aws

개발자 가이드

Amazon Elastic Compute Cloud

Amazon Elastic Compute Cloud: 개발자 가이드

Table of Contents

Amazon EC2에 대한 프로그래밍 방식 액세스

AWS Management Console 또는 프로그래밍 인터페이스를 사용하여 Amazon EC2 리소스를 생성하고 관리할 수 있습니다. Amazon EC2 콘솔 사용에 대한 자세한 내용은 Amazon EC2 사용 설명서를 참조하십시오.

작동 방식

- Amazon EC2 엔드포인트
- 최종 일관성
- 발기 부전
- 요청 제한

프로그래밍 인터페이스

- AWS Command Line Interface (AWS CLI)
- AWS CloudFormation
- AWS SDK
- 로우 레벨 API

시작하기

- 코드 예제
- 콘솔 투 코드

모니터링

- AWS CloudTrail
- 모니터링 요청

Amazon EC2 서비스 엔드포인트

엔드포인트는 웹 서비스의 진입점 역할을 하는 URL입니다. AWS Amazon EC2는 다음과 같은 엔드포인트 유형을 지원합니다.

- IPv4 엔드포인트
- IPv4 및 IPv6를 모두 지원하는 이중 스택 엔드포인트
- FIPS 엔드포인트

요청 시에, 사용할 엔드포인트와 리전을 지정할 수 있습니다. 엔드포인트를 지정하지 않으면 기본적으로 IPv4 엔드포인트가 사용됩니다. 다른 엔드포인트 유형을 사용하려면 요청에서 이를 지정해야 합니다. 이렇게 하는 방법의 예제는 엔드포인트 지정 섹션을 참조하세요.

리전 이름	지역	엔드포인트	프로토콜	
미국 동부(오하이오)	us-east-2	ec2.us-east-2.amazonaws.com ec2-fips.us-east-2.amazonaws.com ec2.us-east-2.api.aws	HTTP 및 HTTPS HTTPS HTTPS	
미국 동부(버지니아 북부)	us-east-1	ec2.us-east-1.amazonaws.com ec2-fips.us-east-1.amazonaws.com ec2.us-east-1.api.aws	HTTP 및 HTTPS HTTPS HTTPS	
미국 서부(캘리포니아 북부)	us-west-1	ec2.us-west-1.amazonaws.com ec2-fips.us-west-1.amazonaws.com ec2.us-west-1.api.aws	HTTP 및 HTTPS HTTPS HTTPS	
미국 서부(오레곤)	us-west-2	ec2.us-west-2.amazonaws.com ec2-fips.us-west-2.amazonaws.com ec2.us-west-2.api.aws	HTTP 및 HTTPS HTTPS HTTPS	

리전 이름	지역	엔드포인트	프로토콜	
아프리카(케이프타운)	af-south-1	ec2.af-south-1.amazonaws.com ec2.af-south-1.api.aws	HTTP 및 HTTPS HTTPS	
아시아 태평양(홍콩)	ap-east-1	ec2.ap-east-1.amazonaws.com ec2.ap-east-1.api.aws	HTTP 및 HTTPS HTTPS	
아시아 태평양(하이데라바드)	ap-south-2	ec2.ap-south-2.amazonaws.com	HTTPS	
아시아 태평양(자카르타)	ap-southeast-3	ec2.ap-southeast-3.amazonaws.com	HTTPS	
아시아 태평양(멜버른)	ap-southeast-4	ec2.ap-southeast-4.amazonaws.com	HTTPS	
아시아 태평양(뭄바이)	ap-south-1	ec2.ap-south-1.amazonaws.com ec2.ap-south-1.api.aws	HTTP 및 HTTPS HTTPS	
아시아 태평양(오사카)	ap-northeast-3	ec2.ap-northeast-3.amazonaws.com	HTTP 및 HTTPS	
아시아 태평양(서울)	ap-northeast-2	ec2.ap-northeast-2.amazonaws.com ec2.ap-northeast-2.api.aws	HTTP 및 HTTPS HTTPS	

리전 이름	지역	엔드포인트	프로토콜
아시아 태평양(싱가포르)	ap-southeast-1	ec2.ap-southeast-1.amazonaws.com ec2.ap-southeast-1.api.aws	HTTP 및 HTTPS HTTPS
아시아 태평양(시드니)	ap-southeast-2	ec2.ap-southeast-2.amazonaws.com ec2.ap-southeast-2.api.aws	HTTP 및 HTTPS HTTPS
아시아 태평양(도쿄)	ap-northeast-1	ec2.ap-northeast-1.amazonaws.com ec2.ap-northeast-1.api.aws	HTTP 및 HTTPS HTTPS
캐나다(중부)	ca-central-1	ec2.ca-central-1.amazonaws.com ec2-fips.ca-central-1.amazonaws.com ec2.ca-central-1.api.aws	HTTP 및 HTTPS HTTPS HTTPS
캐나다 서부(캘거리)	ca-west-1	ec2.ca-west-1.amazonaws.com ec2-fips.ca-west-1.amazonaws.com	HTTPS HTTPS
유럽(프랑크푸르트)	eu-central-1	ec2.eu-central-1.amazonaws.com ec2.eu-central-1.api.aws	HTTP 및 HTTPS HTTPS
유럽(아일랜드)	eu-west-1	ec2.eu-west-1.amazonaws.com ec2.eu-west-1.api.aws	HTTP 및 HTTPS HTTPS

리전 이름	지역	엔드포인트	프로토콜	
유럽(런던)	eu-west-2	ec2.eu-west-2.amazonaws.com ec2.eu-west-2.api.aws	HTTP 및 HTTPS HTTPS	
유럽(밀라노)	eu-south-1	ec2.eu-south-1.amazonaws.com ec2.eu-south-1.api.aws	HTTP 및 HTTPS HTTPS	
유럽(파리)	eu-west-3	ec2.eu-west-3.amazonaws.com ec2.eu-west-3.api.aws	HTTP 및 HTTPS HTTPS	
유럽(스페인)	eu-south-2	ec2.eu-south-2.amazonaws.com	HTTPS	
유럽(스톡홀름)	eu-north-1	ec2.eu-north-1.amazonaws.com ec2.eu-north-1.api.aws	HTTP 및 HTTPS HTTPS	
유럽(취리히)	eu-central-2	ec2.eu-central-2.amazonaws.com	HTTPS	
이스라엘(텔아비브)	il-central-1	ec2.il-central-1.amazonaws.com	HTTPS	
중동(바레인)	me-south-1	ec2.me-south-1.amazonaws.com ec2.me-south-1.api.aws	HTTP 및 HTTPS HTTPS	
중동(UAE)	me-central-1	ec2.me-central-1.amazonaws.com	HTTPS	

리전 이름	지역	엔드포인트	프로토콜	
남아메리카(상파울루)	sa-east-1	ec2.sa-east-1.amazonaws.com ec2.sa-east-1.api.aws	HTTP 및 HTTPS HTTPS	
AWS GovCloud (미국 동부)	us-gov-east-1	ec2.us-gov-east-1.amazonaws.com ec2.us-gov-east-1.api.aws	HTTPS HTTPS	
AWS GovCloud (미국 서부)	us-gov-west-1	ec2.us-gov-west-1.amazonaws.com ec2.us-gov-west-1.api.aws	HTTPS HTTPS	

지역에 대한 자세한 내용은 Amazon EC2 사용 설명서의 지역 및 가용 영역을 참조하십시오. Amazon EC2의 엔드포인트 목록은 Amazon Web Services 일반 참조의 리전 및 엔드포인트를 참조하십시오.

주제

- IPv4 엔드포인트
- 이중 스택(IPv4 및 IPv6) 엔드포인트
- 엔드포인트 지정

FIPS 엔드포인트에 대한 자세한 내용은 Amazon Web Services 일반 참조에서 FIPS 엔드포인트를 참조하세요.

IPv4 엔드포인트

IPv4 엔드포인트는 IPv4 트래픽만 지원합니다. IPv4 엔드포인트는 모든 리전에 사용할 수 있습니다.

범용 엔드포인트 `ec2.amazonaws.com`을 지정하면 `us-east-1`의 엔드포인트를 사용합니다. 다른 리전을 사용하려면 연결된 엔드포인트를 지정해야 합니다. 예를 들어 `ec2.us-east-2.amazonaws.com`을 엔드포인트로 지정하면 요청이 `us-east-2` 엔드포인트로 전달됩니다.

IPv4 엔드포인트 이름에는 다음 명명 규칙이 사용됩니다.

- *service*.*region*.amazonaws.com

예를 들어 eu-west-1 리전의 IPv4 엔드포인트 이름은 ec2.eu-west-1.amazonaws.com입니다. Amazon EC2의 엔드포인트 목록은 Amazon Web Services 일반 참조의 리전 및 엔드포인트를 참조하십시오.

이중 스택(IPv4 및 IPv6) 엔드포인트

이중 스택 엔드포인트는 IPv4 트래픽과 IPv6 트래픽을 모두 지원합니다. 이중 스택 엔드포인트는 다음 지역에서만 사용할 수 있습니다.

- us-east-1—미국 동부 (버지니아 북부)
- us-east-2—미국 동부 (오하이오)
- us-west-2—미국 서부 (오레곤)
- eu-west-1—유럽 (아일랜드)
- ap-south-1—아시아 태평양 (뭄바이)
- sa-east-1—남미 (상파울루)
- us-gov-east-1— (미국 동부)AWS GovCloud
- us-gov-west-1—AWS GovCloud (미국 서부)

이중 스택 엔드포인트에 요청하는 경우, 엔드포인트 URL이 네트워크 및 클라이언트에서 사용하는 프로토콜에 따라 IPv6 또는 IPv4 주소로 확인됩니다.

Amazon EC2는 지역별 이중 스택 엔드포인트만 지원합니다. 즉, 엔드포인트 이름의 일부로 지역을 지정해야 합니다. 이중 스택 엔드포인트 이름에는 다음 명명 규칙이 사용됩니다.

- ec2.*region*.api.aws

예를 들어 eu-west-1 리전의 이중 스택 엔드포인트 이름은 ec2.eu-west-1.api.aws입니다. Amazon EC2의 엔드포인트 목록은 Amazon Web Services 일반 참조의 리전 및 엔드포인트를 참조하십시오.

엔드포인트 지정

이 섹션에서는 요청 시에 엔드포인트를 지정하는 방법을 몇 가지 예로 보여줍니다.

AWS CLI

다음 예는 를 사용하여 us-east-2 지역의 엔드포인트를 지정하는 방법을 보여줍니다. AWS CLI

- 이중 스택

```
aws ec2 describe-regions --region us-east-2 --endpoint-url https://ec2.us-east-2.api.aws
```

- IPv4

```
aws ec2 describe-regions --region us-east-2 --endpoint-url https://ec2.us-east-2.amazonaws.com
```

AWS SDK for Java 2.x

다음 예제는 를 사용하여 us-east-2 리전의 엔드포인트를 지정하는 방법을 보여줍니다 AWS SDK for Java 2.x.

- 이중 스택

```
Ec2Client client = Ec2Client.builder()
    .region(Region.US_EAST_2)
    .endpointOverride(URI.create("https://ec2.us-east-2.api.aws"))
    .build();
```

- IPv4

```
Ec2Client client = Ec2Client.builder()
    .region(Region.US_EAST_2)
    .endpointOverride(URI.create("https://ec2.us-east-2.amazonaws.com"))
    .build();
```

AWS SDK for Java 1.x

다음 예제는 AWS SDK for Java 1.x를 사용하여 eu-west-1 리전의 엔드포인트를 지정하는 방법을 보여줍니다.

- 이중 스택

```
AmazonEC2 s3 = AmazonEC2ClientBuilder.standard()
     .withEndpointConfiguration(new EndpointConfiguration(
          "https://ec2.eu-west-1.api.aws",
          "eu-west-1"))
     .build();
```

- IPv4

```
AmazonEC2 s3 = AmazonEC2ClientBuilder.standard()
     .withEndpointConfiguration(new EndpointConfiguration(
          "https://ec2.eu-west-1.amazonaws.com",
          "eu-west-1"))
     .build();
```

AWS SDK for Go

다음 예제는 를 사용하여 us-east-1 리전의 엔드포인트를 지정하는 방법을 보여줍니다. AWS SDK for Go

- 이중 스택

```
sess := session.Must(session.NewSession())
svc := ec2.New(sess, &aws.Config{
    Region: aws.String(endpoints.UsEast1RegionID),
    Endpoint: aws.String("https://ec2.us-east-1.api.aws")
})
```

- IPv4

```
sess := session.Must(session.NewSession())
svc := ec2.New(sess, &aws.Config{
    Region: aws.String(endpoints.UsEast1RegionID),
    Endpoint: aws.String("https://ec2.us-east-1.amazonaws.com")
})
```

Amazon EC2 API의 최종 일관성

Amazon EC2 API는 API를 지원하는 시스템의 분산형 특성 때문에 최종 일관성 모델을 따릅니다. 즉, Amazon EC2 리소스에 영향을 미치는 API 명령 실행 결과가 이후에 실행하는 모든 명령에 즉시 표시

되지 않을 수 있습니다. 이전 API 명령 바로 다음에 나오는 API 명령을 실행할 때는 이 점을 염두에 두어야 합니다.

최종 일관성은 리소스 관리 방식에 영향을 미칠 수 있습니다. 예를 들어, 명령을 실행하여 리소스를 만들면 결국 다른 명령에서도 해당 리소스를 볼 수 있게 됩니다. 즉, 방금 만든 리소스를 수정하거나 설명하는 명령을 실행하면 해당 ID가 시스템 전체에 전파되지 않았을 수 있으며 리소스가 존재하지 않는다는 오류 메시지가 표시됩니다.

최종 일관성을 관리하려면 다음과 같이 할 수 있습니다.

- 명령을 실행하여 리소스를 수정하기 전에 리소스의 상태를 확인하십시오. 지수 백오프 알고리즘을 사용하여 적절한 `Describe` 명령을 실행하여 이전 명령이 시스템에 전파되는 데 충분한 시간을 허용하는지 확인합니다. 이 작업을 수행하려면 몇 초의 대기 시간부터 시작하여 점차 최대 5분까지 늘려가며 `Describe` 명령을 반복적으로 실행합니다.
- 명령이 정확한 응답을 반환하더라도 후속 `Describe` 명령 사이에 대기 시간을 추가하십시오. 몇 초의 대기 시간으로 시작하는 지수 백오프 알고리즘을 적용하고 대기 시간을 약 5분까지 점진적으로 늘립니다.

최종 일관성 오류의 예

다음은 최종 일관성의 결과로 발생할 수 있는 오류 코드의 예입니다.

- `InvalidInstanceID.NotFound`

  `RunInstances`명령을 성공적으로 실행한 다음 의 `RunInstances` 응답에 제공된 인스턴스 ID를 사용하여 다른 명령을 즉시 실행하면 `InvalidInstanceID.NotFound` 오류가 반환될 수 있습니다. 그렇다고 인스턴스가 존재하지 않는 것은 아닙니다.

  영향을 받을 수 있는 몇 가지 특정 명령은 다음과 같습니다.

  - `DescribeInstances`: 인스턴스의 실제 상태를 확인하려면 지수 백오프 알고리즘을 사용하여 이 명령을 실행합니다.
  - `TerminateInstances`: 인스턴스의 상태를 확인하려면 먼저 지수 백오프 알고리즘을 사용하여 `DescribeInstances` 명령을 실행합니다.

  > **Important**
  >
  > 실행 `TerminateInstances` 후 `InvalidInstanceID.NotFound` 오류가 발생한다고 해서 인스턴스가 종료되었거나 종료될 예정인 것은 아닙니다. 인스턴스가 아직 실행

중일 수 있습니다. 따라서 를 사용하여 인스턴스의 상태를 먼저 확인하는 것이 중요합니다DescribeInstances.

- `InvalidGroup.NotFound`

  `CreateSecurityGroup`명령을 성공적으로 실행한 다음 응답에 제공된 보안 그룹 ID를 사용하여 다른 명령을 즉시 실행하면 `InvalidGroup.NotFound` 오류가 반환될 수 있습니다. `CreateSecurityGroup` 보안 그룹의 상태를 확인하려면 지수 백오프 `DescribeSecurityGroups` 알고리즘을 사용하여 명령을 실행합니다.

- `InstanceLimitExceeded`

  현재 인스턴스 한도가 지정된 인스턴스 유형에 허용하는 것보다 더 많은 인스턴스를 요청했습니다. 종료된 인스턴스는 종료된 후 한동안 인스턴스 한도 계산에 포함되므로 인스턴스를 빠르게 시작하고 종료하는 경우 이 한도에 예기치 않게 도달할 수 있습니다.

Amazon EC2 API 요청의 불능성 보장

변형 API 요청을 만들 때 요청은 일반적으로 작업의 비동기 워크플로가 완료되기 전에 결과를 반환합니다. 요청이 이미 결과를 반환했더라도 작업이 완료되기 전에 시간이 초과되거나 다른 서버 문제가 발생할 수도 있습니다. 이로 인해 요청의 성공 여부를 판단하기 어려울 수 있으며 작업이 성공적으로 완료되었는지 확인하기 위해 여러 번의 재시도가 발생할 수 있습니다. 그러나 원래 요청과 후속 재시도가 성공하면 작업이 여러 번 완료됩니다. 즉, 의도한 것보다 더 많은 리소스를 생성할 수 있습니다.

멱등성은 API 요청이 한 번만 완료되도록 합니다. 멱등성 요청을 사용하면 원래 요청이 성공적으로 완료되면 추가 작업을 수행하지 않고 후속 재시도가 성공적으로 완료됩니다. 하지만 결과에는 현재 생성 상태와 같은 업데이트된 정보가 포함될 수 있습니다.

내용

아마존 EC2의 불능성

다음 API 작업은 기본적으로 불능이며 추가 구성이 필요하지 않습니다. 또한 해당 AWS CLI 명령은 기본적으로 항등성을 지원합니다.

기본적으로 불능입니다.

- AssociateAddress
- CreateVpnConnection
- DisassociateAddress
- ReplaceNetworkAclAssociation
- TerminateInstances

다음 API 액션은 선택적으로 클라이언트 토큰을 사용한 직등성을 지원합니다. 또한 해당 AWS CLI 명령은 클라이언트 토큰을 사용한 직등성을 지원합니다. 클라이언트 토큰은 대소문자를 구분하여 최대 64개의 ASCII 문자로 구성된 고유한 문자열입니다. 이러한 작업 중 하나를 사용하여 동등한 API 요청을 하려면 요청에 클라이언트 토큰을 지정하십시오. 다른 API 요청에 동일한 클라이언트 토큰을 재사용해서는 안 됩니다. 동일한 클라이언트 토큰과 동일한 파라미터를 사용하여 성공적으로 완료된 요청을 재시도하면 추가 작업을 수행하지 않고도 재시도가 성공합니다. 동일한 클라이언트 토큰을 사용하여 성공적인 요청을 재시도하지만 지역 또는 가용 영역을 제외하고 하나 이상의 파라미터가 다르면 재시도가 실패하고 오류가 발생합니다. `IdempotentParameterMismatch`

클라이언트 토큰을 사용한 Impotent

- AllocateHosts
- AllocateIpamPoolCidr
- AssociateClientVpnTargetNetwork
- AssociateIpamResourceDiscovery
- AttachVerifiedAccessTrustProvider
- AuthorizeClientVpnIngress
- CopyFpgaImage
- CopyImage
- CreateCapacityReservation
- CreateCapacityReservationFleet

- CreateClientVpnEndpoint
- CreateClientVpnRoute
- CreateEgressOnlyInternetGateway
- CreateFleet
- CreateFlowLogs
- CreateFpgaImage
- CreateInstanceConnectEndpoint
- CreateIpam
- CreateIpamPool
- CreateIpamResourceDiscovery
- CreateIpamScope
- CreateLaunchTemplate
- CreateLaunchTemplateVersion
- CreateManagedPrefixList
- CreateNatGateway
- CreateNetworkAcl
- CreateNetworkInsightsAccessScope
- CreateNetworkInsightsPath
- CreateNetworkInterface
- CreateReplaceRootVolumeTask
- CreateReservedInstancesListing
- CreateRouteTable
- CreateTrafficMirrorFilter
- CreateTrafficMirrorFilterRule
- CreateTrafficMirrorSession
- CreateTrafficMirrorTarget
- CreateVerifiedAccessEndpoint
- CreateVerifiedAccessGroup

- CreateVerifiedAccessInstance
- CreateVerifiedAccessTrustProvider
- CreateVolume
- CreateVpcEndpoint
- CreateVpcEndpointConnectionNotification
- CreateVpcEndpointServiceConfiguration
- DeleteVerifiedAccessEndpoint
- DeleteVerifiedAccessGroup
- DeleteVerifiedAccessInstance
- DeleteVerifiedAccessTrustProvider
- DetachVerifiedAccessTrustProvider
- ExportImage
- ImportImage
- ImportSnapshot
- ModifyInstanceCreditSpecification
- ModifyLaunchTemplate
- ModifyReservedInstances
- ModifyVerifiedAccessEndpoint
- ModifyVerifiedAccessEndpointPolicy
- ModifyVerifiedAccessGroup
- ModifyVerifiedAccessGroupPolicy
- ModifyVerifiedAccessInstance
- ModifyVerifiedAccessInstanceLoggingConfiguration
- ModifyVerifiedAccessTrustProvider
- ProvisionIpamPoolCidr
- PurchaseHostReservation
- RequestSpotFleet
- RequestSpotInstances

- RunInstances
- StartNetworkInsightsAccessScopeAnalysis
- StartNetworkInsightsAnalysis

발기 부전의 유형

- 지역 — 요청은 각 지역에서 효력이 없습니다. 하지만 다른 지역에서 동일한 클라이언트 토큰을 포함한 동일한 요청을 사용할 수 있습니다.
- 영역 — 요청은 지역 내 각 가용 영역에서 동등합니다. 예를 들어 동일한 지역에 있는 두 번의 호출에서 동일한 클라이언트 토큰을 지정하는 경우 매개변수에 다른 값을 지정하면 호출이 성공합니다. AllocateHosts AvailabilityZone

RunInstances 불능성

RunInstancesAPI 액션은 지역 및 영역 발기부전을 모두 사용합니다.

사용되는 불능성 유형은 API 요청에서 가용 영역을 지정하는 방법에 따라 달라집니다. RunInstances 요청은 다음과 같은 경우에 영역 불능성을 사용합니다.

- Placement 데이터 유형의 파라미터를 사용하여 가용 영역을 명시적으로 지정하는 경우 AvailabilityZone
- 파라미터를 사용하여 가용 영역을 암시적으로 지정하는 경우 SubnetId

가용 영역을 명시적 또는 묵시적으로 지정하지 않는 경우 요청은 지역별 동등성을 사용합니다.

영역 불능성

영역 불능성은 RunInstances API 요청이 지역 내 각 가용 영역에서 무력화되도록 합니다. 이렇게 하면 동일한 클라이언트 토큰을 사용한 요청이 지역 내 각 가용 영역 내에서 한 번만 완료될 수 있습니다. 하지만 동일한 클라이언트 토큰을 사용하여 해당 지역의 다른 가용 영역에서 인스턴스를 시작할 수 있습니다.

예를 들어 가용 영역에서 인스턴스를 시작하라는 idempotent 요청을 보낸 다음 가용 영역의 요청에 동일한 클라이언트 토큰을 사용하면 각 가용 영역에서 인스턴스가 시작됩니다. `us-east-1a` `us-east-1b` 파라미터 중 하나 이상이 다른 경우 해당 가용 영역에서 동일한 클라이언트 토큰을 사용한 후속 재시도는 추가 작업을 수행하지 않고 성공적으로 반환되거나 오류와 함께 실패합니다. `IdempotentParameterMismatch`

지역적 불능성

지역별 불능성은 RunInstances API 요청이 한 지역에서 무력화되도록 합니다. 이렇게 하면 동일한 클라이언트 토큰을 사용한 요청을 한 지역 내에서 한 번만 완료할 수 있습니다. 하지만 동일한 클라이언트 토큰을 사용한 똑같은 요청을 사용하여 다른 지역에서 인스턴스를 시작할 수 있습니다.

예를 들어, `us-east-1` 리전에서 인스턴스를 시작하라는 idempotent 요청을 보낸 다음 해당 리전의 요청에 동일한 클라이언트 토큰을 사용하면 각 `eu-west-1` 리전에서 인스턴스가 시작됩니다. 파라미터 중 하나 이상이 다른 경우 해당 지역에서 동일한 클라이언트 토큰을 사용한 후속 재시도는 추가 작업을 수행하지 않고 성공적으로 반환되거나 오류가 발생하여 실패합니다. `IdempotentParameterMismatch`

> **Tip**
>
> 요청된 지역의 가용 영역 중 하나를 사용할 수 없는 경우 지역 동일성을 사용하는 RunInstances 요청은 실패할 수 있습니다. AWS 인프라에서 제공하는 가용 영역 기능을 활용하려면 인스턴스를 시작할 때 영역 불능성을 사용하는 것이 좋습니다. RunInstances 영역 불능성을 사용하고 가용 영역을 대상으로 하는 요청은 요청된 지역의 다른 가용 영역을 사용할 수 없는 경우에도 성공합니다.

예

AWS CLI 명령 예제

AWS CLI 명령을 무효화하려면 옵션을 추가하십시오. `--client-token`

예 1: 불능성

다음 allocate-hosts 명령은 클라이언트 토큰을 포함하므로 동일성을 사용합니다.

```
aws ec2 allocate-hosts  --instance-type m5.large  --availability-zone eu-west-1a  --
auto-placement on  --quantity 1 --client-token 550e8400-e29b-41d4-a716-446655440000
```

예 2: 실행 인스턴스의 지역별 동일성

다음 run-instance 명령은 클라이언트 토큰을 포함하기 때문에 지역별 동일성을 사용하지만 가용 영역을 명시적 또는 묵시적으로 지정하지는 않습니다.

```
aws ec2 run-instances --image-id ami-b232d0db --count 1 --key-name my-key-pair --
client-token 550e8400-e29b-41d4-a716-446655440000
```

예 3: 실행 인스턴스의 영역 동일성

다음 run-instance 명령은 클라이언트 토큰과 명시적으로 지정된 가용 영역을 포함하므로 영역 동일성을 사용합니다.

```
aws ec2 run-instances  --placement "AvailabilityZone=us-east-1a" --image-id ami-
b232d0db --count 1 --key-name my-key-pair --client-token 550e8400-e29b-41d4-
a716-446655440000
```

API 요청 예제

API 요청을 무효화하려면 파라미터를 추가하세요. `ClientToken`

예 1: 불능성

다음 AllocateHostsAPI 요청은 클라이언트 토큰을 포함하므로 항등성을 사용합니다.

```
https://ec2.amazonaws.com/?Action=AllocateHosts
&AvailabilityZone=us-east-1b
&InstanceType=m5.large
&Quantity=1
&AutoPlacement=off
&ClientToken=550e8400-e29b-41d4-a716-446655440000
&AUTHPARAMS
```

예시 2: 지역별 불능성 RunInstances

다음 RunInstancesAPI 요청은 클라이언트 토큰을 포함하기 때문에 지역별 동등성을 사용하지만 가용 영역을 명시적 또는 묵시적으로 지정하지는 않습니다.

```
https://ec2.amazonaws.com/?Action=RunInstances
&ImageId=ami-3ac33653
&MaxCount=1
&MinCount=1
&KeyName=my-key-pair
&ClientToken=550e8400-e29b-41d4-a716-446655440000
&AUTHPARAMS
```

예 3: 영역 불능성 RunInstances

다음 RunInstancesAPI 요청은 클라이언트 토큰과 명시적으로 지정된 가용 영역을 포함하므로 영역 불능성을 사용합니다.

```
https://ec2.amazonaws.com/?Action=RunInstances
&Placement.AvailabilityZone=us-east-1d
&ImageId=ami-3ac33653
&MaxCount=1
&MinCount=1
&KeyName=my-key-pair
&ClientToken=550e8400-e29b-41d4-a716-446655440000
&AUTHPARAMS
```

무력한 요청에 대한 재시도 권장 사항

다음 테이블은 멱등성 API 요청에 대해 얻을 수 있는 몇 가지 일반적인 응답을 보여주고 재시도 권장 사항을 제공합니다.

응답	권장 사항	설명
200 OK	다시 시도하지 않음	원래 요청이 성공적으로 완료되었습니다. 이후의 모든 재시도는 성공적으로 반환됩니다.
400 시리즈 응답 코드(클라이언트 오류)	다시 시도하지 않음	다음 중에서는 요청에 문제가 있습니다. • 유효하지 않은 파라미터 또는 파라미터 조합이 포함되어 있습니다. • 권한이 없는 작업 또는 리소스를 사용합니다. • 상태를 변경하는 과정에 있는 리소스를 사용합니다. 요청에 상태 변경 프로세스에 있는 리소스가 포함된 경우 요청 재시도가 성공할 수 있습니다.

응답	권장 사항	설명
500 시리즈 응답 코드 (서버 오류)	재시도	이 오류는 AWS 서버 측 문제로 인해 발생하며 일반적으로 일시적입니다. 적절한 백오프 전략으로 요청을 반복합니다.

Amazon EC2 API에 대한 요청 속도 조절

Amazon EC2는 리전별로 각 AWS 계정에 대한 EC2 API 요청을 제한합니다. 이는 서비스 성능을 높이고 모든 Amazon EC2 고객이 공평하게 사용할 수 있도록 하기 위한 것입니다. 스로틀을 사용하면 Amazon EC2 API에 대한 호출이 최대 허용 API 요청 한도를 초과하지 않도록 할 수 있습니다. API 호출에는 요청 한도가 적용됩니다. 발신지는 다음과 같습니다.

- 타사 애플리케이션
- 명령줄 도구
- 아마존 EC2 콘솔

API 제한 한도를 초과하면 오류 코드가 나타납니다. `RequestLimitExceeded`

내용
- 제한 적용 방법
- 제한 한계
- API 스로틀링 모니터링
- 재시도 및 지수 백오프
- 한도 증가 요청

제한 적용 방법

Amazon EC2는 토큰 버킷 알고리즘을 사용하여 API 제한을 구현합니다. 이 알고리즘을 사용하면 계정에 특정 수의 토큰을 보관하는 버킷이 있습니다. 버킷의 토큰 수는 특정 초당 전송률 제한 한도를 나타냅니다.

Amazon EC2는 두 가지 유형의 API 제한을 구현합니다.

API 스로틀링 유형

요청 속도 제한

요청 속도 제한을 사용하면 API 요청 수가 제한됩니다. 제출한 각 요청은 버킷에서 하나의 토큰을 제거합니다. 예를 들어 non-mutating (`Describe*`) API 작업의 버킷 크기는 토큰 100개이므로 1초에 최대 100개의 요청을 보낼 수 있습니다. `Describe*` 1초에 요청이 100개를 초과하면 병목 현상이 발생하고 1초 이내에 남은 요청은 실패합니다.

버킷은 설정된 속도로 자동으로 다시 채워집니다. 버킷이 최대 용량 미만인 경우 최대 용량에 도달할 때까지 1초마다 설정된 수의 토큰이 버킷에 다시 추가됩니다. 리필 토큰이 도착했을 때 버킷이 가득 차면 토큰은 폐기됩니다. 버킷에는 최대 토큰 수보다 많은 토큰을 담을 수 없습니다. 예를 들어 non-mutating (`Describe*`) API 작업의 버킷 크기는 토큰 100개이고 리필 속도는 초당 토큰 20개입니다. 1초에 100개의 `Describe*` API 요청을 하면 버킷은 즉시 제로 (0) 토큰으로 줄어듭니다. 그러면 최대 토큰 100개에 도달할 때까지 매초 20개의 토큰이 버킷에 다시 채워집니다. 즉, 이전에 비어 있던 버킷이 5초 후에 최대 용량에 도달합니다.

API 요청을 하기 전에 버킷이 완전히 가득 찰 때까지 기다릴 필요가 없습니다. 버킷에 추가된 토큰은 그대로 사용할 수 있습니다. 리필 토큰을 즉시 사용하면 버킷이 최대 용량에 도달하지 못합니다. 예를 들어, 콘솔 논뮤테이팅 액션의 버킷 크기는 토큰 100개이고 리필 속도는 초당 토큰 10개입니다. 1초에 100개의 API 요청을 생성하여 버킷을 고갈시키더라도 계속해서 초당 10개의 API 요청을 할 수 있습니다. 초당 API 요청 수가 10개 미만인 경우에만 버킷을 최대 용량까지 다시 채울 수 있습니다.

리소스 속도 제한

다음 표에 설명된 것과 같은 일부 API 작업은 요청 속도 `RunInstances` 제한과 `TerminateInstances` 함께 리소스 속도 제한을 사용합니다. 이러한 API 작업에는 요청의 영향을 받는 리소스 수에 따라 고갈되는 별도의 리소스 토큰 버킷이 있습니다. 요청 토큰 버킷과 마찬가지로 리소스 토큰 버킷도 버스트가 가능한 최대 버킷과 필요한 기간 동안 일정한 비율의 요청을 유지할 수 있는 리필 비율이 있습니다. 다음 API 호출을 지원하기 위해 버킷을 아직 리필하지 않은 경우를 포함하여 API의 특정 버킷 한도를 초과하는 경우, 총 API 스로틀 한도에 도달하지 않았더라도 API의 동작이 제한됩니다.

예를 들어 의 리소스 토큰 버킷 크기는 토큰 `RunInstances` 1000개이고 리필 속도는 초당 토큰 2개입니다. 따라서 1,000개 인스턴스에 대한 요청 1개 또는 250개 인스턴스에 대한 요청 4개와 같이 원하는 수의 API 요청을 사용하여 1,000개의 인스턴스를 즉시 시작할 수 있습니다. 리소스 토큰 버킷이 비워지

면 인스턴스 2개에 요청 1개 또는 인스턴스 1개에 요청 2개를 사용하여 초당 최대 두 개의 인스턴스를 시작할 수 있습니다.

자세한 정보는 리소스 토큰 버킷 크기 및 리필 비율을 참조하세요.

제한 한계

다음 섹션에서는 요청 토큰 버킷과 리소스 토큰 버킷 크기 및 리필 속도에 대해 설명합니다.

Limits

- 요청 토큰 버킷 크기 및 리필 속도
- 리소스 토큰 버킷 크기 및 리필 비율

요청 토큰 버킷 크기 및 리필 속도

요청 속도 제한을 위해 API 작업은 다음 범주로 그룹화됩니다.

- 비변경 작업 — 리소스에 대한 데이터를 검색하는 API 작업입니다. 이 범주에는 일반적으로,`DescribeRouteTables`, `DescribeImages` 등의 모든 `Describe*` 작업이 포함됩니다. `DescribeHosts` 이러한 API 작업은 일반적으로 API 제한 제한이 가장 높습니다.
- 필터링되지 않고 페이지가 지정되지 않은 비 변경 작업 — 페이지 매김이나 필터를 지정하지 않고 호출하면 더 작은 토큰 버킷의 토큰을 사용하는 비변경 API 작업의 특정 하위 집합입니다. 표준 (대형) 토큰 버킷에서 토큰이 차감되도록 페이지 매김 및 필터링을 사용하는 것이 좋습니다.
- 변경 작업 — 리소스를 생성, 수정 또는 삭제하는 API 작업입니다. 이 범주에는 일반적으로,, 등 비변경 작업으로 분류되지 않은 모든 API 작업이 포함됩니다. `CreateVolume` `ModifyHosts` `DeleteSnapshot` 이러한 작업은 변경하지 않는 API 호출보다 스로틀링 한도가 낮습니다.
- 리소스 집약적 작업 — 완료하는 데 시간이 가장 많이 걸리고 가장 많은 리소스를 소비하는 API 작업을 변경합니다. 이러한 액션은 뮤테이팅 액션보다 스로틀링 한도가 훨씬 낮습니다. 이들은 다른 뮤테이팅 액션과는 별도로 조절됩니다.
- 콘솔 비변경 작업 — Amazon EC2 콘솔에서 호출되는 비변경 API 작업입니다. 이러한 API 작업은 다른 비변경 API 작업과 별도로 조절됩니다.
- 분류되지 않은 작업 — 이러한 API 작업은 정의상 다른 범주 중 하나에 해당되더라도 고유한 토큰 버킷 크기와 리필 비율을 받습니다.

다음 표는 모든 지역의 요청 토큰 버킷 크기와 리필 비율을 보여줍니다. AWS

API 작업 카테고리	작업	버킷 최대 용량	버킷 리필 속도
변이가 없는 행동	• Describe* • Get*	100	20
필터링되지 않고 페이지가 지정되지 않은, 변경 불가능한 액션	• DescribeInstances • DescribeNetworkInterfaces • DescribeVolumes • DescribeInstanceStatus • DescribeSnapshots • DescribeSecurityGroups • DescribeSpotInstanceRequests	50	10
뮤테이팅 액션	비변경 작업으로 분류되지 않은 API 작업	200	5
리소스 집약적인 작업	•	50	5

API 작업 카테고리	작업	버킷 최대 용량	버킷 리필 속도
	AuthorizeSecurityGroupIngress • CancelSpotInstanceRequests • CreateKeyPair • RequestSpotInstances • RevokeSecurityGroupIngress • CreateVpcPeeringConnection • AcceptVpcPeeringConnection • RejectVpcPeeringConnection • DeleteVpcPeeringConnection		

API 작업 카테고리	작업	버킷 최대 용량	버킷 리필 속도
콘솔 논뮤테이팅 액션	• Describe* • Get*	100	10
분류되지 않은 액션	RunInstances	5	2
	StartInstances	5	2
	CreateVpcEndpoint	4	0.3
	ModifyVpcEndpoint	4	0.3
	DeleteVpcEndpoints	4	0.3
	AcceptVpcEndpointConnections	10	1
	RejectVpcEndpointConnections	10	1
	CreateVpcEndpointServiceConfiguration	10	1
	ModifyVpcEndpointServiceConfiguration	10	1

API 작업 카테고리	작업	버킷 최대 용량	버킷 리필 속도
	DeleteVpcEndpointServiceConfigurations	10	1
	CreateDefaultVpc	1	1
	CreateDefaultSubnet	1	1
	MoveAddressToVpc	1	1
	RestoreAddressToClassic	1	1
	DescribeMovingAddresses	1	1
	AdvertiseByoipCidr	1	0.1
	ProvisionByoipCidr	1	0.1
	DescribeByoipCidrs	1	0.5
	DeprovisionByoipCidr	1	0.1
	WithdrawByoipCidr	1	0.1

API 작업 카테고리	작업	버킷 최대 용량	버킷 리필 속도
	DescribeReservedInstancesOfferings	10	10
	PurchaseReservedInstancesOffering	5	5
	DescribeSpotFleetRequests	50	3
	DescribeSpotFleetInstances	100	5
	DescribeSpotFleetRequestHistory	100	5
	AssociateEnclaveCertificateIamRole	10	1
	DisassociateEnclaveCertificateIamRole	10	1
	GetAssociatedEnclaveCertificateIamRoles	10	1

API 작업 카테고리	작업	버킷 최대 용량	버킷 리필 속도
	GetConsoleScreenshot	계정당 5개 인스턴스당 2개	계정당 5개 인스턴스당 1개

리소스 토큰 버킷 크기 및 리필 비율

다음 표에는 리소스 속도 제한을 사용하는 API 작업에 대한 리소스 토큰 버킷 크기 및 리필 비율이 나와 있습니다.

API 작업	버킷 최대 용량	버킷 리필 비율
RunInstances	1000	2
TerminateInstances	1000	20
StartInstances	1000	2
StopInstances	1000	20

API 스로틀링 모니터링

CloudWatch Amazon을 사용하여 Amazon EC2 API 호출을 모니터링하고 API 제한과 관련된 지표를 수집 및 추적할 수 있습니다. API 제한 한도에 가까워지면 경고하는 경보를 생성할 수도 있습니다. 자세한 정보는 Amazon을 사용하여 Amazon EC2 API 요청을 모니터링합니다. CloudWatch 을 참조하세요.

재시도 및 지수 백오프

애플리케이션에서 API 요청을 재시도해야 할 수 있습니다. 예:

- 리소스 상태가 업데이트되었는지 확인하려면
- 많은 수의 리소스 (예: 모든 볼륨) 를 열거하려면
- 서버 오류 (5xx) 또는 병목 오류로 인해 요청이 실패한 후 요청을 재시도하려면

하지만 클라이언트 오류 (4xx) 의 경우 요청을 다시 시도하기 전에 문제를 해결하도록 요청을 수정해야 합니다.

리소스 상태 변경

상태 업데이트를 확인하기 위한 폴링을 시작하기 전에 요청이 완료될 수 있도록 시간을 주세요. 예를 들어 인스턴스가 활성 상태인지 확인하기 전에 몇 분 정도 기다려 보십시오. 폴링을 시작할 때는 연속적인 요청 사이에 적절한 휴면 간격을 두어 API 요청 비율을 낮추십시오. 최상의 결과를 얻으려면 휴면 간격을 늘리거나 가변적으로 사용하세요.

또는 EventBridge Amazon을 사용하여 일부 리소스의 상태를 알려줄 수도 있습니다. 예를 들어, EC2 인스턴스 상태 변경 알림 이벤트를 사용하여 인스턴스의 상태 변경을 알릴 수 있습니다. 자세한 내용은 Amazon EC2를 사용한 자동화를 참조하십시오. EventBridge

재시도

API 요청을 폴링하거나 재시도해야 하는 경우 지수 백오프 알고리즘을 사용하여 API 호출 간의 휴면 간격을 계산하는 것이 좋습니다. 지수 백오프의 기본 개념은 오류 응답이 연이어 나올 때마다 재시도 간 대기 시간을 점진적으로 늘리는 것입니다. 최대 지연 간격과 최대 재시도 횟수를 구현해야 합니다. 지터 (무작위 지연) 를 사용하여 연속적인 충돌을 방지할 수도 있습니다. 자세한 내용은 시간 제한, 재시도 및 지터를 사용한 백오프를 참조하세요.

각 SDK는 자동 재시도 로직을 구현합니다. AWS 자세한 내용은 AWS SDK 및 도구 참조 가이드의 재시도 동작을 참조하십시오.

한도 증가 요청

에 대한 API 스로틀링 한도 증가를 요청할 수 있습니다. AWS 계정

이 기능에 대한 액세스를 요청하려면

1. AWS Support 센터를 여세요.
2. 사례 생성을 선택합니다.
3. 계정 및 결제 지원을 선택합니다.
4. 서비스에서 일반 정보 및 시작하기를 선택합니다.
5. 카테고리에서 사용 AWS 및 서비스를 선택합니다.
6. 다음 단계: 추가 정보를 선택합니다
7. 제목에 **`Request an increase in my Amazon EC2 API throttling limits`**을 입력합니다.

8. 설명에 **Please increase the API throttling limits for my account. Related page: https://docs.aws.amazon.com/AWSEC2/latest/APIReference/throttling.html**를 입력합니다. 또한 다음 정보를 포함합니다.

   - 사용 사례에 대한 설명.
   - 증가가 필요한 지역.
   - 최대 전송률 조절 또는 사용량이 발생한 1시간 단위 (UTC 기준) (새 스로틀링 한도 계산 기준)
9. 다음 단계: 지금 해결하거나 문의하기를 선택합니다.
10. 연락처 탭에서 원하는 연락처 언어와 연락 방법을 선택합니다.
11. 제출을 선택합니다.

를 사용하여 Amazon EC2 리소스를 생성합니다. AWS CLI

명령줄 셸의 AWS Command Line Interface (AWS CLI) 를 사용하여 Amazon EC2 리소스를 생성하고 관리할 수 있습니다. AWS CLI 에서는 Amazon EC2와 같은 API에 AWS 서비스직접 액세스할 수 있습니다.

Amazon EC2용 명령의 구문 및 예제는 명령 참조의 AWS CLI ec2를 참조하십시오. 깃허브의 aws-cli/awscli/examples/ec2에서도 이러한 예제를 찾을 수 있습니다.

에 대해 자세히 알아보십시오. AWS CLI

에 대해 자세히 AWS CLI알아보려면 다음 리소스를 참조하십시오.

- AWS Command Line Interface
- AWS Command Line Interface 버전 2 사용 설명서
- AWS Command Line Interface 버전 1 사용 설명서

를 사용하여 Amazon EC2 리소스를 생성합니다. AWS CloudFormation

Amazon EC2는 리소스와 AWS CloudFormation인프라를 생성하고 관리하는 데 소요되는 시간을 줄일 수 있도록 AWS 리소스를 모델링하고 설정하는 데 도움이 되는 서비스인 와 통합되어 있습니다. 필요한 리소스 (예: 인스턴스 및 서브넷) 를 설명하는 템플릿을 만들고 해당 AWS 리소스를 AWS CloudFormation 프로비저닝 및 구성합니다.

를 사용하면 AWS CloudFormation템플릿을 재사용하여 Amazon EC2 리소스를 일관되고 반복적으로 설정할 수 있습니다. 리소스를 한 번 설명한 다음 여러 AWS 계정 지역 및 지역에서 동일한 리소스를 반복해서 프로비저닝하십시오.

아마존 EC2 및 템플릿 AWS CloudFormation

Amazon EC2 및 관련 서비스를 위한 리소스를 프로비저닝하고 구성하려면 템플릿을AWS CloudFormation 이해해야 합니다. 템플릿은 JSON 또는 YAML로 서식 지정된 텍스트 파일입니다. 이 템플릿은 AWS CloudFormation 스택에 프로비저닝할 리소스를 설명합니다. JSON이나 YAML에 익숙하지 않은 경우 AWS CloudFormation Designer를 사용하여 템플릿을 시작하는 데 도움을 받을 수 있습니다. AWS CloudFormation 자세한 내용은 디자이너란 무엇입니까? 를 참조하십시오. AWS CloudFormation AWS CloudFormation 사용 설명서에서.

Amazon EC2용 리소스

컴퓨팅 리소스

- AWS::EC2::CapacityReservation
- AWS::EC2::CapacityReservation플릿
- AWS::EC2::EC2Fleet
- AWS::EC2::EC2Fleet
- AWS::EC2::Host
- AWS::EC2::Instance
- AWS::EC2::InstanceConnect엔드포인트
- AWS::EC2::LaunchTemplate
- AWS::EC2::PlacementGroup

- AWS::EC2::SpotFleet

네트워킹 리소스

- AWS::EC2::CarrierGateway
- AWS::EC2::ClientVpnAuthorizationRule
- AWS::EC2::ClientVpn엔드포인트
- AWS::EC2::ClientVpn경로
- AWS::EC2::ClientVpnTargetNetworkAssociation
- AWS::EC2::CustomerGateway
- AWS::EC2::DHCPOptions
- AWS::EC2::EgressOnlyInternetGateway
- AWS::EC2::EIP
- AWS::EC2::EIPAssociation
- AWS::EC2::FlowLog
- AWS::EC2::GatewayRouteTableAssociation
- AWS::EC2::InternetGateway
- AWS::EC2::IPAM
- AWS::EC2::IPAMAllocation
- AWS::EC2::IPAMPool
- AWS::EC2::IPAMPoolCidr
- AWS::EC2::IPAMResourceDiscovery
- AWS::EC2::IPAMResourceDiscoveryAssociation
- AWS::EC2::IPAMScope
- AWS::EC2::LocalGateway노선
- AWS::EC2::LocalGatewayRouteTable
- AWS::EC2::LocalGatewayRouteTableVirtualInterfaceGroupAssociation
- AWS::EC2::LocalGatewayRouteTableVPC 협회
- AWS::EC2::NatGateway
- AWS::EC2::NetworkInterface
- AWS::EC2::NetworkInsightsAccessScope

- AWS::EC2::NetworkInsightsAccessScopeAnalysis
- AWS::EC2::NetworkInsights분석
- AWS::EC2::NetworkInsights경로
- AWS::EC2::NetworkInterface첨부파일
- AWS::EC2::NetworkInterface허가
- AWS::EC2::NetworkPerformanceMetricSubscription
- AWS::EC2::PrefixList
- AWS::EC2::Route
- AWS::EC2::RouteTable
- AWS::EC2::Subnet
- AWS::EC2::SubnetCidr차단
- AWS::EC2::SubnetNetworkAclAssociation
- AWS::EC2::SubnetRouteTableAssociation
- AWS::EC2::TrafficMirror필터
- AWS::EC2::TrafficMirrorFilterRule
- AWS::EC2::TrafficMirror세션
- AWS::EC2::TrafficMirror타겟
- AWS::EC2::TransitGateway
- AWS::EC2::TransitGateway첨부파일
- AWS::EC2::TransitGatewayConnect
- AWS::EC2::TransitGatewayMulticastDomain
- AWS::EC2::TransitGatewayMulticastDomainAssociation
- AWS::EC2::TransitGatewayMulticastGroupMember
- AWS::EC2::TransitGatewayMulticastGroupSource
- AWS::EC2::TransitGatewayPeeringAttachment
- AWS::EC2::TransitGateway루트
- AWS::EC2::TransitGatewayRouteTable
- AWS::EC2::TransitGatewayRouteTableAssociation
- AWS::EC2::TransitGatewayRouteTablePropagation
- AWS::EC2::TransitGatewayVpcAttachment

- AWS::EC2::VPC
- AWS::EC2::VPCCidrBlock
- AWS: :EC2: :VPCDHCP OptionsAssociation
- AWS::EC2::VPCEndpoint
- AWS::EC2::VPCEndpointConnectionNotification
- AWS::EC2::VPCEndpointService
- AWS::EC2::VPCEndpointServicePermissions
- AWS::EC2::VPCGatewayAttachment
- AWS::EC2::VPCPeeringConnection
- AWS::EC2::VPNConnection
- AWS: :EC2: :VPN ConnectionRoute
- AWS::EC2::VPNGateway
- AWS: :EC2: :VPN GatewayRoutePropagation

보안 리소스

- AWS::EC2::KeyPair
- AWS::EC2::NetworkAcl
- AWS::EC2::NetworkAcl엔트리
- AWS::EC2::SecurityGroup
- AWS::EC2::SecurityGroup출구
- AWS::EC2::SecurityGroup인그레스
- AWS::EC2::VerifiedAccess엔드포인트
- AWS::EC2::VerifiedAccess그룹
- AWS::EC2::VerifiedAccess인스턴스
- AWS::EC2::VerifiedAccessTrustProvider

스토리지 리소스

- AWS::EC2::SnapshotBlockPublicAccess
- AWS::EC2::Volume
- AWS::EC2::VolumeAttachment

자세히 알아보기 AWS CloudFormation

자세히 AWS CloudFormation알아보려면 다음 리소스를 참조하십시오.

- AWS CloudFormation
- AWS CloudFormation 사용 설명서

SDK를 사용하여 Amazon EC2 리소스 생성 AWS

AWS 널리 사용되는 여러 프로그래밍 언어를 위한 소프트웨어 개발 키트 (SDK) 를 제공합니다. SDK는 다음을 제공하여 개발을 더욱 효율적으로 만듭니다.

- 애플리케이션에 통합할 수 있는 사전 빌드된 구성 요소 및 라이브러리
- 언어별 도구 (예: 컴파일러 및 디버거)
- 서비스 요청의 암호화 서명
- 요청 재시도
- 오류 응답 처리

Amazon EC2 API의 코드 예제

에서 제공하는 코드 예제는 API를 사용하고 특정 작업을 수행하는 방법을 AWS 보여줍니다. Amazon EC2 API의 예는 Amazon EC2의 코드 예제를 참조하십시오. 추가 예제는 AWS SDK 또는 aws-doc-sdk-examplesgithub의 코드 예제 찾기를 참조하십시오.

SDK에 대해 자세히 알아보세요. AWS

AWS SDK에 대해 자세히 알아보려면 다음 리소스를 참조하십시오.

- AWS SDK 및 도구 참조 가이드
- 구축을 위한 도구 AWS
- SDK란 무엇인가요?

Amazon EC2용 저수준 API

Amazon EC2용 하위 수준 API는 Amazon EC2의 프로토콜 수준 인터페이스입니다. 하위 수준 API를 사용할 때는 모든 HTTPS 요청의 형식을 올바르게 지정하고 모든 요청에 유효한 디지털 서명을 추가해야 합니다. 자세한 내용은 Amazon EC2 API 참조의 Amazon EC2 API에 요청하기를 참조하십시오. 또는 사용자를 대신하여 요청을 구성하고 서명하는 AWS SDK를 사용할 수도 있습니다. 자세한 정보는 SDK 사용 AWS 을 참조하세요.

Amazon EC2 API는 여러 서비스에 대한 작업 및 데이터 유형으로 구성되어 있습니다. 각 서비스에 대한 작업을 보려면 Amazon EC2 API 참조의 다음 페이지를 참조하십시오.

- AWS Client VPN actions
- Amazon EBS 작업
- 아마존 EC2 액션
- AWS Network Manager actions
- AWS 니트로 엔클레이브 액션
- AWS Outposts actions
- AWS PrivateLink actions
- 휴지통 조치
- AWS Site-to-Site VPN actions
- AWS Transit Gateway actions
- AWS Verified Access actions
- VM 가져오기/내보내기 작업
- 아마존 VPC 액션
- 아마존 VPC IPAM 액션
- AWS Wavelength actions

콘솔 투 코드를 사용하여 콘솔 작업을 위한 코드 생성

콘솔 투 코드는 Amazon EC2의 평가판 릴리스이기 때문에 변경될 수 있습니다. 미국 동부(버지니아 북부) 리전에서만 사용 가능합니다.

콘솔은 리소스를 생성하고 프로토타입을 테스트하기 위한 안내 경로를 제공합니다. 대규모로 동일한 리소스를 생성하려면 자동화 코드가 필요합니다. 콘솔 투 코드는 자동화 코드를 시작하는 데 도움이 되는 Amazon EC2 콘솔의 기능입니다. 콘솔 투 코드는 기본값과 호환 가능한 파라미터를 포함하여 콘솔 작업을 기록합니다. 그런 다음 제너레이티브 AI를 사용하여 원하는 작업에 대해 선호하는 infrastructure-as-code (IaC) 형식으로 코드를 제안합니다. 코드를 시작점으로 사용하여 특정 사용 사례에 맞게 프로덕션에 바로 사용할 수 있도록 사용자 지정할 수 있습니다.

콘솔 투 코드 사용에 따른 추가 비용은 없습니다.

작동 방식

콘솔 투 코드는 다음과 같이 자동화 코드를 시작하는 데 도움이 됩니다.

1. 콘솔에서 인스턴스 시작 또는 세부 모니터링 활성화와 같은 작업을 수행합니다.
2. 콘솔 투 코드는 콘솔에서 제공하는 모든 기본 설정과 호환 가능한 파라미터를 포함하여 모든 작업을 기록합니다.
3. 자동화 스크립트에서 사용하려는 작업을 선택합니다. 작업은 변경 작업, 읽기 전용(변경 불가) 작업 또는 두 작업 유형 모두일 수 있습니다.
4. 콘솔-코드는 원하는 infrastructure-as-code (IaC) 형식 (예:) 으로 코드를 생성합니다. TypeScript
5. 코드 개발 도구에서 사용할 코드를 복사하거나 다운로드하여 공유할 수 있습니다.
6. 그런 다음 해당 코드를 자동화 스크립트의 시작점으로 사용합니다. 코드가 의도를 충족하고 파라미터가 리소스를 예상대로 구성하는지 검증해야 합니다. 사용 사례에 맞게 프로덕션에 바로 사용할 수 있도록 코드를 사용자 지정해야 합니다. 코드에 만족하면 자동화 스크립트에 사용할 수 있습니다.

Amazon EC2 콘솔에서 콘솔 투 코드를 사용하는 방법에 대한 지침은 콘솔 투 코드 사용 섹션을 참조하세요.

제한 사항

다음은 콘솔 투 코드를 사용할 때 적용되는 제한 사항입니다.

지원되는 리전

현재 미국 동부(버지니아 북부) 리전에서만 사용할 수 있습니다.

지원되는 코드 형식

콘솔-코드는 현재 다음과 같은 코드 형식으로 생성 infrastructure-as-code (IAC) 할 수 있습니다.

- CDK Java
- CDK Python
- CDK TypeScript
- CloudFormation JSON
- CloudFormation YAML

유지된 작업

- 현재 세션: 현재 세션 중 수행된 작업만 기록된 동작 표에 표시됩니다. 이전 세션에서 수행한 동작은 유지되지 않습니다.
- 브라우저 새로 고침: 브라우저 탭을 새로 고치면 기록된 동작이 손실됩니다.
- 탭 격리: 기록된 동작 표는 동작이 수행된 브라우저 탭에만 해당됩니다. 한 탭에서 수행된 동작은 다른 탭의 기록된 동작 표에 표시되지 않습니다.

기록된 동작 표

다음 표에서는 콘솔 투 코드 콘솔의 기록된 동작 테이블에 있는 열을 나열하고 설명합니다.

열 제목	설명
콘솔 페이지	동작이 수행된 콘솔 페이지입니다.
Operation	API 작업입니다.

열 제목	설명
Type	동작의 유형입니다. • 변형 - 리소스를 생성, 수정 또는 삭제하는 API 작업입니다. • 읽기 전용 - 리소스에 대한 데이터를 검색하는 API 작업(일반적으로 모든 `Describe*` 작업)입니다.
CLI 명령	파라미터와 값을 포함하여 수행된 동작에 대한 세부 정보입니다.
생성 시간	동작이 수행된 시간입니다.

콘솔 투 코드 사용

다음 지침에 따라 Amazon EC2 콘솔에서 콘솔 투 코드를 사용하여 코드를 생성합니다.

이러한 단계의 애니메이션을 보려면 애니메이션 보기: Amazon EC2 콘솔에서 콘솔 투 코드를 사용하여 코드 생성 단원을 참조하세요.

콘솔 투 코드를 사용하여 코드 생성

1. https://console.aws.amazon.com/ec2/home?region=us-east-1에서 미국 동부(버지니아 북부) 리전의 Amazon EC2 콘솔을 엽니다.

   > **Note**
   >
   > 콘솔 투 코드는 평가판 릴리스이며 현재 미국 동부(버지니아 북부) 리전에서만 사용할 수 있습니다. 이 리전에서 수행된 작업만 기록됩니다.

2. 콘솔을 사용하여 리소스를 생성하고 프로토타입을 테스트합니다. 예를 들어, 콘솔을 사용하여 인스턴스를 구성 및 시작하고 세부 모니터링을 활성화합니다.

   콘솔 투 코드는 사용자의 모든 동작을 기록합니다.

3. 왼쪽 탐색 창에서 콘솔-투-코드를 선택합니다.
4. 기록된 동작 표에서 기록된 동작을 검토하고 코드 생성에 포함할 동작을 결정합니다.

   • 검색 필드를 사용하여 특정 콘솔 페이지 또는 동작을 기준으로 표를 필터링합니다. 입력을 시작하면 표가 필터링됩니다.

- 유형 드롭다운을 사용하여 모든 동작, 변형 동작 또는 읽기 전용 동작을 기준으로 필터링합니다.

> **Note**
>
> 현재 세션 중에 수행한 동작만 나열됩니다. 자세한 정보는 유지된 작업을 참조하세요.

5. 코드를 생성해야 하는 각 동작 옆의 확인란을 선택합니다.

> **Note**
>
> 한 번에 최대 5개의 동작을 선택할 수 있습니다.

6. {code} 코드 생성 버튼을 선택합니다.

   버튼 레이블의 기본값은 마지막으로 선택한 코드 형식입니다. 다른 코드 형식을 선택하려면 버튼 옆의 화살표를 선택합니다.

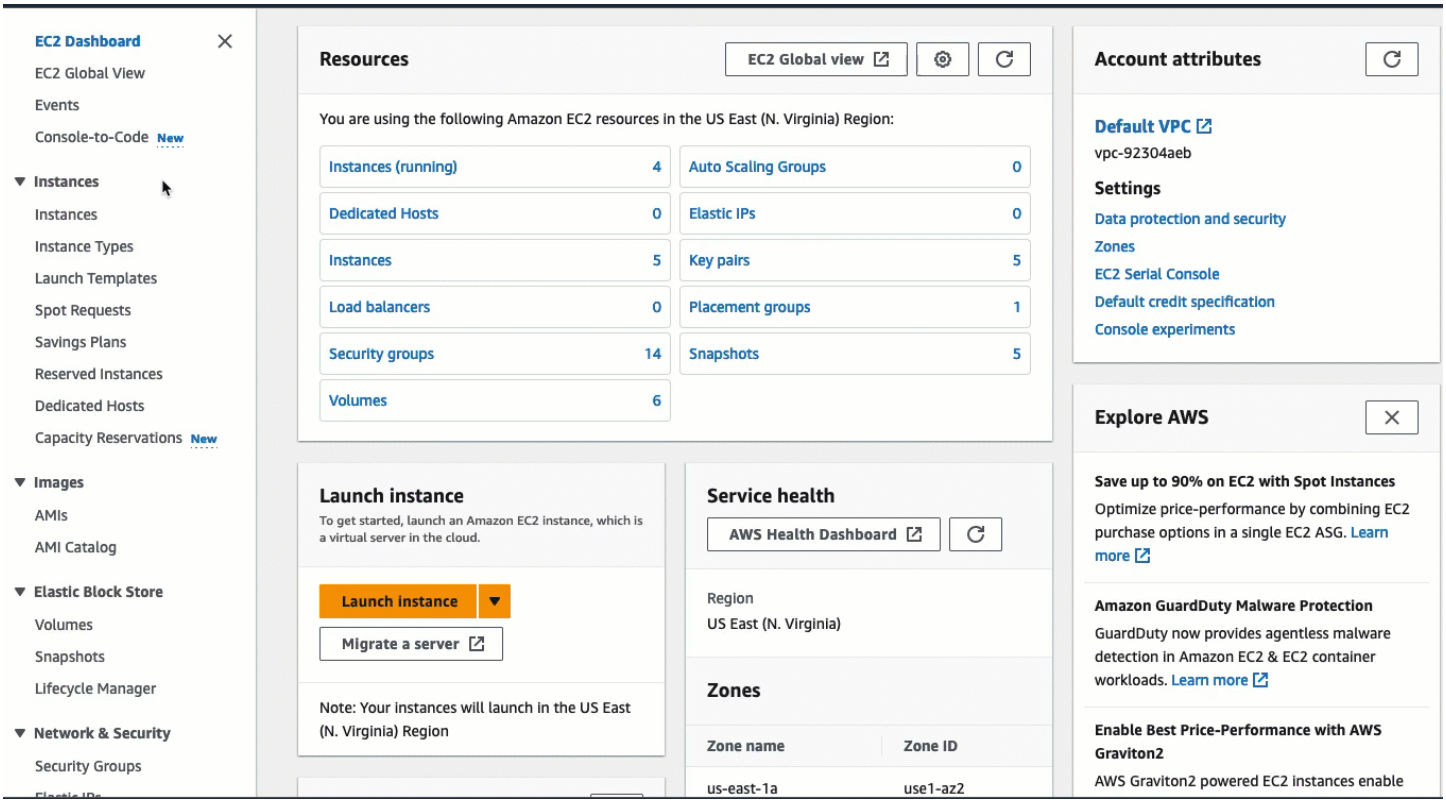
7. 코드 리뷰에서 복사를 선택하여 개발 도구에서 사용할 코드를 복사하거나 다운로드를 선택하여 공유할 파일을 다운로드합니다.

8. 코드를 시작점으로 사용하세요. infrastructure-as-code 특정 사용 사례에 맞게 프로덕션에 바로 사용할 수 있도록 코드를 사용자 지정해야 합니다.

> **Note**
>
> 코드를 제작할 준비가 되지 않은 경우 개선 방법에 대한 피드백을 제공해 주세요 (다음 9단계 참조). AWS Support 생성된 코드나 사용자 지정 코드 개발에는 도움을 드릴 수 없습니다.

9. (선택 사항) 좋아요 또는 싫어요를 선택하여 콘솔 투 코드가 도움이 되었는지 여부를 알려줍니다. 싫어요를 선택한 경우 피드백 제공을 선택하여 더 나은 도움을 드릴 수 있도록 코드를 개선할 수 있는 방법을 알려주세요.

애니메이션 보기: Amazon EC2 콘솔에서 콘솔 투 코드를 사용하여 코드 생성



SDK를 사용하는 Amazon EC2의 코드 예제 AWS

다음 코드 예제는 AWS 소프트웨어 개발 키트 (SDK) 와 함께 Amazon EC2를 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 안내서 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

시작하기

Hello Amazon EC2

다음 코드 예제는 Amazon EC2 사용을 시작하는 방법을 보여줍니다.

.NET

AWS SDK for .NET

> **Note**
>
> 자세한 내용은 에서 확인할 수 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
namespace EC2Actions;

public class HelloEc2
{
    /// <summary>
    /// HelloEc2 lists the existing security groups for the default users.
```

```
    /// </summary>
    /// <param name="args">Command line arguments</param>
    /// <returns>A Task object.</returns>
    static async Task Main(string[] args)
    {
        // Set up dependency injection for Amazon Elastic Compute Cloud (Amazon
 EC2).
        using var host =
 Microsoft.Extensions.Hosting.Host.CreateDefaultBuilder(args)
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonEC2>()
                .AddTransient<EC2Wrapper>()
            )
            .Build();

        // Now the client is available for injection.
        var ec2Client = host.Services.GetRequiredService<IAmazonEC2>();

        var request = new DescribeSecurityGroupsRequest
        {
            MaxResults = 10,
        };


        // Retrieve information about up to 10 Amazon EC2 security groups.
        var response = await ec2Client.DescribeSecurityGroupsAsync(request);

        // Now print the security groups returned by the call to
        // DescribeSecurityGroupsAsync.
        Console.WriteLine("Security Groups:");
        response.SecurityGroups.ForEach(group =>
        {
            Console.WriteLine($"Security group: {group.GroupName} ID:
 {group.GroupId}");
        });
    }
}
```

- API 세부 정보는 AWS SDK for .NET API DescribeSecurityGroups참조를 참조하십시오.

C++

SDK for C++

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

C MakeLists .txt CMake 파일의 코드입니다.

```
# Set the minimum required version of CMake for this project.
cmake_minimum_required(VERSION 3.13)

# Set the AWS service components used by this project.
set(SERVICE_COMPONENTS ec2)

# Set this project's name.
project("hello_ec2")

# Set the C++ standard to use to build this target.
# At least C++ 11 is required for the AWS SDK for C++.
set(CMAKE_CXX_STANDARD 11)

# Use the MSVC variable to determine if this is a Windows build.
set(WINDOWS_BUILD ${MSVC})

if (WINDOWS_BUILD) # Set the location where CMake can find the installed
 libraries for the AWS SDK.
    string(REPLACE ";" "/aws-cpp-sdk-all;" SYSTEM_MODULE_PATH
 "${CMAKE_SYSTEM_PREFIX_PATH}/aws-cpp-sdk-all")
    list(APPEND CMAKE_PREFIX_PATH ${SYSTEM_MODULE_PATH})
endif ()

# Find the AWS SDK for C++ package.
find_package(AWSSDK REQUIRED COMPONENTS ${SERVICE_COMPONENTS})

if (WINDOWS_BUILD AND AWSSDK_INSTALL_AS_SHARED_LIBS)
     # Copy relevant AWS SDK for C++ libraries into the current binary directory
 for running and debugging.
```

```
    # set(BIN_SUB_DIR "/Debug") # If you are building from the command line, you
 may need to uncomment this
                                    # and set the proper subdirectory to the
 executables' location.

    AWSSDK_CPY_DYN_LIBS(SERVICE_COMPONENTS ""
 ${CMAKE_CURRENT_BINARY_DIR}${BIN_SUB_DIR})
endif ()

add_executable(${PROJECT_NAME}
        hello_ec2.cpp)

target_link_libraries(${PROJECT_NAME}
        ${AWSSDK_LINK_LIBRARIES})
```

hello_ec2.cpp 소스 파일의 코드입니다.

```
#include <aws/core/Aws.h>
#include <aws/ec2/EC2Client.h>
#include <aws/ec2/model/DescribeInstancesRequest.h>
#include <iomanip>
#include <iostream>

/*
 *  A "Hello EC2" starter application which initializes an Amazon Elastic Compute
 Cloud (Amazon EC2) client and describes
 *  the Amazon EC2 instances.
 *
 *  main function
 *
 *  Usage: 'hello_ec2'
 *
 */

int main(int argc, char **argv) {
    Aws::SDKOptions options;
    // Optionally change the log level for debugging.
//   options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
    int result = 0;
    {
        Aws::Client::ClientConfiguration clientConfig;
```

```
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        Aws::EC2::EC2Client ec2Client(clientConfig);
        Aws::EC2::Model::DescribeInstancesRequest request;
        bool header = false;
        bool done = false;
        while (!done) {
            auto outcome = ec2Client.DescribeInstances(request);
            if (outcome.IsSuccess()) {
                if (!header) {
                    std::cout << std::left <<
                              std::setw(48) << "Name" <<
                              std::setw(20) << "ID" <<
                              std::setw(25) << "Ami" <<
                              std::setw(15) << "Type" <<
                              std::setw(15) << "State" <<
                              std::setw(15) << "Monitoring" << std::endl;
                    header = true;
                }

                const std::vector<Aws::EC2::Model::Reservation> &reservations =
                        outcome.GetResult().GetReservations();

                for (const auto &reservation: reservations) {
                    const std::vector<Aws::EC2::Model::Instance> &instances =
                            reservation.GetInstances();
                    for (const auto &instance: instances) {
                        Aws::String instanceStateString =

Aws::EC2::Model::InstanceStateNameMapper::GetNameForInstanceStateName(
                                        instance.GetState().GetName());

                        Aws::String typeString =

Aws::EC2::Model::InstanceTypeMapper::GetNameForInstanceType(
                                        instance.GetInstanceType());

                        Aws::String monitorString =

Aws::EC2::Model::MonitoringStateMapper::GetNameForMonitoringState(
                                        instance.GetMonitoring().GetState());
                        Aws::String name = "Unknown";
```

```
                        const std::vector<Aws::EC2::Model::Tag> &tags =
 instance.GetTags();
                        auto nameIter = std::find_if(tags.cbegin(), tags.cend(),
                                                     [](const
 Aws::EC2::Model::Tag &tag) {
                                                         return tag.GetKey() ==
 "Name";
                                                     });
                        if (nameIter != tags.cend()) {
                            name = nameIter->GetValue();
                        }
                        std::cout <<
                                  std::setw(48) << name <<
                                  std::setw(20) << instance.GetInstanceId() <<
                                  std::setw(25) << instance.GetImageId() <<
                                  std::setw(15) << typeString <<
                                  std::setw(15) << instanceStateString <<
                                  std::setw(15) << monitorString << std::endl;
                    }
                }

                if (!outcome.GetResult().GetNextToken().empty()) {
                    request.SetNextToken(outcome.GetResult().GetNextToken());
                } else {
                    done = true;
                }
            } else {
                std::cerr << "Failed to describe EC2 instances:" <<
                          outcome.GetError().GetMessage() << std::endl;
                result = 1;
                break;
            }
        }
    }


    Aws::ShutdownAPI(options); // Should only be called once.
    return result;
}
```

- API에 대한 자세한 내용은 API 레퍼런스를 참조하십시오 DescribeSecurityGroups.AWS SDK for C++

Java

SDK for Java 2.x

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
   public static void describeSecurityGroups(Ec2Client ec2, String groupId) {
       try {
           DescribeSecurityGroupsRequest request =
DescribeSecurityGroupsRequest.builder()
               .groupIds(groupId)
               .build();

           // Use a paginator.
           DescribeSecurityGroupsIterable listGroups =
ec2.describeSecurityGroupsPaginator(request);
           listGroups.stream()
               .flatMap(r -> r.securityGroups().stream())
               .forEach(group -> System.out
                   .println(" Group id: " +group.groupId() + " group name = " +
group.groupName()));

       } catch (Ec2Exception e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

- API 세부 정보는 AWS SDK for Java 2.x API DescribeSecurityGroups참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { DescribeSecurityGroupsCommand } from "@aws-sdk/client-ec2";

import { client } from "./libs/client.js";

// Call DescribeSecurityGroups and display the result.
export const main = async () => {
  try {
    const { SecurityGroups } = await client.send(
      new DescribeSecurityGroupsCommand({}),
    );

    const securityGroupList = SecurityGroups.slice(0, 9)
      .map((sg) => ` • ${sg.GroupId}: ${sg.GroupName}`)
      .join("\n");

    console.log(
      "Hello, Amazon EC2! Let's list up to 10 of your security groups:",
    );
    console.log(securityGroupList);
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API DescribeSecurityGroups참조를 참조하십시오.

Kotlin

SDK for Kotlin

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
suspend fun describeEC2SecurityGroups(groupId: String) {
    val request =
        DescribeSecurityGroupsRequest {
            groupIds = listOf(groupId)
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->

        val response = ec2.describeSecurityGroups(request)
        response.securityGroups?.forEach { group ->
            println("Found Security Group with id ${group.groupId}, vpc id
 ${group.vpcId} and description ${group.description}")
        }
    }
}
```

- API 세부 정보는 Kotlin API용AWS SDK 레퍼런스를 참조하세요 DescribeSecurityGroups.

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 여기에서 확인할 수 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import boto3
```

```
def hello_ec2(ec2_resource):
    """
    Use the AWS SDK for Python (Boto3) to create an Amazon Elastic Compute Cloud
    (Amazon EC2) resource and list the security groups in your account.
    This example uses the default settings specified in your shared credentials
    and config files.

    :param ec2_resource: A Boto3 EC2 ServiceResource object. This object is a
 high-level
                         resource that wraps the low-level EC2 service API.
    """
    print("Hello, Amazon EC2! Let's list up to 10 of your security groups:")
    for sg in ec2_resource.security_groups.limit(10):
        print(f"\t{sg.id}: {sg.group_name}")


if __name__ == "__main__":
    hello_ec2(boto3.resource("ec2"))
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 DescribeSecurityGroups.

코드 예시

- SDK를 사용하는 Amazon EC2용 작업 AWS
  - AWS SDK 또는 AcceptVpcPeeringConnection CLI와 함께 사용
  - AWS SDK 또는 AllocateAddress CLI와 함께 사용
  - AWS SDK 또는 AllocateHosts CLI와 함께 사용
  - AWS SDK 또는 AssignPrivateIpAddresses CLI와 함께 사용
  - AWS SDK 또는 AssociateAddress CLI와 함께 사용
  - AWS SDK 또는 AssociateDhcpOptions CLI와 함께 사용
  - AWS SDK 또는 AssociateRouteTable CLI와 함께 사용
  - AWS SDK 또는 AttachInternetGateway CLI와 함께 사용
  - AWS SDK 또는 AttachNetworkInterface CLI와 함께 사용
  - AWS SDK 또는 AttachVolume CLI와 함께 사용

- AWS SDK 또는 AttachVpnGateway CLI와 함께 사용
- AWS SDK 또는 AuthorizeSecurityGroupEgress CLI와 함께 사용
- AWS SDK 또는 AuthorizeSecurityGroupIngress CLI와 함께 사용
- AWS SDK 또는 CancelCapacityReservation CLI와 함께 사용
- AWS SDK 또는 CancelImportTask CLI와 함께 사용
- AWS SDK 또는 CancelSpotFleetRequests CLI와 함께 사용
- AWS SDK 또는 CancelSpotInstanceRequests CLI와 함께 사용
- AWS SDK 또는 ConfirmProductInstance CLI와 함께 사용
- AWS SDK 또는 CopyImage CLI와 함께 사용
- AWS SDK 또는 CopySnapshot CLI와 함께 사용
- AWS SDK 또는 CreateCapacityReservation CLI와 함께 사용
- AWS SDK 또는 CreateCustomerGateway CLI와 함께 사용
- AWS SDK 또는 CreateDhcpOptions CLI와 함께 사용
- AWS SDK 또는 CreateFlowLogs CLI와 함께 사용
- AWS SDK 또는 CreateImage CLI와 함께 사용
- AWS SDK 또는 CreateInstanceExportTask CLI와 함께 사용
- AWS SDK 또는 CreateInternetGateway CLI와 함께 사용
- AWS SDK 또는 CreateKeyPair CLI와 함께 사용
- AWS SDK 또는 CreateLaunchTemplate CLI와 함께 사용
- AWS SDK 또는 CreateNetworkAcl CLI와 함께 사용
- AWS SDK 또는 CreateNetworkAclEntry CLI와 함께 사용
- AWS SDK 또는 CreateNetworkInterface CLI와 함께 사용
- AWS SDK 또는 CreatePlacementGroup CLI와 함께 사용
- AWS SDK 또는 CreateRoute CLI와 함께 사용
- AWS SDK 또는 CreateRouteTable CLI와 함께 사용
- AWS SDK 또는 CreateSecurityGroup CLI와 함께 사용
- AWS SDK 또는 CreateSnapshot CLI와 함께 사용
- AWS SDK 또는 CreateSpotDatafeedSubscription CLI와 함께 사용
- AWS SDK 또는 CreateSubnet CLI와 함께 사용
- AWS SDK 또는 CreateTags CLI와 함께 사용

- AWS SDK 또는 CreateVolume CLI와 함께 사용
- AWS SDK 또는 CreateVpc CLI와 함께 사용
- AWS SDK 또는 CreateVpcEndpoint CLI와 함께 사용
- AWS SDK 또는 CreateVpnConnection CLI와 함께 사용
- AWS SDK 또는 CreateVpnConnectionRoute CLI와 함께 사용
- AWS SDK 또는 CreateVpnGateway CLI와 함께 사용
- AWS SDK 또는 DeleteCustomerGateway CLI와 함께 사용
- AWS SDK 또는 DeleteDhcpOptions CLI와 함께 사용
- AWS SDK 또는 DeleteFlowLogs CLI와 함께 사용
- AWS SDK 또는 DeleteInternetGateway CLI와 함께 사용
- AWS SDK 또는 DeleteKeyPair CLI와 함께 사용
- AWS SDK 또는 DeleteLaunchTemplate CLI와 함께 사용
- AWS SDK 또는 DeleteNetworkAcl CLI와 함께 사용
- AWS SDK 또는 DeleteNetworkAclEntry CLI와 함께 사용
- AWS SDK 또는 DeleteNetworkInterface CLI와 함께 사용
- AWS SDK 또는 DeletePlacementGroup CLI와 함께 사용
- AWS SDK 또는 DeleteRoute CLI와 함께 사용
- AWS SDK 또는 DeleteRouteTable CLI와 함께 사용
- AWS SDK 또는 DeleteSecurityGroup CLI와 함께 사용
- AWS SDK 또는 DeleteSnapshot CLI와 함께 사용
- AWS SDK 또는 DeleteSpotDatafeedSubscription CLI와 함께 사용
- AWS SDK 또는 DeleteSubnet CLI와 함께 사용
- AWS SDK 또는 DeleteTags CLI와 함께 사용
- AWS SDK 또는 DeleteVolume CLI와 함께 사용
- AWS SDK 또는 DeleteVpc CLI와 함께 사용
- AWS SDK 또는 DeleteVpnConnection CLI와 함께 사용
- AWS SDK 또는 DeleteVpnConnectionRoute CLI와 함께 사용
- AWS SDK 또는 DeleteVpnGateway CLI와 함께 사용
- AWS SDK 또는 DeregisterImage CLI와 함께 사용
- AWS SDK 또는 DescribeAccountAttributes CLI와 함께 사용

- AWS SDK 또는 DescribeAddresses CLI와 함께 사용
- AWS SDK 또는 DescribeAvailabilityZones CLI와 함께 사용
- AWS SDK 또는 DescribeBundleTasks CLI와 함께 사용
- AWS SDK 또는 DescribeCapacityReservations CLI와 함께 사용
- AWS SDK 또는 DescribeCustomerGateways CLI와 함께 사용
- AWS SDK 또는 DescribeDhcpOptions CLI와 함께 사용
- AWS SDK 또는 DescribeFlowLogs CLI와 함께 사용
- AWS SDK 또는 DescribeHostReservationOfferings CLI와 함께 사용
- AWS SDK 또는 DescribeHosts CLI와 함께 사용
- AWS SDK 또는 DescribeIamInstanceProfileAssociations CLI와 함께 사용
- AWS SDK 또는 DescribeIdFormat CLI와 함께 사용
- AWS SDK 또는 DescribeIdentityIdFormat CLI와 함께 사용
- AWS SDK 또는 DescribeImageAttribute CLI와 함께 사용
- AWS SDK 또는 DescribeImages CLI와 함께 사용
- AWS SDK 또는 DescribeImportImageTasks CLI와 함께 사용
- AWS SDK 또는 DescribeImportSnapshotTasks CLI와 함께 사용
- AWS SDK 또는 DescribeInstanceAttribute CLI와 함께 사용
- AWS SDK 또는 DescribeInstanceStatus CLI와 함께 사용
- AWS SDK 또는 DescribeInstanceTypes CLI와 함께 사용
- AWS SDK 또는 DescribeInstances CLI와 함께 사용
- AWS SDK 또는 DescribeInternetGateways CLI와 함께 사용
- AWS SDK 또는 DescribeKeyPairs CLI와 함께 사용
- AWS SDK 또는 DescribeNetworkAcls CLI와 함께 사용
- AWS SDK 또는 DescribeNetworkInterfaceAttribute CLI와 함께 사용
- AWS SDK 또는 DescribeNetworkInterfaces CLI와 함께 사용
- AWS SDK 또는 DescribePlacementGroups CLI와 함께 사용
- AWS SDK 또는 DescribePrefixLists CLI와 함께 사용
- AWS SDK 또는 DescribeRegions CLI와 함께 사용
- AWS SDK 또는 DescribeRouteTables CLI와 함께 사용
- AWS SDK 또는 DescribeScheduledInstanceAvailability CLI와 함께 사용

- AWS SDK 또는 DescribeScheduledInstances CLI와 함께 사용
- AWS SDK 또는 DescribeSecurityGroups CLI와 함께 사용
- AWS SDK 또는 DescribeSnapshotAttribute CLI와 함께 사용
- AWS SDK 또는 DescribeSnapshots CLI와 함께 사용
- AWS SDK 또는 DescribeSpotDatafeedSubscription CLI와 함께 사용
- AWS SDK 또는 DescribeSpotFleetInstances CLI와 함께 사용
- AWS SDK 또는 DescribeSpotFleetRequestHistory CLI와 함께 사용
- AWS SDK 또는 DescribeSpotFleetRequests CLI와 함께 사용
- AWS SDK 또는 DescribeSpotInstanceRequests CLI와 함께 사용
- AWS SDK 또는 DescribeSpotPriceHistory CLI와 함께 사용
- AWS SDK 또는 DescribeSubnets CLI와 함께 사용
- AWS SDK 또는 DescribeTags CLI와 함께 사용
- AWS SDK 또는 DescribeVolumeAttribute CLI와 함께 사용
- AWS SDK 또는 DescribeVolumeStatus CLI와 함께 사용
- AWS SDK 또는 DescribeVolumes CLI와 함께 사용
- AWS SDK 또는 DescribeVpcAttribute CLI와 함께 사용
- AWS SDK 또는 DescribeVpcClassicLink CLI와 함께 사용
- AWS SDK 또는 DescribeVpcClassicLinkDnsSupport CLI와 함께 사용
- AWS SDK 또는 DescribeVpcEndpointServices CLI와 함께 사용
- AWS SDK 또는 DescribeVpcEndpoints CLI와 함께 사용
- AWS SDK 또는 DescribeVpcs CLI와 함께 사용
- AWS SDK 또는 DescribeVpnConnections CLI와 함께 사용
- AWS SDK 또는 DescribeVpnGateways CLI와 함께 사용
- AWS SDK 또는 DetachInternetGateway CLI와 함께 사용
- AWS SDK 또는 DetachNetworkInterface CLI와 함께 사용
- AWS SDK 또는 DetachVolume CLI와 함께 사용
- AWS SDK 또는 DetachVpnGateway CLI와 함께 사용
- AWS SDK 또는 DisableVgwRoutePropagation CLI와 함께 사용
- AWS SDK 또는 DisableVpcClassicLink CLI와 함께 사용
- AWS SDK 또는 DisableVpcClassicLinkDnsSupport CLI와 함께 사용

- AWS SDK 또는 DisassociateAddress CLI와 함께 사용
- AWS SDK 또는 DisassociateRouteTable CLI와 함께 사용
- AWS SDK 또는 EnableVgwRoutePropagation CLI와 함께 사용
- AWS SDK 또는 EnableVolumeIo CLI와 함께 사용
- AWS SDK 또는 EnableVpcClassicLink CLI와 함께 사용
- AWS SDK 또는 EnableVpcClassicLinkDnsSupport CLI와 함께 사용
- AWS SDK 또는 GetConsoleOutput CLI와 함께 사용
- AWS SDK 또는 GetHostReservationPurchasePreview CLI와 함께 사용
- AWS SDK 또는 GetPasswordData CLI와 함께 사용
- AWS SDK 또는 ImportImage CLI와 함께 사용
- AWS SDK 또는 ImportKeyPair CLI와 함께 사용
- AWS SDK 또는 ImportSnapshot CLI와 함께 사용
- AWS SDK 또는 ModifyCapacityReservation CLI와 함께 사용
- AWS SDK 또는 ModifyHosts CLI와 함께 사용
- AWS SDK 또는 ModifyIdFormat CLI와 함께 사용
- AWS SDK 또는 ModifyImageAttribute CLI와 함께 사용
- AWS SDK 또는 ModifyInstanceAttribute CLI와 함께 사용
- AWS SDK 또는 ModifyInstanceCreditSpecification CLI와 함께 사용
- AWS SDK 또는 ModifyNetworkInterfaceAttribute CLI와 함께 사용
- AWS SDK 또는 ModifyReservedInstances CLI와 함께 사용
- AWS SDK 또는 ModifySnapshotAttribute CLI와 함께 사용
- AWS SDK 또는 ModifySpotFleetRequest CLI와 함께 사용
- AWS SDK 또는 ModifySubnetAttribute CLI와 함께 사용
- AWS SDK 또는 ModifyVolumeAttribute CLI와 함께 사용
- AWS SDK 또는 ModifyVpcAttribute CLI와 함께 사용
- AWS SDK 또는 MonitorInstances CLI와 함께 사용
- AWS SDK 또는 MoveAddressToVpc CLI와 함께 사용
- AWS SDK 또는 PurchaseHostReservation CLI와 함께 사용
- AWS SDK 또는 PurchaseScheduledInstances CLI와 함께 사용
- AWS SDK 또는 RebootInstances CLI와 함께 사용

- AWS SDK 또는 RegisterImage CLI와 함께 사용
- AWS SDK 또는 RejectVpcPeeringConnection CLI와 함께 사용
- AWS SDK 또는 ReleaseAddress CLI와 함께 사용
- AWS SDK 또는 ReleaseHosts CLI와 함께 사용
- AWS SDK 또는 ReplaceIamInstanceProfileAssociation CLI와 함께 사용
- AWS SDK 또는 ReplaceNetworkAclAssociation CLI와 함께 사용
- AWS SDK 또는 ReplaceNetworkAclEntry CLI와 함께 사용
- AWS SDK 또는 ReplaceRoute CLI와 함께 사용
- AWS SDK 또는 ReplaceRouteTableAssociation CLI와 함께 사용
- AWS SDK 또는 ReportInstanceStatus CLI와 함께 사용
- AWS SDK 또는 RequestSpotFleet CLI와 함께 사용
- AWS SDK 또는 RequestSpotInstances CLI와 함께 사용
- AWS SDK 또는 ResetImageAttribute CLI와 함께 사용
- AWS SDK 또는 ResetInstanceAttribute CLI와 함께 사용
- AWS SDK 또는 ResetNetworkInterfaceAttribute CLI와 함께 사용
- AWS SDK 또는 ResetSnapshotAttribute CLI와 함께 사용
- AWS SDK 또는 RevokeSecurityGroupEgress CLI와 함께 사용
- AWS SDK 또는 RevokeSecurityGroupIngress CLI와 함께 사용
- AWS SDK 또는 RunInstances CLI와 함께 사용
- AWS SDK 또는 RunScheduledInstances CLI와 함께 사용
- AWS SDK 또는 StartInstances CLI와 함께 사용
- AWS SDK 또는 StopInstances CLI와 함께 사용
- AWS SDK 또는 TerminateInstances CLI와 함께 사용
- AWS SDK 또는 UnassignPrivateIpAddresses CLI와 함께 사용
- AWS SDK 또는 UnmonitorInstances CLI와 함께 사용

- SDK를 사용하는 Amazon EC2의 시나리오 AWS
  - SDK를 사용하여 복원력이 뛰어난 서비스를 구축하고 관리합니다. AWS
  - SDK를 사용하여 Amazon EC2 인스턴스로 시작하기 AWS

SDK를 사용하는 Amazon EC2용 작업 AWS

다음 코드 예제는 SDK를 사용하여 개별 Amazon EC2 작업을 수행하는 방법을 보여줍니다. AWS 이들 발췌문은 Amazon EC2 API를 직접적으로 호출하며, 컨텍스트에서 실행되어야 하는 더 큰 프로그램에서 발췌한 코드입니다. 각 예제에는 코드 GitHub 설정 및 실행 지침을 찾을 수 있는 링크가 포함되어 있습니다.

다음 예제에는 가장 일반적으로 사용되는 작업만 포함되어 있습니다. 전체 목록은 Amazon Elastic Compute Cloud (Amazon EC2) API 참조에서 확인하세요.

예

- AWS SDK 또는 AcceptVpcPeeringConnection CLI와 함께 사용
- AWS SDK 또는 AllocateAddress CLI와 함께 사용
- AWS SDK 또는 AllocateHosts CLI와 함께 사용
- AWS SDK 또는 AssignPrivateIpAddresses CLI와 함께 사용
- AWS SDK 또는 AssociateAddress CLI와 함께 사용
- AWS SDK 또는 AssociateDhcpOptions CLI와 함께 사용
- AWS SDK 또는 AssociateRouteTable CLI와 함께 사용
- AWS SDK 또는 AttachInternetGateway CLI와 함께 사용
- AWS SDK 또는 AttachNetworkInterface CLI와 함께 사용
- AWS SDK 또는 AttachVolume CLI와 함께 사용
- AWS SDK 또는 AttachVpnGateway CLI와 함께 사용
- AWS SDK 또는 AuthorizeSecurityGroupEgress CLI와 함께 사용
- AWS SDK 또는 AuthorizeSecurityGroupIngress CLI와 함께 사용
- AWS SDK 또는 CancelCapacityReservation CLI와 함께 사용
- AWS SDK 또는 CancelImportTask CLI와 함께 사용
- AWS SDK 또는 CancelSpotFleetRequests CLI와 함께 사용
- AWS SDK 또는 CancelSpotInstanceRequests CLI와 함께 사용
- AWS SDK 또는 ConfirmProductInstance CLI와 함께 사용
- AWS SDK 또는 CopyImage CLI와 함께 사용
- AWS SDK 또는 CopySnapshot CLI와 함께 사용
- AWS SDK 또는 CreateCapacityReservation CLI와 함께 사용

- AWS SDK 또는 CreateCustomerGateway CLI와 함께 사용
- AWS SDK 또는 CreateDhcpOptions CLI와 함께 사용
- AWS SDK 또는 CreateFlowLogs CLI와 함께 사용
- AWS SDK 또는 CreateImage CLI와 함께 사용
- AWS SDK 또는 CreateInstanceExportTask CLI와 함께 사용
- AWS SDK 또는 CreateInternetGateway CLI와 함께 사용
- AWS SDK 또는 CreateKeyPair CLI와 함께 사용
- AWS SDK 또는 CreateLaunchTemplate CLI와 함께 사용
- AWS SDK 또는 CreateNetworkAcl CLI와 함께 사용
- AWS SDK 또는 CreateNetworkAclEntry CLI와 함께 사용
- AWS SDK 또는 CreateNetworkInterface CLI와 함께 사용
- AWS SDK 또는 CreatePlacementGroup CLI와 함께 사용
- AWS SDK 또는 CreateRoute CLI와 함께 사용
- AWS SDK 또는 CreateRouteTable CLI와 함께 사용
- AWS SDK 또는 CreateSecurityGroup CLI와 함께 사용
- AWS SDK 또는 CreateSnapshot CLI와 함께 사용
- AWS SDK 또는 CreateSpotDatafeedSubscription CLI와 함께 사용
- AWS SDK 또는 CreateSubnet CLI와 함께 사용
- AWS SDK 또는 CreateTags CLI와 함께 사용
- AWS SDK 또는 CreateVolume CLI와 함께 사용
- AWS SDK 또는 CreateVpc CLI와 함께 사용
- AWS SDK 또는 CreateVpcEndpoint CLI와 함께 사용
- AWS SDK 또는 CreateVpnConnection CLI와 함께 사용
- AWS SDK 또는 CreateVpnConnectionRoute CLI와 함께 사용
- AWS SDK 또는 CreateVpnGateway CLI와 함께 사용
- AWS SDK 또는 DeleteCustomerGateway CLI와 함께 사용
- AWS SDK 또는 DeleteDhcpOptions CLI와 함께 사용
- AWS SDK 또는 DeleteFlowLogs CLI와 함께 사용
- AWS SDK 또는 DeleteInternetGateway CLI와 함께 사용

- AWS SDK 또는 DeleteKeyPair CLI와 함께 사용
- AWS SDK 또는 DeleteLaunchTemplate CLI와 함께 사용
- AWS SDK 또는 DeleteNetworkAcl CLI와 함께 사용
- AWS SDK 또는 DeleteNetworkAclEntry CLI와 함께 사용
- AWS SDK 또는 DeleteNetworkInterface CLI와 함께 사용
- AWS SDK 또는 DeletePlacementGroup CLI와 함께 사용
- AWS SDK 또는 DeleteRoute CLI와 함께 사용
- AWS SDK 또는 DeleteRouteTable CLI와 함께 사용
- AWS SDK 또는 DeleteSecurityGroup CLI와 함께 사용
- AWS SDK 또는 DeleteSnapshot CLI와 함께 사용
- AWS SDK 또는 DeleteSpotDatafeedSubscription CLI와 함께 사용
- AWS SDK 또는 DeleteSubnet CLI와 함께 사용
- AWS SDK 또는 DeleteTags CLI와 함께 사용
- AWS SDK 또는 DeleteVolume CLI와 함께 사용
- AWS SDK 또는 DeleteVpc CLI와 함께 사용
- AWS SDK 또는 DeleteVpnConnection CLI와 함께 사용
- AWS SDK 또는 DeleteVpnConnectionRoute CLI와 함께 사용
- AWS SDK 또는 DeleteVpnGateway CLI와 함께 사용
- AWS SDK 또는 DeregisterImage CLI와 함께 사용
- AWS SDK 또는 DescribeAccountAttributes CLI와 함께 사용
- AWS SDK 또는 DescribeAddresses CLI와 함께 사용
- AWS SDK 또는 DescribeAvailabilityZones CLI와 함께 사용
- AWS SDK 또는 DescribeBundleTasks CLI와 함께 사용
- AWS SDK 또는 DescribeCapacityReservations CLI와 함께 사용
- AWS SDK 또는 DescribeCustomerGateways CLI와 함께 사용
- AWS SDK 또는 DescribeDhcpOptions CLI와 함께 사용
- AWS SDK 또는 DescribeFlowLogs CLI와 함께 사용
- AWS SDK 또는 DescribeHostReservationOfferings CLI와 함께 사용
- AWS SDK 또는 DescribeHosts CLI와 함께 사용

- AWS SDK 또는 DescribeIamInstanceProfileAssociations CLI와 함께 사용
- AWS SDK 또는 DescribeIdFormat CLI와 함께 사용
- AWS SDK 또는 DescribeIdentityIdFormat CLI와 함께 사용
- AWS SDK 또는 DescribeImageAttribute CLI와 함께 사용
- AWS SDK 또는 DescribeImages CLI와 함께 사용
- AWS SDK 또는 DescribeImportImageTasks CLI와 함께 사용
- AWS SDK 또는 DescribeImportSnapshotTasks CLI와 함께 사용
- AWS SDK 또는 DescribeInstanceAttribute CLI와 함께 사용
- AWS SDK 또는 DescribeInstanceStatus CLI와 함께 사용
- AWS SDK 또는 DescribeInstanceTypes CLI와 함께 사용
- AWS SDK 또는 DescribeInstances CLI와 함께 사용
- AWS SDK 또는 DescribeInternetGateways CLI와 함께 사용
- AWS SDK 또는 DescribeKeyPairs CLI와 함께 사용
- AWS SDK 또는 DescribeNetworkAcls CLI와 함께 사용
- AWS SDK 또는 DescribeNetworkInterfaceAttribute CLI와 함께 사용
- AWS SDK 또는 DescribeNetworkInterfaces CLI와 함께 사용
- AWS SDK 또는 DescribePlacementGroups CLI와 함께 사용
- AWS SDK 또는 DescribePrefixLists CLI와 함께 사용
- AWS SDK 또는 DescribeRegions CLI와 함께 사용
- AWS SDK 또는 DescribeRouteTables CLI와 함께 사용
- AWS SDK 또는 DescribeScheduledInstanceAvailability CLI와 함께 사용
- AWS SDK 또는 DescribeScheduledInstances CLI와 함께 사용
- AWS SDK 또는 DescribeSecurityGroups CLI와 함께 사용
- AWS SDK 또는 DescribeSnapshotAttribute CLI와 함께 사용
- AWS SDK 또는 DescribeSnapshots CLI와 함께 사용
- AWS SDK 또는 DescribeSpotDatafeedSubscription CLI와 함께 사용
- AWS SDK 또는 DescribeSpotFleetInstances CLI와 함께 사용
- AWS SDK 또는 DescribeSpotFleetRequestHistory CLI와 함께 사용
- AWS SDK 또는 DescribeSpotFleetRequests CLI와 함께 사용

- AWS SDK 또는 DescribeSpotInstanceRequests CLI와 함께 사용
- AWS SDK 또는 DescribeSpotPriceHistory CLI와 함께 사용
- AWS SDK 또는 DescribeSubnets CLI와 함께 사용
- AWS SDK 또는 DescribeTags CLI와 함께 사용
- AWS SDK 또는 DescribeVolumeAttribute CLI와 함께 사용
- AWS SDK 또는 DescribeVolumeStatus CLI와 함께 사용
- AWS SDK 또는 DescribeVolumes CLI와 함께 사용
- AWS SDK 또는 DescribeVpcAttribute CLI와 함께 사용
- AWS SDK 또는 DescribeVpcClassicLink CLI와 함께 사용
- AWS SDK 또는 DescribeVpcClassicLinkDnsSupport CLI와 함께 사용
- AWS SDK 또는 DescribeVpcEndpointServices CLI와 함께 사용
- AWS SDK 또는 DescribeVpcEndpoints CLI와 함께 사용
- AWS SDK 또는 DescribeVpcs CLI와 함께 사용
- AWS SDK 또는 DescribeVpnConnections CLI와 함께 사용
- AWS SDK 또는 DescribeVpnGateways CLI와 함께 사용
- AWS SDK 또는 DetachInternetGateway CLI와 함께 사용
- AWS SDK 또는 DetachNetworkInterface CLI와 함께 사용
- AWS SDK 또는 DetachVolume CLI와 함께 사용
- AWS SDK 또는 DetachVpnGateway CLI와 함께 사용
- AWS SDK 또는 DisableVgwRoutePropagation CLI와 함께 사용
- AWS SDK 또는 DisableVpcClassicLink CLI와 함께 사용
- AWS SDK 또는 DisableVpcClassicLinkDnsSupport CLI와 함께 사용
- AWS SDK 또는 DisassociateAddress CLI와 함께 사용
- AWS SDK 또는 DisassociateRouteTable CLI와 함께 사용
- AWS SDK 또는 EnableVgwRoutePropagation CLI와 함께 사용
- AWS SDK 또는 EnableVolumeIo CLI와 함께 사용
- AWS SDK 또는 EnableVpcClassicLink CLI와 함께 사용
- AWS SDK 또는 EnableVpcClassicLinkDnsSupport CLI와 함께 사용
- AWS SDK 또는 GetConsoleOutput CLI와 함께 사용

- AWS SDK 또는 GetHostReservationPurchasePreview CLI와 함께 사용
- AWS SDK 또는 GetPasswordData CLI와 함께 사용
- AWS SDK 또는 ImportImage CLI와 함께 사용
- AWS SDK 또는 ImportKeyPair CLI와 함께 사용
- AWS SDK 또는 ImportSnapshot CLI와 함께 사용
- AWS SDK 또는 ModifyCapacityReservation CLI와 함께 사용
- AWS SDK 또는 ModifyHosts CLI와 함께 사용
- AWS SDK 또는 ModifyIdFormat CLI와 함께 사용
- AWS SDK 또는 ModifyImageAttribute CLI와 함께 사용
- AWS SDK 또는 ModifyInstanceAttribute CLI와 함께 사용
- AWS SDK 또는 ModifyInstanceCreditSpecification CLI와 함께 사용
- AWS SDK 또는 ModifyNetworkInterfaceAttribute CLI와 함께 사용
- AWS SDK 또는 ModifyReservedInstances CLI와 함께 사용
- AWS SDK 또는 ModifySnapshotAttribute CLI와 함께 사용
- AWS SDK 또는 ModifySpotFleetRequest CLI와 함께 사용
- AWS SDK 또는 ModifySubnetAttribute CLI와 함께 사용
- AWS SDK 또는 ModifyVolumeAttribute CLI와 함께 사용
- AWS SDK 또는 ModifyVpcAttribute CLI와 함께 사용
- AWS SDK 또는 MonitorInstances CLI와 함께 사용
- AWS SDK 또는 MoveAddressToVpc CLI와 함께 사용
- AWS SDK 또는 PurchaseHostReservation CLI와 함께 사용
- AWS SDK 또는 PurchaseScheduledInstances CLI와 함께 사용
- AWS SDK 또는 RebootInstances CLI와 함께 사용
- AWS SDK 또는 RegisterImage CLI와 함께 사용
- AWS SDK 또는 RejectVpcPeeringConnection CLI와 함께 사용
- AWS SDK 또는 ReleaseAddress CLI와 함께 사용
- AWS SDK 또는 ReleaseHosts CLI와 함께 사용
- AWS SDK 또는 ReplaceIamInstanceProfileAssociation CLI와 함께 사용
- AWS SDK 또는 ReplaceNetworkAclAssociation CLI와 함께 사용
- AWS SDK 또는 ReplaceNetworkAclEntry CLI와 함께 사용

- AWS SDK 또는 ReplaceRoute CLI와 함께 사용
- AWS SDK 또는 ReplaceRouteTableAssociation CLI와 함께 사용
- AWS SDK 또는 ReportInstanceStatus CLI와 함께 사용
- AWS SDK 또는 RequestSpotFleet CLI와 함께 사용
- AWS SDK 또는 RequestSpotInstances CLI와 함께 사용
- AWS SDK 또는 ResetImageAttribute CLI와 함께 사용
- AWS SDK 또는 ResetInstanceAttribute CLI와 함께 사용
- AWS SDK 또는 ResetNetworkInterfaceAttribute CLI와 함께 사용
- AWS SDK 또는 ResetSnapshotAttribute CLI와 함께 사용
- AWS SDK 또는 RevokeSecurityGroupEgress CLI와 함께 사용
- AWS SDK 또는 RevokeSecurityGroupIngress CLI와 함께 사용
- AWS SDK 또는 RunInstances CLI와 함께 사용
- AWS SDK 또는 RunScheduledInstances CLI와 함께 사용
- AWS SDK 또는 StartInstances CLI와 함께 사용
- AWS SDK 또는 StopInstances CLI와 함께 사용
- AWS SDK 또는 TerminateInstances CLI와 함께 사용
- AWS SDK 또는 UnassignPrivateIpAddresses CLI와 함께 사용
- AWS SDK 또는 UnmonitorInstances CLI와 함께 사용

AWS SDK 또는 **AcceptVpcPeeringConnection** CLI와 함께 사용

다음 코드 예제는 AcceptVpcPeeringConnection의 사용 방법을 보여줍니다.

CLI

AWS CLI

VPC 피어링 연결을 수락하려면

이 예제는 지정된 VPC 피어링 연결 요청을 수락합니다.

명령:

```
aws ec2 accept-vpc-peering-connection --vpc-peering-connection-id pcx-1a2b3c4d
```

출력:

```
{
  "VpcPeeringConnection": {
    "Status": {
      "Message": "Provisioning",
      "Code": "provisioning"
    },
    "Tags": [],
    "AccepterVpcInfo": {
      "OwnerId": "444455556666",
      "VpcId": "vpc-44455566",
      "CidrBlock": "10.0.1.0/28"
    },
    "VpcPeeringConnectionId": "pcx-1a2b3c4d",
    "RequesterVpcInfo": {
      "OwnerId": "444455556666",
      "VpcId": "vpc-111abc45",
      "CidrBlock": "10.0.0.0/28"
    }
  }
}
```

- API 세부 정보는 AWS CLI 명령 AcceptVpcPeeringConnection참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 요청된 VpcPeeringConnectionId pcx-1dfad234b56ff78be를 승인합니다.

```
Approve-EC2VpcPeeringConnection -VpcPeeringConnectionId pcx-1dfad234b56ff78be
```

출력:

```
AccepterVpcInfo        : Amazon.EC2.Model.VpcPeeringConnectionVpcInfo
ExpirationTime         : 1/1/0001 12:00:00 AM
RequesterVpcInfo       : Amazon.EC2.Model.VpcPeeringConnectionVpcInfo
Status                 : Amazon.EC2.Model.VpcPeeringConnectionStateReason
Tags                   : {}
VpcPeeringConnectionId : pcx-1dfad234b56ff78be
```

- AWS Tools for PowerShell API에 대한 AcceptVpcPeeringConnection자세한 내용은 Cmdlet 참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **AllocateAddress** CLI와 함께 사용

다음 코드 예제는 AllocateAddress의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 인스턴스 시작하기

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
/// <summary>
/// Allocate an Elastic IP address.
/// </summary>
/// <returns>The allocation Id of the allocated address.</returns>
public async Task<string> AllocateAddress()
{
    var request = new AllocateAddressRequest();

    var response = await _amazonEC2.AllocateAddressAsync(request);
    return response.AllocationId;
}
```

- API 세부 정보는 AWS SDK for .NET API AllocateAddress참조를 참조하십시오.

Bash

AWS CLI Bash 스크립트 사용

> **Note**
>
> 더 많은 정보가 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
###############################################################################
# function ec2_allocate_address
#
# This function allocates an Elastic IP address for use with Amazon Elastic
 Compute Cloud (Amazon EC2) instances in a specific AWS Region.
#
# Parameters:
#       -d domain - The domain for the Elastic IP address (either 'vpc' or
 'standard').
#
# Returns:
#       The allocated Elastic IP address, or an error message if the operation
 fails.
# And:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
function ec2_allocate_address() {
  local domain response

  # Function to display usage information
  function usage() {
    echo "function ec2_allocate_address"
    echo "Allocates an Elastic IP address for use with Amazon Elastic Compute
 Cloud (Amazon EC2) instances in a specific AWS Region."
    echo "  -d domain - The domain for the Elastic IP address (either 'vpc' or
 'standard')."
    echo ""
  }

  # Parse the command-line arguments
```

```
  while getopts "d:h" option; do
    case "${option}" in
      d) domain="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Validate the input parameters
  if [[ -z "$domain" ]]; then
    errecho "ERROR: You must provide a domain with the -d parameter (either 'vpc'
 or 'standard')."
    return 1
  fi

  if [[ "$domain" != "vpc" && "$domain" != "standard" ]]; then
    errecho "ERROR: Invalid domain value. Must be either 'vpc' or 'standard'."
    return 1
  fi

  # Allocate the Elastic IP address
  response=$(aws ec2 allocate-address \
    --domain "$domain" \
    --query "[PublicIp,AllocationId]" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports allocate-address operation failed."
    errecho "$response"
    return 1
  }

  echo "$response"
  return 0
}
```

이 예제에 사용된 유틸리티 함수

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
##############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- API 세부 정보는 AWS CLI 명령 AllocateAddress참조를 참조하십시오.

C++

SDK for C++

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
Aws::EC2::EC2Client ec2Client(clientConfiguration);

Aws::EC2::Model::AllocateAddressRequest request;
request.SetDomain(Aws::EC2::Model::DomainType::vpc);

const Aws::EC2::Model::AllocateAddressOutcome outcome =
        ec2Client.AllocateAddress(request);
if (!outcome.IsSuccess()) {
    std::cerr << "Failed to allocate Elastic IP address:" <<
              outcome.GetError().GetMessage() << std::endl;
    return false;
}

allocationId = outcome.GetResult().GetAllocationId();
```

- API 세부 정보는 AWS SDK for C++ API AllocateAddress참조를 참조하십시오.

CLI

AWS CLI

예제 1: Amazon 주소 풀에서 탄력적 IP 주소를 할당하는 방법

다음 `allocate-address` 예제는 탄력적 IP 주소를 할당합니다. Amazon EC2는 Amazon 주소 풀에서 주소를 선택합니다.

```
aws ec2 allocate-address
```

출력:

```
{
    "PublicIp": "70.224.234.241",
    "AllocationId": "eipalloc-01435ba59eEXAMPLE",
    "PublicIpv4Pool": "amazon",
    "NetworkBorderGroup": "us-west-2",
    "Domain": "vpc"
}
```

자세한 내용은 Amazon EC2 사용 설명서에서 탄력적 IP 주소를 참조하세요.

예제 2: 탄력적 IP 주소를 할당하고 네트워크 경계 그룹에 연결하는 방법

다음 `allocate-address` 예제에서는 탄력적 IP 주소를 할당하고 해당 주소를 지정된 네트워크 경계 그룹에 연결합니다.

```
aws ec2 allocate-address \
    --network-border-group us-west-2-lax-1
```

출력:

```
{
    "PublicIp": "70.224.234.241",
    "AllocationId": "eipalloc-e03dd489ceEXAMPLE",
    "PublicIpv4Pool": "amazon",
    "NetworkBorderGroup": "us-west-2-lax-1",
    "Domain": "vpc"
}
```

자세한 내용은 Amazon EC2 사용 설명서에서 탄력적 IP 주소를 참조하세요.

예 3: 소유한 주소 풀에서 탄력적 IP 주소를 할당하는 방법

다음 `allocate-address` 예제에서는 Amazon Web Services 계정으로 가져온 주소 풀에서 탄력적 IP 주소를 할당합니다. Amazon EC2는 주소 풀에서 주소를 선택합니다.

```
aws ec2 allocate-address \
```

```
    --public-ipv4-pool ipv4pool-ec2-1234567890abcdef0
```

출력:

```
{
    "AllocationId": "eipalloc-02463d08ceEXAMPLE",
    "NetworkBorderGroup": "us-west-2",
    "CustomerOwnedIp": "18.218.95.81",
    "CustomerOwnedIpv4Pool": "ipv4pool-ec2-1234567890abcdef0",
    "Domain": "vpc"
    "NetworkBorderGroup": "us-west-2",
}
```

자세한 내용은 Amazon EC2 사용 설명서에서 탄력적 IP 주소를 참조하세요.

- API 세부 정보는 AWS CLI 명령 AllocateAddress참조를 참조하십시오.

Java

SDK for Java 2.x

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    public static String allocateAddress(Ec2Client ec2) {
        try {
            AllocateAddressRequest allocateRequest =
AllocateAddressRequest.builder()
                    .domain(DomainType.VPC)
                    .build();

            AllocateAddressResponse allocateResponse =
ec2.allocateAddress(allocateRequest);
            return allocateResponse.allocationId();

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
```

```
        }
        return "";
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API AllocateAddress참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { AllocateAddressCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new AllocateAddressCommand({});

  try {
    const { AllocationId, PublicIp } = await client.send(command);
    console.log("A new IP address has been allocated to your account:");
    console.log(`ID: ${AllocationId} Public IP: ${PublicIp}`);
    console.log(
      "You can view your IP addresses in the AWS Management Console for Amazon
 EC2. Look under Network & Security > Elastic IPs",
    );
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API AllocateAddress참조를 참조하십시오.

Kotlin

SDK for Kotlin

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
suspend fun getAllocateAddress(instanceIdVal: String?): String? {
    val allocateRequest =
        AllocateAddressRequest {
            domain = DomainType.Vpc
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val allocateResponse = ec2.allocateAddress(allocateRequest)
        val allocationIdVal = allocateResponse.allocationId

        val request =
            AssociateAddressRequest {
                instanceId = instanceIdVal
                allocationId = allocationIdVal
            }

        val associateResponse = ec2.associateAddress(request)
        return associateResponse.associationId
    }
}
```

- API 세부 정보는 Kotlin API용AWS SDK 레퍼런스를 참조하세요 AllocateAddress.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예제는 VPC의 인스턴스에 사용할 엘라스틱 IP 주소를 할당합니다.

```
New-EC2Address -Domain Vpc
```

출력:

```
AllocationId         Domain      PublicIp
------------         ------      --------
eipalloc-12345678    vpc         198.51.100.2
```

예 2: 이 예시에서는 EC2-Classic의 인스턴스에 사용할 엘라스틱 IP 주소를 할당합니다.

```
New-EC2Address
```

출력:

```
AllocationId         Domain      PublicIp
------------         ------      --------
                     standard    203.0.113.17
```

- API 세부 정보는 Cmdlet 참조를 참조하십시오 AllocateAddress.AWS Tools for PowerShell

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class ElasticIpWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) Elastic IP address
 actions."""

    def __init__(self, ec2_resource, elastic_ip=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param elastic_ip: A Boto3 VpcAddress object. This is a high-level object
 that
```

```
                            wraps Elastic IP actions.
        """
        self.ec2_resource = ec2_resource
        self.elastic_ip = elastic_ip

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def allocate(self):
        """
        Allocates an Elastic IP address that can be associated with an Amazon EC2
        instance. By using an Elastic IP address, you can keep the public IP
address
        constant even when you restart the associated instance.

        :return: The newly created Elastic IP object. By default, the address is
not
                 associated with any instance.
        """
        try:
            response =
self.ec2_resource.meta.client.allocate_address(Domain="vpc")
            self.elastic_ip =
self.ec2_resource.VpcAddress(response["AllocationId"])
        except ClientError as err:
            logger.error(
                "Couldn't allocate Elastic IP. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return self.elastic_ip
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 AllocateAddress.

Ruby

SDK for Ruby

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
# Creates an Elastic IP address in Amazon Virtual Private Cloud (Amazon VPC).
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @return [String] The allocation ID corresponding to the Elastic IP address.
# @example
#   puts allocate_elastic_ip_address(Aws::EC2::Client.new(region: 'us-west-2'))
def allocate_elastic_ip_address(ec2_client)
  response = ec2_client.allocate_address(domain: "vpc")
  return response.allocation_id
rescue StandardError => e
  puts "Error allocating Elastic IP address: #{e.message}"
  return "Error"
end
```

- API 세부 정보는 AWS SDK for Ruby API AllocateAddress참조를 참조하십시오.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    TRY.
        oo_result = lo_ec2->allocateaddress( iv_domain = 'vpc' ).   " oo_result
 is returned for testing purposes. "
```

```
        MESSAGE 'Allocated an Elastic IP address.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- API 세부 정보는 SAP용AWS SDK ABAP API 참조를 참조하십시오 AllocateAddress.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **AllocateHosts** CLI와 함께 사용

다음 코드 예제는 AllocateHosts의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 전용 호스트를 할당하려면

다음 allocate-hosts 예시에서는 가용 eu-west-1a 영역에 단일 전용 호스트를 할당하여 인스턴스를 시작할 m5.large 수 있습니다. 기본적으로 전용 호스트는 대상 인스턴스 시작만 허용하고 호스트 복구는 지원하지 않습니다.

```
aws ec2 allocate-hosts \
    --instance-type m5.large \
    --availability-zone eu-west-1a \
    --quantity 1
```

출력:

```
{
    "HostIds": [
        "h-07879acf49EXAMPLE"
    ]
}
```

예 2: 자동 배치 및 호스트 복구가 활성화된 전용 호스트를 할당하려면

다음 allocate-hosts 예에서는 자동 배치 및 호스트 복구가 활성화된 가용 eu-west-1a 영역에 단일 전용 호스트를 할당합니다.

```
aws ec2 allocate-hosts \
    --instance-type m5.large \
    --availability-zone eu-west-1a \
    --auto-placement on \
    --host-recovery on \
    --quantity 1
```

출력:

```
{
     "HostIds": [
         "h-07879acf49EXAMPLE"
     ]
}
```

예 3: 태그를 사용하여 전용 호스트를 할당하려면

다음 allocate-hosts 예시에서는 단일 전용 호스트를 할당하고 키 이름이 purpose 지정되고 값이 인 태그를 적용합니다. production

```
aws ec2 allocate-hosts \
    --instance-type m5.large \
    --availability-zone eu-west-1a \
    --quantity 1 \
    --tag-specifications 'ResourceType=dedicated-
host,Tags={Key=purpose,Value=production}'
```

출력:

```
{
    "HostIds": [
        "h-07879acf49EXAMPLE"
    ]
}
```

자세한 내용은 Linux 인스턴스용 Amazon Elastic Compute 클라우드 사용 설명서의 전용 호스트 할당을 참조하십시오.

- API 세부 정보는 AWS CLI 명령 AllocateHosts참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 인스턴스 유형 및 가용 영역에 대해 계정에 전용 호스트를 할당합니다.

```
New-EC2Host -AutoPlacement on -AvailabilityZone eu-west-1b -InstanceType
 m4.xlarge -Quantity 1
```

출력:

```
h-01e23f4cd567890f3
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 AllocateHosts.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **AssignPrivateIpAddresses** CLI와 함께 사용

다음 코드 예제는 AssignPrivateIpAddresses의 사용 방법을 보여줍니다.

CLI

AWS CLI

특정 보조 사설 IP 주소에 네트워크 인터페이스를 할당하려면

이 예제에서는 지정된 보조 사설 IP 주소를 지정된 네트워크 인터페이스에 할당합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 assign-private-ip-addresses --network-interface-id eni-e5aa89a3 --
private-ip-addresses 10.0.0.82
```

Amazon EC2가 선택한 보조 사설 IP 주소를 네트워크 인터페이스에 할당하려면

이 예제에서는 지정된 네트워크 인터페이스에 두 개의 보조 사설 IP 주소를 할당합니다. Amazon EC2는 네트워크 인터페이스가 연결된 서브넷의 CIDR 블록 범위 내에서 사용 가능한 IP 주소를 사용하여 이러한 IP 주소를 자동으로 할당합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 assign-private-ip-addresses --network-interface-id eni-e5aa89a3 --
secondary-private-ip-address-count 2
```

- API 세부 정보는 명령 참조를 참조하십시오 AssignPrivateIpAddresses.AWS CLI

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 보조 사설 IP 주소를 지정된 네트워크 인터페이스에 할당합니다.

```
Register-EC2PrivateIpAddress -NetworkInterfaceId eni-1a2b3c4d -PrivateIpAddress
 10.0.0.82
```

예 2: 이 예에서는 두 개의 보조 사설 IP 주소를 생성하여 지정된 네트워크 인터페이스에 할당합니다.

```
Register-EC2PrivateIpAddress -NetworkInterfaceId eni-1a2b3c4d -
SecondaryPrivateIpAddressCount 2
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 AssignPrivateIpAddresses.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **AssociateAddress** CLI와 함께 사용

다음 코드 예제는 AssociateAddress의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 인스턴스 시작하기

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Associate an Elastic IP address to an EC2 instance.
    /// </summary>
    /// <param name="allocationId">The allocation Id of an Elastic IP address.</
param>
    /// <param name="instanceId">The instance Id of the EC2 instance to
    /// associate the address with.</param>
    /// <returns>The association Id that represents
    /// the association of the Elastic IP address with an instance.</returns>
    public async Task<string> AssociateAddress(string allocationId, string
 instanceId)
    {
        var request = new AssociateAddressRequest
        {
            AllocationId = allocationId,
            InstanceId = instanceId
        };

        var response = await _amazonEC2.AssociateAddressAsync(request);
        return response.AssociationId;
    }
```

- API 세부 정보는 AWS SDK for .NET API AssociateAddress참조를 참조하십시오.

Bash

AWS CLI Bash 스크립트 사용

> **Note**
>
> 더 많은 정보가 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
###############################################################################
# function ec2_associate_address
#
# This function associates an Elastic IP address with an Amazon Elastic Compute
 Cloud (Amazon EC2) instance.
#
# Parameters:
#       -a allocation_id - The allocation ID of the Elastic IP address to
 associate.
#       -i instance_id - The ID of the EC2 instance to associate the Elastic IP
 address with.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
function ec2_associate_address() {
  local allocation_id instance_id response

  # Function to display usage information
  function usage() {
    echo "function ec2_associate_address"
    echo "Associates an Elastic IP address with an Amazon Elastic Compute Cloud
 (Amazon EC2) instance."
    echo "  -a allocation_id - The allocation ID of the Elastic IP address to
 associate."
    echo "  -i instance_id - The ID of the EC2 instance to associate the Elastic
 IP address with."
    echo ""
  }
```

```
  # Parse the command-line arguments
  while getopts "a:i:h" option; do
    case "${option}" in
      a) allocation_id="${OPTARG}" ;;
      i) instance_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Validate the input parameters
  if [[ -z "$allocation_id" ]]; then
    errecho "ERROR: You must provide an allocation ID with the -a parameter."
    return 1
  fi

  if [[ -z "$instance_id" ]]; then
    errecho "ERROR: You must provide an instance ID with the -i parameter."
    return 1
  fi

  # Associate the Elastic IP address
  response=$(aws ec2 associate-address \
    --allocation-id "$allocation_id" \
    --instance-id "$instance_id" \
    --query "AssociationId" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports associate-address operation failed."
    errecho "$response"
    return 1
  }

  echo "$response"
  return 0
}
```

이 예제에 사용된 유틸리티 함수

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
###############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi
```

```
  return 0
}
```

- API 세부 정보는 AWS CLI 명령 AssociateAddress참조를 참조하십시오.

C++

SDK for C++

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);

    Aws::EC2::Model::AssociateAddressRequest associate_request;
    associate_request.SetInstanceId(instanceId);
    associate_request.SetAllocationId(allocationId);

    const Aws::EC2::Model::AssociateAddressOutcome associate_outcome =
            ec2Client.AssociateAddress(associate_request);
    if (!associate_outcome.IsSuccess()) {
        std::cerr << "Failed to associate Elastic IP address " << allocationId
                  << " with instance " << instanceId << ":" <<
                  associate_outcome.GetError().GetMessage() << std::endl;
        return false;
    }

    std::cout << "Successfully associated Elastic IP address " << allocationId
              << " with instance " << instanceId << std::endl;
```

- API 세부 정보는 AWS SDK for C++ API AssociateAddress참조를 참조하십시오.

CLI

AWS CLI

EC2-Classic에서 탄력적 IP 주소를 연결하는 방법

이 예제에서는 탄력적 IP 주소를 EC2-Classic의 인스턴스에 연결합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 associate-address --instance-id i-07ffe74c7330ebf53 --public-ip
 198.51.100.0
```

EC2-VPC에서 탄력적 IP 주소를 연결하는 방법

이 예제에서는 VPC의 인스턴스에 탄력적 IP 주소를 연결합니다.

명령:

```
aws ec2 associate-address --instance-id i-0b263919b6498b123 --allocation-id
 eipalloc-64d5890a
```

출력:

```
{
    "AssociationId": "eipassoc-2bebb745"
}
```

이 예제에서는 네트워크 인터페이스에 탄력적 IP 주소를 연결합니다.

명령:

```
aws ec2 associate-address --allocation-id eipalloc-64d5890a --network-interface-
id eni-1a2b3c4d
```

이 예제에서는 네트워크 인터페이스와 연결된 프라이빗 IP 주소에 탄력적 IP를 연결합니다.

명령:

```
aws ec2 associate-address --allocation-id eipalloc-64d5890a --network-interface-
id eni-1a2b3c4d --private-ip-address 10.0.0.85
```

- API 세부 정보는 AWS CLI 명령 AssociateAddress참조를 참조하십시오.

Java

SDK for Java 2.x

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
   public static String associateAddress(Ec2Client ec2, String instanceId,
String allocationId) {
       try {
           AssociateAddressRequest associateRequest =
AssociateAddressRequest.builder()
                   .instanceId(instanceId)
                   .allocationId(allocationId)
                   .build();

           AssociateAddressResponse associateResponse =
ec2.associateAddress(associateRequest);
           return associateResponse.associationId();

       } catch (Ec2Exception e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
       return "";
   }
```

- API 세부 정보는 AWS SDK for Java 2.x API AssociateAddress참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { AssociateAddressCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  // You need to allocate an Elastic IP address before associating it with an
 instance.
  // You can do that with the AllocateAddressCommand.
  const allocationId = "ALLOCATION_ID";
  // You need to create an EC2 instance before an IP address can be associated
 with it.
  // You can do that with the RunInstancesCommand.
  const instanceId = "INSTANCE_ID";
  const command = new AssociateAddressCommand({
    AllocationId: allocationId,
    InstanceId: instanceId,
  });

  try {
    const { AssociationId } = await client.send(command);
    console.log(
      `Address with allocation ID ${allocationId} is now associated with instance
 ${instanceId}.`,
      `The association ID is ${AssociationId}.`,
    );
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API AssociateAddress참조를 참조하십시오.

Kotlin

SDK for Kotlin

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
suspend fun associateAddressSc(
    instanceIdVal: String?,
    allocationIdVal: String?,
): String? {
    val associateRequest =
        AssociateAddressRequest {
            instanceId = instanceIdVal
            allocationId = allocationIdVal
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val associateResponse = ec2.associateAddress(associateRequest)
        return associateResponse.associationId
    }
}
```

- API 세부 정보는 Kotlin API용AWS SDK 레퍼런스를 참조하세요 AssociateAddress.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예제는 지정된 엘라스틱 IP 주소를 VPC의 지정된 인스턴스와 연결합니다.

```
C:\> Register-EC2Address -InstanceId i-12345678 -AllocationId eipalloc-12345678
```

출력:

```
eipassoc-12345678
```

예 2: 이 예제는 지정된 엘라스틱 IP 주소를 EC2-Classic의 지정된 인스턴스와 연결합니다.

```
C:\> Register-EC2Address -InstanceId i-12345678 -PublicIp 203.0.113.17
```

- API 세부 정보는 Cmdlet 참조를 참조하십시오 AssociateAddress.AWS Tools for PowerShell

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class ElasticIpWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) Elastic IP address
 actions."""

    def __init__(self, ec2_resource, elastic_ip=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param elastic_ip: A Boto3 VpcAddress object. This is a high-level object
 that
                           wraps Elastic IP actions.
        """
        self.ec2_resource = ec2_resource
        self.elastic_ip = elastic_ip

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def associate(self, instance):
        """
```

```
        Associates an Elastic IP address with an instance. When this association
is
        created, the Elastic IP's public IP address is immediately used as the
public
        IP address of the associated instance.

        :param instance: A Boto3 Instance object. This is a high-level object
that wraps
                         Amazon EC2 instance actions.
        :return: A response that contains the ID of the association.
        """
        if self.elastic_ip is None:
            logger.info("No Elastic IP to associate.")
            return

        try:
            response = self.elastic_ip.associate(InstanceId=instance.id)
        except ClientError as err:
            logger.error(
                "Couldn't associate Elastic IP %s with instance %s. Here's why:
%s: %s",
                self.elastic_ip.allocation_id,
                instance.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        return response
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 AssociateAddress.

Ruby

SDK for Ruby

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
# Associates an Elastic IP address with an Amazon Elastic Compute Cloud
# (Amazon EC2) instance.
#
# Prerequisites:
#
# - The allocation ID corresponding to the Elastic IP address.
# - The Amazon EC2 instance.
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @param allocation_id [String] The ID of the allocation corresponding to
#   the Elastic IP address.
# @param instance_id [String] The ID of the instance.
# @return [String] The assocation ID corresponding to the association of the
#   Elastic IP address to the instance.
# @example
#   puts allocate_elastic_ip_address(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'eipalloc-04452e528a66279EX',
#     'i-033c48ef067af3dEX')
def associate_elastic_ip_address_with_instance(
  ec2_client,
  allocation_id,
  instance_id
)
  response = ec2_client.associate_address(
    allocation_id: allocation_id,
    instance_id: instance_id,
  )
  return response.association_id
rescue StandardError => e
  puts "Error associating Elastic IP address with instance: #{e.message}"
  return "Error"
```

```
end
```

- API 세부 정보는 AWS SDK for Ruby API AssociateAddress참조를 참조하십시오.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    TRY.
        oo_result = lo_ec2->associateaddress(                          " oo_result
 is returned for testing purposes. "
            iv_allocationid = iv_allocation_id
            iv_instanceid = iv_instance_id
        ).
        MESSAGE 'Associated an Elastic IP address with an EC2 instance.' TYPE
 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- API 세부 정보는 SAP용AWS SDK ABAP API 참조를 참조하십시오 AssociateAddress.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **AssociateDhcpOptions** CLI와 함께 사용

다음 코드 예제는 AssociateDhcpOptions의 사용 방법을 보여줍니다.

CLI

AWS CLI

DHCP 옵션 세트를 VPC와 연결하려면

이 예제는 지정된 DHCP 옵션 세트를 지정된 VPC와 연결합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 associate-dhcp-options --dhcp-options-id dopt-d9070ebb --vpc-id vpc-a01106c2
```

기본 DHCP 옵션 세트를 VPC와 연결하려면

이 예제는 기본 DHCP 옵션 세트를 지정된 VPC와 연결합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 associate-dhcp-options --dhcp-options-id default --vpc-id vpc-a01106c2
```

- API 세부 정보는 명령 참조를 참조하십시오 AssociateDhcpOptions.AWS CLI

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 DHCP 옵션 세트를 지정된 VPC와 연결합니다.

```
Register-EC2DhcpOption -DhcpOptionsId dopt-1a2b3c4d -VpcId vpc-12345678
```

예 2: 이 예제는 기본 DHCP 옵션 세트를 지정된 VPC와 연결합니다.

```
Register-EC2DhcpOption -DhcpOptionsId default -VpcId vpc-12345678
```

- API 세부 정보는 Cmdlet 참조를 참조하십시오 AssociateDhcpOptions.AWS Tools for PowerShell

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **AssociateRouteTable** CLI와 함께 사용

다음 코드 예제는 AssociateRouteTable의 사용 방법을 보여줍니다.

CLI

AWS CLI

라우팅 테이블을 서브넷에 연결하려면

이 예제는 지정된 라우팅 테이블을 지정된 서브넷에 연결합니다.

명령:

```
aws ec2 associate-route-table --route-table-id rtb-22574640 --subnet-id
 subnet-9d4a7b6c
```

출력:

```
{
    "AssociationId": "rtbassoc-781d0d1a"
}
```

- API 세부 정보는 AWS CLI 명령 AssociateRouteTable참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 라우팅 테이블을 지정된 서브넷과 연결합니다.

```
Register-EC2RouteTable -RouteTableId rtb-1a2b3c4d -SubnetId subnet-1a2b3c4d
```

출력:

```
rtbassoc-12345678
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 AssociateRouteTable.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **AttachInternetGateway** CLI와 함께 사용

다음 코드 예제는 AttachInternetGateway의 사용 방법을 보여줍니다.

CLI

AWS CLI

VPC에 인터넷 게이트웨이를 연결하는 방법

다음 attach-internet-gateway 예제는 지정된 인터넷 게이트웨이를 특정 VPC에 연결합니다.

```
aws ec2 attach-internet-gateway \
    --internet-gateway-id igw-0d0fb496b3EXAMPLE \
    --vpc-id vpc-0a60eb65b4EXAMPLE
```

이 명령은 출력을 생성하지 않습니다.

자세한 내용은 Amazon VPC 사용 설명서의 인터넷 게이트웨이를 참조하세요.

- API 세부 정보는 AWS CLI 명령 AttachInternetGateway참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 인터넷 게이트웨이를 지정된 VPC에 연결합니다.

```
Add-EC2InternetGateway -InternetGatewayId igw-1a2b3c4d -VpcId vpc-12345678
```

예 2: 이 예제는 VPC와 인터넷 게이트웨이를 만든 다음 인터넷 게이트웨이를 VPC에 연결합니다.

```
$vpc = New-EC2Vpc -CidrBlock 10.0.0.0/16
New-EC2InternetGateway | Add-EC2InternetGateway -VpcId $vpc.VpcId
```

- API 세부 정보는 Cmdlet 참조를 참조하십시오 AttachInternetGateway.AWS Tools for PowerShell

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **AttachNetworkInterface** CLI와 함께 사용

다음 코드 예제는 AttachNetworkInterface의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 인스턴스에 네트워크 인터페이스 연결하기

다음 attach-network-interface 예제는 지정된 네트워크 인터페이스를 지정된 인스턴스에 연결합니다.

```
aws ec2 attach-network-interface \
    --network-interface-id eni-0dc56a8d4640ad10a \
    --instance-id i-1234567890abcdef0 \
    --device-index 1
```

출력:

```
{
    "AttachmentId": "eni-attach-01a8fc87363f07cf9"
}
```

자세한 내용은 Amazon EC2 사용 설명서의 탄력적 네트워크 인터페이스를 참조하세요.

예 2: 여러 네트워크 카드가 있는 인스턴스에 네트워크 인터페이스 연결하기

다음 attach-network-interface 예제는 지정된 네트워크 인터페이스를 지정된 인스턴스와 네트워크 카드에 연결합니다.

```
aws ec2 attach-network-interface \
    --network-interface-id eni-07483b1897541ad83 \
    --instance-id i-01234567890abcdef \
    --network-card-index 1 \
    --device-index 1
```

출력:

```
{
    "AttachmentId": "eni-attach-0fbd7ee87a88cd06c"
}
```

자세한 내용은 Amazon EC2 사용 설명서의 탄력적 네트워크 인터페이스를 참조하세요.

- API 세부 정보는 AWS CLI 명령 AttachNetworkInterface참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 네트워크 인터페이스를 지정된 인스턴스에 연결합니다.

```
Add-EC2NetworkInterface -NetworkInterfaceId eni-12345678 -InstanceId i-1a2b3c4d -
DeviceIndex 1
```

출력:

```
eni-attach-1a2b3c4d
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 AttachNetworkInterface.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **AttachVolume** CLI와 함께 사용

다음 코드 예제는 AttachVolume의 사용 방법을 보여줍니다.

CLI

AWS CLI

볼륨을 인스턴스에 연결하려면

이 예제 명령은 볼륨 (vol-1234567890abcdef0) 을 인스턴스 (i-01474ef662b89480) 에 다음과 같이 /dev/sdf 연결합니다.

명령:

```
aws ec2 attach-volume --volume-id vol-1234567890abcdef0 --instance-id
 i-01474ef662b89480 --device /dev/sdf
```

출력:

```
{
    "AttachTime": "YYYY-MM-DDTHH:MM:SS.000Z",
    "InstanceId": "i-01474ef662b89480",
    "VolumeId": "vol-1234567890abcdef0",
    "State": "attaching",
    "Device": "/dev/sdf"
}
```

- API 세부 정보는 AWS CLI 명령 AttachVolume참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 볼륨을 지정된 인스턴스에 연결하고 지정된 장치 이름과 함께 노출합니다.

```
Add-EC2Volume -VolumeId vol-12345678 -InstanceId i-1a2b3c4d -Device /dev/sdh
```

출력:

```
AttachTime          : 12/22/2015 1:53:58 AM
DeleteOnTermination : False
```

```
Device              : /dev/sdh
InstanceId          : i-1a2b3c4d
State               : attaching
VolumeId            : vol-12345678
```

- API에 대한 자세한 내용은 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 AttachVolume.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **AttachVpnGateway** CLI와 함께 사용

다음 코드 예제는 AttachVpnGateway의 사용 방법을 보여줍니다.

CLI

AWS CLI

VPC에 가상 프라이빗 게이트웨이를 연결하는 방법

다음 attach-vpn-gateway 예제는 지정된 가상 프라이빗 게이트웨이를 지정된 VPC에 연결합니다.

```
aws ec2 attach-vpn-gateway \
    --vpn-gateway-id vgw-9a4cacf3 \
    --vpc-id vpc-a01106c2
```

출력:

```
{
    "VpcAttachment": {
        "State": "attaching",
        "VpcId": "vpc-a01106c2"
    }
}
```

- API 세부 정보는 AWS CLI 명령 AttachVpnGateway참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 가상 프라이빗 게이트웨이를 지정된 VPC에 연결합니다.

```
Add-EC2VpnGateway -VpnGatewayId vgw-1a2b3c4d -VpcId vpc-12345678
```

출력:

```
State        VpcId
-----        -----
attaching    vpc-12345678
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 AttachVpnGateway.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **AuthorizeSecurityGroupEgress** CLI와 함께 사용

다음 코드 예제는 AuthorizeSecurityGroupEgress의 사용 방법을 보여줍니다.

CLI

AWS CLI

특정 주소 범위로의 아웃바운드 트래픽을 허용하는 규칙을 추가하려면

이 예제 명령은 TCP 포트 80의 지정된 주소 범위에 대한 액세스 권한을 부여하는 규칙을 추가합니다.

명령(Linux):

```
aws ec2 authorize-security-group-egress --group-id sg-1a2b3c4d --ip-permissions
 IpProtocol=tcp,FromPort=80,ToPort=80,IpRanges='[{CidrIp=10.0.0.0/16}]'
```

명령(Windows):

```
aws ec2 authorize-security-group-egress --group-id sg-1a2b3c4d --ip-permissions
 IpProtocol=tcp,FromPort=80,ToPort=80,IpRanges=[{CidrIp=10.0.0.0/16}]
```

특정 보안 그룹에 대한 아웃바운드 트래픽을 허용하는 규칙을 추가하려면

이 예제 명령은 TCP 포트 80의 지정된 보안 그룹에 액세스 권한을 부여하는 규칙을 추가합니다.

명령(Linux):

```
aws ec2 authorize-security-group-egress --group-id sg-1a2b3c4d --ip-permissions
 IpProtocol=tcp,FromPort=80,ToPort=80,UserIdGroupPairs='[{GroupId=sg-4b51a32f}]'
```

명령(Windows):

```
aws ec2 authorize-security-group-egress --group-id sg-1a2b3c4d --ip-permissions
 IpProtocol=tcp,FromPort=80,ToPort=80,UserIdGroupPairs=[{GroupId=sg-4b51a32f}]
```

- API 세부 정보는 AWS CLI 명령 AuthorizeSecurityGroupEgress참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 EC2-VPC 보안 그룹에 대한 송신 규칙을 정의합니다. 이 규칙은 TCP 포트 80의 지정된 IP 주소 범위에 대한 액세스 권한을 부여합니다. 이 예제에서 사용하는 구문에는 PowerShell 버전 3 이상이 필요합니다.

```
$ip = @{ IpProtocol="tcp"; FromPort="80"; ToPort="80";
 IpRanges="203.0.113.0/24" }
Grant-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission $ip
```

예 2: PowerShell 버전 2에서는 New-Object를 사용하여 객체를 생성해야 합니다. IpPermission

```
$ip = New-Object Amazon.EC2.Model.IpPermission
$ip.IpProtocol = "tcp"
$ip.FromPort = 80
$ip.ToPort = 80
$ip.IpRanges.Add("203.0.113.0/24")
```

```
Grant-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission $ip
```

예 3: 이 예제는 TCP 포트 80의 지정된 소스 보안 그룹에 대한 액세스 권한을 부여합니다.

```
$ug = New-Object Amazon.EC2.Model.UserIdGroupPair
$ug.GroupId = "sg-1a2b3c4d"
$ug.UserId = "123456789012"

Grant-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission
 @( @{ IpProtocol="tcp"; FromPort="80"; ToPort="80"; UserIdGroupPairs=$ug } )
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 AuthorizeSecurityGroupEgress.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **AuthorizeSecurityGroupIngress** CLI와 함께 사용

다음 코드 예제는 AuthorizeSecurityGroupIngress의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 인스턴스 시작하기

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Authorize the local computer ingress to EC2 instances associated
    /// with the virtual private cloud (VPC) security group.
```

```
    /// </summary>
    /// <param name="groupName">The name of the security group.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AuthorizeSecurityGroupIngress(string groupName)
    {
        // Get the IP address for the local computer.
        var ipAddress = await GetIpAddress();
        Console.WriteLine($"Your IP address is: {ipAddress}");
        var ipRanges = new List<IpRange> { new IpRange { CidrIp =
 $"{ipAddress}/32" } };
        var permission = new IpPermission
        {
            Ipv4Ranges = ipRanges,
            IpProtocol = "tcp",
            FromPort = 22,
            ToPort = 22
        };
        var permissions = new List<IpPermission> { permission };
        var response = await _amazonEC2.AuthorizeSecurityGroupIngressAsync(
            new AuthorizeSecurityGroupIngressRequest(groupName, permissions));
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Authorize the local computer for ingress to
    /// the Amazon EC2 SecurityGroup.
    /// </summary>
    /// <returns>The IPv4 address of the computer running the scenario.</returns>
    private static async Task<string> GetIpAddress()
    {
        var httpClient = new HttpClient();
        var ipString = await httpClient.GetStringAsync("https://
checkip.amazonaws.com");

        // The IP address is returned with a new line
        // character on the end. Trim off the whitespace and
        // return the value to the caller.
        return ipString.Trim();
    }
```

- API 세부 정보는 AWS SDK for .NET API AuthorizeSecurityGroupIngress참조를 참조하십시오.

Bash

AWS CLI Bash 스크립트 사용

> **Note**
>
> 더 많은 정보가 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
#############################################################################
# function ec2_authorize_security_group_ingress
#
# This function authorizes an ingress rule for an Amazon Elastic Compute Cloud
 (Amazon EC2) security group.
#
# Parameters:
#       -g security_group_id - The ID of the security group.
#       -i ip_address - The IP address or CIDR block to authorize.
#       -p protocol - The protocol to authorize (e.g., tcp, udp, icmp).
#       -f from_port - The start of the port range to authorize.
#       -t to_port - The end of the port range to authorize.
#
# And:
#       0 - If successful.
#       1 - If it fails.
#############################################################################
function ec2_authorize_security_group_ingress() {
  local security_group_id ip_address protocol from_port to_port response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_authorize_security_group_ingress"
    echo "Authorizes an ingress rule for an Amazon Elastic Compute Cloud (Amazon
 EC2) security group."
    echo "  -g security_group_id - The ID of the security group."
    echo "  -i ip_address - The IP address or CIDR block to authorize."
    echo "  -p protocol - The protocol to authorize (e.g., tcp, udp, icmp)."
    echo "  -f from_port - The start of the port range to authorize."
    echo "  -t to_port - The end of the port range to authorize."
    echo ""
```

```
}

# Retrieve the calling parameters.
while getopts "g:i:p:f:t:h" option; do
  case "${option}" in
    g) security_group_id="${OPTARG}" ;;
    i) ip_address="${OPTARG}" ;;
    p) protocol="${OPTARG}" ;;
    f) from_port="${OPTARG}" ;;
    t) to_port="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$security_group_id" ]]; then
  errecho "ERROR: You must provide a security group ID with the -g parameter."
  usage
  return 1
fi

if [[ -z "$ip_address" ]]; then
  errecho "ERROR: You must provide an IP address or CIDR block with the -i
parameter."
  usage
  return 1
fi

if [[ -z "$protocol" ]]; then
  errecho "ERROR: You must provide a protocol with the -p parameter."
  usage
  return 1
fi

if [[ -z "$from_port" ]]; then
  errecho "ERROR: You must provide a start port with the -f parameter."
```

```
    usage
    return 1
  fi

  if [[ -z "$to_port" ]]; then
    errecho "ERROR: You must provide an end port with the -t parameter."
    usage
    return 1
  fi

  response=$(aws ec2 authorize-security-group-ingress \
    --group-id "$security_group_id" \
    --cidr "${ip_address}/32" \
    --protocol "$protocol" \
    --port "$from_port-$to_port" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports authorize-security-group-ingress operation
 failed.$response"
    return 1
  }

  return 0
}
```

이 예제에 사용된 유틸리티 함수

```
########################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
########################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

########################################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
```

```
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
################################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- API 세부 정보는 AWS CLI 명령 AuthorizeSecurityGroupIngress참조를 참조하십시오.

C++

SDK for C++

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
```

```
    Aws::EC2::Model::IpRange ip_range;
    ip_range.SetCidrIp("0.0.0.0/0");

    Aws::EC2::Model::IpPermission permission1;
    permission1.SetIpProtocol("tcp");
    permission1.SetToPort(80);
    permission1.SetFromPort(80);
    permission1.AddIpRanges(ip_range);

    authorize_request.AddIpPermissions(permission1);

    Aws::EC2::Model::IpPermission permission2;
    permission2.SetIpProtocol("tcp");
    permission2.SetToPort(22);
    permission2.SetFromPort(22);
    permission2.AddIpRanges(ip_range);

    authorize_request.AddIpPermissions(permission2);

    const Aws::EC2::Model::AuthorizeSecurityGroupIngressOutcome authorizeOutcome
=
            ec2Client.AuthorizeSecurityGroupIngress(authorizeRequest);

    if (!authorizeOutcome.IsSuccess()) {
        std::cerr << "Failed to set ingress policy for security group " <<
                  groupName << ":" << authorizeOutcome.GetError().GetMessage() <<
                  std::endl;
        return false;
    }

    std::cout << "Successfully added ingress policy to security group " <<
              groupName << std::endl;
```

- API 세부 정보는 AWS SDK for C++ API AuthorizeSecurityGroupIngress참조를 참조하십시오.

CLI

AWS CLI

예제 1: 인바운드 SSH 트래픽을 허용하는 규칙을 추가하는 방법

다음 authorize-security-group-ingress 예제에서는 TCP 포트 22(SSH)의 인바운드 트래픽을 허용하는 규칙을 추가합니다.

```
aws ec2 authorize-security-group-ingress \
    --group-id sg-1234567890abcdef0 \
    --protocol tcp \
    --port 22 \
    --cidr 203.0.113.0/24
```

출력:

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-01afa97ef3e1bedfc",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIpv4": "203.0.113.0/24"
        }
    ]
}
```

예제 2: 다른 보안 그룹의 인바운드 HTTP 트래픽을 허용하는 규칙을 추가하는 방법

다음 authorize-security-group-ingress 예제에서는 소스 보안 그룹 sg-1a2b3c4d에서 TCP 포트 80의 인바운드 액세스를 허용하는 규칙을 추가합니다. 보안 그룹은 동일한 VPC 또는 피어 VPC에 있어야 합니다(VPC 피어링 연결이 필요함). 유입 트래픽은 퍼블릭 IP 주소 또는 탄력적 IP 주소가 아닌 소스 보안 그룹과 연결된 인스턴스의 프라이빗 IP 주소를 기반으로 허용됩니다.

```
aws ec2 authorize-security-group-ingress \
    --group-id sg-1234567890abcdef0 \
    --protocol tcp \
    --port 80 \
    --source-group sg-1a2b3c4d
```

출력:

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-01f4be99110f638a7",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "ReferencedGroupInfo": {
                "GroupId": "sg-1a2b3c4d",
                "UserId": "123456789012"
            }
        }
    ]
}
```

예제 3: 동일한 직접 호출에서 여러 규칙을 추가하는 방법

다음 `authorize-security-group-ingress` 예제에서는 `ip-permissions` 파라미터를 사용하여 TCP 포트 3389(RDP)의 인바운드 액세스를 허용하는 하나의 인바운드 규칙과 Ping/ICMP를 허용하는 다른 인바운드 규칙(총 2개)을 추가합니다.

aws ec2 authorize-security-group-ingress --group-id sg-1234567890abcdef0 --ip-permissions IpProtocol =tcp, =3389, FromPort =3389, = "[{=172.31.0.0/16}]” =icmp, =-1, = "[{=172.31.0.0/16}ToPort]” IpRanges CidrIp IpProtocol FromPort ToPort IpRanges CidrIp

출력:

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-00e06e5d3690f29f3",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 3389,
```

```
            "ToPort": 3389,
            "CidrIpv4": "172.31.0.0/16"
        },
        {
            "SecurityGroupRuleId": "sgr-0a133dd4493944b87",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": -1,
            "ToPort": -1,
            "CidrIpv4": "172.31.0.0/16"
        }
    ]
}
```

예제 4: ICMP 트래픽에 대한 규칙을 추가하는 방법

다음 `authorize-security-group-ingress` 예제에서는 `ip-permissions` 파라미터를 사용하여 어디서나 ICMP 메시지 `Destination Unreachable: Fragmentation Needed and Don't Fragment was Set`(유형 3, 코드 4)를 허용하는 인바운드 규칙을 추가합니다.

aws ec2 authorize-security-group-ingress --group-id sg-1234567890abcdef0 --ip-permissions IpProtocol =icmp, FromPort =3, ToPort =4, IpRanges = "[{CidrIp=0.0.0.0/0}]”

출력:

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0de3811019069b787",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "icmp",
            "FromPort": 3,
            "ToPort": 4,
            "CidrIpv4": "0.0.0.0/0"
        }
    ]
}
```

예제 5: IPv6 트래픽에 대한 규칙을 추가하는 방법

다음 `authorize-security-group-ingress` 예제에서는 `ip-permissions` 파라미터를 사용하여 IPv6 범위 `2001:db8:1234:1a00::/64`에서 SSH 액세스(포트 22)를 허용하는 인바운드 규칙을 추가합니다.

aws ec2 authorize-security-group-ingress --group-id sg-1234567890abcdef0 --ip-permissions IpProtocol =tcp, FromPort =22, Ipv6Ranges= "[{ToPortCidrIpv6=2001:db 8:1234:1 a00: :/64}]”

출력:

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0455bc68b60805563",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIpv6": "2001:db8:1234:1a00::/64"
        }
    ]
}
```

예제 6: ICMPv6 트래픽에 대한 규칙을 추가하는 방법

다음 `authorize-security-group-ingress` 예제에서는 `ip-permissions` 파라미터를 사용하여 어디서나 ICMPv6 트래픽을 허용하는 인바운드 규칙을 추가합니다.

aws ec2 authorize-security-group-ingress --group-id sg-1234567890abcdef0 --ip-permissions IpProtocol =icmpv6, Ipv6Ranges= "[{CidrIpv6=: :/0}]”

출력:

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
```

```
            "SecurityGroupRuleId": "sgr-04b612d9363ab6327",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "icmpv6",
            "FromPort": -1,
            "ToPort": -1,
            "CidrIpv6": "::/0"
        }
    ]
}
```

예제 7: 설명이 포함된 규칙 추가

다음 `authorize-security-group-ingress` 예제에서는 `ip-permissions` 파라미터를 사용하여 지정된 IPv4 주소 범위에서 RDP 트래픽을 허용하는 인바운드 규칙을 추가합니다. 이 규칙에는 나중에 식별하는 데 도움이 되는 설명이 포함됩니다.

aws ec2 authorize-security-group-ingress --group-id sg-1234567890abcdef0 --ip-permissions IpProtocol =tcp, =3389, FromPort =3389, IpRanges = "[{CidrIp=203.0.113.0/24, ToPort 설명 ='뉴욕 사무소에서 RDP 액세스'}]”

출력:

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0397bbcc01e974db3",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 3389,
            "ToPort": 3389,
            "CidrIpv4": "203.0.113.0/24",
            "Description": "RDP access from NY office"
        }
    ]
}
```

예제 8: 접두사 목록을 사용하는 인바운드 규칙을 추가하는 방법

다음 `authorize-security-group-ingress` 예제에서는 `ip-permissions` 파라미터를 사용하여 지정된 접두사 목록에 있는 CIDR 범위의 모든 트래픽을 허용하는 인바운드 규칙을 추가합니다.

aws ec2 authorize-security-group-ingress --group-id sg-04a351bfe432d4e71 --ip-permissions IpProtocol =all, PrefixListIds = "[{PrefixListId=pl-002dc3ec097de1514}]”

출력:

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-09c74b32f677c6c7c",
            "GroupId": "sg-1234567890abcdef0",
            "GroupOwnerId": "123456789012",
            "IsEgress": false,
            "IpProtocol": "-1",
            "FromPort": -1,
            "ToPort": -1,
            "PrefixListId": "pl-0721453c7ac4ec009"
        }
    ]
}
```

자세한 내용을 알아보려면 Amazon VPC 사용 설명서의 보안 그룹을 참조하세요.

- API에 대한 자세한 내용은 명령 참조를 참조하십시오. AuthorizeSecurityGroupIngressAWS CLI

Java

SDK for Java 2.x

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
   public static String createSecurityGroup(Ec2Client ec2, String groupName,
String groupDesc, String vpcId,
           String myIpAddress) {
       try {
           CreateSecurityGroupRequest createRequest =
CreateSecurityGroupRequest.builder()
                   .groupName(groupName)
                   .description(groupDesc)
                   .vpcId(vpcId)
                   .build();

           CreateSecurityGroupResponse resp =
ec2.createSecurityGroup(createRequest);
           IpRange ipRange = IpRange.builder()
                   .cidrIp(myIpAddress + "/0")
                   .build();

           IpPermission ipPerm = IpPermission.builder()
                   .ipProtocol("tcp")
                   .toPort(80)
                   .fromPort(80)
                   .ipRanges(ipRange)
                   .build();

           IpPermission ipPerm2 = IpPermission.builder()
                   .ipProtocol("tcp")
                   .toPort(22)
                   .fromPort(22)
                   .ipRanges(ipRange)
                   .build();

           AuthorizeSecurityGroupIngressRequest authRequest =
AuthorizeSecurityGroupIngressRequest.builder()
                   .groupName(groupName)
                   .ipPermissions(ipPerm, ipPerm2)
                   .build();

           ec2.authorizeSecurityGroupIngress(authRequest);
           System.out.println("Successfully added ingress policy to security
group " + groupName);
           return resp.groupId();

       } catch (Ec2Exception e) {
```

```
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API AuthorizeSecurityGroupIngress참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { AuthorizeSecurityGroupIngressCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

// Grant permissions for a single IP address to ssh into instances
// within the provided security group.
export const main = async () => {
  const command = new AuthorizeSecurityGroupIngressCommand({
    // Replace with a security group ID from the AWS console or
    // the DescribeSecurityGroupsCommand.
    GroupId: "SECURITY_GROUP_ID",
    IpPermissions: [
      {
        IpProtocol: "tcp",
        FromPort: 22,
        ToPort: 22,
        // Replace 0.0.0.0 with the IP address to authorize.
        // For more information on this notation, see
        // https://en.wikipedia.org/wiki/Classless_Inter-
Domain_Routing#CIDR_notation
        IpRanges: [{ CidrIp: "0.0.0.0/32" }],
```

```
      },
    ],
  });

  try {
    const { SecurityGroupRules } = await client.send(command);
    console.log(JSON.stringify(SecurityGroupRules, null, 2));
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API AuthorizeSecurityGroupIngress참조를 참조하십시오.

Kotlin

SDK for Kotlin

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
suspend fun createEC2SecurityGroupSc(
    groupNameVal: String?,
    groupDescVal: String?,
    vpcIdVal: String?,
    myIpAddress: String?,
): String? {
    val request =
        CreateSecurityGroupRequest {
            groupName = groupNameVal
            description = groupDescVal
            vpcId = vpcIdVal
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val resp = ec2.createSecurityGroup(request)
```

```
        val ipRange =
            IpRange {
                cidrIp = "$myIpAddress/0"
            }

        val ipPerm =
            IpPermission {
                ipProtocol = "tcp"
                toPort = 80
                fromPort = 80
                ipRanges = listOf(ipRange)
            }

        val ipPerm2 =
            IpPermission {
                ipProtocol = "tcp"
                toPort = 22
                fromPort = 22
                ipRanges = listOf(ipRange)
            }

        val authRequest =
            AuthorizeSecurityGroupIngressRequest {
                groupName = groupNameVal
                ipPermissions = listOf(ipPerm, ipPerm2)
            }
        ec2.authorizeSecurityGroupIngress(authRequest)
        println("Successfully added ingress policy to Security Group
 $groupNameVal")
        return resp.groupId
    }
}
```

- API 세부 정보는 Kotlin API용AWS SDK 레퍼런스를 참조하세요 AuthorizeSecurityGroupIngress.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예에서는 EC2-VPC 보안 그룹의 인그레스 규칙을 정의합니다. 이 규칙은 SSH (포트 22) 및 RDC (포트 3389) 의 특정 IP 주소에 대한 액세스 권한을 부여합니다. 보안 그룹 이름이

아닌 보안 그룹 ID를 사용하여 EC2-VPC 보안 그룹을 식별해야 한다는 점에 유의하십시오. 이 예제에서 사용하는 구문에는 PowerShell 버전 3 이상이 필요합니다.

```
$ip1 = @{ IpProtocol="tcp"; FromPort="22"; ToPort="22";
 IpRanges="203.0.113.25/32" }
$ip2 = @{ IpProtocol="tcp"; FromPort="3389"; ToPort="3389";
 IpRanges="203.0.113.25/32" }

Grant-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission @( $ip1, $ip2 )
```

예 2: PowerShell 버전 2에서는 New-Object를 사용하여 객체를 생성해야 합니다. IpPermission

```
$ip1 = New-Object Amazon.EC2.Model.IpPermission
$ip1.IpProtocol = "tcp"
$ip1.FromPort = 22
$ip1.ToPort = 22
$ip1.IpRanges.Add("203.0.113.25/32")

$ip2 = new-object Amazon.EC2.Model.IpPermission
$ip2.IpProtocol = "tcp"
$ip2.FromPort = 3389
$ip2.ToPort = 3389
$ip2.IpRanges.Add("203.0.113.25/32")

Grant-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission @( $ip1, $ip2 )
```

예 3: 이 예에서는 EC2-Classic의 보안 그룹에 대한 인그레스 규칙을 정의합니다. 이 규칙은 SSH (포트 22) 및 RDC (포트 3389) 의 특정 IP 주소에 대한 액세스 권한을 부여합니다. 이 예제에서 사용하는 구문에는 PowerShell 버전 3 이상이 필요합니다.

```
$ip1 = @{ IpProtocol="tcp"; FromPort="22"; ToPort="22";
 IpRanges="203.0.113.25/32" }
$ip2 = @{ IpProtocol="tcp"; FromPort="3389"; ToPort="3389";
 IpRanges="203.0.113.25/32" }

Grant-EC2SecurityGroupIngress -GroupName "my-security-group" -IpPermission
 @( $ip1, $ip2 )
```

예 4: PowerShell 버전 2에서는 New-Object를 사용하여 객체를 생성해야 합니다. IpPermission

```
$ip1 = New-Object Amazon.EC2.Model.IpPermission
```

```
$ip1.IpProtocol = "tcp"
$ip1.FromPort = 22
$ip1.ToPort = 22
$ip1.IpRanges.Add("203.0.113.25/32")

$ip2 = new-object Amazon.EC2.Model.IpPermission
$ip2.IpProtocol = "tcp"
$ip2.FromPort = 3389
$ip2.ToPort = 3389
$ip2.IpRanges.Add("203.0.113.25/32")

Grant-EC2SecurityGroupIngress -GroupName "my-security-group" -IpPermission
 @( $ip1, $ip2 )
```

예 5: 이 예에서는 지정된 소스 보안 그룹 (sg-1a2b3c4d) 에서 지정된 보안 그룹 (sg-12345678) 에 대한 TCP 포트 8081 액세스 권한을 부여합니다.

```
$ug = New-Object Amazon.EC2.Model.UserIdGroupPair
$ug.GroupId = "sg-1a2b3c4d"
$ug.UserId = "123456789012"

Grant-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission
 @( @{ IpProtocol="tcp"; FromPort="8081"; ToPort="8081"; UserIdGroupPairs=$ug } )
```

예 6: 이 예에서는 TCP 포트 22 트래픽에 대한 CIDR 5.5.5.5/32를 TCP 포트 22 트래픽에 대한 보안 그룹 sg-1234abcd의 인그레스 규칙에 설명과 함께 추가합니다.

```
$IpRange = New-Object -TypeName Amazon.EC2.Model.IpRange
$IpRange.CidrIp = "5.5.5.5/32"
$IpRange.Description = "SSH from Office"
$IpPermission = New-Object Amazon.EC2.Model.IpPermission
$IpPermission.IpProtocol = "tcp"
$IpPermission.ToPort = 22
$IpPermission.FromPort = 22
$IpPermission.Ipv4Ranges = $IpRange
Grant-EC2SecurityGroupIngress -GroupId sg-1234abcd -IpPermission $IpPermission
```

- API AuthorizeSecurityGroupIngress세부AWS Tools for PowerShell 정보는 Cmdlet 참조를 참조하십시오.

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SecurityGroupWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) security group
 actions."""

    def __init__(self, ec2_resource, security_group=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param security_group: A Boto3 SecurityGroup object. This is a high-level
 object
                               that wraps security group actions.
        """
        self.ec2_resource = ec2_resource
        self.security_group = security_group

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def authorize_ingress(self, ssh_ingress_ip):
        """
        Adds a rule to the security group to allow access to SSH.

        :param ssh_ingress_ip: The IP address that is granted inbound access to
 connect
                               to port 22 over TCP, used for SSH.
        :return: The response to the authorization request. The 'Return' field of
 the
```

```
            response indicates whether the request succeeded or failed.
    """
    if self.security_group is None:
        logger.info("No security group to update.")
        return

    try:
        ip_permissions = [
            {
                # SSH ingress open to only the specified IP address.
                "IpProtocol": "tcp",
                "FromPort": 22,
                "ToPort": 22,
                "IpRanges": [{"CidrIp": f"{ssh_ingress_ip}/32"}],
            }
        ]
        response = self.security_group.authorize_ingress(
            IpPermissions=ip_permissions
        )
    except ClientError as err:
        logger.error(
            "Couldn't authorize inbound rules for %s. Here's why: %s: %s",
            self.security_group.id,
            err.response["Error"]["Code"],
            err.response["Error"]["Message"],
        )
        raise
    else:
        return response
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 AuthorizeSecurityGroupIngress.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CancelCapacityReservation** CLI와 함께 사용

다음 코드 예제는 CancelCapacityReservation의 사용 방법을 보여줍니다.

CLI

AWS CLI

용량 예약을 취소하려면

다음 `cancel-capacity-reservation` 예시에서는 지정된 용량 예약을 취소합니다.

```
aws ec2 cancel-capacity-reservation \
    --capacity-reservation-id cr-1234abcd56EXAMPLE
```

출력:

```
{
    "Return": true
}
```

자세한 내용은 Linux 인스턴스용 Amazon Elastic Compute 클라우드 사용 설명서의 용량 예약 취소를 참조하십시오.

- API 세부 정보는 AWS CLI 명령 CancelCapacityReservation참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 용량 예약을 취소합니다. cr-0c1f2345db6f7cdba

```
Remove-EC2CapacityReservation -CapacityReservationId cr-0c1f2345db6f7cdba
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EC2CapacityReservation
 (CancelCapacityReservation)" on target "cr-0c1f2345db6f7cdba".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): y
True
```

- AWS Tools for PowerShell API에 대한 CancelCapacityReservation자세한 내용은 Cmdlet 참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CancelImportTask** CLI와 함께 사용

다음 코드 예제는 CancelImportTask의 사용 방법을 보여줍니다.

CLI

AWS CLI

가져오기 작업을 취소하려면

다음 cancel-import-task 예제에서는 지정된 이미지 가져오기 작업을 취소합니다.

```
aws ec2 cancel-import-task \
    --import-task-id import-ami-1234567890abcdef0
```

출력:

```
{
    "ImportTaskId": "import-ami-1234567890abcdef0",
    "PreviousState": "active",
    "State": "deleting"
}
```

- API 세부 정보는 AWS CLI 명령 CancelImportTask참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 가져오기 작업 (스냅샷 또는 이미지 가져오기) 을 취소합니다. 필요한 경우 **-CancelReason** 매개변수를 사용하여 이유를 제공할 수 있습니다.

```
Stop-EC2ImportTask -ImportTaskId import-ami-abcdefgh
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CancelImportTask.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CancelSpotFleetRequests** CLI와 함께 사용

다음 코드 예제는 CancelSpotFleetRequests의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 스팟 플릿 요청을 취소하고 관련 인스턴스를 종료하려면

다음 cancel-spot-fleet-requests 예제는 스팟 플릿 요청을 취소하고 연결된 온디맨드 인스턴스와 스팟 인스턴스를 종료합니다.

```
aws ec2 cancel-spot-fleet-requests \
    --spot-fleet-request-ids sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --terminate-instances
```

출력:

```
{
    "SuccessfulFleetRequests": [
        {
            "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
            "CurrentSpotFleetRequestState": "cancelled_terminating",
            "PreviousSpotFleetRequestState": "active"
        }
    ],
    "UnsuccessfulFleetRequests": []
}
```

자세한 내용은 Linux 인스턴스용 Amazon Elastic Compute 클라우드 사용 설명서의 스팟 플릿 요청 취소를 참조하십시오.

예 2: 연결된 인스턴스를 종료하지 않고 스팟 플릿 요청을 취소하려면

다음 cancel-spot-fleet-requests 예제는 연결된 온디맨드 인스턴스 및 스팟 인스턴스를 종료하지 않고 스팟 플릿 요청을 취소합니다.

```
aws ec2 cancel-spot-fleet-requests \
    --spot-fleet-request-ids sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --no-terminate-instances
```

출력:

```
{
    "SuccessfulFleetRequests": [
        {
            "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
            "CurrentSpotFleetRequestState": "cancelled_running",
            "PreviousSpotFleetRequestState": "active"
        }
    ],
    "UnsuccessfulFleetRequests": []
}
```

자세한 내용은 Linux 인스턴스용 Amazon Elastic Compute 클라우드 사용 설명서의 스팟 플릿 요청 취소를 참조하십시오.

- API 세부 정보는 AWS CLI 명령 CancelSpotFleetRequests참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 스팟 플릿 요청을 취소하고 연결된 스팟 인스턴스를 종료합니다.

```
Stop-EC2SpotFleetRequest -SpotFleetRequestId sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE -TerminateInstance $true
```

예 2: 이 예제는 연결된 스팟 인스턴스를 종료하지 않고 지정된 스팟 플릿 요청을 취소합니다.

```
Stop-EC2SpotFleetRequest -SpotFleetRequestId sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE -TerminateInstance $false
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CancelSpotFleetRequests.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CancelSpotInstanceRequests** CLI와 함께 사용

다음 코드 예제는 CancelSpotInstanceRequests의 사용 방법을 보여줍니다.

CLI

AWS CLI

스팟 인스턴스 요청을 취소하려면

이 예제 명령은 스팟 인스턴스 요청을 취소합니다.

명령:

```
aws ec2 cancel-spot-instance-requests --spot-instance-request-ids sir-08b93456
```

출력:

```
{
    "CancelledSpotInstanceRequests": [
        {
            "State": "cancelled",
            "SpotInstanceRequestId": "sir-08b93456"
        }
    ]
}
```

- API 세부 정보는 AWS CLI 명령 CancelSpotInstanceRequests참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 스팟 인스턴스 요청을 취소합니다.

```
Stop-EC2SpotInstanceRequest -SpotInstanceRequestId sir-12345678
```

출력:

```
SpotInstanceRequestId    State
---------------------    -----
sir-12345678             cancelled
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CancelSpotInstanceRequests.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ConfirmProductInstance** CLI와 함께 사용

다음 코드 예제는 ConfirmProductInstance의 사용 방법을 보여줍니다.

CLI

AWS CLI

제품 인스턴스를 확인하려면

이 예제는 지정된 제품 코드가 지정된 인스턴스와 연결되어 있는지 여부를 확인합니다.

명령:

```
aws ec2 confirm-product-instance --product-code 774F4FF8 --instance-id
 i-1234567890abcdef0
```

출력:

```
{
  "OwnerId": "123456789012"
}
```

- API 세부 정보는 AWS CLI 명령 ConfirmProductInstance참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 제품 코드가 지정된 인스턴스와 연결되어 있는지 여부를 확인합니다.

```
Confirm-EC2ProductInstance -ProductCode 774F4FF8 -InstanceId i-12345678
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 ConfirmProductInstance.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CopyImage** CLI와 함께 사용

다음 코드 예제는 CopyImage의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: AMI를 다른 지역으로 복사하려면

다음 copy-image 예제 명령은 지정된 AMI를 us-west-2 리전에서 리전으로 복사하고 간단한 설명을 추가합니다. us-east-1

```
aws ec2 copy-image \
    --region us-east-1 \
    --name ami-name \
    --source-region us-west-2 \
    --source-image-id ami-066877671789bd71b \
    --description "This is my copied image."
```

출력:

```
{
    "ImageId": "ami-0123456789abcdefg"
}
```

자세한 내용은 Amazon EC2 사용 설명서의 AMI 복사를 참조하십시오.

예 2: AMI를 다른 지역에 복사하고 지원 스냅샷을 암호화하는 방법

다음 copy-image 명령은 지정된 AMI를 us-west-2 지역에서 현재 지역으로 복사하고 지정된 KMS 키를 사용하여 백업 스냅샷을 암호화합니다.

```
aws ec2 copy-image \
    --source-region us-west-2 \
    --name ami-name \
    --source-image-id ami-066877671789bd71b \
    --encrypted \
    --kms-key-id alias/my-kms-key
```

출력:

```
{
    "ImageId": "ami-0123456789abcdefg"
}
```

자세한 내용은 Amazon EC2 사용 설명서의 AMI 복사를 참조하십시오.

예 3: AMI를 복사할 때 사용자 정의 AMI 태그를 포함하려면

다음 `copy-image` 명령은 AMI를 복사할 때 `--copy-image-tags` 파라미터를 사용하여 사용자 정의 AMI 태그를 복사합니다.

```
aws ec2 copy-image \
    --region us-east-1 \
    --name ami-name \
    --source-region us-west-2 \
    --source-image-id ami-066877671789bd71b \
    --description "This is my copied image." 
    --copy-image-tags
```

출력:

```
{
    "ImageId": "ami-0123456789abcdefg"
}
```

자세한 내용은 Amazon EC2 사용 설명서의 AMI 복사를 참조하십시오.

- API 세부 정보는 AWS CLI 명령 CopyImage참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 'EU (아일랜드) '지역의 지정된 AMI를 '미국 서부 (오레곤)' 지역으로 복사합니다. -Region이 지정되지 않은 경우 현재 기본 지역이 대상 지역으로 사용됩니다.

```
Copy-EC2Image -SourceRegion eu-west-1 -SourceImageId ami-12345678 -Region us-
west-2 -Name "Copy of ami-12345678"
```

출력:

```
ami-87654321
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CopyImage.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CopySnapshot** CLI와 함께 사용

다음 코드 예제는 CopySnapshot의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 스냅샷을 다른 지역으로 복사하려면

다음 copy-snapshot 예제 명령은 지정된 스냅샷을 us-west-2 지역에서 지역으로 복사하고 간단한 설명을 추가합니다. us-east-1

```
aws ec2 copy-snapshot \
    --region us-east-1 \
    --source-region us-west-2 \
    --source-snapshot-id snap-066877671789bd71b \
    --description "This is my copied snapshot."
```

출력:

```
{
    "SnapshotId": "snap-066877671789bd71b"
}
```

자세한 내용은 Amazon EC2 사용 설명서의 Amazon EBS 스냅샷 복사를 참조하십시오.

예 2: 암호화되지 않은 스냅샷을 복사하고 새 스냅샷을 암호화하려면

다음 copy-snapshot 명령은 지정된 암호화되지 않은 스냅샷을 us-west-2 지역의 암호화되지 않은 스냅샷을 현재 지역으로 복사하고 지정된 KMS 키를 사용하여 새 스냅샷을 암호화합니다.

```
aws ec2 copy-snapshot \
    --source-region us-west-2 \
    --source-snapshot-id snap-066877671789bd71b \
    --encrypted \
    --kms-key-id alias/my-kms-key
```

출력:

```
{
    "SnapshotId": "snap-066877671789bd71b"
}
```

자세한 내용은 Amazon EC2 사용 설명서의 Amazon EBS 스냅샷 복사를 참조하십시오.

- API 세부 정보는 AWS CLI 명령 CopySnapshot참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 스냅샷을 EU (아일랜드) 지역에서 미국 서부 (오레곤) 지역으로 복사합니다.

```
Copy-EC2Snapshot -SourceRegion eu-west-1 -SourceSnapshotId snap-12345678 -Region
 us-west-2
```

예 2: 기본 지역을 설정하고 지역 매개 변수를 생략한 경우 기본 대상 지역이 기본 지역이 됩니다.

```
Set-DefaultAWSRegion us-west-2
Copy-EC2Snapshot -SourceRegion eu-west-1 -SourceSnapshotId snap-12345678
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CopySnapshot.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateCapacityReservation** CLI와 함께 사용

다음 코드 예제는 CreateCapacityReservation의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 용량 예약을 생성하려면

다음 create-capacity-reservation 예시에서는 가용 eu-west-1a 영역에 용량 예약을 생성하여 Linux/Unix 운영 체제를 실행하는 세 개의 t2.medium 인스턴스를 시작할 수 있습니다. 기본적으로 용량 예약은 임시 스토리지를 지원하지 않는 오픈 인스턴스 매칭 기준으로 생성되며 수동으로 취소할 때까지 활성 상태로 유지됩니다.

```
aws ec2 create-capacity-reservation \
    --availability-zone eu-west-1a \
    --instance-type t2.medium \
    --instance-platform Linux/UNIX \
    --instance-count 3
```

출력:

```
{
    "CapacityReservation": {
        "CapacityReservationId": "cr-1234abcd56EXAMPLE ",
        "EndDateType": "unlimited",
        "AvailabilityZone": "eu-west-1a",
        "InstanceMatchCriteria": "open",
        "EphemeralStorage": false,
        "CreateDate": "2019-08-16T09:27:35.000Z",
        "AvailableInstanceCount": 3,
```

```
        "InstancePlatform": "Linux/UNIX",
        "TotalInstanceCount": 3,
        "State": "active",
        "Tenancy": "default",
        "EbsOptimized": false,
        "InstanceType": "t2.medium"
    }
}
```

예 2: 지정된 날짜/시간에 자동으로 종료되는 용량 예약 생성하기

다음 `create-capacity-reservation` 예제에서는 가용 `eu-west-1a` 영역에 용량 예약을 생성하여 Linux/Unix 운영 체제를 실행하는 세 개의 `m5.large` 인스턴스를 시작할 수 있습니다. 이 용량 예약은 2019년 8월 31일 23:59:59 에 자동으로 종료됩니다.

```
aws ec2 create-capacity-reservation \
    --availability-zone eu-west-1a \
    --instance-type m5.large \
    --instance-platform Linux/UNIX \
    --instance-count 3 \
    --end-date-type limited \
    --end-date 2019-08-31T23:59:59Z
```

출력:

```
{
    "CapacityReservation": {
        "CapacityReservationId": "cr-1234abcd56EXAMPLE ",
        "EndDateType": "limited",
        "AvailabilityZone": "eu-west-1a",
        "EndDate": "2019-08-31T23:59:59.000Z",
        "InstanceMatchCriteria": "open",
        "EphemeralStorage": false,
        "CreateDate": "2019-08-16T10:15:53.000Z",
        "AvailableInstanceCount": 3,
        "InstancePlatform": "Linux/UNIX",
        "TotalInstanceCount": 3,
        "State": "active",
        "Tenancy": "default",
        "EbsOptimized": false,
        "InstanceType": "m5.large"
    }
```

```
}
```

예 3: 대상 인스턴스 시작만 허용하는 용량 예약 생성하기

다음 `create-capacity-reservation` 예제는 대상 인스턴스 시작만 허용하는 용량 예약을 생성합니다.

```
aws ec2 create-capacity-reservation \
    --availability-zone eu-west-1a \
    --instance-type m5.large \
    --instance-platform Linux/UNIX \
    --instance-count 3 \
    --instance-match-criteria targeted
```

출력:

```
{
    "CapacityReservation": {
        "CapacityReservationId": "cr-1234abcd56EXAMPLE ",
        "EndDateType": "unlimited",
        "AvailabilityZone": "eu-west-1a",
        "InstanceMatchCriteria": "targeted",
        "EphemeralStorage": false,
        "CreateDate": "2019-08-16T10:21:57.000Z",
        "AvailableInstanceCount": 3,
        "InstancePlatform": "Linux/UNIX",
        "TotalInstanceCount": 3,
        "State": "active",
        "Tenancy": "default",
        "EbsOptimized": false,
        "InstanceType": "m5.large"
    }
}
```

자세한 내용은 Linux 인스턴스용 Amazon Elastic Compute 클라우드 사용 설명서의 용량 예약 생성을 참조하십시오.

- API 세부 정보는 AWS CLI 명령 CreateCapacityReservation참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 속성을 사용하여 새 용량 예약을 생성합니다.

```
Add-EC2CapacityReservation -InstanceType m4.xlarge -InstanceCount 2 -
AvailabilityZone eu-west-1b -EbsOptimized True -InstancePlatform Windows
```

출력:

```
AvailabilityZone       : eu-west-1b
AvailableInstanceCount : 2
CapacityReservationId  : cr-0c1f2345db6f7cdba
CreateDate             : 3/28/2019 9:29:41 AM
EbsOptimized           : True
EndDate                : 1/1/0001 12:00:00 AM
EndDateType            : unlimited
EphemeralStorage       : False
InstanceMatchCriteria  : open
InstancePlatform       : Windows
InstanceType           : m4.xlarge
State                  : active
Tags                   : {}
Tenancy                : default
TotalInstanceCount     : 2
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CreateCapacityReservation.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **`CreateCustomerGateway`** CLI와 함께 사용

다음 코드 예제는 `CreateCustomerGateway`의 사용 방법을 보여줍니다.

CLI

AWS CLI

고객 게이트웨이를 만들려면

이 예에서는 외부 인터페이스에 지정된 IP 주소를 사용하여 고객 게이트웨이를 생성합니다.

명령:

```
aws ec2 create-customer-gateway --type ipsec.1 --public-ip 12.1.2.3 --bgp-asn
 65534
```

출력:

```
{
    "CustomerGateway": {
        "CustomerGatewayId": "cgw-0e11f167",
        "IpAddress": "12.1.2.3",
        "State": "available",
        "Type": "ipsec.1",
        "BgpAsn": "65534"
    }
}
```

- API 세부 정보는 AWS CLI 명령 CreateCustomerGateway참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 고객 게이트웨이를 생성합니다.

```
New-EC2CustomerGateway -Type ipsec.1 -PublicIp 203.0.113.12 -BgpAsn 65534
```

출력:

```
BgpAsn            : 65534
CustomerGatewayId : cgw-1a2b3c4d
IpAddress         : 203.0.113.12
State             : available
Tags              : {}
```

```
Type                : ipsec.1
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CreateCustomerGateway.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateDhcpOptions** CLI와 함께 사용

다음 코드 예제는 CreateDhcpOptions의 사용 방법을 보여줍니다.

CLI

AWS CLI

DHCP 옵션 세트를 만들려면

다음 create-dhcp-options 예에서는 도메인 이름, 도메인 이름 서버 및 NetBIOS 노드 유형을 지정하는 DHCP 옵션 세트를 만듭니다.

```
aws ec2 create-dhcp-options \
    --dhcp-configuration \
        "Key=domain-name-servers,Values=10.2.5.1,10.2.5.2" \
        "Key=domain-name,Values=example.com" \
        "Key=netbios-node-type,Values=2"
```

출력:

```
{
    "DhcpOptions": {
        "DhcpConfigurations": [
            {
                "Key": "domain-name",
                "Values": [
                    {
                        "Value": "example.com"
                    }
                ]
            },
```

```
            {
                "Key": "domain-name-servers",
                "Values": [
                    {
                        "Value": "10.2.5.1"
                    },
                    {
                        "Value": "10.2.5.2"
                    }
                ]
            },
            {
                "Key": "netbios-node-type",
                "Values": [
                    {
                        "Value": "2"
                    }
                ]
            }
        ],
        "DhcpOptionsId": "dopt-06d52773eff4c55f3"
    }
}
```

- API에 대한 자세한 내용은 AWS CLI 명령 참조를 참조하십시오 CreateDhcpOptions.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 DHCP 옵션 세트를 만듭니다. 이 예제에서 사용하는 구문에는 PowerShell 버전 3 이상이 필요합니다.

```
$options = @( @{Key="domain-name";Values=@("abc.local")}, @{Key="domain-name-
servers";Values=@("10.0.0.101","10.0.0.102")})
New-EC2DhcpOption -DhcpConfiguration $options
```

출력:

```
DhcpConfigurations                    DhcpOptionsId    Tags
------------------                    -------------    ----
{domain-name, domain-name-servers}    dopt-1a2b3c4d    {}
```

예 2: PowerShell 버전 2에서는 New-Object를 사용하여 각 DHCP 옵션을 생성해야 합니다.

```
$option1 = New-Object Amazon.EC2.Model.DhcpConfiguration
$option1.Key = "domain-name"
$option1.Values = "abc.local"

$option2 = New-Object Amazon.EC2.Model.DhcpConfiguration
$option2.Key = "domain-name-servers"
$option2.Values = @("10.0.0.101","10.0.0.102")

New-EC2DhcpOption -DhcpConfiguration @($option1, $option2)
```

출력:

```
DhcpConfigurations                    DhcpOptionsId    Tags
------------------                    -------------    ----
{domain-name, domain-name-servers}    dopt-2a3b4c5d    {}
```

- API에 대한 자세한 내용은 Cmdlet 참조를 참조하십시오 CreateDhcpOptions.AWS Tools for PowerShell

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateFlowLogs** CLI와 함께 사용

다음 코드 예제는 CreateFlowLogs의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 흐름 로그를 만들려면

다음 create-flow-logs 예에서는 지정된 네트워크 인터페이스에 대해 거부된 모든 트래픽을 캡처하는 흐름 로그를 만듭니다. 흐름 로그는 지정된 IAM 역할의 권한을 사용하여 Logs의 CloudWatch 로그 그룹에 전달됩니다.

```
aws ec2 create-flow-logs \
    --resource-type NetworkInterface \
```

```
    --resource-ids eni-11223344556677889 \
    --traffic-type REJECT \
    --log-group-name my-flow-logs \
    --deliver-logs-permission-arn arn:aws:iam::123456789101:role/publishFlowLogs
```

출력:

```
{
    "ClientToken": "so0eNA2uSHUNlHI0S2cJ305GuIX1CezaRdGtexample",
    "FlowLogIds": [
        "fl-12345678901234567"
    ],
    "Unsuccessful": []
}
```

자세한 내용은 Amazon VPC 사용 설명서의 VPC 흐름 로그를 참조하세요.

예 2: 사용자 지정 형식의 흐름 로그를 만들려면

다음 `create-flow-logs` 예제는 지정된 VPC의 모든 트래픽을 캡처하고 Amazon S3 버킷으로 흐름 로그를 전송하는 흐름 로그를 생성합니다. `--log-format` 파라미터는 흐름 로그 레코드의 사용자 지정 형식을 지정합니다. Windows에서 이 명령을 실행하려면 작은따옴표 (') 를 큰따옴표 (“) 로 변경하십시오.

```
aws ec2 create-flow-logs \
    --resource-type VPC \
    --resource-ids vpc-00112233344556677 \
    --traffic-type ALL \
    --log-destination-type s3 \
    --log-destination arn:aws:s3:::flow-log-bucket/my-custom-flow-logs/ \
    --log-format '${version} ${vpc-id} ${subnet-id} ${instance-id} ${srcaddr}
 ${dstaddr} ${srcport} ${dstport} ${protocol} ${tcp-flags} ${type} ${pkt-srcaddr}
 ${pkt-dstaddr}'
```

자세한 내용은 Amazon VPC 사용 설명서의 VPC 흐름 로그를 참조하세요.

예 3: 최대 집계 간격이 1분인 흐름 로그를 만들려면

다음 `create-flow-logs` 예제는 지정된 VPC의 모든 트래픽을 캡처하고 Amazon S3 버킷으로 흐름 로그를 전송하는 흐름 로그를 생성합니다. `--max-aggregation-interval`파라미터는 최대 집계 간격을 60초 (1분) 로 지정합니다.

```
aws ec2 create-flow-logs \
    --resource-type VPC \
    --resource-ids vpc-00112233344556677 \
    --traffic-type ALL \
    --log-destination-type s3 \
    --log-destination arn:aws:s3:::flow-log-bucket/my-custom-flow-logs/ \
    --max-aggregation-interval 60
```

자세한 내용은 Amazon VPC 사용 설명서의 VPC 흐름 로그를 참조하세요.

- API 세부 정보는 AWS CLI 명령 CreateFlowLogs참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 '관리자' 역할의 권한을 사용하여 모든 'REJECT' 트래픽에 대해 서브넷 서브넷-1d234567의 EC2 플로우 로그를 cloud-watch-log 지정된 'subnet1-log'에 생성합니다.

```
New-EC2FlowLog -ResourceId "subnet-1d234567" -LogDestinationType cloud-watch-
logs -LogGroupName subnet1-log -TrafficType "REJECT" -ResourceType Subnet -
DeliverLogsPermissionArn "arn:aws:iam::98765432109:role/Admin"
```

출력:

```
ClientToken                                    FlowLogIds            Unsuccessful
-----------                                    ----------            ------------
m1VN2cxP3iB4qo//VUKl5EU6cF7gQLOxcqNefvjeTGw= {fl-012fc34eed5678c9d} {}
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오. CreateFlowLogs

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateImage** CLI와 함께 사용

다음 코드 예제는 CreateImage의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: Amazon EBS 기반 인스턴스에서 AMI를 만들려면

다음 `create-image` 예제는 지정된 인스턴스에서 AMI를 생성합니다.

```
aws ec2 create-image \
    --instance-id i-1234567890abcdef0 \
    --name "My server" \
    --description "An AMI for my server"
```

출력:

```
{
    "ImageId": "ami-abcdef01234567890"
}
```

AMI의 블록 디바이스 매핑을 지정하는 방법에 대한 자세한 내용은 Amazon EC2 사용 설명서의 AMI용 블록 디바이스 매핑 지정을 참조하십시오.

예 2: 재부팅 없이 Amazon EBS 기반 인스턴스에서 AMI를 생성하는 방법

다음 `create-image` 예제에서는 AMI를 생성하고 --no-reboot 파라미터를 설정하여 이미지가 생성되기 전에 인스턴스가 재부팅되지 않도록 합니다.

```
aws ec2 create-image \
    --instance-id i-1234567890abcdef0 \
    --name "My server" \
    --no-reboot
```

출력:

```
{
    "ImageId": "ami-abcdef01234567890"
}
```

AMI의 블록 디바이스 매핑을 지정하는 방법에 대한 자세한 내용은 Amazon EC2 사용 설명서의 AMI용 블록 디바이스 매핑 지정을 참조하십시오.

예 3: 생성 시 AMI 및 스냅샷에 태그 지정하기

다음 `create-image` 예시에서는 AMI를 생성하고 AMI와 스냅샷에 동일한 태그를 지정합니다. `cost-center=cc123`

```
aws ec2 create-image \
    --instance-id i-1234567890abcdef0 \
    --name "My server" \
    --tag-specifications "ResourceType=image,Tags=[{Key=cost-
center,Value=cc123}]" "ResourceType=snapshot,Tags=[{Key=cost-
center,Value=cc123}]"
```

출력:

```
{
    "ImageId": "ami-abcdef01234567890"
}
```

생성 시 리소스에 태그를 지정하는 방법에 대한 자세한 내용은 Amazon EC2 사용 설명서의 리소스 생성 시 태그 추가를 참조하십시오.

- API 세부 정보는 AWS CLI 명령 CreateImage참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 인스턴스에서 지정된 이름과 설명으로 AMI를 생성합니다. Amazon EC2는 이미지를 생성하기 전에 인스턴스를 완전히 종료하려고 시도하고 완료 시 인스턴스를 다시 시작합니다.

```
New-EC2Image -InstanceId i-12345678 -Name "my-web-server" -Description "My web
 server AMI"
```

예 2: 이 예제는 지정된 인스턴스에서 지정된 이름과 설명으로 AMI를 생성합니다. Amazon EC2는 인스턴스를 종료하고 다시 시작하지 않고 이미지를 생성하므로 생성된 이미지의 파일 시스템 무결성을 보장할 수 없습니다.

```
New-EC2Image -InstanceId i-12345678 -Name "my-web-server" -Description "My web
 server AMI" -NoReboot $true
```

예 3: 이 예제는 세 개의 볼륨으로 AMI를 생성합니다. 첫 번째 볼륨은 Amazon EBS 스냅샷을 기반으로 합니다. 두 번째 볼륨은 비어 있는 100GiB 아마존 EBS 볼륨입니다. 세 번째 볼륨은 인스턴스 스토어 볼륨입니다. 이 예제에서 사용하는 구문에는 PowerShell 버전 3 이상이 필요합니다.

```
$ebsBlock1 = @{SnapshotId="snap-1a2b3c4d"}
$ebsBlock2 = @{VolumeSize=100}

New-EC2Image -InstanceId i-12345678 -Name "my-web-server" -Description
 "My web server AMI" -BlockDeviceMapping @( @{DeviceName="/dev/sdf";Ebs=
$ebsBlock1}, @{DeviceName="/dev/sdg";Ebs=$ebsBlock2}, @{DeviceName="/dev/
sdc";VirtualName="ephemeral0"})
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CreateImage.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateInstanceExportTask** CLI와 함께 사용

다음 코드 예제는 CreateInstanceExportTask의 사용 방법을 보여줍니다.

CLI

AWS CLI

인스턴스를 내보내려면

이 예제 명령은 i-1234567890abcdef0 인스턴스를 Amazon S3 버킷 myexportbucket으로 내보내는 작업을 생성합니다.

명령:

```
aws ec2 create-instance-export-task --description "RHEL5 instance" --instance-
id i-1234567890abcdef0 --target-environment vmware --export-to-s3-task
 DiskImageFormat=vmdk,ContainerFormat=ova,S3Bucket=myexportbucket,S3Prefix=RHEL5
```

출력:

```
{
    "ExportTask": {
        "State": "active",
        "InstanceExportDetails": {
            "InstanceId": "i-1234567890abcdef0",
            "TargetEnvironment": "vmware"
        },
        "ExportToS3Task": {
            "S3Bucket": "myexportbucket",
            "S3Key": "RHEL5export-i-fh8sjjsq.ova",
            "DiskImageFormat": "vmdk",
            "ContainerFormat": "ova"
        },
        "Description": "RHEL5 instance",
        "ExportTaskId": "export-i-fh8sjjsq"
    }
}
```

- API AWS CLI 세부 정보는 명령 참조를 참조하십시오. CreateInstanceExportTask

PowerShell

도구: PowerShell

예 1: 이 예제는 중지된 인스턴스를 가상 하드 디스크 (VHD) 로 S3 **testbucket-export-instances-2019** 버킷으로 내보냅니다. **i-0800b00a00EXAMPLE** 대상 환경은 이며**Microsoft**, 사용자의 기본 Region은 us-east-1이 아닌 반면 인스턴스는 지역에 있기 때문에 AWS 지역 매개변수가 추가됩니다. **us-east-1** 내보내기 작업의 상태를 확인하려면 이 명령의 결과에서 **ExportTaskId** 값을 복사한 다음 실행하십시오. **Get-EC2ExportTask -ExportTaskId export_task_ID_from_results.**

```
New-EC2InstanceExportTask -InstanceId i-0800b00a00EXAMPLE -
ExportToS3Task_DiskImageFormat VHD -ExportToS3Task_S3Bucket "testbucket-export-
instances-2019" -TargetEnvironment Microsoft -Region us-east-1
```

출력:

```
Description           :
ExportTaskId          : export-i-077c73108aEXAMPLE
ExportToS3Task        : Amazon.EC2.Model.ExportToS3Task
```

```
InstanceExportDetails : Amazon.EC2.Model.InstanceExportDetails
State                 : active
StatusMessage         :
```

- API에 대한 자세한 내용은 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CreateInstanceExportTask.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateInternetGateway** CLI와 함께 사용

다음 코드 예제는 CreateInternetGateway의 사용 방법을 보여줍니다.

CLI

AWS CLI

인터넷 게이트웨이를 만들려면

다음 create-internet-gateway 예시에서는 태그를 사용하여 인터넷 게이트웨이를 만듭니다Name=my-igw.

```
aws ec2 create-internet-gateway \
    --tag-specifications ResourceType=internet-gateway,Tags=[{Key=Name,Value=my-
igw}]
```

출력:

```
{
    "InternetGateway": {
        "Attachments": [],
        "InternetGatewayId": "igw-0d0fb496b3994d755",
        "OwnerId": "123456789012",
        "Tags": [
            {
                "Key": "Name",
                "Value": "my-igw"
            }
```

```
        ]
    }
}
```

자세한 내용은 Amazon VPC 사용 설명서의 인터넷 게이트웨이를 참조하세요.

- API 세부 정보는 AWS CLI 명령 CreateInternetGateway참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 인터넷 게이트웨이를 생성합니다.

```
New-EC2InternetGateway
```

출력:

```
Attachments    InternetGatewayId    Tags
-----------    -----------------    ----
{}             igw-1a2b3c4d         {}
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CreateInternetGateway.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateKeyPair** CLI와 함께 사용

다음 코드 예제는 CreateKeyPair의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 인스턴스 시작하기

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
/// <summary>
/// Create an Amazon EC2 key pair.
/// </summary>
/// <param name="keyPairName">The name for the new key pair.</param>
/// <returns>The Amazon EC2 key pair created.</returns>
public async Task<KeyPair?> CreateKeyPair(string keyPairName)
{
    var request = new CreateKeyPairRequest
    {
        KeyName = keyPairName,
    };

    var response = await _amazonEC2.CreateKeyPairAsync(request);

    if (response.HttpStatusCode == HttpStatusCode.OK)
    {
        var kp = response.KeyPair;
        return kp;
    }
    else
    {
        Console.WriteLine("Could not create key pair.");
        return null;
    }
}

/// <summary>
/// Save KeyPair information to a temporary file.
/// </summary>
/// <param name="keyPair">The name of the key pair.</param>
/// <returns>The full path to the temporary file.</returns>
public string SaveKeyPair(KeyPair keyPair)
```

```
    {
        var tempPath = Path.GetTempPath();
        var tempFileName = $"{tempPath}\\{Path.GetRandomFileName()}";
        var pemFileName = Path.ChangeExtension(tempFileName, "pem");

        // Save the key pair to a file in a temporary folder.
        using var stream = new FileStream(pemFileName, FileMode.Create);
        using var writer = new StreamWriter(stream);
        writer.WriteLine(keyPair.KeyMaterial);

        return pemFileName;
    }
```

- API 세부 정보는 AWS SDK for .NET API CreateKeyPair참조를 참조하십시오.

Bash

AWS CLI Bash 스크립트 사용

> **Note**
>
> 더 많은 정보가 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
###############################################################################
# function ec2_create_keypair
#
# This function creates an Amazon Elastic Compute Cloud (Amazon EC2) ED25519 or
 2048-bit RSA key pair
# and writes it to a file.
#
# Parameters:
#       -n key_pair_name - A key pair name.
#       -f file_path - File to store the key pair.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_create_keypair() {
```

```
  local key_pair_name file_path response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_create_keypair"
    echo "Creates an Amazon Elastic Compute Cloud (Amazon EC2) ED25519 or 2048-
bit RSA key pair"
    echo " and writes it to a file."
    echo "  -n key_pair_name - A key pair name."
    echo "  -f file_path - File to store the key pair."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:f:h" option; do
    case "${option}" in
      n) key_pair_name="${OPTARG}" ;;
      f) file_path="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$key_pair_name" ]]; then
    errecho "ERROR: You must provide a key name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$file_path" ]]; then
    errecho "ERROR: You must provide a file path with the -f parameter."
    usage
    return 1
  fi
```

```
  response=$(aws ec2 create-key-pair \
    --key-name "$key_pair_name" \
    --query 'KeyMaterial' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  }

  if [[ -n "$file_path" ]]; then
    echo "$response" >"$file_path"
  fi

  return 0
}
```

이 예제에 사용된 유틸리티 함수

```
#############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
#############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
```

```
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- API 세부 정보는 AWS CLI 명령 CreateKeyPair참조를 참조하십시오.

C++

SDK for C++

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::CreateKeyPairRequest request;
    request.SetKeyName(keyPairName);

    Aws::EC2::Model::CreateKeyPairOutcome outcome =
 ec2Client.CreateKeyPair(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Failed to create key pair:" <<
                  outcome.GetError().GetMessage() << std::endl;
    }
```

```
    else {
        std::cout << "Successfully created key pair named " <<
                  keyPairName << std::endl;
    }
```

- API 세부 정보는 AWS SDK for C++ API CreateKeyPair참조를 참조하십시오.

CLI

AWS CLI

키 페어를 생성하는 방법

이 예제에서는 이름이 MyKeyPair인 키 페어를 생성합니다.

명령:

```
aws ec2 create-key-pair --key-name MyKeyPair
```

출력은 프라이빗 키 및 키 지문의 ASCII 버전입니다. 키는 파일에 저장해야 합니다.

자세한 내용은 AWS Command Line Interface 사용 설명서의 키 페어 사용을 참조하세요.

- API 세부 정보는 AWS CLI 명령 CreateKeyPair참조를 참조하십시오.

Java

SDK for Java 2.x

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    public static void createKeyPair(Ec2Client ec2, String keyName, String
fileName) {
        try {
            CreateKeyPairRequest request = CreateKeyPairRequest.builder()
```

```
                    .keyName(keyName)
                    .build();

            CreateKeyPairResponse response = ec2.createKeyPair(request);
            String content = response.keyMaterial();
            BufferedWriter writer = new BufferedWriter(new FileWriter(fileName));
            writer.write(content);
            writer.close();
            System.out.println("Successfully created key pair named " + keyName);

        } catch (Ec2Exception | IOException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API CreateKeyPair참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { CreateKeyPairCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  try {
    // Create a key pair in Amazon EC2.
    const { KeyMaterial, KeyName } = await client.send(
      // A unique name for the key pair. Up to 255 ASCII characters.
      new CreateKeyPairCommand({ KeyName: "KEY_PAIR_NAME" }),
    );
    // This logs your private key. Be sure to save it.
    console.log(KeyName);
```

```
    console.log(KeyMaterial);
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API CreateKeyPair참조를 참조하십시오.

Kotlin

SDK for Kotlin

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
suspend fun createEC2KeyPair(keyNameVal: String) {
    val request =
        CreateKeyPairRequest {
            keyName = keyNameVal
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.createKeyPair(request)
        println("The key ID is ${response.keyPairId}")
    }
}
```

- API 세부 정보는 Kotlin APIAWS SDK 레퍼런스를 참조하세요 CreateKeyPair.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예제에서는 키 페어를 생성하고 지정된 이름의 파일에 PEM으로 인코딩된 RSA 개인 키를 캡처합니다. 를 사용하는 경우 유효한 키를 PowerShell 생성하려면 인코딩을 ASCII

로 설정해야 합니다. 자세한 내용은 AWS 명령줄 인터페이스 사용 설명서의 Amazon EC2 키 페어 생성, 표시 및 삭제 (https://docs.aws.amazon.com/cli/latest/userguide/ cli-services-ec 2-keypairs.html) 를 참조하십시오.

```
(New-EC2KeyPair -KeyName "my-key-pair").KeyMaterial | Out-File -Encoding ascii -
FilePath C:\path\my-key-pair.pem
```

- API에 대한 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CreateKeyPair.

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class KeyPairWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) key pair
 actions."""

    def __init__(self, ec2_resource, key_file_dir, key_pair=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param key_file_dir: The folder where the private key information is
 stored.
                             This should be a secure folder.
        :param key_pair: A Boto3 KeyPair object. This is a high-level object that
                         wraps key pair actions.
        """
        self.ec2_resource = ec2_resource
        self.key_pair = key_pair
        self.key_file_path = None
        self.key_file_dir = key_file_dir
```

```
    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource, tempfile.TemporaryDirectory())


    def create(self, key_name):
        """
        Creates a key pair that can be used to securely connect to an EC2
instance.
        The returned key pair contains private key information that cannot be
retrieved
        again. The private key data is stored as a .pem file.

        :param key_name: The name of the key pair to create.
        :return: A Boto3 KeyPair object that represents the newly created key
pair.
        """
        try:
            self.key_pair = self.ec2_resource.create_key_pair(KeyName=key_name)
            self.key_file_path = os.path.join(
                self.key_file_dir.name, f"{self.key_pair.name}.pem"
            )
            with open(self.key_file_path, "w") as key_file:
                key_file.write(self.key_pair.key_material)
        except ClientError as err:
            logger.error(
                "Couldn't create key %s. Here's why: %s: %s",
                key_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return self.key_pair
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 CreateKeyPair.

Ruby

SDK for Ruby

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
# This code example does the following:
# 1. Creates a key pair in Amazon Elastic Compute Cloud (Amazon EC2).
# 2. Displays information about available key pairs.
# 3. Deletes the key pair.

require "aws-sdk-ec2"

# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @param key_pair_name [String] The name for the key pair and private
#   key file.
# @return [Boolean] true if the key pair and private key file were
#   created; otherwise, false.
# @example
#   exit 1 unless key_pair_created?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'my-key-pair'
#   )
def key_pair_created?(ec2_client, key_pair_name)
  key_pair = ec2_client.create_key_pair(key_name: key_pair_name)
  puts "Created key pair '#{key_pair.key_name}' with fingerprint " \
    "'#{key_pair.key_fingerprint}' and ID '#{key_pair.key_pair_id}'."
  filename = File.join(Dir.home, key_pair_name + ".pem")
  File.open(filename, "w") { |file| file.write(key_pair.key_material) }
  puts "Private key file saved locally as '#{filename}'."
  return true
rescue Aws::EC2::Errors::InvalidKeyPairDuplicate
  puts "Error creating key pair: a key pair named '#{key_pair_name}' " \
    "already exists."
  return false
rescue StandardError => e
  puts "Error creating key pair or saving private key file: #{e.message}"
```

```
  return false
end

# Displays information about available key pairs in
# Amazon Elastic Compute Cloud (Amazon EC2).
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @example
#   describe_key_pairs(Aws::EC2::Client.new(region: 'us-west-2'))
def describe_key_pairs(ec2_client)
  result = ec2_client.describe_key_pairs
  if result.key_pairs.count.zero?
    puts "No key pairs found."
  else
    puts "Key pair names:"
    result.key_pairs.each do |key_pair|
      puts key_pair.key_name
    end
  end
rescue StandardError => e
  puts "Error getting information about key pairs: #{e.message}"
end

# Deletes a key pair in Amazon Elastic Compute Cloud (Amazon EC2).
#
# Prerequisites:
#
# - The key pair to delete.
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @param key_pair_name [String] The name of the key pair to delete.
# @return [Boolean] true if the key pair was deleted; otherwise, false.
# @example
#   exit 1 unless key_pair_deleted?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'my-key-pair'
#   )
def key_pair_deleted?(ec2_client, key_pair_name)
  ec2_client.delete_key_pair(key_name: key_pair_name)
  return true
rescue StandardError => e
  puts "Error deleting key pair: #{e.message}"
  return false
end
```

```
# Example usage:
def run_me
  key_pair_name = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:   ruby ec2-ruby-example-key-pairs.rb KEY_PAIR_NAME REGION"
    puts "Example: ruby ec2-ruby-example-key-pairs.rb my-key-pair us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  elsif ARGV.count.zero?
    key_pair_name = "my-key-pair"
    region = "us-west-2"
  # Otherwise, use the values as specified at the command prompt.
  else
    key_pair_name = ARGV[0]
    region = ARGV[1]
  end

  ec2_client = Aws::EC2::Client.new(region: region)

  puts "Displaying existing key pair names before creating this key pair..."
  describe_key_pairs(ec2_client)

  puts "-" * 10
  puts "Creating key pair..."
  unless key_pair_created?(ec2_client, key_pair_name)
    puts "Stopping program."
    exit 1
  end

  puts "-" * 10
  puts "Displaying existing key pair names after creating this key pair..."
  describe_key_pairs(ec2_client)

  puts "-" * 10
  puts "Deleting key pair..."
  unless key_pair_deleted?(ec2_client, key_pair_name)
    puts "Stopping program. You must delete the key pair yourself."
    exit 1
  end
  puts "Key pair deleted."
```

```
  puts "-" * 10
  puts "Now that the key pair is deleted, " \
    "also deleting the related private key pair file..."
  filename = File.join(Dir.home, key_pair_name + ".pem")
  File.delete(filename)
  if File.exist?(filename)
    puts "Could not delete file at '#{filename}'. You must delete it yourself."
  else
    puts "File deleted."
  end

  puts "-" * 10
  puts "Displaying existing key pair names after deleting this key pair..."
  describe_key_pairs(ec2_client)
end

run_me if $PROGRAM_NAME == __FILE__
```

- API 세부 정보는 AWS SDK for Ruby API CreateKeyPair참조를 참조하십시오.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    TRY.
        oo_result = lo_ec2->createkeypair( iv_keyname = iv_key_name ).
                 " oo_result is returned for testing purposes. "
        MESSAGE 'Amazon EC2 key pair created.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- API 세부 정보는 SAP용AWS SDK ABAP API 참조를 참조하십시오 CreateKeyPair.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateLaunchTemplate** CLI와 함께 사용

다음 코드 예제는 CreateLaunchTemplate의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 복원력이 뛰어난 서비스 구축 및 관리

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
   /// <summary>
   /// Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling.
   /// The launch template specifies a Bash script in its user data field that
runs after
   /// the instance is started. This script installs the Python packages and
starts a Python
   /// web server on the instance.
   /// </summary>
   /// <param name="startupScriptPath">The path to a Bash script file that is
run.</param>
   /// <param name="instancePolicyPath">The path to a permissions policy to
create and attach to the profile.</param>
```

```
    /// <returns>The template object.</returns>
    public async Task<Amazon.EC2.Model.LaunchTemplate> CreateTemplate(string
startupScriptPath, string instancePolicyPath)
    {
        await CreateKeyPair(_keyPairName);
        await CreateInstanceProfileWithName(_instancePolicyName,
_instanceRoleName, _instanceProfileName, instancePolicyPath);

        var startServerText = await File.ReadAllTextAsync(startupScriptPath);
        var plainTextBytes = System.Text.Encoding.UTF8.GetBytes(startServerText);

        var amiLatest = await _amazonSsm.GetParameterAsync(
            new GetParameterRequest() { Name = _amiParam });
        var amiId = amiLatest.Parameter.Value;
        var launchTemplateResponse = await _amazonEc2.CreateLaunchTemplateAsync(
            new CreateLaunchTemplateRequest()
            {
                LaunchTemplateName = _launchTemplateName,
                LaunchTemplateData = new RequestLaunchTemplateData()
                {
                    InstanceType = _instanceType,
                    ImageId = amiId,
                    IamInstanceProfile =
                        new

LaunchTemplateIamInstanceProfileSpecificationRequest()
                        {
                            Name = _instanceProfileName
                        },
                    KeyName = _keyPairName,
                    UserData = System.Convert.ToBase64String(plainTextBytes)
                }
            });
        return launchTemplateResponse.LaunchTemplate;

    }
```

- API 세부 정보는 AWS SDK for .NET API CreateLaunchTemplate참조를 참조하십시오.

CLI

AWS CLI

예 1: 시작 템플릿을 생성하는 방법

다음 `create-launch-template` 예제에서는 인스턴스를 시작하고 인스턴스에 퍼블릭 IP 주소 및 IPv6 주소를 할당하며 인스턴스에 대한 태그를 생성할 서브넷을 지정하는 시작 템플릿을 생성합니다.

```
aws ec2 create-launch-template \
    --launch-template-name TemplateForWebServer \
    --version-description WebVersion1 \
    --launch-template-data '{"NetworkInterfaces":
[{"AssociatePublicIpAddress":true,"DeviceIndex":0,"Ipv6AddressCount":1,"SubnetId":"subnet
[{"ResourceType":"instance","Tags":[{"Key":"purpose","Value":"webserver"}]}]}'
```

출력:

```
{
    "LaunchTemplate": {
        "LatestVersionNumber": 1,
        "LaunchTemplateId": "lt-01238c059e3466abc",
        "LaunchTemplateName": "TemplateForWebServer",
        "DefaultVersionNumber": 1,
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "CreateTime": "2019-01-27T09:13:24.000Z"
    }
}
```

자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 시작 템플릿에서 인스턴스 시작을 참조하세요. JSON 형식 파라미터에서 따옴표 사용에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서의 문자열에 따옴표 사용을 참조하세요.

예제 2: Amazon EC2 Auto Scaling에 대한 시작 템플릿을 생성하는 방법

다음 `create-launch-template` 예제에서는 인스턴스를 시작할 때 추가 EBS 볼륨을 지정하도록 여러 태그 및 블록 디바이스 매핑을 사용하는 시작 템플릿을 생성합니다. Auto Scaling이 인스턴스를 시작하는 VPC의 보안 그룹에 해당하는 `Groups`에 대한 값을 지정합니다. Auto Scaling의 속성으로 VPC 및 서브넷을 지정합니다.

```
aws ec2 create-launch-template \
```

```
    --launch-template-name TemplateForAutoScaling \
    --version-description AutoScalingVersion1 \
    --launch-template-data '{"NetworkInterfaces":
[{"DeviceIndex":0,"AssociatePublicIpAddress":true,"Groups":
["sg-7c227019,sg-903004f8"],"DeleteOnTermination":true}],"ImageId":"ami-
b42209de","InstanceType":"m4.large","TagSpecifications":
[{"ResourceType":"instance","Tags":[{"Key":"environment","Value":"production"},
{"Key":"purpose","Value":"webserver"}]},{"ResourceType":"volume","Tags":
[{"Key":"environment","Value":"production"},{"Key":"cost-
center","Value":"cc123"}]}],"BlockDeviceMappings":[{"DeviceName":"/dev/
sda1","Ebs":{"VolumeSize":100}}]}' --region us-east-1
```

출력:

```
{
    "LaunchTemplate": {
        "LatestVersionNumber": 1,
        "LaunchTemplateId": "lt-0123c79c33a54e0abc",
        "LaunchTemplateName": "TemplateForAutoScaling",
        "DefaultVersionNumber": 1,
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "CreateTime": "2019-04-30T18:16:06.000Z"
    }
}
```

자세한 내용은 Amazon EC2 Auto Scaling 사용 설명서의 Auto Scaling 그룹에 대한 시작 템플릿 생성을 참조하세요. JSON 형식 파라미터에서 따옴표 사용에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서의 문자열에 따옴표 사용을 참조하세요.

예제 3: EBS 볼륨의 암호화를 지정하는 시작 템플릿을 생성하는 방법

다음 `create-launch-template` 예제에서는 암호화되지 않은 스냅샷에서 생성된 암호화된 EBS 볼륨을 포함하는 시작 템플릿을 생성합니다. 또한 생성 중에 볼륨에 태그도 지정합니다. 기본적으로 암호화가 비활성화된 경우 다음 예제에 표시된 대로 `"Encrypted"` 옵션을 지정해야 합니다. `"KmsKeyId"` 옵션을 사용하여 고객 관리형 CMK를 지정하는 경우 기본적으로 암호화가 활성화되어 있더라도 `"Encrypted"` 옵션도 지정해야 합니다.

```
aws ec2 create-launch-template \
  --launch-template-name TemplateForEncryption \
  --launch-template-data file://config.json
```

`config.json`의 콘텐츠:

```
{
    "BlockDeviceMappings":[
        {
            "DeviceName":"/dev/sda1",
            "Ebs":{
                "VolumeType":"gp2",
                "DeleteOnTermination":true,
                "SnapshotId":"snap-066877671789bd71b",
                "Encrypted":true,
                "KmsKeyId":"arn:aws:kms:us-east-1:012345678910:key/abcd1234-
a123-456a-a12b-a123b4cd56ef"
            }
        }
    ],
    "ImageId":"ami-00068cd7555f543d5",
    "InstanceType":"c5.large",
    "TagSpecifications":[
        {
            "ResourceType":"volume",
            "Tags":[
                {
                    "Key":"encrypted",
                    "Value":"yes"
                }
            ]
        }
    ]
}
```

출력:

```
{
    "LaunchTemplate": {
        "LatestVersionNumber": 1,
        "LaunchTemplateId": "lt-0d5bd51bcf8530abc",
        "LaunchTemplateName": "TemplateForEncryption",
        "DefaultVersionNumber": 1,
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "CreateTime": "2020-01-07T19:08:36.000Z"
    }
}
```

자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 스냅샷에서 Amazon EBS 볼륨 복원 및 암호화 기본 제공을 참조하세요.

- API 세부 정보는 AWS CLI 명령 CreateLaunchTemplate참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
const ssmClient = new SSMClient({});
const { Parameter } = await ssmClient.send(
  new GetParameterCommand({
    Name: "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
  }),
);
const ec2Client = new EC2Client({});
await ec2Client.send(
  new CreateLaunchTemplateCommand({
    LaunchTemplateName: NAMES.launchTemplateName,
    LaunchTemplateData: {
      InstanceType: "t3.micro",
      ImageId: Parameter.Value,
      IamInstanceProfile: { Name: NAMES.instanceProfileName },
      UserData: readFileSync(
        join(RESOURCES_PATH, "server_startup_script.sh"),
      ).toString("base64"),
      KeyName: NAMES.keyPairName,
    },
  }),
```

- API 세부 정보는 AWS SDK for JavaScript API CreateLaunchTemplate참조를 참조하십시오.

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

이 예제에서는 인스턴스에 특정 권한을 부여하는 인스턴스 프로파일과 인스턴스가 시작된 후 인스턴스에서 실행되는 사용자 데이터 Bash 스크립트가 포함된 시작 템플릿을 생성합니다.

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
```

```
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def create_template(self, server_startup_script_file, instance_policy_file):
        """
        Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling. The
        launch template specifies a Bash script in its user data field that runs
after
        the instance is started. This script installs Python packages and starts
a
        Python web server on the instance.

        :param server_startup_script_file: The path to a Bash script file that is
run
                                           when an instance starts.
        :param instance_policy_file: The path to a file that defines a
permissions policy
                                     to create and attach to the instance
profile.
        :return: Information about the newly created template.
        """
        template = {}
        try:
            self.create_key_pair(self.key_pair_name)
            self.create_instance_profile(
                instance_policy_file,
                self.instance_policy_name,
                self.instance_role_name,
                self.instance_profile_name,
            )
            with open(server_startup_script_file) as file:
```

```
                start_server_script = file.read()
            ami_latest = self.ssm_client.get_parameter(Name=self.ami_param)
            ami_id = ami_latest["Parameter"]["Value"]
            lt_response = self.ec2_client.create_launch_template(
                LaunchTemplateName=self.launch_template_name,
                LaunchTemplateData={
                    "InstanceType": self.inst_type,
                    "ImageId": ami_id,
                    "IamInstanceProfile": {"Name": self.instance_profile_name},
                    "UserData": base64.b64encode(
                        start_server_script.encode(encoding="utf-8")
                    ).decode(encoding="utf-8"),
                    "KeyName": self.key_pair_name,
                },
            )
            template = lt_response["LaunchTemplate"]
            log.info(
                "Created launch template %s for AMI %s on %s.",
                self.launch_template_name,
                ami_id,
                self.inst_type,
            )
        except ClientError as err:
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.AlreadyExistsException"
            ):
                log.info(
                    "Launch template %s already exists, nothing to do.",
                    self.launch_template_name,
                )
            else:
                raise AutoScalerError(
                    f"Couldn't create launch template
{self.launch_template_name}: {err}."
                )
        return template
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 CreateLaunchTemplate.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateNetworkAcl** CLI와 함께 사용

다음 코드 예제는 CreateNetworkAcl의 사용 방법을 보여줍니다.

CLI

AWS CLI

네트워크 ACL을 만들려면

이 예제에서는 지정된 VPC에 대한 네트워크 ACL을 생성합니다.

명령:

```
aws ec2 create-network-acl --vpc-id vpc-a01106c2
```

출력:

```
{
    "NetworkAcl": {
        "Associations": [],
        "NetworkAclId": "acl-5fb85d36",
        "VpcId": "vpc-a01106c2",
        "Tags": [],
        "Entries": [
            {
                "CidrBlock": "0.0.0.0/0",
                "RuleNumber": 32767,
                "Protocol": "-1",
                "Egress": true,
                "RuleAction": "deny"
            },
            {
                "CidrBlock": "0.0.0.0/0",
                "RuleNumber": 32767,
                "Protocol": "-1",
                "Egress": false,
                "RuleAction": "deny"
            }
```

```
        ],
        "IsDefault": false
    }
}
```

- API 세부 정보는 AWS CLI 명령 CreateNetworkAcl참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 VPC에 대한 네트워크 ACL을 생성합니다.

```
New-EC2NetworkAcl -VpcId vpc-12345678
```

출력:

```
Associations : {}
Entries      : {Amazon.EC2.Model.NetworkAclEntry,
 Amazon.EC2.Model.NetworkAclEntry}
IsDefault    : False
NetworkAclId : acl-12345678
Tags         : {}
VpcId        : vpc-12345678
```

- API에 대한 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CreateNetworkAcl.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateNetworkAclEntry** CLI와 함께 사용

다음 코드 예제는 CreateNetworkAclEntry의 사용 방법을 보여줍니다.

CLI

AWS CLI

네트워크 ACL 항목을 만들려면

이 예제에서는 지정된 네트워크 ACL에 대한 항목을 생성합니다. 이 규칙은 UDP 포트 53 (DNS) 의 모든 IPv4 주소 (0.0.0.0/0) 에서 연결된 서브넷으로의 인그레스 트래픽을 허용합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 create-network-acl-entry --network-acl-id acl-5fb85d36 --ingress --rule-
number 100 --protocol udp --port-range From=53,To=53 --cidr-block 0.0.0.0/0 --
rule-action allow
```

이 예제에서는 지정된 네트워크 ACL에 대해 TCP 포트 80 (HTTP) 의 모든 IPv6 주소 (:: /0) 에서 들어오는 인그레스 트래픽을 허용하는 규칙을 생성합니다.

명령:

```
aws ec2 create-network-acl-entry --network-acl-id acl-5fb85d36 --ingress --rule-
number 120 --protocol tcp --port-range From=80,To=80 --ipv6-cidr-block ::/0 --
rule-action allow
```

- API 세부 정보는 명령 참조를 참조하십시오. CreateNetworkAclEntryAWS CLI

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 네트워크 ACL에 대한 항목을 생성합니다. 이 규칙은 UDP 포트 53 (DNS) 의 모든 위치 (0.0.0.0/0) 에서 연결된 서브넷으로 들어오는 인바운드 트래픽을 허용합니다.

```
New-EC2NetworkAclEntry -NetworkAclId acl-12345678 -Egress $false -RuleNumber
 100 -Protocol 17 -PortRange_From 53 -PortRange_To 53 -CidrBlock 0.0.0.0/0 -
RuleAction allow
```

- API에 대한 자세한 내용은 Cmdlet 참조를 참조하십시오. CreateNetworkAclEntryAWS Tools for PowerShell

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateNetworkInterface** CLI와 함께 사용

다음 코드 예제는 CreateNetworkInterface의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 네트워크 인터페이스의 IPv4 주소 지정하기

다음 create-network-interface 예제에서는 지정된 기본 IPv4 주소를 사용하여 지정된 서브넷의 네트워크 인터페이스를 만듭니다.

```
aws ec2 create-network-interface \
    --subnet-id subnet-00a24d0d67acf6333 \
    --description "my network interface" \
    --groups sg-09dfba7ed20cda78b \
    --private-ip-address 10.0.8.17
```

출력:

```
{
    "NetworkInterface": {
        "AvailabilityZone": "us-west-2a",
        "Description": "my network interface",
        "Groups": [
            {
                "GroupName": "my-security-group",
                "GroupId": "sg-09dfba7ed20cda78b"
            }
        ],
        "InterfaceType": "interface",
        "Ipv6Addresses": [],
        "MacAddress": "06:6a:0f:9a:49:37",
        "NetworkInterfaceId": "eni-0492b355f0cf3b3f8",
        "OwnerId": "123456789012",
        "PrivateDnsName": "ip-10-0-8-18.us-west-2.compute.internal",
        "PrivateIpAddress": "10.0.8.17",
        "PrivateIpAddresses": [
            {
                "Primary": true,
                "PrivateDnsName": "ip-10-0-8-17.us-west-2.compute.internal",
```

```
                "PrivateIpAddress": "10.0.8.17"
            }
        ],
        "RequesterId": "AIDA4Z3Y7GSXTMEXAMPLE",
        "RequesterManaged": false,
        "SourceDestCheck": true,
        "Status": "pending",
        "SubnetId": "subnet-00a24d0d67acf6333",
        "TagSet": [],
        "VpcId": "vpc-02723a0feeeb9d57b"
    }
}
```

예 2: IPv4 주소와 IPv6 주소를 사용하여 네트워크 인터페이스를 만들려면

다음 `create-network-interface` 예제는 Amazon EC2에서 선택한 IPv4 주소와 IPv6 주소를 사용하여 지정된 서브넷의 네트워크 인터페이스를 생성합니다.

```
aws ec2 create-network-interface \
    --subnet-id subnet-00a24d0d67acf6333 \
    --description "my dual stack network interface" \
    --ipv6-address-count 1 \
    --groups sg-09dfba7ed20cda78b
```

출력:

```
{
    "NetworkInterface": {
        "AvailabilityZone": "us-west-2a",
        "Description": "my dual stack network interface",
        "Groups": [
            {
                "GroupName": "my-security-group",
                "GroupId": "sg-09dfba7ed20cda78b"
            }
        ],
        "InterfaceType": "interface",
        "Ipv6Addresses": [
            {
                "Ipv6Address": "2600:1f13:cfe:3650:a1dc:237c:393a:4ba7",
                "IsPrimaryIpv6": false
            }
```

```
        ],
        "MacAddress": "06:b8:68:d2:b2:2d",
        "NetworkInterfaceId": "eni-05da417453f9a84bf",
        "OwnerId": "123456789012",
        "PrivateDnsName": "ip-10-0-8-18.us-west-2.compute.internal",
        "PrivateIpAddress": "10.0.8.18",
        "PrivateIpAddresses": [
            {
                "Primary": true,
                "PrivateDnsName": "ip-10-0-8-18.us-west-2.compute.internal",
                "PrivateIpAddress": "10.0.8.18"
            }
        ],
        "RequesterId": "AIDA4Z3Y7GSXTMEXAMPLE",
        "RequesterManaged": false,
        "SourceDestCheck": true,
        "Status": "pending",
        "SubnetId": "subnet-00a24d0d67acf6333",
        "TagSet": [],
        "VpcId": "vpc-02723a0feeeb9d57b",
        "Ipv6Address": "2600:1f13:cfe:3650:a1dc:237c:393a:4ba7"
    }
}
```

예 3: 연결 추적 구성 옵션을 사용하여 네트워크 인터페이스를 만들려면

다음 `create-network-interface` 예제에서는 네트워크 인터페이스를 만들고 유휴 연결 추적 제한 시간을 구성합니다.

```
aws ec2 create-network-interface \
    --subnet-id subnet-00a24d0d67acf6333 \
    --groups sg-02e57dbcfe0331c1b \
    --connection-tracking-specification TcpEstablishedTimeout=86400,UdpTimeout=60
```

출력:

```
{
    "NetworkInterface": {
        "AvailabilityZone": "us-west-2a",
        "ConnectionTrackingConfiguration": {
            "TcpEstablishedTimeout": 86400,
            "UdpTimeout": 60
```

```
        },
        "Description": "",
        "Groups": [
            {
                "GroupName": "my-security-group",
                "GroupId": "sg-02e57dbcfe0331c1b"
            }
        ],
        "InterfaceType": "interface",
        "Ipv6Addresses": [],
        "MacAddress": "06:4c:53:de:6d:91",
        "NetworkInterfaceId": "eni-0c133586e08903d0b",
        "OwnerId": "123456789012",
        "PrivateDnsName": "ip-10-0-8-94.us-west-2.compute.internal",
        "PrivateIpAddress": "10.0.8.94",
        "PrivateIpAddresses": [
            {
                "Primary": true,
                "PrivateDnsName": "ip-10-0-8-94.us-west-2.compute.internal",
                "PrivateIpAddress": "10.0.8.94"
            }
        ],
        "RequesterId": "AIDA4Z3Y7GSXTMEXAMPLE",
        "RequesterManaged": false,
        "SourceDestCheck": true,
        "Status": "pending",
        "SubnetId": "subnet-00a24d0d67acf6333",
        "TagSet": [],
        "VpcId": "vpc-02723a0feeeb9d57b"
    }
}
```

예 4: 엘라스틱 패브릭 어댑터를 만들려면

다음 `create-network-interface` 예제는 EFA를 생성합니다.

```
aws ec2 create-network-interface \
    --interface-type efa \
    --subnet-id subnet-00a24d0d67acf6333 \
    --description "my efa" \
    --groups sg-02e57dbcfe0331c1b
```

출력:

```
{
    "NetworkInterface": {
        "AvailabilityZone": "us-west-2a",
        "Description": "my efa",
        "Groups": [
            {
                "GroupName": "my-efa-sg",
                "GroupId": "sg-02e57dbcfe0331c1b"
            }
        ],
        "InterfaceType": "efa",
        "Ipv6Addresses": [],
        "MacAddress": "06:d7:a4:f7:4d:57",
        "NetworkInterfaceId": "eni-034acc2885e862b65",
        "OwnerId": "123456789012",
        "PrivateDnsName": "ip-10-0-8-180.us-west-2.compute.internal",
        "PrivateIpAddress": "10.0.8.180",
        "PrivateIpAddresses": [
            {
                "Primary": true,
                "PrivateDnsName": "ip-10-0-8-180.us-west-2.compute.internal",
                "PrivateIpAddress": "10.0.8.180"
            }
        ],
        "RequesterId": "AIDA4Z3Y7GSXTMEXAMPLE",
        "RequesterManaged": false,
        "SourceDestCheck": true,
        "Status": "pending",
        "SubnetId": "subnet-00a24d0d67acf6333",
        "TagSet": [],
        "VpcId": "vpc-02723a0feeeb9d57b"
    }
}
```

자세한 내용은 Amazon EC2 사용 설명서의 탄력적 네트워크 인터페이스를 참조하세요.

- API 세부 정보는 AWS CLI 명령 CreateNetworkInterface참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 네트워크 인터페이스를 생성합니다.

```
New-EC2NetworkInterface -SubnetId subnet-1a2b3c4d -Description "my network
 interface" -Group sg-12345678 -PrivateIpAddress 10.0.0.17
```

출력:

```
Association        :
Attachment         :
AvailabilityZone   : us-west-2c
Description        : my network interface
Groups             : {my-security-group}
MacAddress         : 0a:72:bc:1a:cd:7f
NetworkInterfaceId : eni-12345678
OwnerId            : 123456789012
PrivateDnsName     : ip-10-0-0-17.us-west-2.compute.internal
PrivateIpAddress   : 10.0.0.17
PrivateIpAddresses : {}
RequesterId        :
RequesterManaged   : False
SourceDestCheck    : True
Status             : pending
SubnetId           : subnet-1a2b3c4d
TagSet             : {}
VpcId              : vpc-12345678
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CreateNetworkInterface.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreatePlacementGroup** CLI와 함께 사용

다음 코드 예제는 CreatePlacementGroup의 사용 방법을 보여줍니다.

CLI

AWS CLI

플레이스먼트 그룹을 만들려면

이 예제 명령은 지정된 이름을 가진 배치 그룹을 만듭니다.

명령:

```
aws ec2 create-placement-group --group-name my-cluster --strategy cluster
```

파티션 배치 그룹을 만들려면

이 예제 명령은 다섯 개의 HDFS-Group-A 파티션으로 이름이 지정된 파티션 배치 그룹을 만듭니다.

명령:

```
aws ec2 create-placement-group --group-name HDFS-Group-A --strategy partition --
partition-count 5
```

- API 세부 정보는 AWS CLI 명령 CreatePlacementGroup참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예시에서는 지정된 이름의 배치 그룹을 생성합니다.

```
New-EC2PlacementGroup -GroupName my-placement-group -Strategy cluster
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CreatePlacementGroup.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateRoute** CLI와 함께 사용

다음 코드 예제는 CreateRoute의 사용 방법을 보여줍니다.

CLI

AWS CLI

라우트를 만들려면

이 예제에서는 지정된 라우팅 테이블에 대한 경로를 생성합니다. 경로는 모든 IPv4 트래픽(0.0.0.0/0) 과 일치하여 지정된 인터넷 게이트웨이로 라우팅합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 create-route --route-table-id rtb-22574640 --destination-cidr-block
 0.0.0.0/0 --gateway-id igw-c0a643a9
```

이 예제 명령은 라우팅 테이블 rtb-g8ff4ea2에 경로를 생성합니다. 이 경로는 IPv4 CIDR 블록 10.0.0.0/16의 트래픽을 매칭하여 VPC 피어링 연결인 pcx-111aaa22로 라우팅합니다. 이 경로를 사용하면 트래픽이 VPC 피어링 연결의 피어 VPC로 전달될 수 있습니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 create-route --route-table-id rtb-g8ff4ea2 --destination-cidr-block
 10.0.0.0/16 --vpc-peering-connection-id pcx-1a2b3c4d
```

이 예제에서는 모든 IPv6 트래픽 (::/0) 과 일치하는 경로를 지정된 라우팅 테이블에 생성하여 지정된 외부 전용 인터넷 게이트웨이로 라우팅합니다.

명령:

```
aws ec2 create-route --route-table-id rtb-dce620b8 --destination-ipv6-cidr-
block ::/0 --egress-only-internet-gateway-id eigw-01eadbd45ecd7943f
```

- API 세부 정보는 명령 참조를 참조하십시오. CreateRouteAWS CLI

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 라우팅 테이블에 대해 지정된 경로를 생성합니다. 경로는 모든 트래픽을 매칭하여 지정된 인터넷 게이트웨이로 전송합니다.

```
New-EC2Route -RouteTableId rtb-1a2b3c4d -DestinationCidrBlock 0.0.0.0/0 -
GatewayId igw-1a2b3c4d
```

출력:

```
True
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CreateRoute.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateRouteTable** CLI와 함께 사용

다음 코드 예제는 CreateRouteTable의 사용 방법을 보여줍니다.

CLI

AWS CLI

라우팅 테이블을 생성하는 방법

이 예제에서는 지정된 VPC에 대한 라우팅 테이블을 생성합니다.

명령:

```
aws ec2 create-route-table --vpc-id vpc-a01106c2
```

출력:

```
{
    "RouteTable": {
        "Associations": [],
        "RouteTableId": "rtb-22574640",
        "VpcId": "vpc-a01106c2",
        "PropagatingVgws": [],
        "Tags": [],
        "Routes": [
            {
                "GatewayId": "local",
```

```
                "DestinationCidrBlock": "10.0.0.0/16",
                "State": "active"
            }
        ]
    }
}
```

- API 세부 정보는 AWS CLI 명령 CreateRouteTable참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 VPC에 대한 라우팅 테이블을 생성합니다.

```
New-EC2RouteTable -VpcId vpc-12345678
```

출력:

```
Associations    : {}
PropagatingVgws : {}
Routes          : {}
RouteTableId    : rtb-1a2b3c4d
Tags            : {}
VpcId           : vpc-12345678
```

- API에 대한 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CreateRouteTable.

Ruby

SDK for Ruby

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
require "aws-sdk-ec2"
```

```
# Prerequisites:
#
# - A VPC in Amazon VPC.
# - A subnet in that VPC.
# - A gateway attached to that subnet.
#
# @param ec2_resource [Aws::EC2::Resource] An initialized
#   Amazon Elastic Compute Cloud (Amazon EC2) resource object.
# @param vpc_id [String] The ID of the VPC for the route table.
# @param subnet_id [String] The ID of the subnet for the route table.
# @param gateway_id [String] The ID of the gateway for the route.
# @param destination_cidr_block [String] The destination CIDR block
#   for the route.
# @param tag_key [String] The key portion of the tag for the route table.
# @param tag_value [String] The value portion of the tag for the route table.
# @return [Boolean] true if the route table was created and associated;
#   otherwise, false.
# @example
#   exit 1 unless route_table_created_and_associated?(
#     Aws::EC2::Resource.new(region: 'us-west-2'),
#     'vpc-0b6f769731EXAMPLE',
#     'subnet-03d9303b57EXAMPLE',
#     'igw-06ca90c011EXAMPLE',
#     '0.0.0.0/0',
#     'my-key',
#     'my-value'
#   )
def route_table_created_and_associated?(
  ec2_resource,
  vpc_id,
  subnet_id,
  gateway_id,
  destination_cidr_block,
  tag_key,
  tag_value
)
  route_table = ec2_resource.create_route_table(vpc_id: vpc_id)
  puts "Created route table with ID '#{route_table.id}'."
  route_table.create_tags(
    tags: [
      {
        key: tag_key,
        value: tag_value
```

```
      }
    ]
  )
  puts "Added tags to route table."
  route_table.create_route(
    destination_cidr_block: destination_cidr_block,
    gateway_id: gateway_id
  )
  puts "Created route with destination CIDR block " \
    "'#{destination_cidr_block}' and associated with gateway " \
    "with ID '#{gateway_id}'."
  route_table.associate_with_subnet(subnet_id: subnet_id)
  puts "Associated route table with subnet with ID '#{subnet_id}'."
  return true
rescue StandardError => e
  puts "Error creating or associating route table: #{e.message}"
  puts "If the route table was created but not associated, you should " \
    "clean up by deleting the route table."
  return false
end

# Example usage:
def run_me
  vpc_id = ""
  subnet_id = ""
  gateway_id = ""
  destination_cidr_block = ""
  tag_key = ""
  tag_value = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage: ruby ec2-ruby-example-create-route-table.rb " \
      "VPC_ID SUBNET_ID GATEWAY_ID DESTINATION_CIDR_BLOCK " \
      "TAG_KEY TAG_VALUE REGION"
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    puts "Example: ruby ec2-ruby-example-create-route-table.rb " \
      "vpc-0b6f769731EXAMPLE subnet-03d9303b57EXAMPLE igw-06ca90c011EXAMPLE " \
      "'0.0.0.0/0' my-key my-value us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  elsif ARGV.count.zero?
    vpc_id = "vpc-0b6f769731EXAMPLE"
    subnet_id = "subnet-03d9303b57EXAMPLE"
```

```
    gateway_id = "igw-06ca90c011EXAMPLE"
    destination_cidr_block = "0.0.0.0/0"
    tag_key = "my-key"
    tag_value = "my-value"
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    region = "us-west-2"
  # Otherwise, use the values as specified at the command prompt.
  else
    vpc_id = ARGV[0]
    subnet_id = ARGV[1]
    gateway_id = ARGV[2]
    destination_cidr_block = ARGV[3]
    tag_key = ARGV[4]
    tag_value = ARGV[5]
    region = ARGV[6]
  end

  ec2_resource = Aws::EC2::Resource.new(region: region)

  if route_table_created_and_associated?(
    ec2_resource,
    vpc_id,
    subnet_id,
    gateway_id,
    destination_cidr_block,
    tag_key,
    tag_value
  )
    puts "Route table created and associated."
  else
    puts "Route table not created or not associated."
  end
end

run_me if $PROGRAM_NAME == __FILE__
```

- API 세부 정보는 AWS SDK for Ruby API CreateRouteTable참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오SDK를 사용하여 Amazon EC2 리소스 생성 AWS. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateSecurityGroup** CLI와 함께 사용

다음 코드 예제는 CreateSecurityGroup의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 인스턴스 시작하기

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Create an Amazon EC2 security group.
    /// </summary>
    /// <param name="groupName">The name for the new security group.</param>
    /// <param name="groupDescription">A description of the new security group.</
param>
    /// <returns>The group Id of the new security group.</returns>
    public async Task<string> CreateSecurityGroup(string groupName, string
 groupDescription)
    {
        var response = await _amazonEC2.CreateSecurityGroupAsync(
            new CreateSecurityGroupRequest(groupName, groupDescription));

        return response.GroupId;
    }
```

- API 세부 정보는 AWS SDK for .NET API CreateSecurityGroup참조를 참조하십시오.

Bash

AWS CLI Bash 스크립트 사용

> **Note**
>
> 더 많은 정보가 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
#############################################################################
# function ec2_create_security_group
#
# This function creates an Amazon Elastic Compute Cloud (Amazon EC2) security
 group.
#
# Parameters:
#       -n security_group_name - The name of the security group.
#       -d security_group_description - The description of the security group.
#
# Returns:
#       The ID of the created security group, or an error message if the
 operation fails.
# And:
#       0 - If successful.
#       1 - If it fails.
#
#############################################################################
function ec2_create_security_group() {
  local security_group_name security_group_description response

  # Function to display usage information
  function usage() {
    echo "function ec2_create_security_group"
    echo "Creates an Amazon Elastic Compute Cloud (Amazon EC2) security group."
    echo "  -n security_group_name - The name of the security group."
    echo "  -d security_group_description - The description of the security
 group."
    echo ""
  }
```

```
 # Parse the command-line arguments
 while getopts "n:d:h" option; do
   case "${option}" in
     n) security_group_name="${OPTARG}" ;;
     d) security_group_description="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 # Validate the input parameters
 if [[ -z "$security_group_name" ]]; then
   errecho "ERROR: You must provide a security group name with the -n
parameter."
   return 1
 fi

 if [[ -z "$security_group_description" ]]; then
   errecho "ERROR: You must provide a security group description with the -d
parameter."
   return 1
 fi

 # Create the security group
 response=$(aws ec2 create-security-group \
   --group-name "$security_group_name" \
   --description "$security_group_description" \
   --query "GroupId" \
   --output text) || {
   aws_cli_error_log ${?}
   errecho "ERROR: AWS reports create-security-group operation failed."
   errecho "$response"
   return 1
 }
```

```
  echo "$response"
  return 0
}
```

이 예제에 사용된 유틸리티 함수

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
##############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
```

```
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- API 세부 정보는 AWS CLI 명령 CreateSecurityGroup참조를 참조하십시오.

C++

SDK for C++

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);

    Aws::EC2::Model::CreateSecurityGroupRequest request;

    request.SetGroupName(groupName);
    request.SetDescription(description);
    request.SetVpcId(vpcID);

    const Aws::EC2::Model::CreateSecurityGroupOutcome outcome =
            ec2Client.CreateSecurityGroup(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Failed to create security group:" <<
                  outcome.GetError().GetMessage() << std::endl;
        return false;
    }

    std::cout << "Successfully created security group named " << groupName <<
              std::endl;
```

- API 세부 정보는 AWS SDK for C++ API CreateSecurityGroup참조를 참조하십시오.

CLI

AWS CLI

EC2-Classic에 대한 보안 그룹을 생성하는 방법

이 예제에서는 이름이 MySecurityGroup인 보안 그룹을 생성합니다.

명령:

```
aws ec2 create-security-group --group-name MySecurityGroup --description "My
 security group"
```

출력:

```
{
    "GroupId": "sg-903004f8"
}
```

EC2-VPC에 대한 보안 그룹을 생성하는 방법

이 예제에서는 지정된 VPC에 대해 이름이 MySecurityGroup인 보안 그룹을 생성합니다.

명령:

```
aws ec2 create-security-group --group-name MySecurityGroup --description "My
 security group" --vpc-id vpc-1a2b3c4d
```

출력:

```
{
    "GroupId": "sg-903004f8"
}
```

자세한 내용은 AWS Command Line Interface 사용 설명서의 보안 그룹 사용을 참조하세요.

- API 세부 정보는 AWS CLI 명령 CreateSecurityGroup참조를 참조하십시오.

Java

SDK for Java 2.x

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
   public static String createSecurityGroup(Ec2Client ec2, String groupName,
String groupDesc, String vpcId,
           String myIpAddress) {
       try {
           CreateSecurityGroupRequest createRequest =
CreateSecurityGroupRequest.builder()
                   .groupName(groupName)
                   .description(groupDesc)
                   .vpcId(vpcId)
                   .build();

           CreateSecurityGroupResponse resp =
ec2.createSecurityGroup(createRequest);
           IpRange ipRange = IpRange.builder()
                   .cidrIp(myIpAddress + "/0")
                   .build();

           IpPermission ipPerm = IpPermission.builder()
                   .ipProtocol("tcp")
                   .toPort(80)
                   .fromPort(80)
                   .ipRanges(ipRange)
                   .build();

           IpPermission ipPerm2 = IpPermission.builder()
                   .ipProtocol("tcp")
                   .toPort(22)
                   .fromPort(22)
                   .ipRanges(ipRange)
                   .build();
```

```
            AuthorizeSecurityGroupIngressRequest authRequest =
 AuthorizeSecurityGroupIngressRequest.builder()
                    .groupName(groupName)
                    .ipPermissions(ipPerm, ipPerm2)
                    .build();

            ec2.authorizeSecurityGroupIngress(authRequest);
            System.out.println("Successfully added ingress policy to security
 group " + groupName);
            return resp.groupId();

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API CreateSecurityGroup참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { CreateSecurityGroupCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new CreateSecurityGroupCommand({
    // Up to 255 characters in length. Cannot start with sg-.
    GroupName: "SECURITY_GROUP_NAME",
    // Up to 255 characters in length.
    Description: "DESCRIPTION",
  });
```

```
  try {
    const { GroupId } = await client.send(command);
    console.log(GroupId);
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API CreateSecurityGroup참조를 참조하십시오.

Kotlin

SDK for Kotlin

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
suspend fun createEC2SecurityGroup(
    groupNameVal: String?,
    groupDescVal: String?,
    vpcIdVal: String?,
): String? {
    val request =
        CreateSecurityGroupRequest {
            groupName = groupNameVal
            description = groupDescVal
            vpcId = vpcIdVal
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val resp = ec2.createSecurityGroup(request)
        val ipRange =
            IpRange {
                cidrIp = "0.0.0.0/0"
            }

        val ipPerm =
```

```
            IpPermission {
                ipProtocol = "tcp"
                toPort = 80
                fromPort = 80
                ipRanges = listOf(ipRange)
            }

        val ipPerm2 =
            IpPermission {
                ipProtocol = "tcp"
                toPort = 22
                fromPort = 22
                ipRanges = listOf(ipRange)
            }

        val authRequest =
            AuthorizeSecurityGroupIngressRequest {
                groupName = groupNameVal
                ipPermissions = listOf(ipPerm, ipPerm2)
            }
        ec2.authorizeSecurityGroupIngress(authRequest)
        println("Successfully added ingress policy to Security Group
 $groupNameVal")
        return resp.groupId
    }
}
```

- API 세부 정보는 Kotlin API용AWS SDK 레퍼런스를 참조하세요 CreateSecurityGroup.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예제에서는 지정된 VPC에 대한 보안 그룹을 생성합니다.

```
New-EC2SecurityGroup -GroupName my-security-group -Description "my security
 group" -VpcId vpc-12345678
```

출력:

```
sg-12345678
```

예 2: 이 예제에서는 EC2-Classic용 보안 그룹을 생성합니다.

```
New-EC2SecurityGroup -GroupName my-security-group -Description "my security
 group"
```

출력:

```
sg-45678901
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CreateSecurityGroup.

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SecurityGroupWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) security group
 actions."""

    def __init__(self, ec2_resource, security_group=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param security_group: A Boto3 SecurityGroup object. This is a high-level
 object
                               that wraps security group actions.
        """
        self.ec2_resource = ec2_resource
        self.security_group = security_group

    @classmethod
```

```
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def create(self, group_name, group_description):
        """
        Creates a security group in the default virtual private cloud (VPC) of
the
        current account.

        :param group_name: The name of the security group to create.
        :param group_description: The description of the security group to
create.
        :return: A Boto3 SecurityGroup object that represents the newly created
security group.
        """
        try:
            self.security_group = self.ec2_resource.create_security_group(
                GroupName=group_name, Description=group_description
            )
        except ClientError as err:
            logger.error(
                "Couldn't create security group %s. Here's why: %s: %s",
                group_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return self.security_group
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 CreateSecurityGroup.

Ruby

SDK for Ruby

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
# This code example does the following:
# 1. Creates an Amazon Elastic Compute Cloud (Amazon EC2) security group.
# 2. Adds inbound rules to the security group.
# 3. Displays information about available security groups.
# 4. Deletes the security group.

require "aws-sdk-ec2"

# Creates an Amazon Elastic Compute Cloud (Amazon EC2) security group.
#
# Prerequisites:
#
# - A VPC in Amazon Virtual Private Cloud (Amazon VPC).
#
# @param ec2_client [Aws::EC2::Client] An initialized
#   Amazon EC2 client.
# @param group_name [String] A name for the security group.
# @param description [String] A description for the security group.
# @param vpc_id [String] The ID of the VPC for the security group.
# @return [String] The ID of security group that was created.
# @example
#   puts create_security_group(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'my-security-group',
#     'This is my security group.',
#     'vpc-6713dfEX'
#   )
def create_security_group(
  ec2_client,
  group_name,
  description,
```

```
  vpc_id
)
  security_group = ec2_client.create_security_group(
    group_name: group_name,
    description: description,
    vpc_id: vpc_id
  )
  puts "Created security group '#{group_name}' with ID " \
    "'#{security_group.group_id}' in VPC with ID '#{vpc_id}'."
  return security_group.group_id
rescue StandardError => e
  puts "Error creating security group: #{e.message}"
  return "Error"
end

# Adds an inbound rule to an Amazon Elastic Compute Cloud (Amazon EC2)
# security group.
#
# Prerequisites:
#
# - The security group.
#
# @param ec2_client [Aws::EC2::Client] An initialized Amazon EC2 client.
# @param security_group_id [String] The ID of the security group.
# @param ip_protocol [String] The network protocol for the inbound rule.
# @param from_port [String] The originating port for the inbound rule.
# @param to_port [String] The destination port for the inbound rule.
# @param cidr_ip_range [String] The CIDR IP range for the inbound rule.
# @return
# @example
#   exit 1 unless security_group_ingress_authorized?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'sg-030a858e078f1b9EX',
#     'tcp',
#     '80',
#     '80',
#     '0.0.0.0/0'
#   )
def security_group_ingress_authorized?(
  ec2_client,
  security_group_id,
  ip_protocol,
  from_port,
  to_port,
```

```
  cidr_ip_range
)
  ec2_client.authorize_security_group_ingress(
    group_id: security_group_id,
    ip_permissions: [
      {
        ip_protocol: ip_protocol,
        from_port: from_port,
        to_port: to_port,
        ip_ranges: [
          {
            cidr_ip: cidr_ip_range
          }
        ]
      }
    ]
  )
  puts "Added inbound rule to security group '#{security_group_id}' for protocol
 " \
    "'#{ip_protocol}' from port '#{from_port}' to port '#{to_port}' " \
    "with CIDR IP range '#{cidr_ip_range}'."
  return true
rescue StandardError => e
  puts "Error adding inbound rule to security group: #{e.message}"
  return false
end

# Displays information about a security group's IP permissions set in
# Amazon Elastic Compute Cloud (Amazon EC2).
#
# Prerequisites:
#
# - A security group with inbound rules, outbound rules, or both.
#
# @param p [Aws::EC2::Types::IpPermission] The IP permissions set.
# @example
#   ec2_client = Aws::EC2::Client.new(region: 'us-west-2')
#   response = ec2_client.describe_security_groups
#   unless sg.ip_permissions.empty?
#     describe_security_group_permissions(
#       response.security_groups[0].ip_permissions[0]
#     )
#   end
def describe_security_group_permissions(perm)
```

```
  print "  Protocol: #{perm.ip_protocol == '-1' ? 'All' : perm.ip_protocol}"

  unless perm.from_port.nil?
    if perm.from_port == "-1" || perm.from_port == -1
      print ", From: All"
    else
      print ", From: #{perm.from_port}"
    end
  end

  unless perm.to_port.nil?
    if perm.to_port == "-1" || perm.to_port == -1
      print ", To: All"
    else
      print ", To: #{perm.to_port}"
    end
  end

  if perm.key?(:ipv_6_ranges) && perm.ipv_6_ranges.count.positive?
    print ", CIDR IPv6: #{perm.ipv_6_ranges[0].cidr_ipv_6}"
  end

  if perm.key?(:ip_ranges) && perm.ip_ranges.count.positive?
    print ", CIDR IPv4: #{perm.ip_ranges[0].cidr_ip}"
  end

  print "\n"
end

# Displays information about available security groups in
# Amazon Elastic Compute Cloud (Amazon EC2).
#
# @param ec2_client [Aws::EC2::Client] An initialized Amazon EC2 client.
# @example
#   describe_security_groups(Aws::EC2::Client.new(region: 'us-west-2'))
def describe_security_groups(ec2_client)
  response = ec2_client.describe_security_groups

  if response.security_groups.count.positive?
    response.security_groups.each do |sg|
      puts "-" * (sg.group_name.length + 13)
      puts "Name:        #{sg.group_name}"
      puts "Description: #{sg.description}"
      puts "Group ID:    #{sg.group_id}"
```

```
      puts "Owner ID:    #{sg.owner_id}"
      puts "VPC ID:      #{sg.vpc_id}"

      if sg.tags.count.positive?
        puts "Tags:"
        sg.tags.each do |tag|
          puts "  Key: #{tag.key}, Value: #{tag.value}"
        end
      end

      unless sg.ip_permissions.empty?
        puts "Inbound rules:" if sg.ip_permissions.count.positive?
        sg.ip_permissions.each do |p|
          describe_security_group_permissions(p)
        end
      end

      unless sg.ip_permissions_egress.empty?
        puts "Outbound rules:" if sg.ip_permissions.count.positive?
        sg.ip_permissions_egress.each do |p|
          describe_security_group_permissions(p)
        end
      end
    end
  else
    puts "No security groups found."
  end
rescue StandardError => e
  puts "Error getting information about security groups: #{e.message}"
end

# Deletes an Amazon Elastic Compute Cloud (Amazon EC2)
# security group.
#
# Prerequisites:
#
# - The security group.
#
# @param ec2_client [Aws::EC2::Client] An initialized
#   Amazon EC2 client.
# @param security_group_id [String] The ID of the security group to delete.
# @return [Boolean] true if the security group was deleted; otherwise, false.
# @example
#   exit 1 unless security_group_deleted?(
```

```
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'sg-030a858e078f1b9EX'
#   )
def security_group_deleted?(ec2_client, security_group_id)
  ec2_client.delete_security_group(group_id: security_group_id)
  puts "Deleted security group '#{security_group_id}'."
  return true
rescue StandardError => e
  puts "Error deleting security group: #{e.message}"
  return false
end

# Example usage:
def run_me
  group_name = ""
  description = ""
  vpc_id = ""
  ip_protocol_http = ""
  from_port_http = ""
  to_port_http = ""
  cidr_ip_range_http = ""
  ip_protocol_ssh = ""
  from_port_ssh = ""
  to_port_ssh = ""
  cidr_ip_range_ssh = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:   ruby ec2-ruby-example-security-group.rb " \
      "GROUP_NAME DESCRIPTION VPC_ID IP_PROTOCOL_1 FROM_PORT_1 TO_PORT_1 " \
      "CIDR_IP_RANGE_1 IP_PROTOCOL_2 FROM_PORT_2 TO_PORT_2 " \
      "CIDR_IP_RANGE_2 REGION"
    puts "Example: ruby ec2-ruby-example-security-group.rb " \
      "my-security-group 'This is my security group.' vpc-6713dfEX " \
      "tcp 80 80 '0.0.0.0/0' tcp 22 22 '0.0.0.0/0' us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  elsif ARGV.count.zero?
    group_name = "my-security-group"
    description = "This is my security group."
    vpc_id = "vpc-6713dfEX"
    ip_protocol_http = "tcp"
    from_port_http = "80"
    to_port_http = "80"
```

```
  cidr_ip_range_http = "0.0.0.0/0"
  ip_protocol_ssh = "tcp"
  from_port_ssh = "22"
  to_port_ssh = "22"
  cidr_ip_range_ssh = "0.0.0.0/0"
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  region = "us-west-2"
# Otherwise, use the values as specified at the command prompt.
else
  group_name = ARGV[0]
  description = ARGV[1]
  vpc_id = ARGV[2]
  ip_protocol_http = ARGV[3]
  from_port_http = ARGV[4]
  to_port_http = ARGV[5]
  cidr_ip_range_http = ARGV[6]
  ip_protocol_ssh = ARGV[7]
  from_port_ssh = ARGV[8]
  to_port_ssh = ARGV[9]
  cidr_ip_range_ssh = ARGV[10]
  region = ARGV[11]
end

security_group_id = ""
security_group_exists = false
ec2_client = Aws::EC2::Client.new(region: region)

puts "Attempting to create security group..."
security_group_id = create_security_group(
  ec2_client,
  group_name,
  description,
  vpc_id
)
if security_group_id == "Error"
  puts "Could not create security group. Skipping this step."
else
  security_group_exists = true
end

if security_group_exists
  puts "Attempting to add inbound rules to security group..."
  unless security_group_ingress_authorized?(
    ec2_client,
```

```
      security_group_id,
      ip_protocol_http,
      from_port_http,
      to_port_http,
      cidr_ip_range_http
    )
      puts "Could not add inbound HTTP rule to security group. " \
        "Skipping this step."
    end

    unless security_group_ingress_authorized?(
      ec2_client,
      security_group_id,
      ip_protocol_ssh,
      from_port_ssh,
      to_port_ssh,
      cidr_ip_range_ssh
    )
      puts "Could not add inbound SSH rule to security group. " \
        "Skipping this step."
    end
  end

  puts "\nInformation about available security groups:"
  describe_security_groups(ec2_client)

  if security_group_exists
    puts "\nAttempting to delete security group..."
    unless security_group_deleted?(ec2_client, security_group_id)
      puts "Could not delete security group. You must delete it yourself."
    end
  end
end

run_me if $PROGRAM_NAME == __FILE__
```

- API 세부 정보는 AWS SDK for Ruby API CreateSecurityGroup참조를 참조하십시오.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    TRY.
        oo_result = lo_ec2->createsecuritygroup(                 " oo_result is
 returned for testing purposes. "
          iv_description = 'Security group example'
          iv_groupname = iv_security_group_name
          iv_vpcid = iv_vpc_id
        ).
        MESSAGE 'Security group created.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- API 세부 정보는 SAP용AWS SDK ABAP API 참조를 참조하십시오 CreateSecurityGroup.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateSnapshot** CLI와 함께 사용

다음 코드 예제는 CreateSnapshot의 사용 방법을 보여줍니다.

CLI

AWS CLI

스냅샷을 만들려면

이 예제 명령은 볼륨 ID와 스냅샷을 식별하는 간단한 설명을 사용하여 볼륨의 vol-1234567890abcdef0 스냅샷을 생성합니다.

명령:

```
aws ec2 create-snapshot --volume-id vol-1234567890abcdef0 --description "This is
 my root volume snapshot"
```

출력:

```
{
    "Description": "This is my root volume snapshot",
    "Tags": [],
    "Encrypted": false,
    "VolumeId": "vol-1234567890abcdef0",
    "State": "pending",
    "VolumeSize": 8,
    "StartTime": "2018-02-28T21:06:01.000Z",
    "Progress": "",
    "OwnerId": "012345678910",
    "SnapshotId": "snap-066877671789bd71b"
}
```

태그가 있는 스냅샷을 만들려면

이 예제 명령은 스냅샷을 생성하고 목적=prod와 비용센터=123이라는 두 개의 태그를 적용합니다.

명령:

```
aws ec2 create-snapshot --volume-id vol-1234567890abcdef0
 --description 'Prod backup' --tag-specifications
 'ResourceType=snapshot,Tags=[{Key=purpose,Value=prod},
{Key=costcenter,Value=123}]'
```

출력:

```
{
    "Description": "Prod backup",
    "Tags": [
        {
            "Value": "prod",
```

```
            "Key": "purpose"
        },
        {
            "Value": "123",
            "Key": "costcenter"
        }
    ],
    "Encrypted": false,
    "VolumeId": "vol-1234567890abcdef0",
    "State": "pending",
    "VolumeSize": 8,
    "StartTime": "2018-02-28T21:06:06.000Z",
    "Progress": "",
    "OwnerId": "012345678910",
    "SnapshotId": "snap-09ed24a70bc19bbe4"
}
```

- CreateSnapshot AWS CLIAPI에 대한 자세한 내용은 명령 참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 지정된 볼륨의 스냅샷을 생성합니다.

```
New-EC2Snapshot -VolumeId vol-12345678 -Description "This is a test"
```

출력:

```
DataEncryptionKeyId :
Description         : This is a test
Encrypted           : False
KmsKeyId            :
OwnerAlias          :
OwnerId             : 123456789012
Progress            :
SnapshotId          : snap-12345678
StartTime           : 12/22/2015 1:28:42 AM
State               : pending
StateMessage        :
Tags                : {}
VolumeId            : vol-12345678
```

```
VolumeSize          : 20
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CreateSnapshot.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateSpotDatafeedSubscription** CLI와 함께 사용

다음 코드 예제는 CreateSpotDatafeedSubscription의 사용 방법을 보여줍니다.

CLI

AWS CLI

스팟 인스턴스 데이터 피드를 생성하려면

다음 create-spot-datafeed-subscription 예제는 스팟 인스턴스 데이터 피드를 생성합니다.

```
aws ec2 create-spot-datafeed-subscription \
    --bucket my-bucket \
    --prefix spot-data-feed
```

출력:

```
{
    "SpotDatafeedSubscription": {
        "Bucket": "my-bucket",
        "OwnerId": "123456789012",
        "Prefix": "spot-data-feed",
        "State": "Active"
    }
}
```

데이터 피드는 지정한 Amazon S3 버킷에 저장됩니다. 이 데이터 피드의 파일 이름은 다음과 같은 형식입니다.

```
my-bucket.s3.amazonaws.com/spot-data-feed/123456789012.YYYY-MM-DD-
HH.n.abcd1234.gz
```

자세한 내용은 Linux 인스턴스용 Amazon Elastic Compute 클라우드 사용 설명서의 스팟 인스턴스 데이터 피드를 참조하십시오.

- API 세부 정보는 AWS CLI 명령 CreateSpotDatafeedSubscription참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 스팟 인스턴스 데이터 피드를 생성합니다.

```
New-EC2SpotDatafeedSubscription -Bucket my-s3-bucket -Prefix spotdata
```

출력:

```
Bucket  : my-s3-bucket
Fault   :
OwnerId : 123456789012
Prefix  : spotdata
State   : Active
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CreateSpotDatafeedSubscription.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateSubnet** CLI와 함께 사용

다음 코드 예제는 CreateSubnet의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: IPv4 CIDR 블록만 사용하여 서브넷을 생성하는 방법

다음 create-subnet 예제에서는 지정된 IPv4 CIDR 블록을 사용하여 지정된 VPC에서 서브넷을 생성합니다.

```
aws ec2 create-subnet \
    --vpc-id vpc-081ec835f3EXAMPLE \
    --cidr-block 10.0.0.0/24 \
    --tag-specifications ResourceType=subnet,Tags=[{Key=Name,Value=my-ipv4-only-
subnet}]
```

출력:

```
{
    "Subnet": {
        "AvailabilityZone": "us-west-2a",
        "AvailabilityZoneId": "usw2-az2",
        "AvailableIpAddressCount": 251,
        "CidrBlock": "10.0.0.0/24",
        "DefaultForAz": false,
        "MapPublicIpOnLaunch": false,
        "State": "available",
        "SubnetId": "subnet-0e99b93155EXAMPLE",
        "VpcId": "vpc-081ec835f3EXAMPLE",
        "OwnerId": "123456789012",
        "AssignIpv6AddressOnCreation": false,
        "Ipv6CidrBlockAssociationSet": [],
        "Tags": [
            {
                "Key": "Name",
                "Value": "my-ipv4-only-subnet"
            }
        ],
        "SubnetArn": "arn:aws:ec2:us-west-2:123456789012:subnet/
subnet-0e99b93155EXAMPLE"
    }
}
```

예제 2: IPv4 및 IPv6 CIDR 블록을 모두 사용하여 서브넷을 생성하는 방법

다음 `create-subnet` 예제에서는 지정된 IPv4 및 IPv6 CIDR 블록을 사용하여 지정된 VPC에서 서브넷을 생성합니다.

```
aws ec2 create-subnet \
    --vpc-id vpc-081ec835f3EXAMPLE \
    --cidr-block 10.0.0.0/24 \
    --ipv6-cidr-block 2600:1f16:cfe:3660::/64 \
```

```
    --tag-specifications ResourceType=subnet,Tags=[{Key=Name,Value=my-ipv4-ipv6-
subnet}]
```

출력:

```
{
    "Subnet": {
        "AvailabilityZone": "us-west-2a",
        "AvailabilityZoneId": "usw2-az2",
        "AvailableIpAddressCount": 251,
        "CidrBlock": "10.0.0.0/24",
        "DefaultForAz": false,
        "MapPublicIpOnLaunch": false,
        "State": "available",
        "SubnetId": "subnet-0736441d38EXAMPLE",
        "VpcId": "vpc-081ec835f3EXAMPLE",
        "OwnerId": "123456789012",
        "AssignIpv6AddressOnCreation": false,
        "Ipv6CidrBlockAssociationSet": [
            {
                "AssociationId": "subnet-cidr-assoc-06c5f904499fcc623",
                "Ipv6CidrBlock": "2600:1f13:cfe:3660::/64",
                "Ipv6CidrBlockState": {
                    "State": "associating"
                }
            }
        ],
        "Tags": [
            {
                "Key": "Name",
                "Value": "my-ipv4-ipv6-subnet"
            }
        ],
        "SubnetArn": "arn:aws:ec2:us-west-2:123456789012:subnet/
subnet-0736441d38EXAMPLE"
    }
}
```

예제 3: IPv6 CIDR 블록만 사용하여 서브넷을 생성하는 방법

다음 `create-subnet` 예제에서는 지정된 IPv6 CIDR 블록을 사용하여 지정된 VPC에서 서브넷을 생성합니다.

```
aws ec2 create-subnet \
    --vpc-id vpc-081ec835f3EXAMPLE \
    --ipv6-native \
    --ipv6-cidr-block 2600:1f16:115:200::/64 \
    --tag-specifications ResourceType=subnet,Tags=[{Key=Name,Value=my-ipv6-only-
subnet}]
```

출력:

```
{
    "Subnet": {
        "AvailabilityZone": "us-west-2a",
        "AvailabilityZoneId": "usw2-az2",
        "AvailableIpAddressCount": 0,
        "DefaultForAz": false,
        "MapPublicIpOnLaunch": false,
        "State": "available",
        "SubnetId": "subnet-03f720e7deEXAMPLE",
        "VpcId": "vpc-081ec835f3EXAMPLE",
        "OwnerId": "123456789012",
        "AssignIpv6AddressOnCreation": true,
        "Ipv6CidrBlockAssociationSet": [
            {
                "AssociationId": "subnet-cidr-assoc-01ef639edde556709",
                "Ipv6CidrBlock": "2600:1f13:cfe:3660::/64",
                "Ipv6CidrBlockState": {
                    "State": "associating"
                }
            }
        ],
        "Tags": [
            {
                "Key": "Name",
                "Value": "my-ipv6-only-subnet"
            }
        ],
        "SubnetArn": "arn:aws:ec2:us-west-2:123456789012:subnet/
subnet-03f720e7deEXAMPLE"
    }
}
```

자세한 내용은 Amazon VPC 사용 설명서의 VPC 및 서브넷을 참조하세요.

- API 세부 정보는 AWS CLI 명령 CreateSubnet참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 지정된 CIDR을 사용하여 서브넷을 생성합니다.

```
New-EC2Subnet -VpcId vpc-12345678 -CidrBlock 10.0.0.0/24
```

출력:

```
AvailabilityZone        : us-west-2c
AvailableIpAddressCount : 251
CidrBlock               : 10.0.0.0/24
DefaultForAz            : False
MapPublicIpOnLaunch     : False
State                   : pending
SubnetId                : subnet-1a2b3c4d
Tag                     : {}
VpcId                   : vpc-12345678
```

- API에 대한 자세한 내용은 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CreateSubnet.

Ruby

SDK for Ruby

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
require "aws-sdk-ec2"

# Creates a subnet within a virtual private cloud (VPC) in
# Amazon Virtual Private Cloud (Amazon VPC) and then tags
```

```
# the subnet.
#
# Prerequisites:
#
# - A VPC in Amazon VPC.
#
# @param ec2_resource [Aws::EC2::Resource] An initialized
#   Amazon Elastic Compute Cloud (Amazon EC2) resource object.
# @param vpc_id [String] The ID of the VPC for the subnet.
# @param cidr_block [String] The IPv4 CIDR block for the subnet.
# @param availability_zone [String] The ID of the Availability Zone
#   for the subnet.
# @param tag_key [String] The key portion of the tag for the subnet.
# @param tag_vlue [String] The value portion of the tag for the subnet.
# @return [Boolean] true if the subnet was created and tagged;
#   otherwise, false.
# @example
#   exit 1 unless subnet_created_and_tagged?(
#     Aws::EC2::Resource.new(region: 'us-west-2'),
#     'vpc-6713dfEX',
#     '10.0.0.0/24',
#     'us-west-2a',
#     'my-key',
#     'my-value'
#   )
def subnet_created_and_tagged?(
  ec2_resource,
  vpc_id,
  cidr_block,
  availability_zone,
  tag_key,
  tag_value
)
  subnet = ec2_resource.create_subnet(
    vpc_id: vpc_id,
    cidr_block: cidr_block,
    availability_zone: availability_zone
  )
  subnet.create_tags(
    tags: [
      {
        key: tag_key,
        value: tag_value
      }
```

```
    ]
  )
  puts "Subnet created with ID '#{subnet.id}' in VPC with ID '#{vpc_id}' " \
    "and CIDR block '#{cidr_block}' in availability zone " \
    "'#{availability_zone}' and tagged with key '#{tag_key}' and " \
    "value '#{tag_value}'."
  return true
rescue StandardError => e
  puts "Error creating or tagging subnet: #{e.message}"
  return false
end

# Example usage:
def run_me
  vpc_id = ""
  cidr_block = ""
  availability_zone = ""
  tag_key = ""
  tag_value = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:   ruby ec2-ruby-example-create-subnet.rb " \
      "VPC_ID CIDR_BLOCK AVAILABILITY_ZONE TAG_KEY TAG_VALUE REGION"
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    puts "Example: ruby ec2-ruby-example-create-subnet.rb " \
      "vpc-6713dfEX 10.0.0.0/24 us-west-2a my-key my-value us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  elsif ARGV.count.zero?
    vpc_id = "vpc-6713dfEX"
    cidr_block = "10.0.0.0/24"
    availability_zone = "us-west-2a"
    tag_key = "my-key"
    tag_value = "my-value"
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    region = "us-west-2"
  # Otherwise, use the values as specified at the command prompt.
  else
    vpc_id = ARGV[0]
    cidr_block = ARGV[1]
    availability_zone = ARGV[2]
    tag_key = ARGV[3]
    tag_value = ARGV[4]
```

```
    region = ARGV[5]
  end

  ec2_resource = Aws::EC2::Resource.new(region: region)

  if subnet_created_and_tagged?(
    ec2_resource,
    vpc_id,
    cidr_block,
    availability_zone,
    tag_key,
    tag_value
  )
    puts "Subnet created and tagged."
  else
    puts "Subnet not created or not tagged."
  end
end

run_me if $PROGRAM_NAME == __FILE__
```

- API 세부 정보는 AWS SDK for Ruby API CreateSubnet참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오SDK를 사용하여 Amazon EC2 리소스 생성 AWS. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateTags** CLI와 함께 사용

다음 코드 예제는 CreateTags의 사용 방법을 보여줍니다.

C++

SDK for C++

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);

    Aws::EC2::Model::Tag nameTag;
    nameTag.SetKey("Name");
    nameTag.SetValue(instanceName);

    Aws::EC2::Model::CreateTagsRequest createRequest;
    createRequest.AddResources(instanceID);
    createRequest.AddTags(nameTag);

    Aws::EC2::Model::CreateTagsOutcome createOutcome = ec2Client.CreateTags(
            createRequest);
    if (!createOutcome.IsSuccess()) {
        std::cerr << "Failed to tag ec2 instance " << instanceID <<
                  " with name " << instanceName << ":" <<
                  createOutcome.GetError().GetMessage() << std::endl;
        return false;
    }
```

- API 세부 정보는 AWS SDK for C++ API CreateTags참조를 참조하십시오.

CLI

AWS CLI

예 1: 리소스에 태그 추가하기

다음 `create-tags` 예제에서는 지정된 이미지에 `Stack=production` 태그를 추가하거나 태그 키가 `Stack`인 AMI의 기존 태그를 덮어씁니다.

```
aws ec2 create-tags \
    --resources ami-1234567890abcdef0 \
    --tags Key=Stack,Value=production
```

자세한 내용은 Linux 인스턴스용 Amazon Elastic Compute 클라우드 사용 설명서의 주제 제목 입니다.

예 2: 여러 리소스에 태그 추가하기

다음 create-tags 예제에서는 AMI와 인스턴스에 대해 두 개의 태그를 추가하거나 덮어씁니다. 태그 중 하나에서 키(webserver)는 있지만 값이 없습니다(값이 빈 문자열로 설정됨). 다른 태그에는 키(stack)와 값(Production)이 있습니다.

```
aws ec2 create-tags \
    --resources ami-1a2b3c4d i-1234567890abcdef0 \
    --tags Key=webserver,Value=   Key=stack,Value=Production
```

자세한 내용은 Linux 인스턴스용 Amazon Elastic Compute 클라우드 사용 설명서의 주제 제목입니다.

예 3: 특수 문자가 포함된 태그를 추가하려면

다음 create-tags 예제에서는 인스턴스에 [Group]=test 태그를 추가합니다. 대괄호([및])는 이스케이프해야 하는 특수 문자입니다. 다음 예제에서는 각 환경에 적합한 줄 연속 문자도 사용합니다.

Windows를 사용하는 경우 다음과 같이 특수 문자가 있는 요소를 큰따옴표(")로 묶은 다음, 각 큰따옴표 문자 앞에 백슬래시(\)를 붙입니다.

```
aws ec2 create-tags ^
    --resources i-1234567890abcdef0 ^
    --tags Key=\"[Group]\",Value=test
```

PowerShellWindows를 사용하는 경우 다음과 같이 특수 문자가 있는 값을 큰따옴표 (“) 로 묶고 각 큰따옴표 문자 앞에 백슬래시 (\) 를 붙인 다음 전체 키와 값 구조를 작은따옴표 (') 로 묶습니다.

```
aws ec2 create-tags `
    --resources i-1234567890abcdef0 `
    --tags 'Key=\"[Group]\",Value=test'
```

Linux 또는 OS X를 사용하는 경우 다음과 같이 특수 문자가 있는 요소를 큰따옴표(")로 묶은 다음, 전체 키 및 값 구조를 작음따옴표(')로 묶습니다.

```
aws ec2 create-tags \
    --resources i-1234567890abcdef0 \
    --tags 'Key="[Group]",Value=test'
```

자세한 내용은 Linux 인스턴스용 Amazon Elastic Compute 클라우드 사용 설명서의 주제 제목입니다.

- API 세부 정보는 AWS CLI 명령 CreateTags참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 지정된 리소스에 단일 태그를 추가합니다. 태그 키는 'MyTag'이고 태그 값은 myTagValue "입니다. 이 예제에서 사용하는 구문에는 PowerShell 버전 3 이상이 필요합니다.

```
New-EC2Tag -Resource i-12345678 -Tag @{ Key="myTag"; Value="myTagValue" }
```

예 2: 이 예제는 지정된 리소스를 업데이트하거나 지정된 태그를 추가합니다. 이 예제에서 사용하는 구문에는 PowerShell 버전 3 이상이 필요합니다.

```
New-EC2Tag -Resource i-12345678 -Tag @( @{ Key="myTag"; Value="newTagValue" },
 @{ Key="test"; Value="anotherTagValue" } )
```

예 3: PowerShell 버전 2에서는 New-Object를 사용하여 Tag 매개 변수에 대한 태그를 만들어야 합니다.

```
$tag = New-Object Amazon.EC2.Model.Tag
$tag.Key = "myTag"
$tag.Value = "myTagValue"

New-EC2Tag -Resource i-12345678 -Tag $tag
```

- API에 대한 자세한 내용은 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CreateTags.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateVolume** CLI와 함께 사용

다음 코드 예제는 `CreateVolume`의 사용 방법을 보여줍니다.

CLI

AWS CLI

빈 범용 SSD (gp2) 볼륨을 만들려면

다음 `create-volume` 예시에서는 지정된 가용 영역에 80GiB 범용 SSD (gp2) 볼륨을 생성합니다. 현재 지역은 다음과 같아야 합니다. `us-east-1` 그렇지 않으면 `--region` 파라미터를 추가하여 명령에 사용할 지역을 지정할 수 있습니다.

```
aws ec2 create-volume \
    --volume-type gp2 \
    --size 80 \
    --availability-zone us-east-1a
```

출력:

```
{
    "AvailabilityZone": "us-east-1a",
    "Tags": [],
    "Encrypted": false,
    "VolumeType": "gp2",
    "VolumeId": "vol-1234567890abcdef0",
    "State": "creating",
    "Iops": 240,
    "SnapshotId": "",
    "CreateTime": "YYYY-MM-DDTHH:MM:SS.000Z",
    "Size": 80
}
```

볼륨 유형을 지정하지 않은 경우 기본 볼륨 유형은 입니다gp2.

```
aws ec2 create-volume \
    --size 80 \
    --availability-zone us-east-1a
```

예 2: 스냅샷에서 프로비저닝된 IOPS SSD (io1) 볼륨 생성하기

다음 `create-volume` 예시에서는 지정된 스냅샷을 사용하여 지정된 가용 영역에 1,000개의 프로비저닝된 IOPS가 있는 프로비저닝된 IOPS SSD (io1) 볼륨을 생성합니다.

```
aws ec2 create-volume \
```

```
    --volume-type io1 \
    --iops 1000 \
    --snapshot-id snap-066877671789bd71b \
    --availability-zone us-east-1a
```

출력:

```
{
    "AvailabilityZone": "us-east-1a",
    "Tags": [],
    "Encrypted": false,
    "VolumeType": "io1",
    "VolumeId": "vol-1234567890abcdef0",
    "State": "creating",
    "Iops": 1000,
    "SnapshotId": "snap-066877671789bd71b",
    "CreateTime": "YYYY-MM-DDTHH:MM:SS.000Z",
    "Size": 500
}
```

예 3: 암호화된 볼륨을 만들려면

다음 `create-volume` 예에서는 EBS 암호화용 기본 CMK를 사용하여 암호화된 볼륨을 생성합니다. 암호화가 기본적으로 비활성화된 경우 다음과 같이 `--encrypted` 파라미터를 지정해야 합니다.

```
aws ec2 create-volume \
    --size 80 \
    --encrypted \
    --availability-zone us-east-1a
```

출력:

```
{
    "AvailabilityZone": "us-east-1a",
    "Tags": [],
    "Encrypted": true,
    "VolumeType": "gp2",
    "VolumeId": "vol-1234567890abcdef0",
    "State": "creating",
    "Iops": 240,
    "SnapshotId": "",
```

```
    "CreateTime": "YYYY-MM-DDTHH:MM:SS.000Z",
    "Size": 80
}
```

암호화가 기본적으로 활성화되어 있는 경우 다음 예제 명령은 `--encrypted` 매개 변수가 없더라도 암호화된 볼륨을 만듭니다.

```
aws ec2 create-volume \
    --size 80 \
    --availability-zone us-east-1a
```

`--kms-key-id`파라미터를 사용하여 고객 관리형 CMK를 지정하는 경우 기본적으로 암호화가 활성화되어 있더라도 `--encrypted` 파라미터를 지정해야 합니다.

```
aws ec2 create-volume \
    --volume-type gp2 \
    --size 80 \
    --encrypted \
    --kms-key-id 0ea3fef3-80a7-4778-9d8c-1c0c6EXAMPLE \
    --availability-zone us-east-1a
```

예 4: 태그가 있는 볼륨 생성하기

다음 `create-volume` 예제에서는 볼륨을 생성하고 태그 두 개를 추가합니다.

```
aws ec2 create-volume \
    --availability-zone us-east-1a \
    --volume-type gp2 \
    --size 80 \
    --tag-specifications
 'ResourceType=volume,Tags=[{Key=purpose,Value=production},{Key=cost-
center,Value=cc123}]'
```

- API 세부 정보는 AWS CLI 명령 CreateVolume참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 지정된 볼륨을 생성합니다.

```
New-EC2Volume -Size 50 -AvailabilityZone us-west-2a -VolumeType gp2
```

출력:

```
Attachments      : {}
AvailabilityZone : us-west-2a
CreateTime       : 12/22/2015 1:42:07 AM
Encrypted        : False
Iops             : 150
KmsKeyId         :
Size             : 50
SnapshotId       :
State            : creating
Tags             : {}
VolumeId         : vol-12345678
VolumeType       : gp2
```

예 2: 이 예제 요청은 볼륨을 생성하고 스택 키와 프로덕션 값이 포함된 태그를 적용합니다.

```
$tag = @{ Key="stack"; Value="production" }

$tagspec = new-object Amazon.EC2.Model.TagSpecification
$tagspec.ResourceType = "volume"
$tagspec.Tags.Add($tag)

New-EC2Volume -Size 80 -AvailabilityZone "us-west-2a" -TagSpecification $tagspec
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CreateVolume.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateVpc** CLI와 함께 사용

다음 코드 예제는 CreateVpc의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: VPC를 생성하는 방법

다음 `create-vpc` 예제에서는 지정된 IPv4 CIDR 블록과 이름 태그를 사용하여 VPC를 생성합니다.

```
aws ec2 create-vpc \
    --cidr-block 10.0.0.0/16 \
    --tag-specifications ResourceType=vpc,Tags=[{Key=Name,Value=MyVpc}]
```

출력:

```
{
    "Vpc": {
        "CidrBlock": "10.0.0.0/16",
        "DhcpOptionsId": "dopt-5EXAMPLE",
        "State": "pending",
        "VpcId": "vpc-0a60eb65b4EXAMPLE",
        "OwnerId": "123456789012",
        "InstanceTenancy": "default",
        "Ipv6CidrBlockAssociationSet": [],
        "CidrBlockAssociationSet": [
            {
                "AssociationId": "vpc-cidr-assoc-07501b79ecEXAMPLE",
                "CidrBlock": "10.0.0.0/16",
                "CidrBlockState": {
                    "State": "associated"
                }
            }
        ],
        "IsDefault": false,
        "Tags": [
            {
                "Key": "Name",
                "Value": MyVpc"
            }
        ]
    }
}
```

예제 2: 전용 테넌시를 사용하여 VPC를 생성하는 방법

다음 create-vpc 예제에서는 지정된 IPv4 CIDR 블록과 전용 테넌시를 사용하여 VPC를 생성합니다.

```
aws ec2 create-vpc \
    --cidr-block 10.0.0.0/16 \
    --instance-tenancy dedicated
```

출력:

```
{
    "Vpc": {
        "CidrBlock": "10.0.0.0/16",
        "DhcpOptionsId": "dopt-19edf471",
        "State": "pending",
        "VpcId": "vpc-0a53287fa4EXAMPLE",
        "OwnerId": "111122223333",
        "InstanceTenancy": "dedicated",
        "Ipv6CidrBlockAssociationSet": [],
        "CidrBlockAssociationSet": [
            {
                "AssociationId": "vpc-cidr-assoc-00b24cc1c2EXAMPLE",
                "CidrBlock": "10.0.0.0/16",
                "CidrBlockState": {
                    "State": "associated"
                }
            }
        ],
        "IsDefault": false
    }
}
```

예제 3: IPv6 CIDR 블록을 사용하여 VPC를 생성하는 방법

다음 create-vpc 예제에서는 Amazon에서 제공하는 IPv6 CIDR 블록을 사용하여 VPC를 생성합니다.

```
aws ec2 create-vpc \
    --cidr-block 10.0.0.0/16 \
    --amazon-provided-ipv6-cidr-block
```

출력:

```
{
    "Vpc": {
        "CidrBlock": "10.0.0.0/16",
        "DhcpOptionsId": "dopt-dEXAMPLE",
        "State": "pending",
        "VpcId": "vpc-0fc5e3406bEXAMPLE",
        "OwnerId": "123456789012",
        "InstanceTenancy": "default",
        "Ipv6CidrBlockAssociationSet": [
            {
                "AssociationId": "vpc-cidr-assoc-068432c60bEXAMPLE",
                "Ipv6CidrBlock": "",
                "Ipv6CidrBlockState": {
                    "State": "associating"
                },
                "Ipv6Pool": "Amazon",
                "NetworkBorderGroup": "us-west-2"
            }
        ],
        "CidrBlockAssociationSet": [
            {
                "AssociationId": "vpc-cidr-assoc-0669f8f9f5EXAMPLE",
                "CidrBlock": "10.0.0.0/16",
                "CidrBlockState": {
                    "State": "associated"
                }
            }
        ],
        "IsDefault": false
    }
}
```

예제 4: IPAM 풀에서 CIDR을 사용하여 VPC를 생성하는 방법

다음 `create-vpc` 예제에서는 Amazon VPC IP 주소 관리자(IPAM) 풀에서 CIDR을 사용하여 VPC를 생성합니다.

Linux 및 macOS:

```
aws ec2 create-vpc \
    --ipv4-ipam-pool-id ipam-pool-0533048da7d823723 \
```

```
    --tag-specifications
 ResourceType=vpc,Tags='[{Key=Environment,Value="Preprod"},
{Key=Owner,Value="Build Team"}]'
```

Windows:

```
aws ec2 create-vpc ^
    --ipv4-ipam-pool-id ipam-pool-0533048da7d823723 ^
    --tag-specifications
 ResourceType=vpc,Tags=[{Key=Environment,Value="Preprod"},{Key=Owner,Value="Build
 Team"}]
```

출력:

```
{
    "Vpc": {
        "CidrBlock": "10.0.1.0/24",
        "DhcpOptionsId": "dopt-2afccf50",
        "State": "pending",
        "VpcId": "vpc-010e1791024eb0af9",
        "OwnerId": "123456789012",
        "InstanceTenancy": "default",
        "Ipv6CidrBlockAssociationSet": [],
        "CidrBlockAssociationSet": [
            {
                "AssociationId": "vpc-cidr-assoc-0a77de1d803226d4b",
                "CidrBlock": "10.0.1.0/24",
                "CidrBlockState": {
                    "State": "associated"
                }
            }
        ],
        "IsDefault": false,
        "Tags": [
            {
                "Key": "Environment",
                "Value": "Preprod"
            },
            {
                "Key": "Owner",
                "Value": "Build Team"
            }
        ]
```

```
    }
}
```

자세한 내용은 Amazon VPC IPAM 사용 설명서에서 IPAM 풀 CIDR을 사용하는 VPC 생성을 참조하세요.

- API 세부 정보는 AWS CLI 명령 CreateVpc참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제에서는 지정된 CIDR을 사용하여 VPC를 생성합니다. 또한 Amazon VPC는 VPC를 위해 기본 DHCP 옵션 세트, 기본 라우팅 테이블, 기본 네트워크 ACL 등을 생성합니다.

```
New-EC2VPC -CidrBlock 10.0.0.0/16
```

출력:

```
CidrBlock       : 10.0.0.0/16
DhcpOptionsId   : dopt-1a2b3c4d
InstanceTenancy : default
IsDefault       : False
State           : pending
Tags            : {}
VpcId           : vpc-12345678
```

- API에 대한 세부 정보는 Cmdlet 참조를 참조하십시오 CreateVpc.AWS Tools for PowerShell

Ruby

SDK for Ruby

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
require "aws-sdk-ec2"

# Creates a virtual private cloud (VPC) in
# Amazon Virtual Private Cloud (Amazon VPC) and then tags
# the VPC.
#
# @param ec2_resource [Aws::EC2::Resource] An initialized
#   Amazon Elastic Compute Cloud (Amazon EC2) resource object.
# @param cidr_block [String] The IPv4 CIDR block for the subnet.
# @param tag_key [String] The key portion of the tag for the VPC.
# @param tag_value [String] The value portion of the tag for the VPC.
# @return [Boolean] true if the VPC was created and tagged;
#   otherwise, false.
# @example
#   exit 1 unless vpc_created_and_tagged?(
#     Aws::EC2::Resource.new(region: 'us-west-2'),
#     '10.0.0.0/24',
#     'my-key',
#     'my-value'
#   )
def vpc_created_and_tagged?(
  ec2_resource,
  cidr_block,
  tag_key,
  tag_value
)
  vpc = ec2_resource.create_vpc(cidr_block: cidr_block)

  # Create a public DNS by enabling DNS support and DNS hostnames.
  vpc.modify_attribute(enable_dns_support: { value: true })
  vpc.modify_attribute(enable_dns_hostnames: { value: true })

  vpc.create_tags(tags: [{ key: tag_key, value: tag_value }])

  puts "Created VPC with ID '#{vpc.id}' and tagged with key " \
    "'#{tag_key}' and value '#{tag_value}'."
  return true
rescue StandardError => e
  puts "#{e.message}"
  return false
end

# Example usage:
def run_me
```

```
  cidr_block = ""
  tag_key = ""
  tag_value = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:   ruby ec2-ruby-example-create-vpc.rb " \
      "CIDR_BLOCK TAG_KEY TAG_VALUE REGION"
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    puts "Example: ruby ec2-ruby-example-create-vpc.rb " \
      "10.0.0.0/24 my-key my-value us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  elsif ARGV.count.zero?
    cidr_block = "10.0.0.0/24"
    tag_key = "my-key"
    tag_value = "my-value"
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    region = "us-west-2"
  # Otherwise, use the values as specified at the command prompt.
  else
    cidr_block = ARGV[0]
    tag_key = ARGV[1]
    tag_value = ARGV[2]
    region = ARGV[3]
  end

  ec2_resource = Aws::EC2::Resource.new(region: region)

  if vpc_created_and_tagged?(
    ec2_resource,
    cidr_block,
    tag_key,
    tag_value
  )
    puts "VPC created and tagged."
  else
    puts "VPC not created or not tagged."
  end
end

run_me if $PROGRAM_NAME == __FILE__
```

- API 세부 정보는 AWS SDK for Ruby API CreateVpc참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오SDK를 사용하여 Amazon EC2 리소스 생성 AWS. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateVpcEndpoint** CLI와 함께 사용

다음 코드 예제는 CreateVpcEndpoint의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 게이트웨이 엔드포인트를 만들려면

다음 create-vpc-endpoint 예제는 us-east-1 리전의 VPC와 vpc-1a2b3c4d Amazon S3 사이에 게이트웨이 VPC 엔드포인트를 생성하고 라우팅 테이블을 엔드포인트와 연결합니다. rtb-11aa22bb

```
aws ec2 create-vpc-endpoint \
    --vpc-id vpc-1a2b3c4d \
    --service-name com.amazonaws.us-east-1.s3 \
    --route-table-ids rtb-11aa22bb
```

출력:

```
{
    "VpcEndpoint": {
        "PolicyDocument": "{\"Version\":\"2008-10-17\",\"Statement\":[{\"Sid\":
\"\",\"Effect\":\"Allow\",\"Principal\":\"\*\",\"Action\":\"\*\",\"Resource\":
\"\*\"}]}",
        "VpcId": "vpc-1a2b3c4d",
        "State": "available",
        "ServiceName": "com.amazonaws.us-east-1.s3",
        "RouteTableIds": [
            "rtb-11aa22bb"
        ],
        "VpcEndpointId": "vpc-1a2b3c4d",
        "CreationTimestamp": "2015-05-15T09:40:50Z"
    }
```

```
}
```

자세한 내용은 가이드의 게이트웨이 엔드포인트 생성을 참조하십시오.AWS PrivateLink

예 2: 인터페이스 엔드포인트 생성하기

다음 `create-vpc-endpoint` 예시에서는 리전의 VPC와 `vpc-1a2b3c4d` Amazon S3 사이에 인터페이스 VPC 엔드포인트를 생성합니다. `us-east-1` 이 명령은 서브넷에 엔드포인트를 생성하고`subnet-1a2b3c4d`, 이를 보안 그룹에 연결하고`sg-1a2b3c4d`, 키가 “Service”이고 값이 “S3"인 태그를 추가합니다.

```
aws ec2 create-vpc-endpoint \
    --vpc-id vpc-1a2b3c4d \
    --vpc-endpoint-type Interface \
    --service-name com.amazonaws.us-east-1.s3 \
    --subnet-ids subnet-7b16de0c \
    --security-group-id sg-1a2b3c4d \
    --tag-specifications ResourceType=vpc-endpoint,Tags=[{Key=service,Value=S3}]
```

출력:

```
{
    "VpcEndpoint": {
        "VpcEndpointId": "vpce-1a2b3c4d5e6f1a2b3",
        "VpcEndpointType": "Interface",
        "VpcId": "vpc-1a2b3c4d",
        "ServiceName": "com.amazonaws.us-east-1.s3",
        "State": "pending",
        "RouteTableIds": [],
        "SubnetIds": [
            "subnet-1a2b3c4d"
        ],
        "Groups": [
            {
                "GroupId": "sg-1a2b3c4d",
                "GroupName": "default"
            }
        ],
        "PrivateDnsEnabled": false,
        "RequesterManaged": false,
        "NetworkInterfaceIds": [
            "eni-0b16f0581c8ac6877"
```

```
        ],
        "DnsEntries": [
            {
                "DnsName": "*.vpce-1a2b3c4d5e6f1a2b3-9hnenorg.s3.us-
east-1.vpce.amazonaws.com",
                "HostedZoneId": "Z7HUB22UULQXV"
            },
            {
                "DnsName": "*.vpce-1a2b3c4d5e6f1a2b3-9hnenorg-us-east-1c.s3.us-
east-1.vpce.amazonaws.com",
                "HostedZoneId": "Z7HUB22UULQXV"
            }
        ],
        "CreationTimestamp": "2021-03-05T14:46:16.030000+00:00",
        "Tags": [
            {
                "Key": "service",
                "Value": "S3"
            }
        ],
        "OwnerId": "123456789012"
    }
}
```

자세한 내용은 사용 설명서의 인터페이스 엔드포인트 생성을 참조하십시오. AWS PrivateLink

예 3: 게이트웨이 로드 밸런서 엔드포인트를 만들려면

다음 `create-vpc-endpoint` 예시에서는 `vpc-111122223333aabbc` VPC와 게이트웨이 로드 밸런서를 사용하여 구성된 서비스 사이에 게이트웨이 로드 밸런서 엔드포인트를 생성합니다.

```
aws ec2 create-vpc-endpoint \
    --service-name com.amazonaws.vpce.us-east-1.vpce-svc-123123a1c43abc123 \
    --vpc-endpoint-type GatewayLoadBalancer \
    --vpc-id vpc-111122223333aabbc \
    --subnet-ids subnet-0011aabbcc2233445
```

출력:

```
{
    "VpcEndpoint": {
```

```
        "VpcEndpointId": "vpce-aabbaabbaabbaabba",
        "VpcEndpointType": "GatewayLoadBalancer",
        "VpcId": "vpc-111122223333aabbc",
        "ServiceName": "com.amazonaws.vpce.us-east-1.vpce-svc-123123a1c43abc123",
        "State": "pending",
        "SubnetIds": [
            "subnet-0011aabbcc2233445"
        ],
        "RequesterManaged": false,
        "NetworkInterfaceIds": [
            "eni-01010120203030405"
        ],
        "CreationTimestamp": "2020-11-11T08:06:03.522Z",
        "OwnerId": "123456789012"
    }
}
```

자세한 내용은 사용 설명서의 게이트웨이 Load Balancer 엔드포인트를 참조하십시오. AWS PrivateLink

- API 세부 정보는 AWS CLI 명령 CreateVpcEndpoint참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예제 1: 이 예시에서는 vpc-0fc1ff23f45b678eb에 com.amazonaws.eu-west-1.s3 서비스를 위한 새 VPC 엔드포인트를 생성합니다.

```
New-EC2VpcEndpoint -ServiceName com.amazonaws.eu-west-1.s3 -VpcId
 vpc-0fc1ff23f45b678eb
```

출력:

```
ClientToken VpcEndpoint
----------- -----------
            Amazon.EC2.Model.VpcEndpoint
```

- API에 대한 세부 정보는 Cmdlet 참조를 참조하십시오. CreateVpcEndpointAWS Tools for PowerShell

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateVpnConnection** CLI와 함께 사용

다음 코드 예제는 CreateVpnConnection의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 동적 라우팅을 사용하여 VPN 연결 생성하기

다음 create-vpn-connection 예에서는 지정된 가상 프라이빗 게이트웨이와 지정된 고객 게이트웨이 간에 VPN 연결을 만들고 VPN 연결에 태그를 적용합니다. 출력에는 고객 게이트웨이 디바이스의 구성 정보가 XML 형식으로 포함됩니다.

```
aws ec2 create-vpn-connection \
    --type ipsec.1 \
    --customer-gateway-id cgw-001122334455aabbc \
    --vpn-gateway-id vgw-1a1a1a1a1a1a2b2b2 \
    --tag-specification 'ResourceType=vpn-connection,Tags=[{Key=Name,Value=BGP-
VPN}]'
```

출력:

```
{
    "VpnConnection": {
        "CustomerGatewayConfiguration": "...configuration information...",
        "CustomerGatewayId": "cgw-001122334455aabbc",
        "Category": "VPN",
        "State": "pending",
        "VpnConnectionId": "vpn-123123123123abcab",
        "VpnGatewayId": "vgw-1a1a1a1a1a1a2b2b2",
        "Options": {
            "EnableAcceleration": false,
            "StaticRoutesOnly": false,
            "LocalIpv4NetworkCidr": "0.0.0.0/0",
            "RemoteIpv4NetworkCidr": "0.0.0.0/0",
            "TunnelInsideIpVersion": "ipv4",
            "TunnelOptions": [
                {},
```

```
                {}
            ]
        },
        "Routes": [],
        "Tags": [
             {
                "Key": "Name",
                "Value": "BGP-VPN"
            }
        ]
    }
}
```

자세한 내용은 사이트 간 VPN 사용 AWS 설명서의 사이트 간 VPN 작동 방식을 참조하십시오.AWS

예 2: 고정 라우팅을 사용하여 VPN 연결 생성하기

다음 create-vpn-connection 예에서는 지정된 가상 프라이빗 게이트웨이와 지정된 고객 게이트웨이 간에 VPN 연결을 생성합니다. 옵션은 고정 라우팅을 지정합니다. 출력에는 고객 게이트웨이 디바이스의 구성 정보가 XML 형식으로 포함됩니다.

```
aws ec2 create-vpn-connection \
    --type ipsec.1 \
    --customer-gateway-id cgw-001122334455aabbc \
    --vpn-gateway-id vgw-1a1a1a1a1a1a2b2b2 \
    --options "{\"StaticRoutesOnly\":true}"
```

출력:

```
{
    "VpnConnection": {
        "CustomerGatewayConfiguration": "..configuration information...",
        "CustomerGatewayId": "cgw-001122334455aabbc",
        "Category": "VPN",
        "State": "pending",
        "VpnConnectionId": "vpn-123123123123abcab",
        "VpnGatewayId": "vgw-1a1a1a1a1a1a2b2b2",
        "Options": {
            "EnableAcceleration": false,
            "StaticRoutesOnly": true,
            "LocalIpv4NetworkCidr": "0.0.0.0/0",
            "RemoteIpv4NetworkCidr": "0.0.0.0/0",
```

```
            "TunnelInsideIpVersion": "ipv4",
            "TunnelOptions": [
                {},
                {}
            ]
        },
        "Routes": [],
        "Tags": []
    }
}
```

자세한 내용은 사이트 간 VPN 사용 AWS 설명서의 사이트 간 VPN 작동 방식을 참조하십시오.AWS

예 3: VPN 연결을 만들고 자체 내부 CIDR 및 사전 공유 키를 지정하려면

다음 create-vpn-connection 예에서는 VPN 연결을 만들고 각 터널에 대해 내부 IP 주소 CIDR 블록과 사용자 지정 사전 공유 키를 지정합니다. 지정된 값이 정보에 반환됩니다. CustomerGatewayConfiguration

```
aws ec2 create-vpn-connection \
    --type ipsec.1 \
    --customer-gateway-id cgw-001122334455aabbc \
    --vpn-gateway-id vgw-1a1a1a1a1a1a2b2b2 \
    --options
 TunnelOptions='[{TunnelInsideCidr=169.254.12.0/30,PreSharedKey=ExamplePreSharedKey1},
{TunnelInsideCidr=169.254.13.0/30,PreSharedKey=ExamplePreSharedKey2}]'
```

출력:

```
{
    "VpnConnection": {
        "CustomerGatewayConfiguration": "..configuration information...",
        "CustomerGatewayId": "cgw-001122334455aabbc",
        "Category": "VPN",
        "State": "pending",
        "VpnConnectionId": "vpn-123123123123abcab",
        "VpnGatewayId": "vgw-1a1a1a1a1a1a2b2b2",
        "Options": {
            "EnableAcceleration": false,
            "StaticRoutesOnly": false,
            "LocalIpv4NetworkCidr": "0.0.0.0/0",
            "RemoteIpv4NetworkCidr": "0.0.0.0/0",
```

```
            "TunnelInsideIpVersion": "ipv4",
            "TunnelOptions": [
                {
                    "OutsideIpAddress": "203.0.113.3",
                    "TunnelInsideCidr": "169.254.12.0/30",
                    "PreSharedKey": "ExamplePreSharedKey1"
                },
                {
                    "OutsideIpAddress": "203.0.113.5",
                    "TunnelInsideCidr": "169.254.13.0/30",
                    "PreSharedKey": "ExamplePreSharedKey2"
                }
            ]
        },
        "Routes": [],
        "Tags": []
    }
}
```

자세한 내용은 사이트 간 VPN 사용 AWS 설명서의 사이트 간 VPN 작동 방식을 참조하십시오.AWS

예 4: IPv6 트래픽을 지원하는 VPN 연결을 만들려면

다음 `create-vpn-connection` 예에서는 지정된 전송 게이트웨이와 지정된 고객 게이트웨이 간에 IPv6 트래픽을 지원하는 VPN 연결을 만듭니다. 두 터널의 터널 옵션은 IKE 협상을 AWS 시작해야 하는 터널을 지정합니다.

```
aws ec2 create-vpn-connection \
    --type ipsec.1 \
    --transit-gateway-id tgw-12312312312312312 \
    --customer-gateway-id cgw-001122334455aabbc \
    --options TunnelInsideIpVersion=ipv6,TunnelOptions=[{StartupAction=start},
{StartupAction=start}]
```

출력:

```
{
    "VpnConnection": {
        "CustomerGatewayConfiguration": "..configuration information...",
        "CustomerGatewayId": "cgw-001122334455aabbc",
        "Category": "VPN",
        "State": "pending",
```

```
        "VpnConnectionId": "vpn-11111111122222222",
        "TransitGatewayId": "tgw-12312312312312312",
        "Options": {
            "EnableAcceleration": false,
            "StaticRoutesOnly": false,
            "LocalIpv6NetworkCidr": "::/0",
            "RemoteIpv6NetworkCidr": "::/0",
            "TunnelInsideIpVersion": "ipv6",
            "TunnelOptions": [
                {
                    "OutsideIpAddress": "203.0.113.3",
                    "StartupAction": "start"
                },
                {
                    "OutsideIpAddress": "203.0.113.5",
                    "StartupAction": "start"
                }
            ]
        },
        "Routes": [],
        "Tags": []
    }
}
```

자세한 내용은 사이트 간 VPN 사용 AWS 설명서의 사이트 간 VPN 작동 방식을 참조하십시오.AWS

- API 세부 정보는 명령 참조를 참조하십시오 CreateVpnConnection.AWS CLI

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 지정된 가상 프라이빗 게이트웨이와 지정된 고객 게이트웨이 간에 VPN 연결을 생성합니다. 출력에는 네트워크 관리자에게 필요한 구성 정보가 XML 형식으로 포함됩니다.

```
New-EC2VpnConnection -Type ipsec.1 -CustomerGatewayId cgw-1a2b3c4d -VpnGatewayId
 vgw-1a2b3c4d
```

출력:

```
CustomerGatewayConfiguration : [XML document]
```

```
CustomerGatewayId            : cgw-1a2b3c4d
Options                      :
Routes                       : {}
State                        : pending
Tags                         : {}
Type                         :
VgwTelemetry                 : {}
VpnConnectionId              : vpn-12345678
VpnGatewayId                 : vgw-1a2b3c4d
```

예 2: 이 예제는 VPN 연결을 만들고 지정된 이름의 파일에 구성을 캡처합니다.

```
(New-EC2VpnConnection -CustomerGatewayId cgw-1a2b3c4d -VpnGatewayId
 vgw-1a2b3c4d).CustomerGatewayConfiguration | Out-File C:\path\vpn-
configuration.xml
```

예 3: 이 예에서는 지정된 가상 프라이빗 게이트웨이와 지정된 고객 게이트웨이 간에 정적 라우팅을 사용하여 VPN 연결을 생성합니다.

```
New-EC2VpnConnection -Type ipsec.1 -CustomerGatewayId cgw-1a2b3c4d -VpnGatewayId
 vgw-1a2b3c4d -Options_StaticRoutesOnly $true
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CreateVpnConnection.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateVpnConnectionRoute** CLI와 함께 사용

다음 코드 예제는 CreateVpnConnectionRoute의 사용 방법을 보여줍니다.

CLI

AWS CLI

VPN 연결을 위한 고정 경로를 만들려면

이 예시에서는 지정된 VPN 연결에 대한 고정 경로를 만듭니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 create-vpn-connection-route --vpn-connection-id vpn-40f41529 --
destination-cidr-block 11.12.0.0/16
```

- API 세부 정보는 AWS CLI 명령 CreateVpnConnectionRoute참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 지정된 VPN 연결에 대해 지정된 고정 경로를 만듭니다.

```
New-EC2VpnConnectionRoute -VpnConnectionId vpn-12345678 -DestinationCidrBlock
 11.12.0.0/16
```

- API에 대한 자세한 내용은 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CreateVpnConnectionRoute.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **CreateVpnGateway** CLI와 함께 사용

다음 코드 예제는 CreateVpnGateway의 사용 방법을 보여줍니다.

CLI

AWS CLI

가상 프라이빗 게이트웨이를 만들려면

이 예시에서는 가상 프라이빗 게이트웨이를 생성합니다.

명령:

```
aws ec2 create-vpn-gateway --type ipsec.1
```

출력:

```
{
    "VpnGateway": {
        "AmazonSideAsn": 64512,
        "State": "available",
        "Type": "ipsec.1",
        "VpnGatewayId": "vgw-9a4cacf3",
        "VpcAttachments": []
    }
}
```

특정 Amazon 측 ASN으로 가상 프라이빗 게이트웨이를 생성하려면

이 예제에서는 가상 프라이빗 게이트웨이를 생성하고 BGP 세션의 Amazon 측에 대한 자율 시스템 번호 (ASN) 를 지정합니다.

명령:

```
aws ec2 create-vpn-gateway --type ipsec.1 --amazon-side-asn 65001
```

출력:

```
{
    "VpnGateway": {
        "AmazonSideAsn": 65001,
        "State": "available",
        "Type": "ipsec.1",
        "VpnGatewayId": "vgw-9a4cacf3",
        "VpcAttachments": []
    }
}
```

- API 세부 정보는 명령 참조를 참조하십시오 CreateVpnGateway.AWS CLI

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 지정된 가상 프라이빗 게이트웨이를 생성합니다.

```
New-EC2VpnGateway -Type ipsec.1
```

출력:

```
AvailabilityZone :
State            : available
Tags             : {}
Type             : ipsec.1
VpcAttachments   : {}
VpnGatewayId     : vgw-1a2b3c4d
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 CreateVpnGateway.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteCustomerGateway** CLI와 함께 사용

다음 코드 예제는 DeleteCustomerGateway의 사용 방법을 보여줍니다.

CLI

AWS CLI

고객 게이트웨이를 삭제하려면

이 예에서는 지정된 고객 게이트웨이를 삭제합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 delete-customer-gateway --customer-gateway-id cgw-0e11f167
```

- API 세부 정보는 AWS CLI 명령 DeleteCustomerGateway참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 지정된 고객 게이트웨이를 삭제합니다. Force 매개 변수도 함께 지정하지 않는 한 작업이 진행되기 전에 확인 메시지가 표시됩니다.

```
Remove-EC2CustomerGateway -CustomerGatewayId cgw-1a2b3c4d
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2CustomerGateway (DeleteCustomerGateway)" on
 Target "cgw-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DeleteCustomerGateway.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteDhcpOptions** CLI와 함께 사용

다음 코드 예제는 DeleteDhcpOptions의 사용 방법을 보여줍니다.

CLI

AWS CLI

DHCP 옵션 세트를 삭제하려면

이 예제에서는 지정된 DHCP 옵션 세트를 삭제합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 delete-dhcp-options --dhcp-options-id dopt-d9070ebb
```

- API 세부 정보는 AWS CLI 명령 DeleteDhcpOptions참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 지정된 DHCP 옵션 세트를 삭제합니다. Force 매개 변수도 지정하지 않은 경우 작업이 진행되기 전에 확인 메시지가 표시됩니다.

```
Remove-EC2DhcpOption -DhcpOptionsId dopt-1a2b3c4d
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2DhcpOption (DeleteDhcpOptions)" on Target
 "dopt-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DeleteDhcpOptions.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteFlowLogs** CLI와 함께 사용

다음 코드 예제는 DeleteFlowLogs의 사용 방법을 보여줍니다.

CLI

AWS CLI

흐름 로그를 삭제하려면

다음 delete-flow-logs 예제에서는 지정된 흐름 로그를 삭제합니다.

```
aws ec2 delete-flow-logs --flow-log-id fl-11223344556677889
```

출력:

```
{
    "Unsuccessful": []
}
```

- API 세부 정보는 AWS CLI 명령 DeleteFlowLogs참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예제 1: 이 예제는 주어진 FlowLogId fl-01a2b3456a789c01을 제거합니다.

```
Remove-EC2FlowLog -FlowLogId fl-01a2b3456a789c01
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EC2FlowLog (DeleteFlowLogs)" on target
 "fl-01a2b3456a789c01".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): Y
```

- AWS Tools for PowerShell API에 대한 DeleteFlowLogs자세한 내용은 Cmdlet 참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteInternetGateway** CLI와 함께 사용

다음 코드 예제는 DeleteInternetGateway의 사용 방법을 보여줍니다.

CLI

AWS CLI

인터넷 게이트웨이를 삭제하려면

다음 delete-internet-gateway 예제에서는 지정된 인터넷 게이트웨이를 삭제합니다.

```
aws ec2 delete-internet-gateway \
    --internet-gateway-id igw-0d0fb496b3EXAMPLE
```

이 명령은 출력을 생성하지 않습니다.

자세한 내용은 Amazon VPC 사용 설명서의 인터넷 게이트웨이를 참조하세요.

- API 세부 정보는 AWS CLI 명령 DeleteInternetGateway참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 지정된 인터넷 게이트웨이를 삭제합니다. Force 매개 변수도 함께 지정하지 않는 한 작업이 진행되기 전에 확인 메시지가 표시됩니다.

```
Remove-EC2InternetGateway -InternetGatewayId igw-1a2b3c4d
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2InternetGateway (DeleteInternetGateway)" on
 Target "igw-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DeleteInternetGateway.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteKeyPair** CLI와 함께 사용

다음 코드 예제는 DeleteKeyPair의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 인스턴스 시작하기

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Delete an Amazon EC2 key pair.
    /// </summary>
    /// <param name="keyPairName">The name of the key pair to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteKeyPair(string keyPairName)
    {
        try
        {
            await _amazonEC2.DeleteKeyPairAsync(new
DeleteKeyPairRequest(keyPairName)).ConfigureAwait(false);
            return true;
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Couldn't delete the key pair because:
{ex.Message}");
            return false;
        }
    }

    /// <summary>
    /// Delete the temporary file where the key pair information was saved.
    /// </summary>
    /// <param name="tempFileName">The path to the temporary file.</param>
    public void DeleteTempFile(string tempFileName)
```

```
    {
        if (File.Exists(tempFileName))
        {
            File.Delete(tempFileName);
        }
    }
```

- API 세부 정보는 AWS SDK for .NET API DeleteKeyPair참조를 참조하십시오.

Bash

AWS CLI Bash 스크립트 사용

> **Note**
>
> 더 많은 정보가 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
###############################################################################
# function ec2_delete_keypair
#
# This function deletes an Amazon EC2 ED25519 or 2048-bit RSA key pair.
#
# Parameters:
#       -n key_pair_name - A key pair name.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_delete_keypair() {
  local key_pair_name response

  local option OPTARG # Required to use getopts command in a function.
  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_delete_keypair"
    echo "Deletes an Amazon EC2 ED25519 or 2048-bit RSA key pair."
    echo "  -n key_pair_name - A key pair name."
    echo ""
```

```
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) key_pair_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$key_pair_name" ]]; then
    errecho "ERROR: You must provide a key pair name with the -n parameter."
    usage
    return 1
  fi

  response=$(aws ec2 delete-key-pair \
    --key-name "$key_pair_name") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports delete-key-pair operation failed.$response"
    return 1
  }

  return 0
}
```

이 예제에 사용된 유틸리티 함수

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
```

```
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
#############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- API 세부 정보는 AWS CLI 명령 DeleteKeyPair참조를 참조하십시오.

C++

SDK for C++

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::DeleteKeyPairRequest request;

    request.SetKeyName(keyPairName);
    const Aws::EC2::Model::DeleteKeyPairOutcome outcome =
ec2Client.DeleteKeyPair(
            request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Failed to delete key pair " << keyPairName <<
                  ":" << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully deleted key pair named " << keyPairName <<
                  std::endl;
    }
```

- API 세부 정보는 AWS SDK for C++ API DeleteKeyPair참조를 참조하십시오.

CLI

AWS CLI

키 페어를 삭제하는 방법

다음 `delete-key-pair` 예제에서는 지정된 키 쌍을 삭제합니다.

```
aws ec2 delete-key-pair \
    --key-name my-key-pair
```

출력:

```
{
    "Return": true,
    "KeyPairId": "key-03c8d3aceb53b507"
}
```

자세한 내용은 AWS 명령줄 인터페이스 사용 설명서의 키 쌍 생성 및 삭제를 참조하십시오.

- API 세부 정보는 AWS CLI 명령 DeleteKeyPair참조를 참조하십시오.

Java

SDK for Java 2.x

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    public static void deleteKeys(Ec2Client ec2, String keyPair) {
        try {
            DeleteKeyPairRequest request = DeleteKeyPairRequest.builder()
                    .keyName(keyPair)
                    .build();

            ec2.deleteKeyPair(request);
            System.out.println("Successfully deleted key pair named " + keyPair);

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API DeleteKeyPair참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { DeleteKeyPairCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new DeleteKeyPairCommand({
    KeyName: "KEY_PAIR_NAME",
  });

  try {
    await client.send(command);
    console.log("Successfully deleted key pair.");
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API DeleteKeyPair참조를 참조하십시오.

Kotlin

SDK for Kotlin

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
suspend fun deleteKeys(keyPair: String?) {
    val request =
        DeleteKeyPairRequest {
            keyName = keyPair
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.deleteKeyPair(request)
        println("Successfully deleted key pair named $keyPair")
    }
}
```

- API 세부 정보는 Kotlin API용AWS SDK 레퍼런스를 참조하세요 DeleteKeyPair.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예제는 지정된 키 쌍을 삭제합니다. Force 매개 변수도 함께 지정하지 않는 한 작업이 진행되기 전에 확인 메시지가 표시됩니다.

```
Remove-EC2KeyPair -KeyName my-key-pair
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2KeyPair (DeleteKeyPair)" on Target "my-key-pair".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DeleteKeyPair.

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class KeyPairWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) key pair
 actions."""

    def __init__(self, ec2_resource, key_file_dir, key_pair=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param key_file_dir: The folder where the private key information is
 stored.
                             This should be a secure folder.
        :param key_pair: A Boto3 KeyPair object. This is a high-level object that
                         wraps key pair actions.
        """
        self.ec2_resource = ec2_resource
        self.key_pair = key_pair
        self.key_file_path = None
        self.key_file_dir = key_file_dir

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource, tempfile.TemporaryDirectory())


    def delete(self):
        """
        Deletes a key pair.
        """
        if self.key_pair is None:
```

```
            logger.info("No key pair to delete.")
            return

        key_name = self.key_pair.name
        try:
            self.key_pair.delete()
            self.key_pair = None
        except ClientError as err:
            logger.error(
                "Couldn't delete key %s. Here's why: %s : %s",
                key_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 DeleteKeyPair.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    TRY.
        lo_ec2->deletekeypair( iv_keyname = iv_key_name ).
        MESSAGE 'Amazon EC2 key pair deleted.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- API 세부 정보는 SAP용AWS SDK ABAP API 참조를 참조하십시오 DeleteKeyPair.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteLaunchTemplate** CLI와 함께 사용

다음 코드 예제는 DeleteLaunchTemplate의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 복원력이 뛰어난 서비스 구축 및 관리

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
/// <summary>
/// Delete a launch template by name.
/// </summary>
/// <param name="templateName">The name of the template to delete.</param>
/// <returns>Async task.</returns>
public async Task DeleteTemplateByName(string templateName)
{
    try
    {
        await _amazonEc2.DeleteLaunchTemplateAsync(
            new DeleteLaunchTemplateRequest()
            {
                LaunchTemplateName = templateName
            });
    }
```

```
        catch (AmazonClientException)
        {
            Console.WriteLine($"Unable to delete template {templateName}.");
        }
    }
```

- API 세부 정보는 AWS SDK for .NET API DeleteLaunchTemplate참조를 참조하십시오.

CLI

AWS CLI

시작 템플릿을 삭제하는 방법

다음 예제에서는 지정된 시작 템플릿을 삭제합니다.

명령:

```
aws ec2 delete-launch-template --launch-template-id lt-0abcd290751193123
```

출력:

```
{
  "LaunchTemplate": {
      "LatestVersionNumber": 2,
      "LaunchTemplateId": "lt-0abcd290751193123",
      "LaunchTemplateName": "TestTemplate",
      "DefaultVersionNumber": 2,
      "CreatedBy": "arn:aws:iam::123456789012:root",
      "CreateTime": "2017-11-23T16:46:25.000Z"
  }
}
```

- API 세부 정보는 AWS CLI 명령 DeleteLaunchTemplate참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
  await client.send(
    new DeleteLaunchTemplateCommand({
      LaunchTemplateName: NAMES.launchTemplateName,
    }),
  );
```

- API 세부 정보는 AWS SDK for JavaScript API DeleteLaunchTemplate참조를 참조하십시오.

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
```

```
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def delete_template(self):
        """
        Deletes a launch template.
        """
        try:
            self.ec2_client.delete_launch_template(
                LaunchTemplateName=self.launch_template_name
            )
            self.delete_instance_profile(
                self.instance_profile_name, self.instance_role_name
            )
```

```
            log.info("Launch template %s deleted.", self.launch_template_name)
        except ClientError as err:
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.NotFoundException"
            ):
                log.info(
                    "Launch template %s does not exist, nothing to do.",
                    self.launch_template_name,
                )
            else:
                raise AutoScalerError(
                    f"Couldn't delete launch template
{self.launch_template_name}: {err}."
                )
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 DeleteLaunchTemplate.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteNetworkAcl** CLI와 함께 사용

다음 코드 예제는 DeleteNetworkAcl의 사용 방법을 보여줍니다.

CLI

AWS CLI

네트워크 ACL을 삭제하려면

이 예에서는 지정된 네트워크 ACL을 삭제합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 delete-network-acl --network-acl-id acl-5fb85d36
```

- API 세부 정보는 AWS CLI 명령 DeleteNetworkAcl참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 지정된 네트워크 ACL을 삭제합니다. Force 매개 변수도 함께 지정하지 않는 한 작업이 진행되기 전에 확인 메시지가 표시됩니다.

```
Remove-EC2NetworkAcl -NetworkAclId acl-12345678
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2NetworkAcl (DeleteNetworkAcl)" on Target
 "acl-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DeleteNetworkAcl.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteNetworkAclEntry** CLI와 함께 사용

다음 코드 예제는 DeleteNetworkAclEntry의 사용 방법을 보여줍니다.

CLI

AWS CLI

네트워크 ACL 항목을 삭제하려면

이 예제는 지정된 네트워크 ACL에서 인그레스 규칙 번호 100을 삭제합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 delete-network-acl-entry --network-acl-id acl-5fb85d36 --ingress --rule-
number 100
```

- API 세부 정보는 명령 참조를 참조하십시오 DeleteNetworkAclEntry.AWS CLI

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 지정된 네트워크 ACL에서 지정된 규칙을 제거합니다. Force 매개 변수도 지정하지 않는 한 작업이 진행되기 전에 확인 메시지가 표시됩니다.

```
Remove-EC2NetworkAclEntry -NetworkAclId acl-12345678 -Egress $false -RuleNumber
 100
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2NetworkAclEntry (DeleteNetworkAclEntry)" on
 Target "acl-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DeleteNetworkAclEntry.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **`DeleteNetworkInterface`** CLI와 함께 사용

다음 코드 예제는 `DeleteNetworkInterface`의 사용 방법을 보여줍니다.

CLI

AWS CLI

네트워크 인터페이스를 삭제하려면

이 예제는 지정된 네트워크 인터페이스를 삭제합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 delete-network-interface --network-interface-id eni-e5aa89a3
```

- API 세부 정보는 AWS CLI 명령 DeleteNetworkInterface참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 지정된 네트워크 인터페이스를 삭제합니다. Force 매개 변수도 함께 지정하지 않는 한 작업이 진행되기 전에 확인 메시지가 표시됩니다.

```
Remove-EC2NetworkInterface -NetworkInterfaceId eni-12345678
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2NetworkInterface (DeleteNetworkInterface)" on
 Target "eni-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DeleteNetworkInterface.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeletePlacementGroup** CLI와 함께 사용

다음 코드 예제는 DeletePlacementGroup의 사용 방법을 보여줍니다.

CLI

AWS CLI

플레이스먼트 그룹을 삭제하려면

이 예제 명령은 지정된 배치 그룹을 삭제합니다.

명령:

```
aws ec2 delete-placement-group --group-name my-cluster
```

- API 세부 정보는 AWS CLI 명령 DeletePlacementGroup참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 지정된 배치 그룹을 삭제합니다. Force 매개 변수도 함께 지정하지 않는 한 작업이 진행되기 전에 확인 메시지가 표시됩니다.

```
Remove-EC2PlacementGroup -GroupName my-placement-group
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2PlacementGroup (DeletePlacementGroup)" on Target
 "my-placement-group".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DeletePlacementGroup.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteRoute** CLI와 함께 사용

다음 코드 예제는 DeleteRoute의 사용 방법을 보여줍니다.

CLI

AWS CLI

라우트를 삭제하려면

이 예제는 지정된 라우팅 테이블에서 지정된 경로를 삭제합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 delete-route --route-table-id rtb-22574640 --destination-cidr-block
 0.0.0.0/0
```

- API 세부 정보는 AWS CLI 명령 DeleteRoute참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 지정된 라우팅 테이블에서 지정된 경로를 삭제합니다. Force 파라미터도 지정하지 않은 경우 작업이 진행되기 전에 확인 메시지가 표시됩니다.

```
Remove-EC2Route -RouteTableId rtb-1a2b3c4d -DestinationCidrBlock 0.0.0.0/0
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2Route (DeleteRoute)" on Target "rtb-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DeleteRoute.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteRouteTable** CLI와 함께 사용

다음 코드 예제는 DeleteRouteTable의 사용 방법을 보여줍니다.

CLI

AWS CLI

라우팅 테이블을 삭제하려면

이 예제는 지정된 라우팅 테이블을 삭제합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 delete-route-table --route-table-id rtb-22574640
```

- API 세부 정보는 AWS CLI 명령 DeleteRouteTable참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 지정된 라우팅 테이블을 삭제합니다. Force 매개 변수도 함께 지정하지 않는 한 작업이 진행되기 전에 확인 메시지가 표시됩니다.

```
Remove-EC2RouteTable -RouteTableId rtb-1a2b3c4d
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2RouteTable (DeleteRouteTable)" on Target
 "rtb-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DeleteRouteTable.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteSecurityGroup** CLI와 함께 사용

다음 코드 예제는 DeleteSecurityGroup의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 인스턴스 시작하기

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Delete an Amazon EC2 security group.
    /// </summary>
    /// <param name="groupName">The name of the group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteSecurityGroup(string groupId)
    {
        var response = await _amazonEC2.DeleteSecurityGroupAsync(new
DeleteSecurityGroupRequest { GroupId = groupId });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- API 세부 정보는 AWS SDK for .NET API DeleteSecurityGroup참조를 참조하십시오.

Bash

AWS CLI Bash 스크립트 사용

> **Note**
>
> 더 많은 정보가 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
#############################################################################
# function ec2_delete_security_group
#
# This function deletes an Amazon Elastic Compute Cloud (Amazon EC2) security
 group.
#
# Parameters:
#       -i security_group_id - The ID of the security group to delete.
#
# And:
#       0 - If successful.
#       1 - If it fails.
#############################################################################
function ec2_delete_security_group() {
  local security_group_id response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_delete_security_group"
    echo "Deletes an Amazon Elastic Compute Cloud (Amazon EC2) security group."
    echo "  -i security_group_id - The ID of the security group to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) security_group_id="${OPTARG}" ;;
      h)
        usage
        return 0
```

```
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$security_group_id" ]]; then
    errecho "ERROR: You must provide a security group ID with the -i parameter."
    usage
    return 1
  fi

  response=$(aws ec2 delete-security-group --group-id "$security_group_id" --
output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports delete-security-group operation failed.$response"
    return 1
  }

  return 0
}
```

이 예제에 사용된 유틸리티 함수

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
```

```
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
################################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- API 세부 정보는 AWS CLI 명령 DeleteSecurityGroup참조를 참조하십시오.

C++

SDK for C++

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::DeleteSecurityGroupRequest request;

    request.SetGroupId(securityGroupID);
    auto outcome = ec2Client.DeleteSecurityGroup(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Failed to delete security group " << securityGroupID <<
                  ":" << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully deleted security group " << securityGroupID <<
                  std::endl;
    }
```

- API 세부 정보는 AWS SDK for C++ API DeleteSecurityGroup참조를 참조하십시오.

CLI

AWS CLI

[EC2-Classic] 보안 그룹을 삭제하는 방법

이 예제에서는 이름이 `MySecurityGroup`인 보안 그룹을 삭제합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 delete-security-group --group-name MySecurityGroup
```

[EC2-VPC] 보안 그룹을 삭제하는 방법

이 예제에서는 ID가 `sg-903004f8`인 보안 그룹을 삭제합니다. 이름으로 EC2-VPC에 대한 보안 그룹을 참조할 수 없습니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 delete-security-group --group-id sg-903004f8
```

자세한 내용은 AWS Command Line Interface 사용 설명서의 보안 그룹 사용을 참조하세요.

- API 세부 정보는 AWS CLI 명령 DeleteSecurityGroup참조를 참조하십시오.

Java

SDK for Java 2.x

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
   public static void deleteEC2SecGroup(Ec2Client ec2, String groupId) {
       try {
           DeleteSecurityGroupRequest request =
DeleteSecurityGroupRequest.builder()
                   .groupId(groupId)
                   .build();

           ec2.deleteSecurityGroup(request);
           System.out.println("Successfully deleted security group with Id " +
groupId);

       } catch (Ec2Exception e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

- API 세부 정보는 AWS SDK for Java 2.x API DeleteSecurityGroup참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { DeleteSecurityGroupCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new DeleteSecurityGroupCommand({
    GroupId: "GROUP_ID",
  });

  try {
    await client.send(command);
    console.log("Security group deleted successfully.");
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API DeleteSecurityGroup참조를 참조하십시오.

Kotlin

SDK for Kotlin

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
suspend fun deleteEC2SecGroup(groupIdVal: String) {
    val request =
        DeleteSecurityGroupRequest {
            groupId = groupIdVal
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.deleteSecurityGroup(request)
        println("Successfully deleted Security Group with id $groupIdVal")
    }
}
```

- API 세부 정보는 Kotlin API용AWS SDK 레퍼런스를 참조하세요 DeleteSecurityGroup.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예에서는 지정된 EC2-VPC 보안 그룹을 삭제합니다. Force 매개 변수도 지정하지 않은 경우 작업이 진행되기 전에 확인 메시지가 표시됩니다.

```
Remove-EC2SecurityGroup -GroupId sg-12345678
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2SecurityGroup (DeleteSecurityGroup)" on Target
 "sg-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

예 2: 이 예에서는 EC2-Classic에 대해 지정된 보안 그룹을 삭제합니다.

```
Remove-EC2SecurityGroup -GroupName my-security-group -Force
```

- API에 대한 세부 정보는 Cmdlet DeleteSecurityGroup참조의 내용을 참조하십시오.AWS Tools for PowerShell

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SecurityGroupWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) security group
 actions."""

    def __init__(self, ec2_resource, security_group=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param security_group: A Boto3 SecurityGroup object. This is a high-level
 object
                               that wraps security group actions.
        """
        self.ec2_resource = ec2_resource
        self.security_group = security_group

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def delete(self):
        """
        Deletes the security group.
        """
        if self.security_group is None:
            logger.info("No security group to delete.")
            return

        group_id = self.security_group.id
```

```
        try:
            self.security_group.delete()
        except ClientError as err:
            logger.error(
                "Couldn't delete security group %s. Here's why: %s: %s",
                group_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 DeleteSecurityGroup.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    TRY.
        lo_ec2->deletesecuritygroup( iv_groupid = iv_security_group_id ).
        MESSAGE 'Security group deleted.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- API 세부 정보는 SAP용AWS SDK ABAP API 참조를 참조하십시오 DeleteSecurityGroup.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteSnapshot** CLI와 함께 사용

다음 코드 예제는 DeleteSnapshot의 사용 방법을 보여줍니다.

CLI

AWS CLI

스냅샷을 삭제하는 방법

이 예제 명령은 스냅샷 ID가 snap-1234567890abcdef0인 스냅샷을 삭제합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 delete-snapshot --snapshot-id snap-1234567890abcdef0
```

- API 세부 정보는 AWS CLI 명령 DeleteSnapshot참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 지정된 스냅샷을 삭제합니다. Force 매개 변수도 함께 지정하지 않는 한 작업이 진행되기 전에 확인 메시지가 표시됩니다.

```
Remove-EC2Snapshot -SnapshotId snap-12345678
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EC2Snapshot (DeleteSnapshot)" on target
 "snap-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DeleteSnapshot.

Rust

SDK for Rust

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
async fn delete_snapshot(client: &Client, id: &str) -> Result<(), Error> {
    client.delete_snapshot().snapshot_id(id).send().await?;

    println!("Deleted");

    Ok(())
}
```

- API에 대한 자세한 내용은 Rust용AWS SDK API 레퍼런스를 참조하십시오 DeleteSnapshot.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteSpotDatafeedSubscription** CLI와 함께 사용

다음 코드 예제는 `DeleteSpotDatafeedSubscription`의 사용 방법을 보여줍니다.

CLI

AWS CLI

스팟 인스턴스 데이터 피드 구독을 취소하려면

이 예제 명령은 계정의 스팟 데이터 피드 구독을 삭제합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 delete-spot-datafeed-subscription
```

- API 세부 정보는 AWS CLI 명령 DeleteSpotDatafeedSubscription참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 스팟 인스턴스 데이터 피드를 삭제합니다. Force 파라미터를 함께 지정하지 않는 한 작업이 진행되기 전에 확인 메시지가 표시됩니다.

```
Remove-EC2SpotDatafeedSubscription
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2SpotDatafeedSubscription
 (DeleteSpotDatafeedSubscription)" on Target "".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DeleteSpotDatafeedSubscription.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteSubnet** CLI와 함께 사용

다음 코드 예제는 DeleteSubnet의 사용 방법을 보여줍니다.

CLI

AWS CLI

서브넷을 삭제하려면

이 예제는 지정된 서브넷을 삭제합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 delete-subnet --subnet-id subnet-9d4a7b6c
```

- API 세부 정보는 AWS CLI 명령 DeleteSubnet참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 지정된 서브넷을 삭제합니다. Force 매개 변수도 지정하지 않은 경우 작업이 진행되기 전에 확인 메시지가 표시됩니다.

```
Remove-EC2Subnet -SubnetId subnet-1a2b3c4d
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2Subnet (DeleteSubnet)" on Target
 "subnet-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DeleteSubnet.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteTags** CLI와 함께 사용

다음 코드 예제는 DeleteTags의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 리소스에서 태그 삭제하기

다음 delete-tags 예제는 지정된 Stack=Test 이미지에서 태그를 삭제합니다. 값과 키 이름을 모두 지정하는 경우 태그의 값이 지정된 값과 일치하는 경우에만 태그가 삭제됩니다.

```
aws ec2 delete-tags \
    --resources ami-1234567890abcdef0 \
    --tags Key=Stack,Value=Test
```

태그 값을 지정하는 것은 선택 사항입니다. 다음 delete-tags 예제에서는 태그의 태그 값에 상관없이 지정된 인스턴스에서 키 이름이 purpose 있는 태그를 삭제합니다.

```
aws ec2 delete-tags \
    --resources i-1234567890abcdef0 \
    --tags Key=purpose
```

빈 문자열을 태그 값으로 지정하는 경우 태그의 값이 빈 문자열인 경우에만 태그가 삭제됩니다. 다음 delete-tags 예제에서는 삭제할 태그의 태그 값으로 빈 문자열을 지정합니다.

```
aws ec2 delete-tags \
    --resources i-1234567890abcdef0 \
    --tags Key=Name,Value=
```

예 2: 여러 리소스에서 태그 삭제하기

다음 delete-tags 예제는 인스턴스와 AMI 모두에서 ``목적=테스트`` 태그를 삭제합니다. 이전 예제에서 볼 수 있듯이 명령에서 태그 값을 생략할 수 있습니다.

```
aws ec2 delete-tags \
    --resources i-1234567890abcdef0 ami-1234567890abcdef0 \
    --tags Key=Purpose
```

- API 세부 정보는 AWS CLI 명령 DeleteTags참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 태그 값에 관계없이 지정된 리소스에서 지정된 태그를 삭제합니다. 이 예제에서 사용하는 구문에는 PowerShell 버전 3 이상이 필요합니다.

```
Remove-EC2Tag -Resource i-12345678 -Tag @{ Key="myTag" } -Force
```

예 2: 이 예제는 태그 값이 일치하는 경우에만 지정된 리소스에서 지정된 태그를 삭제합니다. 이 예제에서 사용하는 구문에는 PowerShell 버전 3 이상이 필요합니다.

```
Remove-EC2Tag -Resource i-12345678 -Tag @{ Key="myTag";Value="myTagValue" } -
Force
```

예 3: 이 예제는 태그 값에 관계없이 지정된 리소스에서 지정된 태그를 삭제합니다.

```
$tag = New-Object Amazon.EC2.Model.Tag
$tag.Key = "myTag"

Remove-EC2Tag -Resource i-12345678 -Tag $tag -Force
```

예 4: 이 예제는 태그 값이 일치하는 경우에만 지정된 리소스에서 지정된 태그를 삭제합니다.

```
$tag = New-Object Amazon.EC2.Model.Tag
$tag.Key = "myTag"
$tag.Value = "myTagValue"

Remove-EC2Tag -Resource i-12345678 -Tag $tag -Force
```

- API에 대한 자세한 내용은 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DeleteTags.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteVolume** CLI와 함께 사용

다음 코드 예제는 DeleteVolume의 사용 방법을 보여줍니다.

CLI

AWS CLI

볼륨을 삭제하려면

이 예제 명령은 볼륨 ID가 인 사용 가능한 볼륨을 삭제합니다. vol-049df61146c4d7901 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 delete-volume --volume-id vol-049df61146c4d7901
```

- API 세부 정보는 AWS CLI 명령 DeleteVolume참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 지정된 볼륨을 분리합니다. Force 매개 변수도 지정하지 않은 경우 작업이 진행되기 전에 확인 메시지가 표시됩니다.

```
Remove-EC2Volume -VolumeId vol-12345678
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EC2Volume (DeleteVolume)" on target
 "vol-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DeleteVolume.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteVpc** CLI와 함께 사용

다음 코드 예제는 DeleteVpc의 사용 방법을 보여줍니다.

CLI

AWS CLI

VPC를 삭제하는 방법

이 예제는 지정된 VPC를 삭제합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 delete-vpc --vpc-id vpc-a01106c2
```

- API 세부 정보는 AWS CLI 명령 DeleteVpc참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 지정된 VPC를 삭제합니다. Force 파라미터를 함께 지정하지 않는 한 작업이 진행되기 전에 확인 메시지가 표시됩니다.

```
Remove-EC2Vpc -VpcId vpc-12345678
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2Vpc (DeleteVpc)" on Target "vpc-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DeleteVpc.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteVpnConnection** CLI와 함께 사용

다음 코드 예제는 DeleteVpnConnection의 사용 방법을 보여줍니다.

CLI

AWS CLI

VPN 연결을 삭제하려면

이 예제는 지정된 VPN 연결을 삭제합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 delete-vpn-connection --vpn-connection-id vpn-40f41529
```

- API 세부 정보는 AWS CLI 명령 DeleteVpnConnection참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 지정된 VPN 연결을 삭제합니다. Force 매개 변수도 함께 지정하지 않는 한 작업이 진행되기 전에 확인 메시지가 표시됩니다.

```
Remove-EC2VpnConnection -VpnConnectionId vpn-12345678
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2VpnConnection (DeleteVpnConnection)" on Target
 "vpn-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DeleteVpnConnection.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteVpnConnectionRoute** CLI와 함께 사용

다음 코드 예제는 DeleteVpnConnectionRoute의 사용 방법을 보여줍니다.

CLI

AWS CLI

VPN 연결에서 고정 경로를 삭제하려면

이 예제는 지정된 VPN 연결에서 지정된 고정 경로를 삭제합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 delete-vpn-connection-route --vpn-connection-id vpn-40f41529 --
destination-cidr-block 11.12.0.0/16
```

- API 세부 정보는 AWS CLI 명령 DeleteVpnConnectionRoute참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 지정된 VPN 연결에서 지정된 고정 경로를 제거합니다. Force 매개 변수도 지정하지 않는 한 작업이 진행되기 전에 확인 메시지가 표시됩니다.

```
Remove-EC2VpnConnectionRoute -VpnConnectionId vpn-12345678 -DestinationCidrBlock
 11.12.0.0/16
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2VpnConnectionRoute (DeleteVpnConnectionRoute)" on
 Target "vpn-12345678".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DeleteVpnConnectionRoute.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeleteVpnGateway** CLI와 함께 사용

다음 코드 예제는 DeleteVpnGateway의 사용 방법을 보여줍니다.

CLI

AWS CLI

가상 프라이빗 게이트웨이를 삭제하려면

이 예에서는 지정된 가상 프라이빗 게이트웨이를 삭제합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 delete-vpn-gateway --vpn-gateway-id vgw-9a4cacf3
```

- API 세부 정보는 AWS CLI 명령 DeleteVpnGateway참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 지정된 가상 프라이빗 게이트웨이를 삭제합니다. Force 매개 변수도 함께 지정하지 않는 한 작업이 진행되기 전에 확인 메시지가 표시됩니다.

```
Remove-EC2VpnGateway -VpnGatewayId vgw-1a2b3c4d
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing operation "Remove-EC2VpnGateway (DeleteVpnGateway)" on Target
 "vgw-1a2b3c4d".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"):
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DeleteVpnGateway.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DeregisterImage** CLI와 함께 사용

다음 코드 예제는 DeregisterImage의 사용 방법을 보여줍니다.

CLI

AWS CLI

AMI 등록을 취소하려면

이 예제는 지정된 AMI를 등록 취소합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 deregister-image --image-id ami-4fa54026
```

- API 세부 정보는 AWS CLI 명령 DeregisterImage참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 지정된 AMI를 등록 취소합니다.

```
Unregister-EC2Image -ImageId ami-12345678
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DeregisterImage.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeAccountAttributes** CLI와 함께 사용

다음 코드 예제는 DescribeAccountAttributes의 사용 방법을 보여줍니다.

CLI

AWS CLI

AWS 계정의 모든 속성을 설명하려면

이 예제에서는 AWS 계정의 속성을 설명합니다.

명령:

```
aws ec2 describe-account-attributes
```

출력:

```
{
    "AccountAttributes": [
        {
            "AttributeName": "vpc-max-security-groups-per-interface",
            "AttributeValues": [
                {
                    "AttributeValue": "5"
                }
            ]
        },
        {
            "AttributeName": "max-instances",
            "AttributeValues": [
                {
                    "AttributeValue": "20"
                }
            ]
        },
        {
            "AttributeName": "supported-platforms",
            "AttributeValues": [
                {
                    "AttributeValue": "EC2"
                },
                {
                    "AttributeValue": "VPC"
                }
            ]
        },
        {
            "AttributeName": "default-vpc",
            "AttributeValues": [
                {
                    "AttributeValue": "none"
                }
```

```
            ]
        },
        {
            "AttributeName": "max-elastic-ips",
            "AttributeValues": [
                {
                    "AttributeValue": "5"
                }
            ]
        },
        {
            "AttributeName": "vpc-max-elastic-ips",
            "AttributeValues": [
                {
                    "AttributeValue": "5"
                }
            ]
        }
    ]
}
```

AWS 계정의 단일 속성에 대해 설명하려면

이 예제에서는 AWS 계정의 `supported-platforms` 속성을 설명합니다.

명령:

```
aws ec2 describe-account-attributes --attribute-names supported-platforms
```

출력:

```
{
    "AccountAttributes": [
        {
            "AttributeName": "supported-platforms",
            "AttributeValues": [
                {
                    "AttributeValue": "EC2"
                },
                {
                    "AttributeValue": "VPC"
                }
            ]
```

```
        }
    ]
}
```

- API 세부 정보는 AWS CLI 명령 DescribeAccountAttributes참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 해당 지역의 EC2-Classic 및 EC2-VPC 버전으로 인스턴스를 시작할 수 있는지 아니면 EC2-VPC 에서만 시작할 수 있는지를 설명합니다.

```
(Get-EC2AccountAttribute -AttributeName supported-platforms).AttributeValues
```

출력:

```
AttributeValue
--------------
EC2
VPC
```

예 2: 이 예제는 기본 VPC를 설명하며, 해당 지역에 기본 VPC가 없는 경우에는 '없음'으로 표시됩니다.

```
(Get-EC2AccountAttribute -AttributeName default-vpc).AttributeValues
```

출력:

```
AttributeValue
--------------
vpc-12345678
```

예 3: 이 예제에서는 실행할 수 있는 온디맨드 인스턴스의 최대 수를 설명합니다.

```
(Get-EC2AccountAttribute -AttributeName max-instances).AttributeValues
```

출력:

```
AttributeValue
--------------
20
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeAccountAttributes.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeAddresses** CLI와 함께 사용

다음 코드 예제는 DescribeAddresses의 사용 방법을 보여줍니다.

C++

SDK for C++

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::DescribeAddressesRequest request;
    auto outcome = ec2Client.DescribeAddresses(request);
    if (outcome.IsSuccess()) {
        std::cout << std::left << std::setw(20) << "InstanceId" <<
                  std::setw(15) << "Public IP" << std::setw(10) << "Domain" <<
                  std::setw(30) << "Allocation ID" << std::setw(25) <<
                  "NIC ID" << std::endl;

        const auto &addresses = outcome.GetResult().GetAddresses();
        for (const auto &address: addresses) {
            Aws::String domainString =
                    Aws::EC2::Model::DomainTypeMapper::GetNameForDomainType(
                            address.GetDomain());
```

```
            std::cout << std::left << std::setw(20) <<
                      address.GetInstanceId() << std::setw(15) <<
                      address.GetPublicIp() << std::setw(10) << domainString <<
                      std::setw(30) << address.GetAllocationId() << std::setw(25)
                      << address.GetNetworkInterfaceId() << std::endl;
        }
    }
    else {
        std::cerr << "Failed to describe Elastic IP addresses:" <<
                  outcome.GetError().GetMessage() << std::endl;
    }
```

- API 세부 정보는 AWS SDK for C++ API DescribeAddresses참조를 참조하십시오.

CLI

AWS CLI

예제 1: 모든 탄력적 IP 주소에 대한 세부 정보를 검색하는 방법

다음 `describe addresses` 예제에서는 탄력적 IP 주소에 대한 세부 정보를 표시합니다.

```
aws ec2 describe-addresses
```

출력:

```
{
    "Addresses": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "PublicIp": "198.51.100.0",
            "PublicIpv4Pool": "amazon",
            "Domain": "standard"
        },
        {
            "Domain": "vpc",
            "PublicIpv4Pool": "amazon",
            "InstanceId": "i-1234567890abcdef0",
            "NetworkInterfaceId": "eni-12345678",
            "AssociationId": "eipassoc-12345678",
            "NetworkInterfaceOwnerId": "123456789012",
```

```
            "PublicIp": "203.0.113.0",
            "AllocationId": "eipalloc-12345678",
            "PrivateIpAddress": "10.0.1.241"
        }
    ]
}
```

예제 2: EC2-VPC에 대한 탄력적 IP 주소의 세부 정보를 검색하는 방법

다음 `describe-addresses` 예제에서는 VPC의 인스턴스에서 사용할 탄력적 IP 주소의 세부 정보를 표시합니다.

```
aws ec2 describe-addresses \
    --filters "Name=domain,Values=vpc"
```

출력:

```
{
    "Addresses": [
        {
            "Domain": "vpc",
            "PublicIpv4Pool": "amazon",
            "InstanceId": "i-1234567890abcdef0",
            "NetworkInterfaceId": "eni-12345678",
            "AssociationId": "eipassoc-12345678",
            "NetworkInterfaceOwnerId": "123456789012",
            "PublicIp": "203.0.113.0",
            "AllocationId": "eipalloc-12345678",
            "PrivateIpAddress": "10.0.1.241"
        }
    ]
}
```

예제 3: 할당 ID로 지정된 탄력적 IP 주소의 세부 정보를 검색하는 방법

다음 `describe-addresses` 예제에서는 EC2-VPC의 인스턴스와 연결된, 지정된 할당 ID를 보유한 탄력적 IP 주소의 세부 정보를 표시합니다.

```
aws ec2 describe-addresses \
    --allocation-ids eipalloc-282d9641
```

출력:

```
{
    "Addresses": [
        {
            "Domain": "vpc",
            "PublicIpv4Pool": "amazon",
            "InstanceId": "i-1234567890abcdef0",
            "NetworkInterfaceId": "eni-1a2b3c4d",
            "AssociationId": "eipassoc-123abc12",
            "NetworkInterfaceOwnerId": "1234567891012",
            "PublicIp": "203.0.113.25",
            "AllocationId": "eipalloc-282d9641",
            "PrivateIpAddress": "10.251.50.12"
        }
    ]
}
```

예제 4: VPC 프라이빗 IP 주소로 지정된 탄력적 IP 주소의 세부 정보를 검색하는 방법

다음 `describe-addresses` 예제에서는 EC2-VPC 내 특정 프라이빗 IP 주소와 연결된 탄력적 IP 주소의 세부 정보를 표시합니다.

```
aws ec2 describe-addresses \
    --filters "Name=private-ip-address,Values=10.251.50.12"
```

예제 5: EC2-Classic에서 탄력적 IP 주소의 세부 정보를 검색하는 방법

다음 `describe-addresses` 예제에서는 EC2-Classic에서 사용할 탄력적 IP 주소의 세부 정보를 표시합니다.

```
aws ec2 describe-addresses \
    --filters "Name=domain,Values=standard"
```

출력:

```
{
    "Addresses": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "PublicIp": "203.0.110.25",
            "PublicIpv4Pool": "amazon",
            "Domain": "standard"
```

```
        }
    ]
}
```

예제 6: 퍼블릭 IP 주소로 지정된 탄력적 IP 주소의 세부 정보를 검색하는 방법

다음 `describe-addresses` 예제에서는 EC2-Classic의 인스턴스와 연결된, 값이 `203.0.110.25`인 탄력적 IP 주소의 세부 정보를 표시합니다.

```
aws ec2 describe-addresses \
    --public-ips 203.0.110.25
```

출력:

```
{
    "Addresses": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "PublicIp": "203.0.110.25",
            "PublicIpv4Pool": "amazon",
            "Domain": "standard"
        }
    ]
}
```

- API 세부 정보는 AWS CLI 명령 DescribeAddresses참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { DescribeAddressesCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";
```

```
export const main = async () => {
  const command = new DescribeAddressesCommand({
    // You can omit this property to show all addresses.
    AllocationIds: ["ALLOCATION_ID"],
  });

  try {
    const { Addresses } = await client.send(command);
    const addressList = Addresses.map((address) => ` • ${address.PublicIp}`);
    console.log("Elastic IP addresses:");
    console.log(addressList.join("\n"));
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API DescribeAddresses참조를 참조하십시오.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예에서는 EC2-Classic의 인스턴스에 지정된 엘라스틱 IP 주소를 설명합니다.

```
Get-EC2Address -AllocationId eipalloc-12345678
```

출력:

```
AllocationId            : eipalloc-12345678
AssociationId           : eipassoc-12345678
Domain                  : vpc
InstanceId              : i-87654321
NetworkInterfaceId      : eni-12345678
NetworkInterfaceOwnerId : 12345678
PrivateIpAddress        : 10.0.2.172
PublicIp                : 198.51.100.2
```

예 2: 이 예제에서는 VPC 내 인스턴스의 엘라스틱 IP 주소를 설명합니다. 이 구문에는 PowerShell 버전 3 이상이 필요합니다.

```
Get-EC2Address -Filter @{ Name="domain";Values="vpc" }
```

예 3: 이 예에서는 EC2-Classic의 인스턴스에 지정된 엘라스틱 IP 주소를 설명합니다.

```
Get-EC2Address -PublicIp 203.0.113.17
```

출력:

```
AllocationId            :
AssociationId           :
Domain                  : standard
InstanceId              : i-12345678
NetworkInterfaceId      :
NetworkInterfaceOwnerId :
PrivateIpAddress        :
PublicIp                : 203.0.113.17
```

예 4: 이 예제에서는 EC2-Classic의 인스턴스에 대한 엘라스틱 IP 주소를 설명합니다. 이 구문에는 PowerShell 버전 3 이상이 필요합니다.

```
Get-EC2Address -Filter @{ Name="domain";Values="standard" }
```

예 5: 이 예에서는 모든 엘라스틱 IP 주소를 설명합니다.

```
Get-EC2Address
```

예 6: 이 예제는 필터에 제공된 인스턴스 ID의 퍼블릭 및 프라이빗 IP를 반환합니다.

```
Get-EC2Address -Region eu-west-1 -Filter @{Name="instance-
id";Values="i-0c12d3f4f567ffb89"} | Select-Object PrivateIpAddress, PublicIp
```

출력:

```
PrivateIpAddress PublicIp
---------------- --------
10.0.0.99        63.36.5.227
```

예제 7: 이 예제는 할당 ID, 연결 ID 및 인스턴스 ID와 함께 모든 엘라스틱 IP를 검색합니다.

```
Get-EC2Address -Region eu-west-1 | Select-Object InstanceId, AssociationId,
 AllocationId, PublicIp
```

출력:

```
InstanceId          AssociationId              AllocationId
 PublicIp
----------          -------------              ------------
 --------
                                               eipalloc-012e3b456789e1fad
 17.212.120.178
i-0c123dfd3415bac67 eipassoc-0e123456bb7890bdb eipalloc-01cd23ebf45f7890c
 17.212.124.77
                                               eipalloc-012345678eeabcfad
 17.212.225.7
i-0123d405c67e89a0c eipassoc-0c123b456783966ba eipalloc-0123cdd456a8f7892
 37.216.52.173
i-0f1bf2f34c5678d09 eipassoc-0e12934568a952d96 eipalloc-0e1c23e4d5e6789e4
 37.218.222.278
i-012e3cb4df567e8aa eipassoc-0d1b2fa4d67d03810 eipalloc-0123f456f78a01b58
 37.210.82.27
i-0123bcf4b567890e1 eipassoc-01d2345f678903fb1 eipalloc-0e1db23cfef5c45c7
 37.215.222.270
```

예 8: 이 예제는 값이 'Prod'인 태그 키 '카테고리'와 일치하는 EC2 IP 주소 목록을 가져옵니다.

```
Get-EC2Address -Filter @{Name="tag:Category";Values="Prod"}
```

출력:

```
AllocationId            : eipalloc-0123f456f81a01b58
AssociationId           : eipassoc-0d1b23a456d103810
CustomerOwnedIp         :
CustomerOwnedIpv4Pool   :
Domain                  : vpc
InstanceId              : i-012e3cb4df567e1aa
NetworkBorderGroup      : eu-west-1
NetworkInterfaceId      : eni-0123f41d5a60d5f40
NetworkInterfaceOwnerId : 123456789012
PrivateIpAddress        : 192.168.1.84
PublicIp                : 34.250.81.29
PublicIpv4Pool          : amazon
```

```
Tags                    : {Category, Name}
```

- API 세부 정보는 Cmdlet 참조를 참조하십시오 DescribeAddresses.AWS Tools for PowerShell

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    TRY.
        oo_result = lo_ec2->describeaddresses( ) .                      "
 oo_result is returned for testing purposes. "
        DATA(lt_addresses) = oo_result->get_addresses( ).
        MESSAGE 'Retrieved information about Elastic IP addresses.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- API 세부 정보는 SAP용AWS SDK ABAP API 참조를 참조하십시오 DescribeAddresses.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeAvailabilityZones** CLI와 함께 사용

다음 코드 예제는 DescribeAvailabilityZones의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 복원력이 뛰어난 서비스 구축 및 관리

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Get a list of Availability Zones in the AWS Region of the Amazon EC2
Client.
    /// </summary>
    /// <returns>A list of availability zones.</returns>
    public async Task<List<string>> DescribeAvailabilityZones()
    {
        var zoneResponse = await _amazonEc2.DescribeAvailabilityZonesAsync(
            new DescribeAvailabilityZonesRequest());
        return zoneResponse.AvailabilityZones.Select(z => z.ZoneName).ToList();
    }
```

- API 세부 정보는 AWS SDK for .NET API DescribeAvailabilityZones참조를 참조하십시오.

C++

SDK for C++

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
```

```
    Aws::EC2::Model::DescribeAvailabilityZonesRequest describe_request;
    auto describe_outcome =
ec2Client.DescribeAvailabilityZones(describe_request);

    if (describe_outcome.IsSuccess()) {
        std::cout << std::left <<
                  std::setw(32) << "ZoneName" <<
                  std::setw(20) << "State" <<
                  std::setw(32) << "Region" << std::endl;

        const auto &zones =
                describe_outcome.GetResult().GetAvailabilityZones();

        for (const auto &zone: zones) {
            Aws::String stateString =

Aws::EC2::Model::AvailabilityZoneStateMapper::GetNameForAvailabilityZoneState(
                            zone.GetState());
            std::cout << std::left <<
                      std::setw(32) << zone.GetZoneName() <<
                      std::setw(20) << stateString <<
                      std::setw(32) << zone.GetRegionName() << std::endl;
        }
    }
    else {
        std::cerr << "Failed to describe availability zones:" <<
                  describe_outcome.GetError().GetMessage() << std::endl;
        result = false;
    }
```

- API 세부 정보는 AWS SDK for C++ API DescribeAvailabilityZones참조를 참조하십시오.

CLI

AWS CLI

가용 영역을 설명하는 방법

다음 `describe-availability-zones` 예제에서는 사용 가능한 가용 영역에 대한 세부 정보를 표시합니다. 응답에는 현재 리전의 가용 영역만 포함됩니다. 이 예제에서는 프로파일의 기본 `us-west-2`(오레곤) 리전을 사용합니다.

```
aws ec2 describe-availability-zones
```

출력:

```
{
    "AvailabilityZones": [
        {
            "State": "available",
            "OptInStatus": "opt-in-not-required",
            "Messages": [],
            "RegionName": "us-west-2",
            "ZoneName": "us-west-2a",
            "ZoneId": "usw2-az1",
            "GroupName": "us-west-2",
            "NetworkBorderGroup": "us-west-2"
        },
        {
            "State": "available",
            "OptInStatus": "opt-in-not-required",
            "Messages": [],
            "RegionName": "us-west-2",
            "ZoneName": "us-west-2b",
            "ZoneId": "usw2-az2",
            "GroupName": "us-west-2",
            "NetworkBorderGroup": "us-west-2"
        },
        {
            "State": "available",
            "OptInStatus": "opt-in-not-required",
            "Messages": [],
            "RegionName": "us-west-2",
            "ZoneName": "us-west-2c",
            "ZoneId": "usw2-az3",
            "GroupName": "us-west-2",
            "NetworkBorderGroup": "us-west-2"
        },
        {
            "State": "available",
            "OptInStatus": "opt-in-not-required",
            "Messages": [],
            "RegionName": "us-west-2",
            "ZoneName": "us-west-2d",
            "ZoneId": "usw2-az4",
```

```
            "GroupName": "us-west-2",
            "NetworkBorderGroup": "us-west-2"
        },
        {
            "State": "available",
            "OptInStatus": "opted-in",
            "Messages": [],
            "RegionName": "us-west-2",
            "ZoneName": "us-west-2-lax-1a",
            "ZoneId": "usw2-lax1-az1",
            "GroupName": "us-west-2-lax-1",
            "NetworkBorderGroup": "us-west-2-lax-1"
        }
    ]
}
```

- API 세부 정보는 AWS CLI 명령 DescribeAvailabilityZones참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 현재 지역에서 사용할 수 있는 가용 영역을 설명합니다.

```
Get-EC2AvailabilityZone
```

출력:

```
Messages    RegionName    State        ZoneName
--------    ----------    -----        --------
{}          us-west-2     available    us-west-2a
{}          us-west-2     available    us-west-2b
{}          us-west-2     available    us-west-2c
```

예 2: 이 예에서는 장애가 발생한 모든 가용 영역을 설명합니다. 이 예제에서 사용하는 구문에는 PowerShell 버전 3 이상이 필요합니다.

```
Get-EC2AvailabilityZone -Filter @{ Name="state";Values="impaired" }
```

예 3: PowerShell 버전 2에서는 New-Object를 사용하여 필터를 만들어야 합니다.

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = "impaired"

Get-EC2AvailabilityZone -Filter $filter
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeAvailabilityZones.

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is
                          created.
```

```
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def get_availability_zones(self):
        """
        Gets a list of Availability Zones in the AWS Region of the Amazon EC2
client.

        :return: The list of Availability Zones for the client Region.
        """
        try:
            response = self.ec2_client.describe_availability_zones()
            zones = [zone["ZoneName"] for zone in response["AvailabilityZones"]]
        except ClientError as err:
            raise AutoScalerError(f"Couldn't get availability zones: {err}.")
        else:
            return zones
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 DescribeAvailabilityZones.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    TRY.
        oo_result = lo_ec2->describeavailabilityzones( ) .
 " oo_result is returned for testing purposes. "
        DATA(lt_zones) = oo_result->get_availabilityzones( ).
        MESSAGE 'Retrieved information about Availability Zones.' TYPE 'I'.

      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- API 세부 정보는 SAP용AWS SDK ABAP API 참조를 참조하십시오 DescribeAvailabilityZones.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeBundleTasks** CLI와 함께 사용

다음 코드 예제는 DescribeBundleTasks의 사용 방법을 보여줍니다.

CLI

AWS CLI

번들 태스크에 대해 설명하려면

이 예제에서는 모든 번들 작업에 대해 설명합니다.

명령:

```
aws ec2 describe-bundle-tasks
```

출력:

```
{
  "BundleTasks": [
    {
      "UpdateTime": "2015-09-15T13:26:54.000Z",
      "InstanceId": "i-1234567890abcdef0",
      "Storage": {
        "S3": {
            "Prefix": "winami",
            "Bucket": "bundletasks"
        }
      },
      "State": "bundling",
      "StartTime": "2015-09-15T13:24:35.000Z",
      "Progress": "3%",
      "BundleId": "bun-2a4e041c"
    }
  ]
}
```

- API 세부 정보는 AWS CLI 명령 DescribeBundleTasks참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제에서는 지정된 번들 작업을 설명합니다.

```
Get-EC2BundleTask -BundleId bun-12345678
```

예 2: 이 예에서는 상태가 '완료' 또는 '실패'인 번들 작업을 설명합니다.

```
$filter = New-Object Amazon.EC2.Model.Filter
```

```
$filter.Name = "state"
$filter.Values = @( "complete", "failed" )

Get-EC2BundleTask -Filter $filter
```

- API 세부 정보는 Cmdlet 참조를 참조하십시오 DescribeBundleTasks.AWS Tools for PowerShell

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeCapacityReservations** CLI와 함께 사용

다음 코드 예제는 DescribeCapacityReservations의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 하나 이상의 용량 예약을 설명하려면

다음 describe-capacity-reservations 예는 현재 AWS 지역의 모든 수용 인원 예약에 대한 세부 정보를 표시합니다.

```
aws ec2 describe-capacity-reservations
```

출력:

```
{
    "CapacityReservations": [
        {
            "CapacityReservationId": "cr-1234abcd56EXAMPLE ",
            "EndDateType": "unlimited",
            "AvailabilityZone": "eu-west-1a",
            "InstanceMatchCriteria": "open",
            "Tags": [],
            "EphemeralStorage": false,
            "CreateDate": "2019-08-16T09:03:18.000Z",
            "AvailableInstanceCount": 1,
```

```
            "InstancePlatform": "Linux/UNIX",
            "TotalInstanceCount": 1,
            "State": "active",
            "Tenancy": "default",
            "EbsOptimized": true,
            "InstanceType": "a1.medium"
        },
        {
            "CapacityReservationId": "cr-abcdEXAMPLE9876ef ",
            "EndDateType": "unlimited",
            "AvailabilityZone": "eu-west-1a",
            "InstanceMatchCriteria": "open",
            "Tags": [],
            "EphemeralStorage": false,
            "CreateDate": "2019-08-07T11:34:19.000Z",
            "AvailableInstanceCount": 3,
            "InstancePlatform": "Linux/UNIX",
            "TotalInstanceCount": 3,
            "State": "cancelled",
            "Tenancy": "default",
            "EbsOptimized": true,
            "InstanceType": "m5.large"
        }
    ]
}
```

예 2: 하나 이상의 용량 예약에 대해 설명하기

다음 `describe-capacity-reservations` 예제는 지정된 용량 예약에 대한 세부 정보를 표시합니다.

```
aws ec2 describe-capacity-reservations \
    --capacity-reservation-ids cr-1234abcd56EXAMPLE
```

출력:

```
{
    "CapacityReservations": [
        {
            "CapacityReservationId": "cr-1234abcd56EXAMPLE",
            "EndDateType": "unlimited",
            "AvailabilityZone": "eu-west-1a",
```

```
            "InstanceMatchCriteria": "open",
            "Tags": [],
            "EphemeralStorage": false,
            "CreateDate": "2019-08-16T09:03:18.000Z",
            "AvailableInstanceCount": 1,
            "InstancePlatform": "Linux/UNIX",
            "TotalInstanceCount": 1,
            "State": "active",
            "Tenancy": "default",
            "EbsOptimized": true,
            "InstanceType": "a1.medium"
        }
    ]
}
```

자세한 내용은 Linux 인스턴스용 Amazon Elastic Compute 클라우드 사용 설명서의 용량 예약 보기를 참조하십시오.

- API 세부 정보는 AWS CLI 명령 DescribeCapacityReservations참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 해당 지역의 용량 예약 중 하나 이상을 설명합니다.

```
Get-EC2CapacityReservation -Region eu-west-1
```

출력:

```
AvailabilityZone       : eu-west-1b
AvailableInstanceCount : 2
CapacityReservationId  : cr-0c1f2345db6f7cdba
CreateDate             : 3/28/2019 9:29:41 AM
EbsOptimized           : True
EndDate                : 1/1/0001 12:00:00 AM
EndDateType            : unlimited
EphemeralStorage       : False
InstanceMatchCriteria  : open
InstancePlatform       : Windows
InstanceType           : m4.xlarge
State                  : active
```

```
Tags                    : {}
Tenancy                 : default
TotalInstanceCount      : 2
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeCapacityReservations.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeCustomerGateways** CLI와 함께 사용

다음 코드 예제는 DescribeCustomerGateways의 사용 방법을 보여줍니다.

CLI

AWS CLI

고객 게이트웨이를 설명하려면

이 예시는 고객 게이트웨이를 설명합니다.

명령:

```
aws ec2 describe-customer-gateways
```

출력:

```
{
    "CustomerGateways": [
        {
            "CustomerGatewayId": "cgw-b4dc3961",
            "IpAddress": "203.0.113.12",
            "State": "available",
            "Type": "ipsec.1",
            "BgpAsn": "65000"
        },
        {
            "CustomerGatewayId": "cgw-0e11f167",
            "IpAddress": "12.1.2.3",
```

```
            "State": "available",
            "Type": "ipsec.1",
            "BgpAsn": "65534"
        }
    ]
}
```

특정 고객 게이트웨이를 설명하려면

이 예에서는 지정된 고객 게이트웨이를 설명합니다.

명령:

```
aws ec2 describe-customer-gateways --customer-gateway-ids cgw-0e11f167
```

출력:

```
{
    "CustomerGateways": [
        {
            "CustomerGatewayId": "cgw-0e11f167",
            "IpAddress": "12.1.2.3",
            "State": "available",
            "Type": "ipsec.1",
            "BgpAsn": "65534"
        }
    ]
}
```

- API 세부 정보는 AWS CLI 명령 DescribeCustomerGateways참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 지정된 고객 게이트웨이를 설명합니다.

```
Get-EC2CustomerGateway -CustomerGatewayId cgw-1a2b3c4d
```

출력:

```
BgpAsn            : 65534
CustomerGatewayId : cgw-1a2b3c4d
IpAddress         : 203.0.113.12
State             : available
Tags              : {}
Type              : ipsec.1
```

예 2: 이 예에서는 상태가 보류 중이거나 사용 가능인 모든 고객 게이트웨이를 설명합니다.

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = @( "pending", "available" )

Get-EC2CustomerGateway -Filter $filter
```

예 3: 이 예에서는 모든 고객 게이트웨이를 설명합니다.

```
Get-EC2CustomerGateway
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeCustomerGateways.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeDhcpOptions** CLI와 함께 사용

다음 코드 예제는 DescribeDhcpOptions의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: DHCP 옵션 설명하기

다음 describe-dhcp-options 예제는 DHCP 옵션에 대한 세부 정보를 검색합니다.

```
aws ec2 describe-dhcp-options
```

출력:

```
{
    "DhcpOptions": [
        {
            "DhcpConfigurations": [
                {
                    "Key": "domain-name",
                    "Values": [
                        {
                            "Value": "us-east-2.compute.internal"
                        }
                    ]
                },
                {
                    "Key": "domain-name-servers",
                    "Values": [
                        {
                            "Value": "AmazonProvidedDNS"
                        }
                    ]
                }
            ],
            "DhcpOptionsId": "dopt-19edf471",
            "OwnerId": "111122223333"
        },
        {
            "DhcpConfigurations": [
                {
                    "Key": "domain-name",
                    "Values": [
                        {
                            "Value": "us-east-2.compute.internal"
                        }
                    ]
                },
                {
                    "Key": "domain-name-servers",
                    "Values": [
                        {
                            "Value": "AmazonProvidedDNS"
                        }
                    ]
                }
```

```
            ],
            "DhcpOptionsId": "dopt-fEXAMPLE",
            "OwnerId": "111122223333"
        }
    ]
}
```

자세한 내용은 AWS VPC 사용 설명서의 DHCP 옵션 세트 사용을 참조하십시오.

예 2: DHCP 옵션을 설명하고 출력을 필터링하려면

다음 `describe-dhcp-options` 예에서는 DHCP 옵션을 설명하고 필터를 사용하여 도메인 이름 서버에 있는 `example.com` DHCP 옵션만 반환합니다. 이 예제에서는 `--query` 매개변수를 사용하여 구성 정보와 ID만 출력에 표시합니다.

```
aws ec2 describe-dhcp-options \
    --filters Name=key,Values=domain-name-servers Name=value,Values=example.com \
    --query "DhcpOptions[*].[DhcpConfigurations,DhcpOptionsId]"
```

출력:

```
[
    [
        [
            {
                "Key": "domain-name",
                "Values": [
                    {
                        "Value": "example.com"
                    }
                ]
            },
            {
                "Key": "domain-name-servers",
                "Values": [
                    {
                        "Value": "172.16.16.16"
                    }
                ]
            }
        ],
        "dopt-001122334455667ab"
```

```
    ]
]
```

자세한 내용은 AWS VPC 사용 설명서의 DHCP 옵션 세트 사용을 참조하십시오.

- API 세부 정보는 AWS CLI 명령 DescribeDhcpOptions참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 DHCP 옵션 세트를 나열합니다.

```
Get-EC2DhcpOption
```

출력:

```
DhcpConfigurations                    DhcpOptionsId    Tag
------------------                    -------------    ---
{domain-name, domain-name-servers}    dopt-1a2b3c4d    {}
{domain-name, domain-name-servers}    dopt-2a3b4c5d    {}
{domain-name-servers}                 dopt-3a4b5c6d    {}
```

예 2: 이 예에서는 지정된 DHCP 옵션 세트에 대한 구성 세부 정보를 가져옵니다.

```
(Get-EC2DhcpOption -DhcpOptionsId dopt-1a2b3c4d).DhcpConfigurations
```

출력:

```
Key                  Values
---                  ------
domain-name          {abc.local}
domain-name-servers  {10.0.0.101, 10.0.0.102}
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeDhcpOptions.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeFlowLogs** CLI와 함께 사용

다음 코드 예제는 DescribeFlowLogs의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 모든 흐름 로그를 설명하려면

다음 describe-flow-logs 예제는 모든 흐름 로그의 세부 정보를 표시합니다.

```
aws ec2 describe-flow-logs
```

출력:

```
{
    "FlowLogs": [
        {
            "CreationTime": "2018-02-21T13:22:12.644Z",
            "DeliverLogsPermissionArn": "arn:aws:iam::123456789012:role/flow-
logs-role",
            "DeliverLogsStatus": "SUCCESS",
            "FlowLogId": "fl-aabbccdd112233445",
            "MaxAggregationInterval": 600,
            "FlowLogStatus": "ACTIVE",
            "LogGroupName": "FlowLogGroup",
            "ResourceId": "subnet-12345678901234567",
            "TrafficType": "ALL",
            "LogDestinationType": "cloud-watch-logs",
            "LogFormat": "${version} ${account-id} ${interface-id} ${srcaddr}
 ${dstaddr} ${srcport} ${dstport} ${protocol} ${packets} ${bytes} ${start} ${end}
 ${action} ${log-status}"
        },
        {
            "CreationTime": "2020-02-04T15:22:29.986Z",
            "DeliverLogsStatus": "SUCCESS",
            "FlowLogId": "fl-01234567890123456",
            "MaxAggregationInterval": 60,
            "FlowLogStatus": "ACTIVE",
            "ResourceId": "vpc-00112233445566778",
            "TrafficType": "ACCEPT",
            "LogDestinationType": "s3",
```

```
            "LogDestination": "arn:aws:s3:::my-flow-log-bucket/custom",
            "LogFormat": "${version} ${vpc-id} ${subnet-id} ${instance-id}
 ${interface-id} ${account-id} ${type} ${srcaddr} ${dstaddr} ${srcport}
 ${dstport} ${pkt-srcaddr} ${pkt-dstaddr} ${protocol} ${bytes} ${packets}
 ${start} ${end} ${action} ${tcp-flags} ${log-status}"
        }
    ]
}
```

예 2: 흐름 로그의 하위 집합을 설명하려면

다음 `describe-flow-logs` 예제에서는 필터를 사용하여 Amazon Logs의 지정된 로그 그룹에 있는 흐름 CloudWatch 로그의 세부 정보만 표시합니다.

```
aws ec2 describe-flow-logs \
    --filter "Name=log-group-name,Values=MyFlowLogs"
```

- API 세부 정보는 AWS CLI 명령 DescribeFlowLogs참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 로그 대상 유형이 's3'인 하나 이상의 흐름 로그를 설명합니다.

```
Get-EC2FlowLog -Filter @{Name="log-destination-type";Values="s3"}
```

출력:

```
CreationTime             : 2/25/2019 9:07:36 PM
DeliverLogsErrorMessage  :
DeliverLogsPermissionArn :
DeliverLogsStatus        : SUCCESS
FlowLogId                : fl-01b2e3d45f67f8901
FlowLogStatus            : ACTIVE
LogDestination           : arn:aws:s3:::my-bucket-dd-tata
LogDestinationType       : s3
LogGroupName             :
ResourceId               : eni-01d2dda3456b7e890
TrafficType              : ALL
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeFlowLogs.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeHostReservationOfferings** CLI와 함께 사용

다음 코드 예제는 DescribeHostReservationOfferings의 사용 방법을 보여줍니다.

CLI

AWS CLI

전용 호스트 예약 오퍼링 설명

이 예제에서는 구매 가능한 M4 인스턴스 패밀리의 전용 호스트 예약을 설명합니다.

명령:

```
aws ec2 describe-host-reservation-offerings --filter Name=instance-
family,Values=m4
```

출력:

```
{
  "OfferingSet": [
      {
          "HourlyPrice": "1.499",
          "OfferingId": "hro-03f707bf363b6b324",
          "InstanceFamily": "m4",
          "PaymentOption": "NoUpfront",
          "UpfrontPrice": "0.000",
          "Duration": 31536000
      },
      {
          "HourlyPrice": "1.045",
          "OfferingId": "hro-0ef9181cabdef7a02",
          "InstanceFamily": "m4",
          "PaymentOption": "NoUpfront",
          "UpfrontPrice": "0.000",
          "Duration": 94608000
      },
      {
```

```
            "HourlyPrice": "0.714",
            "OfferingId": "hro-04567a15500b92a51",
            "InstanceFamily": "m4",
            "PaymentOption": "PartialUpfront",
            "UpfrontPrice": "6254.000",
            "Duration": 31536000
        },
        {
            "HourlyPrice": "0.484",
            "OfferingId": "hro-0d5d7a9d23ed7fbfe",
            "InstanceFamily": "m4",
            "PaymentOption": "PartialUpfront",
            "UpfrontPrice": "12720.000",
            "Duration": 94608000
        },
        {
            "HourlyPrice": "0.000",
            "OfferingId": "hro-05da4108ca998c2e5",
            "InstanceFamily": "m4",
            "PaymentOption": "AllUpfront",
            "UpfrontPrice": "23913.000",
            "Duration": 94608000
        },
        {
            "HourlyPrice": "0.000",
            "OfferingId": "hro-0a9f9be3b95a3dc8f",
            "InstanceFamily": "m4",
            "PaymentOption": "AllUpfront",
            "UpfrontPrice": "12257.000",
            "Duration": 31536000
        }
    ]
}
```


- API 세부 정보는 AWS CLI 명령 DescribeHostReservationOfferings참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 지정된 필터 '인스턴스 패밀리'에 대해 구매할 수 있는 전용 호스트 예약을 설명합니다. 여기서 PaymentOption " NoUpfront

```
Get-EC2HostReservationOffering -Filter @{Name="instance-family";Values="m4"} |
 Where-Object PaymentOption -eq NoUpfront
```

출력:

```
CurrencyCode   :
Duration       : 94608000
HourlyPrice    : 1.307
InstanceFamily : m4
OfferingId     : hro-0c1f234567890d9ab
PaymentOption  : NoUpfront
UpfrontPrice   : 0.000

CurrencyCode   :
Duration       : 31536000
HourlyPrice    : 1.830
InstanceFamily : m4
OfferingId     : hro-04ad12aaaf34b5a67
PaymentOption  : NoUpfront
UpfrontPrice   : 0.000
```

- API 세부 정보는 Cmdlet 참조를 참조하십시오 DescribeHostReservationOfferings.AWS Tools for PowerShell

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeHosts** CLI와 함께 사용

다음 코드 예제는 DescribeHosts의 사용 방법을 보여줍니다.

CLI

AWS CLI

전용 호스트에 대한 세부 정보를 보려면

다음 describe-hosts 예시는 AWS 계정의 available 전용 호스트에 대한 세부 정보를 표시합니다.

```
aws ec2 describe-hosts --filter "Name=state,Values=available"
```

출력:

```
{
    "Hosts": [
        {
            "HostId": "h-07879acf49EXAMPLE",
            "Tags": [
                {
                    "Value": "production",
                    "Key": "purpose"
                }
            ],
            "HostProperties": {
                "Cores": 48,
                "TotalVCpus": 96,
                "InstanceType": "m5.large",
                "Sockets": 2
            },
            "Instances": [],
            "State": "available",
            "AvailabilityZone": "eu-west-1a",
            "AvailableCapacity": {
                "AvailableInstanceCapacity": [
                    {
                        "AvailableCapacity": 48,
                        "InstanceType": "m5.large",
                        "TotalCapacity": 48
                    }
                ],
                "AvailableVCpus": 96
            },
            "HostRecovery": "on",
            "AllocationTime": "2019-08-19T08:57:44.000Z",
            "AutoPlacement": "off"
        }
    ]
}
```

자세한 내용은 Linux 인스턴스용 Amazon Elastic Compute 클라우드 사용 설명서의 전용 호스트 보기를 참조하십시오.

- API 세부 정보는 AWS CLI 명령 DescribeHosts참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 EC2 호스트 세부 정보를 반환합니다.

```
Get-EC2Host
```

출력:

```
AllocationTime    : 3/23/2019 4:55:22 PM
AutoPlacement     : off
AvailabilityZone  : eu-west-1b
AvailableCapacity : Amazon.EC2.Model.AvailableCapacity
ClientToken       :
HostId            : h-01e23f4cd567890f1
HostProperties    : Amazon.EC2.Model.HostProperties
HostReservationId :
Instances         : {}
ReleaseTime       : 1/1/0001 12:00:00 AM
State             : available
Tags              : {}
```

예 2: 이 예에서는 호스트 AvailableInstanceCapacity h-01e23f4cd567899f1에 대해 쿼리합니다.

```
Get-EC2Host -HostId h-01e23f4cd567899f1 | Select-Object -ExpandProperty
 AvailableCapacity | Select-Object -expand AvailableInstanceCapacity
```

출력:

```
AvailableCapacity InstanceType TotalCapacity
----------------- ------------ -------------
11                m4.xlarge    11
```

- AWS Tools for PowerShell API에 DescribeHosts대한 자세한 내용은 Cmdlet 참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeIamInstanceProfileAssociations** CLI와 함께 사용

다음 코드 예제는 DescribeIamInstanceProfileAssociations의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 복원력이 뛰어난 서비스 구축 및 관리

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Get the instance profile association data for an instance.
    /// </summary>
    /// <param name="instanceId">The Id of the instance.</param>
    /// <returns>Instance profile associations data.</returns>
    public async Task<IamInstanceProfileAssociation> GetInstanceProfile(string
instanceId)
    {
        var response = await
_amazonEc2.DescribeIamInstanceProfileAssociationsAsync(
            new DescribeIamInstanceProfileAssociationsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("instance-id", new List<string>() { instanceId })
                },
```

```
            });
        return response.IamInstanceProfileAssociations[0];
    }
```

- API 세부 정보는 AWS SDK for .NET API DescribeIamInstanceProfileAssociations참조를 참조하십시오.

CLI

AWS CLI

IAM 인스턴스 프로파일 연결을 설명하는 방법

이 예제에서는 모든 IAM 인스턴스 프로파일 연결을 설명합니다.

명령:

```
aws ec2 describe-iam-instance-profile-associations
```

출력:

```
{
  "IamInstanceProfileAssociations": [
      {
          "InstanceId": "i-09eb09efa73ec1dee",
          "State": "associated",
          "AssociationId": "iip-assoc-0db249b1f25fa24b8",
          "IamInstanceProfile": {
              "Id": "AIPAJVQN4F5WVLGCJDRGM",
              "Arn": "arn:aws:iam::123456789012:instance-profile/admin-role"
          }
      },
      {
          "InstanceId": "i-0402909a2f4dffd14",
          "State": "associating",
          "AssociationId": "iip-assoc-0d1ec06278d29f44a",
          "IamInstanceProfile": {
              "Id": "AGJAJVQN4F5WVLGCJABCM",
              "Arn": "arn:aws:iam::123456789012:instance-profile/user1-role"
          }
      }
   ]
```

```
}
```

- API 세부 정보는 AWS CLI 명령 DescribeIamInstanceProfileAssociations참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
const ec2Client = new EC2Client({});
const { IamInstanceProfileAssociations } = await ec2Client.send(
  new DescribeIamInstanceProfileAssociationsCommand({
    Filters: [
      { Name: "instance-id", Values: [state.targetInstance.InstanceId] },
    ],
  }),
);
```

- API 세부 정보는 AWS SDK for JavaScript API DescribeIamInstanceProfileAssociations참조를 참조하십시오.

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class AutoScaler:
```

```
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def get_instance_profile(self, instance_id):
```

```
        """
        Gets data about the profile associated with an instance.

        :param instance_id: The ID of the instance to look up.
        :return: The profile data.
        """
        try:
            response =
self.ec2_client.describe_iam_instance_profile_associations(
                Filters=[{"Name": "instance-id", "Values": [instance_id]}]
            )
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't get instance profile association for instance
{instance_id}: {err}"
            )
        else:
            return response["IamInstanceProfileAssociations"][0]
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 DescribeIamInstanceProfileAssociations.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeIdFormat** CLI와 함께 사용

다음 코드 예제는 DescribeIdFormat의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 리소스의 ID 형식 설명하기

다음 describe-id-format 예에서는 보안 그룹의 ID 형식을 설명합니다.

```
aws ec2 describe-id-format \
    --resource security-group
```

다음 예제 출력에서 Deadline 값은 이 리소스 유형이 짧은 ID 형식에서 긴 ID 형식으로 영구적으로 전환하는 기한이 2018년 8월 15일 00:00 UTC에 만료되었음을 나타냅니다.

```
{
    "Statuses": [
        {
            "Deadline": "2018-08-15T00:00:00.000Z",
            "Resource": "security-group",
            "UseLongIds": true
        }
    ]
}
```

예 2: 모든 리소스의 ID 형식 설명하기

다음 describe-id-format 예에서는 모든 리소스 유형의 ID 형식을 설명합니다. 짧은 ID 형식을 지원하는 모든 리소스 유형이 긴 ID 형식을 사용하도록 전환되었습니다.

```
aws ec2 describe-id-format
```

- API 세부 정보는 AWS CLI 명령 DescribeIdFormat참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제에서는 지정된 리소스 유형의 ID 형식을 설명합니다.

```
Get-EC2IdFormat -Resource instance
```

출력:

```
Resource       UseLongIds
--------       ----------
instance       False
```

예 2: 이 예제에서는 더 긴 ID를 지원하는 모든 리소스 유형의 ID 형식을 설명합니다.

```
Get-EC2IdFormat
```

출력:

```
Resource       UseLongIds
--------       ----------
reservation    False
instance       False
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeIdFormat.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeIdentityIdFormat** CLI와 함께 사용

다음 코드 예제는 DescribeIdentityIdFormat의 사용 방법을 보여줍니다.

CLI

AWS CLI

IAM 역할의 ID 형식 설명하기

다음 describe-identity-id-format 예는 계정의 IAM EC2Role 역할로 생성한 인스턴스가 받는 ID 형식을 설명합니다. AWS

```
aws ec2 describe-identity-id-format \
    --principal-arn arn:aws:iam::123456789012:role/my-iam-role \
    --resource instance
```

다음 출력은 이 역할로 생성된 인스턴스가 긴 ID 형식의 ID를 받는다는 것을 나타냅니다.

```
{
    "Statuses": [
        {
            "Deadline": "2016-12-15T00:00:00Z",
            "Resource": "instance",
            "UseLongIds": true
        }
    ]
}
```

IAM 사용자의 ID 형식 설명

다음 `describe-identity-id-format` 예는 계정에서 IAM 사용자가 `AdminUser` 생성한 스냅샷이 받는 ID 형식을 설명합니다. AWS

```
aws ec2 describe-identity-id-format \
    --principal-arn arn:aws:iam::123456789012:user/AdminUser \
    --resource snapshot
```

출력은 이 사용자가 생성한 스냅샷이 긴 ID 형식의 ID를 받는다는 것을 나타냅니다.

```
{
    "Statuses": [
        {
            "Deadline": "2016-12-15T00:00:00Z",
            "Resource": "snapshot",
            "UseLongIds": true
        }
    ]
}
```

- API 세부 정보는 AWS CLI 명령 DescribeIdentityIdFormat참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 주어진 역할에 대한 리소스 '이미지'의 ID 형식을 반환합니다.

```
Get-EC2IdentityIdFormat -PrincipalArn arn:aws:iam::123456789511:role/JDBC -
Resource image
```

출력:

```
Deadline             Resource UseLongIds
--------             -------- ----------
8/2/2018 11:30:00 PM image    True
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeIdentityIdFormat.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeImageAttribute** CLI와 함께 사용

다음 코드 예제는 DescribeImageAttribute의 사용 방법을 보여줍니다.

CLI

AWS CLI

AMI의 시작 권한을 설명하려면

이 예제에서는 지정된 AMI의 시작 권한을 설명합니다.

명령:

```
aws ec2 describe-image-attribute --image-id ami-5731123e --attribute
 launchPermission
```

출력:

```
{
    "LaunchPermissions": [
        {
            "UserId": "123456789012"
        }
    ],
    "ImageId": "ami-5731123e",
}
```

AMI의 제품 코드를 설명하려면

이 예제에서는 지정된 AMI의 제품 코드를 설명합니다. 참고로 이 AMI에는 제품 코드가 없습니다.

명령:

```
aws ec2 describe-image-attribute --image-id ami-5731123e --attribute productCodes
```

출력:

```
{
    "ProductCodes": [],
    "ImageId": "ami-5731123e",
}
```

- API 세부 정보는 AWS CLI 명령 DescribeImageAttribute참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제에서는 지정된 AMI에 대한 설명을 가져옵니다.

```
Get-EC2ImageAttribute -ImageId ami-12345678 -Attribute description
```

출력:

```
BlockDeviceMappings : {}
Description         : My image description
ImageId             : ami-12345678
KernelId            :
LaunchPermissions   : {}
ProductCodes        : {}
RamdiskId           :
SriovNetSupport     :
```

예 2: 이 예제는 지정된 AMI에 대한 시작 권한을 가져옵니다.

```
Get-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission
```

출력:

```
BlockDeviceMappings : {}
Description         :
ImageId             : ami-12345678
KernelId            :
LaunchPermissions   : {all}
ProductCodes        : {}
RamdiskId           :
SriovNetSupport     :
```

예 3: 이 예시에서는 향상된 네트워킹이 활성화되었는지 여부를 테스트합니다.

```
Get-EC2ImageAttribute -ImageId ami-12345678 -Attribute sriovNetSupport
```

출력:

```
BlockDeviceMappings : {}
Description         :
ImageId             : ami-12345678
KernelId            :
LaunchPermissions   : {}
ProductCodes        : {}
RamdiskId           :
SriovNetSupport     : simple
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeImageAttribute.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeImages** CLI와 함께 사용

다음 코드 예제는 DescribeImages의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 인스턴스 시작하기

Bash

AWS CLI Bash 스크립트 사용

> **Note**
>
> 더 많은 정보가 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
##############################################################################
# function ec2_describe_images
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 images.
#
# Parameters:
#       -i image_ids - A space-separated  list of image IDs (optional).
#       -h - Display help.
#
# And:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function ec2_describe_images() {
  local image_ids response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_images"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 images."
    echo "  -i image_ids - A space-separated list of image IDs (optional)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) image_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
```

```
  export OPTIND=1

  local aws_cli_args=()

  if [[ -n "$image_ids" ]]; then
    # shellcheck disable=SC2206
    aws_cli_args+=("--image-ids" $image_ids)
  fi

  response=$(aws ec2 describe-images \
    "${aws_cli_args[@]}" \
    --query 'Images[*].[Description,Architecture,ImageId]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports describe-images operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}
```

이 예제에 사용된 유틸리티 함수

```
########################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
########################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

########################################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
```

```
#  Returns:
#          0: - Success.
#
##############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- API 세부 정보는 AWS CLI 명령 DescribeImages참조를 참조하십시오.

CLI

AWS CLI

예제 1: AMI를 설명하는 방법

다음 `describe-images` 예제에서는 지정된 리전에서 지정된 AMI를 설명합니다.

```
aws ec2 describe-images \
    --region us-east-1 \
    --image-ids ami-1234567890EXAMPLE
```

출력:

```
{
    "Images": [
        {
            "VirtualizationType": "hvm",
            "Description": "Provided by Red Hat, Inc.",
            "PlatformDetails": "Red Hat Enterprise Linux",
            "EnaSupport": true,
            "Hypervisor": "xen",
            "State": "available",
            "SriovNetSupport": "simple",
            "ImageId": "ami-1234567890EXAMPLE",
            "UsageOperation": "RunInstances:0010",
            "BlockDeviceMappings": [
                {
                    "DeviceName": "/dev/sda1",
                    "Ebs": {
                        "SnapshotId": "snap-111222333444aaabb",
                        "DeleteOnTermination": true,
                        "VolumeType": "gp2",
                        "VolumeSize": 10,
                        "Encrypted": false
                    }
                }
            ],
            "Architecture": "x86_64",
            "ImageLocation": "123456789012/RHEL-8.0.0_HVM-20190618-x86_64-1-
Hourly2-GP2",
            "RootDeviceType": "ebs",
            "OwnerId": "123456789012",
            "RootDeviceName": "/dev/sda1",
            "CreationDate": "2019-05-10T13:17:12.000Z",
            "Public": true,
            "ImageType": "machine",
            "Name": "RHEL-8.0.0_HVM-20190618-x86_64-1-Hourly2-GP2"
        }
    ]
}
```

자세한 내용은 Amazon EC2 사용 설명서에서 Amazon Machine Image(AMI)를 참조하세요.

예제 2: 필터를 기반으로 AMI를 설명하는 방법

다음 `describe-images` 예제에서는 Amazon에서 제공하는 Amazon EBS 지원 Windows AMI를 설명합니다.

```
aws ec2 describe-images \
    --owners amazon \
    --filters "Name=platform,Values=windows" "Name=root-device-type,Values=ebs"
```

`describe-images` 출력 예제는 예제 1을 참조하세요.

필터를 사용하는 추가 예제는 Amazon EC2 사용 설명서에서 리소스 나열 및 필터링을 참조하세요.

예제 3: 태그를 기반으로 AMI를 설명하는 방법

다음 `describe-images` 예제에서는 `Type=Custom` 태그가 있는 모든 AMI를 설명합니다. 이 예제에서는 `--query` 파라미터를 사용하여 AMI ID만 표시합니다.

```
aws ec2 describe-images \
    --filters "Name=tag:Type,Values=Custom" \
    --query 'Images[*].[ImageId]' \
    --output text
```

출력:

```
ami-1234567890EXAMPLE
ami-0abcdef1234567890
```

태그 필터를 사용하는 추가 예제는 Amazon EC2 사용 설명서에서 태그 작업을 참조하세요.

- API에 대한 자세한 내용은 AWS CLI 명령 참조를 참조하십시오 DescribeImages.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { paginateDescribeImages } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

// List at least the first i386 image available for EC2 instances.
export const main = async () => {
  // The paginate function is a wrapper around the base command.
  const paginator = paginateDescribeImages(
    // Without limiting the page size, this call can take a long time. pageSize
 is just sugar for
    // the MaxResults property in the base command.
    { client, pageSize: 25 },
    {
      // There are almost 70,000 images available. Be specific with your
 filtering
      // to increase efficiency.
      // See https://docs.aws.amazon.com/AWSJavaScriptSDK/v3/latest/clients/
client-ec2/interfaces/describeimagescommandinput.html#filters
      Filters: [{ Name: "architecture", Values: ["x86_64"] }],
    },
  );

  try {
    const arm64Images = [];
    for await (const page of paginator) {
      if (page.Images.length) {
        arm64Images.push(...page.Images);
        // Once we have at least 1 result, we can stop.
        if (arm64Images.length >= 1) {
          break;
        }
      }
    }
    console.log(arm64Images);
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API DescribeImages참조를 참조하십시오.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예제에서는 지정된 AMI를 설명합니다.

```
Get-EC2Image -ImageId ami-12345678
```

출력:

```
Architecture        : x86_64
BlockDeviceMappings : {/dev/xvda}
CreationDate        : 2014-10-20T00:56:28.000Z
Description         : My image
Hypervisor          : xen
ImageId             : ami-12345678
ImageLocation       : 123456789012/my-image
ImageOwnerAlias     :
ImageType           : machine
KernelId            :
Name                : my-image
OwnerId             : 123456789012
Platform            :
ProductCodes        : {}
Public              : False
RamdiskId           :
RootDeviceName      : /dev/xvda
RootDeviceType      : ebs
SriovNetSupport     : simple
State               : available
StateReason         :
Tags                : {Name}
VirtualizationType  : hvm
```

예 2: 이 예제는 소유하고 있는 AMI를 설명합니다.

```
Get-EC2Image -owner self
```

예 3: 이 예에서는 Microsoft Windows Server를 실행하는 퍼블릭 AMI를 설명합니다.

```
Get-EC2Image -Filter @{ Name="platform"; Values="windows" }
```

예 4: 이 예에서는 'us-west-2' 지역의 모든 퍼블릭 AMI를 설명합니다.

```
Get-EC2Image -Region us-west-2
```

- API 세부 정보는 Cmdlet 참조를 참조하십시오 DescribeImages.AWS Tools for PowerShell

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
 actions."""

    def __init__(self, ec2_resource, instance=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param instance: A Boto3 Instance object. This is a high-level object
 that
                          wraps instance actions.
        """
        self.ec2_resource = ec2_resource
        self.instance = instance

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def get_images(self, image_ids):
        """
```

```
        Gets information about Amazon Machine Images (AMIs) from a list of AMI
 IDs.

        :param image_ids: The list of AMIs to look up.
        :return: A list of Boto3 Image objects that represent the requested AMIs.
        """
        try:
            images = list(self.ec2_resource.images.filter(ImageIds=image_ids))
        except ClientError as err:
            logger.error(
                "Couldn't get images. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return images
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 DescribeImages.

Rust

SDK for Rust

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
// A simple Rust program to list all Amazon images in the currently configured
 region.
#[tokio::main]
async fn main() -> Result<(), Error> {
    let config = aws_config::load_from_env().await;
    let client = Client::new(&config);

    let resp = client.describe_images().owners("amazon").send().await?;
```

```
    println!("AWS SDK for Rust v{}", PKG_VERSION);
    println!("Describing Amazon Machine Images (AMIs):");

    let mut images: Vec<_> = resp
        .images()
        .iter()
        .filter(|i| {
            i.description()
                .filter(|i| i.contains("Amazon Linux AMI 2023"))
                .is_some()
        })
        .collect();
    images.sort_by(|a, b| a.description.cmp(&b.description));

    if images.is_empty() {
        println!("No images found.");
        return Ok(());
    }

    for image in images {
        let id = image.image_id().unwrap_or_default();
        let description = image.description().unwrap_or_default();

        println!("{id}: {description}");
    }

    Ok(())
}
```

- API에 대한 자세한 내용은 Rust용AWS SDK API 레퍼런스를 참조하십시오 DescribeImages.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeImportImageTasks** CLI와 함께 사용

다음 코드 예제는 DescribeImportImageTasks의 사용 방법을 보여줍니다.

CLI

AWS CLI

이미지 가져오기 작업을 모니터링하려면

다음 `describe-import-image-tasks` 예제에서는 지정된 이미지 가져오기 작업의 상태를 확인합니다.

```
aws ec2 describe-import-image-tasks \
    --import-task-ids import-ami-1234567890abcdef0
```

진행 중인 이미지 가져오기 작업의 출력입니다.

```
{
    "ImportImageTasks": [
        {
            "ImportTaskId": "import-ami-1234567890abcdef0",
            "Progress": "28",
            "SnapshotDetails": [
                {
                    "DiskImageSize": 705638400.0,
                    "Format": "ova",
                    "Status": "completed",
                    "UserBucket": {
                        "S3Bucket": "my-import-bucket",
                        "S3Key": "vms/my-server-vm.ova"
                    }
                }
            ],
            "Status": "active",
            "StatusMessage": "converting"
        }
    ]
}
```

완료된 이미지 가져오기 작업의 출력입니다. 결과 AMI의 ID는 에서 제공합니다`ImageId`.

```
{
    "ImportImageTasks": [
        {
            "ImportTaskId": "import-ami-1234567890abcdef0",
```

```
            "ImageId": "ami-1234567890abcdef0",
            "SnapshotDetails": [
                {
                    "DiskImageSize": 705638400.0,
                    "Format": "ova",
                    "SnapshotId": "snap-1234567890abcdef0"
                    "Status": "completed",
                    "UserBucket": {
                        "S3Bucket": "my-import-bucket",
                        "S3Key": "vms/my-server-vm.ova"
                    }
                }
            ],
            "Status": "completed"
        }
    ]
}
```

- API 세부 정보는 AWS CLI 명령 DescribeImportImageTasks참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제에서는 지정된 이미지 가져오기 작업을 설명합니다.

```
Get-EC2ImportImageTask -ImportTaskId import-ami-hgfedcba
```

출력:

```
Architecture    : x86_64
Description     : Windows Image 2
Hypervisor      :
ImageId         : ami-1a2b3c4d
ImportTaskId    : import-ami-hgfedcba
LicenseType     : AWS
Platform        : Windows
Progress        :
SnapshotDetails : {/dev/sda1}
Status          : completed
StatusMessage   :
```

예 2: 이 예제에서는 모든 이미지 가져오기 작업을 설명합니다.

```
Get-EC2ImportImageTask
```

출력:

```
Architecture    :
Description     : Windows Image 1
Hypervisor      :
ImageId         :
ImportTaskId    : import-ami-abcdefgh
LicenseType     : AWS
Platform        : Windows
Progress        :
SnapshotDetails : {}
Status          : deleted
StatusMessage   : User initiated task cancelation

Architecture    : x86_64
Description     : Windows Image 2
Hypervisor      :
ImageId         : ami-1a2b3c4d
ImportTaskId    : import-ami-hgfedcba
LicenseType     : AWS
Platform        : Windows
Progress        :
SnapshotDetails : {/dev/sda1}
Status          : completed
StatusMessage   :
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeImportImageTasks.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeImportSnapshotTasks** CLI와 함께 사용

다음 코드 예제는 DescribeImportSnapshotTasks의 사용 방법을 보여줍니다.

CLI

AWS CLI

스냅샷 가져오기 작업을 모니터링하려면

다음 `describe-import-snapshot-tasks` 예제에서는 지정된 스냅샷 가져오기 작업의 상태를 확인합니다.

```
aws ec2 describe-import-snapshot-tasks \
    --import-task-ids import-snap-1234567890abcdef0
```

진행 중인 스냅샷 가져오기 작업의 출력:

```
{
    "ImportSnapshotTasks": [
        {
            "Description": "My server VMDK",
            "ImportTaskId": "import-snap-1234567890abcdef0",
            "SnapshotTaskDetail": {
                "Description": "My server VMDK",
                "DiskImageSize": "705638400.0",
                "Format": "VMDK",
                "Progress": "42",
                "Status": "active",
                "StatusMessage": "downloading/converting",
                "UserBucket": {
                    "S3Bucket": "my-import-bucket",
                    "S3Key": "vms/my-server-vm.vmdk"
                }
            }
        }
    ]
}
```

완료된 스냅샷 가져오기 작업의 출력입니다. 결과 스냅샷의 ID는 에서 제공합니다`SnapshotId`.

```
{
    "ImportSnapshotTasks": [
        {
            "Description": "My server VMDK",
```

```
            "ImportTaskId": "import-snap-1234567890abcdef0",
            "SnapshotTaskDetail": {
                "Description": "My server VMDK",
                "DiskImageSize": "705638400.0",
                "Format": "VMDK",
                "SnapshotId": "snap-1234567890abcdef0"
                "Status": "completed",
                "UserBucket": {
                    "S3Bucket": "my-import-bucket",
                    "S3Key": "vms/my-server-vm.vmdk"
                }
            }
        }
    ]
}
```

- API 세부 정보는 AWS CLI 명령 DescribeImportSnapshotTasks참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제에서는 지정된 스냅샷 가져오기 작업을 설명합니다.

```
Get-EC2ImportSnapshotTask -ImportTaskId import-snap-abcdefgh
```

출력:

```
Description             ImportTaskId               SnapshotTaskDetail

-----------------       --------------------       ------------------

Disk Image Import 1     import-snap-abcdefgh
 Amazon.EC2.Model.SnapshotTaskDetail
```

예 2: 이 예제에서는 모든 스냅샷 가져오기 작업을 설명합니다.

```
Get-EC2ImportSnapshotTask
```

출력:

```
Description             ImportTaskId               SnapshotTaskDetail

-----------------       --------------------       ------------------

Disk Image Import 1     import-snap-abcdefgh
 Amazon.EC2.Model.SnapshotTaskDetail
Disk Image Import 2     import-snap-hgfedcba
 Amazon.EC2.Model.SnapshotTaskDetail
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeImportSnapshotTasks.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeInstanceAttribute** CLI와 함께 사용

다음 코드 예제는 DescribeInstanceAttribute의 사용 방법을 보여줍니다.

CLI

AWS CLI

인스턴스 유형을 설명하려면

이 예제에서는 지정된 인스턴스의 인스턴스 유형을 설명합니다.

명령:

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --attribute
 instanceType
```

출력:

```
{
    "InstanceId": "i-1234567890abcdef0"
    "InstanceType": {
        "Value": "t1.micro"
    }
```

```
}
```

disableApiTermination 속성을 설명하려면

이 예제에서는 지정된 인스턴스의 `disableApiTermination` 속성을 설명합니다.

명령:

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --attribute
 disableApiTermination
```

출력:

```
{
"InstanceId": "i-1234567890abcdef0"
    "DisableApiTermination": {
        "Value": "false"
    }
}
```

인스턴스의 블록 디바이스 매핑을 설명하려면

이 예제에서는 지정된 인스턴스의 `blockDeviceMapping` 속성을 설명합니다.

명령:

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --attribute
 blockDeviceMapping
```

출력:

```
{
    "InstanceId": "i-1234567890abcdef0"
    "BlockDeviceMappings": [
        {
            "DeviceName": "/dev/sda1",
            "Ebs": {
                "Status": "attached",
                "DeleteOnTermination": true,
                "VolumeId": "vol-049df61146c4d7901",
```

```
                "AttachTime": "2013-05-17T22:42:34.000Z"
            }
        },
        {
            "DeviceName": "/dev/sdf",
            "Ebs": {
                "Status": "attached",
                "DeleteOnTermination": false,
                "VolumeId": "vol-049df61146c4d7901",
                "AttachTime": "2013-09-10T23:07:00.000Z"
            }
        }
    ],
}
```

- API 세부 정보는 AWS CLI 명령 DescribeInstanceAttribute참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 지정된 인스턴스의 인스턴스 유형을 설명합니다.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute instanceType
```

출력:

```
InstanceType                    : t2.micro
```

예 2: 이 예제에서는 지정된 인스턴스에 향상된 네트워킹이 활성화되어 있는지 여부를 설명합니다.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute sriovNetSupport
```

출력:

```
SriovNetSupport                 : simple
```

예 3: 이 예에서는 지정된 인스턴스의 보안 그룹을 설명합니다.

```
(Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute groupSet).Groups
```

출력:

```
GroupId
-------
sg-12345678
sg-45678901
```

예 4: 이 예제에서는 지정된 인스턴스에 대해 EBS 최적화가 활성화되었는지 여부를 설명합니다.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute ebsOptimized
```

출력:

```
EbsOptimized                    : False
```

예 5: 이 예제에서는 지정된 인스턴스의 disableApiTermination " 속성을 설명합니다.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute disableApiTermination
```

출력:

```
DisableApiTermination           : False
```

예 6: 이 예제에서는 지정된 인스턴스의 'instanceInitiatedShutdownBehavior' 속성을 설명합니다.

```
Get-EC2InstanceAttribute -InstanceId i-12345678 -Attribute
 instanceInitiatedShutdownBehavior
```

출력:

```
InstanceInitiatedShutdownBehavior : stop
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeInstanceAttribute.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeInstanceStatus** CLI와 함께 사용

다음 코드 예제는 DescribeInstanceStatus의 사용 방법을 보여줍니다.

CLI

AWS CLI

인스턴스 상태를 설명하는 방법

다음 describe-instance-status 예제에서는 지정된 인스턴스의 현재 상태를 설명합니다.

```
aws ec2 describe-instance-status \
    --instance-ids i-1234567890abcdef0
```

출력:

```
{
    "InstanceStatuses": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "InstanceState": {
                "Code": 16,
                "Name": "running"
            },
            "AvailabilityZone": "us-east-1d",
            "SystemStatus": {
                "Status": "ok",
                "Details": [
                    {
                        "Status": "passed",
                        "Name": "reachability"
                    }
                ]
            },
            "InstanceStatus": {
                "Status": "ok",
                "Details": [
                    {
```

```
                        "Status": "passed",
                        "Name": "reachability"
                    }
                ]
            }
        }
    ]
}
```

자세한 내용은 Amazon EC2 사용 설명서에서 인스턴스 상태 모니터링을 참조하세요.

- API 세부 정보는 AWS CLI 명령 DescribeInstanceStatus참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 지정된 인스턴스의 상태를 설명합니다.

```
Get-EC2InstanceStatus -InstanceId i-12345678
```

출력:

```
AvailabilityZone : us-west-2a
Events           : {}
InstanceId       : i-12345678
InstanceState    : Amazon.EC2.Model.InstanceState
Status           : Amazon.EC2.Model.InstanceStatusSummary
SystemStatus     : Amazon.EC2.Model.InstanceStatusSummary
```

```
$status = Get-EC2InstanceStatus -InstanceId i-12345678
$status.InstanceState
```

출력:

```
Code    Name
----    ----
16      running
```

```
$status.Status
```

출력:

```
Details           Status
-------           ------
{reachability}    ok
```

```
$status.SystemStatus
```

출력:

```
Details           Status
-------           ------
{reachability}    ok
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeInstanceStatus.

Rust

SDK for Rust

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
async fn show_all_events(client: &Client) -> Result<(), Error> {
    let resp = client.describe_regions().send().await.unwrap();

    for region in resp.regions.unwrap_or_default() {
        let reg: &'static str =
 Box::leak(Box::from(region.region_name().unwrap()));
        let region_provider =
 RegionProviderChain::default_provider().or_else(reg);
        let config = aws_config::from_env().region(region_provider).load().await;
        let new_client = Client::new(&config);

        let resp = new_client.describe_instance_status().send().await;
```

```
        println!("Instances in region {}:", reg);
        println!();

        for status in resp.unwrap().instance_statuses() {
            println!(
                "  Events scheduled for instance ID: {}",
                status.instance_id().unwrap_or_default()
            );
            for event in status.events() {
                println!("    Event ID:     {}",
 event.instance_event_id().unwrap());
                println!("    Description:  {}", event.description().unwrap());
                println!("    Event code:   {}", event.code().unwrap().as_ref());
                println!();
            }
        }
    }

    Ok(())
}
```

- API에 대한 자세한 내용은 Rust용AWS SDK API 레퍼런스를 참조하십시오 DescribeInstanceStatus.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeInstanceTypes** CLI와 함께 사용

다음 코드 예제는 DescribeInstanceTypes의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 인스턴스 시작하기

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Describe the instance types available.
    /// </summary>
    /// <returns>A list of instance type information.</returns>
    public async Task<List<InstanceTypeInfo>>
DescribeInstanceTypes(ArchitectureValues architecture)
    {
        var request = new DescribeInstanceTypesRequest();

        var filters = new List<Filter>
            { new Filter("processor-info.supported-architecture", new
List<string> { architecture.ToString() }) };
        filters.Add(new Filter("instance-type", new() { "*.micro", "*.small" }));

        request.Filters = filters;
        var instanceTypes = new List<InstanceTypeInfo>();

        var paginator = _amazonEC2.Paginators.DescribeInstanceTypes(request);
        await foreach (var instanceType in paginator.InstanceTypes)
        {
            instanceTypes.Add(instanceType);
        }
        return instanceTypes;
    }
```

- API 세부 정보는 AWS SDK for .NET API DescribeInstanceTypes참조를 참조하십시오.

Bash

AWS CLI Bash 스크립트 사용

> **Note**
>
> 더 많은 정보가 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
##############################################################################
# ec2_describe_instance_types
#
# This function describes EC2 instance types filtered by processor architecture
# and optionally by instance type. It takes the following arguments:
#
# -a, --architecture ARCHITECTURE  Specify the processor architecture (e.g.,
 x86_64)
# -t, --type INSTANCE_TYPE         Comma-separated list of instance types (e.g.,
 t2.micro)
# -h, --help                       Show the usage help
#
# The function prints the instance type and supported architecture for each
# matching instance type.
##############################################################################
function ec2_describe_instance_types() {
  local architecture=""
  local instance_types=""

  # bashsupport disable=BP5008
  function usage() {
    echo "Usage: ec2_describe_instance_types [-a|--architecture ARCHITECTURE] [-
t|--type INSTANCE_TYPE] [-h|--help]"
    echo "  -a, --architecture ARCHITECTURE  Specify the processor architecture
 (e.g., x86_64)"
    echo "  -t, --type INSTANCE_TYPE         Comma-separated list of instance
 types (e.g., t2.micro)"
    echo "  -h, --help                       Show this help message"
  }

  while [[ $# -gt 0 ]]; do
    case "$1" in
```

```
      -a | --architecture)
        architecture="$2"
        shift 2
        ;;
      -t | --type)
        instance_types="$2"
        shift 2
        ;;
      -h | --help)
        usage
        return 0
        ;;
      *)
        echo "Unknown argument: $1"
        return 1
        ;;
    esac
  done

  if [[ -z "$architecture" ]]; then
    errecho "Error: Architecture not specified."
    usage
    return 1
  fi

  if [[ -z "$instance_types" ]]; then
    errecho "Error: Instance type not specified."
    usage
    return 1
  fi

  local tmp_json_file="temp_ec2.json"
  echo -n '[
    {
      "Name": "processor-info.supported-architecture",
      "Values": [' >"$tmp_json_file"

  local items
  IFS=',' read -ra items <<<"$architecture"
  local array_size
  array_size=${#items[@]}
  for i in $(seq 0 $((array_size - 1))); do
    echo -n '"'"${items[$i]}"'"' >>"$tmp_json_file"
    if [[ $i -lt $((array_size - 1)) ]]; then
```

```
      echo -n ',' >>"$tmp_json_file"
    fi
  done
  echo -n ']},
    {
    "Name": "instance-type",
      "Values": [' >>"$tmp_json_file"
  IFS=',' read -ra items <<<"$instance_types"
  local array_size
  array_size=${#items[@]}
  for i in $(seq 0 $((array_size - 1))); do
    echo -n '"'"${items[$i]}"'"' >>"$tmp_json_file"
    if [[ $i -lt $((array_size - 1)) ]]; then
      echo -n ',' >>"$tmp_json_file"
    fi
  done

  echo -n ']}]' >>"$tmp_json_file"

  local response
  response=$(aws ec2 describe-instance-types --filters file://"$tmp_json_file" \
    --query 'InstanceTypes[*].[InstanceType]' --output text)

  local error_code=$?

  rm "$tmp_json_file"

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    echo "ERROR: AWS reports describe-instance-types operation failed."
    return 1
  fi

  echo "$response"
  return 0
}
```

이 예제에 사용된 유틸리티 함수

```
########################################################################################
# function errecho
#
```

```
# This function outputs everything sent to it to STDERR (standard error output).
#########################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#########################################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
#########################################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- API 세부 정보는 AWS CLI 명령 DescribeInstanceTypes참조를 참조하십시오.

CLI

AWS CLI

예제 1: 인스턴스 유형을 설명하는 방법

다음 `describe-instance-types` 예제에서는 지정된 인스턴스 유형의 세부 정보를 표시합니다.

```
aws ec2 describe-instance-types \
    --instance-types t2.micro
```

출력:

```
{
    "InstanceTypes": [
        {
            "InstanceType": "t2.micro",
            "CurrentGeneration": true,
            "FreeTierEligible": true,
            "SupportedUsageClasses": [
                "on-demand",
                "spot"
            ],
            "SupportedRootDeviceTypes": [
                "ebs"
            ],
            "BareMetal": false,
            "Hypervisor": "xen",
            "ProcessorInfo": {
                "SupportedArchitectures": [
                    "i386",
                    "x86_64"
                ],
                "SustainedClockSpeedInGhz": 2.5
            },
            "VCpuInfo": {
                "DefaultVCpus": 1,
                "DefaultCores": 1,
                "DefaultThreadsPerCore": 1,
                "ValidCores": [
                    1
                ],
```

```
                "ValidThreadsPerCore": [
                    1
                ]
            },
            "MemoryInfo": {
                "SizeInMiB": 1024
            },
            "InstanceStorageSupported": false,
            "EbsInfo": {
                "EbsOptimizedSupport": "unsupported",
                "EncryptionSupport": "supported"
            },
            "NetworkInfo": {
                "NetworkPerformance": "Low to Moderate",
                "MaximumNetworkInterfaces": 2,
                "Ipv4AddressesPerInterface": 2,
                "Ipv6AddressesPerInterface": 2,
                "Ipv6Supported": true,
                "EnaSupport": "unsupported"
            },
            "PlacementGroupInfo": {
                "SupportedStrategies": [
                    "partition",
                    "spread"
                ]
            },
            "HibernationSupported": false,
            "BurstablePerformanceSupported": true,
            "DedicatedHostsSupported": false,
            "AutoRecoverySupported": true
        }
    ]
}
```


자세한 내용은 Linux 인스턴스용 Amazon Elastic Compute 클라우드 사용 설명서의 인스턴스 유형을 참조하십시오.

예제 2: 사용 가능한 인스턴스 유형을 필터링하는 방법

필터를 지정하여 결과 범위를 특정 특성의 인스턴스 유형으로 지정할 수 있습니다. 다음 `describe-instance-types` 예제에서는 최대 절전 모드를 지원하는 인스턴스 유형을 나열합니다.

```
aws ec2 describe-instance-types \
    --filters Name=hibernation-supported,Values=true --query
 'InstanceTypes[*].InstanceType'
```

출력:

```
[
    "m5.8xlarge",
    "r3.large",
    "c3.8xlarge",
    "r5.large",
    "m4.4xlarge",
    "c4.large",
    "m5.xlarge",
    "m4.xlarge",
    "c3.large",
    "c4.8xlarge",
    "c4.4xlarge",
    "c5.xlarge",
    "c5.12xlarge",
    "r5.4xlarge",
    "c5.4xlarge"
]
```

자세한 내용은 Linux 인스턴스용 Amazon Elastic Compute 클라우드 사용 설명서의 인스턴스 유형을 참조하십시오.

- API 세부 정보는 AWS CLI 명령 DescribeInstanceTypes참조를 참조하십시오.

Java

SDK for Java 2.x

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    // Get a list of instance types.
    public static String getInstanceTypes(Ec2Client ec2) {
```

```
        String instanceType;
        try {
            DescribeInstanceTypesRequest typesRequest =
DescribeInstanceTypesRequest.builder()
                .maxResults(10)
                .build();

            DescribeInstanceTypesResponse response =
ec2.describeInstanceTypes(typesRequest);
            List<InstanceTypeInfo> instanceTypes = response.instanceTypes();
            for (InstanceTypeInfo type : instanceTypes) {
                System.out.println("The memory information of this type is " +
type.memoryInfo().sizeInMiB());
                System.out.println("Network information is " +
type.networkInfo().toString());
                System.out.println("Instance type is " +
type.instanceType().toString());
                instanceType = type.instanceType().toString();
                if (instanceType.compareTo("t2.2xlarge")  == 0){
                    return instanceType;
                }
            }

        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
        return "";
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API DescribeInstanceTypes참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import {
  paginateDescribeInstanceTypes,
  DescribeInstanceTypesCommand,
} from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

// List at least the first arm64 EC2 instance type available.
export const main = async () => {
  // The paginate function is a wrapper around the underlying command.
  const paginator = paginateDescribeInstanceTypes(
    // Without limiting the page size, this call can take a long time. pageSize
 is just sugar for
    // the MaxResults property in the underlying command.
    { client, pageSize: 25 },
    {
      Filters: [
        { Name: "processor-info.supported-architecture", Values: ["x86_64"] },
        { Name: "free-tier-eligible", Values: ["true"] },
      ],
    }
  );

  try {
    const instanceTypes = [];

    for await (const page of paginator) {
      if (page.InstanceTypes.length) {
        instanceTypes.push(...page.InstanceTypes);

        // When we have at least 1 result, we can stop.
        if (instanceTypes.length >= 1) {
          break;
        }
      }
    }
    console.log(instanceTypes);
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API DescribeInstanceTypes참조를 참조하십시오.

Kotlin

SDK for Kotlin

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
// Get a list of instance types.
suspend fun getInstanceTypesSc(): String {
    var instanceType = ""
    val filterObs = ArrayList<Filter>()
    val filter =
        Filter {
            name = "processor-info.supported-architecture"
            values = listOf("arm64")
        }

    filterObs.add(filter)
    val typesRequest =
        DescribeInstanceTypesRequest {
            filters = filterObs
            maxResults = 10
        }
    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.describeInstanceTypes(typesRequest)
        response.instanceTypes?.forEach { type ->
            println("The memory information of this type is
 ${type.memoryInfo?.sizeInMib}")
            println("Maximum number of network cards is
 ${type.networkInfo?.maximumNetworkCards}")
            instanceType = type.instanceType.toString()
        }
        return instanceType
    }
}
```

- API 세부 정보는 Kotlin API용AWS SDK 레퍼런스를 참조하세요 DescribeInstanceTypes.

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 여기에서 확인할 수 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
 actions."""

    def __init__(self, ec2_resource, instance=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param instance: A Boto3 Instance object. This is a high-level object
 that
                          wraps instance actions.
        """
        self.ec2_resource = ec2_resource
        self.instance = instance

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def get_instance_types(self, architecture):
        """
        Gets instance types that support the specified architecture and are
 designated
        as either 'micro' or 'small'. When an instance is created, the instance
 type
        you specify must support the architecture of the AMI you use.
```

```
        :param architecture: The kind of architecture the instance types must
support,
                             such as 'x86_64'.
        :return: A list of instance types that support the specified architecture
                 and are either 'micro' or 'small'.
        """
        try:
            inst_types = []
            it_paginator = self.ec2_resource.meta.client.get_paginator(
                "describe_instance_types"
            )
            for page in it_paginator.paginate(
                Filters=[
                    {
                        "Name": "processor-info.supported-architecture",
                        "Values": [architecture],
                    },
                    {"Name": "instance-type", "Values": ["*.micro", "*.small"]},
                ]
            ):
                inst_types += page["InstanceTypes"]
        except ClientError as err:
            logger.error(
                "Couldn't get instance types. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return inst_types
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 DescribeInstanceTypes.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeInstances** CLI와 함께 사용

다음 코드 예제는 DescribeInstances의 사용 방법을 보여줍니다.

작업 예시는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 복원력이 뛰어난 서비스 구축 및 관리
- 인스턴스 시작하기

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
/// <summary>
/// Get information about existing EC2 images.
/// </summary>
/// <returns>Async task.</returns>
public async Task DescribeInstances()
{
    // List all EC2 instances.
    await GetInstanceDescriptions();

    string tagName = "IncludeInList";
    string tagValue = "Yes";
    await GetInstanceDescriptionsFiltered(tagName, tagValue);
}

/// <summary>
/// Get information for all existing Amazon EC2 instances.
/// </summary>
/// <returns>Async task.</returns>
public async Task GetInstanceDescriptions()
{
    Console.WriteLine("Showing all instances:");
```

```
        var paginator = _amazonEC2.Paginators.DescribeInstances(new
DescribeInstancesRequest());

        await foreach (var response in paginator.Responses)
        {
            foreach (var reservation in response.Reservations)
            {
                foreach (var instance in reservation.Instances)
                {
                    Console.Write($"Instance ID: {instance.InstanceId}");
                    Console.WriteLine($"\tCurrent State: {instance.State.Name}");
                }
            }
        }
    }

    /// <summary>
    /// Get information about EC2 instances filtered by a tag name and value.
    /// </summary>
    /// <param name="tagName">The name of the tag to filter on.</param>
    /// <param name="tagValue">The value of the tag to look for.</param>
    /// <returns>Async task.</returns>
    public async Task GetInstanceDescriptionsFiltered(string tagName, string
tagValue)
    {
        // This tag filters the results of the instance list.
        var filters = new List<Filter>
        {
            new Filter
            {
                Name = $"tag:{tagName}",
                Values = new List<string>
                {
                    tagValue,
                },
            },
        };
        var request = new DescribeInstancesRequest
        {
            Filters = filters,
        };

        Console.WriteLine("\nShowing instances with tag: \"IncludeInList\" set to
\"Yes\".");
```

```
        var paginator = _amazonEC2.Paginators.DescribeInstances(request);

        await foreach (var response in paginator.Responses)
        {
            foreach (var reservation in response.Reservations)
            {
                foreach (var instance in reservation.Instances)
                {
                    Console.Write($"Instance ID: {instance.InstanceId} ");
                    Console.WriteLine($"\tCurrent State: {instance.State.Name}");
                }
            }
        }
    }
```

- API 세부 정보는 AWS SDK for .NET API DescribeInstances참조를 참조하십시오.

Bash

AWS CLI Bash 스크립트 사용

> **Note**
>
> 더 많은 정보가 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
###############################################################################
# function ec2_describe_instances
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i instance_id - The ID of the instance to describe (optional).
#       -q query - The query to filter the response (optional).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
```

```
###############################################################################
function ec2_describe_instances() {
  local instance_id query response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_instances"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances."
    echo "  -i instance_id - The ID of the instance to describe (optional)."
    echo "  -q query - The query to filter the response (optional)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:q:h" option; do
    case "${option}" in
      i) instance_id="${OPTARG}" ;;
      q) query="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local aws_cli_args=()

  if [[ -n "$instance_id" ]]; then
    # shellcheck disable=SC2206
    aws_cli_args+=("--instance-ids" $instance_id)
  fi

  local query_arg=""
  if [[ -n "$query" ]]; then
    query_arg="--query '$query'"
```

```
  else
    query_arg="--query Reservations[*].Instances[*].
[InstanceId,ImageId,InstanceType,KeyName,VpcId,PublicIpAddress,State.Name]"
  fi

  # shellcheck disable=SC2086
  response=$(aws ec2 describe-instances \
    "${aws_cli_args[@]}" \
    $query_arg \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports describe-instances operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}
```

이 예제에 사용된 유틸리티 함수

```
########################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
########################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

########################################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
```

```
###############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- API 세부 정보는 AWS CLI 명령 DescribeInstances참조를 참조하십시오.

C++

SDK for C++

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::DescribeInstancesRequest request;
    bool header = false;
    bool done = false;
    while (!done) {
        auto outcome = ec2Client.DescribeInstances(request);
```

```
        if (outcome.IsSuccess()) {
            if (!header) {
                std::cout << std::left <<
                          std::setw(48) << "Name" <<
                          std::setw(20) << "ID" <<
                          std::setw(25) << "Ami" <<
                          std::setw(15) << "Type" <<
                          std::setw(15) << "State" <<
                          std::setw(15) << "Monitoring" << std::endl;
                header = true;
            }

            const std::vector<Aws::EC2::Model::Reservation> &reservations =
                    outcome.GetResult().GetReservations();

            for (const auto &reservation: reservations) {
                const std::vector<Aws::EC2::Model::Instance> &instances =
                        reservation.GetInstances();
                for (const auto &instance: instances) {
                    Aws::String instanceStateString =

Aws::EC2::Model::InstanceStateNameMapper::GetNameForInstanceStateName(
                                    instance.GetState().GetName());

                    Aws::String typeString =

Aws::EC2::Model::InstanceTypeMapper::GetNameForInstanceType(
                                    instance.GetInstanceType());

                    Aws::String monitorString =

Aws::EC2::Model::MonitoringStateMapper::GetNameForMonitoringState(
                                    instance.GetMonitoring().GetState());
                    Aws::String name = "Unknown";

                    const std::vector<Aws::EC2::Model::Tag> &tags =
instance.GetTags();
                    auto nameIter = std::find_if(tags.cbegin(), tags.cend(),
                                                 [](const Aws::EC2::Model::Tag
&tag) {
                                                         return tag.GetKey() ==
"Name";
                                                 });
                    if (nameIter != tags.cend()) {
```

```
                    name = nameIter->GetValue();
                }
                std::cout <<
                          std::setw(48) << name <<
                          std::setw(20) << instance.GetInstanceId() <<
                          std::setw(25) << instance.GetImageId() <<
                          std::setw(15) << typeString <<
                          std::setw(15) << instanceStateString <<
                          std::setw(15) << monitorString << std::endl;
            }
        }

        if (!outcome.GetResult().GetNextToken().empty()) {
            request.SetNextToken(outcome.GetResult().GetNextToken());
        }
        else {
            done = true;
        }
    }
    else {
        std::cerr << "Failed to describe EC2 instances:" <<
                  outcome.GetError().GetMessage() << std::endl;
        return false;
    }
}
```

- API 세부 정보는 AWS SDK for C++ API DescribeInstances참조를 참조하십시오.

CLI

AWS CLI

예제 1: 인스턴스를 설명하는 방법

다음 `describe-instances` 예제에서는 지정된 인스턴스를 설명합니다.

```
aws ec2 describe-instances \
    --instance-ids i-1234567890abcdef0
```

출력:

```
{
```

```
    "Reservations": [
        {
            "Groups": [],
            "Instances": [
                {
                    "AmiLaunchIndex": 0,
                    "ImageId": "ami-0abcdef1234567890",
                    "InstanceId": "i-1234567890abcdef0",
                    "InstanceType": "t3.nano",
                    "KeyName": "my-key-pair",
                    "LaunchTime": "2022-11-15T10:48:59+00:00",
                    "Monitoring": {
                        "State": "disabled"
                    },
                    "Placement": {
                        "AvailabilityZone": "us-east-2a",
                        "GroupName": "",
                        "Tenancy": "default"
                    },
                    "PrivateDnsName": "ip-10-0-0-157.us-east-2.compute.internal",
                    "PrivateIpAddress": "10-0-0-157",
                    "ProductCodes": [],
                    "PublicDnsName": "ec2-34-253-223-13.us-
east-2.compute.amazonaws.com",
                    "PublicIpAddress": "34.253.223.13",
                    "State": {
                        "Code": 16,
                        "Name": "running"
                    },
                    "StateTransitionReason": "",
                    "SubnetId": "subnet-04a636d18e83cfacb",
                    "VpcId": "vpc-1234567890abcdef0",
                    "Architecture": "x86_64",
                    "BlockDeviceMappings": [
                        {
                            "DeviceName": "/dev/xvda",
                            "Ebs": {
                                "AttachTime": "2022-11-15T10:49:00+00:00",
                                "DeleteOnTermination": true,
                                "Status": "attached",
                                "VolumeId": "vol-02e6ccdca7de29cf2"
                            }
                        }
                    ],
```

```
                    "ClientToken": "1234abcd-1234-abcd-1234-d46a8903e9bc",
                    "EbsOptimized": true,
                    "EnaSupport": true,
                    "Hypervisor": "xen",
                    "IamInstanceProfile": {
                        "Arn": "arn:aws:iam::111111111111:instance-profile/
AmazonSSMRoleForInstancesQuickSetup",
                        "Id": "11111111111111111111"
                    },
                    "NetworkInterfaces": [
                        {
                            "Association": {
                                "IpOwnerId": "amazon",
                                "PublicDnsName": "ec2-34-253-223-13.us-
east-2.compute.amazonaws.com",
                                "PublicIp": "34.253.223.13"
                            },
                            "Attachment": {
                                "AttachTime": "2022-11-15T10:48:59+00:00",
                                "AttachmentId": "eni-attach-1234567890abcdefg",
                                "DeleteOnTermination": true,
                                "DeviceIndex": 0,
                                "Status": "attached",
                                "NetworkCardIndex": 0
                            },
                            "Description": "",
                            "Groups": [
                                {
                                    "GroupName": "launch-wizard-146",
                                    "GroupId": "sg-1234567890abcdefg"
                                }
                            ],
                            "Ipv6Addresses": [],
                            "MacAddress": "00:11:22:33:44:55",
                            "NetworkInterfaceId": "eni-1234567890abcdefg",
                            "OwnerId": "104024344472",
                            "PrivateDnsName": "ip-10-0-0-157.us-
east-2.compute.internal",
                            "PrivateIpAddress": "10-0-0-157",
                            "PrivateIpAddresses": [
                                {
                                    "Association": {
                                        "IpOwnerId": "amazon",
```

```
                                        "PublicDnsName": "ec2-34-253-223-13.us-
east-2.compute.amazonaws.com",
                                        "PublicIp": "34.253.223.13"
                                    },
                                    "Primary": true,
                                    "PrivateDnsName": "ip-10-0-0-157.us-
east-2.compute.internal",
                                    "PrivateIpAddress": "10-0-0-157"
                                }
                            ],
                            "SourceDestCheck": true,
                            "Status": "in-use",
                            "SubnetId": "subnet-1234567890abcdefg",
                            "VpcId": "vpc-1234567890abcdefg",
                            "InterfaceType": "interface"
                        }
                    ],
                    "RootDeviceName": "/dev/xvda",
                    "RootDeviceType": "ebs",
                    "SecurityGroups": [
                        {
                            "GroupName": "launch-wizard-146",
                            "GroupId": "sg-1234567890abcdefg"
                        }
                    ],
                    "SourceDestCheck": true,
                    "Tags": [
                        {
                            "Key": "Name",
                            "Value": "my-instance"
                        }
                    ],
                    "VirtualizationType": "hvm",
                    "CpuOptions": {
                        "CoreCount": 1,
                        "ThreadsPerCore": 2
                    },
                    "CapacityReservationSpecification": {
                        "CapacityReservationPreference": "open"
                    },
                    "HibernationOptions": {
                        "Configured": false
                    },
                    "MetadataOptions": {
```

```
                        "State": "applied",
                        "HttpTokens": "optional",
                        "HttpPutResponseHopLimit": 1,
                        "HttpEndpoint": "enabled",
                        "HttpProtocolIpv6": "disabled",
                        "InstanceMetadataTags": "enabled"
                    },
                    "EnclaveOptions": {
                        "Enabled": false
                    },
                    "PlatformDetails": "Linux/UNIX",
                    "UsageOperation": "RunInstances",
                    "UsageOperationUpdateTime": "2022-11-15T10:48:59+00:00",
                    "PrivateDnsNameOptions": {
                        "HostnameType": "ip-name",
                        "EnableResourceNameDnsARecord": true,
                        "EnableResourceNameDnsAAAARecord": false
                    },
                    "MaintenanceOptions": {
                        "AutoRecovery": "default"
                    }
                }
            ],
            "OwnerId": "111111111111",
            "ReservationId": "r-1234567890abcdefg"
        }
    ]
}
```

예제 2: 지정된 유형으로 인스턴스를 필터링하는 방법

다음 `describe-instances` 예제에서는 필터를 사용하여 결과 범위를 지정된 유형의 인스턴스로 지정합니다.

```
aws ec2 describe-instances \
    --filters Name=instance-type,Values=m5.large
```

예제 출력은 예제 1을 참조하세요.

자세한 내용은 Amazon EC2 사용 설명서에서 CLI를 사용하여 나열 및 필터링을 참조하세요.

예제 3: 지정된 유형 및 가용 영역으로 인스턴스를 필터링하는 방법

다음 describe-instances 예제에서는 여러 필터를 사용하여 결과 범위를 지정된 가용 영역에도 있는 지정된 유형의 인스턴스로 지정합니다.

```
aws ec2 describe-instances \
    --filters Name=instance-type,Values=t2.micro,t3.micro Name=availability-
zone,Values=us-east-2c
```

예제 출력은 예제 1을 참조하세요.

예제 4: JSON 파일을 사용하여 지정된 유형과 가용 영역의 인스턴스를 필터링하는 방법

다음 describe-instances 예제에서는 JSON 입력 파일을 사용하여 이전 예제와 동일한 필터링을 수행합니다. 필터가 복잡해지면 JSON 파일에서 필터를 더 쉽게 지정할 수 있습니다.

```
aws ec2 describe-instances \
    --filters file://filters.json
```

filters.json의 콘텐츠:

```
[
    {
        "Name": "instance-type",
        "Values": ["t2.micro", "t3.micro"]
    },
    {
        "Name": "availability-zone",
        "Values": ["us-east-2c"]
    }
]
```

예제 출력은 예제 1을 참조하세요.

예제 5: 지정된 소유자 태그로 인스턴스를 필터링하는 방법

다음 describe-instances 예제에서는 태그 필터를 사용하여 결과 범위를 태그 값에 관계없이 지정된 태그 키(소유자)의 태그가 있는 인스턴스로 지정합니다.

```
aws ec2 describe-instances \
    --filters "Name=tag-key,Values=Owner"
```

예제 출력은 예제 1을 참조하세요.

예제 6: 지정된 my-team 태그 값으로 인스턴스를 필터링하는 방법

다음 `describe-instances` 예제에서는 태그 필터를 사용하여 결과 범위를 태그 값에 관계없이 지정된 태그 값(my-team)의 태그가 있는 인스턴스로 지정합니다.

```
aws ec2 describe-instances \
    --filters "Name=tag-value,Values=my-team"
```

예제 출력은 예제 1을 참조하세요.

예제 7: 지정된 소유자 태그와 my-team 값으로 인스턴스를 필터링하는 방법

다음 `describe-instances` 예제에서는 태그 필터를 사용하여 결과 범위를 지정된 태그의 인스턴스(소유자=my-team)로 지정합니다.

```
aws ec2 describe-instances \
    --filters "Name=tag:Owner,Values=my-team"
```

예제 출력은 예제 1을 참조하세요.

예제 8: 모든 인스턴스의 인스턴스 및 서브넷 ID만 표시하는 방법

다음 `describe-instances` 예제에서는 `--query` 파라미터를 사용하여 모든 인스턴스의 인스턴스 및 서브넷 ID만 JSON 형식으로 표시합니다.

Linux 및 macOS:

```
aws ec2 describe-instances \
    --query 'Reservations[*].Instances[*].{Instance:InstanceId,Subnet:SubnetId}'
 \
    --output json
```

Windows:

```
aws ec2 describe-instances ^
    --query "Reservations[*].Instances[*].{Instance:InstanceId,Subnet:SubnetId}"
 ^
    --output json
```

출력:

```
[
    {
        "Instance": "i-057750d42936e468a",
        "Subnet": "subnet-069beee9b12030077"
    },
    {
        "Instance": "i-001efd250faaa6ffa",
        "Subnet": "subnet-0b715c6b7db68927a"
    },
    {
        "Instance": "i-027552a73f021f3bd",
        "Subnet": "subnet-0250c25a1f4e15235"
    }
    ...
]
```

예제 9: 지정된 유형의 인스턴스를 필터링하고 해당 인스턴스 ID만 표시하는 방법

다음 `describe-instances` 예제에서는 필터를 사용하여 결과 범위를 지정된 유형의 인스턴스로 지정하고 `--query` 파라미터를 사용하여 인스턴스 ID만 표시합니다.

```
aws ec2 describe-instances \
    --filters "Name=instance-type,Values=t2.micro" \
    --query "Reservations[*].Instances[*].[InstanceId]" \
    --output text
```

출력:

```
i-031c0dc19de2fb70c
i-00d8bff789a736b75
i-0b715c6b7db68927a
i-0626d4edd54f1286d
i-00b8ae04f9f99908e
i-0fc71c25d2374130c
```

예제 10: 지정된 유형의 인스턴스를 필터링하고 인스턴스 ID, 가용 영역, 지정된 태그 값만 표시하는 방법

다음 `describe-instances` 예제에서는 이름이 `tag-key`인 태그의 인스턴스에 대해 인스턴스 ID, 가용 영역, `Name` 태그 값을 테이블 형식으로 표시합니다.

Linux 및 macOS:

```
aws ec2 describe-instances \
    --filters Name=tag-key,Values=Name \
    --query 'Reservations[*].Instances[*].
{Instance:InstanceId,AZ:Placement.AvailabilityZone,Name:Tags[?Key==`Name`]|
[0].Value}' \
    --output table
```

Windows:

```
aws ec2 describe-instances ^
    --filters Name=tag-key,Values=Name ^
    --query "Reservations[*].Instances[*].
{Instance:InstanceId,AZ:Placement.AvailabilityZone,Name:Tags[?Key=='Name']|
[0].Value}" ^
    --output table
```

출력:

```
------------------------------------------------------------
|                    DescribeInstances                     |
+--------------+----------------------+--------------------+
|      AZ      |       Instance       |        Name        |
+--------------+----------------------+--------------------+
|  us-east-2b  |  i-057750d42936e468a  |  my-prod-server    |
|  us-east-2a  |  i-001efd250faaa6ffa  |  test-server-1     |
|  us-east-2a  |  i-027552a73f021f3bd  |  test-server-2     |
+--------------+----------------------+--------------------+
```

예제 11: 파티션 배치 그룹에서 인스턴스를 설명하는 방법

다음 `describe-instances` 예제에서는 지정된 인스턴스를 설명합니다. 응답에는 인스턴스의 배치 정보가 포함되며, 이 정보는 인스턴스의 배치 그룹 이름 및 파티션 번호를 포함합니다.

```
aws ec2 describe-instances \
    --instance-ids i-0123a456700123456 \
    --query "Reservations[*].Instances[*].Placement"
```

출력:

```
[
    [
```

```
        {
            "AvailabilityZone": "us-east-1c",
            "GroupName": "HDFS-Group-A",
            "PartitionNumber": 3,
            "Tenancy": "default"
        }

    ]
]
```

자세한 내용은 Linux 인스턴스용 Amazon EC2 사용 설명서에서 배치 그룹의 인스턴스 설명을 참조하세요.

예제 12: 지정된 배치 그룹과 파티션 번호로 인스턴스를 필터링하는 방법

다음 `describe-instances` 예제에서는 결과를 지정된 배치 그룹 및 파티션 번호의 인스턴스로만 필터링합니다.

```
aws ec2 describe-instances \
    --filters "Name=placement-group-name,Values=HDFS-Group-A" "Name=placement-
partition-number,Values=7"
```

다음에서는 출력의 관련 정보만 보여줍니다.

```
"Instances": [
    {
        "InstanceId": "i-0123a456700123456",
        "InstanceType": "r4.large",
        "Placement": {
            "AvailabilityZone": "us-east-1c",
            "GroupName": "HDFS-Group-A",
            "PartitionNumber": 7,
            "Tenancy": "default"
        }
    },
    {
        "InstanceId": "i-9876a543210987654",
        "InstanceType": "r4.large",
        "Placement": {
            "AvailabilityZone": "us-east-1c",
            "GroupName": "HDFS-Group-A",
            "PartitionNumber": 7,
            "Tenancy": "default"
```

```
        }
    ],
```

자세한 내용은 Linux 인스턴스용 Amazon EC2 사용 설명서에서 배치 그룹의 인스턴스 설명을 참조하세요.

예제 13: 인스턴스 메타데이터에서 태그에 액세스할 수 있도록 구성된 인스턴스를 필터링하는 방법

다음 `describe-instances` 예제에서는 인스턴스 메타데이터에서 인스턴스 태그에 액세스할 수 있도록 구성된 인스턴스로만 결과를 필터링합니다.

```
aws ec2 describe-instances \
    --filters "Name=metadata-options.instance-metadata-tags,Values=enabled" \
    --query "Reservations[*].Instances[*].InstanceId" \
    --output text
```

다음에서는 예상 출력을 보여줍니다.

```
i-1234567890abcdefg
i-abcdefg1234567890
i-11111111aaaaaaaaa
i-aaaaaaaa111111111
```

자세한 내용은 Amazon EC2 사용 설명서에서 인스턴스 메타데이터의 인스턴스 태그 작업을 참조하세요.

- API 세부 정보는 AWS CLI 명령 DescribeInstances참조를 참조하십시오.

Java

SDK for Java 2.x

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.ec2.Ec2Client;
```

```
import software.amazon.awssdk.services.ec2.model.DescribeInstancesRequest;
import software.amazon.awssdk.services.ec2.model.Ec2Exception;
import software.amazon.awssdk.services.ec2.paginators.DescribeInstancesIterable;

/**
 * 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 DescribeInstances {
    public static void main(String[] args) {
        Region region = Region.US_EAST_1;
        Ec2Client ec2 = Ec2Client.builder()
            .region(region)
            .build();

        describeEC2Instances(ec2);
        ec2.close();
    }

    public static void describeEC2Instances(Ec2Client ec2) {
        try {
            DescribeInstancesRequest request = DescribeInstancesRequest.builder()
                .maxResults(10)
                .build();

            DescribeInstancesIterable instancesIterable =
 ec2.describeInstancesPaginator(request);
            instancesIterable.stream()
                .flatMap(r -> r.reservations().stream())
                .flatMap(reservation -> reservation.instances().stream())
                .forEach(instance -> {
                    System.out.println("Instance Id is " +
 instance.instanceId());
                    System.out.println("Image id is " + instance.imageId());
                    System.out.println("Instance type is " +
 instance.instanceType());
                    System.out.println("Instance state name is " +
 instance.state().name());
```

```
                    System.out.println("Monitoring information is " +
 instance.monitoring().state());
                });

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorCode());
            System.exit(1);
        }
    }
}
```

- API 세부 정보는 AWS SDK for Java 2.x API DescribeInstances참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { DescribeInstancesCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

// List all of your EC2 instances running with x86_64 architecture that were
// launched this month.
export const main = async () => {
  const d = new Date();
  const year = d.getFullYear();
  const month = `0${d.getMonth() + 1}`.slice(-2);
  const launchTimePattern = `${year}-${month}-*`;
  const command = new DescribeInstancesCommand({
    Filters: [
      { Name: "architecture", Values: ["x86_64"] },
      { Name: "instance-state-name", Values: ["running"] },
      {
        Name: "launch-time",
        Values: [launchTimePattern],
```

```
      },
    ],
  });

  try {
    const { Reservations } = await client.send(command);
    const instanceList = Reservations.reduce((prev, current) => {
      return prev.concat(current.Instances);
    }, []);

    console.log(instanceList);
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API DescribeInstances참조를 참조하십시오.

Kotlin

SDK for Kotlin

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
suspend fun describeEC2Instances() {
    val request =
        DescribeInstancesRequest {
            maxResults = 6
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.describeInstances(request)
        response.reservations?.forEach { reservation ->
            reservation.instances?.forEach { instance ->
                println("Instance Id is ${instance.instanceId}")
                println("Image id is ${instance.imageId}")
                println("Instance type is ${instance.instanceType}")
```

```
                println("Instance state name is ${instance.state?.name}")
                println("monitoring information is
 ${instance.monitoring?.state}")
            }
        }
    }
}
```

- API 세부 정보는 Kotlin API용AWS SDK 레퍼런스를 참조하세요 DescribeInstances.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예제는 지정된 인스턴스를 설명합니다.

```
(Get-EC2Instance -InstanceId i-12345678).Instances
```

출력:

```
AmiLaunchIndex        : 0
Architecture          : x86_64
BlockDeviceMappings   : {/dev/sda1}
ClientToken           : TleEy1448154045270
EbsOptimized          : False
Hypervisor            : xen
IamInstanceProfile    : Amazon.EC2.Model.IamInstanceProfile
ImageId               : ami-12345678
InstanceId            : i-12345678
InstanceLifecycle     :
InstanceType          : t2.micro
KernelId              :
KeyName               : my-key-pair
LaunchTime            : 12/4/2015 4:44:40 PM
Monitoring            : Amazon.EC2.Model.Monitoring
NetworkInterfaces     : {ip-10-0-2-172.us-west-2.compute.internal}
Placement             : Amazon.EC2.Model.Placement
Platform              : Windows
PrivateDnsName        : ip-10-0-2-172.us-west-2.compute.internal
PrivateIpAddress      : 10.0.2.172
ProductCodes          : {}
```

```
PublicDnsName         :
PublicIpAddress       :
RamdiskId             :
RootDeviceName        : /dev/sda1
RootDeviceType        : ebs
SecurityGroups        : {default}
SourceDestCheck       : True
SpotInstanceRequestId :
SriovNetSupport       :
State                 : Amazon.EC2.Model.InstanceState
StateReason           :
StateTransitionReason :
SubnetId              : subnet-12345678
Tags                  : {Name}
VirtualizationType    : hvm
VpcId                 : vpc-12345678
```

예 2: 이 예에서는 현재 지역의 모든 인스턴스를 예약별로 그룹화하여 설명합니다. 인스턴스 세부 정보를 보려면 각 예약 개체 내의 인스턴스 컬렉션을 확장하십시오.

```
Get-EC2Instance
```

출력:

```
GroupNames    : {}
Groups        : {}
Instances     : {}
OwnerId       : 123456789012
RequesterId   : 226008221399
ReservationId : r-c5df370c

GroupNames    : {}
Groups        : {}
Instances     : {}
OwnerId       : 123456789012
RequesterId   : 854251627541
ReservationId : r-63e65bab
...
```

예 3: 이 예제는 필터를 사용하여 VPC의 특정 서브넷에서 EC2 인스턴스를 쿼리하는 방법을 보여줍니다.

```
(Get-EC2Instance -Filter @{Name="vpc-id";Values="vpc-1a2bc34d"},@{Name="subnet-
id";Values="subnet-1a2b3c4d"}).Instances
```

출력:

```
InstanceId          InstanceType Platform PrivateIpAddress PublicIpAddress
 SecurityGroups SubnetId        VpcId
----------          ------------ -------- ---------------- --------------
 -------------- --------        -----
i-01af...82cf180e19 t2.medium    Windows  10.0.0.98                      ...
         subnet-1a2b3c4d vpc-1a2b3c4d
i-0374...7e9d5b0c45 t2.xlarge    Windows  10.0.0.53                      ...
         subnet-1a2b3c4d vpc-1a2b3c4d
```

- API 세부 정보는 Cmdlet 참조를 참조하십시오 DescribeInstances.AWS Tools for PowerShell

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
 actions."""

    def __init__(self, ec2_resource, instance=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param instance: A Boto3 Instance object. This is a high-level object
 that
                          wraps instance actions.
        """
```

```
        self.ec2_resource = ec2_resource
        self.instance = instance

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def display(self, indent=1):
        """
        Displays information about an instance.

        :param indent: The visual indent to apply to the output.
        """
        if self.instance is None:
            logger.info("No instance to display.")
            return

        try:
            self.instance.load()
            ind = "\t" * indent
            print(f"{ind}ID: {self.instance.id}")
            print(f"{ind}Image ID: {self.instance.image_id}")
            print(f"{ind}Instance type: {self.instance.instance_type}")
            print(f"{ind}Key name: {self.instance.key_name}")
            print(f"{ind}VPC ID: {self.instance.vpc_id}")
            print(f"{ind}Public IP: {self.instance.public_ip_address}")
            print(f"{ind}State: {self.instance.state['Name']}")
        except ClientError as err:
            logger.error(
                "Couldn't display your instance. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 DescribeInstances.

Ruby

SDK for Ruby

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
require "aws-sdk-ec2"

# @param ec2_resource [Aws::EC2::Resource] An initialized EC2 resource object.
# @example
#   list_instance_ids_states(Aws::EC2::Resource.new(region: 'us-west-2'))
def list_instance_ids_states(ec2_resource)
  response = ec2_resource.instances
  if response.count.zero?
    puts "No instances found."
  else
    puts "Instances -- ID, state:"
    response.each do |instance|
      puts "#{instance.id}, #{instance.state.name}"
    end
  end
rescue StandardError => e
  puts "Error getting information about instances: #{e.message}"
end

# Example usage:
def run_me
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:   ruby ec2-ruby-example-get-all-instance-info.rb REGION"
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    puts "Example: ruby ec2-ruby-example-get-all-instance-info.rb us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  elsif ARGV.count.zero?
```

```
    region = "us-west-2"
  # Otherwise, use the values as specified at the command prompt.
  else
    region = ARGV[0]
  end
  ec2_resource = Aws::EC2::Resource.new(region: region)
  list_instance_ids_states(ec2_resource)
end

run_me if $PROGRAM_NAME == __FILE__
```

- API 세부 정보는 AWS SDK for Ruby API DescribeInstances참조를 참조하십시오.

Rust

SDK for Rust

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
async fn show_state(client: &Client, ids: Option<Vec<String>>) -> Result<(),
 Error> {
    let resp = client
        .describe_instances()
        .set_instance_ids(ids)
        .send()
        .await?;

    for reservation in resp.reservations() {
        for instance in reservation.instances() {
            println!("Instance ID: {}", instance.instance_id().unwrap());
            println!(
                "State:       {:?}",
                instance.state().unwrap().name().unwrap()
            );
            println!();
        }
    }
```

```
    Ok(())
}
```

- API에 대한 자세한 내용은 Rust용AWS SDK API 레퍼런스를 참조하십시오 DescribeInstances.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> 자세한 내용은 여기에서 확인할 수 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    TRY.
        oo_result = lo_ec2->describeinstances( ) .                      "
oo_result is returned for testing purposes. "

        " Retrieving details of EC2 instances. "
        DATA: lv_istance_id    TYPE /aws1/ec2string,
              lv_status        TYPE /aws1/ec2instancestatename,
              lv_instance_type TYPE /aws1/ec2instancetype,
              lv_image_id      TYPE /aws1/ec2string.
        LOOP AT oo_result->get_reservations( ) INTO DATA(lo_reservation).
          LOOP AT lo_reservation->get_instances( ) INTO DATA(lo_instance).
            lv_istance_id = lo_instance->get_instanceid( ).
            lv_status =  lo_instance->get_state( )->get_name( ).
            lv_instance_type = lo_instance->get_instancetype( ).
            lv_image_id = lo_instance->get_imageid( ).
          ENDLOOP.
        ENDLOOP.
        MESSAGE 'Retrieved information about EC2 instances.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- API 세부 정보는 SAP용AWS SDK ABAP API 참조를 참조하십시오 DescribeInstances.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeInternetGateways** CLI와 함께 사용

다음 코드 예제는 DescribeInternetGateways의 사용 방법을 보여줍니다.

CLI

AWS CLI

인터넷 게이트웨이에 대해 설명하려면

다음 describe-internet-gateways 예에서는 지정된 인터넷 게이트웨이를 설명합니다.

```
aws ec2 describe-internet-gateways \
    --internet-gateway-ids igw-0d0fb496b3EXAMPLE
```

출력:

```
{
    "InternetGateways": [
        {
            "Attachments": [
                {
                    "State": "available",
                    "VpcId": "vpc-0a60eb65b4EXAMPLE"
                }
            ],
            "InternetGatewayId": "igw-0d0fb496b3EXAMPLE",
            "OwnerId": "123456789012",
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "my-igw"
                }
```

```
            ]
        }
    ]
}
```

자세한 내용은 Amazon VPC 사용 설명서의 인터넷 게이트웨이를 참조하세요.

- API 세부 정보는 AWS CLI 명령 DescribeInternetGateways참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 지정된 인터넷 게이트웨이를 설명합니다.

```
Get-EC2InternetGateway -InternetGatewayId igw-1a2b3c4d
```

출력:

```
Attachments       InternetGatewayId     Tags
-----------       -----------------     ----
{vpc-1a2b3c4d}    igw-1a2b3c4d          {}
```

예 2: 이 예에서는 모든 인터넷 게이트웨이를 설명합니다.

```
Get-EC2InternetGateway
```

출력:

```
Attachments       InternetGatewayId     Tags
-----------       -----------------     ----
{vpc-1a2b3c4d}    igw-1a2b3c4d          {}
{}                igw-2a3b4c5d          {}
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeInternetGateways.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeKeyPairs** CLI와 함께 사용

다음 코드 예제는 DescribeKeyPairs의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 인스턴스 시작하기

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
/// <summary>
/// Get information about an Amazon EC2 key pair.
/// </summary>
/// <param name="keyPairName">The name of the key pair.</param>
/// <returns>A list of key pair information.</returns>
public async Task<List<KeyPairInfo>> DescribeKeyPairs(string keyPairName)
{
    var request = new DescribeKeyPairsRequest();
    if (!string.IsNullOrEmpty(keyPairName))
    {
        request = new DescribeKeyPairsRequest
        {
            KeyNames = new List<string> { keyPairName }
        };
    }
    var response = await _amazonEC2.DescribeKeyPairsAsync(request);
    return response.KeyPairs.ToList();
}
```

- API 세부 정보는 AWS SDK for .NET API DescribeKeyPairs참조를 참조하십시오.

Bash

AWS CLI Bash 스크립트 사용

> **Note**
>
> 더 많은 정보가 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
##############################################################################
# function ec2_describe_key_pairs
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 key pairs.
#
# Parameters:
#       -h - Display help.
#
# And:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function ec2_describe_key_pairs() {
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_key_pairs"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2) key
 pairs."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "h" option; do
    case "${option}" in
      h)
        usage
        return 0
        ;;
```

```
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local response

  response=$(aws ec2 describe-key-pairs \
    --query 'KeyPairs[*].[KeyName, KeyFingerprint]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports describe-key-pairs operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}
```

이 예제에 사용된 유틸리티 함수

```
########################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
########################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

########################################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
```

```
#
#  Returns:
#          0: - Success.
#
###############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- API 세부 정보는 AWS CLI 명령 DescribeKeyPairs참조를 참조하십시오.

C++

SDK for C++

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::DescribeKeyPairsRequest request;
```

```
    auto outcome = ec2Client.DescribeKeyPairs(request);
    if (outcome.IsSuccess()) {
        std::cout << std::left <<
                  std::setw(32) << "Name" <<
                  std::setw(64) << "Fingerprint" << std::endl;

        const std::vector<Aws::EC2::Model::KeyPairInfo> &key_pairs =
                outcome.GetResult().GetKeyPairs();
        for (const auto &key_pair: key_pairs) {
            std::cout << std::left <<
                      std::setw(32) << key_pair.GetKeyName() <<
                      std::setw(64) << key_pair.GetKeyFingerprint() << std::endl;
        }
    }
    else {
        std::cerr << "Failed to describe key pairs:" <<
                  outcome.GetError().GetMessage() << std::endl;
    }
```

- API 세부 정보는 AWS SDK for C++ API DescribeKeyPairs참조를 참조하십시오.

CLI

AWS CLI

키 페어를 표시하는 방법

다음 `describe-key-pairs` 예제는 지정된 키 페어에 대한 정보를 표시합니다.

```
aws ec2 describe-key-pairs \
    --key-names my-key-pair
```

출력:

```
{
    "KeyPairs": [
        {
            "KeyPairId": "key-0b94643da6EXAMPLE",
            "KeyFingerprint":
 "1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f",
```

```
            "KeyName": "my-key-pair",
            "KeyType": "rsa",
            "Tags": [],
            "CreateTime": "2022-05-27T21:51:16.000Z"
        }
    ]
}
```

자세한 내용은 Amazon EC2 사용 설명서에서 퍼블릭 키 설명을 참조하세요.

- API 세부 정보는 AWS CLI 명령 DescribeKeyPairs참조를 참조하십시오.

Java

SDK for Java 2.x

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    public static void describeKeys(Ec2Client ec2) {
        try {
            DescribeKeyPairsResponse response = ec2.describeKeyPairs();
            response.keyPairs().forEach(keyPair -> System.out.printf(
                    "Found key pair with name %s " +
                            "and fingerprint %s",
                    keyPair.keyName(),
                    keyPair.keyFingerprint()));

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API DescribeKeyPairs참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { DescribeKeyPairsCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new DescribeKeyPairsCommand({});

  try {
    const { KeyPairs } = await client.send(command);
    const keyPairList = KeyPairs.map(
      (kp) => ` • ${kp.KeyPairId}: ${kp.KeyName}`,
    ).join("\n");
    console.log("The following key pairs were found in your account:");
    console.log(keyPairList);
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API DescribeKeyPairs참조를 참조하십시오.

Kotlin

SDK for Kotlin

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
suspend fun describeEC2Keys() {
    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.describeKeyPairs(DescribeKeyPairsRequest {})
        response.keyPairs?.forEach { keyPair ->
            println("Found key pair with name ${keyPair.keyName} and fingerprint
 ${ keyPair.keyFingerprint}")
        }
    }
}
```

- API 세부 정보는 Kotlin API용AWS SDK 레퍼런스를 참조하세요 DescribeKeyPairs.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예제에서는 지정된 키 쌍을 설명합니다.

```
Get-EC2KeyPair -KeyName my-key-pair
```

출력:

```
KeyFingerprint                                                  KeyName
--------------                                                  -------
1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f my-key-pair
```

예 2: 이 예제에서는 모든 키 페어를 설명합니다.

```
Get-EC2KeyPair
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeKeyPairs.

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class KeyPairWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) key pair
 actions."""

    def __init__(self, ec2_resource, key_file_dir, key_pair=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param key_file_dir: The folder where the private key information is
 stored.
                             This should be a secure folder.
        :param key_pair: A Boto3 KeyPair object. This is a high-level object that
                         wraps key pair actions.
        """
        self.ec2_resource = ec2_resource
        self.key_pair = key_pair
        self.key_file_path = None
        self.key_file_dir = key_file_dir

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource, tempfile.TemporaryDirectory())


    def list(self, limit):
        """
        Displays a list of key pairs for the current account.

        :param limit: The maximum number of key pairs to list.
```

```
        """
        try:
            for kp in self.ec2_resource.key_pairs.limit(limit):
                print(f"Found {kp.key_type} key {kp.name} with fingerprint:")
                print(f"\t{kp.key_fingerprint}")
        except ClientError as err:
            logger.error(
                "Couldn't list key pairs. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 DescribeKeyPairs.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    TRY.
        oo_result = lo_ec2->describekeypairs( ) .                       "
 oo_result is returned for testing purposes. "
        DATA(lt_key_pairs) = oo_result->get_keypairs( ).
        MESSAGE 'Retrieved information about key pairs.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- API 세부 정보는 SAP용AWS SDK ABAP API 참조를 참조하십시오 DescribeKeyPairs.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeNetworkAcls** CLI와 함께 사용

다음 코드 예제는 DescribeNetworkAcls의 사용 방법을 보여줍니다.

CLI

AWS CLI

네트워크 ACL에 대해 설명하려면

다음 describe-network-acls 예제는 네트워크 ACL에 대한 세부 정보를 검색합니다.

```
aws ec2 describe-network-acls
```

출력:

```
{
    "NetworkAcls": [
        {
            "Associations": [
                {
                    "NetworkAclAssociationId": "aclassoc-0c1679dc41EXAMPLE",
                    "NetworkAclId": "acl-0ea1f54ca7EXAMPLE",
                    "SubnetId": "subnet-0931fc2fa5EXAMPLE"
                }
            ],
            "Entries": [
                {
                    "CidrBlock": "0.0.0.0/0",
                    "Egress": true,
                    "Protocol": "-1",
                    "RuleAction": "allow",
                    "RuleNumber": 100
                },
                {
                    "CidrBlock": "0.0.0.0/0",
                    "Egress": true,
                    "Protocol": "-1",
                    "RuleAction": "deny",
```

```
                    "RuleNumber": 32767
                },
                {
                    "CidrBlock": "0.0.0.0/0",
                    "Egress": false,
                    "Protocol": "-1",
                    "RuleAction": "allow",
                    "RuleNumber": 100
                },
                {
                    "CidrBlock": "0.0.0.0/0",
                    "Egress": false,
                    "Protocol": "-1",
                    "RuleAction": "deny",
                    "RuleNumber": 32767
                }
            ],
            "IsDefault": true,
            "NetworkAclId": "acl-0ea1f54ca7EXAMPLE",
            "Tags": [],
            "VpcId": "vpc-06e4ab6c6cEXAMPLE",
            "OwnerId": "111122223333"
        },
        {
            "Associations": [],
            "Entries": [
                {
                    "CidrBlock": "0.0.0.0/0",
                    "Egress": true,
                    "Protocol": "-1",
                    "RuleAction": "allow",
                    "RuleNumber": 100
                },
                {
                    "Egress": true,
                    "Ipv6CidrBlock": "::/0",
                    "Protocol": "-1",
                    "RuleAction": "allow",
                    "RuleNumber": 101
                },
                {
                    "CidrBlock": "0.0.0.0/0",
                    "Egress": true,
                    "Protocol": "-1",
```

```
                    "RuleAction": "deny",
                    "RuleNumber": 32767
                },
                {
                    "Egress": true,
                    "Ipv6CidrBlock": "::/0",
                    "Protocol": "-1",
                    "RuleAction": "deny",
                    "RuleNumber": 32768
                },
                {
                    "CidrBlock": "0.0.0.0/0",
                    "Egress": false,
                    "Protocol": "-1",
                    "RuleAction": "allow",
                    "RuleNumber": 100
                },
                {
                    "Egress": false,
                    "Ipv6CidrBlock": "::/0",
                    "Protocol": "-1",
                    "RuleAction": "allow",
                    "RuleNumber": 101
                },
                {
                    "CidrBlock": "0.0.0.0/0",
                    "Egress": false,
                    "Protocol": "-1",
                    "RuleAction": "deny",
                    "RuleNumber": 32767
                },
                {
                    "Egress": false,
                    "Ipv6CidrBlock": "::/0",
                    "Protocol": "-1",
                    "RuleAction": "deny",
                    "RuleNumber": 32768
                }
            ],
            "IsDefault": true,
            "NetworkAclId": "acl-0e2a78e4e2EXAMPLE",
            "Tags": [],
            "VpcId": "vpc-03914afb3eEXAMPLE",
            "OwnerId": "111122223333"
```

```
        }
    ]
}
```

자세한 내용은 AWS VPC 사용 설명서의 네트워크 ACL을 참조하십시오.

- API 세부 정보는 AWS CLI 명령어 참조를 참조하십시오 DescribeNetworkAcls.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 지정된 네트워크 ACL을 설명합니다.

```
Get-EC2NetworkAcl -NetworkAclId acl-12345678
```

출력:

```
Associations : {aclassoc-1a2b3c4d}
Entries      : {Amazon.EC2.Model.NetworkAclEntry,
 Amazon.EC2.Model.NetworkAclEntry}
IsDefault    : False
NetworkAclId : acl-12345678
Tags         : {Name}
VpcId        : vpc-12345678
```

예 2: 이 예에서는 지정된 네트워크 ACL의 규칙을 설명합니다.

```
(Get-EC2NetworkAcl -NetworkAclId acl-12345678).Entries
```

출력:

```
CidrBlock    : 0.0.0.0/0
Egress       : True
IcmpTypeCode :
PortRange    :
Protocol     : -1
RuleAction   : deny
RuleNumber   : 32767

CidrBlock    : 0.0.0.0/0
```

```
Egress       : False
IcmpTypeCode :
PortRange    :
Protocol     : -1
RuleAction   : deny
RuleNumber   : 32767
```

예 3: 이 예에서는 모든 네트워크 ACL을 설명합니다.

```
Get-EC2NetworkAcl
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeNetworkAcls.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeNetworkInterfaceAttribute** CLI와 함께 사용

다음 코드 예제는 DescribeNetworkInterfaceAttribute의 사용 방법을 보여줍니다.

CLI

AWS CLI

네트워크 인터페이스의 첨부 속성 설명하기

이 예제 명령은 지정된 네트워크 인터페이스의 attachment 속성을 설명합니다.

명령:

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200
 --attribute attachment
```

출력:

```
{
  "NetworkInterfaceId": "eni-686ea200",
```

```
  "Attachment": {
      "Status": "attached",
      "DeviceIndex": 0,
      "AttachTime": "2015-05-21T20:02:20.000Z",
      "InstanceId": "i-1234567890abcdef0",
      "DeleteOnTermination": true,
      "AttachmentId": "eni-attach-43348162",
      "InstanceOwnerId": "123456789012"
  }
}
```

네트워크 인터페이스의 설명 속성을 설명하려면

이 예제 명령은 지정된 네트워크 인터페이스의 `description` 속성을 설명합니다.

명령:

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200
 --attribute description
```

출력:

```
{
  "NetworkInterfaceId": "eni-686ea200",
  "Description": {
      "Value": "My description"
  }
}
```

네트워크 인터페이스의 GroupSet 속성을 설명하려면

이 예제 명령은 지정된 네트워크 `groupSet` 인터페이스의 속성을 설명합니다.

명령:

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200
 --attribute groupSet
```

출력:

```
{
  "NetworkInterfaceId": "eni-686ea200",
```

```
  "Groups": [
      {
          "GroupName": "my-security-group",
          "GroupId": "sg-903004f8"
      }
  ]
}
```

네트워크 인터페이스의 sourceDestCheck 속성을 설명하려면

이 예제 명령은 지정된 네트워크 인터페이스의 `sourceDestCheck` 속성을 설명합니다.

명령:

```
aws ec2 describe-network-interface-attribute --network-interface-id eni-686ea200
 --attribute sourceDestCheck
```

출력:

```
{
  "NetworkInterfaceId": "eni-686ea200",
  "SourceDestCheck": {
      "Value": true
  }
}
```

- API 세부 정보는 AWS CLI 명령 DescribeNetworkInterfaceAttribute참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 지정된 네트워크 인터페이스를 설명합니다.

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute
 Attachment
```

출력:

```
Attachment         : Amazon.EC2.Model.NetworkInterfaceAttachment
```

예 2: 이 예에서는 지정된 네트워크 인터페이스를 설명합니다.

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute
 Description
```

출력:

```
Description        : My description
```

예 3: 이 예제에서는 지정된 네트워크 인터페이스를 설명합니다.

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute
 GroupSet
```

출력:

```
Groups             : {my-security-group}
```

예 4: 이 예제에서는 지정된 네트워크 인터페이스를 설명합니다.

```
Get-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-12345678 -Attribute
 SourceDestCheck
```

출력:

```
SourceDestCheck    : True
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeNetworkInterfaceAttribute.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeNetworkInterfaces** CLI와 함께 사용

다음 코드 예제는 DescribeNetworkInterfaces의 사용 방법을 보여줍니다.

CLI

AWS CLI

네트워크 인터페이스 설명하기

이 예제에서는 모든 네트워크 인터페이스를 설명합니다.

명령:

```
aws ec2 describe-network-interfaces
```

출력:

```
{
  "NetworkInterfaces": [
      {
          "Status": "in-use",
          "MacAddress": "02:2f:8f:b0:cf:75",
          "SourceDestCheck": true,
          "VpcId": "vpc-a01106c2",
          "Description": "my network interface",
          "Association": {
              "PublicIp": "203.0.113.12",
              "AssociationId": "eipassoc-0fbb766a",
              "PublicDnsName": "ec2-203-0-113-12.compute-1.amazonaws.com",
              "IpOwnerId": "123456789012"
          },
          "NetworkInterfaceId": "eni-e5aa89a3",
          "PrivateIpAddresses": [
              {
                  "PrivateDnsName": "ip-10-0-1-17.ec2.internal",
                  "Association": {
                      "PublicIp": "203.0.113.12",
                      "AssociationId": "eipassoc-0fbb766a",
                      "PublicDnsName":
 "ec2-203-0-113-12.compute-1.amazonaws.com",
                      "IpOwnerId": "123456789012"
                  },
                  "Primary": true,
                  "PrivateIpAddress": "10.0.1.17"
              }
          ],
          "RequesterManaged": false,
```

```
            "Ipv6Addresses": [],
            "PrivateDnsName": "ip-10-0-1-17.ec2.internal",
            "AvailabilityZone": "us-east-1d",
            "Attachment": {
                "Status": "attached",
                "DeviceIndex": 1,
                "AttachTime": "2013-11-30T23:36:42.000Z",
                "InstanceId": "i-1234567890abcdef0",
                "DeleteOnTermination": false,
                "AttachmentId": "eni-attach-66c4350a",
                "InstanceOwnerId": "123456789012"
            },
            "Groups": [
                {
                    "GroupName": "default",
                    "GroupId": "sg-8637d3e3"
                }
            ],
            "SubnetId": "subnet-b61f49f0",
            "OwnerId": "123456789012",
            "TagSet": [],
            "PrivateIpAddress": "10.0.1.17"
        },
        {
            "Status": "in-use",
            "MacAddress": "02:58:f5:ef:4b:06",
            "SourceDestCheck": true,
            "VpcId": "vpc-a01106c2",
            "Description": "Primary network interface",
            "Association": {
                "PublicIp": "198.51.100.0",
                "IpOwnerId": "amazon"
            },
            "NetworkInterfaceId": "eni-f9ba99bf",
            "PrivateIpAddresses": [
                {
                    "Association": {
                        "PublicIp": "198.51.100.0",
                        "IpOwnerId": "amazon"
                    },
                    "Primary": true,
                    "PrivateIpAddress": "10.0.1.149"
                }
            ],
```

```
            "RequesterManaged": false,
            "Ipv6Addresses": [],
            "AvailabilityZone": "us-east-1d",
            "Attachment": {
                "Status": "attached",
                "DeviceIndex": 0,
                "AttachTime": "2013-11-30T23:35:33.000Z",
                "InstanceId": "i-0598c7d356eba48d7",
                "DeleteOnTermination": true,
                "AttachmentId": "eni-attach-1b9db777",
                "InstanceOwnerId": "123456789012"
            },
            "Groups": [
                {
                    "GroupName": "default",
                    "GroupId": "sg-8637d3e3"
                }
            ],
            "SubnetId": "subnet-b61f49f0",
            "OwnerId": "123456789012",
            "TagSet": [],
            "PrivateIpAddress": "10.0.1.149"
        }
    ]
}
```

이 예제에서는 Purpose 키와 값이 포함된 태그가 있는 네트워크 인터페이스를 설명합니다Prod.

명령:

```
aws ec2 describe-network-interfaces --filters Name=tag:Purpose,Values=Prod
```

출력:

```
{
    "NetworkInterfaces": [
        {
            "Status": "available",
            "MacAddress": "12:2c:bd:f9:bf:17",
            "SourceDestCheck": true,
            "VpcId": "vpc-8941ebec",
            "Description": "ProdENI",
```

```
            "NetworkInterfaceId": "eni-b9a5ac93",
            "PrivateIpAddresses": [
                {
                    "PrivateDnsName": "ip-10-0-1-55.ec2.internal",
                    "Primary": true,
                    "PrivateIpAddress": "10.0.1.55"
                },
                {
                    "PrivateDnsName": "ip-10-0-1-117.ec2.internal",
                    "Primary": false,
                    "PrivateIpAddress": "10.0.1.117"
                }
            ],
            "RequesterManaged": false,
            "PrivateDnsName": "ip-10-0-1-55.ec2.internal",
            "AvailabilityZone": "us-east-1d",
            "Ipv6Addresses": [],
            "Groups": [
                {
                    "GroupName": "MySG",
                    "GroupId": "sg-905002f5"
                }
            ],
            "SubnetId": "subnet-31d6c219",
            "OwnerId": "123456789012",
            "TagSet": [
                {
                    "Value": "Prod",
                    "Key": "Purpose"
                }
            ],
            "PrivateIpAddress": "10.0.1.55"
        }
    ]
}
```

- API 세부 정보는 AWS CLI 명령 DescribeNetworkInterfaces참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 지정된 네트워크 인터페이스를 설명합니다.

```
Get-EC2NetworkInterface -NetworkInterfaceId eni-12345678
```

출력:

```
Association        :
Attachment         : Amazon.EC2.Model.NetworkInterfaceAttachment
AvailabilityZone   : us-west-2c
Description        :
Groups             : {my-security-group}
MacAddress         : 0a:e9:a6:19:4c:7f
NetworkInterfaceId : eni-12345678
OwnerId            : 123456789012
PrivateDnsName     : ip-10-0-0-107.us-west-2.compute.internal
PrivateIpAddress   : 10.0.0.107
PrivateIpAddresses : {ip-10-0-0-107.us-west-2.compute.internal}
RequesterId        :
RequesterManaged   : False
SourceDestCheck    : True
Status             : in-use
SubnetId           : subnet-1a2b3c4d
TagSet             : {}
VpcId              : vpc-12345678
```

예 2: 이 예제에서는 모든 네트워크 인터페이스를 설명합니다.

```
Get-EC2NetworkInterface
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeNetworkInterfaces.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribePlacementGroups** CLI와 함께 사용

다음 코드 예제는 DescribePlacementGroups의 사용 방법을 보여줍니다.

CLI

AWS CLI

배치 그룹을 설명하려면

이 예제 명령은 모든 배치 그룹을 설명합니다.

명령:

```
aws ec2 describe-placement-groups
```

출력:

```
{
    "PlacementGroups": [
        {
            "GroupName": "my-cluster",
            "State": "available",
            "Strategy": "cluster"
        },
        ...
    ]
}
```

- API 세부 정보는 AWS CLI 명령 DescribePlacementGroups참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 지정된 배치 그룹을 설명합니다.

```
Get-EC2PlacementGroup -GroupName my-placement-group
```

출력:

```
GroupName             State        Strategy
---------             -----        --------
my-placement-group    available    cluster
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribePlacementGroups.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribePrefixLists** CLI와 함께 사용

다음 코드 예제는 DescribePrefixLists의 사용 방법을 보여줍니다.

CLI

AWS CLI

접두사 목록 설명하기

이 예제는 해당 지역에서 사용 가능한 모든 접두사 목록을 나열합니다.

명령:

```
aws ec2 describe-prefix-lists
```

출력:

```
{
  "PrefixLists": [
    {
      "PrefixListName": "com.amazonaws.us-east-1.s3",
      "Cidrs": [
        "54.231.0.0/17"
      ],
      "PrefixListId": "pl-63a5400a"
    }
  ]
}
```

- API 세부 정보는 AWS CLI 명령 DescribePrefixLists참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제는 해당 지역의 접두사 목록 AWS 서비스 형식으로 사용할 수 있는 항목을 가져옵니다.

```
Get-EC2PrefixList
```

출력:

```
Cidrs                                           PrefixListId PrefixListName
-----                                           ------------ --------------
{52.94.5.0/24, 52.119.240.0/21, 52.94.24.0/23} pl-6fa54006  com.amazonaws.eu-
west-1.dynamodb
{52.218.0.0/17, 54.231.128.0/19}                pl-6da54004  com.amazonaws.eu-
west-1.s3
```

- API 세부 정보는 Cmdlet 참조를 참조하십시오 DescribePrefixLists.AWS Tools for PowerShell

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeRegions** CLI와 함께 사용

다음 코드 예제는 DescribeRegions의 사용 방법을 보여줍니다.

C++

SDK for C++

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
```

```
    Aws::EC2::Model::DescribeRegionsRequest request;
    auto outcome = ec2Client.DescribeRegions(request);
    bool result = true;
    if (outcome.IsSuccess()) {
        std::cout << std::left <<
                  std::setw(32) << "RegionName" <<
                  std::setw(64) << "Endpoint" << std::endl;

        const auto &regions = outcome.GetResult().GetRegions();
        for (const auto &region: regions) {
            std::cout << std::left <<
                      std::setw(32) << region.GetRegionName() <<
                      std::setw(64) << region.GetEndpoint() << std::endl;
        }
    }
    else {
        std::cerr << "Failed to describe regions:" <<
                  outcome.GetError().GetMessage() << std::endl;
        result = false;
    }
```

- API 세부 정보는 AWS SDK for C++ API DescribeRegions참조를 참조하십시오.

CLI

AWS CLI

예제 1: 활성화된 모든 리전을 설명하는 방법

다음 `describe-regions` 예제에서는 계정에서 활성화된 모든 리전을 설명합니다.

```
aws ec2 describe-regions
```

출력:

```
{
    "Regions": [
        {
            "Endpoint": "ec2.eu-north-1.amazonaws.com",
            "RegionName": "eu-north-1",
            "OptInStatus": "opt-in-not-required"
```

```
        },
        {
            "Endpoint": "ec2.ap-south-1.amazonaws.com",
            "RegionName": "ap-south-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-west-3.amazonaws.com",
            "RegionName": "eu-west-3",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-west-2.amazonaws.com",
            "RegionName": "eu-west-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-west-1.amazonaws.com",
            "RegionName": "eu-west-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-northeast-3.amazonaws.com",
            "RegionName": "ap-northeast-3",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-northeast-2.amazonaws.com",
            "RegionName": "ap-northeast-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-northeast-1.amazonaws.com",
            "RegionName": "ap-northeast-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.sa-east-1.amazonaws.com",
            "RegionName": "sa-east-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ca-central-1.amazonaws.com",
            "RegionName": "ca-central-1",
```

```
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-southeast-1.amazonaws.com",
            "RegionName": "ap-southeast-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-southeast-2.amazonaws.com",
            "RegionName": "ap-southeast-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-central-1.amazonaws.com",
            "RegionName": "eu-central-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-east-1.amazonaws.com",
            "RegionName": "us-east-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-east-2.amazonaws.com",
            "RegionName": "us-east-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-west-1.amazonaws.com",
            "RegionName": "us-west-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-west-2.amazonaws.com",
            "RegionName": "us-west-2",
            "OptInStatus": "opt-in-not-required"
        }
    ]
}
```

자세한 내용은 Amazon EC2 사용 설명서에서 리전 및 가용 영역을 참조하세요.

예제 2: 이름에 특정 문자열이 포함된 엔드포인트가 있는 활성화된 리전을 설명하는 방법

다음 `describe-regions` 예제에서는 엔드포인트에 'us' 문자열이 포함된 활성화한 모든 리전을 설명합니다.

```
aws ec2 describe-regions \
    --filters "Name=endpoint,Values=*us*"
```

출력:

```
{
    "Regions": [
        {
            "Endpoint": "ec2.us-east-1.amazonaws.com",
            "RegionName": "us-east-1"
        },
        {
            "Endpoint": "ec2.us-east-2.amazonaws.com",
            "RegionName": "us-east-2"
        },
        {
            "Endpoint": "ec2.us-west-1.amazonaws.com",
            "RegionName": "us-west-1"
        },
        {
            "Endpoint": "ec2.us-west-2.amazonaws.com",
            "RegionName": "us-west-2"
        }
    ]
}
```

자세한 내용은 Amazon EC2 사용 설명서에서 리전 및 가용 영역을 참조하세요.

예제 3: 모든 리전을 설명하는 방법

다음 `describe-regions` 예제에서는 비활성화된 리전을 포함하여 사용 가능한 모든 리전을 설명합니다.

```
aws ec2 describe-regions \
    --all-regions
```

출력:

```
{
```

```
    "Regions": [
        {
            "Endpoint": "ec2.eu-north-1.amazonaws.com",
            "RegionName": "eu-north-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-south-1.amazonaws.com",
            "RegionName": "ap-south-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-west-3.amazonaws.com",
            "RegionName": "eu-west-3",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-west-2.amazonaws.com",
            "RegionName": "eu-west-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-west-1.amazonaws.com",
            "RegionName": "eu-west-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-northeast-3.amazonaws.com",
            "RegionName": "ap-northeast-3",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.me-south-1.amazonaws.com",
            "RegionName": "me-south-1",
            "OptInStatus": "not-opted-in"
        },
        {
            "Endpoint": "ec2.ap-northeast-2.amazonaws.com",
            "RegionName": "ap-northeast-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-northeast-1.amazonaws.com",
            "RegionName": "ap-northeast-1",
```

```
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.sa-east-1.amazonaws.com",
            "RegionName": "sa-east-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ca-central-1.amazonaws.com",
            "RegionName": "ca-central-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-east-1.amazonaws.com",
            "RegionName": "ap-east-1",
            "OptInStatus": "not-opted-in"
        },
        {
            "Endpoint": "ec2.ap-southeast-1.amazonaws.com",
            "RegionName": "ap-southeast-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.ap-southeast-2.amazonaws.com",
            "RegionName": "ap-southeast-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.eu-central-1.amazonaws.com",
            "RegionName": "eu-central-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-east-1.amazonaws.com",
            "RegionName": "us-east-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-east-2.amazonaws.com",
            "RegionName": "us-east-2",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-west-1.amazonaws.com",
```

```
            "RegionName": "us-west-1",
            "OptInStatus": "opt-in-not-required"
        },
        {
            "Endpoint": "ec2.us-west-2.amazonaws.com",
            "RegionName": "us-west-2",
            "OptInStatus": "opt-in-not-required"
        }
    ]
}
```

자세한 내용은 Amazon EC2 사용 설명서에서 리전 및 가용 영역을 참조하세요.

예제 4: 리전 이름만 나열하는 방법

다음 `describe-regions` 예제에서는 `--query` 파라미터를 사용하여 출력을 필터링하고 리전 이름만 텍스트로 반환합니다.

```
aws ec2 describe-regions \
    --all-regions \
    --query "Regions[].{Name:RegionName}" \
    --output text
```

출력:

```
eu-north-1
ap-south-1
eu-west-3
eu-west-2
eu-west-1
ap-northeast-3
ap-northeast-2
me-south-1
ap-northeast-1
sa-east-1
ca-central-1
ap-east-1
ap-southeast-1
ap-southeast-2
eu-central-1
us-east-1
us-east-2
us-west-1
```

```
us-west-2
```

자세한 내용은 Amazon EC2 사용 설명서에서 리전 및 가용 영역을 참조하세요.

- API 세부 정보는 AWS CLI 명령 DescribeRegions참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { DescribeRegionsCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new DescribeRegionsCommand({
    // By default this command will not show regions that require you to opt-in.
    // When AllRegions true even the regions that require opt-in will be
 returned.
    AllRegions: true,
    // You can omit the Filters property if you want to get all regions.
    Filters: [
      {
        Name: "region-name",
        // You can specify multiple values for a filter.
        // You can also use '*' as a wildcard. This will return all
        // of the regions that start with `us-east-`.
        Values: ["ap-southeast-4"],
      },
    ],
  });

  try {
    const { Regions } = await client.send(command);
    const regionsList = Regions.map((reg) => ` • ${reg.RegionName}`);
    console.log("Found regions:");
```

```
    console.log(regionsList.join("\n"));
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API DescribeRegions참조를 참조하십시오.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예에서는 사용할 수 있는 지역을 설명합니다.

```
Get-EC2Region
```

출력:

```
Endpoint                            RegionName
--------                            ----------
ec2.eu-west-1.amazonaws.com         eu-west-1
ec2.ap-southeast-1.amazonaws.com    ap-southeast-1
ec2.ap-southeast-2.amazonaws.com    ap-southeast-2
ec2.eu-central-1.amazonaws.com      eu-central-1
ec2.ap-northeast-1.amazonaws.com    ap-northeast-1
ec2.us-east-1.amazonaws.com         us-east-1
ec2.sa-east-1.amazonaws.com         sa-east-1
ec2.us-west-1.amazonaws.com         us-west-1
ec2.us-west-2.amazonaws.com         us-west-2
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeRegions.

Ruby

SDK for Ruby

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
require "aws-sdk-ec2"

# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @example
#   list_regions_endpoints(Aws::EC2::Client.new(region: 'us-west-2'))
def list_regions_endpoints(ec2_client)
  result = ec2_client.describe_regions
  # Enable pretty printing.
  max_region_string_length = 16
  max_endpoint_string_length = 33
  # Print header.
  print "Region"
  print " " * (max_region_string_length - "Region".length)
  print "  Endpoint\n"
  print "-" * max_region_string_length
  print "  "
  print "-" * max_endpoint_string_length
  print "\n"
  # Print Regions and their endpoints.
  result.regions.each do |region|
    print region.region_name
    print " " * (max_region_string_length - region.region_name.length)
    print "  "
    print region.endpoint
    print "\n"
  end
end

# Displays a list of Amazon Elastic Compute Cloud (Amazon EC2)
# Availability Zones available to you depending on the AWS Region
# of the Amazon EC2 client.
```

```
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @example
#   list_availability_zones(Aws::EC2::Client.new(region: 'us-west-2'))
def list_availability_zones(ec2_client)
  result = ec2_client.describe_availability_zones
  # Enable pretty printing.
  max_region_string_length = 16
  max_zone_string_length = 18
  max_state_string_length = 9
  # Print header.
  print "Region"
  print " " * (max_region_string_length - "Region".length)
  print "  Zone"
  print " " * (max_zone_string_length - "Zone".length)
  print "  State\n"
  print "-" * max_region_string_length
  print "  "
  print "-" * max_zone_string_length
  print "  "
  print "-" * max_state_string_length
  print "\n"
  # Print Regions, Availability Zones, and their states.
  result.availability_zones.each do |zone|
    print zone.region_name
    print " " * (max_region_string_length - zone.region_name.length)
    print "  "
    print zone.zone_name
    print " " * (max_zone_string_length - zone.zone_name.length)
    print "  "
    print zone.state
    # Print any messages for this Availability Zone.
    if zone.messages.count.positive?
      print "\n"
      puts "  Messages for this zone:"
      zone.messages.each do |message|
        print "    #{message.message}\n"
      end
    end
    print "\n"
  end
end

# Example usage:
```

```
def run_me
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:   ruby ec2-ruby-example-regions-availability-zones.rb REGION"
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    puts "Example: ruby ec2-ruby-example-regions-availability-zones.rb us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  elsif ARGV.count.zero?
    region = "us-west-2"
  # Otherwise, use the values as specified at the command prompt.
  else
    region = ARGV[0]
  end

  ec2_client = Aws::EC2::Client.new(region: region)

  puts "AWS Regions for Amazon EC2 that are available to you:"
  list_regions_endpoints(ec2_client)
  puts "\n\nAmazon EC2 Availability Zones that are available to you for AWS
 Region '#{region}':"
  list_availability_zones(ec2_client)
end

run_me if $PROGRAM_NAME == __FILE__
```

- API 세부 정보는 AWS SDK for Ruby API DescribeRegions참조를 참조하십시오.

Rust

SDK for Rust

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
async fn show_regions(client: &Client) -> Result<(), Error> {
```

```
    let rsp = client.describe_regions().send().await?;

    println!("Regions:");
    for region in rsp.regions() {
        println!("  {}", region.region_name().unwrap());
    }

    Ok(())
}
```

- API에 대한 자세한 내용은 Rust용AWS SDK API 레퍼런스를 참조하십시오 DescribeRegions.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> 자세한 내용은 여기에서 확인할 수 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    TRY.
        oo_result = lo_ec2->describeregions( ) .                        "
 oo_result is returned for testing purposes. "
        DATA(lt_regions) = oo_result->get_regions( ).
        MESSAGE 'Retrieved information about Regions.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- API 세부 정보는 SAP용AWS SDK ABAP API 참조를 참조하십시오 DescribeRegions.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeRouteTables** CLI와 함께 사용

다음 코드 예제는 DescribeRouteTables의 사용 방법을 보여줍니다.

CLI

AWS CLI

라우팅 테이블을 설명하려면

다음 describe-route-tables 예제는 라우팅 테이블에 대한 세부 정보를 검색합니다.

```
aws ec2 describe-route-tables
```

출력:

```
{
    "RouteTables": [
        {
            "Associations": [
                {
                    "Main": true,
                    "RouteTableAssociationId": "rtbassoc-0df3f54e06EXAMPLE",
                    "RouteTableId": "rtb-09ba434c1bEXAMPLE"
                }
            ],
            "PropagatingVgws": [],
            "RouteTableId": "rtb-09ba434c1bEXAMPLE",
            "Routes": [
                {
                    "DestinationCidrBlock": "10.0.0.0/16",
                    "GatewayId": "local",
                    "Origin": "CreateRouteTable",
                    "State": "active"
                },
                {
                    "DestinationCidrBlock": "0.0.0.0/0",
                    "NatGatewayId": "nat-06c018cbd8EXAMPLE",
                    "Origin": "CreateRoute",
```

```
                    "State": "blackhole"
                }
            ],
            "Tags": [],
            "VpcId": "vpc-0065acced4EXAMPLE",
            "OwnerId": "111122223333"
        },
        {
            "Associations": [
                {
                    "Main": true,
                    "RouteTableAssociationId": "rtbassoc-9EXAMPLE",
                    "RouteTableId": "rtb-a1eec7de"
                }
            ],
            "PropagatingVgws": [],
            "RouteTableId": "rtb-a1eec7de",
            "Routes": [
                {
                    "DestinationCidrBlock": "172.31.0.0/16",
                    "GatewayId": "local",
                    "Origin": "CreateRouteTable",
                    "State": "active"
                },
                {
                    "DestinationCidrBlock": "0.0.0.0/0",
                    "GatewayId": "igw-fEXAMPLE",
                    "Origin": "CreateRoute",
                    "State": "active"
                }
            ],
            "Tags": [],
            "VpcId": "vpc-3EXAMPLE",
            "OwnerId": "111122223333"
        },
        {
            "Associations": [
                {
                    "Main": false,
                    "RouteTableAssociationId": "rtbassoc-0b100c28b2EXAMPLE",
                    "RouteTableId": "rtb-07a98f76e5EXAMPLE",
                    "SubnetId": "subnet-0d3d002af8EXAMPLE"
                }
            ],
```

```
            "PropagatingVgws": [],
            "RouteTableId": "rtb-07a98f76e5EXAMPLE",
            "Routes": [
                {
                    "DestinationCidrBlock": "10.0.0.0/16",
                    "GatewayId": "local",
                    "Origin": "CreateRouteTable",
                    "State": "active"
                },
                {
                    "DestinationCidrBlock": "0.0.0.0/0",
                    "GatewayId": "igw-06cf664d80EXAMPLE",
                    "Origin": "CreateRoute",
                    "State": "active"
                }
            ],
            "Tags": [],
            "VpcId": "vpc-0065acced4EXAMPLE",
            "OwnerId": "111122223333"
        }
    ]
}
```

자세한 내용은 AWS VPC 사용 설명서의 라우팅 테이블 작업을 참조하십시오.

- API 세부 정보는 AWS CLI 명령 DescribeRouteTables참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제에서는 모든 라우팅 테이블을 설명합니다.

```
Get-EC2RouteTable
```

출력:

```
DestinationCidrBlock    : 10.0.0.0/16
DestinationPrefixListId :
GatewayId               : local
InstanceId              :
InstanceOwnerId         :
```

```
NetworkInterfaceId      :
Origin                  : CreateRouteTable
State                   : active
VpcPeeringConnectionId  :

DestinationCidrBlock    : 0.0.0.0/0
DestinationPrefixListId :
GatewayId               : igw-1a2b3c4d
InstanceId              :
InstanceOwnerId         :
NetworkInterfaceId      :
Origin                  : CreateRoute
State                   : active
VpcPeeringConnectionId  :
```

예 2: 이 예제는 지정된 라우팅 테이블의 세부 정보를 반환합니다.

```
Get-EC2RouteTable -RouteTableId rtb-1a2b3c4d
```

예 3: 이 예제에서는 지정된 VPC의 라우팅 테이블을 설명합니다.

```
Get-EC2RouteTable -Filter @{ Name="vpc-id"; Values="vpc-1a2b3c4d" }
```

출력:

```
Associations    : {rtbassoc-12345678}
PropagatingVgws : {}
Routes          : {, }
RouteTableId    : rtb-1a2b3c4d
Tags            : {}
VpcId           : vpc-1a2b3c4d
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeRouteTables.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeScheduledInstanceAvailability** CLI와 함께 사용

다음 코드 예제는 DescribeScheduledInstanceAvailability의 사용 방법을 보여줍니다.

CLI

AWS CLI

이용 가능한 일정을 설명하려면

이 예제에서는 매주 일요일에 지정된 날짜에 시작되는 일정을 설명합니다.

명령:

```
aws ec2 describe-scheduled-instance-availability --recurrence
 Frequency=Weekly,Interval=1,OccurrenceDays=[1] --first-slot-start-time-range
 EarliestTime=2016-01-31T00:00:00Z,LatestTime=2016-01-31T04:00:00Z
```

출력:

```
{
  "ScheduledInstanceAvailabilitySet": [
    {
        "AvailabilityZone": "us-west-2b",
        "TotalScheduledInstanceHours": 1219,
        "PurchaseToken": "eyJ2IjoiMSIsInMiOjEsImMiOi...",
        "MinTermDurationInDays": 366,
        "AvailableInstanceCount": 20,
        "Recurrence": {
            "OccurrenceDaySet": [
                1
            ],
            "Interval": 1,
            "Frequency": "Weekly",
            "OccurrenceRelativeToEnd": false
        },
        "Platform": "Linux/UNIX",
        "FirstSlotStartTime": "2016-01-31T00:00:00Z",
        "MaxTermDurationInDays": 366,
        "SlotDurationInHours": 23,
        "NetworkPlatform": "EC2-VPC",
```

```
        "InstanceType": "c4.large",
        "HourlyPrice": "0.095"
    },
    ...
  ]
}
```

결과 범위를 좁히기 위해 운영 체제, 네트워크 및 인스턴스 유형을 지정하는 필터를 추가할 수 있습니다.

명령:

--필터 이름=플랫폼, 값=리눅스/유닉스 이름=네트워크 플랫폼, 값=EC2-VPC 이름=인스턴스 유형, 값=C4.large

- API에 대한 세부 정보는 명령 참조를 참조하십시오. DescribeScheduledInstanceAvailabilityAWS CLI

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 매주 일요일에 지정된 날짜에 시작되는 일정을 설명합니다.

```
Get-EC2ScheduledInstanceAvailability -Recurrence_Frequency
 Weekly -Recurrence_Interval 1 -Recurrence_OccurrenceDay 1 -
FirstSlotStartTimeRange_EarliestTime 2016-01-31T00:00:00Z -
FirstSlotStartTimeRange_LatestTime 2016-01-31T04:00:00Z
```

출력:

```
AvailabilityZone            : us-west-2b
AvailableInstanceCount      : 20
FirstSlotStartTime          : 1/31/2016 8:00:00 AM
HourlyPrice                 : 0.095
InstanceType                : c4.large
MaxTermDurationInDays       : 366
MinTermDurationInDays       : 366
NetworkPlatform             : EC2-VPC
Platform                    : Linux/UNIX
PurchaseToken               : eyJ2IjoiMSIsInMiOjEsImMiOi...
Recurrence                  : Amazon.EC2.Model.ScheduledInstanceRecurrence
```

```
SlotDurationInHours         : 23
TotalScheduledInstanceHours : 1219

...
```

예 2: 결과 범위를 좁히기 위해 운영 체제, 네트워크, 인스턴스 유형과 같은 기준에 대한 필터를 추가할 수 있습니다.

```
-Filter @{ Name="platform";Values="Linux/UNIX" },@{ Name="network-
platform";Values="EC2-VPC" },@{ Name="instance-type";Values="c4.large" }
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeScheduledInstanceAvailability.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeScheduledInstances** CLI와 함께 사용

다음 코드 예제는 DescribeScheduledInstances의 사용 방법을 보여줍니다.

CLI

AWS CLI

정기 인스턴스에 대해 설명하려면

이 예제에서는 지정된 정기 인스턴스를 설명합니다.

명령:

```
aws ec2 describe-scheduled-instances --scheduled-instance-ids
 sci-1234-1234-1234-1234-123456789012
```

출력:

```
{
  "ScheduledInstanceSet": [
      {
```

```
            "AvailabilityZone": "us-west-2b",
            "ScheduledInstanceId": "sci-1234-1234-1234-1234-123456789012",
            "HourlyPrice": "0.095",
            "CreateDate": "2016-01-25T21:43:38.612Z",
            "Recurrence": {
                "OccurrenceDaySet": [
                    1
                ],
                "Interval": 1,
                "Frequency": "Weekly",
                "OccurrenceRelativeToEnd": false,
                "OccurrenceUnit": ""
            },
            "Platform": "Linux/UNIX",
            "TermEndDate": "2017-01-31T09:00:00Z",
            "InstanceCount": 1,
            "SlotDurationInHours": 32,
            "TermStartDate": "2016-01-31T09:00:00Z",
            "NetworkPlatform": "EC2-VPC",
            "TotalScheduledInstanceHours": 1696,
            "NextSlotStartTime": "2016-01-31T09:00:00Z",
            "InstanceType": "c4.large"
        }
    ]
}
```

이 예제에서는 모든 정기 인스턴스를 설명합니다.

명령:

```
aws ec2 describe-scheduled-instances
```

- API 세부 정보는 AWS CLI 명령 DescribeScheduledInstances참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제에서는 지정된 예약 인스턴스에 대해 설명합니다.

```
Get-EC2ScheduledInstance -ScheduledInstanceId
 sci-1234-1234-1234-1234-123456789012
```

출력:

```
AvailabilityZone            : us-west-2b
CreateDate                  : 1/25/2016 1:43:38 PM
HourlyPrice                 : 0.095
InstanceCount               : 1
InstanceType                : c4.large
NetworkPlatform             : EC2-VPC
NextSlotStartTime           : 1/31/2016 1:00:00 AM
Platform                    : Linux/UNIX
PreviousSlotEndTime         :
Recurrence                  : Amazon.EC2.Model.ScheduledInstanceRecurrence
ScheduledInstanceId         : sci-1234-1234-1234-1234-123456789012
SlotDurationInHours         : 32
TermEndDate                 : 1/31/2017 1:00:00 AM
TermStartDate               : 1/31/2016 1:00:00 AM
TotalScheduledInstanceHours : 1696
```

예 2: 이 예제에서는 모든 정기 인스턴스를 설명합니다.

```
Get-EC2ScheduledInstance
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet DescribeScheduledInstances참조의 내용을 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeSecurityGroups** CLI와 함께 사용

다음 코드 예제는 DescribeSecurityGroups의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 인스턴스 시작하기

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Retrieve information for an Amazon EC2 security group.
    /// </summary>
    /// <param name="groupId">The Id of the Amazon EC2 security group.</param>
    /// <returns>A list of security group information.</returns>
    public async Task<List<SecurityGroup>> DescribeSecurityGroups(string groupId)
    {
        var request = new DescribeSecurityGroupsRequest();
        var groupIds = new List<string> { groupId };
        request.GroupIds = groupIds;

        var response = await _amazonEC2.DescribeSecurityGroupsAsync(request);
        return response.SecurityGroups;
    }

    /// <summary>
    /// Display the information returned by the call to
    /// DescribeSecurityGroupsAsync.
    /// </summary>
    /// <param name="securityGroup">A list of security group information.</param>
    public void DisplaySecurityGroupInfoAsync(SecurityGroup securityGroup)
    {
        Console.WriteLine($"{securityGroup.GroupName}");
        Console.WriteLine("Ingress permissions:");
        securityGroup.IpPermissions.ForEach(permission =>
        {
            Console.WriteLine($"\tFromPort: {permission.FromPort}");
            Console.WriteLine($"\tIpProtocol: {permission.IpProtocol}");

            Console.Write($"\tIpv4Ranges: ");
            permission.Ipv4Ranges.ForEach(range =>
{ Console.Write($"{range.CidrIp} "); });
```

```
            Console.WriteLine($"\n\tIpv6Ranges:");
            permission.Ipv6Ranges.ForEach(range =>
{ Console.Write($"{range.CidrIpv6} "); });

            Console.Write($"\n\tPrefixListIds: ");
            permission.PrefixListIds.ForEach(id => Console.Write($"{id.Id} "));

            Console.WriteLine($"\n\tTo Port: {permission.ToPort}");
        });
        Console.WriteLine("Egress permissions:");
        securityGroup.IpPermissionsEgress.ForEach(permission =>
        {
            Console.WriteLine($"\tFromPort: {permission.FromPort}");
            Console.WriteLine($"\tIpProtocol: {permission.IpProtocol}");

            Console.Write($"\tIpv4Ranges: ");
            permission.Ipv4Ranges.ForEach(range =>
{ Console.Write($"{range.CidrIp} "); });

            Console.WriteLine($"\n\tIpv6Ranges:");
            permission.Ipv6Ranges.ForEach(range =>
{ Console.Write($"{range.CidrIpv6} "); });

            Console.Write($"\n\tPrefixListIds: ");
            permission.PrefixListIds.ForEach(id => Console.Write($"{id.Id} "));

            Console.WriteLine($"\n\tTo Port: {permission.ToPort}");
        });
    }
```

- API 세부 정보는 AWS SDK for .NET API DescribeSecurityGroups참조를 참조하십시오.

Bash

AWS CLI Bash 스크립트 사용

> **Note**
>
> 더 많은 정보가 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
#############################################################################
# function ec2_describe_security_groups
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 security groups.
#
# Parameters:
#       -g security_group_id - The ID of the security group to describe
 (optional).
#
# And:
#       0 - If successful.
#       1 - If it fails.
#############################################################################
function ec2_describe_security_groups() {
  local security_group_id response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_security_groups"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 security groups."
    echo "  -g security_group_id - The ID of the security group to describe
 (optional)."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "g:h" option; do
    case "${option}" in
      g) security_group_id="${OPTARG}" ;;
```

```
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local query="SecurityGroups[*].[GroupName, GroupId, VpcId, IpPermissions[*].
[IpProtocol, FromPort, ToPort, IpRanges[*].CidrIp]]"

  if [[ -n "$security_group_id" ]]; then
    response=$(aws ec2 describe-security-groups --group-ids "$security_group_id"
 --query "${query}" --output text)
  else
    response=$(aws ec2 describe-security-groups --query "${query}" --output text)
  fi

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports describe-security-groups operation failed.
$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

이 예제에 사용된 유틸리티 함수

```
###############################################################################
# function errecho
#
```

```
# This function outputs everything sent to it to STDERR (standard error output).
#########################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#########################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
#########################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- API 세부 정보는 AWS CLI 명령 DescribeSecurityGroups참조를 참조하십시오.

C++

SDK for C++

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::DescribeSecurityGroupsRequest request;

    if (!groupID.empty()) {
        request.AddGroupIds(groupID);
    }

    Aws::String nextToken;
    do {
        if (!nextToken.empty()) {
            request.SetNextToken(nextToken);
        }

        auto outcome = ec2Client.DescribeSecurityGroups(request);
        if (outcome.IsSuccess()) {
            std::cout << std::left <<
                      std::setw(32) << "Name" <<
                      std::setw(30) << "GroupId" <<
                      std::setw(30) << "VpcId" <<
                      std::setw(64) << "Description" << std::endl;

            const std::vector<Aws::EC2::Model::SecurityGroup> &securityGroups =
                    outcome.GetResult().GetSecurityGroups();

            for (const auto &securityGroup: securityGroups) {
                std::cout << std::left <<
                          std::setw(32) << securityGroup.GetGroupName() <<
                          std::setw(30) << securityGroup.GetGroupId() <<
                          std::setw(30) << securityGroup.GetVpcId() <<
                          std::setw(64) << securityGroup.GetDescription() <<
                          std::endl;
            }
```

```
        }
        else {
            std::cerr << "Failed to describe security groups:" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        nextToken = outcome.GetResult().GetNextToken();
    } while (!nextToken.empty());
```

- API 세부 정보는 AWS SDK for C++ API DescribeSecurityGroups참조를 참조하십시오.

CLI

AWS CLI

예제 1: 보안 그룹 설명하는 방법

다음 `describe-security-groups` 예제에서는 지정된 보안 그룹을 설명합니다.

```
aws ec2 describe-security-groups \
    --group-ids sg-903004f8
```

출력:

```
{
    "SecurityGroups": [
        {
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "-1",
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "UserIdGroupPairs": [],
                    "PrefixListIds": []
                }
            ],
            "Description": "My security group",
            "Tags": [
```

```
                {
                    "Value": "SG1",
                    "Key": "Name"
                }
            ],
            "IpPermissions": [
                {
                    "IpProtocol": "-1",
                    "IpRanges": [],
                    "UserIdGroupPairs": [
                        {
                            "UserId": "123456789012",
                            "GroupId": "sg-903004f8"
                        }
                    ],
                    "PrefixListIds": []
                },
                {
                    "PrefixListIds": [],
                    "FromPort": 22,
                    "IpRanges": [
                        {
                            "Description": "Access from NY office",
                            "CidrIp": "203.0.113.0/24"
                        }
                    ],
                    "ToPort": 22,
                    "IpProtocol": "tcp",
                    "UserIdGroupPairs": []
                    }
            ],
            "GroupName": "MySecurityGroup",
            "VpcId": "vpc-1a2b3c4d",
            "OwnerId": "123456789012",
            "GroupId": "sg-903004f8",
        }
    ]
}
```

예제 2: 특정 규칙이 있는 보안 그룹을 설명하는 방법

다음 describe-security-groups 예제에서는 필터를 사용하여 SSH 트래픽을 허용하는 규칙 (포트 22) 과 모든 주소의 트래픽을 허용하는 규칙 () 이 있는 보안 그룹으로 결과 범위를 지정

합니다0.0.0.0/0. 이 예제에서는 --query 파라미터를 사용하여 보안 그룹의 이름만 표시합니다. 보안 그룹이 결과에 반환될 모든 필터와 일치해야 하지만 단일 규칙이 모든 필터와 일치할 필요는 없습니다. 예를 들어 출력은 특정 IP 주소의 SSH 트래픽을 허용하는 규칙과 모든 주소의 HTTP 트래픽을 허용하는 다른 규칙이 포함된 보안 그룹을 반환합니다.

```
aws ec2 describe-security-groups \
    --filters Name=ip-permission.from-port,Values=22 Name=ip-permission.to-
port,Values=22 Name=ip-permission.cidr,Values='0.0.0.0/0' \
    --query "SecurityGroups[*].[GroupName]" \
    --output text
```

출력:

```
default
my-security-group
web-servers
launch-wizard-1
```

예제 3: 태그를 기반으로 보안 그룹을 설명하는 방법

다음 describe-security-groups 예제에서는 필터를 사용하여 결과 범위를 보안 그룹 이름에 test가 포함되고 Test=To-delete 태그가 있는 보안 그룹으로 지정합니다. 이 예제에서는 --query 파라미터를 사용하여 보안 그룹의 이름 및 ID만 표시합니다.

```
aws ec2 describe-security-groups \
    --filters Name=group-name,Values=*test* Name=tag:Test,Values=To-delete \
    --query "SecurityGroups[*].{Name:GroupName,ID:GroupId}"
```

출력:

```
[
    {
        "Name": "testfornewinstance",
        "ID": "sg-33bb22aa"
    },
    {
        "Name": "newgrouptest",
        "ID": "sg-1a2b3c4d"
    }
]
```

태그 필터를 사용하는 추가 예제는 Amazon EC2 사용 설명서에서 태그 작업을 참조하세요.

- API 세부 정보는 AWS CLI 명령 DescribeSecurityGroups참조를 참조하십시오.

Java

SDK for Java 2.x

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
   public static void describeSecurityGroups(Ec2Client ec2, String groupId) {
       try {
           DescribeSecurityGroupsRequest request =
DescribeSecurityGroupsRequest.builder()
               .groupIds(groupId)
               .build();

           // Use a paginator.
           DescribeSecurityGroupsIterable listGroups =
ec2.describeSecurityGroupsPaginator(request);
           listGroups.stream()
               .flatMap(r -> r.securityGroups().stream())
               .forEach(group -> System.out
                   .println(" Group id: " +group.groupId() + " group name = " +
group.groupName()));

       } catch (Ec2Exception e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

- API 세부 정보는 AWS SDK for Java 2.x API DescribeSecurityGroups참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { DescribeSecurityGroupsCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

// Log the details of a specific security group.
export const main = async () => {
  const command = new DescribeSecurityGroupsCommand({
    GroupIds: ["SECURITY_GROUP_ID"],
  });

  try {
    const { SecurityGroups } = await client.send(command);
    console.log(JSON.stringify(SecurityGroups, null, 2));
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API DescribeSecurityGroups참조를 참조하십시오.

Kotlin

SDK for Kotlin

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
suspend fun describeEC2SecurityGroups(groupId: String) {
    val request =
        DescribeSecurityGroupsRequest {
            groupIds = listOf(groupId)
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->

        val response = ec2.describeSecurityGroups(request)
        response.securityGroups?.forEach { group ->
            println("Found Security Group with id ${group.groupId}, vpc id
 ${group.vpcId} and description ${group.description}")
        }
    }
}
```

- API 세부 정보는 Kotlin API용AWS SDK 레퍼런스를 참조하세요 DescribeSecurityGroups.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예제에서는 VPC의 지정된 보안 그룹을 설명합니다. VPC에 속한 보안 그룹을 사용할 때는 이름 (- 파라미터) 이 아닌 보안 그룹 ID (- GroupId 파라미터) 를 사용하여 그룹을 참조해야 합니다. GroupName

```
Get-EC2SecurityGroup -GroupId sg-12345678
```

출력:

```
Description         : default VPC security group
GroupId             : sg-12345678
GroupName           : default
IpPermissions       : {Amazon.EC2.Model.IpPermission}
IpPermissionsEgress : {Amazon.EC2.Model.IpPermission}
OwnerId             : 123456789012
Tags                : {}
VpcId               : vpc-12345678
```

예 2: 이 예제에서는 EC2-Classic에 지정된 보안 그룹을 설명합니다. EC2-Classic용 보안 그룹을 사용할 때는 그룹 이름 (- GroupName 파라미터) 또는 그룹 ID (- GroupId 파라미터) 를 사용하여 보안 그룹을 참조할 수 있습니다.

```
Get-EC2SecurityGroup -GroupName my-security-group
```

출력:

```
Description         : my security group
GroupId             : sg-45678901
GroupName           : my-security-group
IpPermissions       : {Amazon.EC2.Model.IpPermission,
 Amazon.EC2.Model.IpPermission}
IpPermissionsEgress : {}
OwnerId             : 123456789012
Tags                : {}
VpcId               :
```

예 3: 이 예에서는 vpc-0fc1ff23456b789eb에 대한 모든 보안 그룹을 검색합니다.

```
Get-EC2SecurityGroup -Filter @{Name="vpc-id";Values="vpc-0fc1ff23456b789eb"}
```

- API에 대한AWS Tools for PowerShell 세부 정보는 Cmdlet 참조를 참조하십시오. DescribeSecurityGroups

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class SecurityGroupWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) security group
 actions."""

    def __init__(self, ec2_resource, security_group=None):
```

```
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param security_group: A Boto3 SecurityGroup object. This is a high-level
object
                               that wraps security group actions.
        """
        self.ec2_resource = ec2_resource
        self.security_group = security_group

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def describe(self):
        """
        Displays information about the security group.
        """
        if self.security_group is None:
            logger.info("No security group to describe.")
            return

        try:
            print(f"Security group: {self.security_group.group_name}")
            print(f"\tID: {self.security_group.id}")
            print(f"\tVPC: {self.security_group.vpc_id}")
            if self.security_group.ip_permissions:
                print(f"Inbound permissions:")
                pp(self.security_group.ip_permissions)
        except ClientError as err:
            logger.error(
                "Couldn't get data for security group %s. Here's why: %s: %s",
                self.security_group.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 DescribeSecurityGroups.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    TRY.
        DATA lt_group_ids TYPE /aws1/
cl_ec2groupidstrlist_w=>tt_groupidstringlist.
        APPEND NEW /aws1/cl_ec2groupidstrlist_w( iv_value = iv_group_id ) TO
 lt_group_ids.
        oo_result = lo_ec2->describesecuritygroups( it_groupids = lt_group_ids ).
         " oo_result is returned for testing purposes. "
        DATA(lt_security_groups) = oo_result->get_securitygroups( ).
        MESSAGE 'Retrieved information about security groups.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- API 세부 정보는 SAP용AWS SDK ABAP API 참조를 참조하십시오 DescribeSecurityGroups.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeSnapshotAttribute** CLI와 함께 사용

다음 코드 예제는 DescribeSnapshotAttribute의 사용 방법을 보여줍니다.

CLI

AWS CLI

스냅샷의 스냅샷 속성을 설명하려면

다음 `describe-snapshot-attribute` 예제에는 스냅샷이 공유되는 계정이 나열되어 있습니다.

```
aws ec2 describe-snapshot-attribute \
    --snapshot-id snap-01234567890abcedf \
    --attribute createVolumePermission
```

출력:

```
{
    "SnapshotId": "snap-01234567890abcedf",
    "CreateVolumePermissions": [
        {
            "UserId": "123456789012"
        }
    ]
}
```

자세한 내용은 Amazon Elastic Compute 클라우드 사용 설명서의 Amazon EBS 스냅샷 공유를 참조하십시오.

- API 세부 정보는 AWS CLI 명령 DescribeSnapshotAttribute참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제에서는 지정된 스냅샷의 지정된 속성을 설명합니다.

```
Get-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute ProductCodes
```

출력:

```
CreateVolumePermissions    ProductCodes    SnapshotId
-----------------------    ------------    ----------
```

```
{}                          {}              snap-12345678
```

예 2: 이 예제에서는 지정된 스냅샷의 지정된 속성을 설명합니다.

```
(Get-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute
 CreateVolumePermission).CreateVolumePermissions
```

출력:

```
Group    UserId
-----    ------
all
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeSnapshotAttribute.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeSnapshots** CLI와 함께 사용

다음 코드 예제는 DescribeSnapshots의 사용 방법을 보여줍니다.

CLI

AWS CLI

예제 1: 스냅샷을 설명하는 방법

다음 describe-snapshots 예제에서는 지정된 스냅샷을 설명합니다.

```
aws ec2 describe-snapshots \
    --snapshot-ids snap-1234567890abcdef0
```

출력:

```
{
    "Snapshots": [
        {
```

```
            "Description": "This is my snapshot",
            "Encrypted": false,
            "VolumeId": "vol-049df61146c4d7901",
            "State": "completed",
            "VolumeSize": 8,
            "StartTime": "2019-02-28T21:28:32.000Z",
            "Progress": "100%",
            "OwnerId": "012345678910",
            "SnapshotId": "snap-01234567890abcdef",
            "Tags": [
                {
                    "Key": "Stack",
                    "Value": "test"
                }
            ]
        }
    ]
}
```

자세한 내용은 Amazon EC2 사용 설명서에서 Amazon EBS 스냅샷을 참조하세요.

예제 2: 필터를 기반으로 스냅샷을 설명하는 방법

다음 `describe-snapshots` 예제에서는 필터를 사용하여 해당 `pending` 주에 있는 AWS 계정이 소유한 스냅샷으로 결과 범위를 지정합니다. 이 예제에서는 `--query` 파라미터를 사용하여 스냅샷 ID 및 스냅샷이 시작된 시간만 표시합니다.

```
aws ec2 describe-snapshots \
    --owner-ids self \
    --filters Name=status,Values=pending \
    --query "Snapshots[*].{ID:SnapshotId,Time:StartTime}"
```

출력:

```
[
    {
        "ID": "snap-1234567890abcdef0",
        "Time": "2019-08-04T12:48:18.000Z"
    },
    {
        "ID": "snap-066877671789bd71b",
        "Time": "2019-08-04T02:45:16.000Z
```

```
    },
    ...
]
```

다음 `describe-snapshots` 예제에서는 필터를 사용하여 결과 범위를 지정된 리전에서 생성된 스냅샷으로 지정합니다. 이 예제에서는 `--query` 파라미터를 사용하여 스냅샷 ID만 표시합니다.

```
aws ec2 describe-snapshots \
    --filters Name=volume-id,Values=049df61146c4d7901 \
    --query "Snapshots[*].[SnapshotId]" \
    --output text
```

출력:

```
snap-1234567890abcdef0
snap-08637175a712c3fb9
...
```

필터를 사용하는 추가 예제는 Amazon EC2 사용 설명서에서 리소스 나열 및 필터링을 참조하세요.

예제 3: 태그를 기반으로 스냅샷을 설명하는 방법

다음 `describe-snapshots` 예제에서는 태그 필터를 사용하여 결과 범위를 `Stack=Prod` 태그가 있는 스냅샷으로 지정합니다.

```
aws ec2 describe-snapshots \
    --filters Name=tag:Stack,Values=prod
```

`describe-snapshots` 출력 예제는 예제 1을 참조하세요.

태그 필터를 사용하는 추가 예제는 Amazon EC2 사용 설명서에서 태그 작업을 참조하세요.

예제 4: 수명에 기반하여 스냅샷을 설명하는 방법

다음 `describe-snapshots` 예제에서는 JMESPath 식을 사용하여 지정된 날짜 이전에 AWS 계정에서 만든 모든 스냅샷을 설명합니다. 스냅샷 ID만 표시합니다.

```
aws ec2 describe-snapshots \
    --owner-ids 012345678910 \
```

```
    --query "Snapshots[?(StartTime<='2020-03-31')].[SnapshotId]"
```

필터를 사용하는 추가 예제는 Amazon EC2 사용 설명서에서 리소스 나열 및 필터링을 참조하세요.

예제 5: 아카이브된 스냅샷만 보는 방법

다음 `describe-snapshots` 예제에서는 아카이브 티어에 저장된 스냅샷만 나열합니다.

```
aws ec2 describe-snapshots \
    --filters "Name=storage-tier,Values=archive"
```

출력:

```
{
    "Snapshots": [
        {
            "Description": "Snap A",
            "Encrypted": false,
            "VolumeId": "vol-01234567890aaaaaa",
            "State": "completed",
            "VolumeSize": 8,
            "StartTime": "2021-09-07T21:00:00.000Z",
            "Progress": "100%",
            "OwnerId": "123456789012",
            "SnapshotId": "snap-01234567890aaaaaa",
            "StorageTier": "archive",
            "Tags": []
        },
    ]
}
```

자세한 내용을 알아보려면 Amazon Elastic Compute Cloud 사용 설명서에서 인스턴스 유형을 참조하세요.

- API 세부 정보는 명령 참조를 참조하십시오 DescribeSnapshots.AWS CLI

PowerShell

에 대한 도구 PowerShell

예 1: 이 예제에서는 지정된 스냅샷을 설명합니다.

```
Get-EC2Snapshot -SnapshotId snap-12345678
```

출력:

```
DataEncryptionKeyId :
Description         : Created by CreateImage(i-1a2b3c4d) for ami-12345678 from
 vol-12345678
Encrypted           : False
KmsKeyId            :
OwnerAlias          :
OwnerId             : 123456789012
Progress            : 100%
SnapshotId          : snap-12345678
StartTime           : 10/23/2014 6:01:28 AM
State               : completed
StateMessage        :
Tags                : {}
VolumeId            : vol-12345678
VolumeSize          : 8
```

예 2: 이 예에서는 'Name' 태그가 있는 스냅샷을 설명합니다.

```
Get-EC2Snapshot | ? { $_.Tags.Count -gt 0 -and $_.Tags.Key -eq "Name" }
```

예 3: 이 예에서는 'Name' 태그에 값이 'TestValue'인 스냅샷을 설명합니다.

```
Get-EC2Snapshot | ? { $_.Tags.Count -gt 0 -and $_.Tags.Key -eq "Name" -and
 $_.Tags.Value -eq "TestValue" }
```

예 4: 이 예에서는 모든 스냅샷을 설명합니다.

```
Get-EC2Snapshot -Owner self
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeSnapshots.

Rust

SDK for Rust

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

스냅샷의 상태를 보여 줍니다.

```
async fn show_state(client: &Client, id: &str) -> Result<(), Error> {
    let resp = client
        .describe_snapshots()
        .filters(Filter::builder().name("snapshot-id").values(id).build())
        .send()
        .await?;

    println!(
        "State: {}",
        resp.snapshots().first().unwrap().state().unwrap().as_ref()
    );

    Ok(())
}
```

```
async fn show_snapshots(client: &Client) -> Result<(), Error> {
    // "self" represents your account ID.
    // You can list the snapshots for any account by replacing
    // "self" with that account ID.
    let resp = client.describe_snapshots().owner_ids("self").send().await?;
    let snapshots = resp.snapshots();
    let length = snapshots.len();

    for snapshot in snapshots {
        println!(
            "ID:          {}",
            snapshot.snapshot_id().unwrap_or_default()
        );
        println!(
```

```
            "Description: {}",
            snapshot.description().unwrap_or_default()
        );
        println!("State:       {}", snapshot.state().unwrap().as_ref());
        println!();
    }

    println!();
    println!("Found {} snapshot(s)", length);
    println!();

    Ok(())
}
```

- API에 대한 자세한 내용은 Rust용AWS SDK API 레퍼런스를 참조하십시오 DescribeSnapshots.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeSpotDatafeedSubscription** CLI와 함께 사용

다음 코드 예제는 DescribeSpotDatafeedSubscription의 사용 방법을 보여줍니다.

CLI

AWS CLI

계정에 대한 스팟 인스턴스 데이터 피드 구독을 설명하려면

이 예제 명령은 계정의 데이터 피드를 설명합니다.

명령:

```
aws ec2 describe-spot-datafeed-subscription
```

출력:

```
{
    "SpotDatafeedSubscription": {
```

```
        "OwnerId": "123456789012",
        "Prefix": "spotdata",
        "Bucket": "my-s3-bucket",
        "State": "Active"
    }
}
```

- API 세부 정보는 AWS CLI 명령 DescribeSpotDatafeedSubscription참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 스팟 인스턴스 데이터 피드에 대해 설명합니다.

```
Get-EC2SpotDatafeedSubscription
```

출력:

```
Bucket  : my-s3-bucket
Fault   :
OwnerId : 123456789012
Prefix  : spotdata
State   : Active
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeSpotDatafeedSubscription.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeSpotFleetInstances** CLI와 함께 사용

다음 코드 예제는 DescribeSpotFleetInstances의 사용 방법을 보여줍니다.

CLI

AWS CLI

스팟 플릿과 연결된 스팟 인스턴스를 설명하려면

이 예제 명령은 지정된 스팟 플릿과 연결된 스팟 인스턴스를 나열합니다.

명령:

```
aws ec2 describe-spot-fleet-instances --spot-fleet-request-id sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE
```

출력:

```
{
  "ActiveInstances": [
      {
          "InstanceId": "i-1234567890abcdef0",
          "InstanceType": "m3.medium",
          "SpotInstanceRequestId": "sir-08b93456"
      },
      ...
  ],
  "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE"
}
```

- API 세부 정보는 AWS CLI 명령 DescribeSpotFleetInstances참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 지정된 스팟 플릿 요청과 연결된 인스턴스를 설명합니다.

```
Get-EC2SpotFleetInstance -SpotFleetRequestId sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE
```

출력:

```
InstanceId    InstanceType    SpotInstanceRequestId
----------    ------------    ---------------------
i-f089262a    c3.large        sir-12345678
i-7e8b24a4    c3.large        sir-87654321
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeSpotFleetInstances.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeSpotFleetRequestHistory** CLI와 함께 사용

다음 코드 예제는 DescribeSpotFleetRequestHistory의 사용 방법을 보여줍니다.

CLI

AWS CLI

스팟 플릿 기록을 설명하려면

이 예제 명령은 지정된 시간부터 시작하여 지정된 스팟 플릿에 대한 기록을 반환합니다.

명령:

```
aws ec2 describe-spot-fleet-request-history --spot-fleet-request-id sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE --start-time 2015-05-26T00:00:00Z
```

다음 예제 출력은 스팟 플릿에 대한 두 스팟 인스턴스의 성공적인 시작을 보여줍니다.

출력:

```
{
  "HistoryRecords": [
      {
          "Timestamp": "2015-05-26T23:17:20.697Z",
          "EventInformation": {
              "EventSubType": "submitted"
          },
          "EventType": "fleetRequestChange"
      },
      {
          "Timestamp": "2015-05-26T23:17:20.873Z",
          "EventInformation": {
              "EventSubType": "active"
          },
          "EventType": "fleetRequestChange"
      },
      {
```

```
            "Timestamp": "2015-05-26T23:21:21.712Z",
            "EventInformation": {
                "InstanceId": "i-1234567890abcdef0",
                "EventSubType": "launched"
            },
            "EventType": "instanceChange"
        },
        {
            "Timestamp": "2015-05-26T23:21:21.816Z",
            "EventInformation": {
                "InstanceId": "i-1234567890abcdef1",
                "EventSubType": "launched"
            },
            "EventType": "instanceChange"
        }
    ],
    "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
    "NextToken": "CpHNsscimcV5oH7bSbub03CI2Qms5+ypNpNm
+53MNlR0YcXAkp0xFlfKf91yVxSExmbtma3awYxMFzNA663ZskT0AHtJ6TCb2Z8bQC2EnZgyELbymtWPfpZ1ZbauV
+P+TfGlWxWWB/Vr5dk5d4LfdgA/DRAHUrYgxzrEXAMPLE=",
    "StartTime": "2015-05-26T00:00:00Z"
}
```

- API 세부 정보는 AWS CLI 명령 DescribeSpotFleetRequestHistory참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 지정된 스팟 플릿 요청의 기록을 설명합니다.

```
Get-EC2SpotFleetRequestHistory -SpotFleetRequestId sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE -StartTime 2015-12-26T00:00:00Z
```

출력:

```
HistoryRecords     : {Amazon.EC2.Model.HistoryRecord,
 Amazon.EC2.Model.HistoryRecord...}
LastEvaluatedTime  : 12/26/2015 8:29:11 AM
NextToken          :
SpotFleetRequestId : sfr-088bc5f1-7e7b-451a-bd13-757f10672b93
StartTime          : 12/25/2015 8:00:00 AM
```

```
(Get-EC2SpotFleetRequestHistory -SpotFleetRequestId sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE -StartTime 2015-12-26T00:00:00Z).HistoryRecords
```

출력:

```
EventInformation                    EventType            Timestamp
----------------                    ---------            ---------
Amazon.EC2.Model.EventInformation   fleetRequestChange   12/26/2015 8:23:33 AM
Amazon.EC2.Model.EventInformation   fleetRequestChange   12/26/2015 8:23:33 AM
Amazon.EC2.Model.EventInformation   fleetRequestChange   12/26/2015 8:23:33 AM
Amazon.EC2.Model.EventInformation   launched             12/26/2015 8:25:34 AM
Amazon.EC2.Model.EventInformation   launched             12/26/2015 8:25:05 AM
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeSpotFleetRequestHistory.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeSpotFleetRequests** CLI와 함께 사용

다음 코드 예제는 DescribeSpotFleetRequests의 사용 방법을 보여줍니다.

CLI

AWS CLI

스팟 플릿 요청을 설명하려면

이 예에서는 모든 스팟 플릿 요청을 설명합니다.

명령:

```
aws ec2 describe-spot-fleet-requests
```

출력:

```
{
  "SpotFleetRequestConfigs": [
      {
```

```
            "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
            "SpotFleetRequestConfig": {
                "TargetCapacity": 20,
                "LaunchSpecifications": [
                    {
                        "EbsOptimized": false,
                        "NetworkInterfaces": [
                            {
                                "SubnetId": "subnet-a61dafcf",
                                "DeviceIndex": 0,
                                "DeleteOnTermination": false,
                                "AssociatePublicIpAddress": true,
                                "SecondaryPrivateIpAddressCount": 0
                            }
                        ],
                        "InstanceType": "cc2.8xlarge",
                        "ImageId": "ami-1a2b3c4d"
                    },
                    {
                        "EbsOptimized": false,
                        "NetworkInterfaces": [
                            {
                                "SubnetId": "subnet-a61dafcf",
                                "DeviceIndex": 0,
                                "DeleteOnTermination": false,
                                "AssociatePublicIpAddress": true,
                                "SecondaryPrivateIpAddressCount": 0
                            }
                        ],
                        "InstanceType": "r3.8xlarge",
                        "ImageId": "ami-1a2b3c4d"
                    }
                ],
                "SpotPrice": "0.05",
                "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role"
            },
            "SpotFleetRequestState": "active"
        },
        {
            "SpotFleetRequestId": "sfr-306341ed-9739-402e-881b-ce47bEXAMPLE",
            "SpotFleetRequestConfig": {
                "TargetCapacity": 20,
                "LaunchSpecifications": [
                    {
```

```
                    "EbsOptimized": false,
                    "NetworkInterfaces": [
                        {
                            "SubnetId": "subnet-6e7f829e",
                            "DeviceIndex": 0,
                            "DeleteOnTermination": false,
                            "AssociatePublicIpAddress": true,
                            "SecondaryPrivateIpAddressCount": 0
                        }
                    ],
                    "InstanceType": "m3.medium",
                    "ImageId": "ami-1a2b3c4d"
                }
            ],
            "SpotPrice": "0.05",
            "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role"
        },
        "SpotFleetRequestState": "active"
    }
  ]
}
```

스팟 플릿 요청을 설명하려면

이 예제에서는 지정된 스팟 플릿 요청을 설명합니다.

명령:

```
aws ec2 describe-spot-fleet-requests --spot-fleet-request-ids sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE
```

출력:

```
{
  "SpotFleetRequestConfigs": [
    {
        "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
        "SpotFleetRequestConfig": {
            "TargetCapacity": 20,
            "LaunchSpecifications": [
                {
                    "EbsOptimized": false,
```

```
                    "NetworkInterfaces": [
                        {
                            "SubnetId": "subnet-a61dafcf",
                            "DeviceIndex": 0,
                            "DeleteOnTermination": false,
                            "AssociatePublicIpAddress": true,
                            "SecondaryPrivateIpAddressCount": 0
                        }
                    ],
                    "InstanceType": "cc2.8xlarge",
                    "ImageId": "ami-1a2b3c4d"
                },
                {
                    "EbsOptimized": false,
                    "NetworkInterfaces": [
                        {
                            "SubnetId": "subnet-a61dafcf",
                            "DeviceIndex": 0,
                            "DeleteOnTermination": false,
                            "AssociatePublicIpAddress": true,
                            "SecondaryPrivateIpAddressCount": 0
                        }
                    ],
                    "InstanceType": "r3.8xlarge",
                    "ImageId": "ami-1a2b3c4d"
                }
            ],
            "SpotPrice": "0.05",
            "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role"
        },
        "SpotFleetRequestState": "active"
    }
  ]
}
```

- API 세부 정보는 AWS CLI 명령 DescribeSpotFleetRequests참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 지정된 스팟 플릿 요청을 설명합니다.

```
Get-EC2SpotFleetRequest -SpotFleetRequestId sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE | format-list
```

출력:

```
ConfigData            : Amazon.EC2.Model.SpotFleetRequestConfigData
CreateTime            : 12/26/2015 8:23:33 AM
SpotFleetRequestId    : sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
SpotFleetRequestState : active
```

예 2: 이 예에서는 모든 스팟 플릿 요청을 설명합니다.

```
Get-EC2SpotFleetRequest
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeSpotFleetRequests.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeSpotInstanceRequests** CLI와 함께 사용

다음 코드 예제는 DescribeSpotInstanceRequests의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 스팟 인스턴스 요청을 설명하려면

다음 describe-spot-instance-requests 예에서는 지정된 스팟 인스턴스 요청을 설명합니다.

```
aws ec2 describe-spot-instance-requests \
    --spot-instance-request-ids sir-08b93456
```

출력:

```
{
```

```
    "SpotInstanceRequests": [
        {
            "CreateTime": "2018-04-30T18:14:55.000Z",
            "InstanceId": "i-1234567890abcdef1",
            "LaunchSpecification": {
                "InstanceType": "t2.micro",
                "ImageId": "ami-003634241a8fcdec0",
                "KeyName": "my-key-pair",
                "SecurityGroups": [
                    {
                        "GroupName": "default",
                        "GroupId": "sg-e38f24a7"
                    }
                ],
                "BlockDeviceMappings": [
                    {
                        "DeviceName": "/dev/sda1",
                        "Ebs": {
                            "DeleteOnTermination": true,
                            "SnapshotId": "snap-0e54a519c999adbbd",
                            "VolumeSize": 8,
                            "VolumeType": "standard",
                            "Encrypted": false
                        }
                    }
                ],
                "NetworkInterfaces": [
                    {
                        "DeleteOnTermination": true,
                        "DeviceIndex": 0,
                        "SubnetId": "subnet-049df61146c4d7901"
                    }
                ],
                "Placement": {
                    "AvailabilityZone": "us-east-2b",
                    "Tenancy": "default"
                },
                "Monitoring": {
                    "Enabled": false
                }
            },
            "LaunchedAvailabilityZone": "us-east-2b",
            "ProductDescription": "Linux/UNIX",
            "SpotInstanceRequestId": "sir-08b93456",
```

```
            "SpotPrice": "0.010000"
            "State": "active",
            "Status": {
                "Code": "fulfilled",
                "Message": "Your Spot request is fulfilled.",
                "UpdateTime": "2018-04-30T18:16:21.000Z"
            },
            "Tags": [],
            "Type": "one-time",
            "InstanceInterruptionBehavior": "terminate"
        }
    ]
}
```

예 2: 필터를 기반으로 스팟 인스턴스 요청을 설명하려면

다음 `describe-spot-instance-requests` 예제에서는 필터를 사용하여 지정된 가용 영역에서 지정된 인스턴스 유형의 스팟 인스턴스 요청으로 결과 범위를 지정합니다. 이 예제에서는 `--query` 파라미터를 사용하여 인스턴스 ID만 표시합니다.

```
aws ec2 describe-spot-instance-requests \
    --filters Name=launch.instance-type,Values=m3.medium Name=launched-
availability-zone,Values=us-east-2a \
    --query "SpotInstanceRequests[*].[InstanceId]" \
    --output text
```

출력:

```
i-057750d42936e468a
i-001efd250faaa6ffa
i-027552a73f021f3bd
...
```

필터를 사용하는 추가 예제는 Amazon Elastic Compute Cloud 사용 설명서의 리소스 나열 및 필터링을 참조하십시오.

예 3: 태그를 기반으로 스팟 인스턴스 요청을 설명하려면

다음 `describe-spot-instance-requests` 예제에서는 태그 필터를 사용하여 해당 태그가 있는 스팟 인스턴스 요청으로 결과 범위를 지정합니다`cost-center=cc123`.

```
aws ec2 describe-spot-instance-requests \
```

```
    --filters Name=tag:cost-center,Values=cc123
```

describe-spot-instance-requests 출력 예제는 예제 1을 참조하세요.

태그 필터를 사용하는 추가 예제는 Amazon EC2 사용 설명서에서 태그 작업을 참조하세요.

- API 세부 정보는 AWS CLI 명령 DescribeSpotInstanceRequests참조를 참조하십시오.

PowerShell

에 대한 도구 PowerShell

예 1: 이 예에서는 지정된 스팟 인스턴스 요청을 설명합니다.

```
Get-EC2SpotInstanceRequest -SpotInstanceRequestId sir-12345678
```

출력:

```
ActualBlockHourlyPrice   :
AvailabilityZoneGroup    :
BlockDurationMinutes     : 0
CreateTime               : 4/8/2015 2:51:33 PM
Fault                    :
InstanceId               : i-12345678
LaunchedAvailabilityZone : us-west-2b
LaunchGroup              :
LaunchSpecification      : Amazon.EC2.Model.LaunchSpecification
ProductDescription       : Linux/UNIX
SpotInstanceRequestId    : sir-12345678
SpotPrice                : 0.020000
State                    : active
Status                   : Amazon.EC2.Model.SpotInstanceStatus
Tags                     : {Name}
Type                     : one-time
```

예 2: 이 예에서는 모든 스팟 인스턴스 요청을 설명합니다.

```
Get-EC2SpotInstanceRequest
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeSpotInstanceRequests.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeSpotPriceHistory** CLI와 함께 사용

다음 코드 예제는 DescribeSpotPriceHistory의 사용 방법을 보여줍니다.

CLI

AWS CLI

스팟 가격 기록을 설명하려면

이 예제 명령은 1월 특정 날짜의 m1.xlarge 인스턴스에 대한 스팟 가격 기록을 반환합니다.

명령:

```
aws ec2 describe-spot-price-history --instance-types m1.xlarge --start-time
 2014-01-06T07:08:09 --end-time 2014-01-06T08:09:10
```

출력:

```
{
  "SpotPriceHistory": [
          {
              "Timestamp": "2014-01-06T07:10:55.000Z",
              "ProductDescription": "SUSE Linux",
              "InstanceType": "m1.xlarge",
              "SpotPrice": "0.087000",
              "AvailabilityZone": "us-west-1b"
          },
          {
              "Timestamp": "2014-01-06T07:10:55.000Z",
              "ProductDescription": "SUSE Linux",
              "InstanceType": "m1.xlarge",
              "SpotPrice": "0.087000",
              "AvailabilityZone": "us-west-1c"
          },
          {
              "Timestamp": "2014-01-06T05:42:36.000Z",
              "ProductDescription": "SUSE Linux (Amazon VPC)",
```

```
                "InstanceType": "m1.xlarge",
                "SpotPrice": "0.087000",
                "AvailabilityZone": "us-west-1a"
        },
        ...
}
```

리눅스/유닉스 Amazon VPC의 스팟 가격 기록을 설명하려면

이 예제 명령은 1월의 특정 날짜에 대한 m1.xlarge, Linux/UNIX Amazon VPC 인스턴스의 스팟 가격 기록을 반환합니다.

명령:

```
aws ec2 describe-spot-price-history --instance-types m1.xlarge --product-
description "Linux/UNIX (Amazon VPC)" --start-time 2014-01-06T07:08:09 --end-time
 2014-01-06T08:09:10
```

출력:

```
{
  "SpotPriceHistory": [
      {
          "Timestamp": "2014-01-06T04:32:53.000Z",
          "ProductDescription": "Linux/UNIX (Amazon VPC)",
          "InstanceType": "m1.xlarge",
          "SpotPrice": "0.080000",
          "AvailabilityZone": "us-west-1a"
      },
      {
          "Timestamp": "2014-01-05T11:28:26.000Z",
          "ProductDescription": "Linux/UNIX (Amazon VPC)",
          "InstanceType": "m1.xlarge",
          "SpotPrice": "0.080000",
          "AvailabilityZone": "us-west-1c"
      }
  ]
}
```

- API 세부 정보는 명령 참조를 참조하십시오. DescribeSpotPriceHistoryAWS CLI

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 인스턴스 유형 및 가용 영역에 대한 스팟 가격 기록의 최근 10개 항목을 가져옵니다. 참고로 - AvailabilityZone 매개 변수에 지정된 값은 cmdlet의 -Region 매개 변수(예에는 표시되지 않음) 에 제공된 지역 값에 유효하거나 셸에서 기본값으로 설정되어 있어야 합니다. 이 예제 명령은 환경에 기본 지역인 'us-west-2'가 설정되어 있다고 가정합니다.

```
Get-EC2SpotPriceHistory -InstanceType c3.large -AvailabilityZone us-west-2a -
MaxResult 10
```

출력:

```
AvailabilityZone   : us-west-2a
InstanceType       : c3.large
Price              : 0.017300
ProductDescription : Linux/UNIX (Amazon VPC)
Timestamp          : 12/25/2015 7:39:49 AM

AvailabilityZone   : us-west-2a
InstanceType       : c3.large
Price              : 0.017200
ProductDescription : Linux/UNIX (Amazon VPC)
Timestamp          : 12/25/2015 7:38:29 AM

AvailabilityZone   : us-west-2a
InstanceType       : c3.large
Price              : 0.017300
ProductDescription : Linux/UNIX (Amazon VPC)
Timestamp          : 12/25/2015 6:57:13 AM
...
```

- API에 대한 자세한 내용은 Cmdlet 참조를 참조하십시오 DescribeSpotPriceHistory.AWS Tools for PowerShell

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeSubnets** CLI와 함께 사용

다음 코드 예제는 DescribeSubnets의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 복원력이 뛰어난 서비스 구축 및 관리

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Get all the subnets for a Vpc in a set of availability zones.
    /// </summary>
    /// <param name="vpcId">The Id of the Vpc.</param>
    /// <param name="availabilityZones">The list of availability zones.</param>
    /// <returns>The collection of subnet objects.</returns>
    public async Task<List<Subnet>> GetAllVpcSubnetsForZones(string vpcId,
List<string> availabilityZones)
    {
        var subnets = new List<Subnet>();
        var subnetPaginator = _amazonEc2.Paginators.DescribeSubnets(
            new DescribeSubnetsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("vpc-id", new List<string>() { vpcId}),
                    new ("availability-zone", availabilityZones),
                    new ("default-for-az", new List<string>() { "true" })
                }
            });

        // Get the entire list using the paginator.
```

```
        await foreach (var subnet in subnetPaginator.Subnets)
        {
            subnets.Add(subnet);
        }

        return subnets;
    }
```

- API 세부 정보는 AWS SDK for .NET API DescribeSubnets참조를 참조하십시오.

CLI

AWS CLI

예제 1: 모든 서브넷을 설명하는 방법

다음 `describe-subnets` 예제에서는 서브넷의 세부 정보를 표시합니다.

```
aws ec2 describe-subnets
```

출력:

```
{
    "Subnets": [
        {
            "AvailabilityZone": "us-east-1d",
            "AvailabilityZoneId": "use1-az2",
            "AvailableIpAddressCount": 4089,
            "CidrBlock": "172.31.80.0/20",
            "DefaultForAz": true,
            "MapPublicIpOnLaunch": false,
            "MapCustomerOwnedIpOnLaunch": true,
            "State": "available",
            "SubnetId": "subnet-0bb1c79de3EXAMPLE",
            "VpcId": "vpc-0ee975135dEXAMPLE",
            "OwnerId": "111122223333",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": [],
            "CustomerOwnedIpv4Pool:": 'pool-2EXAMPLE',
            "SubnetArn": "arn:aws:ec2:us-east-2:111122223333:subnet/
subnet-0bb1c79de3EXAMPLE",
```

```
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            }
        },
        {
            "AvailabilityZone": "us-east-1d",
            "AvailabilityZoneId": "use1-az2",
            "AvailableIpAddressCount": 4089,
            "CidrBlock": "172.31.80.0/20",
            "DefaultForAz": true,
            "MapPublicIpOnLaunch": true,
            "MapCustomerOwnedIpOnLaunch": false,
            "State": "available",
            "SubnetId": "subnet-8EXAMPLE",
            "VpcId": "vpc-3EXAMPLE",
            "OwnerId": "1111222233333",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": [],
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "MySubnet"
                }
            ],
            "SubnetArn": "arn:aws:ec2:us-east-1:111122223333:subnet/
subnet-8EXAMPLE",
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            }
        }
    ]
}
```


자세한 내용은 AWS VPC 사용 설명서에서 VPC 및 서브넷 작업을 참조하세요.

예제 2: 특정 VPC의 서브넷을 설명하는 방법

다음 describe-subnets 예제에서는 필터를 사용하여 지정된 VPC의 서브넷에 대한 세부 정보를 검색합니다.

```
aws ec2 describe-subnets \
    --filters "Name=vpc-id,Values=vpc-3EXAMPLE"
```

출력:

```
{
    "Subnets": [
        {
            "AvailabilityZone": "us-east-1d",
            "AvailabilityZoneId": "use1-az2",
            "AvailableIpAddressCount": 4089,
            "CidrBlock": "172.31.80.0/20",
            "DefaultForAz": true,
            "MapPublicIpOnLaunch": true,
            "MapCustomerOwnedIpOnLaunch": false,
            "State": "available",
            "SubnetId": "subnet-8EXAMPLE",
            "VpcId": "vpc-3EXAMPLE",
            "OwnerId": "1111222233333",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": [],
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "MySubnet"
                }
            ],
            "SubnetArn": "arn:aws:ec2:us-east-1:111122223333:subnet/
subnet-8EXAMPLE",
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            }
        }
    ]
}
```

자세한 내용은 AWS VPC 사용 설명서에서 VPC 및 서브넷 작업을 참조하세요.

예제 3: 특정 태그의 서브넷을 설명하는 방법

다음 `describe-subnets` 예제에서는 필터를 사용하여 `CostCenter=123` 태그가 있는 해당 서브넷 세부 정보를 검색하고 `--query` 파라미터를 사용하여 이 태그가 있는 서브넷의 서브넷 ID를 표시합니다.

```
aws ec2 describe-subnets \
    --filters "Name=tag:CostCenter,Values=123" \
    --query "Subnets[*].SubnetId" \
    --output text
```

출력:

```
subnet-0987a87c8b37348ef
subnet-02a95061c45f372ee
subnet-03f720e7de2788d73
```

자세한 내용은 Amazon VPC 사용 설명서에서 VPC 및 서브넷 작업을 참조하세요.

- API 세부 정보는 AWS CLI 명령 DescribeSubnets참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    const client = new EC2Client({});
    const { Subnets } = await client.send(
      new DescribeSubnetsCommand({
        Filters: [
          { Name: "vpc-id", Values: [state.defaultVpc] },
          { Name: "availability-zone", Values: state.availabilityZoneNames },
          { Name: "default-for-az", Values: ["true"] },
        ],
```

```
    }),
  );
```

- API 세부 정보는 AWS SDK for JavaScript API DescribeSubnets참조를 참조하십시오.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예에서는 지정된 서브넷을 설명합니다.

```
Get-EC2Subnet -SubnetId subnet-1a2b3c4d
```

출력:

```
AvailabilityZone        : us-west-2c
AvailableIpAddressCount : 251
CidrBlock               : 10.0.0.0/24
DefaultForAz            : False
MapPublicIpOnLaunch     : False
State                   : available
SubnetId                : subnet-1a2b3c4d
Tags                    : {}
VpcId                   : vpc-12345678
```

예 2: 이 예에서는 모든 서브넷을 설명합니다.

```
Get-EC2Subnet
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeSubnets.

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"
```

```
    def get_subnets(self, vpc_id, zones):
        """
        Gets the default subnets in a VPC for a specified list of Availability
Zones.

        :param vpc_id: The ID of the VPC to look up.
        :param zones: The list of Availability Zones to look up.
        :return: The list of subnets found.
        """
        try:
            response = self.ec2_client.describe_subnets(
                Filters=[
                    {"Name": "vpc-id", "Values": [vpc_id]},
                    {"Name": "availability-zone", "Values": zones},
                    {"Name": "default-for-az", "Values": ["true"]},
                ]
            )
            subnets = response["Subnets"]
            log.info("Found %s subnets for the specified zones.", len(subnets))
        except ClientError as err:
            raise AutoScalerError(f"Couldn't get subnets: {err}")
        else:
            return subnets
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 DescribeSubnets.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeTags** CLI와 함께 사용

다음 코드 예제는 DescribeTags의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 단일 리소스의 모든 태그를 설명하려면

다음 describe-tags 예제는 지정된 인스턴스의 태그를 설명합니다.

```
aws ec2 describe-tags \
    --filters "Name=resource-id,Values=i-1234567890abcdef8"
```

출력:

```
{
    "Tags": [
        {
            "ResourceType": "instance",
            "ResourceId": "i-1234567890abcdef8",
            "Value": "Test",
            "Key": "Stack"
        },
        {
            "ResourceType": "instance",
            "ResourceId": "i-1234567890abcdef8",
            "Value": "Beta Server",
            "Key": "Name"
        }
    ]
}
```

예 2: 리소스 유형의 모든 태그 설명하기

다음 describe-tags 예제는 볼륨의 태그를 설명합니다.

```
aws ec2 describe-tags \
    --filters "Name=resource-type,Values=volume"
```

출력:

```
{
    "Tags": [
```

```
        {
            "ResourceType": "volume",
            "ResourceId": "vol-1234567890abcdef0",
            "Value": "Project1",
            "Key": "Purpose"
        },
        {
            "ResourceType": "volume",
            "ResourceId": "vol-049df61146c4d7901",
            "Value": "Logs",
            "Key": "Purpose"
        }
    ]
}
```

예 3: 모든 태그를 설명하려면

다음 `describe-tags` 예시는 모든 리소스의 태그를 설명합니다.

```
aws ec2 describe-tags
```

예 4: 태그 키를 기반으로 리소스의 태그를 설명하려면

다음 `describe-tags` 예시는 키와 함께 태그가 있는 리소스의 태그를 설명합니다`Stack`.

```
aws ec2 describe-tags \
    --filters Name=key,Values=Stack
```

출력:

```
{
    "Tags": [
        {
            "ResourceType": "volume",
            "ResourceId": "vol-027552a73f021f3b",
            "Value": "Production",
            "Key": "Stack"
        },
        {
            "ResourceType": "instance",
            "ResourceId": "i-1234567890abcdef8",
```

```
            "Value": "Test",
            "Key": "Stack"
        }
    ]
}
```

예 5: 태그 키와 태그 값을 기반으로 리소스의 태그를 설명하려면

다음 `describe-tags` 예시는 태그가 있는 리소스의 태그를 설명합니다`Stack=Test`.

```
aws ec2 describe-tags \
    --filters Name=key,Values=Stack Name=value,Values=Test
```

출력:

```
{
    "Tags": [
        {
            "ResourceType": "image",
            "ResourceId": "ami-3ac336533f021f3bd",
            "Value": "Test",
            "Key": "Stack"
        },
        {
            "ResourceType": "instance",
            "ResourceId": "i-1234567890abcdef8",
            "Value": "Test",
            "Key": "Stack"
        }
    ]
}
```

다음 `describe-tags` 예시에서는 대체 구문을 사용하여 태그가 있는 리소스를 설명합니다`Stack=Test`.

```
aws ec2 describe-tags \
    --filters "Name=tag:Stack,Values=Test"
```

다음 `describe-tags` 예제는 키는 `Purpose` 있지만 값은 없는 태그가 있는 모든 인스턴스의 태그를 설명합니다.

```
aws ec2 describe-tags \
    --filters "Name=resource-type,Values=instance" "Name=key,Values=Purpose"
 "Name=value,Values="
```

출력:

```
{
    "Tags": [
        {
            "ResourceType": "instance",
            "ResourceId": "i-1234567890abcdef5",
            "Value": null,
            "Key": "Purpose"
        }
    ]
}
```

- API 세부 정보는 AWS CLI 명령 DescribeTags참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 리소스 유형 '이미지'의 태그를 가져옵니다.

```
Get-EC2Tag -Filter @{Name="resource-type";Values="image"}
```

출력:

```
Key         ResourceId            ResourceType Value
---         ----------            ------------ -----
Name        ami-0a123b4ccb567a8ea image        Win7-Imported
auto-delete ami-0a123b4ccb567a8ea image        never
```

예 2: 이 예제는 모든 리소스의 모든 태그를 가져와서 리소스 유형별로 그룹화합니다.

```
Get-EC2Tag | Group-Object resourcetype
```

출력:

```
Count Name                      Group
----- ----                      -----
    9 subnet                    {Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription...}
   53 instance                  {Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription...}
    3 route-table               {Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription}
    5 security-group            {Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription...}
   30 volume                    {Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription...}
    1 internet-gateway          {Amazon.EC2.Model.TagDescription}
    3 network-interface         {Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription}
    4 elastic-ip                {Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription}
    1 dhcp-options              {Amazon.EC2.Model.TagDescription}
    2 image                     {Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription}
    3 vpc                       {Amazon.EC2.Model.TagDescription,
 Amazon.EC2.Model.TagDescription, Amazon.EC2.Model.TagDescription}
```

예 3: 이 예제는 지정된 지역에 대해 'auto-delete' 태그가 있고 값이 'no'인 모든 리소스를 표시합니다.

```
Get-EC2Tag -Region eu-west-1 -Filter @{Name="tag:auto-delete";Values="no"}
```

출력:

```
Key         ResourceId            ResourceType Value
---         ----------            ------------ -----
auto-delete i-0f1bce234d5dd678b   instance     no
auto-delete vol-01d234aa5678901a2 volume       no
auto-delete vol-01234bfb5def6f7b8 volume       no
auto-delete vol-01ccb23f4c5e67890 volume       no
```

예제 4: 이 예제는 값이 'no' 인 'auto-delete'라는 태그가 있는 모든 리소스를 가져오고 다음 파이프에서 필터를 추가하여 '인스턴스' 리소스 유형만 파싱하고 최종적으로 값이 인스턴스 ID 자체인 각 인스턴스 리소스에 대해 ThisInstance " 태그를 생성합니다.

```
Get-EC2Tag -Region eu-west-1 -Filter @{Name="tag:auto-delete";Values="no"}
 | Where-Object ResourceType -eq "instance" | ForEach-Object {New-EC2Tag -
ResourceId $_.ResourceId -Tag @{Key="ThisInstance";Value=$_.ResourceId}}
```

예제 5: 이 예제는 모든 인스턴스 리소스의 태그와 'Name' 키를 가져와서 테이블 형식으로 표시합니다.

```
Get-EC2Tag -Filter @{Name="resource-
type";Values="instance"},@{Name="key";Values="Name"} | Select-Object ResourceId,
 @{Name="Name-Tag";Expression={$PSItem.Value}} | Format-Table -AutoSize
```

출력:

```
ResourceId          Name-Tag
----------          --------
i-012e3cb4df567e1aa jump1
i-01c23a45d6fc7a89f repro-3
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeTags.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeVolumeAttribute** CLI와 함께 사용

다음 코드 예제는 DescribeVolumeAttribute의 사용 방법을 보여줍니다.

CLI

AWS CLI

볼륨 속성을 설명하려면

이 예제 명령은 ID와 함께 볼륨의 autoEnableIo 속성을 설명합니다vol-049df61146c4d7901.

명령:

```
aws ec2 describe-volume-attribute --volume-id vol-049df61146c4d7901 --attribute
 autoEnableIO
```

출력:

```
{
    "AutoEnableIO": {
        "Value": false
    },
    "VolumeId": "vol-049df61146c4d7901"
}
```

- API 세부 정보는 AWS CLI 명령 DescribeVolumeAttribute참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 볼륨의 지정된 속성을 설명합니다.

```
Get-EC2VolumeAttribute -VolumeId vol-12345678 -Attribute AutoEnableIO
```

출력:

```
AutoEnableIO    ProductCodes    VolumeId
------------    ------------    --------
False           {}              vol-12345678
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeVolumeAttribute.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeVolumeStatus** CLI와 함께 사용

다음 코드 예제는 DescribeVolumeStatus의 사용 방법을 보여줍니다.

CLI

AWS CLI

단일 볼륨의 상태를 설명하려면

이 예제 명령은 볼륨의 상태를 설명합니다vol-1234567890abcdef0.

명령:

```
aws ec2 describe-volume-status --volume-ids vol-1234567890abcdef0
```

출력:

```
{
    "VolumeStatuses": [
        {
            "VolumeStatus": {
                "Status": "ok",
                "Details": [
                    {
                        "Status": "passed",
                        "Name": "io-enabled"
                    },
                    {
                        "Status": "not-applicable",
                        "Name": "io-performance"
                    }
                ]
            },
            "AvailabilityZone": "us-east-1a",
            "VolumeId": "vol-1234567890abcdef0",
            "Actions": [],
            "Events": []
        }
    ]
}
```

손상된 볼륨의 상태를 설명하려면

이 예제 명령은 손상된 모든 볼륨의 상태를 설명합니다. 이 예제 출력에는 손상된 볼륨이 없습니다.

명령:

```
aws ec2 describe-volume-status --filters Name=volume-
status.status,Values=impaired
```

출력:

```
{
    "VolumeStatuses": []
}
```

상태 확인에 실패한 볼륨 (상태가 손상됨) 이 있는 경우 Amazon EC2 사용 설명서의 손상된 볼륨 사용을 참조하십시오.

- API 세부 정보는 AWS CLI 명령 DescribeVolumeStatus참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 볼륨의 상태를 설명합니다.

```
Get-EC2VolumeStatus -VolumeId vol-12345678
```

출력:

```
Actions          : {}
AvailabilityZone : us-west-2a
Events           : {}
VolumeId         : vol-12345678
VolumeStatus     : Amazon.EC2.Model.VolumeStatusInfo
```

```
(Get-EC2VolumeStatus -VolumeId vol-12345678).VolumeStatus
```

출력:

```
Details                       Status
-------                       ------
```

```
{io-enabled, io-performance}    ok
```

```
(Get-EC2VolumeStatus -VolumeId vol-12345678).VolumeStatus.Details
```

출력:

```
Name                            Status
----                            ------
io-enabled                      passed
io-performance                  not-applicable
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeVolumeStatus.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeVolumes** CLI와 함께 사용

다음 코드 예제는 DescribeVolumes의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 볼륨을 설명하려면

다음 describe-volumes 예제는 현재 지역의 지정된 볼륨을 설명합니다.

```
aws ec2 describe-volumes \
    --volume-ids vol-049df61146c4d7901 vol-1234567890abcdef0
```

출력:

```
{
    "Volumes": [
        {
            "AvailabilityZone": "us-east-1a",
```

```
            "Attachments": [
                {
                    "AttachTime": "2013-12-18T22:35:00.000Z",
                    "InstanceId": "i-1234567890abcdef0",
                    "VolumeId": "vol-049df61146c4d7901",
                    "State": "attached",
                    "DeleteOnTermination": true,
                    "Device": "/dev/sda1"
                }
            ],
            "Encrypted": true,
            "KmsKeyId": "arn:aws:kms:us-east-2a:123456789012:key/8c5b2c63-
b9bc-45a3-a87a-5513eEXAMPLE,
            "VolumeType": "gp2",
            "VolumeId": "vol-049df61146c4d7901",
            "State": "in-use",
            "Iops": 100,
            "SnapshotId": "snap-1234567890abcdef0",
            "CreateTime": "2019-12-18T22:35:00.084Z",
            "Size": 8
        },
        {
            "AvailabilityZone": "us-east-1a",
            "Attachments": [],
            "Encrypted": false,
            "VolumeType": "gp2",
            "VolumeId": "vol-1234567890abcdef0",
            "State": "available",
            "Iops": 300,
            "SnapshotId": "",
            "CreateTime": "2020-02-27T00:02:41.791Z",
            "Size": 100
        }
    ]
}
```

예 2: 특정 인스턴스에 연결된 볼륨 설명하기

다음 describe-volumes 예제는 지정된 인스턴스에 연결되어 있고 인스턴스 종료 시 삭제되도록 설정된 모든 볼륨을 설명합니다.

```
aws ec2 describe-volumes \
    --region us-east-1 \
```

```
    --filters Name=attachment.instance-id,Values=i-1234567890abcdef0
 Name=attachment.delete-on-termination,Values=true
```

describe-volumes 출력 예제는 예제 1을 참조하세요.

예 3: 특정 가용 영역에서 사용 가능한 볼륨 설명하기

다음 describe-volumes 예제는 지정된 가용 영역 상태와 해당 가용 영역에 있는 모든 볼륨을 설명합니다. available

```
aws ec2 describe-volumes \
    --filters Name=status,Values=available Name=availability-zone,Values=us-
east-1a
```

describe-volumes 출력 예제는 예제 1을 참조하세요.

예 4: 태그를 기반으로 볼륨 설명하기

다음 describe-volumes 예제는 태그 Name 키와 로 시작하는 값이 있는 모든 볼륨을 설명합니다Test. 그런 다음 볼륨의 태그와 ID만 표시하는 쿼리로 출력을 필터링합니다.

```
aws ec2 describe-volumes \
    --filters Name=tag:Name,Values=Test* \
    --query "Volumes[*].{ID:VolumeId,Tag:Tags}"
```

출력:

```
[
    {
        "Tag": [
            {
                "Value": "Test2",
                "Key": "Name"
            }
        ],
        "ID": "vol-1234567890abcdef0"
    },
    {
        "Tag": [
            {
                "Value": "Test1",
```

```
                "Key": "Name"
            }
        ],
        "ID": "vol-049df61146c4d7901"
    }
]
```

태그 필터를 사용하는 추가 예제는 Amazon EC2 사용 설명서에서 태그 작업을 참조하세요.

- API 세부 정보는 AWS CLI 명령 DescribeVolumes참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 EBS 볼륨을 설명합니다.

```
Get-EC2Volume -VolumeId vol-12345678
```

출력:

```
Attachments      : {}
AvailabilityZone : us-west-2c
CreateTime       : 7/17/2015 4:35:19 PM
Encrypted        : False
Iops             : 90
KmsKeyId         :
Size             : 30
SnapshotId       : snap-12345678
State            : in-use
Tags             : {}
VolumeId         : vol-12345678
VolumeType       : standard
```

예 2: 이 예에서는 상태가 '사용 가능'인 EBS 볼륨을 설명합니다.

```
Get-EC2Volume -Filter @{ Name="status"; Values="available" }
```

출력:

```
Attachments      : {}
```

```
AvailabilityZone : us-west-2c
CreateTime       : 12/21/2015 2:31:29 PM
Encrypted        : False
Iops             : 60
KmsKeyId         :
Size             : 20
SnapshotId       : snap-12345678
State            : available
Tags             : {}
VolumeId         : vol-12345678
VolumeType       : gp2
...
```

예 3: 이 예에서는 모든 EBS 볼륨을 설명합니다.

```
Get-EC2Volume
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeVolumes.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeVpcAttribute** CLI와 함께 사용

다음 코드 예제는 DescribeVpcAttribute의 사용 방법을 보여줍니다.

CLI

AWS CLI

enableDnsSupport 속성을 설명하려면

이 예제에서는 enableDnsSupport 속성을 설명합니다. 이 속성은 VPC에 DNS 확인이 활성화되었는지 여부를 나타냅니다. 이 속성이 true 인 경우 Amazon DNS 서버는 인스턴스의 DNS 호스트 이름을 해당 IP 주소로 확인합니다. 그렇지 않으면 그렇지 않습니다.

명령:

```
aws ec2 describe-vpc-attribute --vpc-id vpc-a01106c2 --attribute enableDnsSupport
```

출력:

```
{
    "VpcId": "vpc-a01106c2",
    "EnableDnsSupport": {
        "Value": true
    }
}
```

속성에 대한 설명: enableDnsHostnames

이 예제에서는 `enableDnsHostnames` 속성을 설명합니다. 이 속성은 VPC에서 시작된 인스턴스가 DNS 호스트 이름을 가져오는지 여부를 나타냅니다. 이 속성이 `true` 인 경우 VPC의 인스턴스는 DNS 호스트 이름을 가져오고 그렇지 않으면 가져오지 않습니다.

명령:

```
aws ec2 describe-vpc-attribute --vpc-id vpc-a01106c2 --attribute
 enableDnsHostnames
```

출력:

```
{
    "VpcId": "vpc-a01106c2",
    "EnableDnsHostnames": {
        "Value": true
    }
}
```

- API 세부 정보는 AWS CLI 명령 DescribeVpcAttribute참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제에서는 'enableDnsSupport' 속성에 대해 설명합니다.

```
Get-EC2VpcAttribute -VpcId vpc-12345678 -Attribute enableDnsSupport
```

출력:

```
EnableDnsSupport
----------------
True
```

예 2: 이 예제에서는 'enableDnsHostnames' 속성에 대해 설명합니다.

```
Get-EC2VpcAttribute -VpcId vpc-12345678 -Attribute enableDnsHostnames
```

출력:

```
EnableDnsHostnames
------------------
True
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeVpcAttribute.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeVpcClassicLink** CLI와 함께 사용

다음 코드 예제는 DescribeVpcClassicLink의 사용 방법을 보여줍니다.

CLI

AWS CLI

VPC의 ClassicLink 상태를 설명하려면

이 예제에서는 ClassicLink vpc-888888의 상태를 나열합니다.

명령:

```
aws ec2 describe-vpc-classic-link --vpc-id vpc-88888888
```

출력:

```
{
  "Vpcs": [
    {
      "ClassicLinkEnabled": true,
      "VpcId": "vpc-88888888",
      "Tags": [
        {
          "Value": "classiclinkvpc",
          "Key": "Name"
        }
      ]
    }
  ]
}
```

이 예제에서는 Classiclink에 사용할 수 있는 VPC (필터 값은 로 설정) 만 나열합니다. `is-classic-link-enabled` `true`

명령:

```
aws ec2 describe-vpc-classic-link --filter "Name=is-classic-link-
enabled,Values=true"
```

- API 세부 정보는 명령 참조를 참조하십시오 DescribeVpcClassicLink.AWS CLI

PowerShell

도구: PowerShell

예 1: 위 예시는 해당 지역의 ClassicLinkEnabled 상태와 함께 모든 VPC를 반환합니다.

```
Get-EC2VpcClassicLink -Region eu-west-1
```

출력:

```
ClassicLinkEnabled Tags   VpcId
------------------ ----   -----
False              {Name} vpc-0fc1ff23f45b678eb
False              {}     vpc-01e23c4a5d6db78e9
False              {Name} vpc-0123456b078b9d01f
```

```
False               {}     vpc-12cf3b4f
False               {Name} vpc-0b12d3456a7e8901d
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeVpcClassicLink.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeVpcClassicLinkDnsSupport** CLI와 함께 사용

다음 코드 예제는 DescribeVpcClassicLinkDnsSupport의 사용 방법을 보여줍니다.

CLI

AWS CLI

VPC에 대한 ClassicLink DNS 지원에 대해 설명하려면

이 예제에서는 모든 VPC의 ClassicLink DNS 지원 상태를 설명합니다.

명령:

```
aws ec2 describe-vpc-classic-link-dns-support
```

출력:

```
{
  "Vpcs": [
    {
      "VpcId": "vpc-88888888",
      "ClassicLinkDnsSupported": true
    },
    {
      "VpcId": "vpc-1a2b3c4d",
      "ClassicLinkDnsSupported": false
    }
  ]
}
```

- API 세부 정보는 AWS CLI 명령 DescribeVpcClassicLinkDnsSupport참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 eu-west-1 지역에 대한 VPC의 ClassicLink DNS 지원 상태를 설명합니다.

```
Get-EC2VpcClassicLinkDnsSupport -VpcId vpc-0b12d3456a7e8910d -Region eu-west-1
```

출력:

```
ClassicLinkDnsSupported VpcId
----------------------- -----
False                   vpc-0b12d3456a7e8910d
False                   vpc-12cf3b4f
```

- API 세부 정보는 Cmdlet 참조를 참조하십시오. DescribeVpcClassicLinkDnsSupportAWS Tools for PowerShell

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeVpcEndpointServices** CLI와 함께 사용

다음 코드 예제는 DescribeVpcEndpointServices의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 모든 VPC 엔드포인트 서비스 설명하기

다음 "describe-vpc-endpoint-services" 예제는 특정 지역의 모든 VPC 엔드포인트 서비스를 나열합니다. AWS

```
aws ec2 describe-vpc-endpoint-services
```

출력:

```
{
    "ServiceDetails": [
        {
            "ServiceType": [
                {
                    "ServiceType": "Gateway"
                }
            ],
            "AcceptanceRequired": false,
            "ServiceName": "com.amazonaws.us-east-1.dynamodb",
            "VpcEndpointPolicySupported": true,
            "Owner": "amazon",
            "AvailabilityZones": [
                "us-east-1a",
                "us-east-1b",
                "us-east-1c",
                "us-east-1d",
                "us-east-1e",
                "us-east-1f"
            ],
            "BaseEndpointDnsNames": [
                "dynamodb.us-east-1.amazonaws.com"
            ]
        },
        {
            "ServiceType": [
                {
                    "ServiceType": "Interface"
                }
            ],
            "PrivateDnsName": "ec2.us-east-1.amazonaws.com",
            "ServiceName": "com.amazonaws.us-east-1.ec2",
            "VpcEndpointPolicySupported": false,
            "Owner": "amazon",
            "AvailabilityZones": [
                "us-east-1a",
                "us-east-1b",
                "us-east-1c",
                "us-east-1d",
                "us-east-1e",
                "us-east-1f"
            ],
```

```
            "AcceptanceRequired": false,
            "BaseEndpointDnsNames": [
                "ec2.us-east-1.vpce.amazonaws.com"
            ]
        },
        {
            "ServiceType": [
                {
                    "ServiceType": "Interface"
                }
            ],
            "PrivateDnsName": "ssm.us-east-1.amazonaws.com",
            "ServiceName": "com.amazonaws.us-east-1.ssm",
            "VpcEndpointPolicySupported": true,
            "Owner": "amazon",
            "AvailabilityZones": [
                "us-east-1a",
                "us-east-1b",
                "us-east-1c",
                "us-east-1d",
                "us-east-1e"
            ],
            "AcceptanceRequired": false,
            "BaseEndpointDnsNames": [
                "ssm.us-east-1.vpce.amazonaws.com"
            ]
        }
    ],
    "ServiceNames": [
        "com.amazonaws.us-east-1.dynamodb",
        "com.amazonaws.us-east-1.ec2",
        "com.amazonaws.us-east-1.ec2messages",
        "com.amazonaws.us-east-1.elasticloadbalancing",
        "com.amazonaws.us-east-1.kinesis-streams",
        "com.amazonaws.us-east-1.s3",
        "com.amazonaws.us-east-1.ssm"
    ]
}
```

자세한 내용은 사용 설명서의 사용 가능한 AWS 서비스 이름 보기를 참조하십시오. AWS PrivateLink

예 2: 엔드포인트 서비스에 대한 세부 정보 설명하기

다음 "describe-vpc-endpoint-services" 예제는 Amazon S3 인터페이스 엔드포인트 서비스의 세부 정보를 나열합니다.

```
aws ec2 describe-vpc-endpoint-services \
    --filter "Name=service-type,Values=Interface" Name=service-
name,Values=com.amazonaws.us-east-1.s3
```

출력:

```
{
    "ServiceDetails": [
        {
            "ServiceName": "com.amazonaws.us-east-1.s3",
            "ServiceId": "vpce-svc-081d84efcdEXAMPLE",
            "ServiceType": [
                {
                    "ServiceType": "Interface"
                }
            ],
            "AvailabilityZones": [
                "us-east-1a",
                "us-east-1b",
                "us-east-1c",
                "us-east-1d",
                "us-east-1e",
            "us-east-1f"
            ],
            "Owner": "amazon",
            "BaseEndpointDnsNames": [
                "s3.us-east-1.vpce.amazonaws.com"
            ],
            "VpcEndpointPolicySupported": true,
            "AcceptanceRequired": false,
            "ManagesVpcEndpoints": false,
            "Tags": []
        }
    ],
    "ServiceNames": [
        "com.amazonaws.us-east-1.s3"
    ]
}
```

자세한 내용은 사용 설명서 양식에서 사용 가능한 AWS 서비스 이름 보기를 참조하십시오. AWS PrivateLink

- API 세부 정보는 AWS CLI 명령 DescribeVpcEndpointServices참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 필터 (이 경우에는 com.amazonaws.eu-west-1.ecs) 를 사용하는 EC2 VPC 엔드포인트 서비스를 설명합니다. 또한 ServiceDetails 속성을 확장하고 세부 정보를 표시합니다.

```
Get-EC2VpcEndpointService -Region eu-west-1 -MaxResult 5 -Filter @{Name="service-
name";Values="com.amazonaws.eu-west-1.ecs"} | Select-Object -ExpandProperty
 ServiceDetails
```

출력:

```
AcceptanceRequired         : False
AvailabilityZones          : {eu-west-1a, eu-west-1b, eu-west-1c}
BaseEndpointDnsNames       : {ecs.eu-west-1.vpce.amazonaws.com}
Owner                      : amazon
PrivateDnsName             : ecs.eu-west-1.amazonaws.com
ServiceName                : com.amazonaws.eu-west-1.ecs
ServiceType                : {Amazon.EC2.Model.ServiceTypeDetail}
VpcEndpointPolicySupported : False
```

예 2: 이 예제는 모든 EC2 VPC 엔드포인트 서비스를 검색하고 일치하는 “ssm”을 반환합니다. ServiceNames

```
Get-EC2VpcEndpointService -Region eu-west-1 | Select-Object -ExpandProperty
 Servicenames | Where-Object { -match "ssm"}
```

출력:

```
com.amazonaws.eu-west-1.ssm
com.amazonaws.eu-west-1.ssmmessages
```

- API 세부 정보는 Cmdlet 참조를 참조하십시오 DescribeVpcEndpointServices.AWS Tools for PowerShell

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeVpcEndpoints** CLI와 함께 사용

다음 코드 예제는 DescribeVpcEndpoints의 사용 방법을 보여줍니다.

CLI

AWS CLI

VPC 엔드포인트를 설명하려면

다음 describe-vpc-endpoints 예제는 모든 VPC 엔드포인트의 세부 정보를 표시합니다.

```
aws ec2 describe-vpc-endpoints
```

출력:

```
{
    "VpcEndpoints": [
        {
            "PolicyDocument": "{\"Version\":\"2008-10-17\",\"Statement\":
[{\"Effect\":\"Allow\",\"Principal\":\"*\",\"Action\":\"*\",\"Resource\":\"*
\"}]}",
            "VpcId": "vpc-aabb1122",
            "NetworkInterfaceIds": [],
            "SubnetIds": [],
            "PrivateDnsEnabled": true,
            "State": "available",
            "ServiceName": "com.amazonaws.us-east-1.dynamodb",
            "RouteTableIds": [
                "rtb-3d560345"
            ],
            "Groups": [],
            "VpcEndpointId": "vpce-032a826a",
            "VpcEndpointType": "Gateway",
            "CreationTimestamp": "2017-09-05T20:41:28Z",
            "DnsEntries": [],
            "OwnerId": "123456789012"
        },
        {
```

```
            "PolicyDocument": "{\n  \"Statement\": [\n    {\n      \"Action\":
 \"*\", \n      \"Effect\": \"Allow\", \n      \"Principal\": \"*\", \n
 \"Resource\": \"*\"\n    }\n  ]\n}",
            "VpcId": "vpc-1a2b3c4d",
            "NetworkInterfaceIds": [
                "eni-2ec2b084",
                "eni-1b4a65cf"
            ],
            "SubnetIds": [
                "subnet-d6fcaa8d",
                "subnet-7b16de0c"
            ],
            "PrivateDnsEnabled": false,
            "State": "available",
            "ServiceName": "com.amazonaws.us-east-1.elasticloadbalancing",
            "RouteTableIds": [],
            "Groups": [
                {
                    "GroupName": "default",
                    "GroupId": "sg-54e8bf31"
                }
            ],
            "VpcEndpointId": "vpce-0f89a33420c1931d7",
            "VpcEndpointType": "Interface",
            "CreationTimestamp": "2017-09-05T17:55:27.583Z",
            "DnsEntries": [
                {
                    "HostedZoneId": "Z7HUB22UULQXV",
                    "DnsName": "vpce-0f89a33420c1931d7-
bluzidnv.elasticloadbalancing.us-east-1.vpce.amazonaws.com"
                },
                {
                    "HostedZoneId": "Z7HUB22UULQXV",
                    "DnsName": "vpce-0f89a33420c1931d7-bluzidnv-us-
east-1b.elasticloadbalancing.us-east-1.vpce.amazonaws.com"
                },
                {
                    "HostedZoneId": "Z7HUB22UULQXV",
                    "DnsName": "vpce-0f89a33420c1931d7-bluzidnv-us-
east-1a.elasticloadbalancing.us-east-1.vpce.amazonaws.com"
                }
            ],
            "OwnerId": "123456789012"
        },
```

```
        {
            "VpcEndpointId": "vpce-aabbaabbaabbaabba",
            "VpcEndpointType": "GatewayLoadBalancer",
            "VpcId": "vpc-111122223333aabbc",
            "ServiceName": "com.amazonaws.vpce.us-east-1.vpce-
svc-123123a1c43abc123",
            "State": "available",
            "SubnetIds": [
                "subnet-0011aabbcc2233445"
            ],
            "RequesterManaged": false,
            "NetworkInterfaceIds": [
                "eni-01010120203030405"
            ],
            "CreationTimestamp": "2020-11-11T08:06:03.522Z",
            "Tags": [],
            "OwnerId": "123456789012"
        }
    ]
}
```

자세한 내용은 Amazon VPC 사용 설명서의 VPC 엔드포인트를 참조하세요.

- API 세부 정보는 AWS CLI 명령 DescribeVpcEndpoints참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제에서는 eu-west-1 지역의 VPC 엔드포인트 중 하나 이상을 설명합니다. 그런 다음 출력을 다음 명령으로 파이프하여 VpcEndpointId 속성을 선택하고 배열 VPC ID를 문자열 배열로 반환합니다.

```
Get-EC2VpcEndpoint -Region eu-west-1 | Select-Object -ExpandProperty
 VpcEndpointId
```

출력:

```
vpce-01a2ab3f4f5cc6f7d
vpce-01d2b345a6787890b
vpce-0012e34d567890e12
vpce-0c123db4567890123
```

예제 2: 이 예제에서는 eu-west-1 지역의 모든 vpc 엔드포인트를 설명하고 VpcEndpointId,, VpcId 속성을 선택하여 표 형식으로 표시합니다. ServiceName PrivateDnsEnabled

```
Get-EC2VpcEndpoint -Region eu-west-1 | Select-Object VpcEndpointId, VpcId,
 ServiceName, PrivateDnsEnabled | Format-Table -AutoSize
```

출력:

```
VpcEndpointId          VpcId                ServiceName
 PrivateDnsEnabled
-------------          -----                -----------
 -----------------
vpce-02a2ab2f2f2cc2f2d vpc-0fc6ff46f65b039eb com.amazonaws.eu-west-1.ssm
             True
vpce-01d1b111a1114561b vpc-0fc6ff46f65b039eb com.amazonaws.eu-west-1.ec2
             True
vpce-0011e23d45167e838 vpc-0fc6ff46f65b039eb com.amazonaws.eu-west-1.ec2messages
             True
vpce-0c123db4567890123 vpc-0fc6ff46f65b039eb com.amazonaws.eu-west-1.ssmmessages
             True
```

예제 3: 이 예에서는 VPC 엔드포인트 vpce-01a2ab3f4f5cc6f7d에 대한 정책 문서를 json 파일로 내보냅니다.

```
Get-EC2VpcEndpoint -Region eu-west-1 -VpcEndpointId vpce-01a2ab3f4f5cc6f7d |
 Select-Object -expand PolicyDocument | Out-File vpce_policyDocument.json
```

- API DescribeVpcEndpoints세부AWS Tools for PowerShell 정보는 Cmdlet 참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeVpcs** CLI와 함께 사용

다음 코드 예제는 DescribeVpcs의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 복원력이 뛰어난 서비스 구축 및 관리

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
/// <summary>
/// Get the default VPC for the account.
/// </summary>
/// <returns>The default VPC object.</returns>
public async Task<Vpc> GetDefaultVpc()
{
    var vpcResponse = await _amazonEc2.DescribeVpcsAsync(
        new DescribeVpcsRequest()
        {
            Filters = new List<Amazon.EC2.Model.Filter>()
            {
                new ("is-default", new List<string>() { "true" })
            }
        });
    return vpcResponse.Vpcs[0];
}
```

- API 세부 정보는 AWS SDK for .NET API DescribeVpcs참조를 참조하십시오.

CLI

AWS CLI

예제 1: 모든 VPC를 설명하는 방법

다음 `describe-vpcs` 예제에서는 VPC에 대한 세부 정보를 검색합니다.

```
aws ec2 describe-vpcs
```

출력:

```
{
    "Vpcs": [
        {
            "CidrBlock": "30.1.0.0/16",
            "DhcpOptionsId": "dopt-19edf471",
            "State": "available",
            "VpcId": "vpc-0e9801d129EXAMPLE",
            "OwnerId": "111122223333",
            "InstanceTenancy": "default",
            "CidrBlockAssociationSet": [
                {
                    "AssociationId": "vpc-cidr-assoc-062c64cfafEXAMPLE",
                    "CidrBlock": "30.1.0.0/16",
                    "CidrBlockState": {
                        "State": "associated"
                    }
                }
            ],
            "IsDefault": false,
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "Not Shared"
                }
            ]
        },
        {
            "CidrBlock": "10.0.0.0/16",
            "DhcpOptionsId": "dopt-19edf471",
            "State": "available",
            "VpcId": "vpc-06e4ab6c6cEXAMPLE",
            "OwnerId": "222222222222",
            "InstanceTenancy": "default",
            "CidrBlockAssociationSet": [
                {
                    "AssociationId": "vpc-cidr-assoc-00b17b4eddEXAMPLE",
                    "CidrBlock": "10.0.0.0/16",
                    "CidrBlockState": {
                        "State": "associated"
```

```
                    }
                }
            ],
            "IsDefault": false,
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "Shared VPC"
                }
            ]
        }
    ]
}
```

예제 2: 지정된 VPC를 설명하는 방법

다음 `describe-vpcs` 예제에서는 지정된 VPC에 대한 세부 정보를 검색합니다.

```
aws ec2 describe-vpcs \
    --vpc-ids vpc-06e4ab6c6cEXAMPLE
```

출력:

```
{
    "Vpcs": [
        {
            "CidrBlock": "10.0.0.0/16",
            "DhcpOptionsId": "dopt-19edf471",
            "State": "available",
            "VpcId": "vpc-06e4ab6c6cEXAMPLE",
            "OwnerId": "111122223333",
            "InstanceTenancy": "default",
            "CidrBlockAssociationSet": [
                {
                    "AssociationId": "vpc-cidr-assoc-00b17b4eddEXAMPLE",
                    "CidrBlock": "10.0.0.0/16",
                    "CidrBlockState": {
                        "State": "associated"
                    }
                }
            ],
            "IsDefault": false,
            "Tags": [
```

```
                {
                    "Key": "Name",
                    "Value": "Shared VPC"
                }
            ]
        }
    ]
}
```

- API 세부 정보는 AWS CLI 명령 DescribeVpcs참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
const client = new EC2Client({});
const { Vpcs } = await client.send(
  new DescribeVpcsCommand({
    Filters: [{ Name: "is-default", Values: ["true"] }],
  }),
);
```

- API 세부 정보는 AWS SDK for JavaScript API DescribeVpcs참조를 참조하십시오.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예제에서는 지정된 VPC를 설명합니다.

```
Get-EC2Vpc -VpcId vpc-12345678
```

출력:

```
CidrBlock       : 10.0.0.0/16
DhcpOptionsId   : dopt-1a2b3c4d
InstanceTenancy : default
IsDefault       : False
State           : available
Tags            : {Name}
VpcId           : vpc-12345678
```

예 2: 이 예제에서는 기본 VPC를 설명합니다 (지역당 하나만 있을 수 있음). 계정이 이 지역에서 EC2-Classic을 지원하는 경우 기본 VPC는 없습니다.

```
Get-EC2Vpc -Filter @{Name="isDefault"; Values="true"}
```

출력:

```
CidrBlock       : 172.31.0.0/16
DhcpOptionsId   : dopt-12345678
InstanceTenancy : default
IsDefault       : True
State           : available
Tags            : {}
VpcId           : vpc-45678901
```

예 3: 이 예에서는 지정된 필터와 일치하는 VPC (즉, '10.0.0.0/16' 값과 일치하고 '사용 가능' 상태인 CIDR이 있는) 를 설명합니다.

```
Get-EC2Vpc -Filter @{Name="cidr";
 Values="10.0.0.0/16"},@{Name="state";Values="available"}
```

예 4: 이 예제에서는 모든 VPC에 대해 설명합니다.

```
Get-EC2Vpc
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeVpcs.

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
```

```
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def get_default_vpc(self):
        """
        Gets the default VPC for the account.

        :return: Data about the default VPC.
        """
        try:
            response = self.ec2_client.describe_vpcs(
                Filters=[{"Name": "is-default", "Values": ["true"]}]
            )
        except ClientError as err:
            raise AutoScalerError(f"Couldn't get default VPC: {err}")
        else:
            return response["Vpcs"][0]
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 DescribeVpcs.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeVpnConnections** CLI와 함께 사용

다음 코드 예제는 DescribeVpnConnections의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: VPN 연결 설명하기

다음 `describe-vpn-connections` 예는 모든 사이트-사이트 간 VPN 연결을 설명합니다.

```
aws ec2 describe-vpn-connections
```

출력:

```
{
    "VpnConnections": [
        {
            "CustomerGatewayConfiguration": "...configuration information...",
            "CustomerGatewayId": "cgw-01234567abcde1234",
            "Category": "VPN",
            "State": "available",
            "Type": "ipsec.1",
            "VpnConnectionId": "vpn-1122334455aabbccd",
            "TransitGatewayId": "tgw-00112233445566aab",
            "Options": {
                "EnableAcceleration": false,
                "StaticRoutesOnly": true,
                "LocalIpv4NetworkCidr": "0.0.0.0/0",
                "RemoteIpv4NetworkCidr": "0.0.0.0/0",
                "TunnelInsideIpVersion": "ipv4"
            },
            "Routes": [],
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "CanadaVPN"
                }
            ],
            "VgwTelemetry": [
                {
                    "AcceptedRouteCount": 0,
                    "LastStatusChange": "2020-07-29T10:35:11.000Z",
                    "OutsideIpAddress": "203.0.113.3",
                    "Status": "DOWN",
                    "StatusMessage": ""
```

```
                },
                {
                    "AcceptedRouteCount": 0,
                    "LastStatusChange": "2020-09-02T09:09:33.000Z",
                    "OutsideIpAddress": "203.0.113.5",
                    "Status": "UP",
                    "StatusMessage": ""
                }
            ]
        }
    ]
}
```

자세한 내용은 사이트 간 VPN 사용 AWS 설명서의 사이트 간 VPN 작동 방식을 참조하십시오.AWS

예 2: 사용 가능한 VPN 연결 설명하기

다음 `describe-vpn-connections` 예에서는 상태가 인 사이트-사이트 간 VPN 연결을 설명합니다. `available`

```
aws ec2 describe-vpn-connections \
    --filters "Name=state,Values=available"
```

자세한 내용은 사이트 간 VPN 사용 AWS 설명서의 사이트 간 VPN 작동 방식을 참조하십시오.AWS

- API 세부 정보는 명령 참조를 참조하십시오 DescribeVpnConnections.AWS CLI

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 VPN 연결을 설명합니다.

```
Get-EC2VpnConnection -VpnConnectionId vpn-12345678
```

출력:

```
CustomerGatewayConfiguration : [XML document]
CustomerGatewayId            : cgw-1a2b3c4d
```

```
Options                         : Amazon.EC2.Model.VpnConnectionOptions
Routes                          : {Amazon.EC2.Model.VpnStaticRoute}
State                           : available
Tags                            : {}
Type                            : ipsec.1
VgwTelemetry                    : {Amazon.EC2.Model.VgwTelemetry,
 Amazon.EC2.Model.VgwTelemetry}
VpnConnectionId                 : vpn-12345678
VpnGatewayId                    : vgw-1a2b3c4d
```

예 2: 이 예에서는 상태가 보류 중이거나 사용 가능인 모든 VPN 연결을 설명합니다.

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = @( "pending", "available" )

Get-EC2VpnConnection -Filter $filter
```

예 3: 이 예에서는 모든 VPN 연결을 설명합니다.

```
Get-EC2VpnConnection
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeVpnConnections.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DescribeVpnGateways** CLI와 함께 사용

다음 코드 예제는 DescribeVpnGateways의 사용 방법을 보여줍니다.

CLI

AWS CLI

가상 사설 게이트웨이를 설명하려면

이 예제에서는 가상 사설 게이트웨이를 설명합니다.

명령:

```
aws ec2 describe-vpn-gateways
```

출력:

```
{
    "VpnGateways": [
        {
            "State": "available",
            "Type": "ipsec.1",
            "VpnGatewayId": "vgw-f211f09b",
            "VpcAttachments": [
                {
                    "State": "attached",
                    "VpcId": "vpc-98eb5ef5"
                }
            ]
        },
        {
            "State": "available",
            "Type": "ipsec.1",
            "VpnGatewayId": "vgw-9a4cacf3",
            "VpcAttachments": [
                {
                    "State": "attaching",
                    "VpcId": "vpc-a01106c2"
                }
            ]
        }
    ]
}
```

- API 세부 정보는 AWS CLI 명령 DescribeVpnGateways참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 가상 프라이빗 게이트웨이를 설명합니다.

```
Get-EC2VpnGateway -VpnGatewayId vgw-1a2b3c4d
```

출력:

```
AvailabilityZone :
State            : available
Tags             : {}
Type             : ipsec.1
VpcAttachments   : {vpc-12345678}
VpnGatewayId     : vgw-1a2b3c4d
```

예 2: 이 예에서는 상태가 보류 중이거나 사용 가능인 모든 가상 프라이빗 게이트웨이를 설명합니다.

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "state"
$filter.Values = @( "pending", "available" )

Get-EC2VpnGateway -Filter $filter
```

예 3: 이 예에서는 모든 가상 사설 게이트웨이를 설명합니다.

```
Get-EC2VpnGateway
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DescribeVpnGateways.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DetachInternetGateway** CLI와 함께 사용

다음 코드 예제는 DetachInternetGateway의 사용 방법을 보여줍니다.

CLI

AWS CLI

VPC에서 인터넷 게이트웨이를 분리하려면

다음 detach-internet-gateway 예제는 특정 VPC에서 지정된 인터넷 게이트웨이를 분리합니다.

```
aws ec2 detach-internet-gateway \
    --internet-gateway-id igw-0d0fb496b3EXAMPLE \
    --vpc-id vpc-0a60eb65b4EXAMPLE
```

이 명령은 출력을 생성하지 않습니다.

자세한 내용은 Amazon VPC 사용 설명서의 인터넷 게이트웨이를 참조하세요.

- API 세부 정보는 AWS CLI 명령 DetachInternetGateway참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 VPC에서 지정된 인터넷 게이트웨이를 분리합니다.

```
Dismount-EC2InternetGateway -InternetGatewayId igw-1a2b3c4d -VpcId vpc-12345678
```

- API에 대한 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DetachInternetGateway.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DetachNetworkInterface** CLI와 함께 사용

다음 코드 예제는 DetachNetworkInterface의 사용 방법을 보여줍니다.

CLI

AWS CLI

인스턴스에서 네트워크 인터페이스를 분리하려면

이 예제는 지정된 인스턴스에서 지정된 네트워크 인터페이스를 분리합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 detach-network-interface --attachment-id eni-attach-66c4350a
```

- API 세부 정보는 AWS CLI 명령 DetachNetworkInterface참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 네트워크 인터페이스와 인스턴스 간의 지정된 연결을 제거합니다.

```
Dismount-EC2NetworkInterface -AttachmentId eni-attach-1a2b3c4d -Force
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DetachNetworkInterface.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DetachVolume** CLI와 함께 사용

다음 코드 예제는 DetachVolume의 사용 방법을 보여줍니다.

CLI

AWS CLI

인스턴스에서 볼륨을 분리하려면

이 예제 명령은 연결된 인스턴스에서 볼륨 (vol-049df61146c4d7901) 을 분리합니다.

명령:

```
aws ec2 detach-volume --volume-id vol-1234567890abcdef0
```

출력:

```
{
    "AttachTime": "2014-02-27T19:23:06.000Z",
```

```
    "InstanceId": "i-1234567890abcdef0",
    "VolumeId": "vol-049df61146c4d7901",
    "State": "detaching",
    "Device": "/dev/sdb"
}
```

- API 세부 정보는 AWS CLI 명령 DetachVolume참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 볼륨을 분리합니다.

```
Dismount-EC2Volume -VolumeId vol-12345678
```

출력:

```
AttachTime          : 12/22/2015 1:53:58 AM
DeleteOnTermination : False
Device              : /dev/sdh
InstanceId          : i-1a2b3c4d
State               : detaching
VolumeId            : vol-12345678
```

예 2: 올바른 볼륨을 분리하고 있는지 확인하기 위해 인스턴스 ID와 디바이스 이름을 지정할 수도 있습니다.

```
Dismount-EC2Volume -VolumeId vol-12345678 -InstanceId i-1a2b3c4d -Device /dev/sdh
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DetachVolume.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DetachVpnGateway** CLI와 함께 사용

다음 코드 예제는 DetachVpnGateway의 사용 방법을 보여줍니다.

CLI

AWS CLI

VPC에서 가상 프라이빗 게이트웨이를 분리하려면

이 예제는 지정된 VPC에서 지정된 가상 프라이빗 게이트웨이를 분리합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 detach-vpn-gateway --vpn-gateway-id vgw-9a4cacf3 --vpc-id vpc-a01106c2
```

- API 세부 정보는 AWS CLI 명령 DetachVpnGateway참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 VPC에서 지정된 가상 프라이빗 게이트웨이를 분리합니다.

```
Dismount-EC2VpnGateway -VpnGatewayId vgw-1a2b3c4d -VpcId vpc-12345678
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DetachVpnGateway.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DisableVgwRoutePropagation** CLI와 함께 사용

다음 코드 예제는 DisableVgwRoutePropagation의 사용 방법을 보여줍니다.

CLI

AWS CLI

경로 전파를 비활성화하려면

이 예제에서는 지정된 가상 프라이빗 게이트웨이가 고정 경로를 지정된 라우팅 테이블로 전파하지 못하도록 합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 disable-vgw-route-propagation --route-table-id rtb-22574640 --gateway-id
 vgw-9a4cacf3
```

- API 세부 정보는 AWS CLI 명령 DisableVgwRoutePropagation참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 VGW가 경로를 지정된 라우팅 테이블에 자동으로 전파하지 못하도록 합니다.

```
Disable-EC2VgwRoutePropagation -RouteTableId rtb-12345678 -GatewayId vgw-1a2b3c4d
```

- API에 대한 자세한 내용은 Cmdlet 참조를 참조하십시오 DisableVgwRoutePropagation.AWS Tools for PowerShell

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DisableVpcClassicLink** CLI와 함께 사용

다음 코드 예제는 DisableVpcClassicLink의 사용 방법을 보여줍니다.

CLI

AWS CLI

VPC를 ClassicLink 비활성화하려면

이 예시에서는 ClassicLink vpc-8888888을 비활성화합니다.

명령:

```
aws ec2 disable-vpc-classic-link --vpc-id vpc-88888888
```

출력:

```
{
  "Return": true
}
```

- API 세부 정보는 명령 참조를 참조하십시오. DisableVpcClassicLinkAWS CLI

PowerShell

도구: PowerShell

예제 1: 이 예에서는 VpcClassicLink vpc-01e23c4a5d6db78e9에 대해 EC2를 비활성화합니다. True 또는 False를 반환합니다.

```
Disable-EC2VpcClassicLink -VpcId vpc-01e23c4a5d6db78e9
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DisableVpcClassicLink.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DisableVpcClassicLinkDnsSupport** CLI와 함께 사용

다음 코드 예제는 DisableVpcClassicLinkDnsSupport의 사용 방법을 보여줍니다.

CLI

AWS CLI

VPC에 대한 ClassicLink DNS 지원을 비활성화하려면

이 예시에서는 에 대한 ClassicLink DNS 지원을 비활성화합니다. vpc-88888888

명령:

```
aws ec2 disable-vpc-classic-link-dns-support --vpc-id vpc-88888888
```

출력:

```
{
  "Return": true
}
```

- API 세부 정보는 AWS CLI 명령 DisableVpcClassicLinkDnsSupport참조를 참조하십시오.

PowerShell

도구: PowerShell

예제 1: 이 예에서는 vpc-0b12d3456a7e8910d에 대한 ClassicLink DNS 지원을 비활성화합니다.

```
Disable-EC2VpcClassicLinkDnsSupport -VpcId vpc-0b12d3456a7e8910d
```

- API에 DisableVpcClassicLinkDnsSupport대한 자세한 AWS Tools for PowerShell 내용은 Cmdlet 참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DisassociateAddress** CLI와 함께 사용

다음 코드 예제는 DisassociateAddress의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 인스턴스 시작하기

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
/// <summary>
/// Disassociate an Elastic IP address from an EC2 instance.
/// </summary>
/// <param name="associationId">The association Id.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> DisassociateIp(string associationId)
{
    var response = await _amazonEC2.DisassociateAddressAsync(
        new DisassociateAddressRequest { AssociationId = associationId });
    return response.HttpStatusCode == HttpStatusCode.OK;
}
```

- API 세부 정보는 AWS SDK for .NET API DisassociateAddress참조를 참조하십시오.

Bash

AWS CLI Bash 스크립트 사용

> **Note**
>
> 더 많은 정보가 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
###############################################################################
# function ec2_disassociate_address
#
# This function disassociates an Elastic IP address from an Amazon Elastic
 Compute Cloud (Amazon EC2) instance.
```

```
#
# Parameters:
#       -a association_id - The association ID that represents the association of
 the Elastic IP address with an instance.
#
# And:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
function ec2_disassociate_address() {
  local association_id response

  # Function to display usage information
  function usage() {
    echo "function ec2_disassociate_address"
    echo "Disassociates an Elastic IP address from an Amazon Elastic Compute
 Cloud (Amazon EC2) instance."
    echo "  -a association_id - The association ID that represents the
 association of the Elastic IP address with an instance."
    echo ""
  }

  # Parse the command-line arguments
  while getopts "a:h" option; do
    case "${option}" in
      a) association_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Validate the input parameters
  if [[ -z "$association_id" ]]; then
    errecho "ERROR: You must provide an association ID with the -a parameter."
    return 1
```

```
  fi

  response=$(aws ec2 disassociate-address \
    --association-id "$association_id") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports disassociate-address operation failed."
    errecho "$response"
    return 1
  }

  return 0
}
```

이 예제에 사용된 유틸리티 함수

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
###############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
```

```
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- API 세부 정보는 AWS CLI 명령 DisassociateAddress참조를 참조하십시오.

CLI

AWS CLI

EC2-Classic에서 탄력적 IP 주소를 연결 해제하는 방법

이 예제에서는 EC2-Classic의 인스턴스에서 탄력적 IP 주소를 연결 해제합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 disassociate-address --public-ip 198.51.100.0
```

EC2-VPC에서 탄력적 IP 주소를 연결 해제하는 방법

이 예제에서는 VPC의 인스턴스에서 탄력적 IP 주소를 연결 해제합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 disassociate-address --association-id eipassoc-2bebb745
```

- API에 대한 자세한 내용은 AWS CLI 명령 참조를 참조하십시오 DisassociateAddress.

Java

SDK for Java 2.x

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    public static void disassociateAddress(Ec2Client ec2, String associationId) {
        try {
            DisassociateAddressRequest addressRequest =
DisassociateAddressRequest.builder()
                    .associationId(associationId)
                    .build();

            ec2.disassociateAddress(addressRequest);
            System.out.println("You successfully disassociated the address!");

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API DisassociateAddress참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { DisassociateAddressCommand } from "@aws-sdk/client-ec2";
```

```
import { client } from "../libs/client.js";

// Disassociate an Elastic IP address from an instance.
export const main = async () => {
  const command = new DisassociateAddressCommand({
    // You can also use PublicIp, but that is for EC2 classic which is being
 retired.
    AssociationId: "ASSOCIATION_ID",
  });

  try {
    await client.send(command);
    console.log("Successfully disassociated address");
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API DisassociateAddress참조를 참조하십시오.

Kotlin

SDK for Kotlin

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
suspend fun disassociateAddressSc(associationIdVal: String?) {
    val addressRequest =
        DisassociateAddressRequest {
            associationId = associationIdVal
        }
    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.disassociateAddress(addressRequest)
        println("You successfully disassociated the address!")
    }
}
```

- API 세부 정보는 Kotlin API용AWS SDK 레퍼런스를 참조하세요 DisassociateAddress.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예제는 VPC의 지정된 인스턴스에서 지정된 엘라스틱 IP 주소를 분리합니다.

```
Unregister-EC2Address -AssociationId eipassoc-12345678
```

예 2: 이 예제는 EC2-Classic의 지정된 인스턴스에서 지정된 엘라스틱 IP 주소를 분리합니다.

```
Unregister-EC2Address -PublicIp 203.0.113.17
```

- API에 대한 세부 정보는 Cmdlet 참조를 참조하십시오 DisassociateAddress.AWS Tools for PowerShell

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class ElasticIpWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) Elastic IP address
 actions."""

    def __init__(self, ec2_resource, elastic_ip=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
```

```
        :param elastic_ip: A Boto3 VpcAddress object. This is a high-level object
that
                           wraps Elastic IP actions.
        """
        self.ec2_resource = ec2_resource
        self.elastic_ip = elastic_ip

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def disassociate(self):
        """
        Removes an association between an Elastic IP address and an instance.
When the
        association is removed, the instance is assigned a new public IP address.
        """
        if self.elastic_ip is None:
            logger.info("No Elastic IP to disassociate.")
            return

        try:
            self.elastic_ip.association.delete()
        except ClientError as err:
            logger.error(
                "Couldn't disassociate Elastic IP %s from its instance. Here's
why: %s: %s",
                self.elastic_ip.allocation_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 DisassociateAddress.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **DisassociateRouteTable** CLI와 함께 사용

다음 코드 예제는 DisassociateRouteTable의 사용 방법을 보여줍니다.

CLI

AWS CLI

라우팅 테이블의 연결을 끊으려면

이 예제는 지정된 서브넷에서 지정된 라우팅 테이블을 연결 해제합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 disassociate-route-table --association-id rtbassoc-781d0d1a
```

- API 세부 정보는 명령 참조를 참조하십시오 DisassociateRouteTable.AWS CLI

PowerShell

도구: PowerShell

예 1: 이 예제는 라우팅 테이블과 서브넷 간의 지정된 연결을 제거합니다.

```
Unregister-EC2RouteTable -AssociationId rtbassoc-1a2b3c4d
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 DisassociateRouteTable.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **EnableVgwRoutePropagation** CLI와 함께 사용

다음 코드 예제는 EnableVgwRoutePropagation의 사용 방법을 보여줍니다.

CLI

AWS CLI

경로 전파를 활성화하려면

이 예제를 사용하면 지정된 가상 프라이빗 게이트웨이가 고정 경로를 지정된 라우팅 테이블에 전파할 수 있습니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 enable-vgw-route-propagation --route-table-id rtb-22574640 --gateway-id
 vgw-9a4cacf3
```

- API 세부 정보는 AWS CLI 명령 EnableVgwRoutePropagation참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제를 사용하면 지정된 VGW가 경로를 지정된 라우팅 테이블에 자동으로 전파할 수 있습니다.

```
Enable-EC2VgwRoutePropagation -RouteTableId rtb-12345678 -GatewayId vgw-1a2b3c4d
```

- API에 대한 자세한 내용은 Cmdlet 참조를 참조하십시오 EnableVgwRoutePropagation.AWS Tools for PowerShell

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **EnableVolumeIo** CLI와 함께 사용

다음 코드 예제는 EnableVolumeIo의 사용 방법을 보여줍니다.

CLI

AWS CLI

볼륨의 I/O를 활성화하려면

이 예시에서는 볼륨 I/O를 vol-1234567890abcdef0 활성화합니다.

명령:

```
aws ec2 enable-volume-io --volume-id vol-1234567890abcdef0
```

출력:

```
{
  "Return": true
}
```

- API 세부 정보는 AWS CLI 명령 EnableVolumeIo참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 I/O 작업이 비활성화된 경우 지정된 볼륨에 대한 I/O 작업을 활성화합니다.

```
Enable-EC2VolumeIO -VolumeId vol-12345678
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 EnableVolumeIo.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **EnableVpcClassicLink** CLI와 함께 사용

다음 코드 예제는 EnableVpcClassicLink의 사용 방법을 보여줍니다.

CLI

AWS CLI

VPC를 활성화하려면 ClassicLink

이 예시에서는 vpc-8888888에 대해 활성화합니다. ClassicLink

명령:

```
aws ec2 enable-vpc-classic-link --vpc-id vpc-88888888
```

출력:

```
{
  "Return": true
}
```

- API 세부 정보는 명령 참조를 참조하십시오. EnableVpcClassicLinkAWS CLI

PowerShell

도구: PowerShell

예제 1: 이 예에서는 다음과 같은 VPC vpc-0123456b789b0d12f를 활성화합니다. ClassicLink

```
Enable-EC2VpcClassicLink -VpcId vpc-0123456b789b0d12f
```

출력:

```
True
```

- API에 대한 EnableVpcClassicLink세부 AWS Tools for PowerShell 정보는 Cmdlet 참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **EnableVpcClassicLinkDnsSupport** CLI와 함께 사용

다음 코드 예제는 EnableVpcClassicLinkDnsSupport의 사용 방법을 보여줍니다.

CLI

AWS CLI

VPC에 대한 ClassicLink DNS 지원을 활성화하려면

이 예시에서는 에 대한 ClassicLink vpc-88888888 DNS 지원을 활성화합니다.

명령:

```
aws ec2 enable-vpc-classic-link-dns-support --vpc-id vpc-88888888
```

출력:

```
{
  "Return": true
}
```

- API 세부 정보는 AWS CLI 명령 EnableVpcClassicLinkDnsSupport참조를 참조하십시오.

PowerShell

도구: PowerShell

예제 1: 이 예제를 사용하면 vpc-0b12d3456a7e8910d에서 다음과 같은 DNS 호스트 이름 확인을 지원할 수 있습니다. ClassicLink

```
Enable-EC2VpcClassicLinkDnsSupport -VpcId vpc-0b12d3456a7e8910d -Region eu-west-1
```

- API에 대한 AWS Tools for PowerShell 자세한 내용은 Cmdlet 참조를 참조하십시오. EnableVpcClassicLinkDnsSupport

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **GetConsoleOutput** CLI와 함께 사용

다음 코드 예제는 GetConsoleOutput의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 콘솔 출력을 가져오려면

다음 get-console-output 예제는 지정된 Linux 인스턴스의 콘솔 출력을 가져옵니다.

```
aws ec2 get-console-output \
    --instance-id i-1234567890abcdef0
```

출력:

```
{
    "InstanceId": "i-1234567890abcdef0",
    "Timestamp": "2013-07-25T21:23:53.000Z",
    "Output": "..."
}
```

자세한 내용은 Amazon EC2 사용 설명서의 인스턴스 콘솔 출력을 참조하십시오.

예 2: 최신 콘솔 출력을 가져오려면

다음 get-console-output 예제는 지정된 Linux 인스턴스의 최신 콘솔 출력을 가져옵니다.

```
aws ec2 get-console-output \
    --instance-id i-1234567890abcdef0 \
    --latest \
    --output text
```

출력:

```
i-1234567890abcdef0 [    0.000000] Command line: root=LABEL=/ console=tty1
 console=ttyS0 selinux=0 nvme_core.io_timeout=4294967295
[    0.000000] x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating point
 registers'
[    0.000000] x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
[    0.000000] x86/fpu: Supporting XSAVE feature 0x004: 'AVX registers'
...
Cloud-init v. 0.7.6 finished at Wed, 09 May 2018 19:01:13 +0000. Datasource
 DataSourceEc2.  Up 21.50 seconds
Amazon Linux AMI release 2018.03
Kernel 4.14.26-46.32.amzn1.x
```

자세한 내용은 Amazon EC2 사용 설명서의 인스턴스 콘솔 출력을 참조하십시오.

- API 세부 정보는 AWS CLI 명령 GetConsoleOutput참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 Linux 인스턴스의 콘솔 출력을 가져옵니다. 콘솔 출력이 인코딩됩니다.

```
Get-EC2ConsoleOutput -InstanceId i-0e19abcd47c123456
```

출력:

```
InstanceId          Output
----------          ------
i-0e194d3c47c123637 WyAgICAwLjAwMDAwMF0gQ29tbW...bGU9dHR5UzAgc2Vs
```

예 2: 이 예제에서는 인코딩된 콘솔 출력을 변수에 저장한 다음 디코딩합니다.

```
$Output_encoded = (Get-EC2ConsoleOutput -InstanceId i-0e19abcd47c123456).Output
[System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String($Output_encoded
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 GetConsoleOutput.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **GetHostReservationPurchasePreview** CLI와 함께 사용

다음 코드 예제는 GetHostReservationPurchasePreview의 사용 방법을 보여줍니다.

CLI

AWS CLI

전용 호스트 예약에 대한 구매 미리보기를 받으려면

이 예시에서는 계정에 지정된 전용 호스트의 특정 전용 호스트 예약 비용을 미리 볼 수 있습니다.

명령:

```
aws ec2 get-host-reservation-purchase-preview --offering-id hro-03f707bf363b6b324
 --host-id-set h-013abcd2a00cbd123
```

출력:

```
{
  "TotalHourlyPrice": "1.499",
  "Purchase": [
      {
          "HourlyPrice": "1.499",
          "InstanceFamily": "m4",
          "PaymentOption": "NoUpfront",
          "HostIdSet": [
              "h-013abcd2a00cbd123"
          ],
          "UpfrontPrice": "0.000",
          "Duration": 31536000
      }
  ],
  "TotalUpfrontPrice": "0.000"
}
```

- API 세부 정보는 AWS CLI 명령 GetHostReservationPurchasePreview참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 전용 호스트의 구성과 일치하는 구성으로 예약 구매를 미리 보여줍니다. h-01e23f4cd567890f1

```
Get-EC2HostReservationPurchasePreview -OfferingId hro-0c1f23456789d0ab -HostIdSet
 h-01e23f4cd567890f1
```

출력:

```
CurrencyCode Purchase TotalHourlyPrice TotalUpfrontPrice
------------ -------- ---------------- -----------------
             {}       1.307            0.000
```

- API GetHostReservationPurchasePreview세부AWS Tools for PowerShell 정보는 Cmdlet 참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **GetPasswordData** CLI와 함께 사용

다음 코드 예제는 GetPasswordData의 사용 방법을 보여줍니다.

CLI

AWS CLI

암호화된 암호를 가져오려면

이 예시에서는 암호화된 비밀번호를 가져옵니다.

명령:

```
aws ec2 get-password-data --instance-id i-1234567890abcdef0
```

출력:

```
{
    "InstanceId": "i-1234567890abcdef0",
    "Timestamp": "2013-08-07T22:18:38.000Z",
    "PasswordData": "gSlJFq+VpcZXqy+iktxMF6NyxQ4qCrT4+gaOuNOenX1MmgXPTj7XEXAMPLE
UQ+YeFfb+L1U4C4AKv652Ux1iRB3CPTYP7WmU3TUnhsuBd+p6LVk7T2lKUml6OXbk6WPW1VYYm/TRPB1
e1DQ7PY4an/DgZT4mwcpRFigzhniQgDDeO1InvSDcwoUTwNs0Y1S8ouri2W4n5GNlriM3Q0AnNVelVz/
53TkDtxbNoU606M1gK9zUWSxqEgwvbV2j8c5rP0WCuaMWSFl4ziDu4bd7q+4RSyi8NUsVWnKZ4aEZffu
DPGzKrF5yLlf3etP2L4ZR6CvG7K1hx7VKOQVN32Dajw=="
}
```

해독된 암호를 가져오려면

이 예제에서는 해독된 비밀번호를 가져옵니다.

명령:

```
aws ec2 get-password-data --instance-id  i-1234567890abcdef0 --priv-launch-key C:
\Keys\MyKeyPair.pem
```

출력:

```
{
    "InstanceId": "i-1234567890abcdef0",
    "Timestamp": "2013-08-30T23:18:05.000Z",
    "PasswordData": "&ViJ652e*u"
}
```

- API 세부 정보는 명령 참조를 참조하십시오 GetPasswordData.AWS CLI

PowerShell

도구: PowerShell

예 1: 이 예제는 Amazon EC2가 지정된 Windows 인스턴스의 관리자 계정에 할당한 암호를 해독합니다. pem 파일이 지정되었으므로 -Decrypt 스위치의 설정이 자동으로 가정됩니다.

```
Get-EC2PasswordData -InstanceId i-12345678 -PemFile C:\path\my-key-pair.pem
```

출력:

```
mYZ(PA9?C)Q
```

예 2: (PowerShell Windows만 해당) 인스턴스를 검사하여 인스턴스를 시작하는 데 사용된 키페어의 이름을 확인한 다음 Visual Studio용 AWS 툴킷의 구성 저장소에서 해당 키페어 데이터를 찾으려고 합니다. 키페어 데이터를 찾으면 암호가 해독됩니다.

```
Get-EC2PasswordData -InstanceId i-12345678 -Decrypt
```

출력:

```
mYZ(PA9?C)Q
```

예 3: 인스턴스의 암호화된 암호 데이터를 반환합니다.

```
Get-EC2PasswordData -InstanceId i-12345678
```

출력:

```
iVz3BAK/WAXV.....dqt8WeMA==
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 GetPasswordData.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ImportImage** CLI와 함께 사용

다음 코드 예제는 ImportImage의 사용 방법을 보여줍니다.

CLI

AWS CLI

VM 이미지 파일을 AMI로 가져오려면

다음 import-image 예제는 지정된 OVA를 가져옵니다.

```
aws ec2 import-image \
  --disk-containers Format=ova,UserBucket="{S3Bucket=my-import-bucket,S3Key=vms/
my-server-vm.ova}"
```

출력:

```
{
    "ImportTaskId": "import-ami-1234567890abcdef0",
    "Progress": "2",
    "SnapshotDetails": [
        {
            "DiskImageSize": 0.0,
            "Format": "ova",
            "UserBucket": {
                "S3Bucket": "my-import-bucket",
                "S3Key": "vms/my-server-vm.ova"
            }
        }
    ],
    "Status": "active",
```

```
    "StatusMessage": "pending"
}
```

- API 세부 정보는 AWS CLI 명령 ImportImage참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제에서는 지정된 Amazon S3 버킷에서 동일 토큰을 사용하여 Amazon EC2로 단일 디스크 가상 머신 이미지를 가져옵니다. 이 예제에서는 VM Import 사전 요구 사항 항목에 설명된 대로, 지정된 버킷에 대한 Amazon EC2 액세스를 허용하는 정책과 함께 기본 이름이 'vmimport'인 VM 가져오기 서비스 역할이 있어야 합니다. 사용자 지정 역할을 사용하려면 파라미터를 사용하여 역할 이름을 지정하십시오. **-RoleName**

```
$container = New-Object Amazon.EC2.Model.ImageDiskContainer
$container.Format="VMDK"
$container.UserBucket = New-Object Amazon.EC2.Model.UserBucket
$container.UserBucket.S3Bucket = "myVirtualMachineImages"
$container.UserBucket.S3Key = "Win_2008_Server_Standard_SP2_64-bit-disk1.vmdk"

$parms = @{
    "ClientToken"="idempotencyToken"
    "Description"="Windows 2008 Standard Image Import"
    "Platform"="Windows"
    "LicenseType"="AWS"
}

Import-EC2Image -DiskContainer $container @parms
```

출력:

```
Architecture    :
Description     : Windows 2008 Standard Image
Hypervisor      :
ImageId         :
ImportTaskId    : import-ami-abcdefgh
LicenseType     : AWS
Platform        : Windows
Progress        : 2
SnapshotDetails : {}
```

```
Status          : active
StatusMessage   : pending
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 ImportImage.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ImportKeyPair** CLI와 함께 사용

다음 코드 예제는 ImportKeyPair의 사용 방법을 보여줍니다.

CLI

AWS CLI

퍼블릭 키를 가져오려면

먼저, 원하는 도구를 사용하여 키 페어를 생성합니다. 예를 들어, 다음 ssh-keygen 명령을 사용하세요.

명령:

```
ssh-keygen -t rsa -C "my-key" -f ~/.ssh/my-key
```

출력:

```
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ec2-user/.ssh/my-key.
Your public key has been saved in /home/ec2-user/.ssh/my-key.pub.
...
```

이 예제 명령은 지정된 공개 키를 가져옵니다.

명령:

```
aws ec2 import-key-pair --key-name "my-key" --public-key-material fileb://~/.ssh/
my-key.pub
```

출력:

```
{
  "KeyName": "my-key",
  "KeyFingerprint": "1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca"
}
```

- API 세부 정보는 AWS CLI 명령 ImportKeyPair참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 퍼블릭 키를 EC2로 가져옵니다. 첫 번째 줄은 퍼블릭 키 파일 (*.pub) 의 콘텐츠를 변수에 저장합니다. **$publickey** 다음으로, 예제에서는 공개 키 파일의 UTF8 형식을 Base64로 인코딩된 문자열로 변환하고 변환된 문자열을 변수에 저장합니다. **$pkbase64** 마지막 줄에서는 변환된 공개 키를 EC2로 가져옵니다. cmdlet은 키 핑거프린트와 이름을 결과로 반환합니다.

```
$publickey=[Io.File]::ReadAllText("C:\Users\TestUser\.ssh\id_rsa.pub")
$pkbase64 =
 [System.Convert]::ToBase64String([System.Text.Encoding]::UTF8.GetBytes($publickey))
Import-EC2KeyPair -KeyName Example-user-key -PublicKey $pkbase64
```

출력:

```
KeyFingerprint                                  KeyName
--------------                                  -------
do:d0:15:8f:79:97:12:be:00:fd:df:31:z3:b1:42:z1 Example-user-key
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 ImportKeyPair.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ImportSnapshot** CLI와 함께 사용

다음 코드 예제는 ImportSnapshot의 사용 방법을 보여줍니다.

CLI

AWS CLI

스냅샷을 가져오려면

다음 `import-snapshot` 예제에서는 지정된 디스크를 스냅샷으로 가져옵니다.

```
aws ec2 import-snapshot \
    --description "My server VMDK" \
    --disk-container Format=VMDK,UserBucket={S3Bucket=my-import-bucket,S3Key=vms/
my-server-vm.vmdk}
```

출력:

```
{
    "Description": "My server VMDK",
    "ImportTaskId": "import-snap-1234567890abcdef0",
    "SnapshotTaskDetail": {
        "Description": "My server VMDK",
        "DiskImageSize": "0.0",
        "Format": "VMDK",
        "Progress": "3",
        "Status": "active",
        "StatusMessage": "pending"
        "UserBucket": {
            "S3Bucket": "my-import-bucket",
            "S3Key": "vms/my-server-vm.vmdk"
        }
    }
}
```

- API 세부 정보는 AWS CLI 명령 ImportSnapshot참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 'VMDK' 형식의 VM 디스크 이미지를 Amazon EBS 스냅샷으로 가져옵니다. 이 예제에는 AWSEC http://docs.aws.amazon.com/ WindowsGuide ImportPrerequisites 2/ latest/ /VM .html의 항목에 설명된 대로 Amazon EC2가 지정된 버킷에 액세스할 수 있도록 허용하는 정책과 함께 기본 이름이 'vmimport'인 **`VM Import Prequisites`** VM 가져오기 서비스

역할이 필요합니다. 사용자 지정 역할을 사용하려면 파라미터를 사용하여 역할 이름을 지정하십시오. **-RoleName**

```
$parms = @{
    "ClientToken"="idempotencyToken"
    "Description"="Disk Image Import"
    "DiskContainer_Description" = "Data disk"
    "DiskContainer_Format" = "VMDK"
    "DiskContainer_S3Bucket" = "myVirtualMachineImages"
    "DiskContainer_S3Key" = "datadiskimage.vmdk"
}

Import-EC2Snapshot @parms
```

출력:

```
Description            ImportTaskId               SnapshotTaskDetail

-----------------      --------------------       ------------------

Disk Image Import      import-snap-abcdefgh
 Amazon.EC2.Model.SnapshotTaskDetail
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 ImportSnapshot.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ModifyCapacityReservation** CLI와 함께 사용

다음 코드 예제는 ModifyCapacityReservation의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 기존 용량 예약으로 예약된 인스턴스 수 변경하기

다음 modify-capacity-reservation 예제는 용량 예약으로 용량을 예약하는 인스턴스의 수를 변경합니다.

```
aws ec2 modify-capacity-reservation \
    --capacity-reservation-id cr-1234abcd56EXAMPLE \
    --instance-count 5
```

출력:

```
{
    "Return": true
}
```

예 2: 기존 용량 예약의 종료 날짜 및 시간 변경하기

다음 `modify-capacity-reservation` 예제에서는 기존 용량 예약을 수정하여 지정된 날짜 및 시간에 종료합니다.

```
aws ec2 modify-capacity-reservation \
    --capacity-reservation-id cr-1234abcd56EXAMPLE \
    --end-date-type limited \
    --end-date 2019-08-31T23:59:59Z
```

자세한 내용은 Linux 인스턴스용 Amazon Elastic Compute 클라우드 사용 설명서의 용량 예약 수정을 참조하십시오.

- API 세부 정보는 AWS CLI 명령 ModifyCapacityReservation참조를 참조하십시오.

PowerShell

도구: PowerShell

예제 1: 이 예제에서는 인스턴스 수를 1로 변경하여 CapacityReservationId cr-0c1f2345db6f7cdba를 수정합니다.

```
Edit-EC2CapacityReservation -CapacityReservationId cr-0c1f2345db6f7cdba -
InstanceCount 1
```

출력:

```
True
```

- AWS Tools for PowerShell API에 ModifyCapacityReservation대한 자세한 내용은 Cmdlet 참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ModifyHosts** CLI와 함께 사용

다음 코드 예제는 ModifyHosts의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 전용 호스트의 자동 배치를 활성화하려면

다음 modify-hosts 예제는 전용 호스트의 자동 배치를 활성화하여 인스턴스 유형 구성과 일치하는 대상으로 지정되지 않은 인스턴스 시작을 수락하도록 합니다.

```
aws ec2 modify-hosts \
    --host-id h-06c2f189b4EXAMPLE \
    --auto-placement on
```

출력:

```
{
    "Successful": [
        "h-06c2f189b4EXAMPLE"
    ],
    "Unsuccessful": []
}
```

예 2: 전용 호스트의 호스트 복구를 활성화하려면

다음 modify-hosts 예에서는 지정된 전용 호스트에 대한 호스트 복구를 활성화합니다.

```
aws ec2 modify-hosts \
    --host-id h-06c2f189b4EXAMPLE \
    --host-recovery on
```

출력:

```
{
    "Successful": [
        "h-06c2f189b4EXAMPLE"
    ],
    "Unsuccessful": []
}
```

자세한 내용은 Linux 인스턴스용 Amazon Elastic Compute 클라우드 사용 설명서의 전용 호스트 자동 배치 수정을 참조하십시오.

- API 세부 정보는 AWS CLI 명령 ModifyHosts참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 전용 호스트 h-01e23f4cd567890f3의 AutoPlacement 설정을 끄기로 수정합니다.

```
Edit-EC2Host -HostId h-03e09f8cd681609f3 -AutoPlacement off
```

출력:

```
Successful            Unsuccessful
----------            ------------
{h-01e23f4cd567890f3} {}
```

- AWS Tools for PowerShell API에 ModifyHosts대한 자세한 내용은 Cmdlet 참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ModifyIdFormat** CLI와 함께 사용

다음 코드 예제는 ModifyIdFormat의 사용 방법을 보여줍니다.

CLI

AWS CLI

리소스에 더 긴 ID 형식을 사용하도록 설정하려면

다음 `modify-id-format` 예시에서는 `instance` 리소스 유형에 더 긴 ID 형식을 사용할 수 있습니다.

```
aws ec2 modify-id-format \
    --resource instance \
    --use-long-ids
```

리소스의 더 긴 ID 형식을 비활성화하려면

다음 `modify-id-format` 예제에서는 `instance` 리소스 유형에 대해 더 긴 ID 형식을 비활성화합니다.

```
aws ec2 modify-id-format \
    --resource instance \
    --no-use-long-ids
```

다음 `modify-id-format` 예시에서는 옵트인 기간 내에 지원되는 모든 리소스 유형에 대해 더 긴 ID 형식을 활성화합니다.

```
aws ec2 modify-id-format \
    --resource all-current \
    --use-long-ids
```

- API 세부 정보는 AWS CLI 명령 ModifyIdFormat참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 리소스 유형에 대해 더 긴 ID 형식을 활성화합니다.

```
Edit-EC2IdFormat -Resource instance -UseLongId $true
```

예 2: 이 예제는 지정된 리소스 유형에 대해 더 긴 ID 형식을 비활성화합니다.

```
Edit-EC2IdFormat -Resource instance -UseLongId $false
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 ModifyIdFormat.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ModifyImageAttribute** CLI와 함께 사용

다음 코드 예제는 ModifyImageAttribute의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: AMI를 퍼블릭으로 설정하려면

다음 modify-instance-attribute 예제는 지정된 AMI를 퍼블릭으로 설정합니다.

```
aws ec2 modify-image-attribute \
    --image-id ami-5731123e \
    --launch-permission "Add=[{Group=all}]"
```

이 명령은 출력을 생성하지 않습니다.

예 2: AMI를 비공개로 설정하려면

다음 modify-instance-attribute 예제는 지정된 AMI를 비공개로 설정합니다.

```
aws ec2 modify-image-attribute \
    --image-id ami-5731123e \
    --launch-permission "Remove=[{Group=all}]"
```

이 명령은 출력을 생성하지 않습니다.

예 3: AWS 계정에 시작 권한 부여하기

다음 modify-instance-attribute 예시는 지정된 AWS 계정에 시작 권한을 부여합니다.

```
aws ec2 modify-image-attribute \
```

```
    --image-id ami-5731123e \
    --launch-permission "Add=[{UserId=123456789012}]"
```

이 명령은 출력을 생성하지 않습니다.

예 4: AWS 계정에서 시작 권한을 제거하려면

다음 `modify-instance-attribute` 예시에서는 지정된 AWS 계정에서 시작 권한을 제거합니다.

```
aws ec2 modify-image-attribute \
    --image-id ami-5731123e \
    --launch-permission "Remove=[{UserId=123456789012}]"
```

- API 세부 정보는 AWS CLI 명령 ModifyImageAttribute참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 AMI에 대한 설명을 업데이트합니다.

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Description "New description"
```

예 2: 이 예제는 AMI를 퍼블릭 (예: 누구나 사용할 AWS 계정 수 있도록) 으로 설정합니다.

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -
OperationType add -UserGroup all
```

예 3: 이 예에서는 AMI를 비공개로 설정합니다 (예: 소유자인 사용자만 사용할 수 있음).

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -
OperationType remove -UserGroup all
```

예 4: 이 예제는 지정된 항목에 시작 권한을 부여합니다 AWS 계정.

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -
OperationType add -UserId 111122223333
```

예 5: 이 예제는 지정된 권한에서 시작 권한을 제거합니다 AWS 계정.

```
Edit-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission -
OperationType remove -UserId 111122223333
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 ModifyImageAttribute.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ModifyInstanceAttribute** CLI와 함께 사용

다음 코드 예제는 ModifyInstanceAttribute의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 인스턴스 유형 수정하기

다음 modify-instance-attribute 예제는 지정된 인스턴스의 인스턴스 유형을 수정합니다. 인스턴스는 stopped 상태여야 합니다.

```
aws ec2 modify-instance-attribute \
    --instance-id i-1234567890abcdef0 \
    --instance-type "{\"Value\": \"m1.small\"}"
```

이 명령은 출력을 생성하지 않습니다.

예 2: 인스턴스에서 향상된 네트워킹 활성화하기

다음 modify-instance-attribute 예제는 지정된 인스턴스에 향상된 네트워킹을 활성화합니다. 인스턴스는 stopped 상태여야 합니다.

```
aws ec2 modify-instance-attribute \
    --instance-id i-1234567890abcdef0 \
    --sriov-net-support simple
```

이 명령은 출력을 생성하지 않습니다.

예 3: sourceDestCheck 속성 수정하기

다음 modify-instance-attribute 예제에서는 지정된 인스턴스의 sourceDestCheck 속성을 로 설정합니다true. 인스턴스는 VPC에 있어야 합니다.

```
aws ec2 modify-instance-attribute --instance-id i-1234567890abcdef0 --source-
dest-check "{\"Value\": true}"
```

이 명령은 출력을 생성하지 않습니다.

예 4: 루트 볼륨의 deleteOnTermination 속성을 수정하려면

다음 modify-instance-attribute 예제는 지정된 Amazon EBS 기반 인스턴스의 루트 볼륨 deleteOnTermination 속성을 로 설정합니다. false 기본적으로 이 속성은 루트 true 볼륨용입니다.

명령:

```
aws ec2 modify-instance-attribute \
  --instance-id i-1234567890abcdef0 \
  --block-device-mappings "[{\"DeviceName\": \"/dev/sda1\",\"Ebs\":
{\"DeleteOnTermination\":false}}]"
```

이 명령은 출력을 생성하지 않습니다.

예 5: 인스턴스에 연결된 사용자 데이터를 수정하려면

다음 modify-instance-attribute 예제에서는 파일의 내용을 지정된 인스턴스의 UserData.txt 내용으로 추가합니다. UserData

원본 파일의 내용UserData.txt:

```
#!/bin/bash
yum update -y
service httpd start
chkconfig httpd on
```

파일 내용은 base64로 인코딩되어야 합니다. 첫 번째 명령은 텍스트 파일을 base64로 변환하고 새 파일로 저장합니다.

명령의 리눅스/맥OS 버전:

```
base64 UserData.txt > UserData.base64.txt
```

이 명령은 출력을 생성하지 않습니다.

명령의 윈도우 버전:

```
certutil -encode UserData.txt tmp.b64 && findstr /v /c:- tmp.b64 >
 UserData.base64.txt
```

출력:

```
Input Length = 67
Output Length = 152
CertUtil: -encode command completed successfully.
```

이제 다음과 같은 CLI 명령에서 해당 파일을 참조할 수 있습니다.

```
aws ec2 modify-instance-attribute \
    --instance-id=i-09b5a14dbca622e76 \
    --attribute userData --value file://UserData.base64.txt
```

이 명령은 출력을 생성하지 않습니다.

자세한 내용은 EC2 사용 설명서의 사용자 데이터 및 AWS CLI를 참조하십시오.

- API 세부 정보는 AWS CLI 명령 ModifyInstanceAttribute참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 인스턴스의 인스턴스 유형을 수정합니다.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -InstanceType m3.medium
```

예 2: 이 예제는 단일 루트 I/O 가상화 (SR-IOV) 네트워크 지원 파라미터의 값으로 “simple”을 지정하여 지정된 인스턴스의 향상된 네트워킹을 활성화합니다. SriovNetSupport

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -SriovNetSupport "simple"
```

예 3: 이 예에서는 지정된 인스턴스의 보안 그룹을 수정합니다. 인스턴스는 VPC에 있어야 합니다. 이름이 아닌 각 보안 그룹의 ID를 지정해야 합니다.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -Group @( "sg-12345678",
 "sg-45678901" )
```

예 4: 이 예에서는 지정된 인스턴스에 대해 EBS I/O 최적화를 활성화합니다. 모든 인스턴스 유형에서 이 기능을 사용할 수 있는 것은 아닙니다. EBS 최적화 인스턴스를 사용할 경우 추가 사용 요금이 적용됩니다.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -EbsOptimized $true
```

예 5: 이 예제에서는 지정된 인스턴스의 원본/목적지 확인을 활성화합니다. NAT 인스턴스가 NAT를 수행하려면 값이 'false'여야 합니다.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -SourceDestCheck $true
```

예 6: 이 예제는 지정된 인스턴스의 종료를 비활성화합니다.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -DisableApiTermination $true
```

예 7: 이 예제는 인스턴스에서 종료가 시작될 때 종료되도록 지정된 인스턴스를 변경합니다.

```
Edit-EC2InstanceAttribute -InstanceId i-12345678 -
InstanceInitiatedShutdownBehavior terminate
```

- API 세부 정보는 Cmdlet 참조를 참조하십시오 ModifyInstanceAttribute.AWS Tools for PowerShell

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ModifyInstanceCreditSpecification** CLI와 함께 사용

다음 코드 예제는 `ModifyInstanceCreditSpecification`의 사용 방법을 보여줍니다.

CLI

AWS CLI

인스턴스의 CPU 사용량에 대한 크레딧 옵션을 수정하려면

이 예제에서는 지정된 지역에서 지정된 인스턴스의 CPU 사용량에 대한 크레딧 옵션을 “무제한”으로 수정합니다. 유효한 크레딧 옵션은 “표준”과 “무제한”입니다.

명령:

```
aws ec2 modify-instance-credit-specification --instance-credit-specification
 "InstanceId=i-1234567890abcdef0,CpuCredits=unlimited"
```

출력:

```
{
  "SuccessfulInstanceCreditSpecifications": [
    {
      "InstanceId": "i-1234567890abcdef0"
    }
  ],
  "UnsuccessfulInstanceCreditSpecifications": []
}
```

- API 세부 정보는 AWS CLI 명령 ModifyInstanceCreditSpecification참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이렇게 하면 i-01234567890abcdef 인스턴스에 T2 무제한 크레딧이 활성화됩니다.

```
$Credit = New-Object -TypeName
 Amazon.EC2.Model.InstanceCreditSpecificationRequest
$Credit.InstanceId = "i-01234567890abcdef"
$Credit.CpuCredits = "unlimited"
Edit-EC2InstanceCreditSpecification -InstanceCreditSpecification $Credit
```

- API에 대한 ModifyInstanceCreditSpecification AWS Tools for PowerShell자세한 내용은 Cmdlet 참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ModifyNetworkInterfaceAttribute** CLI와 함께 사용

다음 코드 예제는 ModifyNetworkInterfaceAttribute의 사용 방법을 보여줍니다.

CLI

AWS CLI

네트워크 인터페이스의 첨부 속성 수정하기

이 예제 명령은 지정된 네트워크 인터페이스의 attachment 속성을 수정합니다.

명령:

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --
attachment AttachmentId=eni-attach-43348162,DeleteOnTermination=false
```

네트워크 인터페이스의 설명 속성을 수정하려면

이 예제 명령은 지정된 네트워크 인터페이스의 description 속성을 수정합니다.

명령:

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --
description "My description"
```

네트워크 인터페이스의 GroupSet 속성을 수정하려면

이 예제 명령은 지정된 네트워크 인터페이스의 groupSet 속성을 수정합니다.

명령:

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --
groups sg-903004f8 sg-1a2b3c4d
```

네트워크 인터페이스의 sourceDestCheck 속성을 수정하려면

이 예제 명령은 지정된 네트워크 인터페이스의 sourceDestCheck 속성을 수정합니다.

명령:

```
aws ec2 modify-network-interface-attribute --network-interface-id eni-686ea200 --
no-source-dest-check
```

- API 세부 정보는 AWS CLI 명령 ModifyNetworkInterfaceAttribute참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 종료 시 지정된 첨부 파일이 삭제되도록 지정된 네트워크 인터페이스를 수정합니다.

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -
Attachment_AttachmentId eni-attach-1a2b3c4d -Attachment_DeleteOnTermination $true
```

예 2: 이 예에서는 지정된 네트워크 인터페이스의 설명을 수정합니다.

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -Description
 "my description"
```

예 3: 이 예에서는 지정된 네트워크 인터페이스의 보안 그룹을 수정합니다.

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -Groups
 sg-1a2b3c4d
```

예 4: 이 예에서는 지정된 네트워크 인터페이스의 원본/대상 검사를 비활성화합니다.

```
Edit-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -
SourceDestCheck $false
```

- API 세부 정보는 Cmdlet 참조를 참조하십시오 ModifyNetworkInterfaceAttribute.AWS Tools for PowerShell

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ModifyReservedInstances** CLI와 함께 사용

다음 코드 예제는 ModifyReservedInstances의 사용 방법을 보여줍니다.

CLI

AWS CLI

예약 인스턴스를 수정하려면

이 예제 명령은 예약 인스턴스를 같은 지역의 다른 가용 영역으로 이동합니다.

명령:

```
aws ec2 modify-reserved-instances --reserved-instances-ids b847fa93-
e282-4f55-b59a-1342f5bd7c02 --target-configurations AvailabilityZone=us-
west-1c,Platform=EC2-Classic,InstanceCount=10
```

출력:

```
{
  "ReservedInstancesModificationId": "rimod-d3ed4335-b1d3-4de6-ab31-0f13aaf46687"
}
```

예약 인스턴스의 네트워크 플랫폼을 수정하려면

이 예제 명령은 EC2-Classic 예약 인스턴스를 EC2-VPC 로 변환합니다.

명령:

```
aws ec2 modify-reserved-instances --reserved-instances-ids f127bd27-
edb7-44c9-a0eb-0d7e09259af0 --target-configurations AvailabilityZone=us-
west-1c,Platform=EC2-VPC,InstanceCount=5
```

출력:

```
{
  "ReservedInstancesModificationId": "rimod-82fa9020-668f-4fb6-945d-61537009d291"
}
```

자세한 내용은 Amazon EC2 사용 설명서의 예약 인스턴스 수정을 참조하십시오.

예약 인스턴스의 인스턴스 크기를 수정하려면

이 예제 명령은 us-west-1c에 m1.small 리눅스/유닉스 인스턴스 10개가 있는 예약 인스턴스를 수정하여 m1.small 인스턴스 8개가 m1.large 인스턴스 2개가 되고 나머지 2m1.small 인스턴스는 동일한 가용 영역에서 1m1.medium 인스턴스가 되도록 수정합니다. 명령:

```
aws ec2 modify-reserved-instances --reserved-instances-
ids 1ba8e2e3-3556-4264-949e-63ee671405a9 --target-
configurations AvailabilityZone=us-west-1c,Platform=EC2-
Classic,InstanceCount=2,InstanceType=m1.large AvailabilityZone=us-
west-1c,Platform=EC2-Classic,InstanceCount=1,InstanceType=m1.medium
```

출력:

```
{
    "ReservedInstancesModificationId": "rimod-acc5f240-080d-4717-
b3e3-1c6b11fa00b6"
}
```

자세한 내용은 Amazon EC2 사용 설명서의 예약 인스턴스 크기 수정을 참조하십시오.

- API 세부 정보는 AWS CLI 명령 ModifyReservedInstances참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 예약 인스턴스의 가용 영역, 인스턴스 수, 플랫폼을 수정합니다.

```
$config = New-Object Amazon.EC2.Model.ReservedInstancesConfiguration
$config.AvailabilityZone = "us-west-2a"
$config.InstanceCount = 1
$config.Platform = "EC2-VPC"

Edit-EC2ReservedInstance `
-ReservedInstancesId @("FE32132D-70D5-4795-B400-AE435EXAMPLE",
 "0CC556F3-7AB8-4C00-B0E5-98666EXAMPLE") `
-TargetConfiguration $config
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 ModifyReservedInstances.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ModifySnapshotAttribute** CLI와 함께 사용

다음 코드 예제는 ModifySnapshotAttribute의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 스냅샷 속성 수정하기

다음 modify-snapshot-attribute 예제는 지정된 스냅샷의 createVolumePermission 속성을 업데이트하여 지정된 사용자에 대한 볼륨 권한을 제거합니다.

```
aws ec2 modify-snapshot-attribute \
    --snapshot-id snap-1234567890abcdef0 \
    --attribute createVolumePermission \
    --operation-type remove \
    --user-ids 123456789012
```

예 2: 스냅샷을 퍼블릭으로 설정하려면

다음 modify-snapshot-attribute 예제에서는 지정된 스냅샷을 퍼블릭으로 설정합니다.

```
aws ec2 modify-snapshot-attribute \
    --snapshot-id snap-1234567890abcdef0 \
    --attribute createVolumePermission \
    --operation-type add \
    --group-names all
```

- API 세부 정보는 AWS CLI 명령 ModifySnapshotAttribute참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제에서는 CreateVolumePermission 속성을 설정하여 지정된 스냅샷을 공개합니다.

```
Edit-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute
 CreateVolumePermission -OperationType Add -GroupName all
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 ModifySnapshotAttribute.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ModifySpotFleetRequest** CLI와 함께 사용

다음 코드 예제는 ModifySpotFleetRequest의 사용 방법을 보여줍니다.

CLI

AWS CLI

스팟 플릿 요청을 수정하려면

이 예제 명령은 지정된 스팟 플릿 요청의 목표 용량을 업데이트합니다.

명령:

```
aws ec2 modify-spot-fleet-request --target-capacity 20 --spot-fleet-request-id
 sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

출력:

```
{
    "Return": true
}
```

이 예제 명령은 결과적으로 스팟 인스턴스를 종료하지 않고 지정된 스팟 플릿 요청의 목표 용량을 줄입니다.

명령:

```
aws ec2 modify-spot-fleet-request --target-capacity 10 --excess-capacity-
termination-policy NoTermination --spot-fleet-request-ids sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE
```

출력:

```
{
    "Return": true
}
```

- API 세부 정보는 AWS CLI 명령 ModifySpotFleetRequest참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 스팟 플릿 요청의 목표 용량을 업데이트합니다.

```
Edit-EC2SpotFleetRequest -SpotFleetRequestId sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE -TargetCapacity 10
```

출력:

```
True
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 ModifySpotFleetRequest.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ModifySubnetAttribute** CLI와 함께 사용

다음 코드 예제는 ModifySubnetAttribute의 사용 방법을 보여줍니다.

CLI

AWS CLI

서브넷의 퍼블릭 IPv4 주소 지정 동작을 변경하려면

이 예제에서는 subnet-1a2b3c4d를 수정하여 이 서브넷에서 시작되는 모든 인스턴스에 퍼블릭 IPv4 주소를 할당하도록 지정합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 modify-subnet-attribute --subnet-id subnet-1a2b3c4d --map-public-ip-on-
launch
```

서브넷의 IPv6 주소 지정 동작을 변경하려면

이 예제에서는 subnet-1a2b3c4d를 수정하여 이 서브넷에서 시작되는 모든 인스턴스에 서브넷 범위의 IPv6 주소를 할당하도록 지정합니다.

명령:

```
aws ec2 modify-subnet-attribute --subnet-id subnet-1a2b3c4d --assign-ipv6-
address-on-creation
```

자세한 내용은 가상 사설 클라우드 사용 설명서의 VPC의 IP 주소AWS 지정을 참조하십시오.

- API 세부 정보는 AWS CLI 명령 ModifySubnetAttribute참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 서브넷의 퍼블릭 IP 주소 지정을 활성화합니다.

```
Edit-EC2SubnetAttribute -SubnetId subnet-1a2b3c4d -MapPublicIpOnLaunch $true
```

예 2: 이 예에서는 지정된 서브넷의 퍼블릭 IP 주소 지정을 비활성화합니다.

```
Edit-EC2SubnetAttribute -SubnetId subnet-1a2b3c4d -MapPublicIpOnLaunch $false
```

- API 세부 정보는 Cmdlet 참조를 참조하십시오 ModifySubnetAttribute.AWS Tools for PowerShell

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ModifyVolumeAttribute** CLI와 함께 사용

다음 코드 예제는 ModifyVolumeAttribute의 사용 방법을 보여줍니다.

CLI

AWS CLI

볼륨 속성을 수정하려면

이 예제에서는 vol-1234567890abcdef0 ID가 인 볼륨의 autoEnableIo 속성을 로 설정합니다true. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 modify-volume-attribute --volume-id vol-1234567890abcdef0 --auto-enable-
io
```

- API 세부 정보는 AWS CLI 명령 ModifyVolumeAttribute참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 볼륨의 지정된 속성을 수정합니다. 데이터가 일치하지 않을 수 있어 일시 중단된 후 볼륨의 I/O 작업이 자동으로 재개됩니다.

```
Edit-EC2VolumeAttribute -VolumeId vol-12345678 -AutoEnableIO $true
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 ModifyVolumeAttribute.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ModifyVpcAttribute** CLI와 함께 사용

다음 코드 예제는 ModifyVpcAttribute의 사용 방법을 보여줍니다.

CLI

AWS CLI

enableDnsSupport 속성을 수정하려면

이 예제는 enableDnsSupport 속성을 수정합니다. 이 속성은 VPC에 DNS 확인이 활성화되었는지 여부를 나타냅니다. 이 속성이 true 인 경우 Amazon DNS 서버는 인스턴스의 DNS 호스트 이름을 해당 IP 주소로 확인합니다. 그렇지 않으면 그렇지 않습니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 modify-vpc-attribute --vpc-id vpc-a01106c2 --enable-dns-support "{\"Value
\":false}"
```

속성을 수정하려면 enableDnsHostnames

이 예제는 enableDnsHostnames 속성을 수정합니다. 이 속성은 VPC에서 시작된 인스턴스가 DNS 호스트 이름을 가져오는지 여부를 나타냅니다. 이 속성이 true 인 경우 VPC의 인스턴스는 DNS 호스트 이름을 가져오고 그렇지 않으면 가져오지 않습니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 modify-vpc-attribute --vpc-id vpc-a01106c2 --enable-dns-hostnames
 "{\"Value\":false}"
```

- API 세부 정보는 AWS CLI 명령 ModifyVpcAttribute참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 VPC의 DNS 호스트 이름을 지원할 수 있습니다.

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsHostnames $true
```

예 2: 이 예제에서는 지정된 VPC의 DNS 호스트 이름 지원을 비활성화합니다.

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsHostnames $false
```

예 3: 이 예제에서는 지정된 VPC에 대한 DNS 확인을 지원할 수 있습니다.

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsSupport $true
```

예 4: 이 예제에서는 지정된 VPC에 대한 DNS 확인 지원을 비활성화합니다.

```
Edit-EC2VpcAttribute -VpcId vpc-12345678 -EnableDnsSupport $false
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 ModifyVpcAttribute.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **MonitorInstances** CLI와 함께 사용

다음 코드 예제는 MonitorInstances의 사용 방법을 보여줍니다.

C++

SDK for C++

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::MonitorInstancesRequest request;
    request.AddInstanceIds(instanceId);
    request.SetDryRun(true);

    auto dry_run_outcome = ec2Client.MonitorInstances(request);
```

```
    if (dry_run_outcome.IsSuccess()) {
        std::cerr
                << "Failed dry run to enable monitoring on instance. A dry run
should trigger an error."
                <<
                std::endl;
        return false;
    }
    else if (dry_run_outcome.GetError().GetErrorType()
             != Aws::EC2::EC2Errors::DRY_RUN_OPERATION) {
        std::cerr << "Failed dry run to enable monitoring on instance " <<
                  instanceId << ": " << dry_run_outcome.GetError().GetMessage()
<<
                  std::endl;
        return false;
    }

    request.SetDryRun(false);
    auto monitorInstancesOutcome = ec2Client.MonitorInstances(request);
    if (!monitorInstancesOutcome.IsSuccess()) {
        std::cerr << "Failed to enable monitoring on instance " <<
                  instanceId << ": " <<
                  monitorInstancesOutcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully enabled monitoring on instance " <<
                  instanceId << std::endl;
    }
```

- API 세부 정보는 AWS SDK for C++ API MonitorInstances참조를 참조하십시오.

CLI

AWS CLI

인스턴스에 대한 세부 모니터링을 활성화하는 방법

이 예제 명령은 지정된 인스턴스에 대한 세부 모니터링을 활성화합니다.

명령:

```
aws ec2 monitor-instances --instance-ids i-1234567890abcdef0
```

출력:

```
{
  "InstanceMonitorings": [
      {
          "InstanceId": "i-1234567890abcdef0",
          "Monitoring": {
              "State": "pending"
          }
      }
  ]
}
```

- API 세부 정보는 AWS CLI 명령 MonitorInstances참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { MonitorInstancesCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

// Turn on detailed monitoring for the selected instance.
// By default, metrics are sent to Amazon CloudWatch every 5 minutes.
// For a cost you can enable detailed monitoring which sends metrics every
 minute.
export const main = async () => {
  const command = new MonitorInstancesCommand({
    InstanceIds: ["INSTANCE_ID"],
  });

  try {
    const { InstanceMonitorings } = await client.send(command);
    const instancesBeingMonitored = InstanceMonitorings.map(
```

```
      (im) =>
        ` • Detailed monitoring state for ${im.InstanceId} is
 ${im.Monitoring.State}.`,
    );
    console.log("Monitoring status:");
    console.log(instancesBeingMonitored.join("\n"));
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API MonitorInstances참조를 참조하십시오.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예제는 지정된 인스턴스에 대한 세부 모니터링을 활성화합니다.

```
Start-EC2InstanceMonitoring -InstanceId i-12345678
```

출력:

```
InstanceId    Monitoring
----------    ----------
i-12345678    Amazon.EC2.Model.Monitoring
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 MonitorInstances.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    DATA lt_instance_ids TYPE /aws1/
cl_ec2instidstringlist_w=>tt_instanceidstringlist.
    APPEND NEW /aws1/cl_ec2instidstringlist_w( iv_value = iv_instance_id ) TO
 lt_instance_ids.

    "Perform dry run"
    TRY.
        " DryRun is set to true. This checks for the required permissions to
 monitor the instance without actually making the request. "
        lo_ec2->monitorinstances(
          it_instanceids = lt_instance_ids
          iv_dryrun = abap_true
        ).
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        " If the error code returned is `DryRunOperation`, then you have the
 required permissions to monitor this instance. "
        IF lo_exception->av_err_code = 'DryRunOperation'.
          MESSAGE 'Dry run to enable detailed monitoring completed.' TYPE 'I'.
          " DryRun is set to false to enable detailed monitoring. "
          lo_ec2->monitorinstances(
            it_instanceids = lt_instance_ids
            iv_dryrun = abap_false
          ).
          MESSAGE 'Detailed monitoring enabled.' TYPE 'I'.
          " If the error code returned is `UnauthorizedOperation`, then you don't
 have the required permissions to monitor this instance. "
        ELSEIF lo_exception->av_err_code = 'UnauthorizedOperation'.
          MESSAGE 'Dry run to enable detailed monitoring failed. User does not
 have the permissions to monitor the instance.' TYPE 'E'.
        ELSE.
          DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
          MESSAGE lv_error TYPE 'E'.
        ENDIF.
    ENDTRY.
```

- API 세부 정보는 SAP용AWS SDK ABAP API 참조를 참조하십시오 MonitorInstances.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **MoveAddressToVpc** CLI와 함께 사용

다음 코드 예제는 MoveAddressToVpc의 사용 방법을 보여줍니다.

CLI

AWS CLI

주소를 EC2-VPC 주소로 이동하려면

이 예에서는 엘라스틱 IP 주소 54.123.4.56을 EC2-VPC 플랫폼으로 이동합니다.

명령:

```
aws ec2 move-address-to-vpc --public-ip 54.123.4.56
```

출력:

```
{
  "Status": "MoveInProgress"
}
```

- API 세부 정보는 명령 참조를 참조하십시오. MoveAddressToVpcAWS CLI

PowerShell

도구: PowerShell

예 1: 이 예에서는 퍼블릭 IP 주소가 12.345.67.89인 EC2 인스턴스를 미국 동부 (버지니아 북부) 지역의 EC2-VPC 플랫폼으로 이동합니다.

```
Move-EC2AddressToVpc -PublicIp 12.345.67.89 -Region us-east-1
```

예 2: 이 예제는 명령 결과를 cmdlet으로 파이프합니다. Get-EC2Instance Move-EC2AddressToVpc 이 Get-EC2Instance 명령은 인스턴스 ID로 지정된 인스턴스를 가져온 다음 인스턴스의 퍼블릭 IP 주소 속성을 반환합니다.

```
(Get-EC2Instance -Instance i-12345678).Instances.PublicIpAddress | Move-
EC2AddressToVpc
```

- API에 대한 자세한 내용은 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 MoveAddressToVpc.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **PurchaseHostReservation** CLI와 함께 사용

다음 코드 예제는 PurchaseHostReservation의 사용 방법을 보여줍니다.

CLI

AWS CLI

전용 호스트 예약을 구매하려면

이 예시에서는 사용자 계정의 지정된 전용 호스트에 대해 지정된 전용 호스트 예약 상품을 구매합니다.

명령:

```
aws ec2 purchase-host-reservation --offering-id hro-03f707bf363b6b324 --host-id-
set h-013abcd2a00cbd123
```

출력:

```
{
  "TotalHourlyPrice": "1.499",
  "Purchase": [
      {
          "HourlyPrice": "1.499",
          "InstanceFamily": "m4",
          "PaymentOption": "NoUpfront",
          "HostIdSet": [
              "h-013abcd2a00cbd123"
          ],
```

```
            "HostReservationId": "hr-0d418a3a4ffc669ae",
            "UpfrontPrice": "0.000",
            "Duration": 31536000
        }
    ],
    "TotalUpfrontPrice": "0.000"
}
```

- API 세부 정보는 AWS CLI 명령 PurchaseHostReservation참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 전용 호스트 h-01e23f4cd567890f1의 구성과 일치하는 구성을 갖춘 예약 오퍼링 hro-0c1f23456789d0ab를 구매합니다.

```
New-EC2HostReservation -OfferingId hro-0c1f23456789d0ab HostIdSet
 h-01e23f4cd567890f1
```

출력:

```
ClientToken       :
CurrencyCode      :
Purchase          : {hr-0123f4b5d67bedc89}
TotalHourlyPrice  : 1.307
TotalUpfrontPrice : 0.000
```

- PurchaseHostReservation AWS Tools for PowerShellAPI에 대한 자세한 내용은 Cmdlet 참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **PurchaseScheduledInstances** CLI와 함께 사용

다음 코드 예제는 PurchaseScheduledInstances의 사용 방법을 보여줍니다.

CLI

AWS CLI

정기 인스턴스를 구매하려면

이 예시에서는 정기 인스턴스를 구입합니다.

명령:

```
aws ec2 purchase-scheduled-instances --purchase-requests file://purchase-
request.json
```

구매 요청.json:

```
[
    {
        "PurchaseToken": "eyJ2IjoiMSIsInMiOjEsImMiOi...",
        "InstanceCount": 1
    }
]
```

출력:

```
{
  "ScheduledInstanceSet": [
      {
          "AvailabilityZone": "us-west-2b",
          "ScheduledInstanceId": "sci-1234-1234-1234-1234-123456789012",
          "HourlyPrice": "0.095",
          "CreateDate": "2016-01-25T21:43:38.612Z",
          "Recurrence": {
              "OccurrenceDaySet": [
                  1
              ],
              "Interval": 1,
              "Frequency": "Weekly",
              "OccurrenceRelativeToEnd": false,
              "OccurrenceUnit": ""
          },
          "Platform": "Linux/UNIX",
          "TermEndDate": "2017-01-31T09:00:00Z",
```

```
            "InstanceCount": 1,
            "SlotDurationInHours": 32,
            "TermStartDate": "2016-01-31T09:00:00Z",
            "NetworkPlatform": "EC2-VPC",
            "TotalScheduledInstanceHours": 1696,
            "NextSlotStartTime": "2016-01-31T09:00:00Z",
            "InstanceType": "c4.large"
        }
    ]
}
```

- API 세부 정보는 명령 참조를 참조하십시오. PurchaseScheduledInstancesAWS CLI

PowerShell

도구: PowerShell

예 1: 이 예에서는 정기 인스턴스를 구매합니다.

```
$request = New-Object Amazon.EC2.Model.PurchaseRequest
$request.InstanceCount = 1
$request.PurchaseToken = "eyJ2IjoiMSIsInMiOjEsImMiOi..."
New-EC2ScheduledInstancePurchase -PurchaseRequest $request
```

출력:

```
AvailabilityZone            : us-west-2b
CreateDate                  : 1/25/2016 1:43:38 PM
HourlyPrice                 : 0.095
InstanceCount               : 1
InstanceType                : c4.large
NetworkPlatform             : EC2-VPC
NextSlotStartTime           : 1/31/2016 1:00:00 AM
Platform                    : Linux/UNIX
PreviousSlotEndTime         :
Recurrence                  : Amazon.EC2.Model.ScheduledInstanceRecurrence
ScheduledInstanceId         : sci-1234-1234-1234-1234-123456789012
SlotDurationInHours         : 32
TermEndDate                 : 1/31/2017 1:00:00 AM
TermStartDate               : 1/31/2016 1:00:00 AM
TotalScheduledInstanceHours : 1696
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 PurchaseScheduledInstances.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **RebootInstances** CLI와 함께 사용

다음 코드 예제는 RebootInstances의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 복원력이 뛰어난 서비스 구축 및 관리

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Reboot EC2 instances.
    /// </summary>
    /// <param name="ec2InstanceId">The instance Id of the instances that will be
rebooted.</param>
    /// <returns>Async task.</returns>
    public async Task RebootInstances(string ec2InstanceId)
    {
        var request = new RebootInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId },
        };

        var response = await _amazonEC2.RebootInstancesAsync(request);
```

```
        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            Console.WriteLine("Instances successfully rebooted.");
        }
        else
        {
            Console.WriteLine("Could not reboot one or more instances.");
        }
    }
```

인스턴스의 프로파일을 바꾸고, 재부팅하며, 웹 서버를 다시 시작합니다.

```
    /// <summary>
    /// Replace the profile associated with a running instance. After the profile
is replaced, the instance
    /// is rebooted to ensure that it uses the new profile. When the instance is
ready, Systems Manager is
    /// used to restart the Python web server.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to update.</param>
    /// <param name="credsProfileName">The name of the new profile to associate
with the specified instance.</param>
    /// <param name="associationId">The Id of the existing profile association
for the instance.</param>
    /// <returns>Async task.</returns>
    public async Task ReplaceInstanceProfile(string instanceId, string
credsProfileName, string associationId)
    {
        await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
                new ReplaceIamInstanceProfileAssociationRequest()
                {
                    AssociationId = associationId,
                    IamInstanceProfile = new IamInstanceProfileSpecification()
                    {
                        Name = credsProfileName
                    }
                });
        // Allow time before resetting.
        Thread.Sleep(25000);
        var instanceReady = false;
        var retries = 5;
        while (retries-- > 0 && !instanceReady)
```

```
        {
            await _amazonEc2.RebootInstancesAsync(
                new RebootInstancesRequest(new List<string>() { instanceId }));
            Thread.Sleep(10000);

            var instancesPaginator =
_amazonSsm.Paginators.DescribeInstanceInformation(
                new DescribeInstanceInformationRequest());
            // Get the entire list using the paginator.
            await foreach (var instance in
instancesPaginator.InstanceInformationList)
            {
                instanceReady = instance.InstanceId == instanceId;
                if (instanceReady)
                {
                    break;
                }
            }
        }
        Console.WriteLine($"Sending restart command to instance {instanceId}");
        await _amazonSsm.SendCommandAsync(
            new SendCommandRequest()
            {
                InstanceIds = new List<string>() { instanceId },
                DocumentName = "AWS-RunShellScript",
                Parameters = new Dictionary<string, List<string>>()
                {
                        {"commands", new List<string>() { "cd / && sudo python3
server.py 80" }}
                }
            });
        Console.WriteLine($"Restarted the web server on instance {instanceId}");
    }
```

- API 세부 정보는 AWS SDK for .NET API RebootInstances참조를 참조하십시오.

C++

SDK for C++

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);

    Aws::EC2::Model::RebootInstancesRequest request;
    request.AddInstanceIds(instanceId);
    request.SetDryRun(true);

    auto dry_run_outcome = ec2Client.RebootInstances(request);
    if (dry_run_outcome.IsSuccess()) {
        std::cerr
                << "Failed dry run to reboot on instance. A dry run should
trigger an error."
                <<
                std::endl;
        return false;
    }
    else if (dry_run_outcome.GetError().GetErrorType()
             != Aws::EC2::EC2Errors::DRY_RUN_OPERATION) {
        std::cout << "Failed dry run to reboot instance " << instanceId << ": "
                  << dry_run_outcome.GetError().GetMessage() << std::endl;
        return false;
    }

    request.SetDryRun(false);
    auto outcome = ec2Client.RebootInstances(request);
    if (!outcome.IsSuccess()) {
        std::cout << "Failed to reboot instance " << instanceId << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully rebooted instance " << instanceId <<
                  std::endl;
    }
```

- API 세부 정보는 AWS SDK for C++ API RebootInstances참조를 참조하십시오.

CLI

AWS CLI

Amazon EC2 인스턴스를 재부팅하는 방법

이 예제에서는 지정된 인스턴스를 재부팅합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 reboot-instances --instance-ids i-1234567890abcdef5
```

자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서에서 인스턴스 재부팅을 참조하세요.

- API 세부 정보는 AWS CLI 명령 RebootInstances참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { RebootInstancesCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new RebootInstancesCommand({
    InstanceIds: ["INSTANCE_ID"],
  });
```

```
  try {
    await client.send(command);
    console.log("Instance rebooted successfully.");
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API RebootInstances참조를 참조하십시오.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예제는 지정된 인스턴스를 재부팅합니다.

```
Restart-EC2Instance -InstanceId i-12345678
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 RebootInstances.

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
```

```
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def replace_instance_profile(
        self, instance_id, new_instance_profile_name, profile_association_id
    ):
        """
        Replaces the profile associated with a running instance. After the
profile is
        replaced, the instance is rebooted to ensure that it uses the new
profile. When
        the instance is ready, Systems Manager is used to restart the Python web
server.
```

```
        :param instance_id: The ID of the instance to update.
        :param new_instance_profile_name: The name of the new profile to
associate with
                                          the specified instance.
        :param profile_association_id: The ID of the existing profile association
for the
                                       instance.
        """
        try:
            self.ec2_client.replace_iam_instance_profile_association(
                IamInstanceProfile={"Name": new_instance_profile_name},
                AssociationId=profile_association_id,
            )
            log.info(
                "Replaced instance profile for association %s with profile %s.",
                profile_association_id,
                new_instance_profile_name,
            )
            time.sleep(5)
            inst_ready = False
            tries = 0
            while not inst_ready:
                if tries % 6 == 0:
                    self.ec2_client.reboot_instances(InstanceIds=[instance_id])
                    log.info(
                        "Rebooting instance %s and waiting for it to to be
ready.",
                        instance_id,
                    )
                tries += 1
                time.sleep(10)
                response = self.ssm_client.describe_instance_information()
                for info in response["InstanceInformationList"]:
                    if info["InstanceId"] == instance_id:
                        inst_ready = True
            self.ssm_client.send_command(
                InstanceIds=[instance_id],
                DocumentName="AWS-RunShellScript",
                Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
            )
            log.info("Restarted the Python web server on instance %s.",
instance_id)
        except ClientError as err:
```

```
            raise AutoScalerError(
                f"Couldn't replace instance profile for association
{profile_association_id}: {err}"
            )
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 RebootInstances.

Rust

SDK for Rust

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
async fn reboot_instance(client: &Client, id: &str) -> Result<(), Error> {
    println!("Rebooting instance.");

    client.reboot_instances().instance_ids(id).send().await?;

    client
        .wait_until_instance_stopped()
        .instance_ids(id)
        .wait(Duration::from_secs(60))
        .await?;
    let wait_status_ok = client
        .wait_until_instance_status_ok()
        .instance_ids(id)
        .wait(Duration::from_secs(60))
        .await;

    match wait_status_ok {
        Ok(_) => println!("Rebooted instance {id}, it is started with status
OK."),
        Err(err) => return Err(err.into()),
    }
```

```
    Ok(())
}
```

- API에 대한 자세한 내용은 Rust용AWS SDK API 레퍼런스를 참조하십시오 RebootInstances.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> 자세한 내용은 여기에서 확인할 수 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    DATA lt_instance_ids TYPE /aws1/
cl_ec2instidstringlist_w=>tt_instanceidstringlist.
    APPEND NEW /aws1/cl_ec2instidstringlist_w( iv_value = iv_instance_id ) TO
 lt_instance_ids.

    "Perform dry run"
    TRY.
        " DryRun is set to true. This checks for the required permissions to
 reboot the instance without actually making the request. "
        lo_ec2->rebootinstances(
          it_instanceids = lt_instance_ids
          iv_dryrun = abap_true
        ).
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        " If the error code returned is `DryRunOperation`, then you have the
 required permissions to reboot this instance. "
        IF lo_exception->av_err_code = 'DryRunOperation'.
          MESSAGE 'Dry run to reboot instance completed.' TYPE 'I'.
          " DryRun is set to false to make a reboot request. "
          lo_ec2->rebootinstances(
             it_instanceids = lt_instance_ids
             iv_dryrun = abap_false
           ).
          MESSAGE 'Instance rebooted.' TYPE 'I'.
```

```
          " If the error code returned is `UnauthorizedOperation`, then you don't
 have the required permissions to reboot this instance. "
        ELSEIF lo_exception->av_err_code = 'UnauthorizedOperation'.
          MESSAGE 'Dry run to reboot instance failed. User does not have
 permissions to reboot the instance.' TYPE 'E'.
        ELSE.
          DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
          MESSAGE lv_error TYPE 'E'.
        ENDIF.
    ENDTRY.
```

- API 세부 정보는 SAP용AWS SDK ABAP API 참조를 참조하십시오 RebootInstances.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **RegisterImage** CLI와 함께 사용

다음 코드 예제는 RegisterImage의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 매니페스트 파일을 사용하여 AMI를 등록하려면

다음 register-image 예제는 Amazon S3의 지정된 매니페스트 파일을 사용하여 AMI를 등록합니다.

```
aws ec2 register-image \
    --name my-image \
    --image-location my-s3-bucket/myimage/image.manifest.xml
```

출력:

```
{
    "ImageId": "ami-1234567890EXAMPLE"
}
```

자세한 내용은 Amazon EC2 사용 설명서에서 Amazon Machine Image(AMI)를 참조하세요.

예 2: 루트 디바이스의 스냅샷을 사용하여 AMI를 등록하는 방법

다음 `register-image` 예제는 EBS 루트 볼륨의 지정된 스냅샷을 디바이스로 사용하여 AMI를 등록합니다. `/dev/xvda` 블록 디바이스 매핑에는 빈 100GiB EBS 볼륨도 디바이스로 포함됩니다. `/dev/xvdf`

```
aws ec2 register-image \
    --name my-image \
    --root-device-name /dev/xvda \
    --block-device-mappings DeviceName=/dev/
xvda,Ebs={SnapshotId=snap-0db2cf683925d191f} DeviceName=/dev/
xvdf,Ebs={VolumeSize=100}
```

출력:

```
{
    "ImageId": "ami-1a2b3c4d5eEXAMPLE"
}
```

자세한 내용은 Amazon EC2 사용 설명서에서 Amazon Machine Image(AMI)를 참조하세요.

- API 세부 정보는 AWS CLI 명령 RegisterImage참조를 참조하십시오.

PowerShell

도구: PowerShell

예제 1: 이 예제에서는 Amazon S3의 지정된 매니페스트 파일을 사용하여 AMI를 등록합니다.

```
Register-EC2Image -ImageLocation my-s3-bucket/my-web-server-ami/
image.manifest.xml -Name my-web-server-ami
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 RegisterImage.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **RejectVpcPeeringConnection** CLI와 함께 사용

다음 코드 예제는 RejectVpcPeeringConnection의 사용 방법을 보여줍니다.

CLI

AWS CLI

VPC 피어링 연결을 거부하려면

이 예제는 지정된 VPC 피어링 연결 요청을 거부합니다.

명령:

```
aws ec2 reject-vpc-peering-connection --vpc-peering-connection-id pcx-1a2b3c4d
```

출력:

```
{
    "Return": true
}
```

- API 세부 정보는 명령 참조를 참조하십시오 RejectVpcPeeringConnection.AWS CLI

PowerShell

도구: PowerShell

예 1: 위 예시에서는 요청 ID pcx-01a2b3ce45fe67eb8에 대한 VpcPeering 요청을 거부합니다.

```
Deny-EC2VpcPeeringConnection -VpcPeeringConnectionId pcx-01a2b3ce45fe67eb8
```

- API에 RejectVpcPeeringConnection대한 AWS Tools for PowerShell 자세한 내용은 Cmdlet 참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ReleaseAddress** CLI와 함께 사용

다음 코드 예제는 ReleaseAddress의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 인스턴스 시작하기

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Release an Elastic IP address.
    /// </summary>
    /// <param name="allocationId">The allocation Id of the Elastic IP address.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> ReleaseAddress(string allocationId)
    {
        var request = new ReleaseAddressRequest
        {
            AllocationId = allocationId
        };

        var response = await _amazonEC2.ReleaseAddressAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- API 세부 정보는 AWS SDK for .NET API ReleaseAddress참조를 참조하십시오.

Bash

AWS CLI Bash 스크립트 사용

> **Note**
>
> 더 많은 정보가 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
#############################################################################
# function ec2_release_address
#
# This function releases an Elastic IP address from an Amazon Elastic Compute
 Cloud (Amazon EC2) instance.
#
# Parameters:
#       -a allocation_id - The allocation ID of the Elastic IP address to
 release.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
#
#############################################################################
function ec2_release_address() {
  local allocation_id response

  # Function to display usage information
  function usage() {
    echo "function ec2_release_address"
    echo "Releases an Elastic IP address from an Amazon Elastic Compute Cloud
 (Amazon EC2) instance."
    echo "  -a allocation_id - The allocation ID of the Elastic IP address to
 release."
    echo ""
  }

  # Parse the command-line arguments
  while getopts "a:h" option; do
    case "${option}" in
      a) allocation_id="${OPTARG}" ;;
```

```
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Validate the input parameters
  if [[ -z "$allocation_id" ]]; then
    errecho "ERROR: You must provide an allocation ID with the -a parameter."
    return 1
  fi

  response=$(aws ec2 release-address \
    --allocation-id "$allocation_id") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports release-address operation failed."
    errecho "$response"
    return 1
  }

  return 0
}
```

이 예제에 사용된 유틸리티 함수

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
```

```
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
################################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- API 세부 정보는 AWS CLI 명령 ReleaseAddress참조를 참조하십시오.

C++

SDK for C++

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
Aws::EC2::EC2Client ec2(clientConfiguration);

Aws::EC2::Model::ReleaseAddressRequest request;
request.SetAllocationId(allocationID);

auto outcome = ec2.ReleaseAddress(request);
if (!outcome.IsSuccess()) {
    std::cerr << "Failed to release Elastic IP address " <<
              allocationID << ":" << outcome.GetError().GetMessage() <<
              std::endl;
}
else {
    std::cout << "Successfully released Elastic IP address " <<
              allocationID << std::endl;
}
```

- API 세부 정보는 AWS SDK for C++ API ReleaseAddress참조를 참조하십시오.

CLI

AWS CLI

EC2-Classic의 탄력적 IP 주소를 해제하는 방법

자세한 내용은 EC2-Classic의 인스턴스에서 사용할 탄력적 IP 주소를 해제합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 release-address --public-ip 198.51.100.0
```

EC2-VPC의 탄력적 IP 주소를 해제하는 방법

이 예제에서는 VPC의 인스턴스에서 사용하도록 탄력적 IP 주소를 해제합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 release-address --allocation-id eipalloc-64d5890a
```

- API 세부 정보는 AWS CLI 명령 ReleaseAddress참조를 참조하십시오.

Java

SDK for Java 2.x

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    public static void releaseEC2Address(Ec2Client ec2, String allocId) {
        try {
            ReleaseAddressRequest request = ReleaseAddressRequest.builder()
                    .allocationId(allocId)
                    .build();

            ec2.releaseAddress(request);
            System.out.println("Successfully released Elastic IP address " +
allocId);
        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API ReleaseAddress참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { ReleaseAddressCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new ReleaseAddressCommand({
    // You can also use PublicIp, but that is for EC2 classic which is being
 retired.
    AllocationId: "ALLOCATION_ID",
  });

  try {
    await client.send(command);
    console.log("Successfully released address.");
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API ReleaseAddress참조를 참조하십시오.

Kotlin

SDK for Kotlin

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
suspend fun releaseEC2AddressSc(allocId: String?) {
    val request =
        ReleaseAddressRequest {
            allocationId = allocId
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.releaseAddress(request)
        println("Successfully released Elastic IP address $allocId")
    }
}
```

- API 세부 정보는 Kotlin API용AWS SDK 레퍼런스를 참조하세요 ReleaseAddress.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예제는 VPC의 인스턴스에 대해 지정된 엘라스틱 IP 주소를 해제합니다.

```
Remove-EC2Address -AllocationId eipalloc-12345678 -Force
```

예 2: 이 예제는 EC2-Classic의 인스턴스에 대해 지정된 엘라스틱 IP 주소를 릴리스합니다.

```
Remove-EC2Address -PublicIp 198.51.100.2 -Force
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 ReleaseAddress.

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class ElasticIpWrapper:
```

```
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) Elastic IP address
actions."""

    def __init__(self, ec2_resource, elastic_ip=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param elastic_ip: A Boto3 VpcAddress object. This is a high-level object
that
                           wraps Elastic IP actions.
        """
        self.ec2_resource = ec2_resource
        self.elastic_ip = elastic_ip

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def release(self):
        """
        Releases an Elastic IP address. After the Elastic IP address is released,
        it can no longer be used.
        """
        if self.elastic_ip is None:
            logger.info("No Elastic IP to release.")
            return

        try:
            self.elastic_ip.release()
        except ClientError as err:
            logger.error(
                "Couldn't release Elastic IP address %s. Here's why: %s: %s",
                self.elastic_ip.allocation_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 ReleaseAddress.

Ruby

SDK for Ruby

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
# Releases an Elastic IP address from an
# Amazon Elastic Compute Cloud (Amazon EC2) instance.
#
# Prerequisites:
#
# - An Amazon EC2 instance with an associated Elastic IP address.
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @param allocation_id [String] The ID of the allocation corresponding to
#   the Elastic IP address.
# @return [Boolean] true if the Elastic IP address was released;
#   otherwise, false.
# @example
#   exit 1 unless elastic_ip_address_released?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'eipalloc-04452e528a66279EX'
#   )
def elastic_ip_address_released?(ec2_client, allocation_id)
  ec2_client.release_address(allocation_id: allocation_id)
  return true
rescue StandardError => e
  puts("Error releasing Elastic IP address: #{e.message}")
  return false
end
```

- API 세부 정보는 AWS SDK for Ruby API ReleaseAddress참조를 참조하십시오.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    TRY.
        lo_ec2->releaseaddress( iv_allocationid = iv_allocation_id ).
        MESSAGE 'Elastic IP address released.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- API 세부 정보는 SAP용AWS SDK ABAP API 참조를 참조하십시오 ReleaseAddress.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ReleaseHosts** CLI와 함께 사용

다음 코드 예제는 ReleaseHosts의 사용 방법을 보여줍니다.

CLI

AWS CLI

계정에서 전용 호스트를 릴리스하려면

계정에서 전용 호스트를 해제하려면 호스트에 있는 인스턴스를 중지하거나 종료해야 호스트를 해제할 수 있습니다.

명령:

```
aws ec2 release-hosts --host-id=h-0029d6e3cacf1b3da
```

출력:

```
{
    "Successful":  [
        "h-0029d6e3cacf1b3da"
         ],
  "Unsuccessful": []

 }
```

- API 세부 정보는 AWS CLI 명령 ReleaseHosts참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 호스트 ID h-0badafd1dcb2f3456을 릴리스합니다.

```
Remove-EC2Host -HostId h-0badafd1dcb2f3456
```

출력:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-EC2Host (ReleaseHosts)" on target
 "h-0badafd1dcb2f3456".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): Y

Successful            Unsuccessful
----------            ------------
{h-0badafd1dcb2f3456} {}
```

- AWS Tools for PowerShell API에 ReleaseHosts대한 자세한 내용은 Cmdlet 참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ReplaceIamInstanceProfileAssociation** CLI와 함께 사용

다음 코드 예제는 ReplaceIamInstanceProfileAssociation의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 복원력이 뛰어난 서비스 구축 및 관리

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Replace the profile associated with a running instance. After the profile
is replaced, the instance
    /// is rebooted to ensure that it uses the new profile. When the instance is
ready, Systems Manager is
    /// used to restart the Python web server.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to update.</param>
    /// <param name="credsProfileName">The name of the new profile to associate
with the specified instance.</param>
    /// <param name="associationId">The Id of the existing profile association
for the instance.</param>
    /// <returns>Async task.</returns>
    public async Task ReplaceInstanceProfile(string instanceId, string
credsProfileName, string associationId)
    {
```

```
        await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
                new ReplaceIamInstanceProfileAssociationRequest()
                {
                    AssociationId = associationId,
                    IamInstanceProfile = new IamInstanceProfileSpecification()
                    {
                        Name = credsProfileName
                    }
                });
        // Allow time before resetting.
        Thread.Sleep(25000);
        var instanceReady = false;
        var retries = 5;
        while (retries-- > 0 && !instanceReady)
        {
            await _amazonEc2.RebootInstancesAsync(
                new RebootInstancesRequest(new List<string>() { instanceId }));
            Thread.Sleep(10000);

            var instancesPaginator =
_amazonSsm.Paginators.DescribeInstanceInformation(
                new DescribeInstanceInformationRequest());
            // Get the entire list using the paginator.
            await foreach (var instance in
instancesPaginator.InstanceInformationList)
            {
                instanceReady = instance.InstanceId == instanceId;
                if (instanceReady)
                {
                    break;
                }
            }
        }
        Console.WriteLine($"Sending restart command to instance {instanceId}");
        await _amazonSsm.SendCommandAsync(
            new SendCommandRequest()
            {
                InstanceIds = new List<string>() { instanceId },
                DocumentName = "AWS-RunShellScript",
                Parameters = new Dictionary<string, List<string>>()
                {
                        {"commands", new List<string>() { "cd / && sudo python3
server.py 80" }}
                }
```

```
            });
        Console.WriteLine($"Restarted the web server on instance {instanceId}");
    }
```

- API 세부 정보는 AWS SDK for .NET API ReplaceIamInstanceProfileAssociation참조를 참조하십시오.

CLI

AWS CLI

인스턴스에 대한 IAM 인스턴스 프로파일을 바꾸는 방법

이 예제에서는 `iip-assoc-060bae234aac2e7fa` 연결로 표시되는 IAM 인스턴스 프로파일을 이름이 `AdminRole`인 IAM 인스턴스 프로파일로 바꿉니다.

```
aws ec2 replace-iam-instance-profile-association \
    --iam-instance-profile Name=AdminRole \
    --association-id iip-assoc-060bae234aac2e7fa
```

출력:

```
{
    "IamInstanceProfileAssociation": {
        "InstanceId": "i-087711ddaf98f9489",
        "State": "associating",
        "AssociationId": "iip-assoc-0b215292fab192820",
        "IamInstanceProfile": {
            "Id": "AIPAJLNLDX3AMYZNWYYAY",
            "Arn": "arn:aws:iam::123456789012:instance-profile/AdminRole"
        }
    }
}
```

- API 세부 정보는 AWS CLI 명령 ReplaceIamInstanceProfileAssociation참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
  await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
    ec2Client.send(
      new ReplaceIamInstanceProfileAssociationCommand({
        AssociationId: state.instanceProfileAssociationId,
        IamInstanceProfile: { Name: NAMES.ssmOnlyInstanceProfileName },
      }),
    ),
  );
```

- API 세부 정보는 AWS SDK for JavaScript API ReplaceIamInstanceProfileAssociation참조를 참조하십시오.

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

이 예제에서는 실행 중인 인스턴스의 인스턴스 프로파일을 교체하고, 인스턴스을 재부팅하고, 인스턴스가 시작된 후 인스턴스에 명령을 보냅니다.

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """
```

```
    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def replace_instance_profile(
        self, instance_id, new_instance_profile_name, profile_association_id
    ):
        """
```

```
        Replaces the profile associated with a running instance. After the
profile is
        replaced, the instance is rebooted to ensure that it uses the new
profile. When
        the instance is ready, Systems Manager is used to restart the Python web
server.

        :param instance_id: The ID of the instance to update.
        :param new_instance_profile_name: The name of the new profile to
associate with
                                          the specified instance.
        :param profile_association_id: The ID of the existing profile association
for the
                                       instance.
        """
        try:
            self.ec2_client.replace_iam_instance_profile_association(
                IamInstanceProfile={"Name": new_instance_profile_name},
                AssociationId=profile_association_id,
            )
            log.info(
                "Replaced instance profile for association %s with profile %s.",
                profile_association_id,
                new_instance_profile_name,
            )
            time.sleep(5)
            inst_ready = False
            tries = 0
            while not inst_ready:
                if tries % 6 == 0:
                    self.ec2_client.reboot_instances(InstanceIds=[instance_id])
                    log.info(
                        "Rebooting instance %s and waiting for it to to be
ready.",
                        instance_id,
                    )
                tries += 1
                time.sleep(10)
                response = self.ssm_client.describe_instance_information()
                for info in response["InstanceInformationList"]:
                    if info["InstanceId"] == instance_id:
                        inst_ready = True
            self.ssm_client.send_command(
                InstanceIds=[instance_id],
```

```
                DocumentName="AWS-RunShellScript",
                Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
            )
            log.info("Restarted the Python web server on instance %s.",
instance_id)
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't replace instance profile for association
{profile_association_id}: {err}"
            )
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 ReplaceIamInstanceProfileAssociation.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ReplaceNetworkAclAssociation** CLI와 함께 사용

다음 코드 예제는 ReplaceNetworkAclAssociation의 사용 방법을 보여줍니다.

CLI

AWS CLI

서브넷과 연결된 네트워크 ACL을 대체하려면

이 예제에서는 지정된 네트워크 ACL을 지정된 네트워크 ACL 연결의 서브넷과 연결합니다.

명령:

```
aws ec2 replace-network-acl-association --association-id aclassoc-e5b95c8c --
network-acl-id acl-5fb85d36
```

출력:

```
{
```

```
    "NewAssociationId": "aclassoc-3999875b"
}
```

- API 세부 정보는 명령 참조를 참조하십시오 ReplaceNetworkAclAssociation.AWS CLI

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 네트워크 ACL을 지정된 네트워크 ACL 연결의 서브넷과 연결합니다.

```
Set-EC2NetworkAclAssociation -NetworkAclId acl-12345678 -AssociationId
 aclassoc-1a2b3c4d
```

출력:

```
aclassoc-87654321
```

- API에 대한 자세한 내용은 Cmdlet 참조를 참조하십시오 ReplaceNetworkAclAssociation.AWS Tools for PowerShell

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ReplaceNetworkAclEntry** CLI와 함께 사용

다음 코드 예제는 ReplaceNetworkAclEntry의 사용 방법을 보여줍니다.

CLI

AWS CLI

네트워크 ACL 항목을 바꾸려면

이 예제는 지정된 네트워크 ACL의 항목을 대체합니다. 새 규칙 100은 UDP 포트 53 (DNS) 의 203.0.113.12/24에서 모든 관련 서브넷으로의 인그레스 트래픽을 허용합니다.

명령:

```
aws ec2 replace-network-acl-entry --network-acl-id acl-5fb85d36 --ingress --rule-
number 100 --protocol udp --port-range From=53,To=53 --cidr-block 203.0.113.12/24
 --rule-action allow
```

- API ReplaceNetworkAclEntry세부AWS CLI 정보는 명령 참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 네트워크 ACL의 지정된 항목을 대체합니다. 새 규칙은 지정된 주소에서 연결된 모든 서브넷으로의 인바운드 트래픽을 허용합니다.

```
Set-EC2NetworkAclEntry -NetworkAclId acl-12345678 -Egress $false -RuleNumber 100
 -Protocol 17 -PortRange_From 53 -PortRange_To 53 -CidrBlock 203.0.113.12/24 -
RuleAction allow
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 ReplaceNetworkAclEntry.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ReplaceRoute** CLI와 함께 사용

다음 코드 예제는 ReplaceRoute의 사용 방법을 보여줍니다.

CLI

AWS CLI

라우트를 바꾸려면

이 예제는 지정된 라우팅 테이블의 지정된 경로를 대체합니다. 새 경로는 지정된 CIDR과 일치하고 트래픽을 지정된 가상 사설 게이트웨이로 보냅니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 replace-route --route-table-id rtb-22574640 --destination-cidr-block
 10.0.0.0/16 --gateway-id vgw-9a4cacf3
```

- API 세부 정보는 AWS CLI 명령 ReplaceRoute참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 라우팅 테이블의 지정된 경로를 대체합니다. 새 경로는 지정된 트래픽을 지정된 가상 프라이빗 게이트웨이로 보냅니다.

```
Set-EC2Route -RouteTableId rtb-1a2b3c4d -DestinationCidrBlock 10.0.0.0/24 -
GatewayId vgw-1a2b3c4d
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 ReplaceRoute.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ReplaceRouteTableAssociation** CLI와 함께 사용

다음 코드 예제는 ReplaceRouteTableAssociation의 사용 방법을 보여줍니다.

CLI

AWS CLI

서브넷에 연결된 라우팅 테이블을 교체하려면

이 예제는 지정된 라우팅 테이블을 지정된 라우팅 테이블 연결의 서브넷과 연결합니다.

명령:

```
aws ec2 replace-route-table-association --association-id rtbassoc-781d0d1a --
route-table-id rtb-22574640
```

출력:

```
{
    "NewAssociationId": "rtbassoc-3a1f0f58"
}
```

- API 세부 정보는 AWS CLI 명령 ReplaceRouteTableAssociation참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 라우팅 테이블을 지정된 라우팅 테이블 연결의 서브넷과 연결합니다.

```
Set-EC2RouteTableAssociation -RouteTableId rtb-1a2b3c4d -AssociationId
 rtbassoc-12345678
```

출력:

```
rtbassoc-87654321
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 ReplaceRouteTableAssociation.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ReportInstanceStatus** CLI와 함께 사용

다음 코드 예제는 ReportInstanceStatus의 사용 방법을 보여줍니다.

CLI

AWS CLI

인스턴스에 대한 상태 피드백을 보고하려면

이 예제 명령은 지정된 인스턴스에 대한 상태 피드백을 보고합니다.

명령:

```
aws ec2 report-instance-status --instances i-1234567890abcdef0 --status impaired
 --reason-codes unresponsive
```

- API 세부 정보는 AWS CLI 명령 ReportInstanceStatus참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 인스턴스에 대한 상태 피드백을 보고합니다.

```
Send-EC2InstanceStatus -Instance i-12345678 -Status impaired -ReasonCode
 unresponsive
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 ReportInstanceStatus.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **RequestSpotFleet** CLI와 함께 사용

다음 코드 예제는 RequestSpotFleet의 사용 방법을 보여줍니다.

CLI

AWS CLI

가격이 가장 낮은 서브넷의 스팟 플릿을 요청하려면

이 예제 명령은 서브넷별로만 다른 두 개의 시작 사양을 사용하여 스팟 플릿 요청을 생성합니다. 스팟 플릿은 지정된 서브넷에서 최저 가격으로 인스턴스를 시작합니다. 인스턴스가 기본 VPC에서 시작되는 경우 기본적으로 퍼블릭 IP 주소를 받습니다. 인스턴스가 기본이 아닌 VPC로 시작되는 경우, 인스턴스는 기본적으로 퍼블릭 IP 주소를 받지 않습니다.

참고로 스팟 플릿 요청에서는 동일한 가용 영역의 서브넷을 다르게 지정할 수 없습니다.

명령:

```
aws ec2 request-spot-fleet --spot-fleet-request-config file://config.json
```

Config.json:

```
{
  "SpotPrice": "0.04",
  "TargetCapacity": 2,
  "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "KeyName": "my-key-pair",
          "SecurityGroups": [
              {
                  "GroupId": "sg-1a2b3c4d"
              }
          ],
          "InstanceType": "m3.medium",
          "SubnetId": "subnet-1a2b3c4d, subnet-3c4d5e6f",
          "IamInstanceProfile": {
              "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
          }
      }
  ]
}
```

출력:

```
{
  "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE"
}
```

가용 영역에서 최저 가격의 스팟 플릿을 요청하려면

이 예제 명령은 가용 영역에서만 다른 두 개의 시작 사양을 사용하여 스팟 플릿 요청을 생성합니다. 스팟 플릿은 지정된 가용 영역에서 최저 가격으로 인스턴스를 시작합니다. 계정이 EC2-VPC 전용을 지원하는 경우 Amazon EC2는 가용 영역의 기본 서브넷에서 스팟 인스턴스를 시작합니다. 계정이 EC2-Classic을 지원하는 경우 Amazon EC2는 가용 영역에서 EC2-Classic의 인스턴스를 시작합니다.

명령:

```
aws ec2 request-spot-fleet --spot-fleet-request-config file://config.json
```

Config.json:

```
{
  "SpotPrice": "0.04",
  "TargetCapacity": 2,
  "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "KeyName": "my-key-pair",
          "SecurityGroups": [
              {
                  "GroupId": "sg-1a2b3c4d"
              }
          ],
          "InstanceType": "m3.medium",
          "Placement": {
              "AvailabilityZone": "us-west-2a, us-west-2b"
          },
          "IamInstanceProfile": {
              "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
          }
      }
  ]
}
```

서브넷에서 스팟 인스턴스를 시작하고 퍼블릭 IP 주소를 할당하려면

이 예제 명령은 기본이 아닌 VPC에서 시작된 인스턴스에 퍼블릭 주소를 할당합니다. 단, 네트워크 인터페이스를 지정할 때는 네트워크 인터페이스를 사용하여 서브넷 ID와 보안 그룹 ID를 포함해야 합니다.

명령:

```
aws ec2 request-spot-fleet --spot-fleet-request-config file://config.json
```

Config.json:

```
{
```

```
  "SpotPrice": "0.04",
  "TargetCapacity": 2,
  "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "KeyName": "my-key-pair",
          "InstanceType": "m3.medium",
          "NetworkInterfaces": [
              {
                  "DeviceIndex": 0,
                  "SubnetId": "subnet-1a2b3c4d",
                  "Groups": [ "sg-1a2b3c4d" ],
                  "AssociatePublicIpAddress": true
              }
          ],
          "IamInstanceProfile": {
              "Arn": "arn:aws:iam::880185128111:instance-profile/my-iam-role"
          }
      }
  ]
}
```

다양한 할당 전략을 사용하여 스팟 플릿을 요청하려면

이 예제 명령은 분산 할당 전략을 사용하여 30개의 인스턴스를 시작하는 스팟 플릿 요청을 생성합니다. 시작 사양은 인스턴스 유형별로 다릅니다. 스팟 플릿은 시작 사양에 따라 인스턴스를 분산하여 각 유형마다 10개의 인스턴스가 있도록 합니다.

명령:

```
aws ec2 request-spot-fleet --spot-fleet-request-config file://config.json
```

Config.json:

```
{
  "SpotPrice": "0.70",
  "TargetCapacity": 30,
  "AllocationStrategy": "diversified",
  "IamFleetRole": "arn:aws:iam::123456789012:role/my-spot-fleet-role",
  "LaunchSpecifications": [
      {
```

```
            "ImageId": "ami-1a2b3c4d",
            "InstanceType": "c4.2xlarge",
            "SubnetId": "subnet-1a2b3c4d"
        },
        {
            "ImageId": "ami-1a2b3c4d",
            "InstanceType": "m3.2xlarge",
            "SubnetId": "subnet-1a2b3c4d"
        },
        {
            "ImageId": "ami-1a2b3c4d",
            "InstanceType": "r3.2xlarge",
            "SubnetId": "subnet-1a2b3c4d"
        }
    ]
}
```

자세한 내용은 Amazon Elastic Compute 클라우드 사용 설명서의 스팟 플릿 요청을 참조하십시오.

- API 세부 정보는 AWS CLI 명령 RequestSpotFleet참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 인스턴스 유형에 대해 최저 가격으로 가용 영역에 스팟 플릿 요청을 생성합니다. 계정이 EC2-VPC 전용을 지원하는 경우 스팟 플릿은 기본 서브넷이 있는 최저 가격의 가용 영역에서 인스턴스를 시작합니다. 계정이 EC2-Classic을 지원하는 경우 스팟 플릿은 최저 가격의 가용 영역에서 EC2-Classic의 인스턴스를 시작합니다. 지불하는 가격은 요청에 지정된 스팟 가격을 초과하지 않는다는 점에 유의하십시오.

```
$sg = New-Object Amazon.EC2.Model.GroupIdentifier
$sg.GroupId = "sg-12345678"
$lc = New-Object Amazon.EC2.Model.SpotFleetLaunchSpecification
$lc.ImageId = "ami-12345678"
$lc.InstanceType = "m3.medium"
$lc.SecurityGroups.Add($sg)
Request-EC2SpotFleet -SpotFleetRequestConfig_SpotPrice 0.04 `
-SpotFleetRequestConfig_TargetCapacity 2 `
-SpotFleetRequestConfig_IamFleetRole arn:aws:iam::123456789012:role/my-spot-
fleet-role `
```

```
-SpotFleetRequestConfig_LaunchSpecification $lc
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 RequestSpotFleet.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **RequestSpotInstances** CLI와 함께 사용

다음 코드 예제는 RequestSpotInstances의 사용 방법을 보여줍니다.

CLI

AWS CLI

스팟 인스턴스를 요청하려면

이 예제 명령은 지정된 가용 영역의 5개 인스턴스에 대한 일회성 스팟 인스턴스 요청을 생성합니다. 계정이 EC2-VPC 전용을 지원하는 경우 Amazon EC2는 지정된 가용 영역의 기본 서브넷에서 인스턴스를 시작합니다. 계정이 EC2-Classic을 지원하는 경우 Amazon EC2는 지정된 가용 영역에서 EC2-Classic의 인스턴스를 시작합니다.

명령:

```
aws ec2 request-spot-instances --spot-price "0.03" --instance-count 5 --type
 "one-time" --launch-specification file://specification.json
```

사양.json:

```
{
  "ImageId": "ami-1a2b3c4d",
  "KeyName": "my-key-pair",
  "SecurityGroupIds": [ "sg-1a2b3c4d" ],
  "InstanceType": "m3.medium",
  "Placement": {
    "AvailabilityZone": "us-west-2a"
  },
  "IamInstanceProfile": {
      "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
  }
```

```
}
```

출력:

```
{
  "SpotInstanceRequests": [
      {
          "Status": {
              "UpdateTime": "2014-03-25T20:54:21.000Z",
              "Code": "pending-evaluation",
              "Message": "Your Spot request has been submitted for review, and is
 pending evaluation."
          },
          "ProductDescription": "Linux/UNIX",
          "SpotInstanceRequestId": "sir-df6f405d",
          "State": "open",
          "LaunchSpecification": {
              "Placement": {
                  "AvailabilityZone": "us-west-2a"
              },
              "ImageId": "ami-1a2b3c4d",
              "KeyName": "my-key-pair",
              "SecurityGroups": [
                  {
                      "GroupName": "my-security-group",
                      "GroupId": "sg-1a2b3c4d"
                  }
              ],
              "Monitoring": {
                  "Enabled": false
              },
              "IamInstanceProfile": {
                  "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
              },
              "InstanceType": "m3.medium"
          },
          "Type": "one-time",
          "CreateTime": "2014-03-25T20:54:20.000Z",
          "SpotPrice": "0.050000"
      },
      ...
  ]
}
```

이 예제 명령은 지정된 서브넷의 5개 인스턴스에 대한 일회성 스팟 인스턴스 요청을 생성합니다. Amazon EC2는 지정된 서브넷에서 인스턴스를 시작합니다. VPC가 기본 VPC가 아닌 경우 인스턴스는 기본적으로 퍼블릭 IP 주소를 받지 않습니다.

명령:

```
aws ec2 request-spot-instances --spot-price "0.050" --instance-count 5 --type
 "one-time" --launch-specification file://specification.json
```

명세서.json:

```
{
  "ImageId": "ami-1a2b3c4d",
  "SecurityGroupIds": [ "sg-1a2b3c4d" ],
  "InstanceType": "m3.medium",
  "SubnetId": "subnet-1a2b3c4d",
  "IamInstanceProfile": {
      "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
  }
}
```

출력:

```
{
  "SpotInstanceRequests": [
      {
          "Status": {
             "UpdateTime": "2014-03-25T22:21:58.000Z",
             "Code": "pending-evaluation",
             "Message": "Your Spot request has been submitted for review, and is
 pending evaluation."
          },
          "ProductDescription": "Linux/UNIX",
          "SpotInstanceRequestId": "sir-df6f405d",
          "State": "open",
          "LaunchSpecification": {
             "Placement": {
                 "AvailabilityZone": "us-west-2a"
             }
             "ImageId": "ami-1a2b3c4d"
             "SecurityGroups": [
                 {
```

```
                    "GroupName": "my-security-group",
                    "GroupID": "sg-1a2b3c4d"
                }
            ]
            "SubnetId": "subnet-1a2b3c4d",
            "Monitoring": {
                "Enabled": false
            },
            "IamInstanceProfile": {
                "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
            },
            "InstanceType": "m3.medium",
        },
        "Type": "one-time",
        "CreateTime": "2014-03-25T22:21:58.000Z",
        "SpotPrice": "0.050000"
      },
      ...
  ]
}
```

이 예제에서는 기본이 아닌 VPC에서 시작하는 스팟 인스턴스에 퍼블릭 IP 주소를 할당합니다. 네트워크 인터페이스를 지정할 때는 네트워크 인터페이스를 사용하여 서브넷 ID와 보안 그룹 ID를 포함해야 합니다.

명령:

```
aws ec2 request-spot-instances --spot-price "0.050" --instance-count 1 --type
 "one-time" --launch-specification file://specification.json
```

사양.json:

```
{
  "ImageId": "ami-1a2b3c4d",
  "KeyName": "my-key-pair",
  "InstanceType": "m3.medium",
  "NetworkInterfaces": [
    {
      "DeviceIndex": 0,
      "SubnetId": "subnet-1a2b3c4d",
      "Groups": [ "sg-1a2b3c4d" ],
      "AssociatePublicIpAddress": true
```

```
    }
  ],
  "IamInstanceProfile": {
      "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
  }
}
```

- API에 대한 자세한 내용은 명령 참조를 참조하십시오 RequestSpotInstances.AWS CLI

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 서브넷에서 일회용 스팟 인스턴스를 요청합니다. 보안 그룹은 지정된 서브넷이 포함된 VPC에 대해 생성되어야 하며, 네트워크 인터페이스를 사용하여 ID로 지정해야 합니다. 네트워크 인터페이스를 지정할 때는 네트워크 인터페이스를 사용하여 서브넷 ID를 포함해야 합니다.

```
$n = New-Object Amazon.EC2.Model.InstanceNetworkInterfaceSpecification
$n.DeviceIndex = 0
$n.SubnetId = "subnet-12345678"
$n.Groups.Add("sg-12345678")
Request-EC2SpotInstance -InstanceCount 1 -SpotPrice 0.050 -Type one-time `
-IamInstanceProfile_Arn arn:aws:iam::123456789012:instance-profile/my-iam-role `
-LaunchSpecification_ImageId ami-12345678 `
-LaunchSpecification_InstanceType m3.medium `
-LaunchSpecification_NetworkInterface $n
```

출력:

```
ActualBlockHourlyPrice   :
AvailabilityZoneGroup    :
BlockDurationMinutes     : 0
CreateTime               : 12/26/2015 7:44:10 AM
Fault                    :
InstanceId               :
LaunchedAvailabilityZone :
LaunchGroup              :
LaunchSpecification      : Amazon.EC2.Model.LaunchSpecification
ProductDescription       : Linux/UNIX
SpotInstanceRequestId    : sir-12345678
SpotPrice                : 0.050000
```

```
State                       : open
Status                      : Amazon.EC2.Model.SpotInstanceStatus
Tags                        : {}
Type                        : one-time
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 RequestSpotInstances.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ResetImageAttribute** CLI와 함께 사용

다음 코드 예제는 ResetImageAttribute의 사용 방법을 보여줍니다.

CLI

AWS CLI

LaunchPermission 속성을 재설정하려면

이 예제에서는 지정된 AMI의 launchPermission 속성을 기본값으로 재설정합니다. 기본적으로 AMI는 비공개입니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 reset-image-attribute --image-id ami-5731123e --attribute
 launchPermission
```

- API 세부 정보는 AWS CLI 명령 ResetImageAttribute참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제에서는 'LaunchPermission' 속성을 기본값으로 재설정합니다. 기본적으로 AMI는 비공개입니다.

```
Reset-EC2ImageAttribute -ImageId ami-12345678 -Attribute launchPermission
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 ResetImageAttribute.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ResetInstanceAttribute** CLI와 함께 사용

다음 코드 예제는 ResetInstanceAttribute의 사용 방법을 보여줍니다.

CLI

AWS CLI

sourceDestCheck 속성을 재설정하려면

이 예제는 지정된 인스턴스의 sourceDestCheck 어트리뷰트를 재설정합니다. 인스턴스는 VPC에 있어야 합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 reset-instance-attribute --instance-id i-1234567890abcdef0 --attribute
 sourceDestCheck
```

커널 속성을 재설정하려면

이 예제는 지정된 인스턴스의 kernel 어트리뷰트를 재설정합니다. 인스턴스는 stopped 상태여야 합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 reset-instance-attribute --instance-id i-1234567890abcdef0 --attribute
 kernel
```

ramdisk 속성을 재설정하려면

이 예제는 지정된 인스턴스의 ramdisk 어트리뷰트를 재설정합니다. 인스턴스는 stopped 상태여야 합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 reset-instance-attribute --instance-id i-1234567890abcdef0 --attribute
 ramdisk
```

- API 세부 정보는 AWS CLI 명령 ResetInstanceAttribute참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 인스턴스의 sriovNetSupport '' 속성을 재설정합니다.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute sriovNetSupport
```

예 2: 이 예제에서는 지정된 인스턴스의 'EBSOptimized' 속성을 재설정합니다.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute ebsOptimized
```

예 3: 이 예제에서는 지정된 인스턴스의 'sourceDestCheck' 속성을 재설정합니다.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute sourceDestCheck
```

예 4: 이 예제에서는 지정된 인스턴스의 disableApiTermination '' 속성을 재설정합니다.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute
 disableApiTermination
```

예 5: 이 예제는 지정된 인스턴스의 'instanceInitiatedShutdownBehavior' 속성을 재설정합니다.

```
Reset-EC2InstanceAttribute -InstanceId i-12345678 -Attribute
 instanceInitiatedShutdownBehavior
```

- API 세부 정보는 Cmdlet 참조를 참조하십시오 ResetInstanceAttribute.AWS Tools for PowerShell

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ResetNetworkInterfaceAttribute** CLI와 함께 사용

다음 코드 예제는 ResetNetworkInterfaceAttribute의 사용 방법을 보여줍니다.

CLI

AWS CLI

네트워크 인터페이스 속성을 재설정하려면

다음 reset-network-interface-attribute 예제에서는 소스/대상 확인 속성의 값을 로 재설정합니다. true

```
aws ec2 reset-network-interface-attribute \
    --network-interface-id eni-686ea200 \
    --source-dest-check
```

이 명령은 출력을 생성하지 않습니다.

- API 세부 정보는 명령 참조를 참조하십시오 ResetNetworkInterfaceAttribute.AWS CLI

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 네트워크 인터페이스의 원본/대상 검사를 재설정합니다.

```
Reset-EC2NetworkInterfaceAttribute -NetworkInterfaceId eni-1a2b3c4d -
SourceDestCheck
```

- API 세부 정보는 Cmdlet 참조를 참조하십시오 ResetNetworkInterfaceAttribute.AWS Tools for PowerShell

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **ResetSnapshotAttribute** CLI와 함께 사용

다음 코드 예제는 ResetSnapshotAttribute의 사용 방법을 보여줍니다.

CLI

AWS CLI

스냅샷 속성을 재설정하려면

이 예제에서는 `snap-1234567890abcdef0` 스냅샷에 대한 볼륨 생성 권한을 재설정합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 reset-snapshot-attribute --snapshot-id snap-1234567890abcdef0 --attribute
 createVolumePermission
```

- API 세부 정보는 AWS CLI 명령 ResetSnapshotAttribute참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 스냅샷의 지정된 속성을 재설정합니다.

```
Reset-EC2SnapshotAttribute -SnapshotId snap-12345678 -Attribute
 CreateVolumePermission
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 ResetSnapshotAttribute.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **RevokeSecurityGroupEgress** CLI와 함께 사용

다음 코드 예제는 `RevokeSecurityGroupEgress`의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 특정 주소 범위로의 아웃바운드 트래픽을 허용하는 규칙을 제거하려면

다음 `revoke-security-group-egress` 예제 명령은 TCP 포트 80에서 지정된 주소 범위에 대한 액세스 권한을 부여하는 규칙을 제거합니다.

```
aws ec2 revoke-security-group-egress \
    --group-id sg-026c12253ce15eff7 \
    --ip-permissions
 [{IpProtocol=tcp,FromPort=80,ToPort=80,IpRanges=[{CidrIp=10.0.0.0/16}]
```

이 명령은 출력을 생성하지 않습니다.

자세한 내용은 Amazon EC2 사용 설명서의 보안 그룹을 참조하십시오.

예 2: 특정 보안 그룹으로의 아웃바운드 트래픽을 허용하는 규칙을 제거하려면

다음 `revoke-security-group-egress` 예제 명령은 TCP 포트 80에서 지정된 보안 그룹에 대한 액세스 권한을 부여하는 규칙을 제거합니다.

```
aws ec2 revoke-security-group-egress \
    --group-id sg-026c12253ce15eff7 \
    --ip-permissions '[{"IpProtocol": "tcp", "FromPort": 443, "ToPort":
 443,"UserIdGroupPairs": [{"GroupId": "sg-06df23a01ff2df86d"}]}]'
```

이 명령은 출력을 생성하지 않습니다.

자세한 내용은 Amazon EC2 사용 설명서의 보안 그룹을 참조하십시오.

- API 세부 정보는 AWS CLI 명령 RevokeSecurityGroupEgress참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 EC2-VPC 보안 그룹에 대한 규칙을 제거합니다. 이렇게 하면 TCP 포트 80에서 지정된 IP 주소 범위에 대한 액세스가 취소됩니다. 이 예제에서 사용하는 구문에는 PowerShell 버전 3 이상이 필요합니다.

```
$ip = @{ IpProtocol="tcp"; FromPort="80"; ToPort="80";
 IpRanges="203.0.113.0/24" }
Revoke-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission $ip
```

예 2: PowerShell 버전 2에서는 New-Object를 사용하여 객체를 생성해야 합니다. IpPermission

```
$ip = New-Object Amazon.EC2.Model.IpPermission
$ip.IpProtocol = "tcp"
$ip.FromPort = 80
$ip.ToPort = 80
$ip.IpRanges.Add("203.0.113.0/24")
Revoke-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission $ip
```

예 3: 이 예에서는 TCP 포트 80에서 지정된 소스 보안 그룹에 대한 액세스를 취소합니다.

```
$ug = New-Object Amazon.EC2.Model.UserIdGroupPair
$ug.GroupId = "sg-1a2b3c4d"
$ug.UserId = "123456789012"
Revoke-EC2SecurityGroupEgress -GroupId sg-12345678 -IpPermission
 @( @{ IpProtocol="tcp"; FromPort="80"; ToPort="80"; UserIdGroupPairs=$ug } )
```

- API 세부 정보는 Cmdlet 참조를 참조하십시오 RevokeSecurityGroupEgress.AWS Tools for PowerShell

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **RevokeSecurityGroupIngress** CLI와 함께 사용

다음 코드 예제는 RevokeSecurityGroupIngress의 사용 방법을 보여줍니다.

CLI

AWS CLI

예 1: 보안 그룹에서 규칙 제거하기

다음 revoke-security-group-ingress 예에서는 기본 VPC의 지정된 보안 그룹에서 203.0.113.0/24 주소 범위에 대한 TCP 포트 22 액세스를 제거합니다.

```
aws ec2 revoke-security-group-ingress \
    --group-name mySecurityGroup
    --protocol tcp \
    --port 22 \
    --cidr 203.0.113.0/24
```

이 명령은 성공해도 출력을 생성하지 않습니다.

자세한 내용은 Amazon EC2 사용 설명서의 보안 그룹을 참조하십시오.

예 2: IP 권한 세트를 사용하여 규칙을 제거하려면

다음 `revoke-security-group-ingress` 예에서는 `ip-permissions` 매개변수를 사용하여 ICMP 메시지를 허용하는 인바운드 규칙 `Destination Unreachable: Fragmentation Needed and Don't Fragment was Set` (유형 3, 코드 4) 을 제거합니다.

```
aws ec2 revoke-security-group-ingress \
    --group-id sg-026c12253ce15eff7 \
    --ip-permissions
 IpProtocol=icmp,FromPort=3,ToPort=4,IpRanges=[{CidrIp=0.0.0.0/0}]
```

이 명령은 성공해도 출력을 생성하지 않습니다.

자세한 내용은 Amazon EC2 사용 설명서의 보안 그룹을 참조하십시오.

- API 세부 정보는 AWS CLI 명령 RevokeSecurityGroupIngress참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 EC2-VPC 보안 그룹의 지정된 주소 범위에서 TCP 포트 22에 대한 액세스를 취소합니다. 보안 그룹 이름이 아닌 보안 그룹 ID를 사용하여 EC2-VPC 보안 그룹을 식별해야 한다는 점에 유의하십시오. 이 예제에서 사용하는 구문에는 PowerShell 버전 3 이상이 필요합니다.

```
$ip = @{ IpProtocol="tcp"; FromPort="22"; ToPort="22";
 IpRanges="203.0.113.0/24" }
Revoke-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission $ip
```

예 2: PowerShell 버전 2에서는 New-Object를 사용하여 객체를 생성해야 합니다. IpPermission

```
$ip = New-Object Amazon.EC2.Model.IpPermission
$ip.IpProtocol = "tcp"
$ip.FromPort = 22
$ip.ToPort = 22
```

```
$ip.IpRanges.Add("203.0.113.0/24")

Revoke-EC2SecurityGroupIngress -GroupId sg-12345678 -IpPermission $ip
```

예 3: 이 예에서는 EC2-Classic의 지정된 보안 그룹에 대해 지정된 주소 범위에서 TCP 포트 22에 대한 액세스를 취소합니다. 이 예제에서 사용하는 구문에는 버전 3 이상이 필요합니다 PowerShell .

```
$ip = @{ IpProtocol="tcp"; FromPort="22"; ToPort="22";
 IpRanges="203.0.113.0/24" }

Revoke-EC2SecurityGroupIngress -GroupName "my-security-group" -IpPermission $ip
```

예 4: PowerShell 버전 2에서는 New-Object를 사용하여 객체를 생성해야 합니다. IpPermission

```
$ip = New-Object Amazon.EC2.Model.IpPermission
$ip.IpProtocol = "tcp"
$ip.FromPort = 22
$ip.ToPort = 22
$ip.IpRanges.Add("203.0.113.0/24")

Revoke-EC2SecurityGroupIngress -GroupName "my-security-group" -IpPermission $ip
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 RevokeSecurityGroupIngress.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **RunInstances** CLI와 함께 사용

다음 코드 예제는 RunInstances의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 인스턴스 시작하기

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Create and run an EC2 instance.
    /// </summary>
    /// <param name="ImageId">The image Id of the image used as a basis for the
    /// EC2 instance.</param>
    /// <param name="instanceType">The instance type of the EC2 instance to
create.</param>
    /// <param name="keyName">The name of the key pair to associate with the
    /// instance.</param>
    /// <param name="groupId">The Id of the Amazon EC2 security group that will
be
    /// allowed to interact with the new EC2 instance.</param>
    /// <returns>The instance Id of the new EC2 instance.</returns>
    public async Task<string> RunInstances(string imageId, string instanceType,
string keyName, string groupId)
    {
        var request = new RunInstancesRequest
        {
            ImageId = imageId,
            InstanceType = instanceType,
            KeyName = keyName,
            MinCount = 1,
            MaxCount = 1,
            SecurityGroupIds = new List<string> { groupId }
        };
        var response = await _amazonEC2.RunInstancesAsync(request);
        return response.Reservation.Instances[0].InstanceId;
    }
```

- API 세부 정보는 AWS SDK for .NET API RunInstances참조를 참조하십시오.

Bash

AWS CLI Bash 스크립트 사용

> **Note**
>
> 더 많은 정보가 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
###############################################################################
# function ec2_run_instances
#
# This function launches one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i image_id - The ID of the Amazon Machine Image (AMI) to use.
#       -t instance_type - The instance type to use (e.g., t2.micro).
#       -k key_pair_name - The name of the key pair to use.
#       -s security_group_id - The ID of the security group to use.
#       -c count - The number of instances to launch (default: 1).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_run_instances() {
  local image_id instance_type key_pair_name security_group_id count response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_run_instances"
    echo "Launches one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances."
    echo "  -i image_id - The ID of the Amazon Machine Image (AMI) to use."
    echo "  -t instance_type - The instance type to use (e.g., t2.micro)."
    echo "  -k key_pair_name - The name of the key pair to use."
    echo "  -s security_group_id - The ID of the security group to use."
    echo "  -c count - The number of instances to launch (default: 1)."
```

```
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:t:k:s:c:h" option; do
    case "${option}" in
      i) image_id="${OPTARG}" ;;
      t) instance_type="${OPTARG}" ;;
      k) key_pair_name="${OPTARG}" ;;
      s) security_group_id="${OPTARG}" ;;
      c) count="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$image_id" ]]; then
    errecho "ERROR: You must provide an Amazon Machine Image (AMI) ID with the -i
parameter."
    usage
    return 1
  fi

  if [[ -z "$instance_type" ]]; then
    errecho "ERROR: You must provide an instance type with the -t parameter."
    usage
    return 1
  fi

  if [[ -z "$key_pair_name" ]]; then
    errecho "ERROR: You must provide a key pair name with the -k parameter."
    usage
    return 1
  fi
```

```
  if [[ -z "$security_group_id" ]]; then
    errecho "ERROR: You must provide a security group ID with the -s parameter."
    usage
    return 1
  fi

  if [[ -z "$count" ]]; then
    count=1
  fi

  response=$(aws ec2 run-instances \
    --image-id "$image_id" \
    --instance-type "$instance_type" \
    --key-name "$key_pair_name" \
    --security-group-ids "$security_group_id" \
    --count "$count" \
    --query 'Instances[*].[InstanceId]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports run-instances operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}
```

이 예제에 사용된 유틸리티 함수

```
################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

################################################################################
# function aws_cli_error_log()
#
```

```
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
########################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- API 세부 정보는 AWS CLI 명령 RunInstances참조를 참조하십시오.

C++

SDK for C++

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);

    Aws::EC2::Model::RunInstancesRequest runRequest;
    runRequest.SetImageId(amiId);
    runRequest.SetInstanceType(Aws::EC2::Model::InstanceType::t1_micro);
    runRequest.SetMinCount(1);
    runRequest.SetMaxCount(1);

    Aws::EC2::Model::RunInstancesOutcome runOutcome = ec2Client.RunInstances(
            runRequest);
    if (!runOutcome.IsSuccess()) {
        std::cerr << "Failed to launch EC2 instance " << instanceName <<
                  " based on ami " << amiId << ":" <<
                  runOutcome.GetError().GetMessage() << std::endl;
        return false;
    }

    const Aws::Vector<Aws::EC2::Model::Instance> &instances =
runOutcome.GetResult().GetInstances();
    if (instances.empty()) {
        std::cerr << "Failed to launch EC2 instance " << instanceName <<
                  " based on ami " << amiId << ":" <<
                  runOutcome.GetError().GetMessage() << std::endl;
        return false;
    }

    instanceID = instances[0].GetInstanceId();
```

- API 세부 정보는 AWS SDK for C++ API RunInstances참조를 참조하십시오.

CLI

AWS CLI

예제 1: 기본 서브넷에서 인스턴스를 시작하는 방법

다음 `run-instances` 예제에서는 현재 리전의 기본 서브넷에서 `t2.micro` 유형의 단일 인스턴스를 시작하고 이를 해당 리전에서 기본 VPC에 대한 기본 서브넷에 연결합니다. 키 페어는 SSH(Linux) 또는 RDP(Windows)를 사용하여 인스턴스에 연결할 계획이 없는 경우 선택 사항입니다.

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type t2.micro \
    --key-name MyKeyPair
```

출력:

```
{
    "Instances": [
        {
            "AmiLaunchIndex": 0,
            "ImageId": "ami-0abcdef1234567890",
            "InstanceId": "i-1231231230abcdef0",
            "InstanceType": "t2.micro",
            "KeyName": "MyKeyPair",
            "LaunchTime": "2018-05-10T08:05:20.000Z",
            "Monitoring": {
                "State": "disabled"
            },
            "Placement": {
                "AvailabilityZone": "us-east-2a",
                "GroupName": "",
                "Tenancy": "default"
            },
            "PrivateDnsName": "ip-10-0-0-157.us-east-2.compute.internal",
            "PrivateIpAddress": "10.0.0.157",
            "ProductCodes": [],
            "PublicDnsName": "",
            "State": {
                "Code": 0,
                "Name": "pending"
            },
            "StateTransitionReason": "",
            "SubnetId": "subnet-04a636d18e83cfacb",
            "VpcId": "vpc-1234567890abcdef0",
            "Architecture": "x86_64",
            "BlockDeviceMappings": [],
            "ClientToken": "",
            "EbsOptimized": false,
            "Hypervisor": "xen",
            "NetworkInterfaces": [
                {
                    "Attachment": {
```

```
                        "AttachTime": "2018-05-10T08:05:20.000Z",
                        "AttachmentId": "eni-attach-0e325c07e928a0405",
                        "DeleteOnTermination": true,
                        "DeviceIndex": 0,
                        "Status": "attaching"
                    },
                    "Description": "",
                    "Groups": [
                        {
                            "GroupName": "MySecurityGroup",
                            "GroupId": "sg-0598c7d356eba48d7"
                        }
                    ],
                    "Ipv6Addresses": [],
                    "MacAddress": "0a:ab:58:e0:67:e2",
                    "NetworkInterfaceId": "eni-0c0a29997760baee7",
                    "OwnerId": "123456789012",
                    "PrivateDnsName": "ip-10-0-0-157.us-east-2.compute.internal",
                    "PrivateIpAddress": "10.0.0.157",
                    "PrivateIpAddresses": [
                        {
                            "Primary": true,
                            "PrivateDnsName": "ip-10-0-0-157.us-
east-2.compute.internal",
                            "PrivateIpAddress": "10.0.0.157"
                        }
                    ],
                    "SourceDestCheck": true,
                    "Status": "in-use",
                    "SubnetId": "subnet-04a636d18e83cfacb",
                    "VpcId": "vpc-1234567890abcdef0",
                    "InterfaceType": "interface"
                }
            ],
            "RootDeviceName": "/dev/xvda",
            "RootDeviceType": "ebs",
            "SecurityGroups": [
                {
                    "GroupName": "MySecurityGroup",
                    "GroupId": "sg-0598c7d356eba48d7"
                }
            ],
            "SourceDestCheck": true,
            "StateReason": {
```

```
                "Code": "pending",
                "Message": "pending"
            },
            "Tags": [],
            "VirtualizationType": "hvm",
            "CpuOptions": {
                "CoreCount": 1,
                "ThreadsPerCore": 1
            },
            "CapacityReservationSpecification": {
                "CapacityReservationPreference": "open"
            },
            "MetadataOptions": {
                "State": "pending",
                "HttpTokens": "optional",
                "HttpPutResponseHopLimit": 1,
                "HttpEndpoint": "enabled"
            }
        }
    ],
    "OwnerId": "123456789012",
    "ReservationId": "r-02a3f596d91211712"
}
```

예제 2: 기본이 아닌 서브넷에서 인스턴스를 시작하고 퍼블릭 IP 주소를 추가하는 방법

다음 `run-instances` 예제에서는 기본이 아닌 서브넷에서 시작하는 인스턴스에 대해 퍼블릭 IP 주소를 요청합니다. 인스턴스는 지정된 보안 그룹에 연결됩니다.

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type t2.micro \
    --subnet-id subnet-08fc749671b2d077c \
    --security-group-ids sg-0b0384b66d7d692f9 \
    --associate-public-ip-address \
    --key-name MyKeyPair
```

`run-instances` 출력 예제는 예제 1을 참조하세요.

예제 3: 추가 볼륨이 포함된 인스턴스를 시작하는 방법

다음 run-instances 예제에서는 시작할 때 추가 볼륨을 연결하도록 mapping.json에 지정된 블록 디바이스 매핑을 사용합니다. 블록 디바이스 매핑은 EBS 볼륨, 인스턴스 스토어 볼륨 또는 EBS 볼륨 및 인스턴스 스토어 볼륨 모두를 지정할 수 있습니다.

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type t2.micro \
    --subnet-id subnet-08fc749671b2d077c \
    --security-group-ids sg-0b0384b66d7d692f9 \
    --key-name MyKeyPair \
    --block-device-mappings file://mapping.json
```

mapping.json의 콘텐츠. 이 예제에서는 크기가 100GiB인 빈 EBS 볼륨(/dev/sdh)을 추가합니다.

```
[
    {
        "DeviceName": "/dev/sdh",
        "Ebs": {
            "VolumeSize": 100
        }
    }
]
```

mapping.json의 콘텐츠. 이 예제에서는 ephemeral1을 인스턴스 스토어 볼륨으로 추가합니다.

```
[
    {
        "DeviceName": "/dev/sdc",
        "VirtualName": "ephemeral1"
    }
]
```

run-instances 출력 예제는 예제 1을 참조하세요.

블록 디바이스 매핑에 대한 자세한 내용은 Amazon EC2 사용 설명서에서 블록 디바이스 매핑을 참조하세요.

예제 4: 인스턴스를 시작하고 생성 시 태그를 추가하는 방법

다음 run-instances 예제에서는 키가 production이고 값이 webserver인 태그를 인스턴스에 추가합니다. 이 명령은 또 생성되는 EBS 볼륨(이 경우에는 루트 볼륨)에 키가 cost-center이고 값이 cc123인 태그를 적용합니다.

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type t2.micro \
    --count 1 \
    --subnet-id subnet-08fc749671b2d077c \
    --key-name MyKeyPair \
    --security-group-ids sg-0b0384b66d7d692f9 \
    --tag-specifications
 'ResourceType=instance,Tags=[{Key=webserver,Value=production}]'
 'ResourceType=volume,Tags=[{Key=cost-center,Value=cc123}]'
```

run-instances 출력 예제는 예제 1을 참조하세요.

예제 5: 사용자 데이터를 포함하는 인스턴스를 시작하는 방법

다음 run-instances 예제에서는 인스턴스의 구성 스크립트가 포함된 my_script.txt 파일에 사용자 데이터를 전달합니다. 스크립트는 시작할 때 실행됩니다.

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type t2.micro \
    --count 1 \
    --subnet-id subnet-08fc749671b2d077c \
    --key-name MyKeyPair \
    --security-group-ids sg-0b0384b66d7d692f9 \
    --user-data file://my_script.txt
```

run-instances 출력 예제는 예제 1을 참조하세요.

인스턴스 사용자 데이터에 대한 자세한 내용은 Amazon EC2 사용 설명서에서 인스턴스 사용자 데이터 작업을 참조하세요.

예제 6: 성능 버스트 기능이 있는 인스턴스를 시작하는 방법

다음 run-instances 예제에서는 unlimited 크레딧 옵션을 사용하여 t2.micro 인스턴스를 시작합니다. T2 인스턴스를 시작할 때 --credit-specification을 지정하지 않으면 기본값은 standard 크레딧 옵션입니다. T3 인스턴스를 시작할 때 기본값은 unlimited 크레딧 옵션입니다.

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type t2.micro \
    --count 1 \
    --subnet-id subnet-08fc749671b2d077c \
    --key-name MyKeyPair \
    --security-group-ids sg-0b0384b66d7d692f9 \
    --credit-specification CpuCredits=unlimited
```

`run-instances` 출력 예제는 예제 1을 참조하세요.

성능 버스트 기능이 있는 인스턴스에 대한 자세한 내용은 Amazon EC2 사용 설명서에서 성능 버스트 기능이 있는 인스턴스를 참조하세요.

- API 세부 정보는 AWS CLI 명령 RunInstances참조를 참조하십시오.

Java

SDK for Java 2.x

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.ec2.Ec2Client;
import software.amazon.awssdk.services.ec2.model.InstanceType;
import software.amazon.awssdk.services.ec2.model.RunInstancesRequest;
import software.amazon.awssdk.services.ec2.model.RunInstancesResponse;
import software.amazon.awssdk.services.ec2.model.Tag;
import software.amazon.awssdk.services.ec2.model.CreateTagsRequest;
import software.amazon.awssdk.services.ec2.model.Ec2Exception;

/**
 * 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
 *
 * This code example requires an AMI value. You can learn more about this value
 * by reading this documentation topic:
 *
 * https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/AMIs.html
 */
public class CreateInstance {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                   <name> <amiId>

                Where:
                   name - An instance name value that you can obtain from the AWS
 Console (for example, ami-xxxxxx5c8b987b1a0).\s
                   amiId - An Amazon Machine Image (AMI) value that you can
 obtain from the AWS Console (for example, i-xxxxxx2734106d0ab).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String name = args[0];
        String amiId = args[1];
        Region region = Region.US_EAST_1;
        Ec2Client ec2 = Ec2Client.builder()
            .region(region)
            .build();

        String instanceId = createEC2Instance(ec2, name, amiId);
        System.out.println("The Amazon EC2 Instance ID is " + instanceId);
        ec2.close();
    }

    public static String createEC2Instance(Ec2Client ec2, String name, String
 amiId) {
        RunInstancesRequest runRequest = RunInstancesRequest.builder()
            .imageId(amiId)
            .instanceType(InstanceType.T1_MICRO)
```

```
            .maxCount(1)
            .minCount(1)
            .build();

        // Use a waiter to wait until the instance is running.
        System.out.println("Going to start an EC2 instance using a waiter");
        RunInstancesResponse response = ec2.runInstances(runRequest);
        String instanceIdVal = response.instances().get(0).instanceId();
        ec2.waiter().waitUntilInstanceRunning(r -> r.instanceIds(instanceIdVal));
        Tag tag = Tag.builder()
            .key("Name")
            .value(name)
            .build();

        CreateTagsRequest tagRequest = CreateTagsRequest.builder()
            .resources(instanceIdVal)
            .tags(tag)
            .build();

        try {
            ec2.createTags(tagRequest);
            System.out.printf("Successfully started EC2 Instance %s based on AMI
 %s", instanceIdVal, amiId);
            return instanceIdVal;

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }

        return "";
    }
}
```

- API 세부 정보는 AWS SDK for Java 2.x API RunInstances참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { RunInstancesCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

// Create a new EC2 instance.
export const main = async () => {
  const command = new RunInstancesCommand({
    // Your key pair name.
    KeyName: "KEY_PAIR_NAME",
    // Your security group.
    SecurityGroupIds: ["SECURITY_GROUP_ID"],
    // An x86_64 compatible image.
    ImageId: "ami-0001a0d1a04bfcc30",
    // An x86_64 compatible free-tier instance type.
    InstanceType: "t1.micro",
    // Ensure only 1 instance launches.
    MinCount: 1,
    MaxCount: 1,
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API RunInstances참조를 참조하십시오.

Kotlin

SDK for Kotlin

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
suspend fun createEC2Instance(
    name: String,
    amiId: String,
): String? {
    val request =
        RunInstancesRequest {
            imageId = amiId
            instanceType = InstanceType.T1Micro
            maxCount = 1
            minCount = 1
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.runInstances(request)
        val instanceId = response.instances?.get(0)?.instanceId
        val tag =
            Tag {
                key = "Name"
                value = name
            }

        val requestTags =
            CreateTagsRequest {
                resources = listOf(instanceId.toString())
                tags = listOf(tag)
            }
        ec2.createTags(requestTags)
        println("Successfully started EC2 Instance $instanceId based on AMI
 $amiId")
        return instanceId
    }
}
```

- API 세부 정보는 Kotlin API용AWS SDK 레퍼런스를 참조하세요 RunInstances.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예제는 EC2-Classic 또는 기본 VPC에서 지정된 AMI의 단일 인스턴스를 시작합니다.

```
New-EC2Instance -ImageId ami-12345678 -MinCount 1 -MaxCount 1 -InstanceType
 m3.medium -KeyName my-key-pair -SecurityGroup my-security-group
```

예 2: 이 예제는 VPC에서 지정된 AMI의 단일 인스턴스를 시작합니다.

```
New-EC2Instance -ImageId ami-12345678 -MinCount 1 -MaxCount 1 -SubnetId
 subnet-12345678 -InstanceType t2.micro -KeyName my-key-pair -SecurityGroupId
 sg-12345678
```

예 3: EBS 볼륨 또는 인스턴스 스토어 볼륨을 추가하려면 블록 디바이스 매핑을 정의하고 명령에 추가합니다. 이 예제에서는 인스턴스 스토어 볼륨을 추가합니다.

```
$bdm = New-Object Amazon.EC2.Model.BlockDeviceMapping
$bdm.VirtualName = "ephemeral0"
$bdm.DeviceName = "/dev/sdf"

New-EC2Instance -ImageId ami-12345678 -BlockDeviceMapping $bdm ...
```

예 4: 현재 Windows AMI 중 하나를 지정하려면 를 사용하여 Get-EC2ImageByName AMI ID를 가져오십시오. 이 예제는 윈도우 서버 2016의 현재 기본 AMI에서 인스턴스를 시작합니다.

```
$ami = Get-EC2ImageByName WINDOWS_2016_BASE

New-EC2Instance -ImageId $ami.ImageId ...
```

예 5: 지정된 전용 호스트 환경에서 인스턴스를 시작합니다.

```
New-EC2Instance -ImageId ami-1a2b3c4d -InstanceType m4.large -KeyName my-key-pair
 -SecurityGroupId sg-1a2b3c4d  -AvailabilityZone us-west-1a -Tenancy host -HostID
 h-1a2b3c4d5e6f1a2b3
```

예 6: 이 요청은 두 개의 인스턴스를 시작하고 웹 서버 키와 프로덕션 값이 포함된 태그를 인스턴스에 적용합니다. 또한 요청은 cost-center 키와 cc123 값을 가진 태그를 생성된 볼륨 (이 경우 각 인스턴스의 루트 볼륨) 에 적용합니다.

```
$tag1 = @{ Key="webserver"; Value="production" }
$tag2 = @{ Key="cost-center"; Value="cc123" }

$tagspec1 = new-object Amazon.EC2.Model.TagSpecification
$tagspec1.ResourceType = "instance"
$tagspec1.Tags.Add($tag1)

$tagspec2 = new-object Amazon.EC2.Model.TagSpecification
$tagspec2.ResourceType = "volume"
$tagspec2.Tags.Add($tag2)

New-EC2Instance -ImageId "ami-1a2b3c4d" -KeyName "my-key-pair" -MaxCount 2 -
InstanceType "t2.large" -SubnetId "subnet-1a2b3c4d" -TagSpecification $tagspec1,
$tagspec2
```

- API 세부 정보는 Cmdlet 참조를 참조하십시오 RunInstances.AWS Tools for PowerShell

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
 actions."""

    def __init__(self, ec2_resource, instance=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
```

```
        :param instance: A Boto3 Instance object. This is a high-level object
that
                         wraps instance actions.
        """
        self.ec2_resource = ec2_resource
        self.instance = instance

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def create(self, image, instance_type, key_pair, security_groups=None):
        """
        Creates a new EC2 instance. The instance starts immediately after
        it is created.

        The instance is created in the default VPC of the current account.

        :param image: A Boto3 Image object that represents an Amazon Machine
Image (AMI)
                      that defines attributes of the instance that is created.
The AMI
                      defines things like the kind of operating system and the
type of
                      storage used by the instance.
        :param instance_type: The type of instance to create, such as 't2.micro'.
                              The instance type defines things like the number of
CPUs and
                              the amount of memory.
        :param key_pair: A Boto3 KeyPair or KeyPairInfo object that represents
the key
                         pair that is used to secure connections to the instance.
        :param security_groups: A list of Boto3 SecurityGroup objects that
represents the
                                security groups that are used to grant access to
the
                                instance. When no security groups are specified,
the
                                default security group of the VPC is used.
        :return: A Boto3 Instance object that represents the newly created
instance.
        """
```

```
        try:
            instance_params = {
                "ImageId": image.id,
                "InstanceType": instance_type,
                "KeyName": key_pair.name,
            }
            if security_groups is not None:
                instance_params["SecurityGroupIds"] = [sg.id for sg in
security_groups]
            self.instance = self.ec2_resource.create_instances(
                **instance_params, MinCount=1, MaxCount=1
            )[0]
            self.instance.wait_until_running()
        except ClientError as err:
            logging.error(
                "Couldn't create instance with image %s, instance type %s, and
key %s. "
                "Here's why: %s: %s",
                image.id,
                instance_type,
                key_pair.name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return self.instance
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 RunInstances.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    " Create tags for resource created during instance launch. "
    DATA lt_tagspecifications TYPE /aws1/
cl_ec2tagspecification=>tt_tagspecificationlist.
    DATA ls_tagspecifications LIKE LINE OF lt_tagspecifications.
    ls_tagspecifications =  NEW /aws1/cl_ec2tagspecification(
      iv_resourcetype = 'instance'
      it_tags = VALUE /aws1/cl_ec2tag=>tt_taglist(
        ( NEW /aws1/cl_ec2tag( iv_key = 'Name' iv_value = iv_tag_value ) )
      )
    ).
    APPEND ls_tagspecifications TO lt_tagspecifications.

    TRY.
        " Create/launch Amazon Elastic Compute Cloud (Amazon EC2) instance. "
        oo_result = lo_ec2->runinstances(                           " oo_result
 is returned for testing purposes. "
          iv_imageid = iv_ami_id
          iv_instancetype = 't2.micro'
          iv_maxcount = 1
          iv_mincount = 1
          it_tagspecifications = lt_tagspecifications
          iv_subnetid = iv_subnet_id
        ).
        MESSAGE 'EC2 instance created.' TYPE 'I'.
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.
```

- API 세부 정보는 SAP용AWS SDK ABAP API 참조를 참조하십시오 RunInstances.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **RunScheduledInstances** CLI와 함께 사용

다음 코드 예제는 RunScheduledInstances의 사용 방법을 보여줍니다.

CLI

AWS CLI

정기 인스턴스를 시작하려면

이 예제는 VPC에서 지정된 정기 인스턴스를 시작합니다.

명령:

```
aws ec2 run-scheduled-instances --scheduled-instance-id
 sci-1234-1234-1234-1234-123456789012 --instance-count 1 --launch-specification
 file://launch-specification.json
```

런치 사양.json:

```
{
  "ImageId": "ami-12345678",
  "KeyName": "my-key-pair",
  "InstanceType": "c4.large",
  "NetworkInterfaces": [
    {
        "DeviceIndex": 0,
        "SubnetId": "subnet-12345678",
        "AssociatePublicIpAddress": true,
        "Groups": ["sg-12345678"]
    }
  ],
  "IamInstanceProfile": {
      "Name": "my-iam-role"
  }
}
```

출력:

```
{
  "InstanceIdSet": [
      "i-1234567890abcdef0"
  ]
}
```

이 예제는 EC2-Classic에서 지정된 정기 인스턴스를 시작합니다.

명령:

```
aws ec2 run-scheduled-instances --scheduled-instance-id
 sci-1234-1234-1234-1234-123456789012 --instance-count 1 --launch-specification
 file://launch-specification.json
```

런치 사양.json:

```
{
  "ImageId": "ami-12345678",
  "KeyName": "my-key-pair",
  "SecurityGroupIds": ["sg-12345678"],
  "InstanceType": "c4.large",
  "Placement": {
    "AvailabilityZone": "us-west-2b"
  }
  "IamInstanceProfile": {
      "Name": "my-iam-role"
  }
}
```

출력:

```
{
  "InstanceIdSet": [
      "i-1234567890abcdef0"
  ]
}
```

- API에 대한 자세한 내용은 명령 참조를 참조하십시오. RunScheduledInstancesAWS CLI

PowerShell

도구: PowerShell

예 1: 이 예제는 지정된 정기 인스턴스를 시작합니다.

```
New-EC2ScheduledInstance -ScheduledInstanceId
 sci-1234-1234-1234-1234-123456789012 -InstanceCount 1 `
-IamInstanceProfile_Name my-iam-role `
-LaunchSpecification_ImageId ami-12345678 `
-LaunchSpecification_InstanceType c4.large `
```

```
-LaunchSpecification_SubnetId subnet-12345678`
-LaunchSpecification_SecurityGroupId sg-12345678
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 RunScheduledInstances.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **StartInstances** CLI와 함께 사용

다음 코드 예제는 StartInstances의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 인스턴스 시작하기

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
/// <summary>
/// Start an EC2 instance.
/// </summary>
/// <param name="ec2InstanceId">The instance Id of the Amazon EC2 instance
/// to start.</param>
/// <returns>Async task.</returns>
public async Task StartInstances(string ec2InstanceId)
{
    var request = new StartInstancesRequest
    {
        InstanceIds = new List<string> { ec2InstanceId },
```

```
        };

        var response = await _amazonEC2.StartInstancesAsync(request);

        if (response.StartingInstances.Count > 0)
        {
            var instances = response.StartingInstances;
            instances.ForEach(i =>
            {
                Console.WriteLine($"Successfully started the EC2 instance with
 instance ID: {i.InstanceId}.");
            });
        }
    }
```

- API 세부 정보는 AWS SDK for .NET API StartInstances참조를 참조하십시오.

Bash

AWS CLI Bash 스크립트 사용

> **Note**
>
> 더 많은 정보가 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
###############################################################################
# function ec2_start_instances
#
# This function starts one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i instance_id - The ID(s) of the instance(s) to start (comma-separated).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
```

```
function ec2_start_instances() {
  local instance_ids
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_start_instances"
    echo "Starts one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances."
    echo "  -i instance_id - The ID(s) of the instance(s) to start (comma-
separated)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) instance_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$instance_ids" ]]; then
    errecho "ERROR: You must provide one or more instance IDs with the -i
 parameter."
    usage
    return 1
  fi

  response=$(aws ec2 start-instances \
    --instance-ids "${instance_ids}") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports start-instances operation failed with $response."
    return 1
```

```
  }

  return 0
}
```

이 예제에 사용된 유틸리티 함수

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
##############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
```

```
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- API 세부 정보는 AWS CLI 명령 StartInstances참조를 참조하십시오.

C++

SDK for C++

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);

    Aws::EC2::Model::StartInstancesRequest start_request;
    start_request.AddInstanceIds(instanceId);
    start_request.SetDryRun(true);

    auto dry_run_outcome = ec2Client.StartInstances(start_request);
    if (dry_run_outcome.IsSuccess()) {
        std::cerr
                << "Failed dry run to start instance. A dry run should trigger an
 error."
                << std::endl;
        return false;
    }
    else if (dry_run_outcome.GetError().GetErrorType() !=
             Aws::EC2::EC2Errors::DRY_RUN_OPERATION) {
        std::cout << "Failed dry run to start instance " << instanceId << ": "
                  << dry_run_outcome.GetError().GetMessage() << std::endl;
        return false;
    }
```

```
    start_request.SetDryRun(false);
    auto start_instancesOutcome = ec2Client.StartInstances(start_request);

    if (!start_instancesOutcome.IsSuccess()) {
        std::cout << "Failed to start instance " << instanceId << ": " <<
                  start_instancesOutcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully started instance " << instanceId <<
                  std::endl;
    }
```

- API 세부 정보는 AWS SDK for C++ API StartInstances참조를 참조하십시오.

CLI

AWS CLI

Amazon EC2 인스턴스를 시작하는 방법

다음 예제에서는 지정된 Amazon EBS 지원 인스턴스를 시작합니다.

명령:

```
aws ec2 start-instances --instance-ids i-1234567890abcdef0
```

출력:

```
{
    "StartingInstances": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "CurrentState": {
                "Code": 0,
                "Name": "pending"
            },
            "PreviousState": {
                "Code": 80,
                "Name": "stopped"
            }
        }
    ]
```

```
}
```

자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서에서 인스턴스 중지 및 시작을 참조하세요.

- API 세부 정보는 AWS CLI 명령 StartInstances참조를 참조하십시오.

Java

SDK for Java 2.x

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
   public static void startInstance(Ec2Client ec2, String instanceId) {
       Ec2Waiter ec2Waiter = Ec2Waiter.builder()
               .overrideConfiguration(b -> b.maxAttempts(100))
               .client(ec2)
               .build();

       StartInstancesRequest request = StartInstancesRequest.builder()
               .instanceIds(instanceId)
               .build();

       System.out.println("Use an Ec2Waiter to wait for the instance to run.
This will take a few minutes.");
       ec2.startInstances(request);
       DescribeInstancesRequest instanceRequest =
DescribeInstancesRequest.builder()
               .instanceIds(instanceId)
               .build();

       WaiterResponse<DescribeInstancesResponse> waiterResponse =
ec2Waiter.waitUntilInstanceRunning(instanceRequest);
       waiterResponse.matched().response().ifPresent(System.out::println);
       System.out.println("Successfully started instance " + instanceId);
   }
```

- API 세부 정보는 AWS SDK for Java 2.x API StartInstances참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { StartInstancesCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new StartInstancesCommand({
    // Use DescribeInstancesCommand to find InstanceIds
    InstanceIds: ["INSTANCE_ID"],
  });

  try {
    const { StartingInstances } = await client.send(command);
    const instanceIdList = StartingInstances.map(
      (instance) => ` • ${instance.InstanceId}`,
    );
    console.log("Starting instances:");
    console.log(instanceIdList.join("\n"));
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API StartInstances참조를 참조하십시오.

Kotlin

SDK for Kotlin

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
suspend fun startInstanceSc(instanceId: String) {
    val request =
        StartInstancesRequest {
            instanceIds = listOf(instanceId)
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.startInstances(request)
        println("Waiting until instance $instanceId starts. This will take a few
 minutes.")
        ec2.waitUntilInstanceRunning {
            // suspend call
            instanceIds = listOf(instanceId)
        }
        println("Successfully started instance $instanceId")
    }
}
```

- API 세부 정보는 Kotlin API용AWS SDK 레퍼런스를 참조하세요 StartInstances.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예제는 지정된 인스턴스를 시작합니다.

```
Start-EC2Instance -InstanceId i-12345678
```

출력:

```
CurrentState                        InstanceId    PreviousState
------------                        ----------    -------------
Amazon.EC2.Model.InstanceState    i-12345678    Amazon.EC2.Model.InstanceState
```

예 2: 이 예제는 지정된 인스턴스를 시작합니다.

```
@("i-12345678", "i-76543210") | Start-EC2Instance
```

예 3: 이 예제는 현재 중지된 인스턴스 세트를 시작합니다. 에서 반환된 인스턴스 Get-EC2Instance 객체는 파이프로 연결됩니다. Start-EC2Instance 이 예제에서 사용하는 구문에는 PowerShell 버전 3 이상이 필요합니다.

```
(Get-EC2Instance -Filter @{ Name="instance-state-name";
 Values="stopped"}).Instances | Start-EC2Instance
```

예 4: PowerShell 버전 2에서는 New-Object를 사용하여 Filter 매개 변수에 대한 필터를 만들어야 합니다.

```
$filter = New-Object Amazon.EC2.Model.Filter
$filter.Name = "instance-state-name"
$filter.Values = "stopped"

(Get-EC2Instance -Filter $filter).Instances | Start-EC2Instance
```

- API에 대한 자세한 내용은 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 StartInstances.

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class InstanceWrapper:
```

```
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
actions."""

    def __init__(self, ec2_resource, instance=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param instance: A Boto3 Instance object. This is a high-level object
that
                         wraps instance actions.
        """
        self.ec2_resource = ec2_resource
        self.instance = instance

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def start(self):
        """
        Starts an instance and waits for it to be in a running state.

        :return: The response to the start request.
        """
        if self.instance is None:
            logger.info("No instance to start.")
            return

        try:
            response = self.instance.start()
            self.instance.wait_until_running()
        except ClientError as err:
            logger.error(
                "Couldn't start instance %s. Here's why: %s: %s",
                self.instance.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
```

```
            return response
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 StartInstances.

Ruby

SDK for Ruby

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
require "aws-sdk-ec2"

# Attempts to start an Amazon Elastic Compute Cloud (Amazon EC2) instance.
#
# Prerequisites:
#
# - The Amazon EC2 instance.
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @param instance_id [String] The ID of the instance.
# @return [Boolean] true if the instance was started; otherwise, false.
# @example
#   exit 1 unless instance_started?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'i-123abc'
#   )
def instance_started?(ec2_client, instance_id)
  response = ec2_client.describe_instance_status(instance_ids: [instance_id])

  if response.instance_statuses.count.positive?
    state = response.instance_statuses[0].instance_state.name
    case state
    when "pending"
```

```
      puts "Error starting instance: the instance is pending. Try again later."
      return false
    when "running"
      puts "The instance is already running."
      return true
    when "terminated"
      puts "Error starting instance: " \
        "the instance is terminated, so you cannot start it."
      return false
    end
  end

  ec2_client.start_instances(instance_ids: [instance_id])
  ec2_client.wait_until(:instance_running, instance_ids: [instance_id])
  puts "Instance started."
  return true
rescue StandardError => e
  puts "Error starting instance: #{e.message}"
  return false
end

# Example usage:
def run_me
  instance_id = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:   ruby ec2-ruby-example-start-instance-i-123abc.rb " \
      "INSTANCE_ID REGION "
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    puts "Example: ruby ec2-ruby-example-start-instance-i-123abc.rb " \
      "i-123abc us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  elsif ARGV.count.zero?
    instance_id = "i-123abc"
    region = "us-west-2"
  # Otherwise, use the values as specified at the command prompt.
  else
    instance_id = ARGV[0]
    region = ARGV[1]
  end
```

```
  ec2_client = Aws::EC2::Client.new(region: region)

  puts "Attempting to start instance '#{instance_id}' " \
    "(this might take a few minutes)..."
  unless instance_started?(ec2_client, instance_id)
    puts "Could not start instance."
  end
end

run_me if $PROGRAM_NAME == __FILE__
```

- API 세부 정보는 AWS SDK for Ruby API StartInstances참조를 참조하십시오.

Rust

SDK for Rust

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
async fn start_instance(client: &Client, id: &str) -> Result<(), Error> {
    // start_instance has no unique errors to handle.
    client.start_instances().instance_ids(id).send().await?;

    println!("Waiting for instance to be running");

    let wait_for_running = client
        .wait_until_instance_running()
        .instance_ids(id)
        .wait(Duration::from_secs(60))
        .await;

    match wait_for_running {
        Ok(_) => println!("Instance is running"),
        Err(err) => match err {
            WaiterError::ExceededMaxWait(exceeded) => {
                println!(
```

```
                    "Exceeded max time waiting for instance to start. Exceeded
 {}s by {}s.",
                    exceeded.max_wait().as_secs(),
                    (exceeded.elapsed() - exceeded.max_wait()).as_secs()
                );
                return Ok(());
            }
            _ => return Err(err.into()),
        },
    }

    println!("Started instance.");

    Ok(())
}
```

- API에 대한 자세한 내용은 Rust용AWS SDK API 레퍼런스를 참조하십시오 StartInstances.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> 자세한 내용은 여기에서 확인할 수 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    DATA lt_instance_ids TYPE /aws1/
cl_ec2instidstringlist_w=>tt_instanceidstringlist.
    APPEND NEW /aws1/cl_ec2instidstringlist_w( iv_value = iv_instance_id ) TO
 lt_instance_ids.

    "Perform dry run"
    TRY.
        " DryRun is set to true. This checks for the required permissions to
 start the instance without actually making the request. "
        lo_ec2->startinstances(
          it_instanceids = lt_instance_ids
          iv_dryrun = abap_true
```

```
        ).
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        " If the error code returned is `DryRunOperation`, then you have the
 required permissions to start this instance. "
        IF lo_exception->av_err_code = 'DryRunOperation'.
          MESSAGE 'Dry run to start instance completed.' TYPE 'I'.
          " DryRun is set to false to start instance. "
          oo_result = lo_ec2->startinstances(           " oo_result is returned
 for testing purposes. "
            it_instanceids = lt_instance_ids
            iv_dryrun = abap_false
          ).
          MESSAGE 'Successfully started the EC2 instance.' TYPE 'I'.
          " If the error code returned is `UnauthorizedOperation`, then you don't
 have the required permissions to start this instance. "
        ELSEIF lo_exception->av_err_code = 'UnauthorizedOperation'.
          MESSAGE 'Dry run to start instance failed. User does not have
 permissions to start the instance.' TYPE 'E'.
        ELSE.
          DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
          MESSAGE lv_error TYPE 'E'.
        ENDIF.
    ENDTRY.
```

- API 세부 정보는 SAP용AWS SDK ABAP API 참조를 참조하십시오 StartInstances.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **StopInstances** CLI와 함께 사용

다음 코드 예제는 StopInstances의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 인스턴스 시작하기

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
/// <summary>
/// Stop an EC2 instance.
/// </summary>
/// <param name="ec2InstanceId">The instance Id of the EC2 instance to
/// stop.</param>
/// <returns>Async task.</returns>
public async Task StopInstances(string ec2InstanceId)
{
    // In addition to the list of instance Ids, the
    // request can also include the following properties:
    //     Force      When true, forces the instances to
    //                stop but you must check the integrity
    //                of the file system. Not recommended on
    //                Windows instances.
    //     Hibernate  When true, hibernates the instance if the
    //                instance was enabled for hibernation when
    //                it was launched.
    var request = new StopInstancesRequest
    {
        InstanceIds = new List<string> { ec2InstanceId },
    };

    var response = await _amazonEC2.StopInstancesAsync(request);

    if (response.StoppingInstances.Count > 0)
    {
        var instances = response.StoppingInstances;
        instances.ForEach(i =>
        {
            Console.WriteLine($"Successfully stopped the EC2 Instance " +
                              $"with InstanceID: {i.InstanceId}.");
        });
```

```
        }
    }
```

- API 세부 정보는 AWS SDK for .NET API StopInstances참조를 참조하십시오.

Bash

AWS CLI Bash 스크립트 사용

> **Note**
>
> 더 많은 정보가 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
###############################################################################
# function ec2_stop_instances
#
# This function stops one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i instance_id - The ID(s) of the instance(s) to stop (comma-separated).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_stop_instances() {
  local instance_ids
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_stop_instances"
    echo "Stops one or more Amazon Elastic Compute Cloud (Amazon EC2) instances."
    echo "  -i instance_id - The ID(s) of the instance(s) to stop (comma-
separated)."
    echo "  -h - Display help."
    echo ""
```

```
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) instance_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$instance_ids" ]]; then
    errecho "ERROR: You must provide one or more instance IDs with the -i
 parameter."
    usage
    return 1
  fi

  response=$(aws ec2 stop-instances \
    --instance-ids "${instance_ids}") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports stop-instances operation failed with $response."
    return 1
  }

  return 0
}
```

이 예제에 사용된 유틸리티 함수

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
```

```
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
##############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- API 세부 정보는 AWS CLI 명령 StopInstances참조를 참조하십시오.

C++

SDK for C++

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::StopInstancesRequest request;
    request.AddInstanceIds(instanceId);
    request.SetDryRun(true);

    auto dry_run_outcome = ec2Client.StopInstances(request);
    if (dry_run_outcome.IsSuccess()) {
        std::cerr
                << "Failed dry run to stop instance. A dry run should trigger an
error."
                << std::endl;
        return false;
    }
    else if (dry_run_outcome.GetError().GetErrorType() !=
             Aws::EC2::EC2Errors::DRY_RUN_OPERATION) {
        std::cout << "Failed dry run to stop instance " << instanceId << ": "
                  << dry_run_outcome.GetError().GetMessage() << std::endl;
        return false;
    }

    request.SetDryRun(false);
    auto outcome = ec2Client.StopInstances(request);
    if (!outcome.IsSuccess()) {
        std::cout << "Failed to stop instance " << instanceId << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully stopped instance " << instanceId <<
                  std::endl;
    }
```

- API 세부 정보는 AWS SDK for C++ API StopInstances참조를 참조하십시오.

CLI

AWS CLI

예제 1: Amazon EC2 인스턴스를 중지하는 방법

다음 `stop-instances` 예제에서는 Amazon EBS 지원 인스턴스를 중지합니다.

```
aws ec2 stop-instances \
    --instance-ids i-1234567890abcdef0
```

출력:

```
{
    "StoppingInstances": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "CurrentState": {
                "Code": 64,
                "Name": "stopping"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서에서 인스턴스 중지 및 시작을 참조하세요.

예제 2: Amazon EC2 인스턴스에서 최대 절전 모드를 적용하는 방법

다음 `stop-instances` 예제에서는 인스턴스에서 최대 절전 모드가 활성화되고 인스턴스가 최대 절전 모드 사전 조건을 충족하는 경우 Amazon EBS 지원 인스턴스를 최대 절전 모드로 전환합니다. 인스턴스가 최대 절전 모드로 전환된 후에 인스턴스가 중지됩니다.

```
aws ec2 stop-instances \
    --instance-ids i-1234567890abcdef0 \
```

```
    --hibernate
```

출력:

```
{
    "StoppingInstances": [
        {
            "CurrentState": {
                "Code": 64,
                "Name": "stopping"
            },
            "InstanceId": "i-1234567890abcdef0",
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서에서 온디맨드 Linux 인스턴스를 최대 절전 모드로 전환을 참조하세요.

- API 세부 정보는 AWS CLI 명령 StopInstances참조를 참조하십시오.

Java

SDK for Java 2.x

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    public static void stopInstance(Ec2Client ec2, String instanceId) {
        Ec2Waiter ec2Waiter = Ec2Waiter.builder()
                .overrideConfiguration(b -> b.maxAttempts(100))
                .client(ec2)
                .build();
        StopInstancesRequest request = StopInstancesRequest.builder()
```

```
                .instanceIds(instanceId)
                .build();

        System.out.println("Use an Ec2Waiter to wait for the instance to stop.
 This will take a few minutes.");
        ec2.stopInstances(request);
        DescribeInstancesRequest instanceRequest =
 DescribeInstancesRequest.builder()
                .instanceIds(instanceId)
                .build();

        WaiterResponse<DescribeInstancesResponse> waiterResponse =
 ec2Waiter.waitUntilInstanceStopped(instanceRequest);
        waiterResponse.matched().response().ifPresent(System.out::println);
        System.out.println("Successfully stopped instance " + instanceId);
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API StopInstances참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { StopInstancesCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new StopInstancesCommand({
    // Use DescribeInstancesCommand to find InstanceIds
    InstanceIds: ["INSTANCE_ID"],
  });

  try {
    const { StoppingInstances } = await client.send(command);
```

```
    const instanceIdList = StoppingInstances.map(
      (instance) => ` • ${instance.InstanceId}`,
    );
    console.log("Stopping instances:");
    console.log(instanceIdList.join("\n"));
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API StopInstances참조를 참조하십시오.

Kotlin

SDK for Kotlin

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
suspend fun stopInstanceSc(instanceId: String) {
    val request =
        StopInstancesRequest {
            instanceIds = listOf(instanceId)
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.stopInstances(request)
        println("Waiting until instance $instanceId stops. This will take a few
 minutes.")
        ec2.waitUntilInstanceStopped {
            // suspend call
            instanceIds = listOf(instanceId)
        }
        println("Successfully stopped instance $instanceId")
    }
}
```

- API 세부 정보는 Kotlin API용AWS SDK 레퍼런스를 참조하세요 StopInstances.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예제는 지정된 인스턴스를 중지합니다.

```
Stop-EC2Instance -InstanceId i-12345678
```

출력:

```
CurrentState                        InstanceId    PreviousState
------------                        ----------    -------------
Amazon.EC2.Model.InstanceState    i-12345678    Amazon.EC2.Model.InstanceState
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 StopInstances.

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
 actions."""

    def __init__(self, ec2_resource, instance=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param instance: A Boto3 Instance object. This is a high-level object
 that
```

```
                        wraps instance actions.
        """
        self.ec2_resource = ec2_resource
        self.instance = instance

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def stop(self):
        """
        Stops an instance and waits for it to be in a stopped state.

        :return: The response to the stop request.
        """
        if self.instance is None:
            logger.info("No instance to stop.")
            return

        try:
            response = self.instance.stop()
            self.instance.wait_until_stopped()
        except ClientError as err:
            logger.error(
                "Couldn't stop instance %s. Here's why: %s: %s",
                self.instance.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 StopInstances.

Ruby

SDK for Ruby

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
require "aws-sdk-ec2"

# Prerequisites:
#
# - The Amazon EC2 instance.
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @param instance_id [String] The ID of the instance.
# @return [Boolean] true if the instance was stopped; otherwise, false.
# @example
#   exit 1 unless instance_stopped?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'i-123abc'
#   )
def instance_stopped?(ec2_client, instance_id)
  response = ec2_client.describe_instance_status(instance_ids: [instance_id])

  if response.instance_statuses.count.positive?
    state = response.instance_statuses[0].instance_state.name
    case state
    when "stopping"
      puts "The instance is already stopping."
      return true
    when "stopped"
      puts "The instance is already stopped."
      return true
    when "terminated"
      puts "Error stopping instance: " \
        "the instance is terminated, so you cannot stop it."
      return false
    end
```

```
  end

  ec2_client.stop_instances(instance_ids: [instance_id])
  ec2_client.wait_until(:instance_stopped, instance_ids: [instance_id])
  puts "Instance stopped."
  return true
rescue StandardError => e
  puts "Error stopping instance: #{e.message}"
  return false
end

# Example usage:
def run_me
  instance_id = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:   ruby ec2-ruby-example-stop-instance-i-123abc.rb " \
      "INSTANCE_ID REGION "
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    puts "Example: ruby ec2-ruby-example-start-instance-i-123abc.rb " \
      "i-123abc us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  elsif ARGV.count.zero?
    instance_id = "i-123abc"
    region = "us-west-2"
  # Otherwise, use the values as specified at the command prompt.
  else
    instance_id = ARGV[0]
    region = ARGV[1]
  end

  ec2_client = Aws::EC2::Client.new(region: region)

  puts "Attempting to stop instance '#{instance_id}' " \
    "(this might take a few minutes)..."
  unless instance_stopped?(ec2_client, instance_id)
    puts "Could not stop instance."
  end
end

run_me if $PROGRAM_NAME == __FILE__
```

- API 세부 정보는 AWS SDK for Ruby API StopInstances참조를 참조하십시오.

Rust

SDK for Rust

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
async fn stop_instance(client: &Client, id: &str) -> Result<(), Error> {
    client.stop_instances().instance_ids(id).send().await?;

    println!("Stopping instance...");

    let wait = client
        .wait_until_instance_stopped()
        .instance_ids(id)
        .wait(Duration::from_secs(60))
        .await;

    match wait {
        Ok(_) => {
            println!("Stopped instance.");
        }
        Err(err) => match err {
            WaiterError::ExceededMaxWait(exceeded) => {
                println!(
                    "Exceeded max time waiting for instance to stop. Exceeded {}s
 by {}s",
                    exceeded.max_wait().as_secs(),
                    (exceeded.elapsed() - exceeded.max_wait()).as_secs()
                )
            }
            _ => return Err(err.into()),
        },
    };
    Ok(())
```

```
}
```

- API에 대한 자세한 내용은 Rust용AWS SDK API 레퍼런스를 참조하십시오 StopInstances.

SAP ABAP

SDK for SAP ABAP

> **Note**
>
> 자세한 내용은 여기에서 확인할 수 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    DATA lt_instance_ids TYPE /aws1/
cl_ec2instidstringlist_w=>tt_instanceidstringlist.
    APPEND NEW /aws1/cl_ec2instidstringlist_w( iv_value = iv_instance_id ) TO
 lt_instance_ids.

    "Perform dry run"
    TRY.
        " DryRun is set to true. This checks for the required permissions to stop
 the instance without actually making the request. "
        lo_ec2->stopinstances(
          it_instanceids = lt_instance_ids
          iv_dryrun = abap_true
        ).
      CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        " If the error code returned is `DryRunOperation`, then you have the
 required permissions to stop this instance. "
        IF lo_exception->av_err_code = 'DryRunOperation'.
          MESSAGE 'Dry run to stop instance completed.' TYPE 'I'.
          " DryRun is set to false to stop instance. "
          oo_result = lo_ec2->stopinstances(           " oo_result is returned
 for testing purposes. "
            it_instanceids = lt_instance_ids
            iv_dryrun = abap_false
          ).
          MESSAGE 'Successfully stopped the EC2 instance.' TYPE 'I'.
```

```
          " If the error code returned is `UnauthorizedOperation`, then you don't
 have the required permissions to stop this instance. "
        ELSEIF lo_exception->av_err_code = 'UnauthorizedOperation'.
          MESSAGE 'Dry run to stop instance failed. User does not have
 permissions to stop the instance.' TYPE 'E'.
        ELSE.
          DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
          MESSAGE lv_error TYPE 'E'.
        ENDIF.
    ENDTRY.
```

- API 세부 정보는 SAP용AWS SDK ABAP API 참조를 참조하십시오 StopInstances.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **TerminateInstances** CLI와 함께 사용

다음 코드 예제는 TerminateInstances의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 인스턴스 시작하기

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    /// <summary>
    /// Terminate an EC2 instance.
```

```
    /// </summary>
    /// <param name="ec2InstanceId">The instance Id of the EC2 instance
    /// to terminate.</param>
    /// <returns>Async task.</returns>
    public async Task<List<InstanceStateChange>> TerminateInstances(string
 ec2InstanceId)
    {
        var request = new TerminateInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId }
        };

        var response = await _amazonEC2.TerminateInstancesAsync(request);
        return response.TerminatingInstances;
    }
```

- API 세부 정보는 AWS SDK for .NET API TerminateInstances참조를 참조하십시오.

Bash

AWS CLI Bash 스크립트 사용

> **Note**
>
> 더 많은 정보가 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
###############################################################################
# function ec2_terminate_instances
#
# This function terminates one or more Amazon Elastic Compute Cloud (Amazon EC2)
# instances using the AWS CLI.
#
# Parameters:
#       -i instance_ids - A space-separated list of instance IDs.
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
```

```
###############################################################################
function ec2_terminate_instances() {
  local instance_ids response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_terminate_instances"
    echo "Terminates one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances."
    echo "  -i instance_ids - A space-separated list of instance IDs."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) instance_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Check if instance ID is provided
  if [[ -z "${instance_ids}" ]]; then
    echo "Error: Missing required instance IDs parameter."
    usage
    return 1
  fi

  # shellcheck disable=SC2086
  response=$(aws ec2 terminate-instances \
    "--instance-ids" $instance_ids \
    --query 'TerminatingInstances[*].[InstanceId,CurrentState.Name]' \
    --output text) || {
```

```
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports terminate-instances operation failed.$response"
    return 1
  }

  return 0
}
```

이 예제에 사용된 유틸리티 함수

```
########################################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
########################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

########################################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
########################################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
```

```
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- API 세부 정보는 AWS CLI 명령 TerminateInstances참조를 참조하십시오.

C++

SDK for C++

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);

    Aws::EC2::Model::TerminateInstancesRequest request;
    request.SetInstanceIds({instanceID});

    Aws::EC2::Model::TerminateInstancesOutcome outcome =
            ec2Client.TerminateInstances(request);
    if (outcome.IsSuccess()) {
        std::cout << "Ec2 instance '" << instanceID <<
                  "' was terminated." << std::endl;
    }
    else {
        std::cerr << "Failed to terminate ec2 instance " << instanceID <<
                  ", " <<
                  outcome.GetError().GetMessage() << std::endl;
        return false;
    }
```

- API 세부 정보는 AWS SDK for C++ API TerminateInstances참조를 참조하십시오.

CLI

AWS CLI

Amazon EC2 인스턴스를 종료하는 방법

이 예제에서는 지정된 인스턴스를 종료합니다.

명령:

```
aws ec2 terminate-instances --instance-ids i-1234567890abcdef0
```

출력:

```
{
    "TerminatingInstances": [
        {
            "InstanceId": "i-1234567890abcdef0",
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

자세한 내용은 AWS Command Line Interface 사용 설명서에서 Amazon EC2 인스턴스 사용을 참조하세요.

- API 세부 정보는 AWS CLI 명령 TerminateInstances참조를 참조하십시오.

Java

SDK for Java 2.x

> **Note**
>
> 자세한 내용은 에서 확인할 수 GitHub 있습니다. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    public static void terminateEC2(Ec2Client ec2, String instanceId) {
        try {
            Ec2Waiter ec2Waiter = Ec2Waiter.builder()
                    .overrideConfiguration(b -> b.maxAttempts(100))
                    .client(ec2)
                    .build();

            TerminateInstancesRequest ti = TerminateInstancesRequest.builder()
                    .instanceIds(instanceId)
                    .build();

            System.out.println("Use an Ec2Waiter to wait for the instance to
terminate. This will take a few minutes.");
            ec2.terminateInstances(ti);
            DescribeInstancesRequest instanceRequest =
DescribeInstancesRequest.builder()
                    .instanceIds(instanceId)
                    .build();

            WaiterResponse<DescribeInstancesResponse> waiterResponse = ec2Waiter
                    .waitUntilInstanceTerminated(instanceRequest);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Successfully started instance " + instanceId);
            System.out.println(instanceId + " is terminated!");

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- API 세부 정보는 AWS SDK for Java 2.x API TerminateInstances참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { TerminateInstancesCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new TerminateInstancesCommand({
    InstanceIds: ["INSTANCE_ID"],
  });

  try {
    const { TerminatingInstances } = await client.send(command);
    const instanceList = TerminatingInstances.map(
      (instance) => ` • ${instance.InstanceId}`,
    );
    console.log("Terminating instances:");
    console.log(instanceList.join("\n"));
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API TerminateInstances참조를 참조하십시오.

Kotlin

SDK for Kotlin

> **Note**
>
> 자세한 내용은 다음과 같습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
suspend fun terminateEC2(instanceID: String) {
    val request =
        TerminateInstancesRequest {
            instanceIds = listOf(instanceID)
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.terminateInstances(request)
        response.terminatingInstances?.forEach { instance ->
            println("The ID of the terminated instance is
 ${instance.instanceId}")
        }
    }
}
```

- API 세부 정보는 Kotlin API용AWS SDK 레퍼런스를 참조하세요 TerminateInstances.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예제는 지정된 인스턴스를 종료합니다 (인스턴스가 실행 중이거나 '중지' 상태일 수 있음). 계속하기 전에 cmdlet에서 확인 메시지를 표시합니다. 프롬프트를 표시하지 않으려면 -Force 스위치를 사용하십시오.

```
Remove-EC2Instance -InstanceId i-12345678
```

출력:

```
CurrentState                    InstanceId    PreviousState
```

```
------------                          ----------    -------------
Amazon.EC2.Model.InstanceState    i-12345678    Amazon.EC2.Model.InstanceState
```

- API에 대한 자세한 내용은 Cmdlet 참조를 참조하십시오. TerminateInstancesAWS Tools for PowerShell

Python

SDK for Python(Boto3)

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
class InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
 actions."""

    def __init__(self, ec2_resource, instance=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param instance: A Boto3 Instance object. This is a high-level object
 that
                          wraps instance actions.
        """
        self.ec2_resource = ec2_resource
        self.instance = instance

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def terminate(self):
        """
```

```
        Terminates an instance and waits for it to be in a terminated state.
        """
        if self.instance is None:
            logger.info("No instance to terminate.")
            return

        instance_id = self.instance.id
        try:
            self.instance.terminate()
            self.instance.wait_until_terminated()
            self.instance = None
        except ClientError as err:
            logging.error(
                "Couldn't terminate instance %s. Here's why: %s: %s",
                instance_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- API에 대한 자세한 내용은 파이썬용AWS SDK (Boto3) API 레퍼런스를 참조하십시오 TerminateInstances.

Ruby

SDK for Ruby

> **Note**
>
> 자세한 내용은 다음과 같습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
require "aws-sdk-ec2"

# Prerequisites:
#
# - The Amazon EC2 instance.
```

```
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @param instance_id [String] The ID of the instance.
# @return [Boolean] true if the instance was terminated; otherwise, false.
# @example
#   exit 1 unless instance_terminated?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'i-123abc'
#   )
def instance_terminated?(ec2_client, instance_id)
  response = ec2_client.describe_instance_status(instance_ids: [instance_id])

  if response.instance_statuses.count.positive? &&
    response.instance_statuses[0].instance_state.name == "terminated"

    puts "The instance is already terminated."
    return true
  end

  ec2_client.terminate_instances(instance_ids: [instance_id])
  ec2_client.wait_until(:instance_terminated, instance_ids: [instance_id])
  puts "Instance terminated."
  return true
rescue StandardError => e
  puts "Error terminating instance: #{e.message}"
  return false
end

# Example usage:
def run_me
  instance_id = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:   ruby ec2-ruby-example-terminate-instance-i-123abc.rb " \
      "INSTANCE_ID REGION "
   # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    puts "Example: ruby ec2-ruby-example-terminate-instance-i-123abc.rb " \
      "i-123abc us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  elsif ARGV.count.zero?
    instance_id = "i-123abc"
```

```
    region = "us-west-2"
  # Otherwise, use the values as specified at the command prompt.
  else
    instance_id = ARGV[0]
    region = ARGV[1]
  end

  ec2_client = Aws::EC2::Client.new(region: region)

  puts "Attempting to terminate instance '#{instance_id}' " \
    "(this might take a few minutes)..."
  unless instance_terminated?(ec2_client, instance_id)
    puts "Could not terminate instance."
  end
end

run_me if $PROGRAM_NAME == __FILE__
```

- API 세부 정보는 AWS SDK for Ruby API TerminateInstances참조를 참조하십시오.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오SDK를 사용하여 Amazon EC2 리소스 생성 AWS. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **UnassignPrivateIpAddresses** CLI와 함께 사용

다음 코드 예제는 UnassignPrivateIpAddresses의 사용 방법을 보여줍니다.

CLI

AWS CLI

네트워크 인터페이스에서 보조 사설 IP 주소 할당을 취소하려면

이 예에서는 지정된 네트워크 인터페이스에서 지정된 사설 IP 주소의 할당을 취소합니다. 이 명령이 성공하면 출력이 반환되지 않습니다.

명령:

```
aws ec2 unassign-private-ip-addresses --network-interface-id eni-e5aa89a3 --
private-ip-addresses 10.0.0.82
```

- API 세부 정보는 AWS CLI 명령 UnassignPrivateIpAddresses참조를 참조하십시오.

PowerShell

도구: PowerShell

예 1: 이 예에서는 지정된 네트워크 인터페이스에서 지정된 사설 IP 주소 할당을 취소합니다.

```
Unregister-EC2PrivateIpAddress -NetworkInterfaceId eni-1a2b3c4d -PrivateIpAddress
 10.0.0.82
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 UnassignPrivateIpAddresses.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

AWS SDK 또는 **UnmonitorInstances** CLI와 함께 사용

다음 코드 예제는 UnmonitorInstances의 사용 방법을 보여줍니다.

작업 예제는 대규모 프로그램에서 발췌한 코드이며 컨텍스트에 맞춰 실행해야 합니다. 다음 코드 예제에서는 컨텍스트 내에서 이 작업을 확인할 수 있습니다.

- 인스턴스 시작하기

C++

SDK for C++

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
    Aws::EC2::EC2Client ec2Client(clientConfiguration);
    Aws::EC2::Model::UnmonitorInstancesRequest unrequest;
```

```
    unrequest.AddInstanceIds(instanceId);
    unrequest.SetDryRun(true);

    auto undryRunOutcome = ec2Client.UnmonitorInstances(unrequest);
    if (undryRunOutcome.IsSuccess()) {
        std::cerr
                << "Failed dry run to disable monitoring on instance. A dry run
should trigger an error."
                <<
                std::endl;
        return false;
    }
    else if (undryRunOutcome.GetError().GetErrorType() !=
             Aws::EC2::EC2Errors::DRY_RUN_OPERATION) {
        std::cout << "Failed dry run to disable monitoring on instance " <<
                  instanceId << ": " << undryRunOutcome.GetError().GetMessage()
<<
                  std::endl;
        return false;
    }

    unrequest.SetDryRun(false);
    auto unmonitorInstancesOutcome = ec2Client.UnmonitorInstances(unrequest);
    if (!unmonitorInstancesOutcome.IsSuccess()) {
        std::cout << "Failed to disable monitoring on instance " << instanceId
                  << ": " << unmonitorInstancesOutcome.GetError().GetMessage() <<
                  std::endl;
    }
    else {
        std::cout << "Successfully disable monitoring on instance " <<
                  instanceId << std::endl;
    }
```

- API 세부 정보는 AWS SDK for C++ API UnmonitorInstances참조를 참조하십시오.

CLI

AWS CLI

인스턴스에 대한 세부 모니터링을 비활성화하는 방법

이 예제 명령은 지정된 인스턴스에 대한 세부 모니터링을 비활성화합니다.

명령:

```
aws ec2 unmonitor-instances --instance-ids i-1234567890abcdef0
```

출력:

```
{
  "InstanceMonitorings": [
      {
          "InstanceId": "i-1234567890abcdef0",
          "Monitoring": {
              "State": "disabling"
          }
      }
  ]
}
```

- API 세부 정보는 AWS CLI 명령 UnmonitorInstances참조를 참조하십시오.

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
import { UnmonitorInstancesCommand } from "@aws-sdk/client-ec2";

import { client } from "../libs/client.js";

export const main = async () => {
  const command = new UnmonitorInstancesCommand({
    InstanceIds: ["i-09a3dfe7ae00e853f"],
  });

  try {
    const { InstanceMonitorings } = await client.send(command);
    const instanceMonitoringsList = InstanceMonitorings.map(
```

```
      (im) =>
        ` • Detailed monitoring state for ${im.InstanceId} is
 ${im.Monitoring.State}.`,
    );
    console.log("Monitoring status:");
    console.log(instanceMonitoringsList.join("\n"));
  } catch (err) {
    console.error(err);
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API UnmonitorInstances참조를 참조하십시오.

PowerShell

다음을 위한 도구 PowerShell

예 1: 이 예제는 지정된 인스턴스에 대한 세부 모니터링을 비활성화합니다.

```
Stop-EC2InstanceMonitoring -InstanceId i-12345678
```

출력:

```
InstanceId    Monitoring
----------    ----------
i-12345678    Amazon.EC2.Model.Monitoring
```

- API 세부 정보는 AWS Tools for PowerShell Cmdlet 참조를 참조하십시오 UnmonitorInstances.

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오. SDK를 사용하여 Amazon EC2 리소스 생성 AWS 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

SDK를 사용하는 Amazon EC2의 시나리오 AWS

다음 코드 예제는 SDK를 사용하여 Amazon EC2에서 일반적인 시나리오를 구현하는 방법을 보여줍니다. AWS 이러한 시나리오에서는 Amazon EC2 내에서 여러 함수를 호출하여 특정 태스크를 수행하는

방법을 보여줍니다. 각 시나리오에는 코드 설정 및 실행 방법에 대한 지침을 찾을 수 있는 링크가 포함되어 있습니다. GitHub

예

- SDK를 사용하여 복원력이 뛰어난 서비스를 구축하고 관리합니다. AWS
- SDK를 사용하여 Amazon EC2 인스턴스로 시작하기 AWS

SDK를 사용하여 복원력이 뛰어난 서비스를 구축하고 관리합니다. AWS

다음 코드 예제는 책, 영화, 노래 추천을 반환하는 로드 밸런싱 웹 서비스를 만드는 방법을 보여줍니다. 이 예제에서는 서비스가 장애에 대응하는 방법과 장애 발생 시 복원력을 높이기 위해 서비스를 재구성하는 방법을 보여줍니다.

- Amazon EC2 Auto Scaling 그룹을 사용하여 시작 템플릿을 기반으로 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스를 생성하고 인스턴스 수를 지정된 범위 내로 유지합니다.
- Elastic Load Balancing으로 HTTP 요청을 처리하고 배포합니다.
- Auto Scaling 그룹의 인스턴스 상태를 모니터링하고 요청을 정상 인스턴스로만 전달합니다.
- 각 EC2 인스턴스에서 Python 웹 서버를 실행하여 HTTP 요청을 처리합니다. 웹 서버는 추천 및 상태 확인으로 응답합니다.
- Amazon DynamoDB 테이블을 사용하여 추천 서비스를 시뮬레이션합니다.
- AWS Systems Manager 매개변수를 업데이트하여 요청 및 상태 확인에 대한 웹 서버 응답을 제어합니다.

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

명령 프롬프트에서 대화형 시나리오를 실행합니다.

```
static async Task Main(string[] args)
{
```

```
        _configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally, load local settings.
            .Build();


        // Set up dependency injection for the AWS services.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonIdentityManagementService>()
                    .AddAWSService<IAmazonDynamoDB>()
                    .AddAWSService<IAmazonElasticLoadBalancingV2>()
                    .AddAWSService<IAmazonSimpleSystemsManagement>()
                    .AddAWSService<IAmazonAutoScaling>()
                    .AddAWSService<IAmazonEC2>()
                    .AddTransient<AutoScalerWrapper>()
                    .AddTransient<ElasticLoadBalancerWrapper>()
                    .AddTransient<SmParameterWrapper>()
                    .AddTransient<Recommendations>()
                    .AddSingleton<IConfiguration>(_configuration)
            )
            .Build();

        ServicesSetup(host);
        ResourcesSetup();

        try
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Welcome to the Resilient Architecture Example
Scenario.");
            Console.WriteLine(new string('-', 80));
            await Deploy(true);

            Console.WriteLine("Now let's begin the scenario.");
            Console.WriteLine(new string('-', 80));
```

```
            await Demo(true);

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Finally, let's clean up our resources.");
            Console.WriteLine(new string('-', 80));

            await DestroyResources(true);

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Resilient Architecture Example Scenario is
complete.");
            Console.WriteLine(new string('-', 80));
        }
        catch (Exception ex)
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine($"There was a problem running the scenario:
{ex.Message}");
            await DestroyResources(true);
            Console.WriteLine(new string('-', 80));
        }
    }

    /// <summary>
    /// Setup any common resources, also used for integration testing.
    /// </summary>
    public static void ResourcesSetup()
    {
        _httpClient = new HttpClient();
    }

    /// <summary>
    /// Populate the services for use within the console application.
    /// </summary>
    /// <param name="host">The services host.</param>
    private static void ServicesSetup(IHost host)
    {
        _elasticLoadBalancerWrapper =
host.Services.GetRequiredService<ElasticLoadBalancerWrapper>();
        _iamClient =
host.Services.GetRequiredService<IAmazonIdentityManagementService>();
        _recommendations = host.Services.GetRequiredService<Recommendations>();
        _autoScalerWrapper =
host.Services.GetRequiredService<AutoScalerWrapper>();
```

```
        _smParameterWrapper =
host.Services.GetRequiredService<SmParameterWrapper>();
    }

    /// <summary>
    /// Deploy necessary resources for the scenario.
    /// </summary>
    /// <param name="interactive">True to run as interactive.</param>
    /// <returns>True if successful.</returns>
    public static async Task<bool> Deploy(bool interactive)
    {
        var protocol = "HTTP";
        var port = 80;
        var sshPort = 22;

        Console.WriteLine(
            "\nFor this demo, we'll use the AWS SDK for .NET to create several
AWS resources\n" +
            "to set up a load-balanced web service endpoint and explore some ways
to make it resilient\n" +
            "against various kinds of failures.\n\n" +
            "Some of the resources create by this demo are:\n");

        Console.WriteLine(
            "\t* A DynamoDB table that the web service depends on to provide
book, movie, and song recommendations.");
        Console.WriteLine(
            "\t* An EC2 launch template that defines EC2 instances that each
contain a Python web server.");
        Console.WriteLine(
            "\t* An EC2 Auto Scaling group that manages EC2 instances across
several Availability Zones.");
        Console.WriteLine(
            "\t* An Elastic Load Balancing (ELB) load balancer that targets the
Auto Scaling group to distribute requests.");
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to start deploying
resources.");
        if (interactive)
            Console.ReadLine();

        // Create and populate the DynamoDB table.
        var databaseTableName = _configuration["databaseName"];
        var recommendationsPath = Path.Join(_configuration["resourcePath"],
```

```
            "recommendations_objects.json");
        Console.WriteLine($"Creating and populating a DynamoDB table named
{databaseTableName}.");
        await _recommendations.CreateDatabaseWithName(databaseTableName);
        await _recommendations.PopulateDatabase(databaseTableName,
recommendationsPath);
        Console.WriteLine(new string('-', 80));

        // Create the EC2 Launch Template.

        Console.WriteLine(
            $"Creating an EC2 launch template that runs
'server_startup_script.sh' when an instance starts.\n"
            + "\nThis script starts a Python web server defined in the
`server.py` script. The web server\n"
            + "listens to HTTP requests on port 80 and responds to requests to
'/' and to '/healthcheck'.\n"
            + "For demo purposes, this server is run as the root user. In
production, the best practice is to\n"
            + "run a web server, such as Apache, with least-privileged
credentials.");
        Console.WriteLine(
            "\nThe template also defines an IAM policy that each instance uses to
assume a role that grants\n"
            + "permissions to access the DynamoDB recommendation table and
Systems Manager parameters\n"
            + "that control the flow of the demo.");

        var startupScriptPath = Path.Join(_configuration["resourcePath"],
            "server_startup_script.sh");
        var instancePolicyPath = Path.Join(_configuration["resourcePath"],
            "instance_policy.json");
        await _autoScalerWrapper.CreateTemplate(startupScriptPath,
instancePolicyPath);
        Console.WriteLine(new string('-', 80));

        Console.WriteLine(
            "Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different\n"
            + "Availability Zone.\n");
        var zones = await _autoScalerWrapper.DescribeAvailabilityZones();
        await _autoScalerWrapper.CreateGroupOfSize(3,
_autoScalerWrapper.GroupName, zones);
        Console.WriteLine(new string('-', 80));
```

```
        Console.WriteLine(
            "At this point, you have EC2 instances created. Once each instance
starts, it listens for\n"
            + "HTTP requests. You can see these instances in the console or
continue with the demo.\n");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to continue.");
        if (interactive)
            Console.ReadLine();

        Console.WriteLine("Creating variables that control the flow of the
demo.");
        await _smParameterWrapper.Reset();

        Console.WriteLine(
            "\nCreating an Elastic Load Balancing target group and load balancer.
The target group\n"
            + "defines how the load balancer connects to instances. The load
balancer provides a\n"
            + "single endpoint where clients connect and dispatches requests to
instances in the group.");

        var defaultVpc = await _autoScalerWrapper.GetDefaultVpc();
        var subnets = await
_autoScalerWrapper.GetAllVpcSubnetsForZones(defaultVpc.VpcId, zones);
        var subnetIds = subnets.Select(s => s.SubnetId).ToList();
        var targetGroup = await
_elasticLoadBalancerWrapper.CreateTargetGroupOnVpc(_elasticLoadBalancerWrapper.TargetGro
protocol, port, defaultVpc.VpcId);

        await
_elasticLoadBalancerWrapper.CreateLoadBalancerAndListener(_elasticLoadBalancerWrapper.Lo
subnetIds, targetGroup);
        await
_autoScalerWrapper.AttachLoadBalancerToGroup(_autoScalerWrapper.GroupName,
targetGroup.TargetGroupArn);
        Console.WriteLine("\nVerifying access to the load balancer endpoint...");
        var endPoint = await
_elasticLoadBalancerWrapper.GetEndpointForLoadBalancerByName(_elasticLoadBalancerWrapper
        var loadBalancerAccess = await
_elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);
```

```
        if (!loadBalancerAccess)
        {
            Console.WriteLine("\nCouldn't connect to the load balancer, verifying
 that the port is open...");

            var ipString = await _httpClient.GetStringAsync("https://
checkip.amazonaws.com");
            ipString = ipString.Trim();

            var defaultSecurityGroup = await
 _autoScalerWrapper.GetDefaultSecurityGroupForVpc(defaultVpc);
            var portIsOpen =
 _autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, port,
 ipString);
            var sshPortIsOpen =
 _autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, sshPort,
 ipString);

            if (!portIsOpen)
            {
                Console.WriteLine(
                    "\nFor this example to work, the default security group for
 your default VPC must\n"
                    + "allows access from this computer. You can either add it
 automatically from this\n"
                    + "example or add it yourself using the AWS Management
 Console.\n");

                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
 inbound traffic from your computer's IP address?"))
                {
                    await
 _autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, port,
 ipString);
                }
            }

            if (!sshPortIsOpen)
            {
                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
 inbound SSH traffic for debugging from your computer's IP address?"))
                {
```

```
                    await
_autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, sshPort,
ipString);
                }
            }
            loadBalancerAccess = await
_elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);
        }

        if (loadBalancerAccess)
        {
            Console.WriteLine("Your load balancer is ready. You can access it by
browsing to:");
            Console.WriteLine($"\thttp://{endPoint}\n");
        }
        else
        {
            Console.WriteLine(
                "\nCouldn't get a successful response from the load balancer
endpoint. Troubleshoot by\n"
                + "manually verifying that your VPC and security group are
configured correctly and that\n"
                + "you can successfully make a GET request to the load balancer
endpoint:\n");
            Console.WriteLine($"\thttp://{endPoint}\n");
        }
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to continue with the
demo.");
        if (interactive)
            Console.ReadLine();
        return true;
    }

    /// <summary>
    /// Demonstrate the steps of the scenario.
    /// </summary>
    /// <param name="interactive">True to run as an interactive scenario.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> Demo(bool interactive)
    {
        var ssmOnlyPolicy = Path.Join(_configuration["resourcePath"],
            "ssm_only_policy.json");
```

```
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Resetting parameters to starting values for demo.");
        await _smParameterWrapper.Reset();

        Console.WriteLine("\nThis part of the demonstration shows how to toggle
different parts of the system\n" +
                          "to create situations where the web service fails, and
shows how using a resilient\n" +
                          "architecture can keep the web service running in spite
of these failures.");
        Console.WriteLine(new string('-', 88));
        Console.WriteLine("At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine($"The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.\n" +
                          $"The table name is contained in a Systems Manager
parameter named '{_smParameterWrapper.TableParameter}'.\n" +
                          $"To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.\n");
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");
        Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as\n" +
                          "healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("Instead of failing when the recommendation service
fails, the web service can return a static response.");
        Console.WriteLine("While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.FailureResponseParameter,
"static");

        Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a static response.");
```

```
        Console.WriteLine("The service still reports as healthy because health
checks are still shallow.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("Let's reinstate the recommendation service.\n");
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
_smParameterWrapper.TableName);
        Console.WriteLine(
            "\nLet's also substitute bad credentials for one of the instances in
the target group so that it can't\n" +
            "access the DynamoDB recommendation table.\n"
        );
        await _autoScalerWrapper.CreateInstanceProfileWithName(
            _autoScalerWrapper.BadCredsPolicyName,
            _autoScalerWrapper.BadCredsRoleName,
            _autoScalerWrapper.BadCredsProfileName,
            ssmOnlyPolicy,
            new List<string> { "AmazonSSMManagedInstanceCore" }
        );
        var instances = await
_autoScalerWrapper.GetInstancesByGroupName(_autoScalerWrapper.GroupName);
        var badInstanceId = instances.First();
        var instanceProfile = await
_autoScalerWrapper.GetInstanceProfile(badInstanceId);
        Console.WriteLine(
            $"Replacing the profile for instance {badInstanceId} with a profile
that contains\n" +
            "bad credentials...\n"
        );
        await _autoScalerWrapper.ReplaceInstanceProfile(
            badInstanceId,
            _autoScalerWrapper.BadCredsProfileName,
            instanceProfile.AssociationId
        );
        Console.WriteLine(
            "Now, sending a GET request to the load balancer endpoint returns
either a recommendation or a static response,\n" +
            "depending on which instance is selected by the load balancer.\n"
        );
        if (interactive)
            await DemoActionChoices();
```

```
        Console.WriteLine("\nLet's implement a deep health check. For this demo,
a deep health check tests whether");
        Console.WriteLine("the web service can access the DynamoDB table that it
depends on for recommendations. Note that");
        Console.WriteLine("the deep health check is only for ELB routing and not
for Auto Scaling instance health.");
        Console.WriteLine("This kind of deep health check is not recommended for
Auto Scaling instance health, because it");
        Console.WriteLine("risks accidental termination of all instances in the
Auto Scaling group when a dependent service fails.");

        Console.WriteLine("\nBy implementing deep health checks, the load
balancer can detect when one of the instances is failing");
        Console.WriteLine("and take that instance out of rotation.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.HealthCheckParameter,
"deep");

        Console.WriteLine($"\nNow, checking target health indicates that the
instance with bad credentials ({badInstanceId})");
        Console.WriteLine("is unhealthy. Note that it might take a minute or two
for the load balancer to detect the unhealthy");
        Console.WriteLine("instance. Sending a GET request to the load balancer
endpoint always returns a recommendation, because");
        Console.WriteLine("the load balancer takes unhealthy instances out of its
rotation.");

        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nBecause the instances in this demo are controlled by
an auto scaler, the simplest way to fix an unhealthy");
        Console.WriteLine("instance is to terminate it and let the auto scaler
start a new instance to replace it.");

        await _autoScalerWrapper.TryTerminateInstanceById(badInstanceId);

        Console.WriteLine($"\nEven while the instance is terminating and the new
instance is starting, sending a GET");
        Console.WriteLine("request to the web service continues to get a
successful recommendation response because");
        Console.WriteLine("starts and reports as healthy, it is included in the
load balancing rotation.");
```

```
        Console.WriteLine("Note that terminating and replacing an instance
typically takes several minutes, during which time you");
        Console.WriteLine("can see the changing health check status until the new
instance is running and healthy.");

        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nIf the recommendation service fails now, deep health
checks mean all instances report as unhealthy.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");

        Console.WriteLine($"\nWhen all instances are unhealthy, the load balancer
continues to route requests even to");
        Console.WriteLine("unhealthy instances, allowing them to fail open and
return a static response rather than fail");
        Console.WriteLine("closed and report failure to the customer.");

        if (interactive)
            await DemoActionChoices();
        await _smParameterWrapper.Reset();

        Console.WriteLine(new string('-', 80));
        return true;
    }

    /// <summary>
    /// Clean up the resources from the scenario.
    /// </summary>
    /// <param name="interactive">True to ask the user for cleanup.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> DestroyResources(bool interactive)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine(
            "To keep things tidy and to avoid unwanted charges on your account,
we can clean up all AWS resources\n" +
            "that were created for this demo."
        );
```

```
        if (!interactive || GetYesNoResponse("Do you want to clean up all demo
 resources? (y/n) "))
        {
            await
 _elasticLoadBalancerWrapper.DeleteLoadBalancerByName(_elasticLoadBalancerWrapper.LoadBal
            await
 _elasticLoadBalancerWrapper.DeleteTargetGroupByName(_elasticLoadBalancerWrapper.TargetGr
            await
 _autoScalerWrapper.TerminateAndDeleteAutoScalingGroupWithName(_autoScalerWrapper.GroupNa
            await
 _autoScalerWrapper.DeleteKeyPairByName(_autoScalerWrapper.KeyPairName);
            await
 _autoScalerWrapper.DeleteTemplateByName(_autoScalerWrapper.LaunchTemplateName);
            await _autoScalerWrapper.DeleteInstanceProfile(
                _autoScalerWrapper.BadCredsProfileName,
                _autoScalerWrapper.BadCredsRoleName
            );
            await
 _recommendations.DestroyDatabaseByName(_recommendations.TableName);
        }
        else
        {
            Console.WriteLine(
                "Ok, we'll leave the resources intact.\n" +
                "Don't forget to delete them when you're done with them or you
 might incur unexpected charges."
            );
        }

        Console.WriteLine(new string('-', 80));
        return true;
    }
```

Auto Scaling과 Amazon EC2 작업을 래핑하는 클래스를 생성합니다.

```
/// <summary>
/// Encapsulates Amazon EC2 Auto Scaling and EC2 management methods.
/// </summary>
public class AutoScalerWrapper
{
    private readonly IAmazonAutoScaling _amazonAutoScaling;
    private readonly IAmazonEC2 _amazonEc2;
```

```
    private readonly IAmazonSimpleSystemsManagement _amazonSsm;
    private readonly IAmazonIdentityManagementService _amazonIam;

    private readonly string _instanceType = "";
    private readonly string _amiParam = "";
    private readonly string _launchTemplateName = "";
    private readonly string _groupName = "";
    private readonly string _instancePolicyName = "";
    private readonly string _instanceRoleName = "";
    private readonly string _instanceProfileName = "";
    private readonly string _badCredsProfileName = "";
    private readonly string _badCredsRoleName = "";
    private readonly string _badCredsPolicyName = "";
    private readonly string _keyPairName = "";

    public string GroupName => _groupName;
    public string KeyPairName => _keyPairName;
    public string LaunchTemplateName => _launchTemplateName;
    public string InstancePolicyName => _instancePolicyName;
    public string BadCredsProfileName => _badCredsProfileName;
    public string BadCredsRoleName => _badCredsRoleName;
    public string BadCredsPolicyName => _badCredsPolicyName;

    /// <summary>
    /// Constructor for the AutoScalerWrapper.
    /// </summary>
    /// <param name="amazonAutoScaling">The injected AutoScaling client.</param>
    /// <param name="amazonEc2">The injected EC2 client.</param>
    /// <param name="amazonIam">The injected IAM client.</param>
    /// <param name="amazonSsm">The injected SSM client.</param>
    public AutoScalerWrapper(
        IAmazonAutoScaling amazonAutoScaling,
        IAmazonEC2 amazonEc2,
        IAmazonSimpleSystemsManagement amazonSsm,
        IAmazonIdentityManagementService amazonIam,
        IConfiguration configuration)
    {
        _amazonAutoScaling = amazonAutoScaling;
        _amazonEc2 = amazonEc2;
        _amazonSsm = amazonSsm;
        _amazonIam = amazonIam;

        var prefix = configuration["resourcePrefix"];
        _instanceType = configuration["instanceType"];
```

```
        _amiParam = configuration["amiParam"];

        _launchTemplateName = prefix + "-template";
        _groupName = prefix + "-group";
        _instancePolicyName = prefix + "-pol";
        _instanceRoleName = prefix + "-role";
        _instanceProfileName = prefix + "-prof";
        _badCredsPolicyName = prefix + "-bc-pol";
        _badCredsRoleName = prefix + "-bc-role";
        _badCredsProfileName = prefix + "-bc-prof";
        _keyPairName = prefix + "-key-pair";
    }

    /// <summary>
    /// Create a policy, role, and profile that is associated with instances with
a specified name.
    /// An instance's associated profile defines a role that is assumed by the
    /// instance.The role has attached policies that specify the AWS permissions
granted to
    /// clients that run on the instance.
    /// </summary>
    /// <param name="policyName">Name to use for the policy.</param>
    /// <param name="roleName">Name to use for the role.</param>
    /// <param name="profileName">Name to use for the profile.</param>
    /// <param name="ssmOnlyPolicyFile">Path to a policy file for SSM.</param>
    /// <param name="awsManagedPolicies">AWS Managed policies to be attached to
the role.</param>
    /// <returns>The Arn of the profile.</returns>
    public async Task<string> CreateInstanceProfileWithName(
        string policyName,
        string roleName,
        string profileName,
        string ssmOnlyPolicyFile,
        List<string>? awsManagedPolicies = null)
    {

        var assumeRoleDoc = "{" +
                                    "\"Version\": \"2012-10-17\"," +
                                    "\"Statement\": [{" +
                                         "\"Effect\": \"Allow\"," +
                                         "\"Principal\": {" +
                                         "\"Service\": [" +
                                             "\"ec2.amazonaws.com\"" +
                                         "]" +
```

```
                                    "}," +
                                "\"Action\": \"sts:AssumeRole\"" +
                                "}]" +
                            "}";

        var policyDocument = await File.ReadAllTextAsync(ssmOnlyPolicyFile);

        var policyArn = "";

        try
        {
            var createPolicyResult = await _amazonIam.CreatePolicyAsync(
                new CreatePolicyRequest
                {
                    PolicyName = policyName,
                    PolicyDocument = policyDocument
                });
            policyArn = createPolicyResult.Policy.Arn;
        }
        catch (EntityAlreadyExistsException)
        {
            // The policy already exists, so we look it up to get the Arn.
            var policiesPaginator = _amazonIam.Paginators.ListPolicies(
                new ListPoliciesRequest()
                {
                    Scope = PolicyScopeType.Local
                });
            // Get the entire list using the paginator.
            await foreach (var policy in policiesPaginator.Policies)
            {
                if (policy.PolicyName.Equals(policyName))
                {
                    policyArn = policy.Arn;
                }
            }

            if (policyArn == null)
            {
                throw new InvalidOperationException("Policy not found");
            }
        }

        try
        {
```

```
            await _amazonIam.CreateRoleAsync(new CreateRoleRequest()
            {
                RoleName = roleName,
                AssumeRolePolicyDocument = assumeRoleDoc,
            });
            await _amazonIam.AttachRolePolicyAsync(new AttachRolePolicyRequest()
            {
                RoleName = roleName,
                PolicyArn = policyArn
            });
            if (awsManagedPolicies != null)
            {
                foreach (var awsPolicy in awsManagedPolicies)
                {
                    await _amazonIam.AttachRolePolicyAsync(new
AttachRolePolicyRequest()
                    {
                        PolicyArn = $"arn:aws:iam::aws:policy/{awsPolicy}",
                        RoleName = roleName
                    });
                }
            }
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Role already exists.");
        }

        string profileArn = "";
        try
        {
            var profileCreateResponse = await
_amazonIam.CreateInstanceProfileAsync(
                new CreateInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            // Allow time for the profile to be ready.
            profileArn = profileCreateResponse.InstanceProfile.Arn;
            Thread.Sleep(10000);
            await _amazonIam.AddRoleToInstanceProfileAsync(
                new AddRoleToInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
```

```
                    RoleName = roleName
                });

        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Policy already exists.");
            var profileGetResponse = await _amazonIam.GetInstanceProfileAsync(
                new GetInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            profileArn = profileGetResponse.InstanceProfile.Arn;
        }
        return profileArn;
    }

    /// <summary>
    /// Create a new key pair and save the file.
    /// </summary>
    /// <param name="newKeyPairName">The name of the new key pair.</param>
    /// <returns>Async task.</returns>
    public async Task CreateKeyPair(string newKeyPairName)
    {
        try
        {
            var keyResponse = await _amazonEc2.CreateKeyPairAsync(
                new CreateKeyPairRequest() { KeyName = newKeyPairName });
            await File.WriteAllTextAsync($"{newKeyPairName}.pem",
                keyResponse.KeyPair.KeyMaterial);
            Console.WriteLine($"Created key pair {newKeyPairName}.");
        }
        catch (AlreadyExistsException)
        {
            Console.WriteLine("Key pair already exists.");
        }
    }

    /// <summary>
    /// Delete the key pair and file by name.
    /// </summary>
    /// <param name="deleteKeyPairName">The key pair to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteKeyPairByName(string deleteKeyPairName)
```

```
    {
        try
        {
            await _amazonEc2.DeleteKeyPairAsync(
                new DeleteKeyPairRequest() { KeyName = deleteKeyPairName });
            File.Delete($"{deleteKeyPairName}.pem");
        }
        catch (FileNotFoundException)
        {
            Console.WriteLine($"Key pair {deleteKeyPairName} not found.");
        }
    }

    /// <summary>
    /// Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling.
    /// The launch template specifies a Bash script in its user data field that
runs after
    /// the instance is started. This script installs the Python packages and
starts a Python
    /// web server on the instance.
    /// </summary>
    /// <param name="startupScriptPath">The path to a Bash script file that is
run.</param>
    /// <param name="instancePolicyPath">The path to a permissions policy to
create and attach to the profile.</param>
    /// <returns>The template object.</returns>
    public async Task<Amazon.EC2.Model.LaunchTemplate> CreateTemplate(string
startupScriptPath, string instancePolicyPath)
    {
        await CreateKeyPair(_keyPairName);
        await CreateInstanceProfileWithName(_instancePolicyName,
_instanceRoleName, _instanceProfileName, instancePolicyPath);

        var startServerText = await File.ReadAllTextAsync(startupScriptPath);
        var plainTextBytes = System.Text.Encoding.UTF8.GetBytes(startServerText);

        var amiLatest = await _amazonSsm.GetParameterAsync(
            new GetParameterRequest() { Name = _amiParam });
        var amiId = amiLatest.Parameter.Value;
        var launchTemplateResponse = await _amazonEc2.CreateLaunchTemplateAsync(
            new CreateLaunchTemplateRequest()
            {
                LaunchTemplateName = _launchTemplateName,
```

```
                LaunchTemplateData = new RequestLaunchTemplateData()
                {
                    InstanceType = _instanceType,
                    ImageId = amiId,
                    IamInstanceProfile =
                        new

 LaunchTemplateIamInstanceProfileSpecificationRequest()
                        {
                            Name = _instanceProfileName
                        },
                    KeyName = _keyPairName,
                    UserData = System.Convert.ToBase64String(plainTextBytes)
                }
            });
        return launchTemplateResponse.LaunchTemplate;

    }

    /// <summary>
    /// Get a list of Availability Zones in the AWS Region of the Amazon EC2
 Client.
    /// </summary>
    /// <returns>A list of availability zones.</returns>
    public async Task<List<string>> DescribeAvailabilityZones()
    {
        var zoneResponse = await _amazonEc2.DescribeAvailabilityZonesAsync(
            new DescribeAvailabilityZonesRequest());
        return zoneResponse.AvailabilityZones.Select(z => z.ZoneName).ToList();
    }

    /// <summary>
    /// Create an EC2 Auto Scaling group of a specified size and name.
    /// </summary>
    /// <param name="groupSize">The size for the group.</param>
    /// <param name="groupName">The name for the group.</param>
    /// <param name="availabilityZones">The availability zones for the group.</
param>
    /// <returns>Async task.</returns>
    public async Task CreateGroupOfSize(int groupSize, string groupName,
 List<string> availabilityZones)
    {
        try
        {
```

```
            await _amazonAutoScaling.CreateAutoScalingGroupAsync(
                new CreateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    AvailabilityZones = availabilityZones,
                    LaunchTemplate =
                        new
Amazon.AutoScaling.Model.LaunchTemplateSpecification()
                        {
                            LaunchTemplateName = _launchTemplateName,
                            Version = "$Default"
                        },
                    MaxSize = groupSize,
                    MinSize = groupSize
                });
            Console.WriteLine($"Created EC2 Auto Scaling group {groupName} with
size {groupSize}.");
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine($"EC2 Auto Scaling group {groupName} already
exists.");
        }
    }

    /// <summary>
    /// Get the default VPC for the account.
    /// </summary>
    /// <returns>The default VPC object.</returns>
    public async Task<Vpc> GetDefaultVpc()
    {
        var vpcResponse = await _amazonEc2.DescribeVpcsAsync(
            new DescribeVpcsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("is-default", new List<string>() { "true" })
                }
            });
        return vpcResponse.Vpcs[0];
    }

    /// <summary>
    /// Get all the subnets for a Vpc in a set of availability zones.
```

```
    /// </summary>
    /// <param name="vpcId">The Id of the Vpc.</param>
    /// <param name="availabilityZones">The list of availability zones.</param>
    /// <returns>The collection of subnet objects.</returns>
    public async Task<List<Subnet>> GetAllVpcSubnetsForZones(string vpcId,
List<string> availabilityZones)
    {
        var subnets = new List<Subnet>();
        var subnetPaginator = _amazonEc2.Paginators.DescribeSubnets(
            new DescribeSubnetsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("vpc-id", new List<string>() { vpcId}),
                    new ("availability-zone", availabilityZones),
                    new ("default-for-az", new List<string>() { "true" })
                }
            });

        // Get the entire list using the paginator.
        await foreach (var subnet in subnetPaginator.Subnets)
        {
            subnets.Add(subnet);
        }

        return subnets;
    }

    /// <summary>
    /// Delete a launch template by name.
    /// </summary>
    /// <param name="templateName">The name of the template to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteTemplateByName(string templateName)
    {
        try
        {
            await _amazonEc2.DeleteLaunchTemplateAsync(
                new DeleteLaunchTemplateRequest()
                {
                    LaunchTemplateName = templateName
                });
        }
        catch (AmazonClientException)
```

```
        {
            Console.WriteLine($"Unable to delete template {templateName}.");
        }
    }

    /// <summary>
    /// Detaches a role from an instance profile, detaches policies from the
role,
    /// and deletes all the resources.
    /// </summary>
    /// <param name="profileName">The name of the profile to delete.</param>
    /// <param name="roleName">The name of the role to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteInstanceProfile(string profileName, string roleName)
    {
        try
        {
            await _amazonIam.RemoveRoleFromInstanceProfileAsync(
                new RemoveRoleFromInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });
            await _amazonIam.DeleteInstanceProfileAsync(
                new DeleteInstanceProfileRequest() { InstanceProfileName =
profileName });
            var attachedPolicies = await
_amazonIam.ListAttachedRolePoliciesAsync(
                new ListAttachedRolePoliciesRequest() { RoleName = roleName });
            foreach (var policy in attachedPolicies.AttachedPolicies)
            {
                await _amazonIam.DetachRolePolicyAsync(
                    new DetachRolePolicyRequest()
                    {
                        RoleName = roleName,
                        PolicyArn = policy.PolicyArn
                    });
                // Delete the custom policies only.
                if (!policy.PolicyArn.StartsWith("arn:aws:iam::aws"))
                {
                    await _amazonIam.DeletePolicyAsync(
                        new Amazon.IdentityManagement.Model.DeletePolicyRequest()
                        {
                            PolicyArn = policy.PolicyArn
```

```
                        });
                }
            }

            await _amazonIam.DeleteRoleAsync(
                new DeleteRoleRequest() { RoleName = roleName });
        }
        catch (NoSuchEntityException)
        {
            Console.WriteLine($"Instance profile {profileName} does not exist.");
        }
    }

    /// <summary>
    /// Gets data about the instances in an EC2 Auto Scaling group by its group
name.
    /// </summary>
    /// <param name="group">The name of the auto scaling group.</param>
    /// <returns>A collection of instance Ids.</returns>
    public async Task<IEnumerable<string>> GetInstancesByGroupName(string group)
    {
        var instanceResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { group }
            });
        var instanceIds = instanceResponse.AutoScalingGroups.SelectMany(
            g => g.Instances.Select(i => i.InstanceId));
        return instanceIds;
    }

    /// <summary>
    /// Get the instance profile association data for an instance.
    /// </summary>
    /// <param name="instanceId">The Id of the instance.</param>
    /// <returns>Instance profile associations data.</returns>
    public async Task<IamInstanceProfileAssociation> GetInstanceProfile(string
instanceId)
    {
        var response = await
_amazonEc2.DescribeIamInstanceProfileAssociationsAsync(
            new DescribeIamInstanceProfileAssociationsRequest()
            {
```

```
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("instance-id", new List<string>() { instanceId })
                },
            });
        return response.IamInstanceProfileAssociations[0];
    }

    /// <summary>
    /// Replace the profile associated with a running instance. After the profile
is replaced, the instance
    /// is rebooted to ensure that it uses the new profile. When the instance is
ready, Systems Manager is
    /// used to restart the Python web server.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to update.</param>
    /// <param name="credsProfileName">The name of the new profile to associate
with the specified instance.</param>
    /// <param name="associationId">The Id of the existing profile association
for the instance.</param>
    /// <returns>Async task.</returns>
    public async Task ReplaceInstanceProfile(string instanceId, string
credsProfileName, string associationId)
    {
        await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
                new ReplaceIamInstanceProfileAssociationRequest()
                {
                    AssociationId = associationId,
                    IamInstanceProfile = new IamInstanceProfileSpecification()
                    {
                        Name = credsProfileName
                    }
                });
        // Allow time before resetting.
        Thread.Sleep(25000);
        var instanceReady = false;
        var retries = 5;
        while (retries-- > 0 && !instanceReady)
        {
            await _amazonEc2.RebootInstancesAsync(
                new RebootInstancesRequest(new List<string>() { instanceId }));
            Thread.Sleep(10000);
```

```
            var instancesPaginator =
_amazonSsm.Paginators.DescribeInstanceInformation(
                new DescribeInstanceInformationRequest());
            // Get the entire list using the paginator.
            await foreach (var instance in
instancesPaginator.InstanceInformationList)
            {
                instanceReady = instance.InstanceId == instanceId;
                if (instanceReady)
                {
                    break;
                }
            }
        }
        Console.WriteLine($"Sending restart command to instance {instanceId}");
        await _amazonSsm.SendCommandAsync(
            new SendCommandRequest()
            {
                InstanceIds = new List<string>() { instanceId },
                DocumentName = "AWS-RunShellScript",
                Parameters = new Dictionary<string, List<string>>()
                {
                        {"commands", new List<string>() { "cd / && sudo python3
server.py 80" }}
                }
            });
        Console.WriteLine($"Restarted the web server on instance {instanceId}");
    }

    /// <summary>
    /// Try to terminate an instance by its Id.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to terminate.</param>
    /// <returns>Async task.</returns>
    public async Task TryTerminateInstanceById(string instanceId)
    {
        var stopping = false;
        Console.WriteLine($"Stopping {instanceId}...");
        while (!stopping)
        {
            try
            {
                await
_amazonAutoScaling.TerminateInstanceInAutoScalingGroupAsync(
```

```
                    new TerminateInstanceInAutoScalingGroupRequest()
                    {
                        InstanceId = instanceId,
                        ShouldDecrementDesiredCapacity = false
                    });
                stopping = true;
            }
            catch (ScalingActivityInProgressException)
            {
                Console.WriteLine($"Scaling activity in progress for
{instanceId}. Waiting...");
                Thread.Sleep(10000);
            }
        }
    }

    /// <summary>
    /// Tries to delete the EC2 Auto Scaling group. If the group is in use or in
progress,
    /// waits and retries until the group is successfully deleted.
    /// </summary>
    /// <param name="groupName">The name of the group to try to delete.</param>
    /// <returns>Async task.</returns>
    public async Task TryDeleteGroupByName(string groupName)
    {
        var stopped = false;
        while (!stopped)
        {
            try
            {
                await _amazonAutoScaling.DeleteAutoScalingGroupAsync(
                    new DeleteAutoScalingGroupRequest()
                    {
                        AutoScalingGroupName = groupName
                    });
                stopped = true;
            }
            catch (Exception e)
                when ((e is ScalingActivityInProgressException)
                      || (e is Amazon.AutoScaling.Model.ResourceInUseException))
            {
                Console.WriteLine($"Some instances are still running.
Waiting...");
                Thread.Sleep(10000);
```

```
            }
        }
    }

    /// <summary>
    /// Terminate instances and delete the Auto Scaling group by name.
    /// </summary>
    /// <param name="groupName">The name of the group to delete.</param>
    /// <returns>Async task.</returns>
    public async Task TerminateAndDeleteAutoScalingGroupWithName(string
groupName)
    {
        var describeGroupsResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { groupName }
            });
        if (describeGroupsResponse.AutoScalingGroups.Any())
        {
            // Update the size to 0.
            await _amazonAutoScaling.UpdateAutoScalingGroupAsync(
                new UpdateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    MinSize = 0
                });
            var group = describeGroupsResponse.AutoScalingGroups[0];
            foreach (var instance in group.Instances)
            {
                await TryTerminateInstanceById(instance.InstanceId);
            }

            await TryDeleteGroupByName(groupName);
        }
        else
        {
            Console.WriteLine($"No groups found with name {groupName}.");
        }
    }


    /// <summary>
    /// Get the default security group for a specified Vpc.
```

```
    /// </summary>
    /// <param name="vpc">The Vpc to search.</param>
    /// <returns>The default security group.</returns>
    public async Task<SecurityGroup> GetDefaultSecurityGroupForVpc(Vpc vpc)
    {
        var groupResponse = await _amazonEc2.DescribeSecurityGroupsAsync(
            new DescribeSecurityGroupsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("group-name", new List<string>() { "default" }),
                    new ("vpc-id", new List<string>() { vpc.VpcId })
                }
            });
        return groupResponse.SecurityGroups[0];
    }

    /// <summary>
    /// Verify the default security group of a Vpc allows ingress from the
calling computer.
    /// This can be done by allowing ingress from this computer's IP address.
    /// In some situations, such as connecting from a corporate network, you must
instead specify
    /// a prefix list Id. You can also temporarily open the port to any IP
address while running this example.
    /// If you do, be sure to remove public access when you're done.
    /// </summary>
    /// <param name="vpc">The group to check.</param>
    /// <param name="port">The port to verify.</param>
    /// <param name="ipAddress">This computer's IP address.</param>
    /// <returns>True if the ip address is allowed on the group.</returns>
    public bool VerifyInboundPortForGroup(SecurityGroup group, int port, string
ipAddress)
    {
        var portIsOpen = false;
        foreach (var ipPermission in group.IpPermissions)
        {
            if (ipPermission.FromPort == port)
            {
                foreach (var ipRange in ipPermission.Ipv4Ranges)
                {
                    var cidr = ipRange.CidrIp;
                    if (cidr.StartsWith(ipAddress) || cidr == "0.0.0.0/0")
                    {
```

```
                        portIsOpen = true;
                    }
                }

                if (ipPermission.PrefixListIds.Any())
                {
                    portIsOpen = true;
                }

                if (!portIsOpen)
                {
                    Console.WriteLine("The inbound rule does not appear to be
open to either this computer's IP\n" +
                                      "address, to all IP addresses (0.0.0.0/0),
or to a prefix list ID.");
                }
                else
                {
                    break;
                }
            }
        }

        return portIsOpen;
    }

    /// <summary>
    /// Add an ingress rule to the specified security group that allows access on
the
    /// specified port from the specified IP address.
    /// </summary>
    /// <param name="groupId">The Id of the security group to modify.</param>
    /// <param name="port">The port to open.</param>
    /// <param name="ipAddress">The IP address to allow access.</param>
    /// <returns>Async task.</returns>
    public async Task OpenInboundPort(string groupId, int port, string ipAddress)
    {
        await _amazonEc2.AuthorizeSecurityGroupIngressAsync(
            new AuthorizeSecurityGroupIngressRequest()
            {
                GroupId = groupId,
                IpPermissions = new List<IpPermission>()
                {
                    new IpPermission()
```

```
                    {
                        FromPort = port,
                        ToPort = port,
                        IpProtocol = "tcp",
                        Ipv4Ranges = new List<IpRange>()
                        {
                            new IpRange() { CidrIp = $"{ipAddress}/32" }
                        }
                    }
                }
            });
    }

    /// <summary>
    /// Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
 Scaling group.
    /// The
    /// </summary>
    /// <param name="autoScalingGroupName">The name of the Auto Scaling group.</
param>
    /// <param name="targetGroupArn">The Arn for the target group.</param>
    /// <returns>Async task.</returns>
    public async Task AttachLoadBalancerToGroup(string autoScalingGroupName,
 string targetGroupArn)
    {
        await _amazonAutoScaling.AttachLoadBalancerTargetGroupsAsync(
            new AttachLoadBalancerTargetGroupsRequest()
            {
                AutoScalingGroupName = autoScalingGroupName,
                TargetGroupARNs = new List<string>() { targetGroupArn }
            });
    }
}
```

Elastic Load Balancing 작업을 래핑하는 클래스를 생성합니다.

```
/// <summary>
/// Encapsulates Elastic Load Balancer actions.
/// </summary>
public class ElasticLoadBalancerWrapper
{
```

```
    private readonly IAmazonElasticLoadBalancingV2 _amazonElasticLoadBalancingV2;
    private string? _endpoint = null;
    private readonly string _targetGroupName = "";
    private readonly string _loadBalancerName = "";
    HttpClient _httpClient = new();

    public string TargetGroupName => _targetGroupName;
    public string LoadBalancerName => _loadBalancerName;

    /// <summary>
    /// Constructor for the Elastic Load Balancer wrapper.
    /// </summary>
    /// <param name="amazonElasticLoadBalancingV2">The injected load balancing v2
client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public ElasticLoadBalancerWrapper(
        IAmazonElasticLoadBalancingV2 amazonElasticLoadBalancingV2,
        IConfiguration configuration)
    {
        _amazonElasticLoadBalancingV2 = amazonElasticLoadBalancingV2;
        var prefix = configuration["resourcePrefix"];
        _targetGroupName = prefix + "-tg";
        _loadBalancerName = prefix + "-lb";
    }

    /// <summary>
    /// Get the HTTP Endpoint of a load balancer by its name.
    /// </summary>
    /// <param name="loadBalancerName">The name of the load balancer.</param>
    /// <returns>The HTTP endpoint.</returns>
    public async Task<string> GetEndpointForLoadBalancerByName(string
loadBalancerName)
    {
        if (_endpoint == null)
        {
            var endpointResponse =
                await _amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                    new DescribeLoadBalancersRequest()
                    {
                        Names = new List<string>() { loadBalancerName }
                    });
            _endpoint = endpointResponse.LoadBalancers[0].DNSName;
        }
```

```
        return _endpoint;
    }

    /// <summary>
    /// Return the GET response for an endpoint as text.
    /// </summary>
    /// <param name="endpoint">The endpoint for the request.</param>
    /// <returns>The request response.</returns>
    public async Task<string> GetEndPointResponse(string endpoint)
    {
        var endpointResponse = await _httpClient.GetAsync($"http://{endpoint}");
        var textResponse = await endpointResponse.Content.ReadAsStringAsync();
        return textResponse!;
    }

    /// <summary>
    /// Get the target health for a group by name.
    /// </summary>
    /// <param name="groupName">The name of the group.</param>
    /// <returns>The collection of health descriptions.</returns>
    public async Task<List<TargetHealthDescription>>
CheckTargetHealthForGroup(string groupName)
    {
        List<TargetHealthDescription> result = null!;
        try
        {
            var groupResponse =
                await _amazonElasticLoadBalancingV2.DescribeTargetGroupsAsync(
                    new DescribeTargetGroupsRequest()
                    {
                        Names = new List<string>() { groupName }
                    });
            var healthResponse =
                await _amazonElasticLoadBalancingV2.DescribeTargetHealthAsync(
                    new DescribeTargetHealthRequest()
                    {
                        TargetGroupArn =
groupResponse.TargetGroups[0].TargetGroupArn
                    });
            ;
            result = healthResponse.TargetHealthDescriptions;
        }
        catch (TargetGroupNotFoundException)
        {
```

```
            Console.WriteLine($"Target group {groupName} not found.");
        }
        return result;
    }

    /// <summary>
    /// Create an Elastic Load Balancing target group. The target group specifies
 how the load balancer forwards
    /// requests to instances in the group and how instance health is checked.
    ///
    /// To speed up this demo, the health check is configured with shortened
 times and lower thresholds. In production,
    /// you might want to decrease the sensitivity of your health checks to avoid
 unwanted failures.
    /// </summary>
    /// <param name="groupName">The name for the group.</param>
    /// <param name="protocol">The protocol, such as HTTP.</param>
    /// <param name="port">The port to use to forward requests, such as 80.</
param>
    /// <param name="vpcId">The Id of the Vpc in which the load balancer
 exists.</param>
    /// <returns>The new TargetGroup object.</returns>
    public async Task<TargetGroup> CreateTargetGroupOnVpc(string groupName,
 ProtocolEnum protocol, int port, string vpcId)
    {
        var createResponse = await
 _amazonElasticLoadBalancingV2.CreateTargetGroupAsync(
            new CreateTargetGroupRequest()
            {
                Name = groupName,
                Protocol = protocol,
                Port = port,
                HealthCheckPath = "/healthcheck",
                HealthCheckIntervalSeconds = 10,
                HealthCheckTimeoutSeconds = 5,
                HealthyThresholdCount = 2,
                UnhealthyThresholdCount = 2,
                VpcId = vpcId
            });
        var targetGroup = createResponse.TargetGroups[0];
        return targetGroup;
    }

    /// <summary>
```

```
   /// Create an Elastic Load Balancing load balancer that uses the specified
subnets
   /// and forwards requests to the specified target group.
   /// </summary>
   /// <param name="name">The name for the new load balancer.</param>
   /// <param name="subnetIds">Subnets for the load balancer.</param>
   /// <param name="targetGroup">Target group for forwarded requests.</param>
   /// <returns>The new LoadBalancer object.</returns>
   public async Task<LoadBalancer> CreateLoadBalancerAndListener(string name,
List<string> subnetIds, TargetGroup targetGroup)
   {
       var createLbResponse = await
_amazonElasticLoadBalancingV2.CreateLoadBalancerAsync(
           new CreateLoadBalancerRequest()
           {
               Name = name,
               Subnets = subnetIds
           });
       var loadBalancerArn = createLbResponse.LoadBalancers[0].LoadBalancerArn;

       // Wait for load balancer to be available.
       var loadBalancerReady = false;
       while (!loadBalancerReady)
       {
           try
           {
               var describeResponse =
                   await
_amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                       new DescribeLoadBalancersRequest()
                       {
                           Names = new List<string>() { name }
                       });

               var loadBalancerState =
describeResponse.LoadBalancers[0].State.Code;

               loadBalancerReady = loadBalancerState ==
LoadBalancerStateEnum.Active;
           }
           catch (LoadBalancerNotFoundException)
           {
               loadBalancerReady = false;
           }
```

```
            Thread.Sleep(10000);
        }
        // Create the listener.
        await _amazonElasticLoadBalancingV2.CreateListenerAsync(
            new CreateListenerRequest()
            {
                LoadBalancerArn = loadBalancerArn,
                Protocol = targetGroup.Protocol,
                Port = targetGroup.Port,
                DefaultActions = new List<Action>()
                {
                    new Action()
                    {
                        Type = ActionTypeEnum.Forward,
                        TargetGroupArn = targetGroup.TargetGroupArn
                    }
                }
            });
        return createLbResponse.LoadBalancers[0];
    }

    /// <summary>
    /// Verify this computer can successfully send a GET request to the
    /// load balancer endpoint.
    /// </summary>
    /// <param name="endpoint">The endpoint to check.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> VerifyLoadBalancerEndpoint(string endpoint)
    {
        var success = false;
        var retries = 3;
        while (!success && retries > 0)
        {
            try
            {
                var endpointResponse = await _httpClient.GetAsync($"http://
{endpoint}");
                Console.WriteLine($"Response: {endpointResponse.StatusCode}.");

                if (endpointResponse.IsSuccessStatusCode)
                {
                    success = true;
                }
                else
```

```
                {
                    retries = 0;
                }
            }
            catch (HttpRequestException)
            {
                Console.WriteLine("Connection error, retrying...");
                retries--;
                Thread.Sleep(10000);
            }
        }

        return success;
    }

    /// <summary>
    /// Delete a load balancer by its specified name.
    /// </summary>
    /// <param name="name">The name of the load balancer to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteLoadBalancerByName(string name)
    {
        try
        {
            var describeLoadBalancerResponse =
                await _amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                    new DescribeLoadBalancersRequest()
                    {
                        Names = new List<string>() { name }
                    });
            var lbArn =
describeLoadBalancerResponse.LoadBalancers[0].LoadBalancerArn;
            await _amazonElasticLoadBalancingV2.DeleteLoadBalancerAsync(
                new DeleteLoadBalancerRequest()
                {
                    LoadBalancerArn = lbArn
                }
            );
        }
        catch (LoadBalancerNotFoundException)
        {
            Console.WriteLine($"Load balancer {name} not found.");
        }
    }
```

```
    /// <summary>
    /// Delete a TargetGroup by its specified name.
    /// </summary>
    /// <param name="groupName">Name of the group to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteTargetGroupByName(string groupName)
    {
        var done = false;
        while (!done)
        {
            try
            {
                var groupResponse =
                    await
_amazonElasticLoadBalancingV2.DescribeTargetGroupsAsync(
                        new DescribeTargetGroupsRequest()
                        {
                            Names = new List<string>() { groupName }
                        });

                var targetArn = groupResponse.TargetGroups[0].TargetGroupArn;
                await _amazonElasticLoadBalancingV2.DeleteTargetGroupAsync(
                    new DeleteTargetGroupRequest() { TargetGroupArn =
targetArn });
                Console.WriteLine($"Deleted load balancing target group
{groupName}.");
                done = true;
            }
            catch (TargetGroupNotFoundException)
            {
                Console.WriteLine(
                    $"Target group {groupName} not found, could not delete.");
                done = true;
            }
            catch (ResourceInUseException)
            {
                Console.WriteLine("Target group not yet released, waiting...");
                Thread.Sleep(10000);
            }
        }
    }
}
```

DynamoDB를 사용하여 추천 서비스를 시뮬레이션하는 클래스를 생성합니다.

```
/// <summary>
/// Encapsulates a DynamoDB table to use as a service that recommends books,
 movies, and songs.
/// </summary>
public class Recommendations
{
    private readonly IAmazonDynamoDB _amazonDynamoDb;
    private readonly DynamoDBContext _context;
    private readonly string _tableName;

    public string TableName => _tableName;

    /// <summary>
    /// Constructor for the Recommendations service.
    /// </summary>
    /// <param name="amazonDynamoDb">The injected DynamoDb client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public Recommendations(IAmazonDynamoDB amazonDynamoDb, IConfiguration
 configuration)
    {
        _amazonDynamoDb = amazonDynamoDb;
        _context = new DynamoDBContext(_amazonDynamoDb);
        _tableName = configuration["databaseName"]!;
    }

    /// <summary>
    /// Create the DynamoDb table with a specified name.
    /// </summary>
    /// <param name="tableName">The name for the table.</param>
    /// <returns>True when ready.</returns>
    public async Task<bool> CreateDatabaseWithName(string tableName)
    {
        try
        {
            Console.Write($"Creating table {tableName}...");
            var createRequest = new CreateTableRequest()
            {
                TableName = tableName,
                AttributeDefinitions = new List<AttributeDefinition>()
```

```
                {
                    new AttributeDefinition()
                    {
                        AttributeName = "MediaType",
                        AttributeType = ScalarAttributeType.S
                    },
                    new AttributeDefinition()
                    {
                        AttributeName = "ItemId",
                        AttributeType = ScalarAttributeType.N
                    }
                },
            KeySchema = new List<KeySchemaElement>()
                {
                    new KeySchemaElement()
                    {
                        AttributeName = "MediaType",
                        KeyType = KeyType.HASH
                    },
                    new KeySchemaElement()
                    {
                        AttributeName = "ItemId",
                        KeyType = KeyType.RANGE
                    }
                },
            ProvisionedThroughput = new ProvisionedThroughput()
            {
                ReadCapacityUnits = 5,
                WriteCapacityUnits = 5
            }
        };
        await _amazonDynamoDb.CreateTableAsync(createRequest);

        // Wait until the table is ACTIVE and then report success.
        Console.Write("\nWaiting for table to become active...");

        var request = new DescribeTableRequest
        {
            TableName = tableName
        };

        TableStatus status;
        do
        {
```

```
                Thread.Sleep(2000);

                var describeTableResponse = await
 _amazonDynamoDb.DescribeTableAsync(request);
                status = describeTableResponse.Table.TableStatus;

                Console.Write(".");
            }
            while (status != "ACTIVE");

            return status == TableStatus.ACTIVE;
        }
        catch (ResourceInUseException)
        {
            Console.WriteLine($"Table {tableName} already exists.");
            return false;
        }
    }

    /// <summary>
    /// Populate the database table with data from a specified path.
    /// </summary>
    /// <param name="databaseTableName">The name of the table.</param>
    /// <param name="recommendationsPath">The path of the recommendations data.</
param>
    /// <returns>Async task.</returns>
    public async Task PopulateDatabase(string databaseTableName, string
 recommendationsPath)
    {
        var recommendationsText = await
 File.ReadAllTextAsync(recommendationsPath);
        var records =

 JsonSerializer.Deserialize<RecommendationModel[]>(recommendationsText);
        var batchWrite = _context.CreateBatchWrite<RecommendationModel>();

        foreach (var record in records!)
        {
            batchWrite.AddPutItem(record);
        }

        await batchWrite.ExecuteAsync();
    }
```

```
    /// <summary>
    /// Delete the recommendation table by name.
    /// </summary>
    /// <param name="tableName">The name of the recommendation table.</param>
    /// <returns>Async task.</returns>
    public async Task DestroyDatabaseByName(string tableName)
    {
        try
        {
            await _amazonDynamoDb.DeleteTableAsync(
                new DeleteTableRequest() { TableName = tableName });
            Console.WriteLine($"Table {tableName} was deleted.");
        }
        catch (ResourceNotFoundException)
        {
            Console.WriteLine($"Table {tableName} not found");
        }
    }
}
```

Systems Manager 작업을 래핑하는 클래스를 생성합니다.

```
/// <summary>
/// Encapsulates Systems Manager parameter operations. This example uses these
 parameters
/// to drive the demonstration of resilient architecture, such as failure of a
 dependency or
/// how the service responds to a health check.
/// </summary>
public class SmParameterWrapper
{
    private readonly IAmazonSimpleSystemsManagement
 _amazonSimpleSystemsManagement;

    private readonly string _tableParameter = "doc-example-resilient-
architecture-table";
    private readonly string _failureResponseParameter = "doc-example-resilient-
architecture-failure-response";
    private readonly string _healthCheckParameter = "doc-example-resilient-
architecture-health-check";
    private readonly string _tableName = "";
```

```
    public string TableParameter => _tableParameter;
    public string TableName => _tableName;
    public string HealthCheckParameter => _healthCheckParameter;
    public string FailureResponseParameter => _failureResponseParameter;

    /// <summary>
    /// Constructor for the SmParameterWrapper.
    /// </summary>
    /// <param name="amazonSimpleSystemsManagement">The injected Simple Systems
 Management client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public SmParameterWrapper(IAmazonSimpleSystemsManagement
 amazonSimpleSystemsManagement, IConfiguration configuration)
    {
        _amazonSimpleSystemsManagement = amazonSimpleSystemsManagement;
        _tableName = configuration["databaseName"]!;
    }

    /// <summary>
    /// Reset the Systems Manager parameters to starting values for the demo.
    /// </summary>
    /// <returns>Async task.</returns>
    public async Task Reset()
    {
        await this.PutParameterByName(_tableParameter, _tableName);
        await this.PutParameterByName(_failureResponseParameter, "none");
        await this.PutParameterByName(_healthCheckParameter, "shallow");
    }

    /// <summary>
    /// Set the value of a named Systems Manager parameter.
    /// </summary>
    /// <param name="name">The name of the parameter.</param>
    /// <param name="value">The value to set.</param>
    /// <returns>Async task.</returns>
    public async Task PutParameterByName(string name, string value)
    {
        await _amazonSimpleSystemsManagement.PutParameterAsync(
            new PutParameterRequest() { Name = name, Value = value, Overwrite =
 true });
    }
}
```

- API 세부 정보는 AWS SDK for .NET API 참조의 다음 항목을 참조하세요.
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate
  - DeleteLoadBalancer
  - DeleteTargetGroup
  - DescribeAutoScalingGroups
  - DescribeAvailabilityZones
  - DescribeIamInstanceProfileAssociations
  - DescribeInstances
  - DescribeLoadBalancers
  - DescribeSubnets
  - DescribeTargetGroups
  - DescribeTargetHealth
  - DescribeVpcs
  - RebootInstances
  - ReplaceIamInstanceProfileAssociation
  - TerminateInstanceInAutoScalingGroup
  - UpdateAutoScalingGroup

Java

SDK for Java 2.x

> **Note**
>
> 더 많은 것이 있어요 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

명령 프롬프트에서 대화형 시나리오를 실행합니다.

```
public class Main {

    public static final String fileName = "C:\\AWS\\resworkflow\
\recommendations.json"; // Modify file location.
    public static final String tableName = "doc-example-recommendation-service";
    public static final String startScript = "C:\\AWS\\resworkflow\
\server_startup_script.sh"; // Modify file location.
    public static final String policyFile = "C:\\AWS\\resworkflow\
\instance_policy.json"; // Modify file location.
    public static final String ssmJSON = "C:\\AWS\\resworkflow\
\ssm_only_policy.json"; // Modify file location.
    public static final String failureResponse = "doc-example-resilient-
architecture-failure-response";
    public static final String healthCheck = "doc-example-resilient-architecture-
health-check";
    public static final String templateName = "doc-example-resilience-template";
    public static final String roleName = "doc-example-resilience-role";
    public static final String policyName = "doc-example-resilience-pol";
    public static final String profileName = "doc-example-resilience-prof";

    public static final String badCredsProfileName = "doc-example-resilience-
prof-bc";

    public static final String targetGroupName = "doc-example-resilience-tg";
    public static final String autoScalingGroupName = "doc-example-resilience-
group";
    public static final String lbName = "doc-example-resilience-lb";
    public static final String protocol = "HTTP";
    public static final int port = 80;
```

```
   public static final String DASHES = new String(new char[80]).replace("\0",
"-");

   public static void main(String[] args) throws IOException,
InterruptedException {
       Scanner in = new Scanner(System.in);
       Database database = new Database();
       AutoScaler autoScaler = new AutoScaler();
       LoadBalancer loadBalancer = new LoadBalancer();

       System.out.println(DASHES);
       System.out.println("Welcome to the demonstration of How to Build and
Manage a Resilient Service!");
       System.out.println(DASHES);

       System.out.println(DASHES);
       System.out.println("A - SETUP THE RESOURCES");
       System.out.println("Press Enter when you're ready to start deploying
resources.");
       in.nextLine();
       deploy(loadBalancer);
       System.out.println(DASHES);
       System.out.println(DASHES);
       System.out.println("B - DEMO THE RESILIENCE FUNCTIONALITY");
       System.out.println("Press Enter when you're ready.");
       in.nextLine();
       demo(loadBalancer);
       System.out.println(DASHES);

       System.out.println(DASHES);
       System.out.println("C - DELETE THE RESOURCES");
       System.out.println("""
               This concludes the demo of how to build and manage a resilient
service.
               To keep things tidy and to avoid unwanted charges on your
account, we can clean up all AWS resources
               that were created for this demo.
               """);

       System.out.println("\n Do you want to delete the resources (y/n)? ");
       String userInput = in.nextLine().trim().toLowerCase(); // Capture user
input

       if (userInput.equals("y")) {
```

```
            // Delete resources here
            deleteResources(loadBalancer, autoScaler, database);
            System.out.println("Resources deleted.");
        } else {
            System.out.println("""
                    Okay, we'll leave the resources intact.
                    Don't forget to delete them when you're done with them or you
might incur unexpected charges.
                    """);
        }
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("The example has completed. ");
        System.out.println("\n Thanks for watching!");
        System.out.println(DASHES);
    }

    // Deletes the AWS resources used in this example.
    private static void deleteResources(LoadBalancer loadBalancer, AutoScaler
autoScaler, Database database)
            throws IOException, InterruptedException {
        loadBalancer.deleteLoadBalancer(lbName);
        System.out.println("*** Wait 30 secs for resource to be deleted");
        TimeUnit.SECONDS.sleep(30);
        loadBalancer.deleteTargetGroup(targetGroupName);
        autoScaler.deleteAutoScaleGroup(autoScalingGroupName);
        autoScaler.deleteRolesPolicies(policyName, roleName, profileName);
        autoScaler.deleteTemplate(templateName);
        database.deleteTable(tableName);
    }

    private static void deploy(LoadBalancer loadBalancer) throws
InterruptedException, IOException {
        Scanner in = new Scanner(System.in);
        System.out.println(
                """
                        For this demo, we'll use the AWS SDK for Java (v2) to
create several AWS resources
                        to set up a load-balanced web service endpoint and
explore some ways to make it resilient
                        against various kinds of failures.

                        Some of the resources create by this demo are:
```

```
                        \t* A DynamoDB table that the web service depends on to
provide book, movie, and song recommendations.
                        \t* An EC2 launch template that defines EC2 instances
that each contain a Python web server.
                        \t* An EC2 Auto Scaling group that manages EC2 instances
across several Availability Zones.
                        \t* An Elastic Load Balancing (ELB) load balancer that
targets the Auto Scaling group to distribute requests.
                        """);

        System.out.println("Press Enter when you're ready.");
        in.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Creating and populating a DynamoDB table named " +
tableName);
        Database database = new Database();
        database.createTable(tableName, fileName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
                Creating an EC2 launch template that runs '{startup_script}' when
an instance starts.
                This script starts a Python web server defined in the `server.py`
script. The web server
                listens to HTTP requests on port 80 and responds to requests to
'/' and to '/healthcheck'.
                For demo purposes, this server is run as the root user. In
production, the best practice is to
                run a web server, such as Apache, with least-privileged
credentials.

                The template also defines an IAM policy that each instance uses
to assume a role that grants
                permissions to access the DynamoDB recommendation table and
Systems Manager parameters
                that control the flow of the demo.
                """);

        LaunchTemplateCreator templateCreator = new LaunchTemplateCreator();
        templateCreator.createTemplate(policyFile, policyName, profileName,
startScript, templateName, roleName);
```

```
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(
                "Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different Availability Zone.");
        System.out.println("*** Wait 30 secs for the VPC to be created");
        TimeUnit.SECONDS.sleep(30);
        AutoScaler autoScaler = new AutoScaler();
        String[] zones = autoScaler.createGroup(3, templateName,
autoScalingGroupName);

        System.out.println("""
                At this point, you have EC2 instances created. Once each instance
starts, it listens for
                HTTP requests. You can see these instances in the console or
continue with the demo.
                Press Enter when you're ready to continue.
                """);

        in.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Creating variables that control the flow of the
demo.");
        ParameterHelper paramHelper = new ParameterHelper();
        paramHelper.reset();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
                Creating an Elastic Load Balancing target group and load
balancer. The target group
                defines how the load balancer connects to instances. The load
balancer provides a
                single endpoint where clients connect and dispatches requests to
instances in the group.
                """);

        String vpcId = autoScaler.getDefaultVPC();
        List<Subnet> subnets = autoScaler.getSubnets(vpcId, zones);
        System.out.println("You have retrieved a list with " + subnets.size() + "
subnets");
```

```
        String targetGroupArn = loadBalancer.createTargetGroup(protocol, port,
vpcId, targetGroupName);
        String elbDnsName = loadBalancer.createLoadBalancer(subnets,
targetGroupArn, lbName, port, protocol);
        autoScaler.attachLoadBalancerTargetGroup(autoScalingGroupName,
targetGroupArn);
        System.out.println("Verifying access to the load balancer endpoint...");
        boolean wasSuccessul =
loadBalancer.verifyLoadBalancerEndpoint(elbDnsName);
        if (!wasSuccessul) {
            System.out.println("Couldn't connect to the load balancer, verifying
that the port is open...");
            CloseableHttpClient httpClient = HttpClients.createDefault();

            // Create an HTTP GET request to "http://checkip.amazonaws.com"
            HttpGet httpGet = new HttpGet("http://checkip.amazonaws.com");
            try {
                // Execute the request and get the response
                HttpResponse response = httpClient.execute(httpGet);

                // Read the response content.
                String ipAddress =
IOUtils.toString(response.getEntity().getContent(),
StandardCharsets.UTF_8).trim();

                // Print the public IP address.
                System.out.println("Public IP Address: " + ipAddress);
                GroupInfo groupInfo = autoScaler.verifyInboundPort(vpcId, port,
ipAddress);
                if (!groupInfo.isPortOpen()) {
                    System.out.println("""
                            For this example to work, the default security group
for your default VPC must
                            allow access from this computer. You can either add
it automatically from this
                            example or add it yourself using the AWS Management
Console.
                            """);

                    System.out.println(
                            "Do you want to add a rule to security group " +
groupInfo.getGroupName() + " to allow");
                    System.out.println("inbound traffic on port " + port + " from
your computer's IP address (y/n) ");
```

```
                    String ans = in.nextLine();
                    if ("y".equalsIgnoreCase(ans)) {
                        autoScaler.openInboundPort(groupInfo.getGroupName(),
String.valueOf(port), ipAddress);
                        System.out.println("Security group rule added.");
                    } else {
                        System.out.println("No security group rule added.");
                    }
                }

            } catch (AutoScalingException e) {
                e.printStackTrace();
            }
        } else if (wasSuccessul) {
            System.out.println("Your load balancer is ready. You can access it by
browsing to:");
            System.out.println("\t http://" + elbDnsName);
        } else {
            System.out.println("Couldn't get a successful response from the load
balancer endpoint. Troubleshoot by");
            System.out.println("manually verifying that your VPC and security
group are configured correctly and that");
            System.out.println("you can successfully make a GET request to the
load balancer.");
        }

        System.out.println("Press Enter when you're ready to continue with the
demo.");
        in.nextLine();
    }

    // A method that controls the demo part of the Java program.
    public static void demo(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
        ParameterHelper paramHelper = new ParameterHelper();
        System.out.println("Read the ssm_only_policy.json file");
        String ssmOnlyPolicy = readFileAsString(ssmJSON);

        System.out.println("Resetting parameters to starting values for demo.");
        paramHelper.reset();

        System.out.println(
                """
```

```
                        This part of the demonstration shows how to toggle
different parts of the system
                        to create situations where the web service fails, and
shows how using a resilient
                        architecture can keep the web service running in spite
of these failures.

                        At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.
                       """);
        demoChoices(loadBalancer);

        System.out.println(
                """
                        The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.
                        The table name is contained in a Systems Manager
parameter named self.param_helper.table.
                        To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.
                       """);
        paramHelper.put(paramHelper.tableName, "this-is-not-a-table");

        System.out.println(
                """
                        \nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as
                        healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.
                       """);
        demoChoices(loadBalancer);

        System.out.println(
                """
                       Instead of failing when the recommendation service fails,
the web service can return a static response.
                       While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.
                       """);
        paramHelper.put(paramHelper.failureResponse, "static");

        System.out.println("""
                Now, sending a GET request to the load balancer endpoint returns
a static response.
```

```
                The service still reports as healthy because health checks are
still shallow.
                """);
        demoChoices(loadBalancer);

        System.out.println("Let's reinstate the recommendation service.");
        paramHelper.put(paramHelper.tableName, paramHelper.dyntable);

        System.out.println("""
                Let's also substitute bad credentials for one of the instances in
the target group so that it can't
                access the DynamoDB recommendation table. We will get an instance
id value.
                """);

        LaunchTemplateCreator templateCreator = new LaunchTemplateCreator();
        AutoScaler autoScaler = new AutoScaler();

        // Create a new instance profile based on badCredsProfileName.
        templateCreator.createInstanceProfile(policyFile, policyName,
badCredsProfileName, roleName);
        String badInstanceId = autoScaler.getBadInstance(autoScalingGroupName);
        System.out.println("The bad instance id values used for this demo is " +
badInstanceId);

        String profileAssociationId =
autoScaler.getInstanceProfile(badInstanceId);
        System.out.println("The association Id value is " +
profileAssociationId);
        System.out.println("Replacing the profile for instance " + badInstanceId
                + " with a profile that contains bad credentials");
        autoScaler.replaceInstanceProfile(badInstanceId, badCredsProfileName,
profileAssociationId);

        System.out.println(
                """
                        Now, sending a GET request to the load balancer endpoint
returns either a recommendation or a static response,
                        depending on which instance is selected by the load
balancer.
                        """);

        demoChoices(loadBalancer);
```

```
        System.out.println("""
                Let's implement a deep health check. For this demo, a deep health
check tests whether
                the web service can access the DynamoDB table that it depends on
for recommendations. Note that
                the deep health check is only for ELB routing and not for Auto
Scaling instance health.
                This kind of deep health check is not recommended for Auto
Scaling instance health, because it
                risks accidental termination of all instances in the Auto Scaling
group when a dependent service fails.
                """);

        System.out.println("""
                By implementing deep health checks, the load balancer can detect
when one of the instances is failing
                and take that instance out of rotation.
                """);

        paramHelper.put(paramHelper.healthCheck, "deep");

        System.out.println("""
                Now, checking target health indicates that the instance with bad
credentials
                is unhealthy. Note that it might take a minute or two for the
load balancer to detect the unhealthy
                instance. Sending a GET request to the load balancer endpoint
always returns a recommendation, because
                the load balancer takes unhealthy instances out of its rotation.
                """);

        demoChoices(loadBalancer);

        System.out.println(
                """
                        Because the instances in this demo are controlled by an
auto scaler, the simplest way to fix an unhealthy
                        instance is to terminate it and let the auto scaler start
a new instance to replace it.
                        """);
        autoScaler.terminateInstance(badInstanceId);

        System.out.println("""
```

```
                Even while the instance is terminating and the new instance is
starting, sending a GET
                request to the web service continues to get a successful
recommendation response because
                the load balancer routes requests to the healthy instances. After
the replacement instance
                starts and reports as healthy, it is included in the load
balancing rotation.
                Note that terminating and replacing an instance typically takes
several minutes, during which time you
                can see the changing health check status until the new instance
is running and healthy.
                """);

        demoChoices(loadBalancer);
        System.out.println(
                "If the recommendation service fails now, deep health checks mean
all instances report as unhealthy.");
        paramHelper.put(paramHelper.tableName, "this-is-not-a-table");

        demoChoices(loadBalancer);
        paramHelper.reset();
    }

    public static void demoChoices(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
        String[] actions = {
                "Send a GET request to the load balancer endpoint.",
                "Check the health of load balancer targets.",
                "Go to the next part of the demo."
        };
        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.println("-".repeat(88));
            System.out.println("See the current state of the service by selecting
one of the following choices:");
            for (int i = 0; i < actions.length; i++) {
                System.out.println(i + ": " + actions[i]);
            }

            try {
                System.out.print("\nWhich action would you like to take? ");
                int choice = scanner.nextInt();
```

```
                System.out.println("-".repeat(88));

                switch (choice) {
                    case 0 -> {
                        System.out.println("Request:\n");
                        System.out.println("GET http://" +
loadBalancer.getEndpoint(lbName));
                        CloseableHttpClient httpClient =
HttpClients.createDefault();

                        // Create an HTTP GET request to the ELB.
                        HttpGet httpGet = new HttpGet("http://" +
loadBalancer.getEndpoint(lbName));

                        // Execute the request and get the response.
                        HttpResponse response = httpClient.execute(httpGet);
                        int statusCode =
response.getStatusLine().getStatusCode();
                        System.out.println("HTTP Status Code: " + statusCode);

                        // Display the JSON response
                        BufferedReader reader = new BufferedReader(
                                new
InputStreamReader(response.getEntity().getContent()));
                        StringBuilder jsonResponse = new StringBuilder();
                        String line;
                        while ((line = reader.readLine()) != null) {
                            jsonResponse.append(line);
                        }
                        reader.close();

                        // Print the formatted JSON response.
                        System.out.println("Full Response:\n");
                        System.out.println(jsonResponse.toString());

                        // Close the HTTP client.
                        httpClient.close();

                    }
                    case 1 -> {
                        System.out.println("\nChecking the health of load
balancer targets:\n");
                        List<TargetHealthDescription> health =
loadBalancer.checkTargetHealth(targetGroupName);
```

```
                        for (TargetHealthDescription target : health) {
                            System.out.printf("\tTarget %s on port %d is %s%n",
 target.target().id(),
                                              target.target().port(),
 target.targetHealth().stateAsString());
                        }
                        System.out.println("""
                                Note that it can take a minute or two for the
 health check to update
                                after changes are made.
                                """);
                    }
                    case 2 -> {
                        System.out.println("\nOkay, let's move on.");
                        System.out.println("-".repeat(88));
                        return; // Exit the method when choice is 2
                    }
                    default -> System.out.println("You must choose a value
 between 0-2. Please select again.");
                }

            } catch (java.util.InputMismatchException e) {
                System.out.println("Invalid input. Please select again.");
                scanner.nextLine(); // Clear the input buffer.
            }
        }
    }

    public static String readFileAsString(String filePath) throws IOException {
        byte[] bytes = Files.readAllBytes(Paths.get(filePath));
        return new String(bytes);
    }
}
```

Auto Scaling과 Amazon EC2 작업을 래핑하는 클래스를 생성합니다.

```
public class AutoScaler {

    private static Ec2Client ec2Client;
    private static AutoScalingClient autoScalingClient;
    private static IamClient iamClient;
```

```
    private static SsmClient ssmClient;

    private IamClient getIAMClient() {
        if (iamClient == null) {
            iamClient = IamClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return iamClient;
    }

    private SsmClient getSSMClient() {
        if (ssmClient == null) {
            ssmClient = SsmClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return ssmClient;
    }

    private Ec2Client getEc2Client() {
        if (ec2Client == null) {
            ec2Client = Ec2Client.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return ec2Client;
    }

    private AutoScalingClient getAutoScalingClient() {
        if (autoScalingClient == null) {
            autoScalingClient = AutoScalingClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return autoScalingClient;
    }

    /**
     * Terminates and instances in an EC2 Auto Scaling group. After an instance
is
     * terminated, it can no longer be accessed.
     */
    public void terminateInstance(String instanceId) {
```

```
        TerminateInstanceInAutoScalingGroupRequest terminateInstanceIRequest =
TerminateInstanceInAutoScalingGroupRequest
                .builder()
                .instanceId(instanceId)
                .shouldDecrementDesiredCapacity(false)
                .build();


getAutoScalingClient().terminateInstanceInAutoScalingGroup(terminateInstanceIRequest);
        System.out.format("Terminated instance %s.", instanceId);
    }

    /**
     * Replaces the profile associated with a running instance. After the profile
is
     * replaced, the instance is rebooted to ensure that it uses the new profile.
     * When
     * the instance is ready, Systems Manager is used to restart the Python web
     * server.
     */
    public void replaceInstanceProfile(String instanceId, String
newInstanceProfileName, String profileAssociationId)
            throws InterruptedException {
        // Create an IAM instance profile specification.
        software.amazon.awssdk.services.ec2.model.IamInstanceProfileSpecification
iamInstanceProfile =
software.amazon.awssdk.services.ec2.model.IamInstanceProfileSpecification
                .builder()
                .name(newInstanceProfileName) // Make sure
'newInstanceProfileName' is a valid IAM Instance Profile
                                              // name.
                .build();

        // Replace the IAM instance profile association for the EC2 instance.
        ReplaceIamInstanceProfileAssociationRequest replaceRequest =
ReplaceIamInstanceProfileAssociationRequest
                .builder()
                .iamInstanceProfile(iamInstanceProfile)
                .associationId(profileAssociationId) // Make sure
'profileAssociationId' is a valid association ID.
                .build();

        try {
            getEc2Client().replaceIamInstanceProfileAssociation(replaceRequest);
```

```
            // Handle the response as needed.
        } catch (Ec2Exception e) {
            // Handle exceptions, log, or report the error.
            System.err.println("Error: " + e.getMessage());
        }
        System.out.format("Replaced instance profile for association %s with
profile %s.", profileAssociationId,
                newInstanceProfileName);
        TimeUnit.SECONDS.sleep(15);
        boolean instReady = false;
        int tries = 0;

        // Reboot after 60 seconds
        while (!instReady) {
            if (tries % 6 == 0) {
                getEc2Client().rebootInstances(RebootInstancesRequest.builder()
                        .instanceIds(instanceId)
                        .build());
                System.out.println("Rebooting instance " + instanceId + " and
waiting for it to be ready.");
            }
            tries++;
            try {
                TimeUnit.SECONDS.sleep(10);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }

            DescribeInstanceInformationResponse informationResponse =
getSSMClient().describeInstanceInformation();
            List<InstanceInformation> instanceInformationList =
informationResponse.instanceInformationList();
            for (InstanceInformation info : instanceInformationList) {
                if (info.instanceId().equals(instanceId)) {
                    instReady = true;
                    break;
                }
            }
        }

        SendCommandRequest sendCommandRequest = SendCommandRequest.builder()
                .instanceIds(instanceId)
                .documentName("AWS-RunShellScript")
                .parameters(Collections.singletonMap("commands",
```

```
                        Collections.singletonList("cd / && sudo python3 server.py
80")))
                .build();

        getSSMClient().sendCommand(sendCommandRequest);
        System.out.println("Restarted the Python web server on instance " +
instanceId + ".");
    }

    public void openInboundPort(String secGroupId, String port, String ipAddress)
{
        AuthorizeSecurityGroupIngressRequest ingressRequest =
AuthorizeSecurityGroupIngressRequest.builder()
                .groupName(secGroupId)
                .cidrIp(ipAddress)
                .fromPort(Integer.parseInt(port))
                .build();

        getEc2Client().authorizeSecurityGroupIngress(ingressRequest);
        System.out.format("Authorized ingress to %s on port %s from %s.",
secGroupId, port, ipAddress);
    }

    /**
     * Detaches a role from an instance profile, detaches policies from the role,
     * and deletes all the resources.
     */
    public void deleteInstanceProfile(String roleName, String profileName) {
        try {
            software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
getInstanceProfileRequest =
software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
                    .builder()
                    .instanceProfileName(profileName)
                    .build();

            GetInstanceProfileResponse response =
getIAMClient().getInstanceProfile(getInstanceProfileRequest);
            String name = response.instanceProfile().instanceProfileName();
            System.out.println(name);

            RemoveRoleFromInstanceProfileRequest profileRequest =
RemoveRoleFromInstanceProfileRequest.builder()
                    .instanceProfileName(profileName)
```

```
                    .roleName(roleName)
                    .build();

            getIAMClient().removeRoleFromInstanceProfile(profileRequest);
            DeleteInstanceProfileRequest deleteInstanceProfileRequest =
DeleteInstanceProfileRequest.builder()
                    .instanceProfileName(profileName)
                    .build();

            getIAMClient().deleteInstanceProfile(deleteInstanceProfileRequest);
            System.out.println("Deleted instance profile " + profileName);

            DeleteRoleRequest deleteRoleRequest = DeleteRoleRequest.builder()
                    .roleName(roleName)
                    .build();

            // List attached role policies.
            ListAttachedRolePoliciesResponse rolesResponse = getIAMClient()
                    .listAttachedRolePolicies(role -> role.roleName(roleName));
            List<AttachedPolicy> attachedPolicies =
rolesResponse.attachedPolicies();
            for (AttachedPolicy attachedPolicy : attachedPolicies) {
                DetachRolePolicyRequest request =
DetachRolePolicyRequest.builder()
                        .roleName(roleName)
                        .policyArn(attachedPolicy.policyArn())
                        .build();

                getIAMClient().detachRolePolicy(request);
                System.out.println("Detached and deleted policy " +
attachedPolicy.policyName());
            }

            getIAMClient().deleteRole(deleteRoleRequest);
            System.out.println("Instance profile and role deleted.");

        } catch (IamException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public void deleteTemplate(String templateName) {
```

```
        getEc2Client().deleteLaunchTemplate(name ->
name.launchTemplateName(templateName));
        System.out.format(templateName + " was deleted.");
    }

    public void deleteAutoScaleGroup(String groupName) {
        DeleteAutoScalingGroupRequest deleteAutoScalingGroupRequest =
DeleteAutoScalingGroupRequest.builder()
                .autoScalingGroupName(groupName)
                .forceDelete(true)
                .build();


getAutoScalingClient().deleteAutoScalingGroup(deleteAutoScalingGroupRequest);
        System.out.println(groupName + " was deleted.");
    }

    /*
     * Verify the default security group of the specified VPC allows ingress from
     * this
     * computer. This can be done by allowing ingress from this computer's IP
     * address. In some situations, such as connecting from a corporate network,
you
     * must instead specify a prefix list ID. You can also temporarily open the
port
     * to
     * any IP address while running this example. If you do, be sure to remove
     * public
     * access when you're done.
     *
     */
    public GroupInfo verifyInboundPort(String VPC, int port, String ipAddress) {
        boolean portIsOpen = false;
        GroupInfo groupInfo = new GroupInfo();
        try {
            Filter filter = Filter.builder()
                    .name("group-name")
                    .values("default")
                    .build();

            Filter filter1 = Filter.builder()
                    .name("vpc-id")
                    .values(VPC)
                    .build();
```

```
            DescribeSecurityGroupsRequest securityGroupsRequest =
DescribeSecurityGroupsRequest.builder()
                    .filters(filter, filter1)
                    .build();

            DescribeSecurityGroupsResponse securityGroupsResponse =
getEc2Client()
                    .describeSecurityGroups(securityGroupsRequest);
            String securityGroup =
securityGroupsResponse.securityGroups().get(0).groupName();
            groupInfo.setGroupName(securityGroup);

            for (SecurityGroup secGroup :
securityGroupsResponse.securityGroups()) {
                System.out.println("Found security group: " +
secGroup.groupId());

                for (IpPermission ipPermission : secGroup.ipPermissions()) {
                    if (ipPermission.fromPort() == port) {
                        System.out.println("Found inbound rule: " +
ipPermission);
                        for (IpRange ipRange : ipPermission.ipRanges()) {
                            String cidrIp = ipRange.cidrIp();
                            if (cidrIp.startsWith(ipAddress) ||
cidrIp.equals("0.0.0.0/0")) {
                                System.out.println(cidrIp + " is applicable");
                                portIsOpen = true;
                            }
                        }

                        if (!ipPermission.prefixListIds().isEmpty()) {
                            System.out.println("Prefix lList is applicable");
                            portIsOpen = true;
                        }

                        if (!portIsOpen) {
                            System.out
                                    .println("The inbound rule does not appear to
be open to either this computer's IP,"
                                            + " all IP addresses (0.0.0.0/0), or
to a prefix list ID.");
                        } else {
                            break;
```

```
                        }
                    }
                }
            }

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }

        groupInfo.setPortOpen(portIsOpen);
        return groupInfo;
    }

    /*
     * Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
     * Scaling group.
     * The target group specifies how the load balancer forward requests to the
     * instances
     * in the group.
     */
    public void attachLoadBalancerTargetGroup(String asGroupName, String
targetGroupARN) {
        try {
            AttachLoadBalancerTargetGroupsRequest targetGroupsRequest =
AttachLoadBalancerTargetGroupsRequest.builder()
                    .autoScalingGroupName(asGroupName)
                    .targetGroupARNs(targetGroupARN)
                    .build();


getAutoScalingClient().attachLoadBalancerTargetGroups(targetGroupsRequest);
            System.out.println("Attached load balancer to " + asGroupName);

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Creates an EC2 Auto Scaling group with the specified size.
    public String[] createGroup(int groupSize, String templateName, String
autoScalingGroupName) {

        // Get availability zones.
```

```
software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
zonesRequest =
software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
               .builder()
               .build();

       DescribeAvailabilityZonesResponse zonesResponse =
getEc2Client().describeAvailabilityZones(zonesRequest);
       List<String> availabilityZoneNames =
zonesResponse.availabilityZones().stream()

.map(software.amazon.awssdk.services.ec2.model.AvailabilityZone::zoneName)
               .collect(Collectors.toList());

       String availabilityZones = String.join(",", availabilityZoneNames);
       LaunchTemplateSpecification specification =
LaunchTemplateSpecification.builder()
               .launchTemplateName(templateName)
               .version("$Default")
               .build();

       String[] zones = availabilityZones.split(",");
       CreateAutoScalingGroupRequest groupRequest =
CreateAutoScalingGroupRequest.builder()
               .launchTemplate(specification)
               .availabilityZones(zones)
               .maxSize(groupSize)
               .minSize(groupSize)
               .autoScalingGroupName(autoScalingGroupName)
               .build();

       try {
           getAutoScalingClient().createAutoScalingGroup(groupRequest);

       } catch (AutoScalingException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
       System.out.println("Created an EC2 Auto Scaling group named " +
autoScalingGroupName);
       return zones;
   }
```

```
    public String getDefaultVPC() {
        // Define the filter.
        Filter defaultFilter = Filter.builder()
                .name("is-default")
                .values("true")
                .build();

        software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest request =
software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest
                .builder()
                .filters(defaultFilter)
                .build();

        DescribeVpcsResponse response = getEc2Client().describeVpcs(request);
        return response.vpcs().get(0).vpcId();
    }

    // Gets the default subnets in a VPC for a specified list of Availability
Zones.
    public List<Subnet> getSubnets(String vpcId, String[] availabilityZones) {
        List<Subnet> subnets = null;
        Filter vpcFilter = Filter.builder()
                .name("vpc-id")
                .values(vpcId)
                .build();

        Filter azFilter = Filter.builder()
                .name("availability-zone")
                .values(availabilityZones)
                .build();

        Filter defaultForAZ = Filter.builder()
                .name("default-for-az")
                .values("true")
                .build();

        DescribeSubnetsRequest request = DescribeSubnetsRequest.builder()
                .filters(vpcFilter, azFilter, defaultForAZ)
                .build();

        DescribeSubnetsResponse response =
getEc2Client().describeSubnets(request);
        subnets = response.subnets();
        return subnets;
```

```
    }

    // Gets data about the instances in the EC2 Auto Scaling group.
    public String getBadInstance(String groupName) {
        DescribeAutoScalingGroupsRequest request =
DescribeAutoScalingGroupsRequest.builder()
                .autoScalingGroupNames(groupName)
                .build();

        DescribeAutoScalingGroupsResponse response =
getAutoScalingClient().describeAutoScalingGroups(request);
        AutoScalingGroup autoScalingGroup = response.autoScalingGroups().get(0);
        List<String> instanceIds = autoScalingGroup.instances().stream()
                .map(instance -> instance.instanceId())
                .collect(Collectors.toList());

        String[] instanceIdArray = instanceIds.toArray(new String[0]);
        for (String instanceId : instanceIdArray) {
            System.out.println("Instance ID: " + instanceId);
            return instanceId;
        }
        return "";
    }

    // Gets data about the profile associated with an instance.
    public String getInstanceProfile(String instanceId) {
        Filter filter = Filter.builder()
                .name("instance-id")
                .values(instanceId)
                .build();

        DescribeIamInstanceProfileAssociationsRequest associationsRequest =
DescribeIamInstanceProfileAssociationsRequest
                .builder()
                .filters(filter)
                .build();

        DescribeIamInstanceProfileAssociationsResponse response = getEc2Client()
                .describeIamInstanceProfileAssociations(associationsRequest);
        return response.iamInstanceProfileAssociations().get(0).associationId();
    }

    public void deleteRolesPolicies(String policyName, String roleName, String
InstanceProfile) {
```

```
        ListPoliciesRequest listPoliciesRequest =
 ListPoliciesRequest.builder().build();
        ListPoliciesResponse listPoliciesResponse =
 getIAMClient().listPolicies(listPoliciesRequest);
        for (Policy policy : listPoliciesResponse.policies()) {
            if (policy.policyName().equals(policyName)) {
                // List the entities (users, groups, roles) that are attached to
 the policy.

 software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
 listEntitiesRequest =
 software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
                        .builder()
                        .policyArn(policy.arn())
                        .build();
                ListEntitiesForPolicyResponse listEntitiesResponse = iamClient
                        .listEntitiesForPolicy(listEntitiesRequest);
                if (!listEntitiesResponse.policyGroups().isEmpty() || !
listEntitiesResponse.policyUsers().isEmpty()
                        || !listEntitiesResponse.policyRoles().isEmpty()) {
                    // Detach the policy from any entities it is attached to.
                    DetachRolePolicyRequest detachPolicyRequest =
 DetachRolePolicyRequest.builder()
                            .policyArn(policy.arn())
                            .roleName(roleName) // Specify the name of the IAM
 role
                            .build();

                    getIAMClient().detachRolePolicy(detachPolicyRequest);
                    System.out.println("Policy detached from entities.");
                }

                // Now, you can delete the policy.
                DeletePolicyRequest deletePolicyRequest =
 DeletePolicyRequest.builder()
                        .policyArn(policy.arn())
                        .build();

                getIAMClient().deletePolicy(deletePolicyRequest);
                System.out.println("Policy deleted successfully.");
                break;
            }
        }
```

```
        // List the roles associated with the instance profile
        ListInstanceProfilesForRoleRequest listRolesRequest =
 ListInstanceProfilesForRoleRequest.builder()
                .roleName(roleName)
                .build();

        // Detach the roles from the instance profile
        ListInstanceProfilesForRoleResponse listRolesResponse =
 iamClient.listInstanceProfilesForRole(listRolesRequest);
        for (software.amazon.awssdk.services.iam.model.InstanceProfile profile :
 listRolesResponse.instanceProfiles()) {
            RemoveRoleFromInstanceProfileRequest removeRoleRequest =
 RemoveRoleFromInstanceProfileRequest.builder()
                    .instanceProfileName(InstanceProfile)
                    .roleName(roleName) // Remove the extra dot here
                    .build();

            getIAMClient().removeRoleFromInstanceProfile(removeRoleRequest);
            System.out.println("Role " + roleName + " removed from instance
 profile " + InstanceProfile);
        }

        // Delete the instance profile after removing all roles
        DeleteInstanceProfileRequest deleteInstanceProfileRequest =
 DeleteInstanceProfileRequest.builder()
                .instanceProfileName(InstanceProfile)
                .build();

        getIAMClient().deleteInstanceProfile(r ->
 r.instanceProfileName(InstanceProfile));
        System.out.println(InstanceProfile + " Deleted");
        System.out.println("All roles and policies are deleted.");
    }
}
```

Elastic Load Balancing 작업을 래핑하는 클래스를 생성합니다.

```
public class LoadBalancer {
    public ElasticLoadBalancingV2Client elasticLoadBalancingV2Client;

    public ElasticLoadBalancingV2Client getLoadBalancerClient() {
        if (elasticLoadBalancingV2Client == null) {
```

```
            elasticLoadBalancingV2Client = ElasticLoadBalancingV2Client.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }

        return elasticLoadBalancingV2Client;
    }

    // Checks the health of the instances in the target group.
    public List<TargetHealthDescription> checkTargetHealth(String
targetGroupName) {
        DescribeTargetGroupsRequest targetGroupsRequest =
DescribeTargetGroupsRequest.builder()
                .names(targetGroupName)
                .build();

        DescribeTargetGroupsResponse tgResponse =
getLoadBalancerClient().describeTargetGroups(targetGroupsRequest);

        DescribeTargetHealthRequest healthRequest =
DescribeTargetHealthRequest.builder()

.targetGroupArn(tgResponse.targetGroups().get(0).targetGroupArn())
                .build();

        DescribeTargetHealthResponse healthResponse =
getLoadBalancerClient().describeTargetHealth(healthRequest);
        return healthResponse.targetHealthDescriptions();
    }

    // Gets the HTTP endpoint of the load balancer.
    public String getEndpoint(String lbName) {
        DescribeLoadBalancersResponse res = getLoadBalancerClient()
                .describeLoadBalancers(describe -> describe.names(lbName));
        return res.loadBalancers().get(0).dnsName();
    }

    // Deletes a load balancer.
    public void deleteLoadBalancer(String lbName) {
        try {
            // Use a waiter to delete the Load Balancer.
            DescribeLoadBalancersResponse res = getLoadBalancerClient()
                    .describeLoadBalancers(describe -> describe.names(lbName));
```

```
            ElasticLoadBalancingV2Waiter loadBalancerWaiter =
getLoadBalancerClient().waiter();
            DescribeLoadBalancersRequest request =
DescribeLoadBalancersRequest.builder()

.loadBalancerArns(res.loadBalancers().get(0).loadBalancerArn())
                    .build();

            getLoadBalancerClient().deleteLoadBalancer(
                    builder ->
builder.loadBalancerArn(res.loadBalancers().get(0).loadBalancerArn()));
            WaiterResponse<DescribeLoadBalancersResponse> waiterResponse =
loadBalancerWaiter
                    .waitUntilLoadBalancersDeleted(request);
            waiterResponse.matched().response().ifPresent(System.out::println);

        } catch (ElasticLoadBalancingV2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
        System.out.println(lbName + " was deleted.");
    }

    // Deletes the target group.
    public void deleteTargetGroup(String targetGroupName) {
        try {
            DescribeTargetGroupsResponse res = getLoadBalancerClient()
                    .describeTargetGroups(describe ->
describe.names(targetGroupName));
            getLoadBalancerClient()
                    .deleteTargetGroup(builder ->
builder.targetGroupArn(res.targetGroups().get(0).targetGroupArn()));
        } catch (ElasticLoadBalancingV2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
        System.out.println(targetGroupName + " was deleted.");
    }

    // Verify this computer can successfully send a GET request to the load
balancer
    // endpoint.
    public boolean verifyLoadBalancerEndpoint(String elbDnsName) throws
IOException, InterruptedException {
        boolean success = false;
        int retries = 3;
```

```
        CloseableHttpClient httpClient = HttpClients.createDefault();

        // Create an HTTP GET request to the ELB.
        HttpGet httpGet = new HttpGet("http://" + elbDnsName);
        try {
            while ((!success) && (retries > 0)) {
                // Execute the request and get the response.
                HttpResponse response = httpClient.execute(httpGet);
                int statusCode = response.getStatusLine().getStatusCode();
                System.out.println("HTTP Status Code: " + statusCode);
                if (statusCode == 200) {
                    success = true;
                } else {
                    retries--;
                    System.out.println("Got connection error from load balancer
endpoint, retrying...");
                    TimeUnit.SECONDS.sleep(15);
                }
            }

        } catch (org.apache.http.conn.HttpHostConnectException e) {
            System.out.println(e.getMessage());
        }

        System.out.println("Status.." + success);
        return success;
    }

    /*
     * Creates an Elastic Load Balancing target group. The target group specifies
     * how
     * the load balancer forward requests to instances in the group and how
instance
     * health is checked.
     */
    public String createTargetGroup(String protocol, int port, String vpcId,
String targetGroupName) {
        CreateTargetGroupRequest targetGroupRequest =
CreateTargetGroupRequest.builder()
                .healthCheckPath("/healthcheck")
                .healthCheckTimeoutSeconds(5)
                .port(port)
                .vpcId(vpcId)
                .name(targetGroupName)
```

```
                .protocol(protocol)
                .build();

        CreateTargetGroupResponse targetGroupResponse =
getLoadBalancerClient().createTargetGroup(targetGroupRequest);
        String targetGroupArn =
targetGroupResponse.targetGroups().get(0).targetGroupArn();
        String targetGroup =
targetGroupResponse.targetGroups().get(0).targetGroupName();
        System.out.println("The " + targetGroup + " was created with ARN" +
targetGroupArn);
        return targetGroupArn;
    }

    /*
     * Creates an Elastic Load Balancing load balancer that uses the specified
     * subnets
     * and forwards requests to the specified target group.
     */
    public String createLoadBalancer(List<Subnet> subnetIds, String
targetGroupARN, String lbName, int port,
            String protocol) {
        try {
            List<String> subnetIdStrings = subnetIds.stream()
                    .map(Subnet::subnetId)
                    .collect(Collectors.toList());

            CreateLoadBalancerRequest balancerRequest =
CreateLoadBalancerRequest.builder()
                    .subnets(subnetIdStrings)
                    .name(lbName)
                    .scheme("internet-facing")
                    .build();

            // Create and wait for the load balancer to become available.
            CreateLoadBalancerResponse lsResponse =
getLoadBalancerClient().createLoadBalancer(balancerRequest);
            String lbARN = lsResponse.loadBalancers().get(0).loadBalancerArn();

            ElasticLoadBalancingV2Waiter loadBalancerWaiter =
getLoadBalancerClient().waiter();
            DescribeLoadBalancersRequest request =
DescribeLoadBalancersRequest.builder()
                    .loadBalancerArns(lbARN)
```

```
                    .build();

            System.out.println("Waiting for Load Balancer " + lbName + " to
 become available.");
            WaiterResponse<DescribeLoadBalancersResponse> waiterResponse =
 loadBalancerWaiter
                    .waitUntilLoadBalancerAvailable(request);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Load Balancer " + lbName + " is available.");

            // Get the DNS name (endpoint) of the load balancer.
            String lbDNSName = lsResponse.loadBalancers().get(0).dnsName();
            System.out.println("*** Load Balancer DNS Name: " + lbDNSName);

            // Create a listener for the load balance.
            Action action = Action.builder()
                    .targetGroupArn(targetGroupARN)
                    .type("forward")
                    .build();

            CreateListenerRequest listenerRequest =
 CreateListenerRequest.builder()

 .loadBalancerArn(lsResponse.loadBalancers().get(0).loadBalancerArn())
                    .defaultActions(action)
                    .port(port)
                    .protocol(protocol)
                    .defaultActions(action)
                    .build();

            getLoadBalancerClient().createListener(listenerRequest);
            System.out.println("Created listener to forward traffic from load
 balancer " + lbName + " to target group "
                    + targetGroupARN);

            // Return the load balancer DNS name.
            return lbDNSName;

        } catch (ElasticLoadBalancingV2Exception e) {
            e.printStackTrace();
        }
        return "";
    }
}
```

DynamoDB를 사용하여 추천 서비스를 시뮬레이션하는 클래스를 생성합니다.

```
public class Database {

    private static DynamoDbClient dynamoDbClient;

    public static DynamoDbClient getDynamoDbClient() {
        if (dynamoDbClient == null) {
            dynamoDbClient = DynamoDbClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return dynamoDbClient;
    }

    // Checks to see if the Amazon DynamoDB table exists.
    private boolean doesTableExist(String tableName) {
        try {
            // Describe the table and catch any exceptions.
            DescribeTableRequest describeTableRequest =
 DescribeTableRequest.builder()
                    .tableName(tableName)
                    .build();

            getDynamoDbClient().describeTable(describeTableRequest);
            System.out.println("Table '" + tableName + "' exists.");
            return true;

        } catch (ResourceNotFoundException e) {
            System.out.println("Table '" + tableName + "' does not exist.");
        } catch (DynamoDbException e) {
            System.err.println("Error checking table existence: " +
 e.getMessage());
        }
        return false;
    }

    /*
     * Creates a DynamoDB table to use a recommendation service. The table has a
     * hash key named 'MediaType' that defines the type of media recommended,
 such
```

```
     * as
     * Book or Movie, and a range key named 'ItemId' that, combined with the
     * MediaType,
     * forms a unique identifier for the recommended item.
     */
    public void createTable(String tableName, String fileName) throws IOException
{
        // First check to see if the table exists.
        boolean doesExist = doesTableExist(tableName);
        if (!doesExist) {
            DynamoDbWaiter dbWaiter = getDynamoDbClient().waiter();
            CreateTableRequest createTableRequest = CreateTableRequest.builder()
                    .tableName(tableName)
                    .attributeDefinitions(
                            AttributeDefinition.builder()
                                    .attributeName("MediaType")
                                    .attributeType(ScalarAttributeType.S)
                                    .build(),
                            AttributeDefinition.builder()
                                    .attributeName("ItemId")
                                    .attributeType(ScalarAttributeType.N)
                                    .build())
                    .keySchema(
                            KeySchemaElement.builder()
                                    .attributeName("MediaType")
                                    .keyType(KeyType.HASH)
                                    .build(),
                            KeySchemaElement.builder()
                                    .attributeName("ItemId")
                                    .keyType(KeyType.RANGE)
                                    .build())
                    .provisionedThroughput(
                            ProvisionedThroughput.builder()
                                    .readCapacityUnits(5L)
                                    .writeCapacityUnits(5L)
                                    .build())
                    .build();

            getDynamoDbClient().createTable(createTableRequest);
            System.out.println("Creating table " + tableName + "...");

            // Wait until the Amazon DynamoDB table is created.
            DescribeTableRequest tableRequest = DescribeTableRequest.builder()
                    .tableName(tableName)
```

```
                    .build();

            WaiterResponse<DescribeTableResponse> waiterResponse =
dbWaiter.waitUntilTableExists(tableRequest);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Table " + tableName + " created.");

            // Add records to the table.
            populateTable(fileName, tableName);
        }
    }

    public void deleteTable(String tableName) {
        getDynamoDbClient().deleteTable(table -> table.tableName(tableName));
        System.out.println("Table " + tableName + " deleted.");
    }

    // Populates the table with data located in a JSON file using the DynamoDB
    // enhanced client.
    public void populateTable(String fileName, String tableName) throws
IOException {
        DynamoDbEnhancedClient enhancedClient = DynamoDbEnhancedClient.builder()
                .dynamoDbClient(getDynamoDbClient())
                .build();
        ObjectMapper objectMapper = new ObjectMapper();
        File jsonFile = new File(fileName);
        JsonNode rootNode = objectMapper.readTree(jsonFile);

        DynamoDbTable<Recommendation> mappedTable =
enhancedClient.table(tableName,
                TableSchema.fromBean(Recommendation.class));
        for (JsonNode currentNode : rootNode) {
            String mediaType = currentNode.path("MediaType").path("S").asText();
            int itemId = currentNode.path("ItemId").path("N").asInt();
            String title = currentNode.path("Title").path("S").asText();
            String creator = currentNode.path("Creator").path("S").asText();

            // Create a Recommendation object and set its properties.
            Recommendation rec = new Recommendation();
            rec.setMediaType(mediaType);
            rec.setItemId(itemId);
            rec.setTitle(title);
            rec.setCreator(creator);
```

```
            // Put the item into the DynamoDB table.
            mappedTable.putItem(rec); // Add the Recommendation to the list.
        }
        System.out.println("Added all records to the " + tableName);
    }
}
```

Systems Manager 작업을 래핑하는 클래스를 생성합니다.

```
public class ParameterHelper {

    String tableName = "doc-example-resilient-architecture-table";
    String dyntable = "doc-example-recommendation-service";
    String failureResponse = "doc-example-resilient-architecture-failure-
response";
    String healthCheck = "doc-example-resilient-architecture-health-check";

    public void reset() {
        put(dyntable, tableName);
        put(failureResponse, "none");
        put(healthCheck, "shallow");
    }

    public void put(String name, String value) {
        SsmClient ssmClient = SsmClient.builder()
                .region(Region.US_EAST_1)
                .build();

        PutParameterRequest parameterRequest = PutParameterRequest.builder()
                .name(name)
                .value(value)
                .overwrite(true)
                .type("String")
                .build();

        ssmClient.putParameter(parameterRequest);
        System.out.printf("Setting demo parameter %s to '%s'.", name, value);
    }
}
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 다음 항목을 참조하세요.

- AttachLoadBalancerTargetGroups
- CreateAutoScalingGroup
- CreateInstanceProfile
- CreateLaunchTemplate
- CreateListener
- CreateLoadBalancer
- CreateTargetGroup
- DeleteAutoScalingGroup
- DeleteInstanceProfile
- DeleteLaunchTemplate
- DeleteLoadBalancer
- DeleteTargetGroup
- DescribeAutoScalingGroups
- DescribeAvailabilityZones
- DescribeIamInstanceProfileAssociations
- DescribeInstances
- DescribeLoadBalancers
- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

명령 프롬프트에서 대화형 시나리오를 실행합니다.

```
#!/usr/bin/env node
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0


import {
  Scenario,
  parseScenarioArgs,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";

/**
 * The workflow steps are split into three stages:
 *   - deploy
 *   - demo
 *   - destroy
 *
 * Each of these stages has a corresponding file prefixed with steps-*.
 */
import { deploySteps } from "./steps-deploy.js";
import { demoSteps } from "./steps-demo.js";
import { destroySteps } from "./steps-destroy.js";

/**
 * The context is passed to every scenario. Scenario steps
 * will modify the context.
 */
const context = {};

/**
 * Three Scenarios are created for the workflow. A Scenario is an orchestration
 class
```

```
 * that simplifies running a series of steps.
 */
export const scenarios = {
  // Deploys all resources necessary for the workflow.
  deploy: new Scenario("Resilient Workflow - Deploy", deploySteps, context),
  // Demonstrates how a fragile web service can be made more resilient.
  demo: new Scenario("Resilient Workflow - Demo", demoSteps, context),
  // Destroys the resources created for the workflow.
  destroy: new Scenario("Resilient Workflow - Destroy", destroySteps, context),
};

// Call function if run directly
import { fileURLToPath } from "url";

if (process.argv[1] === fileURLToPath(import.meta.url)) {
  parseScenarioArgs(scenarios);
}
```

모든 리소스를 배포하기 위한 단계를 생성합니다.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { join } from "node:path";
import { readFileSync, writeFileSync } from "node:fs";
import axios from "axios";

import {
  BatchWriteItemCommand,
  CreateTableCommand,
  DynamoDBClient,
  waitUntilTableExists,
} from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  CreateKeyPairCommand,
  CreateLaunchTemplateCommand,
  DescribeAvailabilityZonesCommand,
  DescribeVpcsCommand,
  DescribeSubnetsCommand,
  DescribeSecurityGroupsCommand,
  AuthorizeSecurityGroupIngressCommand,
} from "@aws-sdk/client-ec2";
```

```
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  AttachRolePolicyCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import { SSMClient, GetParameterCommand } from "@aws-sdk/client-ssm";
import {
  CreateAutoScalingGroupCommand,
  AutoScalingClient,
  AttachLoadBalancerTargetGroupsCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  CreateListenerCommand,
  CreateLoadBalancerCommand,
  CreateTargetGroupCommand,
  ElasticLoadBalancingV2Client,
  waitUntilLoadBalancerAvailable,
} from "@aws-sdk/client-elastic-load-balancing-v2";

import {
  ScenarioOutput,
  ScenarioInput,
  ScenarioAction,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES, RESOURCES_PATH, ROOT } from "./constants.js";
import { initParamsSteps } from "./steps-reset-params.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const deploySteps = [
  new ScenarioOutput("introduction", MESSAGES.introduction, { header: true }),
  new ScenarioInput("confirmDeployment", MESSAGES.confirmDeployment, {
    type: "confirm",
  }),
  new ScenarioAction(
    "handleConfirmDeployment",
    (c) => c.confirmDeployment === false && process.exit(),
```

```
  ),
  new ScenarioOutput(
    "creatingTable",
    MESSAGES.creatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("createTable", async () => {
    const client = new DynamoDBClient({});
    await client.send(
      new CreateTableCommand({
        TableName: NAMES.tableName,
        ProvisionedThroughput: {
          ReadCapacityUnits: 5,
          WriteCapacityUnits: 5,
        },
        AttributeDefinitions: [
          {
            AttributeName: "MediaType",
            AttributeType: "S",
          },
          {
            AttributeName: "ItemId",
            AttributeType: "N",
          },
        ],
        KeySchema: [
          {
            AttributeName: "MediaType",
            KeyType: "HASH",
          },
          {
            AttributeName: "ItemId",
            KeyType: "RANGE",
          },
        ],
      }),
    );
    await waitUntilTableExists({ client }, { TableName: NAMES.tableName });
  }),
  new ScenarioOutput(
    "createdTable",
    MESSAGES.createdTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioOutput(
    "populatingTable",
```

```
    MESSAGES.populatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("populateTable", () => {
    const client = new DynamoDBClient({});
    /**
     * @type {{ default: import("@aws-sdk/client-dynamodb").PutRequest['Item']
[] }}
     */
    const recommendations = JSON.parse(
      readFileSync(join(RESOURCES_PATH, "recommendations.json")),
    );

    return client.send(
      new BatchWriteItemCommand({
        RequestItems: {
          [NAMES.tableName]: recommendations.map((item) => ({
            PutRequest: { Item: item },
          })),
        },
      }),
    );
  }),
  new ScenarioOutput(
    "populatedTable",
    MESSAGES.populatedTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioOutput(
    "creatingKeyPair",
    MESSAGES.creatingKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
  ),
  new ScenarioAction("createKeyPair", async () => {
    const client = new EC2Client({});
    const { KeyMaterial } = await client.send(
      new CreateKeyPairCommand({
        KeyName: NAMES.keyPairName,
      }),
    );

    writeFileSync(`${NAMES.keyPairName}.pem`, KeyMaterial, { mode: 0o600 });
  }),
  new ScenarioOutput(
    "createdKeyPair",
    MESSAGES.createdKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
  ),
```

```
  new ScenarioOutput(
    "creatingInstancePolicy",
    MESSAGES.creatingInstancePolicy.replace(
      "${INSTANCE_POLICY_NAME}",
      NAMES.instancePolicyName,
    ),
  ),
  new ScenarioAction("createInstancePolicy", async (state) => {
    const client = new IAMClient({});
    const {
      Policy: { Arn },
    } = await client.send(
      new CreatePolicyCommand({
        PolicyName: NAMES.instancePolicyName,
        PolicyDocument: readFileSync(
          join(RESOURCES_PATH, "instance_policy.json"),
        ),
      }),
    );
    state.instancePolicyArn = Arn;
  }),
  new ScenarioOutput("createdInstancePolicy", (state) =>
    MESSAGES.createdInstancePolicy
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_POLICY_ARN}", state.instancePolicyArn),
  ),
  new ScenarioOutput(
    "creatingInstanceRole",
    MESSAGES.creatingInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    ),
  ),
  new ScenarioAction("createInstanceRole", () => {
    const client = new IAMClient({});
    return client.send(
      new CreateRoleCommand({
        RoleName: NAMES.instanceRoleName,
        AssumeRolePolicyDocument: readFileSync(
          join(ROOT, "assume-role-policy.json"),
        ),
      }),
    );
  }),
```

```
  new ScenarioOutput(
    "createdInstanceRole",
    MESSAGES.createdInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    ),
  ),
  new ScenarioOutput(
    "attachingPolicyToRole",
    MESSAGES.attachingPolicyToRole
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName)
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName),
  ),
  new ScenarioAction("attachPolicyToRole", async (state) => {
    const client = new IAMClient({});
    await client.send(
      new AttachRolePolicyCommand({
        RoleName: NAMES.instanceRoleName,
        PolicyArn: state.instancePolicyArn,
      }),
    );
  }),
  new ScenarioOutput(
    "attachedPolicyToRole",
    MESSAGES.attachedPolicyToRole
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  new ScenarioOutput(
    "creatingInstanceProfile",
    MESSAGES.creatingInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.instanceProfileName,
    ),
  ),
  new ScenarioAction("createInstanceProfile", async (state) => {
    const client = new IAMClient({});
    const {
      InstanceProfile: { Arn },
    } = await client.send(
      new CreateInstanceProfileCommand({
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
```

```
    state.instanceProfileArn = Arn;

    await waitUntilInstanceProfileExists(
      { client },
      { InstanceProfileName: NAMES.instanceProfileName },
    );
  }),
  new ScenarioOutput("createdInstanceProfile", (state) =>
    MESSAGES.createdInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_PROFILE_ARN}", state.instanceProfileArn),
  ),
  new ScenarioOutput(
    "addingRoleToInstanceProfile",
    MESSAGES.addingRoleToInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  new ScenarioAction("addRoleToInstanceProfile", () => {
    const client = new IAMClient({});
    return client.send(
      new AddRoleToInstanceProfileCommand({
        RoleName: NAMES.instanceRoleName,
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
  }),
  new ScenarioOutput(
    "addedRoleToInstanceProfile",
    MESSAGES.addedRoleToInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  ...initParamsSteps,
  new ScenarioOutput("creatingLaunchTemplate", MESSAGES.creatingLaunchTemplate),
  new ScenarioAction("createLaunchTemplate", async () => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateLaunchTemplate]
    const ssmClient = new SSMClient({});
    const { Parameter } = await ssmClient.send(
      new GetParameterCommand({
        Name: "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
      }),
    );
    const ec2Client = new EC2Client({});
```

```
    await ec2Client.send(
      new CreateLaunchTemplateCommand({
        LaunchTemplateName: NAMES.launchTemplateName,
        LaunchTemplateData: {
          InstanceType: "t3.micro",
          ImageId: Parameter.Value,
          IamInstanceProfile: { Name: NAMES.instanceProfileName },
          UserData: readFileSync(
            join(RESOURCES_PATH, "server_startup_script.sh"),
          ).toString("base64"),
          KeyName: NAMES.keyPairName,
        },
      }),
      // snippet-end:[javascript.v3.wkflw.resilient.CreateLaunchTemplate]
    );
  }),
  new ScenarioOutput(
    "createdLaunchTemplate",
    MESSAGES.createdLaunchTemplate.replace(
      "${LAUNCH_TEMPLATE_NAME}",
      NAMES.launchTemplateName,
    ),
  ),
  new ScenarioOutput(
    "creatingAutoScalingGroup",
    MESSAGES.creatingAutoScalingGroup.replace(
      "${AUTO_SCALING_GROUP_NAME}",
      NAMES.autoScalingGroupName,
    ),
  ),
  new ScenarioAction("createAutoScalingGroup", async (state) => {
    const ec2Client = new EC2Client({});
    const { AvailabilityZones } = await ec2Client.send(
      new DescribeAvailabilityZonesCommand({}),
    );
    state.availabilityZoneNames = AvailabilityZones.map((az) => az.ZoneName);
    const autoScalingClient = new AutoScalingClient({});
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new CreateAutoScalingGroupCommand({
          AvailabilityZones: state.availabilityZoneNames,
          AutoScalingGroupName: NAMES.autoScalingGroupName,
          LaunchTemplate: {
            LaunchTemplateName: NAMES.launchTemplateName,
```

```
            Version: "$Default",
          },
          MinSize: 3,
          MaxSize: 3,
        }),
      ),
    );
  }),
  new ScenarioOutput(
    "createdAutoScalingGroup",
    /**
     * @param {{ availabilityZoneNames: string[] }} state
     */
    (state) =>
      MESSAGES.createdAutoScalingGroup
        .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName)
        .replace(
          "${AVAILABILITY_ZONE_NAMES}",
          state.availabilityZoneNames.join(", "),
        ),
  ),
  new ScenarioInput("confirmContinue", MESSAGES.confirmContinue, {
    type: "confirm",
  }),
  new ScenarioOutput("loadBalancer", MESSAGES.loadBalancer),
  new ScenarioOutput("gettingVpc", MESSAGES.gettingVpc),
  new ScenarioAction("getVpc", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.DescribeVpcs]
    const client = new EC2Client({});
    const { Vpcs } = await client.send(
      new DescribeVpcsCommand({
        Filters: [{ Name: "is-default", Values: ["true"] }],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.DescribeVpcs]
    state.defaultVpc = Vpcs[0].VpcId;
  }),
  new ScenarioOutput("gotVpc", (state) =>
    MESSAGES.gotVpc.replace("${VPC_ID}", state.defaultVpc),
  ),
  new ScenarioOutput("gettingSubnets", MESSAGES.gettingSubnets),
  new ScenarioAction("getSubnets", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.DescribeSubnets]
    const client = new EC2Client({});
```

```
    const { Subnets } = await client.send(
      new DescribeSubnetsCommand({
        Filters: [
          { Name: "vpc-id", Values: [state.defaultVpc] },
          { Name: "availability-zone", Values: state.availabilityZoneNames },
          { Name: "default-for-az", Values: ["true"] },
        ],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.DescribeSubnets]
    state.subnets = Subnets.map((subnet) => subnet.SubnetId);
  }),
  new ScenarioOutput(
    "gotSubnets",
    /**
     * @param {{ subnets: string[] }} state
     */
    (state) =>
      MESSAGES.gotSubnets.replace("${SUBNETS}", state.subnets.join(", ")),
  ),
  new ScenarioOutput(
    "creatingLoadBalancerTargetGroup",
    MESSAGES.creatingLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioAction("createLoadBalancerTargetGroup", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateTargetGroup]
    const client = new ElasticLoadBalancingV2Client({});
    const { TargetGroups } = await client.send(
      new CreateTargetGroupCommand({
        Name: NAMES.loadBalancerTargetGroupName,
        Protocol: "HTTP",
        Port: 80,
        HealthCheckPath: "/healthcheck",
        HealthCheckIntervalSeconds: 10,
        HealthCheckTimeoutSeconds: 5,
        HealthyThresholdCount: 2,
        UnhealthyThresholdCount: 2,
        VpcId: state.defaultVpc,
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.CreateTargetGroup]
```

```
    const targetGroup = TargetGroups[0];
    state.targetGroupArn = targetGroup.TargetGroupArn;
    state.targetGroupProtocol = targetGroup.Protocol;
    state.targetGroupPort = targetGroup.Port;
  }),
  new ScenarioOutput(
    "createdLoadBalancerTargetGroup",
    MESSAGES.createdLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioOutput(
    "creatingLoadBalancer",
    MESSAGES.creatingLoadBalancer.replace("${LB_NAME}", NAMES.loadBalancerName),
  ),
  new ScenarioAction("createLoadBalancer", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateLoadBalancer]
    const client = new ElasticLoadBalancingV2Client({});
    const { LoadBalancers } = await client.send(
      new CreateLoadBalancerCommand({
        Name: NAMES.loadBalancerName,
        Subnets: state.subnets,
      }),
    );
    state.loadBalancerDns = LoadBalancers[0].DNSName;
    state.loadBalancerArn = LoadBalancers[0].LoadBalancerArn;
    await waitUntilLoadBalancerAvailable(
      { client },
      { Names: [NAMES.loadBalancerName] },
    );
    // snippet-end:[javascript.v3.wkflw.resilient.CreateLoadBalancer]
  }),
  new ScenarioOutput("createdLoadBalancer", (state) =>
    MESSAGES.createdLoadBalancer
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${DNS_NAME}", state.loadBalancerDns),
  ),
  new ScenarioOutput(
    "creatingListener",
    MESSAGES.creatingLoadBalancerListener
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName),
  ),
```

```
  new ScenarioAction("createListener", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateListener]
    const client = new ElasticLoadBalancingV2Client({});
    const { Listeners } = await client.send(
      new CreateListenerCommand({
        LoadBalancerArn: state.loadBalancerArn,
        Protocol: state.targetGroupProtocol,
        Port: state.targetGroupPort,
        DefaultActions: [
          { Type: "forward", TargetGroupArn: state.targetGroupArn },
        ],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.CreateListener]
    const listener = Listeners[0];
    state.loadBalancerListenerArn = listener.ListenerArn;
  }),
  new ScenarioOutput("createdListener", (state) =>
    MESSAGES.createdLoadBalancerListener.replace(
      "${LB_LISTENER_ARN}",
      state.loadBalancerListenerArn,
    ),
  ),
  new ScenarioOutput(
    "attachingLoadBalancerTargetGroup",
    MESSAGES.attachingLoadBalancerTargetGroup
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName)
      .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName),
  ),
  new ScenarioAction("attachLoadBalancerTargetGroup", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.AttachTargetGroup]
    const client = new AutoScalingClient({});
    await client.send(
      new AttachLoadBalancerTargetGroupsCommand({
        AutoScalingGroupName: NAMES.autoScalingGroupName,
        TargetGroupARNs: [state.targetGroupArn],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.AttachTargetGroup]
  }),
  new ScenarioOutput(
    "attachedLoadBalancerTargetGroup",
    MESSAGES.attachedLoadBalancerTargetGroup,
  ),
```

```
  new ScenarioOutput("verifyingInboundPort", MESSAGES.verifyingInboundPort),
  new ScenarioAction(
    "verifyInboundPort",
    /**
     *
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup}} state
     */
    async (state) => {
      const client = new EC2Client({});
      const { SecurityGroups } = await client.send(
        new DescribeSecurityGroupsCommand({
          Filters: [{ Name: "group-name", Values: ["default"] }],
        }),
      );
      if (!SecurityGroups) {
        state.verifyInboundPortError = new Error(MESSAGES.noSecurityGroups);
      }
      state.defaultSecurityGroup = SecurityGroups[0];

      /**
       * @type {string}
       */
      const ipResponse = (await axios.get("http://checkip.amazonaws.com")).data;
      state.myIp = ipResponse.trim();
      const myIpRules = state.defaultSecurityGroup.IpPermissions.filter(
        ({ IpRanges }) =>
          IpRanges.some(
            ({ CidrIp }) =>
              CidrIp.startsWith(state.myIp) || CidrIp === "0.0.0.0/0",
          ),
      )
        .filter(({ IpProtocol }) => IpProtocol === "tcp")
        .filter(({ FromPort }) => FromPort === 80);

      state.myIpRules = myIpRules;
    },
  ),
  new ScenarioOutput(
    "verifiedInboundPort",
    /**
     * @param {{ myIpRules: any[] }} state
     */
    (state) => {
```

```
      if (state.myIpRules.length > 0) {
        return MESSAGES.foundIpRules.replace(
          "${IP_RULES}",
          JSON.stringify(state.myIpRules, null, 2),
        );
      } else {
        return MESSAGES.noIpRules;
      }
    },
  ),
  new ScenarioInput(
    "shouldAddInboundRule",
    /**
     * @param {{ myIpRules: any[] }} state
     */
    (state) => {
      if (state.myIpRules.length > 0) {
        return false;
      } else {
        return MESSAGES.noIpRules;
      }
    },
    { type: "confirm" },
  ),
  new ScenarioAction(
    "addInboundRule",
    /**
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup }} state
     */
    async (state) => {
      if (!state.shouldAddInboundRule) {
        return;
      }

      const client = new EC2Client({});
      await client.send(
        new AuthorizeSecurityGroupIngressCommand({
          GroupId: state.defaultSecurityGroup.GroupId,
          CidrIp: `${state.myIp}/32`,
          FromPort: 80,
          ToPort: 80,
          IpProtocol: "tcp",
        }),
```

```
      );
    },
  ),
  new ScenarioOutput("addedInboundRule", (state) => {
    if (state.shouldAddInboundRule) {
      return MESSAGES.addedInboundRule.replace("${IP_ADDRESS}", state.myIp);
    } else {
      return false;
    }
  }),
  new ScenarioOutput("verifyingEndpoint", (state) =>
    MESSAGES.verifyingEndpoint.replace("${DNS_NAME}", state.loadBalancerDns),
  ),
  new ScenarioAction("verifyEndpoint", async (state) => {
    try {
      const response = await retry({ intervalInMs: 2000, maxRetries: 30 }, () =>
        axios.get(`http://${state.loadBalancerDns}`),
      );
      state.endpointResponse = JSON.stringify(response.data, null, 2);
    } catch (e) {
      state.verifyEndpointError = e;
    }
  }),
  new ScenarioOutput("verifiedEndpoint", (state) => {
    if (state.verifyEndpointError) {
      console.error(state.verifyEndpointError);
    } else {
      return MESSAGES.verifiedEndpoint.replace(
        "${ENDPOINT_RESPONSE}",
        state.endpointResponse,
      );
    }
  }),
];
```

데모를 실행하기 위한 단계를 생성합니다.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { readFileSync } from "node:fs";
import { join } from "node:path";
```

```
import axios from "axios";

import {
  DescribeTargetGroupsCommand,
  DescribeTargetHealthCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";
import {
  DescribeInstanceInformationCommand,
  PutParameterCommand,
  SSMClient,
  SendCommandCommand,
} from "@aws-sdk/client-ssm";
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DescribeAutoScalingGroupsCommand,
  TerminateInstanceInAutoScalingGroupCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  DescribeIamInstanceProfileAssociationsCommand,
  EC2Client,
  RebootInstancesCommand,
  ReplaceIamInstanceProfileAssociationCommand,
} from "@aws-sdk/client-ec2";

import {
  ScenarioAction,
  ScenarioInput,
  ScenarioOutput,
} from "@aws-doc-sdk-examples/lib/scenario/scenario.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES, RESOURCES_PATH } from "./constants.js";
import { findLoadBalancer } from "./shared.js";
```

```
const getRecommendation = new ScenarioAction(
  "getRecommendation",
  async (state) => {
    const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
    if (loadBalancer) {
      state.loadBalancerDnsName = loadBalancer.DNSName;
      try {
        state.recommendation = (
          await axios.get(`http://${state.loadBalancerDnsName}`)
        ).data;
      } catch (e) {
        state.recommendation = e instanceof Error ? e.message : e;
      }
    } else {
      throw new Error(MESSAGES.demoFindLoadBalancerError);
    }
  },
);

const getRecommendationResult = new ScenarioOutput(
  "getRecommendationResult",
  (state) =>
    `Recommendation:\n${JSON.stringify(state.recommendation, null, 2)}`,
  { preformatted: true },
);

const getHealthCheck = new ScenarioAction("getHealthCheck", async (state) => {
  // snippet-start:[javascript.v3.wkflw.resilient.DescribeTargetGroups]
  const client = new ElasticLoadBalancingV2Client({});
  const { TargetGroups } = await client.send(
    new DescribeTargetGroupsCommand({
      Names: [NAMES.loadBalancerTargetGroupName],
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.DescribeTargetGroups]

  // snippet-start:[javascript.v3.wkflw.resilient.DescribeTargetHealth]
  const { TargetHealthDescriptions } = await client.send(
    new DescribeTargetHealthCommand({
      TargetGroupArn: TargetGroups[0].TargetGroupArn,
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.DescribeTargetHealth]
  state.targetHealthDescriptions = TargetHealthDescriptions;
```

```
});

const getHealthCheckResult = new ScenarioOutput(
  "getHealthCheckResult",
  /**
   * @param {{ targetHealthDescriptions: import('@aws-sdk/client-elastic-load-
balancing-v2').TargetHealthDescription[]}} state
   */
  (state) => {
    const status = state.targetHealthDescriptions
      .map((th) => `${th.Target.Id}: ${th.TargetHealth.State}`)
      .join("\n");
    return `Health check:\n${status}`;
  },
  { preformatted: true },
);

const loadBalancerLoop = new ScenarioAction(
  "loadBalancerLoop",
  getRecommendation.action,
  {
    whileConfig: {
      whileFn: ({ loadBalancerCheck }) => loadBalancerCheck,
      input: new ScenarioInput(
        "loadBalancerCheck",
        MESSAGES.demoLoadBalancerCheck,
        {
          type: "confirm",
        },
      ),
      output: getRecommendationResult,
    },
  },
);

const healthCheckLoop = new ScenarioAction(
  "healthCheckLoop",
  getHealthCheck.action,
  {
    whileConfig: {
      whileFn: ({ healthCheck }) => healthCheck,
      input: new ScenarioInput("healthCheck", MESSAGES.demoHealthCheck, {
        type: "confirm",
      }),
```

```
      output: getHealthCheckResult,
    },
  },
);

const statusSteps = [
  getRecommendation,
  getRecommendationResult,
  getHealthCheck,
  getHealthCheckResult,
];

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const demoSteps = [
  new ScenarioOutput("header", MESSAGES.demoHeader, { header: true }),
  new ScenarioOutput("sanityCheck", MESSAGES.demoSanityCheck),
  ...statusSteps,
  new ScenarioInput(
    "brokenDependencyConfirmation",
    MESSAGES.demoBrokenDependencyConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("brokenDependency", async (state) => {
    if (!state.brokenDependencyConfirmation) {
      process.exit();
    } else {
      const client = new SSMClient({});
      state.badTableName = `fake-table-${Date.now()}`;
      await client.send(
        new PutParameterCommand({
          Name: NAMES.ssmTableNameKey,
          Value: state.badTableName,
          Overwrite: true,
          Type: "String",
        }),
      );
    }
  }),
  new ScenarioOutput("testBrokenDependency", (state) =>
    MESSAGES.demoTestBrokenDependency.replace(
      "${TABLE_NAME}",
      state.badTableName,
```

```
    ),
  ),
  ...statusSteps,
  new ScenarioInput(
    "staticResponseConfirmation",
    MESSAGES.demoStaticResponseConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("staticResponse", async (state) => {
    if (!state.staticResponseConfirmation) {
      process.exit();
    } else {
      const client = new SSMClient({});
      await client.send(
        new PutParameterCommand({
          Name: NAMES.ssmFailureResponseKey,
          Value: "static",
          Overwrite: true,
          Type: "String",
        }),
      );
    }
  }),
  new ScenarioOutput("testStaticResponse", MESSAGES.demoTestStaticResponse),
  ...statusSteps,
  new ScenarioInput(
    "badCredentialsConfirmation",
    MESSAGES.demoBadCredentialsConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("badCredentialsExit", (state) => {
    if (!state.badCredentialsConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("fixDynamoDBName", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
```

```
    );
  }),
  new ScenarioAction(
    "badCredentials",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-auto-
scaling').Instance }} state
     */
    async (state) => {
      await createSsmOnlyInstanceProfile();
      const autoScalingClient = new AutoScalingClient({});
      const { AutoScalingGroups } = await autoScalingClient.send(
        new DescribeAutoScalingGroupsCommand({
          AutoScalingGroupNames: [NAMES.autoScalingGroupName],
        }),
      );
      state.targetInstance = AutoScalingGroups[0].Instances[0];
      // snippet-start:
[javascript.v3.wkflw.resilient.DescribeIamInstanceProfileAssociations]
      const ec2Client = new EC2Client({});
      const { IamInstanceProfileAssociations } = await ec2Client.send(
        new DescribeIamInstanceProfileAssociationsCommand({
          Filters: [
            { Name: "instance-id", Values: [state.targetInstance.InstanceId] },
          ],
        }),
      );
      // snippet-end:
[javascript.v3.wkflw.resilient.DescribeIamInstanceProfileAssociations]
      state.instanceProfileAssociationId =
        IamInstanceProfileAssociations[0].AssociationId;
      // snippet-start:
[javascript.v3.wkflw.resilient.ReplaceIamInstanceProfileAssociation]
      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
        ec2Client.send(
          new ReplaceIamInstanceProfileAssociationCommand({
            AssociationId: state.instanceProfileAssociationId,
            IamInstanceProfile: { Name: NAMES.ssmOnlyInstanceProfileName },
          }),
        ),
      );
      // snippet-end:
[javascript.v3.wkflw.resilient.ReplaceIamInstanceProfileAssociation]
```

```
      await ec2Client.send(
        new RebootInstancesCommand({
          InstanceIds: [state.targetInstance.InstanceId],
        }),
      );

      const ssmClient = new SSMClient({});
      await retry({ intervalInMs: 20000, maxRetries: 15 }, async () => {
        const { InstanceInformationList } = await ssmClient.send(
          new DescribeInstanceInformationCommand({}),
        );

        const instance = InstanceInformationList.find(
          (info) => info.InstanceId === state.targetInstance.InstanceId,
        );

        if (!instance) {
          throw new Error("Instance not found.");
        }
      });

      await ssmClient.send(
        new SendCommandCommand({
          InstanceIds: [state.targetInstance.InstanceId],
          DocumentName: "AWS-RunShellScript",
          Parameters: { commands: ["cd / && sudo python3 server.py 80"] },
        }),
      );
    },
  ),
  new ScenarioOutput(
    "testBadCredentials",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation}} state
     */
    (state) =>
      MESSAGES.demoTestBadCredentials.replace(
        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
      ),
  ),
  loadBalancerLoop,
  new ScenarioInput(
```

```
    "deepHealthCheckConfirmation",
    MESSAGES.demoDeepHealthCheckConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("deepHealthCheckExit", (state) => {
    if (!state.deepHealthCheckConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("deepHealthCheck", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmHealthCheckKey,
        Value: "deep",
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testDeepHealthCheck", MESSAGES.demoTestDeepHealthCheck),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput(
    "killInstanceConfirmation",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
     */
    (state) =>
      MESSAGES.demoKillInstanceConfirmation.replace(
        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
      ),
    { type: "confirm" },
  ),
  new ScenarioAction("killInstanceExit", (state) => {
    if (!state.killInstanceConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction(
    "killInstance",
    /**
```

```
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
     */
    async (state) => {
      const client = new AutoScalingClient({});
      await client.send(
        new TerminateInstanceInAutoScalingGroupCommand({
          InstanceId: state.targetInstance.InstanceId,
          ShouldDecrementDesiredCapacity: false,
        }),
      );
    },
  ),
  new ScenarioOutput("testKillInstance", MESSAGES.demoTestKillInstance),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput("failOpenConfirmation", MESSAGES.demoFailOpenConfirmation, {
    type: "confirm",
  }),
  new ScenarioAction("failOpenExit", (state) => {
    if (!state.failOpenConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("failOpen", () => {
    const client = new SSMClient({});
    return client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: `fake-table-${Date.now()}`,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testFailOpen", MESSAGES.demoFailOpenTest),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput(
    "resetTableConfirmation",
    MESSAGES.demoResetTableConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("resetTableExit", (state) => {
```

```
    if (!state.resetTableConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("resetTable", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testResetTable", MESSAGES.demoTestResetTable),
  healthCheckLoop,
  loadBalancerLoop,
];

async function createSsmOnlyInstanceProfile() {
  const iamClient = new IAMClient({});
  const { Policy } = await iamClient.send(
    new CreatePolicyCommand({
      PolicyName: NAMES.ssmOnlyPolicyName,
      PolicyDocument: readFileSync(
        join(RESOURCES_PATH, "ssm_only_policy.json"),
      ),
    }),
  );
  await iamClient.send(
    new CreateRoleCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      AssumeRolePolicyDocument: JSON.stringify({
        Version: "2012-10-17",
        Statement: [
          {
            Effect: "Allow",
            Principal: { Service: "ec2.amazonaws.com" },
            Action: "sts:AssumeRole",
          },
        ],
      }),
    }),
```

```
  );
  await iamClient.send(
    new AttachRolePolicyCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      PolicyArn: Policy.Arn,
    }),
  );
  await iamClient.send(
    new AttachRolePolicyCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
    }),
  );
  // snippet-start:[javascript.v3.wkflw.resilient.CreateInstanceProfile]
  const { InstanceProfile } = await iamClient.send(
    new CreateInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
    }),
  );
  await waitUntilInstanceProfileExists(
    { client: iamClient },
    { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
  );
  // snippet-end:[javascript.v3.wkflw.resilient.CreateInstanceProfile]
  await iamClient.send(
    new AddRoleToInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
      RoleName: NAMES.ssmOnlyRoleName,
    }),
  );

  return InstanceProfile;
}
```

모든 리소스를 폐기하는 단계를 생성합니다.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { unlinkSync } from "node:fs";

import { DynamoDBClient, DeleteTableCommand } from "@aws-sdk/client-dynamodb";
import {
```

```
  EC2Client,
  DeleteKeyPairCommand,
  DeleteLaunchTemplateCommand,
} from "@aws-sdk/client-ec2";
import {
  IAMClient,
  DeleteInstanceProfileCommand,
  RemoveRoleFromInstanceProfileCommand,
  DeletePolicyCommand,
  DeleteRoleCommand,
  DetachRolePolicyCommand,
  paginateListPolicies,
} from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DeleteAutoScalingGroupCommand,
  TerminateInstanceInAutoScalingGroupCommand,
  UpdateAutoScalingGroupCommand,
  paginateDescribeAutoScalingGroups,
} from "@aws-sdk/client-auto-scaling";
import {
  DeleteLoadBalancerCommand,
  DeleteTargetGroupCommand,
  DescribeTargetGroupsCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";

import {
  ScenarioOutput,
  ScenarioInput,
  ScenarioAction,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES } from "./constants.js";
import { findLoadBalancer } from "./shared.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const destroySteps = [
  new ScenarioInput("destroy", MESSAGES.destroy, { type: "confirm" }),
  new ScenarioAction(
    "abort",
```

```
    (state) => state.destroy === false && process.exit(),
  ),
  new ScenarioAction("deleteTable", async (c) => {
    try {
      const client = new DynamoDBClient({});
      await client.send(new DeleteTableCommand({ TableName: NAMES.tableName }));
    } catch (e) {
      c.deleteTableError = e;
    }
  }),
  new ScenarioOutput("deleteTableResult", (state) => {
    if (state.deleteTableError) {
      console.error(state.deleteTableError);
      return MESSAGES.deleteTableError.replace(
        "${TABLE_NAME}",
        NAMES.tableName,
      );
    } else {
      return MESSAGES.deletedTable.replace("${TABLE_NAME}", NAMES.tableName);
    }
  }),
  new ScenarioAction("deleteKeyPair", async (state) => {
    try {
      const client = new EC2Client({});
      await client.send(
        new DeleteKeyPairCommand({ KeyName: NAMES.keyPairName }),
      );
      unlinkSync(`${NAMES.keyPairName}.pem`);
    } catch (e) {
      state.deleteKeyPairError = e;
    }
  }),
  new ScenarioOutput("deleteKeyPairResult", (state) => {
    if (state.deleteKeyPairError) {
      console.error(state.deleteKeyPairError);
      return MESSAGES.deleteKeyPairError.replace(
        "${KEY_PAIR_NAME}",
        NAMES.keyPairName,
      );
    } else {
      return MESSAGES.deletedKeyPair.replace(
        "${KEY_PAIR_NAME}",
        NAMES.keyPairName,
      );
```

```
    }
  }),
  new ScenarioAction("detachPolicyFromRole", async (state) => {
    try {
      const client = new IAMClient({});
      const policy = await findPolicy(NAMES.instancePolicyName);

      if (!policy) {
        state.detachPolicyFromRoleError = new Error(
          `Policy ${NAMES.instancePolicyName} not found.`,
        );
      } else {
        await client.send(
          new DetachRolePolicyCommand({
            RoleName: NAMES.instanceRoleName,
            PolicyArn: policy.Arn,
          }),
        );
      }
    } catch (e) {
      state.detachPolicyFromRoleError = e;
    }
  }),
  new ScenarioOutput("detachedPolicyFromRole", (state) => {
    if (state.detachPolicyFromRoleError) {
      console.error(state.detachPolicyFromRoleError);
      return MESSAGES.detachPolicyFromRoleError
        .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    } else {
      return MESSAGES.detachedPolicyFromRole
        .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    }
  }),
  new ScenarioAction("deleteInstancePolicy", async (state) => {
    const client = new IAMClient({});
    const policy = await findPolicy(NAMES.instancePolicyName);

    if (!policy) {
      state.deletePolicyError = new Error(
        `Policy ${NAMES.instancePolicyName} not found.`,
      );
    } else {
```

```
      return client.send(
        new DeletePolicyCommand({
          PolicyArn: policy.Arn,
        }),
      );
    }
  }),
  new ScenarioOutput("deletePolicyResult", (state) => {
    if (state.deletePolicyError) {
      console.error(state.deletePolicyError);
      return MESSAGES.deletePolicyError.replace(
        "${INSTANCE_POLICY_NAME}",
        NAMES.instancePolicyName,
      );
    } else {
      return MESSAGES.deletedPolicy.replace(
        "${INSTANCE_POLICY_NAME}",
        NAMES.instancePolicyName,
      );
    }
  }),
  new ScenarioAction("removeRoleFromInstanceProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new RemoveRoleFromInstanceProfileCommand({
          RoleName: NAMES.instanceRoleName,
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
    } catch (e) {
      state.removeRoleFromInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("removeRoleFromInstanceProfileResult", (state) => {
    if (state.removeRoleFromInstanceProfile) {
      console.error(state.removeRoleFromInstanceProfileError);
      return MESSAGES.removeRoleFromInstanceProfileError
        .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    } else {
      return MESSAGES.removedRoleFromInstanceProfile
        .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
```

```
    }
  }),
  new ScenarioAction("deleteInstanceRole", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new DeleteRoleCommand({
          RoleName: NAMES.instanceRoleName,
        }),
      );
    } catch (e) {
      state.deleteInstanceRoleError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceRoleResult", (state) => {
    if (state.deleteInstanceRoleError) {
      console.error(state.deleteInstanceRoleError);
      return MESSAGES.deleteInstanceRoleError.replace(
        "${INSTANCE_ROLE_NAME}",
        NAMES.instanceRoleName,
      );
    } else {
      return MESSAGES.deletedInstanceRole.replace(
        "${INSTANCE_ROLE_NAME}",
        NAMES.instanceRoleName,
      );
    }
  }),
  new ScenarioAction("deleteInstanceProfile", async (state) => {
    try {
      // snippet-start:[javascript.v3.wkflw.resilient.DeleteInstanceProfile]
      const client = new IAMClient({});
      await client.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
      // snippet-end:[javascript.v3.wkflw.resilient.DeleteInstanceProfile]
    } catch (e) {
      state.deleteInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceProfileResult", (state) => {
    if (state.deleteInstanceProfileError) {
```

```
      console.error(state.deleteInstanceProfileError);
      return MESSAGES.deleteInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.instanceProfileName,
      );
    } else {
      return MESSAGES.deletedInstanceProfile.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.instanceProfileName,
      );
    }
  }),
  new ScenarioAction("deleteAutoScalingGroup", async (state) => {
    try {
      await terminateGroupInstances(NAMES.autoScalingGroupName);
      await retry({ intervalInMs: 60000, maxRetries: 60 }, async () => {
        await deleteAutoScalingGroup(NAMES.autoScalingGroupName);
      });
    } catch (e) {
      state.deleteAutoScalingGroupError = e;
    }
  }),
  new ScenarioOutput("deleteAutoScalingGroupResult", (state) => {
    if (state.deleteAutoScalingGroupError) {
      console.error(state.deleteAutoScalingGroupError);
      return MESSAGES.deleteAutoScalingGroupError.replace(
        "${AUTO_SCALING_GROUP_NAME}",
        NAMES.autoScalingGroupName,
      );
    } else {
      return MESSAGES.deletedAutoScalingGroup.replace(
        "${AUTO_SCALING_GROUP_NAME}",
        NAMES.autoScalingGroupName,
      );
    }
  }),
  new ScenarioAction("deleteLaunchTemplate", async (state) => {
    const client = new EC2Client({});
    try {
      // snippet-start:[javascript.v3.wkflw.resilient.DeleteLaunchTemplate]
      await client.send(
        new DeleteLaunchTemplateCommand({
          LaunchTemplateName: NAMES.launchTemplateName,
        }),
```

```
      );
      // snippet-end:[javascript.v3.wkflw.resilient.DeleteLaunchTemplate]
    } catch (e) {
      state.deleteLaunchTemplateError = e;
    }
  }),
  new ScenarioOutput("deleteLaunchTemplateResult", (state) => {
    if (state.deleteLaunchTemplateError) {
      console.error(state.deleteLaunchTemplateError);
      return MESSAGES.deleteLaunchTemplateError.replace(
        "${LAUNCH_TEMPLATE_NAME}",
        NAMES.launchTemplateName,
      );
    } else {
      return MESSAGES.deletedLaunchTemplate.replace(
        "${LAUNCH_TEMPLATE_NAME}",
        NAMES.launchTemplateName,
      );
    }
  }),
  new ScenarioAction("deleteLoadBalancer", async (state) => {
    try {
      // snippet-start:[javascript.v3.wkflw.resilient.DeleteLoadBalancer]
      const client = new ElasticLoadBalancingV2Client({});
      const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
      await client.send(
        new DeleteLoadBalancerCommand({
          LoadBalancerArn: loadBalancer.LoadBalancerArn,
        }),
      );
      await retry({ intervalInMs: 1000, maxRetries: 60 }, async () => {
        const lb = await findLoadBalancer(NAMES.loadBalancerName);
        if (lb) {
          throw new Error("Load balancer still exists.");
        }
      });
      // snippet-end:[javascript.v3.wkflw.resilient.DeleteLoadBalancer]
    } catch (e) {
      state.deleteLoadBalancerError = e;
    }
  }),
  new ScenarioOutput("deleteLoadBalancerResult", (state) => {
    if (state.deleteLoadBalancerError) {
      console.error(state.deleteLoadBalancerError);
```

```
      return MESSAGES.deleteLoadBalancerError.replace(
        "${LB_NAME}",
        NAMES.loadBalancerName,
      );
    } else {
      return MESSAGES.deletedLoadBalancer.replace(
        "${LB_NAME}",
        NAMES.loadBalancerName,
      );
    }
  }),
  new ScenarioAction("deleteLoadBalancerTargetGroup", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.DeleteTargetGroup]
    const client = new ElasticLoadBalancingV2Client({});
    try {
      const { TargetGroups } = await client.send(
        new DescribeTargetGroupsCommand({
          Names: [NAMES.loadBalancerTargetGroupName],
        }),
      );

      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
        client.send(
          new DeleteTargetGroupCommand({
            TargetGroupArn: TargetGroups[0].TargetGroupArn,
          }),
        ),
      );
    } catch (e) {
      state.deleteLoadBalancerTargetGroupError = e;
    }
    // snippet-end:[javascript.v3.wkflw.resilient.DeleteTargetGroup]
  }),
  new ScenarioOutput("deleteLoadBalancerTargetGroupResult", (state) => {
    if (state.deleteLoadBalancerTargetGroupError) {
      console.error(state.deleteLoadBalancerTargetGroupError);
      return MESSAGES.deleteLoadBalancerTargetGroupError.replace(
        "${TARGET_GROUP_NAME}",
        NAMES.loadBalancerTargetGroupName,
      );
    } else {
      return MESSAGES.deletedLoadBalancerTargetGroup.replace(
        "${TARGET_GROUP_NAME}",
        NAMES.loadBalancerTargetGroupName,
```

```
      );
    }
  }),
  new ScenarioAction("detachSsmOnlyRoleFromProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new RemoveRoleFromInstanceProfileCommand({
          InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
          RoleName: NAMES.ssmOnlyRoleName,
        }),
      );
    } catch (e) {
      state.detachSsmOnlyRoleFromProfileError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyRoleFromProfileResult", (state) => {
    if (state.detachSsmOnlyRoleFromProfileError) {
      console.error(state.detachSsmOnlyRoleFromProfileError);
      return MESSAGES.detachSsmOnlyRoleFromProfileError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
    } else {
      return MESSAGES.detachedSsmOnlyRoleFromProfile
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
    }
  }),
  new ScenarioAction("detachSsmOnlyCustomRolePolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
      await iamClient.send(
        new DetachRolePolicyCommand({
          RoleName: NAMES.ssmOnlyRoleName,
          PolicyArn: ssmOnlyPolicy.Arn,
        }),
      );
    } catch (e) {
      state.detachSsmOnlyCustomRolePolicyError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyCustomRolePolicyResult", (state) => {
    if (state.detachSsmOnlyCustomRolePolicyError) {
```

```
      console.error(state.detachSsmOnlyCustomRolePolicyError);
      return MESSAGES.detachSsmOnlyCustomRolePolicyError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
    } else {
      return MESSAGES.detachedSsmOnlyCustomRolePolicy
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
    }
  }),
  new ScenarioAction("detachSsmOnlyAWSRolePolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DetachRolePolicyCommand({
          RoleName: NAMES.ssmOnlyRoleName,
          PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
        }),
      );
    } catch (e) {
      state.detachSsmOnlyAWSRolePolicyError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyAWSRolePolicyResult", (state) => {
    if (state.detachSsmOnlyAWSRolePolicyError) {
      console.error(state.detachSsmOnlyAWSRolePolicyError);
      return MESSAGES.detachSsmOnlyAWSRolePolicyError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
    } else {
      return MESSAGES.detachedSsmOnlyAWSRolePolicy
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
    }
  }),
  new ScenarioAction("deleteSsmOnlyInstanceProfile", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
        }),
      );
    } catch (e) {
```

```
      state.deleteSsmOnlyInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyInstanceProfileResult", (state) => {
    if (state.deleteSsmOnlyInstanceProfileError) {
      console.error(state.deleteSsmOnlyInstanceProfileError);
      return MESSAGES.deleteSsmOnlyInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.ssmOnlyInstanceProfileName,
      );
    } else {
      return MESSAGES.deletedSsmOnlyInstanceProfile.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.ssmOnlyInstanceProfileName,
      );
    }
  }),
  new ScenarioAction("deleteSsmOnlyPolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
      await iamClient.send(
        new DeletePolicyCommand({
          PolicyArn: ssmOnlyPolicy.Arn,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyPolicyError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyPolicyResult", (state) => {
    if (state.deleteSsmOnlyPolicyError) {
      console.error(state.deleteSsmOnlyPolicyError);
      return MESSAGES.deleteSsmOnlyPolicyError.replace(
        "${POLICY_NAME}",
        NAMES.ssmOnlyPolicyName,
      );
    } else {
      return MESSAGES.deletedSsmOnlyPolicy.replace(
        "${POLICY_NAME}",
        NAMES.ssmOnlyPolicyName,
      );
    }
  }),
```

```
  new ScenarioAction("deleteSsmOnlyRole", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteRoleCommand({
          RoleName: NAMES.ssmOnlyRoleName,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyRoleError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyRoleResult", (state) => {
    if (state.deleteSsmOnlyRoleError) {
      console.error(state.deleteSsmOnlyRoleError);
      return MESSAGES.deleteSsmOnlyRoleError.replace(
        "${ROLE_NAME}",
        NAMES.ssmOnlyRoleName,
      );
    } else {
      return MESSAGES.deletedSsmOnlyRole.replace(
        "${ROLE_NAME}",
        NAMES.ssmOnlyRoleName,
      );
    }
  }),
];

/**
 * @param {string} policyName
 */
async function findPolicy(policyName) {
  const client = new IAMClient({});
  const paginatedPolicies = paginateListPolicies({ client }, {});
  for await (const page of paginatedPolicies) {
    const policy = page.Policies.find((p) => p.PolicyName === policyName);
    if (policy) {
      return policy;
    }
  }
}

/**
 * @param {string} groupName
```

```
 */
async function deleteAutoScalingGroup(groupName) {
  const client = new AutoScalingClient({});
  try {
    await client.send(
      new DeleteAutoScalingGroupCommand({
        AutoScalingGroupName: groupName,
      }),
    );
  } catch (err) {
    if (!(err instanceof Error)) {
      throw err;
    } else {
      console.log(err.name);
      throw err;
    }
  }
}

/**
 * @param {string} groupName
 */
async function terminateGroupInstances(groupName) {
  const autoScalingClient = new AutoScalingClient({});
  const group = await findAutoScalingGroup(groupName);
  await autoScalingClient.send(
    new UpdateAutoScalingGroupCommand({
      AutoScalingGroupName: group.AutoScalingGroupName,
      MinSize: 0,
    }),
  );
  for (const i of group.Instances) {
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new TerminateInstanceInAutoScalingGroupCommand({
          InstanceId: i.InstanceId,
          ShouldDecrementDesiredCapacity: true,
        }),
      ),
    );
  }
}

async function findAutoScalingGroup(groupName) {
```

```
  const client = new AutoScalingClient({});
  const paginatedGroups = paginateDescribeAutoScalingGroups({ client }, {});
  for await (const page of paginatedGroups) {
    const group = page.AutoScalingGroups.find(
      (g) => g.AutoScalingGroupName === groupName,
    );
    if (group) {
      return group;
    }
  }
  throw new Error(`Auto scaling group ${groupName} not found.`);
}
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 다음 주제를 참조하십시오.
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate
  - DeleteLoadBalancer
  - DeleteTargetGroup
  - DescribeAutoScalingGroups
  - DescribeAvailabilityZones
  - DescribeIamInstanceProfileAssociations
  - DescribeInstances
  - DescribeLoadBalancers
  - DescribeSubnets
  - DescribeTargetGroups
  - DescribeTargetHealth

- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Python

SDK for Python(Boto3)

> **Note**
>
> 더 많은 것이 있어요 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

명령 프롬프트에서 대화형 시나리오를 실행합니다.

```
class Runner:
    def __init__(
        self, resource_path, recommendation, autoscaler, loadbalancer,
 param_helper
    ):
        self.resource_path = resource_path
        self.recommendation = recommendation
        self.autoscaler = autoscaler
        self.loadbalancer = loadbalancer
        self.param_helper = param_helper
        self.protocol = "HTTP"
        self.port = 80
        self.ssh_port = 22

    def deploy(self):
        recommendations_path = f"{self.resource_path}/recommendations.json"
        startup_script = f"{self.resource_path}/server_startup_script.sh"
        instance_policy = f"{self.resource_path}/instance_policy.json"

        print(
            "\nFor this demo, we'll use the AWS SDK for Python (Boto3) to create
 several AWS resources\n"
```

```
            "to set up a load-balanced web service endpoint and explore some ways
to make it resilient\n"
            "against various kinds of failures.\n\n"
            "Some of the resources create by this demo are:\n"
        )
        print(
            "\t* A DynamoDB table that the web service depends on to provide
book, movie, and song recommendations."
        )
        print(
            "\t* An EC2 launch template that defines EC2 instances that each
contain a Python web server."
        )
        print(
            "\t* An EC2 Auto Scaling group that manages EC2 instances across
several Availability Zones."
        )
        print(
            "\t* An Elastic Load Balancing (ELB) load balancer that targets the
Auto Scaling group to distribute requests."
        )
        print("-" * 88)
        q.ask("Press Enter when you're ready to start deploying resources.")

        print(
            f"Creating and populating a DynamoDB table named
'{self.recommendation.table_name}'."
        )
        self.recommendation.create()
        self.recommendation.populate(recommendations_path)
        print("-" * 88)

        print(
            f"Creating an EC2 launch template that runs '{startup_script}' when
an instance starts.\n"
            f"This script starts a Python web server defined in the `server.py`
script. The web server\n"
            f"listens to HTTP requests on port 80 and responds to requests to '/'
and to '/healthcheck'.\n"
            f"For demo purposes, this server is run as the root user. In
production, the best practice is to\n"
            f"run a web server, such as Apache, with least-privileged
credentials.\n"
        )
```

```
        print(
            f"The template also defines an IAM policy that each instance uses to
assume a role that grants\n"
            f"permissions to access the DynamoDB recommendation table and Systems
Manager parameters\n"
            f"that control the flow of the demo.\n"
        )
        self.autoscaler.create_template(startup_script, instance_policy)
        print("-" * 88)

        print(
            f"Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different\n"
            f"Availability Zone."
        )
        zones = self.autoscaler.create_group(3)
        print("-" * 88)
        print(
            "At this point, you have EC2 instances created. Once each instance
starts, it listens for\n"
            "HTTP requests. You can see these instances in the console or
continue with the demo."
        )
        print("-" * 88)
        q.ask("Press Enter when you're ready to continue.")

        print(f"Creating variables that control the flow of the demo.\n")
        self.param_helper.reset()

        print(
            "\nCreating an Elastic Load Balancing target group and load balancer.
The target group\n"
            "defines how the load balancer connects to instances. The load
balancer provides a\n"
            "single endpoint where clients connect and dispatches requests to
instances in the group.\n"
        )
        vpc = self.autoscaler.get_default_vpc()
        subnets = self.autoscaler.get_subnets(vpc["VpcId"], zones)
        target_group = self.loadbalancer.create_target_group(
            self.protocol, self.port, vpc["VpcId"]
        )
        self.loadbalancer.create_load_balancer(
            [subnet["SubnetId"] for subnet in subnets], target_group
```

```
        )
        self.autoscaler.attach_load_balancer_target_group(target_group)
        print(f"Verifying access to the load balancer endpoint...")
        lb_success = self.loadbalancer.verify_load_balancer_endpoint()
        if not lb_success:
            print(
                "Couldn't connect to the load balancer, verifying that the port
 is open..."
            )
            current_ip_address = requests.get(
                "http://checkip.amazonaws.com"
            ).text.strip()
            sec_group, port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.port, current_ip_address
            )
            sec_group, ssh_port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.ssh_port, current_ip_address
            )
            if not port_is_open:
                print(
                    "For this example to work, the default security group for
 your default VPC must\n"
                    "allows access from this computer. You can either add it
 automatically from this\n"
                    "example or add it yourself using the AWS Management Console.
\n"
                )
                if q.ask(
                    f"Do you want to add a rule to security group
 {sec_group['GroupId']} to allow\n"
                    f"inbound traffic on port {self.port} from your computer's IP
 address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.port, current_ip_address
                    )
            if not ssh_port_is_open:
                if q.ask(
                    f"Do you want to add a rule to security group
 {sec_group['GroupId']} to allow\n"
                    f"inbound SSH traffic on port {self.ssh_port} for debugging
 from your computer's IP address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
```

```
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.ssh_port, current_ip_address
                    )
            lb_success = self.loadbalancer.verify_load_balancer_endpoint()
        if lb_success:
            print("Your load balancer is ready. You can access it by browsing to:
\n")
            print(f"\thttp://{self.loadbalancer.endpoint()}\n")
        else:
            print(
                "Couldn't get a successful response from the load balancer
 endpoint. Troubleshoot by\n"
                "manually verifying that your VPC and security group are
 configured correctly and that\n"
                "you can successfully make a GET request to the load balancer
 endpoint:\n"
            )
            print(f"\thttp://{self.loadbalancer.endpoint()}\n")
        print("-" * 88)
        q.ask("Press Enter when you're ready to continue with the demo.")

    def demo_choices(self):
        actions = [
            "Send a GET request to the load balancer endpoint.",
            "Check the health of load balancer targets.",
            "Go to the next part of the demo.",
        ]
        choice = 0
        while choice != 2:
            print("-" * 88)
            print(
                "\nSee the current state of the service by selecting one of the
 following choices:\n"
            )
            choice = q.choose("\nWhich action would you like to take? ", actions)
            print("-" * 88)
            if choice == 0:
                print("Request:\n")
                print(f"GET http://{self.loadbalancer.endpoint()}")
                response = requests.get(f"http://{self.loadbalancer.endpoint()}")
                print("\nResponse:\n")
                print(f"{response.status_code}")
                if response.headers.get("content-type") == "application/json":
```

```
                    pp(response.json())
            elif choice == 1:
                print("\nChecking the health of load balancer targets:\n")
                health = self.loadbalancer.check_target_health()
                for target in health:
                    state = target["TargetHealth"]["State"]
                    print(
                        f"\tTarget {target['Target']['Id']} on port
{target['Target']['Port']} is {state}"
                    )
                    if state != "healthy":
                        print(
                            f"\t\t{target['TargetHealth']['Reason']}:
{target['TargetHealth']['Description']}\n"
                        )
                print(
                    f"\nNote that it can take a minute or two for the health
check to update\n"
                    f"after changes are made.\n"
                )
            elif choice == 2:
                print("\nOkay, let's move on.")
                print("-" * 88)

    def demo(self):
        ssm_only_policy = f"{self.resource_path}/ssm_only_policy.json"

        print("\nResetting parameters to starting values for demo.\n")
        self.param_helper.reset()

        print(
            "\nThis part of the demonstration shows how to toggle different parts
of the system\n"
            "to create situations where the web service fails, and shows how
using a resilient\n"
            "architecture can keep the web service running in spite of these
failures."
        )
        print("-" * 88)

        print(
            "At the start, the load balancer endpoint returns recommendations and
reports that all targets are healthy."
        )
```

```
        self.demo_choices()

        print(
            f"The web service running on the EC2 instances gets recommendations
by querying a DynamoDB table.\n"
            f"The table name is contained in a Systems Manager parameter named
'{self.param_helper.table}'.\n"
            f"To simulate a failure of the recommendation service, let's set this
parameter to name a non-existent table.\n"
        )
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        print(
            "\nNow, sending a GET request to the load balancer endpoint returns a
failure code. But, the service reports as\n"
            "healthy to the load balancer because shallow health checks don't
check for failure of the recommendation service."
        )
        self.demo_choices()

        print(
            f"Instead of failing when the recommendation service fails, the web
service can return a static response.\n"
            f"While this is not a perfect solution, it presents the customer with
a somewhat better experience than failure.\n"
        )
        self.param_helper.put(self.param_helper.failure_response, "static")
        print(
            f"\nNow, sending a GET request to the load balancer endpoint returns
a static response.\n"
            f"The service still reports as healthy because health checks are
still shallow.\n"
        )
        self.demo_choices()

        print("Let's reinstate the recommendation service.\n")
        self.param_helper.put(self.param_helper.table,
self.recommendation.table_name)
        print(
            "\nLet's also substitute bad credentials for one of the instances in
the target group so that it can't\n"
            "access the DynamoDB recommendation table.\n"
        )
        self.autoscaler.create_instance_profile(
            ssm_only_policy,
```

```
            self.autoscaler.bad_creds_policy_name,
            self.autoscaler.bad_creds_role_name,
            self.autoscaler.bad_creds_profile_name,
            ["AmazonSSMManagedInstanceCore"],
        )
        instances = self.autoscaler.get_instances()
        bad_instance_id = instances[0]
        instance_profile = self.autoscaler.get_instance_profile(bad_instance_id)
        print(
            f"\nReplacing the profile for instance {bad_instance_id} with a
profile that contains\n"
            f"bad credentials...\n"
        )
        self.autoscaler.replace_instance_profile(
            bad_instance_id,
            self.autoscaler.bad_creds_profile_name,
            instance_profile["AssociationId"],
        )
        print(
            "Now, sending a GET request to the load balancer endpoint returns
either a recommendation or a static response,\n"
            "depending on which instance is selected by the load balancer.\n"
        )
        self.demo_choices()

        print(
            "\nLet's implement a deep health check. For this demo, a deep health
check tests whether\n"
            "the web service can access the DynamoDB table that it depends on for
recommendations. Note that\n"
            "the deep health check is only for ELB routing and not for Auto
Scaling instance health.\n"
            "This kind of deep health check is not recommended for Auto Scaling
instance health, because it\n"
            "risks accidental termination of all instances in the Auto Scaling
group when a dependent service fails.\n"
        )
        print(
            "By implementing deep health checks, the load balancer can detect
when one of the instances is failing\n"
            "and take that instance out of rotation.\n"
        )
        self.param_helper.put(self.param_helper.health_check, "deep")
        print(
```

```
            f"\nNow, checking target health indicates that the instance with bad
credentials ({bad_instance_id})\n"
            f"is unhealthy. Note that it might take a minute or two for the load
balancer to detect the unhealthy \n"
            f"instance. Sending a GET request to the load balancer endpoint
always returns a recommendation, because\n"
            "the load balancer takes unhealthy instances out of its rotation.\n"
        )
        self.demo_choices()

        print(
            "\nBecause the instances in this demo are controlled by an auto
scaler, the simplest way to fix an unhealthy\n"
            "instance is to terminate it and let the auto scaler start a new
instance to replace it.\n"
        )
        self.autoscaler.terminate_instance(bad_instance_id)
        print(
            "\nEven while the instance is terminating and the new instance is
starting, sending a GET\n"
            "request to the web service continues to get a successful
recommendation response because\n"
            "the load balancer routes requests to the healthy instances. After
the replacement instance\n"
            "starts and reports as healthy, it is included in the load balancing
rotation.\n"
            "\nNote that terminating and replacing an instance typically takes
several minutes, during which time you\n"
            "can see the changing health check status until the new instance is
running and healthy.\n"
        )
        self.demo_choices()

        print(
            "\nIf the recommendation service fails now, deep health checks mean
all instances report as unhealthy.\n"
        )
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        print(
            "\nWhen all instances are unhealthy, the load balancer continues to
route requests even to\n"
            "unhealthy instances, allowing them to fail open and return a static
response rather than fail\n"
            "closed and report failure to the customer."
```

```
        )
        self.demo_choices()
        self.param_helper.reset()

    def destroy(self):
        print(
            "This concludes the demo of how to build and manage a resilient
 service.\n"
            "To keep things tidy and to avoid unwanted charges on your account,
 we can clean up all AWS resources\n"
            "that were created for this demo."
        )
        if q.ask("Do you want to clean up all demo resources? (y/n) ",
 q.is_yesno):
            self.loadbalancer.delete_load_balancer()
            self.loadbalancer.delete_target_group()
            self.autoscaler.delete_group()
            self.autoscaler.delete_key_pair()
            self.autoscaler.delete_template()
            self.autoscaler.delete_instance_profile(
                self.autoscaler.bad_creds_profile_name,
                self.autoscaler.bad_creds_role_name,
            )
            self.recommendation.destroy()
        else:
            print(
                "Okay, we'll leave the resources intact.\n"
                "Don't forget to delete them when you're done with them or you
 might incur unexpected charges."
            )


def main():
    parser = argparse.ArgumentParser()
    parser.add_argument(
        "--action",
        required=True,
        choices=["all", "deploy", "demo", "destroy"],
        help="The action to take for the demo. When 'all' is specified, resources
 are\n"
        "deployed, the demo is run, and resources are destroyed.",
    )
    parser.add_argument(
        "--resource_path",
```

```
        default="../../../workflows/resilient_service/resources",
        help="The path to resource files used by this example, such as IAM
 policies and\n"
        "instance scripts.",
    )
    args = parser.parse_args()

    print("-" * 88)
    print(
        "Welcome to the demonstration of How to Build and Manage a Resilient
 Service!"
    )
    print("-" * 88)

    prefix = "doc-example-resilience"
    recommendation = RecommendationService.from_client(
        "doc-example-recommendation-service"
    )
    autoscaler = AutoScaler.from_client(prefix)
    loadbalancer = LoadBalancer.from_client(prefix)
    param_helper = ParameterHelper.from_client(recommendation.table_name)
    runner = Runner(
        args.resource_path, recommendation, autoscaler, loadbalancer,
 param_helper
    )
    actions = [args.action] if args.action != "all" else ["deploy", "demo",
 "destroy"]
    for action in actions:
        if action == "deploy":
            runner.deploy()
        elif action == "demo":
            runner.demo()
        elif action == "destroy":
            runner.destroy()

    print("-" * 88)
    print("Thanks for watching!")
    print("-" * 88)


if __name__ == "__main__":
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    main()
```

Auto Scaling과 Amazon EC2 작업을 래핑하는 클래스를 생성합니다.

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
```

```
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    @classmethod
    def from_client(cls, resource_prefix):
        """
        Creates this class from Boto3 clients.

        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        """
        as_client = boto3.client("autoscaling")
        ec2_client = boto3.client("ec2")
        ssm_client = boto3.client("ssm")
        iam_client = boto3.client("iam")
        return cls(
            resource_prefix,
            "t3.micro",
            "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
            as_client,
            ec2_client,
            ssm_client,
            iam_client,
        )

    def create_instance_profile(
        self, policy_file, policy_name, role_name, profile_name,
aws_managed_policies=()
    ):
        """
        Creates a policy, role, and profile that is associated with instances
created by
        this class. An instance's associated profile defines a role that is
assumed by the
        instance. The role has attached policies that specify the AWS permissions
granted to
        clients that run on the instance.

        :param policy_file: The name of a JSON file that contains the policy
definition to
                            create and attach to the role.
        :param policy_name: The name to give the created policy.
```

```
        :param role_name: The name to give the created role.
        :param profile_name: The name to the created profile.
        :param aws_managed_policies: Additional AWS-managed policies that are
attached to
                                     the role, such as
AmazonSSMManagedInstanceCore to grant
                                     use of Systems Manager to send commands to
the instance.
        :return: The ARN of the profile that is created.
        """
        assume_role_doc = {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {"Service": "ec2.amazonaws.com"},
                    "Action": "sts:AssumeRole",
                }
            ],
        }
        with open(policy_file) as file:
            instance_policy_doc = file.read()

        policy_arn = None
        try:
            pol_response = self.iam_client.create_policy(
                PolicyName=policy_name, PolicyDocument=instance_policy_doc
            )
            policy_arn = pol_response["Policy"]["Arn"]
            log.info("Created policy with ARN %s.", policy_arn)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                log.info("Policy %s already exists, nothing to do.", policy_name)
                list_pol_response = self.iam_client.list_policies(Scope="Local")
                for pol in list_pol_response["Policies"]:
                    if pol["PolicyName"] == policy_name:
                        policy_arn = pol["Arn"]
                        break
            if policy_arn is None:
                raise AutoScalerError(f"Couldn't create policy {policy_name}:
{err}")

        try:
            self.iam_client.create_role(
```

```
                RoleName=role_name,
AssumeRolePolicyDocument=json.dumps(assume_role_doc)
            )
            self.iam_client.attach_role_policy(RoleName=role_name,
PolicyArn=policy_arn)
            for aws_policy in aws_managed_policies:
                self.iam_client.attach_role_policy(
                    RoleName=role_name,
                    PolicyArn=f"arn:aws:iam::aws:policy/{aws_policy}",
                )
            log.info("Created role %s and attached policy %s.", role_name,
policy_arn)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                log.info("Role %s already exists, nothing to do.", role_name)
            else:
                raise AutoScalerError(f"Couldn't create role {role_name}: {err}")

        try:
            profile_response = self.iam_client.create_instance_profile(
                InstanceProfileName=profile_name
            )
            waiter = self.iam_client.get_waiter("instance_profile_exists")
            waiter.wait(InstanceProfileName=profile_name)
            time.sleep(10)  # wait a little longer
            profile_arn = profile_response["InstanceProfile"]["Arn"]
            self.iam_client.add_role_to_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )
            log.info("Created profile %s and added role %s.", profile_name,
role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                prof_response = self.iam_client.get_instance_profile(
                    InstanceProfileName=profile_name
                )
                profile_arn = prof_response["InstanceProfile"]["Arn"]
                log.info(
                    "Instance profile %s already exists, nothing to do.",
profile_name
                )
            else:
                raise AutoScalerError(
```

```
                    f"Couldn't create profile {profile_name} and attach it to
role\n"
                    f"{role_name}: {err}"
                )
        return profile_arn


    def get_instance_profile(self, instance_id):
        """
        Gets data about the profile associated with an instance.

        :param instance_id: The ID of the instance to look up.
        :return: The profile data.
        """
        try:
            response =
self.ec2_client.describe_iam_instance_profile_associations(
                Filters=[{"Name": "instance-id", "Values": [instance_id]}]
            )
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't get instance profile association for instance
{instance_id}: {err}"
            )
        else:
            return response["IamInstanceProfileAssociations"][0]


    def replace_instance_profile(
        self, instance_id, new_instance_profile_name, profile_association_id
    ):
        """
        Replaces the profile associated with a running instance. After the
profile is
        replaced, the instance is rebooted to ensure that it uses the new
profile. When
        the instance is ready, Systems Manager is used to restart the Python web
server.

        :param instance_id: The ID of the instance to update.
        :param new_instance_profile_name: The name of the new profile to
associate with
                                          the specified instance.
```

```
        :param profile_association_id: The ID of the existing profile association
for the
                                       instance.
        """
        try:
            self.ec2_client.replace_iam_instance_profile_association(
                IamInstanceProfile={"Name": new_instance_profile_name},
                AssociationId=profile_association_id,
            )
            log.info(
                "Replaced instance profile for association %s with profile %s.",
                profile_association_id,
                new_instance_profile_name,
            )
            time.sleep(5)
            inst_ready = False
            tries = 0
            while not inst_ready:
                if tries % 6 == 0:
                    self.ec2_client.reboot_instances(InstanceIds=[instance_id])
                    log.info(
                        "Rebooting instance %s and waiting for it to to be
ready.",
                        instance_id,
                    )
                tries += 1
                time.sleep(10)
                response = self.ssm_client.describe_instance_information()
                for info in response["InstanceInformationList"]:
                    if info["InstanceId"] == instance_id:
                        inst_ready = True
            self.ssm_client.send_command(
                InstanceIds=[instance_id],
                DocumentName="AWS-RunShellScript",
                Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
            )
            log.info("Restarted the Python web server on instance %s.",
instance_id)
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't replace instance profile for association
{profile_association_id}: {err}"
            )
```

```
    def delete_instance_profile(self, profile_name, role_name):
        """
        Detaches a role from an instance profile, detaches policies from the
role,
        and deletes all the resources.

        :param profile_name: The name of the profile to delete.
        :param role_name: The name of the role to delete.
        """
        try:
            self.iam_client.remove_role_from_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )

self.iam_client.delete_instance_profile(InstanceProfileName=profile_name)
            log.info("Deleted instance profile %s.", profile_name)
            attached_policies = self.iam_client.list_attached_role_policies(
                RoleName=role_name
            )
            for pol in attached_policies["AttachedPolicies"]:
                self.iam_client.detach_role_policy(
                    RoleName=role_name, PolicyArn=pol["PolicyArn"]
                )
                if not pol["PolicyArn"].startswith("arn:aws:iam::aws"):
                    self.iam_client.delete_policy(PolicyArn=pol["PolicyArn"])
                log.info("Detached and deleted policy %s.", pol["PolicyName"])
            self.iam_client.delete_role(RoleName=role_name)
            log.info("Deleted role %s.", role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "NoSuchEntity":
                log.info(
                    "Instance profile %s doesn't exist, nothing to do.",
profile_name
                )
            else:
                raise AutoScalerError(
                    f"Couldn't delete instance profile {profile_name} or detach "
                    f"policies and delete role {role_name}: {err}"
                )


    def create_key_pair(self, key_pair_name):
        """
```

```
        Creates a new key pair.

        :param key_pair_name: The name of the key pair to create.
        :return: The newly created key pair.
        """
        try:
            response = self.ec2_client.create_key_pair(KeyName=key_pair_name)
            with open(f"{key_pair_name}.pem", "w") as file:
                file.write(response["KeyMaterial"])
            chmod(f"{key_pair_name}.pem", 0o600)
            log.info("Created key pair %s.", key_pair_name)
        except ClientError as err:
            raise AutoScalerError(f"Couldn't create key pair {key_pair_name}:
{err}")


    def delete_key_pair(self):
        """
        Deletes a key pair.

        :param key_pair_name: The name of the key pair to delete.
        """
        try:
            self.ec2_client.delete_key_pair(KeyName=self.key_pair_name)
            remove(f"{self.key_pair_name}.pem")
            log.info("Deleted key pair %s.", self.key_pair_name)
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't delete key pair {self.key_pair_name}: {err}"
            )
        except FileNotFoundError:
            log.info("Key pair %s doesn't exist, nothing to do.",
self.key_pair_name)
        except PermissionError:
            log.info(
                "Inadequate permissions to delete key pair %s.",
self.key_pair_name
            )
        except Exception as err:
            raise AutoScalerError(
                f"Couldn't delete key pair {self.key_pair_name}: {err}"
            )
```

```
    def create_template(self, server_startup_script_file, instance_policy_file):
        """
        Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling. The
        launch template specifies a Bash script in its user data field that runs
after
        the instance is started. This script installs Python packages and starts
a
        Python web server on the instance.

        :param server_startup_script_file: The path to a Bash script file that is
run
                                           when an instance starts.
        :param instance_policy_file: The path to a file that defines a
permissions policy
                                     to create and attach to the instance
profile.
        :return: Information about the newly created template.
        """
        template = {}
        try:
            self.create_key_pair(self.key_pair_name)
            self.create_instance_profile(
                instance_policy_file,
                self.instance_policy_name,
                self.instance_role_name,
                self.instance_profile_name,
            )
            with open(server_startup_script_file) as file:
                start_server_script = file.read()
            ami_latest = self.ssm_client.get_parameter(Name=self.ami_param)
            ami_id = ami_latest["Parameter"]["Value"]
            lt_response = self.ec2_client.create_launch_template(
                LaunchTemplateName=self.launch_template_name,
                LaunchTemplateData={
                    "InstanceType": self.inst_type,
                    "ImageId": ami_id,
                    "IamInstanceProfile": {"Name": self.instance_profile_name},
                    "UserData": base64.b64encode(
                        start_server_script.encode(encoding="utf-8")
                    ).decode(encoding="utf-8"),
                    "KeyName": self.key_pair_name,
                },
            )
```

```
            template = lt_response["LaunchTemplate"]
            log.info(
                "Created launch template %s for AMI %s on %s.",
                self.launch_template_name,
                ami_id,
                self.inst_type,
            )
        except ClientError as err:
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.AlreadyExistsException"
            ):
                log.info(
                    "Launch template %s already exists, nothing to do.",
                    self.launch_template_name,
                )
            else:
                raise AutoScalerError(
                    f"Couldn't create launch template
{self.launch_template_name}: {err}."
                )
        return template


    def delete_template(self):
        """
        Deletes a launch template.
        """
        try:
            self.ec2_client.delete_launch_template(
                LaunchTemplateName=self.launch_template_name
            )
            self.delete_instance_profile(
                self.instance_profile_name, self.instance_role_name
            )
            log.info("Launch template %s deleted.", self.launch_template_name)
        except ClientError as err:
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.NotFoundException"
            ):
                log.info(
                    "Launch template %s does not exist, nothing to do.",
                    self.launch_template_name,
```

```
                )
            else:
                raise AutoScalerError(
                    f"Couldn't delete launch template
{self.launch_template_name}: {err}."
                )


    def get_availability_zones(self):
        """
        Gets a list of Availability Zones in the AWS Region of the Amazon EC2
client.

        :return: The list of Availability Zones for the client Region.
        """
        try:
            response = self.ec2_client.describe_availability_zones()
            zones = [zone["ZoneName"] for zone in response["AvailabilityZones"]]
        except ClientError as err:
            raise AutoScalerError(f"Couldn't get availability zones: {err}.")
        else:
            return zones


    def create_group(self, group_size):
        """
        Creates an EC2 Auto Scaling group with the specified size.

        :param group_size: The number of instances to set for the minimum and
maximum in
                           the group.
        :return: The list of Availability Zones specified for the group.
        """
        zones = []
        try:
            zones = self.get_availability_zones()
            self.autoscaling_client.create_auto_scaling_group(
                AutoScalingGroupName=self.group_name,
                AvailabilityZones=zones,
                LaunchTemplate={
                    "LaunchTemplateName": self.launch_template_name,
                    "Version": "$Default",
                },
                MinSize=group_size,
```

```
                MaxSize=group_size,
            )
            log.info(
                "Created EC2 Auto Scaling group %s with availability zones %s.",
                self.launch_template_name,
                zones,
            )
        except ClientError as err:
            if err.response["Error"]["Code"] == "AlreadyExists":
                log.info(
                    "EC2 Auto Scaling group %s already exists, nothing to do.",
                    self.group_name,
                )
            else:
                raise AutoScalerError(
                    f"Couldn't create EC2 Auto Scaling group {self.group_name}:
{err}"
                )
        return zones


    def get_instances(self):
        """
        Gets data about the instances in the EC2 Auto Scaling group.

        :return: Data about the instances.
        """
        try:
            as_response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[self.group_name]
            )
            instance_ids = [
                i["InstanceId"]
                for i in as_response["AutoScalingGroups"][0]["Instances"]
            ]
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't get instances for Auto Scaling group
{self.group_name}: {err}"
            )
        else:
            return instance_ids
```

```
    def terminate_instance(self, instance_id):
        """
        Terminates and instances in an EC2 Auto Scaling group. After an instance
is
        terminated, it can no longer be accessed.

        :param instance_id: The ID of the instance to terminate.
        """
        try:
            self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                InstanceId=instance_id, ShouldDecrementDesiredCapacity=False
            )
            log.info("Terminated instance %s.", instance_id)
        except ClientError as err:
            raise AutoScalerError(f"Couldn't terminate instance {instance_id}:
{err}")

    def attach_load_balancer_target_group(self, lb_target_group):
        """
        Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
Scaling group.
        The target group specifies how the load balancer forward requests to the
instances
        in the group.

        :param lb_target_group: Data about the ELB target group to attach.
        """
        try:
            self.autoscaling_client.attach_load_balancer_target_groups(
                AutoScalingGroupName=self.group_name,
                TargetGroupARNs=[lb_target_group["TargetGroupArn"]],
            )
            log.info(
                "Attached load balancer target group %s to auto scaling group
%s.",
                lb_target_group["TargetGroupName"],
                self.group_name,
            )
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't attach load balancer target group
{lb_target_group['TargetGroupName']}\n"
                f"to auto scaling group {self.group_name}"
            )
```

```
    def _try_terminate_instance(self, inst_id):
        stopping = False
        log.info(f"Stopping {inst_id}.")
        while not stopping:
            try:
                self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                    InstanceId=inst_id, ShouldDecrementDesiredCapacity=True
                )
                stopping = True
            except ClientError as err:
                if err.response["Error"]["Code"] == "ScalingActivityInProgress":
                    log.info("Scaling activity in progress for %s. Waiting...",
inst_id)
                    time.sleep(10)
                else:
                    raise AutoScalerError(f"Couldn't stop instance {inst_id}:
{err}.")

    def _try_delete_group(self):
        """
        Tries to delete the EC2 Auto Scaling group. If the group is in use or in
progress,
        the function waits and retries until the group is successfully deleted.
        """
        stopped = False
        while not stopped:
            try:
                self.autoscaling_client.delete_auto_scaling_group(
                    AutoScalingGroupName=self.group_name
                )
                stopped = True
                log.info("Deleted EC2 Auto Scaling group %s.", self.group_name)
            except ClientError as err:
                if (
                    err.response["Error"]["Code"] == "ResourceInUse"
                    or err.response["Error"]["Code"] ==
"ScalingActivityInProgress"
                ):
                    log.info(
                        "Some instances are still running. Waiting for them to
stop..."
                    )
```

```
                    time.sleep(10)
                else:
                    raise AutoScalerError(
                        f"Couldn't delete group {self.group_name}: {err}."
                    )

    def delete_group(self):
        """
        Terminates all instances in the group, deletes the EC2 Auto Scaling
 group.
        """
        try:
            response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[self.group_name]
            )
            groups = response.get("AutoScalingGroups", [])
            if len(groups) > 0:
                self.autoscaling_client.update_auto_scaling_group(
                    AutoScalingGroupName=self.group_name, MinSize=0
                )
                instance_ids = [inst["InstanceId"] for inst in groups[0]
["Instances"]]
                for inst_id in instance_ids:
                    self._try_terminate_instance(inst_id)
                self._try_delete_group()
            else:
                log.info("No groups found named %s, nothing to do.",
 self.group_name)
        except ClientError as err:
            raise AutoScalerError(f"Couldn't delete group {self.group_name}:
 {err}.")


    def get_default_vpc(self):
        """
        Gets the default VPC for the account.

        :return: Data about the default VPC.
        """
        try:
            response = self.ec2_client.describe_vpcs(
                Filters=[{"Name": "is-default", "Values": ["true"]}]
            )
        except ClientError as err:
```

```
            raise AutoScalerError(f"Couldn't get default VPC: {err}")
        else:
            return response["Vpcs"][0]


    def verify_inbound_port(self, vpc, port, ip_address):
        """
        Verify the default security group of the specified VPC allows ingress
from this
        computer. This can be done by allowing ingress from this computer's IP
        address. In some situations, such as connecting from a corporate network,
you
        must instead specify a prefix list ID. You can also temporarily open the
port to
        any IP address while running this example. If you do, be sure to remove
public
        access when you're done.

        :param vpc: The VPC used by this example.
        :param port: The port to verify.
        :param ip_address: This computer's IP address.
        :return: The default security group of the specific VPC, and a value that
indicates
                 whether the specified port is open.
        """
        try:
            response = self.ec2_client.describe_security_groups(
                Filters=[
                    {"Name": "group-name", "Values": ["default"]},
                    {"Name": "vpc-id", "Values": [vpc["VpcId"]]},
                ]
            )
            sec_group = response["SecurityGroups"][0]
            port_is_open = False
            log.info("Found default security group %s.", sec_group["GroupId"])
            for ip_perm in sec_group["IpPermissions"]:
                if ip_perm.get("FromPort", 0) == port:
                    log.info("Found inbound rule: %s", ip_perm)
                    for ip_range in ip_perm["IpRanges"]:
                        cidr = ip_range.get("CidrIp", "")
                        if cidr.startswith(ip_address) or cidr == "0.0.0.0/0":
                            port_is_open = True
                    if ip_perm["PrefixListIds"]:
                        port_is_open = True
```

```
                    if not port_is_open:
                        log.info(
                            "The inbound rule does not appear to be open to
either this computer's IP\n"
                            "address of %s, to all IP addresses (0.0.0.0/0), or
to a prefix list ID.",
                            ip_address,
                        )
                    else:
                        break
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't verify inbound rule for port {port} for VPC
{vpc['VpcId']}: {err}"
            )
        else:
            return sec_group, port_is_open


    def open_inbound_port(self, sec_group_id, port, ip_address):
        """
        Add an ingress rule to the specified security group that allows access on
the
        specified port from the specified IP address.

        :param sec_group_id: The ID of the security group to modify.
        :param port: The port to open.
        :param ip_address: The IP address that is granted access.
        """
        try:
            self.ec2_client.authorize_security_group_ingress(
                GroupId=sec_group_id,
                CidrIp=f"{ip_address}/32",
                FromPort=port,
                ToPort=port,
                IpProtocol="tcp",
            )
            log.info(
                "Authorized ingress to %s on port %s from %s.",
                sec_group_id,
                port,
                ip_address,
            )
        except ClientError as err:
```

```
            raise AutoScalerError(
                f"Couldn't authorize ingress to {sec_group_id} on port {port}
 from {ip_address}: {err}"
            )


    def get_subnets(self, vpc_id, zones):
        """
        Gets the default subnets in a VPC for a specified list of Availability
 Zones.

        :param vpc_id: The ID of the VPC to look up.
        :param zones: The list of Availability Zones to look up.
        :return: The list of subnets found.
        """
        try:
            response = self.ec2_client.describe_subnets(
                Filters=[
                    {"Name": "vpc-id", "Values": [vpc_id]},
                    {"Name": "availability-zone", "Values": zones},
                    {"Name": "default-for-az", "Values": ["true"]},
                ]
            )
            subnets = response["Subnets"]
            log.info("Found %s subnets for the specified zones.", len(subnets))
        except ClientError as err:
            raise AutoScalerError(f"Couldn't get subnets: {err}")
        else:
            return subnets
```

Elastic Load Balancing 작업을 래핑하는 클래스를 생성합니다.

```
class LoadBalancer:
    """Encapsulates Elastic Load Balancing (ELB) actions."""

    def __init__(self, target_group_name, load_balancer_name, elb_client):
        """
        :param target_group_name: The name of the target group associated with
 the load balancer.
```

```
        :param load_balancer_name: The name of the load balancer.
        :param elb_client: A Boto3 Elastic Load Balancing client.
        """
        self.target_group_name = target_group_name
        self.load_balancer_name = load_balancer_name
        self.elb_client = elb_client
        self._endpoint = None


    @classmethod
    def from_client(cls, resource_prefix):
        """
        Creates this class from a Boto3 client.

        :param resource_prefix: The prefix to give to AWS resources created by
this class.
        """
        elb_client = boto3.client("elbv2")
        return cls(f"{resource_prefix}-tg", f"{resource_prefix}-lb", elb_client)

    def endpoint(self):
        """
        Gets the HTTP endpoint of the load balancer.

        :return: The endpoint.
        """
        if self._endpoint is None:
            try:
                response = self.elb_client.describe_load_balancers(
                    Names=[self.load_balancer_name]
                )
                self._endpoint = response["LoadBalancers"][0]["DNSName"]
            except ClientError as err:
                raise LoadBalancerError(
                    f"Couldn't get the endpoint for load balancer
{self.load_balancer_name}: {err}"
                )
        return self._endpoint


    def create_target_group(self, protocol, port, vpc_id):
        """
        Creates an Elastic Load Balancing target group. The target group
specifies how
```

```
        the load balancer forward requests to instances in the group and how
instance
        health is checked.

        To speed up this demo, the health check is configured with shortened
times and
        lower thresholds. In production, you might want to decrease the
sensitivity of
        your health checks to avoid unwanted failures.

        :param protocol: The protocol to use to forward requests, such as 'HTTP'.
        :param port: The port to use to forward requests, such as 80.
        :param vpc_id: The ID of the VPC in which the load balancer exists.
        :return: Data about the newly created target group.
        """
        try:
            response = self.elb_client.create_target_group(
                Name=self.target_group_name,
                Protocol=protocol,
                Port=port,
                HealthCheckPath="/healthcheck",
                HealthCheckIntervalSeconds=10,
                HealthCheckTimeoutSeconds=5,
                HealthyThresholdCount=2,
                UnhealthyThresholdCount=2,
                VpcId=vpc_id,
            )
            target_group = response["TargetGroups"][0]
            log.info("Created load balancing target group %s.",
self.target_group_name)
        except ClientError as err:
            raise LoadBalancerError(
                f"Couldn't create load balancing target group
{self.target_group_name}: {err}"
            )
        else:
            return target_group


    def delete_target_group(self):
        """
        Deletes the target group.
        """
        done = False
```

```
        while not done:
            try:
                response = self.elb_client.describe_target_groups(
                    Names=[self.target_group_name]
                )
                tg_arn = response["TargetGroups"][0]["TargetGroupArn"]
                self.elb_client.delete_target_group(TargetGroupArn=tg_arn)
                log.info(
                    "Deleted load balancing target group %s.",
self.target_group_name
                )
                done = True
            except ClientError as err:
                if err.response["Error"]["Code"] == "TargetGroupNotFound":
                    log.info(
                        "Load balancer target group %s not found, nothing to
do.",
                        self.target_group_name,
                    )
                    done = True
                elif err.response["Error"]["Code"] == "ResourceInUse":
                    log.info(
                        "Target group not yet released from load balancer,
waiting..."
                    )
                    time.sleep(10)
                else:
                    raise LoadBalancerError(
                        f"Couldn't delete load balancing target group
{self.target_group_name}: {err}"
                    )


    def create_load_balancer(self, subnet_ids, target_group):
        """
        Creates an Elastic Load Balancing load balancer that uses the specified
subnets
        and forwards requests to the specified target group.

        :param subnet_ids: A list of subnets to associate with the load balancer.
        :param target_group: An existing target group that is added as a listener
to the
                             load balancer.
        :return: Data about the newly created load balancer.
```

```
        """
        try:
            response = self.elb_client.create_load_balancer(
                Name=self.load_balancer_name, Subnets=subnet_ids
            )
            load_balancer = response["LoadBalancers"][0]
            log.info("Created load balancer %s.", self.load_balancer_name)
            waiter = self.elb_client.get_waiter("load_balancer_available")
            log.info("Waiting for load balancer to be available...")
            waiter.wait(Names=[self.load_balancer_name])
            log.info("Load balancer is available!")
            self.elb_client.create_listener(
                LoadBalancerArn=load_balancer["LoadBalancerArn"],
                Protocol=target_group["Protocol"],
                Port=target_group["Port"],
                DefaultActions=[
                    {
                        "Type": "forward",
                        "TargetGroupArn": target_group["TargetGroupArn"],
                    }
                ],
            )
            log.info(
                "Created listener to forward traffic from load balancer %s to
target group %s.",
                self.load_balancer_name,
                target_group["TargetGroupName"],
            )
        except ClientError as err:
            raise LoadBalancerError(
                f"Failed to create load balancer {self.load_balancer_name}"
                f"and add a listener for target group
{target_group['TargetGroupName']}: {err}"
            )
        else:
            self._endpoint = load_balancer["DNSName"]
            return load_balancer


    def delete_load_balancer(self):
        """
        Deletes a load balancer.
        """
        try:
```

```
            response = self.elb_client.describe_load_balancers(
                Names=[self.load_balancer_name]
            )
            lb_arn = response["LoadBalancers"][0]["LoadBalancerArn"]
            self.elb_client.delete_load_balancer(LoadBalancerArn=lb_arn)
            log.info("Deleted load balancer %s.", self.load_balancer_name)
            waiter = self.elb_client.get_waiter("load_balancers_deleted")
            log.info("Waiting for load balancer to be deleted...")
            waiter.wait(Names=[self.load_balancer_name])
        except ClientError as err:
            if err.response["Error"]["Code"] == "LoadBalancerNotFound":
                log.info(
                    "Load balancer %s does not exist, nothing to do.",
                    self.load_balancer_name,
                )
            else:
                raise LoadBalancerError(
                    f"Couldn't delete load balancer {self.load_balancer_name}:
{err}"
                )


    def verify_load_balancer_endpoint(self):
        """
        Verify this computer can successfully send a GET request to the load
balancer endpoint.
        """
        success = False
        retries = 3
        while not success and retries > 0:
            try:
                lb_response = requests.get(f"http://{self.endpoint()}")
                log.info(
                    "Got response %s from load balancer endpoint.",
                    lb_response.status_code,
                )
                if lb_response.status_code == 200:
                    success = True
                else:
                    retries = 0
            except requests.exceptions.ConnectionError:
                log.info(
                    "Got connection error from load balancer endpoint,
retrying..."
```

```
                )
                retries -= 1
                time.sleep(10)
        return success

    def check_target_health(self):
        """
        Checks the health of the instances in the target group.

        :return: The health status of the target group.
        """
        try:
            tg_response = self.elb_client.describe_target_groups(
                Names=[self.target_group_name]
            )
            health_response = self.elb_client.describe_target_health(
                TargetGroupArn=tg_response["TargetGroups"][0]["TargetGroupArn"]
            )
        except ClientError as err:
            raise LoadBalancerError(
                f"Couldn't check health of {self.target_group_name} targets:
 {err}"
            )
        else:
            return health_response["TargetHealthDescriptions"]
```

DynamoDB를 사용하여 추천 서비스를 시뮬레이션하는 클래스를 생성합니다.

```
class RecommendationService:
    """
    Encapsulates a DynamoDB table to use as a service that recommends books,
 movies,
    and songs.
    """

    def __init__(self, table_name, dynamodb_client):
        """
        :param table_name: The name of the DynamoDB recommendations table.
        :param dynamodb_client: A Boto3 DynamoDB client.
```

```
        """
        self.table_name = table_name
        self.dynamodb_client = dynamodb_client

    @classmethod
    def from_client(cls, table_name):
        """
        Creates this class from a Boto3 client.

        :param table_name: The name of the DynamoDB recommendations table.
        """
        ddb_client = boto3.client("dynamodb")
        return cls(table_name, ddb_client)

    def create(self):
        """
        Creates a DynamoDB table to use a recommendation service. The table has a
        hash key named 'MediaType' that defines the type of media recommended,
such as
        Book or Movie, and a range key named 'ItemId' that, combined with the
MediaType,
        forms a unique identifier for the recommended item.

        :return: Data about the newly created table.
        """
        try:
            response = self.dynamodb_client.create_table(
                TableName=self.table_name,
                AttributeDefinitions=[
                    {"AttributeName": "MediaType", "AttributeType": "S"},
                    {"AttributeName": "ItemId", "AttributeType": "N"},
                ],
                KeySchema=[
                    {"AttributeName": "MediaType", "KeyType": "HASH"},
                    {"AttributeName": "ItemId", "KeyType": "RANGE"},
                ],
                ProvisionedThroughput={"ReadCapacityUnits": 5,
"WriteCapacityUnits": 5},
            )
            log.info("Creating table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s created.", self.table_name)
        except ClientError as err:
```

```
            if err.response["Error"]["Code"] == "ResourceInUseException":
                log.info("Table %s exists, nothing to be do.", self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when creating table: {err}."
                )
        else:
            return response

    def populate(self, data_file):
        """
        Populates the recommendations table from a JSON file.

        :param data_file: The path to the data file.
        """
        try:
            with open(data_file) as data:
                items = json.load(data)
            batch = [{"PutRequest": {"Item": item}} for item in items]
            self.dynamodb_client.batch_write_item(RequestItems={self.table_name:
batch})
            log.info(
                "Populated table %s with items from %s.", self.table_name,
data_file
            )
        except ClientError as err:
            raise RecommendationServiceError(
                self.table_name, f"Couldn't populate table from {data_file}:
{err}"
            )

    def destroy(self):
        """
        Deletes the recommendations table.
        """
        try:
            self.dynamodb_client.delete_table(TableName=self.table_name)
            log.info("Deleting table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_not_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s deleted.", self.table_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceNotFoundException":
```

```
                log.info("Table %s does not exist, nothing to do.",
 self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when deleting table: {err}."
                )
```

Systems Manager 작업을 래핑하는 클래스를 생성합니다.

```
class ParameterHelper:
    """
    Encapsulates Systems Manager parameters. This example uses these parameters
 to drive
    the demonstration of resilient architecture, such as failure of a dependency
 or
    how the service responds to a health check.
    """

    table = "doc-example-resilient-architecture-table"
    failure_response = "doc-example-resilient-architecture-failure-response"
    health_check = "doc-example-resilient-architecture-health-check"

    def __init__(self, table_name, ssm_client):
        """
        :param table_name: The name of the DynamoDB table that is used as a
 recommendation
                           service.
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.table_name = table_name

    @classmethod
    def from_client(cls, table_name):
        ssm_client = boto3.client("ssm")
        return cls(table_name, ssm_client)

    def reset(self):
        """
        Resets the Systems Manager parameters to starting values for the demo.
```

```
        These are the name of the DynamoDB recommendation table, no response when
a
        dependency fails, and shallow health checks.
        """
        self.put(self.table, self.table_name)
        self.put(self.failure_response, "none")
        self.put(self.health_check, "shallow")

    def put(self, name, value):
        """
        Sets the value of a named Systems Manager parameter.

        :param name: The name of the parameter.
        :param value: The new value of the parameter.
        """
        try:
            self.ssm_client.put_parameter(
                Name=name, Value=value, Overwrite=True, Type="String"
            )
            log.info("Setting demo parameter %s to '%s'.", name, value)
        except ClientError as err:
            raise ParameterHelperError(
                f"Couldn't set parameter {name} to {value}: {err}"
            )
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 다음 주제를 참조하십시오.
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate

- DeleteLoadBalancer
- DeleteTargetGroup
- DescribeAutoScalingGroups
- DescribeAvailabilityZones
- DescribeIamInstanceProfileAssociations
- DescribeInstances
- DescribeLoadBalancers
- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오SDK를 사용하여 Amazon EC2 리소스 생성 AWS. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

SDK를 사용하여 Amazon EC2 인스턴스로 시작하기 AWS

다음 코드 예제는 다음과 같은 작업을 수행하는 방법을 보여줍니다.

- 키 페어 및 보안 그룹을 생성합니다.
- Amazon Machine Image(AMI) 및 호환되는 인스턴스 유형을 선택한 다음 인스턴스를 생성합니다.
- 인스턴스를 중지한 후 다시 시작합니다.
- 인스턴스와 탄력적 IP 주소 연결.
- SSH로 인스턴스에 연결한 다음 리소스를 정리합니다.

.NET

AWS SDK for .NET

> **Note**
>
> 더 많은 정보가 있습니다 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

명령 프롬프트에서 시나리오를 실행합니다.

```
/// <summary>
/// Show Amazon Elastic Compute Cloud (Amazon EC2) Basics actions.
/// </summary>
public class EC2Basics
{
    /// <summary>
    /// Perform the actions defined for the Amazon EC2 Basics scenario.
    /// </summary>
    /// <param name="args">Command line arguments.</param>
    /// <returns>A Task object.</returns>
    static async Task Main(string[] args)
    {
        // Set up dependency injection for Amazon EC2 and Amazon Simple Systems
        // Management Service.
        using var host =
 Microsoft.Extensions.Hosting.Host.CreateDefaultBuilder(args)
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonEC2>()
                    .AddAWSService<IAmazonSimpleSystemsManagement>()
                    .AddTransient<EC2Wrapper>()
                    .AddTransient<SsmWrapper>()
            )
            .Build();

        // Now the client is available for injection.
        var ec2Client = host.Services.GetRequiredService<IAmazonEC2>();
        var ec2Methods = new EC2Wrapper(ec2Client);

        var ssmClient =
 host.Services.GetRequiredService<IAmazonSimpleSystemsManagement>();
        var ssmMethods = new SsmWrapper(ssmClient);
```

```
        var uiMethods = new UiMethods();

        var uniqueName = Guid.NewGuid().ToString();
        var keyPairName = "mvp-example-key-pair" + uniqueName;
        var groupName = "ec2-scenario-group" + uniqueName;
        var groupDescription = "A security group created for the EC2 Basics
scenario.";

        // Start the scenario.
        uiMethods.DisplayOverview();
        uiMethods.PressEnter();

        // Create the key pair.
        uiMethods.DisplayTitle("Create RSA key pair");
        Console.Write("Let's create an RSA key pair that you can be use to ");
        Console.WriteLine("securely connect to your EC2 instance.");
        var keyPair = await ec2Methods.CreateKeyPair(keyPairName);

        // Save key pair information to a temporary file.
        var tempFileName = ec2Methods.SaveKeyPair(keyPair);

        Console.WriteLine($"Created the key pair: {keyPair.KeyName} and saved it
to: {tempFileName}");
        string? answer;
        do
        {
            Console.Write("Would you like to list your existing key pairs? ");
            answer = Console.ReadLine();
        } while (answer!.ToLower() != "y" && answer.ToLower() != "n");

        if (answer == "y")
        {
            // List existing key pairs.
            uiMethods.DisplayTitle("Existing key pairs");

            // Passing an empty string to the DescribeKeyPairs method will return
            // a list of all existing key pairs.
            var keyPairs = await ec2Methods.DescribeKeyPairs("");
            keyPairs.ForEach(kp =>
            {
                Console.WriteLine($"{kp.KeyName} created at: {kp.CreateTime}
Fingerprint: {kp.KeyFingerprint}");
            });
        }
```

```
        uiMethods.PressEnter();

        // Create the security group.
        Console.WriteLine("Let's create a security group to manage access to your
 instance.");
        var secGroupId = await ec2Methods.CreateSecurityGroup(groupName,
 groupDescription);
        Console.WriteLine("Let's add rules to allow all HTTP and HTTPS inbound
 traffic and to allow SSH only from your current IP address.");

        uiMethods.DisplayTitle("Security group information");
        var secGroups = await ec2Methods.DescribeSecurityGroups(secGroupId);

        Console.WriteLine($"Created security group {groupName} in your default
 VPC.");
        secGroups.ForEach(group =>
        {
            ec2Methods.DisplaySecurityGroupInfoAsync(group);
        });
        uiMethods.PressEnter();

        Console.WriteLine("Now we'll authorize the security group we just created
 so that it can");
        Console.WriteLine("access the EC2 instances you create.");
        var success = await ec2Methods.AuthorizeSecurityGroupIngress(groupName);

        secGroups = await ec2Methods.DescribeSecurityGroups(secGroupId);
        Console.WriteLine($"Now let's look at the permissions again.");
        secGroups.ForEach(group =>
        {
            ec2Methods.DisplaySecurityGroupInfoAsync(group);
        });
        uiMethods.PressEnter();

        // Get list of available Amazon Linux 2 Amazon Machine Images (AMIs).
        var parameters = await ssmMethods.GetParametersByPath("/aws/service/ami-
amazon-linux-latest");

        List<string> imageIds = parameters.Select(param => param.Value).ToList();

        var images = await ec2Methods.DescribeImages(imageIds);

        var i = 1;
        images.ForEach(image =>
```

```
        {
            Console.WriteLine($"\t{i++}\t{image.Description}");
        });

        int choice;
        bool validNumber = false;

        do
        {
            Console.Write("Please select an image: ");
            var selImage = Console.ReadLine();
            validNumber = int.TryParse(selImage, out choice);
        } while (!validNumber);

        var selectedImage = images[choice - 1];

        // Display available instance types.
        uiMethods.DisplayTitle("Instance Types");
        var instanceTypes = await
ec2Methods.DescribeInstanceTypes(selectedImage.Architecture);

        i = 1;
        instanceTypes.ForEach(instanceType =>
        {
            Console.WriteLine($"\t{i++}\t{instanceType.InstanceType}");
        });

        do
        {
            Console.Write("Please select an instance type: ");
            var selImage = Console.ReadLine();
            validNumber = int.TryParse(selImage, out choice);
        } while (!validNumber);

        var selectedInstanceType = instanceTypes[choice - 1].InstanceType;

        // Create an EC2 instance.
        uiMethods.DisplayTitle("Creating an EC2 Instance");
        var instanceId = await ec2Methods.RunInstances(selectedImage.ImageId,
selectedInstanceType, keyPairName, secGroupId);
        Console.Write("Waiting for the instance to start.");
        var isRunning = false;
        do
        {
```

```
            isRunning = await ec2Methods.WaitForInstanceState(instanceId,
 InstanceStateName.Running);
        } while (!isRunning);

        uiMethods.PressEnter();

        var instance = await ec2Methods.DescribeInstance(instanceId);
        uiMethods.DisplayTitle("New Instance Information");
        ec2Methods.DisplayInstanceInformation(instance);

        Console.WriteLine("\nYou can use SSH to connect to your instance. For
 example:");
        Console.WriteLine($"\tssh -i {tempFileName} ec2-
user@{instance.PublicIpAddress}");

        uiMethods.PressEnter();

        Console.WriteLine("Now we'll stop the instance and then start it again to
 see what's changed.");

        await ec2Methods.StopInstances(instanceId);
        var hasStopped = false;
        do
        {
            hasStopped = await ec2Methods.WaitForInstanceState(instanceId,
 InstanceStateName.Stopped);
        } while (!hasStopped);

        Console.WriteLine("\nThe instance has stopped.");

        Console.WriteLine("Now let's start it up again.");
        await ec2Methods.StartInstances(instanceId);
        Console.Write("Waiting for instance to start. ");

        isRunning = false;
        do
        {
            isRunning = await ec2Methods.WaitForInstanceState(instanceId,
 InstanceStateName.Running);
        } while (!isRunning);

        Console.WriteLine("\nLet's see what changed.");

        instance = await ec2Methods.DescribeInstance(instanceId);
```

```
        uiMethods.DisplayTitle("New Instance Information");
        ec2Methods.DisplayInstanceInformation(instance);

        Console.WriteLine("\nNotice the change in the SSH information:");
        Console.WriteLine($"\tssh -i {tempFileName} ec2-
user@{instance.PublicIpAddress}");

        uiMethods.PressEnter();

        Console.WriteLine("Now we will stop the instance again. Then we will
 create and associate an");
        Console.WriteLine("Elastic IP address to use with our instance.");

        await ec2Methods.StopInstances(instanceId);
        hasStopped = false;
        do
        {
            hasStopped = await ec2Methods.WaitForInstanceState(instanceId,
 InstanceStateName.Stopped);
        } while (!hasStopped);

        Console.WriteLine("\nThe instance has stopped.");
        uiMethods.PressEnter();

        uiMethods.DisplayTitle("Allocate Elastic IP address");
        Console.WriteLine("You can allocate an Elastic IP address and associate
 it with your instance\nto keep a consistent IP address even when your instance
 restarts.");
        var allocationId = await ec2Methods.AllocateAddress();
        Console.WriteLine("Now we will associate the Elastic IP address with our
 instance.");
        var associationId = await ec2Methods.AssociateAddress(allocationId,
 instanceId);

        // Start the instance again.
        Console.WriteLine("Now let's start the instance again.");
        await ec2Methods.StartInstances(instanceId);
        Console.Write("Waiting for instance to start. ");

        isRunning = false;
        do
        {
            isRunning = await ec2Methods.WaitForInstanceState(instanceId,
 InstanceStateName.Running);
```

```
        } while (!isRunning);

        Console.WriteLine("\nLet's see what changed.");

        instance = await ec2Methods.DescribeInstance(instanceId);
        uiMethods.DisplayTitle("Instance information");
        ec2Methods.DisplayInstanceInformation(instance);

        Console.WriteLine("\nHere is the SSH information:");
        Console.WriteLine($"\tssh -i {tempFileName} ec2-
user@{instance.PublicIpAddress}");

        Console.WriteLine("Let's stop and start the instance again.");
        uiMethods.PressEnter();

        await ec2Methods.StopInstances(instanceId);

        hasStopped = false;
        do
        {
            hasStopped = await ec2Methods.WaitForInstanceState(instanceId,
 InstanceStateName.Stopped);
        } while (!hasStopped);

        Console.WriteLine("\nThe instance has stopped.");

        Console.WriteLine("Now let's start it up again.");
        await ec2Methods.StartInstances(instanceId);
        Console.Write("Waiting for instance to start. ");

        isRunning = false;
        do
        {
            isRunning = await ec2Methods.WaitForInstanceState(instanceId,
 InstanceStateName.Running);
        } while (!isRunning);

        instance = await ec2Methods.DescribeInstance(instanceId);
        uiMethods.DisplayTitle("New Instance Information");
        ec2Methods.DisplayInstanceInformation(instance);
        Console.WriteLine("Note that the IP address did not change this time.");
        uiMethods.PressEnter();

        uiMethods.DisplayTitle("Clean up resources");
```

```
        Console.WriteLine("Now let's clean up the resources we created.");

        // Terminate the instance.
        Console.WriteLine("Terminating the instance we created.");
        var stateChange = await ec2Methods.TerminateInstances(instanceId);

        // Wait for the instance state to be terminated.
        var hasTerminated = false;
        do
        {
            hasTerminated = await ec2Methods.WaitForInstanceState(instanceId,
InstanceStateName.Terminated);
        } while (!hasTerminated);

        Console.WriteLine($"\nThe instance {instanceId} has been terminated.");
        Console.WriteLine("Now we can disassociate the Elastic IP address and
release it.");

        // Disassociate the Elastic IP address.
        var disassociated = ec2Methods.DisassociateIp(associationId);

        // Delete the Elastic IP address.
        var released = ec2Methods.ReleaseAddress(allocationId);

        // Delete the security group.
        Console.WriteLine($"Deleting the Security Group: {groupName}.");
        success = await ec2Methods.DeleteSecurityGroup(secGroupId);
        if (success)
        {
            Console.WriteLine($"Successfully deleted {groupName}.");
        }

        // Delete the RSA key pair.
        Console.WriteLine($"Deleting the key pair: {keyPairName}");
        await ec2Methods.DeleteKeyPair(keyPairName);
        Console.WriteLine("Deleting the temporary file with the key
information.");
        ec2Methods.DeleteTempFile(tempFileName);
        uiMethods.PressEnter();

        uiMethods.DisplayTitle("EC2 Basics Scenario completed.");
        uiMethods.PressEnter();
    }
```

```
}
```

EC2 작업을 래핑하는 클래스를 정의합니다.

```
/// <summary>
/// Methods of this class perform Amazon Elastic Compute Cloud (Amazon EC2).
/// </summary>
public class EC2Wrapper
{
    private readonly IAmazonEC2 _amazonEC2;

    public EC2Wrapper(IAmazonEC2 amazonService)
    {
        _amazonEC2 = amazonService;
    }

    /// <summary>
    /// Allocate an Elastic IP address.
    /// </summary>
    /// <returns>The allocation Id of the allocated address.</returns>
    public async Task<string> AllocateAddress()
    {
        var request = new AllocateAddressRequest();

        var response = await _amazonEC2.AllocateAddressAsync(request);
        return response.AllocationId;
    }

    /// <summary>
    /// Associate an Elastic IP address to an EC2 instance.
    /// </summary>
    /// <param name="allocationId">The allocation Id of an Elastic IP address.</
param>
    /// <param name="instanceId">The instance Id of the EC2 instance to
    /// associate the address with.</param>
    /// <returns>The association Id that represents
    /// the association of the Elastic IP address with an instance.</returns>
    public async Task<string> AssociateAddress(string allocationId, string
 instanceId)
    {
        var request = new AssociateAddressRequest
        {
```

```
            AllocationId = allocationId,
            InstanceId = instanceId
        };

        var response = await _amazonEC2.AssociateAddressAsync(request);
        return response.AssociationId;
    }

    /// <summary>
    /// Authorize the local computer ingress to EC2 instances associated
    /// with the virtual private cloud (VPC) security group.
    /// </summary>
    /// <param name="groupName">The name of the security group.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AuthorizeSecurityGroupIngress(string groupName)
    {
        // Get the IP address for the local computer.
        var ipAddress = await GetIpAddress();
        Console.WriteLine($"Your IP address is: {ipAddress}");
        var ipRanges = new List<IpRange> { new IpRange { CidrIp =
 $"{ipAddress}/32" } };
        var permission = new IpPermission
        {
            Ipv4Ranges = ipRanges,
            IpProtocol = "tcp",
            FromPort = 22,
            ToPort = 22
        };
        var permissions = new List<IpPermission> { permission };
        var response = await _amazonEC2.AuthorizeSecurityGroupIngressAsync(
            new AuthorizeSecurityGroupIngressRequest(groupName, permissions));
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Authorize the local computer for ingress to
    /// the Amazon EC2 SecurityGroup.
    /// </summary>
    /// <returns>The IPv4 address of the computer running the scenario.</returns>
    private static async Task<string> GetIpAddress()
    {
        var httpClient = new HttpClient();
        var ipString = await httpClient.GetStringAsync("https://
checkip.amazonaws.com");
```

```
        // The IP address is returned with a new line
        // character on the end. Trim off the whitespace and
        // return the value to the caller.
        return ipString.Trim();
    }

    /// <summary>
    /// Create an Amazon EC2 key pair.
    /// </summary>
    /// <param name="keyPairName">The name for the new key pair.</param>
    /// <returns>The Amazon EC2 key pair created.</returns>
    public async Task<KeyPair?> CreateKeyPair(string keyPairName)
    {
        var request = new CreateKeyPairRequest
        {
            KeyName = keyPairName,
        };

        var response = await _amazonEC2.CreateKeyPairAsync(request);

        if (response.HttpStatusCode == HttpStatusCode.OK)
        {
            var kp = response.KeyPair;
            return kp;
        }
        else
        {
            Console.WriteLine("Could not create key pair.");
            return null;
        }
    }

    /// <summary>
    /// Save KeyPair information to a temporary file.
    /// </summary>
    /// <param name="keyPair">The name of the key pair.</param>
    /// <returns>The full path to the temporary file.</returns>
    public string SaveKeyPair(KeyPair keyPair)
    {
        var tempPath = Path.GetTempPath();
        var tempFileName = $"{tempPath}\\{Path.GetRandomFileName()}";
        var pemFileName = Path.ChangeExtension(tempFileName, "pem");
```

```
        // Save the key pair to a file in a temporary folder.
        using var stream = new FileStream(pemFileName, FileMode.Create);
        using var writer = new StreamWriter(stream);
        writer.WriteLine(keyPair.KeyMaterial);

        return pemFileName;
    }

    /// <summary>
    /// Create an Amazon EC2 security group.
    /// </summary>
    /// <param name="groupName">The name for the new security group.</param>
    /// <param name="groupDescription">A description of the new security group.</
param>
    /// <returns>The group Id of the new security group.</returns>
    public async Task<string> CreateSecurityGroup(string groupName, string
 groupDescription)
    {
        var response = await _amazonEC2.CreateSecurityGroupAsync(
            new CreateSecurityGroupRequest(groupName, groupDescription));

        return response.GroupId;
    }


    /// <summary>
    /// Create a new Amazon EC2 VPC.
    /// </summary>
    /// <param name="cidrBlock">The CIDR block for the new security group.</
param>
    /// <returns>The VPC Id of the new VPC.</returns>
    public async Task<string?> CreateVPC(string cidrBlock)
    {

        try
        {
            var response = await _amazonEC2.CreateVpcAsync(new CreateVpcRequest
            {
                CidrBlock = cidrBlock,
            });

            Vpc vpc = response.Vpc;
            Console.WriteLine($"Created VPC with ID: {vpc.VpcId}.");
            return vpc.VpcId;
```

```
        }
        catch (AmazonEC2Exception ex)
        {
            Console.WriteLine($"Couldn't create VPC because: {ex.Message}");
            return null;
        }
    }

    /// <summary>
    /// Delete an Amazon EC2 key pair.
    /// </summary>
    /// <param name="keyPairName">The name of the key pair to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteKeyPair(string keyPairName)
    {
        try
        {
            await _amazonEC2.DeleteKeyPairAsync(new
DeleteKeyPairRequest(keyPairName)).ConfigureAwait(false);
            return true;
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Couldn't delete the key pair because:
{ex.Message}");
            return false;
        }
    }

    /// <summary>
    /// Delete the temporary file where the key pair information was saved.
    /// </summary>
    /// <param name="tempFileName">The path to the temporary file.</param>
    public void DeleteTempFile(string tempFileName)
    {
        if (File.Exists(tempFileName))
        {
            File.Delete(tempFileName);
        }
    }

    /// <summary>
    /// Delete an Amazon EC2 security group.
    /// </summary>
```

```
    /// <param name="groupName">The name of the group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteSecurityGroup(string groupId)
    {
        var response = await _amazonEC2.DeleteSecurityGroupAsync(new
DeleteSecurityGroupRequest { GroupId = groupId });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Delete an Amazon EC2 VPC.
    /// </summary>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteVpc(string vpcId)
    {
        var request = new DeleteVpcRequest
        {
            VpcId = vpcId,
        };

        var response = await _amazonEC2.DeleteVpcAsync(request);

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
    /// Get information about existing Amazon EC2 images.
    /// </summary>
    /// <returns>A list of image information.</returns>
    public async Task<List<Image>> DescribeImages(List<string>? imageIds)
    {
        var request = new DescribeImagesRequest();
        if (imageIds is not null)
        {
            // If the imageIds list is not null, add the list
            // to the request object.
            request.ImageIds = imageIds;
        }

        var response = await _amazonEC2.DescribeImagesAsync(request);
        return response.Images;
    }

    /// <summary>
```

```
    /// Display the information returned by DescribeImages.
    /// </summary>
    /// <param name="images">The list of image information to display.</param>
    public void DisplayImageInfo(List<Image> images)
    {
        images.ForEach(image =>
        {
            Console.WriteLine($"{image.Name} Created on: {image.CreationDate}");
        });

    }

    /// <summary>
    /// Get information about an Amazon EC2 instance.
    /// </summary>
    /// <param name="instanceId">The instance Id of the EC2 instance.</param>
    /// <returns>An EC2 instance.</returns>
    public async Task<Instance> DescribeInstance(string instanceId)
    {
        var response = await _amazonEC2.DescribeInstancesAsync(
            new DescribeInstancesRequest { InstanceIds = new List<string>
{ instanceId } });
        return response.Reservations[0].Instances[0];
    }

    /// <summary>
    /// Display EC2 instance information.
    /// </summary>
    /// <param name="instance">The instance Id of the EC2 instance.</param>
    public void DisplayInstanceInformation(Instance instance)
    {
        Console.WriteLine($"ID: {instance.InstanceId}");
        Console.WriteLine($"Image ID: {instance.ImageId}");
        Console.WriteLine($"{instance.InstanceType}");
        Console.WriteLine($"Key Name: {instance.KeyName}");
        Console.WriteLine($"VPC ID: {instance.VpcId}");
        Console.WriteLine($"Public IP: {instance.PublicIpAddress}");
        Console.WriteLine($"State: {instance.State.Name}");
    }

    /// <summary>
    /// Get information about existing EC2 images.
    /// </summary>
    /// <returns>Async task.</returns>
```

```
    public async Task DescribeInstances()
    {
        // List all EC2 instances.
        await GetInstanceDescriptions();

        string tagName = "IncludeInList";
        string tagValue = "Yes";
        await GetInstanceDescriptionsFiltered(tagName, tagValue);
    }

    /// <summary>
    /// Get information for all existing Amazon EC2 instances.
    /// </summary>
    /// <returns>Async task.</returns>
    public async Task GetInstanceDescriptions()
    {
        Console.WriteLine("Showing all instances:");
        var paginator = _amazonEC2.Paginators.DescribeInstances(new
DescribeInstancesRequest());

        await foreach (var response in paginator.Responses)
        {
            foreach (var reservation in response.Reservations)
            {
                foreach (var instance in reservation.Instances)
                {
                    Console.Write($"Instance ID: {instance.InstanceId}");
                    Console.WriteLine($"\tCurrent State: {instance.State.Name}");
                }
            }
        }
    }

    /// <summary>
    /// Get information about EC2 instances filtered by a tag name and value.
    /// </summary>
    /// <param name="tagName">The name of the tag to filter on.</param>
    /// <param name="tagValue">The value of the tag to look for.</param>
    /// <returns>Async task.</returns>
    public async Task GetInstanceDescriptionsFiltered(string tagName, string
tagValue)
    {
        // This tag filters the results of the instance list.
        var filters = new List<Filter>
```

```
        {
            new Filter
            {
                Name = $"tag:{tagName}",
                Values = new List<string>
                {
                    tagValue,
                },
            },
        };
        var request = new DescribeInstancesRequest
        {
            Filters = filters,
        };

        Console.WriteLine("\nShowing instances with tag: \"IncludeInList\" set to
\"Yes\".");
        var paginator = _amazonEC2.Paginators.DescribeInstances(request);

        await foreach (var response in paginator.Responses)
        {
            foreach (var reservation in response.Reservations)
            {
                foreach (var instance in reservation.Instances)
                {
                    Console.Write($"Instance ID: {instance.InstanceId} ");
                    Console.WriteLine($"\tCurrent State: {instance.State.Name}");
                }
            }
        }
    }

    /// <summary>
    /// Describe the instance types available.
    /// </summary>
    /// <returns>A list of instance type information.</returns>
    public async Task<List<InstanceTypeInfo>>
DescribeInstanceTypes(ArchitectureValues architecture)
    {
        var request = new DescribeInstanceTypesRequest();

        var filters = new List<Filter>
            { new Filter("processor-info.supported-architecture", new
List<string> { architecture.ToString() }) };
```

```
        filters.Add(new Filter("instance-type", new() { "*.micro", "*.small" }));

        request.Filters = filters;
        var instanceTypes = new List<InstanceTypeInfo>();

        var paginator = _amazonEC2.Paginators.DescribeInstanceTypes(request);
        await foreach (var instanceType in paginator.InstanceTypes)
        {
            instanceTypes.Add(instanceType);
        }
        return instanceTypes;
    }

    /// <summary>
    /// Display the instance type information returned by
 DescribeInstanceTypesAsync.
    /// </summary>
    /// <param name="instanceTypes">The list of instance type information.</
param>
    public void DisplayInstanceTypeInfo(List<InstanceTypeInfo> instanceTypes)
    {
        instanceTypes.ForEach(type =>
        {
            Console.WriteLine($"{type.InstanceType}\t{type.MemoryInfo}");
        });
    }

    /// <summary>
    /// Get information about an Amazon EC2 key pair.
    /// </summary>
    /// <param name="keyPairName">The name of the key pair.</param>
    /// <returns>A list of key pair information.</returns>
    public async Task<List<KeyPairInfo>> DescribeKeyPairs(string keyPairName)
    {
        var request = new DescribeKeyPairsRequest();
        if (!string.IsNullOrEmpty(keyPairName))
        {
            request = new DescribeKeyPairsRequest
            {
                KeyNames = new List<string> { keyPairName }
            };
        }
        var response = await _amazonEC2.DescribeKeyPairsAsync(request);
        return response.KeyPairs.ToList();
```

```
    }


    /// <summary>
    /// Retrieve information for an Amazon EC2 security group.
    /// </summary>
    /// <param name="groupId">The Id of the Amazon EC2 security group.</param>
    /// <returns>A list of security group information.</returns>
    public async Task<List<SecurityGroup>> DescribeSecurityGroups(string groupId)
    {
        var request = new DescribeSecurityGroupsRequest();
        var groupIds = new List<string> { groupId };
        request.GroupIds = groupIds;

        var response = await _amazonEC2.DescribeSecurityGroupsAsync(request);
        return response.SecurityGroups;
    }

    /// <summary>
    /// Display the information returned by the call to
    /// DescribeSecurityGroupsAsync.
    /// </summary>
    /// <param name="securityGroup">A list of security group information.</param>
    public void DisplaySecurityGroupInfoAsync(SecurityGroup securityGroup)
    {
        Console.WriteLine($"{securityGroup.GroupName}");
        Console.WriteLine("Ingress permissions:");
        securityGroup.IpPermissions.ForEach(permission =>
        {
            Console.WriteLine($"\tFromPort: {permission.FromPort}");
            Console.WriteLine($"\tIpProtocol: {permission.IpProtocol}");

            Console.Write($"\tIpv4Ranges: ");
            permission.Ipv4Ranges.ForEach(range =>
{ Console.Write($"{range.CidrIp} "); });

            Console.WriteLine($"\n\tIpv6Ranges:");
            permission.Ipv6Ranges.ForEach(range =>
{ Console.Write($"{range.CidrIpv6} "); });

            Console.Write($"\n\tPrefixListIds: ");
            permission.PrefixListIds.ForEach(id => Console.Write($"{id.Id} "));

            Console.WriteLine($"\n\tTo Port: {permission.ToPort}");
```

```
        });
        Console.WriteLine("Egress permissions:");
        securityGroup.IpPermissionsEgress.ForEach(permission =>
        {
            Console.WriteLine($"\tFromPort: {permission.FromPort}");
            Console.WriteLine($"\tIpProtocol: {permission.IpProtocol}");

            Console.Write($"\tIpv4Ranges: ");
            permission.Ipv4Ranges.ForEach(range =>
{ Console.Write($"{range.CidrIp} "); });

            Console.WriteLine($"\n\tIpv6Ranges:");
            permission.Ipv6Ranges.ForEach(range =>
{ Console.Write($"{range.CidrIpv6} "); });

            Console.Write($"\n\tPrefixListIds: ");
            permission.PrefixListIds.ForEach(id => Console.Write($"{id.Id} "));

            Console.WriteLine($"\n\tTo Port: {permission.ToPort}");
        });
    }


    /// <summary>
    /// Disassociate an Elastic IP address from an EC2 instance.
    /// </summary>
    /// <param name="associationId">The association Id.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DisassociateIp(string associationId)
    {
        var response = await _amazonEC2.DisassociateAddressAsync(
            new DisassociateAddressRequest { AssociationId = associationId });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Retrieve a list of available Amazon Linux images.
    /// </summary>
    /// <returns>A list of image information.</returns>
    public async Task<List<Image>> GetEC2AmiList()
    {
        var filter = new Filter { Name = "architecture", Values = new
List<string> { "x86_64" } };
        var filters = new List<Filter> { filter };
```

```
        var response = await _amazonEC2.DescribeImagesAsync(new
 DescribeImagesRequest { Filters = filters });
        return response.Images;
    }

    /// <summary>
    /// Reboot EC2 instances.
    /// </summary>
    /// <param name="ec2InstanceId">The instance Id of the instances that will be
 rebooted.</param>
    /// <returns>Async task.</returns>
    public async Task RebootInstances(string ec2InstanceId)
    {
        var request = new RebootInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId },
        };

        var response = await _amazonEC2.RebootInstancesAsync(request);
        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            Console.WriteLine("Instances successfully rebooted.");
        }
        else
        {
            Console.WriteLine("Could not reboot one or more instances.");
        }
    }

    /// <summary>
    /// Release an Elastic IP address.
    /// </summary>
    /// <param name="allocationId">The allocation Id of the Elastic IP address.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> ReleaseAddress(string allocationId)
    {
        var request = new ReleaseAddressRequest
        {
            AllocationId = allocationId
        };

        var response = await _amazonEC2.ReleaseAddressAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
```

```
    }

    /// <summary>
    /// Create and run an EC2 instance.
    /// </summary>
    /// <param name="ImageId">The image Id of the image used as a basis for the
    /// EC2 instance.</param>
    /// <param name="instanceType">The instance type of the EC2 instance to
create.</param>
    /// <param name="keyName">The name of the key pair to associate with the
    /// instance.</param>
    /// <param name="groupId">The Id of the Amazon EC2 security group that will
be
    /// allowed to interact with the new EC2 instance.</param>
    /// <returns>The instance Id of the new EC2 instance.</returns>
    public async Task<string> RunInstances(string imageId, string instanceType,
string keyName, string groupId)
    {
        var request = new RunInstancesRequest
        {
            ImageId = imageId,
            InstanceType = instanceType,
            KeyName = keyName,
            MinCount = 1,
            MaxCount = 1,
            SecurityGroupIds = new List<string> { groupId }
        };
        var response = await _amazonEC2.RunInstancesAsync(request);
        return response.Reservation.Instances[0].InstanceId;
    }


    /// <summary>
    /// Start an EC2 instance.
    /// </summary>
    /// <param name="ec2InstanceId">The instance Id of the Amazon EC2 instance
    /// to start.</param>
    /// <returns>Async task.</returns>
    public async Task StartInstances(string ec2InstanceId)
    {
        var request = new StartInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId },
        };
```

```
        var response = await _amazonEC2.StartInstancesAsync(request);

        if (response.StartingInstances.Count > 0)
        {
            var instances = response.StartingInstances;
            instances.ForEach(i =>
            {
                Console.WriteLine($"Successfully started the EC2 instance with
instance ID: {i.InstanceId}.");
            });
        }
    }

    /// <summary>
    /// Stop an EC2 instance.
    /// </summary>
    /// <param name="ec2InstanceId">The instance Id of the EC2 instance to
    /// stop.</param>
    /// <returns>Async task.</returns>
    public async Task StopInstances(string ec2InstanceId)
    {
        // In addition to the list of instance Ids, the
        // request can also include the following properties:
        //     Force      When true, forces the instances to
        //                stop but you must check the integrity
        //                of the file system. Not recommended on
        //                Windows instances.
        //     Hibernate  When true, hibernates the instance if the
        //                instance was enabled for hibernation when
        //                it was launched.
        var request = new StopInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId },
        };

        var response = await _amazonEC2.StopInstancesAsync(request);

        if (response.StoppingInstances.Count > 0)
        {
            var instances = response.StoppingInstances;
            instances.ForEach(i =>
            {
                Console.WriteLine($"Successfully stopped the EC2 Instance " +
```

```
                                $"with InstanceID: {i.InstanceId}.");
            });
        }
    }

    /// <summary>
    /// Terminate an EC2 instance.
    /// </summary>
    /// <param name="ec2InstanceId">The instance Id of the EC2 instance
    /// to terminate.</param>
    /// <returns>Async task.</returns>
    public async Task<List<InstanceStateChange>> TerminateInstances(string
ec2InstanceId)
    {
        var request = new TerminateInstancesRequest
        {
            InstanceIds = new List<string> { ec2InstanceId }
        };

        var response = await _amazonEC2.TerminateInstancesAsync(request);
        return response.TerminatingInstances;
    }

    /// <summary>
    /// Wait until an EC2 instance is in a specified state.
    /// </summary>
    /// <param name="instanceId">The instance Id.</param>
    /// <param name="stateName">The state to wait for.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitForInstanceState(string instanceId,
InstanceStateName stateName)
    {
        var request = new DescribeInstancesRequest
        {
            InstanceIds = new List<string> { instanceId }
        };

        // Wait until the instance is running.
        var hasState = false;
        do
        {
            // Wait 5 seconds.
            Thread.Sleep(5000);
```

```
            // Check for the desired state.
            var response = await _amazonEC2.DescribeInstancesAsync(request);
            var instance = response.Reservations[0].Instances[0];
            hasState = instance.State.Name == stateName;
            Console.Write(". ");
        } while (!hasState);

        return hasState;
    }

}
```

- API 세부 정보는 AWS SDK for .NET API 참조의 다음 주제를 참조하십시오.
  - AllocateAddress
  - AssociateAddress
  - AuthorizeSecurityGroupIngress
  - CreateKeyPair
  - CreateSecurityGroup
  - DeleteKeyPair
  - DeleteSecurityGroup
  - DescribeImages
  - DescribeInstanceTypes
  - DescribeInstances
  - DescribeKeyPairs
  - DescribeSecurityGroups
  - DisassociateAddress
  - ReleaseAddress
  - RunInstances
  - StartInstances
  - StopInstances
  - TerminateInstances
  - UnmonitorInstances

Bash

AWS CLI Bash 스크립트 사용

> **Note**
>
> 더 많은 정보가 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

명령 프롬프트에서 대화형 시나리오를 실행합니다.

```
###############################################################################
# function get_started_with_ec2_instances
#
# Runs an interactive scenario that shows how to get started using EC2 instances.
#
#     "EC2 access" permissions are needed to run this code.
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function get_started_with_ec2_instances() {
  # Requires version 4 for mapfile.
  local required_version=4.0

  # Get the current Bash version
  # Check if BASH_VERSION is set
  local current_version
  if [[ -n "$BASH_VERSION" ]]; then
    # Convert BASH_VERSION to a number for comparison
    current_version=$BASH_VERSION
  else
    # Get the current Bash version using the bash command
    current_version=$(bash --version | head -n 1 | awk '{ print $4 }')
  fi

  # Convert version strings to numbers for comparison
  local required_version_num current_version_num
  required_version_num=$(echo "$required_version" | awk -F. '{ print ($1 * 10000)
 + ($2 * 100) + $3 }')
```

```
 current_version_num=$(echo "$current_version" | awk -F. '{ print ($1 * 10000) +
($2 * 100) + $3 }')

 # Compare versions
 if ((current_version_num < required_version_num)); then
   echo "Error: This script requires Bash version $required_version or higher."
   echo "Your current Bash version is number is $current_version."
   exit 1
 fi

 {
   if [ "$EC2_OPERATIONS_SOURCED" != "True" ]; then

     source ./ec2_operations.sh
   fi
 }

 echo_repeat "*" 88
 echo "Welcome to the Amazon Elastic Compute Cloud (Amazon EC2) get started with
instances demo."
 echo_repeat "*" 88
 echo

 echo "Let's create an RSA key pair that you can be use to securely connect to "
 echo "your EC2 instance."

 echo -n "Enter a unique name for your key: "
 get_input
 local key_name
 key_name=$get_input_result

 local temp_dir
 temp_dir=$(mktemp -d)
 local key_file_name="$temp_dir/${key_name}.pem"

 if ec2_create_keypair -n "${key_name}" -f "${key_file_name}"; then
   echo "Created a key pair $key_name and saved the private key to
$key_file_name"
   echo
 else
   errecho "The key pair failed to create. This demo will exit."
   return 1
 fi
```

```
  chmod 400 "${key_file_name}"

  if yes_no_input "Do you want to list some of your key pairs? (y/n) "; then
    local keys_and_fingerprints
    keys_and_fingerprints="$(ec2_describe_key_pairs)" && {
      local image_name_and_id
      while IFS=$'\n' read -r image_name_and_id; do
        local entries
        IFS=$'\t' read -ra entries <<<"$image_name_and_id"
        echo "Found rsa key ${entries[0]} with fingerprint:"
        echo "     ${entries[1]}"
      done <<<"$keys_and_fingerprints"

    }
  fi

  echo_repeat "*" 88
  echo_repeat "*" 88

  echo "Let's create a security group to manage access to your instance."
  echo -n "Enter a unique name for your security group: "
  get_input
  local security_group_name
  security_group_name=$get_input_result
  local security_group_id
  security_group_id=$(ec2_create_security_group -n "$security_group_name" \
    -d "Security group for EC2 instance") || {
    errecho "The security failed to create. This demo will exit."
    clean_up "$key_name" "$key_file_name"
    return 1
  }

  echo "Security group created with ID $security_group_id"
  echo

  local public_ip
  public_ip=$(curl -s http://checkip.amazonaws.com)

  echo "Let's add a rule to allow SSH only from your current IP address."
  echo "Your public IP address is $public_ip"
  echo -n "press return to add this rule to your security group."
  get_input
```

```
  if ! ec2_authorize_security_group_ingress -g "$security_group_id" -i
 "$public_ip" -p tcp -f 22 -t 22; then
    errecho "The security group rules failed to update. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1
  fi

  echo "Security group rules updated"

  local security_group_description
  security_group_description="$(ec2_describe_security_groups -g
 "${security_group_id}")" || {
    errecho "Failed to describe security groups. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1
  }

  mapfile -t parameters <<<"$security_group_description"
  IFS=$'\t' read -ra entries <<<"${parameters[0]}"
  echo "Security group: ${entries[0]}"
  echo "    ID: ${entries[1]}"
  echo "    VPC: ${entries[2]}"
  echo "Inbound permissions:"
  IFS=$'\t' read -ra entries <<<"${parameters[1]}"
  echo "    IpProtocol: ${entries[0]}"
  echo "    FromPort: ${entries[1]}"
  echo "    ToPort: ${entries[2]}"
  echo "    CidrIp: ${parameters[2]}"

  local parameters
  parameters="$(ssm_get_parameters_by_path -p "/aws/service/ami-amazon-linux-
latest")" || {
    errecho "Failed to get parameters. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1

  }

  local image_ids=""
  mapfile -t parameters <<<"$parameters"
  for image_name_and_id in "${parameters[@]}"; do
    IFS=$'\t' read -ra values <<<"$image_name_and_id"
    if [[ "${values[0]}" == *"amzn2"* ]]; then
      image_ids+="${values[1]} "
```

```
    fi
  done

  local images
  images="$(ec2_describe_images -i "$image_ids")" || {
    errecho "Failed to describe images. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1

  }

  new_line_and_tab_to_list "$images"
  local images=("${list_result[@]}")

  # Get the size of the array
  local images_count=${#images[@]}

  if ((images_count == 0)); then
    errecho "No images found. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1
  fi

  echo_repeat "*" 88
  echo_repeat "*" 88

  echo "Let's create an instance from an Amazon Linux 2 AMI. Here are some
options:"
  for ((i = 0; i < images_count; i += 3)); do
    echo "$(((i / 3) + 1)) - ${images[$i]}"
  done

  integer_input "Please enter the number of the AMI you want to use: " 1
"$((images_count / 3))"
  local choice=$get_input_result
  choice=$(((choice - 1) * 3))

  echo "Great choice."
  echo

  local architecture=${images[$((choice + 1))]}
  local image_id=${images[$((choice + 2))]}
  echo "Here are some instance types that support the ${architecture}
architecture of the image:"
```

```
  response="$(ec2_describe_instance_types -a "${architecture}" -t
 "*.micro,*.small")" || {
    errecho "Failed to describe instance types. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1
  }

  local instance_types
  mapfile -t instance_types <<<"$response"

  # Get the size of the array
  local instance_types_count=${#instance_types[@]}

  echo "Here are some options:"
  for ((i = 0; i < instance_types_count; i++)); do
    echo "$((i + 1)) - ${instance_types[$i]}"
  done

  integer_input "Which one do you want to use? " 1 "${#instance_types[@]}
"
  choice=$get_input_result
  local instance_type=${instance_types[$((choice - 1))]}
  echo "Another great choice."
  echo

  echo "Creating your instance and waiting for it to start..."
  local instance_id
  instance_id=$(ec2_run_instances -i "$image_id" -t "$instance_type" -k
 "$key_name" -s "$security_group_id") || {
    errecho "Failed to run instance. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id"
    return 1
  }

  ec2_wait_for_instance_running -i "$instance_id"
  echo "Your instance is ready:"
  echo

  local instance_details
  instance_details="$(ec2_describe_instances -i "${instance_id}")"

  echo
  print_instance_details "${instance_details}"
```

```
 local public_ip
 public_ip=$(echo "${instance_details}" | awk '{print $6}')
 echo
 echo "You can use SSH to connect to your instance"
 echo "If the connection attempt times out, you might have to manually update
the SSH ingress rule"
 echo "for your IP address in the AWS Management Console."
 connect_to_instance "$key_file_name" "$public_ip"

 echo -n "Press Enter when you're ready to continue the demo: "
 get_input

 echo_repeat "*" 88
 echo_repeat "*" 88

 echo "Let's stop and start your instance to see what changes."
 echo "Stopping your instance and waiting until it's stopped..."
 ec2_stop_instances -i "$instance_id"
 ec2_wait_for_instance_stopped -i "$instance_id"

 echo "Your instance is stopped. Restarting..."

 ec2_start_instances -i "$instance_id"
 ec2_wait_for_instance_running -i "$instance_id"

 echo "Your instance is running again."
 local instance_details
 instance_details="$(ec2_describe_instances -i "${instance_id}")"

 print_instance_details "${instance_details}"

 public_ip=$(echo "${instance_details}" | awk '{print $6}')

 echo "Every time your instance is restarted, its public IP address changes"
 connect_to_instance "$key_file_name" "$public_ip"

 echo -n "Press Enter when you're ready to continue the demo: "
 get_input

 echo_repeat "*" 88
 echo_repeat "*" 88

 echo "You can allocate an Elastic IP address and associate it with your
instance"
```

```
  echo "to keep a consistent IP address even when your instance restarts."

  local result
  result=$(ec2_allocate_address -d vpc) || {
    errecho "Failed to allocate an address. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id" "$instance_id"
    return 1
  }

  local elastic_ip allocation_id
  elastic_ip=$(echo "$result" | awk '{print $1}')
  allocation_id=$(echo "$result" | awk '{print $2}')

  echo "Allocated static Elastic IP address: $elastic_ip"

  local association_id
  association_id=$(ec2_associate_address -i "$instance_id" -a "$allocation_id")
|| {
    errecho "Failed to associate an address. This demo will exit."
    clean_up "$key_name" "$key_file_name" "$security_group_id" "$instance_id"
"$allocation_id"
    return 1
  }

  echo "Associated your Elastic IP with your instance."
  echo "You can now use SSH to connect to your instance by using the Elastic IP."
  connect_to_instance "$key_file_name" "$elastic_ip"

  echo -n "Press Enter when you're ready to continue the demo: "
  get_input

  echo_repeat "*" 88
  echo_repeat "*" 88

  echo "Let's stop and start your instance to see what changes."
  echo "Stopping your instance and waiting until it's stopped..."
  ec2_stop_instances -i "$instance_id"
  ec2_wait_for_instance_stopped -i "$instance_id"

  echo "Your instance is stopped. Restarting..."

  ec2_start_instances -i "$instance_id"
  ec2_wait_for_instance_running -i "$instance_id"
```

```
  echo "Your instance is running again."
  local instance_details
  instance_details="$(ec2_describe_instances -i "${instance_id}")"

  print_instance_details "${instance_details}"

  echo "Because you have associated an Elastic IP with your instance, you can"
  echo "connect by using a consistent IP address after the instance restarts."
  connect_to_instance "$key_file_name" "$elastic_ip"

  echo -n "Press Enter when you're ready to continue the demo: "
  get_input

  echo_repeat "*" 88
  echo_repeat "*" 88

  if yes_no_input "Do you want to delete the resources created in this demo: (y/
n) "; then
    clean_up "$key_name" "$key_file_name" "$security_group_id" "$instance_id" \
      "$allocation_id" "$association_id"
  else
    echo "The following resources were not deleted."
    echo "Key pair: $key_name"
    echo "Key file: $key_file_name"
    echo "Security group: $security_group_id"
    echo "Instance: $instance_id"
    echo "Elastic IP address: $elastic_ip"
  fi
}

###############################################################################
# function clean_up
#
# This function cleans up the created resources.
#     $1 - The name of the ec2 key pair to delete.
#     $2 - The name of the key file to delete.
#     $3 - The ID of the security group to delete.
#     $4 - The ID of the instance to terminate.
#     $5 - The ID of the elastic IP address to release.
#     $6 - The ID of the elastic IP address to disassociate.
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
```

```
###############################################################################
function clean_up() {
  local result=0
  local key_pair_name=$1
  local key_file_name=$2
  local security_group_id=$3
  local instance_id=$4
  local allocation_id=$5
  local association_id=$6

  if [ -n "$association_id" ]; then
    # bashsupport disable=BP2002
    if (ec2_disassociate_address -a "$association_id"); then
      echo "Disassociated elastic IP address with ID $association_id"
    else
      errecho "The elastic IP address disassociation failed."
      result=1
    fi
  fi

  if [ -n "$allocation_id" ]; then
    # bashsupport disable=BP2002
    if (ec2_release_address -a "$allocation_id"); then
      echo "Released elastic IP address with ID $allocation_id"
    else
      errecho "The elastic IP address release failed."
      result=1
    fi
  fi

  if [ -n "$instance_id" ]; then
    # bashsupport disable=BP2002
    if (ec2_terminate_instances -i "$instance_id"); then
      echo "Started terminating instance with ID $instance_id"

      ec2_wait_for_instance_terminated -i "$instance_id"
    else
      errecho "The instance terminate failed."
      result=1
    fi
  fi

  if [ -n "$security_group_id" ]; then
    # bashsupport disable=BP2002
```

```
    if (ec2_delete_security_group -i "$security_group_id"); then
      echo "Deleted security group with ID $security_group_id"
    else
      errecho "The security group delete failed."
      result=1
    fi
  fi

  if [ -n "$key_pair_name" ]; then
    # bashsupport disable=BP2002
    if (ec2_delete_keypair -n "$key_pair_name"); then
      echo "Deleted key pair named $key_pair_name"
    else
      errecho "The key pair delete failed."
      result=1
    fi
  fi

  if [ -n "$key_file_name" ]; then
    rm -f "$key_file_name"
  fi

  return $result
}

###############################################################################
# function ssm_get_parameters_by_path
#
# This function retrieves one or more parameters from the AWS Systems Manager
 Parameter Store
# by specifying a parameter path.
#
# Parameters:
#       -p parameter_path - The path of the parameter(s) to retrieve.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ssm_get_parameters_by_path() {
  local parameter_path response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
```

```
  function usage() {
    echo "function ssm_get_parameters_by_path"
    echo "Retrieves one or more parameters from the AWS Systems Manager Parameter
 Store by specifying a parameter path."
    echo "  -p parameter_path - The path of the parameter(s) to retrieve."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "p:h" option; do
    case "${option}" in
      p) parameter_path="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$parameter_path" ]]; then
    errecho "ERROR: You must provide a parameter path with the -p parameter."
    usage
    return 1
  fi

  response=$(aws ssm get-parameters-by-path \
    --path "$parameter_path" \
    --query "Parameters[*].[Name, Value]" \
    --output text) || {
    aws_cli_error_log $?
    errecho "ERROR: AWS reports get-parameters-by-path operation failed.
$response"
    return 1
  }

  echo "$response"

  return 0
```

```
}

#############################################################################
# function print_instance_details
#
# This function prints the details of an Amazon Elastic Compute Cloud (Amazon
 EC2) instance.
#
# Parameters:
#       instance_details - The instance details in the format "InstanceId ImageId
 InstanceType KeyName VpcId PublicIpAddress State.Name".
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
#############################################################################
function print_instance_details() {
  local instance_details="$1"

  if [[ -z "${instance_details}" ]]; then
    echo "Error: Missing required instance details argument."
    return 1
  fi

  local instance_id image_id instance_type key_name vpc_id public_ip state
  instance_id=$(echo "${instance_details}" | awk '{print $1}')
  image_id=$(echo "${instance_details}" | awk '{print $2}')
  instance_type=$(echo "${instance_details}" | awk '{print $3}')
  key_name=$(echo "${instance_details}" | awk '{print $4}')
  vpc_id=$(echo "${instance_details}" | awk '{print $5}')
  public_ip=$(echo "${instance_details}" | awk '{print $6}')
  state=$(echo "${instance_details}" | awk '{print $7}')

  echo "    ID: ${instance_id}"
  echo "    Image ID: ${image_id}"
  echo "    Instance type: ${instance_type}"
  echo "    Key name: ${key_name}"
  echo "    VPC ID: ${vpc_id}"
  echo "    Public IP: ${public_ip}"
  echo "    State: ${state}"

  return 0
}
```

```
##############################################################################
# function connect_to_instance
#
# This function displays the public IP address of an Amazon Elastic Compute Cloud
 (Amazon EC2) instance and prompts the user to connect to the instance via SSH.
#
# Parameters:
#       $1 - The name of the key file used to connect to the instance.
#       $2 - The public IP address of the instance.
#
# Returns:
#       None
##############################################################################
function connect_to_instance() {
  local key_file_name="$1"
  local public_ip="$2"

  # Validate the input parameters
  if [[ -z "$key_file_name" ]]; then
    echo "ERROR: You must provide a key file name as the first argument." >&2
    return 1
  fi

  if [[ -z "$public_ip" ]]; then
    echo "ERROR: You must provide a public IP address as the second argument."
 >&2
    return 1
  fi

  # Display the public IP address and connection command
  echo "To connect, run the following command:"
  echo "    ssh -i ${key_file_name} ec2-user@${public_ip}"

  # Prompt the user to connect to the instance
  if yes_no_input "Do you want to connect now? (y/n) "; then
    echo "After you have connected, you can return to this example by typing
 'exit'"
    ssh -i "${key_file_name}" ec2-user@"${public_ip}"
  fi
}

##############################################################################
# function get_input
#
```

```
# This function gets user input from the command line.
#
# Outputs:
#   User input to stdout.
#
# Returns:
#       0
###############################################################################
function get_input() {

  if [ -z "${mock_input+x}" ]; then
    read -r get_input_result
  else

    if [ "$mock_input_array_index" -lt ${#mock_input_array[@]} ]; then
      get_input_result="${mock_input_array[$mock_input_array_index]}"
      # bashsupport disable=BP2001
      # shellcheck disable=SC2206
      ((mock_input_array_index++))
      echo -n "$get_input_result"
    else
      echo "MOCK_INPUT_ARRAY has no more elements" 1>&2
      return 1
    fi
  fi

  return 0
}

###############################################################################
# function yes_no_input
#
# This function requests a yes/no answer from the user, following to a prompt.
#
# Parameters:
#       $1 - The prompt.
#
# Returns:
#       0 - If yes.
#       1 - If no.
###############################################################################
function yes_no_input() {
  if [ -z "$1" ]; then
    echo "Internal error yes_no_input"
```

```
    return 1
  fi

  local index=0
  local response="N"
  while [[ $index -lt 10 ]]; do
    index=$((index + 1))
    echo -n "$1"
    if ! get_input; then
      return 1
    fi
    response=$(echo "$get_input_result" | tr '[:upper:]' '[:lower:]')
    if [ "$response" = "y" ] || [ "$response" = "n" ]; then
      break
    else
      echo -e "\nPlease enter or 'y' or 'n'."
    fi
  done

  echo

  if [ "$response" = "y" ]; then
    return 0
  else
    return 1
  fi
}

###############################################################################
# function integer_input
#
# This function prompts the user to enter an integer within a specified range
# and validates the input.
#
# Parameters:
#       $1 - The prompt message to display to the user.
#       $2 - The minimum value of the accepted range.
#       $3 - The maximum value of the accepted range.
#
# Returns:
#       The valid integer input from the user.
#       If the input is invalid or out of range, the function will continue
#       prompting the user until a valid input is provided.
###############################################################################
```

```
function integer_input() {
  local prompt="$1"
  local min_value="$2"
  local max_value="$3"
  local input=""

  while true; do
    # Display the prompt message and wait for user input
    echo -n "$prompt"

    if ! get_input; then
      return 1
    fi

    input="$get_input_result"

    # Check if the input is a valid integer
    if [[ "$input" =~ ^-?[0-9]+$ ]]; then
      # Check if the input is within the specified range
      if ((input >= min_value && input <= max_value)); then
        return 0
      else
        echo "Error: Input, $input, must be between $min_value and $max_value."
      fi
    else
      echo "Error: Invalid input- $input. Please enter an integer."
    fi
  done
}
################################################################################
# function new_line_and_tab_to_list
#
# This function takes a string input containing newlines and tabs, and
# converts it into a list (array) of elements.
#
# Parameters:
#       $1 - The input string containing newlines and tabs.
#
# Returns:
#       The resulting list (array) is stored in the global variable
#       'list_result'.
################################################################################
function new_line_and_tab_to_list() {
  local input=$1
```

```
  export list_result

  list_result=()
  mapfile -t lines <<<"$input"
  local line
  for line in "${lines[@]}"; do
    IFS=$'\t' read -ra parameters <<<"$line"
    list_result+=("${parameters[@]}")
  done
}

###############################################################################
# function echo_repeat
#
# This function prints a string 'n' times to stdout.
#
# Parameters:
#       $1 - The string.
#       $2 - Number of times to print the string.
#
# Outputs:
#   String 'n' times to stdout.
#
# Returns:
#       0
###############################################################################
function echo_repeat() {
  local end=$2
  for ((i = 0; i < end; i++)); do
    echo -n "$1"
  done
  echo
}
```

이 시나리오에 사용된 DynamoDB 함수입니다.

```
###############################################################################
# function ec2_create_keypair
#
# This function creates an Amazon Elastic Compute Cloud (Amazon EC2) ED25519 or
 2048-bit RSA key pair
# and writes it to a file.
```

```
#
# Parameters:
#       -n key_pair_name - A key pair name.
#       -f file_path - File to store the key pair.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_create_keypair() {
  local key_pair_name file_path response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_create_keypair"
    echo "Creates an Amazon Elastic Compute Cloud (Amazon EC2) ED25519 or 2048-
bit RSA key pair"
    echo " and writes it to a file."
    echo "  -n key_pair_name - A key pair name."
    echo "  -f file_path - File to store the key pair."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:f:h" option; do
    case "${option}" in
      n) key_pair_name="${OPTARG}" ;;
      f) file_path="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$key_pair_name" ]]; then
    errecho "ERROR: You must provide a key name with the -n parameter."
```

```
    usage
    return 1
  fi

  if [[ -z "$file_path" ]]; then
    errecho "ERROR: You must provide a file path with the -f parameter."
    usage
    return 1
  fi

  response=$(aws ec2 create-key-pair \
    --key-name "$key_pair_name" \
    --query 'KeyMaterial' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  }

  if [[ -n "$file_path" ]]; then
    echo "$response" >"$file_path"
  fi

  return 0
}

###############################################################################
# function ec2_describe_key_pairs
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 key pairs.
#
# Parameters:
#       -h - Display help.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_describe_key_pairs() {
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
```

```
    echo "function ec2_describe_key_pairs"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2) key
 pairs."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "h" option; do
    case "${option}" in
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local response

  response=$(aws ec2 describe-key-pairs \
    --query 'KeyPairs[*].[KeyName, KeyFingerprint]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports describe-key-pairs operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}

################################################################################
# function ec2_create_security_group
#
# This function creates an Amazon Elastic Compute Cloud (Amazon EC2) security
 group.
#
```

```
# Parameters:
#       -n security_group_name - The name of the security group.
#       -d security_group_description - The description of the security group.
#
# Returns:
#       The ID of the created security group, or an error message if the
 operation fails.
# And:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
function ec2_create_security_group() {
  local security_group_name security_group_description response

  # Function to display usage information
  function usage() {
    echo "function ec2_create_security_group"
    echo "Creates an Amazon Elastic Compute Cloud (Amazon EC2) security group."
    echo "  -n security_group_name - The name of the security group."
    echo "  -d security_group_description - The description of the security
 group."
    echo ""
  }

  # Parse the command-line arguments
  while getopts "n:d:h" option; do
    case "${option}" in
      n) security_group_name="${OPTARG}" ;;
      d) security_group_description="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Validate the input parameters
```

```
  if [[ -z "$security_group_name" ]]; then
    errecho "ERROR: You must provide a security group name with the -n
 parameter."
    return 1
  fi

  if [[ -z "$security_group_description" ]]; then
    errecho "ERROR: You must provide a security group description with the -d
 parameter."
    return 1
  fi

  # Create the security group
  response=$(aws ec2 create-security-group \
    --group-name "$security_group_name" \
    --description "$security_group_description" \
    --query "GroupId" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports create-security-group operation failed."
    errecho "$response"
    return 1
  }

  echo "$response"
  return 0
}

#############################################################################
# function ec2_describe_security_groups
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 security groups.
#
# Parameters:
#       -g security_group_id - The ID of the security group to describe
 (optional).
#
# And:
#       0 - If successful.
#       1 - If it fails.
#############################################################################
function ec2_describe_security_groups() {
  local security_group_id response
```

```
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_security_groups"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 security groups."
    echo "  -g security_group_id - The ID of the security group to describe
 (optional)."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "g:h" option; do
    case "${option}" in
      g) security_group_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local query="SecurityGroups[*].[GroupName, GroupId, VpcId, IpPermissions[*].
[IpProtocol, FromPort, ToPort, IpRanges[*].CidrIp]]"

  if [[ -n "$security_group_id" ]]; then
    response=$(aws ec2 describe-security-groups --group-ids "$security_group_id"
 --query "${query}" --output text)
  else
    response=$(aws ec2 describe-security-groups --query "${query}" --output text)
  fi

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
```

```
    errecho "ERROR: AWS reports describe-security-groups operation failed.
$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function ec2_authorize_security_group_ingress
#
# This function authorizes an ingress rule for an Amazon Elastic Compute Cloud
 (Amazon EC2) security group.
#
# Parameters:
#       -g security_group_id - The ID of the security group.
#       -i ip_address - The IP address or CIDR block to authorize.
#       -p protocol - The protocol to authorize (e.g., tcp, udp, icmp).
#       -f from_port - The start of the port range to authorize.
#       -t to_port - The end of the port range to authorize.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_authorize_security_group_ingress() {
  local security_group_id ip_address protocol from_port to_port response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_authorize_security_group_ingress"
    echo "Authorizes an ingress rule for an Amazon Elastic Compute Cloud (Amazon
 EC2) security group."
    echo "  -g security_group_id - The ID of the security group."
    echo "  -i ip_address - The IP address or CIDR block to authorize."
    echo "  -p protocol - The protocol to authorize (e.g., tcp, udp, icmp)."
    echo "  -f from_port - The start of the port range to authorize."
    echo "  -t to_port - The end of the port range to authorize."
    echo ""
  }
```

```
  # Retrieve the calling parameters.
  while getopts "g:i:p:f:t:h" option; do
    case "${option}" in
      g) security_group_id="${OPTARG}" ;;
      i) ip_address="${OPTARG}" ;;
      p) protocol="${OPTARG}" ;;
      f) from_port="${OPTARG}" ;;
      t) to_port="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$security_group_id" ]]; then
    errecho "ERROR: You must provide a security group ID with the -g parameter."
    usage
    return 1
  fi

  if [[ -z "$ip_address" ]]; then
    errecho "ERROR: You must provide an IP address or CIDR block with the -i
parameter."
    usage
    return 1
  fi

  if [[ -z "$protocol" ]]; then
    errecho "ERROR: You must provide a protocol with the -p parameter."
    usage
    return 1
  fi

  if [[ -z "$from_port" ]]; then
    errecho "ERROR: You must provide a start port with the -f parameter."
    usage
    return 1
```

```
  fi

  if [[ -z "$to_port" ]]; then
    errecho "ERROR: You must provide an end port with the -t parameter."
    usage
    return 1
  fi

  response=$(aws ec2 authorize-security-group-ingress \
    --group-id "$security_group_id" \
    --cidr "${ip_address}/32" \
    --protocol "$protocol" \
    --port "$from_port-$to_port" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports authorize-security-group-ingress operation
 failed.$response"
    return 1
  }

  return 0
}

################################################################################
# function ec2_describe_images
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 images.
#
# Parameters:
#       -i image_ids - A space-separated  list of image IDs (optional).
#       -h - Display help.
#
# And:
#       0 - If successful.
#       1 - If it fails.
################################################################################
function ec2_describe_images() {
  local image_ids response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_images"
```

```
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 images."
    echo "  -i image_ids - A space-separated list of image IDs (optional)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) image_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local aws_cli_args=()

  if [[ -n "$image_ids" ]]; then
    # shellcheck disable=SC2206
    aws_cli_args+=("--image-ids" $image_ids)
  fi

  response=$(aws ec2 describe-images \
    "${aws_cli_args[@]}" \
    --query 'Images[*].[Description,Architecture,ImageId]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports describe-images operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}
```

```
################################################################################
# ec2_describe_instance_types
#
# This function describes EC2 instance types filtered by processor architecture
# and optionally by instance type. It takes the following arguments:
#
# -a, --architecture ARCHITECTURE  Specify the processor architecture (e.g.,
 x86_64)
# -t, --type INSTANCE_TYPE         Comma-separated list of instance types (e.g.,
 t2.micro)
# -h, --help                       Show the usage help
#
# The function prints the instance type and supported architecture for each
# matching instance type.
################################################################################
function ec2_describe_instance_types() {
  local architecture=""
  local instance_types=""

  # bashsupport disable=BP5008
  function usage() {
    echo "Usage: ec2_describe_instance_types [-a|--architecture ARCHITECTURE] [-
t|--type INSTANCE_TYPE] [-h|--help]"
    echo "  -a, --architecture ARCHITECTURE  Specify the processor architecture
 (e.g., x86_64)"
    echo "  -t, --type INSTANCE_TYPE         Comma-separated list of instance
 types (e.g., t2.micro)"
    echo "  -h, --help                       Show this help message"
  }

  while [[ $# -gt 0 ]]; do
    case "$1" in
      -a | --architecture)
        architecture="$2"
        shift 2
        ;;
      -t | --type)
        instance_types="$2"
        shift 2
        ;;
      -h | --help)
        usage
        return 0
```

```
      ;;
    *)
      echo "Unknown argument: $1"
      return 1
      ;;
  esac
done

if [[ -z "$architecture" ]]; then
  errecho "Error: Architecture not specified."
  usage
  return 1
fi

if [[ -z "$instance_types" ]]; then
  errecho "Error: Instance type not specified."
  usage
  return 1
fi

local tmp_json_file="temp_ec2.json"
echo -n '[
  {
    "Name": "processor-info.supported-architecture",
    "Values": [' >"$tmp_json_file"

local items
IFS=',' read -ra items <<<"$architecture"
local array_size
array_size=${#items[@]}
for i in $(seq 0 $((array_size - 1))); do
  echo -n '"'"${items[$i]}"'"' >>"$tmp_json_file"
  if [[ $i -lt $((array_size - 1)) ]]; then
    echo -n ',' >>"$tmp_json_file"
  fi
done
echo -n ']},
  {
  "Name": "instance-type",
    "Values": [' >>"$tmp_json_file"
IFS=',' read -ra items <<<"$instance_types"
local array_size
array_size=${#items[@]}
for i in $(seq 0 $((array_size - 1))); do
```

```
    echo -n '"'"${items[$i]}"'"' >>"$tmp_json_file"
    if [[ $i -lt $((array_size - 1)) ]]; then
      echo -n ',' >>"$tmp_json_file"
    fi
  done

  echo -n ']}]' >>"$tmp_json_file"

  local response
  response=$(aws ec2 describe-instance-types --filters file://"$tmp_json_file" \
    --query 'InstanceTypes[*].[InstanceType]' --output text)

  local error_code=$?

  rm "$tmp_json_file"

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    echo "ERROR: AWS reports describe-instance-types operation failed."
    return 1
  fi

  echo "$response"
  return 0
}

###############################################################################
# function ec2_run_instances
#
# This function launches one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i image_id - The ID of the Amazon Machine Image (AMI) to use.
#       -t instance_type - The instance type to use (e.g., t2.micro).
#       -k key_pair_name - The name of the key pair to use.
#       -s security_group_id - The ID of the security group to use.
#       -c count - The number of instances to launch (default: 1).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
```

```
function ec2_run_instances() {
  local image_id instance_type key_pair_name security_group_id count response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_run_instances"
    echo "Launches one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances."
    echo "  -i image_id - The ID of the Amazon Machine Image (AMI) to use."
    echo "  -t instance_type - The instance type to use (e.g., t2.micro)."
    echo "  -k key_pair_name - The name of the key pair to use."
    echo "  -s security_group_id - The ID of the security group to use."
    echo "  -c count - The number of instances to launch (default: 1)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:t:k:s:c:h" option; do
    case "${option}" in
      i) image_id="${OPTARG}" ;;
      t) instance_type="${OPTARG}" ;;
      k) key_pair_name="${OPTARG}" ;;
      s) security_group_id="${OPTARG}" ;;
      c) count="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$image_id" ]]; then
    errecho "ERROR: You must provide an Amazon Machine Image (AMI) ID with the -i
 parameter."
    usage
    return 1
```

```
  fi

  if [[ -z "$instance_type" ]]; then
    errecho "ERROR: You must provide an instance type with the -t parameter."
    usage
    return 1
  fi

  if [[ -z "$key_pair_name" ]]; then
    errecho "ERROR: You must provide a key pair name with the -k parameter."
    usage
    return 1
  fi

  if [[ -z "$security_group_id" ]]; then
    errecho "ERROR: You must provide a security group ID with the -s parameter."
    usage
    return 1
  fi

  if [[ -z "$count" ]]; then
    count=1
  fi

  response=$(aws ec2 run-instances \
    --image-id "$image_id" \
    --instance-type "$instance_type" \
    --key-name "$key_pair_name" \
    --security-group-ids "$security_group_id" \
    --count "$count" \
    --query 'Instances[*].[InstanceId]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports run-instances operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}

###############################################################################
# function ec2_describe_instances
```

```
#
# This function describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i instance_id - The ID of the instance to describe (optional).
#       -q query - The query to filter the response (optional).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_describe_instances() {
  local instance_id query response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_describe_instances"
    echo "Describes one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances."
    echo "  -i instance_id - The ID of the instance to describe (optional)."
    echo "  -q query - The query to filter the response (optional)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:q:h" option; do
    case "${option}" in
      i) instance_id="${OPTARG}" ;;
      q) query="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
```

```
  export OPTIND=1

  local aws_cli_args=()

  if [[ -n "$instance_id" ]]; then
    # shellcheck disable=SC2206
    aws_cli_args+=("--instance-ids" $instance_id)
  fi

  local query_arg=""
  if [[ -n "$query" ]]; then
    query_arg="--query '$query'"
  else
    query_arg="--query Reservations[*].Instances[*].
[InstanceId,ImageId,InstanceType,KeyName,VpcId,PublicIpAddress,State.Name]"
  fi

  # shellcheck disable=SC2086
  response=$(aws ec2 describe-instances \
    "${aws_cli_args[@]}" \
    $query_arg \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports describe-instances operation failed.$response"
    return 1
  }

  echo "$response"

  return 0
}

###############################################################################
# function ec2_stop_instances
#
# This function stops one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i instance_id - The ID(s) of the instance(s) to stop (comma-separated).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
```

```
#       1 - If it fails.
###############################################################################
function ec2_stop_instances() {
  local instance_ids
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_stop_instances"
    echo "Stops one or more Amazon Elastic Compute Cloud (Amazon EC2) instances."
    echo "  -i instance_id - The ID(s) of the instance(s) to stop (comma-
separated)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) instance_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$instance_ids" ]]; then
    errecho "ERROR: You must provide one or more instance IDs with the -i
 parameter."
    usage
    return 1
  fi

  response=$(aws ec2 stop-instances \
    --instance-ids "${instance_ids}") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports stop-instances operation failed with $response."
```

```
    return 1
  }

  return 0
}

###############################################################################
# function ec2_start_instances
#
# This function starts one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances.
#
# Parameters:
#       -i instance_id - The ID(s) of the instance(s) to start (comma-separated).
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_start_instances() {
  local instance_ids
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_start_instances"
    echo "Starts one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances."
    echo "  -i instance_id - The ID(s) of the instance(s) to start (comma-
separated)."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) instance_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
```

```
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$instance_ids" ]]; then
    errecho "ERROR: You must provide one or more instance IDs with the -i
 parameter."
    usage
    return 1
  fi

  response=$(aws ec2 start-instances \
    --instance-ids "${instance_ids}") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports start-instances operation failed with $response."
    return 1
  }

  return 0
}

###############################################################################
# function ec2_allocate_address
#
# This function allocates an Elastic IP address for use with Amazon Elastic
 Compute Cloud (Amazon EC2) instances in a specific AWS Region.
#
# Parameters:
#       -d domain - The domain for the Elastic IP address (either 'vpc' or
 'standard').
#
# Returns:
#       The allocated Elastic IP address, or an error message if the operation
 fails.
# And:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
function ec2_allocate_address() {
```

```
 local domain response

 # Function to display usage information
 function usage() {
   echo "function ec2_allocate_address"
   echo "Allocates an Elastic IP address for use with Amazon Elastic Compute
Cloud (Amazon EC2) instances in a specific AWS Region."
   echo "  -d domain - The domain for the Elastic IP address (either 'vpc' or
'standard')."
   echo ""
 }

 # Parse the command-line arguments
 while getopts "d:h" option; do
   case "${option}" in
     d) domain="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 # Validate the input parameters
 if [[ -z "$domain" ]]; then
   errecho "ERROR: You must provide a domain with the -d parameter (either 'vpc'
or 'standard')."
   return 1
 fi

 if [[ "$domain" != "vpc" && "$domain" != "standard" ]]; then
   errecho "ERROR: Invalid domain value. Must be either 'vpc' or 'standard'."
   return 1
 fi

 # Allocate the Elastic IP address
 response=$(aws ec2 allocate-address \
   --domain "$domain" \
```

```
    --query "[PublicIp,AllocationId]" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports allocate-address operation failed."
    errecho "$response"
    return 1
  }

  echo "$response"
  return 0
}

################################################################################
# function ec2_associate_address
#
# This function associates an Elastic IP address with an Amazon Elastic Compute
 Cloud (Amazon EC2) instance.
#
# Parameters:
#       -a allocation_id - The allocation ID of the Elastic IP address to
 associate.
#       -i instance_id - The ID of the EC2 instance to associate the Elastic IP
 address with.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
#
################################################################################
function ec2_associate_address() {
  local allocation_id instance_id response

  # Function to display usage information
  function usage() {
    echo "function ec2_associate_address"
    echo "Associates an Elastic IP address with an Amazon Elastic Compute Cloud
 (Amazon EC2) instance."
    echo "  -a allocation_id - The allocation ID of the Elastic IP address to
 associate."
    echo "  -i instance_id - The ID of the EC2 instance to associate the Elastic
 IP address with."
    echo ""
  }
```

```
  # Parse the command-line arguments
  while getopts "a:i:h" option; do
    case "${option}" in
      a) allocation_id="${OPTARG}" ;;
      i) instance_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Validate the input parameters
  if [[ -z "$allocation_id" ]]; then
    errecho "ERROR: You must provide an allocation ID with the -a parameter."
    return 1
  fi

  if [[ -z "$instance_id" ]]; then
    errecho "ERROR: You must provide an instance ID with the -i parameter."
    return 1
  fi

  # Associate the Elastic IP address
  response=$(aws ec2 associate-address \
    --allocation-id "$allocation_id" \
    --instance-id "$instance_id" \
    --query "AssociationId" \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports associate-address operation failed."
    errecho "$response"
    return 1
  }

  echo "$response"
  return 0
}
```

```
###############################################################################
# function ec2_disassociate_address
#
# This function disassociates an Elastic IP address from an Amazon Elastic
 Compute Cloud (Amazon EC2) instance.
#
# Parameters:
#       -a association_id - The association ID that represents the association of
 the Elastic IP address with an instance.
#
# And:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
function ec2_disassociate_address() {
  local association_id response

  # Function to display usage information
  function usage() {
    echo "function ec2_disassociate_address"
    echo "Disassociates an Elastic IP address from an Amazon Elastic Compute
 Cloud (Amazon EC2) instance."
    echo "  -a association_id - The association ID that represents the
 association of the Elastic IP address with an instance."
    echo ""
  }

  # Parse the command-line arguments
  while getopts "a:h" option; do
    case "${option}" in
      a) association_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
```

```
  export OPTIND=1

  # Validate the input parameters
  if [[ -z "$association_id" ]]; then
    errecho "ERROR: You must provide an association ID with the -a parameter."
    return 1
  fi

  response=$(aws ec2 disassociate-address \
    --association-id "$association_id") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports disassociate-address operation failed."
    errecho "$response"
    return 1
  }

  return 0
}

###############################################################################
# function ec2_release_address
#
# This function releases an Elastic IP address from an Amazon Elastic Compute
 Cloud (Amazon EC2) instance.
#
# Parameters:
#       -a allocation_id - The allocation ID of the Elastic IP address to
 release.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
#
###############################################################################
function ec2_release_address() {
  local allocation_id response

  # Function to display usage information
  function usage() {
    echo "function ec2_release_address"
    echo "Releases an Elastic IP address from an Amazon Elastic Compute Cloud
 (Amazon EC2) instance."
    echo "  -a allocation_id - The allocation ID of the Elastic IP address to
 release."
```

```
    echo ""
  }

  # Parse the command-line arguments
  while getopts "a:h" option; do
    case "${option}" in
      a) allocation_id="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Validate the input parameters
  if [[ -z "$allocation_id" ]]; then
    errecho "ERROR: You must provide an allocation ID with the -a parameter."
    return 1
  fi

  response=$(aws ec2 release-address \
    --allocation-id "$allocation_id") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports release-address operation failed."
    errecho "$response"
    return 1
  }

  return 0
}

################################################################################
# function ec2_terminate_instances
#
# This function terminates one or more Amazon Elastic Compute Cloud (Amazon EC2)
# instances using the AWS CLI.
#
# Parameters:
```

```
#       -i instance_ids - A space-separated list of instance IDs.
#       -h - Display help.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_terminate_instances() {
  local instance_ids response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_terminate_instances"
    echo "Terminates one or more Amazon Elastic Compute Cloud (Amazon EC2)
 instances."
    echo "  -i instance_ids - A space-separated list of instance IDs."
    echo "  -h - Display help."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) instance_ids="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  # Check if instance ID is provided
  if [[ -z "${instance_ids}" ]]; then
    echo "Error: Missing required instance IDs parameter."
    usage
    return 1
  fi
```

```
  # shellcheck disable=SC2086
  response=$(aws ec2 terminate-instances \
    "--instance-ids" $instance_ids \
    --query 'TerminatingInstances[*].[InstanceId,CurrentState.Name]' \
    --output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports terminate-instances operation failed.$response"
    return 1
  }

  return 0
}

###############################################################################
# function ec2_delete_security_group
#
# This function deletes an Amazon Elastic Compute Cloud (Amazon EC2) security
 group.
#
# Parameters:
#       -i security_group_id - The ID of the security group to delete.
#
# And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function ec2_delete_security_group() {
  local security_group_id response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_delete_security_group"
    echo "Deletes an Amazon Elastic Compute Cloud (Amazon EC2) security group."
    echo "  -i security_group_id - The ID of the security group to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "i:h" option; do
    case "${option}" in
      i) security_group_id="${OPTARG}" ;;
      h)
```

```
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$security_group_id" ]]; then
    errecho "ERROR: You must provide a security group ID with the -i parameter."
    usage
    return 1
  fi

  response=$(aws ec2 delete-security-group --group-id "$security_group_id" --
output text) || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports delete-security-group operation failed.$response"
    return 1
  }

  return 0
}

################################################################################
# function ec2_delete_keypair
#
# This function deletes an Amazon EC2 ED25519 or 2048-bit RSA key pair.
#
# Parameters:
#       -n key_pair_name - A key pair name.
#
# And:
#       0 - If successful.
#       1 - If it fails.
################################################################################
function ec2_delete_keypair() {
  local key_pair_name response

  local option OPTARG # Required to use getopts command in a function.
```

```
  # bashsupport disable=BP5008
  function usage() {
    echo "function ec2_delete_keypair"
    echo "Deletes an Amazon EC2 ED25519 or 2048-bit RSA key pair."
    echo "  -n key_pair_name - A key pair name."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) key_pair_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$key_pair_name" ]]; then
    errecho "ERROR: You must provide a key pair name with the -n parameter."
    usage
    return 1
  fi

  response=$(aws ec2 delete-key-pair \
    --key-name "$key_pair_name") || {
    aws_cli_error_log ${?}
    errecho "ERROR: AWS reports delete-key-pair operation failed.$response"
    return 1
  }

  return 0
}
```

이 시나리오에 사용된 유틸리티 함수입니다.

```
#############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function aws_cli_error_log()
#
# This function is used to log the error messages from the AWS CLI.
#
# The function expects the following argument:
#         $1 - The error code returned by the AWS CLI.
#
#  Returns:
#          0: - Success.
#
#############################################################################
function aws_cli_error_log() {
  local err_code=$1
  errecho "Error code : $err_code"
  if [ "$err_code" == 1 ]; then
    errecho "  One or more S3 transfers failed."
  elif [ "$err_code" == 2 ]; then
    errecho "  Command line failed to parse."
  elif [ "$err_code" == 130 ]; then
    errecho "  Process received SIGINT."
  elif [ "$err_code" == 252 ]; then
    errecho "  Command syntax invalid."
  elif [ "$err_code" == 253 ]; then
    errecho "  The system environment or configuration was invalid."
  elif [ "$err_code" == 254 ]; then
    errecho "  The service returned an error."
  elif [ "$err_code" == 255 ]; then
    errecho "  255 is a catch-all error."
  fi

  return 0
}
```

- API 세부 정보는 AWS CLI 명령 참조의 다음 주제를 참조하십시오.
  - AllocateAddress
  - AssociateAddress
  - AuthorizeSecurityGroupIngress
  - CreateKeyPair
  - CreateSecurityGroup
  - DeleteKeyPair
  - DeleteSecurityGroup
  - DescribeImages
  - DescribeInstanceTypes
  - DescribeInstances
  - DescribeKeyPairs
  - DescribeSecurityGroups
  - DisassociateAddress
  - ReleaseAddress
  - RunInstances
  - StartInstances
  - StopInstances
  - TerminateInstances
  - UnmonitorInstances

Java

SDK for Java 2.x

> **Note**
>
> 더 많은 것이 있어요 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
/**
 * 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
 *
 * This Java example performs the following tasks:
 *
 * 1. Creates an RSA key pair and saves the private key data as a .pem file.
 * 2. Lists key pairs.
 * 3. Creates a security group for the default VPC.
 * 4. Displays security group information.
 * 5. Gets a list of Amazon Linux 2 AMIs and selects one.
 * 6. Gets more information about the image.
 * 7. Gets a list of instance types that are compatible with the selected AMI's
 * architecture.
 * 8. Creates an instance with the key pair, security group, AMI, and an
 * instance type.
 * 9. Displays information about the instance.
 * 10. Stops the instance and waits for it to stop.
 * 11. Starts the instance and waits for it to start.
 * 12. Allocates an Elastic IP address and associates it with the instance.
 * 13. Displays SSH connection info for the instance.
 * 14. Disassociates and deletes the Elastic IP address.
 * 15. Terminates the instance and waits for it to terminate.
 * 16. Deletes the security group.
 * 17. Deletes the key pair.
 */
public class EC2Scenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");

    public static void main(String[] args) throws InterruptedException {

        final String usage = """

                Usage:
                   <keyName> <fileName> <groupName> <groupDesc> <vpcId>

                Where:
                   keyName -  A key pair name (for example, TestKeyPair).\s
                   fileName -  A file name where the key information is written
 to.\s
```

```
                groupName - The name of the security group.\s
                groupDesc - The description of the security group.\s
                vpcId - A VPC Id value. You can get this value from the AWS
Management Console.\s
                myIpAddress - The IP address of your development machine.\s

               """;

       if (args.length != 6) {
           System.out.println(usage);
           System.exit(1);
       }

       String keyName = args[0];
       String fileName = args[1];
       String groupName = args[2];
       String groupDesc = args[3];
       String vpcId = args[4];
       String myIpAddress = args[5];

       Region region = Region.US_WEST_2;
       Ec2Client ec2 = Ec2Client.builder()
               .region(region)
               .build();

       SsmClient ssmClient = SsmClient.builder()
               .region(region)
               .build();

       System.out.println(DASHES);
       System.out.println("Welcome to the Amazon EC2 example scenario.");
       System.out.println(DASHES);

       System.out.println(DASHES);
       System.out.println("1. Create an RSA key pair and save the private key
material as a .pem file.");
       createKeyPair(ec2, keyName, fileName);
       System.out.println(DASHES);

       System.out.println(DASHES);
       System.out.println("2. List key pairs.");
       describeKeys(ec2);
       System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println("3. Create a security group.");
        String groupId = createSecurityGroup(ec2, groupName, groupDesc, vpcId,
myIpAddress);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Display security group info for the newly created
security group.");
        describeSecurityGroups(ec2, groupId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("5. Get a list of Amazon Linux 2 AMIs and selects one
with amzn2 in the name.");
        String instanceId = getParaValues(ssmClient);
        System.out.println("The instance Id is " + instanceId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6. Get more information about an amzn2 image.");
        String amiValue = describeImage(ec2, instanceId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("7. Get a list of instance types.");
        String instanceType = getInstanceTypes(ec2);
        System.out.println("The instance type is " + instanceType);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("8. Create an instance.");
        String newInstanceId = runInstance(ec2, instanceType, keyName, groupName,
amiValue);
        System.out.println("The instance Id is " + newInstanceId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("9. Display information about the running instance.
");
        String ipAddress = describeEC2Instances(ec2, newInstanceId);
        System.out.println("You can SSH to the instance using this command:");
        System.out.println("ssh -i " + fileName + "ec2-user@" + ipAddress);
        System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println("10. Stop the instance and use a waiter.");
        stopInstance(ec2, newInstanceId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("11. Start the instance and use a waiter.");
        startInstance(ec2, newInstanceId);
        ipAddress = describeEC2Instances(ec2, newInstanceId);
        System.out.println("You can SSH to the instance using this command:");
        System.out.println("ssh -i " + fileName + "ec2-user@" + ipAddress);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("12. Allocate an Elastic IP address and associate it
with the instance.");
        String allocationId = allocateAddress(ec2);
        System.out.println("The allocation Id value is " + allocationId);
        String associationId = associateAddress(ec2, newInstanceId,
allocationId);
        System.out.println("The associate Id value is " + associationId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("13. Describe the instance again.");
        ipAddress = describeEC2Instances(ec2, newInstanceId);
        System.out.println("You can SSH to the instance using this command:");
        System.out.println("ssh -i " + fileName + "ec2-user@" + ipAddress);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("14. Disassociate and release the Elastic IP
address.");
        disassociateAddress(ec2, associationId);
        releaseEC2Address(ec2, allocationId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("15. Terminate the instance and use a waiter.");
        terminateEC2(ec2, newInstanceId);
        System.out.println(DASHES);

        System.out.println(DASHES);
```

```
        System.out.println("16. Delete the security group.");
        deleteEC2SecGroup(ec2, groupId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("17. Delete the key.");
        deleteKeys(ec2, keyName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("You successfully completed the Amazon EC2
scenario.");
        System.out.println(DASHES);
        ec2.close();
    }

    public static void deleteEC2SecGroup(Ec2Client ec2, String groupId) {
        try {
            DeleteSecurityGroupRequest request =
DeleteSecurityGroupRequest.builder()
                    .groupId(groupId)
                    .build();

            ec2.deleteSecurityGroup(request);
            System.out.println("Successfully deleted security group with Id " +
groupId);

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void terminateEC2(Ec2Client ec2, String instanceId) {
        try {
            Ec2Waiter ec2Waiter = Ec2Waiter.builder()
                    .overrideConfiguration(b -> b.maxAttempts(100))
                    .client(ec2)
                    .build();

            TerminateInstancesRequest ti = TerminateInstancesRequest.builder()
                    .instanceIds(instanceId)
                    .build();
```

```
            System.out.println("Use an Ec2Waiter to wait for the instance to
terminate. This will take a few minutes.");
            ec2.terminateInstances(ti);
            DescribeInstancesRequest instanceRequest =
DescribeInstancesRequest.builder()
                    .instanceIds(instanceId)
                    .build();

            WaiterResponse<DescribeInstancesResponse> waiterResponse = ec2Waiter
                    .waitUntilInstanceTerminated(instanceRequest);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Successfully started instance " + instanceId);
            System.out.println(instanceId + " is terminated!");

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteKeys(Ec2Client ec2, String keyPair) {
        try {
            DeleteKeyPairRequest request = DeleteKeyPairRequest.builder()
                    .keyName(keyPair)
                    .build();

            ec2.deleteKeyPair(request);
            System.out.println("Successfully deleted key pair named " + keyPair);

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void releaseEC2Address(Ec2Client ec2, String allocId) {
        try {
            ReleaseAddressRequest request = ReleaseAddressRequest.builder()
                    .allocationId(allocId)
                    .build();

            ec2.releaseAddress(request);
            System.out.println("Successfully released Elastic IP address " +
allocId);
```

```
        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void disassociateAddress(Ec2Client ec2, String associationId) {
        try {
            DisassociateAddressRequest addressRequest =
DisassociateAddressRequest.builder()
                    .associationId(associationId)
                    .build();

            ec2.disassociateAddress(addressRequest);
            System.out.println("You successfully disassociated the address!");

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static String associateAddress(Ec2Client ec2, String instanceId,
String allocationId) {
        try {
            AssociateAddressRequest associateRequest =
AssociateAddressRequest.builder()
                    .instanceId(instanceId)
                    .allocationId(allocationId)
                    .build();

            AssociateAddressResponse associateResponse =
ec2.associateAddress(associateRequest);
            return associateResponse.associationId();

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static String allocateAddress(Ec2Client ec2) {
        try {
```

```
            AllocateAddressRequest allocateRequest =
AllocateAddressRequest.builder()
                    .domain(DomainType.VPC)
                    .build();

            AllocateAddressResponse allocateResponse =
ec2.allocateAddress(allocateRequest);
            return allocateResponse.allocationId();

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static void startInstance(Ec2Client ec2, String instanceId) {
        Ec2Waiter ec2Waiter = Ec2Waiter.builder()
                .overrideConfiguration(b -> b.maxAttempts(100))
                .client(ec2)
                .build();

        StartInstancesRequest request = StartInstancesRequest.builder()
                .instanceIds(instanceId)
                .build();

        System.out.println("Use an Ec2Waiter to wait for the instance to run.
This will take a few minutes.");
        ec2.startInstances(request);
        DescribeInstancesRequest instanceRequest =
DescribeInstancesRequest.builder()
                .instanceIds(instanceId)
                .build();

        WaiterResponse<DescribeInstancesResponse> waiterResponse =
ec2Waiter.waitUntilInstanceRunning(instanceRequest);
        waiterResponse.matched().response().ifPresent(System.out::println);
        System.out.println("Successfully started instance " + instanceId);
    }

    public static void stopInstance(Ec2Client ec2, String instanceId) {
        Ec2Waiter ec2Waiter = Ec2Waiter.builder()
                .overrideConfiguration(b -> b.maxAttempts(100))
                .client(ec2)
```

```
                .build();
        StopInstancesRequest request = StopInstancesRequest.builder()
                .instanceIds(instanceId)
                .build();

        System.out.println("Use an Ec2Waiter to wait for the instance to stop.
This will take a few minutes.");
        ec2.stopInstances(request);
        DescribeInstancesRequest instanceRequest =
DescribeInstancesRequest.builder()
                .instanceIds(instanceId)
                .build();

        WaiterResponse<DescribeInstancesResponse> waiterResponse =
ec2Waiter.waitUntilInstanceStopped(instanceRequest);
        waiterResponse.matched().response().ifPresent(System.out::println);
        System.out.println("Successfully stopped instance " + instanceId);
    }

    public static String describeEC2Instances(Ec2Client ec2, String
newInstanceId) {
        try {
            String pubAddress = "";
            boolean isRunning = false;
            DescribeInstancesRequest request = DescribeInstancesRequest.builder()
                    .instanceIds(newInstanceId)
                    .build();

            while (!isRunning) {
                DescribeInstancesResponse response =
ec2.describeInstances(request);
                String state =
response.reservations().get(0).instances().get(0).state().name().name();
                if (state.compareTo("RUNNING") == 0) {
                    System.out.println("Image id is " +
response.reservations().get(0).instances().get(0).imageId());
                    System.out.println(
                            "Instance type is " +
response.reservations().get(0).instances().get(0).instanceType());
                    System.out.println(
                            "Instance state is " +
response.reservations().get(0).instances().get(0).state().name());
                    pubAddress =
response.reservations().get(0).instances().get(0).publicIpAddress();
```

```
                    System.out.println("Instance address is " + pubAddress);
                    isRunning = true;
                }
            }
            return pubAddress;
        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
        return "";
    }

    public static String runInstance(Ec2Client ec2, String instanceType, String
keyName, String groupName,
            String amiId) {
        try {
            RunInstancesRequest runRequest = RunInstancesRequest.builder()
                    .instanceType(instanceType)
                    .keyName(keyName)
                    .securityGroups(groupName)
                    .maxCount(1)
                    .minCount(1)
                    .imageId(amiId)
                    .build();

            System.out.println("Going to start an EC2 instance using a waiter");
            RunInstancesResponse response = ec2.runInstances(runRequest);
            String instanceIdVal = response.instances().get(0).instanceId();
            ec2.waiter().waitUntilInstanceRunning(r ->
r.instanceIds(instanceIdVal));
            System.out.println("Successfully started EC2 instance " +
instanceIdVal + " based on AMI " + amiId);
            return instanceIdVal;

        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
        return "";
    }

    // Get a list of instance types.
    public static String getInstanceTypes(Ec2Client ec2) {
        String instanceType;
```

```
        try {
            DescribeInstanceTypesRequest typesRequest =
DescribeInstanceTypesRequest.builder()
                .maxResults(10)
                .build();

            DescribeInstanceTypesResponse response =
ec2.describeInstanceTypes(typesRequest);
            List<InstanceTypeInfo> instanceTypes = response.instanceTypes();
            for (InstanceTypeInfo type : instanceTypes) {
                System.out.println("The memory information of this type is " +
type.memoryInfo().sizeInMiB());
                System.out.println("Network information is " +
type.networkInfo().toString());
                System.out.println("Instance type is " +
type.instanceType().toString());
                instanceType = type.instanceType().toString();
                if (instanceType.compareTo("t2.2xlarge")  == 0){
                    return instanceType;
                }
            }

        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
        return "";
    }

    // Display the Description field that corresponds to the instance Id value.
    public static String describeImage(Ec2Client ec2, String instanceId) {
        try {
            DescribeImagesRequest imagesRequest = DescribeImagesRequest.builder()
                    .imageIds(instanceId)
                    .build();

            DescribeImagesResponse response = ec2.describeImages(imagesRequest);
            System.out.println("The description of the first image is " +
response.images().get(0).description());
            System.out.println("The name of the first image is " +
response.images().get(0).name());

            // Return the image Id value.
            return response.images().get(0).imageId();
```

```
        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
        return "";
    }

    // Get the Id value of an instance with amzn2 in the name.
    public static String getParaValues(SsmClient ssmClient) {
        try {
            GetParametersByPathRequest parameterRequest =
GetParametersByPathRequest.builder()
                    .path("/aws/service/ami-amazon-linux-latest")
                    .build();

            GetParametersByPathIterable responses =
ssmClient.getParametersByPathPaginator(parameterRequest);
            for
(software.amazon.awssdk.services.ssm.model.GetParametersByPathResponse
response : responses) {
                System.out.println("Test " + response.nextToken());
                List<Parameter> parameterList = response.parameters();
                for (Parameter para : parameterList) {
                    System.out.println("The name of the para is: " +
para.name());
                    System.out.println("The type of the para is: " +
para.type());
                    if (filterName(para.name())) {
                        return para.value();
                    }
                }
            }

        } catch (SsmException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
        return "";
    }

    // Return true if the name has amzn2 in it. For example:
    // /aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-arm64-gp2
    private static boolean filterName(String name) {
```

```
        String[] parts = name.split("/");
        String myValue = parts[4];
        return myValue.contains("amzn2");
    }

    public static void describeSecurityGroups(Ec2Client ec2, String groupId) {
        try {
            DescribeSecurityGroupsRequest request =
DescribeSecurityGroupsRequest.builder()
                .groupIds(groupId)
                .build();

            // Use a paginator.
            DescribeSecurityGroupsIterable listGroups =
ec2.describeSecurityGroupsPaginator(request);
            listGroups.stream()
                .flatMap(r -> r.securityGroups().stream())
                .forEach(group -> System.out
                    .println(" Group id: " +group.groupId() + " group name = " +
group.groupName()));

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static String createSecurityGroup(Ec2Client ec2, String groupName,
String groupDesc, String vpcId,
            String myIpAddress) {
        try {
            CreateSecurityGroupRequest createRequest =
CreateSecurityGroupRequest.builder()
                    .groupName(groupName)
                    .description(groupDesc)
                    .vpcId(vpcId)
                    .build();

            CreateSecurityGroupResponse resp =
ec2.createSecurityGroup(createRequest);
            IpRange ipRange = IpRange.builder()
                    .cidrIp(myIpAddress + "/0")
                    .build();
```

```
            IpPermission ipPerm = IpPermission.builder()
                    .ipProtocol("tcp")
                    .toPort(80)
                    .fromPort(80)
                    .ipRanges(ipRange)
                    .build();

            IpPermission ipPerm2 = IpPermission.builder()
                    .ipProtocol("tcp")
                    .toPort(22)
                    .fromPort(22)
                    .ipRanges(ipRange)
                    .build();

            AuthorizeSecurityGroupIngressRequest authRequest =
AuthorizeSecurityGroupIngressRequest.builder()
                    .groupName(groupName)
                    .ipPermissions(ipPerm, ipPerm2)
                    .build();

            ec2.authorizeSecurityGroupIngress(authRequest);
            System.out.println("Successfully added ingress policy to security
group " + groupName);
            return resp.groupId();

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static void describeKeys(Ec2Client ec2) {
        try {
            DescribeKeyPairsResponse response = ec2.describeKeyPairs();
            response.keyPairs().forEach(keyPair -> System.out.printf(
                    "Found key pair with name %s " +
                            "and fingerprint %s",
                    keyPair.keyName(),
                    keyPair.keyFingerprint()));

        } catch (Ec2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
```

```
        }
    }

    public static void createKeyPair(Ec2Client ec2, String keyName, String
 fileName) {
        try {
            CreateKeyPairRequest request = CreateKeyPairRequest.builder()
                    .keyName(keyName)
                    .build();

            CreateKeyPairResponse response = ec2.createKeyPair(request);
            String content = response.keyMaterial();
            BufferedWriter writer = new BufferedWriter(new FileWriter(fileName));
            writer.write(content);
            writer.close();
            System.out.println("Successfully created key pair named " + keyName);

        } catch (Ec2Exception | IOException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- API 세부 정보는 AWS SDK for Java 2.x API 참조의 다음 주제를 참조하십시오.
  - AllocateAddress
  - AssociateAddress
  - AuthorizeSecurityGroupIngress
  - CreateKeyPair
  - CreateSecurityGroup
  - DeleteKeyPair
  - DeleteSecurityGroup
  - DescribeImages
  - DescribeInstanceTypes
  - DescribeInstances
  - DescribeKeyPairs
  - DescribeSecurityGroups

- DisassociateAddress
- ReleaseAddress
- RunInstances
- StartInstances
- StopInstances
- TerminateInstances
- UnmonitorInstances

JavaScript

JavaScript (v3) 용 SDK

> **Note**
>
> 더 많은 내용이 있습니다. GitHub AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

명령 프롬프트에서 대화형 시나리오를 실행합니다.

```
import { mkdtempSync, writeFileSync, rmSync } from "fs";
import { tmpdir } from "os";
import { join } from "path";
import { get } from "http";

import {
  AllocateAddressCommand,
  AssociateAddressCommand,
  AuthorizeSecurityGroupIngressCommand,
  CreateKeyPairCommand,
  CreateSecurityGroupCommand,
  DeleteKeyPairCommand,
  DeleteSecurityGroupCommand,
  DescribeInstancesCommand,
  DescribeKeyPairsCommand,
  DescribeSecurityGroupsCommand,
  DisassociateAddressCommand,
  EC2Client,
  paginateDescribeImages,
```

```
  paginateDescribeInstanceTypes,
  ReleaseAddressCommand,
  RunInstancesCommand,
  StartInstancesCommand,
  StopInstancesCommand,
  TerminateInstancesCommand,
  waitUntilInstanceStatusOk,
  waitUntilInstanceStopped,
  waitUntilInstanceTerminated,
} from "@aws-sdk/client-ec2";
import { paginateGetParametersByPath, SSMClient } from "@aws-sdk/client-ssm";

import { wrapText } from "@aws-doc-sdk-examples/lib/utils/util-string.js";
import { Prompter } from "@aws-doc-sdk-examples/lib/prompter.js";

const ec2Client = new EC2Client();
const ssmClient = new SSMClient();

const prompter = new Prompter();
const confirmMessage = "Continue?";
const tmpDirectory = mkdtempSync(join(tmpdir(), "ec2-scenario-tmp"));

const createKeyPair = async (keyPairName) => {
  // Create a key pair in Amazon EC2.
  const { KeyMaterial, KeyPairId } = await ec2Client.send(
    // A unique name for the key pair. Up to 255 ASCII characters.
    new CreateKeyPairCommand({ KeyName: keyPairName }),
  );

  // Save the private key in a temporary location.
  writeFileSync(`${tmpDirectory}/${keyPairName}.pem`, KeyMaterial, {
    mode: 0o400,
  });

  return KeyPairId;
};

const describeKeyPair = async (keyPairName) => {
  const command = new DescribeKeyPairsCommand({
    KeyNames: [keyPairName],
  });
  const { KeyPairs } = await ec2Client.send(command);
  return KeyPairs[0];
};
```

```
const createSecurityGroup = async (securityGroupName) => {
  const command = new CreateSecurityGroupCommand({
    GroupName: securityGroupName,
    Description: "A security group for the Amazon EC2 example.",
  });
  const { GroupId } = await ec2Client.send(command);
  return GroupId;
};

const allocateIpAddress = async () => {
  const command = new AllocateAddressCommand({});
  const { PublicIp, AllocationId } = await ec2Client.send(command);
  return { PublicIp, AllocationId };
};

const getLocalIpAddress = () => {
  return new Promise((res, rej) => {
    get("http://checkip.amazonaws.com", (response) => {
      let data = "";
      response.on("data", (chunk) => (data += chunk));
      response.on("end", () => res(data.trim()));
    }).on("error", (err) => {
      rej(err);
    });
  });
};

const authorizeSecurityGroupIngress = async (securityGroupId) => {
  const ipAddress = await getLocalIpAddress();
  const command = new AuthorizeSecurityGroupIngressCommand({
    GroupId: securityGroupId,
    IpPermissions: [
      {
        IpProtocol: "tcp",
        FromPort: 22,
        ToPort: 22,
        IpRanges: [{ CidrIp: `${ipAddress}/32` }],
      },
    ],
  });

  await ec2Client.send(command);
  return ipAddress;
```

```
};

const describeSecurityGroup = async (securityGroupName) => {
  const command = new DescribeSecurityGroupsCommand({
    GroupNames: [securityGroupName],
  });
  const { SecurityGroups } = await ec2Client.send(command);

  return SecurityGroups[0];
};

const getAmznLinux2AMIs = async () => {
  const AMIs = [];
  for await (const page of paginateGetParametersByPath(
    {
      client: ssmClient,
    },
    { Path: "/aws/service/ami-amazon-linux-latest" },
  )) {
    page.Parameters.forEach((param) => {
      if (param.Name.includes("amzn2")) {
        AMIs.push(param.Value);
      }
    });
  }

  const imageDetails = [];

  for await (const page of paginateDescribeImages(
    { client: ec2Client },
    { ImageIds: AMIs },
  )) {
    imageDetails.push(...(page.Images || []));
  }

  const choices = imageDetails.map((image, index) => ({
    name: `${image.ImageId} - ${image.Description}`,
    value: index,
  }));

  /**
   * @type {number}
   */
  const selectedIndex = await prompter.select({
```

```
    message: "Select an image.",
    choices,
  });

  return imageDetails[selectedIndex];
};

/**
 * @param {import('@aws-sdk/client-ec2').Image} imageDetails
 */
const getCompatibleInstanceTypes = async (imageDetails) => {
  const paginator = paginateDescribeInstanceTypes(
    { client: ec2Client, pageSize: 25 },
    {
      Filters: [
        {
          Name: "processor-info.supported-architecture",
          Values: [imageDetails.Architecture],
        },
        { Name: "instance-type", Values: ["*.micro", "*.small"] },
      ],
    },
  );

  const instanceTypes = [];

  for await (const page of paginator) {
    if (page.InstanceTypes.length) {
      instanceTypes.push(...(page.InstanceTypes || []));
    }
  }

  const choices = instanceTypes.map((type, index) => ({
    name: `${type.InstanceType} - Memory:${type.MemoryInfo.SizeInMiB}`,
    value: index,
  }));

  /**
   * @type {number}
   */
  const selectedIndex = await prompter.select({
    message: "Select an instance type.",
    choices,
  });
```

```
  return instanceTypes[selectedIndex];
};

const runInstance = async ({
  keyPairName,
  securityGroupId,
  imageId,
  instanceType,
}) => {
  const command = new RunInstancesCommand({
    KeyName: keyPairName,
    SecurityGroupIds: [securityGroupId],
    ImageId: imageId,
    InstanceType: instanceType,
    MinCount: 1,
    MaxCount: 1,
  });

  const { Instances } = await ec2Client.send(command);
  await waitUntilInstanceStatusOk(
    { client: ec2Client },
    { InstanceIds: [Instances[0].InstanceId] },
  );
  return Instances[0].InstanceId;
};

const describeInstance = async (instanceId) => {
  const command = new DescribeInstancesCommand({
    InstanceIds: [instanceId],
  });

  const { Reservations } = await ec2Client.send(command);
  return Reservations[0].Instances[0];
};

const displaySSHConnectionInfo = ({ publicIp, keyPairName }) => {
  return `ssh -i ${tmpDirectory}/${keyPairName}.pem ec2-user@${publicIp}`;
};

const stopInstance = async (instanceId) => {
  const command = new StopInstancesCommand({ InstanceIds: [instanceId] });
  await ec2Client.send(command);
  await waitUntilInstanceStopped(
    { client: ec2Client },
```

```
    { InstanceIds: [instanceId] },
  );
};

const startInstance = async (instanceId) => {
  const startCommand = new StartInstancesCommand({ InstanceIds: [instanceId] });
  await ec2Client.send(startCommand);
  await waitUntilInstanceStatusOk(
    { client: ec2Client },
    { InstanceIds: [instanceId] },
  );
  return await describeInstance(instanceId);
};

const associateAddress = async ({ allocationId, instanceId }) => {
  const command = new AssociateAddressCommand({
    AllocationId: allocationId,
    InstanceId: instanceId,
  });

  const { AssociationId } = await ec2Client.send(command);
  return AssociationId;
};

const disassociateAddress = async (associationId) => {
  const command = new DisassociateAddressCommand({
    AssociationId: associationId,
  });
  try {
    await ec2Client.send(command);
  } catch (err) {
    console.warn(
      `Failed to disassociated address with association id: ${associationId}`,
      err,
    );
  }
};

const releaseAddress = async (allocationId) => {
  const command = new ReleaseAddressCommand({
    AllocationId: allocationId,
  });

  try {
```

```
    await ec2Client.send(command);
    console.log(`Address with allocation ID ${allocationId} released.\n`);
  } catch (err) {
    console.log(
      `Failed to release address with allocation id: ${allocationId}.`,
      err,
    );
  }
};

const restartInstance = async (instanceId) => {
  console.log("Stopping instance.");
  await stopInstance(instanceId);
  console.log("Instance stopped.");
  console.log("Starting instance.");
  const { PublicIpAddress } = await startInstance(instanceId);
  return PublicIpAddress;
};

const terminateInstance = async (instanceId) => {
  const command = new TerminateInstancesCommand({
    InstanceIds: [instanceId],
  });

  try {
    await ec2Client.send(command);
    await waitUntilInstanceTerminated(
      { client: ec2Client },
      { InstanceIds: [instanceId] },
    );
    console.log(`Instance with ID ${instanceId} terminated.\n`);
  } catch (err) {
    console.warn(`Failed to terminate instance ${instanceId}.`, err);
  }
};

const deleteSecurityGroup = async (securityGroupId) => {
  const command = new DeleteSecurityGroupCommand({
    GroupId: securityGroupId,
  });

  try {
    await ec2Client.send(command);
    console.log(`Security group ${securityGroupId} deleted.\n`);
```

```
  } catch (err) {
    console.warn(`Failed to delete security group ${securityGroupId}.`, err);
  }
};

const deleteKeyPair = async (keyPairName) => {
  const command = new DeleteKeyPairCommand({
    KeyName: keyPairName,
  });

  try {
    await ec2Client.send(command);
    console.log(`Key pair ${keyPairName} deleted.\n`);
  } catch (err) {
    console.warn(`Failed to delete key pair ${keyPairName}.`, err);
  }
};

const deleteTemporaryDirectory = () => {
  try {
    rmSync(tmpDirectory, { recursive: true });
    console.log(`Temporary directory ${tmpDirectory} deleted.\n`);
  } catch (err) {
    console.warn(`Failed to delete temporary directory ${tmpDirectory}.`, err);
  }
};

export const main = async () => {
  const keyPairName = "ec2-scenario-key-pair";
  const securityGroupName = "ec2-scenario-security-group";

  let securityGroupId, ipAllocationId, publicIp, instanceId, associationId;

  console.log(wrapText("Welcome to the Amazon EC2 basic usage scenario."));

  try {
    // Prerequisites
    console.log(
      "Before you launch an instance, you'll need a few things:",
      "\n - A Key Pair",
      "\n - A Security Group",
      "\n - An IP Address",
      "\n - An AMI",
      "\n - A compatible instance type",
```

```
      "\n\n I'll go ahead and take care of the first three, but I'll need your
help for the rest.",
    );

    await prompter.confirm({ message: confirmMessage });

    await createKeyPair(keyPairName);
    securityGroupId = await createSecurityGroup(securityGroupName);
    const { PublicIp, AllocationId } = await allocateIpAddress();
    ipAllocationId = AllocationId;
    publicIp = PublicIp;
    const ipAddress = await authorizeSecurityGroupIngress(securityGroupId);

    const { KeyName } = await describeKeyPair(keyPairName);
    const { GroupName } = await describeSecurityGroup(securityGroupName);
    console.log(`# created the key pair ${KeyName}.\n`);
    console.log(
      `# created the security group ${GroupName}`,
      `and allowed SSH access from ${ipAddress} (your IP).\n`,
    );
    console.log(`# allocated ${publicIp} to be used for your EC2 instance.\n`);

    await prompter.confirm({ message: confirmMessage });

    // Creating the instance
    console.log(wrapText("Create the instance."));
    console.log(
      "You get to choose which image you want. Select an amazon-linux-2 image
from the following:",
    );
    const imageDetails = await getAmznLinux2AMIs();
    const instanceTypeDetails = await getCompatibleInstanceTypes(imageDetails);
    console.log("Creating your instance. This can take a few seconds.");
    instanceId = await runInstance({
      keyPairName,
      securityGroupId,
      imageId: imageDetails.ImageId,
      instanceType: instanceTypeDetails.InstanceType,
    });
    const instanceDetails = await describeInstance(instanceId);
    console.log(`# instance ${instanceId}.\n`);
    console.log(instanceDetails);
    console.log(
```

```
      `\nYou should now be able to SSH into your instance from another
terminal:`,
      `\n${displaySSHConnectionInfo({
        publicIp: instanceDetails.PublicIpAddress,
        keyPairName,
      })}`,
    );

    await prompter.confirm({ message: confirmMessage });

    // Understanding the IP address.
    console.log(wrapText("Understanding the IP address."));
    console.log(
      "When you stop and start an instance, the IP address will change. I'll
restart your",
      "instance for you. Notice how the IP address changes.",
    );
    const ipAddressAfterRestart = await restartInstance(instanceId);
    console.log(
      `\n Instance started. The IP address changed from
${instanceDetails.PublicIpAddress} to ${ipAddressAfterRestart}`,
      `\n${displaySSHConnectionInfo({
        publicIp: ipAddressAfterRestart,
        keyPairName,
      })}`,
    );
    await prompter.confirm({ message: confirmMessage });
    console.log(
      `If you want to the IP address to be static, you can associate an
allocated`,
      `IP address to your instance. I allocated ${publicIp} for you earlier, and
now I'll associate it to your instance.`,
    );
    associationId = await associateAddress({
      allocationId: ipAllocationId,
      instanceId,
    });
    console.log(
      "Done. Now you should be able to SSH using the new IP.\n",
      `${displaySSHConnectionInfo({ publicIp, keyPairName })}`,
    );
    await prompter.confirm({ message: confirmMessage });
    console.log(
```

```
      "I'll restart the server again so you can see the IP address remains the
 same.",
    );
    const ipAddressAfterAssociated = await restartInstance(instanceId);
    console.log(
      `Done. Here's your SSH info. Notice the IP address hasn't changed.`,
      `\n${displaySSHConnectionInfo({
        publicIp: ipAddressAfterAssociated,
        keyPairName,
      })}`,
    );
    await prompter.confirm({ message: confirmMessage });
  } catch (err) {
    console.error(err);
  } finally {
    // Clean up.
    console.log(wrapText("Clean up."));
    console.log("Now I'll clean up all of the stuff I created.");
    await prompter.confirm({ message: confirmMessage });
    console.log("Cleaning up. Some of these steps can take a bit of time.");
    await disassociateAddress(associationId);
    await terminateInstance(instanceId);
    await releaseAddress(ipAllocationId);
    await deleteSecurityGroup(securityGroupId);
    deleteTemporaryDirectory();
    await deleteKeyPair(keyPairName);
    console.log(
      "Done cleaning up. Thanks for staying until the end!",
      "If you have any feedback please use the feedback button in the docs",
      "or create an issue on GitHub.",
    );
  }
};
```

- API 세부 정보는 AWS SDK for JavaScript API 참조의 다음 주제를 참조하십시오.
  - AllocateAddress
  - AssociateAddress
  - AuthorizeSecurityGroupIngress
  - CreateKeyPair
  - CreateSecurityGroup

- DeleteKeyPair
- DeleteSecurityGroup
- DescribeImages
- DescribeInstanceTypes
- DescribeInstances
- DescribeKeyPairs
- DescribeSecurityGroups
- DisassociateAddress
- ReleaseAddress
- RunInstances
- StartInstances
- StopInstances
- TerminateInstances
- UnmonitorInstances

Kotlin

SDK for Kotlin

> **Note**
>
> 더 많은 것이 있어요 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

```
/**
 Before running this Kotlin 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-kotlin/latest/developer-guide/setup.html

 This Kotlin example performs the following tasks:

 1. Creates an RSA key pair and saves the private key data as a .pem file.
```

```
 2. Lists key pairs.
 3. Creates a security group for the default VPC.
 4. Displays security group information.
 5. Gets a list of Amazon Linux 2 AMIs and selects one.
 6. Gets more information about the image.
 7. Gets a list of instance types that are compatible with the selected AMI's
 architecture.
 8. Creates an instance with the key pair, security group, AMI, and an instance
 type.
 9. Displays information about the instance.
 10. Stops the instance and waits for it to stop.
 11. Starts the instance and waits for it to start.
 12. Allocates an Elastic IP address and associates it with the instance.
 13. Displays SSH connection info for the instance.
 14. Disassociates and deletes the Elastic IP address.
 15. Terminates the instance.
 16. Deletes the security group.
 17. Deletes the key pair.
 */

val DASHES = String(CharArray(80)).replace("\u0000", "-")

suspend fun main(args: Array<String>) {
    val usage = """
        Usage:
            <keyName> <fileName> <groupName> <groupDesc> <vpcId> <myIpAddress>

        Where:
            keyName - A key pair name (for example, TestKeyPair).
            fileName - A file name where the key information is written to.
            groupName - The name of the security group.
            groupDesc - The description of the security group.
            vpcId - A VPC ID. You can get this value from the AWS Management
 Console.
            myIpAddress - The IP address of your development machine.

"""

    if (args.size != 6) {
        println(usage)
        exitProcess(0)
    }

    val keyName = args[0]
```

```
    val fileName = args[1]
    val groupName = args[2]
    val groupDesc = args[3]
    val vpcId = args[4]
    val myIpAddress = args[5]
    var newInstanceId: String? = ""

    println(DASHES)
    println("Welcome to the Amazon EC2 example scenario.")
    println(DASHES)

    println(DASHES)
    println("1. Create an RSA key pair and save the private key material as
a .pem file.")
    createKeyPairSc(keyName, fileName)
    println(DASHES)

    println(DASHES)
    println("2. List key pairs.")
    describeEC2KeysSc()
    println(DASHES)

    println(DASHES)
    println("3. Create a security group.")
    val groupId = createEC2SecurityGroupSc(groupName, groupDesc, vpcId,
myIpAddress)
    println(DASHES)

    println(DASHES)
    println("4. Display security group info for the newly created security
group.")
    describeSecurityGroupsSc(groupId.toString())
    println(DASHES)

    println(DASHES)
    println("5. Get a list of Amazon Linux 2 AMIs and select one with amzn2 in
the name.")
    val instanceId = getParaValuesSc()
    if (instanceId == "") {
        println("The instance Id value isn't valid.")
        exitProcess(0)
    }
    println("The instance Id is $instanceId.")
    println(DASHES)
```

```
    println(DASHES)
    println("6. Get more information about an amzn2 image and return the AMI
value.")
    val amiValue = instanceId?.let { describeImageSc(it) }
    if (instanceId == "") {
        println("The instance Id value is invalid.")
        exitProcess(0)
    }
    println("The AMI value is $amiValue.")
    println(DASHES)

    println(DASHES)
    println("7. Get a list of instance types.")
    val instanceType = getInstanceTypesSc()
    println(DASHES)

    println(DASHES)
    println("8. Create an instance.")
    if (amiValue != null) {
        newInstanceId = runInstanceSc(instanceType, keyName, groupName, amiValue)
        println("The instance Id is $newInstanceId")
    }
    println(DASHES)

    println(DASHES)
    println("9. Display information about the running instance. ")
    var ipAddress = describeEC2InstancesSc(newInstanceId)
    println("You can SSH to the instance using this command:")
    println("ssh -i " + fileName + "ec2-user@" + ipAddress)
    println(DASHES)

    println(DASHES)
    println("10.  Stop the instance.")
    if (newInstanceId != null) {
        stopInstanceSc(newInstanceId)
    }
    println(DASHES)

    println(DASHES)
    println("11.  Start the instance.")
    if (newInstanceId != null) {
        startInstanceSc(newInstanceId)
    }
```

```
    ipAddress = describeEC2InstancesSc(newInstanceId)
    println("You can SSH to the instance using this command:")
    println("ssh -i " + fileName + "ec2-user@" + ipAddress)
    println(DASHES)

    println(DASHES)
    println("12. Allocate an Elastic IP address and associate it with the
 instance.")
    val allocationId = allocateAddressSc()
    println("The allocation Id value is $allocationId")
    val associationId = associateAddressSc(newInstanceId, allocationId)
    println("The associate Id value is $associationId")
    println(DASHES)

    println(DASHES)
    println("13. Describe the instance again.")
    ipAddress = describeEC2InstancesSc(newInstanceId)
    println("You can SSH to the instance using this command:")
    println("ssh -i " + fileName + "ec2-user@" + ipAddress)
    println(DASHES)

    println(DASHES)
    println("14. Disassociate and release the Elastic IP address.")
    disassociateAddressSc(associationId)
    releaseEC2AddressSc(allocationId)
    println(DASHES)

    println(DASHES)
    println("15. Terminate the instance and use a waiter.")
    if (newInstanceId != null) {
        terminateEC2Sc(newInstanceId)
    }
    println(DASHES)

    println(DASHES)
    println("16. Delete the security group.")
    if (groupId != null) {
        deleteEC2SecGroupSc(groupId)
    }
    println(DASHES)

    println(DASHES)
    println("17. Delete the key pair.")
    deleteKeysSc(keyName)
```

```
    println(DASHES)

    println(DASHES)
    println("You successfully completed the Amazon EC2 scenario.")
    println(DASHES)
}

suspend fun deleteKeysSc(keyPair: String) {
    val request =
        DeleteKeyPairRequest {
            keyName = keyPair
        }
    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.deleteKeyPair(request)
        println("Successfully deleted key pair named $keyPair")
    }
}

suspend fun deleteEC2SecGroupSc(groupIdVal: String) {
    val request =
        DeleteSecurityGroupRequest {
            groupId = groupIdVal
        }
    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.deleteSecurityGroup(request)
        println("Successfully deleted security group with Id $groupIdVal")
    }
}

suspend fun terminateEC2Sc(instanceIdVal: String) {
    val ti =
        TerminateInstancesRequest {
            instanceIds = listOf(instanceIdVal)
        }
    println("Wait for the instance to terminate. This will take a few minutes.")
    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.terminateInstances(ti)
        ec2.waitUntilInstanceTerminated {
            // suspend call
            instanceIds = listOf(instanceIdVal)
        }
        println("$instanceIdVal is terminated!")
    }
}
```

```
suspend fun releaseEC2AddressSc(allocId: String?) {
    val request =
        ReleaseAddressRequest {
            allocationId = allocId
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.releaseAddress(request)
        println("Successfully released Elastic IP address $allocId")
    }
}

suspend fun disassociateAddressSc(associationIdVal: String?) {
    val addressRequest =
        DisassociateAddressRequest {
            associationId = associationIdVal
        }
    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.disassociateAddress(addressRequest)
        println("You successfully disassociated the address!")
    }
}

suspend fun associateAddressSc(
    instanceIdVal: String?,
    allocationIdVal: String?,
): String? {
    val associateRequest =
        AssociateAddressRequest {
            instanceId = instanceIdVal
            allocationId = allocationIdVal
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val associateResponse = ec2.associateAddress(associateRequest)
        return associateResponse.associationId
    }
}

suspend fun allocateAddressSc(): String? {
    val allocateRequest =
        AllocateAddressRequest {
            domain = DomainType.Vpc
```

```
        }
    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val allocateResponse = ec2.allocateAddress(allocateRequest)
        return allocateResponse.allocationId
    }
}

suspend fun startInstanceSc(instanceId: String) {
    val request =
        StartInstancesRequest {
            instanceIds = listOf(instanceId)
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.startInstances(request)
        println("Waiting until instance $instanceId starts. This will take a few
 minutes.")
        ec2.waitUntilInstanceRunning {
            // suspend call
            instanceIds = listOf(instanceId)
        }
        println("Successfully started instance $instanceId")
    }
}

suspend fun stopInstanceSc(instanceId: String) {
    val request =
        StopInstancesRequest {
            instanceIds = listOf(instanceId)
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        ec2.stopInstances(request)
        println("Waiting until instance $instanceId stops. This will take a few
 minutes.")
        ec2.waitUntilInstanceStopped {
            // suspend call
            instanceIds = listOf(instanceId)
        }
        println("Successfully stopped instance $instanceId")
    }
}

suspend fun describeEC2InstancesSc(newInstanceId: String?): String {
```

```
    var pubAddress = ""
    var isRunning = false
    val request =
        DescribeInstancesRequest {
            instanceIds = listOf(newInstanceId.toString())
        }

    while (!isRunning) {
        Ec2Client { region = "us-west-2" }.use { ec2 ->
            val response = ec2.describeInstances(request)
            val state =
                response.reservations
                    ?.get(0)
                    ?.instances
                    ?.get(0)
                    ?.state
                    ?.name
                    ?. value
            if (state != null) {
                if (state.compareTo("running") == 0) {
                    println("Image id is
 ${response.reservations!!.get(0).instances?.get(0)?.imageId}")
                    println("Instance type is
 ${response.reservations!!.get(0).instances?.get(0)?.instanceType}")
                    println("Instance state is
 ${response.reservations!!.get(0).instances?.get(0)?.state}")
                    pubAddress =
                        response.reservations!!
                            .get(0)
                            .instances
                            ?.get(0)
                            ?.publicIpAddress
                            .toString()
                    println("Instance address is $pubAddress")
                    isRunning = true
                }
            }
        }
    }
    return pubAddress
}

suspend fun runInstanceSc(
    instanceTypeVal: String,
```

```
    keyNameVal: String,
    groupNameVal: String,
    amiIdVal: String,
): String {
    val runRequest =
        RunInstancesRequest {
            instanceType = InstanceType.fromValue(instanceTypeVal)
            keyName = keyNameVal
            securityGroups = listOf(groupNameVal)
            maxCount = 1
            minCount = 1
            imageId = amiIdVal
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.runInstances(runRequest)
        val instanceId = response.instances?.get(0)?.instanceId
        println("Successfully started EC2 Instance $instanceId based on AMI
 $amiIdVal")
        return instanceId.toString()
    }
}

// Get a list of instance types.
suspend fun getInstanceTypesSc(): String {
    var instanceType = ""
    val filterObs = ArrayList<Filter>()
    val filter =
        Filter {
            name = "processor-info.supported-architecture"
            values = listOf("arm64")
        }

    filterObs.add(filter)
    val typesRequest =
        DescribeInstanceTypesRequest {
            filters = filterObs
            maxResults = 10
        }
    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.describeInstanceTypes(typesRequest)
        response.instanceTypes?.forEach { type ->
            println("The memory information of this type is
 ${type.memoryInfo?.sizeInMib}")
```

```
            println("Maximum number of network cards is
 ${type.networkInfo?.maximumNetworkCards}")
            instanceType = type.instanceType.toString()
        }
        return instanceType
    }
}

// Display the Description field that corresponds to the instance Id value.
suspend fun describeImageSc(instanceId: String): String? {
    val imagesRequest =
        DescribeImagesRequest {
            imageIds = listOf(instanceId)
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.describeImages(imagesRequest)
        println("The description of the first image is
 ${response.images?.get(0)?.description}")
        println("The name of the first image is
 ${response.images?.get(0)?.name}")

        // Return the image Id value.
        return response.images?.get(0)?.imageId
    }
}

// Get the Id value of an instance with amzn2 in the name.
suspend fun getParaValuesSc(): String? {
    val parameterRequest =
        GetParametersByPathRequest {
            path = "/aws/service/ami-amazon-linux-latest"
        }

    SsmClient { region = "us-west-2" }.use { ssmClient ->
        val response = ssmClient.getParametersByPath(parameterRequest)
        response.parameters?.forEach { para ->
            println("The name of the para is: ${para.name}")
            println("The type of the para is: ${para.type}")
            println("")
            if (para.name?.let { filterName(it) } == true) {
                return para.value
            }
        }
```

```
    }
    return ""
}

fun filterName(name: String): Boolean {
    val parts = name.split("/").toTypedArray()
    val myValue = parts[4]
    return myValue.contains("amzn2")
}

suspend fun describeSecurityGroupsSc(groupId: String) {
    val request =
        DescribeSecurityGroupsRequest {
            groupIds = listOf(groupId)
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.describeSecurityGroups(request)
        for (group in response.securityGroups!!) {
            println("Found Security Group with id " + group.groupId.toString() +
 " and group VPC " + group.vpcId)
        }
    }
}

suspend fun createEC2SecurityGroupSc(
    groupNameVal: String?,
    groupDescVal: String?,
    vpcIdVal: String?,
    myIpAddress: String?,
): String? {
    val request =
        CreateSecurityGroupRequest {
            groupName = groupNameVal
            description = groupDescVal
            vpcId = vpcIdVal
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val resp = ec2.createSecurityGroup(request)
        val ipRange =
            IpRange {
                cidrIp = "$myIpAddress/0"
            }
```

```
        val ipPerm =
            IpPermission {
                ipProtocol = "tcp"
                toPort = 80
                fromPort = 80
                ipRanges = listOf(ipRange)
            }

        val ipPerm2 =
            IpPermission {
                ipProtocol = "tcp"
                toPort = 22
                fromPort = 22
                ipRanges = listOf(ipRange)
            }

        val authRequest =
            AuthorizeSecurityGroupIngressRequest {
                groupName = groupNameVal
                ipPermissions = listOf(ipPerm, ipPerm2)
            }
        ec2.authorizeSecurityGroupIngress(authRequest)
        println("Successfully added ingress policy to Security Group
 $groupNameVal")
        return resp.groupId
    }
}

suspend fun describeEC2KeysSc() {
    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.describeKeyPairs(DescribeKeyPairsRequest {})
        response.keyPairs?.forEach { keyPair ->
            println("Found key pair with name ${keyPair.keyName} and fingerprint
 ${ keyPair.keyFingerprint}")
        }
    }
}

suspend fun createKeyPairSc(
    keyNameVal: String,
    fileNameVal: String,
) {
    val request =
```

```
        CreateKeyPairRequest {
            keyName = keyNameVal
        }

    Ec2Client { region = "us-west-2" }.use { ec2 ->
        val response = ec2.createKeyPair(request)
        val content = response.keyMaterial
        if (content != null) {
            File(fileNameVal).writeText(content)
        }
        println("Successfully created key pair named $keyNameVal")
    }
}
```

- API 세부 정보는 AWS SDK for Kotlin API reference의 다음 주제를 참조하세요.
  - AllocateAddress
  - AssociateAddress
  - AuthorizeSecurityGroupIngress
  - CreateKeyPair
  - CreateSecurityGroup
  - DeleteKeyPair
  - DeleteSecurityGroup
  - DescribeImages
  - DescribeInstanceTypes
  - DescribeInstances
  - DescribeKeyPairs
  - DescribeSecurityGroups
  - DisassociateAddress
  - ReleaseAddress
  - RunInstances
  - StartInstances
  - StopInstances
  - TerminateInstances
  - UnmonitorInstances

Python

SDK for Python(Boto3)

> **Note**
>
> 더 많은 것이 있어요 GitHub. AWS 코드 예제 리포지토리에서 전체 예제를 찾고 설정 및 실행하는 방법을 배워보세요.

명령 프롬프트에서 대화형 시나리오를 실행합니다.

```
class Ec2InstanceScenario:
    """Runs an interactive scenario that shows how to get started using EC2
 instances."""

    def __init__(self, inst_wrapper, key_wrapper, sg_wrapper, eip_wrapper,
 ssm_client):
        """
        :param inst_wrapper: An object that wraps instance actions.
        :param key_wrapper: An object that wraps key pair actions.
        :param sg_wrapper: An object that wraps security group actions.
        :param eip_wrapper: An object that wraps Elastic IP actions.
        :param ssm_client: A Boto3 AWS Systems Manager client.
        """
        self.inst_wrapper = inst_wrapper
        self.key_wrapper = key_wrapper
        self.sg_wrapper = sg_wrapper
        self.eip_wrapper = eip_wrapper
        self.ssm_client = ssm_client

    @demo_func
    def create_and_list_key_pairs(self):
        """
        1. Creates an RSA key pair and saves its private key data as a .pem file
 in secure
           temporary storage. The private key data is deleted after the example
 completes.
        2. Lists the first five key pairs for the current account.
        """
        print(
            "Let's create an RSA key pair that you can be use to securely connect
 to "
```

```
            "your EC2 instance."
        )
        key_name = q.ask("Enter a unique name for your key: ", q.non_empty)
        self.key_wrapper.create(key_name)
        print(
            f"Created a key pair {self.key_wrapper.key_pair.key_name} and saved
the "
            f"private key to {self.key_wrapper.key_file_path}.\n"
        )
        if q.ask("Do you want to list some of your key pairs? (y/n) ",
q.is_yesno):
            self.key_wrapper.list(5)

    @demo_func
    def create_security_group(self):
        """
        1. Creates a security group for the default VPC.
        2. Adds an inbound rule to allow SSH. The SSH rule allows only
           inbound traffic from the current computer's public IPv4 address.
        3. Displays information about the security group.

        This function uses 'http://checkip.amazonaws.com' to get the current
public IP
        address of the computer that is running the example. This method works in
most
        cases. However, depending on how your computer connects to the internet,
you
        might have to manually add your public IP address to the security group
by using
        the AWS Management Console.
        """
        print("Let's create a security group to manage access to your instance.")
        sg_name = q.ask("Enter a unique name for your security group: ",
q.non_empty)
        security_group = self.sg_wrapper.create(
            sg_name, "Security group for example: get started with instances."
        )
        print(
            f"Created security group {security_group.group_name} in your default
"
            f"VPC {security_group.vpc_id}.\n"
        )

        ip_response = urllib.request.urlopen("http://checkip.amazonaws.com")
```

```
        current_ip_address = ip_response.read().decode("utf-8").strip()
        print("Let's add a rule to allow SSH only from your current IP address.")
        print(f"Your public IP address is {current_ip_address}.")
        q.ask("Press Enter to add this rule to your security group.")
        response = self.sg_wrapper.authorize_ingress(current_ip_address)
        if response["Return"]:
            print("Security group rules updated.")
        else:
            print("Couldn't update security group rules.")
        self.sg_wrapper.describe()

    @demo_func
    def create_instance(self):
        """
        1. Gets a list of Amazon Linux 2 AMIs from AWS Systems Manager.
 Specifying the
           '/aws/service/ami-amazon-linux-latest' path returns only the latest
 AMIs.
        2. Gets and displays information about the available AMIs and lets you
 select one.
        3. Gets a list of instance types that are compatible with the selected
 AMI and
           lets you select one.
        4. Creates an instance with the previously created key pair and security
 group,
           and the selected AMI and instance type.
        5. Waits for the instance to be running and then displays its
 information.
        """
        ami_paginator = self.ssm_client.get_paginator("get_parameters_by_path")
        ami_options = []
        for page in ami_paginator.paginate(Path="/aws/service/ami-amazon-linux-
latest"):
            ami_options += page["Parameters"]
        amzn2_images = self.inst_wrapper.get_images(
            [opt["Value"] for opt in ami_options if "amzn2" in opt["Name"]]
        )
        print(
            "Let's create an instance from an Amazon Linux 2 AMI. Here are some
 options:"
        )
        image_choice = q.choose(
            "Which one do you want to use? ", [opt.description for opt in
 amzn2_images]
```

```
        )
        print("Great choice!\n")

        print(
            f"Here are some instance types that support the "
            f"{amzn2_images[image_choice].architecture} architecture of the
image:"
        )
        inst_types = self.inst_wrapper.get_instance_types(
            amzn2_images[image_choice].architecture
        )
        inst_type_choice = q.choose(
            "Which one do you want to use? ", [it["InstanceType"] for it in
inst_types]
        )
        print("Another great choice.\n")

        print("Creating your instance and waiting for it to start...")
        self.inst_wrapper.create(
            amzn2_images[image_choice],
            inst_types[inst_type_choice]["InstanceType"],
            self.key_wrapper.key_pair,
            [self.sg_wrapper.security_group],
        )
        print(f"Your instance is ready:\n")
        self.inst_wrapper.display()

        print("You can use SSH to connect to your instance.")
        print(
            "If the connection attempt times out, you might have to manually
update "
            "the SSH ingress rule for your IP address in the AWS Management
Console."
        )
        self._display_ssh_info()

    def _display_ssh_info(self):
        """
        Displays an SSH connection string that can be used to connect to a
running
        instance.
        """
        print("To connect, open another command prompt and run the following
command:")
```

```
        if self.eip_wrapper.elastic_ip is None:
            print(
                f"\tssh -i {self.key_wrapper.key_file_path} "
                f"ec2-user@{self.inst_wrapper.instance.public_ip_address}"
            )
        else:
            print(
                f"\tssh -i {self.key_wrapper.key_file_path} "
                f"ec2-user@{self.eip_wrapper.elastic_ip.public_ip}"
            )
        q.ask("Press Enter when you're ready to continue the demo.")

    @demo_func
    def associate_elastic_ip(self):
        """
        1. Allocates an Elastic IP address and associates it with the instance.
        2. Displays an SSH connection string that uses the Elastic IP address.
        """
        print(
            "You can allocate an Elastic IP address and associate it with your
instance\n"
            "to keep a consistent IP address even when your instance restarts."
        )
        elastic_ip = self.eip_wrapper.allocate()
        print(f"Allocated static Elastic IP address: {elastic_ip.public_ip}.")
        self.eip_wrapper.associate(self.inst_wrapper.instance)
        print(f"Associated your Elastic IP with your instance.")
        print(
            "You can now use SSH to connect to your instance by using the Elastic
IP."
        )
        self._display_ssh_info()

    @demo_func
    def stop_and_start_instance(self):
        """
        1. Stops the instance and waits for it to stop.
        2. Starts the instance and waits for it to start.
        3. Displays information about the instance.
        4. Displays an SSH connection string. When an Elastic IP address is
associated
           with the instance, the IP address stays consistent when the instance
stops
           and starts.
```

```
        """
        print("Let's stop and start your instance to see what changes.")
        print("Stopping your instance and waiting until it's stopped...")
        self.inst_wrapper.stop()
        print("Your instance is stopped. Restarting...")
        self.inst_wrapper.start()
        print("Your instance is running.")
        self.inst_wrapper.display()
        if self.eip_wrapper.elastic_ip is None:
            print(
                "Every time your instance is restarted, its public IP address
changes."
            )
        else:
            print(
                "Because you have associated an Elastic IP with your instance,
you can \n"
                "connect by using a consistent IP address after the instance
restarts."
            )
        self._display_ssh_info()

    @demo_func
    def cleanup(self):
        """
        1. Disassociate and delete the previously created Elastic IP.
        2. Terminate the previously created instance.
        3. Delete the previously created security group.
        4. Delete the previously created key pair.
        """
        print("Let's clean everything up. This example created these resources:")
        print(f"\tElastic IP: {self.eip_wrapper.elastic_ip.allocation_id}")
        print(f"\tInstance: {self.inst_wrapper.instance.id}")
        print(f"\tSecurity group: {self.sg_wrapper.security_group.id}")
        print(f"\tKey pair: {self.key_wrapper.key_pair.name}")
        if q.ask("Ready to delete these resources? (y/n) ", q.is_yesno):
            self.eip_wrapper.disassociate()
            print("Disassociated the Elastic IP from the instance.")
            self.eip_wrapper.release()
            print("Released the Elastic IP.")
            print("Terminating the instance and waiting for it to terminate...")
            self.inst_wrapper.terminate()
            print("Instance terminated.")
            self.sg_wrapper.delete()
```

```
            print("Deleted security group.")
            self.key_wrapper.delete()
            print("Deleted key pair.")

    def run_scenario(self):
        logging.basicConfig(level=logging.INFO, format="%(levelname)s:
 %(message)s")

        print("-" * 88)
        print(
            "Welcome to the Amazon Elastic Compute Cloud (Amazon EC2) get started
 with instances demo."
        )
        print("-" * 88)

        self.create_and_list_key_pairs()
        self.create_security_group()
        self.create_instance()
        self.stop_and_start_instance()
        self.associate_elastic_ip()
        self.stop_and_start_instance()
        self.cleanup()

        print("\nThanks for watching!")
        print("-" * 88)


if __name__ == "__main__":
    try:
        scenario = Ec2InstanceScenario(
            InstanceWrapper.from_resource(),
            KeyPairWrapper.from_resource(),
            SecurityGroupWrapper.from_resource(),
            ElasticIpWrapper.from_resource(),
            boto3.client("ssm"),
        )
        scenario.run_scenario()
    except Exception:
        logging.exception("Something went wrong with the demo.")
```

키 페어 작업을 래핑하는 클래스를 정의합니다.

```
class KeyPairWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) key pair
 actions."""

    def __init__(self, ec2_resource, key_file_dir, key_pair=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param key_file_dir: The folder where the private key information is
 stored.
                             This should be a secure folder.
        :param key_pair: A Boto3 KeyPair object. This is a high-level object that
                         wraps key pair actions.
        """
        self.ec2_resource = ec2_resource
        self.key_pair = key_pair
        self.key_file_path = None
        self.key_file_dir = key_file_dir

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource, tempfile.TemporaryDirectory())


    def create(self, key_name):
        """
        Creates a key pair that can be used to securely connect to an EC2
 instance.
        The returned key pair contains private key information that cannot be
 retrieved
        again. The private key data is stored as a .pem file.

        :param key_name: The name of the key pair to create.
        :return: A Boto3 KeyPair object that represents the newly created key
 pair.
        """
        try:
            self.key_pair = self.ec2_resource.create_key_pair(KeyName=key_name)
            self.key_file_path = os.path.join(
                self.key_file_dir.name, f"{self.key_pair.name}.pem"
```

```
            )
            with open(self.key_file_path, "w") as key_file:
                key_file.write(self.key_pair.key_material)
        except ClientError as err:
            logger.error(
                "Couldn't create key %s. Here's why: %s: %s",
                key_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return self.key_pair


    def list(self, limit):
        """
        Displays a list of key pairs for the current account.

        :param limit: The maximum number of key pairs to list.
        """
        try:
            for kp in self.ec2_resource.key_pairs.limit(limit):
                print(f"Found {kp.key_type} key {kp.name} with fingerprint:")
                print(f"\t{kp.key_fingerprint}")
        except ClientError as err:
            logger.error(
                "Couldn't list key pairs. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def delete(self):
        """
        Deletes a key pair.
        """
        if self.key_pair is None:
            logger.info("No key pair to delete.")
            return

        key_name = self.key_pair.name
        try:
```

```
            self.key_pair.delete()
            self.key_pair = None
        except ClientError as err:
            logger.error(
                "Couldn't delete key %s. Here's why: %s : %s",
                key_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

보안 그룹 작업을 래핑하는 클래스를 정의합니다.

```
class SecurityGroupWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) security group
 actions."""

    def __init__(self, ec2_resource, security_group=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param security_group: A Boto3 SecurityGroup object. This is a high-level
 object
                               that wraps security group actions.
        """
        self.ec2_resource = ec2_resource
        self.security_group = security_group

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def create(self, group_name, group_description):
        """
```

```
        Creates a security group in the default virtual private cloud (VPC) of
the
        current account.

        :param group_name: The name of the security group to create.
        :param group_description: The description of the security group to
create.
        :return: A Boto3 SecurityGroup object that represents the newly created
security group.
        """
        try:
            self.security_group = self.ec2_resource.create_security_group(
                GroupName=group_name, Description=group_description
            )
        except ClientError as err:
            logger.error(
                "Couldn't create security group %s. Here's why: %s: %s",
                group_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return self.security_group


    def authorize_ingress(self, ssh_ingress_ip):
        """
        Adds a rule to the security group to allow access to SSH.

        :param ssh_ingress_ip: The IP address that is granted inbound access to
connect
                               to port 22 over TCP, used for SSH.
        :return: The response to the authorization request. The 'Return' field of
the
                 response indicates whether the request succeeded or failed.
        """
        if self.security_group is None:
            logger.info("No security group to update.")
            return

        try:
            ip_permissions = [
                {
```

```
                    # SSH ingress open to only the specified IP address.
                    "IpProtocol": "tcp",
                    "FromPort": 22,
                    "ToPort": 22,
                    "IpRanges": [{"CidrIp": f"{ssh_ingress_ip}/32"}],
                }
            ]
            response = self.security_group.authorize_ingress(
                IpPermissions=ip_permissions
            )
        except ClientError as err:
            logger.error(
                "Couldn't authorize inbound rules for %s. Here's why: %s: %s",
                self.security_group.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response


    def describe(self):
        """
        Displays information about the security group.
        """
        if self.security_group is None:
            logger.info("No security group to describe.")
            return

        try:
            print(f"Security group: {self.security_group.group_name}")
            print(f"\tID: {self.security_group.id}")
            print(f"\tVPC: {self.security_group.vpc_id}")
            if self.security_group.ip_permissions:
                print(f"Inbound permissions:")
                pp(self.security_group.ip_permissions)
        except ClientError as err:
            logger.error(
                "Couldn't get data for security group %s. Here's why: %s: %s",
                self.security_group.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
```

```
            raise


    def delete(self):
        """
        Deletes the security group.
        """
        if self.security_group is None:
            logger.info("No security group to delete.")
            return

        group_id = self.security_group.id
        try:
            self.security_group.delete()
        except ClientError as err:
            logger.error(
                "Couldn't delete security group %s. Here's why: %s: %s",
                group_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

인스턴스 작업을 래핑하는 클래스를 정의합니다.

```
class InstanceWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) instance
 actions."""

    def __init__(self, ec2_resource, instance=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param instance: A Boto3 Instance object. This is a high-level object
 that
                         wraps instance actions.
        """
```

```
        self.ec2_resource = ec2_resource
        self.instance = instance

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def create(self, image, instance_type, key_pair, security_groups=None):
        """
        Creates a new EC2 instance. The instance starts immediately after
        it is created.

        The instance is created in the default VPC of the current account.

        :param image: A Boto3 Image object that represents an Amazon Machine
Image (AMI)
                      that defines attributes of the instance that is created.
The AMI
                      defines things like the kind of operating system and the
type of
                      storage used by the instance.
        :param instance_type: The type of instance to create, such as 't2.micro'.
                              The instance type defines things like the number of
CPUs and
                              the amount of memory.
        :param key_pair: A Boto3 KeyPair or KeyPairInfo object that represents
the key
                         pair that is used to secure connections to the instance.
        :param security_groups: A list of Boto3 SecurityGroup objects that
represents the
                                security groups that are used to grant access to
the
                                instance. When no security groups are specified,
the
                                default security group of the VPC is used.
        :return: A Boto3 Instance object that represents the newly created
instance.
        """
        try:
            instance_params = {
                "ImageId": image.id,
                "InstanceType": instance_type,
```

```
                "KeyName": key_pair.name,
            }
            if security_groups is not None:
                instance_params["SecurityGroupIds"] = [sg.id for sg in
security_groups]
            self.instance = self.ec2_resource.create_instances(
                **instance_params, MinCount=1, MaxCount=1
            )[0]
            self.instance.wait_until_running()
        except ClientError as err:
            logging.error(
                "Couldn't create instance with image %s, instance type %s, and
key %s. "
                "Here's why: %s: %s",
                image.id,
                instance_type,
                key_pair.name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return self.instance


    def display(self, indent=1):
        """
        Displays information about an instance.

        :param indent: The visual indent to apply to the output.
        """
        if self.instance is None:
            logger.info("No instance to display.")
            return

        try:
            self.instance.load()
            ind = "\t" * indent
            print(f"{ind}ID: {self.instance.id}")
            print(f"{ind}Image ID: {self.instance.image_id}")
            print(f"{ind}Instance type: {self.instance.instance_type}")
            print(f"{ind}Key name: {self.instance.key_name}")
            print(f"{ind}VPC ID: {self.instance.vpc_id}")
            print(f"{ind}Public IP: {self.instance.public_ip_address}")
```

```
            print(f"{ind}State: {self.instance.state['Name']}")
        except ClientError as err:
            logger.error(
                "Couldn't display your instance. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def terminate(self):
        """
        Terminates an instance and waits for it to be in a terminated state.
        """
        if self.instance is None:
            logger.info("No instance to terminate.")
            return

        instance_id = self.instance.id
        try:
            self.instance.terminate()
            self.instance.wait_until_terminated()
            self.instance = None
        except ClientError as err:
            logging.error(
                "Couldn't terminate instance %s. Here's why: %s: %s",
                instance_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def start(self):
        """
        Starts an instance and waits for it to be in a running state.

        :return: The response to the start request.
        """
        if self.instance is None:
            logger.info("No instance to start.")
            return

        try:
```

```
            response = self.instance.start()
            self.instance.wait_until_running()
        except ClientError as err:
            logger.error(
                "Couldn't start instance %s. Here's why: %s: %s",
                self.instance.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response


    def stop(self):
        """
        Stops an instance and waits for it to be in a stopped state.

        :return: The response to the stop request.
        """
        if self.instance is None:
            logger.info("No instance to stop.")
            return

        try:
            response = self.instance.stop()
            self.instance.wait_until_stopped()
        except ClientError as err:
            logger.error(
                "Couldn't stop instance %s. Here's why: %s: %s",
                self.instance.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response


    def get_images(self, image_ids):
        """
        Gets information about Amazon Machine Images (AMIs) from a list of AMI
IDs.
```

```
        :param image_ids: The list of AMIs to look up.
        :return: A list of Boto3 Image objects that represent the requested AMIs.
        """
        try:
            images = list(self.ec2_resource.images.filter(ImageIds=image_ids))
        except ClientError as err:
            logger.error(
                "Couldn't get images. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return images


    def get_instance_types(self, architecture):
        """
        Gets instance types that support the specified architecture and are
designated
        as either 'micro' or 'small'. When an instance is created, the instance
type
        you specify must support the architecture of the AMI you use.

        :param architecture: The kind of architecture the instance types must
support,
                             such as 'x86_64'.
        :return: A list of instance types that support the specified architecture
                 and are either 'micro' or 'small'.
        """
        try:
            inst_types = []
            it_paginator = self.ec2_resource.meta.client.get_paginator(
                "describe_instance_types"
            )
            for page in it_paginator.paginate(
                Filters=[
                    {
                        "Name": "processor-info.supported-architecture",
                        "Values": [architecture],
                    },
                    {"Name": "instance-type", "Values": ["*.micro", "*.small"]},
                ]
            ):
```

```
                inst_types += page["InstanceTypes"]
        except ClientError as err:
            logger.error(
                "Couldn't get instance types. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return inst_types
```

탄력적 IP 작업을 래핑하는 클래스를 정의합니다.

```
class ElasticIpWrapper:
    """Encapsulates Amazon Elastic Compute Cloud (Amazon EC2) Elastic IP address
 actions."""

    def __init__(self, ec2_resource, elastic_ip=None):
        """
        :param ec2_resource: A Boto3 Amazon EC2 resource. This high-level
 resource
                             is used to create additional high-level objects
                             that wrap low-level Amazon EC2 service actions.
        :param elastic_ip: A Boto3 VpcAddress object. This is a high-level object
 that
                           wraps Elastic IP actions.
        """
        self.ec2_resource = ec2_resource
        self.elastic_ip = elastic_ip

    @classmethod
    def from_resource(cls):
        ec2_resource = boto3.resource("ec2")
        return cls(ec2_resource)


    def allocate(self):
        """
        Allocates an Elastic IP address that can be associated with an Amazon EC2
```

```
        instance. By using an Elastic IP address, you can keep the public IP
address
        constant even when you restart the associated instance.

        :return: The newly created Elastic IP object. By default, the address is
not
                 associated with any instance.
        """
        try:
            response =
self.ec2_resource.meta.client.allocate_address(Domain="vpc")
            self.elastic_ip =
self.ec2_resource.VpcAddress(response["AllocationId"])
        except ClientError as err:
            logger.error(
                "Couldn't allocate Elastic IP. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return self.elastic_ip


    def associate(self, instance):
        """
        Associates an Elastic IP address with an instance. When this association
is
        created, the Elastic IP's public IP address is immediately used as the
public
        IP address of the associated instance.

        :param instance: A Boto3 Instance object. This is a high-level object
that wraps
                         Amazon EC2 instance actions.
        :return: A response that contains the ID of the association.
        """
        if self.elastic_ip is None:
            logger.info("No Elastic IP to associate.")
            return

        try:
            response = self.elastic_ip.associate(InstanceId=instance.id)
        except ClientError as err:
```

```
            logger.error(
                "Couldn't associate Elastic IP %s with instance %s. Here's why:
%s: %s",
                self.elastic_ip.allocation_id,
                instance.id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        return response


    def disassociate(self):
        """
        Removes an association between an Elastic IP address and an instance.
When the
        association is removed, the instance is assigned a new public IP address.
        """
        if self.elastic_ip is None:
            logger.info("No Elastic IP to disassociate.")
            return

        try:
            self.elastic_ip.association.delete()
        except ClientError as err:
            logger.error(
                "Couldn't disassociate Elastic IP %s from its instance. Here's
why: %s: %s",
                self.elastic_ip.allocation_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def release(self):
        """
        Releases an Elastic IP address. After the Elastic IP address is released,
        it can no longer be used.
        """
        if self.elastic_ip is None:
            logger.info("No Elastic IP to release.")
            return
```

```
        try:
            self.elastic_ip.release()
        except ClientError as err:
            logger.error(
                "Couldn't release Elastic IP address %s. Here's why: %s: %s",
                self.elastic_ip.allocation_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- API 세부 정보는 AWS SDK for Python (Boto3) API 참조의 다음 주제를 참조하십시오.
  - AllocateAddress
  - AssociateAddress
  - AuthorizeSecurityGroupIngress
  - CreateKeyPair
  - CreateSecurityGroup
  - DeleteKeyPair
  - DeleteSecurityGroup
  - DescribeImages
  - DescribeInstanceTypes
  - DescribeInstances
  - DescribeKeyPairs
  - DescribeSecurityGroups
  - DisassociateAddress
  - ReleaseAddress
  - RunInstances
  - StartInstances
  - StopInstances
  - TerminateInstances
  - UnmonitorInstances

AWS SDK 개발자 가이드 및 코드 예제의 전체 목록은 을 참조하십시오SDK를 사용하여 Amazon EC2 리소스 생성 AWS. 이 주제에는 시작하기에 대한 정보와 이전 SDK 버전에 대한 세부 정보도 포함되어 있습니다.

Amazon을 사용하여 Amazon EC2 API 요청을 모니터링합니다. CloudWatch

원시 데이터를 수집하여 읽기 쉬운 거의 실시간 지표로 처리하는 Amazon을 사용하여 Amazon CloudWatch EC2 API 요청을 모니터링할 수 있습니다. 이러한 지표는 시간 경과에 따른 Amazon EC2 API 작업의 사용 및 결과를 추적할 수 있는 간단한 방법을 제공합니다. 이 정보를 통해 웹 애플리케이션의 성능을 더 잘 파악하고 다양한 문제를 식별하고 진단할 수 있습니다. 또한 특정 임계값을 감시하는 경보를 설정하고 해당 임계값에 도달하면 알림을 보내거나 특정 조치를 취할 수 있습니다.

에 대한 CloudWatch 자세한 내용은 Amazon CloudWatch 사용 설명서를 참조하십시오.

> **Important**
>
> Amazon EC2 API 지표는 옵트인 기능입니다. 이 기능에 대한 액세스를 요청해야 합니다. 자세한 내용은 the section called “Amazon EC2 API 메트릭을 활성화합니다.” 단원을 참조하십시오.

목차

Amazon EC2 API 메트릭을 활성화합니다.

다음 절차를 사용하여 이 기능에 대한 액세스를 요청하십시오. AWS 계정

이 기능에 대한 액세스를 요청하려면

1. AWS Support 센터를 여세요.
2. 사례 생성을 선택합니다.
3. 계정 및 결제 지원을 선택합니다.

4. 서비스에서 일반 정보 및 시작하기를 선택합니다.
5. 카테고리에서 사용 AWS 및 서비스를 선택합니다.
6. 다음 단계: 추가 정보를 선택합니다
7. 제목에 **Request access to Amazon EC2 API metrics**을 입력합니다.
8. 설명에 **Please grant my account access to Amazon EC2 API metrics. Related page: https://docs.aws.amazon.com/AWSEC2/latest/APIReference/monitor.html**를 입력합니다. 액세스가 필요한 지역도 포함하세요.
9. 다음 단계: 지금 해결하거나 문의하기를 선택합니다.
10. 연락처 탭에서 원하는 연락처 언어와 연락 방법을 선택합니다.
11. 제출을 선택합니다.

Amazon EC2 API 메트릭 및 디멘션

지표

Amazon EC2 API 지표는 네임스페이스에 포함되어 있습니다. AWS/EC2/API 다음 표에는 Amazon EC2 API 요청에 사용할 수 있는 지표가 나와 있습니다.

지표	설명
ClientErrors	클라이언트 오류로 인해 실패한 API 요청 수. 이러한 오류는 대개 요청에 잘못되거나 잘못된 파라미터를 지정하거나 작업 또는 리소스를 사용할 권한이 없는 사용자를 대신하여 작업 또는 리소스를 사용하는 등 클라이언트가 수행한 작업으로 인해 발생합니다. 단위: 수
RequestLimitExceeded	Amazon EC2 API가 사용자 계정에 허용한 최대 요청 비율을 초과한 횟수 Amazon EC2 API 요청은 서비스 성능을 유지할 수 있도록 조절됩니다. 요청이 병목 현상이 발생한 경우 오류가 발생합니다. Client.RequestLimitExceeded

지표	설명
	단위: 수
ServerErrors	내부 서버 오류로 인해 실패한 API 요청 수 이러한 오류는 일반적으로 AWS 서버측 오류, 예외 또는 실패로 인해 발생합니다. 단위: 수
SuccessfulCalls	성공한 API 요청 수. 단위: 수

차원

Amazon EC2 메트릭 데이터는 모든 EC2 API 작업에서 필터링할 수 있습니다. 치수에 대한 자세한 내용은 Amazon CloudWatch 개념을 참조하십시오.

지표 데이터 보존

Amazon EC2 API 지표는 1분 CloudWatch 간격으로 전송됩니다. CloudWatch 다음과 같이 지표 데이터를 보관합니다.

- 기간이 60초(1분)로 설정된 데이터 요소들은 15일 동안 사용할 수 있습니다.
- 기간이 300초 (5분) 인 데이터 포인트를 63일 동안 사용할 수 있습니다.
- 기간이 3600초 (1시간) 인 데이터 포인트는 455일 (15개월) 동안 사용할 수 있습니다.

사용자를 대신하여 이루어진 모니터링 요청

AWS 서비스 연결 역할이 수행한 요청과 같이 사용자를 대신하여 서비스에서 수행한 API 요청은 API 제한 한도에 포함되지 않으며 계정에 대한 지표를 CloudWatch Amazon으로 전송하지 않습니다. 이러한 요청은 을 사용하여 모니터링할 수 없습니다. CloudWatch

타사 서비스 공급자가 사용자를 대신하여 수행한 API 요청은 API 제한 한도에 포함되며, 타사 서비스 제공업체는 사용자 계정의 지표를 CloudWatch Amazon으로 전송합니다. 를 사용하여 이러한 요청을 모니터링할 수 있습니다. CloudWatch

결제

표준 CloudWatch 요금 및 요금이 적용됩니다. Amazon EC2 API 메트릭 사용에 대해서는 추가 요금이 부과되지 않습니다. 자세한 내용은 Amazon CloudWatch 요금을 참조하십시오.

아마존과 협력하기 CloudWatch

목차

메트릭 보기 CloudWatch

Amazon EC2 API 지표를 보려면 다음 절차를 사용하십시오.

사전 조건

계정의 Amazon EC2 API 지표에 대한 액세스를 활성화해야 합니다. 자세한 정보는 the section called “Amazon EC2 API 메트릭을 활성화합니다.”을 참조하세요.

콘솔을 사용하여 Amazon EC2 API 지표를 보려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 지표, 모든 지표를 선택합니다.
3. 찾아보기 탭에서 EC2/API 메트릭 네임스페이스를 선택합니다.
4. 지표를 보려면 지표 차원을 선택합니다.

명령줄을 사용하여 Amazon EC2 API 지표를 보려면

다음 명령 중 하나를 사용합니다.

- 목록 지표 ()AWS CLI

```
aws cloudwatch list-metrics --namespace "AWS/EC2/API"
```

- MetricListGet-CW ()AWS Tools for Windows PowerShell

```
Get-CWMetricList -Namespace "AWS/EC2/API"
```

알람 CloudWatch 만들기

CloudWatch 경보 상태가 변경될 때 Amazon SNS 메시지를 보내는 경보를 생성할 수 있습니다. 경보는 지정한 기간 동안 단일 지표를 감시합니다. 경보는 기간 수에 대한 주어진 임계값과 지표 값을 비교하여 SNS 주제에 알림을 보냅니다.

예를 들어 서버 측 오류로 인해 실패한 DescribeInstances API 요청 수를 모니터링하는 경보를 생성할 수 있습니다. 다음 경보는 5분 동안 DescribeInstances API 요청 실패 수가 서버측 오류 임계값인 10개에 도달하면 이메일 알림을 보냅니다.

사전 조건

계정의 Amazon EC2 API 지표에 대한 액세스를 활성화해야 합니다. 자세한 정보는 the section called “Amazon EC2 API 메트릭을 활성화합니다.”을 참조하세요.

Amazon EC2 DescribeInstances API 요청 서버 오류에 대한 경보를 생성하려면

1. https://console.aws.amazon.com/cloudwatch/ 에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 경보(Alarms) 모든 경보(All Alarms)를 선택합니다.
3. Create alarm(경보 생성)을 선택하세요.
4. 지표 선택을 선택하고 다음을 지정합니다.
    a. EC2/API를 선택합니다.
    b. 액션별 지표를 선택합니다.
    c. ServerErrors지표 이름과 같은 행에 DescribeInstances있는 옆의 확인란을 선택합니다.
    d. 지표 선택을 선택하세요.
5. 선택한 지표 및 통계에 대한 그래프와 기타 정보가 표시된 Specify metric and conditions(지표 및 조건 지정) 페이지가 나타납니다.
    a. Metric에서 다음을 지정합니다.
        i. Statistic(통계)에서 Sum(합계)를 선택합니다.
        ii. [기간] 에 5분이 선택되어 있는지 확인합니다.
    b. 조건에서 다음을 지정합니다.

i. 임곗값 유형에서 정적을 선택합니다.

ii. Whenever ServerErrors is의 경우 [크게/같음] >=을 선택합니다.

iii. 보다... , 10을 입력합니다.

c. 다음을 선택합니다.

6. 작업 구성 페이지가 표시됩니다.

- 알림에서 다음을 지정합니다.

i. 알람 상태 트리거에 대해 In alam을 선택합니다.

ii. SNS 주제 선택에서 기존 SNS 주제 선택 또는 새 주제 생성을 선택하고 알림에 필요한 필드를 작성합니다.

iii. 다음을 선택합니다.

7. 이름 및 설명 추가 페이지가 나타납니다.

a. 알람 이름에 알람 이름을 입력합니다. 이름은 ASCII 문자만 포함해야 합니다.

b. 알람 설명에는 알람에 대한 선택적 설명을 입력합니다.

c. 다음을 선택합니다.

8. 미리 보기 및 생성 페이지가 나타납니다. 정보가 정확한지 확인한 다음 알람 만들기를 선택합니다.

자세한 내용은 Amazon 사용 설명서의 Amazon CloudWatch 경보 사용을 참조하십시오. CloudWatch

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.