블 지정을 참조하세요.	2018년 7월 26일

동적 자동화 워크플로 생성	기본적으로 Automation 실행 서의 mainSteps 섹션에서 정의하는 단계(또는 작업)는 순차적으로 실행됩니다. 한 작업이 완료된 후에는 mainSteps 섹션에 지정된 다음 작업이 시작됩니다. 이 릴리스에서 이제 조건부 브랜칭을 수행하는 자동화 워크플로를 생성할 수 있습니다. 따라서 조건 변경에 동적으로 응답하여 지정된 단계로 건너뛰는 Automation 워크플로를 생성할 수 있습니다. 자세한 내용은 런북에서 조건문 사용을 참조하세요.	2018년 7월 18일
SSM Agent가 이제 Snap을 사용하여 Ubuntu Server 16.04 AMIs에 사전 설치됩니다.	20180627로 식별되는 Ubuntu Server 16.04 AMIs에서 생성된 인스턴스부터 SSM Agent가 Snap 패키지를 사용하여 사전 설치됩니다. 이전 AMIs에서 생성된 인스턴스에서는 deb 설치 관리자 패키지를 계속 사용해야 합니다. 자세한 정보는 64비트 Ubuntu Server 16.04 인스턴스에서 SSM Agent 설치 정보를 참조하세요.	2018년 7월 7일

SSM Agent에 필요한 최소 S3 권한 검토	새로운 주제인 SSM Agent에 대한 최소 S3 버킷 권한에서는 리소스가 Systems Manager 작업을 수행하기 위해 액세스해야 할 수 있는 Amazon Simple Storage Service(Amazon S3) 버킷에 대한 정보를 제공합니다. 인스턴스 프로파일 또는 VPC 엔드포인트에 대한 S3 버킷 액세스를 Systems Manager 사용에 필요한 최소값으로 제한하려는 경우 사용자 정의 정책에서 이러한 버킷을 지정할 수 있습니다.	2018년 7월 5일
특정 State Manager 연결 ID에 대한 전체 실행 내역 보기	새로운 주제인 연결 내역 보기에서는 특정 연결 ID에 대한 모든 실행을 본 다음 하나 이상의 리소스에 대한 실행 세부 정보를 보는 방법을 설명합니다.	2018년 7월 2일
Patch Manager가 Amazon Linux 2에 대한 지원을 도입합니다.	이제 Patch Manager를 사용하여 패치를 Amazon Linux 2 인스턴스에 적용할 수 있습니다. Patch Manager 운영 체제 지원에 대한 일반 정보는 Patch Manager 사전 조건을 참조하세요. 패치 필터를 정의할 때 Amazon Linux 2에 지원되는 키-값 페어에 대한 자세한 내용은 AWS Systems Manager API Reference의 PatchFilter를 참조하세요.	2018년 26월 6일

Amazon CloudWatch Logs로 명령 출력 전송	새로운 주제 Run Command에 대한 Amazon CloudWatch Logs 구성에서는 CloudWatch Logs로 Run Command 출력을 전송하는 방법을 설명합니다.	2018년 18월 6일
AWS CloudFormation을 사용하여 Inventory의 리소스 데이터 동기화 신속히 생성 또는 삭제	AWS CloudFormation을 사용하여 Systems Manager Inventory의 리소스 데이터 동기화를 생성하거나 삭제할 수 있습니다. AWS CloudFormation을 사용하려면 AWS CloudFormation 템플릿에 AWS::SSM::ResourceDataSync 리소스를 추가합니다. 자세한 내용은 AWS CloudFormation 사용 설명서의 AWS CloudFormation 템플릿 사용을 참조하세요. 또한 Inventory의 리소스 데이터 동기화 구성에 나오는 설명대로 인벤토리의 리소스 데이터 동기화를 수동으로 생성할 수도 있습니다.	2018년 6월 11일

RSS에서 AWS Systems Manager 사용 설명서 업데이트 알림 이용 가능	Systems Manager User Guide의 HTML 버전이 Systems Manager 설명서 업데이트 기록 페이지에 명시된 업데이트의 RSS 피드를 지원합니다. RSS 피드에는 2018년 6월 이후의 업데이트가 포함됩니다. 이전에 발표한 업데이트도 Systems Manager 설명서 업데이트 기록 페이지에서 이용할 수 있습니다. 피드를 구독하려면 상단 메뉴판에서 RSS 버튼을 누릅니다.	2018년 6월 6일
스크립트에서 종료 코드를 지정하여 관리형 인스턴스 재부팅	새로운 주제인 스크립트에서 관리형 인스턴스 재부팅에서는 Systems Manager에서 Run Command로 실행하는 스크립트에서 종료 코드를 지정하여 관리형 인스턴스를 재부팅하는 방법을 설명합니다.	2018년 6월 3일
사용자 정의 인벤토리를 삭제할 때마다 Amazon EventBridge에 이벤트 생성	새로운 주제 EventBridge에서 인벤토리 삭제 작업 보기에서는 사용자 정의 인벤토리를 삭제할 때마다 이벤트를 생성하도록 Amazon EventBridge를 구성하는 방법을 설명합니다.	2018년 6월 1일

2018년 6월 이전 업데이트

다음 표에서는 2018년 6월 이전 AWS Systems Manager 사용 설명서의 각 릴리스에서 변경된 중요 사항에 대해 설명합니다.

변경 사항	설명	릴리스 날짜
AWS 계정의 모든 관리형 인스턴스에 대한 인벤토리 작성	전역 인벤토리 연결을 생성하면 AWS 계정의 모든 관리형 인스턴스에 대한 인벤토리를 쉽게 작성할 수 있습니다. 자세한 내용은 AWS 계정의 모든 관리형 노드에 대한 인벤토리 작성 단원을 참조하십시오. **Note** 전역 인벤토리 연결은 SSM Agent 버전 2.0.790.0 이상에서 사용할 수 있습니다. 인스턴스에서 SSM Agent를 업데이트하는 자세한 방법은 Run Command를 사용하여 SSM Agent 업데이트 섹션을 참조하세요.	2018년 5월 3일
Ubuntu Server 18에 기본적으로 설치되는 SSM Agent	SSM Agent는 기본적으로 Ubuntu Server 18.04 LTS 64비트 및 32비트 AMIs에 설치됩니다.	2018년 5월 2일
새 주제	새로운 주제인 특정 문서 버전을 사용하여 명령 실행에서는 document-version 파라미터를 사용하여 명령 실행 시 사용할 SSM 문서 버전을 지정하는 방법을 설명합니다.	2018년 5월 1일
새 주제	새로운 주제인 사용자 지정 인벤토리 삭제에서는 AWS CLI를 사용하여 Amazon S3에서 사용자 정의 Inventory 데이터를 삭제하는 방법을 설명합니다. 또한 사용자 정의 인벤토리 유형을 해제하거나 삭제하여 사용자 정의 인벤토리를 관리할 때 `SchemaDeleteOption` 을 사용하는 방법도 알려줍니다. 이 새로운 기능에서는 DeleteInventory API 작업을 사용합니다.	2018년 4월 19일
SSM Agent에 대한 Amazon SNS 알림	SSM Agent의 새 버전이 출시될 때 알림을 받으려면 Amazon SNS 주제를 구독하면 됩니다. 자세한 내용은 SSM Agent 알림 구독 단원을 참조하십시오.	2018년 9월 4일
CentOS 패치 지원	이제 Systems Manager가 패치 CentOS 인스턴스를 지원합니다. 지원되는 CentOS 버전에 대한 자세한 내용은 Patch	2018년 3월 29일

변경 사항	설명	릴리스 날짜
	Manager 필수 조건 섹션을 참조하세요. 패칭이 어떻게 수행되는지를 알아보려면 Patch Manager 작업 작동 방식 섹션을 참조하세요.	
새로운 섹션	AWS Systems Manager User Guide에 참조 정보를 위한 단일 소스를 제공하기 위해 AWS Systems Manager 참조 섹션이 새로 추가되었습니다. 사용 가능해지면 이 단원에 추가 콘텐츠가 추가됩니다.	2018년 3월 15일
새 주제	새로운 승인 패치 및 거부 패치 목록의 패키지 이름 형식 정보 항목에서 사용자 지정 패치 기준에 대해 승인 패치 및 거부 패치 목록에 입력할 수 있는 패키지 이름 형식을 설명합니다. Patch Manager에서 지원하는 각 운영 체제 유형에 대해 예제 형식이 제공됩니다.	2018년 3월 9일
새 주제	Systems Manager는 이제 Chef Chef InSpec과 통합됩니다. InSpec은 GitHub 또는 Amazon S3에 육안 판독 프로파일을 생성할 수 있는 오픈 소스 런타임 프레임워크입니다. Systems Manager를 사용하여 규정 준수 검사를 수행하고 준수 및 비준수 인스턴스를 확인할 수 있습니다. 자세한 내용은 Systems Manager Compliance와 함께 Chef InSpec 프로파일 사용 단원을 참조하십시오.	2018년 3월 7일
새 주제	새로운 주제 Systems Manager에 서비스 연결 역할 사용에서는 Systems Manager와 함께 AWS Identity and Access Management(IAM) 서비스 연결 역할을 사용하는 방법을 설명합니다. 현재 서비스 연결 역할은 Systems Manager Inventory를 사용하여 태그 및 Resource Groups에 대한 메타데이터를 수집할 때만 필요합니다.	2018년 2월 27일

변경 사항	설명	릴리스 날짜
신규 및 업데이트된 주제	이제 Patch Manager를 사용하여 인스턴스에 구성된 기본 소스 리포지토리가 아닌 다른 소스 리포지토리에 있는 패치를 설치할 수 있습니다. 이는 보안과 관련되지 않은 업데이트, Ubuntu Server의 PPA(Personal Package Archives) 내용, 사내 애플리케이션에 대한 업데이트 등으로 인스턴스에 패치를 적용하는 경우에 유용합니다. 사용자 지정 패치 기준을 만들 때 대체 패치 소스 리포지토리를 지정합니다. 자세한 내용은 다음 주제를 참조하세요. • 대체 패치 소스 리포지토리를 지정하는 방법(Linux) • 사용자 정의 패치 기준 작업 • 다른 OS 버전에 대한 사용자 지정 리포지토리가 있는 패치 기준 생성 또한 이제 Patch Manager를 사용하여 SUSE Linux Enterprise Server 인스턴스를 패치할 수 있습니다. Patch Manager는 SLES 12.* 버전(64비트 전용) 패치를 지원합니다. 자세한 내용은 다음 주제의 SLES 관련 정보를 참조하십시오. • 보안 패치 선택 방법 • 패치 설치 방법 • SUSE Linux Enterprise Server에서 패치 기준 규칙 작동 방법	2018년 2월 6일
새 주제	새로운 주제인 관리형 노드 패치를 위한 SSM 문서 정보에서는 보안과 관련된 최신 업데이트를 통해 관리형 인스턴스에 대해 지속적으로 패치 작업을 수행하는 데 활용할 수 있는 7개의 SSM 문서를 설명합니다.	2018년 1월 10일

변경 사항	설명	릴리스 날짜
Linux 지원과 관련된 중요 업데이트	다음 정보가 포함되도록 다양한 주제들이 업데이트되었습니다. • SSM Agent는 Amazon Linux 1 기본 AMIs(날짜: 2017년 9월 이후)에 기본적으로 설치됩니다. • Amazon ECS 최적화 AMIs 같은 비기본(non-base) 이미지를 포함하여 다른 버전의 Linux에 SSM Agent를 수동으로 설치해야 합니다.	2018년 1월 9일
새 주제	새로운 주제인 AWS-RunPatchBaseline SSM 문서 정보 단원은 이 SSM 문서가 Windows 및 Linux 시스템 모두에서 어떻게 작동하는지에 대해 자세히 설명합니다. 또한 AWS-RunPatchBaseline 문서, Operation 및 Snapshot ID에서 사용할 수 있는 2개의 파라미터에 대한 내용도 제공합니다.	2018년 1월 5일
새로운 주제	새로운 섹션인 Patch Manager 작업 작동 방식에서는 Patch Manager가 설치할 보안 패치를 결정하는 방법 및 지원되는 각 운영 체제에 이러한 패치가 설치되는 방법에 대한 기술적인 정보를 제공합니다. 또한 Linux 운영 체제의 서로 다른 배포에서 패치 기준 규칙이 작동하는 방법에 대해서도 설명합니다.	2018년 1월 2일
Systems Manager Automation 작업 참조의 제목 변경 및 이동	고객 피드백에 따라 Automation 작업 참조는 이제 Systems Manager Automation 실행서 참조라고 합니다. 또한, 위치가 Shared Resources > Documents 노드로 변경되어 Command 문서 플러그인 참조 단원에 보다 가까워졌습니다. 자세한 내용은 Systems Manager Automation 작업 참조 단원을 참조하십시오.	2017년 12월 20일

변경 사항	설명	릴리스 날짜
새로운 모니터링 장 및 내용	새로운 장인 AWS Systems Manager 모니터링에서는 지표 및 로그 데이터를 Amazon CloudWatch Logs에 보내는 방법을 설명합니다. 새로운 주제인 통합 CloudWatch Logs로 노드 로그 전송(CloudWatch 에이전트)에서는 64비트 Windows Server 인스턴스에서만 인스턴스 상 모니터링 태스크를 SSM Agent에서 CloudWatch 에이전트로 마이그레이션하는 방법을 설명합니다.	2017년 12월 14일
새로운 장	새로운 장인 AWS Systems Manager의 I자격 증명 및 액세스 관리는 AWS Identity and Access Management(IAM) 및 AWS Systems Manager를 사용하여 자격 증명을 통해 리소스에 대한 액세스를 보호할 수 있는 방법에 대한 포괄적인 정보를 제공합니다. 이러한 자격 증명은 AWS 리소스에 액세스하는 데 필요한 권한을 제공합니다(예: S3 버킷에 저장되어 있는 데이터 액세스, EC2 인스턴스에 명령 보내기 및 태그 읽기).	2017년 12월 11일
왼쪽 탐색 창 변경	새로운 AWS Systems Manager 콘솔의 머리글과 일치하도록 본 사용 설명서의 왼쪽 탐색 창의 머리글을 변경했습니다.	2017년 12월 8일
re:Invent 2017의 여러 가지 변경 사항	• AWS Systems Manager 공식 출시: AWS Systems Manager(구 Amazon EC2 Systems Manager)는 여러 AWS 리소스에 걸쳐 운영 데이터를 쉽게 중앙 집중화하고 태스크를 자동화할 수 있게 하는 통합 인터페이스입니다. 여기에서 새 AWS Systems Manager 콘솔에 액세스할 수 있습니다. 자세한 내용은 AWS Systems Manager이란? 섹션을 참조하세요. • YAML 지원: YAML에서 SSM 문서를 생성할 수 있습니다. 자세한 내용은 AWS Systems Manager Documents 단원을 참조하십시오.	2017년 11월 29일

변경 사항	설명	릴리스 날짜
Run Command를 통해 EBS 볼륨에 대해 VSS를 이용하는 스냅샷 찍기	Run Command를 사용하여 Amazon EC2 Windows 인스턴스에 연결된 모든 Amazon Elastic Block Store(Amazon EBS) 볼륨의 애플리케이션 일치 스냅샷을 생성할 수 있습니다. 스냅샷 프로세스에서는 Windows VSS(Volume Shadow Copy Service)를 사용하여 이 애플리케이션과 디스크 사이에 대기 중인 트랜잭션에서 전송되는 데이터를 비롯해 VSS 인식 애플리케이션에 대해 이미지 수준 백업을 받습니다. 뿐만 아니라 연결된 볼륨을 모두 백업해야 하는 경우 인스턴스를 종료하거나 연결을 해제할 필요가 없습니다. 자세한 내용은 Amazon EC2 사용 설명서의 AWS Systems Manager를 통해 EBS 볼륨에 대해 VSS를 이용하는 스냅샷 찍기를 참조하세요.	2017년 11월 20일
VPC 엔드포인트를 통한 Systems Manager 보안 강화	Systems Manager에서 인터페이스 VPC 엔드포인트를 사용하도록 구성함으로써 관리형 인스턴스(하이브리드 환경의 관리형 인스턴스 포함)의 보안 태세를 개선할 수 있습니다. 인터페이스 엔드포인트는 프라이빗 IP 주소를 사용하여 Amazon EC2 및 Systems Manager API에 비공개로 액세스할 수 있는 기술인 PrivateLink로 구동됩니다. PrivateLink는 관리형 인스턴스, Systems Manager 및 EC2 간의 모든 네트워크 트래픽을 Amazon 네트워크로 제한합니다(관리형 인스턴스는 인터넷에 액세스할 수 없음). 또한 인터넷 게이트웨이, NAT 디바이스 또는 가상 프라이빗 게이트웨이가 필요 없습니다. 자세한 내용은 Systems Manager용 VPC 엔드포인트를 사용하여 EC2 인스턴스의 보안 개선을 참조하세요.	2017년 11월 7일

변경 사항	설명	릴리스 날짜
파일, 서비스, Windows 역할 및 Windows 레지스트리에 대한 인벤토리 지원	SSM Inventory는 현재 관리형 인스턴스에서 다음 정보를 수집하는 작업을 지원합니다. • 파일: 이름, 크기, 버전, 설치한 날짜, 수정 및 마지막 액세스 시각 등 • 서비스: 이름, 표시 이름, 상태, 종속 서비스, 서비스 유형, 시작 유형 등 • Windows 레지스트리: 레지스트리 키 경로, 값 이름, 값 유형 및 값 • Windows 역할: 이름, 표시 이름, 경로, 기능 유형, 설치된 상태 등 이 인벤토리 유형에 대한 정보를 수집하기 전에 목록으로 만들고자 하는 인스턴스에 있는 SSM Agent를 업데이트합니다. SSM Agent 최신 버전을 실행하여 지원되는 모든 인벤토리 유형에 대한 메타데이터를 수집할 수 있는지 확인합니다. SSM Agent를 사용해 State Manager 를 업데이트하는 방법에 대한 자세한 내용은 시연: SSM Agent(CLI) 자동 업데이트 섹션을 참조하세요. 인벤토리에 관한 자세한 내용은 Systems Manager Inventory에 대해 자세히 알아보기를 참조하십시오.	2017년 11월 6일
자동화 설명서 업데이트	Systems Manager Automation에 대한 액세스 설정 및 구성 관련 내용 중 몇 가지 문제를 수정함. 자세한 내용은 Automation 설정 단원을 참조하십시오.	2017년 10월 31일

변경 사항	설명	릴리스 날짜
GitHub 및 Amazon S3 통합	원격 스크립트 실행: Systems Manager는 이제 프라이빗 또는 퍼블릭 GitHub 리포지토리로부터 또는 Amazon S3로부터 스크립트의 다운로드 및 실행을 지원합니다. 사용자 정의 SSM 문서에서 `AWS-RunRemoteScript` 사전 정의 SSM 문서 또는 `aws:downloadContent` 플러그인을 사용하여 Ansible Playbooks를 실행하고 Python, Ruby 또는 PowerShell 등의 스크립트를 실행할 수 있습니다. 이러한 변경 사항은 Systems Manager를 사용하여 하이브리드 환경에서 EC2 인스턴스 및 온프레미스 관리형 인스턴스의 구성 및 배포를 자동화할 때 코드형 인프라를 더욱 향상시킵니다. 자세한 내용은 GitHub에서 스크립트 실행 및 Amazon S3에서 스크립트 실행 섹션을 참조하세요. 복합 SSM 문서 생성: Systems Manager는 이제 기본 SSM 문서로부터 하나 이상의 보조 SSM 문서 실행을 지원합니다. 다른 문서를 실행하는 이러한 기본 문서를 복합 문서라고 부릅니다. 복합 문서는 사용자가 안티바이러스 소프트웨어 부트스트랩 또는 인스턴스 도메인 조인과 같은 일반적 태스크를 위한 표준적인 보조 SSM 문서 집합을 생성하고 AWS 계정 간에 공유할 수 있게 해줍니다. Systems Manager, GitHub 또는 Amazon S3에 저장된 복합 및 보조 문서를 실행할 수 있습니다. 복합 문서를 생성한 후 `AWS-RunDocument` 사전 정의 SSM 문서를 사용하여 실행할 수 있습니다. 자세한 내용은 복합 문서 생성 및 원격 위치에서 문서 실행 섹션을 참조하세요. SSM 문서 플러그인 레퍼런스: 보다 간편하게 액세스할 수 있도록 SSM 문서용 SSM 플러그인 레퍼런스를 Systems Manager API 레퍼런스에서 사용 설명서로 옮겼습니다. 자세한 내용은 Command 문서 플러그인 참조 단원을 참조하십시오.	2017년 10월 26일

변경 사항	설명	릴리스 날짜
Parameter Store에서 파라미터 버전 지원	파라미터를 편집하면 이제 Parameter Store가 자동으로 버전 번호를 1씩 증가시킵니다. API 호출 및 SSM 문서에서 파라미터 이름과 특정 버전 번호를 지정할 수 있습니다. 버전 번호를 지정하지 않을 경우 시스템이 자동으로 최신 버전을 사용합니다. 파라미터 버전은 파라미터가 우발적으로 변경되는 경우를 대비하여 보호 계층을 제공합니다. 모든 버전의 값을 볼 수 있으며, 필요하면 이전 버전을 참조할 수 있습니다. 또한 파라미터 버전을 사용하여 일정 기간 동안 파라미터가 몇 번 변경되었는지 확인할 수 있습니다. 자세한 내용은 파라미터 버전 작업 단원을 참조하십시오.	2017년 10월 24일
Systems Manager Documents에 태그 지정 지원	이제 AddTagsToResource API, AWS CLI 또는 AWS Tools for PowerShell을 사용하여 Systems Manager 문서에 키-값 페어로 태그를 지정할 수 있습니다. 태깅하면 지정한 태그를 기반으로 특정 리소스를 빠르게 식별할 수 있습니다. 관리형 인스턴스, 유지 관리 기간, Parameter Store 파라미터 및 패치 기준에 대한 기존의 태그 지정 지원 외에 추가된 기능입니다. 자세한 설명은 Systems Manager 문서에 태그 지정을 참조하세요.	2017년 10월 3일
피드백을 기반으로 오류를 수정하거나 콘텐츠를 업데이트하기 위한 다양한 설명서 업데이트	• Raspbian Linux에 대한 정보를 사용하여 하이브리드 및 멀티클라우드 환경에서 Systems Manager 사용하기 단원을 업데이트했습니다. • Windows Server 인스턴스에 대한 새로운 요구 사항으로 EC2 인스턴스에 Systems Manager 사용 섹션이 업데이트되었습니다. Windows Server 인스턴스에서 레거시 `AWS-ApplyPatchBaseline` SSM 문서와 같은 특정 SSM 문서를 실행하려면 SSM Agent에 Windows PowerShell 3.0 이상이 필요합니다. Windows Server 인스턴스에서 Windows Management Framework 3.0 이상을 실행하고 있는지 확인합니다. 이 프레임워크는 PowerShell을 포함합니다. 자세한 내용은 Windows Management Framework 3.0을 참조하십시오.	2017년 10월 2일

변경 사항	설명	릴리스 날짜
EC2Rescue 자동화 워크플로를 사용하여 연결할 수 없는 Windows 인스턴스 문제 해결	EC2Rescue를 사용하면 Amazon EC2 Windows Server 인스턴스에 대한 문제를 진단하고 해결할 수 있습니다. 또는 AWSSupport-ExecuteEC2Rescue 문서를 사용하여 도구를 Systems Manager Automation 워크플로로 자동 실행할 수 있습니다. AWSSupport-ExecuteEC2Rescue 문서는 EC2Rescue를 사용하는 데 일반적으로 필요한 단계를 자동화하는 Systems Manager 작업, AWS CloudFormation 작업 및 Lambda 함수를 조합하여 수행하도록 설계되었습니다. 자세한 내용은 연결할 수 없는 인스턴스에서 EC2Rescue 도구 실행 단원을 참조하십시오.	2017년 9월 29일
Amazon Linux에 SSM Agent 기본적으로 설치됨	SSM Agent는 Amazon Linux AMIs(날짜: 2017년 9월 이후)에 기본적으로 설치됩니다. Linux용 EC2 인스턴스에서 SSM Agent 사용 섹션에서 설명한 대로 다른 버전의 Linux에는 SSM Agent를 수동으로 설치해야 합니다.	2017년 9월 27일
Run Command 기능 향상	Run Command에는 다음과 같은 향상된 기능이 포함되어 있습니다. • 사용자가 특정 Amazon EC2 태그에 지정된 인스턴스에 대해서만 명령을 실행할 수 있도록 하는 조건이 포함된 IAM 정책을 생성하고 할당하여 명령 실행을 특정 인스턴스로 제한할 수 있습니다. 자세한 내용은 태그를 기반으로 Run Command 액세스 제한 단원을 참조하십시오. • Amazon EC2 태그를 사용하여 인스턴스 대상을 지정하는 다양한 옵션이 있습니다. 이제 명령을 전송할 때 여러 태그 키와 여러 태그 값을 지정할 수 있습니다. 자세한 내용은 대규모로 명령 실행 단원을 참조하십시오.	2017년 9월 12일
Raspbian에서 Systems Manager 지원됨	이제 Raspberry Pi(32비트)를 비롯하여 Raspbian Jessie 및 Raspbian Stretch 디바이스에서 Systems Manager를 실행할 수 있습니다.	2017년 9월 7일

변경 사항	설명	릴리스 날짜
Amazon CloudWatch Logs로 SSM Agent 로그 자동 전송	이제 인스턴스에 대한 구성을 간단히 변경하여 SSM Agent에서 로그 파일을 CloudWatch로 전송하도록 지정할 수 있습니다. 자세한 내용은 CloudWatch Logs에 SSM Agent 로그 전송 단원을 참조하십시오.	2017년 9월 7일
리소스 데이터 동기화 암호화	Systems Manager 리소스 데이터 동기화를 사용하여 중앙 S3 버킷에서 수십 또는 수백 개 관리형 인스턴스에 수집된 Inventory 데이터를 집계할 수 있습니다. 이제 AWS Key Management Service 키를 사용하여 리소스 데이터 동기화를 암호화할 수 있습니다. 자세한 내용은 시연: 리소스 데이터 동기화를 사용하여 인벤토리 데이터 집계 단원을 참조하십시오.	2017년 9월 1일
새로운 State Manager 연습	새로운 연습 두 개를 State Manager 설명서에 추가했습니다. 시연: SSM Agent(CLI) 자동 업데이트 연습: Windows Server용 EC2 인스턴스에서 PV 드라이버 자동 업데이트(콘솔)	2017년 8월 31일
Systems Manager Configuration Compliance	구성 규정 준수를 사용하여 관리형 인스턴스 집합에 대해 패치 규정 준수 및 구성 일관성을 검사할 수 있습니다. 여러 AWS 계정 및 AWS 리전의 데이터를 수집하여 집계한 후 규정을 준수하지 않는 특정 리소스로 드릴다운할 수 있습니다. 기본적으로 Configuration Compliance는 Patch Manager 패치 및 State Manager 연결에 대한 규정 준수 데이터를 표시합니다. 또한 IT 또는 비즈니스 요구 사항에 따라 서비스를 사용자 지정하고 자체 규정 준수 유형을 만들 수도 있습니다. 자세한 내용은 AWS Systems Manager Compliance 단원을 참조하십시오.	2017년 8월 28일

변경 사항	설명	릴리스 날짜
새로운 Automation 작업: `aws:executeAutomation`	보조 Automation 실행서를 호출하여 보조 Automation 워크플로를 실행합니다. 이 작업을 수행하여 가장 일반적인 워크플로에 대한 Automation 실행서를 생성한 다음 Automation 실행 중에 해당 문서를 참조할 수 있습니다. 이 작업을 수행하면 비슷한 실행서 간에 복제 단계가 필요하지 않으므로 Automation 실행서를 간소화할 수 있습니다. 자세한 내용은 `aws:executeAutomation` - 또 다른 자동화 실행 단원을 참조하십시오.	2017년 8월 22일
CloudWatch 이벤트의 대상으로 Automation	Automation 실행서를 Amazon CloudWatch 이벤트의 대상으로 지정하여 Automation 워크플로를 시작할 수 있습니다. 일정에 따라 또는 특정 AWS 시스템 이벤트가 발생할 때 워크플로를 시작할 수 있습니다. 자세한 내용은 이벤트를 기반으로 자동화 실행 단원을 참조하십시오.	2017년 8월 21일
State Manager 연결 버전 관리 및 일반 업데이트	이제 다른 State Manager 연결 버전을 생성할 수 있습니다. 각 연결에 대해 1,000개의 버전 한도가 있습니다. 연결에 대한 이름을 지정할 수도 있습니다. 또한 State Manager 설명서를 업데이트하여 오래된 정보와 불일치를 해결했습니다. 자세한 정보는 AWS Systems Manager State Manager 섹션을 참조하세요.	2017년 8월 21일

변경 사항	설명	릴리스 날짜
Maintenance Windows 변경 사항	Maintenance Windows에서 변경되거나 향상된 기능은 다음과 같습니다. • 이전에는 Maintenance Windows에서 Run Command를 통해서만 작업을 수행할 수 있었습니다. 이제 Systems Manager Automation, AWS Lambda 및 AWS Step Functions를 사용하여 태스크를 수행할 수 있습니다. • 유지 관리 기간의 대상을 편집하고 대상 이름, 설명 및 소유자를 지정할 수 있습니다. • Run Command 및 Automation 태스크에 대한 새 SSM 문서 지정을 비롯하여 유지 관리 기간에서 태스크를 편집할 수 있습니다. • 이제 DocumentHash, DocumentHashType, TimeoutSeconds, Comment, NotificationConfig를 비롯하여 모든 Run Command 파라미터가 지원됩니다. • 이제 대상의 등록을 취소할 때 `safe` 플래그를 사용할 수 있습니다. 설정한 경우 태스크에서 대상을 참조하면 오류가 반환됩니다. 자세한 정보는 AWS Systems Manager Maintenance Windows 섹션을 참조하세요.	2017년 8월 16일
새로운 Automation 작업: `aws:approve`	Automation 실행서에 대한 이 새 작업은 지정된 보안 주체가 작업을 승인하거나 거부할 때까지 Automation 실행을 일시 중지합니다. 필요한 승인 횟수에 도달하면 자동화 실행이 다시 시작됩니다. 자세한 정보는 Systems Manager Automation 작업 참조 섹션을 참조하세요.	2017년 8월 10일

변경 사항	설명	릴리스 날짜
자동화 역할 수임이 더 이상 필요하지 않음	이전에는 자동화를 사용하려면 서비스 역할(또는 assume role)을 지정하여 서비스에서 사용자를 대신하여 작업을 수행할 수 있도록 권한을 부여해야 했습니다. 이제 실행을 호출한 사용자의 환경에 따라 서비스가 운영되므로 더 이상 이 역할이 필요하지 않습니다. 하지만 다음과 같은 경우에는 자동화에 대한 서비스 역할을 지정해야 합니다. • 리소스에 대한 사용자의 권한을 제한하려고 하지만 사용자가 더 높은 권한이 필요한 Automation 워크플로를 실행하도록 하려는 경우입니다. 이 시나리오에서는 더 높은 권한이 있는 서비스 역할을 생성하고 사용자에게 워크플로를 실행하도록 허용할 수 있습니다. • 작업을 12시간 이상 실행해야 하는 경우 서비스 역할이 필요합니다. 자세한 정보는 Automation 설정 섹션을 참조하세요.	2017년 8월 3일
구성 규정 준수	Amazon EC2 Systems Manager Configuration Compliance를 사용하여 관리형 인스턴스 플릿에 대해 패치 규정 준수 및 구성 일관성을 검사할 수 있습니다. 여러 AWS 계정 및 AWS 리전의 데이터를 수집하여 집계한 후 규정을 준수하지 않는 특정 리소스로 드릴다운할 수 있습니다. 자세한 정보는 AWS Systems Manager Compliance 섹션을 참조하세요.	2017년 8월 8일

변경 사항	설명	릴리스 날짜
SSM 문서의 기능 향상	SSM 명령 및 Policy 문서가 이제 크로스 플랫폼 지원 기능을 제공합니다. 이 말은 단일 SSM 문서에서 Windows 및 Linux 운영 체제용 플러그인을 모두 처리할 수 있다는 것을 의미합니다. 크로스 플랫폼 지원 덕분에 관리하는 다수의 문서들을 통합할 수 있게 되었습니다. 이 기능은 스키마 버전 2.2 이상을 사용하는 SSM 문서에서 제공됩니다. 스키마 버전 2.0 이상을 사용하는 SSM Command 문서에서는 이제 동일한 유형의 플러그인을 다수 추가할 수 있게 되었습니다. 예를 들어 Command 문서를 생성하면서 `aws:runRunShellScript` 플러그인을 여러 차례 호출할 수 있습니다. 스키마 버전 2.2의 변경 사항에 대한 자세한 내용은 AWS Systems Manager 문서를 참조하세요. SSM 플러그인에 대한 자세한 내용은 명령 문서 플러그인 참조를 참조하세요.	2017년 7월 12일

변경 사항	설명	릴리스 날짜
Linux 패치	Patch Manager가 이제 다음과 같은 Linux 배포판을 패치할 수 있습니다. 64비트 및 32비트 시스템 • Amazon Linux 2014.03, 2014.09 이상 • Ubuntu Server 16.04 LTS, 14.04 LTS 또는 12.04 LTS • Red Hat Enterprise Linux(RHEL) 6.5 이상 64비트 시스템 전용 • Amazon Linux 2015.03, 2015.09 이상 • Red Hat Enterprise Linux(RHEL) 7.x 이상 자세한 내용은 AWS Systems Manager Patch Manager 단원을 참조하십시오. Note • Linux 인스턴스에 패치를 실행하려면 인스턴스가 SSM Agent 버전 2.0.834.0 이상을 실행 중이어야 합니다. 에이전트 업데이트에 대한 자세한 내용은 콘솔에서 명령 실행의 예제: SSM Agent 업데이트 섹션을 참조하세요. • `AWS-ApplyPatchBaseline` SSM 문서가 `AWS-RunPatchBaseline` 문서로 대체됩니다.	2017년 6월 7일

변경 사항	설명	릴리스 날짜
리소스 데이터 동기화	Systems Manager 리소스 데이터 동기화를 사용하여 모든 관리형 인스턴스에서 수집된 Inventory 데이터를 단일 Amazon S3 버킷으로 전송합니다. 이제 새로운 인벤토리 데이터가 수집될 때마다 리소스 데이터 동기화가 중앙 데이터를 자동으로 업데이트합니다. 모든 Inventory 데이터가 대상 S3 버킷에 저장되면 Amazon Athena 및 Amazon QuickSight 같은 서비스를 사용하여 수집한 데이터에 대한 쿼리를 실행하거나 분석할 수 있습니다. 자세한 내용은 Inventory의 리소스 데이터 동기화 구성 섹션을 참조하세요. 리소스 데이터 동기화의 사용 방법에 대한 예는 시연: 리소스 데이터 동기화를 사용하여 인벤토리 데이터 집계 섹션을 참조하세요.	2017년 6월 29일
Systems Manager 파라미터 계층 구조	수십 또는 수백 개 Systems Manager 파라미터를 하나의 집합 목록으로 관리하면 많은 시간이 들고 오류에 취약합니다. 파라미터 계층 구조를 사용하면 Systems Manager 파라미터를 조직하고 관리하는 데 도움이 됩니다. 계층 구조는 슬래시를 이용해 정의하는 경로를 포함한 파라미터 이름입니다. 이름에 세 개의 계층 구조 수준을 사용해 구별하는 예를 들어보겠습니다. /Environment/Type of computer/Application/Data `/Dev/DBServer/MySQL/db-string13` 자세한 내용은 파라미터 계층 구조 작업 단원을 참조하십시오. 파라미터 계층 구조로 작업하는 방법의 예는 파라미터 계층 구조 작업 섹션을 참조하세요.	2017년 6월 22일
SUSE Linux Enterprise Server에 대한 SSM Agent 지원	64비트 SUSE Linux Enterprise Server(SLES)에 SSM Agent를 설치할 수 있습니다. 자세한 내용은 Linux용 EC2 인스턴스에서 SSM Agent 사용 단원을 참조하십시오.	2017년 6월 14일

문서 규칙

다음은 AWS Systems Manager 사용 설명서에 대한 일반적인 인쇄 규칙입니다.

로컬 운영 체제 또는 명령줄 언어에 대한 차별화된 예제

탭을 사용하여 사용자의 로컬 운영 체제 유형에 따라 다양한 명령 예제를 표시합니다. Linux 및 macOS 예제의 경우, 백슬래시(\) 문자를 사용하여 긴 명령을 여러 줄로 나눕니다. Windows Server 예제의 경우 캐럿(^) 문자를 사용하여 명령을 여러 줄로 나눕니다.

예제

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
```

사용자 인터페이스 요소

서식: 굵은 텍스트

예: [File], [Properties]를 선택합니다.

사용자 입력(사용자가 입력하는 텍스트)

서식: 고정 폭 글꼴 텍스트

예: 이름에 **my-new-resource**를 입력합니다.

필수 값에 대한 자리 표시자 텍스트

서식: *####* 텍스트

예제

```
aws ec2 register-image --image-location DOC-EXAMPLE-BUCKET/image.manifest.xml
```

AWS 용어집

최신 AWS 용어는 AWS 용어집 참조의 AWS 용어집을 참조하세요.